

แอปพลิเคชันสูงวัยมายเฟรนด์

OLD MY FRIENDS

นายนรากร วิเชียรไชย

โครงการนี้เป็นส่วนหนึ่งของการศึกษา

หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

ภาควิชาคณิตศาสตร์ สติทิ และคอมพิวเตอร์ คณะวิทยาศาสตร์

มหาวิทยาลัยอุบลราชธานี

ปีการศึกษา 2561

ลิขสิทธิ์ของมหาวิทยาลัยอุบลราชธานี

โครงงาน : แอปพลิเคชันสูงวัยมายเฟรนด์
OLD MY FRIENDS
โดย : นายนรากร วิเชียรไชย
อาจารย์ที่ปรึกษา : ผศ.ชยaphar แก่นสาร์
ระดับการศึกษา : วิทยาศาสตรบัณฑิต สาขาวิชาการคอมพิวเตอร์
ปีการศึกษา : 2561

ได้รับการพิจารณาให้เป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาการคอมพิวเตอร์

คณะกรรมการสอบประเมินความรู้โครงงานคอมพิวเตอร์

..... อาจารย์ที่ปรึกษา
(ผศ.ชยaphar แก่นสาร์)

..... กรรมการ
(ดร.สุภาวดี ทิรัญพงศ์สิน)

..... กรรมการ
(ดร.วิชิต สมบัติ)

..... หัวหน้าภาควิชา
(ดร.ชัชวิน นามมั่น)

วันที่/...../.....

กิตติกรรมประกาศ

การพัฒนาแอปพลิเคชันสูงวัยมายเพรนด์ (OLD MY FRIENDS) สำเร็จลุล่วงได้ด้วยความกรุณาและความช่วยเหลือจากหลายท่าน ข้าพเจ้าขอขอบคุณทุกท่าน ที่มีส่วนร่วมในการพัฒนาโครงการนี้

ขอขอบพระคุณอาจารย์ที่ปรึกษา ผศ. ชาญพร แก่นสาร อาจารย์ที่ปรึกษาโครงการที่ได้แนะนำทฤษฎีและแนวทางในแก่ปัญหาต่าง ๆ ที่เกิดขึ้นระหว่างการพัฒนาระบบ อีกครั้งยังคงอยู่ตรวจสอบความก้าวหน้าของการทำงานเป็นระยะ ๆ รวมทั้งสร้างกำลังใจในการพัฒนาโครงการนี้ตลอดจนโครงการเสร็จสมบูรณ์

ขอขอบพระคุณอาจารย์ประจำสาขาวิชาการคอมพิวเตอร์ อาจารย์ประจำภาควิชาคณิตศาสตร์ สพิติ และคอมพิวเตอร์ และอาจารย์ในคณะวิทยาศาสตร์ทุก ๆ ท่าน ที่เคยให้คำแนะนำ อบรมสั่งสอน และค่อยช่วยเหลือข้าพเจ้า ในการศึกษาตลอดมา ขอบคุณเจ้าหน้าที่และบุคลากรของคณะวิทยาศาสตร์ ที่ได้อำนวยความสะดวกทางด้านอุปกรณ์และเครื่องมือต่าง ๆ

ขอกราบขอบพระคุณบิดา มารดา ที่ค่อยให้กำลังใจ ค่อยให้ความรักและความห่วงใยเสมอมา ตลอดจนค่อยช่วยเหลือทุนทรัพย์ทางด้านการศึกษาและอุปกรณ์ในการพัฒนาโครงการ

ขอบคุณเพื่อน ๆ สาขาวิชาการคอมพิวเตอร์ชั้นปีที่ 4 ที่ได้ค่อยช่วยแก่ไขปัญหาและให้คำปรึกษาในการพัฒนาโครงการครั้งนี้จนเสร็จสิ้น

นายธนากร วิเชียรไชย วันที่ 26 เมษายน 62

โครงการ : แอปพลิเคชันสูงวัยมายเฟรนด์
 โดย : นายนรากร วิเชียรไชย
 อาจารย์ที่ปรึกษา : ผศ.ชยាមร แก่นสาร์
 ระดับการศึกษา : วิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
 ปีการศึกษา : 2561

บทคัดย่อ

แอปพลิเคชันสูงวัย มายเฟรนด์ (OLD MY FRIENDS) ได้ถูกพัฒนาขึ้นโดยสามารถทำงานได้ทั้งบนระบบปฏิบัติการแอนดรอยด์ (Android) และไอโอเอส (IOS) ซึ่งเป็นแอปพลิเคชันที่ช่วยอำนวยความสะดวกสำหรับกลุ่มผู้สูงอายุ โดยประกอบด้วยระบบแชทบอทเพื่อความตอบเรื่องโรคที่เกิดกับผู้สูงอายุจำนวน 5 โรค ซึ่งสามารถให้ข้อมูลสาเหตุ อาการ วิธีป้องกันและดูแลรักษาเบื้องต้นได้ และระบบยังสามารถแจ้งเตือนการทำงานยาได้ด้วย นอกจากนี้ยังมีระบบโพสต์กระทู้หรือหัวข้อสนทนา เพื่อช่วยแชร์เรื่องราวที่น่าสนใจกับเพื่อนหรือกลุ่มได้ ทั้งยังมีระบบการระบุตำแหน่งของผู้ใช้งานเพื่อช่วยติดตามหรือแจ้งเตือนกรณีที่หลงทาง โดยในการพัฒนานั้นจะใช้ ionic 3 และ dialogflow เพื่อสร้างแชทบอทสำหรับトイตอป โดยประโยชน์ของแอปพลิเคชัน คือช่วยให้คำแนะนำ การส่งเสริมพฤติกรรมสุขภาพที่ดีผ่านระบบแชทบอท และสร้างกลุ่มเพื่อผ่านการพูดคุยทางสังคมออนไลน์ทำให้ช่วยลดความเหงาและสร้างกิจกรรมให้กับผู้สูงอายุได้

คำสำคัญ: แชทบอท ไอโอนิก แอนดรอยด์ ไอโอเอส ผู้สูงอายุ dialogflow

Topic	:	OLD MY FRIENDS
Author	:	MR.NARAKORN VICHIANCHAI
Advisor	:	Chayaporn Kaensar, Asst. Prof.
Degree	:	Bachelor of Science (Computer Science)
Academic Year	:	2018

Abstract

OLD MY FRIENDS is a cross-platform mobile application to facilitate elderly which runs on both Android operating system and iOS operating system. It contains functionality that is needed. The chatbot system to ask and answer about the disease. To help understand the diseases of the elderly 5 disease by can provide information on the causes, symptoms, protection and basic care, Azusa Temperature data logging function private health system can alert about taking care of yourself each disease. Such as exercise, eating, etc., also can ask and answer about the weather. There is also a section for the elderly to post threads or topics. To help share the story interested with friends or groups. The function of tracking of the user to the track or alert in case of lost by using ionic 3 and development dialogflow, which is to create a chatbot to interactive talking by taking advantage of OLD MY FRIENDS. Is to give instructions to promote good health habits via chatbot. And create a group of friends through social talk online make reduce loneliness and construction activities for the elderly.

Keywords: chatbot, ionic, android, ios, elderly, dialogflow

สารบัญ

	หน้า
กิตติกรรมประกาศ	ค
บทคัดย่อภาษาไทย	ง
บทคัดย่อภาษาอังกฤษ	จ
สารบัญ	ฉ
สารบัญตาราง	ฉ
สารบัญภาพ	ญ
บทที่	
1 บทนำ	1
1.1 ที่มาและเหตุผล	1
1.2 วัตถุประสงค์	1
1.3 ขอบเขตของโครงการ	2
1.4 ประโยชน์ที่คาดว่าจะได้รับ	2
1.5 เครื่องมือที่ใช้ในการพัฒนา (Development tools)	2
1.5.1 ไฮาร์ดแวร์	2
1.5.2 ซอฟต์แวร์ (Software)	2
1.5.3 แผนการดำเนินการ	3
2 ทฤษฎีที่เกี่ยวข้อง	4
2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง	4
2.1.1 Hybrid Mobile Application	4
2.1.2 ความรู้พื้นฐานระบบปฏิบัติการแอนดรอยด์	6
2.1.2.1 โครงสร้างของระบบปฏิบัติการแอนดรอยด์	7
2.1.2.2 ข้อเด่นของระบบปฏิบัติการแอนดรอยด์	9
2.1.2.3 การจัดการเกี่ยวกับวัสดุจัดทำและหิวต์ของแอปพลิเคชัน	9
2.1.2.4 กระบวนการเริ่มทำงานของแอคทิวิตี้ (Activity)	10
2.1.3 ความรู้พื้นฐานระบบปฏิบัติการ iOS	12
2.1.3.1 เวอร์ชันของ iOS	13
2.1.4 ความรู้พื้นฐาน Ionic Framework	24
2.1.4.1 เทคโนโลยีที่ใช้ในการพัฒนา Ionic Framework	24

2.1.4.2 การทำงานของ Cordova Application	25
2.1.4.3 ความแตกต่างระหว่าง PhoneGap/Cordova	26
2.1.4.4 เริ่มต้นการใช้งาน	27
2.1.5 Firebase	27
2.1.5.1 บริการหลักของเฟร์เบส	28
2.1.5.2 การพัฒนา Cloud Firestore	29
2.1.6 Dialogflow	31
2.1.6.1 เริ่มต้นใช้งาน Dialogflow	31
2.1.7 Libraries moment.js	37
2.1.7.1 คำสั่งในการติดตั้ง	37
2.1.7.2 รูปแบบการใช้งาน moment.js	38
2.1.8 Google Maps API	39
2.2 เอกสารและงานวิจัยที่เกี่ยวข้อง	41
2.2.1 แอปพลิเคชันเครือข่ายสังคมออนไลน์เพื่อผู้สูงอายุ (OLDSTER)	41
2.2.2 จับใจบท	42
3 การวิเคราะห์และออกแบบระบบ	44
3.1 โครงสร้างภาพรวมของระบบ (System Architecture)	44
3.2 การวิเคราะห์ความต้องการของระบบ (System Requirements)	45
3.2.1 ความต้องการหลักของระบบ (Functional Requirements)	45
3.2.2 Non-functional Requirements	46
3.3 การออกแบบส่วนติดต่อผู้ใช้ (User Interface Design)	46
3.3.1 หน้าเข้าสู่ระบบ	46
3.4 แผนภาพโดยรวม	60
3.4.1 ยูสเคสโดยรวม (Use Case Diagram)	60
3.4.2 ตารางแสดงคำอธิบายของผู้ใช้ (User Case Description)	62
3.4.3 คลาสโดยรวม (Class Diagram)	68
3.4.4 ซีเควนโดยรวม (Sequence Diagram)	73
4 การพัฒนาระบบ	89
4.1 การสมัครสมาชิก	89
4.2 การเข้าสู่ระบบ	92
4.3 ส่วนจัดการแซทบท	96

4.4 การพัฒนาในส่วนของ Dialogflow ที่ใช้ fulfillment ในการเชื่อมต่อกับ Firebase	98
4.5 การจัดการโพสท์	100
4.5.1 การเพิ่มโพสท์	100
4.5.2 การลบโพสท์	104
4.5.3 การกดถูกใจโพสท์	107
4.5.4 การเลือกหมวดโพสท์	108
4.5.5 การจัดการคอมเม้นท์	109
4.6 การค้นหาเพื่อน	111
4.7 การเพิ่มการแจ้งเตือนการทำงานยา	111
5 การทดสอบระบบ	114
5.1 ผลการทดสอบการสมัครสมาชิก	115
5.2 ผลการทดสอบการเข้าสู่ระบบ	116
5.3 ผลการทดสอบการคุยกับแซทบอท	117
5.4 ผลการทดสอบการจัดการโพสท์	118
5.5 ผลการทดสอบการเลือกประเภทโพสท์	119
5.6 ผลการทดสอบการจัดการคอมเม้นท์	120
5.7 ผลการทดสอบค้นหาเพื่อน	121
5.8 ผลการทดสอบเพิ่มการแจ้งเตือนการทำงานยา	121
6 สรุปและข้อเสนอแนะ	122
6.1 สรุปความสามารถของระบบ	122
6.2 ปัญหาและอุปสรรคในการพัฒนา	122
6.3 แนวทางการพัฒนาต่อ	122
บรรณานุกรม	123
ประวัติผู้เขียน	125

สารบัญตาราง

ตารางที่	หน้า
1.1 ขั้นตอนการดำเนินงาน	3
3.1 สัญลักษณ์ของ Use case Diagram	60
3.2 อธิบาย Class Diagram ของคลาสพื้นฐานของระบบ	62
3.3 Use Case เข้าสู่ระบบ	63
3.4 Use Case ออกจากระบบ	63
3.5 Use Case จัดการข้อมูลส่วนตัว	63
3.6 Use Case ดูเพิ่ม แก้ไข หรือลบโพสท์	64
3.7 Use Case โหวหรือส่งข้อความมาชิกครอบครัว	64
3.8 Use Case ดูตำแหน่งสมาชิกครอบครัว	64
3.9 Use Case จัดการครอบครัว	65
3.10 Use Case จัดการกลุ่ม	65
3.11 Use Case จัดการเพื่อน	66
3.12 Use Case เพิ่ม หรือลบการแจ้งเตือน	66
3.13 Use Case ดูวิธีการใช้งาน	66
3.14 Use Case คุยกับเซทบท	67
3.15 สัญลักษณ์ของ Class Diagram	68
3.16 อธิบาย Class Diagram ของแอปพลิเคชันสูงวัยมายเฟรนด์	71
3.17 อธิบาย Class Diagram ของแอปพลิเคชันสูงวัยมายเฟรนด์	72
3.18 สัญลักษณ์ของ Sequence Diagram	73
5.1 ผลการทดสอบการสมัครสมาชิก	115
5.2 ผลการทดสอบการเข้าสู่ระบบ	116
5.3 ผลการทดสอบการคุยกับเซทบท	117
5.4 ผลการทดสอบการจัดการโพสท์	118
5.5 ผลการทดสอบการเลือกประเภทโพสท์	119
5.6 ผลการทดสอบการจัดการคอมเม้นท์	120
5.7 ผลการทดสอบค้นหาเพื่อน	121
5.8 ผลการทดสอบเพิ่มการแจ้งเตือนการทานยา	121

สารบัญภาพ

รูปที่		หน้า
2.1	Hybrid Mobile Application คืออะไร	5
2.2	Cross Platform Frameworks	5
2.3	โครงสร้างของระบบปฏิบัติการแอนดรอยด์	7
2.4	วัสดุจัดของแอคทิฟิลีบระบบปฏิบัติการแอนดรอยด์	11
2.5	iOS 1	13
2.6	iOS 2	14
2.7	iOS 3	15
2.8	iOS 4	16
2.9	iOS 5	17
2.10	iOS 6	18
2.11	iOS 7	19
2.12	iOS 8	20
2.13	iOS 9	21
2.14	iOS 10	22
2.15	iOS 11	23
2.16	iOS 12	24
2.17	การทำงานของ Ionic Framework	25
2.18	Cordova Application	26
2.19	แสดงการติดตั้ง cli	27
2.20	Firebase 2.0	28
2.21	เว็บ https://firebase.google.com	29
2.22	รูปแบบการส่งข้อมูลผู้ใช้ไปยังเซิร์ฟเวอร์	31
2.23	หน้าหลักของ Dialogflow	32
2.24	หน้า Create Agent	33
2.25	หน้า Intent ส่วน Training phrases	34
2.26	หน้า Intent ส่วน Responses	35
2.27	ทดสอบคุยกับบอท	36
2.28	ทดสอบคุยกับบอท	37
2.29	คำสั่งในการติดตั้ง moment.js	37
2.30	รูปแบบการใช้งาน moment.js	38
2.31	เรียกใช้ API KEY	40
2.32	เรียกใช้งาน google maps ใน ionic framework	41
2.33	แอปพลิเคชัน OLDSTER	42
2.34	เซิร์ฟเวอร์ของ จับใจบอท	43
3.1	System architecture แอปพลิเคชันสูงวัยมายเฟรนด์	45

3.2	หน้าจอเข้าสู่ระบบ	46
3.3	หน้าจอหลัก	47
3.4	หน้าແຫບອທຄູຍກັບປຸ່ຈອ້າທິນ	48
3.5	หน้าຈອກຮະດານຂ່າວ	49
3.6	หน้าຄອບຄົວ	50
3.7	หน้าແສດງຕຳແໜ່ງຂອງຄອບຄົວ	51
3.8	หน້າຊຸກເຈີນ	52
3.9	หน້າເລືອກຄອບຄົວ	53
3.10	หน້າວິວິໄຈງານ	54
3.11	หน້າຈອແສດງກລຸ່ມ	55
3.12	หน້າຈອແສດງເພື່ອນ	56
3.13	หน້າແສດງຮາຍການແຈ້ງເຕືອນທານຍາ	57
3.14	หน້າແສດງການເພີ່ມການແຈ້ງເຕືອນທານຍາ	58
3.15	หน້າຈອແສດງຂໍ້ມູນສ່ວນຕົວຂອງຜູ້ໃຊ້	59
3.16	Use Case Diagram ຂອງແອປລີເຄີບເຈັນສູງວ້າຍມາຍເຟຣັນດີ	61
3.17	Class Diagram ຂອງແອປລີເຄີບເຈັນສູງວ້າຍມາຍເຟຣັນດີ	70
3.18	Sequence Diagram ການເຂົ້າສູ່ຮະບບ	75
3.19	Sequence Diagram ມູນຄູກັບແຫບອທ	77
3.20	Sequence Diagram ການແສດງໜ້າຮະດານຂ່າວ	79
3.21	Sequence Diagram ຂອງການເພີ່ມກລຸ່ມ	81
3.22	Sequence Diagram ຂອງການຈັດການຂໍ້ມູນສ່ວນຕົວ	83
3.23	Sequence Diagram ຂອງການຈັດການເພື່ອນ	85
3.24	Sequence Diagram ຂອງຊຸກເຈີນ	87
4.1	Code ການທຳມະນຸດຂອງຮະບບເນື່ອກົດປຸ່ມສົມຄຣສມາເຊີກ	90
4.2	ການທຳມະນຸດຂອງຮະບບເນື່ອກົດເຂົ້າສູ່ຮະບບ	92
4.3	Code ການທຳມະນຸດຂອງຮະບບເນື່ອກົດເຂົ້າສູ່ຮະບບ (ຕ່ອ)	94
4.4	Code ການທຳມະນຸດຂອງການມູນຄູກັບແຫບອທ	96
4.5	Code ສ່ວນຂອງ Dialogflow ທີ່ໃຊ້ fulfillment ໃນການເຊື່ອມຕ່ອກັບ Firebase	98
4.6	Code ການສ້າງ Alert ເນື່ອຄລິກໂພສທ	100
4.7	Code ການສ້າງ Alert ເນື່ອຄລິກໂພສທ (ຕ່ອ)	101
4.8	Code ການເພີ່ມໂພສທ (ຕ່ອ)	102
4.9	Code ການລົບໂພສທ	104
4.10	ການພັດທະນາໃນສ່ວນຂອງລົບໂພສທ (ຕ່ອ)	106
4.11	Code ການກົດຊຸກໃຈໂພສທ	107
4.12	ສ່ວນຂອງການເລືອກໝາດໂພສທ	108
4.13	Code ການຈັດກາຣຄອມເມນທ	109
4.14	Code ການຄັ້ນຫາເພື່ອນ	111
4.15	Code ການເພີ່ມການແຈ້ງເຕືອນທານຍາ	112

บทที่ 1

บทนำ

1.1 ที่มาและเหตุผล

ในภาวะสังคมปัจจุบัน ประเทศไทยในปี 2561 มีผู้สูงอายุทั้งเพศและเพศหญิงเฉลี่ยประมาณ 16.06 เปอร์เซ็นต์ ของประชากรทั้งหมดในประเทศไทย [1] ผู้สูงอายุส่วนใหญ่อยู่บ้านเพียงลำพัง อันเนื่องสาเหตุมาจากการลูกหลานต้องออกไปทำงานที่ต่างจังหวัด หรือในกรุงเทพมหานคร ส่งผลให้ ผู้สูงอายุ สนใจที่จะใช้โซเชียลในการติดต่อสื่อสาร หรือแชร์เรื่องราวที่น่าสนใจในคนวัยเดียวกัน ซึ่ง ในปัจจุบันมีเทคโนโลยีในการสื่อสารมากมาย เช่น เฟสบุ๊ค ไลน์ แอปพลิเคชันเหล่านี้ถูกออกแบบ สำหรับคนรุ่นใหม่มากกว่าผู้สูงอายุ ปัญหาคือการใช้งานมีความซับซ้อน ตัวหนังสือที่มีขนาดเล็ก รวมไปถึงไม่มีฟังก์ชันที่ช่วยอำนวยความสะดวกให้ผู้สูงอายุในกรณีฉุกเฉินได้เฉพาะเจาะจง นอกจากนี้ผู้สูงอายุยังไม่เข้าใจสาเหตุ วิธีดูแล ที่ถูกต้องในเรื่องโรคที่มักเกิดกับผู้สูงอายุ เนื่องจากการค้นหาข้อมูลต่าง ๆ ในปัจจุบันมีความซับซ้อน ทำให้เกิดความสับสนในการใช้งานสำหรับผู้สูงอายุ

ดังนั้นผู้พัฒนาจึงต้องการเสนอแอปพลิเคชันสูงวัยมายเฟรนด์ (OLD MY FRIENDS) ที่ช่วยให้ผู้สูงอายุสามารถสื่อสารหรือแชร์เรื่องราวที่น่าสนใจ กับคนวัยเดียวกัน และยังสามารถให้ ความรู้เบื้องต้น เช่น สาเหตุ อาการ วิธีป้องกัน วิธีรักษา อาการแทรกซ้อน ในเรื่องโรคที่เกิดขึ้นบ่อย กับผู้สูงอายุจำนวน 5 โรค [2] ได้แก่ โรคเบาหวาน โรคซีมเศร้า โรคความดันโลหิตสูง โรคข้อเสื่อม โรคอัลไซเมอร์ ที่จะเป็นประโยชน์สำหรับผู้สูงอายุ และผู้สูงอายุยังสามารถดู ตำแหน่งปัจจุบันของ คนในครอบครัว และยังสามารถแจ้งเหตุเมื่อเกิดเหตุฉุกเฉินด้วยการโทรหรือส่งข้อความไปยังคนใน ครอบครัวได้ นอกจากนี้ แอปพลิเคชัน ยังประกอบด้วยฟังก์ชันแจ้งเตือนการทานยา ซึ่งจะช่วยให้ผู้ สูงอายุสามารถทานยาได้ถูกต้อง และตรงตามเวลา

1.2 วัตถุประสงค์

- เพื่อพัฒนาแอปพลิเคชันให้ความรู้พื้นฐานเรื่องโรคที่เกิดกับผู้สูงอายุผ่านระบบซอฟต์แวร์
- เพื่อพัฒนาระบบแอปพลิเคชันสำหรับผู้สูงอายุให้สามารถแบ่งปันเรื่องราวที่น่าสนใจได้
- เพื่อพัฒนาแอปพลิเคชันอำนวยความสะดวกในการติดต่อสื่อสารในกรณีฉุกเฉิน

1.3 ขอบเขตของโครงการ

แอปพลิเคชันสูงวัยมายเฟรนด์มีข้องเขตการทำงานดังนี้

- ผู้ใช้งานสามารถเข้าสู่ระบบได้
- ผู้ใช้งานสามารถสนทนากับแซทบอทแบบเรียลไทม์ เพื่อสอบถามเรื่องโรคมีจำนวน 5 โรค ได้แก่ โรคเบาหวาน, โรคซึมเศร้า, โรคความดันโลหิตสูง, โรคข้อเสื่อม, โรคอัลไซเมอร์
- ผู้ใช้งานสามารถเพิ่ม ลบ แก้ไข โพสท์และคอมเมนท์ได้
- ผู้ใช้งานสามารถดู เพิ่ม แก้ไขตำแหน่ง และลบสมาชิกในครอบครัวได้
- ผู้ใช้สามารถจัดการการแจ้งเตือน เช่น การแจ้งเตือนการทานยาและแจ้งเตือนฉุกเฉินได้
- ผู้ใช้สามารถติดตามโปรแกรมได้
- ผู้ใช้สามารถเพิ่ม ลบ แก้ไข กลุ่มและเพื่อนได้
- ผู้ใช้สามารถจัดการแก้ไขข้อมูลส่วนตัวได้

1.4 ประโยชน์ที่คาดว่าจะได้รับ

- 1 ช่วยให้ผู้สูงอายุได้รับความรู้ของโรคพื้นฐานที่พบในผู้สูงอายุในรูปแบบแซทบอทแบบเรียลไทม์
- 2 ช่วยให้ผู้สูงอายุสามารถโพสท์แบ่งปันประสบการณ์สร้างกลุ่มเพื่อนในคนวัยเดียวกันได้
- 3 ช่วยอำนวยความสะดวกในการติดต่อสื่อสารกับสมาชิกในครอบครัวได้

1.5 เครื่องมือที่ใช้ในการพัฒนา (Development tools)

1.5.1 ฮาร์ดแวร์

1. ทำงานบนระบบปฏิบัติการแอนดรอยด์เวอร์ชัน 4.4 ขึ้นไป
2. ทำงานบนระบบปฏิบัติการไอโอเอสเวอร์ชัน 10 ขึ้นไป

1.5.2 ซอฟต์แวร์ (Software)

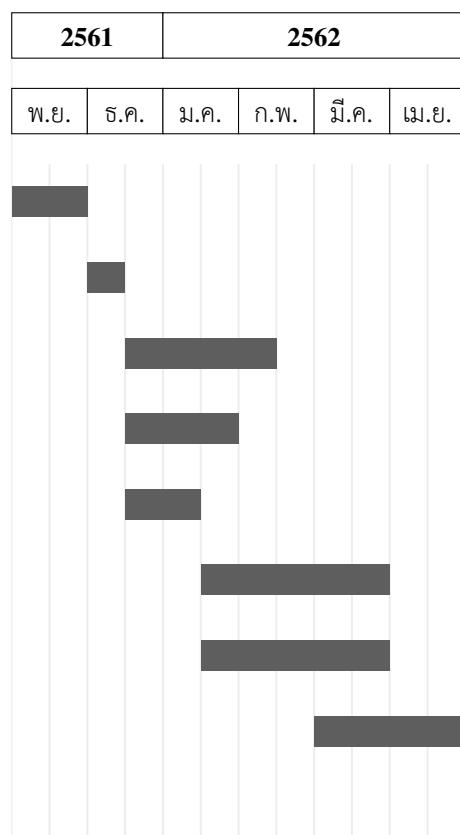
1. Ionic 3 ซึ่งเป็น Frontend Framework สำหรับพัฒนาเว็บแอปพลิเคชัน
2. Firebase คือ Platform ที่รวมเครื่องมือต่าง ๆ ที่ใช้จัดการ Backend หรือ Server side
3. Dialogflow หรือ Api.ai เป็นแพลตฟอร์มที่ใช้ในการสร้างแซทบอทที่รองรับการทำ Natural Language Processing (NLP)

4. Node Package Manager หรือ NPM เป็นซอฟต์แวร์ที่มาพร้อมกับ Node ที่ช่วยให้สามารถนำเข้าโมดูลต่าง ๆ ภายใน Node ได้
5. Library moment.js เป็น JavaScript Library สำหรับจัดการ Date Time
6. Google Maps APIs เป็น API ของ Google ไว้สำหรับเรียกใช้แผนที่
7. Visual Studio Code เครื่องมือสำหรับพัฒนาเว็บแอปพลิเคชัน

1.5.3 แผนการดำเนินการ

ในการสร้างแอปพลิเคชันสูงวัยมายเฟรนด์ ผู้พัฒนาได้แบ่งขั้นตอนการดำเนินงานไว้ด้วยกัน 8 ขั้นตอน ดังต่อไปนี้

ตารางที่ 1.1: ขั้นตอนการดำเนินงาน



บทที่ 2

ทฤษฎีที่เกี่ยวข้อง

ในบทนี้จะอธิบายถึงองค์ความรู้และทฤษฎีที่จำเป็นต่อการพัฒนาแอปพลิเคชันรวมทั้งงานวิจัยที่เกี่ยวข้อง มีรายละเอียดดังนี้

2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง

2.1.1 Hybrid Mobile Application

2.1.2 ความรู้พื้นฐานระบบปฏิบัติการแอนดรอยด์

2.1.3 ความรู้พื้นฐานของระบบปฏิบัติการไอโอเอส

2.1.4 ความรู้พื้นฐาน Ionic Framework

2.1.5 การใช้งาน Firebase เป็นฐานข้อมูล

2.1.6 Dialogflow

2.1.7 Libraries moment.js

2.1.8 Google Maps API

2.2 เอกสารและงานวิจัยที่เกี่ยวข้อง

2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง

2.1.1 **Hybrid Mobile Application**

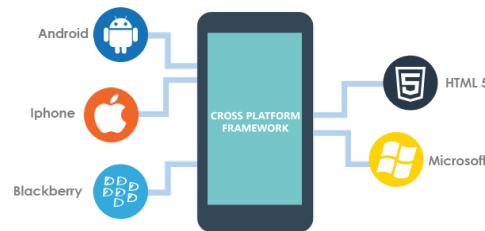
Hybrid Application คือ แอปพลิเคชันที่ถูกพัฒนาขึ้นมาเพื่อให้สามารถทำงานได้บนระบบปฏิบัติการทั้งหมดโดยพัฒนาแค่ครั้งเดียว โดยจำเป็นต้องผ่านเฟรมเวิร์กต่างๆเพื่อให้สามารถทำงานบน OS นั้นๆได้ เช่น PhoneGap (ไฟน์คัป) ซึ่งเป็น Open source framework ด้วยการพัฒนาแอปด้วยเทคโนโลยีเว็บ html, CSS และ Java Script เป็นต้น



รูปที่ 2.1: Hybrid Mobile Application คืออะไร

ที่มา : <https://www.mindphp.com/คุณมีอ/73-คืออะไร/3663-hybrid-application-ไฮบริด-แอปพลิเคชัน-หรือ-hybrid-app-ไฮบริด-แอป-คืออะไร.html>

การพัฒนา Mobile Application แต่เดิมนั้น ถ้าจะเริ่มก็คงต้องแต่ J2ME อาจจะเก่ามาก จนมาถึงยุคสมัยของ IOS และ Android รวมไปถึงน้องสุดท้องอย่าง Windows Phone ซึ่งแต่ก่อนก็ มีพัฒนาจาก Window CE มาก่อนหน้านี้ จนผู้ใช้งงแต่ละฝั่ง platform เริ่มมีความสำคัญใกล้เคียง กัน ดังนั้น การพัฒนาแอป เฉพาะของ iOS หรือ Android เพียงอย่างเดียวถือเป็นการเสียโอกาส ทางธุรกิจเป็นอย่างมาก จนมีคนเริ่มคิดหาวิธีทำให้มีชีวิตง่ายขึ้นโดยการเขียน HTML5 + CSS3 + JavaScript และใช้วิธีทำงานผ่าน Web View Component เป็นส่วนของหน้าเบราว์เซอร์ในแอปอีก ที ของแต่ละ Platform จนกลายมาเป็นโครงการ Cordova และได้มีการพัฒนาส่วนขยาย Plug-In เพิ่มเรื่อยๆ ทำให้ปัจจุบันเราสามารถเข้าถึง Hardware หรือ Sensor ซึ่งโดยปกติ HTML5 ธรรมดา ไม่สามารถเข้าถึงได้



รูปที่ 2.2: Cross Platform Frameworks

ที่มา : <https://www.promaticsindia.com/mobile-app-development-services/cross-platform-frameworks>

การเขียนแอปแบบ Hybrid Application คือ การพัฒนาโดยอาศัย Framework หรือ SDK ที่ถูกสร้างมาจากหลากหลายภาษาและมีเครื่องมือที่เหมาะสมกับ Framework หรือ SDK นั้น ๆ ให้เลือกใช้ในการพัฒนาที่หลากหลายตัวอย่างเช่น codova SDK ใช้ภาษา LUA , Acrobat AIR ใช้ภาษา ACTION SCRIPT 3 หรือ UNITY ใช้ C และ JAVASCRIPT ซึ่งการเขียนในรูปแบบนี้เราสามารถแปลงไปใช้กับระบบปฏิบัติการอื่น ๆ ได้และใช้เวลาน้อยในการเพื่อพัฒนาหลาย ๆ แอปพลิเคชัน

ข้อดีของ Hybrid Mobile Application

- 1) พัฒนาด้วยภาษา HTML, CSS และ JavaScript ทำให้ง่ายและเรียนรู้ได้อย่างรวดเร็ว
- 2) พัฒนาครั้งเดียวสามารถใช้ได้หลาย Platform ทั้ง iOS, Android และ Window Phone
- 3) ใช้ต้นทุนในการพัฒนาน้อยกว่า Native App

ข้อเสียของ Hybrid Mobile Application

- 1) ประสิทธิภาพการทำงานจะด้อยกว่า Native App
- 2) ในบางกรณีอาจจะใช้ความสามารถของอุปกรณ์ได้ไม่เต็มที่ เนื่องจากต้องขึ้นอยู่กับ Framework ที่เลือกในการพัฒนานั้นมี Component ที่ต้องการหรือไม่ ดังนั้n Hybrid App จึงมีจุดเด่นในเรื่องความง่ายและพัฒนาได้รวดเร็ว และ Cross-Platforms คือพัฒนาครั้งเดียวแต่สามารถนำไปติดตั้งในหลาย Platform แต่เมื่อพูดถึงเรื่องประสิทธิภาพในการทำงาน เช่นความเร็ว หรือการเรียกใช้หรือติดต่อ feature ต่าง ๆ ของอุปกรณ์ ก็ต้องยอมรับว่าอาจจะยังด้อยกว่าแอปพลิเคชันที่พัฒนาด้วย Native App ในบางลักษณะการทำงานอยู่ดี

2.1.2 ความรู้พื้นฐานระบบปฏิบัติการแอนดรอยด์

แอนดรอยด์ (Android) คือระบบปฏิบัติการแบบเปิดเผยแพร่ต้นฉบับ (Open Source) โดยบริษัท กูเกิล (Google Inc.) ที่ได้รับความนิยมเป็นอย่างสูง เนื่องจากอุปกรณ์ที่ใช้ระบบปฏิบัติการแอนดรอยด์ มีจำนวนมาก อุปกรณ์มีหลากหลายระดับ หลายราคา รวมทั้งสามารถทำงานบนอุปกรณ์ที่มีขนาดหน้าจอ และความละเอียดแตกต่างกันได้ ทำให้ผู้برمجสามารถเลือกได้ตามต้องการและหากมองในทิศทางสำหรับนักพัฒนาโปรแกรม (Programmer) แล้วนั้นการพัฒนาโปรแกรมเพื่อใช้งานบนระบบปฏิบัติการแอนดรอยด์ ไม่ใช่เรื่องยาก เพราะมีข้อมูลในการพัฒนาร่วมทั้ง Android SDK (Software Development Kit) เตรียมไว้ให้กับนักพัฒนาได้เรียนรู้ และเมื่อ

นักพัฒนาต้องการจะเผยแพร่หรือจำหน่ายโปรแกรมที่พัฒนาแล้วเสร็จแอนดรอยด์ก็ยังมีตลาดในการเผยแพร่โปรแกรม Google PlayStore แต่หากจะกล่าวถึงโครงสร้างภาษาที่ใช้ในการพัฒนานั้น สำหรับ Android SDK จะยึดโครงสร้างของภาษาจาวา (Java language) ในการเขียนโปรแกรม เพราะโปรแกรมที่พัฒนามาได้จะต้องทำงานอยู่ภายใต้ Dalvik Virtual Machine เช่นเดียว กับโปรแกรมจาวา ที่ต้องทำงานอยู่ภายใต้ Java Virtual Machine (Virtual Machine เปรียบได้ กับสภาพแวดล้อมที่โปรแกรมทำงานอยู่)

2.1.2.1 โครงสร้างของระบบปฏิบัติการแอนดรอยด์

การทำความเข้าใจโครงสร้างของระบบปฏิบัติการแอนดรอยด์ [3] ถือว่าเป็นสิ่งสำคัญ เพราะถ้านักพัฒนาโปรแกรม สามารถมองภาพโดยรวมของระบบได้ทั้งหมด จะสามารถเข้าใจถึงกระบวนการทำงานได้ดีขึ้น และสามารถนำไปช่วยในการออกแบบโปรแกรมที่ต้องการพัฒนาเพื่อให้เกิดประสิทธิภาพในการทำงาน



รูปที่ 2.3: โครงสร้างของระบบปฏิบัติการแอนดรอยด์

ที่มา : <https://www.theandroid-mania.com/android-architecture/>

จากโครงสร้างของระบบปฏิบัติการแอนดรอยด์ในรูปที่ 2.3 จะสังเกตได้ว่า มีการแบ่งออกเป็นส่วน ๆ ที่มีความเกี่ยวเนื่องกัน โดยส่วนบนสุดเป็นส่วนที่ผู้ใช้งานทำการติดต่อโดยตรงซึ่งคือส่วน

ของ Applications ลำดับถัดมาเป็นองค์ประกอบอื่น ๆ ตามลำดับ และสุดท้ายเป็นส่วนที่ติดต่อกับ อุปกรณ์โดยผ่านทาง Linux Kernel โครงสร้างของแอนดรอยด์สามารถอธิบายได้ดังนี้

- 1) Applications ส่วนแอปพลิเคชันหรือส่วนของโปรแกรมที่มากับระบบปฏิบัติการ หรือเป็นกลุ่มของโปรแกรมที่ผู้ใช้งานได้ทำการติดตั้งไว้ โดยผู้ใช้งานสามารถเรียกใช้โปรแกรมต่าง ๆ ได้โดยตรงซึ่งการทำงานของแต่ละโปรแกรมจะเป็นไปตามที่ผู้พัฒนาโปรแกรมได้ออกแบบ และเขียนโค้ด (Code) โปรแกรมเอาไว้
- 2) Application Framework เป็นส่วนที่มีการพัฒนาขึ้นเพื่อให้นักพัฒนาสามารถพัฒนาโปรแกรมได้สะดวก และมีประสิทธิภาพมากยิ่งขึ้น โดยนักพัฒนาไม่จำเป็นต้องพัฒนาในส่วนที่มีความยุ่งยากมาก ๆ เพียงแค่ทำการศึกษาถึงวิธีการเรียกใช้งาน Application Framework ในส่วนที่ต้องการใช้งานแล้วนำมาใช้งาน ซึ่งมีหลากหลายกลุ่มด้วยกัน ตัวอย่างเช่น
 - Activities Manager เป็นกลุ่มของชุดคำสั่งที่จัดการเกี่ยวกับวงจรการทำงานของหน้าต่างโปรแกรม (Activity)
 - Content Providers เป็นกลุ่มของชุดคำสั่ง ที่ใช้ในการเข้าถึงข้อมูลของโปรแกรมอื่น และสามารถแบ่งปันข้อมูลให้โปรแกรมอื่นเข้าถึงได้
 - View System เป็นกลุ่มของชุดคำสั่งที่เกี่ยวกับการจัดการโครงสร้างของหน้าจอที่แสดงผลในส่วนที่ติดต่อกับผู้ใช้งาน (User Interface)
 - Telephony Manager เป็นกลุ่มของชุดคำสั่งที่ใช้ในการเข้าถึงข้อมูลด้านโทรศัพท์ เช่น หมายเลขโทรศัพท์ เป็นต้น
 - Resource Manager เป็นกลุ่มของชุดคำสั่งในการเข้าถึงข้อมูลที่เป็นข้อความและรูปภาพ
 - Location Manager เป็นกลุ่มของชุดคำสั่งที่เกี่ยวกับตำแหน่งทางภูมิศาสตร์ที่ระบบปฏิบัติการได้รับค่าจากอุปกรณ์
 - Notification Manager เป็นกลุ่มของชุดคำสั่งที่จะถูกเรียกใช้เมื่อโปรแกรมต้องการแสดงผลให้กับผู้ใช้งาน ผ่านทางแถบสถานะ (Status Bar) ของหน้าจอ
- 3) Libraries เป็นส่วนของชุดคำสั่งที่พัฒนาด้วย C/C++ โดยแบ่งชุดคำสั่งออกเป็นกลุ่มตามวัตถุประสงค์ของการใช้งาน เช่น Surface Manager จัดการเกี่ยวกับการแสดงผล Media Framework จัดการเกี่ยวกับการการแสดงภาพและเสียง Open GL|ES และ SGL จัดการ

เกี่ยวกับภาพ 3 มิติ และ 2 มิติ SQLite จัดการเกี่ยวกับระบบฐานข้อมูล เป็นต้น

- 4) Android Runtime จะมี Darvik Virtual Machine ที่ถูกออกแบบมาเพื่อให้ทำงานบนอุปกรณ์ที่มีหน่วยความจำ (Memory) หน่วยประมวลผลกลาง (CPU) และพลังงาน (Battery) ที่จำกัดซึ่งการทำงานของ Darvik Virtual Machine จะทำการแปลงไฟล์ที่ต้องการทำงานไปเป็นไฟล์ .DEX ก่อนการทำงานเหตุผลเพื่อให้มีประสิทธิภาพเพิ่มขึ้นเมื่อใช้งานกับหน่วยประมวลผลกลางที่มีความเร็วไม่มากส่วนต่อมาคือ Core Libraries ที่เป็นส่วนรวมรวมคำสั่งและชุดคำสั่งสำคัญโดยถูกเขียนด้วยภาษาจาวา (Java Language)
- 5) Linux Kernel เป็นส่วนที่ทำหน้าที่หัวใจสำคัญในจัดการกับบริการหลักของระบบปฏิบัติการ เช่น เรื่องหน่วยความจำ พลังงาน ติดต่อกับอุปกรณ์ต่างๆ ความปลอดภัย เครือข่าย โดยแอนดรอยด์ได้นำเอาส่วนนี้มาจากระบบปฏิบัติการลินุกซ์ รุ่น 2.6 (Linux 2.6. Kernel) ซึ่งได้มีการออกแบบมาเป็นอย่างดี

2.1.2.2 ข้อเด่นของระบบปฏิบัติการแอนดรอยด์

เนื้องจากระบบปฏิบัติการแอนดรอยด์มีการเจริญเติบโตอย่างรวดเร็วและมีส่วนแบ่งตลาดของอุปกรณ์ด้านนี้ขึ้นทุกขณะ ทำให้กลุ่มผู้ใช้งานและกลุ่มนักพัฒนาโปรแกรมให้ความสำคัญกับระบบปฏิบัติการแอนดรอยด์เพิ่มมากขึ้น เมื่อมองในด้านของกลุ่มผลิตภัณฑ์บริษัทที่มีการพัฒนาผลิตภัณฑ์รุ่นใหม่ ได้มีการนำเอาระบบปฏิบัติการแอนดรอยด์ไปใช้ในสินค้าของตนเองพร้อมทั้งยังมีการปรับแต่งให้ระบบปฏิบัติการมีความสามารถ การจัดวาง โปรแกรมและลูกเล่นใหม่ ๆ ที่แตกต่างจากคู่แข่งในท้องตลาดโดยเฉพาะอย่างยิ่งกลุ่มสินค้าที่เป็นมือถือรุ่นใหม่ (SmartPhone) และอุปกรณ์จอสัมผัส(Touch Screen)โดยมีลักษณะแตกต่างกันไป เช่น ขนาดหน้าจอ ระบบโทรศัพท์ ความเร็วของหน่วยประมวลผล ปริมาณหน่วยความจำ แม้กระทั่งอุปกรณ์ตรวจจับ(Sensor)ต่าง ๆ หากมองในด้านของการพัฒนาโปรแกรม ทางบริษัท Google ได้มีการพัฒนา Application Framework ไว้สำหรับนักพัฒนาใช้งานได้อย่างสะดวกและไม่เกิดปัญหาเมื่อนำชุดโปรแกรมที่พัฒนาขึ้นมา ไปใช้กับอุปกรณ์ที่มีลักษณะต่างกัน เช่น ขนาดจออุปกรณ์ไม่เท่ากัน ก็ยังสามารถใช้งานโปรแกรมได้เหมือนกัน เป็นต้น

2.1.2.3 การจัดการเกี่ยวกับวัสดุจัดการแอคทิวิตี้ของแอปพลิเคชัน

ขณะที่ผู้ใช้เปิดใช้งานแอปพลิเคชัน -> ออกจากแอปพลิเคชัน -> แล้วก็กลับเข้ามาในแอปพลิเคชันอีกครั้งแอคทิวิตี้จะมีการรับ Method ต่างๆ เกิดขึ้นในวัสดุจัดการแอคทิวิตี้ ยกตัวอย่างเช่น

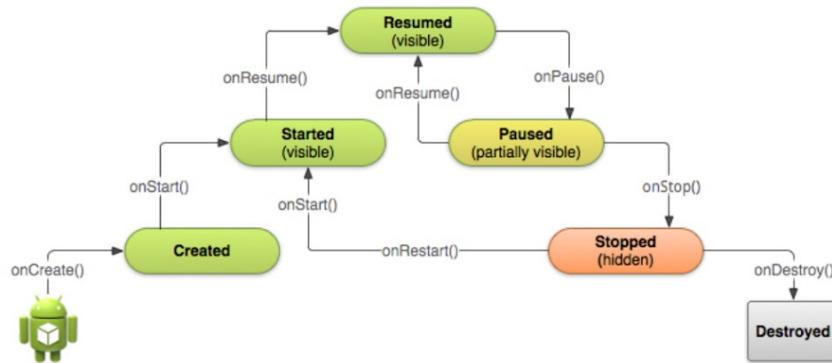
เมื่อแอคทิวิตี้เริ่มทำงานครั้งแรกจะแสดงขึ้นมาอยู่ด้านบนสุดของระบบ (Foreground) และรอรับการทำงานจากผู้ใช้งานระหว่างกระบวนการนี้ระบบจะมีการเรียกใช้งาน Callback Method หรือ Method ที่ถูกเรียกใช้งานอัตโนมัติในแอคทิวิตี้ที่ได้กำหนดการทำงานให้กับ UI และส่วนติดต่ออื่นๆ ได้ ถ้าผู้ใช้มีการใช้งานใดๆ ที่เป็นการเรียกแอคทิวิตี้อื่นขึ้นมาหรือสลับไปใช้งานแอปพลิเคชันอื่นระบบจะเรียก Callback Method อีกอันขึ้นมา เช่น ซ่อนแอปพลิเคชันไว้ด้านหลัง Background (ไม่แสดงแอคทิวิตี้แต่ Instance และ Method นั้นยังทำงานอยู่)

ภายใน Callback Method สามารถกำหนดการทำงานในแอคทิวิตี้เมื่อผู้ใช้ออกจากแอปพลิเคชันและกลับเข้ามาใช้งานแอปพลิเคชันใหม่อีกรอบได้ ตัวอย่าง ถ้าแอปพลิเคชันเป็นแอปพลิเคชัน Streaming Video จะจะสั่งให้ทำการหยุด Video ชั่วคราว และปิดการเชื่อมต่อ Network ไว้ก่อน เมื่อผู้ใช้สลับไปใช้แอปพลิเคชันอื่น และทันทีที่ผู้ใช้กลับมาใช้งานแอปพลิเคชันต่อ ก็ให้ทำการเชื่อมต่อ กับ Network และก้อนอุณหภูมิให้ผู้ใช้กลับไปเล่น Video ในตำแหน่งที่ค้างต่อไปทันทีโดยที่ไม่ต้องเริ่มต้นแอปพลิเคชันใหม่ เป็นต้น

2.1.2.4 กระบวนการเริ่มการทำงานของแอคทิวิตี้ (Activity)

ในระบบแอนดรอยด์การกำหนดโค้ดเริ่มต้นไว้ในแอคทิวิตี้โดยสัมพันธ์กับ Method ที่ถูกเรียกใช้งานอัตโนมัติ (Callback Method) อย่างเป็นลำดับ ตั้งแต่เริ่มต้นแอคทิวิตี้ไปจนถึงสิ้นสุดและปิดการทำงานของ Activity ลง

ในขณะที่แอคทิวิตี้ [4] ทำงานระบบจะเรียกใช้ Callback Method ตามลำดับในลักษณะที่คล้ายกับการก่อพิริมิด นั่นคือ แต่ละขั้นตอนวัฏจักรของแอคทิวิตี้คือส่วนแยกย่อยแต่ละขั้นของพิริมิด เช่น เมื่อระบบสร้าง Instance ของแอคทิวิตี้ขึ้นมาใหม่ Method ที่เรียกใช้งานอัตโนมัติ (Callback Method) จะขับ Activity Method ขึ้นมาด้านบนโดยด้านบนของพิริมิดคือจุดที่แอคทิวิตี้กำลังทำงานแสดงอยู่ด้านหน้า (Foreground Activity) สุดและผู้ใช้กำลังใช้งานอยู่และเมื่อผู้ใช้กำลังจะออกจากแอคทิวิตี้ระบบจะเรียกใช้ Method อื่นซึ่งทำให้ Activity Method ถอยกลับไปอยู่ด้านล่างของพิริมิดตามลำดับเพื่อหยุดการทำงานและลบแอคทิวิตี้ออกไป ในบางกรณีแอคทิวิตี้จะย้ายลงมาอยู่บاجจุดและรอจังหวะที่จะถูกเรียกกลับขึ้นมาด้านบนอีก เช่น ในกรณีเมื่อผู้ใช้สลับไปใช้งานแอปพลิเคชันอื่นแล้วกลับมาใช้งานอีกครั้ง



รูปที่ 2.4: วัฏจักรของแอคทิวิตี้บนระบบปฏิบัติการแอนดรอยด์

ที่มา : <https://www.dev2qa.com/android-activity-lifecycle-example/>

จากรูปที่ 2.4 แสดงวัฏจักรของแอคทิวิตี้ในรูปแบบโครงสร้างพิรามิดโดยแสดงให้เห็นว่า Method ที่เรียกใช้งานอัตโนมัติ (Callback Method) ได้แก่ onCreate(), onStart(), onResume() และ onRestart() จะขยับแอคทิวิตี้ขึ้นไปด้านบนสุดที่ Resumed Method และมี Method ได้แก่ onPause(), onStop() และ onDestroy() ที่จะขยับแอคทิวิตี้ลงมาด้านล่าง แอคทิวิตี้ยังสามารถกลับไปทำงานที่ตำแหน่ง Resumed Method จากตำแหน่ง Paused และ Stopped ได้อีกด้วย

ในบางครั้งไม่จำเป็นต้องเรียกใช้งาน Callback Method ทั้งหมดเสมอไปขึ้นกับความซับซ้อนของแอคทิวิตี้ อย่างไรก็ตามเป็นสิ่งสำคัญที่นักพัฒนาควรทำความเข้าใจแต่ละ Method เพื่อให้มั่นใจได้ว่าแอปพลิเคชันของที่ได้พัฒนาตอบสนองเป็นไปตามที่ผู้ใช้คาดหวัง ดังนั้น ในการใช้งาน Callback Method ที่ถูกวิธีจะช่วยให้แอปพลิเคชันทำงานได้เป็นอย่างดี ดังนี้

- ไม่หยุดการทำงานหรือค้าง กรณีมีสายโทรศัพท์เข้าหรือมีการสลับไปใช้งานแอปพลิเคชันอื่น
- ไม่ใช้ทรัพยากรที่มีค่าของระบบอย่างสูญเปล่า ถ้าไม่มีการใช้งานแอคทิวิตี้ใดๆ
- ไม่กระทบต่อกระบวนการในขั้นตอนการใช้งานของผู้ใช้กรณีออกจากแอปพลิเคชันแล้วกลับเข้ามาใช้งานอีกครั้ง
- ไม่หยุดการทำงานหรือระบบค้างที่กระทบการใช้งานของผู้ใช้กรณีมีการหมุนหน้าจอแนวนอนและแนวตั้งสลับกัน

เหตุการณ์ที่แอคทิวิตี้มีการเปลี่ยน Method ต่าง ๆ ตามแสดงในรูปที่ 2.4 แต่มีอยู่ 3 Method เท่านั้นที่แอคทิวิตี้จะยังคงอยู่ค้างที่ในช่วงเวลาระยะเวลาหนึ่งไม่เปลี่ยนไป Method อื่นในทันที

ได้แก่

- Resumed (แสดงอยู่ ทำงานอยู่) ใน Method นี้แอคทิวิตี้จะแสดงอยู่ด้านหน้าสุดและผู้ใช้กำลังใช้งานอยู่ บ่อยครั้งจะเรียกว่า Running Method
- Paused (แสดงหน้าจอบางส่วน ไม่ถูกบังสนิท) ใน Method นี้แอคทิวิตี้จะถูกบดบังด้วยแอคทิวิตี้อื่น เช่น แอคทิวิตี้อื่นที่อยู่ด้านหน้าสุดที่แสดงในลักษณะกึ่งโปร่งใสหรือไม่ได้แสดงแบบเต็มหน้าจอ แอคทิวิตี้ในสถานะนี้จะไม่สามารถรับค่าจากผู้ใช้และทำงานคำสั่งใด ๆ ได้
- Stopped (แสดงหน้าจอแบบ Background ผู้ใช้มองไม่เห็น) ใน Method นี้ แอคทิวิตี้จะถูกบดบังอย่างสมบูรณ์และผู้ใช้มองไม่เห็นโดยจะถูกย้ายไปอยู่ด้านหลังในขณะที่อยู่ใน Method นี้ ค่า Activity Instance และตัวแปรทั้งหมดจะยังคงอยู่แต่จะไม่สามารถถูกเรียกมาใช้งานจากโค้ดใด ๆ ได้

ในขณะที่ Method อื่น เช่น Created และ Started จะแสดงช่วงระหว่างระบบก็จะเปลี่ยนไป Method อื่นในทันทีที่ Method ถูกเรียกใช้งานอัตโนมัติ นั่นคือ หากจากที่ระบบเรียกใช้งาน onCreate() แล้วก็จะเรียกใช้งาน onStart() ทันทีและสุดท้ายตามด้วย onResume() ซึ่งก็จะเข้าสู่ Resumed Method ทั้งหมดก็คือวัสดุจัดการแอคทิวิตี้เบื้องต้น

2.1.3 ความรู้พื้นฐานระบบปฏิบัติการ iOS

ระบบปฏิบัติการไอโอเอส (iOS) [5] มีชื่อเดิมว่า iPhone OS เริ่มต้นด้วยการเปิดตัวของ iPhone เมื่อวันที่ 29 มิถุนายน 2550 ระบบปฏิบัติการไอโอเอส (iOS) เป็นระบบปฏิบัติการสำหรับสมาร์ทโฟน (Smartphone) ของแอปเปิล โดยเริ่มต้นพัฒนาสำหรับใช้ในโทรศัพท์ iPhone และได้พัฒนาต่อให้สำหรับ iPod Touch และ iPad โดยระบบปฏิบัติการนี้สามารถเชื่อมต่อไปยังแอปส托ร์สำหรับการเข้าถึงถึงแอพพลิเคชัน (Application) มากกว่า 300,000 ตัว ซึ่งมีการดาวน์โหลดไปมากกว่าห้าพันล้านครั้ง และเปลี่ยนไปมีการพัฒนาปรับปรุงสำหรับ iPhone, iPad และ iPod Touch ผ่านทางระบบ iTunes คือโปรแกรมฟรี สำหรับ Mac และ PC ใช้ดูหนังฟังเพลงบนคอมพิวเตอร์ รวมทั้งจัดระเบียบและ sync ทุกอย่าง และเป็นร้านขายความบันเทิงบนคอมพิวเตอร์, บน iPod touch, iPhone และ iPad ที่มีทุกอย่างสำหรับคุณ ในทุกที่และทุกเวลา พัฒนาระบบรักษาความปลอดภัยให้มีความเป็นเลิศ ซึ่งนี้คือข้อได้เปรียบ เมื่อเทียบกับคู่แข่ง

2.1.3.1 เวอร์ชันของ iOS

iOS 1 (iPhone OS)

สำหรับ iOS 1 จะเปิดตัวครั้งแรกที่งาน Macworld วันที่ 9 มกราคม 2007 พร้อมกับ iPhone รุ่นแรก และหลังจากนั้นก็อกร่างสำหรับ iPhone วันที่ 29 มิถุนายน 2007 สำหรับ iOS 1 มาพร้อมฟีเจอร์เด่น ๆ เช่น ระบบสัมผัสแบบมัลติทัช, voicemail, ท่องเว็บผ่าน Safari และดู YouTube แต่ iOS 1 ไม่ได้ฟรีสำหรับผู้ใช้งาน iPod touch จะต้องจ่ายเงิน 19.99 เหรียญ หรือประมาณ 690 บาท เพื่ออัปเดตเป็น iOS 1



รูปที่ 2.5: iOS 1

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 2 (iPhone OS 2.0)

เปิดตัวครั้งแรกที่งาน WWDC 2008 (วันที่ 9 มิถุนายน 2008) และหลังจากนั้นก็อกร่างสำหรับ iPhone 3G วันที่ 11 กรกฎาคม 2008 โดยมีฟีเจอร์เด่น ๆ เช่น App Store, แผนที่พร้อม GPS และระบบแจ้งเตือนอีเมล และเหมือนเช่นเคยผู้ใช้ iPhone สามารถอัปเดตได้ฟรี แต่ผู้ใช้ iPod touch จะต้องจ่ายเงิน 9.95 เหรียญ หรือประมาณ 390 บาท เพื่ออัปเดต iOS 2.x



รูปที่ 2.6: iOS 2

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 3 (iPhone OS 3.0)

เปิดตัวครั้งแรกที่งาน WWDC 2009 (วันที่ 8 มิถุนายน 2009) และหลังจากนั้นก็อกรอบ
จำหน่ายพร้อม iPhone 3GS วันที่ 19 มิถุนายน 2009 สำหรับเวอร์ชันนี้มาพร้อมฟีเจอร์เด่น ๆ อย่าง
Voice Control, สามารถคัดลอกและวางข้อความ และส่ง MMS ได้ เมื่อมีอนเซ่นเคยผู้ใช้ iPhone
สามารถอัปเดตได้ฟรี แต่ผู้ใช้ iPod touch จะต้องจ่ายเงิน 9.95 เหรียญ หรือประมาณ 390 บาท
เพื่ออัปเดต iOS 3.0 - 3.1 และ 4.95 เหรียญ หรือประมาณ 169 บาท สำหรับอัปเดต iOS 3.2



รูปที่ 2.7: iOS 3

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 4

ถือว่าเป็นระบบปฏิบัติการรุ่นแรกของแอปเปิล ที่เปลี่ยนชื่อเรียกจาก iPhone OS มาเป็น iOS โดย iOS 4 เปิดตัวครั้งแรกที่งาน WWDC 2010 (วันที่ 7 มิถุนายน 2010) หลังจากนั้นก็ออกวางจำหน่ายพร้อม iPhone 4 วันที่ 21 มิถุนายน 2010 และมีฟีเจอร์ใหม่ ๆ ที่น่าสนใจมากmany เช่น Multitasking, ไฟลเดอร์, FaceTime, iBook และ iOS 4.2.1 เป็นรุ่นแรกรองรับการใช้งานบน iPad มาถึงเวอร์ชันนี้ทุกอุปกรณ์ iOS ของแอปเปิล สามารถอัปเดตได้ฟรี



รูปที่ 2.8: iOS 4

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 5

เปิดตัวพร้อม iPhone 4S ที่งาน WWDC 2011 (วันที่ 6 มิถุนายน 2011) และหลังจากนั้น ก็อกร่างจำหน่ายพร้อม iPhone 4S วันที่ 12 ตุลาคม 2011 มาพร้อมฟีเจอร์ใหม่ที่น่าสนใจหลายอย่าง เช่น Siri, iCloud, Notification Center, iMessage, Reminders และ Newsstand เป็นต้น



รูปที่ 2.9: iOS 5

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 6

เปิดตัวพร้อม iPhone 5 และ iPad mini ที่งาน WWDC 2012 (วันที่ 11 มิถุนายน 2012) และอุ่นเครื่อง iPhone 5 วันที่ 19 กันยายน 2012 สำหรับฟีเจอร์ใหม่ที่มาพร้อม iOS 6 เช่น การเปลี่ยนไปใช้ระบบแผนที่ของแอปเปิลเอง, สามารถ Facetime ผ่านระบบเซลลูลาร์, ถ่ายภาพแบบพาโนรามา, คีย์บอร์ดภาษาไทยแบบ 4 แฉว, Passbook, อินทิเกรท Facebook, รองรับ LTE และแอปฯ นาฬิกาสำหรับ iPad



รูปที่ 2.10: iOS 6

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 7

อีกกว่าสามัญของแอปเปิล ที่มีการยกเครื่องเปลี่ยนดีไซน์ของ iOS ใหม่ทั้งหมด เปิดตัวครั้งแรกที่งาน WWDC 2013 (วันที่ 10 มิถุนายน 2013) และปล่อยให้อัปเดตวันที่ 18 กันยายน 2013 โดยผู้ที่รับหน้าที่ดูแลการปรับโฉม iOS 7 ครั้งนี้ ก็คือ Jonathan Ive เปลี่ยนมาใช้ดีไซน์แบบ Flat Design, ไอคอนใหม่ทั้งหมด, มี Control Center, AirDrop, Photos, iTunes Radio และ CarPlay



รูปที่ 2.11: iOS 7

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 8

สำหรับ iOS 8 เปิดตัวครั้งแรกที่งาน WWDC 2014 (วันที่ 2 มิถุนายน 2014) และปล่อยให้อัปเดตวันที่ 17 กันยายน 2014 พร้อมการเปิดตัว iPhone 6, 6 Plus และ iPad Air 2 โดยหน้าตาต่าง ๆ ของ iOS 8 ยังคงเหมือนกับ iOS 7 แต่ปรับปรุงประสิทธิภาพการทำงานให้ดีขึ้นกว่าเดิม รวมถึงเพิ่มฟีเจอร์การใช้งานต่าง ๆ เช่น มาอีกอย่างมากmany เช่น iCloud Drive, Apple Pay, Apple Music, QuickType, Family Sharing และแอปฯ Health เป็นต้น



รูปที่ 2.12: iOS 8

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 9

เปิดตัวครั้งแรกที่งาน WWDC 2015 (วันที่ 8 มิถุนายน 2015) และปล่อยให้อัปเดตวันที่ 16 กันยายน 2015 สำหรับเวอร์ชันนี้เน้นปรับปรุงเพิ่มประสิทธิภาพการใช้งานให้ดีขึ้นกว่าเดิม รวมถึงเปลี่ยนแปลงและเพิ่มฟีเจอร์ใหม่ที่ช่วยให้ผู้ใช้สะดวกสบายมากขึ้น โดยฟีเจอร์หลัก ๆ อย่างเรียนรู้จากพกติกรรมผู้ใช้งาน เพื่อตอบสนองสิ่งที่ผู้ใช้งานต้องการมากที่สุด โดยฟีเจอร์ใหม่ที่มาพร้อม iOS 9 เช่น Siri มีความแม่นยำและทำงานได้รวดเร็วกว่าเดิม, รองรับการใช้งาน 2 หน้าจอสำหรับ iPad, เพิ่มแอปฯ News, ปรับปรุงแอปฯ Note, Spotlight ให้สามารถค้นหาสิ่งต่าง ๆ ได้มากขึ้น



รูปที่ 2.13: iOS 9

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 10

เปิดตัวครั้งแรกที่งาน WWDC 2016 (วันที่ 13 มิถุนายน 2016) และปล่อยให้อัปเดตวันที่ 13 กันยายน 2016 แอปเปิลบอกว่า iOS 10 มาพร้อมการปรับปรุงครั้งใหญ่ ภายในงานได้พูดถึง 10 ฟีเจอร์หลัก ๆ ที่มีการเปลี่ยนแปลงหลาย ๆ อย่าง เช่น อินเทอร์เฟซมีการปรับเปลี่ยนใหม่ให้ดูสวยงามขึ้นกว่าเดิม, ระบบแจ้งเตือนแบบใหม่, 3D Touch ที่ใช้งานได้หลากหลายขึ้น, รีติชีฟ์และ Apple Maps, Apple Music และ News เป็นต้น



รูปที่ 2.14: iOS 10

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 11

เปิดตัวครั้งแรกที่งาน WWDC 2017 (วันที่ 5 มิถุนายน 2017) และปล่อยให้อัปเดตวันที่ 19 กันยายน 2017 สำหรับเวอร์ชันนี้แอปเปิลได้ให้คำนิยามไว้ว่า ”เป็นก้าวใหญ่สำหรับ iPhone ก้าวกระโดดสำหรับ iPad” โดยเน้นการปรับปรุงเพิ่มความสามารถรอบด้าน ตอบโจทย์การใช้งานให้ดีขึ้นกว่าเดิม โดยของใหม่ที่มาพร้อม iOS 11 ที่น่าสนใจ เช่น Siri สามารถแปลภาษาได้, ปรับปรุง Control Center ใหม่, Apple Pay รองรับการโอนเงินได้, เพิ่ม API สำหรับระบบ AR เทคโนโลยีที่ผสมผสานระหว่างความเป็นจริงและโลกเสมือน รวมถึงฟีเจอร์ป้องกันการใช้โทรศัพท์ขณะขับรถ



รูปที่ 2.15: iOS 11

ที่มา : <https://mobile.kapook.com/view5432.html>

iOS 12

เปิดตัวครั้งแรกที่งาน WWDC 2018 (วันที่ 4 มิถุนายน 2018) และจะปล่อยให้อัปเดตในช่วงเดือนกันยายน 2018 หรือหลังจากเปิดตัว iPhone รุ่นใหม่ไปแล้ว สำหรับ iOS 12 ยังคงพัฒนาอย่างต่อเนื่อง โดยออกแบบมาเพื่อทำให้การทำงานประจำวันดำเนินไปอย่างรวดเร็วและตอบสนองฉับไวขึ้น iOS 12 จะเปลี่ยนวิธีการที่ผู้ใช้ iOS มองเห็นโลกโดยใช้ AR ทำให้การสื่อสารมีความสนุกสนานและสื่ออารมณ์ด้วย Memoji, Group FaceTime และ Screen Time ช่วยทำความเข้าใจและจัดการกับเวลาการใช้งานอุปกรณ์ iOS ลดการติดมือถือ นอกจากนี้ยัง iOS 12 เปิดตัว Siri Shortcuts ซึ่งจะทำให้ Siri สามารถทำงานร่วมกับแอปฯ ได้ก็ได้



รูปที่ 2.16: iOS 12

ที่มา : <https://mobile.kapook.com/view5432.html>

2.1.4 ความรู้พื้นฐาน Ionic Framework

Ionic Framework [6] คือเครื่องมือในการสร้าง Mobile Application เป็นเครื่องมือสร้างแอปมือถือที่สามารถสร้างที่เดียวสามารถใช้งานได้บนระบบปฏิบัติการ iOS, Android และ Windows ซึ่งก็จะใช้งานร่วมกับ Framework ตัวอื่น ๆ ได้ คือ Angular และ Apache Cordova ในตอนสุดท้ายเพื่อให้ทั้งแอปที่เขียนมาใช้ได้กับทุกระบบปฏิบัติการ

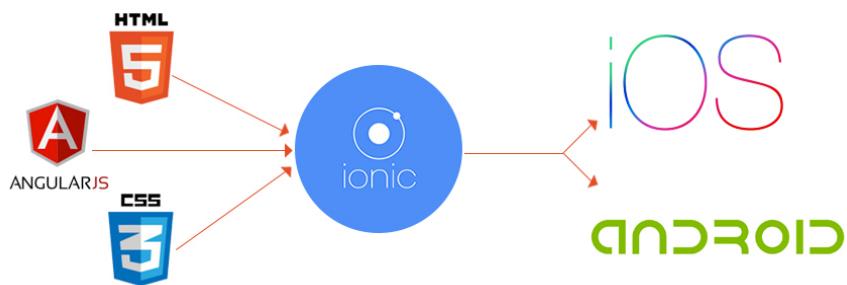
2.1.4.1 เทคโนโลยีที่ใช้ในการพัฒนา Ionic Framework

Ionic Framework พัฒนา Frontend ด้วยภาษา HTML , CSS , JavaScript และถูก Build เป็น Application ด้วย Cordova

- HTML5 คือ คือ ภาษาマークอป ที่ใช้สำหรับเขียน website ซึ่ง HTML5 นี้เป็นภาษาที่ถูกพัฒนาต่อมาจากภาษา HTML และพัฒนาขึ้นมาโดย WHATWG (The Web Hypertext Application Technology Working Group) โดยได้มีการปรับเพิ่ม Feature หลายอย่างเข้ามาเพื่อให้ผู้พัฒนาสามารถใช้งานได้ง่ายมากยิ่งขึ้น
- CSS3 คือ สเตล์ชีท เป็นภาษาที่ใช้เป็นส่วนของการจัดรูปแบบการแสดงผลของ HTML พูด

ง่ายๆ คือทำให้การแสดงผลของ HTML ให้สวยงาม

- AngularJs คือ JavaScript Framework รูปแบบหนึ่งที่พัฒนามาจาก Google หน้าที่ของ มันคือเป็น engine ที่ใช้ควบคุมในส่วน front end ของเว็บได้เป็นอย่างดีมีการทำงานแบบ Model View Controller (MVC)

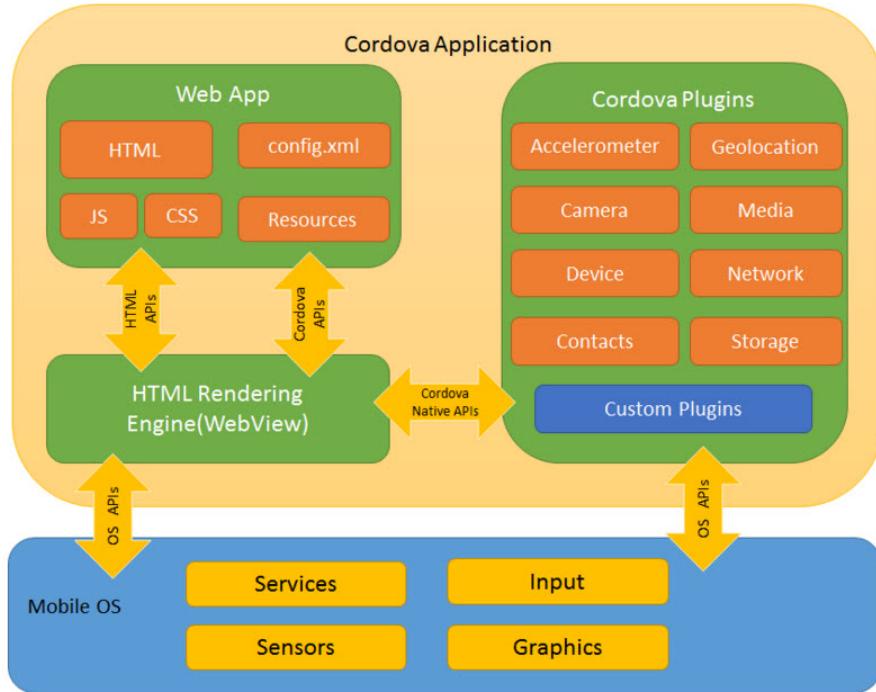


รูปที่ 2.17: การทำงานของ Ionic Framework

ที่มา : <http://blog.prscreative.com/what-is-ionic/>

2.1.4.2 การทำงานของ Cordova Application

การทำงานของ Hybrid Mobile App คือในช่อง Web App จะมี HTML, CSS, Javascript และไฟล์ภาพ เสียงต่างๆ (Resource) แล้วทำงานผ่าน Web View ที่ PhoneGap จัดเตรียมให้ หากต้องการเข้าถึงส่วนอื่นๆ ก็เรียกใช้ Plug-in หรือจะพัฒนา Custom Plug-in ขึ้นมาใช้เองได้ ดังรูปที่ 2.18



รูปที่ 2.18: Cordova Application

ที่มา : <http://blog.prscreative.com/what-is-ionic/>

โดย Cordova มีหน้าที่ห่อหุ้มแอปพลิเคชันไว้ และทำหน้าที่ติดต่อกับ Hardware ของ Mobile เป็นหลัก เพราะมี API ติดต่อกับ Hardware โดยตรง เช่น Camera, Contacts, Media, Network

2.1.4.3 ความแตกต่างระหว่าง PhoneGap/Cordova

PhoneGap และ Cordova มีลักษณะคล้ายกัน เนื่องจากมันเกื้อบจะเหมือนกัน ต่างกันเพียง เรื่องของลิขสิทธิ์ และการนำไปใช้งาน

- 1) PhoneGap : ในยุคแรกถูกพัฒนาโดย Nitobi เปิดให้ใช้งานแบบ Open Source ซึ่งได้รับ ความนิยมในการนำมาใช้เป็นเทคโนโลยี Hybrid ซึ่งต่อมาถูกซื้อโดยบริษัท Adobe เพื่อนำ มาเสริมทัพให้กับโปรแกรม Adobe Dreamweaver เพื่อให้สั่ง Build app จากโปรแกรม Dreamweaver ให้ลองรับหลาย Platform ได้ แต่มีค่าลิขสิทธิ์โปรแกรม
- 2) Cordova : เกิดขึ้นจากการตกลงกันระหว่าง Adobe และ Nitobi ด้วยแนวคิดที่อยากให้ PhoneGap เป็น Open Source ต่อไป จึงได้มีการตกลงกันให้นำโค๊ดของ PhoneGap ไป ตั้งเป็นชื่อใหม่ นั่นก็คือ Cordova เพื่อมอบให้ Apache Foundation ไปดูแลลูกน้ำไปใช้ใน

โครงการ Hybrid mobile application หลายโครงการ เช่น AppGyver, Ionic framework

2.1.4.4 เริ่มต้นการใช้งาน

การเริ่มต้นใช้งาน Ionic Framework สามารถศึกษาการติดตั้งได้ที่ <https://ionicframework.com/docs/v3/intro/installation/> ซึ่งเริ่มต้นเราจะต้องทำการติดตั้ง Cordova CLI และ Ionic CLI ผ่าน npm ก่อน

```

1 $ npm install -g cordova
2 # ติดตั้งCordova CLI
3 $ npm install -g ionic
4 # ติดตั้งIonic CLI
5 $ ionic start ชื่อโปรเจค
6 # สร้างโปรเจคไอโอนิกมีลักษณะเป็นBlank
7 $ cd ชื่อโปรเจค
8 # เข้าไปในโปรเจคที่เราสร้าง
9 $ ionic serve
10 # รันไอโอนิกแบบlocalhost:8000

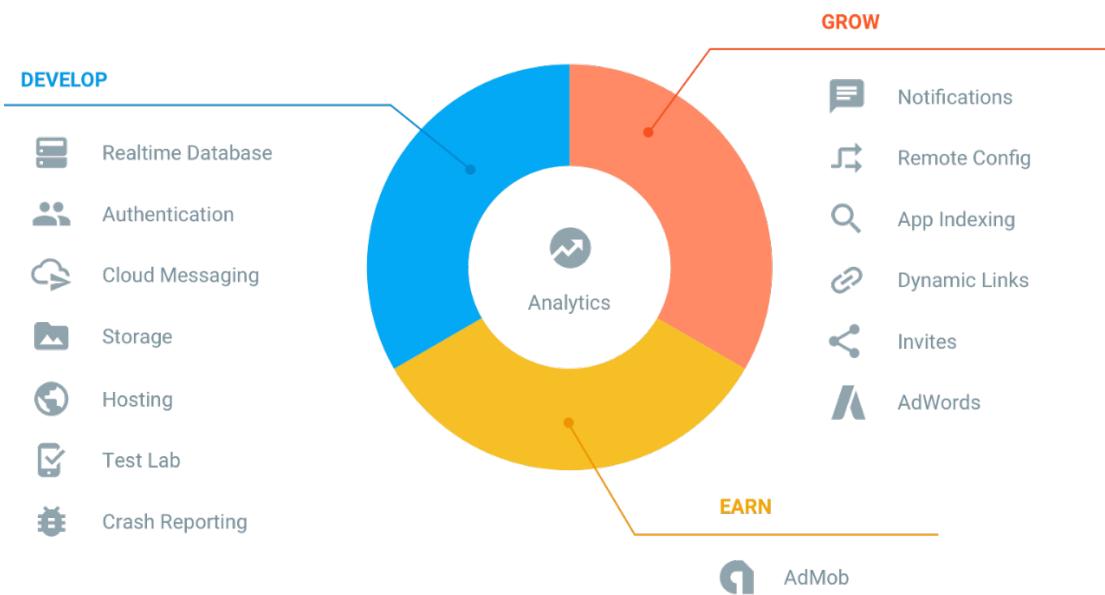
```

รูปที่ 2.19: แสดงการติดตั้ง cli

2.1.5 Firebase

Firebase [7] คือ บริการ Backend และ แพลตฟอร์ม ครบวงจรสำหรับนักพัฒนาแอปพลิเคชัน และโปรแกรมประยุกต์บนเว็บแพลตฟอร์มที่มีเครื่องมือและโครงสร้างพื้นฐานที่ได้รับการออกแบบมาเพื่อช่วยให้นักพัฒนาสามารถสร้างแอปพลิเคชันพลิเคชันที่มีคุณภาพสูง Firebase (ไฟร์เบส) ถูกสร้างขึ้นจากคุณสมบัติเสริมว่านักพัฒนาสามารถผสมและจับคู่เพื่อให้พอดีกับความต้องการของตน บริษัท ก่อตั้งขึ้นในปี 2011 โดยแอนดรูว์และเจมส์ เทมป์ลิน สินค้าเริ่มต้น Firebase เป็นฐานข้อมูลเรียลไทม์ซึ่งมี API ที่ช่วยให้นักพัฒนาในการจัดเก็บและซิงค์ข้อมูล ดังรูป 2.20

Google Firebase 2.0 มีการพัฒนาจากบริการ Backend เก็บข้อมูลอย่างเดียว มาเป็น แพลตฟอร์มครบวงจรสำหรับนักพัฒนาแอปพลิเคชัน (รองรับ iOS, Android, Web) และรองรับบริการทุกอย่างที่นักพัฒนาแอปต้องการใช้งาน



รูปที่ 2.20: Firebase 2.0

ที่มา : www.mindphp.com/คุ้มครอง/73-คืออะไร/3921-what-is-firebase-backend.html

2.1.5.1 บริการหลักของไฟร์เบส

- Realtime Database จัดเก็บและซิงค์ข้อมูลระหว่างผู้ใช้และอุปกรณ์ต่างๆแบบเรียลไทม์โดยใช้ฐานข้อมูล NoSQL ที่ออกแบบมาสำหรับคลาวด์ ซิงค์ข้อมูลที่อัปเดตระหว่างอุปกรณ์ที่เชื่อมต่อเป็นมิลลิวินาทีและข้อมูลจะยังคงมีอยู่ถ้าแอพพลิเคชันอफ์ไลน์
- Authentication จัดการบัญชีผู้ใช้ด้วย Firebase Auth ซึ่งใช้งานง่ายและปลอดภัยมีวิธีการหลายในการสร้างบัญชีผู้ใช้และตรวจสอบความถูกต้อง ได้แก่ อีเมล/รหัสผ่าน, ผู้ให้บริการบุคคลที่สาม เช่น Google หรือ Facebook
- Cloud Storage จัดเก็บภาพเสียงวิดีโอหรือเนื้อหาอื่น ๆ เช่น รูปภาพโปรดайл์ฟ์ผู้ใช้ หรือวิดีโอทัศน์ต่างๆ เป็นต้น ซึ่งมีความปลอดภัยในการอัปโหลดไฟล์และดาวน์โหลดสำหรับแอพพลิเคชัน
- Hosting ใช้ในการเผยแพร่เว็บไซต์ โดยเนื้อหาภายในเว็บเป็นเดชบอร์ด รายงานข้อมูลต่างๆ ของผู้ใช้ ซึ่งต้องทำการเข้าสู่ระบบก่อน
- Crashlytics เป็นบริการล่าสุดที่กูเกิลได้เข้าควบรวมเข้ามาไว้ในบริการไฟร์เบส สามารถรายงานข้อมูลข้อผิดพลาดได้อย่างมีประสิทธิภาพ เปิดใช้งานการวิเคราะห์แบบเรียลไทม์เพื่อช่วยให้

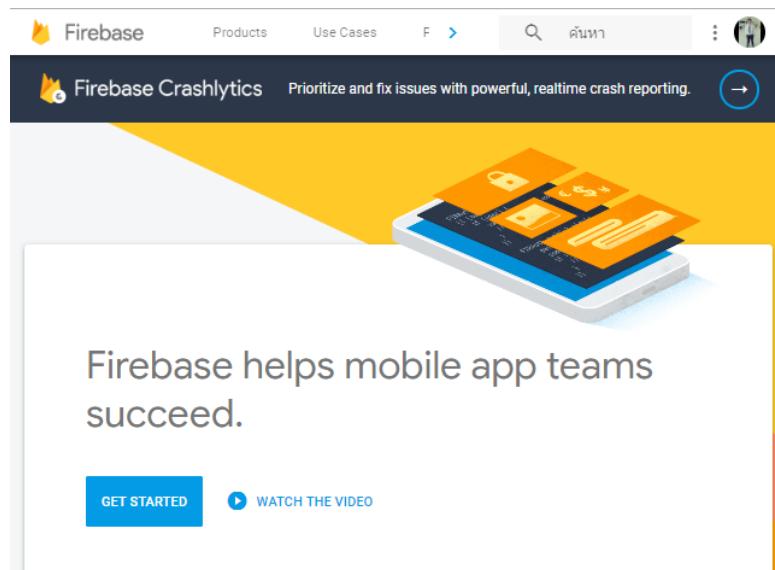
เข้าใจสิ่งที่เกิดขึ้นในแอพพลิเคชัน เครื่องมือวิเคราะห์ข้อมูลจะให้ข้อมูลเชิงลึก

- Cloud Firestore เป็นบริการในส่วนของ Database ที่ใช้ระบบฐานข้อมูลแบบ NoSQL ที่เป็นแบบ Document Database และเป็นการนำเอาข้อดีต่างๆ ของบริการด้านฐานข้อมูลของ Realtime Database มาปรับปรุงพัฒนาต่อและเพิ่มความสามารถขึ้นไปมากขึ้น ซึ่งผู้เขียนจะได้กล่าวถึงในบทถัดไป

2.1.5.2 การพัฒนา Cloud Firestore

การพัฒนา Cloud Firestore แบ่งออกแบบ 5 ขั้นตอน ดังนี้

- การสร้าง Cloud Firestore เพื่อใช้งานในโครงการ ในขั้นตอนแรกทำการสร้าง Database เพื่อที่จะใช้งาน Cloud Firestore ก่อน โดยใช้บัญชี Gmail ที่มาเข้าไปที่เว็บ <https://firebase.google.com> ดูรูปภาพที่ 2.21



รูปที่ 2.21: เว็บ <https://firebase.google.com>

ที่มา :<https://medium.com/20scoops-cnx/เข้มข้นกับ-firebase-cloud-firestore-ระบบฐานข้อมูลที่เปิดตัวใหม่ล่าสุดจาก-firebase-แบบจัดเต็ม-d001e43e2be7>

- ติดตั้ง SDKs เพื่อใช้งาน Cloud Firestore โดย SDKs ที่ Firebase ได้เตรียมไว้ให้ สามารถดูรายละเอียดได้ที่ <https://firebase.google.com/docs/firestore/quickstart>
- ออกแบบโครงสร้างและการจัดการข้อมูล

- ระบบฐานข้อมูลของ Cloud Firestore จะเป็น NoSQL แบบ Document ซึ่งจะแตกต่างจากระบบฐานข้อมูลแบบ SQL โดยจะไม่มีตาราง ไม่มีเวลา แต่เก็บข้อมูล ภายใน Document จะเก็บแบบ Key-value โดยแต่ละ Document จะถูกเก็บไว้ใน Collection ซึ่งใน Document สามารถมี Subcollection ได้
- Collection เป็นการเรียกชื่อแทนของการเก็บห้องๆ เอกสารไว้ด้วยกัน เช่น เก็บข้อมูลของ User จำนวนมากไว้ด้วยกัน จึงตั้งชื่อ Collection ว่า Users ซึ่งใน Collection เดียว กันผู้ใช้งานสามารถใส่ข้อมูลที่แตกต่างชนิดกันในแต่ละ Key และ Document ได้ โดยในแต่ละ Key และ Document จะมีอิสระในการใส่ข้อมูล แต่ควรใส่ข้อมูลในแต่ละ Key ของ Document เป็นประเภทเดียวกัน เพราะจะทำให้การค้นหาและการจัดเรียงลำดับของข้อมูลนั้นง่ายขึ้น
- Subcollection สามารถสร้าง Subcollection ของ Subcollection ไปได้เรื่อยๆ โดย Cloud Firestore ว่าสามารถซ้อนกันไปได้ 100 ลำดับขึ้น

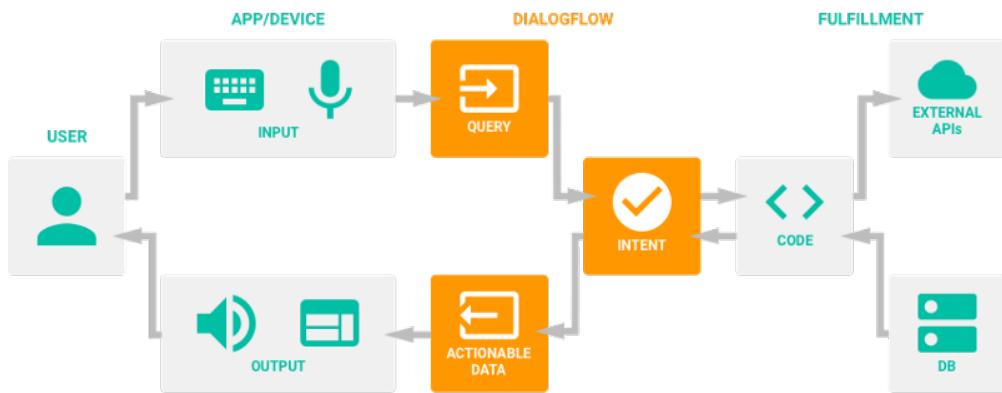
4) การรับและส่งข้อมูล การรับและส่งข้อมูลจาก Cloud Firestore จะมี 2 วิธี โดยจะสามารถใช้ได้ทั้งการรับข้อมูลและการส่งข้อมูล

- การรับข้อมูลเพียงครั้งเดียวจะเป็นการรับข้อมูลเมื่อมีจุดประสงค์ที่จะไม่ต้องการรับรู้การเปลี่ยนแปลงของข้อมูล ซึ่ง ณ ขณะนั้นข้อมูลมีค่าเป็นอะไรมีจะได้ค่านั้นมา หากมีการเปลี่ยนแปลงข้อมูลในภายหลัง ผู้ใช้งานต้องเป็นผู้จัดการรับข้อมูลล่าสุดเอง โดยวิธีการรับข้อมูลเพียงครั้งเดียวจะใช้ Method Get()
- การรับข้อมูลแบบ Realtime update จะเป็นการรับข้อมูลเมื่อผู้ใช้งานมีจุดประสงค์ที่จะต้องการรับรู้การเปลี่ยนแปลงของข้อมูล ซึ่ง ณ ขณะที่ข้อมูลเกิดการเปลี่ยนแปลง จะมีการรับข้อมูลที่เกิดการเปลี่ยนแปลงโดยอัตโนมัติ โดยในครั้งแรกที่มีการรับข้อมูล จะสร้าง Initialize Instance และทุกครั้งที่ข้อมูลมีการเปลี่ยนแปลงก็จะส่ง Callback ให้กับ Listener โดยผ่าน Method onSnapshot()

5) การป้องกันและความปลอดภัยของข้อมูลข้อมูลใน Cloud Firestore ได้มีการออกแบบให้สามารถกำหนดกฎของความปลอดภัยต่างๆ ได้โดยผ่าน Firebase Console ซึ่งหากใช้ Cloud Firestore ผู้ใช้งานสามารถมาทำเรื่องการป้องกันและรักษาความปลอดภัยของข้อมูลเพียงที่เดียว สามารถใช้ได้ทั้งหมดไม่ว่าจะเป็น Web และ Mobile ส่วนฝั่ง Server ก็สามารถใช้ IAM ใน Google Cloud Platform มาจัดการความปลอดภัยสำหรับ Cloud Firestore ได้

2.1.6 Dialogflow

Dialogflow คือ platform สำหรับสร้าง chatbot ของ Google ที่ใช้ machine learning ด้าน Natural Language Processing (NLP) มาช่วยในการทำความเข้าใจถึงความต้องการ (intent) และสิ่งที่ต้องการ (entity) ในประโยคสนใจของผู้ใช้งาน และตอบคำถามตามความต้องการของผู้ใช้งาน ตามกฎ หรือ flow ที่ผู้พัฒนาวางเอาไว้ ซึ่ง Dialogflow จะช่วยเพิ่มความยืดหยุ่นของประโยคที่ chatbot รับมา ว่าไม่จำเป็นต้องตรงตามเงื่อนไข แบบ rule based ก็สามารถเข้าใจถึงความต้องการของผู้ใช้งานได้



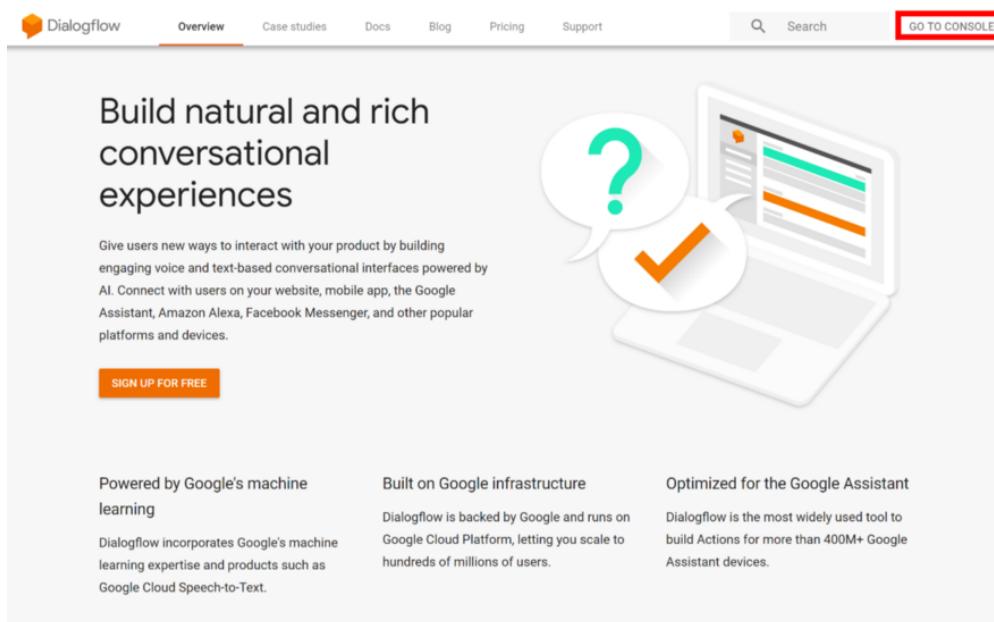
รูปที่ 2.22: รูปแบบการส่งข้อมูลผู้ใช้ไปยังแซทบอท

ที่มา : <https://dialogflow.com/docs/agents>

2.1.6.1 เริ่มต้นใช้งาน Dialogflow

1) ลงทะเบียน/ล็อกอินเข้า Dialogflow

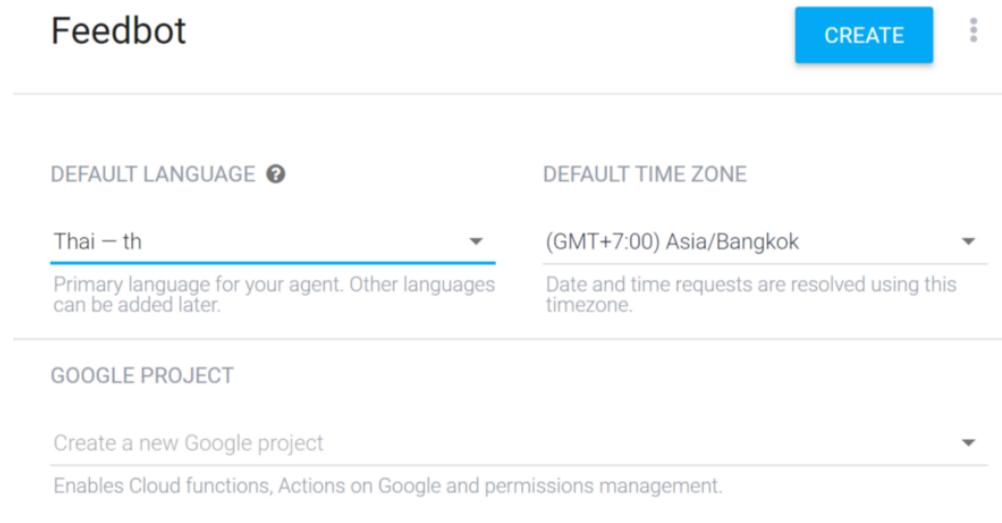
ในการสร้าง Agent เราต้องลงทะเบียนเข้าใช้งานก่อนนะ โดยไปยังหน้าเว็บของ Dialogflow และกดที่ Go Console จากนั้นก็เข้าสู่ขั้นตอนการ Login หรือลงทะเบียน



รูปที่ 2.23: หน้าหลักของ Dialogflow

2) สร้าง Agent

หลังจาก Login สำเร็จเราก็จะเจอกับ Workplace ในการทำแซทบอทลงทะเบียนไปที่เมนูด้านซ้าย และเลือก Create Agent ก็จะพบกับหน้าจอสำหรับตั้งค่าแซทบอทของเรา โดยต้องสามารถตั้งชื่อ ภาษา และ Timezone ที่ต้องการ



รูปที่ 2.24: หน้า Create Agent

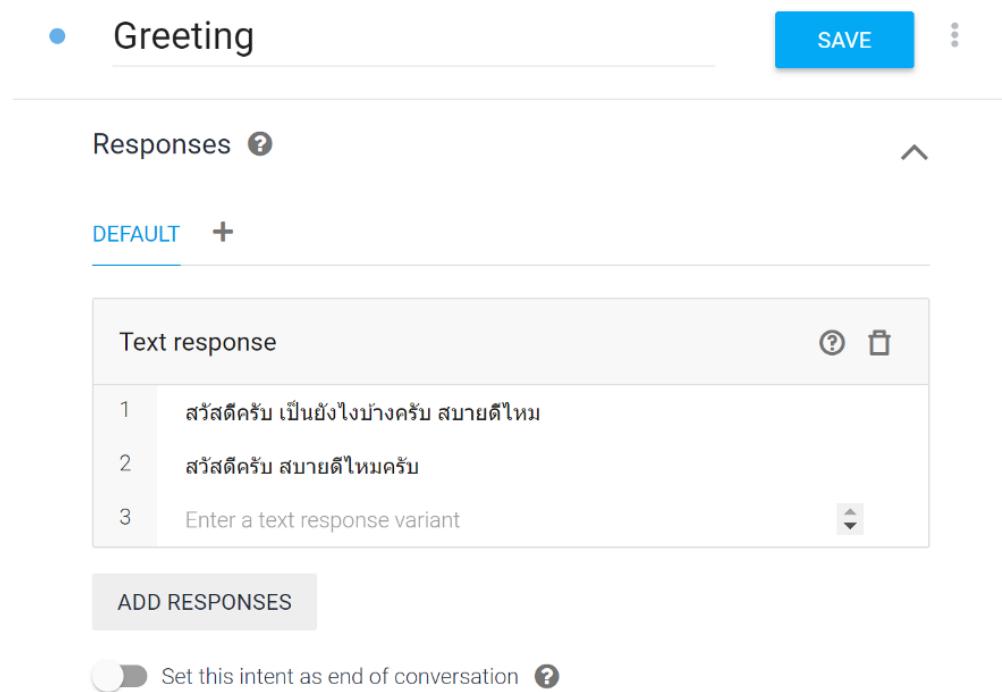
3) สอนบทให้พูดทักษะ

เมื่อสร้างเสร็จแล้วเราจะพบกับ Default Intents มา 2 ตัวก็คือ Default Welcome Intent และ Default Fallback Intent มาให้ ตอนนี้ยังไม่ต้องสนใจมันนะ เพราะเราจะลองสร้าง Intent ใหม่โดยตั้งชื่อว่า Greeting ในการสร้างให้กดที่ปุ่ม Create Intent และตั้งชื่อ Intent นี้ว่า Greeting โดยเราตั้งใจจะให้ Intent นี้ โต้ตอบกับผู้ใช้งาน เวลาที่ผู้ใช้ต้องการที่จะทักทายกับzechbot ที่เราสร้างขึ้นมา จากนั้นไปที่ Training phrases หรือแนวประโยคที่เราจะให้zechbotเข้าใจว่า ถ้าพูดด้วยประโยคประมาณนี้ แสดงว่าผู้ใช้งานตั้งใจจะสื่อถึง Intent นี้ ถ้าดูจากตัวอย่างจะพบว่ามีการระบุ phrases ไว้ว่า สวัสดี, สวัสดีจ้ะ, สวัสดีจ้า

	Training phrases	?
	Add user expression	
	สวัสดีจ้า	
	สวัสดีจะ	
	สวัสดี	

รูปที่ 2.25: หน้า Intent ส่วน Training phrases

จากนั้นเราจะลองไปตั้งค่า Responses หรือประโยคที่เราต้องการให้เขตบทตอบกลับ ในกรณีที่บอทสามารถจับได้ว่าผู้ใช้งานตั้งใจจะสื่อถึง Intent นี้ สำหรับตัวอย่างจะพบว่า ถ้าผู้ใช้พิมพ์ สวัสดี, สวัสดีจะ, สวัสดีจ้า ตาม Training phrases เราจะให้เขตบทของเราตอบกลับว่า สวัสดีครับ เป็นยังไงบ้างครับ สบายดีไหม หรือ สวัสดีครับ สบายดีไหมครับ โดยจะสุ่มขึ้นมาว่าจะตอบอันไหน

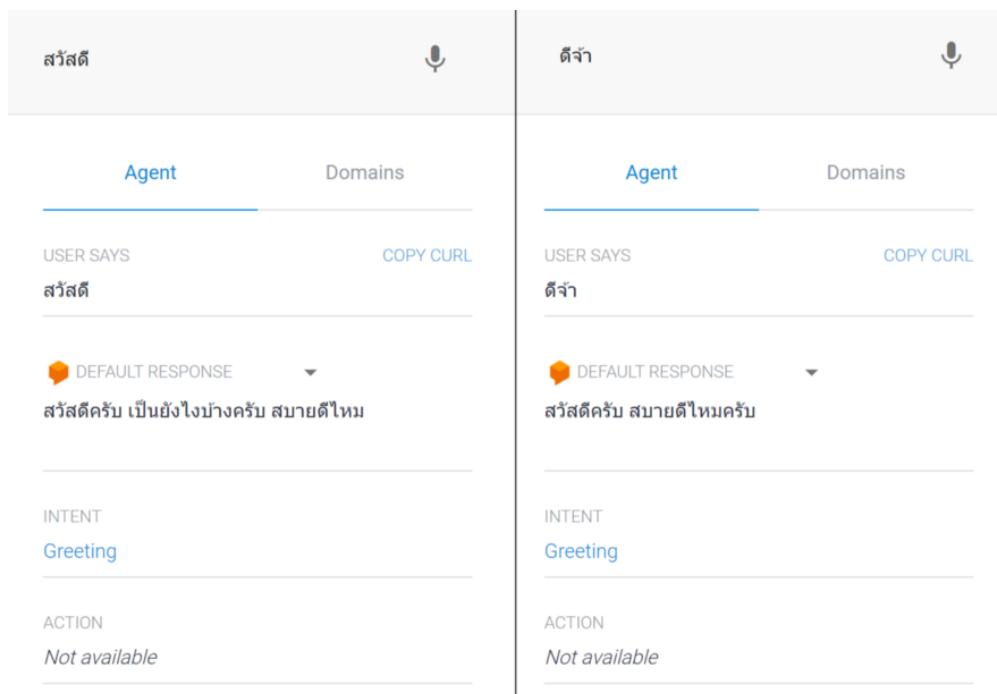


รูปที่ 2.26: หน้า Intent ส่วน Responses

ตรงส่วนของ Responses เราสามารถเพิ่มข้อความ หรือเพิ่ม balloon message ให้ต่อ กัน หลายๆ อันได้ โดยกดที่ปุ่ม Add Responses และถ้าต้องการตั้งค่าว่า intent นี้เป็น intent สุดท้ายในการสนทนากัน ก็สามารถเปิด Checkbox Set this intent as end of conversation ซึ่งเดียวเราค่อนมาคุยกันแบบละเอียดอีกครั้ง ตอนที่ต้องทำ Contexts กันอีกครั้ง

4) ทดสอบคุยกับบอท

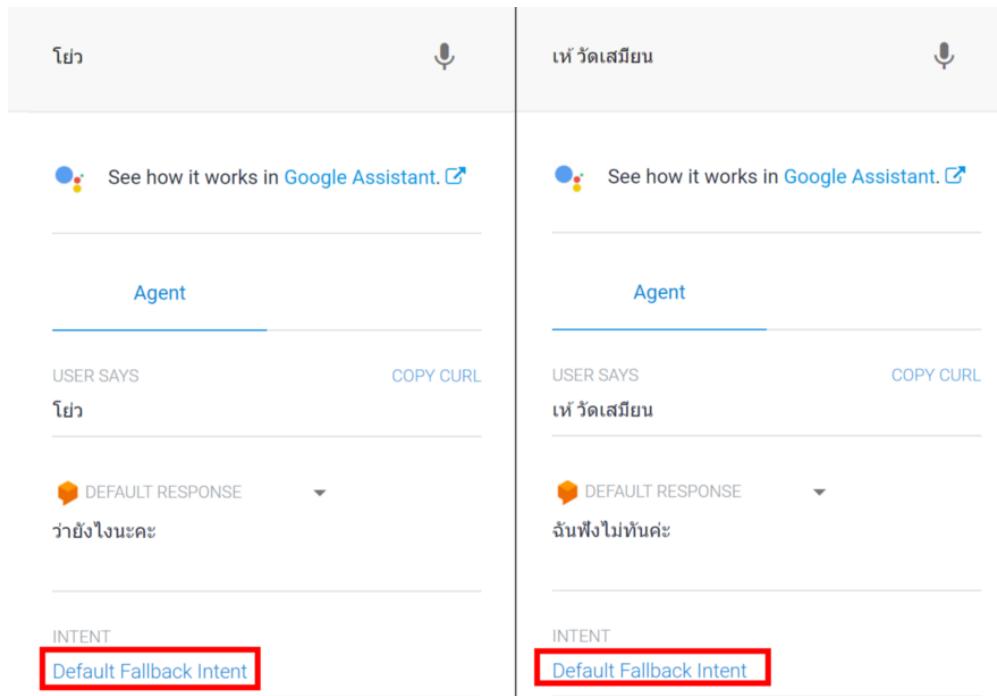
หลังจากเราลองทำ Greeting Intent เสร็จ ก็ถึงเวลาที่เราจะลองทดสอบการใช้งานกัน ซึ่ง เราสามารถทดสอบได้ผ่านกล่องสนทนาที่อยู่ทางด้านขวา โดยลองพิมพ์คำว่า สวัสดี ลงไป ก็จะพบว่าแทบทบทจะตอบเรากลับมาว่า สวัสดีครับ เป็นยังไงบ้างครับ สบายดีไหม ตามที่เรา ตั้งค่าไว้ใน Responses นั้นเอง



รูปที่ 2.27: ทดสอบคุยกับบท

ถ้าดูจากภาพ 2.27 จะพบว่า ถ้าเราพิมพ์คำบางคำที่ไม่ได้มีอยู่ใน Training phrases อย่าง คำว่า ดีจ้า ตัว Dialogflow ก็ฉลาดพอที่จะจับได้ว่านี่คือคำที่อยู่ในกลุ่มเดียวกับ สวัสดี ซึ่ง เป็นคำทักทาย ที่เรากำหนดว่ามันคือ Intent Greeting นั้นเอง

แต่ในขณะเดียวกัน คำบางคำ หรือประโยคบางประโยคตัวแรกบทของเราก็อาจจะยังไม่ เข้าใจ ว่าสิ่งที่ผู้ใช้งานต้องการจะสื่อสารอกรกما มันคือ Intent อะไร ซึ่งเวลาสร้าง Agent Dialogflow ก็จะสร้าง Default Fallback Intent ขึ้นมาให้ พื้อมากับ Responses บางส่วน ในกรณีที่แรกบทไม่สามารถหา Intent ที่เหมาะสมได้ ก็จะมาตกที่คenes ทั้งหมดตามภาพ นั้นเอง



ຮູບທີ 2.28: ທດສອບຄຸງກັບບອກ

2.1.7 Libraries moment.js

moment.js ຄື່ອ javascript library ສໍາຮັບໃໝ່ສໍາຮັບຈັດການ Date ແລະ Time ທີ່ຈະຊ່ວຍ
ອໍານວຍຄວາມສະດວກໃນເຮືອງການຈັດ Format Date ໃຫ້ຕຽງກັບທີ່ເຮົາຕ້ອງການ ມີ Feature ທີ່ຫລາກຫລາຍ
ຄຣອບຄລຸມທີ່ໃໝ່ງານ Format ທີ່ Date , Time , Timezone , Standard Time , Local Time
ເປັນຕົ້ນ ຫຶ່ງການນຳ library moment.Javascript ມາໃໝ່ງານສາມາຮັດຕິດຕັ້ງໄດ້ດັ່ງນີ້

2.1.7.1 ຄຳສັ່ງໃນການຕິດຕັ້ງ

```

1 /* install dependency */
2 npm install moment --save # npm
3 yarn add moment # Yarn
4 Install-Package Moment.js # NuGet
5 spm install moment --save # spm
6 meteor add momentjs:moment # meteor
7 bower install moment --save # bower (deprecated)

```

ຮູບທີ 2.29: ຄຳສັ່ງໃນການຕິດຕັ້ງ moment.js

2.1.7.2 รูปแบบการใช้งาน moment.js

```

1 const moment = require('moment');
2
3 const SLASH_DMY = 'DD/MM/YYYY';
4 const SLASH_DMYHMS = 'DD/MM/YYYY HH:mm:ss';
5 const SLASH_YMD = 'YYYY/MM/DD';
6 const SLASH_YMDHMS = 'YYYY/MM/DD HH:mm:ss';
7 const DASH_DMY = 'DD-MM-YYYY';
8 const DASH_DMYHMS = 'DD-MM-YYYY HH:mm:ss';
9 const DASH_YMD = 'YYYY-MM-DD';
10 const DASH_YMDHMS = 'YYYY-MM-DD HH:mm:ss';
11
12 console.log('sysdate ::==', moment());
13
14 console.log('sysdate ::==', moment().format(SLASH_DMY));
15 console.log('sysdate ::==', moment().format(
16     SLASH_DMYHMS));
17
18 console.log('sysdate ::==', moment().format(SLASH_YMD));
19
20 console.log('sysdate ::==', moment().format(
21     SLASH_YMDHMS));
22
23 console.log('sysdate ::==', moment().format(DASH_DMY));
24 console.log('sysdate ::==', moment().format(
25     DASH_DMYHMS));
26 /*
27 sysdate ::== moment("2018-07-03T21:08:38.248")
28 sysdate ::== 03/07/2018
29 sysdate ::== 03/07/2018 21:08:38
30 sysdate ::== 2018/07/03
31 sysdate ::== 2018/07/03 21:08:38
32 sysdate ::== 03-07-2018
33 sysdate ::== 03-07-2018 21:08:38
34 sysdate ::== 2018-07-03
35 sysdate ::== 2018-07-03 21:08:38
*/

```

รูปที่ 2.30: รูปแบบการใช้งาน moment.js

ทั้งหมดที่ยังไม่ใช้ทั้งหมดที่ Moment JS ทำได้ยังมีความสามารถอีกหลายอย่างที่ยังไม่ได้กล่าวถึงที่ Moment ทำได้เข้าไปดูได้ที่ <https://momentjs.com/docs/>

2.1.8 Google Maps API

Google Maps คือ บริการแผนที่ของ Google ซึ่งให้บริการ Services ที่เกี่ยวข้องกับแผนที่ทั้งหมด โดยในปัจจุบันแผนที่ของ Google นั้นมีอยู่หลากหลายประเภท อาทิ เช่น ที่เราใช้บริการแผนที่บนเว็บไซต์ หรือ App บน Smartphone โดย Services เหล่านี้เรารสามารถ เรียกใช้ได้ฟรีในกรณีที่ผ่าน Application ทั่วไป แต่ถ้าในกรณีที่เราจะมีการเรียกใช้งานในเว็บไซต์หรือแอปพลิเคชัน ที่พัฒนาขึ้นเอง Google Maps ก็จะมี API ให้ใช้งานได้เช่นเดียวกัน แต่ Services ของ Google นั้นมีข้อจำกัดในการใช้งาน ถ้าต้องการใช้ในปริมาณที่สูงขึ้นก็จะต้องซื้อ Package ที่ทาง Google Maps มีมาให้ โดยปกติจะมีการจำกัดจำนวนที่ Request เข้ามาเรียกใช้งาน สำหรับเรียกใช้แผนที่และชุด service ต่าง ๆ ของ Google เพื่อพัฒนา Application ได้เหมือนกับ Google มีบริการ features ให้เรียกใช้ดังต่อไปนี้

- การปรับแต่งแผนที่ (Styled Map)
- ชุดควบคุมแผนที่ (Map Control)
- ชุดเครื่องมือวาดภาพบนแผนที่ (Drawing)
- การนำทางจากจุดหนึ่งไปยังอีกจุดหนึ่ง (Directions Service)
- การคำนวณความสูงของจุดพิกัด (Elevation Service)
- การแปลงที่อยู่เป็นพิกัด Latitude และ Longitude (GeoCoding Service)
- การดึงข้อมูล POI (Point of Interest)
- Street View

ประโยชน์ของการใช้งาน Google Maps

- สามารถค้นหาสถานที่ต่าง ๆ ได้อย่างสะดวกรวดเร็ว
- สามารถค้นหาชื่อถนนและสีแยกได้
- สามารถค้นหาร้านอาหารในพื้นที่ที่ต้องการได้
- สามารถที่จะประชาสัมพันธ์สถานประกอบการทางธุรกิจ
- สามารถย่อหรือขยายแผนที่ทั่วโลกให้เล็กลงได้

- สามารถวางแผนเส้นทางการเดินทางไปยังพื้นที่ต่าง ๆ ได้
- สามารถใช้งานด้านระบบวิทยาในการค้นหาแหล่งเพร่เชื้อ
- สามารถทำแผนที่หรือเส้นทางไปบ้านของตนเองได้
- สามารถแสดงตำแหน่งของตนเองได้
- สามารถดูและมองเห็นแผนที่ต่าง ๆ ทั่วโลกได้อย่างรวดเร็ว

การนำ Google Maps API มาใช้งาน

สิ่งที่ต้องทำเพื่อเรียก Google Maps API มาใช้งานดังนี้

- 1) ทำการสมัครขอ API KEY โดยสามารถเข้าไปสมัครได้ที่ <https://console.developers.google.com>
- 2) เพิ่มคำสั่งในบรรทัดที่ 2 ไว้ในไฟล์ index.html

```

1 <head>
2 <script src="https://maps.googleapis.com/maps/api/js
   ?key={API_KEY}" async defer></script>
3 </head>
```

รูปที่ 2.31: เรียกใช้ API KEY

- 3) ขั้นตอนการเรียกใช้งานสำหรับแสดง Google Maps

```

1 <-- IN HTML -->
2
3 <div #map id="map"></div> // เรียกใช้ map เพื่อแสดง
4
5 <-- IN TYPESCRIPT -->
6
7 map:any; กำหนดตัวแปร// map
8
9 ionViewDidLoad() {
10   this.initMap();
11 } // เรียกใช้งานเมื่อหน้านี้ถูกเรียก
12
13 initMap() {
14   this.map = new google.maps.Map(this.mapElement.nativeElement, {
15     zoom: 7,
16     center: {lat: 41.85, lng: -87.65}
17   });
18   this.directionsDisplay.setMap(this.map);
19 } // ฟังก์ชันสำหรับกำหนดmap
20
21 <-- IN SCSS -->
22   #map {
23     height: 100%;
24   } // ขนาดของmap

```

รูปที่ 2.32: เรียกใช้งาน google maps ใน ionic framework

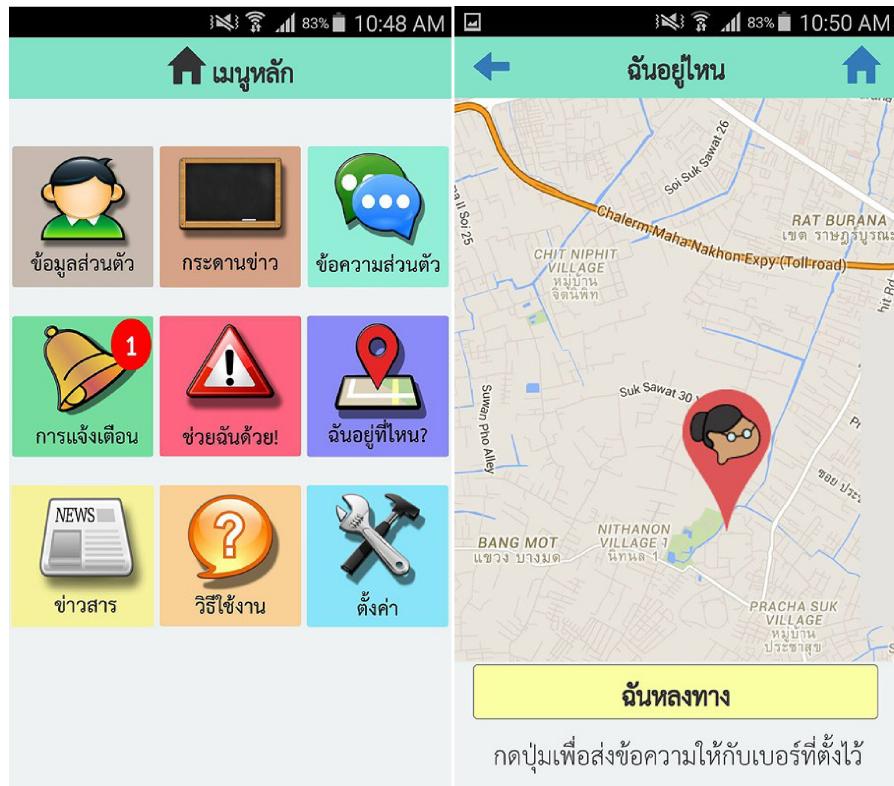
4) เรียกใช้งานด้วยคำสั่ง Google Maps เพื่อใช้งานตามต้องการ

2.2 เอกสารและงานวิจัยที่เกี่ยวข้อง

2.2.1 แอปพลิเคชันเครือข่ายสังคมออนไลน์เพื่อผู้สูงอายุ (OLDSTER)

แอปพลิเคชันเครือข่ายสังคมออนไลน์เพื่อผู้สูงอายุ (OLDSTER) [8] เป็นแอปพลิเคชันที่ถูกออกแบบมาเป็นเครือข่ายสังคมของผู้สูงอายุ โดยแอปพลิเคชันนี้ถูกออกแบบเพื่อผู้สูงอายุเป็นหลัก จึงง่ายต่อการใช้งาน มีรูปแบบที่ไม่ยุ่งยาก ไม่ซับซ้อน ตัวหนังสือเป็นแบบที่ง่ายต่อการอ่าน และขนาดไม่เล็กเกินไป มีพังก์ชันการทำงาน 9 พังก์ชันดังนี้ ข้อมูลส่วนตัว กระดาษข่าว พูดคุย ระหว่างผู้ใช้งาน การแจ้งเตือน วิธีใช้งาน ข่าวสารในหมวดต่าง ๆ เป็นต้น โดยความแตกต่างระหว่างแอปพลิเคชัน OLDSTER กับ สังคมออนไลน์ อยู่ตรงที่แอปพลิเคชันสังคมออนไลน์ จะมีการ

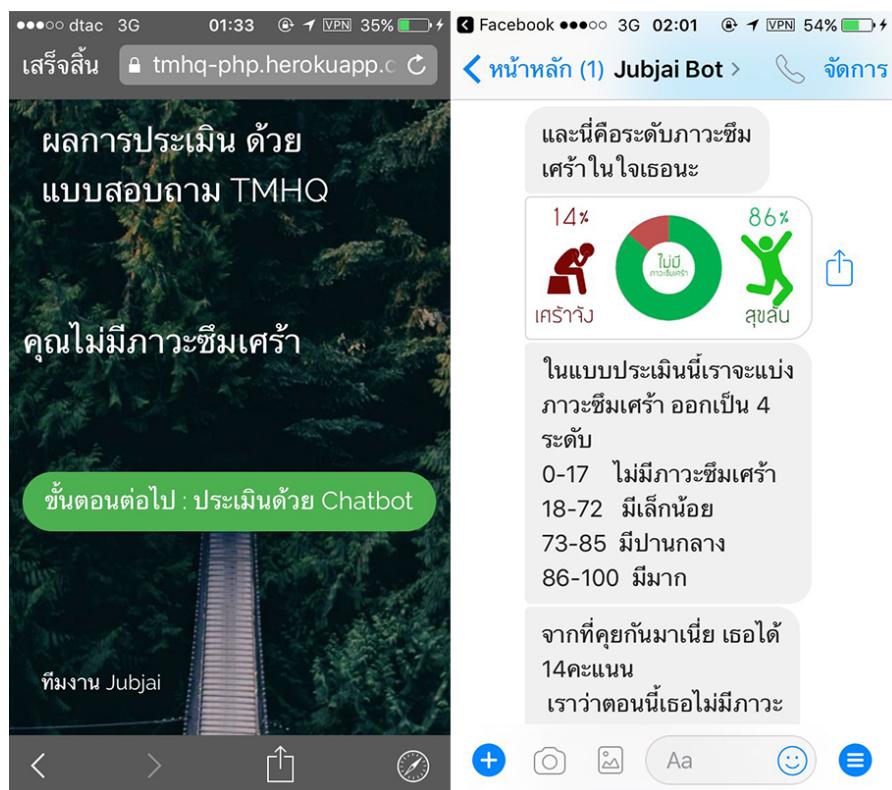
ให้ความรู้ในเรื่องโรคที่ผู้สูงอายุมักพบเจอในรูปแบบของแซทบอท และยังมีการแซทแบบกลุ่ม การค้นหาตำแหน่งของคนในครอบครัว รวมทั้งมีการแจ้งเตือนภัยทางยาอีกด้วย



รูปที่ 2.33: แอปพลิเคชัน OLDSTER

2.2.2 จับใจบอท

จับใจบอท [9] เป็นแซทบอทที่สามารถพูดคุยได้ใน Messenger ของ Facebook เป้าหมายหลักคือเป็นแซทบอทที่ช่วยประเมินอาการซึมเศร้าของผู้ใช้งานได้ หากพบว่าตัวเองมีอาการซึมเศร้าก็จะช่วยในการตัดสินใจในการพบแพทย์ได้เร็วขึ้น



รูปที่ 2.34: แชทบอทของ จับใจบอท

ที่มา : <https://techsauce.co/tech-and-biz/jubjai-depression-detection-chatbot/>

ความแตกต่างระหว่าง จับใจบอท กับแอปพลิเคชันสูญเสียมายเฟรนด์ คือจับใจบอทเป็นการประเมินอาการซึมเศร้าเพียงอย่างเดียว ส่วนแอปพลิเคชันสูญเสียมายเฟรนด์จะแนะนำสาเหตุ และวิธีดูแลรักษาในเบื้องต้นให้แก่ผู้ใช้งาน

บทที่ 3

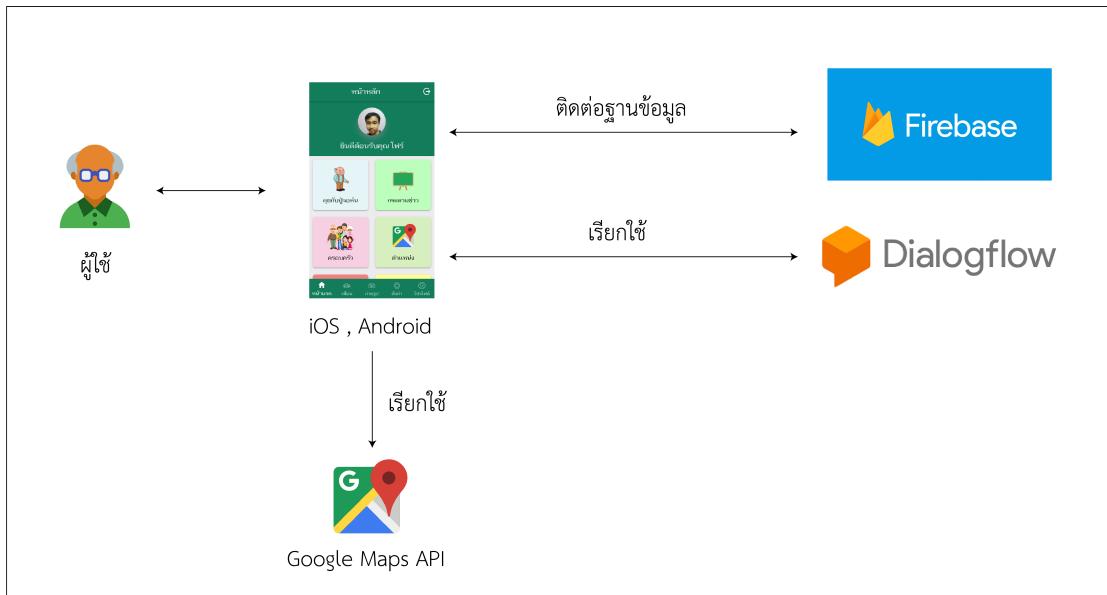
การวิเคราะห์และออกแบบระบบ

แอปพลิเคชันสูงวัยมายเฟรนด์ มีการวิเคราะห์และออกแบบระบบดังนี้

- 3.1 โครงสร้างภาพรวมของระบบ (System Architecture)
- 3.2 การวิเคราะห์ความต้องการของระบบ (System Requirements)
- 3.3 การออกแบบส่วนติดต่อผู้ใช้ (User Interface Design)
- 3.4 แผนภาพโดยแกรม
 - 3.4.1 ยูสเคสโดยแกรม (Use Case Diagram)
 - 3.4.2 ตารางแสดงคำอธิบายของผู้ใช้ (User Case Description)
 - 3.4.3 คลาสโดยแกรม (Class Diagram)
 - 3.4.4 ซีเควนโดยแกรม (Sequence Diagram)

3.1 โครงสร้างภาพรวมของระบบ (System Architecture)

ความหมายของ System Architecture [10] หมายถึง กรอบโครงสร้างของระบบที่อธิบายความสัมพันธ์ขององค์ประกอบต่าง ๆ ไปจนถึงขั้นการเชื่อมต่อกันของระบบอย่างต่าง ๆ โดยจัดกลุ่มองค์ประกอบไว้ในหลาย ๆ ลักษณะเพื่อให้ผู้เกี่ยวข้อง (Stakeholder) จากพื้นฐานสาขาอาชีพที่แตกต่าง กันสามารถทำความเข้าใจได้ง่าย เช่น การจัดแบ่งองค์ประกอบตามลักษณะการทำงานของระบบ (functional components) เป็นต้น การออกแบบ System architecture แสดงภาพรวมและเทคโนโลยีของแอปพลิเคชันสูงวัยมายเฟรนด์ มีรายละเอียดดังรูปที่ 3.1



รูปที่ 3.1: System architecture แอปพลิเคชันสูงวัยมายเพรนด์

จากรูปที่ 3.1 สามารถอธิบายโครงสร้างและเทคโนโลยีของแอปพลิเคชัน โดยเริ่มจากผู้ใช้ ติดต่อระบบผ่านระบบ iOS และ Android ซึ่งจะโหลดข้อมูลจากฐานข้อมูลมาแสดงในหน้าจอ เมื่อ ผู้ใช้ใช้งานแผนที่ระบบจะเรียกใช้งาน Google Maps API เพื่อดึงแผนที่มาแสดงที่หน้าจอ ถ้าหาก ผู้ใช้เลือกคุยกับเซ็ฟบอทจะเรียกข้อมูลจาก Dialogflow เพื่อตอบแบบเรียลไทม์

3.2 การวิเคราะห์ความต้องการของระบบ (System Requirements)

3.2.1 ความต้องการหลักของระบบ (Functional Requirements)

แบ่งความสามารถของระบบได้ดังนี้

- สามารถสมัครสมาชิกและเข้าสู่ระบบได้
- สามารถดู สร้าง แก้ไข ลบ โพสท์ได้
- สามารถคุยกับเซ็ฟบอทได้
- สามารถดูตำแหน่งของคนในครอบครัวได้
- สามารถส่งแจ้งเตือนรูปแบบการโทร หรือข้อความ ไปหาคนในครอบครัวได้
- สามารถเพิ่ม ลบ เพื่อนได้
- สามารถรับการแจ้งเตือนการทานยาได้

- สามารถดูและแก้ไขข้อมูลส่วนตัวได้

3.2.2 Non-functional Requirements

- แอปพลิเคชันสามารถตอบโต้และประมวลผลได้รวดเร็ว
- แอปพลิเคชันสามารถแสดงผลได้แบบเรียลไทม์
- แอปพลิเคชันมีอุปกรณ์พื้นฐานที่ใช้งานง่าย เช่น ช่องทางเข้าสู่ระบบ ภาษาไทย ฯลฯ

3.3 การออกแบบส่วนติดต่อผู้ใช้ (User Interface Design)

ในการออกแบบส่วนติดต่อผู้ใช้ของแอปพลิเคชันสูงวัยมายเฟรนด์ ประกอบด้วยส่วนต่าง ๆ ดังนี้

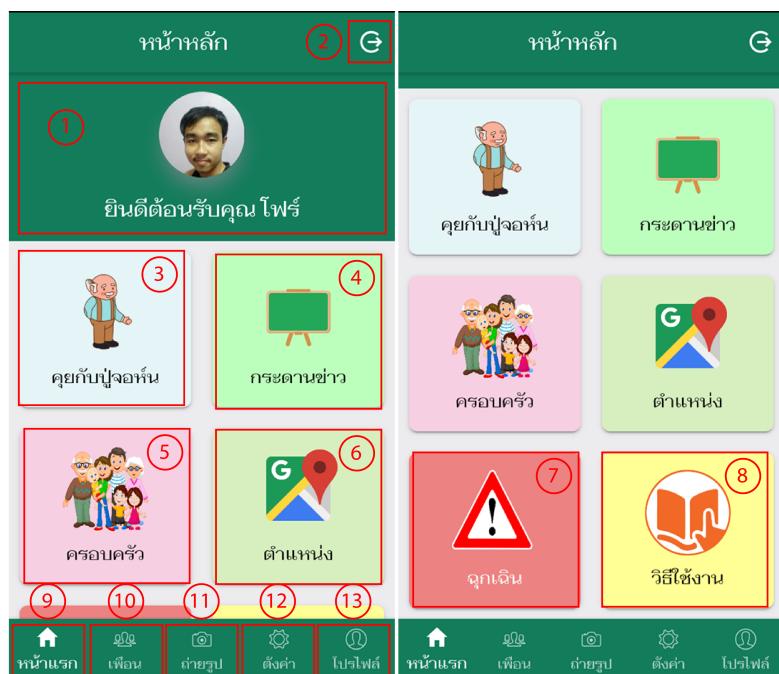
3.3.1 หน้าเข้าสู่ระบบ



รูปที่ 3.2: หน้าจอเข้าสู่ระบบ

จากภาพที่ 3.2 การออกแบบหน้าจอเข้าสู่ระบบประกอบไปด้วย 6 ส่วนดังนี้

- ส่วนที่ 1 โลโก้ของแอปพลิเคชัน
- ส่วนที่ 2 ช่องสำหรับกรอกชื่อผู้ใช้
- ส่วนที่ 3 ช่องสำหรับกรอกรหัสผ่าน
- ส่วนที่ 4 ปุ่มสำหรับเข้าสู่ระบบ
- ส่วนที่ 5 ปุ่มสำหรับสมัครสมาชิก
- ส่วนที่ 6 ข้อความสำหรับกดเมื่อลืมรหัสผ่าน

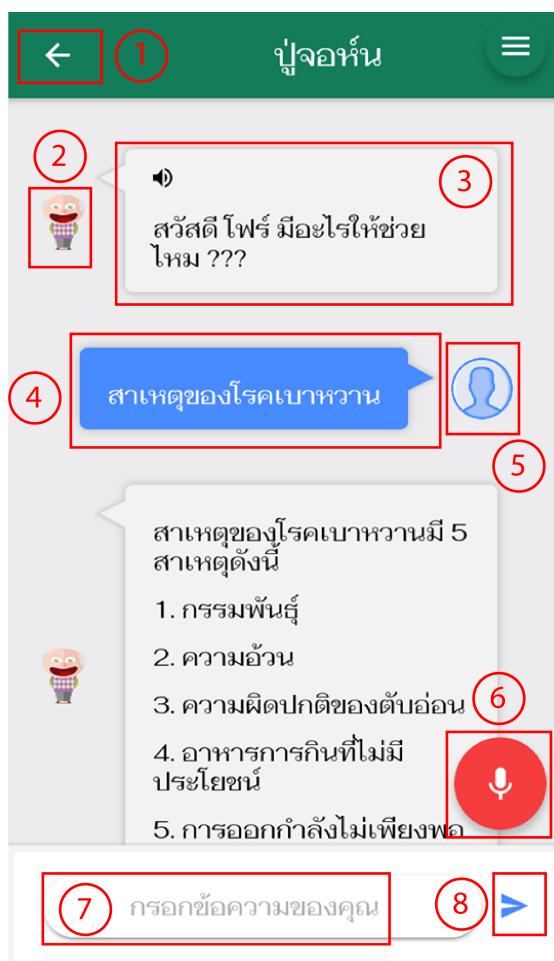


รูปที่ 3.3: หน้าจอหลัก

จากภาพที่ 3.3 การออกแบบหน้าจอหลักประกอบไปด้วย 13 ส่วนดังนี้

- ส่วนที่ 1 ข้อมูลของผู้ใช้ประกอบไปด้วยรูปประจำตัวและชื่อผู้ใช้
- ส่วนที่ 2 ปุ่มสำหรับออกจากระบบ
- ส่วนที่ 3 ปุ่มสำหรับไปยังหน้าปัจจุบันที่เป็นแท็บoth
- ส่วนที่ 4 ปุ่มสำหรับไปยังหน้ากระดานขาว
- ส่วนที่ 5 ปุ่มสำหรับไปยังหน้าครอบครัว
- ส่วนที่ 6 ปุ่มสำหรับไปยังหน้าตำแหน่งของครอบครัว

- ส่วนที่ 7 ปุ่มสำหรับไปยังหน้าฉุกเฉิน
- ส่วนที่ 8 ปุ่มไปยังหน้าวิธีใช้งาน
- ส่วนที่ 9 ปุ่มไปยังหน้าหลักเราจะอยู่หน้านี้ทุกครั้งเมื่อเราเข้าสู่แอปพลิเคชัน
- ส่วนที่ 10 ปุ่มสำหรับไปยังหน้าเพื่อนและครอบครัว
- ส่วนที่ 11 ปุ่มเพื่อโพสท์ข้อความหรือรูปภาพ
- ส่วนที่ 12 ปุ่มสำหรับไปหน้าตั้งค่า
- ส่วนที่ 13 ปุ่มสำหรับไปยังหน้าໂປຣໄຟລ໌หรือข้อมูลส่วนตัว

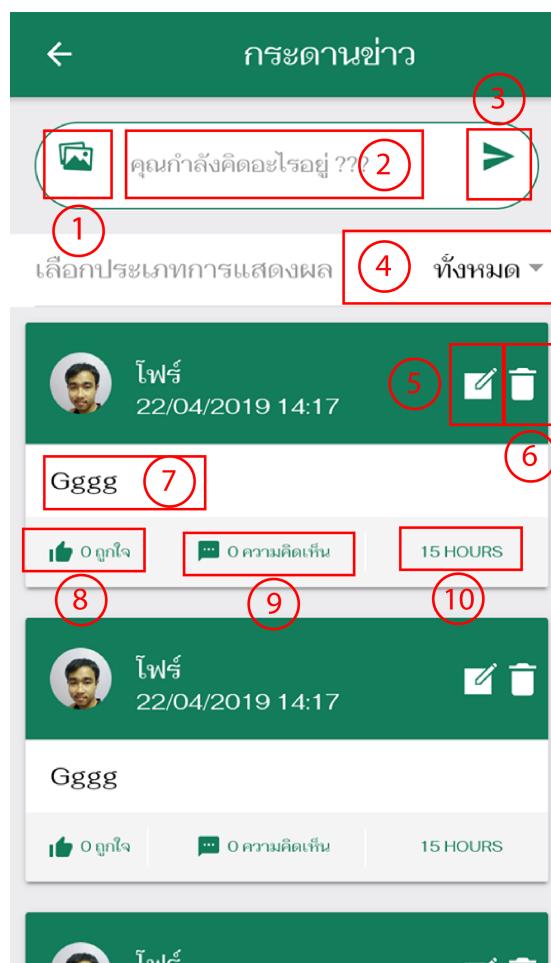


รูปที่ 3.4: หน้าเขตบทอุทคุยกับปู้จโจห์น

จากภาพที่ 3.4 การออกแบบหลักประกอบไปด้วย 8 ส่วนดังนี้

- ส่วนที่ 1 ปุ่มลูกศรกลับไปยังหน้าจอหลัก

- ส่วนที่ 2 รูปประจำตัวของปู่จอห์น
- ส่วนที่ 3 ข้อความที่ปู่จอห์นตอบกลับผู้ใช้
- ส่วนที่ 4 ข้อความของผู้ใช้
- ส่วนที่ 5 รูปประจำตัวของผู้ใช้
- ส่วนที่ 6 ปุ่มสำหรับพิมพ์ข้อความด้วยเสียง (Speech to text)
- ส่วนที่ 7 ช่องสำหรับกรอกข้อความที่จะแสดงในส่วนที่ 4
- ส่วนที่ 8 ปุ่มสำหรับยืนยันข้อความส่วนที่ 7



รูปที่ 3.5: หน้าจอกระดาษข่าว

จากภาพที่ 3.5 การออกแบบหลักประกอบไปด้วย 11 ส่วนดังนี้

- ส่วนที่ 1 ปุ่มสำหรับเพิ่มรูปภาพสำหรับโพสท์มีสองคือ ถ่ายรูปและเลือกรูปจากแกลอรี่

- ส่วนที่ 2 ช่องสำหรับกรอกข้อมูลความที่จะโพสต์
- ส่วนที่ 3 ปุ่มสำหรับยืนยันการโพสต์หลังจากคลิกจะมีการเลือกประเภทได้ 3 แบบคือ กีฬา, ศาสนา, ดนตรี
- ส่วนที่ 4 ลิสต์สำหรับเลือกแสดงเฉพาะประเภทที่ต้องการมี 4 แบบได้แก่ ทั้งหมด, กีฬา, ศาสนา, ดนตรี
- ส่วนที่ 5 ปุ่มสำหรับแก้ไขโพสต์
- ส่วนที่ 6 ปุ่มสำหรับลบโพสต์
- ส่วนที่ 7 ข้อความโพสต์ของผู้ใช้
- ส่วนที่ 8 ปุ่มสำหรับกดถูกใจ
- ส่วนที่ 9 ปุ่มสำหรับแสดงความคิดเห็น
- ส่วนที่ 10 ข้อความแสดงระยะเวลาหลังจากโพสต์ถูกสร้าง

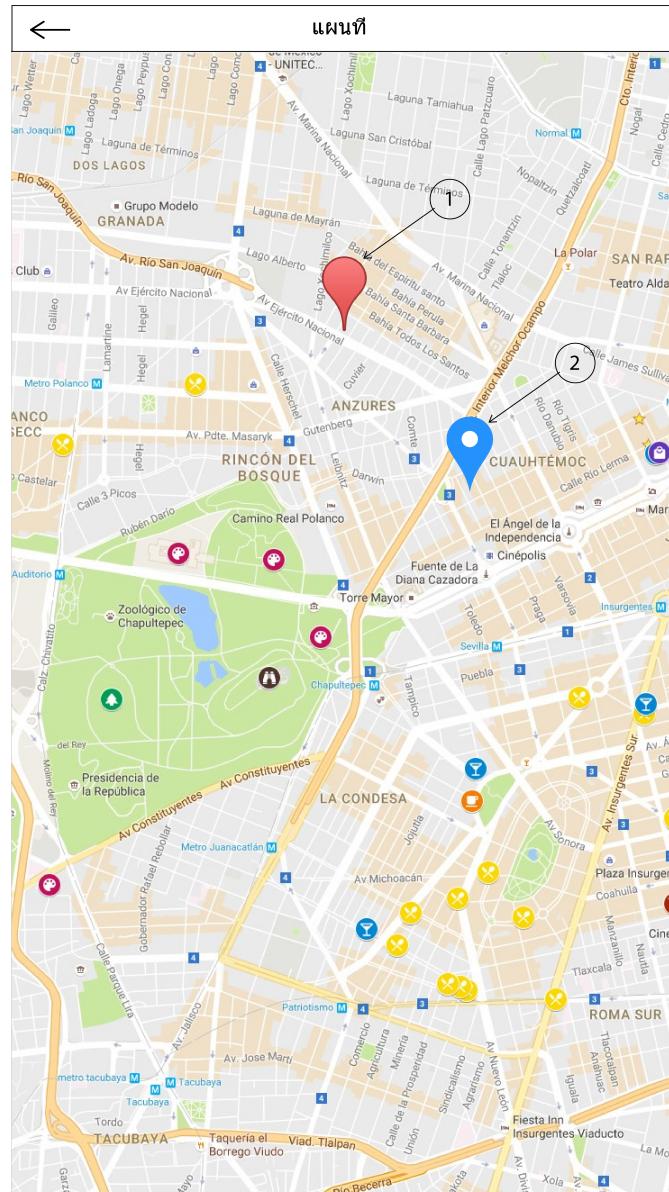


รูปที่ 3.6: หน้าครอบครัว

จากภาพที่ 3.6 การออกแบบหลักประกอบไปด้วย 5 ส่วนดังนี้

- ส่วนที่ 1 ปุ่มสำหรับเพิ่มสมาชิกในครอบครัว
- ส่วนที่ 2 รูปภาพของสมาชิกในครอบครัว
- ส่วนที่ 3 ชื่อของสมาชิกในครอบครัว

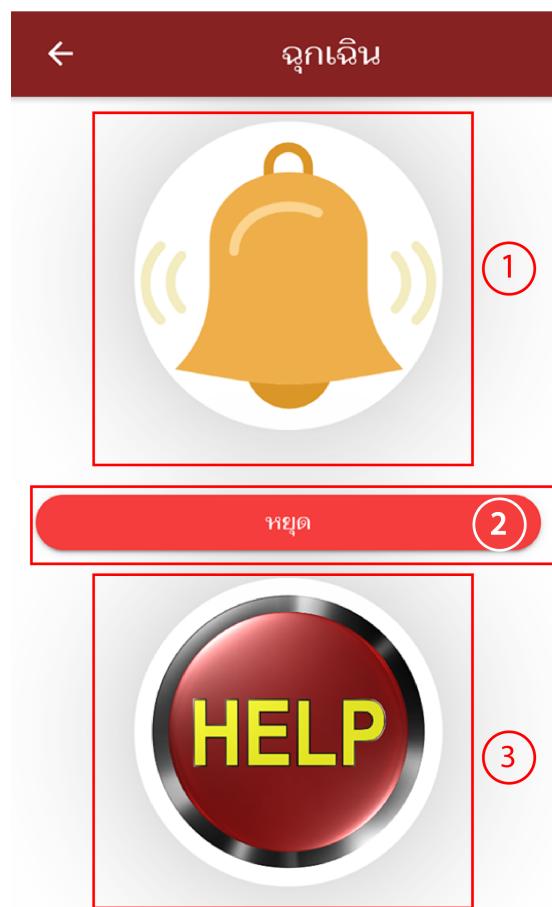
- ส่วนที่ 4 สถานะของครอบครัว
- ส่วนที่ 5 ปั๊มสำหรับจัดการสมาชิกครอบครัวได้แก่ แก้ไขสถานะ ลบสมาชิก



รูปที่ 3.7: หน้าแสดงตำแหน่งของครอบครัว

จากภาพที่ 3.7 การออกแบบหลักประกอบไปด้วย 2 ส่วนดังนี้

- ส่วนที่ 1 แสดงตำแหน่งของผู้ใช้
- ส่วนที่ 2 แสดงตำแหน่งของสมาชิกในครอบครัว



ຮູບທີ 3.8: ຜັນອຸກເນີນ

ຈາກກາພທີ 3.8 ກາຣອອກແບບໜັກປະກອບໄປດ້ວຍ 2 ສ່ວນດັ່ງນີ້

- ສ່ວນທີ 1 ປຸ່ມສໍາຫຼັບແສດງເສີຍເພື່ອຂອງຄວາມຊ່ວຍເຫຼືອ
- ສ່ວນທີ 2 ປຸ່ມສໍາຫຼັບປິດເສີຍເພື່ອຂອງຄວາມຊ່ວຍເຫຼືອ
- ສ່ວນທີ 2 ປຸ່ມສໍາຫຼັບໄປຢັງໜ້າເລືອກຮອບຄວ້າເພື່ອຂອງຄວາມຊ່ວຍເຫຼືອຈາກຮອບຄວ້າ



รูปที่ 3.9: หน้าเลือกครอบครัว

จากภาพที่ 3.9 การออกแบบหลักประกอบไปด้วย 5 ส่วนดังนี้

- ส่วนที่ 1 รูปประจำตัวของสมาชิกในครอบครัว
- ส่วนที่ 2 ชื่อของสมาชิกในครอบครัว
- ส่วนที่ 3 สถานะของครอบครัว
- ส่วนที่ 4 ปุ่มสำหรับโทรศัพท์ไปยังหมายเลขของสมาชิกในครอบครัว
- ส่วนที่ 5 ปุ่มสำหรับส่งข้อความไปยังสมาชิกในครอบครัว มีข้อความดังนี้ “ช่วยด้วย !!!” นี่ [ชื่อผู้ใช้] เอง ตอนนี้มีปัญหาช่วยติดต่อกลับมาที่ [เบอร์ผู้ใช้] ด้วยนะ ด่วนๆ”



รูปที่ 3.10: หน้าวิธีใช้งาน

จากภาพที่ 3.10 การออกแบบหลักประกอบไปด้วย 4 ส่วนดังนี้

- ส่วนที่ 1 รูปภาพตัวอย่างแสดงหน้าจอหนึ่งๆ
- ส่วนที่ 2 ชื่อของหน้าจอหนึ่งๆ
- ส่วนที่ 3 รายละเอียดเพิ่มเติมของหน้าจอหนึ่งๆ
- ส่วนที่ 4 จุดสำหรับแสดงตำแหน่งที่เรารอยู่ ณ ปัจจุบัน



รูปที่ 3.11: หน้าจอแสดงกลุ่ม

จากภาพที่ 3.11 การออกแบบหลักประกอบไปด้วย 7 ส่วนดังนี้

- ส่วนที่ 1 ปุ่มสำหรับเพิ่มเพื่อน
- ส่วนที่ 2 ปุ่มสำหรับไปยังหน้าแสดงกลุ่ม
- ส่วนที่ 3 ปุ่มสำหรับไปยังหน้าจอแสดงเพื่อน
- ส่วนที่ 4 รูปภาพของกลุ่มที่เราเป็นคนสร้าง
- ส่วนที่ 5 ชื่อของกลุ่มที่เราเป็นคนสร้าง
- ส่วนที่ 6 รูปภาพของกลุ่มที่เราเป็นสมาชิก
- ส่วนที่ 7 ชื่อของกลุ่มที่เราเป็นสมาชิก

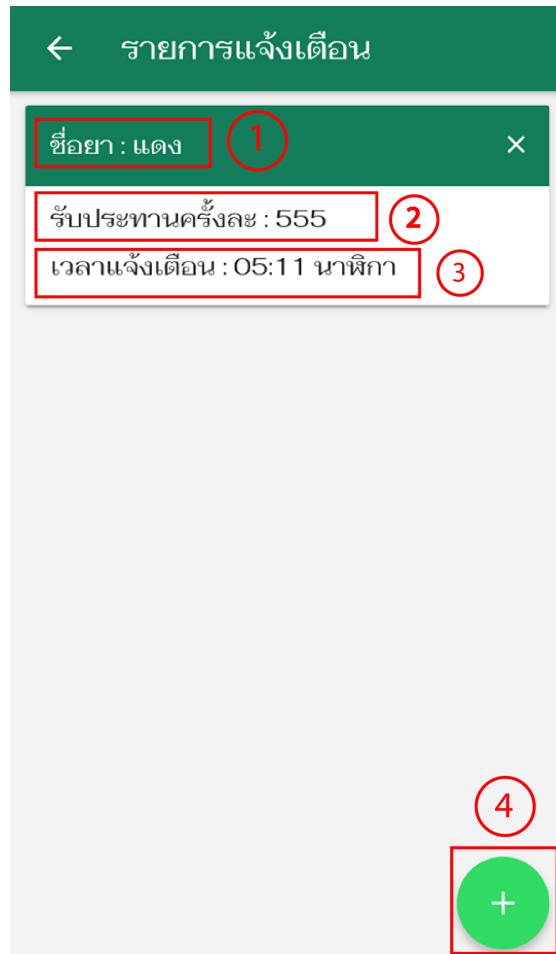


รูปที่ 3.12: หน้าจอแสดงเพื่อน

จากภาพที่ 3.12 การออกแบบหลักประกอบไปด้วย 8 ส่วนดังนี้

- ส่วนที่ 1 ปุ่มสำหรับเพิ่มเพื่อน
- ส่วนที่ 2 ปุ่มสำหรับไปยังหน้าแสดงกลุ่ม
- ส่วนที่ 3 ปุ่มสำหรับไปยังหน้าจอแสดงเพื่อน
- ส่วนที่ 4 รูปภาพของเพื่อนที่รอการยืนยัน
- ส่วนที่ 5 ชื่อของเพื่อนที่รอการยืนยัน
- ส่วนที่ 6 ปุ่มสำหรับยืนยันคำร้องขอเป็นเพื่อน
- ส่วนที่ 7 ปุ่มสำหรับยกเลิกคำขอร้องเป็นเพื่อน
- ส่วนที่ 8 รูปภาพของเพื่อน
- ส่วนที่ 9 ชื่อของเพื่อน

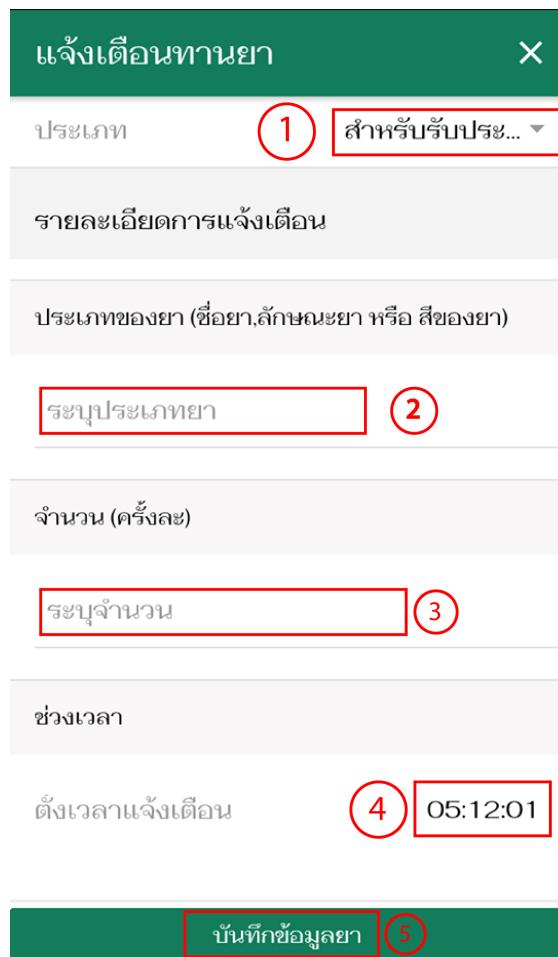
- ส่วนที่ 10 ปุ่มสำหรับแซทกับเพื่อน
- ส่วนที่ 11 ปุ่มสำหรับลบเพื่อน



รูปที่ 3.13: หน้าแสดงรายการแจ้งเตือนการทานยา

จากภาพที่ 3.13 การออกแบบหลักประกอบไปด้วย 4 ส่วนดังนี้

- ส่วนที่ 1 ข้อความแสดงชื่อยา
- ส่วนที่ 2 ข้อความแสดงจำนวนการรับประทานต่อครั้ง
- ส่วนที่ 3 ข้อความแสดงเวลาในการแจ้งเตือน
- ส่วนที่ 4 ปุ่มสำหรับเพิ่มการแจ้งเตือนการทานยา



รูปที่ 3.14: หน้าแสดงการเพิ่มการแจ้งเตือนท่านยา

จากภาพที่ 3.14 การออกแบบหลักประกอบไปด้วย 5 ส่วนดังนี้

- ส่วนที่ 1 ลิสท์สำหรับเลือกประเภทการรับประทานมี 2 ประเภทได้แก่ สำหรับรับประทาน และสำหรับฉีด
- ส่วนที่ 2 ช่องสำหรับกรอกชื่อยา ลักษณะของยา หรือสีของยา
- ส่วนที่ 3 ช่องสำหรับกรอกจำนวนครั้งในการทานยา
- ส่วนที่ 4 ลิสท์สำหรับกำหนดเวลาการแจ้งเตือนการทานยา
- ส่วนที่ 5 ปุ่มสำหรับยืนยันการเพิ่มการแจ้งเตือนการทานยา



รูปที่ 3.15: หน้าจอแสดงข้อมูลส่วนตัวของผู้ใช้

จากภาพที่ 3.15 การออกแบบหลักประกอบไปด้วย 4 ส่วนดังนี้

- ส่วนที่ 1 รูปประจำตัวของผู้ใช้
- ส่วนที่ 2 ปุ่มสำหรับเปลี่ยนรูปประจำตัว
- ส่วนที่ 3 แสดงจำนวนโพสต์ เพื่อน ครอบครัว และปุ่มแก้ไขข้อมูลส่วนตัว
- ส่วนที่ 4 โพสต์ของผู้ใช้

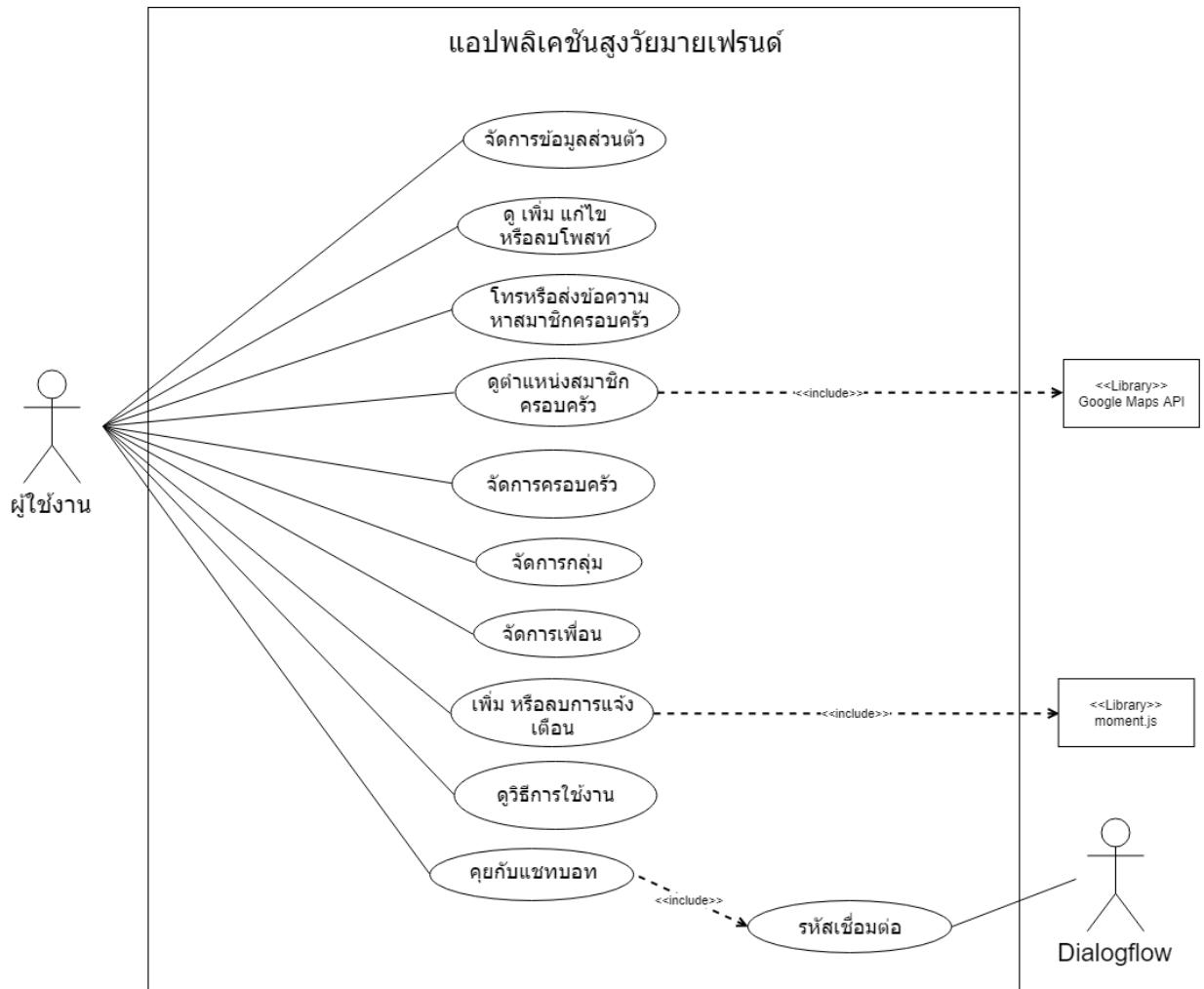
3.4 แผนภาพด้วยแกรม

3.4.1 ยูสเคสด้วยแกรม (Use Case Diagram)

Use Case Diagram เป็นแผนผังเพื่อแสดงฟังก์ชันแสดงการทำงานของระบบโดยรวม แสดงส่วนประกอบในระบบและกิจกรรมที่เกิดขึ้นในระบบซึ่งในระบบจะมีผู้ใช้จำเป็นต้องเข้าสู่ระบบเพื่อใช้งานระบบ สัญลักษณ์ที่ใช้ในการเขียน Use Case Diagram แสดงในตารางที่ 3.1

ตารางที่ 3.1: สัญลักษณ์ของ Use case Diagram

สัญลักษณ์	การใช้งาน
Use case	Use case คือส่วนย่อยของระบบงาน แทนด้วยวงรีและชื่อของ Use case ภายในวงรี
	Actor คือบุคคลหรือระบบงานอื่นที่ใช้งานระบบหรือได้รับประโยชน์จากการซึ่งอยู่ภายนอกระบบ แทนด้วยรูปคนและมีชื่อบาทการใช้งานระบบ
	เส้นตรงที่แสดงถึงการใช้งาน Use case ของผู้กระทำ
	กรอบ สี่เหลี่ยม แสดงถึงขอบเขตของระบบ โดยแสดงชื่อระบบ ภายในหรือด้านบนกรอบสี่เหลี่ยม Use case อยู่ภายในกรอบสี่เหลี่ยม และ actor อยู่ภายนอกกรอบสี่เหลี่ยม
	ความสัมพันธ์แบบ «includes» แสดงว่า Use case หนึ่งดำเนินการตามขั้นตอนของ Use case อื่น โดยแทนด้วยสัญลักษณ์ลูกศรเส้นประซึ่ง Use case ที่ทางลูกศรเรียกใช้งาน Use case ที่หัวลูกศรทุกครั้งที่มีการทำงาน
	ความสัมพันธ์แบบ «extend» แสดงว่า Use case หนึ่งดำเนินการตามขั้นตอนของ Use case อื่น โดยแทนด้วยสัญลักษณ์ลูกศรเส้นประซึ่ง Use case ที่หัวลูกศรเรียกใช้งาน Use case ที่ทางลูกศรแต่การใช้งานไม่จำเป็นต้องเกิดขึ้นทุกครั้งขึ้นอยู่กับเงื่อนไขระหว่างการทำงาน



รูปที่ 3.16: Use Case Diagram ของแอปพลิเคชันสูงวัยมายเฟรนด์

3.4.2 ตารางแสดงคำอธิบายของผู้ใช้ (User Case Description)

ตารางที่ 3.2: อธิบาย Class Diagram ของคลาสพื้นฐานของระบบ

สัญลักษณ์	การใช้งาน
เข้าสู่ระบบ	ใช้สำหรับให้ผู้ใช้เข้าใช้งานแอปพลิเคชัน
ออกจากระบบ	ใช้งานเพื่อให้ผู้ใช้ แก้ไขข้อมูลที่ต้องการเปลี่ยนแปลง
จัดการข้อมูลส่วนตัว	ใช้งานเพื่อให้ผู้ใช้ แก้ไขข้อมูลที่ต้องการเปลี่ยนแปลง
ดู เพิ่ม แก้ไข หรือลบโพสท์	ใช้งานเพื่อให้ผู้ใช้สามารถดู เพิ่ม แก้ไข หรือลบโพสท์ของตัวเอง
โถรหรือส่งข้อความหาครอบครัว	ใช้สำหรับให้ผู้ใช้สามารถแจ้งเตือนไปยังครอบครัวได้
ดูตำแหน่งสมาชิกในครอบครัว	ใช้งานเพื่อให้ผู้ใช้สามารถดูตำแหน่งปัจจุบันของสมาชิกในครอบครัวด้วยแผนที่
จัดการครอบครัว	ใช้สำหรับเพิ่ม แก้ไข หรือลบข้อมูลสมาชิกครอบครัว
จัดการกลุ่ม	ใช้สำหรับเพิ่ม แซท แก้ไข หรือลบกลุ่มได้ และเพิ่ม แก้ไข หรือลบสมาชิกในกลุ่มได้
จัดการเพื่อน	ใช้สำหรับยืนยันคำร้องขอเป็นเพื่อน รวมถึงพูดคุย เพิ่มและลบเพื่อน
เพิ่ม หรือลบการแจ้งเตือน	ใช้สำหรับเพิ่ม หรือลบการแจ้งเตือนการทำงานยา
ดูวิธีการใช้งาน	ใช้สำหรับให้ผู้ใช้ดูรายละเอียดของหน้าต่าง ๆ ในแอปพลิเคชัน
คุยกับเซทบอท	ใช้งานเพื่อให้ผู้ใช้สามารถถามเรื่องโรคหรือเรื่องที่สนใจ

ตารางที่ 3.3: Use Case เข้าสู่ระบบ

Use Case Title : เข้าสู่ระบบ	Use case Id : 1
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถเข้าสู่ระบบเพื่อใช้งานแอปพลิเคชัน	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถเข้าสู่ระบบได้	
Pre-condition : -	

ตารางที่ 3.4: Use Case ออกจากระบบ

Use Case Title : ออกจากระบบ	Use case Id : 2
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถออกจากระบบได้	
Exceptional Flow ที่ 1 : หากยังไม่เข้าสู่ระบบจะไม่สามารถออกจากระบบได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.5: Use Case จัดการข้อมูลส่วนตัว

Use Case Title : จัดการข้อมูลส่วนตัว	Use case Id : 3
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถดู เพิ่ม แก้ไขข้อมูลส่วนตัวได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถดู เพิ่ม แก้ไขข้อมูลส่วนตัวได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.6: Use Case ดูเพิ่ม แก้ไข หรือลบโพสท์

Use Case Title : ดูเพิ่ม แก้ไข หรือลบโพสท์	Use case Id : 4
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถดู เพิ่ม แก้ไข หรือลบโพสท์ได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถดู เพิ่ม แก้ไข หรือลบโพสท์	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.7: Use Case โทรหรือส่งข้อความหาสมาชิกครอบครัว

Use Case Title : โทรหรือส่งข้อความหาสมาชิกครอบครัว	Use case Id : 5
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถโทรหรือส่งข้อความหาสมาชิกครอบครัว และสามารถเปิดเสียงฉุกเฉินได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถโทรหรือส่งข้อความหาสมาชิกครอบครัว และสามารถเปิดเสียงฉุกเฉินได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.8: Use Case ดูตำแหน่งสมาชิกครอบครัว

Use Case Title : ดูตำแหน่งสมาชิกครอบครัว	Use case Id : 6
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถดูตำแหน่งสมาชิกในครอบครัวได้ด้วยแผนที่	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถดูตำแหน่งสมาชิกในครอบครัวได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.9: Use Case จัดการครอบครัว

Use Case Title : จัดการครอบครัว	Use case Id : 7
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถเพิ่ม แก้ไข ลบสมาชิกในครอบครัวได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถเพิ่ม แก้ไข ลบสมาชิกในครอบครัวได้	
Exceptional Flow ที่ 2 : หากผู้ใช้ไม่มีผู้ใช้งานเป็นเพื่อน จะไม่สามารถเพิ่มเข้ามาในครอบครัวได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.10: Use Case จัดการกลุ่ม

Use Case Title : จัดการกลุ่ม	Use case Id : 8
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถเพิ่ม พูดคุย แก้ไข หรือลบกลุ่ม เพิ่ม แก้ไข หรือลบสมาชิกในกลุ่มได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถเพิ่ม พูดคุย แก้ไข หรือลบกลุ่ม เพิ่ม แก้ไข หรือลบสมาชิกในกลุ่มได้	
Exceptional Flow ที่ 2 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ตจะไม่สามารถเพิ่ม พูดคุย แก้ไข หรือลบกลุ่ม เพิ่ม แก้ไข หรือลบสมาชิกในกลุ่มได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.11: Use Case จัดการเพื่อน

Use Case Title : จัดการเพื่อน	Use case Id : 9
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถเพิ่ม พูดคุย แก้ไข หรือลบเพื่อนได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถเพิ่ม พูดคุย แก้ไข หรือลบเพื่อนได้	
Exceptional Flow ที่ 2 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ตจะไม่สามารถเพิ่ม พูดคุย แก้ไข หรือลบเพื่อนได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.12: Use Case เพิ่ม หรือลบการแจ้งเตือน

Use Case Title : เพิ่ม หรือลบการแจ้งเตือน	Use case Id : 10
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถเพิ่ม หรือลบการแจ้งเตือนการทานยาได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถเพิ่ม หรือลบการแจ้งเตือนได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.13: Use Case ดูวิธีการใช้งาน

Use Case Title : ดูวิธีการใช้งาน	Use case Id : 11
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถดูรายละเอียดของหน้าจอต่าง ๆ ได้	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เข้าสู่ระบบจะไม่สามารถดูรายละเอียดของหน้าจอต่าง ๆ ได้	
Pre-condition : เข้าสู่ระบบ	

ตารางที่ 3.14: Use Case คุยกับเซทบอท

Use Case Title : คุยกับเซทบอท	Use case Id : 12
Primary Actor : ผู้ใช้งาน	
Stakeholder Actor : -	
Main Flow : สามารถคุยกับเซทบอทได้ โดยเซทบอทจะต้องรับรหัสการใช้งานจาก Dialogflow สำหรับติดต่อใช้งาน	
Exceptional Flow ที่ 1 : หากผู้ใช้งานไม่เชื่อมต่ออินเทอร์เน็ตจะไม่คุยกับเซทบอทได้	
Exceptional Flow ที่ 2 : หากผู้ใช้งานไม่เข้าสู่ระบบจะไม่คุยกับเซทบอทได้	
Pre-condition : เข้าสู่ระบบ	

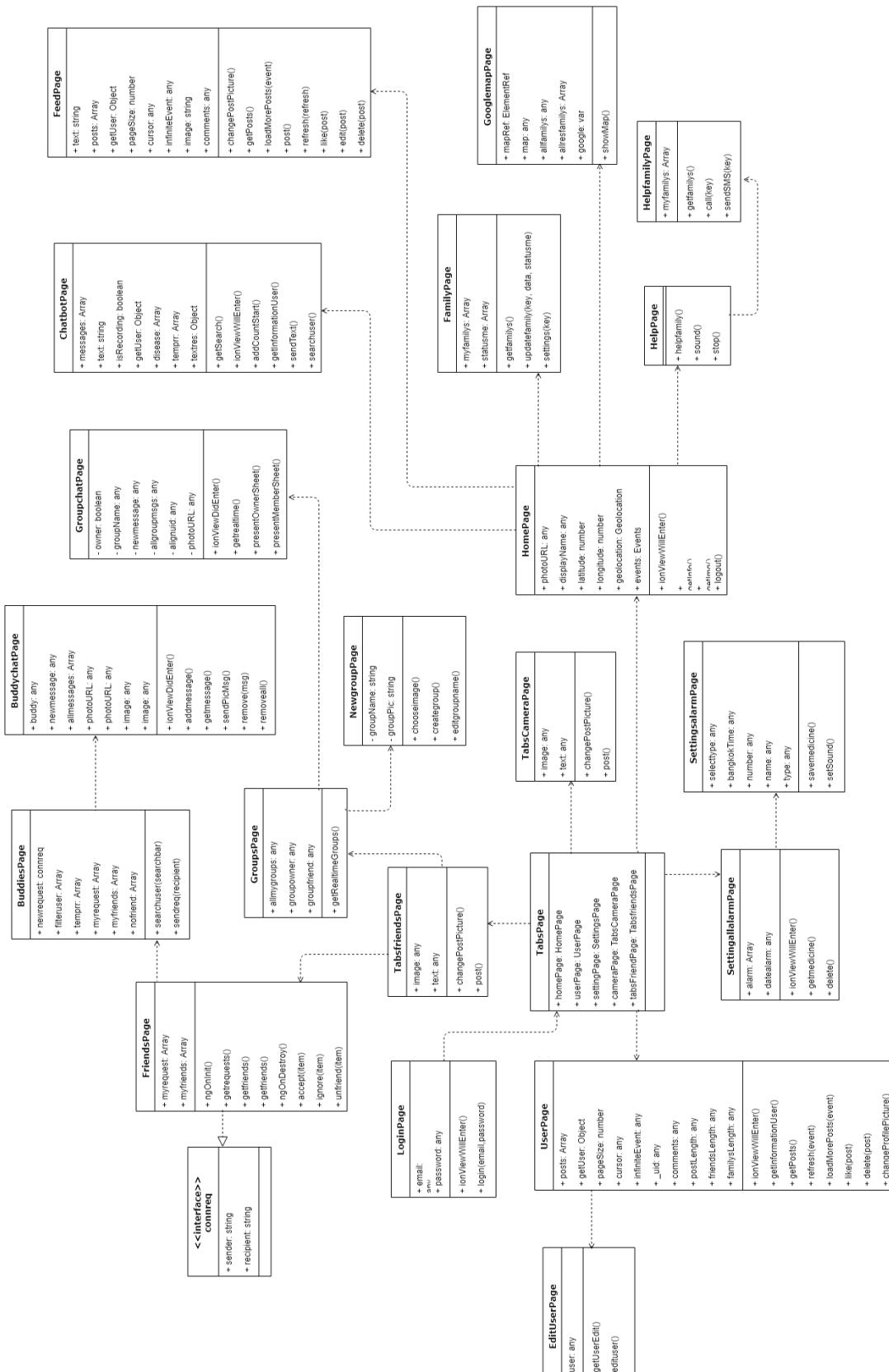
3.4.3 คลาสไดอะแกรม (Class Diagram)

Class Diagram คือแผนภาพที่ใช้แสดงคลาสและความสัมพันธ์ในแบบต่างๆ ระหว่างคลาส สัญลักษณ์ที่ใช้ในการเขียน Class Diagram แสดงในตารางที่ 3.15

ตารางที่ 3.15: สัญลักษณ์ของ Class Diagram

สัญลักษณ์	การใช้งาน
<p>Class Name Attribute Name Operation Name()</p>	<p>คลาส สัญลักษณ์แทนด้วยสี่เหลี่ยมแบ่งเป็น 3 ส่วน ส่วนบน เป็นชื่อของ class ส่วนกลาง เป็นชื่อ Attribute และ ส่วนล่าง เป็น Operation Name หรือ Method ใช้สำหรับเขียนฟังก์ชันในการทำงานของคลาสนั้น ๆ ชนิดของ Visibility ของ Method และ Attribute แบ่งเป็น 3 ชนิด ได้แก่</p> <ol style="list-style-type: none"> 1. Public แทนสัญลักษณ์ด้วยเครื่องหมายบวก (+) 2. Private แทนสัญลักษณ์ด้วยเครื่องหมายลบ (-) 3. Protected แทนสัญลักษณ์ด้วยเครื่องหมายชาร์ป ()
	Dependency Relationship หมายความว่า คลาสที่อยู่ผู้ให้ต้นลูกศร สามารถเรียกใช้คลาสที่อยู่ผู้รับหัวลูกศร
	Composition Relationship เป็นความสัมพันธ์ระหว่างออบเจกต์ หรือคลาสแบบขึ้นต่อกันและมีความเกี่ยวข้องกันเสมอ
	Realization Relationship เป็นความสัมพันธ์ระหว่าง Object หรือ Class ในลักษณะของการสืบทอดคุณสมบัติจาก Class หนึ่ง (Super class) ไปยังอีก Class หนึ่ง (Subclass)
	Connector เป็นสัญลักษณ์แทนด้วยรูปห้าเหลี่ยมและมีชื่ออู่ต่องกลาง จะสร้างสัญลักษณ์นี้ไว้เมื่อต้องการเชื่อมต่อคลาสที่อยู่คนละหน้า

Class Diagram แสดงความสัมพันธ์ในรูปแบบต่างๆ ระหว่างคลาสของแอปพลิเคชันสูงวัย
นายเฟรนด์ อธิบายได้ตามภาพที่ 3.17 ดังต่อไปนี้



ຮູບພາບ 3.17: Class Diagram ທີ່ຈະນອບປະລິດຕັ້ງສັງລຸມມາຍເພື່ອນດໍາລັດ

จากรูปภาพที่ 3.17 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.16: อธิบาย Class Diagram ของแอปพลิเคชันสูงวัยมายเพรนด์

Class Diagram	คำอธิบาย
LoginPage	คลาส LoginPage เป็นคลาสที่ใช้เพื่อให้ผู้ใช้ที่ได้ลงทะเบียนกับระบบไว้แล้วเข้าระบบเพื่อใช้งานแอปพลิเคชัน
TabsPage	คลาส TabsPage เป็นคลาสที่ถูกเรียกเพื่อจัดการ Tabs
TabsfriendsPage	คลาส TabsfriendsPage เป็นคลาสที่ใช้แสดงสมาชิกกลุ่ม คำร้องขอเพื่อนเพื่อนและเพื่อน
FriendsPage	คลาส FriendsPage เป็นคลาสที่ใช้แสดงคำร้องขอเพื่อนและเพื่อน
connreq	คลาส connreq เป็นคลาสที่ใช้เก็บข้อมูลคำร้องขอเพื่อน
BuddiesPage	คลาส BuddiesPage เป็นคลาสที่ใช้สำหรับการเพิ่มเพื่อน
BuddychatPage	คลาส BuddychatPage เป็นคลาสที่ใช้สำหรับแชทกับเพื่อน
BuddychatPage	คลาส BuddychatPage เป็นคลาสที่ใช้สำหรับแชทกับเพื่อน
GroupsPage	คลาส GroupsPage เป็นคลาสที่แสดงกลุ่มของฉันและกลุ่มที่เป็นสมาชิก
NewgroupPage	คลาส NewgroupPage เป็นคลาสที่ใช้สร้างกลุ่ม
GroupchatPage	คลาส GroupchatPage เป็นคลาสสำหรับการแชทกลุ่ม
SettingallalarmPage	คลาส SettingallalarmPage เป็นคลาสที่แสดงการแจ้งเตือนการทانยาทั้งหมด
SettingsalarmPage	คลาส SettingsalarmPage เป็นคลาสที่ใช้บันทึกการแจ้งเตือนการทันยา

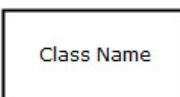
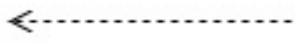
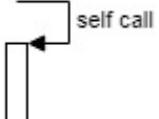
ตารางที่ 3.17: อธิบาย Class Diagram ของแอปพลิเคชันสูงวัยมายเฟรนด์

Class Diagram	คำอธิบาย
TabsCameraPage	คลาส TabsCameraPage เป็นคลาสที่ใช้สำหรับสร้างโพสท์
HomePage	คลาส HomePage เป็นคลาสสำหรับการแสดงหน้าเมนูหลักของแอปพลิเคชัน
ChatbotPage	คลาส ChatbotPage เป็นคลาสที่ใช้สำหรับพูดคุยกับเซฟบอท
FeedPage	คลาส FeedPage เป็นคลาสที่แสดงโพสท์ทั้งหมด
FamilyPage	คลาส FamilyPage เป็นคลาสที่แสดงสมาชิกในครอบครัว
GooglemapPage	คลาส GooglemapPage เป็นคลาสที่แสดงตำแหน่งของสมาชิกในครอบครัวด้วย Google Map
HelpPage	คลาส HelpPage เป็นคลาสที่แสดงรูปภาพขอความช่วยเหลือ 2 แบบ คือ แบบติดต่อครอบครัว และแบบส่งเสียงฉุกเฉิน
HelpfamilyPage	คลาส HelpfamilyPage เป็นคลาสแสดงสมาชิกในครอบครัวเพื่อขอความช่วยเหลือมี 2 แบบได้แก่ โทร และส่งข้อความ

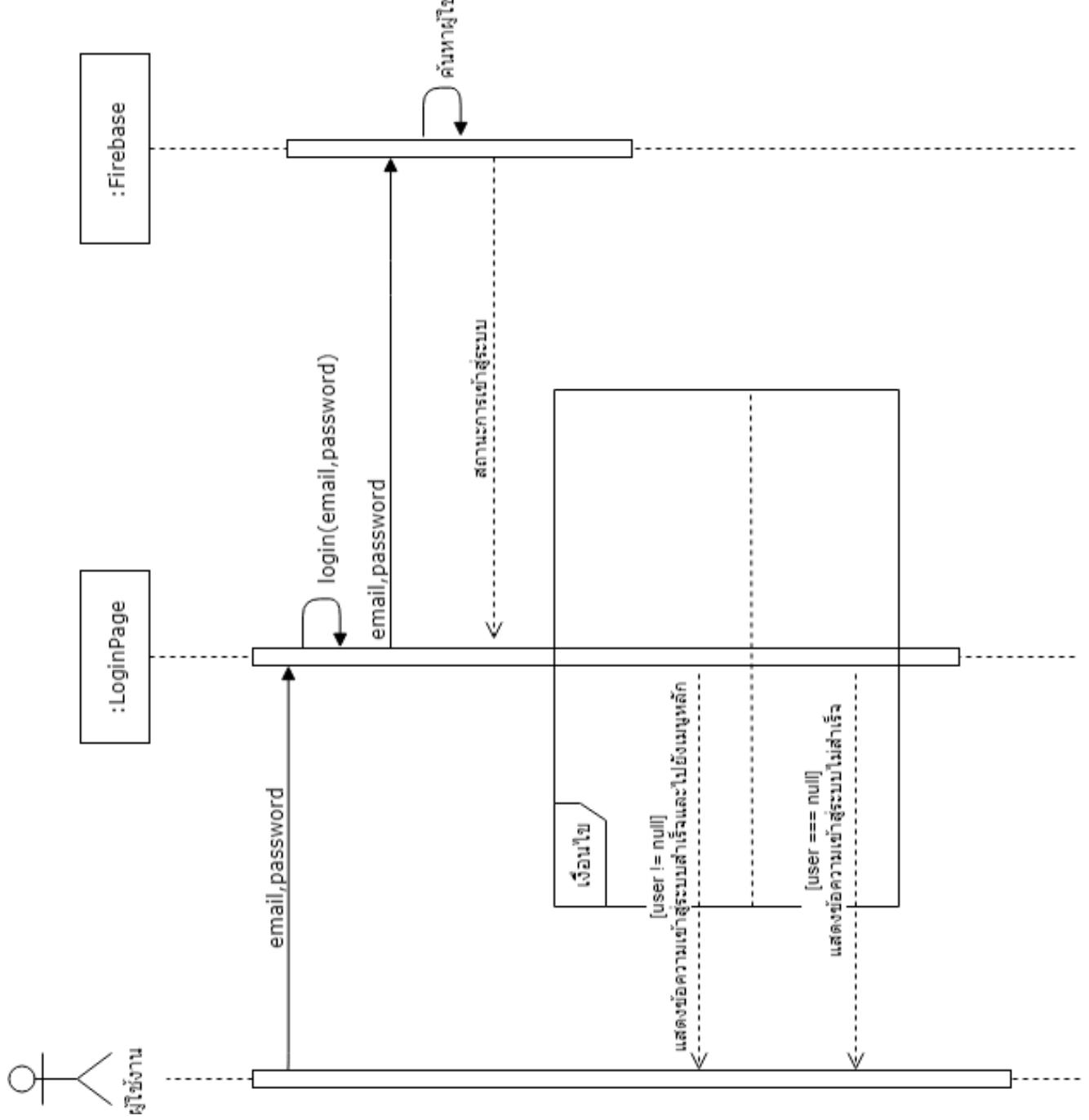
3.4.4 ชีเควนไดอะแกรม (Sequence Diagram)

Sequence Diagram เป็น Diagram ที่แสดงขั้นตอนการทำงานของแต่ละ Use Case ระหว่าง Object ต่างๆ ที่ส่งข้อความลึกลับและกัน โดย Sequence Diagram จะช่วยให้มองเห็นการทำงานของภาพรวมของระบบ ส่วนประกอบสัญลักษณ์ที่ใช้ในการเขียน Sequence Diagram แสดงดังตารางที่ 3.18

ตารางที่ 3.18: สัญลักษณ์ของ Sequence Diagram

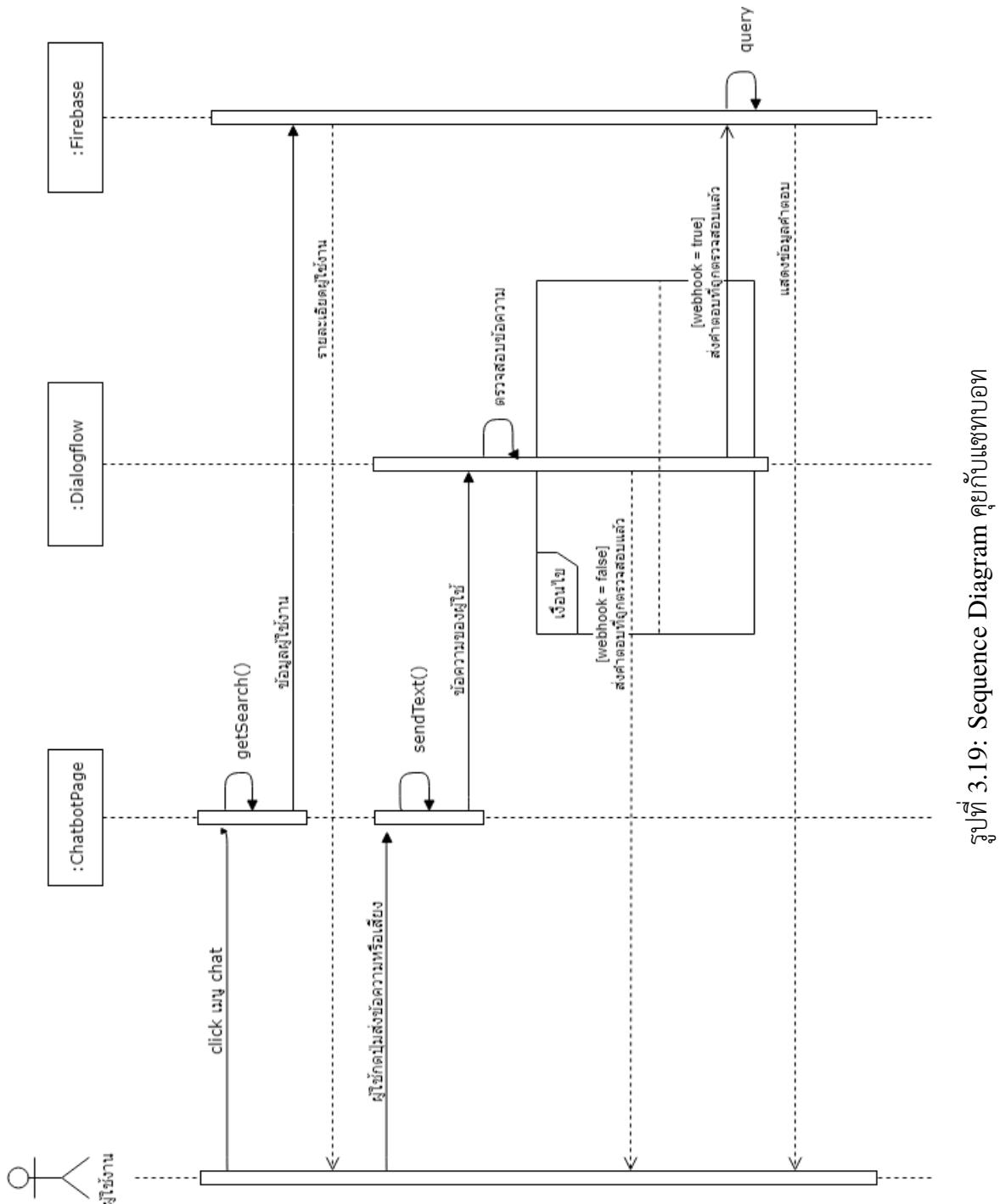
สัญลักษณ์	การใช้งาน
	Class แสดงถึงการทำงานของ Use Case ใน การส่งหรือรับข้อความแทนด้วยสัญลักษณ์สี่เหลี่ยมมีชื่อคลาสอยู่ภายใน
	Lifeline หรือเส้นอายุขัย แสดงช่วงเวลาตั้งแต่เริ่มสร้าง object ในคลาสนั้น จนกระทั่ง object นั้นถูกทำลาย สัญลักษณ์แทนด้วยเส้นประ
	Focus of control หรือจุดควบคุม เป็นจุดควบคุมที่ object ใช้ทำการส่งหรือรับข้อความ สัญลักษณ์แทนด้วยสี่เหลี่ยม
	Message คือ ข้อความที่รับส่งระหว่าง Object สัญลักษณ์แทนด้วยลูกศรและประกอบด้วย 2 ส่วน คือ ข้อมูล (Data) และฟังก์ชัน (Function)
	Return Message เป็นข้อมูลที่ส่งกลับหลังจากทำงานเสร็จ
	Self call เป็นการเรียกฟังก์กการทำงานภายในตัวเอง
	สร้างกรอบการทำงานของโปรแกรม เพื่อให้รู้ขอบเขตของการทำงาน เช่น ลูป(loop)

Sequence Diagram ที่ใช้อธิบายการทำงานของแอปพลิเคชันสูงวัยมายเฟรนด์ มีรายละเอียดดังต่อไปนี้



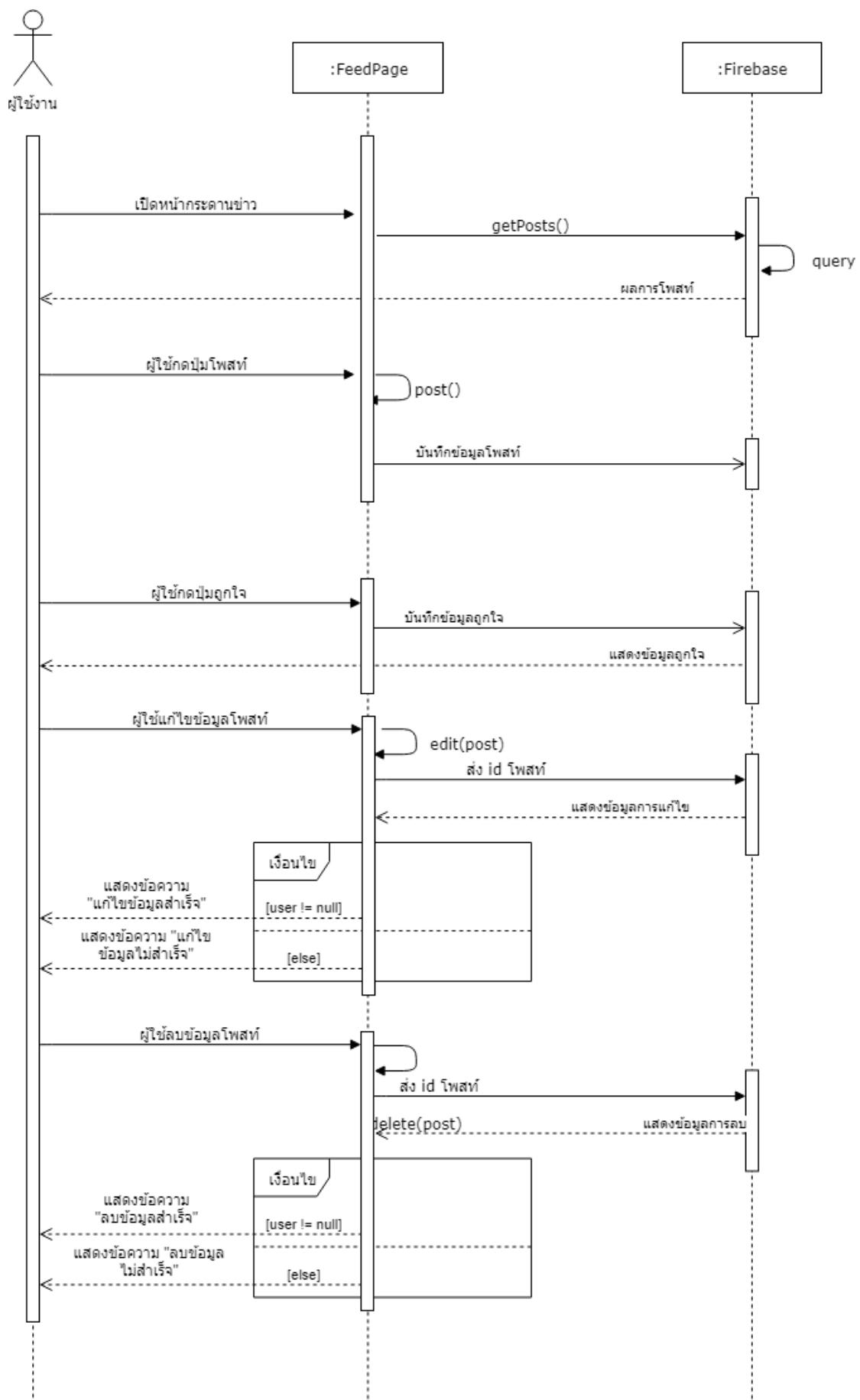
รูปที่ 3.18: Sequence Diagram การใช้เฟิร์บเบส

จากรูปภาพที่ 3.18 สามารถอธิบายแผนภาพ Sequence Diagram การเข้าสู่ระบบ ได้ดังนี้ เมื่อผู้ใช้กรอก e-mail และ password และกดปุ่มเข้าสู่ระบบ จากนั้นระบบจะเรียกฟังก์ชัน login(email,password) เพื่อส่งข้อมูลไปตรวจที่ Firebase เพื่อตรวจสอบ ว่า email และ password ถูกต้องหรือไม่ ถ้าไม่ถูกต้องจะส่งค่ากลับคืนมาแล้วระบบจะแสดงข้อความว่ากรุณากรอกข้อมูลให้ถูกต้อง ถ้าถูกต้องระบบจะแสดงข้อความว่าท่านได้เข้าสู่ระบบสำเร็จแล้ว



รูปที่ 3.19: Sequence Diagram ดูปแบบของ

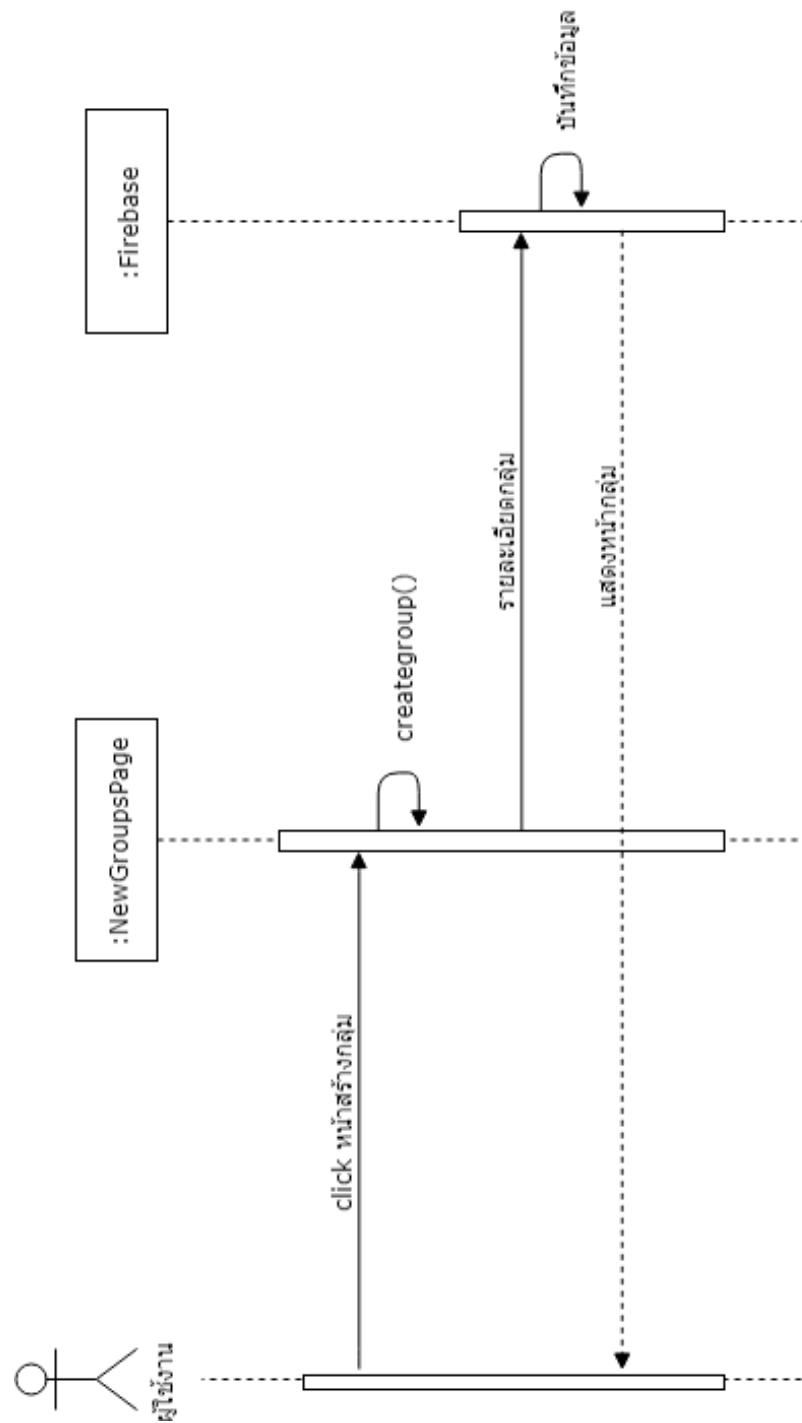
จากภาพที่ 3.19 สามารถอธิบายแผนภาพ Sequence Diagram คุยกับเซทบอท ได้ดังนี้ เมื่อผู้ใช้เปิดใช้งานหน้าปูสันทนา กับเซทบอท ระบบจะทำการบันทึกข้อมูลการค้นหาข้อมูลเป็น 0 ในไฟล์เบส เพื่อใช้สำหรับเช็คการค้นหาที่ถูกค้นหามากที่สุด และไฟล์เบสก็จะส่งข้อมูลกลับคืนมาหากผู้ใช้ ถ้าผู้ใช้ส่งในรูปแบบข้อความหรือเสียงถ้าหากเป็นเสียงจะแปลงเป็นข้อความก่อน หลังจากนั้นข้อความจะถูกส่งไป Dialogflow เพื่อประมวลผลใน Intent เพื่อหาคำตอบถ้าหาก Intent นั้นไม่ได้ใช้งาน webhook Dialogflow จะส่งข้อความกลับคืนหาผู้ใช้ตามคำตอบที่ได้ตั้งค่าไว้ ถ้าหาก Intent นั้น ได้เปิด webhook จะส่งข้อความไปตรวจสอบใน Fullfillment เพื่อตรวจสอบคำสำคัญในข้อความ หลังจากนั้นจะถูกส่งไปเช็คใน Firebase และส่ง คำตอบกลับมายังผู้ใช้



รูปที่ 3.20: Sequence Diagram การแสดงหน้ากระดาษข่าว

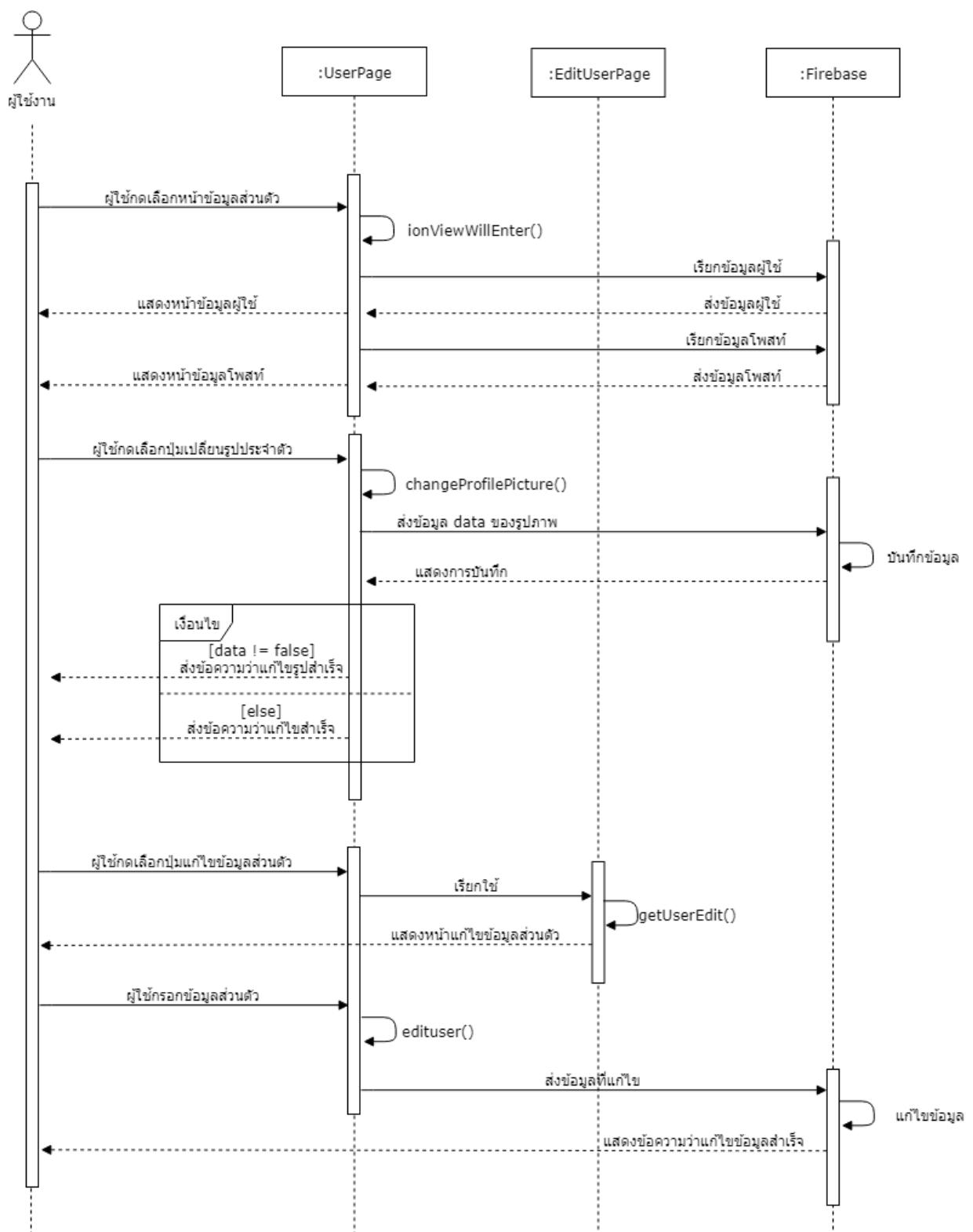
จากภาพที่ 3.20 สามารถอธิบายแผนภาพ Sequence Diagram การแสดงหน้ากระดานข่าวได้ดังนี้

- เมื่อผู้ใช้งานเข้ามายังหน้ากระดานข่าว ระบบจะเรียกฟังก์ชัน `getPost()` จะดึงข้อมูลมาจากการ์บสเพื่อแสดงโพสท์ทั้งหมด เมื่อผู้ใช้กดปุ่มโพสท์ระบบจะเรียกฟังก์ชัน `post()` ระบบจะตรวจสอบว่าข้อมูลถูกต้องหรือไม่ จากนั้นระบบจะบันทึกโพสท์ที่เราสร้างไว้ในฐานข้อมูล
- เมื่อผู้ใช้กดปุ่มถูกใจ ระบบจะส่งข้อมูลไปดีของผู้ที่กดส่งไปเก็บไว้ยังฐานข้อมูล Firebase และจึงส่ง Response กลับมาหาผู้ใช้
- เมื่อผู้ใช้กดเลือกเมนูแก้ไขโพสท์แล้วกดปุ่มบันทึกข้อมูล ระบบจะทำการเรียกใช้ฟังก์ชัน `edit()` เพื่อส่ง id ไปเช็คข้อมูลใน Firebase และจึงแก้ไขข้อมูล หากมี Response กลับคืนมาจะแสดงข้อความว่าบันทึกข้อมูลไม่สำเร็จ
- เมื่อผู้ใช้กดเลือกเมนูลบโพสท์แล้วกดปุ่มยืนยัน ระบบจะทำการเรียกใช้ฟังก์ชัน `delete()` เพื่อส่ง id ไปเช็คข้อมูลใน Firebase และจึงลบข้อมูล หากมี Response กลับคืนมาจะแสดงลบข้อมูลสำเร็จ หากไม่มี Response กลับคืนมาจะแสดงข้อความว่าลบข้อมูลไม่สำเร็จ



รูปที่ 3.21: Sequence Diagram ของ การเพิ่มกลุ่ม

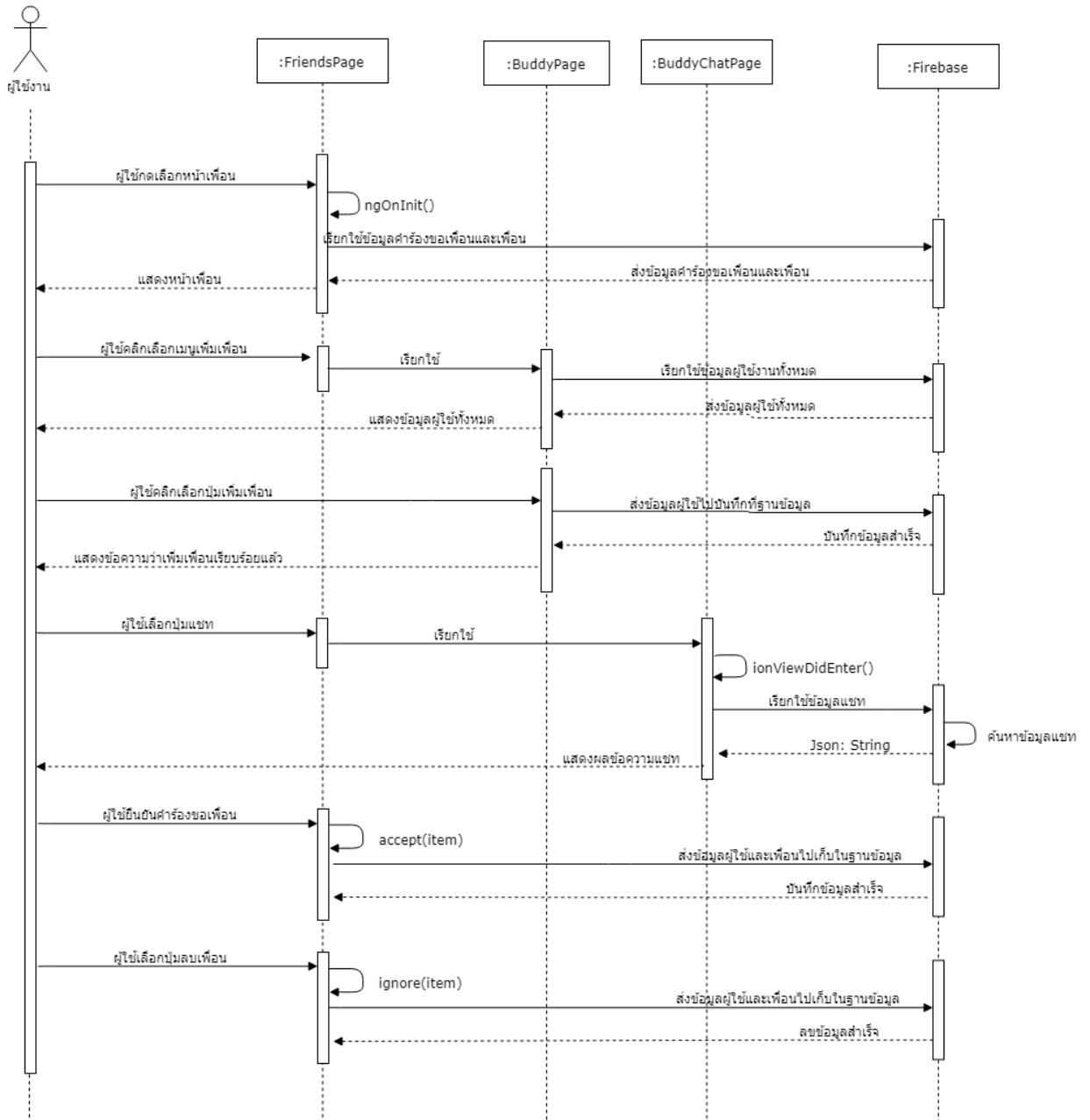
จากภาพที่ 3.21 สามารถอธิบายแผนภาพ Sequence Diagram ของการเพิ่มกลุ่ม ได้ดังนี้ เมื่อผู้ใช้ กดปุ่มเพิ่มกลุ่มระบบ และผู้ใช้เลือกรูปและชื่อกลุ่มเรียบร้อยแล้ว และกดปุ่มสร้างกลุ่ม จากนั้นระบบจะเรียกฟังก์ชัน `creategroup()` ในคลาส `NewGroupsPage` เพื่อบันทึกข้อมูลลงใน Firebase เมื่อเพิ่มฐานข้อมูลเสร็จเรียบร้อยแล้ว ระบบจะแสดงหน้ากลุ่ม



รูปที่ 3.22: Sequence Diagram ของการจัดการข้อมูลส่วนตัว

จากภาพที่ 3.22 สามารถอธิบายแผนภาพ Sequence Diagram ของการจัดการข้อมูลส่วนตัวได้ดังนี้

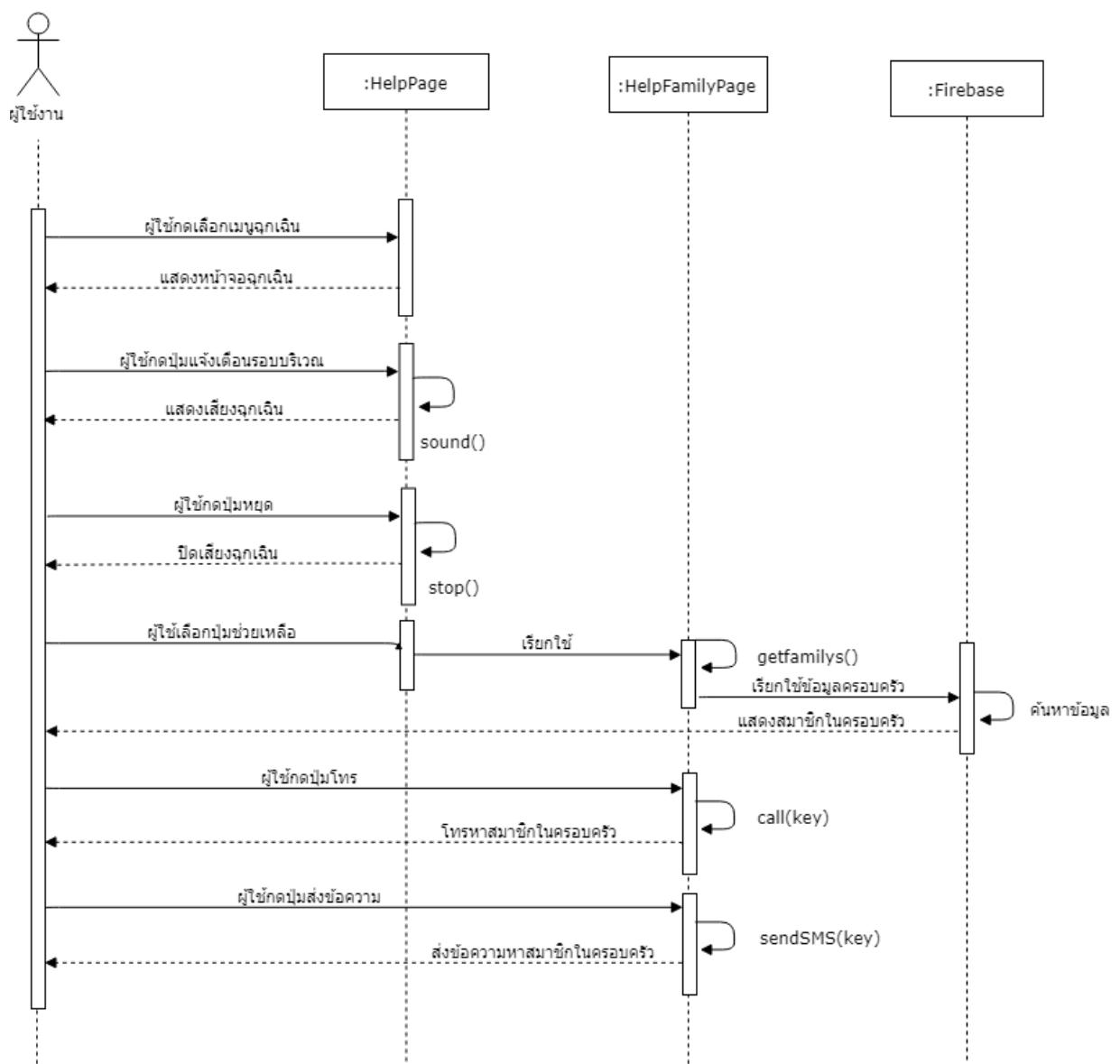
- เมื่อผู้ใช้กดเลือกหน้าข้อมูลส่วนตัว ระบบจะเรียกฟังก์ชัน ionViewWillEnter() เพื่อเรียกข้อมูลผู้ใช้และโพสท์จากฐานข้อมูลมาแสดงหน้าข้อมูลส่วนตัว
- เมื่อผู้ใช้กดเลือกปุ่มเปลี่ยนรูปประจำตัว ระบบจะเรียกฟังก์ชัน changeProfilePicture() เพื่อให้ผู้ใช้เลือกรูปแบบการแก้ไขรูปภาพจาก camera plugin เพื่อใช้งานกล้องและแกลลอรี หลังจากนั้นจะส่งข้อมูล url ไปยังฐานข้อมูลเพื่อทำการเพิ่มหรือแก้ไขข้อมูล แล้วส่งสถานะกลับมายังผู้ใช้งาน ถ้าหากสามารถเปลี่ยนได้จะส่งข้อความว่า แก้ไขรูปสำเร็จ แต่ถ้าไม่สามารถแก้ไขรูปได้จะส่งข้อความว่า แก้ไขไม่สำเร็จ
- เมื่อผู้ใช้เลือกปุ่มแก้ไขข้อมูลส่วนตัว ระบบจะเรียกฟังก์ชัน getUserEdit() ในคลาส EditUserPage จากนั้นจะแสดงหน้าแก้ไขข้อมูลส่วนตัว เมื่อผู้ใช้กรอกข้อมูลส่วนตัวเรียบร้อยแล้ว จะเรียกฟังก์ชัน edituser() จากนั้นจะส่งข้อมูลที่ถูกแก้ไขไปฐานข้อมูล ฐานข้อมูลตรวจสอบและอัพเดทข้อมูลที่ถูกแก้ไข จากนั้นฐานข้อมูลจะส่งข้อความกลับมาว่า แก้ไขข้อมูลสำเร็จแล้ว



รูปที่ 3.23: Sequence Diagram ของการจัดการเพื่อน

จากภาพที่ 3.23 สามารถอธิบายแผนภาพ Sequence Diagram ของการจัดการเพื่อน ได้ดังนี้

- เมื่อผู้ใช้เลือกเมนูเพื่อน ระบบจะเรียกฟังก์ชัน `ngOnInit()` ในคลาส `FriendsPage` จะไป get ข้อมูลจาก Firebase เพื่อมาแสดง
- เมื่อผู้ใช้คลิกเลือกเมนูเพื่อน จะเรียกใช้คลาส `BuddyPage` จากนั้นระบบจะไป get ข้อมูล คำร้องขอเพื่อนและเพื่อนจาก Firebase มาแสดงในคลาส
- เมื่อผู้ใช้เลือกปุ่มเพิ่มเพื่อนในคลาส `BuddyPage` จะส่งข้อมูล `id` ของผู้ และ `id` ของเพื่อนไปเก็บไว้ในฐานข้อมูล หลังจากนั้นฐานข้อมูลจะส่งผลการบันทึกข้อมูลสำเร็จมาที่คลาส และระบบจะส่งข้อความว่าเพิ่มเพื่อนเรียบร้อยแล้วมายังผู้ใช้
- ผู้ใช้เลือกปุ่มแชทในคลาส `FriendsPage` ระบบจะเรียกใช้ฟังก์ชัน `ionViewDidEnter()` ในคลาส `BuddychatPage` เพื่อไปเรียกข้อมูลการแชททั้งหมดจากฐานข้อมูล จากนั้นฐานข้อมูล จะส่งค่า `Json` กลับคืนมายังคลาส `BuddychatPage` และจึงแสดงข้อความแชทให้ผู้ใช้
- เมื่อผู้ใช้ยืนยันคำร้องขอเพื่อน ระบบจะเรียกฟังก์ชัน `accept(item)` จะรับข้อมูล `Json` ของเพื่อนเพื่อติด `id` ไปเก็บไว้ในฐานข้อมูล จากนั้นฐานข้อมูลจะส่งผลการบันทึกข้อความคืนมายังคลาส
- เมื่อผู้ใช้เลือกปุ่มลบเพื่อน ระบบจะทำการเรียกใช้ฟังก์ชัน `ignore(item)` จากนั้นจะส่งค่า `id` ของเพื่อนไปเช็คในฐานข้อมูล และลบข้อมูลผู้ใช้นั้น และจึงส่งผลการลบข้อมูลกลับมายังคลาส



รูปที่ 3.24: Sequence Diagram ของฉุกเฉิน

จากภาพที่ 3.24 สามารถอธิบายแผนภาพ Sequence Diagram ของฉุกเฉิน ได้ดังนี้ ผู้ใช้กดเลือกเมนูฉุกเฉินระบบจะแสดงหน้าจอฉุกเฉิน เมื่อผู้ใช้กดปุ่มแจ้งเตือนรอบบริเวณระบบจะเรียกฟังก์ชัน sound() เพื่อใช้งาน NativeAudio Plugin ที่ใช้สำหรับแสดงเสียงแล้วแสดงเสียงแก่ผู้ใช้งาน เมื่อผู้ใช้คลิกที่ปุ่มหยุด ระบบจะหยุดเสียงที่ผู้ใช้ได้กดปุ่มแจ้งเตือนรอบบริเวณก่อนหน้านี้ เมื่อผู้ใช้เลือกปุ่มช่วยเหลือ จะเรียกใช้คลาส HelpFamilyPage จากนั้น Contructor จะเรียกข้อมูลสมาชิกครอบครัวในฐานข้อมูลมาแสดง ถ้าผู้ใช้กดปุ่มโทร ระบบจะเรียกฟังก์ชัน call(key) จากนั้นจะ get เบอร์มือถือของเพื่อนเพื่อใช้ CallNumber Plugin สำหรับโทร เมื่อผู้ใช้กดปุ่มส่งข้อความ ระบบจะเรียกฟังก์ชัน sendSMS(key) ที่อยู่ภายใต้คลาส HelpFamilyPage และเรียกใช้ SocialSharing Plugin ในการส่ง SMS ที่ มีข้อความว่า ”ช่วยด้วย !!!” นี่ (ชื่อผู้ใช้) เอง ตอนนี้มีปัญหาช่วยติดต่อกลับที่ (เบอร์มือถือผู้ใช้) ด้วยนะ ด่วน ๆ”

บทที่ 4

การพัฒนาระบบ

ในบทนี้จะกล่าวถึงการสร้างแอปพลิเคชันสูงวัยมายเฟรนด์ โดยนำผลที่ได้จากการวิเคราะห์ และออกแบบระบบมาสร้างเป็นระบบงานซึ่งจะอธิบายถึงตัวอย่างการเขียน โปรแกรมการทำงาน ของระบบในส่วนต่างๆ ดังต่อไปนี้

- 4.1 การสมัครสมาชิก
- 4.2 การเข้าสู่ระบบ
- 4.3 ส่วนจัดการเซทบอท
- 4.4 การจัดการส่วนของ Dialogflow
- 4.5 การจัดการโพสท์และคอมเม้นท์
 - 4.5.1 การเพิ่มโพสท์
 - 4.5.2 การลบโพสท์
 - 4.5.3 การกดถูกใจโพสท์
 - 4.5.4 การเลือกหมวดโพสท์
 - 4.5.5 การจัดการคอมเม้นท์
- 4.6 การค้นหาเพื่อน
- 4.7 การเพิ่มการแจ้งเตือนการท่านยา

4.1 การสมัครสมาชิก

เมื่อผู้ใช้กรอกข้อมูลการสมัครเสร็จแล้วทำการกดปุ่มสมัครสมาชิก ระบบจะมีการทำงาน แสดงดังรูปที่ 4.1

```

1 save(user) {
2     if (this.user.email == null || this.user.
3         password == null) {
4             this.toastCtrl.create({
5                 message: "กรุณาระบุชื่อผู้ใช้หรือรหัสผ่านให้ถูกต้อง",
6                 duration: 3000,
7                 position: 'top',
8             }).present();
9     } else {
10         this.registerService.SaveUser(user).then
11             (async (data) => {
12                 await firebase.auth().signOut();
13                 this.user.email = "";
14                 this.user.password = "";
15                 this.toastCtrl.create({
16                     message: "บันทึกข้อมูลสำเร็จ",
17                     duration: 3000,
18                     position: 'top',
19                 }).present();
20             }).catch(() => {
21                 this.toastCtrl.create({
22                     message: "ชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง",
23                     duration: 3000,
24                     position: 'top'
25                 }).present();
26             })
}

```

รูปที่ 4.1: Code การทำงานของระบบเมื่อกดปุ่มสมัครสมาชิก

จากรูปที่ 4.1 โครงสร้างของไฟล์ register.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 พังก์ชัน save() เป็นพังก์ชันที่ใช้ควบคุมการทำงานในการสมัครสมาชิก
- บรรทัดที่ 2 - 7 เช็คอีเมล์และพาสเวิร์ดที่เรารอไว้ว่าจะหรือไม่ ถ้าซองได้ซองหนึ่งจะแสดงข้อความว่า ”กรุณาระบุชื่อผู้ใช้หรือรหัสผ่านให้ถูกต้อง” ถ้าไม่ว่างจะนำไปเช็คข้อมูลต่อไป
- บรรทัดที่ 9 เป็นการเรียกใช้งานพังก์ชัน saveUser() ในคลาส RegisterProvider เพื่อบันทึกข้อมูลผู้ใช้ง Authentication Firebase
- บรรทัดที่ 10 กำหนดให้ user logout เนื่องจากหลังจาก register เรียบร้อยแล้วฐานข้อมูล จะถูกเก็บ user โดยอัตโนมัติ จึงต้อง logout ออกให้อยู่ในสถานะไม่มี user
- บรรทัดที่ 11 - 12 กำหนดค่าของช่องอีเมล์และพาสเวิร์ดให้ว่างหลังจากบันทึกข้อมูลเรียบร้อยแล้ว
- บรรทัดที่ 13 - 17 ถ้าระบบบันทึกข้อมูลลงฐานข้อมูลเรียบร้อยแล้ว จะแสดงข้อความว่า ”บันทึกข้อมูลสำเร็จ”
- บรรทัดที่ 18 - 23 ถูกเรียกใช้กรณีสมัครสมาชิกไม่สำเร็จจะแสดงข้อความว่า ”ชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง”

4.2 การเข้าสู่ระบบ

เมื่อผู้ใช้กรอกข้อมูลทั้งหมดเสร็จแล้วทำการกดปุ่มเข้าสู่ระบบ ระบบจะมีการทำงาน แสดงดังรูปที่ 4.2 - 4.3

```

1 login(user) {
2   let loader = this.loadingCtrl.create({
3     spinner: 'hide',
4     content: ''
5   })
6   loader.present();
7   this.loginService.loginUser(user).then(async(data)
8     => {
9     await this.fireinfo.doc(firebase.auth()
10       .currentUser.uid).get().then((res) => {
11       if (res.data() === undefined) {
12         this.fireinfo.doc(firebase.auth()
13           .currentUser.uid).set({
14             owner: firebase.auth().currentUser.uid,
15             email: firebase.auth().currentUser.email,
16             created: firebase.firestore.FieldValue.
17               serverTimestamp(),
18             }) .then(() => {
19               loader.dismiss();
20               this.toastCtrl.create({
21                 message: "เข้าสู่ระบบสำเร็จ",
22                 duration: 3000,
23                 position: 'top',
24               }) .present();
25               this.navCtrl.push(
26                 RegisterPhotoPage);
27             })
28             .catch(() => {
29               loader.dismiss();
30               this.toastCtrl.create({
31                 message: "บันทึกข้อมูลไม่สำเร็จ",
32                 duration: 3000,
33                 position: 'top'
34               }) .present();
35             })
36           })
37         }
38       )
39     )
40   )
41 }
```

รูปที่ 4.2: การทำงานของระบบเมื่อกดเข้าสู่ระบบ

จากรูปที่ 4.2 โครงสร้างของไฟล์ login.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 - 6 เป็นการใช้ loadingController เพื่อเริ่มให้โหลดหน้าจอเมื่อฟังก์ชันถูกกดใช้
- บรรทัดที่ 7 เป็นการเรียกใช้ฟังก์ชัน loginUser(user) ที่อยู่ในคลาส LoginService เพื่อตรวจสอบการเข้าสู่ระบบว่ามีอยู่ในฐานข้อมูลหรือไม่
- บรรทัดที่ 8 - 22 เป็นการเช็คข้อมูลในฐานข้อมูลว่ามีข้อมูลหรือไม่ ถ้าไม่มีจะสร้างรายละเอียดข้อมูลผู้ใช้ไว้ในฐานข้อมูล ถ้ามีจะไม่ทำในส่วนนี้
- บรรทัดที่ 23 - 30 กรณีไม่สามารถทำการสร้างข้อมูลได้ระบบจะแจ้งว่าบันทึกข้อมูลไม่สำเร็จ

```

1 } else {
2     if(res.data().owner_name == undefined && res.
3         data().age == undefined && res.data().phone
4         == undefined && res.data().photoURL ==
5         undefined && res.data().disease == undefined)
6         {
7             console.log(data);
8             loader.dismiss();
9             this.toastCtrl.create({
10                 message: 'เข้าสู่ระบบสำเร็จ',
11                 duration: 3000,
12                 position: 'top',
13             }).present();
14             this.navCtrl.push(RegisterPhotoPage);
15         }else {
16             loader.dismiss();
17             this.toastCtrl.create({
18                 message: 'เข้าสู่ระบบสำเร็จ',
19                 duration: 3000,
20                 position: 'top',
21             }).present();
22             this.navCtrl.push(TabsPage)
23         }
24     })
25 }) .catch((err) => {
26     console.log(err);
27     loader.dismiss();
28     this.toastCtrl.create({
29         message: 'กรุณากรอกชื่อผู้ใช้หรือรหัสผ่านให้ถูกต้อง',
30         duration: 3000,
31         position: 'top',
32     }).present();
33 })
34 }
```

รูปที่ 4.3: Code การทำงานของระบบเมื่อกดเข้าสู่ระบบ (ต่อ)

จากรูปที่ 4.3 โครงสร้างของไฟล์ login.ts (ต่อ) สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 - 9 ทำการเช็คข้อมูลที่อยู่ในฐานข้อมูล ถ้าหากข้อมูลบางอย่างไม่ถูกสร้าง ระบบจะหยุด LoadingController และจะแสดงข้อความเข้าสู่ระบบสำเร็จ หลังจากนั้นจะเรียกใช้งานคลาส RegisterPhotoPage
- บรรทัดที่ 11 - 18 ถ้าหากข้อมูลถูกสร้างทั้งหมดแล้ว ระบบหยุด LoadingController และจะแสดงข้อความเข้าสู่ระบบสำเร็จ จากนั้นจะเรียกใช้คลาส TabsPage เพื่อไปยังหน้าหลัก
- บรรทัดที่ 22 - 29 ถ้าหากผู้ใช้กรอกข้อมูลไม่ครบหรือไม่ถูกต้องระบบจะแสดงข้อความว่า กรุณารอกรอชื่อผู้ใช้หรือรหัสผ่านให้ถูกต้อง

4.3 ส่วนจัดการแชทบอท

เมื่อผู้ใช้กดเลือกเมนูปุ่ม “**หั่น**” ระบบจะมีการทำงาน แสดงดังรูปที่ 4.5

```

1 sendText () {
2 window["ApiAIPPlugin"].requestText ({
3 query: messages
4 }, (response) => {
5 this.ngZone.run(() => {
6
7 if(response.result.fulfillment.speech) {
8 try {
9     console.log(response);
10    this.textres = JSON.parse(response.result.
11        fulfillment.speech);
12    this.messages.push({
13      text: this.textres.text,
14      img: this.textres.img,
15      button: this.textres.button,
16      list: this.textres.list,
17      sender: "api"
18    })
19 } catch(e) {
20     this.messages.push({
21       text: response.result.fulfillment.speech,
22       sender: "api"
23     })
24 this.content.scrollToBottom();
25 }
26 })
27 }, (error) => {
28 alert (JSON.stringify(error))
29 })
30 }
```

รูปที่ 4.4: Code การทำงานของการคุยกับแชทบอท

จากรูปที่ 4.4 โครงสร้างของไฟล์ chatbot.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 - 3 เป็นการส่งตัวแปร messages ไป query ที่ Dialogflow เพื่อหาคำตอบ
- บรรทัดที่ 4 - 17 คือคำตอบที่ถูก response กลับมาจาก Dialogflow จากนั้นนำ response.result.fulfillment เก็บไว้ในตัวแปร textres และส่งไปแสดงหน้าจอในรูปแบบข้อความหรือรูปภาพหรือลิสท์
- บรรทัดที่ 18 - 22 ถ้าข้อมูลที่ถูกส่งกลับมามีข้อผิดพลาด push ข้อความมาแสดงหน้าจอทันที

4.4 การพัฒนาในส่วนของ Dialogflow ที่ใช้ fulfillment ในการเชื่อมต่อกับ Firebase

เมื่อผู้ใช้ส่งข้อความมายัง Dialogflow เพื่อหาคำตอบที่ถูกต้อง ระบบจะมีการทำงาน แสดงดังรูปที่ 4.5

```

1 function disease(agent) {
2 var dis = request.body.queryResult.parameters.
3     disease;
4         var dis_detail = request.body.queryResult.
5             parameters.desease_detail;
6 var result = "";
7 if(dis === เบาหวานชนิดที่"1" || dis === เบาหวานชนิดที่"2" || dis === เบา
8     หวานชนิดที่"3"){
9     return admin.firestore().collection("disease").docเบาหวาน("")"
10 .collection(dis).doc(dis_detail).get().then((data) => {
11     result = JSON.stringify(data.data());
12 if(result){
13     agent.add(result);
14 }else{
15     agent.addปั๊ມรู้เลย(" ปั๊ขอหาข้อมูลก่อนนะ");
16 }
17 );
18 });
19 if(result){
20     agent.add(result);
21 }else{
22     agent.addปั๊ມรู้เลย(" ปั๊ขอหาข้อมูลก่อนนะ");
23 }
24 );
25 }
26 }
```

รูปที่ 4.5: Code ส่วนของ Dialogflow ที่ใช้ fulfillment ในการเชื่อมต่อกับ Firebase

จากรูปที่ 4.5 โครงสร้างของไฟล์ index.js อยู่ใน fulfillment ของ Dialogflow สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 เก็บค่า parameters ข้อมูลของชื่อโรคจาก Intent มาเก็บไว้ในตัวแปร dis
- บรรทัดที่ 3 เก็บค่า parameters ข้อมูลของรายละเอียดโรคจาก Intent มาเก็บไว้ในตัวแปร disdetail
- บรรทัดที่ 4 สร้างตัวแปร result ให้เท่ากับสตริงเปล่าเพื่อรับค่าที่ถูก return กลับมา
- บรรทัดที่ 5 เช็คชื่อโรคว่าเป็นข้อมูลชนิดเบاحวนหรือไม่
- บรรทัดที่ 6 - 8 เป็นการส่งชื่อโรคและรายละเอียดไปเช็คในฐานข้อมูลแล้วคืนค่ากลับมา จากนั้นแปลงข้อมูลเป็นสตริงแล้วเก็บไว้ในตัวแปร result
- บรรทัดที่ 9 - 10 ส่งค่าตัวแปร result กลับมายังแชทบอท
- บรรทัดที่ 11 - 12 ถ้าข้อมูลที่ถูกส่งกลับคืนมาเป็น null ระบบจะส่งข้อความว่าปูไม่รู้เลย ปู ขอหาข้อมูลก่อนนะ กลับมา
- บรรทัดที่ 15 - 22 ถ้าหากข้อมูลไม่ตรงกับที่ if จะทำการค้นหาและส่งกลับมาเก็บใน result และส่งกลับมายังแชทบอท

4.5 การจัดการโพสท์

4.5.1 การเพิ่มโพสท์

เมื่อผู้ใช้กรอกข้อความหรือเพิ่มรูปภาพแล้วกดปุ่มโพสท์ ระบบจะมีการทำงาน แสดงดังรูปที่

4.7 - 4.8

```

1 post() {
2   const alert = this.alertCtrl.create({
3     title: 'ประเภทโพสท์',
4     inputs: [
5       {
6         name: 'ทัวไป',
7         type: 'radio',
8         label: 'ทัวไป',
9         value: 'other',
10        },
11        {
12          name: 'กีฬา',
13          type: 'radio',
14          label: 'กีฬา',
15          value: 'sport',
16        },
17        {
18          name: 'ดนตรี',
19          type: 'radio',
20          label: 'ดนตรี',
21          value: 'music'
22        },

```

รูปที่ 4.6: Code การสร้าง Alert เมื่อคลิกโพสท์

จากรูปที่ 4.7 โครงสร้างของไฟล์ feed.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 - 17 เป็นการสร้าง alert หลังจากที่กดปุ่มโพสท์จะมีให้ผู้ใช้เลือกประเภทของโพสท์ดังนี้ ทัวไป กีฬา ดนตรี

```

1      {
2          name: 'ศาสนา',
3          type: 'radio',
4          label: 'ศาสนา',
5          value: 'religion'
6      },
7  ],
8 buttons: [
9  {
10     text: 'ยกเลิก',
11     role: 'cancel',
12     cssClass: 'secondary',
13     handler: (data) => {
14         console.log('Confirm Cancel');
15     }
16 }, {

```

รูปที่ 4.7: Code การสร้าง Alert เมื่อคลิกโพสท์ (ต่อ)

จากรูปที่ 4.7 โครงสร้างของไฟล์ feed.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 - 6 เป็นการกำหนดรายละเอียดต่างในปุ่ม alert ที่มีตัวเลือกเป็น ศาสนา
- บรรทัดที่ 8 - 15 เป็นการกำหนดปุ่มยกเลิก

```

1   text: 'โพสท์',
2   handler: (data) => {
3     let loader = this.loadingCtrl.create({
4       spinner: 'hide',
5       content: ''
7     );
8     loader.present();
9     this.CollectionService.PostsCollection().add({
10       text: this.text,
11       created: firebase.firestore.FieldValue.
12         serverTimestamp(),
13       owner: firebase.auth().currentUser.uid,
14       owner_name: firebase.auth().currentUser.
15         displayName,
16       likes: [
17         `${firebase.auth().currentUser.uid}`]:
18         false
19     },
20     likesCount: 0,
21     photoUser: firebase.auth().currentUser.photoURL
22     ,
23     type: data
24   }).then(async (doc) => {
25     if (this.image) {
26       await this._POST.uploadImgPost(doc.id,
27         this.image);
28     }
29     this.text = "";
30     this.image = undefined;
31     loader.dismiss();
32     this.getPosts();
33   }).catch((err) => {
34     loader.dismiss();
35     console.log(err);
36   })
37 }
38 ]
39 );
40 alert.present();
41 }

```

รูปที่ 4.8: Code การเพิ่มโพสท์ (ต่อ)

จากรูปที่ 4.8 โครงสร้างของไฟล์ feed.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 - 2 เป็นปุ่มโพสท์สำหรับยืนยันการโพสท์ จะส่งตัวแปร data ที่เป็นข้อ value ของ alert ที่เราได้เลือก
- บรรทัดที่ 3 เป็นการสร้าง loadingController สำหรับขอข้อมูล
- บรรทัดที่ 4 ปิดการใช้งาน spinner ของ loading
- บรรทัดที่ 5 กำหนด content ของตัวโหลด
- บรรทัดที่ 7 เปิดใช้งาน loading ให้ทำงาน
- บรรทัดที่ 8 - 18 เป็นการเพิ่มข้อมูลโพสท์ไปยังฐานข้อมูล
- บรรทัดที่ 19 - 26 เมื่อโพสท์ถูกสร้างเรียบร้อยแล้วจะตรวจสอบว่าโพสท์มีรูปภาพหรือไม่ ถ้ามีจะเรียกใช้ uploadImgPost() ในคลาส PostProvider จะส่งค่า id และ image ไปสร้างใน Storage และเรียกใช้ฟังก์ชัน getPosts() เพื่อรีเฟรชหน้า
- บรรทัดที่ 27 - 29 ถ้าหากโพสท์ไม่สามารถสร้างได้จะแสดงข้อผิดพลาดกับระบบ

4.5.2 การลับโพสท์

เมื่อเลือกที่ปุ่มลับโพสท์ ระบบจะมีการทำงาน แสดงดังรูปที่ 4.9

```
1 delete(post) {
2     let alert = this.actionSheetCtrl.create({
3         title: "คุณต้องการที่จะลบโพสต์?",  

4         buttons: [
5             {
6                 text: "ยกยื่น",
7                 handler: () => {
8                     let loader = this.loadingCtrl.create({
9                         spinner: 'hide',
10                        content: '',
11                    });
12                    loader.present();
13                    this.CollectionService.PostsCollection().doc(post.id)
14                        .delete()
15                    .then(() => {
16                        console.log("Success Uid Posts = " + post.id);
17                        this.CollectionService.CommentsCollection().where("post", "==", post.id)
18                            .get()
19                            .then((data) => {
20                                data.forEach(function (doc) {
21                                    firebase.firestore().collection("comments").doc(
22                                        doc.id).delete()
23                                    .then(() => {
24                                        console.log("Success Uid Comments = " +
25                                            doc.id);
26                                        loader.dismiss();
27                                    })
28                                .catch((err) => {
29                                    loader.dismiss();
30                                    console.log(err);
31                                })
32                            })
33                        .catch((err) => {
34                            loader.dismiss();
35                            console.log(err);
36                        })
37                    })
38                }
39            )
40        }
41    })
42    alert.present();
43 }
44 
```

รูปที่ 4.9: Code การลบโพสท์

จากรูปที่ 4.9 โครงสร้างของไฟล์ feed.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 เป็นการสร้างแอคชันชีฟเพื่อแสดงข้อความยืนยันการลบโพสท์
- บรรทัดที่ 13 เป็นการกำหนด collection และ documents เพื่อส่ง id ของโพสท์ไปลบที่ฐานข้อมูล
 - บรรทัดที่ 16 ค้นหาตัวแปร post ในฐานข้อมูลที่เท่ากับ id ของโพสท์
 - บรรทัดที่ 19 เป็นการวนลูปตัวแปร data ที่ถูกส่งกลับมาไว้ในตัวแปร doc
 - บรรทัดที่ 20 เป็นการลบคอมเมนท์ตาม id ของโพสท์นั้น

```

1 this.comments = data.docs.length;
2 console.log(this.comments);
3 loader.dismiss();
4 this.getPosts();
5 }).catch((err) => {
6 console.log(err);
7 })
8 ).catch((error) => {
9 console.error("Error removing document: ", error);
10 this.getPosts();
11 });
12 }
13 },
14 {
15 text: กลับ"",
16 handler: ()=> {
17 console.log(`ไม่ได้ลบข้อมูล("")`);
18 }
19 }
20 ]
21 });
22 alert.present();
23 }

```

รูปที่ 4.10: การพัฒนาในส่วนของลบโพสท์ (ต่อ)

จากรูปที่ 4.10 โครงสร้างของไฟล์ feed.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 4 เป็นการเรียกใช้ข้อมูลในฐานข้อมูลหลังจากที่โพสท์ถูกลบเรียบร้อยแล้ว
- บรรทัดที่ 6 แสดงข้อผิดพลาดถ้าหากไม่สามารถลบโพสท์ฐานข้อมูลได้
- บรรทัดที่ 15 - 18 เป็นปุ่มสำหรับยกเลิกการลบโพสท์

4.5.3 การกดถูกใจโพสท์

เมื่อผู้ใช้เลือกปุ่มถูกใจ ระบบจะมีการทำงาน แสดงดังรูปที่ 4.11

```

1 like(post) {
2   let loader = this.loadingCtrl.create({
3     spinner: 'hide',
4     content: ''
5   });
6   loader.present();
7   let body = {
8     postId: post.id,
9     userId: firebase.auth().currentUser.uid,
10    action: post.data().likes && post.data().likes
11      [firebase.auth().currentUser.uid] == true ?
12        "unlike" : "like"
13    }
14   this.http.post("https://us-central1-oldmyfriends.
15   cloudfunctions.net/updateLikesCount", JSON.
16   stringify(body)
17   ,
18   {
19     responseType: "text"
20   }).subscribe((data) => {
21     loader.dismiss();
22     console.log(data);
23   }, (error) => {
24     loader.dismiss();
25     console.log(error);
26   })
27 }
```

รูปที่ 4.11: Code การกดถูกใจโพสท์

จากรูปที่ 4.11 โครงสร้างของไฟล์ feed.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 - 6 เป็นการสร้างการโหลดเพื่อรอการส่งกลับของข้อมูล ใช้งานเมื่อฟังก์ชัน like() ทำงาน
- บรรทัดที่ 7 - 11 เป็นการกำหนดตัวแปรที่จะส่งไปแบบ post
- บรรทัดที่ 12 - 21 เป็นการกำหนดเส้นทางเพื่อส่งข้อมูล JSON ไปตรวจสอบที่ฐานข้อมูล จากนั้นฐานข้อมูลจะเช็คข้อมูลการถูกใจแล้วบันทึก หรือแก้ไขลงฐานข้อมูล

4.5.4 การเลือกหมวดโพสท์

เมื่อผู้ใช้กรอกข้อความแล้วกดปุ่มคอมเมนต์ ระบบจะมีการทำงาน แสดงดังรูปที่ 4.12

```

1  onChange ($event) {
2    console.log($event);
3    if($event === "all") {
4      this.getPosts();
5    }else {
6      this.posts = [];
7      let loader = this.loadingCtrl.create({
8        spinner: 'hide',
9        content: ''
10       });
11      loader.present();
12      firebase.firestore().collection("posts").where(
13        "type", "==", $event).get().then((data) => {
14        data.forEach((doc) => {
15          loader.dismiss();
16          this.posts.push(doc);
17        })
18      })
}

```

รูปที่ 4.12: ส่วนของการเลือกหมวดโพสท์

จากรูปที่ 4.12 โครงสร้างของไฟล์ feed.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 ใช้งานเมื่อผู้ใช้เลือกหมวดและกดยืนยัน
- บรรทัดที่ 3 - 4 เป็นการเช็คข้อมูลที่รับเข้ามาถ้าแสดงโพสท์ทั้งหมด จะเรียกฟังก์ชัน getPosts() เพื่อแสดงโพสท์ทั้งหมด
- บรรทัดที่ 6 ถ้าเลือกอย่างอื่นจะกำหนดตัวแปร posts ให้เป็นอาร์เรย์เปล่า
- บรรทัดที่ 7 - 10 ตัวโหลดถูกเปิดใช้งาน
- บรรทัดที่ 11 - 14 ค้นหาที่ฐานข้อมูลตามประเภทที่เราส่งไป และปิดตัวโหลด จากนั้นเก็บตัวแปร doc ใส่ลงในตัวแปร posts

4.5.5 การจัดการคอมเมนท์

เมื่อผู้ใช้กรอกข้อความแล้วกดปุ่มคอมเมนท์ ระบบจะมีการทำงาน แสดงดังรูปที่ 4.13

```

1 sendComment() {
2     let loader = this.loadingCtrl.create({
3         spinner: 'hide',
4         content: ''
5     );
6     loader.present();
7     if (typeof (this.text) == "string" && this.text.
8         length > 0) {
9         firebase.firestore().collection("comments").add
10            ({
11                text: this.text,
12                post: this.post.id,
13                owner: firebase.auth().currentUser.uid,
14                owner_name: firebase.auth().currentUser.
15                    displayName,
16                created: firebase.firestore.FieldValue.
17                    serverTimestamp(),
18                photoUser: this.photoDisplay
19            }).then((doc) => {
20                console.log(doc);
21                this.text = "";
22                loader.dismiss();
23                this.getComment();
24            }).catch((err) => {
25                loader.dismiss();
26                console.log(err);
27            })
28    } else {
29        this.toastCtrl.create({
30            message: "กรุณารอให้ซองไม่ว่าง",
31            duration: 2000,
32        }).present();
33    }
34}

```

รูปที่ 4.13: Code การจัดการคอมเมนท์

จากรูปที่ 4.13 โครงสร้างของไฟล์ comments.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 - 6 เป็นการเปิดใช้งานตัวໂລດ
- บรรทัดที่ 7 เช็คตัวแปร text ว่าร่างหรือไม่
- บรรทัดที่ 8 - 14 เป็นการเก็บข้อมูลที่คอมเมนท์ไปยังฐานข้อมูล
- บรรทัดที่ 15 - 19 ถ้าหาก text จะบันทึกข้อมูลลงฐานข้อมูล หลังจากบันทึกข้อมูลเรียบร้อยแล้ว จะกำหนดตัวแปร text เท่ากับช่องว่าง และปิดตัวໂລດ จากนั้นเรียกฟังก์ชัน getComment() เพื่อรีเฟรชหน้าจอ
- บรรทัดที่ 20 - 22 ถ้าหากไม่สามารถบันทึกข้อมูลได้ ระบบจะปิดตัวໂລດและแจ้งข้อความผิดพลาดแก่ผู้ใช้
- บรรทัดที่ 24 - 28 ถ้าหากตัวแปร text ว่างจะแสดงข้อความว่ากรุณากรอกให้ช่องไม่ว่าง

4.6 การค้นหาเพื่อน

เมื่อผู้ใช้พิมพ์เพื่อค้นหาเพื่อน ระบบจะมีการทำงาน แสดงดังรูปที่ 4.14

```

1 searchuser(searchbar) {
2     let tempfriends = this.tempmyfriends;
3     var q = searchbar.target.value;
4     if (q.trim() == "") {
5         this.myfriends = this.tempmyfriends;
6         return;
7     }
8     tempfriends = tempfriends.filter((v) => {
9         if (v.data().owner_name.toLowerCase().indexOf(
10            q.toLowerCase()) > -1) {
11             return true;
12         }
13     })
14     this.myfriends = tempfriends;
15 }
```

รูปที่ 4.14: Code การค้นหาเพื่อน

จากรูปที่ 4.14 โครงสร้างของไฟล์ FamilybuddysPage.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 ถูกเรียกใช้เมื่อผู้ใช้กรอกข้อมูลเพื่อค้นโดยจะส่งมาในตัวแปร searchbar
- บรรทัดที่ 2 เก็บข้อมูลเพื่อนทั้งหมดไว้ในตัวแปร tempfriends
- บรรทัดที่ 3 เก็บข้อความที่ผู้ใช้พิมพ์ไว้ที่ตัวแปร q
- บรรทัดที่ 4 - 7 ถ้าช่องค้นหาว่างจะแสดงเพื่อนทั้งหมดใน List
- บรรทัดที่ 8 - 12 กำหนด tempfriends เท่ากับชื่อเพื่อนอย่างน้อยหนึ่งตัว จะ return เป็น true ถ้าไม่ตรงจะ return เป็น false
- บรรทัดที่ 14 ให้ตัวแปร myfriends เท่ากับ tempfriends คือเพื่อนที่ถูกค้นหาเจอ

4.7 การเพิ่มการแจ้งเตือนการทำงาน

เมื่อผู้ใช้กรอกข้อมูลยาที่ต้องการแจ้งเตือนแล้วกดปุ่มบันทึกการแจ้งเตือน ระบบจะมีการทำงาน แสดงดังรูปที่ 4.15

```

1 savemedicine() {
2     if(this.selecttype == "eat") {
3         this.type = "สำหรับรับประทาน"
4     }else {
5         this.type = "สำหรับฉีด"
6     }
7     if(this.name == undefined || this.number == undefined)
8     {
9         this.toastCtrl.create({
10             message: "กรุณากรอกข้อมูลให้ถูกต้อง",
11             duration: 2000,
12             position: 'top',
13         }).present();
14     }else{
15         firebase.firestore().collection("medicine").add({
16             uid: firebase.auth().currentUser.uid,
17             name: this.name,
18             number: this.number,
19             time: this.bangkokTime,
20             type: this.selecttype,
21             timestamp: firebase.firestore.FieldValue.
22                 serverTimestamp()
23         }).then(() => {
24             this.localNotifications.schedule({
25                 title: "ชื่อยา" : " + this.name,
26                 text: "ประเภท" : " + this.type + ", จำนวน: " + this.number,
27                 trigger: {at: new Date(this.bangkokTime)},
28                 led: 'FF0000',
29                 sound: this.setSound()
30             });
31             this.toastCtrl.create({
32                 message: "บันทึกแจ้งเตือนสำเร็จ",
33                 duration: 2000,
34                 position: 'top',
35             }).present();
36             this.viewCtrl.dismiss();
37         }
38     }
}

```

รูปที่ 4.15: Code การเพิ่มการแจ้งเตือนการทานยา

จากรูปที่ 4.15 โครงสร้างของไฟล์ SettingAlarmPage.ts สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 2 - 5 เช็คประเภทข้อมูลว่าเป็นสำหรับรับประทาน หรือสำหรับฉีด จากนั้นเก็บไว้ในตัวแปร type
- บรรทัดที่ 7 - 12 ถ้าหากชื่อยาหรือจำนวนยาไม่ได้ระบุจะแจ้งเตือนข้อความ กรุณารอกรับข้อมูลให้ถูกต้อง
- บรรทัดที่ 14 - 20 เป็นการบันทึกข้อมูลการทานยาลงฐานข้อมูล
- บรรทัดที่ 22 - 27 หลังจากบันทึกข้อมูลเสร็จแล้ว ระบบจะเรียกใช้ localNotifications plugin เพื่อเก็บข้อมูลสำหรับการแจ้งเตือนตามเวลา
- บรรทัดที่ 30 - 35 หลังจากบันทึกข้อมูลเรียบร้อยแล้ว จะแสดงข้อความว่าบันทึกแจ้งเตือนสำเร็จ และกลับไปยังหน้าจอแสดงรายการการแจ้งเตือนทั้งหมด

บทที่ 5

การทดสอบระบบ

การทดสอบการทำงานของระบบมีวัตถุประสงค์เพื่อตรวจสอบความถูกต้องของการทำงาน เพื่อให้ระบบทำงานได้อย่างถูกต้อง ซึ่งในการทดสอบแอปพลิเคชันสูงวัยมายเฟรนด์ ได้ทำการทดสอบระบบแบบ Functional testing (Black Box Testing) โดยการทดสอบแบบ Black Box Testing นั้นจำจำลองห้องระบบเป็นเหมือนกล่องดำ (Black Box) โดยที่จะไม่สนใจกระบวนการทำงานว่ามีการทำงานของฟังก์ชันในระบบอย่างไร แต่จะตรวจสอบว่าเมื่อระบบทำงานสำเร็จผลลัพธ์ที่ได้ถูกต้องหรือไม่โดยในการทดสอบระบบประกอบไปด้วย 14 ส่วนดังนี้

- 5.1) ผลการทดสอบการสมัครสมาชิก
- 5.2) ผลการทดสอบการเข้าสู่ระบบ
- 5.3) ผลการทดสอบการคุยกับแขกบอท
- 5.4) ผลการทดสอบการจัดการโพสท์
- 5.5) ผลการทดสอบการเลือกประเภทโพสท์
- 5.6) ผลการทดสอบการจัดการคอมเมนท์
- 5.7) ผลการทดสอบค้นหาเพื่อน
- 5.8) ผลการทดสอบเพิ่มการแจ้งเตือนการทำงานฯ

5.1 ผลการทดสอบการสมัครสมาชิก

ตารางที่ 5.1: ผลการทดสอบการสมัครสมาชิก

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบการสมัครสมาชิก	ผู้ใช้งานเข้ามาในหน้าสมัครสมาชิก	ระบบแสดงหน้าสมัครสมาชิก
	ผู้ใช้ไม่รอกรอกข้อมูลชื่อและรหัสผ่าน	ระบบจะแสดงข้อความว่าชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง
	ผู้ใช้กดปุ่มสมัครสมาชิกโดยกรอกชื่อซ้ำกับชื่อที่มีอยู่ในระบบ	ระบบจะแสดงข้อความว่าชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง
	ผู้ใช้กดปุ่มสมัครสมาชิกโดยกรอกชื่อไม่ซ้ำกับชื่อที่มีอยู่ในระบบ	มีข้อความแสดงบอกผู้ใช้ว่าสมัครสมาชิกสำเร็จ

5.2 ผลการทดสอบการเข้าสู่ระบบ

ตารางที่ 5.2: ผลการทดสอบการเข้าสู่ระบบ

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบการเข้าสู่ระบบ	ผู้ใช้เข้ามาในหน้าเข้าสู่ระบบ	ระบบแสดงหน้าเข้าสู่ระบบ
	ผู้ใช้กดปุ่มเข้าสู่ระบบ โดยไม่กรอก username และ password	ระบบแสดงข้อความบอกผู้ใช้ว่า กรุณากรอกชื่อผู้ใช้หรือรหัสผ่านให้ถูกต้อง
	ผู้ใช้กดเข้าสู่ระบบ โดยกรอก username ไม่ถูกต้อง	ระบบแสดงข้อความบอกผู้ใช้ว่า กรุณากรอกชื่อผู้ใช้หรือรหัสผ่านให้ถูกต้อง
	ผู้ใช้กดเข้าสู่ระบบ โดยกรอก username ถูกต้อง แต่กรอก password ไม่ถูกต้อง	ระบบแสดงข้อความบอกผู้ใช้ว่า กรุณากรอกชื่อผู้ใช้หรือรหัสผ่านให้ถูกต้อง
	ผู้ใช้กดเข้าสู่ระบบ โดยกรอก username และ password ถูกต้อง และมีชื่อเล่น รูปประจำตัว อายุ เบอร์โทรศัพท์	ระบบแสดงข้อความบอกผู้ใช้ว่า เข้าสู่ระบบสำเร็จ และ จะแสดงชื่อผู้ใช้และรูปโปรไฟล์ที่แบบบันด้านขวา
	ผู้ใช้กดเข้าสู่ระบบ โดยกรอก username และ password ถูกต้อง และไม่มีชื่อเล่น รูปประจำตัว อายุ เบอร์โทรศัพท์	ระบบแสดงข้อความบอกผู้ใช้ว่า เข้าสู่ระบบสำเร็จ และแสดงหน้าให้เพิ่มรูปประจำตัว

5.3 ผลการทดสอบการคุยกับzechborth

ตารางที่ 5.3: ผลการทดสอบการคุยกับzechborth

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบคุยกับzechborth	ผู้ใช้เข้ามายืนหน้าzechborth	ระบบจะแสดงหน้าหน้าzechborth
	ผู้ใช้ส่งข้อความโดยไม่ระบุข้อความ	ระบบจะไม่ส่งข้อความไปยังzechborth
	ผู้ใช้ส่งข้อความโดยระบุข้อความ	ระบบจะส่งข้อความไปยังzechborth
	ผู้ใช้กดเลือกปุ่มพิมพ์ด้วยเสียง และพูด	ระบบ จะ แปลง เสียง เป็นข้อความ แล้ว ส่ง ไป ยัง zechborth
	ผู้ใช้กดเลือกปุ่มพิมพ์ด้วยเสียง และไม่พูด	ระบบจะไม่ส่งข้อความไปยังzechborth
	เมื่อผู้ใช้กดปุ่มพิมพ์ด้วยเสียง ที่ข้อความ	ระบบ จะ พูด ตาม ข้อความ ที่zechborthได้ส่งกลับคืนมา

5.4 ผลการทดสอบการจัดการโพสท์

ตารางที่ 5.4: ผลการทดสอบการจัดการโพสท์

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบการสร้างโพสท์	ผู้ใช้ กดโพสท์โดยไม่ระบุข้อความ	ระบบ จะแสดง ข้อความ กรุณาระบุข้อความให้ถูกต้อง
	ผู้ใช้โพสท์โดยกรอก ข้อความ และไม่เพิ่มรูปภาพ	ระบบ จะบันทึก ข้อมูล เนื้อหาข้อความ
	ผู้ใช้โพสท์โดยกรอก ข้อความ และเพิ่มรูปภาพ	ระบบ จะบันทึก ข้อมูล ข้อความและรูปภาพ
	ผู้ใช้โพสท์โดยไม่เลือกประเภท	ระบบ จะแสดง ข้อความ กรุณาเลือกประเภท
ทดสอบการแก้ไขโพสท์	ผู้ใช้เลือกปุ่มแก้ไขโพสท์	ระบบ จะแสดงหน้าแก้ไขโพสท์
	ผู้ใช้ไม่ได้แก้ไข ข้อความโพสท์	ระบบจะกลับไปหน้ากระดานข่าว
	ผู้ใช้แก้ไขข้อความโพสท์	ระบบจะกลับไปหน้ากระดานข่าว และบันทึกข้อมูลลงฐานข้อมูล
ทดสอบการลบโพสท์	ผู้ใช้เลือกปุ่มลบโพสท์	ระบบจะแสดงตัวเลือกได้แก่ ยืนยันการลบ และกลับ
	ผู้ใช้เลือก ยืนยัน การลบโพสท์	ระบบจะทำการลบโพสท์
	ผู้ใช้เลือกกลับ	ระบบ จะ กลับไปยังหน้ากระดานข่าว

5.5 ผลการทดสอบการเลือกประเภทโพสท์

ตารางที่ 5.5: ผลการทดสอบการเลือกประเภทโพสท์

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบ การเลือกประเภทโพสท์	ผู้ใช้ เลือก ประเภท โพสท์ ทั้งหมด	ระบบ จะ แสดง โพสท์ ทั้งหมด
	ผู้ใช้ เลือก ประเภท โพสท์ ทั่วไป	ระบบจะแสดงโพสท์ทั่วไป
	ผู้ใช้ เลือก ประเภท โพสท์ กิจกรรม	ระบบจะแสดงโพสท์กิจกรรม
	ผู้ใช้ เลือก ประเภท โพสท์ ดูตัวเอง	ระบบจะแสดงโพสท์ดูตัวเอง
	ผู้ใช้ เลือก ประเภท โพสท์ ศ่าสนาน	ระบบ จะ แสดง โพสท์ ศ่าสนาน

5.6 ผลการทดสอบการจัดการคอมเมนท์

ตารางที่ 5.6: ผลการทดสอบการจัดการคอมเมนท์

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบการคอมเมนท์	ผู้ใช้เลือกปุ่มลบคอมเมนท์	ระบบ จะแสดงหน้า คอมเมนท์
	ผู้ใช้กรอกข้อความ และส่ง คอมเมนท์	ระบบ จะ ทำการ บันทึก คอมเมนท์ และ แสดง ยัง หน้าคอมเมนท์
	ผู้ใช้ไม่กรอกข้อความ และ ส่งคอมเมนท์	ระบบ จะ แสดง ข้อความ กรุณากรอกคอมเมนท์
ทดสอบ การ แก้ไข คอมเมนท์	ผู้ใช้ เลือก ปุ่ม แก้ไข คอมเมนท์	ระบบ จะ แสดง หน้า แก้ไข คอมเมนท์
	ผู้ใช้ ไม่ได้ แก้ไข ข้อความ คอมเมนท์	ระบบจะกลับไปหน้ากระดานข่าว
	ผู้ใช้ แก้ไข ข้อความ คอมเมนท์	ระบบจะกลับไปหน้าคอมเมนท์ และบันทึกข้อมูลลงฐานข้อมูล
ทดสอบการลบคอมเมนท์	ผู้ใช้เลือกปุ่มลบคอมเมนท์	ระบบจะแสดงตัวเลือกได้แก่ ยืนยันการลบ และกลับไปหน้าคอมเมนท์
	ผู้ใช้ เลือก ยืนยัน การ ลบ คอมเมนท์	ระบบ จะ ทำการ ลบ คอมเมนท์
	ผู้ใช้เลือกกลับ	ระบบ จะ กลับ ไป ยัง หน้า คอมเมนท์

5.7 ผลการทดสอบค้นหาเพื่อน

ตารางที่ 5.7: ผลการทดสอบค้นหาเพื่อน

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบการค้นหาเพื่อน	ผู้ใช้กรอกช่องค้นหาเพื่อน	ระบบ จะแสดงเพื่อนที่ถูกค้นหา
	ผู้ใช้ กรอก ไม่ ช่อง ค้นหา เพื่อน	ระบบ จะ แสดง เพื่อน ทั้งหมด

5.8 ผลการทดสอบเพิ่มการแจ้งเตือนการทนายฯ

ตารางที่ 5.8: ผลการทดสอบเพิ่มการแจ้งเตือนการทนายฯ

การทดสอบ	เงื่อนไขการทดสอบ	ผลการทดสอบ
ทดสอบเพิ่มการแจ้งเตือนการทนายฯ	ผู้ใช้ เลือก ประเภท กรอก ชื่อยา กรอกจำนวน เลือก ช่วง เวลา และ กด บันทึก ข้อมูลยา	ระบบจะบันทึกข้อมูล และ แสดง ข้อความ บันทึกข้อมูลยาเรียบร้อยแล้ว
	ผู้ใช้ไม่กรอกข้อมูล และ กด บันทึกข้อมูลยา	ระบบ จะ แสดง ข้อความ กรอกข้อมูลให้ถูกต้อง
	ผู้ใช้เลือกตั้งเวลาแจ้งเตือน	ระบบ จะ แสดง รูป แบบ เวลาแบบ 24 Hrs
	ผู้ใช้เลือกประเภท	ระบบ จะ ให้ เลือก ได้ 2 ประเภท ได้แก่ สำหรับรับ ประทาน และ สำหรับฉีด

บทที่ 6

สรุปและข้อเสนอแนะ

การดำเนินโครงการเพื่อพัฒนาแอปพลิเคชันสูงวัยมายเฟรนด์ นี้ พบว่าระบบสามารถทำงานได้ตามที่วิเคราะห์และออกแบบไว้ แต่ก็พบปัญหาและอุปสรรคระหว่างการพัฒนา ในบทนี้ผู้พัฒนาจึงขอสรุปความสามารถของระบบ ซึ่งจะแสดงปัญหาและอุปสรรค พร้อมเสนอแนวทางในการพัฒนาแอปพลิเคชันสูงวัยมายเฟรนด์ต่อไป

6.1 สรุปความสามารถของระบบ

แอปพลิเคชันสูงวัยมายเฟรนด์ ทำงานได้ดังนี้

- ระบบสามารถทำงานได้ทั้งออนไลน์และออฟไลน์
- ผู้ใช้สามารถจัดการข้อมูลผู้ใช้ได้
- ผู้สามารถจัดการโพสท์และคอมเม้นท์ได้
- ผู้ใช้สามารถโต้ตอบกับเซาฟ์บอร์ดได้
- ผู้ใช้สามารถจัดการครอบครัวได้
- ผู้ใช้สามารถจัดการแจ้งเตือนได้

6.2 ปัญหาและอุปสรรคในการพัฒนา

- LocalNotification (Plugin) การแจ้งเตือนไม่สามารถแจ้งเตือนได้ถ้าเราปิดแอปพลิเคชัน แนวทางการแก้ไข : ใช้ Background Mode Plugin เพื่อรับการรับการแจ้งเตือนแบบไม่สมบูรณ์
- Chatbot ยังไม่สามารถตอบคำถามได้ครอบคลุมทุกคำถาม แนวทางการแก้ไข : ต้องเพิ่มข้อมูลในเรื่องการสนทนาระหว่างบอทมาก ๆ

6.3 แนวทางการพัฒนาต่อ

- การพัฒนาในส่วนเซาฟ์บอร์ด จะต้องมีการเพิ่มการตรวจสอบข้อมูลที่ครอบคลุมและถูกต้องแม่นยำมากยิ่งขึ้น และสามารถเชื่อมต่อสู่แพลตฟอร์มต่างๆ ได้ในอนาคต

บรรณานุกรม

- [1] กรมกิจการผู้สูงอายุ (ผส.) (2561). ข้อมูลสถิติจำนวนผู้สูงอายุประเทศไทย ปี 2561 ด้วยระบบ power bi [ออนไลน์]. สืบค้นเมื่อ 31 ธันวาคม 2561. จาก <http://www.dop.go.th/th/know/1/153> .
- [2] Patcharee Bonkham. (2560). ผู้สูงอายุกับ 5 โรคยอดฮิต! [ออนไลน์]. สืบค้นเมื่อ 12 มิถุนายน 2560. จาก <https://www.thaihealth.or.th/Content/37137-ผู้สูงอายุกับ>.
- [3] กอบเกียรติ สารอุบล. 2549. การพัฒนา App Android. กรุงเทพฯ: มีเดีย เนทเวิร์ค.
- [4] Sleeping For Less. (2557). Activity life cycle [ออนไลน์]. สืบค้นเมื่อ 12 พฤษภาคม 2561. จาก <http://www.akexorcist.com/2016/04/why-do-we-need-to-know-about-activity-life-cycle-th.html> .
- [5] kapook (2561). Ionic คืออะไร [ออนไลน์]. สืบค้นเมื่อ 28 มิถุนายน 2561. จาก <https://mobile.kapook.com/view5432.html> .
- [6] โปรแกรมม่อน (2559). Ionic คืออะไร [ออนไลน์]. สืบค้นเมื่อ 30 มีนาคม 2559. จาก <http://blog.prscreative.com/what-is-ionic/> .
- [7] (2560). Firebase คืออะไร ไฟร์เบส บริการ backend และ แพลตฟอร์ม ครบวงจรสำหรับ นัก พัฒนา แอพ [ออนไลน์]. สืบค้น เมื่อ 24 มกราคม 2560. จาก <https://www.mindphp.com/คู่มือ/73-คืออะไร/3921-what-is-firebase-backend.html> .
- [8] Surapong. (2559). จบทุกปัญหาการแทบทด้วย “oldster” สังคมออนไลน์เพื่อผู้สูงอายุโดยเฉพาะ [ออนไลน์]. สืบค้นเมื่อ 19 มกราคม 2559. จาก <http://www.pr.kmutt.ac.th/pr/?q=news/จบทุกปัญหาการแทบทด้วย-”oldster”-สังคมออนไลน์เพื่อผู้สูงอายุโดยเฉพาะ> .
- [9] Tni.Student. (2560). แอปพลิเคชัน estudentloan [ออนไลน์]. สืบค้น เมื่อ 20 พฤษภาคม 2561. จาก <https://play.google.com/store/apps/details?id=th.co.dest.anek.studentloan> .

- [10] Kunchit Phiu-Nual. (2557). ความหมายและความสำคัญของ system architecture [ออนไลน์]. สืบค้นเมื่อ 12 พฤษภาคม 2561. จาก <https://goo.gl/6ZhGQo>.

ประวัติผู้เขียน

ชื่อ-สกุล: นายนรากร วิเชียรไชย

รหัสประจำตัวนักศึกษา: 5811403626

วันเกิด: 06 11 2539

ที่อยู่ที่สามารถติดต่อได้: 74 ม.1 ต.ภูดิน อ.เมือง จ.กาฬสินธุ์ 46000

เบอร์โทรศัพท์: (+66) 88 766 5841

อีเมลล์: narakorn.vi.58@ubu.ac.th

ระดับมัธยมต้น: โรงเรียน สหสัขนรศึกษา จังหวัด กาฬสินธุ์

ระดับมัธยมปลาย: โรงเรียน สหสัขนรศึกษา จังหวัด กาฬสินธุ์

ระดับอุดมศึกษา: ภาควิชาคณิตศาสตร์ สถิติ และคอมพิวเตอร์ สาขาวิชาการ คอมพิวเตอร์ คณะ
วิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี