

ระบบจองคิวร้านเสริมสวย  
Beauty salon queue reservation system

นางสาวปิยพร อารศรี

โครงงานนี้เป็นส่วนหนึ่งของการศึกษา  
หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์  
ภาควิชาคณิตศาสตร์ สถิติ และคอมพิวเตอร์ คณะวิทยาศาสตร์  
มหาวิทยาลัยอุบลราชธานี  
ปีการศึกษา 2562  
ลิขสิทธิ์ของมหาวิทยาลัยอุบลราชธานี

โครงการ : ระบบจองคิวร้านเสริมสวย  
Beauty salon queue reservation system  
โดย : นางสาวปิยพร อารศรี  
อาจารย์ที่ปรึกษา : ดร.ทศพร จูนิม  
ระดับการศึกษา : วิทยาศาสตร์บัณฑิต สาขาวิทยาการคอมพิวเตอร์  
ปีการศึกษา : 2562

---

ได้รับการพิจารณาให้เป็นส่วนหนึ่งของการศึกษา  
ตามหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิทยาการคอมพิวเตอร์

คณะกรรมการสอบประเมินความรู้โครงการคอมพิวเตอร์

..... อาจารย์ที่ปรึกษา  
(ดร.ทศพร จูนิม)

..... กรรมการ  
(อาจารย์ วาโย ปุยะติ)

..... กรรมการ  
(อาจารย์ วาสนา เหง้าเกษ)

..... หัวหน้าภาควิชา  
(ผศ.ดร. สุพจน์ สีบุตร)

วันที่ ... / ... / ...

## กิตติกรรมประกาศ

การพัฒนาโครงงานระบบจองคิวร้านเสริมสวย สำเร็จลุล่วงได้ด้วยความรู้และความช่วยเหลือจากหลายๆ ท่าน ข้าพเจ้าขอขอบพระคุณทุกท่าน ที่มีส่วนร่วมในการพัฒนาโครงงานนี้

ขอขอบพระคุณอาจารย์ ดร. ทศพร จูนิม อาจารย์ที่ปรึกษาโครงงานที่ได้แนะนำทฤษฎีและแนวทางในแก้ปัญหาต่าง ๆ ที่เกิดขึ้นระหว่างการพัฒนาระบบ อีกครั้งยังคอยตรวจสอบความก้าวหน้าของการทำงานเป็นระยะ ๆ รวมทั้งสร้างกำลังใจให้ผู้พัฒนาอยู่เสมอ

ขอบพระคุณอาจารย์ประจำสาขาวิทยาการคอมพิวเตอร์ อาจารย์ประจำภาควิชาคณิตศาสตร์ สถิติ และคอมพิวเตอร์ และอาจารย์ในคณะวิทยาศาสตร์ทุก ๆ ท่าน ที่คอยให้คำแนะนำ อบรมสั่งสอน และคอยช่วยเหลือข้าพเจ้าในการศึกษาตลอดมาขอบคุณเจ้าหน้าที่และบุคลากรของคณะวิทยาศาสตร์ ที่ได้อำนวยความสะดวกทางด้านอุปกรณ์และเครื่องมือต่าง ๆ

ขอบพระคุณบิดา มารดา ที่เป็นกำลังใจ คอยให้ความรักและความห่วงใยเสมอมา ตลอดจนคอยช่วยเหลือทุนทรัพย์ทางการศึกษา และอุปกรณ์ในการพัฒนาโครงงาน

ขอบคุณเพื่อน ๆ สาขาวิทยาการคอมพิวเตอร์ชั้นปีที่ 4 ที่ได้คอยช่วยแก้ไขปัญหาและให้คำปรึกษาในการพัฒนาโครงงานครั้งนี้จนเสร็จสิ้น

นางสาวปิยพร อารศรี

17 มีนาคม 62

โครงการ	:	ระบบจองคิวร้านเสริมสวย
โดย	:	นางสาวปิยพร อารศรี
อาจารย์ที่ปรึกษา	:	ดร.ทศพร จูนิม
ระดับการศึกษา	:	วิทยาศาสตร์บัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
ปีการศึกษา	:	2562

---

## บทคัดย่อ

การเสริมสวยเป็นที่นิยมและเป็นธุรกิจที่ได้รับความนิยมจากลูกค้าเป็นจำนวนมากในปัจจุบัน โดยปกติเมื่อลูกค้าต้องการใช้บริการจะมาที่ร้านโดยไม่ได้นัดหมาย ซึ่งปัญหาคือร้านมีลูกค้าที่ใช้บริการอยู่ในขณะนั้นทำให้ต้องผู้ที่เข้ามาโดยไม่ได้นัดต้องรอคิวหรือถ้ามีลูกค้ากำลังรอรับบริการอยู่เป็นจำนวนมากอาจทำให้ต้องมาใช้บริการในวันอื่นแทน แม้ในบางกรณีที่ลูกค้าโทรมาสอบถามเพื่อทำการจองคิวล่วงหน้า แต่ช่างไม่สะดวกรับโทรศัพท์เนื่องจากกำลังให้บริการลูกค้าคนอื่นอยู่

ดังนั้นผู้พัฒนาจึงมีแนวคิดสร้างเว็บแอปพลิเคชันระบบการจองคิวร้านเสริมสวยขึ้น เพื่อช่วยจัดการปัญหาดังกล่าว ระบบนี้ถูกพัฒนาด้วย React framework , material ui , nodejs และ MySQL โดยระบบสามารถให้ลูกค้าทำการจองคิว ดูคิวว่างของร้านเสริมสวย หาดำแหน่งของร้าน ดูข้อมูลทั่วไปของร้าน และเขียนรีวิวติชมได้ ในส่วนของเจ้าของร้าน สามารถเพิ่มข้อมูลทั่วไปของร้าน และจัดการการจองคิวของร้านเสริมสวยได้ ระบบการจองคิวรองรับการแสดงผลบนอุปกรณ์สมาร์ทโฟนและเว็บเบราว์เซอร์

ระบบที่พัฒนาขึ้นจะช่วยอำนวยความสะดวกในการนัดหมายล่วงหน้า ลดการรอคิวของผู้ใช้บริการ และช่วยให้การจองคิวมีระเบียบมากขึ้น

คำสำคัญ: เว็บแอปพลิเคชัน , ระบบจองคิวร้านเสริมสวย



Topic : Beauty salon queue reservation system  
Author : PIYAPHORN ARPHORNSRI  
Advisor : TOSSAPORN JOOCHIM, Ph.D..  
Degree : Bachelor of Science (Computer Science)  
Academic Year : 2019

---

## Abstract

Beauty is popular and is a business that has been popular with many customers today. Usually when customers want to use the service, will come to the shop without an appointment. The problem is that the shop has customers who use the service at that time, causing those who enter without an appointment to wait in the queue. Or if there are many customers waiting to receive the service, may cause to use the service on another day instead Even in some cases where users call to inquire to reserve a queue in advance But the technician was not comfortable to answer the phone because he was serving other customers Therefore, the developer has the idea to create a salon queue reservation system. To help manage the said problem This system was developed with the React framework, material ui, nodejs and MySQL. The system allows customers to reserve a queue. See the salon queue. Find a store location See general information of the store And can write a review In the part of the store owner Can add general information of the shop And can manage the beauty salon queue reservations The queue reservation system supports display on Smart Phon devices and web browsers. The developed system will help facilitate advance appointments. Reduce user waiting And helps to make queue reservations more organized.

Keywords: Web Application,Beauty salon queue reservation system

# สารบัญ

	หน้า
กิตติกรรมประกาศ . . . . .	ค
บทคัดย่อภาษาไทย . . . . .	ง
บทคัดย่อภาษาอังกฤษ . . . . .	จ
สารบัญ . . . . .	ฉ
สารบัญตาราง . . . . .	ณ
สารบัญภาพ . . . . .	ญ
บทที่	
1 บทนำ . . . . .	1
1.1 ที่มาและเหตุผล . . . . .	1
1.2 วัตถุประสงค์ . . . . .	1
1.3 ขอบเขตของโครงการ . . . . .	1
1.4 ประโยชน์ที่คาดว่าจะได้รับ . . . . .	2
1.5 เครื่องมือที่ใช้ในการพัฒนา (Development tools) . . . . .	2
1.5.1 ฮาร์ดแวร์ . . . . .	2
1.5.2 ซอฟต์แวร์ (Software) . . . . .	3
1.5.3 แผนการดำเนินการ . . . . .	4
2 ทฤษฎีที่เกี่ยวข้อง . . . . .	5
2.1 ความรู้พื้นฐานเกี่ยวกับ React . . . . .	5
2.1.1 React Life Cycle . . . . .	6
2.1.2 React Life Cycle . . . . .	7
2.2 ความรู้พื้นฐานเกี่ยวกับ Node.js . . . . .	8
2.2.1 node.js ทำงานแบบ event driven . . . . .	9
2.2.2 ข้อดีของ Node.js . . . . .	10
2.3 ความรู้พื้นฐานเกี่ยวกับ JavaScript . . . . .	10
2.3.1 ประโยชน์ของ JavaScript . . . . .	11
2.3.2 ข้อดีและข้อเสียของ JavaScript . . . . .	12
2.4 ความรู้พื้นฐานเกี่ยวกับ MySQL . . . . .	12

2.4.1	โครงสร้างของ MySQL . . . . .	13
2.4.2	หลักการทำงานในลักษณะ Client Server . . . . .	13
2.4.3	การทำงานของโปรแกรมของ MySQL . . . . .	14
2.4.4	คุณสมบัติของ MySQL . . . . .	14
2.5	ความรู้เกี่ยวกับ Visual Studio Code . . . . .	15
2.6	ความรู้พื้นฐานเกี่ยวกับ Google Maps API . . . . .	15
2.7	เอกสารและงานวิจัยที่เกี่ยวข้อง . . . . .	17
2.7.1	เว็บแอปพลิเคชัน Gowabi . . . . .	17
2.7.2	ข้อแตกต่างระหว่างเว็บแอปพลิเคชัน Gowabi กับเว็บของโครงการ . . . . .	17
3	การวิเคราะห์และออกแบบระบบ . . . . .	18
3.1	โครงสร้างภาพรวมของระบบ . . . . .	19
3.2	System Requirements . . . . .	20
3.2.1	Functional Requirements . . . . .	20
3.2.2	Non-functional Requirements . . . . .	21
3.3	User Interface Design . . . . .	21
3.4	Use Case Diagram . . . . .	33
3.5	Class Diagram . . . . .	42
3.6	Sequence Diagram . . . . .	52
3.7	โครงสร้างฐานข้อมูลไฟร์เบส(Firebase Database Structure) . . . . .	75
4	การพัฒนาระบบ . . . . .	88
4.1	การพัฒนาเว็บแอปพลิเคชัน . . . . .	88
4.1.1	การเชื่อมต่อ Cloud Firestore . . . . .	88
4.1.2	โครงสร้างของการสร้างหน้าเข้าสู่ระบบ . . . . .	90
4.1.3	โครงสร้างของการสร้างหน้าข่าวสาร . . . . .	93
4.1.4	โครงสร้างของการสร้างหน้าดูรายละเอียดข่าวสาร . . . . .	95
4.1.5	โครงสร้างของการสร้างหน้าสนทนา . . . . .	97
4.1.6	โครงสร้างของการสร้างหน้าปฏิทินแสดงกำหนดการ . . . . .	100
4.1.7	โครงสร้างของการสร้างหน้าสร้างประกาศ . . . . .	102
4.1.8	โครงสร้างของการสร้างหน้าอัปโหลดเอกสารที่เกี่ยวข้อง . . . . .	105

4.1.9	โครงสร้างของการสร้างหน้าสร้างกำหนดการจ้องควงเอกสาร . . . . .	108
4.2	การพัฒนาเว็บแอปพลิเคชันแอนดรอยด์แอปพลิเคชัน . . . . .	109
4.2.1	โครงสร้างของการสร้างหน้า MainActivity . . . . .	110
4.2.2	โครงสร้างของการสร้างหน้า FeedFragment . . . . .	112
4.2.3	โครงสร้างของการสร้างหน้า PostDetailActivity . . . . .	115
4.2.4	โครงสร้างของการสร้างหน้า ChatActivity . . . . .	117
4.2.5	โครงสร้างของการสร้างหน้า SignInActivity . . . . .	120
4.2.6	โครงสร้างของการสร้างหน้า ScheduleFragment . . . . .	122
4.2.7	โครงสร้างของการสร้างหน้า ScheduleFragment . . . . .	124
5	การทดสอบระบบ . . . . .	125
5.1	การทดสอบการใช้งานแอนดรอยด์แอปพลิเคชัน . . . . .	125
5.2	การทดสอบการใช้งานเว็บแอปพลิเคชัน . . . . .	133
6	สรุปและข้อเสนอแนะ . . . . .	139
6.1	สรุปความสามารถของระบบ . . . . .	139
6.1.1	เว็บแอปพลิเคชัน . . . . .	139
6.2	ปัญหาและอุปสรรคในการพัฒนา . . . . .	140
6.3	แนวทางการพัฒนาต่อ . . . . .	140
	บรรณานุกรม . . . . .	141
	ภาคผนวก . . . . .	143
	ภาคผนวก ก การติดตั้งเครื่องมือที่ใช้พัฒนาโปรแกรม . . . . .	143
	ก.1 การติดตั้ง Node.js . . . . .	143
	ก.2 การติดตั้ง React.js . . . . .	146
	ก.3 การติดตั้ง Visual Studio Code . . . . .	146
	ภาคผนวก ข คู่มือการติดตั้งระบบ . . . . .	151
	ภาคผนวก ค คู่มือการใช้งานระบบ . . . . .	152
	ประวัติผู้พัฒนา . . . . .	169

## สารบัญตาราง

ตารางที่	หน้า
1.1 ขั้นตอนการดำเนินงาน . . . . .	4
3.1 สัญลักษณ์ของ Use case Diagram . . . . .	33
3.2 อธิบาย Use Case หน้าที่ของระบบ ในภาพที่ 3.18 . . . . .	35
3.3 อธิบาย Use Case หน้าที่ของระบบ(ต่อ) ในภาพที่ 3.18 . . . . .	36
3.4 Use Case ดูข้อมูลต่างๆของร้านเสริมสวย . . . . .	36
3.5 Use Case ดูตำแหน่งของร้านเสริมสวย . . . . .	36
3.6 Use Case ค้นหาร้านเสริมสวย . . . . .	37
3.7 Use Case ดูผลงานของร้านเสริมสวย . . . . .	37
3.8 Use Case ดูคิวว่างของร้านเสริมสวย . . . . .	37
3.9 Use Case ดูรีวิวก . . . . .	38
3.10 Use Case เขียนรีวิวก . . . . .	38
3.11 Use Case จองคิวร้านเสริมสวย . . . . .	38
3.13 Use Case post ภาพผลงานทั้งหมดของร้าน . . . . .	39
3.14 Use Case เพิ่มแก้ไขและลบข้อมูลร้าน . . . . .	39
3.15 Use Case ดูการจองคิว . . . . .	39
3.16 Use Case เพิ่มข้อมูลช่าง . . . . .	40
3.17 Use Case แก้ไขข้อมูล . . . . .	40
3.18 Use Case ดูตารางงาน . . . . .	40
3.19 Use Case post ภาพผลงานของตัวเอง . . . . .	41
3.20 สัญลักษณ์ของ Class Diagram . . . . .	42
3.21 อธิบาย Class Diagram ของคลาสพื้นฐานของระบบ . . . . .	48
3.22 อธิบาย Class Diagram ของส่วนของการแสดงข่าวสาร . . . . .	49
3.23 อธิบาย Class Diagram ของส่วนของการแสดงรายการเอกสารในระบบ . . . . .	50
3.24 อธิบาย Class Diagram ของส่วนของการส่งสำเนาเอกสาร . . . . .	51
3.25 อธิบาย Class Diagram ของส่วนของการสนทนา . . . . .	51
3.26 สัญลักษณ์ของ Sequence Diagram . . . . .	52
3.27 สัญลักษณ์ของโครงสร้างฐานข้อมูลแบบ Firebase . . . . .	75
3.28 อธิบายโหมดที่ใช้เก็บข้อมูลประกาศ . . . . .	80
3.29 อธิบายโหมดที่ใช้เก็บข้อมูลเอกสารที่เกี่ยวข้อง . . . . .	81
3.30 อธิบายโหมดที่ใช้เก็บข้อมูลประวัติการสนทนา . . . . .	82
3.31 อธิบายโหมดที่ใช้เก็บข้อมูลกำหนดการ . . . . .	83
3.32 อธิบายโหมดที่ใช้เก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา . . . . .	84
3.33 อธิบายโหมดที่ใช้เก็บข้อมูลของนักศึกษา . . . . .	85
3.34 อธิบายโหมดที่ใช้เก็บข้อมูลการจองคิวของนักศึกษา . . . . .	86
3.35 อธิบายโหมดที่ใช้เก็บข้อมูลคำถามที่พบบ่อย . . . . .	87

5.1	ผลการทดสอบเมนูนำทาง . . . . .	126
5.2	ผลการทดสอบเมนูนำทาง(ต่อ) . . . . .	127
5.3	ผลการทดสอบหน้ารายละเอียดประกาศ . . . . .	128
5.4	ผลการทดสอบหน้าสนทนา . . . . .	129
5.5	ผลการทดสอบหน้าปฏิทินกำหนดการ . . . . .	130
5.6	ผลการทดสอบหน้าดาวโหลดเอกสาร . . . . .	131
5.7	ผลการทดสอบหน้าส่งภาพถ่ายสำเนาเอกสาร . . . . .	132
5.8	ผลการทดสอบหน้าจองคิวส่งเอกสาร . . . . .	133
5.9	ผลการทดสอบเมนูนำทาง . . . . .	134
5.10	ผลการทดสอบหน้ารายละเอียดประกาศ . . . . .	135
5.11	ผลการทดสอบหน้าสนทนา . . . . .	136
5.12	ผลการทดสอบหน้าปฏิทินกำหนดการ . . . . .	137
5.13	ผลการทดสอบหน้าดาวโหลดเอกสาร . . . . .	138

## สารบัญภาพ

รูปที่		หน้า
2.1	concept หลักของ React . . . . .	6
2.2	การทำงานแบบ event driven . . . . .	10
2.3	หน้าเว็บบริการของ Google Maps . . . . .	16
2.4	หน้าแรกของเว็บไซต์ Gowabi . . . . .	17
3.1	System architecture ระบบการจองคิวร้านเสริมสวย . . . . .	19
3.2	หน้าจอหลัก . . . . .	21
3.3	หน้าจอข้อมูลร้าน . . . . .	22
3.4	หน้าจอการจองคิว . . . . .	23
3.5	หน้าจอรายการ . . . . .	24
3.6	หน้าจอเขียนรีวิวก . . . . .	25
3.7	หน้าจอลงทะเบียน . . . . .	26
3.8	หน้าจอเข้าสู่ระบบ . . . . .	26
3.9	หน้าจอโปรไฟล์ร้าน . . . . .	27
3.10	หน้าจอเพิ่มข้อมูลร้าน . . . . .	28
3.11	หน้าจอเพิ่มรายการ . . . . .	29
3.12	หน้าจอเพิ่มข้อมูลช่าง . . . . .	29
3.13	หน้าจอจัดการการจองคิว . . . . .	30
3.14	หน้าจอเพิ่มรูปภาพ . . . . .	30
3.15	หน้าจอโปรไฟล์ช่าง . . . . .	31
3.16	หน้าจอติดตามงาน . . . . .	31
3.17	หน้าจอเพิ่มรูปภาพผลงานช่าง . . . . .	32
3.18	Use Case Diagram ของระบบ จองคิวร้านเสริมสวย . . . . .	34
3.19	Class Diagram ของแอปพลิเคชันระบบ XX . . . . .	43
3.20	Class Diagram ของแอปพลิเคชันระบบ XX . . . . .	44
3.21	Class Diagram ของแอปพลิเคชันระบบ XX . . . . .	45
3.22	Class Diagram ของแอปพลิเคชันระบบ XX . . . . .	46
3.23	Class Diagram ของแอปพลิเคชันระบบ XX . . . . .	47
3.24	Sequence Diagram ข้อมูลร้าน . . . . .	53
3.25	Sequence Diagram การแสดงข้อมูลโปรโมชัน . . . . .	55
3.26	Sequence Diagram การแสดงข้อมูลรีวิว . . . . .	57
3.27	Sequence Diagram การเพิ่มรีวิว . . . . .	59
3.28	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . .	61
3.29	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . .	63
3.30	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . .	65
3.31	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . .	67

3.32	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . .	69
3.33	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . .	71
3.34	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . .	73
3.35	โครงสร้างฐานข้อมูลแบบ Firebase . . . . .	76
3.36	โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ) . . . . .	77
3.37	โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ) . . . . .	78
3.38	โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ) . . . . .	79
3.39	โน้ตเก็บข้อมูลประกาศ . . . . .	80
3.40	โน้ตเก็บข้อมูลเอกสารที่เกี่ยวข้อง . . . . .	81
3.41	โน้ตเก็บข้อมูลประวัติการสนทนา . . . . .	82
3.42	โน้ตเก็บข้อมูลกำหนดการ . . . . .	83
3.43	โน้ตเก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา . . . . .	84
3.44	โน้ตเก็บข้อมูลของนักศึกษา . . . . .	85
3.45	โน้ตเก็บข้อมูลการจองคิวของนักศึกษา . . . . .	86
3.46	โน้ตเก็บข้อมูลคำถามที่พบบ่อย . . . . .	87
4.1	ไฟล์ firebaseConfig.js . . . . .	88
4.2	ไฟล์ firebaseInit.js . . . . .	89
4.3	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue . . . . .	90
4.4	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue . . . . .	91
4.5	การสร้างลอจิก (logic) ของหน้าเข้าสู่ระบบ SignIn.vue . . . . .	92
4.6	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าข่าวสาร Home.vue . . . . .	93
4.7	การสร้างลอจิก(logic)ของหน้าข่าวสาร Home.vue . . . . .	94
4.8	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้ารายละเอียดข่าวสาร ViewPost.vue . . . . .	95
4.9	การสร้างลอจิกของหน้ารายละเอียดของข่าวสาร ViewPost.vue . . . . .	95
4.10	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสนทนา Message.vue . . . . .	97
4.11	การสร้างลอจิกของหน้าสนทนา Message.vue . . . . .	99
4.12	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าปฏิทินกำหนดการ Schedule.vue . . . . .	100
4.13	การสร้างลอจิกของหน้าปฏิทินกำหนดการ Schedule.vue . . . . .	101
4.14	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างประกาศ MgPost.vue . . . . .	102
4.15	การสร้างลอจิกของหน้าหน้าสร้างประกาศ MgPost.vue . . . . .	104
4.16	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าอัปโหลดเอกสารที่เกี่ยวข้อง MgDocument.vue105	
4.17	การสร้างลอจิกของหน้าหน้าอัปโหลดเอกสารที่เกี่ยวข้อง MgDocument.vue . . .	106
4.18	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างกำหนดการจองคิวส่งเอกสาร MgQueue.vue108	
4.19	การสร้างลอจิกของหน้าสร้างกำหนดการจองคิวส่งเอกสาร MgQueue.vue . . . . .	109
4.20	ตัวแปรในคลาส MainActivity . . . . .	110
4.21	โค้ดส่วนที่ใช้ในการสร้างเมนูนำทางหลักภายในคลาส MainActivity . . . . .	111
4.22	ตัวแปรในคลาส FeedFragment . . . . .	112
4.23	โค้ดส่วนที่ใช้ในการสืบค้นข้อมูลจาก Cloude Firestore ภายในคลาส FeedFragment113	



4.24	โค้ดส่วนที่ใช้ในการดักอีเวนต์เมื่อผู้ใช้กดที่แถวประกาศในคลาส FeedFragment . .	114
4.25	โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส PostDetailActivity . .	115
4.26	โค้ดส่วนที่ใช้ในการดาวน์โหลดเอกสารของคลาส PostDetailActivity . . . . .	116
4.27	โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส ChatActivity . . . . .	117
4.28	โค้ดส่วนที่ใช้ในการสืบค้นประวัติการสนทนาของคลาส ChatActivity . . . . .	118
4.29	โค้ดส่วนที่ใช้ในการส่งข้อความของคลาส ChatActivity . . . . .	119
4.30	โค้ดส่วนที่ใช้ในการแสดงผลหน้าเข้าสู่ระบบของคลาส SignInActivity . . . . .	120
4.31	โค้ดส่วนที่ใช้ในการเข้าสู่ระบบของคลาส SignInActivity . . . . .	121
4.32	โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment . . . . .	122
4.33	โค้ดส่วนที่ใช้ในการสืบค้นข้อมูลกำหนดการของคลาส ScheduleFragment . . . .	123
4.34	โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment . . . . .	124
ก.1	หน้าเว็บดาวน์โหลด Node.js . . . . .	143
ก.2	ไฟล์ติดตั้งสำหรับติดตั้ง Node.js . . . . .	144
ก.3	หน้าต่างต้อนรับของ Node.js . . . . .	144
ก.4	หน้าต่างข้อตกลงในการใช้ Node.js . . . . .	145
ก.5	หน้าต่างเลือกโพลเดอร์ที่จะทำการติดตั้ง Node.js . . . . .	145
ก.6	หน้าต่างติดตั้ง Node.js . . . . .	146
ก.7	คำสั่งสำหรับติดตั้ง React.js . . . . .	146
ก.8	หน้าเว็บดาวน์โหลด Visual Studio Code . . . . .	147
ก.9	หน้าต่างต้อนรับของ Visual Studio Code . . . . .	148
ก.10	หน้าต่างข้อตกลงการใช้งาน Visual Studio Code . . . . .	148
ก.11	หน้าต่างที่จัดเก็บไฟล์ต่างๆ ของ Visual Studio Code . . . . .	149
ก.12	หน้าต่างการจัดการซอร์สโค้ด ของ Visual Studio Code . . . . .	149
ก.13	หน้าต่างเริ่มทำการติดตั้งทำการกด ของ Visual Studio Code . . . . .	150
ก.14	หน้าต่างเมื่อเข้าโปรแกรมหลังติดตั้งเสร็จ ของ Visual Studio Code . . . . .	150
ข.1	หน้าเว็บดาวน์โหลด ESP installer package . . . . .	151
ข.2	ไฟล์ app-debug.apk บนอุปกรณ์ . . . . .	151
ข.3	หน้าจอต้อนรับการติดตั้งแอปพลิเคชันบนอุปกรณ์แอนดรอยด์ . . . . .	151
ข.4	หน้าจอต้อนรับการติดตั้งแอปพลิเคชันบนอุปกรณ์แอนดรอยด์ . . . . .	151
ค.1	หน้าจอต้อนรับ . . . . .	152
ค.2	หน้าต่างขอสิทธิ์การเข้าถึง . . . . .	153
ค.3	หน้าจอเข้าสู่ระบบ . . . . .	154
ค.4	หน้าแสดงข่าวสาร . . . . .	154
ค.5	หน้ารายละเอียดของข่าวสาร . . . . .	155
ค.6	เมนูนำทางหลักของแอปพลิเคชัน . . . . .	156
ค.7	หน้าจอกำหนดการ . . . . .	156
ค.8	หน้าจอต้อนรับ . . . . .	157
ค.9	หน้าต่างขอสิทธิ์การเข้าถึง . . . . .	158

ค.10	หน้าแสดงกำหนดการ . . . . .	158
ค.11	หน้าจ่อเอกสาร . . . . .	159
ค.12	หน้าจ่อสนทนา . . . . .	160
ค.13	หน้าจ่อส่งเอกสาร . . . . .	161
ค.14	หน้าจ่อถ่ายภาพเอกสาร . . . . .	162
ค.15	หน้าจ่อแสดงภาพปริวิภาพถ่ายสำเนาเอกสาร . . . . .	162
ค.16	หน้าจ่อแสดงปรับแต่งภาพถ่ายสำเนาเอกสาร . . . . .	163
ค.17	หน้าจ่อแสดงภาพปริวิภาพถ่ายสำเนาเอกสาร . . . . .	163
ค.18	หน้าต่างแสดงสถานะการอัปโหลดเอกสาร . . . . .	164
ค.19	หน้าจ่อจองคิว . . . . .	165
ค.20	หน้าต่างปฏิทินเลือกวันที่ต้องการส่งเอกสาร . . . . .	166
ค.21	หน้าต่างนาฬิกาเลือกเวลาที่ต้องการส่งเอกสาร . . . . .	166
ค.22	หน้าจ่อคำถามที่พบบ่อย . . . . .	167
ค.23	หน้าเกี่ยวกับ . . . . .	168

# บทที่ 1

## บทนำ

### 1.1 ที่มาและเหตุผล

เนื่องจากปัจจุบัน การดำเนินธุรกิจร้านเสริมสวยที่ให้บริการเสริมความงาม เช่น การทำผม ตัดผม ออกแบบทรงผม อบไอน้ำ เวลาผู้ใช้บริการมาใช้บริการโดยจะมาที่ร้านเลยโดยไม่ต้องจองคิว พบว่าร้านที่มาใช้บริการมีลูกค้าเป็นจำนวนมาก อาจจะทำให้ต้องรอคิวนานหรือต้องเสียเวลามาใช้บริการในวันอื่นบางครั้งผู้ใช้บริการมีเบอร์ของร้านเสริมสวยก็จะโทรมา สอบถามคิวและจองคิว แต่ช่างตัดผมให้ลูกค้าท่านอื่นก็ไม่สามารถรับโทรศัพท์ได้ผู้พัฒนาจึงมีแนวคิดว่าจะทำระบบการจอง คิวร้านเสริมสวยขึ้น เพื่อแก้ปัญหาการรอคิวนานและให้มีความทันสมัยตลอดจนสามารถรองรับการแสดงผลบนอุปกรณ์ สมาร์ทโฟนในปัจจุบัน ทำให้สามารถจองคิวหรือติดต่อสื่อสารในเรื่องของการจองคิวทำผมกับทางร้านได้สะดวกมากยิ่งขึ้น

แนวทางการแก้ปัญหา จัดทำการพัฒนาเป็นเว็บแอปพลิเคชัน ระบบจองคิวร้านเสริมสวยที่ถูกพัฒนาขึ้นเป็นเว็บแอปพลิเคชัน จะช่วยเพิ่มระเบียบในการจัดการจองคิวให้เป็นระบบ ลดขั้นตอนการดำเนินงานที่ซับซ้อน ลดระยะเวลาในการดำเนินงาน ลดความผิดพลาดที่จะเกิดขึ้นในขั้นตอนการดำเนินงาน และช่วยเพิ่มประสิทธิภาพในการทำงาน

### 1.2 วัตถุประสงค์

1. เพื่อออกแบบและพัฒนาเว็บแอปพลิเคชัน จองคิวร้านเสริมสวย
2. เพื่อแก้ปัญหาการรอคิวร้านเสริมสวย

### 1.3 ขอบเขตของโครงการ

#### 1.3.1 เจ้าของร้าน

- สามารถลงทะเบียนเข้าสู่ระบบด้วย Email ได้
- สามารถดูคิวที่ผู้ใช้บริการได้ทำการจองคิวไว้
- สามารถ post ภาพผลงานทั้งหมดของร้านได้

- สามารถเพิ่ม แก้ไข และลบรายการให้บริการประจำร้านได้
- สามารถเพิ่ม แก้ไข และลบ ข้อมูลร้านได้
- สามารถเพิ่ม แก้ไข และลบข้อมูลตำแหน่งร้านได้

### 1.3.2 ช่างประจำร้าน

- ลงทะเบียนใช้ web ด้วย Email ได้
- สามารถดูตารางการทำงานของตนเองได้
- สามารถ post ภาพผลงานของตัวเองได้
- สามารถแก้ไขข้อมูลส่วนตัวได้

### 1.3.3 ผู้ใช้บริการ

- สามารถลงทะเบียนเข้าสู่ระบบด้วย Email ได้
- สามารถค้นหาร้านเสริมสวยได้
- สามารถจองคิวของร้านเสริมสวยได้
- สามารถดูคิวว่างของร้านเสริมสวยได้
- สามารถดูข้อมูลต่างๆของร้านเสริมสวยได้
- สามารถดูตำแหน่งของทางร้านได้
- สามารถดูผลงานของร้านได้
- สามารถเขียนรีวิว ทิชิม ได้

## 1.4 ประโยชน์ที่คาดว่าจะได้รับ

1. ช่วยลดขั้นตอนการจองคิวร้านเสริมสวย
2. ช่วยลดปัญหาในการใช้บริการที่เกิดจากการรอคิวนาน
3. ช่วยให้การจองคิวมีระเบียบมากขึ้น

## 1.5 เครื่องมือที่ใช้ในการพัฒนา (Development tools)

### 1.5.1 ฮาร์ดแวร์

1. เครื่องคอมพิวเตอร์ส่วนบุคคล (Personal computer) เพื่อใช้ในการพัฒนาโมบายแอปพลิเคชัน โดยมีคุณสมบัติอย่างน้อยดังนี้
  - ทำงานบนระบบปฏิบัติการ Elementary OS พื้นฐานการทำงานบน Windows 10

- หน่วยประมวลผลกลาง AMD Rezen(™) 5 3500U
- หน่วยประมวลผลกราฟิก AMD Radeon(™) Vega(8) Mobile Graphics
- หน่วยความจำหลักอย่างน้อย 8 กิกะไบต์ (Gigabyte, GB)
- หน่วยความจำสำรองอย่างน้อย 256 กิกะไบต์ (Gigabyte, GB)

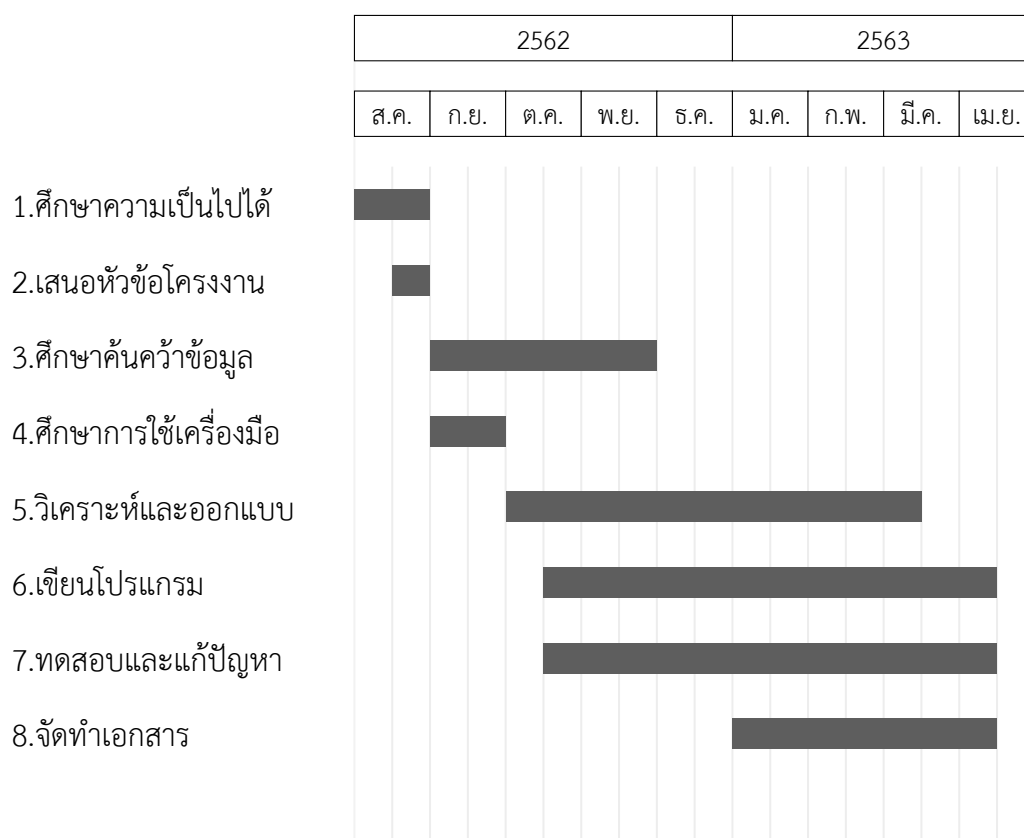
#### 1.5.2 ซอฟต์แวร์ (Software)

1. ReactJS เป็น JavaScript Framework โดยมีชุดคำสั่งและไลบรารี (Library) ให้ใช้งานมากมาย
2. Node.js คือ Cross Platform Runtime Environment หรือเรียกอีกอย่างว่า Backend Framework ใช้สำหรับเป็นเว็บเซิร์ฟเวอร์ (Web Server) ซึ่งเขียนด้วยภาษา JavaScript
3. JavaScript เป็น ภาษาที่ใช้ในการพัฒนาเว็บ Environment)
4. Xampp เป็นโปรแกรม Apache web server ไว้จำลอง web server เพื่อทดสอบระบบระหว่างพัฒนา
5. MySQL เป็น โปรแกรมระบบจัดการฐานข้อมูล
6. Google Map API เทคโนโลยีที่ใช้ในการใช้งานแผนที่
7. Visual Studio Code เครื่องมือสำหรับพัฒนาเว็บแอปพลิเคชัน

### 1.5.3 แผนการดำเนินการ

ในการสร้างระบบแนะนำสถานที่ท่องเที่ยวในจังหวัดอุบลราชธานี ผู้พัฒนาได้แบ่งขั้นตอนการดำเนินงานไว้ด้วยกัน 8 ขั้นตอน ดังตารางที่ 1.1

ตารางที่ 1.1: ขั้นตอนการดำเนินงาน



## บทที่ 2

### ทฤษฎีที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงรายละเอียดเกี่ยวกับทฤษฎีและงานวิจัยที่เกี่ยวข้องกับการพัฒนาระบบจองคิวร้านเสริมสวย โดยแบ่งเนื้อหาออกเป็น 2 ส่วน ได้แก่ ส่วนที่หนึ่งเป็นเนื้อหาพื้นฐานเกี่ยวกับทฤษฎีการเขียนโปรแกรมและเทคโนโลยีที่นำมาใช้ในการพัฒนาในหัวข้อที่ 2.1 - 2.6 ได้แก่ ความรู้พื้นฐานเกี่ยวกับ React Node.js JavaScript MySQL Visual studio code Google maps API และในส่วนที่สองเป็นเนื้อหาเกี่ยวกับเว็บแอปพลิเคชันที่เกี่ยวข้องกับโครงงานนี้เว็บแอปพลิเคชัน Gowabi

#### 2.1 ความรู้พื้นฐานเกี่ยวกับ React

React เป็น JavaScript Library ที่ถูกสร้างโดย Facebook ซึ่ง React ทำหน้าที่เป็นเพียง User Interface (UI) ที่สร้างมาจากพื้นฐานแนวความคิดแบบ Model View Controller (MVC)

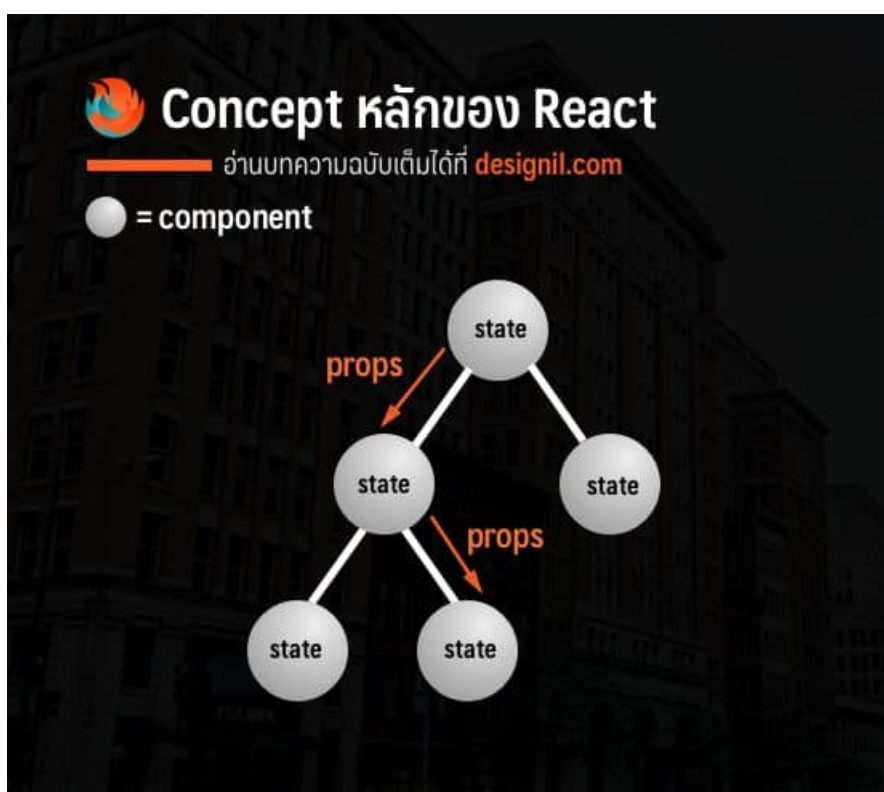
React ทำหน้าที่เฉพาะส่วน View (จาก Model View Controller) เหมาะกับงาน Web Front-End ที่สามารถแบ่งออกเป็น Web Component ย่อยๆ โดยหลักการวิเคราะห์ควรแยกให้ย่อยที่สุดเท่าที่จะทำได้ ซึ่งสามารถแบ่ง Component ออกเป็น 2 รูปแบบ คือ

- Container สำหรับบรรจุ Component หลักย่อยอื่นๆ ซึ่งไม่ควรมีการเก็บค่าใดๆ (สามารถทำหน้าที่เป็นตัวกลางในการส่งผ่านค่าได้) เน้นไปที่การจัด Layout
- Web Component คือ ส่วนที่ต้อง interact กับผู้ใช้จริงๆ เช่น ช่องกรอกข้อมูล ลิสต์แสดงข้อมูล ลาเบล (Label) และ ปุ่ม เป็นต้น ซึ่งอาจมีการเก็บค่าบางค่าเอาไว้ที่ สเตท (State) เพื่อนำมาแสดงผล

การเขียน React จำเป็นต้องมีความรู้ใน 3 ประเด็น ได้แก่

- Component – ส่วนประกอบต่างๆ ในเว็บ จะถูกมองเป็น Component
- State – ข้อมูลที่อยู่ใน Component แต่ละชิ้น เรียกว่า State
- Props – ข้อมูลที่ถูกส่งต่อจาก Component ขึ้นบนลงไปชั้นล่าง ซึ่งเรียกว่า Props (Properties)

อธิบายประเด็นหลักทั้ง 3 ประเด็น ดังรูปที่ 2.1



รูปที่ 2.1: concept หลักของ React

ที่มา: <https://www.designil.com/wp-content/uploads/2017/07/react-concept-designil.jpg>

### 2.1.1 React Life Cycle

การเขียน `render()` ฟังก์ชันใน component [?] นั้น ควรจะเขียนในแบบ pure function ซึ่งจะไม่มีการเปลี่ยน state และสร้าง side effect ต่อภายนอกทั้งสิ้น อย่างเช่น การ call external service แบบ Ajax request, Firebase calling เป็นต้น เพราะหน้าที่ของ `render()` มีแค่การ render UI เท่านั้น หากไม่สามารถทำสิ่งดังกล่าวภายใน `render()` แล้ว กิจกรรมเหล่านั้นจึงสามารถทำได้ที่ life cycle ของ React

ตลอดช่วงวงจรชีวิต [?] สามารถควบคุมเหตุการณ์ต่างๆ ที่เกิดขึ้นในการแสดงผล UI การอัปเดตข้อมูล และการ re-rendering จนกระทั่งข้อมูลนั้นหายไป โดยที่ React ได้มีการเตรียมฟังก์ชันต่างๆ ไว้ สามารถอธิบายการทำงานของฟังก์ชันได้ ดังนี้

- `componentWillMount()` : คุณสมบัติของ `componentWillMount` ไม่มีอะไรเกี่ยวกับ



การใช้งาน component เพราะยังไม่มี mount อะไรขึ้นมา โดยมีหน้าที่ คือ การกำหนดค่าเริ่มต้นสำหรับการใช้งาน

- `componentDidMount()` : เกิดขึ้นเมื่อทำการ Mount เรียบร้อย พร้อมที่จะใช้งาน โดยปกติจะใช้ในการกำหนดค่าทุกอย่างที่ต้องใช้ DOM และรับข้อมูลที่ต้องการมาแสดงผล
- `componentWillReceiveProps(nextProps)` : เมื่อ Component ทำงาน จนกระทั่งมี pro-props ใหม่เข้ามา เพื่อทำการเปลี่ยนแปลงข้อมูล `componentWillReceiveProps` จะถูกเรียก โดยมี `nextProps` เป็นตัวแปรที่ถูกส่งเข้ามา
- `shouldComponentUpdate(nextProps, nextState)` : ถูกเรียกเมื่อ component มีการเปลี่ยนแปลงด้วย `nextProps` กับ `nextState`
- `componentWillUpdate(nextProps, nextState)` : ถูกเรียกก่อนที่จะ render หลังจากได้รับค่าใหม่ของ props หรือ state คุณสมบัติของคล้ายกับ `componentWillReceiveProps`
- `componentDidUpdate(prevProps, prevState)` : ถูกเรียกทันทีหลังจากเกิดการเปลี่ยนแปลงของ component แต่จะไม่ถูกเรียกตอนครั้งแรกที่ render โดยที่ `componentDidUpdate` สามารถใช้งานได้เหมือน `componentDidMount`
- `componentWillUnmount()` : ถูกเรียกก่อนที่ component ทำการ unmount และ destroy โดยปกติแล้วจะใช้เพื่อทำการรีเซ็ต (reset) ค่าต่างๆ

### 2.1.2 React Life Cycle

การเขียน `render()` ฟังก์ชันใน component [?] นั้น ควรจะเขียนในแบบ pure function ซึ่งจะไม่มีการเปลี่ยน state และสร้าง side effect ต่อภายนอกทั้งสิ้น อย่างเช่น การ call external service แบบ Ajax request, Firebase calling เป็นต้น เพราะหน้าที่ของ `render()` มีแค่การ render UI เท่านั้น หากไม่สามารถทำสิ่งดังกล่าวภายใน `render()` แล้ว กิจกรรมเหล่านั้นจึงสามารถทำได้ที่ life cycle ของ React

ตลอดช่วงวงจรชีวิต [?] สามารถควบคุมเหตุการณ์ต่างๆ ที่เกิดขึ้นในการแสดงผล UI การอัปเดตข้อมูล และการ re-rendering จนกระทั่งข้อมูลนั้นหายไป โดยที่ React ได้มีการเตรียมฟังก์ชันต่างๆ ไว้ สามารถอธิบายการทำงานของฟังก์ชันได้ ดังนี้

- `componentWillMount()` : คุณสมบัติของ `componentWillMount` ไม่มีอะไรเกี่ยวกับการใช้งาน component เพราะยังไม่มี mount อะไรขึ้นมา โดยมีหน้าที่ คือ การกำหนด

ค่าเริ่มต้นสำหรับการใช้งาน

- `componentDidMount()` : เกิดขึ้นเมื่อทำการ Mount เรียบร้อย พร้อมทั้งจะใช้งาน โดยปกติจะใช้ในการกำหนดค่าทุกอย่างที่ต้องใช้ DOM และรับข้อมูลที่ต้องการมาแสดงผล
- `componentWillReceiveProps(nextProps)` : เมื่อ Component ทำงาน จนกระทั่งมี pro-ps ใหม่เข้ามา เพื่อทำการเปลี่ยนแปลงข้อมูล `componentWillReceiveProps` จะถูกเรียก โดยมี `nextProps` เป็นตัวแปรที่ถูกส่งเข้ามา
- `shouldComponentUpdate(nextProps, nextState)` : ถูกเรียกเมื่อ component มีการเปลี่ยนแปลงด้วย `nextProps` กับ `nextState`
- `componentWillUpdate(nextProps, nextState)` : ถูกเรียกก่อนที่จะ render หลังจากได้รับค่าใหม่ของ props หรือ state คุณสมบัติของคล้ายกับ `componentWillReceiveProps`
- `componentDidUpdate(prevProps, prevState)` : ถูกเรียกทันทีหลังจากเกิดการเปลี่ยนแปลงของ component แต่จะไม่ถูกเรียกตอนครั้งแรกที่ render โดยที่ `componentDidUpdate` สามารถใช้งานได้เหมือน `componentDidMount`
- `componentWillUnmount()` : ถูกเรียกก่อนที่ component ทำการ unmount และ destroy โดยปกติแล้วจะใช้เพื่อทำการรีเซ็ต (reset) ค่าต่างๆ

## 2.2 ความรู้พื้นฐานเกี่ยวกับ Node.js

Node.js [?] เป็นภาษาที่ทำงานอยู่ในฝั่งเซิร์ฟเวอร์ (server) ซึ่ง syntax ที่ใช้ในการเขียนคือ JavaScript และเป็นภาษาที่ออกแบบมาให้ทำงานแบบ Event-Driven หรือทำงานเมื่อเกิดเหตุการณ์ตามที่กำหนดไว้ และการทำงานแบบ Asynchronous ซึ่งสามารถทำงานในลำดับต่อไป โดยที่ไม่ต้องรอให้งานก่อนหน้าเสร็จก่อนแล้วจึงทำงานขั้นต่อไป แต่ก็สามารถกำหนดให้ทำงานแบบ Synchronous ได้เช่นกัน โดยการกำหนด Callback เมื่องานแรกทำงานเสร็จแล้ว นอกจากนี้ Node.js นั้นจะใช้ Compiler จาก Google JavaScript Engine V8

ส่วนใหญ่จะนิยมใช้ node.js ในงานที่ทำเป็นเบื้องหลัง คือ งานที่ประมวลผลฝั่งเซิร์ฟเวอร์ ซึ่งเป็นงานที่อาจจะต้อง interface กับผู้ใช้ หรือไม่ต้อง interface กับผู้ใช้ ตัวอย่างงานที่ต้อง interface กับผู้ใช้ เช่น การทำตัวเองเป็น http server ในการดึงหน้าเว็บมาแสดงผลให้กับ user หรือว่า การเปิด socket เพื่อรับส่งข้อมูลกันระหว่างเซิร์ฟเวอร์กับผู้ใช้ งาน เช่น ทำเป็นห้อง chat ทำเกม ทำระบบที่ป้อนข้อมูลเพื่อคำนวณผลลัพธ์ เป็นต้น ตัวอย่างงานที่ไม่ต้อง interface กับผู้ใช้

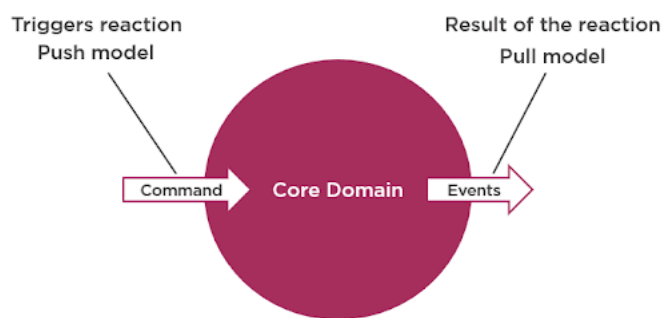
เช่น ทำ spider crawler เว็บ คือ การเปิดเว็บแล้วเก็บข้อมูลไปเรื่อยๆ หรือ โปรแกรมที่ รอรับค่าจาก streaming ต่างๆ เพื่อนำมาบันทึกไว้ ซึ่งการทำงานเหล่านี้ไม่จำเป็นต้อง interface กับผู้ใช้

node.js มีส่วนเสริมที่ชื่อว่า node package management (npm) ซึ่งเปรียบเหมือน google play ใน android หรือ app store ใน iOS ที่สามารถเอา package ที่คนอื่นเขียนเอาไว้แล้ว เพื่อแจกฟรี (free) มาต่อยอดเพื่อใช้ในงานของตนได้ โดยตัวอย่างที่ได้รับความนิยมจะเป็น underscore, async, request และ express เป็นต้น สำหรับการติดตั้ง ใช้คำสั่ง npm install ตามด้วยชื่อ package ที่ต้องการติดตั้ง [? ]

node.js มีการทำงานเป็น Asynchronous คือ การทำงานบางอย่างไม่ต้องรอให้บรรทัดนั้นทำงานเสร็จ เช่น ส่งคำสั่งไป query ข้อมูลจากฐานข้อมูล แล้วสามารถข้ามไปทำงานบรรทัดต่อไปโดยไม่ต้องรอผลจากฐานข้อมูล เมื่อการทำงานนั้นทำงานเสร็จจึงค่อยรอผลลัพธ์กลับมา ดังนั้นปัญหาจะเกิดขึ้นที่ ถ้าการทำงานต่อไปนำผลลัพธ์จากคำสั่งก่อนหน้านี้มาใช้ต่อ ซึ่งส่งผลให้เกิดการทำงานผิดพลาด เพราะผลลัพธ์ยังไม่ได้รับกลับมา

### 2.2.1 node.js ทำงานแบบ event driven

การทำงานของ node [? ] เรียกว่าเป็นการขับเคลื่อนด้วย event ต่างๆ ที่เกิดขึ้น ทำให้สามารถข้ามจาก event หนึ่งทีเสร็จแล้วไปยัง event อื่นได้ด้วยการสั่งงานต่อเนื่องกันไป หรือการสั่งให้ event หลาย event เริ่มทำงานในเวลาใกล้เคียงกัน ประโยชน์ที่ได้จาก event driven คือ การสั่งให้รอรับ event นั้นไปตลอดการณ โดยไม่เปลืองทรัพยากร เช่น การเชื่อมต่อไปยัง streaming channel ที่หนึ่ง ซึ่งอาจเป็น text หรือข้อมูลบางอย่าง เช่น ปริมาณน้ำฝน เอาไว้ หากต้นทางของ streaming ยังไม่มีข้อมูลส่งมา จะไม่เกิด event ใดๆ และ node.js จะรออยู่ แต่หากต้นทาง streaming มีข้อมูลมา node.js จะทำงานเพื่อตอบสนองต่อ event ที่เกิดขึ้นนั้นทันที สามารถแสดงการทำงานดังกล่าวได้ ดังรูปที่ 2.2



รูปที่ 2.2: การทำงานแบบ event driven

ที่มา: [http://meewebfree.com/u/i/nodejs/node\\_js\\_stage.png](http://meewebfree.com/u/i/nodejs/node_js_stage.png)

จากรูปที่ 2.2 สามารถอธิบายการทำงานได้ดังนี้ การทำงานของ stage1 เมื่อเรียกใช้การทำงานของ stage2 แล้ว ไม่จำเป็นต้องรอให้ stage2 ทำงานเสร็จก่อน ซึ่ง stage1 สามารถเรียกการทำงานของ stage3 ได้เลยโดยไม่ต้องรอการทำงานของ stage2

### 2.2.2 ข้อดีของ Node.js

- มีการทำงานแบบ Event-Driven และ Asynchronous
- เหมาะกับการทำ Web แบบ Real time
- ประหยัดทรัพยากร ในการทำงาน
- มีการประมวลผลที่รวดเร็ว

## 2.3 ความรู้พื้นฐานเกี่ยวกับ JavaScript

JavaScript [?] คือ ภาษาคอมพิวเตอร์สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ต (Internet) ที่กำลังได้รับความนิยมอย่างสูง JavaScript เป็น ภาษาสคริปต์ (script) เชิงวัตถุ ที่เรียกกันว่า "สคริปต์" ซึ่งการใช้ JavaScript ในการสร้างและพัฒนาเว็บไซต์ (ใช้ร่วมกับ HTML) จะช่วยให้เว็บไซต์ดูมีการเคลื่อนไหว สามารถตอบสนองผู้ใช้งานได้มากขึ้น โดยมีวิธีการทำงานในลักษณะ "แปลความและดำเนินงานไปทีละคำสั่ง" (interpret) หรือเรียกว่า โปรแกรมเชิงวัตถุ (Object Oriented Programming) ที่มีเป้าหมายในการ ออกแบบและพัฒนาโปรแกรมในระบบอินเทอร์เน็ต สำหรับผู้เขียนด้วยภาษา HTML สามารถทำงานข้ามแพลตฟอร์มได้ โดยทำงานร่วมกับ ภาษา HTML และภาษา Java ได้ทั้งทางฝั่งไคลเอนต์ (client) และ ทางฝั่งเซิร์ฟเวอร์

JavaScript ถูกพัฒนาขึ้นโดย บริษัท เน็ตสเคปคอมมิวนิเคชันส์ (Netscape Communications Corporation) โดยใช้ชื่อว่า Live Script ออกมาพร้อมกับ Netscape Navigator 2.0 เพื่อใช้สร้างเว็บเพจ (Web page) โดยติดต่อกับเซิร์ฟเวอร์แบบ Live Wire ต่อมาเน็ตสเคปได้ร่วมมือกับ บริษัท ซันไมโครซิสเต็มส์ (Sun Microsystems, Inc) ปรับปรุงระบบของเบราว์เซอร์ (Browser) เพื่อให้สามารถติดต่อใช้งานกับภาษาจาวา (Java) ได้ และได้ปรับปรุง LiveScript ใหม่เมื่อ ปี พ.ศ. 2538 แล้วตั้งชื่อใหม่ว่า JavaScript ซึ่ง JavaScript ทำให้การสร้างเว็บเพจมีลูกเล่นต่างๆ มากมาย และยังสามารถโต้ตอบกับผู้ใช้ได้อย่างทันที เช่น การใช้เมาส์คลิก หรือ การกรอกข้อความในฟอร์ม เป็นต้น

เนื่องจาก JavaScript ช่วยให้ผู้พัฒนา สามารถสร้างเว็บเพจได้ตรงกับความต้องการ และมีความน่าสนใจมากขึ้น ประกอบกับเป็นภาษาเปิด ที่ทุกคนสามารถนำไปใช้ได้ ดังนั้นจึงได้รับความนิยมเป็นอย่างสูง มีการใช้งานอย่างกว้างขวาง รวมทั้งได้ถูกกำหนดให้เป็นมาตรฐานโดย European Computer Manufacturer's Association (ECMA) การทำงานของ JavaScript จะต้องมีการแปลความคำสั่ง ซึ่งขั้นตอนนี้จะถูกจัดการโดยเบราว์เซอร์ (เรียกว่าเป็น client-side script) ดังนั้น JavaScript จึงสามารถทำงานได้เฉพาะบนเบราว์เซอร์ที่สนับสนุน ซึ่งปัจจุบันเบราว์เซอร์เกือบทั้งหมดสามารถสนับสนุน JavaScript แล้ว อย่างไรก็ตาม สิ่งที่ต้องระวังคือ JavaScript มีการพัฒนาเป็นเวอร์ชันใหม่ๆ ออกมาด้วย ดังนั้น ถ้านำโค้ด (code) ของเวอร์ชันใหม่ ไปรันบนเบราว์เซอร์รุ่นเก่าที่ยังไม่สนับสนุน อาจจะทำให้เกิด error ได้

### 2.3.1 ประโยชน์ของ JavaScript

- JavaScript ทำให้สามารถใช้เขียนโปรแกรมแบบง่ายได้ โดยไม่ต้องพึ่งภาษาอื่น
- JavaScript มีคำสั่งที่ตอบสนองกับผู้ใช้งาน เช่น เมื่อผู้ใช้คลิกที่ปุ่ม หรือ Checkbox สามารถสั่งให้เปิดหน้าต่างใหม่ได้ ทำให้เว็บไซต์มีปฏิสัมพันธ์กับผู้ใช้งานมากขึ้น
- JavaScript สามารถเขียนหรือเปลี่ยนแปลง HTML Element ได้ นั่นคือสามารถเปลี่ยนแปลงรูปแบบการแสดงผลของเว็บไซต์ได้ หรือหน้าแสดงเนื้อหาสามารถซ่อนหรือแสดงเนื้อหาได้โดยง่าย
- JavaScript สามารถใช้ตรวจสอบข้อมูลได้ เช่น เมื่อผู้ใช้งานกรอกข้อมูลบางเว็บไซต์ เช่น Email เมื่อบันทึกข้อมูลผิดจะมีหน้าต่างฟ้องขึ้นมาว่ากรอกผิด หรือลืมนับที่กอะไรบางอย่าง เป็นต้น

- JavaScript สามารถใช้ในการตรวจสอบผู้ใช้ได้เช่น ตรวจสอบว่าผู้ใช้ ใช้ web browser อะไร
- JavaScript สร้าง Cookies (เก็บข้อมูลของผู้ใช้ในคอมพิวเตอร์ของผู้ใช้เอง) ได้

### 2.3.2 ข้อดีและข้อเสียของ JavaScript

JavaScript [?] ทำงานบนเว็บเบราว์เซอร์ (client-side script) จึงไม่มีข้อจำกัดว่าจะใช้เซิร์ฟเวอร์แบบไหนก็ตาม เพราะ JavaScript ทำงานเฉพาะในเครื่องของผู้ใช้งานเท่านั้น ซึ่งต่างกับภาษาสคริปต์อื่น เช่น PHP , ASP, JSP หรือ Perl ซึ่งต้องประมวลผลและทำงานที่เครื่องเซิร์ฟเวอร์ (server-side script) จึงจำเป็นต้องใช้บนเซิร์ฟเวอร์ ที่สนับสนุนภาษาเหล่านี้เท่านั้นจึงจะสามารถใช้งาน server-side script ได้ แต่อย่างไรก็ตาม จากลักษณะการทำงานที่กล่าวมาก็ทำให้ JavaScript มีข้อจำกัด กล่าวคือไม่สามารถรับและส่งข้อมูลต่างๆ กับเซิร์ฟเวอร์โดยตรง เช่น การอ่านไฟล์จากเซิร์ฟเวอร์ เพื่อนำมาแสดงบนเว็บเพจ หรือรับข้อมูลจากผู้ชม เพื่อนำไปเก็บบนเซิร์ฟเวอร์ เป็นต้น ดังนั้นงานลักษณะนี้ จึงยังคงต้องอาศัยภาษา server-side script อยู่ (ความจริงมี JavaScript ที่ทำงานบนเซิร์ฟเวอร์เช่นกัน ซึ่งต้องอาศัยเซิร์ฟเวอร์ที่สนับสนุนโดยเฉพาะเช่นกัน แต่ไม่เป็นที่นิยมนัก)

นักพัฒนาเว็บส่วนใหญ่จึงนิยมใช้ JavaScript ร่วมกับ ภาษา Server Script เพื่อทำการส่งข้อมูลระหว่าง เซิร์ฟเวอร์กับเครื่องของผู้ใช้งาน ซึ่งทำให้การแสดงผลของหน้าเว็บมีความสวยงามและราบรื่นมากยิ่งขึ้น

## 2.4 ความรู้พื้นฐานเกี่ยวกับ MySQL

MySQL [?] เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System : RDBMS) ตัวหนึ่ง ซึ่งเป็นที่นิยมกันมากในปัจจุบัน โดยเฉพาะอย่างยิ่งในโลกของอินเทอร์เน็ต สาเหตุเพราะว่า MySQL เป็นฟรีแวร์ ทางด้านของฐานข้อมูลที่มีประสิทธิภาพสูง เป็นทางเลือกใหม่จากผลิตภัณฑ์ระบบจัดการฐานข้อมูลในปัจจุบัน ที่มักจะเป็นการผูกขาดของผลิตภัณฑ์เพียงไม่กี่ตัว นักพัฒนาระบบฐานข้อมูลที่เคยใช้ MySQL ต่างยอมรับในความสามารถความรวดเร็ว การรองรับจำนวนผู้ใช้ และขนาดของข้อมูลจำนวนมาก ทั้งยังสนับสนุนการใช้งานบนระบบปฏิบัติการมากมาย ไม่ว่าจะเป็น Unix , OS/2 , Mac , OS หรือ Windows ก็ตาม นอกจากนี้ MySQL ยังสามารถใช้ งานร่วมกับ Web Development Platform ทั้งหลาย ไม่ว่าจะเป็น C,

C++, Java, Perl, PHP, Python, Tcl หรือ ASP ดังนั้น MySQL จึงได้รับความนิยมอย่างมากในปัจจุบัน และมีแนวโน้มสูงยิ่งขึ้นต่อไปในอนาคต

#### 2.4.1 โครงสร้างของ MySQL

โครงสร้างภายในของ MySQL [?] คือ การออกแบบการทำงานในลักษณะของ Client Server นั้นเอง ซึ่งประกอบด้วย 2 ส่วน คือ ส่วนของผู้ให้บริการ (Server) และ ส่วนของผู้ใช้บริการ (Client) โดยในแต่ละส่วนจะมีโปรแกรมสำหรับการทำงานตามหน้าที่ของโปรแกรมนั้น ส่วนของผู้ให้บริการ (Server) จะเป็นส่วนที่ทำหน้าที่บริหารจัดการระบบฐานข้อมูล (MySQL Server) และเป็นที่จัดเก็บข้อมูลทั้งหมด ข้อมูลที่เก็บไว้นี้มีข้อมูลที่จำเป็นสำหรับการทำงานกับระบบฐานข้อมูล และข้อมูลที่เกิดจากการที่ผู้ใช้แต่ละคนสร้างขึ้นมา ส่วนของผู้ใช้บริการ (Client) คือ ส่วนที่ผู้ใช้ใช้งาน โดยโปรแกรมสำหรับใช้งานในส่วนนี้ได้แก่ MySQL , Client , Access . Web Development เป็นต้น

#### 2.4.2 หลักการทำงานในลักษณะ Client Server

เครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็น Client หรือ Server อาจจะอยู่บนเครื่องเดียวกันหรือแยกเครื่องกันก็ได้ทั้งนี้ขึ้นอยู่กับลักษณะการทำงาน หรือการกำหนดของผู้บริหารระบบตามปกติ ถ้าเป็นการทำงานลักษณะ Web-based มีการใช้ฐานข้อมูลขนาดเล็ก ตัว MySQL Server และ Client มักจะมีอยู่บนเครื่องเดียวกัน โดยเครื่องคอมพิวเตอร์ดังกล่าวจะต้องมีทรัพยากรเพื่อการทำงาน เช่น เนื้อที่ฮาร์ดดิสก์ RAM มากพอสมควร แต่สำหรับการทำงานจริง (Real-world Application) มักจะแยก Client และ Server ออกเป็นคนละเครื่องกัน และสามารถรองรับงานได้ดีมากกว่า ดังนั้น ผู้บริหารระบบหรือผู้กำหนดนโยบายสำหรับการทำงานเครือข่ายจะต้อง คำนึงถึงเรื่องที่เกี่ยวข้องเหล่านี้ให้ดี เพื่อที่จะทำให้ระบบมีการทำงานรับการให้บริการแก่ผู้ใช้ได้อย่างมีประสิทธิภาพ และข้อมูลมีความปลอดภัยมากที่สุด หลักการทำงานในลักษณะ Client Server มีดังนี้

- ที่ฝั่งของ Server จะมีโปรแกรมหรือระบบสำหรับจัดการฐานข้อมูลทำงานรออยู่ เพื่อเตรียมหรือรอ คอยการร้องขอการให้บริการจาก Client
- เมื่อมีการร้องขอการให้บริการเข้ามา Server จะทำการตรวจสอบ เช่น อาจจะมีการให้ผู้ใช้บริการระบุชื่อและรหัสผ่าน และสำหรับ MySQL สามารถกำหนดได้ว่าจะอนุญาตหรือ

ปฏิเสธ Client ใดๆ ในระบบที่จะเข้าใช้บริการอีกด้วย

- ถ้าผ่านการตรวจสอบ Server ก็จะอนุมัติการให้บริการแก่ Client ที่ร้องขอการใช้ บริการ นั้นๆ ต่อไปและถ้าในกรณีที่ไม่ได้ รับการอนุมัติServer ก็จะส่งข่าวสารความผิดพลาดแจ้ง กลับไปที่ Client ที่ร้องขอการใช้ บริการนั้น

#### 2.4.3 การทำงานของโปรแกรมของ MySQL

MySQL ถือเป็นระบบจัดการฐานข้อมูล (Database Management System : DBMS) ฐานข้อมูลมีลักษณะเป็นโครงสร้างของการเก็บรวบรวมข้อมูล การที่จะเพิ่มเติม เข้าถึงหรือประมวลผล ข้อมูลที่เก็บในฐานข้อมูลจำเป็นจะต้องอาศัยระบบจัดการ ฐานข้อมูล ซึ่งจะทำหน้าที่เป็นตัวกลาง ในการจัดการกับข้อมูลในฐานข้อมูลทั้งสำหรับการ ใช้งานเฉพาะ และรองรับการทำงานของแอปพลิเคชัน (Application) ที่ต้องการใช้งานข้อมูลในฐานข้อมูล เพื่อให้ได้รับความสะดวกในการจัดการกับข้อมูลจำนวนมาก MySQL ทำหน้าที่เป็นทั้งตัวฐานข้อมูลและระบบจัดการฐานข้อมูล ดังนี้

- MySQL เป็นระบบจัดการฐานข้อมูลแบบ relational ฐานข้อมูลแบบ relational จะทำการเก็บข้อมูลทั้งหมดในรูปแบบของตารางแทนการเก็บข้อมูลทั้งหมดลงในไฟล์ เพียงไฟล์เดียว ทำให้ทำงานได้รวดเร็วและมีความยืดหยุ่น นอกจากนั้น แต่ละตารางที่เก็บข้อมูลสามารถเชื่อมโยงเข้าหากันทำให้สามารถรวมหรือจัด กลุ่มข้อมูลได้ตามต้องการ โดยอาศัยภาษา SQL ที่เป็นส่วนหนึ่งของโปรแกรม MySQL ซึ่งเป็นภาษามาตรฐานในการเข้าถึงฐานข้อมูล
- MySQL แจกจ่ายให้ใช้งานแบบ Open Source นั่นคือ ผู้ใช้งาน MySQL ทุกคนสามารถใช้งานและปรับแต่งการทำงานได้ตามต้องการ สามารถดาวน์โหลดโปรแกรม MySQL ได้จากอินเทอร์เน็ตและนำมาใช้งานโดยไม่มีค่าใช้จ่าย

#### 2.4.4 คุณสมบัติของ MySQL

1. สนับสนุน Cross-platform support
2. รองรับ Stored procedures
3. รองรับ Triggers และ Cursors
4. สนับสนุน Information schema
5. สนับสนุน SSL
6. รองรับการทำ Query caching



7. รองรับการทำให้ Sub-SELECTs
8. รองรับการทำให้ Replication ทั้งแบบ Master-Master Replication และ Master-Slave Replication
9. รองรับ Unicode

## 2.5 ความรู้เกี่ยวกับ Visual Studio Code

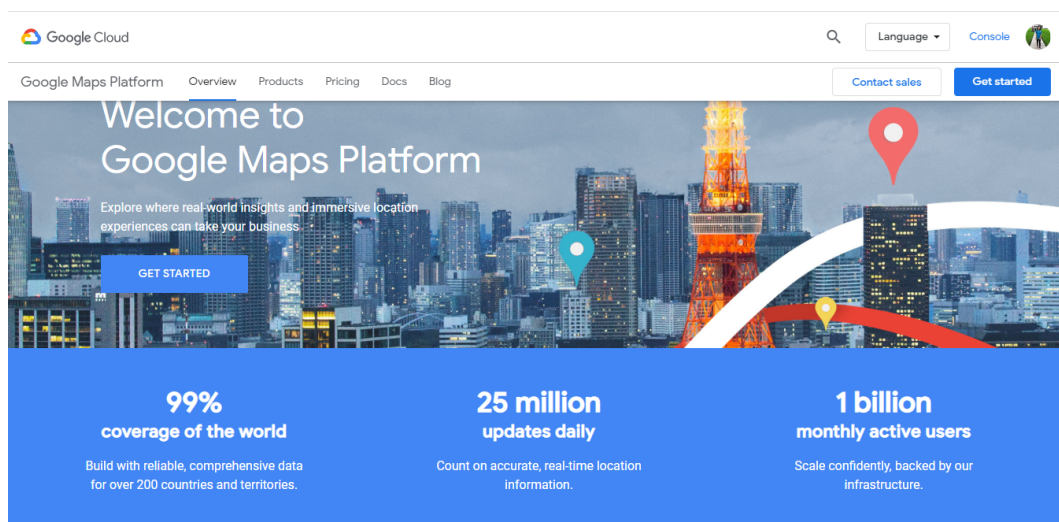
วิชวล สตูดิโอโค้ด (Visual Studio Code หรือ VSCode) [?] เป็นโปรแกรม Code Editor ที่ใช้ในการแก้ไขและปรับแต่งโค้ดถูกพัฒนาโดยค่ายไมโครซอฟท์(Microsoft) มีการพัฒนาออกมาในรูปแบบของ OpenSource จึงสามารถนำมาใช้งานได้แบบไม่ต้องเสียค่าใช้จ่าย Visual Studio Code เหมาะสำหรับนักพัฒนาโปรแกรมที่ต้องการใช้งานข้ามแพลตฟอร์ม (Cross-platform) โดยจะรองรับการใช้งานทั้งบนระบบปฏิบัติการ Windows, macOS และ Linux ซึ่งภาษาที่ Visual Studio Code รองรับการทำงานซึ่งมีมากกว่า 30 ภาษาโปรแกรม เช่น C++, C, CSS, Dockerfile, HTML, JavaScript, JSON, Less, Markdown, PHP, Python, Sass, TypeScript, Node.js และ Java เป็นต้น

## 2.6 ความรู้พื้นฐานเกี่ยวกับ Google Maps API

Google Maps API [?] เป็นชุด API ของ Google สำหรับพัฒนา web application และ mobile application (Android, ios) มีไว้สำหรับเรียกใช้แผนที่และชุด service ต่างๆ มากมายให้เรียกใช้ เช่น

- ชุดควบคุมแผนที่ (Map Control)
- การปรับแต่งแผนที่ (Styled Map)
- การนำทางจากจุดหนึ่งไปยังอีกจุดหนึ่ง (Directions Service)
- Street view
- การดึงข้อมูล POI (Point of Interest) คือข้อมูลสถานที่ต่างๆ ที่ Google รวบรวมไว้ เช่น โรงแรม ห้างสรรพสินค้า โรงเรียน สถานที่ราชการ เป็นต้น เพื่อให้ผู้ใช้งานนำมาใช้งานได้
- การแปลงที่อยู่เป็นพิกัด Latitude และ Longitude (Geocoding Service)
- แผนที่รองรับระบบ 3 มิติ ผู้ใช้งาน สามารถปรับมุมมองเป็นลักษณะ 3 มิติได้ ซึ่งระบบนี้ให้บริการบางพื้นที่

การใช้งาน Google Maps API สามารถขอใช้งานโดยใช้บัญชี Gmail และเข้าไปที่เว็บ  
<https://cloud.google.com/maps-platform/> ดังรูปที่ 2.3



รูปที่ 2.3: หน้าเว็บบริการของ Google Maps

ที่มา : <https://cloud.google.com/maps-platform/>

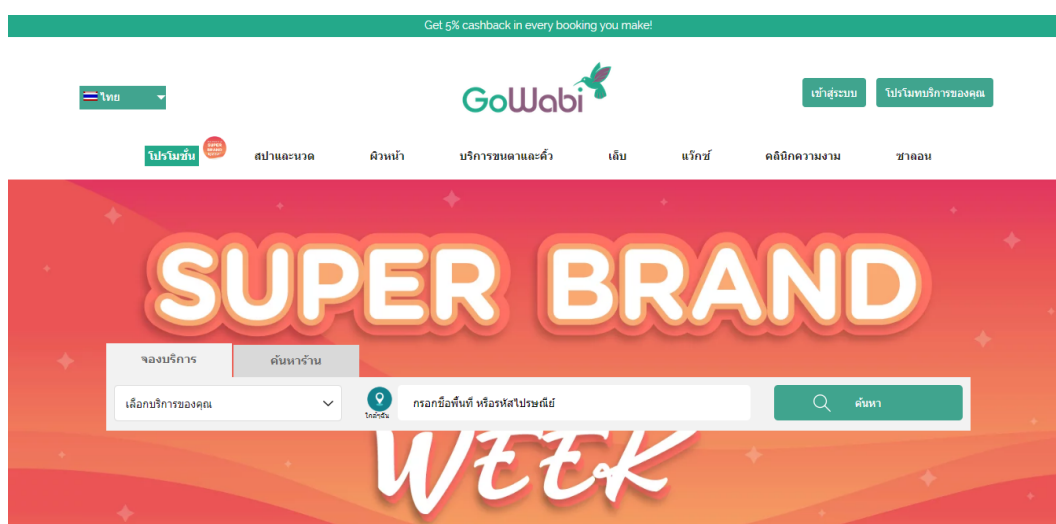
## 2.7 เอกสารและงานวิจัยที่เกี่ยวข้อง

### 2.7.1 เว็บไซต์แอปพลิเคชัน Gowabi

Gowabi[?] ] เป็นเว็บไซต์และแอปพลิเคชันที่ให้บริการเกี่ยวกับการค้นหาและจองคิวร้านเสริมสวย มีฟังก์ชันการทำงานพื้นฐานอันได้แก่ การค้นหา การจองคิว ดูข้อมูล เป็นต้น

### 2.7.2 ข้อแตกต่างระหว่างเว็บไซต์แอปพลิเคชัน Gowabi กับเว็บของโครงการ

เว็บไซต์แอปพลิเคชันของ Gowabi ยังไม่มีฟังก์ชันเลือกรายการ และช่างในการจองคิว ผู้พัฒนาจึงได้ทำฟังก์ชันเลือกรายการและช่างที่เหมาะสมตามรสนิยมของผู้ใช้เพิ่มลงในเว็บของโครงการ



รูปที่ 2.4: หน้าแรกของเว็บไซต์ Gowabi

ที่มา :<https://www.gowabi.com>

## บทที่ 3

### การวิเคราะห์และออกแบบระบบ

การวิเคราะห์และออกแบบระบบก่อนดำเนินการจริงเป็นอีกหนึ่งขั้นตอนที่มีความสำคัญมาก เพราะการวิเคราะห์และออกแบบระบบนั้นเป็นการกระทำที่ทำให้ผู้พัฒนาเห็นรายละเอียดส่วนย่อยของงานทั้งหมด เพิ่มประสิทธิภาพในการวางแผน การทำงาน และยังช่วยลดปัญหาที่อาจเกิดขึ้นในระหว่างพัฒนา เพื่อให้ระบบมีความสมบูรณ์มากยิ่งขึ้น เนื่องจากการวิเคราะห์และออกแบบระบบนั้นจะช่วยให้ให้บริการ จัดการทรัพยากรได้อย่างคุ้มค่าและตรงตามความต้องการของระบบ

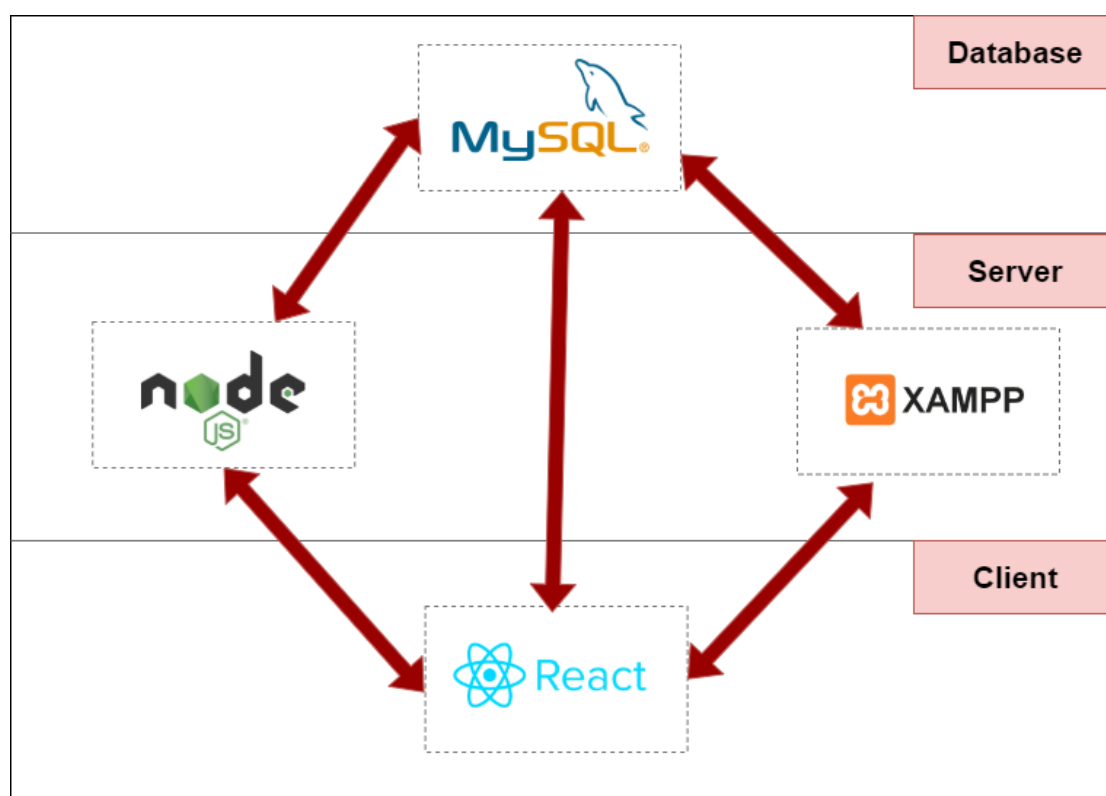
การวิเคราะห์และออกแบบระบบ การจองควิร้านเสริมสวย ในบทนี้จะแบ่งออกเป็น 6 ขั้นตอนเพื่อให้เห็นการดำเนินงานอย่างมีระบบ ในหัวข้อแรกจะนำเสนอภาพรวมของระบบ ก่อนจะนำเสนอเอกสารแสดงความต้องการของระบบซึ่งจะทำให้เห็นที่มาของเพจต่าง ๆ ในขั้นตอนของการออกแบบในหัวข้อที่สาม ส่วนหัวข้อที่เหลือจะแสดงแผนภาพการทำงานของระบบโดยใช้ UML diagram ซึ่งประกอบไปด้วย Use Case, Class และ Sequence Diagram เพื่อแสดงรายละเอียดของระบบก่อนนำไปเขียนคำสั่งด้วยภาษาโปรแกรมในบทต่อไป

- 3.1 โครงสร้างภาพรวมของระบบ (System Architecture) เป็นการออกแบบภาพรวมและเทคโนโลยีของระบบ
- 3.2 System Requirements คือ ความต้องการหรือสิ่งที่ระบบควรจะทำ หรือหน้าที่หลักของระบบที่จะต้องทำ
- 3.3 User Interface Design เป็นการออกแบบส่วนต่อประสานกับผู้ใช้
- 3.4 Use Case Diagram เป็นแผนภาพที่ใช้แสดงให้ทราบว่าระบบทำงานหรือมีหน้าที่ใดบ้าง
- 3.5 Class Diagram เป็นแผนภาพที่ใช้แสดง Class และความสัมพันธ์ระหว่าง Class
- 3.6 Sequence Diagram เป็นแผนภาพที่ใช้แสดงให้เห็นถึงการตอบโต้ข้อมูลระหว่างคลาส เรียงตามลำดับของเวลาที่เกิดเหตุการณ์จากน้อยไปมาก

### 3.1 โครงสร้างภาพรวมของระบบ

ความหมายของ System Architecture [1] หมายถึง กรอบโครงสร้างของระบบที่อธิบายความสัมพันธ์ขององค์ประกอบต่าง ๆ ไปจนถึงขั้นการเชื่อมต่อกันของระบบย่อยต่าง ๆ โดยจัดกลุ่มองค์ประกอบไว้ในหลาย ๆ ลักษณะเพื่อให้ผู้เกี่ยวข้อง (Stakeholder) จากพื้นฐานสาขาอาชีพที่แตกต่างกันสามารถทำความเข้าใจได้ง่าย เช่น การจัดแบ่งองค์ประกอบตามลักษณะการทำงานของระบบ (functional components) เป็นต้น

การออกแบบ System architecture แสดงภาพรวมและเทคโนโลยีของระบบกองทุนเงินให้กู้ยืมเพื่อการศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี มีรายละเอียดดังรูปที่ 3.1



รูปที่ 3.1: System architecture ระบบการจองคิวร้านเสริมสวย

จากรูปที่ 3.1 สามารถอธิบายโครงสร้างและเทคโนโลยีของระบบโดยแบ่งเป็น 3 ส่วนหลักดังนี้

1. Database ระบบให้บริการฐานข้อมูลแบบ MySQL ของไฟร์เบสชื่อ Relational Database Management System : RDBMS

2. Server กระบวนการทำงานในส่วนของเซิร์ฟเวอร์ (server) แบ่งเป็น 2 ส่วนได้แก่
  - ชุดเผยแพร่สำหรับการพัฒนาเว็บไซต์ซึ่งในที่นี้ใช้ Node.js และ Express ในการพัฒนา
  - ชุดบริการ phpmyadmin Api ใช้สำหรับการทำงานกับบริการต่าง ๆ ของ MySQL
3. Client เว็บแอปพลิเคชันการจองคิวร้านเสริมสวย มีวัตถุประสงค์เพื่อรองรับการทำงานของ  
ผู้ใช้งานเว็บเบราว์เซอร์พัฒนาด้วย React

## 3.2 System Requirements

### 3.2.1 Functional Requirements

ระบบการจองคิวร้านเสริมสวย แบ่งความสามารถของระบบตามประเภทของผู้ใช้งานดังนี้

#### 1. เจ้าของร้าน

- สามารถลงทะเบียนเข้าสู่ระบบด้วย Email ได้
- สามารถดูคิวที่ผู้ใช้บริการได้ทำการจองคิวไว้
- สามารถจัดการคิวได้
- สามารถ post ภาพผลงานทั้งหมดของร้านได้
- สามารถเพิ่ม แก้ไข และลบรายการให้บริการประจำร้านได้
- สามารถเพิ่ม แก้ไข และลบ ข้อมูลร้านได้
- สามารถเพิ่ม แก้ไข และลบข้อมูลตำแหน่งร้านได้

#### 2. ช่างประจำร้าน

- ลงทะเบียนใช้ web ด้วย Email ได้
- สามารถดูตารางการทำงานของตนเองได้
- สามารถ post ภาพผลงานของตัวเองได้
- สามารถแก้ไขข้อมูลส่วนตัวได้

#### 3. ผู้ใช้บริการ

- สามารถลงทะเบียนเข้าสู่ระบบด้วย Email ได้
- สามารถค้นหาร้านเสริมสวยได้
- สามารถจองคิวของร้านเสริมสวยได้
- สามารถดูคิวว่างของร้านเสริมสวยได้

- สามารถดูข้อมูลต่างๆของร้านเสริมสวยได้
- สามารถดูตำแหน่งของทางร้านได้
- สามารถดูผลงานของร้านได้
- สามารถเขียนรีวิว ตีชม ได้

### 3.2.2 Non-functional Requirements

#### 1. เว็บแอปพลิเคชัน

- ใช้โปรโตคอล (Protocol) แบบ HTTPS (Hypertext Transfer Protocol Secure) ในการสื่อสารที่ช่วยรักษาความสมบูรณ์ถูกต้องของข้อมูลผู้ใช้และเก็บข้อมูลไว้เป็นความลับระหว่างคอมพิวเตอร์ของผู้ใช้กับเว็บไซต์
- รองรับการใช้งานบนเว็บเบราว์เซอร์และสมาร์ทโฟน

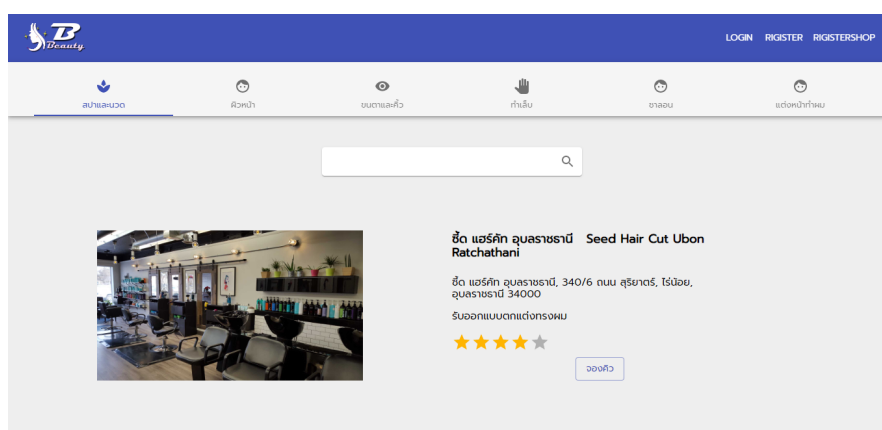
### 3.3 User Interface Design

User Interface Design ของระบบการจองคิวร้านเสริมสวย แบ่งเป็น 3 ส่วน มีดังนี้

#### 1. ผู้ใช้บริการ

User Interface ผู้ใช้บริการ จะออกแบบให้สะดวกต่อการใช้งาน

- การออกแบบหน้าจอหลัก

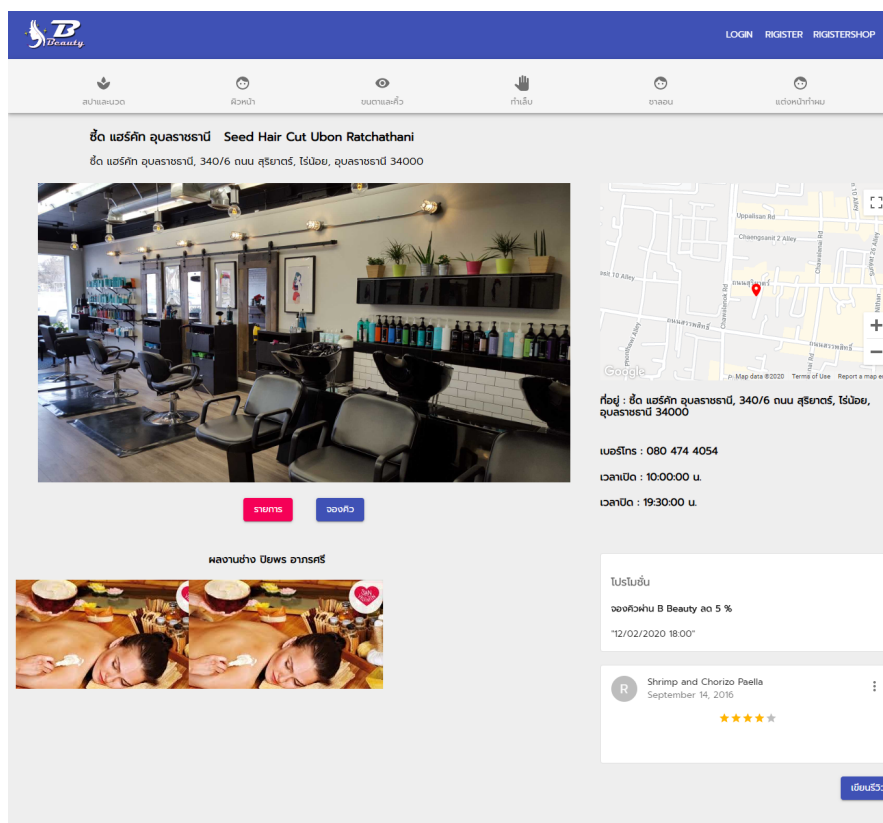


รูปที่ 3.2: หน้าจอหลัก

จากภาพที่ 4.6 แสดงหน้าจอหลักบนเว็บแอปพลิเคชัน เพื่ออำนวยความสะดวกต่อผู้ใช้งาน ในหน้าหลักได้รวบรวมข้อมูลสรุปของร้านและเมนูเข้าถึงด่วนซึ่งแบ่งเป็น 3

ส่วนหลักได้แก่ เมนูจองคิว รายละเอียดร้านและข้อมูลรีวิว

- การออกแบบหน้าจอข้อมูลร้าน

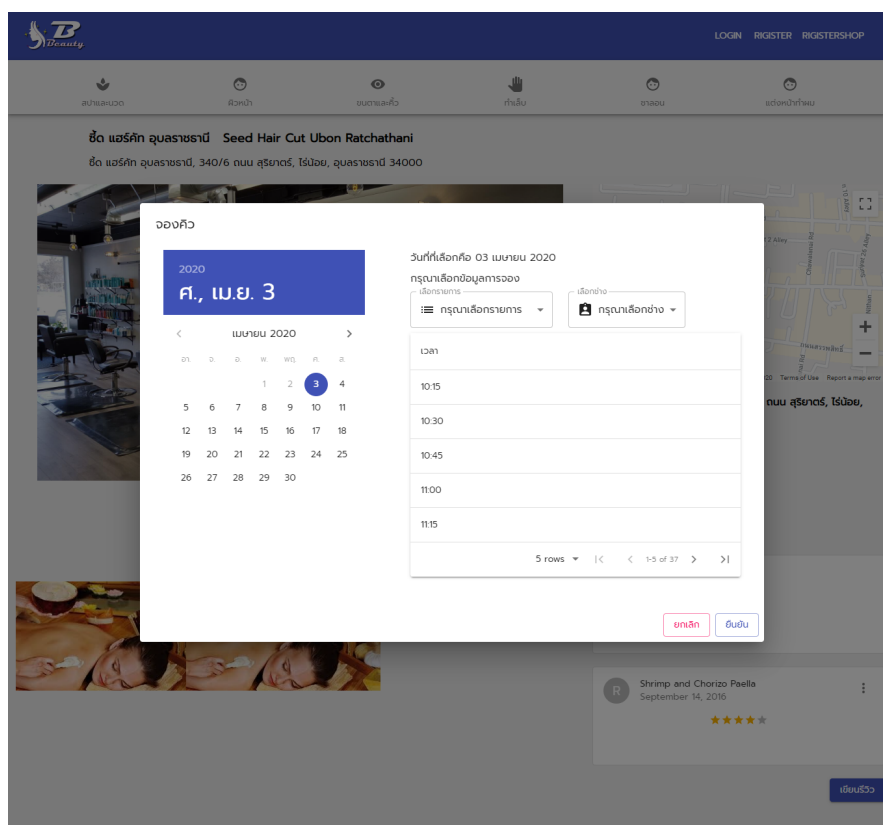


รูปที่ 3.3: หน้าจอข้อมูลร้าน

จากภาพที่ 3.3 แสดงหน้าจอข้อมูลรายละเอียดภายในร้าน



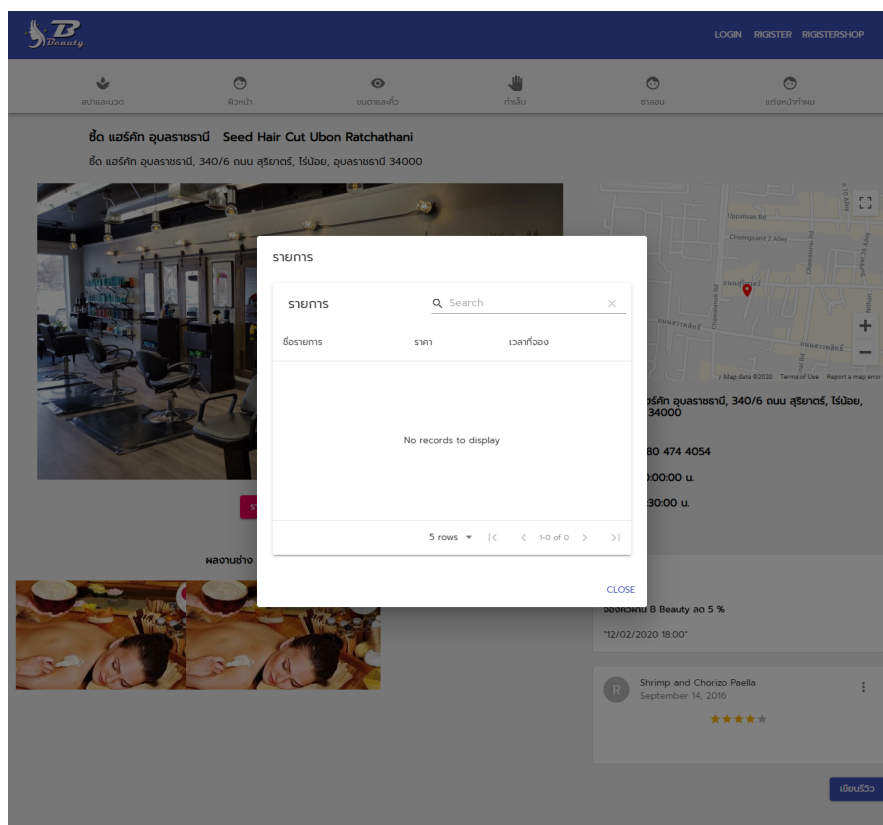
- การออกแบบหน้าจอการจองคิว



รูปที่ 3.4: หน้าจอการจองคิว

จากภาพที่ 3.4 แสดงหน้าจอการจองคิวร้านเสริมสวย ทั้งนี้ผู้ที่มิสิทธิ์ในการจองคิวมีเพียงผู้ใช้งานเท่านั้น

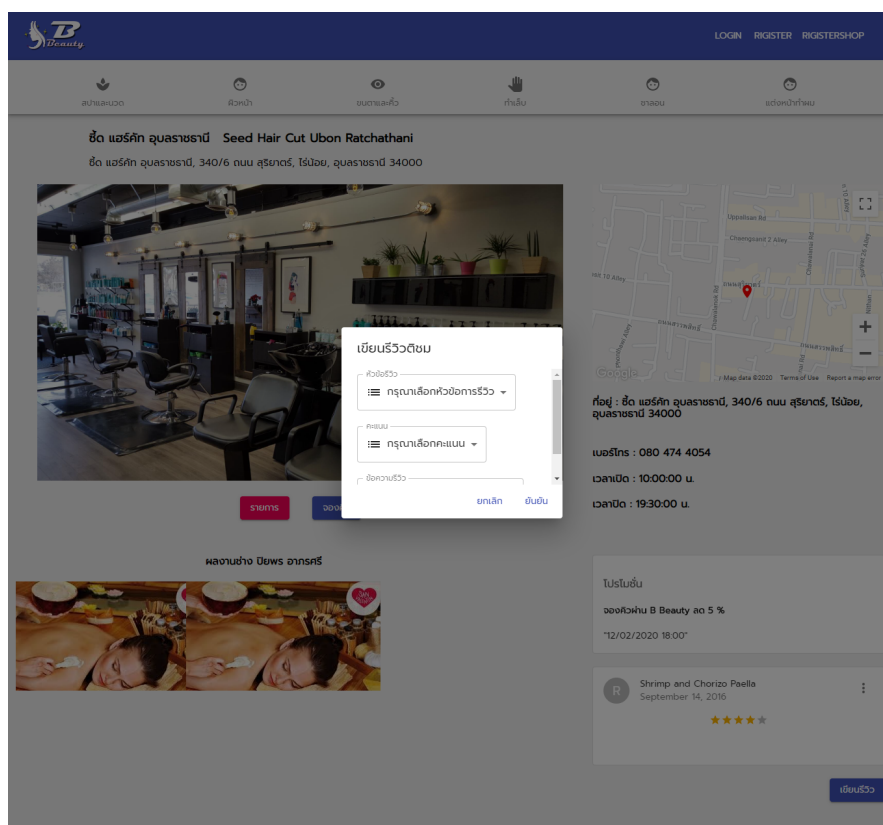
- การออกแบบหน้าจอรายการ



รูปที่ 3.5: หน้าจอรายการ

จากภาพที่ 3.5 แสดงหน้าจอรายการที่ให้บริการประจำร้าน

- การออกแบบหน้าจอเขียนรีวิว



รูปที่ 3.6: หน้าจอเขียนรีวิว

จากภาพที่ 3.6 แสดงหน้าจอการเขียนรีวิว ทั้งนี้ผู้ที่มิสิทธิ์ในการอัปโหลดเอกสารมีเพียงผู้ใช้งานเท่านั้น

- การออกแบบหน้าจอลงทะเบียน

รูปที่ 3.7: หน้าจอลงทะเบียน

จากภาพที่ 3.7 แสดงหน้าจอการลงทะเบียนของผู้ใช้บริการ เพื่อใช้ในการเข้าสู่ระบบ

- การออกแบบหน้าจอเข้าสู่ระบบ

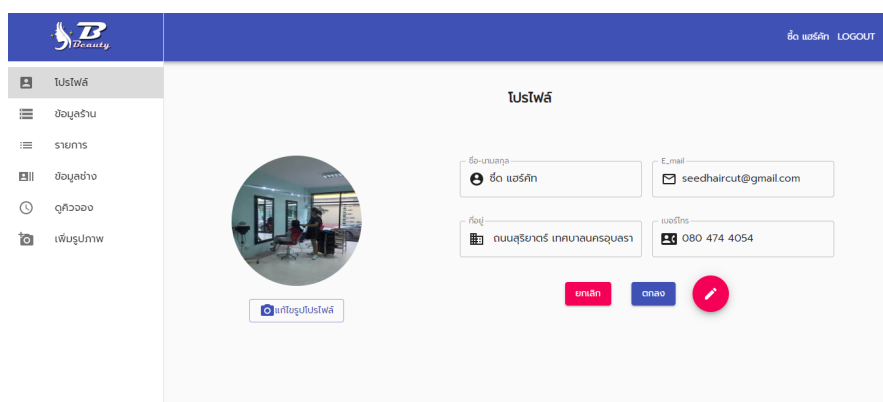
รูปที่ 3.8: หน้าจอเข้าสู่ระบบ

จากภาพที่ 3.8 แสดงหน้าจอการเข้าสู่ระบบของผู้ใช้โดยผู้ใช้จำเป็นต้องกรอกข้อมูลอีเมลและรหัสผ่านเพื่อเข้าใช้งานระบบ

## 2. เจ้าของร้าน

User Interface เจ้าของร้าน จะออกแบบให้สะดวกต่อการใช้งาน

- การออกแบบหน้าจอโปรไฟล์ร้าน



รูปที่ 3.9: หน้าจอโปรไฟล์ร้าน

จากภาพที่ 3.9 แสดงหน้าจอโปรไฟล์บนเว็บแอปพลิเคชัน เพื่อดูข้อมูลของเจ้าของร้าน

- การออกแบบหน้าจอเพิ่มข้อมูลร้าน

ข้อมูลร้าน

ข้อมูลทั่วไป

เพิ่มรูปภาพร้าน

เพิ่มรูปภาพร้าน

ชื่อร้าน

Name

เวลาเปิดร้าน

เวลาปิดร้าน

เบอร์โทรร้าน

ที่อยู่ร้าน

รายละเอียดร้าน

lat

long

Facebook

ประเภทของร้าน

กรุณาเลือกประเภท

ยกเลิก

ตกลง

ข้อมูลโปรโมชัน

ชื่อโปรโมชัน

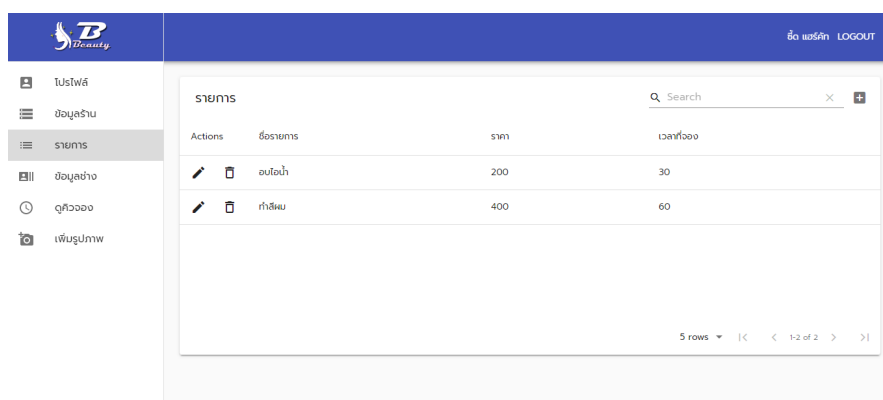
ยกเลิก

ตกลง

รูปที่ 3.10: หน้าจอเพิ่มข้อมูลร้าน

จากภาพที่ 3.10 แสดงหน้าจอเพิ่มข้อมูลร้าน เพื่อให้เจ้าของร้านเพิ่มข้อมูลร้าน

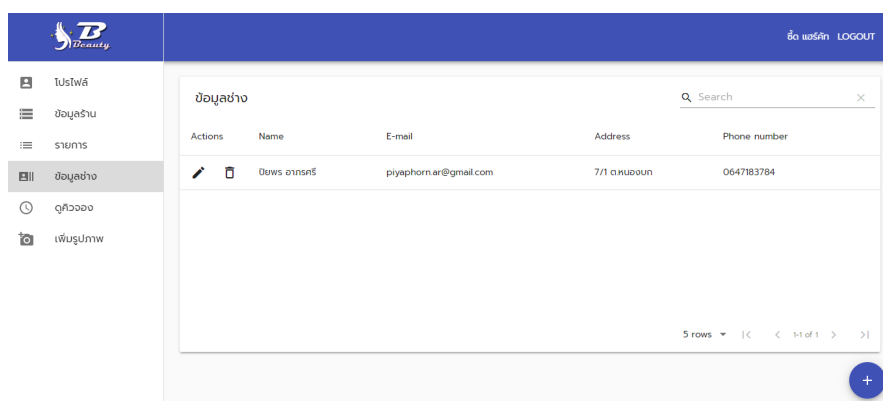
- การออกแบบหน้าจอเพิ่มรายการ



รูปที่ 3.11: หน้าจอเพิ่มรายการ

จากภาพที่ 3.11 แสดงหน้าจอเพิ่มข้อมูลรายการของร้าน เพื่อให้เจ้าของร้านเพิ่มข้อมูลรายการที่ให้บริการประจำร้าน

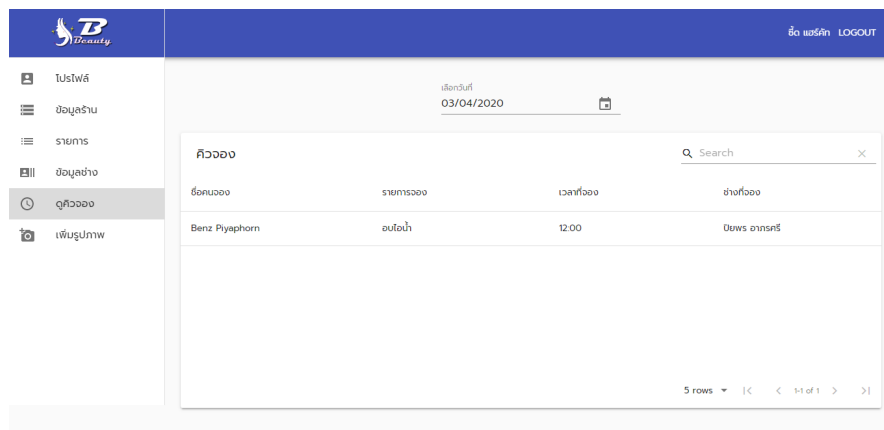
- การออกแบบหน้าจอเพิ่มข้อมูลช่าง



รูปที่ 3.12: หน้าจอเพิ่มข้อมูลช่าง

จากภาพที่ 3.12 แสดงหน้าจอข้อมูลช่าง เพื่อเพิ่มและแสดงข้อมูลช่างประจำร้าน

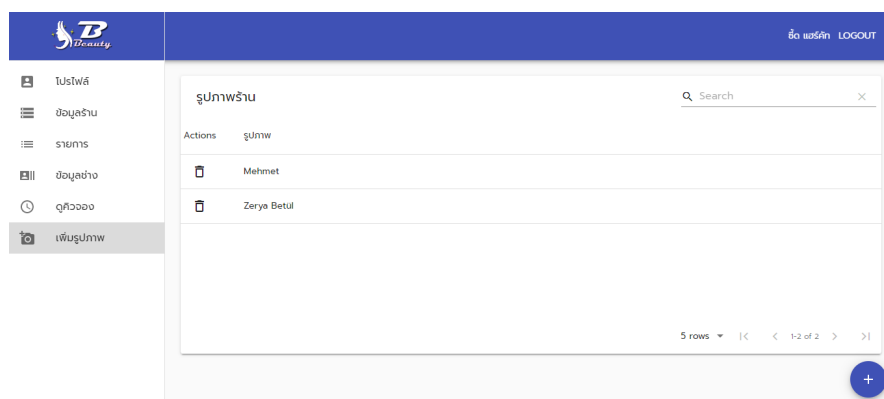
- การออกแบบหน้าจอจัดการจองคิว



รูปที่ 3.13: หน้าจอจัดการจองคิว

จากภาพที่ 3.13 แสดงหน้าจอการจองคิว เพื่อให้เจ้าของร้านดูข้อมูลการจองคิวจากผู้ใช้งาน

- การออกแบบหน้าจอเพิ่มรูปภาพ



รูปที่ 3.14: หน้าจอเพิ่มรูปภาพ

จากภาพที่ 3.14 แสดงหน้าจอเพิ่มรูปภาพ เพื่ออัปโหลดรูปภาพเฉพาะเจ้าของร้าน



### 3. เจ้าของร้าน

User Interface เจ้าของร้าน จะออกแบบให้สะดวกต่อการใช้งาน

- การออกแบบหน้าจอโปรไฟล์ช่าง

รูปที่ 3.15: หน้าจอโปรไฟล์ช่าง

จากภาพที่ 3.15 แสดงหน้าจอโปรไฟล์บนเว็บแอปพลิเคชัน เพื่อดูข้อมูลของช่าง

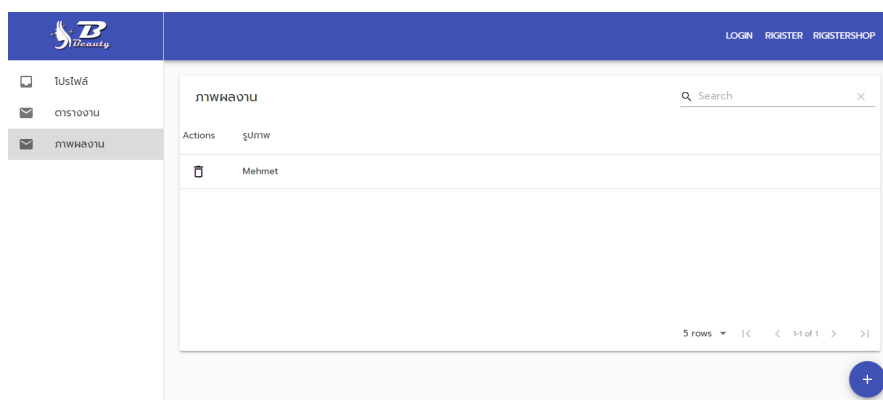
- การออกแบบหน้าจอดูตารางงาน

ชื่อคนจอง	รายการจอง	เวลาที่จอง	ช่างที่จอง
Benz Piyaphorn	อนอเน่	12:00	อิวพร อภรณ์

รูปที่ 3.16: หน้าจอดูตารางงาน

จากภาพที่ 3.16 แสดงหน้าจอตารางงาน เฉพาะช่าง

- การออกแบบหน้าจอเพิ่มรูปภาพผลงานช่าง





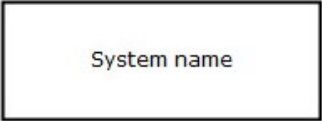
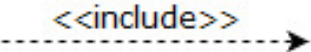
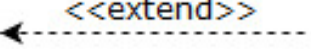
รูปที่ 3.17: หน้าจอเพิ่มรูปภาพผลงานช่าง

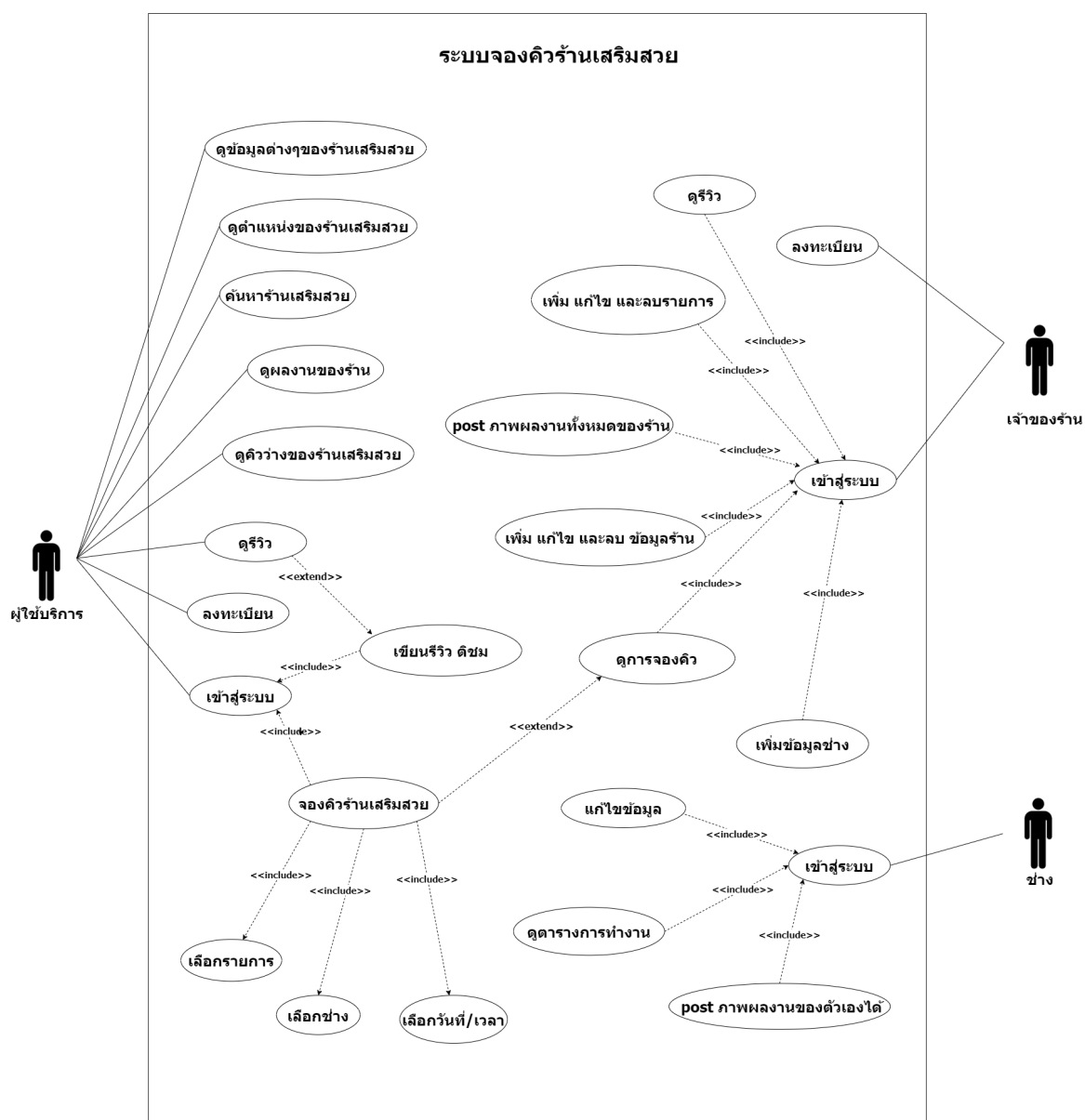
จากภาพที่ 3.17 แสดงหน้าจอเพิ่มรูปภาพ เพื่ออัปโหลดรูปภาพผลงานเฉพาะช่าง

### 3.4 Use Case Diagram

Use Case Diagram เป็นแผนผังเพื่อแสดงฟังก์ชันการทำงานของระบบโดยรวม แสดงส่วนประกอบในระบบและกิจกรรมที่เกิดขึ้นในระบบซึ่งระบบของครัวเรือนเสริมสวย ผู้ใช้จำเป็นต้องเข้าสู่ระบบเพื่อใช้งานระบบ 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 คือบุคคลหรือระบบงานอื่นที่ใช้งานระบบหรือได้รับประโยชน์จากระบบซึ่งอยู่ภายนอกระบบ แทนด้วยรูปคนและมีชื่อบทบาทการใช้งานระบบ
	เส้นตรงที่แสดงถึงการใช้งาน Use case ของผู้กระทำ
	กรอบสี่เหลี่ยม แสดง ถึง ขอบเขต ของ ระบบ โดย แสดง ชื่อ ระบบ ภายในหรือด้านบนกรอบสี่เหลี่ยม Use case อยู่ภายในกรอบสี่เหลี่ยม และ actor อยู่ภายนอกกรอบสี่เหลี่ยม
	ความสัมพันธ์แบบ «includes» แสดงว่า Use case หนึ่งดำเนินการตามขั้นตอนของ Use case อื่น โดยแทนด้วยสัญลักษณ์ลูกศรเส้นประ ซึ่ง Use case ที่หางลูกศรเรียกใช้งาน Use case ที่หัวลูกศรทุกครั้งที่มีการทำงาน
	ความสัมพันธ์แบบ «extend» แสดงว่า Use case หนึ่งดำเนินการตามขั้นตอนของ Use case อื่น โดยแทนด้วยสัญลักษณ์ลูกศรเส้นประ ซึ่ง Use case ที่หัวลูกศรเรียกใช้งาน Use case ที่หางลูกศร แต่การใช้งานไม่จำเป็นต้องเกิดขึ้นทุกครั้งขึ้นอยู่กับเงื่อนไขระหว่างการทำงาน



รูปที่ 3.18: Use Case Diagram ของระบบ จองคิวร้านเสริมสวย

ตารางที่ 3.2: อธิบาย Use Case หน้าที่ของระบบ ในภาพที่ 3.18

Use Case	คำอธิบาย
ดูข้อมูลต่างๆของร้านเสริมสวย	ผู้ใช้บริการสามารถดูข้อมูลได้โดยไม่จำเป็นต้องทำการเข้าสู่ระบบก่อน
ดูตำแหน่งของร้านเสริมสวย	ผู้ใช้บริการสามารถดูตำแหน่งร้านได้ โดยไม่จำเป็นต้องทำการเข้าสู่ระบบก่อน
ค้นหาร้านเสริมสวย	ผู้ใช้งานทั่วไปสามารถค้นหาร้านได้โดยไม่จำเป็นต้องทำการเข้าสู่ระบบก่อน
ดูผลงานของร้านเสริมสวย	ผู้ใช้บริการสามารถดูผลงานของร้านเสริมสวยได้โดยไม่ต้องทำการเข้าสู่ระบบก่อน
ดูคิวว่างของร้านเสริมสวย	ผู้ใช้บริการสามารถดูคิวว่างของร้านเสริมสวยได้โดยไม่ต้องทำการเข้าสู่ระบบก่อน
ดูรีวิว	ผู้ใช้บริการสามารถดูรีวิวได้โดยไม่จำเป็นต้องเข้าสู่ระบบก่อน
เขียนรีวิวดิชม	ผู้ใช้บริการสามารถเขียนรีวิวดิชมได้โดยจำเป็นต้องเข้าสู่ระบบก่อน
จองคิวร้านเสริมสวย	ผู้ใช้บริการสามารถจองคิวร้านเสริมสวยได้โดยจำเป็นต้องเข้าสู่ระบบก่อน และจำเป็นต้อง เลือก วันที่ ราย-การ ช่าง เวลา
เพิ่ม แก้ไข และลบรายการ	ใช้เจ้าของร้านเพื่อ เพิ่ม แก้ไขหรือลบข้อมูลรายการ
post ภาพผลงานทั้งหมดของร้านเสริมสวย	ใช้สำหรับเจ้าของร้านเพื่อ เพิ่ม post รูปภาพผลงานของร้าน
เพิ่มแก้ไขและลบข้อมูลร้าน	ใช้สำหรับเจ้าของร้านเพื่อ เพิ่ม แก้ไขหรือลบข้อมูลร้าน

ตารางที่ 3.3: อธิบาย Use Case หน้าที่ของระบบ(ต่อ) ในภาพที่ 3.18

Use Case	คำอธิบาย
ดูการจองคิว	ใช้สำหรับเจ้าของร้านเพื่อ ดูการจองที่ผู้ใช้บริการได้ทำการจองคิว
เพิ่มข้อมูลช่าง	ใช้สำหรับเจ้าของร้านเพื่อ เพิ่ม ข้อมูลช่าง
แก้ไขข้อมูล	ใช้สำหรับช่างเพื่อ แก้ไขข้อมูลส่วนตัว
ดูตารางงาน	ใช้สำหรับช่างเพื่อ ดูตารางงาน
post ภาพผลงานของตัวเอง	ใช้สำหรับช่างเพื่อ post ภาพผลงาน

ตารางที่ 3.4: Use Case ดูข้อมูลต่างๆของร้านเสริมสวย

Use Case Title : ดูข้อมูลต่างๆของร้านเสริมสวย	Use case Id : 1
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : เจ้าของร้าน	
Main Flow : ผู้ใช้บริการดูข้อมูลต่างๆของร้านเสริมสวยโดยไม่จำเป็นต้องเข้าสู่ระบบ	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถดูข้อมูลต่างๆ ของร้านเสริมสวยได้	

ตารางที่ 3.5: Use Case ดูตำแหน่งของร้านเสริมสวย

Use Case Title : ดูตำแหน่งของร้านเสริมสวย	Use case Id : 2
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : เจ้าของร้าน	
Main Flow : ผู้ใช้บริการสามารถดูตำแหน่งร้านเสริมสวย ไม่จำเป็นต้องเข้าสู่ระบบ	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถดูตำแหน่งของร้านเสริมสวยได้	

ตารางที่ 3.6: Use Case ค้นหาร้านเสริมสวย

Use Case Title : ค้นหาร้านเสริมสวย	Use case Id : 3
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : เจ้าของร้าน	
Main Flow : ผู้ใช้บริการสามารถค้นหาร้านเสริมสวย ไม่จำเป็นต้องเข้าสู่ระบบ	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถค้นหาร้านเสริมสวยได้	

ตารางที่ 3.7: Use Case ดูผลงานของร้านเสริมสวย

Use Case Title : ดูผลงานของร้านเสริมสวย	Use case Id : 4
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : เจ้าของร้าน	
Main Flow : ดูผลงานของร้านเสริมสวย โดยไม่จำเป็นต้องเข้าสู่ระบบ	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถดูผลงานของร้านเสริมสวยได้	

ตารางที่ 3.8: Use Case ดูคิวว่างของร้านเสริมสวย

Use Case Title : ดูคิวว่างของร้านเสริมสวย	Use case Id : 5
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : เจ้าของร้าน	
Main Flow : ดูคิวว่างของร้านเสริมสวยได้ โดยไม่จำเป็นต้องเข้าสู่ระบบ	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถดูคิวว่างของร้านเสริมสวยได้	

ตารางที่ 3.9: Use Case ดูรีวิว

Use Case Title : ดูรีวิว	Use case Id : 6
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : เจ้าของร้าน	
Main Flow : ดูรีวิวได้โดยไม่ต้องเข้าสู่ระบบ	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถดูรีวิวได้	

ตารางที่ 3.10: Use Case เขียนรีวิว

Use Case Title : เขียนรีวิว	Use case Id : 7
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : -	
Main Flow : เขียนรีวิวได้ต้องเข้าสู่ระบบ	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถเขียนรีวิวได้	

ตารางที่ 3.11: Use Case จองคิวร้านเสริมสวย

Use Case Title : จองคิวร้านเสริมสวย	Use case Id : 10
Primary Actor : ผู้ใช้บริการ	
Stakeholder Actor : เจ้าของร้าน,ช่าง	
Main Flow : ผู้ใช้บริการทำการจองคิวร้านเสริมสวย ได้โดยจำเป็นต้องเข้าสู่ระบบ และจำเป็นต้องเลือก วันที่ รายการ ช่าง เวลา	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถ จองคิวร้านเสริมสวยได้	



ตารางที่ 3.13: Use Case post ภาพผลงานทั้งหมดของร้าน

Use Case Title : post ภาพผลงานทั้งหมดของร้าน	Use case Id : 12
Primary Actor : เจ้าของร้าน	
Stakeholder Actor : -	
Main Flow : เจ้าของร้าน post ภาพผลงานทั้งหมดของร้านได้	
Exceptional Flow ที่ 1 : หากเจ้าหน้าที่ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถ post ภาพผลงานทั้งหมดของร้านได้	

ตารางที่ 3.14: Use Case เพิ่มแก้ไขและลบข้อมูลร้าน

Use Case Title : เพิ่มแก้ไขและลบข้อมูลร้าน	Use case Id : 13
Primary Actor :เจ้าของร้าน	
Stakeholder Actor : -	
Main Flow : เจ้าของร้าน เพิ่ม แก้ไขและลบข้อมูลร้าน	
Exceptional Flow ที่ 1 : หากเจ้าหน้าที่ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถ เพิ่ม แก้ไขและลบข้อมูลร้านได้	

ตารางที่ 3.15: Use Case ดูการจองคิว

Use Case Title : ดูการจองคิว	Use case Id : 14
Primary Actor :เจ้าของร้าน	
Stakeholder Actor : -	
Main Flow : ดูการจองคิวที่ผู้ใช้บริการได้ทำการจองคิว โดยผู้ใช้บริการ	
Exceptional Flow ที่ 1 : หากเจ้าของร้านที่ไม่เชื่อมต่ออินเทอร์เน็ต ไม่สามารถดูการจองคิวที่ผู้ใช้บริการได้ทำการจองคิว โดยผู้ใช้บริการได้	

ตารางที่ 3.16: Use Case เพิ่มข้อมูลช่าง

Use Case Title : เพิ่มข้อมูลช่าง	Use case Id : 15
Primary Actor :เจ้าของร้าน	
Stakeholder Actor : -	
Main Flow : เจ้าของร้านเพิ่มข้อมูลช่าง	
Exceptional Flow ที่ 1 : หากเจ้าของร้านที่ไม่เชื่อมต่ออินเทอร์เน็ต ไม่สามารถเพิ่มข้อมูลช่างได้	

ตารางที่ 3.17: Use Case แก้ไขข้อมูล

Use Case Title : แก้ไขข้อมูล	Use case Id : 16
Primary Actor :ช่าง	
Stakeholder Actor : -	
Main Flow : ช่างแก้ไขข้อมูลส่วนตัว	
Exceptional Flow ที่ 1 : หากช่างที่ไม่เชื่อมต่ออินเทอร์เน็ต ไม่สามารถแก้ไขข้อมูลส่วนตัวได้	

ตารางที่ 3.18: Use Case ดูตารางงาน

Use Case Title : ดูตารางงาน	Use case Id : 17
Primary Actor :ช่าง	
Stakeholder Actor : -	
Main Flow : ดูตารางงาน ที่ผู้ใช้บริการได้ทำการจองคิว	
Exceptional Flow ที่ 1 : หากช่างที่ไม่เชื่อมต่ออินเทอร์เน็ต ไม่สามารถดูตารางงานได้	



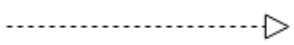
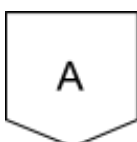
ตารางที่ 3.19: Use Case post ภาพผลงานของตัวเอง

Use Case Title : post ภาพผลงานของตัวเอง	Use case Id : 18
Primary Actor :ช่าง	
Stakeholder Actor : -	
Main Flow : post ภาพผลงานของตัวเองได้	
Exceptional Flow ที่ 1 : หากช่างที่ไม่เชื่อมต่ออินเทอร์เน็ต ไม่สามารถpost ภาพผลงานของตัวเองได้	

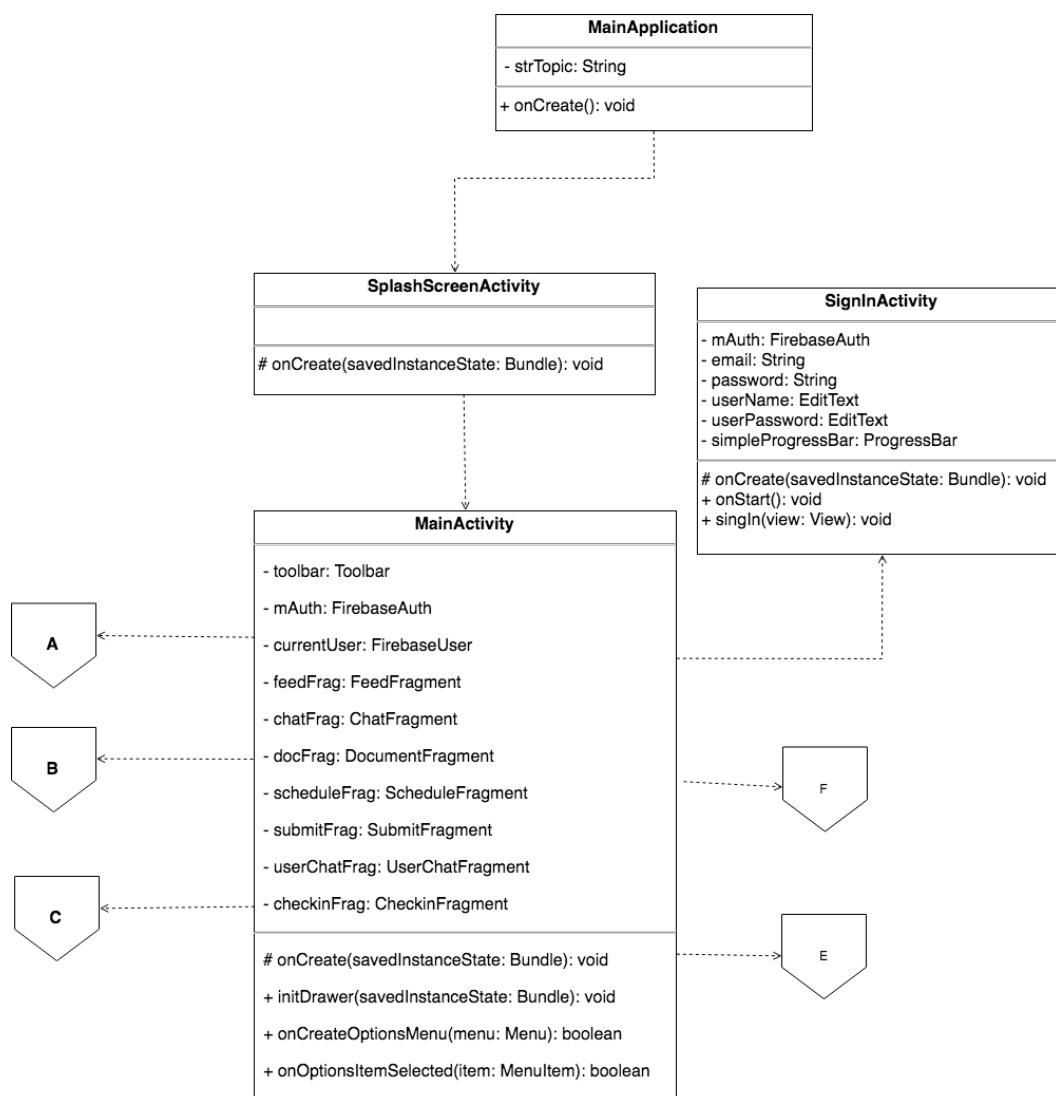
### 3.5 Class Diagram

Class Diagram คือแผนภาพที่ใช้แสดงคลาสและความสัมพันธ์ในแบบต่างๆ ระหว่างคลาส สัญลักษณ์ที่ใช้ในการเขียน Class Diagram แสดงในตารางที่ 3.20

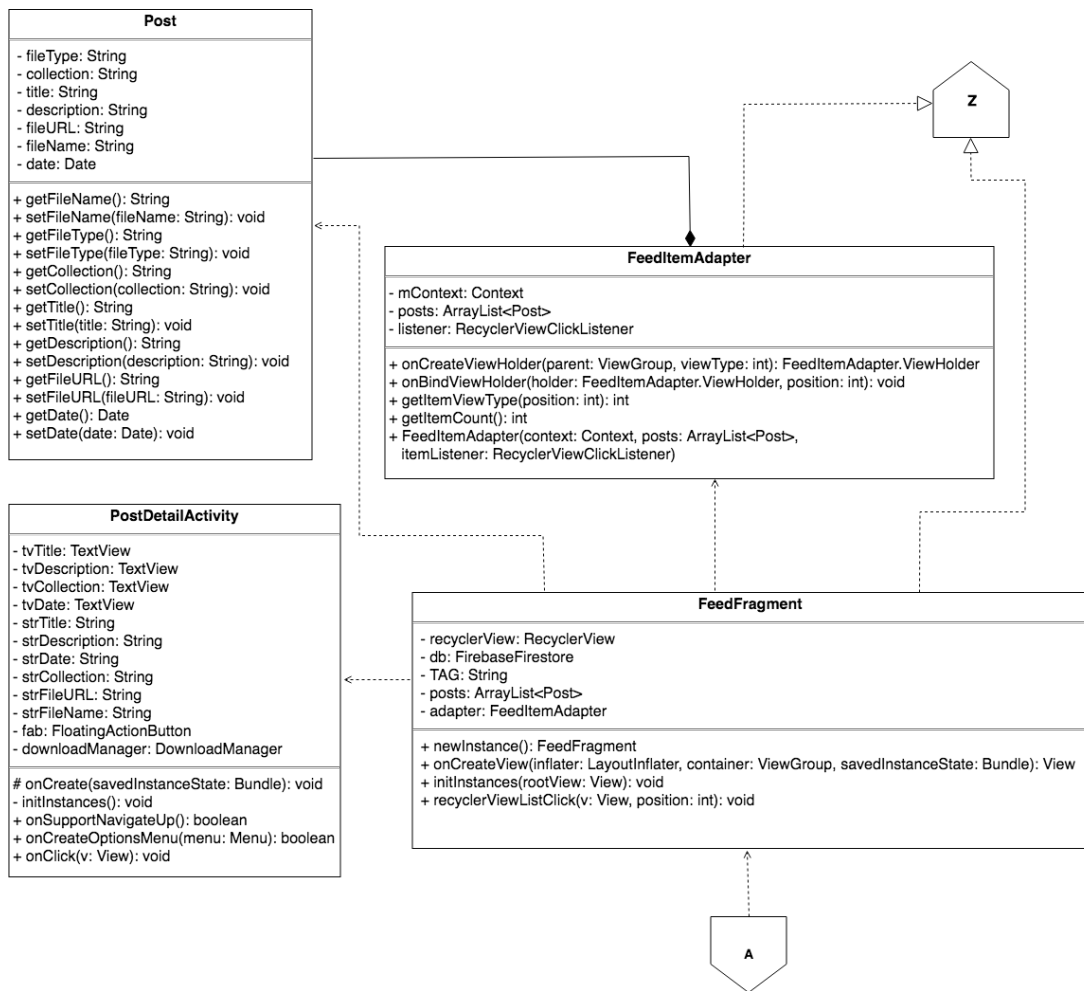
ตารางที่ 3.20: สัญลักษณ์ของ Class Diagram

สัญลักษณ์	การใช้งาน
<div style="border: 1px solid black; padding: 5px; margin: 10px auto; width: fit-content;"> <div style="border-bottom: 1px solid black; padding-bottom: 5px; text-align: center;">Class Name</div> <div style="border-bottom: 1px solid black; padding: 5px 0 5px 20px; text-align: center;">Attribute Name</div> <div style="padding: 5px 0 5px 20px; text-align: center;">Operation Name()</div> </div>	<p>คลาส สัญลักษณ์แทนด้วยสี่เหลี่ยมแบ่งเป็น 3 ส่วน ส่วนบน เป็นชื่อของ class ส่วนกลาง เป็นชื่อ Attribute และส่วนล่างเป็น Operation Name หรือ Method ใช้สำหรับเขียนฟังก์ชันในการทำงานของคลาสนั้น ๆ ชนิดของ Visibility ของ Method และ Attribute แบ่งเป็น 3 ชนิด ได้แก่</p> <ol style="list-style-type: none"> <li>1. Public แทนสัญลักษณ์ด้วยเครื่องหมายบวก (+)</li> <li>2. Private แทนสัญลักษณ์ด้วยเครื่องหมายลบ (-)</li> <li>3. Protected แทนสัญลักษณ์ด้วยเครื่องหมายชาร์ป ( )</li> </ol>
	Dependency Relationship หมายความว่า คลาสที่อยู่ฝั่งต้นลูกศรสามารถเรียกใช้คลาสที่อยู่ฝั่งหัวลูกศร
	Composition Relationship เป็นความสัมพันธ์ระหว่างออบเจกต์หรือคลาสแบบขึ้นต่อกันและมีความเกี่ยวข้องกันเสมอ
	Realization Relationship เป็นความสัมพันธ์ระหว่าง Object หรือ Class ในลักษณะของการสืบทอดคุณสมบัติจาก Class หนึ่ง (Super class) ไปยังอีก Class หนึ่ง (Subclass)
	Connector เป็นสัญลักษณ์แทนด้วยรูปห้าเหลี่ยมและมีชื่ออยู่ตรงกลาง จะสร้างสัญลักษณ์นี้ไว้เมื่อต้องการเชื่อมต่อคลาสที่อยู่คนละหน้า

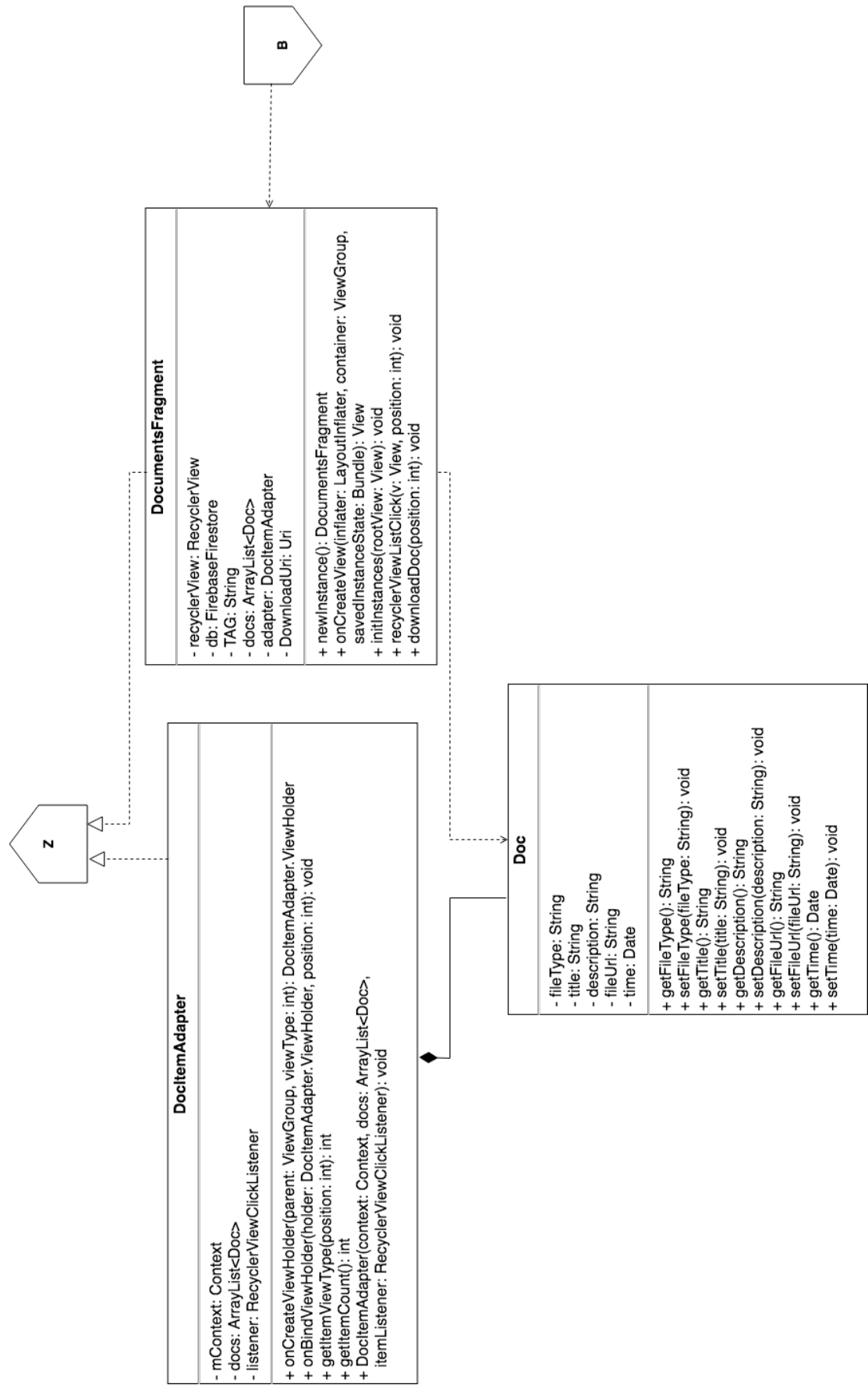
Class Diagram แสดงความสัมพันธ์ในรูปแบบต่างๆ ระหว่างคลาสของแอปพลิเคชันระบบ กองทุนเงินให้กู้ยืมเพื่อการศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี อธิบายได้ตามภาพ ที่ 3.19 ดังต่อไปนี้



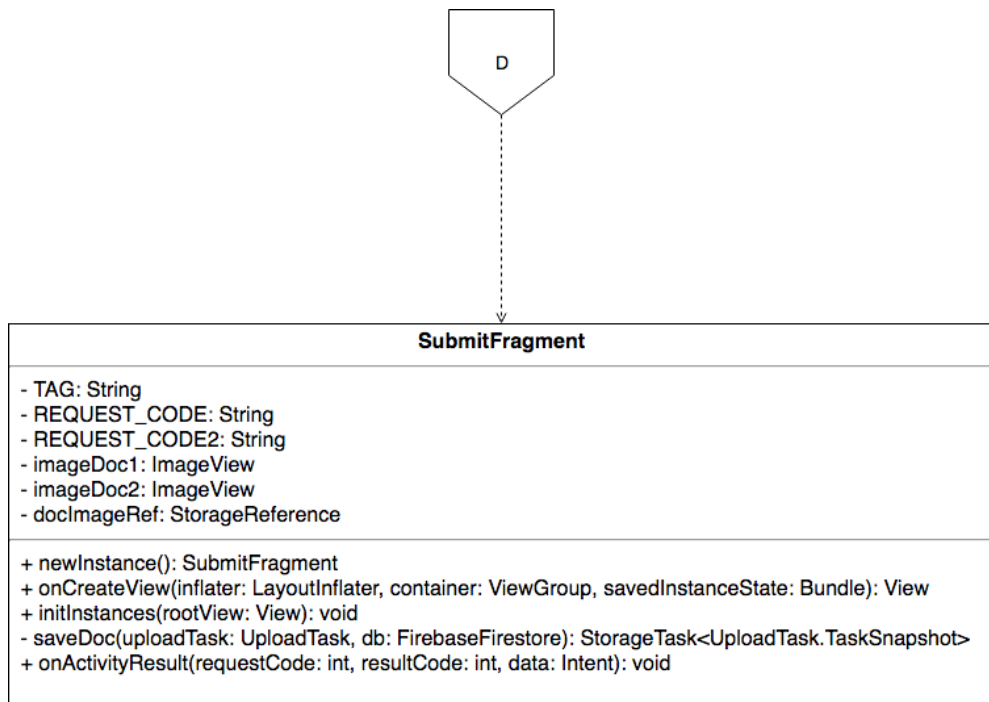
รูปที่ 3.19: Class Diagram ของแอปพลิเคชันระบบ XX



รูปที่ 3.20: Class Diagram ของแอปพลิเคชันระบบ XX

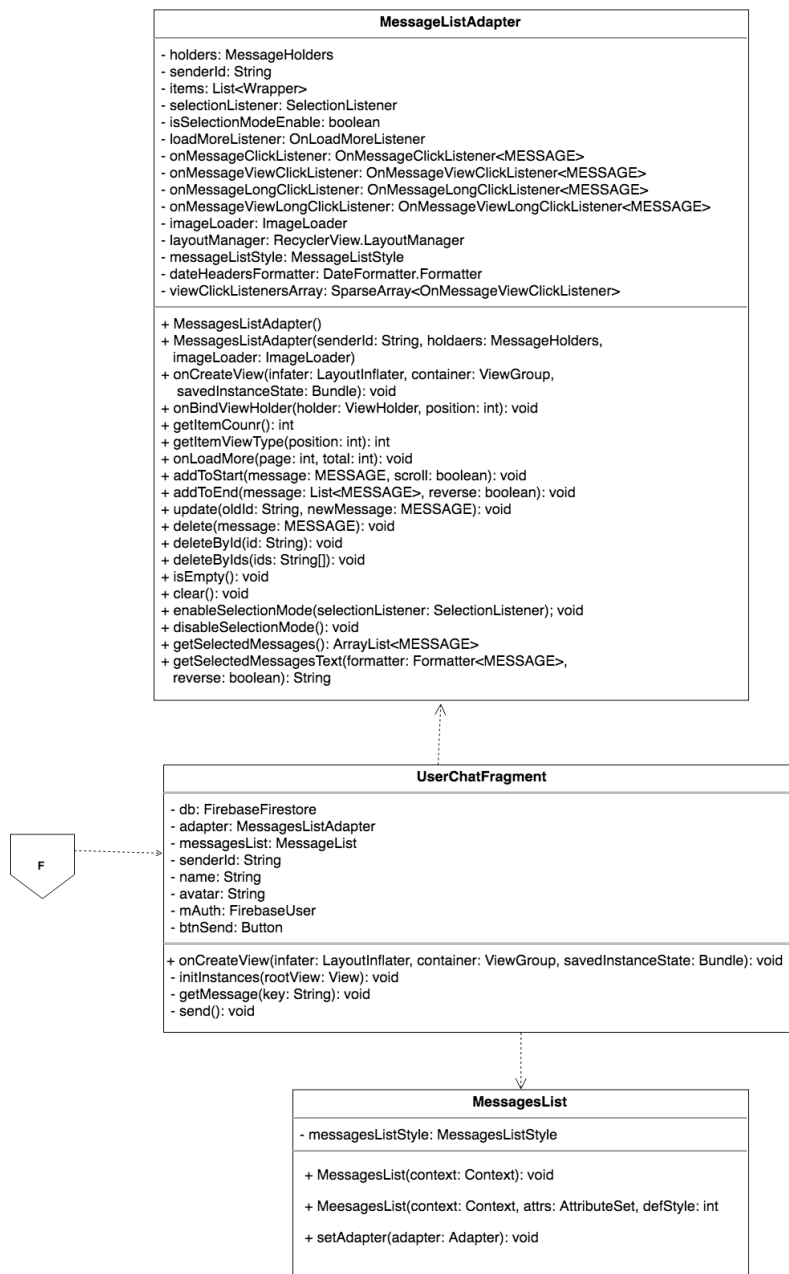


รูปที่ 3.21: Class Diagram ของแอปพลิเคชันระบบ XX



รูปที่ 3.22: Class Diagram ของแอปพลิเคชันระบบ XX





รูปที่ 3.23: Class Diagram ของแอปพลิเคชันระบบ XX

จากรูปภาพที่ 3.19 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.21: อธิบาย Class Diagram ของคลาสพื้นฐานของระบบ

Class Diagram	คำอธิบาย
MainApplication	คลาส MainApplication จะถูกเรียกใช้งานทุกครั้งเมื่อผู้ใช้เปิดแอปพลิเคชัน โดยวัตถุประสงค์การทำงานของคลาสนี้คือ เพื่อใช้จัดการทรัพยากรที่จำเป็นสำหรับการใช้งานในคลาสอื่น ๆ
SplashScreenActivity	คลาส SplashScreenActivity จะถูกเรียกใช้งานทุกครั้งเมื่อผู้ใช้เปิดแอปพลิเคชัน โดยวัตถุประสงค์การทำงานของคลาสนี้คือ เพื่อใช้ตรวจสอบสถานะการเข้าสู่ระบบของผู้ใช้
MainActivity	คลาส MainActivity เป็นคลาสหลักที่ใช้ในการทำงานของแอปพลิเคชันโดยการทำงานของคลาสนี้เน้นไปที่การสร้าง Fragment เพื่อใช้แสดงข้อมูลต่าง ๆ โดยองค์ประกอบการทำงานของคลาสนี้ประกอบไปด้วยสองส่วนหลักๆ ได้แก่ เมนูนำทาง Drawer และ Fragment Container
SignInActivity	คลาส SignInActivity เป็นคลาสที่ใช้เพื่อให้สมาชิกที่ได้ลงทะเบียนกับระบบเข้าสู่ระบบเพื่อใช้งานบริการต่าง ๆ จากระบบ

จากรูปภาพที่ 3.20 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.22: อธิบาย Class Diagram ของส่วนของการแสดงข่าวสาร

Class Diagram	คำอธิบาย
FeedFragment	คลาส FeedFragment เป็นคลาสหลักที่ใช้ในการแสดงข้อมูลข่าวสาร มีการทำงานหลักคือสืบค้นฐานข้อมูลจากไฟร์เบสเพื่อนำมาแสดง
FeedItemAdapter	คลาส FeedItemAdapter เป็นคลาสที่มีหน้าที่ในการแปลงชุดข้อมูลที่ได้จากคลาส FeedFragment แล้วคืนค่ากลับเป็นรายการของชุดข้อมูลนั้น ๆ
Post	คลาส Post เป็นคลาสโมเดลที่กำหนดค่าต่างๆที่จำเป็นสำหรับการสร้างรายการของคลาส FeedItemAdapter
PostDetailActivity	คลาส PostDetailActivity เป็น คลาส ที่ มีหน้าที่ ใน การ แสดง ข้อมูล รายละเอียด ของ ข่าวสาร แต่ละ แฉว ที่ ได้ รับ จาก หน้า FeedFragment ที่จะส่งข้อมูลเมื่อผู้ใช้งานกดที่แถวรายการข่าวสาร
RecyclerViewClickListener	คลาส RecyclerViewClickListener เป็น คลาส อินเทอร์เฟซ(Interface)ที่ใช้ในการสร้างแม่แบบเมื่อคลาสใด ๆ ต้องการใช้งานสำหรับการรับค่าเมื่อผู้ใช้งานกดแถวในรายการ คลาสลูก ที่ทำการสืบทอดคุณสมบัติจะสามารถรับข้อมูลตำแหน่งแถวที่ผู้ใช้งานกดบนรายการได้

จากรูปภาพที่ 3.21 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.23: อธิบาย Class Diagram ของส่วนของการแสดงรายการเอกสารในระบบ

Class Diagram	คำอธิบาย
DocumentsFragment	คลาส DocumentsFragment เป็นคลาสที่ใช้ในการแสดงข้อมูลเอกสารที่ถูกอัปโหลดเข้าสู่ระบบโดยเจ้าหน้าที่ซึ่งจะถูกแสดงเป็นลิสต์รายการ
DocItemAdapter	คลาส DocItemAdapter เป็นคลาสที่มีหน้าที่ในการแปลงชุดข้อมูลที่ได้รับจากคลาส DocumentsFragment เป็นลิสต์รายการแล้วคืนกลับไปยังคลาส DocumentsFragment
Doc	คลาส Doc เป็นคลาสโมเดลที่กำหนดค่าต่าง ๆ ที่จำเป็นสำหรับการสร้างลิสต์รายการของคลาส DocItemAdapter
RecyclerViewClickListener	คลาส RecyclerViewClickListener เป็นคลาสอินเทอร์เฟซที่ใช้ในการสร้างแม่แบบเมื่อคลาสใด ๆ ต้องการใช้งานสำหรับการรับค่าเมื่อผู้ใช้งานกดในลิสต์รายการ คลาสลูกที่ทำการสืบทอดคุณสมบัติจะสามารถรับข้อมูลตำแหน่งแถวที่ผู้ใช้งานบนลิสต์รายการได้

จากรูปภาพที่ 3.22 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.24: อธิบาย Class Diagram ของส่วนของการส่งสำเนาเอกสาร

Class Diagram	คำอธิบาย
SubmitFragment	คลาส SubmitFragment เป็นคลาสที่ใช้ในการแสดงหน้าจอส่งสำเนาเอกสาร โดยมีการดำเนินการภายในคลาสหลัก ๆ ได้แก่ การถ่ายภาพ แปลงภาพและบันทึกภาพเข้าสู่ระบบ

จากรูปภาพที่ 3.23 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

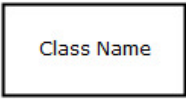


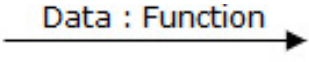
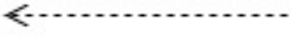


ตารางที่ 3.25: อธิบาย Class Diagram ของส่วนของการสนทนา

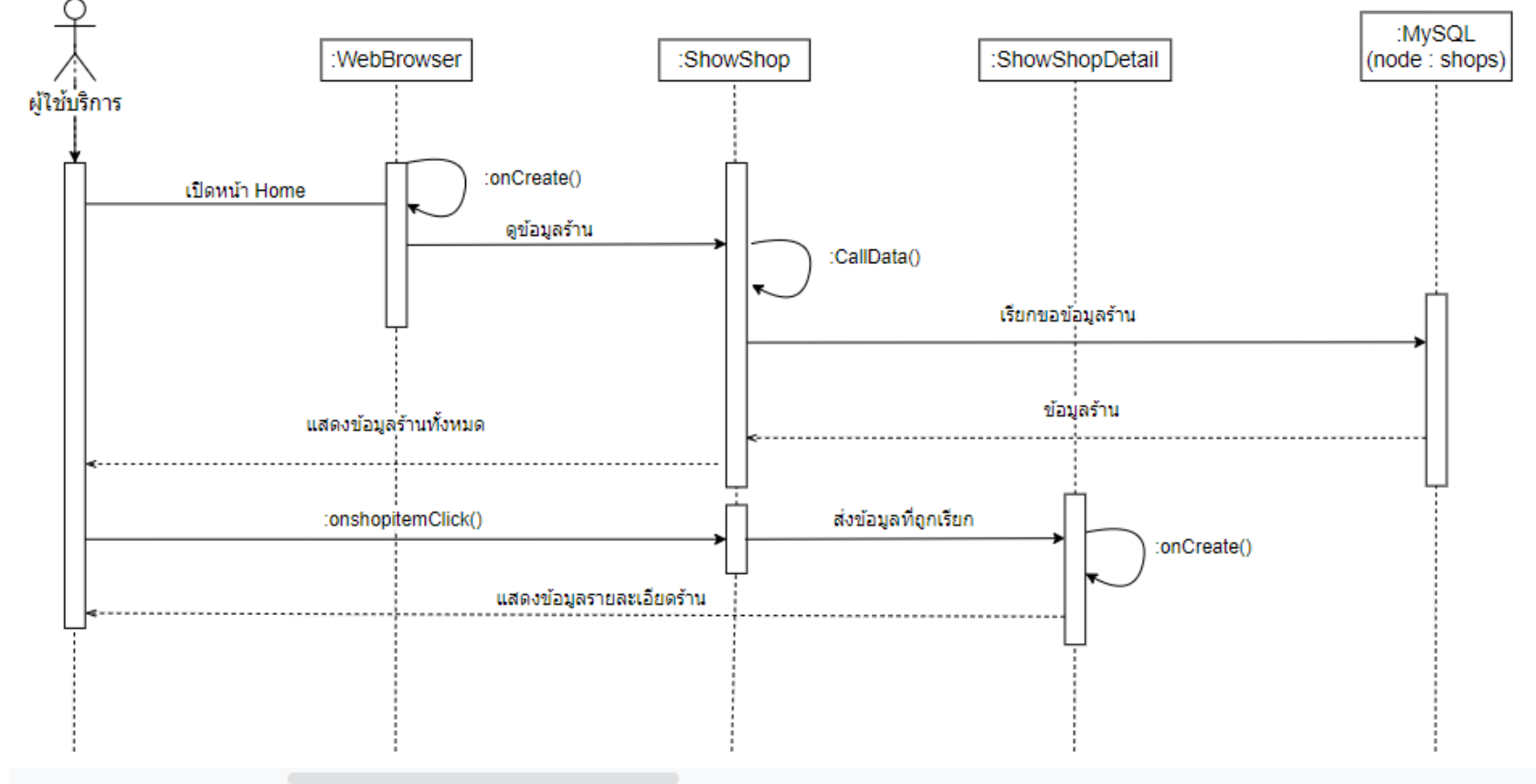
Class Diagram	คำอธิบาย
UserChatFragment	คลาส UserChatFragment เป็นคลาสที่ใช้ในการแสดงหน้าจอสนทนาสำหรับนักศึกษา เพื่อติดต่อสอบถามข้อมูลกับเจ้าหน้าที่ มีการสืบค้นข้อมูลประวัติการสนทนาเพื่อส่งไปแปลงเป็นข้อมูลลิสต์รายการที่คลาส MessagesListAdapter
MessagesListAdapter	คลาส MessagesListAdapter เป็นคลาสที่ใช้ในการแปลงชุดข้อมูลที่ได้รับจากคลาส UserChatFragment เป็นลิสต์รายการแล้วทำการคืนค่าลิสต์รายการที่ได้กลับไปยังคลาส UserChatFragment
MessagesList	คลาส MessagesList เป็นคลาสที่ใช้ในการจัดเก็บข้อมูลภายในคลาส UserChatFragment หลังจากที่ได้ทำการสืบค้นข้อมูลการสนทนาจากไฟร์เบสเพื่อส่งไปยังคลาส MessagesListAdapter
RecyclerViewClickListener	คลาส RecyclerViewClickListener เป็นคลาสอินเทอร์เฟสที่ใช้ในการสร้างแม่แบบเมื่อคลาสใด ๆ ต้องการใช้งานสำหรับการรับค่าเมื่อผู้ใช้กดแถวในลิสต์รายการ คลาสลูกที่ทำการสืบทอดคุณสมบัติจะสามารถรับข้อมูลตำแหน่งแถวที่ผู้ใช้กดบนลิสต์รายการได้

### 3.6 Sequence Diagram

Sequence Diagram เป็น Diagram ที่แสดงขั้นตอนการทำงานของแต่ละ Use Case ระหว่าง Object ต่างๆ ที่ส่งข้อความถึงกันและกัน โดย Sequence Diagram จะช่วยให้มองเห็นการทำงานของภาพรวมของระบบ ส่วนประกอบสัญลักษณ์ที่ใช้ในการเขียน Sequence Diagram แสดงดังตารางที่ 3.26

ตารางที่ 3.26: สัญลักษณ์ของ Sequence Diagram

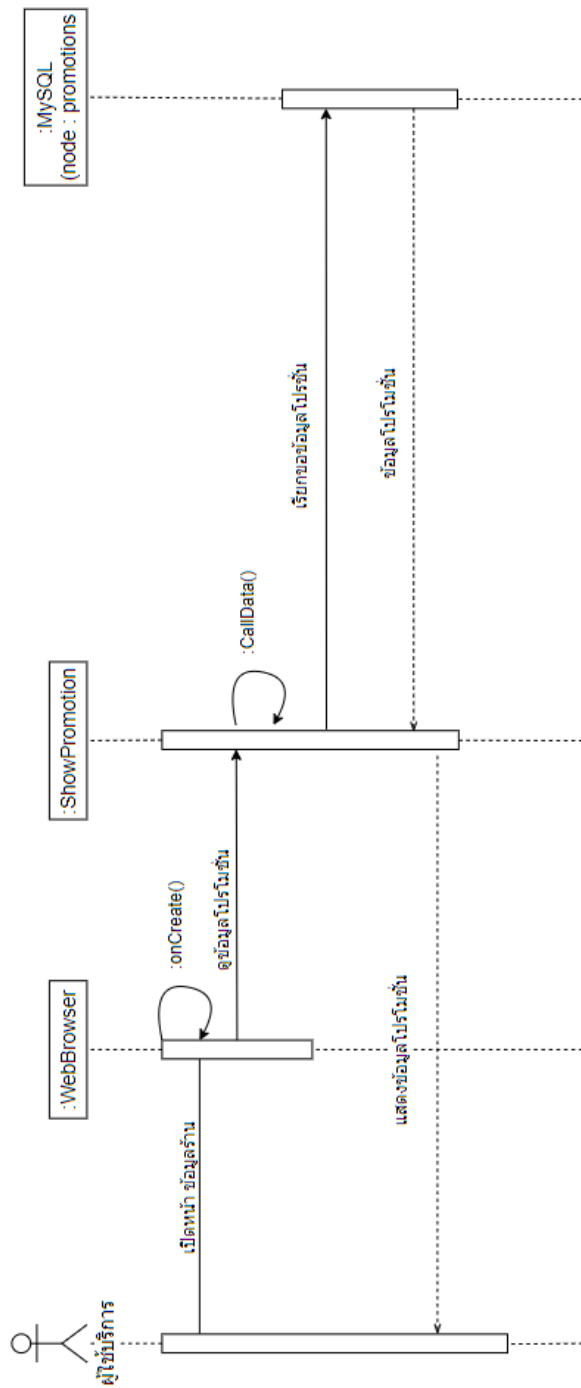
สัญลักษณ์	การใช้งาน
	Class แสดงถึงการทำงานของ Use Case ในการส่งหรือรับข้อความ แทนด้วยสัญลักษณ์สี่เหลี่ยมมีชื่อคลาสอยู่ภายใน
	Lifeline หรือเส้นอายุขัย แสดงช่วงเวลาตั้งแต่เริ่มสร้าง object ในคลาสนั้น จนกระทั่ง object นั้นถูกทำลาย สัญลักษณ์แทนด้วยเส้นประ
	Focus of control หรือจุดควบคุม เป็นจุดควบคุมที่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



รูปที่ 3.24: Sequence Diagram ข้อมูลร้าน

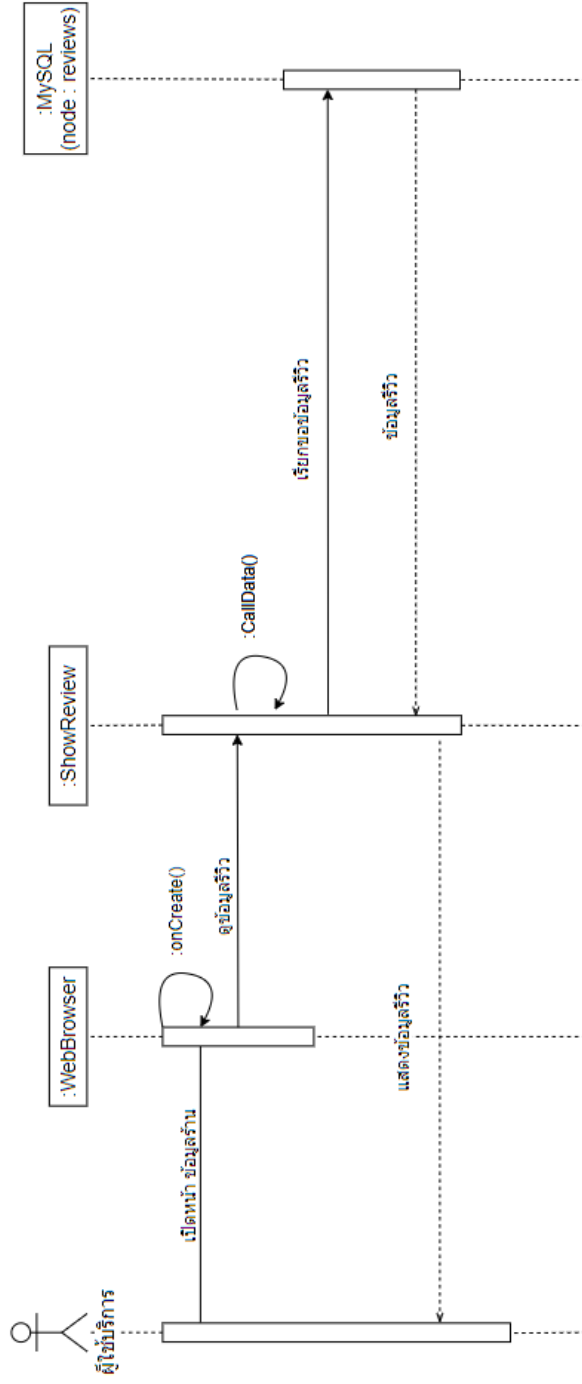
จากภาพที่ 3.24 สามารถอธิบายแผนภาพ Sequence Diagram แสดงข้อมูลร้าน ได้ดังนี้ เมื่อ ผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส WebBrowser เมื่อ Shopdata ถูกติดตั้งบน ShowShop เมธอด callData() จะสืบค้นข้อมูลจากฐานข้อมูลบน MySQL (node:shops) และส่งข้อมูลที่ได้ไปแปลงที่คลาส ShowShop โดยมีการคืนค่าเป็นข้อมูลร้านแต่ละแถวและในชั้น ตอนสุดท้ายคลาส ShowShop จะทำการแสดงรายการข้อมูลร้านทั้งหมดออกทางหน้าจอ หากผู้ใช้มีการกดเลือกร้านบางแถวคลาส ShowShop จะทำการเรียกใช้ ShowShopDetail เพื่อแสดง รายละเอียดข้อมูลร้านของแถวที่ถูกเลือก





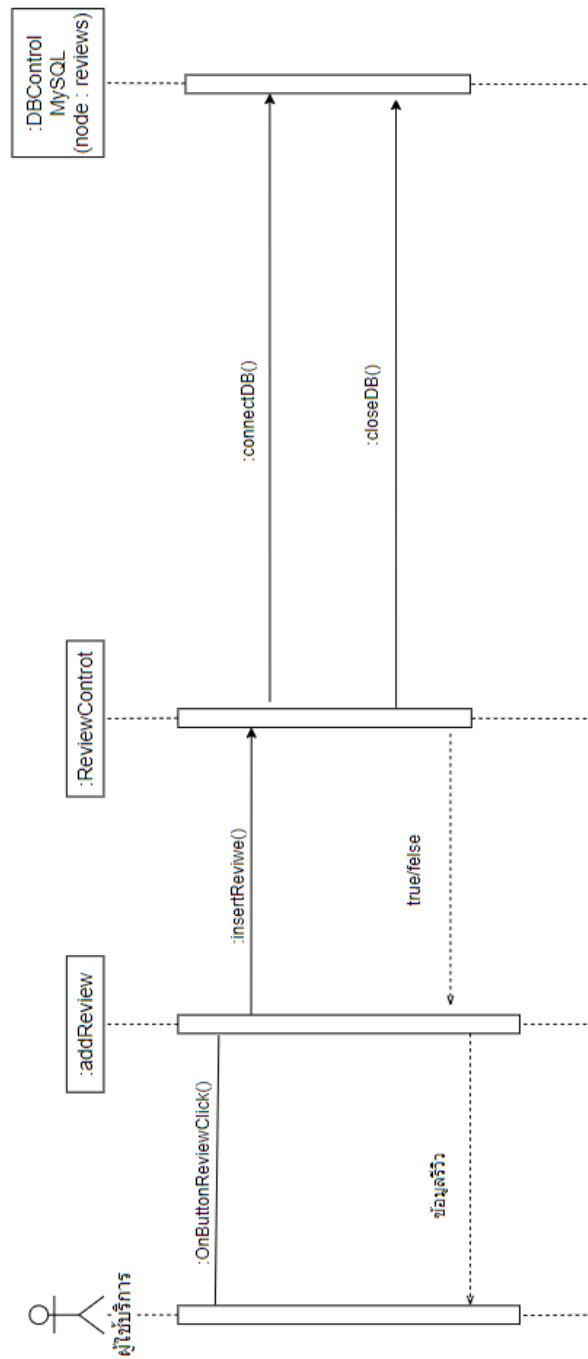
รูปที่ 3.25: Sequence Diagram การแสดงข้อมูลโปรโมชั่น

จากภาพที่ 3.25 สามารถอธิบายแผนภาพ Sequence Diagram แสดงโปรโมชั่น ได้ดังนี้ เมื่อ ผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส WebBrowser ระบบจะทำการสร้าง ข้อมูลโปรโมชั่น ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส WebBrowser เมื่อ ข้อมูลโปรโมชั่น ถูกติดตั้งบน ShowPromotion เมธอด callData() จะสืบค้นข้อมูลกำหนดการของวันปัจจุบันจากฐานข้อมูลบน MySQL node:promotions และส่งข้อมูลที่ได้ไปแปลงที่คลาส ShowPromotion โดยขั้นตอนสุดท้ายที่คลาส ShowPromotion จะทำการแสดงข้อมูลโปรโมชั่นออกทางหน้าจอ



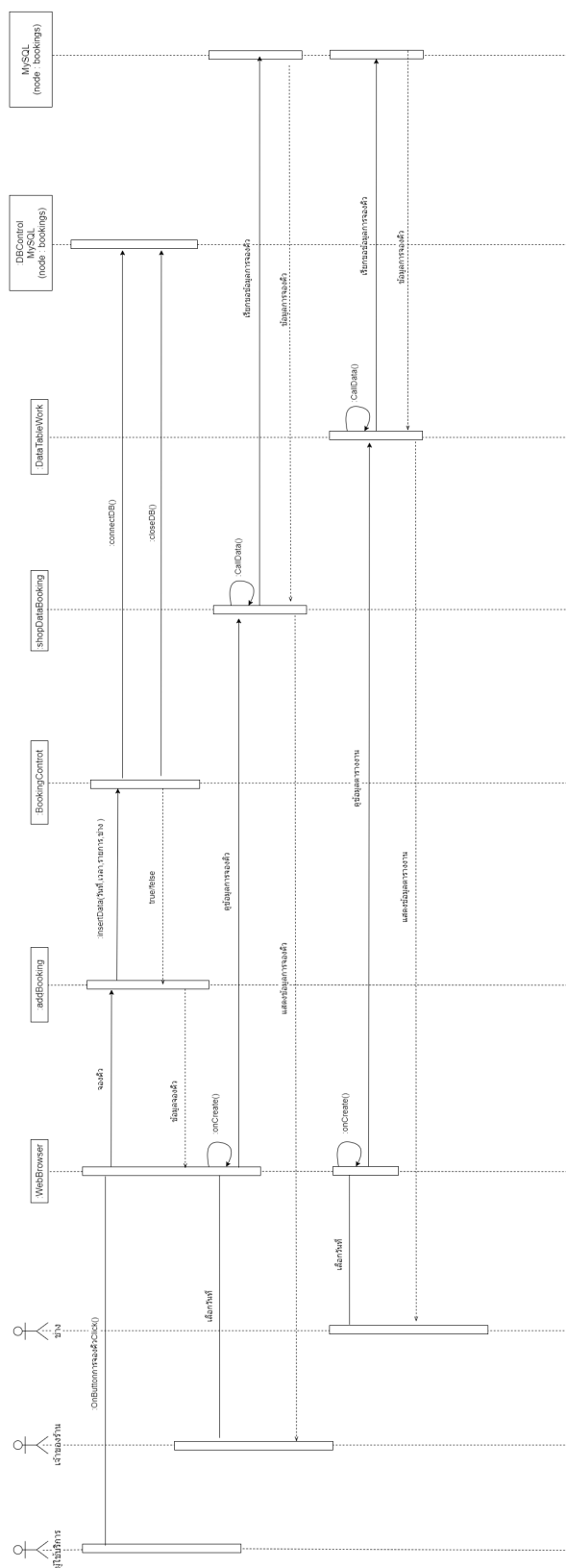
รูปที่ 3.26: Sequence Diagram การแสดงข้อมูลรีวิว

จากภาพที่ 3.26 สามารถอธิบายแผนภาพ Sequence Diagram แสดงข้อมูลรีวิว ได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส WebBrowser ระบบจะทำการสร้าง ข้อมูลรีวิว ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส WebBrowser เมื่อ ข้อมูลรีวิว ถูกติดตั้งบน Showreview เมธอด callData() จะสืบค้นข้อมูลรีวิวทั้งหมดจากฐานข้อมูลบน MySQL node:reviews และส่งข้อมูลที่ได้ไปแปลงที่คลาส Showreview โดยจะทำการแสดงข้อมูลรีวิวออกทางหน้าจอ



รูปที่ 3.27: Sequence Diagram การเพิ่มรีวิว

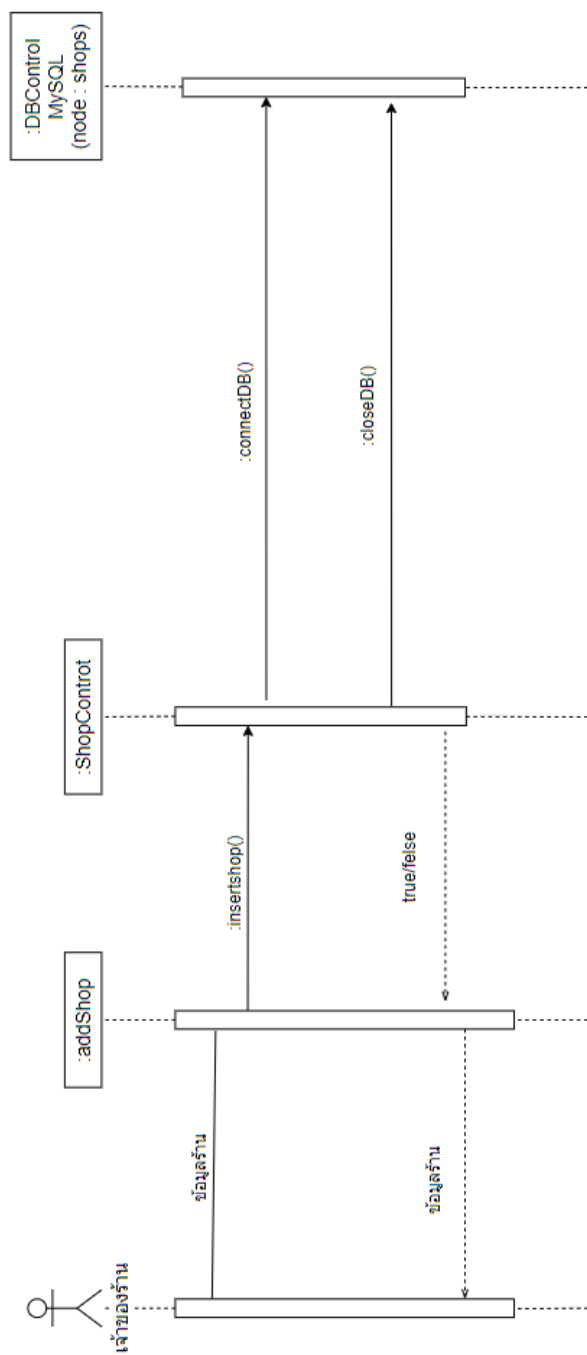
จากภาพที่ 3.27 สามารถอธิบายแผนภาพ Sequence Diagram แสดงการสนทนา ได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส UserChatFragment เมื่อ UserChatFragment ถูกติดตั้งบน MainActivity เมธอด getMessage() จะสืบค้นข้อมูลประวัติการสนทนาของผู้ใช้คนปัจจุบันทั้งหมดจากฐานข้อมูลบน Firebase Firestore และส่งข้อมูลที่ได้ไปแปลงที่คลาส MessagesListAdapter โดยมีการคืนค่าเป็นข้อมูลรายการประวัติการสนทนาทั้งหมดและในขั้นตอนสุดท้ายที่คลาส User-ChatFragment จะทำการแสดงรายการประวัติการสนทนาทั้งหมดออกทางหน้าจอ เมื่อผู้ใช้พิมพ์ข้อความและกดปุ่มส่งระบบจะเรียกให้เมธอด send() เพื่อทำการบันทึกข้อมูลไว้บน Firebase Firestore และทำการแสดงข้อมูลรายการประวัติการสนทนาทั้งหมดที่ถูกอัปเดต



รูปที่ 3.28: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

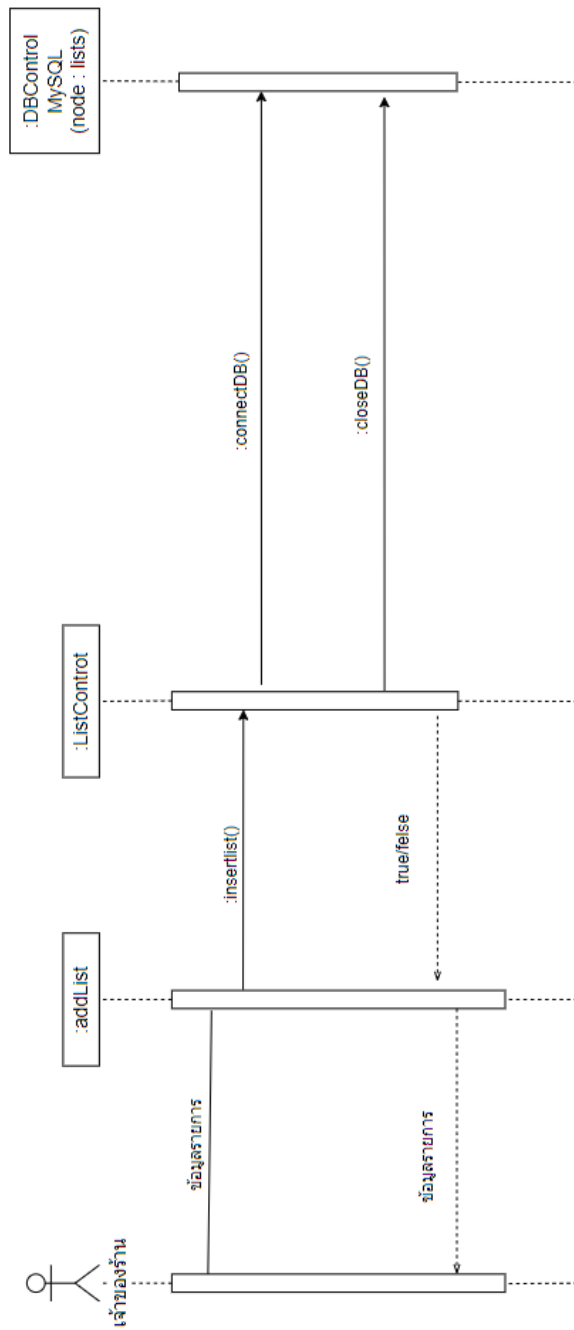
จากภาพที่ 3.28 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายครบทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ





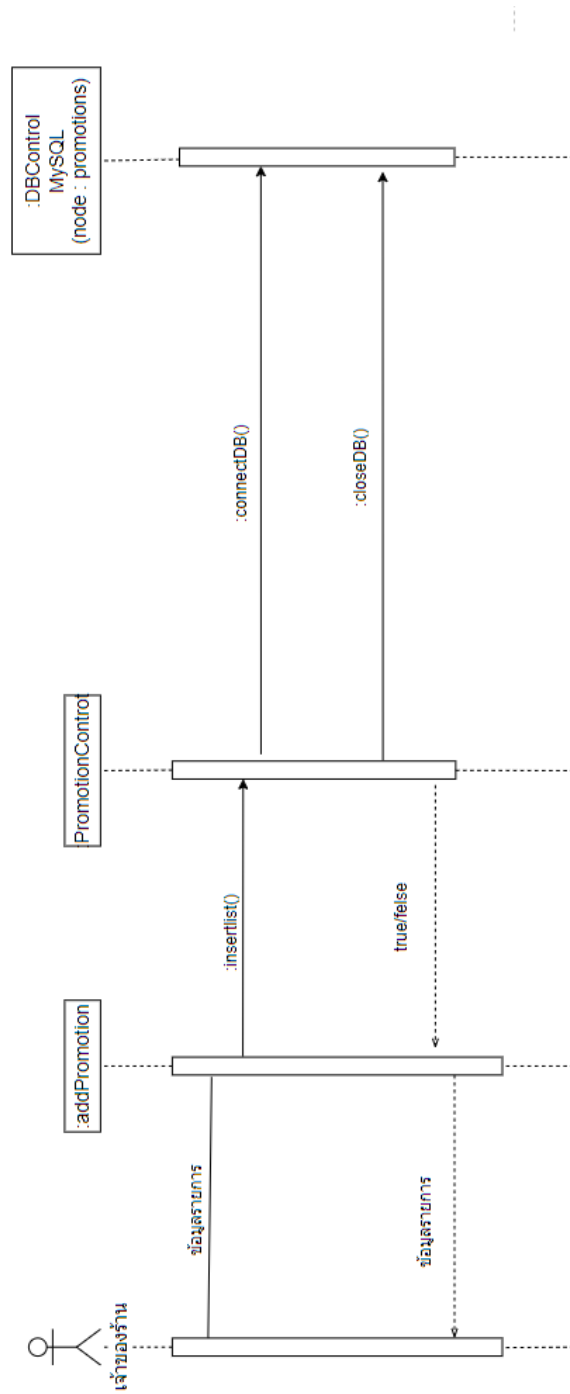
รูปที่ 3.29: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

จากภาพที่ 3.29 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายครบทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ



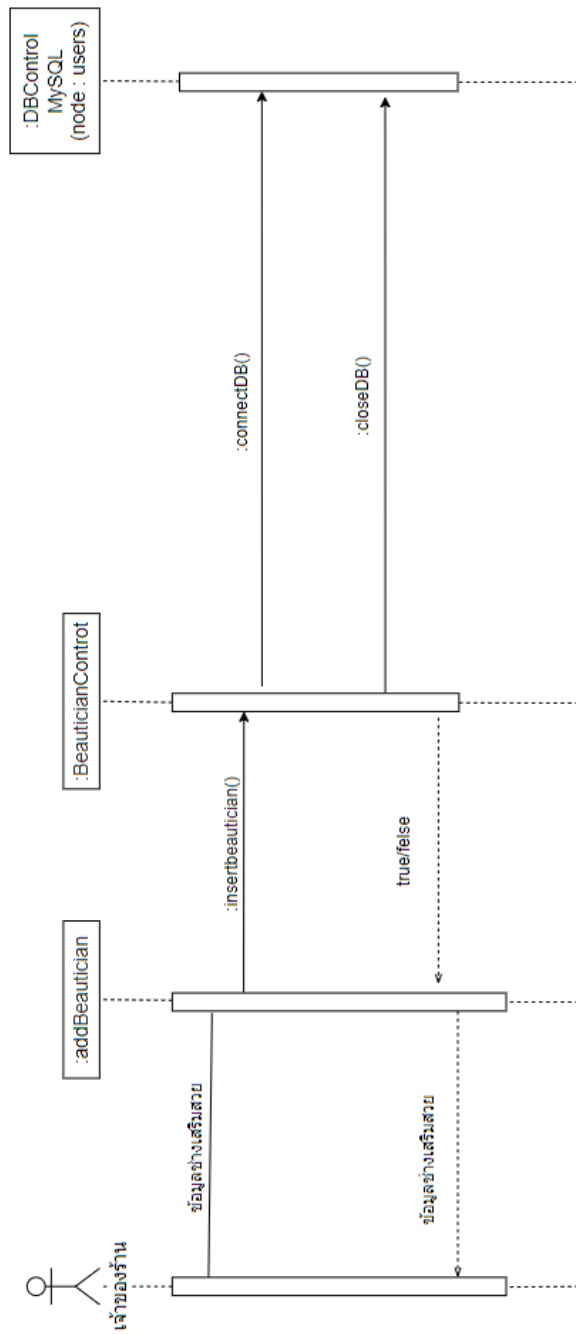
รูปที่ 3.30: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

จากภาพที่ 3.30 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายครบทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ



รูปที่ 3.31: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

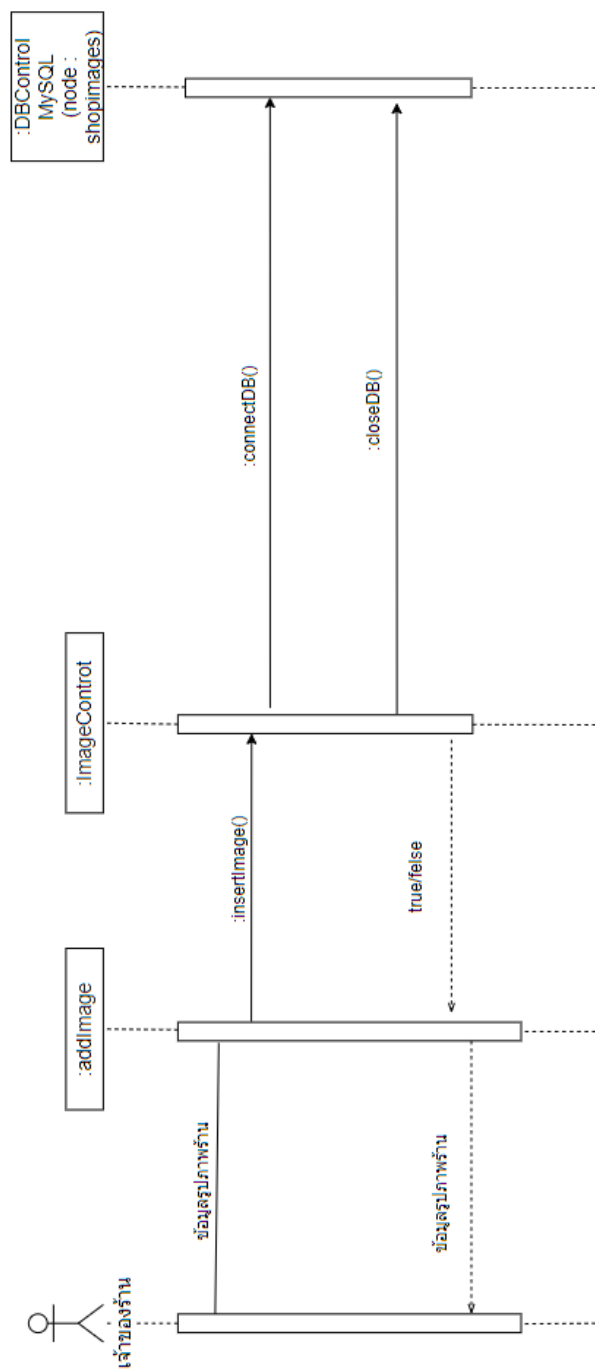
จากภาพที่ 3.31 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายรูปทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ



รูปที่ 3.32: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

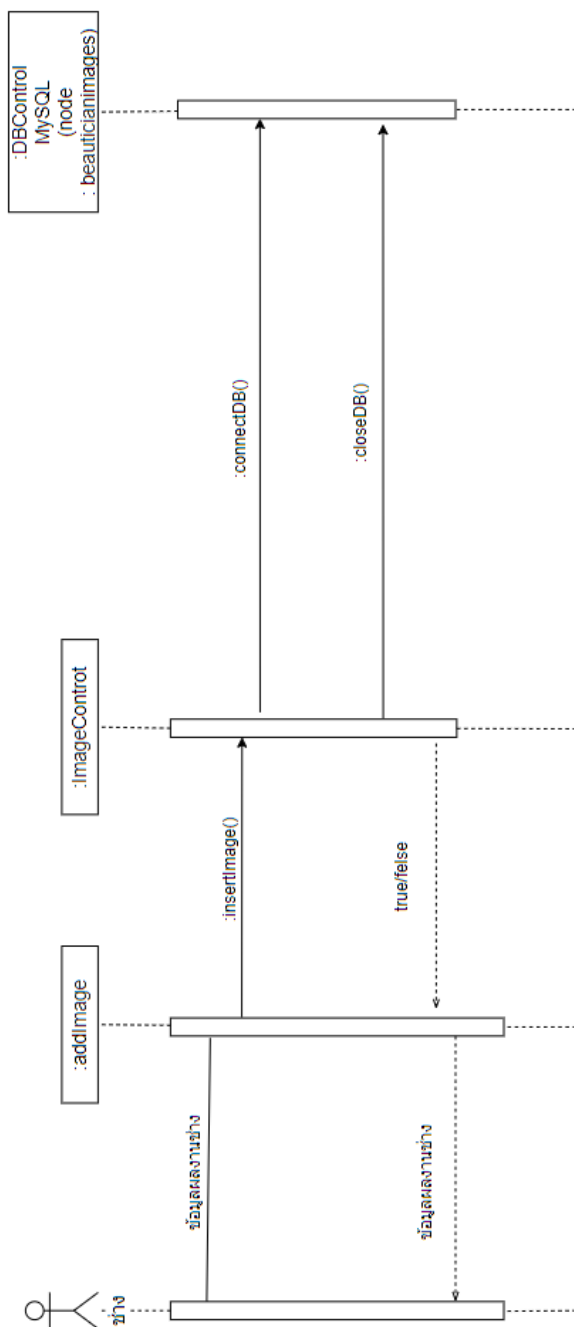
จากภาพที่ 3.32 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายครบทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ





รูปที่ 3.33: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

จากภาพที่ 3.33 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายรูปทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ





รูปที่ 3.34: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

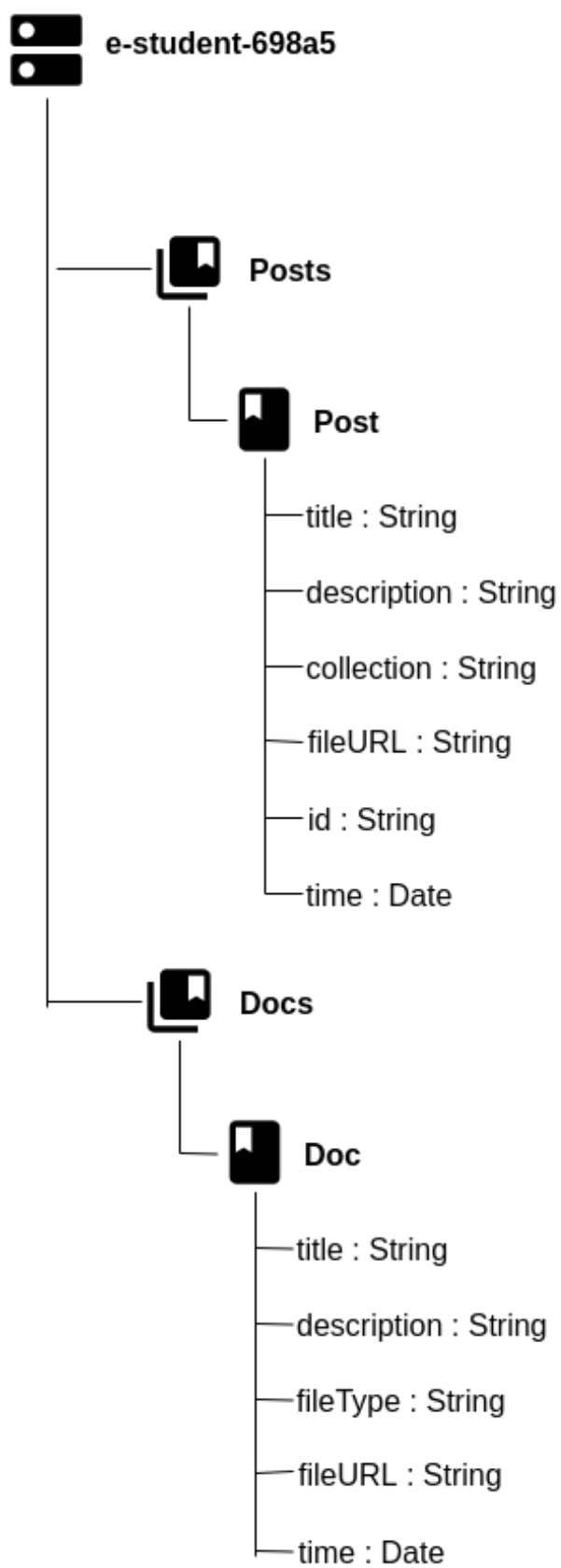
จากภาพที่ 3.34 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายครบทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ

### 3.7 โครงสร้างฐานข้อมูลไฟร์เบส(Firebase Database Structure)

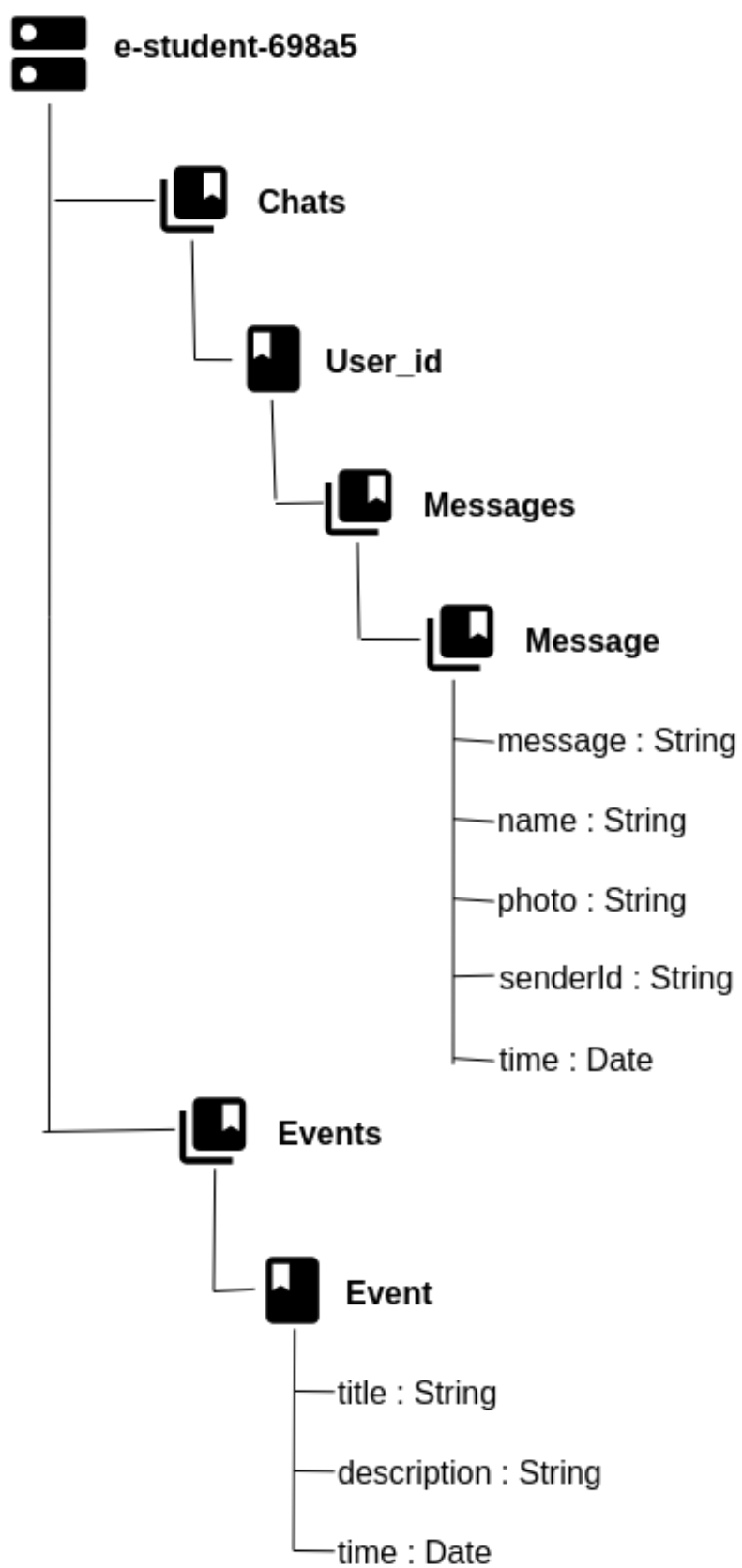
Firebase Database นั้นเป็น Database แบบ NoSQL และเป็น JSON database ที่มีโครงสร้างที่เป็น Key และ Value จัดเก็บข้อมูลในลักษณะโหนด หากต้องการเรียกงานจะเรียกใช้ โดย การท่องไปยังโหนดที่ต้องการ ส่วนประกอบสัญลักษณ์ที่ใช้ในการเขียนโครงสร้างฐานข้อมูลแบบ Firebase แสดงดังตารางที่ 3.27

ตารางที่ 3.27: สัญลักษณ์ของโครงสร้างฐานข้อมูลแบบ Firebase

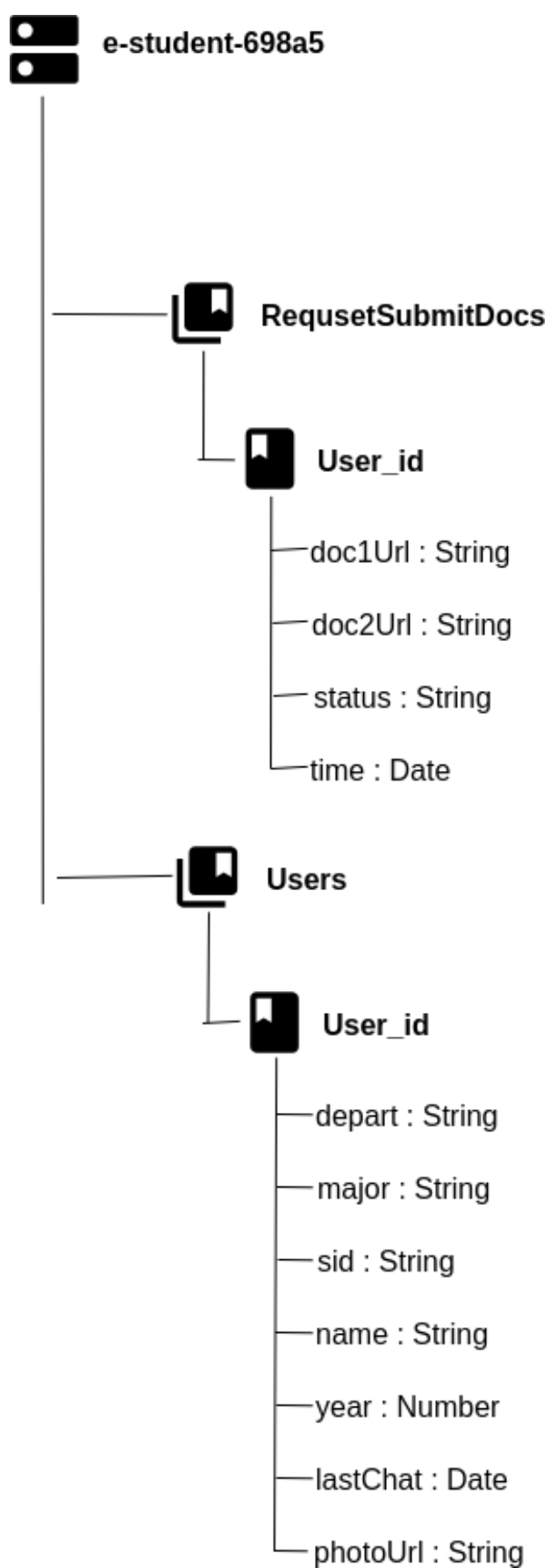
สัญลักษณ์	คำอธิบาย
	Database เป็นการเรียกชื่อแทนโหนด(Node)บนสุดที่ใช้ในการเก็บข้อมูล
	Collection เป็นการเรียกชื่อแทนของการเก็บหลาย ๆ เอกสารไว้ด้วยกัน
	Document เป็นการ เรียก ชื่อ แทน หน่วย การ เก็บ ของ ข้อมูล ใน Cloud Firestore ภายในจะประกอบไปด้วย ชื่อของ Document ชื่อของคีย์ (key) และ ค่าข้อมูล (value) โดยชื่อของ Document ห้ามซ้ำกัน ซึ่งใน Cloud Firestore สามารถระบุประเภทของข้อมูลได้ 9 ประเภทได้แก่ boolean, number, string, geo point, timestamp, array, object, reference และ null



รูปที่ 3.35: โครงสร้างฐานข้อมูลแบบ Firebase

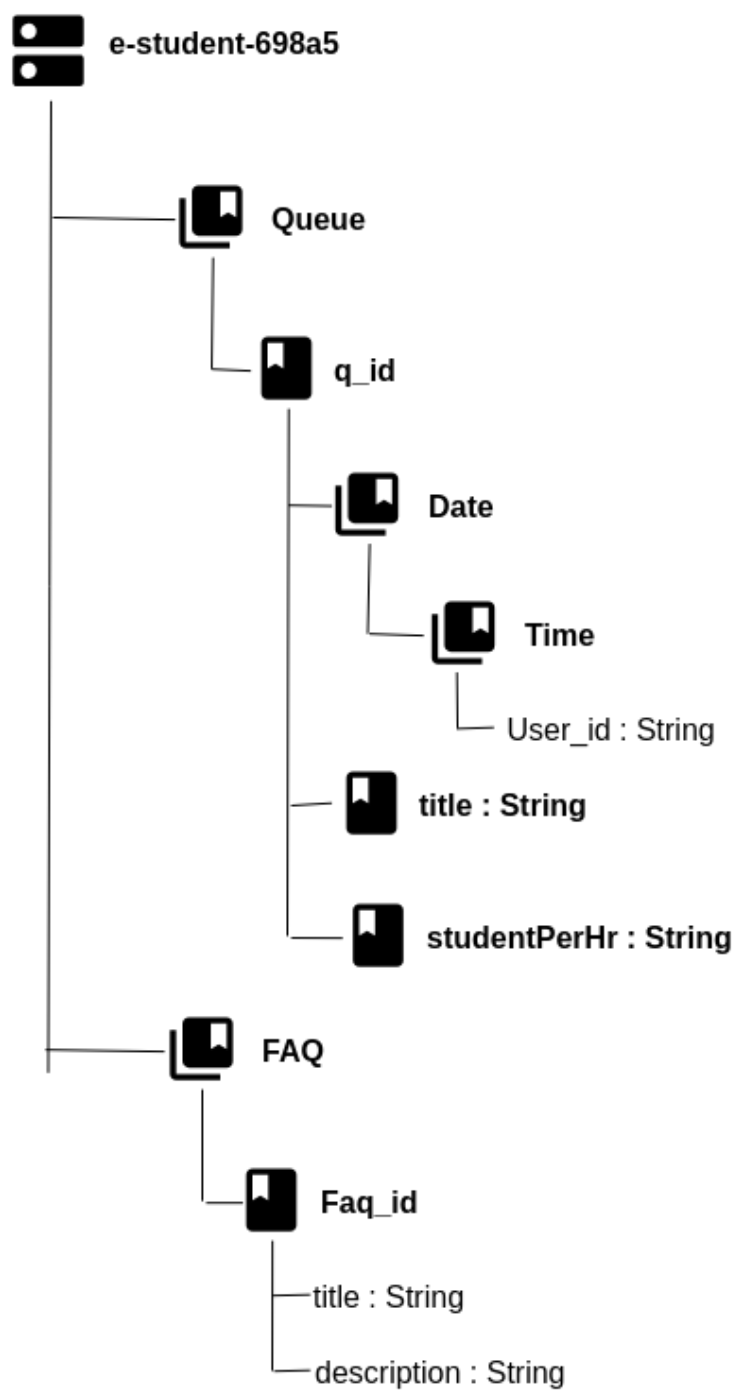


รูปที่ 3.36: โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ)



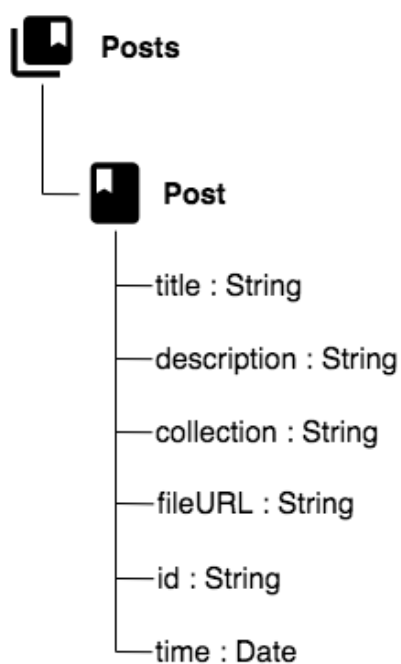
รูปที่ 3.37: โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ)





รูปที่ 3.38: โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ)

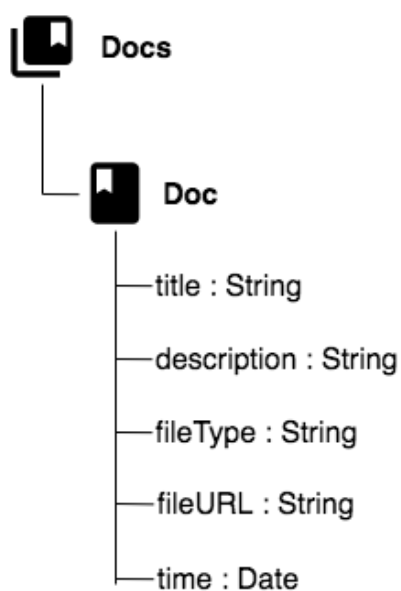
จากรูปที่ 3.35-3.46 สามารถอธิบายโครงสร้างของข้อมูลได้ดังนี้



รูปที่ 3.39: โหนดเก็บข้อมูลประกาศ

ตารางที่ 3.28: อธิบายโน้ตที่ใช้เก็บข้อมูลประกาศ

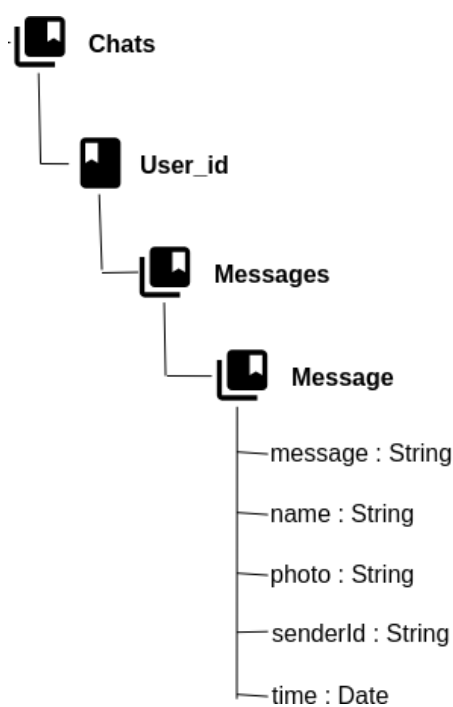
Key	คำอธิบาย
Posts	โน้ตสำหรับเก็บข้อมูลประกาศทั้งหมด
Post	สำหรับเก็บข้อมูลแต่ละประกาศ
title	สำหรับเก็บชื่อหัวข้อประกาศ
description	สำหรับเก็บรายละเอียดประกาศ
collection	สำหรับเก็บประเภทของประกาศได้แก่ สาธารณะและเฉพาะบุคคล
fileURL	สำหรับเก็บ url ของเอกสารแนบประกาศ
id	สำหรับเก็บรหัสของประกาศ
time	สำหรับเก็บเวลาที่ประกาศ



รูปที่ 3.40: โหนดเก็บข้อมูลเอกสารที่เกี่ยวข้อง

ตารางที่ 3.29: อธิบายโหนดที่ใช้เก็บข้อมูลเอกสารที่เกี่ยวข้อง

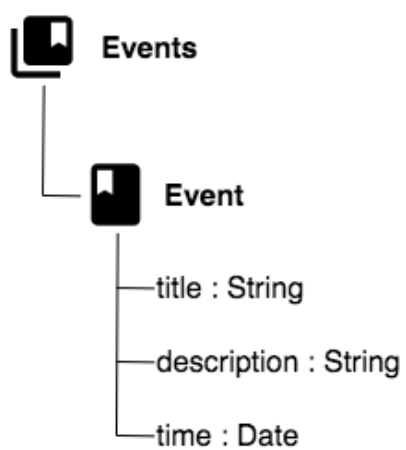
Key	คำอธิบาย
Docs	โหนดสำหรับเก็บข้อมูลของเอกสารที่เกี่ยวข้องทั้งหมด
Doc	สำหรับเก็บข้อมูลเอกสารแต่ละฉบับ
title	สำหรับเก็บชื่อหัวเรื่องของเอกสาร
description	สำหรับเก็บรายละเอียดของเอกสาร
fileType	สำหรับนามสกุลไฟล์เอกสาร เช่น .pdf .png เป็นต้น
fileURL	สำหรับเก็บ url ของเอกสาร
time	สำหรับเก็บเวลาที่ถูกอัปโหลดเข้าสู่ระบบโดยเจ้าหน้าที่



รูปที่ 3.41: โหนดเก็บข้อมูลประวัติการสนทนา

ตารางที่ 3.30: อธิบายโหนดที่ใช้เก็บข้อมูลประวัติการสนทนา

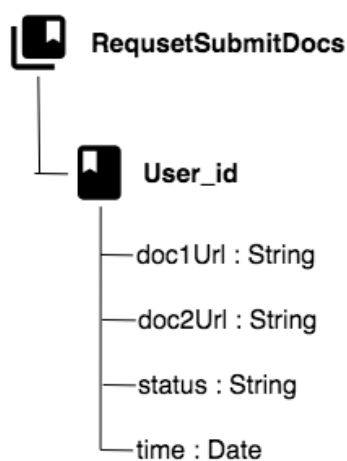
Key	คำอธิบาย
Chats	โหนดสำหรับเก็บข้อมูลประวัติการสนทนาทั้งหมด
User_id	สำหรับเก็บประวัติการสนทนาของผู้ใช้แต่ละคน
Messages	สำหรับเก็บประวัติการสนทนาทั้งหมดของผู้ใช้
Message	สำหรับเก็บข้อมูลของแต่ละข้อความ
message	สำหรับเก็บข้อความ
name	สำหรับเก็บชื่อของผู้ส่งข้อความ
photo	สำหรับเก็บ url รูปภาพของผู้ส่งข้อความ
senderId	สำหรับเก็บรหัสของผู้ส่งข้อความ
time	สำหรับเก็บเวลาที่ข้อความถูกส่ง



รูปที่ 3.42: โหนดเก็บข้อมูลกำหนดการ

ตารางที่ 3.31: อธิบายโหนดที่ใช้เก็บข้อมูลกำหนดการ

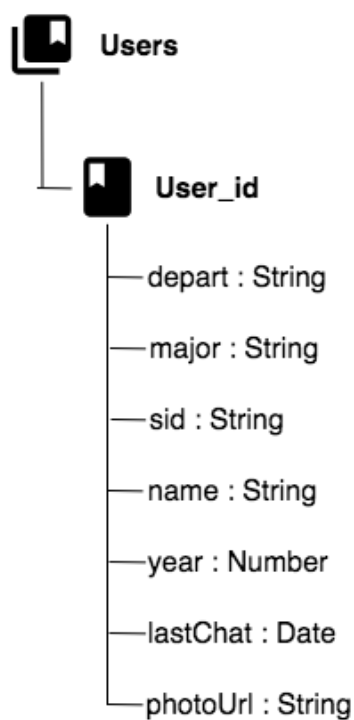
Key	คำอธิบาย
Events	โหนดสำหรับเก็บข้อมูลของกำหนดการทั้งหมด
Event	สำหรับเก็บข้อมูลของแต่ละกำหนดการ
title	สำหรับเก็บชื่อหัวข้อของกำหนดการ
description	สำหรับเก็บรายละเอียดของกำหนดการ
time	สำหรับเก็บเวลาของกำหนดการ



รูปที่ 3.43: โหนดเก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา

ตารางที่ 3.32: อธิบายโหนดที่ใช้เก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา

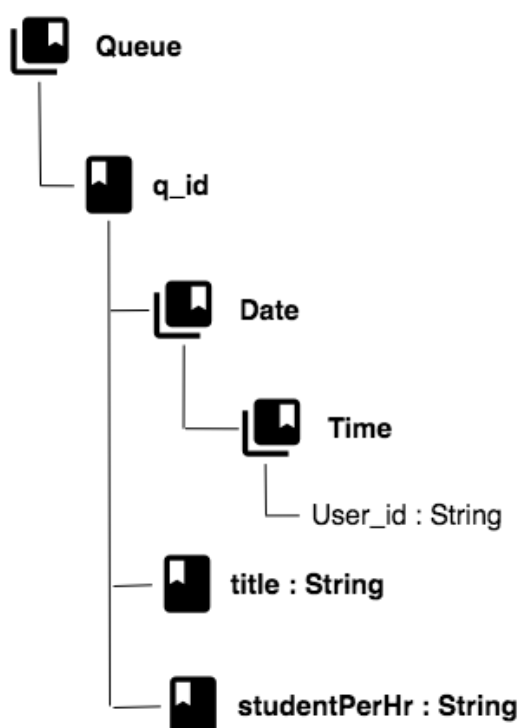
Key	คำอธิบาย
RusetSubmitDocs	โหนดสำหรับเก็บ ข้อมูล การยื่นสำเนาเอกสาร เพื่อ ตรวจสอบ ของนักศึกษาทั้งหมด
User_id	สำหรับเก็บข้อมูลของแต่ละสำเนาเอกสารของนักศึกษาแต่ละคน
doc2	สำหรับเก็บ url ของภาพถ่ายสำเนาเอกสารฉบับที่ 1
doc2	สำหรับเก็บ url ของภาพถ่ายสำเนาเอกสารฉบับที่ 2
status	สำหรับเก็บผลการตรวจสอบของเจ้าหน้าที่
time	สำหรับเก็บเวลาที่สำเนาเอกสารถูกเพิ่มเข้าสู่ระบบ



รูปที่ 3.44: โหนดเก็บข้อมูลของนักศึกษา

ตารางที่ 3.33: อธิบายโหนดที่ใช้เก็บข้อมูลของนักศึกษา

Key	คำอธิบาย
Users	โหนดสำหรับเก็บข้อมูลของนักศึกษา
User_id	สำหรับเก็บข้อมูลของนักศึกษาแต่ละคน
depart	สำหรับเก็บภาควิชาของนักศึกษา
major	สำหรับเก็บสาขาของนักศึกษา
sid	สำหรับเก็บรหัสประจำตัวนักศึกษา
name	สำหรับเก็บชื่อของนักศึกษา
year	สำหรับเก็บชั้นปีของนักศึกษา
lastChat	สำหรับเก็บเวลาที่สนทนากับเจ้าหน้าที่ล่าสุด
photoUrl	สำหรับเก็บ url รูปภาพโปรไฟล์ (Profile)

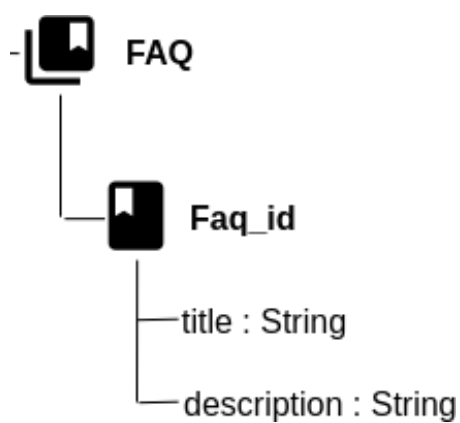


รูปที่ 3.45: โหนดเก็บข้อมูลการจองคิวของนักศึกษา

ตารางที่ 3.34: อธิบายโหนดที่ใช้เก็บข้อมูลการจองคิวของนักศึกษา

Key	คำอธิบาย
Queue	โหนดสำหรับเก็บข้อมูลการจองคิวของนักศึกษาทั้งหมด
q_id	สำหรับเก็บข้อมูลของการจองคิวแต่ละครั้งที่เปิดจองคิว
Date	สำหรับเก็บวันที่สำหรับส่งเอกสาร
Time	สำหรับเก็บรายชื่อของนักศึกษาที่ทำการจองคิวในส่งเอกสารเวลานั้น ๆ
User_id	สำหรับเก็บรหัสของนักศึกษา
title	สำหรับเก็บชื่อหัวเรื่องกำหนดการการจองคิว
studentPerHr	สำหรับเก็บจำนวนนักศึกษาต่อชั่วโมง





รูปที่ 3.46: โหนดเก็บข้อมูลคำถามที่พบบ่อย

ตารางที่ 3.35: อธิบายโหนดที่ใช้เก็บข้อมูลคำถามที่พบบ่อย

Key	คำอธิบาย
Queue	โหนดสำหรับเก็บข้อมูลคำถามที่พบบ่อยทั้งหมด
Faq_id	สำหรับเก็บข้อมูลคำถามที่พบบ่อยแต่ละรายการ
title	สำหรับเก็บคำถาม
description	สำหรับเก็บคำตอบ

## บทที่ 4

### การพัฒนาระบบ

หลังจากที่ได้มีการเตรียมความพร้อมสำหรับการพัฒนาในด้านต่าง ไม่ว่าจะเป็นที่มาและความสำคัญของปัญหา เทคโนโลยีที่มีความเหมาะสมกับระบบ และการออกแบบระบบการทำงาน รวมไปถึงโครงสร้างของข้อมูล ในบทนี้จะเป็นการพูดถึงการสร้างระบบที่ได้มีการออกแบบไว้ในบทที่แล้วจะถูกนำเสนอในบทนี้ โดยการพัฒนาระบบแบ่งได้เป็นส่วนต่าง ๆ ดังนี้

#### 4.1 การพัฒนาเว็บแอปพลิเคชัน

#### 4.2 การพัฒนาเว็บแอปพลิเคชันแอนดรอยด์แอปพลิเคชัน

#### 4.1 การพัฒนาเว็บแอปพลิเคชัน

การพัฒนาระบบกองทุนเงินให้กู้ยืมเพื่อการศึกษาสำหรับเว็บแอปพลิเคชันนั้นวัตถุประสงค์หลังเพื่อสร้างความสะดวกต่อการทำงานของเจ้าหน้าที่อื่นเนื่องมาจากข้อจำกัดบางประการหากใช้ระบบทำงานบนอุปกรณ์สมาร์ตโฟนเพียงอย่างเดียว โดยตัวเว็บแอปพลิเคชันนี้ถูกพัฒนาขึ้นด้วย Vue.js มีรายละเอียดการทำงานดังนี้

##### 4.1.1 การเชื่อมต่อ Cloud Firestore

bsection, ในการเชื่อมต่อเว็บแอปพลิเคชันกับไฟร์เบสเพื่อใช้บริการต่างๆ ของไฟร์เบส ทำได้ดังนี้

```
1 export default {
2   apiKey: "
      XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
3   authDomain: "e-student-698a5.firebaseio.com",
4   databaseURL: "https://e-student-698a5.firebaseio.com",
5   projectId: "e-student-698a5",
6   storageBucket: "e-student-698a5.appspot.com",
7   messagingSenderId: "000000000000"
8 }
```

รูปที่ 4.1: ไฟล์ firebaseConfig.js

จากภาพที่ 4.2 โครงสร้างของไฟล์ firebaseConfig.js สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 เป็นการส่งออกโมดูลเพื่อใช้งานในไฟล์อื่น
- บรรทัดที่ 2-7 เป็นการตั้งค่าระบุตัวตนเพื่อใช้งานบริการไฟร์เบส

```
1 import firebase from 'firebase'
2 import 'firebase/firestore'
3 import firebaseConfig from './firebaseConfig'
4
5 export default firebase.initializeApp(firebaseConfig)
```

รูปที่ 4.2: ไฟล์ firebaseInit.js

จากภาพที่ 4.2 โครงสร้างของไฟล์ firebaseInit.js สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 เป็นการนำเข้าไลบรารีของไฟร์เบส
- บรรทัดที่ 2 เป็นการนำเข้าบริการ Cloud Firestore ของไฟร์เบส
- บรรทัดที่ 8 เป็นการนำเข้าโมดูลตั้งค่าที่ได้จากรูปภาพที่ 4.2
- บรรทัดที่ 5 เป็นการส่งออกโมดูลไฟร์เบสเพื่อใช้ในไฟล์อื่น ๆ ซึ่งเมื่อถึงขั้นตอนนี้การเชื่อมต่อบริการไฟร์เบสถือว่าเป็นอันเสร็จ

#### 4.1.2 โครงสร้างของการสร้างหน้าเข้าสู่ระบบ

```

1 <template>
2   <Row type="flex" justify="center" align="middle
   ">
3     <Col span="8" class="col">
4       <Card style="width:400px">
5         <p slot="title">
6           <Icon type="ios-person" size="20"></Icon>
7           sign in
8         </p>
9         <a href="#" slot="extra" @click.prevent="
           SignUp">
10          create account
11        </a>
12        <Form class="form" ref="formInline" :model="
           formInline" :rules="ruleInline">
13          <FormItem prop="email">
14            <Input type="text" v-model="formInline.
              email" placeholder="email">
15              <Icon type="ios-email" slot="prepend"></
              Icon>
16            </Input>
17          </FormItem>
18          <FormItem prop="password">
19            <Input type="password" v-model="
              formInline.password" placeholder="
              password">
20              <Icon type="ios-locked" slot="prepend
              "></Icon>
21            </Input>
22          </FormItem>
23          <FormItem>
24            <Button type="primary" :loading="loading"
              @click="handleSubmit('formInline')">
25              <span v-if="!loading">sign in</span>
26              <span v-else>signing in...</span>
27            </Button>
28          </FormItem>
29        </Form>
30      </Card>
31    </Col>
32  </Row>
33 </template>

```

รูปที่ 4.3: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue

จากภาพที่ 4.4 โครงสร้างของการสร้างหน้าจอส่วติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-33 เป็นเทมเพลตที่ใช้เพื่อสื่อสารกับ Vue.js ให้แปลงข้อมูลดังกล่าวเป็น HTML
- บรรทัดที่ 2-32 เป็นการควบคุมลักษณะการแสดงผลบนหน้าจอ
- บรรทัดที่ 3-31 เป็นการกำหนดขนาดของเนื้อหาภายใน
- บรรทัดที่ 4-30 เป็นการแสดงเนื้อหาในรูปแบบการ์ด (Card)
- บรรทัดที่ 5-8 เป็นส่วนที่ใช้สำหรับกำหนดหัวเรื่องของการ์ด
- บรรทัดที่ 12 เป็นสร้างฟอร์ม (Form)
- บรรทัดที่ 13 เป็นสร้างช่องกรอกข้อมูลอีเมล (e-mail) จากผู้ใช้
- บรรทัดที่ 18 เป็นสร้างช่องกรอกข้อมูลรหัสผ่าน (password) จากผู้ใช้
- บรรทัดที่ 24 สร้างปุ่มเข้าสู่ระบบ

```

1 data () {
2   return {
3     alert: false,
4     formInline: {
5       email: '',
6       password: ''
7     },
8     ruleInline: {
9       email: [
10        { required: true, message: 'please fill email',
11          trigger: 'blur' }
12      ],
13      password: [
14        { required: true, message: 'please fill password',
15          trigger: 'blur' }
16      ]
17    }
18  }
19 }

```

รูปที่ 4.4: การสร้างหน้าจอส่วติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue

จากภาพที่ 4.4 โครงสร้างของการสร้างหน้าจอส่วติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-7 เป็นการสร้างชุดข้อมูลที่ใช้สำหรับการเข้าสู่ระบบ
- บรรทัดที่ 3 ค่าที่ใช้เก็บสถานะของการเข้าสู่ระบบ
- บรรทัดที่ 4-7 เป็นการเก็บข้อมูลอีเมลและรหัสผ่านในรูปแบบ json
- บรรทัดที่ 8-15 เป็นการกำหนดกฏที่ใช้ในการตรวจสอบความถูกต้องของอีเมลและรหัสผ่าน

```

1  userSignIn({commit}, payload) {
2    commit('setLoading', true)
3    firebase.auth().signInWithEmailAndPassword(payload.
      email, payload.password)
4      .then(firebaseUser => {
5        commit('setUser', firebaseUser)
6        commit('setLoading', false)
7        commit('setError', null)
8      })
9    .catch(error => {
10     commit('setError', error.message)
11     commit('setLoading', false)
12   })
13 }

```

รูปที่ 4.5: การสร้างลอจิก (logic) ของหน้าเข้าสู่ระบบ SignIn.vue

จากภาพที่ 4.5 โครงสร้างลอจิกของหน้าเข้าสู่ระบบ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 เป็นการสร้างฟังก์ชันสำหรับรับข้อมูลที่ใช้ในการเข้าสู่ระบบ
- บรรทัดที่ 2 เรียกใช้ฟังก์ชันอื่นเพื่อทำการอัปเดตสถานะการเข้าสู่ระบบ
- บรรทัดที่ 3-12 เป็นการเรียกใช้บริการไฟร์เบส Authentication พร้อมส่งค่า email และ password เพื่อทำการเข้าสู่ระบบ
- บรรทัดที่ 5-7 เป็นการอัปเดตสถานะเมื่อเข้าสู่ระบบสำเร็จ
- บรรทัดที่ 9-12 เป็นการอัปเดตสถานะเมื่อเข้าสู่ระบบไม่สำเร็จ

#### 4.1.3 โครงสร้างของการสร้างหน้าข่าวสาร

```

1 <template>
2 <div style="padding: 16px;">
3 <Row>
4 <Col span="20" style="padding:16px;">
5 <Row v-for=" (post,index) in postsData" :key="post.id
   " style="margin-bottom:16px;">
6 <Card>
7 <p slot="title">
8 <Icon type="social-rss-outline"></Icon>
9 post
10 <span style="font-size:11px; color: #95a5a6;">
11 {{ post.time }}
12 </span>
13 </p>
14 <a href="#" slot="extra" @click.prevent="showData(
   index)">
15 <!-- <Icon type="ios-loop-strong"></Icon> -->
16 detail
17 </a>
18 <p>
19 {{ post.title }}
20 </p>
21 </Card>
22 </Row>
23 </Col>
24 <Col span="4" style="padding:16px;">
25 <Timeline>
26 <TimelineItem v-for=" (event, index) in eventsData" :
   key="index" >
27 <Icon type="trophy" slot="dot" v-if="index == 0"></
   Icon>
28 <p class="time">{{event.time}}</p>
29 <p class="content">{{event.title}}</p>
30 </TimelineItem>
31 </Timeline>
32 </Col>
33 </Row>
34 </div>
35 </template>

```

รูปที่ 4.6: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าข่าวสาร Home.vue

จากภาพที่ 4.6 โครงสร้างของการสร้างหน้าจอสวนติดต่อผู้ใช้ของหน้าข่าวสาร สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-33 เป็นเทมเพลตที่ใช้เพื่อสื่อสารกับ Vue.js ให้แปลงข้อมูลดังกล่าวเป็น HTML
- บรรทัดที่ 3-33 เป็นการควบคุมลักษณะการแสดงผลบนหน้าจอ
- บรรทัดที่ 6-21 เป็นการแสดงเนื้อหาในรูปแบบการ์ด (Card)
- บรรทัดที่ 10-12 เป็นการแสดงเวลาที่ประกาศข่าว
- บรรทัดที่ 18-20 เป็นการแสดงหัวข้อข่าวสาร
- บรรทัดที่ 25-31 เป็นการแสดงปฏิทินกำหนดการขนาดย่อ

```

1  created() {
2    var vm = this;
3    vm.postsData = [];
4    db.collection("Posts")
5      .orderBy("time", "desc")
6      .get()
7      .then(function(querySnapshot) {
8        querySnapshot.forEach(function(doc) {
9          const data = {
10             id: doc.id,
11             title: doc.data().title,
12             description: doc.data().description,
13             time: doc.data().time.toLocaleString(),
14             fileUrl: doc.data().fileURL[0]
15           }
16           if (vm.postsData) {
17             vm.postsData.push(data);
18           }
19         })
20       })
21     }

```

รูปที่ 4.7: การสร้างลอจิก(logic)ของหน้าข่าวสาร Home.vue

จากภาพที่ 4.9 โครงสร้างลอจิกของหน้าข่าวสาร สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 เป็นการฟังก์ชันที่ถูกเรียกทุกครั้งที่ใช้เปิดหน้าข่าวสาร
- บรรทัดที่ 4-20 เรียกใช้บริการ Cloud Firestore เพื่อทำการสืบค้นข้อมูลข่าวสารทั้งหมด



พร้อมทั้งเรียงลำดับตามวันที่ประกาศ

- บรรทัดที่ 9-18 เป็นการเพิ่มข้อมูลเข้าสู่ลิสต์รายการเพื่อใช้ในการแสดงบนหน้าจอ

#### 4.1.4 โครงสร้างของการสร้างหน้าดูรายละเอียดข่าวสาร

```

1 <template>
2   <div>
3     <h2>{{ post.title }}</h2>
4     <p style="font-size:14px;">{{ post.description
5       }}</p>
6     <br>
7     
8   </div>
9 </template>

```

รูปที่ 4.8: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้ารายละเอียดข่าวสาร ViewPost.vue

จากภาพที่ 4.8 โครงสร้างของการสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าดูรายละเอียดข่าวสารสามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-8 เป็นเทมเพลตที่ใช้เพื่อสื่อสารกับ Vue.js ให้แปลงข้อมูลดังกล่าวเป็น HTML
- บรรทัดที่ 2-7 ครอบทับเนื้อหาทั้งหมดเพื่อให้ง่ายต่อการจัดการการแสดงผล
- บรรทัดที่ 3 เป็นการแสดงหัวข้อข่าวสาร
- บรรทัดที่ 4 เป็นการแสดงรายละเอียดข่าวสาร
- บรรทัดที่ 6 เป็นการแสดงไฟล์แนบ

```

1 created() {
2   let id = this.$route.params.id
3   db.collection('Posts').doc(id).get().then((doc) =>
4     {
5       if(doc.exists){
6         this.post = doc.data()
7       }
8     })
9 }

```

รูปที่ 4.9: การสร้างลอจิกของหน้าดูรายละเอียดของข่าวสาร ViewPost.vue

จากภาพที่ 4.9 โครงสร้างลอจิกของหน้าดูรายละเอียดของข่าวสาร สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 เป็นการฟังก์ชันที่ถูกเรียกทุกครั้งที่ใช้เปิดหน้าดูรายละเอียดข่าวสาร
- บรรทัดที่ 2 ดึงค่าไอดีของประกาศที่ถูกส่งมาจากหน้าแสดงข่าวสาร
- บรรทัดที่ 3 ทำการสืบค้นข้อมูลข่าวสารจาก Cloud Firestore จากไอดีของประกาศ
- บรรทัดที่ 4-6 เป็นการตรวจสอบว่ามีประกาศดังกล่าวอยู่ในฐานข้อมูล Cloud Firestore หรือไม่

## 4.1.5 โครงสร้างของการสร้างหน้าสนทนา

```

1 <template>
2 <Row :gutter="0" type="flex" justify="center" align="
  middle">
3 <Col span="16" style="padding: 0px;">
4 <Card style="min-height: 500px;max-height: 500px;" :
  padding="0">
5 <p v-if="!isAdmin" slot="title" style="text-align:
  center;">
6 ESP
7 </p>
8 <p v-else slot="title" style="text-align:center;">
9 {{ chatTitle }}
10 </p>
11 <Scroll style="background-color: #EEEEEE;">
12 <ul style="padding:6px;padding-right:8px;">
13 <li v-for="(item,index) in messages" :key="index"
  style="margin-bottom:8px;">
14 <Card v-if="user.uid !== item.id" :padding="6" style
  ="text-align:left;display: inline-block;
  background-color: #FAFAFA;">
15 <div>
16 <p>{{ item.message }}</p>
17 </div>
18 </Card>
19 <div v-else style="text-align:right;">
20 <Card :padding="6" style="display: inline-block;
  background-color: #B2E281;">
21 <p>{{ item.message }}</p>
22 </Card>
23 </div>
24 </li>
25 </ul>
26 </Scroll>
27 <div style="padding:16px;text-align:right;" >
28 <Input type="textarea" v-model="formItem.message"
  placeholder="type..." v-on:keyup.enter="send"></
  Input>
29 <Button :loading="loading" type="primary" style="
  margin-top:10px;" size="large" @click="send">send
  </Button>
30 </div>
31 </Card>
32 </Col>
33 </Row>
34 </template>

```

รูปที่ 4.10: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสนทนา Message.vue

จากภาพที่ 4.10 โครงสร้างของการสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสนทนา สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-34 เป็นเทมเพลตที่ใช้เพื่อสื่อสารกับ Vue.js ให้แปลงข้อมูลดังกล่าวเป็น HTML
- บรรทัดที่ 4-22 แสดงหน้าต่างสนทนา
- บรรทัดที่ 9 แสดงชื่อคู่สนทนา
- บรรทัดที่ 11-26 แสดงข้อความสนทนา
- บรรทัดที่ 28 แสดงช่องกรอกข้อความสนทนา
- บรรทัดที่ 29 แสดงปุ่มกดส่งข้อความ

```

1  send() {
2    var vm = this;
3    vm.loading = true;
4    this.formItem.time = new Date();
5
6    // check where data shulde update
7    let key = "";
8    if (this.isAdmin) {
9      key = this.chatId;
10     if (this.chatTitle === "" || this.chatTitle ===
        null) {
11       return;
12     }
13   } else {
14     key = this.formItem.senderId;
15   }
16
17   this.formItem.name = this.user.displayName;
18   this.formItem.photo = this.user.photoURL
19
20   db
21     .collection("Chats")
22     .doc(key)
23     .collection("messages")
24     .add(this.formItem)
25     .then(function(docRef) {
26       vm.loading = false;
27       vm.formItem.message = "";
28       db
29         .collection("Users")
30         .doc(key)
31         .update({ lastChat: new Date() })
32         .catch(function(error) {
33           vm.$Message.error("send message fail");
34           vm.loading = false;
35         });
36     });
37 }

```

รูปที่ 4.11: การสร้างลอจิกของหน้าสนทนา Message.vue

จากภาพที่ 4.11 โครงสร้างลอจิกของหน้าสนทนา สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 ชื่อฟังก์ชัน
- บรรทัดที่ 2 ตรวจสอบไอดีของผู้ใช้
- บรรทัดที่ 3 ดึงค่าโปรไฟล์ผู้ใช้คนปัจจุบัน
- บรรทัดที่ 4-6 เขียนข้อมูลลงฐานข้อมูล Cloud Firestore โดยระบุ path ที่จะทำการจัดเก็บชุดข้อมูล
- บรรทัดที่ 28-35 อัปเดตข้อมูลเวลาสนทnal่าสุดของผู้ใช้
- บรรทัดที่ 33 แสดงสถานะการอัปเดตข้อมูล

#### 4.1.6 โครงสร้างของการสร้างหน้าปฏิทินแสดงกำหนดการ

```

1 <template>
2   <div>
3     <Row :gutter="16">
4       <Col span="6">
5         <Card>
6           <h3>search</h3>
7           <DatePicker v-model="filterDate" format="d-M-yyyy
              " type="date" size="large" placeholder="
              select date" style="margin-top:6px;"></
              DatePicker>
8         </Card>
9       </Col>
10      <Col span="18">
11        <h3>Schedule</h3>
12        <Table border
13          :loading="dataLoading"
14          :columns="columnsName"
15          :data="eventsData"
16          no-data-text="no schedule"
17          style="margin-top:6px;">
18        </Table>
19      </Col>
20    </Row>
21  </div>
22 </template>

```

รูปที่ 4.12: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าปฏิทินกำหนดการ Schedule.vue

จากภาพที่ 4.12 โครงสร้างของการสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าปฏิทินกำหนดการ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-22 เป็นเทมเพลตที่ใช้เพื่อสื่อสารกับ Vue.js ให้แปลงข้อมูลดังกล่าวเป็น HTML
- บรรทัดที่ 4-9 แสดงหน้าต่างเลือกวันที่เพื่อค้นหา
- บรรทัดที่ 11 แสดงชื่อตาราง
- บรรทัดที่ 12-18 แสดงตารางกำหนดการ

```

1  created() {
2    var vm = this;
3    db.collection("Events")
4      .orderBy("time")
5      .onSnapshot(function(querySnapshot) {
6        vm.dataLoading = false;
7        vm.eventsData = [];
8        querySnapshot.forEach((doc) => {
9          const data = {
10             'id': doc.id,
11             'title': doc.data().title,
12             'description': doc.data().description,
13             'time': `${doc.data().time.getDate()}-${doc.data()
14               .time.getMonth()}-${doc.data().time.
15                 getFullYear()}`
16           }
17           vm.eventsData.push(data)
18         })
19       })
20     }

```

รูปที่ 4.13: การสร้างลอจิกของหน้าปฏิทินกำหนดการ Schedule.vue

จากภาพที่ 4.13 โครงสร้างลอจิกของหน้าปฏิทินกำหนดการ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-18 ชื่อฟังก์ชันที่จะถูกเรียกทุกครั้งที่หน้าปฏิทินกำหนดการถูกเปิด
- บรรทัดที่ 3-17 สืบค้นกำหนดการจากฐานข้อมูลโดยมีการเรียงลำดับจากวันที่ล่าสุดไปยังวันที่ก่อนหน้า
- บรรทัดที่ 9-15 จัดเก็บข้อมูลที่สืบค้นได้ เพื่อใช้ในการแสดงผลบนหน้าจอ

## 4.1.7 โครงสร้างของการสร้างหน้าสร้างประกาศ

```

1 <Modal v-model="modalNewPost"
2 title="add post">
3   <Form :model="formItem" :label-width="80">
4     <FormItem label="title">
5       <Input v-model="formItem.title" placeholder="
        Enter something..."></Input>
6     </FormItem>
7     <FormItem label="detail">
8       <Input v-model="formItem.description" type="
        textarea" :autosize="{minRows: 2,maxRows: 5}"
        placeholder="Enter something..."></Input>
9     </FormItem>
10    <FormItem label="target">
11      <Select v-model="formItem.collection">
12        <Option value="public">public</Option>
13        <Option value="group">group</Option>
14        <Option value="volunteer">volunteer</Option>
15      </Select>
16    </FormItem>
17    <FormItem label="contact list" v-if="tags.length >
        0">
18      <Tag closable color="blue" v-for="tag in tags" :
        key="tag" @on-close="handleClose"> {{ tag }}
        </Tag>
19    </FormItem>
20    <FormItem label="file">
21      <Upload :before-upload="handleUpload"
22        action="https://shielded-earth-61349.herokuapp.
        com/">
23        <Button :type="btnAddPostType" icon="ios-cloud-
        upload-outline"> {{ uploadBtnTile }} </Button>
24      </Upload>
25    </FormItem>
26  </Form>
27  <div slot="footer">
28    <Button type="primary" :loading="loading" @click="
        newPost">save</Button>
29    <Button type="ghost" style="margin-left: 8px"
        @click="modalNewPost = !modalNewPost">cancel</
        Button>
30  </div>
31 </Modal>

```

รูปที่ 4.14: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างประกาศ MgPost.vue



จากภาพที่ 4.8 โครงสร้างของการสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างประกาศ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-31 เนื่องจากต้องการให้แสดงหน้าเพิ่มข่าวสารเป็นป๊อปอัพ (Pop up) จึงใช้แท็ก(tag) <Model></Model>
- บรรทัดที่ 2-7 ครอบทับเนื้อหาทั้งหมดเพื่อให้ง่ายต่อการจัดการการแสดงผล
- บรรทัดที่ 3-26 เป็นการสร้างฟอร์มรับข้อมูล
- บรรทัดที่ 4-25 เป็นสร้างช่องรับข้อมูลประกาศ
- บรรทัดที่ 5 เป็นการรับค่าหัวเรื่องประกาศ
- บรรทัดที่ 8 เป็นการรับค่ารายละเอียดประกาศ
- บรรทัดที่ 11-15 เป็นการรับค่ากลุ่มเป้าหมายของประกาศนั้นๆ
- บรรทัดที่ 21-24 เป็นการสร้างปุ่มอัปโหลดไฟล์
- บรรทัดที่ 28 เป็นการสร้างปุ่มบันทึกประกาศ
- บรรทัดที่ 29 เป็นการสร้างปุ่มยกเลิกประกาศ

```

1 newPost() {
2   let key = '';
3   var vm = this;
4   vm.formItem.time = new Date();
5   vm.formItem.tags = vm.tags
6   db.collection("Posts").add(this.formItem)
7     .then(function(docRef) {
8     key = docRef.id;
9     if(vm.file != null){
10      let storageRef = storage.ref('Posts/'+key);
11      let fileRef = storageRef.child(vm.file.name+"");
12      fileRef.put(vm.file).then(function(snapshot) {
13      db.collection("Posts").doc(key)
14      .update(
15      {
16      fileURL: snapshot.metadata.downloadURLs[0],
17      fileType: vm.file.name.slice(vm.file.name.
18        lastIndexOf('.') + "",
19      fileName: vm.file.name
20      })
21      ).then(() => {
22      vm.$Notice.success({
23      title: 'Your post was created',
24      desc: ''
25      });
26      })
27      .catch(function(error) {
28      vm.$Notice.warning({
29      title: 'create post fail',
30      desc: ''
31      });
32      })
33    }
34  }

```

รูปที่ 4.15: การสร้างลอจิกของหน้าสร้างประกาศ MgPost.vue

จากภาพที่ 4.15 โครงสร้างลอจิกของหน้าสร้างประกาศ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1 ชื่อฟังก์ชัน
- บรรทัดที่ 4-5 ข้อมูลของประกาศ
- บรรทัดที่ 6-30 เรียกใช้งาน Cloud Firestore เพื่อบันทึกประกาศลงฐานข้อมูล
- บรรทัดที่ 10-24 เป็นส่วนที่ใช้ในการอัปโหลดเอกสารแนบไปยังไฟร์เบส Storage

- บรรทัดที่ 13-19 เป็นการอัปเดตข้อมูล URL ที่ได้จากการอัปโหลดไฟล์แนบเข้าสู่ฐานข้อมูล Cloud Firestore

#### 4.1.8 โครงสร้างของการสร้างหน้าอัปโหลดเอกสารที่เกี่ยวข้อง

```

1  <Modal v-model="modalNewDoc"
2  title="Upload File">
3    <Form :model="formItem" :label-width="80">
4      <FormItem label="Document title">
5        <Input v-model="formItem.title"></Input>
6      </FormItem>
7      <FormItem label="Document detail">
8        <Input v-model="formItem.description" type="
          textarea" :autosize="{minRows: 2,maxRows:
            5}"></Input>
9      </FormItem>
10     <FormItem label="Select file">
11       <Upload
12         :before-upload="handleUpload"
13         action="https://shielded-earth-61349.herokuapp.
            com/">
14         <Button :type="type" icon="ios-cloud-upload-
            outline"> {{ uploadBtnTile }} </Button>
15       </Upload>
16     </FormItem>
17   </Form>
18   <div slot="footer">
19     <Button type="primary" :loading="loading" @click
        ="newDoc">upload</Button>
20     <Button type="ghost" style="margin-left: 8px"
        @click="modalNewDoc = !modalNewDoc">cancel</
        Button>
21   </div>
22 </Modal>

```

รูป ที่ 4.16: การ สร้าง หน้า จอ ส่วน ติดต่ อ ผู้ ใ้ ของ หน้า อัปโหลด เอกสาร ที่ เกี่ยวข้ อง MgDocument.vue

จากภาพที่ 4.16 โครงสร้างของการสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าอัปโหลดเอกสารที่เกี่ยวข้อง สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-22 สร้างป๊อปอัพแสดงหน้าอัปโหลดไฟล์เอกสาร
- บรรทัดที่ 3-17 สร้างฟอร์ม
- บรรทัดที่ 11-15 สร้างปุ่มอัปโหลดเอกสาร
- บรรทัดที่ 19 สร้างปุ่มบันทึกเอกสาร
- บรรทัดที่ 20 สร้างปุ่มยกเลิกการอัปโหลดเอกสาร

```

1 newDoc() {
2   var vm = this;
3   vm.loading = true;
4   let key = '';
5   this.formItem.time = new Date();
6   db.collection("Docs").add(this.formItem)
7   .then(function(docRef) {
8     key = docRef.id;
9     if(vm.file != null){
10      let storageRef = storage.ref('Docs/'+key);
11      let fileRef = storageRef.child(vm.file.name+"");
12      fileRef.put(vm.file).then(function(snapshot) {
13        db.collection("Docs").doc(key).update({
14          fileURL: snapshot.metadata.downloadURLs[0],
15          fileType: vm.file.name.slice(vm.file.name.
16            lastIndexOf('.')+1)
17        }).then(() => {
18          vm.$Notice.success({
19            title: 'Success',
20            desc: ''
21          });
22        }).catch(function(error) {
23          vm.$Notice.warning({
24            title: 'Fail',
25            desc:''
26          });
27        });
28      });
29    }
30  });
31 }

```

รูปที่ 4.17: การสร้างลอจิกของหน้าอัปโหลดเอกสารที่เกี่ยวข้อง MgDocument.vue

จากภาพที่ 4.17 โครงสร้างลอจิกของหน้าสร้างประกาศ สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 6-26 เป็นการอัปเดตข้อมูลเอกสารเข้าสู่ฐานข้อมูล Cloud Firestore

- บรรทัดที่ 10-12 เป็นการเรียกใช้ไฟร์เบส Storage เพื่อทำการอัปโหลดไฟล์เอกสาร
- บรรทัดที่ 13-20 ใช้ในการอัปเดตข้อมูล URL ไปยังฐานข้อมูล Cloude Firestore

#### 4.1.9 โครงสร้างของการสร้างหน้าสร้างกำหนดการจ้องคิวส่งเอกสาร

```

1 <Modal v-model="modalNewQueue">
2   <p slot="header" style="color:#3498db;text-align:
   center">
3     <Icon type="information-circled"></Icon>
4     <span>Create submit document date</span>
5   </p>
6   <div>
7     <Form :model="formItem" :label-width="80">
8       <FormItem label="title">
9         <Input v-model="formItem.title"></Input>
10      </FormItem>
11      <FormItem label="date">
12        <DatePicker v-model="formItem.date" format="d-
          MMMM-yyyy" type="daterange" placement="bottom-
          end" placeholder="select" style="width: 200px
          "></DatePicker>
13      </FormItem>
14      <FormItem label="time">
15        <TimePicker v-model="formItem.time" format="HH:mm
          " type="timerange" placement="bottom-end"
          placeholder="select" style="width: 200px"></
          TimePicker>
16      </FormItem>
17      <FormItem label="students per hr">
18        <InputNumber :max="100" :min="1" v-model="
          formItem.count" style="width: 200px"></
          InputNumber>
19      </FormItem>
20    </Form>
21  </div>
22  <div slot="footer">
23    <Button type="success" size="large" long @click="
      saveQueue">save</Button>
24  </div>
25 </Modal>

```

รูปที่ 4.18: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างกำหนดการจ้องคิวส่งเอกสาร  
MgQueue.vue

จากภาพที่ 4.18 โครงสร้างของการสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างกำหนดการ จอควิส่งเอกสาร สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 1-25 การสร้างหน้าต่างป๊อปอัพ
- บรรทัดที่ 4 ชื่อหน้าต่างป๊อปอัพ
- บรรทัดที่ 7-20 เป็นการสร้างฟอร์ม
- บรรทัดที่ 23 เป็นการสร้างปุ่มบันทึกกำหนดการ

```

1 saveQueue() {
2   var vm = this;
3   db.collection("Queue").add(this.formItem)
4   .then(function(docRef) {
5     vm.modalNewQueue = false;
6     vm.$Notice.success({
7       title: 'success',
8       desc: ''
9     });
10  }) .catch(function() {
11    vm.modalNewQueue = false;
12    vm.$Notice.warning({
13      title: 'fial',
14      desc: ''
15    });
16  })
17 }
```

รูปที่ 4.19: การสร้างลอจิกของหน้าสร้างกำหนดการจอควิส่งเอกสาร MgQueue.vue

จากภาพที่ 4.19 โครงสร้างลอจิกของหน้าสร้างกำหนดการจอควิส่งเอกสาร สามารถอธิบายการทำงานได้ดังนี้

- บรรทัดที่ 3-15 ที่ได้จากการอัปโหลดไฟล์แนบเข้าสู่ฐานข้อมูล Cloud Firestore

## 4.2 การพัฒนาเว็บแอปพลิเคชันแอนดรอยด์แอปพลิเคชัน

การพัฒนาเว็บแอปพลิเคชันแอนดรอยด์แอปพลิเคชันระบบกองทุนเงินให้กู้ยืมเพื่อการศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี มีวัตถุประสงค์หลักเพื่ออำนวยความสะดวกต่อนักศึกษาที่โดยส่วนใหญ่ใช้อุปกรณ์พกพาและต้องการใช้งานฮาร์ดแวร์(Hardware)เช่น กล้องที่ใช้ใน

การถ่ายภาพสำเนาเอกสาร เป็นต้น

#### 4.2.1 โครงสร้างของการสร้างหน้า MainActivity

```

1 private FirebaseAuth mAuth;
2 private FeedFragment feedFrag = FeedFragment.
   newInstance();
3 private ChatFragment chatFrag = ChatFragment.
   newInstance();
4 private DocumentsFragment docFrag =
   DocumentsFragment.newInstance();
5 private ScheduleFragment sheduleFrag =
   ScheduleFragment.newInstance();
6 private SubmitFragment submitFrag = SubmitFragment.
   newInstance();
7 private UserChatFragment userChatFrag =
   UserChatFragment.newInstance();
8 private CheckinFragment checkinFrag =
   CheckinFragment.newInstance();

```

รูปที่ 4.20: ตัวแปรในคลาส MainActivity

จากภาพที่ 4.20 ตัวแปรที่ประกาศขึ้นเพื่อใช้ในการทำงานของคลาส MainActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ตัวแปร mAuth ใช้ในการจัดเก็บสถานะและข้อมูลของผู้ใช้
- บรรทัดที่ 2 ตัวแปร feedFrag ใช้แสดงผลหน้าจอลำรายชื่อเอกสาร
- บรรทัดที่ 3 ตัวแปร chatFrag ใช้แสดงผลหน้าจอสนทนาสำหรับเจ้าหน้าที่
- บรรทัดที่ 4 ตัวแปร chatFrag ใช้แสดงผลหน้าจอเอกสารที่เกี่ยวข้อง
- บรรทัดที่ 5 ตัวแปร sheduleFrag ใช้แสดงผลหน้าจอปฏิทินกำหนดการ
- บรรทัดที่ 6 ตัวแปร submitFrag ใช้แสดงผลหน้าจอส่งสำเนาเอกสาร
- บรรทัดที่ 7 ตัวแปร userChatFrag ใช้แสดงผลหน้าจอสนทนาสำหรับนักศึกษา
- บรรทัดที่ 8 ตัวแปร checkinFrag ใช้แสดงผลหน้าจอลงชื่อเอกสาร



```

1 new DrawerBuilder()
2 .addDrawerItems(
3     feed, chat, event, doc, submit, checkin, faq,
4     about, setting, account, logout
5 ).withOnDrawerItemClickListener(new Drawer.
6     OnDrawerItemClickListener() {
7 @Override
8     public boolean onItemClick(View view, int position
9         , IDrawerItem drawerItem) {
10         long id = drawerItem.getIdentifier();
11         if (id == 1) {
12             getSupportFragmentManager().beginTransaction()
13                 .replace(R.id.contentContainer, feedFrag)
14                 .commit();
15         } else if (id == 2) {
16             if (currentUser.getEmail().contains("tapgabee"))
17             {
18                 getSupportFragmentManager().beginTransaction()
19                     .replace(R.id.contentContainer, chatFrag)
20                     .commit();
21             } else {
22                 getSupportFragmentManager().beginTransaction()
23                     .replace(R.id.contentContainer, userChatFrag)
24                     .commit();
25             }
26         } else if (id == 3) {
27             getSupportFragmentManager().beginTransaction()
28                 .replace(R.id.contentContainer, sheduleFrag)
29                 .commit();
30         } else if (id == 4) {
31             getSupportFragmentManager().beginTransaction()
32                 .replace(R.id.contentContainer, docFrag)
33                 .commit();
34         } else if (id == 5) {
35             getSupportFragmentManager().beginTransaction()
36                 .replace(R.id.contentContainer, submitFrag)
37                 .commit();
38         } else if (id == 6) {
39             getSupportFragmentManager().beginTransaction()
40                 .replace(R.id.contentContainer, checkinFrag)
41                 .commit();
42         } else if (id == 11) {
43             mAuth.signOut();
44             startActivity(new Intent(MainActivity.this,
45                 MainActivity.class));
46             finish();
47         }
48     }
49 }

```

รูปที่ 4.21: โค้ดส่วนที่ใช้ในการสร้างเมนูนำทางหลักภายในคลาส MainActivity

จากภาพที่ 4.21 สามารถอธิบายการทำงานโค้ดส่วนที่ใช้ในการสร้างเมนูนำทางหลักภายในคลาส MainActivity ได้ดังนี้

- บรรทัดที่ 1 เป็นการสร้างเมนูนำทาง
- บรรทัดที่ 2-3 เป็นการเพิ่ม Fragment ต่างๆ เข้าไปยังเมนูนำทาง
- บรรทัดที่ 4-6 เป็นการเพิ่มการดักจับอีเวนต์ (Event) เพื่อสลับหน้าจอการแสดงผลที่เกิดขึ้นเมื่อผู้ใช้กดที่เมนูนำทาง
- บรรทัดที่ 8-11 เป็นการแสดงผลหน้าข่าวสาร
- บรรทัดที่ 13-16 เป็นการแสดงผลหน้าสนทนาสำหรับเจ้าหน้าที่
- บรรทัดที่ 17-20 เป็นการแสดงผลหน้าสนทนาสำหรับนักศึกษา
- บรรทัดที่ 22-25 เป็นการแสดงผลหน้าปฏิทินกำหนดการ
- บรรทัดที่ 26-29 เป็นการแสดงผลหน้าดาวนโหลดเอกสาร
- บรรทัดที่ 30-33 เป็นการแสดงผลหน้าส่งสำเนาเอกสาร
- บรรทัดที่ 34-37 เป็นการแสดงผลหน้าจอควิสงเอกสาร
- บรรทัดที่ 38-41 เป็นการรีเฟรช(refresh)หน้าจอเมื่อผู้ใช้กดปุ่มออกจากระบบ

#### 4.2.2 โครงสร้างของการสร้างหน้า FeedFragment

```
1 private RecyclerView recyclerView;
2 private FirebaseFirestore db;
3 private ArrayList<Post> posts;
4 private FeedItemAdapter adapter;
```

รูปที่ 4.22: ตัวแปรในคลาส FeedFragment

จากภาพที่ 4.22 ตัวแปรที่ประกาศขึ้นเพื่อใช้ในการทำงานของคลาส FeedFragment สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ตัวแปร recyclerView ใช้ในการแสดงข้อมูลลิสรายการข่าวสาร
- บรรทัดที่ 2 ตัวแปร db ใช้ในการสืบค้นข้อมูลจากข่าวสารจาก Cloud Firestore
- บรรทัดที่ 3 ตัวแปร posts ใช้ในการเก็บชุดข้อมูลที่ได้จากการสืบค้นข้อมูล
- บรรทัดที่ 4 ตัวแปร adapter ใช้ในการแปลงชุดข้อมูลเป็นลิสรายการเพื่อแสดงบน recyclerView

```

1 db.collection("Posts")
2 .orderBy(getString(R.string.key_time), Query.
   Direction.DESENDING)
3 .get()
4 .addOnCompleteListener(new OnCompleteListener<
   QuerySnapshot>() {
5     @Override
6     public void onComplete(@NonNull Task<QuerySnapshot
       > task) {
7         if (task.isSuccessful() && isAdded()) {
8             for (DocumentSnapshot document : task.getResult
               ()) {
9                 Log.d(TAG, document.getId() + " => " +
                   document.getData());
10                Map<String, Object> data = document.getData();
11                Post post = new Post();
12                post.setTitle(data.get(getString(R.string.
                   key_title)).toString());
13                post.setCollection(data.get(getString(R.string
                   .key_collection)).toString());
14                post.setDate((Date) data.get(getString(R.string
                   .key_time)));
15                post.setDescription(data.get(getString(R.
                   string.key_description)) == null ? "" :
                   data.get(getString(R.string.key_description
                   )).toString());
16                post.setFileURL(data.get(getString(R.string.
                   key_fileURL)) == null ? "" : data.get(
                   getString(R.string.key_fileURL)).toString())
                   ;
17                post.setFileName(data.get(getString(R.string.
                   key_fileName)) == null ? "" : data.get(
                   getString(R.string.key_fileName)).toString()
                   );
18                posts.add(post);
19            }
20            recyclerView.setLayoutManager(new
               LinearLayoutManager(getActivity()));
21            recyclerView.setAdapter(adapter);
22            adapter.notifyDataSetChanged();
23        } else {
24            Log.w(TAG, "Error getting documents.", task.
               getException());
25        }
26    }
27 });

```

รูปที่ 4.23: โค้ดส่วน ที่ ใช้ ในการ สืบค้น ข้อมูล จาก Cloude Firestore ภายใน คลาส FeedFragment

จากภาพที่ 4.23 โค้ดส่วนที่ใช้ในการสืบค้นข้อมูลจาก Cloud Firestore ภายในคลาส FeedFragment สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1-3 เริ่มทำการสืบค้นข้อมูลประกาศทั้งหมดพร้อมทั้งเรียงลำดับข้อมูลจากประกาศล่าสุดก่อน
- บรรทัดที่ 4-18 รับผลการสืบค้นพร้อมทั้งเพิ่มข้อมูลที่แต่ละแถวเข้าไว้ที่ตัวแปร posts
- บรรทัดที่ 20-22 ทำการอัปเดตข้อมูลที่แสดงอยู่บน recyclerView

```

1  @Override
2  public void recyclerViewListClicked(View v, int
    position) {
3      Intent intent = new Intent(getActivity(),
        PostDetailActivity.class);
4      intent.putExtra(getString(R.string.key_title),
        posts.get(position).getTitle());
5      intent.putExtra(getString(R.string.key_collection)
        , posts.get(position).getCollection());
6      intent.putExtra(getString(R.string.key_time),
        posts.get(position).getDate());
7      intent.putExtra(getString(R.string.key_description)
        ), posts.get(position).getDescription());
8      intent.putExtra(getString(R.string.key_fileURL),
        posts.get(position).getFileURL());
9      intent.putExtra(getString(R.string.key_fileName),
        posts.get(position).getFileName());
10     if (getActivity() != null)
11         getActivity().startActivity(intent);
12 }

```

รูปที่ 4.24: โค้ดส่วนที่ใช้ในการดักอีเวนต์เมื่อผู้ใช้กดที่แถวประกาศในคลาส FeedFragment

จากภาพที่ 4.24 โค้ดส่วนที่ใช้ในการดักอีเวนต์เมื่อผู้ใช้กดที่แถวประกาศในคลาส FeedFragment สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ประกาศตัวแปร intent ประเภทตัวแปร Intent เพื่อใช้กำหนดแอคทีวิตี้ปลายทางซึ่งในที่นี้คือ PostDetailActivity
- บรรทัดที่ 4-9 เป็นการเพิ่มข้อมูลเข้าเก็บไว้ที่ตัวแปร intent โดยดึงข้อมูลใน posts มาจากตำแหน่งแถวที่ถูกผู้ใช้กด

- บรรทัดที่ 11 เริ่มการทำงานของแอกทิวิตี้ PostDetailActivity

#### 4.2.3 โครงสร้างของการสร้างหน้า PostDetailActivity

```

1 private TextView tvTitle, tvDescription,
   tvCollection, tvDate;
2 private String strTitle, strDescription, strDate,
   strCollection, strFileURL, strFileName;
3 private FloatingActionButton fab;
4 private DownloadManager downloadManager;
5
6 strTitle = getIntent().getStringExtra(getString(R.
   string.key_title));
7 strDescription = getIntent().getStringExtra(
   getString(R.string.key_description));
8 strDate = getIntent().getStringExtra(getString(R.
   string.key_time));
9 strCollection = getIntent().getStringExtra(getString
   (R.string.key_collection));
10 strFileURL = getIntent().getStringExtra(getString(R.
   string.key_fileURL));
11 strFileName = getIntent().getStringExtra(getString(R
   .string.key_fileName));
12 tvTitle.setText(strTitle);
13 tvDescription.setText(strDescription);
14 tvDate.setText(strDate);
15 tvCollection.setText(strCollection);

```

รูปที่ 4.25: โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส PostDetailActivity

จากภาพที่ 4.25 โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส PostDetailActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1-4 เป็นการประกาศตัวแปรที่ใช้ในการเก็บข้อมูลประกาศ
- บรรทัดที่ 6-11 เป็นการดึงค่าที่ถูกส่งมาจากคลาส FeedFragment ผ่านทาง Intent
- บรรทัดที่ 12-15 เป็นการแสดงผลข้อมูลต่างๆ ออกทางหน้าจอแสดงผล

```

1  @Override
2  public void onClick(View v) {
3      int id = v.getId();
4      if (id == R.id.fab) {
5          downloadManager = (DownloadManager)
6              getSystemService(Context.DOWNLOAD_SERVICE);
7          Uri uri = Uri.parse(strFileURL);
8          DownloadManager.Request request = new
9              DownloadManager.Request(uri);
10         request.setNotificationVisibility(DownloadManager.
11             Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED);
12         request.setDestinationInExternalPublicDir(
13             Environment.DIRECTORY_DOWNLOADS, strFileName);
14         downloadManager.enqueue(request);
15     }
16 }

```

รูปที่ 4.26: โค้ดส่วนที่ใช้ในการดาวน์โหลดเอกสารของคลาส PostDetailActivity

จากภาพที่ 4.26 โค้ดส่วนที่ใช้ในการดาวน์โหลดเอกสารของคลาส PostDetailActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1-2 เช็คว่าอีเวนต์ที่เกิดขึ้นมาจากปุ่มดาวน์โหลดเอกสารหรือไม่
- บรรทัดที่ 5-9 เป็นการเตรียมความพร้อมสำหรับการดาวน์โหลดเอกสาร
- บรรทัดที่ 10 ทำการเริ่มดาวน์โหลดเอกสาร

#### 4.2.4 โครงสร้างของการสร้างหน้า ChatActivity

```

1 private FirebaseFirestore db;
2 private MessagesListAdapter adapter;
3 private MessagesList messagesList;
4 private String senderId;
5 private String name;
6 private String avatar;
7 private FirebaseAuth mAuth;
8 private EditText tvMessage;
9 private FirebaseUser currentUser;
10 private Button btnSend;

```

รูปที่ 4.27: โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส ChatActivity

จากภาพที่ 4.27 โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส ChatActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ตัวแปร db ใช้ในการสืบค้นข้อมูลจาก Cloud Firestore
- บรรทัดที่ 2 ตัวแปร adapter ใช้ในการแปลงชุดข้อมูลที่ได้จากการสืบค้นเป็นลิสต์รายการ
- บรรทัดที่ 3 ตัวแปร messagesList ใช้ในการแสดงบทสนทนา
- บรรทัดที่ 4 ตัวแปร senderId ใช้ในการจัดเก็บไอดีของผู้ใช้
- บรรทัดที่ 5 ตัวแปร name ใช้ในการจัดเก็บชื่อของผู้ใช้
- บรรทัดที่ 6 ตัวแปร avatar ใช้ในการจัดเก็บ url รูปภาพของผู้ใช้
- บรรทัดที่ 7 ตัวแปร mAuth ใช้ในการจัดเก็บสถานะและข้อมูลของผู้ใช้
- บรรทัดที่ 8 ตัวแปร tvMessage ใช้ในการกรอกข้อความ
- บรรทัดที่ 9 ตัวแปร currentUser ใช้ในการจัดเก็บสถานะและข้อมูลของผู้ใช้คนปัจจุบัน
- บรรทัดที่ 10 ตัวแปร btnSend เป็นปุ่มที่ใช้สำหรับกดส่งข้อความ

```

1 db.collection("Chats").document(key).collection("
    messages")
2 .orderBy("time")
3 .addSnapshotListener(new ValueEventListener<QuerySnapshot
    >() {
4 @Override
5 public void onEvent(QuerySnapshot documentSnapshots,
    FirebaseFirestoreException e) {
6     if (e != null) {
7         System.err.println("Listen failed:" + e);
8         return;
9     }
10
11     String id;
12     String usrId;
13     String text;
14     Date createdAt;
15     adapter.clear();
16     for (DocumentSnapshot document : documentSnapshots
17         ) {
18         Map<String, Object> data = document.getData();
19         id = document.getId();
20         text = data.get("message").toString();
21         createdAt = (Date) data.get("time");
22         usrId = data.get("senderId").toString();
23         Message m = new Message(id, text, createdAt, new
24             User(usrId, name, avatar));
25         adapter.addToStart(m, true);
26         adapter.notifyDataSetChanged();
27     }
28 );

```

รูปที่ 4.28: โค้ดส่วนที่ใช้ในการสืบค้นประวัติการสนทนาของคลาส ChatActivity

จากภาพที่ 4.28 โค้ดส่วนที่ใช้ในการสืบค้นประวัติการสนทนาของคลาส ChatActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1-2 ทำการสืบค้นข้อมูลประวัติการสนทนาจาก Cloud Firestore
- บรรทัดที่ 11-14 สร้างตัวแปรเพื่อใช้ในการจัดเก็บข้อมูล



- บรรทัดที่ 16-24 ทำการอ่านค่าและเพิ่มเข้าลิสต์รายการ

```

1 final Map<String, Object> map = new HashMap<>();
2 map.put("message", tvMessage.getText().toString());
3 map.put("time", new Date());
4 map.put("senderId", currentUser.getId());
5 map.put("name", currentUser.getDisplayName());
6 map.put("photo", currentUser.getPhotoUrl().toString
  ());
7 tvMessage.setText("");
8 db.collection("Chats").document(senderId)
9 .collection("messages")
10 .add(map)
11 .addOnCompleteListener(new OnCompleteListener<
  DocumentReference>() {
12   @Override
13   public void onComplete(@NonNull Task<
  DocumentReference> task) {
14     Map<String, Object> map1 = new HashMap<>();
15     map1.put("lastChat", new Date());
16     db.collection("Users")
17     .document(senderId)
18     .update(map1);
19   }
20 }
21 );

```

รูปที่ 4.29: โค้ดส่วนที่ใช้ในการส่งข้อความของคลาส ChatActivity

จากภาพที่ 4.29 โค้ดส่วนที่ใช้ในการส่งข้อความของคลาส ChatActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1-6 สร้าง HashMap เพื่อใช้จัดเก็บข้อความ
- บรรทัดที่ 8-11 เป็นการเพิ่มชุดข้อมูลเข้าสู่ Cloud Firestore
- บรรทัดที่ 14-18 เป็นการอัปเดตเวลาสนทนาล่าสุดของผู้ใช้

#### 4.2.5 โครงสร้างของการสร้างหน้า SignInActivity

```
1 private FirebaseAuth mAuth;  
2 private String email, password;  
3 private EditText userName, userPassword;  
4 private ProgressBar simpleProgressBar;
```

รูปที่ 4.30: โค้ดส่วนที่ใช้ในการแสดงผลหน้าเข้าสู่ระบบของคลาส SignInActivity

จากภาพที่ 4.30 โค้ดส่วนที่ใช้ในการแสดงผลหน้าเข้าสู่ระบบของคลาส SignInActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ตัวแปร mAuth ใช้ในการจัดเก็บสถานะการเข้าสู่ระบบของผู้ใช้
- บรรทัดที่ 2 ตัวแปร email จัดเก็บอีเมลของผู้ใช้
- บรรทัดที่ 2 ตัวแปร password จัดเก็บรหัสผ่านของผู้ใช้
- บรรทัดที่ 3 ตัวแปร userName ใช้ในการรับค่าอีเมลจากผู้ใช้
- บรรทัดที่ 3 ตัวแปร userPassword ใช้ในการรับค่ารหัสผ่านจากผู้ใช้

```

1  email = userName.getText().toString();
2  password = userPassword.getText().toString();
3  if(email.isEmpty() || email == null || password.
    isEmpty() || password == null ){
4      Toast.makeText(SignInActivity.this, "Please fill
        data!", Toast.LENGTH_SHORT).show();
5      return;
6  }
7  simpleProgressBar.setVisibility(View.VISIBLE);
8  mAuth.signInWithEmailAndPassword(email, password)
9  .addOnCompleteListener(this, new OnCompleteListener
    <AuthResult>() {
10     @Override
11     public void onComplete(@NonNull Task<AuthResult>
        task) {
12         if (task.isSuccessful()) {
13             startActivity(new Intent(SignInActivity.this,
                MainActivity.class));
14             finish();
15         } else {
16             Toast.makeText(SignInActivity.this, "
                Authentication failed.",
17                 Toast.LENGTH_SHORT).show();
18         }
19         simpleProgressBar.setVisibility(View.INVISIBLE);
20     }
21 }
22 );

```

รูปที่ 4.31: โค้ดส่วนที่ใช้ในการเข้าสู่ระบบของคลาส SignInActivity

จากภาพที่ 4.31 โค้ดส่วนที่ใช้ในการเข้าสู่ระบบของคลาส SignInActivity สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ตัวแปร email จัดเก็บอีเมลของผู้ใช้
- บรรทัดที่ 2 ตัวแปร password จัดเก็บรหัสผ่านของผู้ใช้
- บรรทัดที่ 3 ตรวจสอบความถูกต้องของข้อมูล
- บรรทัดที่ 8-21 เป็นการเรียกใช้เฟิร์มแวร์ Authentication เพื่อเข้าใช้งานระบบ

#### 4.2.6 โครงสร้างของการสร้างหน้า ScheduleFragment

```

1 private RecyclerView recyclerView;
2 private FirebaseFirestore db;
3 private ArrayList<Event> events;
4 private EventItemAdapter adapter;
5 private DatePickerTimeline datePicker;

```

รูปที่ 4.32: โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment

จากภาพที่ 4.34 โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ตัวแปร recyclerView ใช้ในการแสดงลิสต์รายการกำหนดการ
- บรรทัดที่ 2 ตัวแปร db ใช้ในการสืบค้นข้อมูลจาก Cloud Firestore
- บรรทัดที่ 3 ตัวแปร events จัดเก็บข้อมูลกำหนดการ
- บรรทัดที่ 4 ตัวแปร adapter ใช้ในการแปลงชุดข้อมูลกำหนดการเป็นลิสต์รายการเพื่อใช้แสดงใน recyclerView
- บรรทัดที่ 5 ตัวแปร datePicker ใช้ในการแสดงปฏิทิน

```

1 db.collection("Events")
2 .orderBy(getString(R.string.key_time), Query.
   Direction.DESENDING)
3 .get()
4 .addOnCompleteListener(new OnCompleteListener<
   QuerySnapshot>() {
5     @Override
6     public void onComplete(@NonNull Task<QuerySnapshot>
       task) {
7         if (task.isSuccessful() && isAdded()) {
8             for (DocumentSnapshot document : task.getResult
               ()) {
9                 Log.d(TAG, document.getId() + " => " +
                   document.getData());
10                Map<String, Object> data = document.getData();
11                Date dbDate = (Date) data.get(getString(R.
                  string.key_time));
12                if (dbDate.getDate() == datePicker.
                  getSelectedDay() && dbDate.getMonth() ==
                  datePicker.getSelectedMonth()) {
13                    Event event = new Event();
14                    event.setTitle(data.get(getString(R.string.
                      key_title)).toString());
15                    event.setDescription(data.get(getString(R.
                      string.key_description)).toString());
16                    event.setTime((Date) data.get(getString(R.
                      string.key_time)));
17                    events.add(event);
18                }
19            }
20            recyclerView.setLayoutManager(new
               LinearLayoutManager(getActivity()));
21            recyclerView.setAdapter(adapter);
22            adapter.notifyDataSetChanged();
23        } else {
24            Log.w(TAG, "Error getting documents.", task.
               getException());
25        }
26    }
27 }
28 );

```

รูปที่ 4.33: โค้ดส่วนที่ใช้ในการสืบค้นข้อมูลกำหนดการของคลาส ScheduleFragment

จากภาพที่ 4.33 โค้ดส่วนที่ใช้ในการสืบค้นข้อมูลกำหนดการของคลาส ScheduleFragment สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1-4 เป็นการสืบค้นข้อมูลกำหนดการโดยเรียงลำดับข้อมูลล่าสุดก่อน
- บรรทัดที่ 7-19 เป็นการนำข้อมูลที่ได้จากการสืบค้นแปลงเป็นลิสต์รายการและแสดงผล
- บรรทัดที่ 20-22 เป็นการอัปเดตลิสต์รายการ

#### 4.2.7 โครงสร้างของการสร้างหน้า ScheduleFragment

```
1 private RecyclerView recyclerView;
2 private FirebaseFirestore db;
3 private ArrayList<Event> events;
4 private EventItemAdapter adapter;
5 private DatePickerTimeline datePicker;
```

รูปที่ 4.34: โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment

จากภาพที่ 4.34 โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment สามารถอธิบายได้ดังนี้

- บรรทัดที่ 1 ตัวแปร recyclerView ใช้ในการแสดงลิสต์รายการกำหนดการ
- บรรทัดที่ 2 ตัวแปร db ใช้ในการสืบค้นข้อมูลจาก Cloud Firestore
- บรรทัดที่ 3 ตัวแปร events จัดเก็บข้อมูลกำหนดการ
- บรรทัดที่ 4 ตัวแปร adapter ใช้ในการแปลงชุดข้อมูลกำหนดการเป็นลิสต์รายการเพื่อใช้แสดงใน recyclerView
- บรรทัดที่ 5 ตัวแปร datePicker ใช้ในการแสดงปฏิทิน

## บทที่ 5

### การทดสอบระบบ

การทดสอบการทำงานของเว็บไซต์ โดยทำการทดสอบในลักษณะ Black-box Testing [2] หรือ Data-Driven testing ซึ่งเป็นการทดสอบที่ไม่สนใจโปรเซส (Process) การทำงานภายในของโปรแกรมว่าทำงานอย่างไร แต่จะเน้นไปที่ Input และ Result ที่ได้มากกว่าว่าการทำงานต่าง ๆ ถูกต้องตามความต้องการ (Requirement) หรือไม่ ซึ่งการทดสอบการใช้งานแอนดรอยด์ แอปพลิเคชัน และ การใช้งานเว็บแอปพลิเคชัน ได้ผลดังนี้

#### 5.1 การทดสอบการใช้งานแอนดรอยด์แอปพลิเคชัน

- การทดสอบการใช้งานเมนูนำทางของแอนดรอยด์แอปพลิเคชัน การทดสอบเมนูนำทางของแอปพลิเคชันในการนำทางผู้ใช้งาน ซึ่งเมนูหลักประกอบด้วย เมนูหน้าประกาศ เมนูหน้าสนทนา เมนูหน้าปฏิทินกำหนดการ เมนูหน้าดาวโหลดเอกสาร เมนูส่งภาพถ่ายสำเนาเอกสาร เมนูหน้าจองคิวส่งเอกสาร เมนูหน้าคำถามที่พบบ่อย เมนูหน้าเกี่ยวกับ เมนูหน้าข้อมูลส่วนตัวและเมนูออกจากระบบ ผลทดสอบดังตารางที่ 5.9-5.2

ตารางที่ 5.1: ผลการทดสอบเมนูนำทาง

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
เมนูประกาศ	กดปุ่มเมนูประกาศ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ประกาศทั้งหมด
เมนูสนทนา	กดปุ่มเมนูสนทนา	ระบบแสดงผลหน้าจอสนทนา พร้อมทั้งแสดงข้อมูลประวัติ- การสนทนา
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าปฏิทินกำหนดการ	กดปุ่มเมนูปฏิทินกำหนดการ	ระบบแสดงผลหน้าจอสนทนา พร้อมทั้งแสดงข้อมูลประวัติ- การสนทนา
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าดาวนโหลดเอกสาร	กด ปุ่ม เมนู หน้า ดาวนโหลด เอกสาร	ระบบ แสดง ผล หน้า จอ รายู- การเอกสารในระบบพร้อมทั้ง แสดงปุ่มดาวนโหลด
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนู หน้า ส่ง ภาพถ่าย สำเนา เอกสาร	กดปุ่มเมนูหน้าส่งภาพสำเนา เอกสาร	ระบบแสดงผลหน้าส่งเอกสาร ภาพสำเนาเอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด



ตารางที่ 5.2: ผลการทดสอบเมนูนำทาง(ต่อ)

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
เมนูหน้าจอลงชื่อเอกสาร	กดปุ่มเมนูหน้าจอลงชื่อเอกสาร	ระบบแสดงผลหน้าจอลงชื่อเอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ขาว-สาร พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าคำถามที่พบบ่อย	กด ปุ่ม เมนู หน้า คำถาม ที่ พบบ่อย	ระบบ แสดง หน้า จอ ลง ชื่อ เอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ-ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าเกี่ยวกับ	กดปุ่มเมนูหน้าเกี่ยวกับ	ระบบ แสดง ผล หน้า เกี่ยว กับ ซึ่งแสดงข้อมูลผู้พัฒนารวมไปถึงแสดงเครดิต (credit) โลโก้ต่าง ๆ ที่ใช้งานภายในแอปพลิเคชัน
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ-ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าบัญชีผู้ใช้	กดปุ่มเมนูหน้าบัญชีผู้ใช้	ระบบ แสดง ผล หน้า จอ ข้อมูลส่วนตัวโดยมีข้อมูล รูปประจำตัว ชื่อผู้ใช้ สาขาวิชาและภาควิชา
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ-ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูออกจากระบบ	กดปุ่มเมนูออกจากระบบ	ทำการ ออก จาก ระบบ และ แสดงหน้าจอลงชื่อเอกสาร

- การทดสอบหน้ารายละเอียดประกาศ ในการแสดงผลหน้าจอรายละเอียดประกาศนั้นจะประกอบไปด้วยหัวเรื่องประกาศ รายละเอียดประกาศ วันที่ประกาศและเอกสารแนบ ผลการทดสอบดังตารางที่ 5.10

ตารางที่ 5.3: ผลการทดสอบหน้ารายละเอียดประกาศ

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้ารายละเอียดประกาศ	กดปุ่มเมนูประกาศ	ระบบ แสดง ผู้ล หน้า จอ ประ- กาศ พร้อม ทั้ง แสดง รายการ ประกาศทั้งหมด
	กด ปุ่ม อ่าน รายละเอียด ประ- กาศ	ระบบแสดงผลหน้าจอรายละเอียดประกาศ
	กด ปุ่ม ดาวนโหลด เอกสาร แนบ	ระบบแสดงผลการดาวนโหลด เอกสารแนบ
	เมื่อ ดาวนโหลด เสร็จ กด ปุ่ม เปิดเอกสาร	ระบบแสดงผลเอกสาร
	กดปุ่มย้อนกลับ	ระบบแสดงผลหน้าจอรายละเอียดประกาศ
	กดปุ่มย้อนกลับอีกครั้ง	ระบบ แสดง ผู้ล หน้า จอ ประ- กาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าสนทนา ในการแสดงผลหน้าจอสนทนานั้นจะประกอบไปด้วยรายการประวัติการสนทนา ช่องกรอกข้อความและปุ่มส่งข้อความ ผลการทดสอบดังตารางที่ 5.11

ตารางที่ 5.4: ผลการการทดสอบหน้าสนทนา

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้ารายละเอียดประกาศ	กดปุ่มเมนูสนทนา	ระบบแสดงผลหน้าจอสนทนา พร้อม ทั้ง แสดง รายการ ประวัติการสนทนา
	กดปุ่มที่ช่องกรอกข้อความ	ระบบ แสดง ตัว กระ พริบ (cursor) เพื่อ ชี้ ให้ รู้ ว่า ตำแหน่งของการพิมพ์อักขระ
	พิมพ์อักขระ	ระบบ แส ดง ผลอัก ข ระ ที่ ถูก พิมพ์
	กดปุ่มส่งข้อความ	ระบบ แสดง ข้อความ ที่ ถูก พิมพ์ บน รายการ ประวัติสนทนาล่าสุด
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- กาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าปฏิทินกำหนดการ ในการแสดงผลหน้าปฏิทินกำหนดการนั้นจะประกอบไปด้วยรายการประวัติการสนทนา ช่องกรอกข้อความและปุ่มส่งข้อความ ผลการทดสอบดังตารางที่ 5.12

ตารางที่ 5.5: ผลการการทดสอบหน้าปฏิทินกำหนดการ

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้าปฏิทินกำหนดการ	กดปุ่มเมนูปฏิทินกำหนดการ	ระบบแสดงหน้าจอปฏิทินกำหนดการ โดย มี การ แสดง กำหนดการของวันปัจจุบัน
	กดเลือกวันที่ต้องการดูกำหนดการในปฏิทิน	ระบบ แสดง กำหนดการ ของ วันที่ถูกเลือก
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าดาวนโหลดเอกสาร ในการแสดงผลหน้าจอดาวนโหลดเอกสารนั้นจะประกอบไปด้วยรายการเอกสารโดยที่แต่ละฉบับจะแสดงชื่อเอกสารและปุ่มดาวนโหลดเอกสาร ผลการทดสอบดังตารางที่ 5.13

ตารางที่ 5.6: ผลการทดสอบหน้าดาวนโหลดเอกสาร

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้าดาวนโหลดเอกสาร	กด ปุ่ม เมนู หน้า ดาวนโหลดเอกสาร	ระบบ แสดง ผล หน้า จอ รายการเอกสารในระบบพร้อมทั้งแสดงปุ่มดาวนโหลด
	กดปุ่มดาวนโหลดเอกสาร	ระบบแสดงผลการดาวนโหลดเอกสาร
	เมื่อ ดาวนโหลด เสร็จ กด ปุ่ม เปิดเอกสาร	ระบบแสดงผลเอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ รายการเอกสารในระบบพร้อมทั้งแสดงปุ่มดาวนโหลด
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประกาศ พร้อม ทั้ง แสดง รายการข่าวสารทั้งหมด

- การทดสอบหน้าส่งภาพถ่ายสำเนาเอกสาร ในการแสดงผลหน้าส่งภาพถ่ายสำเนาเอกสาร นั้นจะประกอบไปด้วยปุ่มเพิ่มเอกสารฉบับที่ 1 ปุ่มเพิ่มเอกสารฉบับที่ 2 และปุ่มส่งเอกสาร ผลการทดสอบดังตารางที่ 5.7

ตารางที่ 5.7: ผลการทดสอบหน้าส่งภาพถ่ายสำเนาเอกสาร

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้า ส่ง ภาพถ่าย สำเนา เอก- สาร	กดปุ่มเมนูหน้าจอควมส่งเอก- สาร	ระบบ แสดง หน้า จอ ควม ส่ง เอกสาร
	กดปุ่มเพิ่มเอกสารฉบับที่ 1	ระบบแสดงหน้าจอกล้องถ่าย ภาพ
	กดปุ่มถ่ายภาพเอกสาร	ระบบแสดงผลภาพเอกสาร
	กดปุ่มถัดไป	ระบบ แสดง ผล หน้า ปรับ แต่ง ภาพเอกสาร
	กดปุ่มยืนยัน	ระบบ แสดง ผล ภาพ หน้า ส่ง ภาพถ่ายสำเนาเอกสารพร้อม ทั้งแสดงผลภาพเอกสารฉบับ ที่ 1
	กดปุ่มเพิ่มเอกสารฉบับที่ 2	ระบบแสดงหน้าจอกล้องถ่าย ภาพ
	กดปุ่มถ่ายภาพเอกสาร	ระบบแสดงผลภาพเอกสาร
	กดปุ่มถัดไป	ระบบ แสดง ผล หน้า ปรับ แต่ง ภาพเอกสาร
	กดปุ่มยืนยัน	ระบบ แสดง ผล ภาพ หน้า ส่ง ภาพถ่ายสำเนาเอกสารพร้อม ทั้งแสดงผลภาพเอกสารฉบับ ที่ 1 และฉบับที่ 2
	กดส่งเอกสาร	ระบบแสดงผลการส่งเอกสาร และ แสดง สถานะ การ ตรวจ เอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าจอควิส่งเอกสาร ในการแสดงผลหน้าจอควิส่งเอกสารนั้นจะประกอบไปด้วยปุ่มกดเลือกวันที่ ปุ่มกดเลือกเวลา ผลการทดสอบดังตารางที่ 5.8

ตารางที่ 5.8: ผลการทดสอบหน้าจอควิส่งเอกสาร

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้าจอควิส่งเอกสาร	กดปุ่มเมนูหน้าจอควิส่งเอกสาร	ระบบแสดงผลหน้าจอกำหนดการส่งเอกสาร
	กดปุ่มเลือกวันที่ต้องการส่งเอกสาร	ระบบแสดงผลวันที่ถูกเลือก
	กดปุ่มเลือกเวลาที่ต้องการส่งเอกสาร	ระบบแสดงผลเวลาที่ถูกเลือก พร้อมทั้งแสดงปุ่มกดบันทึก
	กดปุ่มบันทึก	ระบบแสดงผลการจองวันที่ส่งเอกสาร
	กดปุ่มย้อนกลับ	ระบบแสดงผลหน้าจอประกาศ พร้อมทั้งแสดงรายการข่าวสารทั้งหมด

## 5.2 การทดสอบการใช้งานเว็บแอปพลิเคชัน

- การทดสอบการใช้งานเมนูนำทางของเว็บแอปพลิเคชัน การทดสอบเมนูนำทางของเว็บแอปพลิเคชันในการนำทางผู้ใช้งาน ซึ่งเมนูหลักประกอบด้วย เมนูหน้าประกาศ เมนูหน้าสนทนา เมนูหน้าปฏิทินกำหนดการ เมนูหน้าดาวโหลดเอกสาร เมนูหน้าคำถามที่พบบ่อย เมนูหน้าเกี่ยวกับและเมนูออกจากระบบ ผลทดสอบดังตารางที่ 5.9-5.2

ตารางที่ 5.9: ผลการทดสอบเมนูนำทาง

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
เมนูประกาศ	กดปุ่มเมนูประกาศ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ประกาศทั้งหมด
เมนูสนทนา	กดปุ่มเมนูสนทนา	ระบบแสดงผลหน้าจอสนทนา พร้อมทั้งแสดงข้อมูลประวัติ- การสนทนา
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าปฏิทินกำหนดการ	กดปุ่มเมนูปฏิทินกำหนดการ	ระบบแสดงผลหน้าจอสนทนา พร้อมทั้งแสดงข้อมูลประวัติ- การสนทนา
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าดาวนโหลดเอกสาร	กด ปุ่ม เมนู หน้า ดาวนโหลด เอกสาร	ระบบ แสดง ผล หน้า จอ ตา- ราง รายการ เอกสาร ใน ระบบ พร้อมทั้งแสดงปุ่มดาวนโหลด
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าคำถามที่พบบ่อย	กด ปุ่ม เมนู หน้า คำถาม ที่ พบ บ่อย	ระบบ แสดง หน้า จอ ง คิว ส่ ง เอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าเกี่ยวกับ	กดปุ่มเมนูหน้าเกี่ยวกับ	ระบบ แสดง ผล หน้า เกี่ยว กับ ซึ่งแสดงข้อมูลผู้พัฒนารวมไป ถึงแสดงเครดิต (credit) โล- บารี่ต่าง ๆ ที่ใช้งานภายใน แอปพลิเคชัน
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูออกจากระบบ	กดปุ่มเมนูออกจากระบบ	ทำการ ออก จาก ระบบ และ แสดงหน้าจอข่าวสาร



- การทดสอบหน้ารายละเอียดประกาศ ในการแสดงผลหน้าจอรายละเอียดประกาศนั้นจะประกอบไปด้วยหัวเรื่องประกาศ รายละเอียดประกาศ วันที่ประกาศและเอกสารแนบ ผลการทดสอบดังตารางที่ 5.10

ตารางที่ 5.10: ผลการทดสอบหน้ารายละเอียดประกาศ

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้ารายละเอียดประกาศ	กดปุ่มเมนูประกาศ	ระบบ แสดง ผล หน้า จอ ประ- กาศ พร้อม ทั้ง แสดง รายการ ประกาศทั้งหมด
	กด ปุ่ม อ่าน รายละเอียด ประ- กาศ	ระบบแสดงผลหน้าจอรายละเอียดประกาศ
	กด ปุ่ม ดาวน์โหลด เอกสาร แนบ	ระบบแสดงผลการดาวน์โหลด เอกสารแนบ
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- กาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าสนทนา ในการแสดงผลหน้าจอสนทนานั้นจะประกอบไปด้วยรายการประวัติการสนทนา ช่องกรอกข้อความและปุ่มส่งข้อความ ผลการทดสอบดังตารางที่ 5.11

ตารางที่ 5.11: ผลการการทดสอบหน้าสนทนา

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้ารายละเอียดประกาศ	กดปุ่มเมนูสนทนา	ระบบแสดงผลหน้าจอสนทนา พร้อม ทั้ง แสดง รายการ ประวัติการสนทนา
	กดปุ่มที่ช่องกรอกข้อความ	ระบบ แสดง ตัว กระ พริบ (cursor) เพื่อ ชี้ ให้ รู้ ว่า ตำแหน่งของการพิมพ์อักขระ
	พิมพ์อักขระ	ระบบ แส ดง ผลอัก ข ระ ที่ ถูก พิมพ์
	กดปุ่มส่งข้อความ	ระบบ แสดง ข้อความ ที่ ถูก พิมพ์ บน รายการ ประวัติสนทนาล่าสุด
	กดปุ่มย้อนกลับ	ระบบ แสดง หน้า จอ ประ- กาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าปฏิทินกำหนดการ ในการแสดงผลหน้าปฏิทินกำหนดการนั้นจะประกอบไปด้วยรายการประวัติการสนทนา ช่องกรอกข้อความและปุ่มส่งข้อความ ผลการทดสอบดังตารางที่ 5.12

ตารางที่ 5.12: ผลการการทดสอบหน้าปฏิทินกำหนดการ

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้าปฏิทินกำหนดการ	กดปุ่มเมนูปฏิทินกำหนดการ	ระบบ แสดง ตาราง กำหนดการ ทั้งหมด
	กดปุ่มย้อนกลับ	ระบบ แสดง หน้าจอ ประกาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าดาวนโหลดเอกสาร ในการแสดงผลหน้าจอดาวนโหลดเอกสารนั้นจะประกอบไปด้วยรายการเอกสารโดยที่แต่ละฉบับจะแสดงชื่อเอกสารและปุ่มดาวนโหลดเอกสาร ผลการทดสอบดังตารางที่ 5.13

ตารางที่ 5.13: ผลการทดสอบหน้าดาวนโหลดเอกสาร

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้าดาวนโหลดเอกสาร	กด ปุ่ม เมนู หน้า ดาวนโหลดเอกสาร	ระบบ แสดง ผล หน้า จอ ตาราง รายการ เอกสาร ใน ระบบ พร้อมทั้งแสดงปุ่มดาวนโหลด
	กดปุ่มดาวนโหลดเอกสาร	ระบบแสดงผลการดาวนโหลดเอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ รายการเอกสารในระบบพร้อมทั้งแสดงปุ่มดาวนโหลด
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประกาศ พร้อม ทั้ง แสดง รายการข่าวสารทั้งหมด

## บทที่ 6

### สรุปและข้อเสนอแนะ

การดำเนินโครงการเพื่อพัฒนาระบบจองคิวร้านเสริมสวยนี้ พบว่าระบบสามารถทำงานได้ตามที่วิเคราะห์และออกแบบไว้ แต่ก็พบปัญหาและอุปสรรคระหว่างการพัฒนา ในบทนี้ผู้พัฒนาจึงขอสรุปความสามารถของระบบ ชี้แจงปัญหาและอุปสรรค พร้อมเสนอแนวทางในการพัฒนาระบบจองคิวร้านเสริมสวย ต่อ ตามลำดับ

#### 6.1 สรุปความสามารถของระบบ

ระบบจองคิวร้านเสริมสวย เว็บแอปพลิเคชันสามารถสรุปความสามารถที่ระบบทำได้ดังนี้

##### 6.1.1 เว็บแอปพลิเคชัน

ความสามารถหลักของเว็บแอปพลิเคชันนั้นเน้นสร้างความสะดวกต่อการจัดการเอกสารเรื่องข้อมูลต่างๆ ที่เกี่ยวข้องกับ ระบบจองคิวร้านเสริมสวย โดยแบ่งความสามารถของระบบตามประเภทของผู้ใช้งานดังนี้

##### 1. เจ้าของร้าน

- สามารถลงทะเบียนเข้าสู่ระบบด้วย Email ได้
- สามารถดูคิวที่ผู้ใช้บริการได้ทำการจองคิวไว้
- สามารถ post ภาพผลงานทั้งหมดของร้านได้
- สามารถเพิ่ม แก้ไข และลบรายการให้บริการประจำร้านได้
- สามารถเพิ่ม แก้ไข ข้อมูลร้านได้
- สามารถเพิ่ม แก้ไข ข้อมูลตำแหน่งร้านได้

##### 2. ช่างประจำร้าน

- ลงทะเบียนใช้ web ด้วย Email ได้
- สามารถดูตารางการทำงานของตนเองได้
- สามารถ post ภาพผลงานของตัวเองได้
- สามารถแก้ไขข้อมูลส่วนตัวได้

### 3. ผู้ใช้บริการ

- สามารถลงทะเบียนเข้าสู่ระบบด้วย Email ได้
- สามารถค้นหาร้านเสริมสวยได้
- สามารถจองคิวของร้านเสริมสวยได้
- สามารถดูคิวว่างของร้านเสริมสวยได้
- สามารถดูข้อมูลต่างๆของร้านเสริมสวยได้
- สามารถดูตำแหน่งของทางร้านได้
- สามารถดูผลงานของร้านได้
- สามารถเขียนรีวิว ทิชชิ่ง ได้

## 6.2 ปัญหาและอุปสรรคในการพัฒนา

1. เนื่องจากทางผู้พัฒนามีความประสงค์ให้ระบบนี้สามารถใช้งานได้จริง ดังนั้น การพัฒนาในตอนนี้อาจมีข้อจำกัดเรื่องขนาดของข้อมูลที่จัดเก็บบนไฟร์เบสที่สามารถอัปโหลดเข้าสู่ระบบสูงสุดเพียง 5 GB ซึ่งหากระบบถูกใช้งานจริงจำนวนข้อมูลในระบบจะเกินจำนวนที่ไฟร์เบสให้ใช้งานฟรี

แนวทางการแก้ไข : ทำการบีบอัดข้อมูลให้มีขนาดเล็กลง ส่วนในอนาคตอาจจำเป็นต้องศึกษาแนวทางการสร้างเซิร์ฟเวอร์ (Server) เป็นของระบบเอง

## 6.3 แนวทางการพัฒนาต่อ

1. การพัฒนาช่องทางการติดต่อ
2. เจ้าของร้านสามารถยืนยันการจองคิวได้
3. การพัฒนาเป็นแอปพลิเคชัน

## บรรณานุกรม

- [1] Kunchit Phiu-Nual. (2557). ความหมายและความสำคัญของ system architecture [ออนไลน์]. สืบค้นเมื่อ 12 พฤษภาคม 2561. จาก <https://goo.gl/6ZhGQo> .
- [2] Atthaboon S. (2555). Black-box testing strategy [ออนไลน์]. สืบค้นเมื่อ 20 พฤษภาคม 2561. จาก <http://everybitsconsult.com/blog/2015/06/22/black-box-testing.html>

ภาคผนวก



## ภาคผนวก ก

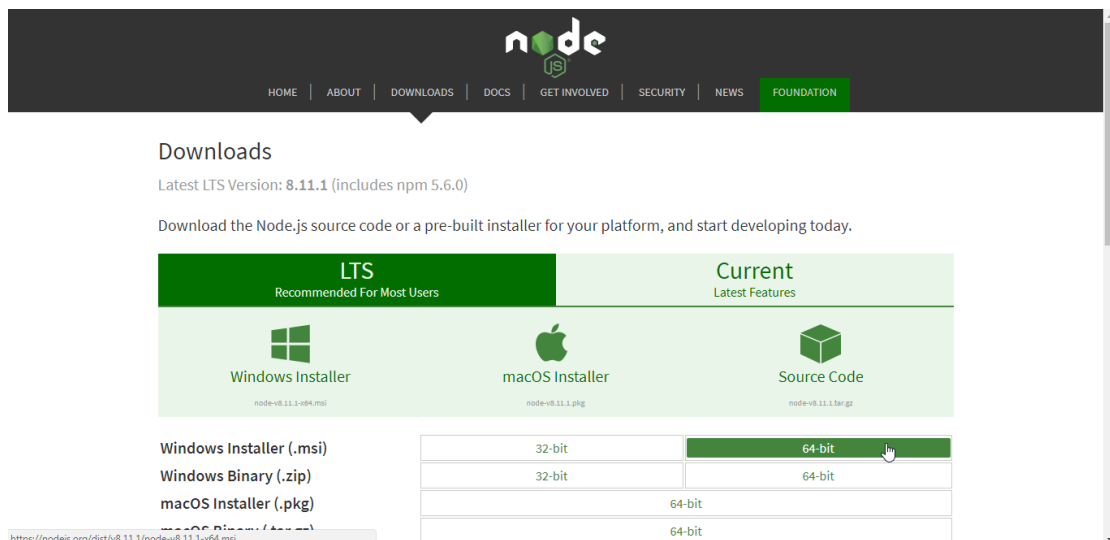
### การติดตั้งเครื่องมือที่ใช้พัฒนาโปรแกรม

การติดตั้งเครื่องมือที่ใช้ในการพัฒนาระบบสหกิจศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี มีโปรแกรมที่จำเป็นในการพัฒนาระบบดังต่อไปนี้

- การติดตั้ง Node.js
- การติดตั้ง React.js
- การติดตั้ง Visual Studio Code

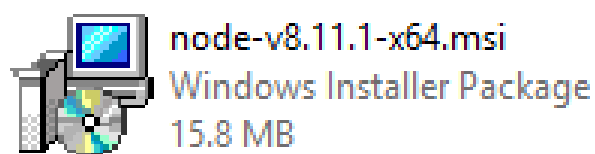
#### ก.1 การติดตั้ง Node.js

- 1) สามารถดาวน์โหลด Node.js ได้ที่ <https://nodejs.org/en/download/> แสดงดังรูปที่ ก.1



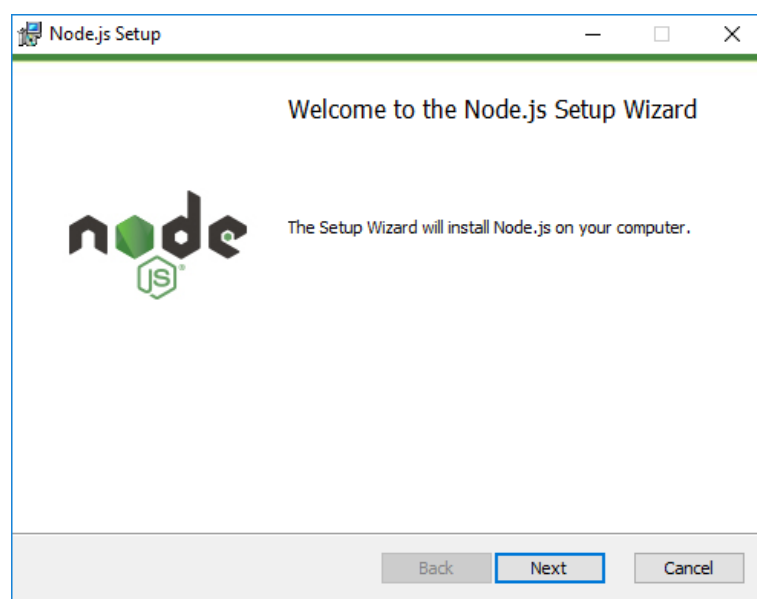
รูปที่ ก.1: หน้าเว็บดาวน์โหลด Node.js

2) เปิดไฟล์ติดตั้ง ชื่อ node-v10.15.3-x64.msi เพื่อติดตั้ง แสดงดังรูปที่ ก.2



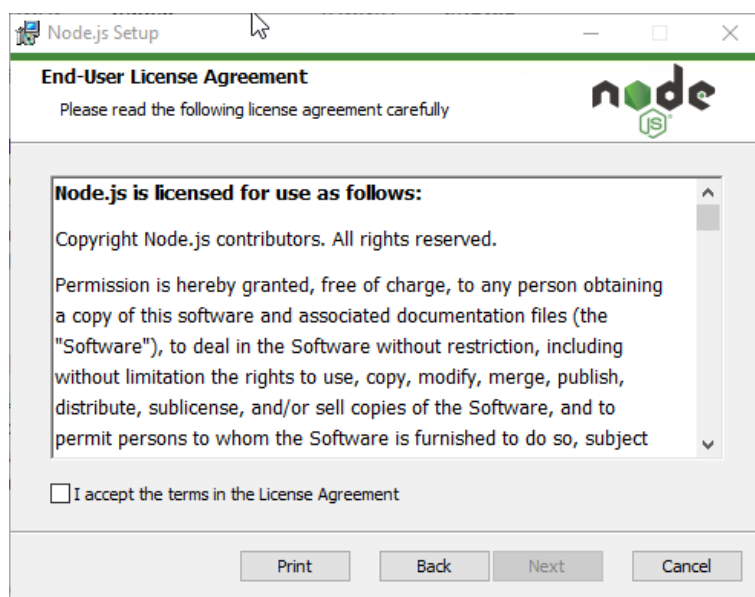
รูปที่ ก.2: ไฟล์ติดตั้งสำหรับติดตั้ง Node.js

3) แสดงหน้าต่างตอนรับของ Node.js ให้กด Next แสดงดังรูปที่ ก.3



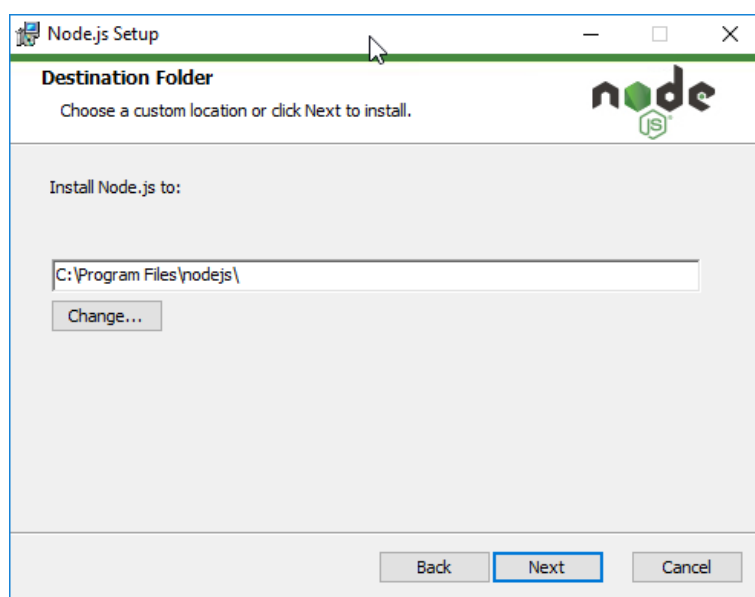
รูปที่ ก.3: หน้าต่างตอนรับของ Node.js

- 4) แสดงหน้าต่างข้อตกลงในการใช้ Node.js ให้เลือกช่อง I accept the terms in the License Agreement และกด Next แสดงดังรูปที่ ก.4



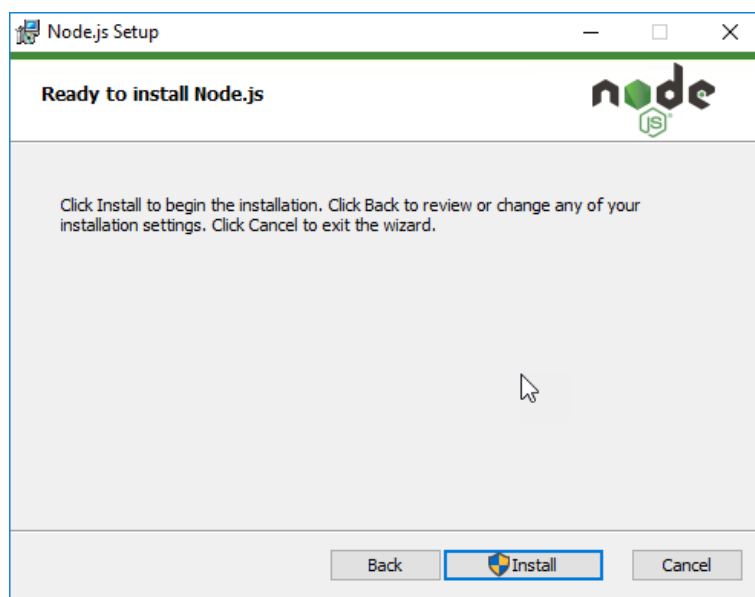
รูปที่ ก.4: หน้าต่างข้อตกลงในการใช้ Node.js

- 5) แสดงหน้าต่างเลือกโฟลเดอร์ที่จะทำการติดตั้ง แสดงดังรูปที่ ก.5



รูปที่ ก.5: หน้าต่างเลือกโฟลเดอร์ที่จะทำการติดตั้ง Node.js

6) แสดงหน้าต่างสำหรับติดตั้ง Node.js ให้กด Install เพื่อทำงานติดตั้ง แสดงดังรูปที่ ก.6



รูปที่ ก.6: หน้าต่างติดตั้ง Node.js

## ก.2 การติดตั้ง React.js

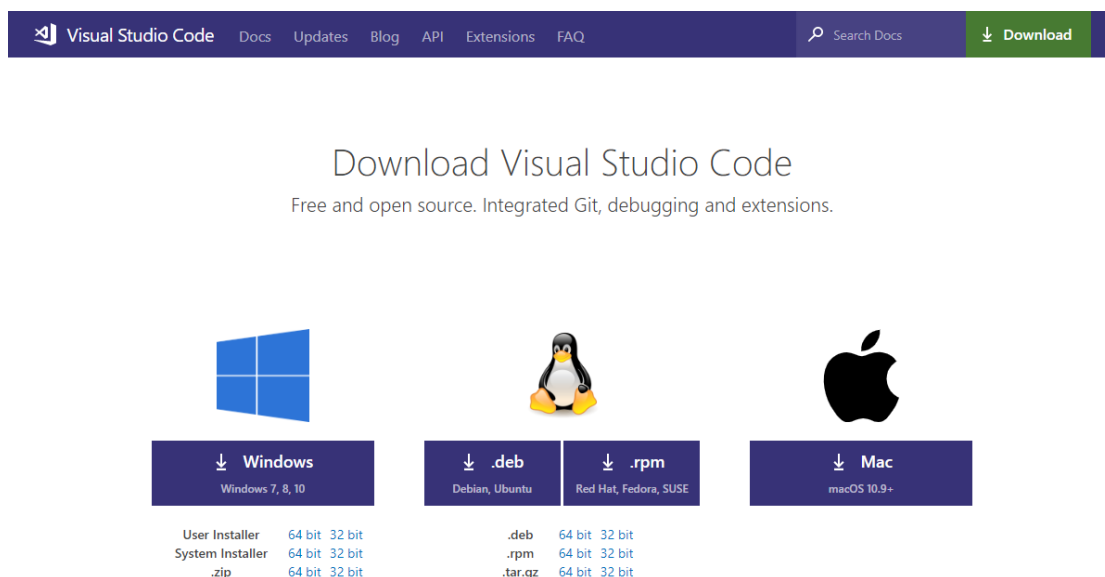
การติดตั้ง React.js สามารถผ่านคำสั่ง command line ได้ แสดงดังรูปที่ ก.7

```
npm install create-react-app --save
```

รูปที่ ก.7: คำสั่งสำหรับติดตั้ง React.js

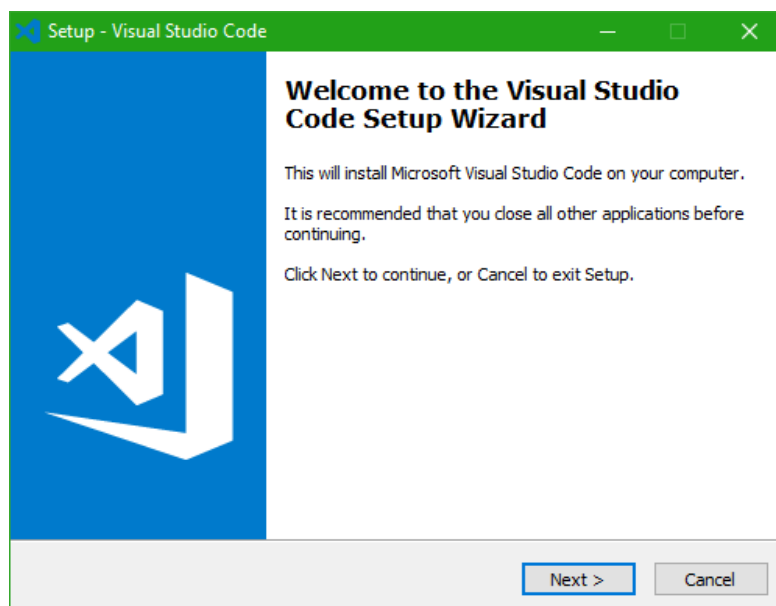
## ก.3 การติดตั้ง Visual Studio Code

1. สามารถดาวน์โหลด Visual Studio Code ได้ที่ <https://code.visualstudio.com/download> ดังแสดงในรูปที่ ก.8



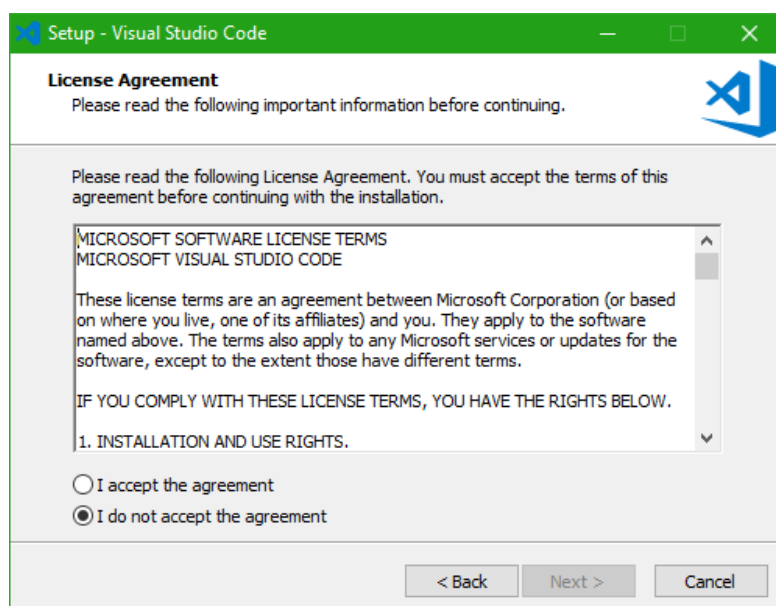
รูปที่ ก.8: หน้าเว็บดาวน์โหลด Visual Studio Code

2. เมื่อเปิดตัวติดตั้งขึ้นมาแล้ว จะแสดงหน้าจอ Welcome to the Visual Studio Code Setup Wizard ให้กดปุ่ม Next เพื่อเริ่มกระบวนการติดตั้ง ดังแสดงในรูปที่ ก.9



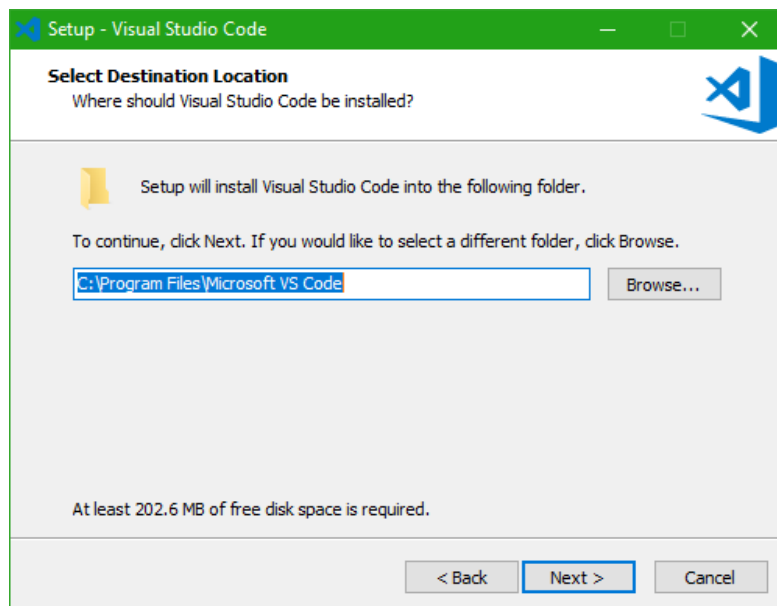
รูปที่ ก.9: หน้าต่างต้อนรับของ Visual Studio Code

3. หลังจากนั้นจะแสดงหน้าต่างข้อตกลงการใช้งาน Visual Studio Code ทำการติ๊กที่ I accept the areement แล้วกด Next ดังแสดงในรูปที่ ก.10



รูปที่ ก.10: หน้าต่างข้อตกลงการใช้งาน Visual Studio Code

4. จากนั้นจะแสดงหน้าต่างที่จัดเก็บไฟล์ต่างๆ ของ Visual Studio Code ทำการกด Next ดังแสดงในรูปที่ ก.11

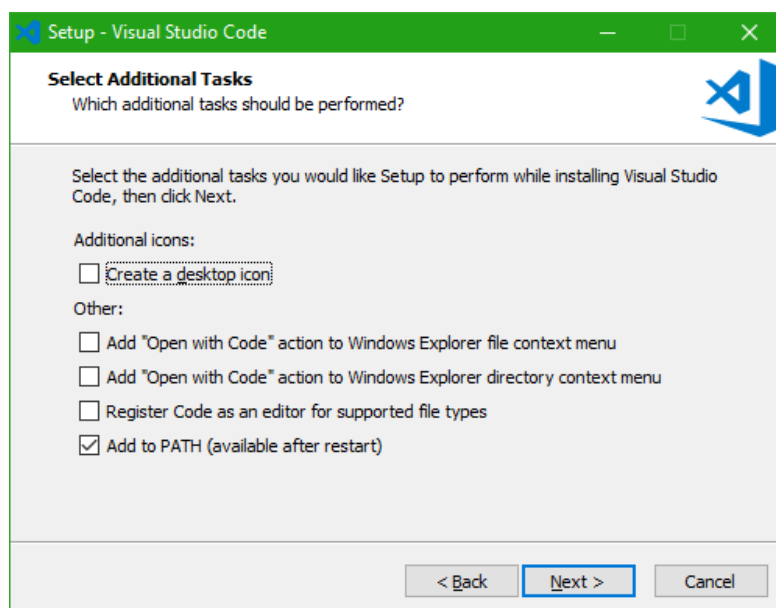


รูปที่ ก.11: หน้าต่างที่จัดเก็บไฟล์ต่างๆ ของ Visual Studio Code

5. จากนั้นจะแสดงหน้าต่างการจัดการซอร์สโค้ดของ Visual Studio Code ทำการกด Next ดังแสดงในรูปที่ ก.12

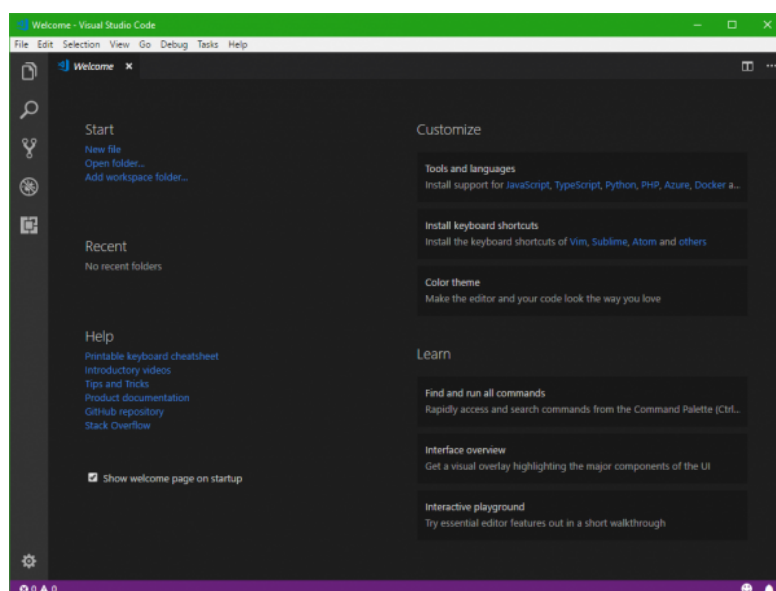
รูปที่ ก.12: หน้าต่างการจัดการซอร์สโค้ด ของ Visual Studio Code

6. จากนั้นแสดงหน้าต่างเริ่มทำการติดตั้งทำการกด Next ดังแสดงในรูปที่ ก.13



รูปที่ ก.13: หน้าต่างเริ่มทำการติดตั้งทำการกด ของ Visual Studio Code

7. จากนั้นจะแสดงหน้าต่างเมื่อเข้าโปรแกรมหลังติดตั้งเสร็จ ดังแสดงในรูปที่ ก.14



รูปที่ ก.14: หน้าต่างเมื่อเข้าโปรแกรมหลังติดตั้งเสร็จ ของ Visual Studio Code



## ภาคผนวก ข

### คู่มือการติดตั้งระบบ

ในการติดตั้งเพื่อใช้งานแอปพลิเคชันระบบกองทุนเงินให้กู้ยืมเพื่อการศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี หรือ ESP สามารถทำได้โดยมีขั้นตอนดังนี้

1. สามารถดาวน์โหลด ESP installer package ได้ที่ <https://drive.google.com/drive/folders/1k6HnoFgLAatgrfLAtFgnJYbRrpJHHwol> ดังแสดงในรูปที่ ข.1

รูปที่ ข.1: หน้าเว็บดาวน์โหลด ESP installer package

2. ดัดลอกไฟล์ app-debug.apk ที่อยู่ในแฟ้มงาน(Folder)ที่อยู่บนคอมพิวเตอร์ไปไว้ในหน่วยความจำบนอุปกรณ์ที่ต้องการ ดังแสดงในรูปที่ ข.2

รูปที่ ข.2: ไฟล์ app-debug.apk บนอุปกรณ์

3. ทำการเปิดไฟล์ app-debug.apk และกด INSTALL เพื่อทำการติดตั้ง ดังแสดงในรูปที่ ข.3

รูปที่ ข.3: หน้าจอต้อนรับการติดตั้งแอปพลิเคชันบนอุปกรณ์แอนดรอยด์

4. เมื่อทำการติดตั้งแอปพลิเคชันสำเร็จระบบจะแสดงผล ดังแสดงในรูปที่ ข.4

รูปที่ ข.4: หน้าจอต้อนรับการติดตั้งแอปพลิเคชันบนอุปกรณ์แอนดรอยด์

## ภาคผนวก ค

### คู่มือการใช้งานระบบ

คู่มือการใช้งานทั้งหมดของระบบ สามารถแบ่งออกเป็น 2 ส่วน ดังนี้

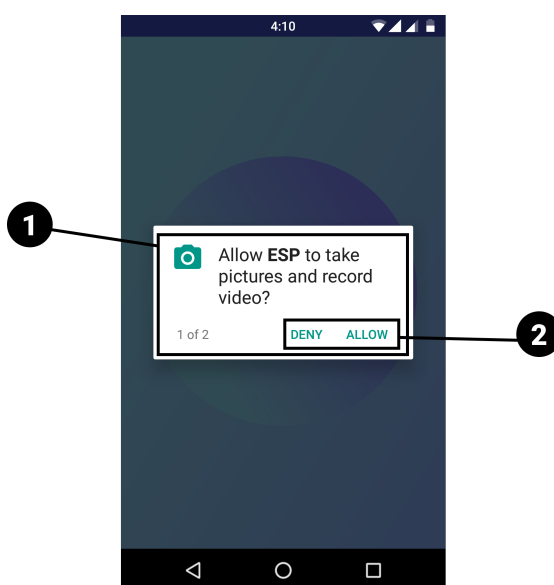
#### 1. ส่วนของหน้าเมนูแอปพลิเคชันสำหรับนักศึกษา

- หน้าจอต้อนรับแสดงผลทุกครั้งเมื่อผู้ใช้ทำการเปิดใช้งานแอปพลิเคชัน ดังแสดงในรูปแบบที่ ค.8



รูปที่ ค.1: หน้าจอต้อนรับ

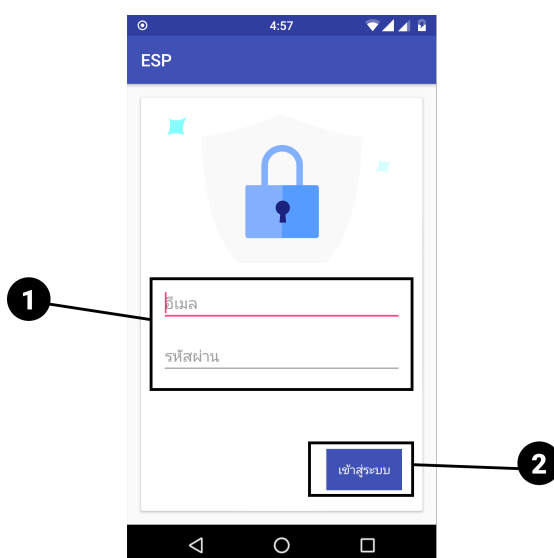
- เมื่อระบบทำการตรวจสอบว่ามีสิทธิ์(Permission)ในการใช้งานแอปพลิเคชันที่ผู้ใช้ยังไม่ได้อนุญาตให้เข้าถึง ระบบจะแสดงหน้าต่างขอสิทธิ์การเข้าถึง ดังแสดงในรูปแบบที่ ค.9



รูปที่ ค.2: หน้าต่างขอสิทธิ์การเข้าถึง

จากรูปที่ ค.9 สามารถอธิบายการใช้งานได้ดังนี้

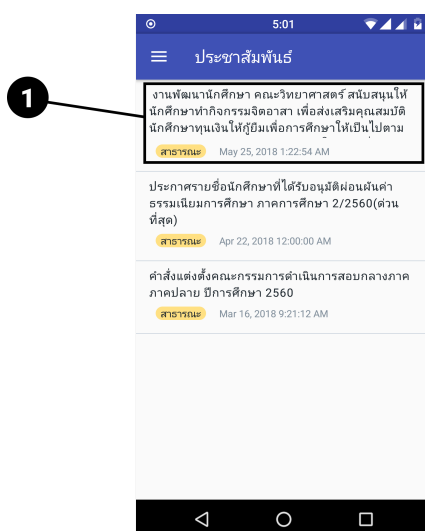
- หมายเลข 1 คือ หน้าต่างขอสิทธิ์การเข้าถึงข้อมูล
- หมายเลข 2 คือ ปุ่มให้สิทธิ์และยกเลิกการให้สิทธิ์
- ระบบทำการตรวจสอบทุกครั้งเมื่อผู้ใช้งานเปิดใช้งานแอปพลิเคชัน หากผู้ใช้งานคนปัจจุบันยังไม่ได้เข้าสู่ระบบ ระบบจะแสดงหน้าจอเข้าสู่ระบบโดยผู้ใช้งานจำเป็นต้องทำการกรอกข้อมูลคือ อีเมลและรหัสผ่าน ดังแสดงในรูปที่ ค.3



รูปที่ ค.3: หน้าจอเข้าสู่ระบบ

จากรูปที่ ค.3 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ ส่วนของฟอร์มในการกรอกข้อมูลอีเมลและรหัสผ่าน
- หมายเลข 2 คือ ปุ่มกดเข้าสู่ระบบ
- หน้าแสดงข่าวสารประชาสัมพันธ์ซึ่งเป็นหน้าจอหลักของแอปพลิเคชัน ดังแสดงในรูปที่ ค.4



รูปที่ ค.4: หน้าแสดงข่าวสาร

จากรูปที่ ค.4 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ ข่าวสารที่มีข้อมูล หัวข้อข่าวสาร แท็ก(Tag)และวันที่ประกาศข่าวสาร
- เมื่อผู้ใช้กดเลือกดูรายละเอียดของข่าวสาร ระบบจะแสดงหน้าจอรายละเอียดข่าวสาร ดังแสดงในรูปที่ ค.5



รูปที่ ค.5: หน้ารายละเอียดของข่าวสาร

จากรูปที่ ค.5 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ หัวข้อข่าวสาร
- หมายเลข 1 คือ รายละเอียดของข่าวสาร
- เมื่อผู้ใช้กดเลือกเมนู Drawer ระบบจะแสดงเมนูนำทางหลักของแอปพลิเคชันซึ่งประกอบไปด้วยเมนู ประชาสัมพันธ์ ข้อความ กำหนดการ เอกสาร ส่งเอกสาร จอกรับ คำถามที่พบบ่อย เกี่ยวกับเรา บัญชีผู้ใช้และออกจากระบบ ดังแสดงในรูปที่ ค.6

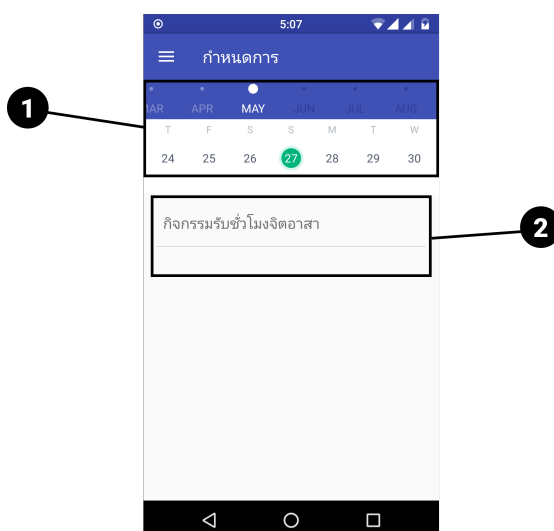


รูปที่ ค.6: เมื่อนำทางหลักของแอปพลิเคชัน

จากรูปที่ ค.6 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ รายการเมื่อนำทางหลักของแอปพลิเคชัน
- หมายเลข 2 คือ เมื่อนำทาง
- เมื่อผู้ใช้เลือกเมนูกำหนดการ ระบบจะแสดงหน้าจอกำหนดการ ดังแสดงในรูปที่ ค.

10



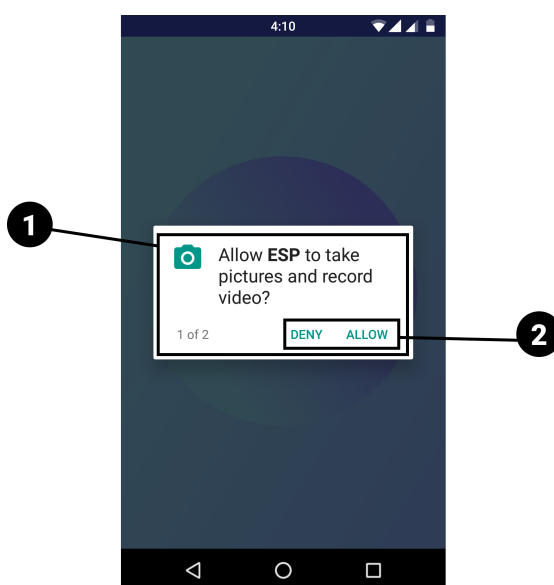
รูปที่ ค.7: หน้าจอกำหนดการ

- ส่วนของหน้าเมนูแอปพลิเคชันสำหรับนักศึกษา
- หน้าจอต้อนรับแสดงผลทุกครั้งเมื่อผู้ใช้ทำการเปิดใช้งานแอปพลิเคชัน ดังแสดงในรูปที่ ค.8



รูปที่ ค.8: หน้าจอต้อนรับ

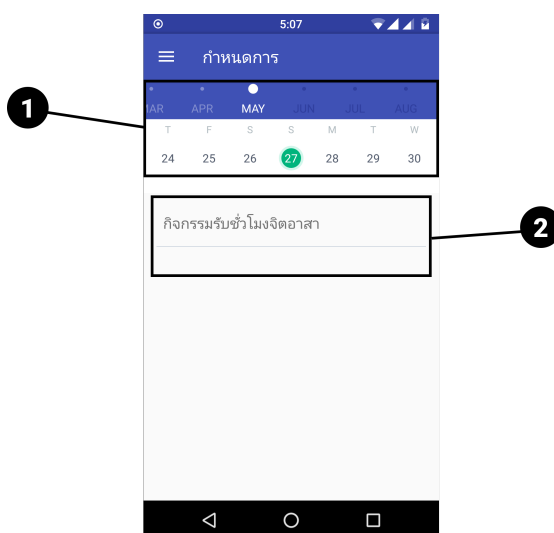
- เมื่อระบบทำการตรวจสอบว่ามีสิทธิ์(Permission)ในการใช้งานแอปพลิเคชันที่ผู้ใช้ยังไม่ได้อนุญาตให้เข้าถึง ระบบจะแสดงหน้าต่างขอสิทธิ์การเข้าถึง ดังแสดงในรูปที่ ค.9



รูปที่ ค.9: หน้าต่างขอสิทธิ์การเข้าถึง

จากรูปที่ ค.9 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ หน้าต่างขอสิทธิ์การเข้าถึงข้อมูล
- หมายเลข 2 คือ ปุ่มให้สิทธิ์และยกเลิกการให้สิทธิ์



รูปที่ ค.10: หน้าแสดงกำหนดการ

จากรูปที่ ค.10 สามารถอธิบายการใช้งานได้ดังนี้

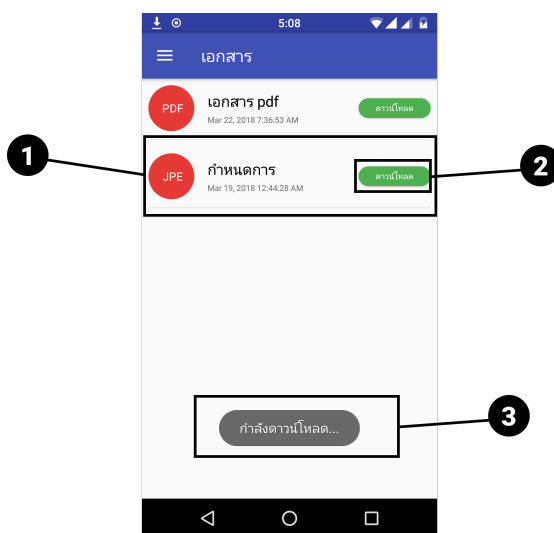
- หมายเลข 1 คือ ปฏิทินแสดงวันที่ปัจจุบันและผู้ใช้สามารถเลือกวันที่ต้องการ



เพื่อดูกำหนดการก่อนหน้า

- หมายเลข 2 คือ แสดงรายการกำหนดการของวันนั้น ๆ
- เมื่อผู้ใช้เลือกเมนูเอกสาร ระบบจะแสดงหน้าจอรายการเอกสาร ดังแสดงในรูปที่ ค.

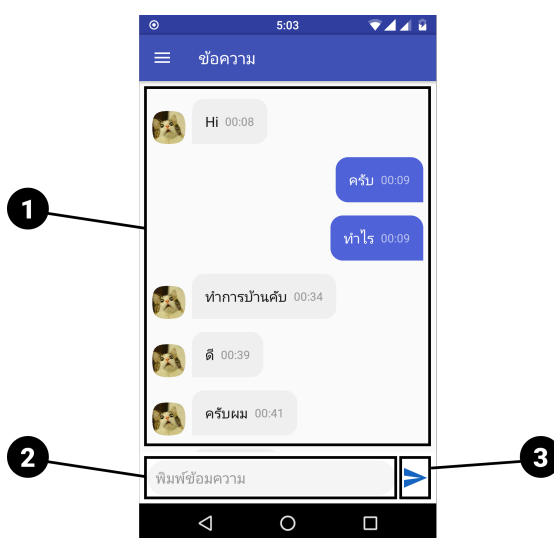
11



รูปที่ ค.11: หน้าจอเอกสาร

จากรูปที่ ค.11 สามารถอธิบายการใช้งานได้ดังนี้

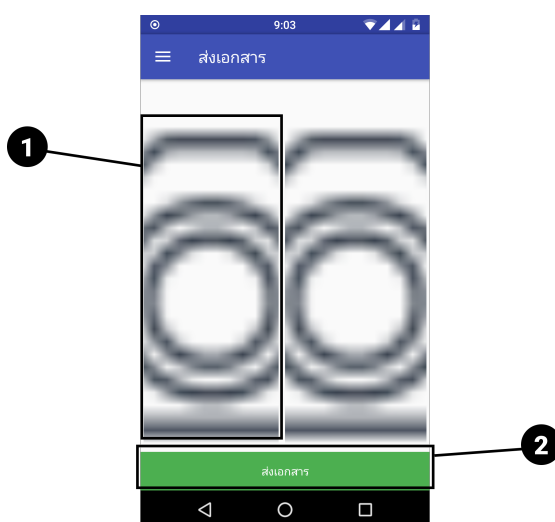
- หมายเลข 1 คือ รายการเอกสาร
- หมายเลข 2 คือ ปุ่มดาวน์โหลดเอกสาร
- หมายเลข 3 คือ สถานะการดาวน์โหลดเอกสาร
- เมื่อผู้ใช้เลือกเมนูข้อความ ระบบจะแสดงหน้าจอสนทนา ดังแสดงในรูปที่ ค.12



รูปที่ ค.12: หน้าจอสนทนา

จากรูปที่ ค.12 สามารถอธิบายการใช้งานได้ดังนี้

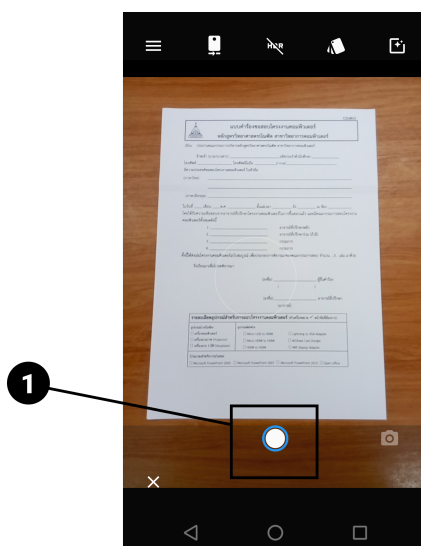
- หมายเลข 1 คือ รายการประวัติการสนทนา
- หมายเลข 2 คือ ช่องกรอกข้อความเพื่อสนทนา
- หมายเลข 3 คือ ปุ่มส่งข้อความ
- เมื่อผู้ใช้เลือกเมนูส่งเอกสารระบบจะตรวจสอบข้อมูลว่าเจ้าหน้าที่ได้ทำการเปิดให้นักศึกษาส่งเอกสารได้หรือไม่ หากตรวจสอบแล้วพบว่าสามารถส่งได้ระบบจะแสดงหน้าจอส่งเอกสาร ดังแสดงในรูปที่ ค.13



รูปที่ ค.13: หน้าจอส่งเอกสาร

จากรูปที่ ค.13 สามารถอธิบายการใช้งานได้ดังนี้

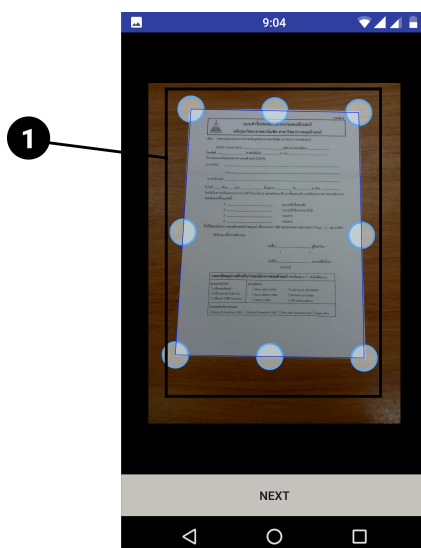
- หมายเลข 1 คือ เมื่อผู้ใช้เข้ามาครั้งแรกเมื่อผู้ใช้กดรูปภาพเอกสารระบบจะนำผู้ใช้ไปยังหน้าจอถ่ายภาพเอกสารและแสดงรูปภาพพรีวิว(Preview)ภาพถ่ายเอกสาร
- หมายเลข 2 คือ ปุ่มกดส่งเอกสาร
- เมื่อผู้ใช้กดที่ปุ่มถ่ายภาพเอกสารระบบจะแสดงหน้าจอถ่ายภาพเอกสาร ดังแสดงในรูปที่ ค.14



รูปที่ ค.14: หน้าจถ่ายภาพเอกสาร

จากรูปที่ ค.14 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ ปุ่มกดถ่ายภาพสำเนาเอกสาร
- หน้าจอแสดงภาพพรีวิวภาพถ่ายสำเนาเอกสาร ดังแสดงในรูปที่ ค.15

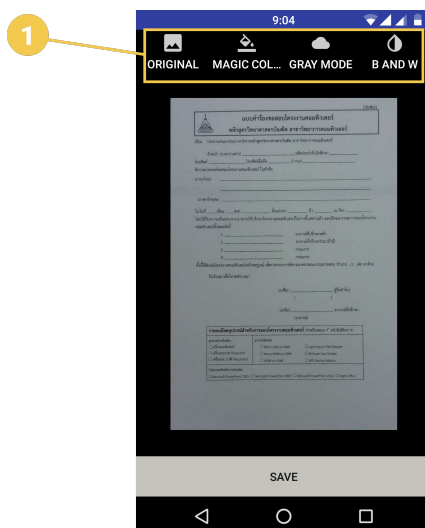


รูปที่ ค.15: หน้าจอแสดงภาพพรีวิวภาพถ่ายสำเนาเอกสาร

จากรูปที่ ค.15 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ ปุ่มปรับมุมภาพทั้ง 8 มุมเพื่อปรับขนาดภาพถ่ายเอกสาร

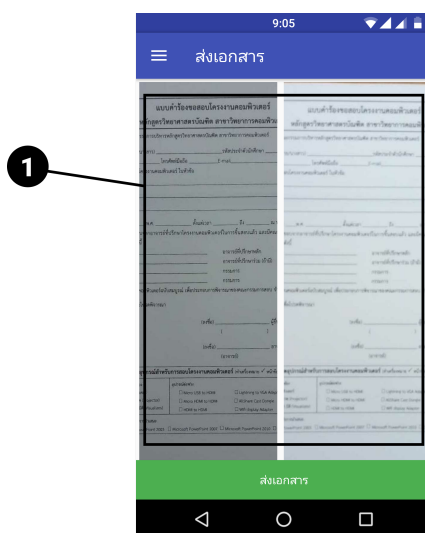
- หน้าจอแสดงปรับแต่งภาพถ่ายสำเนาเอกสาร ดังแสดงในรูปที่ ค.16



รูปที่ ค.16: หน้าจอแสดงปรับแต่งภาพถ่ายสำเนาเอกสาร

จากรูปที่ ค.16 สามารถอธิบายการใช้งานได้ดังนี้

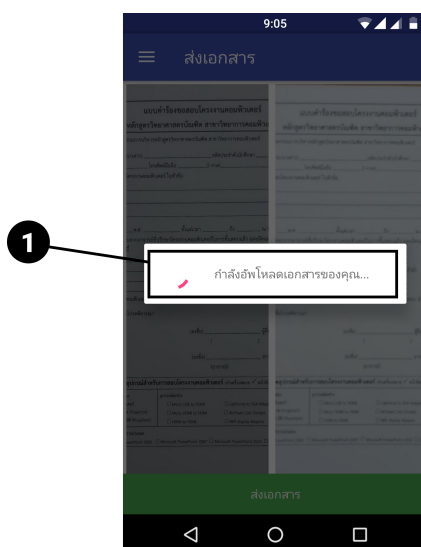
- หมายเลข 1 คือ เมื่อผู้ใช้งานต้องการปรับความคมชัดของภาพสามารถทำได้จาก 4 ปุ่ม คือ ปุ่มภาพต้นฉบับ ปุ่มปรับสีอัตโนมัติ ปุ่มปรับปรับสีเป็นสีเทา ปุ่มปรับสีเป็นสีขาวดำ
- หน้าจอแสดงภาพเปรียบเทียบภาพถ่ายสำเนาเอกสารทั้งสองฉบับ ดังแสดงในรูปที่ ค.17



รูปที่ ค.17: หน้าจอแสดงภาพเปรียบเทียบภาพถ่ายสำเนาเอกสาร

จากรูปที่ ค.17 สามารถอธิบายการใช้งานได้ดังนี้

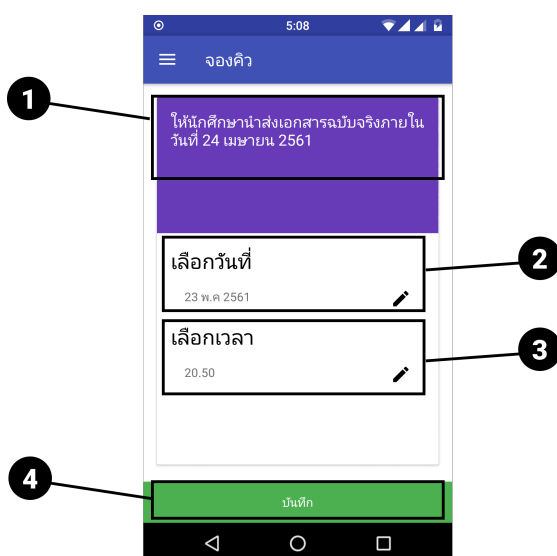
- หมายเลข 1 คือ แสดงภาพปริวิภาพถ่ายสำเนาเอกสารทั้งสองฉบับ
- เมื่อผู้ใช้กดปุ่มส่งเอกสาร ระบบจะแสดงหน้าต่างแสดงสถานะการอัปโหลดเอกสาร ดังแสดงในรูปที่ ค.18



รูปที่ ค.18: หน้าต่างแสดงสถานะการอัปโหลดเอกสาร

จากรูปที่ ค.18 สามารถอธิบายการใช้งานได้ดังนี้

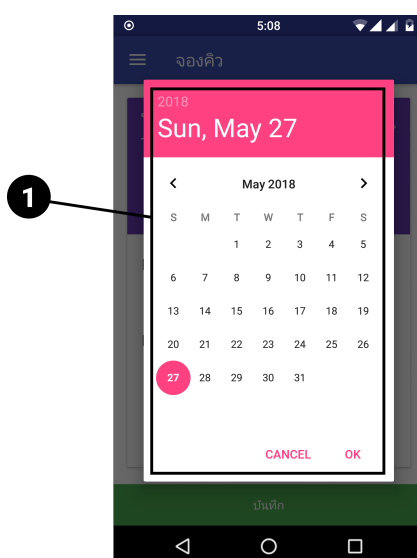
- หมายเลข 1 คือ หน้าต่างสถานะการอัปโหลดเอกสาร
- เมื่อผู้ใช้กดเมนูจองคิว ระบบจะตรวจสอบสถานะการส่งเอกสารว่าถูกเจ้าหน้าที่ตรวจสอบแล้วหรือไม่ ถ้าเจ้าหน้าที่อนุมัติแล้วระบบจะแสดงหน้าจอจองวันที่ส่งเอกสาร ดังแสดงในรูปที่ ค.19



รูปที่ ค.19: หน้าจองคิว

จากรูปที่ ค.19 สามารถอธิบายการใช้งานได้ดังนี้

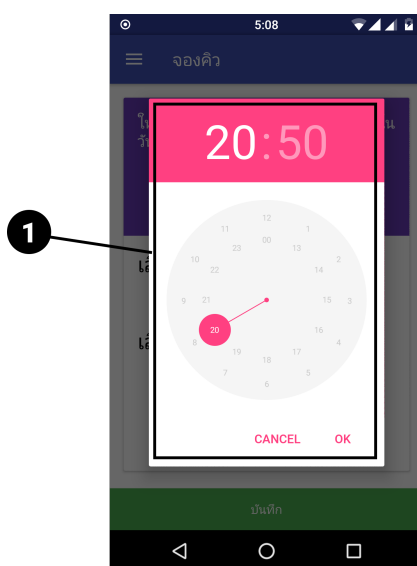
- หมายเลข 1 คือ รายละเอียด
  - หมายเลข 2 คือ ส่วนของการเลือกวันที่ที่ต้องการส่งเอกสาร
  - หมายเลข 3 คือ ส่วนของการเลือกเวลาที่ต้องการส่งเอกสาร
  - หมายเลข 4 คือ ปุ่มกดส่งเอกสาร
- เมื่อผู้ใช้กดปุ่มเลือกวันที่ที่ต้องการส่งเอกสารระบบจะแสดงหน้าต่างเลือกวันที่โดยจะแสดงเฉพาะวันที่ที่เจ้าหน้าที่เลือกไว้เท่านั้น ดังแสดงในรูปที่ ค.20



รูปที่ ค.20: หน้าต่างปฏิทินเลือกวันที่ต้องการส่งเอกสาร

จากรูปที่ ค.20 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ หน้าต่างปฏิทินเลือกวันที่ต้องการส่งเอกสาร
- เมื่อผู้ใช้งานกดปุ่มเลือกเวลาที่ต้องการส่งเอกสารระบบจะแสดงหน้าต่างเลือกเวลาโดยจะแสดงเฉพาะเวลาที่เจ้าหน้าที่เลือกไว้เท่านั้น ดังแสดงในรูปที่ ค.21



รูปที่ ค.21: หน้าต่างนาฬิกาเลือกเวลาที่ต้องการส่งเอกสาร

จากรูปที่ ค.21 สามารถอธิบายการใช้งานได้ดังนี้



- หมายเลข 1 คือ หน้าต่างนาฬิกาเลือกเวลาที่ต้องการส่งเอกสาร
- เมื่อผู้ใช้กดเมนูคำถามที่พบบ่อยระบบจะแสดงหน้าจอคำถามที่พบบ่อย ดังแสดงในรูปที่ ค.22



รูปที่ ค.22: หน้าจอคำถามที่พบบ่อย

จากรูปที่ ค.22 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ คำถาม
- หมายเลข 2 คือ คำตอบ
- เมื่อผู้ใช้กดปุ่มเมนูเกี่ยวกับเราในระบบจะ แสดงข้อมูลรายละเอียดของงานพัฒนานักศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี ดังแสดงในรูปที่ ค.23



รูปที่ ค.23: หน้าเกี่ยวกับ

จากรูปที่ ค.23 สามารถอธิบายการใช้งานได้ดังนี้

- หมายเลข 1 คือ ข้อมูลติดต่องานพัฒนานักศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี
- ปุ่มกดสำหรับเปิดกลุ่มงานพัฒนานักศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานีบน Facebook

## ประวัติผู้พัฒนา

ชื่อ-สกุล: นางสาวปิยพร อารศรี

รหัสประจำตัวนักศึกษา: 59110440259

วันเกิด: 12 06 2540

ที่อยู่ที่สามารถติดต่อได้: 7 ม.1 ต.หนองบก อ.เหล่าเสือโก้ก จ.อุบลราชธานี 34000

เบอร์โทรศัพท์: (+66) 99 468 2013

อีเมลล์: piyaphorn.ar.59@ubu.ac.th

ระดับมัธยมต้น: โรงเรียนหกลีปรรชาวิทยาคมอุบลราชธานี จังหวัดอุบลราชธานี

ระดับมัธยมปลาย: โรงเรียนหกลีปรรชาวิทยาคมอุบลราชธานี จังหวัดอุบลราชธานี

ระดับอุดมศึกษา: ภาควิชาคณิตศาสตร์ สถิติ และคอมพิวเตอร์ สาขาวิทยาการ คอมพิวเตอร์ คณะ  
วิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี