ระบบร้านค้าอัจฉริยะ
INTELLIGENT SHOP SYSTEM

ธนเดช เกิดศิริ

ธนวัฒน์ โชติถิตดิ

ธนวัฒน์ นิโกรธา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมระบบควบคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2557

ระบบร้านค้าอัจฉริยะ

INTELLIGENT SHOP SYSTEM

ธนเดช เกิดศิริ
ธนวัฒน์ โชติกิตติ
ธนวัฒน์ นิโกรธา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมระบบควบคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

INTELLIGENT SHOP SYSTEM

THANADET KERDSIRI
THANAWAT CHOATKITTI
THANAWAT NIKROTHA

THIS THESIS IS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF ENGINEERING IN CONTROL ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2014

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2557

ภาควิชาวิศวกรรมการวัดและควบคุม คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบร้านค้าอัจฉริยะ
INTELLIGENT SHOP SYSTEM

ผู้จัดทำ

นายธนเดช	เกิดศิริ	54010536
นายธนวัฒน์	โชติกิตติ	54010562
นายธนวัฒน์	นิโกรธา	54010564

.....อาจารย์ที่ปรึกษา
(ดร.รัชนี กุลยานนท์)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบร้านค้าอัจฉริยะ

โดย

นายธนเดช	เกิดศิริ	54010536
นายธนวัฒน์	โชติกิตติ	54010562
นายธนวัฒน์	นิโกรธา	54010564

อาจารย์ที่ปรึกษา

ดร.รัชนี กุลยานนท์

ปีการศึกษา 2557

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้ เป็นการนำเสนอโครงงานการออกแบบและพัฒนาระบบร้านค้าอัจฉริยะ โดยระบบร้านค้าอัจฉริยะนี้เป็นการออกแบบและพัฒนาในขั้นต้น จุดประสงค์หลักคือ เพื่อเพิ่มทางเลือกที่หลากหลายในการเลือกซื้อสินค้าให้แก่ผู้บริโภค โดยผู้ใช้งานจะสามารถเลือกซื้อสินค้าได้ 2 ทางเลือกจากโปรแกรม คือ 1. สั่งด้วยคำสั่งเสียง 2. สั่งผ่านหน้าจอสัมผัส หลักการทำงานของโปรแกรมจะเริ่มจากการสั่งงานผ่านทางคำสั่งเสียงหรือหน้าจอสัมผัส จากนั้นโปรแกรมจะทำการจองจำนวนสินค้าจากคลังสินค้าในแต่ละสาขา โดยใช้วิธีการหักจำนวนสินค้าที่สั่งในแต่ละรายการ ออกจากจำนวนสินค้าคงคลังของแต่ละสาขา พร้อมทั้งบันทึกรายการสั่งซื้อสินค้าและแสดงรายละเอียดที่จำเป็นในการชำระเงินให้แก่ผู้บริโภค

ผลการทดลองสามารถสรุปได้ว่าโปรแกรมระบบร้านค้าอัจฉริยะจะสามารถทำงานได้ดี ในสถานที่ที่มีลักษณะเปิดกว้างมากกว่าในสถานที่ที่มีลักษณะปิด และมีความแม่นยำค่อนข้างสูงเมื่อทดสอบกับการใช้งานในสถานที่จริง และระยะที่ใช้ในการออกคำสั่งเสียงที่มีความแม่นยำมากที่สุดคือระยะ 1 เมตร ซึ่งพบว่าในแต่ละสถานที่ที่ใช้ในการทดลอง ระยะ 1 เมตรเป็นระยะที่โปรแกรมสามารถแสดงค่าที่ถูกต้องและแม่นยำที่สุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

INTELLIGENT SHOP SYSTEM

By

Mr.THANADET KERDSIRI 54010536

Mr.THANAWAT CHOATKITTI 54010562

Mr.THANAWAT NIKROTHA 54010564

Advisor

Dr.Rutchanee Gullayanon

Academic Year 2014

ABSTRACT

This thesis A presentation of the design and development of intelligent stores. The system stores this genius in the design and development stage. The main aim is to increase the variety of goods offered to consumers. The user can then choose to buy two alternative programs ordered by order 1. Party. 2. Order via the touch screen The working principle of the program is started from the command via voice command or touch screen, the program will make a reservation of items from a warehouse in each branch, using less quantity of product on each item. Out of a total inventory of each branch. The record order and the information required to make payment to the consumer.

The results conclude that the application of intelligent shop system will work well. In a place that looks more open than in places that look off. And a relatively high accuracy when tested for use in a real place. And the term used to issue voice commands with more accuracy the distance of 1 meter, which is found in each of the locations used in the Phase 1 meter is programmed to display the accurate and precise as possible.

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จได้ด้วยดี คณะผู้จัดทำขอกราบขอบพระคุณอาจารย์ที่ปรึกษา ดร.รัชนี กุลยานนท์ เป็นอย่างสูงที่คอยให้คำปรึกษาแนะนำวิธีการแก้ปัญหาต่างๆ ในโครงงานทั้งทางทฤษฎี และทางปฏิบัติแก่คณะผู้จัดทำมาโดยตลอด ทำให้ผู้จัดทำเข้าใจถึงที่มาของปัญหาและสามารถแก้ไขปัญหาต่างๆ ได้อย่างถูกวิธี รวมทั้งยังเอื้อเฟื้ออุปกรณ์ที่จำเป็น และความช่วยเหลืออื่นๆ ที่เป็นประโยชน์ต่อโครงงาน

ขอขอบพระคุณ รุ่นพี่และรุ่นน้องภาควิชาวิศวกรรมการวัดและควบคุม ที่ช่วยให้คำปรึกษาและช่วยแก้ไขในส่วนที่คณะผู้จัดทำยังไม่มีความเข้าใจเป็นอย่างดี

ขอขอบพระคุณ อาจารย์สองเมือง นันทขว้าง และทีม ที่ได้เอื้อเฟื้อหุ่นยนต์ ให้แก่คณะผู้จัดทำ

ขอขอบคุณเพื่อนๆ ทุกคน ที่คอยให้กำลังใจสนับสนุนอุปกรณ์ที่ขาดเหลือกระตุ้นเตือนรวมทั้งคอยถามไถ่ความคืบหน้าของโครงงานอยู่เสมอ

สุดท้ายนี้คณะผู้จัดทำขอขอบพระคุณบิดา มารดา และครอบครัว ที่เป็นกำลังใจที่ดีตลอดมา รวมถึงการสนับสนุนในเรื่องของงบประมาณที่ขาดเหลือ ตลอดจนเป็นแรงบันดาลใจที่ทำให้โครงงานนี้สำเร็จสมบูรณ์ได้

ผู้จัดทำ

ธนเดช เกิดศิริ

ธนวัฒน์ โชติกิตติ

ธนวัฒน์ นิโกรธา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้า
บทคัดย่อ	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญรูป	VI
สารบัญตาราง	VIII
บทที่ 1 บทนำ	1
1.1 ที่มาและความสำคัญ	1
1.2 วัตถุประสงค์	1
1.3 ขอบเขตการศึกษาค้นคว้า	1
1.4 ประโยชน์ที่คาดว่าจะได้รับ	2
1.5 ขั้นตอนการดำเนินงานวิจัย	2
1.6 รายละเอียดของปริญญานิพนธ์	3
บทที่ 2 ทฤษฎีพื้นฐาน	4
2.1 Voice Engine	4
2.1.1 PocketSphinx	6
2.1.2 Windows Speech Recognition And Windows Speech Synthesis	7
2.2 การออกแบบ User Interface	9
2.2.1 ภาษาที่มองเห็นได้ (Visible Language)	10
2.2.2 การจัดตั้ง	11
2.2.3 ความเหมาะสม	11
2.2.4 เค้าโครงหน้าจอ	13
2.2.5 ความสัมพันธ์	13
2.2.6 ตำแหน่ง	14
2.2.7 การสื่อสาร	15
2.2.8 การอ่าน	15
2.2.9 การพิมพ์	16
2.2.10 มุมมองในการมองเห็น	16
2.3 ภาษา C#	16
2.4 ภาษา XML	18

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

	หน้า
2.5 ภาษา SQL	18
2.6 โปรแกรม MySQL	19
2.7 โปรแกรม MySql Workbench	20
2.8 โปแกรม phpMyAdmin	21
2.9 โปรแกรม Microsoft Visual Studio 2013	21
2.10 QR Code	22
บทที่ 3 การสร้างและการออกแบบ	24
3.1 หลักการทำงานของโปรแกรมในส่วนของ Robot Application	24
3.2 Voice Engine	27
3.2.1 Windows Speech Recognition	27
3.2.2 Windows Speech Synthesis	28
3.3 Graphical User Interface	31
3.3.1 Central Back Office Application	31
3.3.2 Central Daily Backup	35
3.3.3 Branch Back Office Application	36
3.3.4 Cashier Application	39
3.3.5 Robot Application	40
บทที่ 4 การทดลองและผลการทดลอง	44
4.1 การทดลองสถานที่ห้อง Lab ECC-208	45
4.2 การทดลองสถานที่สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง	46
4.3 การทดลองสถานที่โรงอาหาร A คณะวิศวกรรมศาสตร์	47
4.4 การทดลองสถานที่โรงยิมเนเซียม	49
4.5 การทดลองสถานที่ Top Supermarket	51
บทที่ 5 บทสรุปและวิจารณ์	54
5.1 สรุป	54
5.2 ปัญหาที่พบในโครงงาน	54
5.3 ผลที่คาดว่าจะได้รับในโครงงาน	55
5.4 แนวทางในการพัฒนาและการประยุกต์	55
เอกสารอ้างอิง	56
ภาคผนวก	57

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

รูปที่	หน้า
2.1 Acoustic–Phonetic Approach	4
2.2 Pattern-Recognition Approach	5
2.3 Artificial Intelligence (AI) Approach	6
2.4 ตัวอย่างหน้าจอที่ไม่เป็นระเบียบ	11
2.5 ตัวอย่างหน้าจอที่เป็นระเบียบ	11
2.6 ตัวอย่างชนิดของแต่ละองค์ประกอบที่แสดงในพื้นที่เดียวกัน	12
2.7 ตัวอย่างความสอดคล้องภายนอกสำหรับ Text Tool Icon โดย Icon	12
2.8 ตัวอย่างความสอดคล้องกับความจริง	12
2.9 ตัวอย่างรูปร่าง ตำแหน่งและความสำคัญในการมองเห็นโดยความสัมพันธ์นี้อาจยังไม่เหมาะสม	13
2.10 ตัวอย่างความชัดเจน ความสอดคล้อง ความเหมาะสมและความสัมพันธ์ที่ดี	14
2.11 ตัวอย่างการออกแบบที่แย่	14
2.12 ตัวอย่างการออกแบบที่ดี	15
2.13 ตัวอย่างตัวหนังสือที่มีความชัดเจนและไม่ชัดเจน	15
2.14 ตัวอย่างชนิดของตัวอักษรและรูปแบบของตัวอักษร	16
2.15 ตัวอย่างแบบอักษร และการมองเห็น	16
3.1 หลักการทำงานของโปรแกรมในส่วนของ Robot Application	26
3.2 ส่วนของโครงสร้างฐานข้อมูลของศูนย์กระจายสินค้า	29
3.3 โครงสร้างฐานข้อมูลของสาขาย่อย	30
3.4 Product Access	31
3.5 Create Branch Template	32
3.6 Set Quantity for Template	33
3.7 New Branch	33
3.8 Sell Detail	34
3.9 Central Back Office Configuration	34
3.10 Account Manage	35
3.11 Call for Backup	35
3.12 Branch Account Manage	36
3.13 Branch Back Office Configuration	36
3.14 Sell Detail	37
3.15 Set Quantity in Branch	38
3.16 Stock Watching	38
3.17 Product Check In	39

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป(ต่อ)

รูปที่	หน้า
3.18 Cashier Application	39
3.19 Cashier Application Configuration	40
3.20 Robot Application Configuration	40
3.21 หน้าจอขณะอยู่ใน Normal Mode	41
3.22 หน้าจอขณะอยู่ใน Voice Mode	41
3.23 หน้าจอ Pop Up	42
3.24 หน้าจอแสดงการยืนยันการสั่งซื้อ	42
3.25 หน้าจอ Order Detail	43
4.1 สถานที่ห้อง Lab ECC-208	46
4.2 สถานที่สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง	47
4.3 สถานที่โรงอาหาร A คณะวิศวกรรมศาสตร์	48
4.4 สถานที่โรงยิมเนเซียม	50
4.5 สถานที่ Top Supermarket	52
4.6 สรุปการทดลองในแต่ละระยะของสถานที่การทดลองทั้งหมด	52

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตารางที่ **หน้า**

3.1 ข้อดีข้อเสียของ Windows Speech Recognition 27

3.2 ข้อดีและข้อเสียของ Window Speech Synthesis 28

4.1 สถานที่ห้อง Lab ECC-208 45

4.2 สถานที่สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง 46

4.3 สถานที่โรงอาหาร A คณะวิศวกรรมศาสตร์ 47

4.4 สถานที่โรงยิมเนเซียม 49

4.5 สถานที่ Top Supermarket 51

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญ

ในปัจจุบันช่องทางการเลือกซื้อสินค้าของผู้บริโภคทั้งในส่วนของกลุ่มวัยรุ่น วัยทำงาน ผู้สูงอายุ ผู้พิการหรือบกพร่องทางร่างกายในด้านต่างๆ มีอยู่ไม่มากนัก อาจด้วยข้อจำกัดหลายอย่าง เช่น สถานที่ตั้ง เทคโนโลยี รวมทั้งกลุ่มผู้บริโภคที่มีเพศ และวัยที่แตกต่างกัน จึงทำให้ผู้บริโภคขาดความเท่าเทียมกันในการเลือกซื้อสินค้า เช่น ระบบร้านค้าออนไลน์ ประสบผลสำเร็จแก่กลุ่มผู้ใช้ที่อยู่ในวัยรุ่น และวัยทำงาน มากกว่ากลุ่มของผู้สูงอายุและผู้ที่มีความบกพร่องทางร่างกาย เนื่องจากระบบการเลือกซื้อสินค้าออนไลน์นั้นส่วนใหญ่มีระบบการเลือกซื้อที่สลับซับซ้อน ตัวหนังสือเล็กเกินไป รวมทั้งมีโฆษณาอื่นๆ ที่ไม่เกี่ยวข้องมาก ซึ่งสร้างความสับสนให้กับผู้บริโภคได้ง่าย จะเห็นได้ว่าระบบดังกล่าวยังต้องพัฒนาในส่วนของส่วนติดต่อผู้ใช้ให้มีความเรียบง่ายมากที่สุด โดยยังคงฟังก์ชันการทำงานไว้อย่างครบถ้วน ดังนั้น “ระบบร้านค้าอัจฉริยะ” นี้เองที่จะช่วยแก้ปัญหาดังกล่าวอีกทั้งลดความซับซ้อนในการเข้าถึงฟังก์ชันการใช้งานต่างๆ อันเกิดจากการออกแบบส่วนต่อประสานงานของผู้ใช้ที่ไม่เหมาะสมต่อกลุ่มเป้าหมาย ทำให้ผู้ใช้ไม่สามารถใช้ประโยชน์จากโปรแกรมได้อย่างเต็มประสิทธิภาพ เพื่อช่วยลดระยะเวลาในการเข้าถึงสินค้า ตลอดจนสิ้นสุดขั้นตอนการเลือกซื้อสินค้าได้อย่างง่ายดายและรวดเร็ว

1.2 วัตถุประสงค์

เพื่อเป็นการเพิ่มทางเลือกในการเลือกซื้อสินค้าทั้งวัยเด็ก วัยทำงาน ผู้สูงอายุ หรือแม้กระทั่งผู้ที่มีความบกพร่องทางด้านร่างกายในด้านต่างๆ ให้สามารถเลือกซื้อสินค้าได้อย่างสะดวกและรวดเร็วมากยิ่งขึ้น โดยเข้าถึงข้อมูลต่างๆ ของสินค้าผ่านระบบการสั่งงานด้วยจอ Touch Screen หรือการเข้าถึงข้อมูลโดยผ่านการสั่งงานด้วยเสียง และยังช่วยประหยัดในเรื่องของเวลาในการเลือกซื้อสินค้าประเภทต่างๆ

1.3 ขอบเขตการศึกษาค้นคว้า

ระบบการสั่งงานด้วยเสียงพัฒนาด้วย .Net (Netframework 4.5) Speech ดังนั้นจึงสามารถทำงานได้เฉพาะบนระบบปฏิบัติการ Windows ระบบ User Interface ของโปรแกรมพัฒนาได้บนภาษา C# และใช้ MySQL ในการเก็บข้อมูลต่างๆ เช่น รายละเอียดสินค้า รายละเอียดการขาย เป็นต้น

ผู้ใช้สามารถนำระบบร้านค้าอัจฉริยะไปพัฒนาให้สามรถทำงานได้อย่างมีประสิทธิภาพมากขึ้น ในอนาคตโดยการพัฒนาเสียงให้สามรถรองรับการใช้งานด้วยภาษาไทย ตกแต่งให้ระบบ User Interface มีความเหมาะสมกับผู้ใช้ในแต่ละวัยเพื่อเพิ่มความหน้าสนใจในการใช้งานและยังสามารถเพิ่มฐานข้อมูลให้มีความครอบคลุมมากยิ่งขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

1.4 ประโยชน์ที่คาดว่าจะได้รับ

ผู้บริโภคสามารถสั่งซื้อสินค้าเป้าหมายที่ต้องการได้ โดยการใช้งานใน Mode Touchscreen และ Voice Mode ในการเลือกซื้อสินค้า ซึ่งทำให้ประหยัดระยะเวลาในการสั่งซื้อสินค้าลงได้ ช่วยเหลือผู้สูงอายุ หรือผู้บกพร่องทางด้านร่างกายที่ต้องการซื้อสินค้าประเภทต่างๆ และทำให้กลุ่มเป้าหมายเหล่านี้สามารถช่วยเหลือตัวเองได้อย่างสะดวก โดยเลือกการใช้งานใน Mode ใด Mode หนึ่งภายในระบบ

1.5 ขั้นตอนการดำเนินงานวิจัย

1. ศึกษาทฤษฎีพื้นฐานในส่วนของเสียง Voice Engine ได้แก่ PocketSphinx, Windows Speech Recognition, Windows Speech Synthesis เป็นต้น

2. ศึกษาทฤษฎีพื้นฐานของการออกแบบ User Interface และการใช้งานภาษา C# ในการออกแบบหน้าแสดงผล

3. ศึกษาทฤษฎีพื้นฐานของการออกแบบระบบฐานข้อมูลต่างๆ ได้แก่ XML, SQL, MySQL, MySQL Workbench, phpMyAdmin, Microsoft Visual Studio 2013, QR Code เป็นต้น

4. ออกแบบและสร้างระบบโดยเป็นส่วนของการใช้งานด้วยเสียง โดยใช้ Speech Synthesis

5. ออกแบบและสร้างระบบโดยเป็นส่วนของการใช้งานด้วยเสียง โดยใช้ Speech Recognition

6. ออกแบบและสร้างระบบโดยเป็นส่วนของการใช้งานด้วยหน้าจอสัมผัส Touchscreen Mode

7. ออกแบบและสร้างฐานข้อมูลของระบบโดยแบ่งเป็นส่วนของศูนย์กระจายสินค้าและสาขาย่อย

8. ทดสอบการใช้งานด้วยระบบการใช้คำสั่งเสียงในสถานที่ต่างๆ โดยใช้ระยะการทดลองที่ต่างกัน

9. ตรวจสอบการแสดงผลใน Touchscreen Mode ว่าสามารถแสดงผลของข้อมูลได้ถูกต้องและมีความคลาดเคลื่อนมาดเพียงใด

10. ทำการรวบรวมการออกแบบของระบบในส่วนต่างๆ เข้าด้วยกัน

11. ทำการตรวจสอบประสิทธิภาพการใช้งานและความถูกต้องของการใช้งานใน Voice Mode และ Touchscreen Mode ในการสั่งซื้อสินค้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.6 รายละเอียดของปริญญานิพนธ์

ในปริญญานิพนธ์ฉบับนี้ประกอบไปด้วย 5 บท โดยมีรายละเอียดของแต่ละบทดังต่อไปนี้

บทที่ 1 จะกล่าวถึงที่มาและความสำคัญในการทำโครงงาน วัตถุประสงค์ของโครงงาน ขอบเขตของโครงงาน และประโยชน์ที่ได้รับของการทำโครงงาน

บทที่ 2 จะอธิบายเกี่ยวกับทฤษฎีพื้นฐานต่างๆ ที่นำมาใช้เป็นส่วนประกอบของโครงงาน ทั้งในเรื่องของ Voice และ Database

บทที่ 3 จะกล่าวถึงการสร้างและการออกแบบส่วนประกอบต่างๆ ที่อยู่ภายในระบบร้านค้าอัจฉริยะและภาพรวมของระบบ

บทที่ 4 จะเป็นส่วนที่ได้จากการทดลอง และวิเคราะห์ข้อมูลในส่วนของการใช้งานระบบการสั่งงานด้วยคำสั่งเสียงในแต่ละสถานที่ และระยะการสั่งงานด้วยเสียงในระยะต่างๆ และผลการทดลอง

บทที่ 5 เป็นบทสุดท้าย จะสรุปรวมและวิเคราะห์ปัญหาที่พบ ตลอดจนแนวทางในการแก้ไขต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

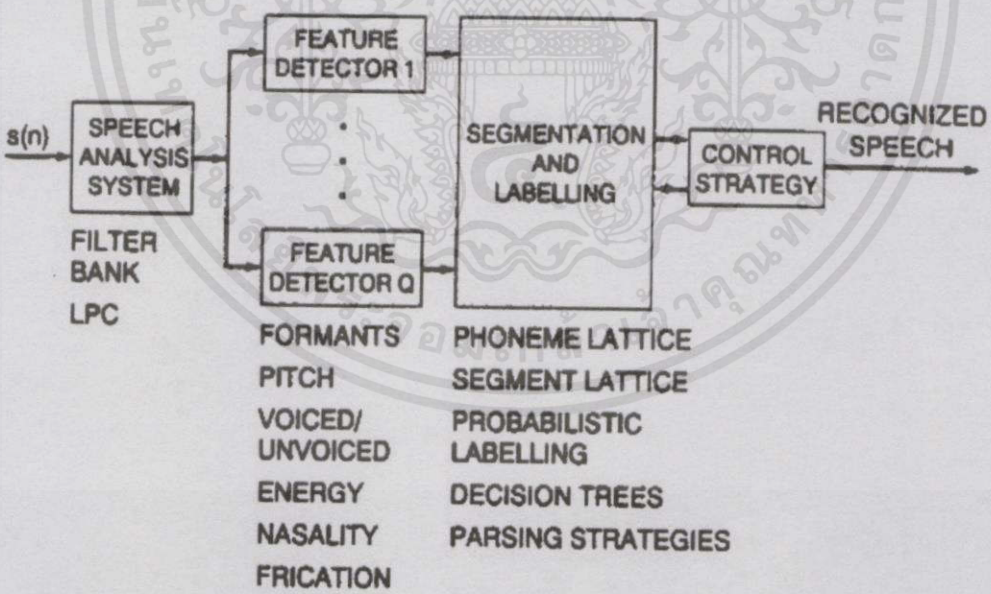
บทที่ 2

ทฤษฎีพื้นฐาน

เนื่องจากระบบร้านค้าอัจฉริยะดังกล่าวจะประกอบไปด้วยส่วนสำคัญหลักๆ 2 ส่วนคือ ในส่วนของ Voice และในส่วนของ Database ในการสร้างและออกแบบในส่วนต่างๆ ของระบบ ในบทนี้ จึงจะกล่าวถึงทฤษฎีที่ปัจจุบันมีใช้สำหรับการสร้าง Voice และ Database ซึ่งในการสร้างส่วนต่างๆ จำเป็นต้องศึกษาทฤษฎีที่เกี่ยวข้องและต้องอาศัยการศึกษาเพื่อนำมาพัฒนาส่วนต่างๆ ภายในระบบ ประกอบด้วย Voice Engine, ทฤษฎีการออกแบบ User Interface, ภาษา C#, ภาษา XML, ภาษา SQL, โปรแกรม MySQL, โปรแกรม MySQL Workbench, โปรแกรม phpMyAdmin, โปรแกรม Microsoft Visual Studio 2013, QR Code เป็นต้น

2.1 Voice Engine

แนวคิดเกี่ยวกับการสร้างเครื่องมือในการรู้จำเสียงนั้น ได้เริ่มต้นขึ้นครั้งแรกในปี ค.ศ. 1950 เป็นต้นมา โดยอาศัยความรู้ทางด้านคณิตศาสตร์ คอมพิวเตอร์ และวิศวกรรม จนได้แนวคิดในการจัดจำแนกเสียงพูดต่างๆ ออกมาเป็นข้อความได้ ซึ่งในปัจจุบันเทคนิคในการวิเคราะห์เสียงพูดเพื่อให้ได้มาซึ่งข้อความของคำพูดนั้นๆ ประกอบไปด้วย 3 หลักการคือ หลักการ Acoustic-Phonetic Approach, Pattern-Recognition Approach และ Artificial-Intelligent Approach

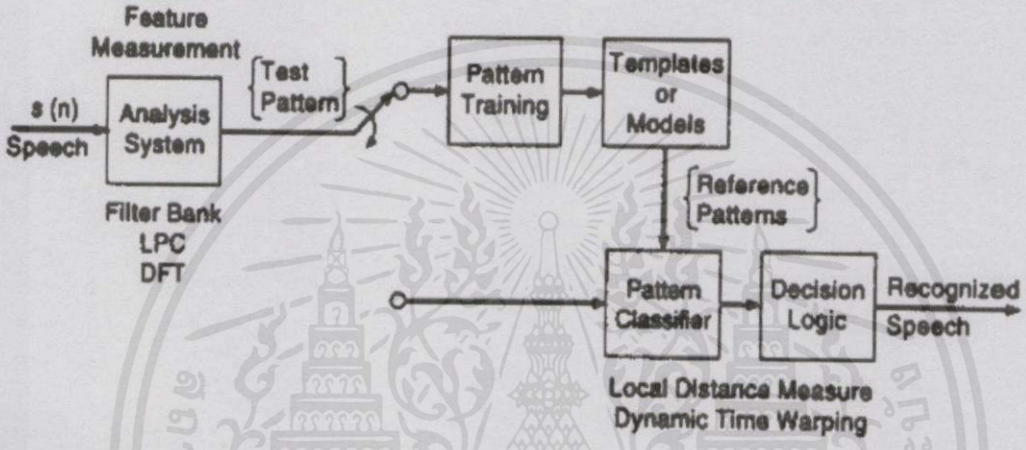


รูปที่ 2.1 Acoustic- Phonetic Approach

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

Acoustic-Phonetic Approach จะรับเสียงพูด (Voice) เข้ามาเป็นข้อมูลขาเข้า จากนั้นจะเป็นในส่วนของการแปลงสัญญาณเสียงดังกล่าวให้อยู่ในรูปของสเปคตรัม ซึ่งเป็นหน้าที่ของส่วน Speech Analysis System จากนั้นจะได้สัญญาณเสียงในรูปของสเปคตรัม ซึ่งจะใช้เป็นสัญญาณ

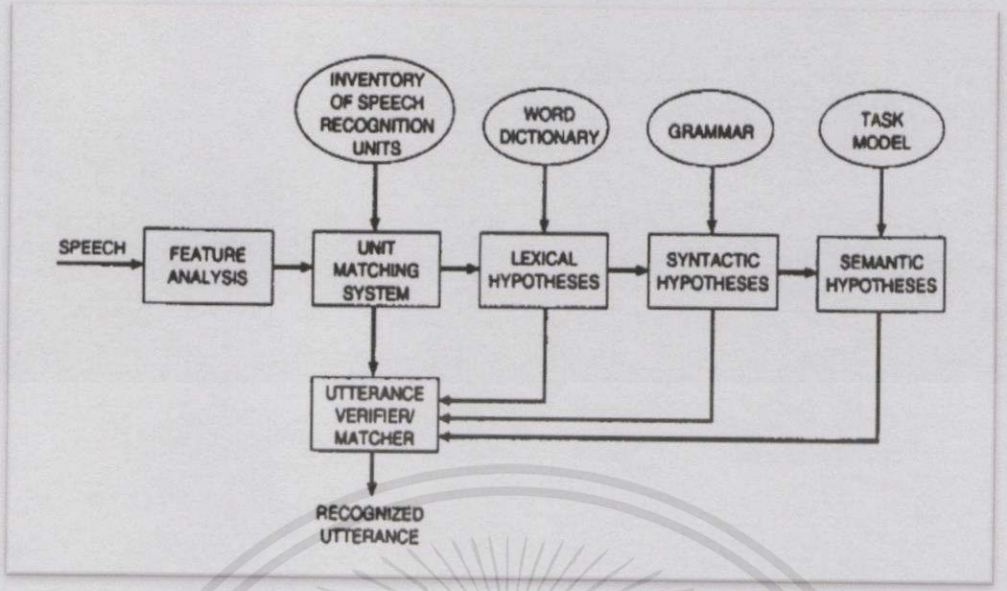
อินพุตแก่ส่วน Feature Detector ซึ่งในส่วนนี้จะมีหน้าที่ให้การตรวจหาคุณสมบัติต่างๆ ของเสียง เช่น Formants (คุณสมบัติการเรโซแนนซ์ของเสียง) และ Nasality (เป็นเสียงที่ได้จากโพรงจมูก) เป็นต้น จากนั้นจะเป็นการนำงานในส่วนของ Segmentation and labelling ในส่วนนี้จะนำข้อมูลเสียงทั้งหมดที่ได้จากส่วนก่อนหน้านี้ (Feature Detector) มาวิเคราะห์ให้ผลออกมาเป็นลำดับการออกเสียง (Phonetic Sequent) และท้ายที่สุดในส่วนของ Control Strategy จะทำการเลือกลำดับการออกเสียงที่ดีที่สุด โดยพิจารณาจากฐานข้อมูลคำศัพท์ที่มีอยู่ นอกจากนี้ในการตรวจจับประโยคจะใช้ข้อมูลการเรียงลำดับคำศัพท์ที่เป็นไปตามโครงสร้างทางภาษาอีกด้วย เพื่อให้ได้ประโยคที่ถูกต้องและสมบูรณ์ที่สุดดังแสดงในรูปที่ 2.1



รูปที่ 2.2 Pattern-Recognition Approach

Pattern Recognition Approach เป็นกระบวนการรู้จำเสียงอีกวิธีหนึ่งที่ใช้กันในปัจจุบัน โดยขั้นตอนการทำงานของ Pattern-Recognition Approach เริ่มต้นที่รับค่าสัญญาณอินพุตเป็นสัญญาณเสียงเข้าสู่โปรแกรม จากนั้นในส่วนของ Analysis System จะทำการเปลี่ยนสัญญาณเสียงให้อยู่ ในรูปสเปคตรัม ซึ่งจะถูกส่งต่อให้ในส่วนของ Pattern Training ในกรณีที่ต้องการเพิ่มคำศัพท์ใหม่ที่จะใช้ในการตรวจจับเพื่อสร้างแม่แบบเสียง โดยใช้หลักการทางคณิตศาสตร์และสถิติโดยแม่แบบที่ได้จะเรียกว่า “แม่แบบอ้างอิง (Reference Pattern)” และในกรณีที่เสียงที่ใช้ในการตรวจสอบมีรูปแบบที่สร้างไว้แล้วในขั้นตอนก่อนหน้านี้ (Pattern Training) สเปคตรัมจะถูกส่งให้ในส่วนของ Pattern Classifier เพื่อใช้ในการจัดจำแนกเสียงอ้างอิงจากแม่แบบอ้างอิง (Reference Pattern) และสุดท้ายแล้วโปรแกรมจะทำการเลือกค่ารูปแบบเสียงที่ใกล้เคียงแม่แบบที่สุด ซึ่งกระบวนการตัดสินใจนี้จะเกิดขึ้นในส่วนของ Decision Logic ดังแสดงในรูปที่ 2.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.3 Artificial Intelligence (AI) Approach

Artificial Intelligence (AI) Approach เป็นกระบวนการรู้จำเสียงที่สร้างขึ้นบนองค์ความรู้จากหลากหลายแขนง เช่น ความรู้ด้านการออกเสียง (Phonetic), ความรู้ด้านความหมายของคำและการสร้างประโยค (Sematic and Language Model) เป็นต้น ซึ่งการนำความรู้ในหลากหลายแขนงมาสร้างเป็นกระบวนการรู้จำเสียงแบบ Artificial Intelligence นั้นช่วยให้ข้อความที่ตรวจจับได้มีความแม่นยำและใช้งานได้กว้างขวางยิ่งขึ้น ซึ่งการตรวจจับและวิเคราะห์เสียงด้วยวิธีดังกล่าวสามารถลดค่าความคลาดเคลื่อนของการตรวจจับลงได้แต่มีความซับซ้อนอย่างมากในการสร้างเครื่องมือตรวจจับขึ้น ดังแสดงในรูปที่ 2.3

ในการศึกษาทฤษฎีของเสียงนี้จึงได้เลือกศึกษาโปรแกรมที่ได้รับความนิยมในการศึกษาและมีประสิทธิภาพในการทำงาน โดยโปรแกรมที่ใช้ในการศึกษาและนำมาใช้สร้างในส่วนของเสียงนี้จะประกอบไปด้วย 2 โปรแกรมหลักคือ PocketSphinx และ Windows Speech Recognition And Windows Speech Synthesis

2.1.1 PocketSphinx

PocketSphinx Speech Recognition Engine ถูกพัฒนาโดย Carnegie Mellon University มีความยืดหยุ่นแก่นักพัฒนา Speech Recognition Engine เพราะนอกจาก PocketSphinx จะเปิดโอกาสให้ได้ใช้งานกันแบบฟรีแล้ว ตัว PocketSphinx ยังสามารถทำงานได้ทั้ง Windows OS, Linux OS หรือแม้แต่ Windows Phone และมีฐานข้อมูลทางภาษาให้เลือกใช้อยู่แล้ว เช่น ภาษาอังกฤษ ซึ่งการใช้งานโดยทั่วไปสามารถทำได้เพียงแค่ติดตั้ง และเซ็ตค่าต่างๆ ตามคู่มือเท่านั้น

หลักการทำงานของโปรแกรมคือ โปรแกรมจะรับไฟล์เสียงผ่านทาง Input Device เช่น Mic จากนั้นไฟล์เสียงจะถูกวิเคราะห์โดยโปรแกรม PocketSphinx ให้อยู่ในรูปของชุดตัวเลขจำนวน 39 ตัวเลข / 10 มิลลิวินาที โดยเรียกชุดตัวเลขดังกล่าวนี้ว่า “Feature Vector” ซึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Feature Vector สามารถหาได้จากสเปคตรัมของชุดเสียงที่รับเข้ามาผ่าน Input Device ซึ่ง PocketSphix จะใช้เสียงเงียบในการแยกคำต่างๆ ออกจากประโยค จากนั้นโปรแกรม PockSphinx จะทำการเปรียบเทียบค่า Feature Vector ของเสียงที่รับเข้ามากับฐานข้อมูล Feature Vector ของคำต่างๆ ว่าตรงกับคำไหนในฐานข้อมูลโดยใช้การจับคู่ที่มีค่า Error น้อยที่สุด และทำการแสดงคำที่ตรงกันออกทางจอมอนิเตอร์ จะเห็นว่าในการตรวจจับและวิเคราะห์เสียงจะเป็นหน้าที่ของโปรแกรม PocketSphinx แต่ทว่า PocketSphix จะไม่สามารถทำงานได้เลย หากไม่มีฐานข้อมูลของคำต่างๆ ซึ่งส่วนการจัดการฐานข้อมูลของคำและเสียงต่างๆ นั้น จะใช้โปรแกรม Sphinxbase ในการจัดการ โดยภายใน Sphinxbase จะเก็บคุณสมบัติต่างๆ ของคำต่างๆ ที่ในการอ้างอิงของ PocketSphinx โดยเราจะเรียกฐานข้อมูลนี้ว่า “Model” ซึ่งตัว Model จะประกอบไปด้วย 3 ส่วน คือ

1. Acoustic Model จะทำการเก็บรวบรวมค่า Feature Vector ของคำต่างๆ ที่จะใช้ในการสร้างฐานข้อมูลอ้างอิง

2. Phonetic Dictionary จะเป็นการ Mapping คำให้เข้ากับเสียงที่มีอยู่ในฐานข้อมูล ซึ่งเป็นการบอกว่าคำนี้ออกเสียงอย่างไร เช่น

Bird = B ER D

Bat = B AE T

3. Language Model ในส่วนสุดท้ายของการสร้าง Model คือ การสร้าง Language Model ซึ่ง Language Model จะเก็บข้อมูลการ Mapping คำต่างๆ ให้เป็นประโยค เพื่อใช้ในการหาโอกาสการตรวจจับคำต่อไป ซึ่งจะเรียงลำดับเป็น Sequence เช่น

My name is Richard = My + Name + Is + Richard

และ

My + *Name* + *Is* + *Richard* ≠ My + *Is* + *Richard* + *Name*

2.1.2 Windows Speech Recognition And Windows Speech Synthesis

Windows Speech Recognition เป็นโปรแกรมที่ใช้ในการตรวจจับ และสั่งงานด้วยเสียง ซึ่งมีรวมอยู่ใน OS ของ Microsoft ตั้งแต่ Windows Vista เป็นต้นมา ดังนั้นผู้ใช้งาน OS ของ Microsoft ก็สามารถที่จะใช้งานโปรแกรม Windows Speech Recognition กันได้แบบฟรี เช่นเดียวกับ PocketSphinx โดยนักพัฒนาโปรแกรมสามารถเรียกใช้งาน Windows Speech Recognition ผ่าน Namespace System.Speech.Recognition นอกจากนี้ Microsoft ยังมีโปรแกรมที่ให้สังเคราะห์เสียงจาก Text ได้อีกด้วย โดยเรียกใช้งานผ่านทาง Namespace System.Speech.Synthesis ซึ่งสามารถพัฒนาโปรแกรมได้จากภาษา C# นอกจากนี้ผู้ใช้งาน Windows ยังสามารถเรียกใช้งาน Windows Speech Recognition และ Windows Speech Synthesis จากหน้าต่าง Control Panel ได้เลย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

หลักการทำงานของ Windows Speech Recognition จะสามารถแบ่งได้เป็น 2 ส่วนหลักๆ คือ ส่วนรับและแปลงสัญญาณเสียง และส่วนจับคู่ข้อมูลเสียง โดยเมื่อผู้ใช้เปล่งเสียงพูด

ออกมา 1 คำ หรือ 1 ประโยค Computer จะสามารถรับสัญญาณเสียงได้จาก Input Device (Microphone) จากนั้นโปรแกรมจะทำการแปลงข้อมูลเสียงที่รับเข้ามาให้อยู่ในรูปของชุดตัวเลข ซึ่งชุดตัวเลขหนึ่งๆ จะเฉพาะเจาะจงกับเสียงแต่ละชุด หลังจากนั้นจะเป็นการทำงานของโปรแกรมในส่วนที่สองคือ ส่วนจับคู่ข้อมูลเสียง โดยโปรแกรมในส่วนนี้จะทำการจับคู่ข้อมูลเสียงที่ได้จากส่วนแรกของโปรแกรม กับฐานข้อมูลเสียงทั้ง 3 ฐานข้อมูลของ Windows Speech Program ซึ่งประกอบไปด้วย

1. Acoustic Model Database เป็นฐานข้อมูลที่เก็บรวบรวมข้อมูลเสียงต่างๆของคำๆหนึ่ง และมีความจำเพาะไปตามแต่ภาษา และแต่ละบุคคล

2. Lexicon Database หรือฐานข้อมูลคำ ซึ่งจะเก็บรวบรวมวิธีการออกเสียงของคำต่างๆไว้ในฐานข้อมูลนี้

3. Language Model Database เป็นฐานข้อมูลที่เก็บรวบรวมประโยคต่างๆ ที่ได้จากการจัดเรียงคำหลายๆ ซึ่งทำให้โปรแกรมสามารถคาดการณ์ได้ว่าคำต่อไปควรจะเป็นคำไหน

ในส่วนการทำงานของ Windows Speech Program นั้นนอกจากจะสามารถใช้ในการตรวจจับบทความสนทนาแล้วแสดงผลเป็นข้อความได้แล้ว (Windows Speech Recognition) ยังสามารถสังเคราะห์เสียงพูดจากข้อความได้โดยผ่านโปรแกรม Windows Speech Synthesis หรืออาจเรียกใช้ผ่าน Namespace System.Speech.Synthesis ของภาษา C# ซึ่งหลักการทำงานของ Windows Speech Synthesis จะสามารถแบ่งได้แบ่ง 2 ส่วน คือ ส่วนวิเคราะห์ข้อความ (Text Analysis) และส่วนสังเคราะห์เสียง (Sound Generation) ในส่วนของการวิเคราะห์ข้อความจะรับข้อมูลที่เป็นคำ วลี หรือประโยค จากนั้นตัวโปรแกรมจะทำการวิเคราะห์คำต่างๆ ในประโยคนั้นๆ ว่าเป็นคำนาม คำเชื่อม คำกริยา ฯลฯ (Part of Speech) และวิเคราะห์กาลของประโยค (Tense) รวมทั้งประเภทของประโยคด้วย เช่น ประโยคคำถาม ประโยคคำสั่ง จากนั้นจะส่งค่าต่างๆ ออกสู่โปรแกรมในส่วนที่สองคือ ส่วนสังเคราะห์เสียง โดยในส่วนสังเคราะห์เสียงนี้จะสร้างระดับเสียงสูง ต่ำให้สอดคล้องตามรูปแบบของคำและประโยคที่รับเข้ามา เช่น ในภาษาอังกฤษการออกเสียงในประโยคคำถาม จะมีการเน้นเสียงให้สูงขึ้นในส่วนท้ายของประโยค เป็นต้น

เมื่อศึกษาทฤษฎีในส่วนของการสร้างเสียงเบื้องต้นแล้ว ต่อมาคือการศึกษาในส่วนของการสร้างหน้าจอแสดงผลจึงต้องอาศัยการศึกษาทฤษฎีเบื้องต้นของการสร้าง User Interface เพื่อทำให้การแสดงผลของหน้าจอแสดงผลมีความน่าสนใจ มีความเป็นระเบียบในการแสดงผล และใช้งานได้อย่างมีประสิทธิภาพมากที่สุดจึงทำการเริ่มต้นศึกษาดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 การออกแบบ User Interface

การใช้การพิมพ์, สัญลักษณ์, สี และอื่นๆ โดยส่วนมากจะใช้ถ่ายทอดความเป็นจริง ทั้งในด้านของแนวคิดและอารมณ์ โดยการออกแบบ Graphic อย่างเป็นระบบนั้นจะทำให้ผู้ใช้เข้าใจถึงข้อมูลที่ซับซ้อน โดยการสื่อสารด้วยภาพที่ประสบความสำเร็จนั้นจะต้องมุ่งเน้นถึงข้อมูลโครงสร้างที่สำคัญของการออกแบบระบบ Graphic เป็นสิ่งสำคัญ

สำหรับการออกแบบ User Interface ที่ประสบความสำเร็จนั้น จะแบ่งปัจจัยที่สำคัญได้ 2 ปัจจัยคือ

1. ปัจจัยการพัฒนา ช่วยปรับปรุงการสื่อสารด้วยภาพ โดยมีชุดเครื่องมือและส่วนประกอบของ Libraries เพื่อช่วยสร้างหรือปรับปรุงการสร้างต้นแบบ

2. ปัจจัยการมองเห็นหรือปัจจัยการตอบสนอง แสดงเอกลักษณ์ของผลิตภัณฑ์, รูปแบบความคิดที่ชัดเจนและการนำเสนอสิ่งต่างๆ

การออกแบบควรจะรวบรวมขึ้นเพื่อทำการเชื่อมต่ออย่างมีเหตุผล และมีความสำคัญโดยทางเลือกที่มีประโยชน์ควรตั้งอยู่ในวามชัดเจน และแบบจำลองนั้นควรจะสอดคล้องกันอย่างชัดเจนและสามารถจดจำผู้ใช้งานได้ หรือควรเกี่ยวข้องกับการรวมสิ่งของที่ใช้ร่วมกัน และแยกสิ่งที่ไม่เกี่ยวข้องออกจากกัน หลักการที่เกี่ยวข้องกับ User Interface ที่สำคัญมีดังนี้

1. หลักการความเรียบง่าย (The Simplicity Principle) การออกแบบควรทำให้ง่ายต่อการใช้งาน โดยการติดต่อสื่อสารกันควรมีความชัดเจนและง่ายในการใช้ภาษาของตัวเอง ที่สำคัญอย่างหนึ่งคือ การใช้ทางลัดที่ดีในการให้ความหมายเกี่ยวข้องกับขั้นตอนต่อไป

2. หลักการในการมองเห็น (The Visibility Principle) การออกแบบควรทำให้ตัวเลือกที่จำเป็นทั้งหมด และ วัสดุของงานที่กำหนดนั้น สามมารถมองเห็นได้โดยไม่รบกวนผู้ใช้ภายนอกหรือมีข้อมูลที่ซ้ำซ้อน โดยการออกแบบที่ดีไม่ควรครอบงำทางเลือกของผู้ใช้ หรือสับสนกับข้อมูลที่ไม่จำเป็น

3. หลักการตอบกลับ (The Feedback Principle) การออกแบบควรจะทำให้ผู้ใช้ทราบตัวการกระทำหรือการตีความ การเปลี่ยนแปลงสภาวะและข้อผิดพลาดหรือข้อยกเว้นที่เกี่ยวข้องและเป็นที่สนใจอย่างชัดเจนโดยมีความกระชับและใช้ภาษาที่คุ้นเคยกับผู้ใช้

4. หลักการความยืดหยุ่น (The Tolerance Principle) การออกแบบควรมีความยืดหยุ่นและลดค่าความผิดพลาดและโดยอนุญาตให้มีการแก้ไขและทำใหม่อีกครั้ง

5. หลักการนำไปใช้ (The Reuse Principle) การออกแบบควรนำมาใช้กับส่วนประกอบภายในและส่วนประกอบภายนอก มีความสอดคล้องกับวัตถุประสงค์มากกว่าการใช้ที่ไม่เหมาะสมเพื่อช่วยลดความต้องการสำหรับผู้ใช้ในการคิดใหม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากหนังสือของ Jef Raskin ชื่อ The Humane Interface มีกฎ 2 ข้อในการออกแบบ User Interface[https://books.google.co.th/books?id=D39vjmLfO3kC&printsec=frontcover&hl=th&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false]

กฎข้อแรก : คอมพิวเตอร์จะไม่เป็นอันตรายต่อการทำงานของคุณ มีการแบ่งสรรงานของคุณก่อนที่จะก่อให้เกิดอันตราย

กฎข้อสอง : คอมพิวเตอร์จะไม่ทำให้คุณเสียเวลา หรือทำให้คุณทำงานมากขึ้น หรือ มากกว่าความจำเป็นอย่างมาก

โดย Jef Raskin กล่าวไว้ว่า ผู้ใช้ควรตั้งค่าความก้าวหน้าของการมีปฏิสัมพันธ์ ซึ่งหมายความว่า ผู้ใช้ควรเข้าถึงข้อมูลได้ทันที

ในหัวข้อนี้จะแสดงให้เห็นถึงการออกแบบในส่วนต่างๆ ว่าควรออกแบบในส่วนของหน้าจอ User Interface อย่างไรให้มีความเหมาะสมกับการใช้งาน และทำให้ผู้ใช้เข้าถึงหรือใช้งานในส่วนของระบบได้อย่างมีประสิทธิภาพและง่ายมากขึ้น โดยจะพูดถึงหลักการ ได้แก่ ภาษาที่มองเห็นได้ (Visible Language), การจัดตั้ง, ความเหมาะสม, เค้าโครงหน้าจอ, ความสัมพันธ์, ตำแหน่ง, การสื่อสาร, การอ่าน, การพิมพ์, มุมมองในการมองเห็น

2.2.1 ภาษาที่มองเห็นได้ (Visible Language)

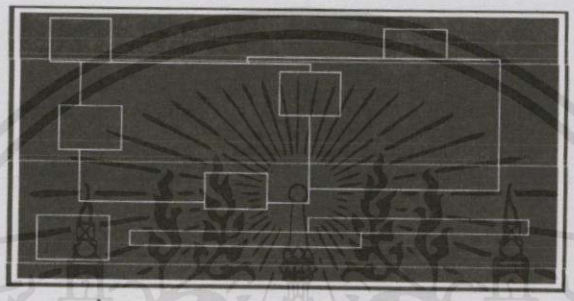
โดยจะกล่าวถึงเทคนิคทั้งหมดของ Graphic ที่ใช้ในการสื่อสารแบบข้อความหรือสิ่งแวดล้อม ประกอบด้วย [http://visiblelanguagejournal.com]

- โครงสร้าง : ประกอบด้วยการจัดรูปแบบการจัดสัดส่วนและการสร้างตารางต่างๆ ของหน้าจอแสดงผลเพื่อง่ายในการเข้าถึงของผู้ใช้
- การพิมพ์ : การเลือกรูปแบบอักษรและการเรียงพิมพ์ ประกอบด้วย ขนาดของตัวแปร และค่าคงที่ต่างๆ ควรเลือกให้เหมาะสมกับโครงสร้างหน้าจอที่ทำการออกแบบ
- สีและเนื้อหา : สี, เนื้อหา และแสง เพื่อการสื่อสารข้อมูลที่ซับซ้อน และความชัดเจนของรูปภาพที่แสดง เพื่อให้ผู้ใช้สามารถเข้าถึงข้อมูลได้อย่างสะดวกที่สุด
- ภาพ : สัญลักษณ์ และเครื่องหมายจากรูปภาพจริง โดยสัญลักษณ์หรือรูปภาพต่างๆ ที่แสดงนั้นควรเป็นภาพที่ผู้ใช้สามารถเข้าใจโดยง่ายว่าต้องการจะสื่อถึงอะไร
- การสร้างภาพเคลื่อนไหว : การเคลื่อนไหวหน้าจอ (สำคัญมากสำหรับภาพหรือวีดีโอที่เกี่ยวข้อง) เพื่อแสดงถึงความน่าสนใจของข้อมูลต่างๆ และแสดงรายละเอียดเพิ่มเติมของข้อมูลที่มีความเกี่ยวข้องและต้องการอธิบายเพิ่มเติม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.2 การจัดตั้ง

คำนึงถึงความเหมาะสม รูปแบบหน้าจอ ความสัมพันธ์และตำแหน่งของข้อมูลต่างๆ เพื่อทำให้ผู้ใช้สามารถเข้าถึงข้อมูลได้อย่างสะดวก โดยการจัดรูปแบบหน้าจอให้เป็นสัดส่วนเพื่อแสดงลำดับความสำคัญของข้อมูล ซึ่งการจัดตั้งหรือเรียงลำดับข้อมูลนั้นควรคำนึงถึงผู้ใช้ที่ไม่มีความรู้เกี่ยวกับระบบหรือสิ่งที่ต้องการแสดงเป็นสำคัญ โดยออกแบบให้มีความง่ายและสะดวกในการใช้งานมากที่สุด ซึ่งตัวอย่างหน้าจอที่ไม่เป็นระเบียบดังรูปที่ 2.4 อาจทำให้การเข้าถึงข้อมูลของผู้เริ่มต้นใช้งานเป็นไปด้วยความยากลำบากเพราะการแสดงผลของข้อมูลไม่มีความเป็นระเบียบ โดยรูปแบบการวางตำแหน่งที่มีการใช้งานได้ง่ายจึงเป็นปัจจัยสำคัญในการออกแบบเพื่อการเข้าถึงข้อมูลได้สะดวกดังรูปที่ 2.5



รูปที่ 2.4 ตัวอย่างหน้าจอที่ไม่เป็นระเบียบ

รูปที่ 2.5 ตัวอย่างหน้าจอที่เป็นระเบียบ

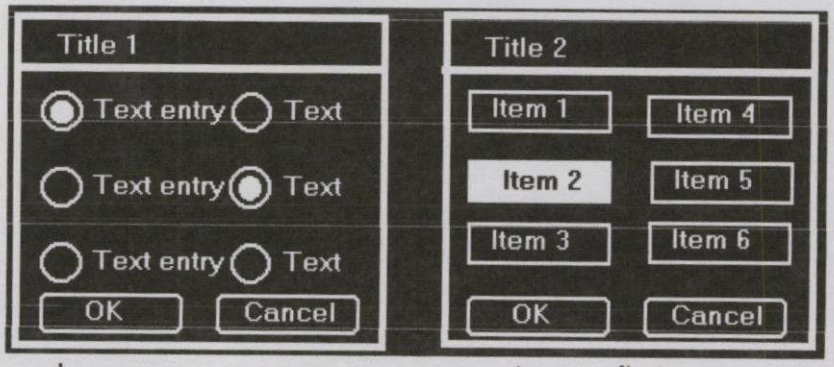
2.2.3 ความเหมาะสม

แบ่งได้เป็น 4 อย่างคือ ความสอดคล้องภายใน, ความสอดคล้องภายนอก, ความสอดคล้องกับจริง และเมื่อไม่มีความสอดคล้อง

1. ความสอดคล้องภายใน

การออกแบบให้เกิดความสอดคล้องภายในนั้นคือ การนำกลุ่มของข้อมูลที่เหมือนกันมาประกอบเข้าด้วยกันเพื่อความง่ายในการเข้าถึง โดยนำข้อมูลที่เป็นประเภทเดียวกันหรือหมวดหมู่เดียวกันมาไว้ในส่วนเดียวกัน ดังรูปที่ 2.6

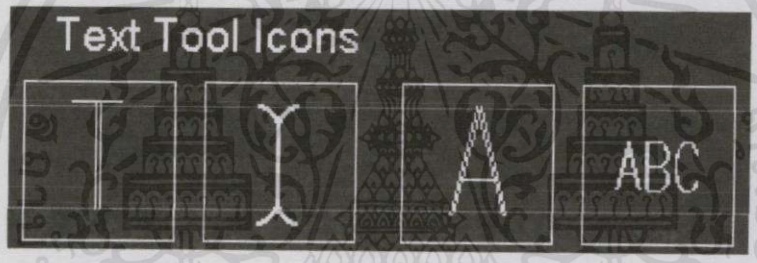
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.6 ตัวอย่างชนิดของแต่ละองค์ประกอบที่แสดงในพื้นที่เดียวกัน

2. ความสอดคล้องภายนอก

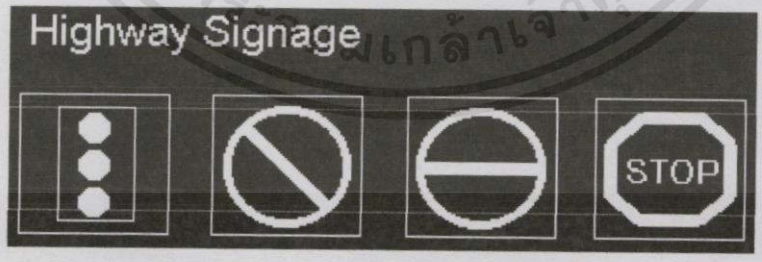
จะออกแบบโดยใช้ตัวอักษรที่ได้รับการยอมรับหรือคนส่วนใหญ่เข้าใจในความหมาย หรือรับรู้ได้ว่าตัวอักษรนั้นๆ ต้องการที่จะสื่อหรือเชื่อมต่อกับข้อมูลใดๆ ตามความหมายของแต่ละพื้นที่ ดังรูปที่ 2.7



รูปที่ 2.7 ตัวอย่างความสอดคล้องภายนอกสำหรับ Text Tool Icon โดย Icon

3. ความสอดคล้องกับความจริง

ออกแบบเพื่อทำให้สอดคล้องกับความเป็นจริงโดยการสังเกตและการรับรู้ของผู้ใช้ เช่น สัญลักษณ์ที่มีความหมายซึ่งเป็นที่ยอมรับโดยทั่วไปในสากล ได้แก่ สัญลักษณ์ทางจราจร ดังรูปที่ 2.8



รูปที่ 2.8 ตัวอย่างความสอดคล้องกับความจริง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. ความไม่มีความสอดคล้อง

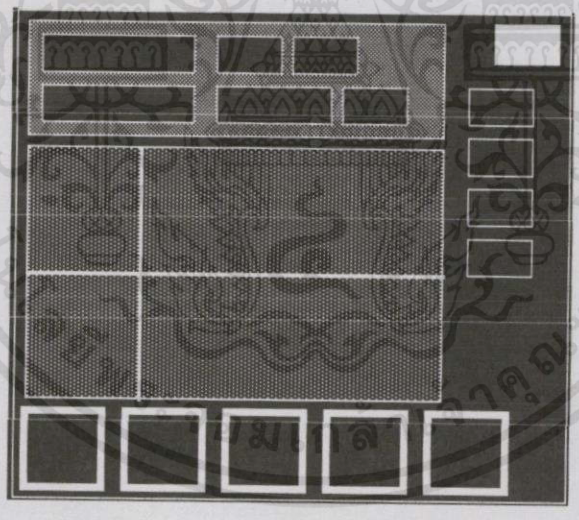
การปรับปรุงข้อมูลที่ไม่สอดคล้องกับกลุ่มของข้อมูลที่แสดงผลอยู่บนหน้าจอให้มีความสอดคล้อง เพื่อให้ผู้ใช้สามารถเข้าใจหรือเข้าถึงข้อมูลได้มากขึ้น โดยทำให้กลุ่มของข้อมูลในหน้าจอแสดงผลมีความสอดคล้องกัน

2.2.4 เค้าโครงหน้าจอ

การออกแบบหน้าจอควรคำนึงถึงความเข้าใจของผู้ใช้เป็นหลัก โดยหลักการออกแบบที่ควรคำนึงถึงในการออกแบบขั้นพื้นฐานนั้น ได้แก่ การใช้โครงสร้างตารางเพื่อแสดงข้อมูล มาตรฐานของรูปแบบหน้าจอ และกลุ่มข้อมูลที่เกี่ยวข้องกับองค์ประกอบหน้าจอแสดงผล

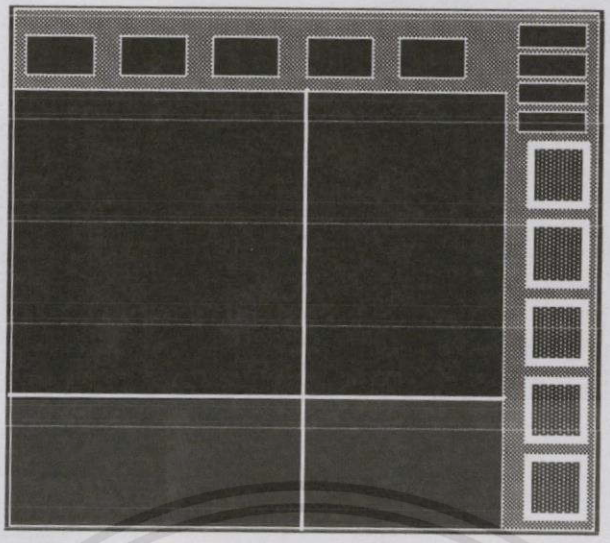
2.2.5 ความสัมพันธ์

ความสัมพันธ์ของหน้าจอเป็นอีกปัจจัยสำคัญที่ช่วยให้ผู้ใช้งานระบบนั้น สามารถเชื่อมโยงข้อมูลที่สามารถมองเห็นผ่านทางหน้าจอ โดยการออกแบบที่ดีนั้นควรจัดหมวดหมู่ของข้อมูลหรือส่วนที่ต้องการให้เชื่อมโยงกันอย่างเป็นระเบียบ เพื่อเพิ่มประสิทธิภาพในการใช้งานของผู้ใช้งานและง่ายต่อการใช้งานดังรูปที่ 2.10 แต่หากผู้ออกแบบนั้นออกแบบหน้าจอที่มีความเชื่อมโยงหรือรายละเอียดมากเกินไปอาจทำให้ผู้ใช้งานเกิดความสับสนกับข้อมูลที่ต้องการแสดงหรือต้องการเชื่อมโยง และทำให้การใช้งานนั้นเป็นไปอย่างยากลำบากดังรูปที่ 2.9 และรูปที่ 2.10



รูปที่ 2.9 ตัวอย่างรูปร่าง ตำแหน่งและความสำคัญในการมองเห็นโดยความสัมพันธ์นี้อาจยังไม่เหมาะสม

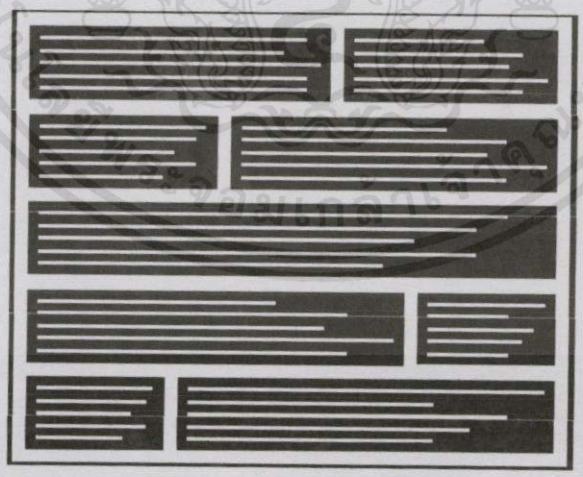
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.10 ตัวอย่างความชัดเจน ความสอดคล้อง ความเหมาะสมและความสัมพันธ์ที่ดี

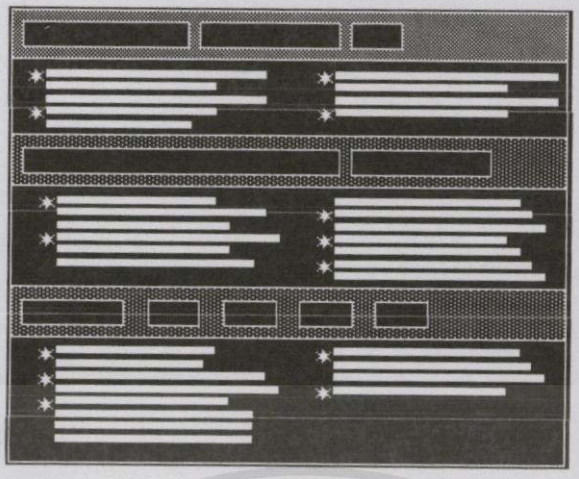
2.2.6 ตำแหน่ง

การออกแบบตำแหน่งการจัดวางข้อมูลในตำแหน่งต่างๆ อย่างเหมาะสมถือเป็นการให้ความสำคัญกับการเริ่มต้นการใช้งาน และยังเป็นการเพิ่มความสนใจเกี่ยวกับข้อมูลที่ปรากฏบนหน้าจอให้กับผู้ใช้งาน และเชื่อมต่อข้อมูลที่เกี่ยวข้องเพื่อให้ง่ายสำหรับการใช้งานของผู้ใช้ดังแสดงในรูปที่ 2.12 หากออกแบบให้ข้อมูลอยู่ที่ตำแหน่งที่ไม่เหมาะสมจะทำให้ผู้ใช้เข้าถึงข้อมูลได้ยากดังรูปที่ 2.11



รูปที่ 2.11 ตัวอย่างการออกแบบที่แย่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.12 ตัวอย่างการออกแบบที่ดี

2.2.7 การสื่อสาร

เป็นสิ่งสำคัญเพื่อทำให้ผู้ใช้สามารถเข้าใจข้อมูล ที่ต้องการแสดงบนหนาจอแสดงผลทั้งการคำนึงถึงความสวยงาม ความสมดุลของข้อมูลที่แสดงผล การอ่านข้อมูลของผู้ใช้ สัญลักษณ์ที่เลือกใช้เพื่อให้ผู้ใช้สามารถเข้าใจในข้อมูลที่ต้องการสื่อสาร ความหลากหลายของมุมมองการแสดงผล สีที่ใช้เพื่อเน้นหรือแสดงข้อมูลต่างๆ เพื่อให้การติดต่อสื่อสารนั้นประสบความสำเร็จดังรูปที่ 2.13

รูปที่ 2.13 ตัวอย่างตัวหนังสือที่มีความชัดเจนและไม่ชัดเจน

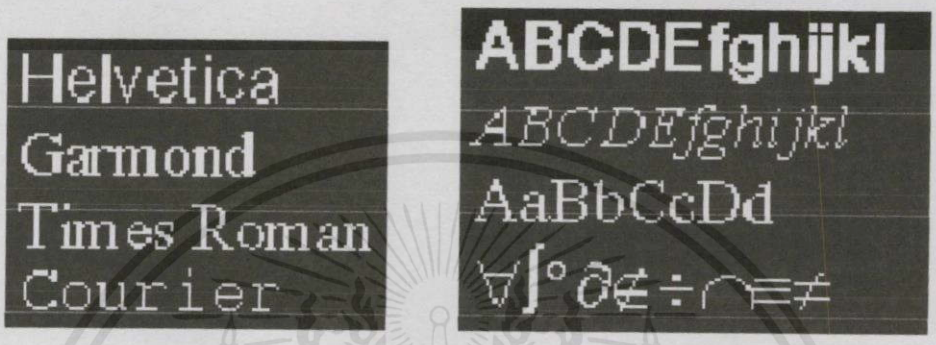
2.2.8 การอ่าน

การออกแบบในส่วนของหน้าจอแสดงผลควรทำให้ง่ายต่อการอ่าน และมีสัดส่วนของข้อมูลต่างๆ ที่เหมาะสมกับหน้าจอในส่วนติดต่อกับผู้ใช้ ซึ่งการออกแบบที่ดีควรทำให้ผู้ใช้แยกแยะข้อมูลที่ต้องการสื่อสารหรือข้อมูลที่แสดงบนหน้าจอแสดงผลได้อย่างเป็นระเบียบและถูกต้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.9 การพิมพ์

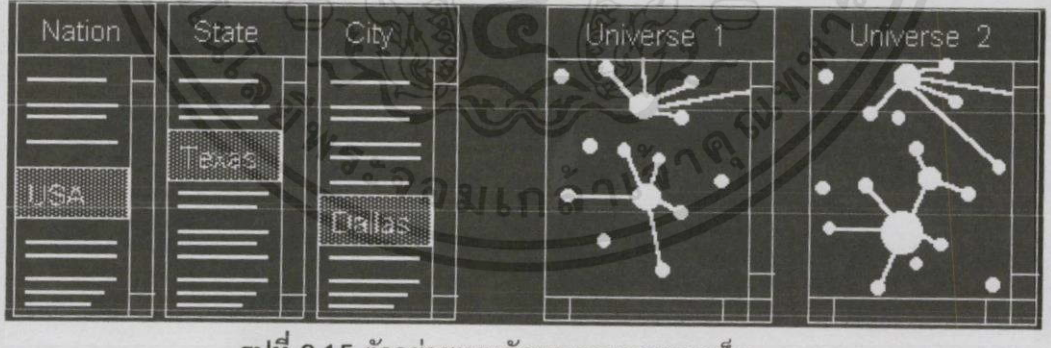
การพิมพ์ตัวอักษรต้องทำให้ผู้ใช้อ่านง่าย ชัดเจนและมีการแสดงผลของข้อมูลที่เด่นชัด โดยตัวอักษรหรือภาษาที่ใช้ในการพิมพ์ควรเป็นอักษรที่เป็นที่ยอมรับของผู้ใช้งานส่วนใหญ่ การออกแบบรูปแบบและขนาดของตัวอักษรที่ใช้ควรมีความเหมาะสม ไม่เล็ก หรือใหญ่จนเกินไปเพื่อให้เข้ากับลักษณะของผู้ใช้งานในทุกช่วงอายุ ดังรูปที่ 2.14



รูปที่ 2.14 ตัวอย่างชนิดของตัวอักษร และ รูปแบบของตัวอักษร

2.2.10 มุมมองในการมองเห็น

การออกแบบในส่วนของมุมมองในการมองเห็น ควรออกแบบให้มีมุมมองหลากหลายมุมมองในการแสดงผลของกระบวนการและโครงสร้างที่มีความซับซ้อน หรือข้อมูลที่เข้าถึงยากให้ผู้ใช้สามารถเข้าถึงข้อมูลและมีความเข้าใจในข้อมูลนั้นได้ง่าย โดยการออกแบบมุมมองที่หลากหลายยังสามารถเพิ่มความน่าสนใจของข้อมูลที่แสดงอยู่บนหน้าจอแสดงผล และน่าใช้มากยิ่งขึ้น ดังรูปที่ 2.15



รูปที่ 2.15 ตัวอย่างแบบอักษร และการมองเห็น

2.3 ภาษา C#

ภาษาC# เป็นภาษา โปรแกรมเชิงวัตถุทำงานบน .NET Framework โดยมีรากฐานมาจากภาษา C++ และภาษาอื่นๆ (โดยเฉพาะภาษาเดลไฟและจาวา) โดยปัจจุบันภาษา C เป็นภาษามาตรฐานรองรับโดย ECMA และ ISO ซึ่งในปัจจุบันได้พัฒนาและปรับรูปแบบของ ภาษา C# อยู่ตลอดเวลาโดยทาง Microsoft ได้นำภาษา C# ไปอยู่ในชุดพัฒนา Software อย่าง Visual Studio ซึ่งทำให้เป็นที่นิยมเพิ่มมากขึ้นไมโครซอฟท์ส่งมาตรฐานภาษา C# ให้กับ ECMA และได้รับการยอมรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้นำไปดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

เป็นมาตรฐาน ECMA ในเดือนธันวาคมปี ค.ศ.2001 ในชื่อว่า ECMA-334 C# Language Specification ในปี ค.ศ. 2003 ภาษา C# ได้รับการยอมรับเป็นมาตรฐาน ISO (ISO/IEC 23270) มาตรฐาน ISO/IEC 23270:2003 ระบุรูปแบบ และกำหนดการแปล (ตีความ) โปรแกรมที่เขียนด้วยภาษา C# ด้วยหนังสือ ภาษา C# เป็นภาษาที่มีต้นกำเนิดมาจากภาษา C เช่นเดียวกับ C++ และถูกพัฒนาขึ้นโดยเป็นส่วนหนึ่งในการพัฒนาโครงสร้างพื้นฐานของ .NET Framework ซึ่งเป็นการนำข้อดีของภาษาต่างๆ เช่น Delphi, C++ มาปรับปรุงเพื่อให้มีความเป็น Object-Oriented Programming อย่างถึงที่สุด ขณะเดียวกันก็พัฒนาให้มีความเรียบง่ายกว่าภาษา C++ นอกจากนี้ภาษา C# ใช้การเขียนโปรแกรมที่ซับซ้อน เช่น โปรแกรมควบคุมฮาร์ดแวร์ และโปรแกรมเกมสามมิติ เพราะ .NET จะมีภาษาให้เลือกใช้หลายสิบภาษา แต่ภาษา C# เป็นพื้นเมืองของ .NET เมื่อค้นใน Web จะพบข้อมูลและตัวอย่างโค้ดเป็นภาษา C# มากกว่าภาษาอื่น โดยข้อดีของภาษา C#นั้นมีดังนี้

1. C# เป็นรูปแบบของภาษาที่ทำงานเป็นลำดับ (Sequential) และต้องผ่านการแปลงไฟล์ให้อยู่ในรูปแบบพร้อมใช้งาน หรือ EXE File (ที่เรียกว่าการ Compile) เช่นเดียวกับภาษา C โดย C# เป็นภาษาที่ถูกผลักดันโดยบริษัทไมโครซอฟท์ให้ออกมายิ่งใหญ่โดยมีจุดเด่นดังนี้

2. เขียนคล่อง C# มีเครื่องมือในการช่วยเขียนอย่าง Visual Studio ที่มีฟีเจอร์ในการไฮไลท์โค้ด และส่วนช่วยในการเขียนโปรแกรมที่สะดวกมาก

3. เขียนง่ายจุดเด่นของตัวแปรและ Object บน C# เด่นชัดในเรื่องของการจัดการคุณสมบัติ (Properties) และการตั้งค่าเริ่มต้นที่ช่วยให้สามารถพัฒนาระบบได้สะดวกรวดเร็วขึ้น

4. อ่านง่าย การจัดระเบียบโดยตัว Visual Studio เป็นส่วนที่ชอบที่สุดเลยก็ว่าได้ เขียนโค้ดไปซักพักหนึ่ง จะเริ่มรู้สึกลายตากับย่อหน้าที่งงๆ แต่ Visual Studio โดยปกติเมื่อจะจัดการกับย่อหน้าทั้งหมดเหล่านั้นให้อยู่ในระเบียบสะอาดตาที่สุดเลยทีเดียว

5. เป็น Object เนื่องจากภาษา C# นั้นมีแม่แบบมาจากภาษา Java ซึ่งจุดเด่นคือการทำทุกสิ่งให้เป็นวัตถุ (Object) ทำให้สามารถเขียนและพัฒนาได้ง่าย

6. ประสิทธิภาพสูง C# เป็นภาษาที่พัฒนาขึ้นมาภายใต้ .NET Framework ซึ่งสามารถดึงเอาความสามารถของเทคโนโลยีบน .NET ออกมาใช้ได้อย่างเต็มประสิทธิภาพ

7. สามารถทำงานระดับลึก ภาษา C# สามารถทำงานกับหน่วยความจำรวมถึงระบบคอมพิวเตอร์ในระดับลึก โดยผ่าน พอยเตอร์ (Pointer) หรือทำงานกับโปรโตคอล TCP/IP ที่ต่ำกว่าระดับ4ได้

8. เน้นที่ XML C# ออกแบบมาเพื่อใช้งานร่วมกับ XML ได้อย่างราบรื่นที่สุดด้วยความช่วยเหลือของ .NET Framework

9. คุณสมบัติ Generic Type ช่วยยืดหยุ่นในการประกาศตัวแปล

10. เทคโนโลยี LINQ ช่วยในการทำงานกับฐานข้อมูลได้อย่างมีประสิทธิภาพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4 ภาษา XML

XML ย่อมาจาก Extensible Markup Language คือ ภาษาหนึ่งที่ใช้ในการแสดงผลข้อมูล ซึ่งภาษาที่ใช้กำหนดรูปแบบของคำสั่งภาษา HTML หรือที่เรียกว่า Meta Data ซึ่งจะใช้สำหรับกำหนดรูปแบบของคำสั่ง Markup ต่าง ๆ ถ้าเปรียบเทียบกับภาษา HTML จะแตกต่างกันที่ HTML ถูกออกแบบมาเพื่อการแสดงผลอย่างเดียวเท่านั้น เช่น ให้แสดงผลตัวเล็ก ตัวหนา ตัวเอียง เหมือนที่เคยเห็นในเว็บเพจทั่วไป แต่ภาษา XML นั้นถูกออกแบบมาเพื่อเก็บข้อมูล โดยทั้งข้อมูลและโครงสร้างของข้อมูลนั้นๆ ไว้ด้วยกัน ส่วนการแสดงผล XSL (Extensible Stylesheet Language)

ภาษา XML มีโครงสร้างที่ประกอบด้วยแท็กเปิด และแท็กปิด เช่นเดียวกับภาษา HTML แต่ภาษา XML สามารถสร้างแท็กรวมทั้งกำหนดโครงสร้างของข้อมูลได้เอง ซึ่งความสามารถตรงนี้ตัวภาษา HTML ทำไม่ได้เพราะภาษา HTML ถูกกำหนดแท็กตายตัวโดย W3C หรือ World Wide Web Consortium อาจกล่าวได้ว่า XML เป็นส่วนเสริมของ HTML เพราะตัว XML ไม่สามารถแสดงผลได้ในตัวของมันเอง หากต้องการแสดงผลที่ถูกต้อง จะต้องมีการใช้ร่วมกับภาษาอื่น เช่น HTML, JSP, PHP, ASP, C, C++, C# หรือภาษาอื่นๆ ที่สนับสนุน XML จะมีนามสกุลเป็น .XML สามารถสร้างขึ้นจากโปรแกรมประเภท Text Editor ใดก็ได้ เช่น Notepad, Editplus , DreamWeaver, MS Word เป็นต้น โดยข้อดีของภาษา XML มีดังนี้

1. ดูเอกสารได้ง่าย สะดวก และได้ผลดีเหมือน HTML
2. สนับสนุนการประยุกต์ใช้กับงานต่างๆ และสนับสนุนโปรแกรมประยุกต์ต่าง ๆ
3. อ่านได้ด้วยมนุษย์ โดยไม่ต้องอาศัยโปรแกรมหรือเครื่องมือช่วยแปล
4. การเขียน XML ทำได้ด้วยการใช้ Text Editor ทั่วไป และไม่ต้องการเครื่องมือที่ซับซ้อน

5. ใช้เป็นตัวควบคุมข้อมูล (Meta Data) จึงเป็นแนวทางในการขนส่งข้อมูล และสร้างการเชื่อมโยง ระหว่างแอปพลิเคชั่นได้ง่าย

6. สนับสนุน UNICODE ทำให้ใช้ได้กับหลากหลายภาษา และผสมกันได้หลากหลายภาษา

7. ดึงเอกสาร XML มาใช้งานได้ง่าย และใช้ร่วมกับโปรแกรมประยุกต์อื่นได้ง่าย เช่นโปรแกรม DB2, Oracle, SAP เป็นต้น

8. นำไปประยุกต์ใช้ในการดำเนินกิจกรรมบนเครือข่ายได้มาก เช่น E-Business, EDI, E-Commerce, การจัดการ Supply Chain/ Demand Chain Management, การดำเนินการแบบ Intranet และ Web Base Application

2.5 ภาษา SQL

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอสคิวแอล (SQL) คือ ภาษาสอบถามข้อมูล หรือภาษาจัดการข้อมูลอย่างมีโครงสร้าง มีการพัฒนาภาษาคอมพิวเตอร์ และโปรแกรมฐานข้อมูลที่รองรับมากมาย เพราะจัดการข้อมูลได้ง่าย เช่น MySQL, MsSQL, PostgreSQL หรือ MS Access เป็นต้น สำหรับโปรแกรมฐานข้อมูลที่ได้รับความ

นิยมคือ MySQL เป็น Open Source ที่ใช้งานได้ทั้งใน Linux และ Windows โดย SQL เป็นภาษาที่ใช้ในการเขียนโปรแกรม เพื่อจัดการกับฐานข้อมูลโดยเฉพาะ เราสามารถแบ่งการทำงานได้เป็น 4 ประเภท ดังนี้

1. Select Query ใช้สำหรับดึงข้อมูลที่ต้องการ
2. Update Query ใช้สำหรับแก้ไขข้อมูล
3. Insert Query ใช้สำหรับการเพิ่มข้อมูล
4. Delete Query ใช้สำหรับลบข้อมูล

SQL เป็นภาษาเกี่ยวกับการจัดการฐานข้อมูล ไม่ว่าจะเป็นในเรื่องของการนิยามข้อมูล การเรียกใช้ หรือการควบคุม การใช้คำสั่งเหล่านี้ที่มีในระบบจัดการฐานข้อมูล จะช่วยประหยัดเวลาในการพัฒนาระบบงาน หรือนำไปใช้ในส่วนของการสร้างฟอร์ม การทำรายงาน ของระบบงานต่างๆ ได้รวดเร็วยิ่งขึ้น ประโยชน์ของภาษา SQL อีกอย่างหนึ่งคือ โปรแกรมระบบฐานข้อมูลส่วนใหญ่สนับสนุนภาษา SQL แทบทั้งสิ้น ดังนั้นถ้าเข้าใจภาษา SQL เท่ากับว่าเราจะสามารถเขียนโปรแกรมติดต่อกับฐานข้อมูลอะไรก็ได้

2.6 โปรแกรม MySQL

MySQL จัดเป็นโปรแกรมจัดการฐานข้อมูลเชิงสัมพันธ์ (RDBMS: Relational Database Management System) ตัวหนึ่ง ซึ่งเป็นที่นิยมกันมากในปัจจุบัน โดยเฉพาะอย่างยิ่งในโลกของอินเตอร์เน็ต สาเหตุเพราะว่า MySQL เป็นฟรีแวร์ทางด้านฐานข้อมูลที่มีประสิทธิภาพสูง เป็นทางเลือกใหม่จากผลิตภัณฑ์ระบบจัดการฐานข้อมูลในปัจจุบัน ที่มักจะเป็นการผูกขาดของผลิตภัณฑ์เพียงไม่กี่ตัว นักพัฒนาระบบฐานข้อมูลที่เคยใช้ MySQL ต่างยอมรับในความสามารถความรวดเร็ว การรองรับจำนวนผู้ใช้ และขนาดของข้อมูลจำนวนมหาศาล ทั้งยังสนับสนุนการใช้งานบนระบบปฏิบัติการมากมาย ไม่ว่าจะเป็น Unix, OS/2, Mac OS หรือWindows ก็ตาม นอกจากนี้ MySQL ยังสามารถใช้งานร่วมกับ Web Development Platform ทั้งหลาย ไม่ว่าจะเป็น C, C++, Java, Perl, PHP, Python, Tcl หรือ ASP ก็ตามที ดังนั้นจึงไม่เป็นที่น่าแปลกใจเลยว่า ทำไม MySQL จึงได้รับความนิยมอย่างมากในปัจจุบัน และมีแนวโน้มสูงยิ่งขึ้นต่อๆ ไปในอนาคต

MySQL จัดเป็นซอฟต์แวร์ประเภท Open Source Software สามารถดาวน์โหลด Source Code ต้นฉบับได้จากอินเตอร์เน็ต โดยไม่เสียค่าใช้จ่ายใดๆ การแก้ไขก็สามารถกระทำได้ตามความต้องการ MySQL ยึดถือสิทธิบัตรตาม GPL (GNU General Public License) ซึ่งเป็นข้อกำหนดของซอฟต์แวร์ประเภทนี้ส่วนใหญ่ โดยจะเป็นการชี้แจงว่าสิ่งใดทำได้ หรือทำไม่ได้สำหรับการใช้งานในกรณีต่างๆ ทั้งนี้ถ้าต้องการข้อมูลเพิ่มเติม หรือรายละเอียดของ GPL สามารถหาข้อมูลได้จากเว็บไซต์ http://www.gnu.org/ [http://kmops.moph.go.th/index.php/km-test/ict/124-mysql]

MySQL ได้รับการยอมรับและทดสอบเรื่องของความรวดเร็วในการใช้งาน โดยจะมีการทดสอบและเปรียบเทียบกับผลิตภัณฑ์ทางด้านฐานข้อมูลอื่นอยู่เสมอ มีการพัฒนาอย่างต่อเนื่อง โดยเริ่มตั้งแต่เวอร์ชันแรกๆ ที่ยังไม่ค่อยมีความสามารถมากนัก มาจนถึงทุกวันนี้ MySQL ได้รับการพัฒนาให้มีความสามารถมากยิ่งขึ้น รองรับข้อมูลจำนวนมหาศาล สามารถใช้งานหลายผู้ใช้ได้พร้อมๆ กัน (Multi-user) มีการออกแบบให้สามารถแตกงานออก เพื่อช่วยการทำงานให้เร็วยิ่งขึ้น รองรับข้อมูลจำนวนมหาศาล เพื่อช่วยการทำงานเร็วยิ่งขึ้น (Multi-threaded) วิธีและการเชื่อมต่อที่ดีขึ้น การ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้นำเนื้อหาไปดัดแปลงต่อเติม หรือเผยแพร่ต่อในรูปแบบอื่นใดไปใช้

กำหนดสิทธิและการรักษาความปลอดภัยของข้อมูลมีความรัดกุมน่าเชื่อถือยิ่งขึ้น เครื่องมือหรือโปรแกรมสนับสนุนทั้งของตัวเองและของผู้พัฒนาอื่นๆ มีมากยิ่งขึ้น นอกจากนี้สิ่งหนึ่งที่สำคัญคือ MySQL ได้รับการพัฒนาไปในแนวทางตามข้อกำหนดมาตรฐาน SQL ดังนั้นสามารถใช้คำสั่ง SQL ในการทำงาน MySQL ได้นักพัฒนาที่ใช้ SQL มาตรฐานอยู่แล้ว ไม่ต้องศึกษาคำสั่งเพิ่มเติม แต่อาจจะต้องเรียนรู้ถึงรูปแบบและข้อจำกัดบางอย่างโดยเฉพาะ ทั้งนี้ทั้งนั้นทางทีมงานผู้พัฒนา MySQL มีเป้าหมายอย่างชัดเจนที่จะพัฒนาให้ MySQL มีความสามารถสนับสนุนตามข้อกำหนด SQL92 มากที่สุด และจะพัฒนาให้เป็นไปตามข้อกำหนด SQL99 ต่อไป

ทุกวันนี้มีการนำ MySQL ไปใช้ในระบบต่างๆ มากมาย ไม่ว่าจะเป็นระบบเล็กๆ ที่มีจำนวนตารางข้อมูลน้อย มีความสัมพันธ์ของข้อมูลในแต่ละตารางไม่ซับซ้อน เช่น ระบบฐานข้อมูลบุคคลในแผนกเล็กๆ ไปจนถึงระบบจัดการข้อมูลขนาดใหญ่ ที่ประกอบด้วยตารางข้อมูลมากมาย มีความสัมพันธ์ของข้อมูลในแต่ละตารางซับซ้อน เช่น ระบบสต็อกสินค้า ระบบบัญชีเงินเดือน เป็นต้น โดยเฉพาะอย่างยิ่งในปัจจุบัน มีการใช้ MySQL เป็น Database Server เพื่อการทำงานสำหรับ Web Database Application ในโลกของอินเตอร์เน็ตมากขึ้น สิ่งนี้เป็นคำตอบว่าทำไมคุณต้องเรียนรู้และใช้งาน MySQL โดยข้อดีข้อดีของ MySQL มีดังนี้

1. เพิ่ม Performance ให้กับแอพพลิเคชั่น เมื่อถูกสร้างขึ้น จะถูกคอมไพล์และเก็บไว้ใน Database แต่ MySQL นั้นเขียน Stored Procedure ในรูปแบบที่แตกต่างไปเล็กน้อยคือ มันจะถูก Compile on Demand เท่านั้น และหลังจาก Compile เสร็จจะเก็บไว้ใน Cache ซึ่งจะเก็บไว้ใช้ใน 1 Connection ซึ่งหมายความว่ามันจะมีประโยชน์ และเพิ่ม Speed ได้ถ้ามีการเรียกใช้หลายๆ ครั้งใน 1 Connection หากไม่แล้วก็จะทำงานเหมือนกับ Query ธรรมดา (ควรพิจารณาดูว่าจะเขียนหรือไม่โดยใช้หลักการข้อนี้)

2. ช่วย ลด Traffic ระหว่าง Application และ Database Server (โดยเฉพาะอย่างยิ่ง กรณีที่ อยู่คนละเครื่องแล้วต้องมีการส่ง Query เส้นยาวๆ หลายๆ ครั้ง ซึ่งต่างกับ Stored Procedure ซึ่งจะมีการส่งแค่ชื่อฟังก์ชัน และ Parameter เท่านั้น)

3. ช่วยในการใช้ซ้ำและช่อนเร้นการทำงาน ในกรณีที่นำไปใช้กับ Application อื่น เนื่องจากแสดงเป็นรูปแบบ Interface ของ Database เท่านั้น ซึ่งเป็นการลดงานของ Developer

4. ความปลอดภัยของ Store Procedure จำกัดโดย Database Admin ซึ่งสามารถกำหนด Permission ต่างๆ ในการเรียกใช้ Stored Procedure ได้

2.7 โปรแกรม MySql Workbench

โปรแกรม MySQL Workbench เป็นโปรแกรมจัดการฐานข้อมูลในรูปแบบของ Windows Application ซึ่งก็เหมือนกับ MySQL ทั่วไปที่อยู่ใน Web Application โดยโปรแกรม MySQL Workbench จะประกอบไปด้วย 3 ส่วนดังนี้

1. SQL Developer ส่วนนี้ทำหน้าที่ในการเชื่อมต่อออกไปยัง Database ต่างๆ ตามที่ได้สร้างเอาไว้ ซึ่งส่วนใหญ่จะใช้หัวข้อนี้เป็นหลัก เพราะว่าจะทำกับโครงสร้าง Table และข้อมูลใน table เป็นส่วนใหญ่ เช่น สร้างตาราง ลบตาราง แก้ไขตาราง เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. Data Modeling ส่วนนี้ทำหน้าที่เป็นเครื่องมือช่วยออกแบบ Database ทั้งโครงสร้างและความสัมพันธ์กันในรูปแบบต่างๆ และสร้างออกมาเป็น SQL Code ให้เลย

3. Server Administration ส่วนนี้ทำหน้าที่เอาไว้ ควบคุม ดูแล จัดการ Server MySQL รวมถึงการ Import/Export SQL โดยเน้นสำหรับคนที่เป็น DBA (Database Administrator) ใช้งานเป็นหลัก

2.8 โปรแกรม phpMyAdmin

โปรแกรม phpMyAdmin เป็นโปรแกรมส่วนต่อประสานที่สร้างโดยภาษาพีเอชพี ซึ่งใช้จัดการฐานข้อมูล MySQL ผ่าน Web Browser โดยสามารถที่จะทำการสร้างฐานข้อมูลใหม่ หรือทำการสร้าง TABLE ใหม่ๆ และยังมี Function ที่ใช้สำหรับการทดสอบการ Query ข้อมูลด้วยภาษา SQL พร้อมกันนั้น ยังสามารถทำการ Insert Delete Update หรือแม้กระทั่งใช้คำสั่งต่างๆ เหมือนกับกันการใช้ภาษา SQL ในการสร้างตารางข้อมูล โดย phpMyAdmin เป็นโปรแกรมประเภท MySQL Client ที่ใช้ในการจัดการข้อมูล MySQL ผ่าน Web Browser ได้โดยตรง phpMyAdmin ตัวนี้จะทำงานบน Web Server เป็น PHP Application ที่ใช้ควบคุมจัดการ MySQL Server โดยความสามารถของ MySQL Sever มีดังนี้

1. สร้าง ลบ คัดลอก และแก้ไข Record/Table/Field/Index
2. ประมวลผลและแก้ไขคำสั่ง SQL
3. โหลดข้อมูลจากไฟล์ข้อความเข้าไปเก็บเป็นข้อมูลในตารางได้
4. หาผลสรุป (Query) ด้วยคำสั่ง SQL
5. บริหารจัดการได้หลายเซิร์ฟเวอร์

2.9 โปรแกรม Microsoft Visual Studio 2013

Microsoft Visual Studio 2013 คือ ชุดพัฒนาโปรแกรมด้วยภาษา Visual Basic .NET กล่าวคือ เป็นเครื่องมือที่ช่วยให้เขียนโปรแกรมด้วยภาษา Visual Basic .NET ได้ Visual Studio เป็นชุดพัฒนาแบบ IDE (Integrated Development Environment) ซึ่งหมายถึง สภาพแวดล้อมที่รวบรวมเครื่องมือและคุณสมบัติทุกอย่างที่จำเป็นสำหรับการพัฒนาโปรแกรมเข้าไว้ด้วยกัน ในที่เดียวกันไม่ว่าจะเป็นการออกแบบหน้าจอ เขียนโค้ด รันเพื่อทดสอบการทำงาน ค้นหาและแก้ไขข้อผิดพลาด เผยแพร่โปรแกรม ฯลฯ นอกจาก Visual Basic .NET แล้ว Visual Studio ยังมีภาษาอื่นๆ ให้ใช้เขียนโปรแกรมได้อีก เช่น ภาษา C# และภาษา C++

ขั้นตอนการเขียนโปรแกรมด้วย Visual Basic .NET

1. ออกแบบหน้าจอของโปรแกรม โดยนำคอนโทรลต่างๆ จาก Toolbox มาวางบนฟอร์ม แล้วปรับขนาดและตำแหน่งตามที่ต้องการ

2. กำหนด Properties ของฟอร์มและออบเจ็กต์ต่างๆ บนฟอร์ม โดยใช้ Windows Properties

3. การเขียนโค้ดควบคุมการทำงานของโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. รันโปรแกรมเพื่อทดสอบว่าโปรแกรมทำงานได้ตามที่ต้องการหรือไม่
5. ค้นหาข้อผิดพลาด ถ้าหากโปรแกรมเกิดข้อผิดพลาดหรือทำงานไม่ได้ตามที่ต้องการ
6. คอมไพล์โปรเจ็คไปเป็นไฟล์โปรแกรมที่รันได้ EXE File

2.10 QR Code

QR Code เป็นบาร์โค้ดสองมิติชนิดหนึ่ง ที่ประกอบด้วยมอดูลสีดำเรียงตัวกัน มีสัณฐานสี่เหลี่ยม มีพื้นหลังสีขาว ที่สามารถอ่านได้ด้วยเครื่องสแกนคิวอาร์ ในโทรศัพท์มือถือที่มีกล้อง และสมาร์ทโฟน เพื่อถอดข้อมูลในรูปข้อความ หรือโปรแกรมชี้แหล่งทรัพยากรสากล

QR Code ทำหน้าที่คล้ายกับ Bar Code ซึ่งสามารถเก็บข้อมูลได้ โดย QR Code หรือเรียกกันอีกชื่อหนึ่งว่า Two-dimensional Bar Code (2D Bar Code) ทำหน้าที่ไว้เก็บข้อมูลต่างๆ ได้เหมือนกันแต่ว่าทำงานเร็วกว่า ใช้งานง่ายกว่า และมีลูกเล่นเยอะกว่า Bar Code ชื่อของ QR Code นั้นมาจากนิยามความหมายว่า Quick Response หรือการตอบสนองที่รวดเร็ว ซึ่งมาจากความตั้งใจของผู้คิดค้นที่จะให้ QR Code นี้สามารถถูกอ่านได้อย่างรวดเร็ว ซึ่ง QR Code นี้ถูกคิดค้นขึ้นในปี ค.ศ. 1994 โดยบริษัทสัญชาติญี่ปุ่น ที่ชื่อ Denso-Wave และได้จดทะเบียนลิขสิทธิ์ชื่อ QR Code ไปแล้วทั้งในญี่ปุ่น และทั่วโลก โดยข้อดีของ QR CODE มีดังนี้

1. สามารถบรรจุข้อมูลได้ในปริมาณสูง

บาร์โค้ดแบบธรรมดานั้นสามารถบรรจุข้อมูลได้สูงสุดเพียง 20 Digits (ตัวเลขจำนวนเดียว 20 ตัว) แต่ QR CODE นั้นสามารถบรรจุข้อมูลได้มากกว่า บาร์โค้ดธรรมดาหลายเท่าตัว และการบรรจุข้อมูลของ QR CODE นั้น ก็ยังไม่ได้จำกัดอยู่เพียงแค่ตัวเลขเท่านั้น แต่ยังสามารถบรรจุ ตัวอักษรเลข (Alphanumeric) ตัวอักษรภาษาญี่ปุ่น (ทั้ง Kanji และ Hiragana) สัญลักษณ์ตัวเลขฐานสอง (Binary) และรหัสสี (Colure Code) อีกด้วย โดยทั้งหมดนี้สามารถจะบรรจุไว้ได้ในคราวเดียวกัน

2. ขนาดเล็ก

QR CODE นั้นสามารถบันทึกข้อมูลได้ทั้งในแนวตั้งและแนวนอน ทำให้ความสามารถในการบรรจุข้อมูลเมื่อเปรียบเทียบกับบาร์โค้ดแบบธรรมดานั้น (ในจำนวนข้อมูลที่เท่ากัน) มีพื้นที่การบันทึกที่น้อยกว่าอย่างเห็นได้ชัด

3. ป้องกันคราบสกปรกและการฉีกขาด

QR CODE นั้นสามารถที่จะอ่านข้อมูลหรือกู้ข้อมูลได้แม้ว่าจะมีการฉีกขาดหรือมีคราบสกปรกเพียงบางส่วน โดยสามารถกู้คืนได้มากที่สุด 30% ของ CODEWORD (1 codeword= 8 bits หรือ 16 ตัวอักษร) และความมากน้อยของข้อมูลที่ถูกกู้คืนจะขึ้นอยู่กับความเสียหายที่เกิดขึ้น

4. สามารถอ่านข้อมูลได้ 360 องศา

QR CODE มีความสามารถในการอ่านข้อมูล 360 องศาด้วยความเร็วสูง โดยความสามารถดังกล่าว ทำได้โดยผ่านรูปแบบของการตรวจสอบตำแหน่ง ที่อยู่ทั้ง 3 มุมของสัญลักษณ์ซึ่งรูปแบบการตรวจสอบเหล่านี้ทำให้เครื่องอ่านมีความเสถียร ในเรื่องของความเร็วในการอ่าน และเป็นตัวป้องกันการรบกวนของพื้นหลังอีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. คุณสมบัติในการควบรวม

QR CODE สามารถแบ่งข้อมูลที่หนึ่งสัญลักษณ์ลงในหลายๆ สัญลักษณ์ได้ และในขณะเดียวกันก็สามารถจะนำสัญลักษณ์ดังกล่าวนั้น มาวางติดกันแล้วอ่านข้อมูลออกมาเป็นชิ้นเดียวกันได้ โดย 1 สัญลักษณ์สามารถแบ่งได้สูงสุดถึง 16 สัญลักษณ์ จึงเป็นประโยชน์อย่างมากต่อการใช้งานในพื้นที่จำกัด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การสร้างและการออกแบบ

ในบทของการสร้างและการออกแบบนี้จะอธิบายเกี่ยวกับโครงสร้างการทำงานโดยคร่าวๆ ของระบบร้านค้าอัจฉริยะ โดยจะได้อธิบายถึงการทำงานของฟังก์ชันต่างๆ ที่อยู่ภายในระบบทั้งในส่วนของศูนย์กระจายสินค้าและในส่วนของสาขาย่อย โดยในส่วนของฟังก์ชันที่แตกต่างกันนี้จะเป็นส่วนที่ทำให้ระบบร้านค้าอัจฉริยะนั้นสามารถทำงานร่วมกับผู้ใช้ได้อย่างมีประสิทธิภาพและทำงานได้อย่างแม่นยำ โดยแบ่งเป็น 3 หลักการดังนี้

1. หลักการทำงานของโปรแกรมในส่วนของ Robot Application
2. Voice Engine
3. Graphical User Interface

3.1 หลักการทำงานของโปรแกรมในส่วนของ Robot Application

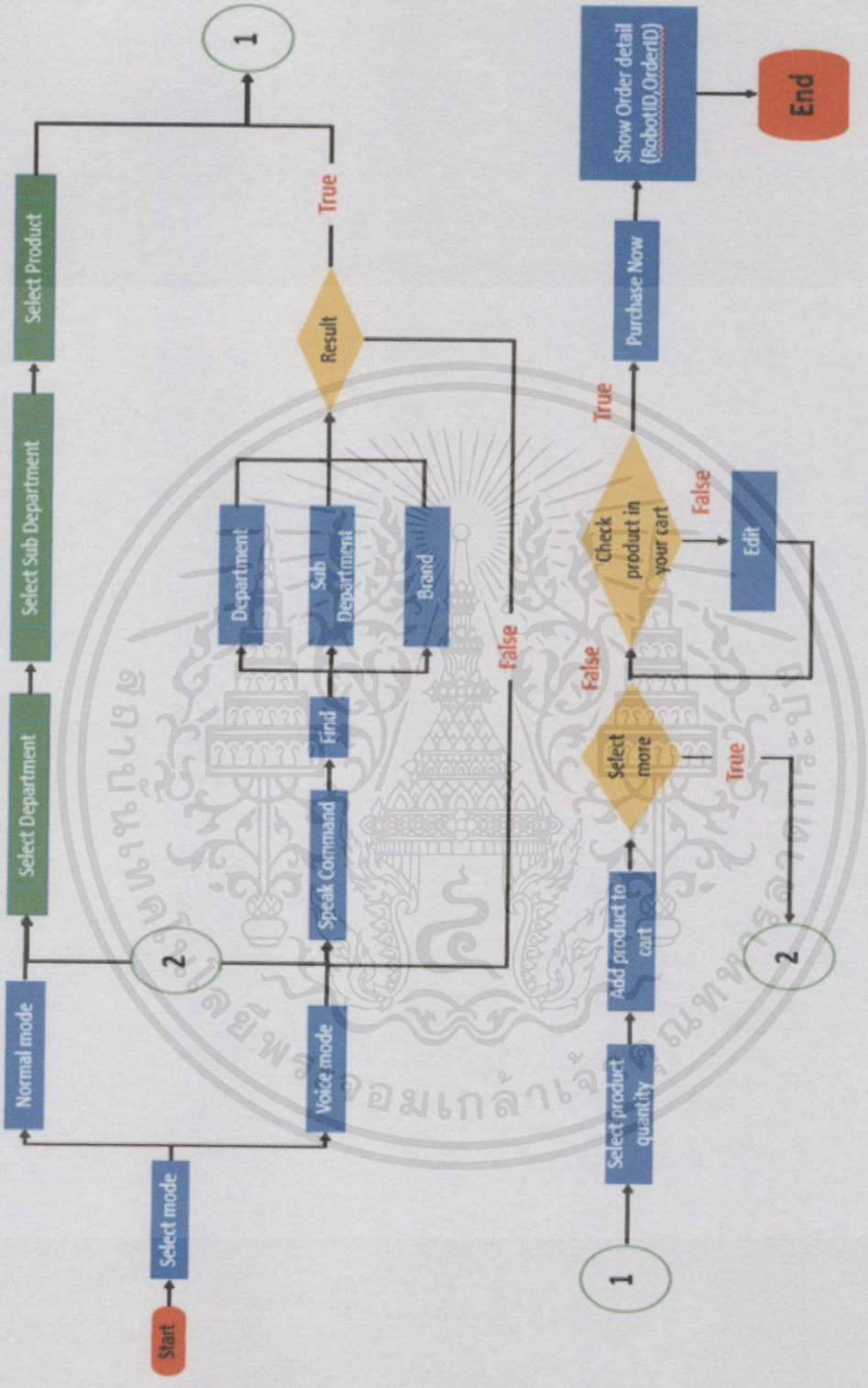
หลักการทำงานโดยรวมในส่วนของระบบนั้นจะแบ่งออกเป็น 2 Mode หลักคือ Voice Mode และ Normal Mode หากเริ่มต้นด้วยการใช้ Voice Mode ขั้นตอนในการเลือกซื้อสินค้าก็คือจะใช้คำสั่งเสียงนั้นในการเลือกไปที่โหมดของเสียง ต่อมาผู้ใช้จะทำการบอกชื่อสินค้าหรือผลิตภัณฑ์ที่ต้องการหรือสนใจที่จะเลือกซื้อ จากนั้นระบบจะทำการประมวลผลเพื่อค้นหาว่าสินค้าที่ผู้ใช้งานนั้นอยู่ใน Department, Sub Department, Brand ซึ่งในส่วนนี้จะเป็นส่วนที่อำนวยความสะดวกสำหรับผู้ใช้มากที่สุดคือ ผู้ใช้ไม่จำเป็นต้องรู้ว่าสินค้าชนิดนั้นเป็นสินค้าประเภทใด อยู่ในหมวดหมู่ของสินค้าใด ซึ่งในส่วนนี้ระบบจะเป็นผู้ทำการประมวลผลและค้นหาให้ผู้ใช้เองโดยหากมีสิ่งใดสิ่งหนึ่งที่ตรงกับสิ่งที่ระบบมีอยู่ในฐานข้อมูลอยู่แล้ว ระบบจะทำการประมวลผลและแสดงสินค้าที่ผู้ใช้ต้องการในหน้าจอ Touch Screen เพื่อให้ผู้ใช้มองเห็นสินค้าที่ผู้ใช้นั้นต้องการก่อนที่ผู้ใช้จะทำการยืนยันเพื่อสั่งซื้อ ซึ่งเมื่อผู้ใช้ยืนยันแล้วว่าสินค้าที่ผู้ใช้ได้สั่งซื้อไปมีความถูกต้องแล้วผู้ใช้จะสามารถเพิ่มหรือลดจำนวนสินค้าตามจำนวนที่ต้องการได้ ซึ่งเมื่อเลือกสินค้าและทำการป้อนจำนวนที่ต้องการสู่ระบบแล้ว ผู้ใช้สามารถทำการเพิ่มสินค้าลงในตะกร้าสินค้าที่แสดงอยู่บนหน้าจอได้ ซึ่งเมื่อผู้ค้าทำการยืนยันสินค้าและจำนวนที่ต้องการไปในระบบแล้ว ระบบจะให้ผู้ใช้ยืนยันการทำรายการอีกครั้งหนึ่งเพื่อให้แน่ใจว่าสินค้าที่ผู้ใช้สั่งนั้นมีความครบถ้วนและถูกต้อง ซึ่งในกระบวนการนี้หากผู้ใช้ยืนยันการทำรายการแล้วสินค้าที่ผู้ใช้เลือก และจำนวนที่ผู้ใช้เลือกจะถูกเพิ่มเข้าไปในใบสั่งซื้อรายการทันที ซึ่งผู้ใช้สามารถทำรายการอื่นต่อไปได้โดยทำตามกระบวนการข้างต้นอีกครั้ง จนกว่าผู้ใช้จะทำการสั่งซื้อสำเร็จตามขั้นตอนต่างๆ และเมื่อเลือกซื้อสินค้าแต่ละประเภทแล้ว ลูกค้าสามารถรอรับสินค้าที่ Cashier ซึ่งจะมีรหัส Order สินค้าที่ตัวของผู้ใช้ทำการยืนยันไปในตอนแรก และสามารถทำการจ่ายเงินได้ทันที หรือโดยอาจตรวจสอบความถูกต้องเพื่อให้แน่ใจอีกครั้งโดยการสแกน QR CODE ที่แสดงในระบบเพื่อตรวจสอบความถูกต้องได้อีกครั้ง แต่หากผู้ใช้เริ่มการใช้งานโดยการใช้ Normal Mode จะมีขั้นตอนดังนี้คือ ผู้ใช้จะเป็นผู้เลือกและหาสินค้าเองทั้งหมดผ่านระบบ Touch Screen โดยในส่วนนี้ผู้ใช้จำเป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น รวมทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะต้องทราบว่าสินค้านั้นเป็นสินค้าประเภทใด อยู่ในหมวดหมู่ใด หรือเป็นแบรนด์ใด เพื่อความสะดวกและรวดเร็วในการใช้งาน แต่หากผู้ใช้ไม่แน่ใจหรืออาจต้องการที่จะเลือกซื้อสินค้าจากฐานข้อมูลทั้งหมดก็สามารถทำได้เช่นกัน เพราะสามารช่วยลดระยะเวลาในการเดินหาสินค้าและทำให้ผู้ใช้สามารถตรวจสอบราคาของสินค้าต่างๆ เพียงแค่ทำการตรวจสอบดูในหน้าจอ Touch Screen แทน เมื่อผู้ใช้ได้สินค้าที่ต้องการแล้วระบบก็จะทำงานเช่นเดิมคือ ให้ผู้ใช้เพิ่มจำนวนสินค้าตามที่ผู้ใช้ต้องการและเพิ่มสินค้านั้นๆ ลงในตะกร้าสินค้า ซึ่งเมื่อเลือกสินค้าที่ต้องการลงในตะกร้าแล้วผู้ใช้ก็สามารถไปเลือกซื้อสินค้าประเภทอื่นอีกได้เช่นกัน เมื่อเลือกสินค้าตามที่ต้องการและจำนวนที่แน่นอนแล้วระบบจะให้ลูกค้าตรวจสอบความถูกต้องของสินค้า และยืนยันเพื่อจะทำการสั่งซื้อและไปรอชำระเงินที่ Cashier เป็นอันเสร็จกระบวนการ ซึ่งจะเห็นได้ว่าการใช้ระบบนี้เข้ามาช่วยเหลือผู้ใช้ในการเลือกซื้อสินค้าทำให้เกิดความสะดวกและรวดเร็ว อีกทั้งยังมีประโยชน์สำหรับผู้ที่ไม่สามารถช่วยเหลือตัวเองได้มากนัก เช่น ผู้ชรา หรือคนที่บกพร่องทางร่างกายต่างๆ และยังเป็นการเพิ่มทางเลือกใหม่ในการใช้บริการร้านค้า เพื่อที่จะทำให้เกิดความน่าสนใจและเป็นผลดีต่อธุรกิจนั้นอีกด้วยดังแสดงในรูปที่ 3.1

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.1 หลักการทำงานของโปรแกรมในส่วนของ Robot Application

โดยในส่วนของ Voice Engine นี้จะประกอบไปด้วยโปรแกรมที่นำมาใช้ในระบบหลัก 2 โปรแกรมคือ Windows Speech Recognition And Windows Speech Synthesis

3.2 Voice Engine

3.2.1 Windows Speech Recognition

Windows Speech Recognition จะมีขั้นตอนการปฏิบัติอยู่ทั้งหมด 4 ขั้นตอน ซึ่งสามารถอธิบายโดยสรุปได้ดังนี้

ขั้นที่ 1 แปลงคลื่นเสียงที่มากระทบในขั้นต้นให้เป็นตัวเลขที่ต้องการเพื่อทำความเข้าใจ

ขั้นที่ 2 คำนวณถึงลักษณะเฉพาะซึ่งเป็นสัญลักษณ์ของ Spectral โดย Domain ที่เป็นหัวเรื่องของ Speech ลักษณะเหล่านั้นจะถูกคำนวณทุกๆ 10 msec โดยแต่ละ 10 msec จะถูกเรียกว่า “Frame“

ขั้นที่ 3 Artificial Neural Network (ANN) แบบ Multi – Layer Perceptron (MLP) จะถูกใช้เพื่อแยกชั้นของกลุ่มของลักษณะเหล่านั้นไปสู่ Phonetic-based Categories ในแต่ละ Frame

ขั้นที่ 4 Viterbi Search จะทำการจับคู่ Neural Network Output Scores กับคำศัพท์ที่ต้องการ

Viterbi Search ใช้กำหนดถึงความต่อเนื่องของหน่วยพื้นฐานของเสียงจากความเป็นไปได้สูงสุด โดยคำนวณผ่าน Gaussian Mixture Model ข้อดีของ Viterbi Search คือการประมวลผลข้อมูลเป็นแบบ Real Time แต่ข้อเสียคือ ระบบจะเลือกตัด Path ที่ Prob ต่ำเกินกว่ากำหนดไปและไม่นำกลับมาพิจารณาอีกเลย ทั้งๆ ที่บางครั้ง Path นั้นอาจจะมีค่าความน่าจะเป็นรวมสูงขึ้นกว่า Path อื่นเมื่อ Search ถึง Word ท้ายของประโยค ข้อดีและข้อเสีย แสดงดังในตารางที่ 3.1

ตารางที่ 3.1 ข้อดีข้อเสียของ Windows Speech Recognition

ข้อดี	ข้อเสีย
- สามารถใช้ได้ในหลายสภาวการณ์ทั้งที่ปกติและไม่ปกติ เช่น ขณะที่มือไม่ว่าง, ต้องการความคล่องตัว, สายตาไม่ว่าง, ไม่ต้องการใช้คีย์บอร์ด, มีข้อจำกัดทางร่างกาย ฯลฯ	- ผู้ใช้ต้องออกเสียงชัดเจน และพูดตามอักขระให้ถูกต้อง
- ช่วยประหยัดเวลาในการทำงาน	- ขณะใช้ต้องปราศจากเสียงรบกวนจากภายนอก เพื่อไม่ให้โปรแกรมประมวลผลผิดพลาด
- ช่วยอำนวยความสะดวกมากยิ่งขึ้น	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.2 Windows Speech Synthesis

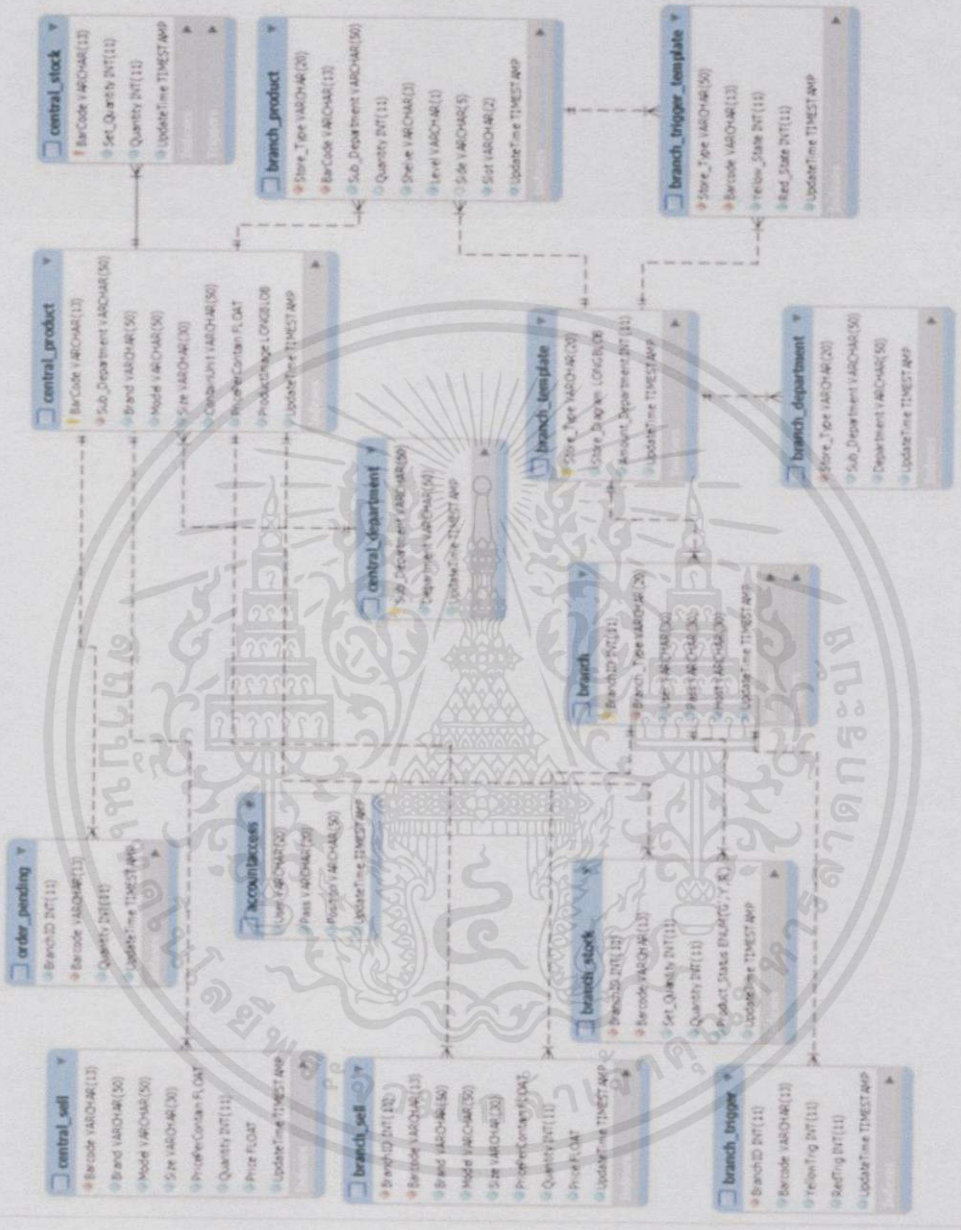
หลักการทำงานของ Windows Speech Synthesis จะสามารถแบ่งได้แบ่ง 2 ส่วน คือ ส่วนวิเคราะห์ข้อความ (Text Analysis) และส่วนสังเคราะห์เสียง (Sound Generation) ในส่วนของการวิเคราะห์ข้อความจะรับข้อมูลที่เป็นคำ วลี หรือประโยค จากนั้นตัวโปรแกรมจะทำการวิเคราะห์คำต่างๆ ในประโยคนั้นๆ ว่าเป็นคำนาม คำเชื่อม คำกริยา ฯลฯ (Part Of Speech) และวิเคราะห์กาลของประโยค (Tense) รวมทั้งประเภทของประโยคด้วย เช่น ประโยคคำถาม ประโยคคำสั่ง จากนั้นจะส่งค่าต่างๆ ออกสู่โปรแกรมในส่วนที่สองคือ ส่วนสังเคราะห์เสียง โดยในส่วนสังเคราะห์เสียงนี้จะสร้างระดับเสียงสูง ต่ำ ให้สอดคล้องตามรูปแบบของคำและประโยคที่รับเข้ามา เช่น ในภาษาอังกฤษการออกเสียงในประโยคคำถาม จะมีการเน้นเสียงให้สูงขึ้นในส่วนท้ายของประโยค เป็นต้น ข้อดีและข้อเสียแสดงดังในตารางที่ 3.2

ตารางที่ 3.2 ข้อดีและข้อเสียของ Window Speech Synthesis

ข้อดี	ข้อเสีย
- ทำให้ผู้ใช้งานสามารถรับรู้สถานะที่จำเป็นต่างๆ ของระบบได้ ด้วยข้อความตอบรับแบบเสียง	- ในกรณีที่มีการทำงานพร้อมกันหลายๆ เครื่อง อาจทำให้ผู้ใช้งานสับสนได้ว่าเสียงตอบรับนั้นมาจากเครื่องใด วิธีแก้ไขคือ อาจใช้โทนเสียงที่แตกต่างกันเพื่อลดความสับสนที่อาจเกิดขึ้นได้ผู้ใช้งาน
- ทำ Application เป็นมิตรยิ่งขึ้น	
- ลดช่องว่างระหว่างเครื่องจักรกับมนุษย์มากยิ่งขึ้น	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในส่วนของภาพรวมในการทำงานนั้นได้แบ่งระบบการจัดการกับทั้งสาขาใหญ่และสาขาย่อย ออกเป็น 2 ใหญ่ด้วยกันคือ ในส่วนของโครงสร้างฐานข้อมูลของศูนย์กระจายสินค้า และในส่วนของโครงสร้างฐานข้อมูลของสาขาย่อย โดยในแต่ละส่วน จะมีการอธิบายรายละเอียดดังต่อไปนี้

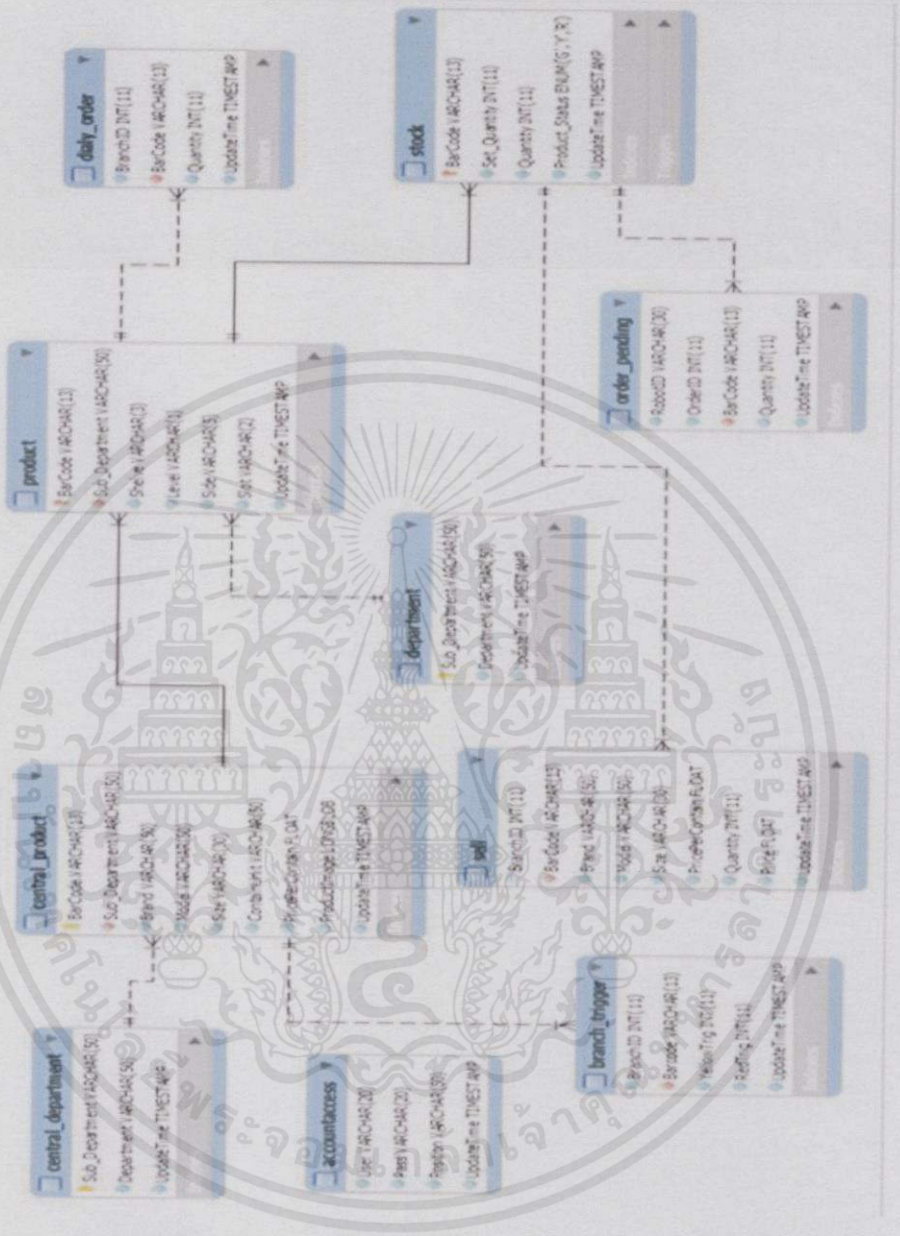


รูปที่ 3.2 ส่วนของโครงสร้างฐานข้อมูลของศูนย์กระจายสินค้า

Central Database (ER Diagram) สำนักงานใหญ่จะมีหน้าที่ในการบริหารจัดการการดำเนินงานของสาขาย่อย รวมทั้งกำกับดูแลให้ผลประกอบการเป็นไปตามเป้าหมายที่วางไว้ และยังเป็นศูนย์กลางฐานข้อมูลของทุกสาขา เช่น ข้อมูลสินค้า ข้อมูลการขาย รายละเอียดของสาขาแต่ละสาขา ยอดขายในแต่ละสาขา รายการสินค้าที่ต้องจัดส่งให้สาขาย่อยแต่ละสาขาในวันถัดไป ฯลฯ นอกจากนี้ยังทำหน้าที่ร้องขอข้อมูลการขาย การสั่งซื้อสินค้าของแต่ละสาขาเพื่อใช้ในการวางแผน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้นำไปใช้

การตลาดต่อไป และสามารถประเมินผลการขายได้ ทั้งในระยะสั้น (รายวัน) หรือระยะยาว (รายเดือน รายไตรมาส หรือรายปี) ได้ ดังแสดงในรูปที่ 3.4



รูปที่ 3.3 โครงสร้างฐานข้อมูลของสาขาย่อย

Branch Database (ER Diagram) ข้อมูลที่จำเป็นต่างๆ ที่สาขาย่อยแต่ละสาขาจำเป็นที่จะต้องใช้ เช่น ข้อมูลสินค้าภายในร้าน (Central Product Table และ Product Table) ข้อมูลร้านค้า (Central Department Table และ Department Table) ข้อมูลการขาย (Order Pending Table และ Sell Table) การบริหารสินค้าคงคลัง (Stock Table, Trigger Table และ Daily Order Table) ฯลฯ จะถูกจับเก็บในรูปแบบของข้อมูลเชิงสัมพันธ์ด้วยเครื่องมือสร้างฐานข้อมูลอย่าง MySQL โดยพนักงานในแต่ละส่วนสามารถเข้าจัดการฐานข้อมูลได้เฉพาะในส่วนที่ตนเองรับผิดชอบ เช่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

พนักงานบัญชีสามารถเข้าตรวจสอบ และจัดการข้อมูลเกี่ยวกับการขายได้ เป็นต้น ซึ่งจะอธิบายได้ดังแสดงในรูปที่ 3.3

3.3 Graphical User Interface

เป็นการออกแบบส่วนติดต่อระหว่างผู้ใช้งานกับโปรแกรม ในการสร้างระบบร้านค้าอัจฉริยะ ทั้งการออกแบบร้านค้า การช่วยเพิ่มฐานข้อมูล การสำรองฐานข้อมูล การช่วยจัดการฐานข้อมูลตามตำแหน่งของผู้ใช้งาน การกำหนดปริมาณสินค้า การดูปริมาณการขายสินค้า ระบบการคิดเงิน และส่วนติดต่อระหว่างหุ่นยนต์กับลูกค้า โดยแบ่งการใช้งานเป็น 5 ส่วนใหญ่ๆ คือ

1. Central Back Office Application
2. Central Daily Backup
3. Branch Back Office Application
4. Cashier Application
5. Robot Application

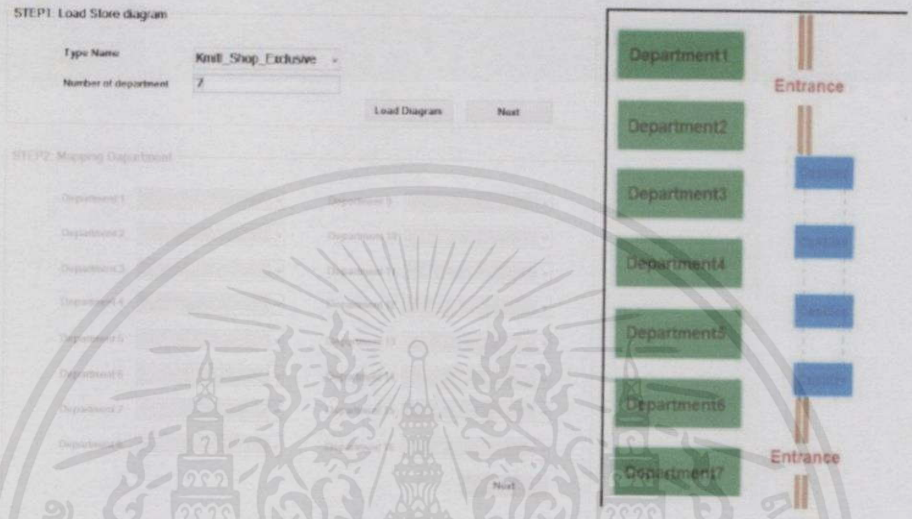
3.3.1 Central Back Office Application

หน้าที่งาน Branch Editor สร้างหน้าต่าง Product Access เพื่อทำหน้าที่ช่วยเพิ่มสินค้าลงในฐานข้อมูล โดยจะสามารถกำหนดว่าสินค้านั้นอยู่ใน Department, Sub Department, Brand, Model, Size, Contain Unit, Bar Code เมื่อทำการกำหนดทั้งหมดเรียบร้อยแล้วจึงนำภาพของสินค้านั้นๆ มาใส่ตามความเหมาะสมและเพิ่มลงไปในฐานข้อมูล

รูปที่ 3.4 Product Access

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Create Branch Template เลือก Template ให้กับร้านค้า และออกแบบการวางสินค้าในแต่ละแผนกว่าสินค้าแต่ละประเภทควรตั้งอยู่ตำแหน่งใด เพื่อให้เกิดความสะดวกในการเลือกหรือการจัดการว่าสินค้าลักษณะใดควรวางอยู่ที่ตำแหน่งใด โดยสามารถออกแบบได้ตามความเหมาะสมในแต่ละสาขา โดยสามารถตั้งชื่อสาขาต่างๆ ได้ที่หน้าจอนี้ และสามารถกำหนด Template ได้ตามที่ออกแบบ



รูปที่ 3.5 Create Branch Template

- Quantity for Template ออกแบบหน้าต่าง Set Guantity for Template เพื่อกำหนดปริมาณสินค้าในแต่ละ Template และกำหนดปริมาณสำหรับสถานะการแจ้งเตือน Yellow State (สถานะสินค้าเหลือน้อย) และ Red State (สถานะสินค้าเหลือน้อยมาก) โดยสามารถเข้าไปกำหนดปริมาณของสินค้าได้ในช่อง Quanitity และเลือกสินค้าที่ต้องการจะกำหนดสถานะหรือปริมาณได้ในช่อง Department และ Sub Department และยังสามารถเลือกปรับได้ในแต่ละสาขาที่ทำการกำหนดชื่อไว้แล้วด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.6 Set Quantity for Template

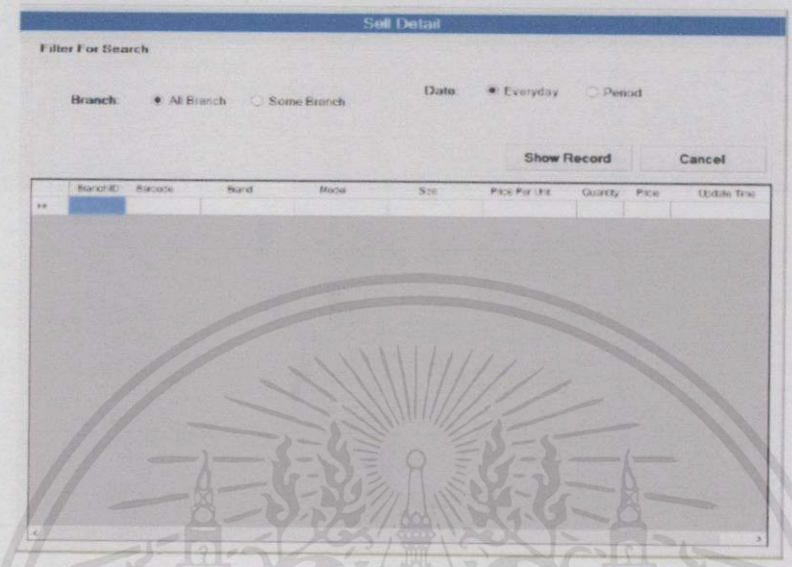
- New Branch ออกแบบหน้าต่างในการสร้างสาขาย่อย โดยกำหนดหมายเลขสาขา และชนิดของ Template ซึ่งในส่วนนี้จะเป็นในส่วนสาขาที่ต้องการจะเพิ่ม โดยเมื่อเพิ่มสาขาแล้วสามารถเข้าไปปรับตั้งค่าต่างๆ ดังที่กล่าวไว้ข้างต้นแล้ว

รูปที่ 3.7 New Branch

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้าที่งาน Finance

- Sell Detail ออกแบบหน้าต่าง Sell Detail เพื่อเรียกดูปริมาณการขายสินค้าในแต่ละสาขาหรือทุกสาขา โดยสามารถดูการขายของทุกวันหรือช่วงเวลาใดเวลาหนึ่งก็ได้ ตามที่ผู้ที่มีหน้าที่รับผิดชอบต้องการที่จะเข้าไปตรวจสอบเพื่อพิจารณายอดขายหรือปริมาณสินค้า

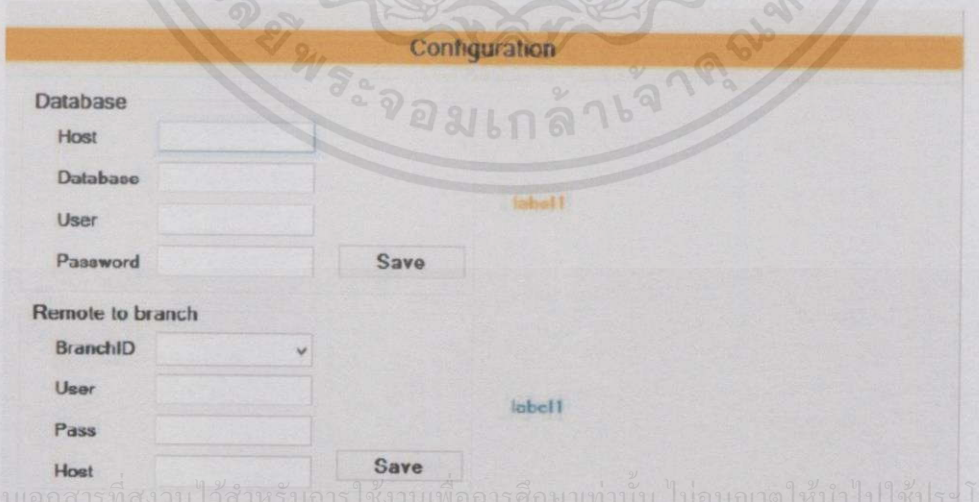


รูปที่ 3.8 Sell Detail

หน้าที่งาน IT

- Central Back Office Configuration ออกแบบหน้าต่าง Central Back Office Configuration สำหรับตั้งค่าโปรแกรมในส่วนของศูนย์กระจายสินค้า ว่าควรใช้ฐานข้อมูลไหน และใช้รหัสอะไร Login โดยเมื่อทำการเข้ารหัสที่มีการตั้งค่าไว้แล้วนั้นจะทำให้สามารถปรับเปลี่ยนหรือแก้ไขฐานข้อมูลของแต่ละสาขาได้ตามที่กำหนดไว้ในครั้งแรกของการตั้งค่าในตารางนี้

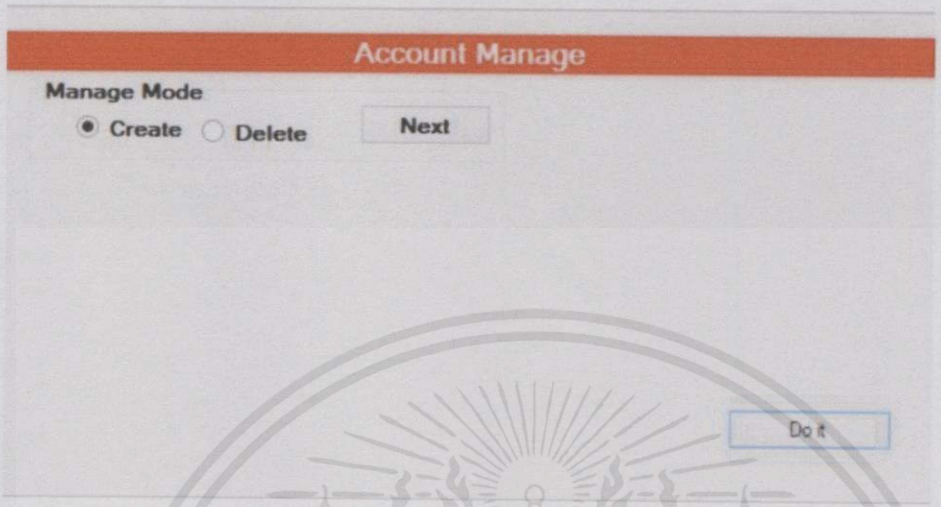
รูปที่ 3.9 Central Back Office Configuration



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Account Manage

- ออกแบบหน้าต่าง Account Manage สำหรับสร้างหรือลบบัญชีผู้ใช้งาน เพื่อใช้ในการจัดการฐานข้อมูลใน Central Back Office

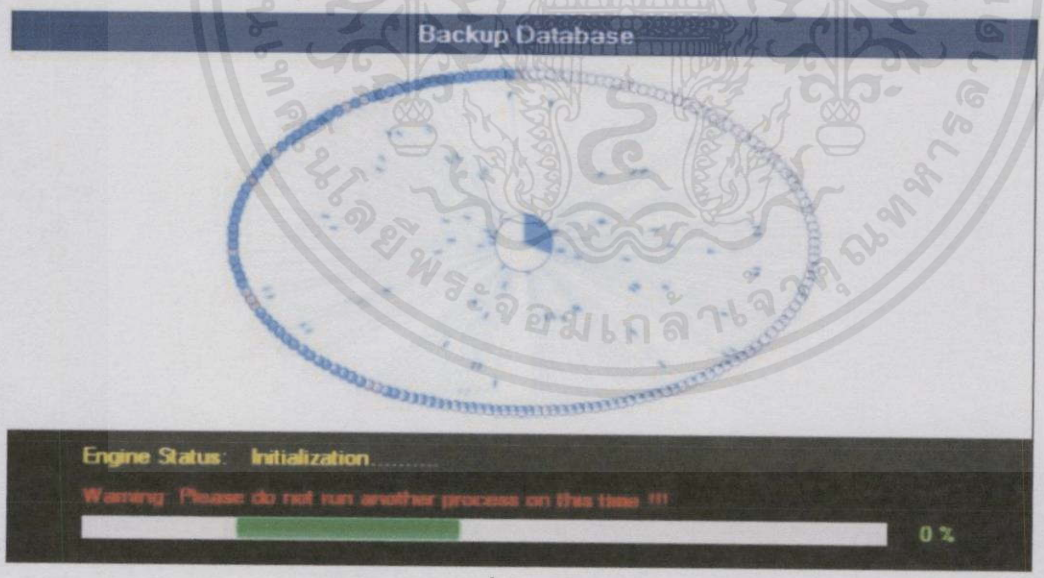


รูปที่ 3.10 Account Manage

3.3.2 Central Daily backup

Call for Backup

- ออกแบบหน้าต่าง Call for Backup เพื่อใช้ติดต่อกับโปรแกรมในการสำรองข้อมูลทั้งหมดจากขาสาย่อย เข้ามาไว้ใน Central Back Office



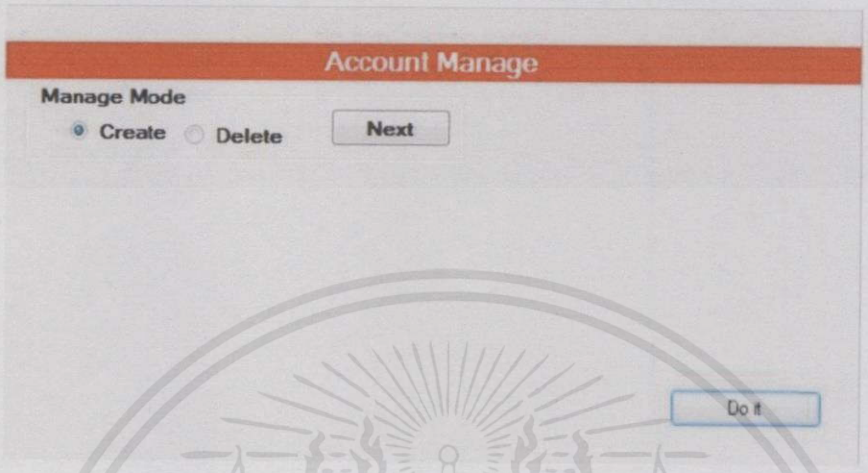
รูปที่ 3.11 Call for Backup

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.3 Branch Back Office Application

หน้าที่งาน IT

- Account Manage ออกแบบหน้าต่าง Account Manage สำหรับสร้างหรือลบบัญชีผู้ใช้งาน เพื่อใช้ในการจัดการฐานข้อมูลใน Branch Back Office



รูปที่ 3.12 Branch Account Manage

Branch Back Office Configuration

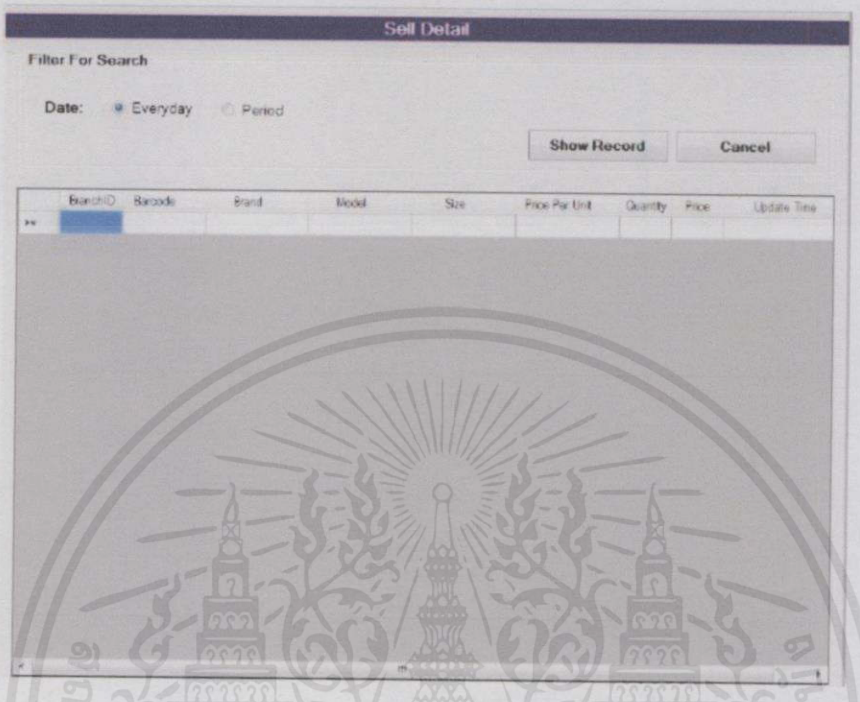
- ออกแบบหน้าต่าง Branch Back Office Configuration สำหรับตั้งค่าโปรแกรมในส่วนของ Branch Back Office ว่าควรใช้ฐานข้อมูลไหน และใช้รหัสอะไร Login

รูปที่ 3.13 Branch Back Office Configuration

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้าที่งาน Finance

- Sell Detail ออกแบบหน้าต่าง Sell Detail เพื่อเรียกดูปริมาณการขายสินค้าสำหรับสาขาย่อย โดยสามารถดูการขายของทุกวันหรือช่วงเวลาใดเวลาหนึ่งก็ได้

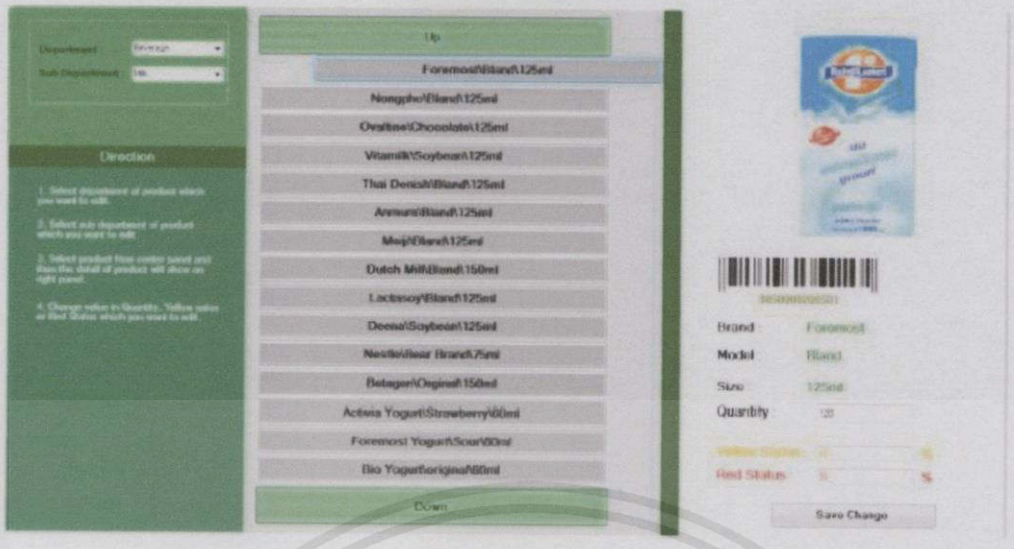


รูปที่ 3.14 Sell Detail

ตำแหน่งงาน Stock Manage

- Set Guantity in Branch ออกแบบหน้าต่าง Set Guantity in Branch เพื่อกำหนดปริมาณสินค้าในสาขาย่อย และกำหนดปริมาณสำหรับสถานะการแจ้งเตือน Yellow State (สถานะสินค้าเหลือน้อย) Red State (สถานะสินค้าเหลือน้อยมาก)

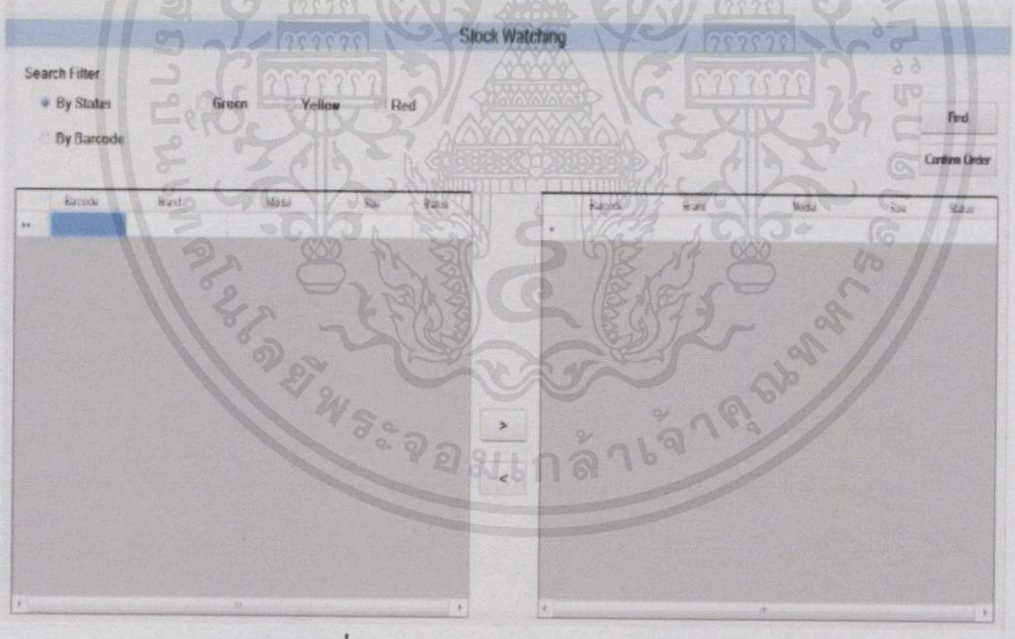
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.15 Set Quantity in Branch

Stock watching

- ออกแบบหน้าต่าง Stock Watching เพื่อดูปริมาณสินค้าที่คงเหลือในแต่ละสาขาย่อย โดยสามารถเรียกดูจากสถานะคงเหลือของสินค้า หรือจากบาร์โค้ดก็ได้

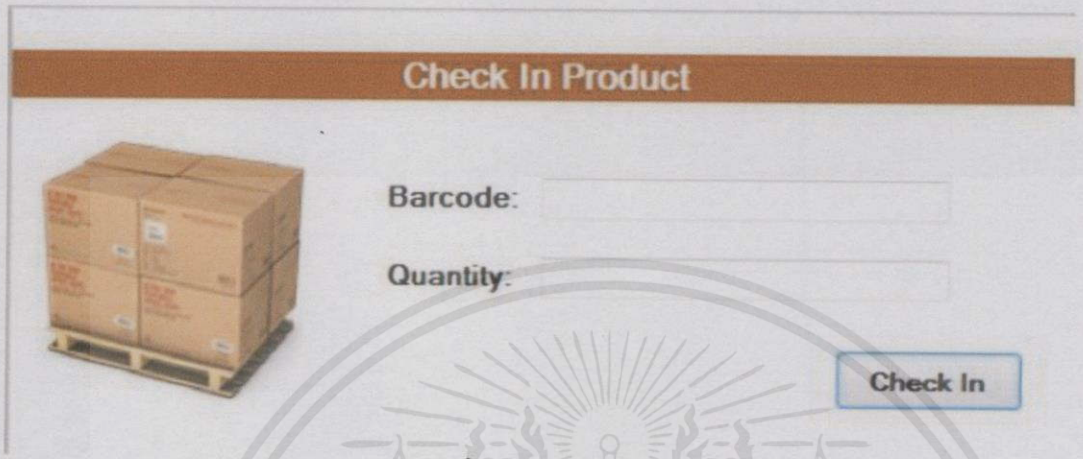


รูปที่ 3.16 Stock Watching

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Product Check In

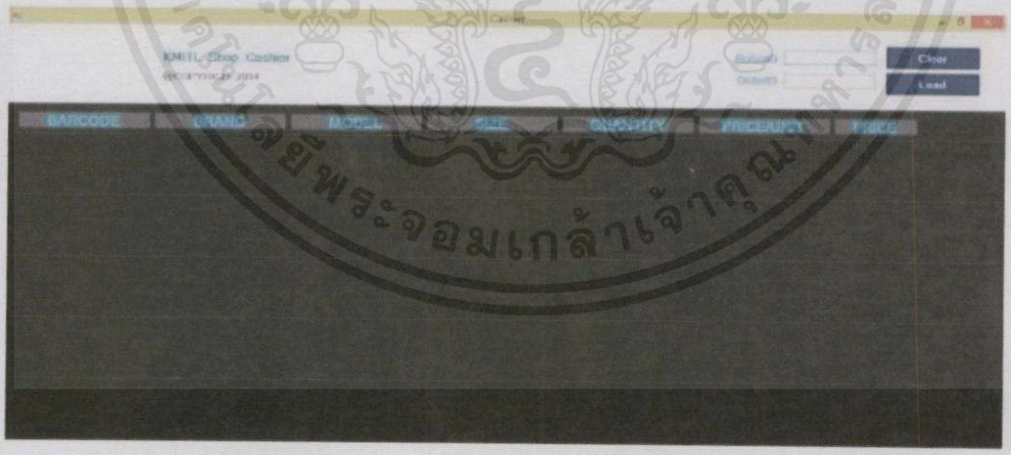
- ออกแบบหน้าต่าง Product Check In เพื่อให้พนักงานในแต่ละสาขาเช็คของเข้าคลังสินค้า โดยกำหนดบาร์โค้ด และจำนวนสินค้า



รูปที่ 3.17 Product Check In

3.3.4 Cashier Application

- Cashier Application ออกแบบหน้าต่างสำหรับ Cashier ในการดูแลรับผิดชอบเกี่ยวกับการชำระเงิน



รูปที่ 3.18 Cashier Application

- Cashier Application Configuration ออกแบบหน้าต่าง Cashier Application Configuration เพื่อใช้ตั้งค่าต่างๆให้กับโปรแกรม Cashier

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.19 Cashier Application Configuration

3.3.5 Robot Application

- Robot Application Configuration ออกแบบหน้าต่าง Robot Application Configuration เพื่อใช้ตั้งค่าต่างๆ ให้กับโปรแกรม Robot โดยในส่วนนี้จะสามารถตั้งค่าการ Login เพื่อแก้ไขในส่วนของโปรแกรมภายในระบบของร้านค้า และยังสามารถตั้งค่าชื่อของ Host, Database, User, Password และยังสามารถตั้งชื่อ Robot ในตารางนี้ได้อีกด้วย

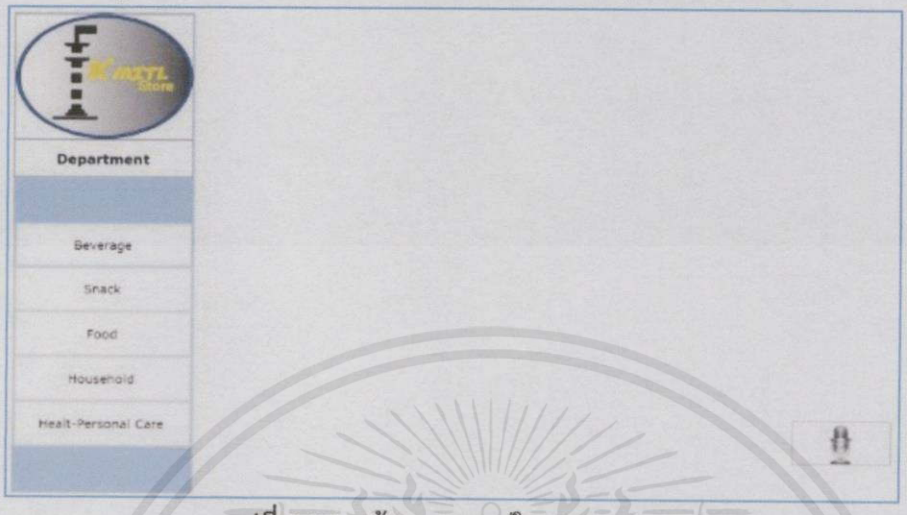
รูปที่ 3.20 Robot Application Configuration

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ทำซ้ำ ดัดแปลง เผยแพร่ หรือนำไปใช้

- Robot Application

1. Main ออกแบบหน้าต่างส่วนติดต่อหลักระหว่างหุ่นยนต์กับผู้ใช้งาน ซึ่งสามารถเลือกสั่งงานได้ 2 โหมดคือ คำสั่งเสียง และหน้าจอสัมผัสโดยสามารถเลือกใช้งานใน Voice Mode ได้โดย

การคลิกที่รูป Microphone โดยหน้าจอจะแสดงพื้นหลังสีแดงด้านหลัง Microphone และมี Icon เคลื่อนไหวอยู่ด้านบนซ้ายเพื่อแสดงว่าระบบกำลังอยู่ใน Voice Mode เมื่อต้องการกลับสู่ Normal Mode ให้คลิกที่รูป Microphone อีกครั้งจนพื้นหลังของ Microphone เป็นสีขาว

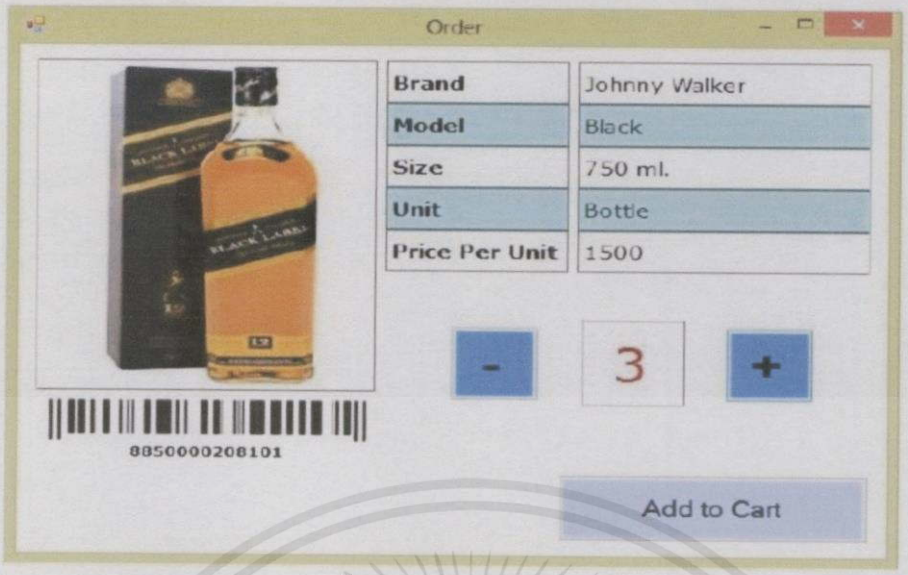


รูปที่ 3.21 หน้าจอขณะอยู่ใน Normal Mode

รูปที่ 3.22 หน้าจอขณะอยู่ใน Voice Mode

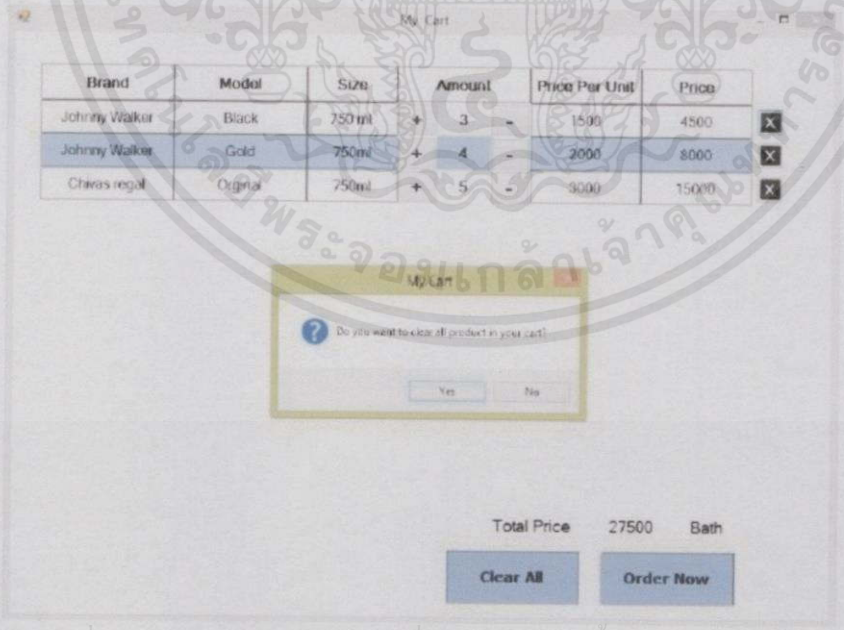
2. Pop Up ออกแบบหน้าต่าง Pop Up สำหรับแสดงรายละเอียดสินค้า เลือกจำนวนสินค้าและเพิ่มสินค้าลงในตะกร้า โดยเมื่อต้องการลดปริมาณสินค้าให้กดที่ปุ่มที่มีเครื่องหมายลบ และเช่นเดียวกันหากต้องการเพิ่มจำนวนสินค้าให้มีจำนวนมากกว่าเดิมให้กดปุ่มที่มีเครื่องหมายบวก และเมื่อผู้ใช้งานต้องการเพิ่มสินค้าเข้าไปในรายการสั่งซื้อให้กดที่ปุ่ม Add to Cart

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.23 หน้าจอ Pop Up

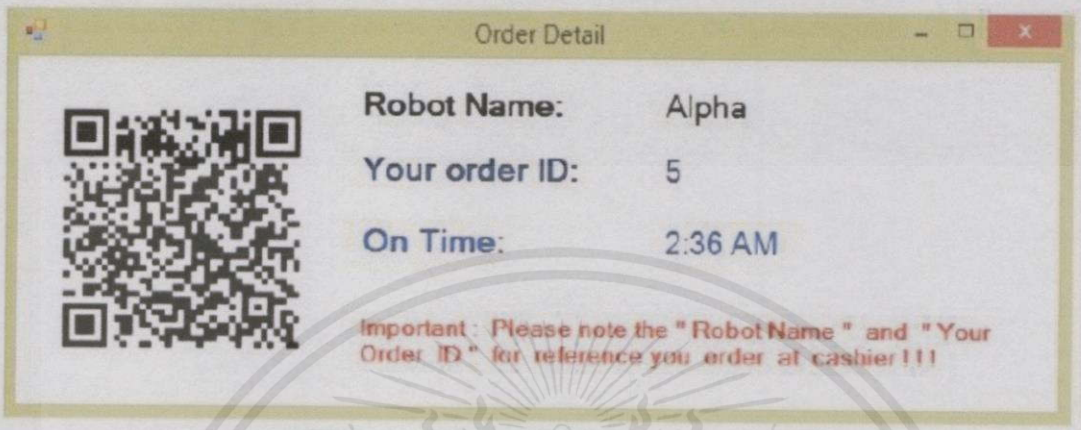
3. Cart สร้างหน้าต่าง Cart หรือตะกร้าสินค้าเพื่อดูรายการสินค้าในตะกร้า ซึ่งสามารถเพิ่ม ลดจำนวนสินค้า สั่งซื้อสินค้า หรือยกรายการสินค้าได้อีกครั้งเพื่อให้ผู้ใช้งานได้ทำการตรวจสอบให้แน่ใจอีกครั้งว่า สินค้าที่ผู้ใช้สั่งในรายการการสั่งซื้อมีความถูกต้องหรือสามารถเพิ่มหรือลดจำนวนของสินค้านั้นๆ ได้อีกครั้ง หากจำนวนสินค้าที่สั่งไปนั้นมีปริมาณไม่เพียงพอหรือมีปริมาณที่น้อยเกินไป แต่หากผู้ใช้ต้องการยกเลิกรายการทั้งหมดผู้ใช้งานสามารถคลิกที่ปุ่ม Clear All สินค้าทั้งหมดจะถูกลบออกจากใบสั่งรายการ แต่หากสินค้าและจำนวนมีความถูกต้องแล้วนั้นผู้ใช้สามารถกดปุ่ม Order Now เพื่อทำการสั่งซื้อสินค้าและชำระเงินในขั้นตอนสุดท้ายในการใช้งานระบบ



รูปที่ 3.24 หน้าจอแสดงการยืนยันการสั่งซื้อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. Order detail ออกแบบหน้าต่าง Order Detail เพื่อแสดงข้อมูลการสั่งซื้อแก่ลูกค้า โดยลูกค้าสามารถสแกน QR Code ผ่านทาง Smartphone โดยเป็นการเพิ่มความสะดวกและความหลากหลายให้กับลูกค้าผู้ใช้งานที่ต้องการทราบยอดรวม โดยไม่ต้องเก็บใบสั่งซื้อรายการเพื่อเป็นหลักฐานยืนยันการสั่งซื้อสินค้าหรือรายการต่างๆ ภายในระบบ



รูปที่ 3.25 หน้าจอ Order Detail

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การทดลองและผลการทดลอง

ในบทนี้จะกล่าวถึงการทดลองและผลการทดลองในการใช้งานโปรแกรม เพื่อใช้ในการวัดประสิทธิภาพในการใช้งานในโหมดการใช้เสียงของระบบร้านค้าอัจฉริยะ โดยจะใช้ระยะในการใช้คำสั่งเสียงและสภาพแวดล้อมของในแต่ละสถานที่เป็นตัววัดผลการทดลอง ว่าสถานที่ในลักษณะแบบใดและระยะการใช้งานระยะใดทำให้การทำงานของโปรแกรมมีประสิทธิภาพมากที่สุดโดยออกแบบการทดลองดังนี้

เนื่องจากการใช้คำสั่งเสียงในการใช้งานโปรแกรมนั้น จะมีประสิทธิภาพการทำงานแปลผันกับสภาพแวดล้อมของแต่ละสถานที่ที่ใช้ในการทดลอง โดยในแต่ละสถานที่จะมีลักษณะเสียงก้องหรือเสียงสะท้อนแตกต่างกันออกไป จึงจัดทำการทดลองว่าสถานที่ในลักษณะใดสามารถใช้งานเข้ากับโปรแกรมได้ดีที่สุด เพื่อนำไปพิจารณาเมื่อต้องใช้งานกับสถานที่จริง โดยศึกษาจากสภาพแวดล้อมของแต่ละสถานที่โดยสุ่มการเลือกผลิตภัณฑ์มา 5 ชนิดจากฐานข้อมูลทั้งหมด ดังนี้

1. ห้อง Lab ECC-208
2. สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
3. โรงอาหาร A คณะวิศวกรรมศาสตร์
4. โรงยิมเนเซียม
5. Top Supermarket

การทดลองแบ่งระยะการทดลองออกเป็น 3 ระยะ เพื่อทดสอบประสิทธิภาพในการประมวลผลของแต่ละระยะ การสั่งการด้วยเสียงว่าระยะใดมีประสิทธิภาพมากที่สุดโดยแบ่งออกเป็น 3 ระยะดังนี้

1. ระยะ 1 เมตร
2. ระยะ 1.5 เมตร
3. ระยะ 2 เมตร

โดยกำหนดให้เลข 1 ในตารางผลการทดลองคือ การแสดงผลของโปรแกรมที่แสดงผลถูกต้อง และเลข 0 ในตารางผลการทดลองคือ การแสดงผลของโปรแกรมที่แสดงผลผิดพลาด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1 การทดลองสถานที่ห้อง Lab ECC-208

ตารางที่ 4.1 สถานที่ห้อง Lab ECC-208

การทดลองครั้งที่	ผลที่วัดได้ของระยะในการใช้คำสั่งเสียง(เมตร)ต่อ 5 ผลิตภัณฑ์														
	1					1.5					2				
1	1	1	1	1	0	1	1	1	1	0	1	1	0	1	1
2	1	1	0	1	1	1	1	0	1	1	0	1	1	1	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
5	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1
6	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1
7	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
9	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1
10	1	1	1	1	1	1	1	0	1	0	0	0	0	1	0
ค่าความผิดพลาด (จำนวนครั้ง)	0	0	1	0	1	0	0	4	0	3	4	6	5	3	4
ค่าความผิดพลาด(%)	0	0	10	0	10	0	0	40	0	30	40	60	50	30	40
ค่าความผิดพลาดเฉลี่ย	4					14					44				
ค่าความแม่นยำเฉลี่ย	96					86					56				

จากการทดลองสถานที่ห้อง Lab ECC-208 ความแม่นยำที่ทดลองได้คิดเป็นเปอร์เซ็นต์ได้ผลการทดลองดังนี้

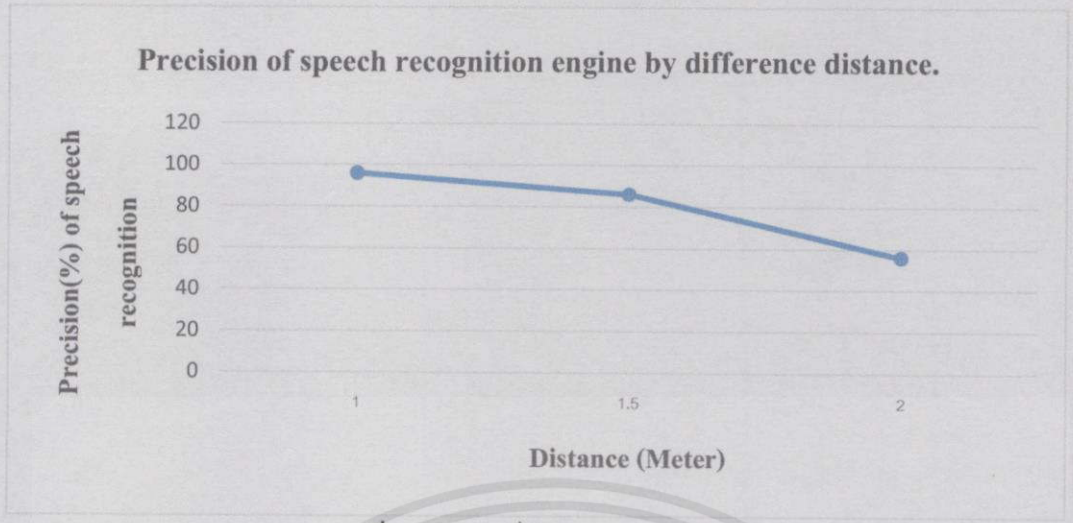
ระยะ 1 เมตร เปอร์เซ็นต์ความแม่นยำ = 96 %

ระยะ 1.5 เมตร เปอร์เซ็นต์ความแม่นยำ = 86 %

ระยะ 2 เมตร เปอร์เซ็นต์ความแม่นยำ = 56 %

กราฟแสดงความแม่นยำในการทำงานของโปรแกรม (%) เทียบกับระยะการสั่งงานในแต่ละระยะ (เมตร) ดังรูปที่ 4.1 โดยนำค่าเปอร์เซ็นต์ความแม่นยำและระยะทางจากตารางการทดลองมาใช้แสดงผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.1 สถานที่ห้อง Lab ECC-208

4.2 การทดลองสถานที่สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ตารางที่ 4.2 สถานที่สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองครั้งที่	ผลที่วัดได้ของระยะในการใช้ค่าสั่งเสียง(เมตร)ต่อ 5 ผลิตภัณฑ์														
	1					1.5					2				
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
3	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1
4	1	1	1	1	1	1	1	0	1	1	1	1	0	0	1
5	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0
6	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1
7	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1
8	1	1	1	1	0	1	1	1	1	0	0	1	1	0	1
9	1	1	1	1	1	1	1	0	1	1	1	0	0	0	1
10	1	1	1	0	1	1	1	1	0	1	1	1	0	1	0
ค่าความผิดพลาด (จำนวนครั้ง)	0	0	0	1	1	0	0	3	2	1	2	3	5	4	2
ค่าความผิดพลาด (%)	0	0	0	10	10	0	0	30	20	10	20	30	50	40	20
ค่าความผิดพลาดเฉลี่ย	4					12					32				
ค่าความแม่นยำเฉลี่ย	96					88					68				

จากการทดลองสถานที่สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ความแม่นยำที่ทดลองได้คิดเป็นเปอร์เซ็นต์ได้ผลการทดลองดังนี้

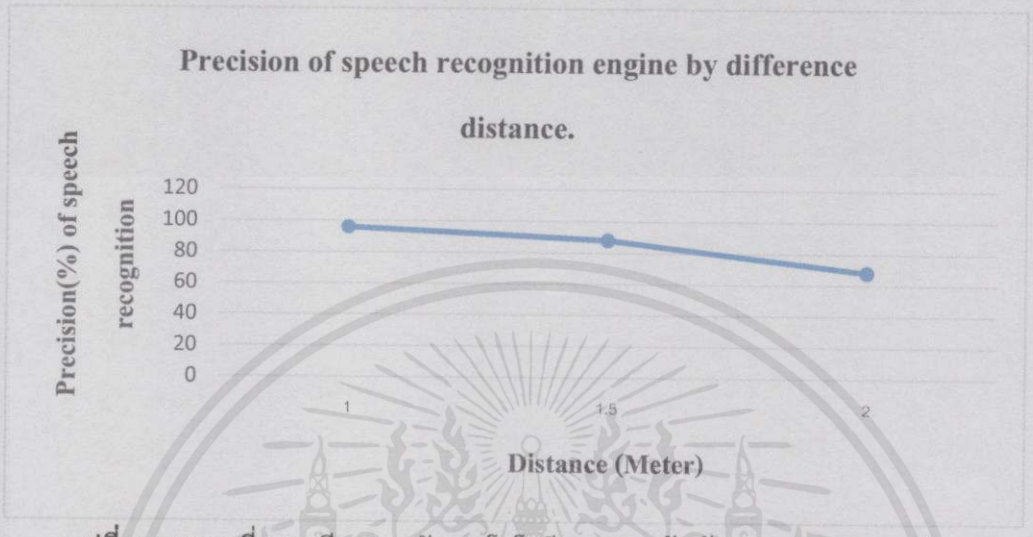
ระยะ 1 เมตร เปอร์เซ็นต์ความแม่นยำ = 96 %

ระยะ 1.5 เมตร เปอร์เซ็นต์ความแม่นยำ = 88 %

ระยะ 2 เมตร เปอร์เซ็นต์ความแม่นยำ = 68 %

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กราฟแสดงความแม่นยำในการทำงานของโปรแกรม(%)เทียบกับระยะการสั่งงานในแต่ละระยะ (เมตร) ดังรูปที่ 4.2 โดยนำค่าเปอร์เซ็นต์ความแม่นยำและระยะทางจากตารางการทดลองมาใช้แสดงผล



รูปที่ 4.2 สถานที่สนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

4.3 การทดลองสถานที่โรงอาหาร A คณะวิศวกรรมศาสตร์

ตารางที่ 4.3 สถานที่ โรงอาหาร A คณะวิศวกรรมศาสตร์

การทดลองครั้งที่	ผลที่วัดได้ของระยะในการใช้ค่าสั่งเสียง(เมตร)ต่อ 5 ผลิตภัณฑ์														
	1					1.5					2				
1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	0	1	0	1	1	0	0	1
4	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1
5	1	1	1	1	1	1	0	1	0	1	1	1	0	1	0
6	1	1	0	1	1	1	1	0	1	1	1	1	1	0	1
7	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0
8	1	1	1	1	1	1	1	1	1	0	1	0	0	1	1
9	1	1	1	0	1	1	0	1	1	1	1	0	1	0	1
10	1	1	1	1	1	1	1	0	1	1	0	1	0	1	1
ค่าความผิดพลาด (จำนวนครั้ง)	0	0	1	1	0	0	2	3	1	2	3	4	5	3	3
ค่าความผิดพลาด (%)	0	0	10	10	0	0	20	30	10	20	30	40	50	30	30
ค่าความผิดพลาดเฉลี่ย	4					16					36				
ค่าความแม่นยำเฉลี่ย	96					84					64				

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

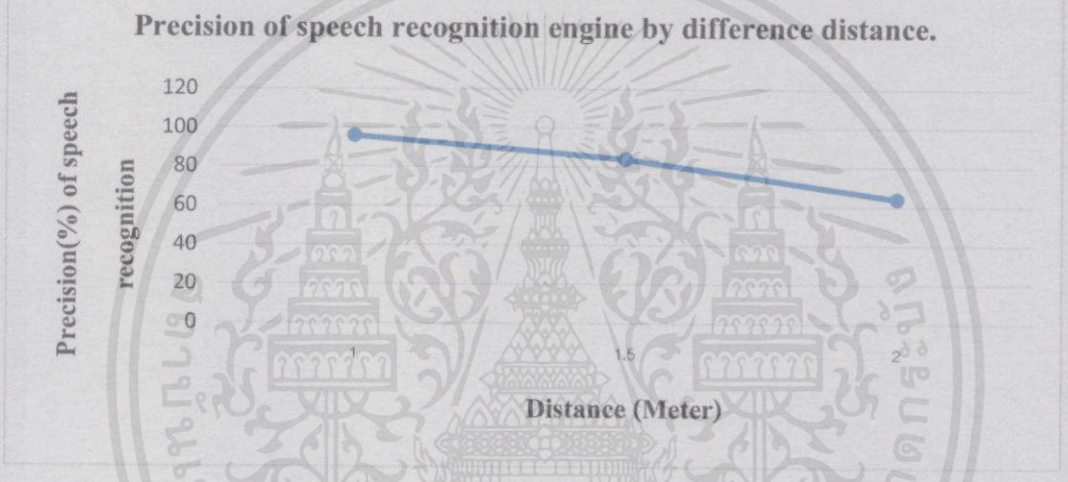
จากการทดลองสถานที่โรงอาหาร A คณะวิศวกรรมศาสตร์ ความแม่นยำที่ทดลองได้คิดเป็นเปอร์เซ็นต์ได้ผลการทดลองดังนี้

ระยะ 1 เมตร เปอร์เซ็นต์ความแม่นยำ = 96 %

ระยะ 1.5 เมตร เปอร์เซ็นต์ความแม่นยำ = 84 %

ระยะ 2 เมตร เปอร์เซ็นต์ความแม่นยำ = 64 %

กราฟแสดงความแม่นยำในการทำงานของโปรแกรม (%) เทียบกับระยะการสั่งงานในแต่ละระยะ (เมตร) ดังรูปที่ 4.3 โดยนำค่าเปอร์เซ็นต์ความแม่นยำและระยะทางจากตารางการทดลองมาใช้แสดงผล



รูปที่ 4.3 สถานที่โรงอาหาร A คณะวิศวกรรมศาสตร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 การทดลองสถานที่โรงยิมเนเซียม

ตารางที่ 4.4 สถานที่โรงยิมเนเซียม

การทดลองครั้งที่	ผลที่วัดได้ของระยะในการใช้ค่าสั่งเสียง(เมตร)ต่อ 5 ผลิตภัณฑ์														
	1					1.5					2				
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
2	1	1	0	1	1	1	1	1	1	1	0	0	0	0	0
3	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
6	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
7	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0
9	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
10	1	1	1	1	1	0	1	1	1	1	0	0	0	0	0
ค่าความผิดพลาด (จำนวนครั้ง)	0	0	1	0	0	1	1	0	1	1	10	10	10	10	10
ค่าความผิดพลาด(%)	0	0	10	0	0	10	10	0	10	10	100	100	100	100	100
ค่าความผิดพลาดเฉลี่ย	2					8					100				
ค่าความแม่นย่าเฉลี่ย	98					92					0				

จากการทดลองสถานที่โรงยิมเนเซียม ความแม่นยำที่ทดลองได้คิดเป็นเปอร์เซ็นต์ได้ผลการทดลองดังนี้

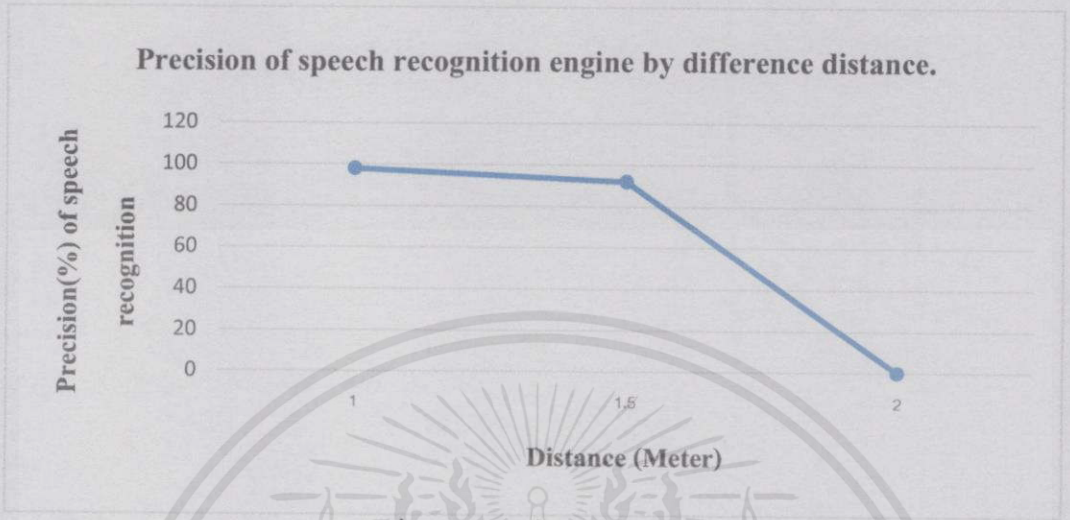
ระยะ 1 เมตร เปอร์เซ็นต์ความแม่นยำ = 98 %

ระยะ 1.5 เมตร เปอร์เซ็นต์ความแม่นยำ = 92 %

ระยะ 2 เมตร เปอร์เซ็นต์ความแม่นยำ = 0 %

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กราฟแสดงความแม่นยำในการทำงานของโปรแกรม (%) เทียบกับระยะการสั่งงานในแต่ละระยะ (เมตร) ดังรูปที่ 4.4 โดยนำค่าเปอร์เซ็นต์ความแม่นยำและระยะทางจากตารางการทดลองมาใช้แสดงผล



รูปที่ 4.4 สถานที่โรงยิมเนเซียม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5 การทดลองสถานที่ Top Supermarket

ตารางที่ 4.5 สถานที่ Top Supermarket

การทดลองครั้งที่	ผลที่วัดได้ของระยะในการใช้คำสั่งเสียง(เมตร)ต่อ 5 ผลิตภัณฑ์														
	1					1.5					2				
1	1	1	1	1	1	1	1	0	1	1	1	0	0	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0
3	1	1	0	1	1	1	1	1	1	1	0	1	0	1	1
4	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1
5	1	1	1	1	1	1	1	1	1	1	0	1	0	0	1
6	1	1	1	1	1	1	1	1	1	0	1	1	1	0	0
7	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0
8	1	1	1	0	1	1	0	0	1	1	1	1	1	1	0
9	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
10	1	1	1	1	1	1	1	0	1	0	0	0	1	0	1
ค่าความผิดพลาด (จำนวนครั้ง)	0	0	1	1	0	0	1	3	1	1	4	4	5	4	5
ค่าความผิดพลาด (%)	0	0	10	10	0	0	10	30	10	10	40	40	50	40	50
ค่าความผิดพลาดเฉลี่ย	4					12					44				
ค่าความแม่นยำเฉลี่ย	96					88					56				

จากการทดลองสถานที่ Top Supermarket ความแม่นยำที่ทดลองได้คิดเป็นเปอร์เซ็นต์ ได้ผลการทดลองดังนี้

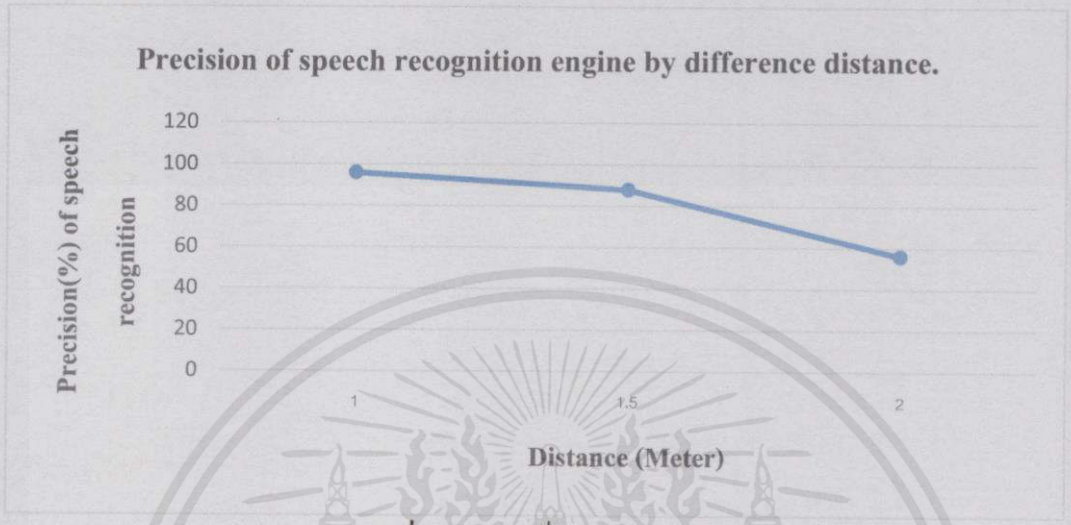
ระยะ 1 เมตร เปอร์เซ็นต์ความแม่นยำ = 96 %

ระยะ 1.5 เมตร เปอร์เซ็นต์ความแม่นยำ = 88 %

ระยะ 2 เมตร เปอร์เซ็นต์ความแม่นยำ = 56 %

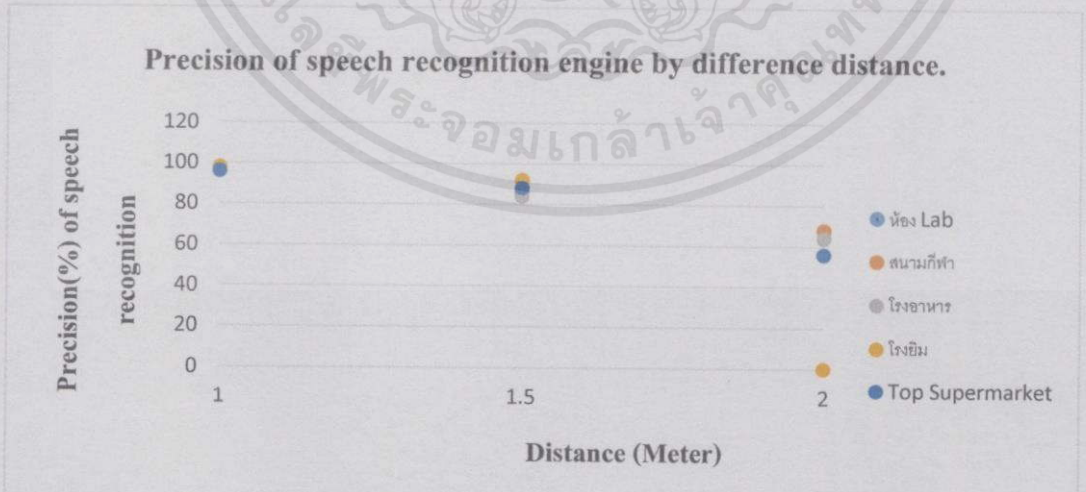
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กราฟแสดงความแม่นยำในการทำงานของโปรแกรม (%) เทียบกับระยะการสั่งงานในแต่ละระยะ (เมตร) ดังรูปที่ 4.5 โดยนำค่าเปอร์เซ็นต์ความแม่นยำและระยะทางจากตารางการทดลองมาใช้แสดงผล



รูปที่ 4.5 สถานที่ Top Supermarket

กราฟแสดงความแม่นยำในการทำงานของโปรแกรม (%) เทียบกับระยะการสั่งงานในแต่ละระยะ (เมตร) ดังรูปที่ 4.6 โดยนำค่าเปอร์เซ็นต์ความแม่นยำและระยะทางจากตารางการทดลองมาใช้แสดงผล



รูปที่ 4.6 สรุปการทดลองในแต่ละระยะของสถานที่การทดลองทั้งหมด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สรุปผลการทดลองทั้ง 5 สถานที่ได้ความแม่ยำในแต่ละระยะการสั่งงานด้วยเสียงในแต่ละระยะดังนี้

ระยะที่ 1 เมตร เปอร์เซ็นต์ความแม่นยำ = 96 %

ระยะที่ 1.5 เมตร เปอร์เซ็นต์ความแม่นยำ = 87.6 %

ระยะที่ 2 เมตร เปอร์เซ็นต์ความแม่นยำ = 50.8 %

จากการทดลองทั้งหมดแสดงให้เห็นว่าสถานที่ และระยะการทดลองที่ทำให้การใช้คำสั่งเสียงในการใช้งานระบบร้านค้าอัจฉริยะคือ พื้นที่ที่มีลักษณะแบบห้อง Lab ECC-208 และ Top Supermarket โดยระยะการใช้งานในการใช้คำสั่งเสียงใช้งานโปรแกรมระยะ 1 เมตร ซึ่งจะทำให้โปรแกรมของระบบร้านค้าอัจฉริยะมีประสิทธิภาพสูงที่สุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

บทวิจารณ์และสรุป

5.1 สรุป

จากการออกแบบการทดลองในการใช้งานคำสั่งเสียงในการสั่งงานระบบร้านค้าอัจฉริยะใน Voice Mode เพื่อนำไปพัฒนาโปรแกรมให้สามารถใช้งานได้อย่างมีประสิทธิภาพและแม่นยำมากที่สุด พบว่าลักษณะของสภาพแวดล้อมในการวางอุปกรณ์ ระยะในการออกคำสั่งและรับคำสั่งของอุปกรณ์ มีส่วนสำคัญเป็นอย่างมากในการประมวลผลของโปรแกรมเพื่อแสดงรายการสินค้าอย่างถูกต้อง จะเห็นว่าในการวางอุปกรณ์ไว้ในพื้นที่ที่มีลักษณะเป็นห้องแคบจนเกินไป เช่น ห้อง Lab ECC-208 การประมวลผลของโปรแกรมและการแสดงผลของโปรแกรมในแต่ละระยะการทดลองจะเกิดความผิดพลาดบ้าง เนื่องจากลักษณะห้องจะทำให้เกิดเสียงก้อง และเสียงสะท้อน ซึ่งทำให้อุปกรณ์รับคำสั่งเสียงเกิดความผิดพลาดในบางครั้ง ซึ่งลักษณะเสียงก้องนี้จะส่งผลกระทบอย่างมากเมื่อลักษณะพื้นที่ที่จะใช้งานมีความกว้างและใหญ่มาก ซึ่งสังเกตได้จากการทดลองในโรงยิมเนเซียม ซึ่งโปรแกรมจะไม่สามารถแสดงผลอย่างถูกต้อง และเมื่อเพิ่มระยะในการสั่งงานด้วยคำสั่งเสียงก็จะทำให้โปรแกรมไม่สามารถแสดงผลได้อย่างถูกต้องเลยแม้แต่ผลิตภัณฑ์เดียว ซึ่งเมื่อเปลี่ยนจากสถานที่ที่ปิดมิดชิดและมีความกว้างแบบโรงยิมเนเซียม ไปเป็นสถานที่ที่มีความโล่งกว้างแบบสนามกีฬาสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง และโรงอาหาร A คณะวิศวกรรมศาสตร์ จะพบว่าการใช้งานด้วย Voice Mode มีความแม่นยำในแต่ละระยะเพิ่มมากขึ้น ซึ่งสามารถสรุปได้ว่าพื้นที่ที่มีลักษณะเปิดกว้างจะสามารถใช้งานคำสั่งเสียงได้ดีกว่าในห้องหรือสถานที่ที่มีลักษณะปิด และเมื่อได้นำโปรแกรมของระบบร้านค้าอัจฉริยะไปทดลองใน Top Supermarket เพื่อทดสอบการใช้งานการสั่งสินค้าด้วย Voice Mode ในสถานที่ที่มีการใช้งานจริง พบว่าโปรแกรมสามารถประมวลผลและแสดงค่าได้แม่นยำที่สุด โดยประสิทธิภาพการใช้งานด้วย Voice Mode ในแต่ละระยะการใช้งานยังมีความแม่นยำมากกว่าลักษณะพื้นที่ทั้งหมดที่ได้มีการทดลองอีกด้วย ซึ่งระยะที่ใช้ในการออกคำสั่งที่มีความแม่นยำมากที่สุดคือระยะ 1 เมตร ซึ่งพบว่าในแต่ละสถานที่ระยะ 1 เมตรเป็นระยะที่โปรแกรมสามารถแสดงค่าที่ถูกต้องและแม่นยำที่สุด

5.2 ปัญหาที่พบในโครงงาน

เนื่องจากการจัดทำโครงงานนี้จำเป็นต้องศึกษาพื้นฐานการจัดทำส่วนของ User Interface เพื่อการออกแบบใหม่ทั้งหมดและต้องศึกษาภาษาต่างๆ ที่ต้องนำมาใช้ร่วมกับระบบร้านค้าอัจฉริยะ รวมทั้งการวางโครงสร้างการจัดการในร้านขึ้นมาใหม่ทั้งหมด จึงทำให้ต้องเริ่มศึกษาโปรแกรมต่าง ๆ โดยเริ่มจากพื้นฐานทั้งหมดเพื่อจะนำมาประยุกต์ใช้กับโปแกรมซึ่งในแต่ละโปแกรมหรือภาษาต่าง ๆ นั้นมีความยากง่ายแตกต่างกันออกไป จึงทำให้ระยะเวลาในการศึกษาโปรแกรมต่างๆ มีระยะเวลาที่ต้องใช้ในการศึกษามากพอสมควร ประกอบด้วยระยะเวลาในการทำโครงงานระบบร้านค้าอัจฉริยะนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

มีขอบเขตเวลาที่กำหนด จึงทำให้การทำงานในช่วงศึกษาโปรแกรมหรือภาษาต่างๆ มีความล่าช้าในช่วงเริ่มต้นการศึกษา

5.3 ผลที่คาดว่าจะได้รับในโครงงาน

ในการจัดทำโครงงานระบบร้านค้าอัจฉริยะนี้ต้องนำความรู้ในแต่ละด้านมาประยุกต์ใช้เข้าด้วยกัน โดยในแต่ละภาษาหรือโครงสร้างต่างๆ ภายในโปรแกรมล้วนแล้วแต่มีเอกลักษณ์ หรือการทำงานที่แตกต่างกันออกไป จึงทำให้โครงงานเกิดทักษะในการนำจุดเด่นต่างๆ ของแต่ละโปรแกรมหรือแต่ละภาษามาใช้งานร่วมกันอย่างมีประสิทธิภาพและมีความสมบูรณ์มากที่สุด อีกทั้งยังรวมไปถึงการฝึกฝนการทำงานร่วมกันเป็นหมู่คณะ ซึ่งทั้งหมดนี้ทำให้โครงงานได้ฝึกฝนทักษะและความรู้ใหม่ในการนำไปใช้สำหรับการทำงานในอนาคต การคิดวิเคราะห์วางแผนในการทำงานต่างๆ และสามารถนำไปใช้ในการประกอบอาชีพในอนาคตอย่างมีประสิทธิภาพ

5.4 แนวทางในการพัฒนา และการประยุกต์

ในส่วนของแนวทางในการพัฒนา และประยุกต์นั้น สามารถพัฒนาการใช้ Voice Mode โดยทำให้โปรแกรมสามารถรองรับการใช้งานเป็นภาษาไทย เพื่อง่ายต่อการใช้งานของประชาชนในส่วนที่ไม่ชอบการใช้งานด้วยภาษาอังกฤษ รวมทั้งยังสามารถนำระบบร้านค้าอัจฉริยะนี้ไปใช้งานได้จริงในการประกอบธุรกิจ ที่ต้องการเพิ่มทางเลือกในการซื้อสินค้าให้กับผู้บริโภคที่ต้องการความสะดวกสบาย และเป็นการริเริ่มการพัฒนาระบบร้านค้าอัจฉริยะนี้ให้มีความแม่นยำในการใช้งาน และสามรถทำงานร่วมกับระบบร้านค้าเดิมเพื่อทำให้ธุรกิจภายในประเทศมีการพัฒนาอย่างทันสมัยมากขึ้นอีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

[1] Lawrence Rabiner. **Fundamental of Speech Recognition**. englewood cliffs, new jersey : ptr prentice hall inc, 1993.

[2] **Speech Recognition**. (Online). แหล่งที่มา : http://msmisthammasat.blogspot.com/2010/11/speech-recognition.html 24 มกราคม2558

[3] **An Overview of Speech Synthesis and Recognition**. (Online). แหล่งที่มา : http://fileadmin.cs.lth.se/cs/Personal/Pierre_Nugues/ilppp/chapters/speech.pdf 24 มกราคม 2558

[4] **ภาษา SQL**. (Online). แหล่งที่มา : https://sites.google.com/site/piyanathw5505sql/ 4 กุมภาพันธ์ 2558

[5] **โปรแกรมจัดการฐานข้อมูล MySQL แบบฟรี**. (Online). แหล่งที่มา : http://meewebfree.com/site/start-website-builder/399-free-mysql-gui-like-navicat 4 กุมภาพันธ์ 2558

[6] **วิธีการใช้งาน MySQL Workbench**. (Online). แหล่งที่มา : http://www.mindphp.com/forums/viewtopic.php?f=29&t=11860 4 กุมภาพันธ์ 2558

[7] **phpMyAdmin กับการบริหารจัดการระบบฐานข้อมูล**. (Online). แหล่งที่มา : http://www.lamptech.ac.th/webprg/article_top/file_ar/a547539a183690%BA%B7%A4%C7%D2%C1%C7%D4%AA%D2%A1%D2%C3%20%CD.%BB%C3%C1%D4%B9%B7%C3%EC.pdf 1 กุมภาพันธ์ 2558

[8] **QR Code**. (Online). แหล่งที่มา : http://www.coe.or.th/e_engineers/knc_detail.php?id=169 7 กุมภาพันธ์ 2558

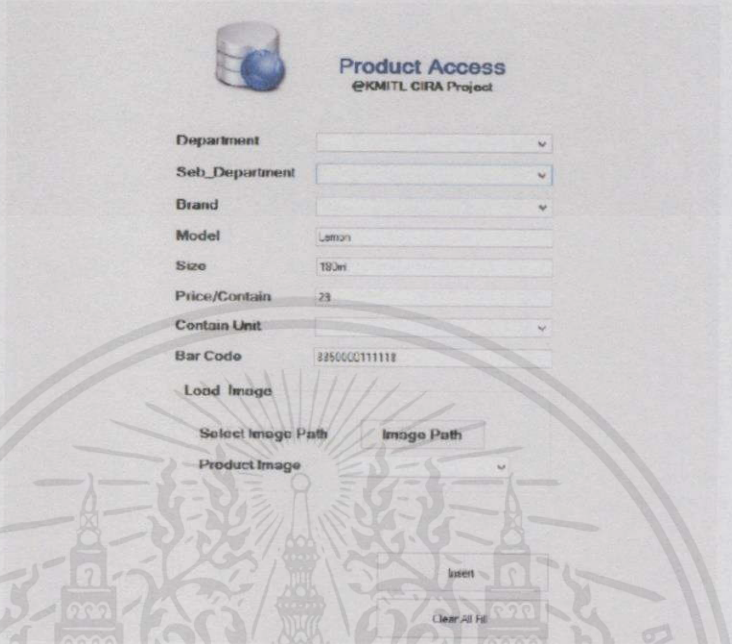
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก: Central Back Office Application

Add_Product.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;
using System.Collections;
using System.Diagnostics;

namespace DataAccess
{
    public partial class AddProduct : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;



        public AddProduct()
        {
            InitializeComponent();
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword ;
//boong3013
            logoPcb.Image = Properties.Resources.databaseIcon;
        }

        private void comboBox3_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (BrandCBB.Text!=null)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            {
                string SQL = "select distinct containunit from central_product where
brand='" + BrandCBB.Text + "';";
                MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
                DataTable DTB = new DataTable();
                DA.Fill(DTB);
                string[] mySubDept = DTB.AsEnumerable().Select(r =>
r.Field<string>("ContainUnit")).ToArray();
                ConunitCBB.Text = null;
                ConunitCBB.Items.Clear();
                ConunitCBB.Items.AddRange(mySubDept);
            }
        }

        private void ProductAccess_Load(object sender, EventArgs e)
        {
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }

            string SQL = "select distinct department from central_department;";
            MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
            DataTable DTB = new DataTable();
            DA.Fill(DTB);
            string[] myDept = DTB.AsEnumerable().Select(r =>
r.Field<string>("Department")).ToArray();
            departCBB.Items.AddRange(myDept);
        }

        public void button1_Click(object sender, EventArgs e)
        {
            FolderBrowserDialog imagePath = new FolderBrowserDialog();
            DialogResult result = imagePath.ShowDialog();
            globalVar.imagePATH = imagePath.SelectedPath;
            string[] filters = new[] { "*.jpg", "*.png", "*.gif", "*.jpeg", "*.bmp" };
            string[] filePaths = filters.SelectMany(f =>
Directory.GetFiles(globalVar.imagePATH, f)).ToArray();
            int imageLength = filePaths.Length;
            for (int addImgCount = 0; addImgCount < imageLength; addImgCount++)
            {
                string filename = Path.GetFileName(filePaths[addImgCount]);
                selectImgCBB.Items.Add(filename);

            }
        }

        private void insertBtn_Click(object sender, EventArgs e)
        {

            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();

            }

            //Add into Central_Department<TABLE>
            if(subDepartCBB.FindStringExact(subDepartCBB.Text)==-1)
            {
                connection.Open();
                string sqlDepart = "INSERT INTO
central_department(Sub_Department,Department)";
                sqlDepart += " VALUES (" + "'" + subDepartCBB.Text + "'," + "'" +
departCBB.Text + "'" + ");";
                SQLCommand = new MySqlCommand(sqlDepart, connection);
                SQLCommand.ExecuteNonQuery();
                connection.Close();
            }

            connection.Open();
            string sqlSubDepart = "INSERT INTO
central_product(BarCode,Sub_Department,Brand,Model,Size,ContainUnit,PricePerContain,Pr
oductImage)";
```

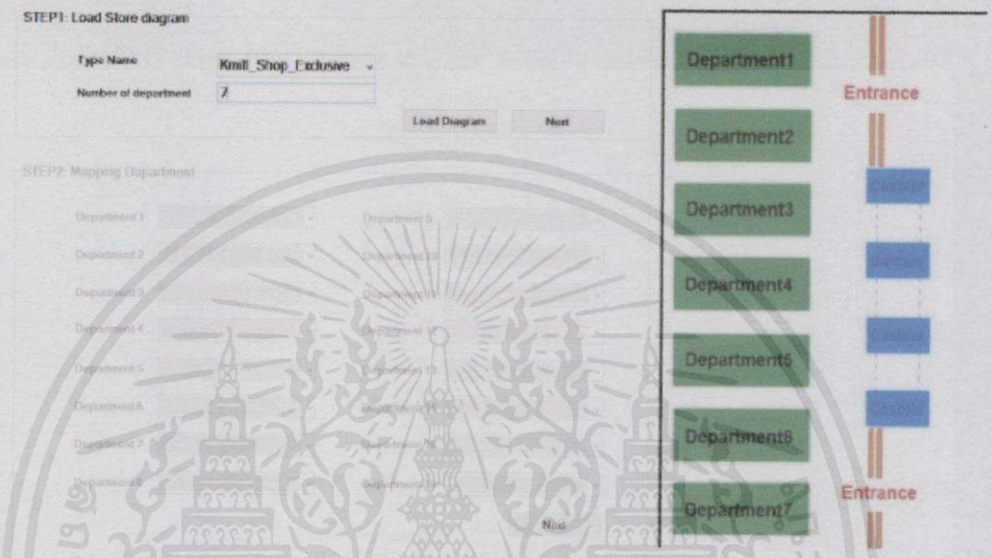
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            sqlSubDepart += " VALUES (" + "'" + barcodeTxt.Text + "'," + "'" +
 subDepartCBB.Text + "'," + "'" + BrandCBB.Text + "'," + "'" + modelTxt.Text + "'," +
 "'" + sizeTxt.Text + "'," + "'" + ConunitCBB.Text + "'," + "'" + pricePerUintTxt.Text
 + "'," + "load_file('" + globalVar.selectImagePath.Replace("\\", "/") + "')" + ");";
            SQLCommand = new MySqlCommand(sqlSubDepart, connection);
            SQLCommand.ExecuteNonQuery();
            connection.Close();
            sqlSubDepart = null;

            string SQL = "select distinct department from central_department;";
            MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
            DataTable DTB = new DataTable();
            DA.Fill(DTB);
            string[] myDept = DTB.AsEnumerable().Select(r =>
 r.Field<string>("Department")).ToArray();
            departCBB.Text = null;
            departCBB.Items.Clear();
            departCBB.Items.AddRange(myDept);


        }

        private void selectImgCBB_SelectedIndexChanged(object sender, EventArgs e)
        {
            string imagefile = selectImgCBB.Text;
            if (imagefile != "")
            {
                globalVar.selectImagePath = globalVar.imagePATH + "\\" + imagefile;
                productImgPB.Image = new Bitmap(globalVar.selectImagePath);
            }

        }

        private void clearAllFillBtn_Click(object sender, EventArgs e)
        {
            departCBB.Text = null;
            subDepartCBB.Text = null;
            BrandCBB.Text = null;
            modelTxt.Text = null;
            sizeTxt.Text = null;
            barcodeTxt.Text = null;
            pricePerUintTxt.Text = null;
            productImgPB.Image.Dispose();
            productImgPB.Image = null;
            selectImgCBB.Text = null;
            ConunitCBB.Text = null;

        }

        private void departCBB_SelectedIndexChanged(object sender, EventArgs e)
        {
            string SQL = "select distinct sub_department from central_department where
 department='" + departCBB.Text + "';";
            MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
            DataTable DTB = new DataTable();
            DA.Fill(DTB);
            string[] mySubDept = DTB.AsEnumerable().Select(r =>
 r.Field<string>("sub_Department")).ToArray();
            subDepartCBB.Text = null;
            subDepartCBB.Items.Clear();
            subDepartCBB.Items.AddRange(mySubDept);
        }

        private void subDepartCBB_SelectedIndexChanged(object sender, EventArgs e)
        {
            if(subDepartCBB.Text!=null)
            {
                string SQL = "select distinct brand from central_product where
 sub_department='" + subDepartCBB.Text + "';";
                MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
                DataTable DTB = new DataTable();
                DA.Fill(DTB);
                string[] mySubDept = DTB.AsEnumerable().Select(r =>
 r.Field<string>("Brand")).ToArray();
                BrandCBB.Text = null;
                BrandCBB.Items.Clear();
                BrandCBB.Items.AddRange(mySubDept);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            }
        }
    }
}
```

Add_Shelves.cs




```
using System;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;
using System.Speech.Synthesis;

namespace DataAccess
{
    public partial class AddShelves : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;

        public AddShelves()
        {
            InitializeComponent();
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }

        private void Access_Shelves_Load(object sender, EventArgs e)
        {
            step2NextBtn.Enabled = false;
            step2GBox.Enabled = false;
            for (int i = 1; i <= 16; i++ )
            {

                string nameCtrl = "select_department"+Convert.ToString(i) + "CBB";
                Control[] output = this.Controls.Find(nameCtrl, true);
                string size = Convert.ToString(output.Length);
                if (output != null)
                {
                    output[0].Enabled = false;
                }


            }
            step3NextBtn.Enabled = false;
            step3NextBtn.Enabled = false;
            for (int i = 1; i <= 16; i++)
            {

                string nameCtrl = "DeptGRB" + Convert.ToString(i);
                Control[] output = this.Controls.Find(nameCtrl, true);
                string size = Convert.ToString(output.Length);
                if (output != null)
                {
                    output[0].Enabled = false;
                }

            }
            step4selectDeptGBB.Enabled = false;
            step4ShelveGBB.Enabled = false;
            for (int i = 1; i <= 3; i++)
            {

                string nameCtrl = "shelveLV" + Convert.ToString(i)+"GBB";
                Control[] output = this.Controls.Find(nameCtrl, true);
                string size = Convert.ToString(output.Length);
                if (output != null)
                {
                    output[0].Enabled = false;
                }

            }
        }

        private void button1_Click(object sender, EventArgs e)
        {
            OpenFileDialog storeDiagramDialog = new OpenFileDialog();
            storeDiagramDialog.Filter = "Image files (*.jpg, *.jpeg, *.jpe, *.jfif,
*.png) | *.jpg; *.jpeg; *.jpe; *.jfif; *.png";
            DialogResult storeDiagram = storeDiagramDialog.ShowDialog();
            globalVar.storeDiagramPATH = storeDiagramDialog.FileName;
            diagramPcb.Image = new Bitmap(globalVar.storeDiagramPATH);

        }

        #region Add Department NO.1 - 16

        private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
        {

            select_department2CBB.Items.AddRange(globalVar.loadDept2CBBStep2);

        }

        private void comboBox2_SelectedIndexChanged_1(object sender, EventArgs e)
        {
            select_department3CBB.Items.AddRange(globalVar.loadDept2CBBStep2);

        }

        private void comboBox12_SelectedIndexChanged(object sender, EventArgs e)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ ดัดแปลง เผยแพร่ หรือแสวงหาประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ลบเปลี่ยนแปลง และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {
            select_department13CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void comboBox13_SelectedIndexChanged(object sender, EventArgs e)
        {
            select_department14CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void comboBox4_SelectedIndexChanged(object sender, EventArgs e)
        {
            select_department5CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void comboBox7_SelectedIndexChanged(object sender, EventArgs e)
        {
            select_department8CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }


        private void select_department3CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department4CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department5CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department6CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department6CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department7CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department8CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department9CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department9CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department10CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department10CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department11CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department11CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department12CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department14CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department15CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department15CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            select_department16CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        private void select_department16CBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
```

เอกสารนี้จัดทำขึ้นสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            select_department16CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
        }

        #endregion

        private void step2NextBtn_Click(object sender, EventArgs e)
        {
            globalVar.departmentName = new string[globalVar.numOfDepartment];
            for (int i = 1; i <= globalVar.numOfDepartment; i++)
            {

                string controlName = "select_department" + Convert.ToString(i) +
"CBB";
                Control[] output = this.Controls.Find(controlName, true);
                if (output != null)
                {
                    globalVar.departmentName[i - 1] = output[0].Text;

                }
            }
            if
((globalVar.departmentName.Distinct().Count())!=(globalVar.departmentName.Count()))
            {

                globalVar.departmentName = null;
                MessageBox.Show("Duplicate value was found in department ComboBox",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

            }
            else
            {
                globalVar.sub_departmentLength = new int[globalVar.numOfDepartment];
                connection.Open();
                for (int i = 1; i <= globalVar.numOfDepartment; i++)
                {

                    string controlName = "DeptGRB" + Convert.ToString(i);
                    Control[] output = this.Controls.Find(controlName, true);
                    if (output != null)
                    {

                        output[0].Enabled=true;
                        output[0].Text = globalVar.departmentName[i - 1];
                        string SQL = "select distinct sub_department from
central_department where department='" + globalVar.departmentName[i-1] + "';";
                        MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
                        DataTable DTB = new DataTable();
                        DA.Fill(DTB);
                        string[] mySubDept = DTB.AsEnumerable().Select(r =>
r.Field<string>("sub_department")).ToArray();
                        globalVar.sub_departmentLength[i-1] = mySubDept.Length;
                        for (int j = 1; j <= globalVar.sub_departmentLength[i - 1];
j++)
                        {
                            string step3CLB = "Dept" + Convert.ToString(i) + "Sub" +
Convert.ToString(j) + "Clb";
                            Control[] fillstep3CLB = this.Controls.Find(step3CLB,
true);
                            if (fillstep3CLB != null)
                            {
                                fillstep3CLB[0].Text = mySubDept[j - 1];
                                fillstep3CLB[0].Visible = true;
                            }
                        }
                    }
                }
                tabControl1.SelectedTab = tabPage2;
                step3NextBtn.Enabled = true;
                connection.Close();
            }
            connection.Open();
            string sqlDepart = "INSERT INTO
Branch_template(Store_Type,Store_Diagram,Amount_Department)";
            sqlDepart += " VALUES (" + "'" + globalVar.typeName + "'," + "load_file('"
+ globalVar.storeDiagramPATH.Replace("\\", "/") + "')," + "'" +
Convert.ToString(globalVar.numOfDepartment) + "'" + ");";
            SQLCommand = new MySqlCommand(sqlDepart, connection);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            SQLCommand.ExecuteNonQuery();
            connection.Close();
        }

        private void checkBox3_CheckedChanged(object sender, EventArgs e)
        {

        }

        private void step1NextBtn_Click(object sender, EventArgs e)
        {


if((typeNameTxt.Text!=null)&&(numberOfTxt.Text!="")&&(globalVar.storeDiagramPATH!=null
))
            {
                globalVar.numOfDepartment = Convert.ToInt32(numberOfTxt.Text);
                step2GBox.Enabled = true;
                step2NextBtn.Enabled = true;
                globalVar.typeName = typeNameTxt.Text;
                connection = new MySqlConnection(globalVar.mysqlConnectionString);
                if (connection.State == ConnectionState.Open)
                {
                    Console.Beep();
                    connection.Close();
                }
                connection.Open();
                string SQL = "select distinct department from central_department;";
                MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
                DataTable DTB = new DataTable();
                DA.Fill(DTB);
                globalVar.loadDept2CBBStep2 = DTB.AsEnumerable().Select(r =>
r.Field<string>("Department")).ToArray();
                select_department1CBB.Items.AddRange(globalVar.loadDept2CBBStep2);
                connection.Close();

                for (int i = 1; i <= globalVar.numOfDepartment; i++)
                {
                    string nameCtrl = "select_department" + Convert.ToString(i) +
"CBB";
                    Control[] output = this.Controls.Find(nameCtrl, true);
                    string size = Convert.ToString(output.Length);
                    if (output != null)
                    {
                        output[0].Enabled = true;

                    }
                }
            }

        }

        private void button1_Click_1(object sender, EventArgs e)
        {
            tabControl1.SelectedTab = tabPage3;
            shelvesOverviewPcb.Image = Properties.Resources.ShelveMini;
            shelveSlotPCB.Image = Properties.Resources.ShelveSlotMini;
            step4selectDeptGBB.Enabled = true;
            step4DepartmentCBB.Items.Clear();
            step4DepartmentCBB.Text = null;
            step4Sub_DepartmentCBB.Items.Clear();
            step4Sub_DepartmentCBB.Text = null;
            step4DepartmentCBB.Items.AddRange(globalVar.departmentName);
            globalVar.sub_departmentName = new
List<string>[globalVar.numOfDepartment];
            for (int i = 1; i <= globalVar.numOfDepartment; i++)
            {
                string nameCtrl = "DeptGRB" + Convert.ToString(i);
                Control[] output = this.Controls.Find(nameCtrl, true);
                if (output != null)
                {
                    globalVar.sub_departmentName[i-1] = new List<string>();
                    foreach (Control checkbox in output[0].Controls)
                    {
                        if (checkbox is CheckBox)
                        {
                            int currentLength = globalVar.sub_departmentLength.Length;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

```
                            CheckBox temp = (CheckBox)checkbox;
                            if (temp.Checked)
                            {
                                globalVar.sub_departmentName[i-1].Add(checkbox.Text);
                            }
                        }
                    }
                }
            }
        }

        private void step4DepartmentCBB_SelectedIndexChanged(object sender, EventArgs
e)
        {
            step4Sub_DepartmentCBB.Text = null;
            step4Sub_DepartmentCBB.Items.Clear();
            int departIndx=step4DepartmentCBB.SelectedIndex;

step4Sub_DepartmentCBB.Items.AddRange(globalVar.sub_departmentName[departIndx].ToArray
());
        }

        private void step4Sub_DepartmentCBB_SelectedIndexChanged(object sender,
EventArgs e)
        {
            step4ShelveGBB.Enabled = true;
            shelveNumTxt.Text = Convert.ToString(globalVar.shelveNumber);
            string SQL = "select BarCode,Brand,Model,Size from central_product where
sub_department='" + step4Sub_DepartmentCBB.Text + "';";
            MySqlDataAdapter DA = new MySqlDataAdapter(SQL, connection);
            DataTable DTB = new DataTable();
            DA.Fill(DTB);
            globalVar.BarCodeAdd = DTB.AsEnumerable().Select(r =>
r.Field<string>("BarCode")).ToArray();
            globalVar.BrandAdd = DTB.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
            globalVar.ModelAdd = DTB.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
            globalVar.SizeAdd = DTB.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
            globalVar.productAdd = new string[globalVar.BrandAdd.Length];
            for (int pointer = 0; pointer <= globalVar.BrandAdd.Length - 1;pointer++ )
            {
                globalVar.productAdd[pointer] = globalVar.BrandAdd[pointer] + "  " +
globalVar.ModelAdd[pointer] + "(" + globalVar.SizeAdd[pointer] + ")";
            }
                for (int i = 1; i <= 3; i++)
                {
                    string nameCtrl = "shelveLV" + Convert.ToString(i) + "GBB";
                    Control[] output = this.Controls.Find(nameCtrl, true);
                    if (output != null)
                    {
                        output[0].Enabled = true;
                        foreach (Control comboBox in output[0].Controls)
                        {
                            if (comboBox is ComboBox)
                            {
                                ComboBox temp = (ComboBox)comboBox;
                                temp.Items.Clear();
                                temp.Text = null;
                                temp.Items.AddRange(globalVar.productAdd);
                            }
                        }
                    }
                }
        }

        private void Step4OkBtn_Click(object sender, EventArgs e)
        {
            globalVar.slotList.Clear();
            globalVar.productInSlotList.Clear();
            globalVar.BarCodeInSlotList.Clear();

            for (int i = 1; i <= 3; i++)
            {
                string nameCtrl = "shelveLV" + Convert.ToString(i) + "GBB";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น โดยต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                Control[] output = this.Controls.Find(nameCtrl, true);
                if (output != null)
                {
                    foreach (Control comboBox in output[0].Controls)
                    {
                        if (comboBox is ComboBox)
                        {
                            ComboBox temp = (ComboBox)comboBox;
                            int indx = temp.Items.IndexOf(temp.Text);
                            if (indx != -1)
                            {

                                globalVar.slotList.Add(temp.Name);

globalVar.productInSlotList.Add(globalVar.productAdd[indx]);

globalVar.BarCodeInSlotList.Add(globalVar.BarCodeAdd[indx]);
                            }

                        }
                    }
                }
            }
            string[] tempCheck = globalVar.BarCodeInSlotList.ToArray(typeof(string))
as string[];
            if(tempCheck.Distinct().Count()!=tempCheck.Count())
            {
                globalVar.slotList.Clear();
                globalVar.productInSlotList.Clear();
                globalVar.BarCodeInSlotList.Clear();
                MessageBox.Show("Duplicate value was found in department ComboBox",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                connection.Open();
                string sqlDepart2 = "INSERT INTO
Branch_department(Store_Type,Sub_Department,Department)";
                sqlDepart2 += " VALUES (" + "'" + globalVar.typeName + "'," + "'" +
step4Sub_DepartmentCBB.Text + "'," + "'" + step4DepartmentCBB.Text + "'" + ");";
                SQLCommand = new MySqlCommand(sqlDepart2, connection);
                SQLCommand.ExecuteNonQuery();
                connection.Close();

                int memloop = globalVar.slotList.Count;
                for(int mem=1;mem<=memloop;mem++)
                {
                    string show = string.Format("{0}", globalVar.slotList[mem-1]);
                    string BarCode = string.Format("{0}",
globalVar.BarCodeInSlotList[mem-1]);
                    string showReplace = (show.Replace("shelveL",
null)).Replace("SlotCBB",";");
                    string[] levelSlotSide = showReplace.Split(';');
                    string level = levelSlotSide[0];
                    char Si = levelSlotSide[1].FirstOrDefault();
                    string Side;
                    if(Si=='L')
                    {
                        Side = "Left";
                    }
                    else
                    {
                        Side = "Right";
                    }
                    string Slot = levelSlotSide[1].TrimStart(new char[]{ 'L', 'R' });
                    connection.Open();
                    string sqlDepart3 = "INSERT INTO
Branch_product(Store_Type,BarCode,Sub_Department,Shelve,Level,Side,Slot)";
                    sqlDepart3 += " VALUES (" + "'" + globalVar.typeName + "'," + "'"
+ BarCode + "'," + "'" + step4Sub_DepartmentCBB.Text +"'," + "'" +
Convert.ToString(globalVar.shelveNumber) + "'," + "'" + level +"'," + "'" + Side +"',"
+ "'" + Slot +"'" + ");";
                    SQLCommand = new MySqlCommand(sqlDepart3, connection);
                    SQLCommand.ExecuteNonQuery();
                    connection.Close();
                    show = null;
                    showReplace =null;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    levelSlotSide = null;
                    level = null;
                    Side=null;
                }
                globalVar.shelveNumber = Convert.ToInt32(shelveNumTxt.Text)+1;
                MessageBox.Show(Convert.ToString(globalVar.shelveNumber));
            }

        }
    }
}
```

SetQuantityInBranch.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.IO;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;
using System.Speech.Synthesis;

namespace DataAccess
{
    public partial class SetQuantityInBranch : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        public SetQuantityInBranch()
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

```
            InitializeComponent();
            setFrm();
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }

            string loadStoreTemp = "select distinct store_type from Branch_template;";
            MySqlDataAdapter storeTemp = new MySqlDataAdapter(loadStoreTemp,
connection);
            DataTable storeTB = new DataTable();
            storeTemp.Fill(storeTB);
            globalVar.storeType = storeTB.AsEnumerable().Select(r =>
r.Field<string>("Store_Type")).ToArray();
            storeTempCbb.Items.AddRange(globalVar.storeType);
            barCodeImg.Image = Properties.Resources.barcode;

        }

        public void setFrm()
        {
            for(int i=1;i<=15;i++)
            {
                string dispLblStr = "prodBtn" + Convert.ToString(i);
                Control[] dispLbl = this.Controls.Find(dispLblStr,true);
                if(dispLbl!=null)
                {
                    dispLbl[0].Text = null;
                    dispLbl[0].Visible=false;
                }
            }
            upBtn.Visible = false;
            upBtn.Enabled = false;
            downBtn.Visible = false;
            downBtn.Enabled = false;
            prodSplitter.Visible = false;
            productPcb.Visible = false;
            barCodeLbl.Visible = false;
            barCodeLbl.Text = null;
            barCodeImg.Visible = false;
            brandLbl.Visible = false;
            brandDispLbl.Text = null;
            brandDispLbl.Visible = false;
            modelLbl.Visible = false;
            modelDispLbl.Text = null;
            modelDispLbl.Visible = false;
            sizeLbl.Visible = false;
            sizeDispLbl.Text = null;
            sizeDispLbl.Visible = false;
            quantityLbl.Visible = false;
            setQuantityTxt.Text = null;
            setQuantityTxt.Visible = false;
            redTrigHeadLbl.Visible = false;
            redTrigPercentLbl.Visible = false;
            redStatusTxt.Text = null;
            redStatusTxt.Visible = false;
            yellowStatusTxt.Text = null;
            yellowStatusTxt.Visible = false;
            yellowTrigHeadLbl.Visible = false;
            yellowTrigPercentLbl.Visible = false;
            saveBtn.Visible = false;
            globalVar.prouductPage = 1;
            setBtn_initialPosition();
        }

        public void setBtn_initialPosition()
        {
            for (int i = 1; i <= 15; i++)
            {
                string dispLblStr = "prodBtn" + Convert.ToString(i);
                Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                if (dispLbl != null)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                {
                    dispLbl[0].Left = 324;
                }
            }
        }

        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            string loadStoreImg = "select Store_Diagram from Branch_template where
Store_Type='" + storeTempCbb.Text + "';";
            MySqlDataAdapter storeImg = new MySqlDataAdapter(loadStoreImg,
connection);
            DataTable storeImgTB = new DataTable();
            storeImg.Fill(storeImgTB);
            byte[] img = new byte[0];
            img = (byte[])storeImgTB.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            storeDiagramPcb.Image = Bitmap.FromStream(ms);

            globalVar.department = null;
            string loadDept = "select distinct Department from Branch_department where
Store_Type='" + storeTempCbb.Text + "';";
            MySqlDataAdapter loadDepartment = new MySqlDataAdapter(loadDept,
connection);
            DataTable department = new DataTable();
            loadDepartment.Fill(department);
            globalVar.department = department.AsEnumerable().Select(r =>
r.Field<string>("Department")).ToArray();
            deptCbb.Text = null;
            deptCbb.Items.Clear();
            deptCbb.Items.AddRange(globalVar.department);
        }

        private void deptCbb_SelectedIndexChanged(object sender, EventArgs e)
        {
            if(storeTempCbb.SelectedIndex!=-1)
            {
                setFrm();
                globalVar.sub_department = null;
                string loadSubDept = "select distinct sub_Department from
Branch_department Where department='" + deptCbb.Text + "' and Store_Type='" +
storeTempCbb.Text + "';";
                MySqlDataAdapter loadSubDepartment = new MySqlDataAdapter(loadSubDept,
connection);
                DataTable subDepartment = new DataTable();
                loadSubDepartment.Fill(subDepartment);
                globalVar.sub_department = subDepartment.AsEnumerable().Select(r =>
r.Field<string>("Sub_Department")).ToArray();
                subDeptCbb.Text = null;
                subDeptCbb.Items.Clear();
                subDeptCbb.Items.AddRange(globalVar.sub_department);
            }

        }

        private void subDeptCbb_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (subDeptCbb.SelectedIndex != -1)
            {
                setFrm();
                globalVar.brand = null;
                globalVar.model = null;
                globalVar.size = null;
                globalVar.barcode2Trig = null;
                globalVar.yellowTrig = null;
                globalVar.redTrig = null;

                globalVar.productDisp = null;
                prodSplitter.Visible = true;
                refreshDatabase();
                globalVar.productLength = globalVar.barCode.Length;

                if(globalVar.productLength >= 15)
                {
                    upBtn.Visible = true;
                    upBtn.Enabled = true;
                    downBtn.Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    downBtn.Enabled = true;
                    for (int i = 1; i <= 15; i++)
                    {
                        string dispLbl = "prodBtn" + Convert.ToString(i);
                        Control[] dispLblCtrl = this.Controls.Find(dispLbl, true);
                        if (dispLblCtrl != null)
                        {
                            dispLblCtrl[0].Visible = true;
                            dispLblCtrl[0].Text = globalVar.brand[i - 1] + "\\" +
globalVar.model[i - 1] + "\\" + globalVar.size[i - 1];
                        }
                    }
                }
                else
                {
                    upBtn.Visible = true;
                    downBtn.Visible = true;
                    for (int i = 1; i <= globalVar.productLength; i++)
                    {
                        string dispLbl = "prodBtn" + Convert.ToString(i);
                        Control[] dispLblCtrl = this.Controls.Find(dispLbl, true);
                        if (dispLblCtrl != null)
                        {
                            dispLblCtrl[0].Visible = true;
                            dispLblCtrl[0].Text = globalVar.brand[i - 1] + "\\" +
globalVar.model[i - 1] + "\\" + globalVar.size[i - 1];
                        }
                    }
                }
            }
        }

        public void refreshDatabase()
        {
            globalVar.barcode2Trig=null;
            globalVar.yellowTrig=null;
            globalVar.redTrig=null;
            globalVar.barCode=null;
            globalVar.brand=null;
            globalVar.model=null;
            globalVar.size=null;
            globalVar.quantity = null;
            MySqlDataAdapter loadTrigState = new MySqlDataAdapter("select
Barcode,Yellow_State,Red_State from branch_trigger_template where Store_Type='" +
storeTempCbb.Text + "';", connection);
            DataTable loadTrigDT = new DataTable();
            loadTrigState.Fill(loadTrigDT);
            globalVar.barcode2Trig = loadTrigDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
            globalVar.yellowTrig = loadTrigDT.AsEnumerable().Select(r =>
r.Field<int>("Yellow_State")).ToArray();
            globalVar.redTrig = loadTrigDT.AsEnumerable().Select(r =>
r.Field<int>("Red_State")).ToArray();
            string loadProd = "select
Branch_product.BarCode,central_product.Brand,central_product.Model,central_product.Siz
e,Branch_product.quantity from Branch_product left join central_product on
Branch_product.BarCode = central_product.BarCode and Branch_product.store_type ='" +
storeTempCbb.Text + "' and Branch_product.sub_department ='" + subDeptCbb.Text + "'
where brand is not null;";
            MySqlDataAdapter loadProduct = new MySqlDataAdapter(loadProd, connection);
            DataTable product = new DataTable();
            loadProduct.Fill(product);
            globalVar.barCode = product.AsEnumerable().Select(r =>
r.Field<string>("BarCode")).ToArray();
            globalVar.brand = product.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
            globalVar.model = product.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
            globalVar.size = product.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
            globalVar.quantity = product.AsEnumerable().Select(r =>
r.Field<int>("Quantity")).ToArray();
        }
        private void prodBtn1_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage-1));
            setBtn_initialPosition();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            prodBtn1.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }


            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible=true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn2_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 1;
            setBtn_initialPosition();
            prodBtn2.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }
            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn3_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 2;
            setBtn_initialPosition();
            prodBtn3.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn4_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 3;
            setBtn_initialPosition();
            prodBtn4.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn5_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 4;
            setBtn_initialPosition();
            prodBtn5.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);
            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn6_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 5;
            setBtn_initialPosition();
            prodBtn6.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn7_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 6;
            setBtn_initialPosition();
            prodBtn7.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn8_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 7;
            setBtn_initialPosition();
            prodBtn8.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น โดยไม่ได้รับอนุญาตจากผู้เขียนเอกสาร และขอสงวนสิทธิ์ในการฟ้องร้องดำเนินคดีตามกฎหมายกับผู้ที่นำเอกสารนี้ไปใช้ในการค้า

```
            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn9_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 8;
            setBtn_initialPosition();
            prodBtn9.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn10_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 9;
            setBtn_initialPosition();
            prodBtn10.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn11_Click(object sender, EventArgs e)
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น การอ้างอิงหรือสำเนาต่อเติมเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            int indx = (15 * (globalVar.prouductPage - 1)) + 10;
            setBtn_initialPosition();
            prodBtn11.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn12_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 11;
            setBtn_initialPosition();
            prodBtn12.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }
            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn13_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 12;
            setBtn_initialPosition();
            prodBtn13.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn14_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 13;
            setBtn_initialPosition();
            prodBtn14.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn15_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 14;
            setBtn_initialPosition();
            prodBtn15.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if(indxOfTrig!=-1)
            {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void downBtn_Click(object sender, EventArgs e)
        {
            for (int i = 1; i <= 15; i++)
            {
                string dispLblStr = "prodBtn" + Convert.ToString(i);
                Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                if (dispLbl != null)
                {
                    dispLbl[0].Left = 324;
                    dispLbl[0].Text = null;
                    dispLbl[0].Visible = false;
                }
            }

            if(globalVar.prouductPage>=1)
            {
                if((globalVar.productLength - ((15*(globalVar.prouductPage))))<=15)
                {
                    for (int i = 1; i <= (globalVar.productLength - ((15 *
(globalVar.prouductPage)))); i++)
                    {
                        string dispLblStr = "prodBtn" + Convert.ToString(i);
                        Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                        if (dispLbl != null)
                        {   dispLbl[0].Visible = true;
                        dispLbl[0].Text = globalVar.brand[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.model[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.size[(15 *
globalVar.prouductPage) + (i - 1)];
                        }
                        upBtn.Visible = true;
                        downBtn.Visible = false;
                    }
                    globalVar.prouductPage = globalVar.prouductPage + 1;
                }
                else
                {

                    for (int i = 1; i <= 15; i++)
```

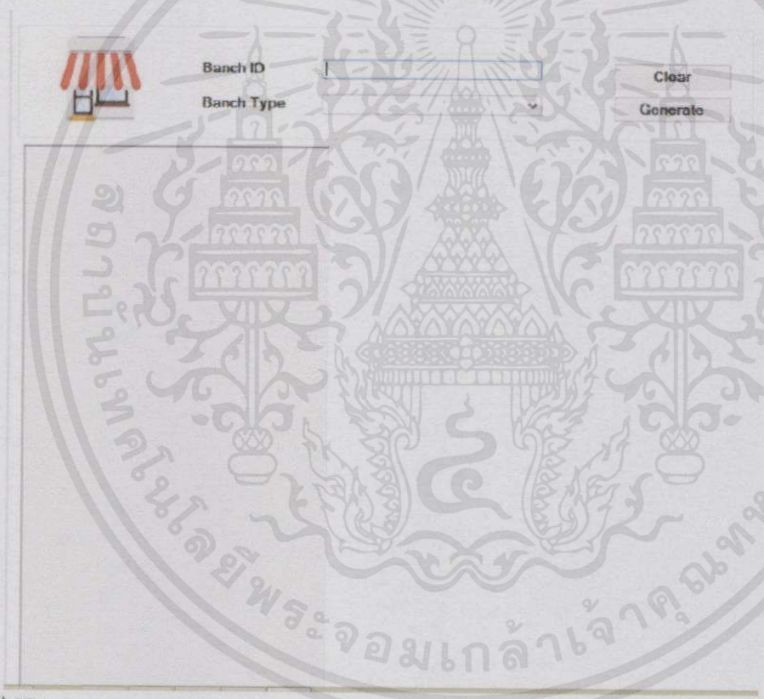
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    {
                        string dispLblStr = "prodBtn" + Convert.ToString(i);
                        Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                        if (dispLbl != null)
                        {
                            dispLbl[0].Visible = true;
                            dispLbl[0].Text = globalVar.brand[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.model[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.size[(15 *
globalVar.prouductPage) + (i - 1)];
                        }
                    }
                    globalVar.prouductPage = globalVar.prouductPage + 1;
                }
            }
        }

        private void upBtn_Click(object sender, EventArgs e)
        {
            if (globalVar.prouductPage > 1)
            {
                for (int i = 1; i <= 15; i++)
                {
                    string dispLblStr = "prodBtn" + Convert.ToString(i);
                    Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                    if (dispLbl != null)
                    {
                        dispLbl[0].Left = 324;
                        dispLbl[0].Text = null;
                        dispLbl[0].Visible = false;
                    }
                }
                downBtn.Visible = true;
                globalVar.prouductPage = globalVar.prouductPage - 1;
                for (int i = 1; i <= 15; i++)
                {
                    string dispLblStr = "prodBtn" + Convert.ToString(i);
                    Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                    if (dispLbl != null)
                    {
                        dispLbl[0].Visible = true;
                        dispLbl[0].Text = globalVar.brand[(15 *
(globalVar.prouductPage-1)) + (i - 1)] + "\\" + globalVar.model[(15 *
(globalVar.prouductPage-1)) + (i - 1)] + "\\" + globalVar.size[(15 *
(globalVar.prouductPage-1)) + (i - 1)];
                    }
                }

            }
            else
            {
                upBtn.Visible = false;
            }
        }

        private void saveBtn_Click(object sender, EventArgs e)
        {
            if(setQuantityTxt.Text!="")
            {
                connection.Open();
                string updateQuantity = "update Branch_product set Quantity=" +
setQuantityTxt.Text + " where barcode=" + barCodeLbl.Text + " and Store_Type='" +
storeTempCbb.Text + "';";
                SQLCommand = new MySqlCommand(updateQuantity, connection);
                SQLCommand.ExecuteNonQuery();


                SQLCommand = new MySqlCommand();
                SQLCommand.Connection = connection;
                SQLCommand.CommandText = "setBranchTrig";
                SQLCommand.CommandType = CommandType.StoredProcedure;
                SQLCommand.Parameters.Add("?Store_Type_Input", storeTempCbb.Text);
                SQLCommand.Parameters["?Store_Type_Input"].Direction =
ParameterDirection.Input;
                SQLCommand.Parameters.Add("?Barcode_Input", barCodeLbl.Text);
                SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

```
                SQLCommand.Parameters.Add("?YellowState", yellowStatusTxt.Text);
                SQLCommand.Parameters["?YellowState"].Direction =
ParameterDirection.Input;
                SQLCommand.Parameters.Add("?RedState", redStatusTxt.Text);
                SQLCommand.Parameters["?RedState"].Direction =
ParameterDirection.Input;
                SQLCommand.ExecuteNonQuery();
                connection.Close();
                MessageBox.Show("BarCode: " + barCodeLbl.Text + "\n" + "Brand: " +
brandDispLbl.Text + "\n" + "Model: " + modelDispLbl.Text + "\n" + "Size: " +
sizeDispLbl.Text + "\n has been updated", "Quantity change system",
MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
            else
            {
                MessageBox.Show("Error: you have to fill in quantity textbox",
"Quantity change system", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

    }
}
```

NewBranch.cs




```
using System;
using System.IO;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;
using System.Speech.Synthesis;

namespace DataAccess
{
    public partial class NewBranch : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        SpeechSynthesizer speechEngine = new SpeechSynthesizer();
        MySqlConnection connection;
        MySqlCommand SQLCommand;

        public NewBranch()
        {
            InitializeComponent();
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            MySqlDataAdapter contrainBranchID = new MySqlDataAdapter("select BranchID
from Branch;",connection);
            DataTable branchIDDT = new DataTable();
            contrainBranchID.Fill(branchIDDT);
            globalVar.branchIDInput = branchIDDT.AsEnumerable().Select(r =>
r.Field<int>("BranchID")).ToArray();
            MySqlDataAdapter contrainStoreType = new MySqlDataAdapter("select
Store_type from Branch_Template",connection);
            DataTable storeTypeDT = new DataTable();
            contrainStoreType.Fill(storeTypeDT);
            globalVar.storeTypeInput = storeTypeDT.AsEnumerable().Select(r =>
r.Field<string>("Store_Type")).ToArray();
            branchTypeCBB.Items.AddRange(globalVar.storeTypeInput);

        }

        private void centralStockAccessFrm_Load(object sender, EventArgs e)
        {

        }

        private void generateBtn_Click(object sender, EventArgs e)
        {
            if (branchIdTxt.Text == "")
            {
                MessageBox.Show("Please fill in \"Branch ID\"", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            if (branchTypeCBB.Text == "")
            {
                MessageBox.Show("Please select \"Branch Type\"", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                if (branchIdTxt.Text != "")
                {
                    if
(globalVar.branchIDInput.Contains(Convert.ToInt32(branchIdTxt.Text)) != true)
                    {
                        if (branchIdTxt.Text != " ")
                        {
                            if(branchIdTxt.Text.Length==4)
                            {

monitorRTB.AppendText("=============================================\n");
                                monitorRTB.SelectionColor = Color.BlueViolet;
                                monitorRTB.AppendText("                                New
Branch                    \n");
                                monitorRTB.SelectionColor = Color.Black;

monitorRTB.AppendText("=============================================\n");
                                connection.Open();
                                monitorRTB.SelectionColor = Color.Blue;

                                //Insert into Branch Table
                                monitorRTB.AppendText("Task: Insert Into Branch
Table\n");
                                string insert2Branch = "insert into
branch(BranchID,Branch_Type)";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                insert2Branch += " values(" + branchIdTxt.Text + ",'"
+ branchTypeCBB.Text + "');";
                                SQLCommand = new MySqlCommand(insert2Branch,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Copy Branch_Trigger
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize record to
\"Branch_Tringger\" table\n");
                                string synBranchTriggerTB = "insert into
Branch_Trigger(BranchID,Barcode,YellowTrig,RedTrig) select
Branch.BranchID,Branch_product.Barcode,Branch_Trigger_Template.Yellow_State,Branch_Tri
gger_Template.Red_State from Branch_product left join branch on
Branch_product.store_Type=Branch.Branch_Type and Branch_product.store_type='" +
branchTypeCBB.Text + "' and branch.BranchID=" + branchIdTxt.Text + " left join
Branch_trigger_template on Branch_trigger_template.Store_Type='" + branchTypeCBB.Text
+ "' and Branch_Trigger_Template.Barcode=Branch_Product.Barcode where BranchID is not
null;";
                                SQLCommand = new MySqlCommand(synBranchTriggerTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create database ==> kmitl_shop_bXXXX
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create \"kmitl_shop_B" +
branchIdTxt.Text + "\" database\n");
                                string sqlDepart = "create database kmitl_shop_B" +
branchIdTxt.Text + ";";
                                SQLCommand = new MySqlCommand(sqlDepart, connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Central_Product table and copy record
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create
\"Central_Product\" table\n");
                                string createProductDetailTB = "create table
Kmitl_Shop_B" + branchIdTxt.Text + ".Central_Product(BarCode varchar(13) not null
primary key,Sub_Department varchar(50) not null,Brand varchar(50) not null,Model
varchar(50) not null,Size varchar(30) not null,ContainUnit varchar(50) not
null,PricePerContain float not null,ProductImage longblob not null,UpdateTime
timestamp(12) not null) type=InnoDB;";
                                SQLCommand = new MySqlCommand(createProductDetailTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize record to
\"Central_Product\" table\n");
                                string synProductDetailTB = "insert into kmitl_shop_b"
+ branchIdTxt.Text +
".Central_Product(BarCode,Sub_Department,Brand,Model,Size,ContainUnit,PricePerContain,
ProductImage) select
BarCode,Sub_Department,Brand,Model,Size,ContainUnit,PricePerContain,ProductImage from
kmitl_shop.central_product;";
                                SQLCommand = new MySqlCommand(synProductDetailTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Product table and copy record
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create \"Product\"
table\n");
                                string createProductTB = "create table Kmitl_Shop_B" +
branchIdTxt.Text + ".Product(BarCode varchar(13) not null primary key,Sub_Department
varchar(50) not null,Shelve varchar(3) not null,Level varchar(1) not null,Side
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
varchar(5) not null,Slot varchar(2) not null,UpdateTime timestamp(12) not null)
type=InnoDB;";
                                SQLCommand = new MySqlCommand(createProductTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize record to
\"Product\" table\n");
                                string synProductTB = "insert into Kmitl_Shop_B" +
branchIdTxt.Text + ".Product(BarCode,Sub_Department,Shelve,Level,Side,Slot) select
BarCode,Sub_Department,Shelve,Level,Side,Slot from kmitl_shop.Branch_product where
store_type ='" + branchTypeCBB.Text + "';";
                                SQLCommand = new MySqlCommand(synProductTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Department table and copy record
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create \"Department\"
table\n");
                                string createDepartmentTB = "create table
Kmitl_Shop_B" + branchIdTxt.Text + ".Department(Sub_Department varchar(50) not null
primary key,Department varchar(50) not null,UpdateTime timestamp(12) not null)
type=InnoDB;";
                                SQLCommand = new MySqlCommand(createDepartmentTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize record to
\"Department\" table\n");
                                string synDepartmentTB = "insert into Kmitl_Shop_B" +
branchIdTxt.Text + ".Department(Sub_Department,Department) select
Sub_Department,Department from kmitl_shop.Branch_department where store_type='" +
branchTypeCBB.Text + "';";
                                SQLCommand = new MySqlCommand(synDepartmentTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Central_Department table and copy record
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create
\"Central_Department\" table\n");
                                string createCentralDepartmentTB = "create table
Kmitl_Shop_B" + branchIdTxt.Text + ".Central_Department(Sub_Department varchar(50) not
null primary key,Department varchar(50) not null,UpdateTime timestamp(12) not null)
type=InnoDB;";
                                SQLCommand = new
MySqlCommand(createCentralDepartmentTB, connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize record to
\"Central_Department\" table\n");
                                string synCenDepartmentTB = "insert into Kmitl_Shop_B"
+ branchIdTxt.Text + ".Central_Department(Sub_Department,Department) select
Sub_Department,Department from kmitl_shop.central_department;";
                                SQLCommand = new MySqlCommand(synCenDepartmentTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Trigger table and copy record
                                monitorRTB.SelectionColor = Color.Blue;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

```
                                monitorRTB.AppendText("Task: Create \"Branch_Trigger\"
table\n");
                                string createBranchTriggerTB = "create table
Kmitl_Shop_B" + branchIdTxt.Text + ".Branch_Trigger(BranchID int not null default " +
branchIdTxt.Text + ",Barcode varchar(13) not null,YellowTrig int not null,RedTrig int
not null,UpdateTime timestamp(12) not null) Type=InnoDB;";
                                SQLCommand = new MySqlCommand(createBranchTriggerTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize record to
\"Branch_Trigger\" table\n");
                                string synBranchTriggerTB2 = "insert into
Kmitl_Shop_B" + branchIdTxt.Text + ".Branch_Trigger(Barcode,YellowTrig,RedTrig) select
Barcode,Yellow_State,Red_State from kmitl_shop.Branch_Trigger_Template where
Store_Type='" + branchTypeCBB.Text + "';";
                                SQLCommand = new MySqlCommand(synBranchTriggerTB2,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Stock table and copy record
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create \"Stock\"
table\n");
                                string createStockTB = "create table Kmitl_Shop_B" +
branchIdTxt.Text + ".Stock(BarCode varchar(13) not null primary key,Set_Quantity int
not null,Quantity int not null,Product_Status enum('G','Y','R') not null,UpdateTime
timestamp(12) not null) type=InnoDB;";
                                SQLCommand = new MySqlCommand(createStockTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize record to
\"Stock table\" except quantity\n");
                                string synStockTB = "insert into Kmitl_Shop_B" +
branchIdTxt.Text + ".Stock(BarCode,Set_Quantity) select BarCode,Quantity from
kmitl_shop.Branch_product where store_type='" + branchTypeCBB.Text + "';";
                                SQLCommand = new MySqlCommand(synStockTB, connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                    //Create Trigger[ProductStatus]
                                    monitorRTB.SelectionColor = Color.Blue;
                                    monitorRTB.AppendText("Task: Create
\"Trigger[ProductStatus]:Stock table\" \n");
                                    SQLCommand = new MySqlCommand("use kmitl_shop_b" +
branchIdTxt.Text, connection);
                                    SQLCommand.ExecuteNonQuery();
                                    SQLCommand.CommandText = "Create trigger
prodStatusTrigger before update on Stock" +
                                                             " For Each Row" +
                                                             " Begin" +
                                                                " declare YState float
default 0;" +
                                                                "declare RState float
default 0;" +
                                                                "declare YValue int
default 0;" +
                                                                "declare RValue int
default 0;" +
                                                                "select YellowTrig
from Branch_Trigger Where Barcode=new.Barcode into YState;" +
                                                                "set
YValue=((new.Set_Quantity*YState)/100);" +
                                                                "select RedTrig from
Branch_Trigger where Barcode=new.Barcode into RState;" +
                                                                "set
RValue=((new.Set_Quantity*RState)/100);" +
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
"if(new.Quantity<=RValue) then set new.Product_Status='R';" +

"elseif((new.Quantity>RValue) and (new.Quantity<=YValue)) then set
new.Product_Status='Y';" +
                                                        "else set
new.Product_Status='G';" +
                                                        "end if;" +
                                                    "end;";

                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Synconize quantity
column to \"Stock table\" \n");
                                SQLCommand = new MySqlCommand("use kmitl_shop_b" +
branchIdTxt.Text, connection);
                                SQLCommand.ExecuteNonQuery();
                                string synProdQuantity = "update Stock,(select
Barcode,Quantity from kmitl_shop.Branch_product where store_type='" +
branchTypeCBB.Text + "') temp set stock.quantity=temp.quantity where
stock.barcode=temp.barcode;";
                                SQLCommand = new MySqlCommand(synProdQuantity,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Procedure[stockQuantityChange]
                                monitorRTB.SelectionColor = Color.Purple;
                                monitorRTB.AppendText("Task: Create
\"Procedure[stockQuantityChange]:Stock Table\" \n");
                                SQLCommand = new MySqlCommand("use kmitl_shop_b" +
branchIdTxt.Text, connection);
                                SQLCommand.ExecuteNonQuery();
                                SQLCommand.CommandText = "create procedure
orderQuantityChange(in Barcode_Input varchar(13),in isPlus bool,out isDone bool)" +
                                                        " begin if (isPlus=true)
then" +
                                                        " if (select Quantity from
stock where Barcode=Barcode_Input)>=1 then" +
                                                        " update stock set
Quantity=Quantity-1 where Barcode=Barcode_Input;" +
                                                        " set isDone=true;" +
                                                    " else" +
                                                        " set isDone=false;" +
                                                    "end if;" +
                                                    "elseif (isPlus=false)
then" +
                                                        " update stock set
Quantity=Quantity+1 where Barcode=Barcode_Input;" +
                                                        " set isDone=true;" +
                                                    "end if;end;";
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");
                                //Create Procedure[returnQuantity]
                                monitorRTB.SelectionColor = Color.Purple;
                                monitorRTB.AppendText("Task: Create
\"Procedure[returnQuantity]:Stock Table\" table\n");
                                SQLCommand = new MySqlCommand("use kmitl_shop_b" +
branchIdTxt.Text, connection);
                                SQLCommand.ExecuteNonQuery();
                                SQLCommand.CommandText = "create procedure
returnQuantity(Barcode_Input varchar(13),Quantity_Input int)" +
                                                        " begin" +
                                                        " update stock set
Quantity=Quantity+Quantity_Input where Barcode=Barcode_Input;" +
                                                    "end;";
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                            //Create Order_Opending table
                            monitorRTB.SelectionColor = Color.Blue;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

```
                                monitorRTB.AppendText("Task: Create \"Order_Pending\"
table\n");
                                string createOrderTB = "create table Kmitl_Shop_B" +
branchIdTxt.Text + ".Order_Pending(RobotID varchar(30) not null,OrderID int not
null,BarCode varchar(13) not null,Quantity int not null,UpdateTime timestamp(12) not
null) Type=InnoDB;";
                                SQLCommand = new MySqlCommand(createOrderTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Dialy_Order table
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create \"Dialy_Order\"
table\n");
                                string createDialyOrderTB = "create table
Kmitl_Shop_B" + branchIdTxt.Text + ".Dialy_Order(BranchID int not null default " +
branchIdTxt.Text + ",BarCode varchar(13) not null,Quantity int not null,UpdateTime
timestamp(12) not null) Type=InnoDB;";
                                SQLCommand = new MySqlCommand(createDialyOrderTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create AccountAccess table and set root account
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create \"AccountAccess\"
table\n");
                                string createAccountAccessTB = "create table
Kmitl_Shop_B" + branchIdTxt.Text + ".AccountAccess(User varchar(20) not null,Pass
varchar(20) not null,Position varchar(50) not null,UpdateTime timestamp(12) not null)
Type=InnoDB;";
                                SQLCommand = new MySqlCommand(createAccountAccessTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Insert root account into
\"AccountAccess\" table\n");
                                string insertRootAccount = "insert into
AccountAccess(User,Pass,Position) values('root','555','IT');";
                                SQLCommand = new MySqlCommand(insertRootAccount,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                //Create Sell table
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Create \"Sell\"
table\n");
                                string createDSellTB = "create table Kmitl_Shop_B" +
branchIdTxt.Text + ".Sell(BranchID int default " + branchIdTxt.Text + ", BarCode
varchar(13) not null,Brand varchar(50) not null,Model varchar(50) not null,Size
varchar(30) not null,PricePerContain float not null,Quantity int not null,Price float
not null,UpdateTime timestamp(12) not null) Type=InnoDB;";
                                SQLCommand = new MySqlCommand(createDSellTB,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");
                                    //Create Procedure[add2SellTB]
                                    monitorRTB.SelectionColor = Color.Purple;
                                    monitorRTB.AppendText("Task: Create
\"Procedure[Add2SellTB]:Sell Table\" table\n");
                                    SQLCommand = new MySqlCommand("use kmitl_shop_b" +
branchIdTxt.Text, connection);
                                    SQLCommand.ExecuteNonQuery();
                                    SQLCommand.CommandText = "create procedure
add2SellTB(Barcode_Input varchar(13),Quantity_Input int)" +
                                                             " begin declare memCount
int default 0;" +
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

```
                                                    "declare pricePC float
default 0;" +
                                                    "declare sumPrice float
default 0;" +
                                                    "declare updateQuantity
int default 0;" +
                                                    "select PricePerContain
from central_product where barcode=Barcode_Input into pricePC;" +
                                                    "select count(Barcode)
from sell where Barcode=Barcode_Input and" +
                                                    "
month(now())=month((select UpdateTime from sell where Barcode=Barcode_Input)) and" +
                                                    " year(now())=year((select
UpdateTime from sell where Barcode=Barcode_Input)) into memCount;" +
                                                    "if memCount=0 then" +
                                                    " set
sumPrice=pricePC*Quantity_Input;" +
                                                    "insert into
sell(Barcode,PricePerContain,Quantity,Price)
values(Barcode_Input,PricePC,Quantity_Input,sumPrice);" +
                                                    "update sell set
Brand=(select Brand from central_product where Barcode=Barcode_Input) where
Barcode=Barcode_Input;" +
                                                    "update sell set
Model=(select Model from central_product where Barcode=Barcode_Input) where
Barcode=Barcode_Input;" +
                                                    "update sell set
Size=(select Size from central_product where Barcode=Barcode_Input) where
Barcode=Barcode_Input;" +
                                                    "else" +
                                                    " update sell set
Quantity=Quantity+Quantity_Input where barcode=Barcode_Input and" +
                                                    "
month(now())=month((select updatetime from (select updatetime from sell as temp1 where
Barcode=Barcode_Input)as temp2)) and" +
                                                    " year(now())=year((select
updatetime from (select updatetime from sell as temp3 where Barcode=Barcode_Input)as
temp4));" +
                                                    "set
updateQuantity=(select Quantity from sell where barcode=Barcode_Input and" +
                                                    "
month(now())=month((select updatetime from (select updatetime from sell as temp1 where
Barcode=Barcode_Input)as temp2)) and" +
                                                    " year(now())=year((select
updatetime from (select updatetime from sell as temp3 where Barcode=Barcode_Input)as
temp4)));" +
                                                    "set
sumPrice=PricePC*updateQuantity;" +
                                                    "update sell set
Price=sumPrice where barcode=Barcode_Input and" +
                                                    "
month(now())=month((select updatetime from (select updatetime from sell as temp1 where
Barcode=Barcode_Input)as temp2)) and" +
                                                    " year(now())=year((select
updatetime from (select updatetime from sell as temp3 where Barcode=Barcode_Input)as
temp4));" +
                                                    "end if; end;";
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");


                                //Define Foreign Key
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"centralProduct2centralDepartment\"\n");
                                string createFKcenProd2cenDept = "alter table
Kmitl_Shop_B" + branchIdTxt.Text + ".central_product add constraint
centralProduct2centralDepartment foreign key(Sub_Department) references Kmitl_Shop_B"
+ branchIdTxt.Text + ".Central_Department(Sub_Department) on delete cascade on update
cascade;";
                                SQLCommand = new MySqlCommand(createFKcenProd2cenDept,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");
```

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

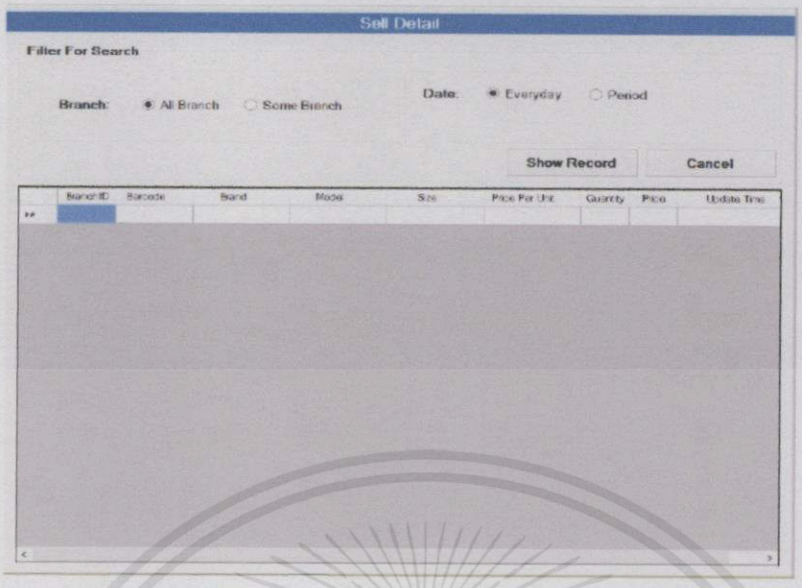
```
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"product2centralProduct\"\n");
                                string createFKprod2cenProd = "alter table
Kmitl_Shop_B" + branchIdTxt.Text + ".product add constraint product2centralProduct
foreign key(BarCode) references Kmitl_Shop_B" + branchIdTxt.Text +
".Central_Product(BarCode) on delete cascade on update cascade;";
                                SQLCommand = new MySqlCommand(createFKprod2cenProd,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"product2department\"\n");
                                string createFKprod2dept = "alter table Kmitl_Shop_B"
+ branchIdTxt.Text + ".product add constraint product2department foreign
key(Sub_Department) references Kmitl_Shop_B" + branchIdTxt.Text +
".Department(Sub_Department) on delete cascade on update cascade;";
                                SQLCommand = new MySqlCommand(createFKprod2dept,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"stock2product\"\n");
                                string createFKstock2prod = "alter table Kmitl_Shop_B"
+ branchIdTxt.Text + ".stock add constraint stock2product foreign key(BarCode)
references Kmitl_Shop_B" + branchIdTxt.Text + ".Product(BarCode) on delete cascade on
update cascade;";
                                SQLCommand = new MySqlCommand(createFKstock2prod,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"orderPending2stock\"\n");
                                string createFKorderPe2stock = "alter table
Kmitl_Shop_B" + branchIdTxt.Text + ".order_pending add constraint orderPending2stock
foreign key(BarCode) references Kmitl_Shop_B" + branchIdTxt.Text + ".stock(BarCode) on
delete cascade on update cascade;";
                                SQLCommand = new MySqlCommand(createFKorderPe2stock,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"sell2stock\"\n");
                                string createFKsell2stock = "alter table Kmitl_Shop_B"
+ branchIdTxt.Text + ".sell add constraint sell2stock foreign key(BarCode) references
Kmitl_Shop_B" + branchIdTxt.Text + ".stock(BarCode) on delete cascade on update
cascade;";
                                SQLCommand = new MySqlCommand(createFKsell2stock,
connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");

                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"branchTrigger2centralProduct\"\n");
                                string createFKbranchTrig2cenDept = "alter table
Kmitl_Shop_B" + branchIdTxt.Text + ".Branch_Trigger add constraint
branchTrigger2centralProduct foreign key(Barcode) references Kmitl_Shop_B" +
branchIdTxt.Text + ".central_product(Barcode) on delete cascade on update cascade;";
                                SQLCommand = new
MySqlCommand(createFKbranchTrig2cenDept, connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

```
                                monitorRTB.SelectionColor = Color.Blue;
                                monitorRTB.AppendText("Task: Define foreign key
constraint \"centralProduct2centralDepartment\"\n");
                                string createFKdialyOrder2product = "alter table
Kmitl_Shop_B" + branchIdTxt.Text + ".Dialy_Order add constraint dialyOrder2product
foreign key(Barcode) references Kmitl_Shop_B" + branchIdTxt.Text + ".Product(Barcode)
on delete cascade on update cascade;";
                                SQLCommand = new
MySqlCommand(createFKdialyOrder2product, connection);
                                SQLCommand.ExecuteNonQuery();
                                monitorRTB.SelectionColor = Color.Green;
                                monitorRTB.AppendText("Status: Done\n");


monitorRTB.AppendText("=============================================\n\n");
                                speechEngine.SpeakAsync("Process is done");
                                MessageBox.Show("Process is done", "System Status",
MessageBoxButtons.OK, MessageBoxIcon.Information);



                            }
                            else
                            {
                                MessageBox.Show("Please fill in \"Branch ID\" with 4
digit", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                            }
                        }

                    }
                    else
                    {
                        MessageBox.Show("Branch ID does exist in database.", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                }
            }
        }

        private void branchTypeCBB_SelectedIndexChanged(object sender, EventArgs e)
        {
            string sqlLoadStoreDagram = "select store_diagram from Branch_template
where store_type='" + branchTypeCBB.Text + "';";
            MySqlDataAdapter storeDiagramDA = new MySqlDataAdapter(sqlLoadStoreDagram,
connection);
            DataTable storeDiagramDT = new DataTable();
            storeDiagramDA.Fill(storeDiagramDT);
            byte[] img = new byte[0];
            img = (byte[])storeDiagramDT.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            branchDiagramPcb.Image = Bitmap.FromStream(ms);
        }

    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SellDetail.cs




```
        using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using MySql.Data.MySqlClient;

namespace DataAccess
{
    public partial class SellDetail : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        DataTable loadSellTB = new DataTable();
        public int[] allBranchID;
        public SellDetail()
        {
            InitializeComponent();
            fromDateLbl.Visible = false;
            fromDateMtb.Visible = false;
            toDateLbl.Visible = false;
            toDateMtb.Visible = false;
            fromDateMtb.Mask = "0000/00/00";
            toDateMtb.Mask = "0000/00/00";
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            MySqlDataAdapter loadBranch = new MySqlDataAdapter("select distinct
BranchID from branch;", connection);
            DataTable BranchIDTB = new DataTable();
            loadBranch.Fill(BranchIDTB);
            allBranchID =
BranchIDTB.AsEnumerable().Select(r=>r.Field<int>("BranchID")).ToArray();

            showRecordDGV.AutoGenerateColumns = false;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            showRecordDGV.Columns["BranchID"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Barcode"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Brand"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Model"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Size"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["PricePerUnit"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Quantity"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Price"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["UpdateTime"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
        }

        #region Date to search

        private void fromDateMtb_InputReject(object sender, MaskInputRejectedEventArgs
e)
        {
            if (fromDateMtb.MaskFull)
            {
                fromDateTt.ToolTipTitle = "Input Rejected - Too Much Data";
                fromDateTt.Show("You cannot enter any more data into the date field.
Delete some characters in order to insert more data.", fromDateMtb, 0, -20, 5000);
            }
            else if (e.Position == fromDateMtb.Mask.Length)
            {
                fromDateTt.ToolTipTitle = "Input Rejected - End of Field";
                fromDateTt.Show("You cannot add extra characters to the end of this
date field.", fromDateMtb, 0, -20, 5000);
            }
            else
            {
                fromDateTt.ToolTipTitle = "Input Rejected";
                fromDateTt.Show("You can only add numeric characters (0-9) into this
date field.", fromDateMtb, 0, -20, 5000);
            }
        }
        private void toDateMtb_MaskInputRejected(object sender,
MaskInputRejectedEventArgs e)
        {
            if (toDateMtb.MaskFull)
            {
                toDateTt.ToolTipTitle = "Input Rejected - Too Much Data";
                toDateTt.Show("You cannot enter any more data into the date field.
Delete some characters in order to insert more data.", toDateMtb, 0, -20, 5000);
            }
            else if (e.Position == toDateMtb.Mask.Length)
            {
                toDateTt.ToolTipTitle = "Input Rejected - End of Field";
                toDateTt.Show("You cannot add extra characters to the end of this date
field.", toDateMtb, 0, -20, 5000);
            }
            else
            {
                toDateTt.ToolTipTitle = "Input Rejected";
                toDateTt.Show("You can only add numeric characters (0-9) into this
date field.", toDateMtb, 0, -20, 5000);
            }
        }

        private void toDateMtb_KeyDown(object sender, KeyEventArgs e)
        {
            toDateTt.Hide(toDateMtb);
        }
        private void fromDateMtb_KeyDown(object sender, KeyEventArgs e)
        {
            fromDateTt.Hide(fromDateMtb);
        }

        #endregion
```

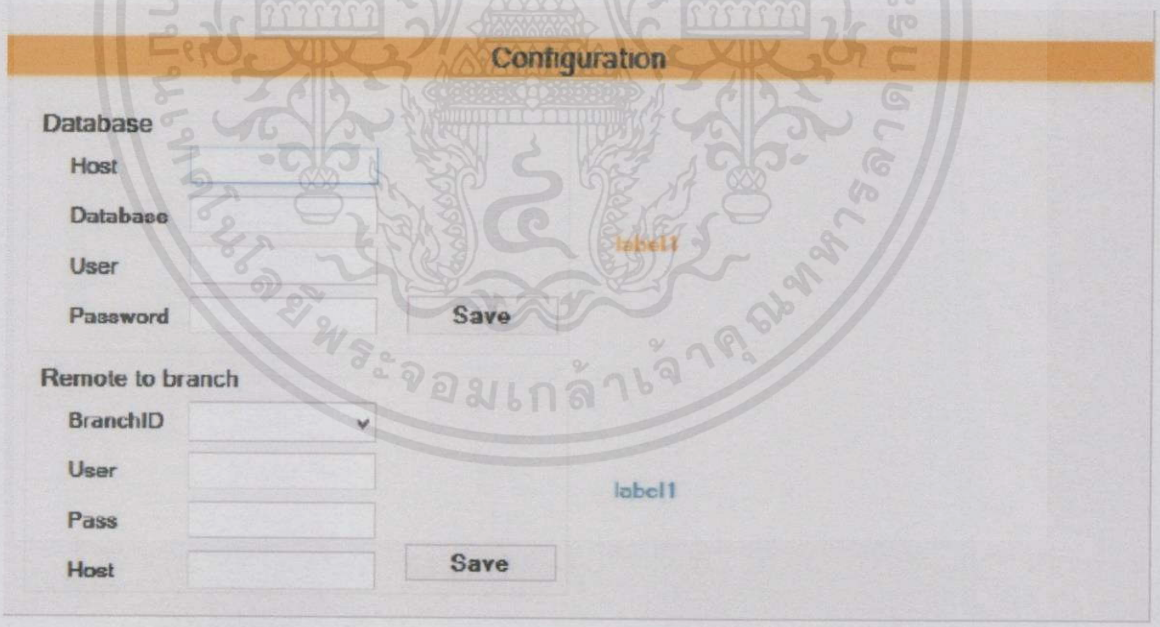
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        private void showRecordBtn_Click(object sender, EventArgs e)
        {
            loadSellTB.Clear();
            string MySQLStrCommandHead = "select BranchID as BranchID,Barcode as
Barcode,Brand as Brand,Model as Model,Size as Size,PricePerContain as 'Price Per
Contain',Quantity as Quantity,Price as Price,UpdateTime as 'Update Time' from
branch_sell ";

            string MySQLStrCommandTail = "";

            #region create MySQL String Command

            if (allBranchRbtn.Checked == true)
            {
                if(everyDateRtb.Checked==true)
                {
                    MySQLStrCommandTail = "";
                }
                else if(periodDayRtn.Checked==true)
                {
                    MySQLStrCommandTail = "where date(updateTime) between date('" +
fromDateMtb.Text + "') and date('" + toDateMtb.Text + "') ";
                }
            }
            else if(someBranchRbt.Checked==true)
            {
                selectBranchFrm selectBranchForm = new selectBranchFrm(allBranchID);
                DialogResult branchSelectResult = selectBranchForm.ShowDialog();
                if(branchSelectResult==DialogResult.OK)
                {
                    int countBranch = selectBranchForm.branchResult.Count();
                    if (countBranch > 0)
                    {
                        MySQLStrCommandTail = "where BranchID='" +
selectBranchForm.branchResult[0] + "' ";
                        if (countBranch > 1)
                        {
                            for (int addStr = 1; addStr <= countBranch-1; addStr++)
                            {
                                MySQLStrCommandTail = MySQLStrCommandTail + "or
BranchID='" + selectBranchForm.branchResult[addStr] + "' ";
                            }
                        }
                    }
                    else
                    {
                        MySQLStrCommandTail = "";
                        return;
                    }

                    if (periodDayRtn.Checked == true)
                    {
                        MySQLStrCommandTail = MySQLStrCommandTail + " and
date(updateTime) between date('" + fromDateMtb.Text + "') and date('" + toDateMtb.Text
+ "') ";
                    }
                }
                else
                {
                    return;
                }
            }

            #endregion

            string MySQLStrCommand = MySQLStrCommandHead + MySQLStrCommandTail + ";";
            MySqlDataAdapter loadSell = new MySqlDataAdapter(MySQLStrCommand,
connection);
            loadSell.Fill(loadSellTB);
            showRecordDGV.Columns["BranchID"].DataPropertyName = "BranchID";
            showRecordDGV.Columns["Barcode"].DataPropertyName = "Barcode";
            showRecordDGV.Columns["Brand"].DataPropertyName = "Brand";
            showRecordDGV.Columns["Model"].DataPropertyName = "Model";
            showRecordDGV.Columns["Size"].DataPropertyName = "Size";
            showRecordDGV.Columns["PricePerUnit"].DataPropertyName = "Price Per
Contain";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

```
            showRecordDGV.Columns["Quantity"].DataPropertyName = "Quantity";
            showRecordDGV.Columns["Price"].DataPropertyName = "Price";
            showRecordDGV.Columns["UpdateTime"].DataPropertyName = "Update Time";
            showRecordDGV.DataSource = loadSellTB;
            showRecordDGV.Show();
        }

        private void periodDayRtn_CheckedChanged(object sender, EventArgs e)
        {
            if(periodDayRtn.Checked==true)
            {
                fromDateLbl.Visible = true;
                fromDateMtb.Visible = true;
                toDateLbl.Visible = true;
                toDateMtb.Visible = true;
            }
            else
            {
                fromDateLbl.Visible = false;
                fromDateMtb.Visible = false;
                toDateLbl.Visible = false;
                toDateMtb.Visible = false;
            }
        }

        private void cancelBtn_Click(object sender, EventArgs e)
        {
            loadSellTB.Clear();
        }
    }
}
```

Configuration.Frm



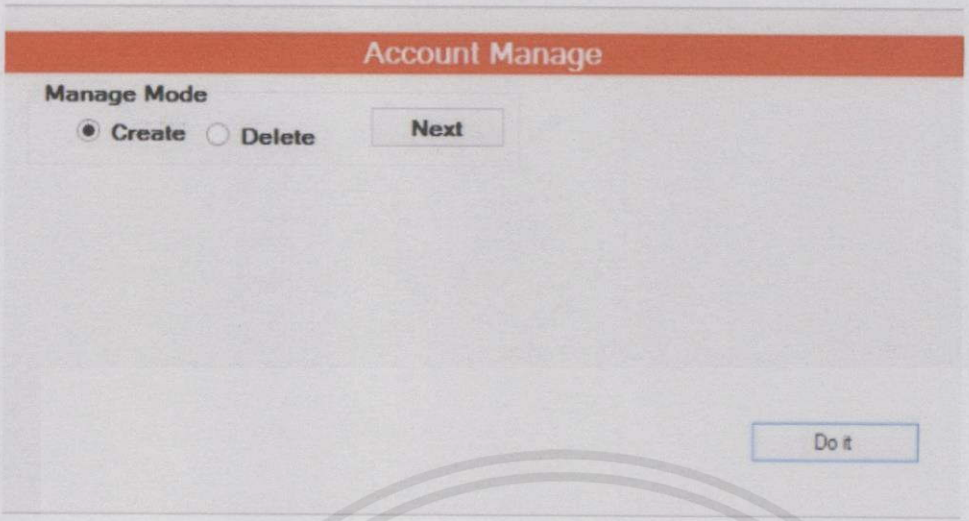

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using MySql.Data.MySqlClient;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
namespace DataAccess
{
    public partial class Configuration : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        DataAccessSetting setting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        int[] branchID;
        public Configuration()
        {
            InitializeComponent();
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            MySqlDataAdapter loadBranchID = new MySqlDataAdapter("select BranchID from
branch;", connection);
            DataTable branchIDDT = new DataTable();
            loadBranchID.Fill(branchIDDT);
            branchID = branchIDDT.AsEnumerable().Select(r =>
r.Field<int>("BranchID")).ToArray();
            if(branchID.Length>0)
            {
                for(int i=0;i<=branchID.Length-1;i++)
                {
                    branchIDCBB.Items.Add(branchID[i]);
                }
            }

        }

        private void dbSaveBtn_Click(object sender, EventArgs e)
        {

            dataSetting.mysqlHost = dbHostTxt.Text;
            dataSetting.mysqlCentralDBName = dbNameTxt.Text;
            dataSetting.mysqlUsername = dbUserTxt.Text;
            dataSetting.mysqlPassword = dbPassTxt.Text;
            dataSetting.Save();
            dbDspLbl.Text = "Host: " + dataSetting.mysqlHost + "\nDatabase Name: " +
dataSetting.mysqlCentralDBName + "\nDatabase User: " + dataSetting.mysqlUsername +
"\nDatabase Password: " + dataSetting.mysqlPassword;
            MessageBox.Show("Database is changed successfully", "Configuration
Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }

        private void remoteSave_Click(object sender, EventArgs e)
        {
            SQLCommand = new MySqlCommand("update branch set user='" +
remoteUserTxt.Text + "',Pass='" + remotePass.Text + "',Host='" + remoteHost.Text + "'
Where BranchID=" + branchIDCBB.Text + ";", connection);
            connection.Open();
            SQLCommand.ExecuteNonQuery();
            connection.Close();
            remoteBranchDspLbl.Text = "Host: " + dataSetting.mysqlHost + "\nDatabase
Name: " + dataSetting.mysqlCentralDBName + "\nDatabase User: " +
dataSetting.mysqlUsername + "\nDatabase Password: " + dataSetting.mysqlPassword;
            MessageBox.Show("Database is changed successfully", "Configuration
Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AccountManage.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using MySql.Data.MySqlClient;

namespace DataAccess
{
    public partial class AccountManage : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        public string[] userDB, PassDB, PositionDB;
        public AccountManage()
        {
            InitializeComponent();
            accDspLbl.Text = null;
            accDspLbl.Visible = false;
            userTxt.Text = null;
            userTxt.Visible = false;
            userLbl.Visible = false;
            passTxt.Text = null;
            passTxt.Visible = false;
            passLbl.Visible = false;
            positionGb.Visible = false;
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            MySqlDataAdapter loadAccount = new MySqlDataAdapter("select* from
accountaccess;", connection);
            DataTable accountTB = new DataTable();
            loadAccount.Fill(accountTB);
            userDB = accountTB.AsEnumerable().Select(r =>
r.Field<string>("User")).ToArray();
```

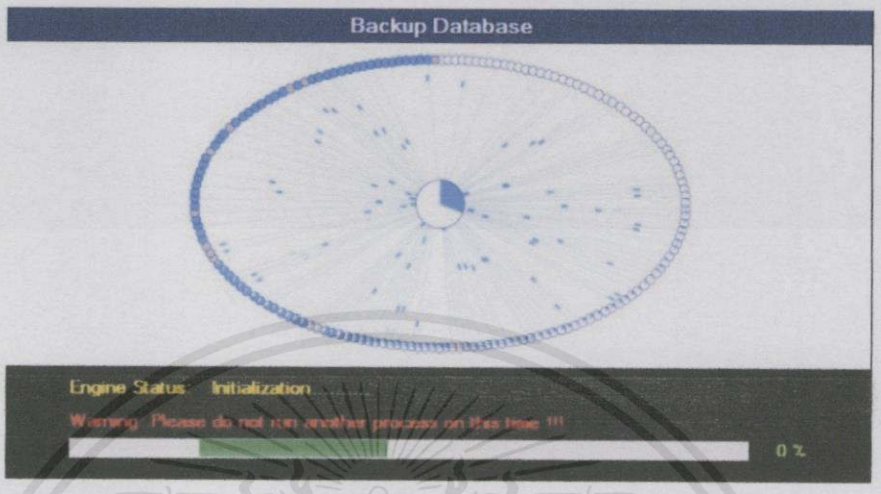
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น โดยไม่ได้รับอนุญาตจากเจ้าของเอกสารก่อน และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            PassDB = accountTB.AsEnumerable().Select(r =>
r.Field<string>("Pass")).ToArray();
            PositionDB = accountTB.AsEnumerable().Select(r =>
r.Field<string>("Position")).ToArray();
        }

        private void doItBtn_Click(object sender, EventArgs e)
        {
            var selectMode =
manageModeGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
            if (selectMode.Text == "Delete")
            {
                var positionCheck =
positionGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
                SQLCommand = new MySqlCommand("delete from accountaccess where User='"
+ userTxt.Text + "' and Position='" + positionCheck.Text + "';", connection);
                connection.Open();
                SQLCommand.ExecuteNonQuery();
                connection.Close();
                accDspLbl.Visible = true;
                accDspLbl.Text = "/////// This account was delete /////// \nUser: " +
userTxt.Text + "\nPosition: " + positionCheck.Text;
            }
            else if (selectMode.Text == "Create")
            {
                bool isUser = false;
                accDspLbl.Text = null;
                isUser = userDB.Contains(userTxt.Text);
                if (isUser == false)
                {
                    var check =
positionGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
                    SQLCommand = new MySqlCommand("insert into
accountaccess(User,Pass,Position) values('" + userTxt.Text + "','" + passTxt.Text +
"','" + check.Text + "');", connection);
                    connection.Open();
                    SQLCommand.ExecuteNonQuery();
                    connection.Close();
                    accDspLbl.Visible = true;
                    accDspLbl.Text = "/////// New account was created /////// \nUser:
" + userTxt.Text + "\nPass: " + passTxt.Text + "\nPosition: " + check.Text;
                }
            }
            userTxt.Text = null;
            userTxt.Visible = false;
            userLbl.Visible = false;
            passTxt.Text = null;
            passTxt.Visible = false;
            passLbl.Visible = false;
            positionGb.Visible = false;
        }

        private void step1Btn_Click(object sender, EventArgs e)
        {
            var selectMode =
manageModeGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
            switch (selectMode.Text)
            {
                case "Create":
                    accDspLbl.Visible = false;
                    userTxt.Visible = true;
                    userLbl.Visible = true;
                    passTxt.Visible = true;
                    passTxt.Enabled = true;
                    passLbl.Visible = true;
                    positionGb.Visible = true;
                    break;
                case "Delete":
                    accDspLbl.Visible = false;
                    userTxt.Visible = true;
                    userLbl.Visible = true;
                    passTxt.Visible = true;
                    passTxt.Enabled = false;
                    passLbl.Visible = true;
                    positionGb.Visible = true;
                    break;
            }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
    }
}
```

Form1.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Data.SqlClient;
using System.Xml;
using System.Threading;

namespace KMITL_SHOP_CallBackup
{
    public partial class callBackupFrm : Form
    {
        Setting setFile = new Setting();
        MySqlConnection connectionCentral,connectionBranch;
        MySqlCommand SQLCommand;
        string CFGPath = (Environment.GetEnvironmentVariable("KMITL_SHOP",
EnvironmentVariableTarget.Machine)) + "\\CallBackup_Config.xml";
        string MySQLStrConnectionCentral, MySQLStrConnectionBranch;
        int[] BranchID;
        string[] User, Pass, Host;
        int memberOfBranch = 0;
        bool isDone = false;
        public callBackupFrm()
        {
            InitializeComponent();
            loadCFG();
            statusPgb.Minimum = 0;
            statusPgb.Maximum = 100;
            engineStatusLbl.Text = "Initialization.....";
            transferAnimatePCB.Image = Properties.Resources.transferAnimate;
            MySQLStrConnectionCentral = "SERVER=" + setFile.dbHost + ";" + "DATABASE="
+ setFile.dbName + ";" + "UID=" + setFile.dbUser + ";" + "PASSWORD=" + setFile.dbPass;
            connectionCentral = new MySqlConnection(MySQLStrConnectionCentral);
            if (connectionCentral.State == ConnectionState.Open)
            {
                Console.Beep();
                connectionCentral.Close();
            }
            MySqlDataAdapter loadBranch = new MySqlDataAdapter("Select
BranchID,User,Pass,Host from Branch", connectionCentral);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            DataTable loadBranchTB = new DataTable();
            loadBranch.Fill(loadBranchTB);
            BranchID = loadBranchTB.AsEnumerable().Select(r =>
r.Field<int>("BranchID")).ToArray();
            User = loadBranchTB.AsEnumerable().Select(r =>
r.Field<string>("User")).ToArray();
            Pass = loadBranchTB.AsEnumerable().Select(r =>
r.Field<string>("Pass")).ToArray();
            Host = loadBranchTB.AsEnumerable().Select(r =>
r.Field<string>("Host")).ToArray();
            memberOfBranch = BranchID.Length;
            time2Start.Start();

        }

        #region Load Configuraton File
        public void loadCFG()
        {
            XmlReader readTxt = XmlReader.Create(CFGPath);
            while (readTxt.Read())
            {
                if (readTxt.IsStartElement())
                {
                    switch (readTxt.Name)
                    {
                        case "Attribute":
                            string attribute = readTxt["Name"];
                            if (attribute != null)
                            {
                                readTxt.Read();
                                string context = readTxt.Value.Trim();
                                switch (attribute)
                                {
                                    case "dbHost":
                                        setFile.dbHost = context;
                                        setFile.Save();
                                        break;
                                    case "dbName":
                                        setFile.dbName = context;
                                        setFile.Save();
                                        break;
                                    case "dbUser":
                                        setFile.dbUser = context;
                                        setFile.Save();
                                        break;
                                    case "dbPass":
                                        setFile.dbPass = context;
                                        setFile.Save();
                                        break;
                                }
                            }
                            break;
                    }
                }
            }
        }

        #endregion

        public void backUp()
        {
            engineStatusLbl.Text = "Running...";

            int stepValue = 100 / memberOfBranch;
            for (int i = 0; i <= memberOfBranch - 1;i++ )
            {
                statusPgb.Value = (((stepValue)/5)*(i+1));//-------------------------
------ 1/5 of Loop
                percentValueLbl.Text = statusPgb.Value.ToString()+" %";
                string[] barcode=null,product_status=null;
                int[] branchID = null, quantity = null, YellowTrig = null, RedTrig =
null, Set_Quantity = null;
                int memInLoadData=0;

                MySQLStrConnectionBranch = "";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                MySQLStrConnectionBranch = "SERVER=" + Host[i] + ";" +
"DATABASE=Kmitl_Shop_B" + BranchID[i] + ";" + "UID=" + User[i] + ";" + "PASSWORD=" +
Pass[i];
                connectionBranch = new MySqlConnection(MySQLStrConnectionBranch);
                if (connectionBranch.State == ConnectionState.Open)
                {
                    Console.Beep();
                    connectionBranch.Close();
                }
                MySqlDataAdapter loadSellDA = new MySqlDataAdapter("select
BranchID,Barcode,Quantity from sell where date(UpdateTime)=date(now());",
connectionBranch);
                DataTable loadSellDT = new DataTable();
                MessageBox.Show(MySQLStrConnectionBranch);
                loadSellDA.Fill(loadSellDT);
                branchID = loadSellDT.AsEnumerable().Select(r =>
r.Field<int>("BranchID")).ToArray();
                barcode = loadSellDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
                quantity = loadSellDT.AsEnumerable().Select(r =>
r.Field<int>("Quantity")).ToArray();
                memInLoadData = barcode.Length;

                connectionCentral.Open();
                for(int j=0;j<=memInLoadData-1;j++)
                {
                    SQLCommand=new MySqlCommand();
                    SQLCommand.Connection=connectionCentral;
                    SQLCommand.CommandText = "add2BranchSellTB";
                    SQLCommand.CommandType = CommandType.StoredProcedure;
                    SQLCommand.Parameters.Add("?BranchID_Input", branchID[j]);
                    SQLCommand.Parameters["?BranchID_Input"].Direction =
ParameterDirection.Input;
                    SQLCommand.Parameters.Add("?Barcode_Input", barcode[j]);
                    SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;
                    SQLCommand.Parameters.Add("?Quantity_Input", quantity[j]);
                    SQLCommand.Parameters["?Quantity_Input"].Direction =
ParameterDirection.Input;
                    SQLCommand.ExecuteNonQuery();
                }

                connectionCentral.Close();
                barcode = null;
                branchID = null;
                quantity = null;
                memInLoadData = 0;

                statusPgb.Value = (((2 * stepValue) / 5) * (i + 1));//----------------
---------------- 2/5 of Loop
                percentValueLbl.Text = statusPgb.Value.ToString() + " %";
                SQLCommand = new MySqlCommand("Truncate order_pending;",
connectionCentral);
                connectionCentral.Open();
                SQLCommand.ExecuteNonQuery();
                connectionCentral.Close();

                MySqlDataAdapter loadDialyOrderDA = new MySqlDataAdapter("select
BranchID,Barcode,Quantity from Dialy_Order where date(UpdateTime)=date(now());",
connectionBranch);
                DataTable loadDialyOrderDT = new DataTable();
                loadDialyOrderDA.Fill(loadDialyOrderDT);
                branchID = loadDialyOrderDT.AsEnumerable().Select(r =>
r.Field<int>("BranchID")).ToArray();
                barcode = loadDialyOrderDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
                quantity = loadDialyOrderDT.AsEnumerable().Select(r =>
r.Field<int>("Quantity")).ToArray();
                memInLoadData = barcode.Length;
                connectionCentral.Open();
                for (int j = 0; j <= memInLoadData - 1; j++)
                {
                    SQLCommand = new MySqlCommand("insert into
order_pending(BranchID,Barcode,Quantity) values(" + branchID[j] + ",'" + barcode[j] +
"'," + quantity[j] + ");", connectionCentral);
                    SQLCommand.ExecuteNonQuery();
                }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                connectionCentral.Close();


                SQLCommand = new MySqlCommand("Truncate
Dialy_Order;",connectionBranch);
                connectionBranch.Open();
                SQLCommand.ExecuteNonQuery();
                connectionBranch.Close();


                barcode = null;
                branchID = null;
                quantity = null;
                memInLoadData = 0;
                statusPgb.Value = (((3 * stepValue) / 5) * (i + 1));//----------------
---------------- 3/5 of Loop
                percentValueLbl.Text = statusPgb.Value.ToString() + " %";

                MySqlDataAdapter loadBranchTriggerDA = new MySqlDataAdapter("select
BranchID,Barcode,YellowTrig,RedTrig from Branch_Trigger;", connectionBranch);
                DataTable loadBranchTriggerDT = new DataTable();
                loadBranchTriggerDA.Fill(loadBranchTriggerDT);
                branchID = loadBranchTriggerDT.AsEnumerable().Select(r =>
r.Field<int>("BranchID")).ToArray();
                barcode = loadBranchTriggerDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
                YellowTrig = loadBranchTriggerDT.AsEnumerable().Select(r =>
r.Field<int>("YellowTrig")).ToArray();
                RedTrig = loadBranchTriggerDT.AsEnumerable().Select(r =>
r.Field<int>("RedTrig")).ToArray();
                memInLoadData = barcode.Length;
                connectionCentral.Open();
                for (int j = 0; j <= memInLoadData - 1; j++)
                {
                    SQLCommand = new MySqlCommand("update Branch_Trigger set
YellowTrig="+YellowTrig[j]+",RedTrig="+RedTrig[j]+" where BranchID="+branchID[j]+" and
Barcode='"+barcode[j]+"';", connectionCentral);
                    SQLCommand.ExecuteNonQuery();
                }
                connectionCentral.Close();
                barcode = null;
                quantity = null;
                YellowTrig = null;
                RedTrig = null;
                memInLoadData = 0;
                statusPgb.Value = (((4 * stepValue) / 5) * (i + 1));//----------------
---------------- 4/5 of Loop
                percentValueLbl.Text = statusPgb.Value.ToString() + " %";
                MySqlDataAdapter loadBranchStockDA = new MySqlDataAdapter("select
Barcode,Set_Quantity,Quantity,Product_Status from Stock;", connectionBranch);
                DataTable loadBranchStockDT = new DataTable();
                loadBranchStockDA.Fill(loadBranchStockDT);
                barcode = loadBranchStockDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
                Set_Quantity = loadBranchStockDT.AsEnumerable().Select(r =>
r.Field<int>("Set_Quantity")).ToArray();
                quantity = loadBranchStockDT.AsEnumerable().Select(r =>
r.Field<int>("Quantity")).ToArray();
                product_status = loadBranchStockDT.AsEnumerable().Select(r =>
r.Field<string>("Product_Status")).ToArray();
                memInLoadData = barcode.Length;
                connectionCentral.Open();
                SQLCommand = new MySqlCommand("Truncate branch_stock;",
connectionCentral);
                SQLCommand.ExecuteNonQuery();
                for (int j = 0; j <= memInLoadData - 1; j++)
                {
                    SQLCommand = new MySqlCommand("insert into
Branch_Stock(BranchID,Barcode,Set_Quantity,Quantity,Product_Status) values(" +
branchID[0] + ",'" + barcode[j] + "'," + Set_Quantity[j] + "," + quantity[j] + ",'" +
product_status[j] + "');", connectionCentral);
                    SQLCommand.ExecuteNonQuery();
                }
                connectionCentral.Close();
                branchID = null;
                branchID = null;
                barcode = null;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                Set_Quantity = null;
                quantity = null;
                YellowTrig = null;
                RedTrig = null;
                product_status = null;
                memInLoadData = 0;
                statusPgb.Value = (stepValue * (i + 1));//----------------------------
---- 1 Loop
                percentValueLbl.Text = statusPgb.Value.ToString() + " %";
            }
            statusPgb.Value = 100;
            percentValueLbl.Text = statusPgb.Value.ToString() + " %";
            engineStatusLbl.ForeColor = Color.Green;
            engineStatusLbl.Text = "Backup is done";
            closeTrig.Start();
            engineStatusLbl.Text = "Backup is done ....<Prepare to close>";
        }

        private void close_trigger(object sender, EventArgs e)
        {
            Environment.Exit(0);
        }

        private void Form_Opened(object sender, EventArgs e)
        {
            time2Start.Stop();
            engineStatusLbl.Text = "System is running";
            backUp();
        }
    }
}
```

ภาคผนวก ข: Branch Back Office Application

AccountManage.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using MySql.Data.MySqlClient;

namespace DataAccess
{
```

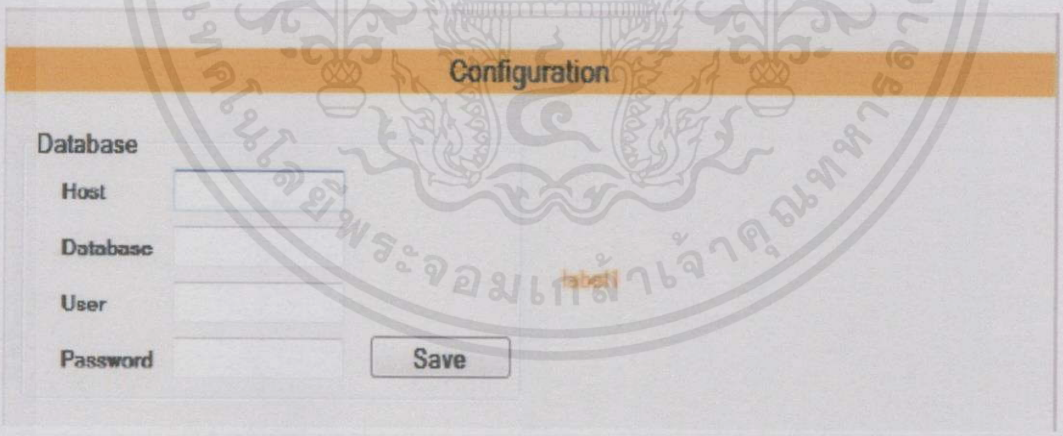
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    public partial class AccountManage : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;

        public string[] userDB, PassDB, PositionDB;
        public AccountManage()
        {
            InitializeComponent();
            accDspLbl.Text = null;
            accDspLbl.Visible = false;
            userTxt.Text = null;
            userTxt.Visible = false;
            userLbl.Visible = false;
            passTxt.Text = null;
            passTxt.Visible = false;
            passLbl.Visible = false;
            positionGb.Visible = false;
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            MySqlDataAdapter loadAccount = new MySqlDataAdapter("select* from
accountaccess;", connection);
            DataTable accountTB = new DataTable();
            loadAccount.Fill(accountTB);
            userDB = accountTB.AsEnumerable().Select(r =>
r.Field<string>("User")).ToArray();
            PassDB = accountTB.AsEnumerable().Select(r =>
r.Field<string>("Pass")).ToArray();
            PositionDB = accountTB.AsEnumerable().Select(r =>
r.Field<string>("Position")).ToArray();
        }

        private void accountCreateBtn_Click(object sender, EventArgs e)
        {
            var selectMode =
manageModeGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
            if(selectMode.Text=="Delete")
            {
                var positionCheck =
positionGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
                SQLCommand = new MySqlCommand("delete from accountaccess where User='"
+ userTxt.Text +"' and Position='" + positionCheck.Text +"';", connection);
                connection.Open();
                SQLCommand.ExecuteNonQuery();
                connection.Close();
                accDspLbl.Visible = true;
                accDspLbl.Text = "/////// This account was delete /////// \nUser: " +
userTxt.Text + "\nPosition: " + positionCheck.Text;
            }
            else if (selectMode.Text == "Create")
            {
                bool isUser = false;
                accDspLbl.Text = null;
                isUser = userDB.Contains(userTxt.Text);
                if (isUser == false)
                {
                    var check =
positionGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
                    SQLCommand = new MySqlCommand("insert into
accountaccess(User,Pass,Position) values('" + userTxt.Text + "','" + passTxt.Text +
"','" + check.Text + "');", connection);
                    connection.Open();
                    SQLCommand.ExecuteNonQuery();
                    connection.Close();
                    accDspLbl.Visible = true;
                    accDspLbl.Text = "/////// New account was created /////// \nUser:
" + userTxt.Text + "\nPass: " + passTxt.Text + "\nPosition: " + check.Text;
                }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            }
            userTxt.Text = null;
            userTxt.Visible = false;
            userLbl.Visible = false;
            passTxt.Text = null;
            passTxt.Visible = false;
            passLbl.Visible = false;
            positionGb.Visible = false;
        }

        private void step1Btn_Click(object sender, EventArgs e)
        {
            var selectMode =
manageModeGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked);
            switch(selectMode.Text)
            {
                case "Create":
                    accDspLbl.Visible = false;
                    userTxt.Visible = true;
                    userLbl.Visible = true;
                    passTxt.Visible = true;
                    passTxt.Enabled = true;
                    passLbl.Visible = true;
                    positionGb.Visible = true;
                    break;
                case "Delete":
                    accDspLbl.Visible = false;
                    userTxt.Visible = true;
                    userLbl.Visible = true;
                    passTxt.Visible = true;
                    passTxt.Enabled = false;
                    passLbl.Visible = true;
                    positionGb.Visible = true;
                    break;
            }
        }
    }
}
```

ConfigurationFrm.cs

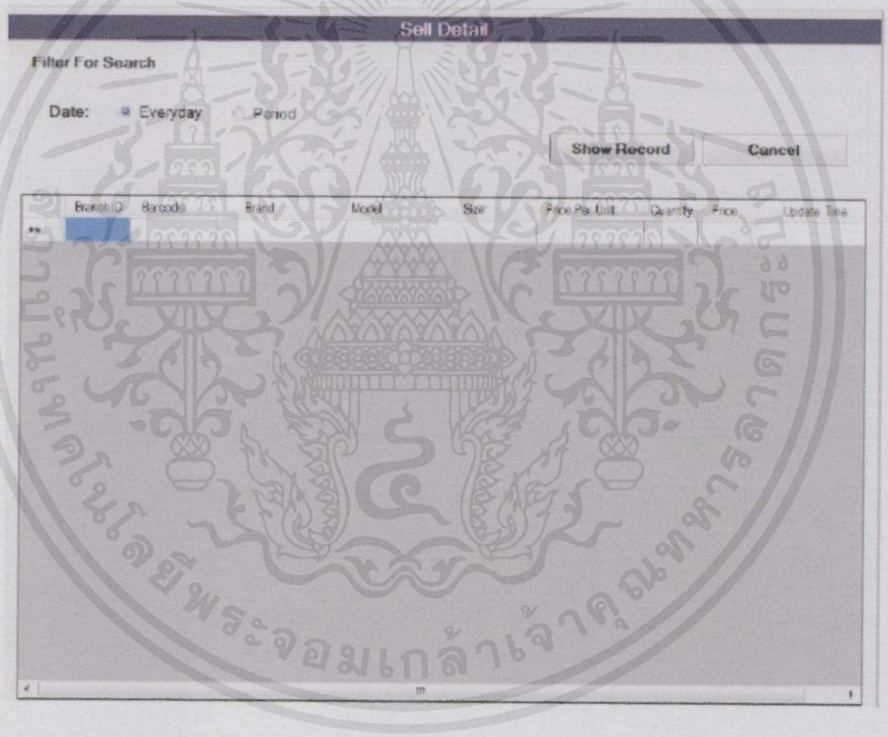



```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DataAccess
{
    public partial class Configuration : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        public Configuration()
        {
            InitializeComponent();
        }

        private void dbSaveBtn_Click(object sender, EventArgs e)
        {

            dataSetting.mysqlHost = dbHostTxt.Text;
            dataSetting.mysqlCentralDBName = dbNameTxt.Text;
            dataSetting.mysqlUsername = dbUserTxt.Text;
            dataSetting.mysqlPassword = dbPassTxt.Text;
            dataSetting.Save();
            dbDspLbl.Text = "Host: " + dataSetting.mysqlHost + "\nDatabase Name: " +
dataSetting.mysqlCentralDBName + "\nDatabase User: " + dataSetting.mysqlUsername +
"\nDatabase Password: " + dataSetting.mysqlPassword;
            MessageBox.Show("Database is changed successfully", "Configuration
Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}
```

SellDetail.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using MySql.Data.MySqlClient;

namespace DataAccess
{
    public partial class SellDetail : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        DataTable loadSellTB = new DataTable();
        public int[] allBranchID;
        public SellDetail()
        {
            InitializeComponent();
            fromDateLbl.Visible = false;
            fromDateMtb.Visible = false;
            toDateLbl.Visible = false;
            toDateMtb.Visible = false;
            fromDateMtb.Mask = "0000/00/00";
            toDateMtb.Mask = "0000/00/00";
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }

            showRecordDGV.AutoGenerateColumns = false;
            showRecordDGV.Columns["BranchID"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Barcode"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Brand"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Model"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Size"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["PricePerUnit"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Quantity"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["Price"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            showRecordDGV.Columns["UpdateTime"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
        }

        #region Date to search

        private void fromDateMtb_InputReject(object sender, MaskInputRejectedEventArgs
e)
        {
            if (fromDateMtb.MaskFull)
            {
                fromDateTt.ToolTipTitle = "Input Rejected - Too Much Data";
                fromDateTt.Show("You cannot enter any more data into the date field.
Delete some characters in order to insert more data.", fromDateMtb, 0, -20, 5000);
            }
            else if (e.Position == fromDateMtb.Mask.Length)
            {
                fromDateTt.ToolTipTitle = "Input Rejected - End of Field";
                fromDateTt.Show("You cannot add extra characters to the end of this
date field.", fromDateMtb, 0, -20, 5000);
            }
            else
            {
                fromDateTt.ToolTipTitle = "Input Rejected";
                fromDateTt.Show("You can only add numeric characters (0-9) into this
date field.", fromDateMtb, 0, -20, 5000);
            }
        }
        private void toDateMtb_MaskInputRejected(object sender,
MaskInputRejectedEventArgs e)
        {
            if (toDateMtb.MaskFull)
            {
                toDateTt.ToolTipTitle = "Input Rejected - Too Much Data";
                toDateTt.Show("You cannot enter any more data into the date field.
Delete some characters in order to insert more data.", toDateMtb, 0, -20, 5000);
            }
            else if (e.Position == toDateMtb.Mask.Length)
            {
```

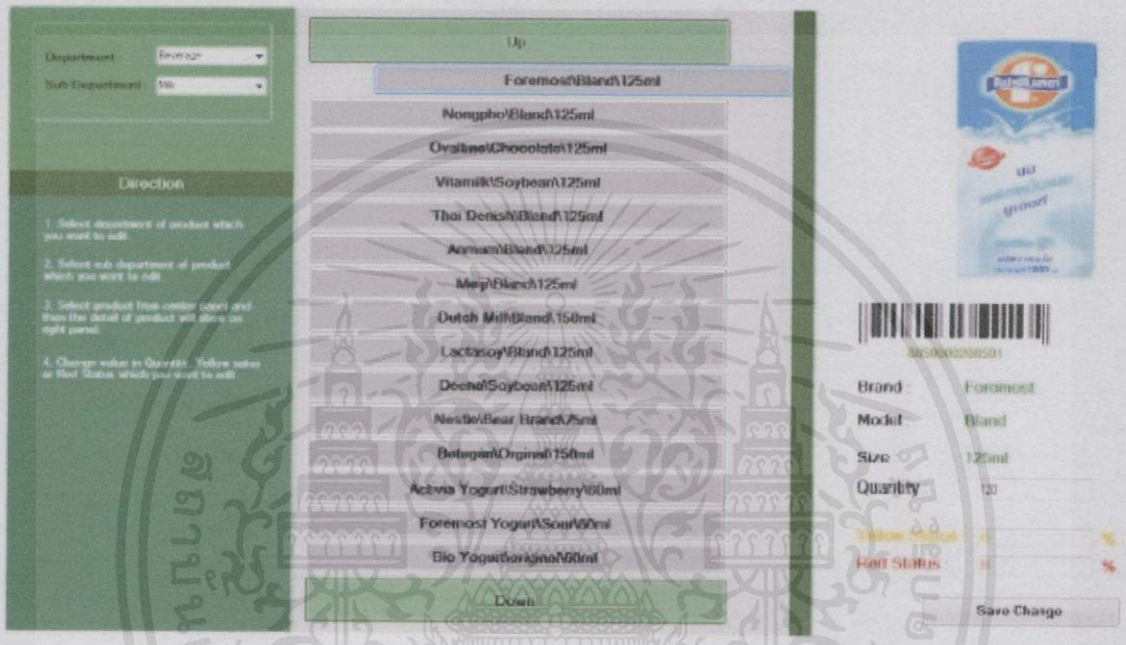
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                toDateTt.ToolTipTitle = "Input Rejected - End of Field";
                toDateTt.Show("You cannot add extra characters to the end of this date
field.", toDateMtb, 0, -20, 5000);
            }
            else
            {
                toDateTt.ToolTipTitle = "Input Rejected";
                toDateTt.Show("You can only add numeric characters (0-9) into this
date field.", toDateMtb, 0, -20, 5000);
            }
        }

        private void toDateMtb_KeyDown(object sender, KeyEventArgs e)
        {
            toDateTt.Hide(toDateMtb);
        }
        private void fromDateMtb_KeyDown(object sender, KeyEventArgs e)
        {
            fromDateTt.Hide(fromDateMtb);
        }

        #endregion

        private void showRecordBtn_Click(object sender, EventArgs e)
        {
            loadSellTB.Clear();
            string MySQLStrCommandHead = "select BranchID as BranchID,Barcode as
Barcode,Brand as Brand,Model as Model,Size as Size,PricePerContain as 'Price Per
Contain',Quantity as Quantity,Price as Price,UpdateTime as 'Update Time' from sell ";
            string MySQLStrCommandTail = "";

            #region create MySQL String Command

            if(everyDateRtb.Checked==true)
            {
                MySQLStrCommandTail = "";
            }
            else if(periodDayRtn.Checked==true)
            {
                MySQLStrCommandTail = "where date(updateTime) between date('" +
fromDateMtb.Text + "') and date('" + toDateMtb.Text + "') ";
            }

            #endregion

            string MySQLStrCommand = MySQLStrCommandHead + MySQLStrCommandTail + ";";
            MySqlDataAdapter loadSell = new MySqlDataAdapter(MySQLStrCommand,
connection);
            loadSell.Fill(loadSellTB);
            showRecordDGV.Columns["BranchID"].DataPropertyName = "BranchID";
            showRecordDGV.Columns["Barcode"].DataPropertyName = "Barcode";
            showRecordDGV.Columns["Brand"].DataPropertyName = "Brand";
            showRecordDGV.Columns["Model"].DataPropertyName = "Model";
            showRecordDGV.Columns["Size"].DataPropertyName = "Size";
            showRecordDGV.Columns["PricePerUnit"].DataPropertyName = "Price Per
Contain";
            showRecordDGV.Columns["Quantity"].DataPropertyName = "Quantity";
            showRecordDGV.Columns["Price"].DataPropertyName = "Price";
            showRecordDGV.Columns["UpdateTime"].DataPropertyName = "Update Time";
            showRecordDGV.DataSource = loadSellTB;
            showRecordDGV.Show();
        }

        private void periodDayRtn_CheckedChanged(object sender, EventArgs e)
        {
            if(periodDayRtn.Checked==true)
            {
                fromDateLbl.Visible = true;
                fromDateMtb.Visible = true;
                toDateLbl.Visible = true;
                toDateMtb.Visible = true;
            }
            else
            {
                fromDateLbl.Visible = false;
                fromDateMtb.Visible = false;
                toDateLbl.Visible = false;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                toDateMtb.Visible = false;
            }
        }

        private void cancelBtn_Click(object sender, EventArgs e)
        {
            loadSellTB.Clear();
        }
    }
}
```

SetQuantityInBranch.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.IO;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;
using System.Speech.Synthesis;

namespace DataAccess
{
    public partial class SetQuantityInBranch : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        public SetQuantityInBranch()
        {
            InitializeComponent();
            setFrm();
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น โดยผู้ใช้งานต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                Console.Beep();
                connection.Close();
            }
            globalVar.department = null;
            string loadDept = "select distinct Department from Department;";
            MySqlDataAdapter loadDepartment = new MySqlDataAdapter(loadDept,
connection);
            DataTable department = new DataTable();
            loadDepartment.Fill(department);
            globalVar.department = department.AsEnumerable().Select(r =>
r.Field<string>("Department")).ToArray();
            deptCbb.Text = null;
            deptCbb.Items.Clear();
            deptCbb.Items.AddRange(globalVar.department);
            barCodeImg.Image = Properties.Resources.barcode;

        }

        public void setFrm()
        {
            for(int i=1;i<=15;i++)
            {
                string dispLblStr = "prodBtn" + Convert.ToString(i);
                Control[] dispLbl = this.Controls.Find(dispLblStr,true);
                if(dispLbl!=null)
                {
                    dispLbl[0].Text = null;
                    dispLbl[0].Visible=false;
                }
            }
            upBtn.Visible = false;
            upBtn.Enabled = false;
            downBtn.Visible = false;
            downBtn.Enabled = false;
            prodSplitter.Visible = false;
            productPcb.Visible = false;
            barCodeLbl.Visible = false;
            barCodeLbl.Text = null;
            barCodeImg.Visible = false;
            brandLbl.Visible = false;
            brandDispLbl.Text = null;
            brandDispLbl.Visible = false;
            modelLbl.Visible = false;
            modelDispLbl.Text = null;
            modelDispLbl.Visible = false;
            sizeLbl.Visible = false;
            sizeDispLbl.Text = null;
            sizeDispLbl.Visible = false;
            quantityLbl.Visible = false;
            setQuantityTxt.Text = null;
            setQuantityTxt.Visible = false;
            redTrigHeadLbl.Visible = false;
            redTrigPercentLbl.Visible = false;
            redStatusTxt.Text = null;
            redStatusTxt.Visible = false;
            yellowStatusTxt.Text = null;
            yellowStatusTxt.Visible = false;
            yellowTrigHeadLbl.Visible = false;
            yellowTrigPercentLbl.Visible = false;
            saveBtn.Visible = false;
            globalVar.prouductPage = 1;
            setBtn_initialPosition();
        }

        public void setBtn_initialPosition()
        {
            for (int i = 1; i <= 15; i++)
            {
                string dispLblStr = "prodBtn" + Convert.ToString(i);
                Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                if (dispLbl != null)
                {
                    dispLbl[0].Left = 324;
                }
            }
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        private void deptCbb_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (deptCbb.SelectedIndex != -1)
            {
                setFrm();
                globalVar.sub_department = null;
                string loadSubDept = "select distinct sub_Department from department
Where department='" + deptCbb.Text + "';";
                MySqlDataAdapter loadSubDepartment = new MySqlDataAdapter(loadSubDept,
connection);
                DataTable subDepartment = new DataTable();
                loadSubDepartment.Fill(subDepartment);
                globalVar.sub_department = subDepartment.AsEnumerable().Select(r =>
r.Field<string>("Sub_Department")).ToArray();
                subDeptCbb.Text = null;
                subDeptCbb.Items.Clear();
                subDeptCbb.Items.AddRange(globalVar.sub_department);
            }

        }

        private void subDeptCbb_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (subDeptCbb.SelectedIndex != -1)
            {
                setFrm();
                globalVar.brand = null;
                globalVar.model = null;
                globalVar.size = null;
                globalVar.barcode2Trig = null;
                globalVar.yellowTrig = null;
                globalVar.redTrig = null;

                globalVar.productDisp = null;
                prodSplitter.Visible = true;
                refreshDatabase();
                globalVar.productLength = globalVar.brand.Length;

                if(globalVar.productLength >= 15)
                {
                    upBtn.Visible = true;
                    upBtn.Enabled = true;
                    downBtn.Visible = true;
                    downBtn.Enabled = true;
                    for (int i = 1; i <= 15; i++)
                    {
                        string displbl = "prodBtn" + Convert.ToString(i);
                        Control[] displblCtrl = this.Controls.Find(displbl, true);
                        if (displblCtrl != null)
                        {
                            displblCtrl[0].Visible = true;
                            displblCtrl[0].Text = globalVar.brand[i - 1] + "\\" +
globalVar.model[i - 1] + "\\" + globalVar.size[i - 1];
                        }
                    }
                }
                else
                {
                    upBtn.Visible = true;
                    downBtn.Visible = true;
                    for (int i = 1; i <= globalVar.productLength; i++)
                    {
                        string displbl = "prodBtn" + Convert.ToString(i);
                        Control[] displblCtrl = this.Controls.Find(displbl, true);
                        if (displblCtrl != null)
                        {
                            displblCtrl[0].Visible = true;
                            displblCtrl[0].Text = globalVar.brand[i - 1] + "\\" +
globalVar.model[i - 1] + "\\" + globalVar.size[i - 1];
                        }
                    }
                }
            }
        }

        public void refreshDatabase()
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            globalVar.barcode2Trig=null;
            globalVar.yellowTrig=null;
            globalVar.redTrig=null;
            globalVar.barCode=null;
            globalVar.brand=null;
            globalVar.model=null;
            globalVar.size=null;
            globalVar.quantity = null;
            MySqlDataAdapter loadTrigState = new MySqlDataAdapter("select
Barcode,YellowTrig,RedTrig from branch_trigger;", connection);
            DataTable loadTrigDT = new DataTable();
            loadTrigState.Fill(loadTrigDT);
            globalVar.barcode2Trig = loadTrigDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
            globalVar.yellowTrig = loadTrigDT.AsEnumerable().Select(r =>
r.Field<int>("YellowTrig")).ToArray();
            globalVar.redTrig = loadTrigDT.AsEnumerable().Select(r =>
r.Field<int>("RedTrig")).ToArray();
            string loadProd = "select
product.BarCode,central_product.Brand,central_product.Model,central_product.Size,Stock
.Set_quantity from product left join central_product on product.BarCode =
central_product.BarCode and product.sub_department ='" + subDeptCbb.Text + "' left
join stock on product.barcode=stock.barcode where brand is not null;";
            MySqlDataAdapter loadProduct = new MySqlDataAdapter(loadProd, connection);
            DataTable product = new DataTable();
            loadProduct.Fill(product);
            globalVar.barCode = product.AsEnumerable().Select(r =>
r.Field<string>("BarCode")).ToArray();
            globalVar.brand = product.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
            globalVar.model = product.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
            globalVar.size = product.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
            globalVar.quantity = product.AsEnumerable().Select(r =>
r.Field<int>("Set_Quantity")).ToArray();
        }
        private void prodBtn1_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage-1));
            setBtn_initialPosition();
            prodBtn1.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }


            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);

            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible=true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn2_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 1;
            setBtn_initialPosition();
            prodBtn2.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn3_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 2;
            setBtn_initialPosition();
            prodBtn3.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn4_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 3;
            setBtn_initialPosition();
            prodBtn4.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn5_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 4;
            setBtn_initialPosition();
            prodBtn5.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        private void prodBtn6_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 5;
            setBtn_initialPosition();
            prodBtn6.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn7_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 6;
            setBtn_initialPosition();
            prodBtn7.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn8_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 7;
            setBtn_initialPosition();
            prodBtn8.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn9_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 8;
            setBtn_initialPosition();
            prodBtn9.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn10_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 9;
            setBtn_initialPosition();
            prodBtn10.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn11_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 10;
            setBtn_initialPosition();
            prodBtn11.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);

            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn12_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 11;
            setBtn_initialPosition();
            prodBtn12.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn13_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 12;
            setBtn_initialPosition();
            prodBtn13.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น มีการตรวจสอบการคัดลอกเนื้อหาและข้อความอิงลึกด้วยเอกสารทุกครั้งที่มีการนำไปใช้

```
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn14_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 13;
            setBtn_initialPosition();
            prodBtn14.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if (indxOfTrig != -1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }

        private void prodBtn15_Click(object sender, EventArgs e)
        {
            int indx = (15 * (globalVar.prouductPage - 1)) + 14;
            setBtn_initialPosition();
            prodBtn15.Left = 394;
            barCodeLbl.Text = globalVar.barCode[indx];
            brandDispLbl.Text = globalVar.brand[indx];
            modelDispLbl.Text = globalVar.model[indx];
            sizeDispLbl.Text = globalVar.size[indx];
            setQuantityTxt.Text = Convert.ToString(globalVar.quantity[indx]);

            yellowStatusTxt.Text = "";
            redStatusTxt.Text = "";
            int indxOfTrig = Array.IndexOf(globalVar.barcode2Trig, barCodeLbl.Text);
            if(indxOfTrig!=-1)
            {
                yellowStatusTxt.Text =
Convert.ToString(globalVar.yellowTrig[indxOfTrig]);
                redStatusTxt.Text = Convert.ToString(globalVar.redTrig[indxOfTrig]);
            }

            string productImgStr = "select productImage from central_product where
barcode='" + barCodeLbl.Text + "';";
            MySqlDataAdapter loadProductImg = new MySqlDataAdapter(productImgStr,
connection);
            DataTable productImg = new DataTable();
            loadProductImg.Fill(productImg);
            byte[] img = new byte[0];
            img = (byte[])productImg.Rows[0][0];
            MemoryStream ms = new MemoryStream(img);
            productPcb.Image = Bitmap.FromStream(ms);


            productPcb.Visible = true;
            barCodeImg.Visible = true;
            barCodeLbl.Visible = true;
            brandLbl.Visible = true;
            brandDispLbl.Visible = true;
            modelLbl.Visible = true;
            modelDispLbl.Visible = true;
            sizeLbl.Visible = true;
            sizeDispLbl.Visible = true;
            quantityLbl.Visible = true;
            setQuantityTxt.Visible = true;
            redStatusTxt.Visible = true;
            redTrigHeadLbl.Visible = true;
            redTrigPercentLbl.Visible = true;
            yellowTrigHeadLbl.Visible = true;
            yellowTrigPercentLbl.Visible = true;
            yellowStatusTxt.Visible = true;
            saveBtn.Visible = true;
            refreshDatabase();
        }
```

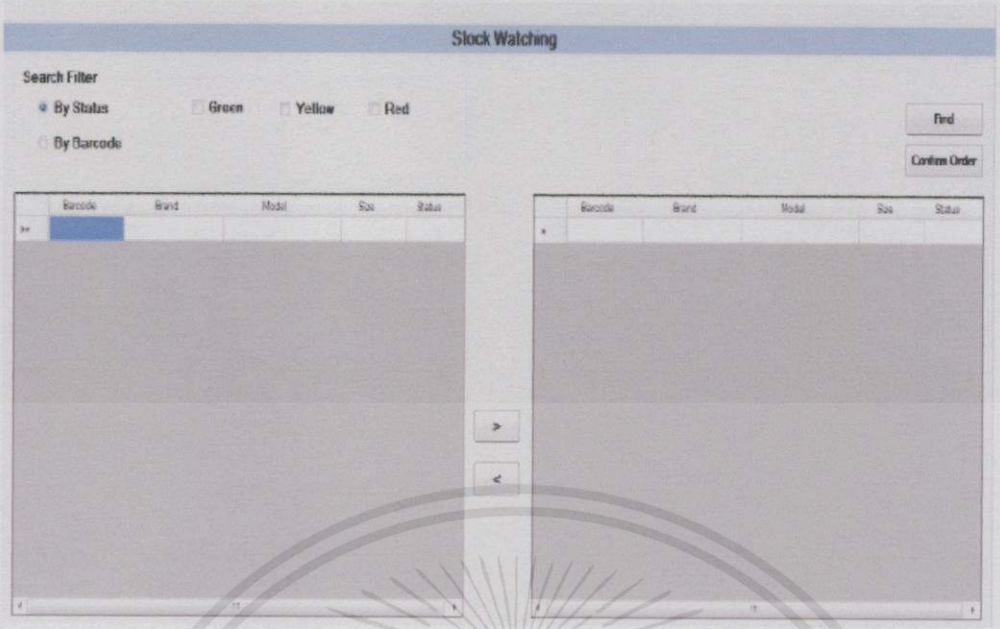
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น โดยไม่ได้รับอนุญาตจากเจ้าของเอกสาร ต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        private void downBtn_Click(object sender, EventArgs e)
        {
            for (int i = 1; i <= 15; i++)
            {
                string dispLblStr = "prodBtn" + Convert.ToString(i);
                Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                if (dispLbl != null)
                {
                    dispLbl[0].Left = 324;
                    dispLbl[0].Text = null;
                    dispLbl[0].Visible = false;
                }
            }

            if(globalVar.prouductPage>=1)
            {
                if((globalVar.productLength - ((15*(globalVar.prouductPage+1))))<=15)
                {
                    for (int i = 1; i <= (globalVar.productLength - ((15 *
(globalVar.prouductPage)))); i++)
                    {
                        string dispLblStr = "prodBtn" + Convert.ToString(i);
                        Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                        if (dispLbl != null)
                        {   dispLbl[0].Visible = true;
                        dispLbl[0].Text = globalVar.brand[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.model[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.size[(15 *
globalVar.prouductPage) + (i - 1)];
                        }
                        upBtn.Visible = true;
                        downBtn.Visible = false;
                    }
                    globalVar.prouductPage = globalVar.prouductPage + 1;
                }
                else
                {

                    for (int i = 1; i <= 15; i++)
                    {
                        string dispLblStr = "prodBtn" + Convert.ToString(i);
                        Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                        if (dispLbl != null)
                        {
                            dispLbl[0].Visible = true;
                            dispLbl[0].Text = globalVar.brand[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.model[(15 *
globalVar.prouductPage) + (i - 1)] + "\\" + globalVar.size[(15 *
globalVar.prouductPage) + (i - 1)];
                        }
                    }
                    globalVar.prouductPage = globalVar.prouductPage + 1;
                }
            }
        }

        private void upBtn_Click(object sender, EventArgs e)
        {
            if (globalVar.prouductPage > 1)
            {
                for (int i = 1; i <= 15; i++)
                {
                    string dispLblStr = "prodBtn" + Convert.ToString(i);
                    Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                    if (dispLbl != null)
                    {
                        dispLbl[0].Left = 324;
                        dispLbl[0].Text = null;
                        dispLbl[0].Visible = false;
                    }
                }
                downBtn.Visible = true;
                globalVar.prouductPage = globalVar.prouductPage - 1;
                for (int i = 1; i <= 15; i++)
                {
                    string dispLblStr = "prodBtn" + Convert.ToString(i);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    Control[] dispLbl = this.Controls.Find(dispLblStr, true);
                    if (dispLbl != null)
                    {
                        dispLbl[0].Visible = true;
                        dispLbl[0].Text = globalVar.brand[(15 *
(globalVar.prouductPage-1)) + (i - 1)] + "\\" + globalVar.model[(15 *
(globalVar.prouductPage-1)) + (i - 1)] + "\\" + globalVar.size[(15 *
(globalVar.prouductPage-1)) + (i - 1)];
                    }
                }

            }
            else
            {
                upBtn.Visible = false;
            }
        }

        private void saveBtn_Click(object sender, EventArgs e)
        {
            if(setQuantityTxt.Text!="")
            {
                connection.Open();
                string updateQuantity = "update stock set Set_Quantity=" +
setQuantityTxt.Text + " where barcode='" + barCodeLbl.Text +"';";
                SQLCommand = new MySqlCommand(updateQuantity, connection);
                SQLCommand.ExecuteNonQuery();

                string updateTrigger = "update Branch_Trigger set YellowTrig=" +
yellowStatusTxt.Text + ",RedTrig=" + redStatusTxt.Text + " where barcode='" +
barCodeLbl.Text + "';";
                SQLCommand = new MySqlCommand(updateTrigger, connection);
                SQLCommand.ExecuteNonQuery();
                connection.Close();

                MessageBox.Show("BarCode: " + barCodeLbl.Text + "\n" + "Brand: " +
brandDispLbl.Text + "\n" + "Model: " + modelDispLbl.Text + "\n" + "Size: " +
sizeDispLbl.Text + "\n has been updated", "Quantity change system",
MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
            else
            {
                MessageBox.Show("Error: you have to fill in quantity textbox",
"Quantity change system", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using MySql.Data.MySqlClient;
using System.Data.Common;

namespace DataAccess
{
    public partial class StockWatching : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        DataTable productSelectTB = new DataTable();
        DataTable productUSelectTB = new DataTable();
        BindingSource productAvailableBS = new BindingSource();
        BindingSource productSelectBS = new BindingSource();
        string[] selectedProduct = new string[5];
        bool isSelect = true;
        public StockWatching()
        {
            InitializeComponent();
            barcodeTxt.Visible = false;
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }

            productSelectTB.Columns.Add(new DataColumn("Barcode"));
            productSelectTB.Columns.Add(new DataColumn("Brand"));
            productSelectTB.Columns.Add(new DataColumn("Model"));
            productSelectTB.Columns.Add(new DataColumn("Size"));
            productSelectTB.Columns.Add(new DataColumn("Status"));

            productUSelectTB.Columns.Add(new DataColumn("Barcode"));
            productUSelectTB.Columns.Add(new DataColumn("Brand"));
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

```
            productUSelectTB.Columns.Add(new DataColumn("Model"));
            productUSelectTB.Columns.Add(new DataColumn("Size"));
            productUSelectTB.Columns.Add(new DataColumn("Status"));

            productAvailableDGV.AutoGenerateColumns = false;
            productAvailableDGV.Columns["Barcode"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productAvailableDGV.Columns["Brand"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productAvailableDGV.Columns["Model"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productAvailableDGV.Columns["Size"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productAvailableDGV.Columns["Status"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productAvailableDGV.Columns["Barcode"].DataPropertyName = "Barcode";
            productAvailableDGV.Columns["Brand"].DataPropertyName = "Brand";
            productAvailableDGV.Columns["Model"].DataPropertyName = "Model";
            productAvailableDGV.Columns["Size"].DataPropertyName = "Size";
            productAvailableDGV.Columns["Status"].DataPropertyName = "Status";
            productAvailableBS.DataSource = productUSelectTB;
            productAvailableDGV.DataSource = productAvailableBS;
            productAvailableDGV.Show();

            productSelectDGV.AutoGenerateColumns = false;
            productSelectDGV.Columns["BarcodeSelect"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productSelectDGV.Columns["BrandSelect"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productSelectDGV.Columns["ModelSelect"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productSelectDGV.Columns["SizeSelect"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productSelectDGV.Columns["StatusSelect"].HeaderCell.Style.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            productSelectDGV.Columns["BarcodeSelect"].DataPropertyName = "Barcode";
            productSelectDGV.Columns["BrandSelect"].DataPropertyName = "Brand";
            productSelectDGV.Columns["ModelSelect"].DataPropertyName = "Model";
            productSelectDGV.Columns["SizeSelect"].DataPropertyName = "Size";
            productSelectDGV.Columns["StatusSelect"].DataPropertyName = "Status";
            productSelectBS.DataSource = productSelectTB;
            productSelectDGV.DataSource = productSelectBS;
            productSelectDGV.Show();
        }

        private void byStatusRbtn_CheckedChanged(object sender, EventArgs e)
        {
            if(byStatusRbtn.Checked==true)
            {
                greenStatusCb.Visible = true;
                yellowStatusCb.Visible = true;
                redStatusCb.Visible = true;
            }
            else
            {
                greenStatusCb.Visible = false;
                yellowStatusCb.Visible = false;
                redStatusCb.Visible = false;
            }


        }

        private void byBarcodeRbtn_CheckedChanged(object sender, EventArgs e)
        {
            if(byBarcodeRbtn.Checked==true)
            {
                barcodeTxt.Text = null;
                barcodeTxt.Visible = true;
            }
            else
            {
                barcodeTxt.Text = null;
                barcodeTxt.Visible = false;
            }
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        #region Searching Function

        public void searching()
        {
            productUSelectTB.Clear();
            string filterSelect =
filterGb.Controls.OfType<RadioButton>().FirstOrDefault(r => r.Checked).Text;
            string stringSearch = "";
            switch (filterSelect)
            {
                case "By Status":

if(greenStatusCb.Checked==true&&yellowStatusCb.Checked==true&&redStatusCb.Checked==tru
e)
                    {
                        stringSearch = "select
Stock.Barcode,Central_product.Brand,Central_product.Model,Central_product.Size,Stock.P
roduct_Status from stock left join Central_product on
stock.Barcode=Central_product.Barcode";

                    }
                    else
if(greenStatusCb.Checked==true||yellowStatusCb.Checked==true||redStatusCb.Checked==tru
e)
                    {
                        stringSearch = "select
Stock.Barcode,Central_product.Brand,Central_product.Model,Central_product.Size,Stock.P
roduct_Status from stock left join Central_product on
stock.Barcode=Central_product.Barcode and (";
                        string
mid1StringSearch="",mid2StringSearch="",mid3StringSearch="";
                        if(greenStatusCb.Checked==true)
                        {
                            mid1StringSearch = "stock.product_status='G'";
                        }
                        else
                        {
                            mid1StringSearch = "";
                        }
                        if(yellowStatusCb.Checked==true)
                        {
                            mid2StringSearch = " or stock.product_status='Y'";
                        }
                        else
                        {
                            mid2StringSearch = "";

                        }
                        if(redStatusCb.Checked==true)
                        {
                            mid3StringSearch = " or stock.product_status='R'";
                        }
                        else
                        {
                            mid3StringSearch = "";
                        }
                        string midStringSearch = mid1StringSearch + mid2StringSearch +
mid3StringSearch;
                        if(midStringSearch.StartsWith(" or")==true)
                        {
                            midStringSearch = midStringSearch.Remove(0,3);
                        }
                        stringSearch = stringSearch + midStringSearch + ") where Brand
is not null;";
                    }

                    break;
                case "By Barcode":
                    stringSearch = "select
Stock.Barcode,Central_product.Brand,Central_product.Model,Central_product.Size,Stock.P
roduct_Status from stock left join Central_product on
stock.Barcode=Central_product.Barcode and Stock.barcode='" + barcodeTxt.Text + "'
where Brand is not null;";
                    break;

            }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            if(stringSearch!="")
            {
                MySqlDataAdapter loadProduct = new MySqlDataAdapter(stringSearch,
connection);
                DataTable loadProductTB = new DataTable();
                loadProduct.Fill(loadProductTB);
                foreach(DataRow row in loadProductTB.Rows)
                {
                    DataRow toRow = productUSelectTB.NewRow();
                    toRow["Barcode"] = row["Barcode"];
                    toRow["Brand"] = row["Brand"];
                    toRow["Model"] = row["Model"];
                    toRow["Size"] = row["Size"];
                    toRow["Status"] = row["Product_Status"];
                    if(productSelectBS.Find("Barcode",toRow["Barcode"])==-1)
                    {
                        productUSelectTB.Rows.Add(toRow);
                    }

                }
                loadProductTB.Clear();
            }
        }

        #endregion

        private void findBtn_Click(object sender, EventArgs e)
        {
            searching();
        }

        private void productAvailbleDGV_CellFormatting(object sender,
DataGridViewCellFormattingEventArgs e)
        {
            if((e.RowIndex!=-
1)&&(e.ColumnIndex==this.productAvailableDGV.Columns["Status"].Index))
            {
                if(e.Value!=null)
                {
                string status = e.Value.ToString();
                switch(status)
                {
                    case "G":

this.productAvailableDGV.Rows[e.RowIndex].DefaultCellStyle.BackColor = Color.Lime;
                        break;
                    case "Y":

this.productAvailableDGV.Rows[e.RowIndex].DefaultCellStyle.BackColor = Color.Gold;
                        break;
                    case "R":

this.productAvailableDGV.Rows[e.RowIndex].DefaultCellStyle.BackColor = Color.Red;
                        break;
                }
                }
            }
        }

        private void productAvailbleDGV_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
            if(e.RowIndex>=0)
            {
                isSelect = true;
                DataGridViewRow rowSelect = productAvailableDGV.Rows[e.RowIndex];
                selectedProduct[0] = rowSelect.Cells["Barcode"].Value.ToString();
                selectedProduct[1] = rowSelect.Cells["Brand"].Value.ToString();
                selectedProduct[2] = rowSelect.Cells["Model"].Value.ToString();
                selectedProduct[3] = rowSelect.Cells["Size"].Value.ToString();
                selectedProduct[4] = rowSelect.Cells["Status"].Value.ToString();
            }
        }

        private void selectBtn_Click(object sender, EventArgs e)
        {
```

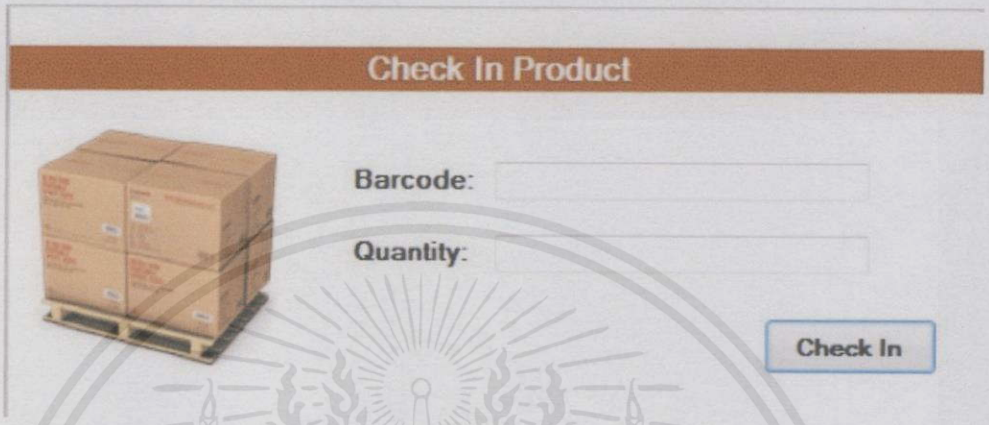
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            if (isSelect == true)
            {
                int findProdIndx =  productAvailableBS.Find("Barcode",
selectedProduct[0]);
                if (findProdIndx != -1)
                {
                    productUSelectTB.Rows.RemoveAt(findProdIndx);
                    DataRow addRow = productSelectTB.NewRow();
                    addRow["Barcode"] = selectedProduct[0];
                    addRow["Brand"] = selectedProduct[1];
                    addRow["Model"] = selectedProduct[2];
                    addRow["Size"] = selectedProduct[3];
                    addRow["Status"] = selectedProduct[4];
                    productSelectTB.Rows.Add(addRow);
                }
            }
        }

        private void productSelectDGV_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
            if (e.RowIndex >= 0)
            {
                isSelect = false;
                DataGridViewRow rowSelect = productSelectDGV.Rows[e.RowIndex];
                selectedProduct[0] =
rowSelect.Cells["BarcodeSelect"].Value.ToString();
                selectedProduct[1] = rowSelect.Cells["BrandSelect"].Value.ToString();
                selectedProduct[2] = rowSelect.Cells["ModelSelect"].Value.ToString();
                selectedProduct[3] = rowSelect.Cells["SizeSelect"].Value.ToString();
                selectedProduct[4] = rowSelect.Cells["StatusSelect"].Value.ToString();
            }
        }

        private void unSelectBtn_Click(object sender, EventArgs e)
        {
            if(isSelect==false)
            {
                int findProdIndx = productSelectBS.Find("Barcode",
selectedProduct[0]);
                if (findProdIndx != -1)
                {
                    productSelectTB.Rows.RemoveAt(findProdIndx);
                    searching();
                }
            }
        }

        private void confirmOrderBtn_Click(object sender, EventArgs e)
        {
            string[] Barcode = productSelectTB.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
            int barcodeLength = Barcode.Length;
            if(barcodeLength>0)
            {
                DialogResult confirmOrderResult = MessageBox.Show("Are you show to
order in this below list, and all product will come to you store in " +
DateTime.Now.Date.AddDays(1).ToShortDateString(),"Dialy Order
Confirm",MessageBoxButtons.YesNo,MessageBoxIcon.Question);
                if(confirmOrderResult==DialogResult.Yes)
                {
                    connection.Open();
                    for(int i=0;i<=barcodeLength-1;i++)
                    {
                        SQLCommand = new MySqlCommand("insert into
dialy_order(Barcode) values('" + Barcode[i] + "');", connection);
                        SQLCommand.ExecuteNonQuery();
                    }
                    connection.Close();
                    MessageBox.Show("Your dialy order was add to database successful",
"Dialy Order", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    productSelectTB.Clear();
                    productUSelectTB.Clear();
                }
            }
            else
            {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                MessageBox.Show("Please select product for add to dialy order", "Dialy
Order", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
        }
    }
}
```

CheckInProduct.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using MySql.Data.MySqlClient;

namespace DataAccess
{
    public partial class CheckInProduct : Form
    {
        DataAccessSetting dataSetting = new DataAccessSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        public string[] barcodeAvailable;
        public CheckInProduct()
        {
            InitializeComponent();
            logoPcb.Image = Properties.Resources.productStock;
            barcodeTxt.Text = null;
            quantityTxt.Text = null;
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.mysqlHost + ";"
+ "DATABASE=" + dataSetting.mysqlCentralDBName + ";" + "UID=" +
dataSetting.mysqlUsername + ";" + "PASSWORD=" + dataSetting.mysqlPassword; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            MySqlDataAdapter barcodeInBranch = new MySqlDataAdapter("select distinct
Barcode from product;", connection);
            DataTable barcodeTB = new DataTable();
            barcodeInBranch.Fill(barcodeTB);
            barcodeAvailable = barcodeTB.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
        }

        private void checkInBtn_Click(object sender, EventArgs e)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {
            bool isExistProduct = false;
            isExistProduct = barcodeAvailable.Contains(barcodeTxt.Text);
            if(isExistProduct==true)
            {
                SQLCommand = new MySqlCommand("update stock set Quantity=" +
quantityTxt.Text + " where barcode='" + barcodeTxt.Text + "';", connection);
                connection.Open();
                SQLCommand.ExecuteNonQuery();
                connection.Close();
                MessageBox.Show("Your stock is update on product where\nBarcode: " +
barcodeTxt.Text + "\nQuantity: " + quantityTxt.Text, "Check In Product",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else
            {
                MessageBox.Show("Your stock is fail to update. Because this barcode
does not available in your store, Please check you barcode in \"Barcode Textbox\" and
try again later.","Check In Product" , MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }
}
```

ภาคผนวก ค: Cashier Application

BranchAPP\Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Data.SqlClient;
using System.Xml;

namespace BranchAPP
{
    public partial class cashierFrm : Form
    {
        GlobalVariables globalVar = new GlobalVariables();
        BranchAppSetting branchSetting = new BranchAppSetting();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        string CFGPath =
(Environment.GetEnvironmentVariable("KMITL_SHOP",EnvironmentVariableTarget.Machine)) +
"\\Cashier_Config.xml";
        public cashierFrm()
        {
            InitializeComponent();
            clearControl();
            loadCFG();
            logoPcb.Image = Properties.Resources.Logo;
            globalVar.mysqlConnectionString = "SERVER=" + branchSetting.dbHost + ";" +
"DATABASE=" + branchSetting.dbName + ";" + "UID=" + branchSetting.dbUser + ";" +
"PASSWORD=" + branchSetting.dbPass;
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
        }


        #region Read Config File

        public void loadCFG()
        {
            XmlReader readTxt = XmlReader.Create(CFGPath);
            while (readTxt.Read())
            {
                if (readTxt.IsStartElement())
                {
                    switch (readTxt.Name)
                    {
                        case "Attribute":
                            string attribute = readTxt["Name"];
                            if (attribute != null)
                            {
                                readTxt.Read();
                                string context = readTxt.Value.Trim();
                                switch (attribute)
                                {
                                    case "cfgUser":
                                        branchSetting.cfgUser = context;
                                        branchSetting.Save();
                                        break;
                                    case "cfgPass":
                                        branchSetting.cfgPass = context;
                                        branchSetting.Save();
                                        break;
                                    case "dbHost":
                                        branchSetting.dbHost = context;
                                        branchSetting.Save();
                                        break;
                                    case "dbName":
                                        branchSetting.dbName = context;
                                        branchSetting.Save();
                                        break;
                                    case "dbUser":
                                        branchSetting.dbUser = context;
                                        branchSetting.Save();
                                        break;
                                    case "dbPass":
                                        branchSetting.dbPass = context;
                                        branchSetting.Save();
                                        break;
                                }
                            }
                            break;
                    }
                }
            }
        }

        #endregion
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        private void loadOrderBtn_Click(object sender, EventArgs e)
        {
            orderIDTxt.ReadOnly = true;
            robotIDTxt.ReadOnly = true;
            clearOrderVariables();
            loadOrder(orderIDTxt.Text, robotIDTxt.Text);
            showOrderList();
            if(globalVar.barcode.Count()==0)
            {
                MessageBox.Show("Your order doesn't exist in database, Please recheck
your OrderID and RobotID and try again later", "System Message", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                robotIDTxt.Text = null;
                robotIDTxt.ReadOnly = false;
                orderIDTxt.Text = null;
                orderIDTxt.ReadOnly = false;
                clearControl();
            }

        }

        #region returnQuantity

        public void returnQuantity(string Barcode, int Quantity)
        {

            connection.Open();
            SQLCommand = new MySqlCommand();
            SQLCommand.Connection = connection;
            SQLCommand.CommandText = "returnQuantity";
            SQLCommand.CommandType = CommandType.StoredProcedure;
            SQLCommand.Parameters.Add("?Barcode_Input", Barcode);
            SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.Parameters.Add("?Quantity_Input", Quantity);
            SQLCommand.Parameters["?Quantity_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.ExecuteNonQuery();
            connection.Close();

        }

        #endregion


        #region Clear Control Function

        public void clearControl()
        {
            for(int i=1;i<=18;i++)
            {
                Control[] barCtrl = this.Controls.Find("barcodeLbl" + i, true);
                if(barCtrl!=null)
                {
                    barCtrl[0].Visible = false;
                }
                Control[] brandCtrl = this.Controls.Find("brandLbl" + i, true);
                if (brandCtrl != null)
                {
                    brandCtrl[0].Visible = false;
                }
                Control[] modelCtrl = this.Controls.Find("modelLbl" + i, true);
                if (modelCtrl != null)
                {
                    modelCtrl[0].Visible = false;
                }
                Control[] sizeCtrl = this.Controls.Find("sizeLbl" + i, true);
                if (sizeCtrl != null)
                {
                    sizeCtrl[0].Visible = false;
                }
                Control[] plusBtnCtrl = this.Controls.Find("plusBtn" + i, true);
                if (plusBtnCtrl != null)
                {
                    plusBtnCtrl[0].Visible = false;
                }
                Control[] quantityCtrl = this.Controls.Find("quantityLbl" + i, true);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                if (quantityCtrl != null)
                {
                    quantityCtrl[0].Visible = false;
                }
                Control[] minusBtnCtrl = this.Controls.Find("minusBtn" + i, true);
                if (minusBtnCtrl != null)
                {
                    minusBtnCtrl[0].Visible = false;
                }
                Control[] pricePUCtrl = this.Controls.Find("pricePULbl" + i, true);
                if (pricePUCtrl != null)
                {
                    pricePUCtrl[0].Visible = false;
                }
                Control[] priceCtrl = this.Controls.Find("priceLbl" + i, true);
                if (priceCtrl != null)
                {
                    priceCtrl[0].Visible = false;
                }
                Control[] cancleBtnCtrl = this.Controls.Find("cancleBtn" +
Convert.ToString(i), true);
                if (cancleBtnCtrl != null)
                {
                    cancleBtnCtrl[0].Visible = false;
                }
                productScrollbar1.Visible = false;
                purchaseBtn.Visible = false;
                deleteOrderBtn.Visible = false;
                totalpriceHeadLbl.Visible = false;
                totalpriceLbl.Visible = false;
                totalpriceUnitLbl.Visible = false;
                orderIDTxt.ReadOnly = false;
                orderIDTxt.Text = null;
                robotIDTxt.ReadOnly = false;
                robotIDTxt.Text = null;

            }

        }

        #endregion

        #region Clear Order Variables

        public void clearOrderVariables()
        {
            globalVar.barcode = null;
            globalVar.brand = null;
            globalVar.model = null;
            globalVar.size = null;
            globalVar.quantity = null;
            globalVar.pricePU = null;
        }

        #endregion

        #region Load Order From Database

        public void loadOrder(string orderID,string RobotID)
        {
            string loadOrder = "select
order_pending.barcode,central_product.brand,central_product.model,central_product.size
,order_pending.quantity,central_product.pricePerContain from order_pending left join
central_product on order_pending.barcode=central_product.barcode and
order_pending.robotID='" + RobotID + "' and order_pending.orderid=" + orderID + "
where brand is not null;";
            MySqlDataAdapter loadOrder2App = new MySqlDataAdapter(loadOrder,
connection);
            DataTable orderDT = new DataTable();
            loadOrder2App.Fill(orderDT);
            globalVar.barcode = orderDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
            globalVar.brand = orderDT.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
            globalVar.model = orderDT.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            globalVar.size = orderDT.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
            globalVar.quantity = orderDT.AsEnumerable().Select(r =>
r.Field<int>("Quantity")).ToArray();
            globalVar.pricePU = orderDT.AsEnumerable().Select(r =>
r.Field<float>("PricePerContain")).ToArray();
            int listQuantity = globalVar.barcode.Count();
            globalVar.totalPrice = new float[listQuantity];
            for (int i = 1; i <= listQuantity; i++)
            {
                globalVar.totalPrice[i - 1] = globalVar.pricePU[i - 1] *
globalVar.quantity[i - 1];
            }
        }

        #endregion

        #region Show Order List Function
        public void showOrderList()
        {
            int amountOfOrder = globalVar.barcode.Count();
            if (amountOfOrder > 0)
            {
                if (amountOfOrder <= 18)
                {
                    for (int i = 1; i <= amountOfOrder; i++)
                    {
                        string barcodeStr = "barcodeLbl" + Convert.ToString(i);
                        Control[] barcodeCtrl = this.Controls.Find(barcodeStr, true);
                        if (barcodeCtrl != null)
                        {
                            barcodeCtrl[0].Text =
globalVar.barcode[(productScrollbar1.Value + i) - 1];
                            barcodeCtrl[0].Visible = true;
                        }
                        string brandStr = "brandLbl" + Convert.ToString(i);
                        Control[] brandLblCtrl = this.Controls.Find(brandStr, true);
                        if (brandLblCtrl != null)
                        {
                            brandLblCtrl[0].Text = globalVar.brand[i - 1];
                            brandLblCtrl[0].Visible = true;
                        }
                        string modelStr = "modelLbl" + Convert.ToString(i);
                        Control[] modelLblCtrl = this.Controls.Find(modelStr, true);
                        if (modelLblCtrl != null)
                        {
                            modelLblCtrl[0].Text = globalVar.model[i - 1];
                            modelLblCtrl[0].Visible = true;
                        }
                        string sizeStr = "sizeLbl" + Convert.ToString(i);
                        Control[] sizeLblCtrl = this.Controls.Find(sizeStr, true);
                        if (sizeLblCtrl != null)
                        {
                            sizeLblCtrl[0].Text = globalVar.size[i - 1];
                            sizeLblCtrl[0].Visible = true;
                        }
                        string plusStr = "plusBtn" + Convert.ToString(i);
                        Control[] plusBtnCtrl = this.Controls.Find(plusStr, true);
                        if (plusBtnCtrl != null)
                        {
                            plusBtnCtrl[0].Visible = true;
                        }
                        string quantityStr = "quantityLBl" + Convert.ToString(i);
                        Control[] quantityCtrl = this.Controls.Find(quantityStr,
true);
                        if (quantityCtrl != null)
                        {
                            quantityCtrl[0].Text =
Convert.ToString(globalVar.quantity[i - 1]);
                            quantityCtrl[0].Visible = true;
                        }
                        string minusStr = "minusBtn" + Convert.ToString(i);
                        Control[] minusBtnCtrl = this.Controls.Find(minusStr, true);
                        if (minusBtnCtrl != null)
                        {
                            minusBtnCtrl[0].Visible = true;
                        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ทำซ้ำ ดัดแปลง เผยแพร่ หรือนำไปใช้ในรูปแบบอื่นๆ ที่มีการนำไปใช้

```
                        string pricePUStr = "pricePULbl" + Convert.ToString(i);
                        Control[] pricePULblCtrl = this.Controls.Find(pricePUStr,
true);
                        if (pricePULblCtrl != null)
                        {
                            pricePULblCtrl[0].Text =
Convert.ToString(globalVar.pricePU[i - 1]);
                            pricePULblCtrl[0].Visible = true;
                        }
                        string priceStr = "priceLbl" + Convert.ToString(i);
                        Control[] priceLblCtrl = this.Controls.Find(priceStr, true);
                        if (priceLblCtrl != null)
                        {
                            priceLblCtrl[0].Text =
Convert.ToString(globalVar.totalPrice[i - 1]);
                            priceLblCtrl[0].Visible = true;
                        }
                        string cancleStr = "cancleBtn" + Convert.ToString(i);
                        Control[] cancleBtnCtrl = this.Controls.Find(cancleStr, true);
                        if (cancleBtnCtrl != null)
                        {
                            cancleBtnCtrl[0].Visible = true;
                        }
                    }
                }
                else
                {
                    productScrollbar1.Visible = true;
                    productScrollbar1.Maximum = amountOfOrder - 18;
                    productScrollbar1.Minimum = 0;
                    for (int i = 1; i <= 18; i++)
                    {
                        string barcodeStr = "barcodeLbl" + Convert.ToString(i);
                        Control[] barcodeCtrl = this.Controls.Find(barcodeStr, true);
                        if (barcodeCtrl != null)
                        {
                            barcodeCtrl[0].Text =
globalVar.barcode[(productScrollbar1.Value + i) - 1];
                            barcodeCtrl[0].Visible = true;
                        }
                        string brandStr = "brandLbl" + Convert.ToString(i);
                        Control[] brandLblCtrl = this.Controls.Find(brandStr, true);
                        if (brandLblCtrl != null)
                        {
                            brandLblCtrl[0].Text =
globalVar.brand[(productScrollbar1.Value + i) - 1];
                            brandLblCtrl[0].Visible = true;
                        }
                        string modelStr = "modelLbl" + Convert.ToString(i);
                        Control[] modelLblCtrl = this.Controls.Find(modelStr, true);
                        if (modelLblCtrl != null)
                        {
                            modelLblCtrl[0].Text =
globalVar.model[(productScrollbar1.Value + i) - 1];
                            modelLblCtrl[0].Visible = true;
                        }
                        string sizeStr = "sizeLbl" + Convert.ToString(i);
                        Control[] sizeLblCtrl = this.Controls.Find(sizeStr, true);
                        if (sizeLblCtrl != null)
                        {
                            sizeLblCtrl[0].Text =
globalVar.size[(productScrollbar1.Value + i) - 1];
                            sizeLblCtrl[0].Visible = true;
                        }
                        string plusStr = "plusBtn" + Convert.ToString(i);
                        Control[] plusBtnCtrl = this.Controls.Find(plusStr, true);
                        if (plusBtnCtrl != null)
                        {
                            plusBtnCtrl[0].Visible = true;
                        }
                        string quantityStr = "quantityLBl" + Convert.ToString(i);
                        Control[] quantityCtrl = this.Controls.Find(quantityStr,
true);
                        if (quantityCtrl != null)
                        {
                            quantityCtrl[0].Text =
Convert.ToString(globalVar.quantity[(productScrollbar1.Value + i) - 1]);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        quantityCtrl[0].Visible = true;
                    }
                    string minusStr = "minusBtn" + Convert.ToString(i);
                    Control[] minusBtnCtrl = this.Controls.Find(minusStr, true);
                    if (minusBtnCtrl != null)
                    {
                        minusBtnCtrl[0].Visible = true;
                    }
                    string pricePUStr = "pricePULbl" + Convert.ToString(i);
                    Control[] pricePULblCtrl = this.Controls.Find(pricePUStr,
true);
                    if (pricePULblCtrl != null)
                    {
                        pricePULblCtrl[0].Text =
Convert.ToString(globalVar.pricePU[(productScrollbar1.Value + i) - 1]);
                        pricePULblCtrl[0].Visible = true;
                    }
                    string priceStr = "priceLbl" + Convert.ToString(i);
                    Control[] priceLblCtrl = this.Controls.Find(priceStr, true);
                    if (priceLblCtrl != null)
                    {
                        priceLblCtrl[0].Text =
Convert.ToString(globalVar.totalPrice[(productScrollbar1.Value + i) - 1]);
                        priceLblCtrl[0].Visible = true;
                    }
                    string cancleStr = "cancleBtn" + Convert.ToString(i);
                    Control[] cancleBtnCtrl = this.Controls.Find(cancleStr, true);
                    if (cancleBtnCtrl != null)
                    {
                        cancleBtnCtrl[0].Visible = true;
                    }
                }
            }
            totalpriceHeadLbl.Visible = true;
            totalpriceLbl.Visible = true;
            totalpriceUnitLbl.Visible = true;
            purchaseBtn.Visible = true;
            deleteOrderBtn.Visible = true;
            totalpriceLbl.Text = Convert.ToString(globalVar.totalPrice.Sum());
        }
        else
        {
            totalpriceHeadLbl.Visible = false;
            totalpriceLbl.Visible = false;
            totalpriceUnitLbl.Visible = false;
            purchaseBtn.Visible = false;
            deleteOrderBtn.Visible = false;
        }
    }

    #endregion


    #region CheckQuantityAvilable Function

    public bool checkQuantityAvilable(string Barcode, bool isPlus)
    {
        connection.Open();
        SQLCommand = new MySqlCommand();
        SQLCommand.Connection = connection;
        SQLCommand.CommandText = "orderQuantityChange";
        SQLCommand.CommandType = CommandType.StoredProcedure;
        SQLCommand.Parameters.Add("?Barcode_Input", Barcode);
        SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;

        SQLCommand.Parameters.Add("?isPlus", isPlus);
        SQLCommand.Parameters["?isPlus"].Direction = ParameterDirection.Input;
        SQLCommand.Parameters.AddWithValue("?isDone", MySqlDbType.Enum);
        SQLCommand.Parameters["?isDone"].Direction = ParameterDirection.Output;
        SQLCommand.ExecuteNonQuery();
        connection.Close();
        return Convert.ToBoolean(SQLCommand.Parameters["?isDone"].Value);
    }

    #endregion
```

เอกสารนี้เป็นเอกสาร... ไม่ว่ากรณีใดๆทั้งสิ้น ...

```
        #region Product Amount Change
        public void amountChange(bool isIncrease, int controlIndx)
        {
            string amountLblStr = "quantityLbl" + Convert.ToString(controlIndx);
            Control[] amountLbl = this.Controls.Find(amountLblStr, true);
            int amount = 0;
            if (amountLbl != null)
            {
                amount = Convert.ToInt16(amountLbl[0].Text);
                if (isIncrease == true)
                {
                    bool result =
checkQuantityAvilable(globalVar.barcode[(productScrollbar1.Value + controlIndx) - 1],
true);
                    if (result == true)
                    {
                        amount = amount + 1; ;
                    }
                    else
                    {
                        Control[] plusBtnHind = this.Controls.Find("plusBtn" +
controlIndx, true);
                        if (plusBtnHind != null)
                        {
                            plusBtnHind[0].Visible = false;
                        }
                        MessageBox.Show("This product is avaliable on " + amount + "
items", "System Message", MessageBoxButtons.OK, MessageBoxIcon.Hand);
                    }
                    globalVar.quantity[(productScrollbar1.Value + controlIndx) - 1] =
amount;
                    amountLbl[0].Text = Convert.ToString(amount);
                    Control[] priceLbl = this.Controls.Find("priceLbl" +
Convert.ToString(controlIndx), true);
                    if (priceLbl != null)
                    {
                        priceLbl[0].Text = null;
                        Control[] pricePU = this.Controls.Find("pricePULbl" +
Convert.ToString(controlIndx), true);
                        if (pricePU != null)
                        {
                            globalVar.totalPrice[(productScrollbar1.Value +
controlIndx) - 1] = (globalVar.quantity[(productScrollbar1.Value + controlIndx) - 1])
* (globalVar.pricePU[(productScrollbar1.Value + controlIndx) - 1]);
                            priceLbl[0].Text =
Convert.ToString(globalVar.totalPrice[(productScrollbar1.Value + controlIndx) - 1]);
                        }
                    }
                }
                else
                {
                    if (amount > 0)
                    {
                        bool result =
checkQuantityAvilable(globalVar.barcode[(productScrollbar1.Value + controlIndx) - 1],
false);
                        if (result == true)
                        {
                            amount = amount - 1;
                        }
                        Control[] plusBtnHind = this.Controls.Find("plusBtn" +
controlIndx, true);
                        if (plusBtnHind != null)
                        {
                            if (plusBtnHind[0].Visible == false)
                            {
                                plusBtnHind[0].Visible = true;
                            }
                        }
                        globalVar.quantity[(productScrollbar1.Value + controlIndx) -
1] = amount;
                        amountLbl[0].Text = Convert.ToString(amount);
                        Control[] priceLbl = this.Controls.Find("priceLbl" +
Convert.ToString(controlIndx), true);
                        if (priceLbl != null)
                        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                            priceLbl[0].Text = null;
                            Control[] pricePU = this.Controls.Find("pricePULbl" +
Convert.ToString(controlIndx), true);
                            if (pricePU != null)
                            {
                                globalVar.totalPrice[(productScrollbar1.Value +
controlIndx) - 1] = (globalVar.quantity[(productScrollbar1.Value + controlIndx) - 1])
* (globalVar.pricePU[(productScrollbar1.Value + controlIndx) - 1]);
                                priceLbl[0].Text =
Convert.ToString(globalVar.totalPrice[(productScrollbar1.Value + controlIndx) - 1]);
                            }
                        }
                    }
                }
            }
            totalpriceLbl.Text = Convert.ToString(globalVar.totalPrice.Sum());

        }

        #endregion

        #region Delete Order Function
        public void deleteOrder(int controlIndx)
        {
            List<string> barcode = new List<string>(globalVar.barcode);
            List<string> brand = new List<string>(globalVar.brand);
            List<string> model = new List<string>(globalVar.model);
            List<string> size = new List<string>(globalVar.size);
            List<int> quantity = new List<int>(globalVar.quantity);
            List<float> pricePU = new List<float>(globalVar.pricePU);
            List<float> price = new List<float>(globalVar.totalPrice);


            SQLCommand = new MySqlCommand("delete from order_pending where Barcode='"
+ globalVar.barcode[(controlIndx + productScrollbar1.Value) - 1] + "' and orderID = "
+ orderIDTxt.Text + " and robotID ='" + robotIDTxt.Text +"';",connection);
            connection.Open();
            SQLCommand.ExecuteNonQuery();
            connection.Close();
            returnQuantity(globalVar.barcode[(controlIndx + productScrollbar1.Value) -
1], globalVar.quantity[(controlIndx + productScrollbar1.Value) - 1]);


            barcode.RemoveAt((controlIndx + productScrollbar1.Value) - 1);
            brand.RemoveAt((controlIndx + productScrollbar1.Value) - 1);
            model.RemoveAt((controlIndx + productScrollbar1.Value) - 1);
            size.RemoveAt((controlIndx + productScrollbar1.Value) - 1);
            quantity.RemoveAt((controlIndx + productScrollbar1.Value) - 1);
            pricePU.RemoveAt((controlIndx + productScrollbar1.Value) - 1);
            price.RemoveAt((controlIndx + productScrollbar1.Value) - 1);

            globalVar.barcode = barcode.ToArray();
            globalVar.brand = brand.ToArray();
            globalVar.model = model.ToArray();
            globalVar.size = size.ToArray();
            globalVar.quantity = quantity.ToArray();
            globalVar.pricePU = pricePU.ToArray();
            globalVar.totalPrice = price.ToArray();

            string robotName = robotIDTxt.Text;
            string orderName = orderIDTxt.Text;
            clearControl();
            showOrderList();
            robotIDTxt.Text = robotName;
            orderIDTxt.Text = orderName;
        }

        #endregion

        #region Cancle Botton NO.1 - 18

        private void cancleBtn1_Click(object sender, EventArgs e)
        {
            deleteOrder(1);
        }

        private void cancleBtn2_Click(object sender, EventArgs e)
```

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
    deleteOrder(2);
}

private void cancleBtn3_Click(object sender, EventArgs e)
{
    deleteOrder(3);
}

private void cancleBtn4_Click(object sender, EventArgs e)
{
    deleteOrder(4);
}

private void cancleBtn5_Click(object sender, EventArgs e)
{
    deleteOrder(5);
}

private void cancleBtn6_Click(object sender, EventArgs e)
{
    deleteOrder(6);
}

private void cancleBtn7_Click(object sender, EventArgs e)
{
    deleteOrder(7);
}

private void cancleBtn8_Click(object sender, EventArgs e)
{
    deleteOrder(8);
}

private void cancleBtn9_Click(object sender, EventArgs e)
{
    deleteOrder(9);
}

private void cancleBtn10_Click(object sender, EventArgs e)
{
    deleteOrder(10);
}

private void cancleBtn11_Click(object sender, EventArgs e)
{
    deleteOrder(11);
}

private void cancleBtn12_Click(object sender, EventArgs e)
{
    deleteOrder(12);
}

private void cancleBtn13_Click(object sender, EventArgs e)
{
    deleteOrder(13);
}

private void cancleBtn14_Click(object sender, EventArgs e)
{
    deleteOrder(14);
}

private void cancleBtn15_Click(object sender, EventArgs e)
{
    deleteOrder(15);
}

private void cancleBtn16_Click(object sender, EventArgs e)
{
    deleteOrder(16);
}

private void cancleBtn17_Click(object sender, EventArgs e)
{
    deleteOrder(17);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}

private void cancleBtn18_Click(object sender, EventArgs e)
{
    deleteOrder(18);
}

#endregion

#region Plus Botton NO.1 - 18

private void plusBtn1_Click(object sender, EventArgs e)
{
    amountChange(true,1);
}

private void plusBtn2_Click(object sender, EventArgs e)
{
    amountChange(true, 2);
}

private void plusBtn3_Click(object sender, EventArgs e)
{
    amountChange(true, 3);
}

private void plusBtn4_Click(object sender, EventArgs e)
{
    amountChange(true, 4);
}

private void plusBtn5_Click(object sender, EventArgs e)
{
    amountChange(true, 5);
}

private void plusBtn6_Click(object sender, EventArgs e)
{
    amountChange(true, 6);
}

private void plusBtn7_Click(object sender, EventArgs e)
{
    amountChange(true, 7);
}

private void plusBtn8_Click(object sender, EventArgs e)
{
    amountChange(true, 8);
}

private void plusBtn9_Click(object sender, EventArgs e)
{
    amountChange(true, 9);
}

private void plusBtn10_Click(object sender, EventArgs e)
{
    amountChange(true, 10);
}

private void plusBtn11_Click(object sender, EventArgs e)
{
    amountChange(true, 11);
}

private void plusBtn12_Click(object sender, EventArgs e)
{
    amountChange(true, 12);
}

private void plusBtn13_Click(object sender, EventArgs e)
{
    amountChange(true, 13);
}

private void plusBtn14_Click(object sender, EventArgs e)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
    amountChange(true, 14);
}

private void plusBtn15_Click(object sender, EventArgs e)
{
    amountChange(true, 15);
}

private void plusBtn16_Click(object sender, EventArgs e)
{
    amountChange(true, 16);
}

private void plusBtn17_Click(object sender, EventArgs e)
{
    amountChange(true, 17);
}

private void plusBtn18_Click(object sender, EventArgs e)
{
    amountChange(true, 18);
}

#endregion

#region Minus Botton NO.1 - 18

private void minusBtn1_Click(object sender, EventArgs e)
{
    amountChange(false, 1);
}

private void minusBtn2_Click(object sender, EventArgs e)
{
    amountChange(false, 2);
}

private void minusBtn3_Click(object sender, EventArgs e)
{
    amountChange(false, 3);
}

private void minusBtn4_Click(object sender, EventArgs e)
{
    amountChange(false, 4);
}

private void minusBtn5_Click(object sender, EventArgs e)
{
    amountChange(false, 5);
}

private void minusBtn6_Click(object sender, EventArgs e)
{
    amountChange(false, 6);
}

private void minusBtn7_Click(object sender, EventArgs e)
{
    amountChange(false, 7);
}

private void minusBtn8_Click(object sender, EventArgs e)
{
    amountChange(false, 8);
}

private void minusBtn9_Click(object sender, EventArgs e)
{
    amountChange(false, 9);
}

private void minusBtn10_Click(object sender, EventArgs e)
{
    amountChange(false, 10);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        private void minusBtn11_Click(object sender, EventArgs e)
        {
            amountChange(false, 11);
        }

        private void minusBtn12_Click(object sender, EventArgs e)
        {
            amountChange(false, 12);
        }

        private void minusBtn13_Click(object sender, EventArgs e)
        {
            amountChange(false, 13);
        }

        private void minusBtn14_Click(object sender, EventArgs e)
        {
            amountChange(false, 14);
        }

        private void minusBtn15_Click(object sender, EventArgs e)
        {
            amountChange(false, 15);
        }

        private void minusBtn16_Click(object sender, EventArgs e)
        {
            amountChange(false, 16);
        }

        private void minusBtn17_Click(object sender, EventArgs e)
        {
            amountChange(false, 17);
        }

        private void minusBtn18_Click(object sender, EventArgs e)
        {
            amountChange(false, 18);
        }

        #endregion

        private void productScrollbar1_Scroll(object sender, ScrollEventArgs e)
        {
            showOrderList();
        }

        private void clearBtn_Click(object sender, EventArgs e)
        {
            robotIDTxt.Text = null;
            robotIDTxt.ReadOnly = false;
            orderIDTxt.Text = null;
            orderIDTxt.ReadOnly = false;
            clearControl();
        }

        private void deleteOrderBtn_Click(object sender, EventArgs e)
        {
            DialogResult confirmResult = MessageBox.Show("Are you sure to delete\n" +
"OrderID: " + orderIDTxt.Text + "\nRobotID: " + robotIDTxt.Text + "\n from database",
"System Message", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
            if(confirmResult==DialogResult.Yes)
            {
                int lengthOfOrder = globalVar.barcode.Count();
                for (int i = 1; i <= lengthOfOrder;i++ )
                {
                    SQLCommand = new MySqlCommand("delete from order_pending where
Barcode='" + globalVar.barcode[i-1] + "' and orderID = " + orderIDTxt.Text + " and
robotID ='" + robotIDTxt.Text +"';",connection);
                    connection.Open();
                    SQLCommand.ExecuteNonQuery();
                    connection.Close();
                    returnQuantity(globalVar.barcode[i - 1], globalVar.quantity[i -
1]);
                }
                clearControl();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                MessageBox.Show("OrderID: " + orderIDTxt.Text + "\nRobotID: " +
robotIDTxt.Text + "\n was deleted from database", "System Message",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                clearControl();
            }


        }

        private void purchaseBtn_Click(object sender, EventArgs e)
        {
            int amountOfOrder = globalVar.barcode.Count();
            connection.Open();
            for (int i = 1; i <= amountOfOrder; i++)
            {
                SQLCommand = new MySqlCommand();
                SQLCommand.Connection = connection;
                SQLCommand.CommandText = "add2SellTB";
                SQLCommand.CommandType = CommandType.StoredProcedure;
                SQLCommand.Parameters.Add("?Barcode_Input", globalVar.barcode[i - 1]);
                SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;

                SQLCommand.Parameters.Add("?Quantity_Input", globalVar.quantity[i -
1]);
                SQLCommand.Parameters["?Quantity_Input"].Direction =
ParameterDirection.Input;
                SQLCommand.ExecuteNonQuery();
            }
            SQLCommand = new MySqlCommand("delete from order_pending where robotID='"
+ robotIDTxt.Text + "' and orderID='" + orderIDTxt.Text + "';", connection);
            SQLCommand.ExecuteNonQuery();
            connection.Close();
            MessageBox.Show("Purchase is done", "System Message",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            clearControl();
        }

    }
}
```

Branch_Config\Form1.cs

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Xml;
using System.IO;

namespace Branch_Config
{
    public partial class branchCfgFrm : Form
    {
        BranchSetting branchConfigSetting = new BranchSetting();
        string CFGPath = (Environment.GetEnvironmentVariable("KMITL_SHOP",
EnvironmentVariableTarget.Machine)) + "\\Cashier_Config.xml";
        public branchCfgFrm()
        {
            InitializeComponent();
            initialFrm();
            readWriteCFGFile("Read", null, null, null);
        }
        public void readWriteCFGFile(string Mode, string Element, string Attribute,
string value)
        {
            if (Mode == "Read")
            {
                XmlReader readTxt = XmlReader.Create(CFGPath);
                while (readTxt.Read())
                {
                    if (readTxt.IsStartElement())
                    {
                        switch (readTxt.Name)
                        {
                            case "Attribute":
                                string attribute = readTxt["Name"];
                                if (attribute != null)
                                {
                                    readTxt.Read();
                                    string context = readTxt.Value.Trim();
                                    switch (attribute)
                                    {
                                        case "cfgUser":
                                            branchConfigSetting.cfgUser = context;
                                            branchConfigSetting.Save();
                                            break;
                                        case "cfgPass":
                                            branchConfigSetting.cfgPass = context;
                                            branchConfigSetting.Save();
                                            break;
                                        case "dbHost":
                                            branchConfigSetting.dbHost = context;
                                            branchConfigSetting.Save();
                                            break;
                                        case "dbName":
                                            branchConfigSetting.dbName = context;
                                            branchConfigSetting.Save();
                                            break;
                                        case "dbUser":
                                            branchConfigSetting.dbUser = context;
                                            branchConfigSetting.Save();
                                            break;
                                        case "dbPass":
                                            branchConfigSetting.dbPass = context;
                                            branchConfigSetting.Save();
                                            break;
                                    }
                                }
                                break;
                        }
                    }
                }
                readTxt.Close();
            }
            else if (Mode == "Write")
            {
                XmlDocument xmlChange = new XmlDocument();
                xmlChange.Load(CFGPath);
                XmlNode nodeChange = xmlChange.SelectSingleNode("//Attribute[@Name='"
+ Attribute + "']");
                nodeChange.InnerText = value;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ลบข้อความนี้ออกจากเอกสารทุกครั้งที่มีการนำไปใช้

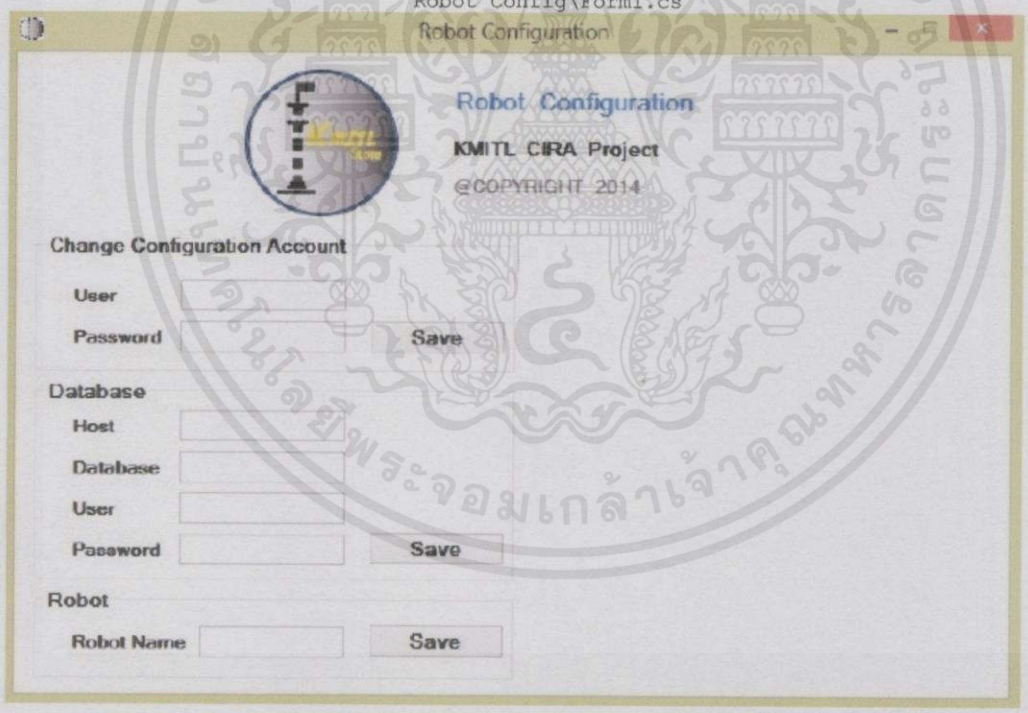
```
                xmlChange.Save(CFGPath);

            }
        }
        public void initialFrm()
        {
            configLogoPcb.Image = Properties.Resources.Logo;
            configLogoPcb.Left = 12;
            configHead1Lbl.Left = 145;
            configHead2Lbl.Left = 145;
            configHead3Lbl.Left = 145;
            configAccountGB.Visible = false;
            configAccDspLbl.Visible = false;
            dbGB.Visible = false;
            dbDspLbl.Visible = false;
        }

        private void loginBtn_Click(object sender, EventArgs e)
        {
            if ((userTxt.Text == branchConfigSetting.cfgUser) && (passwordTxt.Text ==
branchConfigSetting.cfgPass))
            {
                MessageBox.Show("Login is successful", "Login Status",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                loginGB.Visible = false;
                configLogoPcb.Left = 150;
                configHead1Lbl.Left = 303;
                configHead2Lbl.Left = 303;
                configHead3Lbl.Left = 303;
                configAccountGB.Visible = true;
                configAccDspLbl.Visible = true;
                configAccDspLbl.Text = null;
                dbGB.Visible = true;
                dbDspLbl.Visible = true;
                dbDspLbl.Text = null;
            }
            else
            {
                var result = MessageBox.Show("Fail to login: Username or Password
doesn't matching", "Login Status", MessageBoxButtons.RetryCancel,
MessageBoxIcon.Hand);
                if (result == DialogResult.Cancel)
                {
                    this.Close();
                }
                else if (result == DialogResult.Retry)
                {
                    userTxt.Text = null;
                    passwordTxt.Text = null;
                }
            }
        }

        private void saveAccCfgBtn_Click(object sender, EventArgs e)
        {
            if ((cfgUserTxt.Text != "") && (cfgUserTxt.Text !=
branchConfigSetting.cfgUser))
            {
                if ((cfgPassTxt.Text != "") && (cfgPassTxt.Text !=
branchConfigSetting.cfgPass))
                {
                    branchConfigSetting.cfgUser = cfgUserTxt.Text;
                    branchConfigSetting.cfgPass = cfgPassTxt.Text;
                    branchConfigSetting.Save();
                    readWriteCFGFile("Write", "CONFIG_LOGIN", "cfgUser",
cfgUserTxt.Text);
                    readWriteCFGFile("Write", "CONFIG_LOGIN", "cfgPass",
cfgPassTxt.Text);
                    configAccDspLbl.Text = "New  User: " + branchConfigSetting.cfgUser
+ "\nNew Password: " + branchConfigSetting.cfgPass;
                    MessageBox.Show("Configuration Account is changed successfully",
"Configuration Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
            }
            else
            {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                MessageBox.Show("Please make sure that newUser and newPassword don't
same value and null value", "Configuration Message", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            }
        }

        private void dbSaveBtn_Click(object sender, EventArgs e)
        {
            branchConfigSetting.dbHost = dbHostTxt.Text;
            branchConfigSetting.dbName = dbNameTxt.Text;
            branchConfigSetting.dbUser = dbUserTxt.Text;
            branchConfigSetting.dbPass = dbPassTxt.Text;
            branchConfigSetting.Save();
            readWriteCFGFile("Write", "DATABASE", "dbHost", dbHostTxt.Text);
            readWriteCFGFile("Write", "DATABASE", "dbName", dbNameTxt.Text);
            readWriteCFGFile("Write", "DATABASE", "dbUser", dbUserTxt.Text);
            readWriteCFGFile("Write", "DATABASE", "dbPass", dbPassTxt.Text);
            dbDspLbl.Text = "Host: " + branchConfigSetting.dbHost + "\nDatabase Name:
" + branchConfigSetting.dbName + "\nDatabase User: " + branchConfigSetting.dbUser +
"\nDatabase Password: " + branchConfigSetting.dbPass;
            MessageBox.Show("Database is changed successfully", "Configuration
Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}
```

ภาคผนวก ง: Robot Application

Robot Config\Form1.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.Xml;

namespace Robot_Config
{
```

เอกสารนี้จัดทำขึ้นเพื่อใช้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    public partial class robotCfgFrm : Form
    {
        RobotConfigSetting robotConfigSetting = new RobotConfigSetting();
        public string CFGPath =
(Environment.GetEnvironmentVariable("KMITL_SHOP",EnvironmentVariableTarget.Machine)) +
"\\Robot_Config.xml";
        public robotCfgFrm()
        {
            InitializeComponent();
            initialFrm();

            readWriteCFGFile("Read",null,null,null);
        }
        public void readWriteCFGFile(string Mode,string Element,string
Attribute,string value)
        {

            if(Mode=="Read")
            {
                XmlReader readTxt = XmlReader.Create(CFGPath);
                while (readTxt.Read())
                {
                    if (readTxt.IsStartElement())
                    {
                        switch (readTxt.Name)
                        {
                            case "Attribute":
                                string attribute = readTxt["Name"];
                                if (attribute != null)
                                {
                                    readTxt.Read();
                                    string context = readTxt.Value.Trim();
                                    switch (attribute)
                                    {
                                        case "cfgUser":

                                            robotConfigSetting.ConfigUser = context;
                                            robotConfigSetting.Save();
                                            break;
                                        case "cfgPass":
                                            robotConfigSetting.ConfigPass = context;
                                            robotConfigSetting.Save();
                                            break;
                                        case "robotName":
                                            robotConfigSetting.robotName = context;
                                            robotConfigSetting.Save();
                                            break;
                                        case "dbHost":
                                            robotConfigSetting.dbHost = context;
                                            robotConfigSetting.Save();
                                            break;
                                        case "dbName":
                                            robotConfigSetting.dbName = context;
                                            robotConfigSetting.Save();
                                            break;
                                        case "dbUser":
                                            robotConfigSetting.dbUser = context;
                                            robotConfigSetting.Save();
                                            break;
                                        case "dbPass":
                                            robotConfigSetting.dbPass = context;
                                            robotConfigSetting.Save();
                                            break;
                                    }
                                }
                                break;
                        }
                    }
                }
                readTxt.Close();
            }
            else if(Mode=="Write")
            {
                XmlDocument xmlChange = new XmlDocument();
                xmlChange.Load(CFGPath);
                XmlNode nodeChange = xmlChange.SelectSingleNode("//Attribute[@Name='"
+ Attribute + "']");
```

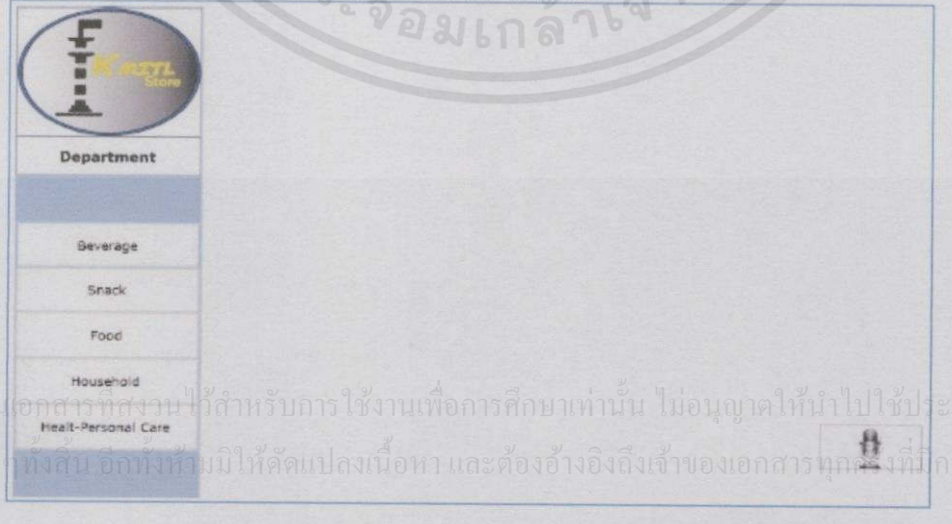
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                nodeChange.InnerText = value;
                xmlChange.Save(CFGPath);

            }
        }

        public void initialFrm()
        {
            configLogoPcb.Image = Properties.Resources.CIRA_Logo;
            configLogoPcb.Left = 12;
            configHead1Lbl.Left = 145;
            configHead2Lbl.Left = 145;
            configHead3Lbl.Left = 145;
            configAccountGB.Visible = false;
            configAccDspLbl.Visible = false;
            dbGB.Visible = false;
            dbDspLbl.Visible = false;
            robotGB.Visible = false;
            robotDspLbl.Visible = false;
        }
        private void loginBtn_Click(object sender, EventArgs e)
        {
            if ((userTxt.Text == robotConfigSetting.ConfigUser) && (passwordTxt.Text
== robotConfigSetting.ConfigPass))
            {
                MessageBox.Show("Login is successful", "Login Status",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                loginGB.Visible = false;
                configLogoPcb.Left = 150;
                configHead1Lbl.Left = 303;
                configHead2Lbl.Left = 303;
                configHead3Lbl.Left = 303;
                configAccountGB.Visible = true;
                configAccDspLbl.Visible = true;
                configAccDspLbl.Text = null;
                dbGB.Visible = true;
                dbDspLbl.Visible = true;
                dbDspLbl.Text = null;
                robotGB.Visible = true;
                robotDspLbl.Visible = true;
                robotDspLbl.Text = null;
            }
            else
            {
                var result = MessageBox.Show("Fail to login: Username or Password
doesn't matching", "Login Status", MessageBoxButtons.RetryCancel,
MessageBoxIcon.Hand);
                if (result == DialogResult.Cancel)
                {
                    this.Close();
                }
                else if (result == DialogResult.Retry)
                {
                    userTxt.Text = null;
                    passwordTxt.Text = null;
                }
            }
        }

        private void saveAccCfgBtn_Click(object sender, EventArgs e)
        {

if((cfgUserTxt.Text!="")&&(cfgUserTxt.Text!=robotConfigSetting.ConfigUser))
            {

if((cfgPassTxt.Text!="")&&(cfgPassTxt.Text!=robotConfigSetting.ConfigPass))
                {
                    robotConfigSetting.ConfigUser = cfgUserTxt.Text;
                    robotConfigSetting.ConfigPass = cfgPassTxt.Text;
                    robotConfigSetting.Save();
                    readWriteCFGFile("Write", "CONFIG_LOGIN", "cfgUser",
cfgUserTxt.Text);
                    readWriteCFGFile("Write", "CONFIG_LOGIN", "cfgPass",
cfgPassTxt.Text);
                    configAccDspLbl.Text = "New  User: " +
robotConfigSetting.ConfigUser + "\nNew Password: " + robotConfigSetting.ConfigPass;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    MessageBox.Show("Configuration Account is changed successfully",
"Configuration Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
            }
            else
            {
                MessageBox.Show("Please make sure that newUser and newPassword don't
same value and null value", "Configuration Message", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            }
        }

        private void dbSaveBtn_Click(object sender, EventArgs e)
        {
            robotConfigSetting.dbHost = dbHostTxt.Text;
            robotConfigSetting.dbName = dbNameTxt.Text;
            robotConfigSetting.dbUser = dbUserTxt.Text;
            robotConfigSetting.dbPass = dbPassTxt.Text;
            robotConfigSetting.Save();
            readWriteCFGFile("Write", "DATABASE", "dbHost", dbHostTxt.Text);
            readWriteCFGFile("Write", "DATABASE", "dbName", dbNameTxt.Text);
            readWriteCFGFile("Write", "DATABASE", "dbUser", dbUserTxt.Text);
            readWriteCFGFile("Write", "DATABASE", "dbPass", dbPassTxt.Text);
            dbDspLbl.Text = "Host: " + robotConfigSetting.dbHost + "\nDatabase Name: "
+ robotConfigSetting.dbName + "\nDatabase User: " + robotConfigSetting.dbUser +
"\nDatabase Password: " + robotConfigSetting.dbPass;
            MessageBox.Show("Database is changed successfully", "Configuration
Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }

        private void robotSaveBtn_Click(object sender, EventArgs e)
        {

if((robotNameTxt.Text!=robotConfigSetting.robotName)&&(robotNameTxt.Text!=""))
            {
                robotConfigSetting.robotName = robotNameTxt.Text;
                robotConfigSetting.Save();
                readWriteCFGFile("Write", "ROBOT", "robotName", robotNameTxt.Text);
                robotDspLbl.Text = "New Robot Name: " + robotConfigSetting.robotName;
                MessageBox.Show("Robot Name is changed successfully", "Configuration
Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else
            {
                MessageBox.Show("Please make sure that new robot name don't same value
and null value", "Configuration Message", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }
}
```

RobotAPP\MainFrm.cs



เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;
using System.Speech.Recognition;
using System.Speech.Synthesis;
using System.Speech;
using System.IO;
using System.Xml;

namespace BranchApp
{
    public partial class mainFrm : Form
    {
        Cart CartFrm = new Cart();
        BranchAppSetting dataSetting = new BranchAppSetting();
        GlobalVariables globalVar = new GlobalVariables();
        SpeechSynthesizer speechSynthesis = new SpeechSynthesizer();
        SpeechRecognitionEngine speechEngine = new SpeechRecognitionEngine(new
System.Globalization.CultureInfo("en-US"));
        MySqlConnection connection;
        MySqlCommand SQLCommand;


        public string[] department;
        public string[] sub_department;
        public string[] brand;
        public string[] model;
        public string[] gramLoad;
        public string headVoiceCommand;
        public int timeCount = 20;
        public string CFGPath =
(Environment.GetEnvironmentVariable("KMITL_SHOP",EnvironmentVariableTarget.Machine)) +
"\\Robot_Config.xml";
        public mainFrm()
        {
            InitializeComponent();
            loadCFG();
            FormBorderStyle = FormBorderStyle.None;
            WindowState = FormWindowState.Maximized;
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.dbHost + ";" +
"DATABASE=" + dataSetting.dbName + ";" + "UID=" + dataSetting.dbUser + ";" +
"PASSWORD=" + dataSetting.dbPass; //boong3013
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            menuBtnIndx1.Visible = false;
            menuBtnIndx2.Visible = false;
            menuBtnIndx3.Visible = false;
            menuBtnIndx4.Visible = false;
            menuBtnIndx5.Visible = false;


            #region Setup Speech Engine
            headVoiceCommand = dataSetting.robotName + " show me the ";
            string loadDB = "select
product.Barcode,department.Department,central_product.Sub_department,central_product.B
rand from product left join central_product on product.barcode=central_product.barcode
left join department on central_product.sub_department=department.sub_department where
brand is not null;";
            MySqlDataAdapter loadDatabase = new MySqlDataAdapter(loadDB, connection);
            DataTable loadeDB2DT = new DataTable();
            loadDatabase.Fill(loadeDB2DT);
            department = loadeDB2DT.AsEnumerable().Select(r =>
r.Field<string>("Department")).Distinct().ToArray();
            sub_department = loadeDB2DT.AsEnumerable().Select(r =>
r.Field<string>("Sub_department")).Distinct().ToArray();
            brand = loadeDB2DT.AsEnumerable().Select(r =>
r.Field<string>("Brand")).Distinct().ToArray();
            for (int i = 1; i <= department.Length; i++)
            {
                string stringChange = department[i - 1];
                department[i - 1] = headVoiceCommand + stringChange.ToLower();
            }
            for (int i = 1; i <= sub_department.Length; i++)
            {
                string stringChange = sub_department[i - 1];
                sub_department[i - 1] = headVoiceCommand + stringChange.ToLower();
            }
            for (int i = 1; i <= brand.Length; i++)
            {
                string stringChange = brand[i - 1];
                brand[i - 1] = headVoiceCommand + stringChange.ToLower();
            }

            gramLoad = new string[department.Length + sub_department.Length +
brand.Length];
            department.CopyTo(gramLoad, 0);
            sub_department.CopyTo(gramLoad, department.Length);
            brand.CopyTo(gramLoad, (department.Length + sub_department.Length));
            File.WriteAllLines("temp.txt", gramLoad);
            speechEngine.LoadGrammarAsync(new Grammar(new GrammarBuilder(new
Choices(gramLoad))));
            speechEngine.SetInputToDefaultAudioDevice();
            speechEngine.SpeechRecognized += new
EventHandler<SpeechRecognizedEventArgs>(speech_Detect);

            #endregion
        }


        private void Main_Load(object sender, EventArgs e)
        {
            for (int i = 1; i <= 9; i++)
            {

                string productPCB = "productPCB" + Convert.ToString(i);
                Control[] productPCBControl = this.Controls.Find(productPCB, true);
                if (productPCBControl != null)
                {
                    productPCBControl[0].Visible = false;
                }
                string productNameLbl = "productNameLbl" + Convert.ToString(i);
                Control[] productNameControl = this.Controls.Find(productNameLbl,
true);
                if (productNameControl != null)
                {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                productNameControl[0].Visible = false;
            }
            string productPriceLbl = "productPriceLbl" + Convert.ToString(i);
            Control[] productPriceControl = this.Controls.Find(productPriceLbl,
true);
            if (productPriceControl != null)
            {
                productPriceControl[0].Visible = false;
            }
        }

        productPreviousPageBtn.Visible = false;
        productNextPageBtn.Visible = false;
        mycartLbl.Visible = false;
        mainMenuLbl.Text = null;
        menuBtnIndx1.Visible = false;
        menuBtnIndx2.Visible = false;
        menuBtnIndx3.Visible = false;
        menuBtnIndx4.Visible = false;
        menuBtnIndx5.Visible = false;
        initialFrm();

    }

    #region Read Config File

    public void loadCFG()
    {
        XmlReader readTxt = XmlReader.Create(CFGPath);
        while (readTxt.Read())
        {
            if (readTxt.IsStartElement())
            {
                switch (readTxt.Name)
                {
                    case "Attribute":
                        string attribute = readTxt["Name"];
                        if (attribute != null)
                        {
                            readTxt.Read();
                            string context = readTxt.Value.Trim();
                            switch (attribute)
                            {
                                case "cfgUser":
                                    dataSetting.configUser = context;
                                    dataSetting.Save();
                                    break;
                                case "cfgPass":
                                    dataSetting.configPass = context;
                                    dataSetting.Save();
                                    break;
                                case "robotName":
                                    dataSetting.robotName = context;
                                    dataSetting.Save();
                                    break;
                                case "dbHost":
                                    dataSetting.dbHost = context;
                                    dataSetting.Save();
                                    break;
                                case "dbName":
                                    dataSetting.dbName = context;
                                    dataSetting.Save();
                                    break;
                                case "dbUser":
                                    dataSetting.dbUser = context;
                                    dataSetting.Save();
                                    break;
                                case "dbPass":
                                    dataSetting.dbPass = context;
                                    dataSetting.Save();
                                    break;
                            }
                        }
                        break;
                }
            }
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }

        #endregion

        public void initialFrm()
        {
            upMenuBtn.Visible = false;
            productNextPageBtn.Visible = false;
            productPreviousPageBtn.Visible = false;
            globalVar.menuPage = 1;
            homePCB.Image = Properties.Resources.CIRA_Logo;
            voiceEnginePcb.Image = Properties.Resources.micIcon;
            voicePanel("initial");
            string departmentSQL = "Select distinct Department from department";
            MySqlDataAdapter loadDept = new MySqlDataAdapter(departmentSQL,
connection);
            DataTable DeptTB = new DataTable();
            loadDept.Fill(DeptTB);
            globalVar.departmentArr = DeptTB.AsEnumerable().Select(r =>
r.Field<string>("Department")).ToArray();
            globalVar.lengthDept = globalVar.departmentArr.Length;
            globalVar.isDepartmentState = true;
            if (globalVar.isDepartmentState == true)
            {
                mainMenuLbl.Text = "Department";
                if(globalVar.lengthDept>=5)
                {
                    for (int i=1; i <= 5; i++)
                    {
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            downMenuBtn.Visible = false;
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text = globalVar.departmentArr[i - 1];
                        }
                    }
                }
                else
                {
                    for (int i = 1; i <= globalVar.lengthDept; i++)
                    {
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text = globalVar.departmentArr[i - 1];
                        }
                    }
                }
            }
        }

        #region Voice Engine Power Switch

        private void voiceEnginePcb_Click(object sender, EventArgs e)
        {
            if (voiceEnginePcb.BackColor == Color.White)
            {
                voicePanel("On");
                speechSynthesis.SpeakAsync("Please tell me something  about product
such as Department sub department or brand");
                speechEngine.RecognizeAsync(RecognizeMode.Multiple);
                speechCommandLogoPcb.Image = Properties.Resources.VoiceStart;
                voiceEnginePcb.BackColor = Color.Red;
            }
            else if (voiceEnginePcb.BackColor == Color.Red)
            {
                voicePanel("Off");
                speechEngine.RecognizeAsyncCancel();
                voiceEnginePcb.BackColor = Color.White;
                speechCommandLogoPcb.Image = Properties.Resources.micIcon;
            }
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        #endregion

        #region Speech Detection

        private void speech_Detect(object sender, SpeechRecognizedEventArgs e)
        {
            string speech = e.Result.Text;
            wordDetectLbl.Text = speech.Replace(headVoiceCommand, null);
            bool isDepartment = department.Contains(speech);
            bool isSubDepartment = sub_department.Contains(speech);
            bool isBrand = brand.Contains(speech);
            string[] evenDetect = { "0", "0", "0" };
            bool isOnGram = gramLoad.Contains(speech);
            if(isOnGram==true)
            {
                _time2ClcOrder.Start();
                timeCount = 20;
            }
            if (isDepartment == true)
            {
                evenDetect[0] = "1";
            }
            if (isSubDepartment == true)
            {
                evenDetect[1] = "1";
            }
            if (isBrand == true)
            {
                evenDetect[2] = "1";
            }
            string searchDetect = evenDetect[0] + evenDetect[1] + evenDetect[2];
            speech = speech.Replace(headVoiceCommand, null);

            switch (searchDetect)
            {
                case "000":
                    speechSynthesis.SpeakAsync("I don't know the " + speech);
                    break;

                case "001":
                    speechSynthesis.SpeakAsync(speech + "is found as brand");
                    voiceLoadProduct("Brand", speech);
                    break;

                case "010":
                    speechSynthesis.SpeakAsync(speech + "is found as sub department");
                    voiceLoadProduct("Sub_Department", speech);
                    break;

                case "011":
                    speechSynthesis.SpeakAsync(speech + "is sub department or brand");
                    break;

                case "100":
                    speechSynthesis.SpeakAsync(speech + "is found as department");
                    voiceLoadProduct("Department", speech);
                    break;

                case "101":
                    speechSynthesis.SpeakAsync(speech + "is department or brand");
                    break;

                case "110":
                    speechSynthesis.SpeakAsync(speech + "is department or sub
department");
                    break;

                case "111":
                    speechSynthesis.SpeakAsync(speech + "is department,sub department
or brand");
                    break;
            }
        }

        #endregion

        #region voice panel
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        public void voicePanel(string VoicePanelState)
        {
            switch (VoicePanelState)
            {
                case "initial":
                    timeCount = 20;
                    speechComPanel.Visible = false;
                    speechCommandLogoPcb.Visible = false;
                    wordDetectTopicLbl.Visible = false;
                    wordDetectLbl.Visible = false;
                    how2useTopicLbl.Visible = false;
                    timeRemainTopicLbl.Visible = false;
                    timeRemainLbl.Visible = false;
                    speechCommandLogoPcb.Image = Properties.Resources.micIcon;
                    break;
                case "On":
                    timeCount = 20;
                    speechComPanel.Visible = true;
                    speechCommandLogoPcb.Visible = true;
                    wordDetectTopicLbl.Visible = true;
                    wordDetectLbl.Visible = true;
                    how2useTopicLbl.Visible = true;
                    timeRemainTopicLbl.Visible = true;
                    timeRemainLbl.Visible = true;
                    break;
                case "Off":
                    _time2ClcOrder.Stop();
                    timeCount = 20;
                    timeRemainLbl.Text = null;
                    speechComPanel.Visible = false;
                    speechCommandLogoPcb.Visible = false;
                    wordDetectTopicLbl.Visible = false;
                    wordDetectLbl.Visible = false;
                    how2useTopicLbl.Visible = false;
                    timeRemainTopicLbl.Visible = false;
                    timeRemainLbl.Visible = false;
                    break;
            }
        }

        #endregion

        #region Voice Load Product

        public void voiceLoadProduct(string searchBy,string value)
        {
            globalVar.barCode = null;
            globalVar.brand = null;
            globalVar.model = null;
            globalVar.size = null;
            globalVar.pricePerUnit = null;
            globalVar.containUnit = null;
            globalVar.lengthProduct = 0;
            globalVar.productPage = 1;
            productPreviousPageBtn.Visible = false;
            productNextPageBtn.Visible = false;
            globalVar.product = null;

            switch(searchBy)
            {
                case "Brand":
                    string loadProdByBrand = "select
product.BarCode,central_product.Brand,central_product.Model,central_product.Size,centr
al_product.containUnit,central_product.pricePerContain,central_product.productImage
from product left join central_product on product.BarCode = central_product.BarCode
and product.sub_department = central_product.Sub_department and central_product.brand
='" + value + "' where brand is not null;";
                    MySqlDataAdapter loadProduct = new
MySqlDataAdapter(loadProdByBrand, connection);
                    globalVar.product = new DataTable();
                    loadProduct.Fill(globalVar.product);
                    globalVar.barCode = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("BarCode")).ToArray();
                    globalVar.brand = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
```

เอกสารนี้มีลิขสิทธิ์ สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    globalVar.model = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
                    globalVar.size = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
                    globalVar.pricePerUnit = globalVar.product.AsEnumerable().Select(r
=> r.Field<float>("PricePerContain")).ToArray();
                    globalVar.containUnit = globalVar.product.AsEnumerable().Select(r
=> r.Field<string>("ContainUnit")).ToArray();
                    globalVar.lengthProduct = globalVar.brand.Length;
                    DisplayProduct();
                    break;
                case "Department":
                    string loadProdByDepartment = "select
product.BarCode,central_product.Brand,central_product.Model,central_product.Size,centr
al_product.containUnit,central_product.pricePerContain,central_product.productImage,ce
ntral_product.sub_department,department.department from central_product left join
department on central_product.sub_department = department.sub_department and
department.department = '" + value + "' left join product on
central_product.barcode=product.barcode where department is not null;";
                    MySqlDataAdapter loadProductByDepartmentDA = new
MySqlDataAdapter(loadProdByDepartment, connection);
                    globalVar.product = new DataTable();
                    loadProductByDepartmentDA.Fill(globalVar.product);
                    globalVar.barCode = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("BarCode")).ToArray();
                    globalVar.brand = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
                    globalVar.model = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
                    globalVar.size = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
                    globalVar.pricePerUnit = globalVar.product.AsEnumerable().Select(r
=> r.Field<float>("PricePerContain")).ToArray();
                    globalVar.containUnit = globalVar.product.AsEnumerable().Select(r
=> r.Field<string>("ContainUnit")).ToArray();
                    globalVar.lengthProduct = globalVar.brand.Length;
                    DisplayProduct();
                    break;
                case "Sub_Department":
                    string loadProdBySubDepartment = "select
product.BarCode,central_product.Brand,central_product.Model,central_product.Size,centr
al_product.containUnit,central_product.pricePerContain,central_product.productImage,ce
ntral_product.sub_department from product left join central_product on product.BarCode
= central_product.BarCode and product.sub_department = central_product.Sub_department
and central_product.sub_department ='" + value + "' where brand is not null;";
                    MySqlDataAdapter loadProductBySubDepartmentDA = new
MySqlDataAdapter(loadProdBySubDepartment, connection);
                    globalVar.product = new DataTable();
                    loadProductBySubDepartmentDA.Fill(globalVar.product);
                    globalVar.barCode = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("BarCode")).ToArray();
                    globalVar.brand = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
                    globalVar.model = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
                    globalVar.size = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
                    globalVar.pricePerUnit = globalVar.product.AsEnumerable().Select(r
=> r.Field<float>("PricePerContain")).ToArray();
                    globalVar.containUnit = globalVar.product.AsEnumerable().Select(r
=> r.Field<string>("ContainUnit")).ToArray();
                    globalVar.lengthProduct = globalVar.brand.Length;
                    DisplayProduct();
                    break;
            }


        }
        #endregion
        #region returnQuantity

        public void returnQuantity(string Barcode, int Quantity)
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            connection.Open();
            SQLCommand = new MySqlCommand();
            SQLCommand.Connection = connection;
            SQLCommand.CommandText = "returnQuantity";
            SQLCommand.CommandType = CommandType.StoredProcedure;
            SQLCommand.Parameters.Add("?Barcode_Input", Barcode);
            SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.Parameters.Add("?Quantity_Input", Quantity);
            SQLCommand.Parameters["?Quantity_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.ExecuteNonQuery();
            connection.Close();

        }

        #endregion

        public void DisplayProduct()
        {

            for (int i = 1; i <= 9; i++)
            {
                string productImageStr = "productPCB" + Convert.ToString(i);
                Control[] productImage = this.Controls.Find(productImageStr, true);
                if (productImage != null)
                {
                    productImage[0].Visible = false;
                    productImage[0].BackgroundImage = null;
                }
                string productNameStr = "productNameLbl" + Convert.ToString(i);
                Control[] productNameLbl = this.Controls.Find(productNameStr, true);
                if (productNameLbl != null)
                {
                    productNameLbl[0].Text = null;
                    productNameLbl[0].Visible = false;
                }
                string productPriceStr = "productPriceLbl" + Convert.ToString(i);
                Control[] productPriceLbl = this.Controls.Find(productPriceStr, true);
                if (productPriceLbl != null)
                {
                    productPriceLbl[0].Text = null;
                    productPriceLbl[0].Visible = false;
                }

            }

            if((globalVar.lengthProduct - (9*(globalVar.productPage-1))) > 9)
            {
                mycartLbl.Visible = true;
                productNextPageBtn.Visible = true;
                if(globalVar.productPage>1)
                {
                    productPreviousPageBtn.Visible = true;
                }
                else
                {
                    productPreviousPageBtn.Visible = false;
                }
                for (int i = 1; i <= 9; i++)
                {

                    byte[] img = new byte[0];
                    img = (byte[])globalVar.product.Rows[(9 * (globalVar.productPage -
1)) + (i - 1)][6];
                    MemoryStream ms = new MemoryStream(img);
                    string productImageStr = "productPCB" + Convert.ToString(i);
                    Control[] productImage = this.Controls.Find(productImageStr,true);
                    if(productImage!=null)
                    {
                        productImage[0].Visible = true;
                        productImage[0].BackgroundImage = Bitmap.FromStream(ms);
                    }

                    string productNameStr = "productNameLbl" + Convert.ToString(i);
                    Control[] productNameLbl = this.Controls.Find(productNameStr,
true);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของลิขสิทธิ์ทุกครั้งที่มีการนำไปใช้

```
                    if (productNameLbl != null)
                    {
                        productNameLbl[0].Text = globalVar.brand[(9 *
(globalVar.productPage - 1)) + (i - 1)];
                        productNameLbl[0].Visible = true;
                    }
                    string productPriceStr = "productPriceLbl" + Convert.ToString(i);
                    Control[] productPriceCtrl = base.Controls.Find(productPriceStr,
true);
                    if (productPriceCtrl != null)
                    {
                        productPriceCtrl[0].Text =
Convert.ToString(globalVar.pricePerUnit[(9 * (globalVar.productPage - 1)) + (i - 1)]);
                        productPriceCtrl[0].Visible = true;
                    }
                }
            }
            else
            {
                mycartLbl.Visible = true;
                if (globalVar.productPage > 1)
                {
                    productPreviousPageBtn.Visible = true;
                }
                else
                {
                    productPreviousPageBtn.Visible = false;
                }
                productNextPageBtn.Visible = false;
                for (int i = 1; i <= (globalVar.lengthProduct - (9 *
(globalVar.productPage - 1))); i++)
                {
                    byte[] img = new byte[0];
                    img = (byte[])globalVar.product.Rows[(9 * (globalVar.productPage -
1)) + (i - 1)][6];
                    MemoryStream ms = new MemoryStream(img);
                    string productImageStr = "productPCB" + Convert.ToString(i);
                    Control[] productImage = this.Controls.Find(productImageStr,
true);
                    if (productImage != null)
                    {
                        productImage[0].Visible = true;
                        productImage[0].BackgroundImage = Bitmap.FromStream(ms);
                    }
                    string productNameStr = "productNameLbl" + Convert.ToString(i);
                    Control[] productNameLbl = this.Controls.Find(productNameStr,
true);
                    if (productNameLbl != null)
                    {
                        productNameLbl[0].Text = globalVar.brand[(9 *
(globalVar.productPage - 1)) + (i - 1)];
                        productNameLbl[0].Visible = true;
                    }
                    string productPriceStr = "productPriceLbl" + Convert.ToString(i);
                    Control[] productPriceCtrl = base.Controls.Find(productPriceStr,
true);
                    if (productPriceCtrl != null)
                    {
                        productPriceCtrl[0].Text =
Convert.ToString(globalVar.pricePerUnit[(9 * (globalVar.productPage - 1)) + (i - 1)]);
                        productPriceCtrl[0].Visible = true;
                    }
                }
            }
        }

        private void MyCart_Click(object sender, EventArgs e)
        {
            Cart mycartFrm = new Cart();
            mycartFrm.Show();
        }

        private void Form_Closing(object sender, FormClosingEventArgs e)
        {
            string[] lodeOrderList = File.ReadAllLines(globalVar.productTxtPath);
            int orderLength = lodeOrderList.Count();
            for (int i = 1; i <= orderLength; i++)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            {
                string[] splitTemp = lodeOrderList[i - 1].Split(new string[] {"\t"},
StringSplitOptions.None);
                returnQuantity(splitTemp[0], Convert.ToInt16(splitTemp[5]));

            }
            FileStream rebuild = File.Open(globalVar.productTxtPath,
FileMode.Truncate);
            rebuild.Close();
            Application.Exit();
        }

        private void Up_Click(object sender, EventArgs e)
        {
            if(globalVar.menuPage>1)
            {
                globalVar.menuPage = globalVar.menuPage - 1;
                downMenuBtn.Visible = true;
                if(globalVar.menuPage==1)
                {
                    upMenuBtn.Visible = false;
                }
            }
            else
            {
                    upMenuBtn.Visible = false;
            }

            if(globalVar.isDepartmentState == true)
            {
                for (int i = 1; i <= 5; i++)
                {
                    string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                    Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                    if (menuBtnControl != null)
                    {
                        menuBtnControl[0].Visible = true;
                        menuBtnControl[0].Text = globalVar.departmentArr[(5 *
(globalVar.menuPage - 1)) + (i - 1)];
                    }
                }
            }
            else
            {
                for (int i = 1; i <= 5; i++)
                {
                    string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                    Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                    if (menuBtnControl != null)
                    {
                        menuBtnControl[0].Visible = true;
                        menuBtnControl[0].Text = globalVar.sub_DepartmentArr[(5 *
(globalVar.menuPage - 1)) + (i - 1)];
                    }
                }
            }
        }

        private void pictureBox1_Click(object sender, EventArgs e)
        {
            goHome();
        }

        #region Go to home function
        public void goHome()
        {
            mycartLbl.Visible = false;
            if (globalVar.isDepartmentState == false)
            {
                initialFrm();
                globalVar.menuPage = 1;
                for (int i = 1; i <= 9; i++)
                {
                    string productImageStr = "productPCB" + Convert.ToString(i);
                    Control[] productImage = this.Controls.Find(productImageStr,
true);
                    if (productImage != null)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    {
                        productImage[0].Visible = false;
                        productImage[0].BackgroundImage = null;
                    }
                    string productNameStr = "productNameLbl" + Convert.ToString(i);
                    Control[] productNameLbl = this.Controls.Find(productNameStr,
true);
                    if (productNameLbl != null)
                    {
                        productNameLbl[0].Text = null;
                        productNameLbl[0].Visible = false;
                    }
                    string productPriceStr = "productPriceLbl" + Convert.ToString(i);
                    Control[] productPriceLbl = this.Controls.Find(productPriceStr,
true);
                    if (productPriceLbl != null)
                    {
                        productPriceLbl[0].Text = null;
                        productPriceLbl[0].Visible = false;
                    }
                }
            }
        }

        #endregion

        private void downMenuBtn_Click(object sender, EventArgs e)
        {
            globalVar.menuPage = globalVar.menuPage + 1;
            if (globalVar.isDepartmentState == true)
            {
                upMenuBtn.Visible = true;
                mainMenuLbl.Text = "Department";
                if (globalVar.lengthDept - (5*(globalVar.menuPage-1)) <= 5)
                {
                    downMenuBtn.Visible = false;
                    menuBtnIndx1.Visible = false;
                    menuBtnIndx2.Visible = false;
                    menuBtnIndx3.Visible = false;
                    menuBtnIndx4.Visible = false;
                    menuBtnIndx5.Visible = false;
                    for (int i = 1; i <= globalVar.lengthDept - (5 *
(globalVar.menuPage - 1)); i++)
                    {
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text =
globalVar.departmentArr[(5*(globalVar.menuPage-1)) + (i - 1)];
                        }
                    }
                }
                else
                {
                    for (int i = 1; i <= 5; i++)
                    {
                        upMenuBtn.Visible = true;
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text = globalVar.departmentArr[(5 *
(globalVar.menuPage-1)) + (i - 1)];
                        }
                    }
                }
            }
            else
            {
                upMenuBtn.Visible = true;
                mainMenuLbl.Text = "Sub Department";
                if (globalVar.lengthSub_Dept - (5 * (globalVar.menuPage - 1)) <= 5)
                {
                    downMenuBtn.Visible = false;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    menuBtnIndx1.Visible = false;
                    menuBtnIndx2.Visible = false;
                    menuBtnIndx3.Visible = false;
                    menuBtnIndx4.Visible = false;
                    menuBtnIndx5.Visible = false;
                    for (int i = 1; i <= globalVar.lengthSub_Dept - (5 *
(globalVar.menuPage - 1)); i++)
                    {
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text = globalVar.sub_DepartmentArr[(5 *
(globalVar.menuPage - 1)) + (i - 1)];
                        }
                    }
                }
                else
                {
                    for (int i = 1; i <= 5; i++)
                    {
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text = globalVar.sub_DepartmentArr[(5 *
(globalVar.menuPage - 1)) + (i - 1)];
                        }
                    }
                }
            }
        }

        private void productNextPageBtn_Click(object sender, EventArgs e)
        {
            if(globalVar.lengthProduct - (9*(globalVar.productPage-1))>9)
            {
                globalVar.productPage = globalVar.productPage + 1;
            }
            DisplayProduct();
        }

        private void productPreviousPageBtn_Click(object sender, EventArgs e)
        {
            if(globalVar.productPage>1)
            {
                globalVar.productPage = globalVar.productPage - 1;
            }
            DisplayProduct();
        }

        #region Menu Control

        private void menuBtnIndx1_Click(object sender, EventArgs e)
        {
            menuControl(menuBtnIndx1.Text);
        }
        private void menuBtnIndx2_Click(object sender, EventArgs e)
        {
            menuControl(menuBtnIndx2.Text);
        }
        private void menuBtnIndx3_Click(object sender, EventArgs e)
        {
            menuControl(menuBtnIndx3.Text);
        }
        private void menuBtnIndx4_Click(object sender, EventArgs e)
        {
            menuControl(menuBtnIndx4.Text);
        }
        private void menuBtnIndx5_Click(object sender, EventArgs e)
        {
            menuControl(menuBtnIndx5.Text);
        }

        #endregion
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        #region Product Box Display

        private void productPCB1_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 1;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB1.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB2_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 2;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB2.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB3_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 3;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB3.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB4_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 4;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB4.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB5_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 5;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB5.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB6_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 6;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB6.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB7_Click(object sender, EventArgs e)
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

```
            globalVar.productPCBIndx = 7;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB7.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB8_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 8;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB8.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }
        private void productPCB9_Click(object sender, EventArgs e)
        {
            globalVar.productPCBIndx = 9;
            _time2ClcOrder.Stop();
            timeCount = 20;
            timeRemainLbl.Text = "";
            Bitmap prodImage = new Bitmap(productPCB9.BackgroundImage);
            Order orderFrm = new Order(globalVar.productPage,
globalVar.productPCBIndx, globalVar.barCode, globalVar.brand, globalVar.model,
globalVar.size, globalVar.containUnit, globalVar.pricePerUnit, prodImage);
            orderFrm.Show();
        }

        #endregion

        #region Menu Control Function

        public void menuControl(string menuText)
        {
            if (globalVar.isDepartmentState == true)
            {
                globalVar.sub_DepartmentArr = null;
                menuBtnIndx1.Visible = false;
                menuBtnIndx2.Visible = false;
                menuBtnIndx3.Visible = false;
                menuBtnIndx4.Visible = false;
                menuBtnIndx5.Visible = false;
                homePCB.Image = Properties.Resources.homeIcon;
                mainMenuLbl.Text = "Sub Department";
                string sub_DeptLoad = "select distinct Sub_Department from department
where Department='" + menuText + "';";
                MySqlDataAdapter loadSub_Dept = new MySqlDataAdapter(sub_DeptLoad,
connection);
                DataTable sub_Department = new DataTable();
                loadSub_Dept.Fill(sub_Department);
                globalVar.sub_DepartmentArr = sub_Department.AsEnumerable().Select(r
=> r.Field<string>("Sub_Department")).ToArray();
                globalVar.lengthSub_Dept = globalVar.sub_DepartmentArr.Length;
                if (globalVar.lengthSub_Dept - (5 * (globalVar.menuPage - 1)) <= 5)
                {
                    downMenuBtn.Visible = false;
                    for (int i = 1; i <= globalVar.lengthSub_Dept - (5 *
(globalVar.menuPage - 1)); i++)
                    {
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text = globalVar.sub_DepartmentArr[(5 *
(globalVar.menuPage - 1)) + (i - 1)];
                        }
                    }
                }
                else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                {
                    downMenuBtn.Visible = true;
                    for (int i = 1; i <= 5; i++)
                    {
                        string menuBtn = "menuBtnIndx" + Convert.ToString(i);
                        Control[] menuBtnControl = this.Controls.Find(menuBtn, true);
                        if (menuBtnControl != null)
                        {
                            menuBtnControl[0].Visible = true;
                            menuBtnControl[0].Text = globalVar.sub_DepartmentArr[(5 *
(globalVar.menuPage - 1)) + (i - 1)];
                        }
                    }
                }
            }
            else
            {
                globalVar.barCode = null;
                globalVar.brand = null;
                globalVar.model = null;
                globalVar.size = null;
                globalVar.pricePerUnit = null;
                globalVar.containUnit = null;
                globalVar.lengthProduct = 0;
                globalVar.productPage = 1;
                productPreviousPageBtn.Visible = false;
                productNextPageBtn.Visible = false;
                globalVar.product = null;
                string loadProd = "select
product.BarCode,central_product.Brand,central_product.Model,central_product.Size,centr
al_product.containUnit,central_product.pricePerContain,central_product.productImage
from product left join central_product on product.BarCode = central_product.BarCode
and product.sub_department = central_product.Sub_department and product.sub_department
='" + menuText + "' where brand is not null;";
                MySqlDataAdapter loadProduct = new MySqlDataAdapter(loadProd,
connection);
                globalVar.product = new DataTable();
                loadProduct.Fill(globalVar.product);
                globalVar.barCode = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("BarCode")).ToArray();
                globalVar.brand = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Brand")).ToArray();
                globalVar.model = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Model")).ToArray();
                globalVar.size = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("Size")).ToArray();
                globalVar.pricePerUnit = globalVar.product.AsEnumerable().Select(r =>
r.Field<float>("PricePerContain")).ToArray();
                globalVar.containUnit = globalVar.product.AsEnumerable().Select(r =>
r.Field<string>("ContainUnit")).ToArray();
                globalVar.lengthProduct = globalVar.brand.Length;
                DisplayProduct();
            }
            globalVar.isDepartmentState = false;
        }

        #endregion

        private void cancleOrder_Tick(object sender, EventArgs e)
        {
            timeCount = (((timeCount*1000) - _time2ClcOrder.Interval)/1000);
            if(timeCount>10)
            {
                timeRemainLbl.ForeColor = Color.Black;
            }
            else if ((timeCount <= 10) && (timeCount > 5))
            {
                timeRemainLbl.ForeColor = Color.Yellow;
            }
            else if ((timeCount <= 5) && (timeCount > 0))
            {
                timeRemainLbl.ForeColor = Color.Red;
                Console.Beep();
            }
            else if(timeCount==0)
            {
                timeRemainLbl.Text = Convert.ToString(timeCount);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                _time2ClcOrder.Stop();
                clearOrder();

            }
            timeRemainLbl.Text = Convert.ToString(timeCount);
        }
        public void clearOrder()
        {
            DialogResult clearResult = MessageBox.Show("Do you want to clear all
product in your cart?", "My Cart", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
            if (clearResult == DialogResult.Yes)
            {
                globalVar.orderBarcode = null;
                globalVar.orderBrand = null;
                globalVar.orderModel = null;
                globalVar.orderSize = null;
                globalVar.orderPriePU = null;
                globalVar.orderAmount = null;
                globalVar.orderTotalPrice = null;
                FileStream rebuild = File.Open(globalVar.productTxtPath,
FileMode.Truncate);
                rebuild.Close();
                timeCount = 20;
                speechSynthesis.SpeakAsync("Your cart is empty");
            }
            else
            {
                timeCount = 20;
                _time2ClcOrder.Start();

            }
        }

        private void mainFrm_Activated(object sender, EventArgs e)
        {
            if(voiceEnginePcb.BackColor==Color.Red)
            {
                _time2ClcOrder.Start();
            }
        }
    }
}
```

PopUp.cs

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;
using System.IO;

namespace BranchApp
{
    public partial class Order : Form
    {
        int count = 0;
        public Bitmap prodImg;
        GlobalVariables globalVar = new GlobalVariables();
        BranchAppSetting branchSetting = new BranchAppSetting();
        MySqlConnection connection;
        MySqlCommand SQLCommand;
        public bool isOrder = false;
        public Order(int productPage,int productPcbIndex,string[] BarCode,string[]
Brand,string[] Model,string[] Size,string[] Unit,float[] PricePerUnit,Bitmap
ProductImg)
        {
            InitializeComponent();
            globalVar.barCode = BarCode;
            globalVar.brand = Brand;
            globalVar.model = Model;
            globalVar.size = Size;
            globalVar.containUnit = Unit;
            globalVar.pricePerUnit = PricePerUnit;
            globalVar.productPage = productPage;
            globalVar.productPCBIndx = productPcbIndex;
            prodImg = ProductImg;
        }

        private void PopUp_Load(object sender, EventArgs e)
        {
            globalVar.mysqlConnectionString = "SERVER=" + branchSetting.dbHost + ";" +
"DATABASE=" + branchSetting.dbName + ";" + "UID=" + branchSetting.dbUser + ";" +
"PASSWORD=" + branchSetting.dbPass;
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            popBarcodeDispLbl.Text=globalVar.barCode[(9 * (globalVar.productPage - 1))
+ (globalVar.productPCBIndx - 1)];
            popBrandLBL.Text = globalVar.brand[(9 * (globalVar.productPage - 1)) +
(globalVar.productPCBIndx - 1)];
            popModelLbl.Text = globalVar.model[(9 * (globalVar.productPage - 1)) +
(globalVar.productPCBIndx - 1)];
            popSizeLbl.Text = globalVar.size[(9 * (globalVar.productPage - 1)) +
(globalVar.productPCBIndx - 1)];
            popUnitLbl.Text=globalVar.containUnit[(9 * (globalVar.productPage - 1)) +
(globalVar.productPCBIndx - 1)];
            popPriceLbl.Text = Convert.ToString(globalVar.pricePerUnit[(9 *
(globalVar.productPage - 1)) + (globalVar.productPCBIndx - 1)]);
            popProductPcb.Image = prodImg;
        }

        public bool checkQuantityAvilable(string Barcode,bool isPlus)
        {
            connection.Open();
            SQLCommand = new MySqlCommand();
            SQLCommand.Connection = connection;
            SQLCommand.CommandText = "orderQuantityChange";
            SQLCommand.CommandType = CommandType.StoredProcedure;
            SQLCommand.Parameters.Add("?Barcode_Input", Barcode);
            SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;

            SQLCommand.Parameters.Add("?isPlus", isPlus);
            SQLCommand.Parameters["?isPlus"].Direction = ParameterDirection.Input;
            SQLCommand.Parameters.AddWithValue("?isDone", MySqlDbType.Enum);
```

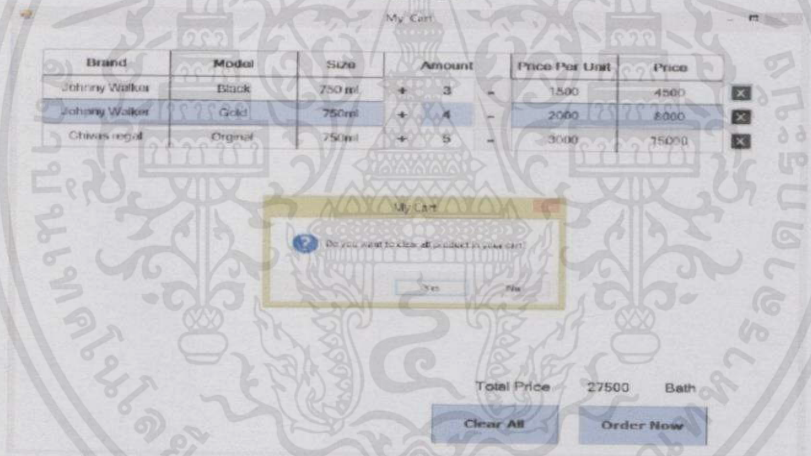
เอกสารนี้เป็นเอกสาร... ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
ไม่ว่า...

```
            SQLCommand.Parameters["?isDone"].Direction = ParameterDirection.Output;
            SQLCommand.ExecuteNonQuery();
            connection.Close();
            return Convert.ToBoolean(SQLCommand.Parameters["?isDone"].Value);
        }

        #region returnQuantity

        public void returnQuantity(string Barcode, int Quantity)
        {
            connection.Open();
            SQLCommand = new MySqlCommand();
            SQLCommand.Connection = connection;
            SQLCommand.CommandText = "returnQuantity";
            SQLCommand.CommandType = CommandType.StoredProcedure;
            SQLCommand.Parameters.Add("?Barcode_Input", Barcode);
            SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.Parameters.Add("?Quantity_Input", Quantity);
            SQLCommand.Parameters["?Quantity_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.ExecuteNonQuery();
            connection.Close();

        }

        #endregion

        private void button2_Click(object sender, EventArgs e)
        {
            bool result = checkQuantityAvilable(popBarcodeDispLbl.Text,true);
            if(result==true)
            {
                count = count + 1;
                showCountLbl.Text = Convert.ToString(count);
            }
            else
            {
                button2.Visible = false;
                MessageBox.Show("This product is avaliable on " + showCountLbl.Text +
" items", "System Message", MessageBoxButtons.OK, MessageBoxIcon.Hand);
            }
        }

        private void button1_Click(object sender, EventArgs e)
        {
            if (button2.Visible == false)
            {
                button2.Visible = true;
            }
            if (count != 0)
            {
                bool result = checkQuantityAvilable(popBarcodeDispLbl.Text, false);
                if (result == true)
                {
                    count = count - 1;
                    showCountLbl.Text = Convert.ToString(count);
                }
            }
        }

        private void AddCartBtn_Click(object sender, EventArgs e)
        {
            string[] loadProduct= File.ReadAllLines(globalVar.productTxtPath);
            int lengthOfOrder = loadProduct.Length;
            string[] splitDetail = new string[6];
            string[] existProduct = new string[lengthOfOrder];
            for(int i=1;i<=lengthOfOrder;i++)
            {
                splitDetail = loadProduct[i - 1].Split(new string[] { "\t" },
StringSplitOptions.None);
                existProduct[i - 1] = splitDetail[0];
            }
            List<string> productlist = new List<string>(loadProduct);
            string productAdd = popBarcodeDispLbl.Text + "\t" + popBrandLBL.Text +
"\t" + popModelLbl.Text + "\t" + popSizeLbl.Text + "\t" + popPriceLbl.Text + "\t" +
showCountLbl.Text;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            bool checkExist = existProduct.Contains(popBarcodeDispLbl.Text);
            if(checkExist==false)
            {
                productlist.Add(productAdd);
                File.WriteAllLines(globalVar.productTxtPath, productlist,
Encoding.UTF8);
            }
            else
            {
                MessageBox.Show("This item is exist in your cart already", "System
Warning", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            isOrder = true;
            this.Close();

        }

        private void Form_Closing(object sender, FormClosingEventArgs e)
        {
            if(isOrder==false)
            {
                returnQuantity(popBarcodeDispLbl.Text,
Convert.ToInt16(showCountLbl.Text));
            }

        }
    }
}
```

Cart.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using MySql.Data.MySqlClient;
using System.Configuration;
using System.Data.SqlClient;

namespace BranchApp
{
    public partial class Cart : Form
    {
        BranchAppSetting dataSetting = new BranchAppSetting();
        GlobalVariables globalVar = new GlobalVariables();
        MySqlCommand SQLCommand;
        MySqlConnection connection;
        public Cart()
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น รวมถึงห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            globalVar.mysqlConnectionString = "SERVER=" + dataSetting.dbHost + ";" +
"DATABASE=" + dataSetting.dbName + ";" + "UID=" + dataSetting.dbUser + ";" +
"PASSWORD=" + dataSetting.dbPass; //boong3013
            connection = new MySqlConnection(globalVar.mysqlConnectionString);
            if (connection.State == ConnectionState.Open)
            {
                Console.Beep();
                connection.Close();
            }
            InitializeComponent();
            clearControl();
            loadOrder();
            showOrderList();
        }

        #region Generate  Order ID
        public void generateOrderID()
        {
            globalVar.orderID = 0;
            string readOrderStr = "select OrderID from order_pending where RobotID =
'" + dataSetting.robotName + "';";
            MySqlDataAdapter readOrderID = new MySqlDataAdapter(readOrderStr,
connection);
            DataTable OrderIDTB = new DataTable();
            readOrderID.Fill(OrderIDTB);
            int[] orderIDArr = OrderIDTB.AsEnumerable().Select(r =>
r.Field<int>("OrderID")).ToArray();

            if (orderIDArr.Length>=1)
            {
                globalVar.orderID = orderIDArr.Max() + 1;
            }
            else
            {
                globalVar.orderID = 1;

            }

        }
        #endregion

        #region Load Order From Text File
        public void loadOrder()
        {
            string[] loadproduct = File.ReadAllLines(globalVar.productTxtPath);
            int amountProduct = loadproduct.Length;
            for (int i = 1; i <= amountProduct; i++)
            {
                string[] dataDetail = null;
                dataDetail = loadproduct[i - 1].Split(new string[] { "\t" },
StringSplitOptions.None);
                globalVar.orderBarcode.Add(dataDetail[0]);
                globalVar.orderBrand.Add(dataDetail[1]);
                globalVar.orderModel.Add(dataDetail[2]);
                globalVar.orderSize.Add(dataDetail[3]);
                globalVar.orderPriePU.Add(Convert.ToDouble(dataDetail[4]));
                globalVar.orderAmount.Add(Convert.ToInt16(dataDetail[5]));
                globalVar.orderTotalPrice.Add(Convert.ToDouble(dataDetail[4]) *
Convert.ToInt16(dataDetail[5]));
            }
        }

        #endregion

        #region Show Order List Function
        public void showOrderList()
        {
            int amountOfOrder = globalVar.orderBarcode.Count;
            if(amountOfOrder>0)
            {
                if (amountOfOrder <= 12)
                {
                    for (int i = 1; i <= amountOfOrder; i++)
                    {
                        string brandStr = "brandDispLbl" + Convert.ToString(i);
                        Control[] brandLblCtrl = this.Controls.Find(brandStr, true);
                        if (brandLblCtrl != null)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        {
                            brandLblCtrl[0].Text = globalVar.orderBrand[i - 1];
                            brandLblCtrl[0].Visible = true;
                        }
                        string modelStr = "modelDispLbl" + Convert.ToString(i);
                        Control[] modelLblCtrl = this.Controls.Find(modelStr, true);
                        if (modelLblCtrl != null)
                        {
                            modelLblCtrl[0].Text = globalVar.orderModel[i - 1];
                            modelLblCtrl[0].Visible = true;
                        }
                        string sizeStr = "sizeDispLbl" + Convert.ToString(i);
                        Control[] sizeLblCtrl = this.Controls.Find(sizeStr, true);
                        if (sizeLblCtrl != null)
                        {
                            sizeLblCtrl[0].Text = globalVar.orderSize[i - 1];
                            sizeLblCtrl[0].Visible = true;
                        }
                        string plusStr = "plusBtn" + Convert.ToString(i);
                        Control[] plusBtnCtrl = this.Controls.Find(plusStr, true);
                        if (plusBtnCtrl != null)
                        {
                            plusBtnCtrl[0].Visible = true;
                        }
                        string amountStr = "amountDispLbl" + Convert.ToString(i);
                        Control[] amountLblCtrl = this.Controls.Find(amountStr, true);
                        if (amountLblCtrl != null)
                        {
                            amountLblCtrl[0].Text =
Convert.ToString(globalVar.orderAmount[i - 1]);
                            amountLblCtrl[0].Visible = true;
                        }
                        string minusStr = "minusBtn" + Convert.ToString(i);
                        Control[] minusBtnCtrl = this.Controls.Find(minusStr, true);
                        if (minusBtnCtrl != null)
                        {
                            minusBtnCtrl[0].Visible = true;
                        }
                        string pricePUStr = "pricePUDispLbl" + Convert.ToString(i);
                        Control[] pricePULblCtrl = this.Controls.Find(pricePUStr,
true);
                        if (pricePULblCtrl != null)
                        {
                            pricePULblCtrl[0].Text =
Convert.ToString(globalVar.orderPriePU[i - 1]);
                            pricePULblCtrl[0].Visible = true;
                        }
                        string priceStr = "priceDispLbl" + Convert.ToString(i);
                        Control[] priceLblCtrl = this.Controls.Find(priceStr, true);
                        if (priceLblCtrl != null)
                        {
                            priceLblCtrl[0].Text =
Convert.ToString(globalVar.orderTotalPrice[i-1]);
                            priceLblCtrl[0].Visible = true;
                        }
                        string cancleStr = "cancleBtn" + Convert.ToString(i);
                        Control[] cancleBtnCtrl = this.Controls.Find(cancleStr, true);
                        if (cancleBtnCtrl != null)
                        {
                            cancleBtnCtrl[0].Visible = true;
                        }
                    }
                }
                else
                {
                    vScrollBar1.Visible = true;
                    vScrollBar1.Maximum = amountOfOrder - 12;
                    vScrollBar1.Minimum = 0;
                    for (int i = 1; i <= 12; i++)
                    {
                        string brandStr = "brandDispLbl" + Convert.ToString(i);
                        Control[] brandLblCtrl = this.Controls.Find(brandStr, true);
                        if (brandLblCtrl != null)
                        {
                            brandLblCtrl[0].Text =
globalVar.orderBrand[(vScrollBar1.Value + i) - 1];
                            brandLblCtrl[0].Visible = true;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        }
                        string modelStr = "modelDispLbl" + Convert.ToString(i);
                        Control[] modelLblCtrl = this.Controls.Find(modelStr, true);
                        if (modelLblCtrl != null)
                        {
                            modelLblCtrl[0].Text =
globalVar.orderModel[(vScrollBar1.Value + i) - 1];
                            modelLblCtrl[0].Visible = true;
                        }
                        string sizeStr = "sizeDispLbl" + Convert.ToString(i);
                        Control[] sizeLblCtrl = this.Controls.Find(sizeStr, true);
                        if (sizeLblCtrl != null)
                        {
                            sizeLblCtrl[0].Text =
globalVar.orderSize[(vScrollBar1.Value + i) - 1];
                            sizeLblCtrl[0].Visible = true;
                        }
                        string plusStr = "plusBtn" + Convert.ToString(i);
                        Control[] plusBtnCtrl = this.Controls.Find(plusStr, true);
                        if (plusBtnCtrl != null)
                        {
                            plusBtnCtrl[0].Visible = true;
                        }
                        string amountStr = "amountDispLbl" + Convert.ToString(i);
                        Control[] amountLblCtrl = this.Controls.Find(amountStr, true);
                        if (amountLblCtrl != null)
                        {
                            amountLblCtrl[0].Text =
Convert.ToString(globalVar.orderAmount[(vScrollBar1.Value + i) - 1]);
                            amountLblCtrl[0].Visible = true;
                        }
                        string minusStr = "minusBtn" + Convert.ToString(i);
                        Control[] minusBtnCtrl = this.Controls.Find(minusStr, true);
                        if (minusBtnCtrl != null)
                        {
                            minusBtnCtrl[0].Visible = true;
                        }
                        string pricePUStr = "pricePUDispLbl" + Convert.ToString(i);
                        Control[] pricePULblCtrl = this.Controls.Find(pricePUStr,
true);
                        if (pricePULblCtrl != null)
                        {
                            pricePULblCtrl[0].Text =
Convert.ToString(globalVar.orderPriePU[(vScrollBar1.Value + i) - 1]);
                            pricePULblCtrl[0].Visible = true;
                        }
                        string priceStr = "priceDispLbl" + Convert.ToString(i);
                        Control[] priceLblCtrl = this.Controls.Find(priceStr, true);
                        if (priceLblCtrl != null)
                        {
                            priceLblCtrl[0].Text =
Convert.ToString(globalVar.orderTotalPrice[(vScrollBar1.Value + i) - 1]);
                            priceLblCtrl[0].Visible = true;
                        }
                        string cancleStr = "cancleBtn" + Convert.ToString(i);
                        Control[] cancleBtnCtrl = this.Controls.Find(cancleStr, true);
                        if (cancleBtnCtrl != null)
                        {
                            cancleBtnCtrl[0].Visible = true;
                        }
                    }
                }
                clearAllLbl.Visible=true;
                orderNowLbl.Visible=true;
                priceTagLbl.Visible = true;
                totalPriceLbl.Visible = true;
                totalPrice();
                priceUnitLbl.Visible = true;
            }
            else
            {
                clearAllLbl.Visible = false;
                orderNowLbl.Visible = false;
                priceTagLbl.Visible = false;
                totalPriceLbl.Text = null;
                totalPriceLbl.Visible = false;
                priceUnitLbl.Visible = false;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            }
        }

        #endregion

        #region Total Price Calculate
        public void totalPrice()
        {
            string totalSumPrice = Convert.ToString(globalVar.orderTotalPrice.Sum());
            totalPriceLbl.Text = totalSumPrice;
        }

        #endregion

        #region Delete Order Function
        public void deleteOrder(int controlIndx)
        {
            returnQuantity(globalVar.orderBarcode[(controlIndx + vScrollBar1.Value) -
1], globalVar.orderAmount[(controlIndx + vScrollBar1.Value) - 1]);
            globalVar.orderBarcode.RemoveAt((controlIndx + vScrollBar1.Value) - 1);
            globalVar.orderBrand.RemoveAt((controlIndx + vScrollBar1.Value) - 1);
            globalVar.orderModel.RemoveAt((controlIndx + vScrollBar1.Value) - 1);
            globalVar.orderSize.RemoveAt((controlIndx + vScrollBar1.Value) - 1);
            globalVar.orderAmount.RemoveAt((controlIndx + vScrollBar1.Value) - 1);
            globalVar.orderPriePU.RemoveAt((controlIndx + vScrollBar1.Value) - 1);
            globalVar.orderTotalPrice.RemoveAt((controlIndx + vScrollBar1.Value) - 1);
            int amountOfOrder = globalVar.orderBrand.Count();
            string[] reBuildOrderTemp=new string[amountOfOrder];

            for(int i=1;i<=amountOfOrder;i++)
            {
                reBuildOrderTemp[i-1] = globalVar.orderBarcode[i - 1] + "\t" +
globalVar.orderBrand[i - 1] + "\t" + globalVar.orderModel[i - 1] + "\t" +
globalVar.orderSize[i - 1] + "\t" + globalVar.orderPriePU[i - 1] + "\t" +
globalVar.orderAmount[i - 1];
            }
            File.WriteAllLines(globalVar.productTxtPath, reBuildOrderTemp,
Encoding.UTF8);
            clearControl();
            showOrderList();
        }

        #endregion

        #region CheckQuantityAvilable Function

        public bool checkQuantityAvilable(string Barcode, bool isPlus)
        {
            connection.Open();
            SQLCommand = new MySqlCommand();
            SQLCommand.Connection = connection;
            SQLCommand.CommandText = "orderQuantityChange";
            SQLCommand.CommandType = CommandType.StoredProcedure;
            SQLCommand.Parameters.Add("?Barcode_Input", Barcode);
            SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;

            SQLCommand.Parameters.Add("?isPlus", isPlus);
            SQLCommand.Parameters["?isPlus"].Direction = ParameterDirection.Input;
            SQLCommand.Parameters.AddWithValue("?isDone", MySqlDbType.Enum);
            SQLCommand.Parameters["?isDone"].Direction = ParameterDirection.Output;
            SQLCommand.ExecuteNonQuery();
            connection.Close();
            return Convert.ToBoolean(SQLCommand.Parameters["?isDone"].Value);
        }

        #endregion

        #region returnQuantity
        public void returnQuantity(string Barcode,int Quantity)
        {
            connection.Open();
            SQLCommand = new MySqlCommand();
            SQLCommand.Connection = connection;
            SQLCommand.CommandText = "returnQuantity";
            SQLCommand.CommandType = CommandType.StoredProcedure;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            SQLCommand.Parameters.Add("?Barcode_Input", Barcode);
            SQLCommand.Parameters["?Barcode_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.Parameters.Add("?Quantity_Input", Quantity);
            SQLCommand.Parameters["?Quantity_Input"].Direction =
ParameterDirection.Input;
            SQLCommand.ExecuteNonQuery();
            connection.Close();

        }

        #endregion

        #region Product Amount Change
        public void amountChange(bool isIncrease,int controlIndx)
        {
            string amountLblStr = "amountDispLbl"+Convert.ToString(controlIndx);
            Control[] amountLbl = this.Controls.Find(amountLblStr,true);
            int amount=0;
            if(amountLbl!=null)
            {
                amount = Convert.ToInt16(amountLbl[0].Text);
                if (isIncrease == true)
                {
                    bool result =
checkQuantityAvilable(globalVar.orderBarcode[(vScrollBar1.Value+controlIndx)-1],
true);
                    if (result == true)
                    {
                        amount = amount + 1;;
                    }
                    else
                    {
                        Control[] plusBtnHind = this.Controls.Find("plusBtn" +
controlIndx, true);
                        if(plusBtnHind != null)
                        {
                            plusBtnHind[0].Visible = false;
                        }
                        MessageBox.Show("This product is avaliable on " + amount + "
items", "System Message", MessageBoxButtons.OK, MessageBoxIcon.Hand);
                    }

                    globalVar.orderAmount[(vScrollBar1.Value+controlIndx)-1]=amount;
                    amountLbl[0].Text = Convert.ToString(amount);
                    Control[] priceLbl = this.Controls.Find("priceDispLbl" +
Convert.ToString(controlIndx), true);
                    if(priceLbl!=null)
                    {
                        priceLbl[0].Text = null;
                        Control[] pricePU = this.Controls.Find("pricePUDispLbl" +
Convert.ToString(controlIndx), true);
                        if(pricePU!=null)
                        {
                            globalVar.orderTotalPrice[(vScrollBar1.Value+controlIndx)-
1]=(globalVar.orderAmount[(vScrollBar1.Value+controlIndx)-
1])*(globalVar.orderPriePU[(vScrollBar1.Value+controlIndx)-1]);
                            priceLbl[0].Text =
Convert.ToString(globalVar.orderTotalPrice[(vScrollBar1.Value + controlIndx) - 1]);
                            totalPrice();
                        }
                    }
                }
                else
                {
                    if(amount>0)
                    {

                        bool result =
checkQuantityAvilable(globalVar.orderBarcode[(vScrollBar1.Value + controlIndx) - 1],
false);
                        if (result == true)
                        {
                            amount = amount - 1;
                        }
                        Control[] plusBtnHind = this.Controls.Find("plusBtn" +
controlIndx,true);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        if(plusBtnHind != null)
                        {
                            if(plusBtnHind[0].Visible==false)
                            {
                                plusBtnHind[0].Visible = true;
                            }
                        }

                        globalVar.orderAmount[(vScrollBar1.Value + controlIndx) - 1] =
amount;
                        amountLbl[0].Text = Convert.ToString(amount);
                        Control[] priceLbl = this.Controls.Find("priceDispLbl" +
Convert.ToString(controlIndx), true);
                        if (priceLbl != null)
                        {
                            priceLbl[0].Text = null;
                            Control[] pricePU = this.Controls.Find("pricePUDispLbl" +
Convert.ToString(controlIndx), true);
                            if (pricePU != null)
                            {
                                globalVar.orderTotalPrice[(vScrollBar1.Value +
controlIndx) - 1] = (globalVar.orderAmount[(vScrollBar1.Value + controlIndx) - 1]) *
(globalVar.orderPriePU[(vScrollBar1.Value + controlIndx) - 1]);
                                priceLbl[0].Text =
Convert.ToString(globalVar.orderTotalPrice[(vScrollBar1.Value + controlIndx) - 1]);
                                totalPrice();
                            }
                        }
                    }
                }
            }

        }

        #endregion

        #region Clear All Control
        public void clearControl()
        {
            for(int i=1;i<=12;i++)
            {
                string brandStr = "brandDispLbl" + Convert.ToString(i);
                Control[] brandLblCtrl = this.Controls.Find(brandStr, true);
                if(brandLblCtrl!=null)
                {
                    brandLblCtrl[0].Text = null;
                    brandLblCtrl[0].Visible = false;
                }
                string modelStr = "modelDispLbl" + Convert.ToString(i);
                Control[] modelLblCtrl = this.Controls.Find(modelStr, true);
                if (modelLblCtrl != null)
                {
                    modelLblCtrl[0].Text = null;
                    modelLblCtrl[0].Visible = false;
                }
                string sizeStr = "sizeDispLbl" + Convert.ToString(i);
                Control[] sizeLblCtrl = this.Controls.Find(sizeStr, true);
                if (sizeLblCtrl != null)
                {
                    sizeLblCtrl[0].Text = null;
                    sizeLblCtrl[0].Visible = false;
                }
                string plusStr = "plusBtn" + Convert.ToString(i);
                Control[] plusBtnCtrl = this.Controls.Find(plusStr, true);
                if (plusBtnCtrl != null)
                {
                    plusBtnCtrl[0].Visible = false;
                }
                string amountStr = "amountDispLbl" + Convert.ToString(i);
                Control[] amountLblCtrl = this.Controls.Find(amountStr, true);
                if (amountLblCtrl != null)
                {
                    amountLblCtrl[0].Text = null;
                    amountLblCtrl[0].Visible = false;
                }
                string minusStr = "minusBtn" + Convert.ToString(i);
                Control[] minusBtnCtrl = this.Controls.Find(minusStr, true);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            if (minusBtnCtrl != null)
            {
                minusBtnCtrl[0].Visible = false;
            }
            string pricePUStr = "pricePUDispLbl" + Convert.ToString(i);
            Control[] pricePULblCtrl = this.Controls.Find(pricePUStr, true);
            if (pricePULblCtrl != null)
            {
                pricePULblCtrl[0].Text = null;
                pricePULblCtrl[0].Visible = false;
            }
            string priceStr = "priceDispLbl" + Convert.ToString(i);
            Control[] priceLblCtrl = this.Controls.Find(priceStr, true);
            if (priceLblCtrl != null)
            {
                priceLblCtrl[0].Text = null;
                priceLblCtrl[0].Visible = false;
            }
            string cancleStr = "cancleBtn" + Convert.ToString(i);
            Control[] cancleBtnCtrl = this.Controls.Find(cancleStr, true);
            if (cancleBtnCtrl != null)
            {
                cancleBtnCtrl[0].Visible = false;
            }

            priceTagLbl.Visible = false;
            totalPriceLbl.Text = null;
            totalPriceLbl.Visible = false;
            priceUnitLbl.Visible = false;
            clearAllLbl.Visible = false;
            orderNowLbl.Visible = false;
            vScrollBar1.Visible = false;

        }
    }

    #endregion

    #region PlusBtn No.1 - No.12

    private void plusBtn1_Click(object sender, EventArgs e)
    {
        amountChange(true, 1);
    }

    private void plusBtn2_Click(object sender, EventArgs e)
    {
        amountChange(true, 2);
    }

    private void plusBtn3_Click(object sender, EventArgs e)
    {
        amountChange(true, 3);
    }

    private void plusBtn4_Click(object sender, EventArgs e)
    {
        amountChange(true, 4);
    }

    private void plusBtn5_Click(object sender, EventArgs e)
    {
        amountChange(true, 5);
    }

    private void plusBtn6_Click(object sender, EventArgs e)
    {
        amountChange(true, 6);
    }

    private void plusBtn7_Click(object sender, EventArgs e)
    {
        amountChange(true, 7);
    }

    private void plusBtn8_Click(object sender, EventArgs e)
    {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        amountChange(true, 8);
    }

    private void plusBtn9_Click(object sender, EventArgs e)
    {
        amountChange(true, 9);
    }

    private void plusBtn10_Click(object sender, EventArgs e)
    {
        amountChange(true, 10);
    }

    private void plusBtn11_Click(object sender, EventArgs e)
    {
        amountChange(true, 11);
    }

    private void plusBtn12_Click(object sender, EventArgs e)
    {
        amountChange(true, 12);
    }

    #endregion

    #region MinusBtn No.1 - No.12

    private void minusBtn1_Click(object sender, EventArgs e)
    {
        amountChange(false, 1);
    }

    private void minusBtn2_Click(object sender, EventArgs e)
    {
        amountChange(false, 2);
    }

    private void minusBtn3_Click(object sender, EventArgs e)
    {
        amountChange(false, 3);
    }

    private void minusBtn4_Click(object sender, EventArgs e)
    {
        amountChange(false, 4);
    }

    private void minusBtn5_Click(object sender, EventArgs e)
    {
        amountChange(false, 5);
    }

    private void minusBtn6_Click(object sender, EventArgs e)
    {
        amountChange(false, 6);
    }

    private void minusBtn7_Click(object sender, EventArgs e)
    {
        amountChange(false, 7);
    }

    private void minusBtn8_Click(object sender, EventArgs e)
    {
        amountChange(false, 8);
    }

    private void minusBtn9_Click(object sender, EventArgs e)
    {
        amountChange(false, 9);
    }

    private void minusBtn10_Click(object sender, EventArgs e)
    {
        amountChange(false, 10);
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
private void minusBtn11_Click(object sender, EventArgs e)
{
    amountChange(false, 11);
}

private void minusBtn12_Click(object sender, EventArgs e)
{
    amountChange(false, 12);
}

#endregion

#region CancleBtn No.1 - No.12
private void cancleBtn1_Click(object sender, EventArgs e)
{
    deleteOrder(1);
}

private void cancleBtn2_Click(object sender, EventArgs e)
{
    deleteOrder(2);
}

private void cancleBtn3_Click(object sender, EventArgs e)
{
    deleteOrder(3);
}

private void cancleBtn4_Click(object sender, EventArgs e)
{
    deleteOrder(4);
}

private void cancleBtn5_Click(object sender, EventArgs e)
{
    deleteOrder(5);
}

private void cancleBtn6_Click(object sender, EventArgs e)
{
    deleteOrder(6);
}

private void cancleBtn7_Click(object sender, EventArgs e)
{
    deleteOrder(7);
}

private void cancleBtn8_Click(object sender, EventArgs e)
{
    deleteOrder(8);
}

private void cancleBtn9_Click(object sender, EventArgs e)
{
    deleteOrder(9);
}

private void cancleBtn10_Click(object sender, EventArgs e)
{
    deleteOrder(10);
}

private void cancleBtn11_Click(object sender, EventArgs e)
{
    deleteOrder(11);
}

private void cancleBtn12_Click(object sender, EventArgs e)
{
    deleteOrder(12);
}
#endregion

private void orderNow_Click(object sender, EventArgs e)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

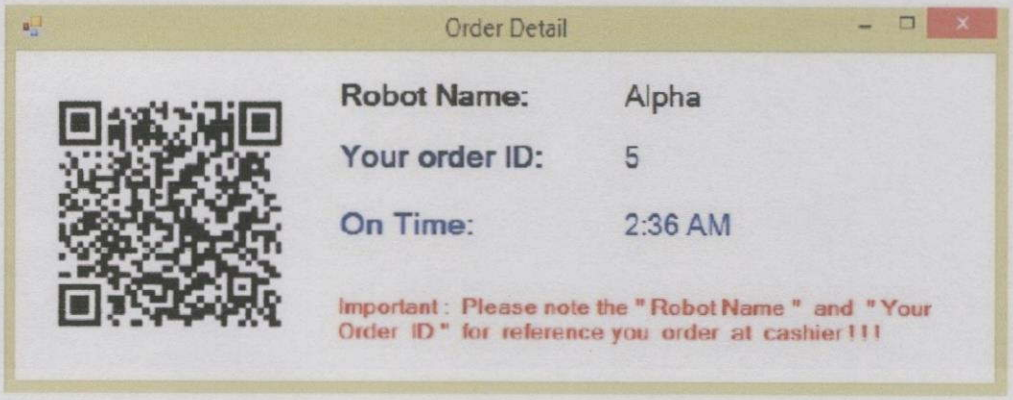
```
            DialogResult orderNowConfirm = MessageBox.Show("Do you want to order
now?", "Check out", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
            if(orderNowConfirm==DialogResult.Yes)
            {
                generateOrderID();
                int amountOfOrder = globalVar.orderBarcode.Count;
                connection.Open();
                for (int i = 1; i <= amountOfOrder;i++ )
                {
                    string addCartStr = "Insert into
order_pending(RobotID,OrderID,BarCode,Quantity)";
                    addCartStr += "values('" + dataSetting.robotName + "'," +
globalVar.orderID + ",'" + globalVar.orderBarcode[i-1] + "','" +
globalVar.orderAmount[i-1] + "');";
                    SQLCommand = new MySqlCommand(addCartStr, connection);
                    SQLCommand.ExecuteNonQuery();
                }
                connection.Close();

                clearControl();
                globalVar.orderBarcode = null;
                globalVar.orderBrand = null;
                globalVar.orderModel = null;
                globalVar.orderSize = null;
                globalVar.orderPriePU = null;
                globalVar.orderAmount = null;
                globalVar.orderTotalPrice = null;
                FileStream rebuild = File.Open(globalVar.productTxtPath,
FileMode.Truncate);
                rebuild.Close();
                if (globalVar.orderBarcode != null)
                {
                    int length = globalVar.orderBarcode.Count;
                    if (length > 0)
                    {
                        string[] rebuildProductTemp = new string[length];
                        for (int i = 1; i <= length; i++)
                        {
                            rebuildProductTemp[i - 1] = globalVar.orderBarcode[i - 1]
+ "\t" + globalVar.orderBrand[i - 1] + "\t" + globalVar.orderModel[i - 1] + "\t" +
globalVar.orderSize[i - 1] + "\t" + globalVar.orderPriePU[i - 1] + "\t" +
globalVar.orderAmount[i - 1];
                        }
                        File.WriteAllLines(globalVar.productTxtPath,
rebuildProductTemp, Encoding.UTF8);
                    }
                }
                orderDetailFrm orderDetailFrm = new
orderDetailFrm(dataSetting.robotName,Convert.ToString(globalVar.orderID),estimateTime(
));
                orderDetailFrm.Show();
                this.Close();
            }
        }

        public int estimateTime()
        {
            string loadShelveListStr = "select
order_pending.barcode,order_pending.quantity,product.shelve from order_pending left
join product on order_pending.barcode=product.barcode and order_pending.robotId ='" +
dataSetting.robotName + "' and order_pending.orderId=" + globalVar.orderID + " where
shelve is not null;";
            MySqlDataAdapter loadShelveListDA = new
MySqlDataAdapter(loadShelveListStr, connection);
            DataTable shelveListDT = new DataTable();
            loadShelveListDA.Fill(shelveListDT);
            string[] barcode = shelveListDT.AsEnumerable().Select(r =>
r.Field<string>("Barcode")).ToArray();
            int[] quantity = shelveListDT.AsEnumerable().Select(r =>
r.Field<int>("Quantity")).ToArray();
            string[] shelves = shelveListDT.AsEnumerable().Select(r =>
r.Field<string>("Shelve")).ToArray();
            string[] distinctShelve = shelves.Distinct().ToArray();
            int timeToAccess = 0;
            for(int i=1;i<=distinctShelve.Length;i++)
            {
                int lengthOfOrder = shelves.Length;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                int inSHConstTime2Acc = 15;
                int inSHTimeAcc = 0;
                for(int j=1;j<=lengthOfOrder;j++)
                {
                    if (distinctShelve[i - 1] == shelves[j - 1])
                    {
                        inSHTimeAcc = (inSHTimeAcc + ((quantity[j - 1]) *
(inSHConstTime2Acc)));

                    }
                }
                timeToAccess = (timeToAccess + inSHTimeAcc);
            }
            return timeToAccess;

        }
        private void clearAllLbl_Click(object sender, EventArgs e)
        {
            DialogResult clearResult = MessageBox.Show("Do you want to clear all
product in your cart?", "My Cart", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
            if (clearResult == DialogResult.Yes)
            {

                int orderLength = globalVar.orderBarcode.Count;
                for (int i = 1; i <= orderLength;i++ )
                {
                    returnQuantity(globalVar.orderBarcode[i - 1],
globalVar.orderAmount[i - 1]);
                }
                globalVar.orderBarcode = null;
                globalVar.orderBrand = null;
                globalVar.orderModel = null;
                globalVar.orderSize = null;
                globalVar.orderPriePU = null;
                globalVar.orderAmount = null;
                globalVar.orderTotalPrice = null;
                FileStream rebuild = File.Open(globalVar.productTxtPath,
FileMode.Truncate);
                clearControl();
                rebuild.Close();

            }
        }
        private void vScrollBar1_Scroll(object sender, ScrollEventArgs e)
        {
            showOrderList();
        }

        private void myCartFrm_Closing(object sender, FormClosingEventArgs e)
        {

            if(globalVar.orderBarcode!=null)
            {
                int length = globalVar.orderBarcode.Count;
                if(length>0)
                {
                    string[] rebuildProductTemp = new string[length];
                    for (int i = 1; i <= length; i++)
                    {
                        rebuildProductTemp[i - 1] = globalVar.orderBarcode[i - 1] +
"\t" + globalVar.orderBrand[i - 1] + "\t" + globalVar.orderModel[i - 1] + "\t" +
globalVar.orderSize[i - 1] + "\t" + globalVar.orderPriePU[i - 1] + "\t" +
globalVar.orderAmount[i - 1];
                    }
                    File.WriteAllLines(globalVar.productTxtPath, rebuildProductTemp,
Encoding.UTF8);
                }
            }
        }
    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Order Detail.cs




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using QRCoder;

namespace BranchApp
{
    public partial class orderDetailFrm : Form
    {
        BranchAppSetting dataSetting = new BranchAppSetting();
        GlobalVariables globalVar = new GlobalVariables();

        public void generateQRCode(string robotName,string orderID,string
timeToAccess)
        {
            string text2Generate = "RobotName: " + robotName + "\n" + "OrderID: " +
orderID + "\n" + "TimeToAccess: " + timeToAccess;
            QRCodeGenerator qrGenerator = new QRCodeGenerator();
            QRCodeGenerator.QRCode qrCode = qrGenerator.CreateQrCode(text2Generate,
QRCodeGenerator.ECCLevel.Q);
            QRCodePcb.BackgroundImage = qrCode.GetGraphic(20);
        }
        public orderDetailFrm(string robotName,string orderId,int timeToAccess)
        {
            InitializeComponent();
            robotNameDispLbl.Text = robotName;
            orderIDDispLbl.Text = orderId;
            onTimeLbl.Text =
Convert.ToString(DateTime.Now.AddSeconds(timeToAccess).ToShortTimeString());
            generateQRCode(robotName, orderId,
DateTime.Now.AddSeconds(timeToAccess).ToShortTimeString());
        }
    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้