



โครงการ  
เรื่อง Tramer - Tram Tracking Application

โดย

กลุ่ม Epsilon

นายณัฐวีร์	เกิงฝาก	6213125
นายพสวัต	แตงอ่อน	6213129
นายกันตพัฒน์	เทียนธนวินท์	6213194
นายชุติวัต	วัตรเอก	6213195

เสนอ

อาจารย์ ฉนัท พูลสวัสดิ์

โครงการนี้เป็นส่วนหนึ่งของวิชา

EGCO321 Database System และ EGCO 343 Software Design

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยมหิดล

ภาคเรียนที่ 1 ปีการศึกษา 2564

## คำนำ

รายงานฉบับนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของวิชา Software Designs (EGCO343) เพื่ออธิบายเกี่ยวกับ Software Architecture, การจัดเก็บข้อมูลใน Database, Business Process, Services ต่าง ๆ ที่ใช้ ตลอดไปจนถึง UX/UI Design ของ Tramer Platform ทั้งนี้ก็เพื่อให้ เกิดความเข้าใจในกระบวนการขั้นตอนต่าง ๆ ในการออกแบบ Software

ผู้จัดทำหวังว่า รายงานเล่มนี้จะเป็นประโยชน์กับผู้อ่าน หรือนักเรียน นักศึกษา ที่กำลังหา ข้อมูลตัวอย่างเกี่ยวกับการออกแบบ Software

คณะผู้จัดทำ

## สารบัญ

คำนำ.....	๑
สารบัญ.....	๒
บทนำ.....	๓
ทีมและความสำคัญ.....	๔
วัตถุประสงค์.....	๕
User Story.....	๖
เครื่องมือที่ใช้งาน.....	๗
Google Maps APIs.....	๗
Firebase .....	๘
Software Analysis.....	๙
Main layer.....	๙
Layer 0 .....	๑๐
Layer 1 .....	๑๑
Layer 5 .....	๑๒
Layer 5.2.....	๑๓
Layer 5.3.....	๑๔
Layer 6 .....	๑๕
Software Architecture.....	๑๖
Front-End (UX/UI Design).....	๑๗
ส่วนของ User.....	๑๗
1. หน้าหลัก[User] .....	๑๘
2. สายการเดินรถ(Fliter) .....	๑๙
3. ป้ายถัดไป(Next Stop).....	๒๐
4. คันถัดไป(Next Tram).....	๒๑
5. ค้นหาเส้นทาง(Search) .....	๒๒
6. แจ้งปัญหา(Feedback) .....	๒๓
7. เข้าสู่ระบบ(Login) .....	๒๔
ส่วนของ Driver .....	๒๕
1. หน้าหลัก[Driver].....	๒๖
2. QR Code Scanner.....	๒๗
3. ขับรถ(Driving).....	๒๘

## สารบัญ

ส่วนของ Admin.....	24
1. หน้า Sign in .....	24
2. หน้า Home Page .....	25
3. แถบ Menu.....	26
4. หน้า Personal information .....	26
5. หน้า Edit Tram information .....	27
6. หน้า Edit bus stop information .....	29
7. หน้า Database.....	31
Back-End (Services).....	34
ส่วนของ User.....	34
ส่วนของ Driver .....	40
ส่วนของ Admin.....	42
Database.....	44
Database : service_provider.....	44
Database : tram .....	45
Database : feedback.....	47
Database : one_day .....	48
Database : log_service.....	49
Database : log_user.....	50
ข้อมูลที่เกี่ยวข้อง.....	53
บรรณานุกรม .....	54

## บทนำ

Tramer เป็นแพลตฟอร์ม สำหรับอำนวยความสะดวกในการเดินทางและวางแผนการเดินทางด้วยรถแทรเมร์ในมหาวิทยาลัยมหิดล ผู้ใช้จะสามารถ ตรวจสอบตารางเวลา และ เส้นทางการเดินรถแทรเมร์ อีกทั้งยังสามารถ ค้นหาป้ายสถานีรถแทรเมร์ใกล้ ๆ สถานที่ต่าง ๆ ค้นหาเส้นทางในการเดินทาง คำนวณเวลาการเดินทาง ของรถแทรเมร์ โดยการตีงข้อมูลจาก google map api เพื่อช่วยเหลือผู้ที่ต้องการวางแผนการเดินทาง และ ผู้ที่ไม่ชำนาญเส้นทางและสถานที่ต่าง ๆ ในมหาวิทยาลัยมหิดลนั้น สามารถ เดินทางได้ง่ายและสะดวกมากขึ้น

## ที่มาและความสำคัญ

ปัจจุบันการเดินทางด้วยรถแทรมในมหาวิทยาลัยยังอำนวยความสะดวกไม่ได้มากนักสำหรับนักศึกษาที่ไม่ทราบรายละเอียดสถานที่ต่าง ๆ ในมหาวิทยาลัยมหิดล การขึ้นและลงรถแทรมในจุดต่าง ๆ และผู้ที่ต้องการคำนวณเวลาในการเดินทางเพื่อวางแผนการเดินทาง ทั้งนี้ก็เพื่ออำนวยความสะดวกให้กับการเดินทางด้วยรถแทรมในมหาวิทยาลัยมหิดล ผู้จัดทำจึงคิดค้นแพลตฟอร์ม Tramer ที่จะมาอำนวยความสะดวกสำหรับการเดินทางด้วยรถแทรมในมหาวิทยาลัยมหิดล ช่วยค้นหาป้ายสถานีรถแทรมใกล้ ๆ สถานที่ต่าง ๆ ช่วยค้นหาเส้นทางกับสายที่ขึ้นจุดที่ลง พร้อมทั้งคำนวณเวลาการมาถึงและไปยังจุดหมาย ต่าง ๆ ของเส้นทางการเดินรถ

## วัตถุประสงค์

เป็นตัวช่วยสำหรับการเดินทางด้วยรถแทรมในมหาวิทยาลัยมหิดล เพื่อให้สามารถตรวจสอบเส้นทางและตารางเวลาของรถแท้ม เพื่อให้สามารถค้นหาป้ายรถแทรมที่ใกล้สถานที่ต่าง ๆ เพื่อให้สามารถค้นหาเส้นทางการนั่งรถและคำนวณเวลาที่ใช้ในการเดินทาง

## User Story

- ผู้โดยสารสามารถทราบตำแหน่งของรถแทรม เพื่อช่วยในการตัดสินใจขึ้นรถ
- ผู้โดยสารสามารถทราบตารางเวลาของรถแทรม เพื่อคำนวณเวลาที่ใช้ในการรอรถแทรมเพื่อเดินทาง
- ผู้โดยสารสามารถทราบสายรถที่ต้องขึ้นจากต้นทางไปปลายทาง
- ผู้โดยสารสามารถทราบแพนที่การเดินทางของรถทุกสาย เพื่อวางแผนการเดินทาง
- ผู้โดยสารสามารถทราบตำแหน่งของป้ายรถแทรมที่ใกล้ที่สุด เพื่อที่จะสามารถทราบสถานที่ ๆ จะไปรอรถ
- ผู้โดยสารสามารถทราบว่ารถแทรมจะมาถึงและไปถึงป้ายที่เลือกในอีก กี่นาที เพื่อที่จะสามารถเตรียมตัวขึ้นและลงรถได้
- ผู้โดยสารสามารถบวชีการใช้แอพได้ เพื่อที่จะสามารถใช้แอพได้อย่างง่ายดาย
- ผู้โดยสารสามารถแจ้งปัญหาการใช้แอพได้ เพื่อที่ผู้ดูแลจะได้นำไปปรับปรุงแอพต่อไป
- คนขับรถแทรมสามารถส่งตำแหน่ง GPS เพื่อใช้ในฟังก์ชันต่าง ๆ ของแอพ
- คนขับรถแทรมสามารถบันทึกเวลาเดินรถเพื่อกেย์ไปเป็นประวัติการเดินรถ
- คนขับรถแทรมสามารถดูข้อมูลส่วนตัวของคนขับได้
- คนขับรถแทรมสามารถอัพเดทป้ายรถแทรมที่ไปถึงได้
- คนขับรถแทรมและผู้ดูแลระบบสามารถเข้าและออกงานได้ เพื่อบันทึกเวลาการทำงาน
- ผู้ดูแลระบบสามารถเรียกดูและแก้ไขข้อมูลในระบบได้ เพื่อดูตามและทำให้ข้อมูลถูกต้อง เป็นปัจจุบัน
- ผู้ดูแลระบบสามารถบันทึกและเรียกดู ประวัติใช้งานแอพและการแก้ไขข้อมูลต่าง ๆ ได้

## เครื่องมือที่ใช้งาน

# Google Maps APIs

1. ในส่วนของแผนที่จะแก้ไขนิดของแผนที่ จะใช้จัดการศูนย์กลางและกำหนดขีดจำกัดของการย่อขยายแผนที่
2. ใช้กำหนดต่อคอมมาร์คเกอร์สำหรับสถานที่และสถานีต่าง ๆ
3. ใช้คำนวณและแสดง เส้นทางกับระยะเวลาในการเดินทางของรถ
4. ใช้เพื่อให้สามารถค้นหาสถานที่ได ๆ ในม.มหิดล ในการค้นหาป้ายสถานีรถแทรม ใกล้ ๆ



Firebase คือฐานข้อมูลแบบ NoSQL โดยจะไม่ใช้ภาษา SQL ในการจัดการข้อมูล แต่ออกแบบให้มีความยืดหยุ่นและเน้นความเร็วในการใช้งาน ซึ่งมีการเก็บข้อมูลแบบ JSON โดยที่มีตารางเหมือนกับ SQL แต่ไม่มีคอลัมน์ ในหนึ่งแวรสามารถเก็บข้อมูลได้ทั้งข้อความ (String) ตัวเลข (Number) รวมไปถึง Object อื่น ๆ สถาเหตุที่ใช้

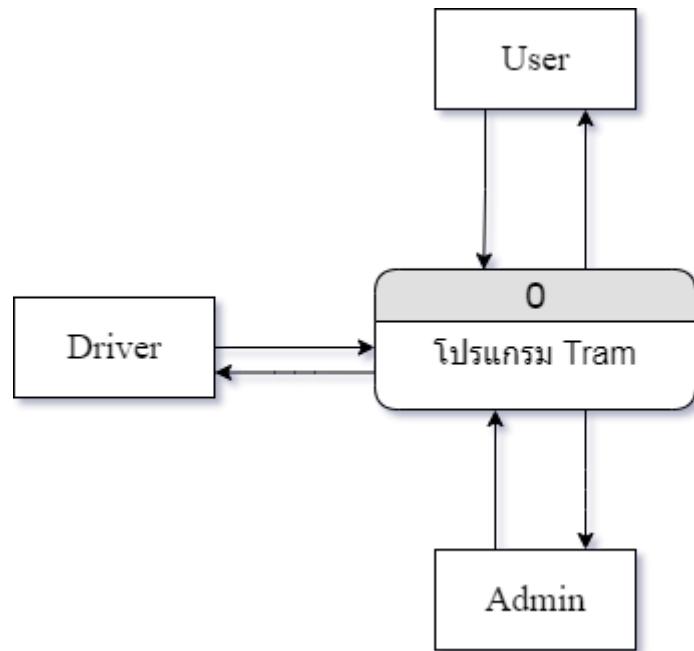
- Hosting ฟรี
- ใช้งานง่าย
- มี documentation ชัดเจน
- Realtime Database
- มี google analytics และ Crashlytics

## Software Analysis

โดยจะทำการอธิบายส่วน Software analysis ในรูปของ Dataflow โดยจะมีรูปแบบ ดังนี้

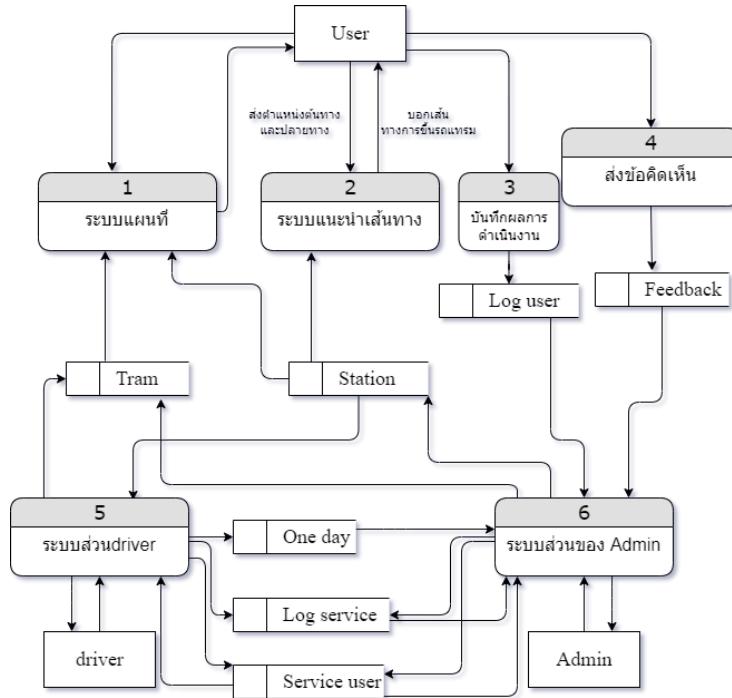
### Main layer

โดยจะแสดงถึงตัวละครทั้งหมดที่ใช้โปรแกรมและการเคลื่อนย้ายของข้อมูล โดยจะมี 3 ตัวละครคือ User, Driver และ Admin



## Layer 0

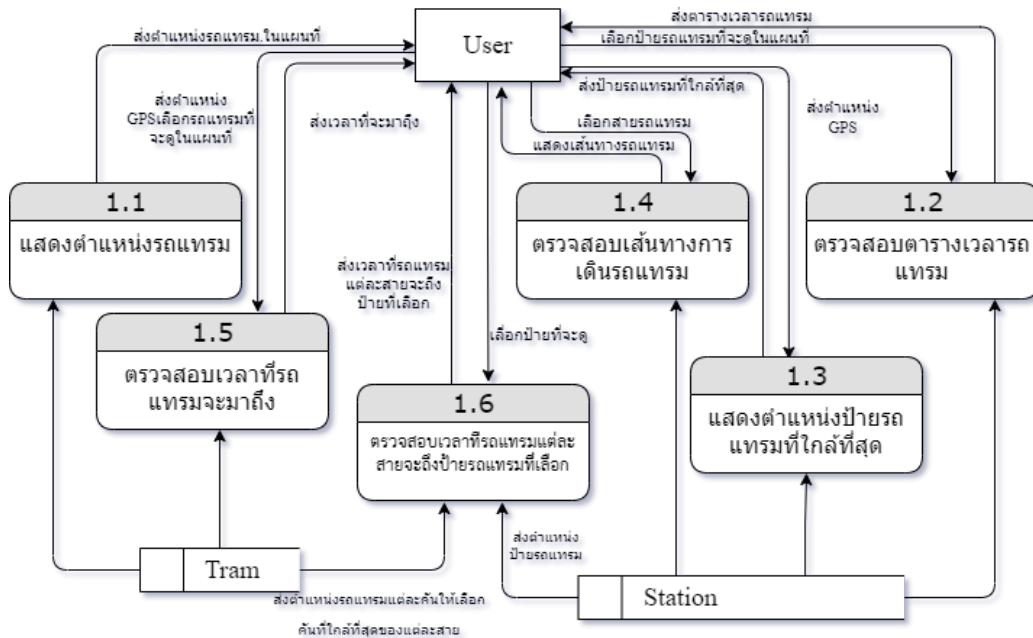
จะแสดงถึงภาพรวมทั้งหมดของการเคลื่อนไหวของข้อมูลทั้งระบบ โดยมีรูปแบบดังนี้



- ระบบแผนที่ เป็นระบบผู้ใช้งาน User ที่จะแสดงการเคลื่อนไหวของข้อมูลในฟังก์ชันที่ใช้ในฟังก์ชันแผนที่ โดยจะใช้ข้อมูลจาก Database Tram และ Station ในการทำงานของฟังก์ชัน
- ระบบแนะนำเส้นทาง เป็นระบบผู้ใช้งาน User ที่จะแนะนำเส้นทางการเดินทางจากจุดเริ่มต้นไปจุดปลายทาง โดยจะให้ User ส่งตำแหน่งเริ่มต้นและปลายทาง ระบบจะทำการตีดีงข้อมูลจาก Station เพื่อทำการประมวลผลหาเส้นทางที่ดีที่สุดแล้วส่งแสดงผลให้ User ต่อไป
- บันทึกผลการดำเนินงาน เป็นระบบผู้ใช้งาน User ที่จะเก็บการใช้งานทั้งหมดของผู้ใช้งาน และนำไปเก็บใน Log user
- ส่งข้อคิดเห็น เป็นระบบผู้ใช้งาน User ที่จะให้ User ส่งข้อคิดเห็นการใช้งานแอปแล้วนำไปเก็บไว้ใน Feedback
- ระบบส่วน Driver เป็นระบบโดยรวมของผู้ใช้งาน Driver ที่จะรวมการทำงานต่างๆของผู้ใช้งาน Driver
- ระบบส่วนของ Admin เป็นระบบโดยรวมของผู้ใช้งาน Admin ที่จะรวมการทำงานต่างๆของผู้ใช้งาน Admin

## Layer 1

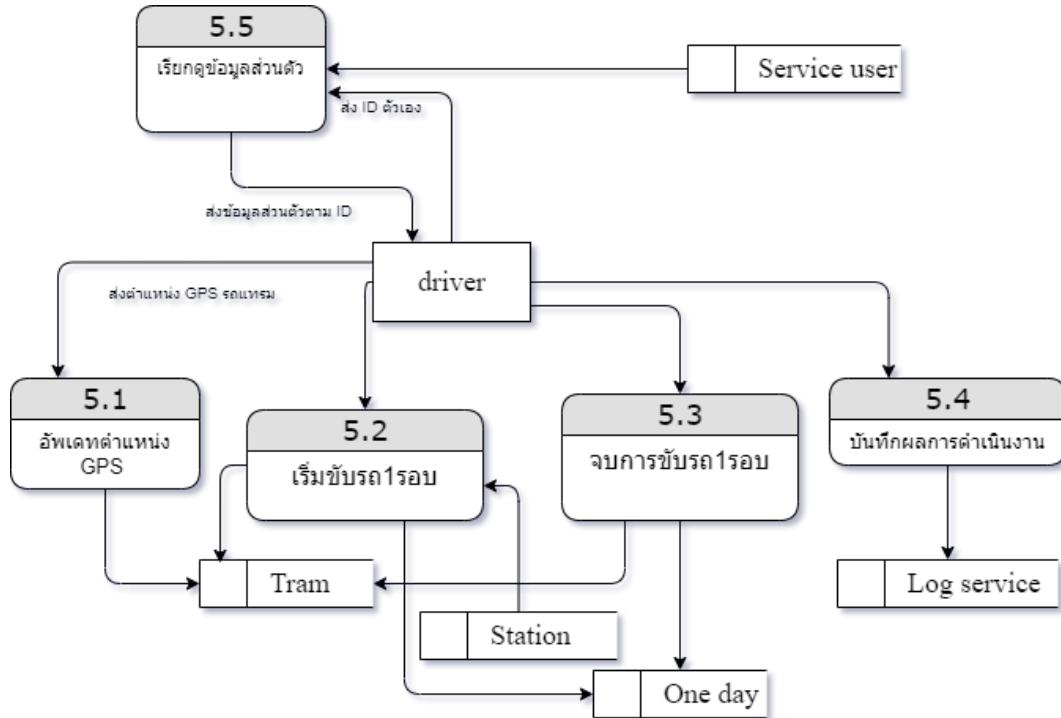
อธิบายการทำงานทั้งหมดในส่วนระบบแผนที่จากใน Layer 0 โดยมีรูปแบบดังรูป



- 1.1. แสดงตำแหน่งรถแทรม โดยจะแสดงตำแหน่งรถแทรมในแผนที่เป็นจุดๆ โดยจะขอตำแหน่งรถแทรมจาก Tram เพื่อนำมาแสดงผล
- 1.2. ตรวจสอบตารางเวลารถแทรม โดย User จะเลือกดูตารางเวลาจาก การกรดที่ป้ายรถแทรม และนำข้อมูลตารางเวลาจาก Station มาแสดงผล
- 1.3. แสดงตำแหน่งรถแทรมที่ใกล้ที่สุด เป็นการแนะนำให้ User รู้ว่าป้ายรถแทรมที่ใกล้ที่สุดคืออะไร โดยทำการนำตำแหน่ง GPS ของผู้ใช้ และตำแหน่งของป้ายรถแทรมมาประมวลผล หาป้ายที่ใกล้ที่สุดแล้วแสดงผลในแผนที่
- 1.4. ตรวจสอบเส้นทางการเดินรถแทรม เป็นการแสดงเส้นทางการเดินรถแทรมแต่ละสายโดยให้ User เลือกสายที่จะดูและใช้ข้อมูล line จาก Station ที่ต่อ Station เป็นสายและทำมาแสดงผลในแผนที่
- 1.5. ตรวจสอบเวลาการมาถึงของรถแทรม จะตรวจสอบว่ารถแทรมจะมาถึงป้ายที่ User รอในกี่นาทีโดยดังข้อมูลตำแหน่ง GPS ของ User และรถแทรมที่เลือกแล้วนำมาคำนวณเวลา
- 1.6. ตรวจสอบเวลาที่รถแทรมแต่ละสายจะถึงป้ายรถแทรมที่เลือก จะตรวจสอบว่ารถแทรมแต่ละสายจะมาถึงป้ายรถแทรมที่เลือกในกี่นาทีโดยดังข้อมูลตำแหน่ง GPS ของ Tram คันลัดไปที่จะผ่านของแต่ละสายและ Station ที่เลือกแล้วนำมาคำนวณเวลา

## Layer 5

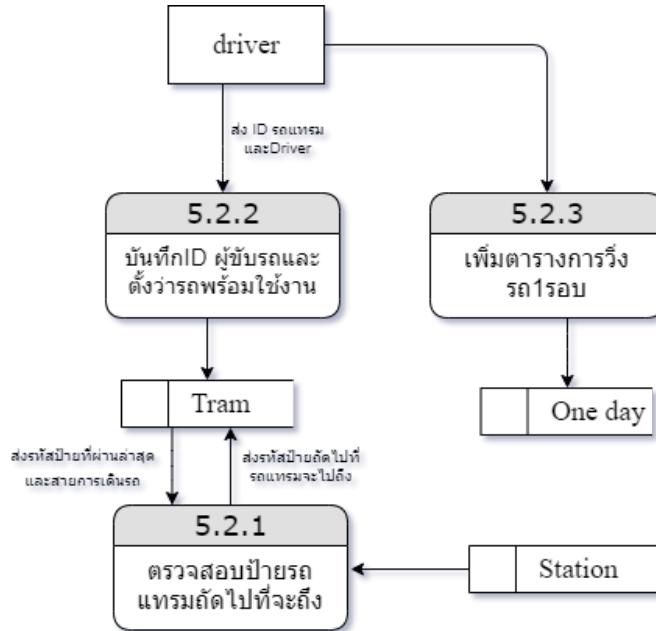
อธิบายการทำงานทั้งหมดในส่วนระบบของ Driver จากใน Layer 0 โดยมีรูปแบบดังรูป



- 5.1. อัพเดตตำแหน่ง GPS โดยจะส่งตำแหน่ง GPS ของ Driver ที่รถแทรมที่ขึ้นไปเก็บไว้ใน position ของ Tram
- 5.2. เริ่มขับรถ 1 รอบ จะเป็นกระบวนการทำงานเมื่อรถแทรมเริ่มวิ่ง 1 รอบ
- 5.3. จบการขับรถ 1 รอบ จะเป็นกระบวนการทำงานเมื่อรถแทรมจบการวิ่ง 1 รอบ
- 5.4. บันทึกผลการดำเนินงาน เป็นระบบฝั่ง Service Provider ในส่วนของ Driver ที่จะเก็บการใช้งานทั้งหมดของ Driver และนำไปเก็บใน Log service
- 5.5. เรียกคุณข้อมูลส่วนตัว โดยจะส่ง ID ของ Driver เพื่อเรียกคุณข้อมูลส่วนตัวของ Driver เอง

## Layer 5.2

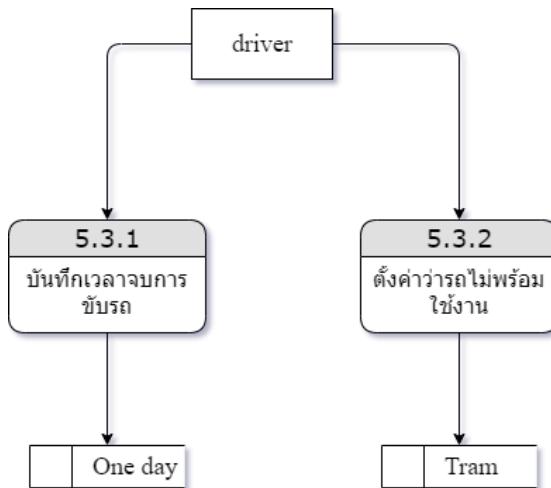
อธิบายการทำงานส่วนของ Driver ในส่วนกระบวนการเมื่อเริ่มขับรถ 1 รอบ จากใน Layer 5 โดยมีรูปแบบดังรูป



- 5.2.1. ตรวจสอบป้ายรถแทรมถัดไปที่จะไปถึง โดยเมื่อรถแทรมผ่านป้ายไหน Tram จะทำการส่งป้ายล่าสุดและสายการเดินรถที่เลือกไปให้ Station เพื่ออัพเดทป้ายถัดไปที่จะถึง
- 5.2.2. บันทึก ID ผู้ขับรถและตั้งค่าว่ารถพร้อมใช้งาน เมื่อรถเริ่มวิ่งก็จะทำการไปอัพเดทว่าใครขับรถคันนี้ และให้ Status เป็น 1 ที่แสดงถึงว่ารถคันนี้พร้อมใช้งานและกำลังวิ่งอยู่ในสาย
- 5.2.3. เพิ่มตารางการวิ่งรถ 1 รอบ เป็นการบันทึกข้อมูลการวิ่งใน 1 รอบ ในวันนั้นๆอัตโนมัติโดย ส่งเวลาเริ่มวิ่ง รหัสรถแทรม รหัส Driver และสายที่วิ่ง โดยจะจัดเก็บไว้ใน One day

## Layer 5.3

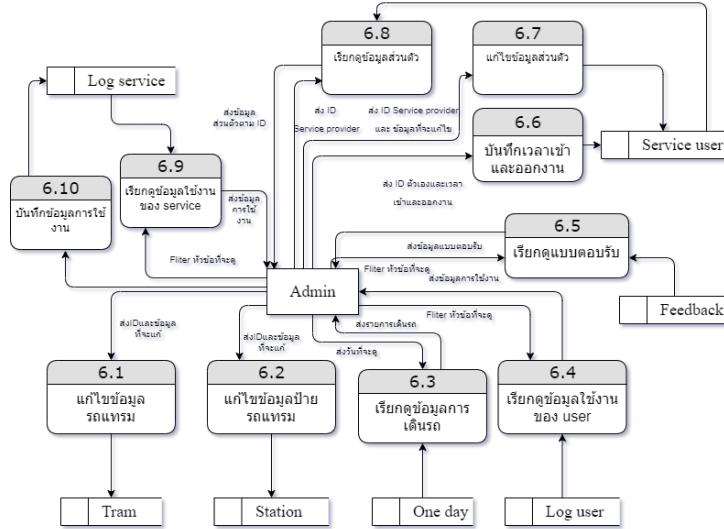
อธิบายการทำงานส่วนของ Driver ในส่วนกระบวนการเมื่อjobการขับรถ 1 รอบ จากใน Layer 5 โดยมีรูปแบบดังรูป



- 
- 5.3.1. บันทึกเวลาจับการขับรถ เป็นการบันทึกเวลาวิ่งรอบเสร็จใน 1 รอบ โดยส่งเวลาจบไปเก็บใน One day ส่วนของเวลาจบ
  - 5.3.2. ตั้งค่าว่ารถไม่พร้อมใช้งาน เป็นการอัพเดท Status ให้เป็น 0 แสดงว่ารถไม่พร้อมใช้งาน

## Layer 6

อธิบายการทำงานทั้งหมดในส่วนระบบของ Admin จากใน Layer 0 โดยมีรูปแบบดังรูป



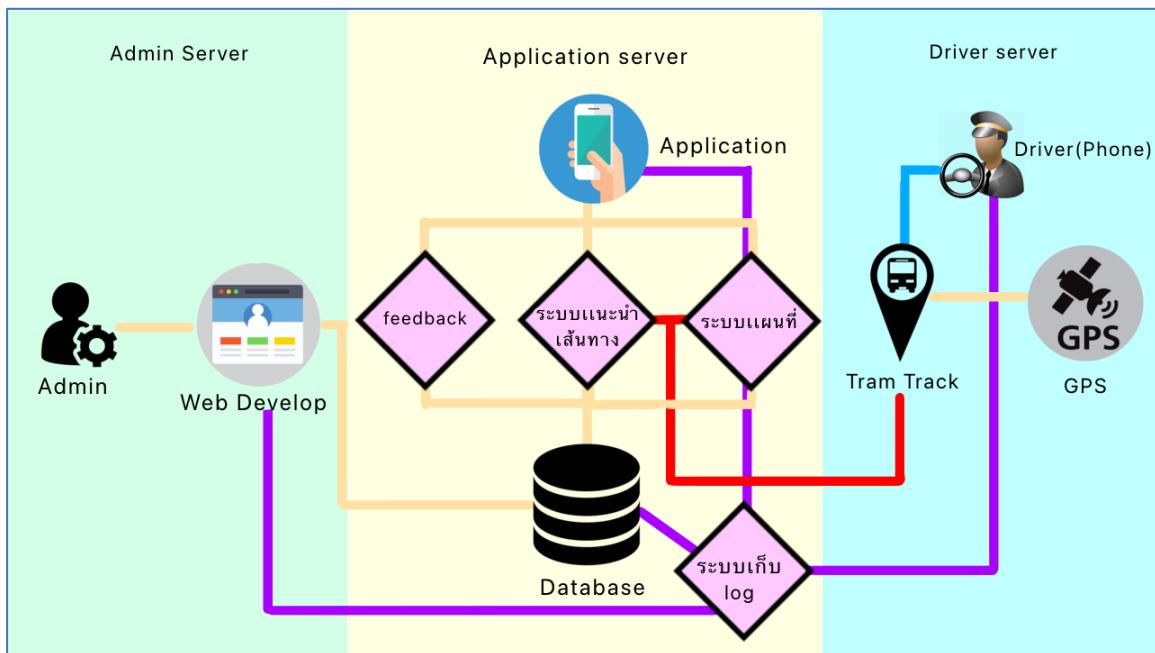
- 6.1. แก้ไขข้อมูลรถแทรม Admin จะสามารถแก้ไขข้อมูลส่วนต่างๆ ของ Tram ส่วนไหนได้เข้าถึงโดย tram\_id
- 6.2. แก้ไขข้อมูลป้ายรถแทรม Admin จะสามารถแก้ไขข้อมูลส่วนต่างๆ ของ Station ส่วนไหนก็ได้โดย station\_id
- 6.3. เรียกดูข้อมูลการเดินรถ Admin จะส่งว่าจะดูข้อมูลการเดินรถวันที่เท่าไรแล้ว One day ก็จะส่งข้อมูลการเดินรถพร้อมรายละเอียดในวันนั้นๆ มา
- 6.4. เรียกดูข้อมูลใช้งานของ User Admin จะสามารถดูประวัติการใช้แอปของ User ได้ โดยมี filter คือ log\_service\_id, service\_user\_id, job\_position, service\_action.type
- 6.5. เรียกดูแบบตอบรับ Admin จะสามารถเข้าไปดูแบบสอบถามการใช้งานแอปที่ได้จาก User ได้โดยมี filter คือ type ของ feedback
- 6.6. บันทึกเวลาเข้าออกงาน โดยจะให้ Admin ทำการแก้ Status การเข้า-ออกงานโดยการส่ง ID ของ Admin และเวลาเข้า-ออกงาน
- 6.7. แก้ไขข้อมูลส่วนตัว โดยจะส่ง ID ของ Service provider และข้อมูลที่แก้แล้วนำไปแก้ข้อมูลใน Service user โดยจะแก่ที่ Service Provider คนไหนก็ได้
- 6.8. เรียกดูข้อมูลส่วนตัว โดยจะส่ง ID ของ Service provider เพื่อเรียกดูข้อมูลส่วนตัวของ Service provider ที่ค้นหา
- 6.9. เรียกดูข้อมูลใช้งานของ service โดยจะดูประวัติการแก้ไขข้อมูลของผู้ใช้งาน Service provider โดยจะมี filter คือ log\_user\_id, user\_id, user\_action, nearest\_station\_id, log\_time
- 6.10. บันทึกข้อมูลการใช้งาน เป็นระบบผู้ใช้งาน Service Provider ในส่วนของ Admin ที่จะเก็บการใช้งานทั้งหมดของ Admin และนำไปเก็บใน Log service

## Software Architecture

ในส่วนของ Software Architecture จะมีการใช้ architecture แบบ monolithic architecture กล่าวคือ component ต่าง ๆ ของแต่ละระบบที่มาจากการ environment เดียวกัน จะมีการใช้ database ร่วมกัน ซึ่งภายใน Web Server จะประกอบไปด้วย 4 Sub-Systems ได้แก่

1. ระบบ feedback
2. ระบบแนะนำเส้นทาง
3. ระบบแผนที่
4. ระบบเก็บ log

โดยจะเห็นได้ว่า Admin จะแก้ไขข้อมูลจาก web develop และส่งข้อมูลไปเก็บใน Database ส่วนรถแทรมจะดึงข้อมูลจาก GPS และส่งข้อมูลเข้าไปใน มือถือของ Driver และ user หรือผู้ใช้ปกติจะใช้จาก Application ซึ่งในตัว App จะดึงข้อมูลจากระบบย่อยต่างๆซึ่งระบบย่อยต่างๆก็จะดึงข้อมูลมาจาก Database อีกที และสุดท้ายข้อมูลการกระทำทั้งหมดของผู้ใช้ต่างๆทั้ง Admin, Driver และ User จะถูกระบบเก็บ log จะดึงเก็บข้อมูลและส่งเข้า Database



## Front-End (UX/UI Design)

ในการออกแบบ User Interface จะเน้นการออกแบบให้แทบควบคุมและปุ่มต่างๆ อยู่บริเวณด้านล่างของแอพพลิเคชัน เพราะว่าเพื่อให้สะดวกต่อการใช้ด้วยมือเดียว และไม่ต้องเอื้อมนิ้วขึ้นไปบริเวณด้านบนของหน้าจอ และเลือกใช้โทนสีหลักคือ ดำ/เหลือง เพื่อให้โทนสีทั้งแอพพลิเคชันไม่สว่างจนเกินไป

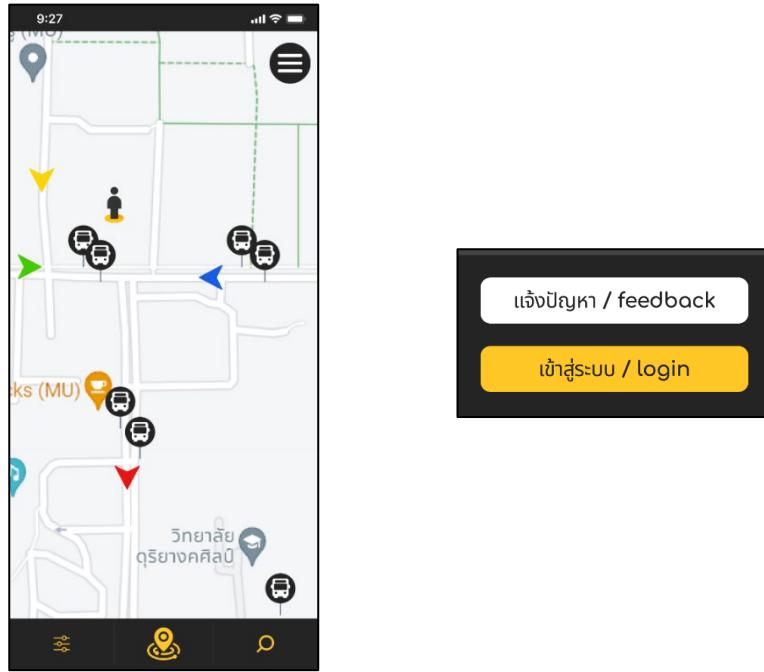
ในหมวดการใช้งานจะแบ่งเป็นส่วนของ User และ Driver โดยการจะใช้หมวด Driver จะต้องเข้าสู่ระบบโดยใช้ driverID และรหัสผ่าน ที่ได้รับจาก Admin ของระบบโดยตรงเท่านั้น ไม่สามารถสมัครสมาชิกได้

### ส่วนของ User



หน้าแรกของแอพพลิเคชันที่แสดงโลโก้ของแอพพลิเคชันส่วนของ User โดยที่ไว้แล้ว ผู้ใช้จะสามารถใช้ฟังก์ชันทั้งหมดได้โดยไม่ต้องล็อกอินเข้าสู่ระบบ

## 1. หน้าหลัก[User]



จะแสดงรถแทรมและป้ายทั้งหมดที่บริเวณนั้น

จะมีฟังก์ชัน 3 อย่างบริเวณแทบด้านล่าง

- สายการเดินรถ(Fliter) : เลือกแสดงเฉพาะสายที่กำหนด
- Nearby : แสดงรถแทรมคันถัดไป(Next Tram)จากป้ายที่ใกล้ที่สุด
- ค้นหาเส้นทาง(Search) : ค้นหาเส้นทาง

จะมีฟังก์ชัน 2 อย่างที่แน่นที่

- กดที่รถแทรม : แสดงป้ายถัดไป(Next Stop)ที่รถแทรมจะผ่าน
- กดที่ป้ายแทรม : แสดงรถแทรมคันถัดไป(Next Tram)ที่จะมาถึง

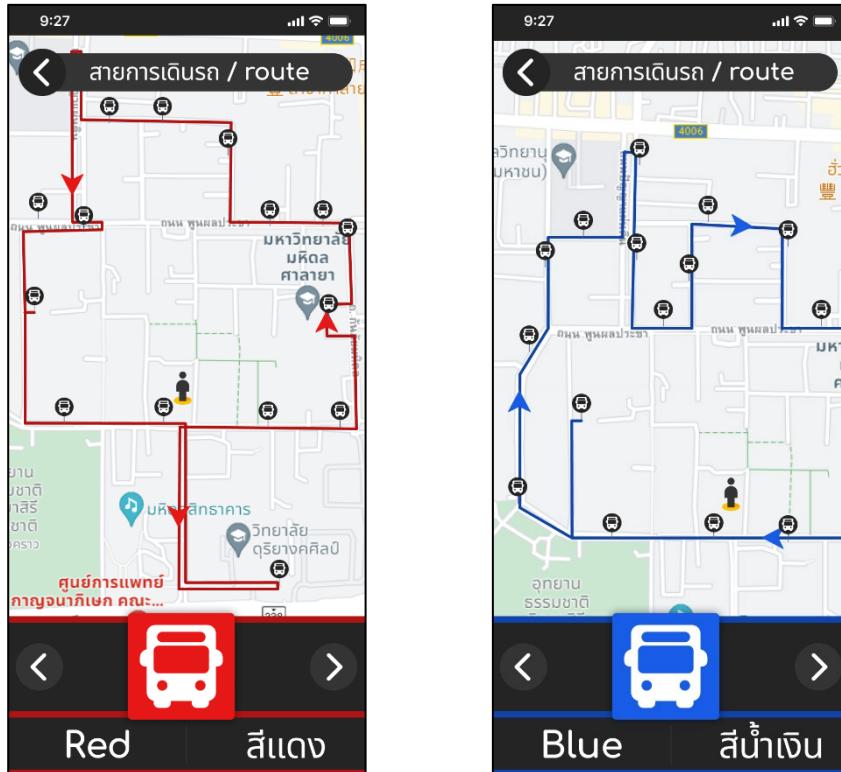
และมี Overlay แบบเมนูเพิ่มเติมเมื่อกดปุ่มด้านขวาบน

- แจ้งปัญหา(Feedback) : แจ้งปัญหาการและคำติชม
- เข้าสู่ระบบ(Login) : เข้าสู่ระบบเพื่อใช้งาน Tramer[Driver]

Service/API ที่ใช้

- Service ตรวจสอบตำแหน่งรถแทรม
- Service แสดงป้ายรถแทรมที่ใกล้ที่สุดจากตำแหน่งผู้ใช้

## 2. สายการเดินรถ(Fliter)



แสดงเส้นทางการเดินรถทั้งหมดของสายสีต่างๆ

จะมีฟังก์ชัน 2 อย่างบริเวณแท็บด้านล่าง

- ลูกศรซ้าย : เลื่อนไปยังสายก่อนหน้า
- ลูกศรขวา : เลื่อนไปยังสายถัดไป

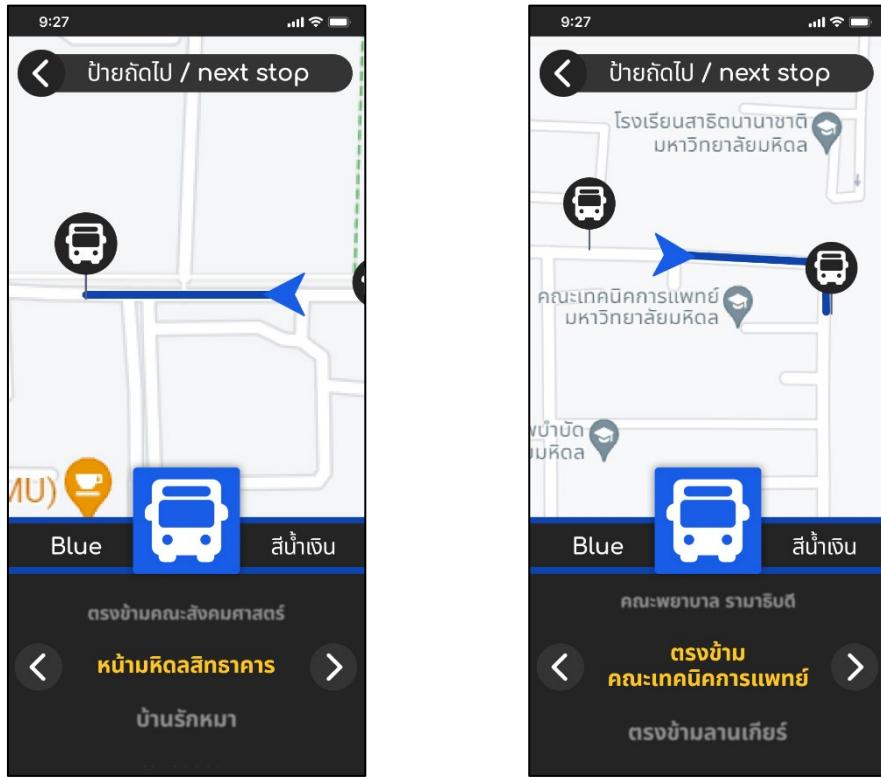
และมีฟังก์ชัน 1 อย่างที่บริเวณแผนที่

- กดที่รุ่นแทรม : แสดงป้ายถัดไป(Next Stop)ที่รุ่นแทรมจะผ่าน

Service/API ที่ใช้

- Service ตรวจสอบตำแหน่งรถแทรม
- Service ตรวจสอบเส้นทางการเดินรถแทรมพร้อมแสดงรถแทรมในสาย

### 3. ป้ายถัดไป(Next Stop)



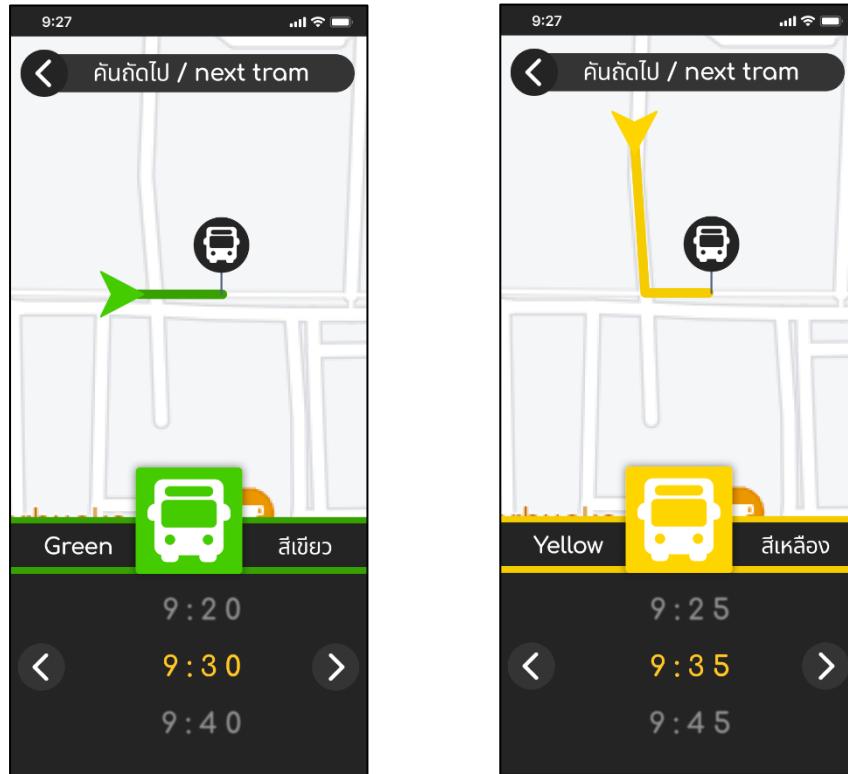
แสดงป้ายต่างๆที่รถแทรมคันนั้นจะผ่าน  
จะมีฟังก์ชัน 2 อย่างบริเวณแท็บด้านล่าง

- ลูกศรซ้าย : เลื่อนไปยังคันก่อนหน้า
- ลูกศรขวา : เลื่อนไปยังคันถัดไป

Service/API ที่ใช้

- Service ตรวจสอบตำแหน่งรถแทรม
- Service ตรวจสอบเวลาที่รถแทรมจะมีถึงป้ายสถานีที่รอ

#### 4. คันถัดไป(Next Tram)



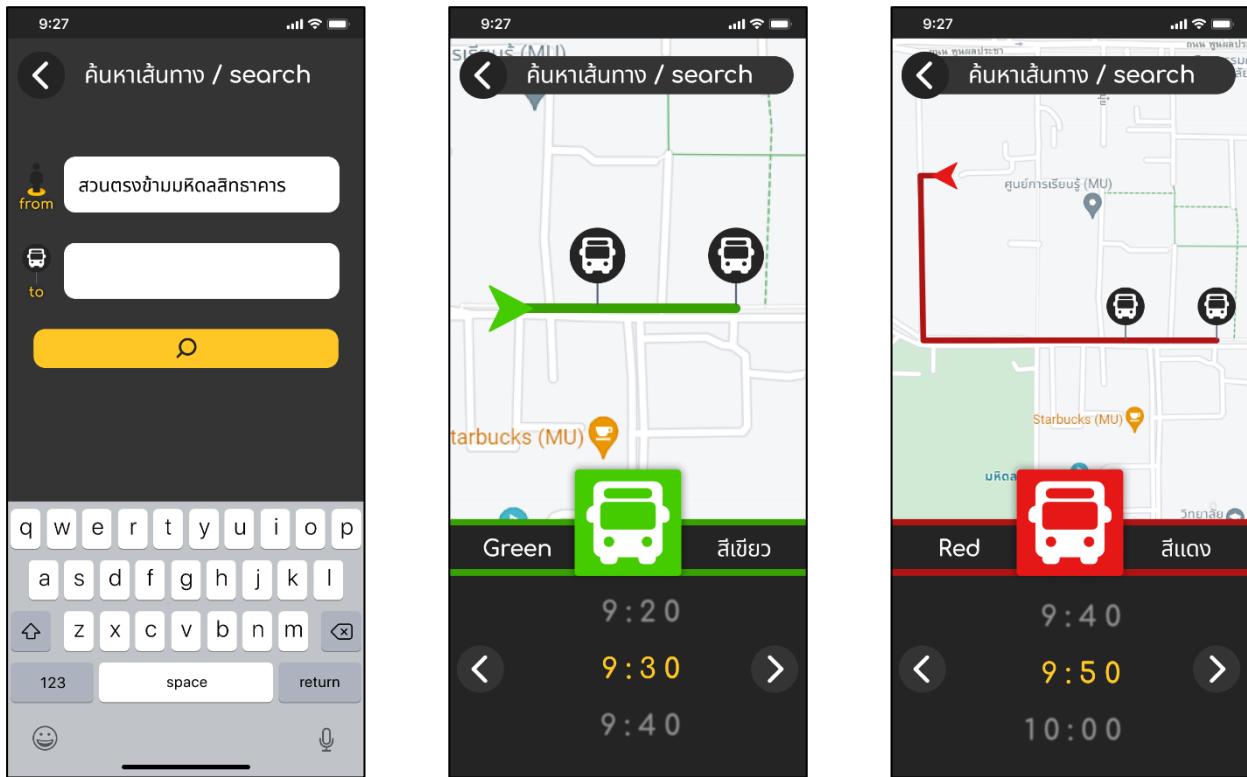
แสดงเวลาที่รถแทรมสายต่างๆที่ผ่านป้ายนั้นๆจะมาถึง  
จะมีฟังก์ชัน 2 อย่างบริเวณแท็บด้านล่าง

- ลูกศรซ้าย : เลื่อนไปยังสายก่อนหน้า
- ลูกศรขวา : เลื่อนไปยังสายถัดไป

Service/API ที่ใช้

- Service ตรวจสอบตำแหน่งรถแทรม
- Service ตรวจสอบเวลาที่รถแทรมจะถึงป้ายรถแทรมที่เลือก
- Service ตรวจสอบตารางเวลารถแทรม

## 5. ค้นหาเส้นทาง(Search)



ค้นหาเส้นทางการเดินทางไปที่หมาย โดยเมื่อกดเข้ามาจะกรอกต้นทางเป็นสถานที่ใกล้ที่สุดโดยอัตโนมัติ

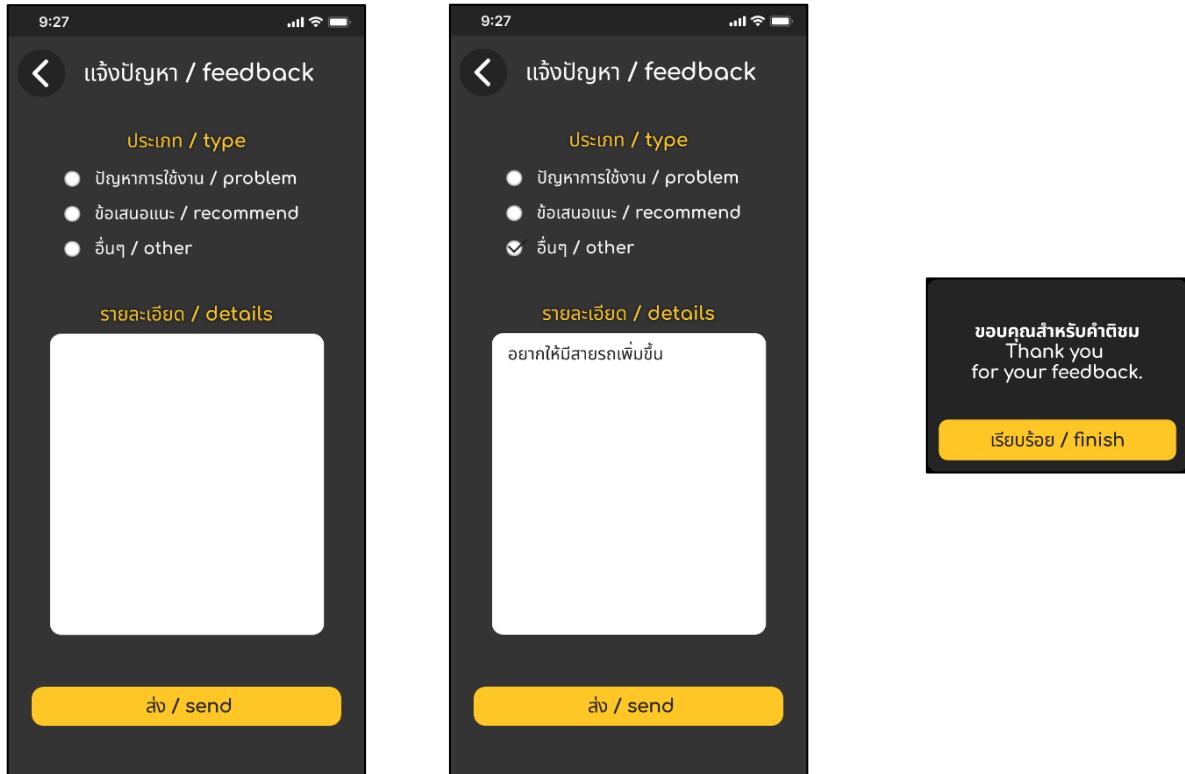
จะมีฟังก์ชัน 2 อย่างบริเวณแท็บด้านล่าง

- ลูกศรซ้าย : เลื่อนไปยังสายก่อนหน้า
- ลูกศรขวา : เลื่อนไปยังสายถัดไป

Service/API ที่ใช้

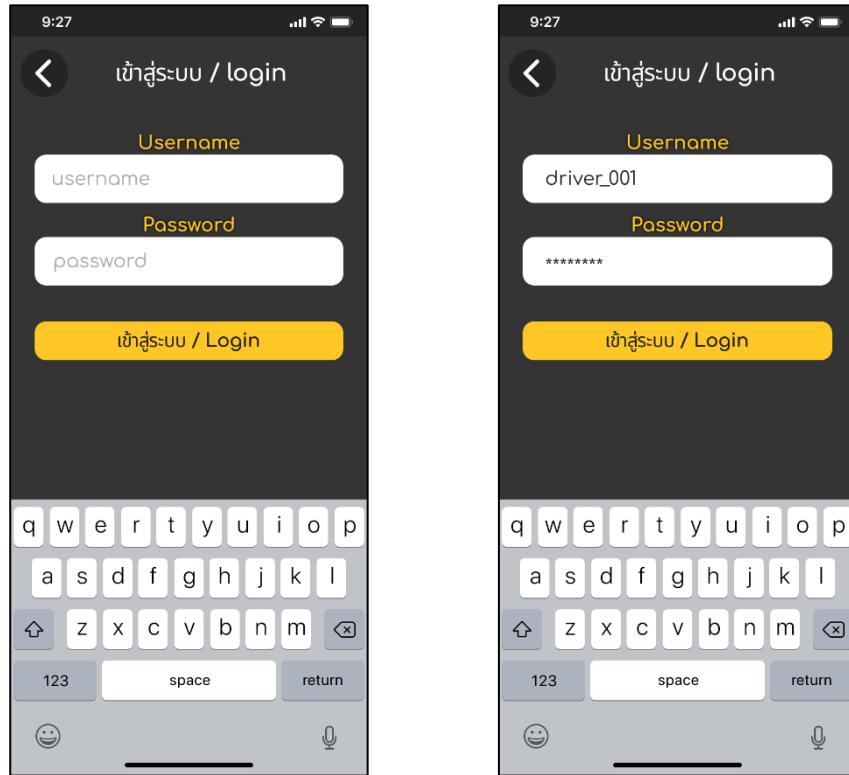
- Service ตรวจสอบตำแหน่งรถแทรม
- Service ค้นหาวิธีการเดินทางจากจุดเริ่มต้นไปปลายทาง

## 6. แจ้งปัญหา(Feedback)



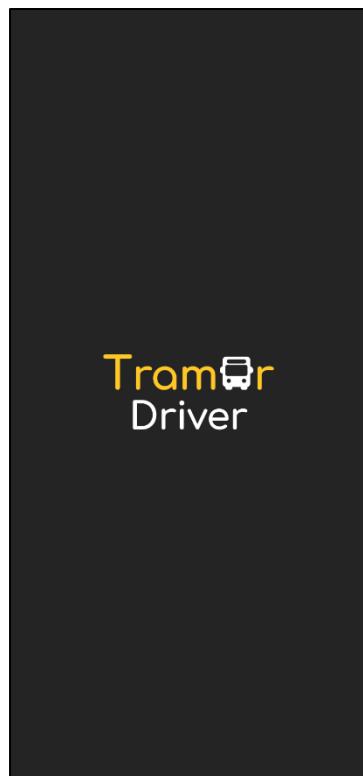
แจ้งปัญหาหรือให้คำแนะนำติชมการใช้งานแอพพลิเคชัน โดยจะมีให้ติํกเลือกประเภท และมีช่องให้กรอกรายละเอียด เมื่อกรอกเสร็จและกดส่งแล้ว จะมีหน้าต่างแสดงคำขอบคุณและล้างข้อมูลทั้งหมดเพื่อกรอกใหม่

## 7. เข้าสู่ระบบ(Login)



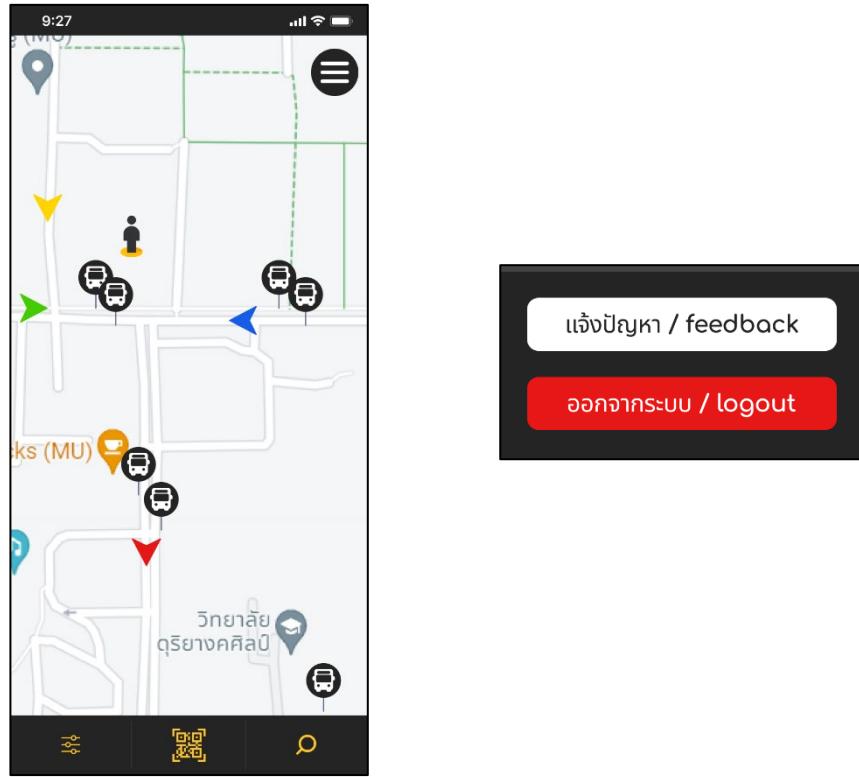
เข้าสู่ระบบเพื่อใช้งานระบบ Tramer[Driver] โดย username และ password จะได้รับจาก Admin ระบบเท่านั้น ไม่สามารถสมัคร username และกำหนด password เองได้

## ส่วนของ Driver



หน้าแรกของแอปพลิเคชันที่แสดงโลโก้ของแอปพลิเคชันส่วนของ Driver โดยพังก์ชันทั้งหมดของแอปพลิเคชันใช้ได้เหมือนกับผู้ใช้ปกติทั้งหมด ยกเว้นแค่ฟังก์ชัน Nearby ที่ถูกแทนที่ด้วยฟังก์ชัน QR Code ที่ใช้สแกน QR Code ที่รถแทรมเพื่อเข้าใจงานฟังก์ชันข้อมูลรถแทรมและฟังก์ชันบันทึกประวัติการขึ้บรถ

## 1. หน้าหลัก[Driver]



จะมีฟังก์ชันการทำงานเหมือนแบบ User ทุกอย่างยกเว้น

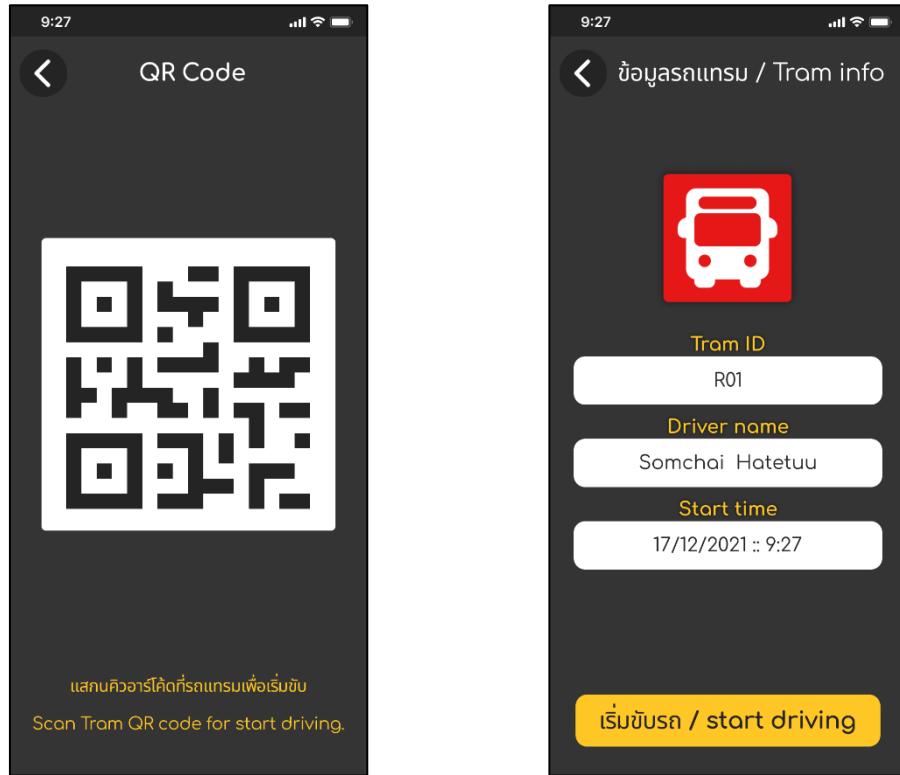
ฟังก์ชัน 1 อย่างบริเวณแท็บด้านล่างที่เปลี่ยนจาก Nearby เป็น

- QR Code Scanner : เปิดกล้องเพื่อสแกน QR Code เพื่อดูข้อมูลของรถแทรมคันนั้นๆ และมี Overlay แสดงเมนูเพิ่มเติมที่เปลี่ยนไปคือ
- ออกจากระบบ(Logout) : ออกจากระบบเพื่อกลับไปใช้งาน Tramer[User]

Service/API ที่ใช้

- Service เข้าทำงานและออกงาน

## 2. QR Code Scanner

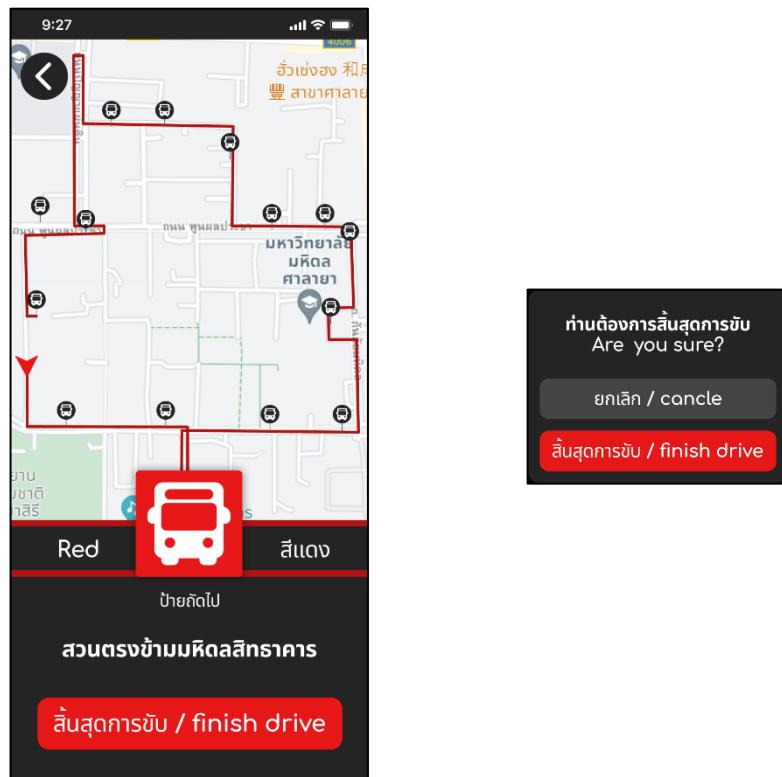


เมื่อสแกน QR Code ที่รถแล้ว จะแสดงหน้า ข้อมูลรถแทรม(Tram info) ขึ้นมาเพื่อให้ตรวจสอบ ข้อมูลขี่อผู้ขับ และเวลาเริ่มขับรถแทรม เมื่อกดปุ่มเริ่มขับรถ ระบบจะบันทึกเวลาเริ่มขับไปที่ฐานข้อมูล ก่อนที่จะไปที่หน้า ขับรถ(Driving) ต่อไป

Service/API ที่ใช้

- Service เริ่มการขับรถ 1 รอบ

### 3. ขับรถ(Driving)



หน้าขับรถจะแสดงเส้นทางทั้งหมดที่รถต้องขับไป และแสดงป้ายต่อๆไปที่จะต้องจอด เมื่อผู้ขับต้องการจะสิ้นสุดการขับ ก็ให้กดที่ปุ่มสิ้นสุดการขับ ระบบจะให้กดยืนยันก่อนอีกครั้ง เมื่อกดยืนยัน ระบบจะบันทึกเวลาสิ้นสุดการขับเข้าไปที่ฐานข้อมูล และกลับไปที่หน้าหลัก

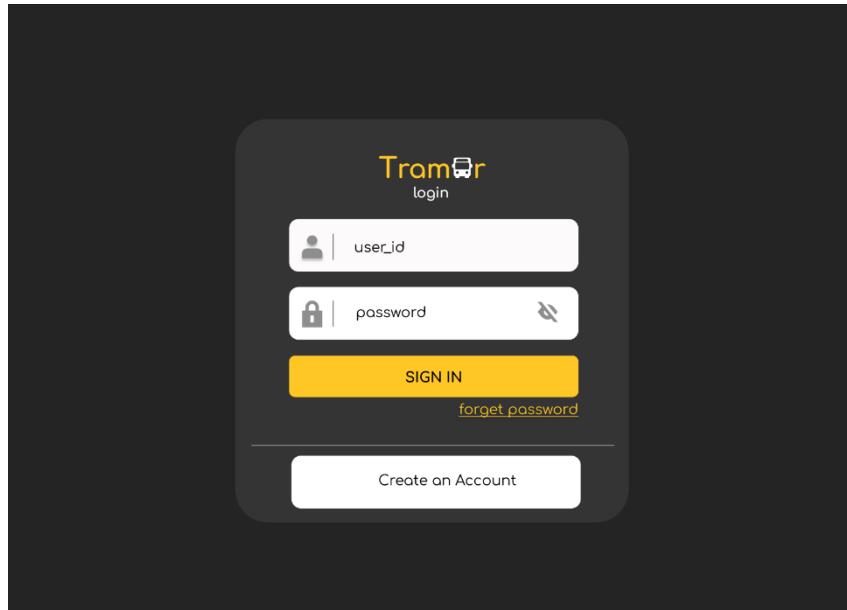
#### Service/API ที่ใช้

- Service เริ่มการขับรถ 1 รอบ
- Service จบการขับรถ 1 รอบ

## ส่วนของ Admin

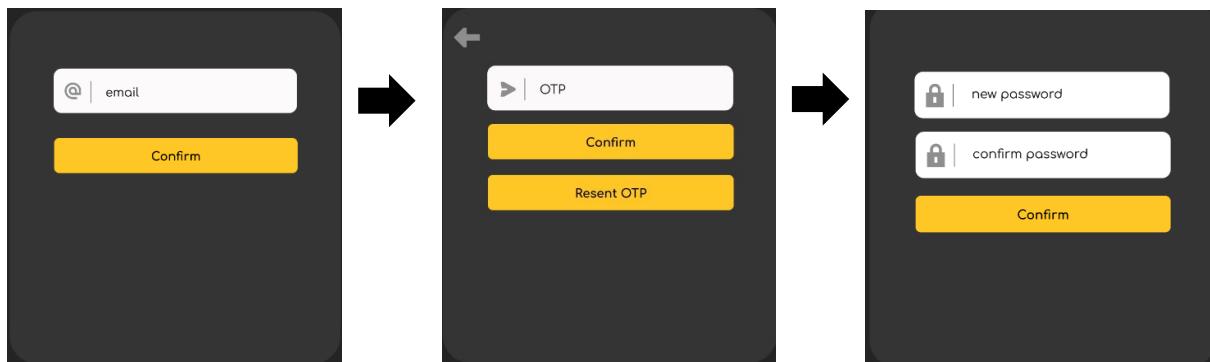
ในส่วน Admin จะต้องเข้าใช้งานผ่านเว็บไซต์โดยใช้ Username และ Password ที่กำหนด เพื่อใช้ระบบหลังบ้านที่ค่อยควบคุมการทำงานของแอพพลิเคชัน เช่นการเปิดปิดใช้งานรถ เป็นต้น

### 1. หน้า Sign in

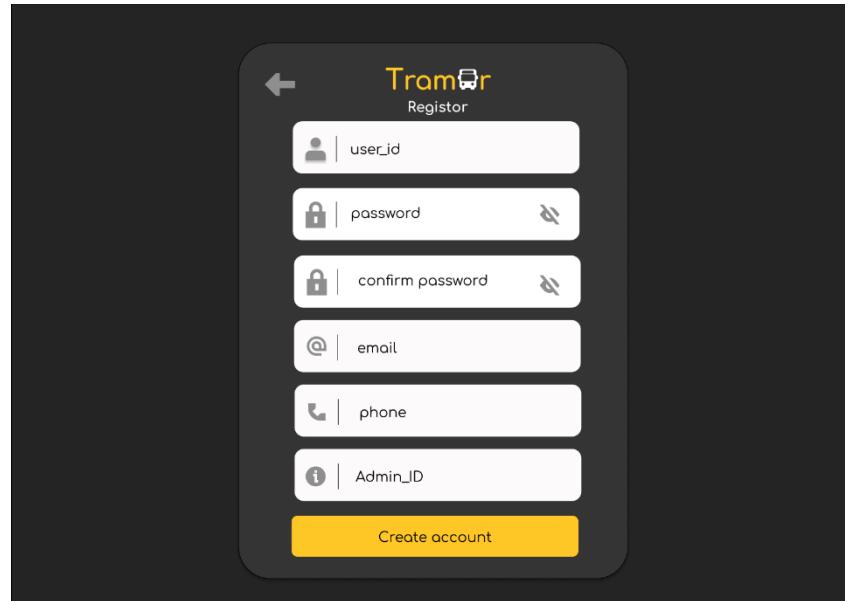


ส่วนแรกที่จะได้พบเมื่อเข้าเว็บไซต์มาคือหน้า sign in โดยในส่วนนี้จะแบ่งระบบออกเป็น 3 ส่วน ใหญ่ๆได้แก่ ส่วน sign in ปกติดังภาพ

ส่วนต่อมาจะเป็นส่วนของการลืม password โดยจะให้ทำการยืนยันจาก e-mail และหลังจากยืนยัน

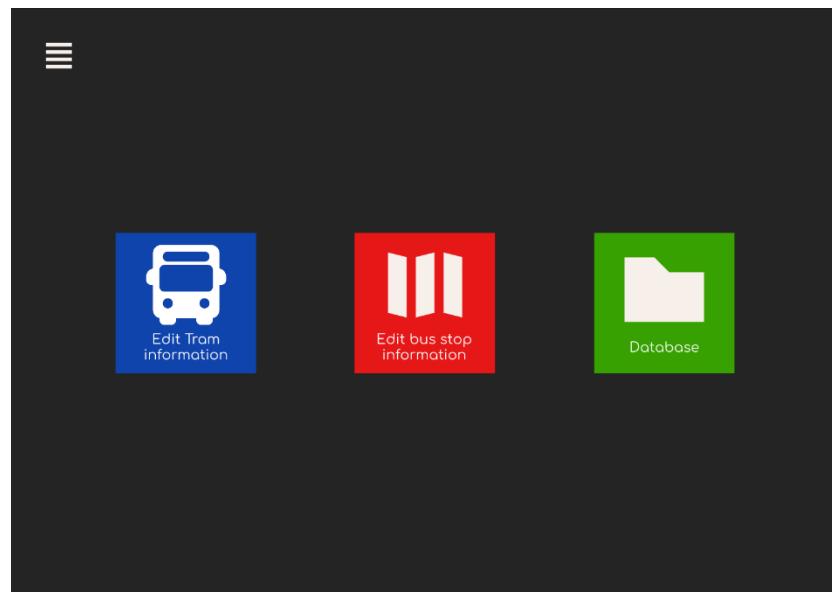


เสร็จจะให้เปลี่ยนรหัสผ่านแล้วจะให้กลับไปหน้า sign in อีกครั้ง



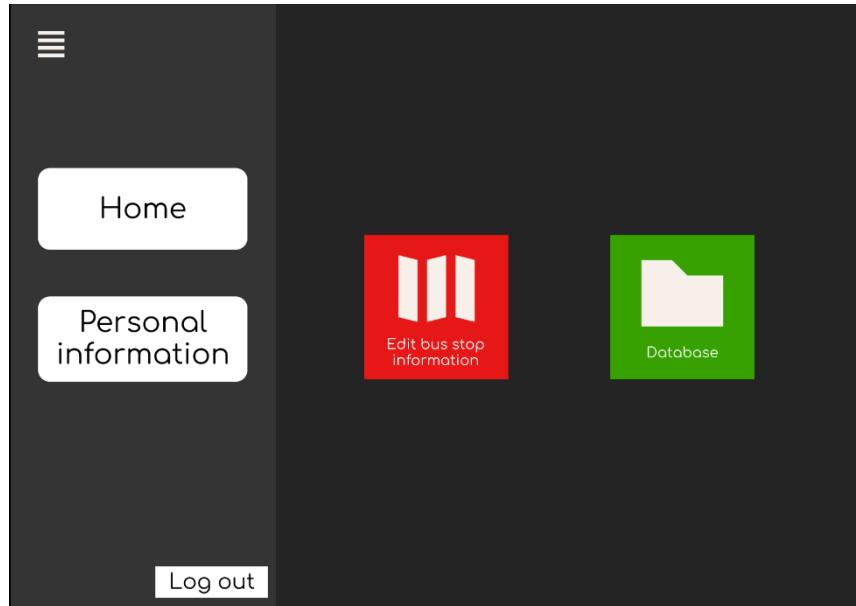
ส่วนสุดท้ายจะเป็นหน้าลงทะเบียนบัญชีใหม่โดยผู้ลงทะเบียนจำเป็นจะต้องใส่ Admin\_ID เพื่อเป็นการยืนยันว่าไม่ใช่ผู้ใช้งานทั่วไป

## 2. หน้า Home Page



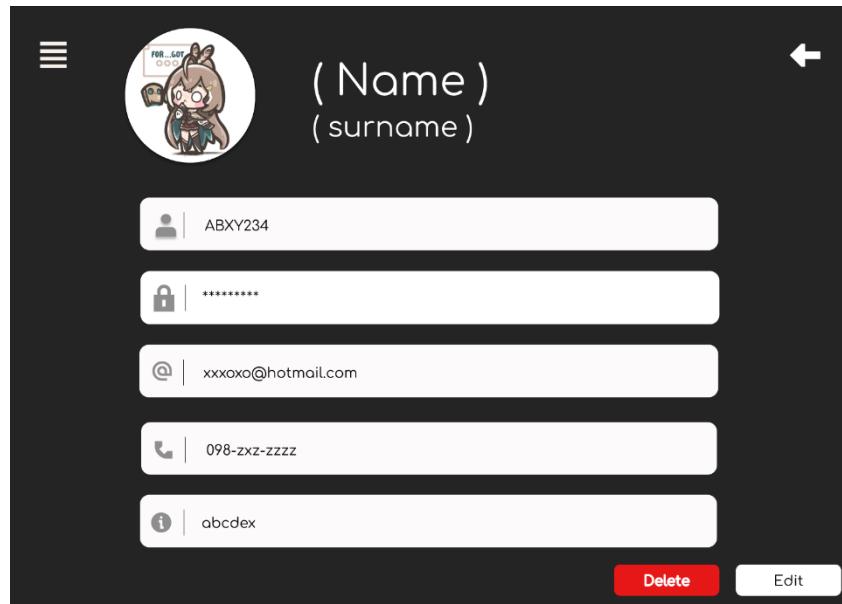
ในส่วนของ Home Page จะแบ่งออกเป็น 3 ส่วนได้แก่ ส่วนแก้ไขข้อมูลรถแทรม (edit tram information) ส่วนแก้ไขข้อมูลป้ายรถแทรม (edit bus stop information) และส่วนของการเก็บข้อมูลทั้งหมด(Database)

### 3. ແກບ Menu

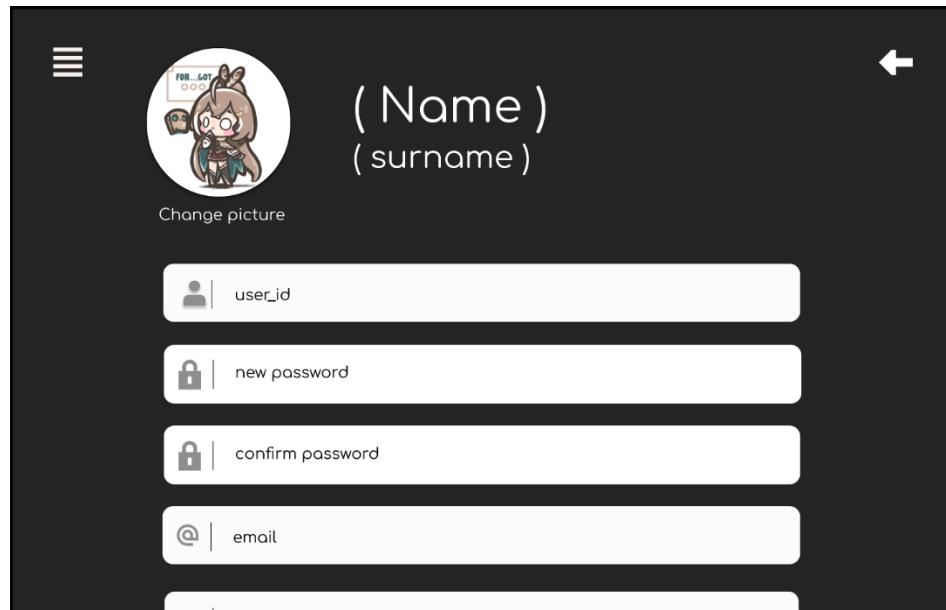


ເນື່ອກົດແກບເມຸນທາງດ້ານບ່ນໜ້າຍຮະບບຈະຫຼື້ນ Overlay menu ຫີ້ນາ ທີ່ຈະແສດງເມຸນອອກນາ 3 ສ່ວນ  
ໄດ້ແກ່ ປຸ່ມ Home ( ພາກເລືອກປຸ່ມນີ້ຈະພາກລັບມາໜ້າ Home Page ) ປຸ່ມ Personal information ແລະ ປຸ່ມ  
log out ທີ່ຈະພາກລັບໄປທີ່ໜ້າ Sign in

### 4. ໜ້າ Personal information



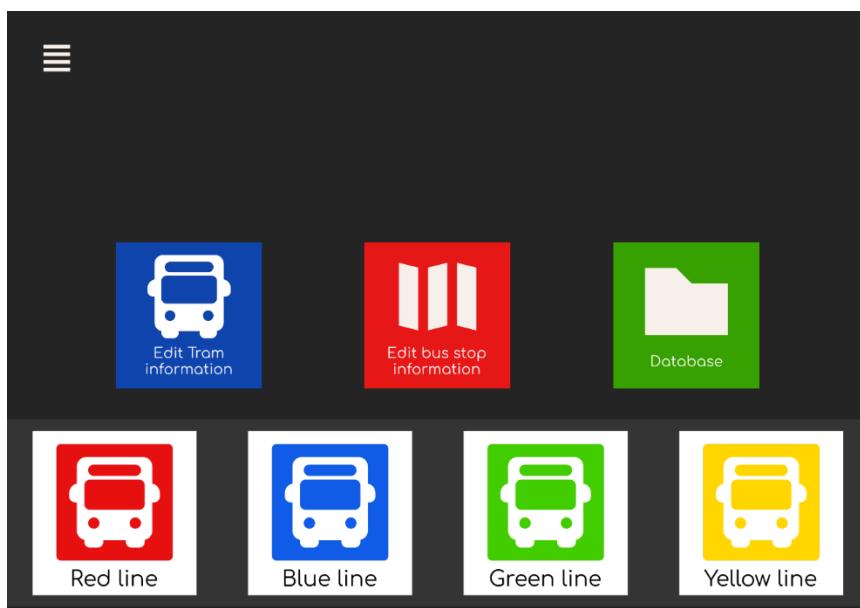
ໜ້ານີ້ຈະເປັນໜ້າແສດງຂໍ້ອມູນຂອງບໍລິສັດໃຫ້ຈະແສດງຊື່, ນາມສກຸລ, user\_ID, email, ເບອຣີໂທຮັກພົກ  
ແລະ Admin\_ID ໂດຍຈະສາມາດແກ້ໄຂຂໍ້ອມູນທີ່ຈະແສດງຂໍ້ອມູນຂຶ້ນໄດ້



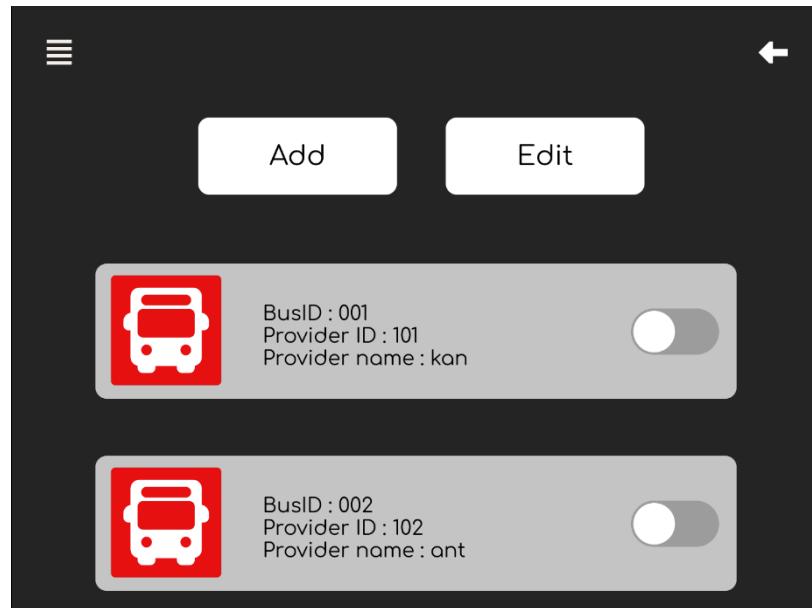
ช่องเปลี่ยน password และเปลี่ยนภาพโปรไฟล์ จะขึ้นมาให้ใส่เฉพาะหน้าแก้ไขข้อมูล Service/API ที่ใช้

- Service แก้ไขข้อมูลส่วนตัว

##### 5. หน้า Edit Tram information



เมื่อเรากดปุ่ม Edit Tram information ระบบจะแสดง overlay ขึ้นมาให้เราเลือกว่าจะดูข้อมูลของ รถแทรมสีอะไร



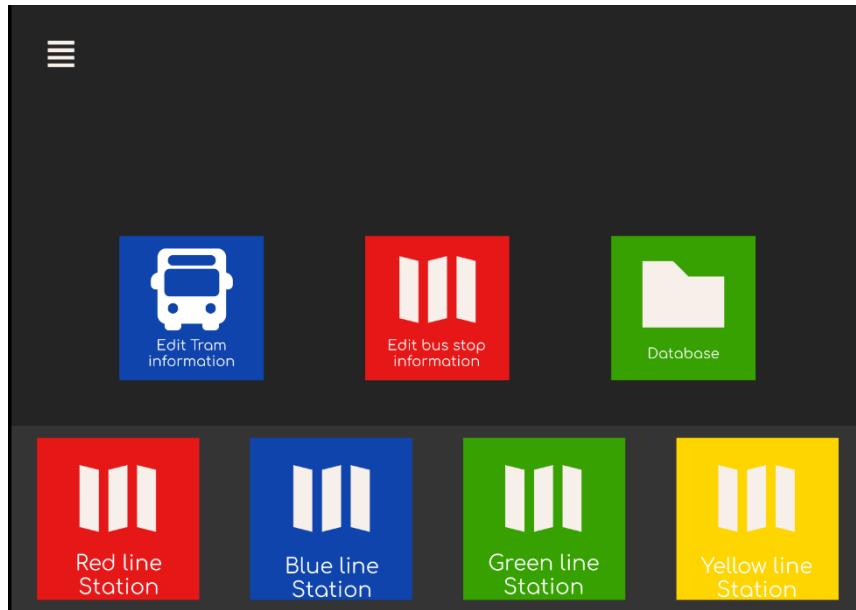
เมื่อเลือกสีของรถแทรมได้ระบบจะพามาหน้าดูข้อมูลและเราจะสามารถเลือกเปิดใช้งานรถแทรม เพิ่มรถแทรมหรือแก้ไขข้อมูลได้



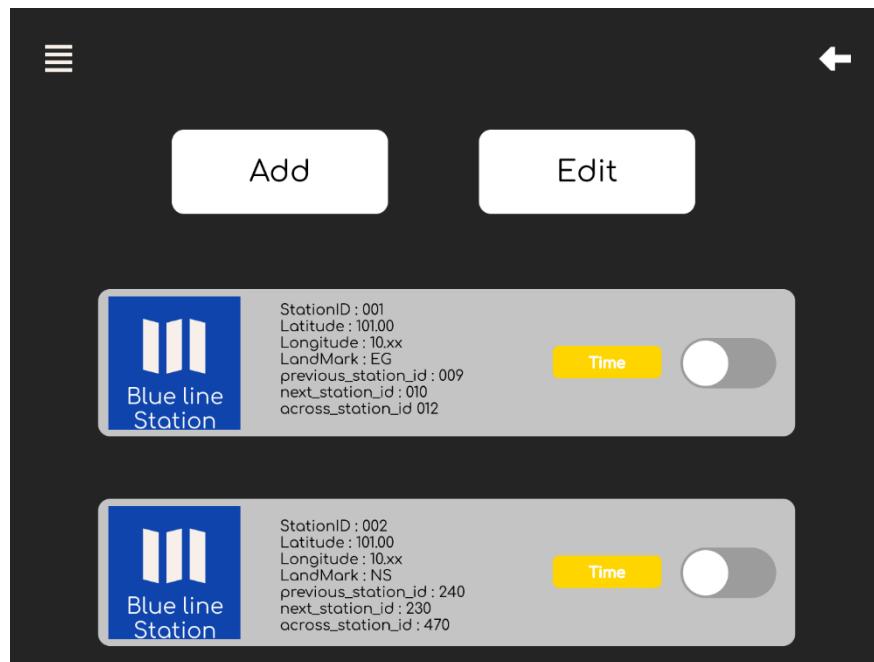
ในหน้า edit จะสามารถย้ายสีของแทรมหรือลบข้อมูลได้ และในการ Add จะต้องกรอกข้อมูลดังภาพ Service/API ที่ใช้

- Service แก้ไขข้อมูลของรถแทรม

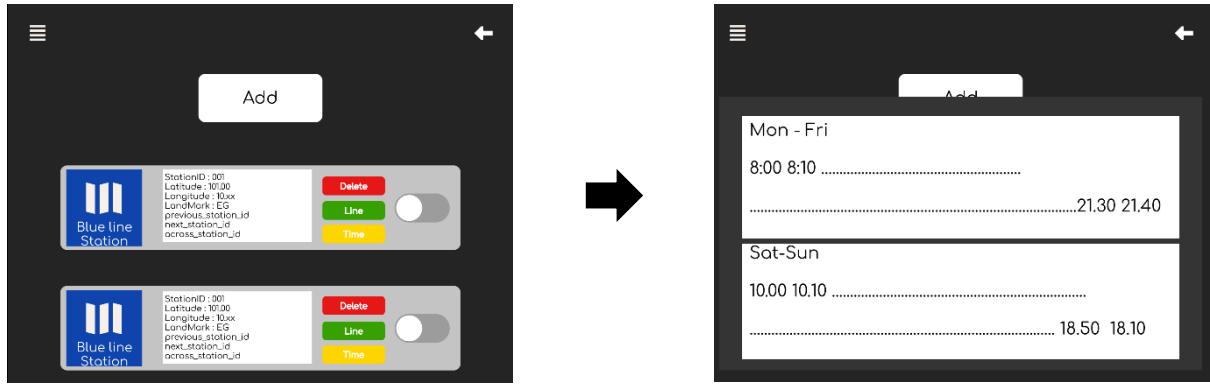
## 6. หน้า Edit bus stop information



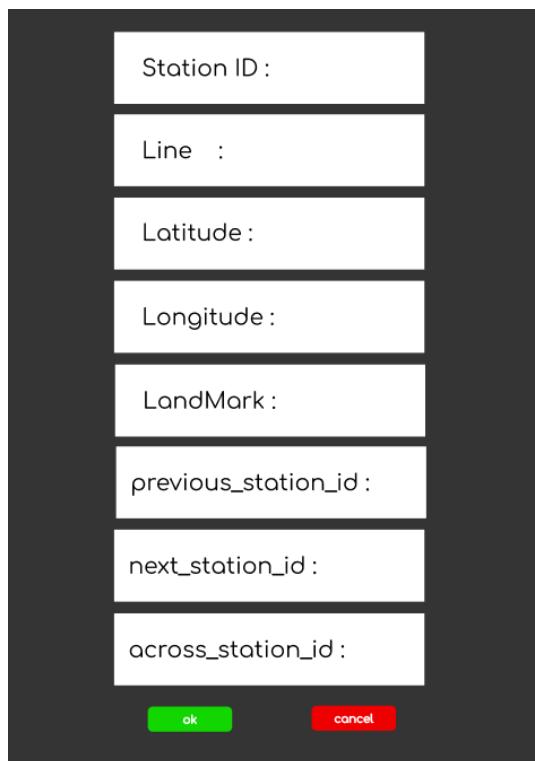
หน้า Edit bus stop information จะคล้ายๆ กับหน้า Edit Tram information คือจะแสดง overlay ขึ้นมาให้เลือกดูสถานีของแต่ละสาย



โดยจะต่างกับ Edit Tram information ตรงที่จะเพิ่มปุ่ม Time ที่จะแสดงตารางเวลาที่รถแทรมจะมาถึงได้และสามารถแก้ไขข้อมูลตารางเวลาในหน้า edit ได้



หน้า edit จะสามารถย้ายสีของป้ายเพิ่มหรือลบข้อมูลได้และยังสามารถแก้ไขตารางเวลาได้อีกด้วย

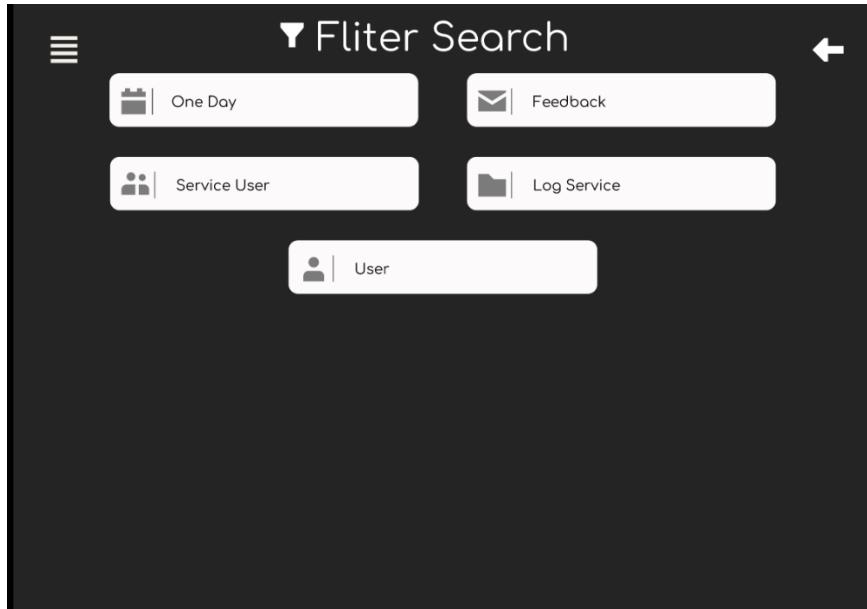


ข้อมูลที่ต้องกรอกเมื่อนำมาเพิ่มป้ายรถแทรมป้ายใหม่

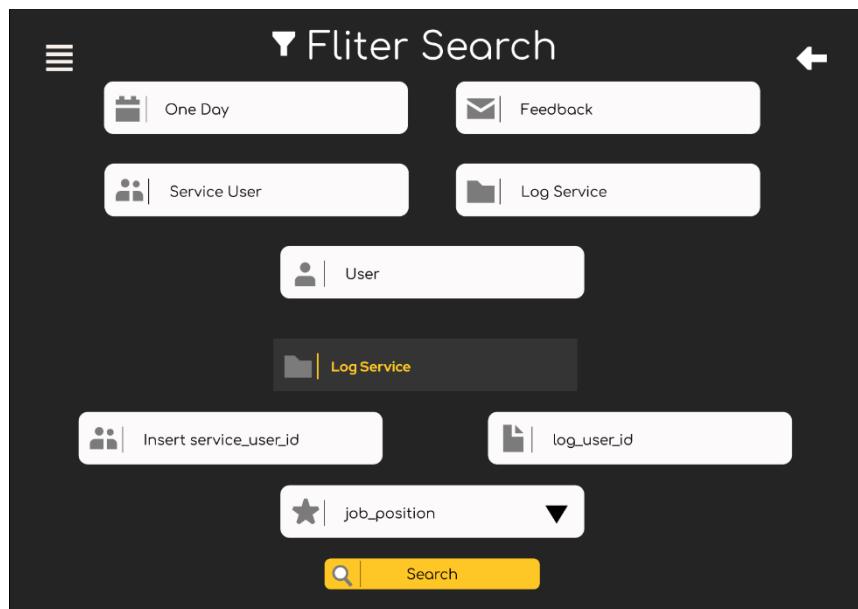
### Service/API ที่ใช้

- Service แก้ไขข้อมูลของป้ายรถแทรม
- Service เรียกดูข้อมูลการเดินรถใน 1 วัน

## 7. หน้า Database



เมื่อเลือกปุ่ม Database ระบบจะพามาหน้า filter ซึ่งผู้ใช้จะต้องเลือกว่าจะดึงข้อมูลเกี่ยวกับเรื่องอะไรแล้วระบบจะขึ้น Overlay มาให้กรอกข้อมูลที่เกี่ยวข้องซึ่งเราไม่จะเป็นต้องกรอกข้อมูลทั้งหมดระบบจะทำการประมวลผลดึงเฉพาะข้อมูลที่เกี่ยวข้องกับข้อมูลที่เรากรอกออกมาให้



ตัวอย่างการเลือก log service ระบบจะเพิ่มข้อมูลที่เกี่ยวข้องมาให้กรอกและดึงเฉพาะข้อมูลที่กรอกออกมามาให้หากไม่กรอกก็จะดึงข้อมูลออกมากทั้งหมด

**Log Service**

log_service_id	service_provider_id	job_position	login_time	logout_time
10000	20000	Admin	10:00	18:00
10001	20001	Driver	8:00	20:00

จะแสดงตารางออกบานาตาม position ที่เลือก ( เลือกดี admin or driver ท่านนั้น )

service\_action.type

- Add
- Edit
- Delete
- Read

**Log Service**

log_service_id	service_provider_id	job_position	login_time	logout_time
10000	20000	Admin	10:00	18:00
10001	20001	Driver	8:00	20:00

จะแสดงตารางออกบานาตาม position ที่เลือก ( เลือกดี admin or driver ท่านนั้น )

service\_action.type.Edit

type	time	database_name	access_id	position_command	previous_info	edit_info	delete_info
Edit	12:00	feedback.json	10000	ABC	XYZ		
Edit	12:30	log_service.json	10000		123	456	

ภายใน log service สามารถเลือก action type ที่เราต้องการออกมาได้จะเห็นว่าภายในการ edit จะเก็บข้อมูลก่อนและหลังเอาไว้

**Service User**

service_provider_id	first_name	last_name	job_position	email	phone_no	status
10000	kan	bbb	admin	pod@....	08xx	available
10001	dfsf	fsfds	driver	podod@...	02xxxx	unavailable

provider\_ID 10000

date	time_in	time_out
15/01/20xx	8:00	20:00
16/01/20xx	5:00	21:00
18/01/ 20xx	6:00	18:00

ในส่วนของ Service user เราสามารถเลือกกดปุ่ม period เพื่อดูตารางเวลาการเข้าออกงานของ Admin\_ID นั้นได้

The image consists of two vertically stacked screenshots of a mobile application interface, likely a dashboard or log viewer.

**Screenshot 1:** The top screenshot shows a table of user logs. The columns are: log\_user\_id, user\_id, nearest\_station\_id, Day, and Time. The data rows are:

log_user_id	user_id	nearest_station_id	Day	Time
10655	50000	10201	Monday	12:00
12332	50002	20020	Monday	16:00

Below the table is a section titled "หากกรอกฟิลเตอร์แล้วนับก็" (If you enter a filter, it will count) containing a dropdown menu labeled "user\_action". The options listed are: tram\_click, station\_click, line\_click, click\_on\_guide, send\_feedback, and search\_way.

**Screenshot 2:** The bottom screenshot shows the same table of user logs. The data rows are identical to Screenshot 1.

Below the table is a section titled "หากกรอกฟิลเตอร์แล้วนับก็" (If you enter a filter, it will count) containing a dropdown menu labeled "user\_action.tram\_click". The options listed are: tram\_click, line, tram\_id, 10000 red, and 20000 blue.

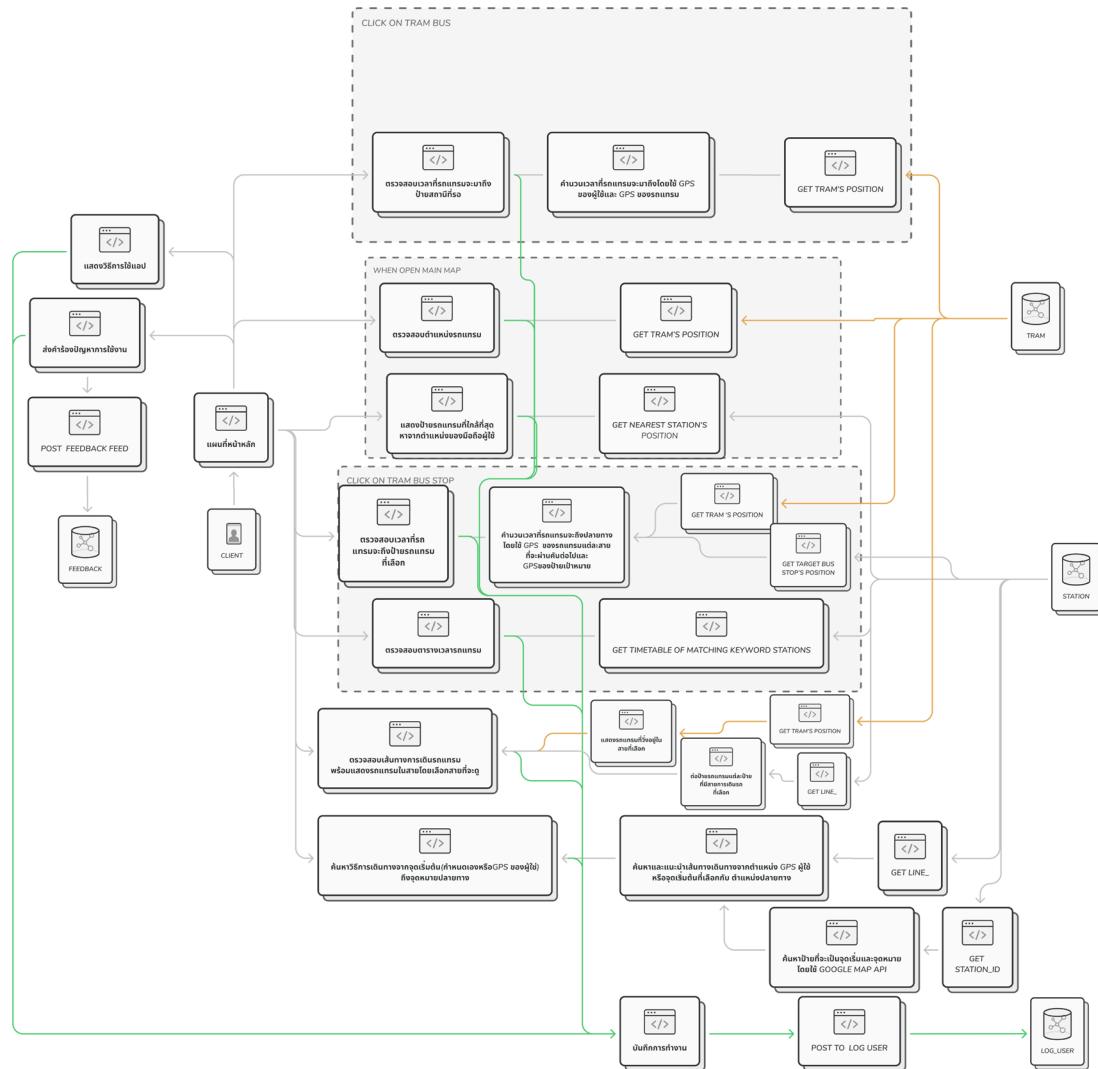
ตัวอย่างการดึง Database ของ User

### Service/API ที่ใช้

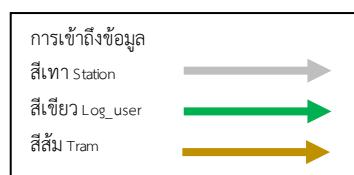
- Service เรียกดูข้อมูล Log user
- Service เรียกดูข้อมูลการเดินรถใน 1 วัน
- Service เรียกดูแบบตอบรับ
- Service เรียกดูข้อมูล Log service
- Service แก้ไขข้อมูลส่วนตัว(เฉพาะส่วน get info)

## Back-End (Services)

## ส่วนของ User

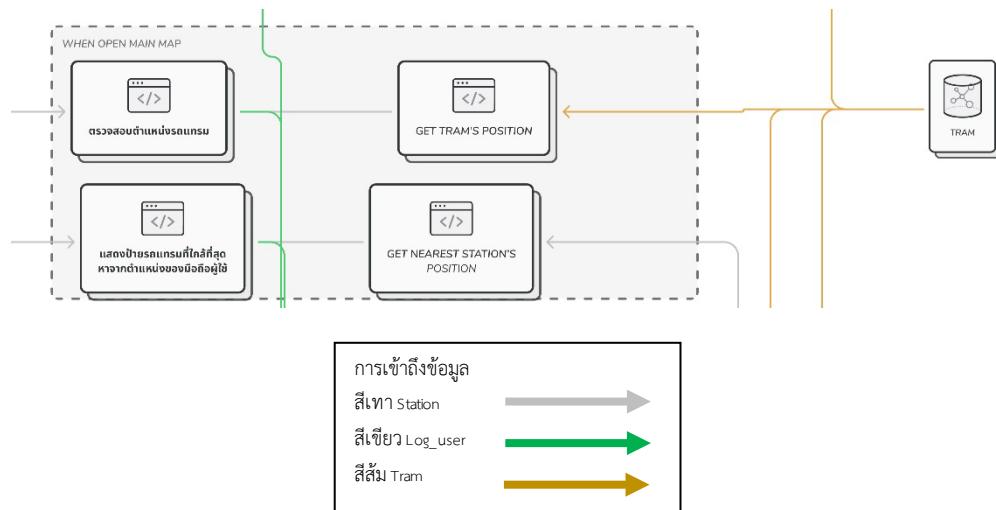


จะแสดงฟังก์ชันการทำงานของ User โดยจะมีการทำงานโดยรวมดังรูป



โดยเมื่อเข้าใช้งานผ่าน user ก็จะเข้าสู่หน้าແພນທີ່ຫລັກທີ່ແສດງຄ່າຕ່າງໆໃນແພນທີ່ ໂດຍຈະມີຄຳສັ່ງຢ່ອຍຕາມ  
ປະເທດກາຮະທຳຂອງ User ໂດຍຈະແບ່ງການทำงานດັ່ງນີ້

## 1. เมื่อเข้าแผนที่หน้าหลัก



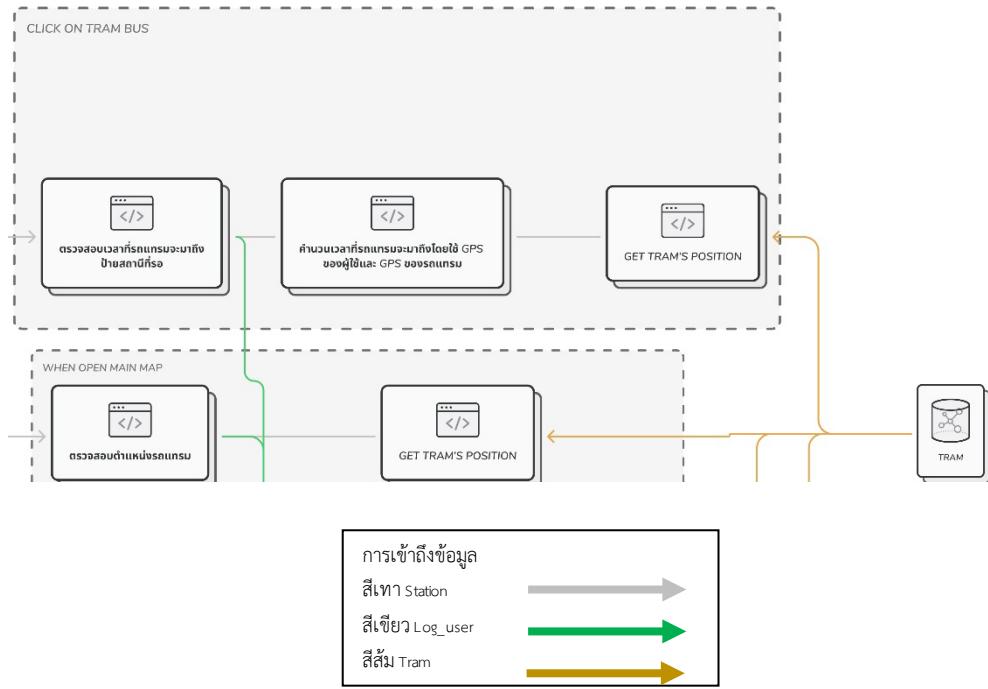
### ตรวจสอบตำแหน่งรถแทรม

เป็นการแสดงตำแหน่งรถแทรมจากในแผนที่โดยใช้ข้อมูลตำแหน่ง latitude และ longitude จากใน Tram database โดยใช้ Google map api ในการแสดงตำแหน่ง

### แสดงป้ายรถแทรมที่ใกล้ที่สุดจากตำแหน่งผู้ใช้

เป็นการแสดงป้ายที่มีตำแหน่งใกล้ที่สุดจากตัวผู้ใช้งานโดยเรียกใช้ตำแหน่งป้ายรถแทรมแต่ละป้ายกับตำแหน่งผู้ใช้แล้วมาเปรียบเทียบหาป้ายที่ใกล้ที่สุดแล้วมาแสดงผลทางหน้าจอ

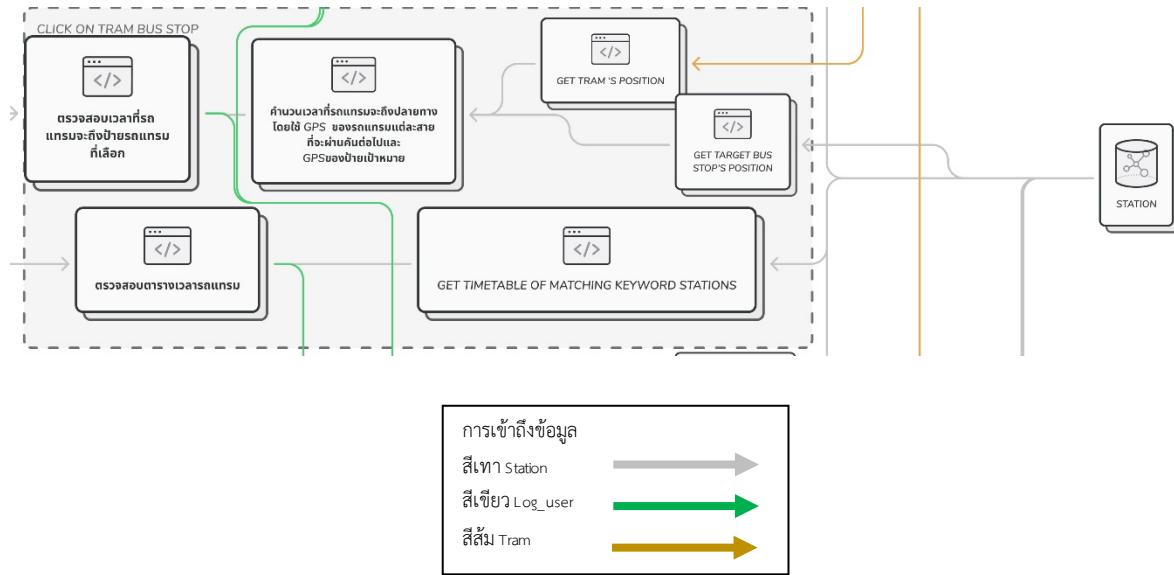
## 2. เมื่อกดที่รถแทรม



### ตรวจสอบเวลาที่รถแทรมจะมาถึงป้ายสถานีที่รอด

เมื่อกดที่รถแทรม จะทำการคำนวณเวลาที่รถแทรมคันนั้นจะมาถึงที่ป้ายรถแทรมที่รอด โดยจะดึงข้อมูลตำแหน่งของรถแทรมมาจาก Tram และใช้ข้อมูลตำแหน่งของ User มาคำนวณโดยใช้ Google Map API และแสดงผลเวลาที่รถแทรมจะมาถึงป้ายที่รอด

### 3. เมื่อกดที่ป้ายรถแทรม



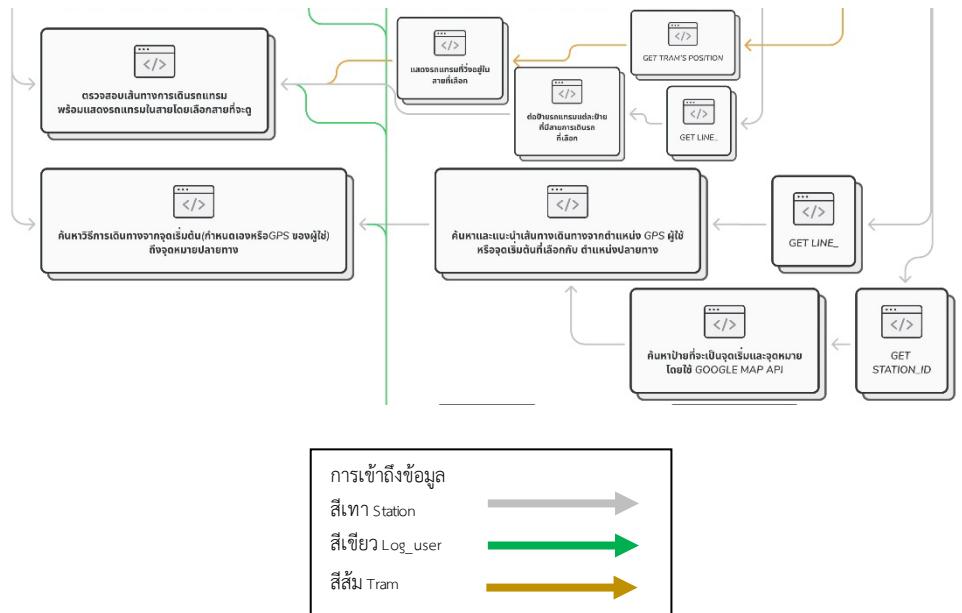
#### ตรวจสอบเวลาที่รถแทรมจะถึงป้ายรถแทรมที่เลือก

เมื่อ User กดที่ป้ายรถแทรมนั้น จะแสดงรถแทรมที่จะมาถึงป้ายที่เลือกคันล่าสุดของแต่ละสาย โดยจะเรียกใช้ข้อมูลตำแหน่ง Tram คันต่อไปที่จะผ่านป้ายรถแทรมที่เลือกในแต่ละสาย และเรียกดูตำแหน่ง Station ที่เลือกแล้วนำมารасคำนวนเวลาที่จะมีถึงโดยใช้ Google Map API

#### ตรวจสอบตารางเวลารถแทรม

จะแสดงตารางเวลาการมาถึงของรถแทรมในเวลาต่างๆ โดยจะแบ่งเป็นจังหวัด-ศุกร์ และเสาร์-อาทิตย์ โดยจะเรียกดู time\_table ของป้ายรถแทรมที่เลือกแล้วแสดงผล

#### 4. อื่นๆ

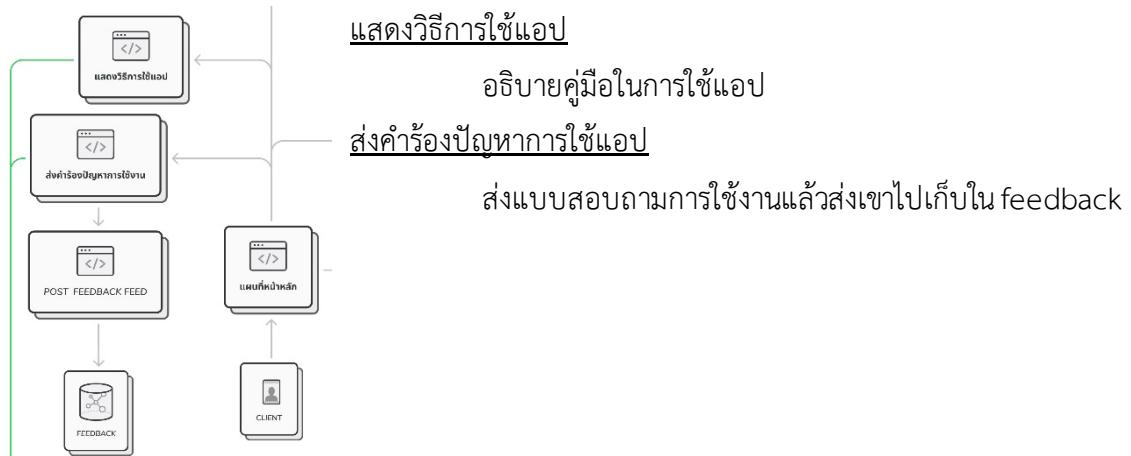


#### ตรวจสอบเส้นทางการเดินรถแทรมพร้อมแสดงรถแทรมในสาย

เป็นการให้ User เลือกว่าจะดูสายการเดินรถแทรมของสายไหน โดยพอดีกสาย ก็จะไปดึงข้อมูลจาก Station และทำการเชื่อมป้ายรถแทรมแต่ละป้ายที่มีสายที่เลือกโดยเริ่มจากสถานีต้นทางแล้วเลือกเชื่อมป้ายรถแทรมที่ตัวแปร next\_station\_id ของสายการเดินรถที่เลือก ทำต่อเรื่อยๆ จนวนมาที่สถานีต้นทาง และจะได้เส้นทางการเดินรถที่เลือกที่แสดงผล

#### ค้นหาวิธีการเดินทางจากจุดเริ่มต้นไปปลายทาง

โดยจะให้ User นำเข้าข้อมูลจุดเริ่มต้น(จะพิมพ์ค้นหาหรือใช้ GPS ของ User เอง) และจุดหมายปลายทางที่เลือก โดยการกำหนดจุดเริ่มและจุดหมายจะใช้เป็นป้ายรถแทรมที่ใกล้ที่สุดแล้วทำการทดลองเชื่อมป้ายรถแทรมจากตัวแปร next\_station\_id ใน station และถ้าจุดหมายอยู่ในสายรถแทรมที่อยู่ตรงข้ามถนนในข้อมูลป้ายรถแทรมใน station จะเก็บตัวแปร across\_station\_id เพื่อเก็บป้ายตรงข้ามของถนนเพื่อทำการข้ามถนนและหาเส้นทางการเดินรถจนกว่าจะถึงปลายทาง โดยที่ซึ่งค้นหาจะใช้ข้อมูลพื้นที่จาก Google map API ในการบอกว่าเมื่อเราพิมพ์ค้นหาสถานที่ แล้วจะได้ผลลัพธ์เป็นการบอกว่าสถานที่ที่ค้นหาคือที่ต่อไปนี้ แล้วนำสถานที่นั้นไปหาว่าป้ายที่ใกล้ที่สุดคือที่ไหน และกำหนดจุดนั้นว่าเป็นต้นทางและปลายทางและทำการค้นหาเส้นทางต่อไป



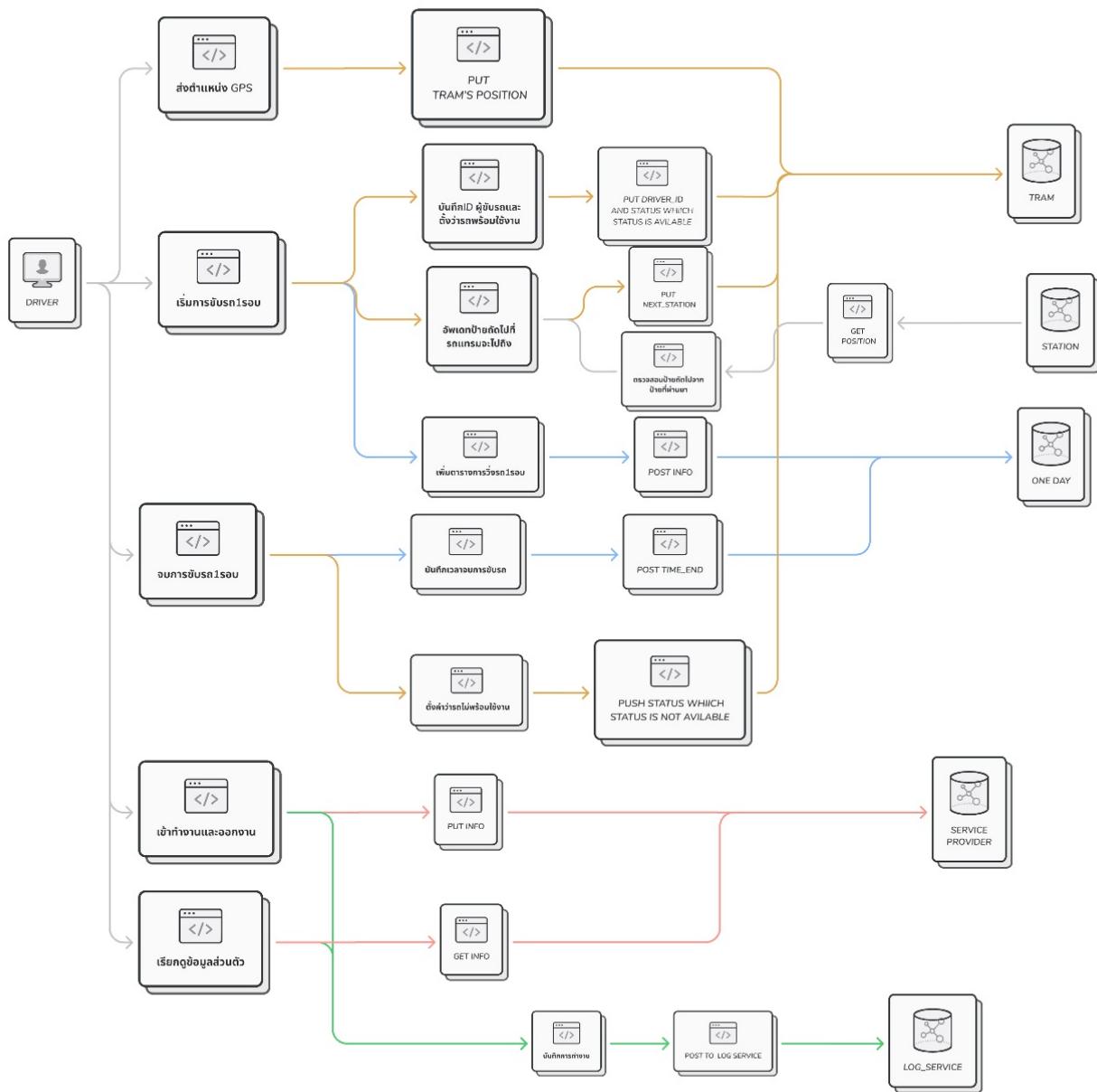
### ส่วนบันทึกการใช้งาน



โดยจะเก็บการกระทำทุกอย่างของ User ไว้เป็นประวัติการใช้งาน โดยจะเก็บทุกการกระทำ(สายสีเขียวที่อยู่ทุกฟังก์ชันการใช้งาน)ไว้ใน Log\_user

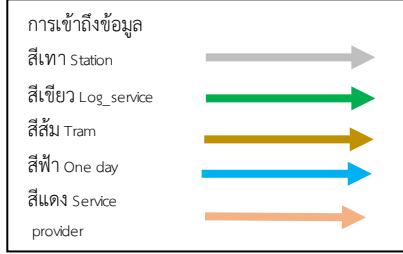
## ส่วนของ Driver

จะแสดงฟังก์ชันการทำงานของ Driver โดยจะมีการทำงานโดยรวมดังรูป



## 1. ส่งตำแหน่ง GPS

Driver จะส่งตำแหน่งรถแทรมผ่านทาง GPS แล้วเก็บใน Tram ตาม ID รถแทรมที่ขับ



## 2. เริ่มการขับรถ 1 รอบ

- เมื่อ Driver สแกน QR code ที่อยู่ที่รถแทรม จะถือว่าเป็นการเริ่มการขับรถ 1 รอบ โดยเมื่อเริ่มการขับรถจะมีกระบวนการย่ออยดังนี้
- บันทึก ID ของ Driver ใน Tram ที่ขับ แล้วตั้งสถานะของรถแทรมว่าพร้อมใช้งาน
  - อัพเดทป้ายรถแทรมที่จะถึงถัดไปแล้วใส่ไว้ใน Tram โดยจะตรวจสอบว่าแทรมป้ายถัดไปคือป้ายอะไรจากตำแหน่งใน Station
  - บันทึกการเดินรถรอบนั้นๆในวันนั้น ไว้ใน One\_day โดยบันทึกเวลาเริ่ม ID Driver และ ID รถแทรม

## 3. จบการขับรถ 1 รอบ

เมื่อ Driver ทำการขับรถ 1 รอบเสร็จแล้วกดปุ่มว่างรถเสร็จสิ้นแล้ว จะถือว่าจบการขับรถในรอบนั้นๆแล้ว โดยจะมีกระบวนการย่ออยดังนี้

- บันทึกเวลาจบการขับรถ โดยจะเก็บเวลาที่ขับรถเสร็จไว้ใน one day ในส่วนที่เพิ่มเข้ามาตอนเริ่มการขับรถ
- ตั้งค่าว่ารถไม่พร้อมใช้งาน โดยไปปรับ status ใน Tram ตาม tram\_id ที่ขับว่ารถไม่พร้อมให้บริการจนกว่าจะมีคนมาขับรถ

## 4. เข้าทำงานและออกงาน

จะเก็บเมื่อ Driver ได้ทำการ login และ logout เข้าสู่ระบบ โดยจะตั้ง status ว่าทำงานและเก็บเวลาเริ่มเมื่อเข้าสู่ระบบ และจะตั้ง status ว่าไม่ทำงานและเก็บเวลาจบเมื่ออกจากระบบ

## 5. เรียกดูข้อมูลส่วนตัว

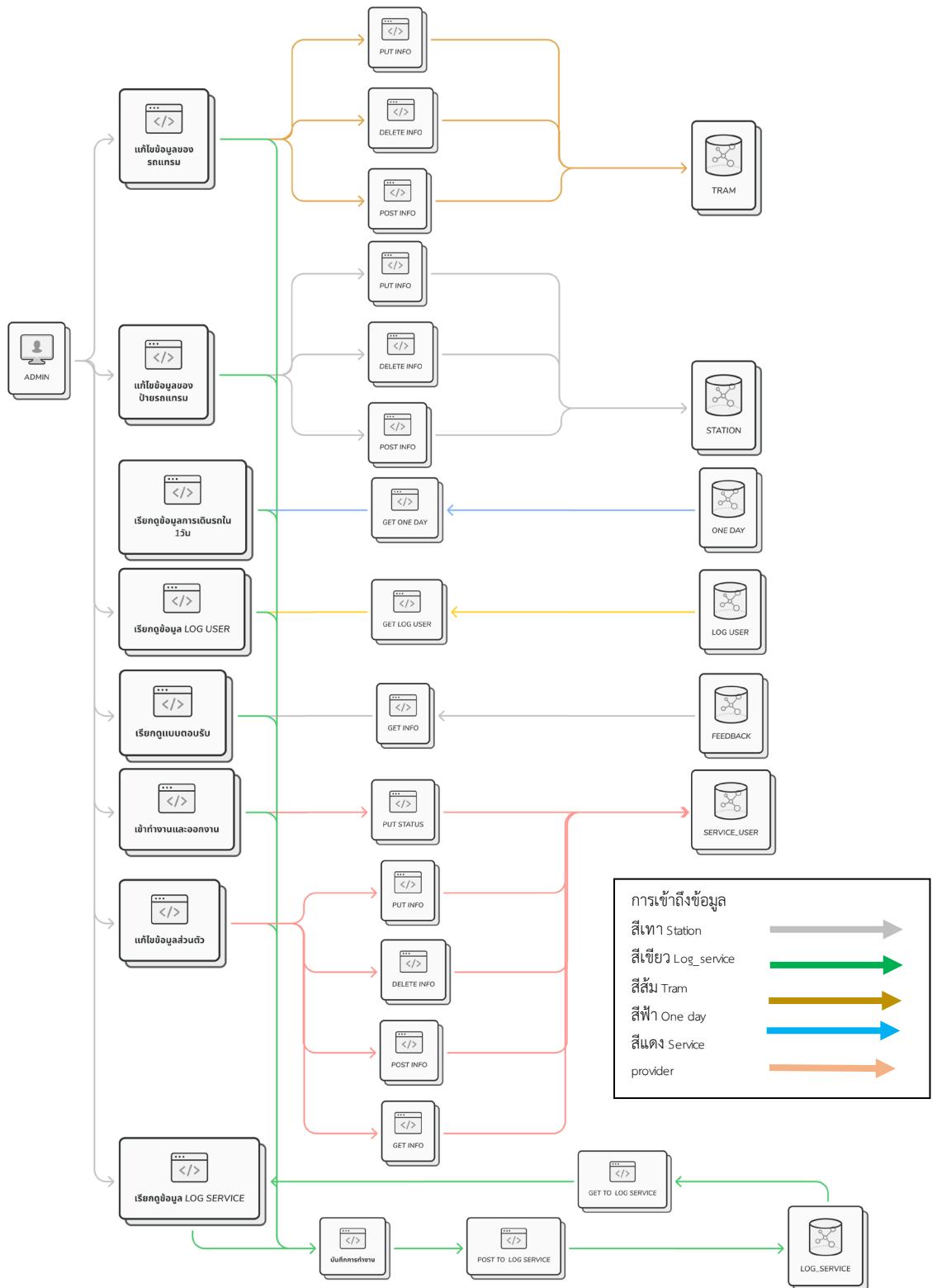
Driver จะสามารถดูข้อมูลส่วนตัวได้ คือ ID ชื่อ-นามสกุล อีเมล เบอร์โทรศัพท์ และตำแหน่งงาน

## 6. ส่วนบันทึกการใช้งาน

โดยจะเก็บการกระทำในส่วน เข้าทำงานและออกงาน และ เรียกดูข้อมูลส่วนตัว ไว้เป็นประวัติแล้วเก็บไว้ใน log\_service

## ส่วนของ Admin

จะแสดงฟังก์ชันการทำงานของ Admin โดยจะมีการทำงานโดยรวมดังรูป



## 1. แก้ไขข้อมูลส่วนรถแทรม

โดย Admin จะสามารถเพิ่ม ลบ แก้ ข้อมูลใน Tram ได้ โดยสามารถแก้ tram\_id, status, line\_ ได้

## 2. แก้ไขข้อมูลส่วนป้ายรถแทรม

โดย Admin จะสามารถเพิ่ม ลบ แก้ ข้อมูลใน Station ได้ โดยสามารถแก้ station\_id, position(latitude, longitude), land\_mark และข้อมูลใน line\_ ได้

## 3. ส่วนการเรียกดูข้อมูล

Admin นั้น จะสามารถดูข้อมูลส่วนไหนก็ได้ใน Database โดยจะเป็นข้อมูลที่ดูได้เท่านั้น ไม่สามารถแก้ได้ ซึ่งแต่ละข้อมูลก็จะมี Filter ที่ใช้เรียกดูแตกต่างกัน โดยมีดังนี้

- one day filter วันที่ที่ต้องการดู
- feedback filter ประเภทของ feedback ที่ต้องการดู
- service user filter ใช้ service\_user\_id ที่จะดู
- log service filter log\_service\_id, service\_user\_id, job\_position, service\_action.type
- log user filter log\_user\_id, user\_id, user\_action, nearest\_station\_id, log\_time

## 4. ส่วนบันทึกการใช้งาน

โดยจะเก็บการกระทำทั้งหมดของ Admin ไว้เป็นประวัติแล้วเก็บไว้ใน log\_service

## Database

### Database : service\_provider

ในส่วนของตัวละครที่จะใช้ในโปรเจคนี้คือ 3 ตัวละครคือ ผู้ใช้ทั่วไปหรือ User Driver และ Admin โดย 2 ตัวละครหลังจะสามารถตรวจนิ่งว่าเป็นผู้ให้บริการหรือ Service provider โดย Database นี้จะเป็น Database ไว้เก็บข้อมูลทั่วไปของส่วน Service Provider แต่ส่วนของ User ไม่ได้มีการใช้ Login เข้าสู่ระบบ จึงไม่มีการเก็บข้อมูลส่วนนี้จากทาง User โดยมีโครงสร้างดังนี้

```
{
    "service_provider_id": "number",
    "first_name": "string",
    "last_name": "string",
    "job_position": "string",
    "email": "string",
    "phone_no": "string",
    "status": "number",
    "password": "string",
    "period": [
        {
            "date_": "date",
            "time_in": [
                {
                    "time_in": "time",
                    "time_out": "time"
                }
            ]
        }
    ]
}
```

### อธิบายตัวแปรใน Database

**service\_provider\_id :**

รหัสประจำตัวของ Service provider ที่ใช้ในการระบุ Service Provider แต่ละคน

**first\_name & last name :**

เก็บชื่อและนามสกุลของ Service Provider

**job\_position :**

ระบุตำแหน่งงานของ Service Provider แต่ละคนว่าอยู่ตำแหน่งอะไรระหว่าง Driver กับ Admin

**email :**

บอกรถีทางอีเมลของ Service provider

**phone\_no :**

บอกรถีทางโทรศัพท์ของ Service provider

**status :**

ระบุว่า Service Provider คนนั้นๆ ได้ทำงานอยู่หรือไม่ โดยกำหนดให้ 0 = ไม่ทำงาน 1 = ทำงาน และอาจจะมีระบบสถานะเพิ่มเติมในอนาคต

**period :**

บันทึกวันเวลาที่ได้เข้าทำงานวันเวลาไหนบ้าง โดยจะสามารถเก็บได้หลายช่วงเวลา โดยจะมีตัวแปรย่ออยดังนี้

**date\_** : ระบุวันที่เข้าระบบ

**time\_in & time\_out** : เก็บเวลาที่เข้า-ออกงาน โดย 1 วันอาจจะมีการเข้า-ออกงานหลายรอบก็ได้

**Database : tram**

เป็น Database ส่วนของรถแทรมแต่ละคันโดย มีโครงสร้างดังนี้

```
{
    "tram_id": "number",
    "service_provider_id": "number",
    "line": "number",
    "status": "number",
    "next_station": "number",
    "position": {
        "latitude": "number",
        "longitude": "number"
    }
}
```

อธิบายตัวแปรใน Database**tram\_id :**

รหัสประจำตัวของแต่ละรถแทรม ที่ใช้ในการระบุความเป็นรถแทรมแต่ละคัน

**service\_provider\_id(เฉพาะ Driver) :**

รหัสประจำตัวของ Service provider หรือ Driver ที่ใช้ในการระบุคนขับรถแทรมแต่ละคันในขณะนั้น

**line :**

ระบุสายของรถแทรมว่ารถแทรมคันนั้นๆ วิ่งสายไหน โดยจะแบ่งคือ 1: สาย MLC 2: สายสีน้ำเงิน

3: สายสีแดง 4: สายสีเขียว

**status :**

ระบุสถานะของรถแทรม โดยกำหนดให้ 0 = ไม่ทำการวิ่ง 1 = ทำการวิ่ง 2 = ซ่อมบำรุง 3 = ปลดประจำการ และอาจจะมีระบุสถานะเพิ่มเติมในอนาคต

**position :**

ระบุตำแหน่งของรถแทรมที่จะมีการอัพเดตตลอดเวลาที่มีการวิ่งของรถโดยจะเก็บเป็น latitude และ longitude

## Database : station

ไว้เก็บข้อมูลของป้ายรถแทรมของแต่ละป้าย โดยข้อมูลเส้นทางจะเก็บเป็นให้ป้ายรถแทรมต่อๆ กันจนเป็นสาย โดยมีโครงสร้างดังนี้

```
{
    "station_id": "number",
    "latitude": "number",
    "longitude": "number",
    "land_mark": ["string"],
    "line_": [
        {
            "line": "number",
            "previous_station_id": "number",
            "next_station_id": "number",
            "across_station_id": "number",
            "time_table": {
                "monday-friday": ["time"],
                "saturday-sunday": ["time"]
            }
        }
    ]
}
```

### อธิบายตัวแปรใน Database

**station\_id :**

รหัสของแต่ละป้ายรถแทรม ที่ใช้ในการระบุป้ายรถแทรมแต่ละป้าย

**latitude & longitude :**

ระบุตำแหน่งของป้ายแทรมที่ใช้บอกตำแหน่งรถแทรม โดยตำแหน่งจะคงที่ตลอด

**landmark :**

ไว้เก็บตึกริเวณป้ายโดยเมื่อใช้ข้อมูลตึกหรือคณะในการค้นหาจะทำการแนะนำป้ายบริเวณนั้นที่ใกล้ที่สุดให้ โดยอาจจะมา Landmark ได้หลายตึกต่อ 1 ป้าย

**line\_ :**

ใช้เก็บว่าป้ายนั้นมีสายเดินรถสายไหนผ่านป้ายนี้บ้าง โดยแต่ละสายก็จะมีวิธีการดินรถแตกต่างกัน และเก็บข้อมูลเฉพาะของแต่ละสาย โดยจะมีข้อมูลดังนี้

**line :**

ระบุสายของรถแทรมว่ารถแทรมคันนั้นๆ วิ่งสายไหน โดยจะแบ่งคือ 1: สาย MLC  
2: สายสีน้ำเงิน 3: สายสีแดง 4: สายสีเขียว

**previous\_station\_id :**

เก็บข้อมูลว่าป้ายก่อนหน้านี้ของรถแทรมสายนั้นผ่านป้ายอะไรมา ไว้ทำการเชื่อมป้ายแต่ละป้ายเป็นเส้นทาง

**next\_station\_id :**

เก็บข้อมูลว่าป้ายต่อไปของรถแทรมสายนั้นคือป้ายไว้ทำการเชื่อมป้ายแต่ละป้ายเป็นเส้นทาง และใช้สำหรับระบบแนะนำเส้นทาง

**across\_station\_id :**

เก็บข้อมูลว่าป้ายที่อยู่ติดกันฝั่งตรงข้ามคือป้ายอะไร ไว้สำหรับระบบแนะนำเส้นทางเมื่อการเดินทางต้องเปลี่ยนฝั่งถนนเพื่อใช้ในการเดินทาง

**time\_table :**

เก็บตารางเวลาที่รถแทรมแต่ละสายจะถึงป้ายแต่ละป้าย โดยจะแบ่งวันเป็น จันทร์-ศุกร์ และ เสาร์-อาทิตย์

**Database : feedback**

ไว้เก็บข้อมูลข้อคิดเห็นการใช้งานของโปรแกรมจาก user โดยจะมีโครงสร้างดังนี้

```
{
  "feedback_id": "number",
  "type": ["string"],
  "note": "string"
}
```

**อธิบายตัวแปรใน Database****station\_id :**

รหัสของแต่ละ feedback ที่ส่งมา

**type :**

ไว้เก็บว่าปัญหาที่ได้รับมานั้นอยู่ในประเภทอะไร อาทิ ความเร็วการใช้งาน ปัญหาทรัพยากร หรือ ความสะดวกสบายในการใช้

note :

ไว้เก็บข้อมูลรายละเอียดประวัติการเดินรถในรอบ 1 วันของแต่ละวัน โดยจะมีโครงสร้างดังนี้

## Database : one\_day

ไว้เก็บข้อมูลรายละเอียดประวัติการเดินรถในรอบ 1 วันของแต่ละวัน โดยจะมีโครงสร้างดังนี้

```
{
    "date_id": "number",
    "date_": "date",
    "bus_history": [
        {
            "service_provider_id": "number",
            "tram_id": "number",
            "line": "number",
            "time_start": "time",
            "time_end": "time"
        }
    ]
}
```

### อธิบายตัวแปรใน Database

date\_id :

รหัสของแต่ละข้อมูลของ one\_day

date\_ :

เก็บข้อมูลวันว่าข้อมูลที่บันทึกนั้นว่าบันทึกของวันไหน

bus\_history :

เก็บประวัติการเดินรถในรอบ 1 วันโดยจะมีรายละเอียดอยู่ ดังนี้

service\_provider\_id(เฉพาะ Driver) :

รหัสประจำตัวของ Service provider หรือ Driver ที่ใช้ในการระบุคนขับรถแทรมในรอบนั้น

tram\_id :

รหัสประจำตัวรถแทรมที่ขับในรอบนั้น

line :

ระบุว่ารถแทรมคันนี้ได้ขับรถสายไหน

time\_start&time\_end :

ระบุเวลาเริ่ม-จบของการเดินรถแทรมในรอบนั้น

## Database : log\_service

ไว้เก็บประวัติการเข้าถึงข้อมูลของผู้ Service provider โดยจะมีวิธีกี่เก็บข้อมูลดังนี้

```
{
    "log_service_id": "number",
    "service_provider_id": "number",
    "job_position": "string",
    "login_time": "datetime",
    "logout_time": "datetime",
    "service_action": [
        {
            "type": "string",
            "time": "datetime",
            "access_position": {
                "database_name": "string",
                "access_id": "number",
                "position_command": "string"
            },
            "edit": {
                "previous_info": "string",
                "edit_info": "string"
            },
            "delete_info": "string"
        }
    ]
}
```

### อธิบายตัวแปรใน Database

**log\_service\_id :**

รหัสประจำตัวของ Log ที่บ่งบอกว่าคือ Log อะไร

**service\_provider\_id :**

รหัสประจำตัวของ Service Provider ว่าเก็บ log มาจากใคร

**job\_position :**

ระบุว่า Service Provider คนนั้นอยู่ตำแหน่งอะไร Driver หรือ Admin

**Login\_time&logout\_time :**

ระบุเวลาที่อยู่ในระบบของ Service Provider คนนั้น

**service\_action :**

เก็บรายละเอียดการเข้าถึงข้อมูลของ Service Provider คนนั้น โดยมารายละเอียดโดย ดังนี้

**type :**

ระบุประเภทของการเข้าถึงข้อมูล ว่าคือ การอ่าน เพิ่ม แก้ หรือลบข้อมูล

**time :**

ระบุเวลาที่เข้าถึงข้อมูลนั้น

**access\_position :**

ระบุตำแหน่งที่เข้าถึงข้อมูลว่าเข้าถึงตำแหน่งไหน โดยมีรายละเอียดคือ

database\_name : ระบุชื่อ Database ที่เข้าถึง

access\_id : ระบุ ID ใน Database ที่เข้าถึง

position\_command : เก็บตำแหน่งที่เข้าถึงโดยเก็บเป็น command ของ Nosql ที่เข้าถึง  
เนื่องจากไม่สามารถระบุความลึกของการ query ข้อมูลทุกข้อมูลแบบเจาะจงได้

**edit :**

ถ้าเป็นการแก้ไขข้อมูลจะทำการเก็บข้อมูลเก่าไว้ใน previous\_info และเก็บข้อมูลใหม่ไว้ใน

**edit\_info****delete\_info :**

ถ้าเป็นการลบข้อมูลก็จะเก็บข้อมูลที่ลบในตัวแปรนี้

**Database : log\_user**

ໄwake กับประวัติการใช้งานแอปของผู้ใช้ Service provider โดย 1 log จะเก็บเพียง 1 การใช้งานเท่านั้น โดยจะมีรีสิรีกีกับข้อมูลดังนี้

```
{
    "log_user_id": "number",
    "user_id": "number",
    "log_time": {
        "action_date": "date",
        "action_time": "time"
    },
    "log_position": {
        "latitude": "number",
        "longitude": "number",
        "nearest_station_id": "number"
    },
    "user_action": {
        "action": "string"
    }
}
```

**อธิบายตัวแปรใน Database****Log\_user\_id :**

รหัสประจำตัวของ log ที่บ่งบอกว่าคือ log อะไร

**user\_id :**

รหัสประจำตัวของ user ว่าเก็บ log มาจากใคร โดยเนื่องจากแอปไม่ต้อง login เพื่อเข้าใช้งานจึงเก็บเป็น ID ของโทรศัพท์แทน

**log\_time :**

ระบุเวลาในการใช้งานฟังก์ชันของแอปนั้นๆ โดยจะเก็บวันและเวลาใน action\_date กับ action\_time

**log\_position :**

เก็บข้อมูลว่าได้ใช้งานฟังก์ชันของแอปที่บริเวณไหน โดยเก็บเป็น latitude กับ longitude และ nearest\_station\_id ที่บอกรายละเอียดว่าได้ใช้งานฟังก์ชันของแอป

**user\_action :**

ระบุรายละเอียดว่าได้ใช้งานฟังก์ชันอะไร โดย 1 user\_action จะเก็บเพียงการทำงานเดียวเท่านั้น โดยมีประเภทการทำงานดังนี้

```
"user_action":{  
    "tram_click":{  
        "tram_id":"number",  
        "line": "number"  
    },  
    "station_click":{  
        "station_id":"number",  
        "check_time_table":"boolean",  
        "time_table":[{  
            "time_table_line":"number"  
        }]  
    },  
    "line_click":"boolean",  
    "click_on_guide":"boolean",  
    "feedback_id":"number",  
    "search_way":{  
        "begin":{  
            "search_begin":"string",  
            "station_start_id":"number"  
        },  
        "end":{  
            "search_end":"string",  
            "station_end_id":"number"  
        }  
    }  
}
```

**tram\_click :**

บอกว่าได้กดดูรถแทรมคันไหน โดยบอกเป็น tram\_id และสายรถแทรมที่วิ่ง

**station\_click :**

บอกว่าได้กดดูป้ายรถแทรมป้ายไหน โดยบอกเป็น station\_id และเช็คว่าได้กดดูส่วน timetable ใหม่และดู timetable ของสายอะไร

**line\_click :**

บอกว่าได้เลือกสายที่ดูหรือไม่ และดูที่สายไหน

**click\_on\_guide :**

เช็คว่าได้ทำการดูวิธีการใช้งานแอปใหม่

**feedback\_id :**

เช็คว่ามีการส่ง feedback ใหม่โดยเก็บเป็น feedback\_id ที่ส่ง

**search\_way :**

เก็บข้อมูลการใช้งานฟังก์ชันการแนะนำเส้นทางโดยเก็บข้อมูลที่ค้นหา(พิมพ์ค้นหาว่าอะไร) ของจุดเริ่มและจุดสิ้นสุดใน search\_begin และ search\_end และป้ายรถแทรมที่เป็นจุดมาร์ค เริ่มต้น และสิ้นสุดที่ station\_start\_id และ station\_end\_id ตามลำดับ

## ข้อมูลที่เกี่ยวข้อง

Git hub ของโครงการ

[https://github.com/Domineice/Tram\\_tracking\\_draft](https://github.com/Domineice/Tram_tracking_draft)

Backend design

<https://www.figma.com/files/project/43364045/Team-project?fuid=1044613070526616251>

UI ส่วน User และ Driver Application

<https://www.figma.com/file/VhyuUDX6ykVqMXXOXRMFaD/Tramer?node-id=0%3A1>

UI ส่วน Admin

[https://www.figma.com/file/9DNdWLhyLUTpgVUu2d8bM1/tram\\_tracking?node-id=0%3A1](https://www.figma.com/file/9DNdWLhyLUTpgVUu2d8bM1/tram_tracking?node-id=0%3A1)

## บรรณานุกรม

Google. (ม.ป.ป.). *FirBase Product*. เรียกใช้เมื่อ 14 ธันวาคม 2021 จาก FirBase:

<https://firebase.google.com/products-build>

Google. (ม.ป.ป.). *Google Maps Reference*. เรียกใช้เมื่อ 14 ธันวาคม 2021 จาก Google Maps

Platform: <https://developers.google.com/maps/documentation/javascript/reference>