แอพพลิเคชั่นเพื่อช่วยในการสั่งอาหารภายในร้าน

Food ordering application in a restaurant

กฤษฎา สุกใส

ชัยสิญจ์ อุปลวัณณา

รายงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2562

แอพพลิเคชั่นเพื่อช่วยในการสั่งอาหารภายในร้าน ปีการศึกษา 2562

แอพพลิเคชั่นเพื่อช่วยในการสั่งอาหารภายในร้าน

นายกฤษฎา สุกใส 58010043
นายชัยสิญจ์ อุปลวัณณา 58010275
รศ.ดร.บุญธีร์ เครือตราชู อาจารย์ที่ปรึกษา
ปีการศึกษา 2562

บทคัดย่อ

ในปัจจุบันมีร้านอาหารอยู่เป็นจำนวนมาก ไม่ว่าจะเป็นร้านอาหารบุฟเฟ่ต์ ร้านอาหารจานเดียว ร้านก๋วยเตี๋ยว ร้านพิซซ่า และร้านอาหารอื่นๆ ซึ่งร้านอาหารเหล่านี้ส่วนมากจะใช้พนักงานหรือการจดใส่กระดาษในการสั่งอาหาร ซึ่งบางครั้งอาจจะทำให้เกิดความล่าช้าหรือมีความผิดพลาดในการสั่งอาหาร ทางคณะผู้จัดทำได้มีความสนใจในการศึกษาวิธีการพัฒนาแอพพลิเคชั่นและต้องการพัฒนาแอพพลิเคชั่นให้สามารถแก้ปัญหาดังกล่าวและเพิ่มประสิทธิภาพให้กับร้านอาหารให้มากยิ่งขึ้น ซึ่งแอพพลิเคชั่นนี้สามารถทำงานในระบบปฏิบัติการแอนดรอยด์ โดยใช้โปรแกรม Android Studio และภาษา Kotlin ในการพัฒนาแอพพลิเคชั่น

Food ordering application in a restaurant

Mr. Kritsada	Sooksai	58010043
Mr. Chaisin	Upalawanna	58010275
Assoc.Prof.Dr.Boontee	Kruatrachue	Advisor

Academic Year 2019

ABSTRACT

Nowadays, there are many restaurants such as buffet restaurant, À la carte restaurant, one dish restaurant, noodle shop and other restaurants. Most of these restaurants use staff or taking notes to order food. Sometimes cause delays or error in food ordering. The developers are interested in studying how to develop applications and want to develop applications to solve such problems and increase efficiency for restaurants. This application can run in the Android operating system. Using the Android Studio program as an application development program and Kotlin as a programming language.

กิตติกรรมประกาศ

ปริญญานิพจน์เล่มนี้สามารถจัดทำขึ้นมาได้อย่างสำเร็จลุล่วงได้ ต้องขอขอบคุณ รศ. ดร. บุญธีร์ เครือตราชู อาจารย์ที่ปรึกษาที่คอยรับคำปรึกษาและให้คำแนะนำต่างๆ เอาใจใส่ดูแล สอบถามความคืบหน้า และให้ความช่วยเหลือในทุกๆเรื่องเป็นอย่างดี

ขอขอบคุณอาจารย์ทุกท่านได้อบรมและสั่งสอนให้วิชาความรู้ต่างๆ ทำให้ทางคณะผู้จัดทำได้นำความรู้ต่างๆ มาประยุกต์ใช้ในการจัดทำปริญญานิพนธ์เล่มนี้ รวมถึงความช่วยเหลือในด้านต่างๆ อีกมากมาย

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ทั้งในภาควิศวกรรมคอมพิวเตอร์ หรือภาควิชาอื่นๆของสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังหรือมหาลัยอื่นๆที่ได้ให้คำแนะนำต่างๆ รวมถึงขอขอบคุณห้องปฏิบัติการต่างๆของภาควิศวกรรมตคอมพิวเตอร์ที่ให้ใช้เป็นสถานที่ทำงานในการทำปริญญานิพนธ์เล่มนี้

และท้ายที่สุด ขอขอบคุณผู้ปกครองของคณะผู้จัดทำที่คอยให้ความช่วยเหลือต่างๆ รวมถึงผู้ที่มีส่วนเกี่ยวข้องที่ไม่ได้กล่าวถึงทุกท่าน

กฤษฎา สุกใส
ชัยสิญจ์ อุปลวัณณา

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตาราง **หน้า**

สารบัญรูป

รูป **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาของปัญหา

ในปัจจุบันมีร้านอาหารอยู่เป็นจำนวนมาก ไม่ว่าจะเป็นร้านอาหารบุฟเฟ่ต์ ร้านอาหารตามสั่ง ร้านอาหารจานเดียว ร้านก๋วยเตี๋ยว และร้านอาหารอื่นๆ ซึ่งร้านอาหารเหล่านี้ส่วนมากจะใช้พนักงานหรือการจดใส่กระดาษในการสั่งอาหาร ซึ่งบางครั้งอาจจะทำให้เกิดความล่าช้าหรือมีความผิดพลาดในการสั่งอาหาร

เนื่องจากทางคณะผู้จัดทำได้มีความสนใจในการศึกษาวิธีการพัฒนาแอพพลิเคชั่นและต้องการให้ออกมามีคุณภาพ ทางคณะผู้จัดทำจึงต้องการพัฒนาแอพพลิเคชั่นที่สามารถแก้ปัญหาดังกล่าวและสามารถเพิ่มประสิทธิภาพให้กับร้านอาหารให้มากยิ่งขึ้น จึงเป็นที่มาของโครงงานนี้

1.2 วัตถุประสงค์ของโครงงาน

1) เพื่อศึกษากระบวนการสร้าง ออกแบบ และพัฒนาแอพพลิเคชั่น
2) เพื่อศึกษาและฝึกการใช้งานโปรแกรมในการสร้างแอพพลิเคชั่น
3) เพื่อศึกษาการเขียนโปรแกรมบนระบบปฏิบัติการแอนดรอยด์ด้วยภาษา Kotlin
4) เพื่อพัฒนาแอพพลิเคชั่นที่สามารถนำไปใช้งานได้ในร้านอาหาร

1.3 ขอบเขตของโครงงาน

1) แอพพลิเคชั่นสามารถทำงานบนระบบปฏิบัติการแอนดรอยด์
2) ผู้ใช้งานที่เป็นลูกค้าสามารถสั่งอาหารผ่านแอพพลิเคชั่นภายในร้านได้
3) ผู้ใช้งานที่เป็นลูกค้าสามารถดูได้ว่าตนเองสั่งอาหารอะไรบ้าง
4) ผู้ใช้งานที่เป็นลูกค้าสามารถทราบคิวของตนเองผ่านแอพพลิเคชั่นภายในร้านได้
5) ผู้ใช้งานที่เป็นลูกค้าสามารถทราบสถานะของอาหารที่สั่งแล้วได้ ว่าอยู่ในขั้นตอนการรอหรือกำลังจัดเตรียม
6) ผู้ใช้งานที่เป็นลูกค้าสามารถยกเลิกเมนูที่สั่งไว้นานแล้วแต่ยังไม่ได้รับได้
7) ผู้ใช้งานที่เป็นลูกค้าสามารถเรียกพนักงานเพื่อชำระเงินผ่านแอพพลิเคชั่นภายในร้านได้
8) ผู้ใช้งานที่เป็นพนักงานสามารถทราบได้ว่าลูกค้าโต๊ะใด สั่งอะไรบ้างผ่านแอพพลิเคชั่น
9) ผู้ใช้งานที่เป็นพนักงานสามารถทราบได้ว่าลูกค้าโต๊ะใด สั่งอาหารก่อนหรือหลังผ่านแอพพลิเคชั่น
10) ผู้ใช้งานที่เป็นพนักงานสามารถทราบได้ว่าลูกค้าโต๊ะใด กำลังเรียกชำระเงินบ้าง
11) ผู้ใช้งานที่เป็นพนักงานสามารถเพิ่มเมนูใหม่ๆ หรือแก้ไขเมนูที่มีอยู่ได้ โดยไม่ต้องพึ่งพาผู้พัฒนา

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) ช่วยให้พนักงานบริหารจัดการเมนูและคิวต่างๆได้อย่างมีประสิทธิภาพ
2) ช่วยให้ทางร้านลดข้อผิดพลาดเรื่องการจดรายการอาหารผิด และช่วยให้จัดการอาหารได้ง่ายยิ่งขึ้น
3) ลูกค้าสามารถเห็นภาพอาหารและราคาก่อนซื้อ ช่วยในการตัดสินใจได้ง่ายขึ้น
4) ช่วยให้ลูกค้าสามารถสั่งอาหารได้อย่างสะดวกสบายมากยิ่งขึ้น

1.5 แผนการดำเนินงาน

ตาราง 1.1 แผนการดำเนินงาน

งานที่ทำ	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.
กำหนดหัวข้อและขอบเขตของโครงงาน									
ศึกษาการใช้งาน Android Studio และภาษา Kotlin									
ออกแบบแอพพลิเคชั่น									
ทดลองเขียนแอพพลิเคชั่นตามที่ได้ออกแบบไว้									
ทำเล่มรายงาน									
พัฒนาแอพพลิเคชั่นตามที่ออกแบบไว้									
ปรับปรุง-แก้ไขแอพพลิเคชั่นตามความเหมาะสม									

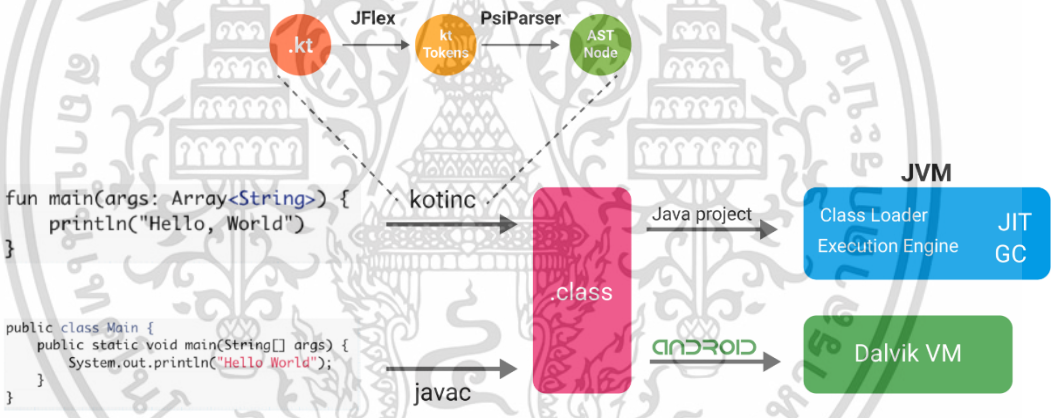
บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ภาษา Kotlin

Kotlin เป็นภาษาใหม่พัฒนาขึ้นมาเพื่อใช้แทนภาษา Java ที่มีจุดอ่อนอยู่หลายอย่าง ที่ไม่สามารถแก้ไขอะไรได้มากนัก ซึ่งหลักๆจะเป็นเรื่อง backward compatibility JAVA 6 7 8 และ feature อีกหลายๆอย่างที่ JAVA ไม่มีเช่น null safety, Extension function เป็นต้น

แนวคิดของ Kotlin คือเข้ากันได้ 100% กับ Java เพื่อให้สามารถใช้ประโยชน์จาก library API และ tools จำนวนมากที่มีอยู่แล้วในโลกของ JAVA และใส่ feature ที่ JAVA ไม่มีเข้าไปอีกด้วย

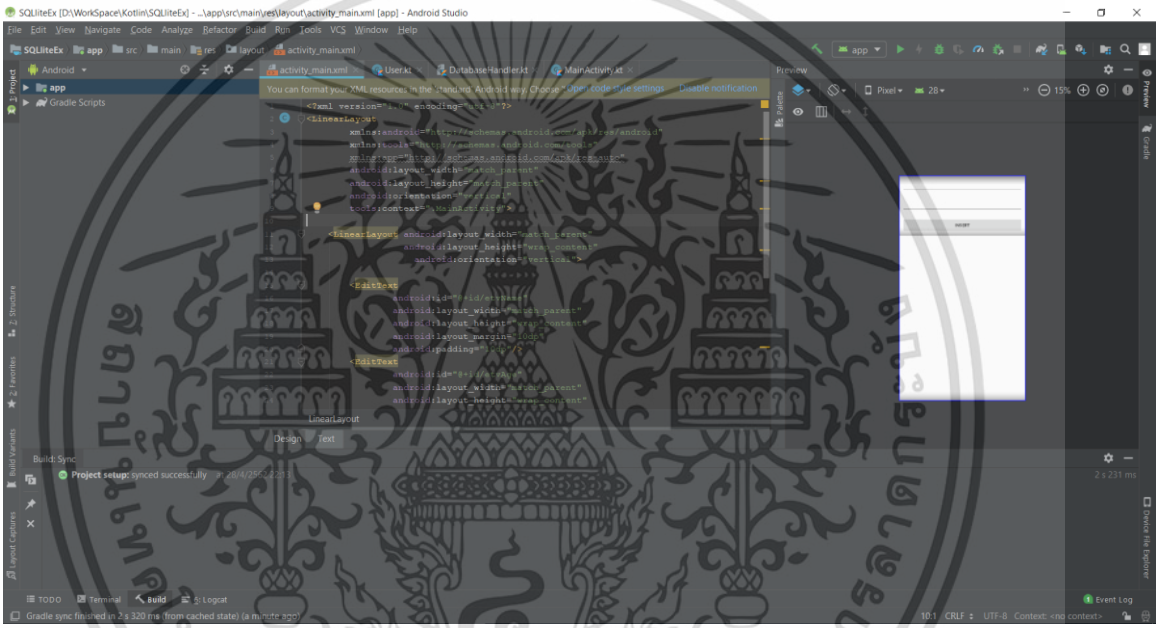
เป้าหมายของ Kotlin คือนำมาใช้แทน JAVA โดยยังรักษาโค้ด JAVA เพื่อให้ทำงานร่วมกับโปรเจคเก่าที่เขียนไปแล้วได้



รูป 2.1 ภาษา Kotlin เทียบกับภาษา Java

2.2 Android Studio

Android Studio เป็น IDE Tool จาก Google ไว้พัฒนา Android โดยเฉพาะ โดยพัฒนาจากแนวคิดพื้นฐานมาจาก InteliJ IDEA คล้าย ๆ กับการทำงานของ Eclipse และ Android ADT Plugin โดยวัตถุประสงค์ของ Android Studio คือต้องการพัฒนาเครื่องมือ IDE ที่สามารถพัฒนา App บน Android ให้มีประสิทธิภาพมากขึ้น ทั้งด้านการออกแบบ GUI ที่ช่วยให้สามารถ Preview ตัว App มุมมองที่แตกต่างกันบน Smart Phone แต่ล่ะรุ่น สามารถแสดงผลบางอย่างได้ทันทีโดนไม่ต้องทำการรัน App บน Emulator รวมทั้งยังแก้ไขปรับปรุงในเรื่องของความเร็วของ Emulator ที่ยังเจอปัญหากันอยู่ในปัจจุบัน



รูป 2.2 หน้าตาโปรแกรม Android Studio

2.3 Android Architecture Components

เป็นชุดคำสั่งหรือตัวช่วยในการจัดการโครงสร้างการเขียนแอพพลิเคชั่นบนระบบปฏิบัติการแอนดรอยด์ ภาษา Java หรือ Kotlin โดยจะมีประโยชน์คือทำให้การเขียนโครงสร้างแบบนี้สามารถแยกทดสอบได้ไว และเร็วยิ่งขึ้น เนื่องจากแยกส่วนต่างๆเป็น atomic ได้มากขึ้น โดยจะมี Library ที่น่าสนใจคือ

2.3.1 Room

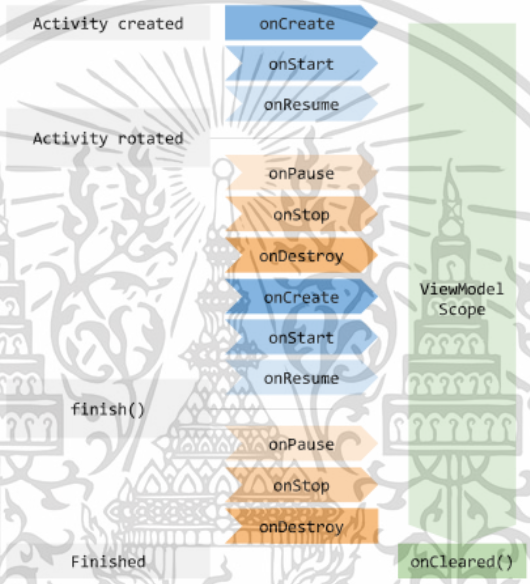
เป็น Library ที่ช่วยในการจัดการ Abstraction Layer บน SQLite ซึ่งเพิ่มประสิทธิ์ภาพในการจัดการฐานข้อมูล

2.3.2 ViewModel

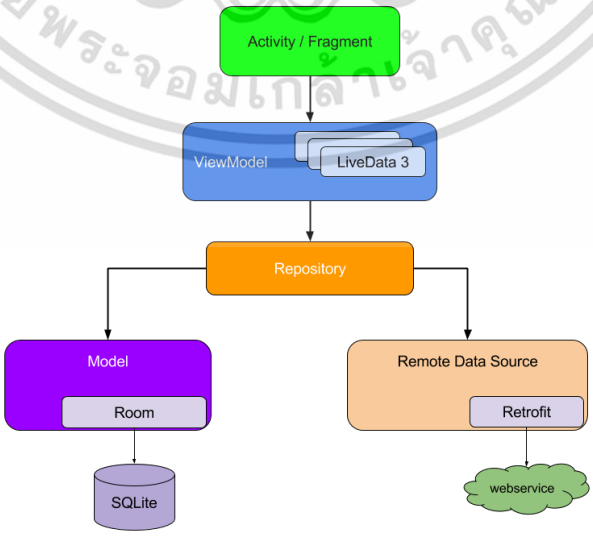
ViewModel เป็น component ใน AAC ที่นอกจากจะช่วยให้จัดโค้ดที่ไม่เกี่ยวกับการแสดงผล UI ออกมาจาก Activity หรือ Fragment แล้ว มันยังถูกออกแบบมาให้แก้ปัญหาวิธีการจัดการ Configuration Changes ใน Activity อย่างเช่นเวลาจอหมุนแล้วทำให้ Activity ถูกสร้างใหม่

2.3.3 Livedata

คือ Data Holder ที่ทำงานร่วมกับ Lifecycle ของ Activity/Fragment/Service และมีลักษณะการทำงานเป็น Observer ที่คอยสังเกตการณ์การเปลี่ยนแปลงของข้อมูล



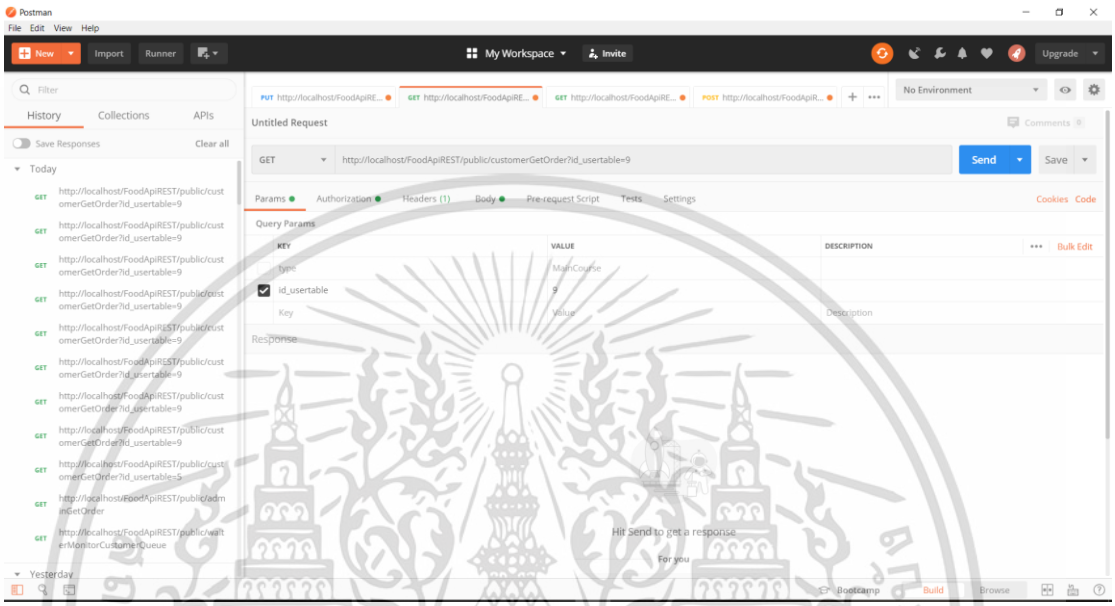
รูป 2.3 Viewmodel Lifecycle เมื่อเทียบกับ Lifecycle ของ Activity หรือ Fragment



รูป 2.4 Android Architecture Components

2.4 Postman

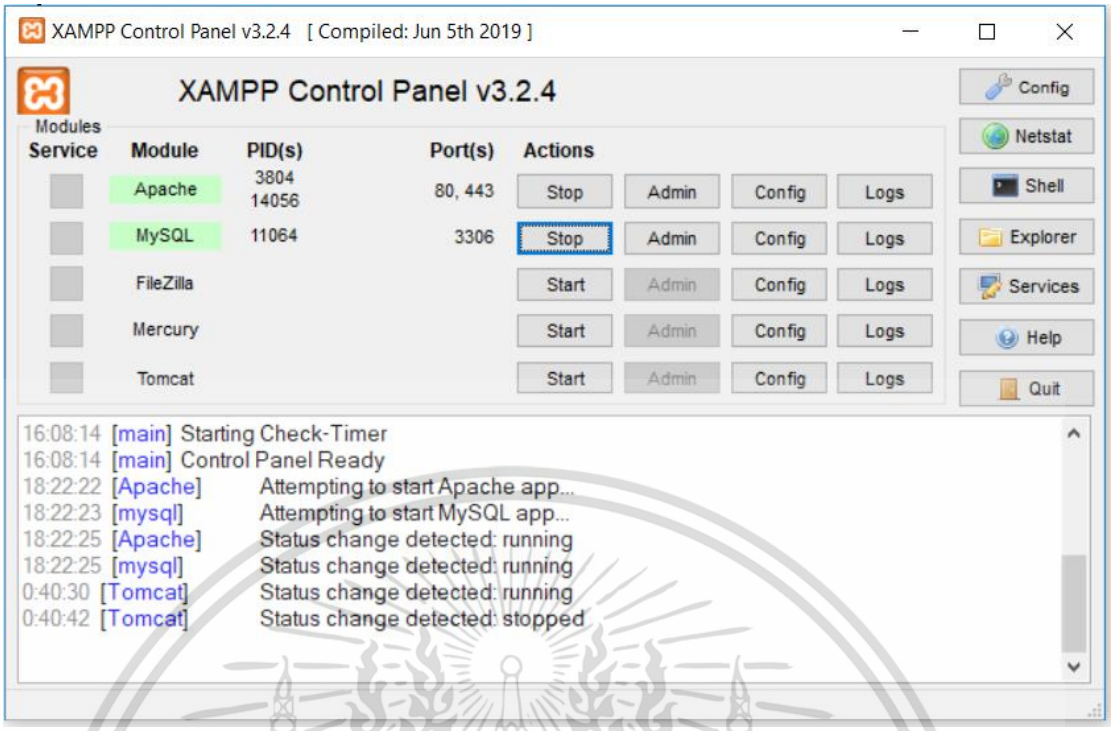
Postman คือเครื่องมือสำหรับช่วยในการพัฒนา API ทดสอบการทำงานของ Service เป็นโปรแกรมที่ใช้งานง่าย มี UI ที่สวยงาม โดยผู้ใช้งานไม่จำเป็นต้องมีความรู้เรื่องภาษาโปรแกรมมิ่งก็สามารถใช้งานได้



รูป 2.5 หน้าตาโปรแกรม Postman

2.5 XAMPP

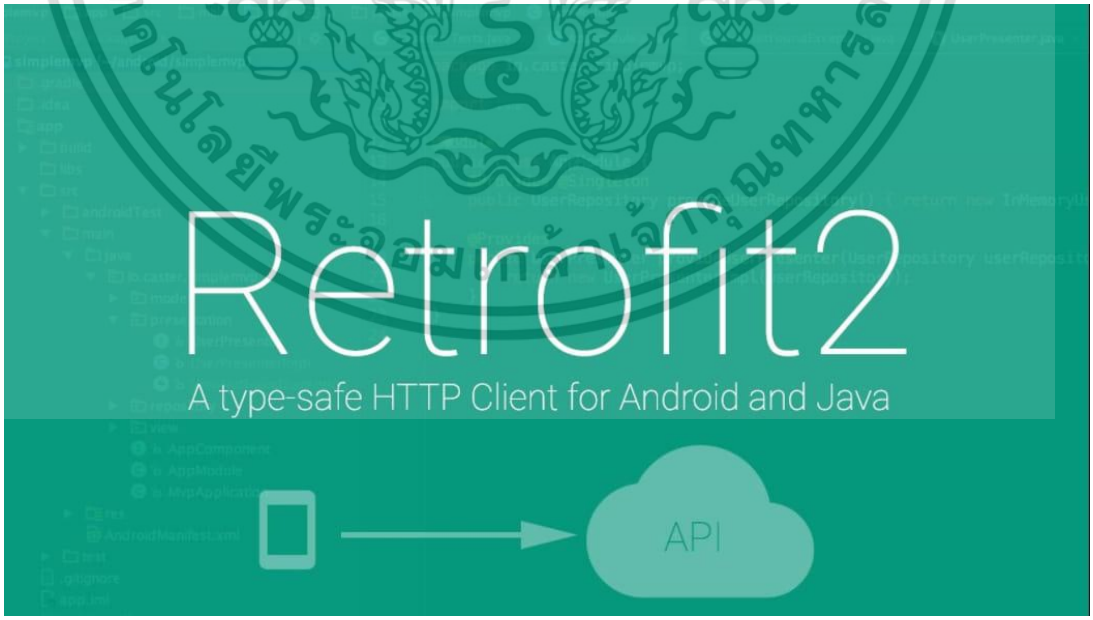
เป็นโปรแกรม Apache web server ไว้จำลอง web server เพื่อไว้ทดสอบ สคริปหรือเว็บไซต์ในเครื่องของตนเอง โดยที่ไม่ต้องเชื่อมต่ออินเตอร์เน็ตและไม่ต้องมีค่าใช้จ่ายใดๆ ง่ายต่อการติดตั้งและใช้งานโปรแกรม XAMPP จะมาพร้อมกับ PHP ภาษาสำหรับพัฒนาเว็บแอพลิเคชั่นที่เป็นที่นิยม, MySQL ฐานข้อมูล, Apache จะทำหน้าที่เป็นเว็บเซิร์ฟเวอร์, Perl อีกทั้งยังมาพร้อมกับ OpenSSL, phpMyadmin ระบบบริหารฐานข้อมูลที่พัฒนาโดย PHP เพื่อใช้เชื่อมต่อไปยังฐานข้อมูล สนับสนุนฐานข้อมูล MySQL และ SQLite



รูป 2.6 หน้าตาโปรแกรม Xampp

2.6 Retrofit2

Retrofit เป็น HTTP Client Library ที่ใช้ติดต่อกับฝั่ง Server ที่ได้รับความนิยมอันดับต้นๆใน แอนดรอยด์จากความเรียบง่ายในการใช้และมีประสิทธิภาพ



รูป 2.7 Retrofit Library

2.7 ภาษา PHP

เป็นภาษา Server-Side Script ซึ่งใช้ในการจัดทำเว็บไซต์และสามารถประมวลผลออกมาในรูปแบบ HTML โดยมีรากฐานโครงสร้างคำสั่งมาจากภาษา ภาษาซี ภาษาจาวา และ ภาษาเพิร์ล เป้าหมายหลักของภาษา PHP คือให้นักพัฒนาเว็บไซต์สามารถเขียนเว็บเพจ ที่มีการตอบโต้ได้อย่างรวดเร็ว มีข้อดีคือเรียนรู้ได้ง่าย และสามารถทำงานร่วมกับฐานข้อมูลได้หลายชนิด

2.8 MySQL

MySQL (มายเอสคิวแอล) เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System) แบบซอฟต์แวร์โอเพนซอร์ส สามารถใช้ร่วมกันได้ดีกับ Xampp และภาษา PHP

รูป 2.8 โลโก้ MySQL

2.9 REST API

REST หรือ Representational State Transfer มันเป็นวิธีในการสร้าง Web Service รูปแบบนึงที่อาศัย HTTP Method (GET, POST, PUT, DELETE) ในการทำงาน และส่งผลกลับมาในรูปแบบของ JSON หรือ XML ส่งผลให้สามารถรับส่งข้อมูลไปมาข้าม Platform ได้อย่างสะดวก

2.10 ขั้นตอนในการพัฒนาแอพพลิเคชั่น

ในการพัฒนาโปรแกรมหรือพัฒนาแอพพลิเคชั่น ควรมีการพัฒนาอย่างเป็นระบบแบบแผน ซึ่งจะช่วยให้นักพัฒนาสามารถเห็นกรอบและแนวทางในการทำงาน มีขั้นตอนดังนี้

2.10.1 วางแผน

การสร้างโปรแกรมเริ่มจากไอเดีย ต้องรู้ว่าจะทำอะไร จากนั้นค่อยๆ กระจายไอเดียนั้นว่าควรจะมีฟีเจอร์หรือระบบอะไรบ้าง

2.10.2.1 ออกแบบ UX

ออกแบบโครงสร้างข้อมูล เป็นขั้นตอนที่ตัดสินใจว่าแอพต้องแสดงข้อมูลอะไร ทำงานอะไรได้ ฟีเจอร์อะไรบ้าง และเมื่อต้องแสดงข้อมูล จะแสดงที่หน้าไหนของแอพ จากนั้นสร้างเป็น wireframe

2.10.2.1.1 Wireframe

ขั้นต่อไป เริ่มสร้างแต่ละ screen กำหนดการทำงาน กำหนดข้อมูลที่จะแสดง และกำหนดที่อยู่ข้อมูลที่จะแสดง เมื่อมี screen ครบแล้ว ถัดไปสร้างแอพ workflow

2.10.2.1.2 Workflow

Workflow คือเส้นทางที่ผู้ใช้งานสามารถท่องไปในแอพ แต่ละอย่างที่ผู้ใช้งานทำจะเกิดผลอะไรบ้าง และผู้ใช้งานจะไปยัง screen ที่ต้องการได้อย่างไร มีขั้นตอนอะไรบ้าง โดยควรจะลดความซับซ้อนให้ได้มากที่สุด ไม่ใช้จำนวนการกดมากเกินไป หากเกิดปัญหาใน workflow กลับไปแก้ไข wireframe และก็ลองใหม่ด้วยการทดสอบทุก flow ตั้งแต่ต้น เพื่อให้มั่นใจว่าการแก้ flow นี้ให้ง่ายขึ้น ไม่ได้ทำให้อีก flow ยากขึ้นแทน

2.10.2.1.3 Click-through model

Click-through model ช่วยในการทดสอบ wireframe และ workflow โดยการให้ผู้ใช้งานได้ทดลองเหมือนใช้งานแอพจริง ในขั้นตอนนี้จะเป็นการทดลองกดเพื่อเปลี่ยนภาพ screen และทดสอบ navigation ของแอพเท่านั้น

2.10.2.2 ออกแบบ UI

2.10.2.2.1 Style guide

Style guide หรือ UI Kit เป็นเหมือนต้นแบบของสิ่งต่างๆในแอพ การมี style ที่ชัดเจนจะช่วยให้ผู้ใช้งานไม่เกิดความสับสนกับการใช้งานแอพ การมี style ที่ชัดเจนและตรงกันทั้งแอพจะทำให้ผู้ใช้งานใช้งานได้ลื่นขึ้น

การกำหนด Style guide ต้องรู้ว่าผู้ใช้งานเป็นใคร แอพจะถูกใช้งานเวลาใด หากใช้กลางคืนควรออกแบบให้มืด ถ้าถูกใช้โดยพนักงานที่งานต้องการความรวดเร็ว ก็ควรลดขั้นตอนให้ง่ายขึ้น และทำให้งานสำเร็จได้โดยไว

2.10.2.2.2 Rendered design

เป็นการเปลี่ยน wireframe สีขาวดำให้กลายเป็นหน้าตาแอพจริง โดยอิงจาก Style guide หากมีจุดไหนต้องอัปเดตหรือเพิ่ม Style guide ก็สามารถกลับไปอัปเดตได้ แต่ต้องให้มั่นใจว่าผลลัพธ์ออกมามีความสอดคล้องกันทั้งแอพ

2.10.2.2.3 Rendered click-through model

หลังได้หน้าตาแอพจริงทั้งหมดมาแล้ว จะทำการ click-through model อีกรอบ ในขั้นตอนนี้อาจจะต้องใช้เวลามาก เพื่อให้มั่นใจว่าจะไม่มีการแก้ไขใหญ่ๆอีกแล้ว เพราะหลังจากขั้นตอนนี้ การเปลี่ยนอะไรบางอย่างจะมีราคาแพงและใช้เวลานานขึ้น

การส่งไม้ต่อจาก Design ไปยัง Development

หลังจากการออกแบบว่าแอพจะมีหน้าตายังไงและทำอะไรได้บ้าง เป็นเรื่องจำเป็นอย่างมากที่จะทำให้ dev team เข้าใจใน vision ตรงกัน เพื่อที่แอพจะได้เป็นไปตามการออกแบบมากที่สุด

2.10.2.3 เขียน Code

2.10.2.3.1 เลือก Tech stack

มีหลายวิธีและ technology ที่ใช้ในการพัฒนาแอพได้ แต่ละวิธีก็มีข้อดีข้อเสียต่างกัน บางอย่างอาจจะราคาถูก แต่ประสิทธิภาพของแอพตก บางอย่างอาจจะซับซ้อน และใช้เวลานานเกินไปสำหรับแอพที่จะทำ และไม่ควรเลือกใช้ technology ที่ขาดการอัปเดต หรือไม่เสถียร

ฝั่ง Front-end

ในการทำ mobile app มี 3 ทางเลือก

1. Native เป็นการสร้างแอพแยก platform จึงไม่สามารถ reuse code ระหว่างกันได้ วิธีนี้ทำให้ปรับแต่งได้สูงสุด UI เป็นของ platform 100% แอพเร็วและมีประสิทธิภาพสูง เป็นวิธีที่ค่าใช้จ่ายสูงที่สุด เพราะต้องทำแต่ละ platform ใหม่ทั้งหมด

2. Cross-platform เป็นเทคโนโลยีที่มี code บางส่วนหรือทั้งหมดแชร์กัน แต่ก็ยัง build ไปรันเป็น Native เช่น React Native, Xamarin อันนี้เป็นวิธีที่ประหยัดเงินและเวลา ถ้าไม่ต้องการ performance สูงสุดขนาดแบบแรก

3. Hybrid เป็นวิธีที่ใช้เทคโนโลยีของ web (HTML, CSS, Javascript) และ install ผ่าน native wrapper เช่น Cordova, Phone Gap, Ionic วิธีนี้อาจจะราคาถูกสุด แต่แอพออกมาคุณภาพค่อนข้างต่ำ และช้ามาก

ฝั่ง Back-end (Web API & Server)

Server มีผลอย่างมากกับประสิทธิภาพของแอพ และการขยายจำนวนผู้ใช้ที่แอพรองรับได้ มีสิ่งที่ต้องตัดสินใจก่อนเริ่มเขียน code ดังนี้

1. ภาษา Java, C#, Go-lang, Javascript, PHP, Python, Ruby แต่ละภาษาจะมี framework ที่ช่วยในการเขียนของตัวเอง

2. Database มี 2 ประเภทหลักๆ SQL และ noSQL SQL เป็นของดั้งเดิมที่มักจะเป็นทางเลือกที่ดีที่สุดในแทบทุกกรณี มีหลากหลายให้เลือก เช่น MSSQL, MySQL, PostgreSQL เป็นต้น noSQL มักใช้ในสถานการณ์ที่ต้องการเก็บข้อมูลปริมาณมหาศาลหรือไม่มีการแก้ไขข้อมูลเดิมบ่อยๆ นอกจากการเลือก database แล้วยังมีเรื่องของการออกแบบ schema ว่าจะเก็บข้อมูลยังไง การออกแบบที่ดีจะส่งผลต่อความสำเร็จในระยะยาว ดังนั้นในขั้นตอนนี้ต้องมั่นใจว่า database ได้ออกแบบไว้อย่างดี และรองรับอนาคต

3. Hosting environment (Infrastructure) เลือกว่าจะ host server ไว้ที่ไหน ตัดสินใจจากราคา ความสามารถในการขยายสเกล ความสามารถที่ทำได้ บริการเสริม การใช้งานง่าย ประสิทธิภาพ และความสเถียร มีหลายเจ้า เช่น AWS, Google Cloud, Heroku, Azure นอกเหนือจากนั้น ยังต้องวางแผนเผื่อการสเกล เมื่อจำนวนผู้ใช้เพิ่มมากขึ้น Cloud-base ทำให้สามารถจ่ายเงินเพื่อเพิ่มหรือลดได้ตามความจำเป็น นอกจากนี้ยังมีบริการเสริม เช่น backup database, server uptime, update os และอีกมากมาย

2.10.2.3.2 Agile development process

คือการแตกย่อยงานที่ต้องพัฒนาทั้งหมดออกมาเป็น milestone ที่เล็กลง และเริ่มพัฒนาแอพเป็นรอบๆ วนลูปไปเรื่อยๆ ในแต่ละรอบจะมีการวางแผน, การพัฒนา, การเทส, และการรีวิว ซึ่งจะมีขั้นตอนต่างๆดังนี้

การวางแผน

การวางแผนคือการแบ่งงานออกเป็นลิสต์ของ task ที่จะ code ในรอบ iteration นี้ แต่ละ task จะต้องมี requirement ที่ชัดเจน หลังจากที่ developer เข้าใจ requirement ดีแล้ว developer ก็จะประเมินเวลาที่ต้องใช้ในการทำแต่ละ task เพื่อที่จะกระจายงานกันไปทำได้อย่างสมดุลกับปริมาณงานที่ทีมทำได้ในแต่ละ sprint

developer จะเริ่มวางแผนว่าจะเขียน solution ยังไงเพื่อมาแก้ปัญหาในแต่ละ task developer เก่งๆจะหาทาง reuse code ทั้งแอพให้มากที่สุด เช่นพวก style หรือ function ที่ใช้ร่วมกันได้ ถ้าต้องเปลี่ยน design ก็แค่ไปอัปเดตที่เดียวก็จะเปลี่ยนทั้งหมด ไม่ต้องไปนั่งไล่อัปเดตทุกๆที่

การพัฒนา

developer จะเริ่ม code ตาม requirement ใน task ที่ได้รับมอบหมาย developer ต้องเข้าใจเป้าหมายสูงสุดของแอพ และเจตนาของแต่ละ task ถ้าบางอย่างมันทำไปแล้วดูไม่ถูกต้อง developer ต้องพูดคุยเพื่อหาทางออกกันทันที เมื่อ task นั้นๆ เสร็จก็จะ deploy ขึ้น development version ให้ tester ทดสอบได้

การทดสอบ

การทดสอบควรทำโดยคนที่ไม่ใช่ developer ที่ code แอพนี้ขึ้นมา เพราะ developer จะรู้อยู่แล้วว่าตรงไหนทำอะไรได้ บางทีก็จะไม่เจอสิ่งที่ผู้ใช้งานที่ใช้งานจริงจะเจอเมื่อใช้งานทั่วไป การทดสอบมีหลายประเภทในแต่ละความคืบหน้าของการพัฒนา

Functional Test การทดสอบว่าฟีเจอร์ นี้ทำงานได้ถูกต้องตาม requirement หรือไม่ ทีม QA จะมี test case, action step และผลลัพธ์ที่คาดหวังว่าจะให้มันเกิด

Usability Testing ทดสอบว่าผู้ใช้ใช้งานได้หรือไม่ และใช้งานง่ายพอมั้ย การทดสอบ ควรเลือกคนที่เคยเห็นแอพเป็นครั้งแรกมาทดสอบ เพื่อจำลองว่าคนที่เพิ่งเริ่มใช้จริงๆ โดยกำหนด target group เพื่อนำมาทดสอบ หลังจากทดสอบเสร็จให้สอบถาม feedback และนำมาปรับปรุงแอพต่อไป

Performance Testing ถ้าแอพใช้เวลา 20 วิในการเปิด ต่อให้ทำงานถูกต้องก็คงไม่มีใครใช้ Performance Testing ต้องทำก่อนปล่อยให้ผู้ใช้งานจริงใช้ แต่ถ้าทดสอบเจอตั้งแต่แรกๆ ก็อาจจะทำให้แก้ไขได้ง่ายกว่าไปแก้ตอนท้าย

Regression Testing ทดสอบฟีเจอร์ ที่เคยทำเสร็จและทดสอบผ่านไปใน sprint ก่อนๆ เพราะการทำงานใน sprint นี้อาจส่งผลกระทบกับฟีเจอร์ เก่าทำให้ทำงานผิดพลาดได้ tester ที่ดีควรจะมี list ของ test case เพื่อมาทดสอบของ sprint ที่ผ่านไปแล้วด้วย

Device-Specific Testing ทดสอบบนหลายๆ screen size และ OS version หรือ browser มีหลาย tool ที่ช่วยจำลองเครื่องหลายๆรุ่นได้ แต่ก็ต้องเทสบนเครื่องจริงจำนวนนึง เพื่อให้มั่นใจว่ามันทำงานได้บนเครื่องส่วนใหญ่

User Acceptance Testing ให้ผู้ใช้งานจริงๆ ทดสอบ และเก็บ feedback จริง เมื่อเจอบัคก็ต้องสร้าง task ให้ developer ไปแก้ไขและปิด issue task นี้

Sprint รีวิว/retrospective

ตอนท้ายของทุก sprint จะมีประชุมเพื่อคุยกับทุกคนที่เกี่ยวข้องว่า sprint ที่ผ่านมาเป็นยังไงบ้าง อะไรดี อะไรไม่ดี ถ้ามีปัญหาอะไรก็จะได้พยายามแก้ไขไม่ให้เกิดอีกใน sprint หน้า ถ้าบางอย่างมันดีในส่วนไหน ก็นำมาใช้กับส่วนอื่นๆ พอจบ sprint review ก็จะต้องวนกลับไปขั้นตอนการวางแผน และทำวนไปเรื่อยๆ จนแอพเสร็จ

Beta testing

เมื่อแอพเสร็จเรียบร้อย อาจจะทำ beta launch อีกรอบ beta launch คือการให้ผู้ใช้งานกลุ่มเล็กๆ ใช้งานจริงในสถานการณ์จริงเหมือนกับแอพ launch ไปแล้ว ในขั้นตอนนี้จะได้ข้อมูลที่หลากหลายขึ้นอย่างมาก เพราะอาจเจอปัญหาที่ไม่เคยทดสอบเจอมาก่อน ซึ่งดีกว่าปล่อยแอพและทำการตลาดแล้วเพิ่งมาเจอปัญหาทีหลัง

หลังจาก Beta testing แล้วแก้ปัญหาที่พบไปจนหมด และไม่มีปัญหาใหม่ๆ รายงานเพิ่ม จึงเริ่มขั้นตอนต่อไป

2.10.2.4 ปล่อยแอพ

แอพทั่วไปจะมี 2 ส่วนที่สำคัญที่ต้อง deploy ก็คือ web server กับ client ที่ไว้บน Google Play หรือ Apple Store

Web API (Server)

ส่วนนี้คือการรับส่งและเก็บข้อมูลของ mobile app ถ้า server โดนใช้งานหนักเกินไป หรือล่ม แอพก็ทำงานไม่ได้ไปด้วย server ต้องสามารถ scale ได้เมื่อเกิดเหตุการณ์ไม่คาดฝันหรือแอพเกิดเป็นที่นิยมขึ้นมา เทคโนโลยีฝั่ง server ก็มีหลากหลาย ตั้งแต่ Infra ไปจนถึง programming ยกตัวอย่างเช่น AWS, Google Cloud, Kubernetes, Docker, Node.js, RoR, etc.

App Store

การอัปแอพขึ้น store ต้องมั่นใจว่าแอพได้ config ถูกต้องสำหรับ production มีขั้นตอนหลายขั้นตอนในการทำ มีฟอร์มหลายอันที่ต้องกรอก ต้องทำ screenshot และเนื้อหาสำหรับ marketing ต้องเขียนคำอธิบาย จากนั้น submit ขึ้นไป

2.10.2.5 สังเกตการณ์ และวัดผล

เมื่อ submit app ขึ้นไปแล้ว จะเห็นได้ว่าแอพดังๆทุกแอพมีลิสต์ของอัปเดต ไม่ว่าจะเป็น แก้บัค, เปลี่ยน, เพิ่มฟีเจอร์ การสังเกตการณ์สำคัญมากๆ เพื่อจะได้รู้ว่าต้องอัปเดตอะไร

Crashes

มีหลายเครื่องมือที่ track app crash ได้ Sentry, HockeyApp, Rollbar, Fabric ฯลฯ สามารถดูได้ด้วยว่าก่อน crash ผู้ใช้งานทำอะไรอยู่หน้าไหน เครื่องรุ่นอะไร และข้อมูลแวดล้อมอื่นๆ และสามารถแจ้งเตือนแบบ realtime ผ่าน email/sms ได้ทันทีที่ crash

Analytics

analytics ใหม่ๆสามารถ track ข้อมูลได้แทบทุกอย่างของผู้ใช้งานทำให้คุณเข้าใจว่าผู้ใช้งานคือใคร อายุ เพศ สถานที่ ภาษา และใช้งานแอพยังไง เวลาที่เล่น เวลาที่ใช้ในแอพ หน้าที่ดู และไปจนถึงพฤติกรรมเช่น heatmap หรือปุ่มไหนโดนคลิกเยอะสุด ใช้ข้อมูลตรงนี้ทำความเข้าใจผู้ใช้งานให้มากที่สุด และตรวจสอบตรงไหนควรลงทุนลงแรงในอนาคต แต่ลงในส่วนที่มีความเป็นไปได้ว่าจะเกิดการเติบโตสูงที่สุด tool ที่ใช้ Facebook Analytics, Apptentive, Google Analytics, Appsee

Performance

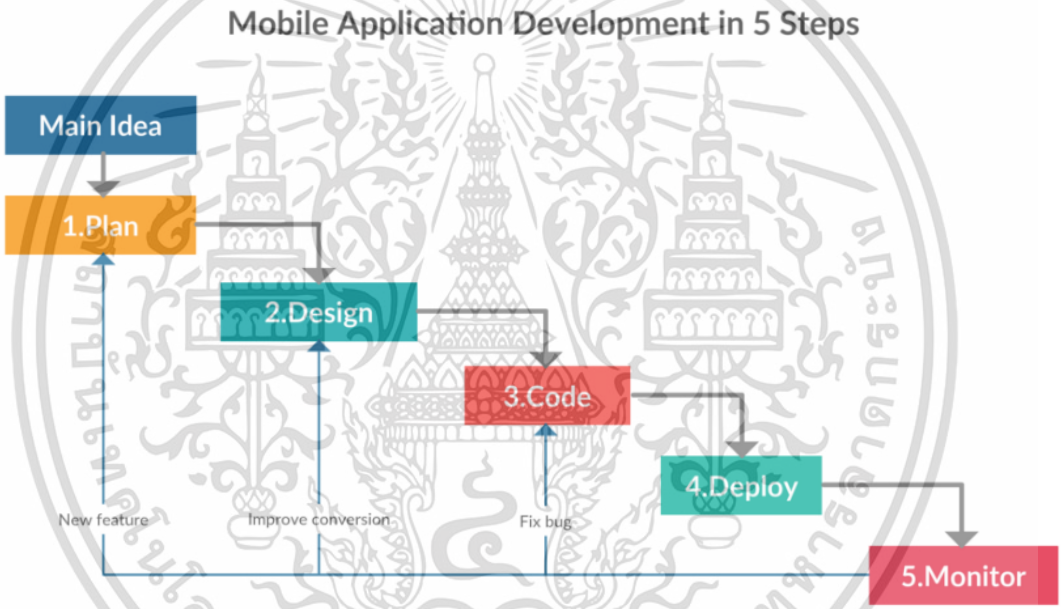
แอพเปิดได้ไวแค่ไหน ทำงานได้ไวแค่ไหน จะดูได้ว่า action ไหนใช้เวลาเท่าไหร่ แล้วจะเจอจุดที่ต้องปรับปรุงที่นั่น สามารถตั้งให้ alert เมื่อเกิดเหตุการณ์ที่บาง action ใช้เวลานานผิดปกติ จะได้เข้าไปดูว่าเกิดปัญหาอะไรขึ้นหรือไม่ tool ที่ใช้ Prometheus, New Relic

App Store Management

Rating และ Review ใน Store เป็นสิ่งสำคัญมากๆ ทุกครั้งที่มี review ใหม่ ควรตรวจสอบคำรีวิว เพราะหากมีบริการที่ดีก็ทำให้ชื่อเสียงดีตามไปด้วย

Further Iteration and Improvement

เป้าหมายของการสังเกตการณ์คือ จะได้รู้ว่าควรทำอะไรเป็นสิ่งต่อไป เพราะถ้าเปิดให้คนใช้ และดูแลแอพนี้ไปสักพัก ก็จะต้องมีฟีเจอร์ ใหม่ๆเพิ่มเข้ามาเพื่อปรับปรุงให้ดีขึ้นไปเรื่อยๆ โดยใช้ข้อมูลจากผู้ใช้ และการ monitor แล้วเอาไปเข้า development process loop เพิ่ม conversion rate, install number, และรายได้ไปเรื่อยๆ



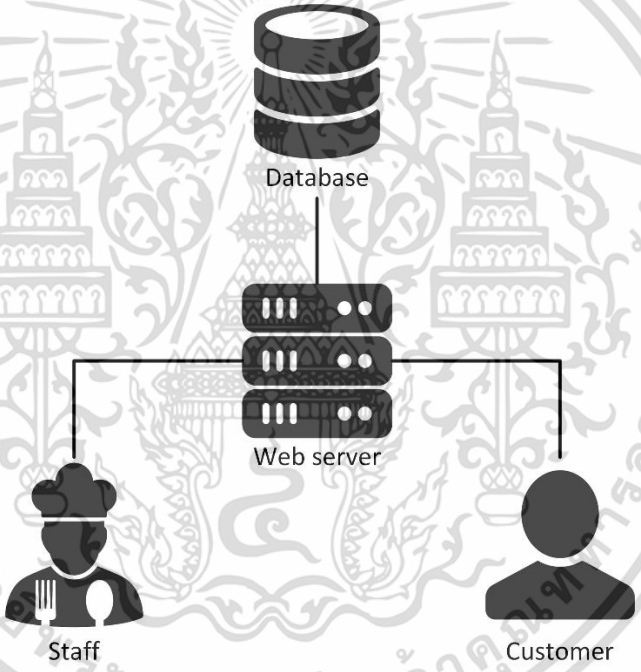
รูป 2.8 ขั้นตอนในการพัฒนาแอพพลิเคชั่น

บทที่ 3
การออกแบบและการพัฒนา

3.1 ระบบของแอพพลิเคชั่น

3.1.1 ภาพรวมของระบบ

แอพพลิเคชั่นออกแบบมาเพื่อให้ใช้งานภายในร้านอาหาร เริ่มต้นจะให้พนักงานล็อคอินเข้าใช้งานแท็บเล็ตของลูกค้าแล้วไปวางประจำบนโต๊ะเพื่อให้ลูกค้าสั่งอาหารผ่านแท็บเล็ตที่เตรียมไว้ให้ ข้อมูลทุกอย่างจะถูกส่งผ่าน Local Web server ภายในร้าน เมื่อลูกค้าสั่งอาหารพนักงานก็จะทราบได้ผ่านแอพพลิเคชั่น



รูป 3.1 ภาพรวมของระบบ

3.1.2 ระบบภายในแอพพลิเคชั่น

แอพพลิเคชั่นนี้จะแบ่งออกเป็น 2 ส่วน คือส่วนของพนักงาน และส่วนของลูกค้า ทั้ง 2 ส่วนนี้จะทำงานร่วมกัน โดยจะมีรายละเอียดดังนี้

3.1.2.1 ส่วนของลูกค้า (Customer)

ลูกค้าจะสามารถเห็นเมนูอาหารต่างๆ และราคาได้อย่างชัดเจน เริ่มแรกจะต้องเลือกเมนูที่จะสั่ง (Order) ก่อน เมนูที่สั่งแล้วจะปรากฏในรายการอาหาร (List) ซึ่งจะเห็นทั้งชื่ออาหารที่สั่งและราคาโดยรวม สามารถกด “CONFIRM” เพื่อเป็นการดูรายการอย่างละเอียดอีกครั้ง และกด “SUBMIT” เพื่อเป็นการยืนยันการสั่งอาหาร หลังจากสั่งอาหารเรียบร้อยแล้ว แอพพลิเคชั่นจะแสดงจำนวนคิวที่รออยู่ และสถานะของอาหาร หากต้องการยกเลิกอาหารนั้นๆ และสถานะของอาหารเป็น กำลังรอการทำ (Waiting) ลูกค้าจะสามารถยกเลิกเมนูได้ เมื่อได้รับอาหารครบแล้วจะมีปุ่มเรียกพนักงาน “Bill” เป็นการคิดราคาอาหาร

3.1.2.2 ส่วนของพนักงาน (Staff)

พนักงานแบ่งออกเป็น 3 ตำแหน่งคือ Manager, Chef และ Waiter ซึ่งแต่ละตำแหน่งมีหน้าที่ไม่เหมือนกัน ซึ่งมีรายละเอียดดังนี้

Manager จะสามารถเพิ่มอาหารใหม่หรือแก้ไขรายละเอียดเมนูต่างๆได้

Chef จะสามารถดูรายการอาหารที่ลูกค้าสั่งเข้ามาได้ โดยจะมีหมายเลขโต๊ะที่ลูกค้าสั่ง รายการอาหารที่สั่ง จำนวน และมีสถานะของอาหารไว้เพื่ออัปเดตให้ลูกค้าทราบว่าอาหารที่สั่งอยู่ในสถานะใด ซึ่งมี กำลังรอการทำ (Waiting), กำลังจัดเตรียม (Preparing) หรือทำเสร็จแล้ว (Done) หากอาหารอยู่ในสถานะกำลังรอการทำ (Waiting) Chef ก็สามารถยกเลิกเมนูนั้นได้เช่นกัน เมนูนั้นก็จะแสดงสถานะว่ายกเลิก (Cancel) บนจอของลูกค้า

Waiter จะสามารถดูและอัปเดตสถานะของอาหารที่ทำเสร็จแล้ว (Done) เป็น Served ได้ เพื่อรองรับกรณีที่เสิร์ฟอาหารเรียบร้อยแล้ว และเมื่อลูกค้ากด “Bill” แอพพลิเคชั่นจะแจ้งบนหน้าจอว่าลูกค้าโต๊ะใดต้องการพนักงาน

3.2 User requirement

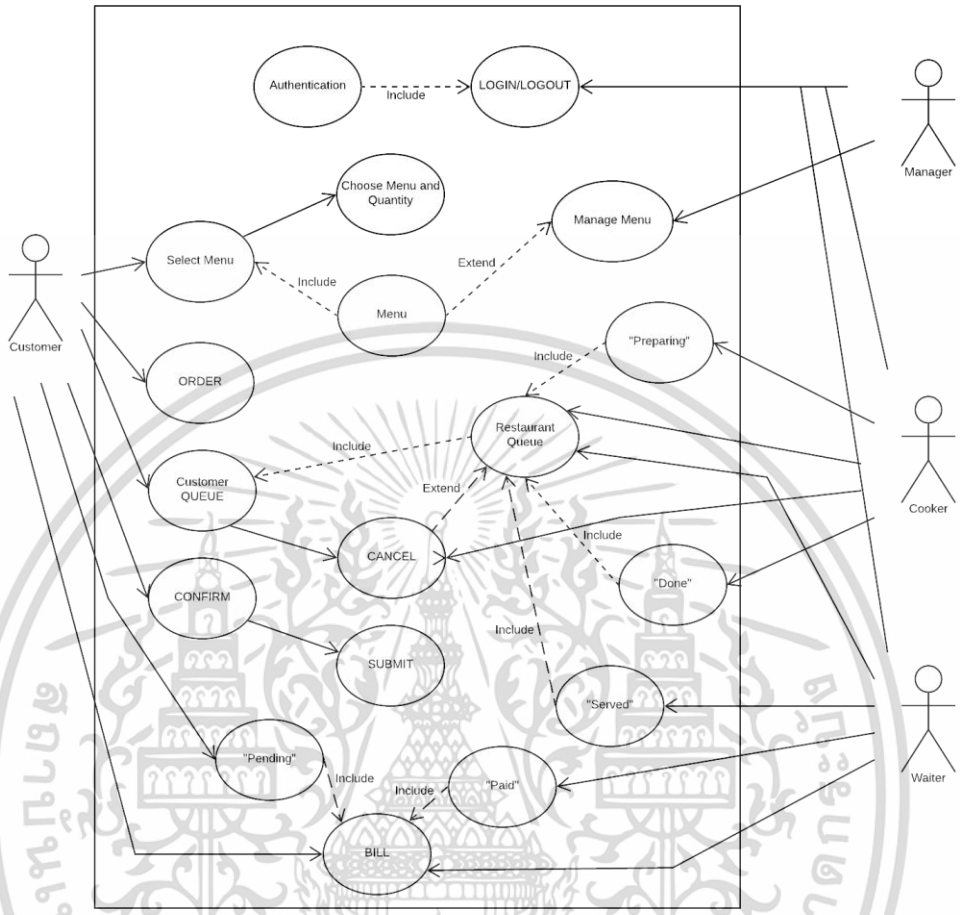
ตาราง 3.1 User requirement specification (Customer)

Details	**Type**	**Priority**
สามารถเลือกดูชนิดของอาหารได้	-User Interface -Functional	จำเป็นต้องมี
สามารถเลือกจำนวนอาหารได้	-User Interface -Functional	จำเป็นต้องมี
สามารถลดหรือยกเลิกอาหารที่เลือกไว้ได้	-Functional	จำเป็นต้องมี
มีลิสต์รายการอาหารที่เลือกไว้	-User Interface -Functional	จำเป็นต้องมี
มีระบบคำนวณราคาอาหารที่สั่งไป	-Function	จำเป็นต้องมี
มีระบบยืนยันการสั่งอาหารที่เลือกไว้	-Function	จำเป็นต้องมี
มีปุ่มกดเรียกพนักงานเพื่อชำระเงิน	-User Interface -Functional	จำเป็นต้องมี
สามารถดูสถานะของอาหารได้	-System -Functional	ควรจะมี
สามารถยกเลิกอาหารที่สั่งไปแล้ว แต่อยู่ในสถานะกำลังรอ (Waiting) ได้	-User Interface -Functional	จำเป็นต้องมี

ตาราง 3.2 User requirement specification (Staff)

Details	**Type**	**Priority**
เข้าสู่ระบบและสามารถเลือกเป็นพนักงาน (Staff)	-System -User Interface -Functional	จำเป็นต้องมี
เข้าสู่ระบบและสามารถเลือกเป็นลูกค้า (Customer)	-System -User Interface -Functional	จำเป็นต้องมี
สามารถเพิ่มเมนูใหม่ๆได้	-User Interface -Functional	จำเป็นต้องมี
สามารถแก้ไขรายละเอียดอาหารได้	-User Interface -Functional	จำเป็นต้องมี
สามารถดูได้ว่าลูกค้าโต๊ะใดสั่งอาหารอะไรได้	-User Interface -Functional	จำเป็นต้องมี
แจ้งเตือนว่าลูกค้าโต๊ะใดต้องการพนักงานเพื่อเก็บเงิน	-User Interface -Functional	จำเป็นต้องมี
สามารถบอกสถานะของอาหารกับลูกค้าได้	-Functional	ควรจะมี

3.3 Use Case Diagram



รูป 3.2 Use Case Diagram

ตาราง 3.3 รายละเอียดของ Use case ฝั่งลูกค้า (Customer)

Use case	**คำอธิบาย**
LOGIN	เข้าสู่ระบบ โดยเลือกเป็นลูกค้า
Select Menu	เมนูจะประกอบไปด้วย “MAINCOURSE", "SALAD", "DESSERT”, “DRINK” ให้เลือก เมื่อ User เลือกเมนูอย่างใดอย่างหนึ่งแล้ว การสั่งอาหารจะขึ้นรูปอาหารด้านล่างจะมีให้เพิ่ม "+", ลด "-" จำนวน
ORDER	สั่งอาหารที่เลือกจาก Select Menu ไว้ จากนั้นจะแสดงรายการที่จะสั่งทางขวามือ
CONFIRM	กดเพื่อแสดงรายการอาหารที่จดไว้ พร้อมราคารวม เมื่อกด “SUBMIT” จะเป็นการยืนยันการสั่งอาหาร และส่งรายการที่สั่งไปยังเซิฟเวอร์
QUEUE	แสดงรายการอาหารที่สั่ง สถานะของอาหาร และแสดงว่าลูกค้ากำลังอยู่ในคิวที่เท่าไหร่
CANCEL	ยกเลิกอาหารที่อยู่ในสถานะ Waiting
BILL	จะเป็นการเรียกพนักงานเพื่อทำการชำระเงิน โดยจะแสดงรายการอาหารที่สั่งไปแล้วทั้งหมด ราคารวม และหมายเลขอ้างอิง

ตาราง 3.4 รายละเอียดของ Use case ฝั่งพนักงาน (Staff)

Use case	**คำอธิบาย**
LOGIN	เข้าสู่ระบบ โดยเลือกเป็นพนักงาน
Manage Menu	เจ้าของร้านสามารถเพิ่มเมนูใหม่ และสามารถแก้ไขสูตรของแต่ละเมนูได้
QUEUE	แสดงรายการและสถานะอาหารที่อยู่ในคิว และโต๊ะที่สั่ง
Preparing	จะเป็นการบอกสถานะว่าอาหารกำลังทำ (Preparing)
Done	จะเป็นการแจ้งพนักงานบริการว่าอาหารพร้อมส่งแล้ว (Done)
Served	พนักงานบริการเปลี่ยนสถานะของอาหารว่าเสิร์ฟแล้ว (Served)
CANCEL	ยกเลิกอาหารที่อยู่ในสถานะ Waiting
BILL	แสดงว่าลูกค้าโต๊ะใดต้องการชำระเงิน (Bill อยู่ในสถานะ Pending)
Paid	เมื่อชำระเรียบร้อย พนักงานสามารถกด CHECK เพื่อจบบิล (Bill เปลี่ยนสถานะเป็น Paid)

ตาราง 3.5 รายละเอียดของ Actor

Actor	**คำอธิบาย**
Customer	ลูกค้า
Manager	ผู้จัดการร้าน/เจ้าของร้าน
Chef	คนครัว
Waiter	คนส่งอาหาร/พนักงานบริการ

3.5.1 Fully Dressed Use Case

Use case: LOGIN โดยเลือกเป็นลูกค้า Customer
Actors: Customer
Preconditions: ไม่มี
Flow of events: 1. กดเข้าสู่ระบบโดยเลือกเป็นลูกค้า
Post conditions: ไปที่หน้าหลักของลูกค้า

Use case: เลือกดูเมนู Select Menu (Customer)
Actors: Customer
Preconditions: 1. ต้อง LOGIN เข้าสู่ระบบในโหมดลูกค้า
Flow of events: 1. ในหน้าแรก กดเลือกประเภทของอาหาร (“MAINCOURSE", "SNACK", "DESSERT”, “DRINK”)
Post conditions: แสดงเมนูในหมวดหมู่ที่เลือกไว้

Use case: เลือกสั่งเมนูและระบุจำนวน Choose Menu and Quantity (Customer)
Actors: Customer
Preconditions: 1. ต้องเลือกประเภทของอาหาร (“MAINCOURSE", "SNACK", "DESSERT”, “DRINK”) ก่อน
Flow of events: 1. เลือกเมนูที่จะสั่ง 2. เลือกจำนวนที่จะส่ง โดยกด “+” เพื่อเพิ่มจำนวนหรือกด “-” เพื่อลดจำนวน
Post conditions: ไม่มี

Use case: กดจดอาหาร ORDER (Customer)
Actors: Customer
Preconditions: 1. ต้องเลือกเมนูและจำนวนอาหารที่จะสั่ง
Flow of events: 1. กด “ORDER” เพื่อสั่งอาหารที่เลือกไว้
Post conditions: แสดงรายการอาหารที่จะสั่งทางขวามือ

Use case: กดดูรายการอาหารที่จด CONFIRM (Customer)
Actors: Customer
Preconditions: ไม่มี
Flow of events: 1. กด “CONFIRM”
Post conditions: แสดงรายการอาหารที่จะสั่งทางขวามือ

Use case: กดยืนยันการสั่งอาหาร SUBMIT (Customer)
Actors: Customer
Preconditions: 1. ต้องเลือกเมนูและจำนวนอาหารที่จะสั่ง 2. ต้องอยู่ที่หน้า CONFIRM
Flow of events: 1. กด “SUBMIT” ยืนยันการสั่งอาหาร
Post conditions: ไปที่หน้า QUEUE

Use case: กดดู QUEUE (Customer)
Actors: Customer
Preconditions: ไม่มี
Flow of events: - กดปุ่ม SUBMIT เพื่อยืนยันการสั่งอาหารและจะเข้าหน้า QUEUE ทันที - เข้าผ่านปุ่ม “QUEUE” ในหน้าหลัก
Post conditions: แสดงรายการ และสถานะอาหารในคิว

Use case: กดยกเลิกอาหาร CANCEL (Customer)
Actors: Customer
Preconditions: 1. ต้องเลือกเมนูและจำนวนอาหารที่จะสั่ง 2. ต้องอยู่ที่หน้า CONFIRM
Flow of events: 1. กด “CANCEL” เพื่อยกเลิกการสั่งอาหาร
Post conditions: อาหารจานนั้นเปลี่ยนเป็นสถานะ Cancel

Use case: กดเรียก BILL (Customer)
Actors: Customer
Preconditions: 1. ได้รับอาหารที่สั่งครบแล้ว (อยู่ในสถานะ Served) 2. อยู่ในหน้า QUEUE
Flow of events: 1. กดปุ่ม “Bill” เรียกพนักงานเพื่อทำการชำระเงิน
Post conditions: แสดงรายการอาหารทั้งหมด พร้อมราคา แจ้งเตือนให้พนักงานทราบ

Use case: LOGIN โดยเลือกเป็นพนักงาน Staff (Manager, Chef, Waiter)
Actors: Staff (Manager, Chef, Waiter)
Preconditions: ไม่มี
Flow of events: 1. เข้าสู่ระบบโดยเลือกเป็นพนักงาน
Post conditions: ไปที่หน้าหลักของพนักงาน

Use case: เพิ่มเมนูใหม่ Add Menu (Staff)
Actors: Manager
Preconditions: 1. ต้อง LOGIN เข้าสู่ระบบในโหมดพนักงาน
Flow of events: 1. กด “ADD MENU” 2. เพิ่มรูป 3. ใส่รายละเอียดของอาหารที่จะเพิ่ม 4. กดปุ่ม ADD MENU
Post conditions: เมนูอาหารจะถูกเพิ่มเข้าไปในระบบ

Use case: แก้ไขเมนู Edit Menu (Staff)
Actors: Manager
Preconditions: 1. ต้องมีเมนูอย่างน้อย 1 เมนูในระบบ
Flow of events: 1. สามารถแก้ไขรายละเอียดเมนูอาหารได้
Post conditions: เมนูอาหารที่เลือกถูกแก้ไข

Use case: ดู QUEUE ของลูกค้า (Staff)
Actors: Manager
Preconditions: 1. ต้อง LOGIN เข้าสู่ระบบในโหมดพนักงาน
Flow of events: 1. กด “QUEUE”
Post conditions: แสดงรายการอาหารที่เหลืออยู่ในคิว โดยจะระบุเมนูอาหารและหมายเลขโต๊ะที่สั่ง

Use case: เปลี่ยนสถานะอาหารเป็นกำลังทำอาหาร Preparing (Staff)
Actors: Chef
Preconditions: 1. ต้อง LOGIN เข้าสู่ระบบในโหมดพนักงาน 2. อยู่ในหน้า QUEUE
Flow of events: 1. กดเลือกอาหารที่อยู่ในสถานะ Waiting 2. กดปุ่ม “UPDATE”
Post conditions: แจ้งเตือนลูกค้าว่าอาหารกำลังทำ (Preparing)

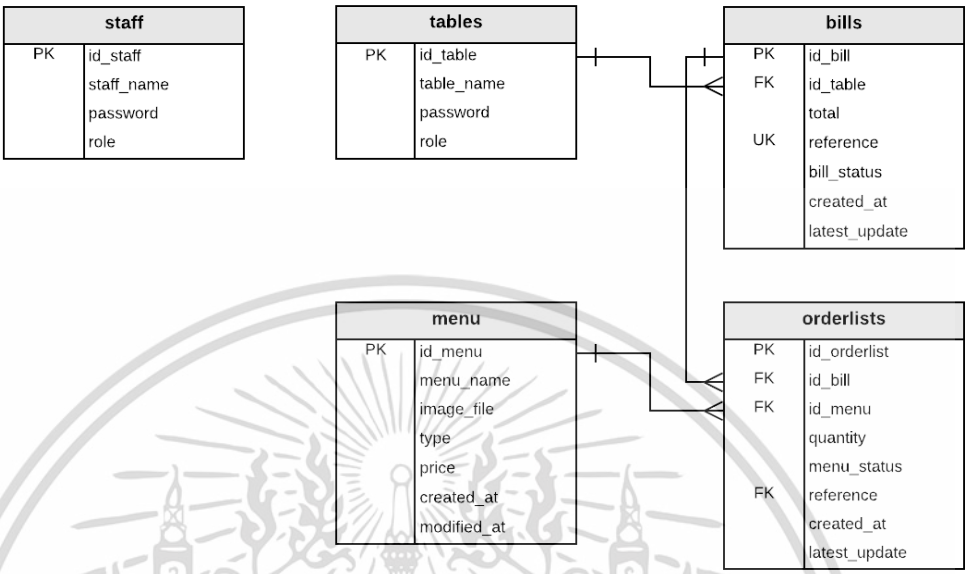
Use case: เปลี่ยนสถานะของอาหารเป็นปรุงเสร็จเรียบร้อยแล้ว Done (Staff)
Actors: Chef
Preconditions: 1. อาหารอยู่ในสถานะ Preparing
Flow of events: 1. กดเลือกอาหารที่อยู่ในสถานะ Preparing 2. กดปุ่ม “UPDATE”
Post conditions: แสดงสถานะอาหารแจ้งเตือนลูกค้าว่าอาหารปรุงเสร็จแล้ว (Done)

Use case: เปลี่ยนสถานะของอาหารเป็นเสิร์ฟแล้ว Served (Staff)
Actors: Waiter
Preconditions: 1. อาหารอยู่ในสถานะ Done
Flow of events: 1. กดเลือกอาหารที่อยู่ในสถานะ Done 2. กดปุ่ม “UPDATE”
Post conditions: แสดงสถานะอาหารกับลูกค้าว่าอาหารเสิร์ฟแล้ว (Served)

Use case: กดดู BILL เพื่อตรวจสอบการเรียกพนักงานของลูกค้าเพื่อชำระเงิน (Staff)
Actors: Waiter
Preconditions: ไม่มี
Flow of events: กดปุ่ม “BILL” เพื่อตรวจสอบการเรียกพนักงานของลูกค้าเพื่อชำระเงิน
Post conditions: แสดงข้อมูล และรายละเอียดอาหารของโต๊ะที่ลูกค้าต้องการชำระเงิน

Use case: เปลี่ยนสถานะบิลเป็นจ่ายแล้ว Paid (Staff)
Actors: Waiter
Preconditions: 1. บิลอยู่ในสถานะ Pending
Flow of events: 1. กดปุ่ม “CHECK” เพื่อยืนยันการชำระเงิน
Post conditions: เปลี่ยนสถานะบิลเป็นจ่ายแล้ว Paid

3.4 Entity Relationship Diagram (ER Diagram)



รูป 3.3 ER Diagram

3.4.1 ตารางพนักงาน (staff)

ตารางพนักงาน เป็นตารางที่มีไว้เพื่อให้พนักงานเข้าใช้งาน ไม่ว่าจะเป็นพนักงานตำแหน่งใดก็ตามเมื่อเข้าแอพพลิเคชั่นต้องผ่านการยืนยันตัวตนก่อน ภายในตารางนี้จะประกอบไปด้วย ไอดีของพนักงาน ชื่อของพนักงาน รหัสของพนักงาน และตำแหน่งของพนักงาน

3.4.2 ตารางโต๊ะของลูกค้า (tables)

ตารางโต๊ะของลูกค้า เป็นตารางที่มีไว้เพื่อให้พนักงานเข้าสู่ระบบ และวางไว้บนโต๊ะอาหารเพื่อให้ลูกค้าใช้บริการสั่งอาหารผ่านแอพพลิเคชั่น ภายในตารางนี้จะประกอบไปด้วย ไอดีของโต๊ะ หมายเลขโต๊ะ รหัสของโต๊ะ และตำแหน่งของโต๊ะ(ในที่นี้จะกำหนดเป็นลูกค้าทั้งหมด)

3.4.3 ตารางบิลอาหาร (bills)

ตารางบิลอาหาร เป็นตารางที่มีไว้เพื่อเก็บข้อมูลที่เกี่ยวกับการสั่งอาหาร ภายในตารางนี้จะประกอบไปด้วย ไอดีเลขบิล ไอดีของโต๊ะ ราคาอาหารทั้งหมดในบิลนั้นๆ หมายเลขอ้างอิง สถานะของบิล เวลาที่สร้างบิล และเวลาที่อัปเดตบิลล่าสุด

3.4.4 ตารางรายการอาหารที่สั่ง (orderlists)

ตารางรายการอาหารที่สั่ง เป็นตารางที่มีไว้เพื่อเก็บข้อมูลรายละเอียดรายการอาหารที่สั่งทั้งหมด ภายในตารางนี้จะประกอบไปด้วย ไอดีรายการอาหาร ไอดีเลขบิล ไอดีของเมนูอาหาร จำนวนอาหารที่สั่ง สถานะของอาหาร หมายเลขอ้างอิง เวลาที่สั่ง และเวลาที่อัปเดตสถานะอาหารล่าสุด

3.4.5 ตารางเมนูอาหาร (menu)

ตารางเมนูอาหาร เป็นตารางที่เก็บข้อมูลอาหาร ภายในตารางนี้จะประกอบไปด้วย ไอดีของอาหาร ชื่ออาหาร ชื่อรูปของอาหาร ชนิดของอาหาร ราคาของอาหาร เวลาที่สร้างรายการอาหารนั้นๆ และเวลาที่แก้ไขรายการอาหารนั้นๆล่าสุด

ตาราง 3.6 รายละเอียดตารางพนักงาน (staff)

ชนิดของคีย์	ชื่อแอตทริบิวต์	ชนิดตัวแปร	คำอธิบาย
PK	id_staff	Int	ไอดีของพนักงาน
	staff_name	String	ชื่อของพนักงาน
	password	String	รหัสของพนักงาน
	role	String	ตำแหน่งของพนักงาน

ตาราง 3.7 รายละเอียดตารางโต๊ะของลูกค้า (tables)

ชนิดของคีย์	ชื่อแอตทริบิวต์	ชนิดตัวแปร	คำอธิบาย
PK	id_table	Int	ไอดีของโต๊ะ
	table_name	String	หมายเลขโต๊ะ
	password	String	รหัสของโต๊ะ
	role	String	ตำแหน่งของโต๊ะ(กำหนดเป็นลูกค้าทั้งหมด)

ตาราง 3.8 รายละเอียดตารางบิลอาหาร (bills)

ชนิดของคีย์	ชื่อแอตทริบิวต์	ชนิดตัวแปร	คำอธิบาย
PK	id_bill	Int	ไอดีของบิล
FK	id_table	Int	ไอดีของโต๊ะ
	total	Int	ราคาอาหารรวมของบิล
UK	reference	String	หมายเลขอ้างอิง
	bill_status	String	สถานะบิล
	created_at	Datetime	เวลาที่สร้างบิล
	latest_update	Datetime	เวลาที่อัปเดตบิลล่าสุด

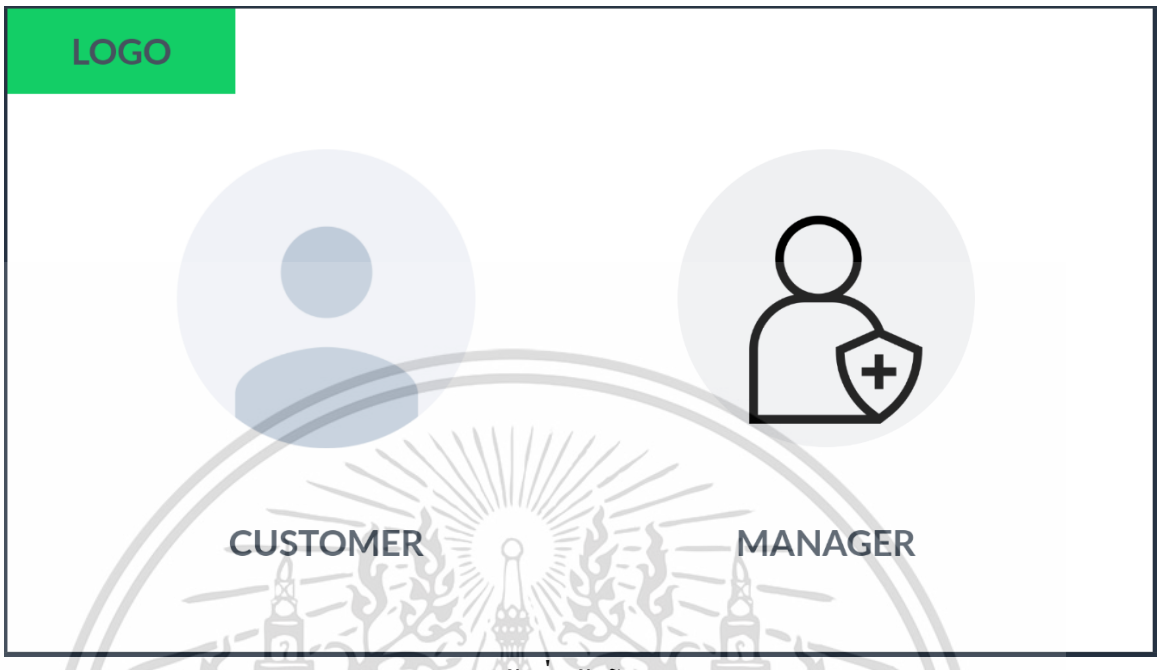
ตาราง 3.9 รายละเอียดตารางรายการอาหารที่สั่ง (orderlists)

ชนิดของคีย์	ชื่อแอตทริบิวต์	ชนิดตัวแปร	คำอธิบาย
PK	id_orderlist	Int	ไอดีของรายการอาหารที่สั่ง
FK	id_bill	Int	ไอดีบิลของรายการอาหาร
FK	id_menu	Int	ไอดีเมนูอาหาร
	quantity	Int	จำนวนของอาหาร
	menu_status	String	สถานะของอาหาร
FK	reference	String	หมายเลขอ้างอิง
	created_at	Datetime	เวลาที่สั่งอาหาร
	latest_update	Datetime	เวลาที่อัปเดตสถานะล่าสุด

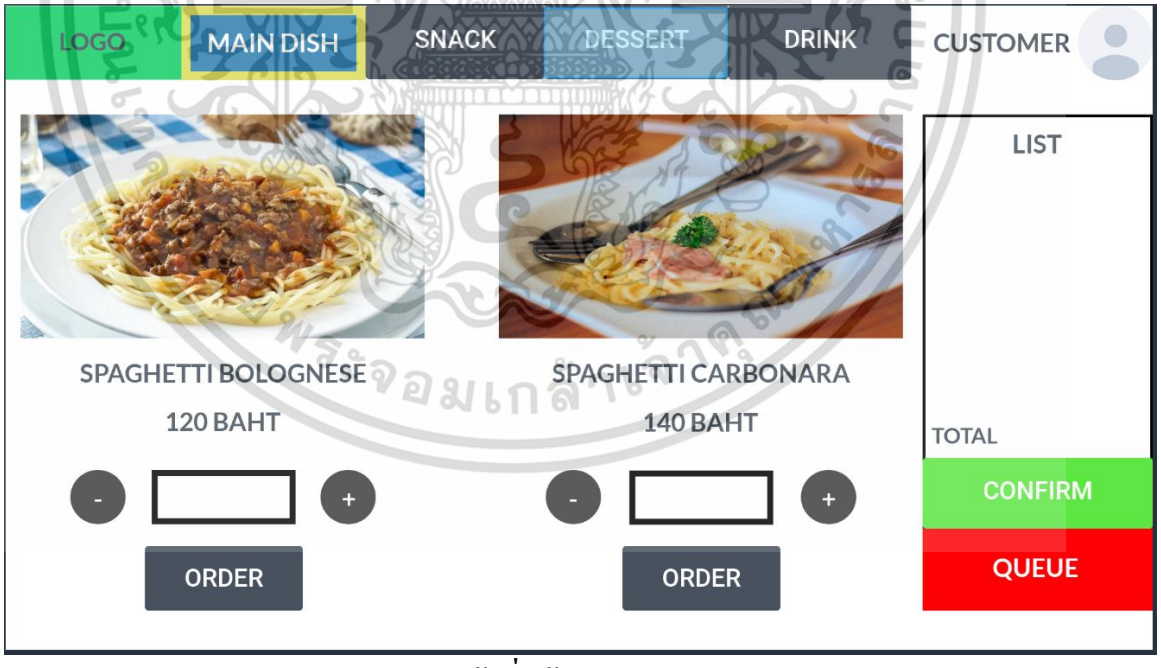
ตาราง 3.10 รายละเอียดตารางเมนูอาหาร (menu)

ชนิดของคีย์	ชื่อแอตทริบิวต์	ชนิดตัวแปร	คำอธิบาย
PK	id_menu	Int	ไอดีของอาหาร
	menu_name	String	ชื่อของอาหาร
	image_file	String	ชื่อไฟล์ของอาหาร
	type	String	ชนิดของอาหาร
	price	Int	ราคาของอาหาร
	created_at	Datetime	เวลาที่สร้างเมนู
	modified_at	Datetime	เวลาที่แก้ไขเมนูล่าสุด

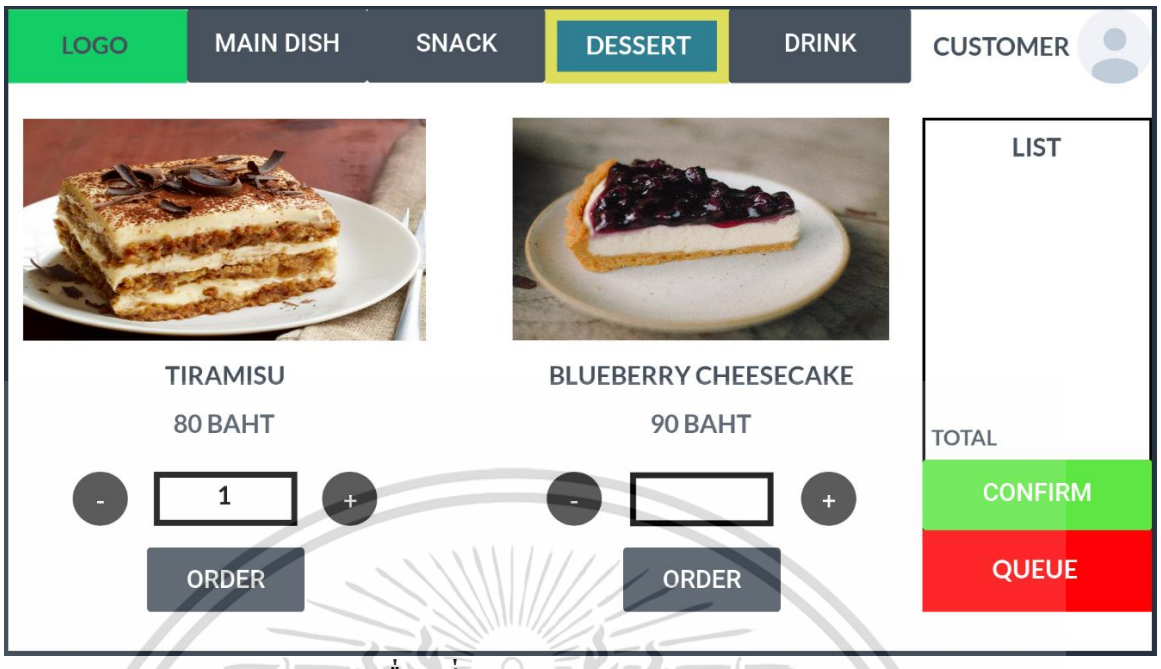
3.5 การออกแบบหน้าตาโปรแกรม



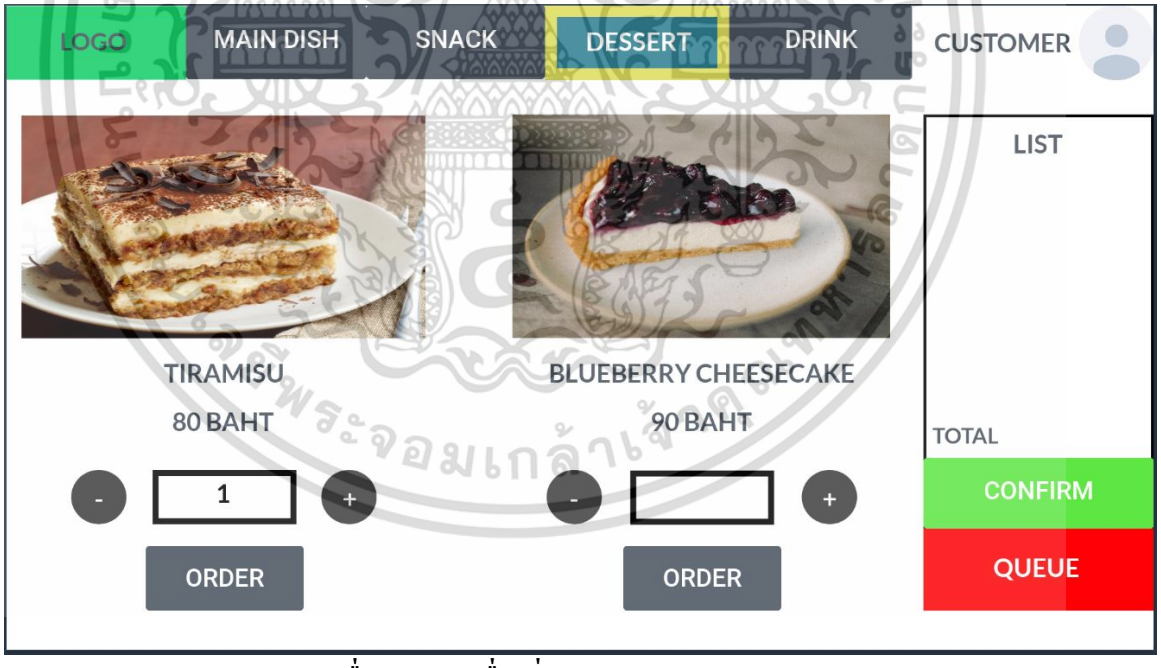
รูป 3.4 หน้าเริ่มต้นโปรแกรม



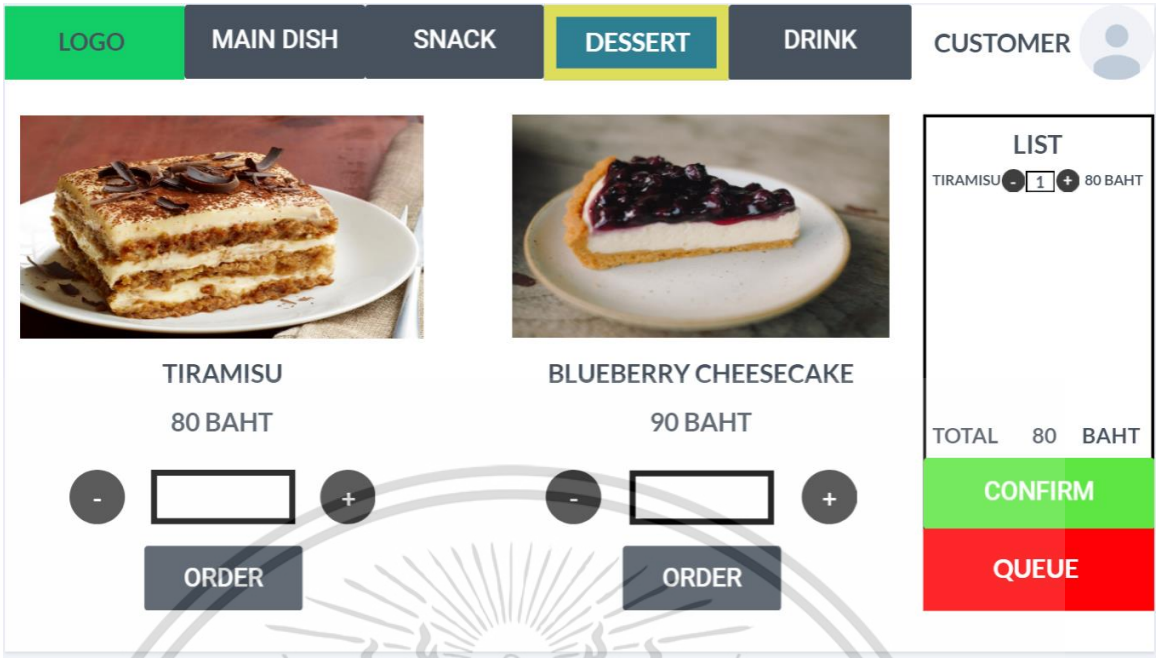
รูป 3.5 หน้าเริ่มต้นของ Customer



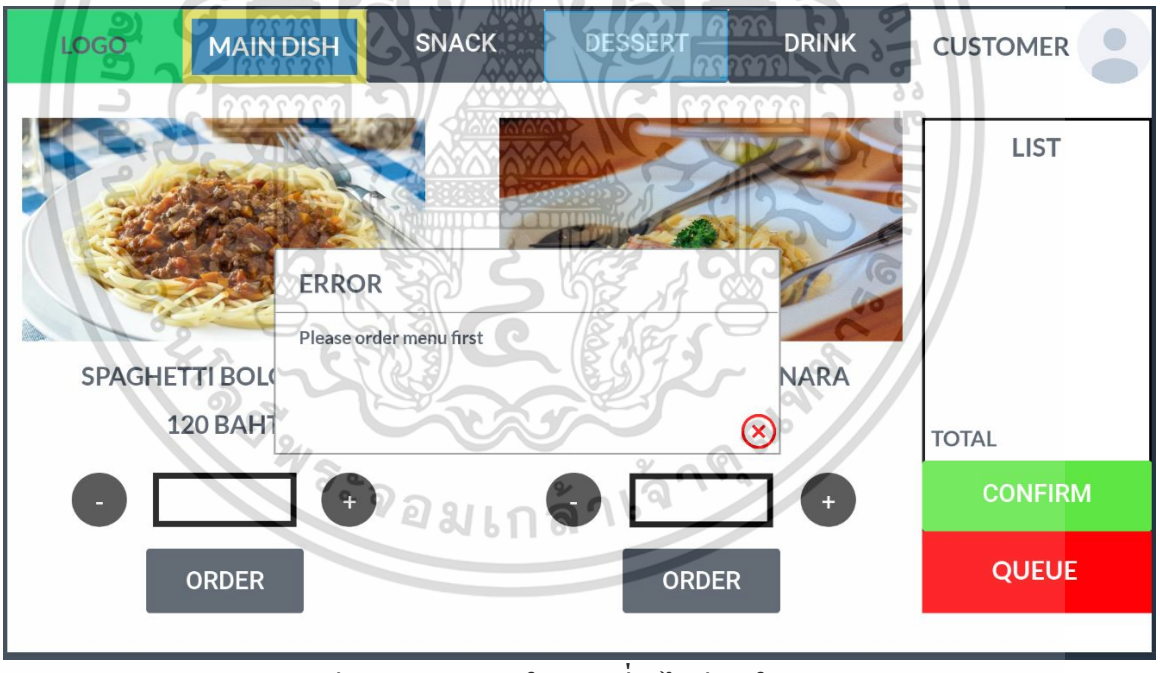
รูป 3.6 เมื่อเปลี่ยนหมวดหมู่อาหาร (Customer)



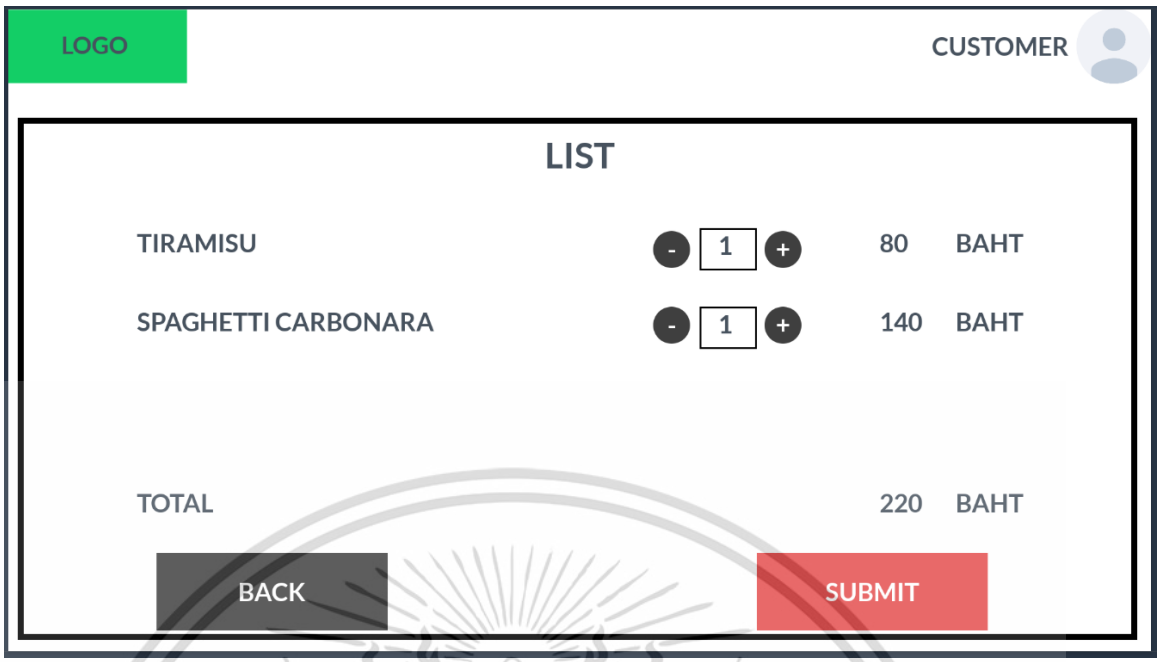
รูป 3.7 เมื่อกด “+” เพื่อเพิ่มจำนวนอาหาร (Customer)



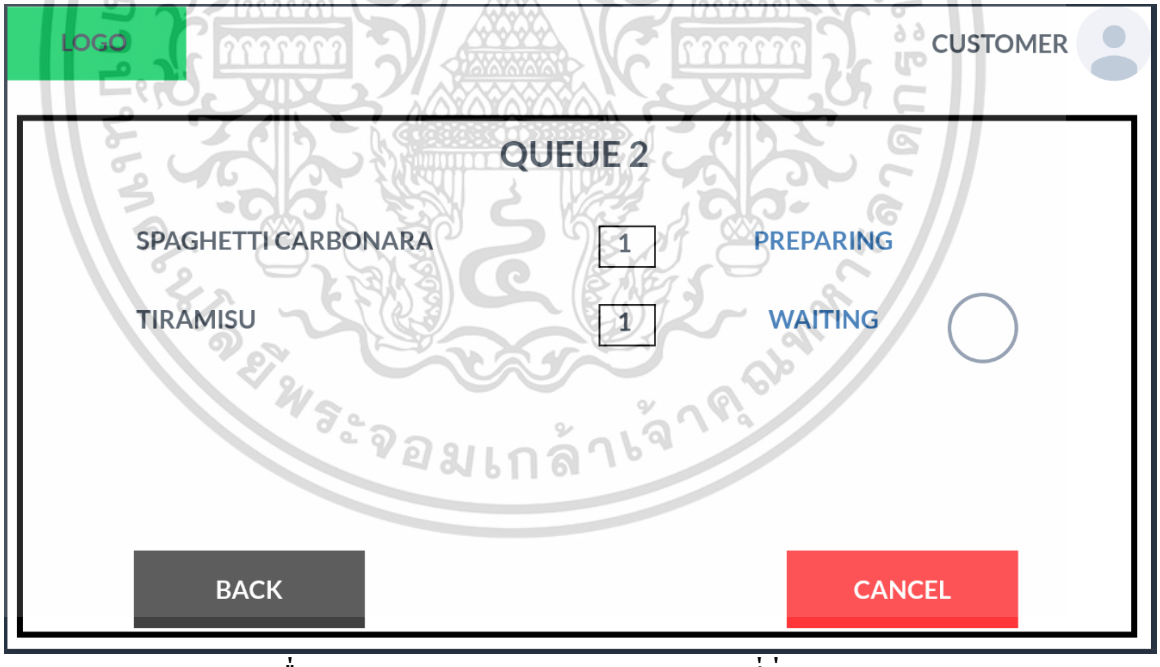
รูป 3.8 เมื่อกด “ORDER” เพื่อสั่งอาหาร (Customer)



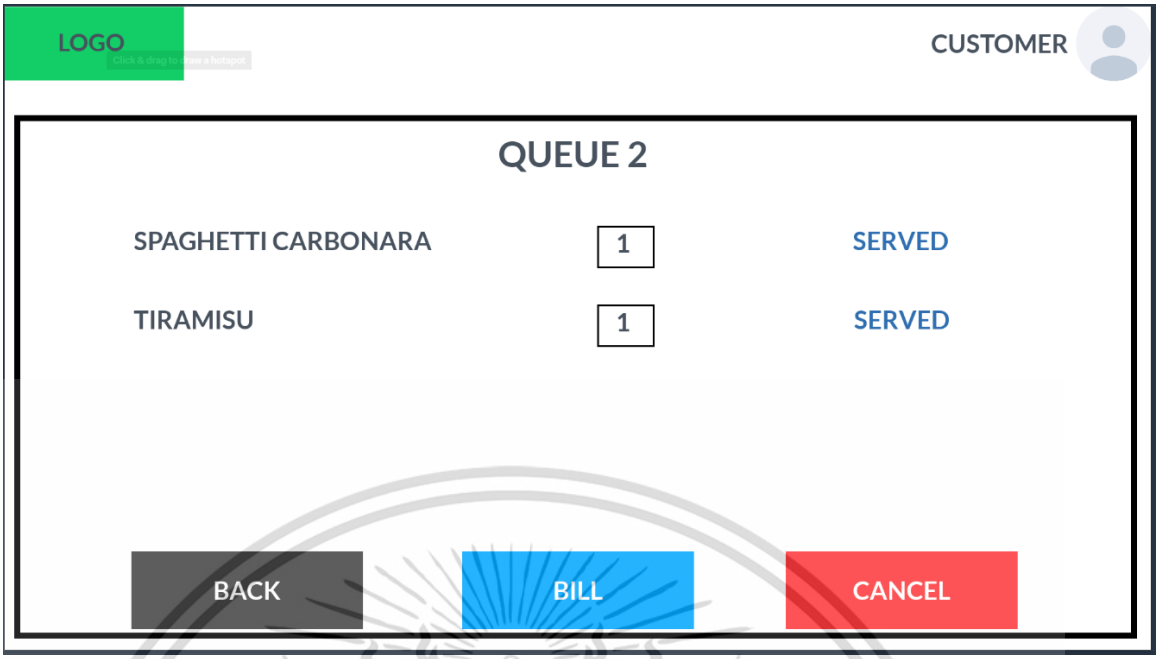
รูป 3.9 Error กรณีกด CONFIRM ในขณะที่ยังไม่มีเมนูใน List (Customer)



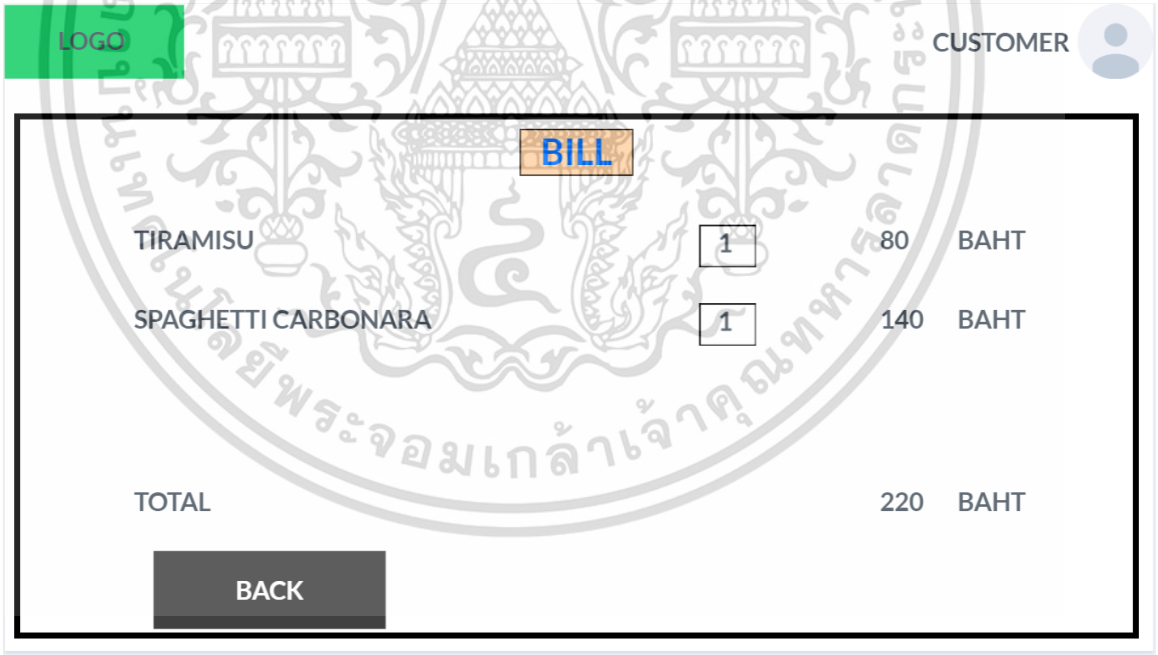
รูป 3.10 เมื่อกด “CONFIRM” จะแสดงรายการอาหารที่รายการไว้ (Customer)



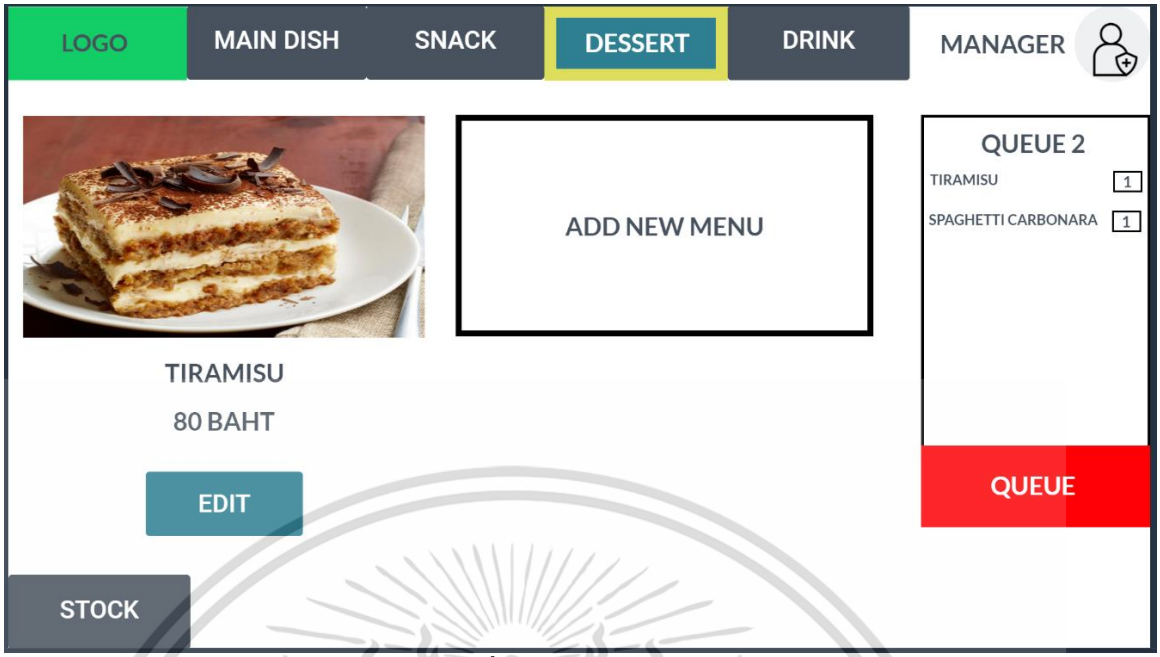
รูป 3.11 เมื่อกด “QUEUE” จะแสดงรายการอาหารที่สั่ง (Customer)



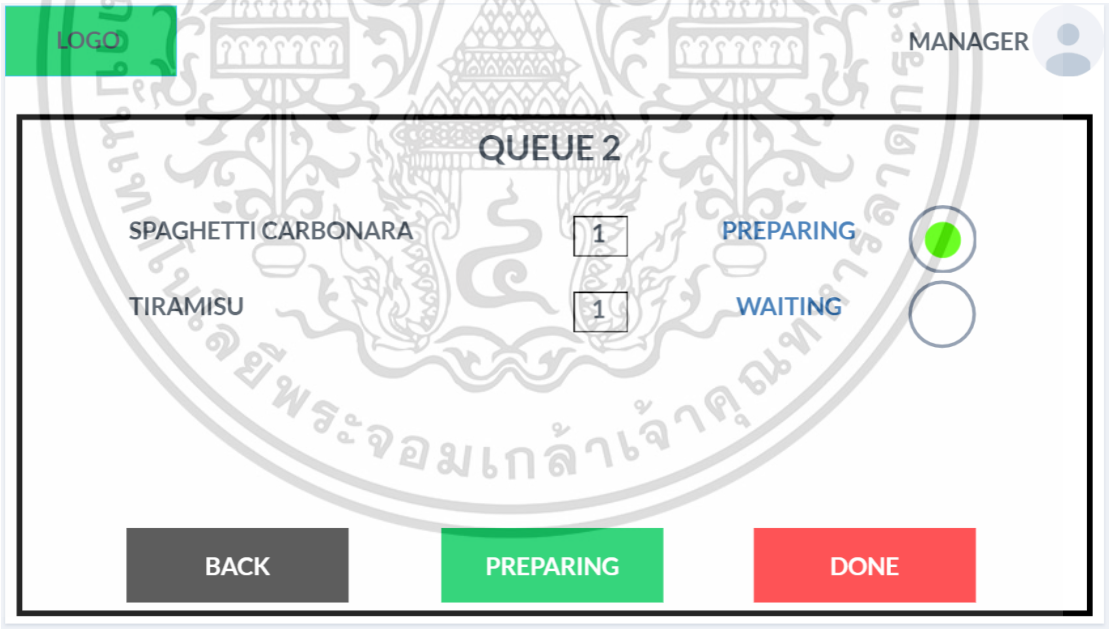
รูป 3.12 หน้า QUEUE กรณีอาหารถูกเสิร์ฟหมดแล้วกด BILL เรียกพนักงานได้ (Customer)



รูป 3.13 หน้า BILL (Customer)



รูป 3.14 หน้าเริ่มต้นของพนักงาน (Staff)



รูป 3.15 หน้า QUEUE (Chef, Waiter)

บทที่ 4
การทดลองและผลการทดลอง

4.1 ร่างต้นแบบ (Mock up)

แอพพลิเคชั่นออกแบบเป็น 2 ส่วนคือส่วนของลูกค้า (Customer) และส่วนของพนักงาน (Staff) โดยทั้ง 2 ส่วนนี้ออกมาแบบเพื่อให้รองรับการใช้งานบน Tablet บนระบบปฏิบัติการแอนดรอยด์

4.1.1 ร่างต้นแบบของส่วนลูกค้า (Customer)

การออกแบบในส่วนของลูกค้า เริ่มต้นจะให้พนักงานเข้าสู่ระบบให้ก่อน จากนั้นโปรแกรมจะเข้าสู่หน้าหลักในส่วนของลูกค้า ซึ่งจะมีรายการให้ลูกค้าทำดังต่อไปนี้

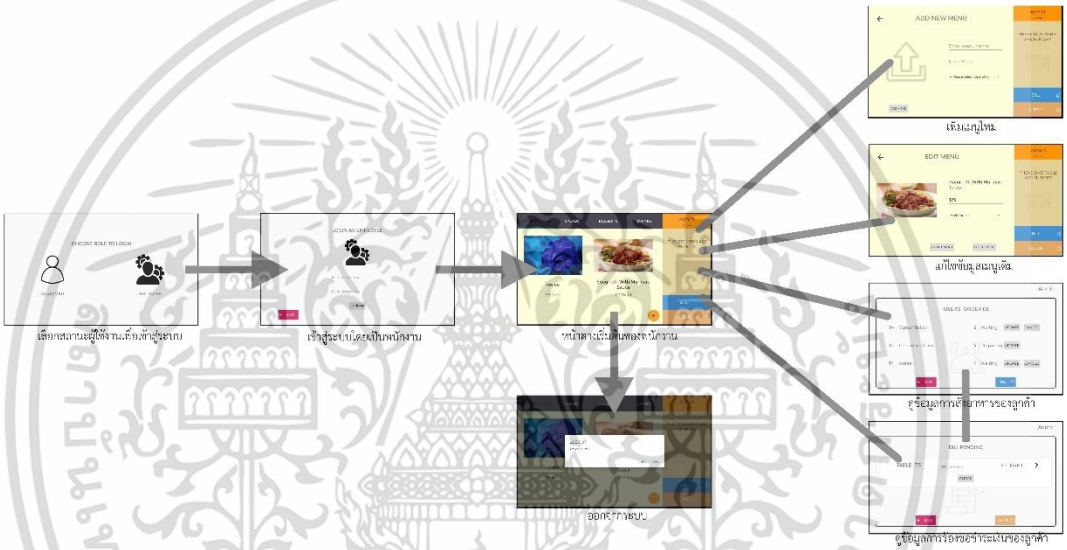
- สั่งอาหาร
- เพิ่มหรือลดจำนวนอาหารที่จดไว้ ซึ่งในส่วนนี้ลูกค้าสามารถเพิ่มหรือลดจำนวนได้ทั้งทางหน้าหลัก หรือหน้าแสดงรายการอาหารที่จดไว้ทั้งหมด
- สามารถแสดงรายการอาหารจดไว้ได้ทันที
- เมื่อกดยืนยันการสั่งอาหารแล้ว จะสามารถรับรู้คิวของตนเอง และสถานะของอาหารที่สั่งไปแล้วได้
- สามารถแสดงเมนูที่สั่งและได้รับเรียบร้อยไปแล้วได้เมื่อเรียกชำระเงิน

รูป 4.1 ร่างต้นแบบของส่วนลูกค้า (Customer)

4.1.2 ร่างต้นแบบของส่วนพนักงาน (Staff)

การออกแบบในส่วนของพนักงาน เริ่มต้นจะให้พนักงานจะต้องเข้าสู่ระบบโดยเลือกเป็นพนักงานก่อน จากนั้นโปรแกรมจะเข้าสู่หน้าหลักในส่วนของพนักงาน ซึ่งจะมีรายการให้ทำดังต่อไปนี้

- เพิ่มเมนูใหม่ หรือแก้ไขเมนูที่มีอยู่แล้ว
- แสดงการสั่งอาหารของลูกค้าได้คร่าวๆในหน้าหลัก
- แสดงรายละเอียดการสั่งอาหารของลูกค้าได้
- แสดงรายละเอียดการชำระเงินของลูกค้าได้

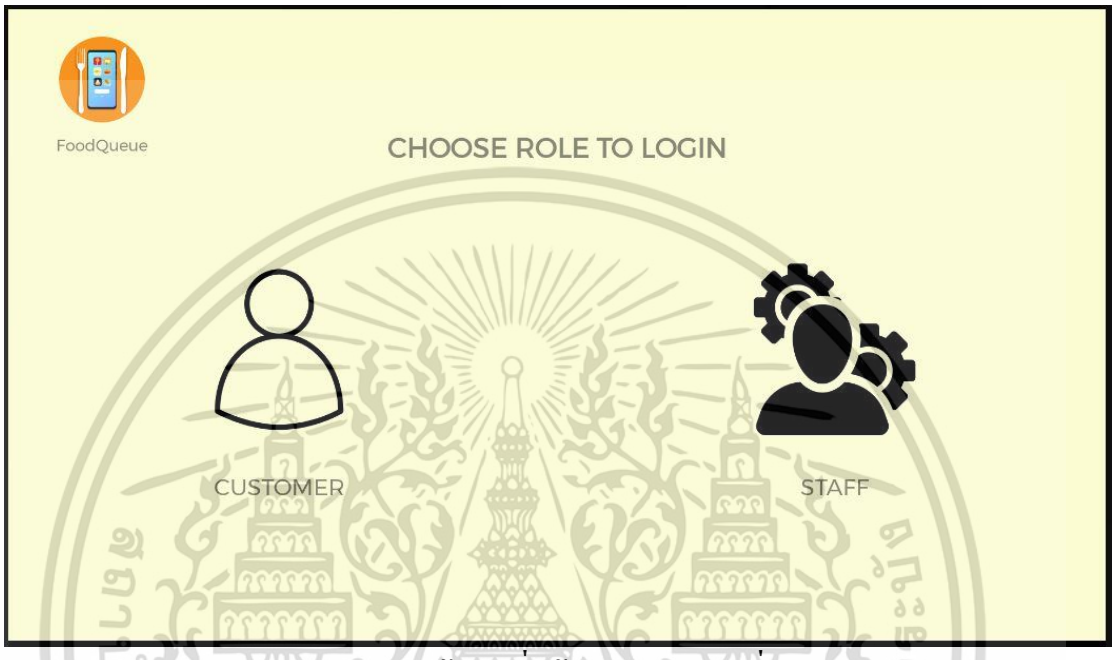


รูป 4.2 ร่างต้นแบบของส่วนพนักงาน (Staff)

4.2 ผลการดำเนินการ

4.2.1 แอพพลิเคชั่นในส่วนของลูกค้า (Customer)

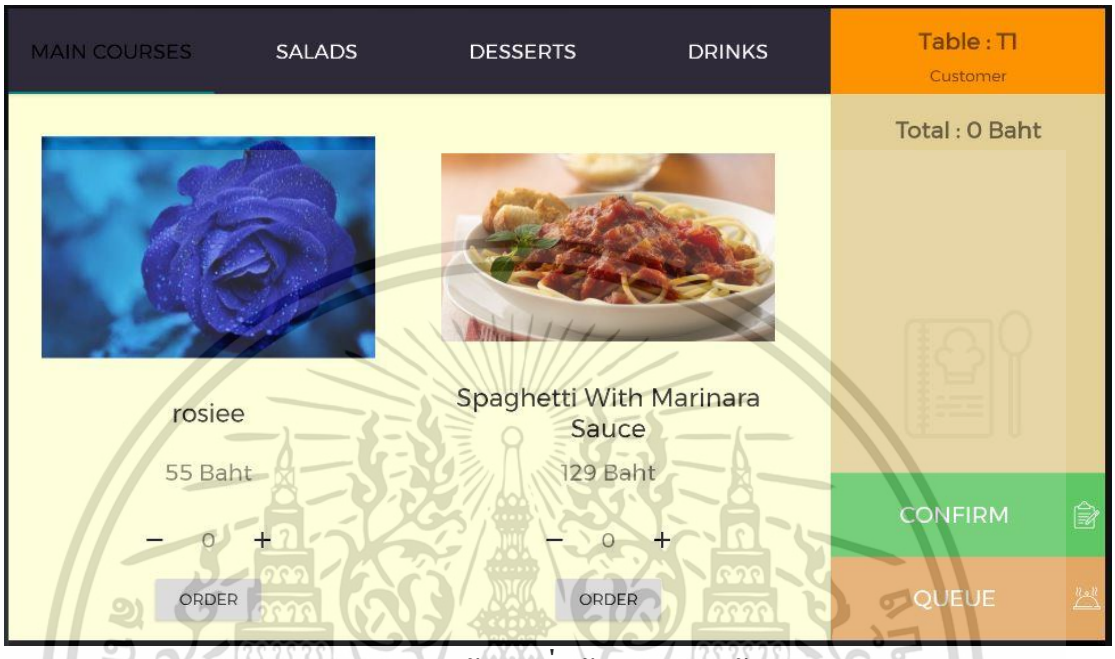
ในการเริ่มใช้โปรแกรมครั้งแรกหากต้องการเข้าสู่ระบบโดยเป็นลูกค้า ให้พนักงานเข้าสู่ระบบบนเครื่องนั้นไว้ก่อน



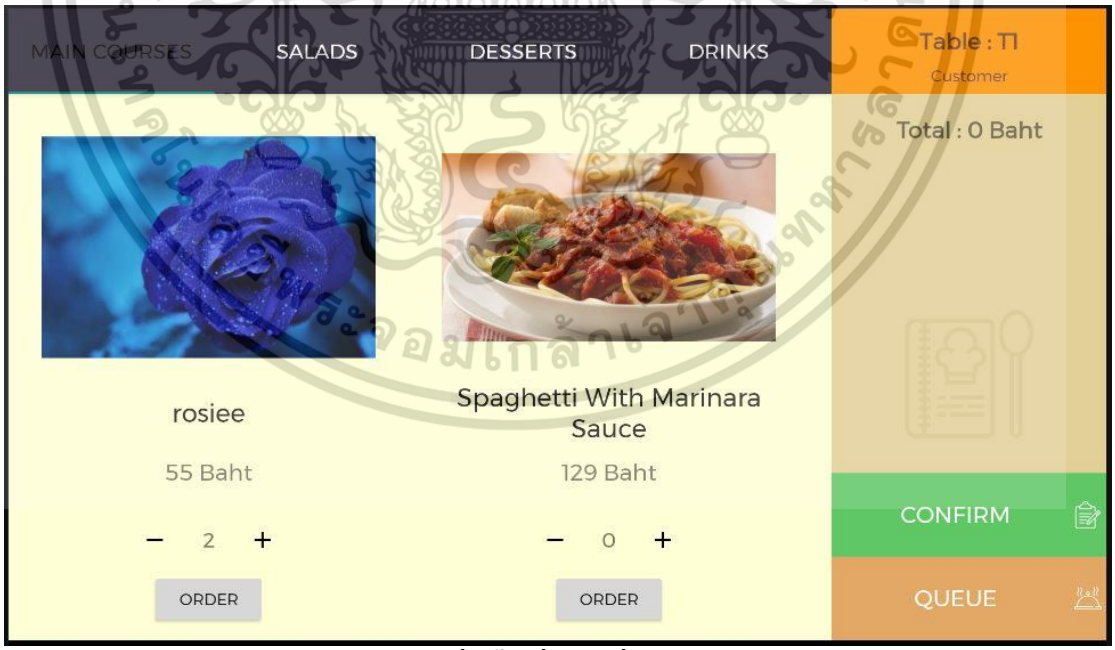
รูป 4.3 หน้าต่างเริ่มต้นของแอพพลิเคชั่น

รูป 4.4 หน้าต่างเข้าสู่ระบบสำหรับของลูกค้า

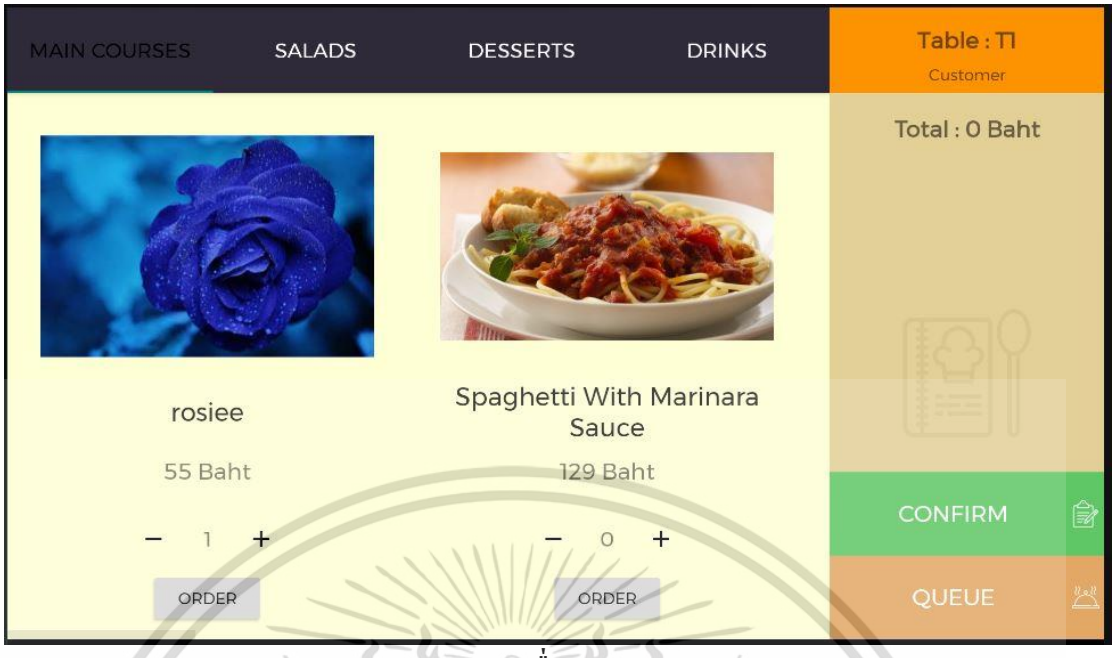
เมื่อเข้าสู่ระบบได้แล้วจะมีหน้าต่างเริ่มต้นของลูกค้า เลขโต๊ะของลูกค้าจะปรากฏอยู่ที่มุมขวาบน หมวดหมู่ของอาหารจะอยู่ทางด้านบน หมวดหมู่ใดที่กำลังเลือกอยู่ ตัวอักษรของหมวดหมู่นั้นจะเป็นสีเข้ม และในส่วนนี้ลูกค้าสามารถเพิ่มหรือลดอาหารลงรายการที่จดไว้ได้



รูป 4.5 หน้าต่างเริ่มต้นสำหรับลูกค้า

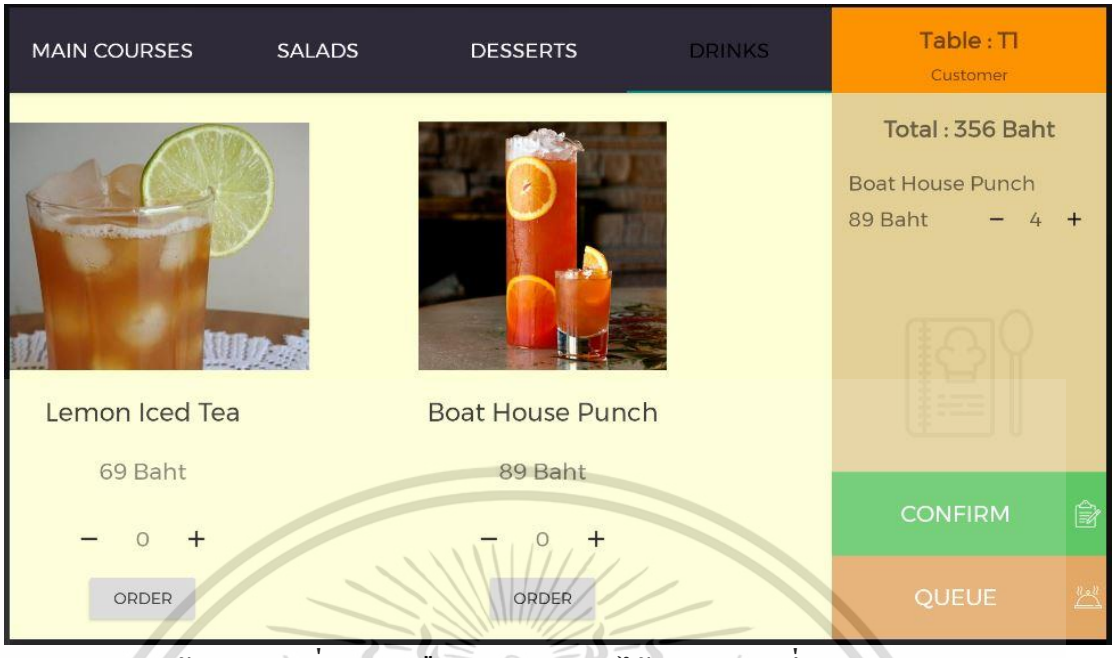


รูป 4.6 จำนวนเพิ่มขึ้นเมื่อกดเพิ่ม “+” (Customer)



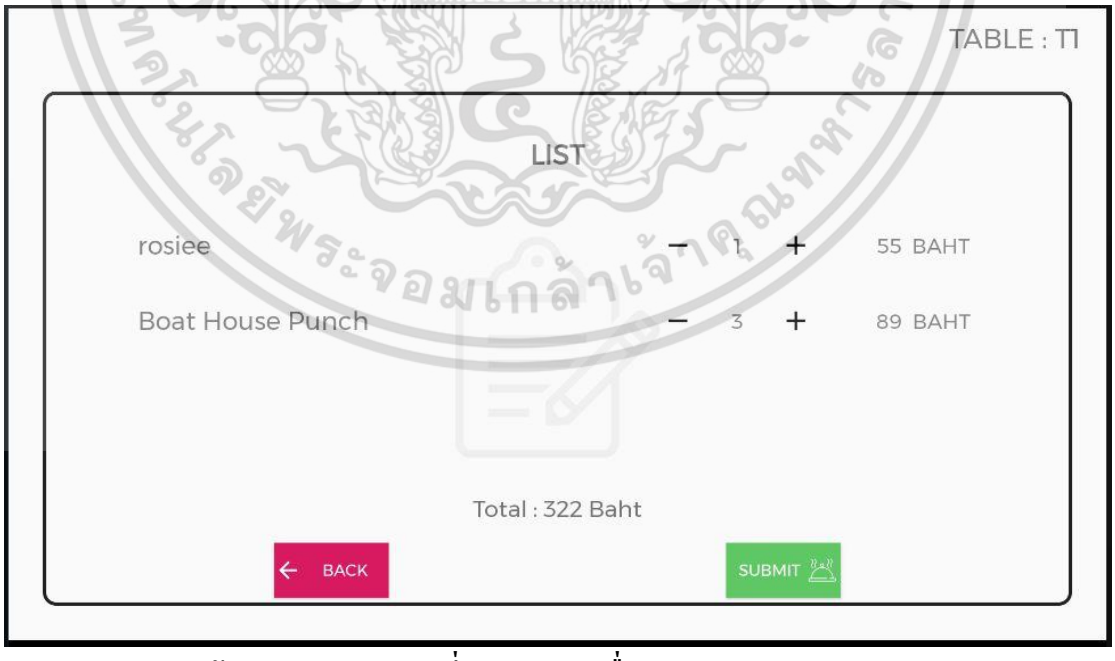
รูป 4.7 จำนวนลดลงเมื่อกดลด “-” (Customer)

รูป 4.8 จดเข้ารายการอาหารที่จะสั่งเมื่อกดปุ่ม “ORDER” (Customer)

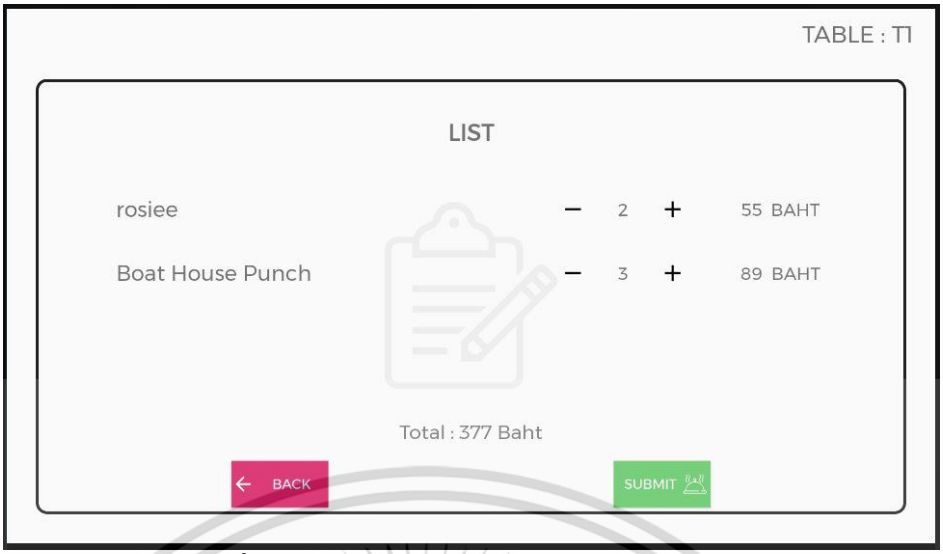


รูป 4.9 ลูกค้าสามารถเพิ่ม “+” หรือลด “-” จำนวนได้ทางรายการที่จดเช่นกัน (Customer)

เมื่อกดปุ่ม “CONFIRM” จะพามายังหน้าต่าง LIST แสดงรายการอาหารที่ได้จดไว้ ในหน้านี้ลูกค้ายังคงสามารถเพิ่มหรือลดจำนวนอาหารได้ หรือย้อนกลับไปหน้าหลักเพื่อเพิ่มเมนูใหม่ได้เช่นกัน หากต้องการยืนยันการสั่ง ให้กดปุ่ม “SUMBIT” เพื่อยืนยันการสั่ง และส่งข้อมูลไปยังเซิฟเวอร์เพื่อให้พนักงานทราบ



รูป 4.10 หน้าต่างรายการอาหารที่จด จะแสดงเมื่อกดปุ่ม “CONFIRM” (Customer)



รูป 4.11 ลูกค้าสามารถเพิ่ม “+” หรือลด “-” จำนวนได้ผ่านทางรายการอาหาร (Customer)

เมื่อยืนยันการสั่งแล้ว จะพามายังหน้าต่าง QUEUE หน้านี้จะแสดงจำนวนคิวที่รอ และหน้าแสดงว่าลูกค้าสั่งอะไรไปแล้วบ้าง และสถานะของอาหารเป็นอย่างไร โดยสถานะของอาหารมีดังนี้

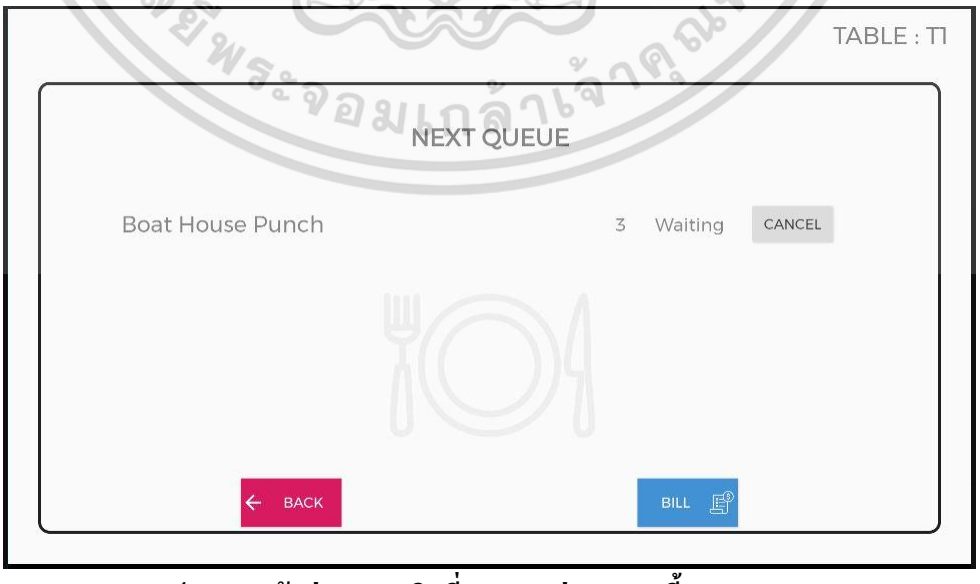
Waiting คือกำลังรออาหาร ในตอนนี้ลูกค้าสามารถยกเลิกอาหารที่สั่งไปแล้วได้โดยการกด “CANCEL”

Preparing คือกำลังอยู่ในขั้นตอนทำอาหาร

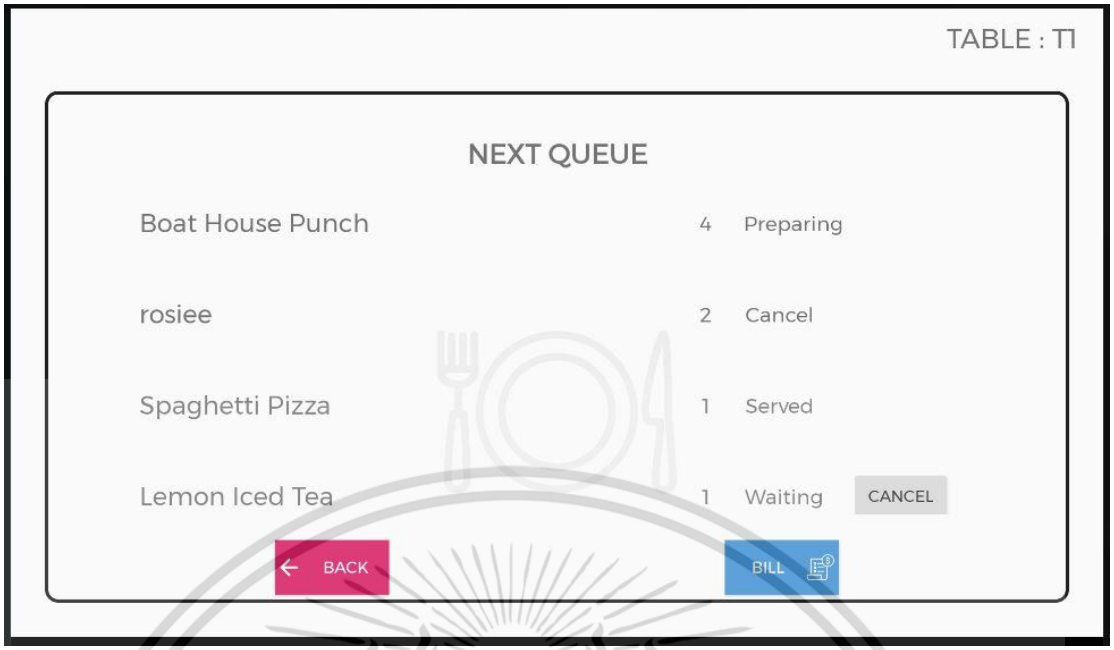
Done คืออาหารปรุงเสร็จแล้ว

Served คืออาหารถูกเสิร์ฟเรียบร้อยแล้ว

Cancel คืออาหารถูกยกเลิก



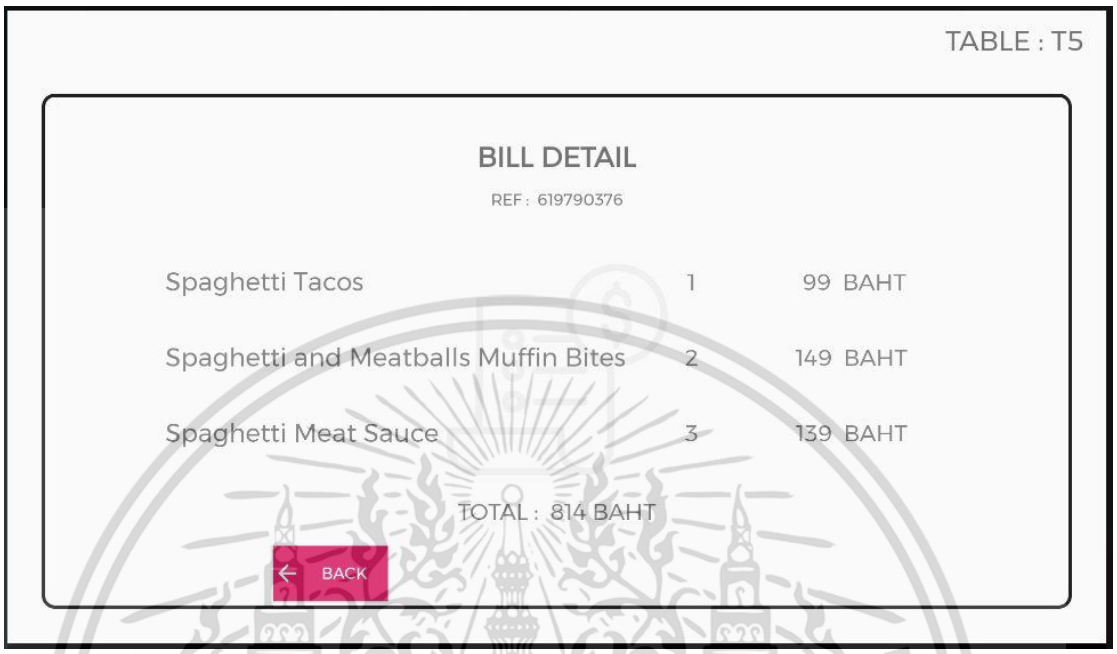
รูป 4.12 หน้าต่างแสดงคิวที่ตนเองอยู่ ณ ขณะนี้ (Customer)



รูป 4.13 แสดงสถานะของอาหาร (Customer)

รูป 4.14 เมื่อลูกค้าได้รับอาหารครบทุกจานแล้ว และต้องการชำระเงิน (Customer)

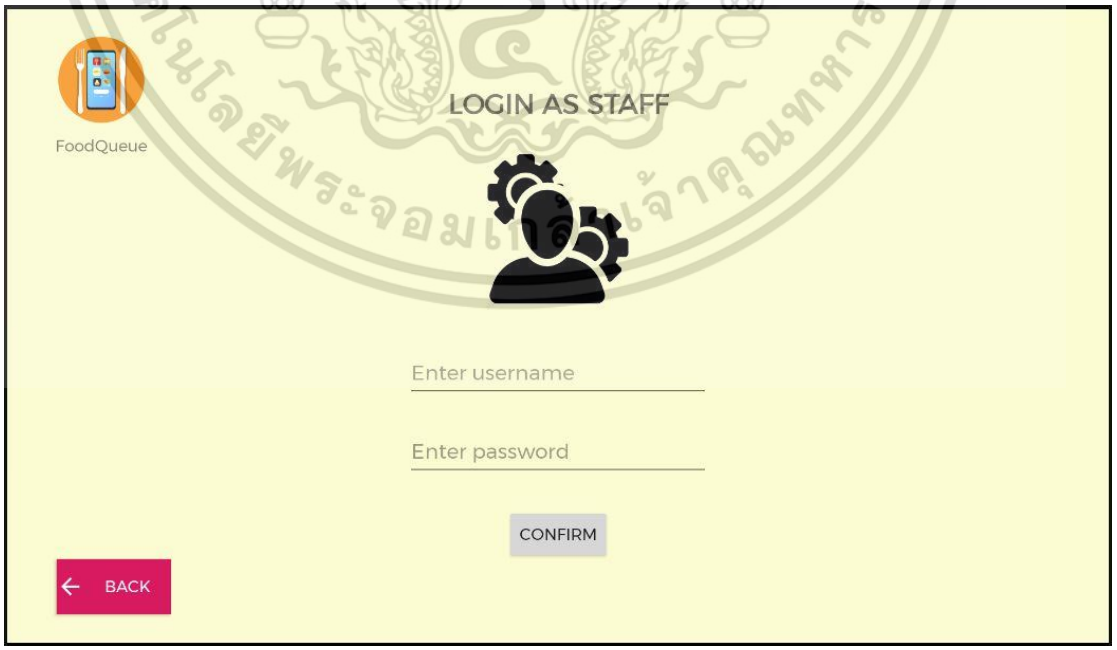
หลังจากกดปุ่มเรียกชำระเงินแล้ว จะแสดงหน้า BILL และจะแสดงรายละเอียดการสั่งอาหาร รวมทั้งเลขอ้างอิง จำนวน และราคารวมทั้งหมด



รูป 4.15 แสดงรายละเอียดรายการอาหารที่สั่ง (Customer)

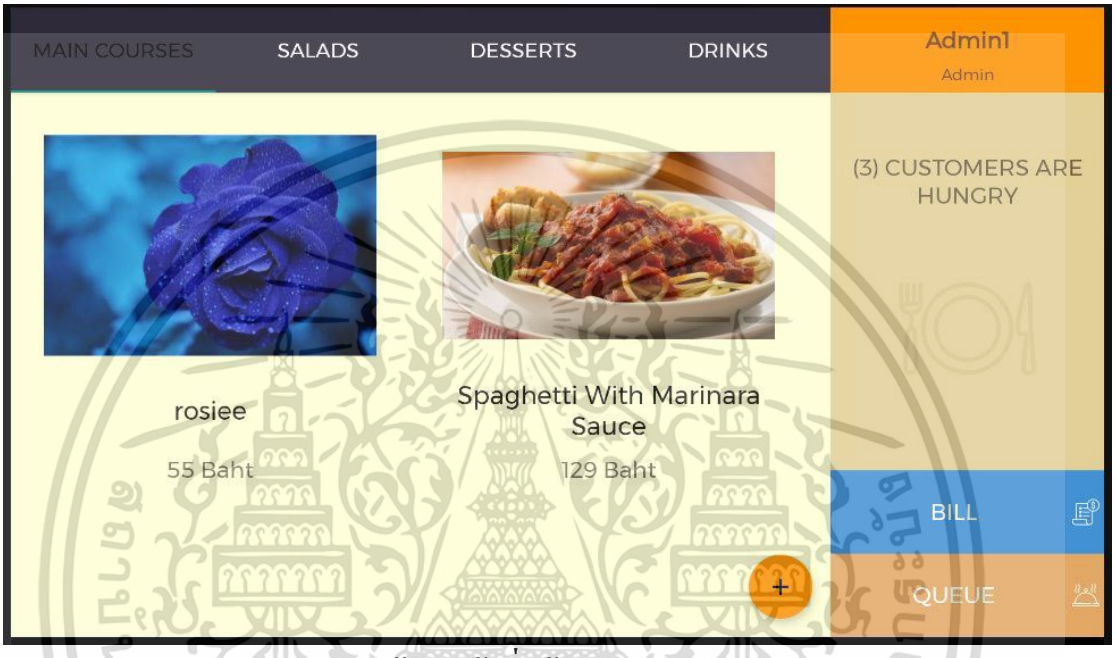
4.2.2 แอพพลิเคชั่นในส่วนของพนักงาน (Staff)

ในการเริ่มใช้โปรแกรมครั้งแรกต้องเลือกการเข้าสู่ระบบโดยเลือกเป็นพนักงาน



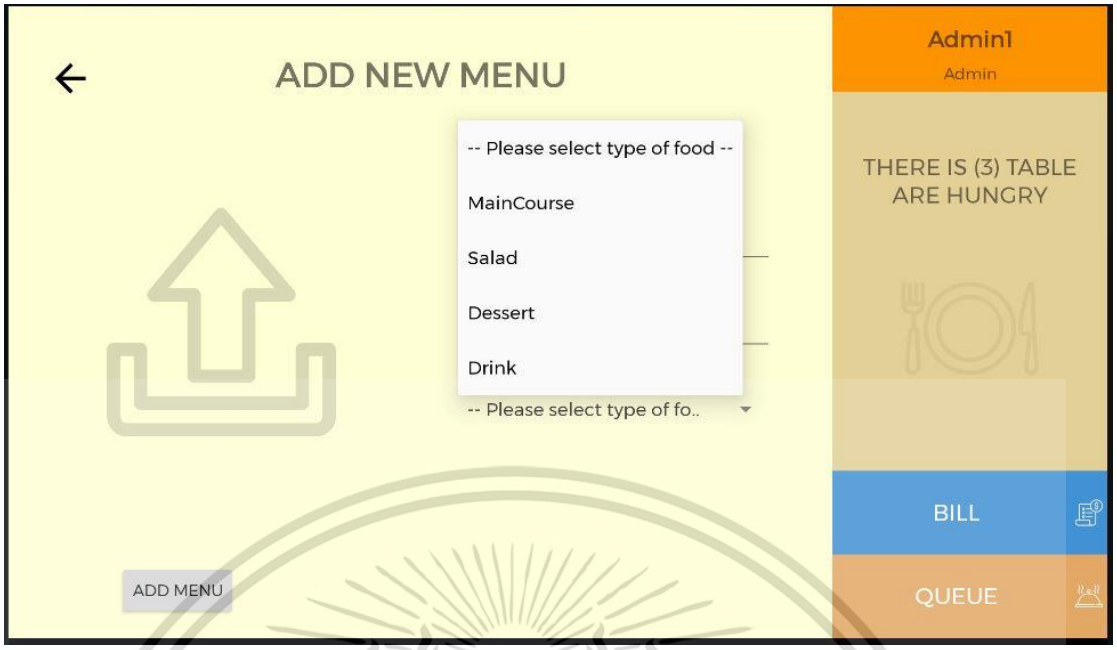
รูป 4.16 หน้าจอการเลือกเข้าใช้งานโดยเป็นพนักงาน (Staff)

เมื่อเข้าสู่ระบบแล้วจะเข้าหน้าหลักของพนักงาน โดยพนักงานจะแบ่งเป็น 3 ลักษณะซึ่งมีหน้าที่แตกต่างกันไปดังนี้ Manager สามารถจัดการเกี่ยวกับเมนอาหารได้, Chef คนทำอาหารสามารถอัปเดตสถานะอาหารให้ลูกค้าทราบได้, Waiter พนักงานบริการสามารถอัปเดตสถานะอาหารเป็นเสิร์ฟแล้ว และสามารถจัดการการชำระเงินได้ และทางด้านขวาจะแสดงว่ามีการสั่งอาหารมาแล้วกี่โต๊ะ

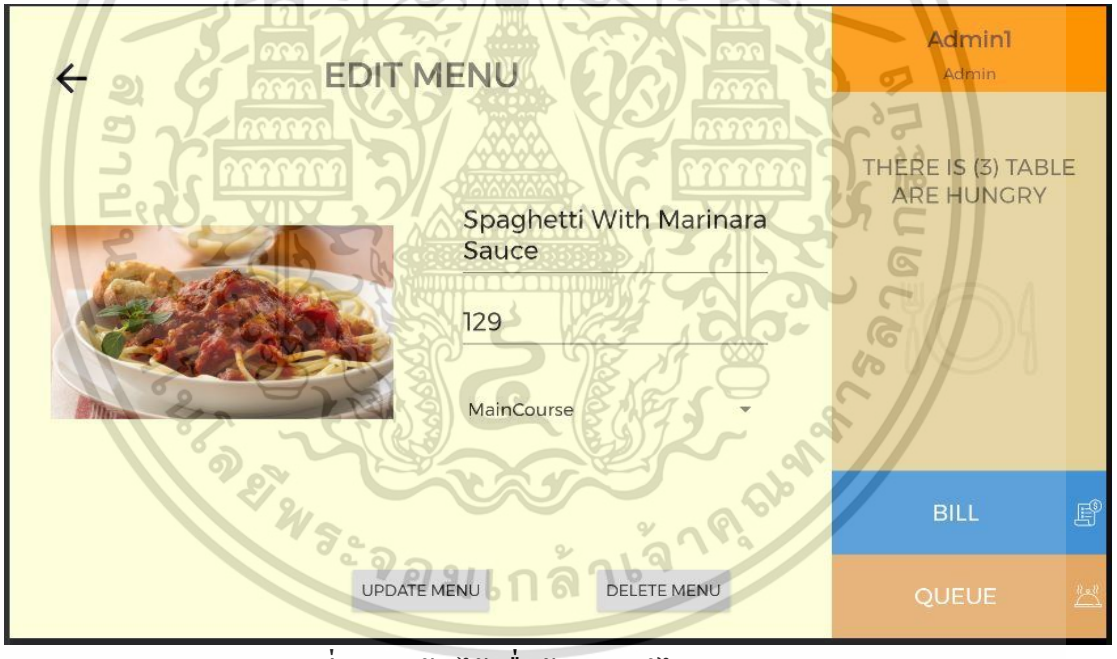


รูป 4.17 หน้าต่างเข้าเริ่มต้นสำหรับพนักงาน (Staff)

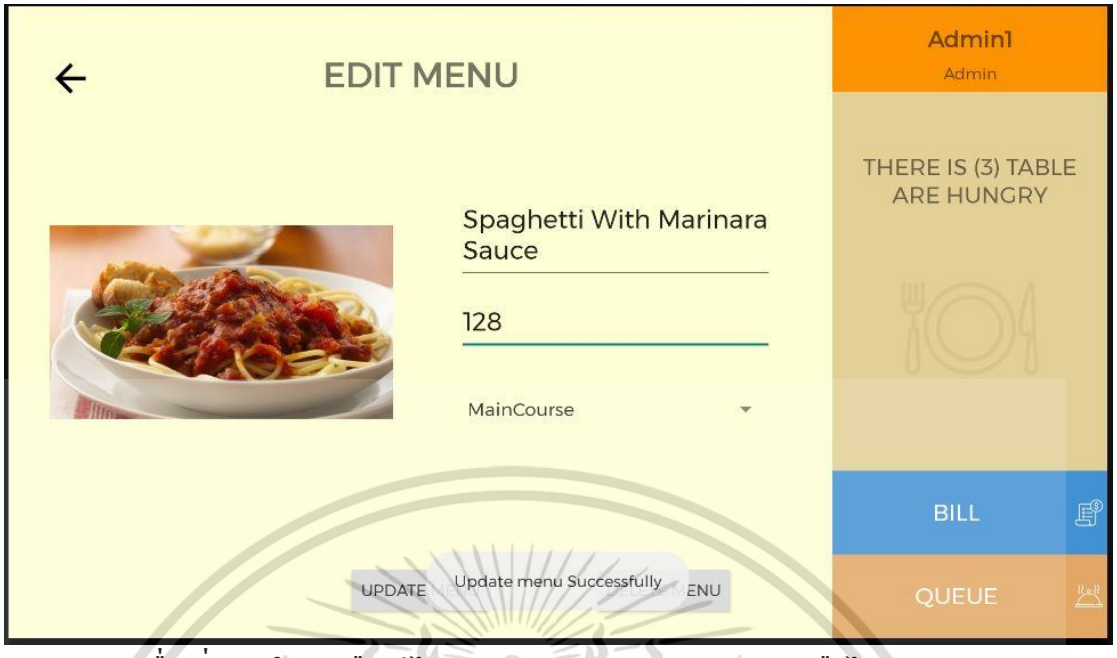
รูป 4.18 กดปุ่ม “+” เมื่อต้องการเพิ่มเมนูใหม่ (Manager only)



รูป 4.19 การใส่รายละเอียดของอาหาร (Manager only)

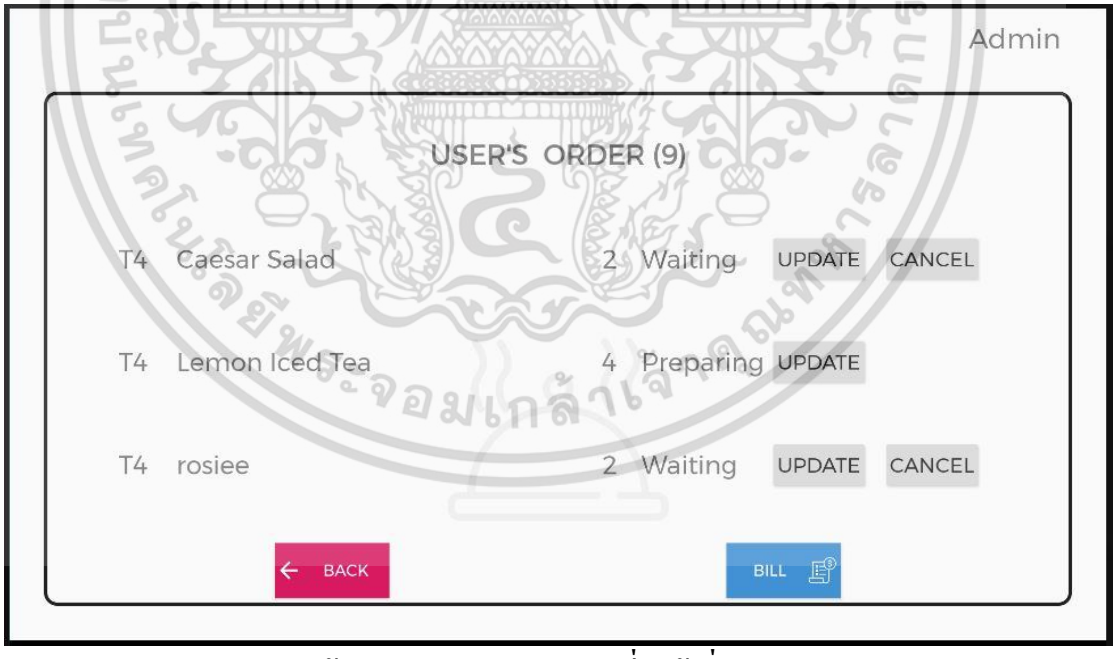


รูป 4.20 กดที่รูปเมนูค้างไว้ เมื่อต้องการแก้ไขเมนู (Manager only)



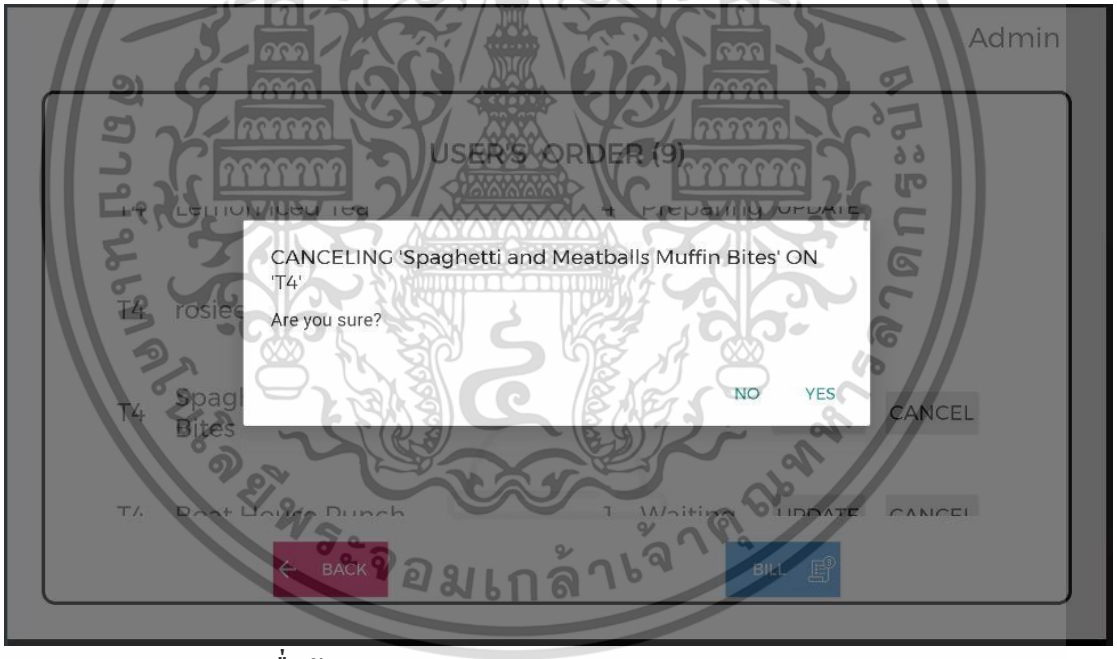
รูป 4.21 เมื่อเพิ่มเมนูใหม่ หรือแก้ไขเมนูจะแสดงรายการว่าทำสำเร็จหรือไม่ (Manager only)

เมื่อกดปุ่ม “QUEUE” จะเข้าสู่หน้าแสดงรายการอาหารที่ลูกค้าสั่ง ว่าโต๊ะใดต้องการสั่งอะไรบ้าง เป็นจำนวนเท่าไหร่ จะมีสถานะให้ทำการอัปเดต หรือยกเลิก



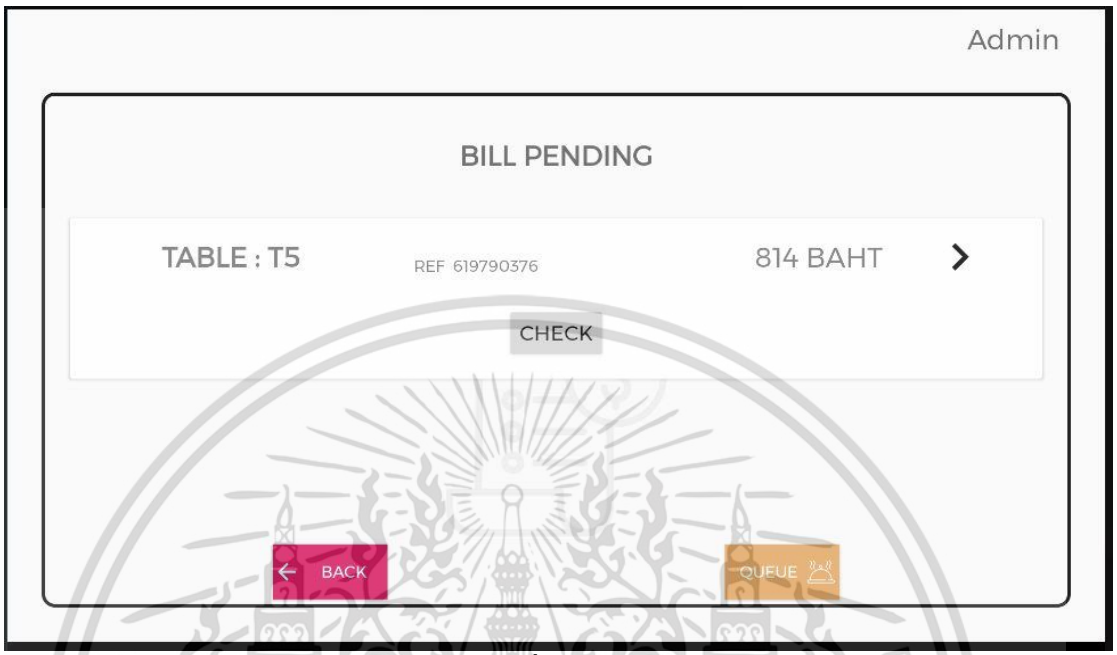
รูป 4.22 หน้าต่างแสดงรายการอาหารที่ลูกค้าสั่ง (Chef, Waiter)

รูป 4.23 เมื่อต้องการอัปเดตสถานะของอาหารกดปุ่ม “UPDATE” (Chef, Waiter)

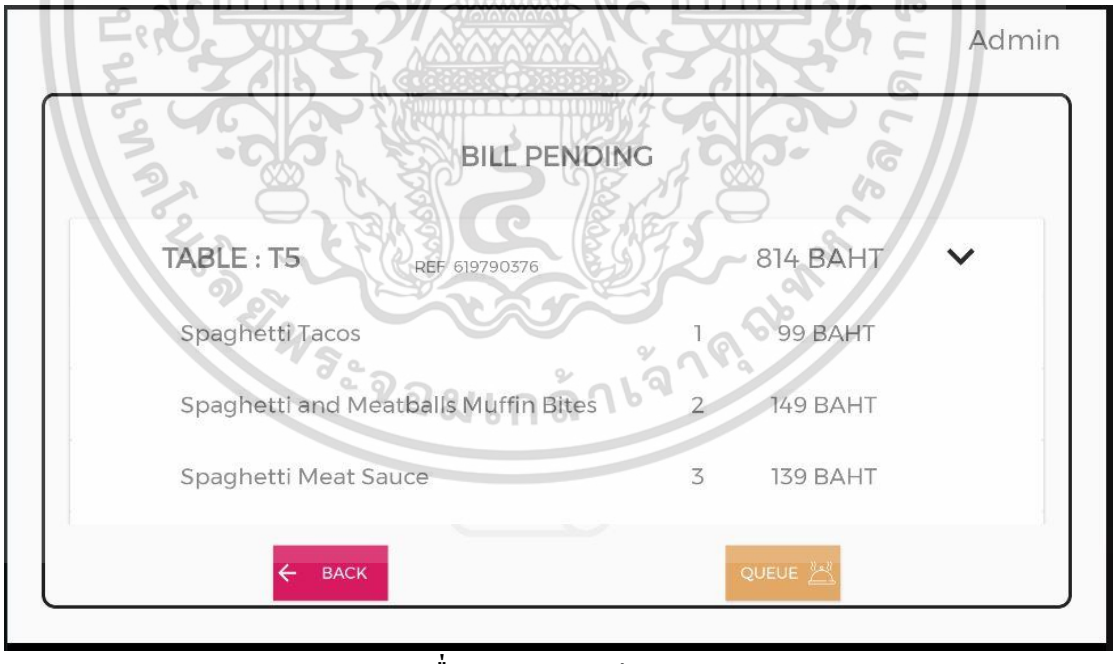


รูป 4.24 เมื่อต้องการยกเลิกอาหารกดปุ่ม “CANCEL” (Chef, Waiter)

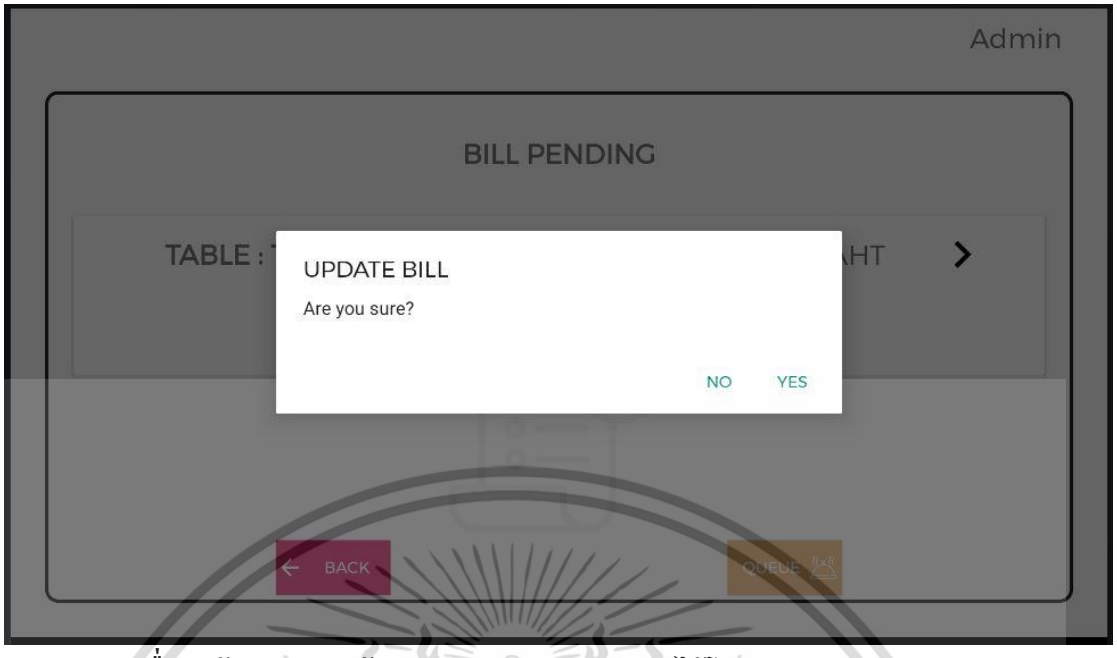
หากต้องการทราบว่าโต๊ะใดเรียกชำระเงิน ให้กดปุ่ม BILL ในหน้านี้จะแสดงรายละเอียดการสั่งอาหาร ราคา และเลขอ้างอิง ของลูกค้าโต๊ะนั้น



รูป 4.25 กดปุ่ม "BILL" เพื่อดูการเรียกชำระเงิน (Waiter)



รูป 4.26 กดปุ่มลูกศรเพื่อแสดงรายละเอียดรายการอาหาร (Waiter)



รูป 4.27 เมื่อลูกค้าชำระเงินแล้ว สามารถทำการอัปเดตบิลได้โดยการกด “CHECK” (Waiter)

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผลที่ได้จากโครงงาน

1) ได้รับความรู้ในการพัฒนาแอพพลิเคชั่นบนระบบปฏิบัติการแอนดรอยด์ (Android OS) โดยใช้ภาษา Kotlin และพัฒนาโดยอ้างอิงจาก Android Architecture Components

2) ทำให้ได้เรียนรู้และเข้าใจการเขียนโปรแกรมทั้งบนฝั่ง Front-End Develop และ Back-End Develop

3) ทำให้เกิดความรู้ความเข้าใจการออกแบบและพัฒนาแอพพลิเคชั่นทั้งระบบ

4) ทำให้เกิดทักษะในการเรียนรู้ด้วยตนเอง เพื่อนำมาใช้การทำงาน

5) ได้ฝึกการทำงานแบบ Agile Development

5.2 ปัญหาและอุปสรรค

เนื่องจากการพัฒนาแอพพลิเคชั่นเป็นการพัฒนาบนระบบปฏิบัติการแอนดรอยด์ ซึ่งผู้จัดทำไม่มีประสบการณ์มาก่อน จึงต้องพยายามเรียนรู้วิธีการใช้งาน ศึกษาภาษา Syntax Library เครื่องมือในการใช้พัฒนา ตลอดจนการติดต่อกับ Web server ในช่วงแรกจึงต้องใช้เวลาไปกับการเรียนรู้จำนวนมาก ส่งผลให้การทำงานเกิดความล่าช้าไปจากแผนที่ได้วางไว้

ในการพัฒนานี้เกินความยากลำบากในการหาข้อมูลเป็นอย่างมาก เนื่องจากเป็นภาษาที่เกิดขึ้นใหม่ ทำให้ยากต่อการหาแหล่งอ้างอิง และกรอบของเวลาที่ไม่เพียงพอต่อการหาข้อมูลเพื่อนำมาประยุกต์ใช้และด้วยเหตุนี้เองทำให้โปรแกรมยังนี้ยังไม่สามารถพัฒนาฟีเจอร์ที่ได้ออกแบบไว้ข้างต้นได้ทั้งหมด

5.3 แนวทางในการพัฒนาต่อ

1) เพิ่มฟีเจอร์ที่ควรจะมีเช่น การนับสต็อกของอาหาร และโปรโมชั่น

2) การทำให้แอพพลิเคชั่นดึงข้อมูลจากเซิฟเวอร์แบบเรียลไทม์ ซึ่งจะทำให้แอพพลิเคชั่นมีประสิทธิ์ภาพเมื่อเพิ่มฟีเจอร์ข้างต้นไปมากขึ้น

3) การปรับปรุง UI ให้มีความน่าดึงดูดมากขึ้น เช่น Animation เมื่อกำลังรอคิว และระบบแจ้งเตือนพนักงานที่ชัดเจน

4) เพิ่มความสะดวกของเจ้าของร้านที่ต้องการดูข้อมูลการสั่งอาหารผ่านทางแอพพลิเคชั่นได้ทันที โดยไม่ต้องเข้าผ่านทางเซิฟเวอร์โดยตรง

5) พัฒนาระบบ Data analytics เพื่อทำการวิเคราะห์ข้อมูลที่ได้จากการสั่งอาหารของลูกค้า เพื่อใช้ในกรณี เช่น ค้นหาช่วงเวลาที่มีลูกค้าเยอะ หรือเมนูที่ได้รับความนิยมในช่วงเวลาใดเวลาหนึ่ง เป็นต้น

บรรณานุกรม

Pichaya Srifar. 2561. **5 ขั้นตอน กระบวนการพัฒนา Mobile Application.** [Online].

Available : https://medium.com/@pichayas/5-ขั้นตอน-กระบวนการพัฒนา-mobile-application-ตั้งแต่ต้นจนจบ-5926634aeabf

The Khaeng. 2560. **ทำไมต้องหันมาใช้ Kotlin พร้อม 「code lab」 ลด learning curve.** [Online].

Available : https://blog.nextzy.me/why-we-use-kotlin-and-code-labs-3b0b02a2ad6b

JetBrains s.r.o.. **Kotlin Standard Library.** [Online].

Available : https://devdocs.io/kotlin/

Kotlin. 2562. **Learn Kotlin.** [Online].

Available : https://kotlinlang.org/docs/reference/

Square, Inc 2563. **Retrofit.** [Online].

Available : https://square.github.io/retrofit/

Androiddevelopers. 2562. **Android Architecture Components.** [Online].

Available : https://developer.android.com/topic/libraries/architecture

Androiddevelopers. 2562. **Fragments.** [Online].

Available : https://developer.android.com/guide/components/fragments

Zell. 2560. **Understanding And Using REST APIs.** [Online].

Available : https://www.smashingmagazine.com/2018/01/understanding-using-rest-api/

Restapi. 2563. **What is REST – Learn to create timeless REST APIs.** [Online].

Available : https://restfulapi.net/