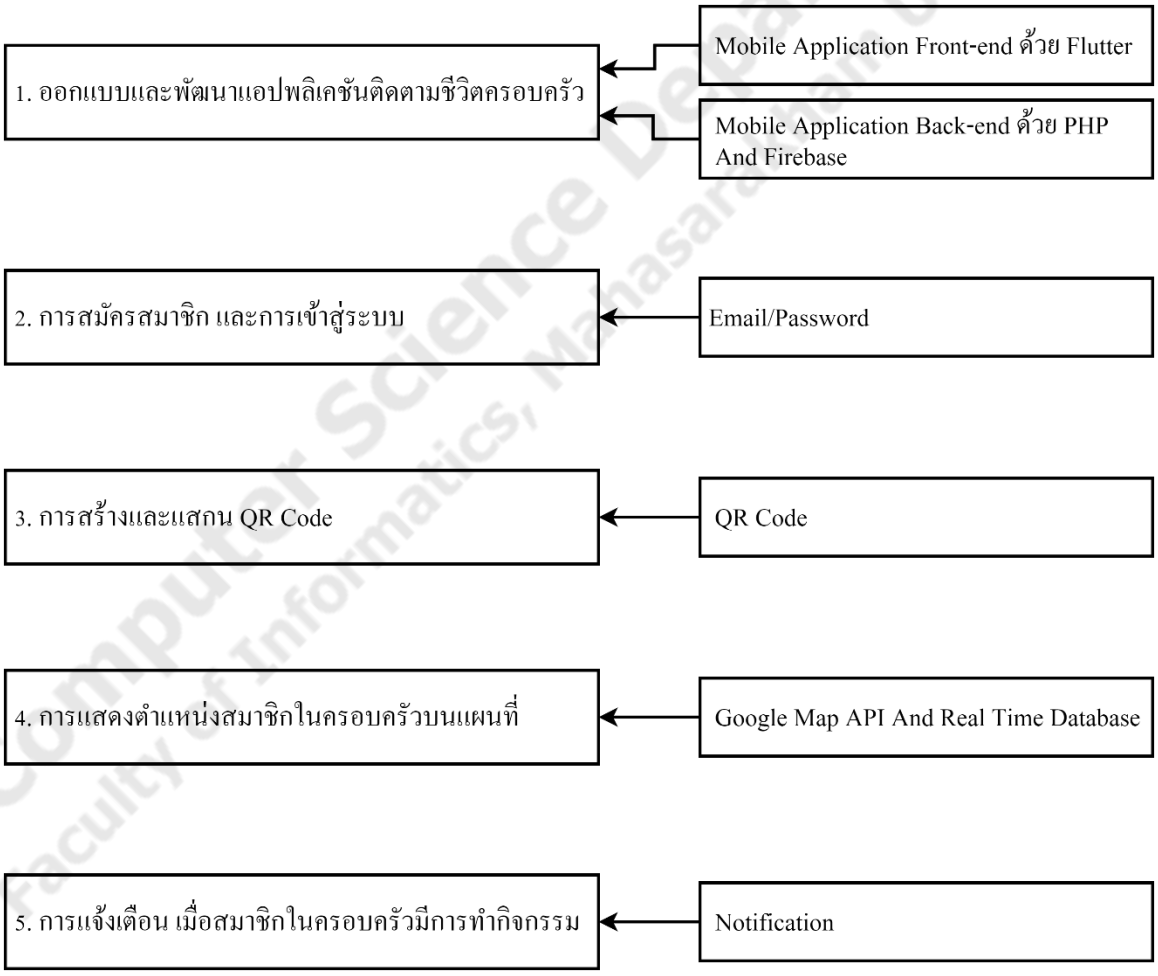
บทที่ 3

ขั้นตอนการดำเนินงาน

สำหรับในบทนี้จะกล่าวถึงขั้นตอนในการดำเนินงานของโครงงานปริญญานิพนธ์ ซึ่งทำให้ทราบถึงการวิเคราะห์และการออกแบบระบบโดยละเอียดว่ามีแนวทางในการทำงานหรือมีขั้นตอนในการทำงานของระบบอย่างไรบ้างโดยขั้นตอนในการดำเนินงานมีรายละเอียดดังนี้

3.1 กรอบการดำเนินงาน



ภาพประกอบที่ 3.1 กรอบการพัฒนา

คำอธิบาย

1. เขียนโปรแกรมออกแบบและพัฒนาแอปพลิเคชัน ติดตามชีวิตครอบครัว สำหรับผู้ใช้งานแอปพลิเคชันผ่านสมาร์ทโฟน โดยการสร้างแอปพลิเคชันเป็น 3 ส่วนหลัก ๆ ได้แก่

- แอปพลิเคชันที่ทำงานอยู่บนสมาร์ทโฟน (Front-end) โดยเครื่องมือที่ใช้ในการพัฒนาแอปพลิเคชันคือ Android Studio โดยใช้ Flutter Framework ซึ่งเป็นภาษา Dart

- แอปพลิเคชันที่ทำงานติดต่อกับฐานข้อมูล (Back end) โดยเครื่องมือที่ใช้ในการพัฒนาแอปพลิเคชันคือ phpMyAdmin ซึ่งเป็นภาษา PHP ใช้จัดการฐานข้อมูล MySQL

- แอปพลิเคชันที่ทำงานติดต่อกับฐานข้อมูล (Back end) โดยเครื่องมือที่ใช้ในการพัฒนาแอปพลิเคชันคือ Firebase ใช้จัดการฐานข้อมูลในรูปแบบ Real Time Database

2. การเข้าสู่ระบบและการสมัครสมาชิกโดยการ Authentication ของ Email/Password จาก Firebase

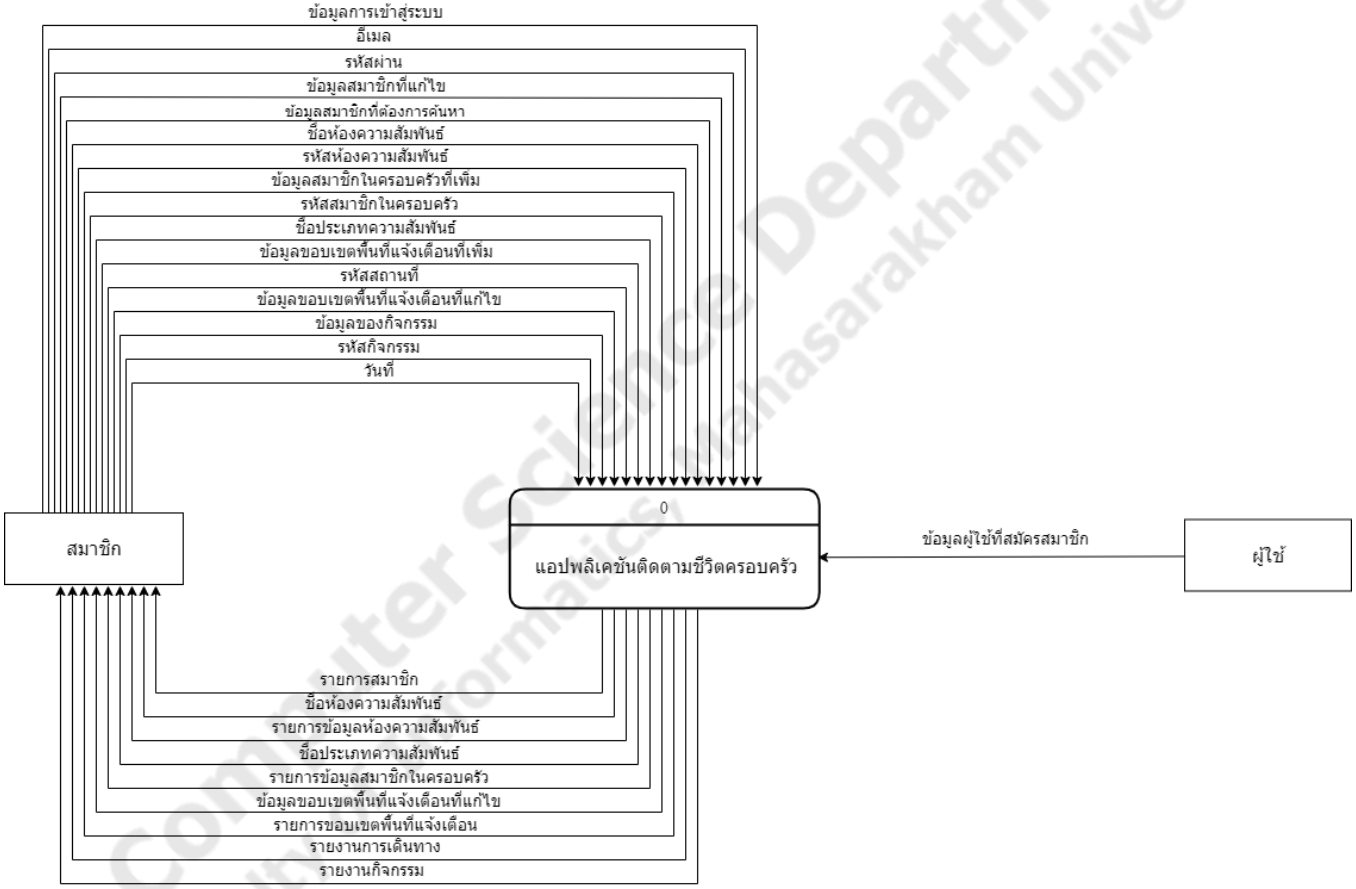
3. การสร้างและแสกน QR Code คือการสร้าง QR Code ขึ้นมาบนแอปพลิเคชัน และมีการแสกน QR Code เพื่ออ่านข้อมูลที่สร้างขึ้นบนแอปพลิเคชัน ซึ่งนำมาพัฒนา Mobile Application ในส่วนของการสร้างและการเข้าร่วมห้องความสัมพันธ์

4. การแสดงตำแหน่งสมาชิกในครอบครัวบนแผนที่ เรียกดูตำแหน่งสถานที่ของสมาชิกในครอบครัวแบบ Real Time โดยใช้ Google Map API สำหรับพัฒนา Mobile Application ไว้สำหรับเรียกใช้แผนที่ และ Sevice ต่าง ๆ ของ Google เพื่อพัฒนา Application โดยการเขียนโปรแกรมด้วย Flutter ต้องลงไลบรารี ชื่อ google_maps_flutter เพื่อเรียกใช้ Google Maps API

5. การแจ้งเตือน Notification คือการแจ้งเตือนที่แอปพลิเคชันนำข้อมูลมาแสดงในการแจ้งเตือนของระบบปฏิบัติการนั้น ๆ และนำมาพัฒนา Mobile Application ในส่วนของการทำงานที่มีการแจ้งเตือน ได้แก่ การแจ้งเตือนเมื่อสมาชิกในครอบครัวมีการทำกิจกรรม

3.2 การออกแบบระบบ

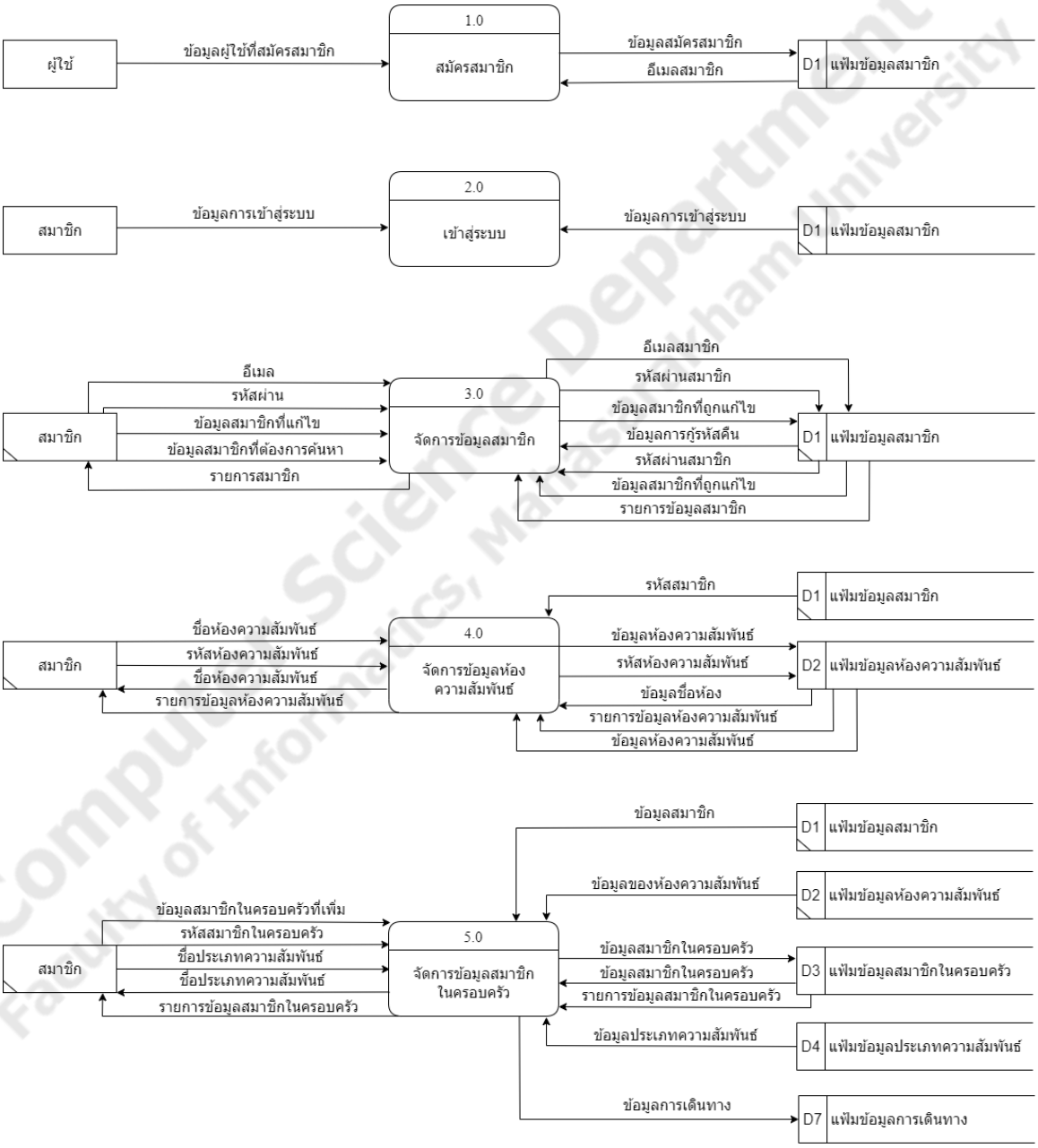
3.2.1 แผนภาพการไหลของข้อมูล (Context Diagram)



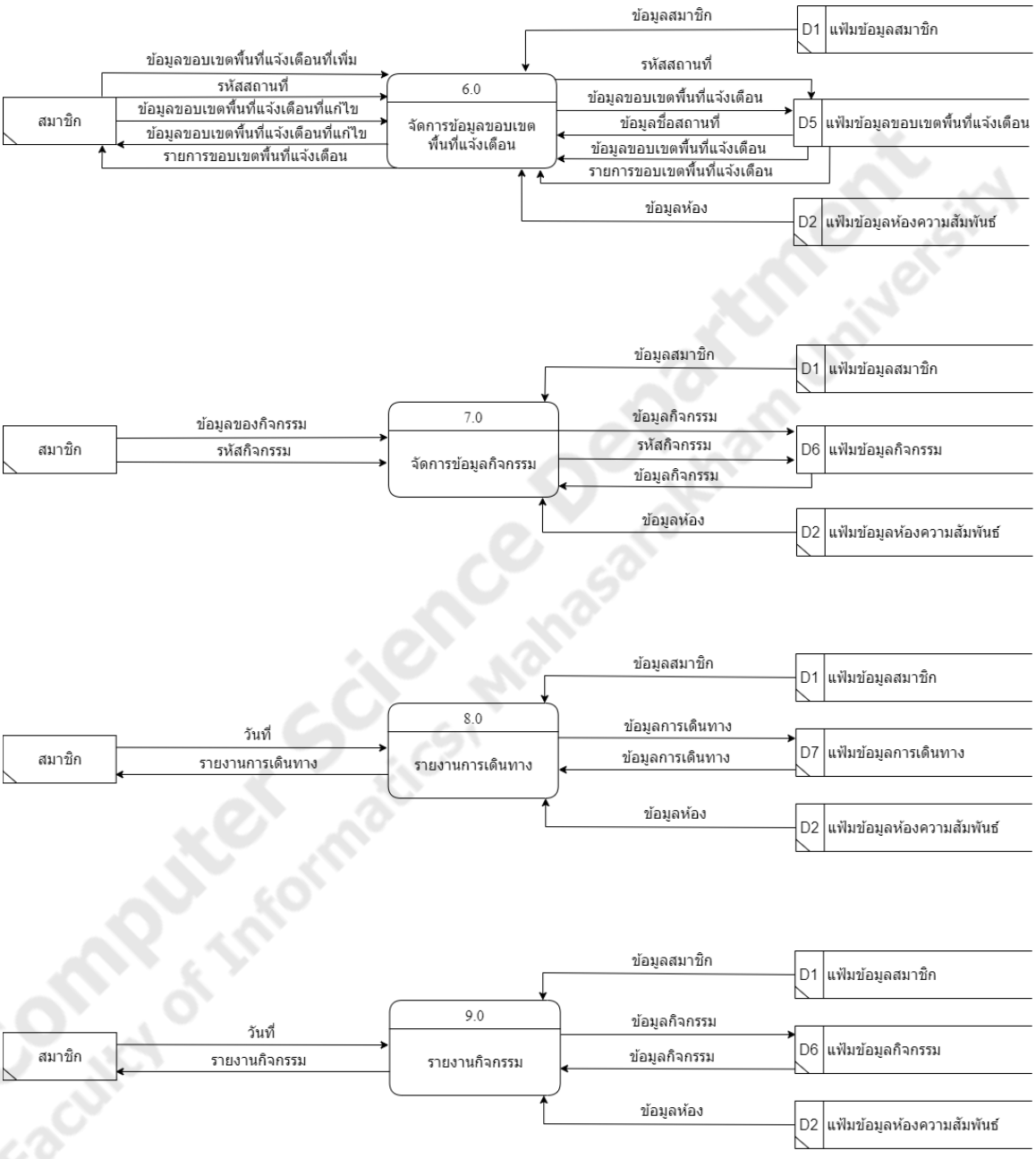
ภาพประกอบที่ 3.2 Context Diagram

3.2.2 Data Flow Diagram Level 1

แผนภาพกระแสข้อมูลในระดับที่แสดงขั้นตอนการทำงานหลักทั้งหมด (Process หลัก) ของระบบ แสดงทิศทางการไหลของ Data Flow และแสดงรายละเอียดของแหล่งจัดเก็บข้อมูล (Data Store)



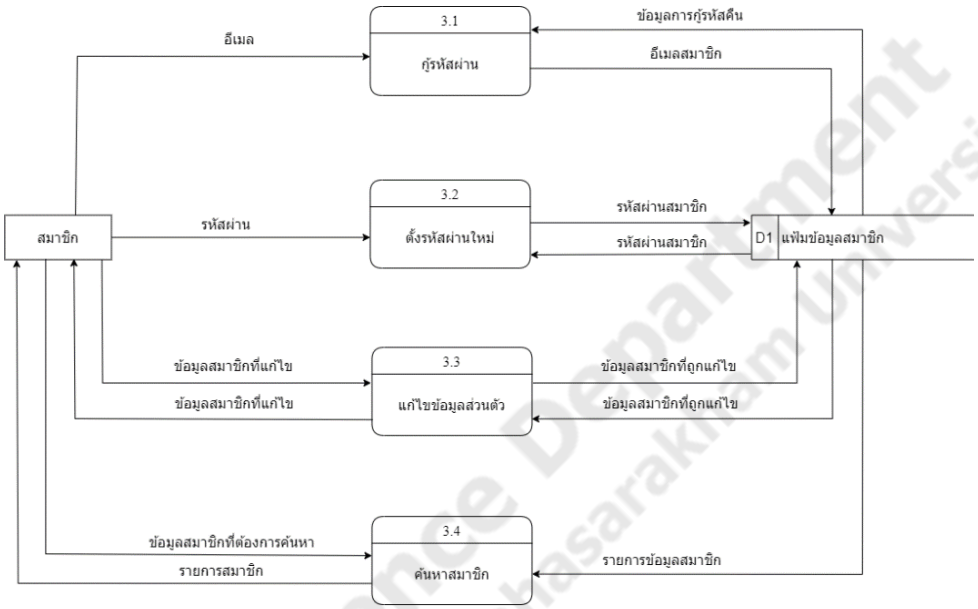
ภาพประกอบที่ 3.3 Data Flow Diagram Level 1



ภาพประกอบที่ 3.3 Data Flow Diagram Level 1 (ต่อ)

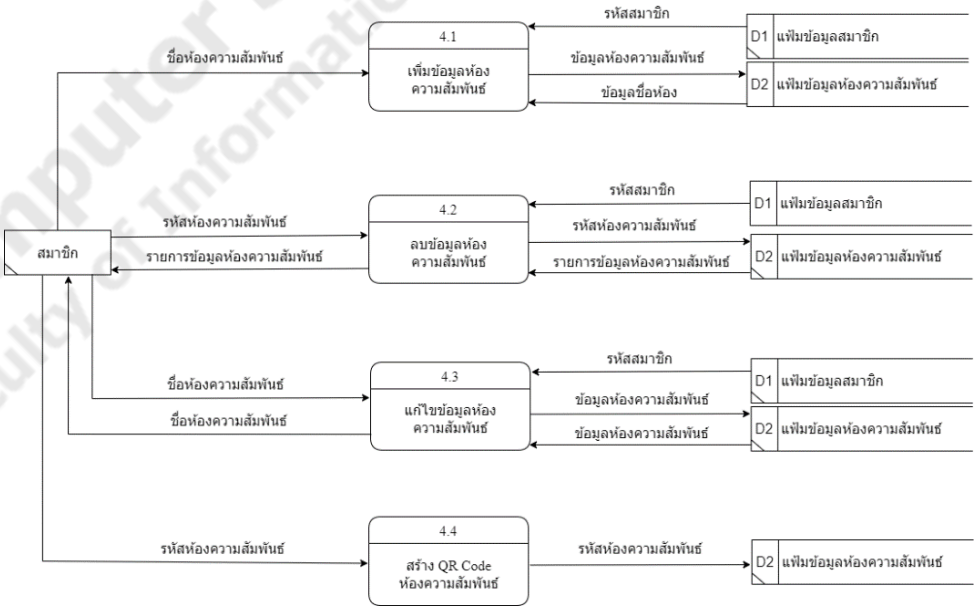
3.2.3 Data Flow Diagram Level 2

3.2.3.1 แผนภาพการไหลของข้อมูลระดับ 2 กระบวนการที่ 3



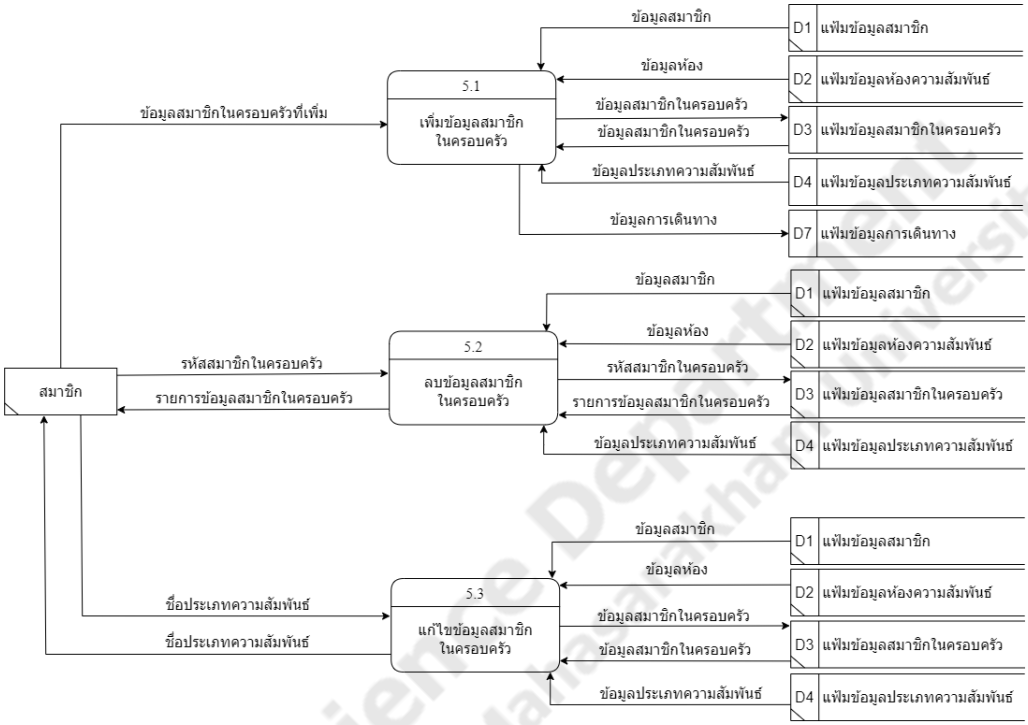
ภาพประกอบที่ 3.4 Data Flow Diagram Level 2 Process 3

3.2.3.2 แผนภาพการไหลของข้อมูลระดับ 2 กระบวนการที่ 4



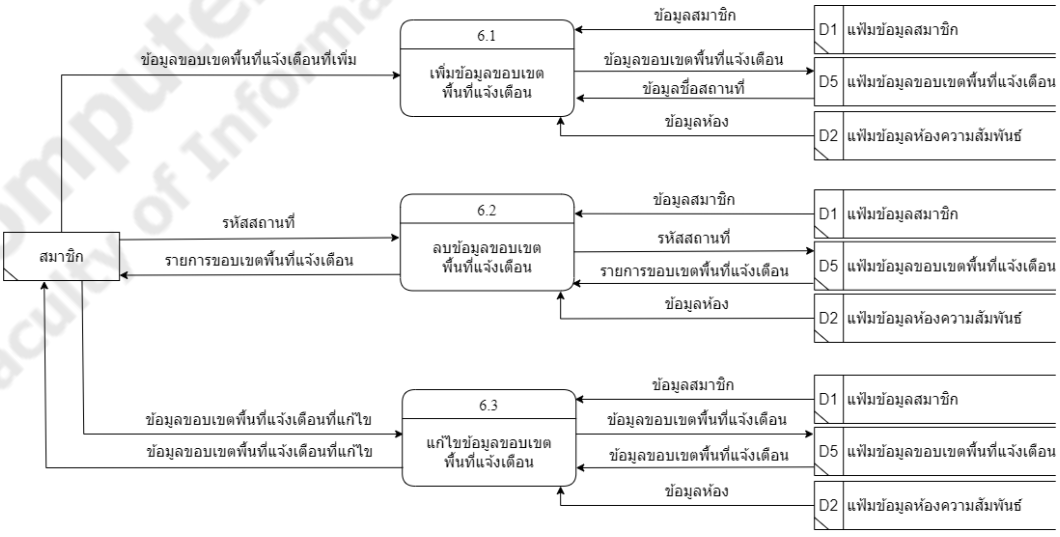
ภาพประกอบที่ 3.5 Data Flow Diagram Level 2 Process 4

3.2.3.3 แผนภาพการไหลของข้อมูลระดับ 2 กระบวนการที่ 5



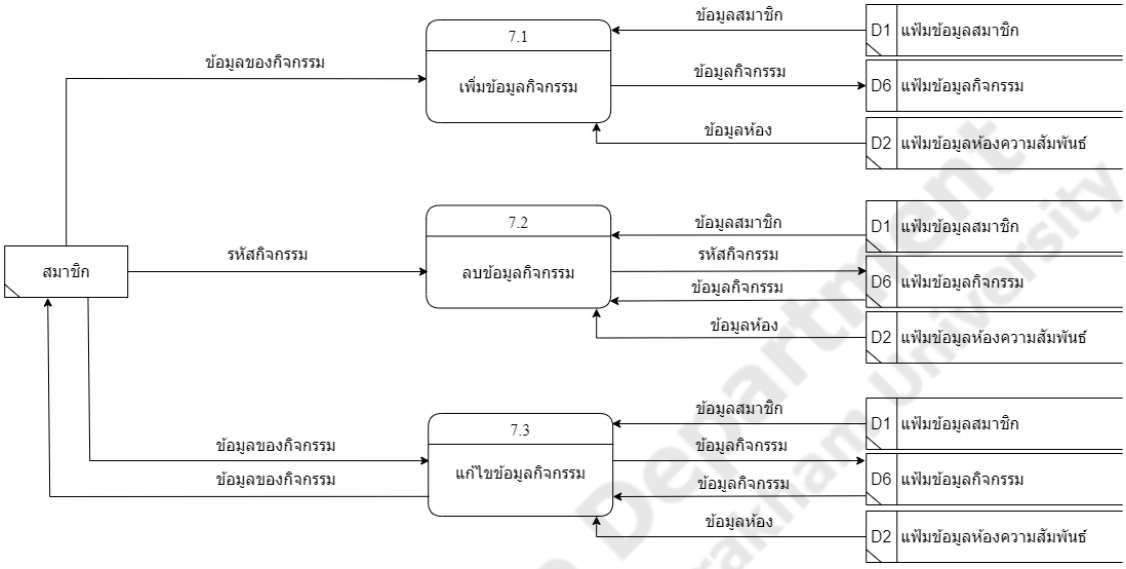
ภาพประกอบที่ 3.6 Data Flow Diagram Level 2 Process 5

3.2.3.4 แผนภาพการไหลของข้อมูลระดับ 2 กระบวนการที่ 6



ภาพประกอบที่ 3.7 Data Flow Diagram Level 2 Process 6

3.2.3.5 แผนภาพการไหลของข้อมูลระดับ 2 กระบวนการที่ 7



ภาพประกอบที่ 3.8 Data Flow Diagram Level 2 Process 7

3.3 พจนานุกรมข้อมูล (Data Dictionary)

3.3.1 External Entity Description

ตารางที่ 3.1 External Entity Description

Name	Description	Input Data Flow	Output Data flow
ผู้ใช้	ผู้ใช้งานแอปพลิเคชัน		- ข้อมูลผู้ใช้ที่สมัครสมาชิก
สมาชิก	ผู้ใช้งานแอปพลิเคชัน	- รายการสมาชิก - ชื่อห้องความสัมพันธ์ - รายการข้อมูลห้องความสัมพันธ์ - ชื่อประเภทความสัมพันธ์ - รายการข้อมูลสมาชิกในครอบครัว - ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่แก้ไข	- ข้อมูลการเข้าสู่ระบบ - อีเมล - รหัสผ่าน - ข้อมูลสมาชิกที่แก้ไข - ข้อมูลสมาชิกที่ต้องการค้นหา - ชื่อห้องความสัมพันธ์ - รหัสห้องความสัมพันธ์

ตารางที่ 3.1 External Entity Description (ต่อ)

Name	Description	Input Data Flow	Output Data flow
		- รายการขอบเขตพื้นที่แจ้งเตือน - รายงานการเดินทาง - รายงานกิจกรรม	- ข้อมูลสมาชิกในครอบครัวที่เพิ่ม - รหัสสมาชิกในครอบครัว - ชื่อประเภทความสัมพันธ์ - รหัสสถานที่ - ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่แก้ไข - ข้อมูลของกิจกรรม - รหัสกิจกรรม - วันที่

3.3.2 Data Flow Description and Data Structure

เป็นขั้นตอนการทำงานของแอปพลิเคชันซึ่งทำให้เราทราบถึงการรับ-ส่งข้อมูลแสดงถึงการไหลของข้อมูลทั้งข้อมูลเข้า (Input) และข้อมูลส่งออก (Output) ระหว่างข้อมูลต้นทางถึงข้อมูลปลายทางโดยอธิบายข้อมูลและขั้นตอนในการดำเนินงานดังต่อไปนี้

ตารางที่ 3.2 Data Flow Description and Data Structure

Name	Description	Source	Destination	Data Structure
ข้อมูลผู้ใช้ที่สมัครสมาชิก	รายละเอียดของข้อมูลการสมัคร	ผู้ใช้	Process 1.0 สมัครสมาชิก	[ชื่อ+นามสกุล+อีเมล+รหัสผ่าน+รูปภาพโปรไฟล์ | บัญชี Facebook]
ข้อมูลสมัครสมาชิก	รายละเอียดข้อมูลส่วนตัวของสมาชิก	Process 1.0 สมัครสมาชิก	D1แฟ้มข้อมูลสมาชิก	รหัสสมาชิก+ชื่อ+นามสกุล+อีเมล+รหัสผ่าน+รูปภาพโปรไฟล์+รหัสโทเค็นการแจ้งเตือน
อีเมลสมาชิก	ตรวจสอบอีเมลสมาชิก	D1แฟ้มข้อมูลสมาชิก	Process 1.0 สมัครสมาชิก	รหัสสมาชิก+อีเมล
		Process 3.1 กู้รหัสผ่าน	D1 แฟ้มข้อมูลสมาชิก	

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
ข้อมูลการเข้าสู่ระบบ	การตรวจสอบการเข้าสู่ระบบ	สมาชิก	Process 2.0 เข้าสู่ระบบ	[อีเมล+รหัสผ่าน | บัญชี Facebook]
		D1แฟ้มข้อมูลสมาชิก	Process 2.0 เข้าสู่ระบบ	
อีเมล	ตรวจสอบอีเมลของการกู้รหัส	สมาชิก	Process 3.1 กู้รหัสผ่าน	อีเมล
ข้อมูลการกู้รหัสคืน	การกู้รหัสคืนไปที่อีเมลของสมาชิก	D1 แฟ้มข้อมูลสมาชิก	Process 3.1 กู้รหัสผ่าน	อีเมล+รหัสผ่าน
รหัสผ่าน	การตั้งรหัสใหม่	สมาชิก	Process 3.2 ตั้งรหัสผ่านใหม่	รหัสผ่าน
รหัสผ่านสมาชิก	รหัสผ่านของสมาชิก	Process 3.2 ตั้งรหัสผ่านใหม่	D1 แฟ้มข้อมูลสมาชิก	รหัสสมาชิก+รหัสผ่าน
		D1 แฟ้มข้อมูลสมาชิก	สมาชิก	
ข้อมูลสมาชิกที่แก้ไข	ข้อมูลที่สมาชิกต้องการแก้ไข	สมาชิก	Process 3.3 แก้ไขข้อมูลส่วนตัว	ชื่อ+นามสกุล+รูปภาพโปรไฟล์
		Process 3.3 แก้ไขข้อมูลส่วนตัว	สมาชิก	
ข้อมูลสมาชิกที่ถูกแก้ไข	ข้อมูลสมาชิกที่ถูกแก้ไข	Process 3.3 แก้ไขข้อมูลส่วนตัว	D1 แฟ้มข้อมูลสมาชิก	รหัสสมาชิก+ชื่อ+นามสกุล+รูปภาพโปรไฟล์

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		D1 แฟ้มข้อมูลสมาชิก	Process 3.3 แก้ไขข้อมูลส่วนตัว	
ข้อมูลสมาชิกที่ต้องการค้นหา	ค้นหาสมาชิกที่ต้องการค้นหา	สมาชิก	Process 3.4 ค้นหาสมาชิก	[ชื่อ | นามสกุล]
รายการข้อมูลสมาชิก	ข้อมูลสมาชิก	D1 แฟ้มข้อมูลสมาชิก	Process 3.4 ค้นหาสมาชิก	รหัสสมาชิก+ชื่อ+นามสกุล
รายการสมาชิก	รายการของสมาชิก	Process 3.4 ค้นหาสมาชิก	สมาชิก	{ชื่อ+นามสกุล}
ชื่อห้องความสัมพันธ์	ข้อมูลชื่อของห้องความสัมพันธ์	สมาชิก	Process 4.1 เพิ่มข้อมูลห้องความสัมพันธ์	ชื่อห้องความสัมพันธ์
		สมาชิก	Process 4.3 แก้ไขข้อมูลห้องความสัมพันธ์	
		Process 4.3 แก้ไขข้อมูลห้องความสัมพันธ์	สมาชิก	
ข้อมูลชื่อห้อง	ตรวจสอบชื่อห้องความสัมพันธ์	D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 4.1 เพิ่มข้อมูลห้องความสัมพันธ์	รหัสห้องความสัมพันธ์+ชื่อห้องความสัมพันธ์
รหัสสมาชิก	รหัสสมาชิกของข้อมูลสมาชิก	D1 แฟ้มข้อมูลสมาชิก	Process 4.1 เพิ่มข้อมูลห้องความสัมพันธ์	รหัสสมาชิก

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		D1 แฟ้มข้อมูลสมาชิก	Process 4.2 ลบข้อมูลห้องความสัมพันธ์	
		D1 แฟ้มข้อมูลสมาชิก	Process 4.3 แก้ไขข้อมูลห้องความสัมพันธ์	
ข้อมูลห้องความสัมพันธ์	รายละเอียดข้อมูลห้องความสัมพันธ์	Process 4.1 เพิ่มข้อมูลห้องความสัมพันธ์	D2 แฟ้มข้อมูลห้องความสัมพันธ์	รหัสห้องความสัมพันธ์+ ชื่อห้องความสัมพันธ์+ รหัสสมาชิก
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 4.3 แก้ไขข้อมูลห้องความสัมพันธ์	
		Process 4.3 แก้ไขข้อมูลห้องความสัมพันธ์	D2 แฟ้มข้อมูลห้องความสัมพันธ์	
รายการข้อมูลห้องความสัมพันธ์	รายละเอียดของข้อมูลข้อมูลห้องความสัมพันธ์	D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 4.2 ลบข้อมูลห้องความสัมพันธ์	{ชื่อห้องความสัมพันธ์}
		Process 4.2 ลบข้อมูลห้องความสัมพันธ์	สมาชิก	
รหัสห้องความสัมพันธ์	รหัสของห้องความสัมพันธ์ที่สร้าง QR Code	สมาชิก	Process 4.2 ลบข้อมูลห้องความสัมพันธ์	รหัสห้องความสัมพันธ์
		Process 4.2 ลบข้อมูลห้อง	D2 แฟ้มข้อมูลห้องความ	

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		ความสัมพันธ์	สัมพันธ์	
		สมาชิก	Process 4.4 สร้าง QR Code ห้องความสัมพันธ์	
		Process 4.4 สร้าง QR Code ห้องความสัมพันธ์	D2 แฟ้มข้อมูลห้องความสัมพันธ์	
ข้อมูลสมาชิกในครอบครัวที่เพิ่ม	ข้อมูลสมาชิกในครอบครัวที่เพิ่ม	สมาชิก	Process 5.1 เพิ่มข้อมูลสมาชิกในครอบครัว	รหัสสมาชิก+ รหัสห้องความสัมพันธ์+ รหัสประเภทความสัมพันธ์
ข้อมูลการเดินทาง	เพิ่มข้อมูลการเดินทางเมื่อเพิ่มสมาชิกในครอบครัว	Process 5.1 เพิ่มข้อมูลสมาชิกในครอบครัว	D7 แฟ้มข้อมูลการเดินทาง	รหัสการเดินทาง+รหัสสมาชิก +ระยะทาง+วันที่เวลา+ ละติจูด+ลองจิจูด
รหัสสมาชิกในครอบครัว	สมาชิกในครอบครัวที่จะลบ	สมาชิก	Process 5.2 ลบข้อมูลสมาชิกในครอบครัว	รหัสสมาชิกในครอบครัว
		Process 5.2 ลบข้อมูลสมาชิกในครอบครัว	D3 แฟ้มข้อมูลสมาชิกในครอบครัว	

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
ชื่อประเภทความสัมพันธ์	สมาชิกในครอบครัวแก้ไขชื่อประเภทความสัมพันธ์	สมาชิก	Process 5.3 แก้ไขข้อมูลสมาชิกในครอบครัว	ชื่อประเภทความสัมพันธ์
		Process 5.3 แก้ไขข้อมูลสมาชิกในครอบครัว	สมาชิก	
ข้อมูลสมาชิก	รหัสสมาชิกชื่อและรูปโปรไฟล์ของสมาชิก	D1 แฟ้มข้อมูลสมาชิก	Process 5.1 เพิ่มข้อมูลสมาชิกในครอบครัว	รหัสสมาชิก+ชื่อ+รูปภาพโปรไฟล์
		D1 แฟ้มข้อมูลสมาชิก	Process 5.2 ลบข้อมูลสมาชิกในครอบครัว	
		D1 แฟ้มข้อมูลสมาชิก	Process 5.3 แก้ไขข้อมูลสมาชิกในครอบครัว	
		D1 แฟ้มข้อมูลสมาชิก	Process 6.1 เพิ่มข้อมูขอบเขตพื้นที่แจ้งเตือน	
		D1 แฟ้มข้อมูลสมาชิก	Process 6.2 ลบข้อมูลขอบเขตพื้นที่แจ้งเตือน	

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		D1 แฟ้มข้อมูลสมาชิก	Process 6.3 แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน	
		D1 แฟ้มข้อมูลสมาชิก	Process 7.1 เพิ่มข้อมูลกิจกรรม	
		D1 แฟ้มข้อมูลสมาชิก	Process 7.2 ลบข้อมูลกิจกรรม	
		D1 แฟ้มข้อมูลสมาชิก	Process 7.3 แก้ไขข้อมูลกิจกรรม	
		D1 แฟ้มข้อมูลสมาชิก	Process 8.0 ออกรายงานการเดินทาง	
		D1 แฟ้มข้อมูลสมาชิก	Process 9.0 ออกรายงานกิจกรรม	
ข้อมูลห้อง	รหัสและชื่อของห้องความสัมพันธ์	D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 5.1 เพิ่มข้อมูลสมาชิกในครอบครัว	รหัสห้องความสัมพันธ์+ชื่อห้องความสัมพันธ์
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 5.2 ลบข้อมูลสมาชิกในครอบครัว	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 5.3 แก้ไขข้อมูลสมาชิกในครอบครัว	

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 6.1 เพิ่มข้อมูลขอบเขตพื้นที่แจ้งเตือน	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 6.2 ลบข้อมูลขอบเขตพื้นที่แจ้งเตือน	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 6.3 แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 7.1 เพิ่มข้อมูลกิจกรรม	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 7.2 ลบข้อมูลกิจกรรม	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 7.3 แก้ไขข้อมูลกิจกรรม	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 8.0 ออกรายงานการเดินทาง	
		D2 แฟ้มข้อมูลห้องความสัมพันธ์	Process 9.0 ออกรายงานกิจกรรม	

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
ข้อมูลประเภทความสัมพันธ์	รหัสและชื่อของประเภทความสัมพันธ์	D4 แฟ้มข้อมูลประเภทความสัมพันธ์	Process 5.1 เพิ่มข้อมูลสมาชิกในครอบครัว	รหัสประเภทความสัมพันธ์+ ชื่อประเภทความสัมพันธ์
		D4 แฟ้มข้อมูลประเภทความสัมพันธ์	Process 5.2 ลบข้อมูลสมาชิกในครอบครัว	
		D4 แฟ้มข้อมูลประเภทความสัมพันธ์	Process 5.3 แก้ไขข้อมูลสมาชิกในครอบครัว	
ข้อมูลสมาชิกในครอบครัว	รายละเอียดของข้อมูลสมาชิกในครอบครัว	Process 5.1 เพิ่มข้อมูลสมาชิกในครอบครัว	D3 แฟ้มข้อมูลสมาชิกในครอบครัว	รหัสสมาชิกในครอบครัว+ รหัสสมาชิก+ รหัสห้องความสัมพันธ์+ รหัสประเภทความสัมพันธ์
		D3 แฟ้มข้อมูลสมาชิกในครอบครัว	Process 5.1 เพิ่มข้อมูลสมาชิกในครอบครัว	
		D3 แฟ้มข้อมูลสมาชิกในครอบครัว	Process 5.3 แก้ไขข้อมูลสมาชิกในครอบครัว	
		Process 5.3 แก้ไขข้อมูลสมาชิกในครอบครัว	D3 แฟ้มข้อมูลสมาชิกในครอบครัว	

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
รายการข้อมูลสมาชิกในครอบครัว	รายละเอียดของข้อมูลสมาชิกในครอบครัว	D3 แฟ้มข้อมูลสมาชิกในครอบครัว	Process 5.2 ลบข้อมูลสมาชิกในครอบครัว	{รหัสสมาชิก+ รหัสประเภทความสัมพันธ์}
		Process 5.2 ลบข้อมูลสมาชิกในครอบครัว	สมาชิก	
ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่เพิ่ม	ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่เพิ่ม	สมาชิก	Process 6.1 เพิ่มข้อมูลขอบเขตพื้นที่แจ้งเตือน	ชื่อสถานที่+ ละติจูด+ ลองจิจูด+ความยาวรัศมีของพื้นที่
ข้อมูลชื่อสถานที่	ตรวจสอบชื่อสถานที่	D5 แฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน	Process 6.1 เพิ่มข้อมูลขอบเขตพื้นที่แจ้งเตือน	รหัสสถานที่+ชื่อสถานที่
รหัสสถานที่	เลือกสถานที่ที่ต้องการลบ	สมาชิก	Process 6.2 ลบข้อมูลขอบเขตพื้นที่แจ้งเตือน	รหัสสถานที่
		Process 6.2 ลบข้อมูลขอบเขตพื้นที่แจ้งเตือน	D5 แฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน	
ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่แก้ไข	ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่แก้ไขแล้ว	สมาชิก	Process 6.3 แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน	ชื่อสถานที่+ละติจูด+ ลองจิจูด+ความยาวรัศมีของพื้นที่

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		Process 6.3 แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน	สมาชิก	
ข้อมูลขอบเขตพื้นที่แจ้งเตือน	รายละเอียดของข้อมูลขอบเขตพื้นที่แจ้งเตือน	Process 6.1 เพิ่มข้อมูลขอบเขตพื้นที่แจ้งเตือน	D5 แฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน	รหัสสถานที่+รหัสสมาชิก+รหัสห้องความสัมพันธ์+ชื่อสถานที่+ละติจูด+ลองจิจูด+ความยาวรัศมีของพื้นที่
		D5 แฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน	Process 6.3 แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน	
		Process 6.3 แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน	D5 แฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน	
รายการขอบเขตพื้นที่แจ้งเตือน	รายละเอียดของข้อมูลขอบเขตพื้นที่แจ้งเตือน	D5 แฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน	Process 6.2 ลบข้อมูลขอบเขตพื้นที่แจ้งเตือน	{ชื่อสถานที่}
		Process 6.2 ลบข้อมูลขอบเขตพื้นที่แจ้งเตือน	สมาชิก	
ข้อมูลของกิจกรรม	ข้อมูลของกิจกรรม	สมาชิก	Process 7.1 เพิ่มข้อมูลกิจกรรม	ข้อความ+อิโมจิ+วันที่เวลา

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		สมาชิก	Process 7.3 แก้ไขข้อมูลกิจกรรม	
		Process 7.3 แก้ไขข้อมูลกิจกรรม	สมาชิก	
รหัสกิจกรรม	ข้อมูลกิจกรรมที่จะลบ	สมาชิก	Process 7.2 ลบข้อมูลกิจกรรม	รหัสกิจกรรม
		Process 7.2 ลบข้อมูลกิจกรรม	D6 แฟ้มข้อมูลกิจกรรม	
ข้อมูลกิจกรรม	รายละเอียดของข้อมูลกิจกรรม	Process 7.1 เพิ่มข้อมูลกิจกรรม	D6 แฟ้มข้อมูลกิจกรรม	รหัสกิจกรรม+รหัสสมาชิก+ รหัสห้องความสัมพันธ์+ ข้อความ+อิโมจิ+ วันที่เวลา
		D6 แฟ้มข้อมูลกิจกรรม	Process 7.2 ลบข้อมูลกิจกรรม	
		D6 แฟ้มข้อมูลกิจกรรม	Process 7.3 แก้ไขข้อมูลกิจกรรม	
		Process 7.3 แก้ไขข้อมูลกิจกรรม	D6 แฟ้มข้อมูลกิจกรรม	
		Process 9.0 ออกรายงานกิจกรรม	D6 แฟ้มข้อมูลกิจกรรม	
		D6 แฟ้มข้อมูลกิจกรรม	Process 9.0 ออกรายงานกิจกรรม	
วันที่	เลือกวันที่	สมาชิก	Process 8.0 ออกรายงานการเดินทาง	วันที่เวลา

ตารางที่ 3.2 Data Flow Description and Data Structure (ต่อ)

Name	Description	Source	Destination	Data Structure
		สมาชิก	Process 9.0 ออกรายงานกิจกรรม	
ข้อมูลการเดินทาง	ข้อมูลการเดินทาง	Process 8.0 ออกรายงานการเดินทาง	D7 แฟ้มข้อมูลการเดินทาง	รหัสการเดินทาง+รหัสสมาชิก+รหัสห้องความสัมพันธ์+ระยะทาง+วันที่เวลา+ละติจูด+ลองจิจูด
		D7 แฟ้มข้อมูลการเดินทาง	Process 8.0 ออกรายงานการเดินทาง	
รายงานการเดินทาง	รายงานการเดินทาง	Process 8.0 ออกรายงานเดินทาง	สมาชิก	{ระยะทาง+วันที่เวลา+ละติจูด+ลองจิจูด}
รายงานกิจกรรม	รายงานกิจกรรม	Process 9.0 ออกรายงานกิจกรรม	สมาชิก	{ข้อความ+อิโมจิ+วันที่เวลา}

3.3.3 Data Store Description and Data Structure

คือการนำเสนอข้อมูลเข้าไปจัดการในฐานข้อมูล โดยมีการแยกออกเป็นแฟ้มข้อมูล

ตารางที่ 3.3 Data Store Description and Data Structure

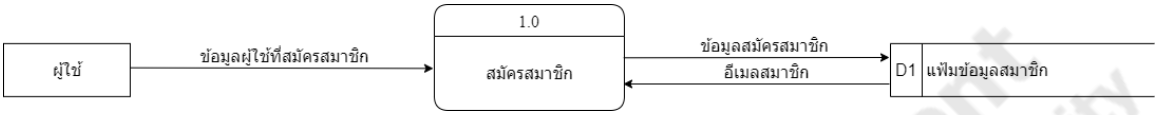
ID	Data Store Name	Description	Data Structure
D1	ข้อมูลสมาชิก	เป็นที่เก็บข้อมูลสมาชิก	<u>รหัสสมาชิก</u> + ชื่อ + นามสกุล + อีเมล + รหัสผ่าน + รูปภาพโปรไฟล์ + สถานะบัญชี + รหัสไฟร์เบส + รหัสโทเค็นการแจ้งเตือน + บัญชีเฟซบุ๊ก

ตารางที่ 3.3 Data Store Description and Data Structure (ต่อ)

ID	Data Store Name	Description	Data Structure
D2	ข้อมูลห้องความสัมพันธ์	เป็นที่เก็บข้อมูลห้องความสัมพันธ์	รหัสห้องความสัมพันธ์ + รหัสสมาชิก + ชื่อห้องความสัมพันธ์
D3	ข้อมูลสมาชิกในครอบครัว	เป็นที่เก็บข้อมูลสมาชิกในครอบครัว	รหัสสมาชิกในครอบครัว+ รหัสสมาชิก + รหัสห้องความสัมพันธ์ + รหัสประเภทความสัมพันธ์
D4	ข้อมูลประเภทความสัมพันธ์	เป็นที่เก็บข้อมูลประเภทความสัมพันธ์	รหัสประเภทความสัมพันธ์ + ชื่อประเภทความสัมพันธ์
D5	ข้อมูลขอบเขตพื้นที่แจ้งเตือน	เป็นที่เก็บข้อมูลขอบเขตพื้นที่แจ้งเตือน	รหัสสถานที่ + รหัสสมาชิก + รหัสห้องความสัมพันธ์ + ชื่อสถานที่ + ละติจูด + ลองจิจูด + ความยาวรัศมีของพื้นที่
D6	ข้อมูลกิจกรรม	เป็นที่เก็บข้อมูลกิจกรรม	รหัสกิจกรรม + รหัสสมาชิก + รหัสห้องความสัมพันธ์ + ข้อความ + อิโมจิ + วันที่เวลา
D7	ข้อมูลการเดินทาง	เป็นที่เก็บข้อมูลการเดินทาง	รหัสการเดินทาง + รหัสสมาชิก + รหัสห้องความสัมพันธ์ + ระยะทาง + วันที่เวลา + ละติจูด + ลองจิจูด

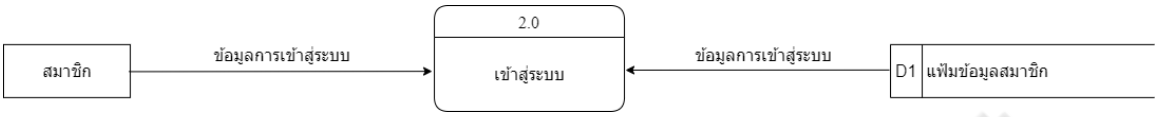
3.3.4 Process Description

3.3.4.1 Process 1.0 สมัครสมาชิก



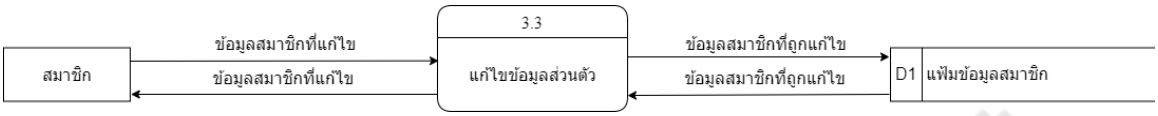
ID	1.0
Name	สมัครสมาชิก
Description	ผู้ใช้สมัครสมาชิกเพื่อเป็นสมาชิก
Input Data Flow	- ข้อมูลผู้ใช้ที่สมัครสมาชิก - อีเมลสมาชิก
Output Data Flow	- ข้อมูลสมัครสมาชิก
Process Description	เริ่มต้น 1. รับข้อมูลที่ผู้ใช้สมัครสมาชิก 2. ตรวจสอบข้อมูลสมัครสมาชิกว่าครบถ้วนและถูกต้องหรือไม่ 2.1 ถ้าข้อมูลครบถ้วนและถูกต้องทุกรายการ ไปที่ข้อ 3. 2.2 ถ้าข้อมูลสมัครสมาชิกครบถ้วนแต่อีเมลไม่ถูกต้องให้แสดงข้อความ “รูปแบบอีเมลไม่ถูกต้อง” กลับไปข้อ 1. 2.3 ถ้ารหัสผ่านไม่ครบ 6 ตัวอักษรให้แสดงข้อความ “รหัสผ่านต้องมีอย่างน้อย 6 ตัวอักษร” กลับไปข้อ 1. 3. ตรวจสอบในแฟ้มข้อมูลสมาชิกว่ามีข้อมูลอีเมลแล้วหรือไม่ 3.1 ถ้ามีข้อมูลอีเมลแล้วให้แสดงข้อความ “อีเมลนี้มีอยู่ในระบบแล้ว” กลับไปที่ข้อ 1. 3.2 ถ้าไม่มีข้อมูลอีเมลซ้ำในระบบให้เพิ่มข้อมูลสมัครสมาชิกลงในแฟ้มข้อมูลสมาชิกและแสดงข้อความ “สมัครสมาชิกสำเร็จ” จบการทำงาน

3.3.4.2 Process 2.0 เข้าสู่ระบบ



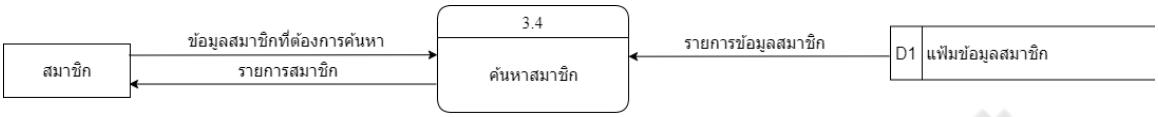
ID	2.0
Name	เข้าสู่ระบบ
Description	เข้าสู่ระบบเพื่อใช้งานแอปพลิเคชัน
Input Data Flow	- ข้อมูลการเข้าสู่ระบบ
Output Data Flow	- ข้อมูลการเข้าสู่ระบบ
Process Description	เริ่มต้น 1. รับข้อมูลการเข้าสู่ระบบ (อีเมล และ รหัสผ่าน) 2. ตรวจสอบข้อมูลการเข้าสู่ระบบว่ามีข้อมูลที่กรอกครบถ้วนถูกต้องหรือไม่ 2.1 ถ้าป้อนข้อมูลการเข้าสู่ระบบครบถ้วน ไปที่ข้อ 3. 2.2 ถ้าป้อนข้อมูลช่องอีเมลไม่ครบ ให้แสดงข้อความ “กรอกอีเมล” กลับไปข้อ 1. 3. ตรวจสอบข้อมูลการเข้าสู่ระบบในแฟ้มข้อมูลสมาชิก 3.1 ถ้ามีข้อมูลการเข้าสู่ระบบในแฟ้มข้อมูลสมาชิก ให้แสดงข้อความ “เข้าสู่ระบบสำเร็จ” 3.2 ถ้าไม่มีข้อมูลการเข้าสู่ระบบในแฟ้มข้อมูลสมาชิก ให้แสดงข้อความ “ข้อมูลไม่ถูกต้อง โปรดตรวจสอบ อีเมล และ รหัสผ่าน ใหม่อีกครั้ง” กลับไปที่ข้อ 1. จบการทำงาน

3.3.4.3 Process 3.3 แก้ไขข้อมูลส่วนตัว



ID	3.3
Name	แก้ไขข้อมูลส่วนตัว
Description	แก้ไขข้อมูลส่วนตัวที่ต้องการแก้ไข
Input Data Flow	- ข้อมูลสมาชิกที่แก้ไข - ข้อมูลสมาชิกที่ถูกแก้ไข
Output Data Flow	- ข้อมูลสมาชิกที่ถูกแก้ไข - ข้อมูลสมาชิกที่แก้ไข
Process Description	เริ่มต้น 1. สมาชิกป้อนข้อมูลสมาชิกที่ต้องการแก้ไข 2. ตรวจสอบว่าข้อมูลครบถ้วนและถูกต้องหรือไม่ 2.1 ถ้าข้อมูลสมาชิกครบถ้วน และถูกต้องทุกรายการให้บันทึกข้อมูลสมาชิกลงในแฟ้มข้อมูลสมาชิก และให้แสดงข้อความ “แก้ไขข้อมูลส่วนตัวสำเร็จ” 2.2 ถ้าข้อมูลไม่ครบ ให้แสดงข้อความ “กรุณาป้อนชื่อและนามสกุล” กลับไปข้อ 1. จบการทำงาน

3.3.4.4 Process 3.4 ค้นหาสมาชิก

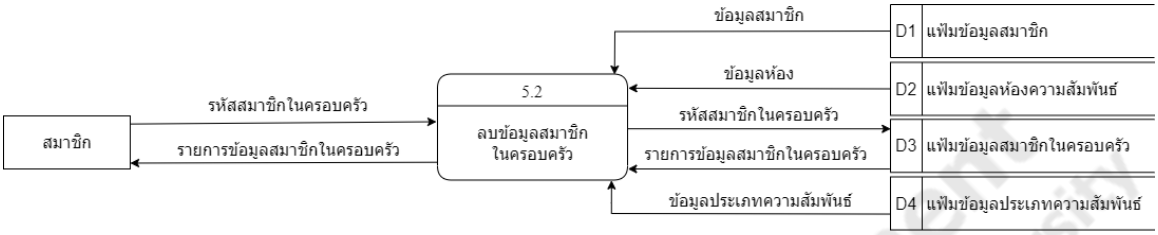


ID	3.4
Name	ค้นหาสมาชิก
Description	การค้นหาสมาชิก
Input Data Flow	- ข้อมูลสมาชิกที่ต้องการค้นหา - รายการข้อมูลสมาชิก
Output Data Flow	- รายการสมาชิก
Process Description	เริ่มต้น 1. รับข้อมูลสมาชิกที่ต้องการค้นหา (ชื่อ) 2. ค้นหาข้อมูลสมาชิก จากแฟ้มข้อมูลสมาชิก ตามข้อมูลที่รับเข้ามา 3. ตรวจสอบข้อมูลสมาชิกว่ามีข้อมูลสมาชิกหรือไม่ 3.1 ถ้ามีข้อมูลสมาชิก แสดงรายการสมาชิก 3.2 ถ้าไม่มีข้อมูลสมาชิก ระบบจะแสดงข้อความ “ไม่พบข้อสมาชิก” กลับไปข้อ 1. จบการทำงาน

3.3.4.5 Process 4.1 เพิ่มข้อมูลห้องความสัมพันธ์ในครอบครัว

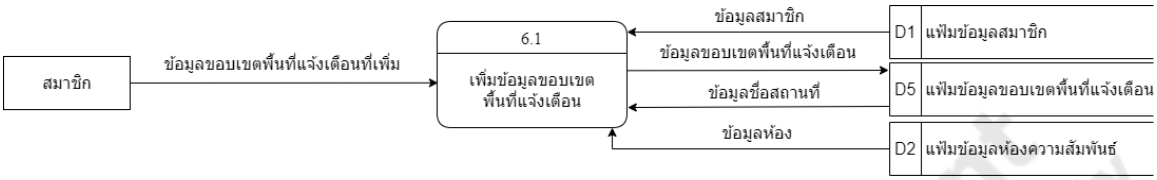
ID	4.1
Name	เพิ่มข้อมูลห้องความสัมพันธ์
Description	การสร้างห้องความสัมพันธ์ของครอบครัว
Input Data Flow	- ข้อมูลห้องความสัมพันธ์ที่เพิ่ม - ข้อมูลชื่อห้อง - รหัสสมาชิก
Output Data Flow	- ข้อมูลห้องความสัมพันธ์
Process Description	เริ่มต้น 1. รับข้อมูลห้องความสัมพันธ์ที่ต้องการเพิ่ม 2. ตรวจสอบข้อมูลห้องความสัมพันธ์ว่าครบถ้วนและถูกต้องหรือไม่ 2.1 ถ้าป้อนข้อมูลห้องความสัมพันธ์ครบถ้วนและถูกต้อง ไปที่ข้อ 3. 2.2 ถ้าข้อมูลห้องความสัมพันธ์ไม่ครบให้แสดงข้อความ “กรุณาป้อนชื่อห้อง” กลับไปข้อ 1. 3. บันทึกข้อมูลห้องความสัมพันธ์ ลงในแฟ้มข้อมูลห้องความสัมพันธ์ และให้แสดงข้อความ “เพิ่มห้องความสัมพันธ์สำเร็จ” จบการทำงาน

3.3.4.6 Process 5.2 ลบข้อมูลสมาชิกในครอบครัว



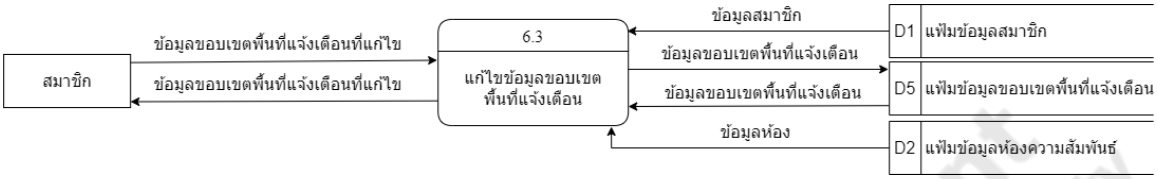
ID	5.2
Name	ลบข้อมูลสมาชิกในครอบครัว
Description	การลบสมาชิกในครอบครัวที่ต้องการลบ
Input Data Flow	- รหัสสมาชิกในครอบครัว - ข้อมูลสมาชิก - ข้อมูลห้อง - ข้อมูลประเภทความสัมพันธ์ - รายการข้อมูลสมาชิกในครอบครัว
Output Data Flow	- รายการข้อมูลสมาชิกในครอบครัว - รหัสสมาชิกในครอบครัว
Process Description	เริ่มต้น 1. แสดงรายการข้อมูลสมาชิกในครอบครัว 2. เลือกสมาชิกในครอบครัวที่ต้องการลบ 3. แสดงหน้ายืนยันการลบ (ยกเลิก และ ตกลง) 3.1 ถ้าตกลง ให้ทำการลบสมาชิกในครอบครัวออกจากลบแฟ้มข้อมูลสมาชิกในครอบครัว และให้แสดงข้อความ “ลบ (ชื่อ) ออกจากห้องแล้ว” 3.2 ถ้ายกเลิก ให้กลับไปข้อ 2. จบการทำงาน

3.3.4.7 Process 6.1 เพิ่มข้อมูลขอบเขตพื้นที่แจ้งเตือน



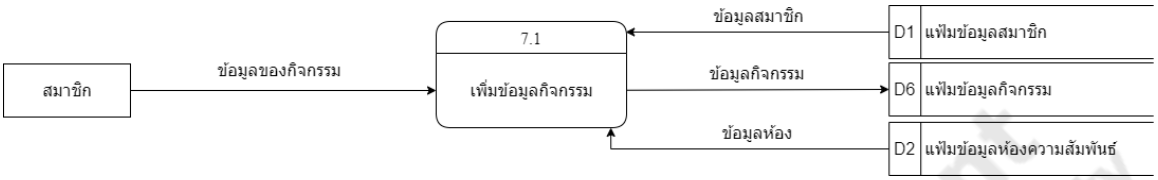
ID	6.1
Name	เพิ่มข้อมูลขอบเขตพื้นที่แจ้งเตือน
Description	เพิ่มข้อมูลขอบเขตพื้นที่แจ้งเตือนที่ต้องการเพิ่ม
Input Data Flow	- ข้อมูลสมาชิก - ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่เพิ่ม - ข้อมูลชื่อสถานที่ - ข้อมูลห้อง
Output Data Flow	- ข้อมูลขอบเขตพื้นที่แจ้งเตือน
Process Description	เริ่มต้น 1. รับข้อมูลขอบเขตพื้นที่แจ้งเตือนที่ต้องการเพิ่ม 2. ตรวจสอบข้อมูลขอบเขตพื้นที่แจ้งเตือนว่าครบถ้วน และถูกต้องหรือไม่ 2.1 ถ้าข้อมูลขอบเขตพื้นที่แจ้งเตือนครบถ้วนและถูกต้อง ไปที่ข้อ 3. 2.2 ถ้าข้อมูลขอบเขตพื้นที่แจ้งเตือนไม่ครบให้แสดงข้อความ “กรุณาป้อนข้อมูลขอบเขตพื้นที่แจ้งเตือนให้ครบ” กลับไปข้อ 1. 3. บันทึกขอบเขตพื้นที่แจ้งเตือนลงในแฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน และให้แสดงข้อความ “บันทึกสถานที่สำเร็จ” จบการทำงาน

3.3.4.8 Process 6.3 แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน



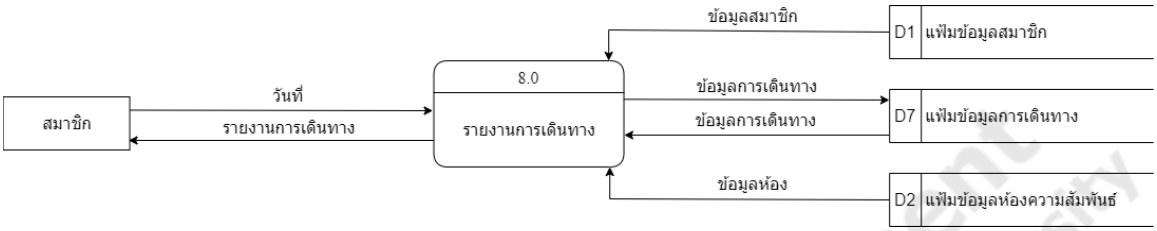
ID	6.3
Name	แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือน
Description	แก้ไขข้อมูลขอบเขตพื้นที่แจ้งเตือนที่ต้องการแก้ไข
Input Data Flow	- ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่แก้ไข - ข้อมูลขอบเขตพื้นที่แจ้งเตือน - ข้อมูลสมาชิก - ข้อมูลห้อง
Output Data Flow	- ข้อมูลขอบเขตพื้นที่แจ้งเตือนที่แก้ไข - ข้อมูลขอบเขตพื้นที่แจ้งเตือน
Process Description	เริ่มต้น 1. สมาชิกป้อนข้อมูลขอบเขตพื้นที่แจ้งเตือนที่ต้องการแก้ไข 2. ตรวจสอบว่าข้อมูลครบถ้วนและถูกต้องหรือไม่ 2.1 ถ้าข้อมูลขอบเขตพื้นที่แจ้งเตือนครบถ้วน และถูกต้องทุกรายการให้บันทึกข้อมูลขอบเขตพื้นที่แจ้งเตือนลงในแฟ้มข้อมูลขอบเขตพื้นที่แจ้งเตือน และให้แสดงข้อความ “แก้ไขสถานที่สำเร็จ” 2.2 ถ้าข้อมูลไม่ครบให้แสดงข้อความ “กรุณาป้อนชื่อสถานที” กลับไปข้อ 1 จบการทำงาน

3.3.4.9 Process 7.1 เพิ่มข้อมูลกิจกรรม



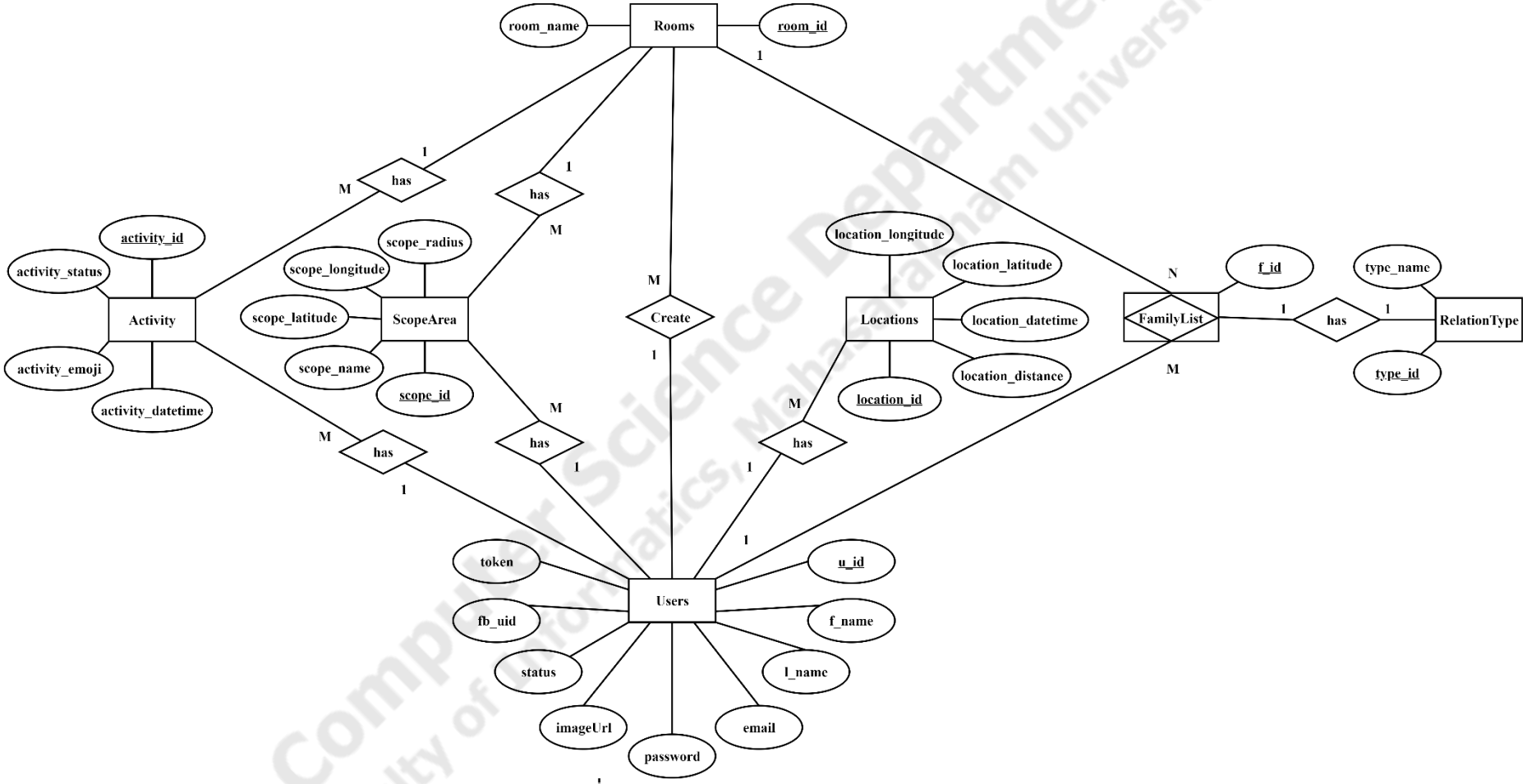
ID	7.1
Name	เพิ่มข้อมูลกิจกรรม
Description	การเพิ่มข้อมูลกิจกรรมให้สมาชิกในห้องความสัมพันธ์ได้รับรู้
Input Data Flow	- ข้อมูลสมาชิก - ข้อมูลของกิจกรรม - ข้อมูลห้อง
Output Data Flow	- ข้อมูลกิจกรรม
Process Description	เริ่มต้น 1. รับข้อมูลของกิจกรรมที่ต้องการเพิ่ม 2. ตรวจสอบข้อมูลของกิจกรรมว่าครบถ้วนและถูกต้องหรือไม่ 2.1 ถ้าข้อมูลกิจกรรมครบ กดบันทึกจะเพิ่มข้อมูลกิจกรรมลงในแฟ้มข้อมูลกิจกรรม และให้แสดงข้อความ “เพิ่มกิจกรรมสำเร็จ” 2.2 ถ้าข้อมูลกิจกรรมครบแต่ป้อนเกิน 30 ตัวอักษร จะแสดงข้อความ “กรุณาป้อนกิจกรรมไม่เกิน 30 ตัวอักษร” กลับไปข้อ 1. จบการทำงาน

3.3.4.10 Process 8.0 ออกรายงานการเดินทาง



ID	8.0
Name	ออกรายงานการเดินทาง
Description	การดูประวัติการเดินทางว่าใน 1 วันเดินทางไปไหนมาบ้าง
Input Data Flow	- ข้อมูลสมาชิก - วันที่ - ข้อมูลการเดินทาง - ข้อมูลห้อง
Output Data Flow	- รายงานการเดินทาง - ข้อมูลการเดินทาง
Process Description	เริ่มต้น 1. รับวันที่ที่ต้องการค้นหารายงานการเดินทาง 2. ตรวจสอบวันที่ว่ามีข้อมูลการเดินทางในแฟ้มข้อมูลการเดินทางหรือไม่ 2.1 ถ้าวันที่นั้นมีข้อมูลการเดินทาง ระบบจะแสดงรายงานการเดินทาง 2.2 ถ้าวันที่นั้นไม่มีข้อมูลการเดินทาง ให้ระบบแสดงข้อความ ”ไม่มีการเดินทาง” กลับไปข้อ 1. จบการทำงาน

3.4 ความสัมพันธ์ (Entity Relationship Diagram)



ภาพประกอบที่ 3.9 Entity Relationship Diagram

3.5 การออกแบบฐานข้อมูล (Database Design)

ตารางที่ 3.4 ข้อมูลสมาชิก (Users)

Id	Column	Type	Description	Example Data	Constraints
1	u_id	int	รหัสสมาชิก	1	PK
2	f_name	varchar (100)	ชื่อ	อาทิตย์	Not null
3	l_name	varchar (100)	นามสกุล	ทวีบท	Not null
4	email	varchar (100)	อีเมล	Artsshit7 @gmail.com	Not null
5	password	varchar (100)	รหัสผ่าน	Art12345	Not null
6	imageUrl	varchar (1000)	ภาพโปรไฟล์	A1.png	Null
7	status	int	สถานะบัญชี	1	Not null
8	fb_uid	varchar (100)	รหัสสมาชิก ของไฟร์เบส	C3KYAiXN6jhC QJL5k0KK4wPN 5qY2	Not null
9	token	varchar(1000)	รหัสโทเค็น การแจ้งเตือน	e4NcOBYCTma dub3Ws5fwv:A PA91bFARGN_ NsDAPRPee37fl IS1I0WSlGAT9D dHy3ERP8tVGJ Cqo5fjw8zMUC WJYqgX1kJF0p sr37IuyBS7bDU Z6x6KLussc20h j4C6DYDmEpK1 8BQGNaZaQ_Y	Not null

ตารางที่ 3.5 ข้อมูลห้องความสัมพันธ์ (Rooms)

Id	Column	Type	Description	Example Data	Constraints
1	room_id	int	รหัสห้องความสัมพันธ์	1	PK
2	room_name	varchar (100)	ชื่อห้องความสัมพันธ์	friends	Not null
3	u_id	int	รหัสสมาชิก	1	FK reference from Users (u_id) ON DELETE CASCADE ON UPDATE CASCADE

ตารางที่ 3.6 ข้อมูลประเภทความสัมพันธ์ (RelationType)

Id	Column	Type	Description	Example Data	Constraints
1	type_id	int	รหัสประเภทความสัมพันธ์	17	PK
2	type_name	varchar (100)	ชื่อประเภทความสัมพันธ์	เพื่อน	Not null

ตารางที่ 3.7 ข้อมูลสมาชิกในครอบครัว (FamilyList)

Id	Column	Type	Description	Example Data	Constraints
1	f_id	int	รหัสสมาชิกในครอบครัว	1	PK

ตารางที่ 3.7 ข้อมูลสมาชิกในครอบครัว (FamilyList) (ต่อ)

Id	Column	Type	Description	Example Data	Constraints
2	u_id	int	รหัสสมาชิก	1	FK reference from Users (u_id) ON DELETE CASCADE ON UPDATE CASCADE
3	room_id	int	รหัสห้องความสัมพันธ์	1	FK reference from Rooms (room_id) ON DELETE CASCADE ON UPDATE CASCADE
4	type_id	int	รหัสประเภทความสัมพันธ์	17	FK reference from RelationType (type_id) ON DELETE SET NULL ON UPDATE SET NULL

ตารางที่ 3.8 ข้อมูลขอบเขตพื้นที่แจ้งเตือน (ScopeArea)

Id	Column	Type	Description	Example Data	Constraints
1	scope_id	int	รหัสสถานที่	1	PK
2	scope_name	varchar (100)	ชื่อสถานที่	บ้าน	Not null
3	scope_latitude	double	ละติจูด	16.2931	Not null
4	scope_longitude	double	ลองจิจูด	102.6157	Not null
5	scope_radius	varchar (100)	ความยาวรัศมีของพื้นที่	100	Not null
6	u_id	int	รหัสสมาชิก	1	FK reference from Users (u_id) ON DELETE CASCADE ON UPDATE CASCADE
7	room_id	int	รหัสห้องความสัมพันธ์	1	FK reference from Rooms (room_id) ON DELETE CASCADE ON UPDATE CASCADE

ตารางที่ 3.9 ข้อมูลกิจกรรม (Activity)

Id	Column	Type	Description	Example Data	Constraints
1	activity_id	int	รหัสกิจกรรม	1	PK

ตารางที่ 3.9 ข้อมูลกิจกรรม (Activity) (ต่อ)

Id	Column	Type	Description	Example Data	Constraints
2	activity_status	varchar (100)	ข้อความ	เรียน Mobile	Not null
3	activity_emoji	varchar (100)	อิโมจิ	:)	Not null
4	activity_datetime	datetime	วันที่เวลา	10/12/2022 13:00:00	Not null
5	u_id	int	รหัสสมาชิก	1	FK reference from Users (u_id) ON DELETE CASCADE ON UPDATE CASCADE
6	room_id	int	รหัสห้องความสัมพันธ์	1	FK reference from Rooms (room_id) ON DELETE CASCADE ON UPDATE CASCADE

ตารางที่ 3.10 ข้อมูลรายงานการเดินทาง (Locations)

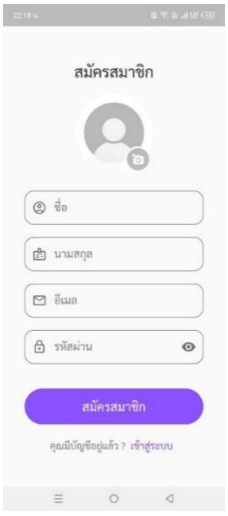
Id	Column	Type	Description	Example Data	Constraints
1	location_id	int	รหัสการเดินทาง	1	PK
2	location _distance	varchar (100)	ระยะทาง	2,000	Not null

ตารางที่ 3.10 ข้อมูลรายงานการเดินทาง (Locations) (ต่อ)

Id	Column	Type	Description	Example Data	Constraints
3	location_datetime	datetime	วันที่เวลา	10/12/2022 12:50:00	Not null
4	location _latitude	double	ละติจูด	14.2451	Not null
5	location_longitude	double	ลองจิจูด	105.2879	Not null
6	u_id	int	รหัสสมาชิก	1	FK reference from Users (u_id) ON DELETE CASCADE ON UPDATE CASCADE

3.6 การพัฒนาระบบ

3.6.1 การสมัครสมาชิก



ภาพประกอบที่ 3.10 ตัวอย่างหน้าเข้าสู่ระบบ

3.6.1.1 MySQL เก็บข้อมูลการสมัครสมาชิก

```
3   class MyServer {
4     String domain = '                    ..000webhostapp.com';
5     MyServer();
6   }
```

ภาพประกอบที่ 3.11 เก็บ Server ของ Class MyServer

บรรทัดที่ 3 การสร้าง Class MyServer เพื่อประกาศตัวแปรเก็บ Server

บรรทัดที่ 4 ประกาศตัวแปร domain ที่มีชนิดข้อมูลเป็น String เพื่อรับค่า Server

```
506 Future<void> getUserToMySQL() async {
507   String fName = _nameController;
508   String lName = _lastnameController;
509   String email = _emailController;
510   String password = _pwdController;
511   int status = 1;
512   String? token = await FirebaseMessaging.instance.getToken();
513   String path =
514       '${MyServer().domain}/familylife/getUserWhereUser.php?isAdd=true'
515       '&email=$email';
516   try {
517     await Dio().get(path).then((value) async {
518       if (value.toString() == 'null') {
519         InsertToMySQL(
520             f_name: fName,
521             l_name: lName,
522             email: email,
523             password: password,
524             imageUrl: _image != null ? imageUrl! : tempUserImageUrl,
525             status: status,
526             fb_uid: firebaseAuth.currentUser!.uid,
527             token: token
528         );
529       } else {}
530     });
531   } catch (e) {
532     print('Error getUserToMySQL : $e');
533   }
534 }
```

ภาพประกอบที่ 3.12 การสร้างฟังก์ชัน getUserToMySQL()

บรรทัดที่ 506 การสร้างฟังก์ชัน getUserToMySQL() เมื่อทำงานกับฟังก์ชันนี้แบบ Asynchonous ใช้งานร่วมกับคำสั่ง async และ await สำหรับรอให้ทำงานจนเสร็จ

บรรทัดที่ 507-512 ประกาศตัวแปร ที่มีชนิดข้อมูลเป็น String เพื่อรับค่า

บรรทัดที่ 517 เรียกใช้ Dio().get(path).then เพื่อเชื่อมต่อกับ Api จาก path ของบรรทัดที่ 513

บรรทัดที่ 518 ถ้า value.toString() == 'null' ให้ทำงานที่ InsertToMySQL() บรรทัดที่ 519-528

```
538   Future<void> InsertToMySQL({
539     String? f_name,
540     String? l_name,
541     String? email,
542     String? password,
543     String? imageUrl,
544     int? status,
545     String? fb_uid,
546     String? token,
547   }) async {
548     String apiInsertUser =
549         '${MyServer().domain}/familylife/insertUser.php?isAdd=true'
550         '&f_name=$f_name'
551         '&l_name=$l_name'
552         '&email=$email'
553         '&password=$password'
554         '&imageUrl=$imageUrl'
555         '&status=$status'
556         '&fb_uid=$fb_uid'
557         '&token=$token';
558     print('apiInsertFamilyList : $apiInsertUser');
559     try {
560       await Dio().get(apiInsertUser).then((value) {
561         print('value : $value');
562       });
563     } catch (e) {
564       print('Error InsertToMySQL : $e');
565     }
566   }
567 }
```

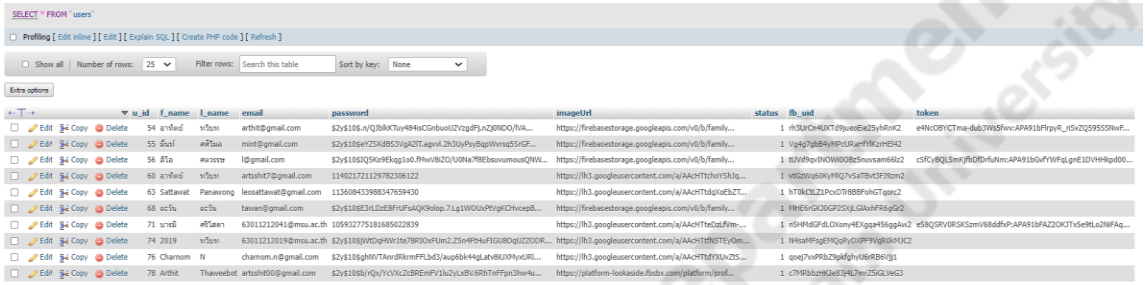
ภาพประกอบที่ 3.13 การสร้างฟังก์ชัน InsertToMySQL()

บรรทัดที่ 538 สร้างฟังก์ชัน InsertToMySQL()

บรรทัดที่ 539-546 ประกาศตัวแปร ที่มีชนิดข้อมูลเป็น String? เพื่อรับค่าจาก บรรทัดที่ 520-527

บรรทัดที่ 548-557 สร้างตัวแปร String apiInsertUser เพื่อเรียก Api จาก Server แล้วนำค่ามาเก็บ

บรรทัดที่ 560 เรียกใช้ Dio().get(path).then เพื่อเชื่อมต่อกับ Api จาก apiInsertUser



	u_id	f_name	l_name	email	password	imageUrl	status	fb_uid	token
Edit Copy Delete	54	ธาทิพย์	ทวีมาก	arthit@gmail.com	$2y$10$.n/QJbIkKTuy484isCGnbuoU2VzgdFJnZJ0fNDO/IVA...	https://firebasestorage.googleapis.com/v0/b/family...	1	rh3UrCm4UXTd9jueoEie25yhRnK2	e4NcOBYCTma-dub2Wi6Sfwv:APA91bFIrpyR_rISvZQ595S5NwF...
Edit Copy Delete	55	นันท์	ศศิโสม	mint@gmail.com	$2y$10$eYZSKd8S3VgA2iT.agvvL2h3UyPsyBqpWvrsq5SrGF...	https://firebasestorage.googleapis.com/v0/b/family...	1	Vq4g7qbB4yMPcURaHfYlKzrHEH42	
Edit Copy Delete	56	อีโอ	ศพวรรธ	l@gmail.com	$2y$10$IQSKo9Ekqg1o0.fHwVBiZO/U0Na7fBEbsuvumousQNW...	https://firebasestorage.googleapis.com/v0/b/family...	1	tlJVd9qvINOWI0OBz5nuvsam66lz2	c5fCyBQLSmKJfbDfDrfuNm:APA91bGvfYWFqLgnE1DVHHkpd00...
Edit Copy Delete	60	ธาทิพย์	ทวีมาก	artsshit7@gmail.com	114021721129782306122	https://lh3.googleusercontent.com/a/AAcHTtchoYSh3g...	1	vtGzWo60KyMlQ7vSaTBvt3FJ9zm2	
Edit Copy Delete	63	Sattawat	Panawong	leosattawat@gmail.com	113608439888347659430	https://lh3.googleusercontent.com/a/AAcHTtdqXoEbZT...	1	hT0kCtLZ1PcxDTr8BBFohGTqorc2	
Edit Copy Delete	68	ธวริน	ธะวิน	tawan@gmail.com	$2y$10$E3rLDzE8FrUFsAQKSolop.7.Lg1WOUxPtVgKCHvcepB...	https://firebasestorage.googleapis.com/v0/b/family...	1	MHE6rGrJ0GP2SXjLGkxhFR6gGr2	
Edit Copy Delete	71	นารมี	ศรีโสดา	6301121204 1@msu.ac.th	105932775181685022839	https://lh3.googleusercontent.com/a/AAcHTteDzLfVm-...	1	n5HMdGFdLOXomy4EXggx456ggAw2	e58QSRV0RSKSzmV68ddfxP:APA91bFAZ2OKJTx5e9tLo2NiFAq...
Edit Copy Delete	74	2019	ทวีมาก	6301121201 9@msu.ac.th	$2y$10$JWtDqHWr1te78RJOxFUm2.ZSn4PtHuFIGU8DqUIZODR...	https://lh3.googleusercontent.com/a/AAcHTtfNSTEyOm...	1	N4saMFsgEMQqRyDXPF9VgRbKMJC2	
Edit Copy Delete	76	Charnom	N	charnom.n@gmail.com	$2y$10$ghNVTAnrdRkrmFFLbd3/aup6bk44gLatvBiUXMyxURl...	https://lh3.googleusercontent.com/a/AAcHTtdYXUwZtS...	1	qoej7vxPRbZ9pkfghyU6rRB6Vjj1	
Edit Copy Delete	78	Arthit	Thaveebot	artsshit00@gmail.com	$2y$10$b/rQx/YcVXcZcBREmFV1lu2yLxBV.6RhTnFFpn3lhw4u...	https://platform-lookaside.fbsbx.com/platform/prof...	1	c7MRbbzHKJe83j4L7mvZ5iGLVeG3	

ภาพประกอบที่ 3.14 ตัวอย่างข้อมูลที่บันทึงลง MySQL

3.6.1.2 Firebase การสมัครสมาชิก

```
6   FirebaseAuth _auth = FirebaseAuth.instance;
7
8   class User {
9     final String uid;
10    late final String? name;
11    late final String? lastname;
12    late final String? imageUrl;
13    late final String? status;
14    late final String? token;
15
16    User({
17      required this.uid,
18      this.name = '',
19      this.lastname = '',
20      this.imageUrl = '',
21      this.status = '',
22      this.token = '',
23    });
```

ภาพประกอบที่ 3.15 ตัวแปรของ class User

บรรทัดที่ 6 ประกาศ instance ของ FirebaseAuth

บรรทัดที่ 8 สร้าง Class User

บรรทัดที่ 9-14 ประกาศตัวแปร uid, name, lastname, imageUrl, status, token ที่มีชนิดข้อมูลเป็น String

```
25  Future addUser() async {
26    assert(_auth.currentUser != null);
27    String uid = _auth.currentUser!.uid;
28    imageUrl ??= this.imageUrl;
29    name ??= this.name;
30    lastname ??= this.lastname;
31    status ??= this.status;
32    String? token = await FirebaseMessaging.instance.getToken();
33    CollectionReference users = FirebaseFirestore.instance.collection('users');
34    DocumentSnapshot documentSnapshot = await users.doc(uid).get();
35    if (!documentSnapshot.exists) {
36      try {
37        await users.doc(uid).set({
38          'uid': uid,
39          'fname': name,
40          'lname': lastname,
41          'imageUrl': imageUrl,
42          'status': status,
43          'token': token,
44        }, SetOptions(merge: true));
45      } catch (e) {}
46    }
47  }
48 }
```

ภาพประกอบที่ 3.16 การสร้าง Future type addUser()

บรรทัดที่ 25 สร้างฟังก์ชัน addUser() เพื่อใช้ในการเพิ่มข้อมูลสมาชิก

บรรทัดที่ 26 assert เพื่อทดสอบว่า _auth.currentUser != null หรือไม่

บรรทัดที่ 28-31 ให้ตัวแปรที่ประกาศ บรรทัดที่ 28-31 = ตัวแปรที่ required บรรทัดที่ 17

บรรทัดที่ 33 ให้ users = users ใน collection ของ FirebaseFirestore

บรรทัดที่ 34 ให้ข้อมูลมาเก็บที่ doc.(uid)

บรรทัดที่ 37-43 จากนั้นนำข้อมูลบันทึกไปที่ users ใน collection ของ Firebase ตามdoc.(uid)

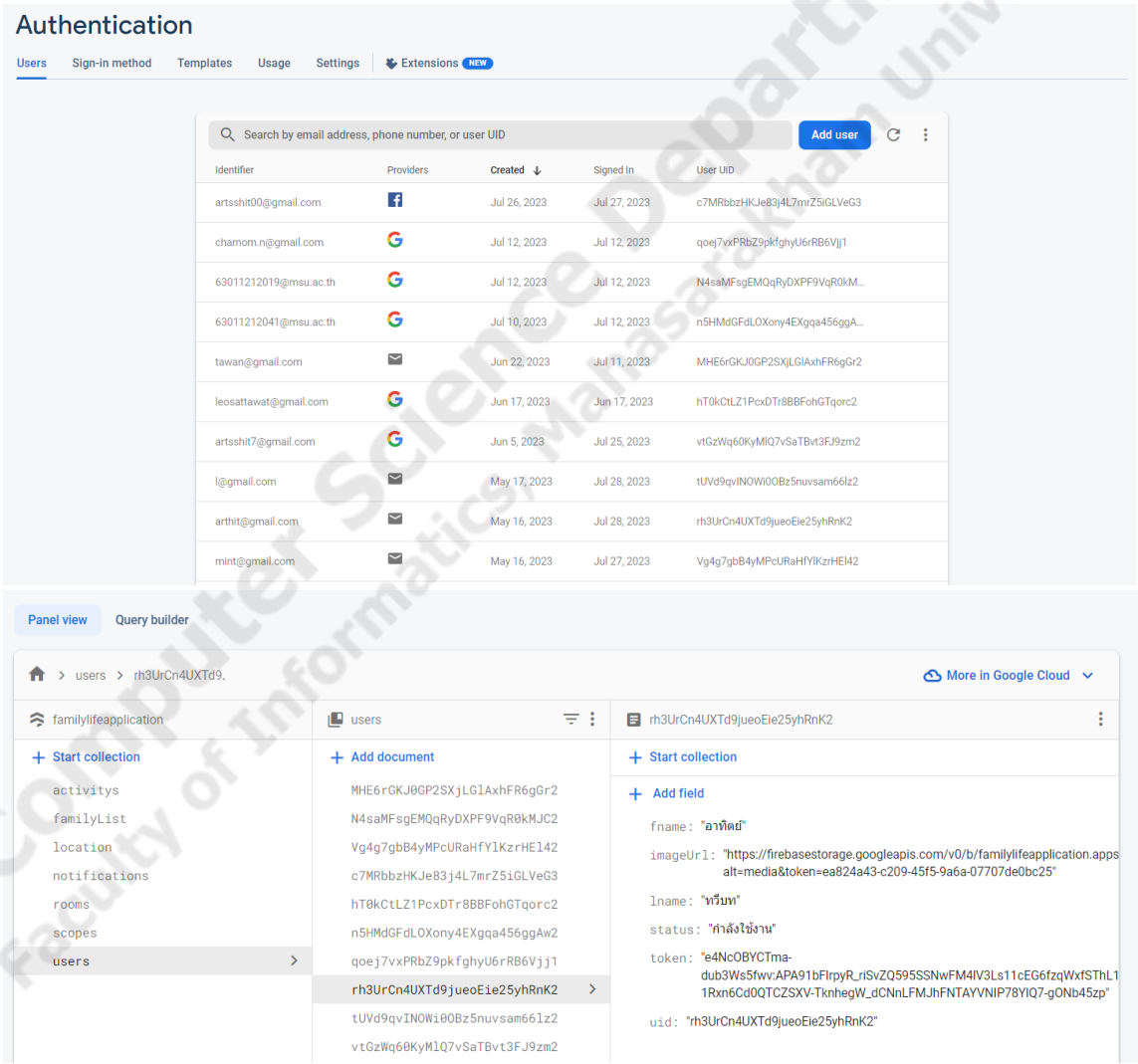
```dart
signUp() async {
  if (formKey.currentState!.validate()) {
    formKey.currentState!.save();
    setState(() {
      circular = true;
    });
    try {
      QuerySnapshot querySnapshot = await FirebaseFirestore.instance
          .collection('users')
          .where('email', isEqualTo: _emailController)
          .get();
      if (querySnapshot.docs.isNotEmpty) {
        setState(() {
          circular = false;
        });
        ScaffoldMessenger.of(context).showSnackBar(
            const SnackBar(content: Text('อีเมลนี้มีอยู่ในระบบแล้ว')));
        return;
      } else {
        UserCredential userCredential = await FirebaseAuth.instance.createUserWithEmailAndPassword(
          email: _emailController,
          password: _pwdController,
        );
        String? token = await FirebaseMessaging.instance.getToken();
        userData.User user = userData.User(
          uid: firebaseAuth.currentUser!.uid,
          name: _nameController,
          lastname: _lastnameController,
          imageUrl: _image != null ? imageUrl! : tempUserImageUrl,
          status: 'กำลังใช้งาน',
          token: token
        );
        SharedPreferences sharedpreferences = await SharedPreferences.getInstance();
        sharedpreferences.setString('uid', FirebaseAuth.instance.currentUser!.uid);
        sharedpreferences.setString('fname', _nameController);
        sharedpreferences.setString('imageUrl', _image != null ? imageUrl! : tempUserImageUrl);
        await user.addUser();
        getUserToMySQL();
        print(userCredential.user!.email);
        setState(() {
          circular = false;
        });
        Navigator.pushNamedAndRemoveUntil(context, '/', (route) => false);
      }
    } catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('อีเมลนี้มีอยู่ในระบบแล้ว')),
      );
      setState(() {
        circular = false;
      });
    }
  }
}
```

ภาพประกอบที่ 3.17 การเรียกใช้ signUp() และการสมัคร พร้อม Login

บรรทัดที่ 468-470 firebaseAuth.createUserWithEmailAndPassword จากการป้อน ข้อมูล email, password

บรรทัดที่ 473-480 userData.User = Class User ที่ประกาศตัวแปร บรรทัดที่ 6

บรรทัดที่ 485 ให้ user นั้นไปที่ฟังก์ชัน addUser() เรียกใช้เมื่อมีการป้อนข้อมูล และกดปุ่ม สมัครสมาชิก เช่นเดียวกันกับ getUserToMySQL() ของ MySQL จากนั้น เมื่อกดปุ่มแล้วก็จะเข้าสู่ระบบเลย โดยไม่ต้องไปหน้าเข้าสู่ระบบอีกครั้ง



ภาพประกอบที่ 3.18 ตัวอย่าข้อมูลที่บันทึกลง Firebase

3.6.2 การเข้าสู่ระบบ

3.6.2.1 การเข้าสู่ระบบด้วย Email/Password

```
25  final CollectionReference usersReference = FirebaseFirestore.instance.collection('users');
26  firebase_auth.FirebaseAuth firebaseAuth = firebase_auth.FirebaseAuth.instance;
27  final TextEditingController _emailController = TextEditingController();
28  final TextEditingController _pwdController = TextEditingController();
```

```
362 signIn() async {
363   if (formKey.currentState!.validate()) {
364     formKey.currentState!.save();
365     setState(() {
366       circular = true;
367     });
368     try {
369       firebase_auth.UserCredential userCredential = await firebaseAuth.signInWithEmailAndPassword(
370           email: _emailController.text,
371           password: _pwdController.text
372       );
373       setState(() {
374         circular = false;
375       });
376       addTokenUser(_auth.currentUser!.uid);
377       addDataPreferences();
378       addTokenUserToMySQL();
379       editPasswordToMySQL(_emailController.text, _pwdController.text);
380       Navigator.pushNamedAndRemoveUntil(context, '/', (Route<dynamic> route) => false);
381     } catch (e) {
382       showError(e.toString());
383       setState(() {
384         circular = false;
385       });
386     }
387   }
388 }
```

ภาพประกอบที่ 3.19 การเข้าสู่ระบบ Email & Password

บรรทัดที่ 27-28 สร้างTextEditingController สำหรับ email และ password เพื่อดึงค่าจาก Text, TextField

บรรทัดที่ 369-372 firebaseAutheAuth.signInWithEmailAndPassword จากการป้อนข้อมูล email, password

ภาพประกอบที่ 3.20 ตัวอย่างหน้าการเข้าสู่ระบบ

3.6.3 การสร้างห้องความสัมพันธ์

```
28  final CollectionReference familyListReference = FirebaseFirestore.instance.collection('familyList');
29  final CollectionReference roomReference = FirebaseFirestore.instance.collection('rooms');
30  final CollectionReference userReference = FirebaseFirestore.instance.collection('users');
31  final CollectionReference locationReference = FirebaseFirestore.instance.collection('location');
32
33  Future<void> createRoom() async {
34    await roomReference.doc(roomId).set({
35      'roomName': roomName,
36      'rid': rid,
37      'familyList': familyList,
38      'roomId': roomId,
39      'uid': uid,
40    });
```

ภาพประกอบที่ 3.21 การสร้างฟังก์ชัน createRoom()

บรรทัดที่ 29 ประกาศ roomRef = rooms ใน collection ของ FirebaseFirestore

บรรทัดที่ 30 ประกาศ userRef = users ใน collection ของ FirebaseFirestore

บรรทัดที่ 33 สร้างฟังก์ชัน createRoom() เพื่อใช้ในการสร้างห้องความสัมพันธ์

บรรทัดที่ 34-39 จากนั้นนำข้อมูลบันทึกไปที่ rooms ใน collection ของ Firebase ตาม room.Id

```dart
Future<List<User>> familyLists() async {
  List<User> userList = <User>[];
  for (String uid in familyList) {
    DocumentSnapshot snapshot = await userReference.doc(uid).get();
    Map<String, dynamic> data = snapshot.data() as Map<String, dynamic>;
    User user = User(
        uid: data['uid'],
        name: data['fname'],
        lastname: data['lname'],
        imageUrl: data['imageUrl'],
        status: data['status'],
        token: data['token']
    );
    userList.add(user);
  }
  return userList;
}
```

ภาพประกอบที่ 3.22 การสร้างฟังก์ชัน Future List familyLists()

บรรทัดที่ 43 สร้างฟังก์ชัน Future List familyLists() จาก Class User เพื่อใช้ในการ เก็บสมาชิกครอบครัวในห้องความสัมพันธ์

บรรทัดที่ 44 สร้าง List userList เพื่อเก็บเป็นสมาชิกในครอบครัว จาก Class User

บรรทัดที่ 45 ทำการ For loop uid in familyList

บรรทัดที่ 46 ให้ข้อมูลมาเก็บที่ doc.(uid) จาก userRef

บรรทัดที่ 47 สร้าง Map กำหนดตัวแปร data เพื่อมาเก็บข้อมูลจาก user บรรทัดที่ 27-32

บรรทัดที่ 56 นำข้อมูลที่ได้จาก user มาเก็บลงใน userList

บรรทัดที่ 58 return ส่งค่าคืน userList

```dart
List<String> selectedUsernamesUid = <String>[];
List<String> selectedUserFnames = <String>[];
List<String> selectedUserimageUrl = <String>[];

for (var doc in _selectedUsernamesDoc) {
  selectedUsernamesUid.add(doc['uid']);
  selectedUserFnames.add(doc['fname']);
  selectedUserimageUrl.add(doc['imageUrl']);
}
```

ภาพประกอบที่ 3.23 การสร้างห้องความสัมพันธ์ Rooms

```
126
127
128
129
130
131
132
133
134
```

```dart
DocumentReference randomDoc = roomsReference.doc();
Rooms newRoom = Rooms(
  rid: rid,
  roomId: randomDoc.id,
  roomName: roomName,
  familyList: selectedUsernamesUid,
  uid: _auth.currentUser!.uid,
);  // Rooms
await newRoom.createRoom();
```

ภาพประกอบที่ 3.23 การสร้างห้องความสัมพันธ์ Rooms (ต่อ)

บรรทัดที่ 103 loop ข้อมูล _selectedUsernamesUid ผู้ดูแล มาเก็บใน List บรรทัดที่ 99-101

บรรทัดที่ 127-134 สร้าง Class Rooms ตัวแปร newRoom มาเก็บ ชื่อห้องความสัมพันธ์ และสมาชิกในครอบครัว จาก roomName และ สมาชิก เพื่อสร้างห้อง

ภาพประกอบที่ 3.24 ตัวอย่างการสร้างห้องความสัมพันธ์

3.6.4 การสแกน QR code

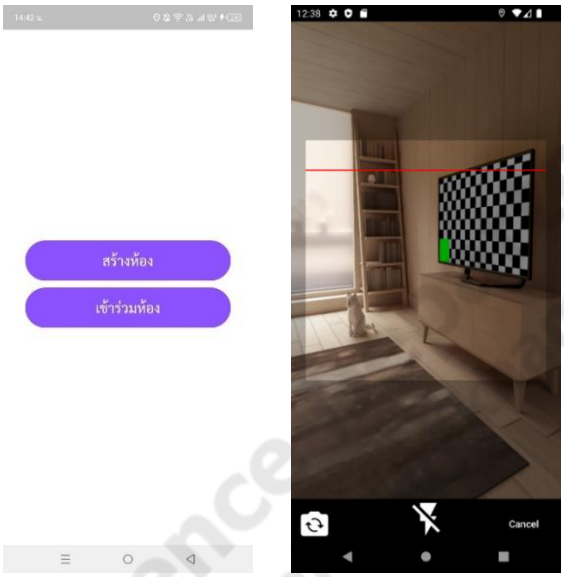
```
87
88
89
90
91
92
93
94
95
96
```

```dart
onPressed: () async {
  var data = await FlutterBarcodeScanner.scanBarcode(
      "red", "Cancel", true, ScanMode.QR);
  setState(() {
    qrData = data.toString();
    hasdata = true;
    qrData = encryption
        .decryptMsg(encryption.getCode(qrData))
        .toString();
  });
```

ภาพประกอบที่ 3.25 การเรียกใช้กล้อง scanBarcode

บรรทัดที่ 88-90 เป็นการเรียกใช้กล้องจาก FlutterBarcodeScanner.scanBarcode

บรรทัดที่ 91-96 qrData = ค่าจากการอ่านค่า Barcode ที่สร้างขึ้นจากตัวแปร encrypter จากClass AESEncryption



ภาพประกอบที่ 3.26 ตัวอย่างหน้าแสกน QR Code

3.6.5 การสร้าง QR Code

```dart
import 'package:encrypt/encrypt.dart' as encrypt;

class AESEncryption {
  static final key = encrypt.Key.fromLength(32);
  static final iv = encrypt.IV.fromLength(16);
  static final encrypter = encrypt.Encrypter(encrypt.AES(key));
  encryptMsg(String text) => encrypter.encrypt(text, iv: iv);
  decryptMsg(encrypt.Encrypted text) => encrypter.decrypt(text, iv: iv);
  getCode(String encoded) => encrypt.Encrypted.fromBase16(encoded);
}
```

ภาพประกอบที่ 3.27 การสร้าง QR Code จากเลขฐาน 16

บรรทัดที่ 4-11 สร้าง class AESEncryption สำหรับการสร้าง Qrcode โดยสร้างจากเลขฐาน 16

```dart
class _QRGeneratorroomState extends State<QRGeneratorroom> {
  TextEditingController mycontroller = TextEditingController();
  AESEncryption encryption = AESEncryption();
```

ภาพประกอบที่ 3.28 สร้าง class _QRGeneratorState

บรรทัดที่ 26-28 สร้าง class _QRGeneratorState สำหรับการเรียกใช้ Qrcode จากตัวแปร encrypter จากคลาส AESEncryption

```
84        onPressed: () {
85          Clipboard.setData(ClipboardData(text: roomID!));
86          Clipboard.getData(Clipboard.kTextPlain).then((value) {
87            mycontroller.text = mycontroller.text + roomID!;
88          });
89          String encryptedMsg = encryption.encryptMsg(mycontroller.text).base16;
90          Navigator.push(
91            context,
92            MaterialPageRoute(
93              builder: (context) => GeneratedQRroom(encryptedMsg),
94            ),  // MaterialPageRoute
95          );
96        },
```

ภาพประกอบที่ 3.29 กดปุ่มสร้าง QRcode

บรรทัดที่ 84-96 เมื่อกดปุ่มสร้าง QRcode ให้ส่งค่าไปยังหน้า GeneratedQR

```
15  class GeneratedQRroom extends StatefulWidget {
16    final String myQR;
17    const GeneratedQRroom(this.myQR, {super.key});
```

```
34            Center(
35              child: RepaintBoundary(
36                key: qrKey,
37                child: QrImage(
38                  data: widget.myQR,
39                  version: QrVersions.auto,
40                  size: 300,
41                  gapless: false,
42                ),  // QrImage
43              ),  // RepaintBoundary
44            )  // Center
```

ภาพประกอบที่ 3.30 แสดงภาพ QRcode

บรรทัดที่ 15-17 สร้าง myQR มารับค่าที่ส่งมา

บรรทัดที่ 35-43 สร้างภาพของ QrImage โดยที่ data คือข้อมูล myQR ที่ได้รับ เป็นภาพ Qrcode

ภาพประกอบที่ 3.31 ตัวอย่างหน้าสร้าง QR Code

3.6.6 แสดงตำแหน่ง Google Map

```
19  final CollectionReference roomRef = FirebaseFirestore.instance.collection('rooms');
20  final FirebaseAuth _auth = FirebaseAuth.instance;
21  BehaviorSubject<double> radius = BehaviorSubject();
22  bool isLoading = true;
23  double _value = 0.0;
24  String _label = 'Adjust Radius';
25  late Set<Marker> _markers = {};
26  late Stream<List<DocumentSnapshot>> stream;
27  late String grpDocID;
28  GoogleMapController? mapController;
29  Geoflutterfire geo = Geoflutterfire();
```

ภาพประกอบที่ 3.32 การประกาศตัวแปรที่ใช้ในการสร้าง Map

บรรทัดที่ 25 สร้างตัวแปรชนิด Marker เพื่อใช้ตั้งค่า ในการแสดง Marker บนแผนที่ ที่สร้าง List เพื่อเก็บ Marker

บรรทัดที่ 28 สร้าง object GoogleMapController เพื่อใช้ในการควบคุมแผนที่

บรรทัดที่ 29 สร้าง class Geoflutterfire เพื่อจะได้ใช้ฟังก์ชันใน Library มาดึงตำแหน่งปัจจุบัน

```
64  void _onMapCreated(GoogleMapController mapController) {
65    StreamSubscription<Position> positionStream = Geolocator.getPositionStream(
66        intervalDuration: Duration(minutes: 1)).listen((Position position) async {
67      double prevZoom = await mapController.getZoomLevel();
68      mapController.animateCamera(
69        CameraUpdate.newCameraPosition(CameraPosition(
70          target: LatLng(position.latitude, position.longitude),
71          zoom: prevZoom,
72        )),  // CameraPosition
73      );
74      GeoFirePoint myLocation = Geoflutterfire().point(
75          latitude: position.latitude,
76          longitude: position.longitude);
77      addToDB(myLocation);
78    });
79    setState(() {
80      this.mapController = mapController;
81    });
82    stream.listen((List<DocumentSnapshot> documentList) {
83      updateMarkers(documentList);
84    });
85  }
```

ภาพประกอบที่ 3.33 การสร้าง GoogleMap

บรรทัดที่ 64 สร้าง Method _onMapCreated เพื่อสร้างแผนที่ในแอปพลิเคชัน

บรรทัดที่ 67-66 StreamSubscription object ของ Position คือตำแหน่ง เป็นการใช้ข้อมูล Stream Geolocator.getPositionStream โดยทำการอัปเดตข้อมูล ตำแหน่งอย่างอย่างต่อเนื่อง intervalDuration ระยะเวลา 1 นาที

บรรทัดที่ 68-73 การสร้าง CameraUpdate.newCameraPositionคือการเลื่อนจอบนMap สามารถ zoom เข้าออกได้

บรรทัดที่ 74-78 สร้าง class GeoFirePoint ประกาศตัวแปร myLocation ในการเก็บlatitude และ longitude และสร้าง addToDB มาเก็บข้อมูล

```
302  addToDB(myLocation) {
303    roomsReference
304        .doc(roomID)
305        .collection('locations')
306        .doc(_auth.currentUser!.uid)
307        .set({
308        'uid': _auth.currentUser!.uid,
309        'fname': _name,
310        'imageUrl': _imageUrl,
311        'position': myLocation.data
312    });
313  }
```

ภาพประกอบที่ 3.34 เก็บข้อมูล locations

บรรทัดที่ 302 สร้าง addToDB มาเก็บข้อมูล myLocation

บรรทัดที่ 303 เรียกใช้ roomRef = rooms ใน collection ของ FirebaseFirestore

บรรทัดที่ 304 ตาม doc(grpDocID) คือ id ห้องความสัมพันธ์

บรรทัดที่ 305 เก็บข้อมูลใน collection ของ FirebaseFirestore ที่ชื่อว่า locations

บรรทัดที่ 306 ตาม _auth.currentUser!.uid คือตาม id Users ที่มีการ Authen อยู่

บรรทัดที่ 308-311 บันทึกข้อมูล uid, fname, imageUrl, position ใน collection ของ Firebase

```
515  updateMarkers(List<DocumentSnapshot> documentList) {
516    setState(() {
517      _markers.clear();
518    });
519
520    if (showCurrentLocation) {
521      showCurrentLocationMarker();
522    }
523
524    for (var document in documentList) {
525      final GeoPoint point = document['position']['geopoint'];
526      addMarker(point.latitude, point.longitude, document['uid'], document['fname'], document['imageUrl']);
527    }
528  }
```

```
705      _markers.add(
706        Marker(
707          markerId: MarkerId(name),
708          position: LatLng(latitude, longitude),
709          draggable: false,
710          icon: customMarkerIcon,
711          infoWindow: InfoWindow(
712            title: fname,
713          ),  // InfoWindow
714        ),  // Marker
715      );
716      setState(() {
717      });
```

ภาพประกอบที่ 3.35 การแสดงจุด Markers

บรรทัดที่ 515 สร้าง Method updateMarkers ตาม List ของข้อมูล

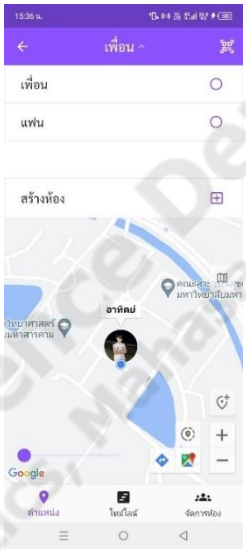
บรรทัดที่ 524-528 ทำการ forEach loop documentList เพื่อบันทึกข้อมูล position และ geopoint จากการ addMarker

บรรทัดที่ 706-717 _markers จะรับค่าจาก Marker โดยที่ Marker คือจุดที่แสดง ตำแหน่งสมาชิก ตาม MarkerId(name), LatLng(latitude, longitude), icon และ title บน Map

```
94          GoogleMap(
95            padding: const EdgeInsets.only(top: 155),
96            myLocationEnabled: isMapVisible,
97            mapType: isMapVisible ? _currentMapType : MapType.none,
98            markers: markers.union(_markers),
99            circles: circles,
100           initialCameraPosition: _currentCameraPosition,
101           onMapCreated: onMapCreated,
102         ),  // GoogleMap
```

ภาพประกอบที่ 3.36 การแสดง GoogleMap บนแผนที่

บรรทัดที่ 94-101 การเรียกใช้ onMapCreated, _markers เพื่อแสดง Map และพิกัดบนแอปพลิเคชัน



ภาพประกอบที่ 3.37 ตัวอย่างหน้าแสดง Map

3.6.7 การแจ้งเตือน

```
3   import 'package:firebase_messaging/firebase_messaging.dart';
4
5   class FirebaseMessagingService {
6     final _firebaseMessaging = FirebaseMessaging.instance;
7
8     Future<void> getToken() async {
9       await _firebaseMessaging.requestPermission();
10      final token = await _firebaseMessaging.getToken();
11      print('token : $token');
12    }
13  }
```

ภาพประกอบที่ 3.38 สร้าง Class FirebaseMessagingService

บรรทัดที่ 3 Import package ของ firebase_messaging เพื่อใช้งาน

บรรทัดที่ 5-13 สร้าง Class FirebaseMessagingService ใช้ในการจัดการกับ Cloud Messaging

บรรทัดที่ 6 สร้างตัวแปร _firebaseMessaging และกำหนดให้เป็น instance ให้กับตัวแปรของ FirebaseMessaging เพื่อใช้ในการเชื่อมต่อ

บรรทัดที่ 8-12 สร้าง Method getToken() เพื่อรับค่า token ของอุปกรณ์

```
440  Future<void> notification(String token, String activityStatus, String fname) async {
441    String title = '$fname (${widget.roomName})';
442    String body = activityStatus;
443    String urlSendToken =
444        '${MyServer().domain}/familylife/notification.php?isAdd=true&token=$token&title=$title&body=$body';
445    sendNotification(urlSendToken);
446  }
450  Future<void> sendNotification(String urlSendToken) async {
451    await Dio().get(urlSendToken).then((value) {
452      print('value : $value');
453    });
454  }
```

ภาพประกอบที่ 3.39 การส่งการแจ้งเตือน

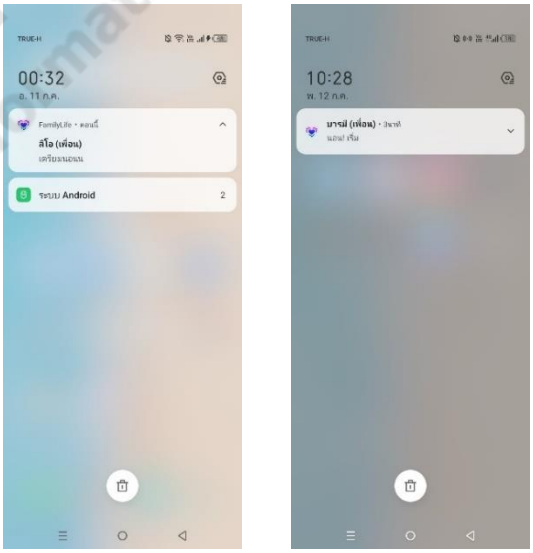
บรรทัดที่ 440-446 สร้างMethod notification() เพื่อรับพารามิเตอร์ token, activityStatus, fname จากการกดปุ่มบันทึกกิจกรรม

บรรทัดที่ 441 ตัวแปร title จะเป็น String รับค่า fname และ roomName

บรรทัดที่ 442 ตัวแปร body จะเป็น String รับค่า activityStatus

บรรทัดที่ 443-445 การส่งข้อมูลการแจ้งเตือน (notification) ไปยังเซิร์ฟเวอร์

บรรทัดที่ 450-454 สร้าง Method sendNotification() รับพารามิเตอร์ urlSendToken ในการส่ง



ภาพประกอบที่ 3.40 ตัวอย่างการแสดงการแจ้งเตือน