SHARE X

SHARE X

ณัชภัทร สืบบุก
ธรนันท์ ศิริดลธนเกษม

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2560

SHARE X

SHARE X

ณัชภัทร สืบบุก
ธรนันท์ ศิริดลธนเกษม

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2560

ปริญญานิพนธ์ปีการศึกษา 2560

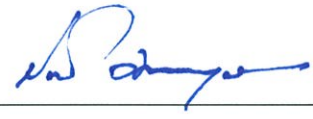
ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง SHARE X

ผู้จัดทำ

1. นายณัชภัทร สืบบุก รหัสนักศึกษา 57010393
2. นายธรนันท์ ศิริดลธนเกษม รหัสนักศึกษา 57010597



_____ อาจารย์ที่ปรึกษา

(ดร.ปกรณ์ วัฒนจตุรพร)

. _____ อาจารย์ที่ปรึกษา

(รศ.ดร.เจริญ วงษ์ชุ่มเย็น)

SHARE X

นายณัชภัทร	สืบบุก	57010393
นายธรนันท์	ศิริดลธนเกษม	57010597
ดร.ปกรณ์	วัฒนจตุรพร	อาจารย์ที่ปรึกษา
รศ.ดร.เจริญ	วงษ์ชุ่มเย็น	อาจารย์ที่ปรึกษาร่วม

ปีการศึกษา 2560

บทคัดย่อ

ในปัจจุบันเราปฏิเสธไม่ได้เลยว่าการเดินทางข้ามอำเภอ, จังหวัด หรือการเดินทางไกลๆนั้นเป็นส่วนหนึ่งของชีวิตเรา ไม่เพียงแต่บุคคุลที่มีอาชีพขนส่งสินค้าเท่านั้นแต่รวมไปถึงคนที่เดินทางไปที่ต่างๆเป็นประจำ โดยจะมีสิ่งหนึ่งที่ตามมาพร้อมการเดินทางเสมอคือ ค่าใช้จ่ายในการเดินทาง ถ้ามองไปถึงรถบรรทุกขนาดใหญ่ที่เดินทางไปส่งสินค้าแล้วต้องกลับรถเปล่านั้นมันยิ่งน่าเสียดายทรัพยากรที่สามารถทำให้เกิดประโยชน์ หรือรายได้ที่เพิ่มขึ้นได้

จากแนวคิดดังกล่าวทางทีมงานผู้พัฒนาแอปพลิเคชัน “ShareX” จึงเล็งเห็นวิธีการที่สามารถหารายได้หรือลดต้นทุนการเดินทางโดยผู้ใช้งานแอปพลิเคชันจะแบ่งออกเป็น 2 กลุ่มคือ 1) ผู้ใช้บริการขนส่งสินค้า จะทำการเลือกประเภทของรถที่ต้องการใช้บริการเลือกจุดรับของและส่งของโดยแอปพลิเคชันจะคำนวณค่าใช้จ่ายให้เลย 2) ผู้ให้บริการขนส่งสินค้า จะทำการเลือกเส้นทางการเดินทาง เมื่อเลือกเส้นทางแล้วและมีคำร้องขอใช้บริการบริเวณใกล้ๆกับเส้นทางที่วางแผนไว้นั้น ก็จะมีแจ้งเตือนให้สามารถรับสินค้าได้

แอปพลิเคชัน ShareX คาดหวังว่าจะเพิ่มการเจริญเติบโตของเศรษฐกิจด้วยการลดต้นทุนการขนส่งในภาพรวมได้

SHARE X

Mr. Nutchaphut	Suebbuk	57010393
Mr. Toranun	Siridontanakasame	57010597
Dr.Pakorn	Watanachaturaporn	Advisor
Assoc.Prof.Dr.Charoen	Vongchumyen	Co-Advisor

Academic Year 2017

ABSTRACT

Presently, travelling across districts, Provinces or further away is part of our life. Not only one with logistic career but other business travelers. Travelling is costly. For a logistic truck, travelling back and forth with empty truck is wasteful and losses profitable opportunity.

From the stated problem, the ShareX application is developed to gain profit or at least reduce cost from travelling. The user are classified to two group. The first group of users is a customer who wants to ship a merchandise. They selec a vehicle type, set a receiving location and set a destination. The application calculates the cost of shipment for the service. The second group is a logistic driver. They select a route of travelling. Request for shipping along the selected travelling route are notified to the driver to choose whether to accept the requests.

The ShareX is expected to increase the economic growth by reducing the shipping cost.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดีด้วยความช่วยเหลือดูแลจากหลายฝ่าย ปริญญานิพนธ์ฉบับนี้จะสำเร็จไม่ได้หากปราศจากความช่วยเหลือจากอาจารย์ที่ปรึกษา ดร.ปกรณ์ วัฒนจตุรพร และอาจารย์ที่ปรึกษาร่วม รศ.ดร.เจริญ วงษ์ชุ่มเย็น ที่ให้โอกาสในการทำโครงงานนี้โดยได้รับคำแนะนำดีๆจากอาจารย์เสมอมาจนสำเร็จลุล่วง ขอขอบคุณอาจารย์และบุคลากรต่าง ๆ ในสาขาวิชาวิศวกรรมคอมพิวเตอร์ที่ได้ให้คำแนะนำและสั่งสอนความรู้ต่าง ๆ มาโดยตลอด รวมถึงห้องวิจัยฮาร์ดแวร์ที่ได้เอื้อเฟื้อสถานที่ในการพัฒนาโครงงาน ขอขอบคุณรุ่นพี่และเพื่อนๆในภาควิชาวิศวกรรมคอมพิวเตอร์ที่ได้ให้คำแนะนำและแบ่งปันความรู้ในหลายๆด้านและให้กำลังใจสู้ไปด้วยกันเสมอมา ที่ขาดไม่ได้เลยคือ ขอขอบคุณ บิดา มารดา และครอบครัวของผู้จัดทำที่ให้การสนับสนุนอย่างเต็มที่ทั้งกำลังใจและกำลังทรัพย์จนโครงงานนี้สำเร็จลุล่วง

ณัชภัทร สืบบุก

ธรนันท์ ศิริดลธนเกษม

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญรูป

รูป หน้า

สารบัญรูป(ต่อ)

รูป หน้า

สารบัญรูป(ต่อ)

รูป หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ในปัจจุบันเทคโนโลยีสารสนเทศนั้นมีบทบาทในการพัฒนาประเทศเป็นอย่างมาก ทั้งในด้านการติดต่อสื่อสารที่สะดวกรวดเร็วไร้พรมแดน การเรียนรู้ที่ไร้ขีดจำกัดผ่านระบบเครือข่าย หรือกระทั่งการช่วยให้ธุรกิจมีประสิทธิภาพมากยิ่งขึ้น เช่น การลดต้นทุน ลดแรงงาน เพิ่มผลกำไรให้ผู้ประกอบการ เป็นต้น

การขนส่งเป็นสิ่งที่จำเป็นสำหรับการดำเนินธุรกิจ โดยผู้ประกอบการทั่วๆไปที่ทำการขนส่งสินค้านั้น เมื่อส่งสินค้าไปยังจุดหมายปลายทางเสร็จเรียบร้อยแล้วนั้น จะวิ่งเที่ยวเปล่ากลับไปยังต้นทางส่งผลให้ต้นทุนการขนส่งเพิ่มขึ้นอีกเท่าตัว เป็นการสูญเสียทรัพยากรโดยไม่ก่อให้เกิดประโยชน์ต่อธุรกิจ

ซึ่งทางคณะผู้จัดทำได้มีความพยายามที่จะทำให้ปัญหาการวิ่งเที่ยวเปล่านั้นลดน้อยลง โดยได้มีแนวคิดที่จะพัฒนา Application ที่ทำการเชื่อมต่อระหว่าง Customer ที่ต้องการจะส่งสินค้ากับ Driver ที่วิ่งเที่ยวเปล่ากลับไปยังต้นทางอยู่แล้วให้สามารถติดต่อใช้บริการกันได้ โดย Customer ทำการประกาศความต้องการที่จะส่งสินค้าไปยังจุดหมายปลายทาง และDriver สามารถเลือกรับสินค้าได้ เพื่อให้ Driver ได้รับประโยชน์สูงสุดเช่นกัน ซึ่งจะเป็นการช่วยแบ่งเบาภาระค่าใช้จ่ายในส่วนที่ต้องเสียไปอยู่แล้วสำหรับค่าเชื้อเพลิงแก่ Driver ได้ สำหรับ Customer นั้นจะได้การบริการส่งสินค้าไปยังเป้าหมายที่มีความสะดวกรวดเร็วยิ่งขึ้นและมีราคาค่าขนส่งที่ถูกกว่าปกติเพื่อเป็นการดึงดูดให้มีผู้ที่สนใจใช้บริการเพิ่มมากขึ้น

1.2 วัตถุประสงค์

1) เพื่อศึกษาการสร้าง Android Mobile Application
2) เพื่อเป็นการเพิ่มช่องทางแก่ผู้ที่ต้องการใช้บริการส่งสินค้า
3) เพื่อเป็นการเพิ่มรายได้ให้แก่ผู้ที่ทำการขนส่งสินค้า โดยเป็นการใช้ทรัพยากรที่มีอยู่แล้วให้มีประสิทธิภาพมากยิ่งขึ้น

1.3 ขอบเขตของโครงงาน

1) จัดทำ Android Mobile Application ที่สามารถใช้งานได้ตามวัตถุประสงค์ที่ได้วางไว้
2) มีระบบบันทึกข้อมูลส่วนตัวของผู้ใช้งาน

3) สามารถเลือกสถานที่บนแผนที่ได้ทั้งโดยการค้นหาจากชื่อ หรือการกำหนดจุดด้วยตนเอง
4) ลูกค้าสามารถทำการประกาศความต้องการส่งสินค้าไปยังเป้าหมายปลางทางได้
5) Driverสามารถเลือกรับสินค้าที่แสดงใน Application ได้
6) มีระบบคำนวณระยะทางในการส่งสินค้าที่เพิ่มขึ้นจากระยะทางเดิม เพื่อเป็นตัวช่วยในการจัดหา Driver ที่เหมาะสมมาให้บริการ
7) มีระบบคำนวณอัตราค่าบริการโดยอ้างอิงจากระยะทางที่ใช้และขนาดของสินค้าได้
8) มีระบบในการตรวจสอบสถานะของสินค้าผ่านระบบ GPS
9) มีระบบจัดการ User ผ่าน Web Application

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) ฝึกการพัฒนาผลิตภัณฑ์ร่วมกันเป็นทีม
2) ได้ทักษะในการจัดทำ Android Mobile Application
3) นำความรู้ที่ได้ศึกษามาประยุกต์ใช้กับผลิตภัณฑ์จริง

1.5 แผนการดำเนินงาน

1) ปรึกษาความเป็นไปได้ของหัวข้อโครงงานกับอาจารย์ที่ปรึกษา
2) กำหนดขอบเขตของโครงงาน
3) ออกแบบระบบการทำงานของ Application
4) ศึกษาการสร้าง Application โดยการใช้ Android studio ในการสร้าง
5) ศึกษาการใช้งาน Google map API ,การใช้งาน mySQL
6) ทำการพัฒนา Applicationตามที่ได้ออกแบบไว้
7) ปรับปรุงแก้ไข และพัฒนา Application ให้มีประสิทธิภาพยิ่งขึ้น
8) สรุปผลการทำโครงงานและจัดทำเอกสาร

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 Android

แอนดรอยด์ (Android) คือ แอปพลิเคชันเฟรมเวิร์ค(Application Framework)ที่สามารถสร้างแอปพลิเคชันและเกมส์สำหรับโทรศัพท์มือถือได้โดยใช้ภาษา Java

แอปพลิเคชันมีจุดเข้าโปรแกรมหลายจุด (Apps provide multiple entry point) โดยในแอพพลิเคชันนั้นถูกสร้างมาจากการรวมกันของส่วนต่างๆของโปรแกรม ตัวอย่างเช่น activity นั้นทำหน้าที่เป็นส่วนต่อประสานกับผู้ใช้ ส่วนserviceจะปฏิบัติการในพื้นหลัง (Background)

2.1.1 แอปพลิเคชันขั้นพื้นฐาน(Application Fundament)

แอนดรอยด์แอปพลิเคชันสามารถเขียนได้โดยใช้ภาษา Kotlin,Java และC++ ใช้ SDK tool ในการคอมไพล์(compile)โปรแกรมที่ได้เขียนไว้และไฟล์ต่างๆให้เป็น APK (Android package) ซึ่งถูกจัดเก็บอยู่ในไฟล์สกุล .apk ไฟล์ APK 1 ไฟล์นั้นจะรวบรวมข้อมูลทั้งหมดในแต่ละแอปพลิเคชัน โดยไฟล์ apk นี้จะสามารถใช้ได้กับอุปกรณ์ Android เท่านั้น ในแต่ละแอนดรอยด์แอปพลิเคชันจะอยู่ในSecurity Sandbox ของมันเอง ซึ่งมีคุณสมบัติดังนี้

1) ระบบปฏิบัติการแอนดรอยด์เป็นระบบลินุกซ์(Linux)แบบผู้ใช้หลายคน ซึ่งในแต่ละแอปพลิเคชันจะมีผู้ใช้งานที่ต่างกัน
2) ในค่าเริ่มต้น ระบบจะกำหนดเลขผู้ใช้ลินุกซ์(Linux user ID)โดยเฉพาะในแต่ละแอพพลิเคชัน ระบบจะตั้งสิทธิในทุกๆไฟล์ในแอปพลิเคชัน มีเพียง user ID ที่กำหนดเท่านั้นสามารถเข้าใช้งานไฟล์ต่างๆได้
3) ในแต่ละโพรเซส(Process) จะมีเครื่องเสมือน(Virtual Machine) เป็นของตัวเอง ทำให้แอปพลิเคชันสามารถทำงานได้โดยแยกจากแอปพลิเคชันอื่นๆอย่างสิ้นเชิง
4) ในค่าเริ่มต้น ทุกๆแอปพลิเคชันจะทำงานใน Linux process ของตัวเอง ระบบแอนดรอยด์จะเริ่มประมวลผลเมื่อส่วนใดส่วนหนึ่งของแอปพลิเคชันถูก execute และปิดการประมวลผลเมื่อไม่ได้ใช้แล้ว หรือ เมื่อระบบต้องการเรียกคืนหน่วยความจำจากแอปพลิเคชันอื่นๆ

ระบบแอนดรอยด์จะใช้หลักการให้สิทธิน้อยที่สุด กล่าวคือ ในแต่ละแอปพลิเคชันนั้นจะสามารถเข้าถึงได้เฉพาะในส่วนที่จำเป็นในการทำงานของมันเท่านั้น ซึ่งเป็นการสร้างความปลอดภัย ในแอปพลิเคชันใดๆจะไม่สามารถเข้าถึงส่วนใดส่วนหนึ่งของระบบได้เลย

ถ้าไม่ได้รับอนุญาต อย่างไรก็ตามยังมีวิธีอื่นอีกในการแบ่งปันข้อมูลกันกับแอปพลิเคชันอื่นๆ โดยสำหรับแอปพลิเคชันที่ต้องเข้าถึงบริการระบบนั้น สามารถให้ 2 แอปพลิเคชันใช้ Linux user ID เดียวกันได้ ซึ่งจะส่งผลให้สามารถเข้าถึงไฟล์ของกันและกันได้ เพื่อรักษาทรัพยากรระบบนั้น แอปพลิเคชันที่มี user ID เดียวกันนั้นสามารถจัดการให้ทำงานได้ใน Linux process เดียวกัน และใช้งาน VM ร่วมกัน โดยแอปพลิเคชันที่จะทำสิ่งเหล่านี้ได้นั้นต้องมีใบรับรอง(certificate) ที่เหมือนกัน แอปพลิเคชันสามารถขอสิทธิการเข้าถึงข้อมูลในเครื่องได้ เช่น รายชื่อติดต่อของผู้ใช้ ,ข้อความ SMS,หน่วยจัดเก็บข้อมูล (SD Card),กล้องและบลูทูธได้ โดยผู้ใช้ต้องอนุญาตสิทธิการเข้าถึงเหล่านี้ก่อนจึงจะใช้งานได้

2.1.2 ส่วนประกอบของแอปพลิเคชัน(App Component)

ส่วนประกอบของแอปพลิเคชันเป็นส่วนสำคัญสำหรับ แอนดรอยด์แอพพลิเคชัน ในแต่ละส่วนจะเป็นจุดเข้าสู่โปรแกรมผ่านทางระบบหรือผู้ใช้เข้าใช้งานแอปพลิเคชัน บางส่วนขึ้นอยู่กับส่วนอื่น โดยส่วนประกอบของแอปพลิเคชันมี 4ชนิด ซึ่งในแต่ละชนิดนั้นจะทำงานแตกต่างกันออกไปและมีช่วงวงจรชีวิตที่กำหนดการสร้างและทำลายของส่วนประกอบต่างๆ ที่แตกต่างกัน ส่วนประกอบของแอปพลิเคชันมีดังนี้

2.1.2.1 แอคทิวิตี้(Activities)

Activity คือจุดเข้าสู่โปรแกรมโดยมีการโต้ตอบกับผู้ใช้ โดยแสดงผลส่วนต่อประสานกับผู้ใช้บนหน้าจอ ตัวอย่างเช่น อีเมลล์แอปพลิเคชัน(Email App) อาจมีแอคทิวิตี้หนึ่งสำหรับแสดงรายการจดหมายมาใหม่ แอคทิวิตี้หนึ่งไว้สำหรับอ่านอีเมลล์ อีกแอคทิวิตี้หนึ่งไว้สำหรับเขียนอีเมลล์ แม้ว่าในแต่ละแอคทิวิตี้จะทำงานด้วยกันเพื่อสร้าง User experience ในอีเมลล์แอปพลิเคชันที่ติดต่อกับ แต่ในแต่ละแอคทิวิตี้ก็แยกออกจากกันอย่างสิ้นเชิง เมื่อเป็นเช่นนี้แอปพลิเคชันอื่นๆก็สามารถเริ่มแอคทิวิตี้ใดก็ได้ถ้า อีเมลล์แอปพลิเคชันนั้นอนุญาต ตัวอย่างเช่น แอปพลิเคชันกล้องสามารถเริ่มแอคทิวิตี้เขียนอีเมลล์ในอีเมลล์แอปพลิเคชัน เพื่อให้ผู้ใช้แบ่งปันรูปถ่าย แอคทิวิตี้ที่เอื้อต่อการมีปฏิสัมพันธ์ระหว่างระบบกับแอปพลิเคชัน

1) ติดตามกับสิ่งที่ผู้ใช้สนใจในปัจจุบัน(ที่แสดงบนหน้าจอ) เพื่อให้มั่นใจว่าระบบนั้นประมวลผลที่โฮสติ้ง(hosting)แอคทิวิตี้นั้นอยู่
2) รู้ว่าการประมวลผลที่ใช้ก่อนหน้านั้นอาจจะมีสิ่งที่ผู้ใช้กลับมาใช้ ดังนั้นจึงสำคัญมากที่ต้องเก็บการประมวลผลนั้นไว้เสมอ
3) ช่วยจัดการแอปพลิเคชันที่การประมวลผลถูกกำจัด (Process killed) ทำให้ผู้ใช้สามารถกลับมายังแอคทิวิตี้ที่มีสถานะเดิมกลับคืนมาได้

4) ให้แนวทางแอปพลิเคชันในการจัดการกระแสผู้ใช้ (User flow) ระหว่างกัน เพื่อให้ระบบประสานงานกับกระแสผู้ใช้เหล่านี้

2.1.2.2 เซอร์วิส(Services)

เซอร์วิสเป็นจุดเริ่มต้นโปรแกรมที่ทำงานในพื้นหลัง(background) ที่ดำเนินการในระยะยาว หรือเพื่อทำการประมวลผลในทางไกล(Remote) เซอร์วิสนี้จะไม่มีส่วนติดต่อกับผู้ใช้ ตัวอย่างเช่น เซอร์วิสนี้อาจจะเล่นเพลงในพื้นหลัง(Background) ในขณะที่ผู้ใช้นั้นใช้งานแอปพลิเคชันอื่นๆอยู่ หรือทำการเรียกข้อมูลผ่านระบบเครือข่ายโดยไม่ไปกีดขวางกับส่วนปฏิสัมพันธ์กับผู้ใช้(User Interaction)ในแอคทิวิตี้ ในส่วนประกอบ(Component)อื่นๆ เช่น แอคทิวิตี้(Activity) สามารถเริ่มเซอร์วิสแล้วปล่อยให้มันทำงานเอง หรือเรียกใช้ก็ได้ถ้าต้องการโต้ตอบกับมัน

การเริ่มเซอร์วิสเพื่อบอกระบบให้ทำงานไปจนกว่างานจะเสร็จสิ้น เช่น การซิงค์(Sync)ข้อมูล หรือเล่นเพลงไปเรื่อยๆจนกว่าผู้ใช้จะปิดแอปพลิเคชัน เป็นการแทนชนิดของการเริ่มเซอร์วิสที่ปรับเปลี่ยนวิธีการที่ระบบจัดการกับมัน 2 ชนิดที่แตกต่างกัน

1) การเล่นเพลงบางครั้งผู้ใช้ก็จดจ่อหรือให้ความสำคัญกับมัน ดังนั้นแอพพลิเคชันต้องบอกกับระบบว่าต้องทำงานในด้านหน้า(foreground) ที่มีการแจ้งเตือนให้ผู้ใช้ด้วย ในเคสนี้ระบบจะรู้ว่าต้องทำทุกวิถีทางที่จะทำให้การประมวลผลเซอร์วิสนี้ทำงานอยู่ตลอดมิเช่นนั้นผู้ใช้ต้องโกรธแน่ถ้ามันดับขึ้นมา
2) เซอร์วิสที่ทำงานในพื้นหลัง(background)ทั่วๆไปนั้น ไม่ใช่สิ่งที่ผู้ใช้นั้นสนใจในขณะที่มันกำลังทำงานอยู่ ระบบจะมีอิสระในการจัดการกับการประมวลผลเหล่านี้ อาจจะให้หยุดการทำงาน(อาจจะเริ่มการทำงานใหม่ในภายหลังได้)ถ้าระบบต้องการRAMในการใช้กับสิ่งที่มีความต้องการแก่ผู้ใช้มากกว่า

2.1.2.3 บรอดคาสท์ รีซีฟเวอร์ (Broadcast receivers)

เป็นส่วนที่ทำให้ระบบสามารถส่ง broadcast event ไปยังแอปพลิเคชันได้แม้ว่าไม่ได้ทำงานอยู่ ตัวอย่างเช่น แอปพลิเคชันสามารถกำหนดเวลาเตือนเพื่อแจ้งeventที่จะเกิดขึ้นแก่ผู้ใช้และการส่งการแจ้งเตือนไปยังแอปพลิเคชันนั้น ไม่จำเป็นที่แอปพลิเคชันต้องทำงานอยู่ตลอดจนกว่าสัญญาณแจ้งเตือนจะหมดไป Broadcast หลายๆอันมาจากระบบ เช่น ประกาศว่าทำการปิดจอ แจ้งเตือนแบตเตอรี่ต่ำ หรือแจ้งเตือนการจับภาพหน้าจอ และแอปพลิเคชันสามารถเริ่ม broadcast ได้ เช่น แจ้งให้แอปพลิเคชันอื่นๆทราบว่ามีข้อมูลถูกดาวน์โหลดมายังเครื่องนี้และ

สามารถเรียกใช้ได้ ถึงแม้ว่า Broadcast Receiver นั้นจะไม่แสดงในหน้าส่วนต่อประสานกับผู้ใช้ แต่อาจจะสร้าง Status Bar ขึ้นมาเพื่อแสดงการแจ้งเตือนเมื่อเกิดเหตุการณ์ใดๆขึ้น

2.1.2.4 คอนเทนต์โพรไวเดอร์ (Content providers)

เป็นส่วนจัดการกลุ่มข้อมูลที่สามารถจัดเก็บไว้ในระบบ,ในฐานข้อมูล, บนเว็บ หรือแหล่งจัดเก็บข้อมูลไหนก็ตามที่แอปพลิเคชันสามารถเข้าถึงได้ แอปพลิเคชันอื่นๆ สามารถ Query หรือเปลี่ยนแปลงข้อมูลได้ถ้าได้รับอนุญาต ตัวอย่างเช่น ระบบแอนดรอยด์ให้ Content Provider จัดการรายชื่อผู้ติดต่อของผู้ใช้ ดังนั้นแอปพลิเคชันใดๆที่ได้รับอนุญาตให้เข้าถึงข้อมูลแล้ว สามารถเรียกใช้ข้อมูลได้ ประมาณว่า ContactsContract.Data เป็นต้น เพื่ออ่านและเขียนข้อมูลสำหรับเฉพาะบุคคล

แอปพลิเคชันสามารถปรับได้กับในอุปกรณ์ที่แตกต่างกัน(Apps adapt to different devices) แอนดรอยด์มีแอปพลิเคชันเฟรมเวิร์คที่ทำให้สามารถกำหนดทรัพยากรและการตั้งค่าให้ในแต่ละอุปกรณ์ที่ต่างๆกัน ตัวอย่างเช่น สามารถสร้าง XML Layout file ที่ต่างๆกันสำหรับแต่ละขนาดหน้าจอที่ต่างกันและระบบจะเลือก Layout ที่เหมาะสมโดยขึ้นอยู่กับขนาดหน้าจอให้แต่ละอุปกรณ์

2.1.3 Intents

Intent คือ Object ข้อความที่สามารถร้องขอการกระทำต่างๆ (Action) จาก App Component อื่นๆได้ ถึงแม้ว่า Intent นั้นจะอำนวยความสะดวกในการติดต่อสื่อสารระหว่าง Componentในหลายๆด้าน แต่ก็มีการใช้งานพื้นฐาน 3 ชนิด คือ

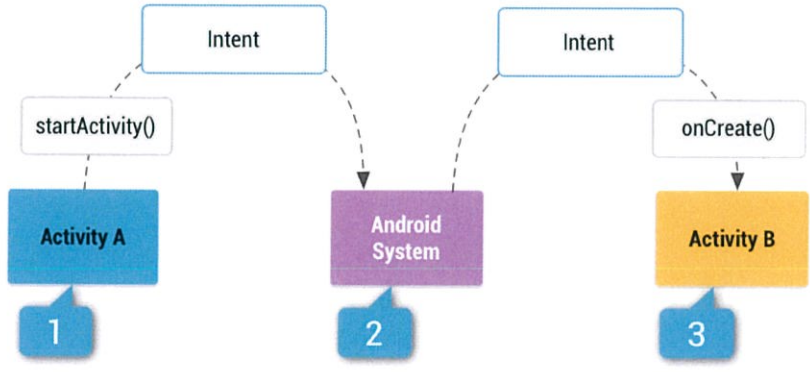
1) เพื่อเริ่มแอคทิวิตี้(Activity)ใหม่ที่เป็นหน้าๆหนึ่งในแอปพลิเคชัน โดยทำการ pass ค่า Intent ให้ทำการstartActivity() โดยภายใน Intentนั้นจะมีให้เริ่มActivityใหม่และข้อมูลที่จำเป็น ถ้าต้องการรับผลลัพธ์จากแอคทิวิตี้เมื่อทำงานเสร็จ เรียก startActivityForResult() แอคทิวิตี้ที่เป็นตัวรับจะรับผลมาเป็น Intent ใน onActivityResult() callback

2) เพื่อเริ่มเซอร์วิส(Service) โดยเซอร์วิสนั้นปฏิบัติงานในพื้นหลังไม่มีส่วนต่อ-ประสานกับผู้ใช้ ด้วยAPI level 21+ จะเริ่มเซอร์วิสได้ด้วย JobScheduler แต่ถ้าAPI เวอร์ชันก่อนหน้าจะเริ่มเซอร์วิสได้โดยเรียก method จากคลาสเซอร์วิส โดยสามารถสั่งให้ทำงานแบบการทำงานครั้งเดียว(อย่างการดาวน์โหลดไฟล์) โดยเรียก startService() ซึ่งภายในจะมีชื่อเซอร์วิสที่จะเริ่มและข้อมูลที่จำเป็น ถ้าเซอร์วิสนั้นเป็นแบบ Client-Server interface จะเรียก bindService()แทน

3) ส่งบรอดคาสต์(Broadcast) Broadcast คือข้อความที่ทุกแอปพลิเคชันสามารถรับได้ ระบบจะส่ง Broadcastต่างๆสำหรับเหตุการณ์ในระบบ อย่างเช่น เมื่อระบบเริ่มทำงาน หรือทำการชาร์จอุปกรณ์ โดยเราสามารถส่ง Broadcast ไปยังแอปพลิเคชันอื่นๆได้โดยใช้ Intent sentBroadcast( ) หรือ sendOrderedBroadcast().

มีชนิดของ Intent ดังนี้

1) Explicit intent จะกำหนดชื่อของ Component ที่จะเริ่ม โดยปกติแล้วจะใช้ Explicit intent เพื่อเริ่ม Component ในแอปพลิเคชันของตัวเอง เพราะเรารู้ชื่อคลาสของแอคทิวิตี้หรือเซอร์วิสที่ต้องการจะให้เริ่ม ตัวอย่างเช่น เมื่อผู้ใช้กระทำสิ่งใดสิ่งหนึ่งจะเริ่ม Activityเพื่อเป็นการตอบกลับ หรือ เริ่ม Service เพื่อดาวน์โหลดไฟล์ในพื้นหลัง เป็นต้น
2) Implicit Intent จะไม่กำหนดเจาะจงชื่อ Component แต่จะประกาศการกระทำเพื่อดำเนินการ ซึ่งจะอนุญาต Componentจากแอปพลิเคชันอื่นเพื่อจัดการด้วย เช่น ต้องการแสดงตำแหน่งปัจจุบันของผู้ใช้บนแผนที่ สามารถใช้ Implicit Intent เพื่อร้องขอจากแอปพลิเคชันอื่นที่สามารถแสดงตำแหน่งที่กำหนดได้บนแผนที่



รูปที่ 2.1 แสดงการทำงานของ Intent

รูปแสดงการทำงานของ Intent เมื่อ Activity A สร้าง Intent แล้วส่งพร้อมข้อมูลไปยัง startActivity() จากนั้นระบบแอนดรอยด์จะค้นหาทุกแอปพลิเคชันเพื่อจับคู่กับIntent filters ที่ตรงกัน เมื่อค้นหาพบแล้วจะเริ่ม Activity นั้นโดยเรียก onCreate() พร้อมทั้งส่งข้อมูล Intent ไปยัง Activity นั้นด้วย

เมื่อสร้าง Implicit intent ระบบแอนดรอยด์จะค้นหา Component ที่เหมาะสมเพื่อเริ่มทำงานโดยเปรียบเทียบกับเนื้อหาของ Intent กับ Intent filters ที่ประกาศไว้ในไฟล์ Manifest

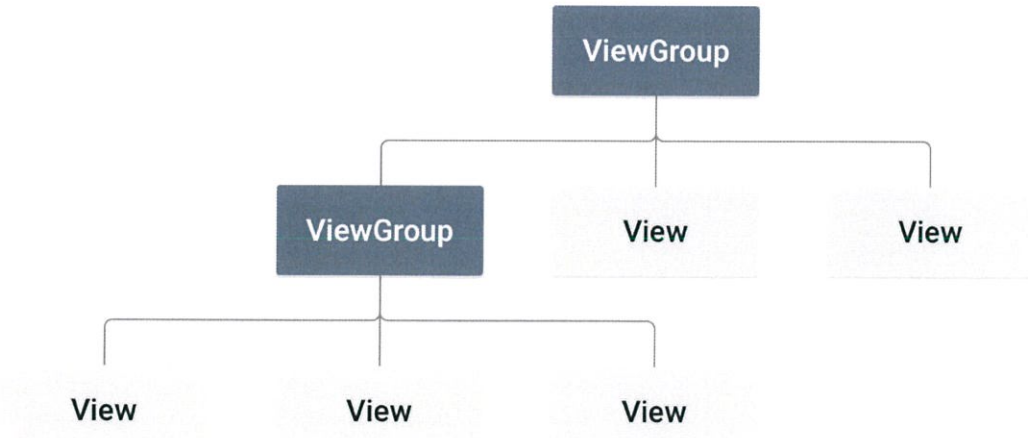
ของแอปพลิเคชันอื่นในอุปกรณ์ ถ้าค้นหาพบ ระบบจะเริ่ม Component และส่ง Intent Object แต่ถ้าพบ Intent filters หลายอันที่เข้ากันได้ ระบบจะแสดงหน้าแอปพลิเคชันทั้งหมดให้ผู้ใช้เลือกเพื่อใช้งาน

2.1.4 ภาพรวมของส่วนต่อประสานกับผู้ใช้(UI Overview)

Element ของส่วนต่อประสานกับผู้ใช้ทั้งหมดในแอนดรอยด์แอปพลิเคชั้นสร้างขึ้นโดยใช้ View และ Viewgroup objects โดย View นั้นเป็น object ที่แสดงผลบนหน้าจอแล้วผู้ใช้สามารถโต้ตอบได้ Viewgroup คือ object ที่รวม View(และ ViewGroup) object อื่นๆ เพื่อกำหนดเป็น Layout ของหน้าอินเตอร์เฟส(interface) แอนดรอยด์จะมีชุดของทั้ง View และ ViewGroup subclasses ที่เป็นส่วนควบคุมอินพุต(เช่น ปุ่ม และ ช่องข้อความ) และ Layout model อื่นๆ เช่น LinearLayout หรือ RelativeLayout

2.1.4.1 รูปแบบของส่วนต่อประสานผู้ใช้(User Interface Layout)

ส่วนต่อประสานผู้ใช้ในแต่ละส่วนของแอปพลิเคชันนั้นกำหนดลำดับขั้นของ View และ ViewGroup object ในแต่ละ ViewGroup จะเปรียบเสมือนกล่องล่องหนที่จัดระเบียบกับ child View ในขณะที่ view นั้นอาจจะเป็นส่วนควบคุมอินพุต(Input) หรือวิดเจ็ท (Widget)อื่นๆที่แสดงในบางส่วนบนส่วนต่อประสานผู้ใช้ต้นไม้ลำดับขั้น(Hierarchy Tree) อาจจะมีทั้งเรียบง่ายและซับซ้อนตามที่เราต้องการได้(แต่เรียบง่ายที่สุดจะมีประสิทธิภาพดีที่สุด)



รูปที่ 2.2 แผนผัง ViewGroup

ในการประกาศ Layoutนั้นวิธีที่ง่ายที่สุดและมีประสิทธิภาพมากที่สุดคือการกำหนด Layout ด้วย XML ซึ่ง ไฟล์XML จะเป็นโครงสร้างที่มนุษย์สามารถอ่านแล้วเข้าใจได้ สำหรับ Layout มีความคล้ายคลึงกับ HTML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              android:layout_width="fill_parent"
              android:layout_height="fill_parent"
              android:orientation="vertical" >
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="I am a TextView" />
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="I am a Button" />
</LinearLayout>
```

รูปที่ 2.3 แสดง Verticle Layout อย่างง่าย โดยมี Text View และ ปุ่ม

2.2 Java

Java เป็นภาษาการเขียนโปรแกรมคอมพิวเตอร์ทั่วไปที่ทำงานพร้อมกัน (Concurrent) เป็นแบบคลาส (Class base) และเป็นแบบเชิงวัตถุ (Object-Orient) โดยมีจุดมุ่งหมายเพื่อให้นักพัฒนาแอปพลิเคชัน "เขียนครั้งเดียวทำงานได้ทุกที่" (Write Once,Run anywhere) ซึ่งหมายความว่าโค้ด Java ที่คอมไพล์เสร็จแล้วสามารถเรียกใช้บนแพลตฟอร์มทั้งหมดที่สนับสนุน Javaได้โดยไม่ต้องคอมไพล์ใหม่อีก Java Application โดยทั่วไปจะถูกคอมไพล์เป็น bytecode ที่สามารถรันบนเครื่องเสมือน Java (JVM) โดยไม่คำนึงถึงสถาปัตยกรรมของคอมพิวเตอร์ ในปีพ.ศ. 2559 Java เป็นภาษาโปรแกรมที่ได้รับความนิยมมากที่สุดแห่งหนึ่งในโลกโดยเฉพาะอย่างยิ่งสำหรับเว็บแอปพลิเคชันแบบไคลแอนท์-เซิร์ฟเวอร์ (Client-Server) โดยJavaนั้นถูกพัฒนาโดย James Gosling ที่บริษัท Sun Microsystems และถูกเผยแพร่ในปีพ. ศ. 2538

ไวยากรณ์ของ Java ส่วนใหญ่ได้รับอิทธิพลมาจาก C++ แต่แตกต่างกับ C++ ตรงที่รวมไวยากรณ์สำหรับโครงสร้าง(Syntax for Structure) และการเป็นการเขียนโปรแกรมแบบเชิงวัตถุ โดยJava ถูกสร้างขึ้นเกือบเป็นภาษาเชิงวัตถุ โค้ดทั้งหมดถูกเขียนขึ้นภายในคลาสและข้อมูลเป็นอ็อบเจ็กต์ทั้งหมดยกเว้นชนิดข้อมูลดั้งเดิม (ได้แก่ Integer,Floating-point numbers,boolean และ character) และJavaยังนำบางฟังก์ชันของ C++ที่นิยมมาใช้อีกด้วย เช่น printf () เป็นต้น

รูปที่ 2.4 สัญลักษณ์ของ Java

ข้อดีของภาษา Java

1) เป็นภาษาที่มาจากภาษา C มีการเขียนที่คล้ายกันกับในตระกูลภาษา C อื่นๆด้วย เช่น C++, C# เป็นต้น ถ้าหากสามารถเขียนภาษาข้างต้นได้ การเรียนรู้การเขียนภาษา Java ก็สามารถทำได้ง่าย
2) มี Library ให้เลือกใช้มากมาย ทั้งจากที่มาจากผู้พัฒนาJava เองและจาก Library ของคนอื่นๆซึ่งอาจจะมีประสิทธิภาพไกล้เคียงกับที่ปล่อยมาจาก Javaเอง ก็สามารถนำมาใช้ได้เช่นกัน
3) ไวยกรณ์นั้นเรียบง่ายกว่าภาษาในตระกูลภาษา C ซึ่งเป็นจุดเริ่มต้นที่ดีสำหรับการเริ่มต้นในการเขียนโปรแกรมก่อนที่จะเขียนภาษาอื่นๆในอนาคต
4) เปิดให้ใช้งานได้ฟรี

2.3 Android Studio

Android Studio เป็น IDE (Integrate Development Environemnt) อย่างเป็นทางการสำหรับการพัฒนาแอนดรอยด์แอปพลิเคชันและ Android Studio ก็ยังเพิ่มคุณสมบัติพิเศษต่างๆ เพื่อเพิ่มประสิทธิภาพในการสร้างแอนดรอยด์แอปพลิเคชัน เช่น

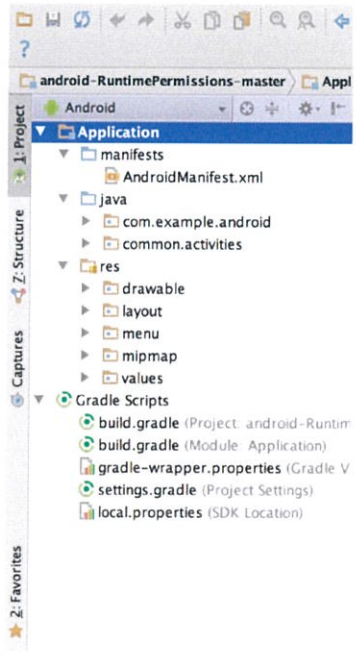
1) จัดทำ Android Mobile Application ที่สามารถใช้งานได้ตามวัตถุประสงค์ที่ได้วางไว้
2) มีอุปกรณ์เสมือน(Emulator)ที่รวดเร็วและมีลูกเล่นที่มีประโยชน์
3) มีสิ่งแวดล้อม(Environment)แบบครบวงจรสำหรับการพัฒนาในทุกอุปกรณ์แอนดรอยด์
4) สามารถเริ่มทำงานโปรแกรมที่เปลี่ยนแปลงจากเดิมได้ทันทีโดยไม่ต้องสร้าง APK ไฟล์ใหม่
5) มีโค้ดแม่แบบ(Code Template) เพื่อช่วยในการสร้างแอปพลิเคชันที่มีคุณสมบัติ(Features)คล้ายๆกัน

6) รองรับภาษา C++ และ NDK
7) มีเครื่องมือที่เรียกว่า Lint ในการตรวจประสิทธิภาพ การใช้งาน ความเข้ากันได้กับรุ่นต่างๆ และปัญหาอื่นๆ
8) มีเครื่องมือที่รองรับ Google Cloud Platform ในตัว ทำให้ง่ายสำหรับการทำงานร่วมกับ Google Cloud Message และ App Engine

2.3.1 Project Structure

แต่ละโปรเจคใน Android Studio นั้นจะมีตั้งแต่ 1 Modules ขึ้นไป รวมไปถึงไฟล์ Source Code และไฟล์ Resource ชนิดของ Module มีดังนี้

1) Module แอนดรอยด์แอปพลิเคชัน
2) Module Library
3) Module Google App Engine



รูปที่ 2.5 โครงสร้างไฟล์ของโปรเจค

โดยปกติ Android Studio จะแสดงไฟล์โปรเจคดังภาพ รูปแบบนี้จะช่วยให้เข้าถึงไฟล์ต่างๆใน Source File ได้ง่ายขึ้น มีส่วนประกอบดังนี้

1) Manifest ส่วนเก็บไฟล์AndroidManifest.xml
2) Java ส่วนรวบรวมไฟล์ Source Code และไฟล์ Testต่างๆ
3) Res ส่วนรวบรวมไฟล์อื่นๆ เช่นไฟล์ XML Layout,String และไฟล์รูป Bitmap

นอกจากนี้ยังสามารถปรับแต่งมุมมองของไฟล์โปรเจคได้เพื่อมุ่งไปยังส่วนที่ต้องการสำหรับการพัฒนาแอปพลิเคชันได้ ตัวอย่างเช่น เลือกมุมมอง Problems จะแสดงส่วนเชื่อมต่อไปยังไฟล์ที่มีข้อผิดพลาด(Syntax Error) เช่น แทกปิด(</...>)ในไฟล์XML layout หายไป เป็นต้น

2.4 Git

Git คือ Version Control แบบ Distributed ตัวหนึ่ง เป็นระบบที่ใช้จัดเก็บและควบคุมการเปลี่ยนแปลงที่เกิดขึ้นกับไฟล์ชนิดใดก็ได้ ไม่ว่าจะเป็น Text File หรือ Binary File (หรือเรียกรวมกันว่า Source Code)

Track version ของ Source Code ย้อนกลับได้เมื่อจัดเก็บไฟล์เข้าไปในระบบของ Git จะเรียกว่า Git Repository ซึ่งเก็บสำรองข้อมูลและการเปลี่ยนแปลงของ Source Code ทำให้สามารถย้อนกลับไปที่เวอร์ชั่นใดๆ ก่อนหน้า และดูรายละเอียดการเปลี่ยนแปลงของแต่ละเวอร์ชั่นได้ นอกจากนั้นยังสามารถดูได้ว่าใครเป็นคนแก้ไข

ช่วยในการพัฒนาซอฟต์แวร์เป็นทีมGit สามารถเก็บบันทึกการเปลี่ยนแปลงของ Source Code เวอร์ชั่นล่าสุดไว้ที่ Local Repository ซึ่งสามารถทำงานได้โดยที่ไม่ต้องต่อกับอินเตอร์เน็ต และเมื่อต้อง Update การเปลี่ยนแปลงของ Source Code เวอร์ชั่นล่าสุดให้กับเพื่อนร่วมทีมก็สามารถที่จะ Push ขึ้นไปเก็บที่ Remote Repository(Git Hosting) และเพื่อนร่วมทีมก็สามารถ Pull เวอร์ชั้นล่าสุดนั้นมารวม(Auto Merge) ที่เครื่องของเขาเอง ทำให้ Source Code ที่พัฒนาร่วมกันกับคนภายในทีมเป็นเวอร์ชั่นล่าสุดเสมอ

2.4.1 Git Status

สถานะของ Source Code ที่เก็บอยู่ในระบบของ Git นั้นมีดังนี้

1) Untracked เป็นสถานะที่ Source Code ถูกเพิ่มเข้ามาใหม่และยังไม่ได้ถูกเก็บไว้ในระบบของ Git
2) Working Directory เป็นสถานะที่กำลังมีการเปลี่ยนแปลงหรือแก้ไข Source Code หรืออาจจะเรียกสถานะนี้ว่า Modified
3) Staged เป็นสถานะที่ Source Code กำลังเตรียมที่จะ Commit เพื่อยืนยันการเปลี่ยนแปลงก่อนที่จะเก็บลงในสถานะ Local Repository
4) Local Repository เป็นสถานะที่มีการเก็บบันทึกข้อมูลการเปลี่ยนแปลงของ Source Code ลงไปที่ Git Repository ที่เป็น Local (ที่เครื่องตัวเอง)
5) Remote Repository เป็นสถานะที่มีการเก็บบันทึกข้อมูลการเปลี่ยนแปลงของ Source Code ลงไปที่ Git Repository ที่เป็น Hosting (ที่เครื่องเซิร์ฟเวอร์)

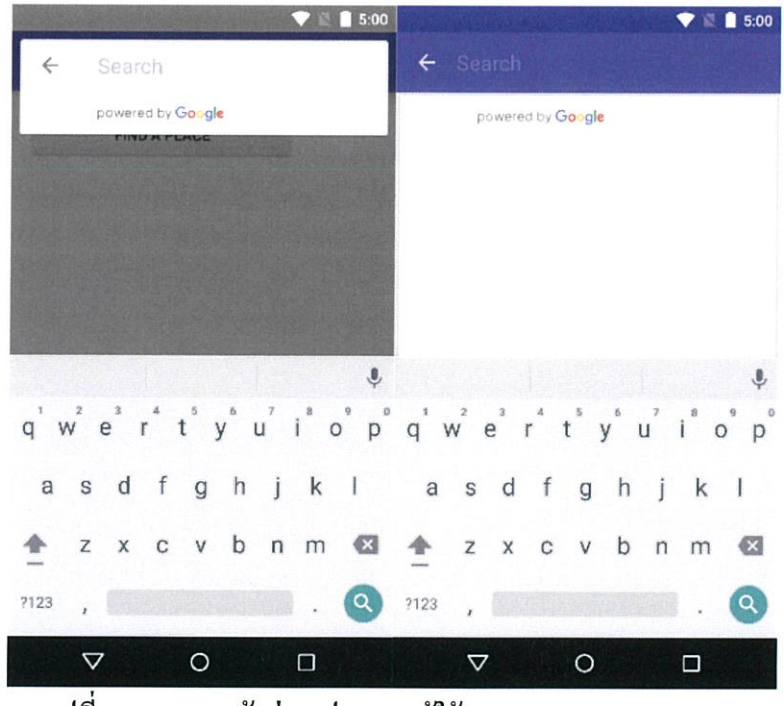
2.5 Google Maps Android API

ด้วย Google Map Android API จะสามารถเพิ่มแผนที่ที่มาจาก Google Map ได้โดยตรงในแอปพลิเคชัน โดย API จะจัดการการเข้าถึงเซิร์ฟเวอร์ของGoogle Map อย่างอัตโนมัติ การดาวน์โหลดข้อมูล การแสดงแผนที่ และการตอบสนองกับการกระทำกับ(Gesture)แผนที่ และยังสามารถใช้ API เพื่อเพิ่ม Graphics ต่างๆลงในแผนที่ได้

1) ไอคอนที่ระบุตำแหน่งบนแผนที่ (Marker)
2) ชุดในส่วนของเส้น (Polylines)
3) ส่วนที่ล้อมกรอบ (Polygons)
4) ภาพกราฟิกบิตแมพ (Bitmap Graphics) ที่วางอยู่ ณ ตำแหน่งหนึ่งๆบนแผนที่ (Ground Overlays)
5) ชุดของภาพที่แสดงบนภาพที่แบ่งเป็นตาราง (Tile Overlays)

2.6 Place Autocomplete

เป็น Google Place API ที่ช่วยในการคาดเดาชื่อสถานที่/ที่อยู่ ในช่องที่ผู้ใช้ค้นหา เมื่อผู้ใช้พิมพ์ชื่อสถานที่ API นี้จะเสนอแนะสถานที่ให้ เช่น ย่านธุรกิจ ที่อยู่ และจุดสนใจต่างๆ โดยสามารถเรียกใช้วิดเจ็ต(Widget)ที่เป็นช่องค้นหาที่มาพร้อมกับฟังก์ชันเหล่านั้น โดยจะแสดงรายชื่อสถานที่แล้วเมื่อผู้ใช้เลือก จะทำการ return ค่า Placeinstance ซึ่งสามารถนำค่านี้มาใช้ในแอปพลิเคชันได้



รูปที่ 2.6 แสดงหน้าส่วนประสานผู้ใช้ของ Autocomplete

จะมีการใช้งานเป็น 2 แบบ ดังนี้

1) ฝัง PlaceAutocompleteFragment ในแอปพลิเคชัน วิธีนี้นั้นจะเป็นการเรียกใช้ API มาใช้งานใช้ส่วนต่อประสานผู้ใช้ (User Interface)ตามที่ API ให้มา
2) ใช้ Intent เพื่อเรียกใช้แอคทิวิตี้ Autocomplete วิธีนี้ไม่ต้องฝัง Fragment ในแอปพลิเคชัน สามารถปรับแต่งส่วนต่อประสานผู้ใช้ได้ง่าย เลือกโหมดการแสดงผลได้

2.7 Google Maps Directions API

เป็นเซอร์วิส API ที่คำนวณเส้นทางระหว่างสถานที่โดยใช้ HTTP request โดยสามารถค้นหาเส้นทางได้ในหลากหลายโหมด เช่น การใช้ขนส่งสาธารณะ การขับรถ การเดิน หรือแม้กระทั่งการปั่นจักรยาน

กำหนดจุดเริ่มต้น จุดหมายปลายทางและจุดระหว่างทาง(Waypoint) ในรูปของ string(เช่น "Chicago,IL" หรือ "Darwin,NT,Australia") หรือในรูปของพิกัดละติจูด,ลองติจูด หรือด้วย IDของแต่ละสถานที่

โดย APIนี้เมื่อคำนวณเส้นทางแล้วจะส่งเส้นทางที่ดีที่สุด โดยนับจากเวลาการเดินทางเป็นตัวหลัก แต่ API สามารถใช้ระยะทาง,จำนวนการเลี้ยว หรืออื่นๆเพื่อเป็นตัวหลักในการวัดเส้นทางที่ดีที่สุดได้เช่นกัน

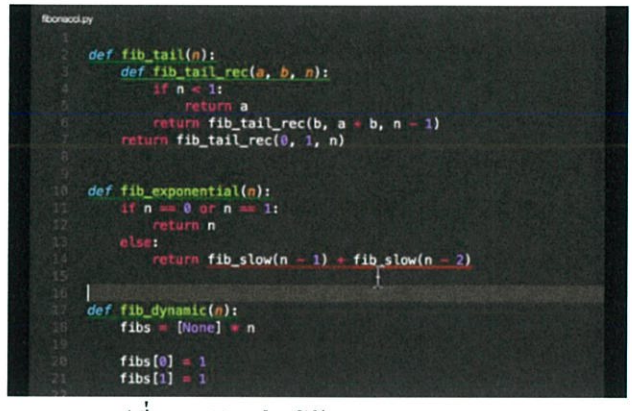
2.8 Python

ภาษา Python ถูกพัฒนาขึ้นในสถาบันวิจัยทางด้านคณิตศาสตร์และคอมพิวเตอร์ ประเทศเน เทอร์แลนด์ โดย Guido van Rossum ในช่วงปี ค.ศ. 1980-1990 ภาษา Python นั้นถูกพัฒนามาจากหลายๆภาษา เช่น C,C++ และภาษา script ต่างๆ เป็นต้น

รูปที่ 2.7 สัญลักษณ์ของ Python

ภาษา Python การประมวนผลเป็นแบบ Interpreter เป็นภาษาแบบ High level และ Object-oriented ซึ่ง Python นั้นถูกออกแบบมาให้อ่านได้ง่ายแม้ว่าผู้ที่อ่านยังไม่มีความรู้ภาษา Python

มาก่อนก็ตาม โดยจะใช้ศัพท์ภาษาอังกฤษที่ใช้กันบ่อยๆมาใช้ และยังมีส่วนของไวยกรณ์ที่น้อยกว่าภาษาอื่นๆ



รูปที่ 2.8 ตัวอย่างโค้ดของ Python

คุณลักษณะเด่นของภาษา Python

1) มี keyword ที่ต้องจดจำไม่มาก โครงสร้างภาษาแบบเรียบง่าย จึงเรียนรู้ได้ง่าย
2) เนื่องจากมีโครงสร้างภาษาที่เรียบง่าย สามารถอ่านได้ง่าย จึงปรับปรุงโค้ดได้ง่าย
3) มี Library ให้เลือกใช้จำนวนมากรวมไปถึงสามารถใช้งาน Library ข้ามแพลตฟอร์มได้
4) Python ประมวลผลตอน runtime โดย Interpreter จึงไม่จำเป็นต้อง Compile ก่อนใช้งาน
5) รองรับการเขียนโปรแกรม Object-Oriented
6) รองรับการทำ Garbage collection แบบอัตโนมัติ

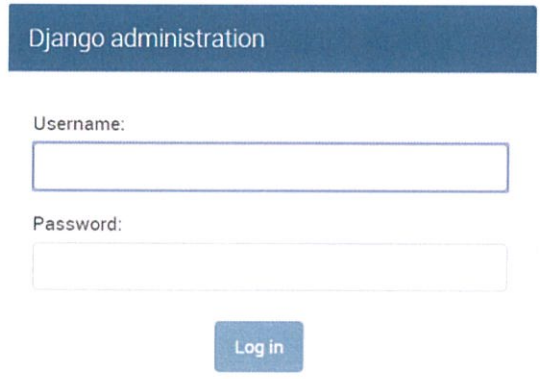
2.9 Django

Django (อ่านว่าจังโก้ หรือแจงโก้ โดยไม่ออกเสียงตัว D) เป็น framework ที่ใช้ในการสร้าง Web Application ในฝั่งของ Back End ที่พัฒนาด้วยภาษา Python โดยในตัว framework จะมีส่วนประกอบทุกอย่างที่จำเป็นตั้งแต่การเชื่อมต่อฐานข้อมูล ไปจนถึงการ render ข้อมูลออกมาให้ฝั่ง Front End แสดงผลข้อมูลเหล่านั้นได้ ซึ่ง framework ในรูปแบบนี้ในภาษาอื่นๆ เช่น Ruby on rails สำหรับภาษา Ruby, Play Framework สำหรับภาษา Java หรือ Scala, Groovy on Grails สำหรับภาษา Groovy, Laravel สำหรับภาษา PHP, หรือ Express สำหรับภาษา Javascript ของ Node.js เป็นต้น ซึ่งข้อมูลที่อ้างอิงมาจาก www.hotframeworks.com จะเห็นว่า Django มีการใช้งานอย่างแพร่หลายมาก

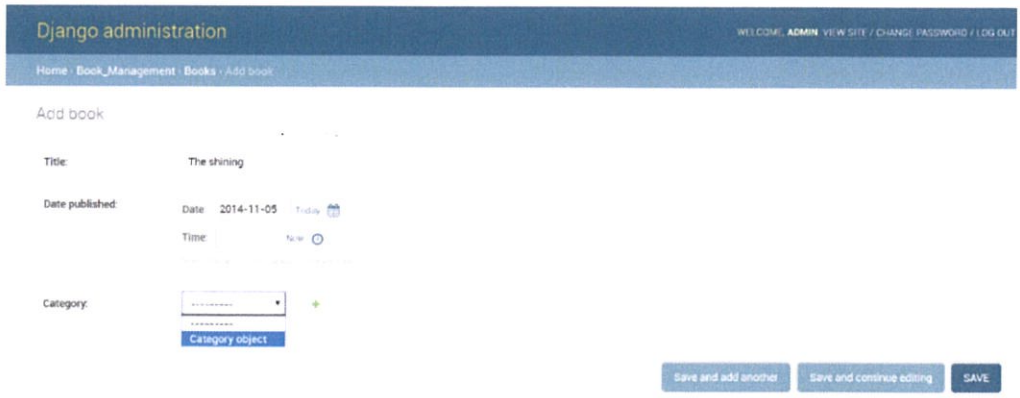
รูปที่ 2.9 สัญลักษณ์ Django

2.9.1 Django Admin Site

อีกหนึ่ง Feature เด่นสำหรับ Django คือหน้า Web Page ซึ่งเป็นหน้าสำหรับใช้ในการทำ Database CRUD operation (Create, Read, Update, Delete) เหมาะสำหรับใช้เป็นหลังบ้านสำหรับ Web Admin ใช้บริหารงานต่าง ๆ โดยไม่ต้องเข้าถึงฐานข้อมูลโดยตรง หรือสามารถใช้ในการ debug ข้อมูลได้อย่างรวดเร็วอีกด้วย และยิ่งไปกว่านั้น Django Admin ยังสามารถปรับแต่งแก้ไขให้แสดงรายละเอียดตามที่ต้องการได้หลากหลาย เช่น ต้องการใส่ search box, ใส่ filter เปลี่ยนการแสดงผล เช่น Object Book ต้องการแสดงแค่ชื่อของหนังสือเพียงอย่างเดียว เป็นต้น



รูปที่ 2.10 หน้า Administration Django



รูปที่ 2.11 หน้าตัวอย่างการใช้งาน django

2.10 SQLite

SQLite เป็นระบบฐานข้อมูลเชิงสัมพันธ์ (Relational Database) เป็นระบบที่มีการจัดเก็บข้อมูลให้อยู่ในรูปแบบของตาราง นิยมใช้กันมากเนื่องจากง่ายต่อการทำความเข้าใจ และสามารถออกรีพอร์ตได้ง่าย ซึ่ง SQLite นี้มีข้อดีหลายอย่างเช่น ทำงานได้อย่างรวดเร็ว ใช้พื้นที่หน่วยความจำน้อย และที่สำคัญคือความเป็น serverless นั่นเอง

ส่วนใหญ่แล้ว SQL database นั้นจะถูกดำเนินการด้วยหน่วยประมวลผลของเซิฟเวอร์ โปรแกรมที่ต้องการจะเข้าถึงฐานข้อมูลจะติดต่อกับเซิฟเวอร์ผ่านทางการสื่อสารระหว่างโปรเซส(TCP/IP) เพื่อส่งคำร้องขอไปยังเซิฟเวอร์และรับผลลัพท์ ซึ่ง SQLite นั้นไม่ได้ทำงานในรูปแบบนั้น สำหรับ SQLite การทำงานจะเป็นการเข้าไปอ่านหรือเขียนไฟล์ฐานข้อมูลโดยตรงจากดิสก์เลยโดยไม่ต้องมีเซิฟเวอร์เป็นตัวกลาง

รูปที่ 2.12 สัญลักษณ์ SQLite

2.10.1 ข้อดีของการเป็น serverless

ข้อดีหลักๆของการเป็น serverless นั้นคือการที่เราไม่ต้องมีการติดตั้ง, ตั้งค่า, จัดการ , แก้ปัญหา กับเซิฟเวอร์ นี่เป็นเหตุผลว่าทำไม SQLite ถึงเป็น ฐานข้อมูลแบบ "zero-configuration" สำหรับโปรแกรมที่ใช้ SQLite นั้นไม่ต้องการการช่วยเหลือในการบริหารจัดการสำหรับการติดตั้งก่อนการใช้งานเลย ทุกโปรแกรมสามารถเข้าถึงดิสก์โดยใช้ฐานข้อมูล SQLite ได้เลย

SQLite นั้นเป็นเพียงฐานข้อมูลเดียวที่ยอมให้หลายๆแอปพลิเคชันสามารถเข้าถึงฐานข้อมูลเดียวกันในช่วงเวลาเดียวกันได้

2.11 RouteXL API

RouteXL API เป็นตัวที่ช่วยเพิ่มประสิทธิภาพในการเดินทางแบบหลายๆจุด ยกตัวอย่างเช่น จะเดินทางไปจากสถานที่หนึ่งไปยังอีกสถานที่หนึ่งจำนวน20สถานที่ จะได้เมตริกซ์เวลาขนาด 20*20ช่อง และจำนวนของเส้นทางที่เป็นไปได้ยิ่งมีขนาดใหญ่ยิ่งกว่า โดยมีค่าประมาณ 20! เลยทีเดียว ซึ่งปัญหาเหล่านี้นักคณิตศาสตร์จะเรียกว่าเป็นปัญหาที่ยากและไม่มีคำตอบที่ตายตัวว่า 20สถานที่ต้องไปสถานที่ตำแหน่งนี้ก่อนเสมอ โดยนักคณิตศาสตร์ได้ตั้งชื่อปัญหานี้ว่า The Travelling Salesman Problem (TSP) ซึ่งทาง RouteXL ได้คิดค้นวิธีแก้ปัญหา The Travelling

Salesman Problem (TSP)ที่สามารถใช้งานหาเส้นทางที่ดีที่สุดได้เกือบทั้งหมด แต่ถ้าเทียบในเรื่องของความเร็วในการคำนวณ RouteXL อาจจะไม่ใช่ตัวเลือกที่ดีที่สุดที่จะนำมาใช้

รูปที่ 2.13 สัญลักษณ์ RouteXL API

หากผู้ใช้ต้องการใช้งาน RouteXL API ต้องทำการส่งข้อมูลของสถานที่ที่จะเดินทาง RouteXL นี้จะทำการส่งค่ากลับมาเป็นเส้นทางที่ดีที่สุด เรียงตามลำดับแต่ละจุดที่จะต้องเดินทางในรูปของ JSON ซึ่งการส่งข้อมูลไปยัง API นั้นต้องใช้ Username และ password ที่ได้ทำการสมัครสมาชิกกับทางเว็บไซต์เรียบร้อยแล้ว(https://www.routexl.nl/register) ในแต่ละสถานที่ๆจะส่งต้องเป็นพิกัดละติจูดและลองจิจูดเท่านั้น ไม่สามารถส่งมาเป็นที่อยู่ของสถานที่ได้ โดยทางผู้พัฒนาได้จำกัดจำนวนสถานที่ที่สามารถส่งให้ API ทำการคำนวณได้มากที่สุด 10 สถานที่ ถ้าหากต้องการส่งจำนวนสถานที่เพิ่มขึ้นสามารถทำได้โดยทำการอัพเกรดระดับสมาชิกให้สูงขึ้นโดยส่งได้มากที่สุดถึง 200 สถานที่ต่อครั้ง และจำนวนครั้งในการส่งข้อมูลต่อวันนั้นไม่จำกัด แต่สามารถส่งได้ครั้งละหนึ่งครั้งเท่านั้น

```
curl \
--url https://api.routexl.nl/tour/ \
--user <username>:<password> \
--data 'locations=[{"address":"1","lat":"52.05429","lng":"4.248618"},{"ad-
dress":"2","lat":"52.076892","lng":"4.26975"},{"ad-
dress":"3","lat":"51.669946","lng":"5.61852"},{"ad-
dress":"4","lat":"51.589548","lng":"5.432482"}]&skipOptimisation=true'
```

รูปที่ 2.14 ตัวอย่างการส่ง Request ไปยัง RouteXL API

2.11.2 Post Tour

Post tour เป็นหนึ่งในฟังก์ชันของ RouteXL API โดยจะเป็นการส่งข้อมูลที่อยู่โดยเพิ่มเงื่อนไขต่างๆในแต่ละจุดลงไปด้วย เช่น จุด A ต้องมาก่อน จุด B เป็นต้น ซึ่ง API จะทำการตอบกลับข้อมูลเป็นลำดับการเดินทางไปสถานที่ที่มีประสิทธิภาพมากที่สุด

2.11.2.1 Input

Locations จะอยู่ในรูปของ JSON Array โดยในแต่ละสถานที่จะต้องมีชื่อของสถานที่ พิกัดละติจูดและลองจิจูด ลำดับแรกของ JSON Array จะเป็นจุดเริ่มต้น และลำดับสุดท้ายจะเป็นจุดสิ้นสุด ถ้าต้องการจุดเริ่มต้นและจุดสิ้นสุดเป็นจุดๆเดียวกันก็ต้องกำหนดจุดนั้น2 ครั้งในตำแหน่งแรกและตำแหน่งสุดท้ายในแต่ละ Locations สามารถเพิ่ม Restrictions เพิ่มได้ ซึ่งจะเป็นตัวกำหนดข้อจำกัดต่างๆ ดังนี้

1) Ready เวลาที่เร็วที่สุดที่จะมายังสถานที่นี้ได้(ป้อนข้อมูลเป็นหน่วยเวลามีค่ามากกว่า 0)
2) Due เวลาที่ช้าที่สุดที่จะมายังสถานที่นี้ได้(ป้อนข้อมูลเป็นหน่วยเวลามีค่ามากกว่า 0)
3) Before ป้อนค่า Index ของสถานที่ที่จะเป็นจุด Pickup ของสถานที่นี้ (ต้องไปยังสถานที่ในค่า Indexนี้ก่อนถึงจะสามารถมายังสถานที่นี้ได้)
4) After ป้อนค่า Index ของสถานที่ที่จะเป็นจุด Delivery ของสถานที่นี้ (ต้องมายังสถานที่นี้ก่อน ถึงจะไปยังสถานที่ในค่า Index ได้)
5) *skipOptimisation* เป็นค่า boolean หากตั้งค่าเป็น false ระบบจะไม่คำนวณหาเส้นทางที่ดีที่สุด จะตอบกลับมาเป็นลำดับของสถานที่เดิม

```
locations=[{"address":"The Hague, The Netherlands","lat":"52.05429","lng":"4.248618"},
{"address":"The Hague, The Netherlands","lat":"52.076892","lng":"4.26975"},{"ad-
dress":"Uden, The Netherlands","lat":"51.669946","lng":"5.61852"},{"address":"Sint-
Oedenrode, The Netherlands","lat":"51.589548","lng":"5.432482"}]
```

รูปที่ 2.15 ตัวอย่าง Input ที่จะส่งไปยัง RouteXL API

2.11.2.2 Output

1) id เป็นค่า unique ID ในแต่ละคำขอ
2) count เป็นค่าตัวเลขแสดงจำนวนสถานที่ทั้งหมด
3) feasible เป็นค่า boolean แสดงความเป็นไปได้ หากเส้นทางที่ได้นั้นตรงกับความต้องการตาม input ทั้งหมด feasible จะมีค่าเป็น true หากอันใดอันหนึ่งขัดกับข้อจำกัดในแต่ละจุดค่า feasible จะเป็น false
4) route เป็น array ของwaypoint โดยจะเป็นลำดับของจุดที่ใช้เวลาในการเดินทางน้อยที่สุด จุดแรกของ array จะเป็นจุดเริ่มต้น และจุดสุดท้ายของ arrayจะเป็นจุดสิ้นสุด

ในแต่ละ waypoint จะมีค่าระยะเวลาในการเดินทางเป็นนาที และระยะสะสมในแต่ละจุดเป็นหน่วยกิโลเมตร


```
{"id":"4T4446Pj","count":4,"feasible":true,"route":{"0":{"name":"The Hague, The Nether-
lands","arrival":0,"distance":0},"1":{"name":"The Hague, The Netherlands","ar-
rival":5,"distance":3.9},"2":{"name":"Uden, The Netherlands","arrival":93,"dis-
tance":137.1},"3":{"name":"Sint-Oedenrode, The Netherlands","arrival":112,"dis-
tance":160.5}}}
```


รูปที่ 2.16 ตัวอย่าง Output ที่ RouteXL API ทำการตอบกลับมา

2.11.3 Response Code

API จะทำการตอบกลับด้วย HTTP Status Code ดังนี้

1) 401 การยืนยันตนมีปัญหา(Authenticate Problem)
2) 403 จำนวนสถานที่ที่ส่งมามากเกินไป
3) 429 มีการคำนวณลำดับของสถานที่ที่ส่งมาก่อนหน้าค้างอยู่
4) 204 ไม่สามารถคำนวณ Distance Matrix, Tour หรือ Route ได้
5) 200 สำเร็จ

2.12 Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) เป็นตัวช่วยในการส่งข้อความที่น่าเชื่อถือข้ามแพลตฟอร์มได้โดยไม่เสียค่าใช้จ่าย อาจจะใช้ FCM ในการแจ้งเตือนแอปพลิเคชันฝั่ง Client ว่ามี E-mail ใหม่เข้ามาหรือมีข้อมูลที่พร้อมจะซิงค์ข้อมูล สามารถส่งแจ้งเตือนเพื่อเป็นการกระตุ้นให้ผู้ใช้มีการเข้ามาใช้งานแอปพลิเคชัน โดยสามารถส่งข้อความขนาด 4KB ไปยังแอปพลิเคชันฝั่ง Client ได้

รูปที่ 2.17 สัญลักษณ์ Firebase Cloud Messaging

2.12.1 ความสามารถหลักของ Firebase Cloud Messaging

1) ส่งข้อความแจ้งเตือน หรือส่งข้อมูล ไปยังผู้ใช้ หรือแสดงข้อมูลและกำหนดสิ่งที่จะเกิดขึ้นในแอปพลิเคชันได้อย่างสมบูรณ์แบบ

2) สามารถกำหนดกลุ่มเป้าหมายตามวัตถุประสงค์ได้ ส่งข้อความไปยังแอปพลิเคชัน ฝั่งผู้ใช้ได้ 3แบบ คือ ส่งไปยังอุปกรณ์ที่กำหนดเพียงเครื่องเดียว ส่งไปยังกลุ่มอุปกรณ์ทั้งหมด และส่งไปยังผู้ใช้ที่ทำการสมัครสมาชิกหัวข้อนั้นๆ

2.12.2 หลักการทำงานของ Firebase Cloud Messaging

การใช้งาน Firebase Cloud Messaging นั้นประกอบไปด้วย 2ส่วนหลักๆ คือ ส่วนรับข้อมูล และส่วนส่งข้อมูล

1) สภาพแวดล้อมที่เชื่อถือได้อย่างฟังก์ชันคลาวด์ของ Firebase หรือ แอปพลิเคชันฝั่งเซิร์ฟเวอร์ที่สามารถสร้างข้อความ กำหนดกลุ่มเป้าหมาย และส่งข้อความ
2) แอปพลิเคชันบนอุปกรณ์ IOS, แอนดรอยด์ หรือเว็บที่ทำการรับข้อความ

รูปที่ 2.18 หลักการทำงานของ Firebase Cloud Messaging

2.12.3 วิธีการใช้งาน Firebase Cloud Messaging

1) ติดตั้งFCM SDK โดยติดตั้ง Firebase และ FCM ในแอปพลิเคชันซึ่งแต่ละแพลตฟอร์มจะมีขั้นตอนการติดตั้งที่ต่างกันไป
2) พัฒนาแอปพลิเคชันฝั่งผู้ใช้ โดยมีการจัดการข้อความ การสมัครสมาชิกหัวข้อต่างๆ และพัฒนาคุณสมบัติอื่นๆตามความต้องการของแต่ละแอปพลิเคชัน ในระหว่างการพัฒนานั้น สามารถทดลองส่งข้อความได้โดยใช้หน้าควบคุมของ Firebase ได้
3) พัฒนาโปรแกรมฝั่งเซิร์ฟเวอร์ อาจจะใช้ Admin SDK หรือ โปรโตคอลเซิร์ฟเวอร์อันใดอันหนึ่ง ที่สามารถสร้างตรรกะในการส่งข้อความ,การยืนยันตน,การส่ง Request ,การจัดการ Response และอื่นๆ

2.13 Firebase Realtime Database

จัดเก็บและซิงค์ข้อมูลด้วยฐานข้อมูลแบบ NoSQL ในระบบคลาวด์ ข้อมูลจะซิงค์กับผู้ใช้แบบเรียลไทม์และยังสามารถใช้งานได้แม้ไม่ได้เชื่อมต่อกับอินเทอร์เน็ต ข้อมูลจะจัดเก็บในรูป JSON และซิงค์ข้อมูลแบบเรียลไทม์สำหรับผู้ใช้ที่เชื่อมต่อเข้ามาทุกคน ไม่ว่าผู้ใช้นั้นจะใช้งานจากอุปกรณ์ iOS แอนดรอยด์ หรือบนเว็บก็ตามสามารถแบ่งปันข้อมูลระหว่างกันได้ และได้รับข้อมูลที่ปรับปรุงใหม่ล่าสุดเสมอ

รูปที่ 2.19 สัญลักษณ์ Firebase Realtime Database

2.13.1 ความสามารถหลักของ Firebase Realtime Database

1) เรียลไทม์ แทนที่จะทำการส่ง HTTP request แบบทั่วๆไป Firebase Realtime Database จะใช้วิธีการซิงค์ข้อมูลตลอดเมื่อมีข้อมูลเปลี่ยนแปลง ส่งผลให้อุปกรณ์ใดๆที่เชื่อมต่อกับฐานข้อมูลนี้ได้รับข้อมูลที่ปรับปรุงแล้วภายในเวลาเพียงเสี้ยววินาที
2) ใช้งานแบบออฟไลน์ได้ Firebase ยังคงสามารถใช้งานได้แม้ไม่ได้เชื่อมต่ออินเทอร์เน็ตแล้วเพราะ Firebase Realtime Database SDK นั้นจะเก็บข้อมูลไว้ในฮาร์ดดิสก์ด้วย เมื่อมีการเชื่อมต่อใหม่อีกครั้ง อุปกรณ์ฝั่งผู้ใช้จะได้รับข้อมูลปรับปรุงล่าสุดทันที
3) สามารถเข้าถึงได้จากฝั่งไคลแอนท์ โดยสามารถเข้าถึงฐานข้อมูลได้โดยตรงจากอุปกรณ์ฝั่งไคลแอนท์ได้เลย ไม่จำเป็นต้องมีแอปพลิเคชันฝั่งเซิร์ฟเวอร์ ความปลอดภัยและการตรวจสอบความถูกต้องของข้อมูลนั้นทำผ่าน Firebase Realtime Database Security Rules โดยจะใช้กฎเหล่านั้นเมื่อข้อมูลถูกเขียนหรืออ่าน

2.13.2 หลักการทำงานของ Firebase Realtime Database

Firebase Realtime Database ช่วยสร้างแอปพลิเคชันที่สามารถเข้าถึงข้อมูลที่มีการป้องกันได้โดยตรงจากฝั่งไคลแอนท์ ข้อมูลนั้นจะถูกเก็บไว้ในหน่วยจัดเก็บภายในเครื่องด้วย เมื่อเวลาที่ผู้ใช้ไม่ได้เชื่อมต่ออินเทอร์เน็ตก็ยังสามารถเข้าถึงข้อมูลได้อยู่ส่งผลให้

ผู้ใช้ได้รับการตอบสนองอยู่ตลอดเวลา เมื่ออุปกรณ์กลับมาเชื่อมต่ออินเทอร์เน็ตอีกครั้ง จะทำการซิงค์ข้อมูลกับฐานข้อมูลเพื่อรับข้อมูลล่าสุดเสมอ

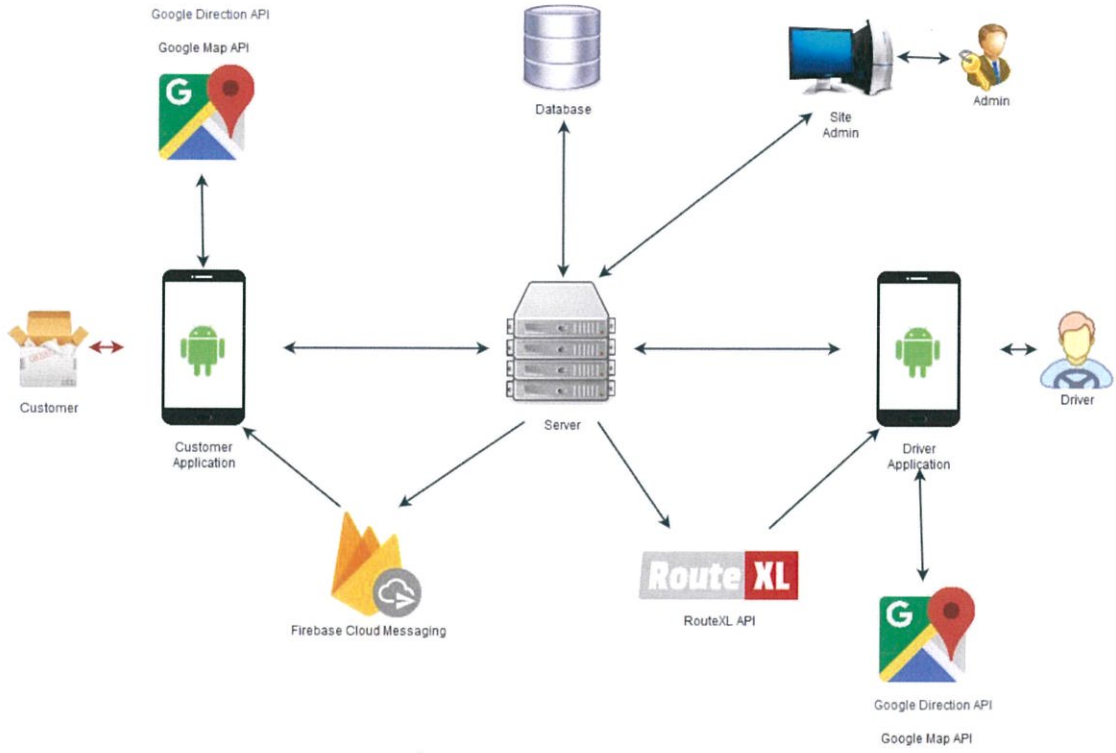
Firebase Realtime Database นั้นมีความยืดหยุ่นตามกฎที่เราตั้งไว้ (expression-based rules) เรียกว่า Firebase Realtime Database Security Rules โดยจะกำหนดว่าข้อมูลจะถูกสร้างขึ้นมาได้อย่างไร ข้อมูลจะสามารถถูกอ่านหรือเขียนได้เมื่อไร เมื่อนำมารวมกับ Firebase Authentication แล้วผู้พัฒนาสามารถกำหนดว่าใครสามารถเข้าถึงข้อมูลเหล่านี้ได้ และวิธีที่พวกเขาเข้าถึงข้อมูล

เป็นฐานข้อมูลแบบ NoSQL ซึ่งมีประสิทธิภาพและการทำงานที่ต่างกับฐานข้อมูลเชิงสัมพันธ์ ฐานข้อมูลแบบเรียลไทม์นั้นออกแบบมาให้มีการดำเนินการอย่างรวดเร็ว ส่งผลให้สามารถสร้างประสบการณ์การตอบสนองได้อย่างราบรื่นกับผู้ใช้หลักล้านคน

บทที่ 3

การออกแบบ และพัฒนาระบบ

3.1 ภาพรวมของระบบ



รูปที่ 3.1 ภาพรวมของระบบ

ทาง Customer จะทำการเรียกบริการผ่าน Android Application บน smart phone โดยทำการเลือกจุดที่จะให้มารับสินค้าไปส่งกับจุดหมายปลายทางของสินค้าจากนั้นเลือกชนิดของยานพาหนะ ระบบจะคำนวณค่าใช้จ่ายของบริการ จากนั้นจะส่ง Request ไปเก็บไว้ในDatabase

ฝั่ง Driver ทำการกำหนดจุดเริ่มต้นของการเดินทางกับจุดหมายปลายทางบน Android Application บน Smartphone แล้วส่งไปยัง Database เพื่อเก็บข้อมูลไว้ เมื่อมีการส่งคำขอมาจาก Driver ทาง Server จะทำการค้นหาคำขอของ Customer ที่มีระยะการเดินทางที่เพิ่มขึ้นไม่เกินที่กำหนด ชนิดยานพาหนะที่ต้องการเดียวกันกับของ Driver และยังไม่มี Driver คนใดทำการเลือก แล้ว จากนั้น Server จะส่งรายการของคำขอที่เหมาะสมให้ Driver เลือกว่าจะให้บริการคำขอของ Customer ใดบ้าง โดยจะมีรายละเอียดที่ช่วยในการตัดสินใจ คือ ระยะทาง และค่าบริการ

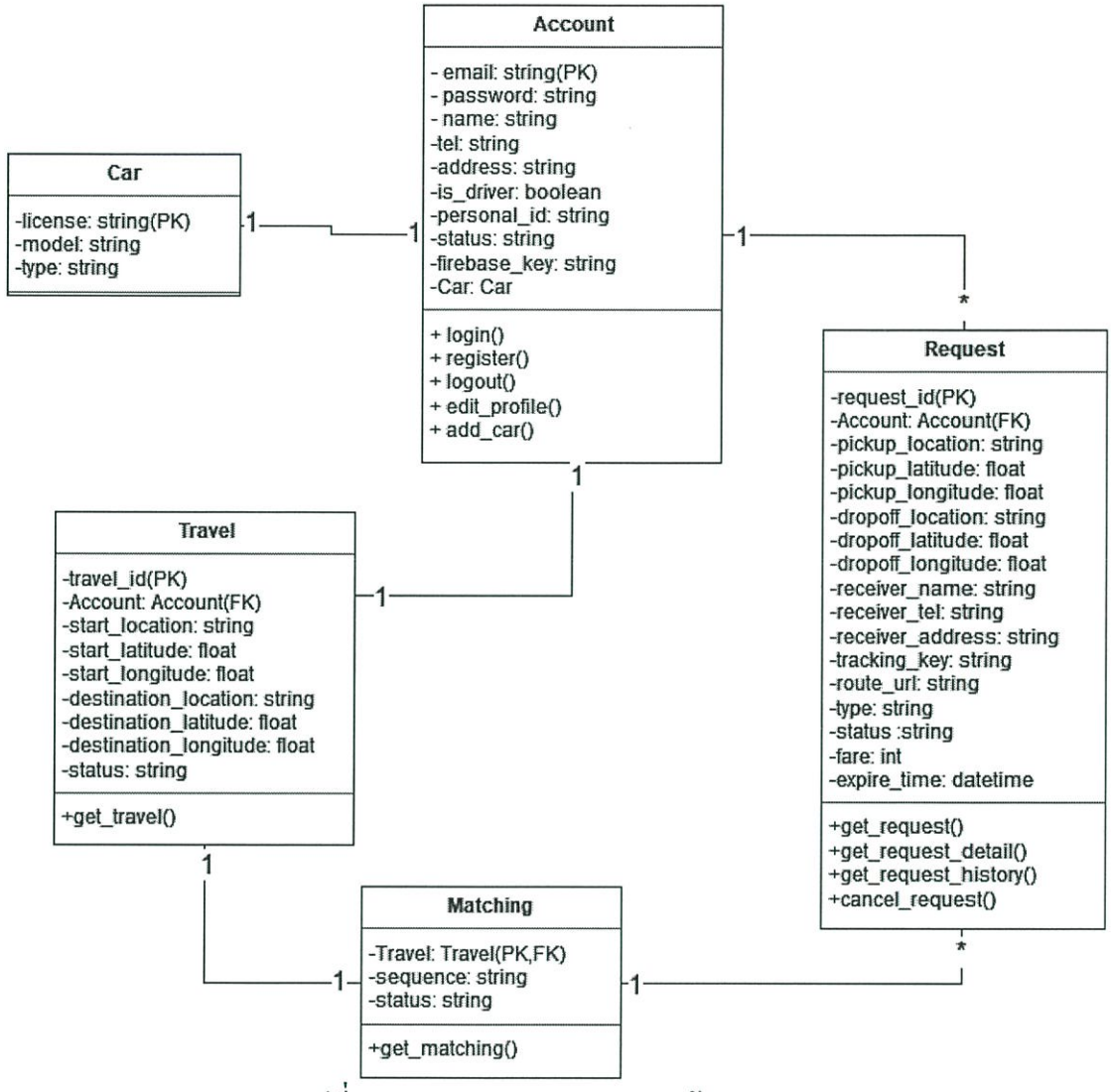
เมื่อ Driver ทำการเลือกคำขอของ Customer ได้แล้ว โดย ณ ปัจจุบัน สามารถเลือกได้มากที่สุด 4 คำขอ เนื่องจากข้อจำกัดของ API ที่ทางผู้จัดทำได้ใช้อยู่ในปัจจุบัน จะส่งไปยัง Server ให้ทำการคำนวณเส้นทางที่ดีที่สุดในการเดินทางไปให้บริการ Customer

ทาง Server จะทำการจัดลำดับการเดินทางของ Driver โดยจะส่งการคำนวณการเดินทางไปยัง RouteXL API เมื่อทำการเรียงลำดับการเดินทางได้แล้วจะส่งการแจ้งเตือนการตอบรับคำขอไปยัง Customer ผ่าน Firebase Cloud Messaging และทำการส่งลำดับการเดินทางพร้อมรายละเอียดข้อมูลของแต่ละตำแหน่งไปยัง Driver

3.2 ความต้องการของผู้ใช้

1) Customer สามารถสมัครสมาชิกและลงชื่อเข้าใช้ได้
2) Customer สามารถแก้ไขรายละเอียดของสมาชิกได้
3) ผู้ใช้ที่เป็นCustomerและ Driver สามารถกำหนดตำแหน่งบนแผนที่ได้ด้วยวิธีการค้นหาสถานที่ หรือการเลือกตำแหน่งบนแผนที่ได้
4) Customer สามารถเลือกขนาดของยานพาหนะที่จะเรียกใช้ได้
5) มีระบบคำนวณราคาค่าบริการโดยอ้างอิงจากระยะทางและขนาดของพาหนะที่เลือกใช้
6) Customerสามารถเรียกดูประวัติการใช้งานของตนได้ได้
7) Driverสามารถเลือกที่จะรับหรือไม่รับบริการที่ทางServerได้ส่งไปให้ได้
8) มีระบบแจ้งเตือน เมื่อมีการจับคู่กันระหว่าง Customer กับ Driver
9) Customer สามารถยกเลิกการเรียกใช้บริการได้
10) มีระบบคะแนน (Rating) ของ Driver

3.3 การออกแบบฐานข้อมูล



รูปที่ 3.2 UML Diagram ของฐานข้อมูล

3.3.1 Account

1) email ใช้เก็บที่อยู่ E-mail ของผู้ใช้งาน ใช้แทน Username ของบัญชีผู้ใช้งานในแอปพลิเคชัน
2) password รหัสผ่านผู้ใช้งานที่ถูกเข้ารหัสไว้
3) name ชื่อ-สกุล ผู้ใช้งาน
4) tel เบอร์โทรศัพท์ของผู้ใช้งาน
5) address ที่อยู่ของผู้ใช้งาน
6) is_driver สถานะผู้ใช้งานว่าเป็น Driver หรือไม่
7) personal_id เลขบัตรประจำตัวประชาชน (สำหรับ Driver)

8) status สถานะของผู้ใช้งาน (Driver) ณ ขณะนั้นว่าพร้อมรับบริการหรือไม่
9) firebase_key เป็น key ที่ใช้ในการแจ้งเตือน(Notification) ผ่านแอปพลิเคชัน ในกรณีที่ Request มีการ Match กับ Travel
10) car ข้อมูลรถของผู้ใช้งาน (Driver)
11) login() ฟังชั่นในการเข้าสู่ระบบ
12) register() ฟังชั่นในการสมัครสมาชิกเพื่อใช้งานแอปพลิเคชัน
13) logout() ฟังชั่นในการออกจากระบบ
14) edit_profile() ฟังชั่นในการแก้ไขข้อมูลของผู้ใช้งาน
15) add_car() ฟังชั่นในการเพิ่มยานพาหนะของผู้ใช้งาน (สำหรับ Driver)

3.3.2 Car

1) license ทะเบียนรถของผู้ใช้งาน (Driver)
2) model รุ่น ยี่ห้อ รายละเอียดต่างๆของรถของผู้ใช้งาน (Driver)
3) type ประเภทของรถของผู้ใช้งาน (Driver)

3.3.3 Travel

1) travel_id เก็บ ID ของแต่ละ Travel
2) Account ผู้ใช้งาน(Driver) ที่ต้องการ get_travel()
3) start_location ข้อมูลสถานที่เริ่มต้นที่ผู้ใช้งาน (Driver) จะให้บริการ
4) start_latitude ค่า latitude ของสถานที่เริ่มต้นที่ผู้ใช้งาน (Driver) จะให้บริการ
5) start_longitude ค่า longitude ของสถานที่เริ่มต้นที่ผู้ใช้งาน (Driver) จะให้บริการ
6) destination_location ข้อมูลสถานที่สิ้นสุดการให้บริการของผู้ใช้งาน (Driver) ที่จะให้บริการ
7) destination_latitude ค่า Latitude ของสถานที่สิ้นสุดการให้บริการของผู้ใช้งาน (Driver) ที่จะให้บริการ
8) destination_longitude ค่า Longitude ของสถานที่สิ้นสุดการให้บริการของผู้ใช้งาน (Driver) ที่จะให้บริการ
9) status สถานะของ Travel ว่ากำลังดำเนินการ, ยกเลิก, หรือสำเร็จแล้ว
10) get_travel() ฟังชั่นในการเริ่มต้นให้บริการของผู้ใช้งาน (Driver)

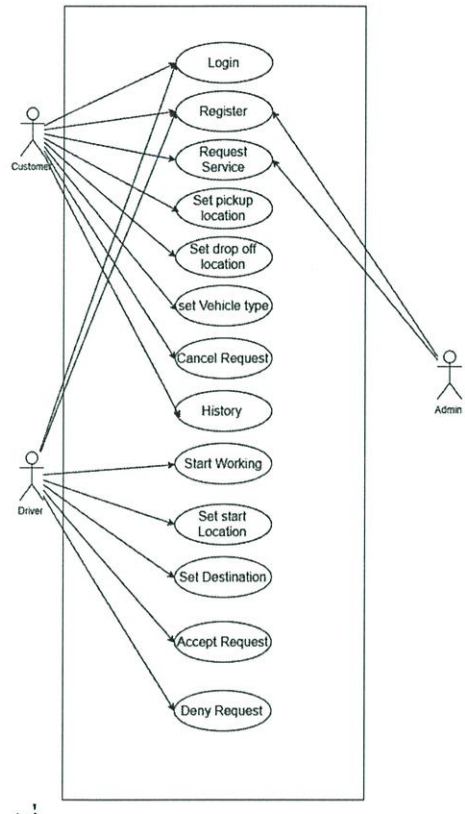
3.3.4 Request

1) request_id เก็บ ID ของแต่ละ Request
2) Account ผู้ใช้งาน (Customer) ที่ต้องการ get_request()
3) pickup_location ข้อมูลสถานที่เริ่มต้นที่ผู้ใช้งาน (Customer) ต้องการรับบริการ
4) pickup_latitude ค่า Latitude ของสถานที่เริ่มต้นที่ผู้ใช้งาน (Customer) ต้องการรับบริการ
5) pickup_longitude ค่า Longitude ของสถานที่เริ่มต้นที่ผู้ใช้งาน (Customer) ต้องการรับบริการ
6) dropoff_location ข้อมูลสถานทีสิ้นสุดการรับบริการของผู้ใช้งาน (Customer)
7) dropoff_latitude ค่า Latitude ของสถานที่สิ้นสุดการรับบริการของผู้ใช้งาน (Customer)
8) dropoff_longitude ค่า Longitude ของสถานที่สิ้นสุดการรับบริการของผู้ใช้งาน (Customer)
9) type ชนิดยานพาหนะที่ต้องการรับบริการ
10) receiver_name ชื่อผู้รับสินค้า
11) receiver_tel เบอร์โทรศัพท์ของผู้รับสินค้า
12) receiver_address ที่อยู่ของผู้รับสินค้า
13) tracking_key เป็น Key ที่ใช้ในการ Tracking ผู้ใช้งาน (Driver)
14) route_url เป็น URL ของ Route เส้นทางของ Request นั้นๆ
15) status สถานะของ Request ว่ากำลังดำเนินการ, ยกเลิก, หรือสำเร็จแล้ว
16) fare คือบริการของ Request นั้นๆ
17) expire_time เวลาหมดอายุของ Request นั้นๆ
18) get_request() ฟังชั่นในการร้องขอรับการรับบริการของผู้ใช้งาน (Customer)
19) get_request_history() ฟังชั่นในการดูประวัติการรับบริการ
20) cancel_request() ยกเลิกการขอใช้บริการ (ต้องยกเลิกก่อนมีการ Matching)

3.3.5 Matching

1) Travel Travel ที่ทำการ Matching
2) sequence ลำดับการให้บริการของผู้ให้บริการ (Driver)
3) status สถานะของ Matching ว่ากำลังดำเนินการ, ยกเลิก, หรือสำเร็จแล้ว
4) get_matching ฟังชั่นในการร้องขอการจับคู่ของ Request และ Travel ของผู้ใช้งาน (Driver)

3.4 แผนภาพ Use Case



รูปที่ 3.3 แผนภาพ Use Case Diagram

โดยอธิบายการทำงานของแผนภาพได้ดังนี้

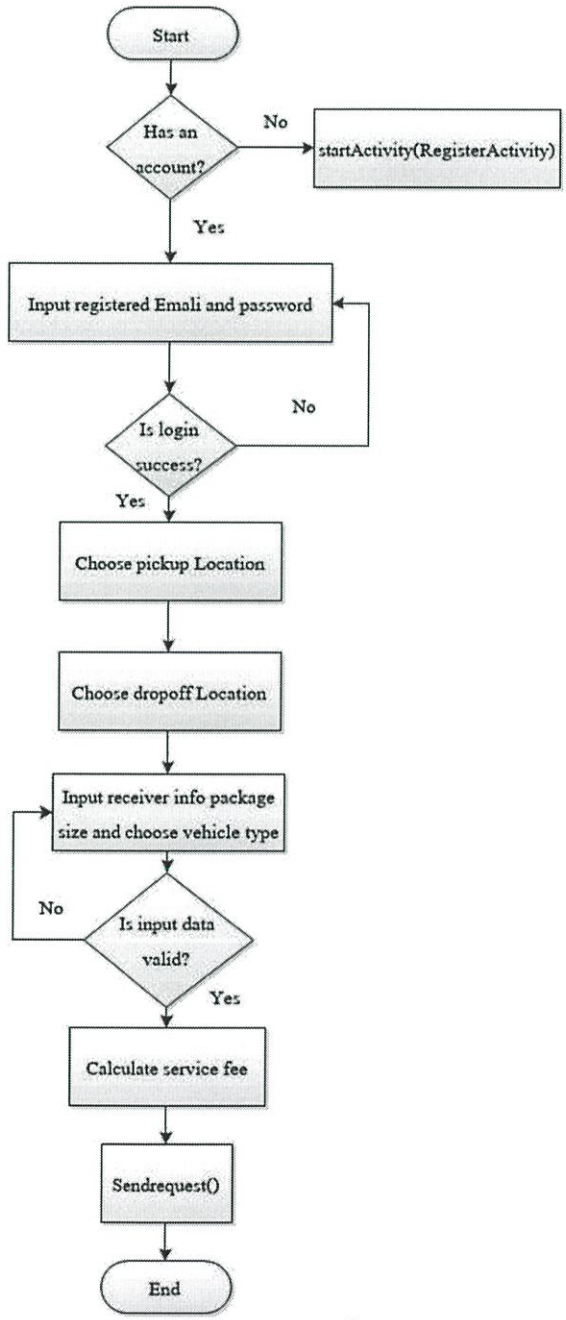
1) Login : Customer และ Driver สามารถลงชื่อเข้าใช้งานแอปพลิเคชันได้
2) Register : Customer และ Driver สามารถสมัครสมาชิกเพื่อใช้งานแอปพลิเคชันได้
3) Request Service : การขอใช้บริการขนส่งสินค้า
4) Set pickup Location : Customer เลือกตำแหน่งที่จะให้มารับสินค้าไปส่ง
5) Set Drop off Location : Customer เลือกตำแหน่งปลายทางของสินค้า
6) Set Vehicle Type : Customer เลือกชนิดของยานพาหนะได้
7) Cancle Request : Customer สามารถยกเลิก Request ได้ ตราบใดที่ยังไม่มีการจับคู่กับ Driver
8) History : Customer สามารถเรียกดูประวัติการใช้งานของตนได้
9) Start Working : Driver เริ่มทำงาน
10) Set Start Location : Driver เลือกตำแหน่งเริ่มทำงาน
11) Set Destination : Driver เลือกจุดหมายปลายทาง
12) Accept Request : Driver ตอบรับคำขอจาก Customer

13) Deny Request : Driver ปฏิเสธคำขอจาก Customer

3.5 แผนผังการทางานของระบบ (Flowchart Diagram)

3.5.1 แผนผังการทำงานของระบบในฝั่งCustomer

จะมีการทำงานคร่าวๆดังนี้

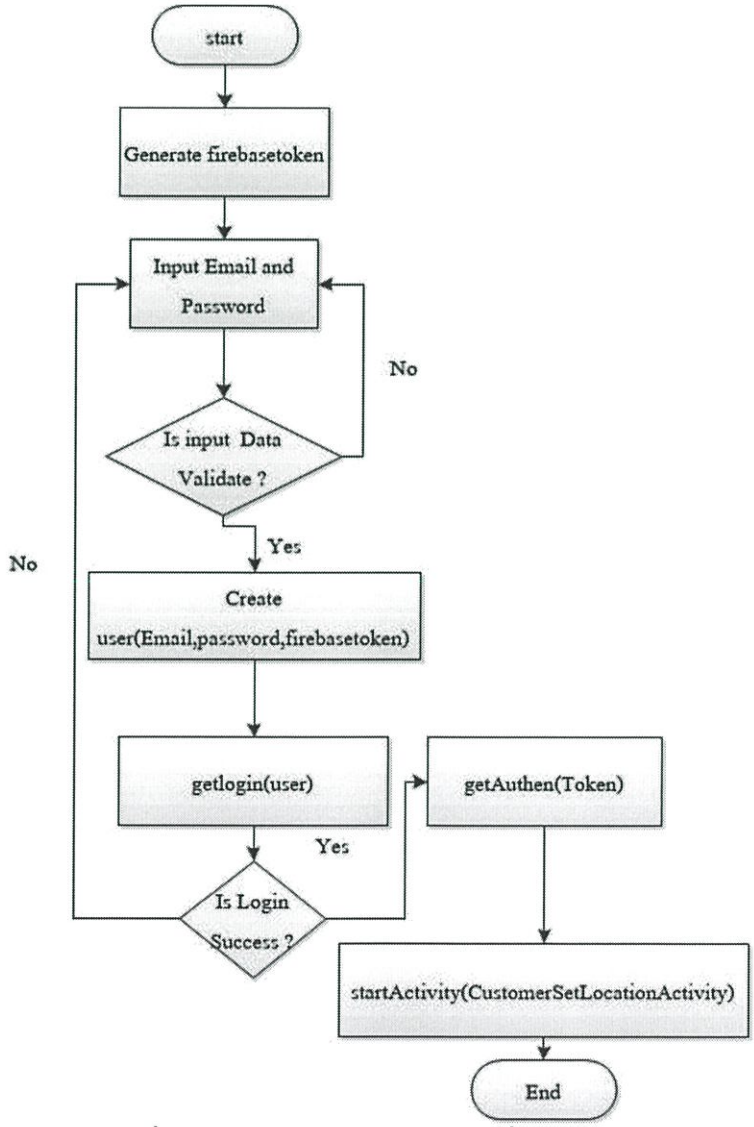


รูปที่ 3.4 แผนผังการทำงานของระบบในฝั่ง Customerแบบคร่าวๆ

เริ่มต้นจะให้ผู้ใช้ทำการลงทะเบียนกับทาง ShareX เพื่อรับบัญชีผู้ใช้มาใช้งานก่อน เมื่อมีบัญชีผู้ใช้แล้ว จะให้ลงชื่อเข้าใช้ตามบัญชีที่ได้ลงทะเบียนไว้ เมื่อลงชื่อเข้าใช้เสร็จสิ้นจะไปยังหน้าเลือกจุดรับส่งสินค้า และหน้ากรอกรายละเอียดของผู้รับ โดยในหน้านี้จะให้ผู้ใช้กรอกขนาดของสินค้า และเลือกชนิดของยานพาหนะที่จะใช้บริการด้วย โดยถ้าหากข้อมูลที่กรอกมานั้นไม่ถูกต้อง ระบบจะไม่คำนวณค่าบริการให้ ต้องแก้ไขข้อมูลให้ถูกต้อง โดยอาจจะเป็นกรอกรายละเอียดผู้รับไม่ตรงกับแบบฟอร์ม หรือขนาดของสินค้ามีขนาดเกินขีดจำกัดของยานพาหนะที่เลือกใช้ ถ้าไม่มีข้อผิดพลาดในการกรอกข้อมูล ระบบจะทำการคำนวณค่าบริการให้อัตโนมัติ และส่งคำขอไปยังเว็บเซิร์ฟเวอร์ต่อไป

3.5.1.1 แผนผังการทำงานหน้าลงชื่อเข้าใช้

มีแผนผังการทำงานดังนี้



รูปที่ 3.5 แผนผังการทำงานหน้าลงชื่อเข้าใช้

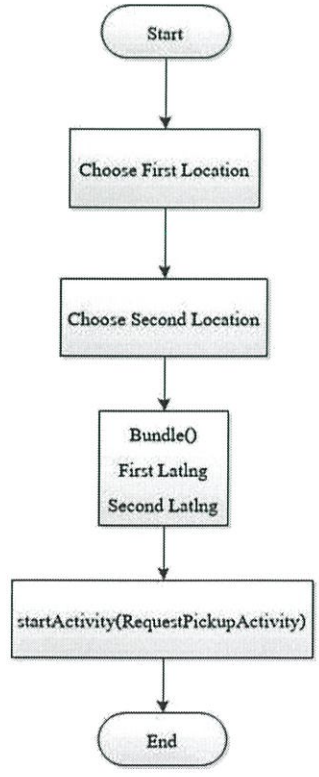
เริ่มต้นหน้าลงชื่อเข้าใช้จะทำการสร้าง firebasetoken key ขึ้นมาเพื่อไว้ใช้ในการส่งแจ้งเตือนมายังอุปกรณ์นี้ จากนั้นจะให้ผู้ใช้กรอก Email และ password ที่ได้ลงทะเบียนไว้ ถ้าแบบฟอร์มที่กรอกมานั้นถูก จะสร้าง Object User ขึ้นมาโดยประกอบไปด้วย Email password และ firebasetoken จากนั้นจะเรียกใช้งานฟังก์ชัน getlogin(User) ถ้าหากลงชื่อเข้าใช้สำเร็จจะมี response code เป็น 200 ซึ่งทางเว็บเซิร์ฟเวอร์จะตอบกลับมาด้วยข้อมูล คือ

1) Token เป็น Token ที่ใช้ในการAuthentication กับทาง API
2) is_driver จะตอบกลับมาเป็น String มีค่าเป็น True หรือ False
3) account_status ตัวแปรนี้ไว้บอกว่า driver นั้นมีงานค้างอยู่หรือไม่ ถ้ามีจะขึ้นสถานะว่า busy ถ้าไม่มีจะขึ้นสถานะ free

จากนั้นทางผู้จัดทำจะทำการยืนยันตนอีกครั้งหนึ่งด้วย Token ที่ได้รับมาจากเว็บเซิร์ฟเวอร์ โดยใช้งานฟังก์ชัน getAuthen(Token) ถ้าหากยืนยันตนผ่านจะตอบกลับมาเป็น Email ที่ผู้ใช้ทำการลงชื่อเข้าใช้ แอพพลิเคชันจะให้ผู้ใช้เข้าไปยังหน้าหลักต่อไป

3.5.1.2 แผนผังการทำงานหน้ากำหนดจุดรับส่งสินค้า

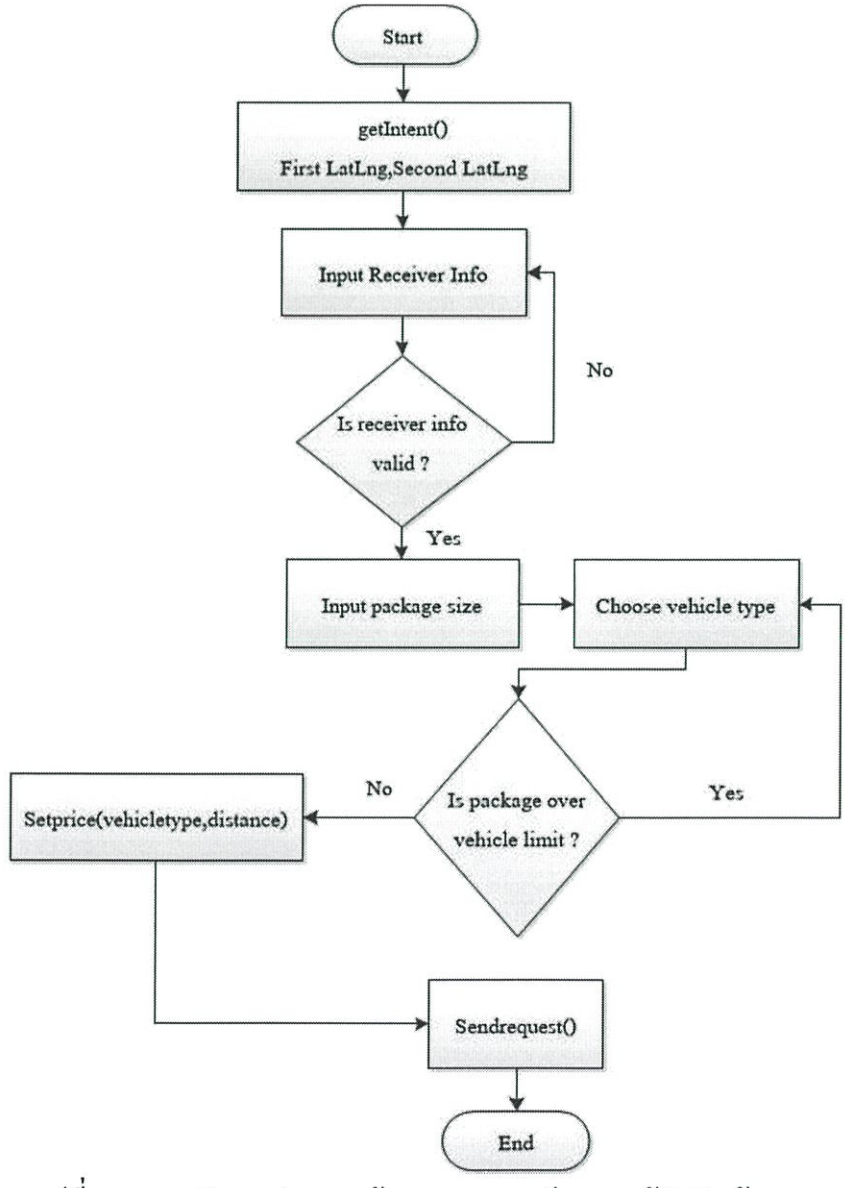
จะให้ผู้ใช้กำหนดจุดรับส่งสินค้าโดยเลือกจากจุดบนแผนที่หรือค้นหาจากชื่อบนแผนที่ก็ได้ เมื่อกำหนดจุดเสร็จเรียบร้อยแล้วตัวแอพพลิเคชันจะส่งค่าพิกัดจุดรับและจุดส่งสินค้าไปยัง RequestPickupActivity ด้วยวิธี Bundle() เพื่อทำการกรอกรายละเอียดต่อไป



รูปที่ 3.6 แผนผังการทำงานหน้ากำหนดจุดรับส่งสินค้า

3.5.1.3 แผนผังการทำงานหน้ากรอกรายละเอียดของผู้รับสินค้า

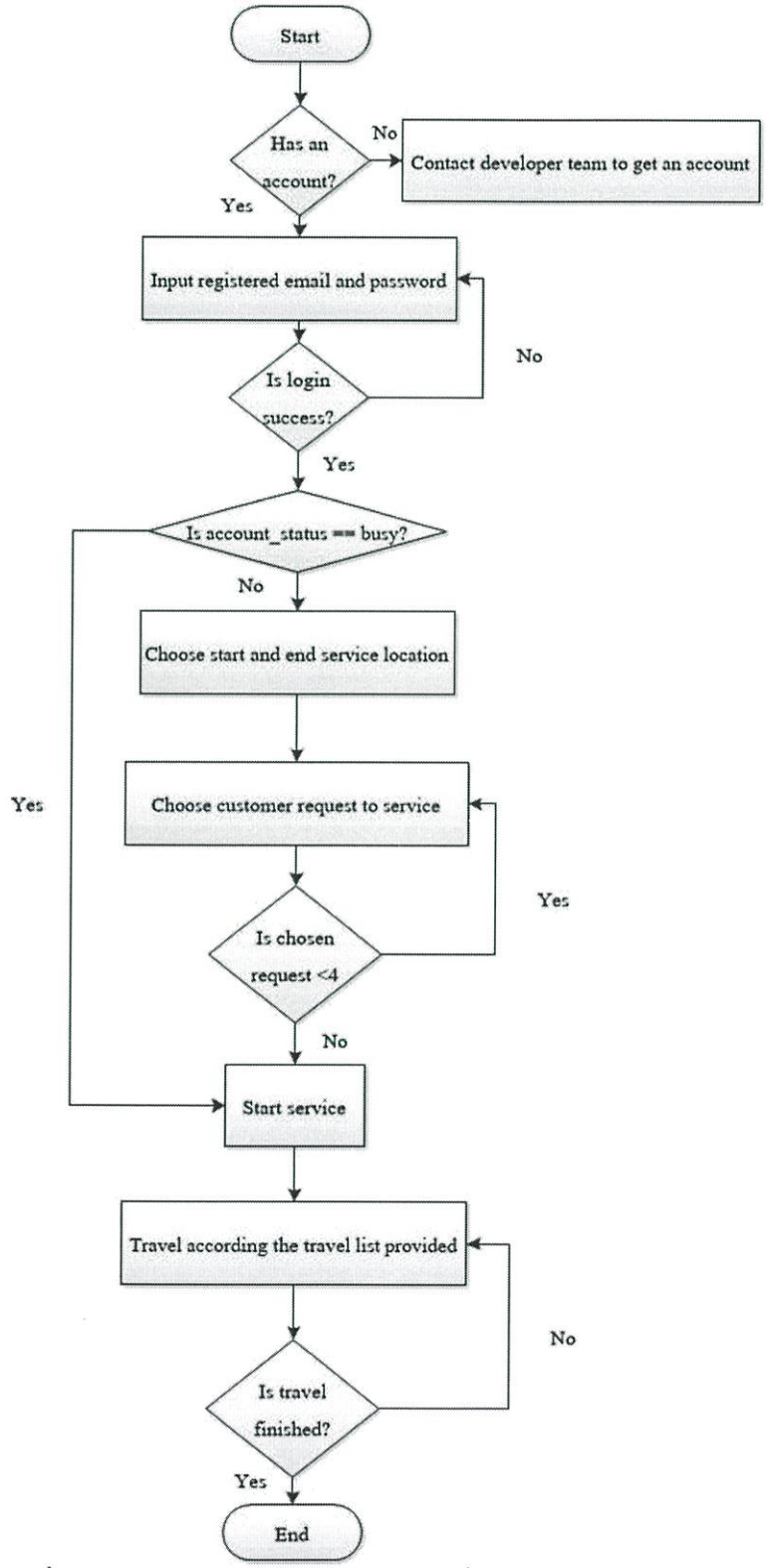
จะรับค่าพิกัดจุดรับจุดส่งสินค้ามาจากหน้ากำหนดจุดรับส่งสินค้า (CustomerSetLocationActivity) จากนั้นจะให้ผู้ใช้กรอกรายละเอียดของผู้รับ ระบบจะตรวจสอบความถูกต้องของข้อมูลที่กรอกเข้ามา ถ้ามีข้อผิดพลาดจะแสดง Error ให้เห็นในกล่องข้อความนั้นๆ จากนั้นจะให้ผู้ใช้กรอกขนาดของสินค้า และเลือกชนิดของยานพาหนะ ระบบจะตรวจสอบความสัมพันธ์ของขนาดสินค้ากับชนิดของยานพาหนะที่เลือกใช้ ถ้าขนาดของสินค้าเกินขีดจำกัดที่ยานพาหนะที่เลือกใช้รับได้จะแสดง Error และระบบจะคำนวณค่าบริการให้อัตโนมัติก็ต่อเมื่อไม่มี Error ในแต่ละส่วนเลย เมื่อข้อมูลทั้งหมดมีความถูกต้องจะส่ง request นี้ไปยังเว็บเซิร์ฟเวอร์ต่อไป



รูปที่ 3.7 แผนผังการทำงานหน้ากรอกรายละเอียดของผู้รับสินค้า

3.5.2 แผนผังการทำงานของระบบในฝั่ง Driver

มีการทำงานคร่าวๆ ดังนี้



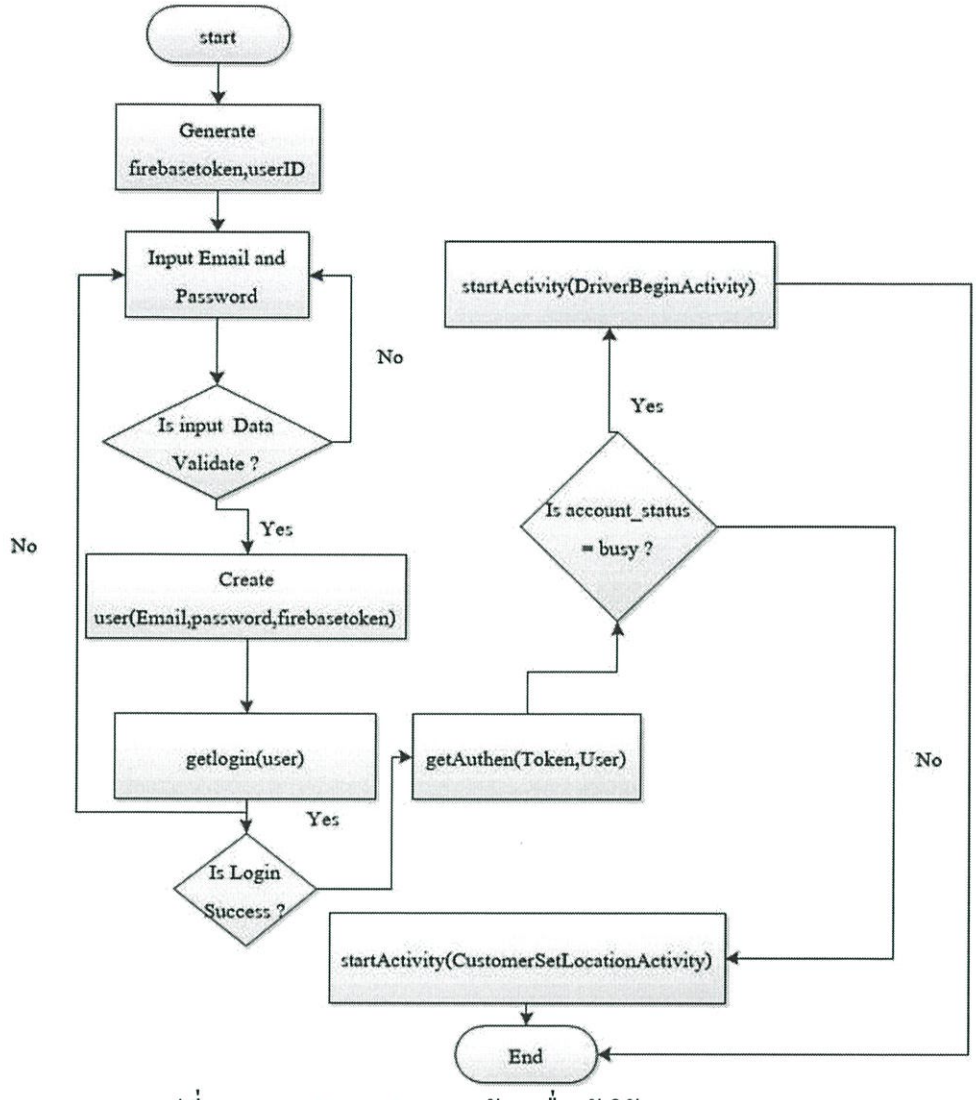
รูปที่ 3.8 แผนผังการทำงานของระบบในฝั่ง Driver แบบคร่าวๆ

เริ่มต้นผู้ให้บริการทำการลงชื่อเข้าใช้ ถ้าหากไม่มีบัญชีผู้ใช้ให้ติดต่อกับทางผู้พัฒนาเพื่อสร้างบัญชีผู้ใช้ให้ ถ้าหากลงชื่อเข้าใช้สำเร็จจะดูค่าของ account_status ที่ทางเว็บเซิร์ฟเวอร์ตอบกลับมาว่ามีค่าเป็น busy หรือ free ถ้า busy แสดงว่าผู้ให้บริการคนนี้มีงานบริการค้างอยู่ให้ทำงานที่ค้างต่อให้เสร็จก่อนถึงจะให้บริการใหม่ได้ ถ้า account_status เป็น free จะให้ผู้ให้บริการทำการกำหนดจุดเริ่มต้นและจุดสิ้นสุดให้บริการ จากนั้นส่งการเดินทางนี้ไปคำนวณในเว็บเซิร์ฟเวอร์ ค้นหาคำขอของผู้ใช้บริการที่ตรงตามอัลกอริทึมที่ได้ออกแบบไว้ ระบบจะตอบกลับมาเป็น List ของคำขอของผู้ใช้บริการให้ผู้ให้บริการเลือกรับบริการ โดยสามารถรับบริการได้มากที่สุด 4คำขอต่อการให้บริการ 1 ครั้ง เมื่อเลือกคำขอที่จะให้บริการเสร็จแล้วจะเริ่มการให้บริการ โดยส่งคำขอเหล่านั้นมาคำนวณลำดับการเดินทางกับทางเว็บเซิร์ฟเวอร์ก่อน ระบบจะตอบกลับมาเป็นลำดับการเดินทางที่ใช้ระยะเวลาน้อยที่สุดกลับมา ภายในแอพพลิเคชันก็จะแสดงเส้นทางการเดินทาง จุดที่ต้องเดินทางไป และรายละเอียดในแต่ละจุดที่จะต้องเดินทางว่าจุดนั้นเป็นจุดรับหรือจุดส่งสินค้า เดินทางไปยังจุดต่างๆตามที่กำหนดไว้จนครบเป็นอันเสร็จสิ้นกระบวนการ

3.5.2.1 แผนผังการทำงานหน้าลงชื่อเข้าใช้ของ Driver

เนื่องจากเราใช้หน้าลงชื่อเข้าใช้ร่วมกับของ Customer จึงมีการทำงานที่คล้ายๆกัน โดยส่วนที่เพิ่มเติมเข้ามา คือ จะมีการสร้าง userID ขึ้นมาเพื่อไว้เก็บตำแหน่งของ Firebase Database ใช้สำหรับในการเก็บค่าตำแหน่งปัจจุบันของ Driver แบบ Real-Time เพื่อใช้ในระบบติดตามสถานะของการให้บริการเมื่อมีการจับคู่กับคำของของCustomer

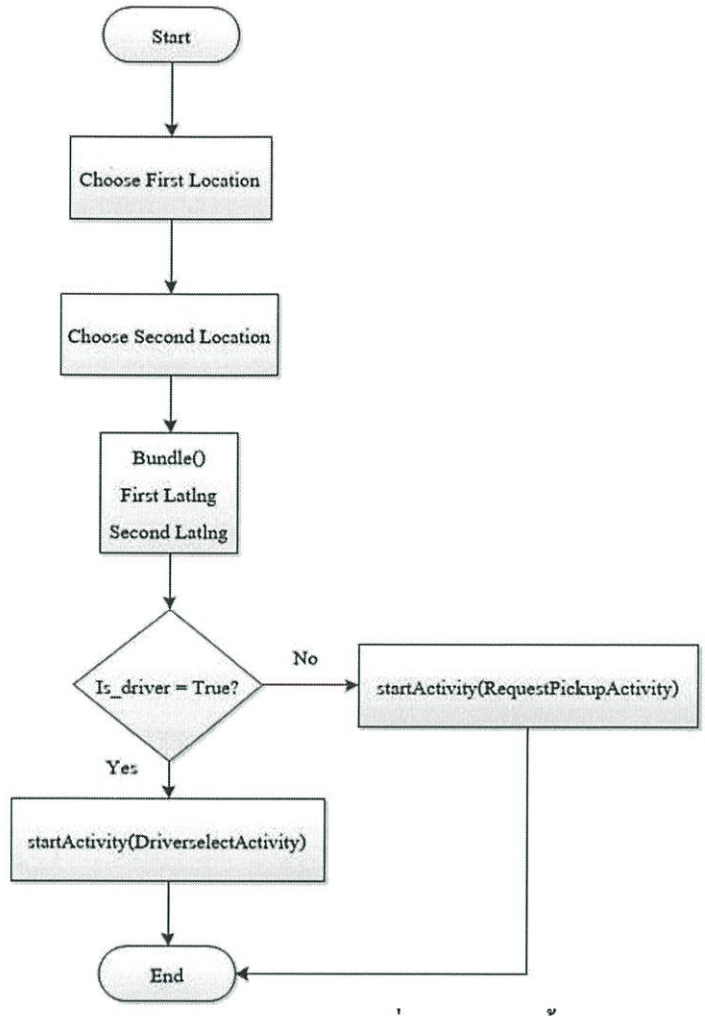
จากนั้นจะให้ Driver กรอก Email และ รหัสผ่านเพื่อทำการลงชื่อเข้าใช้ ระบบจะตรวจสอบความถูกต้องของข้อมูลที่ป้อนเข้ามา ถ้าข้อมูลไม่ถูกต้องจะขึ้นแจ้งเตือนให้ Driver ทราบเพื่อแก้ไข ถ้าข้อมูลถูกต้องแล้วระบบจะสร้าง Object User ขึ้นมา โดยประกอบไปด้วย Email password และ firebasetoken จากนั้นจะเรียกใช้งานฟังก์ชัน getlogin(User) ถ้าลงชื่อเข้าใช้ไม่สำเร็จจะแสดงแจ้งเตือนให้ผู้ใช้ทราบ แต่ถ้าหากลงชื่อเข้าใช้สำเร็จทางแอพพลิเคชันจะทำการยืนยันตนอีกครั้งหนึ่งจาก Token ที่ได้รับจากการลงชื่อเข้าใช้ ถ้าสำเร็จจึงจะสามารถเข้าใช้งานแอพพลิเคชันได้ โดยจะตรวจสอบค่าของ account_status ว่ามีค่าเป็น busy หรือ free ถ้า มีสถานะเป็นbusy แสดงว่าผู้ให้บริการคนนี้มีงานบริการค้างอยู่ให้ทำงานที่ค้างต่อให้เสร็จก่อนถึงจะให้บริการใหม่ได้ จึงเริ่มใช้งานแอพพลิเคชันที่หน้าเริ่มให้บริการ ถ้า account_status เป็น free แสดงว่า Driver คนนี้ไม่มีบริการที่ค้างอยู่สามารถให้บริการครั้งใหม่ได้ จะให้ไปเริ่มใช้งานแอพพลิเคชันที่หน้า CustomerSetLocationActivity



รูปที่ 3.9 แผนผังการทำงานหน้าลงชื่อเข้าใช้ของ Driver

3.5.2.2 แผนผังการทำงานหน้ากำหนดจุดเริ่มต้นและจุดสิ้นสุดการให้บริการ

เมื่อลงชื่อเข้าใช้เสร็จเรียบร้อยแล้ว จะให้ Driver กำหนดจุดรับส่งสินค้า โดยเลือกจากจุดบนแผนที่หรือค้นหาจากชื่อบนแผนที่ก็ได้ เมื่อกำหนดจุดเสร็จเรียบร้อยแล้ว ตัวแอพพลิเคชันจะส่งค่าพิกัดจุดเริ่มต้น และจุดสิ้นสุดให้บริการไปยัง DriverSelectActivity ด้วยวิธี Bundle() เพื่อทำการเลือกให้บริการคำขอของ Customer ต่อไป



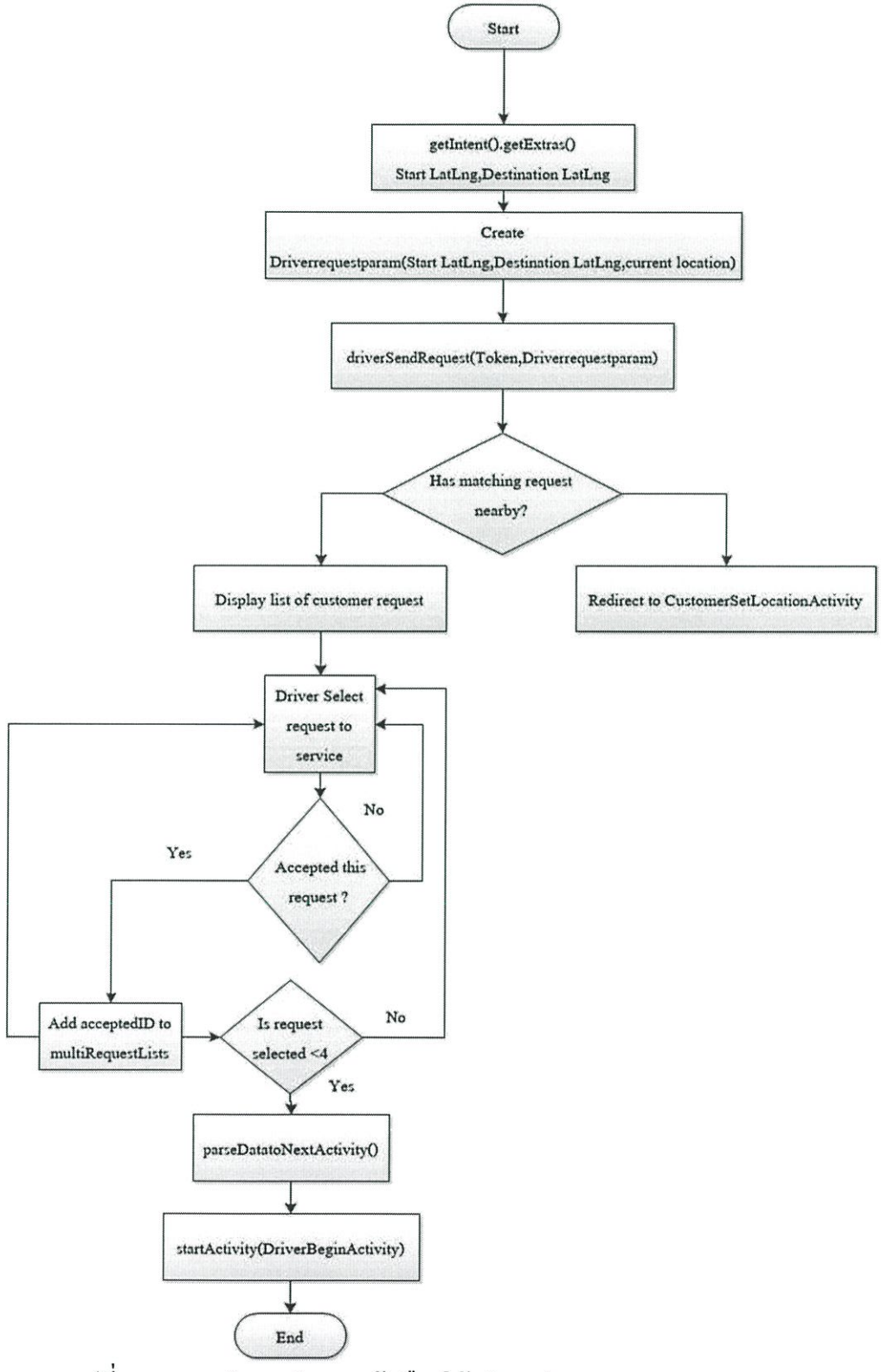
รูปที่ 3.10 แผนผังการทำงานหน้ากำหนดจุดเริ่มต้นและจุดสิ้นสุดการให้บริการ

3.5.2.3 แผนผังการทำงานหน้าเลือกให้บริการคำของของ Customer

เมื่อกำหนดจุดเริ่มต้นและจุดสิ้นสุดให้บริการเรียบร้อยแล้ว จะส่งค่าพิกัดจุดเหล่านั้นมายังหน้านี้ โดยแอพพลิเคชันจะทำการสร้าง Object Driverrequestparam ขึ้นมาโดยจะประกอบด้วย พิกัดจุดเริ่ม จุดสิ้นสุดการให้บริการ พิกัดตำแหน่งปัจจุบัน จากนั้นจะเรียกใช้งาน API driverSendRequest โดยจะส่ง Token และ Driverrequestparam ไปยังเว็บเซิร์ฟเวอร์ ระบบจะตอบกลับมาเป็น list ของคำขอของ Customer ถ้าหากไม่มีคำขอของCustomerที่เหมาะสมกันเลยจะไปยังหน้ากำหนดจุดเริ่มต้น และจุดสิ้นสุดให้บริการให้แต่ถ้าหากมีคำขอที่เหมาะสมจะแสดงเป็น List ให้ driver เลือกว่าจะให้บริการคำขอใดบ้าง เมื่อเลือกครบ 4 คำขอแล้วจะส่งข้อมูลไปยังหน้าเริ่มต้นให้บริการ(DriverBeginActivity) ต่อไป คือ

1) พิกัดจุดเริ่มต้นและจุดสิ้นสุดให้บริการ
2) Travel_id

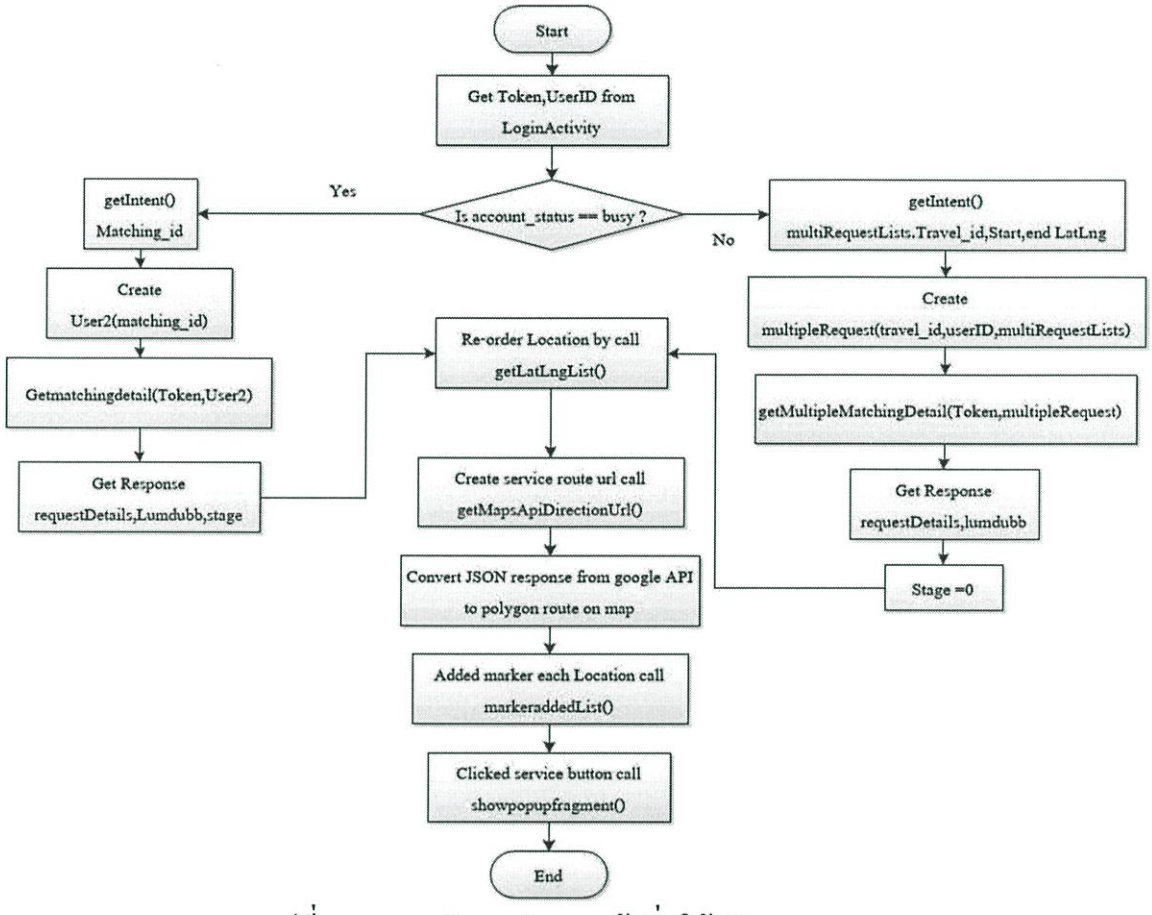
3) Arraylist ของ multiRequestLists ที่เก็บเลข request_id ที่ driver เลือกจะให้บริการ



รูปที่ 3.11 แผนผังการทำงานหน้าเลือกให้บริการคำของของ Customer

3.5.2.4 แผนผังการทำงานหน้าเริ่มให้บริการ

มีการทำงานดังนี้



รูปที่ 3.12 แผนผังการทำงานหน้าเริ่มให้บริการ

จะรับค่า Token ที่ใช้ในการยืนยันตน กับค่า userID ที่ใช้ในการบอกตำแหน่งข้อมูลใน Firebase Realtime Database จากหน้าลงชื่อเข้าใช้ (LoginActivity) จากนั้นจะตรวจสอบสถานะของ account_status ว่ามีสถานะ busy หรือ free

ในกรณีที่สถานะเป็น busy จะรับค่า matching_id มาจากหน้าลงชื่อเข้าใช้ จากนั้นแอพพลิเคชันจะสร้าง Object User2(matching_id) ขึ้นมาแล้วเรียกใช้งาน API Getmatchindetail(Token,User2)

แต่ในกรณีที่สถานะเป็น free จะรับค่า multiRequestLists, travel_id, start LatLng ,end Latlng และอื่นๆมาจาก หน้าเลือกให้บริการคำขอของ Customer(DriverSelectActivity) จากนั้นแอพพลิเคชันจะสร้าง Object multipleRequest ขึ้นมา โดยใช้ค่าต่างๆ คือ travel_id, userID และ multipleRequestLists แล้วเรียกใช้งาน API getMultipleMatchingDetail (Token,multipleRequest)

API ทั้ง 2มีลักษณะการทำงานคล้ายๆกัน จะส่งไปยังเว็บเซิร์ฟเวอร์แล้วระบบจะตอบกลับมาเป็น

1) Object requestDetails ส่วนนี้จะเก็บรายละเอียดของคำขอของ Customer ทั้งหมด โดยจะมี เช่น ชื่อ เบอร์โทร ที่อยู่ พิกัดของผู้รับและผู้ส่ง
2) Object Lumdubb ส่วนนี้จะเก็บลำดับการเดินทางในการให้บริการ โดยจะมี request_id, status ของลำดับนั้นว่าเป็นจุดรับสินค้า (pickup) หรือจุดส่งสินค้า (dropoff)
3) Stage ถ้าหากสถานะ account_status เป็น free แสดงว่า driverพึ่งจะเริ่มให้บริการ จึงเริ่มที่ตำแหน่งเริ่มต้น(ค่าเป็น 0) แต่ถ้าหากสถานะเป็น busy Driver อาจจะเริ่มให้บริการไปแล้วก็ได้จึงส่งค่า stage มาให้ Driver จะได้เริ่มทำงานต่อได้เลยไม่ต้องเริ่มที่จุดเริ่มต้นใหม่

เมื่อได้ข้อมูลมาครบแล้วจะเริ่มทำการเรียงลำดับพิกัดในการให้บริการ เนื่องจาก Object Lumdubb นั้นไม่ได้ส่งค่าพิกัดมาด้วยจึงต้องทำการเทียบกับ request_id ใน Object Lumdubb กับ request_id ใน requestDetails และต้องเทียบสถานะด้วยว่าลำดับนั้นเป็นจุดรับสินค้า หรือจุดส่งสินค้า โดยจะเรียงลำดับไว้ใน List<LatLng> ของ testlatlng

เมื่อเรียงลำดับพิกัดตามการเดินทางในการให้บริการเรียบร้อยแล้วจะทำการสร้าง url ที่ใช้ในการส่งไปยัง google API เพื่อสร้างเส้นทางการเดินทางตามถนนบนแผนที่

จากนั้น Google API จะตอบกลับมาในรูปของ JSON แอพพลิเคชันก็จะแปลง JSON ให้อยู่ในรูปที่แสดงผลบนแผนที่ในแอพพลิเคชันได้คือ polygon

เมื่อแสดงเส้นทางบนแผนที่ได้แล้วจะเพิ่ม Marker ตามจุดต่างๆที่ต้องเดินทางบนแผนที่ในแอพพลิเคชัน โดยในแต่ละ request_id จะมี Marker สีเดียวกัน

จากนั้นเมื่อเดินทางไปถึงยังจุดต่างๆให้ Driver กดปุ่มให้บริการบนแผนที่จะเรียกใช้งาน showpopupfragment() คือจะแสดงรายละเอียดของจุดนั้นๆ

บทที่ 4

การทดลอง และผลการทดลอง

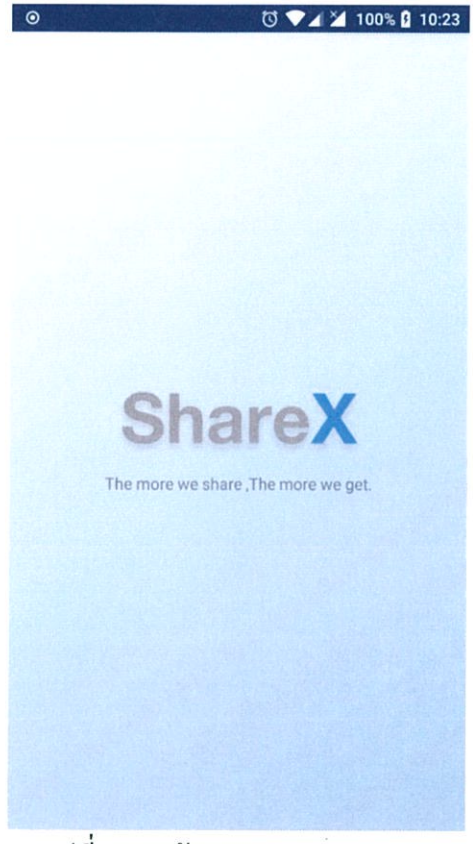
4.1 Wireframe ของแอปพลิเคชันฝั่ง Customer

เป็นการแสดงการทำงานของแอปพลิเคชันฝั่ง Customer แบบภาพรวม โดยจะมีรายละเอียดในแต่ละหน้าในหัวข้อย่อยถัดไป

รูปที่ 4.1 Wireframe ฝั่ง Customer

4.1.1 หน้าต้อนรับ Splash Screen

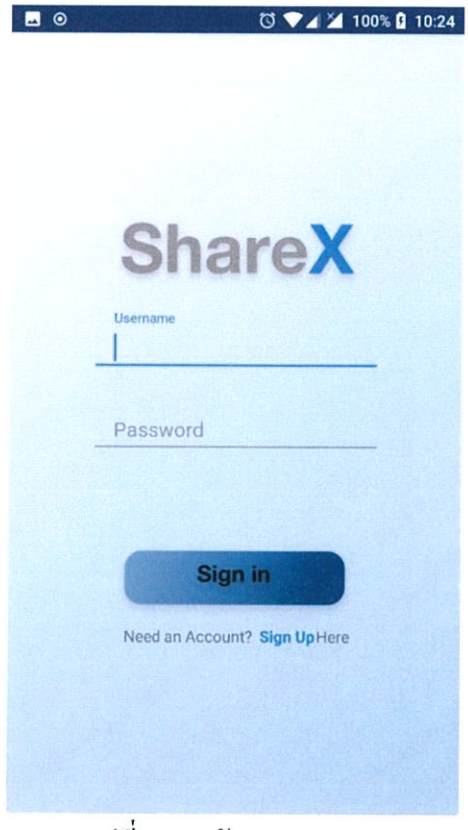
เป็นหน้าแสดงสัญลักษณ์ของแอปพลิเคชันเมื่อผู้ใช้ทำการเข้าใช้งานแอปพลิเคชัน



รูปที่ 4.2 หน้า Splash Screen

4.1.2 หน้าลงชื่อเข้าใช้

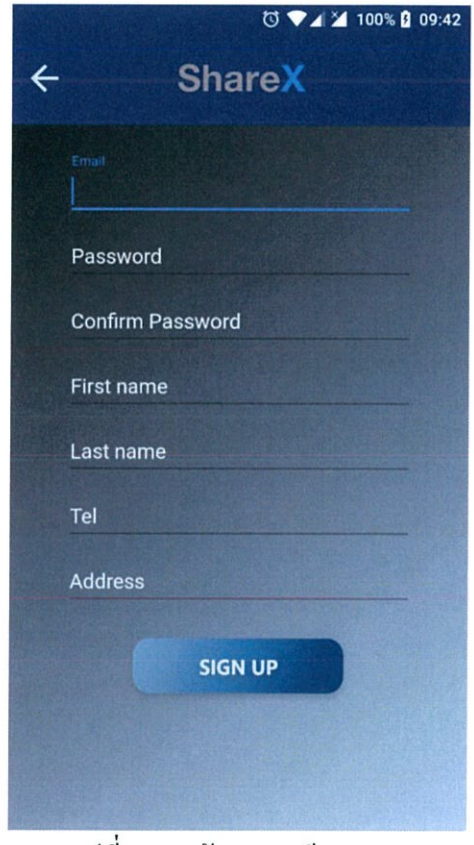
เมื่อผู้ใช้เข้าใช้งานแอปพลิเคชัน หลังจากปรากฏหน้าแรกเสร็จแล้วทางแอปพลิเคชันจะนำมายังหน้าลงชื่อเข้าใช้อัตโนมัติ



รูปที่ 4.3 หน้า Log in

4.1.3 หน้าลงทะเบียน

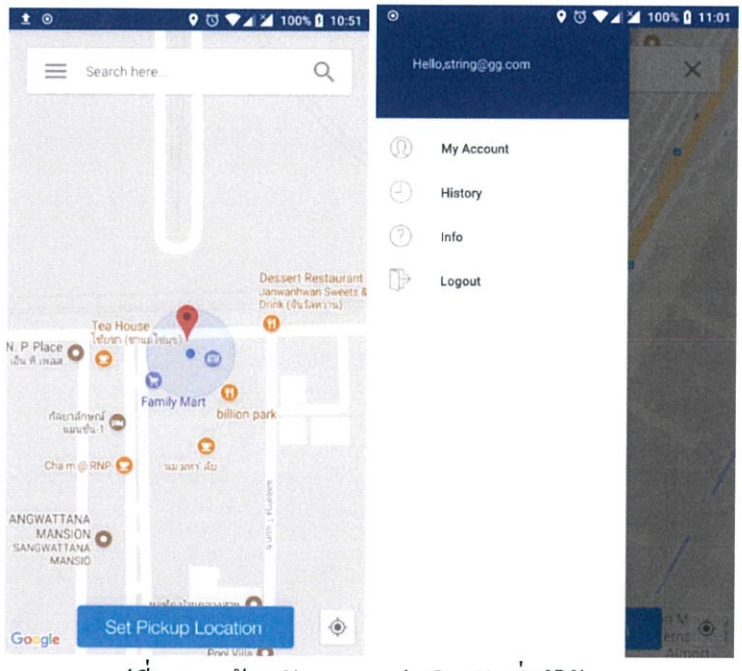
ถ้าหากผู้ใช้ยังไม่มีบัญชีผู้ใช้สามารถทำการสมัครสมาชิกได้ โดยกด Sign Up ที่หน้า Log in จะมายังหน้าสมัครสมาชิก



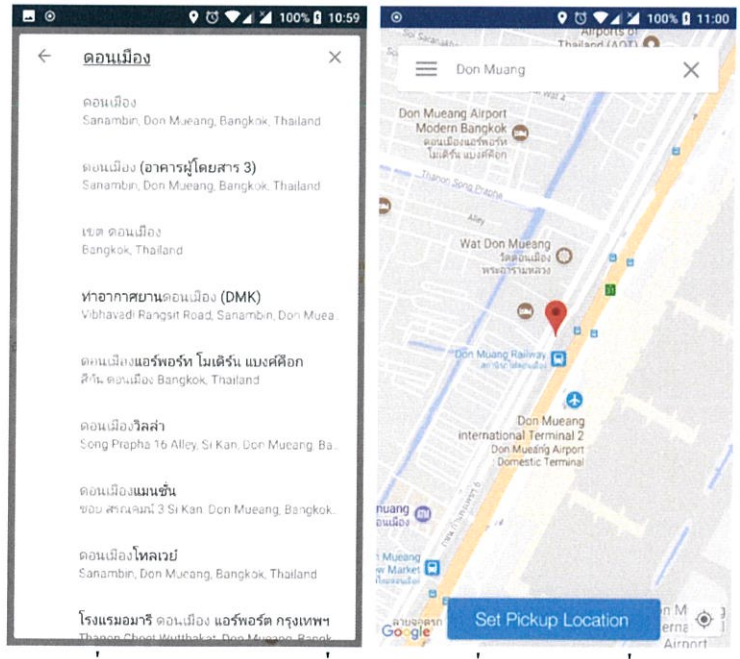
รูปที่ 4.4 หน้าลงทะเบียน

4.1.4 หน้าหลัก

เมื่อผู้ใช้ทำการลงชื่อเข้าใช้เสร็จเรียบร้อยแล้วจะแสดงหน้าหลักของแอปพลิเคชัน โดยสามารถทำการสร้างคำขอใช้บริการได้โดยกำหนดจุดรับสินค้าบนแผนที่ หรือค้นหาจากชื่อสถานที่ได้ นอกจากนี้ยังมี Navigation Drawer เพื่อเข้าถึงหน้าเมนูต่างๆได้ โดยผู้ใช้ทำการกดที่ปุ่ม Hamburger Menu หรือทำการปัดจอไปทางซ้ายจะปรากฏหน้าเมนูขึ้นมา



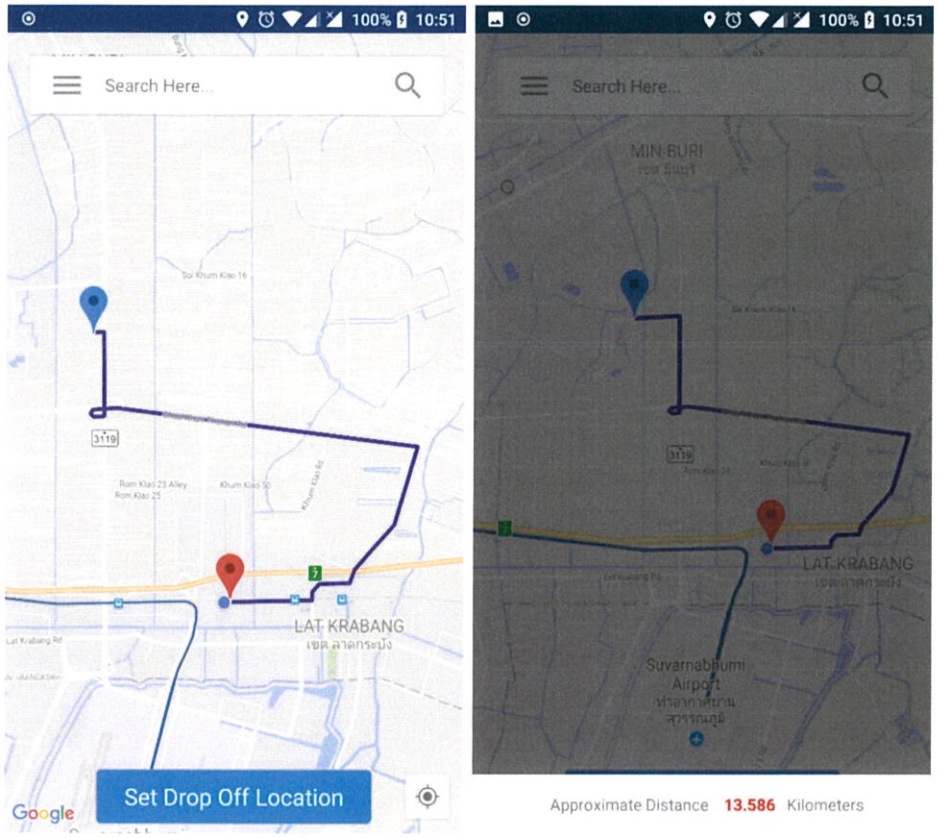
รูปที่ 4.5 หน้าหลักของแอปพลิเคชันฝั่งผู้ใช้



รูปที่ 4.6 หน้าแสดงผลเมื่อค้นหาด้วยชื่อของสถานที่

4.1.5 หน้ากำหนดจุดส่งสินค้า

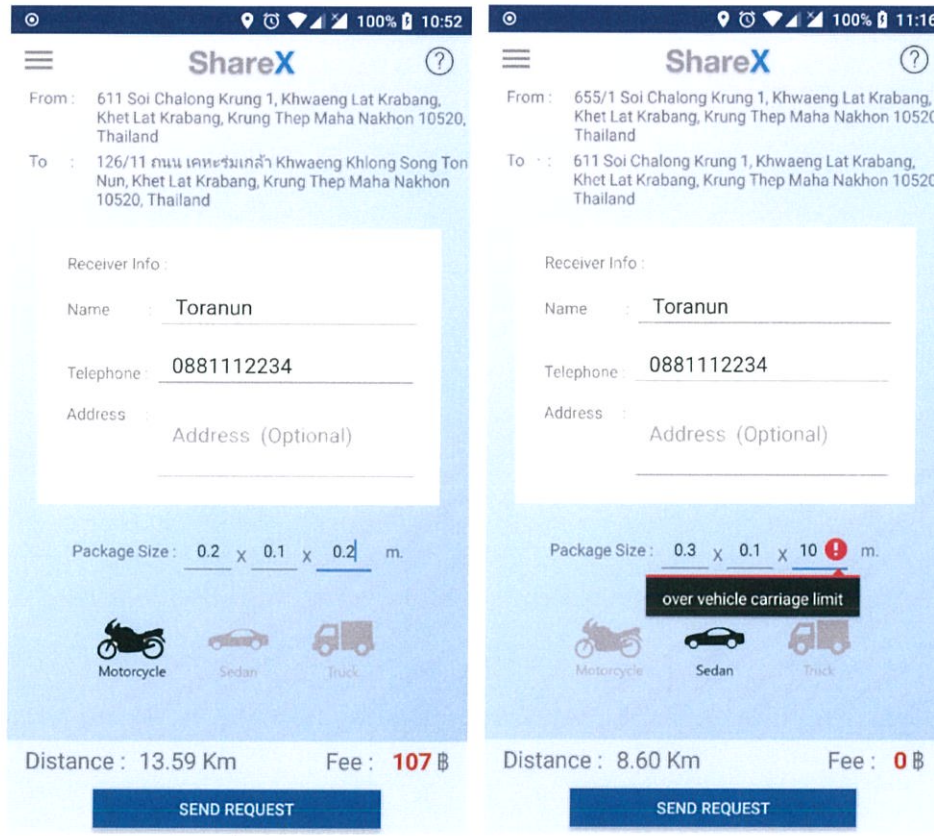
เมื่อผู้ใช้แอปพลิเคชันทำการกำหนดจุดรับสินค้าเสร็จเรียบร้อยแล้ว จะมายังหน้ากำหนดจุดส่งสินค้าโดยผู้ใช้สามารถกำหนดจุดส่งสินค้าได้ในวิธีเดียวกันกับการกำหนดจุดรับสินค้า คือ เลือกจุดบนแผนที่ หรือทำการค้นหาจากชื่อของสถานที่ก็ได้ เมื่อกำหนดจุดส่งสินค้าเรียบร้อยแล้ว แอปพลิเคชันจะแสดงเส้นทางตามการเดินทางบนถนนจริง และบอกระยะทางที่ใช้



รูปที่ 4.7 หน้ากำหนดจุดส่งสินค้า

4.1.6 หน้ากรอกรายละเอียดของผู้รับสินค้า

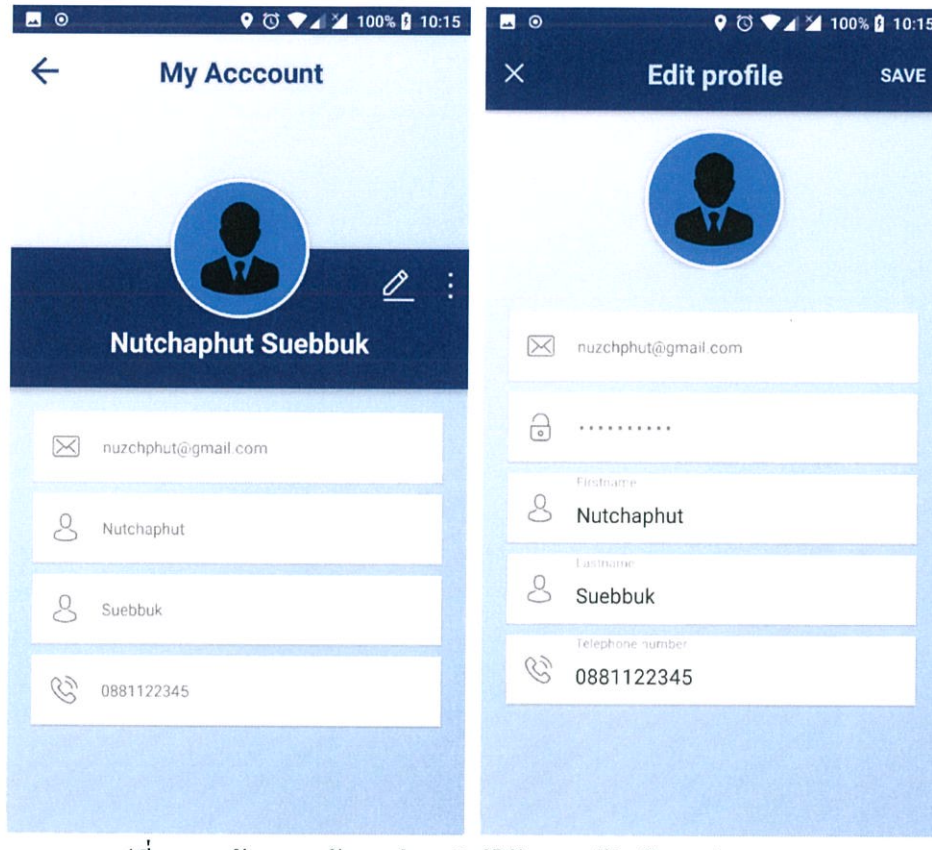
เมื่อผู้ใช้กำหนดจุดรับ-ส่งสินค้าเสร็จเรียบร้อยแล้วจะมายังหน้ากรอกรายละเอียดของผู้รับสินค้า โดยผู้ใช้ต้องทำการกรอกชื่อ เบอร์โทรศัพท์ของผู้รับ กรอกขนาดของสินค้า โดยแต่ละชนิดของยานพาหนะที่ผู้ใช้เลือกนั้นมีข้อจำกัดของขนาดสินค้าที่สามารถรับได้ และเลือกชนิดยานพาหนะที่ต้องการใช้ จากนั้นระบบจะคำนวณค่าบริการโดยอ้างอิงจาก ระยะทาง ขนาดของสินค้า และชนิดของยานพาหนะ



รูปที่ 4.8 หน้ากรอกรายละเอียดของผู้รับสินค้า

4.1.7 หน้าแสดงข้อมูลส่วนตัวผู้ใช้ และแก้ไขข้อมูลส่วนตัว

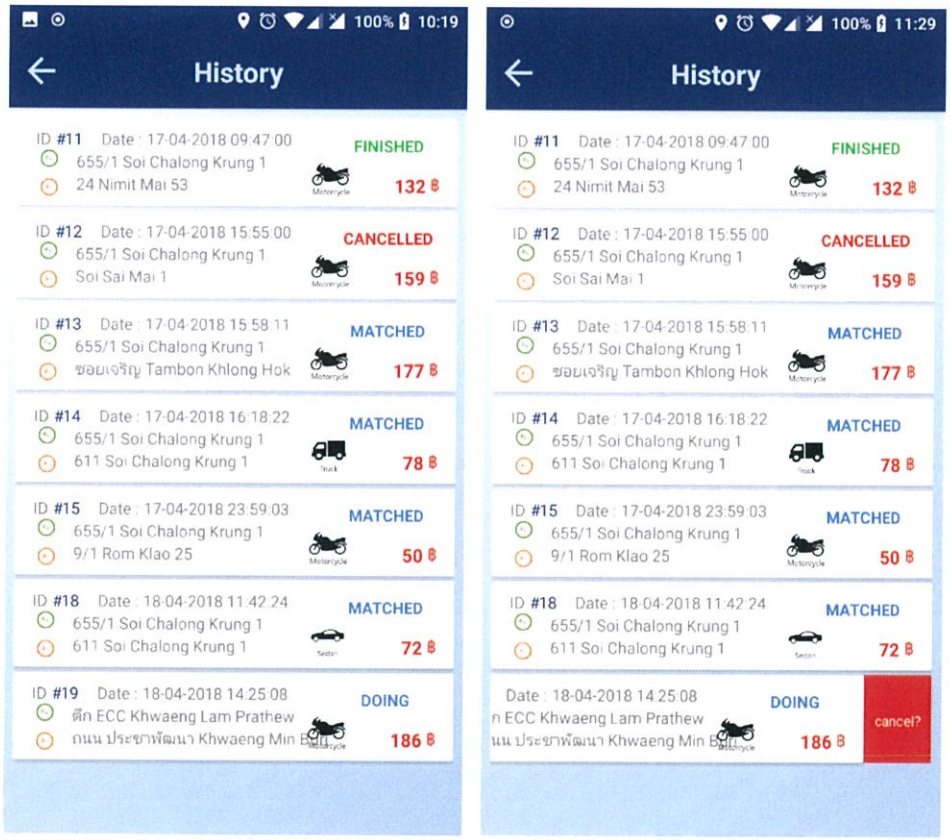
เมื่อผู้ใช้เปิด Navigation Drawer เพื่อเข้าถึงเมนูต่างๆ และกด My Account จะมายังหน้าแสดงรายละเอียดของผู้ใช้ และในหน้านี้ผู้ใช้สามารถกดปุ่มแก้ไขข้อมูลเพื่อทำการแก้ไขข้อมูลส่วนตัวได้ เช่น ชื่อ-นามสกุล เบอร์โทร เป็นต้น



รูปที่ 4.9 หน้าแสดงข้อมูลส่วนตัวผู้ใช้ และแก้ไขข้อมูลส่วนตัว

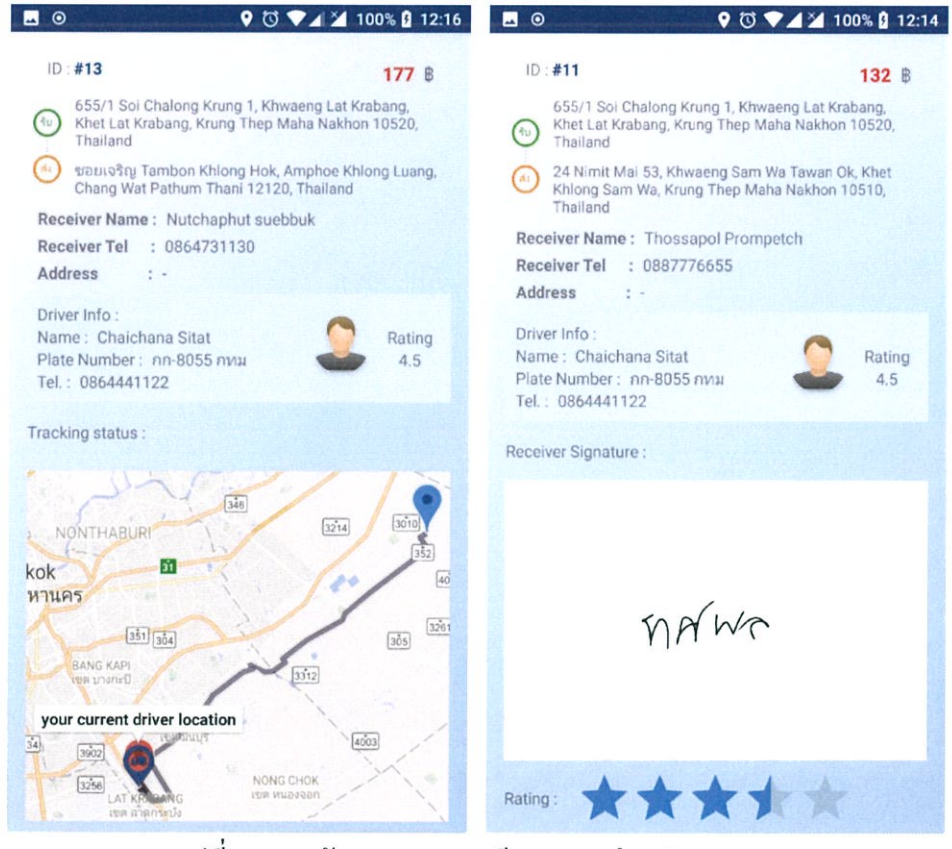
4.1.8 หน้าแสดงประวัติคำขอของผู้ใช้

เมื่อผู้ใช้เปิด Navigation Drawer เพื่อเข้าถึงเมนูต่างๆ และกด Histroy จะมายังหน้าแสดงประวัติคำขอของผู้ใช้ โดยจะแสดงรายละเอียดของแต่ละคำขอและมีสถานะของคำขอนั้นๆ ด้วย คำขอของผู้ใช้ที่ยังไม่ได้รับการตอบรับจาก Driver จะมีสถานะว่า DOING โดยผู้ใช้สามารถทำการยกเลิกคำขอได้ ในคำขอที่มีสถานะว่า Doing เท่านั้น หากคำขอนั้นได้รับการตอบรับจาก Driver แล้ว จะแสดงสถานะว่า MATCHED ไม่สามารถยกเลิกคำขอได้ คำขอที่ผู้ใช้ทำการยกเลิก จะมีสถานะว่า CANCELLED และคำขอใดที่เสร็จสิ้นแล้วจะแสดงสถานะว่า FINISHED



รูปที่ 4.10 หน้าแสดงประวัติคำขอของผู้ใช้

เมื่อผู้ใช้กดในแต่ละคำขอ จะมายังหน้าแสดงรายละเอียดของแต่ละคำขอนั้นๆ ถ้าคำขอใดได้รับการตอบรับจาก Driver แล้ว มีสถานะว่า MATCHED บนแผนที่จะแสดงตำแหน่งปัจจุบันของ Driverให้สามารถทำการติดตามตำแหน่งได้ หากสถานะของคำขอนั้นๆเป็น FINISHED จะแสดงลายมือชื่อของผู้รับและคะแนนความพึงพอใจของบริการ



รูปที่ 4.11 หน้าแสดงรายละเอียดของแต่ละคำขอ

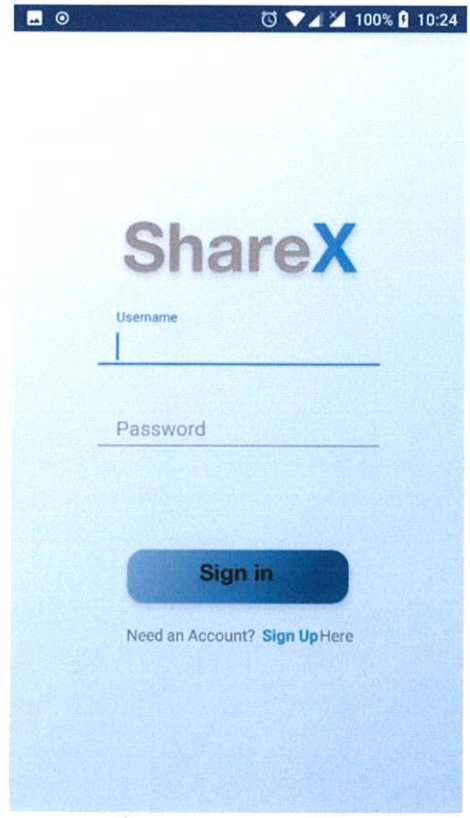
4.2 Wireframe ของแอปพลิเคชันฝั่ง Driver

เป็นการแสดงการทำงานของแอปพลิเคชันฝั่ง Driver แบบภาพรวม โดยจะมีรายละเอียดในแต่ละหน้าของแอปพลิเคชันในหัวข้อย่อยถัดไป

รูปที่ 4.12 Wireframe ฝั่ง Driver

4.2.1 หน้าลงชื่อเข้าใช้

จะเป็นหน้าเดียวกันกับหน้าลงชื่อเข้าใช้ของ Customer



รูปที่ 4.13 หน้า Log in

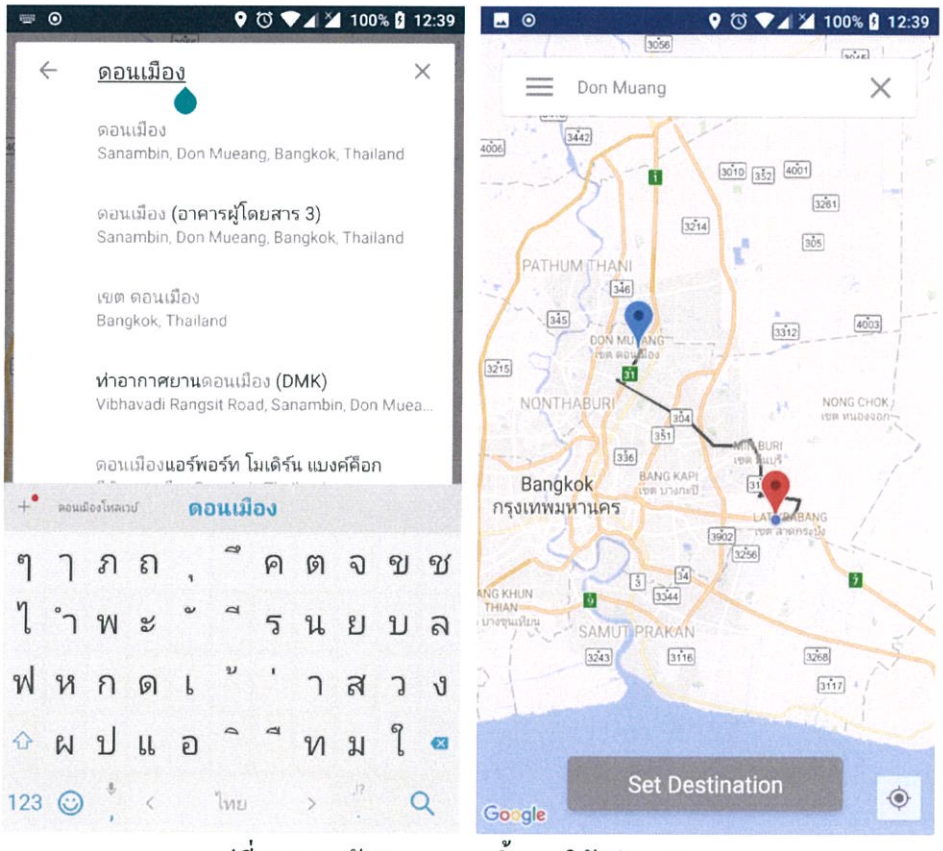
4.2.2 หน้าหลักของ Driver

เมื่อ Driver ทำการลงชื่อเข้าใช้เสร็จเรียบร้อยแล้วจะแสดงหน้าหลักของแอปพลิเคชัน โดยสามารถทำการสร้างเส้นทางการให้บริการได้โดยกำหนดจุดเริ่มต้นการให้บริการบนแผนที่ หรือค้นหาจากชื่อสถานที่ได้

รูปที่ 4.14 หน้าหลักของ Driver

4.2.3 หน้ากำหนดจุดสิ้นสุดให้บริการ

เมื่อ Driver กำหนดจุดเริ่มต้นให้บริการเสร็จเรียบร้อยแล้ว จะมายังหน้ากำหนดจุดสิ้นสุดการให้บริการ โดยสามารถกำหนดจุดบนแผนที่ หรือค้นหาจากชื่อสถานที่ได้



รูปที่ 4.15 หน้ากำหนดจุดสิ้นสุดให้บริการ

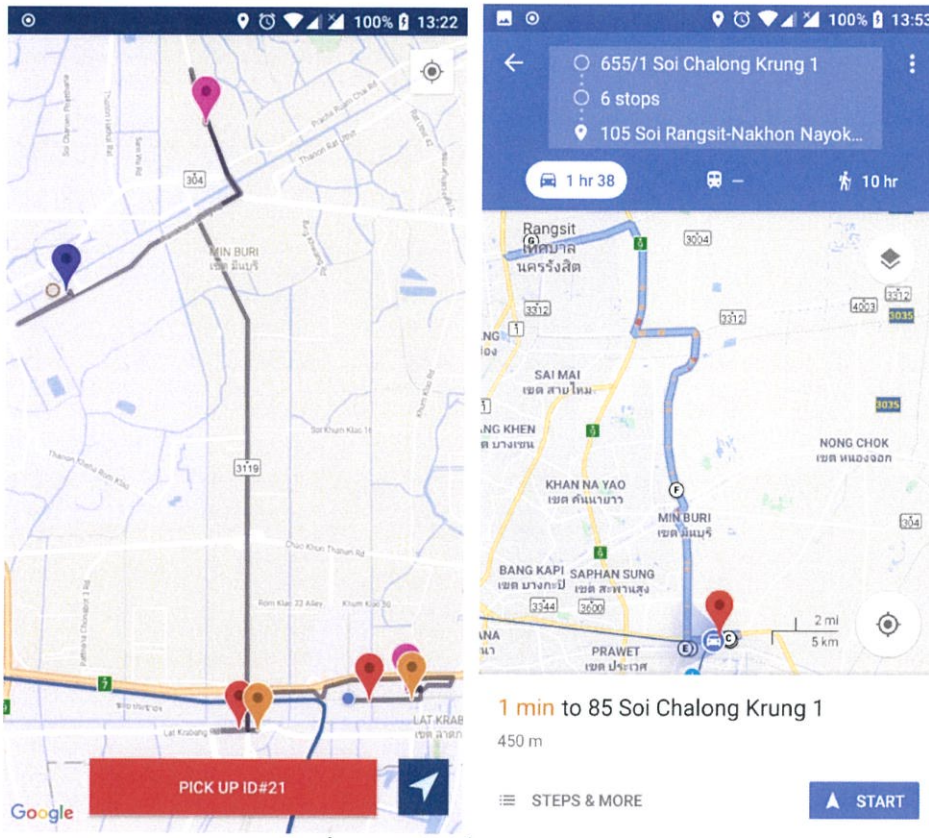
4.2.4 หน้าเลือกให้บริการคำขอของ Customer

เมื่อ Driver กำหนดเส้นทางการให้บริการเสร็จเรียบร้อยแล้ว จะมายังหน้าเลือกให้บริการคำขอจาก Customer โดยจะแสดงรายละเอียดของ Customer ที่ช่วยในการตัดสินใจ คือ ระยะทาง และ ค่าบริการ โดยระบบจะแสดงคำขอที่ใช้ระยะทางเพิ่มขึ้นจากเดิมน้อยที่สุดก่อน ทั้งนี้ ณ ปัจจุบัน Driver สามารถทำการเลือกคำขอจาก Customer ได้มากที่สุดเพียง 4คำขอต่อครั้งเท่านั้น เนื่องจากข้อจำกัดของ API ที่ทางผู้จัดทำเลือกใช้

รูปที่ 4.16 หน้าเลือกให้บริการคำขอของ Customer

4.2.5 หน้าเริ่มการให้บริการ

เมื่อ Driver เลือกคำขอของ Customer ที่จะทำบริการแล้ว จะส่งคำขอนั้นไปคำนวณเส้นทางการให้บริการที่เซิร์ฟเวอร์ จากนั้นจะแสดงเส้นทาง และจุดบนแผนที่ โดยแต่ละคำขอจะมีสีของ Marker สีเดียวกัน จะมีปุ่มสำหรับนำทางตามจุดต่างๆ โดยเมื่อเรากดปุ่มนั้นจะนำไปยัง Google Map Application เพื่อทำการนำทางต่อไป



รูปที่ 4.17 หน้าเริ่มให้บริการ

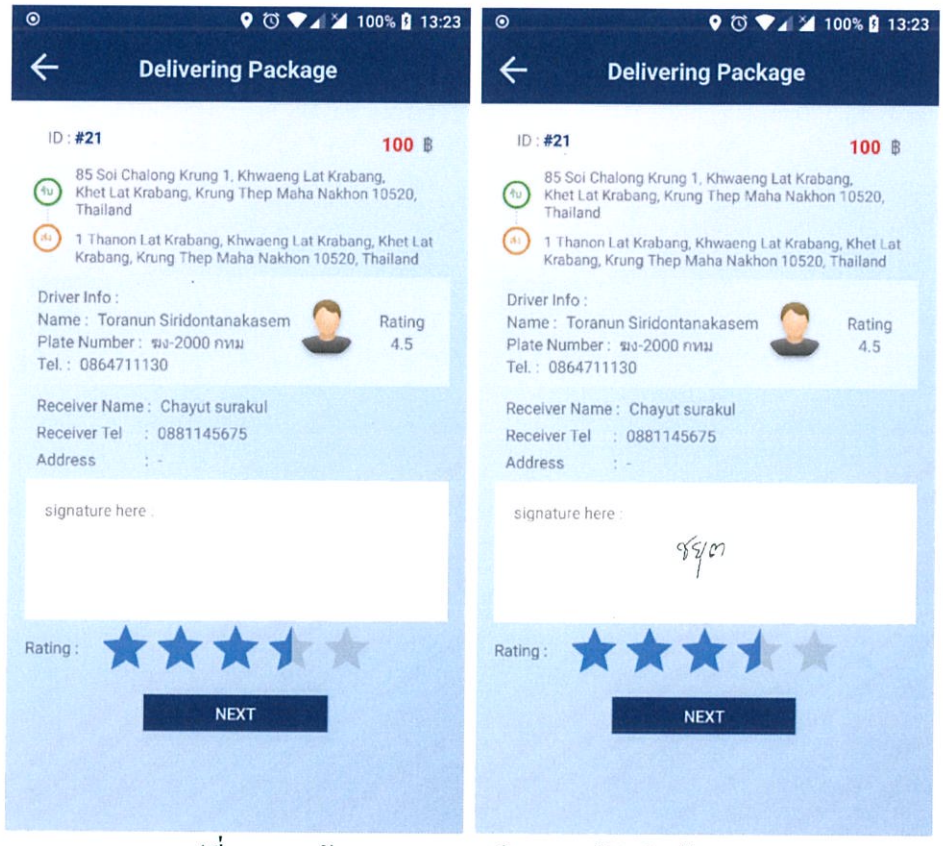
4.2.6 หน้าแสดงรายละเอียดของ Customer

เมื่อ Driver มาถึงยังตำแหน่งของจุดรับสินค้า กดปุ่ม PICK UP ID#... เพื่อแสดงรายละเอียดของ Customer

รูปที่ 4.18 หน้าแสดงรายละเอียดของ Customer

4.2.7 หน้าแสดงรายละเอียดของผู้รับสินค้า

เมื่อ Driver มาถึงยังตำแหน่งของจุดส่งสินค้า กดปุ่ม DROP OFF ID#... เพื่อแสดงรายละเอียดของผู้รับสินค้า และให้ผู้รับสินค้าทำการยืนยันการรับสินค้าโดยการลงลายมือชื่อและให้คะแนนการให้บริการ



รูปที่ 4.19 หน้าแสดงรายละเอียดของผู้รับสินค้า

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 บทสรุปของโครงงาน

ในส่วนของแอปพลิเคชัน ณ ปัจจุบันนั้นสามารถทำงานได้ตามความต้องการของผู้ใช้แล้วเป็นส่วนใหญ่ แต่ยังมีข้อจำกัดเรื่องจำนวนคำขอที่ Driver สามารถให้บริการได้เพียง 4คำขอต่อครั้งเท่านั้นเนื่องจากข้อจำกัดของ API ที่ทางผู้จัดทำเลือกใช้ คือ RouteXL API ซึ่งเป็น API ที่ช่วยในการจัดลำดับการเดินทางไปยังจุดต่างๆให้มีระยะทางน้อยที่สุดได้ ถ้าหากต้องการให้มีคำขอได้มากขึ้นนั้นต้องเป็นสมาชิกแบบเสียเงินรายเดือนคิดเป็นเงินประมาณ1,300บาทต่อเดือน แต่ถ้าใช้งานแบบสมาชิกแบบเสียเงินนั้น จะสามารถรับคำขอได้มากถึง50คำขอต่อครั้ง ซึ่งทางผู้จัดทำเห็นว่าโครงงานนี้เป็นเพียงการทดลองให้สามารถใช้งานได้ ซึ่งถ้ามีโอกาสพัฒนาต่อ จะใช้งานเป็นสมาชิกแบบเสียเงินและขยายการบริการจากคำขอมากที่สุด 4 คำขอต่อการบริการ 1ครั้ง ให้มากขึ้นอีกได้

นอกจากนี้ข้อจำกัดอีกข้อหนึ่ง คือ จำนวนอุปกรณ์แอนดรอยด์ที่มีอยู่ใน ปัจจุบันนั้นมีอยู่มากมาย ทั้ง Version ของแอนดรอยด์ และขนาดของหน้าจอที่หลากหลาย การแสดงผลในปัจจุบัน ถ้าหาก นำไปใช้ในอุปกรณ์อื่นที่ไม่ใช่ Android 8.1.0 API 27 ขนาดหน้าจอ 5.15นิ้ว ความละเอียด 1080*1920 pixel ที่ทางผู้จัดทำได้มาทดลองใช้ขณะการพัฒนานั้น อาจจะมีการทำงานของแอปพลิเคชันที่ผิดเพี้ยนเนื่องจากขนาดหน้าจอ หรือแสดงสมรรถะได้ไม่เต็มประสิทธิภาพได้ และ ทางผู้จัดทำกังวลมากที่สุด คือ ในแอปพลิเคชันนี้นั้น ไม่สามารถทำการนำทางไปยังจุดต่างๆได้เอง ต้องอาศัยแอปพลิเคชัน Google Map ในการนำทางให้ ซึ่งถ้าใช้งานทั้ง 2 แอปพลิเคชันพร้อมกันนั้น อัตราการใช้หน่วยความจำ(RAM) และอัตราการใช้แบตเตอรี่จะค่อนข้างสูง

5.2 ปัญหาและอุปสรรค

1) โครงงานที่ออกแบบไว้ในตอนแรกนั้นเมื่อเริ่มพัฒนาจะเห็นได้ว่าการมีปัญหาตามมา ภายหลัง จึงต้องออกแบบส่วนที่มีปัญหาใหม่
2) ที่อยู่ที่แปลงจากค่าพิกัดนั้น ยังมีความคลาดเคลื่อนอยู่หากใช้งานจริงอาจจะมีอุปสรรคใน การขนส่ง เช่นอยู่ในหอที่ตึกไกล้ๆกัน จะแสดงที่อยู่อย่างเดียว ไม่แสดงชื่อตึก
3) ไม่สามารถทำการนำทาง (Navigate) ในแอปพลิเคชันได้ต้องส่งพิกัดไปยัง Google Map แอปพลิเคชันเพื่อทำการนำทาง
4) Driver สามารถให้บริการได้มากที่สุดเพียง 4 คำขอต่อครั้ง

5) อุปกรณ์แอนดรอยด์ที่นำมาทดสอบนั้นยังไม่หลากหลาย หากนำไปใช้งานจริง ต้องทดสอบกับอุปกรณ์แอนดรอยด์รุ่นอื่นๆอีกเพื่อทดสอบความเสถียรของแอปพลิเคชัน และแสดงผลได้ตามที่ทางผู้จัดทำคาดหวังไว้

5.3 แนวทางการพัฒนาต่อ

1) พัฒนาแอปพลิเคชันให้มีความยืดหยุ่นกับอุปกรณ์แอนดรอยด์อื่นๆ ที่มีระบบปฏิบัติการ หน่วยประมวลผล และหน่วยความจำที่ต่างกัน
2) พัฒนาอัลกอริทึมที่ช่วยในการแก้ไขปัญหาลำดับการเดินทางก่อนหลังแทนที่การเรียกใช้ API ภายนอก
3) พัฒนาระบบชำระเงินออนไลน์จากฝั่ง Customer เมื่อคำขอนั้นเสร็จสิ้นจึงโอนเงินไปยัง Driver เนื่องจากความรู้ด้านความปลอดภัยและการจัดการข้อมูลส่วนตัวของผู้จัดทำนั้นยังมีน้อย ต้องศึกษาให้รอบคอบก่อนพัฒนา

บรรณานุกรม

Amplysoft. 2013. **เริ่มต้นการเขียนเว็บไซต์ ทำเว็บไซต์ด้วยภาษา Python กับ Django Framework**.[Online].Available:http://www.amplysoft.com/knowledge/what-is-django-framework-python.html.

Android Developer. 2017. **Meet Android Studio**. [Online].
Available : https://developer.android.com/studio/intro/index.html.

Android Developer. 2017. **Introduction to Android.** [Online].
Available : https://developer.android.com/guide/index.html.

Wasin Thiengkunakrit. 2017. **เริ่มพัฒนา Web Application กับภาษา Python ด้วย Django Framework**. [Online]. Available : https://goo.gl/3CSuiB.

Macfeteria. 2013. **SQLite**. [Online].
Available : http://macfeteria.com/tag/sqlite.

SQLite. 2017. **SQLite IS Serverless**. [Online].
Available : https://www.sqlite.org/serverless.html.

Google Developers. 2017. **Google Maps Directions API**. [Online].
Available : https://developers.google.com/maps/documentation/directions/intro.

Google Developers. 2017. **Google Places API**. [Online].
Available : https://developers.google.com/places/web-service.

Google Developers. 2017. **LocationAPI**. [Online].
Available : https://developers.google.com/maps/documentation/android-api/location.

Mindphp. 2017. **Java คืออะไร.** [Online].
Available : https://goo.gl/trxG1M.

Github. 2017. **Google-Direction-Android**. [Online].
Available : https://github.com/jd-alexander/Google-Directions-Android.

Akexorcist. 2015. **[Android Code] ใช้งาน Google Maps Direction API ให้ง่ายๆขึ้นด้วย Google Direction Library**. [Online]. Available : http://www.akexorcist.com/2015/12/google-direction-library-for-android-th.html.

Google Developers. 2018. **Firebase Cloud Messaging**. [Online].
Available : https://firebase.google.com/docs/cloud-messaging/

Google Developers. 2018. **Firebase Realtime Database**. [Online].
Available : https://firebase.google.com/docs/database/.

RouteXL. 2018. **RouteXL API**. [Online].
Available : https://www.routexl.nl/blog/api/.

Jirawatee. 2016. **รู้จัก Firebase Cloud Messaging (FCM) ตั้งแต่ Zero จนเป็น Hero**. [Online].
Available : https://goo.gl/VZrb1q.

Tutorialpoints. 2018. **Python Overview**. [Online].
Available : https://www.tutorialspoint.com/python/python_overview.htm.