

ระบบสแกนใบหน้าสำหรับแคชเชียร์  
Recognition for cashier

นางสาวภัทราภรณ์ อินทา

โครงงานนี้เป็นส่วนหนึ่งของการศึกษา  
หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์  
ภาควิชาคณิตศาสตร์ สถิติ และคอมพิวเตอร์ คณะวิทยาศาสตร์  
มหาวิทยาลัยอุบลราชธานี  
ปีการศึกษา 2562  
ลิขสิทธิ์ของมหาวิทยาลัยอุบลราชธานี

โครงการ : ระบบสแกนใบหน้าสำหรับแคชเชียร์  
Recognition for cashier  
โดย : นางสาวภัทราภรณ์ อินทา  
อาจารย์ที่ปรึกษา : ดร.ไพชยนต์ คงไชย  
ระดับการศึกษา : วิทยาศาสตร์บัณฑิต สาขาวิทยาการคอมพิวเตอร์  
ปีการศึกษา : 2562

---

ได้รับการพิจารณาให้เป็นส่วนหนึ่งของการศึกษา  
ตามหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิทยาการคอมพิวเตอร์

คณะกรรมการสอบประเมินความรู้โครงการคอมพิวเตอร์

..... อาจารย์ที่ปรึกษา  
(ดร.ไพชยนต์ คงไชย)

..... กรรมการ  
(ดร. วราวุฒิ ผ้าเจริญ)

..... กรรมการ  
(ดร. ทศพร จูนิม)

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

วันที่ ... / ... / ...

## กิตติกรรมประกาศ

การพัฒนาโครงการระบบ สแกนใบหน้าสำหรับแคชเชียร์ สำเร็จลุล่วงได้ด้วยความกรุณาและความช่วยเหลือจากหลายๆ ท่าน ข้าพเจ้าขอขอพระคุณทุกท่าน ที่มีส่วนร่วมในการพัฒนาโครงการนี้

ขอขอบพระคุณอาจารย์ที่ปรึกษา ดร.ไพชยนต์ คงไชย อาจารย์ที่ปรึกษาโครงการที่ได้แนะนำทฤษฎีและแนวทางในแก้ปัญหาต่าง ๆ ที่เกิดขึ้นระหว่างการพัฒนา ระบบ อีกครั้งยังคอยตรวจสอบความก้าวหน้าของการทำงานเป็นระยะ ๆ รวมทั้งสร้างกำลังใจให้ผู้พัฒนาอยู่เสมอ

ขอขอบพระคุณเจ้าหน้าที่ภาควิชาคณิตศาสตร์ สถิติและคอมพิวเตอร์ ที่คอยเอื้ออำนวยความสะดวกทั้งเรื่องอุปกรณ์และสถานที่ต่อการปฏิบัติงานของผู้พัฒนา

ขอกราบขอบพระคุณบิดา มารดา ที่คอยให้กำลังใจ คอยให้ความรักและความห่วงใยเสมอมา ตลอดจนคอยช่วยเหลือทุนทรัพย์ทางด้านการศึกษาและอุปกรณ์ในการพัฒนาโครงการ

นางสาวภัทราภรณ์ อินทา

เมษายน 62

โครงการ	:	ระบบสแกนใบหน้าสำหรับแคชเชียร์
โดย	:	นางสาวภัทราภรณ์ อินทา
อาจารย์ที่ปรึกษา	:	ดร.ไพชยนต์ คงไชย
ระดับการศึกษา	:	วิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
ปีการศึกษา	:	2562

---

## บทคัดย่อ

แอปพลิเคชันสแกนใบหน้าสำหรับแคชเชียร์(Face Recognition for Cashier) แอปพลิเคชันสำหรับการชำระเงินของลูกค้า โดยจะมีการสแกนใบหน้าเพื่อเก็บข้อมูลของลูกค้าที่มาใช้บริการ และชำระสินค้าในครั้งแรกแล้วต้องการสมัครสมาชิก และการสแกนใบหน้านี้อาจใช้แทนบัตรสมาชิกสำหรับลูกค้าที่เป็นสมาชิกแล้ว เมื่อสแกนใบหน้าแล้วจะมีการแสดงข้อมูลในระบบ จะประกอบด้วย ประวัติลูกค้า แต้มสะสม(point) สำหรับลูกค้าที่เป็นสมาชิกเท่านั้น และยังมีการแสดงประวัติรายการสั่งซื้อ ส่วนที่ไม่เป็นสมาชิกจะไม่มีเก็บข้อมูลลูกค้า จะสามารถสั่งซื้อสินค้าเพียงเท่านั้น จุดเด่นของแอปพลิเคชันจะมีเนื้อหาชี้ให้เห็นได้ชัดเจนถึงการสแกนใบหน้าเพื่อเก็บข้อมูลของลูกค้า และบันทึกข้อมูล ประวัติการสั่งซื้อสินค้าของลูกค้าทุกครั้ง เพื่อนำไปใช้ในฟังก์ชันแนะนำเมนูให้ลูกค้าเพื่อช่วยให้ลูกค้าตัดสินใจได้ง่ายขึ้น

คำสำคัญ: เว็บแอปพลิเคชัน

Topic : Recognition for cashier  
Author : PATTARAPORN INTA  
Advisor : PAICHAYON KONGCHAI , Ph.D.  
Degree : Bachelor of Science (Computer Science)  
Academic Year : 2019

---

## Abstract

Face Recognition for Cashier, an application for payment of customers There will be a face scan to collect the information of customers who use the service and pay the product for the first time. And this face scan will be used as a membership card for members already Once the face has been scanned, data will be displayed in the system, consisting of customer history Accumulate (point) for customers that are members only And also showing order history for the past 1 month. Non-members will have order history only And the customer's face only There will be no accumulation (point) when passing the face scan. And the system will display the customer information The order history will show a list of past products. And to the order function And the menu navigation function for customers In order to help customers make decisions quickly Customers must go through a face scan before accessing the various functions. The highlight of the application is the content that clearly shows the face scan to collect customer data and save data. This purchase history of every customer's product. To be used in the menu suggestion function for customers to help customers make easier decisions

Keywords: Web Application

# 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 Python . . . . .	5
2.2 ความรู้พื้นฐาน Django Framework . . . . .	5
2.3 ความรู้พื้นฐานเกี่ยวกับ Bootstrap . . . . .	7
2.4 ความรู้พื้นฐานเกี่ยวกับ OpenCV(OpenSourceComputerVision) . . . . .	7
2.4.1 การประมวลผลภาพ (Image Processing) . . . . .	7
2.4.2 Machine Learning . . . . .	10
2.5 ความรู้พื้นฐานเกี่ยวกับ Pillow . . . . .	11
2.6 MySQL . . . . .	12
2.7 XAMPP . . . . .	13
2.8 ความรู้พื้นฐานเกี่ยวกับปัญญาประดิษฐ์ (Artificial Intelligence) . . . . .	14

2.8.1	โครงข่ายประสาทเทียม (Neural Network)	14
2.9	งานวิจัยที่เกี่ยวข้อง	15
3	การวิเคราะห์และออกแบบระบบ	16
3.1	โครงสร้างภาพรวมของระบบ	17
3.2	System Requirements	18
3.2.1	Functional Requirements	18
3.2.2	Non-functional Requirements	18
3.3	User Interface Design	18
3.4	Use Case Diagram	22
3.5	Class Diagram	27
3.6	Sequence Diagram	37
3.7	โครงสร้างฐานข้อมูลไฟร์เบส(Firebase Database Structure)	48
4	การพัฒนาระบบ	61
4.1	การพัฒนาเว็บแอปพลิเคชัน	61
4.1.1	การเชื่อมต่อ Cloud Firestore	61
4.1.2	โครงสร้างของการสร้างหน้าเข้าสู่ระบบ	63
4.1.3	โครงสร้างของการสร้างหน้าข่าวสาร	66
4.1.4	โครงสร้างของการสร้างหน้าดูรายละเอียดข่าวสาร	68
4.1.5	โครงสร้างของการสร้างหน้าสนทนา	70
4.1.6	โครงสร้างของการสร้างหน้าปฏิทินแสดงกำหนดการ	73
4.1.7	โครงสร้างของการสร้างหน้าสร้างประกาศ	75
4.1.8	โครงสร้างของการสร้างหน้าอัปโหลดเอกสารที่เกี่ยวข้อง	78
4.1.9	โครงสร้างของการสร้างหน้าสร้างกำหนดการจองคิวส่งเอกสาร	80
4.2	การพัฒนาเว็บแอปพลิเคชันแอนดรอยด์แอปพลิเคชัน	81
4.2.1	โครงสร้างของการสร้างหน้า MainActivity	82
4.2.2	โครงสร้างของการสร้างหน้า FeedFragment	84
4.2.3	โครงสร้างของการสร้างหน้า PostDetailActivity	87
4.2.4	โครงสร้างของการสร้างหน้า ChatActivity	88
4.2.5	โครงสร้างของการสร้างหน้า SignInActivity	92

4.2.6	โครงสร้างของการสร้างหน้า ScheduleFragment . . . . .	94
4.2.7	โครงสร้างของการสร้างหน้า ScheduleFragment . . . . .	96
5	การทดสอบระบบ . . . . .	97
5.1	การทดสอบการใช้งานแอนดรอยด์แอปพลิเคชัน . . . . .	97
5.2	การทดสอบการใช้งานเว็บแอปพลิเคชัน . . . . .	105
6	สรุปและข้อเสนอแนะ . . . . .	111
6.1	สรุปความสามารถของระบบ . . . . .	111
6.1.1	เว็บแอปพลิเคชัน . . . . .	111
6.1.2	แอนดรอยด์พลิเคชัน . . . . .	112
6.2	ปัญหาและอุปสรรคในการพัฒนา . . . . .	112
6.3	แนวทางการพัฒนาต่อ . . . . .	113
	บรรณานุกรม . . . . .	114
	ประวัติผู้พัฒนา . . . . .	115



## สารบัญตาราง

ตารางที่	หน้า
1.1 ขั้นตอนการดำเนินงาน . . . . .	4
3.1 สัญลักษณ์ของ Use case Diagram . . . . .	22
3.2 อธิบาย Use Case หน้าที่ของระบบ ในภาพที่ 3.6 . . . . .	24
3.3 Use Caseสมัครสมาชิก . . . . .	24
3.4 Use Case สแกนใบหน้าลูกค้า . . . . .	25
3.5 Use Case แสดงข้อมูลลูกค้า . . . . .	25
3.6 Use Case แนะนำเมนู . . . . .	26
3.7 Use Case สั่งซื้อสินค้า . . . . .	26
3.8 สัญลักษณ์ของ Class Diagram . . . . .	27
3.9 อธิบาย Class Diagram ของคลาสพื้นฐานของระบบ . . . . .	33
3.10 อธิบาย Class Diagram ของส่วนของการแสดงข่าวสาร . . . . .	34
3.11 อธิบาย Class Diagram ของส่วนของการแสดงรายการเอกสารในระบบ . . . . .	35
3.12 อธิบาย Class Diagram ของส่วนของการส่งสำเนาเอกสาร . . . . .	36
3.13 อธิบาย Class Diagram ของส่วนของการสนทนา . . . . .	36
3.14 สัญลักษณ์ของ Sequence Diagram . . . . .	37
3.15 สัญลักษณ์ของโครงสร้างฐานข้อมูลแบบ Firebase . . . . .	48
3.16 อธิบายโหมดที่ใช้เก็บข้อมูลประกาศ . . . . .	53
3.17 อธิบายโหมดที่ใช้เก็บข้อมูลเอกสารที่เกี่ยวข้อง . . . . .	54
3.18 อธิบายโหมดที่ใช้เก็บข้อมูลประวัติการสนทนา . . . . .	55
3.19 อธิบายโหมดที่ใช้เก็บข้อมูลกำหนดการ . . . . .	56
3.20 อธิบายโหมดที่ใช้เก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา . . . . .	57
3.21 อธิบายโหมดที่ใช้เก็บข้อมูลของนักศึกษา . . . . .	58
3.22 อธิบายโหมดที่ใช้เก็บข้อมูลการจองคิวของนักศึกษา . . . . .	59
3.23 อธิบายโหมดที่ใช้เก็บข้อมูลคำถามที่พบบ่อย . . . . .	60
5.1 ผลการทดสอบเมนูนำทาง . . . . .	98
5.2 ผลการทดสอบเมนูนำทาง(ต่อ) . . . . .	99
5.3 ผลการทดสอบหน้ารายละเอียดประกาศ . . . . .	100
5.4 ผลการทดสอบหน้าสนทนา . . . . .	101
5.5 ผลการทดสอบหน้าปฏิทินกำหนดการ . . . . .	102
5.6 ผลการทดสอบหน้าดาวน์โหลดเอกสาร . . . . .	103
5.7 ผลการทดสอบหน้าส่งภาพถ่ายสำเนาเอกสาร . . . . .	104
5.8 ผลการทดสอบหน้าจองคิวส่งเอกสาร . . . . .	105
5.9 ผลการทดสอบเมนูนำทาง . . . . .	106
5.10 ผลการทดสอบหน้ารายละเอียดประกาศ . . . . .	107
5.11 ผลการทดสอบหน้าสนทนา . . . . .	108

5.12 ผลการทดสอบหน้าปฏินิกำหนดการ . . . . .	109
5.13 ผลการทดสอบหน้าดาวนโหลดเอกสาร . . . . .	110

## สารบัญภาพ

รูปที่	หน้า
2.1	รูปแบบ MVC ของ Django . . . . . 6
2.2	ภาพแบบ Binary . . . . . 8
2.3	ภาพแบบ Grayscale Image . . . . . 8
2.4	ภาพแบบ RGB Image . . . . . 9
2.5	สูตรการหาความสัมพันธ์ Association Rule . . . . . 10
2.6	การขยายรูปโดยใช้ Pillow . . . . . 11
2.7	ขั้นตอนของโครงข่ายประสาทเทียมแบบคอนโวลูชันนัล . . . . . 15
3.1	ภาพรวมและโครงสร้างการทำงานของระบบ . . . . . 17
3.2	หน้าเข้าสู่ระบบสำหรับพนักงาน . . . . . 19
3.3	หน้าจอหลัก . . . . . 19
3.4	หน้าจอแสดงข้อมูลลูกค้า . . . . . 20
3.5	หน้าจอสั่งซื้อสินค้า . . . . . 20
3.6	Use Case diagram ของระบบ สแกนใบหน้าสำหรับแคชเชียร์ . . . . . 23
3.7	Class Diagram ของแอปพลิเคชันระบบสแกนใบหน้าสำหรับแคชเชียร์ . . . . . 28
3.8	Class Diagram ของแอปพลิเคชันระบบ XX . . . . . 29
3.9	Class Diagram ของแอปพลิเคชันระบบ XX . . . . . 30
3.10	Class Diagram ของแอปพลิเคชันระบบ XX . . . . . 31
3.11	Class Diagram ของแอปพลิเคชันระบบ XX . . . . . 32
3.12	Sequence Diagram การแสดงข่าวสาร . . . . . 38
3.13	Sequence Diagram การแสดงปฏิทินกำหนดการ . . . . . 40
3.14	Sequence Diagram การแสดงดาวนโหลดเอกสาร . . . . . 42
3.15	Sequence Diagram การแสดงบทสนทนา . . . . . 44
3.16	Sequence Diagram แสดงส่งเอกสารตรวจสอบ . . . . . 46
3.17	โครงสร้างฐานข้อมูลแบบ Firebase . . . . . 49
3.18	โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ) . . . . . 50
3.19	โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ) . . . . . 51
3.20	โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ) . . . . . 52
3.21	โน้ตเก็บข้อมูลประกาศ . . . . . 53
3.22	โน้ตเก็บข้อมูลเอกสารที่เกี่ยวข้อง . . . . . 54
3.23	โน้ตเก็บข้อมูลประวัติการสนทนา . . . . . 55
3.24	โน้ตเก็บข้อมูลกำหนดการ . . . . . 56
3.25	โน้ตเก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา . . . . . 57
3.26	โน้ตเก็บข้อมูลของนักศึกษา . . . . . 58
3.27	โน้ตเก็บข้อมูลการจองคิวของนักศึกษา . . . . . 59
3.28	โน้ตเก็บข้อมูลคำถามที่พบบ่อย . . . . . 60

4.1	ไฟล์ firebaseConfig.js . . . . .	61
4.2	ไฟล์ firebaseInit.js . . . . .	62
4.3	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue . . . . .	63
4.4	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue . . . . .	64
4.5	การสร้างลอจิก (logic) ของหน้าเข้าสู่ระบบ SignIn.vue . . . . .	65
4.6	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าข่าวสาร Home.vue . . . . .	66
4.7	การสร้างลอจิก(logic)ของหน้าข่าวสาร Home.vue . . . . .	67
4.8	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้ารายละเอียดข่าวสาร ViewPost.vue . . . .	68
4.9	การสร้างลอจิกของหน้าดูรายละเอียดของข่าวสาร ViewPost.vue . . . . .	68
4.10	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสนทนา Message.vue . . . . .	70
4.11	การสร้างลอจิกของหน้าสนทนา Message.vue . . . . .	72
4.12	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าปฏิทินกำหนดการ Schedule.vue . . . . .	73
4.13	การสร้างลอจิกของหน้าปฏิทินกำหนดการ Schedule.vue . . . . .	74
4.14	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างประกาศ MgPost.vue . . . . .	75
4.15	การสร้างลอจิกของหน้าหน้าสร้างประกาศ MgPost.vue . . . . .	77
4.16	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าอัปโหลดเอกสารที่เกี่ยวข้อง MgDocument.vue	78
4.17	การสร้างลอจิกของหน้าหน้าอัปโหลดเอกสารที่เกี่ยวข้อง MgDocument.vue . . .	79
4.18	การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างกำหนดการจองคิวส่งเอกสาร MgQueue.vue	80
4.19	การสร้างลอจิกของหน้าสร้างกำหนดการจองคิวส่งเอกสาร MgQueue.vue . . . . .	81
4.20	ตัวแปรในคลาส MainActivity . . . . .	82
4.21	โค้ดส่วนที่ใช้ในการสร้างเมนูนำทางหลักภายในคลาส MainActivity . . . . .	83
4.22	ตัวแปรในคลาส FeedFragment . . . . .	84
4.23	โค้ดส่วนที่ใช้ในการสืบค้นข้อมูลจาก Cloude Firestore ภายในคลาส FeedFragment	85
4.24	โค้ดส่วนที่ใช้ในการดักอีเวนต์เมื่อผู้ใช้กดที่แถวประกาศในคลาส FeedFragment . .	86
4.25	โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส PostDetailActivity . .	87
4.26	โค้ดส่วนที่ใช้ในการดาวน์โหลดเอกสารของคลาส PostDetailActivity . . . . .	88
4.27	โค้ดส่วนที่ใช้ในการแสดงผลรายละเอียดประกาศของคลาส ChatActivity . . . . .	88
4.28	โค้ดส่วนที่ใช้ในการสืบค้นประวัติการสนทนาของคลาส ChatActivity . . . . .	90
4.29	โค้ดส่วนที่ใช้ในการส่งข้อความของคลาส ChatActivity . . . . .	91
4.30	โค้ดส่วนที่ใช้ในการแสดงผลหน้าเข้าสู่ระบบของคลาส SignInActivity . . . . .	92
4.31	โค้ดส่วนที่ใช้ในการเข้าสู่ระบบของคลาส SignInActivity . . . . .	93
4.32	โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment . . . . .	94
4.33	โค้ดส่วนที่ใช้ในการสืบค้นข้อมูลกำหนดการของคลาส ScheduleFragment . . . .	95
4.34	โค้ดส่วนที่ใช้ในการแสดงผลหน้าปฏิทินของคลาส ScheduleFragment . . . . .	96

# บทที่ 1

## บทนำ

### 1.1 ที่มาและเหตุผล

ปัจจุบันเทคโนโลยีเกี่ยวกับการประมวลผลภาพได้ถูกนำไปประยุกต์ใช้มากมาย เช่น การสแกนใบหน้าเพื่อเข้าใช้งานโทรศัพท์มือถือ การสแกนลายนิ้วมือ อีกทั้งการนำไปจดจำใบหน้าเพื่อเก็บข้อมูล และนำข้อมูลมาใช้ในครั้งต่อไป และเทคโนโลยีการประมวลผลภาพนี้ยังเป็นที่ยอมรับในสังคมตอนนี้ เพราะมันง่ายต่อการใช้งาน ไม่ต้องมีอุปกรณ์ที่ช่วยในการเข้าถึงข้อมูล ใช้แค่ใบหน้าในการตรวจสอบ อีกทั้งยังมีการนำ Machine learning มาใช้ในการช่วยตัดสินใจในการสั่งสินค้าของลูกค้า

ด้วยเหตุนี้จึงมีความประสงค์ที่จะนำเทคโนโลยีการประมวลผลภาพมาใช้ เพื่อที่จะบันทึกใบหน้าของลูกค้าที่มาชำระเงิน เพื่อนำข้อมูลมาใช้ในครั้งต่อไป แทนการใช้อุปกรณ์ที่ช่วยในการเข้าถึงข้อมูลลูกค้าที่ยุ่งยาก และสะดวกต่อลูกค้าเพื่อช่วยแก้ไขการลืมบัตรสมาชิก สะกดชื่อไม่ถูก ลืมเบอร์โทรศัพท์ที่ใช้ในการสมัคร และช่วยในการตัดสินใจสั่งสินค้าได้รวดเร็วขึ้น โดยใช้ Decision Tree มาช่วยในการแนะนำเมนู

### 1.2 วัตถุประสงค์

1. เพื่อออกแบบและพัฒนาแอปพลิเคชัน จดจำใบหน้าในการชำระเงิน
2. เพื่อดูข้อมูล และประวัติการสั่งซื้อที่ผ่านมาของลูกค้าได้ โดยผ่านการสแกนใบหน้าลูกค้า
3. เพื่อแนะนำ เมนูแก่ลูกค้า เวลาที่ลูกค้าไม่รู้ว่าจะสั่งอะไร
4. ใช้การสแกนใบหน้าแทนการใช้บัตรสมาชิก
5. ช่วยในการตัดสินใจ การสั่งซื้อสินค้าของลูกค้าได้
6. เพื่อช่วยลดทรัพยากรที่ใช้ในการผลิตบัตรสมาชิก และอุปกรณ์อื่นๆที่ต้องใช้ร่วมกัน

### 1.3 ขอบเขตของโครงการ

1. ใช้สำหรับพนักงานแคชเชียร์เท่านั้น

2. เพื่อดูข้อมูล และประวัติการสั่งซื้อที่ผ่านมาของลูกค้าได้อย่างรวดเร็ว โดยผ่านการสแกนใบหน้าลูกค้าเท่านั้น
3. ไม่ต้องมีบัตร ก็สามารถดูข้อมูลต่างๆของลูกค้าได้ จากการสแกนใบหน้า
4. ช่วยในการตัดสินใจการสั่งซื้อสินค้าของลูกค้าได้
5. พนักงานแคชเชียร์สามารถส่งสินค้าให้ลูกค้าได้เท่านั้น

#### 1.4 ประโยชน์ที่คาดว่าจะได้รับ

1. ช่วยลดปริมาณทรัพยากรที่ใช้ในการผลิตบัตร
2. ช่วยให้พนักงานแคชเชียร์ทำงานได้ถูกต้องและแม่นยำ
3. ช่วยลดปัญหาลูกค้าลืมบัตรสมาชิก ลืมชื่อที่ใช้สมัครสมาชิก
4. สามารถแนะนำเมนูให้ลูกค้าโดยเทียบจากประวัติการสั่งซื้อ

#### 1.5 เครื่องมือที่ใช้ในการพัฒนา (Development tools)

##### 1.5.1 ฮาร์ดแวร์

##### 1. สมาร์ทโฟน (Smart phone)

- ทำงานบนระบบปฏิบัติการแอนดรอยด์เวอร์ชัน 5.0 หรือ API Level 21
- หน่วยประมวลผลกลาง Mediatek MT6753 Octa-core ความเร็ว 1.3 กิกะเฮิร์ตซ์ (Gigahertz, GHz)
- หน่วยประมวลผลกราฟฟิกลำดับที่น้อย Mali-T720MP3
- หน่วยความจำหลักอย่างน้อย 2 กิกะไบต์ (Gigabyte, GB)
- หน่วยความจำสำรองอย่างน้อย 16 กิกะไบต์ (Gigabyte, GB)
- หน้าจอแสดงผลความละเอียดอย่างน้อย 1080 x 1920 พิกเซล (Pixel)
- หน้าจอแสดงผลขนาดอย่างน้อย 5 นิ้ว
- กล้องถ่ายภาพความละเอียดอย่างน้อย 13 เมกะพิกเซล (Megapixel)

##### 2. เครื่องคอมพิวเตอร์ส่วนบุคคล (Personal computer)

- ทำงานบนระบบปฏิบัติการ Elementary os พื้นฐานการทำงานบน Linux
- หน่วยประมวลผลกลาง Intel Core i3-3217U ความเร็ว 1.80 กิกะเฮิร์ตซ์ (Gigahertz, GHz)

- หน่วยประมวลผลกราฟฟิก NVIDIA GeForce GT 720M ความจำ 2 กิกะไบต์ (Gigabyte, GB)
- หน่วยความจำหลัก 4 กิกะไบต์ (Gigabyte, GB)
- หน่วยความจำสำรอง 120 กิกะไบต์ (Gigabyte, GB)

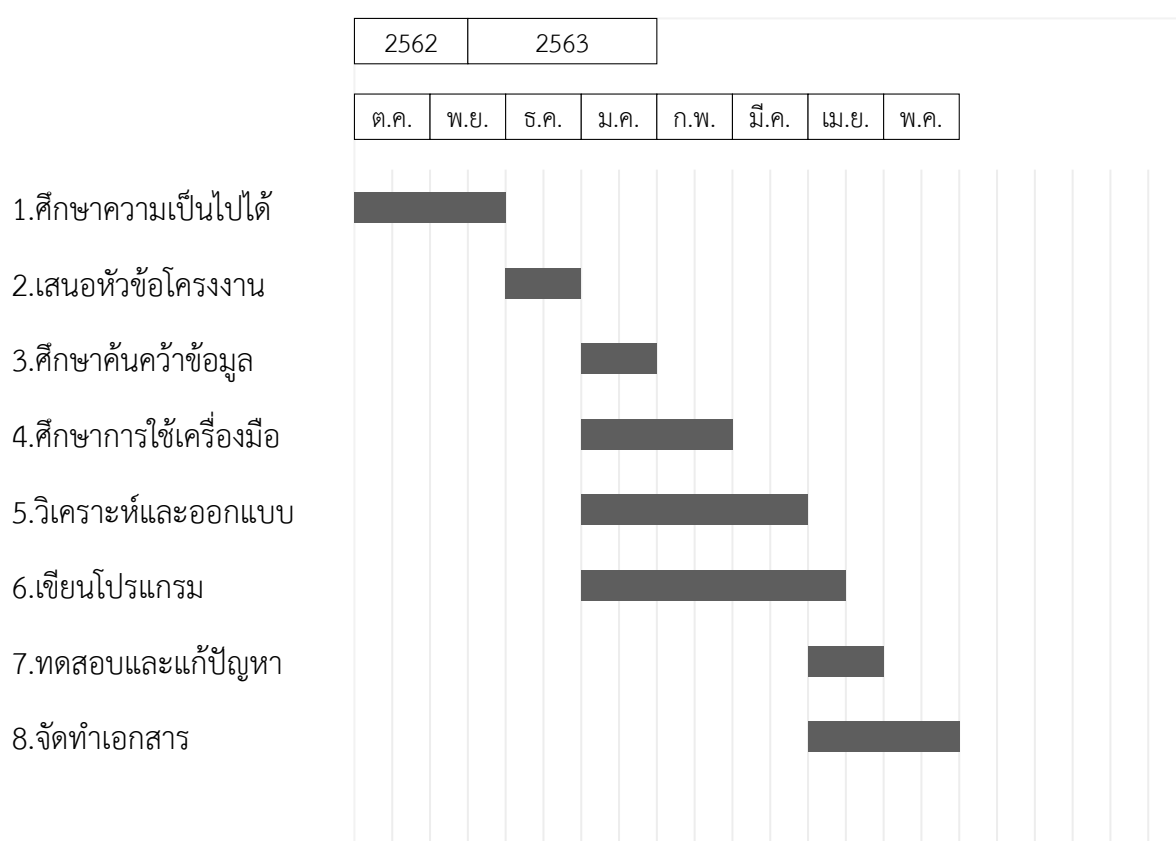
#### 1.5.2 ซอฟต์แวร์ (Software)

1. Visual Studio Code เป็นโปรแกรมสำหรับพัฒนาเว็บแอปพลิเคชัน พัฒนาด้วยภาษา HTML, JavaScript, Python และ Java
2. Django เป็น Framework ที่ใช้สำหรับสร้าง UI สำหรับระบบ หรือเรียกอีกอย่างว่า Backend Framework ใช้สำหรับเป็นเว็บเซิร์ฟเวอร์ (Web Server)
3. OpenCv เป็นไลบรารีฟังก์ชันการเขียนโปรแกรมที่มุ่งเน้นไปทาง การแสดงผลแบบเรียลไทม์ ซึ่งนำมาใช้ในส่วนของการสแกนใบหน้า
4. Machine Learning ในส่วนของการเรียนรู้ของอัลกอริทึม Association Rule
5. Sqlite3 เป็นตัวจัดการฐานข้อมูล

### 1.5.3 แผนการดำเนินการ

ในการสร้างระบบสแกนใบหน้าสำหรับแคชเชียร์ ผู้พัฒนาได้แบ่งขั้นตอนการดำเนินงานไว้ด้วยกัน 7 ขั้นตอน ดังต่อไปนี้

ตารางที่ 1.1: ขั้นตอนการดำเนินงาน





## บทที่ 2

### ทฤษฎีที่เกี่ยวข้อง

บทนี้จะเป็นรายละเอียดเกี่ยวกับทฤษฎีและงานวิจัยที่เกี่ยวข้องกับการพัฒนาโปรแกรมในครั้งนี้ โดยที่แต่ละหัวข้อจะมีความสัมพันธ์กันเป็นลำดับ โดยหัวข้อที่หนึ่งจะแนะนำความรู้เรื่อง Python เพื่อให้เข้าใจพื้นฐานเบื้องต้นเกี่ยวกับที่มาของโครงการ หัวข้อที่สองที่สามจะช่วยเตรียมให้อ่านเข้าใจเทคโนโลยีที่ใช้ในการออกแบบและพัฒนา ส่วน ...

#### 2.1 ความรู้พื้นฐานเกี่ยวกับ Python

Python คือ ภาษาเขียนโปรแกรมระดับสูงที่ใช้กันอย่างกว้างขวางในการเขียนโปรแกรมสำหรับวัตถุประสงค์ทั่วไป ภาษา Python นั้นเป็นภาษาแบบ Interpreter ที่ถูกออกแบบให้โค้ดอ่านได้ง่ายขึ้น และโครงสร้างของภาษานั้นจะทำให้โปรแกรมเมอร์สามารถเข้าใจแนวความคิดการเขียนโค้ดโดยใช้บรรทัดที่น้อยลงกว่าภาษาอย่าง C++ และ Java ซึ่งภาษานั้นถูกกำหนดให้มีโครงสร้างที่ตั้งใจให้การเขียนโค้ดเข้าใจง่าย ทั้งในโปรแกรมเล็กไปจนถึงโปรแกรมขนาดใหญ่ คุณลักษณะเด่นของภาษา Python

1. สนับสนุนแนวแบบคิดออบเจกต์โอเรียนเทด หรือ OOP (Object Oriented Programming)
2. เป็น Open Source
3. โค้ดที่เขียนด้วย Python สามารถนำไปรันบนระบบปฏิบัติการได้หลากหลาย
4. สามารถประมวลผลทางด้านวิทยาศาสตร์ และวิศวกรรมศาสตร์ได้อย่างมีประสิทธิภาพ
5. มีฟังก์ชันสนับสนุนฐานข้อมูล เช่น MySQL, Sybase, Oracle , Informix, ODBC และอื่นๆ
6. มี Library สนับสนุนด้านการสร้างภาพกราฟฟิก ตลอดจนบันทึกไฟล์ในรูปแบบต่างๆ ได้อย่างสะดวกและมีประสิทธิภาพ
7. มี Library สนับสนุนด้านปัญญาประดิษฐ์

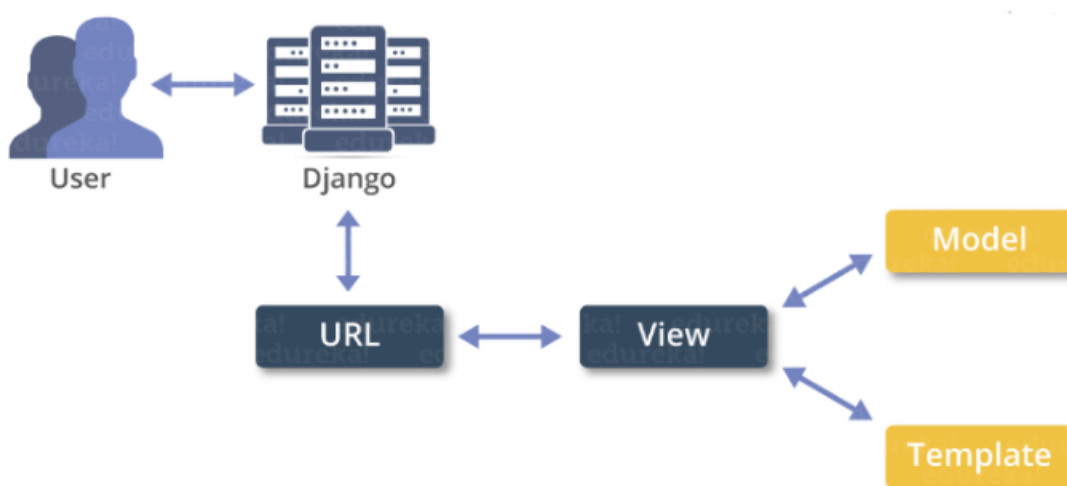
#### 2.2 ความรู้พื้นฐาน Django Framework

Django Framework คือ ชุดเครื่องมือ Framework สำหรับ การพัฒนาเว็บไซต์ด้วยภาษา Python ซึ่งความเป็นจริงแล้วทุกวันนี้มี Framework สำหรับการเขียนเว็บไซต์ด้วยภาษา Python ค่อนข้างเยอะ ซึ่ง Django Framework เป็นหนึ่งใน Framework สำหรับการพัฒนาเว็บไซต์ และ

ทำเว็บไซต์ด้วยภาษา Python ด้วยเช่นกัน โดยปัจจุบันภาษา Python นั้นค่อนข้างได้รับความนิยมเพิ่มมากขึ้น Django Framework [?] มีคุณสมบัติดังนี้

1. Object-relational mapper คือ การกำหนด Data Model ในภาษา Python เพื่อการทำงานด้านข้อมูล และสนับสนุน Dynamic database-access API
2. Automatic admin interface คือ ส่วนของการสร้าง Interface อัตโนมัติสำหรับการ add, edit , delete และ search ด้วย Django Framework
3. Elegant URL design คือ การทำให้ URL มีความสวยงาม สั้น กระชับ และสื่อความหมายของหน้านั้น ๆ ได้อย่างชัดเจน เหมาะสมกับการทำ SEO ในปัจจุบัน
4. Template system คือ Django นั้นมีการออกแบบ Template Language เพื่อการเขียนแยกส่วนระหว่าง Design และ Business Logic
5. Cache system คือ ส่วนของการบันทึก หรือจัดการข้อมูลที่มีการดาวน์โหลดไปแล้ว เพื่อเพิ่มประสิทธิภาพการทำงานของเว็บไซต์ด้านความเร็ว และด้านอื่น ๆ
6. Internationalization คือ Django สนับสนุน Application ที่มีความหลากหลายด้านภาษาในการแสดงผล

รูปแบบ MVC ของ Django เรียกอีกอย่างว่า MTV คือ Model Template View ดังรูปที่ 2.1



รูปที่ 2.1: รูปแบบ MVC ของ Django

ที่มา: <https://www.edureka.co/blog/django-tutorial/>

## 2.3 ความรู้พื้นฐานเกี่ยวกับ Bootstrap

Bootstrap คือ Front-end Framework ที่ช่วยให้สามารถสร้างเว็บแอปพลิเคชันได้อย่างสวยงาม ตัว Bootstrap มีทั้ง CSS Component และ JavaScript Plugin ให้เรียกใช้งานได้ และยังถูกออกแบบมาให้รองรับการทำงานแบบ Responsive Web ซึ่งทำให้ การพัฒนาเว็บแค่ครั้งเดียวสามารถนำไปใช้งานบนเบราว์เซอร์ใน มือถือ แท็บเล็ต และคอมพิวเตอร์ ได้ และผู้พัฒนาไม่ต้องมีความรู้ด้าน CSS มาก สามารถสร้างเว็บที่สวยงามได้เช่นกัน

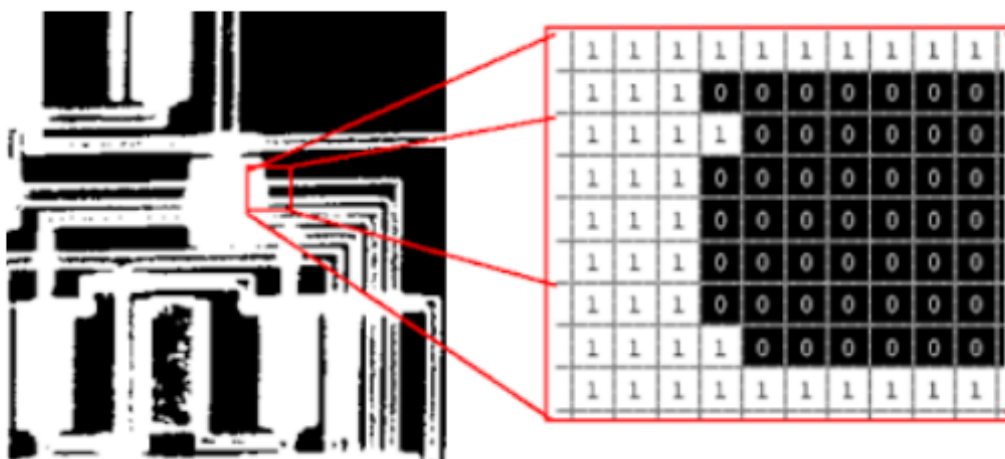
## 2.4 ความรู้พื้นฐานเกี่ยวกับ OpenCV(OpenSourceComputerVision)

OpenCV เป็น Library ที่รวบรวมฟังก์ชันสำหรับการประมวลผลภาพและคอมพิวเตอร์วิทัศน์ศาสตร์ ไว้เป็นจำนวนมากอยู่ภายใต้ใบอนุญาต BSD ซึ่งสามารถใช้ได้ฟรีทั้งทางด้าน การศึกษา และทางการค้า เนื่องจากนี้ยังมีการประยุกต์ใช้งาน OpenCV ในแบบต่างๆ ได้แก่ ระบบตรวจจับใบหน้า (Face Detection) การจดจำใบหน้า (Face Recognition) การติดตามวัตถุ (Object Tracking) การเรียนรู้ของเครื่อง (Machine Learning) เป็นต้น

### 2.4.1 การประมวลผลภาพ (Image Processing)

การประมวลผลภาพคือการนำรูปที่มีอยู่แล้ว หรือรูปที่รับเข้ามาจากอุปกรณ์ต่างๆ หรือรูป ที่มีอยู่มาประมวลผลเพื่อหาลักษณะเด่นบางประการของรูปที่มีอยู่ หรืออาจจะเป็นการตีความ หมายของภาพรวมถึงการปรับคุณลักษณะของภาพให้เป็นไปตามต้องการโดยใช้กระบวนการทาง คณิตศาสตร์ การประมวลผลภาพแนวคิดทางคณิตศาสตร์ที่ใช้ในการประมวลผล Signalprocessing มาทำการประยุกต์ใช้กับสัญญาณภาพ และภาพจะเก็บอยู่ในรูปของอาร์เรย์ (array) โดยกลุ่มของ ขออาร์เรย์กลุ่มหนึ่งจะเป็นค่าของภาพหนึ่งพิกเซล เช่น ภาพแบบ RBG ใช้อาร์เรย์ 3 ช่องเพื่อเก็บ ค่าสีของ RBG ในหนึ่งพิกเซล รูปแบบการจัดเก็บภาพแต่ละชนิดจะแตกต่างกัน ขึ้นอยู่กับระบบสี ของภาพโดยแบ่งชนิดของภาพได้ดังนี้

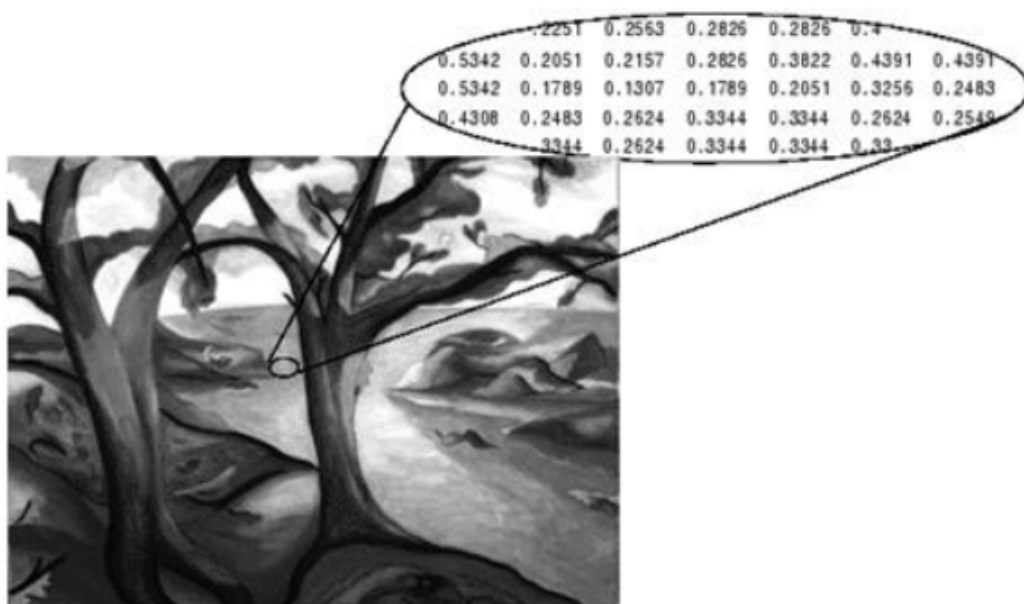
- Binary imageหรือ ภาพขาว-ดำ เป็นรูปที่ใช้เนื้อที่เพียง 1 บิต ต่อ พิกเซล โดยค่าสีจะมีแค่สองค่าคือ 0 หรือสีดำ และ 1 หรือสีขาว ดังรูปที่ 2.2



รูปที่ 2.2: ภาพแบบ Binary

ที่มา: <https://nextsoftwares.wordpress.com/2014/05/22/>

- Grayscale Image เป็นรูปที่เก็บโดยใช้รูปแบบของอาร์เรย์ 2 มิติ โดยค่าที่เก็บจะมีค่าอยู่ในช่วงๆหนึ่ง ซึ่งระดับของสีขึ้นอยู่กับขนาดของบิตที่ใช้เก็บค่าสี ดังรูปที่ 2.3



รูปที่ 2.3: ภาพแบบ Grayscale Image

ที่มา: <https://nextsoftwares.wordpress.com/2014/05/22/>

- RGB Image หรือ Tricolor Image เป็นรูปที่เก็บโดยใช้อาร์เรย์ 3 มิติ ขนาด  $m \times n \times 3$  โดยที่  $m$  คือความยาว และ  $n$  คือความกว้างของภาพในหน่วยพิกเซล ส่วนมิติสุดท้ายนั้น ในแต่ละมิติจะเก็บค่าสีแยกกัน คือสีแดง (Red) สีเขียว (Green) และสีน้ำเงิน (Blue) ดังรูปที่ 2.4



รูปที่ 2.4: ภาพแบบ RGB Image

ที่มา: <https://nextsoftwares.wordpress.com/2014/05/22/>

## 2.4.2 Machine Learning

Machine Learning คือ ส่วนการเรียนรู้ของเครื่อง ถูกใช้งานเสมือนเป็นสมองของ AI (Artificial Intelligence) เราอาจพูดได้ว่า AI ใช้ Machine Learning ในการสร้างความฉลาด มักจะใช้เรียกโมเดลที่เกิดจากการเรียนรู้ของปัญญาประดิษฐ์ ไม่ได้เกิดจากการเขียนโดยใช้มนุษย์ มนุษย์มีหน้าที่เขียนโปรแกรมให้ AI (เครื่อง) เรียนรู้จาก ข้อมูลเท่านั้น ที่เหลือเครื่องจัดการเอง Machine Learning เรียนรู้จากสิ่งที่เราส่งเข้าไปกระตุ้นแล้วจดจำเอาไว้เป็นมันสมอง ส่งผลลัพธ์ออกมาเป็นตัวเลข code หรือ ที่ส่งต่อไปแสดงผล หรือให้เจ้าตัว AI นำไปแสดงการกระทำ Machine Learning เองสามารถเอาไปใช้งานได้หลายรูปแบบ ต้องอาศัยกลไกที่เป็นโปรแกรม หรือเรียกว่า Algorithm ที่มีหลากหลายแบบโดยมี Data Scientist เป็นผู้ออกแบบ หนึ่งใน Algorithm ที่ได้รับความนิยมสูง คือ DeepLearning ซึ่งถูกออกแบบมาให้ใช้งานได้ง่าย และประยุกต์ใช้ได้หลายลักษณะงาน อย่างไรก็ตาม ในการทำงานจริงของ Data Scientist จำเป็นต้องออกแบบตัวแปรต่างๆ ทั้งในตัวของ Deep Learning เองและต้องหา Algorithm อื่นๆ มาเป็นคู่เปรียบเทียบ เพื่อมองหา Algorithm ที่เหมาะสมที่สุดในการใช้งานจริง

Algorithm ที่นำมาใช้ในการวิเคราะห์ ในส่วนของฟังก์ชันคือกฎความสัมพันธ์(Associations) การทำเหมืองข้อมูลเพื่อหาความสัมพันธ์ของเหมืองข้อมูลมักใช้ในธุรกิจการค้าปลีก(retailing Business) เช่น ร้านค้าสะดวกซื้อ หรือ ซูเปอร์มาเก็ต เป็นการวิเคราะห์การตลาด(Marketing basket Analysis) เพื่อศึกษาพฤติกรรม การซื้อสินค้าของลูกค้า และหาความสัมพันธ์ของสินค้าที่ลูกค้าซื้อ

ความสัมพันธ์ของสินค้าที่ลูกค้าซื้อจะแสดงในรูปของกฎความสัมพันธ์ Association Rule ดังนี้

$A \rightarrow B[\text{Support, Confident}]$  โดยที่ A,B แทนรายการสินค้า

สูตรการหาความสัมพันธ์

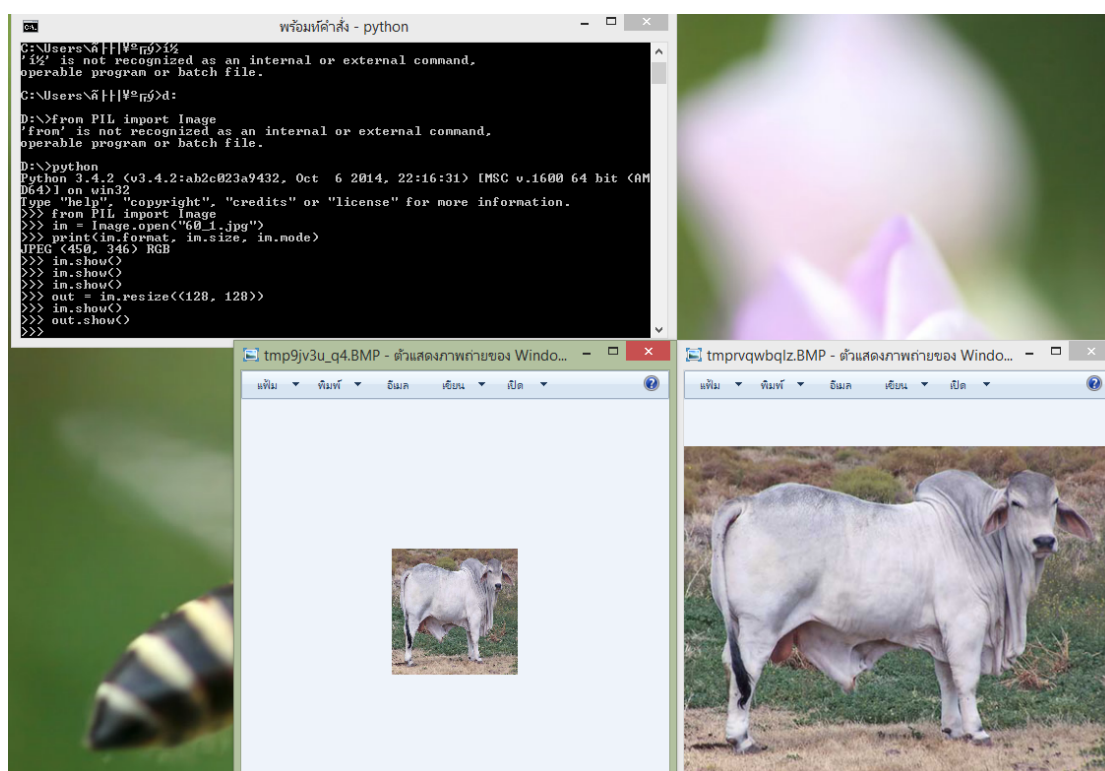
รูปที่ 2.5: สูตรการหาความสัมพันธ์ Association Rule

เช่น Milk  $\rightarrow$  Eggs [Support = 25 เปอร์เซ็นต์ ,Confident=33.34 เปอร์เซ็นต์] หมายความว่า

ว่า 25เปอร์เซ็นต์ ของทรานแซคชั่นทั้งหมด ลูกค้า จะซื้อนม (Milk) และซื้อไข่ (Eggs) พร้อมกันและ 33.34 เปอร์เซ็นต์ ของลูกค้าที่ซื้อนมแล้วจะซื้อไข่ด้วย กฎความสัมพันธ์ที่สนใจหรือกฎความสัมพันธ์ที่แข็งแกร่ง (Strong Association Rules) คือ กฎความสัมพันธ์ที่มีค่าสนับสนุน (support) และค่าความเชื่อมั่น (confidence) ผ่านเกณฑ์ขั้นต่ำ (Minimum Threshold) ที่ผู้วิเคราะห์ข้อกำหนดขึ้นมา

## 2.5 ความรู้พื้นฐานเกี่ยวกับ Pillow

Pillow เป็น Library ด้านความสามารถในการประมวลผลภาพและกราฟฟิก หรือโมดูลจัดการและประมวลผลรูปภาพบน Python ใน Python มีโมดูลด้านที่ชื่อว่า Python Imaging Library (PIL) ซึ่งรองรับแต่ Python 2 จึงมีคนมาพัฒนาเป็นโมดูล Pillow ที่รองรับทั้ง Python 2 และ Python 3 โมดูล Pillow สามารถจัดการจัดเก็บรูปภาพ (Image Archives) แสดงรูปภาพ (Image Display) ประมวลผลรูปภาพ เช่น ปรับขนาดรูปภาพ แปลงไฟล์รูปภาพ ใส่ตัวอักษรลงในภาพ และอื่น ๆ เป็นต้น ดังรูปที่ 2.6



รูปที่ 2.6: การขยายรูปโดยใช้ Pillow

ที่มา: <https://python3.wannaphong.com/2014/11/image-processing-python.html>

## 2.6 MySQL

MySQL คือ โปรแกรมระบบจัดการฐานข้อมูล ที่พัฒนาโดยบริษัท MySQL AB มีหน้าที่เก็บข้อมูลอย่างเป็นระบบ รองรับคำสั่ง SQL เป็นเครื่องมือสำหรับเก็บข้อมูลที่ต้องใช้ร่วมกับเครื่องมือหรือโปรแกรมอย่างผสมผสาน เพื่อให้ได้ระบบงานที่รองรับความต้องการของผู้ใช้ เช่น ทำงานร่วมกับเครื่องบริการเว็บ (Web Server) เพื่อให้บริการแก่ภาษาสคริปต์ที่ทำงานฝั่งเครื่องบริการ Server-Side Script เช่น ภาษา PHP ภาษา ASP.NET หรือภาษา JSP เป็นต้น หรือทำงานร่วมกับโปรแกรมประยุกต์ (Application Program) เช่น ภาษาวิซวลเบสิกดอทเน็ต (Visual Basic.NET) ภาษาจาวา (JAVA) หรือภาษาซีชาร์ป (C#) เป็นต้น โปรแกรมถูกออกแบบให้สามารถทำงานได้บนระบบปฏิบัติการที่หลากหลาย และเป็นระบบฐานข้อมูลโอเพนซอร์ส (Open Source) ที่ถูกนำไปใช้งานมากที่สุด

ความสามารถและการทำงานของโปรแกรม MySQL มีดังต่อไปนี้

- MySQL (DataBase Management System : DBMS) เป็นฐานข้อมูลที่มีลักษณะเป็นโครงสร้างของการเก็บรวบรวมข้อมูล การที่จะเพิ่ม เข้าถึงหรือประมวลผลข้อมูลที่เก็บในฐานข้อมูลจำเป็นจะต้องอาศัยระบบจัดการฐานข้อมูล ซึ่งจะทำหน้าที่เป็นตัวกลางในการจัดการกับข้อมูลในฐานข้อมูลทั้งสำหรับการใช้งานเฉพาะ และรองรับการทำงานของแอปพลิเคชันอื่นๆ ที่ต้องการใช้งานข้อมูลในฐานข้อมูล เพื่อให้ได้รับความสะดวกในการจัดการกับข้อมูลจำนวนมาก MySQL ทำหน้าที่เป็นทั้งตัวฐานข้อมูลและระบบจัดการฐานข้อมูล
- MySQL เป็นระบบจัดการฐานข้อมูลแบบ relational ซึ่งฐานข้อมูลแบบ relational จะทำการเก็บข้อมูลทั้งหมดในรูปแบบของตารางแทนการเก็บข้อมูลทั้งหมดลงในไฟล์ เพียงไฟล์เดียว ทำให้ทำงานได้รวดเร็วและมีความยืดหยุ่น นอกจากนั้น แต่ละตารางที่เก็บข้อมูลสามารถเชื่อมโยงเข้าหากันทำให้สามารถรวมหรือจัดกลุ่มข้อมูลได้ตามต้องการ โดยอาศัยภาษา SQL ที่เป็นส่วนหนึ่งของโปรแกรม MySQL ซึ่งเป็นภาษามาตรฐานในการเข้าถึงฐานข้อมูล
- MySQL แจกจ่ายให้ใช้งานแบบ Open Source คือ ผู้ใช้งาน MySQL ทุกคนสามารถใช้งานและปรับแต่งการทำงานได้ตามต้องการ สามารถดาวน์โหลดโปรแกรม MySQL ได้จากอินเทอร์เน็ต (Internet) และนำมาใช้งานโดยไม่มีค่าใช้จ่าย



## 2.7 XAMPP

การพัฒนาเว็บไซต์ หรือโปรแกรมที่ทำงานบนเว็บแอปพลิเคชัน จำเป็นต้องอาศัยเครื่องแม่ข่ายเว็บ (Web server) ซึ่งอาจจะเป็นภาระสำหรับผู้เรียน หรือผู้พัฒนาบางกลุ่ม แนวทางหนึ่งที่นิยมกันคือ การจำลองเครื่องพีซีให้เป็นเครื่องแม่ข่ายเว็บด้วยโปรแกรมสำเร็จรูป ซึ่งมีให้เลือกหลายค่ายซึ่ง XAMPP เป็นหนึ่งในผลิตภัณฑ์ที่มีการพัฒนาออกมาให้ใช้งาน

XAMPP พัฒนาโดยโครงการ Apache Friends ที่เป็นโครงการไม่แสวงหาผลกำไร ที่จัดตั้งในปี ค.ศ. 2002 โดย Kai ‘Oswald’ Seidler และ Kay Vogelgesang ทั้งนี้ XAMPP ประกอบด้วยโปรแกรมย่อยได้แก่ โปรแกรม Apache โปรแกรมฐานข้อมูล MySQL โปรแกรมภาษา PHP และภาษา Perl อีกทั้ง XAMPP มีการปรับปรุงอย่างต่อเนื่องเพื่อให้ทำงานได้กับระบบปฏิบัติการทั้ง Microsoft Windows, Mac OS x, Linux และ Solaris นอกจากนี้ยังไม่มีค่าใช้จ่ายในการดาวน์โหลดใช้งาน

### แนวคิดของ XAMPP

XAMPP โปรแกรมรวมการติดตั้งสำหรับนักพัฒนาโปรแกรมของ Apache เพื่อทำให้โปรแกรมนั้นใช้งานง่ายและสะดวกสำหรับนักพัฒนาโปรแกรม โปรแกรม XAMPP นั้นได้ตั้งค่าเริ่มต้นให้สามารถใช้งานโปรแกรม XAMPP แบบใช้งานโปรแกรมได้ในทุกอย่าง (all features turn on) ค่ามาตรฐานนี้ อาจจะเป็นจุดอ่อนในเรื่องความปลอดภัย หากนำไปใช้ในเครื่องที่ใช้งานจริง เพราะโปรแกรม XAMPP ถูกออกแบบมาสำหรับเครื่องของนักพัฒนาโปรแกรม

### ประโยชน์ของ XAMPP

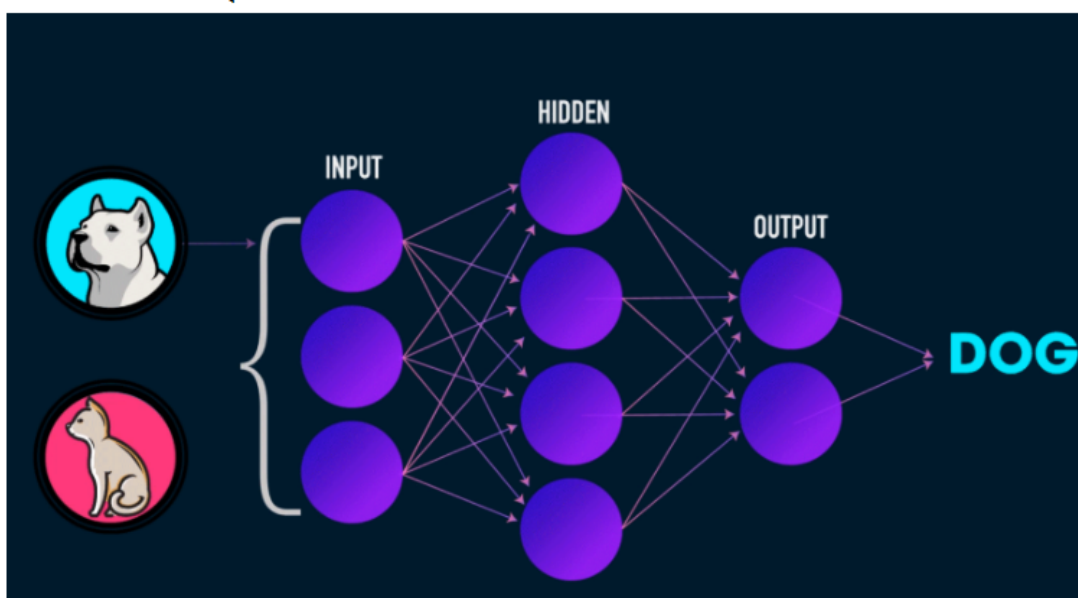
- สะดวกในการติดตั้งและถอนการติดตั้ง
- โปรแกรมมีคำสั่งต่างๆ ทำให้ง่ายต่อการใช้งาน
- โปรแกรมสามารถใช้งานได้หลายระบบปฏิบัติการ เช่น Linux, Windows, Mac OSX หรือ ระบบปฏิบัติการอื่นๆ
- เป็นโปรแกรมฟรีที่สนับสนุนในการใช้งานของผู้ที่ต้องการพัฒนาระบบ

## 2.8 ความรู้พื้นฐานเกี่ยวกับปัญญาประดิษฐ์ (Artificial Intelligence)

ปัญญาประดิษฐ์ (AI) เป็นศาสตร์แขนงหนึ่งของวิทยาศาสตร์ คอมพิวเตอร์ ที่เกี่ยวข้องกับวิธีการทำให้คอมพิวเตอร์มีความสามารถคล้ายมนุษย์ หรือเลียนแบบพฤติกรรมมนุษย์ โดยเฉพาะความสามารถในการคิดเองได้ หรือมีปัญญานั้นเอง ปัญหานี้มนุษย์เป็นผู้สร้างให้คอมพิวเตอร์ จึงเรียกว่าปัญญาประดิษฐ์

### 2.8.1 โครงข่ายประสาทเทียม (Neural Network)

โครงข่ายประสาทเทียม คือ โครงข่ายประสาทเทียมที่เป็นการจำลองมาจากสมองของมนุษย์ เพื่อจำลองการทำงานของเครือข่ายประสาทในสมองมนุษย์ ด้วยวัตถุประสงค์ที่จะสร้างเครื่องมือซึ่งมีความสามารถในการเรียนรู้การจดจำรูปแบบ (Pattern Recognition) และการสร้างความรู้ใหม่ (Knowledge Extraction) เช่นเดียวกับความสามารถที่มีในสมองมนุษย์นั่นเอง หลักการของ Neural Network ในคอมพิวเตอร์ neurons ประกอบด้วย input และ output เหมือนกัน โดยจำลองให้ input แต่ละอันมี weight เป็นตัวกำหนดน้ำหนักของ input โดย neuron แต่ละหน่วยจะมีค่า threshold เป็นตัวกำหนดว่าน้ำหนักรวมของ input ต้องมากขนาดไหนจึงจะสามารถส่ง output ไปยัง neurons ตัวอื่นได้ เมื่อนำ neuron แต่ละหน่วยมาต่อกันให้ทำงานร่วมกัน การทำงานนี้ในทางตรรกแล้ว จะเหมือนกับปฏิกิริยาเคมีที่เกิดในสมอง เพียงแต่ในคอมพิวเตอร์ทุกอย่างเป็นตัวเลขเท่านั้นเอง ดังรูปที่ 2.7



รูปที่ 2.7: ขั้นตอนของโครงข่ายประสาทเทียมแบบคอนโวลูชัน

ที่มา: <https://analyticsindiamag.com/most-common-activation-functions-in-neural-networks-and-rationale-behind-it/>

## 2.9 งานวิจัยที่เกี่ยวข้อง

นางสาวสุภาภรณ์ จินดาวงษ์ ได้นำเสนอวิจัยเกี่ยวกับ การศึกษาพฤติกรรมการเลือกใช้บริการร้านกาแฟสดของผู้บริโภค ผู้ศึกษาจึงสนใจเกี่ยวกับ พฤติกรรมการเลือกใช้บริการร้านกาแฟสดของผู้บริโภคในอำเภอเมือง จังหวัด นครปฐม กรณีศึกษา ร้านบ้านไร่กาแฟ สาขาที่29 เพื่อทราบถึงเหตุผลการเข้าใช้บริการร้านกาแฟสดบ้านไร่กาแฟ สาขาที่29 ผลการวิจัยทำให้ สามารถใช้เป็นแนวทางในการปรับปรุงธุรกิจให้ตรงกับความต้องการผู้บริโภคและเป็นแนวทาง ดำเนินกิจการแก่ผู้สนใจธุรกิจร้านกาแฟ

นางสาวกานดา เสือจำศีล ได้นำเสนอวิจัยเกี่ยวกับ พฤติกรรมการเข้าใช้บริการร้านกาแฟสด อเมซอน ของผู้บริโภค เพื่อศึกษาพฤติกรรมการเข้าใช้บริการร้านกาแฟสด อเมซอน ปัจจัยส่วนบุคคลของผู้บริโภคที่เข้าใช้บริการร้านกาแฟสด อเมซอน และปัจจัยส่วนผสมทางการตลาดต่อการเข้าใช้บริการร้านกาแฟสด อเมซอน กลุ่มตัวอย่างที่ใช้ คือ ผู้บริโภคในจังหวัดปทุมธานีที่เข้าใช้บริการร้านกาแฟสด อเมซอน จำนวน 400 คน

## บทที่ 3

### การวิเคราะห์และออกแบบระบบ

การวิเคราะห์และออกแบบระบบก่อนดำเนินการจริงเป็นอีกหนึ่งขั้นตอนที่มีความสำคัญมาก เพราะการวิเคราะห์และออกแบบระบบนั้นเป็นการกระทำที่ทำให้ผู้พัฒนาเห็นรายละเอียดส่วนย่อยของงานทั้งหมด เพิ่มประสิทธิภาพในการวางแผน การทำงาน และยังช่วยลดปัญหาที่อาจเกิดขึ้นในระหว่างพัฒนา เพื่อให้ระบบมีความสมบูรณ์มากยิ่งขึ้น เนื่องจากการวิเคราะห์และออกแบบระบบนั้นจะช่วยให้ให้บริการ จัดการทรัพยากรได้อย่างคุ้มค่าและตรงตามความต้องการของระบบ

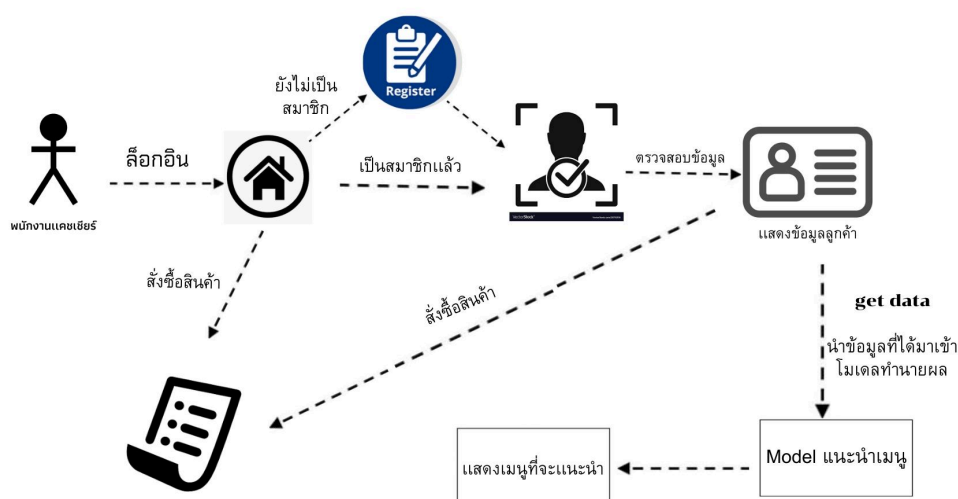
การวิเคราะห์และออกแบบระบบสแกนใบหน้าสำหรับแคชเชียร์ ในบทนี้จะแบ่งออกเป็น 6 ขั้นตอนเพื่อให้เห็นการดำเนินงานอย่างมีระบบ ในหัวข้อแรกจะนำเสนอภาพรวมของระบบ ก่อนจะนำเสนอเอกสารแสดงความต้องการของระบบซึ่งจะทำให้เห็นที่มาของเพจต่าง ๆ ในขั้นตอนของการออกแบบในหัวข้อที่สาม ส่วนหัวข้อที่เหลือจะแสดงแผนภาพการทำงานของระบบโดยใช้ 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 หมายถึง กรอบโครงสร้างของระบบที่อธิบายความสัมพันธ์ขององค์ประกอบต่าง ๆ ไปจนถึงขั้นการเชื่อมต่อกันของระบบย่อยต่าง ๆ โดยจัดกลุ่มองค์ประกอบไว้ในหลาย ๆ ลักษณะเพื่อให้ผู้เกี่ยวข้อง (Stakeholder) จากพื้นฐานสาขาอาชีพที่แตกต่าง กันสามารถทำความเข้าใจได้ง่าย เช่น การจัดแบ่งองค์ประกอบตามลักษณะการทำงานของระบบ (functional components) เป็นต้น

การออกแบบ System architecture แสดงภาพรวมและเทคโนโลยีของระบบสแกนใบหน้า สำหรับแคชเชียร์ มีรายละเอียดดังรูปที่ 3.1



รูปที่ 3.1: ภาพรวมและโครงสร้างการทำงานของระบบ

จากรูปที่ 3.1 เมื่อพนักงานแคชเชียร์ทำการล็อกอินแล้วจะเข้าหน้าหลัก ยังไม่เป็นสมาชิก จะทำการสมัครก่อน ถ้าเป็นสมาชิกแล้วจะเข้าสู่ระบบสแกนใบหน้าสำหรับลูกค้า โดยระบบจะทำการตรวจสอบรูปภาพที่สแกนว่าตรงกับรูปภาพของในระบบ แล้วแสดงข้อมูลของลูกค้าให้กับพนักงานแคชเชียร์แล้วยังหน้าส่งสินค้า หรือต้องการใช้ระบบแนะนำเมนูแก่ลูกค้าระบบจะทำการดึงข้อมูลไปให้โมเดลทำนาย แล้วจะแสดงเมนูที่จะแนะนำให้ลูกค้าแล้วทำการส่งสินค้า

## 3.2 System Requirements

### 3.2.1 Functional Requirements

ระบบสแกนใบหน้าสำหรับแคชเชียร์ แบ่งความสามารถของระบบตามประเภทของผู้ใช้งาน ดังนี้

#### 1. พนักงานแคชเชียร์

- สามารถทำการเข้าสู่ระบบได้
- สแกนใบหน้าของลูกค้าได้
- สมัครงานให้ลูกค้าได้
- แก้ไขข้อมูลลูกค้าเมื่อลูกค้าต้องการได้
- สามารถแนะนำเมนูแก่ลูกค้าได้
- สามารถสั่งสินค้าให้ลูกค้าได้

#### 2. ลูกค้า

- สแกนใบหน้าเพื่อทำการแสดงข้อมูล
- สั่งสินค้ากับพนักงาน

### 3.2.2 Non-functional Requirements

#### 1. เว็บแอปพลิเคชัน

- ใช้โปรโตคอล (Protocol) แบบ HTTPS (Hypertext Transfer Protocol Secure) ในการสื่อสารที่ช่วยรักษาความสมบูรณ์ถูกต้องของข้อมูลผู้ใช้และเก็บข้อมูลไว้เป็นความลับระหว่างคอมพิวเตอร์ของผู้ใช้กับเว็บไซต์
- ใช้ดีลิบ (dlib) ซึ่งมี Machine learning algorithms ในการจดจำหน้าของลูกค้าเพื่อระบุตัวตน
- ใช้ต้นไม้ตัดสินใจ (Decision Trees) ในการทำโมเดล (Model) ในการทำระบบแนะนำเมนู

## 3.3 User Interface Design

ในการออกแบบ User Interface Design ของระบบสแกนใบหน้าสำหรับแคชเชียร์

- การออกแบบหน้าจอเข้าสู่ระบบ

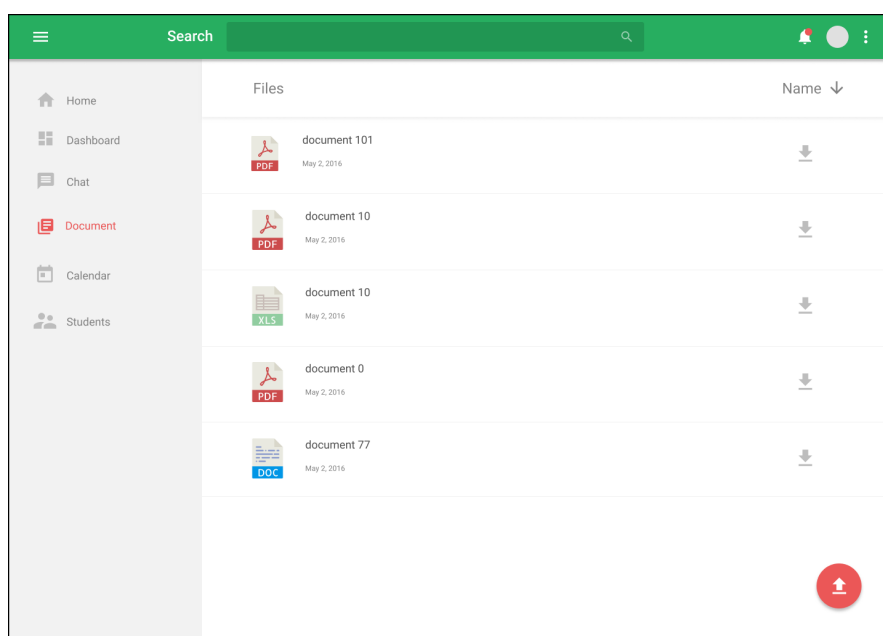
รูปที่ 3.2: หน้าเข้าสู่ระบบสำหรับพนักงาน

จากภาพที่ 3.2 แสดงหน้าจอการเข้าสู่ระบบของพนักงานแคชเชียร์ จำเป็นต้องกรอกข้อมูลรหัสพนักงานและรหัสผ่านเพื่อเข้าใช้งานระบบ

- การออกแบบหน้าจอหลัก

รูปที่ 3.3: หน้าจอหลัก

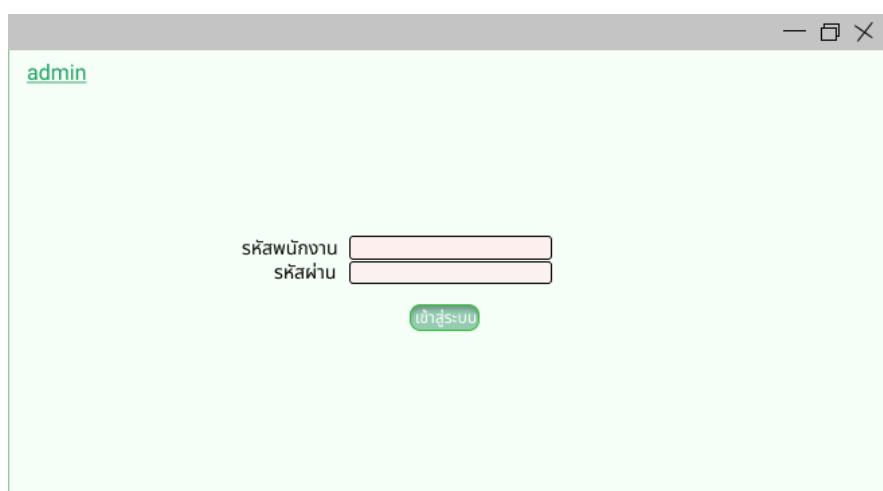
จากภาพที่ 3.3แสดงหน้าจอหลัก ให้พนักงานเลือก ใ้พนักงานเลือก ไ้ยังหน้า สมัครสมาชิก เข้าสู่ระบบโดยการสแกนใบหน้า ดาเข้าสู่ระบบโดยล็อกอินผ่านเบอร์โทรศัพท์



รูปที่ 3.4: หน้าจอแสดงข้อมูลลูกค้า

จากภาพที่ 3.4 แสดงหน้าจอ ข้อมูลลูกค้า หลังจากที่ได้ผ่านการสแกนใบหน้า จะแสดงข้อมูลลูกค้าที่สมัครสมาชิก แสดงประวัติการสั่งซื้อสินค้า แต่มั้สะสม และมีการแนะนำเมนูแก่ลูกค้า ทั้งนี้จะแสดงเฉพาะลูกค้าที่เป็นสมาชิกเท่านั้น

- การออกแบบหน้าจอสั่งซื้อสินค้า



รูปที่ 3.5: หน้าจอสั่งซื้อสินค้า

จากภาพที่ 3.5 แสดงหน้าจอรายการเมนูที่มีในร้าน โดยจะมีอยู่ 3 ประเภทหลัก คือ



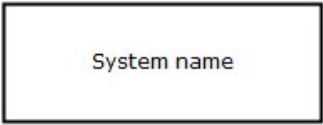
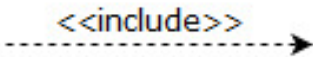
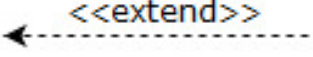


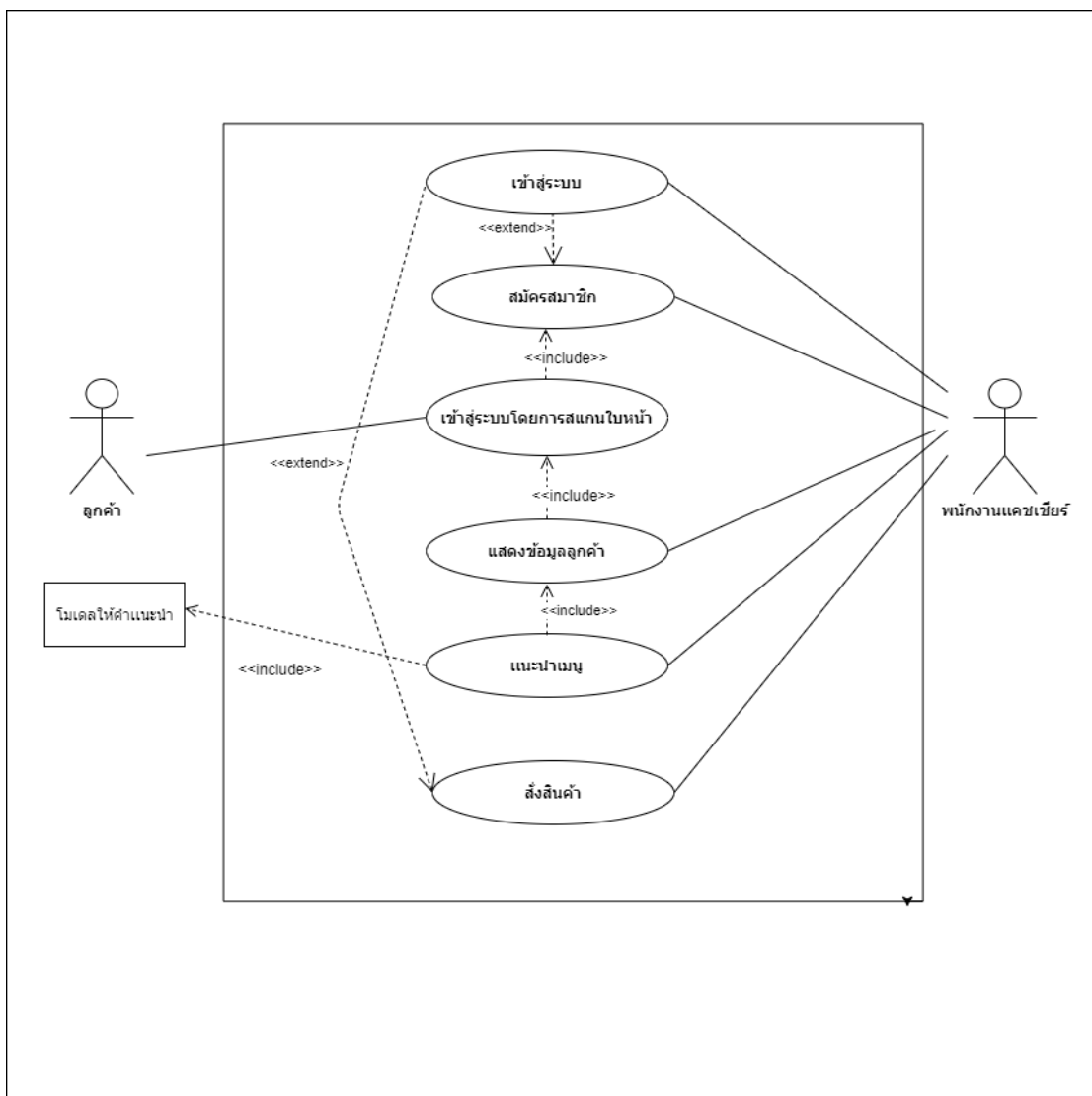
เมนูร้อน เมนูเย็น เมนูปั่น เมื่อลูกค้าสั่งจะมีการแสดงรายการที่สั่ง ราคา จำนวนที่  
ลูกค้าสั่ง

### 3.4 Use Case Diagram

Use Case Diagram เป็นแผนผังเพื่อแสดงฟังก์ชันการทำงานของระบบโดยรวม แสดงส่วนประกอบในระบบและกิจกรรมที่เกิดขึ้นในระบบ สัญลักษณ์ที่ใช้ในการเขียน Use Case Diagram แสดงในตารางที่ 3.1

ตารางที่ 3.1: สัญลักษณ์ของ Use case Diagram

สัญลักษณ์	การใช้งาน
Use case	Use case คือส่วนย่อยของระบบงาน แทนด้วยวงรีและชื่อของ Use case ภายในในวงรี
	Actor คือบุคคลหรือระบบงานอื่นที่ใช้งานระบบหรือได้รับประโยชน์จากระบบซึ่งอยู่นอกระบบ 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 3.6: Use Case diagram ของระบบ สแกนใบหน้าสำหรับแคชเชียร์

ตารางที่ 3.2: อธิบาย Use Case หน้าที่ของระบบ ในภาพที่ 3.6

Use Case	คำอธิบาย
สมัครสมาชิก	พนักงานทำการสมัครสมาชิก แก่ลูกค้า ที่ต้องการเป็นสมาชิก โดยผ่านการล็อกอินของพนักงานก่อน
สแกนใบหน้าลูกค้า	พนักงานสแกนใบหน้าลูกค้าที่เป็นสมาชิกในระบบ โดยผ่านการล็อกอินก่อน
แสดงข้อมูลลูกค้า	แสดงข้อมูลลูกค้าที่เป็นสมาชิกในระบบ โดยต้องผ่านการสแกนใบหน้าก่อนหรือพนักงานที่เป็นสมาชิก
แนะนำเมนู	แสดงเมนูที่จะแนะนำแก่ลูกค้าที่เป็นสมาชิก
สั่งซื้อสินค้า	พนักงานสั่งซื้อสินค้าตามที่ลูกค้าต้องการ โดยไม่ต้องสมัครเป็นสมาชิก ก็สามารถสั่งซื้อสินค้าได้

ตารางที่ 3.3: Use Caseสมัครสมาชิก

Use Case Title : สมัครสมาชิก	Use case Id : 1
Primary Actor : พนักงานแคชเชียร์	
Stakeholder Actor : ลูกค้า	
Main Flow : พนักงานแคชเชียร์สมัครสมาชิกให้ลูกค้าที่ต้องการเป็นสมาชิก โดยพนักงานแคชเชียร์ต้องผ่านการล็อกอิน	
Exceptional Flow ที่ 1 : หากผู้ใช้งานไม่ได้เชื่อมต่ออินเทอร์เน็ตจะไม่สามารถสมัครสมาชิกให้แก่ลูกค้าได้	
Exceptional Flow ที่ 2 : พนักงานแคชเชียร์จะไม่สามารถสมัครสมาชิกให้แก่ลูกค้าได้ถ้าไม่ทำการล็อกอินก่อน	

ตารางที่ 3.4: Use Case สแกนใบหน้าลูกค้า

Use Case Title : สแกนใบหน้าลูกค้า	Use case Id : 2
Primary Actor : พนักงานแคชเชียร์	
Stakeholder Actor : ลูกค้า	
Main Flow : พนักงานแคชเชียร์ทำการสแกนหน้าลูกค้าที่เป็นสมาชิกแล้ว โดยพนักงานจะต้องผ่านล็อกอินเข้าสู่ระบบก่อน	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถสแกนใบหน้าลูกค้าได้	
Exceptional Flow ที่ 2 : พนักงานแคชเชียร์จะไม่สามารถสแกนใบหน้าลูกค้าได้ ถ้าไม่ทำการล็อกอินเข้าสู่ระบบ	
Exceptional Flow ที่ 3 : พนักงานแคชเชียร์จะไม่สามารถสแกนใบหน้าลูกค้าได้ ถ้าไม่สมัครสมาชิกแก่ลูกค้า	

ตารางที่ 3.5: Use Case แสดงข้อมูลลูกค้า

Use Case Title : แสดงข้อมูลลูกค้า	Use case Id : 3
Primary Actor : พนักงานแคชเชียร์	
Stakeholder Actor : ลูกค้า	
Main Flow : ระบบจะแสดงข้อมูลลูกค้าที่เป็นสมัครสมาชิก โดยผ่านการสแกนใบหน้า	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถสแกนใบหน้าลูกค้าได้	
Exceptional Flow ที่ 2 : ระบบจะไม่สามารถแสดงข้อมูลลูกค้าได้ถ้าไม่ผ่านการสแกนใบหน้า	

ตารางที่ 3.6: Use Case แนะนำเมนู

Use Case Title : แนะนำเมนู	Use case Id : 4
Primary Actor : พนักงานแคชเชียร์	
Stakeholder Actor : ลูกค้า	
Main Flow : จะแสดงเมนูที่จะแนะนำแก่ลูกค้า	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถแสดงเมนูที่จะแนะนำลูกค้าได้	
Exceptional Flow ที่ 2 : ถ้าลูกค้าไม่เป็นสมาชิกจะไม่สามารถแนะนำเมนูให้ลูกค้าได้	

ตารางที่ 3.7: Use Case สั่งซื้อสินค้า

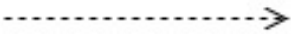
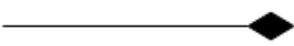
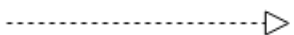

Use Case Title : สม	Use case Id : 5
Primary Actor : พนักงานแคชเชียร์	
Stakeholder Actor : ลูกค้า	
Main Flow เมื่อพนักงานแคชเชียร์เข้าสู่ระบบแล้ว จะทำการสั่งสินค้าให้ ลูกค้าจะเป็นสมาชิก หรือไม่สมาชิก	
Exceptional Flow ที่ 1 : หากผู้ใช้ไม่เชื่อมต่ออินเทอร์เน็ต จะไม่สามารถสั่งสินค้าให้ลูกค้าได้	
Exceptional Flow ที่ 2 : พนักงานแคชเชียร์จะไม่สามารถสั่งสินค้าให้ลูกค้าได้ถ้าไม่ล็อกอินเข้าสู่ระบบ	

### 3.5 Class Diagram

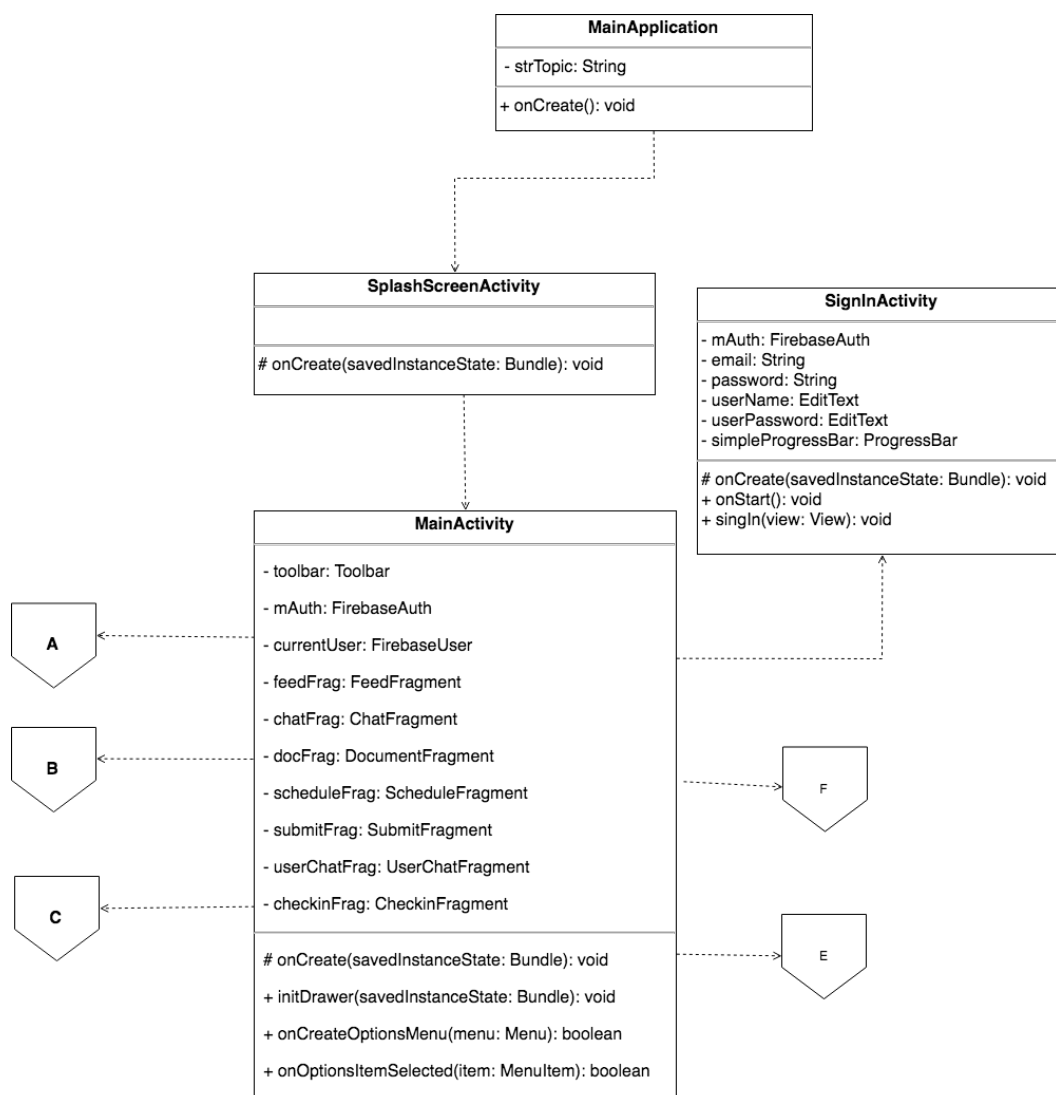
Class Diagram คือแผนภาพที่ใช้แสดงคลาสและความสัมพันธ์ในแบบต่างๆ ระหว่างคลาส

สัญลักษณ์ที่ใช้ในการเขียน Class Diagram แสดงในตารางที่ 3.8

ตารางที่ 3.8: สัญลักษณ์ของ Class Diagram

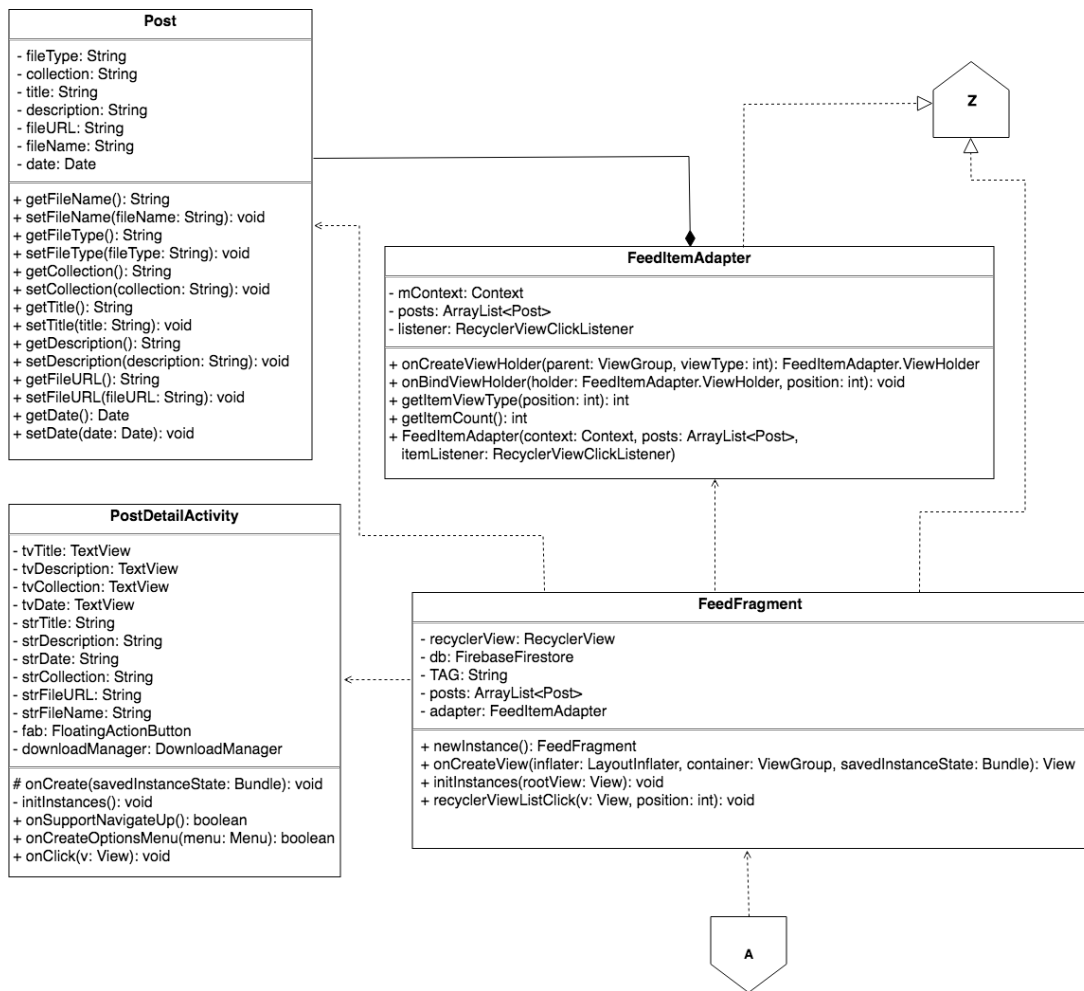
สัญลักษณ์	การใช้งาน
<div style="border: 1px solid black; padding: 5px; margin: 10px auto; width: fit-content;"> <div style="border-bottom: 1px solid black; padding-bottom: 5px; text-align: center;">Class Name</div> <div style="border-bottom: 1px solid black; padding: 5px 0 5px 20px; text-align: center;">Attribute 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> <p>(a) Public แทนสัญลักษณ์ด้วยเครื่องหมายบวก (+)</p> <p>(b) Private แทนสัญลักษณ์ด้วยเครื่องหมายลบ (-)</p> <p>(c) Protected แทนสัญลักษณ์ด้วยเครื่องหมายชาร์ป ( )</p>
	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 ระหว่างคลาสของระบบสแกนใบหน้า  
สำหรับแคชเชียร์ อธิบายได้ตามภาพที่ 3.7 ดังต่อไปนี้

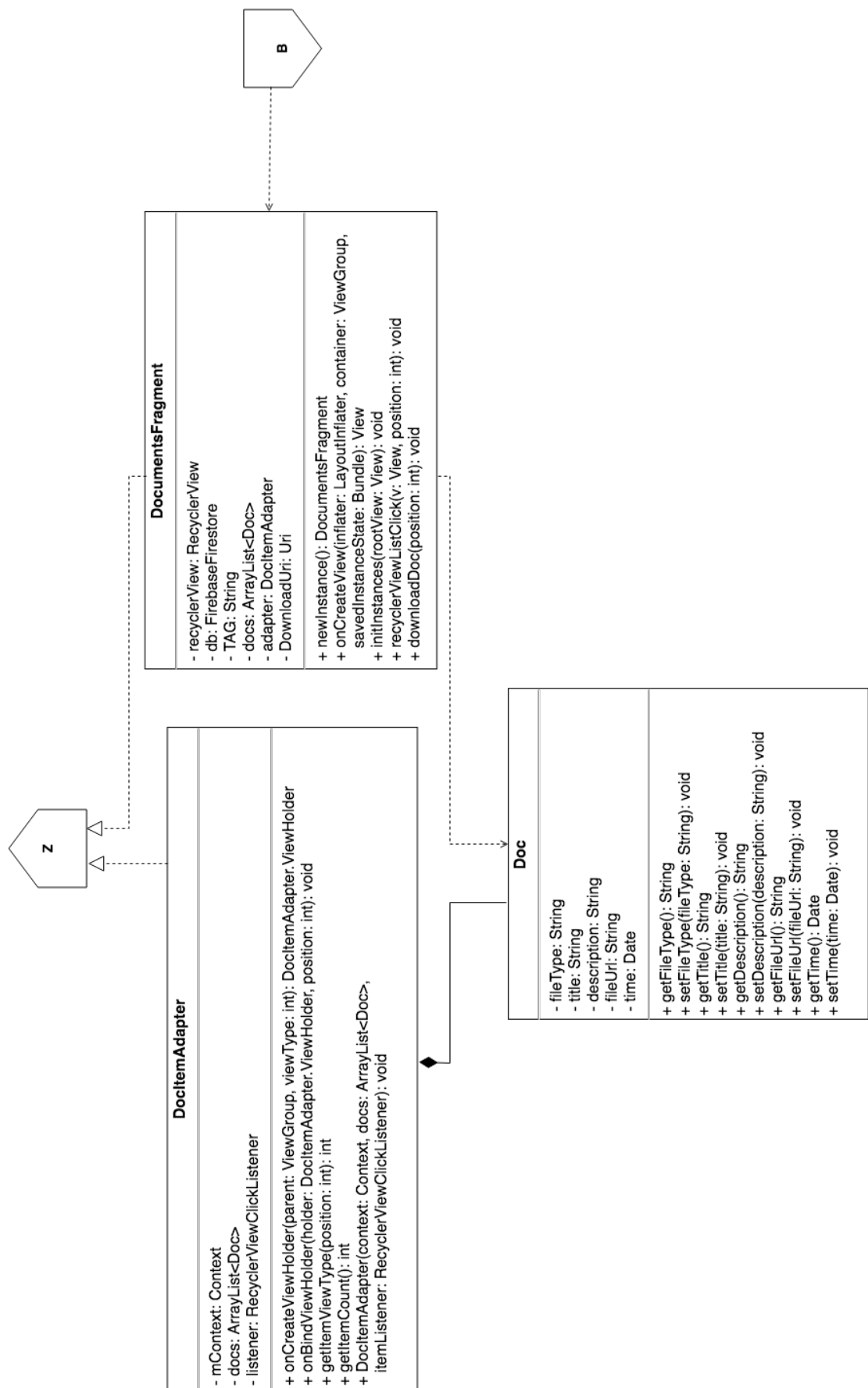


รูปที่ 3.7: Class Diagram ของแอปพลิเคชันระบบสแกนใบหน้าสำหรับแคชเชียร์

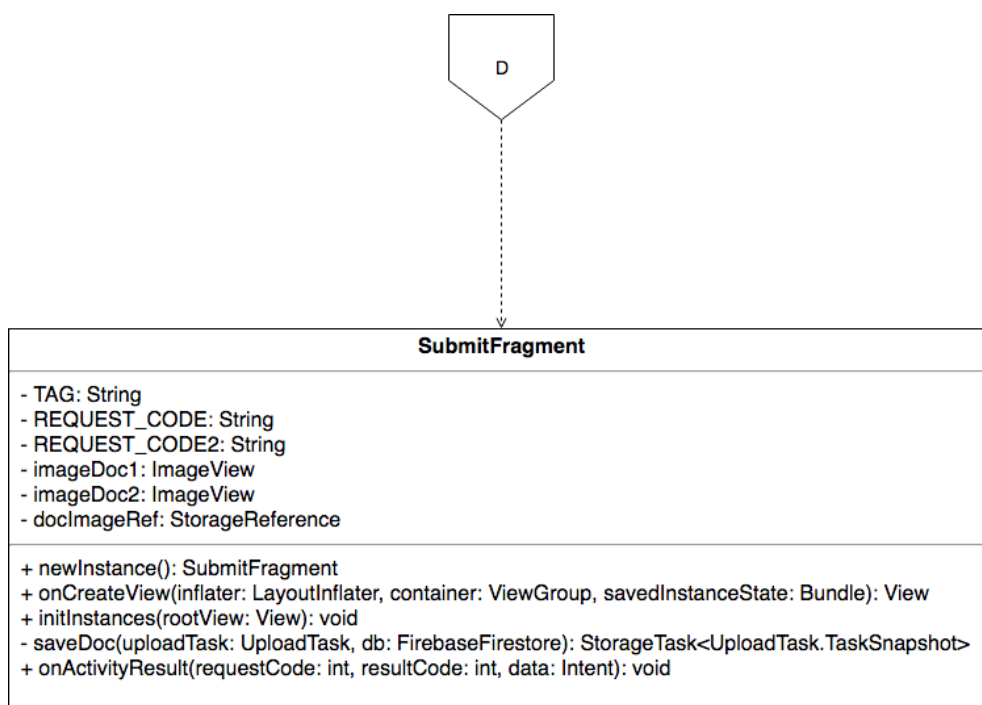




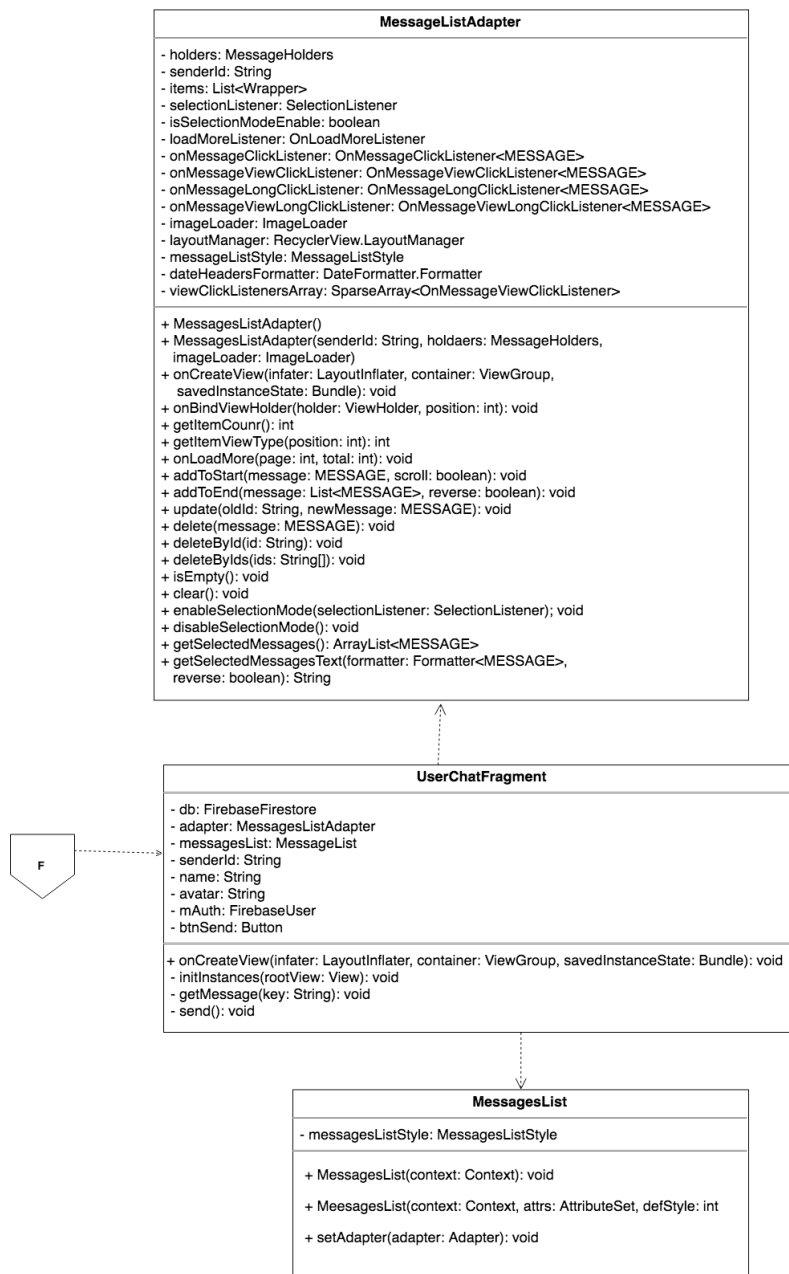
รูปที่ 3.8: Class Diagram ของแอปพลิเคชันระบบ XX



รูปที่ 3.9: Class Diagram ของแอปพลิเคชันระบบ XX



รูปที่ 3.10: Class Diagram ของแอปพลิเคชันระบบ XX



รูปที่ 3.11: Class Diagram ของแอปพลิเคชันระบบ XX

จากรูปภาพที่ 3.7 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.9: 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 3.8 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.10: อธิบาย Class Diagram ของส่วนของการแสดงข่าวสาร

Class Diagram	คำอธิบาย
FeedFragment	คลาส FeedFragment เป็นคลาสหลักที่ใช้ในการแสดงข้อมูลข่าวสาร มีการทำงานหลักคือสืบค้นฐานข้อมูลจากไฟร์เบสเพื่อนำมาแสดง
FeedItemAdapter	คลาส FeedItemAdapter เป็นคลาสที่มีหน้าที่ในการแปลงชุดข้อมูลที่ได้จากคลาส FeedFragment แล้วคืนค่ากลับเป็นลิสต์รายการของชุดข้อมูลนั้น ๆ
Post	คลาส Post เป็นคลาสโมเดลที่กำหนดค่าต่างๆที่จำเป็นสำหรับใช้ในการสร้างลิสต์รายการของคลาส FeedItemAdapter
PostDetailActivity	คลาส PostDetailActivity เป็น คลาส ที่ มีหน้า ที่ 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 ต้องการใช้งานสำหรับการรับค่าเมื่อผู้ใช้งานกดแถวในลิสต์รายการ คลาสลูกที่ทำการสืบทอดคุณสมบัติจะสามารถรับข้อมูลตำแหน่งแถวที่ผู้ใช้งานกดบนลิสต์รายการได้

จากรูปภาพที่ 3.9 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.11: 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 3.10 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.12: อธิบาย Class Diagram ของส่วนของการส่งสำเนาเอกสาร

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

จากรูปภาพที่ 3.11 สามารถอธิบายแผนภาพ Class Diagram ได้ดังนี้

ตารางที่ 3.13: อธิบาย Class Diagram ของส่วนของการสนทนา

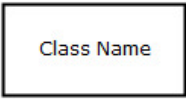


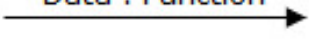
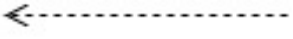


Class Diagram	คำอธิบาย
UserChatFragment	คลาส UserChatFragment เป็นคลาสที่ใช้ในการแสดงหน้าจอสนทนาสำหรับนักศึกษา เพื่อติดต่อสอบถามข้อมูลกับเจ้าหน้าที่ มีการสืบค้นข้อมูลประวัติการสนทนาเพื่อส่งไปแปลงเป็นข้อมูลลิสต์รายการที่คลาส MessagesListAdapter
MessagesListAdapter	คลาส MessagesListAdapter เป็นคลาสที่ใช้ในการแปลงชุดข้อมูลที่ได้รับจากคลาส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

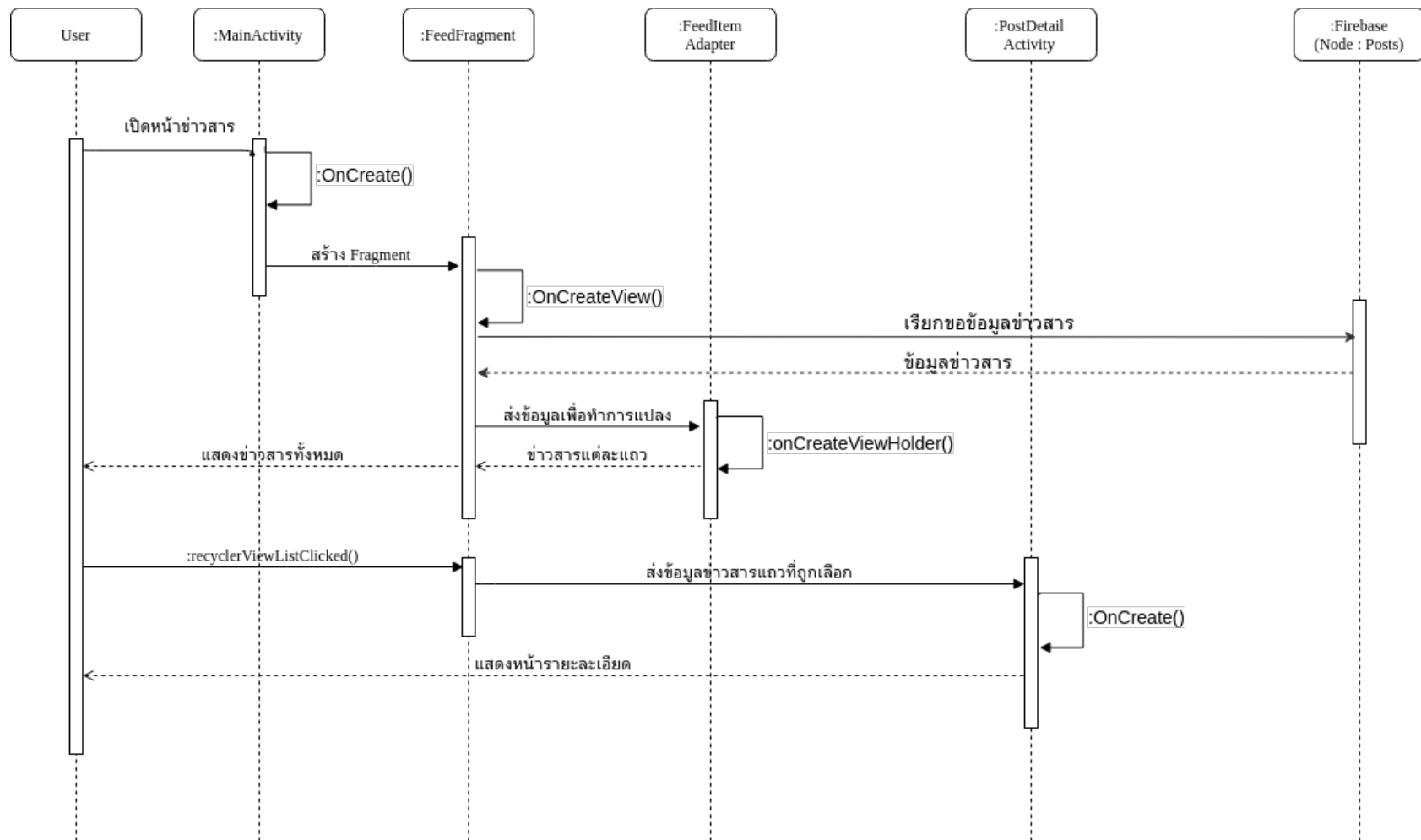


### 3.6 Sequence Diagram

Sequence Diagram เป็น Diagram ที่แสดงขั้นตอนการทำงานของแต่ละ Use Case ระหว่าง Object ต่างๆ ที่ส่งข้อความถึงกันและกัน โดย Sequence Diagram จะช่วยให้มองเห็นการทำงานของภาพรวมของระบบ ส่วนประกอบสัญลักษณ์ที่ใช้ในการเขียน Sequence Diagram แสดงดังตารางที่ 3.14

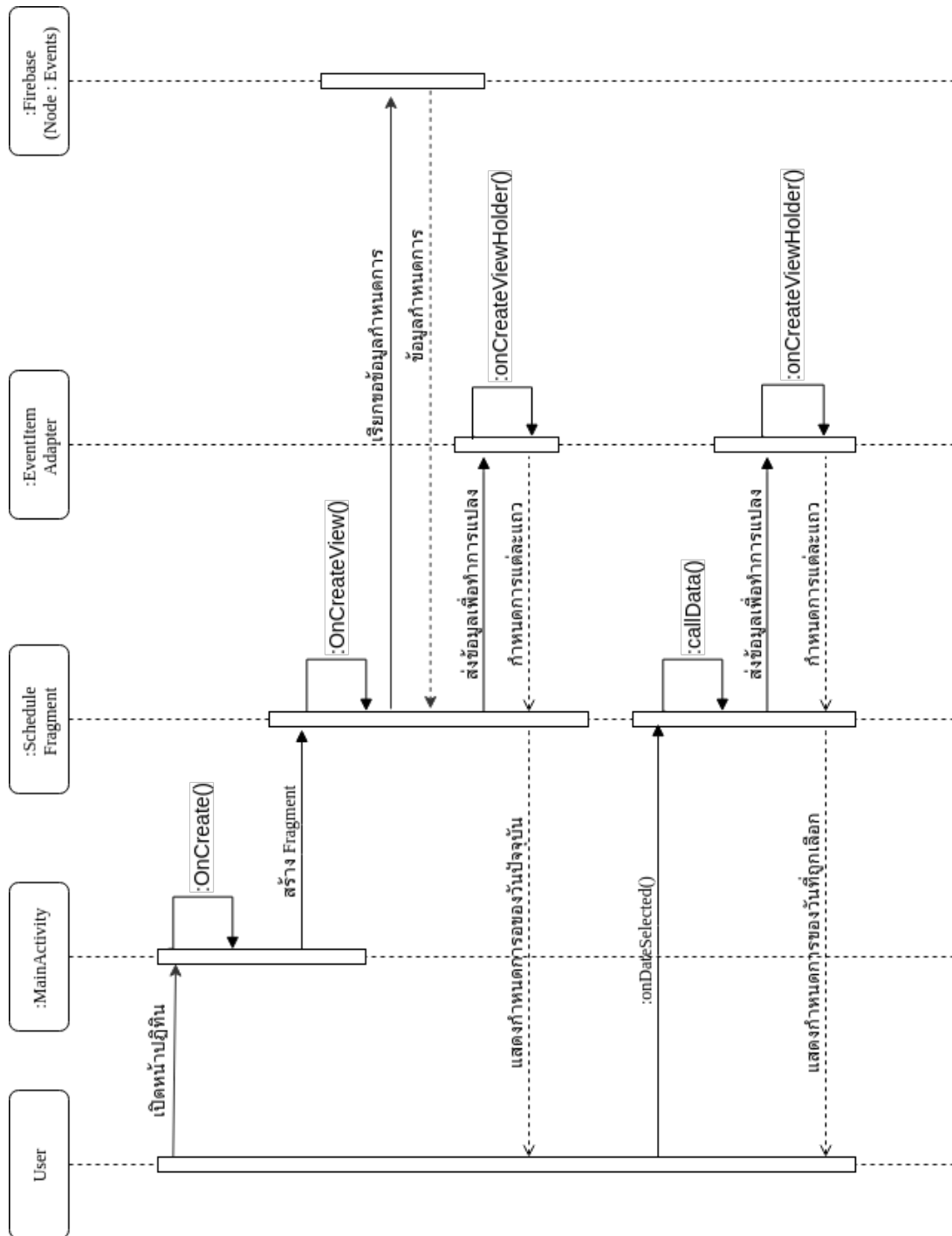
ตารางที่ 3.14: สัญลักษณ์ของ Sequence Diagram

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)



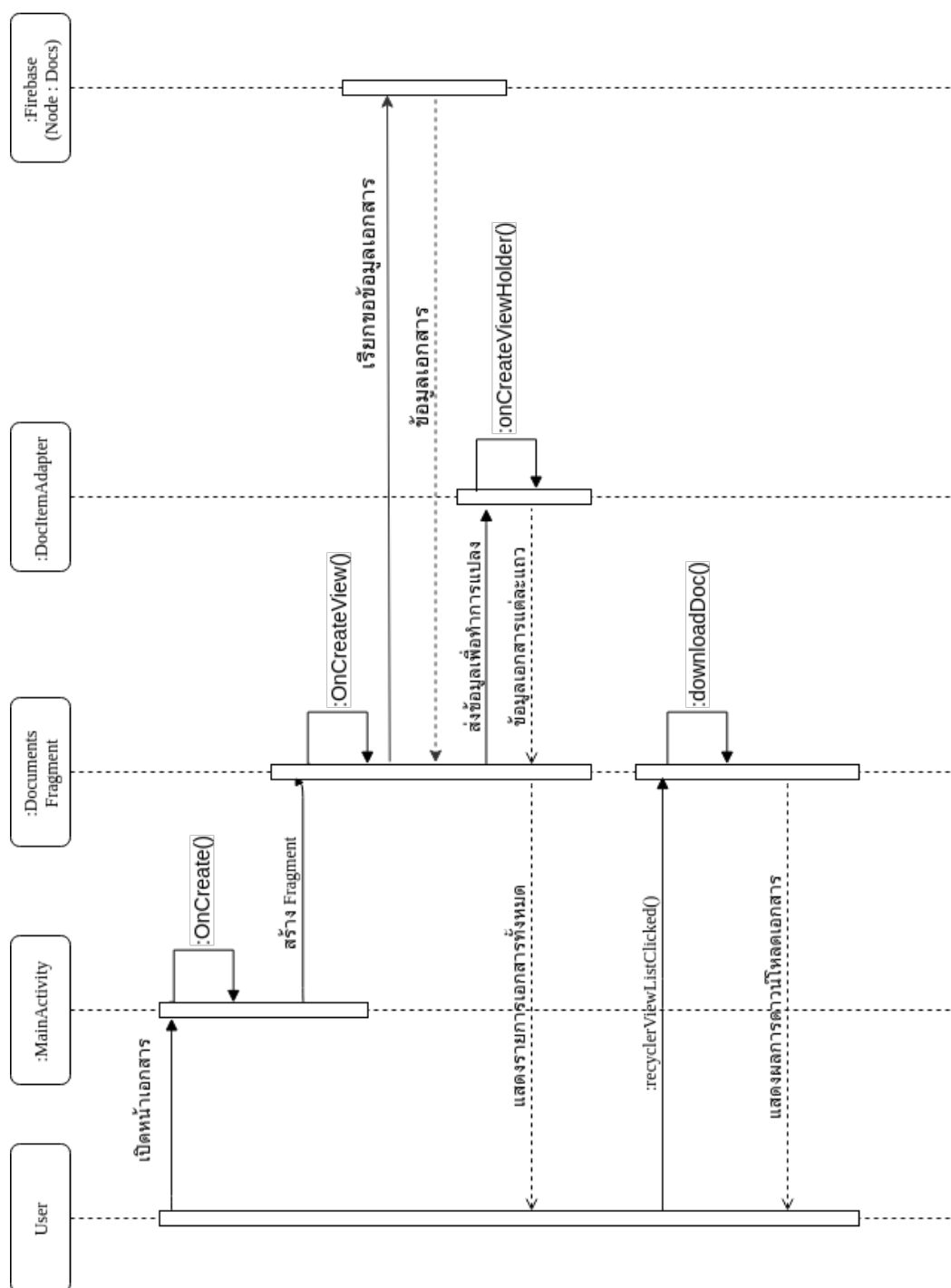
รูปที่ 3.12: Sequence Diagram การแสดงข่าวสาร

จากภาพที่ 3.12 สามารถอธิบายแผนภาพ Sequence Diagram แสดงข่าวสาร ได้ดังนี้ เมื่อ ผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบ จะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส FeedFragment เมื่อ FeedFragment ถูกติดตั้งบน MainActivity เมธอด callData() จะสืบค้นข้อมูลจากฐานข้อมูลบน Firebase Firestore และส่งข้อมูลที่ได้ไปแปลงที่คลาส FeedItemAdapter โดยมีการคืนค่าเป็นข้อมูลข่าวสารแต่ละแถวและในขั้นตอนสุดท้ายคลาส FeedFragment จะทำการแสดงรายการข้อมูลข่าวสารทั้งหมดออกทางหน้าจอ หากผู้ใช้มีการกดเลือกข่าวสาร บางแถวคลาส FeedFragment จะทำการเรียกใช้ PostDetailActivity เพื่อแสดงรายละเอียดข้อมูลข่าวสารของแถวที่ถูกเลือก



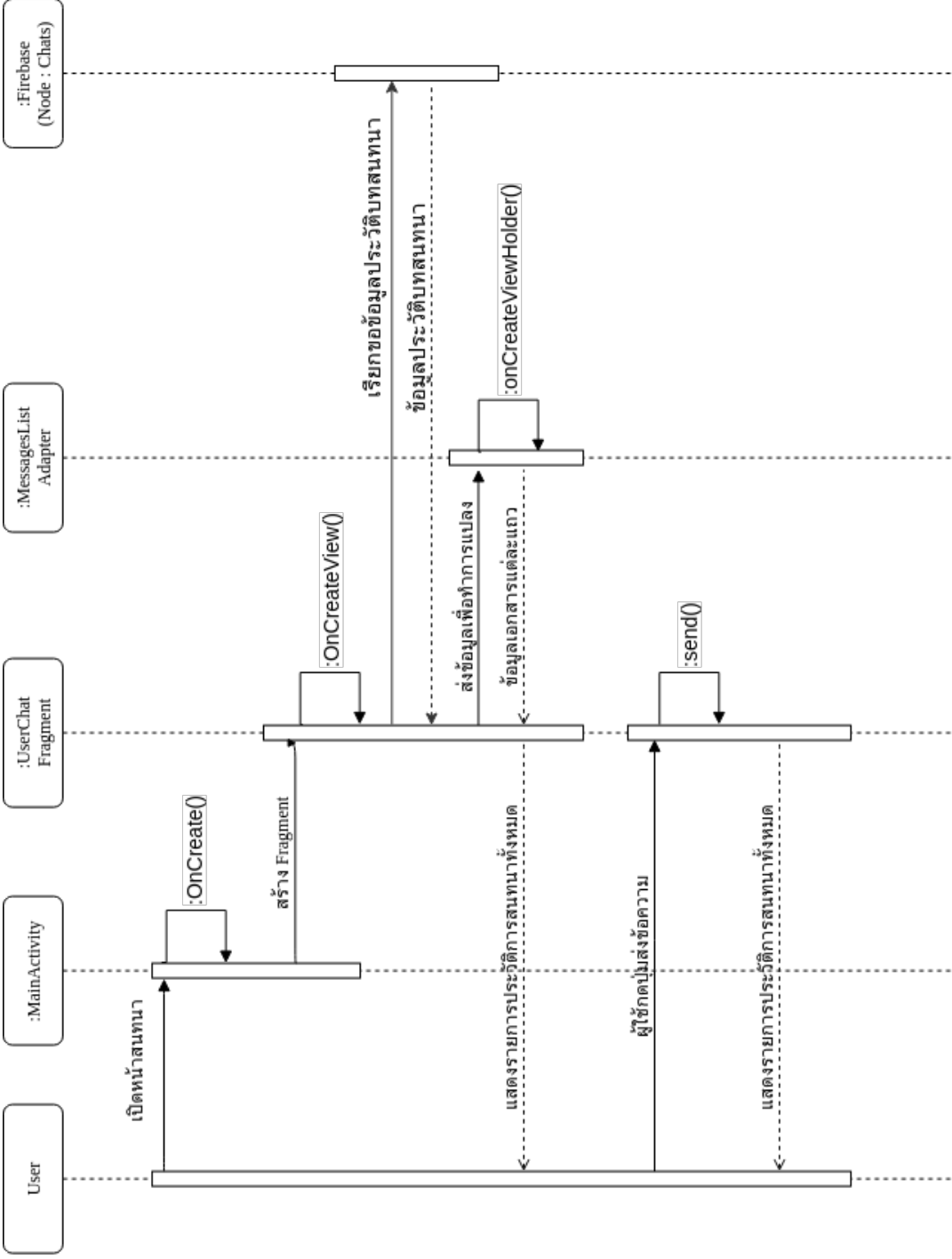
รูปที่ 3.13: Sequence Diagram การแสดงปฏิทินกำหนดการ

จากภาพที่ 3.13 สามารถอธิบายแผนภาพ Sequence Diagram แสดงปฏิทินกำหนดการได้ดังนี้ เมื่อ ผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส ScheduleFragment เมื่อ ScheduleFragment ถูกติดตั้งบน MainActivity เมธอด callData() จะสืบค้นข้อมูลกำหนดการของวันปัจจุบันจากฐานข้อมูลบน Firebase Firestore และส่งข้อมูลที่ได้ไปแปลงที่คลาส Schedule-ItemAdapter โดยมีการคืนค่าเป็นข้อมูลกำหนดการแต่ละแถว และในขั้นตอนสุดท้ายคลาส Schedule-Fragment จะทำการแสดงรายการกำหนดการวันปัจจุบันออกทางหน้าจอ หากผู้ใช้มีการกดเลือกวันที่ที่ต้องการทราบกำหนดการจากปฏิทิน คลาส ScheduleFragment จะทำการเรียกใช้ callData() อีกครั้งโดยสืบค้นข้อมูลกำหนดการของวันที่ถูกเลือกจากฐานข้อมูลบน Firebase Firestore และส่งข้อมูลที่ได้ไปแปลงที่คลาส ScheduleItemAdapter โดยมีการคืนค่าเป็นข้อมูลแต่กำหนดการละแถวและในขั้นตอนสุดท้ายคลาส ScheduleFragment จะทำการแสดงรายการกำหนดการวันที่ผู้ใช้เลือกออกทางหน้าจอ



รูปที่ 3.14: Sequence Diagram การแสดงดาวน์โหลดเอกสาร

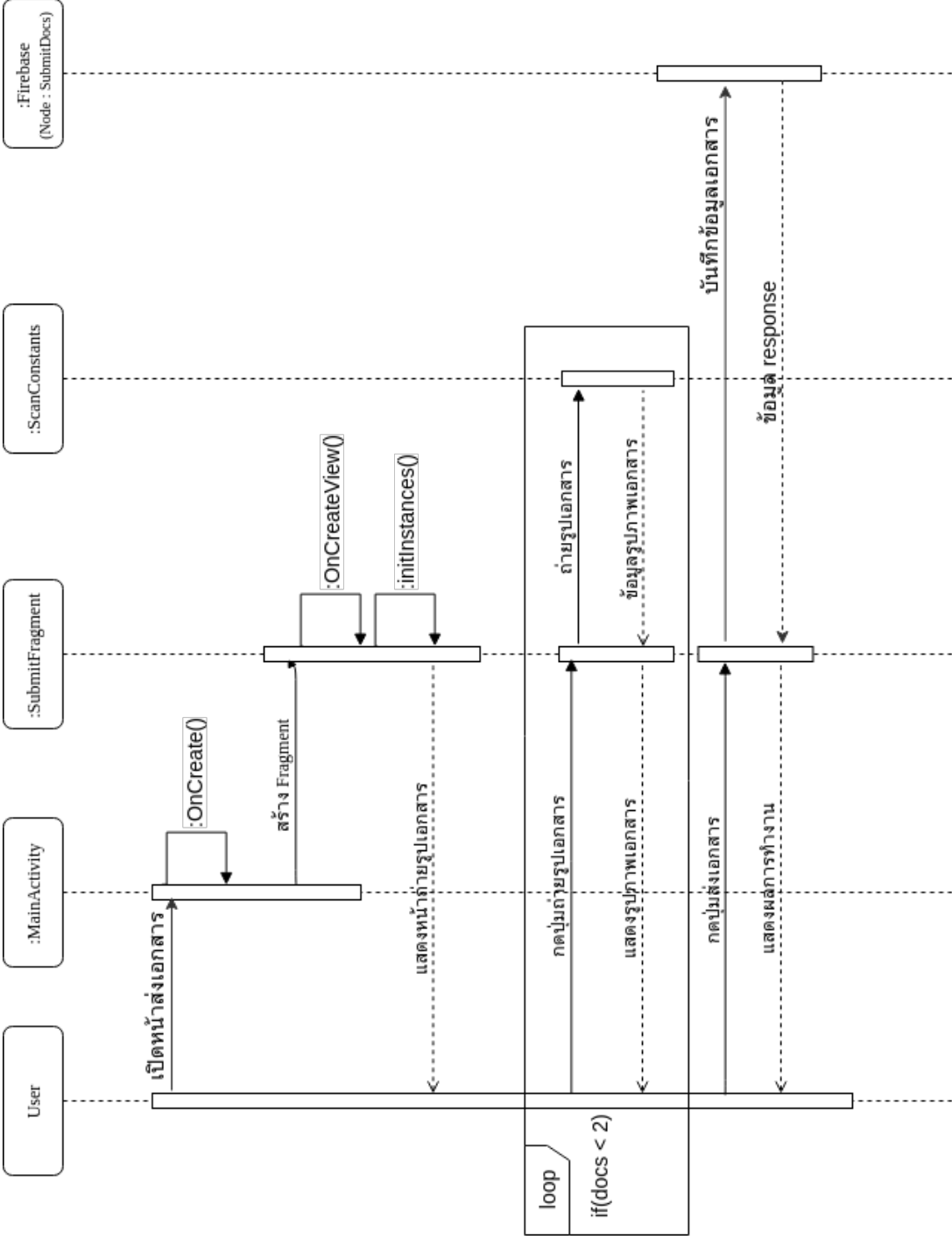
จากภาพที่ 3.14 สามารถอธิบายแผนภาพ Sequence Diagram แสดงทาวนโหนดเอกสารได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส DocumentsFragment เมื่อ DocumentsFragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะสืบค้นข้อมูลเอกสารทั้งหมดจากฐานข้อมูลบน Firebase Firestore และส่งข้อมูลที่ได้ไปแปลงที่คลาส DocItem-Adapter โดยมีการคืนค่าเป็นข้อมูลเอกสารแต่ละแถวและในขั้นตอนสุดท้ายคลาส Documents-Fragment จะทำการแสดงรายการกำหนดการวันปัจจุบันออกทางหน้าจอ



รูปที่ 3.15: Sequence Diagram การแสดงบทสนทนา



จากภาพที่ 3.15 สามารถอธิบายแผนภาพ Sequence Diagram แสดงการสนทนา ได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส UserChatFragment เมื่อ UserChatFragment ถูกติดตั้งบน MainActivity เมธอด getMessage() จะสืบค้นข้อมูลประวัติการสนทนาของผู้ใช้คนปัจจุบันทั้งหมดจากฐานข้อมูลบน Firebase Firestore และส่งข้อมูลที่ได้ออกไปแปลงที่คลาส MessagesListAdapter โดยมีการคืนค่าเป็นข้อมูลรายการประวัติการสนทนาทั้งหมดและในขั้นตอนสุดท้ายคลาส User-ChatFragment จะทำการแสดงรายการประวัติการสนทนาทั้งหมดออกทางหน้าจอ เมื่อผู้ใช้พิมพ์ข้อความและกดปุ่มส่งระบบจะเรียกให้เมธอด send() เพื่อทำการบันทึกข้อมูลไว้บน Firebase Firestore และทำการแสดงข้อมูลรายการประวัติการสนทนาทั้งหมดที่ถูกอัปเดต



รูปที่ 3.16: Sequence Diagram แสดงส่งเอกสารตรวจสอบ

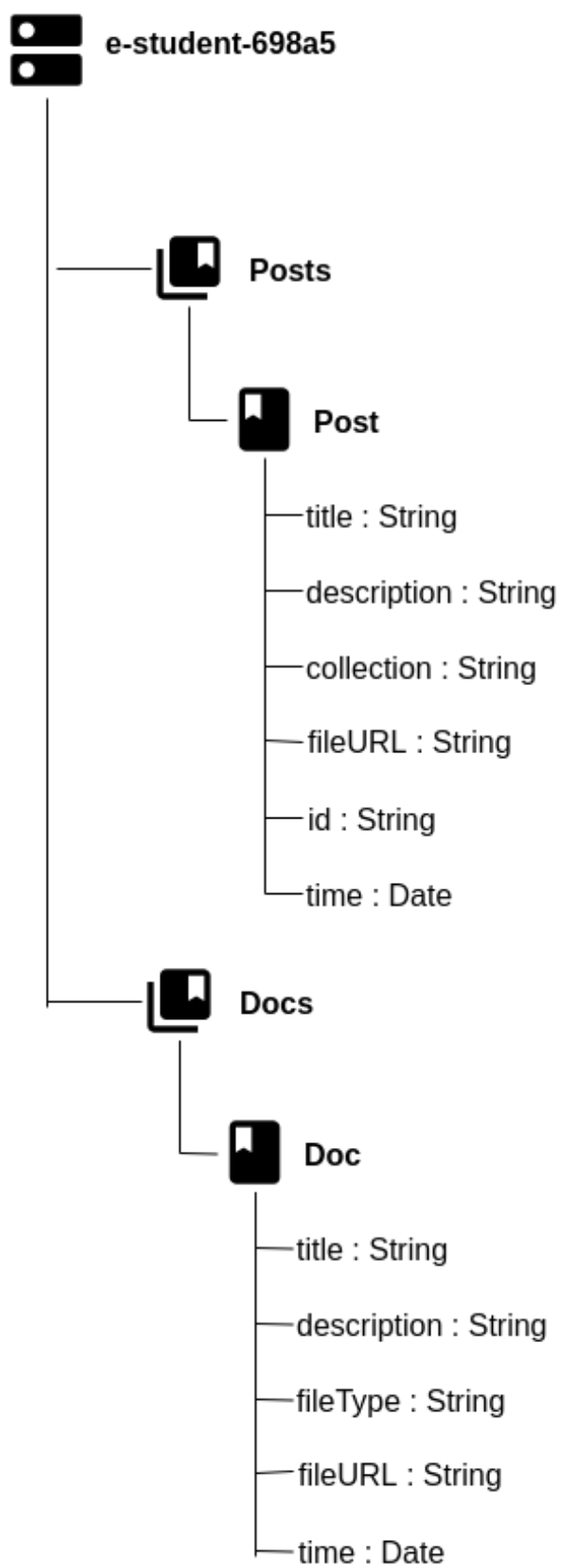
จากภาพที่ 3.16 สามารถอธิบายแผนภาพ Sequence Diagram แสดงส่งเอกสารตรวจสอบได้ดังนี้ เมื่อผู้ใช้เปิดโปรแกรมระบบจะเรียกใช้เมธอด onCreate() ที่คลาส MainActivity ระบบจะทำการสร้าง Fragment ขึ้นมาโดยใช้เมธอด onCreate() ที่คลาส SubmitFragment เมื่อ Submit-Fragment ถูกติดตั้งบน MainActivity เมธอด initInstances() จะถูกเรียกเพื่อสร้างหน้าจอแสดงผลเมื่อผู้ใช้กดปุ่มถ่ายรูประบบจะเรียกใช้ไลบรารี ScanConstants เพื่อถ่ายภาพเอกสารและรอให้ผู้ใช้ถ่ายครบทั้งสองแผ่นจึงจะแสดงปุ่มกดส่งเอกสารเพื่อตรวจสอบ

### 3.7 โครงสร้างฐานข้อมูลไฟร์เบส(Firebase Database Stucture)

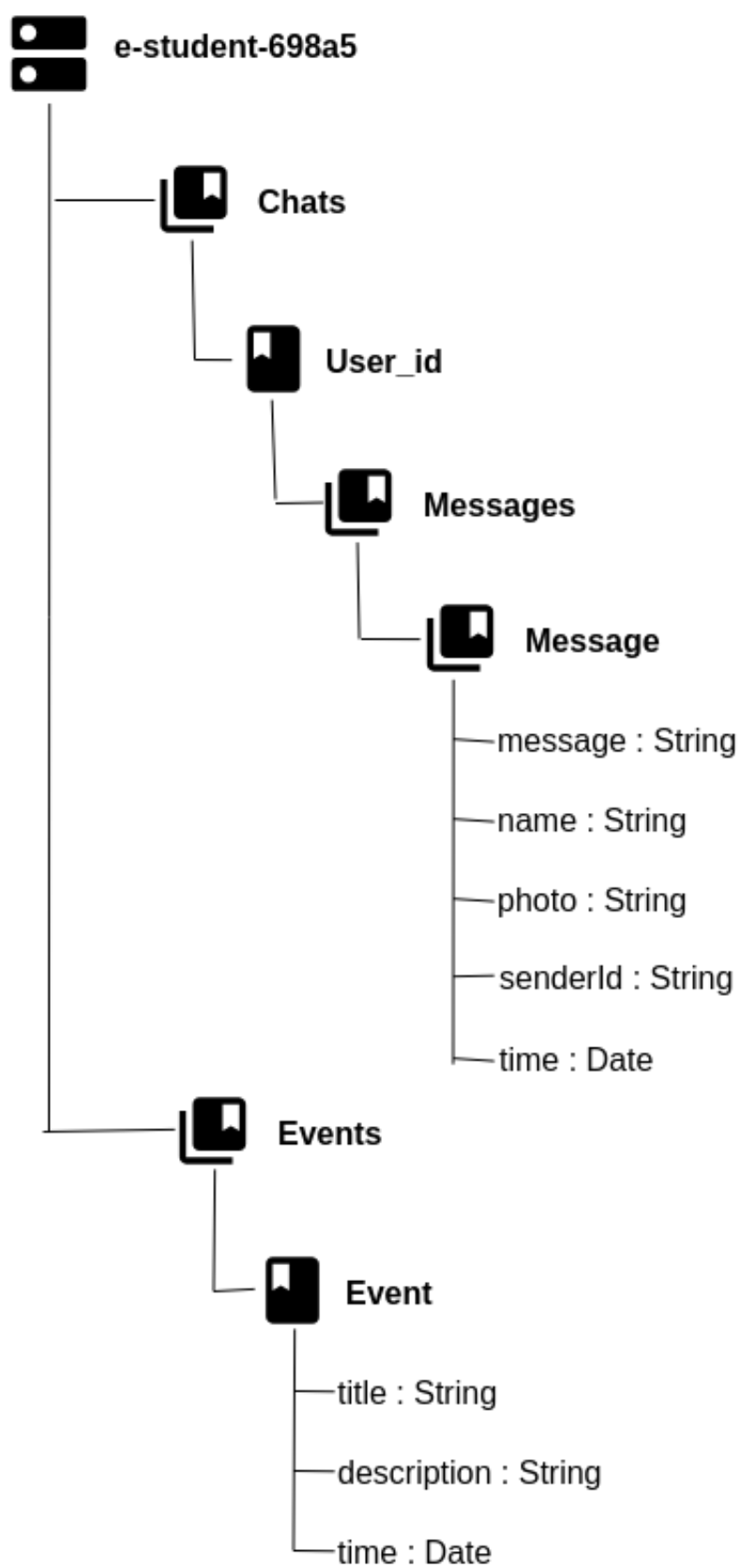
Firebase Database นั้นเป็น Database แบบ NoSQL และเป็น JSON database ที่มีโครงสร้างที่เป็น Key และ Value จัดเก็บข้อมูลในลักษณะโหนด หากต้องการเรียกงานจะเรียกใช้โดย การท่องไปยังโหนดที่ต้องการ ส่วนประกอบสัญลักษณ์ที่ใช้ในการเขียนโครงสร้างฐานข้อมูลแบบ Firebase แสดงดังตารางที่ 3.15

ตารางที่ 3.15: สัญลักษณ์ของโครงสร้างฐานข้อมูลแบบ Firebase

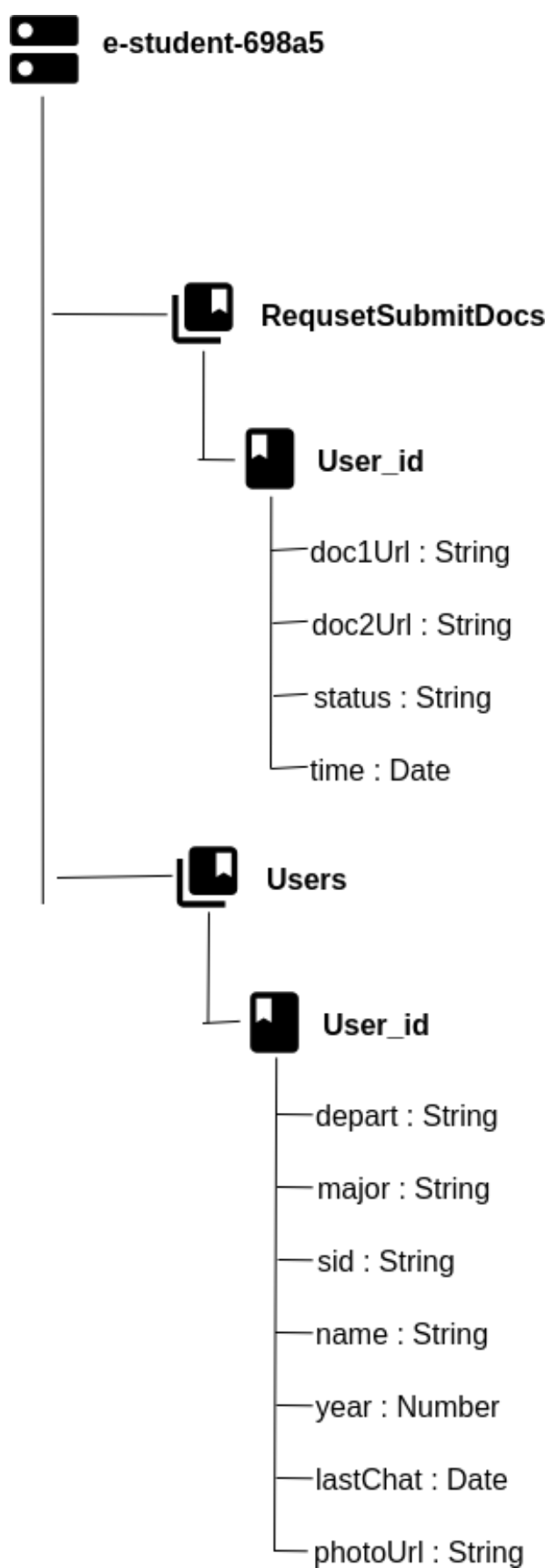
สัญลักษณ์	คำอธิบาย
	Database เป็นการเรียกชื่อแทนโหนด(Node)บนสุดที่ใช้ในการเก็บข้อมูล
	Collection เป็นการเรียกชื่อแทนของการเก็บหลาย ๆ เอกสารไว้ด้วยกัน
	Document เป็นการ เรียก ชื่อ แทน หน่วย การ เก็บ ของ ข้อมูล ใน Cloud Firestore ภายในจะประกอบไปด้วย ชื่อของ Document ชื่อของคีย์ (key) และ ค่าข้อมูล (value)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



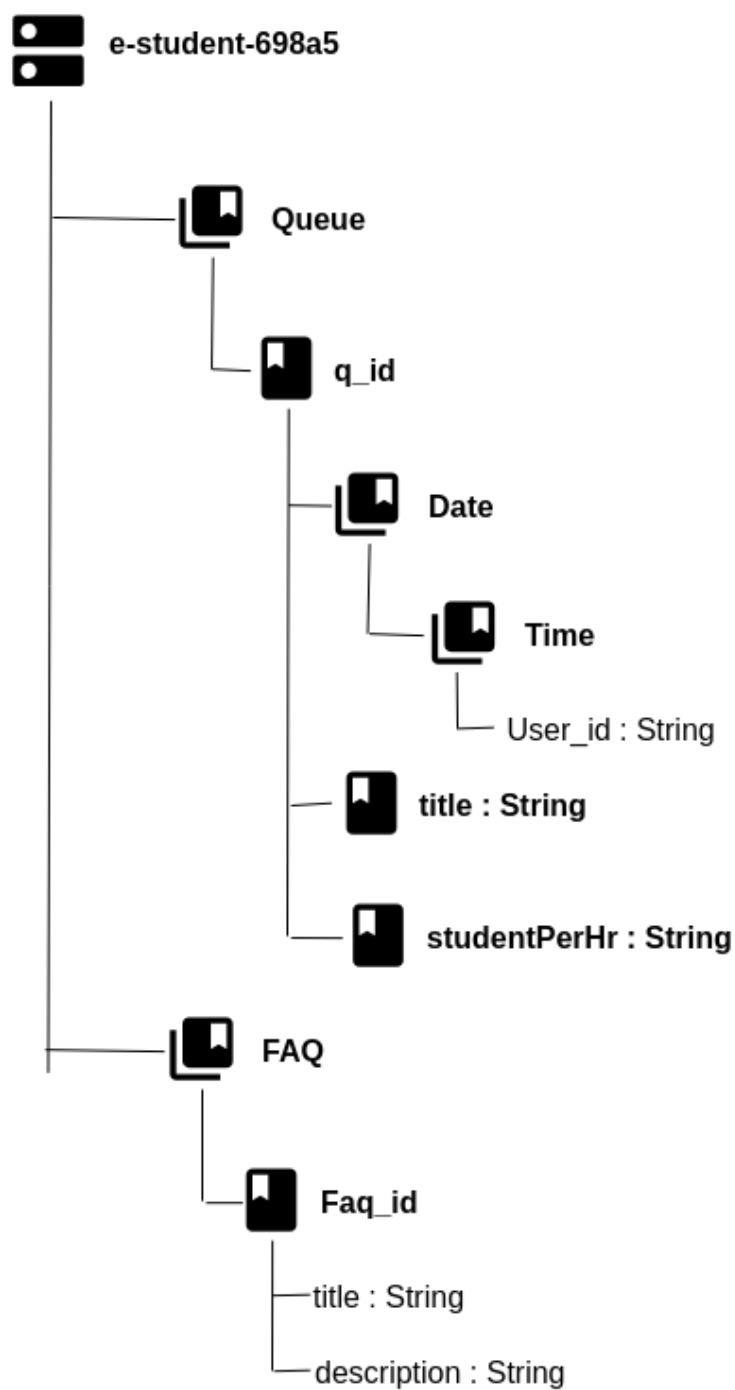
รูปที่ 3.17: โครงสร้างฐานข้อมูลแบบ Firebase



รูปที่ 3.18: โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ)



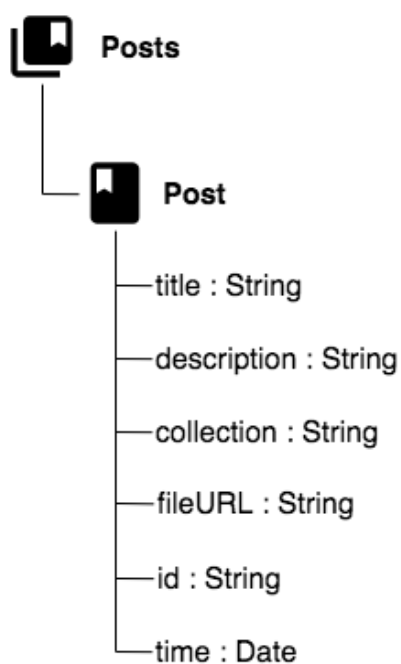
รูปที่ 3.19: โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ)



รูปที่ 3.20: โครงสร้างฐานข้อมูลแบบ Firebase(ต่อ)



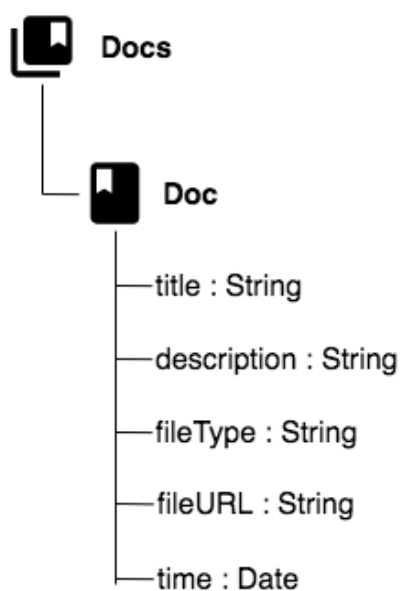
จากรูที่ 3.17-3.28 สามารถอธิบายโครงสร้างของข้อมูลได้ดังนี้



รูปที่ 3.21: โหนดเก็บข้อมูลประกาศ

ตารางที่ 3.16: อธิบายโหนดที่ใช้เก็บข้อมูลประกาศ

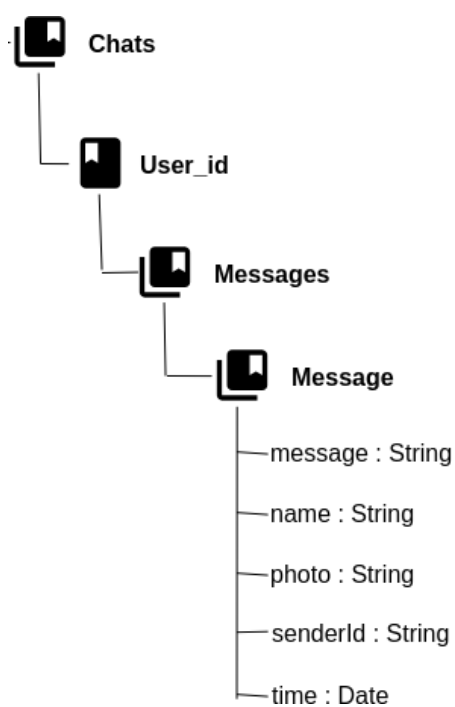
Key	คำอธิบาย
Posts	โหนดสำหรับเก็บข้อมูลประกาศทั้งหมด
Post	สำหรับเก็บข้อมูลแต่ละประกาศ
title	สำหรับเก็บชื่อหัวข้อประกาศ
description	สำหรับเก็บรายละเอียดประกาศ
collection	สำหรับเก็บประเภทของประกาศได้แก่ สาธารณะและเฉพาะบุคคล
fileURL	สำหรับเก็บ url ของเอกสารแนบประกาศ
id	สำหรับเก็บรหัสของประกาศ
time	สำหรับเก็บเวลาที่ประกาศ



รูปที่ 3.22: โหนดเก็บข้อมูลเอกสารที่เกี่ยวข้อง

ตารางที่ 3.17: อธิบายโหนดที่ใช้เก็บข้อมูลเอกสารที่เกี่ยวข้อง

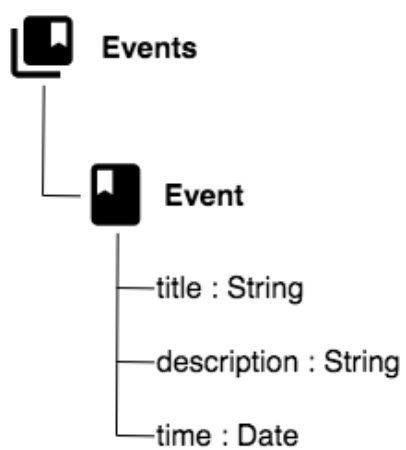
Key	คำอธิบาย
Docs	โหนดสำหรับเก็บข้อมูลของเอกสารที่เกี่ยวข้องทั้งหมด
Doc	สำหรับเก็บข้อมูลเอกสารแต่ละฉบับ
title	สำหรับเก็บชื่อหัวเรื่องของเอกสาร
description	สำหรับเก็บรายละเอียดของเอกสาร
fileType	สำหรับนามสกุลไฟล์เอกสาร เช่น .pdf .png เป็นต้น
fileURL	สำหรับเก็บ url ของเอกสาร
time	สำหรับเก็บเวลาที่ถูกอัปโหลดเข้าสู่ระบบโดยเจ้าหน้าที่



รูปที่ 3.23: โหนดเก็บข้อมูลประวัติการสนทนา

ตารางที่ 3.18: อธิบายโหนดที่ใช้เก็บข้อมูลประวัติการสนทนา

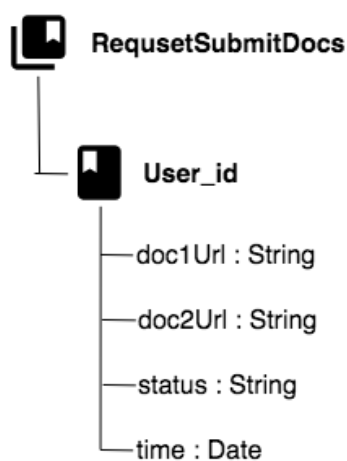
Key	คำอธิบาย
Chats	โหนดสำหรับเก็บข้อมูลประวัติการสนทนาทั้งหมด
User_id	สำหรับเก็บประวัติการสนทนาของผู้ใช้แต่ละคน
Messages	สำหรับเก็บประวัติการสนทนาทั้งหมดของผู้ใช้
Message	สำหรับเก็บข้อมูลของแต่ละข้อความ
message	สำหรับเก็บข้อความ
name	สำหรับเก็บชื่อของผู้ส่งข้อความ
photo	สำหรับเก็บ url รูปภาพของผู้ส่งข้อความ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



รูปที่ 3.24: โหนดเก็บข้อมูลกำหนดการ

ตารางที่ 3.19: อธิบายโหนดที่ใช้เก็บข้อมูลกำหนดการ

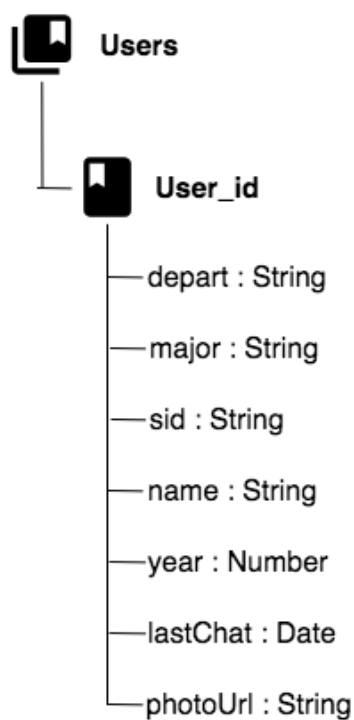
Key	คำอธิบาย
Events	โหนดสำหรับเก็บข้อมูลของกำหนดการทั้งหมด
Event	สำหรับเก็บข้อมูลของแต่ละกำหนดการ
title	สำหรับเก็บชื่อหัวข้อของกำหนดการ
description	สำหรับเก็บรายละเอียดของกำหนดการ
time	สำหรับเก็บเวลาของกำหนดการ



รูปที่ 3.25: โหนดเก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา

ตารางที่ 3.20: อธิบายโหนดที่ใช้เก็บข้อมูลการยื่นสำเนาเอกสารเพื่อตรวจสอบของนักศึกษา

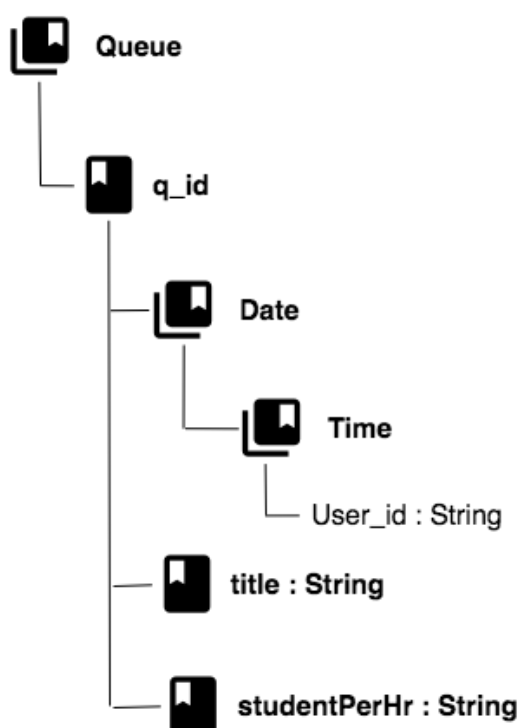
Key	คำอธิบาย
RusetSubmitDocs	โหนดสำหรับเก็บ ข้อมูล การยื่นสำเนาเอกสาร เพื่อ ตรวจสอบ ของนักศึกษาทั้งหมด
User_id	สำหรับเก็บข้อมูลของแต่ละสำเนาเอกสารของนักศึกษาแต่ละคน
doc2	สำหรับเก็บ url ของภาพถ่ายสำเนาเอกสารฉบับที่ 1
doc2	สำหรับเก็บ url ของภาพถ่ายสำเนาเอกสารฉบับที่ 2
status	สำหรับเก็บผลการตรวจสอบของเจ้าหน้าที่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



รูปที่ 3.26: โหนดเก็บข้อมูลของนักศึกษา

ตารางที่ 3.21: อธิบายโหนดที่ใช้เก็บข้อมูลของนักศึกษา

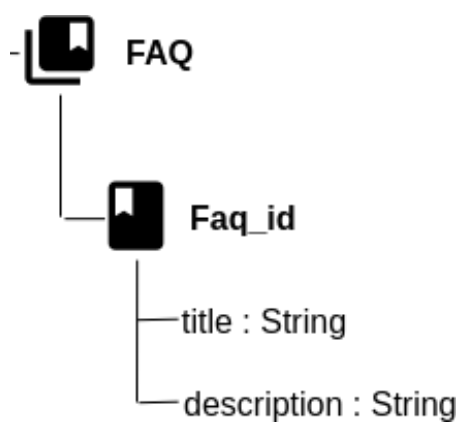
Key	คำอธิบาย
Users	โหนดสำหรับเก็บข้อมูลของนักศึกษา
User_id	สำหรับเก็บข้อมูลของนักศึกษาแต่ละคน
depart	สำหรับเก็บภาควิชาของนักศึกษา
major	สำหรับเก็บสาขาของนักศึกษา
sid	สำหรับเก็บรหัสประจำตัวนักศึกษา
name	สำหรับเก็บชื่อของนักศึกษา
year	สำหรับเก็บชั้นปีของนักศึกษา
lastChat	สำหรับเก็บเวลาที่สนทนากับเจ้าหน้าที่ล่าสุด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



รูปที่ 3.27: โหนดเก็บข้อมูลการจองคิวของนักศึกษา

ตารางที่ 3.22: อธิบายโหนดที่ใช้เก็บข้อมูลการจองคิวของนักศึกษา

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



รูปที่ 3.28: โหนดเก็บข้อมูลคำถามที่พบบ่อย

ตารางที่ 3.23: อธิบายโหนดที่ใช้เก็บข้อมูลคำถามที่พบบ่อย

Key	คำอธิบาย
Queue	โหนดสำหรับเก็บข้อมูลคำถามที่พบบ่อยทั้งหมด
Faq_id	สำหรับเก็บข้อมูลคำถามที่พบบ่อยแต่ละรายการ
title	สำหรับเก็บคำถาม
description	สำหรับเก็บคำตอบ



## บทที่ 4

## การพัฒนาารระบบ

หลังจากที่ได้มีการเตรียมความพร้อมสำหรับการพัฒนาในด้านต่าง ไม่ว่าจะเป็นที่มาและความสำคัญของปัญหา เทคโนโลยีที่มีความเหมาะสมกับระบบ และการออกแบบระบบการทำงานรวมไปถึงโครงสร้างของข้อมูล ในบทนี้จะเป็นการพูดถึงการสร้างระบบที่ได้มีการออกแบบไว้ในบทที่แล้วจะถูกนำเสนอในบทนี้ โดยการพัฒนาระบบแบ่งได้เป็นส่วนต่าง ๆ ดังนี้

#### 4.1 การพัฒนาเว็บแอปพลิเคชัน

## 4.2 การพัฒนาเว็บแอปพลิเคชันแอนดรอยด์แอปพลิเคชัน

## 4.1 การพัฒนาเว็บแอปพลิเคชัน

การพัฒนาระบบกองทุนเงินให้กู้ยืมเพื่อการศึกษาสำหรับเว็บแอปพลิเคชันนั้นวัตถุประสงค์  
 หลังเพื่อสร้างความสะดวกต่อการดำเนินงานของเจ้าหน้าที่อื่นเนื่องมาจากข้อจำกัดบางประการ  
 หากใช้ระบบทำงานบนอุปกรณ์สมาร์ตโฟนเพียงอย่างเดียว โดยตัวเว็บแอปพลิเคชันนี้ถูก  
 พัฒนาขึ้นด้วย Vue.js มีรายละเอียดการทำงานดังนี้

### 4.1.1 การเชื่อมต่อ Cloud Firestore

bsection, ในการเชื่อมต่อเว็บแอปพลิเคชันกับไพร์เบสเพื่อให้บริการต่างๆ ของไพร์เบส ทำได้ดังนี้

```
1 export default {
2   apiKey: "
      XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
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 3 เป็นการนำเข้าโมดูลตั้งค่าที่ได้จากรูปภาพที่ 4.2
- บรรทัดที่ 5 เป็นการส่งออกโมดูลไฟร์เบสเพื่อใช้ในไฟล์อื่น ๆ ซึ่งเมื่อถึงขั้นตอนนี้การเชื่อมต่อบริการไฟร์เบสถือว่าเป็นอันเสร็จ

#### 4.1.2 โครงสร้างของการสร้างหน้าเข้าสู่ระบบ

```

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

```

รูปที่ 4.3: การสร้างหน้าจอสวนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue

จากภาพที่ 4.4 โครงสร้างของการสร้างหน้าจอสวนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ 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: การสร้างหน้าจอสวนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ SignIn.vue

จากภาพที่ 4.4 โครงสร้างของการสร้างหน้าจอสวนติดต่อผู้ใช้ของหน้าเข้าสู่ระบบ สามารถ

อธิบายการทำงานได้ดังนี้

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

```

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 โครงสร้างของการสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าข่าวสาร 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 Cloude Firestore เพื่อทำการสืบค้นข้อมูลข่าวสาร

ทั้งหมดพร้อมทั้งเรียงลำดับตามวันที่ประกาศ

- 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 ดึงค่าโปรไฟล์ผู้ใช้งานปัจจุบัน
- บรรทัดที่ 4-6 เขียนข้อมูลลงฐานข้อมูล Cloud Firestore โดยระบุ path ที่จะทำการจัดเก็บชุดข้อมูล
- บรรทัดที่ 28-35 อัปเดตข้อมูลเวลาสนทนาล่าสุดของผู้ใช้
- 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()
              .time.getMonth()}-${doc.data().time.
              getFullYear()}`
14         }
15         vm.eventsData.push(data)
16       })
17     })
18 }

```

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  })
35 }

```

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>

```

รูปที่ 4.16: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าอัปโหลดเอกสารที่เกี่ยวข้อง  
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: การสร้างลอจิกของหน้าหน้าอัปโหลดเอกสารที่เกี่ยวข้อง 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 Cloud Firestore

#### 4.1.9 โครงสร้างของการสร้างหน้าสร้างกำหนดการจองคิวส่งเอกสาร

```

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: การสร้างหน้าจอส่วนติดต่อผู้ใช้ของหน้าสร้างกำหนดการจองคิวส่งเอกสาร 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 มีวัตถุประสงค์หลักเพื่ออำนวยความสะดวกแก่นักศึกษาที่โดยส่วนใหญ่ใช้อุปกรณ์พกพาและต้องการใช้งานฮาร์ดแวร์(Hardware)เช่น กล้องที่ใช้ในการถ่ายภาพสำเนาเอกสาร 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 ใช้แสดงผลหน้าจอลำข่าวนิตยสาร
- บรรทัดที่ 3 ตัวแปร chatFrag ใช้แสดงผลหน้าจอลำข่าวนิตยสารสำหรับเจ้าหน้าที่
- บรรทัดที่ 4 ตัวแปร chatFrag ใช้แสดงผลหน้าจอลำข่าวนิตยสารที่เกี่ยวข้อง
- บรรทัดที่ 5 ตัวแปร sheduleFrag ใช้แสดงผลหน้าจอลำข่าวนิตยสารกำหนดการ
- บรรทัดที่ 6 ตัวแปร submitFrag ใช้แสดงผลหน้าจอลำข่าวนิตยสารส่งสำเนาเอกสาร
- บรรทัดที่ 7 ตัวแปร userChatFrag ใช้แสดงผลหน้าจอลำข่าวนิตยสารสำหรับนักศึกษา
- บรรทัดที่ 8 ตัวแปร checkinFrag ใช้แสดงผลหน้าจอลำข่าวนิตยสารจองคิวส่งเอกสาร

```

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 เป็นการแสดงผลหน้าจอควิซเอกสาร
- 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 ใช้ในการแสดงข้อมูลลิสต์รายการข่าวสาร
- บรรทัดที่ 2 ตัวแปร db ใช้ในการสืบค้นข้อมูลจากข่าวสารจาก Cloud Firestore
- บรรทัดที่ 3 ตัวแปร posts ใช้ในการเก็บชุดข้อมูลที่ได้จากการสืบค้นข้อมูล
- บรรทัดที่ 4 ตัวแปร adapter ใช้ในการแปลงชุดข้อมูลเป็นลิสต์รายการเพื่อแสดงบน 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 รับผลการสืบค้นพร้อมทั้งเพิ่มข้อมูลที่ได้แต่ละแถวเข้าไปในตัวแปร 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 เริ่มการทำงานของแอคทिवิตี 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 EventListener<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         Map<String, Object> data = document.getData();
18         id = document.getId();
19         text = data.get("message").toString();
20         createdAt = (Date) data.get("time");
21         usrId = data.get("senderId").toString();
22         Message m = new Message(id, text, createdAt, new
            User(usrId, name, avatar));
23         adapter.addToStart(m, true);
24         adapter.notifyDataSetChanged();
25     }
26 }
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.getUid());
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 เป็นการอัปเดตเวลาสนทนาย่าสุดของผู้ใช้

#### 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 เป็นการเรียกใช้ไฟร์เบส Authentication เพื่อใช้งานระบบ

#### 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

การทดสอบการทำงานของแอนดรอยด์แอปพลิเคชันระบบกองทุนเงินให้กู้ยืมเพื่อการศึกษา คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานีและทดสอบการทำงานในส่วนของเว็บไซต์ โดยทำการทดสอบในลักษณะ Black-box Testing [2] หรือ Data-Driven testing ซึ่งเป็นการทดสอบที่ไม่สนใจโปรเซส (Process) การทำงานภายในของโปรแกรมว่าทำงานอย่างไร แต่จะเน้นไปที่ Input และ Result ที่ได้มากกว่าว่าการทำงานต่าง ๆ ถูกต้องตามความต้องการ (Requirement) หรือไม่ ซึ่งการทดสอบการใช้งานแอนดรอยด์แอปพลิเคชัน และการใช้งานเว็บแอปพลิเคชัน ได้ผลดังนี้

#### 5.1 การทดสอบการใช้งานแอนดรอยด์แอปพลิเคชัน

- การทดสอบการใช้งานเมนูนำทางของแอนดรอยด์แอปพลิเคชัน การทดสอบเมนูนำทางของแอปพลิเคชันในการนำทางผู้ใช้งาน ซึ่งเมนูหลักประกอบด้วย เมนูหน้าประกาศ เมนูหน้าสนทนา เมนูหน้าปฏิทินกำหนดการ เมนูหน้าดาวน์โหลดเอกสาร 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
เมนูหน้าจองคิวง์เอกสาร	กดปุ่มเมนูหน้าจองคิวง์เอกสาร	ระบบแสดงผลหน้าจองคิวง์เอกสาร
	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 คิวง์ เอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ-ภาค พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าเกี่ยวกับ	กดปุ่มเมนูหน้าเกี่ยวกับ	ระบบ แสดง ผล หน้า เกี่ยว กับ ซึ่งแสดงข้อมูลผู้พัฒนารวมไป ถึงแสดงเครดิต (credit) โล-บารรีต่าง ๆ ที่ใช้งานภายใน แอปพลิเคชัน
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ-ภาค พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูหน้าบัญชีผู้ใช้	กดปุ่มเมนูหน้าบัญชีผู้ใช้	ระบบ แสดง ผล หน้า จอ ข้อมูล ส่วนตัวโดยมีข้อมูล รูปประจำ ตัว ชื่อผู้ใช้ สาขาวิชาและภาค วิชา
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ-ภาค พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด
เมนูออกจากระบบ	กดปุ่มเมนูออกจากระบบ	ทำการ ออก จาก ระบบ และ แสดงหน้าจอข่าวสาร

- การทดสอบหน้ารายละเอียดประกาศ ในการแสดงผลหน้าจอรายละเอียดประกาศ นั้นจะประกอบไปด้วยหัวเรื่องประกาศ รายละเอียดประกาศ วันที่ประกาศและเอกสารแนบ ผลการทดสอบดังตารางที่ 5.10

ตารางที่ 5.3: 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
	เมื่อ ดาวน์โหลด เสร็จ กด ปุ่ม เปิดเอกสาร	ระบบแสดงผลเอกสาร
	กดปุ่มย้อนกลับ	ระบบแสดงผลหน้าจอรายละเอียดประกาศ
	กดปุ่มย้อนกลับอีกครั้ง	ระบบ แสดง ผล 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 กระทบ พริบ (cursor) เพื่อ ชี้ ให้ รู้ ว่า ตำแหน่งของการพิมพ์อักขระ
	พิมพ์อักขระ	ระบบ แสดง 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 ในการแสดงผลหน้าปฏิทินกำหนดการ นั้นจะประกอบไปด้วยรายการประวัติการสนทนา ช่องกรอกข้อความและปุ่มส่งข้อความ ผลการทดสอบดังตารางที่ 5.12

ตารางที่ 5.5: ผลการการทดสอบหน้าปฏิทินกำหนดการ

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

- การทดสอบหน้าดาวนโหลดเอกสาร ในการแสดงผลหน้าจอดาวนโหลดเอกสารนั้น จะประกอบไปด้วยรายการเอกสารโดยที่แต่ละฉบับจะแสดงชื่อเอกสารและปุ่มดาวนโหลดเอกสาร 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 ในการแสดงผลหน้าส่งภาพถ่ายสำเนาเอกสารนั้นจะประกอบไปด้วยปุ่มเพิ่มเอกสารฉบับที่ 1 ปุ่มเพิ่มเอกสารฉบับที่ 2 และปุ่มส่งเอกสาร ผลการทดสอบดังตารางที่ 5.7

ตารางที่ 5.7: ผลการทดสอบหน้าส่งภาพถ่ายสำเนาเอกสาร

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้า ส่ง ภาพถ่าย สำเนา เอก- สาร	กดปุ่มเมนูหน้าจอสั่งเอกสาร	ระบบ แสดง หน้า จอ สั่ง เอกสาร
	กดปุ่มเพิ่มเอกสารฉบับที่ 1	ระบบแสดงหน้าจอกล้องถ่ายภาพ
	กดปุ่มถ่ายภาพเอกสาร	ระบบแสดงผลภาพเอกสาร
	กดปุ่มถัดไป	ระบบ แสดง ผล หน้า ปรับ แต่ง ภาพเอกสาร
	กดปุ่มยืนยัน	ระบบ แสดง ผล ภาพ หน้า ส่ง ภาพถ่าย สำเนาเอกสารพร้อม ทั้งแสดงผลภาพเอกสารฉบับที่ 1
	กดปุ่มเพิ่มเอกสารฉบับที่ 2	ระบบแสดงหน้าจอกล้องถ่ายภาพ
	กดปุ่มถ่ายภาพเอกสาร	ระบบแสดงผลภาพเอกสาร
	กดปุ่มถัดไป	ระบบ แสดง ผล หน้า ปรับ แต่ง ภาพเอกสาร
	กดปุ่มยืนยัน	ระบบ แสดง ผล ภาพ หน้า ส่ง ภาพถ่าย สำเนาเอกสารพร้อม ทั้งแสดงผลภาพเอกสารฉบับที่ 1 และฉบับที่ 2
	กดส่งเอกสาร	ระบบแสดงผลการส่งเอกสาร และ แสดง สถานะ การ ตรวจ เอกสาร
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประ- ภาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

- การทดสอบหน้าจอควีสั่งเอกสาร ในการแสดงผลหน้าจอควีสั่งเอกสารนั้นจะประกอบไปด้วยปุ่มกดเลือกวันที่ ปุ่มกดเลือกเวลา ผลการทดสอบดังตารางที่ 5.8

ตารางที่ 5.8: ผลการทดสอบหน้าจอควีสั่งเอกสาร

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้าจอควีสั่งเอกสาร	กดปุ่มเมนูหน้าจอควีสั่งเอกสาร	ระบบแสดงผลหน้าจอกำหนดการสั่งเอกสาร
	กดปุ่มเลือกวันที่ที่ต้องการส่งเอกสาร	ระบบแสดงผลวันที่ถูกเลือก
	กดปุ่มเลือกเวลาที่ต้องการส่งเอกสาร	ระบบแสดงผลเวลาที่ถูกเลือก พร้อมทั้งแสดงปุ่มกดบันทึก
	กดปุ่มบันทึก	ระบบแสดงผลการจองวันที่สั่งเอกสาร
	กดปุ่มย้อนกลับ	ระบบแสดงผลหน้าจอประกาศ พร้อมทั้งแสดงรายการข่าวสารทั้งหมด

## 5.2 การทดสอบการใช้งานเว็บแอปพลิเคชัน

- การทดสอบการใช้งานเมนูนำทางของเว็บแอปพลิเคชัน การทดสอบเมนูนำทางของเว็บแอปพลิเคชันในการนำทางผู้ใช้งาน ซึ่งเมนูหลักประกอบด้วย เมนูหน้าประกาศ เมนูหน้าสนทนา เมนูหน้าปฏิทินกำหนดการ เมนูหน้าดาวน์โหลดเอกสาร 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 ส่ง 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- บรารีต่าง 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 ในการแสดงผลหน้าจอรายละเอียดประกาศ นั้นจะประกอบไปด้วยหัวเรื่องประกาศ 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: ผลการทดสอบหน้าสนทนา

การทำงาน	เงื่อนไขการทดสอบ	ผลการทดสอบ
หน้ารายละเอียดประกาศ	กดปุ่มเมนูสนทนา	ระบบแสดงผลหน้าจอสนทนา พร้อม ทั้ง แสดง รายการ ประวัติการสนทนา
	กดปุ่มที่ช่องกรอกข้อความ	ระบบ แสดง ตัว กระทบ พริบ (cursor) เพื่อ ชี้ ให้ รู้ ว่า ตำแหน่งของการพิมพ์อักขระ
	พิมพ์อักขระ	ระบบ แสดง ผลอัก ข ระ ที่ ถูก พิมพ์
	กดปุ่มส่งข้อความ	ระบบ แสดง ข้อความ ที่ ถูก พิมพ์ บน รายการ ประวัติสนทนาล่าสุด
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประกาศ 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 ในการแสดงผลหน้าจอดาวนโหลดเอกสารนั้น จะประกอบไปด้วยรายการเอกสารโดยที่แต่ละฉบับจะแสดงชื่อเอกสารและปุ่มดาวนโหลดเอกสาร 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 รายการ เอกสารในระบบ พร้อมทั้งแสดงปุ่มดาวนโหลด
	กดปุ่มย้อนกลับ	ระบบ แสดง ผล หน้า จอ ประกาศ พร้อม ทั้ง แสดง รายการ ข่าวสารทั้งหมด

## บทที่ 6

### สรุปและข้อเสนอแนะ

การดำเนินโครงการเพื่อพัฒนาระบบ XX นี้ พบว่าระบบสามารถทำงานได้ตามที่วิเคราะห์และออกแบบไว้ แต่ก็พบปัญหาและอุปสรรคระหว่างการพัฒนา ในบทนี้ผู้พัฒนาจึงขอสรุปความสามารถของระบบ ชี้แจงปัญหาและอุปสรรค พร้อมเสนอแนวทางในการพัฒนาระบบ XX ต่อ ตามลำดับ

#### 6.1 สรุปความสามารถของระบบ

ระบบ XX ทั้งเว็บแอปพลิเคชันและโมบายแอปพลิเคชันสามารถสรุปความสามารถที่ระบบทำได้ดังนี้

##### 6.1.1 เว็บแอปพลิเคชัน

ความสามารถหลักของเว็บแอปพลิเคชันนั้นเน้นสร้างความสะดวกต่อการจัดการเอกสารเรื่องข้อมูลต่างๆ ที่เกี่ยวข้องกับ XX โดยแบ่งความสามารถของระบบตามประเภทของผู้ใช้งานดังนี้

##### (a) เจ้าหน้าที่

- สร้างและแก้ไข XX ได้
- สร้างกำหนดการการดำเนินงานของ XX ได้
- สนทนากับ XX ได้
- อัปโหลดเอกสารที่เกี่ยวข้องเข้าสู่ฐานข้อมูลได้
- สร้างกำหนดการส่งสำเนาเอกสารได้

##### (b) นักศึกษา

- สมัครสมาชิกได้
- ดูประกาศได้
- ดูปฏิทินกำหนดการได้
- ดูกำหนดการส่งเอกสารได้
- แก้ไขข้อมูลส่วนตัวได้

### 6.1.2 แอนดรอยด์พลิเคชัน

เพื่อสร้างความสะดวกในการติดต่อและติดตามประกาศเนื่องจากทำงานบนอุปกรณ์พกพา  
ทั้งนี้ยังมีบางฟังก์ชันที่จำเป็นต้องทำงานบนอุปกรณ์พกพาด้วย เช่น กล้องถ่ายภาพ เป็นต้น

#### (a) เจ้าหน้าที่

- ดูประกาศได้
- ดูกำหนดการดำเนินงานของ XX ได้
- สนทนากับ XX ได้
- อัปโหลดเอกสารที่เกี่ยวข้องเข้าสู่ฐานข้อมูลได้

#### (b) นักศึกษา

- ดูประกาศได้
- ดูปฏิทินกำหนดการได้
- ดูกำหนดการส่งเอกสารได้
- ดาวน์โหลดเอกสารที่เกี่ยวข้องของ XX ได้
- สนทนากับ XX ได้
- ส่งภาพถ่ายสำเนาเอกสารได้
- รับแจ้งเตือนต่างๆ ได้

## 6.2 ปัญหาและอุปสรรคในการพัฒนา

### (a) ไลบรารี (Library) ที่ใช้ในการกรอกข้อมูลลงเอกสาร pdf ไม่รองรับภาษาไทย

แนวทางการแก้ไข : เปลี่ยนขั้นตอนการทำงานเป็นถ่ายภาพเอกสารฉบับจริงแล้ว  
ทำการส่งให้เจ้าหน้าที่ตรวจสอบเพื่อยืนยันความถูกต้อง

### (b) เนื่องจากทางผู้พัฒนามีความประสงค์ให้ระบบนี้สามารถใช้งานได้จริง ดังนั้น การพัฒนาในตอนนี้ยังมีข้อจำกัดเรื่องขนาดของเอกสารที่จัดเก็บบนโฟลเดอร์ที่สามารถอัปโหลดเข้าสู่ระบบสูงสุดเพียง 5 GB ซึ่งหากระบบถูกใช้งานจริงจำนวนข้อมูลในระบบจะเกินจำนวนที่โฟลเดอร์ให้ใช้งานฟรี

แนวทางการแก้ไข : ทำการบีบอัดข้อมูลให้มีขนาดเล็กลง ส่วนในอนาคตอาจจำเป็นต้องศึกษาแนวทางการสร้างเซิร์ฟเวอร์ (Server) เป็นของตัวเอง

### 6.3 แนวทางการพัฒนาต่อ

- (a) สร้าง Web server ของระบบซึ่งเป็นโปรแกรมที่มีหน้าที่ให้บริการด้านการจัดการเว็บไซต์และ Database server ซึ่งเป็นโปรแกรมที่ทำหน้าที่ให้บริการด้านการจัดการดูแลข้อมูลต่าง ๆ ภายในเว็บไซต์ โปรแกรมที่มีการใช้งานส่วนใหญ่เป็น mysql, postgresql, DB2
- (b) การพัฒนาส่วนแจ้งเตือนล่วงหน้าก่อนถึงกำหนดการต่าง ๆ
- (c) การพัฒนาส่วนการกรอกข้อมูลตามแบบฟอร์มของ XX ผ่านระบบได้
- (d) การพัฒนาให้ระบบเป็นระบบ e-document ทั้งระบบโดยเกี่ยวข้องกับผู้ใช้ได้แก่ XX YY และ ZZ

## 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> .

## ประวัติผู้พัฒนา

ชื่อ-สกุล: นายชื่อ สกุล

รหัสประจำตัวนักศึกษา: 5811400000

วันเกิด: XX YY 25ZZ

ที่อยู่ที่สามารถติดต่อได้: XX ม.XX ต.XX อ.XX จ.XX 34XXX

เบอร์โทรศัพท์: (+66) XX XXX XXXX

อีเมลล์: xxxxxxx.yy.59@ubu.ac.th

ระดับมัธยมต้น: โรงเรียน XX จังหวัด XX

ระดับมัธยมปลาย: โรงเรียน XX จังหวัด XX

ระดับอุดมศึกษา: ภาควิชาคณิตศาสตร์ สถิติ และคอมพิวเตอร์ สาขาวิทยาการ คอมพิวเตอร์

คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี