รายงาน

เรื่อง ระบบจัดการคิวสำหรับผู้ป่วยโควิด-19

นำเสนอ

ดร.ปริญญา เอกปริญญา

โดย

นางสาวธนภรณ์	ศรีบุญพงศ์	รหัสนักศึกษา 63010417
นางสาวธัญลักษณ์	อำไพ	รหัสนักศึกษา 63010459
นางสาวนภัสวรรณ	ละมนเทียร	รหัสนักศึกษา 63010495
นางสาวนิชาภา	พักเง้า	รหัสนักศึกษา 63010524
นางสาวประภัสสร	สุดใจใหม่	รหัสนักศึกษา 63010569
นางสาวปิยวรรณ	หน่อปาล์ม	รหัสนักศึกษา 63010608
นางสาวพัณณ์ชิตา	วรต่าย	รหัสนักศึกษา 63010666
นางสาวพิมพ์ลพัทร	สุระแย้ม	รหัสนักศึกษา 63010690
นางสาวภาวิดา	สัตยมุข	รหัสนักศึกษา 63010749

รายงานนี้เป็นส่วนหนึ่งของการศึกษา

วิชา 01076024 Software Architecture and Design

ภาคเรียนที่ 1 ปีการศึกษา 2565

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

ชื่อโครงงาน ระบบจัดการคิวสำหรับผู้ป่วยโควิด-19

ชื่อระบบ Q-vid

ผู้จัดทำ 63010417 นางสาวธนภรณ์ ศรีบุญพงศ์

63010459 นางสาวธัญลักษณ์ อำไพ

63010495 นางสาวนภัสวรรณ ละมนเทียร

63010524 นางสาวนิชาภา พักเง้า

63010569 นางสาวประภัสสร สุดใจใหม่

63010608 นางสาวปิยวรรณ หน่อปาล์ม

63010666 นางสาวพัณณ์ชิตา วรต่าย

63010690 นางสาวพิมพ์ลพัทร สุระแย้ม

63010749 นางสาวภาวิดา สัตยมุข

อาจารย์ที่ปรึกษา ดร.ปริญญา เอกปริญญา

ปีการศึกษา 2565

สืบเนื่องด้วยจากการที่มีคลัสเตอร์นักศึกษาติดเชื้อ Covid 19 แล้วต้องทำการกักตัวรักษาอาการที่หอพัก ซึ่งมักเกิดปัญหาการจัดการคิวและการจัดยาให้แก่ผู้ป่วย ไม่ว่าจะเป็นด้านการติดตามอาการ การรับเรื่อง ส่งผลให้ต่อผู้ป่วยซึ่งได้รับยาล่าช้า คณะผู้จัดทำจึงเล็งเห็นปัญหาและต้องการสร้างระบบจัดการคิวสำหรับผู้ป่วยโควิด-19 เพื่อมาใช้แก้ไขปัญหาของการรอคิวในการเข้ารับการรักษา

จากปัญหาข้างต้นพบว่าการที่ลดการส่งข้อมูล โดยจัดการคิวสำหรับผู้ป่วยโควิด-19 ให้นักศึกษาติดต่อแค่กับตัวของ chatbot ทางเดียวทำให้อำนวยความสะดวกแก่ผู้ใช้งาน อีกทั้งการทำเป็นสื่อกลางในการส่งข้อมูลให้กับสถานพยาบาลทำให้เข้าถึงข้อมูลของผู้ป่วยได้มากขึ้นและทำให้ผู้ป่วยทุกคนได้เข้ารับการรักษา

สารบัญ

เรื่อง	หน้า
บทที่ 1 บทนำ	1
1.1 ที่มาและความสำคัญ	1
1.2 วิธีการแก้ไข	1
1.3 ขอบเขตการศึกษา	2
บทที่ 2 การออกแบบ	3
2.1 Event storming	3
2.1.1. การระบุตัวตน	3
2.1.1.1. สถานะ user	3
2.1.1.2 สถานะ admin	3
2.1.2 แชทบอท	3
2.1.3 admin	3
2.1.4 ขั้นตอนการรับการรักษา	3
2.1.5 การติดตามอาการ	4
2.2 คุยกับ Domain expert	4
2.2.1 การเก็บข้อมูลประวัติส่วนตัวของผู้ป่วย	4
2.2.2 ข้อมูลอาการติดเชื้อ Covid 19	4
2.2.3 การแบ่งกลุ่มเสี่ยงผู้ป่วยและการดำเนินการกับผู้ป่วย	4

เรื่อง หน้า

2.2.3.1 โรคประจำตัวที่เข้าข่ายผู้ป่วยกลุ่มสีแดง 5

2.2.3.2 กลุ่มผู้ป่วยกลุ่มสีเขียวและสีเหลือง 5

2.2.4 การดูแลตัวเองเบื้องต้น 5

2.2.5 ยาที่ใช้ในการรักษา 5

2.2.6 แพทย์ที่มารับเคส 5

2.3 ออกแบบ UX UI 6

2.3.1 หน้าไลน์แชทบอท 6

2.3.2 หน้าเว็บแอดมินสำหรับmedical staff จัดการข้อมูล 6

2.3.2.1 หน้าlogin 6

2.3.2.2 หน้าสำหรับ admin สถิติผู้ป่วยรายวัน 7

2.3.2.3 หน้าสำหรับ admin ข้อมูลผู้ป่วยทั้งหมด 7

2.3.2.4 หน้าสำหรับ admin ข้อมูลผู้ป่วยรายคน 8

2.4 ออกแบบระบบ 8

2.4.1 Chatbot 8

2.4.2 Google sheet 8

2.4.3 ฐานข้อมูล (mySQL) 8

2.4.4 เว็บไซต์ 9

บทที่ 3 การประยุกต์ใช้จากบทเรียน 10

3. UML 10

เรื่อง หน้า

3.1 UML Component 11

3.2 Domain Model 11

3.2.1 Authenticator Context 11

3.2.2 Admin Context 12

2.2.3 Newcase Context 13

2.2.4 Followup Context 13

3.3 Design patterns 14

3.3.1 Strategy 14

3.3.2 Builder 17

3.3.3 Abstract factory 20

3.4 Quality Attribute Scenario 22

3.4.1 Availability 22

3.4.2 Integrability 22

3.4.3 Modifiability 22

3.4.4 Performance 22

3.4.5 Performance 23

3.4.6 Security 23

3.4.7 Usability 23

3.4.8 Usability 23

เรื่อง หน้า

3.5 Software Architectural Style 24

3.5.1 MVC (Model-View-Controller) 24

3.5.2 Plug-in Pabbly Connect 24

บทที่ 4 การทำงาน 25

4.1 การแบ่งงาน 25

4.1.1 การระบุตัวตนของผู้ป่วย 25

4.1.2 chatbot 25

4.1.3 แอดมิน 25

4.1.4 ขั้นตอนการรับการรักษา 25

4.1.5 การติดตามอาการ 25

4.2 Coding 26

4.3 database 26

4.4 ส่วนเก็บข้อมูลผู้ป่วย 26

4.4.1 google form 26

4.4.1.1 แบบฟอร์มข้อมูลส่วนตัวของผู้ป่วย 26

4.4.1.2 แบบฟอร์มคัดแยกอาการผู้ป่วย 26

4.4.1.3 แบบฟอร์มติดตามอาการผู้ป่วย 26

4.4.2 google sheet 26

4.4.2.1 ตารางเก็บข้อมูลส่วนตัวของผู้ป่วย 26

เรื่อง หน้า

4.4.2.2 ตารางเก็บข้อมูลอาการผู้ป่วย 26

4.4.2.3 ตารางเก็บข้อมูลติดตามอาการผู้ป่วย 27

บทที่ 5 สรุปผลการทำงาน 28

สารบัญรูปภาพ

รูป **หน้า**

รูปที่ 1 event storming 3
รูปที่ 2 หน้าของ Line chatbot 6
รูปที่ 3 หน้าเข้าสู่ระบบ 6
รูปที่ 4 หน้าแสดงสถิติผู้ป่วยรายวันของ role admin 7
รูปที่ 5 หน้าแสดงข้อมูลผู้ป่วยทั้งหมดของ role admin 7
รูปที่ 6 หน้าแสดงข้อมูลของผู้ป่วยรายคนของ role admin 8
รูปที่ 7Component Diagram 10
รูปที่ 8 Authenticator Context 11
รูปที่ 9 Admin Context 12
รูปที่ 10 Newcase Context 13
รูปที่ 11 Followup Context 13
รูปที่ 12 interface StrategyOrder ใช้กำหนดความสามารถในการเรียงลำดับข้อมูล 14
รูปที่ 13 รูป class FollowUpOrder_Timestamp สำหรับการเรียงข้อมูลตามวันเวลาที่ติดตามอาการ 15
รูปที่ 14 client เรียกใช้งาน Strategy เมื่อต้องการเรียงข้อมูลตามวันเวลาที่ติดตามอาการ 16
รูปที่ 15 ส่วน interface ของ builder.php 17
รูปที่ 16 Function สำหรับการใช้งานเพื่อ query ไปยัง sql (1) 18
รูปที่ 17 Function สำหรับการใช้งานเพื่อ query ไปยัง sql (2) 19
รูปที่ 18 ตัวอย่างการเรียกใช้งาน SQL Query Builder 19
รูปที่ 19 การแสดงจำนวน ผู้ป่วยแบบแยกประเภท ทั้งแยกตามระดับอาการผู้ป่วย และแบ่งสถานะการรักษาของผู้ป่วย 20
รูปที่ 20 class diagram แสดงการใช้ Abstract Factory 21
รูปที่ 21 ตัวอย่าง code ที่เรียกใช้ Abstract Factory ใน file : index.php 21

บทที่ 1
บทนำ

ระบบจัดการคิวสำหรับผู้ป่วยโควิด-19

ชื่อแชทบอท Q - vid

1.1 ที่มาและความสำคัญ

เนื่องด้วยสถานการณ์การแพร่ระบาดของเชื้อไวรัสโควิด 19 ส่งผลให้ประชาชนทั่วไปติดเชื้อและต้องได้รับการรักษาอย่างถูกวิธี แม้รัฐบาลจะได้ทำการจัดเตรียมการฉีดวัคซีนให้แก่บุคคลทั่วไปแล้วก็ตาม แต่เพียงแค่ฉีดวัคซีนก็ไม่สามารถช่วยให้ไม่ติดโควิดได้ เพียงแต่ระงับอาการรุนแรงและยังคงติดเชื้ออยู่ ที่ผ่านมามีผู้ติดเชื้อเป็นจำนวนมากทำให้สามารถเข้ารับการรักษาที่โรงพยาบาลได้ทุกคนทำให้ผู้ป่วยที่มีอาการไม่รุนแรงมากถูกจัดอยู่ในกลุ่มผู้ป่วยสีเขียวและสีเหลืองต้องกักตัวและรักษาตัวที่บ้าน

ทั้งนี้เนื่องจากช่วงเปิดเทอมที่ผ่านมา มีการเกิดคลัสเตอร์ใหม่ในกลุ่มนักศึกษา แล้วทำให้นักศึกษาส่วนใหญ่ได้รับผลกระทบ จากสถานการณ์ดังกล่าวทำให้มองเห็นถึงปัญหาการจัดการคิวและการจัดยาให้แก่ผู้ป่วย ไม่ว่าจะเป็นด้านการติดตามอาการ การรับเรื่อง ทำให้ส่งผลต่อตัวของผู้ป่วยซึ่งได้รับยาที่ใช้ในการรักษาช้าถึง 2 วัน อีกทั้งยังมีปัญหาการจัดการที่ ไม่เป็นระเบียบยากต่อการทำงานของบุคลากรแพทย์

1.2 วิธีการแก้ไข

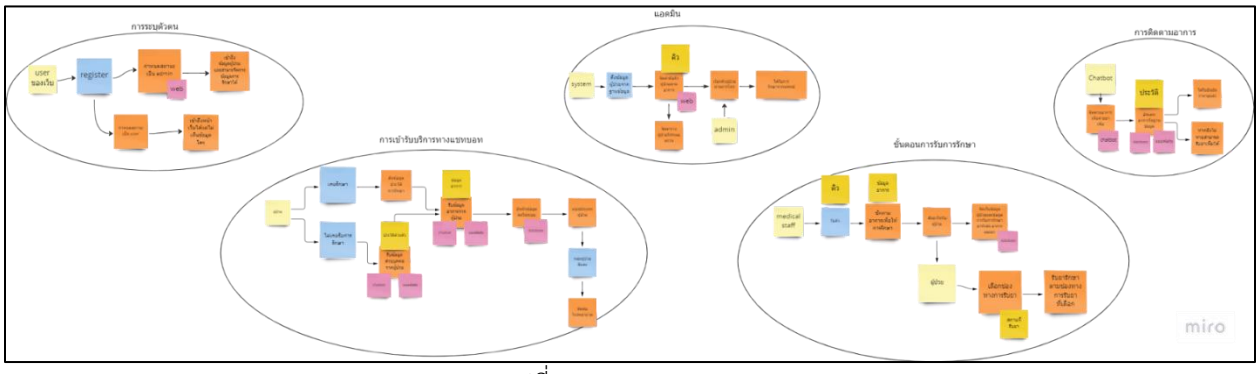
ทางคณะผู้จัดทำได้เล็งเห็นถึงปัญหานี้และเลือกที่จะพัฒนาระบบแชทบอท
เพื่อให้ง่ายต่อการเข้าถึงข้อมูลข่าวสารต่างๆ โดยจะให้แชทบอทเป็นสื่อกลางในการส่งข้อมูลข่าวสารประชาสัมพันธ์เกี่ยวกับขั้นตอนการรับการรักษาเมื่อติดเชื้อ Covid 19 โดยจะรับข้อมูลที่จำเป็นผ่านแบบฟอร์มของ google form, มีการคัดแยกกลุ่มเสี่ยงจากอาการของผู้ป่วย, การจัดลำดับคิวของผู้ป่วย, ประสานงานระหว่างผู้ป่วยและแพทย์, มีการสอบถามอาการและให้คำแนะนำในการดูแลรักษาตัวในช่วงที่กักตัวอยู่ ซึ่งในที่นี้จะสร้างเว็บ เพื่อใช้ในการจัดเก็บข้อมูล , ใช้ฐานข้อมูลเป็น Google sheet
กับ mySQL เพื่อที่เจ้าหน้าที่จะสามารถติดตามข้อมูลได้ง่าย แล้วส่งผลให้ข้อมูลของผู้ป่วยที่ติดเชื้อโควิด 19 ถูกจัดเก็บอย่างเป็นระบบ

1.3 ขอบเขตการศึกษา

1. การระบุตัวตน
2. แชทบอท
3. แอดมิน
4. ขั้นตอนการรับการรักษา
5. การติดตามอาการ

บทที่ 2
การออกแบบ

2.1 Event storming



รูปที่ 1 event storming

จากการทำ Event storming จะได้ bound context ดังนี้

2.1.1 การระบุตัวตน : กำหนด role ของ user ที่จะเข้าใช้งานเว็บไซต์โดยแบ่งเป็น

2.1.1.1 สถานะ user : ไม่สามารถเห็นข้อมูล

2.1.1.2 สถานะ admin : เข้าถึงข้อมูลส่วนตัวและข้อมูลอาการของผู้ป่วยรายบุคคลและดูสถิติของผู้ป่วยรายวันได้ โดยจะมีการแบ่งตามประเภทผู้ป่วยสีแดง สีเหลือง สีเขียวและผู้ป่วยทั้งหมด

2.1.2 แชทบอท : มีการเก็บข้อมูลจากผู้ป่วยจากการกรอกฟอร์ม โดยจะแยกเก็บข้อมูลส่วนตัวและข้อมูลอาการของผู้ป่วย จากนั้นจะบันทึกข้อมูลลงใน database และระบบจะแบ่งประเภทของผู้ป่วย หากเป็นผู้ป่วยสีแดง จะมีการแจ้งโรงพยาบาลอื่น ส่วนผู้ป่วยประเภทอื่นๆจะดำเนินการในลำดับต่อไป

2.1.3 admin : ระบบมีการจัดลำดับคิว เมื่อถึงคิว admin จะแจ้งให้ medical staff จะโทรหาผู้ป่วย เพื่อเข้าสู่ขั้นตอนการรับการรักษา

2.1.4 ขั้นตอนการรับการรักษา : medical staff ซักถามอาการเพื่อให้การรักษาและสั่งยาให้กับผู้ป่วย ผู้ป่วยจะรับยาตามที่สถานที่ที่เลือกไว้ก่อนหน้า และมีการจัดเก็บข้อมูลการรักษาใน database

2.1.5 การติดตามอาการ : ติดตามอาการจากผู้ป่วยผ่านการกรอกฟอร์ม นำข้อมูลที่ได้ไปอัพเดตใน database หากยังไม่หายสามารถรับยาเพิ่มได้

2.2 คุยกับ Domain expert

2.2.1 การเก็บข้อมูลประวัติส่วนตัวของผู้ป่วย

- ชื่อ นามสกุล
- วันเดือนปีเกิด
- เลขบัตรประชาชน
- เบอร์โทร
- ที่อยู่ผู้ป่วย กรณีต้องมีการจัดส่งยา
- วัคซีนที่ผู้ป่วยเคยได้รับ

2.2.2 ข้อมูลอาการติดเชื้อ Covid 19

- มีการแบ่งกลุ่มอาการออกเป็น 3 กลุ่มตามความรุนแรงของอาการ ได้แก่ แดง, เขียว และเหลือง
- มีอาการที่เชื่อมโยงกับความรุนแรงของผู้ป่วย คืออาการเหนื่อยหอบ ในที่นี้จะสังเกตจากการโทรคุย ผู้ป่วยจะมีการหายใจที่ผิดปกติ

2.2.3 การแบ่งกลุ่มเสี่ยงผู้ป่วยและการดำเนินการกับผู้ป่วย

2.2.3.1 โรคประจำตัวที่เข้าข่ายผู้ป่วยกลุ่มสีแดง

- โรคเบาหวาน
- โรคความดันโลหิตสูง
- โรคอ้วน
- หอบหืด ปอดอักเสบเรื้อรัง
- ภูมิคุ้มกันบกพร่อง
- ตับแข็ง ตับอักเสบเรื้อรัง
- ไตเรื้อรัง ผู้ป่วยฟอกไต และปลูกถ่ายไต
- เส้นเลือดหัวใจตีบ หัวใจเต้นผิดจังหวะ

สำหรับผู้ป่วยที่เข้าข่ายผู้ป่วยกลุ่มสีแดงจะมีการติดต่อประสานงานกับทางโรงพยาบาลในทันที

2.2.3.2 สำหรับกลุ่มผู้ป่วยกลุ่มสีเขียวและสีเหลืองจะได้รับการรักษาตามอาการ แต่ถ้าอาการของผู้ป่วยไม่ดีขึ้นหรือมีความรุนแรงของอาการมากกว่าเดิม จะถูกจัดเป็นผู้ป่วยกลุ่มสีแดงและดำเนินการประสานงานกับโรงพยาบาลทันที

2.2.4 การดูแลตัวเองเบื้องต้น

- ให้คำแนะนำ โดยยึดตามการรักษาของกระทรวงสาธารณสุข ซึ่งจะให้กักตัวระหว่างการรักษาเพียง 5 วัน
- ระหว่างการรักษาจะมีการติดตามอาการเป็นระยะเวลา 5 วัน เพื่อดูว่ามีอาการรุนแรงขึ้นหรือไม่ หากพบจะมีการส่งข้อมูลไปให้สถานพยาบาลเพื่อวางแผนการรักษา

2.2.5 ยาที่ใช้ในการรักษา

- กลุ่มผู้ป่วยสีเขียว/สีเหลืองรักษาตามอาการ เช่น ยาแก้ไข (ยาพาราเซตามอล ยาฟ้าทะลายโจร), ยาแก้ไอ, ยาลดน้ำมูก
- กลุ่มผู้ป่วยสีแดงจะมีการจ่ายยาฟาวิฟิราเวีย ทั้งนี้จะขึ้นอยู่ในความดูแลของแพทย์และสถานพยาบาลในการดูแลการรักษา

2.2.6 แพทย์ที่จะมารับเคสในการซักถามไม่จำเป็นต้องระบุชื่อ เพียงแค่ใช้เว็บนี้เป็นตัวกลางในการประสานงาน ซึ่งแพทย์จะมีการสับเปลี่ยนในแต่ละวันไม่เหมือนกัน โดยหน้าที่ของแพทย์มีเพียงแค่วิเคราะห์อาการจากที่ผู้ป่วยกรอกฟอร์มและซักถามอาการคร่าวๆ

ทั้งนี้จึงได้ข้อมูลไปเพิ่มเติมในส่วนของอาการที่จะมีการให้คะแนนที่ต่างกันไปตามความรุนแรงของอาการ และได้ข้อสรุปว่าจะมีการรับคิวตามกลุ่มอาการ ไม่จำเป็นต้องเรียงให้อาการรุนแรงมาก่อนหรือหลัง

2.3 ออกแบบ UX UI

2.3.1 หน้าไลน์แชทบอท

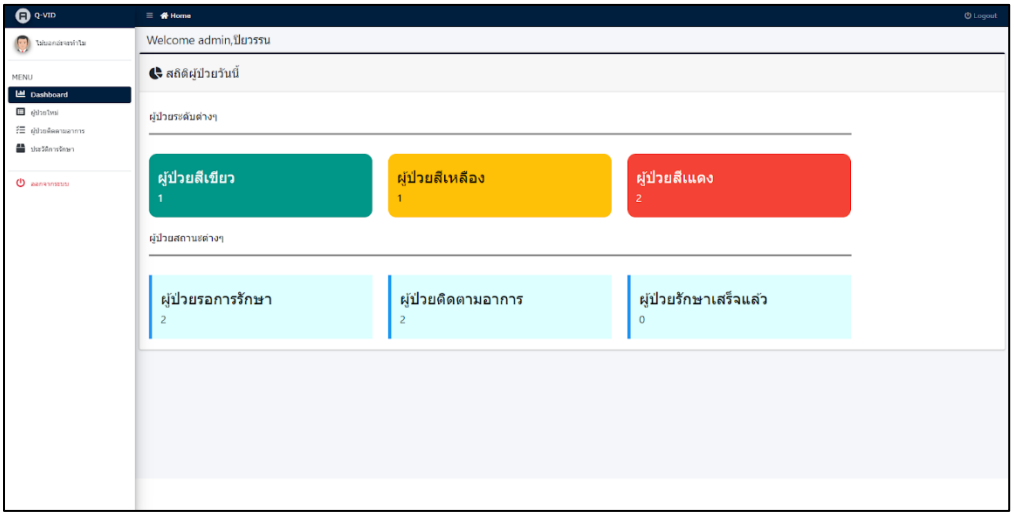
รูปที่ 2 หน้าของ Line chatbot

2.3.2 หน้าเว็บแอดมินสำหรับmedical staff จัดการข้อมูล

2.3.2.1 หน้าlogin

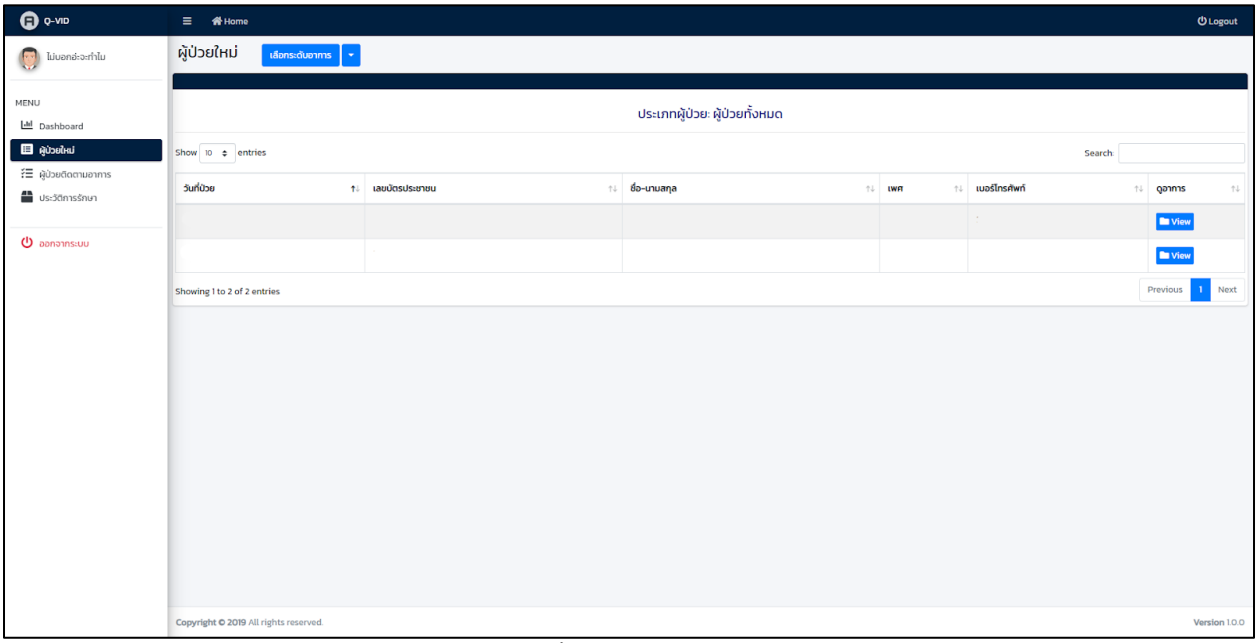
รูปที่ 3 หน้าเข้าสู่ระบบ

2.3.2.2 หน้าสำหรับ admin สถิติผู้ป่วยรายวัน



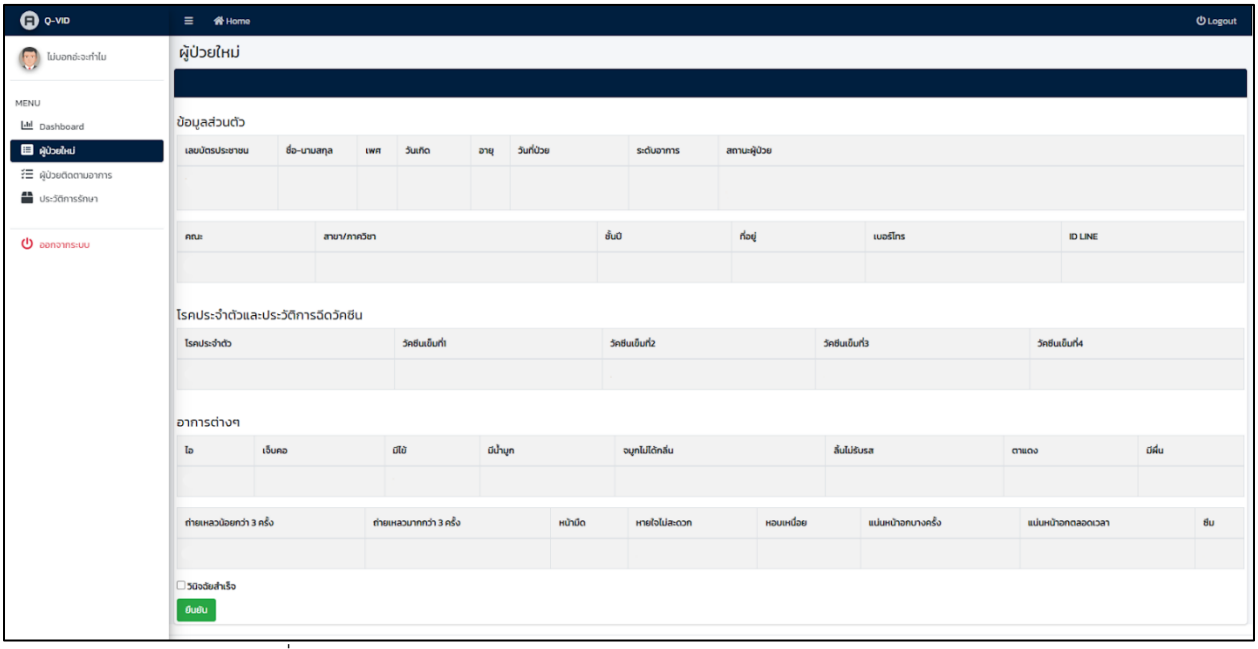
รูปที่ 4 หน้าแสดงสถิติผู้ป่วยรายวันของ role admin

2.3.2.3 หน้าสำหรับ admin ข้อมูลผู้ป่วยทั้งหมด



รูปที่ 5 หน้าแสดงข้อมูลผู้ป่วยทั้งหมดของ role admin

2.3.2.4 หน้าสำหรับ admin ข้อมูลผู้ป่วยรายคน



รูปที่ 6 หน้าแสดงข้อมูลของผู้ป่วยรายคนของ role admin

2.4 ออกแบบระบบ

2.4.1 Chatbot

- ส่งฟอร์มให้ผู้ป่วยกรอก โดยแบ่งเป็นฟอร์มประวัติส่วนตัวและฟอร์มอาการ
- ให้ผู้ป่วยกรอกฟอร์มติดตามอาการ หลังจากได้รับการรักษาจาก medical staff
- สามารถดูข้อมูลเกี่ยวกับคลินิค ไม่ว่าจะเป็นเวลาเปิด-ปิด สถานที่ตั้ง ได้ผ่านเมนูลัด
- สามารถดูขั้นตอนการจองคิวได้ผ่านเมนูลัด

2.4.2 Google sheet

- แบบฟอร์มกรอกข้อมูลส่วนตัว
- แบบฟอร์มกรอกอาการเบื้องต้น เพื่อคัดกรองกลุ่มผู้ป่วย
- แบบฟอร์มแบบฟอร์มติดตามอาการ 5 วัน

2.4.3 ฐานข้อมูล (mySQL)

- follow up : เก็บข้อมูลการติดตามอาการ
- level : เก็บข้อมูลสถานะการรักษาว่าแบ่งเป็นสถานะรอและติดตาม
- patients : เก็บข้อมูลส่วนตัวของผู้ป่วย
- sorting : เก็บข้อมูลสำหรับคัดกรองผู้ป่วย
- status : เก็บข้อมูลสถานะของผู้ป่วย

- role : เก็บข้อมูล role ของ user ที่ใช้ระบบ
- users : ตัวแปลงว่าการส่งรหัสสถานะของ admin (เวลากด) เป็นผู้ป่วยสีเขียว สีเหลือง สีแดง

2.4.4 เว็บไซต์

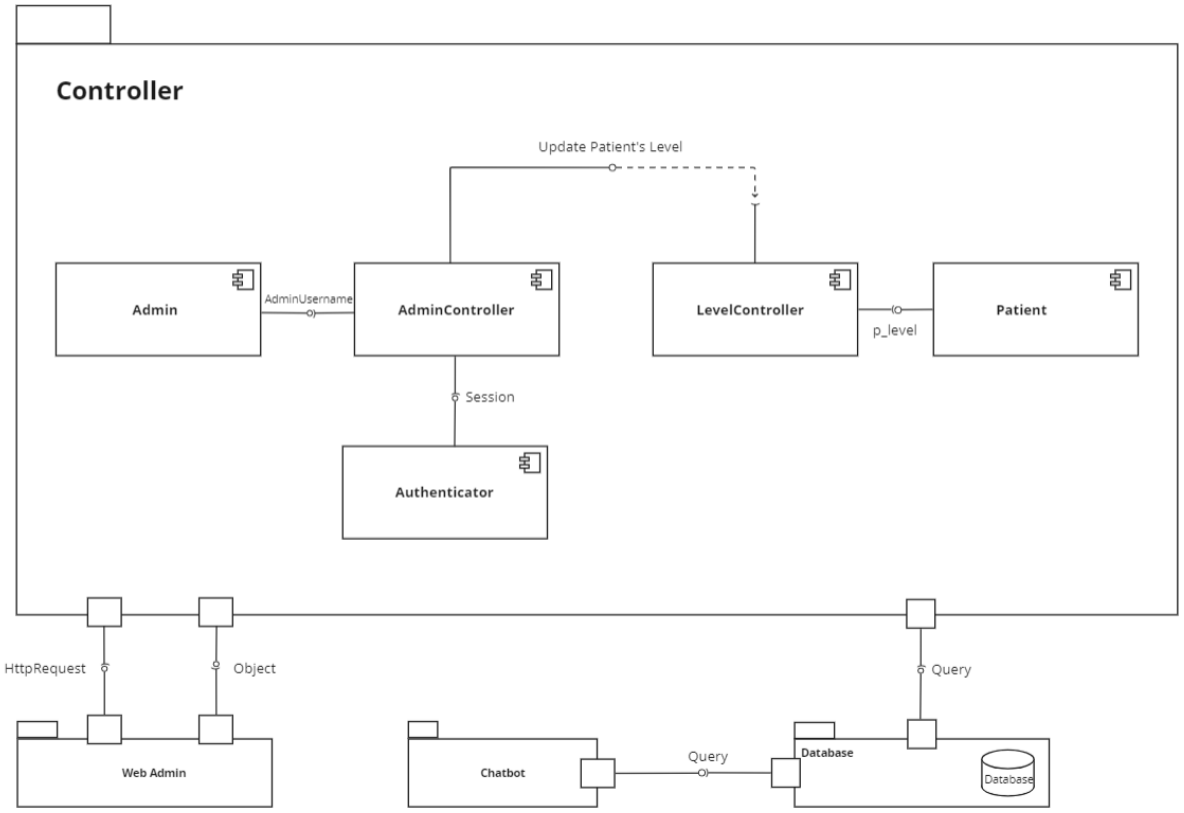
- มีการแบบ Role สำหรับ Admin และ User
- สำหรับ Admin
    - ดูหน้า dashboard เพื่อแสดงจำนวนผู้ป่วยแต่ละประเภทได้(สถานะกลุ่มผู้ป่วย, สถานะการรักษา)
    - ดูข้อมูลประวัติส่วนตัว อาการ กลุ่มอาการของผู้ป่วย สถานะการรักษาของผู้ป่วย
    - แสดงตารางติดตามอาการผู้ป่วย 5 วัน
    - เก็บประวัติการรักษา
- สำหรับ User
    - ดูหน้าแสดงรอการยืนยัน

บทที่ 3

การประยุกต์ใช้จากบทเรียน

3. UML

3.1 UML Component

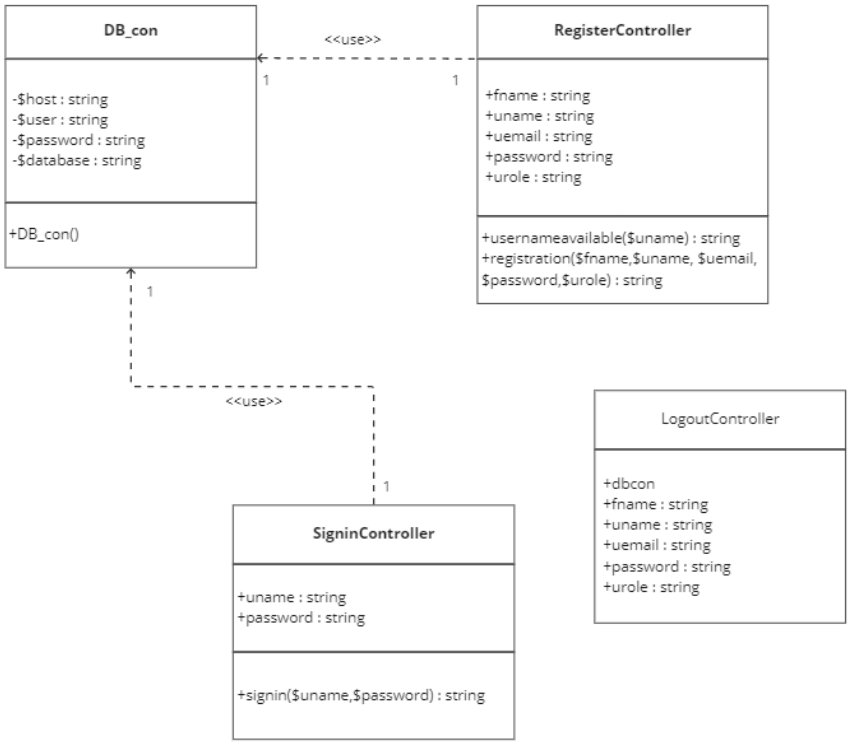


รูปที่ 7Component Diagram

3.2 Domain Model

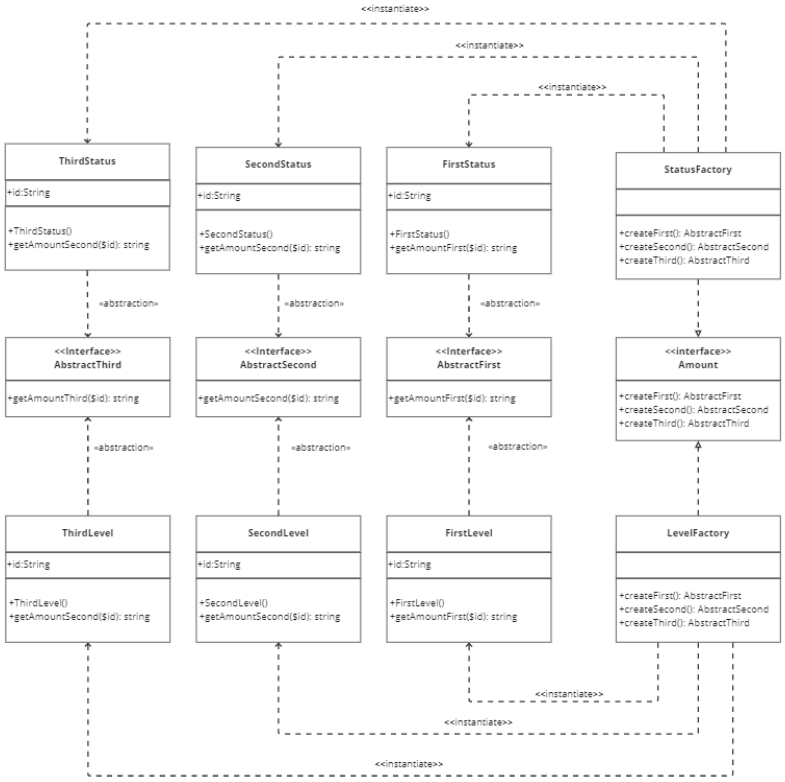
แบ่งออกเป็น 4 bounded context

3.2.1 Authenticator Context



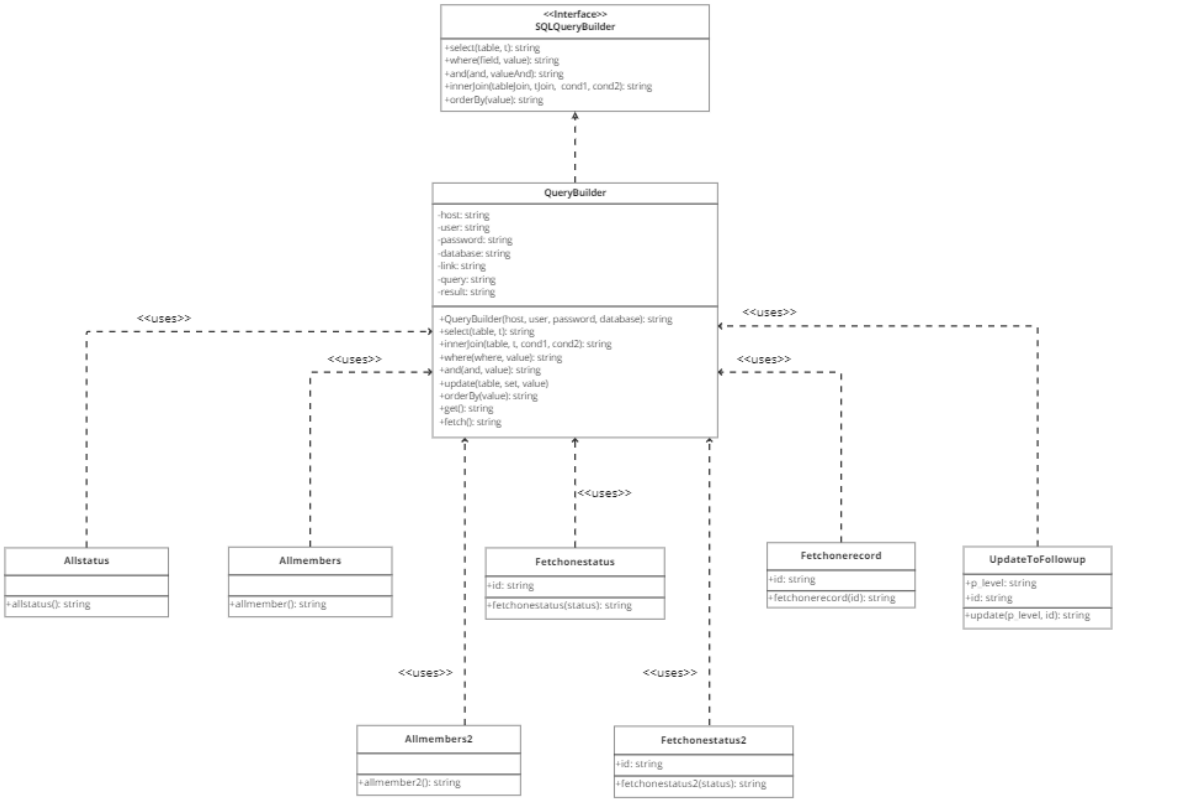
รูปที่ 8 Authenticator Context

3.2.2 Admin Context



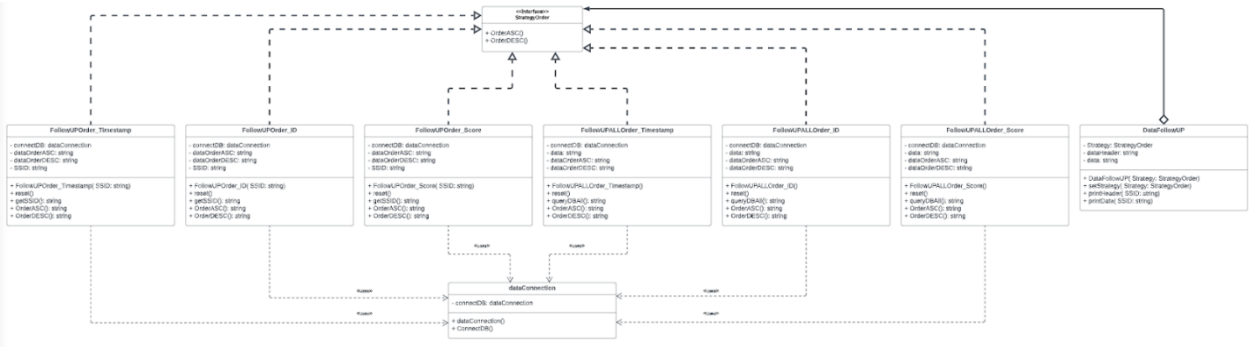
รูปที่ 9 Admin Context

3.2.3 New case Context



รูปที่ 10 Newcase Context

3.2.4 Followup Context



รูปที่ 11 Followup Context

3.3 Design patterns

3.3.1 Strategy

เหตุผลที่ใช้ Strategy ใน Behavioral Design Patterns เพราะ ต้องการขยายความสามารถของ data ใน database ให้มีการเรียงลำดับหรือจัดลำดับตามที่ต้องการเพื่อความสะดวกในการใช้งาน เช่น การเรียงลำดับวันที่ของการติดตามอาการ จะช่วยให้ดูข้อมูลง่ายขึ้นตามที่เราต้องการ หากมีการอยากอยากรู้ว่าวันไหนที่อาการหนักสุดก็สามารถเรียงได้จากอาการมากสุด - น้อยสุด หรือเรียงจากอาการน้อยสุด - มากสุด


```php
<?php
include 'dataConnection.php';

interface StrategyOrder
{
    public function OrderASC();
    public function OrderDESC();
}

class FollowUPOrder_Timestamp implements StrategyOrder
{
    private $connectDB;
    private $dataOrderASC;
    private $dataOrderDESC;
    private $SSID;
    public function __construct($SSID)
    {
        $this->SSID = $SSID;
        $this->reset();
    }
    public function reset()
    {
        $this->connectDB = new dataConnection();
    }

    public function getSSID()
    {
        return $this->SSID;
    }

    public function OrderASC()
    {
        $this->dataOrderASC = "SELECT * FROM tbl_followup AS F,tbl_patients AS P
        WHERE F.p_SSID = P.p_id AND P.p_id=$this->SSID ORDER BY F.p_timestamp ASC";
        $result = $this->connectDB->ConnectDB()->query($this->dataOrderASC);
        $result->fetch_all();
        if ($result->num_rows > 0) {
            return $result;
        } else {
            $this->reset();
        }
    }
    public function OrderDESC()
    {
        $this->dataOrderDESC = "SELECT * FROM tbl_followup AS F ,tbl_patients AS P
        WHERE F.p_SSID = P.p_id AND P.p_id=$this->SSID ORDER BY F.p_timestamp DESC";
        $result = $this->connectDB->ConnectDB()->query($this->dataOrderDESC);
        $result->fetch_all();
        if ($result->num_rows > 0) {
            return $result;
        } else {
            $this->reset();
```

รูปที่ 12 interface StrategyOrder ใช้กำหนดความสามารถในการเรียงลำดับข้อมูล

```php
class FollowUPOrder_Timestamp implements StrategyOrder
{
    private $connectDB;
    private $dataOrderASC;
    private $dataOrderDESC;
    private $SSID;
    public function __construct($SSID)
    {
        $this->SSID = $SSID;
        $this->reset();
    }
    public function reset()
    {
        $this->connectDB = new dataConnection();
    }

    public function getSSID()
    {
        return $this->SSID;
    }

    public function OrderASC()
    {
        $this->dataOrderASC = "SELECT * FROM tbl_followup AS F,tbl_patients AS P
        WHERE F.p_SSID = P.p_id AND P.p_id=$this->SSID ORDER BY F.p_timestamp ASC";
        $result = $this->connectDB->ConnectDB()->query($this->dataOrderASC);
        $result->fetch_all();
        if ($result->num_rows > 0) {
            return $result;
        } else {
            $this->reset();
        }
    }
    public function OrderDESC()
    {
        $this->dataOrderDESC = "SELECT * FROM tbl_followup AS F ,tbl_patients AS P
        WHERE F.p_SSID = P.p_id AND P.p_id=$this->SSID ORDER BY F.p_timestamp DESC";
        $result = $this->connectDB->ConnectDB()->query($this->dataOrderDESC);
        $result->fetch_all();
        if ($result->num_rows > 0) {
            return $result;
        } else {
            $this->reset();
        }
    }
}
```

รูปที่ 13 รูป class FollowUpOrder_Timestamp สำหรับการเรียงข้อมูลตามวันเวลาที่ติดตามอาการ

จากรูป class FollowUpOrder_Timestamp เป็นหนึ่งใน class ที่อยู่ใน UML diagram ซึ่งอยู่ใน Followup Context ณ ที่นี้ เป็นการยกตัวอย่างการ code โดยทำหน้าที่เรียงลำดับข้อมูลอาการที่ติดตามต่อวันของคนไข้แบบตามวันและเวลาที่ได้ส่งฟอร์มผ่านทางแชทบอทสามารถเรียงได้ 2 รูปแบบ ตาม interface StrategyOrder คือ เรียงจากวันเวลาล่าสุด-วันเวลานานสุด และ วันเวลานานสุด - วันเวลาล่าสุด ในส่วน class อื่นๆใน Followup Context จะเรียงลำดับข้อมูลต่างกันเนื่องด้วยเห็นถึงปัญหาที่ผู้ใช้งานไม่ได้ต้องการเรียงลำดับเพียงแค่วันเวลา อาจต้องการเรียงตาม id ของข้อมูลติดตามอาการ

Class ใน Followup Context และความสามารถ มีดังนี้

- StrategyOrder ใช้กำหนดความสามารถในการเรียงลำดับข้อมูล
- FollowUpOrder_Timestamp เรียงลำดับข้อมูลส่วนบุคคลตามวันเวลาที่ส่งข้อมูลติดตามอาการ
- FollowUpOrder_ID เรียงลำดับข้อมูลส่วนบุคคลตาม id ของข้อมูลติดตามอาการ
- FollowUpOrder_Score เรียงลำดับข้อมูลส่วนบุคคลตามความรุนแรงของอาการ
- FollowUpAllOrder_Timestamp เรียงลำดับข้อมูลทั้งหมดตามวันเวลาที่ส่งข้อมูลติดตามอาการ
- FollowUpAllOrder_ID เรียงลำดับข้อมูลทั้งหมดตาม id ของข้อมูลติดตามอาการ
- FollowUpAllOrder_Score เรียงลำดับข้อมูลทั้งหมดตามความรุนแรงของอาการ
- DataFollowUP สำหรับเรียกใช้ StrategyOrder ซึ่งการเรียกใช้ ณ ที่นี้ ก็ตามความต้องการของผู้ใช้ และสำหรับแสดงข้อมูลผลลัพธ์ที่ถูกประมวลผลแล้ว

```php
<?php
$SSID = $_GET['p_id'];
$PatientFollowing = new DataFollowUP(new FollowUPOrder_Timestamp($SSID));
$PatientFollowing->printHeader($SSID);
$PatientFollowing->printData($SSID);
?>
```

รูปที่ 14 client เรียกใช้งาน Strategy เมื่อต้องการเรียงข้อมูลตามวันเวลาที่ติดตามอาการ

3.3.2 Builder

นำมาใช้ในการทำ SQL Query Builder เพื่อให้สามารถทำการสร้าง SQL query ได้ง่าย รวดเร็ว และสะดวกยิ่งขึ้น เพราะได้สร้างคำสั่งของการเข้าถึง sql เช่น select, inner join, where, order by และ update เป็นต้น โดยนำคำสั่งมาใช้เป็นแต่ละส่วนมาประกอบกันเพื่อเข้าถึง database ทำให้ไม่ซับซ้อน เป็นระเบียบ และหาจุดผิดพลาดได้ง่ายกว่าการ query แบบปกติ ใช้ในการเข้าถึงข้อมูลที่ต้องการนำมาแสดง ทั้งการแสดงคิวของผู้ป่วยใหม่ทั้งหมด ข้อมูลส่วนตัวของผู้ป่วยใหม่ รายการผู้ป่วยใหม่ตามระดับอาการต่างๆ และใช้ในการอัพเดทสถานะการรักษาของผู้ป่วยเพื่อรอติดตามอาการต่อไป

```
interface SQLQueryBuilder
{
    public function select(string $table,string $t);

    public function where(string $field, string $value);

        public function and(string $and, string $valueAnd);

        public function innerJoin(string $table, string $t, string $cond1, string $cond2);

        public function update(string $table, string $set, string $value);

        public function orderBy(string $value);
}
```

รูปที่ 15 ส่วน interface ของ builder.php

```php
class QueryBuilder implements SQLQueryBuilder {
        // for database connection
        private $host;
        private $user;
        private $password;
        private $database;

        private $link = NULL;

        private $query;
        private $result;

        // constructor
        public function __construct($host, $user, $password, $database)
        {
                if (FALSE === ($this->link = mysqli_connect($host, $user, $password, $database))) {
                        throw new Exception('Error', mysqli_connect_error());
                }
        }

        public function select($table,$t)
        {
                $this->query = 'SELECT * FROM '.$table.' AS '.$t;
                return $this;
        }

    public function innerJoin($table = NULL, $t = NULL,$cond1= NULL,$cond2= NULL)
        {
                $this->query .= ' INNER JOIN '.$table. ' AS '.$t .' ON '.$cond1. '='.$cond2;
                return $this;
        }

        public function where($where = NULL, $value = NULL)
        {
                $this->query .= ' WHERE '.$where. '='.$value;
                return $this;
        }
```

รูปที่ 16 Function สำหรับการใช้งานเพื่อ query ไปยัง sql (1)

```php
public function and($and = NULL, $value = NULL)
{
        $this->query .= ' AND '.$and. '='.$value;
        return $this;
}

public function update($table = NULL, $set = NULL, $value = NULL)
{
        $this->query .= 'UPDATE '.$table.' SET '.$set. '='.$value;
        return $this;
}

public function orderBy($value = 'id desc')
{
        $this->query .= ' ORDER BY '.$value;
        return $this;
}

// Final method for testing this chaining worked
public function get()
{
        $this->result = $this->query;
        return $this->result;
}

// Final method return as fetch
public function fetch()
{
        if (FALSE === ($this->result = mysqli_query($this->link, $this->query))) {
                throw new Exception('Error Query', $this->query);
        }
        return $this->result;
}
```

รูปที่ 17 Function สำหรับการใช้งานเพื่อ query ไปยัง sql (2)

Function สำหรับการใช้งานเพื่อ query ไปยัง sql

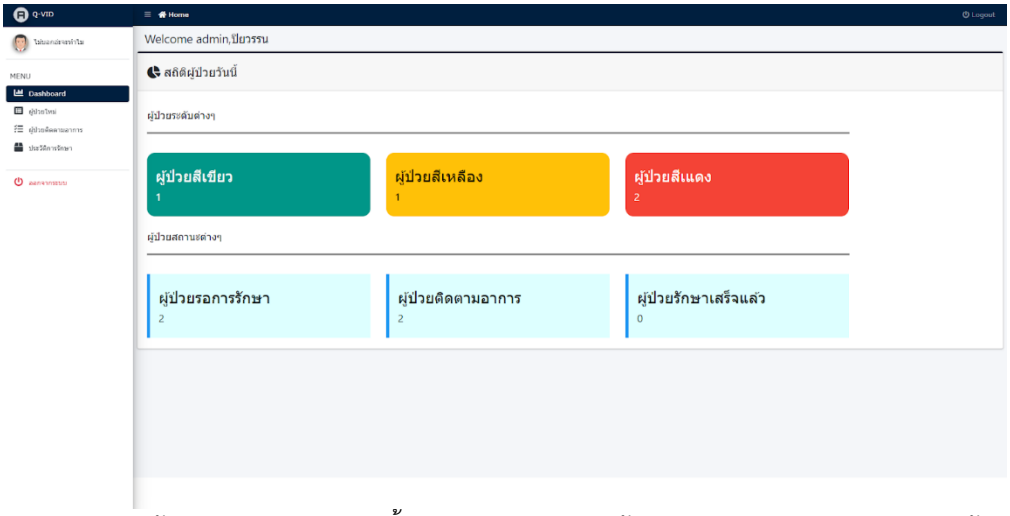
```php
class Fetchhistory {
    public function fetchhistory() {
        $builder = new QueryBuilder(DB_SERVER, DB_USER, DB_PASS, DB_NAME);
        $result = $builder
            ->select('tbl_patients','p')
            ->innerJoin('tbl_sorting','s','p.p_id','s.p_id')
            ->innerJoin('tbl_status','st','s.s_status','st.s_status')
            ->where('p_level','3')
            ->fetch();
        return $result;
```

รูปที่ 18 ตัวอย่างการเรียกใช้งาน SQL Query Builder

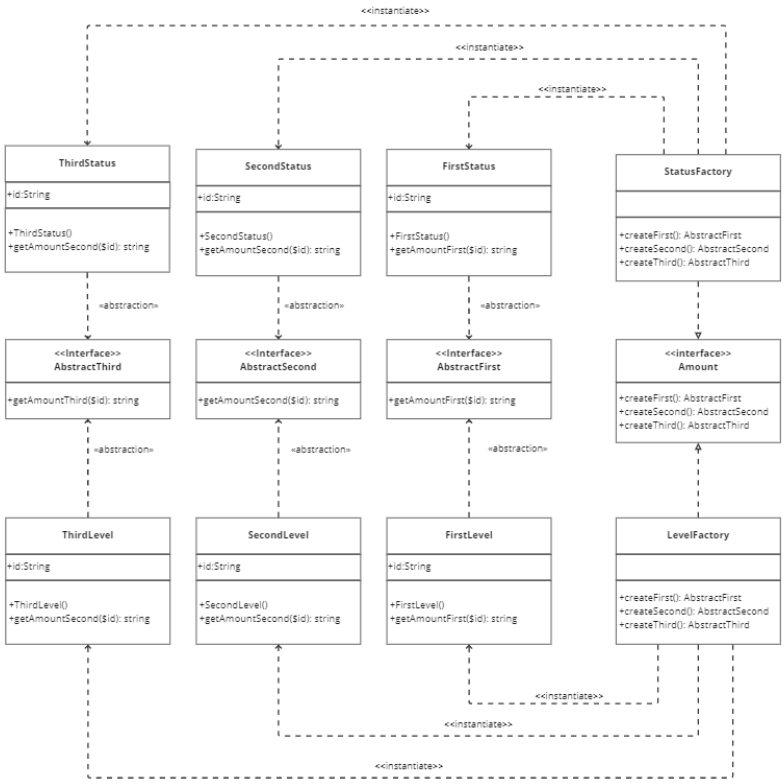
3.3.3 Abstract Factory

เหตุผลที่นำ Abstract Factory เพราะต้องการสร้างกลุ่มของ Object ที่เกี่ยวข้องกันเป็นตระกูล โดยที่ไม่ต้องระบุ subclass ว่าชื่ออะไร

โดยกลุ่มของเรานำมาใช้ในการแสดงจำนวน ผู้ป่วยแบบแยกประเภท ทั้งแยกตามระดับอาการผู้ป่วย และแบ่งสถานะการรักษาของผู้ป่วย



รูปที่ 19 การแสดงจำนวน ผู้ป่วยแบบแยกประเภท ทั้งแยกตามระดับอาการผู้ป่วย และแบ่งสถานะการรักษาของผู้ป่วย โดยใช้ Abstract Factory เป็น Design Pattern



รูปที่ 20 class diagram แสดงการใช้ Abstract Factory

```php
<h4>ผู้ป่วยสถานะต่างๆ</h4>
<hr>

<div class="row">
    <div class="col-md-4">
        <div class="w3-panel w3-padding-16 w3-pale-blue w3-leftbar w3-border-blue">
            <h2>ผู้ป่วยรอการรักษา</h2>
            <h4><?php
            $factory = new LevelFactory();
            $First = $factory->createFirst();
            echo $First->getAmountFirst("1") . "\n";
            ?></h4>
        </div>/.w3-panel.w3-padding-16.w3-pale-blue.w3-leftbar.w3-border-blue
    </div>/.col-md-4
    <div class="col-md-4">
        <div class="w3-panel w3-padding-16 w3-pale-blue w3-leftbar w3-border-blue">
            <h2>ผู้ป่วยติดตามอาการ</h2>
            <h4><?php
            $factory = new LevelFactory();
            $Second = $factory->createSecond();
            echo $Second->getAmountSecond("2") . "\n";
            ?></h4>
        </div>/.w3-panel.w3-padding-16.w3-pale-blue.w3-leftbar.w3-border-blue
    </div>/.col-md-4
    <div class="col-md-4">
        <div class="w3-panel w3-padding-16 w3-pale-blue w3-leftbar w3-border-blue">
            <h2>ผู้ป่วยรักษาเสร็จแล้ว</h2>
            <h4><?php
            $factory = new LevelFactory();
            $Third = $factory->createThird();
            echo $Third->getAmountThird("3") . "\n";
            ?></h4>
            <!-- <h4>รู้หมุน <?php echo clientCode(new LevelFactory("3")); ?> คน</h4> -->
        </div>/.w3-panel.w3-padding-16.w3-pale-blue.w3-leftbar.w3-border-blue
    </div>/.col-md-4
</div>/.row
  <!-- ./col -->
```

รูปที่ 21 ตัวอย่าง code ที่เรียกใช้ Abstract Factory ใน file : index.php

3.4 Quality Attribute Scenario

3.4.1 Availability

- Source of stimulus : Developer
- Stimulus : Server ปิดตัวลง
- Environment : Runtime
- Artifacts : Server
- Response : ทำให้ Server รันได้ปกติ
- Response measure : Server กลับมาใช้งานได้ปกติภายในเวลา 2 ชม.

3.4.2 Integrability

- Source of stimulus : Users
- Stimulus : ต้องการเข้าใช้งานหน้าเว็บไซต์
- Environment : เว็บไซต์
- Artifacts : System services
- Response : User ที่มี role ต่างกันก็จะเข้าใช้งานหน้าเว็บไซต์ได้ต่างกัน
- Response measure : จำนวนการเข้าถึงจากคนที่มี role ต่างๆ

3.4.3 Modifiability

- Source of stimulus : Developers
- Stimulus : ต้องการเพิ่ม ลบ แก้ไขคลาสฟังก์ชันในโครงสร้าง
- Environment : Design time
- Artifacts : Code
- Response : Make modification
- Response measure : Abstract method pattern ทำให้เพิ่มคลาสได้ง่าย

3.4.4 Performance

- Source of stimulus : google form
- Stimulus : เพิ่มข้อมูลใน Database
- Environment : normal mode
- Artifacts : Server ของเว็บ, Database
- Response : เชื่อมต่อ web กับ Database ได้อย่าง real time
- Response measure : เมื่อผู้ป่วยกรอกฟอร์มเสร็จสิ้น ระบบจะใช้เวลาในการบันทึกข้อมูลลงในฐานข้อมูล < 2 นาที

3.4.5 Performance

- Source of stimulus : ผู้ป่วย
- Stimulus : พิมพ์สอบถามข้อมูลจาก chatbot
- Environment : Normal mode
- Artifacts : Chatbot
- Response : การตอบสนองของ chatbot
- Response measure : ตอบกลับผู้ป่วยภายในเวลาไม่เกิน 1 นาที

3.4.6 Security

- Source of stimulus : Users ที่ยังไม่มี role admin
- Stimulus : ต้องการเข้าใช้งานหน้าเว็บไซต์
- Environment : เว็บไซต์
- Artifacts : System services
- Response : ไม่ให้ user ที่ไม่มี role เป็น admin เข้าใช้งานหน้าเว็บ
- Response measure : unauthorized access = 0%

3.4.7 Usability

- Source of stimulus : ผู้ป่วย
- Stimulus : ต้องการใช้งาน chatbot
- Environment : Runtime
- Artifacts : Chatbot
- Response : มีตัวช่วยในการตอบ, เมนูลัดสำหรับแสดงข้อมูลต่างๆ
- Response measure : line สามารถรองรับได้หลาย device

3.4.8 Usability

- Source of stimulus : ผู้ป่วย
- Stimulus : ต้องการกรอกข้อมูลเพื่อรับการรักษา
- Environment : Runtime
- Artifacts : Chatbot
- Response : ลิ้งค์google formเพื่อกรอกข้อมูลส่วนตัวและหลักฐานการติดเชื้อ
- Response measure : ตอบกลับภายในเวลาไม่เกิน 1 นาที

3.5 Software Architectural Style

3.5.1 MVC (Model-View-Controller)

นำมาใช้ในการแบ่ง code เป็นส่วนๆแยกการทำงานชัดเจน เพื่อให้ง่ายต่อการปรับเปลี่ยน เช่น ปรับเปลี่ยน code ในส่วนของ view ก็จะไม่ส่งผลกระทบกับส่วนอื่นๆ

การจัดองค์ประกอบโค้ดที่

3.5.2 Plug-in Pabbly Connect

Pabbly Connect เป็น integration platform ที่ช่วยให้สามารถผสานรวมหลาย ๆ แอพพลิเคชั่นโดยจัดการ data flow ได้อย่างราบรื่น ซึ่งเราใช้เป็นส่วนเสริมในการเชื่อมต่อระหว่างข้อมูลจาก google sheet และ database

บทที่ 4
การทำงาน

4.1 การแบ่งงาน

แบ่งงานตาม 5 bound context ดังนี้

4.1.1 การระบุตัวตนของผู้ป่วย

ผู้รับผิดชอบมีดังนี้

1. นางสาวประภัสสร สุดใจใหม่
2. นางสาวธัญลักษณ์ อำไพ

4.1.2 chatbot

ผู้รับผิดชอบมีดังนี้

1. นางสาวธนภรณ์ ศรีบุญพงศ์
2. นางสาวนิชาภา พักเง้า

4.1.3 แอดมิน

ผู้รับผิดชอบมีดังนี้

1. นางสาวนภัสวรรณ ละมนเทียร
2. นางสาวภาวิดา สัตยมุข

4.1.4 ขั้นตอนการรับการรักษา

ผู้รับผิดชอบมีดังนี้

1. นางสาวนภัสวรรณ ละมนเทียร
2. นางสาวปิยวรรณ หน่อปาล์ม
3. นางสาวภาวิดา สัตยมุข

4.1.5 การติดตามอาการ

ผู้รับผิดชอบมีดังนี้

1. นางสาวพัณณ์ชิตา วรต่าย
2. นางสาวพิมพ์ลพัทร สุระแย้ม

4.2 Coding

4.2.1 Chatbot

ใช้ dialogflow ในการพัฒนา chatbot เนื่องจากเป็นเครื่องมือที่ช่วยในการวิเคราะห์หาข้อความที่ผู้ใช้พิมพ์มานั้น มีเนื้อความแบบใด โดยใช้วิธีด้าน Natural Language Processing (NLP) และนอกจากนี้ยังช่วยในการจัดการส่งข้อมูลต่างๆ ในระบบให้ผู้ใช้งานสามารถใช้งานได้สะดวกยิ่งขึ้น

4.2.2 Web admin

ใช้ PHP ในการพัฒนา ซึ่ง PHP เป็นภาษาสำหรับใช้ในการเขียนโปรแกรมบนเว็บไซต์ เขียนได้หลากหลาย สนับสนุนการเขียนโปรแกรมเชิงวัตถุ(OOP) และมีความสามารถในการทำงานร่วมกับระบบจัดการฐานข้อมูลที่หลากหลาย

4.3 Database

ใช้ mySQL เป็นฐานข้อมูล เนื่องจาก mySQL สนับสนุนการทำงานได้เกือบทุกระบบปฏิบัติการณ์ และเป็นฐานข้อมูลที่มีการจัดการฐานข้อมูลแบบโครงสร้างในรูปแบบของตารางเพื่อช่วยให้สามารถเข้าหาและสืบค้นข้อมูลได้ง่าย ส่งผลให้การทำงานนั้นรวดเร็วและยืดหยุ่น และข้อมูลทุกๆตารางจะเชื่อมโยงกันทำให้สามารถจัดการข้อมูลต่างๆได้ตามต้องการ และอีกทั้งยังเป็น Open Source License ที่สามารถใช้งานได้ฟรี

4.4 ส่วนเก็บข้อมูลผู้ป่วย

4.4.1 google form

4.4.1.1 แบบฟอร์มข้อมูลส่วนตัวของผู้ป่วย

แบบฟอร์มข้อมูลส่วนตัวของผู้ติดโควิด (google.com)

4.4.1.2 แบบฟอร์มคัดแยกอาการผู้ป่วย

คัดแยกอาการผู้ป่วย COVID-19 (google.com)

4.4.1.3 แบบฟอร์มติดตามอาการผู้ป่วย

แบบสอบถามติดตามอาการผู้ป่วยCOIVD-19 (google.com)

4.4.2 google sheet

4.4.2.1 ตารางเก็บข้อมูลส่วนตัวของผู้ป่วย

ข้อมูลส่วนตัวของผู้ติดโควิด - Google ชีต

4.4.2.2 ตารางเก็บข้อมูลอาการผู้ป่วย

คัดแยกอาการผู้ป่วย (การตอบกลับ) - Google ชีต

4.4.2.3 ตารางเก็บข้อมูลติดตามอาการผู้ป่วย

ติดตามอาการ (การตอบกลับ) - Google ชีต

บทที่ 5
สรุปผลการทำงาน

จากการแบ่งงานสมาชิกในกลุ่มทุกคนทำหน้าที่ตามที่ได้รับมอบหมายได้ดีโดยเริ่มการศึกษาในส่วนของฐานข้อมูล(Database) ที่ใช้ร่วมกับภาษา php และการทำ chatbot ด้วย Dialogflow แล้วมีการพัฒนาฟอร์มในการเก็บข้อมูลให้สมบูรณ์แบบ หลังจากสอบถาม Domain Expert ซึ่งการทำงานช่วงแรกจะเป็นการแยกกันทำตามหน้าที่ โดยเมื่อพัฒนาในแต่ละครั้งจะให้สมาชิกทุกคนทดสอบระบบเสมอ และเมื่อพัฒนาสมบูรณ์แล้วจึงนำส่วนประกอบต่างๆมารวมกัน

เนื่องได้นำองค์ความรู้ด้าน Software Architecture มาใช้ ทำให้เมื่อมีการนำ code มารวมกันแล้ว ใช้เวลาในการจัดการ และแก้ไข code ให้ไปด้วยกันได้ภายในเวลาไม่นาน และทำให้ระบบมีประสิทธิภาพที่ดี