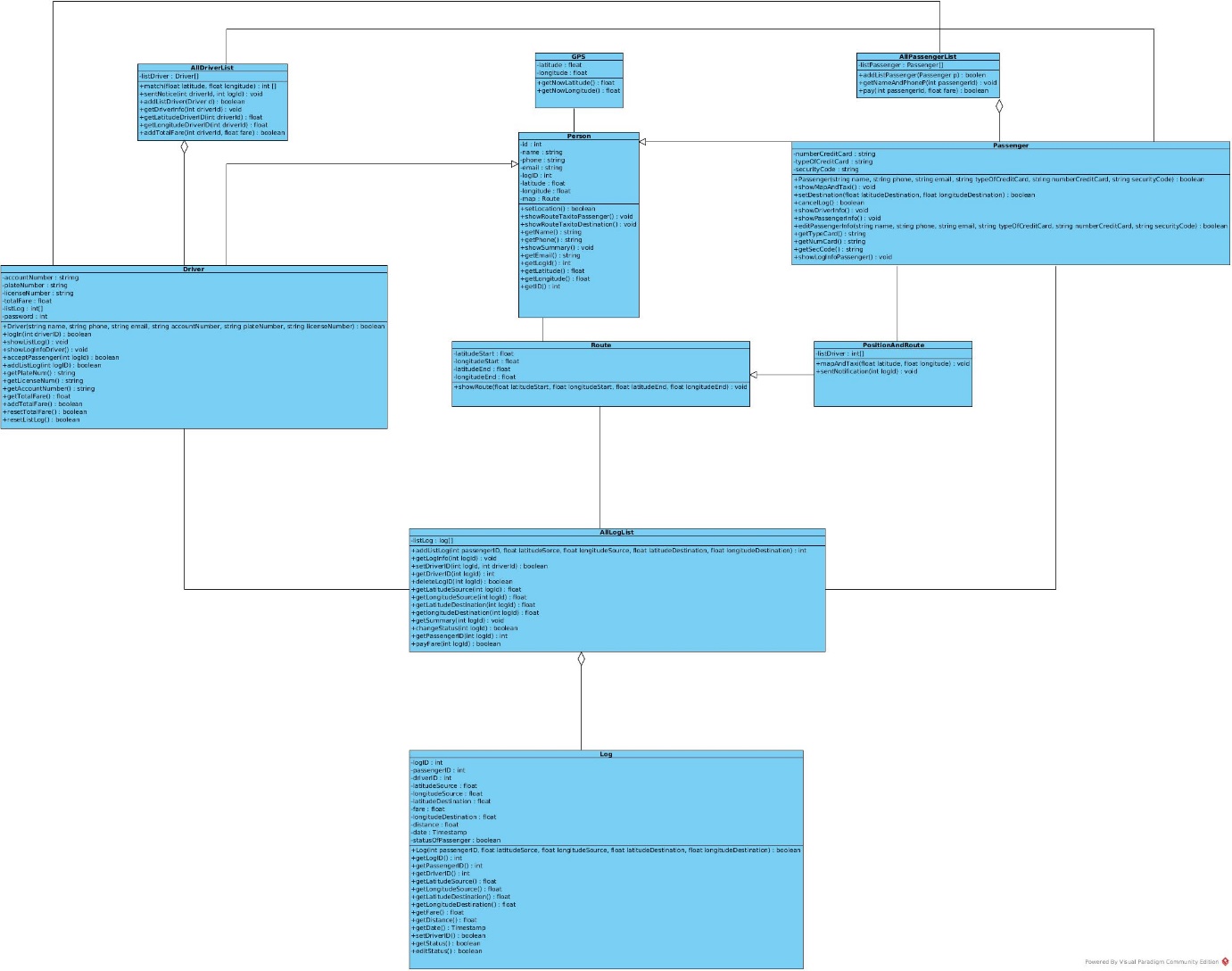
**Mali Taxi**

**Software Design Document II**

10/04/2016

v.1.0

**4. COMPONENT-LEVEL DESIGN**



Component level designed ของ Mali Taxi แบ่งออกเป็น 3 Component หลัก คือ

1. Passenger component

2. Driver component

3. Server component

4.1. DESCRIPTION COMPONENT 1: Passenger component

4.1.1. PROCESSING NARRATIVE (PSPEC) FOR COMPONENT 1

Passenger Component สร้างขึ้นเพื่อใช้จัดการในแอปพลิเคชั่นส่วนของผู้โดยสาร และ ข้อมูลของผู้โดยสารแต่ละบุคคลประกอบไปด้วย

Class Person เป็น Class ที่เก็บข้อมูลพื้นฐานของบุคคล Class นี้จะถูกใช้ร่วมกันทั้ง Passenger Component และ Driver Component ประกอบไปด้วย Attribute พื้นฐานดังต่อไปนี้ id, name, phone และ email และ Attribute เพิ่มเติม คือ logID, latitude และ longitude เพื่อใช้ในการ ระบุ logID ที่ใช้อยู่ในปัจจุบันของ แอปพลิเคชั่น และ ตำแหน่งที่อยู่ปัจจุบัน และมี Method setLocation, showRouteTaxitoPassenger, showRouteTaxitiDestination, getName, getPhone, showSummary, getEmail, getLogId, getLatitude, getLongitude และ getID

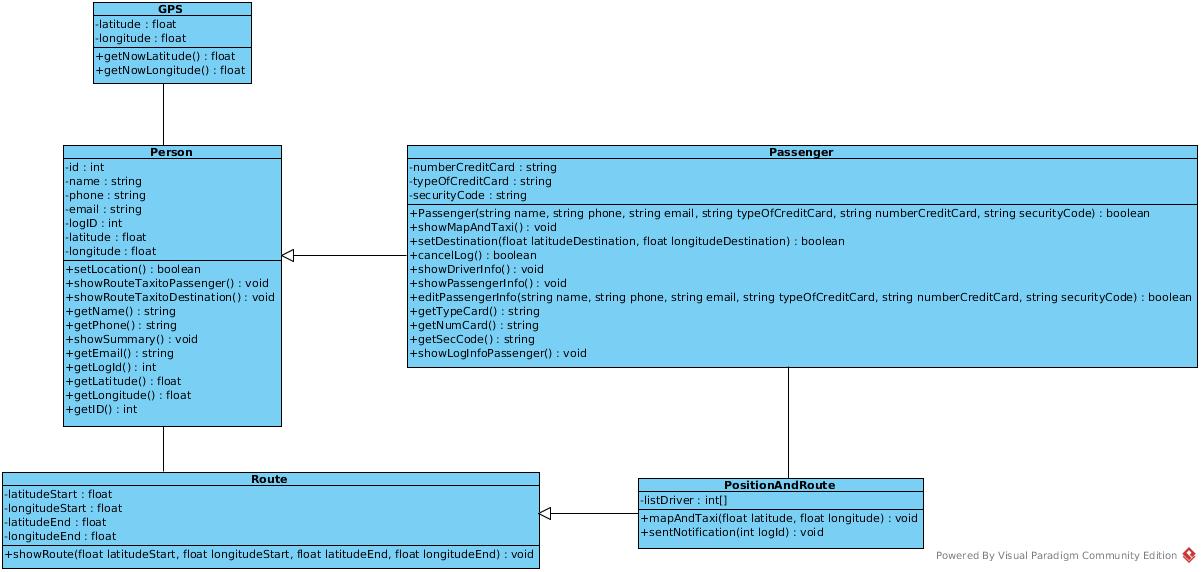
Class Passenger เป็น Class ที่สืบทอดมาจาก Class Person และเป็น Class หลักของ Passenger Component เพื่อใช้ในการเก็บข้อมูลของผู้โดยสาร และติดต่อกับ Server Component ในการจัดการแอปดพลิเคชั่น ประกอบไปด้วย Attribute numberCreditCard, typeOfCreditCard, securityCode เพื่อใช้สนับสนุนระบบการชำระค่าโดยสารอัตโนมัติ มี Method Passenger, showMapAndTaxi, setDestination, cancelLog, showDriverInfo, showPassengerInfo, editPassngerInfo, getTypteCard, getNumCard, getSecCode และ showLogInfoPassenger

Class GPS เป็น Class ที่ใช้ในการดึงค่าตำแหน่งที่อยู่ปัจจุบันของเครื่อง เพื่อในการเรียกบริการ ประกอบไปด้วย Attribute latitude, longitude เพื่อเก็บค่าตำแหน่งที่อยู่ปัจจุบัน มี Method getNowLatitude, getNowLongitude

Class Route เป็น Class ที่ใช้ในการ แสดงเส้นทางและแผนที่จากจุดเริ่มต้นถึงจุดหมายปลายทาง Class นี้จะถูกใช้ร่วมกันทั้ง Passenger Component และ Driver Component และใช้ในการติดต่อกับ Server Component ประกอบด้วย Attribute latitudestart, longitudeStart, latitudeEnd, longitudeEnd เพื่อใช้ระบุตำแหน่งเริ่มต้นและตำแหน่งสิ้นสุด มี Method showRoute

Class PositionAndRoute เป็น Class ที่สืบทอดมาจาก Class Route เพื่อใช้ สร้างแผนที่ที่แสดงจุดที่มีรถโดยสารและจุดที่มีผู้โดยสาร และติดต่อกับ Server Component ในการส่งการแจ้งเตือนไปยังผู้ขับที่ถูก Match กับ ผู้โดยสาร ประกอบด้วย Attribute listDriver เพื่อเก็บข้อมูล driverID ทั้งหมดที่ผู้โดยสารถูก Match ด้วย มี Method mapAndTaxi และ sentNotification

4.1.2. COMPONENT 1 PROCESSING DETAIL

4.1.2.1. Design Class hierarchy for component 1

4.1.2.2. Restrictions/limitations for component 1

* การใช้งาน Passenger Component จะถูกใช้งานได้เมื่อใช้ผ่านแอปพลิเคชั่นของ Passenger เท่านั้น
* จำเป็นจะต้องมีบัตรเครดิตจึงสามารถสมัครสมาชิกได้
* การเข้าสู่ระบบและการสมัครสมาชิกต้องผ่าน Facebook จึงจะสามารถใช้งานได้
* จำเป็นต้องใช้งานผ่านอินเตอร์เน็ต จึงสามารถเชื่อมต่อกับ Component อื่นได้

4.1.2.3. Performance issues for component 1

* ไม่สามารถเชื่อมต่อกับ Driver Component ที่เป็นผู้ให้บริการได้โดยตรง
* หากไม่มีบัตรเครดิตจะไม่สามารถสมัครสมาชิกและใช้บริการได้
* จำเป็นจะต้องเป็นสมาชิกของ Facebookและลงทะเบียนก่อน
* หากไม่สามารถใช้บริการอินเตอร์เน็ตจะไม่สามารถใช้บริการแอปพลิเคชั่นได้

4.1.2.4. Design constraints for component 1

* เมื่อไม่มีผู้ขับคนใดอยู่ในบริเวณที่ผู้โดยสารเรียกใช้บริการ จะไม่สามารถให้บริการแก่ผู้โดยสารในครั้งนั้นได้

4.1.2.5. Processing detail for each operation of component n

**Class Person**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

setLocation () : เป็น Pubilc Method ที่ไม่จำเป็นต้องมี Parameter เมื่อทำการเรียกใช้จะทำการเรียก Method getNowLatitude, getNowLongitude ของ Class GPS เเล้วบันทึกค่าลงใน Attribute หากสำเร็จ Return True และไม่สำเร็จ Return False

4.1.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

เรียกใช้ Method getNowLatitude และ getNowLongitude จาก Class GPS

หากสำเร็จ มีค่า return กลับมา

เก็บค่า latitude และ longitude ลงใน Attribute

Return true

หากไม่สำเร็จ

Return false

END

4.1.2.5.2.1 Processing narrative (PSPEC) for each operation

showRouteTaxitoPassenger () : เป็น Public Method ที่ไม่ต้องการ Parameter เพื่อแสดงเส้นทางและระยะทางระหว่าง ผู้โดยสารและผู้ขับที่ทำการตอบรับการเรียกใช้บริการ โดยจะนำ logID ส่งไปยัง Method getDriverID ใน Class AllLogList ของ Server Component เพื่อทำการเรียก driverID ที่ให้บริการในครั้งนั้น แล้วส่ง driverID นั้นไปยัง Method getLatitudeDriverID, getLongitudeDriverID ใน Class AllDriverList ของ Server Component เพื่อหาตำแหน่งของผู้ขับในขณะนั้น และส่ง logID ส่งไปยัง Method getLatitudeSource, getLogitudeSource ใน Class AllLogList ของ Server Component เพื่อทำการเรียกตำแหน่งของจุดที่ผู้โดยสารต้องการให้ไปรับออกมา แล้วเรียกใช้ map ส่ง latitude, longitude ของ ผู้ขับเป็นจุดเริ่มต้นและ latitude, longitude ของ ผู้โดยสารเป็นจุดสิ้นสุด ไปยัง Method showRoute ใน Class Route เพื่อทำการแสดงเป็น เส้นทางและแผนที่ออกมา

4.1.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

นำ logID ส่งไปยัง Method getDriverID ใน Class AllLogList

นำ driverID มารับค่าจาก Method

ส่ง driverID นั้นไปยัง Method getLatitudeDriverID, getLongitudeDriverID ใน Class AllDriverList

นำตัวแปรมารับค่า

ส่ง logID ไปยัง Method getLatitudeSource, getLogitudeSource ใน Class AllLogList

นำตัวแปรมารับค่า

ส่ง latitude, longitude ของ ผู้ขับและ latitude,longitude ของ ผู้โดยสารไปยัง Method showRoute ใน Class Route

แสดงเป็น เส้นทางและแผนที่ออกมา

END

4.1.2.5.3.1 Processing narrative (PSPEC) for each operation

showRouteTaxitoDestination () : เป็น Public Method ที่ไม่ต้องการ Parameter เพื่อแสดงเส้นทางและระยะทางระหว่าง ผู้ขับไปถึงจุดหมายปลาายทาง โดยจะนำ logID จากผู้เรียกใช้ ส่งไปยัง Method getDriverID ใน Class AllLogList ของ Server Component เพื่อทำการเรียก driverID ที่ให้บริการในครั้งนัั้น แล้วส่ง driverID นั้นไปยัง Method getLatitudeDriverID, getLongitudeDriverID ใน Class AllDriverList ของ Server Component เพื่อหาตำแหน่งของผู้ขับในขณะนั้น พร้อมกับนำ logID ส่งไปยัง Method getLatitudeDestination, getLogitudeDestination ใน Class AllLogList ของ Server Component เพื่อทำการเรียกตำแหน่งของจุดหมายปลายทางออกมา แล้วส่ง Latitude, Longitude ของ ผู้ขับเป็นจุดเริ่มต้นและ Latitude, Longitude ของ จุดหมายปลายทางเป็นจุดสิ้นสุด ไปยัง Method showRoute ใน Class Route เพื่อทำการแสดงเป็น เส้นทางและแผนที่ออกมา

4.1.2.5.3.2 Algorithmic model (e.g., PDL) for each operation

START

นำ logID ส่งไปยัง Method getDriverID ใน Class AllLogList

นำตัวแปรมารับค่า driverID

ส่ง driverID ไป Method getLatitudeDriverID, getLongitudeDriverID ใน Class AllDriverList

นำตัวแปรมารับค่า

นำ logID ส่งไปยัง Method getLatitudeDestination, getLogitudeDestination ใน Class AllLogList

นำตัวแปรมารับค่า

ส่ง latitude, longitudeของ ผู้ขับและ latitude, longitude ของ จุดหมายปลายทางไปยัง Method showRoute ใน Class Route

แสดงเป็น เส้นทางและแผนที่ออกมา

END

4.1.2.5.4.1 Processing narrative (PSPEC) for each operation

getName () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return name กลับไป

4.1.2.5.4.2 Algorithmic model (e.g., PDL) for each operation

START

Return name

END

4.1.2.5.5.1 Processing narrative (PSPEC) for each operation

getPhone () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return phone กลับไป

4.1.2.5.5.2 Algorithmic model (e.g., PDL) for each operation

START

Return phone

END

4.1.2.5.6.1 Processing narrative (PSPEC) for each operation

showSummary () : เป็น Public Method ที่ไม่ต้องการ Parameter เพื่อแสดงเมื่อผู้ขับ ถึงจุดหมายปลายทางแล้ว โดยจะเรียกส่ง logID ไปยัง Method getSummary ใน Class AllLogList ของ Server Component จะแสดง ราคา ระยะทาง และเวลาที่ใช้

4.1.2.5.6.2 Algorithmic model (e.g., PDL) for each operation

START

ส่งlogID ไปยัง Method getSummary ใน Class AllLogList

แสดง ราคา ระยะทาง และ เวลาที่ใช้

END

4.1.2.5.7.1 Processing narrative (PSPEC) for each operation

getLogID () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return logId กลับไป

4.1.2.5.7.2 Algorithmic model (e.g., PDL) for each operation

START

Return logId

END

4.1.2.5.8.1 Processing narrative (PSPEC) for each operation

getEmail () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return email กลับไป

4.1.2.5.8.2 Algorithmic model (e.g., PDL) for each operation

START

Return email

END

4.1.2.5.9.1 Processing narrative (PSPEC) for each operation

getLatitude () : เป็น Public Method ที่ไม่ต้องรับ Parameter เมื่อถูกเรียกใช้จะ Return ค่า latitude

4.1.2.5.9.2 Algorithmic model (e.g., PDL) for each operation

START

Return latitude

END

4.1.2.5.10.1 Processing narrative (PSPEC) for each operation

getLongitude () : เป็น Public Method ที่ไม่ต้องรับ Parameter เมื่อถูกเรียกใช้จะ Return ค่า longitude

4.1.2.5.10.2 Algorithmic model (e.g., PDL) for each operation

START

Return longitude

END

4.1.2.5.11.1 Processing narrative (PSPEC) for each operation

getID (): เป็น Public Method ที่ไม่รับParameter เพื่อ Return id

4.1.2.5.11.2 Algorithmic model (e.g., PDL) for each operation

START

Return id

END

**Class Passenger**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

Passenger (string name, string phone, string email, string typeOfCreditCard, string numberCreditCard, string securityCode) : เป็น Pubilc Method ที่มี Parameter เป็น name, phone, email, typeOfCreditCard, numberCreditCard และ securityCode ใช้กับระบบลงทะเบียนการใช้บริการของผู้โดยสารในครั้งแรก โดยหลังจากบันทึกข้อมูลทั้งหมดเก็บไว้ใน Attribute ส่วน logID, latitude, longitude จะบันทึกไว้เป็นค่า null แล้วทำการเรียกใช้ Method addListPassenger ใน Class AllPassengerList ใน Server Component เพื่อทำการเก็บ Passenger ใหม่เข้าระบบ เมื่อทำงานสำเร็จค่าจะ Return เป็น true หากไม่สำเร็จจะเเจ้งเตือนและ Return false

4.1.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

บันทึก name, phone, email, typeOfCreditCard, numberCreditCard และ securityCode ลง Attribute

บันทึก null ลง logID, latitude, longitude

เรียกใช้ Method addListPassenger ใน Class AllPassengerList

หากสร้างและบันทึกสำเร็จ

Return true

หากไม่สำเร็จ

แจ้งเตือนให้ทำใหม่

Return false

END

4.1.2.5.2.1 Processing narrative (PSPEC) for each operation

showMapAndTaxi () : เป็น Pubilc Method ที่ไม่จำเป็นต้องมี Parameter ใช้เมื่อผู้โดยสารเข้าสู่หน้าเรียกใช้บริการ จะปรากฏเป็นรูปตำแหน่งของผู้โดยสารและผู้ขับในบริเวณนั้น โดยเริ่มจากการเรียกใช้ Method setLocation เพื่อบันทึกค่าที่อยู่ในปัจจุบันไปใน Attribute latitude, longitude และส่งค่าไปยัง Method mapAndTaxi ใน Class PositionAndRoute เพื่อใช้ในการแสดงผลแผนที่และรถบริเวณโดยรอบ

4.1.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

เรียกใช้ Method setLocation

ส่งค่า latitude, longitude ไปยัง Method mapAndTaxi ใน Class PositionAndRoute

แสดงผลแผนที่และรถบริเวณโดยรอบ

END

4.1.2.5.3.1 Processing narrative (PSPEC) for each operation

setDestination (float latitudeDestination, float longitudeDestination) : เป็น Pubilc Method มี Parameter เป็น latitude, longitude ของตำแหน่งจุดหมายปลายทางที่ผู้โดยสารเป็นผู้กำหนด เมื่อทำการเรียกใช้จำเป็นการติดต่อกับ Server Component โดยจะส่ง latitude, longitude ใน Passenger เพื่อเป็น latitudeSource, longitudeSource และ latitude, longitude ของจุดหมายปลายทาง เพื่อเป็น latitudeDestiantion, longitudeDestination และส่ง ID ของ Passenger เพื่อเป็น passengerID ไปยัง Method addListLog ใน Class AllLogList เพื่อสร้าง log ใหม่ขึ้นมา และ return logID ใหม่เพื่อส่งมายัง setDestination ให้บันทึกค่าลง Attribute logID แล้วส่ง logID ไปยัง Method sentNotification เพื่อใช้ในการส่งการแจ้งเตือนการเรียกใช้บริการไปยังผู้ขับบริเวณโดยรอบ หากสำเร็จจะ return true หากไม่จะแจ้งเตือนและ Return false

4.1.2.5.3.2 Algorithmic model (e.g., PDL) for each operation

START

ส่ง latitude, longitude ใน Passenger และ latitude, longitude ของจุดหมายปลายทาง และ ส่ง ID ของ Passenger

ไปยัง Method addListLog ใน Class AllLogList

นำตัวแปรมารับ logID

บันทึกค่าลง logID

ส่ง logID ไปยัง Method sentNotification

หากมีค่า logID

Return true

หากไม่

แจ้งเตือนให้ทำใหม่

Return false

END

4.1.2.5.4.1 Processing narrative (PSPEC) for each operation

cancelLog () : เป็น Public Method ที่ไม่ต้องการ Parameter เป็นการยกเลิกการเรียกใช้บริการ เมื่อถูกเรียกใช้จะทำการแจ้งเตือนผู้โดยสารว่าต้องการหรือไม่หากผู้โดยสารกดยืนยันอีกครั้ง จะทำการส่ง logID ไปยัง Method deleteLogID ใน Class AllLogList ของ Server Component เพื่อทำการลบ logID ที่เรียกออกจากระบบ หากสำเร็จจะ Return true หากไม่จะแจ้งเตือนและ Return false

4.1.2.5.4.2 Algorithmic model (e.g., PDL) for each operation

START

ส่ง logID ไปยัง Method deleteLogID ใน Class AllLogList

ค้นหา logIDใน logList

หากค้นหาพบ

ลบ logID นั้นออกจาก list

Return true

หากไม่

แจ้งเตือน

Return false

END

4.1.2.5.5.1 Processing narrative (PSPEC) for each operation

showDriverInfo () : เป็น Public Method ที่ไม่ต้องการ Parameter จะแสดงข้อมูลของผู้ขับออกมา โดยจะนำ logID ส่งไปยัง Method getDriverID ใน Class AllLogList ของ Server Component เพื่อทำการเรียก driverID ที่ให้บริการในครั้งนัั้น แล้วส่ง driverID นั้นไปยัง Method getDriverInfo ใน Class AllDriverList ของ Server Component เพื่อ แสดงเป็นข้อมูลของผู้ขับออกมา

4.1.2.5.5.2 Algorithmic model (e.g., PDL) for each operation

START

นำ logID ส่งไปยัง Method getDriverID ใน Class AllLogList

นำตัวแปรมารับ driverID

ส่ง driverID ไปยัง Method getDriverInfo ใน Class AllDriverList

แสดงข้อมูลของผู้ขับ

END

4.1.2.5.6.1 Processing narrative (PSPEC) for each operation

showPassengerInfo () : เป็น Public Method ที่ไม่ต้องการ Parameter เพื่อแสดงข้อมูลของผู้ใช้บริการออกมา โดยจะแสดง name, phone, email, typeOfCreditCard และ numberCreditCard

4.1.2.5.6.2 Algorithmic model (e.g., PDL) for each operation

START

แสดง name, phone, email, typeOfCreditCard และ numberCreditCard

END

4.1.2.5.7.1 Processing narrative (PSPEC) for each operation

editPassengerInfo () : เป็น Public Method ที่รับ Parameter เป็น name, phone, email, typeOfCreditCard, numberCreditCard และ securityCode ใช้ในการแก้ไขข้อมูลของผู้โดยสาร โดยจะทำการบันทึกค่าทั้งหมดลง Attribute หากสำเร็จจะ Return true หากไม่ จะเเจ้งเตือนและ Return false

4.1.2.5.7.2 Algorithmic model (e.g., PDL) for each operation

START

รับ name, phone, email, typeOfCreditCard, numberCreditCard และ securityCode จากผู้ใช้

บันทึกค่าทั้งหมดลง Attribute

หากแก้ไขสำเร็จ

Return true

หากไม่

เเจ้งเตือน

Return false

END

4.1.2.5.8.1 Processing narrative (PSPEC) for each operation

getTypeCard () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ return typeOfCreditCard กลับไป

4.1.2.5.8.2 Algorithmic model (e.g., PDL) for each operation

START

Return typeOfCreditCard

END

4.1.2.5.9.1 Processing narrative (PSPEC) for each operation

getNumCard () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ return numberCreditCard กลับไป

4.1.2.5.9.2 Algorithmic model (e.g., PDL) for each operation

START

Return numberCreditCard

END

4.1.2.5.10.1 Processing narrative (PSPEC) for each operation

getSucCode () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return securityCode กลับไป

4.1.2.5.10.2 Algorithmic model (e.g., PDL) for each operation

START

Return securityCode

END

4.1.2.5.11.1 Processing narrative (PSPEC) for each operation

showLogInfoPassenger () : เป็น Public Method ที่ไม่ต้องการ Parameter จะถูกเรียกใช้หลังจากผู้ใช้บริการทำการเลือกจุดหมายปลายทางเรียบร้อย โดยจะแสดงหน้า Waiting หากยังไม่มีผู้ขับตอบรับและ แสดง แผนที่ระหว่างผู้ขับกับผู้โดยสารพร้อมทั้งข้อมูลของผู้ขับหากถูกตอบรับแล้ว เมื่อถูกเรียกใช้จะทำการส่ง logID ไปยัง Method getLogInfo ใน Class AllLogList ของ Server Component เพื่อใช้ในการแสดงผลข้อมูลของ LogID นั้นๆ โดยค่าที่ได้กลับมาจะถูกนำมาแสดงผ่านแอปพลิเคชั่น ถ้าค่า statusOfLog ยังเป็น False จะแสดงหน้า Waiting และทำการวนเรียกใช้จนกว่าจะเป็น True แล้วจะเรียกใช้ showRouteTaxitoPassenger และ showDriverInfo เฉพาะในส่วนของ name และ plate ให้แก่ผู้ใช้

4.1.2.5.11.2 Algorithmic model (e.g., PDL) for each operation

START

ส่ง logID ไปยัง Method getLogInfo ใน Class AllLogList

หาก statusOfLog ยังเป็น false

แสดงหน้า waiting

กลับไปตรวจสอบใหม่

หากเป็น true

เรียกใช้ showRouteTaxitoPassenger และ showDriverInfo

แสดง name และ plate

END

**Class GPS**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

getNowLatitude () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return latitude ของตำแหน่งปัจจุบัน กลับไป โดยจะทำการเรียกใช้งาน Android เพื่อรับตำแหน่งที่อยู่เข้ามา

4.1.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

ตรวจสอบ latitude ณ ตำแหน่งปัจจุบัน

บันทึกค่าลง latitude

Return latitude

END

4.1.2.5.2.1 Processing narrative (PSPEC) for each operation

getNowLongitude () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ return longitude ของตำแหน่งปัจจุบัน กลับไป โดยจะทำการเรียกใช้งาน Android เพื่อรับตำแหน่งที่อยู่เข้ามา

4.1.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

ตรวจสอบ longitude ณ ตำแหน่งปัจจุบัน

บันทึกค่าลง longitude

Return longitude

END

**Class Route**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

showRoute (float latitudeStrat, float longitudeStart, float latitudeEnd, float longitudeEnd) : เป็น Public Method ที่มี Parameter เป็น latitudeStart, longitudeStart, latitudeEnd, longitudeEnd เพื่อใช้ในการแสดงแผนที่และเส้นทางที่เริ่มต้นไปถึงสิ้นสุด โดยจะบันทึกค่าทั้งหมดลง Attribute และเรียกใช้ Google map เพื่อสร้างเส้นทางและแผนที่แล้วนำมาแสดงผล

4.1.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

รับ latitudeStart, longitudeStart, latitudeEnd, longitudeEnd

บันทึกลงใน Attribute

แสดง แผนที่และเส้นทาง

END

**Class PositionAndRoute**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

mapAndTaxi (float latitude, float longitude) : เป็น Public Method มี Parameter เป็น latitude, longitude ของผู้โดยสาร แล้วทำการส่งค่าไปยัง Method match ใน Class AllDriverList เพื่อใช้ในการค้นหารถที่อยู่ในบริเวณนั้น โดยจะ Return ค่าออกมาเป็น driverID ทั้งหมดที่อยู่ในบริเวณเดียวกับผู้โดยสาร และ บันทึกลงใน Attribute listDriver แล้วทำการส่ง driverID ใน listDriver ไปยัง Class AllDriverlist อีกครั้ง ใน Method getLatitudeDriverID, getLogitudeDriverID เพื่อทำการหาว่า driverID นั้นมีตำแหน่งอยู่ที่ไหน แล้วแสดง แผนที่ ตำแหน่งผู้โดยสารและตำแหน่งผู้ขับทั้งหมด

4.1.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

ส่งค่า latitude, longitudeไปยัง Method match ใน Class AllDriverList

บันทึก driverID ทั้งหมดลง listDriver

วนส่ง driverID ใน listDriver ไปยัง Method getLatitudeDriverID, getLogitudeDriverID ของ Class AllDriverlist

นำตัวแปรมารับค่า

แสดง แผนที่ ตำแหน่งผู้โดยสารและตำแหน่งผู้ขับทั้งหมด

END

4.1.2.5.2.1 Processing narrative (PSPEC) for each operation

sentNotification (int logId) : เป็น Public Method มี Parameter เป็น logID เพื่อทำการส่ง logID นั้นไปยังผู้ขับทุกคนที่อยู่ใน listDriver โดยจะวนส่ง DriverID และ logID ไปยัง Method sentNotice ใน Class AllDriverList ของ Driver Component เพื่อแจ้งเตือนให้ผู้ขับคนนั้นๆ ทราบถึงการเรียกใช้บริการ

4.1.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

วนส่ง logID และ DriverID ที่อยู่ใน listDriver ไปยัง Method sentNotice ใน Class AllDriverList

END

4.1.3. COMPONENT 1 TEST POINTS LIST AND DESCRIPTION

|  |  |
| --- | --- |
| **Test** | **Description** |
| Passenger | สามารถสร้าง Passenger ได้อย่างถูกต้อง |
| showMapAndTaxi | สามารถแสดง แผนที่และรถในบริเวณใกล้เคียงกับผู้ใช้ได้อย่างถูกต้อง |
| setDestination | สามารถสร้าง TravelLog เพิ่ม และ มีค่า logId ได้ |
| cancelLog | สามารถลบ logId นั้นออกจาก list ได้ |
| showDriverInfo | สามารถแสดงข้อมูลผู้ขับตาม driverID ได้อย่างถูกต้อง |
| editPassengerInfo | สามารถแก้ไขข้อมูลของผู้ใช้ได้ |
| showLogInfoPassenger | สามารถแสดงหน้าข้อมูลของผู้ใช้ได้ |

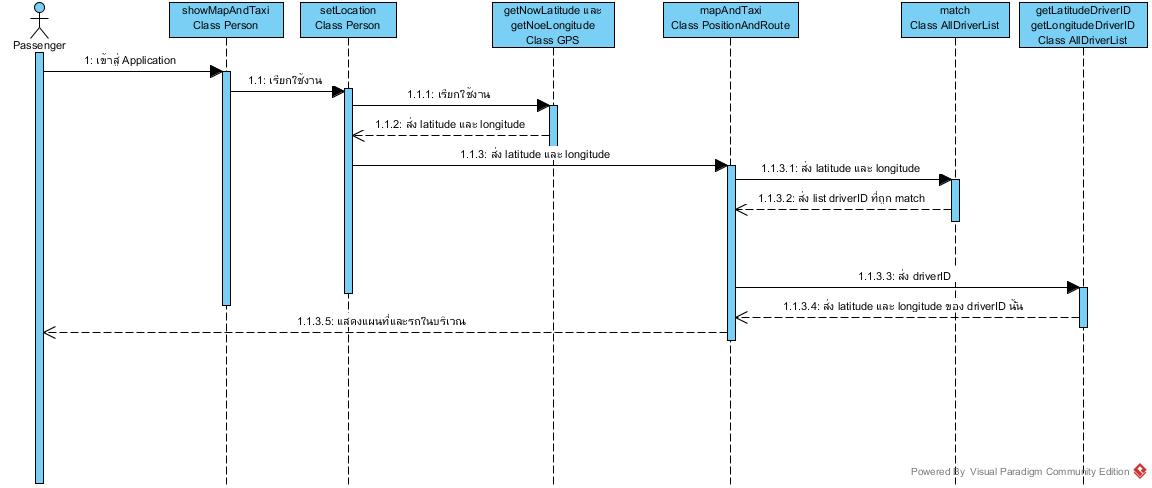
**Class Passenger**

**Class Person**

|  |  |
| --- | --- |
| **Test** | **Description** |
| setLocation | สามารถบันทึกข้อมูลตำแหน่งปัจจุบันของผู้ใช้ได้อย่างถูกต้อง |
| showRouteTaxitoPassenger | สามารถแสดงเส้นทางระหว่างรถโดยสารถึงผู้โดยสารได้ |
| showRouteTaxitoDestination | สามารถแสดงเส้นทางระหว่างรถโดยสารถึงจุดหมายปลายทางได้ |
| showSummary | สามารถแสดงข้อมูลสรุปจาก logId ได้อย่างถูกต้อง |

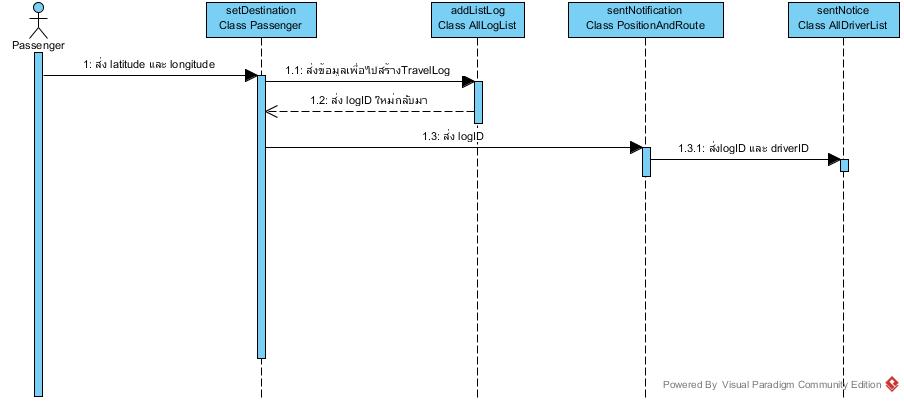
4.1.4. COMPONENT 1 DYNAMIC BEHAVIOR

**Passenger: การเรียกใช้บริการรถโดยสารไปยังจุดหมายปลายทาง**



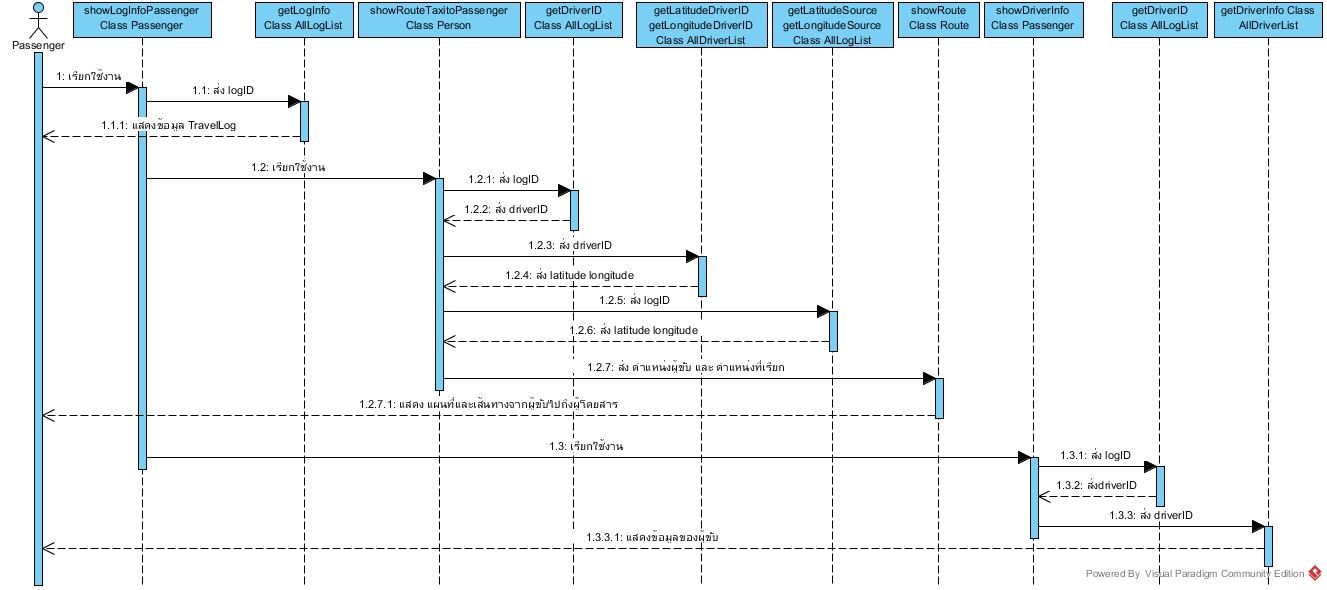
เริ่มต้นที่ Passenger เริ่มใช้งานแอปพลิเคชั่น จะเป็นการเรียกใช้ Method showMapAndTaxi เพื่อแสดงที่อยู่ปัจจุบันและรถโดยสารบริเวณโดยรอบ โดยจะเรียกใช้ Method setLocation ต่อเพื่อหาค่า latitude และ longitude ณ ตำแหน่งปัจจุบัน ผ่านการใช้งาน Method getNowLatitdue และ getNowLongitude หลังจากนั้นจะส่ง latitude และ longitude ที่ได้ไปยัง Method match เพื่อใช้ตรวจสอบว่ามีรถโดยสารคันใดอยู่ในบริเวณผู้โดยสารบ้าง และส่ง driverID ทั้งหมดกลับออกมา หลังจากนั้นจึงวนส่งค่า driverID ทั้งหมด ไปยัง Method getLatitudeDriverID และ getLongitudeDriverID เพื่อหาตำแหน่งของรถโดยสารแต่ละคัน และแสดงตำแหน่งผู้ใช้บริการและผู้ให้บริการทั้งหมดมาให้แก่ผู้ใช้บริการ

**Passenger: การเรียกใช้บริการรถโดยสารไปยังจุดหมายปลายทาง**



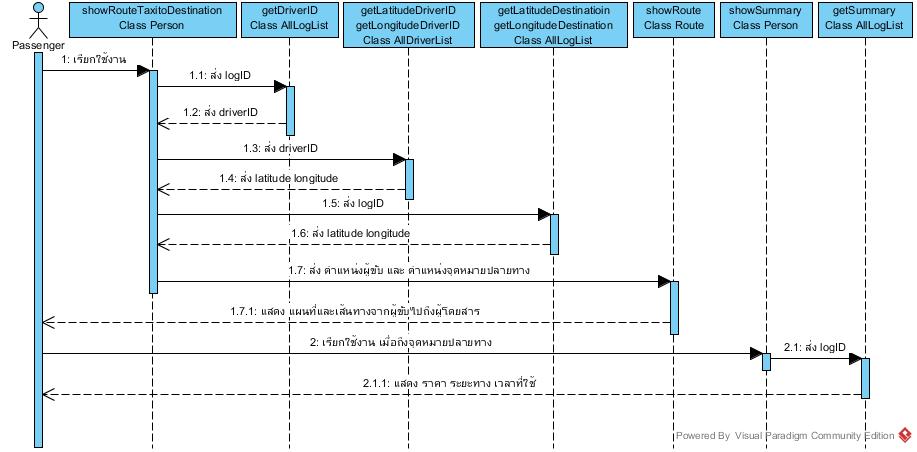
หลังจากที่ผู้โดยสารเห็นตำแหน่งและรถโดยสารของตนในบริเวณโดยรอบแล้ว ผู้โดยสารจะส่งค่า latitude และ longitude ของตำแหน่งจุดหมายปลายทาง ไปยัง Method setDestiantion เพื่อสร้าง TravelLog ใหม่ โดย จะส่งข้อมูล passengerID, latitudeSource, longitudeSource, latitudeDestination และ longitudeDestination ไปยัง Method addListLog หลังจากนั้นจะได้ logID ใหม่กลับมา แล้วส่ง logID นั้นไปยัง Method sentNotification เพื่อใช้ในการส่งการแจ้งเตือนไปยังผู้ขับแต่ละคน โดยจะส่ง logID และ driverID ของทุกอันที่อยู่ใน driverList ที่ได้จากการ match ไปยัง sentNotice เพื่อใช้ในการแจ้งเตือนต่อไป

**Passenger: การเรียกใช้บริการรถโดยสารไปยังจุดหมายปลายทาง**



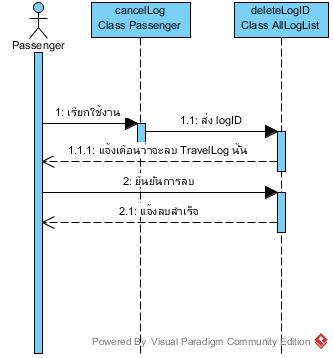
หลังจากระบุจุดหมายปลายทางแล้ว จะแสดงหน้า ข้อมูลของ TravelLog โดยการเรียกใช้ Method showLogInfoPassenger จะส่ง logID ไปยัง getLogInfo เพื่อใช้ในการแสดงข้อมูล TravelLog อันนั้นออกมา หากยังไม่มีผู้ขับตอบรับการให้บริการจะแสดงหน้า Wating และเมื่อมีผู้ขับตอบรับแล้ว จะเรียกใช้งาน Method showRouteTaxitoPassenger เพื่อแสดงเส้นทางจากผู้ขับไปยังผู้โดยสาร โดยจะส่ง logID ไป Method getDiverID เพื่อหา driverID ที่ตอบรับ และส่ง driverID นั้นไป Method getLatitudeDriverID และ getLongitudeDriverID เพื่อหาตำแหน่งของรถโดยสารคันนั้น และส่ง logID ไปยัง getLatitudeSource และ getLongitudeSource เพื่อหาตำแหน่งที่เรียกใช้บริการออกมาและส่งตำแหน่งทั้งหมดไปยัง Method showRoute เพื่อใช้ในการแสดงแผนที่และเส้นทาง หากหลังจากนั้นผู้โดยสารต้องการทราบข้อมูลของผู้ขับผู้โดยสารจะเรียกใช้ Method showDriverInfo เพื่อแสดงข้อมูลของผู้ขับ โดยส่ง logID ไปยัง Method getDriverID เพื่อนำ driverID ไป Method getDriverInfo เพื่อแสดงข้อมูลของผู้ขับคนนั้นออกมา

**Passenger: การเรียกใช้บริการรถโดยสารไปยังจุดหมายปลายทาง**



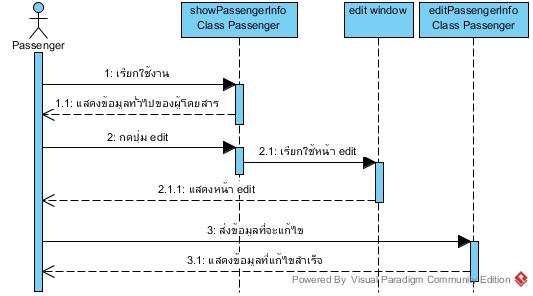
หลังจากที่ผู้ขับมารับผู้โดยสารแล้วจะเรียก Method showRouteTaxitoDestination เพื่อแสดงเส้นทางจากผู้ขับไปยังจุดหมายปลายทาง โดยจะส่ง logID ไป Method getDiverID เพื่อหา driverID ที่ตอบรับ และส่ง driverID นั้นไป Method getLatitudeDriverID และ getLongitudeDriverID เพื่อหาตำแหน่งของรถโดยสารคันนั้น และส่ง logID ไปยัง getLatitudeDestination และ getLongitudeDestination เพื่อหาตำแหน่งจุดหมายปลายทางออกมาและส่งตำแหน่งทั้งหมดไปยัง Method showRoute เพื่อใช้ในการแสดงแผนที่และเส้นทาง เมื่อถึงจุดหมายปลายทาง จะเรียกใช้ Method showSummary เพื่อสรุปผลของการเดินทางออกมา โดยส่ง logID ไปยัง Method getSummary เพื่อแสดง ระยะทาง ราคา และ เวลาที่ใช้ในหน้าสรุปการเดินทาง

**Passenger: ยกเลิกการใช้บริการ**



เมื่อผู้ใช้บริการเรียกใช้บริการแล้ว ต้องการยกเลิกการใช้บริการ จะเรียกใช้งาน Method cancelLog เพื่อลบ TravelLog นั้นออก โดยจะส่ง logID นั้งไปยัง Method deleteLogID เพื่อแจ้งความต้องการที่จะลบ logID นั้น จะมีการแจ้งเตือนก่อนการลบ หากผู้ใช้บริการยืนยันการยกเลิกการใช้บริการ จะทำการลบ logID นั้นออกจากระบบ และ แจ้งผู้ใช้บริการเมื่อสำเร็จ

**Passenger: แก้ไขข้อมูลผู้ใช้บริการ**



เมื่อผู้ใช้บริการต้องการแก้ไขข้อมูลของผู้ใช้บริการ จะเริ่มจาก การเรียกใช้ Method showPassengerInfo เพื่อแสดงข้อมูลของผู้ใช้บริการนั้นออกมา หลังจากนั้นผู้ใช้บริการจะกดปุ่ม Edit ระบบจะแสดงหน้า Edit ออกมาโดยมีค่าเดิมอยู่ หากผู้ใช้ต้องการแก้ไขจึงค่อยแก้ไขข้อมูลในส่วนนั้น และส่งข้อมูลทั้งหมดไปยัง Method editPassengerInfo เพื่อใช้ในการแก้ไข และแจ้งแก่ ผู้ใช้บริการเมื่อแก้ไขสำเร็จ

4.2. DESCRIPTION COMPONENT 2: Driver Component

4.2.1. PROCESSING NARRATIVE (PSPEC) FOR COMPONENT 2

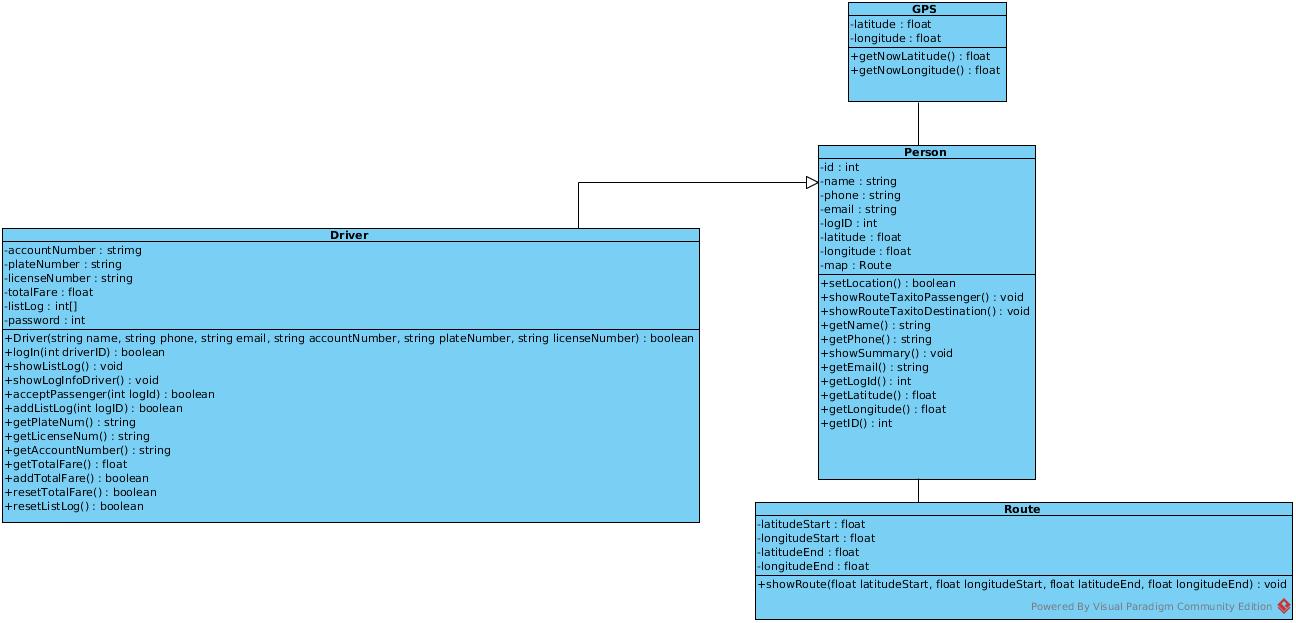
Driver Component สร้างขึ้นเพื่อใช้จัดการในแอปพลิเคชั่นส่วนของผู้ขับ และ ข้อมูลของผู้ขับแต่ละบุคคล ประกอบไปด้วย

Class Driver เป็น class ที่สืบทอดมาจาก Class Person และเป็น Class หลักของ Driver Component เพื่อใช้ในการเก็บข้อมูลของผู้ขับ และติดต่อกับ Server Component ในการจัดการแอปพลิเคชั่น ประกอบไปด้วย Attribute plateNumber, licenseNumber เพื่อใช้เป็นข้อมูลพื้นฐานของผู้ขับ Password เพื่อใช้ในส่วนของการยืนยันตัวตน accountNumber, totalFare เพื่อใช้สนับสนุนระบบการโอนค่าโดยสารอัตโนมัติเป็นรายสัปดาห์ ส่วน attribute logID, listLog, latitude, longitude, map เพื่อใช้สนับสนุนการทำงานของแอปพลิเคชั่น โดย logID ใช้ในการเก็บ logID ที่กำลังให้บริการอยู่ในตอนนั้น listLogID ใช้ในการเก็บ logID ทั้งหมดของผู้โดยสารที่ถูก Match ด้วย latitude, longitude ใช้เพื่อเก็บตำแหน่งของผู้ขับ ในขณะนั้น และ Password ใช้เพื่อการ Login เข้าสู่ระบบการใช้งาน และมี Method Driver, logIn, showListLog, showLogInfoDriver, acceptPassenger, addListLog, getPlateNum, getAccountNumber, getLicenseNum, addToTotalFare, resetTotalFare, getTotalFare, setLocationและ resetListLog

ส่วน Class Person, Class GPS และ Class Route ได้กล่าวไว้ใน Passenger Component ก่อนหน้านี้แล้ว

4.2.2. COMPONENT 2 PROCESSING DETAIL

4.2.2.1. Design Class hierarchy for component 2



4.2.2.2. Restrictions/limitations for component 2

- การใช้งาน Driver Component จะถูกใช้งานได้เมื่อใช้ผ่านแอปพลิเคชั่นของ Driver เท่านั้น

- จำเป็นจะต้องเป็นสมัครสมาชิกก่อนจึงจะสามารถใช้แอปพลิเคชั่นได้

- การสมัครสมาชิกต้องผ่านผู้ให้บริการแอปพลิเคชั่นเท่านั้น

- จำเป็นต้องใช้งานผ่านอินเตอร์เน็ต จึงสามารถเชื่อมต่อกับ Component อื่นได้

- ต้องมีเลขที่บัญชีเพื่อใช้ในการโอนเงินค่าโดยสารอัตโนมัติเป็นรายสัปดาห์

4.2.2.3. Performance issues for component 2

- ไม่สามารถเชื่อมต่อกับ Passenger Component ที่เป็นผู้โดยสารได้โดยตรง

- หากไม่สมัครสมาชิกก่อนจะไม่สามารถใช้แอปพลิเคชั่นได้

- หากไม่ติดต่อผู้ให้บริการเพื่อยืนยันตัวตนก่อนจะไม่สามารถสมัครเป็นสมาชิกได้

- หากไม่สามารถใช้บริการอินเตอร์เน็ตจะไม่สามารถใช้บริการแอปพลิเคชั่นได้

- ไม่ได้รับค่าโดยสารทันทีเมื่อการให้บริการสิ้นสุด ต้องผ่านบัญชีธนาคารเท่านั้น

4.2.2.4. Design constraints for component 2

- หากกดรับผู้โดยสารแล้วผู้ขับไม่สามารถยกเลิกการให้บริการได้ด้วยตัวเอง ต้องผ่านผู้โดยสารเท่านั้น

4.2.2.5. Processing detail for each operation of component 2

**Class Driver**

4.2.2.5.1.1 Processing narrative (PSPEC) for each operation

Driver (string name, string phone, string email, string accountNumber, string plateNumber, string licenseNumber) : เป็น Pubilc Method ที่มี Parameter เป็น name, phone, email, accountNumber, plateNumber และ licenseNumber ใช้กับระบบลงทะเบียนการใช้บริการของผู้ขับในครั้งแรก โดยหลังจากบันทึกข้อมูลทั้งหมดเก็บไว้ใน Attribute ส่วน logID, latitude, longitudeจะบันทึกไว้เป็นค่า null แล้วทำการเรียกใช้ Method addListDriver ใน Class AllDriverList ใน Server Component เพื่อทำการเก็บ Driver ใหม่เข้าระบบ เมื่อทำงานสำเร็จค่าจะ Return เป็น True หากไม่สำเร็จจะเเจ้งเตือนและ Return False

4.2.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

รับ name, phone, email, accountNumber, plateNumber และ licenseNumber

บันทึกข้อมูลทั้งหมดเก็บไว้ใน Attribute

บันทึก null ลง logID, latitude, longitude

เรียกใช้ Method addListDriver ใน Class AllDriverList หากสร้างและบันทึกสำเร็จ

Return true

หากไม่สำเร็จ

เเจ้งเตือน

Return false

END

4.2.2.5.2.1 Processing narrative (PSPEC) for each operation

logIn (int driverId, int password) : เป็น Public Method ที่มี Parameter เป็น driverID และ password เพื่อใช้ในการ เข้าสู่ระบบ โดยระบบจะ ทำการเปรียบเทียบกับ ข้อมูลหากเหมือนกันจะ Return True หาไม่เหมือนจะแจ้งเตือนและ Return False

4.2.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

รับ driverID และ password

เปรียบเทียบกับ ข้อมูล

หากเหมือนกัน

เรียกใช้ Method showListLog

Return true

หากไม่

แจ้งเตือน

Return false

END

4.2.2.5.3.1 Processing narrative (PSPEC) for each operation

showListLog () : เป็น Public Method ที่ไม่ต้องการ Parameter เพื่อแสดงข้อมูลผู้เรียกใช้บริการที่เรียกใช้บริการโดยจะส่ง logID ที่อยู่ใน listLog ไปยัง Method getLatitudeSource, getLongitudeSource, getLatidueDestination, getLongitudeDestination และ getPassengerID ใน Class AllLogList ของ Server Component เพื่อแสดงข้อมูลจุดรับส่งผู้โดยสารออกมาและนำ passengerID ส่งไปยัง Method getNameAndPhoneP ใน Class AllPassengerList เพื่อใช้ในการแสดงผล ชื่อและเบอร์ของ ผู้เรียกใช้บริการของ Log นั้นๆ

4.2.2.5.3.2 Algorithmic model (e.g., PDL) for each operation

START

วนส่ง logID ที่อยู่ใน listLog ไปยัง Method getLatitudeSource, getLongitudeSource, getLatidueDestination, getLongitudeDestination และ getPassengerID ใน Class AllLogList

แสดงข้อมูลจุดรับส่งผู้โดยสาร

วนส่ง passengerID แต่ละ listไปยัง Method getNameAndPhoneP ใน Class AllPassengerList

แสดงชื่อและเบอร์ ของ ผู้เรียกใช้บริการ

END

4.2.2.5.4.1 Processing narrative (PSPEC) for each operation

showLogInfoDriver () : เป็น Public Method ที่ไม่ต้องการ Parameter จะถูกเรียกใช้หลังจากผู้ขับทำการเลือกผู้โดยสารเพื่อแสดง แผนที่ระหว่างผู้ขับกับผู้โดยสารพร้อมทั้งข้อมูลของผู้โดยสาร โดยเมื่อถูกเรียกใช้จะทำการส่ง logID ไปยัง Method getLogInfo ใน Class AllLogList ของ Server Component เพื่อใช้ในการแสดงผลข้อมูลของ LogID นั้นๆ โดยค่าที่ได้กลับมาจะถูกนำมาแสดงผ่านแอปพลิเคชั่น ถ้า ค่า statusOfLog ยังเป็น false จะแสดง Waiting และหากผู้ขับเลือกรับจะเป็น Accepted นอกจากนั้นจะนำpassengerID ที่ได้ ส่งไปยัง Method getNameAndPhoneP ใน Class AllPassengerList เพื่อใช้ในการแสดงผลชื่อและเบอร์โทรศัพท์ของ ผู้เรียกใช้บริการของ Log นั้นๆ

4.2.2.5.4.2 Algorithmic model (e.g., PDL) for each operation

START

ส่ง logID ไปยัง Method getLogInfo ใน Class AllLogList

หาก ค่า statusOfLog เป็น false

แสดง Waiting

หากไม่

แสดง Accepted

นำ passengerID ส่งไปยัง Method getNameAndPhoneP ใน Class AllPassengerList

แสดง ชื่อและเบอร์โทรศัพท์ของ ผู้เรียกใช้บริการ

END

4.2.2.5.5.1 Processing narrative (PSPEC) for each operation

acceptPassenger (int logId) : เป็น Public Method ที่มี Parameter เป็น LogID เพื่อใช้แสดงแผนที่การเดินทางจากจุดที่รับไปยังจุดหมายปลายทาง พร้อมกับข้อมูลการเรียกใช้บริการและข้อมูลผู้เรียกใช้บริการ โดย จะส่ง driverID และ logIDไปยัง Method setDriverID และ changeStatus ใน Class AllLogList ของ Server Component เพื่อบันทึก driverID ลงใน LogID นั้นๆ และเปลี่ยน Status เป็น True จากนั้นเรียกใช้ Method showRouteTaxitoDestination เพื่อแสดงเส้นทางและระยะทางระหว่าง ผู้ขับและจุดหมายปลายทาง เมื่อทำงานสำเร็จค่าจะ Return เป็น True หากไม่สำเร็จจะเเจ้งเตือนและ Return False

4.2.2.5.5.2 Algorithmic model (e.g., PDL) for each operation

START

ส่ง driverID และ logIDไปยัง Method setDriverID และ changeStatus ใน Class AllLogList

บันทึก driverID ลงใน LogID นั้นๆ

เปลี่ยน Status เป็น True

เรียกใช้ Method showRouteTaxitoDestination

แสดงเส้นทางและระยะทางระหว่าง ผู้ขับและจุดหมายปลายทาง

หากสำเร็จค่า

Return true

หากไม่

เเจ้งเตือน

Return false

END

4.2.2.5.6.1 Processing narrative (PSPEC) for each operation

addListLog (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อนำไปเพิ่ม listLog บ่งบอกถึงมีผู้เรียกใช้คนใดที่เราสามารถให้บริการได้ โดย เพิ่ม logId ลงใน listLog หากเพิ่มสำเร็จ Return true หากไม่ Return false

4.2.2.5.6.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logId

เพิ่ม logID ลงใน listLog

หากเพิ่มสำเร็จ

Return true

หากไม่

Return false

END

4.2.2.5.7.1 Processing narrative (PSPEC) for each operation

getPlateNum () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return plateNum

4.2.2.5.7.2 Algorithmic model (e.g., PDL) for each operation

START

Return plateNum

END

4.2.2.5.8.1 Processing narrative (PSPEC) for each operation

getLicenseNum () : เป็น Public Method ที่ไม่รับParameter เพื่อ Return licenseNum

4.2.2.5.8.2 Algorithmic model (e.g., PDL) for each operation

START

Return licenseNum

END

4.2.2.5.9.1 Processing narrative (PSPEC) for each operation

getAccountNumber () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return accountNumber

4.2.2.5.9.2 Algorithmic model (e.g., PDL) for each operation

START

Return accountNumber

END

4.2.2.5.10.1 Processing narrative (PSPEC) for each operation

getTotalFare () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return totalFare

4.2.2.5.10.2 Algorithmic model (e.g., PDL) for each operation

START

Return totalFare

END

4.2.2.5.11.1 Processing narrative (PSPEC) for each operation

addTotalFare (float fare) : เป็น Public Method ที่รับ Parameter เป็น fareเพื่อ ใช้เพิ่มค่าโดยสารของผู้ขับ โดยจะเพิ่มเมื่อส่งผู้โดยสารถึงที่หมายเรียบร้อยแล้ว แล้วจึงนำยอดค่าโดยสารรวมในแต่ละสัปดาห์มาใช้ในการโอนเข้าบัญชีผู้ขับ Return True เมื่อสำเร็จ Return False เมื่อไม่สำเร็จ

4.2.2.5.11.2 Algorithmic model (e.g., PDL) for each operation

START

รับ fare

เพิ่ม fare ไปใน totalFare จากค่าเดิม

หากเพิ่มสำเร็จ

Return true

หากไม่

แจ้งเตือน

Return false

END

4.2.2.5.12.1 Processing narrative (PSPEC) for each operation

resetTotalFare () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ ใช้ในการตั้งค่า totalFare ให้เป็น 0 ในแต่ละสัปดาห์ Return True เมื่อสำเร็จ Return False เมื่อไม่สำเร็จ

4.2.2.5.12.2 Algorithmic model (e.g., PDL) for each operation

START

ตั้งค่า totalFare เป็น 0

หากค่า totalFare เป็น 0

Return true

หากไม่

แจ้งเตือน

Rerturn false

END

4.2.2.5.13.1 Processing narrative (PSPEC) for each operation

resetListLog () : เป็น Public Method ที่ไม่รับ Parameter เพื่อใช้ในการตั้งค่า listLog ให้ไม่มีข้อมูลในแต่ละวัน Return True เมื่อสำเร็จ Return False เมื่อไม่สำเร็จ

4.2.2.5.13.2 Algorithmic model (e.g., PDL) for each operation

START

ลบ logId ทั้งหมดออกจาก listLog

หากไม่เหลือ logId

Return true

หากเหลือ

แจ้งเตือน

Return false

END

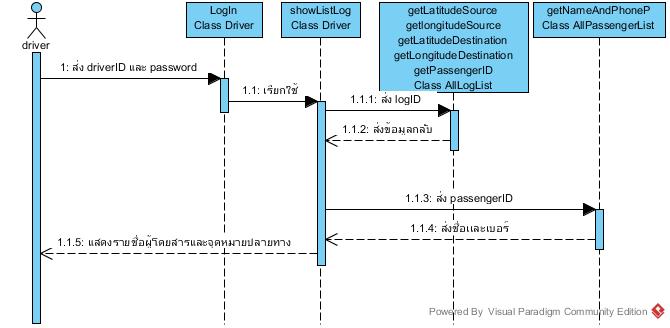
4.2.3. COMPONENT 2 TEST POINTS LIST AND DESCRIPTION

**Class Driver**

|  |  |
| --- | --- |
| **Test** | **Description** |
| Driver | สามารถสร้าง Driver ได้อย่างถูกต้อง |
| logIn | เมื่อกรอก ID และ Password ที่ถูกต้องจึงสามารถใช้บริการได้ |
| showListLog | แสดงข้อมูลของ เที่ยวการเดินทางและผู้ใช้ได้อย่างถูกต้องและครบถ้วนทุกเที่ยวการเดินทาง |
| showLogInfoDriver | แสดงข้อมูลของ เที่ยวการเดินทางของแต่ละ logId และ แสดงผล สถานะของผู้ใช้ได้อย่างถูกต้อง |
| acceptPassenger | สามารถบันทึกค่า driverID ลงใน logId ที่ถูกต้องได้ |
| addListLog | สามารถเพิ่ม logId ลง listLog ได้ |
| addTotalfare | สามารถเพิ่ม fare ใน toatlFare ได้อย่างถูกต้อง |
| resetTotalFare | สามารถตั้งค่า totalFare เป็น 0 ได้ |
| resetListLog | สามารถลบ logId ทั้งหมดจาก listLog ได้ |

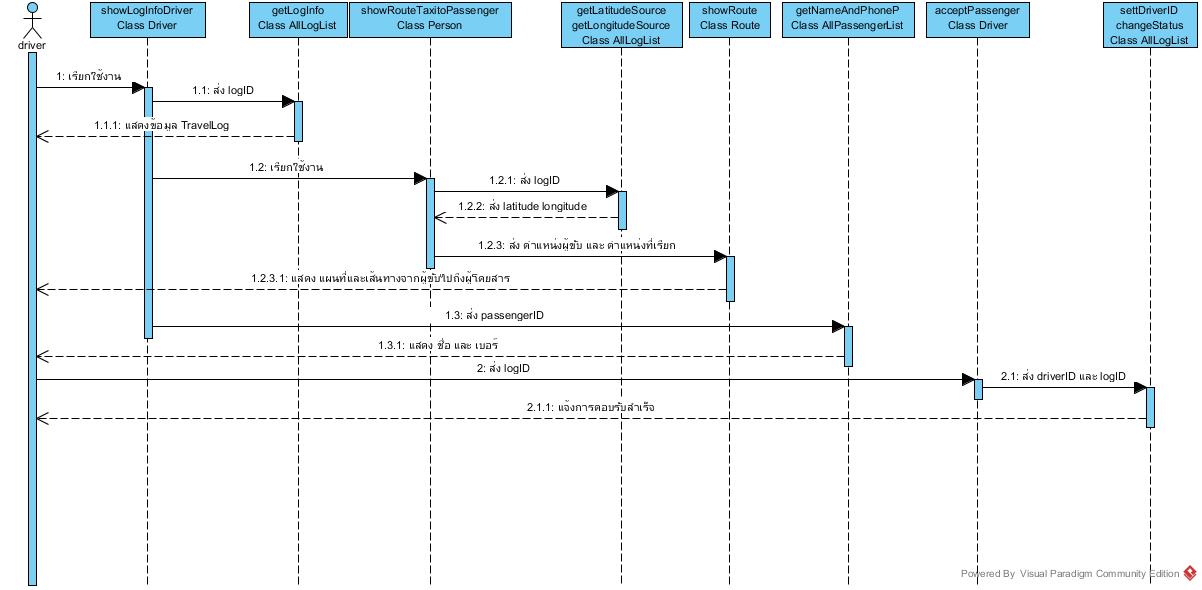
4.2.4. COMPONENT 2 DYNAMIC BEHAVIOR

**Driver: การตอบรับการเรียกใช้บริการของผู้โดยสาร**

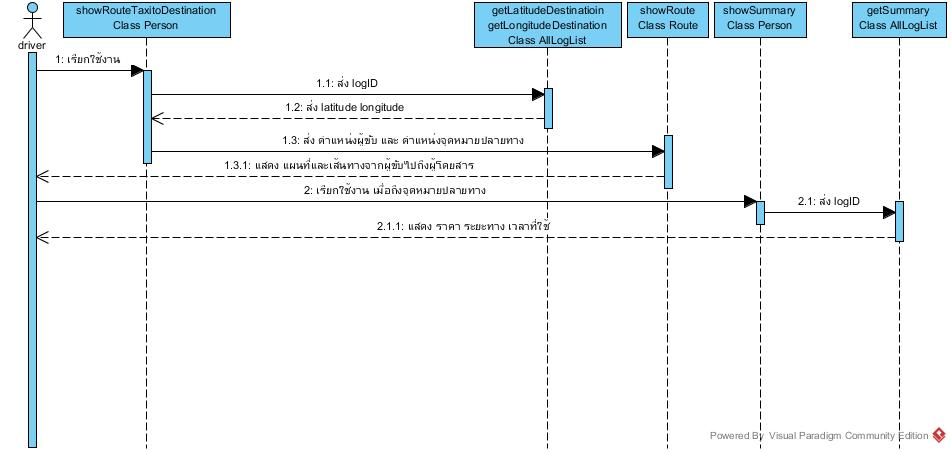


ผู้ขับใช้บริการแอปพลิเคชั่น โดยเริ่มต้นที่ ส่ง driverID และ password ไปยัง Method logIn เพื่อใช้ในการยืนยันตัวตน หากผ่านการอนุมัติ จะไปเรียกใช้ Method showListLog เพื่อแสดงข้อมูลใน listLog ทั้งหมดที่มีของผู้ขับ โดยส่ง logID ไปยัง getLatitudeSource, getLongitudeSource, getLatitudeDestination, getlongitudeDestination และ getPassengerID เพื่อส่ง passengerID ไปยัง Method getNameAndPhoneP เพื่อแสดงจุดรับผู้โดยสาร จุดหมายปลายทาง ชื่อ และ เบอร์โทรของผู้โดยสารในแต่ละ List

**Driver: การตอบรับการเรียกใช้บริการของผู้โดยสาร**



หลังจากที่ผู้ขับกดเลือก การเรียกใช้บริการใน List จะเป็นการเรียกใช้ showLogInfoDriver เพื่อแสดงข้อมูลโดยละเอียดออกมา โดยจะส่ง logID ไปยัง Method getLogInfo เพื่อแสดงข้อมูลในส่วนของ จุดรับผู้โดยสาร จุดหมายปลายทาง ราคาค่าโดยสาร และระยะทางออกมา หลังจากนั้นจะเรียกใช้งาน Method showRouteTaxitoPassenger เพื่อแสดงแผนที่ระหว่างผู้โดยสารกับผู้ขับออกมา ตามขั้นตอนที่ได้กล่าวไปแล้ว หลังจากนั้นจะเรียกใช้ Method getNameAndPhoneP เพื่อแสดงข้อมูลชื่อและเบอร์โทรศัพท์ เมื่อผู้ขับสนใจจะกดปุ่ม Accept เพื่อตอบรับการเรียกใช้บริการ โดยจะส่ง logID นั้นไปยัง Method acceptPassenger เพื่อระบุ driverID นั้นลงใน TravelLog โดยจะส่ง driverID และ logID ไปยัง Method setDriverID และ เรียกใช้ changeStatus เพื่อบันทึกค่า driverID ลงใน logID นั้น และเปลี่ยนสถานะของ statusOfPassenger จาก False เป็น True

 **Driver: การตอบรับการเรียกใช้บริการของผู้โดยสาร**

หลังจากที่ผู้ขับมารับผู้โดยสารแล้วจะเรียก Method showRouteTaxitoDestination เพื่อแสดงเส้นทางจากผู้ขับไปยังจุดหมายปลายทาง โดยจะส่ง logID ไปยัง getLatitudeDestination และ getLongitudeDestination เพื่อหาตำแหน่งจุดหมายปลายทางออกมาและส่งตำแหน่งผู้ขับและจุดหมายปลายทางไปยัง Method showRoute เพื่อใช้ในการแสดงแผนที่และเส้นทาง เมื่อถึงจุดหมายปลายทาง จะเรียกใช้ Method showSummary เพื่อสรุปผลของการเดินทางออกมา โดยส่ง logID ไปยัง Method getSummary เพื่อแสดง ระยะทาง ราคา และ เวลาที่ใช้ในหน้าสรุปการเดินทาง

4.3. DESCRIPTION COMPONENT 3: Server Component

4.3.1. PROCESSING NARRATIVE (PSPEC) FOR COMPONENT 3

Server Component สร้างขึ้นเพื่อใช้สนับสนุน การทำงานของแอปพลิเคชั่น ของผู้โดยสารและ เป็น Component ที่เป็นทางเชื่อมระหว่าง Passenger Component กับ Driver Component ให้สามารถติดต่อกันได้ ประกอบไปด้วย

Class AllPassengerList ถูกสร้างขึ้นเพื่อใช้รวบรวม Passenger ทั้งหมดที่ใช้งานเพื่อสามารถจัดการข้อมูลของ Passenger แต่ละคนได้ มี Attribute listPassenger เพื่อใช้ในการเก็บ Passenger ทั้งหมดเป็น List มี Method addListPassenger, getNameAndPhoneP และ pay

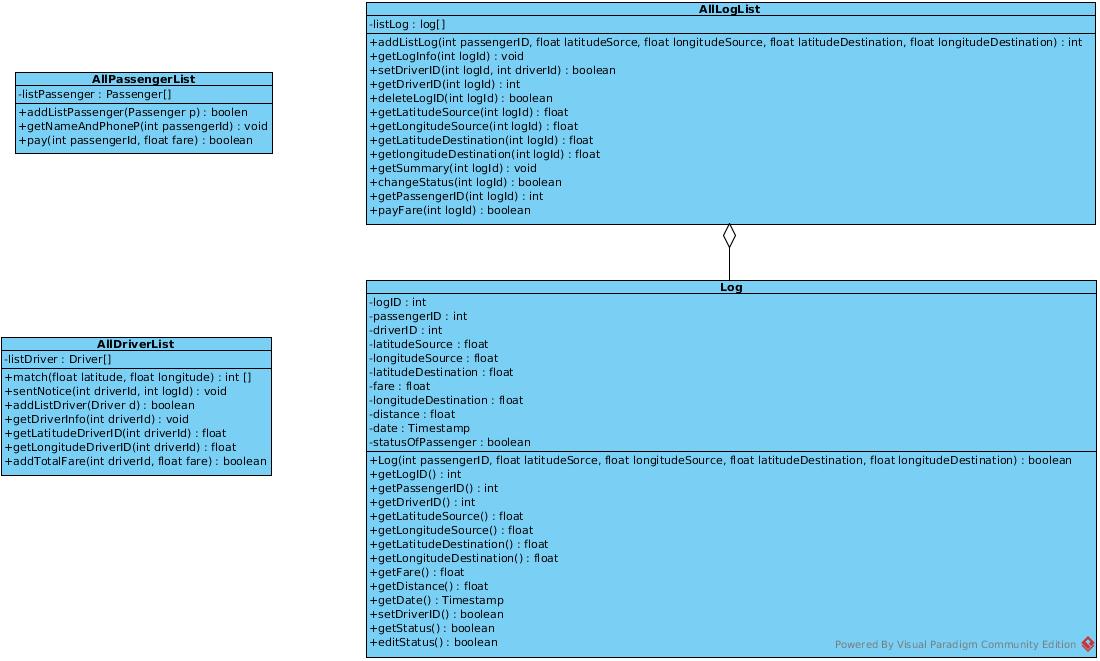
Class AllDriverrList ถูกสร้างขึ้นเพื่อใช้รวบรวม Driver ทั้งหมดที่ใช้งานเเพื่อสามารถจัดการข้อมูลของ Driver แต่ละคนได้ มี Attribute listDriver เพื่อใช้ในการเก็บ Driver ทั้งหมดเป็น List มี Method match, sentNotice, payTotalFareAllDriver, addListDriver, getDriverInfo, getLatitudeDriverID, getLongitudeDriverID และ addTotalFare

Class AllLogList ถูกสร้างขึ้นเพื่อใช้รวบรวม Log หรือการใช้งานทั้งหมดไว้ เพื่อสามารถจัดการข้อมูลของแต่ละ Log มี Attribute log เพื่อใช้ในการเก็บ Log ทั้งหมดเป็น List มี Method AddListLog, getLogInfo, setDriverID, getDriverID, deleteLogID, getLatitudeSource, getLongitudeSource,getLatitudeDestination, getLongitudeDestination , getSummary, changeStatus, getPassengerID และ payFare

Class Log ถูกสร้างขึ้นเพื่อเก็บข้อมูลการใช้งานในแต่ละครั้ง มี Atttribute logID ใช้เพื่อการอ้างถึง Log นี้, passengerID ใช้เพื่อบ่งบอกผู้เรียกใช้บริการ, driverID ใช้เพื่อบ่งบอกผู้ให้บริการ, latitudeSource, longitudeSource, latitudeDestination, longitudeDestination ใช่เพื่อบอกจุดเริ่มต้นและจุดหมายปลายทาง, fare, distance, date ใช้เพื่อบอกราคา ระยะทาง และ เวลาในการโดยสาร, statusOfPassenger ใช้เพื่อบ่งบอกสถานะของผู้โดยสารว่าถูกรับจากผู้ขับคนใดแล้วหรือไม่ มี Method Log, getLogId, getPassengerID, getDriverID, getLatitudeSource, getLongitudeSource, getLatitudeDestination, getLongitudeDestination, getFare, getDistance, setDriverID, getDriverID, getStatus และ editStatus

4.3.2. COMPONENT 3 PROCESSING DETAIL

4.3.2.1. Design Class hierarchy for component 3



4.3.2.2. Restrictions/limitations for component 3

- ไม่สามารถทำงานได้ด้วยตัวเอง ต้องอาศัยการสั่งงานจาก Passenger Component และ Driver Component

- หากจะแก้ไขข้อมูลส่วนใดก็ตามใน ข้อมูลที่อยู่ภายในโดยไม่ใช่เจ้าของข้อมูลนั้น ต้องผ่านผู้ดูแลระบบเท่านั้น

4.3.2.3. Performance issues for component 3

- หากไม่มีการสั่งงาน จะไม่สามารถติดต่อกับ Component อื่นด้วยตัวเองได้

- เมื่อเกิดปัญหาขึ้น จำเป็นต้องแจ้งผ่านผู้ดูแลเท่านั้น

4.3.2.4. Design constraints for component 3

- การทำงานของ ของผู้ดูแล ระบบที่สร้างขึ้นมานี้ไม่มีการรับรอง

4.3.2.5. Processing detail for each operation of component 3

**Class AllPassengerList**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

addListPassenger (Passenger p) : เป็น Public Method ที่รับ Parameter เป็น Passenger แล้วทำการ เพิ่ม Passenger นั้น เข้าไปใน List หากสำเร็จ Return True หากไม่ Return False

4.3.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

เพิ่ม Passenger ไปใน passengerList

หากสำเร็จ

Return true

หากไม่

Return false

END

4.1.2.5.2.1 Processing narrative (PSPEC) for each operation

getNameAndPhoneP (int passengerId) : เป็น Public Method ที่รับ Parameter เป็น passengerID เพื่อใช้ในการแสดง ชื่อและเบอร์โทรศัพท์ของ passengerID นั้นๆ โดยจะเรียกใช้ Method getName, getPhone ใน Class Passenger ของ Passenger Componet ส่งส่งไปให้ Driver ที่เรียกใช้บริการ

4.3.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

รับ passengerID

เรียกใช้ Method getName, getPhone ใน Class Passenger

ส่งชื่อและเบอร์ ของผู้ใช้ไปยัง Driver ที่เรียกใช้

END

4.1.2.5.3.1 Processing narrative (PSPEC) for each operation

pay (int passengerId, float fare) : เป็น Public Method ที่รับ Parameter เป็น passengerId และ fare เพื่อใช้ในการชำระค่าโดยสารของpassenger ในครั้งนั้น โดยจะเรียกใช้ Method gerTypeCard, getNumCard และ getSecCode ของ passenderId นั้นๆ และทำการตัดเงินผ่านบัตรเครดิต เมื่อสำเร็จ Return True หากไม่ Return False และเเจ้งเตือน

4.3.2.5.3.2 Algorithmic model (e.g., PDL) for each operation

START

รับ passengerId และ fare

เรียกใช้ Method gerTypeCard, getNumCard และ getSecCode ของ passenderId นั้น

ทำการตัดเงินผ่านบัตรเครดิต

หากสำเร็จ

Return true

หากไม่

เเจ้งเตือน

Return false

END

**Class AllDriverList**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

match (float latitude, float longitude) : เป็น Public Method ที่รับ Parameter เป็น latitude, longitude ของผู้โดยสารที่เรียกใช้บริการ โดย จะทำการคำนวณตำแหน่งของผู้โดยสารและผู้ขับทุกคันที่มีอยู่ใน listDriver โดยจะมีระยะห่างไม่เกิน 5 กิโลเมตรจากตำแหน่งของผู้โดยสาร return เป็น List driverID ที่ตรงกับเงื่อนไข

4.3.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

รับ latitude, longitude

คำนวณตำแหน่งของผู้โดยสารและผู้ขับทุกคันที่มีอยู่ใน listDriver ที่มีระยะห่างไม่เกิน 5 กิโลเมตรจากตำแหน่งของผู้โดยสาร

Return List ของdriverID ที่ตรงกับเงื่อนไข

END

4.1.2.5.2.1 Processing narrative (PSPEC) for each operation

sentNotice (int driverId, int logId) : เป็น Public Method ที่รับ Parameter เป็น driverID และ logID เพื่อทำการเพิ่ม listLog ของ driverID นั้นๆ โดยจะทำการส่ง logID ไปยัง Method addListLog ใน Class Driver ของ driverID นั้นๆ

4.3.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

รับ driverID และ logID

ส่ง logID ไปยัง Method addListLog ใน Class Driver ของ driverID

END

4.1.2.5.3.1 Processing narrative (PSPEC) for each operation

addListDriver (Driver d) : เป็น Public Method ที่รับ Parameter เป็น Driver แล้วทำการ เพิ่ม Driver นั้น เข้าไปใน List หากสำเร็จ Return True หากไม่ Return False

4.3.2.5.3.2 Algorithmic model (e.g., PDL) for each operation

START

รับ Driver

เพิ่ม Driver เข้าไปใน List

หากสำเร็จ

Return true

หากไม่

Return false

END

4.1.2.5.4.1 Processing narrative (PSPEC) for each operation

getDriverInfo (int driverId) : เป็น Public Method ที่รับ Parameter เป็น driverID เพื่อใช้ในการแสดงข้อมูลของผู้ขับโดยจะเรียกใช้ getName, getPlateNum, getLicenseNum และ getPhone ของ driverID นั้นๆ และส่งไปยัง Method showDriverInfo ของ Passenger ที่เรียกใช้

4.3.2.5.4.2 Algorithmic model (e.g., PDL) for each operation

START

รับ driverID

เรียกใช้ getName, getPlateNum, getLicenseNum และ getPhone ของ driverID นั้น

ส่งข้อมูลทั้งหมดไปยัง Method showDriverInfo ของ Passenger ที่เรียกใช้

END

4.1.2.5.5.1 Processing narrative (PSPEC) for each operation

getLatitudeDriverID (int driverId) : เป็น Public Method ที่รับ Parameter เป็น driverID เพื่อใช้ในการหา Latitude ของ ผู้ขับคันนั้นๆ โดยจะเรียกใช้ getLatitude ของ driverID นั้นๆ Return เป็นค่า latitude

4.3.2.5.5.2 Algorithmic model (e.g., PDL) for each operation

START

รับ driverID

หา Latitude ของ driverIDนั้น

Return latitude

END

4.1.2.5.6.1 Processing narrative (PSPEC) for each operation

getLogitudeDriverID (int driverId,) : เป็น Public Method ที่รับ Parameter เป็น driverID เพื่อใช้ในการหา Longitude ของ ผู้ขับคันนั้นๆ โดยจะเรียกใช้ getLongitude ของ driverID นั้นๆ return เป็นค่า longitude

4.3.2.5.6.2 Algorithmic model (e.g., PDL) for each operation

START

รับ driverID

หา longitude ของ driverIDนั้น

Return longitude

END

4.1.2.5.7.1 Processing narrative (PSPEC) for each operation

addTotalFare (int driverId, float fare) : เป็น Public Method ที่รับ Parameter เป็น driverID และ fare ใช้สำหรับการเพิ่ม totalFare ของ driverId นั้น โดยส่งค่า fare ไปยัง Method addTotalfare ของ driverId นั้น หากสำเร็จ Return Trueและไม่ Return False และแจ้งเตือน

4.3.2.5.7.2 Algorithmic model (e.g., PDL) for each operation

START

รับ driverID และ fare

ส่ง fare ไปยัง Method addTotalfare ของ driverId นั้น

หากสำเร็จ

Return true

หากไม่

แจ้งเตือน

Return false

END

**Class AllLogList**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

addListLog (int passengerID. float latitudeSource, float longitudeSource, float latitudeDestination, float longitudeDestination) : เป็น Pubilc Method ที่รับ paremeter เป็น passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination จาก Method setDestination ใน Class Passenger ของ Passenger Component โดยจะทำการ สร้าง Logใหม่ขึ้นมาในlistLog โดยการเรียกใช้ Log ของ Class TravelLog ในการสร้าง โดยการส่ง passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination ต่อไป และ return logID ของ Log ใหม่ที่สร้างขึ้น กลับคืนไป

4.3.2.5.1.2 Algorithmic model (e.g., PDL) for each operation

START

รับ passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination

สร้าง TravelLogใหม่

ส่ง passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination

เพิ่ม TravelLog ใน listLog

Return logID

END

4.1.2.5.2.1 Processing narrative (PSPEC) for each operation

getLogInfo (int logId) : เป็น Public Method ที่มี Parameter เป็น logID ใช้เพื่อการแสดงผล logID นั้นๆ โดยจะทำการเรียก getStatus, getLatitudeSource, getLongitudeSource, getLatitudeDestination, getLongitudeDestination, getDistance, getFare ของ logID นั้นๆ แล้วส่งค่าทั้งหมดไปยัง showLogInfo ของ Passenger และ Driver ที่เรียกใช้

4.3.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

เรียกใช้ getStatus, getLatitudeSource, getLongitudeSource, getLatitudeDestination, getLongitudeDestination, getDistance, getFare ของ logID นั้น

ส่งค่าทั้งหมดไปยัง showLogInfo ของ Passenger และ Driver ที่เรียกใช้

END

4.1.2.5.3.1 Processing narrative (PSPEC) for each operation

setDriverID (int logId, int driverId) : เป็น Public Method ที่มี Parameter เป็น logID และ driverID เพื่อบันทึก driverID ลงใน logID นั้นๆ หากสำเร็จ Return True หากไม่ Return False

4.3.2.5.3.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID และ driverID

บันทึก driverID ลงใน logID นั้น

หากสำเร็จ

Return true

หากไม่

Return false

END

4.1.2.5.4.1 Processing narrative (PSPEC) for each operation

getDriverID (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อ return driverID ของ Log นั้นๆ

4.3.2.5.4.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

Return driverID ของ Log นั้น

END

4.1.2.5.5.1 Processing narrative (PSPEC) for each operation

deleteLogID (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อลบ logID นั้นออกจาก listLogหากสำเร็จ Return True หากไม่ Return False

4.3.2.5.5.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

ลบ logID นั้นออกจาก listLog

หากสำเร็จ

Return true

หากไม่

Return false

END

4.1.2.5.6.1 Processing narrative (PSPEC) for each operation

getLatitudeSource : เป็น Public Method ที่มี Parameter เป็น logID เพื่อ return latitudeSource ของ Log นั้นๆ

4.3.2.5.6.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

Return latitudeSource ของ Log นั้น

END

4.1.2.5.7.1 Processing narrative (PSPEC) for each operation

getLongitudeSource (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อ Return longitudeSource ของ Log นั้นๆ

4.3.2.5.7.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

Return longitudeSource ของ Log นั้น

END

4.1.2.5.8.1 Processing narrative (PSPEC) for each operation

getLatitudeDestination (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อ Return latitudeDestination ของ Log นั้นๆ

4.3.2.5.8.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

Return latitudeDestination ของ Log นั้น

END

4.1.2.5.9.1 Processing narrative (PSPEC) for each operation

getLongitudeDestination (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อ return longitudeDestianation ของ Log นั้นๆ

4.3.2.5.9.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

Return longitudeDestination ของ Log นั้น

END

4.1.2.5.10.1 Processing narrative (PSPEC) for each operation

getSummary (int logId) : เป็น Public Method ที่รับ Parameter เป็น logId เพื่อแสดง ราคา ระยะทาง และเวลาที่ใช้ โดยจะทำการเรียกใช้ Method getDistance, getFare และ getDate ใน Class TravelLog และนำเวลาปัจจุบันไปคำนวณเทียบกับ Date เพื่อแสดงผลออกมาเป็นเวลาที่ใช้ และแสดงค่าผลที่ได้ทั้งหมด

4.3.2.5.10.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logIDd

เรียกใช้ Method getDistance, getFare และ getDate ใน Class TravelLog ของ logIdoนั้น

นำเวลาปัจจุบันไปคำนวณเทียบกับ Date

แสดงผลออกมาเป็นเวลาที่ใช้ ระยะทาง และ ราคาค่าโดยสาร

END

4.1.2.5.11.1 Processing narrative (PSPEC) for each operation

changeStatus (int logId) : เป็น Public Method ที่รับ Parameter เป็น logID แล้วทำการเรียกใช้ editStatus ของ logID นั้นๆเพื่อเปลี่ยน Status ของ logID นั้นๆจาก false เป็น true เพื่อแสดงเป็นสถานะว่ามีคนขับตอบรับแล้ว หากสำเร็จ Return True หากไม่ Return False

4.3.2.5.11.2 Algorithmic model (e.g., PDL) for each operation

START

รับ logID

เรียกใช้ editStatus ของ logID นั้น

หากสำเร็จ

Return true

หากไม่

Return false

END

4.1.2.5.12.1 Processing narrative (PSPEC) for each operation

getPassengerID (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อ Return passengerID ของ Log นั้นๆ

4.3.2.5.12.2 Algorithmic model (e.g., PDL) for each operation

START

รับlogID

Return passengerID ของ Log นั้น

END

4.1.2.5.13.1 Processing narrative (PSPEC) for each operation

payFare (int logId) : เป็น Public Method ที่มี Parameter เป็น logID เพื่อใช้ในการชำระค่าโดยสาร โดยจะนำ logID เพื่อใช้ในการ หา driverId, passengerId และfare จาก logId นั้นๆ แล้วนำ driverId และ fare ส่งไปยัง Method addTotalFare ของ Class AllDriverList เพื่อเพิ่มค่าโดยสารที่ผู้ขับได้รับ และ ส่ง passengerId และ fare ไปยัง Method pay ใน Class AllPassengerList เพื่อใช้ในการชำระค่าโดยสารผ่านบัตรเครดิต หากสำเร็จ Return True หากไม่ Return False

4.3.2.5.13.2 Algorithmic model (e.g., PDL) for each operation

START

รับlogID

หา driverId, passengerId และfare จาก logId นั้นๆ

นำ driverId และ fare ส่งไปยัง Method addTotalFare ของ Class AllDriverList

ส่ง passengerId และ fare ไปยัง Method pay ใน Class AllPassengerList

หากสำเร็จ

Return true

หากไม่

Return false

END

**Class TravelLog**

4.1.2.5.1.1 Processing narrative (PSPEC) for each operation

Log (int passengerID, float latitudeSource, float longitudeSource, float latitudeDestination, float longitudeDestination) : เป็น Pubilc Method ที่รับ paremeter เป็น passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination จาก Method addListLog ใน Class AllLogList โดยจะทำการ สร้าง Logใหม่ขึ้นมา ทำการสร้าง logID ใหม่ซึ่งต่อจากของเดิม คำนวณระยะทาง(distance) ระหว่างจุดโดยสารและจุดหมายปลายทาง คำนวณค่าโดยสาร(fare) จากระยะทาง และบันทึกค่า logID, passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination, distance, fare, date(จาก timeStamp ในระบบ) ลงใน Attribute สำหรับ driverID ให้เป็น null ไว้ และ statusOfPassenger ให้เป็น false ซึ่งหมายความว่ายังไม่มีผู้ขับตอบรับ และ Return True ถ้าสร้างสำเร็จ และ False หากไม่สำเร็จ

4.3.2.5.2.2 Algorithmic model (e.g., PDL) for each operation

START

รับ passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination

สร้าง Logใหม่ขึ้นมา ต่อจากของเดิม

คำนวณระยะทาง(distance)ระหว่างจุดโดยสารและจุดหมายปลายทาง

คำนวณค่าโดยสาร(fare)จากระยะทาง

บันทึกค่า logID, passengerID, latitudeSorce, longitudeSource, latitudeDestination, longitudeDestination, distance, fare, date(จาก timeStamp ในระบบ) ลงใน Attribute

บันทึก null ลง driverID

บันทึก False ลง statusOfPassenger

หากสร้างสำเร็จ

Return true

หากไม่

Return false

END

4.1.2.5.3.1 Processing narrative (PSPEC) for each operation

getLogID () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ return logID

4.3.2.5.3.2 Algorithmic model (e.g., PDL) for each operation

START

Return logID

END

4.1.2.5.4.1 Processing narrative (PSPEC) for each operation

getPassengerID () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return passengerID

4.3.2.5.4.2 Algorithmic model (e.g., PDL) for each operation

START

Return passengerID

END

4.1.2.5.5.1 Processing narrative (PSPEC) for each operation

getDriverID () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return driverID

4.3.2.5.5.2 Algorithmic model (e.g., PDL) for each operation

START

Return driverID

END

4.1.2.5.6.1 Processing narrative (PSPEC) for each operation

getLatitudeSource () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return latitudeSource

4.3.2.5.6.2 Algorithmic model (e.g., PDL) for each operation

START

Return latitudeSource

END

4.1.2.5.7.1 Processing narrative (PSPEC) for each operation

getLongitudeSource () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return longitudeSource

4.3.2.5.7.2 Algorithmic model (e.g., PDL) for each operation

START

Return longitudeSource

END

4.1.2.5.8.1 Processing narrative (PSPEC) for each operation

getLatitudeDestination () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return latitudeDestination

4.3.2.5.8.2 Algorithmic model (e.g., PDL) for each operation

START

Return latitudeDestination

END

4.1.2.5.9.1 Processing narrative (PSPEC) for each operation

getLongitudeDestination () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return longitudeDestination

4.3.2.5.9.2 Algorithmic model (e.g., PDL) for each operation

START

Return longitudeDestination

END

4.1.2.5.10.1 Processing narrative (PSPEC) for each operation

getFare () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return fare

4.3.2.5.10.2 Algorithmic model (e.g., PDL) for each operation

START

Return fare

END

4.1.2.5.11.1 Processing narrative (PSPEC) for each operation

getDistance () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return distance

4.3.2.5.11.2 Algorithmic model (e.g., PDL) for each operation

START

Return distance

END

4.1.2.5.12.1 Processing narrative (PSPEC) for each operation

getDate () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return date

4.3.2.5.12.2 Algorithmic model (e.g., PDL) for each operation

START

Return date

END

4.1.2.5.13.1 Processing narrative (PSPEC) for each operation

setDriverID (int driverID) : เป็น Public Method รับ Parameter เป็น driverID เพื่อ บันทึก driverID ของผู้ขับที่ตอบรับลงใน logID นั้นๆ หากสำเร็จ Return True และไม่ Return False

4.3.2.5.13.2 Algorithmic model (e.g., PDL) for each operation

START

รับ driverID

บันทึก driverID ลงใน Attribute

หากสำเร็จ

Return trues

หากไม่

Return false

END

4.1.2.5.14.1 Processing narrative (PSPEC) for each operation

getStatus () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ Return statusOfPassenger

4.3.2.5.14.2 Algorithmic model (e.g., PDL) for each operation

START

Return statusOfPassenger

END

4.1.2.5.15.1 Processing narrative (PSPEC) for each operation

editStatus () : เป็น Public Method ที่ไม่รับ Parameter เพื่อ เปลี่ยน statusOfPassenger ของ logID นั้นๆจาก false เป็น true หากสำเร็จ Return True และไม่ Return False

4.3.2.5.15.2 Algorithmic model (e.g., PDL) for each operation

START

เปลี่ยน statusOfPassenger จาก False เป็น True

หากสำเร็จ

Return true

หากไม่

Return false

END

4.3.3. COMPONENT 3 TEST POINTS LIST AND DESCRIPTION

**Class AllPassengerList**

|  |  |
| --- | --- |
| **Test** | **Description** |
| addListPassenger | สามารถเพิ่ม Passenger ใน passengerList ได้ |
| getNumAndPhoneP | สามารถแสดง ชื่อ และ เบอร์ ของ ผู้โดยสารได้อย่างถูกต้อง |
| pay | สามารถจ่ายค่าโดยสารผ่านบัตรเครดิตได้ |

**Class AllDriverList**

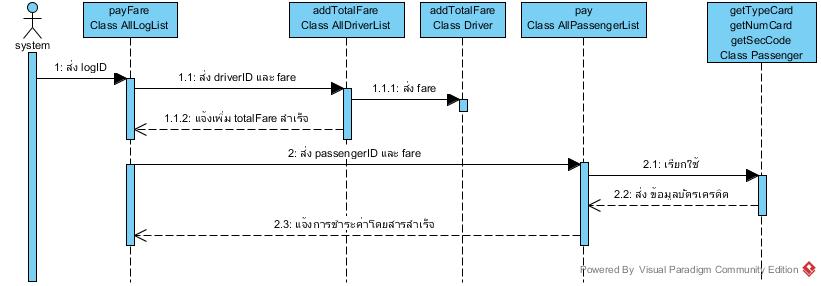
|  |  |
| --- | --- |
| **Test** | **Description** |
| match | สามารถค้นหารถโดยสารในระยะใกล้เคียงได้ |
| sentNotice | สามารถเพิ่ม listLog ของผู้ขับแต่ละคนได้ |
| addListDriver | สามารถเพิ่ม Driver ใน driverList ได้ |
| getDriverInfo | สามารถแสดงข้อมูลของผู้ขับได้อย่างถูกต้อง |
| getLatitudeDriverID | สามารถแสดง latitudeของผู้ขับได้อย่างถูกต้อง |
| getLongitudeDriverID | สามารถแสดง longitude ของผู้ขับได้อย่างถูกต้อง |
| addTotalFare | สามารถเพิ่ม fare ลงใน totalFare ของผู้ขับ ได้อย่างถูกต้อง |

**Class AllLogList**

|  |  |
| --- | --- |
| **Test** | **Description** |
| addListLog | สามารถเพิ่ม TravelLog ใน logList ได้ |
| getLogInfo | สามารถแสดงข้อมูลของเที่ยวการเดินทางได้อย่างถูกต้อง |
| setDriverID | สามารถเพิ่ม driverID ใน TravelLog ได้อย่างถูกต้อง |
| getDriverID | สามารถแสดง driverID ได้อย่างถูกต้อง |
| deleteLogID | สามารถลบ logID ที่ต้องการได้ |
| getLatitudeSource | สามารถแสดง latitudeSourceได้อย่างถูกต้อง |
| getLongitudeSource | สามารถแสดง longitudeSource ได้อย่างถูกต้อง |
| getLatitudeDestination | สามารถแสดง latitudeDestiantion ได้อย่างถูกต้อง |
| getLongitudeDestination | สามารถแสดง longitudeDestination ได้อย่างถูกต้อง |
| getSummary | สามารถแสดงข้อมูล ราคา เวลาที่ใช้ และ ระยะทาง ได้อย่างถูกต้อง |
| changeStatus | สามารถเปลี่ยน statusOfPassenger จาก false เป็น true ได้ |
| getPassengerID | สามารถแสดง passengerID ได้อย่างถูกต้อง |
| payFare | สามารถส่งค่าไปยัง Method addTotalFare และ pay ได้ |

4.3.4. COMPONENT 3 DYNAMIC BEHAVIOR

**Server: ชำระค่าโดยสาร**



หลังจากแสดง หน้าสรุปการเดินทางแล้ว ระบบจะทำการตัดค่าบริการอัตโนมัติ โดยจะส่ง logID ไปยัง Method payFare เพื่อใช้ชำระค่าโดยสารและเพิ่มค่าจ้างของผู้ขับในรอบการขับนั้น โดยจะส่ง driverID และ fare ไปยัง Method addTotalFare เพื่อเพิ่มค่าโดยสารของผู้ขับคนนั้น โดยทำการส่ง fare ไปยัง Method addTotalFare และจะส่ง passengerID และ fare ไปยัง Method pay เพื่อชำระค่าโดยสารผ่านบัตรเครดิต โดยจะเรียกใช้ getTypeCard, getNumCard และ getSecCode ของผู้โดยสารคนนั้นๆ เพื่อนำค่าของบัตรเครดิตมาชำระค่าบริการ และแจ้งเตือนเมื่อชำระสำเร็จ

**5. USER INTERFACE DESIGN**

5.1. USER INTERFACE DESIGN RULES

ในการออกแบบ User Interface จะต้องมีความละเอียดและรอบคอบในการออกแบบ ทั้งนี้ เพื่อเป็นการอำนวยความสะดวกในการใช้งานของผู้ใช้ และแสดงถึงประสิทธิภาพการทำงานของโปรแกรม เพื่อคุณภาพการใช้งานระบบที่ดี ซึ่งระบบที่ดีควรจะมี Interface Design ที่ดีด้วย ซึ่งแอปพลิเคชั่น Mali Taxi และ Mali Driver มีหลักในการออกแบบที่จะช่วยนำทางให้ออกแบบไปในทางที่ดีและมีความสัมพันธ์กัน ดังนี้

1. ควรมีระดับการทำงานที่สอดคล้องกัน ลำดับไหนควรมาก่อนหรือหลัง มีการสร้าง ไอคอน เพื่อสื่อความหมายในการเข้าถึงฟังก์ชั่น ที่มีการเรียกใช้บ่อย เพื่อความสะดวกในการใช้ของผู้ใช้งาน
2. มีการจัดวางแบบที่เรียบง่ายและ ผู้ใช้สามารถเข้าใจได้ง่าย ต้องไม่มีข้อความ หรือ ไอคอนหรือ กราฟฟิกเยอะจนเกินความจำเป็น มีการใช้โทนสี และการจัดวางไอคอน ในการช่วยออกแบบ
3. แต่ละหน้า ควรเป็นโทนสีที่คล้ายกัน เพื่อความง่ายต่อการใช้งานและ และไม่สับสนจนคิดว่าเป็นคนละระบบหรือ โปรแกรมกัน นำเสนอสิ่งที่ผู้ใช้ต้องการหรือเข้าถึงบ่อยก่อน
4. ออกแบบให้เป็นมาตรฐานทั่วไป ทั้งรูปไอคอน การจัดวาง หรือสัญลักษณ์ต่าง ๆ ให้เป็นสากล ที่คนสามารถรู้และเข้าใจง่ายได้โดยง่าย

5.2. COMPONENTS AND DEVELOPMENT TOOLS USED

เครื่องมือที่ใช้ในการพัฒนาโปรแกรมมีดังนี้

1. Adobe Photoshop ใช้สำหรับการออกแบบหน้า User Interface และ โลโก้ต่างๆ
2. Balsamiq Mockups ใช้สำหรับออกแบบหน้า User Interface
3. Adobe Effect ใช้สำหรับการแสดงผลภาพเคลื่อนไหวหรือภาพกราฟฟิกต่างๆ

5.3. SCREEN IMAGES AND DESCRIPTION

แอปพลิเคชั่น ของ Mali Taxi จะแบ่งเป็น 2 แอปพลิเคชั่น โดยแบ่งเป็น 2 ผู้ใช้งาน ได้แก่ ผู้ใช้บริการ (Passenger) มีชื่อแอปพลิเคชั่นว่า “Mali Taxi” และ ผู้ให้บริการ (Driver) มีชื่อแอปพลิเคชั่นว่า “Mali Driver”

5.3.1 แอปพลิเคชั่น Mali Taxi ของ ผู้ใช้บริการ (Passenger)

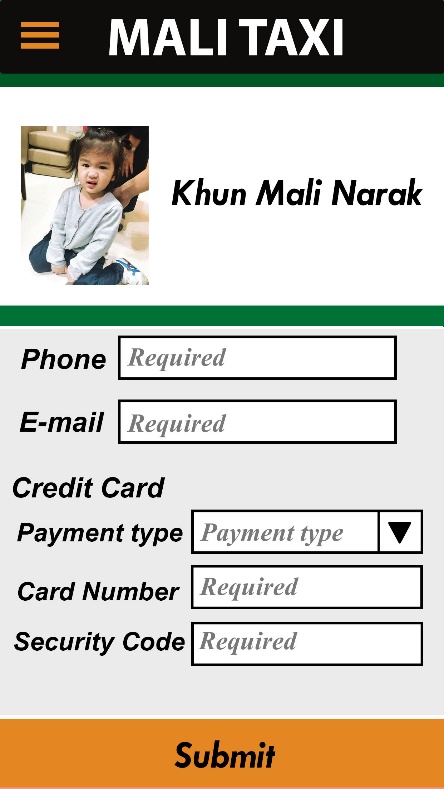
5.3.1.1 ส่วนของ Login ของผู้ใช้บริการ (Passenger)



รูป 5.3.1.1 Interface ของ Mali Taxi หน้า Login

หน้าล็อกอิน จะมีปุ่ม “Login via Facebook” เมื่อกดจะเป็นการล็อกอินผ่าน Facebook เพื่อให้ผู้ใช้งานเชื่อมต่อกับระบบผ่าน Facebook

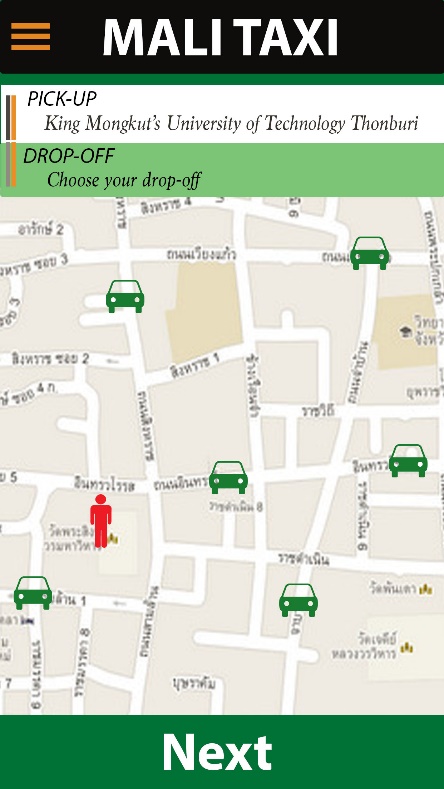
5.3.1.2 ส่วนของ Register ของผู้ใช้งาน (Passenger)



รูป 5.3.1.2 Interface ของ Mali Taxi หน้า Register

หน้าสมัครสมาชิก จะมีรูปโปรไฟล์ ซึ่งมาจาก Facebook รวมถึงชื่อของผู้ใช้งานด้วย ส่วนข้อมูลที่ต้องกรอกเพิ่มได้แก่ เบอร์โทรศัพท์ , อีเมล์ ,ข้อมูลส่วนของบัตรเครดิตที่ใช้สำหรับ ชำระค่าบริการแท็กซี่ โดยจำเป็นต้องเลือกประเภทของบัตร เช่น Visa, MasterCard, Amex เป็นต้น ซึ่งจะต้องกรอกเลขบัตรเครดิต และ Security Code ของบัตรเครดิตด้วย และกดส่งข้อมูล (Submit) ซึ่งหน้าของการจะสมัครจะขึ้นเพียงครั้งเดียว คือครั้งแรกของการใช้แอปพลิเคชั่น ครั้งต่อ ๆ ไป จะข้ามไปยังหน้าเรียกใช้แท็กซี่เลย

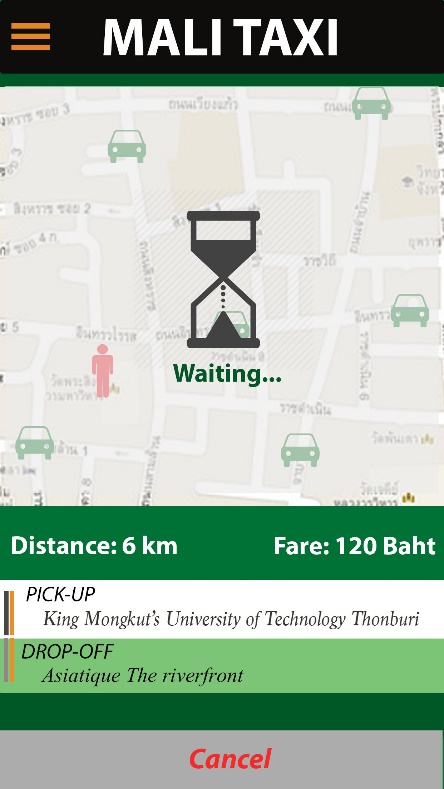
5.3.1.3 ส่วนของ การเรียกใช้งานแท็กซี่ ของผู้ใช้งาน (Passenger)



รูป 5.3.1.3 Interface ของ Mali Taxi หน้าการเรียกใช้งานแท็กซี่

ส่วนของหน้าการเรียกใช้บริการแท็กซี่ จะเป็นส่วนของผู้ใช้งาน(ผู้โดยสาร) โดยช่อง Pick-up หมายถึง จุดที่ผู้ใช้บริการต้องการให้ไปแท็กซี่ (Driver) ไปรับ และช่อง Drop-off หมายถึงจุดหมายปลายทางที่ผู้ใช้บริการต้องการจะไป รูปไอคอนรูปคน(สีแดง) บอกถึงตำแหน่งปัจจุบันของผู้ใช้งาน และรูปรถแท๊กซี่ (สีเขียว) เพื่อบอกผู้ใช้งานว่า ในบริเวณโดยรอบของผู้โดยสาร มีแท็กซี่ให้บริการ กี่คัน เมื่อทำการเลือกจุดหมายปลายทางแล้ว ให้ทำการกด Next ซึ่งจะเป็นการเรียกใช้บริการแท็กซี่

5.3.1.4 ส่วนของ การรอผู้ให้บริการมารับ ของผู้ใช้งาน (Passenger)



รูป 5.3.1.4 Interface ของ Mali Taxi หน้าการรอผู้ให้บริการมารับ

หน้ารอผู้ให้บริการ(แท็กซี่)มารับ เป็นส่วนของผู้ใช้งาน(ผู้โดยสาร) ในส่วนบนจะเป็น แผนที่ ซึ่ง ณ ขณะนั้น สถานะจะเป็น Waiting Taxi จะเป็นการบอกถึงว่ารอแท็กซี่คันใดจะมาเป็นผู้ให้บริการแก่ผู้ใช้ มีการคำนวณระยะทาง และราคาค่าโดยสาร รวมถึงแสดง จุดเริ่มต้น (Pick-Up) ของผู้โดยสาร, จุดหมายปลายทาง (Drop-Off) และ จะมีปุ่ม Cancel เพื่อยกเลิกการใช้บริการ หรือ ผู้ใช้ต้องการเปลี่ยนจุดหมายปลายทาง เมื่อกดจะมี Warning ขึ้นมาเตือน

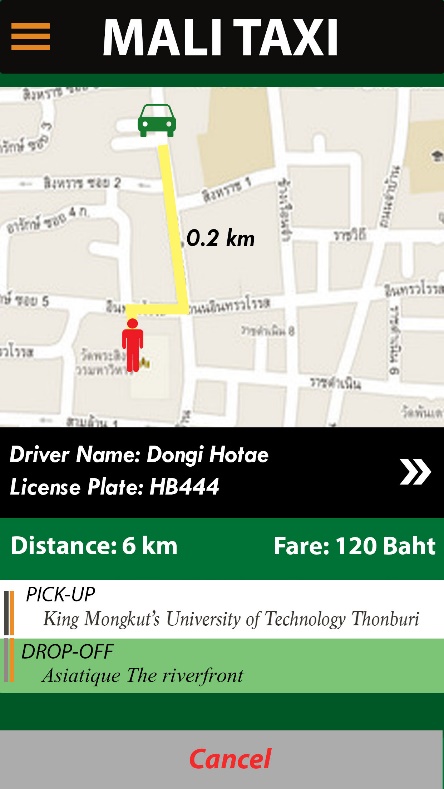
5.3.1.5 ส่วนของ Warning เมื่อทำการยกเลิกใช้บริการ ของผู้ใช้งาน (Passenger)



รูป 5.3.1.5 Interface ของ Mali Taxi หน้า Warning เมื่อทำการยกเลิกใช้บริการ

ส่วนหน้า Warning หลังจากผู้ใช้บริการ กด Cancel (ยกเลิกการใช้บริการ) เป็นการป้องกันการผิดพลาด ซึ่งผู้ใช้บริการอาจจะไม่ได้ตั้งใจไปกด โดยจะมีการเตือนว่า “Do you want to cancel?” จะมีปุ่มให้เลือก Yes หมายถึงต้องการยกเลิกการใช้บริการนี้ และ No หมายถึงไม่ต้องการยกเลิกใช้บริการนี้

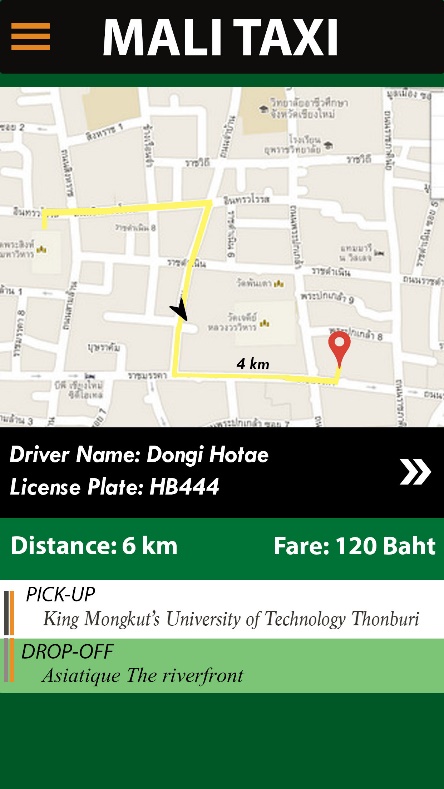
5.3.1.6 ส่วนของ การแสดงรายละเอียดผู้ให้บริการที่มารับ ของผู้ใช้งาน (Passenger)



รูป 5.3.1.6 Interface ของ Mali Taxi หน้า การแสดงรายละเอียดผู้ให้บริการที่มารับ

หน้าแสดงรายละเอียดผู้ให้บริการ(แท็กซี่) ที่จะมารับ คือส่วนของผู้โดยสาร ซึ่งจะทราบว่า แท็กซี่คันใดเป็นผู้ให้บริการ และมีรายละเอียดของคนขับ เช่น ชื่อผู้ให้บริการ และทะเบียนรถแท๊กซี่ โดยจะมี ไอคอน >> ซึ่งมีความหมายว่า ผู้ใช้บริการสามารถกดดูข้อมูลคนขับเพิ่มเติมได้ และในส่วนของแผนที่ก็จะมีลูกศร (สีเหลือง) ซึ่งแสดงเส้นทางระหว่างแท็กซี่ที่ผู้ให้บริการและผู้โดยสาร เป็นแบบ Real Time โดยบอกระยะทางว่าไกลเท่าไร ซึ่งระหว่างนั้น ผู้โดยสารสามารถกดยกเลิกใช้บริการนี้ได้

5.3.1.7 ส่วนของ การแสดงเส้นทางตั้งแต่จุดเริ่ม ถึง จุดหมายปลายทาง ของผู้ใช้งาน (Passenger)



รูป 5.3.1.7 Interface ของ Mali Taxi หน้า การแสดงเส้นทางตั้งแต่จุดเริ่ม ถึง จุดหมายปลายทาง ของผู้ใช้งาน (Passenger)

หน้าแสดงเส้นทางตั้งแต่จุดเริ่มต้น ถึง จุดหมายปลายทางของผู้โดยสาร (ผู้ใช้งาน) คือส่วนที่ผู้ใช้งาน ทราบเส้นทางและ ระยะทางระหว่างรถแท็กซี่กับจุดหมายปลายทาง แบบ Real time โดยไอคอน  แสดงตำแหน่งปัจจุบันของผู้ใช้งาน(ไอคอนสีฟ้า) และ จุดหมายปลายทาง  (ไอคอนสีแดง)

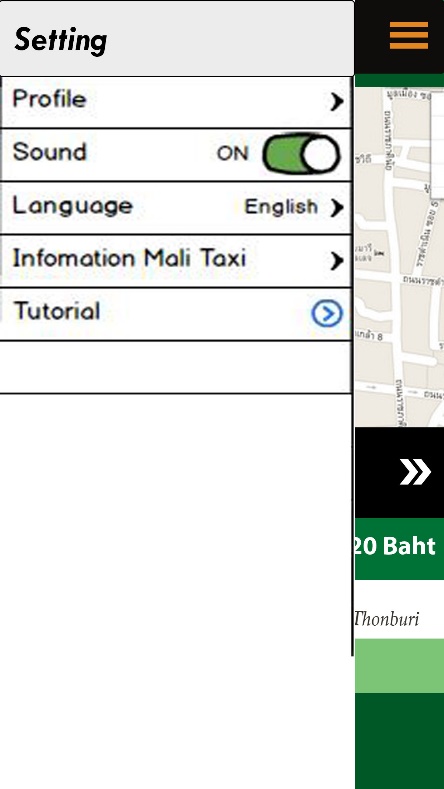
5.3.1.8 ส่วนของ การแสดงรายละเอียดของผู้ให้บริการ (Driver)



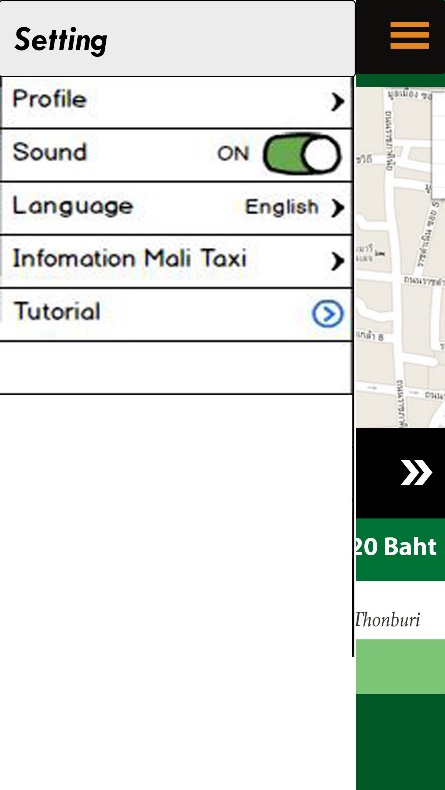
รูป 5.3.1.8 Interface ของ Mali Taxi หน้า การแสดงรายละเอียดของผู้ให้บริการ (Driver)

ส่วนของ การแสดงรายละเอียดของผู้ให้บริการ (Driver) โดยจะแสดงข้อมูลพื้นฐานของผู้ให้บริการที่จะมาให้บริการของผู้โดยสาร โดยจะมีข้อมูล ดังนี้ ภาพถ่าย ของผู้ให้บริการ ทะเบียนรถแท็กซี่ เลขใบขับขี่ เลขบัตรประชาชน และ เบอร์โทรศัพท์ โดยเมื่อกดปุ่ม OK จะเป็นการย้อนไปหน้าก่อนหน้านี้

5.3.1.9 ส่วนของ การตั้งค่าข้อมูล ของผู้ใช้งาน (Passenger)

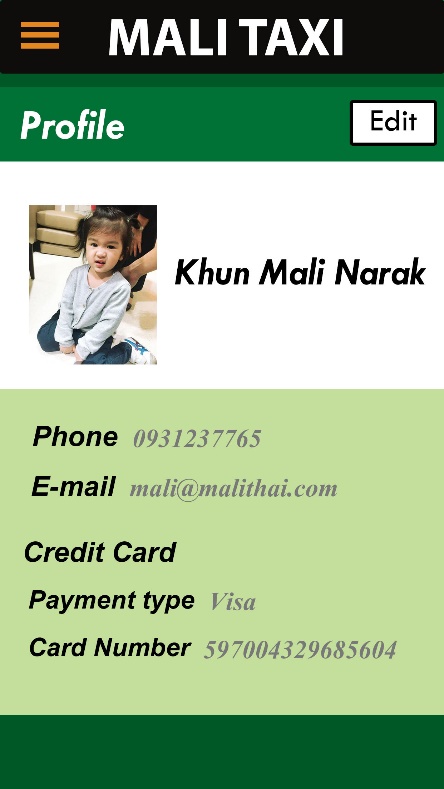


รูป 5.3.1.9 Interface ของ Mali Taxi หน้า Setting ของผู้ใช้งาน (Passenger)

ส่วนของ การตั้งค่าข้อมูล โดยเมื่อกดปุ่ม แทบของ การตั้งค่าข้อมูล จะเด้งแทบ การตั้งค่าข้อมูล ขึ้นมาจากด้านซ้ายมือ โดยจะมีปุ่มดังนี้

* Profile โดยจะแสดงข้อมูลที่ผู้ใช้งานกรอกข้อมูลไว้ใน ส่วนของ Register ในตอนแรก
* Sound จะเป็นปุ่มเปิด-ปิด ของแอปพลิเคชั่น Mali Taxi
* Language เป็นการให้ผู้ใช้ สามารถเลือกใช้ภาษาที่ตนเองถนัดได้ เช่น Thai, English เป็นต้น
* Information Mali Taxi จะเป็นการบอกถึงข้อมูลของแอปพลิเคชั่น เช่น จุดประสงค์ กฏต่างๆ เบอร์ติดต่อศูนย์ Mali Taxi เป็นต้น
* Tutorial จะเป็นการแสดงวิธีการใช้งานของ แอปพลิเคชั่น Mali Taxi โดยจะใช้ลูกศรชี้บอกวิธีการใช้แบบง่ายๆ ซึ่งในครั้งแรกของการใช้งานแอปพลิเคชั่น Mali Taxi จะมีการสอนใช้

5.3.1.10 ส่วนของ โปรไฟล์ ของผู้ใช้งาน (Passenger)



รูป 5.3.1.10 Interface ของ Mali Taxi หน้า โปรไฟล์ ของผู้ใช้งาน (Passenger)

ส่วนของ โปรไฟล์ ส่วนนี้เป็นการแสดงข้อมูลส่วนตัวของผู้ใช้งาน โดยจะแสดงชื่อ และ รูปภาพ ที่ทำการผูกไว้กับ Facebook และแสดงเบอร์โทรศัพท์ อีเมล์ ข้อมูลบัตรเครดิต เมื่อกดปุ่ม OK จะกลับไปสู่หน้าก่อนหน้านี้ ซึ่งจะมีปุ่ม Edit ไว้สำหรับแก้ไขข้อมูล โดยหน้าของการแก้ไขจะเหมือนกับหน้า Register โดยจะสามารถแก้ไขข้อมูลได้ ดังนี้ โทรศัพท์ อีเมล์ และ ข้อมูลบัตรเครดิต

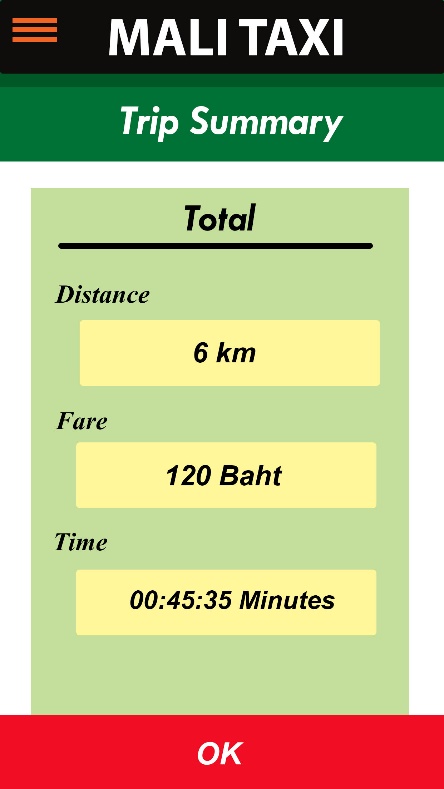
5.3.1.11 ส่วนของ การแก้ไขโปรไฟล์ ของผู้ใช้งาน (Passenger)



รูป 5.3.1.11 Interface ของ Mali Taxi หน้า การแก้ไขโปรไฟล์ ของผู้ใช้งาน (Passenger)

ส่วนของการแก้ไขโปรไฟล์ ของผู้ใช้งาน (Passenger) ไว้สำหรับแก้ไขข้อมูล โดยสามารถแก้ไขข้อมูลได้ ดังนี้ เบอร์โทรศัพท์, อีเมล์, ข้อมูลบัตรเครดิต โดยสามารถ เปลี่ยนประเภทของบัตรเครดิต, หมายเลขบัตรเครดิต และจำเป็นต้องกรอกรหัส Security Code หลังจากนั้นกด Done จะเป็นการแก้ไขข้อมูลสำเร็จ

5.3.1.12 ส่วนของ การสรุปผลการใช้งาน ของผู้ใช้งาน (Passenger)



รูป 5.3.1.12 Interface ของ Mali Taxi หน้า การสรุปผลการใช้งาน ของผู้ใช้งาน (Passenger)

ส่วนของ สรุปผลการใช้งาน ของผู้ใช้งาน (Passenger) โดยส่วนนี้จะเป็นการสรุปผลของเที่ยวโดยสาร โดยจะแสดงข้อมูล ระยะทางทั้งหมดที่เดินทางตั้งแต่จุดเริ่มต้นที่ผู้โดยสารอยู่ ณ ตอนแรก จนถึงจุดปลายทาง และสรุปค่าใช้จ่ายตามระยะทางที่วิ่งมา ซึ่งจะมีการบอกเวลาตั้งแต่ผู้โดยสารขึ้นรถแท็กซี่ จนถึงจุดหมายปลายทาง เมื่อกดปุ่ม OK จะเป็นการสิ้นสุดการใช้บริการ

5.3.2 แอปพลิเคชั่น Mali Driver ของ ผู้ให้บริการ (Driver)

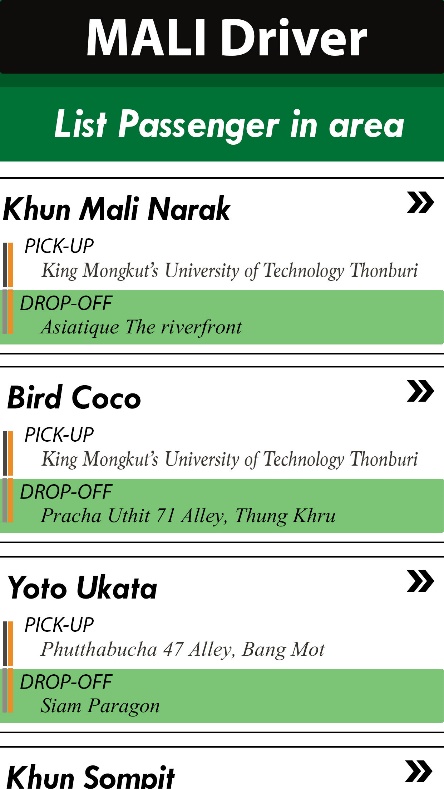
5.3.2.1 ส่วนของ Login ของผู้ให้บริการ (Driver)



รูป 5.3.2.1 Interface ของ Mali Driver หน้า Login

หน้าล็อกอินของผู้ให้บริการ (แท็กซี่) โดยจะต้องกรอก ID และ Password ที่ผู้ใช้ได้ทำการลงทะเบียนกับทางบริษัทไว้ แล้วทำการกด Login เพื่อเริ่มให้บริการผู้ที่ต้องการใช้บริการแท็กซี่

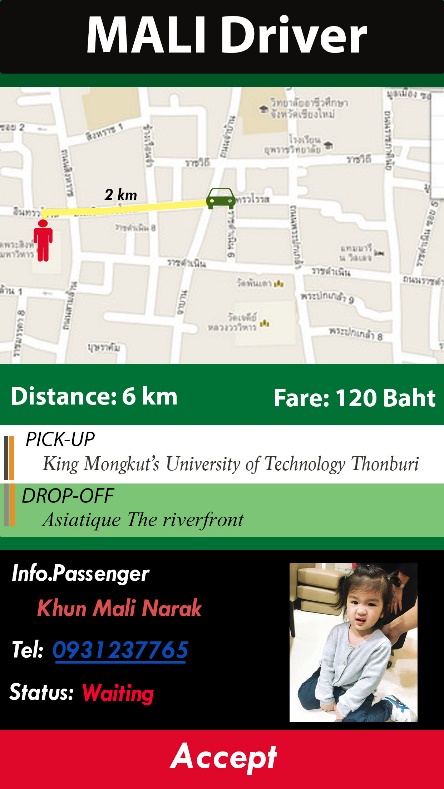
5.3.2.2 ส่วนของ ลิสรายชื่อของผู้ใช้บริการ



รูป 5.3.2.2 Interface ของ Mali Driver หน้า ลิสรายชื่อของผู้ใช้บริการ

ส่วนของ ลิสรายชื่อของผู้ใช้บริการ จะเป็นหน้าแรกเมื่อทำการ Login เข้ามาแล้ว โดยจะแสดงรายชื่อของผู้โดยสาร ที่ต้องการจะใช้บริการ ซึ่งจะแสดง ชื่อของผู้ใช้บริการ จุดที่ต้องการให้ไปรับ และจุดหมายปลายทางที่ต้องผู้โดยสารต้องการจะไป และสัญลักษณ์ **>>** หมายถึง สามารถกดเข้าไปดูรายละเอียดที่มากขึ้นได้ เมื่อใช้มือเลื่อนหน้าจอขึ้น จะเป็นการ Refresh ข้อมูล

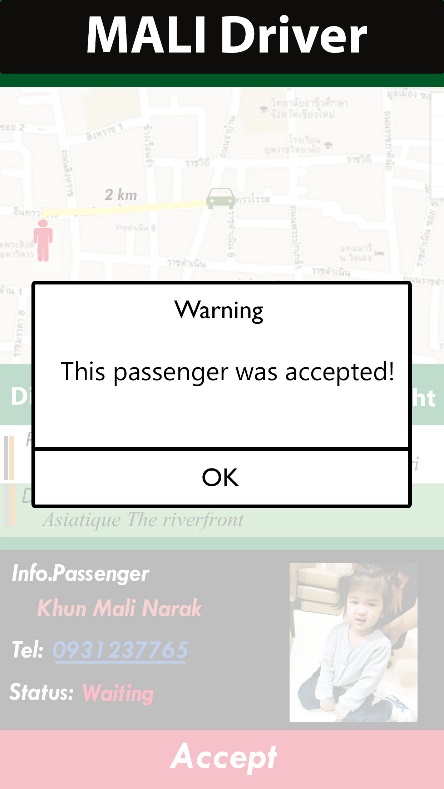
5.3.2.3 ส่วน รายละเอียดผู้โดยสารที่ทำการเรียกใช้บริการ



รูป 5.3.2.3 Interface ของ Mali Driver หน้า รายละเอียดผู้โดยสารที่ทำการเรียกใช้บริการ

ส่วนของ รายละเอียดผู้โดยสารที่ทำการเรียกใช้บริการ โดยจะปรากฏแผนที่ที่มีสัญลักษณ์รูปคน หมายถึงผู้โดยสาร และสัญลักษณ์รูปรถ หมายถึงตัวผู้ขับเอง โดยจะมีการบอกระยะทางระหว่างผู้ขับกับผู้โดยสาร กล่องสีเขียวจะแสดงระยะทางจากจุดที่ผู้โดยสารอยู่จนถึงจุดหมายปลายทาง และราคาค่าโดยสาร และทำการแสดงของข้อมูลผู้โดยสาร โดยจะแสดงชื่อ และเบอร์โทรศัพท์ ซึ่งเบอร์โทรศัพท์สามารถกดที่หมายเลขได้เลย ระบบจะทำการโทรไปหาผู้โดยสารได้ทันที และ Status จะเป็นการบอกถึงสถานะว่า มีแท็กซี่ให้บริการแล้วหรือยัง โดย Waiting หมายถึง ยังไม่มีแท็กซี่คันไหนให้บริการ Accepted หมายถึง มีแท็กซี่ให้บริการผู้โดยสารคนนี้ไปแล้ว มีรูปภาพของผู้โดยสาร และปุ่ม Accept หมายถึง ตอบรับผู้โดยสารว่าจะเป็นผู้ให้บริการ

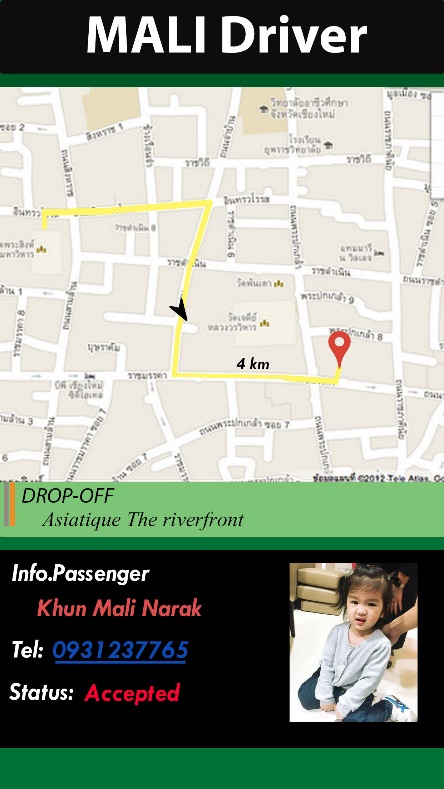
5.3.2.4 ส่วน Warning เมื่อกดรับผู้โดยสารที่ผู้ให้บริการคนอื่นกดให้บริการแล้ว



รูป 5.3.2.4 Interface ของ Mali Driver หน้า Warning เมื่อกดรับผู้โดยสารที่ผู้ให้บริการคนอื่นกดให้บริการแล้ว

ถ้าเมื่อผู้ใช้กด Accept แล้ว มีผู้ให้บริการ บริการไปแล้วจะมี Warning ขึ้นมาเตือน และให้กดปุ่ม OK จะกลับไปหน้า ลิสรายชื่อใหม่ และรายชื่อของผู้ใช้บริการคนนี้จะหายไปจากหน้าลิสรายชื่อ

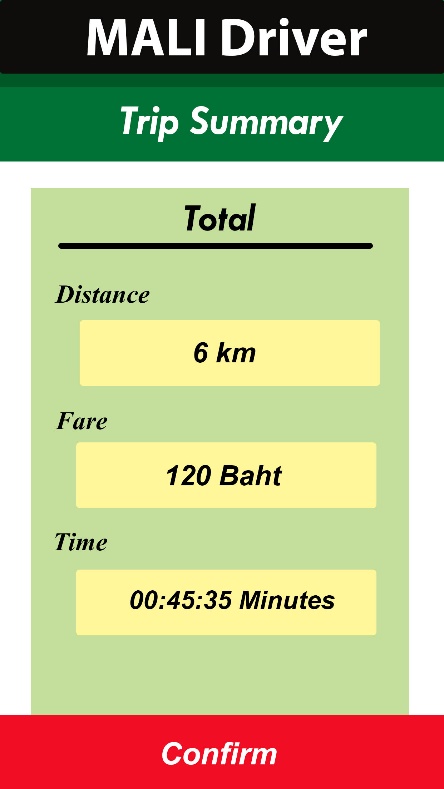
5.3.2.5 ส่วน แผนที่บอกเส้นทาง ของฝั่งผู้ให้บริการ



รูป 5.3.2.5 Interface ของ Mali Driver หน้า แผนที่บอกเส้นทาง ของฝั่งผู้ให้บริการ

ส่วนของ แผนที่บอกเส้นทาง ของฝั่งผู้ให้บริการ โดยจะแสดงแผนที่บอกเส้นทางตั้งแต่จุดเริ่มต้นจนถึงจุดหมายปลายทาง โดยสัญลักษณ์  หมายถึง ตำแหน่งปัจจุบันที่แท็กซี่อยู่ และ สัญลักษณ์  หมายถึง จุดหมายปลายทางที่จะไป

5.3.2.6 ส่วน สรุปผลการใช้งาน ของผู้ให้บริการ (Driver)



รูป 5.3.2.6 Interface ของ Mali Driver หน้า สรุปผลการใช้งาน

ส่วนของ สรุปผลการใช้งาน โดยส่วนนี้จะเป็นการสรุปผลของเที่ยวโดยสาร โดยจะแสดงข้อมูล ระยะทางทั้งหมดที่เดินทางตั้งแต่จุดเริ่มต้นที่ผู้โดยสารอยู่ ณ ตอนแรก จนถึงจุดปลายทาง และสรุปค่าใช้จ่ายตามระยะทางที่วิ่งมา ซึ่งจะมีการบอกเวลาตั้งแต่ผู้โดยสารขึ้นรถแท็กซี่ จนถึงจุดหมายปลายทาง เมื่อกดปุ่ม Confirm จะเป็นการหักค่าโดยสารจากบัตรเครดิตผู้โดยสารอัตโนมัติ

**6. OTHER INTERFACES DESIGN**

6.1. HARDWARE INTERFACES DESIGN

Hardware ที่ใช้ในระบบ Mali Taxi ประกอบไปด้วย

* + Computer server
  + Computer client

Mali Taxi เป็นระบบที่มีการติดต่อรับส่งกับ Hardware ที่มีการเชื่อมต่อเข้าด้วยกันระหว่าง Computer server ซึ่งใช้ระบบปฏิบัติการ Window 10 Enterprise CPU รุ่น Intel Core i7- 6700K Processor 4.00 GHz 4 Core 8 Threadใช้ Main Board ASUS MAXIMUS VIII EXTREME/ASSEMBLY Ram G.SKILL Trident Z DDR4 32GB 3200 (8GBx4) GTZ GPU ASUS ROG ARES III 8GB Hard Disk WD Black 1\*6 TB (RAID 0) SSD Samsung 850 pro (1\*4 TB) และ Computer client ที่ User ใช้งาน โดยเชื่อมต่อกันผ่านเครือข่าย Internet ซึ่ง server มีความเร็วในการรับส่งข้อมูล 5 Mbps

เมื่อ User หรือ Admin ใช้งานระบบ Server ของระบบ จะมีการสำรองข้อมูลลงไปใน Hard disk ของ Server ในทุกๆ 3 วัน โดยจะมีการเก็บข้อมูลด้วยวิธีการเข้ารหัสข้อมูล เพื่อเพิ่มความปลอดภัยและมีความจุสูงสุดถึง 500 GB รวมทั้งมีแหล่งจ่ายไฟสำรองเพื่อป้องกันข้อมูลสูญหายในกรณีที่เกิดเหตุการณ์ที่ไม่สามารถกำหนดได้ เช่น ไฟกระชาก เป็นต้น

6.2. SOFTWARE INTERFACES DESIGN

Software interfaces เป็นรูปแบบการใช้งานที่มีการติดต่อกันระหว่างผู้ใช้งาน, ระบบ และ Software โดยมีการเชื่อมต่อดังนี้

* ลูกค้าสามารถเรียกใช้บริการรถแท็กซี่ผ่าน แอปพลิเคชั่น โดยมือถือต้องเป็นระบบปฏิบัติการ Android และสามารถทำการเชื่อมต่ออินเตอร์เน็ตได้
* ระบบสามารถเชื่อมต่อกับ Facebook ได้ ซึ่งจะง่ายต่อ User จะไม่ต้องสมัคร รหัสใหม่
* ระบบเชื่อมต่อกับ Google API เพื่อที่จะใช้งาน Google map
* ระบบมีส่วนของ GPS เพื่อแสดง Location ของ ลูกค้า และฝั่งคนขับด้วย
* ระบบเชื่อต่อกับ Database ไว้สำหรับเก็บข้อมูลต่าง ๆ ที่จะแสดงผล ทั้งส่วนของคนขับและ User

6.3. COMMUNICATION INTERFACES DESIGN

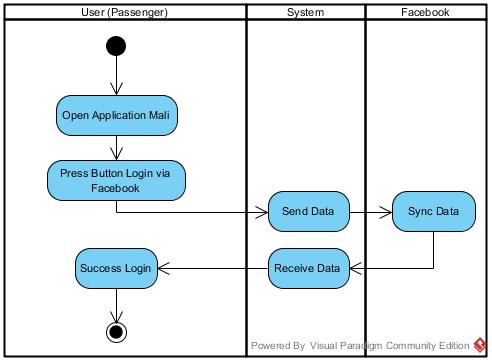
Communication Interface เป็นรูปแบบการใช้งานที่มีการเชื่อมต่อระหว่าง แอปพลิเคชั่นกับระบบต่างๆ โดยมีการเชื่อมต่อดังนี้

* User ทำการเรียกใช้บริการแท็กซี่ผ่านทาง แอปพลิเคชั่น บน Smartphone โดยมีการรับ-ส่งข้อมูลกับ Server
* ระบบจะมีการเชื่อมต่อกับระบบของทางธนาคาร (Banking) เพื่อตรวจสอบและยืนยันสถานการณ์ชำระเงิน ผ่านการตัดยอดทางบัตรเครดิต

**7. HOW TO USE FOR EACH SCENARIO**

7.1 Passenger Scenario

7.1.1 Scenario 1: การเข้าสู่ระบบ (Login) ของ Passenger



1. ผู้ใช้เข้าใช้งานแอปพลิเคชั่น Mali Taxi

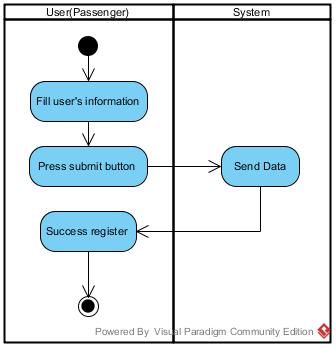
2. ระบบแสดงหน้าจอสำหรับ Login

3. ผู้ใช้เลือกกดผูกบัญชี Account Facebook เข้ากับแอปพลิชั่น

4. ระบบทำการถ่ายโอนข้อมูลจาก Facebook

5. เมื่อสำเร็จจะเข้าสู่หน้าใช้งานต่อไป

7.1.2 Scenario 2: การสมัครสมาชิก (Register)



1. ผู้ใช้กรอกข้อมูลส่วนตัว

1.1 เบอร์โทรศัพท์

1.2 อีเมล์

1.3 ประเภทของบัตรเครดิต

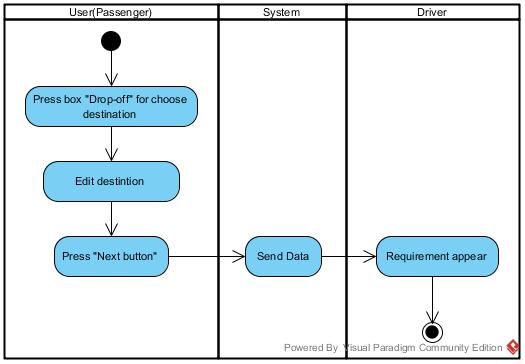
1.4 เลขบัตรเครดิต

1.5 รหัสยืนยัน (เลขหลังบัตรเดรดิต)

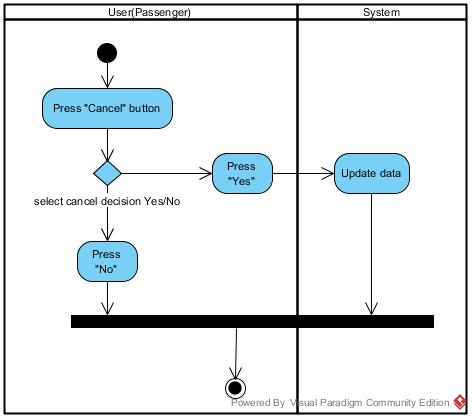
2. กด Submit เพื่อยืนยันการเข้าใช้งาน

3.เมื่อสำเร็จจะเข้าสู่หน้าใช้งานต่อไป

7.1.3 Scenario 3: เรียกใช้บริการแท็กซี่



1. ให้ผู้ใช้สังเกตที่ช่อง Drop-off จากนั้นให้ผู้ใช้กดไปที่ช่อง เพื่อเลือกสถานที่ที่จะไป
2. ให้ผู้ใช้กรอกชื่อสถานที่ที่จะไป
3. เมื่อผู้ใช้กรอกสถานที่ที่จะไปแล้วให้กดปุ่ม Next ระบบจะทำการส่งข้อมูลให้ฝั่งผู้ให้บริการได้รับทราบว่า ผู้ใช้ต้องการเรียกใช้บริการ

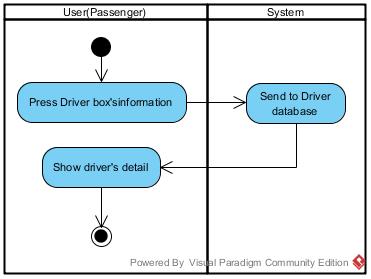
7.1.4 Scenario 4: ยกเลิกการเรียกใช้บริการแท็กซี่

การยกเลิกใช้บริการหลังจากได้เรียกไปแล้ว โดยจะสามารถยกเลิกได้เพียงตอนที่ รอผู้ให้บริการให้บริการ และ ระหว่างที่แท็กซี่กำลังจะมาหาผู้ใช้

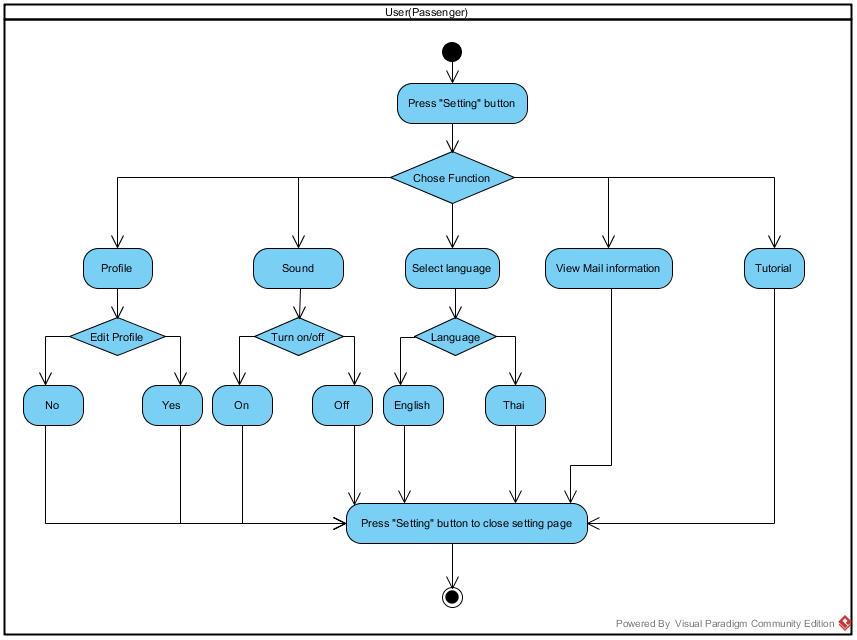
ให้ผู้ใช้สังเกตุ ปุ่ม Cancel ข้างล่างของหน้าจอ เมื่อกดจะมี Warning เตือน

* + - 1. เมื่อกดปุ่ม Yes จะเป็นการยกเลิกการเรียกใช้บริการ และจะกลับไปยังหน้าเรียกใช้บริการ
      2. เมื่อกดปุ่ม No จะเป็นการไม่ยกเลิกการเรียกใช้บริการ และจะยังคงอยู่หน้าการยกเลิกบริการ

7.1.5 Scenario 5: ข้อมูลของผู้ให้บริการ



เมื่อทำการเรียกใช้บริการแท็กซี่ ข้อมูลของผู้ให้บริการจะขึ้นแสดงอยู่เสมอจนถึงจุดหมายปลายทาง ให้ผู้ใช้สังเกตุ กล่องสีดำ ที่เป็นข้อมูลของผู้ให้บริการโดยเมื่อกด จะไปยังหน้า ข้อมูลของผู้ให้บริการ

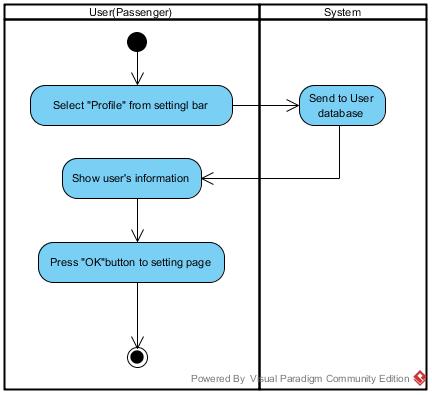
7.1.6 Scenario 6: Setting

เมื่อผู้ใช้กด Setting หน้าจอแสดงผลจะแบ่งส่วนทำงานออกเป็น 5 ส่วน

1. Profile สามารถดูข้อมูล และแก้ไขข้อมูลได้ จะพูดถึงในหัวข้อถัดไป
2. Sound สามารถกดเปิด-ปิด ได้โดยกดที่ปุ่ม on-off
3. Language เมื่อกดเข้าไปจะมีภาษาให้เลือก เช่น ไทย, English เป็นต้น
4. Information Mali Taxi เมื่อกดเปิดจะมีข้อมูลที่เกี่ยวข้องกับ แอปพลิเคชั่น Mali Taxi
5. Tutorial เมื่อกดจะแสดงวิธีการใช้งานของ แอปพลิเคชั่น Mali Taxi โดยจะภาพการใช้แบบง่ายๆ

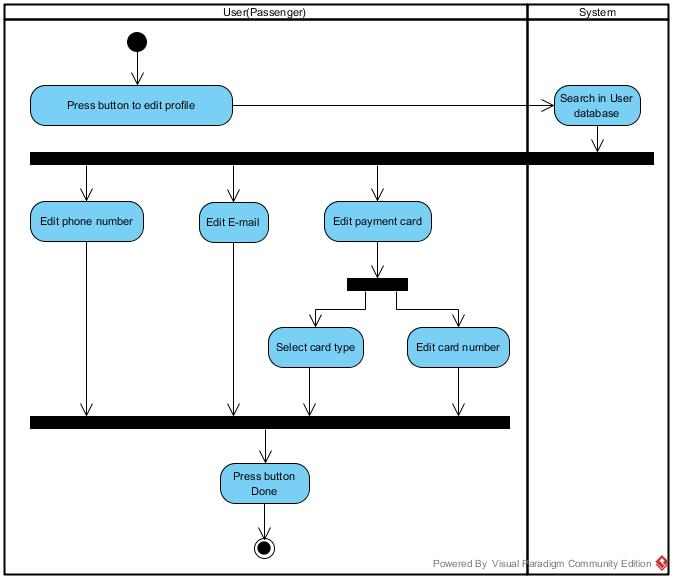
และเมื่อกดปุ่ม Setting อีกครั้งจะเป็นการปิดแถบ Setting

7.1.6 Scenario 7: Profile



เมื่อผู้ใช้กดเลือก Profile จากแถบ Setting จะเป็นการดู ข้อมูลส่วนตัวของผู้ใช้งาน ได้แก่ ชื่อ รูปภาพ เบอร์โทรศัพท์ อีเมล์ ประเภทของบัตรเครดิต และหมายเลขบัตรเครดิต และเมื่อกดปุ่ม OK จะย้อนกลับไปหน้าก่อนหน้า

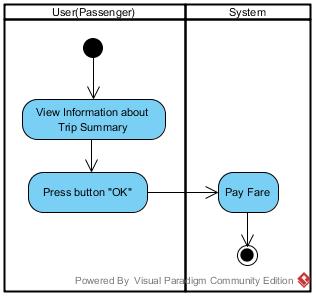
7.1.7 Scenario 7: แก้ไข โปรไฟล์



จากหน้า โปรไฟล์ ปุ่มด้านบนขวามือ เมื่อกด จะเป็นการ แก้ไขข้อมูล ระบบจะแสดงผลข้อมูลของการแสดงเดิมที่เคยสร้างไว้แล้ว เช่น เบอร์โทรศัพท์ อีเมล์ ประเภทของบัตรเครดิต และหมายเลขบัตรเครดิต

1. ถ้าผู้ใช้ต้องการเปลี่ยน เบอร์โทรศัพท์ ผู้ใช้กดที่กล่องเบอร์โทรศัพท์ จากนั้นทำการกรอกข้อมูลใหม่ลงในกล่องข้อความ
2. ถ้าผู้ใช้ต้องการเปลี่ยน อีเมล์ ผู้ใช้กดที่กล่อง อีเมล์ จากนั้นทำการกรอกข้อมูลใหม่ลงในกล่องข้อความ
3. ถ้าผู้ใช้ต้องการเปลี่ยน หมายเลขบัตรเครดิตในการชำระเงิน
   1. ให้ผู้ใช้กดที่ ประเภทของบัตรเครดิต เช่น Visa, MasterCard, Amex เป็นต้น
   2. ให้ผู้ใช้กดที่ ช่องหมายเลขบัตรเครดิต จากนั้นทำการกรอกข้อมูลใหม่ลงในกล่องข้อความ
   3. ให้ผู้ใช้กดที่ รหัสยืนยัน จากนั้นทำการกรอกข้อมูลใหม่ลงในกล่องข้อความ
4. ให้ผู้ใช้กดปุ่ม Done เพื่อเป็นการสิ้นสุดการแก้ไขข้อมูล

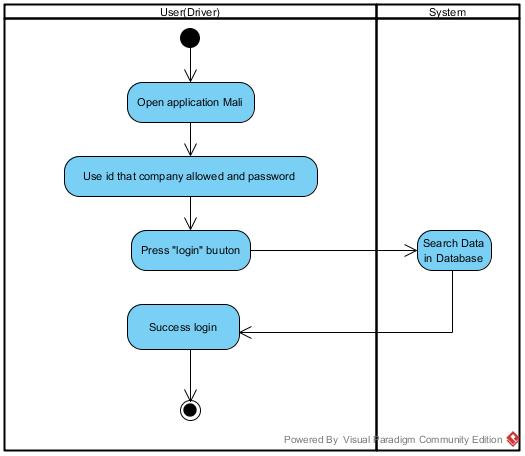
7.1.8 Scenario 8: สรุปผลการใช้งาน



เมื่อผู้ใช้ถึงจุดหมายปลายทางจะแสดงผลการใช้งาน โดยแสดงข้อมูลที่สำคัญในการใช้บริการ ได้แก่ ระยะทาง, ค่าโดยสาร และ เวลาการเดินทาง เมื่อกดปุ่ม OK จะเป็นการสิ้นสุดการใช้บริการ

7.2 Driver Scenario

7.2.1 Scenario 1: การเข้าสู่ระบบ (Login) ของ Driver



1.ผู้ใช้เข้าใช้งานแอปพลิเคชั่น Mali Driver

2.ระบบแสดงหน้าจอสำหรับ Login

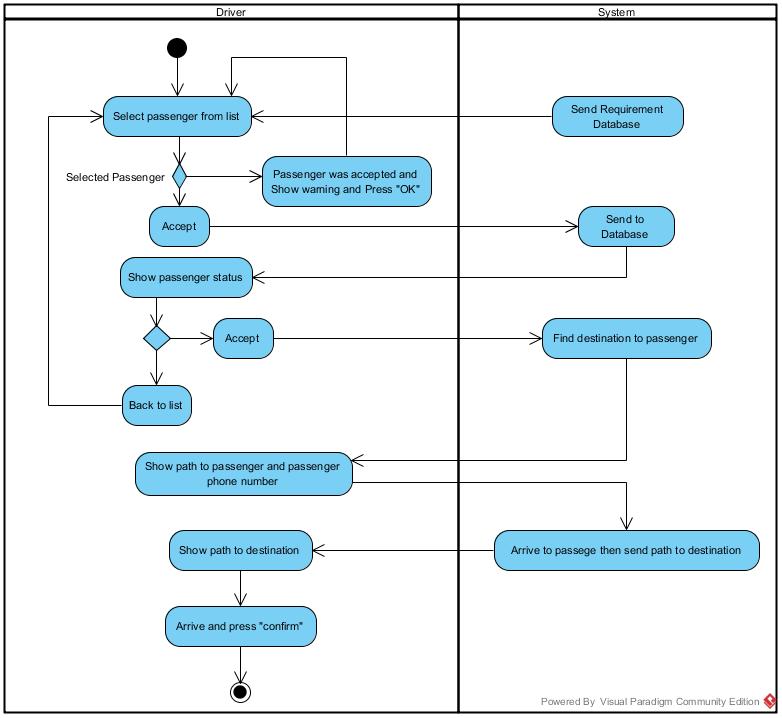
3.ผู้ใช้กดช่อง ID แล้วกรอก ID ที่ทางบริษัทได้ให้ไว้

4.ผู้ใช้กดช่อง Password แล้วกรอก รหัสผ่าน

5.เมื่อผู้ใช้กรอก ID และ รหัสผ่าน ให้กดปุ่ม Login ที่อยู่ล่างสุดของหน้าจอ

6. เมื่อสำเร็จจะเข้าสู่หน้าใช้งานต่อไป

7.2.2 Scenario 2:ให้บริการแก่ผู้ใช้บริการ



1. ผู้ใช้กดเลือกผู้โดยสารจากหน้าจอ ลิสรายชื่อ
2. เมื่อทำการเลือกลูกค้า ให้ผู้ใช้ดูว่า Status เป็น Waiting หรือ Accepted
3. ถ้า Status เป็น Waiting แล้วผู้ใช้ต้องการให้บริการ ให้กดปุ่ม Accept ที่ด้านล่างของจอ ถ้าเมื่อกดแล้ว มีผู้ให้บริการ บริการไปแล้วจะมี Warning ขึ้นมาเตือน และกดปุ่ม OK จะกลับไปหน้า ลิสรายชื่อใหม่ หรือ ผู้ใช้ไม่ต้องการให้บริการ ให้กดกลับจากปุ่มตัวเครื่องของผู้ใช้งาน
4. ถ้า Status เป็น Accepted ให้ผู้ใช้กดกลับจากปุ่มตัวเครื่องของผู้ใช้งาน
5. เมื่อ กด Accept แล้วจะแสดงหน้าที่เป็นเส้นทางไปหาผู้ใช้บริการ โดยสามารถกดที่เบอร์โทรศัพท์ เพื่อโทรติดต่อหาผู้โดยสารได้ทันที
6. เมื่อถึงจุดหมายปลายทางหน้าจอจะแสดงผล หน้าสรุปยอดรวมของเที่ยวโดยสารนี้ ให้ผู้ใช้กดปุ่ม Confirm เพื่อเป็นการยืนยันค่าใช้จ่าย และหักจากบัตรเครดิตของผู้ใช้บริการ

**8. REQUIREMENTS VALIDATION MATRIX**

8.1. FUNCTIONAL REQUIREMENTS CHECKLIST

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Requirement description | Priority | Done | Notes |
| 1 | ส่วนของ Login ของผู้ใช้บริการ (Passenger) | Must have | 🗸 | 2.1 Internal software data structure  2.2 Global data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 2 | Register ของผู้ใช้งาน (Passenger) | Must have | 🗸 | 2.1 Internal software data structure  2.3 Global data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 3 | การเรียกใช้งานแท็กซี่ | Must have | 🗸 | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 4 | การ Match ระหว่างผู้ใช้บริการ กับผู้ให้บริการ | Must have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 5 | การยกเลิกใช้บริการ | Must have | 🗸 | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 6 | แสดงรายละเอียดผู้ให้บริการที่มารับ | Must have | 🗸 | 2.1 Internal software data structure  2.2 Global data structure  2.4 Database Description 3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 7 | การแสดงเส้นทางตั้งแต่จุดเริ่ม ถึง จุดหมายปลายทาง | Must have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 8 | การแสดงข้อมูลของผู้ให้บริการ (Driver) | Must have | 🗸 | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.3 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 9 | ส่วนของ การตั้งค่าข้อมูล ของผู้ใช้งาน (Passenger) | Should have |  | 2.1 Internal software data structure  2.2 Global data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 10 | ส่วนของ โปรไฟล์ ของผู้ใช้งาน (Passenger) | Should have |  | 2.1 Internal software data structure  2.2 Global data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 11 | ส่วนของ การแก้ไขโปรไฟล์ ของผู้ใช้งาน (Passenger) | Should have |  | 2.1 Internal software data structure  2.2 Global data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 12 | ส่วนของ การสรุปผลการใช้งาน ของผู้ใช้งาน (Passenger) | Should have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.1 Description Component1: Passenger component  5. User interface design  7. How to use for each scenario |
| 13 | ส่วนของ Login ของผู้ให้บริการ (Driver) | Must have | 🗸 | 2.1 Internal software data structure  2.2 Global data structure  2.4 Database Description  3.1 Architectural design  4.2 Description Component2: Driver Component  5. User interface design  7. How to use for each scenario |
| 14 | ส่วนของ ลิสรายชื่อของผู้ใช้บริการ | Must have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.2 Description Component2: Driver Component  5. User interface design  7. How to use for each scenario |
| 15 | ส่วน รายละเอียดผู้โดยสารที่ทำการเรียกใช้บริการ | Must have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.2 Description Component2: Driver Component  5. User interface design  7. How to use for each scenario |
| 16 | ส่วน แผนที่บอกเส้นทาง ของฝั่งผู้ให้บริการ | Must have | 🗸 | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.2 Description Component2: Driver Component  5. User interface design  7. How to use for each scenario |
| 17 | ส่วน สรุปผลการใช้งาน ของผู้ให้บริการ (Driver) | Must have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4.2 Description Component2: Driver Component  5. User interface design  7. How to use for each scenario |
| 18 | การจัดการข้อมูลสมาชิก | Could have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design |
| 19 | การจัดการระบบฐานข้อมูล | Could have |  | 2.1 Internal software data structure  2.2 Global data structure  2.3 Temporary data structure  2.4 Database Description  3.1 Architectural design  4. Component level design |

8.2. NON-FUNCTIONAL REQUIREMENTS CHECKLIST

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Requirement description | Priority | Done | Notes |
| 1 | ระบบสามารถใช้งานได้กับ Smartphone บนระบบ Android | Must have | 🗸 | Software Interface Design |
| 2 | ระบบมีความรวดเร็วในการประมวลผล | Should have |  | Tester จะเป็นผู้ที่ทำการทดสอบ Requirement นี้ |
| 3 | Software สามารถใช้งานได้ 2 ภาษา ได้แก่ ภาษาไทย และ ภาษาอังกฤษ | Should have |  | Software interface design |
| 4 | User Manual, Backend Manual, Technical Manual | Must have | 🗸 | Document จะเป็นผู้ดำเนินการ Requirement นี้ |
| 5 | การอบรมใช้ Software | Should have |  | Trainer จะเป็นผู้ดำเนินการ Requirement นี้ |
| 6 | ระบบรองรับการเชื่อมต่อกับอินเตอร์เน็ตได้ | Must have | 🗸 | 3.1 Architectural design |
| 7 | การเชื่อมต่อกับระบบการชำระเงินผ่านบัตรเครดิตของธนาคารต่างๆ | Must have | 🗸 | Communication Interface Design |
| 8 | การรับส่งข้อมูลผ่านเครือข่าย Internet แบบ Client-Server | Must have |  | 6.1 Hardware Interface Design |
| 9 | เวลาการตอบสนองของแอปพลิเคชั่นไม่เกิน 3 วินาที | Should have |  | 6.1 Hardware Interface Design |
| 10 | ความรวดเร็วในการรับส่งข้อมูลกับ Server | Should have |  | 6.1 Hardware Interface Design |
| 11 | ระบบมีการป้องกันการเข้าถึงข้อมูลจากบุคคลภายนอก และกำหนดให้ข้อมูลบางส่วนสามารถเข้าดูได้เฉพาะผู้ดูแลระบบ | Must have | 🗸 | 3.1 Architectural Design |
| 12 | ดึงข้อมูลจาก Facebook เพื่อทำการสมัครสมาชิก | Must have | 🗸 | 3.1 Architectural Design |
| 13 | Update Version ,Backup และ Restore | Should have |  | 3.1 Architectural Design |

หมายเหตุ

* Requirement description คือ คำอธิบายของ Requirement นั้นๆ
* Priority คือ ความสำคัญของ Requirement
* Done คือ สถานะว่า Requirement นั้นๆสำเร็จเรียบร้อยแล้ว
* Notes คือ อธิบาย และ อ้างอิง

**PROCESS MANUAL SPECIFICATIONS**

**4. REQUIREMENTS AND CHANGE MANAGEMENT PROCESS**

Work Request Form

Work Request No: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Project Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Manager Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Organizer Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Change Request | | Reason | Date to change | Signature | |
| From | To | Project Manager | Organizer |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Work Request Form ใช้สำหรับบันทึกการเปลี่ยนแปลงความสามารถของโปรแกรมตามความ

ต้องการของลูกค้า หรือเปลี่ยนแปลงความสามารถของโปรแกรมในกรณีที่เกิดข้อผิดพลาดจากการทำงาน

ของโปรแกรม ทั้งนี้การเปลี่ยนแปลงในแต่ละครั้ง จะต้องได้รับการอนุมัติจากลูกค้าและหัวหน้างาน

ส่วนที่ 1 : ข้อมูลทั่วไปของ Project

* Work Request No (ลำดับใบบันทึกรายการ)
* Project ID (หมายเลขงาน)
* Project Name (ชื่องาน)
* Project Manager Name (ชื่อหัวหน้างาน)
* Organizer Name (ชื่อผู้จัดการแสดง)

ส่วนที่ 2 : แสดงรายการบันทึกของการเปลี่ยนแปลงความสามารถของโปรแกรม โดยมีการระบุถึงปัญหาและวิธีการแก้ไข ซึ่งประกอบด้วยส่วนต่างๆ ดังนี้

* No (ลำดับรายการบันทึก)
* Change Request
  + From (ความสามารถของโปรแกรมเดิม)
  + To (ความสามารถของโปรแกรมใหม่)
* Reason (สาเหตุของการเปลี่ยนแปลงงาน)
* Date To Change (วันที่มีการเปลี่ยนแปลงงาน)
* Signature
  + Project Manager (ลายเซ็นของหัวหน้างาน)
  + Organizer (ลายเซ็นของผู้จัดการแสดง)

**5. CONFIGURATION MANAGEMENT PROCESS**

เป็นกระบวนการจัดการหนึ่งใน มาตรฐานกระบวนการพัฒนางาน (Capability Maturity Model Integration, CMMI) มีวิธีการหรือขั้นตอน (process improvement) เพื่อพัฒนาปรับปรุงคุณภาพของผลิตภัณฑ์ (product, service) ให้มีประสิทธิภาพ ตั้งแต่กระบวนการออกแบบ จนถึงการส่งมอบ (Release) และการบำรุงรักษา (Maintenance)

**Version Control**

เนื่องจาก การจัดทำเอกสารแต่ละครั้งนั้น จะมีการเปลี่ยนแปลงข้อมูลอยู่เสมอ ดังนั้นจึงต้องใช้กระบวนการ การจัดการเพื่อที่จะสำรองไฟล์เดิมไว้ ด้วยการระบุ Version ของไฟล์ ซึ่งจะทำให้สามารถนำไฟล์ที่สำรองไว้กลับมาใช้งานในภายหลังได้

**การกำหนด Version**

หลักการที่นำมาใช้เป็นมาตรฐานในการกำหนด Version ของ Document, Software code และTest Case ได้แก่

1. Document

เลขของ Version ประกอบด้วย 2 ส่วน คือ Version number และ Revision number เช่น Version (Version number . Revision number) Version 1.1

* Version number: เปลี่ยนเมื่อมีการเปลี่ยนโครงสร้างหลักของส่วนต่างๆในโปรแกรม เช่น เปลี่ยน Version 1.1 เป็น 2.0
* Revision number: เปลี่ยนเมื่อมีการเปลี่ยนแปลงเนื้อหาในเอกสาร แต่โครงสร้าง และ การทำงานของระบบไม่เปลี่ยนแปลง

เอกสารแต่ละฉบับจะมี Version ที่แตกต่างกันไป ซึ่งหมายเลข Version ที่ถูกกำหนดนี้จะถูกนำไปใช้เป็นส่วนหนึ่งของชื่อเอกสาร เช่น MaliTaxi\_SDD\_v1.1.docx

2. Software Code/Test case

เลขของ Version ประกอบด้วย 3 ส่วน คือ Version number, Revision number และ Update Character

เช่น Version (Version no . Revision number (Update character))

Version 1.1a

* Version number: เปลี่ยนเมื่อมีการเปลี่ยนโครงสร้างหลักของส่วนต่างๆในโปรแกรม เช่น เปลี่ยน Version 1.1 เป็น 2.0
* Revision number: เปลี่ยนเมื่อมีการเพิ่ม Function หรือ เพิ่ม Module ใหม่ ดังนั้น Software จึงมี Revision เป็น 1.0, 1.1, 1.2 เป็นต้น
* Update character: เปลี่ยนเมื่อมีการแก้ไขข้อผิดพลาดของ Software (fixed bug)โดยไม่ได้เพิ่ม Function ใหม่ลงไป เช่น เปลี่ยน Version 1.1a เป็น 1.1b

หมายเหตุ: หากมีการแก้ไข bug พร้อมกับการเพิ่ม Function ใหม่ลงใน Software ก็จะ Update Revision number โดยตัด Update Character ทิ้งไป เช่น เปลี่ยน Version 1.1b เป็น 1.2

**แบบฟอร์มสำหรับการเปลี่ยนแปลงข้อมูลในแต่ละเอกสาร**

**แบบฟอร์มสำหรับการเปลี่ยนแปลงข้อมูลในแต่ละเอกสาร**

Document No: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Document ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Document Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Manager Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Organizer Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Version | Modify | | Description | Date to change | Signature | |
| Modify No. | Page Number | Editor | Project Manager |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

แบบฟอร์มสำหรับการแก้ในแต่ละ Document ใช้สำหรับบันทึกการเปลี่ยนแปลงข้อมูลในเอกสาร

ต่างๆ จะต้องได้รับการลงชื่อการแก้ไขของผู้ทำการแก้ไขและผู้ดูแลโปรเจค

ส่วนที่ 1 : ข้อมูลทั่วไปของ Document Form

* Document No (ลำดับใบบันทึกรายการ)
* Document ID (หมายเลขงานเอกสาร)
* Project Name (ชื่องาน)
* Project Manager Name (ชื่อหัวหน้างาน)
* Organizer Name (ชื่อผู้จัดการแสดง)

ส่วนที่ 2 : แสดงรายการบันทึกของการเปลี่ยนแปลงความสามารถของโปรแกรม โดยมีการระบุถึงปัญหาและวิธีการแก้ไข ซึ่งประกอบด้วยฟิลด์ต่างๆ ดังนี้

* Version (หมายเลข Version ของเอกสาร)
* Modify
  + - * Modify No. (ลำดับของการแก้ไข)
      * Page Number (เลขบรรทัดและเลขหน้าของเอกสารที่มีการแก้ไข)
* Description (รายละเอียดของการเปลี่ยนแปลงข้อมูลในเอกสาร)
* Date to Change (วันที่มีการเปลี่ยนแปลงงาน)
* Signature
  + - * Editor (ลายเซ็นของผู้แก้ไขงาน)
      * Project Manager (ลายเซ็นของหัวหน้างาน)

**Document Configuration**

เมื่อมีการแก้ไข Document (Requirements specification document, Design specification document, User manual, technical manual) แล้วจะต้องแก้ไขส่วนต่างๆภายใน Document เพื่อเปลี่ยนแปลงข้อมูลให้เป็นปัจจุบัน ซึ่งทุกเอกสารจะมีการแก้ไขในรูปแบบเดียวกัน ดังต่อไปนี้

ตัวอย่างการแก้ไข Software Design Document(SDD)



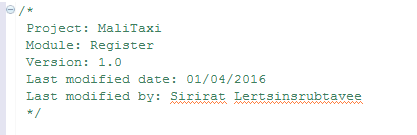
เปลี่ยน วันที่ และ Version ของ Software Design Document(SDD) ให้ตรงตามข้อมูลปัจจุบัน ซึ่งเอกสารฉบับนี้จะถูกนำไปเซ็นรับรองการเปลี่ยนแปลงโดยผู้เกี่ยวข้องกับการพัฒนา Software นี้เพื่อที่จะใช้เป็นเอกสารล่าสุด

**Format ของชื่อไฟล์ที่ใช้บันทึก**

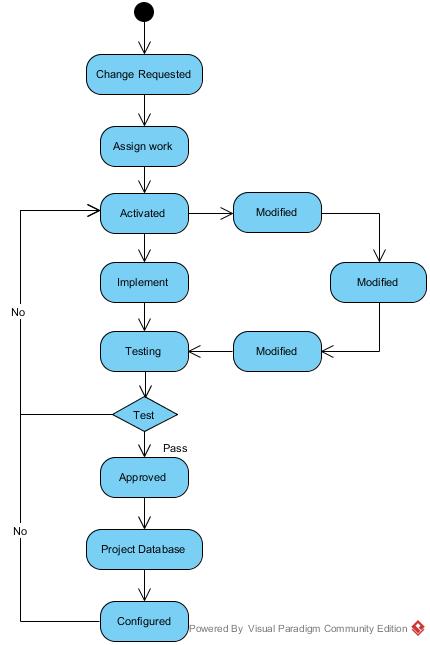
“<Project Name>\_<version>.สกุลไฟล์ “

**Test Case/ Source Code Configuration**

เมื่อมีการแก้ไข Test case/Source Code แล้ว จะต้องแก้ไขส่วนต่างๆภายใน Document เพื่อ

เปลี่ยนแปลงข้อมูลให้เป็นปัจจุบัน ทุกเอกสารจะมีการแก้ไขในรูปแบบเดียวกัน ดังต่อไปนี้

บริษัทบันทึกรายละเอียดต่างๆของการเปลี่ยนแปลง version ของ ซอฟต์แวร์ และเอกสาร ที่บริษัทกำลังพัฒนาอยู่ บริษัทจะดำเนินงานตามขั้นตอนใน Flow Chart ดังนี้



อธิบายขั้นตอนการทำงานได้ดังนี้

1. Activated Developer จะเริ่มทำการพัฒนางานที่ได้รับมอบหมายในขั้นตอนนี้ แบ่งออกเป็น 2 แบบ

- Implement Developer ทำการพัฒนางานที่ได้รับมอบหมาย คืองานที่ลูกค้าทำการ Request เข้ามา

- Modified Developer ทำการแก้ไข ปรับปรุงงานที่ได้ทำไว้ ซึ่งในการแก้ไขนั้น ก่อนที่จะ Modified ได้ จะต้องมีการ Locked เพื่อป้องกันไม่ให้เกิดการแก้ไขซ้ำซ้อนกัน เช่น เกิดการแก้ไขงานมากกว่า 1 คน ซึ่งหลังจาก Developer ทำการแก้ไข หรือปรับปรุงเรียบร้อยแล้ว จะทำการ Unlock เพื่อให้คนอื่นสามารถแก้ไขงานต่อได้

2. Testing ขั้นตอนนี้ Tester จะเป็นผู้ทดสอบโปรแกรม เพื่อหาข้อผิดพลาดต่างๆ ของชิ้นงานที่ถูกพัฒนา โดยที่ Developer จะเป็นผู้ปรับปรุง แก้ไข

3. Approved เป็นกระบวนการที่ให้ลูกค้าพิจารณาซอฟต์แวร์ เวอร์ชั่นใหม่ ที่ได้ทำการแก้ไข ซึ่งในขั้นตอนนี้จะมีการให้ลูกค้าลงชื่อ เพื่อยินยอม และรับทราบ ซอฟต์แวร์เวอร์ชั่นใหม่

4. Database หลังจากที่ลูกค้ารับทราบและยินยอม ซอฟต์แวร์เวอร์ชั่นใหม่ที่ได้ปรับปรุงแล้ว จะมีการเก็บลงในฐานข้อมูล

5. Configured ในขั้นตอนนี้ จะเป็นการกำหนดเวอร์ชั่นของชิ้นงานที่ผ่านการทดสอบแล้ว โดยถ้ามีการแก้ไขชิ้นงานที่เคยมีในระบบแล้ว ก็จะมีการออกเวอร์ชั่นใหม่

**6. MEASURES FOR SUCCESS IN TIMELY DELIVERY**

กระบวนการนี้เป็นกระบวนการที่บริษัทใช้ในการประเมินความสำเร็จของโครงการ ซึ่งสิ่งที่เป็นปัจจัยหลักที่ต้องคำนึงถึงในการประเมินคือ ความล่าช้าในการดำเนินโครงการและการส่งมอบโครงการ อาจเกิดจากสาเหตุต่างๆ หากโครงการมีความล่าช้ามากจะส่งผลให้บริษัทมีค่าใช้จ่ายที่เพิ่มมากขึ้น ซึ่งจะช่วยในการวางแผนการบริหารความเสี่ยงที่อาจจะเกิดขึ้นได้ในอนาคต ความล่าช้าจึงเป็นปัจจัยสำคัญที่ใช้ประเมินความสำเร็จของโครงการ โดยเปรียบเทียบจากตารางเวลาแผนการทำงานที่ออกแบบไว้ กับ ตารางเวลาแผนการทำงานที่ทำจริงโดยเทียบในแต่ละส่วนการทำงาน เพื่อนำผลลัพธ์ในส่วนนี้ไปเป็นตัวชี้วัดความสำเร็จในการส่งซอฟต์แวร์เชิงเวลา

มีวิธีการทำคือ สร้างตารางแบบฟอร์มเปรียบเทียบในแต่ละส่วนการทำงาน ว่าในแต่ละช่วงมีขอบเขตการทำงานอะไรบ้างที่ต้องทำให้เสร็จ พร้อมทำกำหนดใน Master Schedule ว่ามีกำหนดการเสร็จเมื่อไหร่ในทุกๆส่วนของการทำงาน และใน Actual Schedule ตอนแรกจะเว้นว่างเอาไว้ และเมื่อทำงานนั้นๆ เสร็จเมื่อไหร่ก็จะลงเวลาที่ทำเสร็จใส่ลงไปเพื่อเปรียบเทียบช่วงเวลาที่วางแผนไว้กับที่ทำเสร็จจริง ว่ามีความล่าช้าอย่างไรบ้างโดยแบบแรกจะดูในเชิงวันที่ที่เริ่มทำ และ วันที่ที่ทำเสร็จ

ตารางเวลาการวางแผนการทำงานที่ออกแบบไว้ เปรียบเทียบกับ ตารางช่วงเวลาจริงที่ทำงาน โดย

เปรียบเทียบจาก วันที่เริ่มต้น และวันที่สิ้น สุดการทำงานในแต่ละส่วนการทำงาน

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task Name** | **Master Schedule** | | **Actual Schedule** | |
| **Start** | **Finish** | **Start** | **Finish** |
| ***Planning Phase*** | | | | |
| พบลูกค้า | 4/1/2559 | 25/4/2559 | 4/1/2559 | 29/4/2559 |
| ***Requirement Phase*** | | | | |
| เก็บข้อมูล Requirement | 4/1/2559 | 15/1/2559 | 4/1/2559 | 20/1/2559 |
| จัดทำเอกสาร Requirement | 11/1/2559 | 15/1/2559 | 11/1/2559 | 21/1/2559 |
| นำเสนอ Requirement | 18/1/2559 | 22/1/2559 | 22/1/2559 | 27/1/2559 |
| กำหนดขอบเขตการทำงาน | 25/1/2559 | 29/1/2559 | 25/1/2559 | 29/1/2559 |
| ***Design Phase*** | | | | |
| ออกแบบรูปแบบการทำงาน User Interface | 1/2/2559 | 5/2/2559 | 1/2/2559 | 8/2/2559 |
| ออกแบบการทำงานฝั่ง server | 1/2/2559 | 5/2/2559 | 1/2/2559 | 8/2/2559 |
| ออกแบบการส่งข้อมูลระหว่าง Client และ Server | 8/2/2559 | 12/2/2559 | 10/2/2559 | 16/2/2559 |
| ตรวจสอบที่ออกแบบไว้กับ Requirement document | 15/2/2559 | 19/2/2559 | 15/2/2559 | 18/2/2559 |
| นำเสนอ Design document และแก้ไขข้อบกพร่อง | 22/2/2559 | 26/2/2559 | 22/2/2559 | 26/2/2559 |
| ***Implementation Phase*** | | | | |
| เขียนโปรแกรมในส่วนของผู้โดยสารและคนขับ | 29/2/2559 | 11/3/2559 | 29/2/2559 | 15/3/2559 |
| พัฒนาส่วน Server ให้สามารถเก็บข้อมูลได้ตามที่ต้องการ | 1/3/2559 | 4/3/2559 | 1/3/2559 | 4/3/2559 |
| เชื่อมต่อแอปพลิเคชันกับฐานข้อมูล และ Facebook | 14/3/2559 | 18/3/2559 | 14/3/2559 | 16/3/2559 |
| ***Unit and Integration Testing*** | | | | |
| ออกแบบการทดสอบ(ขั้นตอน วิธีการ ผลลัพธ์) | 21/3/2559 | 25/3/2559 | 21/3/2559 | 25/3/2559 |
| ออกแบบ Test case | 21/3/2559 | 25/3/2559 | 21/3/2559 | 25/3/2559 |
| ทำการทดสอบ | 28/3/2559 | 1/4/2559 | 28/3/2559 | 30/3/2559 |
| วิเคราะห์ผลการทดสอบและแก้ไขข้อผิดพลาด | 28/3/2559 | 1/4/2559 | 28/3/2559 | 31/3/2559 |
| สรุปผลการทดสอบ | 4/4/2559 | 8/4/2559 | 4/4/2559 | 8/4/2559 |
| User testing | 11/4/2559 | 15/4/2559 | 12/4/2559 | 18/4/2559 |
| สรุปผล และแก้ไขข้อผิดพลาด | 17/4/2559 | 22/4/2559 | 18/4/2559 | 22/4/2559 |
| ***Release Phase*** | | | | |
| จัดทำเอกสารต่างๆ ที่เกี่ยวข้องกับโปรแกรม | 18/4/2559 | 22/4/2559 | 18/4/2559 | 25/4/2559 |
| จัดส่งโปรแกรม | 25/4/2559 | 29/4/2559 | 27/4/2559 | 5/5/2559 |

อีกรูปแบบหนึ่งในการเปรียบเทียบระยะเวลาความล่าช้านั้น จะใช้วิธีดูจากจำนวนวันทั้งหมดที่วางแผนไว้ในส่วนการทำงานั้น คู่กับจำนวนวันทั้งหมดที่ทำส่วนการทำงานนั้นเสร็จ โดยไม่คำนึงถึงวันที่ที่เริ่มทำหรือวันที่ที่ทำเสร็จ แต่จะดูจาก % ของความล่าช้าในการทำงานแทน ซึ่งมีวิธีการคำนวณ ดังนี้

ความล่าช้าสามารถหาได้จาก \* (100%)

กำหนดว่า

* x คือ Number of days done in Actual Schedule (did not consider the delays Accepted by the user due to user’s own problem)
* y คือ Number of days done in Master Schedule

หมายเหตุ หากค่าที่คิดได้ออกมา น้อยกว่า 0 (ติดลบ) จะถือว่าไม่มี Delay (%Delay = 0%)

เกณฑ์การประเมินความสำเร็จของโครงการและคุณภาพของบริษัท

การประเมินคุณภาพของบริษัท จะทำให้ทราบว่าบริษัทมีคุณภาพอยู่ในเกณฑ์ใด นอกจากนี้จะทำให้ทราบถึงคุณภาพของพนักงาน และการบริหารจัดการงาน ซึ่งในการคำนวณคุณภาพของบริษัทจะคำนวณได้โดยการนำความสำเร็จของแต่ละโครงการมาคำนวณ โดยในการประเมินของบริษัทจะมีเกณฑ์การประเมินดังนี้

1. ความล่าช้าของแต่ละโครงการ (คิดเป็นเปอร์เซ็นต์) จะต้องไม่เกิน 20%

2. ความล่าช้าเฉลี่ยของทุกโครงการ (คิดเป็นเปอร์เซ็นต์) จะต้องไม่เกิน 10%

ตารางเวลาการวางแผนการทำงานที่ออกแบบไว้ เปรียบเทียบกับ ตารางช่วงเวลาจริงที่ทำงาน และตารางช่วงเวลาจริงที่ทำงานแต่ไม่พิจารณาช่วงที่ล่าช้าเรื่องด้วยสาเหตุจากปัญหาของลูกค้า โดยเปรียบเทียบจากจำนวนวันของการทำงาน

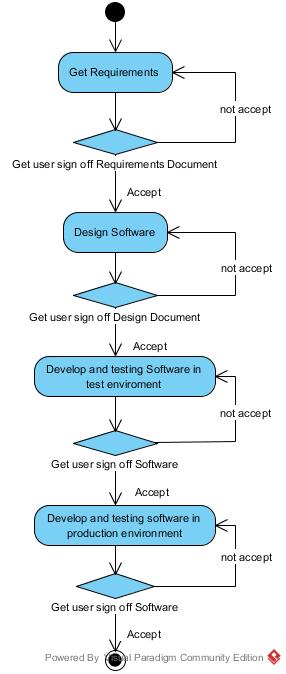
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task Name** | **Master Schedule** | **Actual Schedule** | **Actual Schedule** | **% Delay** |
| **Number of Days**  **Done** | **Number of Days**  **Done** | **(not consider delay due**  **to user’s problems)** |
| ***Planning Phase*** | | | | |
| พบลูกค้า | 20 day | 24 day | 22 day | 20% |
| ***Requirement Phase*** | | | | |
| เก็บข้อมูล Requirement | 10 day | 13 day | 12 day | 30% |
| จัดทำเอกสาร Requirement | 5 day | 9 day | 9 day | 80% |
| นำเสนอ Requirement | 5 day | 4 day | 4 day | 0% |
| กำหนดขอบเขตการทำงาน | 5 day | 5 day | 5 day | 0% |
| ***Design Phase*** | | | | |
| ออกแบบรูปแบบการทำงาน User Interface | 5 day | 6 day | 6 day | 20% |
| ออกแบบการทำงานฝั่ง server | 5 day | 6 day | 6 day | 20% |
| ออกแบบการส่งข้อมูลระหว่าง Client และ Server | 5 day | 5 day | 5 day | 0% |
| ตรวจสอบที่ออกแบบไว้กับ Requirement document | 5 day | 4 day | 4 day | 0% |
| นำเสนอ Design document และแก้ไขข้อบกพร่อง | 5 day | 5 day | 5 day | 0% |
| ***Implementation Phase*** | | | | |
| เขียนโปรแกรมในส่วนของผู้โดยสารและคนขับ | 10 day | 12 day | 12 day | 20% |
| พัฒนาส่วน Server ให้สามารถเก็บข้อมูลได้ตามที่ต้องการ | 5 day | 4 day | 4 day | 0% |
| เชื่อมต่อแอปพลิเคชันกับฐานข้อมูล และ Facebook | 5 day | 3 day | 3 day | 0% |
| ***Unit and Integration Testing*** | | | | |
| ออกแบบการทดสอบ(ขั้นตอน วิธีการ ผลลัพธ์) | 5 day | 5 day | 5 day | 0% |
| ออกแบบ Test case | 5 day | 5 day | 5 day | 0% |
| ทำการทดสอบ | 5 day | 3 day | 3 day | 0% |
| วิเคราะห์ผลการทดสอบและแก้ไขข้อผิดพลาด | 5 day | 4 day | 4 day | 0% |
| สรุปผลการทดสอบ | 5 day | 5 day | 5 day | 0% |
| User testing | 5 day | 5 day | 5 day | 0% |
| สรุปผล และแก้ไขข้อผิดพลาด | 5 day | 3 day | 3 day | 0% |
| ***Release Phase*** | | | | |
| จัดทำเอกสารต่างๆ ที่เกี่ยวข้องกับโปรแกรม | 5 day | 6 day | 6 day | 20% |
| จัดส่งโปรแกรม | 5 day | 5 day | 5 day | 0% |
| ***Total*** | **135** | **141** | **138** | **4.44%** |

เมื่อได้รับข้อมูลจากในตาราง จะทำให้ทราบว่าจากแผนที่ได้กำหนดโครงการจะสิ้นสุดเมื่อวันที่ 29 เมษยน 2559 และจากแผนการดำเนินงานจริงโครงการสิ้นสุดเมื่อวันที่ 5 พฤษภาคม 2559 ซึ่งทำให้โครงการนี้มีความล่าช้าทั้งหมด 6 วัน (ไม่นับวันหยุด) จากแผนการดำเนินงานที่ได้กำหนดไว้ใช้ระยะเวลาทั้งหมด 135 วัน จึงสามารถนำมาใช้ในการคำนวณความล่าช้าโดยคิดเป็นเปอร์เซ็นต์ ได้ดังนี้

ความล้าช้า = = 4.44%

จากการคำนวณความล่าช้าของโครงการ มีค่าเท่ากับ 4.44% ซึ่งมีค่าไม่เกิน 20% จากเกณฑ์ที่ได้กำหนดไว้ ทำให้โครงการนี้ผ่านเกณฑ์การประเมินคุณภาพของบริษัท

**7. USER ACCEPTANCE PROCESS**

 กระบวนการนี้เป็นกระบวนการเพื่อให้ลูกค้ายอมรับซอฟต์แวร์ที่บริษัทได้พัฒนาขึ้น โดยซอฟต์แวร์ที่พัฒนาจะต้องตรงตามความต้องการของลูกค้าที่ได้กำหนดไว้ และเพื่อใช้เป็นเอกสารรับรองความถูกต้องเมื่อส่งมอบโครงการ กระบวนการดังกล่าวมีขั้นตอนดังนี้

1. Get Requirements – เป็นขั้นตอนในการเก็บข้อมูล Requirement ต่างๆ ตามที่ลูกค้าต้องการ เมื่อได้รับข้อมูลครบถ้วนแล้วจะจัดทำ Requirements Document มอบให้แก่ลูกค้า เพื่อตรวจสอบความถูกต้อง
2. Get user sign off Requirements Document – เป็นขั้นตอนยืนยันความถูกต้องของ Requirements Document โดยจะมอบเอกสารนี้ให้แก่ลูกค้า แล้วให้ลูกค้าตรวจสอบว่าข้อมูลถูกต้องตามความต้องการหรือไม่ หากไม่ถูกต้องจะให้ลูกค้าระบุความผิดพลาดและทางบริษัทจะแก้ไขและจัดทำ Requirements Document เวอร์ชั่นใหม่ ให้ตรงตามความต้องการ และนำเอกสารไปให้ลูกค้าตรวจสอบอีกครั้งหนึ่ง หากถูกต้องแล้วจะให้ลูกค้าเซ็นยืนยันในแบบฟอร์มรับรองความถูกต้องของเอกสาร
3. Design Software – เป็นขั้นตอนในการออกแบบซอฟต์แวร์ตาม Requirements ที่ได้รับมา เมื่อออกแบบเสร็จเรียบร้อยแล้วจะจัดทำ Design Document มอบให้แก่ลูกค้า เพื่อตรวจสอบความถูกต้อง
4. Get user sign off Design Document – เป็นขั้นตอนยืนยันความถูกต้องของ Design Document โดยจะมอบเอกสารนี้ให้แก่ลูกค้า แล้วให้ลูกค้าตรวจสอบว่าได้ออกแบบ Software ถูกต้องตาม Requirement Document หรือไม่ หากไม่ถูกต้องจะให้ลูกค้าระบุความผิดพลาดและทางบริษัทจะออกแบบ แก้ไข Software และจัดทำ Design Document เวอร์ชั่นใหม่ ให้ตรงตามความต้องการ และนำเอกสารไปให้ลูกค้าตรวจสอบอีกครั้งหนึ่ง หากถูกต้องแล้วจะให้ลูกค้าเซ็นยืนยันในแบบฟอร์มรับรองความถูกต้องของเอกสาร
5. Develop and testing software in test environment – เป็นขั้นตอนในการพัฒนาซอฟต์แวร์ เมื่อพัฒนาแต่ละโมดูลเสร็จแล้ว จะให้ลูกค้าทดสอบความถูกต้องทีละโมดูล
6. Develop and testing software in production environment – เป็นขั้นตอนในการพัฒนาและทดสอบ Software เช่นเดียวกับ Develop and testing software in test environment แต่จะเป็นการทดสอบในสถานที่จริง
7. Get user sign off software – เป็นขั้นตอนยืนยันความถูกต้องของ Software โดยหลังจากที่ลูกค้าทดสอบ Software แล้ว จะให้ลูกค้ากรอกข้อมูลลงในแบบฟอร์มการทดสอบ Software หากลูกค้ากรอกข้อมูลไม่ยอมรับ Software บริษัทจะพัฒนาและแก้ไข Software ใหม่ให้ถูกต้องตามที่ได้รับข้อมูลจากแบบฟอร์มการทดสอบ Software และอ้างอิงจาก Requirements Document โดยจะไม่แก้ไขอะไรก็ตามที่นอกเหนือจากข้อมูล Requirements Document เมื่อแก้ไข Software เรียบร้อยแล้วจะนำ Software เวอร์ชั่นใหม่ไปให้ลูกค้าทดสอบอีกครั้ง หากลูกค้ากรอกข้อมูล และเซ็นยอมรับ Software แล้ว จะให้ลูกค้าทดสอบโมดูลต่อไปจนครบทุกโมดูล เมื่อลูกค้ายอมรับทุกโมดูล จะจัดการเก็บแบบฟอร์มการทดสอบ Software เพื่อเป็นเอกสารยืนยันเมื่อส่งมอบโครงการ

แบบฟอร์มรับรองความถูกต้องของเอกสาร

เป็นแบบฟอร์มสำหรับลูกค้า เพื่อให้ลูกค้าเซ็นยืนยันความถูกต้องของเอกสารที่ได้ตรวจสอบ โดยแบบฟอร์มนี้จะใช้สำหรับการตรวจสอบ Requirements Document และ Design Document

แบบฟอร์มรับรองความถูกต้องของเอกสาร

หมายเลขโครงการ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ชื่อโครงการ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ผู้รับผิดชอบโครงการ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

วันที่: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

ข้าพเจ้า นาย/นาง/นางสาว \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

เลขประจำตัวประชาชน/เลขหนังสือเดินทาง \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ขอรับรองเอกสาร \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ เวอร์ชั่น \_\_\_\_\_\_\_\_\_\_\_\_\_\_ว่ามีความถูกต้องและ สามารถใช้อ้างอิงในกระบวนการพัฒนาซอฟต์แวร์สาหรับโครงการนี้ได้ หากมีความผิดพลาดที่เกิดจากเอกสารนี้ที่ข้าพเจ้ารับรอง จะถือว่าเป็นความผิดพลาดของตัวข้าพเจ้า และข้าพเจ้ายินดีรับผิดชอบค่าเสียหายทั้งหมดที่เกิดขึ้น

ไม่ขอรับรองเอกสาร \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ เวอร์ชั่น \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_เนื่องจาก\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(………………………………………………………………)

ผู้รับรองเอกสาร

ข้อมูลทั่วไป

* หมายเลขโครงการ: หมายเลขของโครงการที่รับรอง
* ชื่อโครงการ: ชื่อของโครงการที่รับรอง
* ผู้รับผิดชอบโครงการ: ชื่อผู้รับผิดชอบโครงการที่รับรอง
* วันที่: วันที่เขียนแบบฟอร์มรับรองเอกสาร
* ข้าพเจ้า นาย/นาง/นางสาว: ชื่อผู้ที่รับรองเอกสาร
* เลขประจำตัวประชาชน/เลขหนังสือเดินทาง: เลขประจำตัวประชาชนหรือเลขหนังสือเดินทาง ของผู้รับรองเอกสาร
* ขอรับรองเอกสาร: ทำเครื่องหมายถูกและใส่ชื่อเอกสาร หากยืนยันและ รับรองเอกสารนี้
* ไม่ขอรับรองเอกสาร: ทำเครื่องหมายถูกใส่ชื่อเอกสาร หากไม่ยืนยันและ รับรองเอกสารนี้
* เวอร์ชั่น: เวอร์ชั่นของเอกสารที่รับรอง/ไม่รับรอง
* เนื่องจาก: กรอกเหตุผลที่ไม่รับรองเอกสารนี้
* ผู้รับรองเอกสาร: ลายเซ็นชื่อผู้ที่รับรองเอกสาร

แบบฟอร์มทดสอบซอฟต์แวร์

เป็นแบบฟอร์มสำหรับลูกค้า เพื่อให้ลูกค้าทดสอบความถูกต้องของโมดูล และเซ็นยอมรับโมดูลนั้นๆ โดยแบบฟอร์มนี้จะใช้สำหรับการทดสอบในขั้นตอน Develop and testing software in test environment และ Develop and testing software in production environment

แบบฟอร์มทดสอบซอฟต์แวร์

หมายเลขโครงการ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ วันที่: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ชื่อโครงการ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ผู้รับผิดชอบ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | รายละเอียด | ผ่าน | ไม่ผ่าน | หมายเหตุ |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

ยอมรับโมดูลนี้

ไม่ยอมรับโมดูลนี้เนื่องจาก\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(………………………………………………………………)

ผู้ทดสอบ

ชื่อโมดูล: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ เวอร์ชั่น: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ข้อมูลทั่วไป

* หมายเลขโครงการ: หมายเลขของโครงการที่ทดสอบ
* ชื่อโครงการ: ชื่อของโครงการที่ทดสอบ
* ชื่อโมดูล: ชื่อโมดูลที่ทดสอบ
* เวอร์ชั่น: เวอร์ชั่นโมดูลที่ทดสอบ
* วันที่: วันที่เขียนแบบฟอร์มทดสอบ
* ผู้รับผิดชอบ: ชื่อผู้ที่รับผิดชอบโครงการ
* ผ่าน: ทำเครื่องหมายถูก หากรายละเอียดงานในส่วนนั้นๆ ถูกต้อง
* ไม่ผ่าน: ทำเครื่องหมายถูก หากรายละเอียดงานในส่วนนั้นๆ ไม่ถูกต้อง
* หมายเหตุ: กรอกรายละเอียดที่ต้องการแก้ไขโมดูล
* ยอมรับโมดูลนี้: ทำเครื่องหมายถูก หากยอมรับโมดูลนี้
* ไม่ยอมรับโมดูลนี้: ทำเครื่องหมายถูก หากไม่ยอมรับโมดูลนี้
* เนื่องจาก: กรอกเหตุผลที่ไม่ยอมรับโมดูลนี้
* ผู้ทดสอบ: ลายเซ็นชื่อผู้ที่ทดสอบซอฟต์แวร์