การออกแบบและการพัฒนาเว็บเซอร์วิสสำหรับ
ระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์

สิปางญ์ กันทะเสมา

สารนิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตร์มหาบัณฑิต
สาขาวิชาวิศวกรรมเว็บ วิทยาลัยครีเอทีฟดีไซน์
แอนด์ เอ็นเตอร์เทนเมนต์เทคโนโลยี
มหาวิทยาลัยธุรกิจบัณฑิตย์
พ.ศ. 2560

A Design and Development of Web Services for Messenger Services with Real-Time Tracking

Sipang Kanthasem

Thematic Paper Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science in Web Engineering,

College of Creative Design and Entertainment

Technology, Dhurakij Pundit University

2017

หัวข้อสารนิพนธ์	การออกแบบและการพัฒนาเว็บเซอร์วิสสำหรับ ระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์
ชื่อผู้เขียน	สิปางณ์ กันทะเสมา
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.วรสิทธิ์ ชูชัยวัฒนา
สาขา	วิศวกรรมเว็บ
ปีการศึกษา	2559

บทคัดย่อ

ในปัจจุบันธุรกิจประเภทขนส่งสินค้าหรือโลจิสติกมีความต้องการมากขึ้นตามการขยายตัวของธุรกิจอีคอมเมิร์ซ ทำให้เกิดบริการประเภทรับส่งของขึ้นมากมายและที่กำลังเป็นที่นิยมในปัจจุบันคือการรับส่งของแบบเร่งด่วนหรือ Same Day Delivery ซึ่งผู้รับจะได้รับของภายในวันที่ส่งของ สำหรับของที่มีขนาดไม่ใหญ่มากที่สามารถบรรทุกด้วยรถจักรยานยนต์ ซึ่งเป็นภาหนะที่สามารถเดินทางได้ถึงจุดหมายอย่างรวดเร็วแม้ในเวลาเร่งด่วน ซึ่งผู้ประกอบการธุรกิจขนส่งประเภทนี้มีความต้องการที่จะเพิ่มความสามารถในการให้บริการ รวมถึงความต้องการในการติดตามสถานการณ์จัดส่งของผู้ใช้บริการ

สารนิพนธ์นี้จึงนำเสนอ การวิเคราะห์และออกแบบการพัฒนาเว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์ เพื่อตอบสนองความต้องการของผู้ใช้งานและสนับสนุนนักพัฒนาในฝั่งไคลเอนท์ให้สามารถเชื่อมต่อรับส่งข้อมูลเว็บเซอร์วิสได้อย่างสะดวก ซึ่งในการพัฒนาได้นำแนวคิด รวมถึงเทคโนโลยีการสื่อสารผ่านเครือข่ายที่เกี่ยวข้อง เช่น REST API, WebSocket เทคโนโลยี GSP และ รูปแบบข้อมูล JSON รวมถึงการนำ OpenAPI Specification มาประยุกต์ใช้เพื่อให้เกิดประโยชน์ในการพัฒนาระบบซอฟต์แวร์เพื่อสนับสนุนการดำเนินธุรกิจโลจิสติกได้อย่างมีประสิทธิภาพ

Thematic Paper Title	A Design and Development of Web Services for Messenger Services with Real-Time Tracking
Author	Sipang Kanthasema
Thematic Paper Advisor	Asst. Prof. Dr. Worasit Choochaiwattana
Academic Program	Web Engineering
Academic Year	2016

ABSTRACT

Nowadays, logistic or delivery service business is in demand and growing rapidly along with the expansion of e-Commerce businesses, many kinds of delivery services have emerged and one of that is the Same Day Delivery or the express delivery by messenger with motorcycle which is the convenient vehicle that can reach the destination shortly in rush hour despite of huge traffic. Due to it's demanded and delivery business owner wants to improve their services to serve their client and the need of users to track the delivery status.

The purpose of this study is to develop Web Services for Messenger Services with Real-Time Tracking to serve users' needs in express delivery services and to support developers for client application to connect with Web Services smoothly. The technology used in this study is computer network communication technology such as REST API, WebSocket, GSP and JSON data format including OpenAPI Specification. The study applies all these technologies and develop a software system to support logistic business.

กิตติกรรมประกาศ

การจัดทำสารนิพนธ์เรื่อง การออกแบบและการพัฒนาเว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์ ครั้งนี้สำเร็จลุล่วงได้ด้วยดีโดยได้รับความช่วยเหลือและการสนับสนุนจากผู้มีความรู้หลายๆท่าน โดยเฉพาะอย่างยิ่ง ผศ.ดร.วรสิทธิ์ ชูชัยวัฒนา ซึ่งเป็นอาจารย์ปรึกษา ที่ได้เสียสละเวลาให้คำแนะนำ ให้ความรู้ด้านวิชาการ และเทคนิคต่างๆ ตลอดจนข้อคิดเห็นที่เป็นประโยชน์ในการแก้ไขปรับปรุงผลงานสารนิพนธ์ฉบับนี้ให้สมบูรณ์มากยิ่งขึ้น ผู้จัดทำซาบซึ้งในความกรุณาของท่านเป็นอย่างยิ่งและขอกราบขอบพระคุณท่านเป็นอย่างสูง

ขอขอบคุณเพื่อนๆ ร่วมรุ่นทุกๆคน รวมถึงเพื่อนๆที่ทำงาน ที่คอยให้ความช่วยเหลือซึ่งกันและกันมาตลอด ระยะเวลาการศึกษา

ขอกราบขอบพระคุณคุณพ่อคุณแม่ ครอบครัวและบุคคลอันเป็นที่รัก ที่เป็นกำลังใจอันสำคัญยิ่งในการจัดทำสารนิพนธ์จนประสบความสำเร็จลุล่วงด้วยดี ซึ่งทุกท่านจะถูกจารึกไว้ในจิตใจของผู้จัดทำสารนิพนธ์ตลอดไป

ในท้ายที่สุดนี้ผู้จัดทำหวังว่าผลงานสารนิพนธ์ฉบับนี้จะเป็นประโยชน์กับผู้ที่ต้องการศึกษาด้านการพัฒนาระบบจัดการสารสนเทศ และหากมีข้อผิดพลาดประการใดในงานสารนิพนธ์ ฉบับนี้ผู้จัดทำต้องขอกราบขออภัยเป็นอย่างสูงมา ณ ที่นี้ด้วย

สิปางญ์ กันทะเสมา

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญตาราง (ต่อ)

ตารางที่ **หน้า**

สารบัญภาพ

ภาพที่ **หน้า**

สารบัญภาพ (ต่อ)

ภาพที่ **หน้า**

สารบัญภาพ (ต่อ)

ภาพที่ **หน้า**

สารบัญภาพ (ต่อ)

ภาพที่ **หน้า**

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของงาน

ในปัจจุบันการดำเนินธุรกิจมีการแข่งขันสูงมาก ไม่ว่าจะเป็นการแข่งขันภายในประเทศและจากต่างประเทศ ส่งผลให้ธุรกิจมีการเปลี่ยนแปลงไปอย่างรวดเร็ว ดังนั้น องค์กรธุรกิจจึงจำเป็นต้องมีการแสวงหาแนวทางใหม่เพื่อมาสนับสนุนการพัฒนาธุรกิจให้สามารถแข่งขันกับองค์กรธุรกิจอื่น ๆได้ การใช้เทคโนโลยีสารสนเทศมาใช้ในการบริหารจัดการองค์กรก็จัดเป็นวิธีการที่สามารถสนับสนุนการดำเนินงานธุรกิจได้ดีขึ้น แต่การเลือกเทคโนโลยีสารสนเทศมาใช้ สิ่งหนึ่งที่จะต้องคำนึงถึงก็คือเทคโนโลยีนั้นจะต้องเปิดกว้างให้ผู้มีส่วนได้ส่วนเสียกับองค์กรสามารถนำข้อมูลใช้ไปให้เกิดประสิทธิภาพและประสิทธิผลมากที่สุด

ในวงการไอทีในปัจจุบันได้มีการปรับเปลี่ยนสู่ยุคแนวคิดเชิงบริการ (Service Orientation) เพื่อเป็นการตอบสนองต่อความต้องการของธุรกิจในยุคปัจจุบัน และองค์กรทางธุรกิจส่วนใหญ่ต่างก็ปรับเปลี่ยนสถาปัตยกรรมของระบบซอฟต์แวร์ของตนเองให้เป็นเชิงบริการ SOA (Service-Oriented Architecture) ซึ่ง SOA เป็นรูปแบบในการออกแบบระบบ (system) หรือโปรแกรมประยุกต์ (application) ที่จะมองระบบประกอบด้วยการทำงานหรือบริการ (service) ต่าง ๆ และเมื่อมีการนำแนวคิดนี้ผสานเข้ากับเทคโนโลยีการสื่อสารและอินเตอร์เน็ต จึงทำให้เกิดการพัฒนาเป็นเว็บเซอร์วิสขึ้น

เว็บเซอร์วิส (Web service) หรือ Web API คือระบบซอฟต์แวร์ที่ออกแบบมา เพื่อสนับสนุนการแลกเปลี่ยนข้อมูลกัน ระหว่างเครื่องคอมพิวเตอร์ผ่านระบบเครือข่าย เพราะการใช้ไอทีในองค์กรไม่ได้จำกัดอยู่เพียงการใช้ซอฟต์แวร์สำเร็จรูปบนเครื่องคอมพิวเตอร์ Desktop แบบ Stand-Alone เพื่อบันทึกข้อมูลของบุคลากร ซึ่งไม่เพียงพอต่อการทำงานที่เพิ่มขึ้นอีกต่อไป แต่ยังจำเป็นต้องวิเคราะห์และบริการจัดการข้อมูล และสร้างบริการในเชิงลึกอีกมาก ดังนั้นเว็บเซอร์วิส จึงก้าวเข้ามามีบทบาทสำคัญและถูกนำมาประยุกต์ใช้ในการบริหารจัดการงานของแทบทุกองค์กร รวมไปถึงธุรกิจประเภทขนส่งสินค้าหรือโลจิสติก ที่มีความต้องการมากขึ้นตามการขยายตัวของธุรกิจอีคอมเมิร์ซ และไม่เพียงความต้องการบริการแบบออนไลน์เท่านั้น ผู้ประกอบการเองยังมีความต้องการที่จะติดตามการขนส่งและการทำงานของพนักงานแบบเรียลไทม์ได้อีกด้วย เพราะการขนส่งอาจเกิดเหตุการณ์ไม่คาดคิดได้ตลอดเวลา เช่นเกิดอุบัติเหตุหรือเกิดภัยภิบัติต่างๆ ทำให้การขนส่งของหรือสินค้าล่าช้าออกไปไม่สามารถส่งได้ทันเวลา หรือไม่สามารถส่งของได้เลยเนื่องจากเกิดการสูญหายหรือการทุจริตของตัวพนักงานเอง ทำให้เกิดความเสียหายกับธุรกิจได้ ทางกรมการขนส่งเองก็มองเห็นถึงปัญหาดังกล่าว จึงมี

นโยบายให้รถขนส่งติดตั้งเครื่องบันทึกการเดินทางของรถ เพื่อกำกับดูแลผู้ขับรถขนส่งให้ปฏิบัติตามกฎหมายอย่างเครงคัด ตามพระราชบัญญัติของกรมการขนส่งทางบก ประกาศไว้ ณ วันที่ 25 มกราคม 2559 เป็นต้นไป ดังนั้น ผู้ประกอบการขนส่งทั้งรถโดยสารและรถบรรทุกที่กำลังจะจดทะเบียนรถใหม่หรือนำรถไปตรวจสภาพเพื่อต่อทะเบียนของกรมขนส่งทางบก ต้องทำการติดตั้งเครื่องบันทึกการเดินทางของรถหรือ GPS พร้อมเครื่องรูดบัตรเพื่อแสดงตัวตน

ทั้งนี้ข้อกำหนดดังกล่าวมีผลบังคับใช้กับรถขนส่งและรถโดยสารที่เป็นรถยนต์ หรือรถบรรทุกขนาดใหญ่เท่านั้น แต่รูปแบบธุรกิจการขนส่งสำหรับผู้ใช้โดยทั่วไปหรือผู้ประกอบการรายย่อยที่ขนาดและปริมาณของของหรือสินค้าที่มีขนาดไม่ใหญ่มากจึงนิยมหันมาใช้บริการขนส่งแบบเร่งด่วนด้วยรถมอเตอร์ไซแทน ซึ่งความต้องการในการติดตามการวิ่งรถเพื่อส่งของหรือสินค้าก็ยังคงเป็นที่ต้องการเช่นกัน

สารนิพนธ์นี้จึงนำเสนอ การวิเคราะห์และออกแบบการพัฒนาเว็บเซอร์วิสสำหรับระบบบริการขนส่งของและติดตามการขนส่งแบบเรียลไทม์ ซึ่งจะเป็นการนำแนวคิดรวมถึงเทคโนโลยีการสื่อสารผ่านเครือข่ายที่เกี่ยวข้อง เช่น RESTAPI, WebSocket เทคโนโลยี GSP และ รูปแบบข้อมูล JSON รวมถึงการนำเครื่องมือในการจัดทำเอกสาร RESTAPI มาประยุกต์ใช้เพื่อให้เกิดประโยชน์ในการพัฒนาระบบซอฟต์แวร์เพื่อสนับสนุนการดำเนินธุรกิจขององค์กรได้อย่างมีประสิทธิภาพ และนอกจากนี้ยังสามารถนำกรณีศึกษานี้ไปเป็นแนวทางในการพัฒนาและปรับปรุงระบบซอฟต์แวร์อื่น ๆ ที่มีลักษณะคล้ายคลึงกันได้อีกต่อไป

1.2 วัตถุประสงค์ของการศึกษา

1. เพื่อศึกษาปัญหาและหาแนวทางในการพัฒนาเว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์
2. เพื่อวิเคราะห์และออกแบบระบบงานให้มีความเหมาะสม และตรงตามความต้องการของธุรกิจประเภทขนส่งหรือโลจิสติก
3. เพื่อสนับสนุนและอำนวยความสะดวกในการพัฒนาระบบในส่วนของผู้ใช้งานระบบฝั่ง Client ให้สามารถรับส่งข้อมูลกับ Server ได้

1.3 ประโยชน์และผลที่คาดว่าจะได้รับ

1. ได้เว็บเซอร์วิสหรือเว็บ API เพื่อเป็นตัวกลางในการทำงานเชื่อมต่อรับส่งข้อมูลกันภายในระบบระหว่าง Server และ Client สำหรับระบบบริการขนส่งของและติดตามการขนส่งแบบเรียลไทม์
2. สามารถติดตามตำแหน่งการวิ่งรถเพื่อส่งของของพนักงานได้แบบ Real-Time โดยการส่งข้อมูลพิกัดตำแหน่งผ่าน WebSocket
3. ช่วยลดเวลาการทำงานให้กับนักพัฒนาฝั่งไคล์ที่ต้องพัฒนาแอปพลิเคชันติดต่อเข้ามาเรียกใช้ Web API ในฝั่ง Server ซึ่งเป็นผลจากการนำเครื่องมือและเทคโนโลยีด้านการจัดทำเอกสารการอธิบาย REST API มาใช้จัดทำเอกสาร

1.4 ขอบเขตการศึกษา

ในการออกแบบและพัฒนาเว็บเซอร์วิสสำหรับระบบบริการขนส่งของและติดตามการขนส่งแบบเรียลไทม์ ได้นำเทคโนโลยีเว็บแอปพลิเคชันมาประยุกต์ใช้เพื่อสนับสนุนการดำเนินการให้เกิดประสิทธิภาพ ซึ่งมีขอบเขตดังนี้

1. พัฒนาเว็บเซอร์วิสหรือเว็บ API เพื่อเป็นตัวกลางในการทำงานเชื่อมต่อรับส่งข้อมูลกันภายในระบบระหว่าง Server และ Client สำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์
2. พัฒนา WebSocket Server แอปพลิเคชัน เพื่อสนับสนุนการเชื่อมต่อรับส่งข้อมูลแบบ Real-Time ระหว่าง Server และ Client และเป็นการติดตามการทำงานของผู้ใช้ในระบบ
3. นำเครื่องมือและเทคโนโลยีด้านการจัดทำเอกสารการอธิบาย REST API มาใช้จัดทำเอกสารเพื่ออำนวยความสะดวกและลดเวลาการทำงานให้กับนักพัฒนาฝั่งไคล์เอนท์ที่ต้องพัฒนาแอปพลิเคชันติดต่อเข้ามาเรียกใช้ Web API ในฝั่ง Server
4. ข้อมูลที่จัดเก็บในฐานข้อมูล
    - 4.1 ข้อมูลผู้ใช้ทั่วไป
    - 4.2 ข้อมูลพนักงานส่งของ
    - 4.3 ข้อมูลผู้ดูแลระบบ
    - 4.4 ข้อมูลคำสั่งงาน
    - 4.5 ข้อมูลพื้นที่ให้บริการ
    - 4.6 ข้อมูลตำแหน่งพิกัดปัจจุบันของพนักงานส่งของ
    - 4.7 ข้อมูลประวัติการทำงานและประวัติรายได้ของพนักงาน
    - 4.8 ข้อมูลคะแนนการปฏิบัติงานของพนักงานจากผู้ใช้
    - 4.9 ข้อมูลการชำระค่าบริการการขนส่งของผู้ใช้
    - 4.10 ข้อมูลโฆษณาและการส่งเสริมการขาย

4.11 ข้อมูลคำถามที่พบบ่อย

4.12 ข้อมูลการขอใบกำกับภาษี

4.13 ข้อมูลอื่นๆที่จำเป็นในระบบ

บทที่ 2
แนวคิด ทฤษฎี และงานวิจัยที่เกี่ยวข้อง

ในการพัฒนาเว็บเซอร์วิส สำหรับระบบบริการรับส่งของ และติดตามการทำงานแบบเรียลไทม์ ผู้วิจัยได้ทำการศึกษาแนวคิด ทฤษฎี และงานวิจัยที่เกี่ยวข้อง เพื่อเป็นแนวทางในการวิเคราะห์และออกแบบระบบให้มีประสิทธิภาพ โดยเนื้อหาที่ผู้วิจัยศึกษามีดังนี้

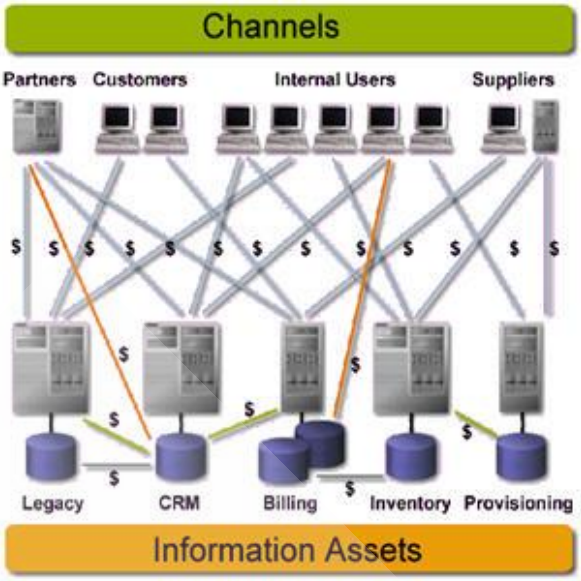
1. สถาปัตยกรรมซอฟต์แวร์เชิงบริการ SOA (Service-Oriented Architecture)
2. เว็บเซอร์วิส (Web service)
3. OpenAPI Specification (OAS)
4. WebSocket
5. Global Positioning System (GPS)
6. JavaScript Object Notation (JSON)
7. งานวิจัยที่เกี่ยวข้อง

2.1 สถาปัตยกรรมซอฟต์แวร์เชิงบริการ SOA (Service-Oriented Architecture)

SOA (Service-Oriented Architecture) เป็นหลักการการออกแบบสถาปัตยกรรมซอฟต์ แวร์ที่ได้รับการกล่าวถึงอย่างมาก หลายองค์กรพยายามที่จะออกแบบระบบทางด้านไอทีให้เข้าสู่ระ บบ SOA ซึ่ง SOA คือการออกแบบที่มุ่งเน้นให้แอปพลิเคชันสามารถทำงานร่วมกันได้ โดยไม่ขึ้น กับแพลตฟอร์มภาษาคอมพิว เตอร์ และเทคโนโลยีที่ใช้ในการพัฒนา

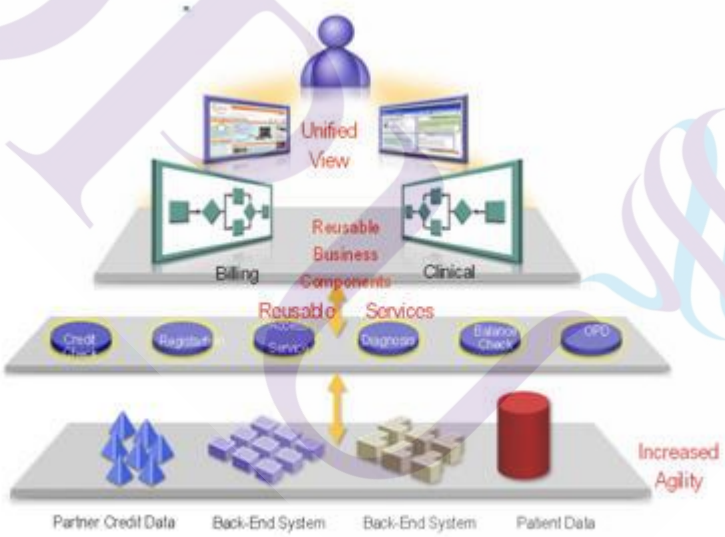
2.1.1 ความหมายของ SOA

ระบบสถาปัตยกรรมเชิงบริการหรือ SOA เป็นแนวคิดในการออกแบบระบบไอทีในองค์กรให้เป็นระบบเชิงบริการ (Service-Oriented) ที่สามารถนำกลับมาใช้ใหม่ได้ ทั้งนี้ระบบไอทีขององค์กรต่าง ๆ ในปัจจุบันมักจะมีสถาปัตยกรรมแบบ Silo-Oriented Architecture ซึ่งการพัฒนาระบบไอทีในแต่ละระบบต่างเป็นอิสระต่อกัน อาจมีระบบที่ใช้เทคโนโลยีที่แตกต่างกันเช่น Java, .NET, Oracle หรือ SAP เป็นต้น จึงทำให้ยากต่อการเชื่อมต่อ บำรุงรักษายาก มีค่าใช้จ่ายสูง ปรับเปลี่ยนระบบได้ยาก และการพัฒนาระบบใหม่ๆ เป็นไปด้วยความล่าช้า ดังแสดงในรูปที่ 2.1



ภาพที่ 2.1 แสดงถึง Silo-Oriented Architecture

ที่มา: ฉัตรสุชล คุ้มถนอม, 2554



ภาพที่ 2.2 แสดงถึง SOA Conceptual Diagram

ที่มา: ฉัตรสุชล คุ้มถนอม, 2554

แนวคิดของระบบ SOA คือการจัดระบบ Silo-Oriented Architecture ใหม่ โดยการสร้างระบบไอทีให้เป็น 4 ชั้น (Layer) ดังแสดงในรูปที่ 2.3 และ 2.4

1. Resource Layer ซึ่งจะเป็นชั้นของระบบโครงสร้างไอทีต่าง ๆ ในปัจจุบัน เช่นระ บบฐานข้อมูล Oracle ระบบโซลูชัน SAP หรือ PeopleSoft เป็นต้น

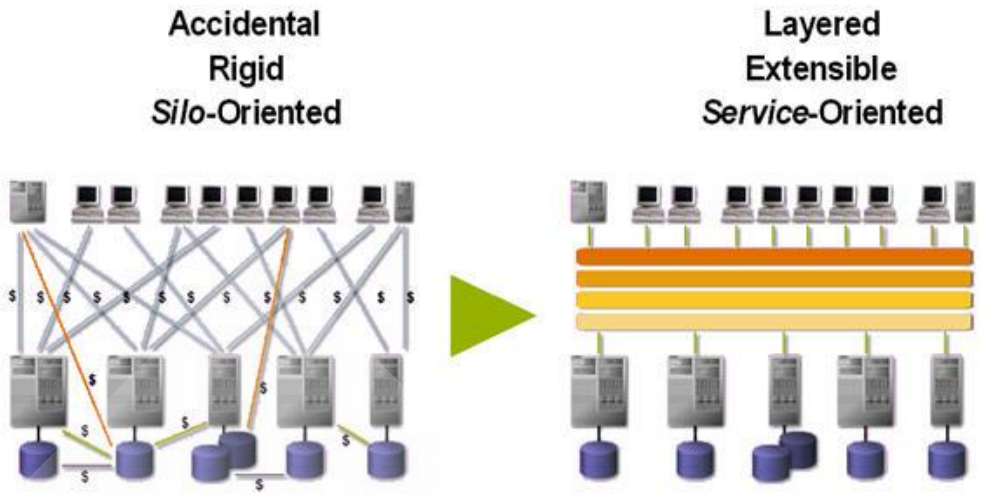
2. Service Layer ซึ่งเป็นชั้นของส่วนประกอบเซอร์วิสต่าง ๆ ที่สามารถนำมาใช้ใหม่ได้ โดยส่วนประกอบเซอร์วิสเหล่านี้จะพัฒนามาจากโมดูล (Module) ต่าง ๆ ที่ทำงานบน Resource Layer เช่น โมดูลของฐานข้อมูล Oracle โมดูลของระบบโซลูชัน SAP หรือ PeopleSoft และโมดูลของโปรแกรมประยุกต์ที่อาจพัฒนาด้วย Java หรือ .NET เป็นต้น

3. Process Layer ซึ่งเป็นชั้นของกระบวนการทางธุรกิจ (Business Process) ที่พัฒนา ขึ้นมาจากการส่วนประกอบเซอร์วิสต่าง ๆ

4. Access Layer ซึ่งเป็นชั้นของการเรียกใช้กระบวนการทางธุรกิจที่พัฒนาขึ้น โดยอาจผ่านทางเว็บไซต์ (Web Site) หรือโทรศัพท์เคลื่อนที่ (Mobile Phone)

ภาพที่ 2.3 แสดง SOA Layers
ที่มา: ฉัตรสุชล คุ้มถนอม, 2554

ดังนั้นจะเห็นได้ว่า SOA เป็นการเปลี่ยนระบบ Silo-Oriented Architecture มาสู่ระบบ Service-Oriented ซึ่งออกแบบเป็นชั้นๆ ทำให้สามารถพัฒนาปรับปรุง หรือเพิ่มเติมโปรแกรมใหม่ได้ง่าย ดังแสดงในรูปที่ 2.4



ภาพที่ 2.4 การปรับเปลี่ยนสถาปัตยกรรมระบบจาก Silo-Oriented มาเป็นสถาปัตยกรรมแบบ SOA
ที่มา: ฉัตรสุชล คุ้มถนอม, 2554

เนื่องจาก SOA เป็นหลักการในการออกแบบ จึงทำให้การทำความเข้าใจและนำไปพัฒ นาให้ใช้งานได้จริงจึงเป็นเรื่องที่ยาก จนเมื่อเว็บเซอร์วิส (Web Service) ซึ่งเป็นวิธีการหนึ่งในการพัฒนาตามหลักการของ SOA เกิดขึ้นมา จึงทำให้แนวคิด SOA ได้รับความนิยมขึ้นมาอย่างมาก จนบางครั้งอาจทำให้คิดว่า SOA และ เว็บเซอร์วิสเป็นเรื่องเดียวกัน ซึ่งในความจริงนั้น SOA เป็นแนวคิดหรือรูปแบบในการออก แบบการให้บริการ ส่วนเว็บเซอร์วิสเป็นวิธีการหนึ่งในการพัฒนาตามหลักการของ SOA เท่านั้น ทั้งนี้อาจใช้แนวทางอื่นในการพัฒนาระบบ SOA เช่นการใช้ CORBA (Common Object Request Broker Architecture) หรือ Java RMI (Remote Method Invocation) ก็ได้เช่นกัน

ระบบ SOA จะมีคุณลักษณะที่สำคัญหลักๆ ดังนี้

1. การติดต่อสื่อสารระหว่างเซอร์วิส จะใช้เอกสารที่เป็น XML ที่นิยามผ่าน XML Schema (. xsd) ทำให้ไม่จำเป็นต้องทราบรายละเอียดของแฟลตฟอร์มและเทคโนโลยีของเซอร์วิสที่ ใช้อยู่

2. เซอร์วิสจะมีตัวเชื่อมต่อ (Interface) ที่อธิบายเซอร์วิส เช่น Service Name, Input Parameter, Output Parameter และข้อมูลอื่นๆ ในรูปแบบของไฟล์ XML ทำให้ไม่ขึ้นกับแฟลตฟอร์มและเทคโนโลยีที่เซอร์วิสนั้นใช้อยู่ โดยมากมักจะใช้มาตรฐาน WSDL (Web Service Description Language) ในการอธิบายเซอร์วิส

3. โปรแกรมประยุกต์ (Application) หรือกระบวนการทางธุรกิจต่าง ๆ สามารถพัฒนา ขึ้นมาจากการใช้เซอร์วิสเดิมที่มีอยู่ ซึ่งมาตรฐานที่นิยมใช้คือ WS-BPEL (Web Service Business Process Execution Language)

4. SOA จะมี Registry ในการเก็บเซอร์วิสต่าง ๆ ที่มีอยู่ ซึ่ง Registry จะทำหน้าที่เหมือนไดเร็กทอรี่ของเซอร์วิส โดยโปรแกรมประยุกต์หรือกระบวนการทางธุรกิจต่าง ๆ จะค้นหาและเรียกใช้

เซอร์วิสจาก Registry นี้ มาตรฐานที่ใช้ในการเก็บ Registry ที่นิยมใช้คือ UDDI (Universal Description Definition and Integration)

5. เซอร์วิสแต่ละตัวจะมีส่วนการควบคุมคุณภาพที่เป็น QoS (Quality of Service) อาทิเช่น การควบคุมความปลอดภัยด้าน Authentication, Authorization, Reliable Message และ Policy

2.1.2 พัฒนาการของ ระบบ Distributed Computing

ระบบสถาปัตยกรรมเชิงบริการหรือ SOA นั้นมีพัฒนาการมาจากระบบ Distributed Computing ซึ่งเริ่มตั้งแต่ยุคแรกที่เป็น Single Tier ไปสู่ Web Tier ดังแสดงในรูปที่ 2.5 ในปัจจุบันหลายๆ องค์กรได้พัฒนา Web Applications ซึ่งส่วนมากจะพัฒนาโดยใช้ Java EE (Servlet/JSP), .NET (ASP) หรือ PHP ทั้งส่วนแสดงผล (Presentation Logic) และส่วนประมวลผล (Business Logic) ภายใน Web Server และผู้พัฒนาจะต้องพัฒนาส่วนที่เป็นเซอร์วิสระบบ (System Service) เช่น Concurrency, Load Balancing, Transaction และ Security เอง ทำให้การพัฒนา Web Application แบบ Web Tier สำหรับระบบขนาดใหญ่ทำได้ยาก

ภาพที่ 2.5 แสดงถึง Web Application Architecture
ที่มา: ศิรประภา ธนุการ, 2553

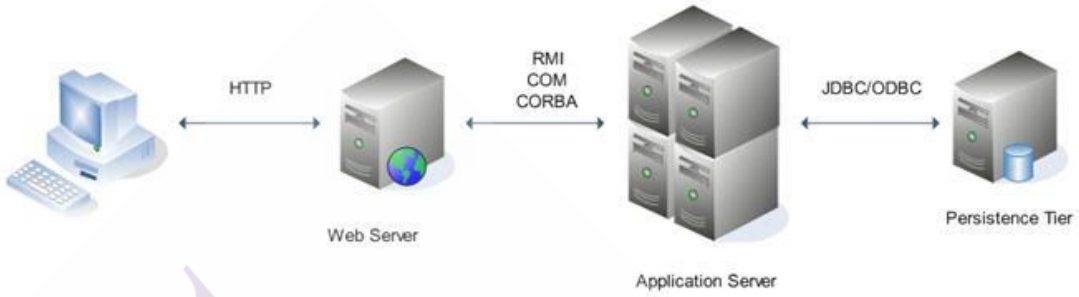
แนวทางการพัฒนาระบบ Distributed Computing ในยุคถัดมา คือการพัฒนาระบบแบบ N-Tier ดังแสดงในรูปที่ 2.6 ซึ่งจะมีการนำเอา Application Server มาเป็นมิดเดิ้ลแวร์ (Middleware) เพื่อจัดการส่วนที่เป็นเซอร์วิสของระบบและเรื่องที่เกี่ยวข้องกับทรัพยากรต่างๆ ของระบบ ทำให้นัก พัฒนาสามารถที่จะเน้นการพัฒนาเฉพาะส่วนประมวลผล โดยทำการสร้างส่วนประกอบซอฟต์แวร์ (Software Component) หรือเซอร์วิส (Service) เพื่อให้ส่วนแสดงผลใน Web Server เรียกใช้งานได้

ซึ่งวิธีการพัฒนาส่วนประกอบ ซอฟต์แวร์หรือเซอร์วิส อาจใช้ Java EE (EJB), .NET (NET Managed Component) หรือระบบ Legacy (IDL/CORBA) โดยใช้โพรโทคอลเฉพาะที่เป็น Binary Protocol ดังนี้

RMI/IIOP สำหรับ Java EE

COM สำหรับ Microsoft

CORBA สำหรับระบบมาตรฐานทั่วไป



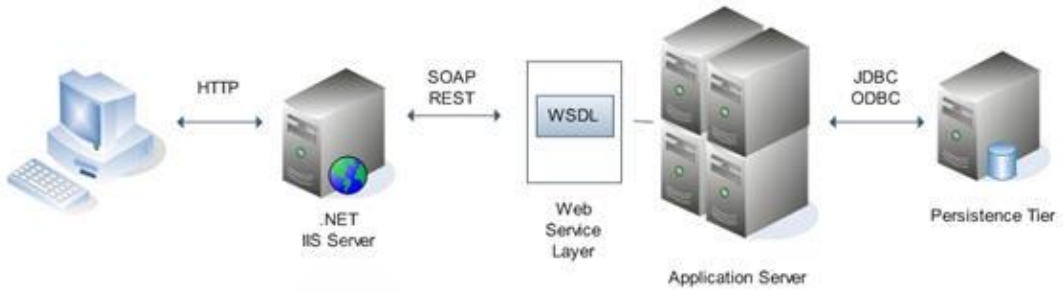
ภาพที่ 2.6 แสดงถึง N-Tier Architecture

ที่มา: ศิรประภา ธนุการ, 2553

ระบบ Distributed Computing ในยุคถัดมาก็คือยุคที่มีการนำเว็บเซอร์วิสเข้ามาใช้ โดยการเปลี่ยนโพรโทคอลเฉพาะเหล่านั้นให้เป็นโพรโทคอลกลางที่มีมาตรฐาน เช่นการนำโพรโทคอล SOAP/REST มาใช้ในการเรียกเซอร์วิสแทนที่การใช้โพรโทคอล RMI/IIOP หรือการนำ WSDL มาใช้ในการประกาศเซอร์วิส ดังตัวอย่างในรูปที่ 2.7 ซึ่งเป็นการแสดงการเปลี่ยนส่วนประกอบซอฟต์ แวร์ที่ใช้ Java EE ให้เป็นส่วนประ กอบ เซอร์วิส (Service Component) โดยการเพิ่ม Web Service Layer

ภาพที่ 2.7 แสดงถึง N -Tier Architecture โดยใช้ Web Service

ที่มา: ศิรประภา ธนุการ, 2553



ภาพที่ 2.8 แสดงถึงการทำงานร่วมกันระหว่าง .NET and Java EE
ที่มา: ศิรประภา ธนุการ, 2553

2.1.3 เหตุผลของการพัฒนา SOA

การพัฒนาสถาปัตยกรรม SOA จะมีประโยชน์ต่อองค์กรในหลายๆ ด้านอาทิ การทำให้ข้อมูลต่างๆภายในองค์กรเชื่อมโยงกัน การลดค่าใช้จ่ายในการบำรุงรักษา การทำให้การพัฒนาโปรแกรมใหม่เป็นไปด้วยความรวดเร็วขึ้น และทำให้ระบบไอทีในองค์กรไม่ผูกติดอยู่กับระบบใดระบบหนึ่ง

โครงสร้างของระบบไอทีขององค์กรขนาดใหญ่ (Information Technology Enterprise) จะประกอบไปด้วยระบบที่หลากหลายทั้งในด้านระบบปฏิบัติการ (Operating System) โปรแกรมประยุกต์และระบบซอฟต์แวร์ ซึ่งโปรแกรมประยุกต์บางโปรแกรม อาจใช้ในการทำงานกับกระบวนการทางธุรกิจบางอย่าง ที่อาจทำงานภายใต้ระบบโครงสร้างไอทีเดิม เช่นพัฒนาโดยใช้เครื่องเมนเฟรม ดังนั้นเมื่อมีความจำเป็นต้องเปลี่ยนแปลงกระบวนการทางธุรกิจ จะทำให้การเปลี่ยนแปลงโดยใช้โครงสร้างไอทีเดิมทำได้ยาก จนอาจมีความต้องการที่จะยกเลิกระบบเดิมและพึ่งพาเทคโนโลยีใหม่ ระบบ SOA จะช่วยคุ้มครองการลงทุนขององค์กร เพื่อให้สามารถนำระบบโครงสร้างไอทีเดิมมาใช้ต่อไปได้ โดยการพัฒนาระบบโปรแกรมเดิมให้เป็น SOA Service และสามารถพัฒนากระบวนการทางธุรกิจจากเซอร์วิสต่างๆที่มีอยู่ จึงทำให้องค์กรสามารถเปลี่ยนกระบวนการทางธุรกิจได้อย่างรวดเร็ว โดยใช้โปรแกรมประยุกต์เดิม และโครงสร้างไอทีเดิมที่มีอยู่

เหตุผลหลักขององค์กรในการพัฒนาระบบ SOA จึงมักจะเริ่มจากความต้องการในการเชื่อมโยงระบบโครงสร้างไอทีต่างๆ ในปัจจุบันเข้าด้วยกัน หรือการทำ Enterprise Application Integration (EAI) แต่ระบบ SOA จะแตกต่างกับระบบ EAI เดิมในแง่ที่ของความสามารถในการพัฒนากระบวนการทางธุรกิจใหม่จากเซอร์วิสเดิมที่มีอยู่ และมีการใช้ถึงมาตรฐานต่างๆ จากนั้นก็จะเป็นการนำ SOA มาใช้เพื่อพัฒนากระบวนการทางธุรกิจใหม่ๆ

2.1.4 ประโยชน์ของการพัฒนา SOA

การพัฒนาระบบโครงสร้างไอทีในองค์กรให้เป็นระบบ SOA จะเกิดประโยชน์ในด้านต่างๆ ดังนี้

1. สามารถเชื่อมโยงธุรกิจต่างๆ

การพัฒนา SOA มีการเชื่อมโยงระบบไอทีต่างๆ ภายในองค์กรและภายนอกองค์กรที่อาจใช้เทคโนโลยีที่ต่างกัน ทำให้เราสามารถเชื่อมโยงธุรกิจต่างๆ ที่อาจอยู่ต่างระบบกัน และสามารถให้บริการกับลูกค้า คู่ค้า และบุคลากรในองค์กรได้

2. ระบบไอทีสามารถปรับเปลี่ยนได้ง่าย

การพัฒนา SOA สามารถที่จะทำให้นำระบบไอทีเดิมมาใช้ใหม่ได้ ดังนั้นการปรับเปลี่ยนกระบวน การทางธุรกิจจึงเป็นไปได้อย่างรวดเร็ว และทำให้สามารถแข่งขันในตลาดธุรกิจได้อย่างรวดเร็ว

3. การลดค่าใช้จ่ายในการบำรุงรักษา และให้ผลตอบแทนการลงทุนที่คุ้มค่า

การพัฒนา SOA ทำให้องค์กรสามารถที่จะใช้เทคโนโลยีที่หลากหลาย จึงทำให้เราสามารถที่จะเลือกใช้เทคโนโลยีต่างๆ ได้ โดยไม่ต้องผูกติดกับเทคโนโลยีใดเทคโนโลยีหนึ่ง ทำให้ค่าใช้จ่ายด้านไอทีในระยะยาวลดลง

4. การทำงานของฝ่ายธุรกิจและฝ่ายไอทีสอดคล้องกันมากขึ้น

การพัฒนา Business Process ของฝ่ายไอทีจะมีขั้นตอนที่ชัดเจนสามารถแสดงในเชิงกราฟฟิกได้และเข้าใจง่าย ขึ้น และหน่วยงานทางธุรกิจที่ต้องเข้าใจด้านกระบวนการทางธุรกิจสามารถที่จะเข้ามา ร่วมทำการพัฒนาร่วมกับฝ่ายไอทีได้ดีขึ้น

2.1.5 การพัฒนา SOA โดยใช้เว็บเซอร์วิส

แม้การพัฒนาสถาปัตยกรรมเชิงบริการ (Service Oriented Architecture หรือ SOA) ในยุคแรกจะสามารถทำได้โดยใช้เทคโนโลยีอื่นๆ อาทิเช่น CORBA, Java RMI และ DCOM หรือสามารถใช้ MOM (Message Oriented Middleware) เพื่อพัฒนา SOA ในรูปของการแลกเปลี่ยนข้อมูลของสถาปัตยกรรม EAI ที่นิยมใช้ทั่วไป แต่ในปัจจุบันการพัฒนา SOA โดยใช้เทคโนโลยีเว็บเซอร์วิสเริ่มได้รับการยอมรับมากกว่าเทคโนโลยีอื่นๆ ด้วยเหตุผลดังนี้

1. เว็บเซอร์วิสอิงอยู่กับมาตรฐานที่เปิดเช่น SOAP WSDL UDDI และมาตรฐานเว็บเซอร์วิสอื่นๆ ทำให้องค์กรต่างไม่จำเป็นที่จะต้องลงทุนกับโซลูชันที่ใช้เทคโนโลยีเฉพาะ และป้องกันการผูกขาดโดยผู้ผลิตรายใดรายหนึ่ง

2. เว็บเซอร์วิสสนับสนุนการเชื่อมโยงกับโซลูชันของผู้ผลิตต่างๆ ทำให้ช่วยลดต้นทุน

3. เว็บเซอร์วิสสนับสนุนการเชื่อมโยงระบบ แพลตฟอร์ม และโซลูชันต่างๆทั้งภายในองค์กร ระหว่างองค์กร และภายนอก

2.2 เว็บเซอร์วิส (Web service)

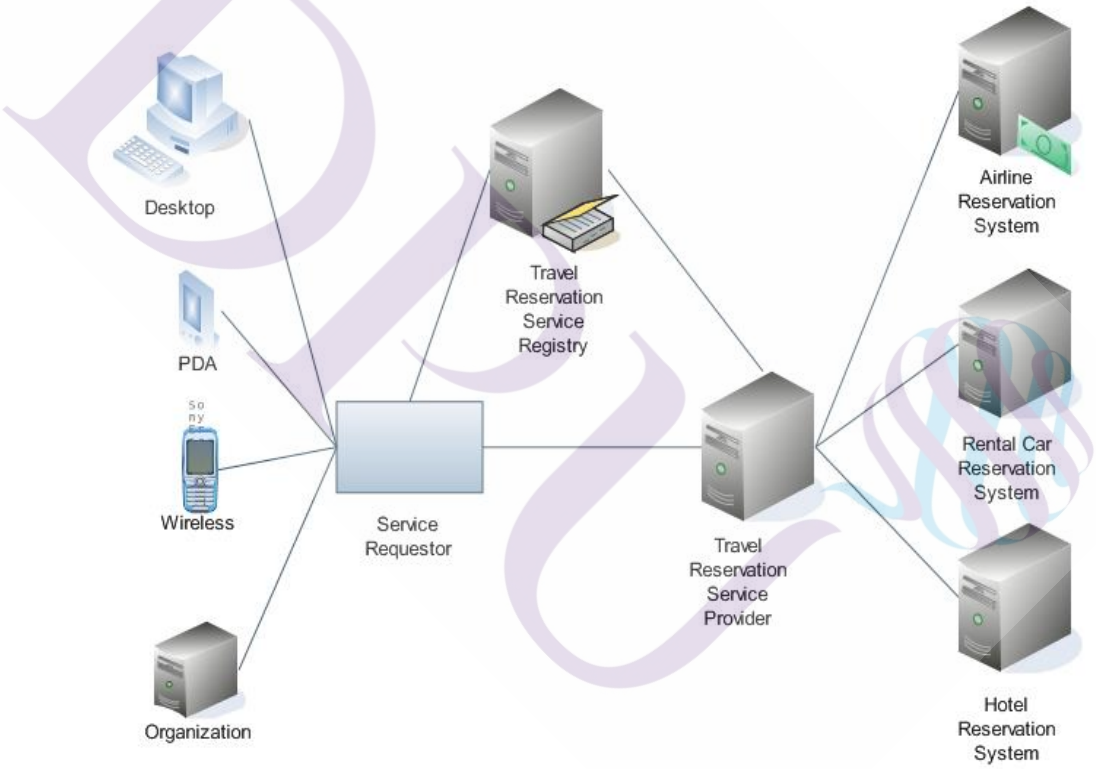
เว็บเซอร์วิส (Web service) คือระบบซอฟต์แวร์ที่ออกแบบมา เพื่อสนับสนุนการแลกเปลี่ยนข้อมูลกัน ระหว่างเครื่องคอมพิวเตอร์ผ่านระบบเครือข่าย โดยที่ภาษาที่ใช้ในการติดต่อสื่อสารระหว่างเครื่องคอมพิวเตอร์ คือเอกซ์เอ็มแอล เว็บเซอร์วิสมีอินเทอร์เฟส ที่ใช้อธิบายรูปแบบข้อมูลที่เครื่องคอมพิวเตอร์ประมวลผลได้ เช่น WSDL ระบบคอมพิวเตอร์ใช้งานสื่อสารโต้ตอบกับเว็บเซอร์วิสตามรูปแบบที่ได้กำหนดไว้แล้ว โดยการส่งสาสน์ตามอินเตอร์เฟสของเว็บเซอร์วิสนั้น โดยที่สาสน์ดังกล่าวอาจแนบไว้ในซอง SOAP หรือส่งตามอินเตอร์เฟสในแนวทางของ REST สาสน์เหล่านี้ปกติแล้วถูกส่งโดยอาศัย HTTP และใช้ XML ร่วมกับมาตรฐานเกี่ยวกับเว็บอื่นๆ โปรแกรมประยุกต์ที่เขียนโดยภาษาต่างๆ และทำงานบนแพลตฟอร์มต่างๆกันสามารถใช้เว็บเซอร์วิสเพื่อแลกเปลี่ยนข้อมูลผ่านทางเครือข่ายคอมพิวเตอร์ เช่น อินเทอร์เน็ต ในลักษณะเดียวกับการสื่อสารระหว่างโปรเซส (Inter-process communication) บนเครื่องเดียวกัน ความสามารถในการแลกเปลี่ยนข้อมูลระหว่างระบบที่ต่างกันนี้ (เช่น การแลกเปลี่ยนข้อมูลระหว่าง โปรแกรมที่เขียนโดยภาษาจาวา และโปรแกรมที่เขียนโดยภาษาไพทอน หรือการแลกเปลี่ยนข้อมูลระหว่างโปรแกรมประยุกต์ที่ทำงานบนไมโครซอฟท์วินโดวส์ และโปรแกรมประยุกต์ที่ทำงานบนลินุกซ์) เกิดขึ้นได้เนื่องจากการใช้มาตรฐานเปิด โดย OASIS และ W3C เป็นคณะกรรมการหลักในการรับผิดชอบมาตรฐานและสถาปัตยกรรมของเว็บเซอร์วิส

ความหมายของเซอร์วิส (Service) หรือบริการในแง่ของเทคโนโลยี เราลองมาพิจารณาความหมายของบริการหรือกระบวนการ (Process) ที่องค์กรต่างๆ ทั้งภาครัฐและเอกชนต่างให้บริการกับประชาชน ลูกค้า พนักงาน หรือคู่ค้า ตัวอย่างเช่นบริการที่พนักงานหน้าเคาท์เตอร์ของธนาคารสามารถให้บริการแก่ลูกค้าอาจมีหลากหลายอาทิเช่น บริการฝาก/ถอนเงิน บริการแลกเปลี่ยนเงินตราต่างประเทศ หรือบริการด้านสินเชื่อเป็นต้น บริการแต่ละบริการอาจจะมีกระบวนการในการทำงานที่ซับซ้อนแตกต่างกัน แต่ในมุมมองของลูกค้าจะไม่สนใจว่าบริการนั้นมีขั้นตอนการทำงานอย่างไร แต่จะมีวัตถุประสงค์หลักเพื่อให้บริการนั้นเสร็จสิ้นและได้ผลลัพธ์ออกมาตามที่ลูกค้าขอใช้บริการนั้นๆ

ความหมายของเซอร์วิสในแง่ของซอฟต์แวร์ ก็มีคุณลักษณะเช่นเดียวกับบริการทั่วๆ ไป กล่าวคือเป็นซอฟต์แวร์คอมโพเนนท์ (Software Component) ที่อาจเป็น ฟังก์ชัน หรือ โมดูล ที่มีกระบวนการการทำงานภายใน สามารถรับอินพุตเข้ามาเพื่อประมวลผล และจะส่งผลลัพธ์กลับออกไป ซอฟต์แวร์เซอร์วิสเหล่านี้เราอาจกำหนดเป็นกระบวนการทางธุรกิจ (Business Process) กล่าวคือจะเป็นฟังก์ชันที่ทำเฉพาะการประมวลผลซึ่งจะไม่เกี่ยวข้องกับส่วนแสดงผล (Presentation Logic) นอกจากนี้ด้วยเทคโนโลยี Distributed Computing ทำให้สามารถที่จะพัฒนาซอฟต์แวร์เซอร์วิสเพื่อเรียกใช้จากระยะไกล (remote) ผ่าน Internet ได้โดยใช้เทคโนโลยีเฉพาะด้าน (proprietary technology) อาทิเช่น RMI, CORBA หรือ DCOM

ตัวอย่างการใช้งานของซอฟต์แวร์เซอร์วิสโดยใช้ Distributed Computing จะเป็นไปดัง

แสดงในรูปที่ 2.9 ซึ่งจะเห็นได้ว่ามีระบบ Back-end ต่างๆ เช่น Airline Reservation System และ Hotel Reservation System ที่มีซอฟต์แวร์เซอร์วิสต่างๆ อยู่ ผู้ใช้ด้าน Front-end ที่จะเป็นผู้ใช้บริการเซอร์วิส (Service Requestor) ซึ่งอาจเรียกใช้จากอุปกรณ์ต่างๆ เช่น คอมพิวเตอร์ หรือโทรศัพท์เคลื่อนที่ (Mobile Phone) จะสามารถเรียกใช้ซอฟต์แวร์เซอร์วิสเหล่านี้ผ่านผู้ให้บริการเซอร์วิส (Services Provider) ซึ่งทำหน้าที่เป็น Middleware การเรียกใช้เซอร์วิสเหล่านี้อาจเป็นการเรียกใช้จากผู้ใช้โดยตรง หรือเรียกใช้โดยโปรแกรมซอฟต์แวร์คอมโพเนนท์ (Program to Program) จากอุปกรณ์ที่ใช้ นอกจากนี้ในกรณีที่ไม่ทราบชื่อหรือเซอร์วิสที่มีอยู่ เราสามารถที่จะค้นหาซอฟต์แวร์เซอร์วิสเหล่านี้ได้จาก Registry ที่ทำหน้าที่เก็บรายละเอียดของซอฟต์แวร์เซอร์วิสต่างๆที่มีอยู่ โดยผู้ให้บริการเซอร์วิสจะทำหน้าที่ลงทะเบียนรายละเอียดของเซอร์วิสไว้ ทั้งนี้การค้นหาเซอร์วิสผ่าน Registry สามารถทำได้อัตโนมัติโดยใช้คำสั่งในโปรแกรมด้านผู้ใช้บริการเซอร์วิส



ภาพที่ 2.9 แสดงถึงซอฟต์แวร์เซอร์วิสโดยใช้ Distributed Computing
ที่มา: ศิรประภา ธนุการ, 2553

เว็บเซอร์วิสจะใช้หลักการของซอฟต์แวร์เซอร์วิสของ Distributed Computing แต่จะใช้โปรโทคอลที่มีมาตรฐานกลาง (Standard Protocol) ที่อยู่ในรูปแบบ XML (eXtensible Markup Language) และจะเป็นซอฟต์แวร์คอมโพเนนท์ที่ให้บริการผ่านอินเตอร์เน็ต

Gartner Research ได้ให้คำนิยามของเว็บเซอร์วิสไว้ดังนี้ “เว็บเซอร์วิสคือ ซอฟต์แวร์คอมโพเนนท์แบบ loosely coupled ที่ส่งบริการผ่านเทคโนโลยีอินเตอร์เน็ตที่มีมาตรฐาน”

2.2.1 พื้นฐานของเว็บเซอร์วิส

พื้นฐานของ Web Service คือ XML และส่วนใหญ่จะใช้ HTTP แต่อาจจจะใช้อินเทอร์เน็ตโพรโทคอลอื่นอย่างเช่น SMTP หรือ FTP ก็ได้ แต่จะพบว่า HTTP ก็เป็นที่รู้จักกันดี และไปได้ทั่วทุกแห่งที่มี internet ส่วน XML คือภาษาสากลที่คุณสามารถปรับแต่งได้ตามใจชอบ เพื่อให้เกิดกิจกรรมระว่าง client และบริการ หรือระหว่างส่วนประกอบต่างๆ เบื้องหลัง Web server ก็คือ ข้อความ XML จะถูกแปลงให้การขอบริการจาก Middle ware และผลที่ได้ก็จะแปลงกลับมาในรูป XML

ยกตัวอย่างให้เห็นง่ายๆ คุณต้องการให้เครื่อง PC อ่านค่าจาก serial port แล้วส่งไปประมวลผลบนเครื่อง UNIX แล้วส่งผลกลับมาแสดงบนจอ PC ถ้าเป็นเมื่อก่อน คุณก็คงต้องแปลงข้อมูลที่ได้ให้อยู่ในรูปของ ASCII แล้วส่งไปยัง UNIX พร้อมคำสั่งว่าให้ทำอะไร ในฝั่ง UNIX คุณก็ต้องมาแยกว่าอันไหนคือคำสั่ง อันไหนคือข้อมูล เมื่อประมวลผลแล้ว จะส่งกลับมาในรูปแบบไหน แล้วถ้าหากจะส่งไปหาเครื่องที่เป็น MAC ท่านจะต้องเขียนโปรแกรมเพิ่มในส่วนไหนบ้าง จะพบว่าเราต้องพัฒนากันเป็นคู่ๆ ไป และต้องนิยามในแต่ละฝั่งให้ชัดเจน แต่หากเป็น Web Service คุณจะพบว่า เราแปลงข้อมูลให้อยู่ในรูป XML แต่ละคุณก็ต้องการรู้แค่ มาตรฐาน XML ก็พอ แล้วต่างคนต่างก็เขียน Service ของตัวเอง ไม่ต้องกังวลเรื่องของการเชื่อมโยงอีกต่อไป และ Protocol ที่ส่งก็คือ HTTP นั่นเอง ถ้าท่านเชื่อมโยงกับ HTTP (หรือเว็บ) ได้ ท่านก็ใช้บริการทุกอย่างได้

คุณลักษณะพื้นฐานของเว็บเซอร์วิสมีดังนี้

1. เว็บเซอร์วิสเป็นซอฟต์แวร์คอมโพเนนท์ที่ระบุตำแหน่งโดยใช้ URI
2. อินเตอร์เฟสและการติดตั้งของเซอร์วิสจะนิยาม อธิบาย และค้นหาโดยใช้ ภาษาXML
3. เว็บเซอร์วิสสนับสนุนการเรียกใช้จากซอฟต์แวร์ประยุกต์อื่นๆ ผ่านโพรโทคอลอินเตอร์เน็ต
4. เว็บเซอร์วิสใช้เอกสารแบบ XML ในการส่งข้อมูลระหว่างผู้ให้บริการและผู้ใช้
5. เว็บเซอร์วิสช่วยในการเชื่อมโยงโปรแกรมประยุกต์ต่างแพลตฟอร์ม (Cross-platform Integration) ผ่านอินเตอร์เน็ต
6. นักพัฒนาสามารถพัฒนาเว็บเซอร์วิสได้โดยใช้โปรแกรมภาษาคอมพิวเตอร์ต่างๆเช่น Java, C, C# หรือ Visual Basic และสามารถพัฒนาโดยการเปลงซอฟต์แวร์คอมโพเนนท์ที่มีอยู่ให้เป็นเว็บเซอร์วิส
7. เว็บเซอร์วิสจะไม่รวมถึงการจัดการส่วนแสดงผลเหมือน HTML
8. เว็บเซอร์วิสจะเป็นซอฟต์แวร์คอมโพเนนท์แบบ loosely couple ดังนั้นแต่ละคอมโพเนนท์จะเป็นอิสระและมีฟังก์ชันที่สมบูรณ์ในตัว

9. เราสามารถที่จะค้นหาและเรียกใช้เว็บเซอร์วิสจาก registry ที่เป็นแบบ public หรือ private โดยใช้มาตรฐานกลางเช่น UDDI และ ebXML

10. เว็บเซอร์วิสสามารถที่จะเรียกใช้โดย client ต่างๆ ได้เช่น คอมพิวเตอร์ โทรศัพท์เคลื่อนที่ หรือ พีดีเอ

2.2.2 เหตุผลของการพัฒนาเว็บเซอร์วิส

เว็บเซอร์วิสจะแตกต่างกับโปรแกรมประยุกต์บนเว็บ (Web Application) และDistributed Computing (Distributed Application) ทั้งนี้เพราะโปรแกรมประยุกต์บนเว็บ จะเป็นโปรแกรมเพื่อให้ผู้ใช้ (End User) สามารถโต้ตอบกับโปรแกรมผ่านเว็บไซต์ได้ ไม่ใช่ Distributed Computing ที่เป็นซอฟต์แวร์เซอร์วิสและไม่สามารถเรียกใช้จากผู้ใช้ที่หลากหลายได้ ส่วนข้อจำกัดของ Distributed Computing คือจะยึดติดกับโพรโทคอลเฉพาะเช่น RMI หรือ CORBA และโพรโทคอลเหล่านี้เป็นแบบไบนารี่ (Binary Protocol) จึงผูกอยู่กับเทคโนโลยีใดเทคโนโลยีหนึ่ง ดังนั้นจะเห็นได้ว่าเว็บเซอร์วิสไม่ใช่แนวคิดใหม่ เพียงแต่เปลี่ยนรูปแบบโพรโทคอลและหลักการบางอย่างที่เคยใช้ใน Distributed Computing เหตุผลสำคัญที่ควรเลือกพัฒนาเว็บเซอร์วิสมากกว่าการพัฒนาโปรแกรมประยุกต์บนเว็บและ Distributed Computing คือ

1. เว็บเซอร์วิสใช้โพรโทคอลที่เป็นมาตรฐานโดยใช้รูปแบบ XML

2. เราสามารถเรียกใช้เว็บเซอร์วิสโดย XML-based RPC จึงทำให้สามารถเรียกผ่าน Firewall ซึ่งแตกต่างกับกรณีของเทคโนโลยีแบบกระจาย

3. เว็บเซอร์วิสสนับสนุนการทำงานร่วมกันของโซลูชัน ที่ข้ามแพลตฟอร์มและใช้ภาษาคอมพิวเตอร์ที่ต่างกันได้ โดยการส่งข้อมูลแบบ XML

4. เว็บเซอร์วิสสนับสนุนการการเรียกใช้จากซอฟต์แวร์ประยุกต์อื่นๆ ผ่านโพรโทคอลอินเตอร์เน็ต ซึ่งแตกต่างกับโปรแกรมประยุกต์บนเว็บที่เป็นการเรียกโดยตรงจากผู้ใช้

จุดเด่นของการพัฒนาเว็บเซอร์วิสสามารถที่จะสรุปได้ดังนี้

1. การเชื่อมโยง (Interoperable) สนับสนุนการเชื่อมโยงกันระหว่างโปรแกรมประยุกต์ที่หลากหลาย (Heterogeneous Applications) ได้ โดยใช้มาตรฐานเว็บที่เป็นกลาง

2. ลดค่าใช้จ่าย (Economical) สนับสนุนการนำซอฟต์แวร์คอมโพเน้นท์กลับมาใช้ใหม่ (reuse) และไม่ต้องยึดติดกับเทคโนโลยีเดิม

3. อัตโนมัติ (Automatic) สนับสนุนการการเรียกใช้จากโปรแกรมโดยตรง โดยไม่ต้องโต้ตอบกับผู้ใช้

4. เข้าถึงได้ (Accessible) สามารถที่จะเรียกใช้โปรแกรมเดิม (Legacy) หรือโปรแกรมภายในผ่านเว็บได้

5. ใช้ได้ตลอด (Available) สนับสนุนการเรียกใช้ได้ทุกที่ ทุกอุปกรณ์ และทุกเวลา

6. ขยายได้ (Scalable) ไม่ได้จำกัดขนาดของโปรแกรมหรือจำนวนของระบบต่างๆ

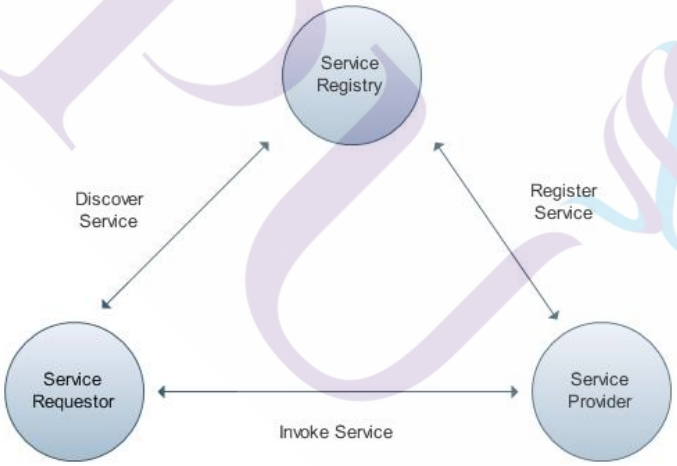
2.2.3 โมเดลการทำงานของเว็บเซอร์วิส

กระบวนการการทำงานของเว็บเซอร์วิสจะมีขั้นตอนการทำงานเช่นเดียวกับซอฟต์แวร์เซอร์วิสที่ใช้ Distributed Computing ดังอธิบายในรูปที่ 2.9 ซึ่งเราสามารถที่จะแบ่งบทบาทองค์ประกอบของเว็บเซอร์วิสได้เป็นสามส่วน โดยทั้งสามองค์ประกอบมีความสัมพันธ์ดังแสดงในรูปที่ 2.10 และสามารถอธิบายได้ดังนี้

1. ผู้ให้บริการ (Service Provider) ผู้ให้บริการจะมีหน้าที่ในการพัฒนาและติดตั้งเว็บเซอร์วิส และเป็นผู้ที่นิยามความหมายของเซอร์วิสและลงทะเบียนเซอร์วิสกับ Service Registry

2. ผู้ใช้บริการ (Service Requestor) ผู้ใช้บริการจะเป็นผู้เรียกใช้เว็บเซอร์วิส โดยอาจทำการค้นหาเซอร์วิสจากเซอร์วิสไดเร็กทอรี่ แล้วทำการเรียกใช้เซอร์วิสจากผู้ให้บริการ

3. Service Registry หรืออาจเรียกว่า Service Broker มีหน้าที่ในการรับลงทะเบียนและช่วยในการค้นหาเว็บเซอร์วิส Service Registry จะเก็บรายละเอียดของเว็บเซอร์วิสต่างๆเช่น นิยาม และตำแหน่งของเว็บเซอร์วิส ทำหน้าที่คล้ายกับสมุดโทรศัพท์เพื่อช่วยให้ผู้ใช้บริการสามารถค้นหาเซอร์วิสที่ต้องการได้ แต่ในแนวทางการพัฒนาเว็บเซอร์วิสบางแนวทาง อย่างเช่น REST ก็จะตัดองค์ประกอบข้อนี้ออกไป



ภาพที่ 2.10 โมเดลการทำงานของเว็บเซอร์วิส
ที่มา: ศิรประภา ธนุการ, 2553

ในปัจจุบันการพัฒนาเว็บเซอร์วิสนั้นมีทางเลือกในการพัฒนาได้หลายแนวทาง และแนวทางที่ถูกพูดถึงมากที่สุด 2 แนวทางคือ การพัฒนาเว็บเซอร์วิสแบบ SOAP และ การพัฒนาเว็บเซอร์วิสแบบ REST ซึ่งในแต่ละแบบก็จะมีรายละเอียดในการพัฒนารวมถึงข้อดีและข้อด้อยที่ต่างกันไป

2.2.4 การพัฒนาเว็บเซอร์วิสแบบ SOAP

2.2.4.1 มาตรฐานหลักของการพัฒนาเว็บเซอร์วิสแบบ SOAP

มาตรฐานหลักของการพัฒนาเว็บเซอร์วิสแบบ SOAP จะประกอบไปด้วยมาตรฐานต่างๆ ดังนี้ XML WSDL SOAP และ UDDI รายละเอียดของแต่ละมาตรฐานมีดังนี้

1. Extensible Markup Language (XML)

XML เป็นมาตรฐานที่ทาง W3C (World Wide Web Consortium) ประกาศให้เป็นมาตรฐานของข้อมูลเมื่อเดือนกุมภาพันธ์ ปี 1998 โดย XML จะอยู่ในรูปของไฟล์ข้อความที่ใช้ Unicode และสามารถที่สร้างรูปแบบในการที่จะแสดงข้อมูลที่ซับซ้อนในรูปแบบของข้อความที่สามารถอ่านได้ง่าย ในปัจจุบัน XML ได้กลายเป็นมาตรฐานสำคัญสำหรับการกำหนดโครงสร้างข้อมูล เนื้อหา และรูปแบบของข้อมูลของเอกสารอิเล็กทรอนิกส์ และยังมีการพัฒนาเพื่อให้สามารถแลกเปลี่ยนข้อมูลระหว่างหน่วยงาน โปรแกรมประยุกต์ ระบบ และอุปกรณ์ต่างผ่านทางอินเตอร์เน็ตได้อีกด้วย

2. Simple Object Access Protocol (SOAP)

SOAP เป็นภาษา XML เพื่อทำหน้าที่เป็นโพรโทคอลข่าวสาร (Message Protocol) สำหรับการแลกเปลี่ยนข้อมูลระหว่างผู้ให้บริการและผู้ใช้บริการ โครงสร้างของ SOAP จะประกอบไปด้วย

SOAP Envelope: ใช้ในการอธิบายข่าวสาร ระบุเนื้อหา และกระบวนการจัดการข้อมูล

SOAP Transport: ใช้ในการอธิบายโพรโทคอลการส่งข้อมูลเช่น HTTP หรือ SMTP

SOAP Encoding: ใช้ในการอธิบายการเข้ารหัสเพื่อจับคู่ชนิดข้อมูล (data type) ที่ใช้ในโปรแกรมประยุกต์กับ XML elements

โพรโทคอล SOAP เปรียบเสมือนจดหมายที่ใช้ในการสื่อสาร แต่ยังต้องใช้โพรโทคอลในการสื่อสารอื่นๆ เช่น HTTP ในการทำหน้าที่ส่งจดหมาย SOAP เป็นโพรโทคอลแบบข้อความ ซึ่งแตกต่างกับโปรโคคอล IIOP ของ CORBA หรือ JRMP ของ RMI ที่เป็นโพรโทคอลแบบไบนารี่ จึงทำให้ SOAP สามารถที่จะใช้ส่งข้อความข้ามแฟลตฟอร์ม และระบบต่างๆ ได้ และเวอร์ชันล่าสุดของ SOAP คือ 2.0

การส่งข้อความ SOAP มีสองรูปแบบคือ SOAP-RPC และ SOAP message โดย SOAP-RPC ใช้ในการส่งข้อความเพื่อใช้เรียกเมธอดหรือ procedure ซึ่งโดยมากจะเป็นรูปแบบ synchronous โดย SOAP จะส่ง SOAP Request และข้อมูลต่างๆ เพื่อเรียกใช้เมธอดในการประมวลผล และจะรอให้ได้ผลลัพธ์การประมวลผลที่ส่งกลับมาแบบ SOAP Response ส่วน SOAP-message ใช้ในการส่งข่าวสารหรือข้อมูลในรูปแบบ XML ระหว่างผู้ให้บริการและผู้ใช้บริการ โดยสามารถส่งได้ทั้งแบบ Synchronous และ Asynchronous

3. Web Services Description Language (WSDL)

WSDL เป็นภาษา XML ที่ใช้อธิบายเว็บเซอร์วิส โดยจะแบ่งการอธิบายเว็บเซอร์วิสเป็น

สองส่วนดังนี้

ส่วนที่เป็นนามธรรม (Abstract) เพื่ออธิบายโอเปอเรชัน (Operation) อินพุตและเอาท์พุต พารามิเตอร์

ส่วนที่เป็นรูปธรรม (Concrete) เพื่ออธิบายโพรโทคอลของเน็ตเวอร์ค ตำแหน่งของจุดปลายทาง (Endpoint Address) และ รูปแบบของข้อมูล

ในปัจจุบัน W3C ได้ออกข้อกำหนดสำหรับ WSDL เป็นเวอร์ชัน 2.0 แต่คำสั่งบางคำสั่งจะไม่สอดคล้องกับเวอร์ชัน 1.0 ดังนั้นการจะเรียกใช้ WSDL ควรมีการตรวจสอบว่าเครื่องมือที่ใช้พัฒนาสอดคล้องกับเวอร์ชันใดWSDL สามารถเปรียบเทียบได้กับ Java interface ที่ใช้ใน RMI หรือ ภาษา IDL (Interface Description Language) ที่ใช้ใน CORBA สำหรับ Distributed Computing

4. Universal Description, Discovery and Integration (UDDI)

UDDI นิยามรูปแบบและกลไกสำหรับ registry ที่ใช้ในการเก็บและประกาศข้อมูลเกี่ยวกับเว็บเซอร์วิสในรูปแบบของภาษา XML โดยที่ UDDI จะเปรียบเสมือนสมุดโทรศัพท์หน้าเหลืองที่องค์กรธุรกิจต่างๆ ใช้ระบุและโฆษณาหมายเลขโทรศัพท์ขององค์กรเพื่อให้ผู้ใช้โทรศัพท์ค้นหาได้ โดยทั่วไป Service Registry จะใช้ UDDI เป็นมาตรฐานเพื่อให้ผู้ให้บริการสามารถลงทะเบียนประกาศเว็บเซอร์วิสได้ และผู้ใช้บริการก็สามารถจะติดต่อกับ UDDI Registry เพื่อค้นหาเซอร์วิสที่ต้องการและเรียกใช้จากผู้ให้บริการต่อไป

ข้อมูลใน UDDI จะประกอบไปด้วยรายละเอียดเกี่ยวกับองค์กร (businessEntity) รายละเอียดเกี่ยวกับเซอร์วิส (businessService) รายละเอียดเกี่ยวกับการติดต่อ (bindingTemplate) URL สำหรับการเรียกใช้เซอร์วิส (accessPoint) และข้อมูลอ้างอิงไปยัง WSDL (tModelInstanceInfo) มาตรฐาน UDDI ล่าสุดเป็นเวอร์ชัน 3.0 นอกจากนี้เรายังสามารถที่จะแบ่ง Registry ได้เป็นสองประเภท คือ public registry ซึ่งเป็น registry ที่เปิดให้ใช้ทั่วไปทั้งภายใน และภายนอกองค์กร กับ private registry ซึ่งเป็น registry ที่เปิดให้ใช้เฉพาะภายใน การควบคุมดูแล public registry จะเป็นไปได้ยากกว่า จึงทำให้องค์กรส่วนมากจะเริ่มต้นการพัฒนาจาก private registry ก่อน

5. มาตรฐานอื่นๆ ของเว็บเซอร์วิส

มาตรฐาน WSDL SOAP และ UDDI เป็นเพียงมาตรฐานพื้นฐานของเว็บเซอร์วิส การพัฒนาเว็บเซอร์วิสในทางปฏิบัติจำเป็นต้องพิจารณาเรื่องอื่นเช่น ความปลอดภัย Transaction หรือ Messaging เป็นต้น ดังแสดงในรูปที่ 3 ซึ่งแสดงตัวอย่างมาตรฐานเว็บเซอร์วิสอื่นๆ ตามฟังก์ชันของการทำงาน โดยจะมีมาตรฐานที่สำคัญ อาทิเช่น

WS-Addressing: มาตรฐานที่ใช้ร่วมกับ SOAP Header ในการระบุโพรโทคอลการสื่อสารและระบบข่าวสาร (Messaging Systems)

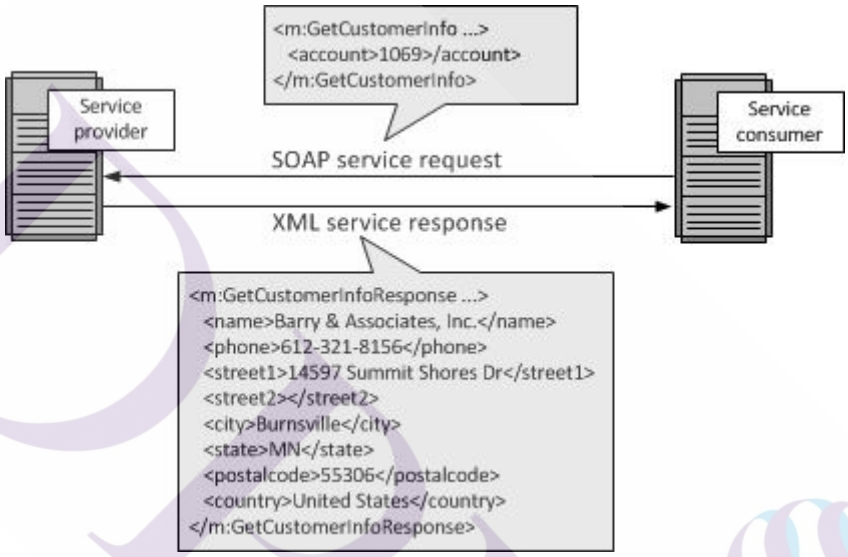
WS-Security: มาตรฐานที่เป็นโครงสร้าง (Framework) เพื่อเชื่อมต่อกับเทคโนโลยีระบบ

ความปลอดภัยต่างๆ

SAML: Security Assertion Markup Language เป็นมาตรฐานที่ทาง OASIS กำหนดขึ้นเพื่อสนับสนุนการทำ Single Sign On (SSO) และ Authentication

WS-BPEL: มาตรฐานสำหรับการประกอบ (orchestration) กระบวนการทางธุรกิจ (Business Process) โดยใช้คำสั่งที่เป็นภาษา XML

WSRP: Web Services for Remote Portal มาตรฐานสำหรับการเรียกใช้ Web Services จากเว็บท่า (Portal)



ภาพที่ 2.11 รูปแบบการทำงานของเว็บเซอร์วิสแบบ SOAP

พื้นฐานการพัฒนาเว็บเซอร์วิสแบบ SOAP นั้นต้องการสร้าง Application logic ให้ออกมาเป็น service ทำให้การตั้งชื่อเป็นดังนี้

getUserInformation

payInvoice

2.2.4.1 ข้อดีและข้อเสียของการพัฒนาเว็บเซอร์วิสแบบ SOAP

ตารางที่ 2.1 ข้อดีและข้อเสียของการพัฒนาเว็บเซอร์วิสแบบ SOAP

ข้อดี	– สามารถทำงานอยู่บน protocol ใด ๆ ก็ได้ – อธิบาย service ด้วย WDSL (Web Service Description Language) – มีความน่าเชื่อถือ เมื่อเกิดปัญหาสามารถทำการ retry ได้ – สนับสนุนเรื่อง security อยู่แล้ว ทั้ง authentication, authorization และ การเข้ารหัสข้อมูล
ข้อเสีย	– ยากต่อการพัฒนา ทำให้ไม่เป็นที่นิยมสำหรับระบบ web และ mobile – สนับสนุนรูปแบบข้อมูล XML เพียงอย่างเดียว – เนื่องจากมันเป็น standard ทำให้มีข้อจำกัดเยอะ – เนื่องจากโครงสร้างมันมีหลายส่วนทำให้มี overhead สูง หรือ ต้องใช้ bandwidthสูงกว่า REST

ดังนั้นควรจะใช้ SOAP เมื่อต้องการจัดการ transaction หรือเมื่อต้องทำงานกับหลาย ๆ ระบบ และต้องการความเข้มงวดในการเชื่อมต่อระหว่าง client/server ตัวอย่างเช่น Financial service และ Telecommunication service ดังนั้นจึงไม่แปลกว่า ทำไมในองค์กรใหญ่ ๆ ถึงใช้ SOAP กันมาก

2.2.5 การพัฒนาเว็บเซอร์วิสแบบ REST

REST (Representational State Transfer) ถูกพูดถึงครั้งแรกในปี 2000 โดย Roy Thomas Fielding เป้าหมายเพื่อเป็นรูปแบบหนึ่งในการออกแบบ Open Web Technology ซึ่ง REST นั้นเป็นแค่ Architecture (สถาปัตยกรรมการสื่อสารข้อมูล) ไม่ใช่มาตรฐาน สังเกตุได้จากไม่มี REST specification อยู่บน W3C และไม่มี REST developer toolkit เพราะว่า REST เป็นเพียงรูปแบบสถาปัตยกรรม เป็นแนวทางใหม่ในการสร้าง Web Service แบบเรียบง่าย โดยเรียกใช้งานผ่านทาง HTTP Method GET/POST/PUT/DELETE และส่งข้อมูลออกมาในรูปของ XML หรือ JSON ทำให้ปริมาณข้อมูลที่รับส่ง น้อยกว่าการใช้โปรโทคอล SOAP อยู่มาก ข้อดีข้อนี้ของ REST ทำให้ นักพัฒนาหลาย ๆ คนหันมาสนใจการเขียนโปรแกรมแบบใช้ RESTful Web Service กันมากขึ้น เพราะมีผลกับเรื่อง Performance ของการใช้งานโปรแกรมด้วย

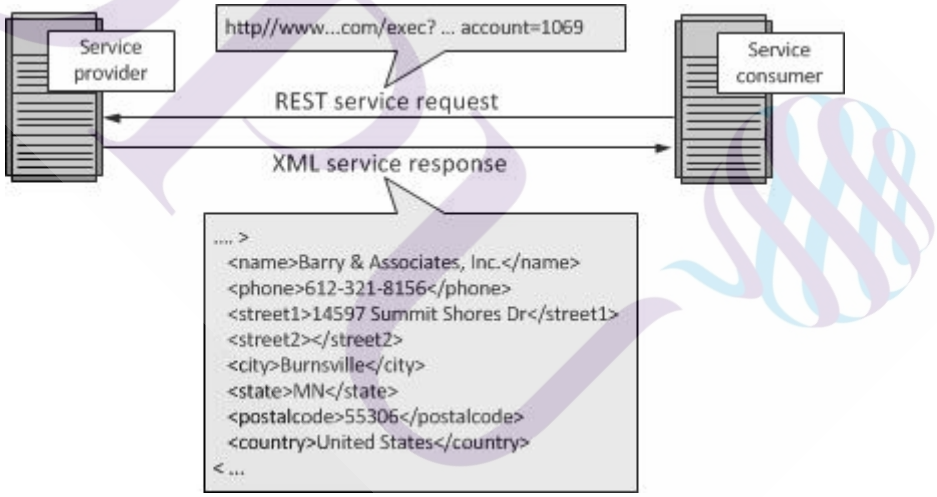
2.2.5.1 พื้นฐานการพัฒนาเว็บเซอร์วิสแบบ REST

พื้นฐานการพัฒนาเว็บเซอร์วิสแบบ REST นั้นเป็นวิธีการที่มองเว็บในฐานะที่เป็นแหล่งข้อมูลหรือทรัพยากร (Resource) โดย Client สามารถเข้าถึงแหล่งข้อมูลนั้นได้ด้วย URL และสามารถจัดการกับแหล่งข้อมูลเหล่านั้นผ่านแอกชั่น (Action) ซึ่งก็คือ HTTP method (GET, POST, PUT, DELETE) ที่เหมาะสม และการดำเนินการต่างๆ จะเป็นการทำงานผ่าน HTTP โปรโทคอลโดยเรียกผ่าน URI ที่สื่อสารได้ชัดเจน ทำให้การตั้งชื่อเป็นดังนี้

userinformation

invoice

รูปแบบการแสดงผล (Representation) ของทรัพยากรนั้นก็จะถูกรีเทิร์นกลับมา รูปแบบการแสดงผลจะถูกนำมาวางลงบน client application ในรูปแบบของสถานะ (State) รูปแบบการแสดงผลใหม่ที่ถูกวางลงบน client application อีกครั้งก็จะถือว่าเป็นอีกสถานะหนึ่ง ดังนั้น client application จะเปลี่ยน (Transfer) สถานะในแต่ละรูปแบบการแสดงผลของทรัพยากร จึงได้ชื่อว่า Representational State Transfer และรูปแบบของข้อมูลที่ใช้สื่อสารกันภายใต้การทำงานแบบ REST จะอยู่ในรูปแบบ XML, JSON, Atom และอื่นๆ อีกมากมาย แต่หนึ่งในรูปแบบที่ได้รับความนิยมคือ JSON



ภาพที่ 2.12 รูปแบบการทำงานของเว็บเซอร์วิสแบบ REST

สามารถสรุปคุณสมบัติของ REST ได้ดังนี้คือ

1. แสดงผล
2. เก็บข้อมูล
3. มี URIs สำหรับการเข้าถึง resourse ต่างๆในระบบ
4. Stateless ทำงานโดยไม่ต้องมี session
5. เชื่อมต่อระหว่าง web service
6. รองรับเรื่อง caching ข้อมูล

2.2.5.2 หลักการออกแบบเว็บเซอร์วิสแบบ REST

หลักการออกแบบเว็บเซอร์วิสแบบ REST มีดังต่อไปนี้

1. การตั้งชื่อต้องใช้ nouns ควรหลีกเลี่ยงการตั้งชื่อด้วย verbs และอย่าตั้งชื่อด้วยตัวเล็กตัวใหญ่

ตารางที่ 2.2 ตัวอย่างการตั้งชื่อเว็บเซอร์วิสแบบ REST

Resource	/books	/books/2 /books/{:id}	/books/authors
GET read	ดึงข้อมูลหนังสือทั้งหมด	ดึงข้อมูลหนังสือตามไอดี	ดึงข้อมูลผู้แต่งหนังสือทั้งหมด
POST create	สร้างหนังสือใหม่	ไม่อนุญาตให้เข้าถึง (HTTP 405)	สร้างข้อมูลผู้แต่งหนังสือ
PUT update	อัพเดตข้อมูลหนังสือทั้งหมด	อัพเดตหนังสือตามไอดี	อัพเดทข้อมูลผู้แต่งหนังสือทั้งหมด
DELETE	ลบหนังสือทั้งหมด	ลบหนังสือ	ลบผู้แต่งหนังสือทั้งหมด

2. อย่าใช้ GET ในการแก้ไขข้อมูล หรือสร้างข้อมูลใหม่ แนะนำให้ใช้ POST เพื่อความปลอดภัยหลีกเลี่ยงการโดนทำ Injection หรือการส่งค่าไม่พึงประสงค์ไปในระบบ

3. ให้ใช้คำ nouns แบบพหูพจน์ (plural nouns)

ใช้ /books แทน /book

ใช้ /authors แทน /author

ใช้ /groups แทน /group

4. ใช้ Standard HTTP Method

ตารางที่ 2.3 Standard HTTP Method

Method	ลักษณะการทำงาน	รูปแบบ URL
GET	สำหรับการดึงข้อมูลแบบหลายรายการ	http://example.com/products/
	สำหรับการดึงข้อมูลแบบทีละรายการ	http://example.com/products/12
POST	สำหรับการสร้างข้อมูล	http://example.com/products/
PUT	สำหรับการแก้ไขข้อมูล	http://example.com/products/12
DELETE	สำหรับลบข้อมูล	http://example.com/products/12

5. เปิดใช้งาน overriding HTTP method เพราะบาง server หรือบาง Proxies เปิดให้เราส่งได้แค่ GET/POST ดังนั่นจำเป็นต้องทำ HTTP Header X-HTTP-Method-Override เพื่อเปลี่ยนจากการทำ POST ไปเป็น PUT หรือ DELETE

6. ใช้ sub name ในการเชื่อมความสัมพันธ์ของข้อมูล

GET /books/2/author ดึงข้อมูลของผู้แต่งของหนังสือหมายเลข 2

GET /books/2/author/10 ดึงข้อมูลผู้แต่งหมายแลข 10 ที่หนังสือหมายเลข 2

7. ควรระบุ content-type ของการสื่อสารด้วยทั้ง client และ server

8. ระบุ version API

/api/v1/book/author

9. ใช้ HTTP Status Codes คู่ไปกับการใช้ Message ในการจัดการกับ Error ของระบบ

200 OK – [GET]

201 CREATED – [POST/PUT/PATCH]

204 NO CONTENT – [DELETE]

304 NOT MODIFIED

400 INVALID REQUEST – [POST/PUT/PATCH]

401 UNAUTHORIZED

403 FORBIDDEN

404 NOT FOUND

500 INTERNAL SERVER ERROR

10. ในการดึงข้อมูลที่ซับซ้อน หรือมี state เยอะๆ ให้ทำ Query String ด้วย GET

การดึงเอาสถานะของหนังสือที่ยังขายอยู่เช่น

GET /api/v1/books?state=true

หรือการเลือกแสดงผลข้อมูลว่าต้องการที่จะดูฟิลด์ Database ในฟิลด์ไหนบ้าง

GET /api/v1/books?fields=name,price,author,type,categories

การทำ sorting หรือ paginate ก็เช่นกัน

GET

/api/v1/books?fields=name,price,author,type,categories&sort=asc&page=1&limit=10

ข้อควรจำอย่าลืมเช็ค Escape string ในกรณีที่ฐานข้อมูลเป็นแบบ Relationship เช่นพวกที่ใช่คำสั่ง SQL ทั้งหลายในการคิวรี่และอย่าลืมเช็คความถูกต้องของข้อมูลก่อนส่งไปดึงที่ Database เสมอเพื่อไม่ให้โดน Cross-site scripting หรือ Injection ต่างๆ

11. ทำเอกสารเพื่ออธิบายเซอร์วิสต่างๆ

2.2.5.3 ข้อดีและข้อเสียของการพัฒนาเว็บเซอร์วิส

ตารางที่ 2.4 ข้อดีและข้อเสียของการพัฒนาเว็บเซอร์วิสแบบ REST

<u>ข้อดี</u>	– ทำการอยู่บน HTTP และทำตามมาตรฐานของ HTTP จึงทำให้พัฒนาได้ง่าย – สนับสนุนรูปแบบข้อมูลมากมาย เช่น XML, JSON, Plain Text และอื่น ๆ อีกมากมาย – รองรับการขยายระบบได้ง่าย – มีประสิทธิภาพการทำงานที่ดี – รองรับเรื่อง caching ข้อมูล
<u>ข้อเสีย</u>	– ทำงานได้เฉพาะ HTTP protocol เท่านั้น – ไม่มีเรื่องของ security และ reliability มาให้ในตัว ดังนั้นต้องทำเอง – รูปแบบข้อมูลที่ส่งไปมาระหว่าง client-server ไม่มีข้อจำกัดอะไรเลย

ดังนั้นควรจะใช้ REST เมื่อต้องการลดขนาดของข้อมูล และ จำนวน Bandwidth ที่ใช้งาน และเมื่อทำงานอยู่บนระบบ Web หรือ Mobile ตัวอย่างเช่น Social media service, Web Chat service ดังนั้นจึงไม่แปลกว่า ทำไมระบบ Web และ API ต่าง ๆ ผ่าน Web จึงเป็น REST

และจากการสังเกตแนวโน้มของเว็บเซอร์วิสที่พัฒนาและเรียกใช้จริงในขณะนี้ พบว่าบริษัทใหญ่ ๆ เริ่มที่จะเลิกสนับสนุนการเรียกใช้ SOAP Web services และบางบริษัทไม่ได้สนับสนุนตั้งแต่แรก เช่น บริษัท Google ได้หยุดการพัฒนาฟังก์ชันใหม่ของ SOAP Search API ตั้งแต่วันที่ 5

ธันวาคม 2549 ส่วนบริษัท Amazon ก็ได้หยุดการให้บริการ Amazon Web services โดยใช้ SOAP กับภาษา Ruby on Rails ส่วนบริษัท Yahoo ไม่เคยสนับสนุนการเรียกใช้ SOAP Web services ตั้งแต่เริ่มให้บริการต่าง ๆ

2.3 OpenAPI Specification

เนื่องจากการพัฒนาเว็บเซอร์วิสในรูปแบบ REST ยังไม่มีมาตรฐานที่กำหนดให้บังคับใช้งานเหมือน SOAP ซึ่งมาตรฐานต่างๆเหล่านี้ ร่วมไปถึงมาตรฐานที่ใช้อธิบายเว็บเซอร์วิส โดยในฝั่งของ SOAP เองจะมี WSDL (Web Services Description Language) ซึ่งจะเป็นเอกสารที่อยู่ในรูปแบบของ XML มีไว้ใช้อธิบายรายละ เอียดของเว็บเซอร์วิส เพื่อให้แอปพลิเคชันที่ต้องการเรียกใช้เว็บเซอร์วิสทราบรายละเอียดของบริการและวิธีการเรียกใช้ โดย WSDL จะบรรจุรายละเอียดที่ต้องใช้ในการสร้าง Request message เพื่อร้องขอบริการเช่น ชื่อบริการ, พารามิเตอร์ หรือโปรโตคอลที่ใช้ในการติดต่อสื่อสาร เพราะจะต้องมีเอกสาร Service Description เพื่อบอกรายละเอียดและวิธีการเรียกใช้ Service ซึ่งในฝั่งการทำงานของ REST นั้นเนื่องจากไม่มีมาตรฐานกำหนดไว้ ปัญหาใหญ่ที่พบบ่อยในการพัฒนาระบบคือการทำเอกสารเพื่ออธิบาย API ต่างๆที่พัฒนาขึ้น จึงทำให้มีการรวมกลุ่มขึ้นมาเพื่อสร้างข้อกำหนด เพื่อเป็นกรอบหรือแนวทางการทำงานกับ RESTful Web Service เรียกว่า OpenAPI Specification

2.3.1 ประวัติความเป็นมา

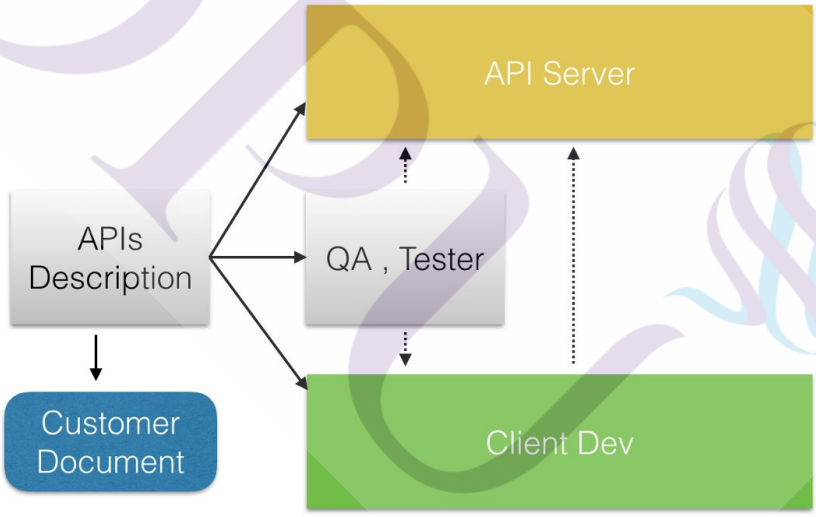
OpenAPI Specification (OAS) หรือที่รู้จักกันครั้งแรกในชื่อ Swagger Specification ซึ่งเกิดขึ้นจากการรวมตัวของกลุ่มอุตสาหกรรมและนักพัฒนาซอฟท์แวร์โดยใช้ชื่อว่า Open API Initiative วัตถุประสงค์คือการกำหนดมาตรฐานและสร้างกรอบการทำงาน สำหรับ REST APIs ขึ้นมา โดย OpenAPI Specification นั้นจะเป็นข้อกำหนดสำหรับการสร้างชุดข้อมูลที่สามารถอ่านและเข้าใจได้โดยมนุษย์และคอมพิวเตอร์ ซึ่งข้อมูลชุดนี้จะสามารถอธิบายโครงสร้างของ API ได้ว่ามีการทำงานอย่างไร มีการส่งข้อมูลและได้ผลตอบกลับอย่างไร รวมไปถึงความสามารถในการนำไปใช้ประโยชน์ในการทำงานอื่นๆ เช่น การสร้างเอกสาร และการสร้างชุดของ Source code เพื่อรองรับการทำงานรวมกับ API ในฝั่ง Server ซึ่งปัจจุบันได้มีการพัฒนาเครื่องมือหลายๆตัวขึ้นมาเพื่อใช้ทำงานร่วมกันภายใต้ OpenAPI Specification

OpenAPI Specification เกิดขึ้นจากการริเริ่มของ Wordnik ซึ่งเป็นองค์การไม่แสวงหาผลกำไร โดยตัว Wordnik เองเป็นแหล่งข้อมูลทางภาษา ที่ให้บริการ Dictionary ภาษาอังกฤษแบบออนไลน์ รวมไปถึง Thesaurus เก็บรวบรวมข้อมูลอธิบายความหมายของคำที่มีความหมายเหมือนหรือคล้ายกัน โดยเริ่มจาก Swagger ซึ่งถูกพัฒนาขึ้นเพื่อใช้ในการทำงานภายในของ Wordnik เอง โดยมีการเริ่มพัฒนาขึ้นในช่วง ต้นปี ค.ศ. 2010 และในช่วงเดือนมีนาปี 2015 บริษัท SmartBear Software ได้เข้ามาพัฒนา Swagger API specification ซึ่งเป็น Open Source Project ต่อจาก Reverb Technologies และในช่วงเดือนพฤศจิกายนของปีเดียวกัน SmartBear ได้ประกาศว่ากำลังจะมีการจัดตั้งองค์กรใหม่ขึ้นด้วยการสนับสนุนจาก Linux Foundation โดยใช้ชื่อว่า Open API Initiative (OAI) โดยมีหลายบริษัทชั้นนำเป็นผู้รวมก่อตั้ง เช่น Google, IBM และ Microsoft ซึ่ง SmartBear ได้ยก Swagger Specification ซึ่งมี

เวอร์ชั่นล่าสุด คือ 2.0 ให้กับทางกลุ่ม ทั้งนี้ RAML และ API Blueprint ซึ่งเป็น framework ที่มีวัตถุประสงค์การทำงานคล้ายกันก็อยู่ในรายชื่อที่ได้รับการพิจารณาจากกลุ่มด้วย และต่อมาเมื่อวันที่ 1 มกราคม ปี 2016 Swagger Specification ได้ถูกเปลี่ยนชื่อมาเป็น OpenAPI Specification อย่างเป็นทางการ และในปี 2017 ก็ได้ออก preview เวอร์ชั่น 3.0 โดยใช้ชื่อว่า OpenAPI Specification 3.0 โดยมีทาง MuleSoft ที่เป็นผู้ร่วมพัฒนาหลักของโปรเจคทางเลือกอย่าง RESTful API Modeling Language (RAML) เข้ามาร่วมทำงานกับ OAS อีกด้วย

2.3.2 การนำ OpenAPI Specification มาใช้งาน

แอปพลิเคชันที่ถูกพัฒนาขึ้นมาบนพื้นฐานของ OpenAPI Specification จะสามารถสร้างเอกสารอธิบาย Method, Parameter และ Model ของ Response ในการเรียกใช้งานแต่ละ service ได้โดยอัตโนมัติ ซึ่งช่วยให้ เอกสาร รวมถึง Client Library และ การทำงานในฝั่ง Server มีความสอดคล้องกัน โดยการกำหนดมาตรฐานนี้ จะบังคับให้เราเขียน API description ให้อยู่ในรูปแบบที่ได้กำหนดไว้ เพื่อที่จะใช้เป็นตัวกลางในการสื่อสารระหว่างผู้ที่เกี่ยวข้อง ตั้งแต่ Developer, Tester รวมไปถึง คนทำเอกสารด้วย



ภาพที่ 2.13 แสดงถึงหลักการทำงานของ OpenAPI Specification

เรายังสามารถนำ Swagger หรือ OpenAPI เข้ามาช่วยในการพัฒนา API ของเราได้ดังต่อไปนี้

1. สำหรับการพัฒนาแบบ Design-first หรือการออกแบบ API ไว้ก่อน แล้วจึงลงมือเขียนโค้ดตามนั้นสามารถใช้ Swagger Codegen เพื่อสร้าง Server Stub สำหรับ API เพื่อเป็นการ Mock-up ข้อมูลเพื่อใช้ในการทดสอบได้ก่อน แล้วจึงเขียน Server logic ตามไปทีหลัง วิธีนี้สามารถทำให้การ

พัฒนาแอปพลิเคชันฝั่ง Server และ Client สามารถทำงานไปพร้อมกันได้

2. สามารถใช้ Swagger Codegen เพื่อสร้าง Library ในฝั่ง Client เพื่อเรียกใช้งาน API โดยมีภาษาที่รองรับมากกว่า 40 ภาษา

3. สามารถใช้ Swagger UI เพื่อสร้าง Interactive API documentation เพื่อให้นักพัฒนาสามารถลองยิงทดสอบเรียก API ได้เลยผ่านเว็บเบราเซอร์

4. สามารถใช้ไฟล์ Spec ทำงานร่วมกับ เครื่องมือที่เกี่ยวข้องกับ API อื่นๆเพื่อเชื่อมต่อเรียกใช้งาน API ของเรา เช่น นำเข้าไฟล์ Spec ไปยัง SoapUI เพื่อทำ Automated Test กับ API ที่พัฒนาขึ้น

2.3.3 เครื่องมือที่ใช้กับ OpenAPI Specification

เครื่องมือหลักที่ถูกพัฒนาขึ้นมาเพื่อใช้กับ OpenAPI Specification มีดังต่อไปนี้

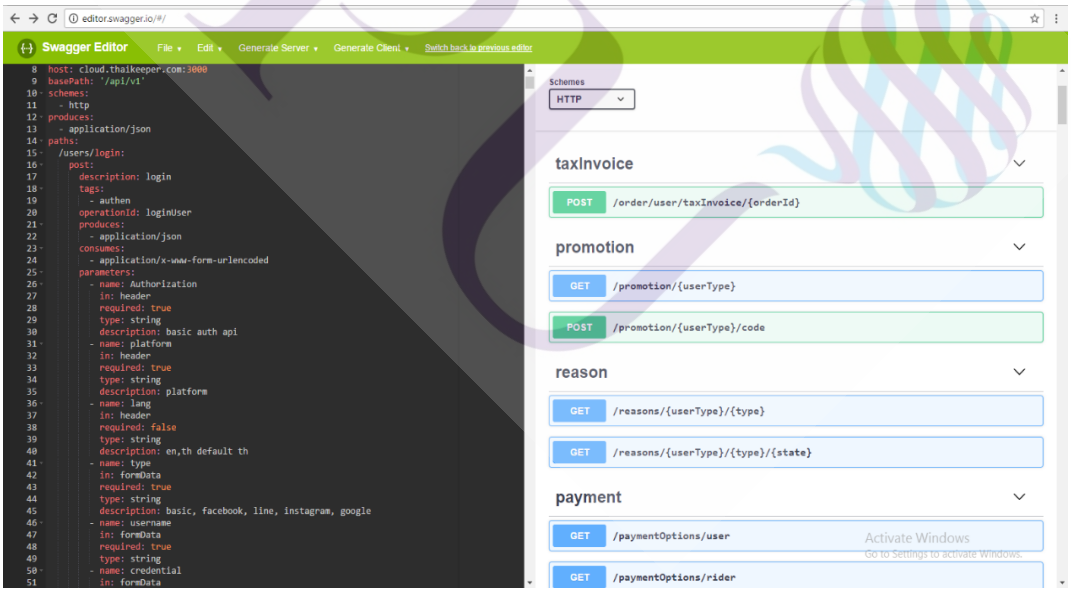
Swagger Editor

Swagger Codegen

Swagger UI

2.3.3.1 Swagger Editor

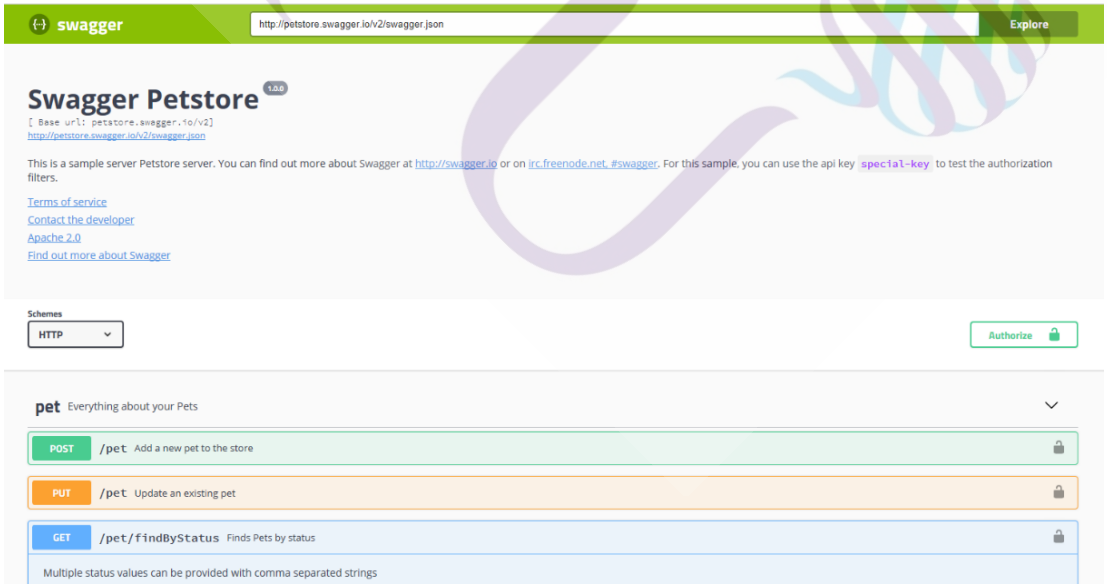
Swagger Editor เป็นเครื่องมือที่ไว้ช่วยเขียน API description รองรับทั้ง แบบ Yaml และ JSON ซึ่งนอกจากการทำตัว เป็น Text Editor แล้ว ทางด้านขวาจะมี Live render จาก API description ที่เขียนมาแสดง และยังสามารถส่ง Request ทดสอบ API ได้จริงอีกด้วย



ภาพที่ 2.14 แสดงหน้าจอการทำงานของ Swagger Editor

```yaml
swagger: "2.0"
info:
  description: "This is a sample server Petstore server.  You can find out more about
    Swagger at [http://swagger.io](http://swagger.io) or on [irc.freenode.net, #swagger](http
    ://swagger.io/irc/).      For this sample, you can use the api key `special-key` to test
    the authorization     filters."
  version: "1.0.0"
  title: "Swagger Petstore"
  termsOfService: "http://swagger.io/terms/"
  contact:
    email: "apiteam@swagger.io"
  license:
    name: "Apache 2.0"
    url: "http://www.apache.org/licenses/LICENSE-2.0.html"
host: "petstore.swagger.io"
basePath: "/v2"
tags:
- name: "pet"
  description: "Everything about your Pets"
  externalDocs:
    description: "Find out more"
    url: "http://swagger.io"
- name: "store"
  description: "Access to Petstore orders"
- name: "user"
  description: "Operations about user"
  externalDocs:
    description: "Find out more about our store"
    url: "http://swagger.io"
schemes:
- "http"
paths:
  /pet:
    post:
      tags:
      - "pet"
      summary: "Add a new pet to the store"
      description: ""
```

ภาพที่ 2.15 แสดงถึงส่วน header ของ swagger และ GET method

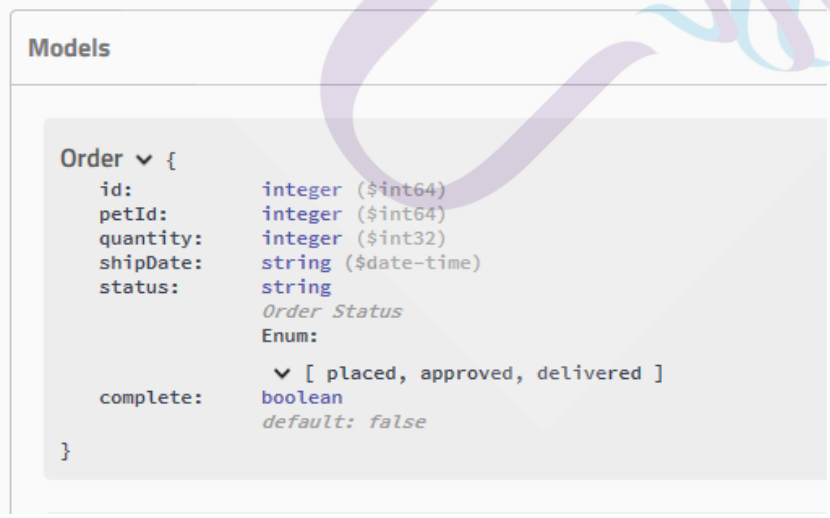


ภาพที่ 2.16 แสดงผลจากการ render Swagger ส่วนของ header

ใน swagger จะมีการกำหนด Model ของ Object เพื่อสามารถใช้กำหนดให้เป็นข้อมูล Response ที่ได้จากการเรียก API จากตัวอย่างจะเป็น Model ของ Order ที่ประกอบไปด้วย property id, petId, quantity เป็นต้น

```
578 definitions:
579   Order:
580     type: "object"
581     properties:
582       id:
583         type: "integer"
584         format: "int64"
585       petId:
586         type: "integer"
587         format: "int64"
588       quantity:
589         type: "integer"
590         format: "int32"
591       shipDate:
592         type: "string"
593         format: "date-time"
594       status:
595         type: "string"
596         description: "Order Status"
597         enum:
598         - "placed"
599         - "approved"
600         - "delivered"
601       complete:
602         type: "boolean"
603         default: false
```

ภาพที่ 2.17 แสดงการกำหนด Model ของข้อมูลใน Swagger



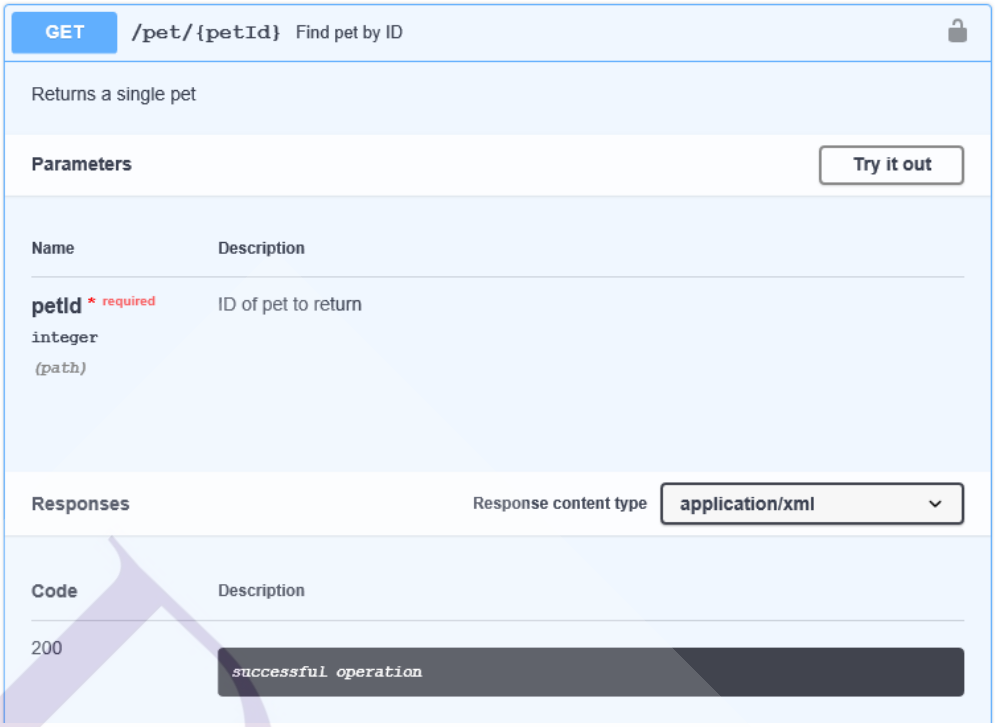
ภาพที่ 2.18 แสดงผลลัพธ์ที่ได้จากการ Render Model

ต่อไปจะเป็นการกำหนด GET Method ที่ใช้ model “Pet” มาเป็นค่า response รวมไปถึงการกำหนด parameter ของ API แบบที่อยู่ใน URL path ด้วย

```
157  /pet/{petId}:
158    get:
159      tags:
160      - "pet"
161      summary: "Find pet by ID"
162      description: "Returns a single pet"
163      operationId: "getPetById"
164      produces:
165      - "application/xml"
166      - "application/json"
167      parameters:
168      - name: "petId"
169        in: "path"
170        description: "ID of pet to return"
171        required: true
172        type: "integer"
173        format: "int64"
174      responses:
175        200:
176          description: "successful operation"
177          schema:
178            $ref: "#/definitions/Pet"
179        400:
180          description: "Invalid ID supplied"
181        404:
182          description: "Pet not found"
```

ภาพที่ 2.19 แสดง Parameters ที่อยู่ใน URL Path

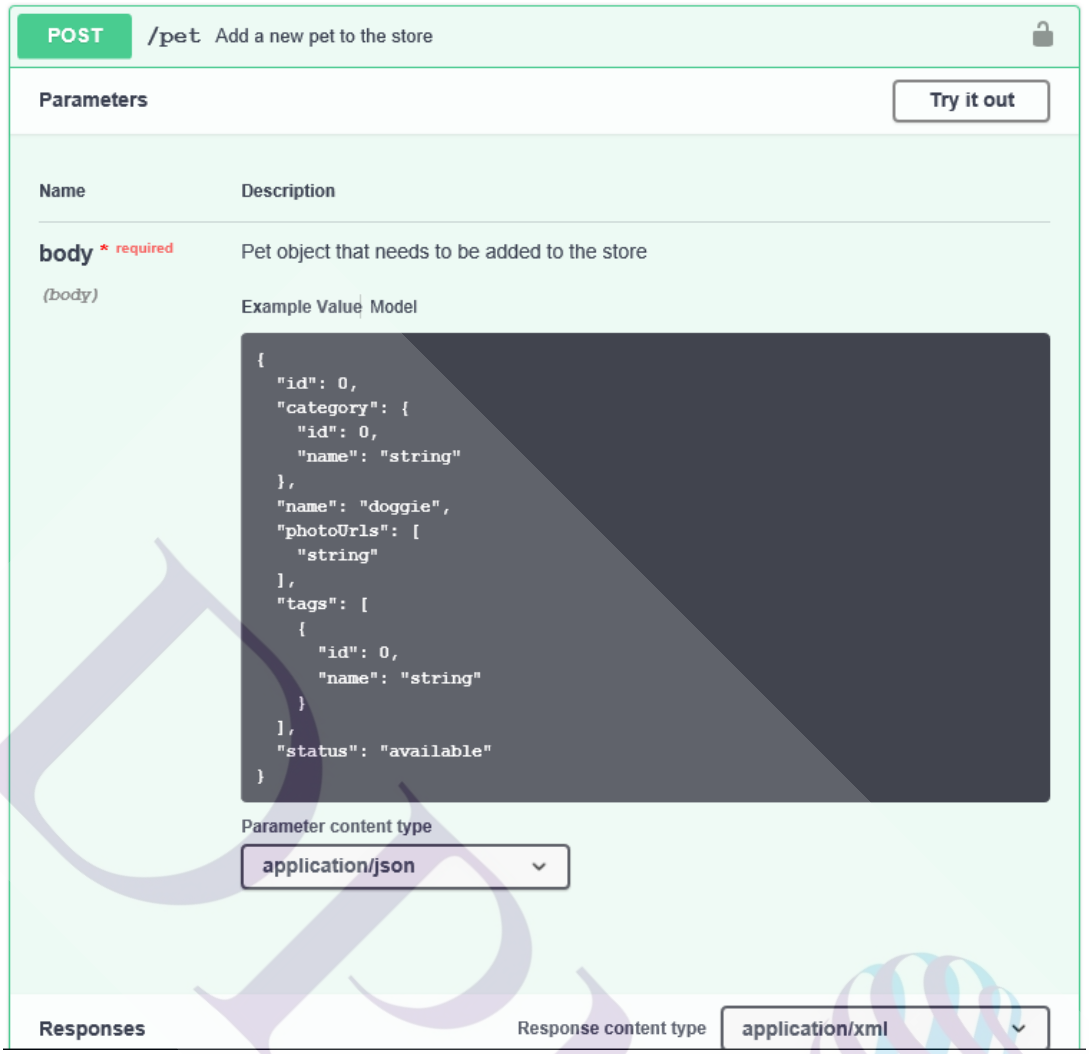
Live render view จะแสดง GET Method ในรูปด้านล่าง จากตรงนี้หากต้องการทดสอบลองยิง Request API ก็สามารถกดที่ปุ่ม [Try it out]



ภาพที่ 2.20 แสดงผลลัพธ์ที่ได้จากการ Render GET method

```
paths:
  /pet:
    post:
      tags:
      - "pet"
      summary: "Add a new pet to the store"
      description: ""
      operationId: "addPet"
      consumes:
      - "application/json"
      - "application/xml"
      produces:
      - "application/xml"
      - "application/json"
      parameters:
      - in: "body"
        name: "body"
        description: "Pet object that needs to be added to the store"
        required: true
        schema:
          $ref: "#/definitions/Pet"
      responses:
        405:
          description: "Invalid input"
```

ภาพที่ 2.21 แสดงการกำหนด POST Method

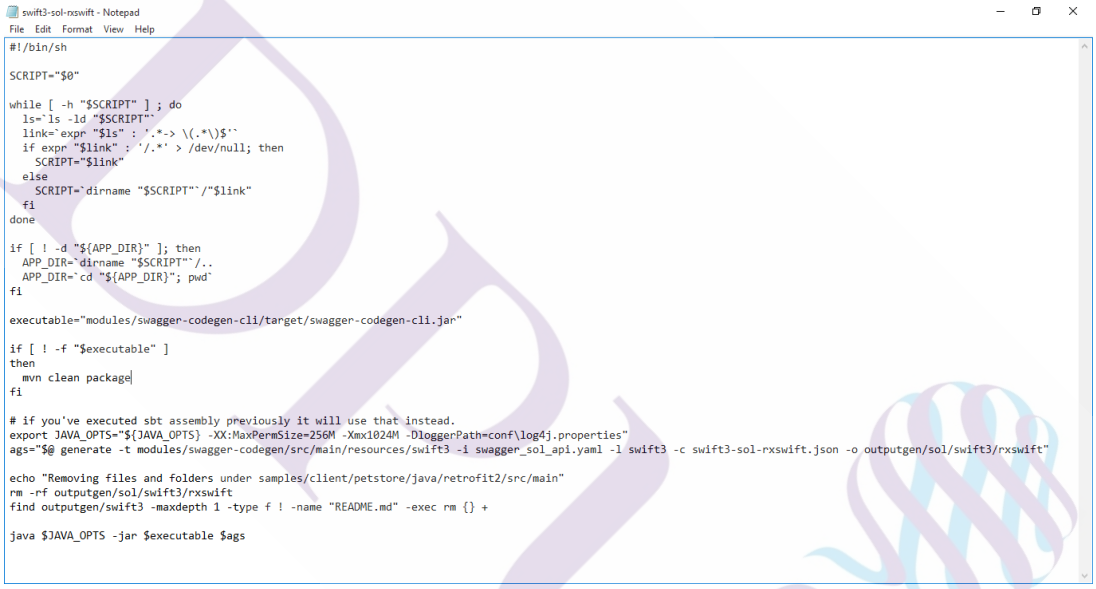


ภาพที่ 2.22 แสดงผลลัพธ์ที่ได้จากการ Render POST method

2.3.3.2 Swagger Codegen

Swagger Codegen เป็น Open Source Code-generator ใช้สำหรับสร้าง Server Stub และ SDK สำหรับฝั่ง Client จาก API description ที่เราได้เขียนขึ้นมา วิธีการใช้งานหลังติดตั้งหลักๆจะต้องเรียกเป็น Command line เพื่อ generate ตัว Source Code ออกมาหรือสามารถเขียนเป็น Shell Script เพื่อ Run คำสั่งเหล่านั้น ซึ่งใน Project Swagger Codegen ได้รวมตัวอย่างไฟล์ Script เหล่านี้เข้าไว้ในตัวอย่างของ Project ด้วย โดยก่อนจะติดตั้งและเรียกใช้งานจำเป็นต้องติดตั้ง Java เวอร์ชัน 7 ขึ้นไปก่อน จึงจะสามารถติดตั้งและเรียกใช้งาน Swagger Codegen ได้ นักพัฒนาสามารถ Download และศึกษาวิธีการติดตั้งรวมถึงวิธีการใช้งานได้จาก

GitHub: https://github.com/swagger-api/swagger-codegen

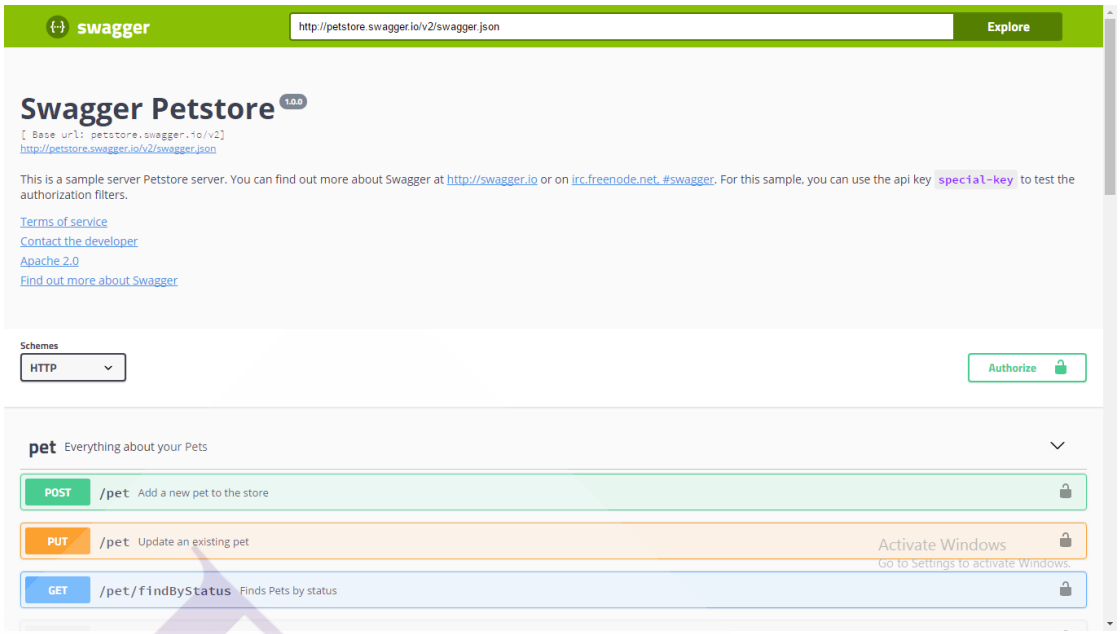


ภาพที่ 2.23 แสดงตัวอย่าง Shell Script สำหรับ Generate Source Code

2.3.3.3 Swagger UI

Swagger UI เป็น Open Source Project ใช้สำหรับ Render เอกสารจาก API description ที่เขียนขึ้นให้ออกมาในรูปแบบเว็บเพจ และสามารถเรียกดูเอกสารได้ผ่าน Web Browser โดยสามารถ Download และศึกษาวิธีการติดตั้งรวมถึงวิธีการใช้งานได้จาก

GitHub: https://github.com/swagger-api/swagger-ui



ภาพที่ 2.24 แสดงหน้าเว็บเพจเอกสาร API ที่ได้จาก Swagger UI

นอกจากเครื่องมือที่กล่าวมานี้ ยังมีเครื่องมืออีกหลายตัวทั้งที่เป็น Open Source และ ที่ใช้ในเชิงพาณิชย์ ที่สามารถทำงานร่วมกับ Swagger Specification หรือ OpenAPI Specification ได้อีกด้วย ซึ่งนักพัฒนาสามารถเข้าไปศึกษารายละเอียดได้จากเว็บไซต์หลักของ Swagger http://swagger.io/ หรือจากเว็บไซต์ของ OpenAPI Initiative https://www.openapis.org/

2.4 WebSocket

Socket คือกลุ่มของ IP และ Port ที่ใช้ระบุถึงตัวตนของอุปกรณ์ที่ใช้งานในเครือข่าย โดย Socket จะมีการทำงานเป็นคู่ โดยต้องตกลงวิธีรับส่งข้อมูลหรือการกำหนด Protocol กันไว้ในทั้งสองฝั่งของ Socket และเมื่อมีการยืนยันตัวตนแบบ CHAP (Challenge-Handshake Authentication Protocol) อย่างถูกต้อง จึงจะถือว่าอุปกรณ์ทั้งคู่เกิดการสถาปนาการเชื่อมต่อกัน (Establishing)

WebSocket เป็นเทคโนโลยีเพื่อใช้ในการติดต่อสื่อสารระหว่าง Web Server กับ Client แบบ Real-Time โดย WebSocket โปรโตคอลได้รับกำหนดมาตรฐานโดย IETF เป็น RFC 6455 และ WebSocket API ใน Web IDL กำลังจะได้รับการกำหนดมาตรฐานโดย W3C ซึ่ง WebSocket นั้นเป็นมาตราฐานการส่งผ่านข้อมูลแบบ full-duplex หรือการรับส่งข้อมูลระหว่างกันแบบสองทาง (2-way communications) โดยทำงานอยู่บนโปรโตคอล TCP ในชั้น Transport Layer ผ่านมาตรฐาน HTTP หรือ HTTPS ซึ่งหลังจากเปิดการเชื่อมต่อแล้วจะไม่มีการส่งคำร้องไปยัง Server อีก ทำให้ลดภาระของ Server ลดการส่งข้อมูลบนเครือข่ายลงได้มาก สามารถเชื่อมต่อพร้อมกันได้มากขึ้น

เบราเซอร์ส่วนใหญ่สามารถรองรับการทำงานของ WebSocket โปรโตคอลได้หมดแล้ว เช่น Google Chrome, Microsoft Edge, Internet Explorer, Firefox, Safari และ Opera ทั้งนี้ แอปพลิเคชันในฝั่ง Server ก็ต้องเขียนให้รองรับการทำงานด้วยเช่นกัน ถึงจะสามารถทำงานร่วมกันได้ และถึงแม้ว่า WebSocket จะถูกออกแบบมาเพื่อการใช้งานในเว็บเบราเซอร์และเว็บเซิร์ฟเวอร์ แต่ก็สามารถนำมาใช้กับ Server หรือ Client ใดๆก็ได้ แต่ Web ที่เราพัฒนาทั่วๆไปไม่จำเป็นจะต้องใช้ WebSocket การนำมาใช้งานจะขึ้นอยู่กับความเหมาะสมและความจำเป็น เช่น เว็บประเภท Chat, Vote, Bid เป็นต้น เพราะเว็บหล่านี้ต้องการข้อมูลที่รวดเร็ว Real Time ในทันที และทำให้การรับ-ส่งข้อมูลจาก Client ไปยัง Server และจาก Server ไปยัง Client ต่างๆ นั้นมีประสิทธิภาพสูงสุด

2.4.1 หลักการทำงานของ WebSocket

WebSocket กำหนดการใช้รูปแบบ URI เป็น 2 รูปแบบ คือ WS สำหรับการเชื่อมต่อปกติไม่ได้เข้ารหัส และ WSS สำหรับการเชื่อมต่อที่เข้ารหัส เช่น ws://localhost:8080 และ wss://localhost:8080

WebSocket มีการทำงานแบ่งเป็น 2 ส่วนด้วยกันคือ

Protocol handshake คือการสถาปนาและ Upgrade โปรโตคอลการเชื่อมต่อ

Data transfer คือการรับส่งข้อมูลที่ใช้งานจริง

2.4.1.1 Protocol handshake

Protocol handshake คือการทำ Handshake เพื่อสถาปนาและ Upgrade การเชื่อมต่อ โดยการทำงานจะเริ่มต้นโดย Client ส่ง Request เช่นเดียวกับ HTTP Request ปกติด้วยการระบุใน Connection header เป็น Upgrade และระบุใน Upgrade header เป็น websocket เพื่อบอก HTTP Server ว่าจะข้อ

เชื่อมต่อและ Upgrade โปรโตคอล จาก HTTP เป็น WebSocket พร้อมด้วย Header ที่จำเป็นอื่นๆ เมื่อ Server ได้รับ Request และรองรับการทำงานกับ WebSocket ก็จะตอบกลับมา ด้วย Connection header และ Upgrade header เช่นเดียวกับในขา Request จึงถือเป็นเป็นการสิ้นสุด การทำ Handshake และเริ่มรับส่งข้อมูลระหว่างกันได้

ตารางที่ 2.5 ตัวอย่าง WebSocket Protocol handshake Request Message

<table>
<tr><td colspan="2">Client request :

GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: x3JJHMbDL1EzLkh9GBhXDw==
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
Origin: http://example.com</td></tr>
<tr><td colspan="2">มีพารามิเตอร์ที่เกี่ยวข้องกับ websocket ดังนี้</td></tr>
<tr><td>Upgrade: websocket

Connection: Upgrade</td><td>เป็นค่าเพื่อบอก Server ว่านี่เป็นการติดต่อมาจาก websocket ให้ทำการสลับการทำงานจาก HTTP Protocol ไปเป็น WebSocket Protocol</td></tr>
<tr><td>Sec-WebSocket-Key: x3JJHMbDL1EzLkh9GBhXDw==</td><td>จะอยู่ในรูปของ base64 encode ทาง Server จะเอาค่านี้ไปใช้งานเพื่อสร้าง handshake สำหรับคุยกันระหว่าง websocket client/server</td></tr>
<tr><td>Origin: http://example.com</td><td>ส่วนของ origin url จะถูกสร้างโดย browser คือ client</td></tr>
<tr><td>Sec-WebSocket-Version: 13</td><td>เป็นการบอก websocket Server ว่าทางฝั่ง Client นั้นใช้ protocol version ไหน ซึ่ง version 13 นั้นเป็น final spec ที่ทาง IEFT กำหนดให้เป็น standard</td></tr>
</table>

ตารางที่ 2.6 ตัวอย่าง WebSocket Protocol handshake Response Message

<table>
<tr><td colspan="2">Server response:

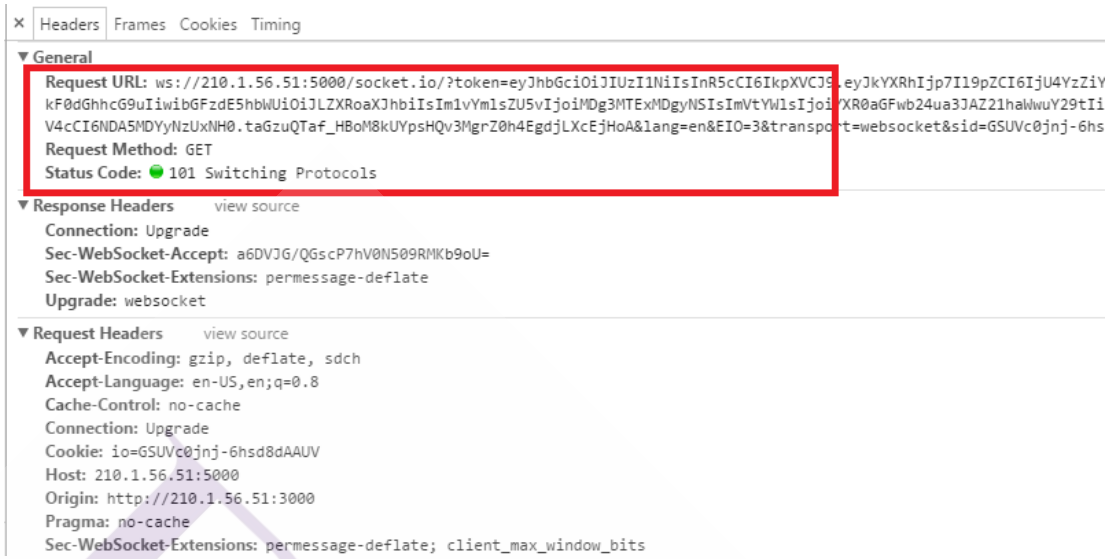
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: HSmrc0sMlYUkAGmm5OPpG2HaGWk=
Sec-WebSocket-Protocol: chat</td></tr>
<tr><td colspan="2">มีพารามิเตอร์ที่เกี่ยวข้องกับ websocket ดังนี้</td></tr>
<tr><td>Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Accept:
HSmrc0sMlYUkAGmm5OPpG2HaGWk=</td><td>ยอมรับการเชื่อมต่อและการ Upgrade Protocol

ตอบกลับด้วยค่า hash ของ key ที่ได้รับมาในขา request เพื่อเป็นการป้องกันการทำ Caching proxy จากการส่งข้อความก่อนหน้า และจะอยู่ในรูปของ base64 encode</td></tr>
</table>

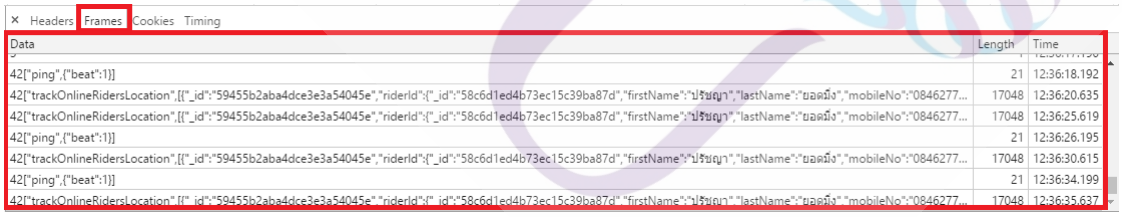
เมื่อมี response ตอบกลับมาตัว HTTP Code จะเป็นค่า 101 Switching Protocols



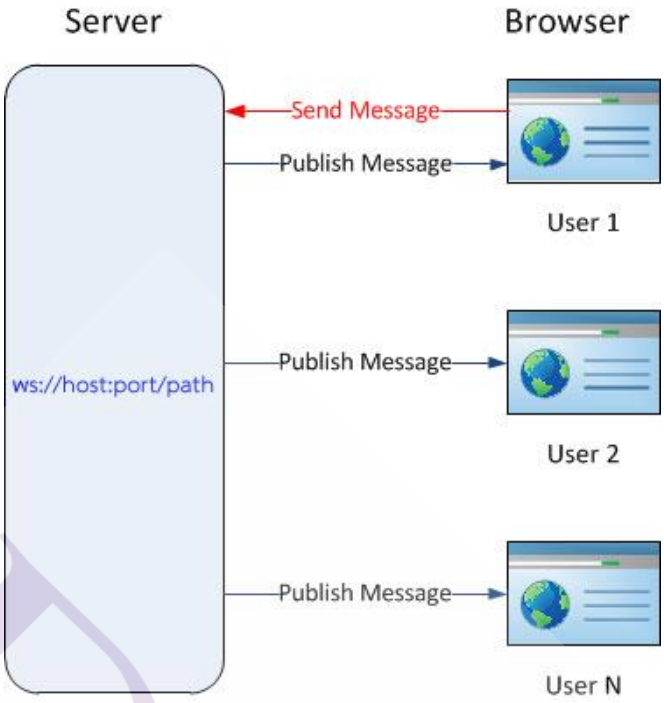
ภาพที่ 2.25 แสดง HTTP Code 101 Switching Protocols

2.4.1.2 Data transfer

เมื่อ Client และ Server เชื่อมต่อกันเรียบร้อยแล้ว จึงเข้าสู่ขั้นตอนของการรับส่งข้อมูลแบบ Full-duplex โดย Message ที่ส่งหากัน สามารถถูกแบ่งเป็น frame ย่อยๆ หลายๆ frame ได้ หาก Message มีขนาดใหญ่ และในแต่ละ frame จะประกอบด้วย header และ payload



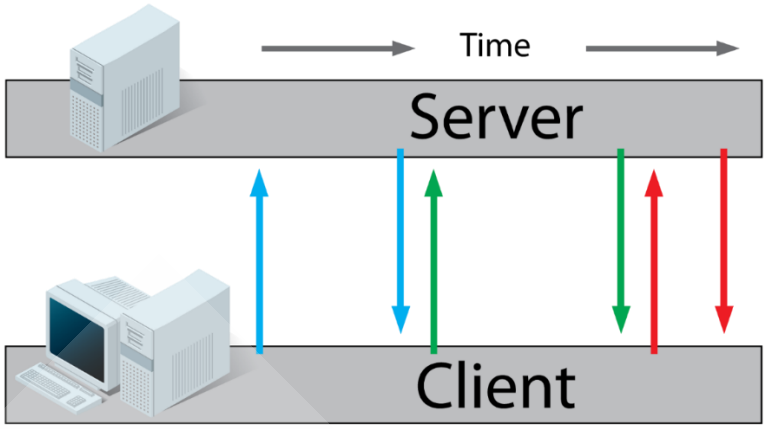
ภาพที่ 2.26 แสดง WebSocket Frame Data



ภาพที่ 2.27 แสดงการทำงานของ WebSocket
ที่มา: ThaiCreate.Com

ในฝั่งของ Server หน้าที่ของ WebSoket คือ จะรัน Service ที่ทำหน้าที่เป็น Protocol ที่ฝั่ง Server เมื่อมี Client ทำการเชื่อมต่อเข้ามา Service ดังกล่าว จะทำหน้าที่ติดตาม Client ไปอย่างใกล้ชิดเพื่อตรวจสอบว่า Client ยังทำการเชื่อมต่อและพร้อมที่จะ รับ-ส่ง ข้อมูลได้ตลอดเวลา โดยที่ไม่จำเป็นว่าฝั่ง Client จะมีการ Request มาหรือไม่ ฉะนั้นเมื่อฝั่ง Server มีข้อมูลมาใหม่ ตัว Service นี้ก็จะทำหน้าที่ Push ข้อมูลเพื่อส่งให้กับทุกๆ Client ที่เชื่อมต่ออยู่ในขณะนั้น ซึ่งตัว Service เองสามารถที่จะตรวจสอบได้ว่า มี Client อะไรบ้างที่กำลังเชื่อมต่ออยู่

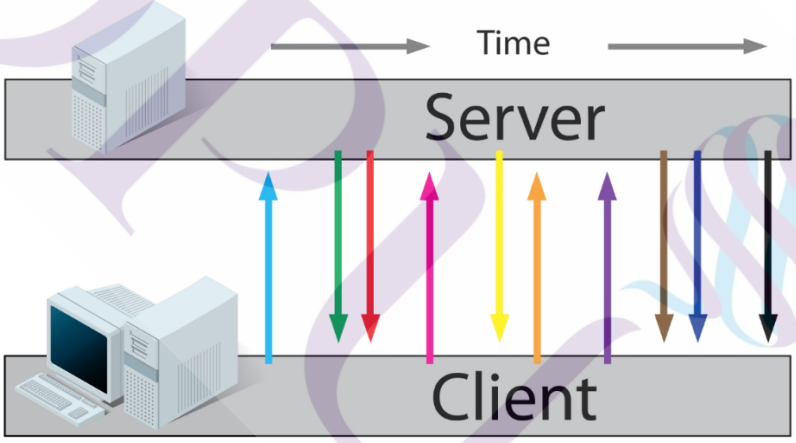
ซึ่งแตกต่างกับการ Request จาก Client โดยทั่วๆ ไป ที่เราจะต้องใช้ Client ทำการ Request ส่ง Post หรือ Get ไปยัง Server และ Server ก็จะทำการ Response ค่ากลับมา โดยที่ Serverไม่มีทางรู้ว่า Client ได้ทำการปิดการเชื่อมต่อไปแล้วหรือยัง Server จะรู้ได้อย่างเดียวก็ต่อเมื่อ Session ได้ Timeout ไปแล้ว



ภาพที่ 2.28 แสดงรูปแบบการเชื่อมต่อแบบปกติทั่วๆ ไป ที่มี Request และ Response กลับมา

ที่มา: ThaiCreate.Com

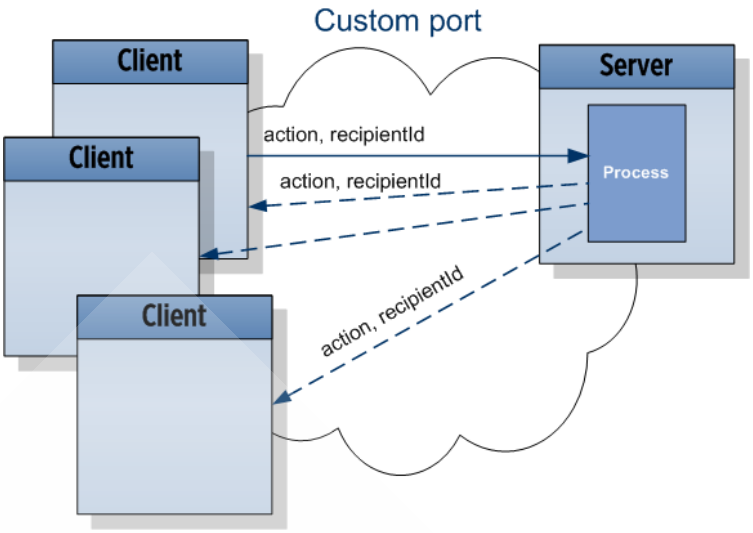
จากรูปแรกจะเห็นว่าสีที่แตกต่างกัน หมายถึง Client ทำการเชื่อมต่อแต่ละครั้งก็จะได้ Response กลับมา 1 ครั้งเท่านั้น



ภาพที่ 2.29 แสดงรูปแบบการทำงานของ WebSocket

ที่มา: ThaiCreate.Com

การทำงานของ WebSocket จะแตกต่างกันกับ HTTP ตรงที่ เมื่อมี Client ใดๆ ทำการเชื่อมต่อไปยัง Server ในฝั่งของ Server จะมี Service ที่ทำงานอยู่ตลอดเวลา ซึ่งจะส่ง Push หรือ Response กลับไปยังทุก ๆ Client ที่เชื่อมต่ออยู่ในขณะนั้น ซึ่ง Client ก็จะได้รับข้อมูลพร้อมกันหมด



ภาพที่ 2.30 แสดงลักษณะการรับ-ส่งข้อมูลระหว่าง Server และ Client ผ่าน WebSocket

ที่มา: ThaiCreate.Com

2.5 Global Positioning System (GPS)

Global Positioning System หรือ GPS ซึ่งถ้าแปลให้ตรงตัวแล้วคือ “ระบบกำหนดตำแหน่งบนพื้นโลก” ระบบนี้ได้พัฒนาขึ้นโดยกระทรวงกลาโหม ประเทศสหรัฐอเมริกา ซึ่งจัดทำโครงการ Global Positioning System มาตั้งแต่ปี พ.ศ. 2521 โดยอาศัยดาวเทียมและระบบคลื่นวิทยุนำร่องและรหัสที่ส่งมาจากดาวเทียม NAVSTAR จำนวน 24 ดวง โดยแบ่งเป็นชุด ชุดละ 4 ดวงโดยทำการโคจรอยู่รอบโลกวันละ 2 รอบ และมีตำแหน่งอยู่เหนือพื้นโลกที่ความสูง 20,200 กิโลเมตร

ภาพที่ 2.31 ลักษณะวงโคจรของดาวเทียม

ที่มา: Peter H. Dana, 1994

GPS เป็นระบบดาวเทียม NAVSTAR ที่ออกแบบและจัดสร้างโดยกองทัพสหรัฐอเมริกาเพื่อใช้ในการนำหน (Navigation) มีวัตถุประสงค์ในการออกแบบคือ

1. เพื่อให้มีผู้ใช้ประโยชน์ทั้งฝ่ายทหารและพลเรือนได้เป็นจำนวนมาก
2. เพื่อเครื่องรับและอุปกรณ์ใช้งานได้ง่ายและมีราคาต่ำ
3. ใช้ได้สะดวกไม่มีข้อจำกัด นั่นคือ ใช้ได้ตลอด 24 ชั่วโมง โดยไม่ขึ้นกับสภาพ ภูมิอากาศและสถานที่
4. ให้ความถูกต้องทางตำแหน่งตามเงื่อนไขที่ฝ่ายทหารกำหนด

ในปัจจุบันนอกจากระบบดาวเทียม GPS ของประเทศสหรัฐอเมริกาแล้ว ยังมีระบบดาวเทียมที่พัฒนาขึ้นอีกหลายระบบ ทำให้มีการเรียกระบบดาวเทียมทั้งหลายว่า ระบบดาวเทียมนำหนของโลก GNSS (Global NavigationSatellite Systems) ซึ่งระบบดังกล่าวในปัจจุบันเช่น

NAVSTAR – ของสหรัฐอเมริกา นิยมเรียกว่า GPS

GLONASS – ของประเทศรัสเซีย

Galileo – ของสหภาพยุโรป

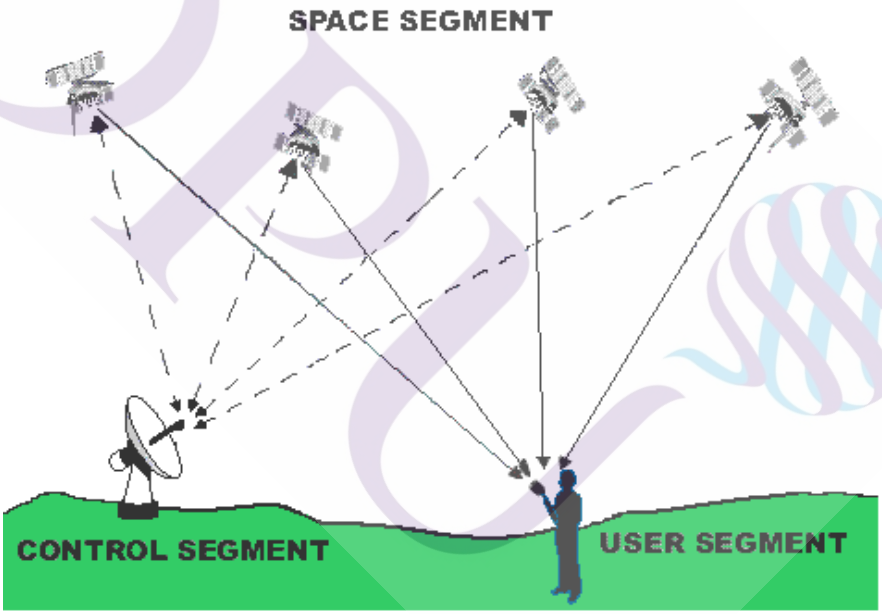
Beidou หรือ COMPASS – ของประเทศจีน

QZSS – ของประเทศญี่ปุ่น

IRNSS – ของประเทศอินเดีย

2.5.1 องค์ประกอบหลักของ GPS

ระบบกำหนดตำแหน่งบนโลก ประกอบด้วย 3 ส่วนหลัก คือ ส่วนอวกาศ (Space segment) ส่วนสถานีควบคุม (Control segment) และส่วนผู้ใช้ (User segment)



ภาพที่ 2.32 องค์ประกอบหลักของ GPS
ที่มา: Frederic G. Snider

1. ส่วนอวกาศ (Space segment) เป็นส่วนที่อยู่บนอวกาศ ประกอบด้วยดาวเทียม 24 ดวง โดยมี 21 ดวง แบ่งเป็น 6 วงโคจร วงโคจรละ 4 ดวง อยู่สูงจากพื้นดินประมาณ 20,200 กิโลเมตร ทำหน้าที่ส่งสัญญาณคลื่นวิทยุจากอวกาศ

หน้าที่สำคัญ ของดาวเทียม GPS มีดังนี้

รับข้อมูล วงโคจรที่ถูกต้องของดาวเทียม (Ephemeris Data) ที่ส่งมาจาก สถานีควบคุมดาวเทียมหลัก (Master Control Station) เพื่อส่งกระจายสัญญาณข้อมูลนี้ ลงไปยังพื้นโลก สำหรับ GPS Receiver ใช้ในการคำนวณ ระยะห่าง (Range) ระหว่างดาวเทียมดวงนั้น กับ ตัวเครื่อง GPS Receiver และตำแหน่งของดาวเทียมบนท้องฟ้า เพื่อใช้คำนวณหา ตำแหน่งพิกัด ของตัวเครื่อง GPS Receiver เอง

ส่งรหัส (Code) และข้อมูล Carrier Phase ไปกับคลื่นวิทยุ ลงไปยังพื้นโลก สำหรับ GPS Receiver ใช้ในการคำนวณ ระยะห่าง (Range) ระหว่างดาวเทียมดวงนั้น กับ ตัวเครื่อง GPS Receiver

ส่งข้อมูลตำแหน่งโดยประมาณของดาวเทียมทั้งหมด (Almanac Information) และข้อมูลสุขภาพ ของดาวเทียม ลงไปยังพื้นโลก สำหรับ GPS Receiver ใช้ในการกำหนดดาวเทียม ที่จะสามารถรับสัญญาณได้

2. ส่วนสถานีควบคุม (Control segment) ประกอบไปด้วยสถานีภาคพื้นดินที่ควบคุมระบบที่กระจายอยู่ตามส่วนต่าง ๆ ของโลก โดยแบ่งออกเป็นสถานีควบคุมหลัก และสถานีติดตามดาวเทียม 5 แห่ง ทำการรังวัดติดตามดาวเทียมตลอดเวลา สถานีรับส่งสัญญาณ 3 แห่ง

สถานีควบคุมภาคพื้นดิน MONITORING AND CONTROLLING ระบบ GPS ถูกควบคุมโดย กองทัพอากาศ สหรัฐอเมริกา จากสถานีควบคุมหลัก ในรัฐโคโลราโด ซึ่งจะคอยตรวจสอบดาวเทียมทุกดวงในระบบ ป้อนคำสั่งควบคุม และป้อนข้อมูล รวมทั้งให้ข่าวสารในการนำร่อง สถานีตรวจสอบภาคพื้นดิน ใช้สายอากาศภาคพื้นดิน ในการควบคุม ดาวเทียม GPS และส่งต่อข้อมูลให้แก่สถานี Master Control เพื่อกำหนดตำแหน่งพิกัดที่แน่นอน ของดาวเทียมแต่ละดวง และปรับปรุงความถูกต้อง ของข้อมูลอยู่ตลอดเวลา ถ้าดาวเทียมดวงใดเกิดความผิดปกติขึ้น สถานีควบคุมภาคพื้นดิน ก็จะทำการกำหนดสุขภาพ ดาวเทียมดวงนั้นเป็น “Un- healthy” เพื่อให้ GPS Receiver ทราบว่า ไม่ควรใช้ข้อมูล จากดาวเทียมดวงนี้ ซึ่งเครื่องรับ ก็จะทำการตรวจสอบได้ จากการตรวจสอบสถานะของดาวเทียม และเครื่องก็จะไม่ทำการ รับข้อมูล จากดาวเทียมดวงดังกล่าว แล้วใช้ดาวเทียมดวงอื่น ที่มีความเหมาะสม ในการคำนวณตำแหน่งพิกัดแทน ในบางครั้งดาวเทียมอาจถูกปิดใช้งานเพื่อทำการบำรุงรักษา หรืออาจจะถูกปิดเพื่อเปลี่ยนวงโคจร ตามความเหมาะสม

3. ส่วนผู้ใช้ (User segment) ประกอบด้วยเครื่องรับสัญญาณ หรือเครื่องรับจีพีเอส GPS ซึ่งมีหลายขนาด สามารถพกพาติดตัวหรือ จะติดไว้ในรถ เรือ เครื่องบินก็ได้

2.5.2 หลักการทำงานของ GPS

ดาวเทียม GPS (Navstar) ประกอบด้วยดาวเทียม 24 ดวง โดยแบ่งเป็น 6 รอบวงโคจร การจรจะเอียงทำมุมเอียง 55 องศากับเส้นศูนย์สูตร (Equator) ในลักษณะสานกันคล้าย ลูกตะกร้อแต่ละวงโคจรมีดาวเทียม 4 ดวง รัศมีวงโคจรจากพื้นโลก 20,162.81 กม. หรือ 12,600 ไมล์ ดาวเทียมแต่ละดวงใช้เวลาในการโคจรรอบโลก 12 ชั่วโมง

GPS ทำงานโดยการรับสัญญาณจากดาวเทียมแต่ละดวง โดยสัญญาณดาวเทียมนี้ประกอบไปด้วยข้อมูลที่ระบุตำแหน่งและเวลาขณะส่งสัญญาณ ตัวเครื่องรับสัญญาณ GPS จะต้องประมวลผลความแตกต่างของเวลาในการรับสัญญาณเทียบกับเวลาจริง ณ ปัจจุบันเพื่อแปรเป็นระยะทางระหว่างเครื่องรับสัญญาณกับดาวเทียมแต่ละดวง ซึ่งได้ระบุมีตำแหน่งของมันมากับสัญญาณดังกล่าวข้างต้น

เพื่อให้เกิดความแม่นยำในการค้นหาตำแหน่งด้วยดาวเทียม ต้องมีดาวเทียมอย่างน้อย 4 ดวง เพื่อบอกตำแหน่งบนผิวโลก ซึ่งระยะห่างจากดาวเทียมทั้ง 3 กับเครื่อง GPS (ที่จุดสีฟ้า) จะสามารถระบุตำแหน่งบนผิวโลกได้ หากพื้นโลกอยู่ในแนวระนาบแต่ในความเป็นจริงพื้นโลกมีความโค้งเนื่องจากสัณฐานของโลกมีลักษณะกลม ดังนั้นดาวเทียมดวงที่ 4 จะทำให้สามารถคำนวณเรื่องความสูงเพื่อทำให้ได้ตำแหน่งที่ถูกต้องมากขึ้น

นอกจากนี้ความแม่นยำของการระบุตำแหน่งนั้นขึ้นอยู่กับตำแหน่งของดาวเทียมแต่ละดวง กล่าวคือถ้าระยะห่างระหว่างดาวเทียมที่ใช้งานอยู่ห่างกันย่อมให้ค่าที่แม่นยำกว่าที่อยู่ใกล้กัน และยิ่งมีจำนวนดาวเทียมที่รับสัญญาณได้มากก็ยิ่งให้ความแม่นยำมากขึ้น ความแปรปรวนของชั้นบรรยากาศชั้นบรรยากาศประกอบด้วยประจุไฟฟ้า ความชื้น อุณหภูมิ และความหนาแน่นที่แปรปรวนตลอดเวลา คลื่นเมื่อตกกระทบ กับวัตถุต่างๆ จะเกิดการหักเหทำให้สัญญาณที่ได้อ่อนลง และสิ่งแวดล้อมในบริเวณรับสัญญาณเช่นมีการบดบังจากกระจก ละอองน้ำ ใบไม้ จะมีผลต่อค่าความถูกต้องของความแม่นยำเนื่องจากถ้าสัญญาณจากดาวเทียมมีการหักเหก็จะทำให้ค่าที่คำนวณได้จากเครื่องรับสัญญาณเพี้ยนไป และสุดท้ายก็คือประสิทธิภาพของเครื่องรับสัญญาณว่ามีความไวในการรับสัญญาณแค่ไหนและความเร็วในการประมวณผลด้วย

การวัดระยะห่างระหว่างดาวเทียมกับเครื่องรับทำได้โดยใช้สูตรคำนวณทางฟิสิกส์คือ ระยะทาง = ความเร็ว * ระยะเวลา วัดระยะเวลาที่คลื่นวิทยุส่งจากดาวเทียมมายังเครื่องรับ GPS คูณด้วยความเร็วของคลื่นวิทยุจะเท่ากับระยะทางที่เครื่องรับ อยู่ห่างจากดาวเทียม โดยเวลาที่วัดได้มาจากนาฬิกาของดาวเทียมที่มีความแม่นยำสูงมีความละเอียดถึงนาโนวินาที และมีการสอบทวนเสมอๆกับสถานีภาคพื้นดิน

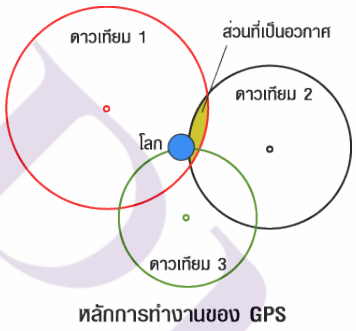
องค์ประกอบสุดท้ายก็คือตำแหน่งของดาวเทียมแต่ละดวงในขณะที่ส่งสัญญาณมาว่าอยู่ที่ใด (Almanac) มายังเครื่องรับ GPS โดยวงโคจรของดาวเทียมได้ถูกกำหนดไว้ล่วงหน้าแล้วเมื่อถูกส่งขึ้นสู่อวกาศ สถานีควบคุมจะคอยตรวจสอบการโคจรของดาวเทียมอยู่ตลอดเวลาเพื่อทวนสอบความถูกต้อง

โดยดาวเทียมทั้ง 3 ดวงจะส่งสัญญาณที่เหมือนกันมายังเครื่อง GPS โดยความเร็วแสง (186,000 ไมล์ต่อวินาที) แต่ระยะเวลาในการรับสัญญาณได้จากดาวเทียมแต่ละดวงนั้นจะไม่เท่ากันเนื่องจากระยะทางไม่เท่ากัน เช่น

ดาวเทียม 1 : ระยะเวลาในการส่งสัญญาณจากดาวเทียมดวงแรกถึงเครื่อง GPS คือ 0.10 วินาที ระยะทางระหว่างดาวเทียมกับ GPS คือ 18,600 ไมล์ (186,000 ไมล์ต่อวินาที X 0.10 วินาที = 18,600 ไมล์) ฉะนั้นตำแหน่งปัจจุบันก็จะสามารถเป็นจุดใดก็ได้ในวงกลมที่มีรัศมี 18,600 ไมล์ ซึ่งจะเห็นว่าดาวเทียมเพียงดวงเดียวยังไม่สามารถบอกตำแหน่งที่แน่นอนได้

ดาวเทียม 2 : ระยะเวลาในการส่งสัญญาณจากดาวเทียมดวงแรกถึงเครื่อง GPS คือ 0.08 วินาที ระยะทางระหว่างดาวเทียมกับ GPS คือ 13,200 ไมล์ (186,000 ไมล์ต่อวินาที X 0.08 วินาที = 13,200 ไมล์) ฉะนั้นตำแหน่งปัจจุบันก็จะสามารถเป็นจุดใดก็ได้ในจุด Intersect ระหว่างวงกลมจากดาวเทียมดวงแรกกับดาวเทียมดวงที่ 2

ดาวเทียม 3 : ระยะเวลาในการส่งสัญญาณจากดาวเทียมดวงแรกถึงเครื่อง GPS คือ 0.06 วินาที ระยะทางระหว่างดาวเทียมกับ GPS คือ 11,160 ไมล์ (186,000 ไมล์ต่อวินาที X 0.06 วินาที = 11,160 ไมล์) ฉะนั้นตำแหน่งปัจจุบันก็จะสามารถเป็นจุดใดก็ได้ในจุด Intersect ระหว่างวงกลมจากดาวเทียมทั้ง 3 ดวง



ภาพที่ 2.33 หลักการทำงานของ GPS
ที่มา: http://gpstrackingz.blogspot.com/2014/07/gps_7.html

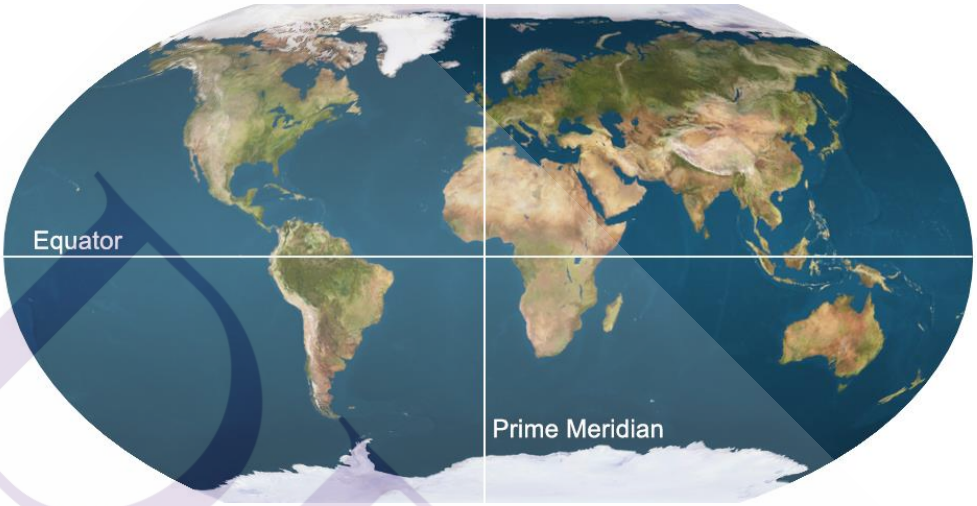
จะเห็นได้ว่าจะเหลือตำแหน่งอยู่ 2 จุดที่บริเวณวงกลมทั้ง 3 ตัดกันคือตำแหน่งที่อยู่ในอวกาศ ซึ่งแน่นอนว่าเราไม่สามารถไปอยู่ในอวกาศได้ตำแหน่งนี้จะถูกตัดทิ้งอัตโนมัติโดยเครื่อง GPS อีกตำแหน่งคือตำแหน่งบนพื้นโลกซึ่งเป็นตำแหน่งที่เรายืนถือเครื่อง GPS อยู่นั้นเอง ซึ่งความถูกต้องแม่นยำของตำแหน่งก็ขึ้นกับจำนวนดาวเทียมที่สามารถรับสัญญาณได้ในขณะนั้นหากมีมากกว่า 3 ดวงก็จะละเอียดมากขึ้น และก็ขึ้นกับเครื่อง GPS ด้วย หากเป็นเครื่องที่มีราคาแพง (ซึ่งมักใช้เฉพาะงาน) ก็จะมีความถูกต้องแม่นยำมากขึ้น

ข้อมูลตำแหน่งที่ได้มานั้น ยังสามารถใช้ร่วมกับโปรแกรมในเครื่อง GPS เพื่อบอกจุดบนแผนที่ และแสดงตำแหน่งของเราว่าอยู่จุดใดของแผนที่ได้อีกด้วย ทั้งนี้ก็ขึ้นกับข้อมูลแผนที่ที่ติดมากับเครื่องด้วยว่ามีความแม่นยำเพียงใด โดยแผนที่พื้นฐานจะไม่ได้ติดตั้งมากับเครื่อง GPS ทุกรุ่น ซึ่งอาจจะ

ต้องซื้อแยกจากตัวเครื่อง

2.4.3 ระบบพิกัดภูมิศาสตร์ (Geographic Coordinate System - GCS)

เป็นระบบที่ใช้กำหนดจุดต่างๆบนพื้นโลกด้วยตัวเลข 3 กลุ่ม คือ ความสูงจากระดับทะเลปานกลาง(elevation) และตัวเลขอีกสองกลุ่มเป็น พิกัดแนวระนาบเป็นค่าระยะเชิงมุม จากเส้นศูนย์สูตร ละติจูด (Latitude) และ ค่าระยะเชิงมุม จากเส้น Prime Meridian เป็นลองจิจูด (Longitude)



ภาพที่ 2.34 ระบบพิกัดภูมิศาสตร์ (Geographic Coordinate System - GCS)
ที่มา: http://www.global5thailand.com/thai/gps.htm

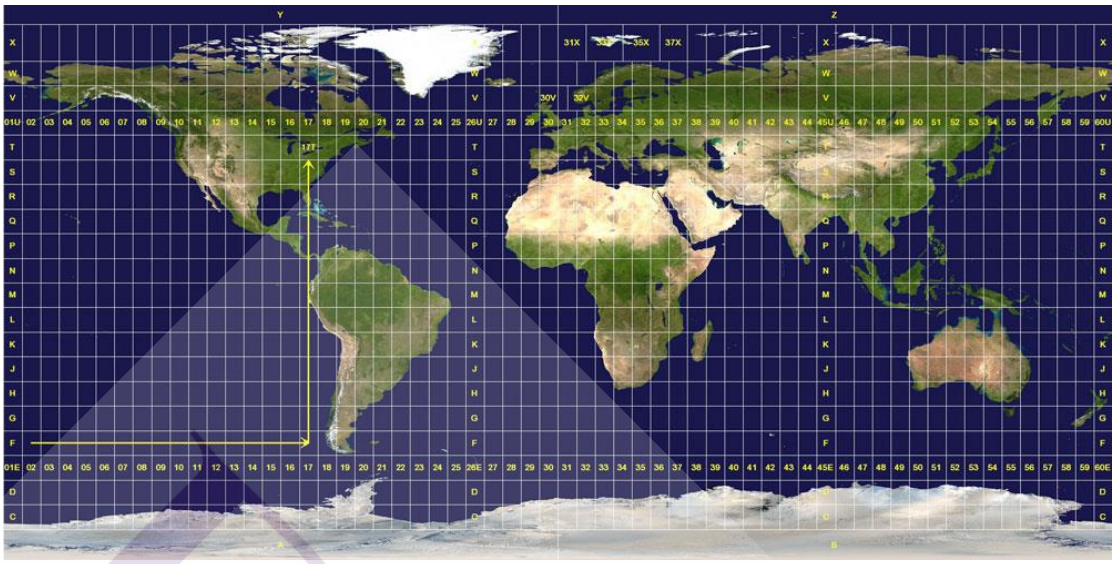
เส้นศูนย์สูตร (Equator) กำหนดขึ้นจากแนวระดับที่ตัดผ่านศูนย์กลางของโลกและตั้งฉากกับแกนหมุน ซึ่งแบ่งโลกออกเป็นซีกโลกเหนือและซีกโลกใต้ เส้นละติจูด (Latitude) จะเริ่มจากศูนย์สูตรเป็น 0 องศา ไปจนถึงขั้วโลกเป็น 90 องศาเหนือ และ ใต้

เส้นพรามเมอริเดียน (Prime Meridian) เป็นเส้นที่กำหนดขึ้นโดยลากผ่านเมือง Greenwich ประเทศอังกฤษไปจรดขั้วโลกเหนือและใต้ โดยจะแบ่งโลกออกเป็นซีกตะวันออกและตะวันตก เส้นลองจิจูด (Longitude)จะเริ่มจากเส้นพรามเมอริเดียนเป็น 0 องศา และไปสุดที่ 180 องศา ตะวันออก และ 180 องศาตะวันตก

ตารางที่ 2.7 รูปแบบพิกัด และการแปลงหน่วยจาก hddd mm ss.s ไปเป็น hddd.dddddd

รูปแบบพิกัดอาจกำหนดได้ดังนี้ hddd.dddddd คือ องศาและจุดทศนิยมขององศา (Decimal) hddd mm.mmm คือ องศา ลิปดา (arcminute) และทศนิยมของลิปดา hddd mm ss.s คือ องศา ลิปดา พิลิปดา (arcsecond) และทศนิยมของพิลิปดา การแปลงหน่วยจาก hddd mm ss.s ไปเป็น hddd.dddddd ทำได้ตามสูตร d + (m + (s / 60)) / 60) เช่น 1 องศา 23ลิปดา 45.6 พิลิปดา 1 23' 45.6" = 1+ (23 + (45.6 / 60))/60 = 1.396

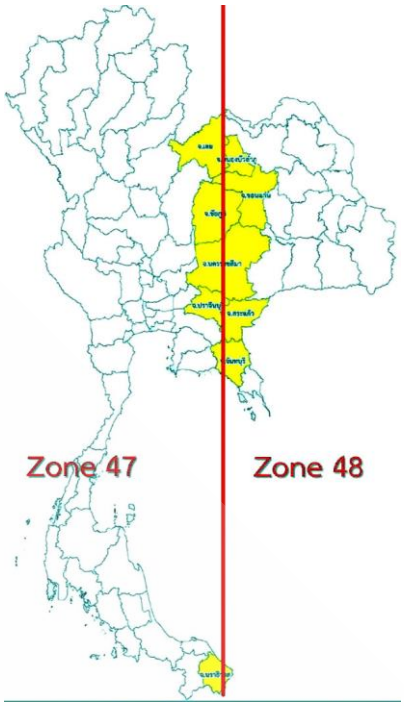
2.4.3 ระบบพิกัดกริด (Grid Coordinate) หรือ UTM (Universal Transverse Mercator)



ภาพที่ 2.35 ระบบพิกัดกริด (Grid Coordinate) หรือ UTM (Universal Transverse Mercator)
ที่มา: http://www.global5thailand.com/thai/gps.htm

ระบบนี้สร้างขึ้นจากการยืดส่วนโค้งทรงกลมของโลกให้เป็นทรงกระบอกแล้วแผ่ให้เป็นระนาบแบน แล้วจะใช้ตารางแบ่งโลกออกเป็นส่วนๆ โดยเริ่มจากขั้วโลกใต้ โดยแนวตั้งจากใต้ไปเหนือตามแนวเส้นละติจูดแทนด้วยตัวอักษร C - X ยกเว้น I และ O (ส่วน A,B เป็นขั้วโลกใต้ Y,Z เป็นขั้วโลกเหนือ) เริ่มจาก C ที่ ละติจูด 80 องศาใต้ (ช่วงละ 8 องศาไปจนถึงเส้นขนานละติจูด 72 องศาเหนือ และจากเส้นละติจูด 72-84 องศาเหนือ เป็นช่องละ 12 องศา) ทั้งหมด 20 ส่วน จนถึง X ที่ ละติจูด 84 องศาเหนือ และแบ่งแนวตะวันตกไปตะวันออก เขตละ 6 องศา รวมเป็น 60 เขต (Zone) แทนด้วยตัวเลขเริ่มจากเส้น Meridian 180 องศาตะวันตก เป็น 01 ไปจนถึง 174 องศาตะวันตก เป็น 60 ทำให้บนพื้นโลก แต่ละช่วงเป็นตารางพื้นที่สี่เหลี่ยมเรียกว่า เขตกริด (Grid zone) ซึ่งมีทั้งหมด 1,200 โซน แล้วจะใช้ชุดเลขพิกัด Northing (เหนือ ตามแนวตั้ง) และ Easting (ตะวันออก ตามแนวนอน) บอกตำแหน่ง โดยเลขของ Northing และ Easting จะมีหน่วยเป็นเมตร

สำหรับประเทศไทย อยู่ในโซน 47 และ 48 โดยเส้นแบ่งโซนนี้อยู่ระหว่างจังหวัด เลย หนองบัวลาภู (บางส่วน) ขอนแก่น ชัยภูมิ นครราชสีมา สระแก้ว ปราจีนบุรี จันทบุรี และ นราธิวาส (บางส่วน)



ภาพที่ 2.36 ตำแหน่งของประเทศไทยตามระบบพิกัดกริด (Grid Coordinate)
ที่มา: http://www.global5thailand.com/thai/gps.htm

2.5.4 การนำ GPS มาประยุกต์ใช้งาน

มีการนำ GPS มาใช้ในหลายวัตถุประสงค์ทางธุรกิจแต่ที่ใช้กันอย่างแพร่หลายมี 2 ประเภท คือ GPS Navigator (อุปกรณ์และระบบนำทาง) และ GPS Tracking System (อุปกรณ์และระบบติดตามรถ ยาพาหะนะหรือสัตว์เลี้ยง) โดยมีการทำงานที่แตกต่างกันดังนี้

1. GPS Navigator (อุปกรณ์และระบบนำทาง) เป็น GPS ที่เราใช้งานในรถยนต์ทั่วไปที่บอกแผนที่การเดินทางด้วยการป้อนข้อมูลของเป้าหมายลงไปเครื่องนำทาง GPS

2. GPS Tracking System (อุปกรณ์และระบบติดตามรถ ยาพาหะนะหรือสัตว์เลี้ยง) ซึ่งเป็น GPS ที่สามารถติดตามการเดินทาง และบอกพิกัดและตำแหน่งของ เครื่อง GPS ได้ด้วย การใช้ GPS ในการติดตามก็มีการใช้งานอย่างแพร่หลายเช่นใน รถบรรทุก รถยนต์สาธารณะ รถพยาบาล รถตำรวจ รถโรงเรียน เรือประมง ฯลฯ เพื่อ การบริหารกลุ่มรถ (Fleet Management), ความปลอดภัย, ติดตามและบันทึกพฤติกรรมการใช้งานยานพาหนะ, การกำหนดพื้นที่ปฏิบัติงาน เป็นต้น

ปัจจุบันนี้ได้มีการนำ GPS มาประยุกต์ใช้งานในรูปแบบต่างๆดังนี้

1. การกำหนดพิกัดของสถานที่ต่าง ๆ การทำแผนที่ โดยส่วนใหญ่นิยมใช้อุปกรณ์ที่สามารถพกพาไปได้ง่าย มีความทนทาน กันน้ำได้ สามารถใช้กับถ่านไฟฉายขนาดมาตรฐานได้ ดูรายละเอียด GPS สำหรับงานสำรวจ

2. การนำทาง ได้รับความนิยมอย่างกว้างขวางมีหลากหลายแบบและขนาด สามารถนำทางได้ทั้งภาพและเสียง ใช้ได้หลายภาษา บางแบบมีภาพเสมือนจริง ภาพสามมิติ และประสิทธิภาพอื่นๆ เพิ่มเติมเช่น multimedia Bluetooth hand free เป็นต้น

3. การวางแผนการใช้ประโยชน์ที่ดิน โครงข่ายหมุดดาวเทียม GPS ของกรมที่ดิน (DOLVRS)

4. การกำหนดจุดเพื่อบรรเทาสาธารณะภัย เช่น เสื้อกั๊กชูชีพที่มีเครื่องส่งสัญญาณจีพีเอส

5. การวางผังสำหรับการจัดส่งสินค้า

6. การนำไปใช้ประโยชน์ในขบวนการยุติธรรม เช่นการติดตามบุคคล

7. การติดตามการค้ายาเสพติด ฯลฯ ดูรายละเอียด GPS เพื่อการติดตาม

8. การนำไปใช้ประโยชน์ทางทหาร

9. การกีฬา เช่นใช้ในการฝึกฝนเพื่อวัดความเร็ว ระยะทาง แคลลอรี่ที่เผาผลาญ ดูรายละเอียดอุปกรณ์ GPS สำหรับกิจกรรมกลางแจ้ง หรือ ใช้ในสนามกอล์ฟเอคำนวณระยะจากจุดที่อยู่ถึงหลุม

10. การสันทนาการ เช่น กำหนดจุดตกปลา หาระยะเวลาที่เหมาะสมในการตกปลา การวัดความเร็ว ระยะทาง บันทึกเส้นทาง เครื่องบิน/รถบังคับวิทยุ ระบบการควบคุมหรือติดตามยานพาหนะ

11. การติดตามบุคคล เพื่อให้ทราบว่ายานพาหนะอยู่ที่ใด มีการเคลื่อนที่หรือไม่ มีการแจ้งเตือนให้กับผู้ติดตามเมื่อมีการเคลื่อนที่เร็วกว่าที่กำหนดหรือเคลื่อนที่ออกนอกพื้นที่หรือเข้าสู่พื้นที่ที่กำหนด นอกจากนั้นยังสามารถนำไปใช้ในการป้องกันการโจรกรรมและติดตามทรัพย์สินคืน

12. การนำข้อมูล GPS มาประกอบกับภาพถ่ายเพื่อการท่องเที่ยว การทำรายงานกิจกรรม เป็นต้น โดยจะต้องมีเครื่องรับสัญญาณ ดาวเทียมติดตั้งอยู่กับกล้องบางรุ่น หรือการใช้ GPS Data Logger ร่วมกับ Software

2.6 JavaScript Object Notation (JSON)

JavaScript Object Notation หรือ JSON เป็นรูปแบบสำหรับแลกเปลี่ยนข้อมูลคอมพิวเตอร์ที่อยู่ในรูปข้อความธรรมดา (Plain Text) ที่ทั้งมนุษย์และโปรแกรมคอมพิวเตอร์สามารถอ่านเข้าใจได้ รูปแบบฟอร์แมตของ JSON ถูกสร้างขึ้นจากชุดข้อมูลในภาษาจาวาสคริป (JavaScript) ซึ่งใช้วงเล็บก้ามปู ([]) แทนแถวลำดับ (Array) และใช้วงเล็บปีกกา ({}) แทนแถวลำดับแบบจับคู่ (Associative Array) แต่ละสมาชิกคั่นด้วยเครื่องหมายจุลภาค (,) และแต่ละชื่อของข้อมูลกับค่าของข้อมูลจะคั่นด้วยเครื่องหมายทวิภาค (:)

มาตรฐานของฟอร์แมต JSON คือ RFC 4627 มี Internet media type เป็น application/json และมีนามสกุลของไฟล์เป็น .json

JSON นั้นใช้ความสัมพันธ์ของภาษาจาวาสคริปต์ แต่ไม่ถูกมองว่าเป็นภาษาโปรแกรม กลับถูกมองว่าเป็นภาษาในการแลกเปลี่ยนข้อมูลมากกว่า ปัจจุบัน JSON นิยมใช้ในเว็บแอปพลิเคชัน โดย JSON เป็นฟอร์แมตทางเลือกในการส่งข้อมูล นอกเหนือไปจาก XML ซึ่งนิยมใช้กันอยู่แต่เดิม สาเหตุที่ JSON เริ่มได้รับความนิยม เป็นเพราะกระชับและเข้าใจง่ายกว่า และยังสามารถนำ JSON ไปประยุกต์กับการรับส่งข้อมูลในรูปแบบอื่น ๆ ได้ เช่นการจับเก็บข้อมูลในรูปแบบของ สายอักขระในข้อความหรือการรับส่งผ่านตัวให้บริการเว็บไซต์ (Web Service) ก็สามารถทำได้เช่นเดียวกัน

ตารางที่ 2.8 ตัวอย่างข้อมูลในรูปแบบ JSON

```
{
  "firstName": "John",
  "lastName": "Smith",
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021",
   },
  "phoneNumbers": [
    "212 555-1234",
    "646 555-4567"
  ]
}
```

2.7 งานวิจัยที่เกี่ยวข้อง

งานวิจัยที่เกี่ยวข้องที่ได้นำมาอ้างอิงเกี่ยวกับเว็บเซอร์วิสเทคโนโลยีและการติดต่อส่งข้อมูลผ่านเครือข่ายแบบเรียลไทม์มีดังนี้

2.7.1 Web Services Based On SOAP and REST Principles

Snehal Mumbaikar และ Puja Padiya (2013) ได้ทำการวิจัยเปรียบเทียบประสิทธิภาพการทำงานระหว่าง SOAP และ RESTful เว็บเซอร์วิส โดยการทดสองแบ่งเป็นสองส่วนคือ

1. การทดลองเรียกใช้งานเว็บเซอร์วิสทั้งสองแบบที่ทำงานให้ผลลัพธ์เหมือนกัน และใช้ข้อมูลประเภท Float และ String เป็น parameter ในการเรียกใช้เซอร์วิส โดยเรียกจากไคลเอนท์ซึ่งเป็น emulator ประเภทอุปกรณ์มือถือ และใช้หลักเกณฑ์ในการเปรียบเทียบคือ ขนาดของข้อมูลที่รับส่งระหว่างกัน และ เวลารวมที่ใช้ในการตอบกลับ โดยผลลัพธ์ที่ได้สามารถสรุปได้คือ

ขนาดของข้อมูลจาก RESTful เว็บเซอร์วิส ทั้งจากข้อมูลประเภท Float และ String น้อยกว่าข้อมูลจาก SOAP เว็บเซอร์วิส 9-10 เท่า

เวลาที่ใช้ในการประมวลผลและส่งข้อมูลตอบกลับมายังไคลเอนท์สำหรับ RESTful เว็บเซอร์วิส น้อยกว่าเวลาตอบกลับของ SOAP เว็บเซอร์วิส 5-6 เท่า

2. การทดลองโดยใช้การประชุมแบบสื่อผสม (multimedia conference) มาทำการทดสอบเพื่อใช้ในการพิจารณาประสิทธิภาพการทำงานของเว็บเซอร์วิสทั้งสองแบบ ซึ่งได้แก่ การประชุมผ่านเสียงและวีดีโอ การเรียนการสอนระยะไกล รวมถึงเกมส์ออนไลน์ ผลการทดลองคือ

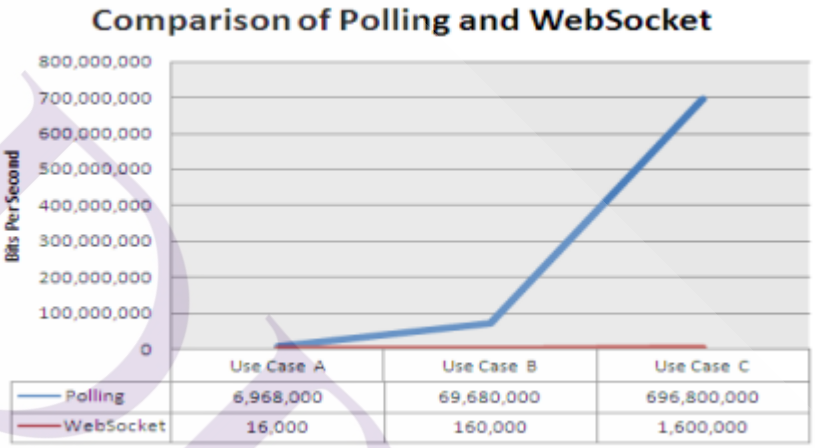
RESTful เว็บเซอร์วิส ใช้เวลาในการส่งข้อมูลน้อยกว่า SOAP เว็บเซอร์วิส 3-5 เท่า

ขนาดข้อมูลที่ส่งไปมาหากันในเครือข่าย RESTful เว็บเซอร์วิส มีขนาดข้อมูลน้อยกว่า SOAP เว็บเซอร์วิสเกือบจะ 3 เท่า

ทำให้สามารถสรุปผลได้ว่าสิทธิภาพการทำงานของ RESTful เว็บเซอร์วิสดีกว่า SOAP เว็บเซอร์วิส เนื่องจากมีการทำงานที่ง่ายและยืดหยุ่นว่า และใช้ overhead ในการส่งข้อมูลที่น้อยกว่า SOAP เว็บเซอร์วิสมาก

2.7.2 Real Time Data Communication over Full Duplex Network Using Websocket

Shruti M. Rakhunde (2014) ได้ทำการวิจัยและพูดถึงหลายๆวิธีการที่ใช้ในการส่งข้อมูลแบบเรียลไทม์และปัญหาที่เกิดขึ้นของการส่งข้อมูลในแต่ละแบบ ซึ่งได้แก่การส่งข้อมูลแบบ HTTP polling และ long polling และข้อดีของ WebSocket ที่สามารถเข้ามาช่วยแก้ปัญหาเหล่านั้น โดยทำการวิจัยการเปรียบเทียบประสิทธิภาพการทำงานของแต่ละวิธีกับ WebSocket กับหลายๆ use case โดยใช้หลักเกณฑ์อย่างเช่น network overhead และ latency หรือความเร็วในการรับส่งข้อมูลเป็นตัววัด ซึ่งได้ผลลัพธ์สรุปได้ว่า WebSocket ให้ประสิทธิภาพการทำงานที่ดีกว่ามาก



ภาพที่ 2.37 กราฟแสดงการเปรียบเทียบประสิทธิภาพการทำงานของ Polling และ WebSocket

บทที่ 3
วิธีการดำเนินการและเครื่องมือ

ในบทนี้จะกล่าวถึงกระบวนการการพัฒนาเว็บเซอร์วิสสำหรับระบบบริการขนส่งของและติดตามการขนส่งแบบเรียลไทม์ ด้วยเทคโนโลยีการสื่อสารผ่านเครือข่ายที่เกี่ยวข้อง เช่น REST API, WebSocket เทคโนโลยี GSP และ รูปแบบข้อมูล JSON รวมถึงการนำเครื่องมือในการจัดทำเอกสาร OpenAPI Specsification มาประยุกต์ใช้เพื่อให้เกิดประโยชน์ในการพัฒนาระบบโดยมีวิธีการดำเนินงานและเครื่องมือ ดังนี้

3.1 ศึกษาปัญหาและความต้องการของระบบ

3.1.1 วิเคราะห์ความต้องการของผู้ใช้งาน

ผู้วิจัยได้สำรวจข้อมูลความต้องการของระบบจากผู้ที่เกี่ยวข้องโดยวิธีสัมภาษณ์ จากนั้นนำข้อมูลที่ได้มาวิเคราะห์และสรุปความต้องการเพื่อใช้ในการพัฒนาระบบ และรวบรวมความต้องการจากผู้ใช้งานสรุปผลได้ว่า มีผู้ใช้งาน 3 กลุ่มใหญ่ๆ คือผู้ใช้งานหรือลูกค้าทั่วไป(User) พนักงานส่งของ (Messenger) และผู้ดูแลระบบ(Administrator) ซึ่งทำให้ความต้องการที่มีต่อระบบถูกแบ่งออกเป็น 3 กลุ่มตามประเภทผู้ใช้งาน คือ

สำหรับผู้ใช้งานทั่วไป (User)

1. การสมัครสมาชิกและล็อกอินเข้าสู่ระบบ
   - 1.1 สามารถสมัครสมาชิกผ่านแบบฟอร์มบนแอปพลิเคชัน
   - 1.2 สามารถล็อกอินเข้าสู่ระบบ
     - 1.2.1 ล็อกอินด้วยหมายเลขโทรศัพท์
     - 1.2.2 ล็อกอินด้วยบัญชี Facebook
     - 1.2.3 ล็อกอินด้วยบัญชี Instagram
     - 1.2.4 ล็อกอินด้วยบัญชี Google
   - 1.3 สามารถส่ง One Time Password (OTP) เพื่อใช้ยืนยันในการสมัครสมาชิก
2. การเพิ่มลบแก้ไขข้อมูลส่วนตัว
   - 2.1 สามารถ แก้ไข ชื่อ E-mail และ เบอร์โทรศัพท์ ของผู้ใช้บริการ โดยมีการส่ง One Time Password (OTP) เพื่อยืนยันก่อนการแก้ไขข้อมูล

2.2 สามารถเพิ่ม แก้ไข ข้อมูลบริษัทเพื่อใช้ออกใบกำกับภาษี

2.3 สามารถเพิ่ม แก้ไข ข้อมูลบัตรเครดิต

2.4 สามารถเพิ่ม ลบ คนขับที่ชื่นชอบ (Favorite Rider)

3. การสร้างและจัดการคำสั่งงาน

3.1 สามารถดูประวัติการสั่งงานย้อนหลัง

3.2 สามารถสร้างงานใหม่ได้ทันที หรือ จองล่วงหน้าได้ไม่เกิน 7 วัน

3.3 สามารถเพิ่ม จุดส่งได้มากกว่า 1 จุด แต่ไม่เกิน 11 จุด

3.4 สามารถเลือกบริการเสริมเพิ่มเติม เช่น เก็บเช็ค,วางบิล ธุรกรรมทางการเงิน ฝากส่งไปรษณีย์

3.5 สามารถเลือกบริการ ไป และ กลับ

3.6 สามารถเปิด และ ปิด การขอรับบริการ รูปภาพการปิดงาน

3.7 สามารถเลือกรับหรือไม่รับใบกับกำภาษี

3.8 สามารถดูสิทธิบริการส่งเสริมการขาย และ กดใช้สิทธิได้

3.9 สามารถเลือกช่องทางการชำระเงิน

3.9.1 ชำระเงินสด ต้นทาง

3.9.2 ชำระเงินสด ปลายทาง

3.9.3 ชำระด้วยบัตรเครดิต

3.9.4 โอนเงินผ่านธนาคาร

3.10 สามารถแจ้งเตือนงานที่จองล่วงหน้าก่อนถึงกำหนด

3.11 หากต้องการขอยกเลิกงาน สามารถยกเลิกการจองล่วงหน้าได้ (ทุกกรณี) แต่ต้องเลือกใส่เหตุผลในการยกเลิก

3.12 สามารถค้นหาพนักงาน ได้จากตำแหน่งใกล้เคียง

3.13 สามารถแจ้งเตือนผู้ใช้งานหากไม่มีพนักงานรับงาน ผู้ใช้สามารถยกเลิกคำสั่งงานหรือลองค้นหาพนักงานอยู่อยู่ใกล้เคียงใหม่ได้

3.14 สามารถยกเลิกการค้นหาพนักงาน

3.15 แสดงข้อมูลพนักงานที่ค้นหาเจอ

3.16 สามารถติดตามสถานะการจัดส่งของพนักงาน

3.17 ดูรายการการจัดส่ง

3.18 สามารถดูการชำระเงิน

3.19 สามารถดูราคาในการจัดส่ง

3.20 สามารถดูแผนที่เพื่อติดตามตำแหน่งของพนักงาน

3.21 สามารถดูเวลาที่ใช้ในการจัดส่ง

3.22 สามารถดูรายละเอียดติดต่อพนักงาน

3.23 สามารถดูรายละเอียดในการนำส่งทั้งหมด

3.24 สามารถให้คะแนนความพึงพอใจต่อพนักงาน

4. สามารถ ดูรายละเอียด คำถามที่พบบ่อย (FAQ)

5. สามารถดูรายละเอียด ข่าวสารและโปรโมชั่น

สำหรับพนักงานส่งของ (Messenger)

1. การสมัครสมาชิกและล็อกอินเข้าสู่ระบบ

1.1 สามารถสมัครสมาชิกผ่านแบบฟอร์มบนแอปพลิเคชัน

1.2 สามารถล็อกอินเข้าสู่ระบบ

1.2.1 ล็อกอินด้วยหมายเลขโทรศัพท์

1.2.2 ล็อกอินด้วยบัญชี Facebook

1.2.3 ล็อกอินด้วยบัญชี Instagram

1.2.4 ล็อกอินด้วยบัญชี Google

1.3 สามารถส่ง One Time Password (OTP) เพื่อใช้ยืนยันในการสมัครสมาชิก

2. แสดง เพิ่ม ลบ แก้ไขข้อมูลส่วนตัว

2.1 สามารถ แก้ไข ชื่อ E-mail และ เบอร์โทรศัพท์ ของผู้ใช้บริการ โดยมีการส่ง One Time Password (OTP) เพื่อยืนยันก่อนการแก้ไขข้อมูล

2.2 สามารถเพิ่มและแก้ไขข้อมูลส่วนตัว

2.3 สามารถแสดงรายละเอียดคะแนนการรับงาน

2.4 สามารถเพิ่มแก้ไขข้อมูลบัญชีธนาคาร

3. การจัดการคำสั่งงาน

3.1 สามารถ เปิด ปิด สถานะในการรับงาน

3.2 มีการแจ้งเตือนเมื่อมีงานใหม่ที่ตรงตามเงื่อนไข

3.3 ดูรายละเอียดและกดรับงาน

3.4 สามารถแสดงรายละเอียดของจุดรับ จุดส่งตามลำดับ

3.5 สามารถแสดงรายได้ในการรับงาน

3.6 แสดงรายละเอียดงานในแต่ละจุดว่าจะต้องทำอะไร เช่น จ่ายบิล ซื้อของ

3.7 มีระบบนำทางไปยังจุดหมายโดย Google Maps

3.8 แจ้งเตือนไปยังผู้ใช้บริการเมื่อนำส่งเสร็จในแต่ละจุด และ เมื่อนำส่งเสร็จสิ้นทั้งหมดจึงจะสามารถปิดงานได้

3.9 สามารถถ่ายภาพ หรือ อัพโหลดรูปภาพการปิดงาน

3.10 สามารถรองรับการเซ็น ลายเซ็นเพื่อใช้สำหรับการปิดงาน

3.11 แสดงประวัติงาน รายวัน รายสัปดาห์ และ รายเดือน

3.12 แสดงรายละเอียดงาน วัน เวลา สถานที่ รายได้

3.13 แสดงยอดเครดิตคงเหลือ

3.14 แสดงรายละเอียดการเติมเครดิต

3.15 มีการแจ้งเตือนพนักงานก่อนงานที่รับไว้จะถึงเวลาเริ่มงาน ทุกวันช่วงเวลา 8 นาฬิกา และ 18 นาฬิกา

3.16 มีการแจ้งเตือนพนักงานก่อนงานที่รับไว้จะถึงเวลาเริ่มงาน 5 นาที

สำหรับผู้ดูแลระบบ (Admin)

1. จัดการข้อมูลสมาชิก

1.1 ดูรายละเอียดข้อมูลส่วนตัวของสมาชิก

1.2 สามารถดูประวัติงานย้อนหลังของสมาชิก

1.3 สามารถแสดงรายละเอียด รายจ่าย ของผู้ใช้บริการ

2. ดูรายละเอียดข้อมูลส่วนตัวของพนักงาน

2.1 รายละเอียดข้อมูลส่วนตัวของพนักงาน

2.2 สามารถดูประวัติงานย้อนหลังของพนักงาน

2.3 สามารถแสดงรายละเอียด รายได้ ของพนักงาน

2.4 สามารถแสดงเครดิต ของพนักงาน

2.5 แจ้งเตือนเมื่อเอกสารหมดอายุ (ใบขับขี่ บัตรประชาชน สำเนาทะเบียนรถ) สำเนาทะเบียนรถไม่สามารถแก้ไขได้ ต้องอัพโหลดเอกสารใหม่แล้วผู้ดูแลระบบทำการตรวจเอกสารและทำการอนุมัติจึงจะแก้ไขข้อมูลในระบบได้

3. การติดตามการทำงาน

3.1 สามารถดูตำแหน่งที่อยู่ของผู้ขับขี่ที่กำลังออนไลน์

3.2 สามารถดูตำแหน่งพนักงานที่กำลังออนไลน์

3.3 มีระบบติดตามการนำส่งงาน

3.4 สามารถส่งการแจ้งเตือนงานไปยังพนักงาน และ ผู้ใช้บริการได้อัตโนมัติเมื่อถึงระยะเวลาที่ตั้งไว้

4. สามารถดูข้อมูลงานในระบบ

4.1 สามารถดูรายละเอียดงานที่ได้รับการจองผ่านระบบ

5. สามารถเพิ่ม ลบ แก้ไข บริการเสริม

6. สามารถเพิ่ม ลบ แก้ไขคำถามที่พบบ่อย

7. สามารถเพิ่ม ลบ แก้ไข รายละเอียดการส่งเสริมการขาย

8. มีระบบอนุมัติในการเติมเครดิตของพนักงาน

9. สามารถดึงข้อมูลบริษัทเพื่อออกใบกำกับภาษี

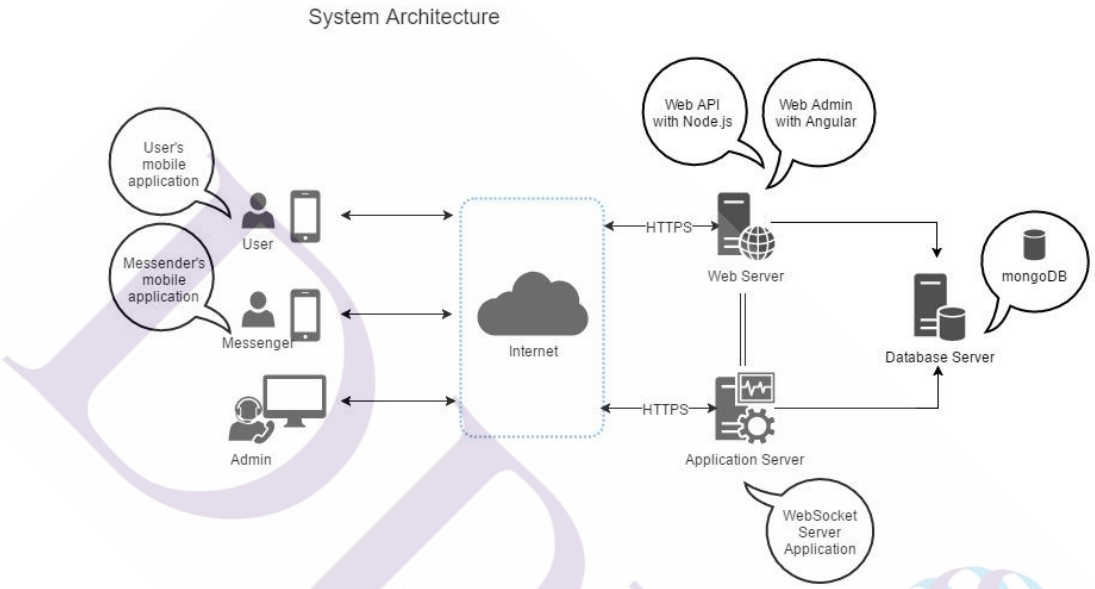
10. สามารถกำหนดขอบเขตพื้นที่ให้บริการ

ซึ่งผู้วิจัยได้นำข้อมูลเหล่านี้ ไปใช้ในการวิเคราะห์เพื่อเป็นแนวทางในการพัฒนาระบบบริการรับส่งของและติดตามในต่อไป

3.2 วิเคราะห์และออกแบบระบบ

3.2.1 สถาปัตยกรรมของระบบ (System Architecture)

เมื่อนำข้อมูลการทำงานของระบบและข้อมูลความต้องการของผู้ใช้งานมาวิเคราะห์ให้มีความสอดคล้องกันและได้สรุปผลการออกแบบการทำงานของระบบ ซึ่งสามารถเขียนออกมาเป็นสถาปัตยกรรมของระบบโดยรวมได้แบบดังภาพที่ 3.1



ภาพที่ 3.1 สถาปัตยกรรมโดยรวมของระบบ

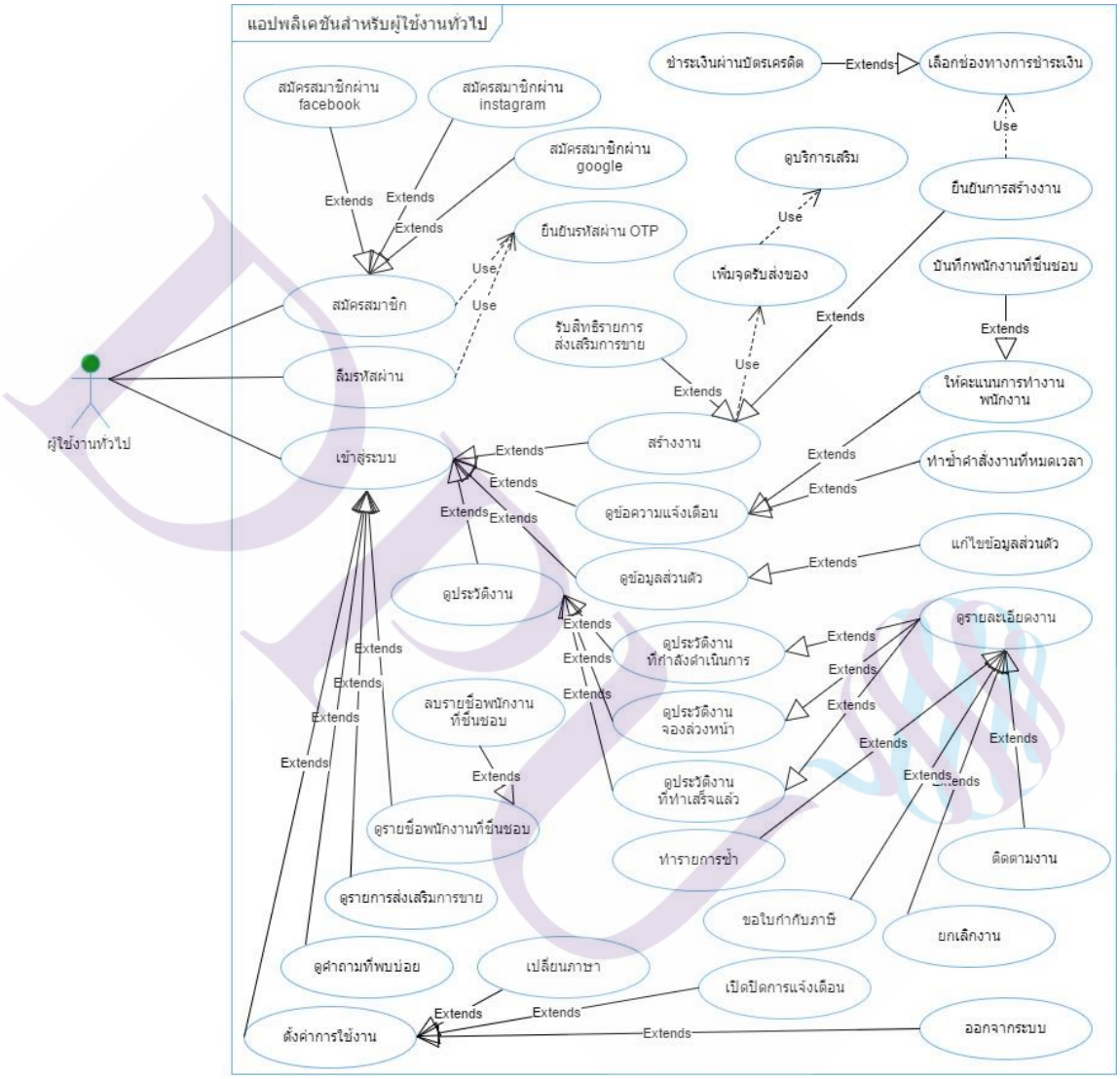
ภาพรวมสถาปัตยกรรมของระบบบริการขนส่งของและติดตามการขนส่งแบบเรียลไทม์มีองค์ประกอบดังนี้

1. Web Server ซึ่ง run เว็บเซอร์วิสที่เรียกใช้งานภายในระบบและเว็บแอปพลิเคชันสำหรับผู้ดูแลระบบ
2. Web Server run WebSocket Application เพื่อการเชื่อมต่อและส่งข้อมูลแบบเรียลไทม์
3. Database Server สำหรับจัดการเก็บข้อมูลที่ใช้ภายในระบบ
4. Mobile Application สำหรับผู้ใช้งานทั่วไป สำหรับส่งคำสั่งงานเข้ามายังระบบ
5. Mobile Application สำหรับพนักงานส่งของ สำหรับรับคำสั่งงานที่ถูกสร้างโดยผู้ใช้งานทั่วไป

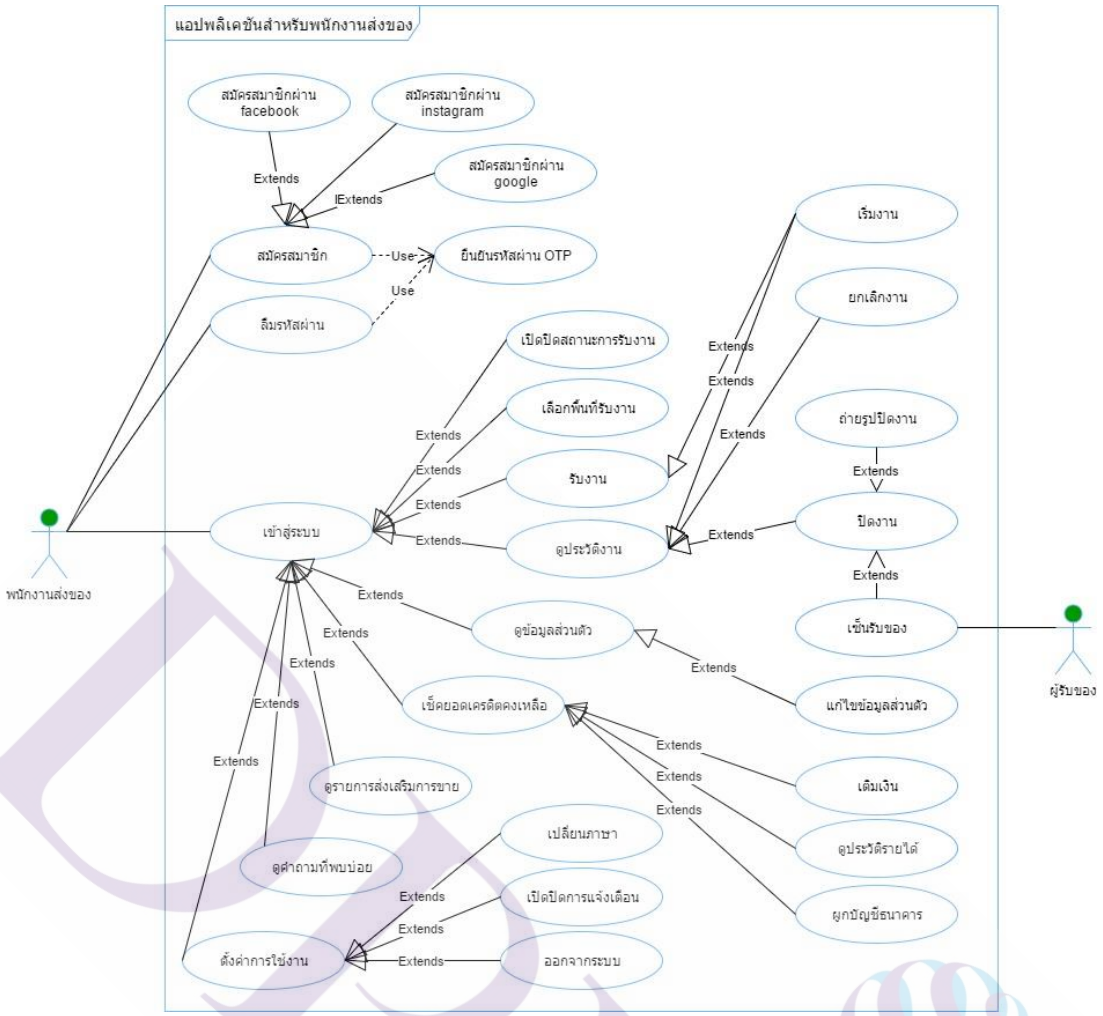
3.2.2 การออกแบบตาม Use Case การใช้งานของผู้ใช้ทั้ง 4 กลุ่ม

3.2.2.1 จากข้อมูลความต้องการของผู้ใช้ที่ได้ สามารถนำมาเขียนในรูปแบบ Use Case Diagram สำหรับแอปพลิเคชันของผู้ใช้ทั่วไป แอปพลิเคชันสำหรับพนักงานส่งของ และเว็บไซต์

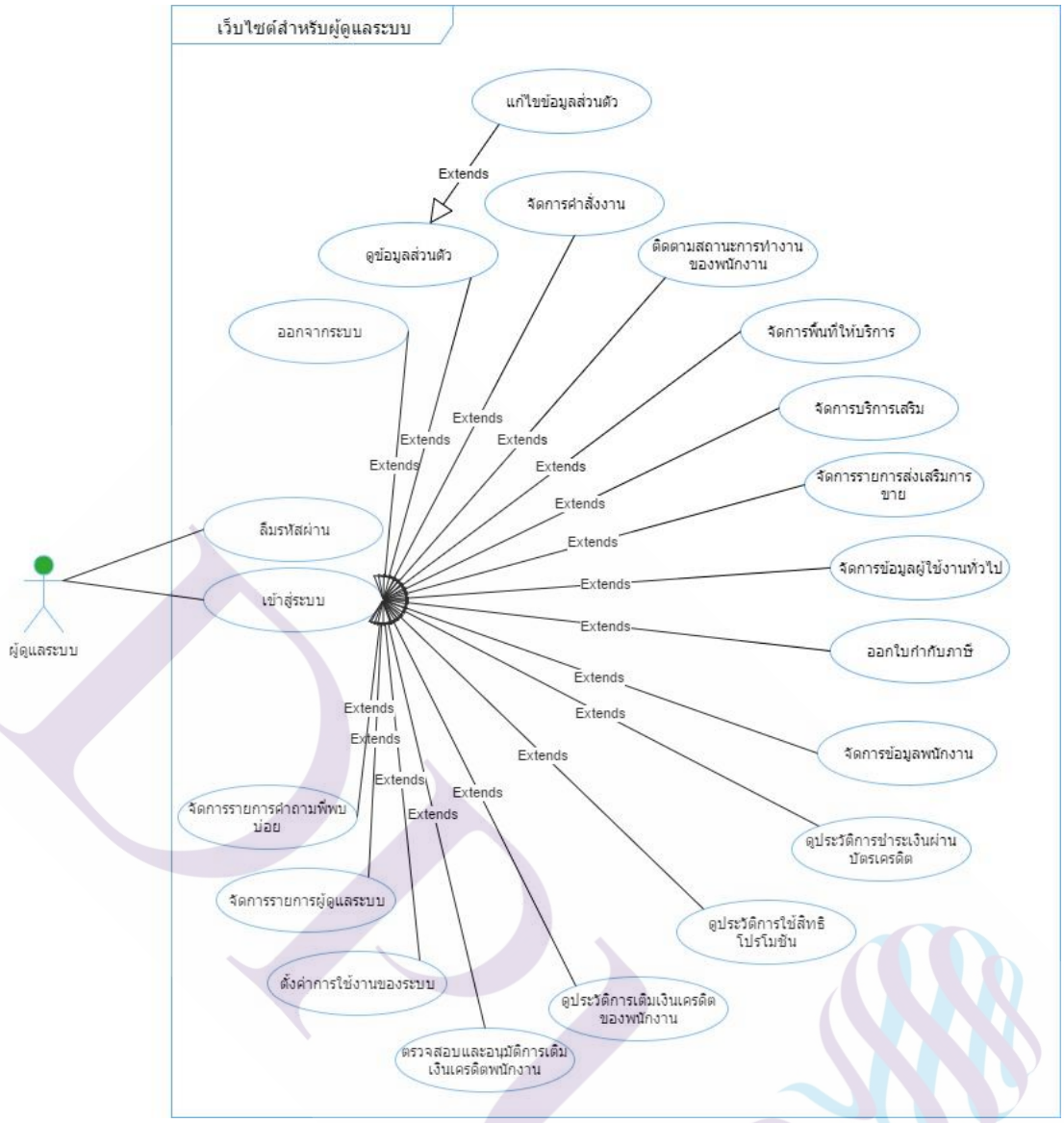
สำหรับผู้ดูแลระบบ แต่ทั้งนี้จากการวิเคราะห์การทำงานภายในระบบ นอกจากกลุ่มผู้ใช้หลักที่มี 3 กลุ่ม ในการทำงานภายในตัวระบบเองก็จำเป็นจะต้องมี Process ภายในเพื่อจัดการให้ระบบสามารถทำงานได้อย่างอัตโนมัติ จึงทำให้มีผู้ใช้งานเพิ่มขึ้นมาเป็น Actor อีกหนึ่งในระบบนั้นคือตัว Process ภายในระบบเอง ทำให้สามารถแสดงเป็น Use Case Diagram ได้ดังภาพที่ 3.2 ภาพที่ 3.3 ภาพที่ 3.4 และภาพที่ 3.5 ตามลำดับ



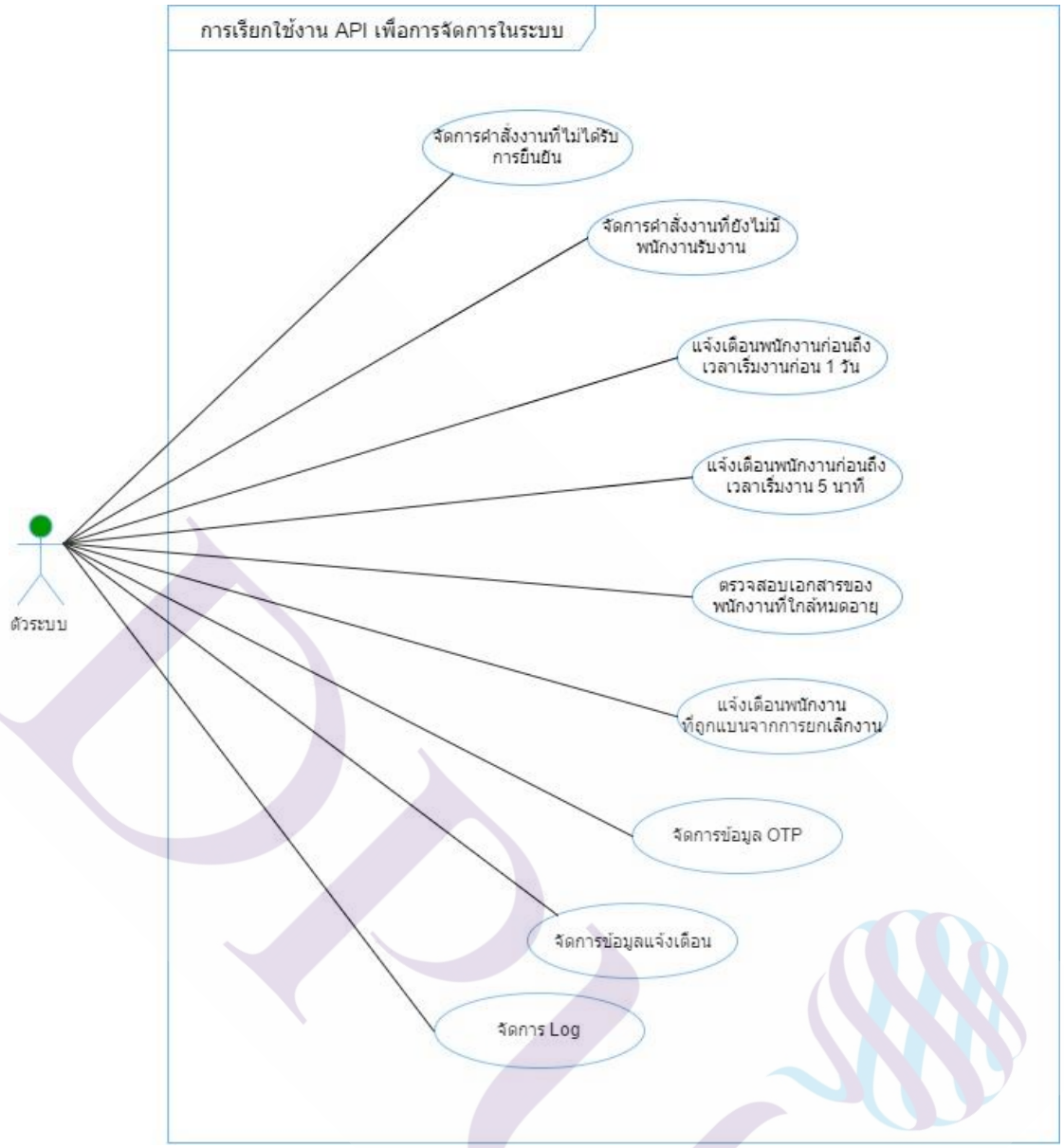
ภาพที่ 3.2 Use Case Diagram สำหรับแอปพลิเคชันสำหรับผู้ใช้ทั่วไป



ภาพที่ 3.3 Use Case Diagram สำหรับแอปพลิเคชันสำหรับพนักงานส่งของ



ภาพที่ 3.4 Use Case Diagram สำหรับเว็บไซต์ผู้ดูแลระบบ



ภาพที่ 3.5 Use Case Diagram สำหรับ Process ที่เรียกใช้งานภายในระบบเอง

3.2.2.2 Use Case Description

สามารถสรุปเป็นรายการ Use Case ได้ตามในตารางที่ 3.1 และสามารถดูรายละเอียดของ Use Case ที่สำคัญๆได้ตามหมายเลขตารางที่ระบุไว้

ตารางที่ 3.1 สรุป Use Case ของระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์

No.	Use Case ID	Use Case Name	Actor	ตาราง
1.	U01000	ผู้ใช้ลืมรหัสผ่าน	ผู้ใช้ทั่วไป	
2.	U02000	ผู้ใช้สมัครสมาชิก	ผู้ใช้ทั่วไป	
3.	U02100	ผู้ใช้สมัครสมาชิกด้วยบัญชี facebook	ผู้ใช้ทั่วไป	
4.	U02200	ผู้ใช้สมัครสมาชิกด้วยบัญชี instagram	ผู้ใช้ทั่วไป	
5.	U02300	ผู้ใช้สมัครสมาชิกด้วยบัญชี google	ผู้ใช้ทั่วไป	
6.	U03000	ผู้ใช้เข้าสู่ระบบด้วยหมายเลขโทรศัพท์	ผู้ใช้ทั่วไป	
7.	U03100	ผู้ใช้เข้าสู่ระบบด้วยบัญชี facebook	ผู้ใช้ทั่วไป	
8.	U03200	ผู้ใช้เข้าสู่ระบบด้วยบัญชี instagram	ผู้ใช้ทั่วไป	
9.	U03300	ผู้ใช้เข้าสู่ระบบด้วยบัญชี google	ผู้ใช้ทั่วไป	
10.	U04000	สร้างงาน	ผู้ใช้ทั่วไป	3.4
11.	U04100	ผู้ใช้ใช้สิทธิ์รายการส่งเสริมการขาย	ผู้ใช้ทั่วไป	
12.	U04200	ชำระเงินช่องทางอื่นๆ(ยกเว้นบัตรเครดิต)	ผู้ใช้ทั่วไป	3.5
13.	U04210	ชำระเงินผ่านบัตรเครดิต	ผู้ใช้ทั่วไป	
14.	U05000	ผู้ใช้ดูข้อความแจ้งเตือน	ผู้ใช้ทั่วไป	
15.	U05100	ทำซ้ำคำสั่งงานที่หมดอายุ	ผู้ใช้ทั่วไป	
16.	U05200	ให้คะแนนการทำงานพนักงาน	ผู้ใช้ทั่วไป	
17.	U05210	บันทึกพนักงานที่ชื่นชอบ	ผู้ใช้ทั่วไป	
18.	U06000	ผู้ใช้ดูประวัติงาน	ผู้ใช้ทั่วไป	
19.	U06100	ผู้ใช้ดูประวัติงานที่กำลังดำเนินการ	ผู้ใช้ทั่วไป	
20.	U06200	ผู้ใช้ดูประวัติงานจองล่วงหน้า	ผู้ใช้ทั่วไป	
21.	U06300	ผู้ใช้ดูประวัติงานที่ทำเสร็จแล้ว	ผู้ใช้ทั่วไป	
22.	U06400	ผู้ใช้ยกเลิกงาน	ผู้ใช้ทั่วไป	

ตารางที่ 3.1 (ต่อ)

No.	Use Case ID	Use Case Name	Actor	ตาราง
23.	U06500	ติดตามงาน	ผู้ใช้ทั่วไป	
24.	U06600	ขอใบกำกับภาษี	ผู้ใช้ทั่วไป	
25.	U07000	ผู้ใช้ดูข้อมูลส่วนตัว	ผู้ใช้ทั่วไป	
26.	U07100	ผู้ใช้แก้ไขข้อมูลส่วนตัว	ผู้ใช้ทั่วไป	
27.	U08000	ดูรายชื่อพนักงานที่ชื่นชอบ	ผู้ใช้ทั่วไป	
28.	U08100	ลบรายชื่อพนักงานที่ชื่นขอบ	ผู้ใช้ทั่วไป	
29.	U09000	ผู้ใช้ดูรายการส่งเสริมการขาย	ผู้ใช้ทั่วไป	
30.	U10000	ผู้ใช้ดูคำถามที่พบบ่อย	ผู้ใช้ทั่วไป	
31.	U11000	ผู้ใช้ดูการตั้งค่าการใช้งาน	ผู้ใช้ทั่วไป	
32.	U11100	ผู้ใช้เปลี่ยนภาษา	ผู้ใช้ทั่วไป	
33.	U11200	ผู้ใช้เปิดปิดการแจ้งเตือน	ผู้ใช้ทั่วไป	
34.	U11300	ผู้ใช้ออกจากระบบ	ผู้ใช้ทั่วไป	
35.	M01000	พนักงานลืมรหัสผ่าน	พนักงาน	
36.	M02000	พนักงานสมัครสมาชิก	พนักงาน	
37.	M02100	พนักงานสมัครสมาชิกด้วยบัญชี facebook	พนักงาน	
38.	M02200	พนักงานสมัครสมาชิกด้วยบัญชี instagram	พนักงาน	
39.	M02300	พนักงานสมัครสมาชิกด้วยบัญชี google	พนักงาน	
40.	M03000	พนักงานเข้าสู่ระบบด้วยหมายเลขโทรศัพท์	พนักงาน	
41.	M03100	พนักงานเข้าสู่ระบบด้วยบัญชี facebook	พนักงาน	
42.	M03200	พนักงานเข้าสู่ระบบด้วยบัญชี instagram	พนักงาน	
43.	M03300	พนักงานเข้าสู่ระบบด้วยบัญชี google	พนักงาน	
44.	M04000	พนักงานเปิดบิดสถานะการรับงาน	พนักงาน	3.2
45.	M05000	พนักงานเลือกพื้นที่รับงาน	พนักงาน	
46.	M05100	พนักงานได้รับ Feed งานตามพื้นที่ที่เลือกไว้	พนักงาน	3.6
47.	M06000	พนักงานรับงาน	พนักงาน	3.7
48.	M07000	พนักงานดูประวัติงาน	พนักงาน	

ตารางที่ 3.1 (ต่อ)

No.	Use Case ID	Use Case Name	Actor	ตาราง
49.	M07100	พนักงานดูรายการประวัติงานวันนี้	พนักงาน	
50.	M07200	พนักงานดูรายการประวัติงานที่จองไว้	พนักงาน	
51.	M07300	พนักงานดูรายการประวัติงานที่เสร็จแล้ว	พนักงาน	
52.	M07400	พนักงานดูรายการประวัติงานทั้งหมด	พนักงาน	
53.	M07410	ดูประวัติงานทั้งหมดฟิลเตอร์วันนี้	พนักงาน	
54.	M07420	ดูประวัติงานทั้งหมดฟิลเตอร์รายสัปดาห์	พนักงาน	
55.	M07430	ดูประวัติงานทั้งหมดฟิลเตอร์รายรายเดือน	พนักงาน	
56.	M07440	ดูประวัติงานทั้งหมดฟิลเตอร์งานที่ยกเลิก	พนักงาน	
57.	M07450	ดูประวัติงานทั้งหมดฟิลเตอร์งานที่ค้างอยู่	พนักงาน	
58.	M07500	พนักงานดูรายละเอียดงาน	พนักงาน	
59.	M07510	เริ่มงาน	พนักงาน	
60.	M07520	โทรออกหาเจ้าของงาน	พนักงาน	
61.	M07530	ปิดงานแต่ละจุด	พนักงาน	
62.	M07531	ผู้รับของเซ็นรับ	พนักงาน ผู้รับของ	
63.	M07532	ถ่ายรูปปิดงาน	พนักงาน	
64.	M07540	พนักงานดูแผนที่	พนักงาน	
65.	M07541	โทรออกหาผู้รับของณตำแหน่งงานปัจจุบัน	พนักงาน	
66.	M07542	กดนำทางไปยังตำแหน่งงานปัจจุบัน	พนักงาน	
67.	M07543	ปิดงานตำแหน่งปัจจุบัน	พนักงาน	
68.	M07600	พนักงานยกเลิกงาน	พนักงาน	
69.	M08000	พนักงานดูประวัติส่วนตัว	พนักงาน	
70.	M08100	พนักงานแก้ไขประวัติส่วนตัว	พนักงาน	
71.	M09000	เช็คยอดเครดิตคงเหลือ	พนักงาน	
72.	M09100	เติมเงิน	พนักงาน	
73.	M09200	ดูประวัติรายได้	พนักงาน	

ตารางที่ 3.1 (ต่อ)

No.	Use Case ID	Use Case Name	Actor	ตาราง
74.	M09210	ดูประวัติรายได้วันนี้	พนักงาน	
75.	M09220	ดูประวัติรายได้รายสัปดาห์	พนักงาน	
76.	M09230	ดูประวัติรายได้รายเดือน	พนักงาน	
77.	M09300	ผูกบัญชีธนาคาร	พนักงาน	
78.	M10000	พนักงานดูรายการส่งเสริมการขาย	พนักงาน	
79.	M11000	พนักงานดูคำถามที่พบบ่อย	พนักงาน	
80.	M12000	พนักงานตั้งค่าการใช้งาน	พนักงาน	
81.	M12100	พนักงานเปลี่ยนภาษา	พนักงาน	
82.	M12200	พนักงานเปิดปิดการแจ้งเตือน	พนักงาน	
83.	M12300	พนักงานออกจากระบบ	พนักงาน	
84.	A01000	ผู้ดูแลระบบลืมรหัสผ่าน	ผู้ดูแลระบบ	
85.	A02000	ผู้ดูแลระบบลงชื่อเข้าใช้งานระบบ	ผู้ดูแลระบบ	
86.	A03000	ผู้ดูแลระบบลงชื่อออกจากระบบ	ผู้ดูแลระบบ	
87.	A04000	ผู้ดูแลระบบดูข้อมูลส่วนตัว	ผู้ดูแลระบบ	
88.	A04100	ผู้ดูแลระบบแก้ไขข้อมูลส่วนตัว	ผู้ดูแลระบบ	
89.	A05000	จัดการคำสั่งงาน	ผู้ดูแลระบบ	
90.	A05100	ฟิลเตอร์คำสั่งงานตามระยะเวลาและสถานะคำสั่งงาน	ผู้ดูแลระบบ	
91.	A05200	Export รายการคำสั่งงานที่ฟิลเตอร์ออกมาได้	ผู้ดูแลระบบ	
92.	A05300	ดูรายละเอียดคำสั่งงาน	ผู้ดูแลระบบ	
93.	A05400	ติดตามการทำงาน	ผู้ดูแลระบบ	
94.	A05500	ยกเลิกคำสั่งงาน	ผู้ดูแลระบบ	
95.	A05600	ปิดงานแทนพนักงาน	ผู้ดูแลระบบ	
96.	A06000	ติดตามการทำงานภายในระบบ	ผู้ดูแลระบบ	
97.	A06100	ดูรายการคำสั่งงานที่ยังไม่มีใครรับ	ผู้ดูแลระบบ	

ตารางที่ 3.1 (ต่อ)

No.	Use Case ID	Use Case Name	Actor	ตาราง
98.	A06200	ดูรายการคำสั่งงานที่ถึงเวลาเริ่มงานแต่พนักงานยังไม่กดเริ่มงานหรือรายการที่เลยเวลาปิดงานมาแล้ว	ผู้ดูแลระบบ	
99.	A06300	ดูตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานจากบนแผนที	ผู้ดูแลระบบ	3.8
100.	A06400	มอบหมายงานให้กับพนักงานที่ยังว่างงาน	ผู้ดูแลระบบ	
101.	A07000	จัดการพื้นที่ให้บริการ	ผู้ดูแลระบบ	
102.	A07100	เพิ่มพื้นที่ให้บริการ	ผู้ดูแลระบบ	
103.	A07200	ลบพื้นที่ให้บริการ	ผู้ดูแลระบบ	
104.	A07300	แก้ไขพื้นที่ให้บริการ	ผู้ดูแลระบบ	
105.	A08000	จัดการบริการเสริม	ผู้ดูแลระบบ	
106.	A08100	เพิ่มบริการเสริม	ผู้ดูแลระบบ	
107.	A08200	ลบบริการเสริม	ผู้ดูแลระบบ	
108.	A08300	แก้ไขบริการเสริม	ผู้ดูแลระบบ	
109.	A09000	จัดการรายการส่งเสริมการขาย	ผู้ดูแลระบบ	
110.	A09100	เพิ่มรายการส่งเสริมการขาย	ผู้ดูแลระบบ	
111.	A09200	ลบรายการส่งเสริมการขาย	ผู้ดูแลระบบ	
112.	A09300	แก้ไขรายการส่งเสริมการขาย	ผู้ดูแลระบบ	
113.	A10000	จัดการข้อมูลผู้ใช้งานทั่วไป	ผู้ดูแลระบบ	
114.	A10100	ฟิลเตอร์ผู้ใช้งานทั่วไป	ผู้ดูแลระบบ	
115.	A10200	ค้นหาผู้ใช้งานทั่วไป	ผู้ดูแลระบบ	
116.	A10300	ดูข้อมูลส่วนตัวผู้ใช้งานทั่วไป	ผู้ดูแลระบบ	
117.	A10310	แก้ไขข้อมูลส่วนตัวผู้ใช้งานทั่วไป	ผู้ดูแลระบบ	
118.	A10400	จัดการคำสั่งงานของผู้ใช้	ผู้ดูแลระบบ	
119.	A10410	ฟิลเตอร์คำสั่งงานของผู้ใช้	ผู้ดูแลระบบ	
120.	A10420	ค้นหาคำสั่งงานของผู้ใช้	ผู้ดูแลระบบ	

ตารางที่ 3.1 (ต่อ)

No.	Use Case ID	Use Case Name	Actor	ตาราง
121.	A10430	Export รายการคำสั่งงานของผู้ใช้	ผู้ดูแลระบบ	
122.	A10440	ดูรายละเอียดคำสั่งงานของผู้ใช้แต่ละรายการ	ผู้ดูแลระบบ	
123.	A10450	ติดตามการทำงานของคำสั่งงาน	ผู้ดูแลระบบ	
124.	A10460	ผู้ดูแลระบบยกเลิกคำสั่งงานแทนผู้ใช้งาน	ผู้ดูแลระบบ	
125.	A10470	ดูยอดรายจ่ายและช่องทางการจ่ายเงินทั้งหมดของผู้ใช้	ผู้ดูแลระบบ	
126.	A11000	จัดการข้อมูลพนักงาน	ผู้ดูแลระบบ	
127.	A11100	ฟิลเตอร์พนักงาน	ผู้ดูแลระบบ	
128.	A11200	ค้นหาพนักงาน	ผู้ดูแลระบบ	
129.	A11300	ดูข้อมูลส่วนตัวพนักงาน	ผู้ดูแลระบบ	
130.	A11310	แก้ไขข้อมูลส่วนตัวพนักงาน	ผู้ดูแลระบบ	
131.	A11400	จัดการคำสั่งงานของพนักงาน	ผู้ดูแลระบบ	
132.	A11410	ฟิลเตอร์คำสั่งงานของพนักงาน	ผู้ดูแลระบบ	
133.	A11420	ค้นหาคำสั่งงานของพนักงาน	ผู้ดูแลระบบ	
134.	A11430	Export รายการคำสั่งงานของพนักงาน	ผู้ดูแลระบบ	
135.	A11440	ดูรายละเอียดคำสั่งงานของพนักงานแต่ละรายการ	ผู้ดูแลระบบ	
136.	A11450	ติดตามการทำงานของคำสั่งงานของพนักงาน	ผู้ดูแลระบบ	
137.	A11460	ผู้ดูแลระบบยกเลิกคำสั่งงานแทนพนักงาน	ผู้ดูแลระบบ	
138.	A11470	ดูยอดรายได้ของพนักงาน	ผู้ดูแลระบบ	
139.	A11471	ฟิลเตอร์ยอดรายได้ของพนักงาน	ผู้ดูแลระบบ	
140.	A11472	Export รายการรายได้ของพนักงาน	ผู้ดูแลระบบ	
141.	A12000	จัดการข้อมูลผู้ดูแลระบบ	ผู้ดูแลระบบ	
142.	A12100	ค้นหาข้อมูลผู้ดูแลระบบ	ผู้ดูแลระบบ	
143.	A12200	เพิ่มข้อมูลผู้ดูแลระบบ	ผู้ดูแลระบบ	
144.	A12300	แก้ไขข้อมูลผู้ดูแลระบบ	ผู้ดูแลระบบ	
145.	A13000	ออกใบกำกับภาษี	ผู้ดูแลระบบ	

ตารางที่ 3.1 (ต่อ)

No.	Use Case ID	Use Case Name	Actor	ตาราง
146.	A13100	ค้นหารายการร้องขอใบกำกับภาษี	ผู้ดูแลระบบ	
147.	A13200	ตรวจสอบการออกใบกำกับภาษี	ผู้ดูแลระบบ	
148.	A13300	ส่งอีเมล์ใบกำกับภาษีให้กับผู้ใช้	ผู้ดูแลระบบ	
149.	A14000	ดูรายการประวัติการชำระเงินผ่านบัตรเครดิต	ผู้ดูแลระบบ	
150.	A14100	ค้นหาประวัติการชำระเงินผ่านบัตรเครดิต	ผู้ดูแลระบบ	
151.	A15000	ดูประวัติการใช้สิทธิ์โปรโมชั่น	ผู้ดูแลระบบ	
152.	A15100	ค้นหาประวัติการใช้สิทธิ์โปรโมชั่น	ผู้ดูแลระบบ	
153.	A16000	ดูรายการประวัติการอนุมัติการเติมเครดิตของพนักงาน	ผู้ดูแลระบบ	
154.	A16100	ค้นหารายการประวัติการอนุมัติการเติมเครดิตของพนักงาน	ผู้ดูแลระบบ	
155.	A17000	ตรวจสอบรายการเติมเครดิตพนักงาน	ผู้ดูแลระบบ	
156.	A17100	ค้นหารายการเติมเครดิตพนักงาน	ผู้ดูแลระบบ	
157.	A17200	ตรวจสอบและอนุมัติการเติมเครดิตพนักงาน	ผู้ดูแลระบบ	
158.	A18000	ตั้งค่าการใช้งานของระบบ	ผู้ดูแลระบบ	
159.	A19000	จัดการรายการคำถามที่พบบ่อย(FAQ)	ผู้ดูแลระบบ	
160.	A19100	เพิ่มรายการคำถามที่พบบ่อย(FAQ)	ผู้ดูแลระบบ	
161.	A19200	แก้ไขรายการคำถามที่พบบ่อย(FAQ)	ผู้ดูแลระบบ	
162.	A19300	ลบรายการคำถามที่พบบ่อย(FAQ)	ผู้ดูแลระบบ	
163.	S01000	จัดการคำสั่งงานที่ไม่ได้รับการยืนยัน	ระบบ	
165.	S03000	แจ้งเตือนพนักงานก่อนถึงเวลาเริ่มงานก่อน 1 วัน	ระบบ	
166.	S04000	แจ้งเตือนพนักงานก่อนถึงเวลาเริ่มงาน 5 นาที	ระบบ	
167.	S05000	ตรวจสอบเอกสารของพนักงานที่ใกล้หมดอายุ	ระบบ	
168.	S06000	แจ้งเตือนพนักงานที่ถูกแบนจากการยกเลิกงาน	ระบบ	
169.	S07000	จัดการข้อมูล OTP	ระบบ	

ตารางที่ 3.1 (ต่อ)

No.	Use Case ID	Use Case Name	Actor	ตาราง
170.	S08000	จัดการข้อมูลแจ้งเตือน	ระบบ	
171.	S09000	จัดการ Log	ระบบ	

ตารางที่ 3.2 Use Case พนักงานเปิดปิดสถานะการรับงาน

Use Case ID	M04000	
Use Case Name	พนักงานเปิดปิดสถานะการรับงาน	
Actor	พนักงาน	
Scenarios	1. พนักงานเปิดปิดสถานะการรับงานจากหน้าแอปพลิเคชัน	
Preconditions	1. ผ่านการทำงานตาม Use Case M03000, M03100, M03200, M03300 อย่างใดอย่างหนึ่ง (ล็อกอินเข้าสู่ระบบ)	
Postconditions	1. ผู้ดูแลระบบต้องเห็นสถานะการทำงานของพนักงานและตำแหน่งปัจจุบันของพนักงานบนหน้าจอ Monitor เมื่อพนักงานเปิดสถานะการรับงาน 2. เมื่อพนักงานปิดสานะการรับงาน ตำแหน่งปัจจุบันของพนักงานจะหายไปจากหน้าจอ Monitor ของผู้ดูแลระบบ 3. สถานะการรับงานจะไม่เปลี่ยนแปลงเมื่อมีการล็อกอินเข้าสู่ระบบใหม่อีกครั้ง และต้องคงอยู่จนกว่าพนักงานจะเปิดปิดสถานะอีกครั้ง 4. สัมพันธ์กับ Use Case A06300	
Basic Flows	Actor	System
	1. กดเปิดสถานะการรับงาน	1.1 ระบบส่งข้อมูลไปบันทึกยัง Server 1.2 แอปพลิเคชันแสดงสัญลักษณ์ว่าพนักงานเปิดรับงานอยู่
	2. กดปิดสถานะการรับงาน	2.1 ระบบส่งข้อมูลไปบันทึกยัง Server 2.2 แอปพลิเคชันแสดงสัญลักษณ์ว่าพนักงานปิดรับงานอยู่

ตารางที่ 3.3 Use Case พนักงานเลือกพื้นที่รับงาน

Use Case ID	M05000	
Use Case Name	พนักงานเลือกพื้นที่รับงาน	
Actor	พนักงาน	
Scenarios	1. เลือกพื้นที่รับงาน	
Preconditions	1. ผ่านการทำงานตาม Use Case M03000, M03100, M03200, M03300 อย่างใดอย่างหนึ่ง (ล็อกอินเข้าสู่ระบบ) 2. ผ่านการทำงานตาม Use Case M04000 (พนักงานเปิดปิดสถานะการรับงาน)	
Postconditions	1. Feed งานที่พนักงานได้รับจะต้องเป็นงานที่อยู่ในพื้นที่ที่พนักงานเลือกไว้ 2. พื้นที่ที่เลือกไว้จะไม่เปลี่ยนแปลงเมื่อมีการล็อกอินเข้าสู่ระบบใหม่อีกครั้ง และต้องคงอยู่จนกว่าพนักงานจะเลือกพื้นที่ใหม่ 3. สัมพันธ์กับ Use Case M05100	
Basic Flows	Actor	System
	1. เลือกพื้นที่รับงานจากรายการพื้นที่ทั้งหมดที่แสดงบนหน้าแอปพลิเคชัน	1.1 ระบบส่งข้อมูลไปบันทึกยัง Server 1.2 แอปพลิเคชันแสดงชื่อพื้นที่ที่พนักงานเลือก
	2. กดปิดสถานะการรับงาน	1.1 ระบบส่งข้อมูลไปบันทึกยัง Server 2.1 แอปพลิเคชันแสดงชื่อพื้นที่ที่พนักงานเลือกสัญลักษณ์ว่าพนักงานปิดรับงาน

ตารางที่ 3.4 Use Case ผู้ใช้สร้างงาน

Use Case ID	U04000	
Use Case Name	ผู้ใช้สร้างงาน	
Actor	ผู้ใช้ทั่วไป	
Scenarios	1. พนักงานสร้างงานโดยการระบุสถานที่รับส่งของจากหน้าแอปพลิเคชัน	
Preconditions	1. ผ่านการทำงานตาม Use Case U03000, U03100, U03200, U03300 อย่างใดอย่างหนึ่ง (ล็อกอินเข้าสู่ระบบ) 2. ผู้ใช้อยู่ที่หน้าจอทำงานหลัก	
Postconditions	1. เมื่อกรอกข้อมูลครบระบบจะคำนวณค่าบริการแสดงให้ผู้ใช้ทรา 2. สัมพันธ์กับ Use Case U04200	
Basic Flows	Actor	System
	1. กดปุ่มเริ่มการใช้งาน	1.1 ระบบแสดงหน้าจอให้ผู้ใช้กรอกข้อมูลระบุสถานที่
	2. กดเพื่อเข้าไปกรอกข้อมูลระบุสถานที่รับของเป็นจุดที่ 1	2.1 ระบบแสดงหน้าจอให้ผู้ใช้กรอกข้อมูลสถานที่รับของ
	3. กรอกข้อมูลที่จำเป้นดังต่อไปนี้ - ชื่อสถานที่ - ชื่อผู้ติดต่อ - หมายเลขโทรศัพท์ผู้ติดต่อ	3.1 แสดงข้อมูลที่ผู้ใช้กรอกบนหน้าแอปพลิเคชัน
	4. กดเข้าไประบุตำแหน่งของสถานที่จากหน้า Map	4.1 แสดงหน้า Map ให้ผู้ใช้สามารถปักหมุดระบุตำแหน่งของสถานที่ โดยแสดงจุดเริ่มต้นอยู่ตรงตำแหน่งปัจจุบันของผู้ใช้เอง
	5. เลือกตำแหน่งบน Map เพื่อทำการปักหมุด	5.1 แสดงตำแหน่งที่ผู้ใช้เลือกบนหน้า Map

ตารางที่ 3.4 (ต่อ)

	6. กดที่ปุ่มเลือกจุดเพื่อปักหมุด	6.1 แอปพลิเคชันบันทึกข้อมุลเก็บไว้ 6.2 ย้อนกลับไปหน้ากรอกข้อมูลก่อนหน้า
	7. กดยืนยันเพื่อบันทึกจุดรับของ	7.1 แอปพลิเคชันบันทึกข้อมุลเก็บไว้ 7.2 ย้อนกลับไปหน้ากรอกข้อมูลเพื่อสร้างคำสั่งงานก่อนหน้า
	8. กดเพื่อเข้าไปกรอกข้อมูลระบุสถานที่ส่งของเป็นจุดที่ 2	8.1 ระบบแสดงหน้าจอให้ผู้ใช้กรอกข้อมูลสถานที่ส่งของ
	9. ทำขั้นตอน 3-7	9.1 แอปพลิเคชันบันทึกข้อมุลเก็บไว้ 9.2 ย้อนกลับไปหน้ากรอกข้อมูลเพื่อสร้างคำสั่งงานก่อนหน้า 9.3 ระบบส่งข้อมูลไปบันทึกยัง Server และคำนวณค่าบริการส่งกลับมา 9.4 แอปพลิเคชันแสดงยอดค่าบริการ

ตารางที่ 3.5 Use Case ผู้ใช้กดชำระค่าบริการผ่านช่องทางอื่นๆ(ยกเว้นบัตรเครดิต)

Use Case ID	U04200	
Use Case Name	ผู้ใช้กดชำระค่าบริการผ่านช่องทางอื่นๆ(ยกเว้นบัตรเครดิต)	
Actor	ผู้ใช้ทั่วไป	
Scenarios	1. ผู้ใช้ยืนยันคำสั่งงานโดยการกดชำระค่าบริการผ่านช่องทางที่ไม่ใช่บัตรเครดิต	
Preconditions	2. ผ่านการทำงานตาม Use Case U04000 (การสร้างงาน) 3. ผู้ใช้อยู่ที่หน้าจอการสร้างงาน	
Postconditions	1. คำสั่งงานจะถูกกระจายส่งไปให้พนักงานที่เปิดรับงานและออนไลน์อยู่ในระบบ 2. พนักงานจะเห็นงานเข้ามาในหน้า Feed งาน 3. สัมพันธ์กับ Use Case M05100	
Basic Flows	Actor	System
	1. กดปุ่มชำระเงิน	1.1 ระบบแสดงรายการช่องทางการชำระเงินบนหน้าจอ
	2. เลือกช่องทางการชำระเงิน (ยกเว้นช่องทางผ่านบัตรเครดิต)	2.1 หน้าจอแสดงสัญลักษ์หน้ารายการที่ถูกเลือก
	3. กดยืนยัน	3.1 ระบบส่งข้อมูลไปบันทึกยัง Server 3.2 แจ้งผลการส่งคำสั่งงานบนหน้าจอให้ผู้ใช้ทราบ

ตารางที่ 3.6 Use Case พนักงานได้รับ Feed งานตามพื้นที่ที่เลือกไว้

Use Case ID	M05100	
Use Case Name	พนักงานได้รับ Feed งานตามพื้นที่ที่เลือกไว้	
Actor	พนักงาน	
Scenarios	1. พนักงานรอรับคำสั่งงานใหม่จากผู้ใช้ที่หน้า Feed งาน	
Preconditions	1. ผ่านการทำงานตาม Use Case U04000, U04200 (การสร้างงาน) 2. ผ่านการทำงานตาม Use Case M04000, M05000 3. โดยที่ Use Case M05000 พนักงานเลือกพื้นที่ในการรับงานเป็นพื้นที่เดียวกับที่ผู้ใช้สร้างงาน หรือเลือกพื้นที่รับงานเป็นทุกเขต 4. พนักงานเปิดหน้าแอปพลิเคชันรอไว้ที่หน้า Feed งาน	
Postconditions	1. มีงานเข้ามาในหน้า Feed งานหลังจากผู้ใช้ยืนยันส่งคำสั่งงาน	
Basic Flows	Actor	System
	1. เปิดหน้าแอปพลิเคชันรอไว้ที่หน้า Feed งาน	1.1 ระบบ push งานส่งมาแสดงที่หน้า Feed งาน

ตารางที่ 3.7 Use Case พนักงานรับงาน

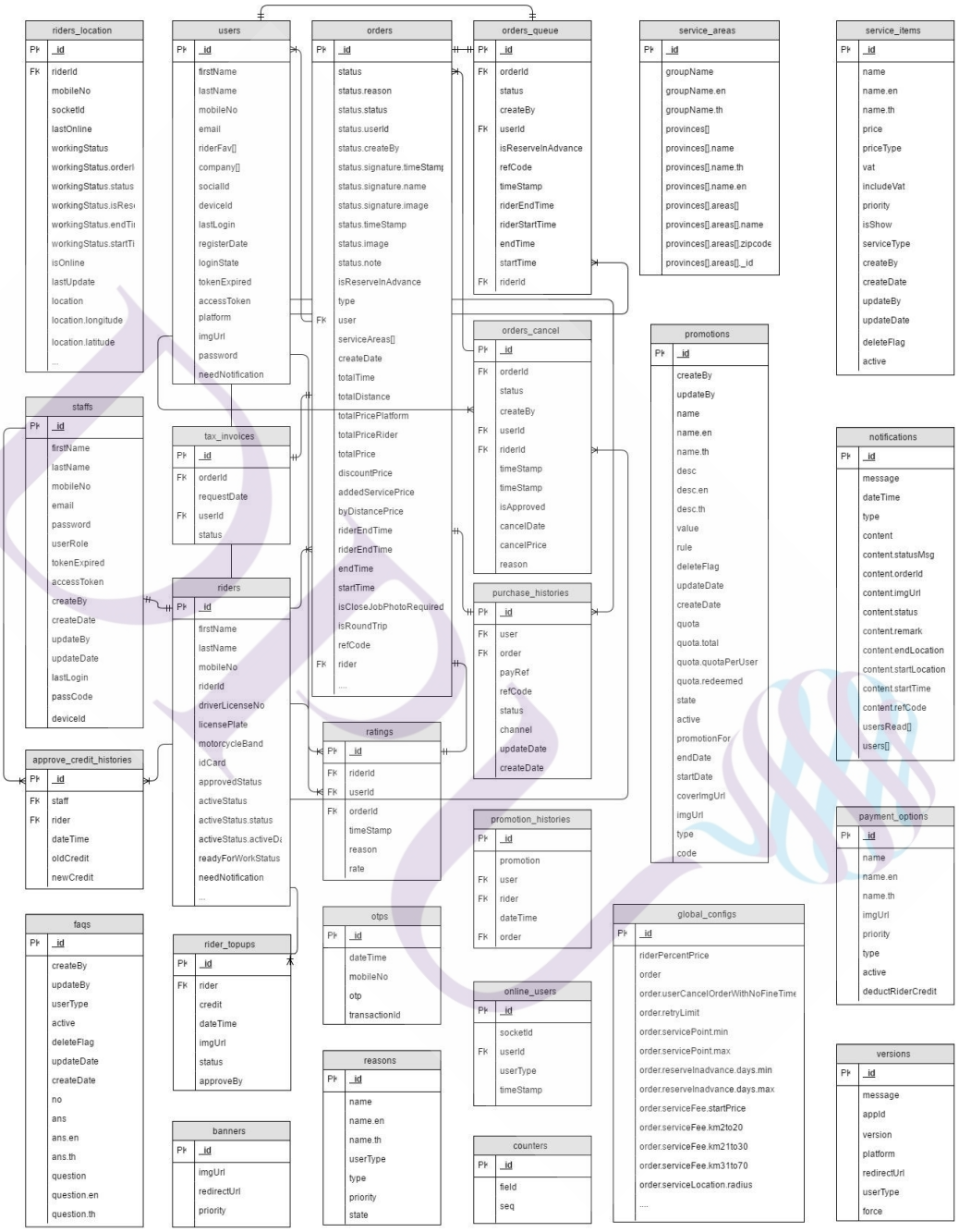
Use Case ID	M06000	
Use Case Name	พนักงานรับงาน	
Actor	พนักงาน	
Scenarios	1. พนักงานกดรับงานที่ได้รับจากหน้า Feed งาน	
Preconditions	1. ผ่านการทำงานตาม Use Case U04000, U04200 (การสร้างงาน) 2. ผ่านการทำงานตาม Use Case M04000, M05000 3. พนักงานได้รับ Feed งานเข้ามาแล้ว	
Postconditions	1. คำสั่งงานเปลี่ยนสถานะเป็น accepted 2. พนักงานสามารถเริ่มปฏิบัติงานตามคำสั่งงานแต่ละจุดได้ 3. สถานการณ์รับงานของพนักงานที่หน้าจอ Monitor ของผู้ดูแลระบบเปลี่ยนเป็นรับงานอยู่	
Basic Flows	Actor	System
	1. กดเข้าไปดูรายละเอียดงาน	1.1 แอปพลิเคชันแสดงหน้ารายละเอียดของงาน
	2. กดรับงาน	2.1 ระบบส่งข้อมูลไปบันทึกยัง Server 2.2 แอปพลิเคชันแสดงผลรายงานการรับงานที่หน้าจอ พร้อมแสดงปุ่มให้ยกเลิกงานและปุ่มเริ่มงาน
	3. กดปุ่มเริ่มงาน	3.1 ระบบส่งข้อมูลไปบันทึกยัง Server 3.2 แอปพลิเคชันแสดงหน้าจอการทำงานของคำสั่งงานนั้น

ตารางที่ 3.8 Use Case ผู้ดูแลระบบดูตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานจากบนแผนที่

Use Case ID	A06300	
Use Case Name	ดูตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานจากบนแผนที่	
Actor	ผู้ดูแลระบบ	
Scenarios	1. ผู้ดูแลระบบติดตามสถานะการทำงานของพนักงานและตำแหน่งปัจจุบันบนหน้าจอแผนที่	
Preconditions	1. มีพนักงานอยู่ในระบบอย่างน้อย 1 คน 2. พนักงานออนไลน์อยู่ในระบบ โดยผ่าน Use Case M04000 และทำการเปิดสถานะพร้อมรับงานไว้ 3. หรือพนักงานผ่านการทำงานตาม Use Case M06000 (พนักงานรับงาน) 4. ผู้ดูแลระบบ ผ่านขั้นตอนตาม Use Case A02000 (ลงชื่อเข้าใช้งานระบบ) 5. ผู้ดูแลระบบ ผ่านขั้นตอนตาม Use Case A06000 (ติดตามการทำงานภายในระบบ)	
Postconditions	1. ผู้ดูแลระบบเห็นสถานะการรับงานของพนักงานและตำแหน่งจากบนหน้า Map	
Basic Flows	Actor	System
	1. กดเข้าสู่หน้าจอ Monitor จากแถบเมนู	1.1 ระบบแสดงหน้าจอ Monitor
	2. เลื่อนดูสถานะการรับงานของพนักงานบน Map	2.1 แสดงหน้าจอตามการควบคุมของผู้ดูแลระบบ

3.2.3 การออกแบบฐานข้อมูล

จากการศึกษาข้อมูลปัญหาและความต้องการของระบบ ได้วิเคราะห์และออกแบบฐานข้อมูลระบบบริการขนส่งของและติดตามการขนส่งแบบเรียลไทม์ ดังแสดงได้ตาม E-R Diagram



ภาพที่ 3.6 ER – Diagram ระบบบริการขนส่งของและติดตามการขนส่งแบบเรียลไทม์

สำหรับรายละเอียดตารางข้อมูล แสดงดังตารางที่ 3.9 และ พจนานุกรมข้อมูล (Data Dictionary) แสดงตามภาคผนวก ก

ตารางที่ 3.9 แสดงรายละเอียดตารางข้อมูลระบบบริการขนส่งของและติดตามการทำงานแบบเรียลไทม์

No.	Table Name	Description
1	approve_credit_histories	ตารางเก็บข้อมูลประวัติการอนุมัติเครดิตพนักงาน
2	banners	ตารางเก็บข้อมูลป้ายโฆษณา
3	counters	ตารางเก็บข้อมูลตัวนับ
4	faqs	ตารางเก็บข้อมูลคำถามที่พบบ่อย
5	global_configs	ตารางเก็บข้อมูลค่าคอนฟิกในระบบ
6	notifications	ตารางเก็บข้อมูลข้อความแจ้งเตือน
7	online_users	ตารางเก็บข้อมูลผู้ใช้ที่ออนไลน์ในระบบ
8	orders	ตารางเก็บข้อมูลคำสั่งงาน
9	orders_cancel	ตารางเก็บข้อมูลคำสั่งงานที่ถูกยกเลิก
10	orders_queue	ตารางเก็บข้อมูลคำสั่งงานที่รอการดำเนินงาน
11	otps	ตารางเก็บข้อมูลรหัสผ่าน OTP
12	payment_options	ตารางเก็บข้อมูลช่องทางการจ่ายเงิน
13	promotion_histories	ตารางเก็บข้อมูลประวัติการใช้สิทธิ์โปรโมชั่น
14	promotions	ตารางเก็บข้อมูลโปรโมชั่น
15	purchase_histories	ตารางเก็บข้อมูลประวัติการชำระค่าบริการผ่านบัตรเคดิต
16	ratings	ตารางเก็บข้อมูลประวัติการให้คะแนนพนักงาน
17	reasons	ตารางเก็บข้อมูลเหตุผลการยกเลิกงานและการให้คะแนน
18	rider_topups	ตารางเก็บข้อมูลการเติมเครดิตพนักงานที่รอการอนุมัติ
19	riders	ตารางเก็บข้อมูลพนักงานส่งของ
20	riders_location	พิตารางเก็บข้อมูลกัดปัจจุบันของพนักงานส่งของ
21	service_areas	ตารางเก็บข้อมูลพื้นที่ให้บริการ

ตารางที่ 3.9 (ต่อ)

No.	Table Name	Description
22	service_items	ตารางเก็บข้อมูลบริการเสริม
23	staffs	ตารางเก็บข้อมูลเจ้าหน้าที่ดูแลระบบ
24	tax_invoices	ตารางเก็บข้อมูลรายการขอใบกำกับภาษี
25	users	ตารางเก็บข้อมูลข้อมูลผู้ใช้
26	versions	ตารางเก็บข้อมูลเวอร์ชั่นล่าสุดของแอพลิเคชัน

3.2.4 แผนผังแอปพลิเคชันที่ทำงานภายในระบบ

จากการศึกษาปัญหาและความต้องการของระบบ การออกแบบโครงสร้างระบบทำโดยพิจารณาจากสภาพการใช้งานระบบจริง ซึ่งแบ่งตามบทบาทของผู้ใช้ได้ดังนี้

1. แผนผังแอปพลิเคชัน สำหรับผู้ใช้งานทั่วไป

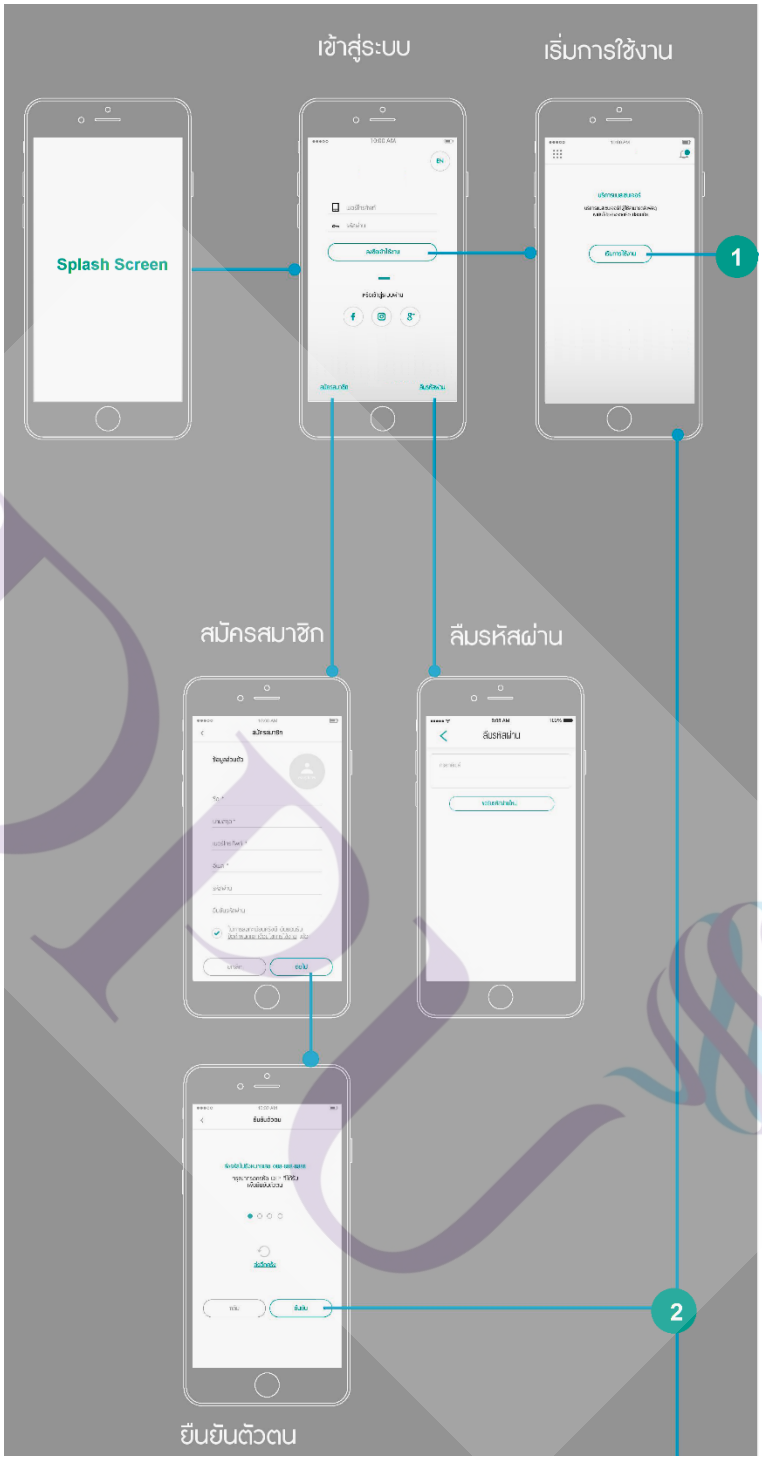
จากภาพที่ 3.7 แผนผังแอปพลิเคชัน สำหรับผู้ใช้งานทั่วไป แสดงแผนผังของระบบที่ผู้ใช้งานทั่วไปสามารถเข้าใช้งานได้ ซึ่งประกอบด้วย การสร้างคำสั่งงาน การดูประวัติงาน การจัดการข้อมูลส่วนตัวของผู้ใช้ การดูรายการส่งเสริมการขาย การดูรายการคำถามที่พบบ่อย และการตั้งค่าการใช้งานของแอปพลิเคชัน

2. แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ

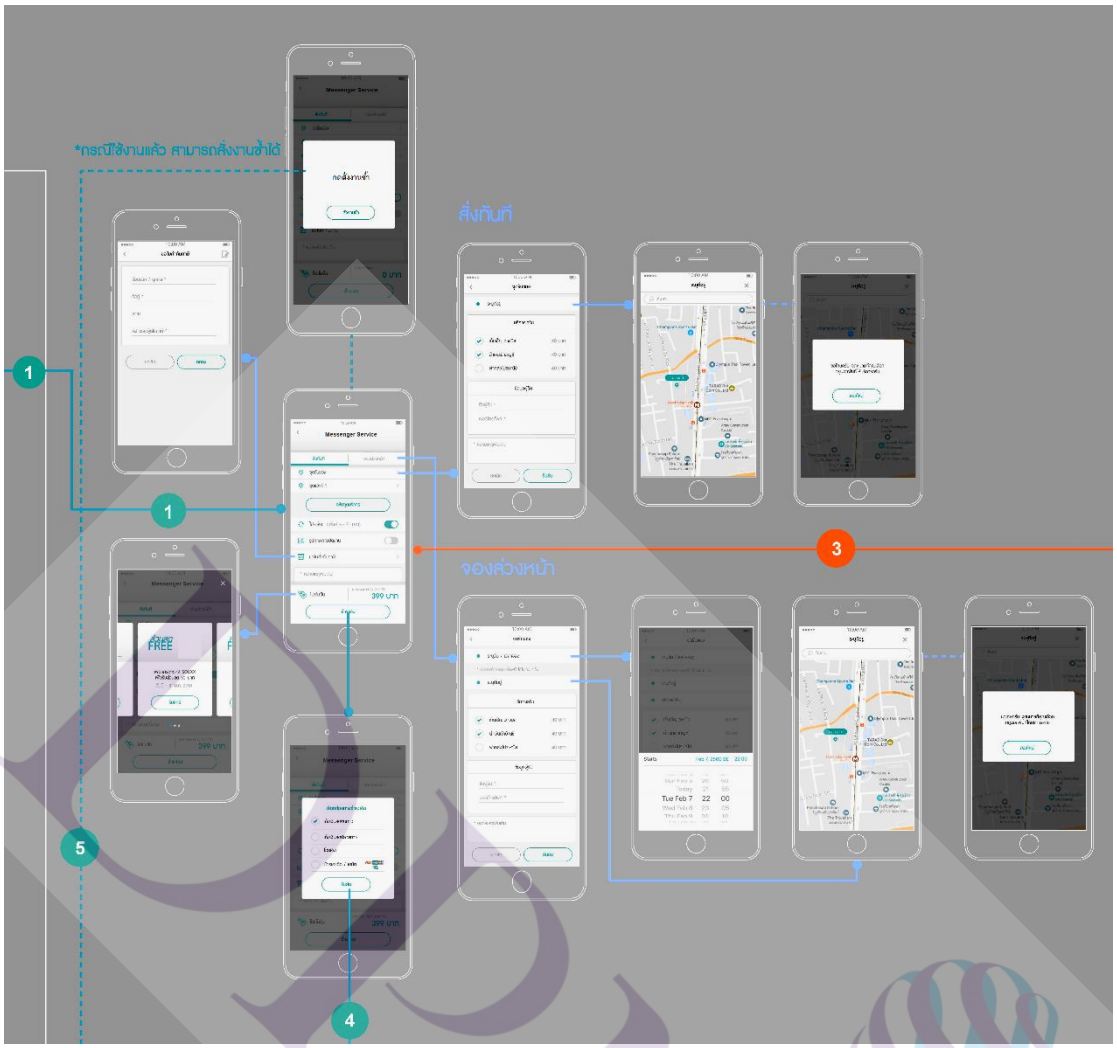
จากภาพที่ 3.8 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ แสดงแผนผังของระบบที่พนักงานส่งของสามารถเข้าใช้งานได้ ซึ่งประกอบด้วย การดูรายการคำสั่งงานที่เข้ามาใหม่ การดูรายละเอียดคำสั่งงาน การรับงาน การดูประวัติงาน การจัดการข้อมูลส่วนตัวของพนักงาน การจัดการเกี่ยวกับเครดิต การดูประวัติรายได้ การดูรายการส่งเสริมการขาย การดูรายการคำถามที่พบบ่อย และการตั้งค่าการใช้งานของแอปพลิเคชัน

3. แผนผังเว็บแอปพลิเคชัน สำหรับผู้ดูแลระบบ

จากภาพที่ 3.9 แผนผังเว็บแอปพลิเคชัน สำหรับผู้ดูแลระบบ แสดงแผนผังของระบบที่ผู้ดูแลระบบสามารถเข้าใช้งานเพื่อจัดการดูแลระบบ ซึ่งประกอบด้วย การแก้ไขข้อมูลส่วนตัวของผู้ดูแลระบบ การจัดการข้อมูลคำสั่งงาน การติดตามการทำงานของพนักงาน การจัดการผู้ใช้ทั่วไป การจัดการข้อมูลพนักงาน การจัดการข้อมูลผู้ดูแลระบบ การจัดการพื้นที่ให้บริการ การจัดการโปรโมชัน การจัดการประวัติการทำงานในระบบ การจัดการคำถามที่พบบ่อย การจัดการบริการเสริม การจัดส่งใบกำกับภาษีให้ผู้ใช้ การอนุมัติเครดิตของพนักงาน และประวัติการจ่ายเงินผ่านบัตรเครดิต การใช้สิทธิ์โปรโมชัน รายการเครดิตที่ได้รับการอนุมัติ

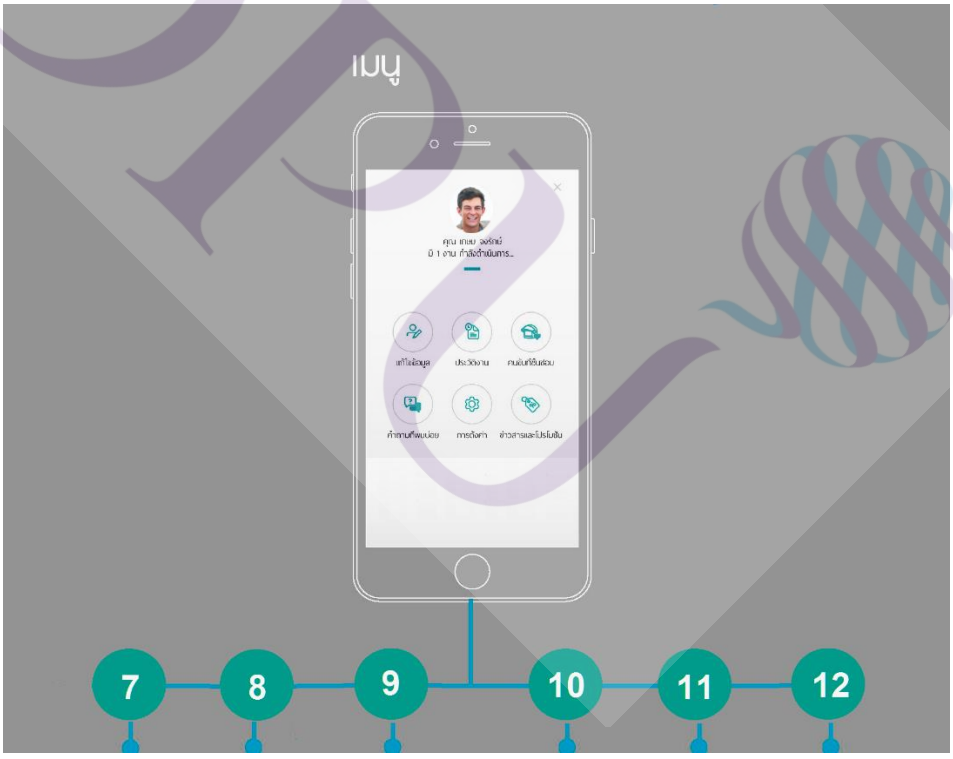


ภาพที่ 3.7 แผนผังแอปพลิเคชันสำหรับผู้ใช้งานทั่วไป (1)

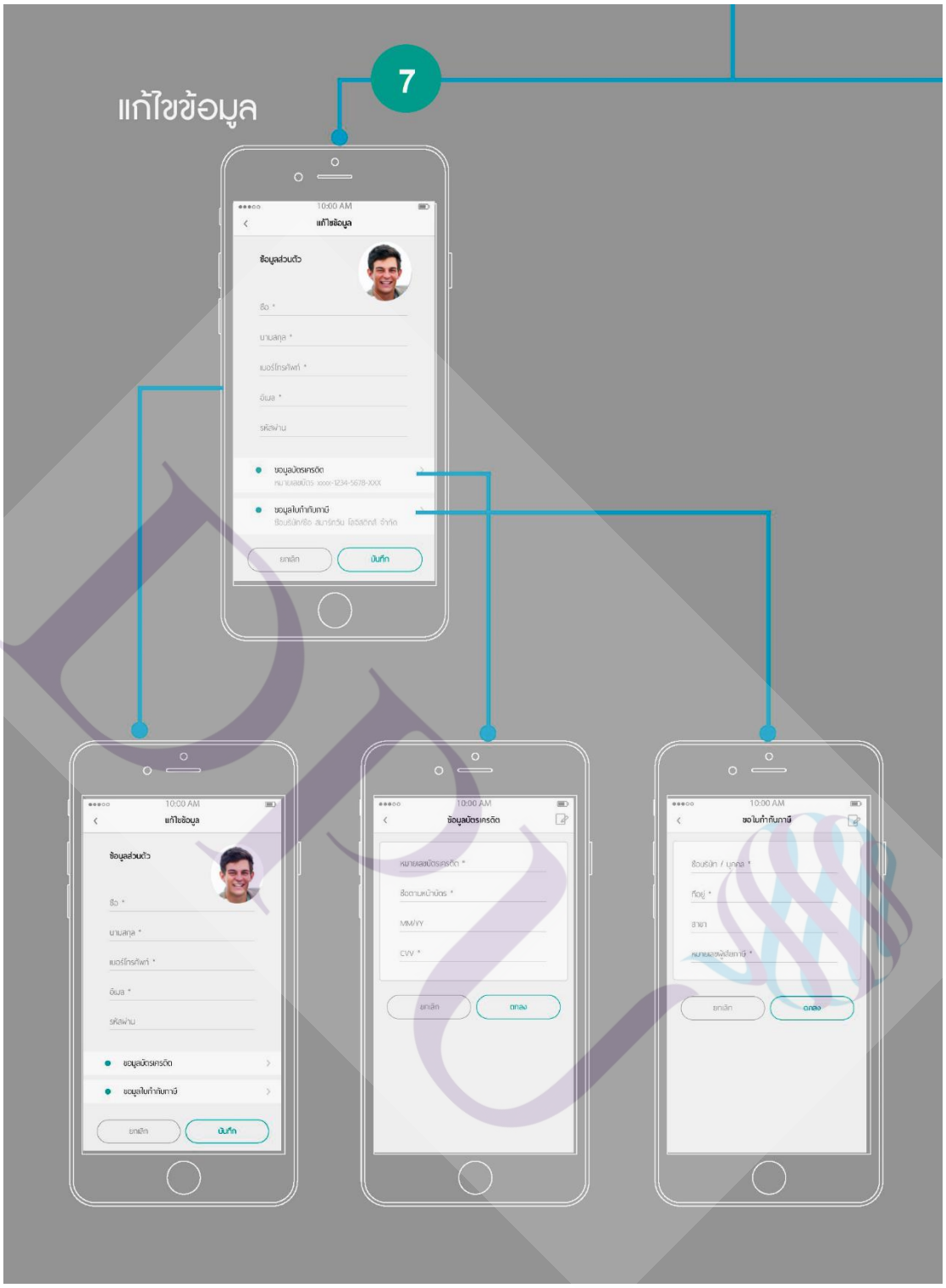


ภาพที่ 3.8 แผนผังแอปพลิเคชันสำหรับผู้ใช้งานทั่วไป (2)

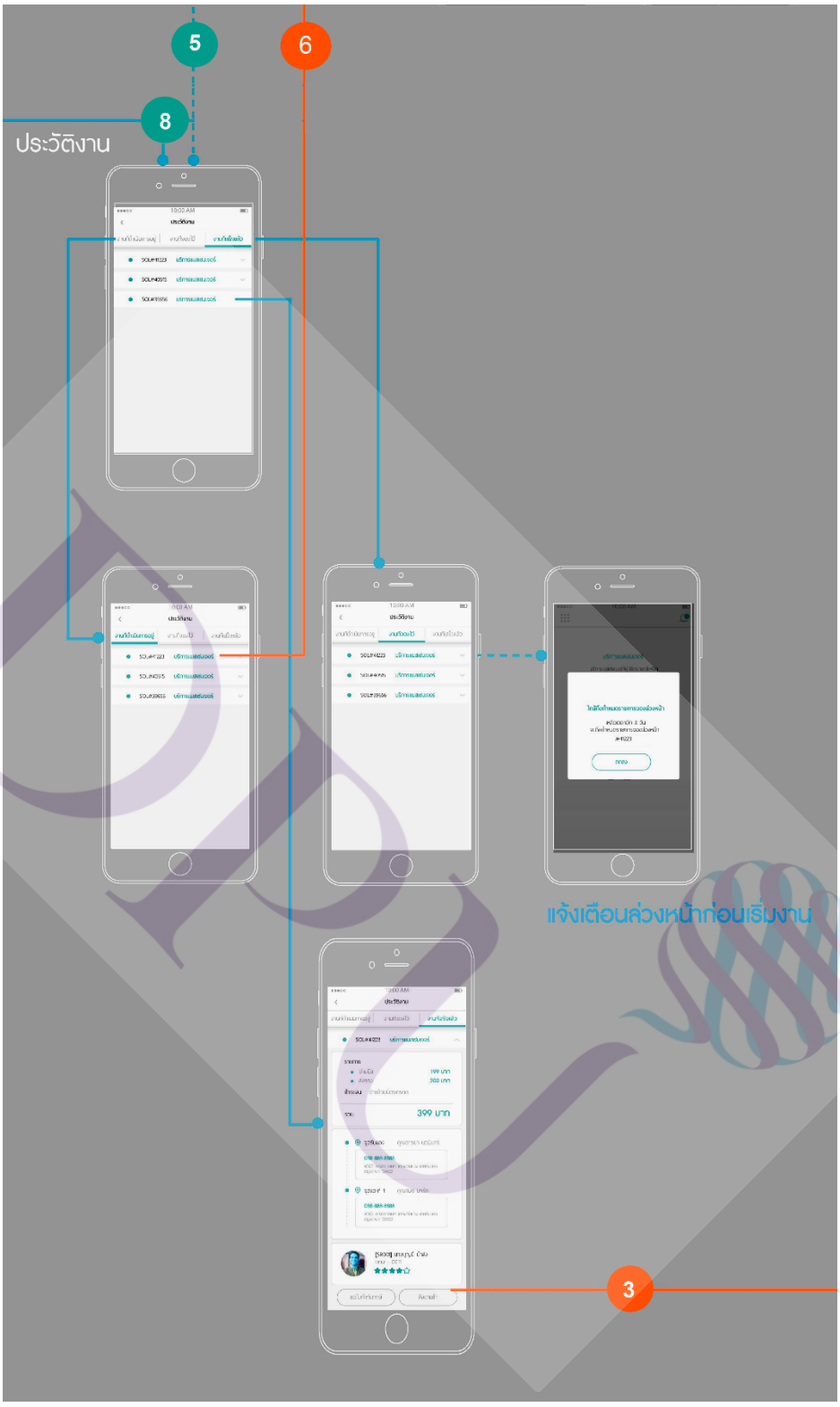
ภาพที่ 3.9 แผนผังแอปพลิเคชันสำหรับผู้ใช้งานทั่วไป (3)



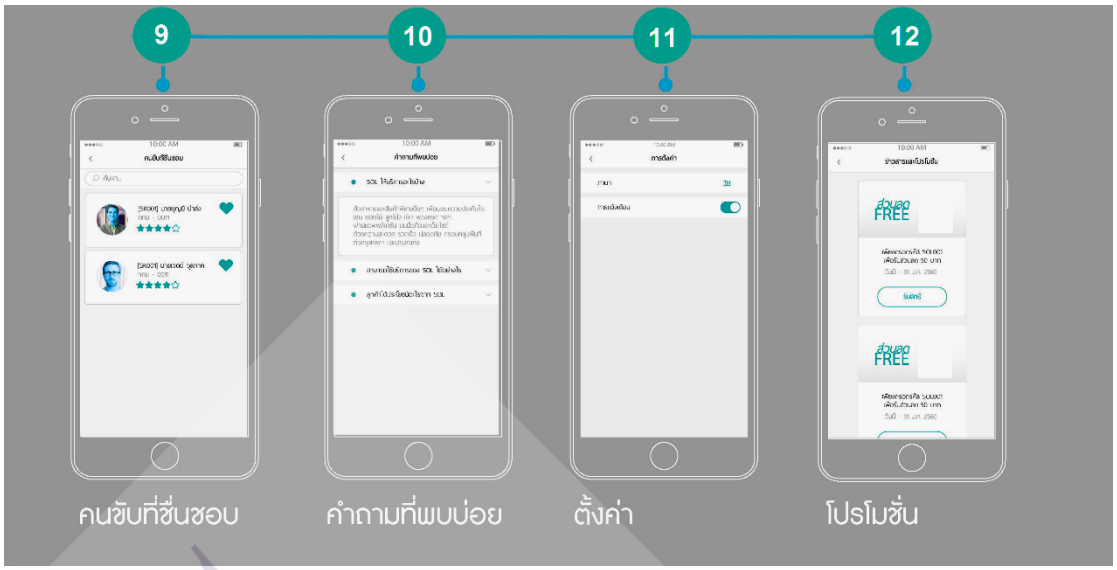
ภาพที่ 3.10 แผนผังแอปพลิเคชันสำหรับผู้ใช้งานทั่วไป (4)



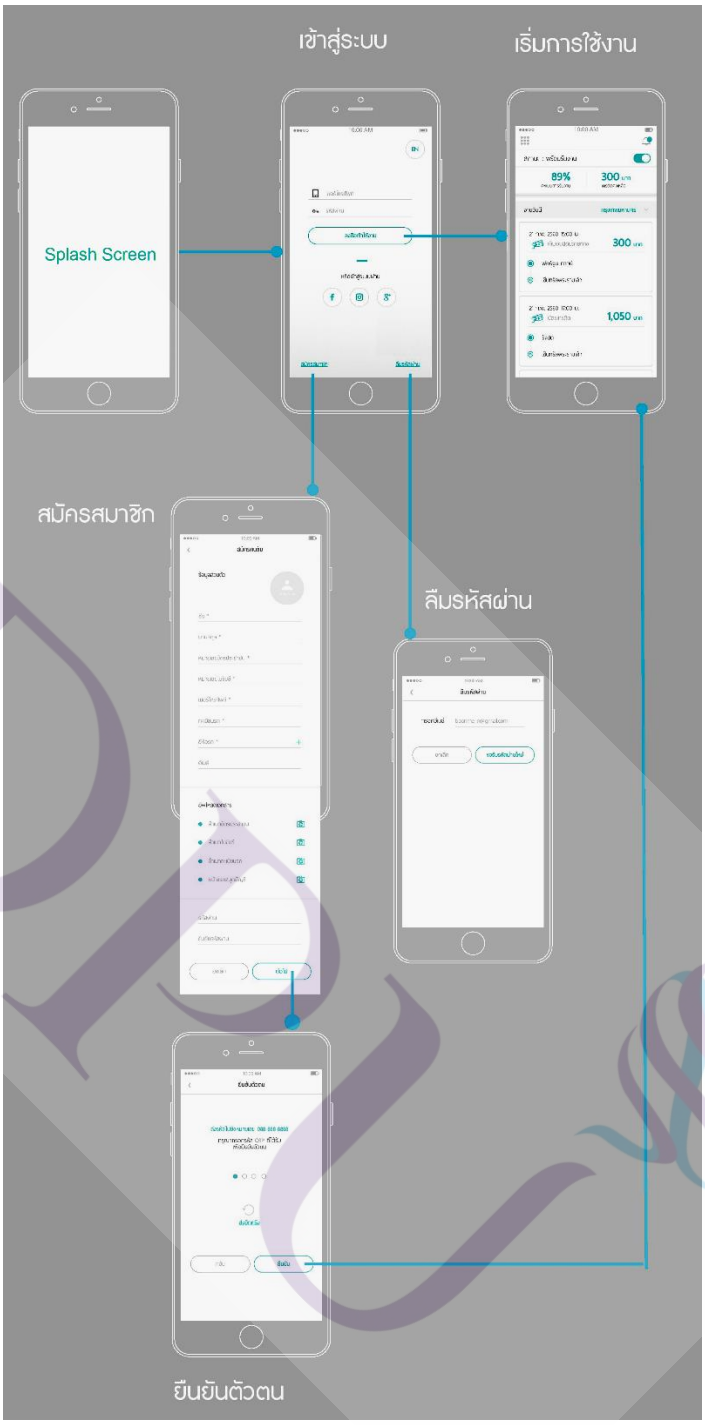
ภาพที่ 3.11 แผนผังแอปพลิเคชันสำหรับผู้ใช้งานทั่วไป (5)



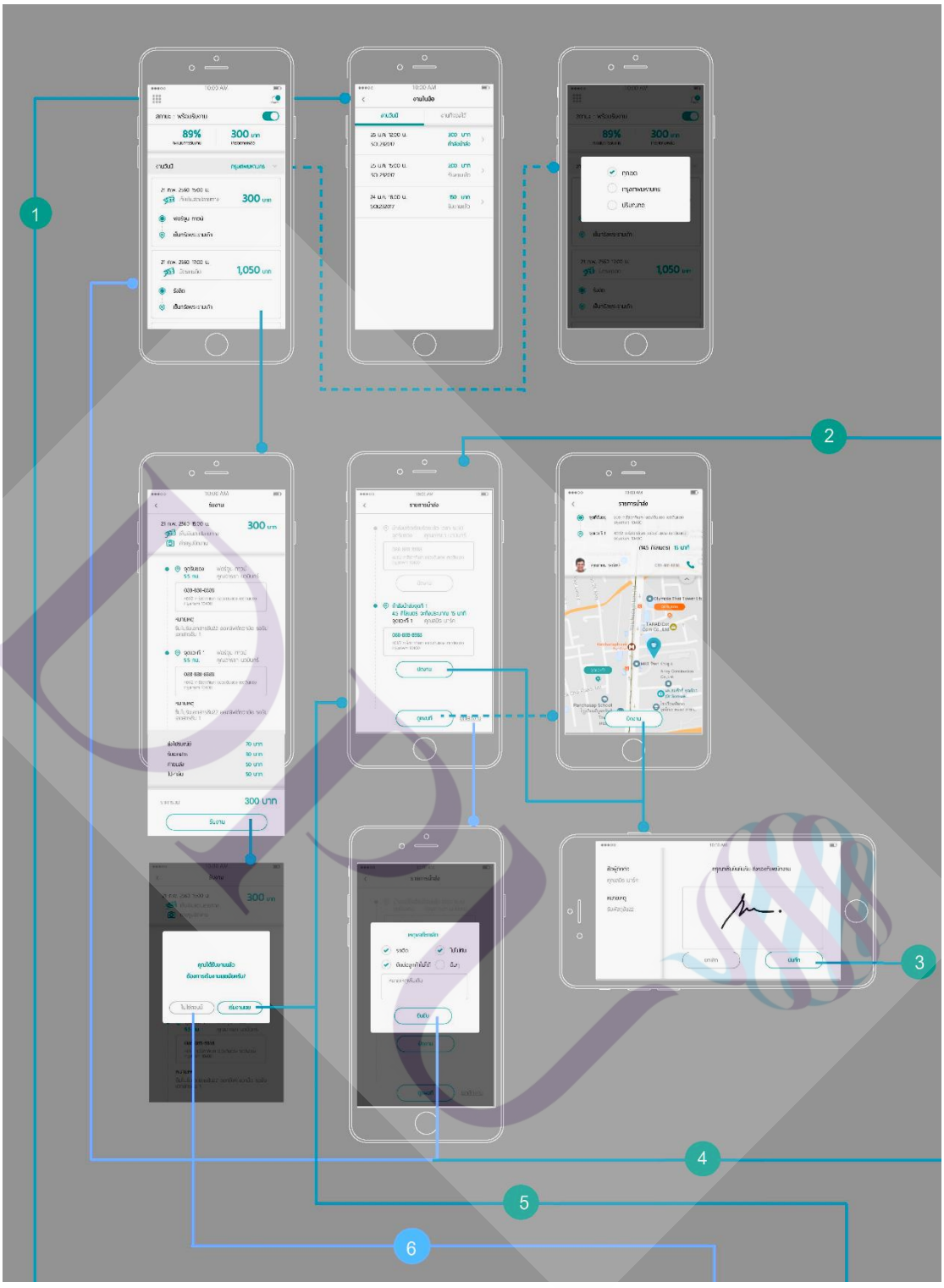
ภาพที่ 3.12 แผนผังแอปพลิเคชันสำหรับผู้ใช้งานทั่วไป (6)



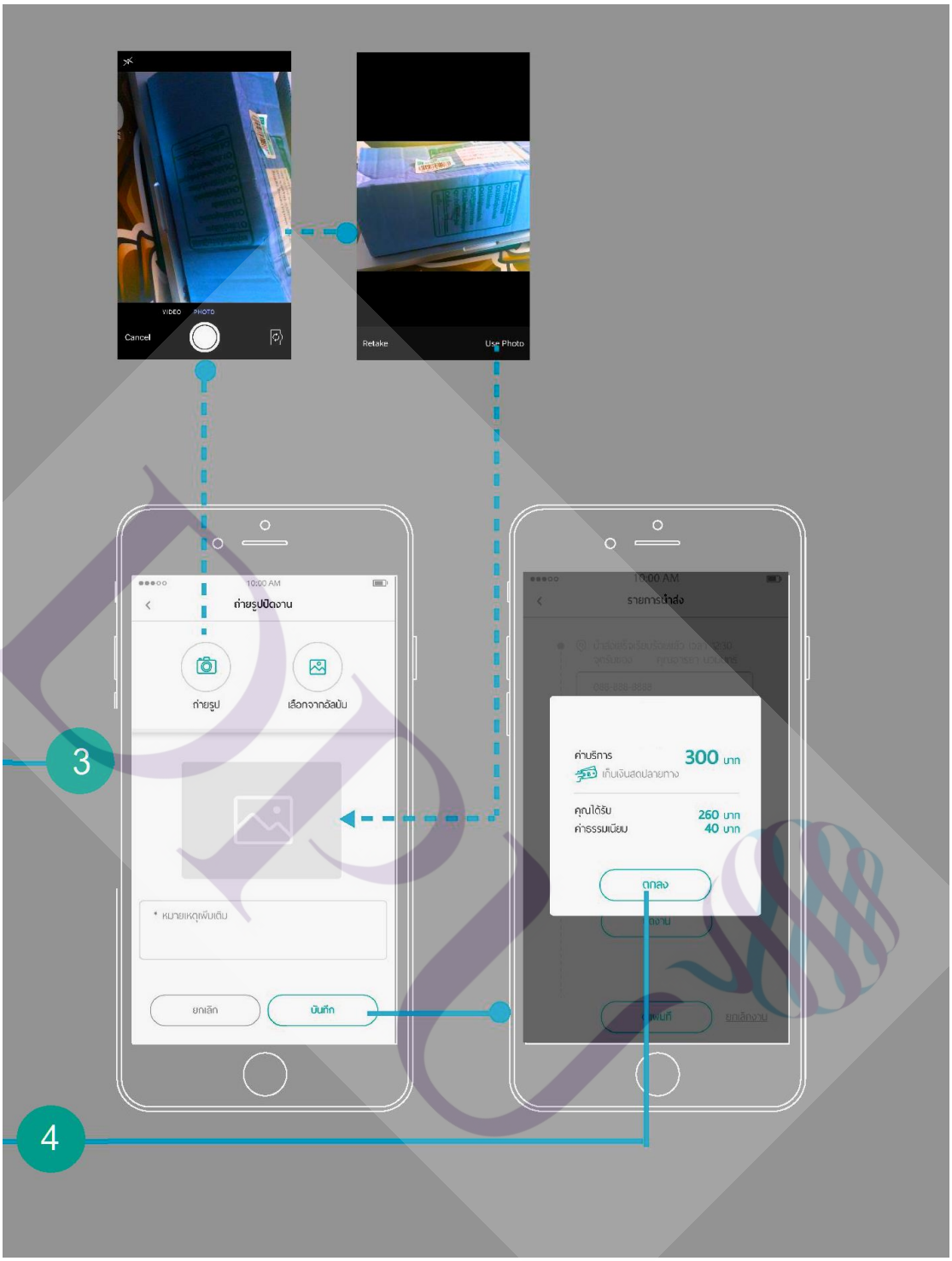
ภาพที่ 3.13 แผนผังแอปพลิเคชันสำหรับผู้ใช้งานทั่วไป (7)



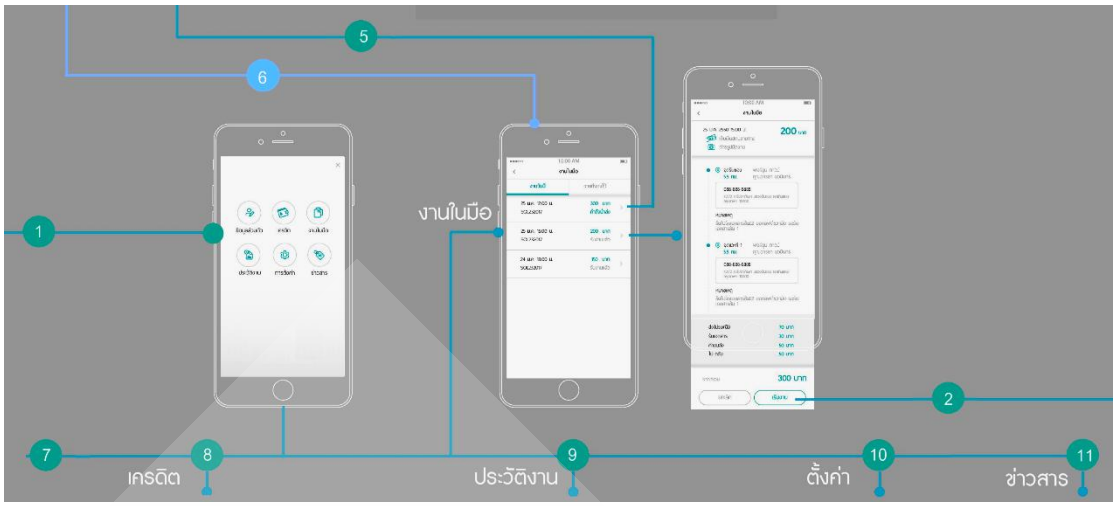
ภาพที่ 3.14 แผนผังแอปพลิเคชันสำหรับพนักงานส่งของ (1)



ภาพที่ 3.15 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ (2)

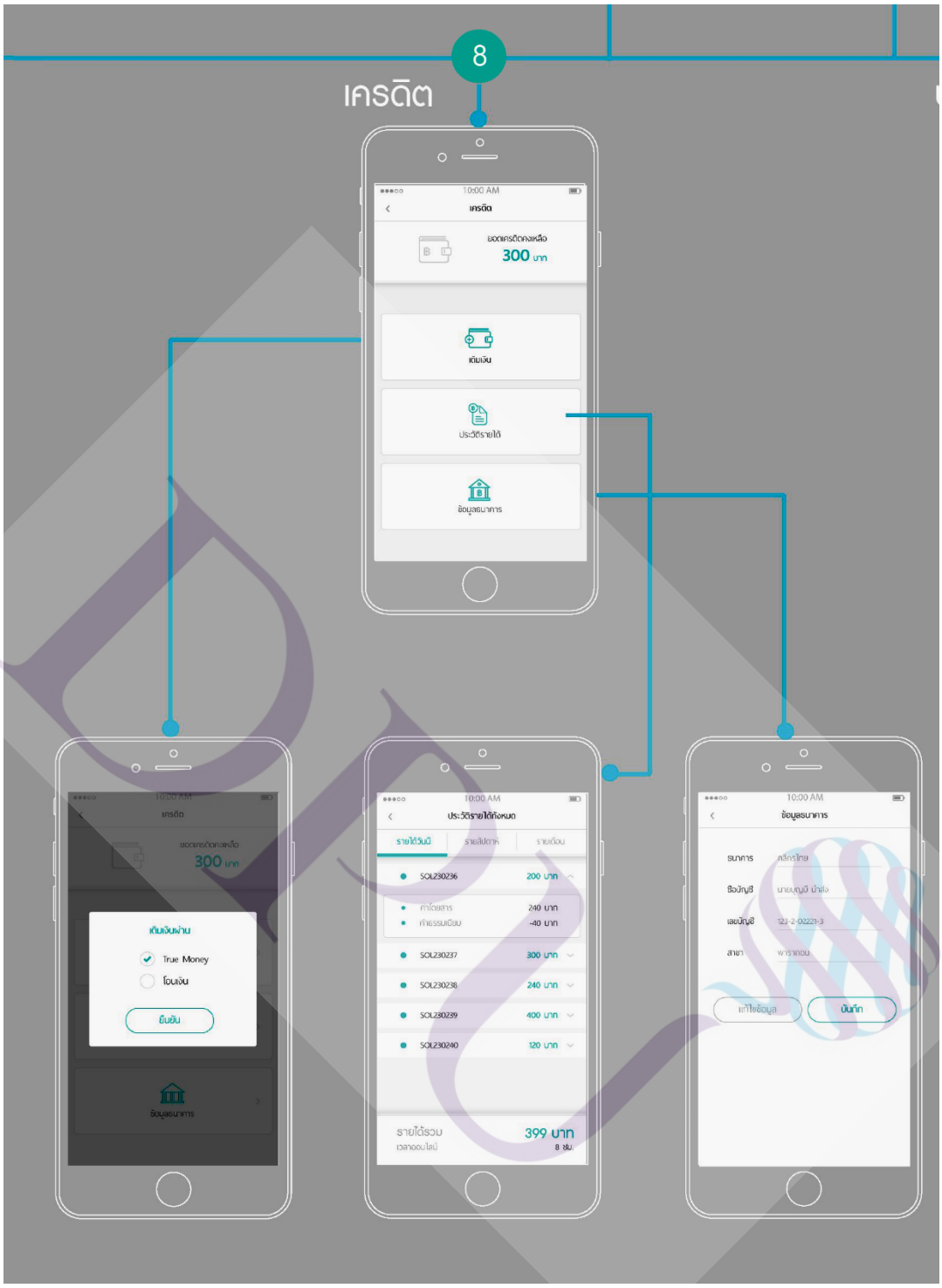


ภาพที่ 3.16 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ (3)

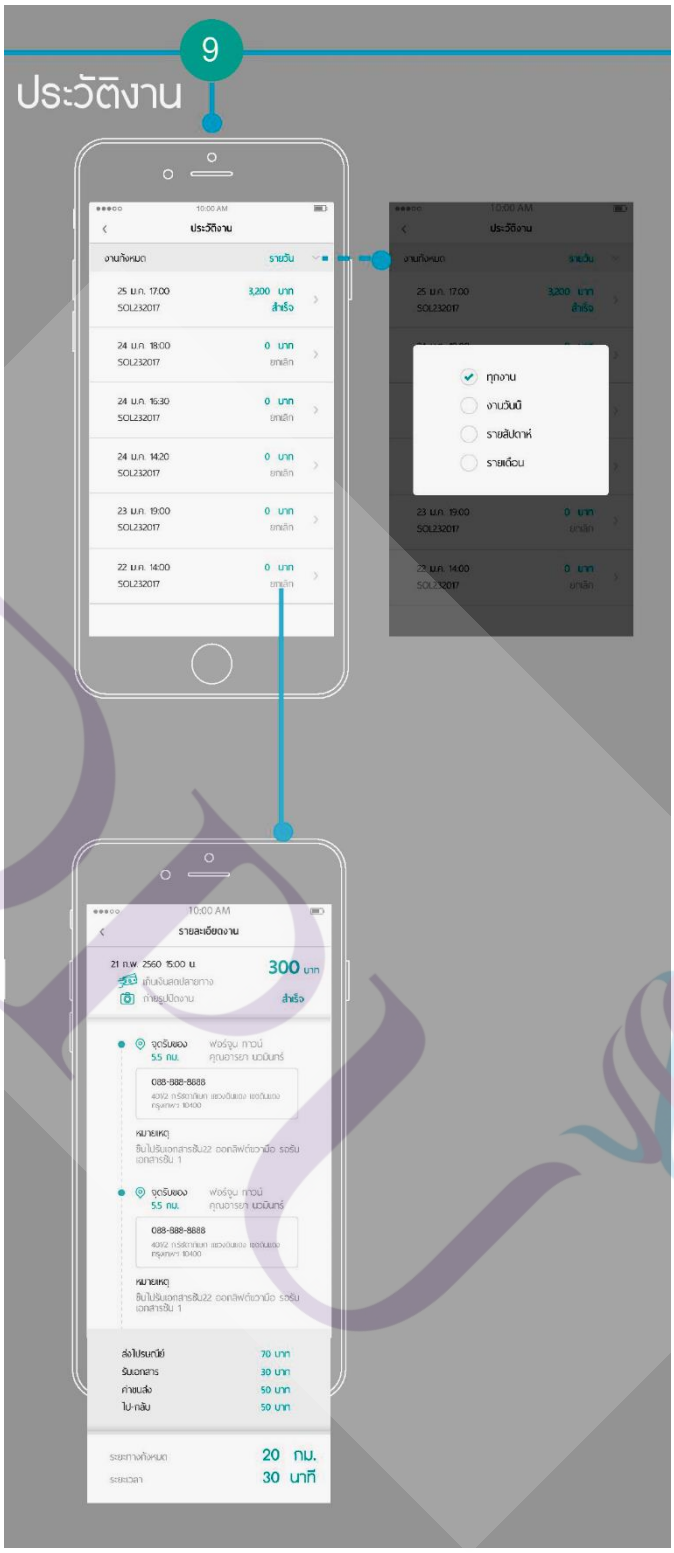


ภาพที่ 3.17 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ (3)

ภาพที่ 3.18 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ (4)

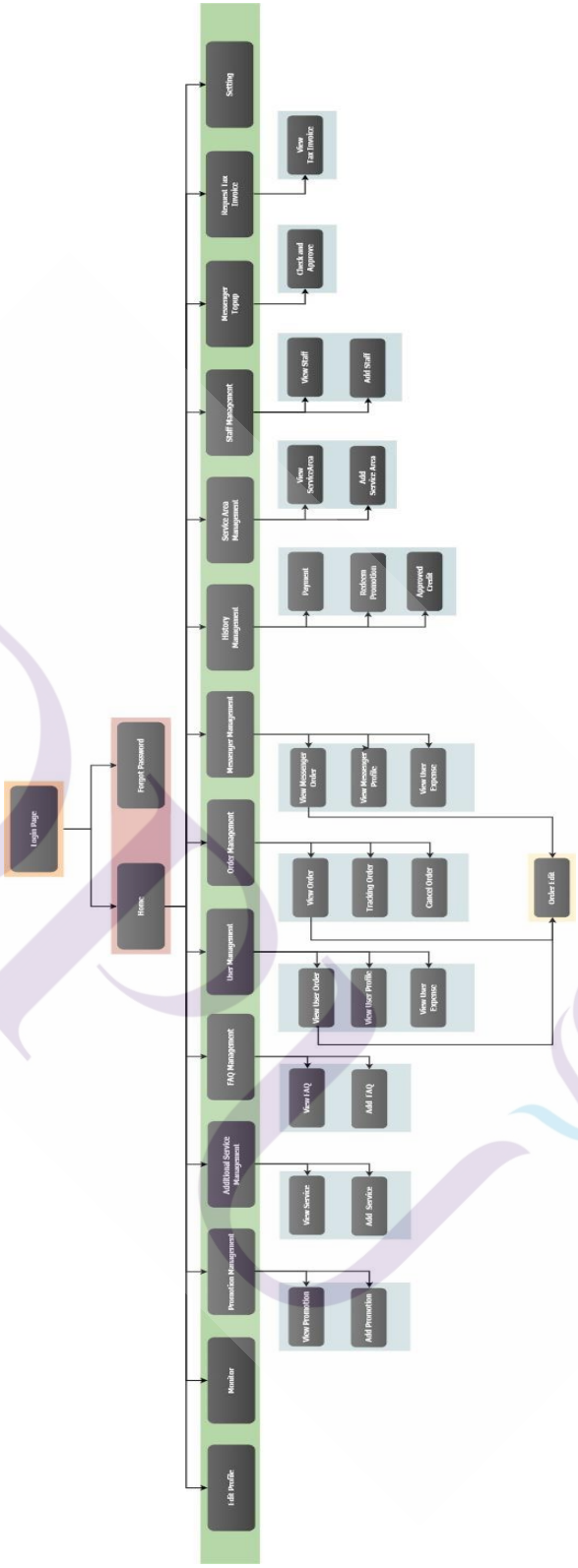


ภาพที่ 3.19 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ (5)



ภาพที่ 3.20 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ (5)

ภาพที่ 3.21 แผนผังแอปพลิเคชัน สำหรับพนักงานส่งของ (6)



ภาพที่ 3.22 แผนผังการทำงานในเว็บแอปพลิเคชันสำหรับผู้ดูแลระบบ

3.2.5 การออกแบบ API

จากการศึกษาปัญหาและความต้องการของระบบและจากการวิเคาระห์ ออกมาเป็น Use Case ทั้งหมด จึงสามารถออกแบบ API สำหรับเว็บเซอร์วิส และ WebSocket Event ได้ดังในตารางที่ 3.10 และ 3.11 ส่วนรายละเอียดของแต่ละ API สามารถดูได้ที่ภาคผนวก ข และ ค

3.2.5.1 REST API

ในส่วนของการออกแบบ REST API หลังจากที่มีการวิเคราะห์และออกแบบมาได้ทั้งหมดแล้วจึงได้มีการเขียน API ทั้งหมดให้อยู่ในรูปแบบของ Swagger หรือ OpenAPI Specification โดยใช้ภาษา Yaml แล้วนำไฟล์ Spec ที่ได้ส่งให้กับนักพัฒนาระบบในฝั่งไคลเอนท์เพื่อนำไปสร้างโค้ดคำสั่งจาก Swagger Codegen สำหรับเรียกใช้งานเว็บเซอร์วิสในฝั่งเซอร์เวอร์ต่อไป โดยรายละเอียดของแต่ละ API สามารถดูได้ที่ภาคผนวก ข

ตารางที่ 3.10 API เว็บเซอร์วิสของระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์

No.	API		Description
1.	**taxInvoice**		
1.1	POST	/order/user/taxInvoice/{orderId}	ผู้ใช้ส่งคำขอใบกำกับภาษี
2.	**promotion**		
2.1	GET	/promotion/{userType}	เรียกดูรายการโปรโมชั่น
2.2	POST	/promotion/{userType}/code	ขอใช้สิทธิ์โปรโมชั่น
3.	**reason**		
3.1	GET	/reasons/{userType}/{type}	เรียกดูรายการเหตุผลการยกเลิกงาน
3.2	GET	/reasons/{userType}/{type}/{state}	เรียกดูรายการเหตุผลตามการใช้งาน
4.	**payment**		
4.1	GET	/paymentOptions/user	เรียกดูช่องทางการชำระเงินของผู้ใช้
4.2	GET	/paymentOptions/rider	เรียกดูช่องทางการชำระเงินของพนักงาน
5.	**terms**		
5.1	GET	/terms	เรียกดูหน้าเว็บเพจเงื่อนไขการใช้งาน
6.	**serviceItem**		
6.1	GET	/serviceItems	เรียกดูรายการบริการเสริม

ตารางที่ 3.10 (ต่อ)

No.	API		Description
7.	**notification**		
7.1	GET	/notifications/{userType}	เรียกดูรายการข้อความแจ้งเตือน
7.2	POST	/notifications/{userType}	อ่านข้อความแจ้งเตือน
7.3	POST	/notifications/{userType}/{notificationId}	อ่านข้อความแจ้งเตือน
7.4	DELETE	/notifications/{userType}/{notificationId}	ลบข้อความแจ้งเตือน
8.	**OTP**		
8.1	POST	/otp/send	ขอรับข้อความรหัสผ่าน OTP
8.2	POST	/otp/confirm	ยืนยันรหัสผ่าน OTP
9.	**user**		
9.1	POST	/users/register	ผู้ใช้สมัครสมาชิกเข้าใช้งานระบบ
9.2	POST	/users/forgotPass	ผู้ใช้แจ้งลืมรหัสผ่าน
9.3	POST	/users/resetPass	ผู้ใช้เปลี่ยนรหัสผ่านหลังจากได้รับ OTP
9.4	POST	/users/updateProfile	ผู้ใช้เปลี่ยนแปลงแก้ไขข้อมูลส่วนตัว
9.5	POST	/users/changeMobileNo	ผู้ใช้ร้องขอเปลี่ยนหมายเลขโทรศัพท์
9.6	POST	/users/updateMobileNo	ผู้ใช้ยืนยันเปลี่ยนหมายเลขโทรศัพท์พร้อม OTP
9.7	POST	/users/changePass	ผู้ใช้ร้องขอเปลี่ยนรหัสผ่าน
9.8	POST	/users/setNewPass	ผู้ใช้เปลี่ยนรหัสผ่านพร้อม OTP

ตารางที่ 3.10 (ต่อ)

No.	API		Description
9.9	POST	/users/logout	ผู้ใช้ลงชื่อออกจากระบบ
9.10	GET	/users/{_id}	เรียกดูข้อมูลส่วนตัว
9.11	GET	/users/rider	เรียกดูรายการพนักงานที่ชื่นชอบ
9.12	POST	/users/rider	บันทึกรายการพนักงานที่ชื่นชอบ
10.	**banner**		
10.1	GET	/banners	เรียกดูรายการป้ายโฆษณา
11.	**order**		
11.1	POST	/order/user	ผู้ใช้สร้างคำสั่งงาน
11.2	GET	/order/user	ผู้ใช้เรียกดูรายการประวัติคำสั่งงาน
11.3	GET	/order/user/{orderId}	ผู้ใช้เรียกดูคำสั่งงานแต่ละรายการ
11.4	DELETE	/order/user/{orderId}	ผู้ใช้ลบคำสั่งงานแต่ละรายการ
11.5	GET	/order/rider	ผู้ใช้เรียกรายการพนักงานที่ชื่นชอบ
11.6	GET	/order/rider/available	พนักงานเรียกดูคำสั่งงานที่สามารถรับได้
11.7	GET	/order/rider/{orderId}	พนักงานเรียกดูรายละเอียดคำสั่งงาน
11.8	POST	/order/user/confirm/{orderId}	ผู้ใช้ยืนยันคำสั่งงาน
11.9	POST	/order/user/retry	ผู้ใช้ลองส่งคำสั่งงานอีกครั้ง
11.10	POST	/order/rider/accept/{orderId}	พนักงานรับงาน
11.11	POST	/order/rider/start/{orderId}	พนักงานเริ่มงาน

ตารางที่ 3.10 (ต่อ)

No.	API		Description
11.12	POST	/order/rider/cancel	พนักงานยกเลิกงาน
11.13	POST	/order/user/cancel	ผู้ใช้งานยกเลิกงาน
11.14	POST	/order/rider/update	พนักงานปิดงาน
11.15	POST	/order/user/rate	ผู้ใช้ให้คะแนนพนักงาน
11.16	GET	/order/user/riderLocation/{orderId}	ผู้ใช้เรียกดูพิกัดพนักงานที่กำลังวิ่งงาน
12.	**sol**		
12.1	GET	/sol/banks	เรียกดูรายการบัญชีธนาคารเพื่อโอนเงิน
13.	**authen**		
13.1	POST	/users/login	ผู้ใช้ลงชื่อเข้าสู่ระบบ
14.	**serviceAreas**		
14.1	GET	/serviceAreas	เรียกดูรายการพื้นที่ให้บริการ
15.	**faq**		
15.1	GET	/faqs	ผู้ใช้เรียกดูรายการคำถามที่พบบ่อย
15.2	GET	/faqs/rider	พนักงานเรียกดูรายการคำถามที่พบบ่อย
16.	version		
16.1	GET	/versions/{userType}	เรียกดูเวอร์ชั่นล่าสุดของแอปพลิเคชัน

ตารางที่ 3.10 (ต่อ)

No.	API		Description
17.	**rider**		
17.1	POST	/riders/register	พนักงานสมัครสมาชิก
17.2	POST	/riders/login	พนักงานลงชื่อเข้าสู่ระบบ
17.3	POST	/riders/forgotPass	พนักงานลืมรหัสผ่าน
17.4	POST	/riders/updateProfile	พนักงานแก้ไขข้อมูลส่วนตัว
17.5	GET	/riders/	เรียกดูรายการพนักงาน
17.6	GET	/riders/income/day	พนักงานเรียกดูข้อมูลรายได้แบบรายวัน
17.7	GET	/riders/income/{filter}	พนักงานเรียกดูข้อมูลรายได้ตามเงื่อนไข
17.8	GET	/riders/income/{filter}/{date}	พนักงานเรียกดูข้อมูลรายได้โดยวันที่
17.9	POST	/riders/resetPass	พนักงานขอเปลี่ยนรหัสผ่าน
17.10	POST	/riders/changePass	พนักงานเปลี่ยนรหัสผ่าน
17.11	POST	/riders/setNewPass	พนักงานตั้งค่ารหัสผ่าน
17.12	POST	/riders/logout	พนักงานลงชื่อออกจากระบบ
17.13	POST	/riders/changeMobileNo	พนักงานเปลี่ยนหมายเลขโทรศัพท์
17.14	POST	/riders/updateMobileNo	พนักงานแก้ไขข้อมูลหมายเลขโทนศัพท์
17.15	POST	/riders/topup/{paymentOptionId}	พนักงานแจ้งโอนเงินเติมเครดิต

ตารางที่ 3.10 (ต่อ)

No.	API		Description
18.	**rating**		
18.1	POST	/rating	ให้คะแนนพนักงาน
18.2	GET	/rating	เรียกดูข้อมูลคะแนนพนักงาน
19.	**staff**		
19.1	POST	/login	ผู้ดูแลระบบล็อกอิน
19.2	POST	/logout	ผู้ดูแลระบบล็อกเอาท์
19.3	POST	/forgotPass	ผู้ดูแลระบบลืมรหัสผ่าน
19.4	POST	/resetPass	ผู้ดูแลระบบขอรีเซ็ทรหัสผ่าน
19.5	POST	/updateProfile	ผู้ดูแลระบบอัพโหลดไฟล์
19.6	GET	/staffs	ดูข้อมูลผู้ดูแลระบบทั้งหมด
19.7	GET	/staffs/{id}	ดูรายละเอียดข้อมูลผู้ดูแลระบบแต่ละคน
19.8	POST	/staffs	เพิ่มผู้ดูแลระบบใหม่
19.9	PUT	/staffs/{id}	แก้ไขข้อมูลผู้ดูแลระบบ
19.10	DELETE	/staffs/{id}	ลบผู้ดูแลระบบ
19.11	GET	/faqs	ดูข้อมูล FAQทั้งหมด
19.12	GET	/faqs/{id}	ดูข้อมูล FAQ แต่ละข้อด้วย id
19.13	POST	/faqs	เพิ่ม FAQ ข้อใหม่
19.14	PUT	/faqs/{id}	แก้ไข FAQ แต่ละข้อ
19.15	DELETE	/faqs/{id}	ลบ FAQ แต่ละข้อ

ตารางที่ 3.10 (ต่อ)

No.	API		Description
19.16	DELETE	/faqs	ลบ FAQ แบบหลายรายการ
19.17	GET	/promotions	ดุข้อมูลโปรโมชันทั้งหมด
19.18	GET	/promotions/histories	ดูประวัติการใช้สิทธิ์โปรโมชัน
19.19	GET	/promotions/{id}	ดูรายละเอียดโปรโมชันแต่ละอันด้วย id
19.20	POST	/promotions	เพิ่มโปรโมชันใหม่
19.21	PUT	/promotions/{id}	แก้ไขโปรโมชัน
19.22	DELETE	/promotions	ลบโปรโมชัน
19.23	GET	/taxInvoices	ดูรายการร้องขอใบกำกับภาษีทั้งหมด
19.24	GET	/taxInvoices/{id}	ดูรายละเอียดข้อมูลใบกับกับภาษีด้วย id
19.25	POST	/taxInvoices/{id}	สร้างใบกำกับภาษีใหม่
19.26	GET	/serviceAreas	ดูพื้นที่ให้บริการทั้งหมด
19.27	GET	/serviceAreas/all	ดูพื้นที่ให้บริการทั้งหมดสำหรับผู้ใช้ทุกกลุ่ม
19.28	POST	/serviceAreas	เพิ่มพื้นที่ให้บริการใหม่
19.29	PUT	/serviceAreas	แก้ไขพื้นที่ให้บริการ
19.30	DELETE	/serviceAreas	ลบพื้นที่ให้บริการ
19.31	GET	/purchase/histories	ดูข้อมูลประวัติการชำระค่าบริการผ่านบัตรเครดิตทั้งหมด

ตารางที่ 3.10 (ต่อ)

No.	API		Description
19.32	GET	/purchase/{id}	ดูข้อมูลประวัติการชำระค่าบริการผ่านบัตรเครดิตแต่ละรายการด้วย id
19.33	GET	/purchase/{paymentChannel}/{status}	ดูข้อมูลประวัติการชำระค่าบริการผ่านบัตรเครดิตตามสถานะที่ทำรายการ
19.34	POST	/purchase/{paymentChannel}/{status}	เพิ่มรายการประวัติการชำระค่าบริการผ่านบัตรเครดิตพร้อมสถานะ
19.35	POST	/orders	ดูรายการคำสั่งงานทั้งหมด
19.36	POST	/orders/cancel/rider	ทำรายการยกเลิกคำสั่งงานแทนพนักงาน
19.37	POST	/orders/cancel/user	ทำรายการยกเลิกคำสั่งงานแทนผู้ใช้งานทั่วไป
19.38	GET	/orders/{id}	ดูรายละเอียดคำสั่งงานแต่ละรายการด้วย id
19.39	POST	/orders/{id}	สรางคำสั่งงานใหม่
19.40	POST	/orders/special/{orderId}	หมอบหมายงานให้พนักงาน
19.41	GET	/riders	เรียกดูรายการพนักงานทั้งหมด
19.42	GET	/riders/approve_credit/histories	เรียกดูประวัติการอนุมัติเครดิตของพนักงาน
19.43	GET	/riders/topup	ดูรายการขออนุมัติเครดิตทั้งหมด
19.44	GET	/riders/topup/{id}	ดูรายละเอียดการขออนุมัติเครดิตด้วย id

ตารางที่ 3.10 (ต่อ)

No.	API		Description
19.45	PUT	/riders/topup/{id}	อัพเดทสถานะการอนุมัติเครดิต
19.46	GET	/riders /{id}	เรียกดูข้อมูลพนักงานทั้งหมด
19.47	PUT	/riders /{id}	แก้ไขข้อมูลพนักงานด้วย id
19.48	GET	/riders/income/{id}	เรียกดูข้อมูลประวัติรายได้ของพนักงานแต่ละคน
19.49	GET	/riders/document/{id}/{passcode}	เรียกดูข้อมูลเอกสารของพนักงานด้วย id และ รหัสผ่าน
19.50	POST	/riders/document/{id}/{passcode}	เพิ่มข้อมูลเอกสารของพนักงานด้วย id และ รหัสผ่าน
19.51	GET	/serviceItems	ดูบริการเสริมทั้งหมด
19.52	GET	/serviceItems/{id}	ดูข้อมุลบริการเสริมแต่ละรายการด้วย id
19.53	POST	/serviceItems	เพิ่มบริการเสริมใหม่
19.54	PUT	/serviceItems/{id}	แก้ไขบริการเสริมแต่ละรายการด้วย id
19.55	DELETE	/serviceItems	ลบบริการเสริมแบบหลายรายการ
19.56	DELETE	/serviceItems/{id}	ลบบริการเสริมด้วย id
19.57	GET	/users	เรียกดูข้อมูลผุ้ใช้งานทั้งหมด
19.58	GET	/users/{id}	เรียกดูข้อมูลผุ้ใช้งานด้วย id
19.59	GET	/users/expense/{id}	เรียกดูข้อมูลค่าใช้จ่ายของผู้ใช้แต่ละคนด้วย id
19.60	POST	/media/upload	อัพโหลดไฟล์

ตารางที่ 3.10 (ต่อ)

No.	API		Description
19.61	GET	/global_config	เรียกดูข้อมูลการตั้งค่าของระบบ
19.63	PUT	/global_config/{id}	แก้ไขข้อมูลการตั้งค่าของระบบ
19.63	GET	/logs/clear	การลบ log

3.2.5.2 WebSocket Event

มีการออกแบบ WebSocket Event เพื่อใช้สำหรับการรับส่งข้อมูลแบบเรียลไทม์ตามตารางที่ 3.11 และสามารถดูรายละเอียดของการเชื่อมต่อสำหรับแต่ละกลุ่มผู้ใช้งานได้ในภาคผนวก

ตารางที่ 3.11 WebSocket Event สำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์

No.	API		Description
1.	**User**		
1.1	Register event	userTrackRiderLocationByOrder	ติดตามการทำงานของพนักงานที่กำลังทำคำสั่งงานอยู่
2.	**Messenger**		
2.1	Emit event	riderUpdateLocation	พนักงานส่งตำแหน่ง location ปัจจุบันมาอัพเดท
2.2	Register event	newConfirmOrder	รับ Feed งานที่มีการสั่งงานเข้ามาใหม่
2.3	Register event	removeAvailableOrder	ลบคำสั่งงานที่ถูกรับไปแล้วออกจากหน้า feed
3.	**Admin**		
3.1	Register event	trackOnlineRidersLocation	ผู้ดูแลระบบติดตามการทำงานของพนักงาน
4.	**System**		
4.1	Emit event	confirmOrder	เมื่อมีการสั่งงานเข้ามาใหม่ระบบส่งงานให้พนักงานที่ออนไลน์อยู่
4.2	Emit event	removeAvailableOrder	เมื่อมีพนักงานรับงานไปแล้ว ระบบ broadcast บอกพนักงานทุกคนที่ออนไลน์อยู่ว่าให้ลบคำสั่งงานนี้ออกจากหน้า Feed

3.3 เครื่องมือการพัฒนาระบบ

เว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์ พัฒนาโดยใช้เครื่องมือต่าง ๆ ดังในตารางที่ 3.12 และ 3.13

ตารางที่ 3.12 อุปกรณ์ฮาร์ดแวร์ที่ใช้ในการพัฒนาระบบ

ลำดับ	รายการ	รายละเอียด
1.	เครื่องคอมพิวเตอร์แม่ข่าย จำนวน **2** เครื่อง ซึ่งมีคุณสมบัติดังต่อไปนี้	**Application Server** CPU 8 core RAM 16GB Hard disk 100 GB ระบบปฏิบัติการภายในเป็น CentOS **7.**x **Database Server** CPU 8 core**.** RAM 16GB Hard disk 50 GB ระบบปฏิบัติการภายในเป็น CentOS **7.**x
2.	เครื่องคอมพิวเตอร์ส่วนบุคคล จำนวน 1 เครื่อง ซึ่งมีคุณสมบัติดัง ต่อไปนี้	คอมพิวเตอร์โน้ตบุ๊ค CPU Intel core i5 @ 2.40 GHz RAM 4.00 GHz Hard disk 500 GB ระบบปฏิบัติการภายในเป็น Windows 10 Pro 64 bit

ตารางที่ 3.13 ซอฟต์แวร์ที่ใช้ในการพัฒนาระบบ

ลำดับ	รายการ	รายละเอียด
1.	Node.js V 6.6.6	ทำหน้าที่เป็นเว็บเซิร์ฟเวอร์
2.	mongoDB version v3.4.2	ใช้เป็นฐานข้อมูล
3.	Robomongo version 0.9.0	ใช้สำหรับเป็นระบบจัดการฐานข้อมูล
4.	WebStorm version 2016.3.2	ใช้เป็นเครื่องมือในการพัฒนาเว็บ API และเว็บไซต์
5.	Web browser คือ IE Firefox Chrome	ใช้ทดสอบเว็บแอปพลิเคชันและ WebSocket
6.	Postman	ใช้เป็นเครื่องมือสำหรับทดสอบ Web APIs

3.4 ระยะเวลาการดำเนินการ

เว็บเซอร์วิสสำหรับระบบบริการขนส่งของและติดตามการขนส่งแบบเรียล์ไทม์ เริ่มศึกษาและพัฒนา ดังตารางที่ 3.14

ตารางที่ 3.14 ระยะเวลาการออกแบบและพัฒนาเว็บเซอร์วิสสำหรับระบบบริการขนส่งของและติดตามการขนส่งแบบเรียล์ไทม์

ที่	แผนดำเนินงาน	สิงหาคม 2559 – มิถุนายน 2560										
		1	2	3	4	5	6	7	8	9	10	11
1.	ศึกษาปัญหาและความต้องการของระบบ	←	→									
2.	ศึกษาเครื่องมือและภาษาที่ใช้ในการพัฒนาระบบ	←	→									
3.	ออกแบบระบบ		←	→								
4.	พัฒนาระบบ			←	→	→	→	→				
5.	ทดสอบและแก้ไขข้อผิดพลาด				←	→	→	→	→	→		
6.	จัดทำคู่มือ เอกสารที่เกี่ยวข้อง		←	→	→	→	→	→	→	→	→	→

บทที่ 4
ผลการดำเนินงาน

ในบทนี้จะกล่าวถึงผลดำเนินการพัฒนาเว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์

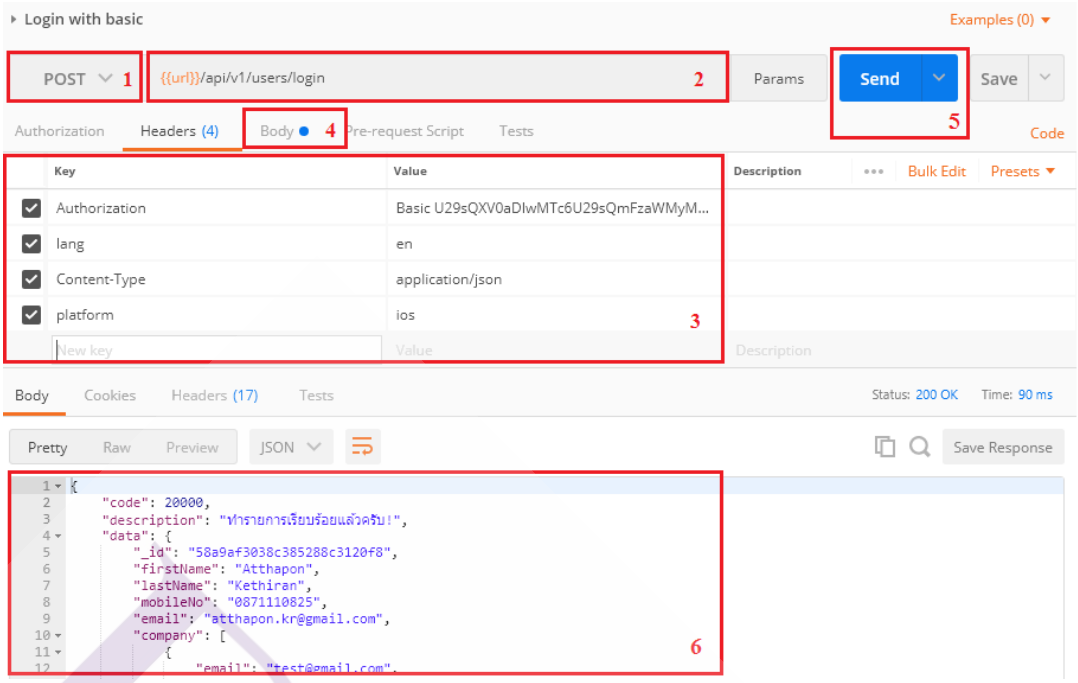
4.1 วิธีการทดสอบการพัฒนาระบบ

วิธีการทดสอบผลดำเนินการพัฒนาแบ่งเป็นสองส่วนคือ การทำ Unit Test โดยใช้เครื่องมือช่วยในการทดสอบ คือ Postman และทดสอบเชื่อมต่อ WebSocket โดยสร้างเว็บแอปพลิเคชันง่ายๆ ขึ้นมาเพื่อเชื่อมต่อและรับส่งข้อมูล และทำ Integrate Test เข้ากับแอปพลิเคชันของผู้ใช้ในฝั่งไคลเอนท์ โดยใช้กรณีทดสอบ ดังในตารางที่ 4.1 โดยมีขั้นตอนการทดสอบด้วย Postman แสดงตัวอย่างในภาพที่ 4.1 ดังต่อไปนี้

1. ระบุรูปแบบ Method ในการทำงาน
2. ใส่ที่อยู่ URL ของบริการที่ต้องการเรียกใช้งาน
3. กำหนดรูปแบบของข้อมูล(Content-Type) ที่ต้องการส่งในส่วนของ Header
4. ใส่ค่าของข้อมูลที่ต้องการส่ง
5. กดปุ่ม SEND
6. ดูผลลัพธ์การทำงาน

ในส่วนของการทดสอบ WebSocket แสดงตัวอย่างในภาพที่ 4.2 โดยมีวิธีการดังนี้คือ

1. ระบุ Token เพื่อใช้เป็น parameter ในการ Authen
2. กดปุ่ม Re-Connect เพื่อส่งคำสั่งเชื่อมต่อ
3. หลังจากการเชื่อมต่อสำเร็จแล้ว เลือก Event ที่จะ Emit ข้อมูลส่งไปพร้อมกับ ระบุข้อมูลที่ต้องการส่ง
4. กดส่งข้อมูลกดปุ่ม SEND
5. ดูผลลัพธ์การทำงาน และข้อมูลที่ได้รับ



ภาพที่ 4.1 ตัวอย่างหน้าจอแสดงขั้นตอนการทดสอบด้วย Postman

ภาพที่ 4.2 หน้าเว็บแอปพลิเคชันที่สร้างขึ้นมาเพื่อทำการทดสอบเชื่อมต่อ WebSocket

และในส่วนของการทำ Inegrate Test จะแสดงภาพหน้าจอผลลัพธ์จากการเรียกใช้งานเซอร์วิสจากแอปพลิเคชันในฝั่งไคลเอนท์ทั้งในส่วนของผู้ใช้งานทั่วไป พนักงานส่งของ และผู้ดูแลระบบตาม Test Case ในตารางที่ 4.1 เช่นเดียวกัน

ตารางที่ 4.1 Test Case ที่นำมาทดสอบการทำงานของระบบ

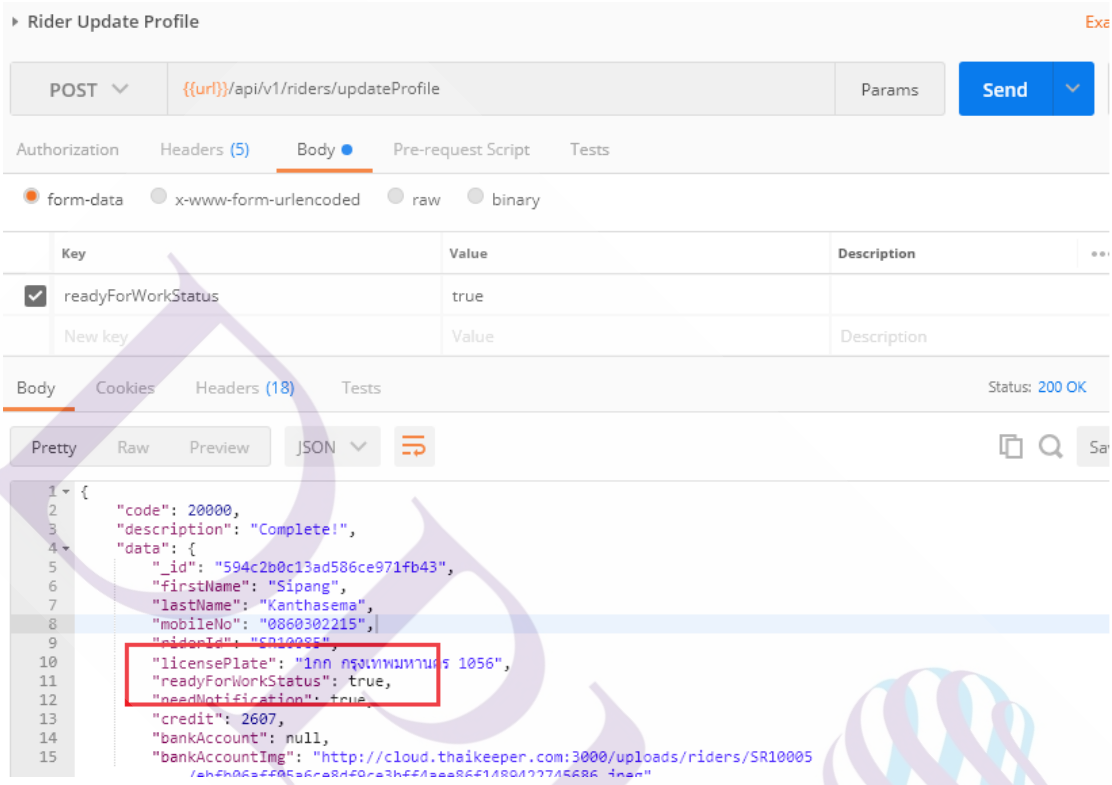
No.	Test Case ID	Test Case	Use Case ID
1.	TC01	พนักงานเปิดปิดสถานะการรับงาน	M04000
2.	TC02	พนักงานเลือกพื้นที่รับงาน	M05000
3.	TC03	สร้างงาน	U04000
4.	TC04	ชำระเงินช่องทางอื่นๆ(ยกเว้นบัตรเครดิต)	U04200
5.	TC05	พนักงานได้รับ Feed งานตามพื้นที่ที่เลือกไว้	M05100
6.	TC06	พนักงานรับงาน	M06000
7.	TC07	ดูตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานจากบนแผนที	A06300

4.2 ผลการพัฒนาระบบ

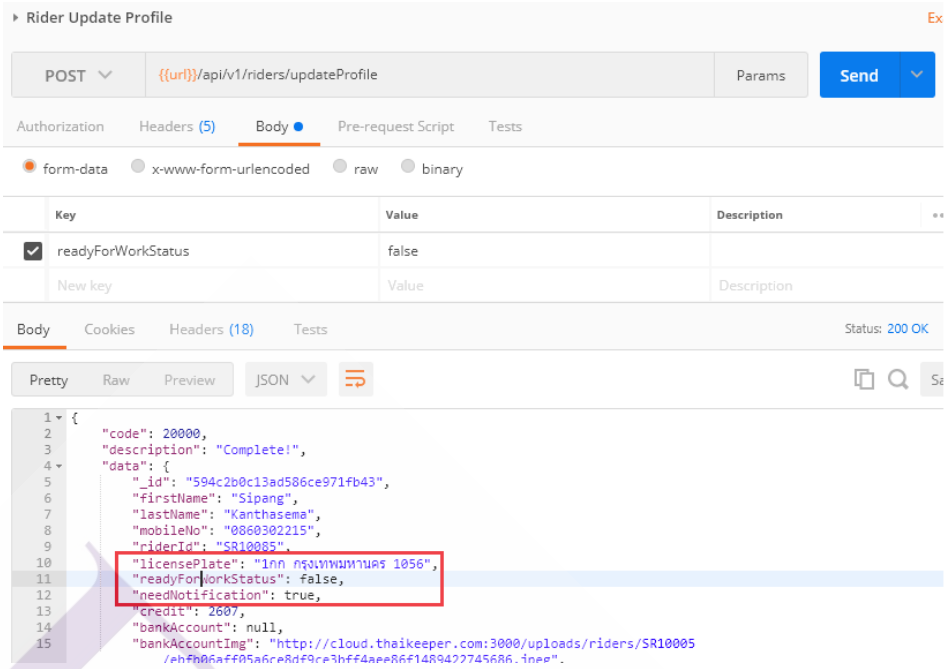
4.2.1 กรณีทดสอบ TC01

สำหรับผลการพัฒนาระบบในส่วนของพนักงานเปิดปิดดสถานะการรับงาน มีดังนี้

1. ทดสอบเรียกเซอร์วิสด้วย Postman ได้ผลลัพธ์ตามภาพที่ 4.3 และ 4.4

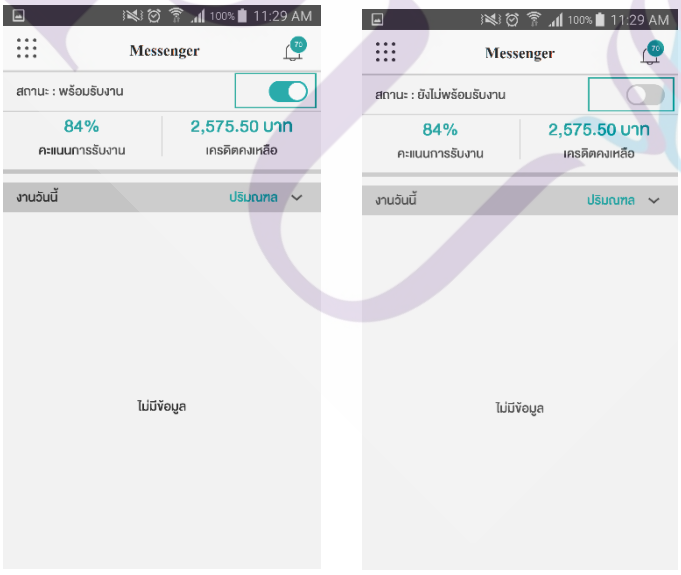


ภาพที่ 4.3 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC01 เพื่อเปิดสถานะพร้อมรับงาน โดยการทำ Unit Test



ภาพที่ 4.4 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC01 เพื่อปิดสถานะพร้อมรับงาน โดยการทำ Unit Test

2. ผลลัพธ์จากการ Integrate เข้ากับแอปพลิเคชัน

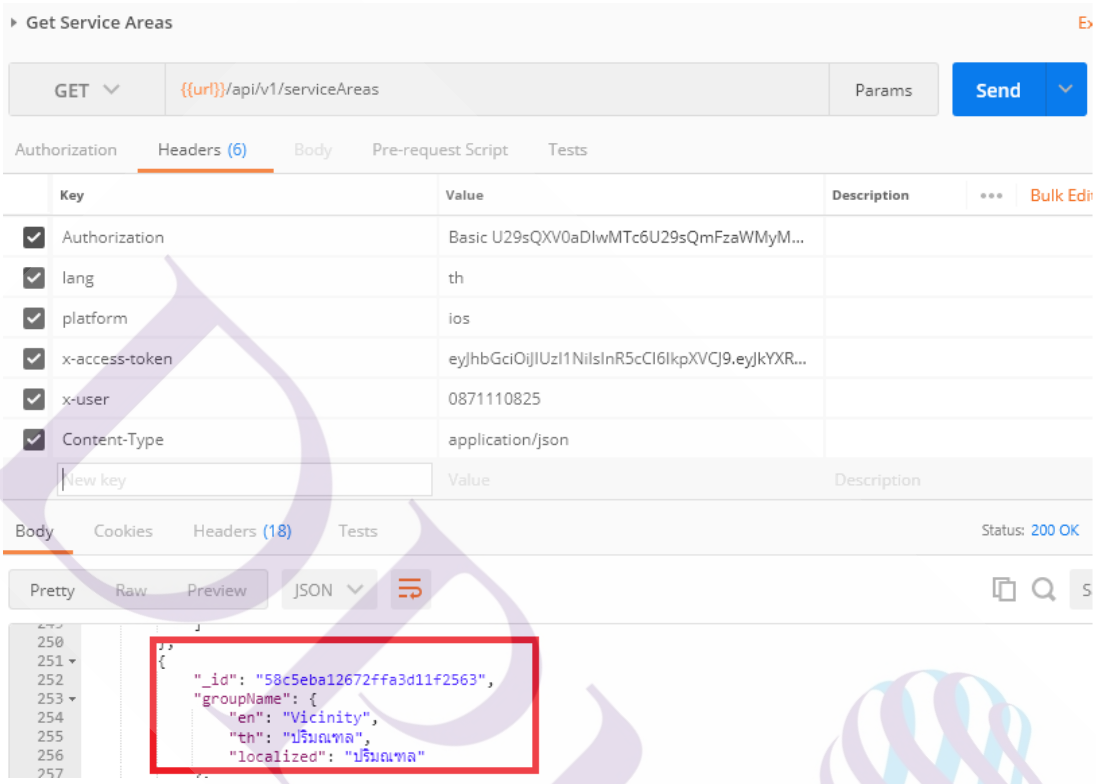


ภาพที่ 4.5 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC01 เพื่อเปิดปิดสถานะพร้อมรับงานผ่านหน้าแอปพลิเคชัน

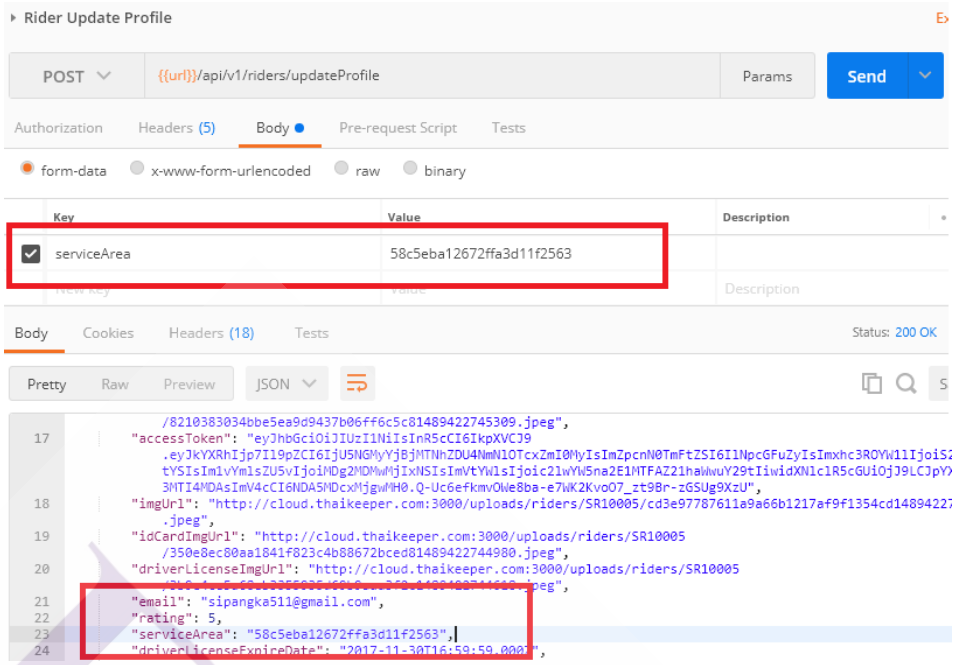
4.2.2 กรณีทดสอบ TC02

สำหรับผลการพัฒนาระบบในส่วนของพนักงานเลือกพื้นที่รับงาน มีดังนี้

1. ทดสอบเรียกเซอร์วิสด้วย Postman มีการเรียกใช้งาน 2 API ได้ผลลัพธ์ตามภาพที่ 4.6 และ 4.7

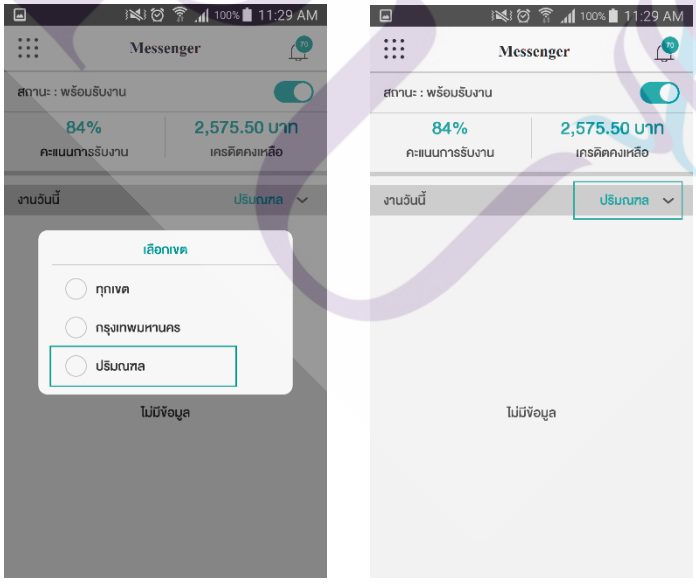


ภาพที่ 4.6 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC01 เพื่อพนักงานเลือกพื้นที่รับงาน โดยการทำ Unit Test



ภาพที่ 4.7 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC01 เพื่อพนักงานเลือกพื้นที่รับงาน โดยการทำ Unit Test

2. ผลลัพธ์จากการ Integrate เข้ากับแอปพลิเคชัน

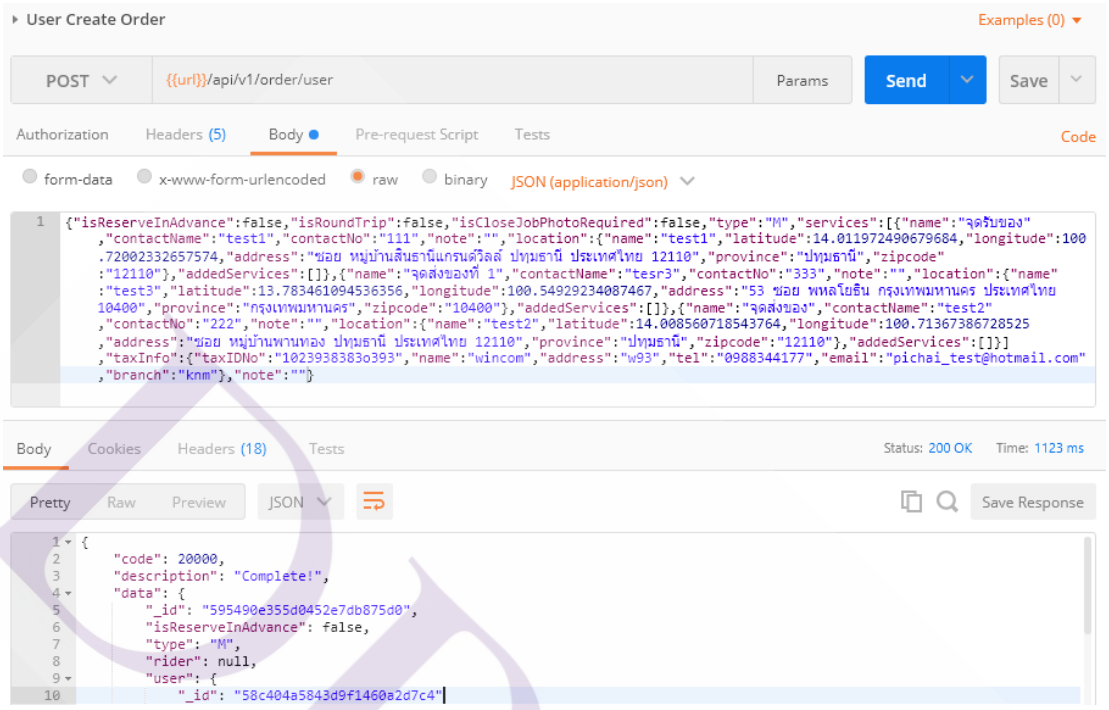


ภาพที่ 4.8 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC02 เพื่อพนักงานเลือกพื้นที่รับงานผ่านหน้าแอปพลิเคชัน

4.2.3 กรณีทดสอบ TC03

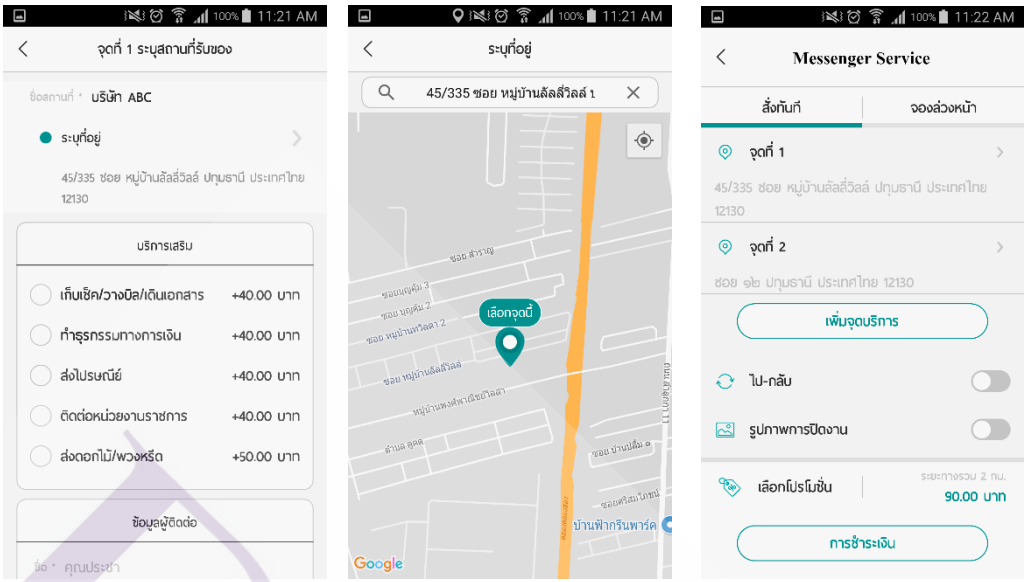
สำหรับผลการพัฒนาระบบในส่วนของผู้ใช้สร้างงาน มีดังนี้

1. ทดสอบเรียกเซอร์วิสด้วย Postman ได้ผลลัพธ์ตามภาพที่ 4.9



ภาพที่ 4.9 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC03 โดยการทำ Unit Test

2. ผลลัพธ์จากการ Integrate เข้ากับแอปพลิเคชัน

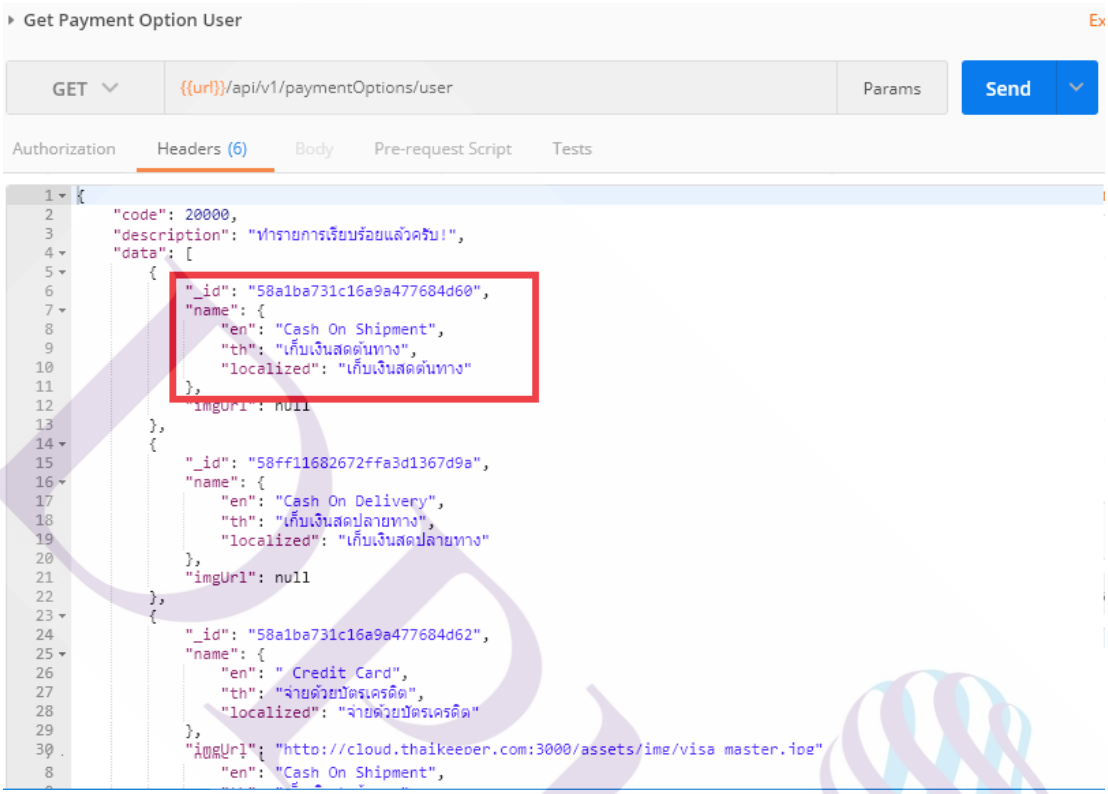


ภาพที่ 4.10 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC03 เพื่อผู้ใช้งานทั่วไปสำหรับสร้างงานผ่านหน้าแอปพลิเคชัน

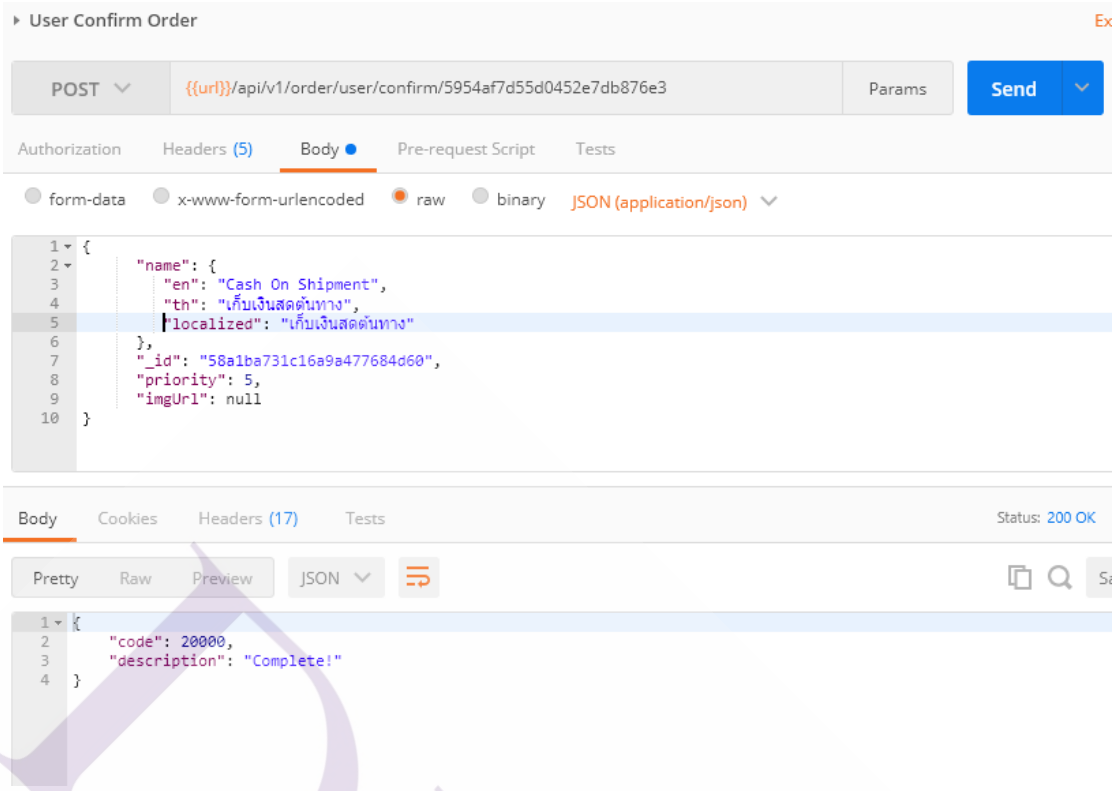
4.2.4 กรณีทดสอบ TC04

สำหรับผลการพัฒนาระบบในส่วนของผู้ใช้ชำระเงินผ่านช่องทางอื่นๆ(ยกเว้นบัตรเครดิต) มีดังนี้

1. ทดสอบเรียกเซอร์วิสด้วย Postman ได้ผลลัพธ์ตามภาพที่ 4.11 และ 4.12

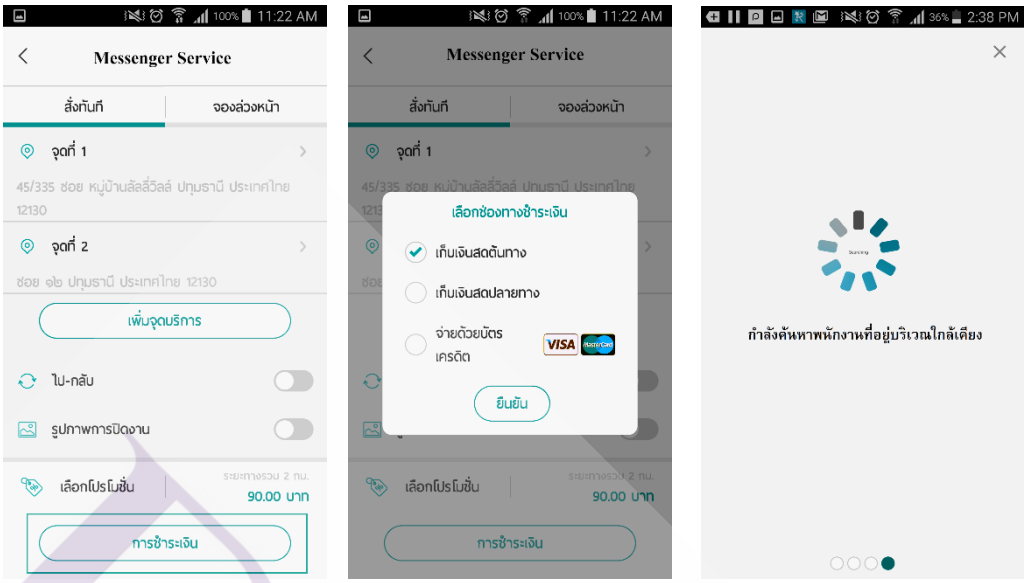


ภาพที่ 4.11 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC04 เพื่อเรียกดูช่องทางการชำระเงินทั้งหมด โดยการทำ Unit Test



ภาพที่ 4.12 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC04 เพื่อยืนยันการสร้างงานและเลือกช่องทางการชำระเงิน โดยการทำ Unit Test

2. ผลลัพธ์จากการ Integrate เข้ากับแอปพลิเคชัน

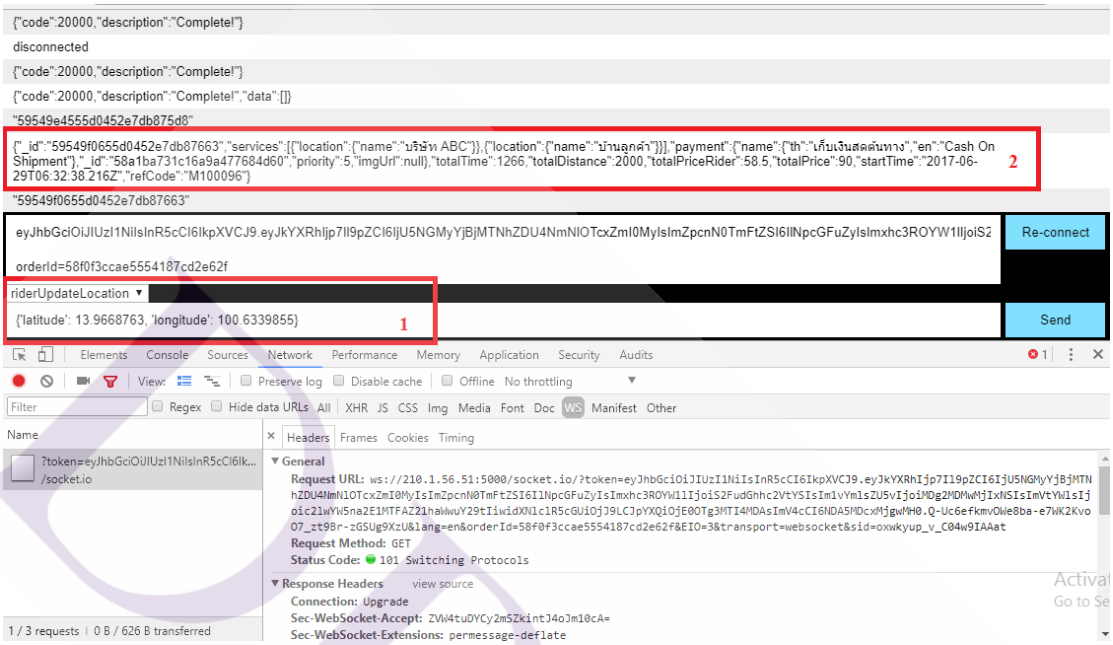


ภาพที่ 4.13 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC04 เพื่อผู้ใช้งานทั่วไปสำหรับยืนยันคำสั่งงานและเลือกช่องทางรำระค่าบริการผ่านหน้าแอปพลิเคชัน

4.2.5 กรณีทดสอบ TC05

สำหรับผลการพัฒนาระบบในส่วนของพนักงานได้รับ Feed งานตามพื้นที่ที่เลือกไว้มีดังนี้

1. ทดสอบเชื่อมต่อ WebSocket โดยสร้าง WebSocket Client ง่ายๆเพื่อเชื่อมต่อและส่งข้อมูลไปยัง WebSocket Server ได้ผลลัพธ์ตามภาพที่ 4.14



ภาพที่ 4.14 หน้าจอแสดงผลลัพธ์การเชื่อมต่อ WebSocket สำหรับกรณีทดสอบ TC05 โดยการทำ Unit Test

โดยการทดสอบทำตามขั้นตอนดังนี้คือ

1. เชื่อมต่อ WebSocket โดยส่ง token ของพนักงาน เป็น parameter เพื่อ Authen และเชื่อมต่อกับ WebSocket Server

2. Emit event riderUpdateLocation พร้อมกับส่งข้อมูล latitude longitude ไปยัง WebSocket Server ตามขั้นตอนที่ 1 ในภาพ 4.14 ลักษณะข้อมูลที่ส่งออกไปคือ {'latitude': 13.9668763, 'longitude': 100.6339855} เป็นข้อมูลตำแหน่งปัจจุบันของพนักงาน

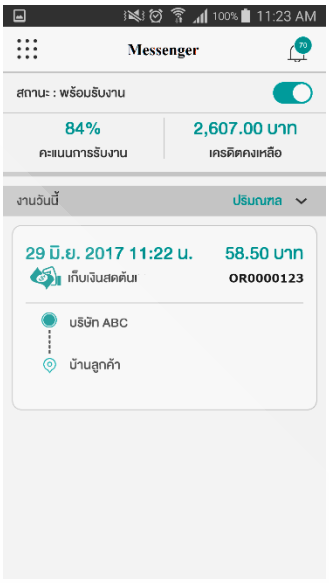
3. Register Event newConfirmOrder เพื่อรอรับข้อมูล คำสั่งงานที่จะเข้ามาตามพื้นที่บริการที่พนักงานคนดังกล่าวได้เลือกไว้ ซึ่งได้ผลลัพธ์ตามขั้นตอนที่ 2 ในภาพที่ 4.14 ข้อมูลที่ได้รับเป็น Object ข้อมูลคำสั่งงานมีลักษณะข้อมูลดังนี้คือ

ตารางที่ 4.2 ตัวอย่างข้อมูลที่ได้รับผ่าน WebSocket ใน Event newConfirmOrder


```
{
 "_id": "59549f0655d0452e7db87663",
 "services": [
  {
   "location": {
    "name": "บริษัท ABC"
   }
  },
  {
   "location": {
    "name": "บ้านลูกค้า"
   }
  }
 ],
 "payment": {
  "name": {
   "th": "เก็บเงินสดต้นทาง",
   "en": "Cash On Shipment"
  },
  "_id": "58a1ba731c16a9a477684d60",
  "priority": 5,
  "imgUrl": null
 },
 "totalTime": 1266,
 "totalDistance": 2000,
 "totalPriceMessenger": 58.5,
 "totalPrice": 90,
 "startTime": "2017-06-29T06:32:38.216Z",
 "refCode": "OR100094"
}
```

2. ผลลัพธ์จากการ Integrate เข้ากับแอปพลิเคชัน

พนักงานจะได้รับ Feed งานเข้ามาตามพื้นที่บริการที่ได้เลือกไว้โดยอัตโนมัติ

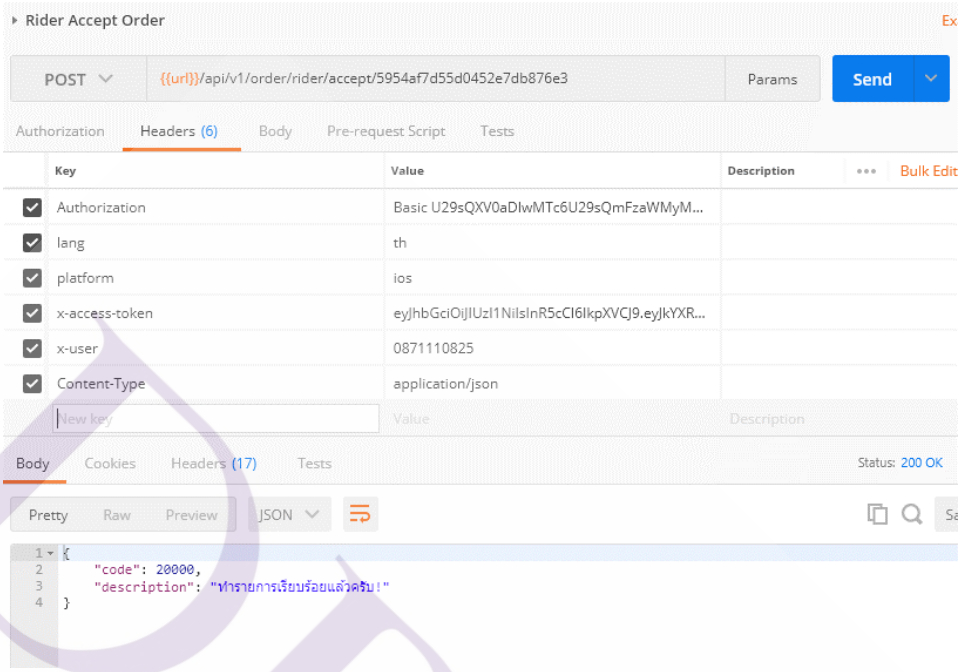


ภาพที่ 4.15 หน้าจอแสดงผลลัพธ์เชื่อมต่อ WebSocket สำหรับกรณีทดสอบ TC05 เพื่อพนักงานได้รับ Feed งานตามพื้นที่ที่เลือกไว้ผ่านหน้าแอปพลิเคชัน

4.2.6 กรณีทดสอบ TC06

สำหรับผลการพัฒนาระบบในส่วนของพนักงานรับงานที่เข้ามาในหน้า Feed มีดังนี้

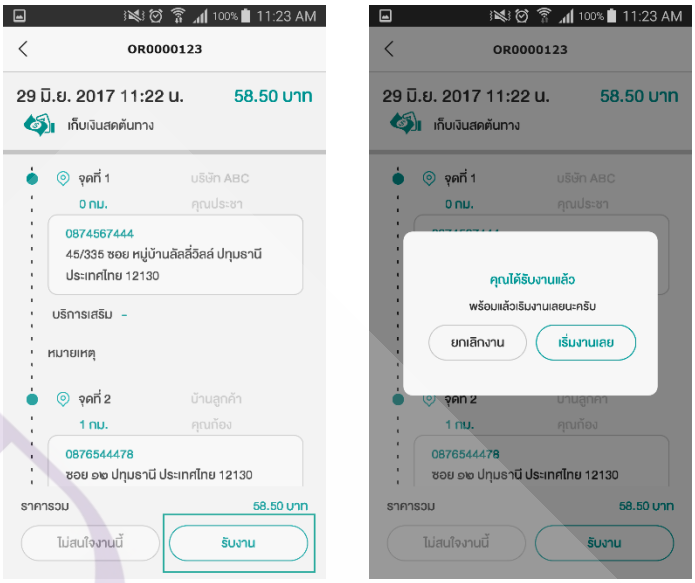
1. ทดสอบเรียกเซอร์วิสด้วย Postman ได้ผลลัพธ์ตามภาพที่ 4.16



ภาพที่ 4.16 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC06 โดยการทำ Unit Test

2. ผลลัพธ์จากการ Integrate เข้ากับแอปพลิเคชัน

พนักงานกดเข้าไปดูรายละเอียดงานที่ได้รับจาก Feed แล้วทำการกดรับงาน



ภาพที่ 4.17 หน้าจอแสดงผลลัพธ์การเรียกเซอร์วิสสำหรับกรณีทดสอบ TC06 พนักงานรับคำสั่งงานผ่านหน้าแอปพลิเคชัน

4.2.7 กรณีทดสอบ TC07

สำหรับผลการพัฒนาระบบในส่วนของผู้ดูแลระบบดูตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานจากบนแผนที่มีดังนี้

1. ทดสอบเชื่อมต่อ WebSocket โดยสร้าง WebSocket Client ง่ายๆเพื่อเชื่อมต่อและส่งข้อมูลไปยัง WebSocket Server ได้ผลลัพธ์ตามภาพที่ 4.18

ภาพที่ 4.18 หน้าจอแสดงผลลัพธ์การเชื่อมต่อ WebSocket สำหรับกรณีทดสอบ TC07 โดยการทำ Unit Test

โดยการทดสอบทำตามขั้นตอนดังนี้คือ

1. เชื่อมต่อ WebSocket โดยส่ง token ของผู้ดูแลระบบ เป็น parameter เพื่อ Authen และเชื่อมต่อกับ WebSocket Server ตามขั้นตอนที่ 1 ในภาพ 4.18

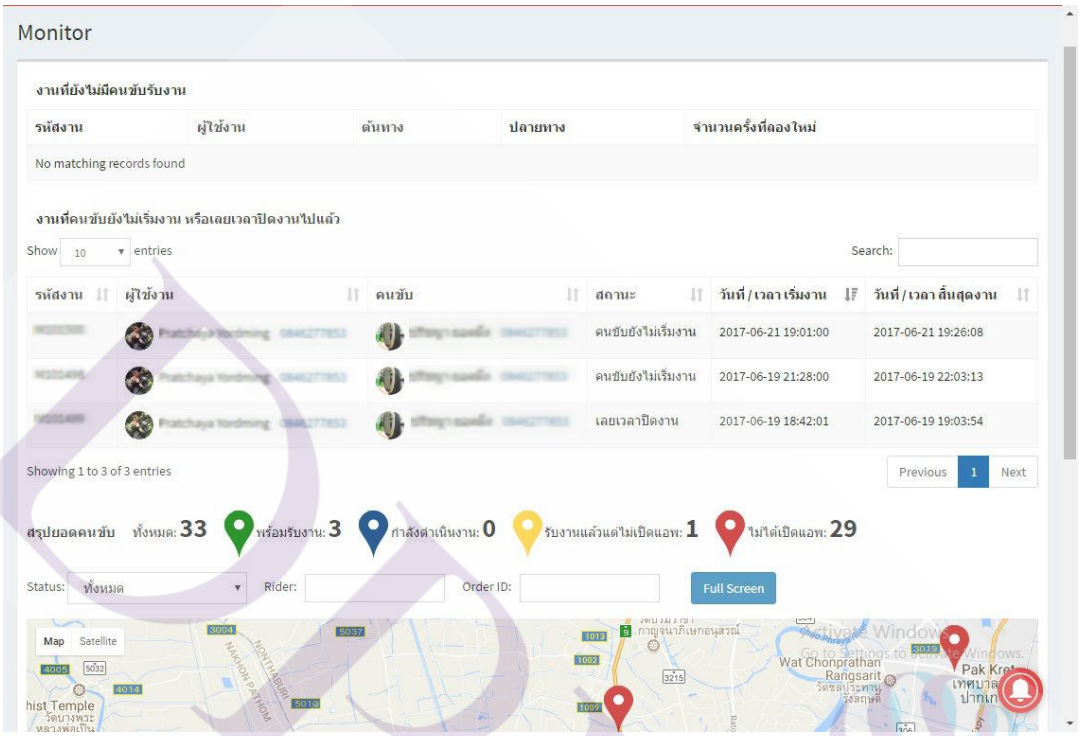
2. Register Event trackOnlineRidersLocation เพื่อรอรับข้อมูลตำแหน่งและสถานะการรับงานของพนักงานที่เปิดสถานะพร้อมรับงานไว้ คำสั่งงานที่จะเข้ามาตามพื้นที่บริการที่พนักงานคนดังกล่าวได้เลือกไว้ ซึ่งได้ผลลัพธ์ตามขั้นตอนที่ 2 ในภาพที่ 4.18 ข้อมูลที่ได้รับเป็น Array Object ข้อมูลตำแหน่งปัจจุบันของพนักงานที่เปิดสถานะพร้อมรับงานอยู่ในระบบ ข้อมูลที่ได้รับมีลักษณะดังนี้คือ

ตารางที่ 4.3 ตัวอย่างข้อมูลที่ได้รับผ่าน WebSocket ใน Event trackOnlineRidersLocation

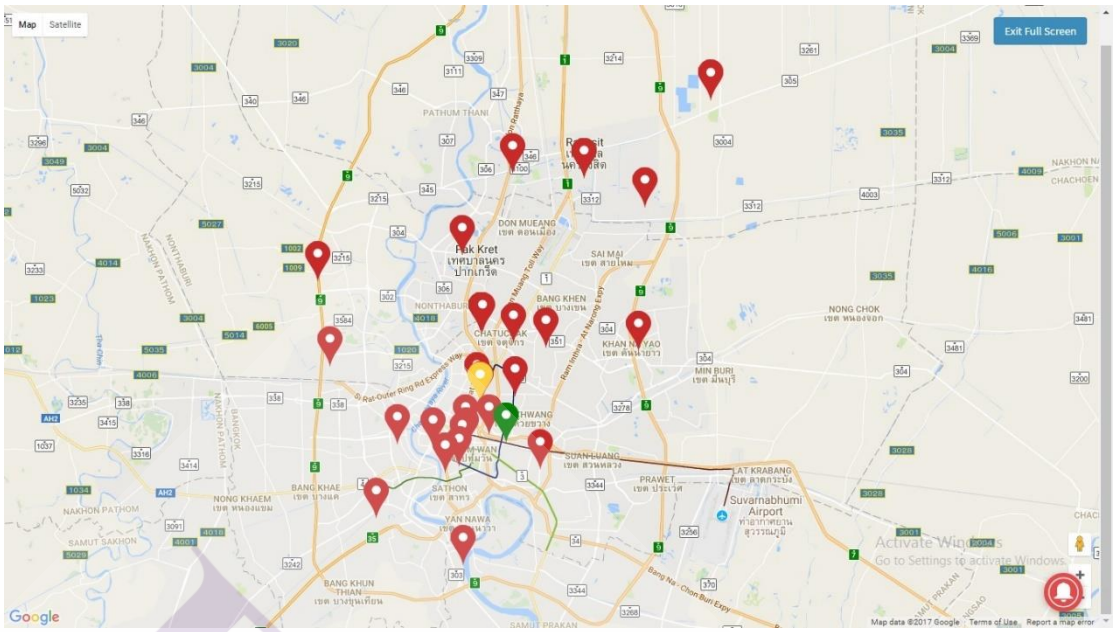

```
[
 {
  "_id": "5954821813ad586ce987d36b",
  "riderId": {
   "_id": "594c2b0c13ad586ce971fb43",
   "firstName": "Sipang",
   "lastName": "Kanthasema",
   "mobileNo": "0860302215",
   "riderId": "MR10085",
   "readyForWorkStatus": true,
   "imgUrl":
"http://cloud.thaikeeper.com:3000/uploads/riders/SR10005/cd3e97787611a9a66b1217af9f1354cd1489422744088.jpeg",
   "email": "sipangka511@gmail.com"
  },
  "lastOnline": "2017-06-29T08:14:29.156Z",
  "workingStatus": {
   "endTime": "2017-06-29T09:46:16.280Z",
   "startTime": "2017-06-29T07:51:44.280Z",
   "isReserveInAdvance": false,
   "orderId": {
    "_id": "5954af7d55d0452e7db876e3",
    "refCode": "OR100097"
   },
   "status": "doing"
  },
  "isOnline": false,
  "location": {
   "type": "Point",
   "coordinates": [
    100.6339855,
    13.9668763
   ],
   "latitude": 13.9668763,
   "longitude": 100.6339855
  },
  "lastUpdate": "2017-06-29T06:31:19.727Z"
 }
]
```

2. ผลลัพธ์จากการ Integrate เข้ากับแอปพลิเคชัน

ในหน้าเว็บสำหรับผู้ดูแลระบบ จะได้รับข้อมูลตำแหน่งปัจจุบันของพนักงานพร้อมกับสถานะการรับงานนำมาแสดงบนหน้าแผนที่เพื่อสามารถติดตามการทำงานของพนักงานได้ ตามภาพที่ 4.19



ภาพที่ 4.19 หน้าจอแสดงผลลัพธ์เชื่อมต่อ WebSocket สำหรับผู้ดูแลระบบ



ภาพที่ 4.20 หน้าจอแสดงผลลัพธ์การเชื่อมต่อ WebSocket สำหรับกรณีทดสอบ TC07 ผู้ดูแลระบบดูตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานจากบนแผนที่

บทที่ 5
สรุปอภิปรายผลการศึกษาและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงข้อสรุปจากการออกแบบและพัฒนาเว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบออนไลน์ รวมทั้งปัญหาและอุปสรรค และข้อเสนอต่าง ๆ ดังรายละเอียดต่อไปนี้

5.1 สรุปและอภิปรายผล

ในการดำเนินการเพื่อการออกแบบและพัฒนาเว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์ จากการวิเคราะห์รวบรวมข้อมูลและออกแบบระบบรวมถึงขั้นตอนการพัฒนาระบบ ทำให้ได้ระบบที่ผู้ใช้สามารถใช้บริการส่งคำสั่งงานไปยังพนักงานที่ออนไลน์อยู่เพื่อดำเนินงานตามคำสั่งงานได้ และผู้ใช้รวมถึงผู้ดูแลระบบยังสามารถติดตามการทำงานและทราบตำแหน่งปัจจุบันของพนักงานได้แบบเรียลไทม์จากหน้าแผนที่

จากการออกแบบและพัฒนาระบบรวมถึงการนำมาใช้งานถือว่าประสบความสำเร็จ เว็บเซอร์วิสและ WebSocket สามารถทำงานได้และตรงตามวัตถุประสงค์การของการจัดทำ และตรงตามความต้องการของผู้ใช้ที่ได้ทำการสำรวจความต้องการไว้ในตอนต้นโดยมีรายละเอียดดังนี้

1. ผู้วิจัยได้ศึกษาปัญหาและได้ศึกษาเทคโนโลยีที่เกี่ยวข้องและได้นำเทคโนโลยีที่เหมาะสมมาประยุกต์ใช้กับการพัฒนาเว็บเซอร์วิสสำหรับระบบบริการรับส่งของและติดตามการทำงานแบบเรียลไทม์
2. ผู้วิจัยได้นำความรู้และทฤษฎีที่เกี่ยวข้องที่ได้ศึกษา มาทำการวิเคราะห์และออกแบบระบบงานทำให้เว็บเซอร์วิสที่พัฒนาขึ้นมีประสิทธิภาพและได้ระบบที่ตรงกับความต้องการธุรกิจประเภทโลจิสติกที่มีความต้องการในการติดตามการนำส่งของหรือพัสดุ ทำให้ผู้ใช้มั่นใจในการเลือกใช้บริการ
3. เว็บเซอร์วิสและ WebSocket ที่พัฒนาขึ้นสามารถทำงานร่วมกับแอปพลิชันในฝั่งไคลเอนท์ได้เป็นอย่างดี สามารถรับส่งข้อมูลไปมาระหว่างกันได้ และจากการนำ Framework ในการออกแบบ API อย่าง OpenAPI Specification เข้ามาช่วยในการพัฒนาระบบ พบว่าสามารถอำนวยความสะดวกในการพัฒนาระบบในฝั่งไคลเอนท์ได้เป็นอย่างดี สามารถลดข้อผิดพลาดในการทำงานร่วมกันระหว่างเซอร์เวอร์และไคลเอนท์ และลดเวลาในการพัฒนาได้มากเนื่องจากโค้ดในส่วนของการเชื่อมต่อกับ API สามารถใช้ Swagger Codegen ช่วยสร้างออกมาได้

5.2 ปัญหาและอุปสรรค

1. เนื่องจากความต้องการจากผู้ใช้มีการเปลี่ยนแปลงและเพิ่มเติมระหว่างการพัฒนา ทำให้ในบางครั้งต้องกลับไปทบทวนและออกแบบระบบใหม่ในส่วนที่มีการเปลี่ยนแปลงทำให้ส่งผลกระทบต่อการพัฒนาระบบทั้งในส่วนของเซอร์เวอร์และไคลเอนท์ ทำให้การพัฒนาล่าช้าออกไปไม่ตรงตามแผนการที่วางไว้

2. เนื่องจากมีการนำ OpenAPI Specification มาใช้ในการออกแบบ API และช่วยในการพัฒนาระบบในฝั่งไคลเอนท์ ซึ่งถือว่าเป็นเรื่องที่ค่อนข้างใหม่ ทำให้นักพัฒนาทั้งในฝั่งเซอร์เวอร์และไคลเอนท์ต้องใช้เวลาในการศึกษาลองผิดลองถูกอยู่พอสมควรในช่วงเริ่มต้น

3. และเนื่องนำ OpenAPI Specification มาใช้ และมีการส่งไฟล์ API Spec นี้ให้กับนักพัฒนาระบบในฝั่งไคลเอนท์นำไปใช้กับเครื่องมือ Swagger Codegen ในการสร้างชุดคำสั่งในการเรียกใช้ API ดังนั้นเมื่อมีการเปลี่ยนแปลงแก้ไข หรือเพิ่ม API ใหม่ รวมถึงเมื่อมีการสร้าง Spec ที่ผิดพลาดจำเป็นต้องมีการแก้ไขไฟล์ Spec นี้และส่งให้นักพัฒนาในฝั่งไคลเอนท์ใหม่ทุกรอบ เพื่อทำการสร้างชุดคำสั่งหรือโค้ดชุดใหม่ขึ้นมาทั้งชุด

4. และเนื่องจากชุดคำสั่งที่ได้จาก Swagger Codegen ยังไม่รองรับการทำงานในบางเรื่องหรือบางจุดทำให้นักพัฒนาต้องทำการดัดแปลงแก้ไขโค้ดที่ได้จากการ Generate ขึ้นมาแทรกเข้าไปซึ่งทำให้เกิดปัญหาต่อเนื่องจากการอัพเดท ชุด API ใหม่และต้องส่งไฟล์ Spec ให้นักพัฒนานำไปสร้างชุดคำสั่งใหม่ นักพัฒนาต้องจำจุดที่เคยปรับแก้แทรกโค้ดคำสั่งเข้า ไปแล้วทำกับโค้ดชุดใหม่อีกรอบ เช่น ปัญหาในการ สร้างโค้ดไคลเอนท์เป็นภาษา Swift ที่ใช้งานบน iOS นักพัฒนาพบว่าโค้ดคำสั่งที่ได้จาก Swagger Codegen ไม่รองรับ format date ในแบบที่เซอร์เวอร์ส่งข้อมูลมาให้ด้วย format "yyyy-MM-dd'T'HH:mm:ss.SSS'Z'" ทำให้ต้องแทรก format นี้เข้าไปเองเพิ่มเติมจากที่รองรับคือ

```
let formatters = [
        "yyyy-MM-dd",
        "yyyy-MM-dd'T'HH:mm:ssZZZZZ",
        "yyyy-MM-dd'T'HH:mm:ss.SSSZZZZZ",
        "yyyy-MM-dd'T'HH:mm:ss'Z'",
        "yyyy-MM-dd'T'HH:mm:ss.SSS",
        "yyyy-MM-dd HH:mm:ss"
        ]
```

5.3 ข้อเสนอแนะ

ข้อเสนอแนะและส่วนที่ควรปรับปรุงของระบบมีดังนี้

1. ปัจจุบันระบบทำงานผ่าน HTTP ปกติ ซึ่งทำให้เห็นว่ายังขาดเรื่องการรักษาความปลอดภัยของระบบ ซึ่งต่อไปเมื่อ Server มีความพร้อม และใช้ Domain Name แทนการเชื่อมต่อเข้ามาโดยใช้ IP ก็สามารถติดตั้ง Certificate และใช้การเชื่อมต่อผ่าน HTTPS ได้

2. ในส่วนของปัญหาที่เกิดจากผู้ใช้งานในส่วนของผู้ดูแลระบบมีความต้องการเพิ่มขึ้นมาภายหลังว่าต้องการจะ Monitor ติดตามคำสั่งงานที่ถึงเวลาเริ่มงานแต่พนักงานยังไม่ได้กดเริ่มงาน และงานที่พนักงานไม่สามารถปิดงานตามกำหนดเวลาและเลยกำหนดปิดไปแล้ว ในหน้าจอ Monitor งาน ซึ่งส่วนนี้เป็น Requirement ที่เพิ่มเข้ามาภายหลังทำให้ไม่สามารถพัฒนาในส่วนของ WebSocket เพื่อรองรับการ Monitor งานในส่วนนี้ได้ทันจึงทำให้นักพัฒนาเว็บสำหรับผู้ดูแลระบบต้องใช้วิธีการ request ข้อมูลผ่าน AJAX เป็นช่วงเวลาเข้ามาแทนเพื่อดึงข้อมูลส่วนนี้ไปแสดง ซึ่งต่อไปก็จะพัฒนา WebSocket ในส่วนนี้เช่นเดียวกันในขั้นตอนการติดตามตำแหน่งพนักงานจากหน้าแผนที

3. ในส่วนของ Feature ของระบบอย่างเช่น การบันทึกพนักงานส่งของคนที่ชื่นชอบ หรือ Add Favorite ข้อมูลพนักงานที่ชื่นชอบในขณะนี้ยังไม่ได้นำมาใช้ประโยชน์เป็นเพียงแค่การเก็บข้อมูลไว้ ซึ่งต่อไปอาจจะพัฒนาเพิ่มเติมให้ผู้ใช้สามารถเลือกพนักงานส่งของได้เองจากรายชื่อพนักงานที่ชื่นชอบที่ได้เคยบันทึกไว้ ซึ่งต้องมีการพัฒนา API เพิ่มเติมต่อไป

4. ในธุรกิจที่คล้ายคลึงกันอย่างเช่น แอปพลิเคชัน LINE MAN มีบริการอย่างอื่นเพิ่มเติมมานอกเหนือจากบริการส่งของ เช่น บริการสั่งอาหาร หรือบริการตามสั่งอื่นๆ ซึ่งหากต้องการเพิ่มความสามารถในการแข่งขันก็สามารถพัฒนาให้มีบริการเพิ่มเติมได้เช่นเดียวกัน

บรรณานุกรม

บรรณานุกรม

ภาษาไทย

ธนัณชัย สิงห์มาตย์. *Service-Oriented Architecture: SOA.* สืบค้น จาก https://www.gotoknow.org/posts/221171

ไทยครีเอททีม. *WebSocket ตอนที่ 1 : WebSocket คืออะไร การรับส่งข้อมูลแบบ Real Time ด้วย PHP.* สืบค้น จาก http://www.thaicreate.com/php/php-websockets-real-time.html

บริษัทโกบอลไฟว์. *ความรู้ทั่วไปเกี่ยวกับ GPS.* สืบค้น จาก http://www.global5thailand.com/thai/gps.htm

ประพันธ์ ไกลฤทธิ์. (2559). *อย่าเพิ่งสร้าง APIs ถ้ายังไม่รู้จัก Swagger.* สืบค้น จาก https://medium.com/@bird.praphan/

ปรัชญา สุพัฒน์โสภณ. (2559). *เลิกเขียน RESTful API แบบแย่ๆ แล้วหันมาเขียนให้มันถูกต้องตามมาตรฐานกันดีกว่า.* สืบค้น จาก https://www.algorithmtut.com/

วิกิพีเดีย. (2559). *เจซัน.* สืบค้น จาก https://th.wikipedia.org/wiki/เจซัน

ศิรประภา ธนุการ. (2553). *เว็บเซอร์วิส (Web Service).* สืบค้น จาก http://oknation.nationtv.tv/blog/Siraprapa/2010/08/30/entry-2

สมเกียรติ ปุ้ยสูงเนิน. (2559). *REST กับ SOAP ต่างกันอย่างไร?.* สืบค้น จาก http://www.somkiat.cc/rest-vs-soap/

อุไรวรรณ คีรีทอง. (2556). *ระบบระบุตำแหน่งบนพื้นโลก (Global Positioning System : GPS).* สืบค้น จาก https://yingpew103.wordpress.com/2013/01/18/

ภาษาต่างประเทศ

Douglas K Barry, *Web Services Explained* [Online], Available: http://www.service-architecture.com/articles/web-services/ [2017, June 22].

Open API Initiative. *About Open API Initiative* [Online]. Available: https://www.openapis.org/about [2017, June 22].

OpenAPI Specification. *The OpenAPI Specification* [Online]. Available: https://github.com/OAI/OpenAPI-Specification [2017, June 22].

Roger L. Costello. *Building Web Services the REST Way* [Online]. Available:

http://www.xfront.com/REST-Web-Services.htm [2017, June 22].

Shruti M. Rakhunde, *Real Time Data Communication over Full Duplex Network Using Websocket*, IOSR Journal of Computer Science (IOSR-JCE), International Conference on Advances in Engineering & Technology, 2014, 15-19

SmartBear Software. *The Swagger documentation* [Online]. Available: http://swagger.io/docs/ [2017, June 22].

Snehal Mumbaikar. Puja Padiya, *Web Services Based On SOAP and REST Principles*, International Journal of Scientific and Research Publications, Vol. 3, Issue. 5, May 2013

Wikipedia. *The OpenAPI Specification* [Online]. Available: https://en.wikipedia.org/wiki/OpenAPI_Specification [2017, June 22].

Wikipedia. *WebSocket* [Online]. Available: https://en.wikipedia.org/wiki/WebSocket [2017, June 22].

ภาคผนวก

ภาคผนวก ก

พจนานุกรมข้อมูล (Data Dictionary)

ตาราง ก.1 ตาราง approve_credit_histories

ลำดับ	แอทริบิวต์	ชนิด	คำอธิบาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	staff	ObjectId	รหัสเจ้าหน้าที่ที่อนุมัติ	FK
3.	rider	ObjectId	รหัสพนักงานที่ทำรายการ	FK
4.	dateTime	Date	วันเวลาที่ทำรายการ	
5.	oldCredit	Number	ยอดเครดิตเดิม	
6.	newCredit	Number	ยอดเครดิตหลังอนุมัติ	

ตาราง ก.2 ตาราง banners

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	imgUrl	String	URL รูปแบนเนอร์	
3.	redirectUrl	String	URL เชื่อมไปยังหน้าข้อมูล	
4.	priority	Number	ลำดับการแสดงผล	

ตาราง ก.3 ตาราง counters

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	field	String	ชื่อข้อมูล	
3.	seq	Number	หมายเลขลำดับปัจจุบัน	

ตาราง ก.4 ตาราง faqs

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	createBy	ObjectId	รหัสเจ้าหน้าที่ที่ทำรายการ	FK
3.	updateBy	ObjectId	รหัสเจ้าหน้าที่ที่แก้ไขข้อมูล	FK
4.	userType	String	จำแนกข้อมูลสำหลับประเภทผู้ใช้งาน	
5.	active	Boolean	เปิดรายการนี้หรือไม่	
6.	deleteFlag	Boolean	ระบุรายการถูกลบ	
7.	updateDate	Date	วันเวลาที่ทำการแก้ไขข้อมูล	
8.	createDate	Date	วันเวลาที่รายการถูกสร้างขึ้น	
9.	no	Number	หมายเลขลำดับการแสดง	
10.	ans	Object	คำตอบ	
11.	ans.en	String	คำตอบภาษาอังกฤษ	
12.	ans.th	String	คำตอบภาษาไทย	
13.	question	Object	คำถาม	
14.	question.en	String	คำถามภาษาอังกฤษ	
15.	question.th	String	คำถามภาษาไทย	

ตาราง ก.5 ตาราง global_configs

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	riderPercentPrice	Number	อัตราส่วนแบ่งรายได้ของพนักงาน	
3.	order	Object	ข้อมูลสำหรับคำสั่งงาน	
4.	order.userCancelOrderWithNoFineTime	Number	ระยะเวลาที่ผู้ใช้สามารถยกเลิกงานโดยไม่เสียค่าปรับ (นาที)	
5.	order.retryLimit	Number	จำนวนครั้งที่ผู้ใช้สามารถส่งคำสั่งงานซ้ำได้หากไม่มีคนรับงาน	
6.	order.servicePoint.min	Number	จำนวนจุดต่ำสุดของสถานที่ที่สามารถเพิ่มได้ในคำสั่งงาน	
7.	order.servicePoint.max	Number	จำนวนจุดสูงสุดของสถานที่ที่สามารถเพิ่มได้ในคำสั่งงาน	
8.	order.reserveInadvance.days.min	Number	จำนวนวันต่ำสุดที่ทำรายการจองล่วงหน้าได้	
9.	order.reserveInadvance.days.max	Number	จำนวนวันสูงสุดที่ทำรายการจองล่วงหน้าได้	
10.	order.serviceFee.startPrice	Number	ค่าบริการเริ่มต้น	
11.	order.serviceFee.km2to20	Number	อัตราค่าบริการจากกิโลเมตรที่ 2 ถึงกิโลเมตนที่ 20	
12.	order.serviceFee.km21to30	Number	อัตราค่าบริการจากกิโลเมตรที่ 21 ถึงกิโลเมตนที่ 30	
13.	order.serviceFee.km31to70	Number	อัตราค่าบริการจากกิโลเมตรที่ 31 ถึงกิโลเมตนที่ 70	

ตาราง ก.5 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
14.	order.serviceFee.kmMoreThan70	Number	อัตราค่าบริการจากกิโลเมตรที่ 70 ขึ้นไป	
15.	order.serviceLocation.radius	Number	รัศมีการแจกงาน หน่วยเป็น (เมตร)	
16.	order.serviceDurationAddedTime	Number	ระยะเวลาที่บวกเพิ่มไปจากการคำนวณเวลาการทำงานในคำสั่งงาน (นาที)	
17.	order.acceptableOrderGabTime.before	Number	ระยะเวลาที่สามารถรับงานซ้อนได้ก่อนจากงานที่รับก่อนหน้า	
18.	order.acceptableOrderGabTime.after	Number	ระยะเวลาที่สามารถรับงานซ้อนได้หลังจากงานที่รับก่อนหน้า	
19.	order.roundTripAddedPrice	Number	อัตราค่าบริการเพิ่มเติมเมื่อเลือกบริการแบบไปกลับ	
20.	order.riderPricePercent	Number	อัตราส่วนแบ่งรายได้ของพนักงาน	
21.	order.riderCanStartOrder.before	Number	ระยะเวลาที่สามารถเริ่มงานได้ก่อนเวลาเริ่มงานที่ระบุในคำสั่งงาน	
22.	order.riderCanStartOrder.after	Number	ระยะเวลาที่สามารถเริ่มงานได้หลังเวลาเริ่มงานที่ระบุในคำสั่งงาน	
23.	order.riderCancelOrderBannedTime	Number	ระยะเวลาที่พนักงานจะถูกพักงานเมื่อมีการยกเลิกงาน(นาที)	
24.	order.workingHour.open	String	เวลาเปิดทำการของพนักงาน	
25.	order.workingHour.close	String	เวลาปิดทำการของพนักงาน	

ตาราง ก.5 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
26.	order.clearCreatedUnconfirmOrderTime	Number	ระยะเวลาในการลบข้อมูลคำสั่งงานที่ผู้ใช้ไม่ได้ทำการยืนยันคำสั่งงาน (นาที)	
27.	order.checkExpiredOrderTime	Number	ระยะเวลาที่คำสั่งงานหมดอายุหากไม่มีคนรับงาน(นาที)	
28.	order.nearByRiderWithinRadius	Number	ระยะทางที่จะแจกงานให้กับพนักงานบริเวณใกล้เคียงภายในรัศมี (เมตร)	
29.	order.userCancelPricePercent	Number	อัตราค่าปรับเมื่อผู้ใช้ยกเลิกงานจากค่าบริการทั้งหมด	
30.	order.riderCancelPriceAmount	Number	จำนวนเงินที่พนักงานจะถูกปรับเมื่อมีการยกเลิกงาน	
31.	bankAccount	Object	ข้อมูลบัญชีธนาคารของบริษัท	
32.	bankAccount.name	String	ชื่อบัญชี	
33.	bankAccount.accounts[].bankName	String	ชื่อบัญชี	
34.	bankAccount.accounts[].imgUrl	String	รูปภาพสัญลักษณ์ธนาคาร	
35.	bankAccount.accounts[].bankNo	String	หมายเลขบัญชี	

ตาราง ก.6 ตาราง notifications

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	message	String	ข้อความแจ้งเตือน	
3.	dateTime	Date	วันที่เวลาในการส่งข้อความแจ้งเตือน	
4.	type	String	ประเภทข้อมูลการแจ้งเตือน	
5.	content	Object	ข้อมูลในการแจ้งเตือน	
6.	content.statusMsg	String	ข้อความสถานะการแจ้งเตือน	
7.	content.orderId	ObjectId	รหัสคำสั่งงาน	FK
8.	content.imgUrl	String	รูปภาพที่แสดงในการแจ้งเตือน	
9.	content.status	String	สถานะคำสั่งงาน	
10.	content.remark	String	ข้อความหมายเหตุ	
11.	content.endLocation	String	ชื่อจุดเริ่มต้นคำสั่งงาน	
12.	content.startLocation	String	ชื่อจุดสิ้นสุดคำสั่งงาน	
13.	content.startTime	Date	เวลาเริ่มคำสั่งงาน	
14.	content.refCode	String	หมายเลขอ้างอิงคำสั่งงาน	
15.	usersRead[]	ObjectId	รหัสข้องผู้ใช้ที่อ่านการแจ้งเตือนนี้	
16.	users[]	ObjectId	รหัสข้องผู้ใช้ที่ได้รับการแจ้งเตือนนี้	

ตาราง ก.7 ตาราง online_users

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	socketId	String	รหัส socket	
3.	userId	ObjectId	รหัสผู้ใช้	
4.	userType	Number	ประเภทของผู้ใช้	
5.	timeStamp	Date	วันที่เวลาในออนไลน์ในระบบ	

ตาราง ก.8 ตาราง orders

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	status	Object	สถานะคำสั่งงาน	
3.	status.reason	String	เหตุผลในการทำรายการกับคำสั่งงาน	
4.	status.status	String	สถานะคำสั่งงาน	
5.	status.userId	ObjectId	รหัสผู้ใช้ที่สั่งงาน	
6.	status.createBy	String	ผู้ทำการเปลี่ยนแปลงสถานะคำสั่งงาน	
7.	status.signature.timeStamp	Date	เวลาที่ทำรายการ	
8.	status.signature.name	String	ชื่อผู้เซ็นปิดงาน	
9.	status.signature.image	String	รูปถ่ายเซ็นปิดงาน	
10.	status.timeStamp	Date	เวลาที่เซ็นปิดงาน	
11.	status.image	String	รูปถ่ายปิดงาน	

ตาราง ก.8 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
12.	status.note	String	หมายเหตุในการทำรายการ	
13.	isReserveInAdvance	Boolean	ระบุประเภทคำสั่งงานแบบจองล่วงหน้า	
14.	type	String	ประเภทคำสั่งงาน	
15.	user	ObjectId	รหัสผู้ใช้ที่สร้างคำสั่งงาน	
16.	serviceAreas[]	ObjectId	รหัสพื้นที่ที่คำสั่งงานครอบคลุม	
17.	createDate	Date	วันที่เวลาที่สั่งงาน	
18.	totalTime	Number	เวลาประเมินที่ใช้ในการดำเนินงาน (วินาที)	
19.	totalDistance	Number	ระยะทางรวมของคำสั่งงาน	
20.	totalPricePlatform	Number	ส่วนแบ่งรายได้ให้กับระบบ	
21.	totalPriceRider	Number	ส่วนแบ่งรายได้ให้กับพนักงาน	
22.	totalPrice	Number	ราคารวมที่ผู้ใช้ต้องจ่าย	
23.	discountPrice	Number	ส่วนลด	
24.	addedServicePrice	Number	ราคาค่าบริการเสริม	
25.	byDistancePrice	Number	ราคาตามระยะทาง	
26.	riderEndTime	Date	วันที่เวลาที่พนักงานปิดงาน	
27.	riderStartTime	Date	วันที่เวลาที่พนักงานเริ่มงาน	
28.	endTime	Date	วันที่เวลาที่ประเมินในการดำเนินการคำสั่งงานแล้วเสร็จ	
29.	startTime	Date	วันที่เวลาที่ระบุให้เริ่มทำคำสั่งงาน	

ตาราง ก.8 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
30.	isCloseJobPhotoRequired	Boolean	ระบุว่าต้องการให้ถ่ายรูปปิดงานหรือไม่	
31.	isRoundTrip	Boolean	ระบุให้คำสั่งงานเป็นแบบไปกลับ	
32.	refCode	String	หมายเลขอ้างอิงคำสั่งงาน	
33.	rider	ObjectId	รหัสพนักงานที่รับงาน	
34.	deductRiderCredit	Number	จำนวนเครดิตที่พนักงานโดนหักหลังจากรับงาน	
35.	rating	ObjectId	รหัสการให้คะแนนกับพนักงานในการทำงาน	
36.	promotion	ObjectId	รหัสโปรโมชั่น	
37.	payment	Object	ช่องทางการชำระเงิน	
38.	payment._id	ObjectId	รหัสช่องทางการชำระเงิน	
39.	payment.name.en	String	ชื่อช่องทางการชำระเงินภาษาอังกฤษ	
40.	payment.name.th	String	ชื่อช่องทางการชำระเงินภาษาไทย	
41.	payment.priority	Number	ลำดับในการแสดงผล	
42.	payment.imgUrl	String	รูปภาพช่องทางการชำระเงิน	
43.	taxInfo	Object	ข้อมูลการออกใบกำกับภาษี	
44.	taxInfo.address	String	ที่อยู่ในการออกใบกำกับภาษี	
45.	taxInfo.taxIDNo	String	หมายเลขผู้เสียภาษี	
46.	taxInfo._id	String	รหัสบริษัท	
47.	taxInfo.name	ObjectId	ชื่อบริษัทผู้เสียภาษี	

ตาราง ก.8 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
48.	taxInfo.email	String	อีเมล์บริษัทผู้เสียภาษี	
49.	taxInfo.tel	String	หมายเลขโทรศัพท์ผู้เสียภาษี	
50.	taxInfo.branch	String	สาขาบริษัทผู้เสียภาษี	
51.	services[]._id	ObjectId	รหัสจุดให้บริการแต่ละจุด	
52.	services[].contactNo	String	หมายเลขโทรศัพท์ผู้ติดต่อในแต่ละจุดให้บริการ	
53.	services[].contactName	String	ชื่อผู้ติดต่อในแต่ละจุดให้บริการ	
54.	services[].name	String	ชื่อจุดให้บริการ	
55.	services[].addedServices[]	ObjectId	บริการเสริมในแต่ละจุด	
56.	services[].status	Object	สถานะการทำงานในแต่ละจุด	
57.	services[].status.reason	String	เหตุผลในการปิดงานในแต่ละจุด	
58.	services[].status.status	String	ชื่อสถานะการทำงานในแต่ละจุด	
59.	services[].status.userId	ObjectId	รหัสผู้ใช้งานเจ้าของคำสั่งงาน	
60.	services[].status.createBy	String	ผู้ทำการเปลี่ยนแปลงสถานะคำสั่งงาน	
61.	services[].status.signature.timeStamp	Date	เวลาที่ทำรายการ	
62.	services[].status.signature.name	String	ชื่อผู้เซ็นปิดงาน	
63.	services[].status.signature.image	String	รูปถ่ายเซ็นปิดงาน	
64.	services[].status.timeStamp	Date	เวลาที่เซ็นปิดงาน	
65.	services[].status.image	String	รูปถ่ายปิดงาน	

ตาราง ก.8 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
66.	services[].status.note	String	หมายเหตุในการทำรายการ	
67.	services[].totalTime	Number	เวลาประเมินที่ใช้ในการดำเนินงาน (วินาที)	
68.	services[].totalDistance	Number	ระยะทางรวมของคำสั่งงาน	
69.	services[].totalPrice	Number	ราคารวมที่ผู้ใช้ต้องจ่ายแต่ละจุด	
70.	services[].endTime	Date	วันที่เวลาที่เริ่มคำสั่งงานแต่ละจุด	
71.	services[].startTime	Date	วันที่เวลาที่ปิดจุดคำสั่งงาน	
72.	services[].note	String	หมายเหตุ	
73.	services[].location	Object	พิกัดจุดบริการ	
74.	services[].location.longitude	Number	ลองจิจูด	
75.	services[].location.latitude	Number	ละติจูด	
76.	services[].location.zipcode	String	รหัสไปรษณีย์	
77.	services[].location.province	String	จังหวัด	
78.	services[].location.address	String	ที่อยู่	
79.	services[].location.name	String	ชื่อสถานที่	
80.	services[].location.coordinates[]	Number	โคออดิเนต ละติจูด ลองจิจูด	
81.	services[].location.type	String	ประเภทของข้อมูล	

ตาราง ก.9 ตาราง orders_cancel

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	orderId	ObjectId	รหัสคำสั่งงาน	FK
3.	status	String	สถานะคำสั่งงาน	
4.	createBy	String	ผู้ทำรายการ	
5.	userId	ObjectId	รหัสผู้ใช้เจ้าของคำสั่งงาน	FK
6.	riderId	ObjectId	รหัสพนักงานที่รับงาน	FK
7.	timeStamp	Date	เวลาที่ทำรายการ	
8.	isApproved	Boolean	รายการไดรับการอนุมัติหรือไม่	
9.	cancelDate	Date	วันที่ที่ทำการยกเลิกคำสั่งงาน	
10.	cancelPrice	Number	ค่าปรับจากการยกเลิกงาน	
11.	reason	String	เหตุผลในการยกเลิกงาน	

ตาราง ก.10 ตาราง orders_queue

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	orderId	ObjectId	รหัสคำสั่งงาน	FK
3.	status	String	สถานะคำสั่งงาน	
4.	createBy	String	ผู้ทำรายการ	
5.	userId	ObjectId	รหัสผู้ใช้เจ้าของคำสั่งงาน	FK
6.	isReserveInAdvance	Boolean	คำสั่งงานแบบจองล่วงหน้า	
7.	refCode	String	หมายเลขอ้างอิงคำสั่งงาน	
8.	timeStamp	Date	เวลาทำรายการ	
9.	riderEndTime	Date	เวลาพนักงานเริ่มงาน	
10.	riderStartTime	Date	เวลาพนักงานปิดงาน	
11.	endTime	Date	วันที่เวลาที่ประเมินในการดำเนินการคำสั่งงานแล้วเสร็จ	
12.	startTime	Date	วันที่เวลาที่ระบุให้เริ่มทำคำสั่งงาน	
13.	riderId	ObjectId	รหัสพนักงานที่รับงาน	FK

ตาราง ก.11 ตาราง otps

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	dateTime	Date	วันที่ที่ส่งข้อความ	
3.	mobileNo	String	หมายเลขโทรศัพท์	
4.	otp	String	รหัส OTP	
5.	transactionId	String	รหัสการทำรายการที่ใช้อ้างอิง	

ตาราง ก.12 ตาราง payment_options

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	name	Object	ช่องทางการชำระเงิน	
3.	name.en	String	ชื่อช่องทางการชำระเงินภาษาอังกฤษ	
4.	name.th	String	ชื่อช่องทางการชำระเงินภาษาไทย	
5.	imgUrl	String	รูปภาพช่องทางการชำระเงิน	
6.	priority	Number	ลำดับในการแสดงผล	
7.	type	String	ประเภทการชำระเงินสำหรับกลุ่มผู้ใช้	
8.	active	Boolean	เปิดใช้งาน	
9.	deductRiderCredit	Boolean	หักเครดิตพนักงานหากมีการชำระเงินผ่านช่องทางนี้	

ตาราง ก.13 ตาราง promotion_histories

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	promotion	ObjectId	รหัสโปรโมชั่น	FK
3.	user	ObjectId	รหัสผู้ใช้	FK
4.	rider	ObjectId	รหัสพนักงาน	FK
5.	dateTime	Date	วันที่รำรายการ	
6.	order	ObjectId	รหัสคำสั่งงาน	FK

ตาราง ก.14 ตาราง promotions

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	createBy	ObjectId	รหัสผู้เพิ่มรายการ	FK
3.	updateBy	ObjectId	รหัสผู้แก้ไขรายการ	FK
4.	name	Object	ชื่อโปรโมชั่น	
5.	name.en	String	ชื่อโปรโมชั่นภาษาอังกฤษ	
6.	name.th	String	ชื่อโปรโมชั่นภาษาไทย	
7.	desc	Object	คำบรรยายโปรโมชั่น	
8.	desc.en	String	คำบรรยายโปรโมชั่นภาษาอังกฤษ	
9.	desc.th	String	คำบรรยายโปรโมชั่นภาษาไทย	
10.	value	Number	มูลค่า	

ตาราง ก.14 ตาราง promotions

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
11.	rule	String	กติกาการใช้สิทธิ์	
12.	deleteFlag	Boolean	ระบุการลบรายการ	
13.	updateDate	Date	วันที่ที่ทำการแก้ไขข้อมูล	
14.	createDate	Date	วันที่ที่ทำการเพิ่มข้อมูล	
15.	quota	Object	โควตาการใช้สิทธิ์	
16.	quota.total	Number	จำนวนโควตาการใช้สิทธิ์ทั้งหมด	
17.	quota.quotaPerUser	Number	จำนวนครั้งที่ใช้สิทธิ์ได้ต่อหนึ่งท่าน	
18.	quota.redeemed	Number	จำนวนครั้งที่มีการใช้สิทธิ์	
19.	state	String	ขั้นตอนการใช้งานโปรโมชั่น	
20.	active	Boolean	สถานะการใช้งาน	
21.	promotionFor	String	จำแนกโปรโมชั่นสำหรับผู้ใช้แต่ละประเภท	
22.	endDate	Date	วันที่สิ้นสุดโปรโมชั่น	
23.	startDate	Date	วันที่เริ่มต้นโปรโมชั่น	
24.	coverImgUrl	String	รูปที่ใช้แสดง	
25.	imgUrl	String	รูปที่ใช้แสดง	
26.	type	String	ประเภทมูลค่าของโปรโมโมชั่น	
27.	code	String	รหัสอ้างอิง	

ตาราง ก.15 ตาราง purchase_histories

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	user	ObjectId	รหัสผู้ใช้	FK
3.	order	ObjectId	รหัสคำสั่งงาน	FK
4.	payRef	String	หมายเลขอ้างอิงการทำรายการ	
5.	refCode	String	หมายเลขอ้างอิงคำสั่งงาน	
6.	status	String	สถานะการทำรายการ	
7.	channel	String	ช่องทางการชำระเงิน	
8.	updateDate	Date	วันที่ที่มีการแก้ไขข้อมูล	
9.	createDate	Date	วันที่ที่ทำรายการ	

ตาราง ก.16 ตาราง ratings

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	riderId	ObjectId	รหัสพนักงาน	FK
3.	userId	ObjectId	รหัสผู้ใช้	FK
4.	orderId	ObjectId	รหัสคำสั่งงาน	FK
5.	timeStamp	Date	วันที่เวลาที่ทำรายการ	
6.	reason	String	เหตุผลการให้คะแนน	
7.	rate	Number	คะแนน	

ตาราง ก.17 ตาราง reasons

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	name	Object	เหตุผล	
3.	name.en	String	เหตุผลภาษาอังกฤษ	
4.	name.th	String	เหตุผลภาษาไทย	
5.	userType	String	เหตุผลของประเภทผู้ใช้งาน	
6.	type	String	ประเภทของเหตุผล	
7.	priority	Number	ลำดับการแสดงผล	
8.	state	String	ขั้นตอนการนำไปใช้งาน	

ตาราง ก.18 ตาราง rider_topups

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	rider	ObjectId	รหัสพนักงาน	FK
3.	credit	Number	จำนวนเครดิต	
4.	dateTime	Date	วันที่ทำรายการ	
5.	imgUrl	String	รูปถ่ายสลิปการโอนเงิน	
6.	status	String	สถานะการอนุมัติ	
7.	approveBy	ObjectId	ผู้ทำการอนุมัติ	FK

ตาราง ก.19 ตาราง riders

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	firstName	String	ชื่อ	
3.	lastName	String	นามสกุล	
4.	mobileNo	String	หมายเลขโทรศัพท์	
5.	riderId	String	รหัสอ้างอิง	
6.	driverLicenseNo	String	หมายเลขใบอนุญาติขับขี่	
7.	licensePlate	String	หมายเลขทะเบียนรถ	
8.	motorcycleBand	String	รุ่นรถ	
9.	idCard	String	หมายเลขประจำตัวประชาชน	
10.	approvedStatus	Boolean	สถานะการอนุมัติการสมัครสมาชิก	
11.	activeStatus	Object	สถานะของพนักงาน	
12.	activeStatus.status	String	ชื่อสถานะของพนักงาน	
13.	activeStatus.activeDate	Date	วันที่เริ่มทำงานได้	
14.	readyForWorkStatus	Boolean	เปิดปิดการรับงาน	
15.	needNotification	Boolean	เปิดปิดการแจ้งเตือน	
16.	totalJob	Number	จำนวนงานทั้งหมด	
17.	unAcceptJob	Number	จำนวนการปฏิเสธงาน	
18.	acceptJob	Number	จำนวนการรับงาน	
19.	credit	Number	ยอดเครดิตคงเหลือ	
20.	bankAccount	Object	บัญชีธนาคาร	

ตาราง ก.19 (ต่อ 1)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
21.	bankAccount.bankAccountNo	String	หมายเลขบัญชี	
22.	bankAccount.bankBranch	String	สาขาธนาคาร	
23.	bankAccount.bankName	String	ชื่อธนาคาร	
24.	bankAccount.bankAccountName	String	ชื่อบัญชี	
25.	bankAccountImg	String	รูปถ่ายเล่มธนาคาร	
26.	licensePlateImg	String	รูปถ่ายทะเบียนรถ	
27.	socialId	String	รหัสโซเชียล	
28.	deviceId	String	รหัสอุปกรณ์มือถือ	
29.	registerDate	Date	วันที่สมัครสมาชิก	
30.	lastLogin	Date	เวลาเข้าสู่ระบบล่าสุด	
31.	loginState	String	สถานะการเข้าสู่ระบบ	
32.	tokenExpired	Number	เวลาที่สามารถใช้ Token ได้	
33.	accessToken	String	Token การเข้าใช้งานได้จากการลงชื่อเข้าสู่ระบบ	
34.	platform	String	ประเภทอุปกรณ์มือถือที่ใช้งาน	
35.	imgUrl	String	รูปโปรไฟล์	
36.	password	String	รหัสผ่าน	
37.	idCardImgUrl	String	รูปถ่ายบัตรประจำตัวประชาชน	
38.	driverLicenseImgUrl	String	รูปถ่ายใบอนุญาติขับขี่	
39.	email	String	ที่อยู่อีเมล์	

ตาราง ก.19 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
40.	rating	Number	คะแนนการทำงาน	
41.	serviceArea	ObjectId	พื้นที่ที่รับงาน	
42.	driverLicenseExpireDate	Date	วันที่ใบอนุญาติขับขี่หมดอายุ	
43.	licensePlateExpireDate	Date	วันที่ทะเบียนรถหมดอายุ	
44.	idCardExpireDate	Date	วันที่บัตรประจำตัวประชาชนหมดอายุ	
45.	updateBy	ObjectId	รหัสผู้ทำการแก้ไขข้อมูล	
46.	updateDate	Date	วันที่ทำการแก้ไขข้อมูล	
47.	lastLocation	Object	ตำแหน่งล่าสุดที่เปิดรับงาน	
48.	lastLocation.longitude	Number	ลองจิจูด	
49.	lastLocation.latitude	Number	ละติจูด	
50.	lastLocation.coordinates[]	Number	โคออดิเนต ลองจิจูด ละติจูด	
51.	lastLocation.type	String	ประเภทข้อมูลพิกัด	

ตาราง ก.20 ตาราง riders_location

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	riderId	ObjectId	รหัสพนักงาน	FK
3.	mobileNo	String	หมายเลขโทรศัพท์	
4.	socketId	String	หมายเลข socket	
5.	lastOnline	Date	วันที่ออนไลน์ล่าสุด	
6.	workingStatus	Object	สถานะงานที่ถืออยู่	
7.	workingStatus.orderId	ObjectId	รหัสงานที่ถืออยู่	
8.	workingStatus.status	String	ชื่อสถานะงาน	
9.	workingStatus.isReserveInAdvance	Boolean	ประเภทงานจองล่วงหน้า	
10.	workingStatus.endTime	Date	วันที่เริ่มงาน	
11.	workingStatus.startTime	Date	วันที่สิ้นสุดงาน	
12.	isOnline	Boolean	สถานะออนไลน์ในระบบ	
13.	lastUpdate	Date	การเปลี่ยนแปลงสถานะครั้งล่าสุด	
14.	location	Object	ตำแหน่งล่าสุดที่เปิดรับงาน	
15.	location.longitude	Number	ลองจิจูด	
16.	location.latitude	Number	ละติจูด	
17.	location.coordinates[]	Number	โคออดิเนต ลองจิจูด ละติจูด	
18.	location.type	String	ประเภทข้อมูลพิกัด	

ตาราง ก.21 ตาราง service_areas

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	groupName	Object	กลุ่มพื้นที่ให้บริการ	
3.	groupName.en	String	ชื่อกลุ่มพื้นที่ให้บริการภาษาอังกฤษ	
4.	groupName.th	String	ชื่อกลุ่มพื้นที่ให้บริการภาษาไทย	
5.	provinces[]	Object	รายการจังหวัด	
6.	provinces[].name	String	ชื่อจังหวัด	
7.	provinces[].name.th	String	ชื่อจังหวัดภาษาอังกฤษ	
8.	provinces[].name.en	String	ชื่อจังหวัดภาษไทย	
9.	provinces[].areas[]	Object	รายการเขตพื้นที่บริการ	
10.	provinces[].areas[].name	String	ชื่อเขตพื้นที่บริการ	
11.	provinces[].areas[].zipcode	String	รหัสไปรษณีย์พื้นที่บริการ	
12.	provinces[].areas[]._id	ObjectId	รหัสพื้นที่บริการ	

ตาราง ก.22 ตาราง service_items

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	name	Object	ชื่อบริการ	
3.	name.en	String	ชื่อบริการภาษาอังกฤษ	
4.	name.th	String	ชื่อบริการภาษาไทย	
5.	price	Number	ค่าบริการ	
6.	priceType	String	ประเภทค่าบริการ	
7.	vat	Number	ภาษีหัก ณ ที่จ่าย	
8.	includeVat	Boolean	รวมภาษีภาษีหัก ณ ที่จ่าย	
9.	priority	Number	ลำดับการแสดงผล	
10.	isShow	Boolean	แสดงรายการนี้	
11.	serviceType	String	ประเภทบริการ	
12.	createBy	ObjectId	รหัสผู้เพิ่มรายการ	FK
13.	createDate	Date	วันที่เพิ่มรายการ	
14.	updateBy	ObjectId	รหัสผู้แก้ไขข้อมูล	FK
15.	updateDate	Date	วันที่ทำการแก้ไขข้อมูล	
16.	deleteFlag	Boolean	ลำบรายการ	
17.	active	Boolean	สถานะการใช้งาน	

ตาราง ก.23 ตาราง staffs

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	firstName	String	ชื่อ	
3.	lastName	String	นามสกุล	
4.	mobileNo	String	หมายเลขโทรศัพท์	
5.	email	String	ที่อยู่อีเมล์	
6.	password	String	รหัสผ่าน	
7.	userRole	Number	รหัสหน้าที่การทำงาน	
8.	tokenExpired	Number	เวลาที่สามารถใช้ Token ได้	
9.	accessToken	String	Token การเข้าใช้งานได้จากการลงชื่อเข้าสู่ระบบ	
10.	createBy	ObjectId	รหัสผู้เพิ่มรายการ	FK
11.	createDate	Date	วันที่เพิ่มรายการ	
12.	updateBy	ObjectId	รหัสผู้แก้ไขข้อมูล	FK
13.	updateDate	Date	วันที่ทำการแก้ไขข้อมูล	
14.	lastLogin	Date	เวลาเข้าสู่ระบบล่าสุด	
15.	passCode	String	รหัสผ่าน	
16.	deviceId	String	รหัสอุปกรณ์มือถือ	

ตาราง ก.24 ตาราง tax_invoices

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	id	ObjectId	รหัสข้อมูล	PK
2.	orderId	ObjectId	รหัสคำสั่งงาน	FK
3.	requestDate	Date	วันที่ทำรายการ	
4.	userId	ObjectId	รหัสผู้ใช้	FK
5.	status	String	สถานะการอนุมัติ	

ตาราง ก.25 ตาราง users

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	firstName	String	ชื่อ	
3.	lastName	String	นามสกุล	
4.	mobileNo	String	หมายเลขโทรศัพท์	
5.	email	String	ที่อยู่อีเมล์	
6.	riderFav[]	String	รายการพนักงานที่ชื่นชอบ	
7.	company[]	Object	ข้อมูลการออกใบกำกับภาษี	
8.	company[].email	String	อีเมล์บริษัทผู้เสียภาษี	
9.	company[].tel	String	หมายเลขโทรศัพท์	
10.	company[].address	String	ที่อยู่ในการออกใบกำกับภาษี	
11.	company[].name	ObjectId	ชื่อบริษัทผู้เสียภาษี	FK

ตาราง ก.25 (ต่อ)

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
12.	company[]._id	String	รหัสบริษัทผู้เสียภาษี	
13.	company[].taxIDNo	String	หมายเลขผู้เสียภาษี	
14.	company[].branch	String	สาขาบริษัทผู้เสียภาษี	
15.	socialId	String	รหัสโซเชียล	
16.	deviceId	String	รหัสอุปกรณ์มือถือ	
17.	lastLogin	Date	เวลาเข้าสู่ระบบล่าสุด	
18.	registerDate	Date	วันที่สมัครสมาชิก	
19.	loginState	String	สถานะการเข้าสู่ระบบ	
20.	tokenExpired	Number	เวลาที่สามารถใช้ Token ได้	
21.	accessToken	String	Token การเข้าใช้งานได้จากการลงชื่อเข้าสู่ระบบ	
22.	platform	String	ประเภทอุปกรณ์มือถือที่ใช้งาน	
23.	imgUrl	String	รูปโปรไฟล์	
24.	password	String	รหัสผ่าน	
25.	needNotification	Boolean	เปิดปิดการแจ้งเตือน	

ตาราง ก.26 ตาราง versions

ลำดับ	แอทริบิวต์	ชนิด	ความหมาย	คีย์
1.	_id	ObjectId	รหัสข้อมูล	PK
2.	message	String	ข้อความแจ้งเตือน	
3.	appId	String	รหัสอ้างอิง	
4.	version	Number	เลขเวอร์ชัน	
5.	platform	String	สำหรับประเภทอุปกรณ์	
6.	redirectUrl	String	URL สำหรับดาวน์โหลดแอพ	
7.	userType	String	ประเภทของแอพ	
8.	force	Boolean	บังคับการอัพเดท	

ภาคผนวก ข

Application Programming Interface (API)

1.1 POST /order/user/taxInvoice/{orderId}

Description

request taxInvoice

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string
body	body		Yes	{ "_id": "string", "name": "string", "address": "string", "email": "string", "tel": "string", "taxIDNo": "string", "branch": "string", "action": "string" }

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

2.1 POST /promotion/{userType}/code

Description

get and check promotion by code

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path		Yes	string
redeemCode	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "name": { "en": "string", "th": "string", "localized": "string" },

Code	Description	Schema

		"code": "string",
		"type": "string",
		"value": "number",
		"desc": {
		"en": "string",
		"th": "string",
		"localized": "string"
		},
		"imgUrl": "string",
		"coverImgUrl": "string",
		"startDate": "string",
		"endDate": "string"
		}
		]
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

2.2 POST /promotion/{userType}/code

Description

get and check promotion by code

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path		Yes	string
redeemCode	formData		Yes	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": [
		{
		"_id": "string",
		"name": {
		"en": "string",
		"th": "string",
		"localized": "string"
		},

Code	Description	Schema


```
    "code": "string",
    "type": "string",
    "value": "number",
    "desc": {
      "en": "string",
      "th": "string",
      "localized": "string"
    },
    "imgUrl": "string",
    "coverImgUrl": "string",
    "startDate": "string",
    "endDate": "string"
  }
]
}
```

3.1 GET /reasons/{userType}/{type}

Description

get reasons rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path	userType is user/rider	Yes	string
type	path	type is cancel/rating	Yes	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": [
		{
		"_id": "string",
		"name": {
		"en": "string",
		"th": "string",
		"localized": "string"
		}
		}

Code	Description	Schema
		] }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

3.2 GET /reasons/{userType}/{type}/{state}

Description

get reasons by state is confirm / doing

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path	userType is user/rider	Yes	string
type	path	type is cancel/rating	Yes	string
state	path	type is confirm / doing	Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "name": { "en": "string", "th": "string", "localized": "string" }

Code	Description	Schema
		}] }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

4.1 GET /paymentOptions/user

Description

get payment option list

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": [
		{
		"_id": "string",
		"name": {
		"en": "string",
		"th": "string",
		"localized": "string"
		},
		"imgUrl": "string",
		"paymentId": "string",
		"isCashOnDelivery": "boolean"
		}
		]
		}

Code	Description	Schema

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

4.2 GET /paymentOptions/rider

Description

get payment option list rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": [
		{
		"_id": "string",
		"name": {
		"en": "string",
		"th": "string",
		"localized": "string"
		},
		"imgUrl": "string",
		"paymentId": "string",
		"isCashOnDelivery": "boolean"
		}
		]
		}

Code	Description	Schema
default	Unexpected error	{ "code": "number", "description": "string" }

5.1 GET /terms

Description

get terms

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": {
		"url": "string",
		"checkUrl": "string"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

6.1 GET /serviceItems

Description

get Service Items

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "name": { "en": "string", "th": "string", "localized": "string" }, "priceType": "string", "price": "number", "note": "string", "serviceType": "string", "isShow": "boolean", "totalPrice": "number"

Code	Description	Schema
		}] }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

7.1 GET /notifications/{userType}

Description

get user / rider notification type is user or rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path	user or rider	Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "dateTime": "string", "message": "string", "type": "string", "content": { "refCode": "string", "startTime": "string", "startLocation": "string",

Code	Description	Schema
		"endLocation": "string", "totalPrice": "number", "remark": "string", "status": "string", "imgUrl": "string", "bannerImgUrl": "string", "content": "string", "orderId": "string" }, "isRead": "boolean" }] }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

7.2 POST /notifications/{userType}

Description

read user / rider notification type is user or rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path	user or rider	Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

7.3 POST /notifications/{userType}/{notificationId}

Description

read user / rider notification type is user or rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path	user or rider	Yes	string
notificationId	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

7.4 DELETE /notifications/{userType}/{notificationId}

Description

delete user / rider notification type is user or rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
userType	path	user or rider	Yes	string
notificationId	path		Yes	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string"
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

8.1 POST /otp/send

Description

send OTP

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header	en,th default th	No	string
mobileNo	formData		Yes	string
email	formData		No	string

Responses

Code	Description	Schema
200	transactinId will be return in data object	{ "code": "number", "description": "string", "data": { "transactionId": "string" } }
default	Unexpected error	{ "code": "number", "description": "string" }

8.2 POST /otp/confirm

Description

confirm OTP

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
mobileNo	formData		Yes	string
transactionId	formData		Yes	string
otp	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "transactionId": "string" } }
default	Unexpected error	{ "code": "number", "description": "string" }

9.1 POST /users/register

Description

register user

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
firstName	formData		Yes	string
lastName	formData		Yes	string
mobileNo	formData		Yes	string
email	formData		Yes	string
transactionId	formData		Yes	string
otp	formData		Yes	string
type	formData		Yes	string
socialId	formData	require for type is social	No	string
socialToken	formData	require for type is social	No	string
password	formData	require for type is basic	No	string
imgUrl	formData		No	string
pic	formData	size 200 x 200px	No	file

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "email": "string", "accessToken": "string", "imgUrl": "string", "notificationUnRead": "number", "company": [{ "_id": "string", "name": "string", "address": "string", "email": "string", "tel": "string", "taxIDNo": "string", "branch": "string", "action": "string" }], "needNotification": "boolean", "orderDoingAmount": "number" } }

Code	Description	Schema
default	Unexpected error	{ "code": "number", "description": "string"}

9.2 POST /users/forgotPass

Description

forgotPass

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
mobileNo	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "transactionId": "string" } }
default	Unexpected error	{ "code": "number", "description": "string" }

9.3 POST /users/resetPass

Description

Reset Password

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
mobileNo	formData		Yes	string
password	formData		Yes	string
otp	formData		Yes	string
transactionId	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

9.4 POST /users/updateProfile

Description

Update user profile

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
firstName	formData		No	string
lastName	formData		No	string
deviceId	formData		No	string
company	formData		No	string
pic	formData		No	file
email	formData		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "firstName": "string",

Code	Description	Schema

		"lastName": "string",
		"mobileNo": "string",
		"email": "string",
		"accessToken": "string",
		"imgUrl": "string",
		"notificationUnRead": "number",
		"company": [
		{
		"_id": "string",
		"name": "string",
		"address": "string",
		"email": "string",
		"tel": "string",
		"taxIDNo": "string",
		"branch": "string",
		"action": "string"
		}
		],
		"needNotification": "boolean",
		"orderDoingAmount": "number"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

9.5 POST /users/changeMobileNo

Description

When user change mobile

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header		Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
mobileNo	formData	New mobileNo for change	No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "transactionId": "string" } }
default	Unexpected error	{ "code": "number", "description": "string" }

9.6 POST /users/updateMobileNo

Description

Call after changeMobileNo for change mobileNo user

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header		Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
mobileNo	formData	New mobileNo for change	No	string
otp	formData		No	string
transactionId	formData		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "email": "string",

Code	Description	Schema

		"accessToken": "string",
		"imgUrl": "string",
		"notificationUnRead": "number",
		"company": [
		{
		"_id": "string",
		"name": "string",
		"address": "string",
		"email": "string",
		"tel": "string",
		"taxIDNo": "string",
		"branch": "string",
		"action": "string"
		}
		],
		"needNotification": "boolean",
		"orderDoingAmount": "number"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

9.7 POST /users/changePass

Description

Change Password

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
oldPassword	formData		Yes	string
newPassword	formData		Yes	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string"
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

9.8 POST /users/setNewPass

Description

Set Password

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
password	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

9.9 POST /users/logout

Description

Logout

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string"
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

9.10 GET /users/{_id}

Description

get user profile

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
_id	path	user id	Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "email": "string", "accessToken": "string", "imgUrl": "string", "notificationUnRead": "number", "company": [

Code	Description	Schema

		{
		"_id": "string",
		"name": "string",
		"address": "string",
		"email": "string",
		"tel": "string",
		"taxIDNo": "string",
		"branch": "string",
		"action": "string"
		}
		],
		"needNotification": "boolean",
		"orderDoingAmount": "number"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

9.11 GET /users/rider

Description

get favorite riders

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "riderId": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "licensePlate": "string", "bankAccountImg": "string", "licensePlateImg": "string", "imgUrl": "string",

Code	Description	Schema

		"idCardImgUrl": "string",
		"driverLicenseImgUrl": "string",
		"email": "string",
		"accessToken": "string",
		"bankAccount": {
		"bankBranch": "string",
		"bankAccountName": "string",
		"bankAccountNo": "string",
		"bankName": "string"
		},
		"serviceArea": "string",
		"needNotification": "boolean",
		"credit": "number",
		"notificationUnRead": "number",
		"totalJobPercent": "number",
		"readyForWorkStatus": "boolean",
		"rating": "number",
		"driverLicenseExpireDate": "string",
		"licensePlateExpireDate": "string",
		"idCardExpireDate": "string"
		}
		]
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

9.12 POST /users/rider

Description

add favorite rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
riderId	formData	riderId is _id from response	Yes	string
action	formData	action is $push/$pull	Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "riderId": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "licensePlate": "string",

Code	Description	Schema

		"bankAccountImg": "string",
		"licensePlateImg": "string",
		"imgUrl": "string",
		"idCardImgUrl": "string",
		"driverLicenseImgUrl": "string",
		"email": "string",
		"accessToken": "string",
		"bankAccount": {
		"bankBranch": "string",
		"bankAccountName": "string",
		"bankAccountNo": "string",
		"bankName": "string"
		},
		"serviceArea": "string",
		"needNotification": "boolean",
		"credit": "number",
		"notificationUnRead": "number",
		"totalJobPercent": "number",
		"readyForWorkStatus": "boolean",
		"rating": "number",
		"driverLicenseExpireDate": "string",
		"licensePlateExpireDate": "string",
		"idCardExpireDate": "string"
		}
		]
		}

default	Unexpected error	{ "code": "number", "description": "string"}

10.1 GET /banners

Description

get banners wait rider accept order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "imgUrl": "string", "redirectUrl": "string" }] }
default	Unexpected error	{ "code": "number", "description": "string", "data": "string" }

11.1 POST /order/user

Description

create order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
body	body		Yes	{ "type": "string", "startTime": "string", "services": [{ "_id": "string", "name": "string", "location": { "latitude": "number", "longitude": "number", "name": "string", "address": "string", "zipcode": "string", "province": "string" }, "contactName": "string", "contactNo": "string",

Name	Located In	Description	Required	Schema


```
    "note": "string",
    "addedServices": [
     {
      "_id": "string"
     }
    ]
   }
  ],
  "isRoundTrip": "boolean",
  "isCloseJobPhotoRequired":
"boolean",
  "promotion": {
   "_id": "string"
  },
"taxInfo": {
   "_id": "string",
   "name": "string",
   "address": "string",
   "email": "string",
   "tel": "string",
   "taxIDNo": "string",
   "branch": "string",
   "action": "string"
  }
}
```

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "refCode": "string", "type": "string", "isReserveInAdvance": "boolean", "startTime": "string", "endTime": "string", "services": [{ "_id": "string", "name": "string", "location": { "latitude": "number", "longitude": "number", "type": "string", "coordinates": ["number"], "name": "string", "address": "string", "zipcode": "string", "province": "string" }, "contactName": "string", "contactNo": "string", "note": "string",

Code	Description	Schema


```
"addedServices": [
 {
  "_id": "string",
  "name": {
   "en": "string",
   "th": "string",
   "localized": "string"
  },
  "priceType": "string",
  "price": "number",
  "note": "string",
  "serviceType": "string",
  "isShow": "boolean",
  "totalPrice": "number"
 }
]
}
],
  "name": {
   "en": "string",
   "th": "string",
   "localized": "string"
  },
  "priceType": "string",
  "price": "number",
  "note": "string",
  "serviceType": "string",
  "isShow": "boolean",
  "totalPrice": "number"
 }
```

Code	Description	Schema


```
    ]
  }
],
"isRoundTrip": "boolean",
"isCloseJobPhotoRequired": "boolean",
"promotion": {
  "_id": "string"
},
"note": "string",
"taxInfo": {
  "_id": "string",
  "name": "string",
  "address": "string",
  "email": "string",
  "tel": "string",
  "taxIDNo": "string",
  "branch": "string",
  "action": "string"
},
"byDistancePrice": "number",
"discountPrice": "number",
"addedServicePrice": "number",
"roundTripPrice": "number",
"totalPrice": "number",
"totalPriceRider": "number",
"totalPricePlatform": "number",
"deductRiderCredit": "number",
"totalDistance": "number",
"totalTime": "number",
"user": {
```

Code	Description	Schema

		"_id": "string"
		},
		"status": {
		"_id": "string",
		"createBy": "string",
		"userId": "string",
		"status": "string",
		"reason": "string",
		"note": "string",
		"image": "string",
		"timeStamp": "string",
		"signature": {
		"imageSignature": "string",
		"name": "string",
		"timeStamp": "string"
		}
		},
		"createDate": "string"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.2 GET /order/user

Description

get user order list

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
status	query	created|confirm|accepted|doing|done|cancel default is all	No	string
period	query	daily,weekly,monthly default is all	No	string
date	query	date format 2017-02-28T13:00:00Z	No	string
detail	query	default false	No	boolean

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "refCode": "string",

Code	Description	Schema


```
"type": "string",
"isReserveInAdvance": "boolean",
"startTime": "string",
"endTime": "string",
"riderStartTime": "string",
"riderEndTime": "string",
"services": [
 {
  "_id": "string",
  "name": "string",
  "location": {
   "latitude": "number",
   "longitude": "number",
   "type": "string",
   "coordinates": [
    "number"
   ],
   "name": "string",
   "address": "string",
   "zipcode": "string",
   "province": "string"
  },
  "contactName": "string",
  "contactNo": "string",
  "note": "string",
  "addedServices": [
   {
    "_id": "string",
    "name": {
     "en": "string",
```

Code	Description	Schema


```
      "th": "string",
      "localized": "string"
    },
    "priceType": "string",
    "price": "number",
    "note": "string",
    "serviceType": "string",
    "isShow": "boolean",
    "totalPrice": "number"
  }
],
"startTime": "string",
"endTime": "string",
"totalPrice": "number",
"totalDistance": "number",
"totalTime": "number",
"status": {
  "_id": "string",
  "createBy": "string",
  "userId": "string",
  "status": "string",
  "reason": "string",
  "note": "string",
  "image": "string",
  "timeStamp": "string",
  "signature": {
    "imageSignature": "string",
    "name": "string",
    "timeStamp": "string"
  }
```

Code	Description	Schema


```
    }
   }
  ],
  "isRoundTrip": "boolean",
  "isCloseJobPhotoRequired": "boolean",
  "promotion": {
   "_id": "string",
   "name": {
    "en": "string",
    "th": "string",
    "localized": "string"
   },
   "code": "string",
   "type": "string",
   "value": "number",
   "desc": {
    "en": "string",
    "th": "string",
    "localized": "string"
   },
   "imgUrl": "string",
   "coverImgUrl": "string",
   "startDate": "string",
   "endDate": "string"
  },
  "note": "string",
  "taxInfo": {
   "_id": "string",
   "name": "string",
   "address": "string",
```

Code	Description	Schema


```
  "email": "string",
  "tel": "string",
  "taxIDNo": "string",
  "branch": "string",
  "action": "string"
},
"byDistancePrice": "number",
"discountPrice": "number",
"addedServicePrice": "number",
"roundTripPrice": "number",
"totalPrice": "number",
"totalPriceRider": "number",
"totalPricePlatform": "number",
"deductRiderCredit": "number",
"totalDistance": "number",
"totalTime": "number",
"payment": {
  "_id": "string",
  "name": {
    "en": "string",
    "th": "string",
    "localized": "string"
  },
  "imgUrl": "string",
  "paymentId": "string",
  "isCashOnDelivery": "boolean"
},
"user": {
  "_id": "string",
  "firstName": "string",
```

Code	Description	Schema


```
  "lastName": "string",
  "mobileNo": "string",
  "email": "string"
 },
 "rider": {
  "_id": "string",
  "riderId": "string",
  "firstName": "string",
  "lastName": "string",
  "mobileNo": "string",
  "imgUrl": "string",
  "email": "string"
 },
 "status": {
  "_id": "string",
  "createBy": "string",
  "userId": "string",
  "status": "string",
  "reason": "string",
  "note": "string",
  "image": "string",
  "timeStamp": "string",
  "signature": {
   "imageSignature": "string",
   "name": "string",
   "timeStamp": "string"
  }
 },
 "rating": {
  "_id": "string",
```

Code	Description	Schema
		"userId": "string", "orderId": "string", "reason": "string", "rate": "number", "riderId": "string", "note": "string", "timeStamp": "string" }, "createDate": "string" }] }

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.3 GET /order/user/{orderId}

Description

get order by id

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "refCode": "string", "type": "string", "isReserveInAdvance": "boolean", "startTime": "string", "endTime": "string", "riderStartTime": "string", "riderEndTime": "string",

Code	Description	Schema

		"services": [
		{
		"_id": "string",
		"name": "string",
		"location": {
		"latitude": "number",
		"longitude": "number",
		"type": "string",
		"coordinates": [
		"number"
		],
		"name": "string",
		"address": "string",
		"zipcode": "string",
		"province": "string"
		},
		"contactName": "string",
		"contactNo": "string",
		"note": "string",
		"addedServices": [
		{
		"_id": "string",
		"name": {
		"en": "string",

default	Unexpected error	{ "code": "number", "description": "string", "data": "string" }

11.4 DELETE /order/user/{orderId}

Description

delete order by id

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": "string" }
default	Unexpected error	{ "code": "number", "description": "string", "data": "string" }

11.5 GET /order/rider

Description

get driver order list

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
status	query	created|confirm|accepted|doing|done|cancel default is all	No	string
period	query	daily,weekly,monthly default is all	No	string
date	query	date format 2017-02-28T13:00:00Z	No	string
detail	query	default false	No	boolean

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "refCode": "string", "type": "string",

Code	Description	Schema


```
"isReserveInAdvance": "boolean",
"startTime": "string",
"endTime": "string",
"riderStartTime": "string",
"riderEndTime": "string",
"services": [
 {
  "_id": "string",
  "name": "string",
  "location": {
   "latitude": "number",
   "longitude": "number",
   "type": "string",
   "coordinates": [
    "number"
   ],
   "name": "string",
   "address": "string",
   "zipcode": "string",
   "province": "string"
  },
  "contactName": "string",
  "contactNo": "string",
  "note": "string",
  "addedServices": [
   {
    "_id": "string",
    "name": {
     "en": "string",
     "th": "string",
```

Code	Description	Schema


```
      "localized": "string"
    },
    "priceType": "string",
    "price": "number",
    "note": "string",
    "serviceType": "string",
    "isShow": "boolean",
    "totalPrice": "number"
  }
],
"startTime": "string",
"endTime": "string",
"totalPrice": "number",
"totalDistance": "number",
"totalTime": "number",
"status": {
  "_id": "string",
  "createBy": "string",
  "userId": "string",
  "status": "string",
  "reason": "string",
  "note": "string",
  "image": "string",
  "timeStamp": "string",
  "signature": {
    "imageSignature": "string",
    "name": "string",
    "timeStamp": "string"
  }
}
```

Code	Description	Schema


```
  }
],
"isRoundTrip": "boolean",
"isCloseJobPhotoRequired": "boolean",
"promotion": {
 "_id": "string",
 "name": {
  "en": "string",
  "th": "string",
  "localized": "string"
 },
 "code": "string",
 "type": "string",
 "value": "number",
 "desc": {
  "en": "string",
  "th": "string",
  "localized": "string"
 },
 "imgUrl": "string",
 "coverImgUrl": "string",
 "startDate": "string",
 "endDate": "string"
},
"note": "string",
"taxInfo": {
 "_id": "string",
 "name": "string",
 "address": "string",
 "email": "string",
```

Code	Description	Schema


```
    "tel": "string",
    "taxIDNo": "string",
    "branch": "string",
    "action": "string"
  },
  "byDistancePrice": "number",
  "discountPrice": "number",
  "addedServicePrice": "number",
  "roundTripPrice": "number",
  "totalPrice": "number",
  "totalPriceRider": "number",
  "totalPricePlatform": "number",
  "deductRiderCredit": "number",
  "totalDistance": "number",
  "totalTime": "number",
  "payment": {
    "_id": "string",
    "name": {
      "en": "string",
      "th": "string",
      "localized": "string"
    },
    "imgUrl": "string",
    "paymentId": "string",
    "isCashOnDelivery": "boolean"
  },
  "user": {
    "_id": "string",
    "firstName": "string",
    "lastName": "string",
```

Code	Description	Schema


```
  "mobileNo": "string",
  "email": "string"
 },
 "rider": {
  "_id": "string",
  "riderId": "string",
  "firstName": "string",
  "lastName": "string",
  "mobileNo": "string",
  "imgUrl": "string",
  "email": "string"
 },
 "status": {
  "_id": "string",
  "createBy": "string",
  "userId": "string",
  "status": "string",
  "reason": "string",
  "note": "string",
  "image": "string",
  "timeStamp": "string",
  "signature": {
   "imageSignature": "string",
   "name": "string",
   "timeStamp": "string"
  }
 },
 "rating": {
  "_id": "string",
  "userId": "string",
```

Code	Description	Schema


```
        "orderId": "string",
        "reason": "string",
        "rate": "number",
        "riderId": "string",
        "note": "string",
        "timeStamp": "string"
      },
      "createDate": "string"
    }
  ]
}
    },
        "priceType": "string",
        "price": "number",
        "note": "string",
        "serviceType": "string",
        "isShow": "boolean",
        "totalPrice": "number"
      }
    ],
    "startTime": "string",
    "endTime": "string",
    "totalPrice": "number",
    "totalDistance": "number",
    "totalTime": "number",
    "status": {
      "_id": "string",
      "createBy": "string",
      "userId": "string",
      "status": "string",
```

Code	Description	Schema


```
      "reason": "string",
      "note": "string",
      "image": "string",
      "timeStamp": "string",
      "signature": {
       "imageSignature": "string",
       "name": "string",
       "timeStamp": "string"
      }
     }
    }
   ],
   "isRoundTrip": "boolean",
   "isCloseJobPhotoRequired": "boolean",
   "promotion": {
    "_id": "string",
    "name": {
     "en": "string",
     "th": "string",
     "localized": "string"
    },
    "code": "string",
    "type": "string",
    "value": "number",
    "desc": {
     "en": "string",
     "th": "string",
     "localized": "string"
    },
    "imgUrl": "string",
```

Code	Description	Schema


```
  "coverImgUrl": "string",
  "startDate": "string",
  "endDate": "string"
 },
 "note": "string",
 "taxInfo": {
  "_id": "string",
  "name": "string",
  "address": "string",
  "email": "string",
  "tel": "string",
  "taxIDNo": "string",
  "branch": "string",
  "action": "string"
 },
 "byDistancePrice": "number",
 "discountPrice": "number",
 "addedServicePrice": "number",
 "roundTripPrice": "number",
 "totalPrice": "number",
 "totalPriceRider": "number",
 "totalPricePlatform": "number",
 "deductRiderCredit": "number",
 "totalDistance": "number",
 "totalTime": "number",
 "payment": {
  "_id": "string",
  "name": {
   "en": "string",
   "th": "string",
```

Code	Description	Schema


```
    "localized": "string"
   },
   "imgUrl": "string",
   "paymentId": "string",
   "isCashOnDelivery": "boolean"
  },
  "user": {
   "_id": "string",
   "firstName": "string",
   "lastName": "string",
   "mobileNo": "string",
   "email": "string"
  },
  "rider": {
   "_id": "string",
   "riderId": "string",
   "firstName": "string",
   "lastName": "string",
   "mobileNo": "string",
   "imgUrl": "string",
   "email": "string"
  },
  "status": {
   "_id": "string",
   "createBy": "string",
   "userId": "string",
   "status": "string",
   "reason": "string",
   "note": "string",
   "image": "string",
```

Code	Description	Schema

		"timeStamp": "string",
		"signature": {
		"imageSignature": "string",
		"name": "string",
		"timeStamp": "string"
		}
		},
		"rating": {
		"_id": "string",
		"userId": "string",
		"orderId": "string",
		"reason": "string",
		"rate": "number",
		"riderId": "string",
		"note": "string",
		"timeStamp": "string"
		},
		"createDate": "string"
		}
		]
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.6 GET /order/rider/available

Description

get rider available order list

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": number "description": "string", "data": [{ "_id": "string", "services": [{ "location": { "name": "string" } }, { "location": {

Code	Description	Schema

		"name": "string"
		}
		},
		{
		"location": {
		"name": "string"
		}
		}
		],
		"payment": {
		"_id": "string",
		"name": {
		"en": "string",
		"th": "string"
		},
		"priority": number,
		"imgUrl": "string"
		},
		"totalTime": number
		"totalDistance": number
		"totalPriceRider": number,
		"totalPrice": number
		"startTime": "string",
		"refCode": "string"
		}
		]}

default	Unexpected error	{ "code": "number", "description": "string", "data": "string" }

11.7 GET /order/rider/{orderId}

Description

get order by id

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": 20000, "description": "string, "data": [{ "_id": "5933b145b9520515d806591d", "status": { "_id": "5933b14fb9520515d8065921", "createBy": "user", "userId": "58c37bed01cde5264454339e", "status": "confirm", "signature": { "timeStamp": "2017-06-04T07:05:51.835Z",

Code	Description	Schema


```
    "name": null,
    "image": null
   },
   "timeStamp": "2017-06-04T07:05:51.834Z",
   "image": null,
   "note": null
  },
  "isReserveInAdvance": false,
  "type": "M",
  "rider": null,
  "user": {
   "_id": "58c37bed01cde5264454339e"
  },
  "retryCounter": 0,
  "default": [

  ],
  "serviceAreas": [
   "58c585ea712bcf878edda22f",
   "58c586a0712bcf878edda27b"
  ],
  "createDate": "2017-06-04T07:05:51.834Z",
  "services": [
   {
    "_id": "5933b144b9520515d8065917",
    "contactNo": "111",
    "contactName": "test1",
    "name": "string,
    "addedServices": [
    ],
```

Code	Description	Schema


```
"status": null,
"totalTime": 0,
"totalDistance": 0,
"totalPrice": 0,
"endTime": null,
"startTime": null,
"note": "",
"location": {
  "longitude": 100.72002332658,
  "latitude": 14.01197249068,
  "zipcode": "12110",
  "province": "string,
  "address": "string,
  "name": "test1",
  "coordinates": [
    100.72002332658,
    14.01197249068
  ],
  "type": "Point"
 }
},
{
 "_id": "5933b144b9520515d8065919",
 "contactNo": "333",
 "contactName": "tesr3",
 "name": "string,
 "addedServices": [

 ],
 "status": null,
```

Code	Description	Schema

"totalTime": 2812,
"totalDistance": 36908,
"totalPrice": 0,
"endTime": null,
"startTime": null,
"note": "",
"location": {
"longitude": 100.54929234087,
"latitude": 13.783461094536,
"zipcode": "10400",
"province": "string",
"address": "string",
"name": "test3",
"coordinates": [
100.54929234087,
13.783461094536
],
"type": "Point"
}
},
{
"_id": "5933b144b9520515d806591b",
"contactNo": "222",
"contactName": "test2",
"name": "string",
"addedServices": [

],
"status": null,
"totalTime": 3155,

Code	Description	Schema


```
    "totalDistance": 41794,
    "totalPrice": 0,
    "endTime": null,
    "startTime": null,
    "note": "",
    "location": {
     "longitude": 100.71367386729,
     "latitude": 14.008560718544,
     "zipcode": "12110",
     "province": "string",
     "address": "string",
     "name": "test2",
     "coordinates": [
      100.71367386729,
      14.008560718544
     ],
     "type": "Point"
    }
   }
  ],
  "rating": null,
  "payment": {
   "_id": "58a1ba731c16a9a477684d60",
   "name": {
    "en": "string",
    "th": "string"
   },
   "priority": 5,
   "imgUrl": null
  },
```

Code	Description	Schema


```
    "promotion": null,
    "taxInfo": {
      "branch": "knm",
      "tel": "0988344177",
      "email": "pichai_test@hotmail.com",
      "address": "w93",
      "name": "wincom",
      "taxIDNo": "1023938383o393"
    },
    "note": "",
    "totalTime": 6867,
    "totalDistance": 79000,
    "deductRiderCredit": 0,
    "totalPricePlatform": 381.5,
    "totalPriceRider": 708.5,
    "totalPrice": 1090,
    "discountPrice": 0,
    "netPrice": 1090,
    "roundTripPrice": null,
    "addedServicePrice": 0,
    "byDistancePrice": 1090,
    "riderEndTime": null,
    "riderStartTime": null,
    "endTime": "2017-06-04T09:00:07.688Z",
    "startTime": "2017-06-04T07:05:40.688Z",
    "isCloseJobPhotoRequired": false,
    "isRoundTrip": false,
    "refCode": "M100353"
  }
]
```

Code	Description	Schema
		}
default	Unexpected error	{ "code": "number", "description": "string", "data": "string"}

11.8 POST /order/user/confirm/{orderId}

Description

user confirm order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string
body	body		Yes	{ "_id": "string", "name": { "en": "string", "th": "string", "localized": "string" }, "imgUrl": "string", "paymentId": "string", "isCashOnDelivery": "boolean" }

Responses

Code	Description	Schema

200	succsess	{
		"code": "number",
		"description": "string",
		"data": {
		"url": "string",
		"checkUrl": "string"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.9 POST /order/user/retry

Description

user retry order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
body	body		Yes	{ "_id": "string", "refCode": "string", "type": "string", "isReserveInAdvance": "boolean", "startTime": "string", "endTime": "string", "riderStartTime": "string", "riderEndTime": "string", "services": [{ "_id": "string", "name": "string", "location": { "latitude": "number", "longitude": "number", "type": "string",

Name	Located In	Description	Required	Schema


```
    "coordinates": [
      "number"
    ],
    "name": "string",
    "address": "string",
    "zipcode": "string",
    "province": "string"
  },
  "contactName": "string",
  "contactNo": "string",
  "note": "string",
  "addedServices": [
    {
      "_id": "string",
      "name": {
        "en": "string",
        "th": "string",
        "localized": "string"
      },
      "priceType": "string",
      "price": "number",
      "note": "string",
      "serviceType": "string",
      "isShow": "boolean",
      "totalPrice": "number"
    }
  ],
  "startTime": "string",
  "endTime": "string",
  "totalPrice": "number",
```

Name	Located In	Description	Required	Schema


```
    "totalDistance": "number",
    "totalTime": "number",
    "status": {
      "_id": "string",
      "createBy": "string",
      "userId": "string",
      "status": "string",
      "reason": "string",
      "note": "string",
      "image": "string",
      "timeStamp": "string",
      "signature": {
        "imageSignature": "string",
        "name": "string",
        "timeStamp": "string"
      }
    }
  }
 ],
 "isRoundTrip": "boolean",
 "isCloseJobPhotoRequired":
"boolean",
 "promotion": {
  "_id": "string",
  "name": {
   "en": "string",
   "th": "string",
   "localized": "string"
  },
  "code": "string",
```

Name	Located In	Description	Required	Schema


```
  "type": "string",
  "value": "number",
  "desc": {
    "en": "string",
    "th": "string",
    "localized": "string"
  },
  "imgUrl": "string",
  "coverImgUrl": "string",
  "startDate": "string",
  "endDate": "string"
},
"note": "string",
"taxInfo": {
  "_id": "string",
  "name": "string",
  "address": "string",
  "email": "string",
  "tel": "string",
  "taxIDNo": "string",
  "branch": "string",
  "action": "string"
},
"byDistancePrice": "number",
"discountPrice": "number",
"addedServicePrice": "number",
"roundTripPrice": "number",
"totalPrice": "number",
"totalPriceRider": "number",
"totalPricePlatform": "number",
```

Name	Located In	Description	Required	Schema


```
"deductRiderCredit": "number",
"totalDistance": "number",
"totalTime": "number",
"payment": {
  "_id": "string",
  "name": {
    "en": "string",
    "th": "string",
    "localized": "string"
  },
  "imgUrl": "string",
  "paymentId": "string",
  "isCashOnDelivery": "boolean"
},
"user": {
  "_id": "string",
  "firstName": "string",
  "lastName": "string",
  "mobileNo": "string",
  "email": "string",
  "accessToken": "string",
  "imgUrl": "string",
  "notificationUnRead":
"number",
  "company": [
    {
      "_id": "string",
      "name": "string",
      "address": "string",
      "email": "string",
```

Name	Located In	Description	Required	Schema


```
      "tel": "string",
      "taxIDNo": "string",
      "branch": "string",
      "action": "string"
    }
  ],
  "needNotification": "boolean",
  "orderDoingAmount": "number"
},
"rider": {
  "_id": "string",
  "riderId": "string",
  "firstName": "string",
  "lastName": "string",
  "mobileNo": "string",
  "licensePlate": "string",
  "bankAccountImg": "string",
  "licensePlateImg": "string",
  "imgUrl": "string",
  "idCardImgUrl": "string",
  "driverLicenseImgUrl":
"string",
  "email": "string",
  "accessToken": "string",
  "bankAccount": {
    "bankBranch": "string",
    "bankAccountName": "string",
    "bankAccountNo": "string",
    "bankName": "string"
  },
```

Name	Located In	Description	Required	Schema


```
    "serviceArea": "string",
    "needNotification": "boolean",
    "credit": "number",
    "notificationUnRead":
"number",
    "totalJobPercent": "number",
    "readyForWorkStatus":
"boolean",
    "rating": "number",
    "driverLicenseExpireDate":
"string",
    "licensePlateExpireDate":
"string",
    "idCardExpireDate": "string"
  },
  "status": {
    "_id": "string",
    "createBy": "string",
    "userId": "string",
    "status": "string",
    "reason": "string",
    "note": "string",
    "image": "string",
    "timeStamp": "string",
    "signature": {
      "imageSignature": "string",
      "name": "string",
      "timeStamp": "string"
    }
  },
```

Name	Located In	Description	Required	Schema

				"rating": {
				"_id": "string",
				"userId": "string",
				"orderId": "string",
				"reason": "string",
				"rate": "number",
				"riderId": "string",
				"note": "string",
				"timeStamp": "string"
				},
				"createDate": "string"
				}

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

default	Unexpected error	{ "code": "number", "description": "string", "data": "string" }

11.10 POST /order/rider/accept/{orderId}

Description

rider accept order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string

Responses

Code	Description	Schema

200		{
		"code": "number",
		"description": "string",
		"data": "string"
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.11 POST /order/rider/start/{orderId}

Description

rider accept order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.12 POST /order/rider/cancel

Description

rider cancel order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	formData	orderId	Yes	string
reason	formData		Yes	string
note	formData		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": "string" }
default	Unexpected error	{ "code": "number", "description": "string", "data": "string" }

11.13 POST /order/user/cancel

Description

user cancel order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	formData	orderId	Yes	string
reason	formData		Yes	string
note	formData		No	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

Code	Description	Schema

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.14 POST /order/rider/update

Description

rider update order

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	formData	orderId	Yes	string
serviceId	formData	serviceId	Yes	string
latitude	formData		Yes	string
longitude	formData		Yes	string
reason	formData		Yes	string
note	formData		No	string
isCloseJob	formData		Yes	boolean
image	formData		No	file
imageSignature	formData		No	file
name	formData		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "refCode": "string", "type": "string", "isReserveInAdvance": "boolean", "startTime": "string", "endTime": "string", "riderStartTime": "string", "riderEndTime": "string", "services": [{ "_id": "string", "name": "string", "location": { "latitude": "number", "longitude": "number", "type": "string", "coordinates": ["number"], "name": "string", "address": "string", "zipcode": "string", "province": "string" }, "contactName": "string",

Code	Description	Schema


```
"contactNo": "string",
"note": "string",
"addedServices": [
  {
    "_id": "string",
    "name": {
      "en": "string",
      "th": "string",
      "localized": "string"
    },
    "priceType": "string",
    "price": "number",
    "note": "string",
    "serviceType": "string",
    "isShow": "boolean",
    "totalPrice": "number"
  }
],
"startTime": "string",
"endTime": "string",
"totalPrice": "number",
"totalDistance": "number",
"totalTime": "number",
"status": {
  "_id": "string",
  "createBy": "string",
  "userId": "string",
  "status": "string",
  "reason": "string",
  "note": "string",
```

Code	Description	Schema


```
      "image": "string",
      "timeStamp": "string",
      "signature": {
        "imageSignature": "string",
        "name": "string",
        "timeStamp": "string"
      }
    }
  }
],
"isRoundTrip": "boolean",
"isCloseJobPhotoRequired": "boolean",
"promotion": {
  "_id": "string",
  "name": {
    "en": "string",
    "th": "string",
    "localized": "string"
  },
  "code": "string",
  "type": "string",
  "value": "number",
  "desc": {
    "en": "string",
    "th": "string",
    "localized": "string"
  },
  "imgUrl": "string",
  "coverImgUrl": "string",
  "startDate": "string",
```

Code	Description	Schema


```
  "endDate": "string"
},
"note": "string",
"taxInfo": {
  "_id": "string",
  "name": "string",
  "address": "string",
  "email": "string",
  "tel": "string",
  "taxIDNo": "string",
  "branch": "string",
  "action": "string"
},
"byDistancePrice": "number",
"discountPrice": "number",
"addedServicePrice": "number",
"roundTripPrice": "number",
"totalPrice": "number",
"totalPriceRider": "number",
"totalPricePlatform": "number",
"deductRiderCredit": "number",
"totalDistance": "number",
"totalTime": "number",
"payment": {
  "_id": "string",
  "name": {
    "en": "string",
    "th": "string",
    "localized": "string"
  },
```

Code	Description	Schema


```
  "imgUrl": "string",
  "paymentId": "string",
  "isCashOnDelivery": "boolean"
 },
 "user": {
  "_id": "string",
  "firstName": "string",
  "lastName": "string",
  "mobileNo": "string",
  "email": "string",
  "accessToken": "string",
  "imgUrl": "string",
  "notificationUnRead": "number",
  "company": [
   {
    "_id": "string",
    "name": "string",
    "address": "string",
    "email": "string",
    "tel": "string",
    "taxIDNo": "string",
    "branch": "string",
    "action": "string"
   }
  ],
  "needNotification": "boolean",
  "orderDoingAmount": "number"
 },
 "rider": {
  "_id": "string",
```

Code	Description	Schema


```
"riderId": "string",
"firstName": "string",
"lastName": "string",
"mobileNo": "string",
"licensePlate": "string",
"bankAccountImg": "string",
"licensePlateImg": "string",
"imgUrl": "string",
"idCardImgUrl": "string",
"driverLicenseImgUrl": "string",
"email": "string",
"accessToken": "string",
"bankAccount": {
 "bankBranch": "string",
  "bankAccountName": "string",
  "bankAccountNo": "string",
  "bankName": "string"
},
"serviceArea": "string",
"needNotification": "boolean",
"credit": "number",
"notificationUnRead": "number",
"totalJobPercent": "number",
"readyForWorkStatus": "boolean",
"rating": "number",
"driverLicenseExpireDate": "string",
"licensePlateExpireDate": "string",
"idCardExpireDate": "string"
},
"status": {
```

Code	Description	Schema

		"_id": "string",
		"createBy": "string",
		"userId": "string",
		"status": "string",
		"reason": "string",
		"note": "string",
		"image": "string",
		"timeStamp": "string",
		"signature": {
		"imageSignature": "string",
		"name": "string",
		"timeStamp": "string"
		}
		},
		"rating": {
		"_id": "string",
		"userId": "string",
		"orderId": "string",
		"reason": "string",
		"rate": "number",
		"riderId": "string",
		"note": "string",
		"timeStamp": "string"
		},
		"createDate": "string"
		}
		}

default	Unexpected error	{ "code": "number",

Code	Description	Schema
		"description": "string", "data": "string" }

11.15 POST /order/user/rate

Description

user give rate to rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
body	body		Yes	{ "_id": "string", "userId": "string", "orderId": "string", "reason": "string", "rate": "number", "riderId": "string", "note": "string", "timeStamp": "string" }

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": "string"

Code	Description	Schema
		}

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

11.16 GET /order/user/riderLocation/{orderId}

Description

User get rider location by order id

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string
orderId	path		Yes	string

Responses

Code	Description	Schema
200	success	{


```
"code": "number",
"description": "string",
"data": {
 "_id": "string",
 "riderId": {
  "_id": "string",
  "firstName": "string",
  "lastName": "string",
```

Code	Description	Schema


```
  "mobileNo": "string",
  "riderId": "string",
  "imgUrl": "string",
  "email": "string",
 },
"workingStatus": {
  "endTime": "string",
  "startTime": "string",
  "isReserveInAdvance": "boolean",
  "orderId": {
   "_id": "string",
   "refCode": "string",
  },
  "status": "string",
},
"isOnline": "boolean",
"location": {
  "type": "string",
  "coordinates": [
   "number",
```

Code	Description	Schema
		"number"], "latitude": "number", "longitude": "number" } } }
default	Unexpected error	{ "code": "number", "description": "string", "data": "string" }

12.1 GET /sol/banks

Description

get sol bank account

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "name": "string", "accounts": [{ "bankName": "string", "imgUrl": "string", "bankNo": "string" }] }

Code	Description	Schema
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

13.1 POST/users/login

Description

login

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header	en,th default th	No	string
type	formData	basic, facebook, line, instagram, google	Yes	string
username	formData		Yes	string
credential	formData		Yes	string

Responses

Code	Description	Schema
200	user object will be return in data object	{ "code": "number", "description": "string", "data": { "_id": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "email": "string", "accessToken": "string", "imgUrl": "string", "notificationUnRead": "number",

Code	Description	Schema

		"company": [
		{
		"_id": "string",
		"name": "string",
		"address": "string",
		"email": "string",
		"tel": "string",
		"taxIDNo": "string",
		"branch": "string",
		"action": "string"
		}
		],
		"needNotification": "boolean",
		"orderDoingAmount": "number"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

14.1 GET /serviceAreas

Description

get service Area list

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "groupName": { "en": "string", "th": "string", "localized": "string" }, "provinces": [{ "_id": "string", "name": { "en": "string", "th": "string",

Code	Description	Schema

		"localized": "string"
		},
		"areas": [
		{
		"name": "string",
		"zipcode": "string"
		}
		]
		}
		]
		}
		]

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

15.1 GET /faqs

Description

get faq

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "question": "string", "ans": "string",

Code	Description	Schema
		"no": "number" }] }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

15.2 GET /faqs/rider

Description

get faq rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string

Responses

Code	Description	Schema

200	success	<pre>{
		"code": "number",
		"description": "string",
		"data": [
		{
		"_id": "string",
		"question": "string",
		"ans": "string",
		"no": "number"
		}
		]
		}</pre>

Code	Description	Schema
default	Unexpected error	{ "code": "number", "description": "string" }

16.1 GET /versions/{userType}

Description

Get Version

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
userType	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "message": "string", "appId": "string", "version": "string", "platform": "string", "redirectUrl": "string", "userType": "string", "force": "boolean" } }

Code	Description	Schema

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

17.1 POST /riders/register

Description

register rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
firstName	formData		Yes	string
lastName	formData		Yes	string
mobileNo	formData		Yes	string
email	formData		Yes	string
transactionId	formData		Yes	string
otp	formData		Yes	string
type	formData		Yes	string
socialId	formData	require for type is social	No	string
socialToken	formData	require for type is social	No	string
password	formData	require for type is basic	No	string
driverLicenseNo	formData		No	string

Name	Located In	Description	Required	Schema
licensePlate	formData		No	string
motorcycleBand	formData		No	string
idCard	formData		No	string
pic	formData		No	file
driverLicenseImg	formData		No	file
idCardImg	formData		No	file
licensePlateImg	formData		No	file
bankAccountImg	formData		No	file
imgUrl	formData		No	string
driverLicenseExpireDate	formData		No	string
licensePlateExpireDate	formData		No	string
idCardExpireDate	formData		No	string
refFriendMobileNo	formData		No	string

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": {
		"_id": "string",

Code	Description	Schema


```
    "riderId": "string",
    "firstName": "string",
    "lastName": "string",
    "mobileNo": "string",
    "licensePlate": "string",
    "bankAccountImg": "string",
    "licensePlateImg": "string",
    "imgUrl": "string",
    "idCardImgUrl": "string",
    "driverLicenseImgUrl": "string",
    "email": "string",
    "accessToken": "string",
    "bankAccount": {
      "bankBranch": "string",
      "bankAccountName": "string",
      "bankAccountNo": "string",
      "bankName": "string"
    },
    "serviceArea": "string",
    "needNotification": "boolean",
    "credit": "number",
    "notificationUnRead": "number",
    "totalJobPercent": "number",
    "readyForWorkStatus": "boolean",
    "rating": "number",
    "driverLicenseExpireDate": "string",
    "licensePlateExpireDate": "string",
    "idCardExpireDate": "string"
  }
}
```

Code	Description	Schema
default	Unexpected error	{ "code": "number", "description": "string"}

17.2 POST /riders/login

Description

login rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
type	formData	basic, facebook, line, instagram, google	Yes	string
username	formData		Yes	string
credential	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "riderId": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "licensePlate": "string", "bankAccountImg": "string", "licensePlateImg": "string",

Code	Description	Schema

		"imgUrl": "string",
		"idCardImgUrl": "string",
		"driverLicenseImgUrl": "string",
		"email": "string",
		"accessToken": "string",
		"bankAccount": {
		"bankBranch": "string",
		"bankAccountName": "string",
		"bankAccountNo": "string",
		"bankName": "string"
		},
		"serviceArea": "string",
		"needNotification": "boolean",
		"credit": "number",
		"notificationUnRead": "number",
		"totalJobPercent": "number",
		"readyForWorkStatus": "boolean",
		"rating": "number",
		"driverLicenseExpireDate": "string",
		"licensePlateExpireDate": "string",
		"idCardExpireDate": "string"
		}
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

17.3 POST /riders/forgotPass

Description

forgotPass

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
mobileNo	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "transactionId": "string" } }
default	Unexpected error	{ "code": "number", "description": "string" }

17.4 POST /riders/updateProfile

Description

UpdateProfile rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header	accessToken	Yes	string
x-user	header	x-user	Yes	string
firstName	formData		Yes	string
lastName	formData		Yes	string
email	formData		No	string
driverLicenseNo	formData		No	string
licensePlate	formData		No	string
motorcycleBand	formData		No	string
idCard	formData		No	string
pic	formData		No	file
driverLicenseImg	formData		No	file
idCardImg	formData		No	file
licensePlateImg	formData		No	file
bankAccountImg	formData		No	file
bankAccount	formData		No	string
serviceArea	formData		No	string

Name	Located In	Description	Required	Schema
deviceId	formData		No	string
needNotification	formData		No	boolean
readyForWorkStatus	formData		No	boolean
driverLicenseExpireDate	formData		No	string
licensePlateExpireDate	formData		No	string
idCardExpireDate	formData		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "riderId": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "licensePlate": "string", "bankAccountImg": "string", "licensePlateImg": "string",

Code	Description	Schema
		"imgUrl": "string", "idCardImgUrl": "string", "driverLicenseImgUrl": "string", "email": "string", "accessToken": "string", "bankAccount": { "bankBranch": "string", "bankAccountName": "string", "bankAccountNo": "string", "bankName": "string" }, "serviceArea": "string", "needNotification": "boolean", "credit": "number", "notificationUnRead": "number", "totalJobPercent": "number", "readyForWorkStatus": "boolean", "rating": "number", "driverLicenseExpireDate": "string", "licensePlateExpireDate": "string",

Code	Description	Schema
		"idCardExpireDate": "string" } }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

17.5 GET /riders/

Description

rider get profile

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string

Responses

Code	Description	Schema

Code	Description	Schema
200	success	{

"code": "number",

"description": "string",

"data": {

"_id": "string",

"riderId": "string",

"firstName": "string",

"lastName": "string",

"mobileNo": "string",

"licensePlate": "string",

"bankAccountImg": "string",

"licensePlateImg": "string",

"imgUrl": "string",

"idCardImgUrl": "string",

"driverLicenseImgUrl": "string",

"email": "string",

"accessToken": "string",

"bankAccount": {

"bankBranch": "string",

"bankAccountName": "string",

Code	Description	Schema
		"bankAccountNo": "string", "bankName": "string" }, "serviceArea": "string", "needNotification": "boolean", "credit": "number", "notificationUnRead": "number", "totalJobPercent": "number", "readyForWorkStatus": "boolean", "rating": "number", "driverLicenseExpireDate": "string", "licensePlateExpireDate": "string", "idCardExpireDate": "string" } }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

17.6 GET /riders/income/day

Description

get income rider by day

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "totalIncome": "number", "totalOnline": "number", "income": [{ "_id": "string", "totalPricePlatform": "number", "totalPriceRider": "number", "totalPrice": "number", "refCode": "string", "type": "string"

Code	Description	Schema
		}] } }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

17.7 GET /riders/income/{filter}

Description

get income rider by filter weekday / month

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
filter	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "totalIncome": "number", "totalOnline": "number", "income": [{ "date": "string", "totalPriceRider": "number"

Code	Description	Schema
		}] } }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

17.8 GET /riders/income/{filter}/{date}

Description

get income rider by filter weekday / month all detail by date

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
filter	path		Yes	string
date	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "totalIncome": "number", "totalOnline": "number", "income": [{ "_id": "string", "totalPricePlatform": "number", "totalPriceRider": "number",

Code	Description	Schema
		"totalPrice": "number", "refCode": "string", "type": "string" }] } }

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}

17.9 POST /riders/resetPass

Description

Reset Password

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
mobileNo	formData		Yes	string
password	formData		Yes	string
otp	formData		Yes	string
transactionId	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

17.10 POST /riders/changePass

Description

Change Password

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
oldPassword	formData		Yes	string
newPassword	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

17.11 POST /riders/setNewPass

Description

Set Password

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
password	formData		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

17.12 POST /riders/logout

Description

Logout

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header	x-access-token	Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

17.13 POST /riders/changeMobileNo

Description

Rider request change mobile no

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header		Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
mobileNo	formData	New mobileNo for change	No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "transactionId": "string" } }
default	Unexpected error	{ "code": "number", "description": "string" }

17.14 POST /riders/updateMobileNo

Description

Call after changeMobileNo for change mobileNo rider

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header		Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
mobileNo	formData	New mobileNo for change	No	string
otp	formData		No	string
transactionId	formData		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": { "_id": "string", "riderId": "string", "firstName": "string", "lastName": "string", "mobileNo": "string", "licensePlate": "string",

Code	Description	Schema

		"bankAccountImg": "string",
		"licensePlateImg": "string",
		"imgUrl": "string",
		"idCardImgUrl": "string",
		"driverLicenseImgUrl": "string",
		"email": "string",
		"accessToken": "string",
		"bankAccount": {
		"bankBranch": "string",
		"bankAccountName": "string",
		"bankAccountNo": "string",
		"bankName": "string"
		},
		"serviceArea": "string",
		"needNotification": "boolean",
		"credit": "number",
		"notificationUnRead": "number",
		"totalJobPercent": "number",
		"readyForWorkStatus": "boolean",
		"rating": "number",
		"driverLicenseExpireDate": "string",
		"licensePlateExpireDate": "string",
		"idCardExpireDate": "string"
		}
		}

default	Unexpected error	{ "code": "number", "description": "string" }

17.15 POST /riders/topup/{paymentOptionId}

Description

Rider Topup by channel

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
x-access-token	header		Yes	string
x-user	header	x-user	Yes	string
lang	header		No	string
transferSlipImg	formData	Transfer Slip Image	Yes	file
paymentOptionId	path		Yes	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string" }
default	Unexpected error	{ "code": "number", "description": "string" }

18.1 POST /rating

Description

user give rate to rider

Parameters

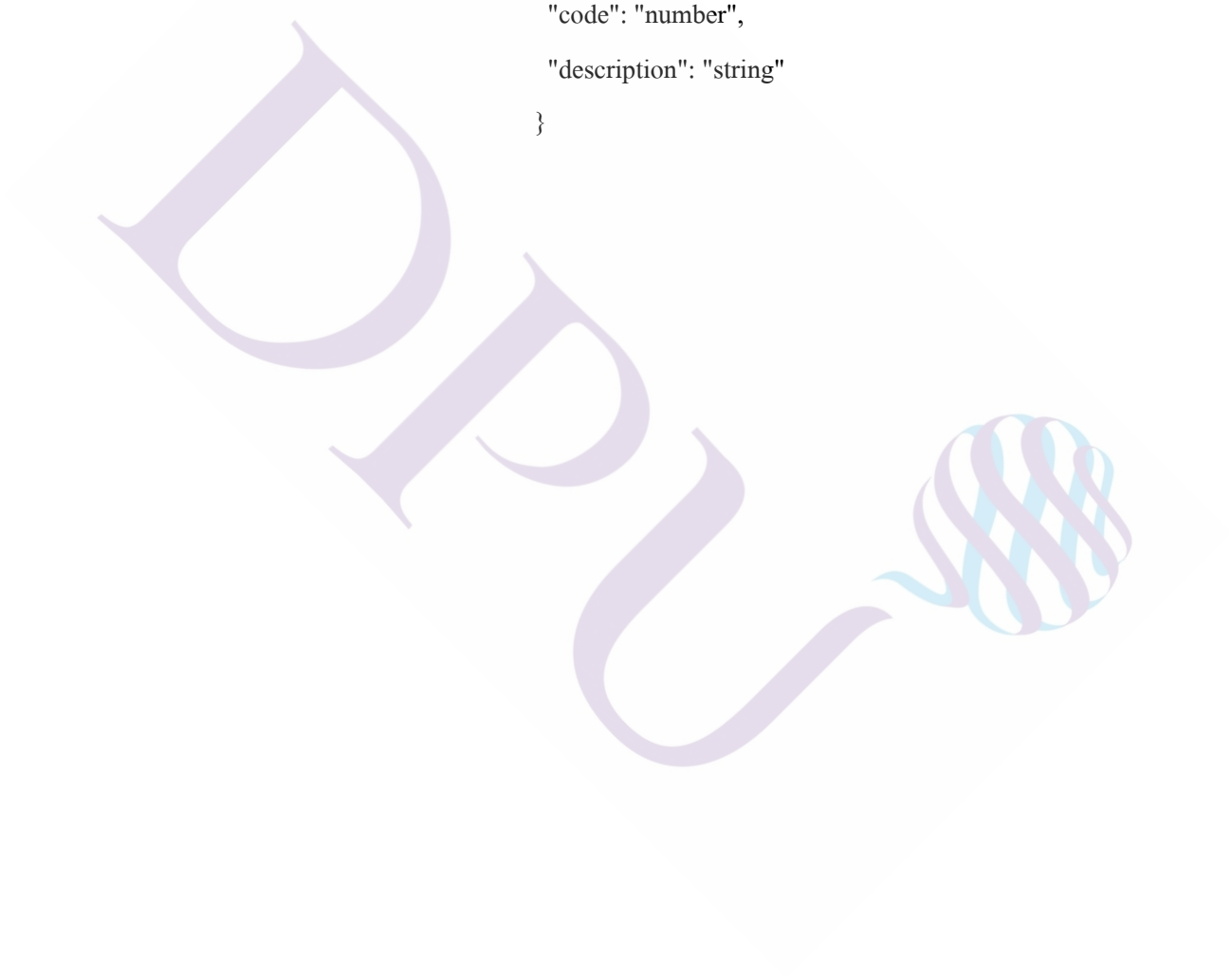
Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		No	string
x-access-token	header		Yes	string
x-user	header		No	string
body	body		Yes	{ "_id": "string", "userId": "string", "orderId": "string", "reason": "string", "rate": "number", "riderId": "string", "note": "string", "timeStamp": "string" }

Responses

Code	Description	Schema

200	success	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

default	Unexpected error	{
		"code": "number",
		"description": "string"
		}



18.2 GET /rating

Description

get rider ranking

Parameters

Name	Located In	Description	Required	Schema
Authorization	header	basic auth api	Yes	string
platform	header	platform	Yes	string
lang	header		Yes	string
x-access-token	header		Yes	string
x-user	header		No	string

Responses

Code	Description	Schema
200	success	{ "code": "number", "description": "string", "data": [{ "_id": "string", "userId": "string", "orderId": "string", "reason": "string", "rate": "number", "riderId": "string", "note": "string", "timeStamp": "string" }

Code	Description	Schema
		] }

default	Unexpected error	{
		"code": "number",
		"description": "string",
		"data": "string"
		}

ภาคผนวก ค

WebSocket Event

ตาราง ค WebSocket Event ที่ใช้งานในระบบ

No.	API		Description

1.	**User**	Connect with parameter
		- query (token, lang, orderId)
		Ex.
		{
		query: 'token=' + token + '&lang=' + 'en' + '&orderId=' + '58f27297135e8c207cb30bb6'
		}	

1.1	Register event	userTrackRiderLocationByOrder	ติดตามการทำงานของพนักงานที่กำลังทำคำสั่งงานอยู่
	Data		รับข้อมูลตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานที่กำลังปฏิบัติงานให้ผู้ใช้อยู่

2.	**Messenger**	Connect with parameter
		- query (token, lang)
		Ex.
		{
		query: 'token=' + token + '&lang=' + 'en'
		}	

2.1	Emit	riderUpdateLocation	พนักงานส่งตำแหน่ง locatotion ปัจจุบันมาอัพเดท

	Data	{
		'latitude': 13.75637532956007,
		'longitude': 100.5302087590098
		}
			ส่งข้อมูล Location (latitude, longitude)

ตาราง ค (ต่อ)

No.	API		Description
2.2	Register event	newConfirmOrder	รับ Feed งานที่มีการสั่งงานเข้ามาใหม่
	Data	Order	รับ Object ของคำสั่งงานในหน้า Feed งาน
2.3	Register event	removeAvailableOrder	ลบคำสั่งงานที่ถูกรับไปแล้วออกจากหน้า feed
	Data	"5953c1df13ad586ce9868be7"	Id ของคำสั่งงาน เมื่อ แอปพลิเคชันได้รับให้ลบ คำสั่งงานที่มี id ออกจากหน้า feed

3.	**Admin**	Connect with parameter
		- query (token, lang)
		Ex.
		{
		query: 'token=' + token + '&lang=' + 'en'
		}	

3.1	Register event	trackOnlineRidersLocation	ผู้ดูแลระบบติดตามการทำงานของพนักงาน

	Data	[
		{
		"_id":
		"5953c1df13ad586ce9868be7",
		"riderId": {
		"_id":
		"594c2b0c13ad586ce971fb43",
		"firstName": "Sipang",
			รับ array ข้อมูลของ ตำแหน่งปัจจุบันและสถานะการรับงานของพนักงานแต่ละคนในระบบ

ตาราง ค (ต่อ)

No.	API	Description

	"lastName": "Kanthasema",
	"mobileNo": "0860302215",
	"riderId": "SR10085",
	"readyForWorkStatus": true,
	"imgUrl": "http:\/\/cloud.thaikeeper...",
	"email": "sipangka511@gmail.com"
	},
	"lastOnline": "2017-06-28T21:06:58.486Z",
	"workingStatus": null,
	"isOnline": true,
	"location": {
	"type": "Point",
	"coordinates": [
	100.5457078,
	13.7828705
	],
	"latitude": 13.7828705,
	"longitude": 100.5457078
	},
	"lastUpdate": "2017-06-28T21:06:58.486Z"
	}
	]	

ตาราง ค (ต่อ)

No.	API		Description

4.	**System**	Connect with parameter
		- query (token, lang)
		Ex.
		{
		query: 'token=' + token + '&lang=' + 'en'
		}	

4.1	Emit	confirmOrder	เมื่อมีการสั่งงานเข้ามาใหม่ระบบส่งงานให้พนักงานที่ออนไลน์อยู่
	Data	Order	ส่ง Object ของคำสั่งงานออกไป
4.2	Emit	removeAvailableOrder	เมื่อมีพนักงานรับงานไปแล้วระบบ broadcast บอกพนักงานทุกคนที่ออนไลน์อยู่ว่าให้ลบคำสั่งงานนี้ออกจากหน้า Feed
	Data	"5953c1df13ad586ce9868be7"	ส่ง id ของคำสั่งงานออกไปให้ แอปพลิเคชันลบออกจากหน้า Feed

ประวัติผู้เขียน

ชื่อ-นามสกุล	นางสาวสิปางญ์ กันทะเสมา
ประวัติการศึกษา	ปีการศึกษา 2546 สำเร็จการศึกษาระดับปริญญาตรี สาขาวิชาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยศิลปากร
ตำแหน่งและสถานที่ทำงานปัจจุบัน	ผู้ชำนาญการด้านเทคโนโลยีสารสนเทศ บริษัทไมโม่เทคจำกัด