**ปริญญานิพนธ์**

**ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย**

**MEDICATION REMINDER MANAGEMENT SYSTEM**

นายชาตรี พรหมประเสริฐ

นายศุภภัทร งามพันธุ์ดิศร

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร  
ครุศาสตร์อุตสาหกรรมบัณฑิต (ครุศาสตร์วิศวกรรม)  
สาขาวิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรมและเทคโนโลยี  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง  
ปีการศึกษา 2560

**ปริญญานิพนธ์**

**ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย**

**MEDICATION REMINDER MANAGEMENT SYSTEM**

นายชาตรี พรหมประเสริฐ

นายศุภภัทร งามพันธุ์ดิศร

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร  
ครุศาสตร์อุตสาหกรรมบัณฑิต (ครุศาสตร์วิศวกรรม)  
สาขาวิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรมและเทคโนโลยี  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง  
ปีการศึกษา 2560

**ปริญญานิพนธ์**

**เรื่อง ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย**

**Medication Reminder Management System**

**วัตถุประสงค์**

1. เพื่อศึกษาการใช้งาน MEAN Stack ในการสร้าง Server ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. เพื่อออกแบบการทำงานในส่วนต่าง ๆ ของระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

3. เพื่อสร้างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

4. เพื่อทดลองการใช้งานและสามารถนำระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยไปใช้งานกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วย

**ประโยชน์ที่คาดว่าจะได้รับ**

1. ได้รับความรู้เกี่ยวกับการใช้งาน MEAN Stack ในการสร้าง Server ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. ได้หน้าเพจเพื่อใช้จัดการกับข้อมูลต่าง ๆ ในฐานข้อมูลของ Server

3. ได้ Server สำหรับจัดการกับข้อมูลต่าง ๆ ของผู้ป่วย

4. ได้ผลการทดลองใช้งานของระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

5. ได้ Server ของระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยที่สามารถใช้งานกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วย

**ชื่อหัวข้อ** ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

นั**กศึกษา** ชาตรี พรหมประเสริฐ

ศุภภัทร งามพันธุ์ดิศร

**อาจารย์ที่ปรึกษา** ผศ.ดร.วรวิทย์ สมหา

**ปริญญา**  ครุศาสตร์อุตสาหกรรมบัณฑิต

**หลักสูตร** ครุศาสตร์วิศวกรรม

**ปีการศึกษา** 2560

**บทคัดย่อ**

ปริญญานิพนธ์ฉบับนี้เสนอการสร้างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยเพื่อใช้เป็นฐานข้อมูลสำหรับจัดการกับข้อมูลจากแอปพลิเคชันแจ้งเตือนการรับประทานของผู้ป่วยโดยการทำงานผ่านทางหน้าเว็บแอปพลิเคชันซึ่งประกอบไปด้วย หน้าเว็บเพจสำหรับเพิ่ม ลบ แก้ไขผู้ใช้งานระบบ ใช้กำหนดสิทธิ์ในการเข้าถึงหรือจัดการกับข้อมูลในระบบ หน้าเว็บเพจสำหรับแสดงข้อมูลในระบบที่มีการรับส่งจากแอปพลิเคชัน หน้าเว็บเพจสำหรับอัพเดต แก้ไข ลบข้อมูลจากแอปพลิเคชัน และหน้าเว็บเพจสำหรับติดต่อหรือให้คำแนะนำเกี่ยวกับการใช้งานระบบ

จากการทดลองใช้งานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยกับฝั่งแอปพลิเคชันพบว่า ระบบสามารถติดต่อกับแอปพลิเคชันเพื่อรับส่งข้อมูลและผู้ใช้งานระบบสามารถจัดการกับข้อมูลผ่านทางหน้าเว็บแอปพลิเคชันได้ตามที่ต้องการ

**Thesis Title** Medication Reminder Management System

**Students** Mr.Chatree Promprasert

Mr.Supapat Ngampandisorn

**Advisor** Asst.Prof.Dr.Worawit Somha

**Degree** Bachelor of Science in Industrial Education

**Program in** Engineering Education

**Academic Year**  2017

**ABSTRACT**

This thesis presents the Medication Reminder Management System. To use as a database to handle data from the Medication Reminder Application. By working through a web page, the application includes: Web page for adding, removing, editing system or users. Set permissions to access or manipulate data in the system. A web page for displaying information on the system that is receive and sent data from the application. Web page for update , delete data from the application. And web pages for contact or advice on using the system.

The test results of the Medication Reminder Management System, this system can contact with application for sent - receive and management information by Medication Reminder Application as user require.

**กิตติกรรมประกาศ**

ปริญญานิพนธ์ฉบับนี้สำเร็จไปได้ด้วยดีเนื่องมาจากการช่วยเหลือและการร่วมมือกันของสมาชิกภายในกลุ่ม คณะผู้จัดทำปริญญานิพนธ์ขอขอบพระคุณท่านผู้ช่วยศาสตราจารย์ด็อกเตอร์วรวิทย์ สมหา อาจารย์ที่ปรึกษาปริญญานิพนธ์ ที่คอยให้คำปรึกษา คำแนะนำ และคอยแนะแนวทางในการแก้ไขปัญหาต่าง ๆ ด้วยความเอาใจใส่ทุกขั้นตอนในการจัดทำปริญญานิพนธ์นี้ ขอขอบพระคุณท่านอาจารย์ปิยะ จิตธรรมมาภิรมย์ ท่านรองศาสตราจารย์ด็อกเตอร์พรพิมล ฉายรัศมี และขอขอบพระคุณอาจารย์ประจำภาควิชาครุศาสตร์วิศวกรรมทุกท่าน ที่ได้กรุณาให้คำแนะนำ ให้ข้อเสนอแนะและแนวทางในการทำงาน ตลอดจนข้อคิดที่สามารถนำไปปรับใช้ในการจัดทำปริญญานิพนธ์นี้ เพื่อให้สามารถสำเร็จลุล่วงไปได้ด้วยดี

ขอกราบขอบพระคุณ บิดา มารดาและผู้มีพระคุณ ที่คอยสนับสนุนด้านการศึกษาตลอดมา คอยให้กำลังใจ ตลอดจนให้คำปรึกษาในทุก ๆ เรื่องและสุดท้าย ขอขอบพระคุณทุก ๆ ท่านที่เป็นกำลังใจให้เสมอมา คณะผู้จัดทำจึงขอขอบคุณมา ณ โอกาสนี้

**สารบัญ**

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| บทคัดย่อภาษาไทย | **I** |
| บทคัดย่อภาษาอังกฤษ | **II** |
| กิตติกรรมประกาศ | **III** |
| สารบัญ | **IV** |
| สารบัญตาราง | **VII** |
| สารบัญรูป | **VIII** |
| บทที่ 1 บทนำ | 1 |
| 1.1 ความเป็นมาและความสำคัญของปัญหา | 1 |
| 1.2 จุดมุ่งหมายของโครงงาน | 1 |
| 1.3 สมมุติฐานของการจัดทำโครงงาน | 2 |
| 1.4 ขีดความสามารถของโครงงาน | 2 |
| 1.5 ขั้นตอนของการทำโครงงาน | 3 |
| 1.6 เนื้อหาโดยสังเขป | 3 |
| บทที่ 2 ทฤษฎีและหลักการ | 5 |
| 2.1 กล่าวนำ | 5 |
| 2.2 เว็บแอปพลิเคชัน | 5 |
| 2.2.1 ส่วนติดต่อกับผู้ใช้งาน | 6 |
| 2.2.2 ส่วนประมวลผลและจัดการข้อมูล | 8 |
| 2.3 MEAN STACK | 10 |
| 2.3.1 MongoDB | 10 |
| 2.3.2 Express | 11 |
| 2.3.3 Angular | 11 |
| 2.2.2 Node.JS | 11 |

**สารบัญ (ต่อ)**

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| 2.4 แอปพลิเคชันโปรแกรมมิงอินเตอร์เฟส | 12 |
| 2.4.1 REST API | 12 |
| 2.5 การออกแบบฐานข้อมูล | 12 |
| 2.5.1 ความสำคัญของระบบฐานข้อมูล | 14 |
| 2.5.2 การออกแบบฐานข้อมูลเว็บไซต์ | 15 |
| 2.5.3 การออกแบบฐานข้อมูลเชิงสัมพันธ์ | 16 |
| 2.5.4 การประยุกต์ใช้การออกแบบฐานข้อมูลเชิงสัมพันธ์กับฐานข้อมูล  NoSQL | 17 |
| 2.4.5 ประโยชน์ของระบบฐานข้อมูล | 19 |
| บทที่ 3 การออกแบบ การสร้าง และการทำงาน | 21 |
| 3.1 กล่าวนำ | 21 |
| 3.2 การออกแบบและการสร้างระบบจัดการข้อมูลแจ้งเตือนการ  รับประทานยาของผู้ป่วย | 22 |
| 3.2.1 การออกแบบและการสร้างส่วนติดต่อกับผู้ใช้งาน | 23 |
| 3.2.2 การทำงาน | 30 |
| 3.2.3 การออกแบบและการสร้างส่วนประมวลผลและจัดการข้อมูล | 31 |
| 3.3 การออกแบบและการสร้าง API เพื่อติดต่อกับฝั่งแอปพลิเคชัน | 33 |
| 3.3.1 ฟังก์ชัน GET | 34 |
| 3.3.2 ฟังก์ชัน POST | 35 |
| 3.3.3 ฟังก์ชัน PUT | 36 |
| 3.3.4 ฟังก์ชัน DELETE | 37 |
| บทที่ 4 การทดลองและผลการทดลอง | 38 |
| 4.1 กล่าวนำ | 38 |

**สารบัญ (ต่อ)**

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| 4.2 การทดลองการทำงานของระบบจัดการข้อมูล | 38 |
| 4.2.1 การทดลองเข้าสู่ระบบของผู้ดูแล | 38 |
| 4.2.2 การทดลองเพิ่มผู้ใช้งานโดยผู้ดูแลระบบ | 40 |
| 4.2.3 การทดลองแก้ไขข้อมูลผู้ป่วยในระบบ | 41 |
| 4.2.4 การทดลองเรียกดูข้อมูลผู้ป่วยในระบบ | 43 |
| 4.3 การทดลองติดต่อรับส่งข้อมูลกับแอปพลิเคชันโดยใช้ API | 43 |
| 4.3.1 การทดลองใช้ฟังก์ชัน GET POST PUT DELETE | 43 |
| บทที่ 5 บทสรุป | 49 |
| 5.1 สรุป | 49 |
| 5.2 ปัญหาและวิธีการแก้ปัญหา | 50 |
| 5.3 แนวทางการพัฒนา | 51 |
| บรรณานุกรม | 52 |
| ภาคผนวก ก ผังงาน | 54 |
| ภาคผนวก ข แผนผังการใช้งาน | 56 |
| ภาคผนวก ค คูมือการใช้งาน | 59 |
| ประวัติผู้แต่ง | 67 |

**สารบัญตาราง**

|  |  |
| --- | --- |
| **ตารางที่** | **หน้า** |
| 2.1 ตารางเปรียบเทียบระหว่างส่วนต่าง ๆ ของ MySQL และ MongoDB | 11 |
| 4.1 ผลการทดลองการเข้าสู่ระบบของผู้ดูแล | 39 |
| 4.2 ผลการทดลองการเข้าใช้งานเมนูตั้งค่าของผู้ใช้งานทั่วไป | 40 |
| 4.3 ตารางบันทึกผลการเพิ่มข้อมูลโดย Method POST รูปแบบที่ 1 และ 2 | 45 |
| 4.4 ตารางบันทึกผลการเพิ่มข้อมูลโดย Method GET รูปแบบที่ 1 และ 2 | 46 |
| 4.5 ตารางบันทึกผลการอัพเดตข้อมูลโดย Method PUT | 47 |
| 4.6 ตารางบันทึกผลการลบข้อมูลโดย Method DELETE | 48 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**สารบัญรูป**

|  |  |
| --- | --- |
| **รูปที่** | **หน้า** |
| 2.1 JSON FORM | 7 |
| 2.2 ผลการสำรวจฐานข้อมูลเชิงสัมพันธ์จากค่ายต่าง ๆ ที่ได้รับความนิยม  ในปัจจุบัน | 9 |
| 3.1 แผนผังภาพรวมการใช้งานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยา  ของผู้ป่วย | 21 |
| 3.2 แผนผังการรับส่งข้อมูลระหว่างแอปพลิเคชันและเซิร์ฟเวอร์โดยการใช้งานAPI | 22 |
| 3.3 แผนภาพการเข้าสู่ระบบด้วยโดยผู้ดูแล | 23 |
| 3.4 รูปแบบหน้าเว็บเพจการเข้าสู่ระบบ | 24 |
| 3.5 แผนภาพการทำงานของผู้ใช้ระบบในการเข้าใช้งานหน้าเว็บเพจตั้งค่า  ข้อมูลผู้ใช้งาน | 25 |
| 3.6 การออกแบบหน้าเว็บเพจการตั้งค่าผู้ใช้งาน | 25 |
| 3.7 แผนภาพการทำงานของผู้ใช้ระบบในการเข้าใช้งานหน้าเว็บเพจแสดง  ข้อมูลผู้ป่วย | 26 |
| 3.8 การออกแบบหน้าเว็บเพจแสดงข้อมูลต่าง ๆ ของผู้ป่วย | 26 |
| 3.9 แผนภาพการทำงานของหน้าเว็บเพจแก้ไข อัพเดต และลบข้อมูลผู้ป่วย  โดยผู้ดูแลระบบ | 27 |
| 3.10 การออกแบบหน้าเว็บเพจสำหรับการเพิ่ม ลบ แก้ไขผู้ป่วย | 28 |
| 3.11 การออกแบบหน้าเว็บเพจสำหรับติดต่อผู้ดูแลระบบ | 28 |
| 3.12 ตัวอย่างคำสั่งการสร้างปุ่มกดเพื่อบันทึกข้อมูลลงในระบบ | 29 |
| 3.13 ตัวอย่างคำสั่งการสร้างฟิลด์รับข้อมูล | 29 |
| 3.14 โครงสร้างของคอลเลคชันเพื่อใช้เก็บข้อมูลผู้ใช้งานระบบ | 31 |
| 3.15 โครงสร้างของคอลเลคชันเพื่อใช้เป็นโครงสร้างเก็บข้อมูลของผู้ป่วย | 32 |
| 3.16 ผังงานการทำงานของเซิร์ฟเวอร์กับผู้ใช้งานผ่านหน้าเว็บแอปพลิเคชัน | 33 |

**สารบัญรูป(ต่อ)**

|  |  |
| --- | --- |
| **รูปที่** | **หน้า** |
| 3.17 ตัวอย่างโครงสร้างสำหรับรับส่งข้อมูลของ API ระหว่างแอปพลิเคชันและ  ฐานข้อมูล | 34 |
| 3.18 ตัวอย่างโค้ดฟังก์ชัน GET ของ API | 34 |
| 3.19 ตัวอย่างโค้ดฟังก์ชัน POST ของ API | 35 |
| 3.20 ตัวอย่างโค้ดการเรียกใช้ฟังก์ชัน PUT ของ API | 36 |
| 3.21 ตัวอย่างโค้ดการเรียกใช้ฟังก์ชัน DELETE ของ API | 37 |
| 4.1 การแจ้งเตือนเมื่อใส่รหัสผ่านผิด | 39 |
| 4.2 ข้อมูลในฐานข้อมูลผู้ป่วยก่อนการแก้ไข | 42 |
| 4.3 ข้อมูลในฐานข้อมูลผู้ป่วยหลังการแก้ไข | 42 |
| 4.4 ตัวอย่าง JSON FORM ของ Method POST รูปแบบที่ 1 | 44 |
| 4.5 ตัวอย่าง JSON FORM ของ Method POST รูปแบบที่ 2 | 45 |
| 4.6 ตัวอย่างข้อมูลที่ถูกดึงจากฐานข้อมูลโดยการใช้ Method GET | 46 |
| 4.7 ตัวอย่างของข้อมูลในรูป JSON FORM | 47 |
| ก.1 ผังงานการทำงานของเซิร์ฟเวอร์กับผู้ใช้งานผ่านหน้าเว็บแอปพลิเคชัน | 55 |
| ข.1 แผนผังภาพรวมการใช้งานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยา  ของผู้ป่วย | 57 |
| ข.2 แผนผังการรับส่งข้อมูลระหว่างแอปพลิเคชันและเซิร์ฟเวอร์โดยการใช้งานAPI | 58 |
| ค.1 ฟิลด์สำหรับกรอกข้อมูล login | 61 |
| ค.2 หน้า HOME | 62 |
| ค.3 การกด save ข้อมูลผู้ป่วย | 62 |
| ค.4 หน้า Contact | 63 |
| ค.5 การเข้าหน้าตั้งค่าผู้ใช้งาน | 63 |
| ค.6 การกรอกข้อมูลโดยใช้ชื่อและรหัสผ่านขงผู้ดูแลระบบ | 64 |
| ค.7 เปรียบเทียบการเข้าใช้งานเมนูการตั้งค่าของผู้ดูแลและผู้ใช้ระบบทั่วไป | 65 |
|  |  |

**สารบัญ (ต่อ)**

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| ค.8 การเรียกใช้งาน Method GET | 66 |

**บทที่ 1**

**บทนำ**

**1.1 ความเป็นมาและความสำคัญ**

จากการสำรวจประชากรผู้สูงอายุในประเทศไทยของมูลนิธิสถาบันวิจัยและพัฒนาผู้สูงอายุไทยพบว่า การเพิ่มขึ้นของจำนวนประชากรผู้สูงอายุในปัจจุบันส่งผลให้มีปัญหาด้านสุขภาพเพิ่มขึ้น ทำให้ยาเป็นองค์ประกอบสำคัญของผู้สูงอายุ โดยผู้สูงอายุบางรายรับประทานยาที่แพทย์สั่งไม่สม่ำเสมอ รับประทานยาไม่ครบหรือรับประทานเกินขนาด บางรายได้รับการรักษาที่ไม่ต่อเนื่องซึ่งเกิดจากการขาดนัดหรือรับประทานยาไม่ถูกต้องทำให้การรักษาเป็นไปได้ช้าหรือยากขึ้นกว่าเดิม บางรายมีการกินยาซ้ำอันเนื่องมาจากอาการหลงลืมหรือคิดช้าลง ทำให้เสี่ยงต่อการเกิดภาวะแทรกซ้อนที่รุนแรงและอาจส่งผลให้เกิดการเสียชีวิตได้

จากปัญหาดังกล่าวจะเห็นว่า การรับประทานยาให้ตรงตามที่แพทย์สั่งมีความสำคัญมากสำหรับผู้ป่วย ดังนั้น ผู้จัดทำจึงคิดทำระบบการจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยเพื่อช่วยให้สามารถบริหารจัดการและเก็บข้อมูลการรับประทานยาของผู้ป่วยได้อย่างเป็นระบบผ่านทางหน้าเว็บแอปพลิเคชัน เพื่อสนับสนุนแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยให้สามารถทำงานได้อย่างมีประสิทธิภาพมากขึ้น

**1.2 จุดมุ่งหมายของโครงงาน**

คณะผู้จัดทำได้สร้างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยขึ้นมา เพื่อใช้จัดการและสนับสนุนการทำงานของแอปพลิเคชั่นแจ้งเตือนการรับประทานยาของผู้ป่วย ซึ่งจะเป็นในลักษณะของเว็บแอปพลิเคชันโดยมีจุดมุ่งหมายคือ

1. เพื่อศึกษาการใช้งาน MEAN Stack ในการสร้าง Server ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. เพื่อออกแบบการทำงานในส่วนต่าง ๆ ของระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

3. เพื่อสร้างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

4. เพื่อทดลองการใช้งานและสามารถนำระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยไปใช้งานกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยได้

**1.3 สมมติฐานของการจัดทำโครงงาน**

การพัฒนาระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยในรูปแบบของเว็บแอปพลิเคชัน ทำให้สามารถใช้งาน MEAN Stack ในการสร้าง Server สามารถออกแบบการทำงานในส่วนต่าง ๆ ของระบบจัดการข้อมูล สามารถสร้างระบบจัดการข้อมูลการแจ้งเตือนการรับประทานยาของผู้ป่วยและนำไปใช้งานกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยได้

**1.4 ขีดความสามารถของโครงงาน**

โครงงานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยมีขีดความสามารถดังนี้

1. สามารถจำแนกผู้ใช้งานได้ คือ

- ผู้ดูแลระบบ (Admin)

- ผู้ใช้ทั่วไป (User)

2. สามารถดูข้อมูลประวัติของผู้ป่วย

3. สามารถตรวจสอบการรับประทานยาของผู้ป่วย

4. สามารถแก้ไข อัพเดต ลบข้อมูลในการรับประทานยาของผู้ป่วย

5. สามารถจำแนกชนิดของยาที่ผู้ป่วยใช้

6. สามารถรับส่งข้อมูลกับแอปพลิเคชันแจ้งเตือนการรับประทานยาผู้ป่วย

**1.5 ขั้นตอนการทำโครงงาน**

โครงงานนี้เป็นโครงงานประเภทซอฟต์แวร์ ประกอบด้วย 2 ส่วน คือ ส่วนติดต่อกับผู้ใช้งาน (Front-end) และส่วนประมวลผลและจัดการข้อมูล (Back-end) โดยขั้นตอนการทำโครงงานจะเริ่มจาก การศึกษาข้อมูลเกี่ยวกับการสร้างเว็บแอปพลิเคชันโดยใช้ MEAN STACK จากนั้นทำการติดตั้ง โดยจะใช้ภาษา HTML CSS และ AngularJS ในการสร้างส่วนติดต่อกับผู้ใช้งาน ใช้ Node.js ในการสร้างเซิร์ฟเวอร์และมี MongoDB เป็นฐานข้อมูลเพื่อเก็บข้อมูลต่าง ๆ ของผู้ป่วย เมื่อศึกษาจนได้ข้อมูลพอสมควรจึงทำการออกแบบและทดลองสร้างส่วนติดต่อกับผู้ใช้และส่วนประมวลผลและจัดการข้อมูลเพื่อให้สามารถ เพิ่ม ลบ แก้ไข ข้อมูลต่าง ๆได้ และทำการพัฒนาให้สามารถติดต่อกับแอปพลิเคชันแจ้งเตือนการรับประทานของผู้ป่วยโดยการใช้ REST API เพื่อให้สามารถรับส่งข้อมูลไปใช้โดยวิธีการเพิ่ม ลบ แก้ไข อัพเดต ซึ่งเป็นบริการพื้นฐานของ API ดังกล่าวลงไปในฐานข้อมูลได้

**1.6 เนื้อหาโดยสังเขป**

เพื่อความเข้าใจในการศึกษาในหัวข้อต่าง ๆ เนื้อหาในปริญญานิพนธ์ฉบับนี้จึงได้ทำการแบ่งส่วนออกเป็นแต่ละบทดังนี้

บทที่ 1 ความเป็นมาและความสำคัญของโครงงาน จุดประสงค์ของโครงงาน สมมติฐานของโครงงาน ขีดความสามารถของโครงงาน ขั้นตอนการทำโครงงาน เนื้อหาโดยสังเขป

บทที่ 2 กล่าวถึงทฤษฎีและหลักการต่าง ๆ ที่เกี่ยวข้องประกอบไปด้วย เว็บแอปพลิเคชัน ชนิดของฐานข้อมูล MAEN STACK หลักการออกแบบฐานข้อมูลและการนำมาประยุกต์ใช้ในฐานข้อมูลแบบ NoSQL และแอปพลิเคชันโปรแกรมมิงอินเตอร์เฟส (API)

บทที่ 3 กล่าวถึงการออกแบบและการทำงานของเว็บแอปพลิเคชันในส่วนของ ส่วนติดต่อกับผู้ใช้งาน (Front-end) ส่วนประมวลผลและจัดการข้อมูล (Back-end) และการพัฒนาให้สามารถติดต่อรับส่งข้อมูลกับแอปพลิเคชันแจ้งเตือนการรับประทานของผู้ป่วย

บทที่ 4 กล่าวถึงการทดลองและผลของการทดลองการใช้งานระบบจัดการข้อมูลการแจ้งเตือนรับประทานยาของผู้ป่วย การทดลองและผลการทดลองการติดต่อรับส่งข้อมูลกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยด้วย API

บทที่ 5 กล่าวถึงสรุปผลเกี่ยวกับการจัดทำโครงงาน ปัญหาที่เกิดขึ้น แนวทางแก้ไขและแนวทางการพัฒนา

**บทที่ 2**

**ทฤษฎีและหลักการ**

**2.1 กล่าวนำ**

เนื้อหาของปริญญานิพนธ์ในบทนี้เป็นทฤษฎีและหลักการที่จะนำมาใช้เพื่อประกอบการสร้าง เว็บแอปพลิเคชันโดยมีรายละเอียดคือ เว็บแอปพลิเคชัน ชนิดของฐานข้อมูล MAEN STACK หลักการออกแบบฐานข้อมูลและการนำมาประยุกต์ใช้ในฐานข้อมูลแบบ NoSQL และแอปพลิเคชันโปรแกรมมิงอินเตอร์เฟส (API)

**2.2 เว็บแอปพลิเคชัน**

เว็บแอปพลิเคชันคือ การพัฒนาระบบงานบนเว็บเพื่อใช้สำหรับการใช้งานเว็บเพจต่าง ๆ ซึ่งถูกปรับแต่งให้แสดงผลแต่ส่วนที่จำเป็น เพื่อช่วยลดการใช้ทรัพยากรในการประมวลผลของตัวเครื่อง ทำให้โหลดหน้าเว็บไซต์ได้เร็วขึ้น ผู้ใช้งานสามารถใช้งานผ่านอินเทอร์เน็ตในความเร็วตํ่าได้ ในการรับและแสดงผลข้อมูล ประมวลผลข้อมูล จัดการกับข้อมูลในฐานข้อมูล โดยการพัฒนาเว็บแอปพลิเคชันตาม Model-View-Controller (MVC) จะสามารถแบ่งเว็บแอปพลิเคชันออกได้เป็นสามส่วนหลัก ๆ ดังนี้

1. ส่วนที่ติดต่อกับผู้ใช้งานเพื่อรับข้อมูลและแสดงผล (View)

2. ส่วนที่ประมวลผลการทำงาน (Controller)

3. ส่วนที่ใช้ในการติดต่อและจัดการกับข้อมูลและฐานข้อมูล (Model)

ในการพัฒนาเว็บแอปพลิเคชันนั้นสามารถพัฒนาด้วยภาษาที่หลากหลายขึ้นตามความต้องการของผู้พัฒนาซึ่งสามารถแบ่งได้ออกเป็นสองส่วนคือ ส่วนติดต่อกับผู้ใช้งาน (Front-End Technology) และส่วนประมวลผลและจัดการข้อมูล (Back-End Technology)

**2.2.1 ส่วนติดต่อกับผู้ใช้งาน**

ส่วนติดต่อกับผู้ใช้งาน (Front-End Technology) คือส่วนที่ใช้ในการติดต่อกับระบบและฐานข้อมูลผ่านทางหน้าเว็บแอปพลิเคชัน โดยการพัฒนาด้วยภาษาคอมพิวเตอร์ที่มีความหลากหลายขึ้นอยู่กับความถนัดและความต้องการของผู้ใช้ซึ่งในปัจจุบันคือภาษา HTML CSS และ JavaScript โดยความหลากหลายของส่วนติดต่อกับผู้ใช้งาน (Front-End Web Technology) จะถูกจำกัดด้วยมาตรฐานกลางของ W3C (World Wide Web Consortium) เพื่อให้ผู้พัฒนาส่วนติดต่อกับผู้ใช้งานสามารถแสดงผลข้อมูลในรูปแบบเดียวกัน เป็นไปในทิศทางที่อยู่ภายใต้มาตาราฐานเดียวกันเพื่อให้เกิดความสะดวกแบะความเข้าใจแก่ผู้พัฒนาและผู้ใช้งาน

**2.2.1.1 HTML**

HTML เป็นภาษาที่ใช้เขียนเพื่อเป็นโครงสร้างสำหรับแสดงรายละเอียดของหน้าตาเว็บไซต์โดยการใช้ <tag> คำสั่งกำกับเพื่อกำหนดรายละเอียดที่ต้องการให้แสดงผลในส่วนต่าง ๆ ของหน้าเว็บเพจซึ่ง

**2.2.1.2 CSS**

เป็นภาษาที่ถูกพัฒนาเพื่อใช้เขียนเป็นส่วนของการตกแต่งหน้าตาเว็บไซต์ที่แสดงให้ผู้ใช้ได้เห็นและใช้ในการจัดรูปแบบและโครงสร้างของเอกสารที่เขียนจากภาษา HTML ซึ่งมีการทำงานควบคู่กัน โดยภาษา CSS นั้นสามารถใช้งานได้หลากหลายและมีความยืดหยุ่นสามารถใช้งานกับภาษา XML SVG และ XUL

**2.2.1.3 JavaScript**

เป็นภาษาสคริปต์เชิงวัตถุที่ใช้ในการสร้างและพัฒนาเว็บไซต์ โดยทำงานร่วมกับภาษา HTML เพื่อให้เว็บไซต์ตอบสนองผู้ใช้งานได้ยิ่งขึ้น ในการออกแบบและพัฒนาร่วมกับภาษา HTML และภาษา JAVA สามารถทำงานข้ามแพลตฟอร์มได้ทั้งฝั่งติดต่อกับผู้ใช้งาน (Front-End Technology) และส่วนประมวลผลและจัดการข้อมูล (Back-End Technology)

**2.2.1.4 UI**

UI (User Interface) หมายถึง ส่วนติดต่อระหว่างผู้ใช้กับระบบ เป็นส่วนที่ใช้ในการแสดงข้อมูลผ่านทางหน้าจอแสดงผลโดยจะเกิดการตอบโต้เมื่อผู้ใช้งานกระทำการป้อนหรือสั่งข้อมูลลงไปในระบบหรือทำการติดต่อต่าง ๆ ผ่านทางระบบ การออกแบบ UI ที่ดี จะช่วยทำให้เว็บไซต์มีความน่าสนใจ ดังนั้นนักออกแบบ UI ต้องมีความสามารถในการนำไอเดียใหม่ ๆมาดึงความสนใจของผู้ใช้งานให้มาใช้งานระบบของตนได้มากยิ่งขึ้น

**2.2.1.5 UX**

UX (User Experience) หมายถึง ประสบการณ์ใช้งานของผู้ใช้ในการใช้งานระบบ เป็นประสบการณ์จากการใช้งานของระบบต่าง ๆ ว่ามีความยากง่ายต่อการเข้าถึงข้อมูลหรือบริการต่าง ๆ ของผู้ให้บริการว่ามีความน่าสนใจเพียงใด ซึ่งการออกแบบ UI ให้สวยงามและใช้งานง่ายจะมีผลกับประสบการณ์การใช้งานด้วยเพราะผู้ออกแบบต้องออกแบบให้ใช้งานง่ายและเข้าใจง่าย มีความสะดวกในการเข้าถึงและเพื่อเป็นการทำให้ผู้ใช้ต้องการที่จะกลับมาใช้งานระบบของผู้ให้บริการนั้น ๆ ในครั้งต่อไป

**2.2.1.6 JSON**

JSON ย่อมากจากคำว่า JavaScript Object Notation เป็นรูปแบบสำหรับการติดต่อเพื่อใช้รับส่งข้อมูลระหว่างส่วนติดต่อกับผู้ใช้งานและส่วนประมวลผลและจัดการข้อมูลให้สามารถทำได้ง่ายขึ้น ผู้ใช้สามารถทำความเข้าใจและสร้างได้ง่าย นิยมใช้งานในเว็บแอปพลิเคชันซึ่งรูปแบบการจัดเก็บข้อมูล JSON ประกอบด้วยคีย์เป็นชื่อหัวข้อของข้อมูลภายในมีค่าของคีย์ซึ่งเป็นข้อมูลต่าง ๆ ที่สอดคล้องกับคีย์นั้น โดยข้อมูลจะอยู่ในลักษณะของอ็อบเจกต์มีรูปแบบการเขียนดังนี้



**รูปที่ 2.1** JSON FORM

**2.2.2 ส่วนประมวลผลและจัดการข้อมูล**

ส่วนประมวลผลและจัดการข้อมูล (Back-End Web Technology) คือส่วนเทคโนโลยีที่ใช้เป็นฐานข้อมูลและระบบสำหรับเว็บแอปพลิเคชันเพื่อใช้ในการประมวลข้อมูลในการทำงานของเว็บแอปพลิเคชันในการบริหารจัดการด้านต่าง ๆ ตั้งแต่ การยันยันตัวตน การตรวจสอบสิทธิ์การเข้าใช้ การเรียกดูข้อมูลในฐานมูล การจัดเก็บข้อมูลในฐานข้อมูล ตลอดจนการลบข้อมูลในฐานข้อมูล โดยจะทำงานเมื่อเว็บ แอปพลิเคชันได้รับ HTTP request จากผู้ใช้งาน ส่วนประมวลผลและจัดการข้อมูลจะทำการประมวลผลและส่งข้อมูลโดยการ response กลับไปให้ผู้ใช้

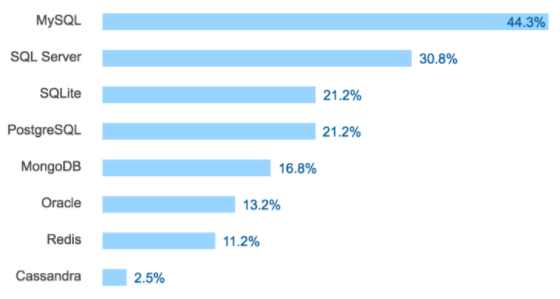
เทคโนโลยีที่ใช้ในการพัฒนาส่วนประมวลผลและจัดการข้อมูลของเว็บแอปพลิเคชันจะมีความหลากหลายกว่าส่วนติดต่อกับผู้ใช้งาน เนื่องจากไม่มีข้อจำกัดด้านมาตรฐานของ W3C เหมือนกับส่วนติดต่อกับผู้ใช้งาน ทำให้สามารถใช้ภาษาในการพัฒนาได้หลากหลายมากกว่าเช่น PHP Python Java โดยแต่ละภาษาจะมีการเก็บข้อมูลที่ต่างกันตามฐานข้อมูล ซึ่งในปัจจุบันสามารถเลือกใช้ได้สองประเภท คือ SQL เช่น MySQL Oracle และ NoSQL เช่น Cassandra MongoDB

**2.2.2.1 SQL**

SQL (Structured Query Language) เป็นภาษาที่ออกแบบมาเพื่อใช้จัดการข้อมูลในลักษณะของ [relational database management system (RDBMS)](https://saixiii.com/what-is-database/) ในการค้นหาข้อมูล เปลี่ยนแปลง เพิ่ม ลด แก้ไข โดยข้อมูลที่ถูกเก็บอยู่ในฐานข้อมูลในรูปแบบตารางมีลักษณะเป็น column และ row ด้วยความสามารถของ SQL ทำให้สามารถสร้างตารางขึ้นมาสามารถลบและเปลี่ยนแปลงค่าของต่าง ๆ ในตารางได้ ตัวอย่างเช่น MySQL Oracle Microsoft SQL

**1. MySQL**

เป็นระบบจัดการฐานข้อมูลที่เป็นฟรีแวร์ พัฒนาโดยบริษัท MySQL AB สามารถรองรับจำนวนผู้ใช้ได้เป็นจำนวนมาก เป็นที่นิยมใช้กันในปัจจุบัน และมีแนวโน้มสูงขึ้นเรื่อย ๆ รองรับคำสั่ง SQL เป็นระบบจัดการฐานข้อมูลที่ถูกออกแบบให้สามารถทำงานได้บนระบบปฏิบัติการที่หลากหลายระบบปฏิบัติการ เช่น UNIX MAC OS Windows ใช้งานร่วมกับภาษาที่ใช้เป็นแพลตฟอร์มเพื่อการออกแบบเว็บไซต์ในลักษณะต่าง ๆ ได้ เช่น C C++ PHP



**รูปภาพที่ 2.2** ผลการสำรวจฐานข้อมูลเชิงสัมพันธ์จากค่ายต่าง ๆ ที่ได้รับความนิยมในปัจุบัน

ที่มา : <https://www.eversql.com/most-popular-databases-in-2017-according-to-stackoverflow-survey/>

**2. Oracle**

Oracle เป็นโปรแกรมจัดการฐานข้อมูลที่ผลิตโดยบริษัทออราเคิล ใช้จัดการฐานข้อมูลในลักษณะที่มีความสัมพันธ์ซึ่งจะทำหน้าที่เป็นตัวกลางคอยติดต่อและประสานงานระหว่างผู้ใช้กับฐานข้อมูล เพื่อทำให้ผู้ใช้งานสามารถใช้งานฐานข้อมูลได้สะดวกและรวดเร็วยิ่งขึ้นเช่น การค้นหาข้อมูลต่าง ๆ ภายในฐานข้อมูลที่ง่ายและสะดวก ผู้ใช้ไม่จำเป็นต้องทราบถึงโครงสร้างภายในของฐานข้อมูลก็สามารถเข้าใช้ฐานข้อมูลที่มีอยู่ได้

**2.2.2.2 NoSQL**

NoSQL หรือ Not Only SQL คือระบบฐานข้อมูลสำหรับงานที่ต้องรองรับข้อมูลขนาดใหญ่และสามารถทำการขยายระบบได้ง่าย รองรับรูปแบบข้อมูลที่หลากหลายและรองรับการใช้งานพร้อมกันของผู้ใช้งานได้ดี มีการขยายตัวในแนวนอน การเก็บข้อมูลต่างจากการเก็บข้อมูลแบบเชิงสัมพันธ์ของ SQL หรือ Relational Databases ทำให้ข้อมูลไม่มีโครงสร้างที่ชัดเจนเหมาะสำหรับงานที่ต้องทำแบบ realtime หรือใช้สำหรับการวิเคราะห์ข้อมูลแบบ offline โดย NoSQL แบ่งตามการใช้งานได้ 4 รูปแบบคือ แบบ Key-Value stores เช่น Couchbase แบบ Document stores เช่น MongoDB แบบ Column stores เช่น Cassandra และแบบ Graph stores เช่น Neo4J โดยผู้ใช้งานสามารถเลือกใช้ตามความชอบและความเหมาะสมกับการใช้งาน

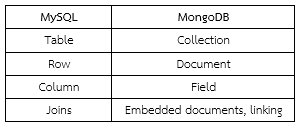
**2.3 MEAN STACK**

MEAN STACK คือ เฟรมเวิร์ค (Framework) ที่ถูกสร้างขึ้นมาเพื่อใช้พัฒนาเว็บไซต์โดยเน้นการใช้ภาษา JavaScript ในการพัฒนา ซึ่งตัวอักษร MEAN สามารถแยกเป็นองค์ประกอบหลักๆที่ใช้ในการพัฒนาเว็บไซต์ได้ดังนี้

**2.3.1 MongoDB**

MongoDB เป็นฐานข้อมูลแบบ NoSQL แบบหนึ่งที่ไม่มีความสัมพันธ์ของตารางแบบฐานข้อมูลแบบ SQL ทั่ว ๆ ไป แต่จะเก็บข้อมูลในลักษณะของรูปแบบ JSON (JavaScript Object Notation) แทนการบันทึกข้อมูลทุก ๆ record โดยการเก็บข้อมูลต่าง ๆ ใน MongoDB จะถูกเก็บไว้ใน Collections ซึ่งเรียกว่า Document ค่าที่เก็บจะเป็น key และ value สามารถเปรียบเทียบได้กับ Table ใน Relational Database ทั่ว ๆ ไปแต่ต่างกันที่ collection ไม่จำเป็นต้องมี schema เหมือนกันก็สามารถบันทึกข้อมูลได้

**ตารางที่ 2.1** ตารางเปรียบเทียบระหว่างส่วนต่าง ๆ ของ MySQL และ MongoDB



**2.3.2 Express**

Express คือ Framework ที่รันอยู่บน Node.JS เป็นส่วนที่ใช้เขียนการควบคุมในส่วนของ Web Server ด้วย JavaScript ได้รับความนิยมมากในปัจจุบัน ใช้งานง่ายไม่ซับซ้อน มีหน้าที่จัดการ Route ต่าง ๆ ให้กับเว็บไซต์สามารถติดตั้งโปรแกรมหรือซอฟต์แวร์เสริมผ่าน Terminal โดยการใช้คำสั่ง npm ทำให้มีความคล่องตัวสูงเหมาะสำหรับผู้เริ่มต้นที่จะเขียนเว็บด้วย JavaScript

**2.3.3 Angular**

Angular คือ framework ในฝั่งของ Front-end ถูกสร้างและพัฒนาโดย Google ซึ่งได้รับความนิยมสูงมากในปัจจุบัน โดย Angular จะมีลักษณะที่เป็น Single Page Application (SPA) หรือเรียกว่าเป็น Web Application ที่มีหน้าเดียวซึ่ง client จะติดต่อกับ server ด้วยการเรียก AJAX ไปที่ API ของ server

**2.3.4 Node.JS**

Node.JS เป็น Cross Platform Runtime Environment คือแพลตฟอร์มที่ เมื่อเปิดในหน้าไหนก็แสดงผลเหมือนกันเป็นเทคโนโลยีที่สร้างขึ้นเพื่อใช้เป็น Web Server เขียนด้วยภาษา JavaScript เพื่อใช้ในการจัดการ Request และ Response จากผู้ใช้และเป็น Interpreter หรือตัวที่ใช้ติดต่อกับฐานข้อมูลเพื่อให้สามารถรัน JavaScript นอก Internet Browser ได้ ลักษณะการทำงานของ Node.JS จะทำในลักษณะที่เรียกว่า “Non-Blocking I/O” หรือ “Asynchronous” ทำให้เซิร์ฟเวอร์สามารถให้บริการผู้ใช้ได้จำนวนมากพร้อม ๆ กัน โดยการใช้งานจะถูกเรียกใช้ผ่าน Terminal ซึ่งผู้ใช้จำเป็นต้องมีความรู้ในการเรียกใช้คำสั่งต่าง ๆ ในเบื้องต้น

**2.4 แอปพลิเคชันโปรแกรมมิงอินเตอร์เฟส**

เรียกสั้น ๆว่า API (Application Programming Interface) เป็นช่องทางในการเชื่อมต่อระหว่างเว็บไซต์หนึ่งไปยังอีกเว็บไซต์หนึ่งหรือเป็นการเชื่อมต่อระหว่างผู้ใช้งานกับ Server หรือจาก Server เชื่อมต่อไปหา Server ซึ่ง API เปรียบเสมือนเป็นสื่อกลางในลักษณะของภาษาคอมพิวเตอร์ที่ทำให้คอมพิวเตอร์สามารถสื่อสารและแลกเปลี่ยนข้อมูลกันได้อย่างอิสระ

**2.4.1 REST API**

REST หรือ Representational State Transfer เป็นบริการของเว็บไซต์รูปแบบหนึ่งโดยอาศัย HTTP Method ในการ GET POST PUT และ DELETE ในการทำงาน และส่งผลกลับมาในรูปแบบของ JSON หรือ XML ทำให้สามารถรับส่งข้อมูลไปมาข้าม Platform ได้อย่างสะดวกเพราะเป็นการเรียกผ่าน HTTP Protocol ที่ใช้ในการเรียกเว็บไซต์ และยังเป็นที่นิยมเพราะมีการใช้ Traffic ที่น้อยเนื่องจากการส่งค่าไปและกลับของ REST จะส่งกลับมาในรูปแบบของ JSON หรือ XML ซึ่งมีขนาดเล็ก และสามารถ Extract ออกมาใช้งานได้ง่าย

**2.5 การออกแบบฐานข้อมูล**

ในการออกแบบฐานข้อมูลต่าง ๆ ผู้พัฒนาต้องมีการออกแบบให้สามารถเข้าใจโครงสร้าง ภายในฐานข้อมูลได้ง่าย สามารถที่จะนำข้อมูลต่าง ๆ ในฐานข้อมูลมาใช้กับเว็บไซต์ได้รวดเร็วซึ่งจะช่วยให้การค้นหา การดึงและแสดงผลข้อมูลที่มีการเปลี่ยนแปลงบ่อยทำได้อย่างมีประสิทธิภาพ

ระบบฐานข้อมูล หมายถึง ส่วนประกอบต่าง ๆ ที่เกี่ยวข้องกับการใช้งานฐานข้อมูลโดยสามารถแบ่งเป็น 3 ส่วนได้แก่

1. ส่วนฮาร์ดแวร์ (Hardware) คือส่วนที่ประกอบด้วยเครื่องคอมพิวเตอร์และอุปกรณ์ต่าง ๆ

ในการเก็บข้อมูลที่เกี่ยวข้องกับฐานข้อมูลหรืออุปกรณ์นำเข้า ในระบบฐานข้อมูลที่มีประสิทธิภาพ ควรมีฮาร์ดแวร์ต่าง ๆ ที่ช่วยเพิ่มความสะดวกในการบริหารระบบงานฐานข้อมูลแก่ผู้พัฒนาเช่น  ขนาดของหน่วยความจำ  ความเร็วของหน่วยประมวลผลกลาง  อุปกรณ์นำเข้าและแสดงผลของข้อมูล รวมถึงหน่วยความจำสำรองที่รองรับการประมวลผลข้อมูลในระบบได้อย่างมีประสิทธิภาพ

2. ส่วนซอฟต์แวร์ (Software) คือโปรแกรมต่าง ๆ ที่ใช้ในการจัดการฐานข้อมูลและระบบการจัดการฐานข้อมูล มีหน้าที่ในการสร้างและเรียกใช้ข้อมูล การจัดทำรายงาน การปรับเปลี่ยนแก้ไขโครงสร้าง ควบคุมการทำงานต่าง ๆ ภายในระบบ ในการประมวลผล ฐานข้อมูลอาจใช้โปรแกรมที่แตกต่างกันขึ้นอยู่กับระบบคอมพิวเตอร์ที่ใช้ว่าเป็นแบบใด

ฐานข้อมูล คือโปรแกรมหรือซอฟท์แวร์ที่ทำหน้าที่ในการจัดการฐานข้อมูลโดยจะเป็นสื่อกลางระหว่างผู้ใช้และโปรแกรมประยุกต์ต่าง ๆ ที่มีอยู่ในฐานข้อมูล

3. ส่วนบุคคลากร คือบุคลากรที่เกี่ยวข้องกับระบบฐานข้อมูลหรือผู้ใช้งานฐานข้อมูล สามารถแบ่งได้เป็น 5 ประเภทดังนี้

3.1. ผู้ใช้ทั่วไป คือบุคลากรที่ใช้ข้อมูลจากระบบฐานข้อมูลเพื่อใช้ในงานต่าง ๆ

3.2. พนักงานปฏิบัติงาน คือผู้ปฏิบัติการด้านการประมวลผล การป้อนข้อมูลต่าง ๆ ลงในเครื่องคอมพิวเตอร์

3.3. นักวิเคราะห์และออกแบบระบบ คือบุคลากรที่ทำหน้าที่วิเคราะห์ระบบฐานข้อมูลและทำหน้าที่ออกแบบระบบงานเพื่อนำไปใช้

3.4. ผู้เขียนโปรแกรมประยุกต์ใช้งาน คือผู้ทำหน้าเขียนโปรแกรมประยุกต์ในการจัดเก็บและการเรียกใช้ข้อมูลให้ตามความต้องการของผู้ใช้ในงานในด้านต่าง ๆ

3.5. ผู้บริหารงานฐานข้อมูล หรือเรียกสั้นๆว่า Admin คือผู้ที่ทำหน้าที่ควบคุมและ บริหารงานต่าง ๆ ในระบบฐานข้อมูลทั้งหมด  เป็นผู้ที่จะต้องตัดสินใจในการรวบรวมข้อมูลต่าง ๆ ตั้งแต่การรวบรวมข้อมูลมาใส่ในฐานข้อมูล วิธีในการจัดเก็บข้อมูล เทคนิคการเรียกใช้ข้อมูล การกำหนดระบบการรักษาความปลอดภัยของข้อมูล การสร้างระบบข้อมูลสำรอง การกู้คืนข้อมูลต่าง ๆ ในกรณีที่เกิดความเสียหายหรือสูญหายของข้อมูลตลอดจนการประสานงานกับผู้ใช้ว่าต้องการใช้ข้อมูลอย่างไร รวมถึงนักวิเคราะห์และออกแบบระบบและโปรแกรมเมอร์  ประยุกต์ใช้งานฐานข้อมูลเพื่อให้การบริหารการใช้งานเป็นไปอย่างมีประสิทธิภาพ

การออกแบบฐานข้อมูลที่ดีนั้นสามารถทำได้โดยอาศัยหลักการหรือแนวทางสำหรับการออกแบบฐานข้อมูล ซึ่งถือเป็นสิ่งสำคัญที่จะช่วยให้ผู้พัฒนาสามารถวางโครงสร้างของฐานข้อมูลได้อย่างมีประสิทธิภาพ หลักการแรกคือการออกแบบเพื่อช่วยลดการซ้ำซ้อนของข้อมูล ทำให้ประหยัดพื้นที่ ช่วยลดความผิดพลาดจากความไม่สอดคล้องกันของข้อมูล หลักการที่สองคือ ความถูกต้องและความสมบูรณ์ของข้อมูล เพื่อให้การเรียกใช้ข้อมูลต่าง ๆ ในฐานข้อมูลเป็นไปอย่างถูกต้องตามข้อมูลที่มีการบันทึกมีความครบถ้วนและเป็นข้อมูลที่ผู้ใช้ต้องการจากฐานข้อมูลนั้น

**2.5.1 ความสำคัญของระบบฐานข้อมูล**

ในการจัดเก็บข้อมูลต่าง ๆ ให้อยู่ในระบบฐานข้อมูลจะทำให้สามารถนำข้อมูลเหล่านั้นไปดำเนินการต่าง ๆ ได้ง่ายเพราะช่วยให้ผู้ใช้งานรวมถึงผู้พัฒนาฐานข้อมูลเรียกใช้ได้ง่าย สะดวก และเกิดความรวดเร็วซึ่งระบบฐานข้อมูลจะช่วยลดปัญหาต่าง ๆ ที่จะเกิดกับข้อมูลในกรณีต่าง ๆ ดังนี้

1. ช่วยลดปัญหาการเก็บข้อมูลที่ซ้ำซ้อน

2. ช่วยในการรักษาความถูกต้องของข้อมูล

3. ทำให้การป้องกันและรักษาความปลอดภัยให้กับข้อมูลทำได้อย่างสะดวก

4. สามารถใช้ข้อมูลร่วมกันได้

5. มีความเป็นอิสระของข้อมูล

6.สามารถขยายงานได้ง่าย

7.ทำให้ข้อมูลบูรณะกลับสู่สภาพปกติได้เร็วและมีมาตรฐาน

**2.5.2 การออกแบบฐานข้อมูลเว็บไซต์**

ระบบฐานข้อมูลที่ดีและมีประสิทธิภาพในการเรียกใช้ การเขียนและการอ่านข้อมูลต่าง ๆ ในฐานข้อมูล ผู้พัฒนาต้องมีการออกแบบให้สามารถเข้าใจโครงสร้างต่าง ๆ ภายในฐานข้อมูลได้ง่าย สามารถที่จะเรียกใช้ข้อมูลต่าง ๆ ในฐานข้อมูลมาใช้จัดการต่าง ๆ กับเว็บไซต์ได้รวดเร็วซึ่งจะช่วยให้การค้นหา การดึงและแสดงผลข้อมูลที่มีการเปลี่ยนแปลงบ่อยโดยฐานข้อมูลต้องมีประสิทธิภาพ ระบบฐานข้อมูลที่สามารถออกแบบและรองรับการใช้งานในลักษณะนี้จะอยู่ในรูปแบบของฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System) ซึ่งได้รับความนิยมอย่างมากในปัจจุบัน

**2.5.2.1 แนวคิดฐานข้อมูลเชิงสัมพันธ์**

ฐานข้อมูลเชิงสัมพันธ์ เป็นฐานข้อมูลที่ได้รับความนิยมและมีการเลือกใช้งานมากที่สุดซึ่งการที่จะสามารถเข้าใจและใช้งานฐานข้อมูลเชิงสัมพันธ์ได้อย่างมีประสิทธิภาพ ผู้ใช้จำเป็นต้องมีพื้นฐานในทางทฤษฎีต่าง ๆ ที่เกี่ยวข้องในการเรียกใช้ การสร้าง การเชื่อมโยงความสัมพันธ์ต่าง ๆ บางประการของฐานข้อมูลพอสมควร

ข้อมูลต่าง ๆ ภายในฐานข้อมูลเชิงสัมพันธ์สามารถจัดเก็บได้หลายรูปแบบขึ้นอยู่กับความต้องการและการออกแบบของผู้พัฒนาโดยข้อมูลแต่ละข้อมูลจะมีความสัมพันธ์ซึ่งกันและกัน โดยในฐานข้อมูลเชิงสัมพันธ์จะแบ่งความสัมพันธ์ของการเก็บข้อมูลออกเป็น 3 ชนิดดังนี้

1. ความสัมพันธ์แบบ หนึ่งต่อหนึ่ง (one-to-one) หมายถึง มี 1 ข้อมูลในความสัมพันธ์แต่ละ

ตาราง

2. ความสัมพันธ์แบบ หนึ่งต่อหลาย (one-to-many) หมายถึง 1 แถว ในตารางข้อมูลหนึ่ง

สามารถเชื่อมกับหลายแถวในอีกตาราง

3. ความสัมพันธ์แบบ หลายต่อหลาย (many-to-many) หมายถึง หลายแถวในตารางข้อมูล

หนึ่งสามารถเชื่อมกับหลายแถวในอีกตาราง

**2.5.3 การออกแบบฐานข้อมูลเชิงสัมพันธ์**

ขั้นตอนการออกแบบฐานข้อมูลเชิงสัมพันธ์มีขั้นตอนการออกแบบดังนี้

1. การกำหนดวัตถุประสงค์ของฐานข้อมูล

ในการออกแบบฐานข้อมูล การกำหนดวัตถุประสงค์เป็นขั้นตอนแรกที่มีความสำคัญเนื่องจากจะช่วยให้ผู้ออกแบบและพัฒนาสามารถที่จะวางแผนและกำหนดส่วนต่าง ๆ ที่เกี่ยวข้องกับระบบฐานข้อมูลได้ตรงตามความต้องการของผู้ใช้

2. การค้นหาและจัดระเบียบข้อมูลที่ต้องการ

เป็นการรวบรวมข้อมูลต่าง ๆ ที่ผู้ออกแบบต้องการบันทึกไว้ในฐานข้อมูล เช่น ชื่อผู้ป่วย เพศ ชื่อยา คิวอาร์โค้ด เบอร์โทรศัพท์

3. การแบ่งข้อมูลเป็นตาราง

เป็นการแบ่งรายการข้อมูลเป็นรายการหรือหัวข้อหลัก เช่น ตารางพนักงาน ตารางการใช้ยาในแต่ละช่วงเวลาของวัน เช้า กลางวันและเย็น โดยแต่ละรายการหรือหัวข้อหลักจะถูกเปลี่ยนให้อยู่ในรูปแบบของตาราง

4. การเปลี่ยนรายการข้อมูลเป็นคอลัมน์

ผู้ออกแบบและพัฒนาต้องตัดสินใจว่าจะใส่ข้อมูลใดไว้ในตาราง แต่ละรายการจะเป็นเขตของข้อมูลและแสดงเป็นคอลัมน์ในตาราง เช่น ตารางพนักงาน อาจมีเขตข้อมูลเช่น ชื่อ นามสกุล

5. การระบุคีย์หลัก

ในแต่ละตารางต้องมีการเลือกคีย์หลักที่ใช้เป็นคอลัมน์ในการระบุแถวแต่ละแถวเพื่อไม่ให้เกิดการไม่ซ้ำกัน เช่น ระบุให้ ID ผู้ใช้งานเป็นคีย์หลักของตาราง

6. การตั้งค่าความสัมพันธ์ตาราง

ผู้ออกแบบและพัฒนาฐานข้อมูลจะต้องมีการตรวจสอบและตัดสินใจว่าในแต่ละตารางมีข้อมูลใดบ้างในตารางที่เกี่ยวข้องกับข้อมูลในตารางอื่น ๆ ในลักษณะต่าง ๆ โดยจะต้องเพิ่มเขตของข้อมูลลงในตารางหรือต้องมีการสร้างตารางใหม่เพื่อทำให้สามารถแบ่งแยกความสัมพันธ์ได้ชัดเจน

7. การจำกัดการออกแบบ

เป็นการวิเคราะห์การออกแบบเพื่อตรวจสอบแก้ไขหากมีข้อผิดพลาดเช่น มีความไม่สัมพันธ์กันของคอลัมน์และข้อมูลในตาราง จะทำการสร้างหรือแก้ไขคอมลัมน์ใหม่แล้วเพิ่มข้อมูลต่าง ๆ ลงไปในแต่ละแถวของข้อมูลเช่น ข้อมูลในบางแถวของคอลัมน์เมื่อผู้ออกแบบทำการค้นหาข้อมูลดังกล่าวให้ดูผลลัพธ์ที่ได้ว่าตรงตามที่ต้องการจากคอลัมน์ในตารางหรือไม่ ถ้าไม่ตรงให้ปรับเปลี่ยนแก้ไขการออกแบบ

8. การนำกฎการนอร์มัลไลซ์ไปใช้

เป็นการนำกฎการนอร์มัลไลซ์ไปใช้กับข้อมูลเพื่อป้องกันการซ้ำซ้อนในตารางว่ามีโครงสร้างที่ถูกต้องหรือไม่ และทำการปรับเปลี่ยนแก้ไขตาราง

**2.5.4 การประยุกต์ใช้การออกแบบฐานข้อมูลเชิงสัมพันธ์กับฐานข้อมูล NoSQL**

เนื่องจากฐานข้อมูลแบบ NoSQL ไม่มีหลักการออกแบบโครงสร้างที่แน่นอนเหมือนกับฐานข้อมูลเชิงสัมพันธ์ แต่โครงสร้างต่าง ๆ เมื่อเปรียบเทียบกับฐานข้อมูลเชิงสัมพันธ์แล้วลักษณะของการใส่ข้อมูลต่าง ๆ มีความคล้ายคลึงกัน ดังนั้นจึงสามารถนำหลักการขั้นตอนการออกแบบฐานข้อมูลเชิงสัมพันธ์ในบางประการมาประยุกต์ใช้เพื่อออกแบบฐานข้อมูลแบบ NoSQL ได้ดังนี้

1. การกำหนดวัตถุประสงค์ของฐานข้อมูล

ทั้งฐานข้อมูลแบบ SQL และ NoSQL ในการออกแบบฐานข้อมูลจะต้องมีการกำหนดวัตถุประสงค์เป็นขั้นตอนแรกเสมอ เพื่อช่วยให้ผู้ออกแบบและพัฒนา สามารถที่จะวางแผนและกำหนดส่วนต่าง ๆ ที่เกี่ยวข้องกับระบบฐานข้อมูลได้ตรงตามความต้องการของผู้ใช้ในกรณีที่เป็นฐานข้อมูลแบบ NoSQL ผู้ใช้ต้องมีการกำหนดวัตถุประสงค์ให้สอดคล้องกับข้อดีของฐานข้อมูลชนิดนี้คือ มีความยืดหยุ่น เข้าถึงข้อมูลได้เร็วและลดการซ้ำซ้อนของข้อมูล เพื่อให้สามารถใช้งานฐานข้อมูลได้เต็มประสิทธิภาพ

2. การค้นหาและจัดระเบียบข้อมูลที่ต้องการ

เป็นข้อมูลที่ผู้ออกแบบต้องการบันทึกและให้ปรากฏในฐานข้อมูลโดยจะต้องมีการรวบรวมไว้ เช่น รหัสประจำตัวผู้ป่วย ชื่อผู้ป่วย ชื่อยา

3. การแบ่งข้อมูลเป็นดอคคิวเมนต์

ในฐานข้อมูลแบบ NoSQL คอลเลคชันสามารถเทียบได้กับตารางในฐานข้อมูลแบบ SQLโดยในแต่ละคอลเลคชันจะมีการเก็บข้อมูลลงในดอคคิวเมนต์ ดังนั้นข้อมูลต่าง ๆ จะต้องมีการแบ่งรายการของข้อมูลเป็นรายการหรือหัวข้อหลัก เช่น อายุ เพศ รหัสประจำตัวของผู้ป่วย โดยแต่ละรายการจะถูกเปลี่ยนให้อยู่ในรูปแบบของดอคคิวเมนต์ซึ่งอยู่ภายในคอลเลคชัน

4. การเปลี่ยนรายการข้อมูลเป็นคีย์และค่าของคีย์

การจัดเก็บข้อมูลใน NoSQL จะมีลักษณะเป็นแนวนอน เมื่อทำการแบ่งข้อมูลที่ต้องการเก็บลงในฐานข้อมูลโดยผู้ออกแบบต้องมีการตัดสินใจว่าจะใส่ข้อมูลของคีย์และค่าของคีย์ใดบ้างไว้ในแต่ละดอคคิวเมนของคอลเลคชันในฐานข้อมูล เช่น คีย์ชื่อผู้ใช้งาน ค่าของคีย์คือสมชาย สมหมาย

5. การจำกัดการออกแบบ

เป็นการวิเคราะห์การออกแบบเพื่อตรวจสอบแก้ไขหากมีข้อผิดพลาด มีความไม่สัมพันธ์กันของคีย์และค่าของคีย์ เช่นข้อมูลของคีย์ในบางคีย์ เมื่อผู้ออกแบบต้องการผลลัพธ์และทำการค้นหาข้อมูลดังกล่าวให้ดูที่ผลลัพธ์ที่ได้ว่าตรงตามที่ต้องการจากภายในดอคคิวเมนต์นั้นหรือไม่ ถ้าไม่ตรงให้ปรับเปลี่ยนแก้ไขการออกแบบ

6. การนำกฎการนอร์มัลไลซ์ไปใช้

ในฐานข้อมูลแบบ NoSQL นั้น การเก็บข้อมูลจะเก็บแบบแนวนอนในลักษณะของ JSON ดังนั้นอาจจะเกิดการซ้ำซ้อนของข้อมูลได้เพราะไม่มีการเก็บข้อมูลในลักษณะของตารางที่มีการแยกความสัมพันธ์ในแต่ละข้อมูลอย่างชัดเจนซึ่งเป็นการนำกฎการนอร์มัลไลซ์ไปใช้ ดังนั้นในการออกแบบฐานข้อมูลของ NoSQL ผู้ออกแบบต้องมีการออกแบบและวางแผนในการกำหนดคีย์ต่าง ๆ ให้เป็นระบบหรือหมวดหมู่และตรวจสอบได้ง่าย เพื่อลดโอกาสที่จะเกิดการซ้ำซ้อนกันของข้อมูลต่าง ๆภายในคอลเลคชัน

**2.5.5 ประโยชน์ของระบบฐานข้อมูล**

การจัดเก็บข้อมูลต่าง ๆ ไว้ในระบบจัดการฐานข้อมูลเดียวกัน สามารถช่วยให้ผู้ออกแบบและผู้ใช้งานได้รับประโยชน์ดังนี้

1. สามารถใช้ข้อมูลร่วมกันได้ แต่ละคนสามารถใช้ข้อมูลในระบบฐานข้อมูลได้และโปรแกรมอาจใช้ฐานข้อมูลร่วมกันได้

2. สามารถกำหนดมาตรฐานของข้อมูลได้ ผู้บริหารฐานข้อมูลสามารถกำหนดมาตรฐานในการจัดเก็บข้อมูลให้เป็นลักษณ์เดียวกัน เช่น โครงสร้างข้อมูล เป็นต้น

3. สามารถลดความซ้ำซ้อนของข้อมูลได้ ผู้ใช้ที่ต้องการจัดเก็บข้อมูลใช้โดยผ่านระบบการจัดการฐานข้อมูล ทำให้ข้อมูลไม่ซ้ำกันและไม่เปลืองเนื้อที่ในการเก็บข้อมูล

4. ลดความขัดแย้งของข้อมูลได้ ข้อมูลชุดเดียวกันที่ปรากฎอยู่หลายแห่งในฐานข้อมูลต้องตรงกัน ถ้ามีการแก้ไขข้อมูล ระบบการจัดการฐานข้อมูลจะต้องแก้ไขให้ถูกต้องตามกันหมดโดยอัตโนมัติ

5. สามารถป้องกันและรักษาความความปลอดภัยของข้อมูลได้ โดยกำหนดสิทธิ์ของผู้ใช้ แต่ละคนตามระดับการใช้งาน เช่น การเข้าใช้ข้อมูลบางอย่างเป็นที่ข้อมูลส่วนตัวบุคคลภายนอกไม่สามารถเข้ามาดูได้

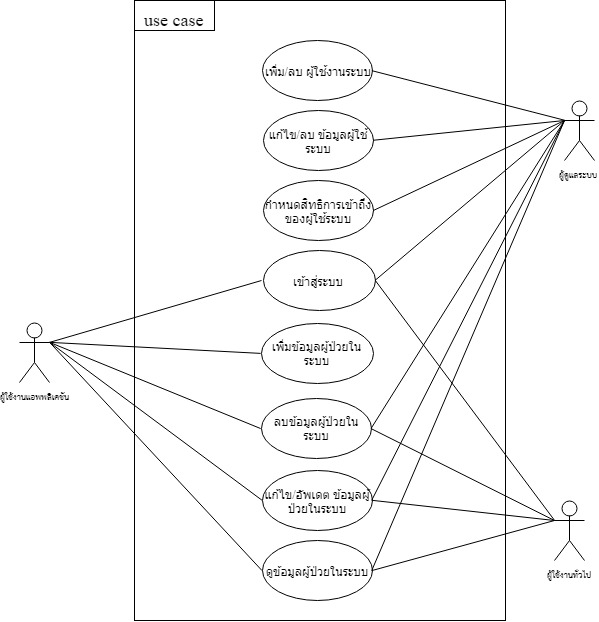
6. สามารถดูแลรักษาฐานข้อมูลได้ง่าย เช่น การทำสำเนาข้อมูล การบูรณะฐานข้อมูลให้กลับสภาพปกติ

**บทที่ 3**

**การออกแบบ การสร้าง และการทำงาน**

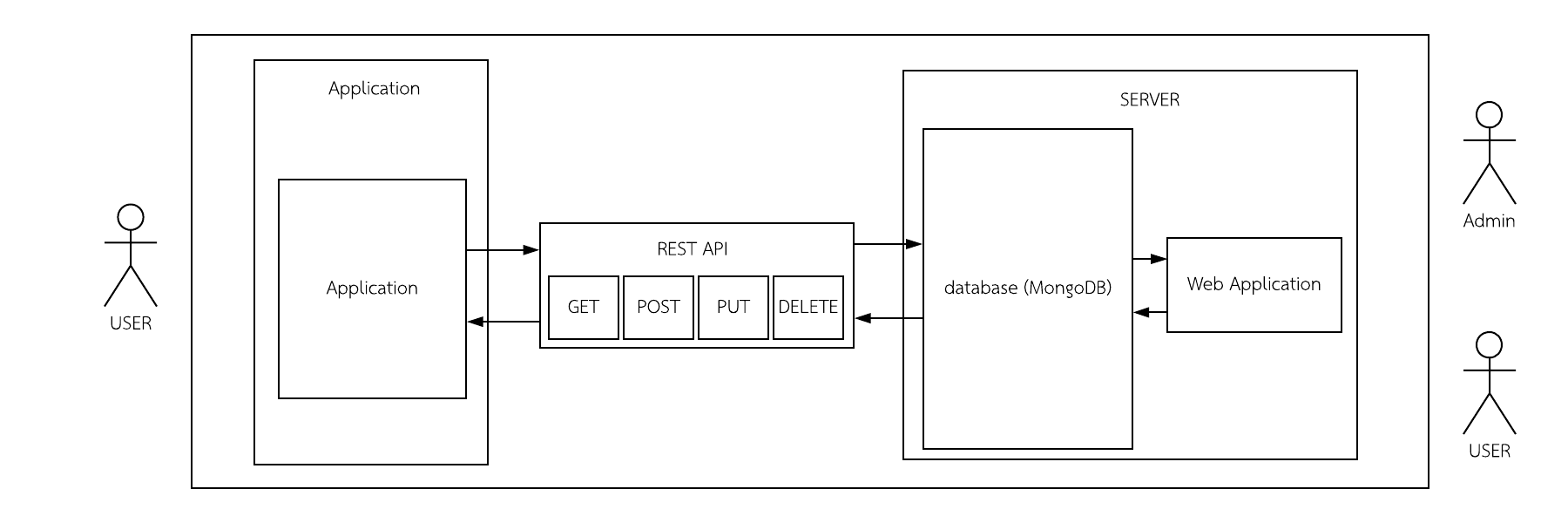
**3.1 กล่าวนำ**

การออกแบบและการสร้างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย จะแบ่งออกเป็น 2 ส่วน คือ ส่วนติดต่อกับผู้ใช้งานและส่วนประมวลผลและจัดการข้อมูล ในส่วนติดต่อกับผู้ใช้งานจะเป็นการสร้างหน้าเว็บแอพพลิเคชั่นโดยใช้ภาษา HTML AngularJS และ CSS ส่วนนี้เป็นส่วนที่ใช้ติดต่อกับผู้ใช้งานโดยตรงในการเพิ่ม ลบ แก้ไขและอ่านข้อมูลต่าง ๆ ในฐานข้อมูลโดยจะต้องมีการเข้าสู่ระบบเพื่อดำเนินการกับข้อมูลต่าง ๆ ที่อยู่ในฐานข้อมูลของระบบ ซึ่งมีการทำงานควบคู่กับส่วนติดต่อกับผู้ใช้งานเมื่อมีการส่ง Request บางอย่างเข้ามา โดยการทำงานมีความแตกต่างกันขึ้นอยู่กับผู้ที่เข้าใช้งานระบบในขณะนั้น ดังรูปที่ 3.1



**รูปที่ 3.1** แผนผังภาพรวมการใช้งานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

ในส่วนของการออกแบบเพื่อติดต่อกับฝั่งแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยสำหรับการเพิ่ม ลบ แก้ไข อัพเดตข้อมูลต่าง ๆ เกี่ยวกับการกินยาและข้อมูลที่เกี่ยวข้องกับตัวผู้ป่วย ได้ออกแบบให้สามารถทำงานได้โดยเรียกใช้บริการของ REST API ซึ่งเป็น API แบบหนึ่งที่มีฟังก์ชันพื้นฐานหลัก ๆ ประกอบไปด้วยการ GET POST PUT และ DELETE ดังรูปที่ 3.2



**รูปที่ 3.2** แผนผังการรับส่งข้อมูลระหว่างแอปพลิเคชันและเซิร์ฟเวอร์โดยการใช้งาน API

**3.2 การออกแบบและการสร้างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย**

ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย ในส่วนของ Server จะประกอบด้วย 2 ส่วนซึ่งมีการทำงานควบคู่กัน คือ

1. ส่วนติดต่อกับผู้ใช้งาน หรือหน้าเว็บแอปพลิเคชัน สร้างจากภาษา HTML CSS และ AngularJS โดยส่วนนี้ประกอบไปด้วย หน้าเว็บเพจการเข้าสู่ระบบ หน้าเว็บเพจการตั้งค่าผู้ใช้งาน หน้าเว็บเพจแสดงข้อมูลต่าง ๆ ของผู้ป่วย หน้าเว็บเพจสำหรับการแก้ไข อัพเดต ลบข้อมูลผู้ป่วย และหน้าเว็บเพจสำหรับติดต่อผู้ดูแลระบบ

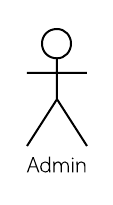
2. ส่วนประมวลผลและจัดการข้อมูล จะใช้ MongoDB เป็นฐานข้อมูลและ Node.js เป็นเซิร์ฟเวอร์โดยในส่วนนี้ประกอบไปด้วยการออกแบบโครงสร้างของฐานข้อมูลที่จะใช้เก็บข้อมูลต่าง ๆ และการนำโครงสร้างที่ออกแบบมาสร้างเป็นโครงสร้างของฐานข้อมูลเพื่อนำไปใช้งานตามความต้องการของผู้ใช้

**3.2.1 การออกแบบและการสร้างส่วนติดต่อกับผู้ใช้งาน**

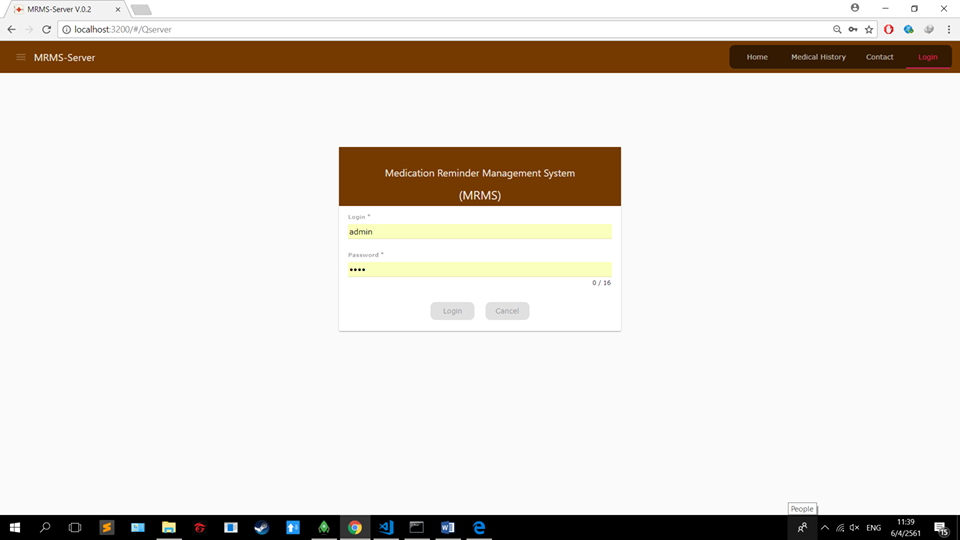
การออกแบบส่วนติดต่อกับผู้ใช้งานประกอบด้วยการออกแบบส่วนต่าง ๆ ดังนี้

**3.2.1.1 การออกแบบและสร้างหน้าเว็บเพจการเข้าสู่ระบบ**

การออกแบบและสร้างหน้าเว็บเพจการเข้าสู่ระบบ ได้ทำการออกแบบมาเพื่อใช้เป็นหน้าสำหรับการเข้าใช้งานระบบโดยการยืนยันตัวตนด้วยชื่อและรหัสผ่านซึ่งจะมีฟิลด์สำหรับใส่ข้อมูลในช่อง Login และช่อง Password เพื่อให้สามารถเข้ามาจัดการกับข้อมูลในระบบได้ตามสิทธิ์ที่กำหนดไว้ดังรูปที่ 3.4 โดยผู้ที่ใช้งานระบบจะแบ่งเป็น 2 กลุ่มคือ ผู้ดูแลระบบและผู้ใช้งานทั่วไป โดยถ้าเข้าสู่ระบบด้วยชื่อของผู้ใช้ทั่วไปจะสามารถจัดการข้อมูลต่าง ๆ ในระบบได้ยกเว้นการเข้าหน้าสำหรับตั้งค่าผู้ใช้งาน ถ้าเข้าสู่ระบบด้วยชื่อของผู้ดูแลระบบจะสามารถจัดการข้อมูลต่าง ๆ ในระบบได้ทั้งหมดรวมถึงการจัดการตั้งค่าการเข้าใช้งานให้กับผู้งานระบบทั่วไป



**รูปที่ 3.3** แผนภาพการเข้าสู่ระบบด้วยโดยผู้ดูแล

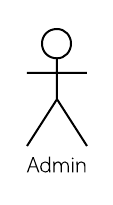


**รูปที่ 3.4** รูปแบบหน้าเว็บเพจการเข้าสู่ระบบ

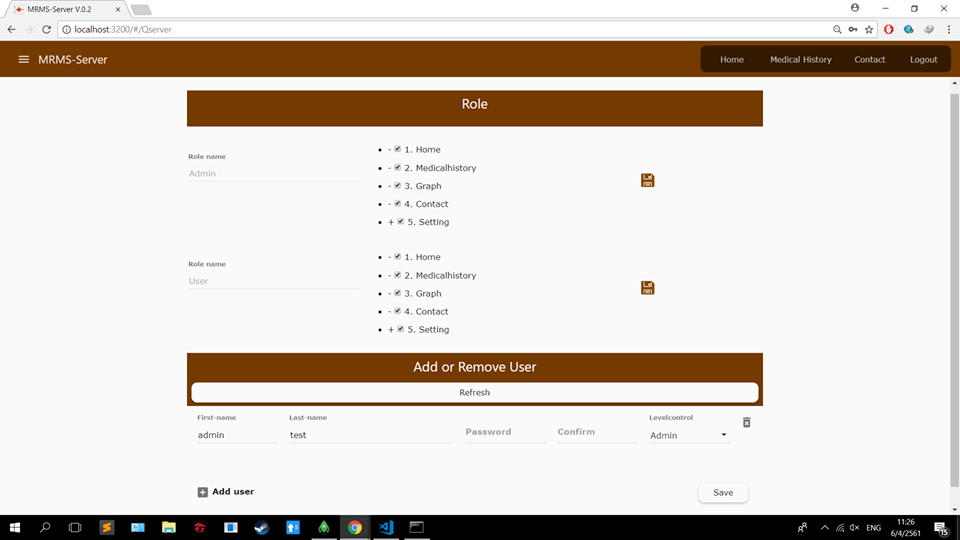
**3.2.1.2 การออกแบบหน้าเว็บเพจการตั้งค่าผู้ใช้งาน**

ในการเข้าถึงฐานข้อมูลเพื่อดำเนินการต่าง ๆ ทั้งการเพิ่ม การลบ การแก้ไขรวมถึงการอ่านข้อมูลของผู้ใช้งานระบบจะต้องมีการออกแบบเพื่อกำหนดสิทธิ์ให้สามารถเข้าถึงข้อมูลภายในฐานข้อมูลได้ดังรูปที่ 3.4 เป็นการป้องกันไม่ให้เกิดความผิดพลาดในกรณีต่าง ๆ เช่น ผู้ใช้งานทั่วไปจะไม่สามารถเข้ามาจัดการกับข้อมูลของผู้ใช้คนอื่นได้ และเพื่อเป็นการจำแนกด้วยว่าเมื่อมีการเข้าสู่ระบบด้วยชื่อใด ๆ จะสามารถมีสิทธิ์ในการดำเนินการต่าง ๆ ได้ไม่เท่ากันเช่น หากเข้าสู่ระบบด้วยชื่อผู้ดูแลระบบจะสามารถเข้าถึงและจัดการข้อมูลต่าง ๆ ของผู้ใช้งานระบบได้ดังรูปที่ 3.6

ในหน้านี้จะมีฟิลด์สำหรับกำหนดว่า เมื่อผู้ใช้งานเข้าสู่ระบบด้วยชื่อใด ๆ จะสามารถเข้าใช้งานหน้าเพจได้ตามสิทธิ์ที่ถูกกำหนดไว้เท่านั้น



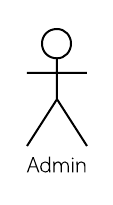
**รูปที่ 3.5** แผนภาพการทำงานของผู้ใช้ระบบในการเข้าใช้งานหน้าเว็บเพจตั้งค่าข้อมูลผู้ใช้งาน



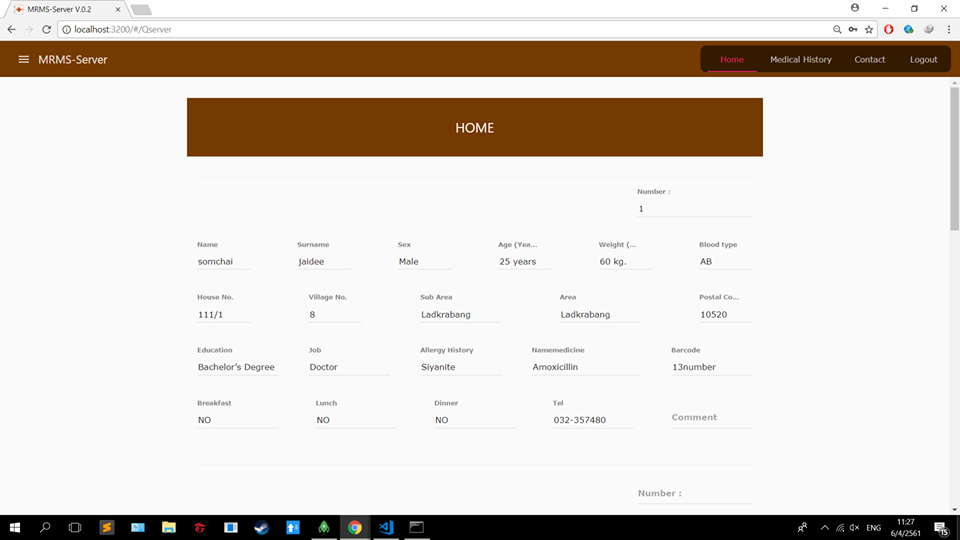
**รูปที่ 3.6** การออกแบบหน้าเว็บเพจการตั้งค่าผู้ใช้งาน

**3.2.1.3 การออกแบบหน้าเว็บเพจแสดงข้อมูลต่าง ๆ ของผู้ป่วย**

ภายในฐานข้อมูลจะประกอบไปด้วยข้อมูลต่าง ๆ ที่เกี่ยวข้องกับผู้ป่วย เพื่อให้สามารถเรียกดูหรืออ่านข้อมูลที่อยู่ภายในฐานข้อมูลได้ จึงต้องมีการสร้างหน้าเว็บเพจที่ใช้เรียกดูข้อมูลต่าง ๆ ของผู้ป่วยเพื่อให้มีความสะดวกและรวดเร็วในการบริหารและจัดการกับผู้ป่วยเป็นรายบุคคลดังรูปที่ 3.8



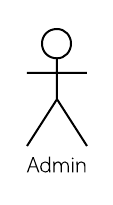
**รูปที่ 3.7** แผนภาพการทำงานของผู้ใช้ระบบในการเข้าใช้งานหน้าเว็บเพจแสดงข้อมูลผู้ป่วย



**รูปที่ 3.8** การออกแบบหน้าเว็บเพจแสดงข้อมูลต่าง ๆ ของผู้ป่วย

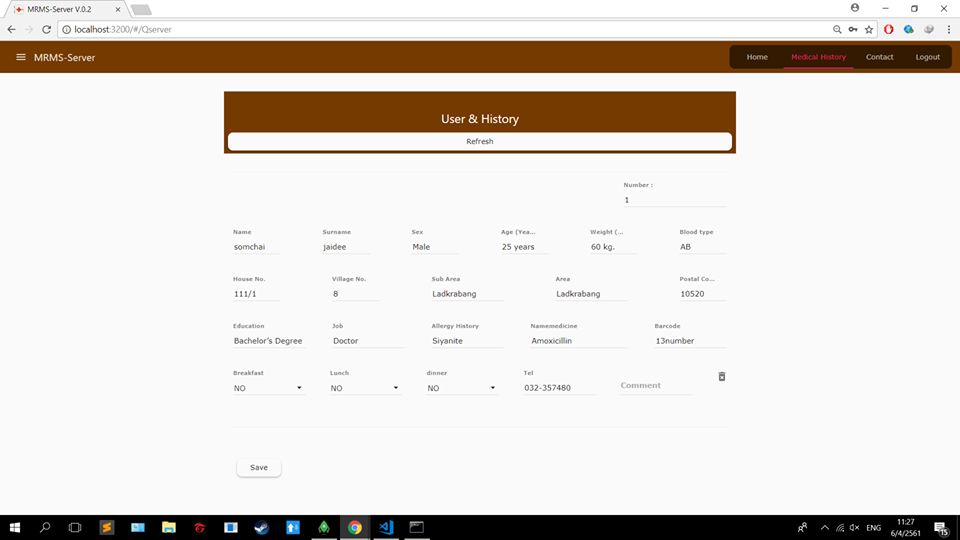
**3.2.1.4 การออกแบบหน้าเว็บเพจสำหรับการแก้ไข อัพเดต ลบข้อมูลผู้ป่วย**

ในการแก้ไข อัพเดต ลบประวัติหรือข้อมูลต่าง ๆ เป็นรายบุคคลจะต้องมีการออกแบบและสร้างหน้าเว็บเพจสำหรับการแก้ไข อัพเดต ลบผู้ป่วยดังรูปที่ 3.9



**รูปที่ 3.9** แผนภาพการทำงานของหน้าเว็บเพจแก้ไข อัพเดต และลบข้อมูลผู้ป่วยโดยผู้ดูแลระบบ

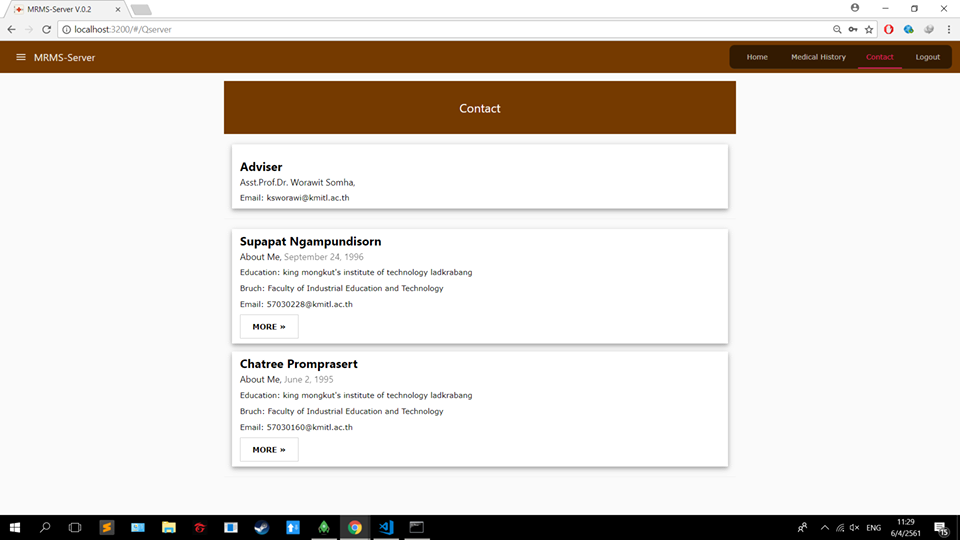
เพื่อให้เกิดประสิทธิภาพและดำเนินการได้อย่างถูกต้อง การออกแบบต้องมีความสอดคล้องกับคีย์ที่ใช้เก็บข้อมูลภายในฐานข้อมูล ซึ่งคีย์ดังกล่าวจะมีความสัมพันธ์กับการใส่ข้อมูลหรือแก้ไขข้อมูลของผู้ใช้ โดยผู้ใช้ระบบสามารถแก้ไข อัพเดต ลบข้อมูลและสามารถบันทึกลงไปยังคีย์ที่มีการใส่ข้อมูลนั้นภายในฐานข้อมูลได้ด้วยการกดปุ่ม save และสามารถลบข้อมูลโดยกดปุ่มลบเพื่อลบข้อมูลออกจากภายในฐานข้อมูลที่มีการออกแบบไว้ดังแสดงในรูปที่ 3.10



**รูปที่ 3.9** การออกแบบหน้าเว็บเพจสำหรับการเพิ่ม ลบ แก้ไขผู้ป่วย

**3.2.1.5 การออกแบบหน้าเว็บเพจสำหรับติดต่อผู้ดูแลระบบ**

เนื่องจากระบบต้องมีการพัฒนาตลอดเวลาเพื่อตอบสนองต่อความต้องการของผู้ใช้และเพื่อเป็นการแจ้งเกี่ยวกับการทำงานต่าง ๆ ภายในระบบว่าสามารถทำงานได้ถูกต้องและสมบูรณ์ตามการออกแบบจึงต้องมีการสร้างหน้าเว็บเพจที่บอกข้อมูลของผู้พัฒนาและช่องทางการใช้ติดต่อกับผู้ดูแลระบบเช่น email เพื่อนำไปปรับปรุงระบบให้มีความพร้อมในการใช้งานอยู่ตลอดเวลาดังรูปที่ 3.11



**รูปที่ 3.11** การออกแบบหน้าเว็บเพจสำหรับติดต่อผู้ดูแลระบบ

**3.2.1.6 การสร้างปุ่มกด**

การสร้างปุ่มกดสามารถสร้างได้โดยใช้แท็กคำสั่ง <md-button> ... </md-button> โดยสามารถเพิ่มคำสั่งสำหรับการทำงานเมื่อเกิดเหตุการณ์หรือการกระทำกับปุ่มกด ng-click="ชื่อฟังก์ชัน ()" ตัวอย่างเช่น ng-click = "saveEmployee ($event)" เป็นการเรียกใช้ฟังก์ชัน saveEmployee เมื่อมีการกดปุ่ม



**รูปที่ 3.12** ตัวอย่างคำสั่งการสร้างปุ่มกดเพื่อบันทึกข้อมูลลงในระบบ

**3.2.2.7 การสร้างฟิลด์รับข้อมูล**

การสร้างฟิลด์สำหรับรับข้อมูลที่ผู้ใช้งานระบบป้อนเข้ามา สามารถสร้างได้โดยการเรียกใช้คำสั่ง <md-input-container > … </md-input-container> ภายในแท็กคำสั่งสามารถใช้คำสั่งเสริมเพื่อให้ทำงานร่วมกันได้ โดยสามารถกำหนดรูปแบบชนิดข้อมูลที่ต้องการให้ใส่ได้ เช่น text password เป็นต้น



**รูปที่ 3.13** ตัวอย่างคำสั่งการสร้างฟิลด์รับข้อมูล

**3.2.2 การทำงาน**

การทำงานในส่วนติดต่อกับผู้ใช้งานผ่านทางหน้าเว็บแอปพลิเคชัน จะสามารถแบ่งออกได้เป็น 2 กลุ่มผู้ใช้ดังนี้

**3.2.2.1 การทำงานสำหรับผู้ใช้ทั่วไป**

การทำงานสำหรับผู้ใช้ทั่วไปจะถูกกำหนดให้สามารถเข้าใช้งานได้ด้วยการเข้าสู่ระบบผ่านการ login และจะเข้าสู่หน้า Home Page ซึ่งประกอบด้วยเมนูต่าง ๆ เพื่อไปยังหน้าเพจอื่นได้คือ หน้าเว็บเพจ Medical History และหน้าเว็บเพจ Contact ซึ่งในหน้าเพจดังกล่าวมีรายละเอียดการทำงานดังนี้

เมนู Home Page เป็นหน้าที่แสดงข้อมูลผู้ป่วยที่มีอยู่ในระบบ

เมนู Medical History เป็นหน้าที่สำหรับใช้แก้ไขข้อมูล อัพเดตและลบข้อมูลผู้ป่วยที่อยู่ในระบบ

เมนู Contact เป็นหน้าสำหรับแจ้งผู้ดูแลระบบหากเกิดข้อผิดพลาดหรือต้องคำแนะสำหรับปรับปรุงระบบให้มีคามต้องการตรงกับผู้ใช้งาน

เมนู Login ใช้สำหรับออกจากระบบ

**3.2.2.2 การทำงานสำหรับผู้ดูแลระบบ**

การทำงานสำหรับผู้ดูแลระบบ จะถูกกำหนดให้สามารถเข้าใช้งานได้ด้วยการเข้าสู่ระบบผ่านการ login เช่นเดียวกับผู้ใช้ระบบทั่วไปและจะเข้าสู่หน้า Home Page ซึ่งประกอบด้วยเมนูเพื่อไปยังหน้าเพจอื่นได้คือหน้าเว็บเพจ Medical History และหน้าเว็บเพจContact แต่จะมีความแตกต่างที่ผู้ดูแลสามารถเข้าเมนู Setting เพื่อเข้าไปจัดการกับข้อมูลของผู้ใช้ที่อยู่ในระบบทุกคนได้

เมนู Setting เป็นหน้าสำหรับตั้งค่าเกี่ยวกับผู้ใช้ที่อยู่ในระบบ ใช้กำหนดสิทธิ์ในการเข้าถึงหน้าเมนูต่าง ๆ ของผู้ใช้แต่ละคน

เมนู Home Page เป็นหน้าที่แสดงข้อมูลผู้ป่วยที่มีอยู่ในระบบ

เมนู Medical History เป็นหน้าที่สำหรับใช้แก้ไขข้อมูล อัพเดตและลบข้อมูลผู้ป่วยที่อยู่ในระบบ

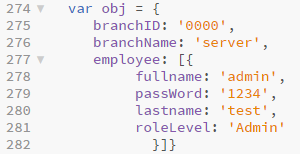
เมนู Contact เป็นหน้าสำหรับแจ้งผู้ดูแลระบบหากเกิดข้อผิดพลาดหรือต้องคำแนะสำหรับปรับปรุงระบบให้มีคามต้องการตรงกับผู้ใช้งาน

เมนู Login ใช้สำหรับออกจากระบบ

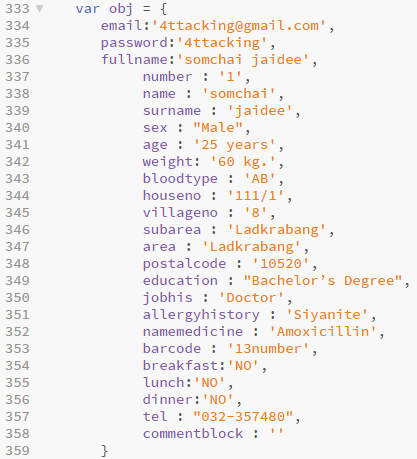
**3.2.3 การออกแบบและการสร้างส่วนประมวลผลและจัดการข้อมูล**

การออกแบบและสร้างระบบจัดการข้อมูลการแจ้งเตือนการรับประทานของผู้ป่วยจะเริ่มจากการเลือกชนิดฐานข้อมูลที่ต้องการใช้เพื่อเป็นฐานข้อมูล

เนื่องจากข้อมูลของผู้ป่วยนั้น ในการเก็บข้อมูลจะต้องเก็บข้อมูลเป็นจำนวนมากทำให้ในอนาคตอาจต้องการขยายระบบและเพื่อให้ง่ายต่อการเพิ่มข้อมูล จึงเลือกฐานข้อมูลที่ขยายระบบได้ง่าย ไม่ต้องคำนึงถึงความสัมพันธ์ของตาราง จึงได้ทำการเลือกระบบจัดการฐานข้อมูลแบบ NoSQL จากนั้นรวบรวมข้อมูลเพื่อใช้เป็นโครงสร้างของฐานข้อมูลโดยข้อมูลแต่ละตัวจะมีการระบุเป็นคีย์และค่าของคีย์ซึ่งจะอยู่ภายในคอลเลคชันเพื่อใช้แยกเก็บข้อมูลที่อยู่คนละกลุ่มกัน เช่น ข้อมูลของผู้ใช้ระบบ ข้อมูลของผู้ป่วย เมื่อทำการกำหนดโครงสร้างจึงทำการสร้างตามโครงสร้างที่ได้ทำการออกแบบไว้



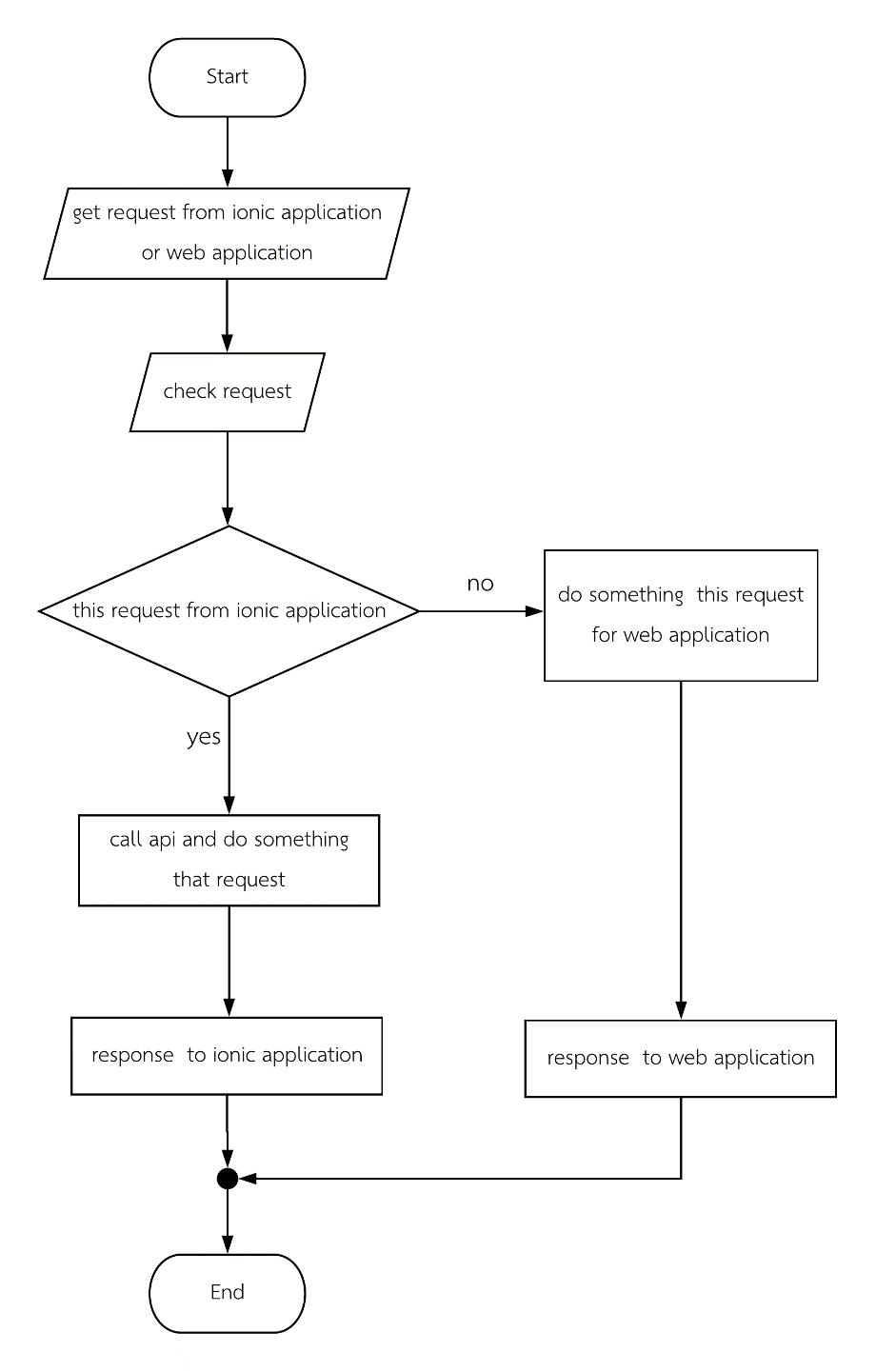
**รูปที่ 3.14** โครงสร้างของคอลเลคชันเพื่อใช้เก็บข้อมูลผู้ใช้งานระบบ



**รูปที่ 3.15** โครงสร้างของคอลเลคชันเพื่อใช้เป็นโครงสร้างเก็บข้อมูลของผู้ป่วย

**3.2.3.1** การออกแบบและการทำงานของเซิร์ฟเวอร์ Node.js

ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย ทำงานโดยใช้ Node.js เป็นเซิร์ฟเวอร์ ในการทำงานเพื่อติดต่อรับส่งข้อมูลแบ่งได้เป็น 2 ส่วนคือ ส่วนที่ใช้สำหรับทำงานกับผู้ใช้ผ่านทางหน้าเว็บแอปพลิเคชันและส่วนที่สองใช้สำหรับทำงานกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วย โดยในการทำงานทั้งสองส่วนนั้นจะทำงานเมื่อมีการขอ request ซึ่งเซิร์ฟเวอร์จะทำการประมวลผลและส่ง response กลับไปดังรูปที่ 3.16



รูปที่ 3.16 ผังงานการทำงานของเซิร์ฟเวอร์กับผู้ใช้งานผ่านหน้าเว็บแอปพลิเคชัน

**3.3 การออกแบบและการสร้าง API เพื่อติดต่อกับฝั่งแอปพลิเคชัน**

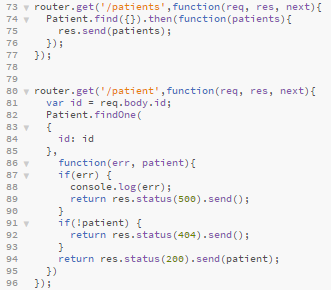
ในส่วนของการติดต่อกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยจะใช้ REST API เป็นตัวกลางในการติดต่อ ทำให้สามารถรับส่งข้อมูลข้าม Platform ได้ โดยการเรียกใช้ฟังก์ชันพื้นฐาน ได้แก่การ GET POST PUT DELETE ซึ่งฟังก์ชันพื้นฐานดังกล่าวจะเป็นฟังก์ชันที่นำมาใช้เพื่อรับส่งข้อมูลระหว่างเซิร์ฟเวอร์และแอปพลิเคชัน โดยการที่จะรับหรือส่งข้อมูลให้กับฐานข้อมูลได้ ต้องมีการกำหนดโครงสร้างที่จะใช้รับส่งให้เหมือนกับโครงสร้างของฐานข้อมูลดังรูปที่ 3.17



รูปที่ 3.17 ตัวอย่างโครงสร้างสำหรับรับส่งข้อมูลของ API ระหว่างแอปพลิเคชันและฐานข้อมูล

**3.3.1 ฟังก์ชัน GET**

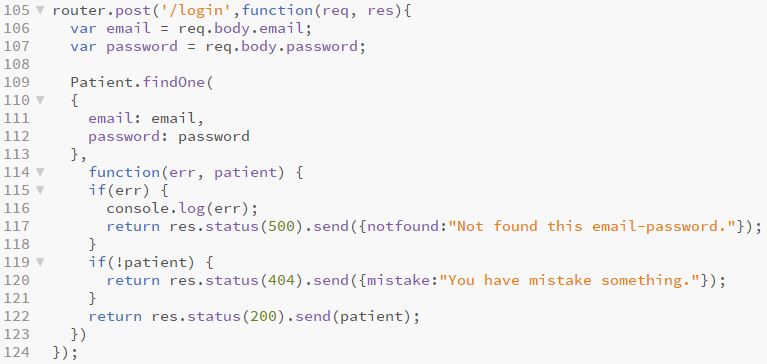
รูปแบบของฟังก์ชัน GET เป็นฟังก์ชันที่ใช้สำหรับการนำข้อมูลในฐานข้อมูลที่ต้องการมาแสดงยังหน้าแอปพลิเคชัน ในการทำงานจะเรียกข้อมูลผ่าน id ของผู้ป่วยที่อยู่ภายในฐานข้อมูล กรณีที่ id ไม่ตรงกับผู้ป่วยคนใด ๆ จะมีการแจ้งเตือนให้ผู้ใช้งานแอปพลิเคชันได้รับทราบ โดยมีรูปแบบการเขียนและการสร้างดังรูปที่ 3.18



รูปที่ 3.18 ตัวอย่างโค้ดฟังก์ชัน GET ของ API

**3.3.2 ฟังก์ชัน POST**

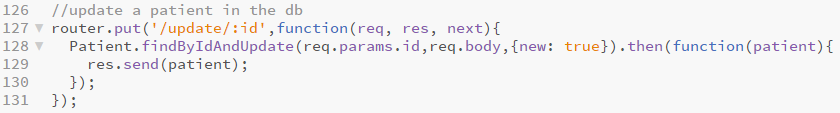
รูปแบบของฟังก์ชัน POST เป็นฟังก์ชันที่ใช้สำหรับการเพิ่มข้อมูลที่ต้องการและบันทึกไปยังฐานข้อมูล ในการเพิ่มแต่ละครั้ง จะต้องมีการสร้างข้อมูลตามโครงสร้างที่ได้ทำการออกแบบไว้ในฝั่งของ api ซึ่งข้อมูลที่เพิ่มจะต้องมีคีย์ที่เหมือนกับโครงสร้างของฐานข้อมูล โดยมีรูปแบบการเขียนและการสร้างดังรูปที่ 3.19



รูปที่ 3.19 ตัวอย่างโค้ดฟังก์ชัน POST ของ API

**3.3.3 ฟังก์ชัน PUT**

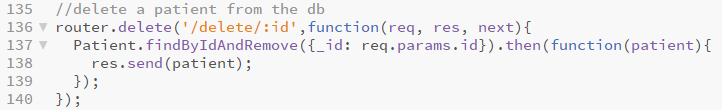
รูปแบบของฟังก์ชัน PUT เป็นฟังก์ชันที่ใช้สำหรับการอัพเดตข้อมูลไปยังฐานข้อมูลที่ต้องการ ในการอัพเดตแต่ละครั้งจะทำการค้นหาไปยัง id และจะทำการบันทึกทับข้อมูลเดิมที่ต้องการภายในฐานข้อมูล ซึ่งการอัพเดตผู้ใช้งานแอปพลิเคชันจะต้องระบุคีย์ของข้อมูลให้ตรงกันทั้งฝั่งเซิร์ฟเวอร์และฝั่งแอปพลิเคชัน ฟังก์ชัน PUT สามารถเขียนและการสร้างได้ดังรูปที่ 3.20



รูปที่ 3.20 ตัวอย่างโค้ดการเรียกใช้ฟังก์ชัน PUT ของ API

**3.3.4 ฟังก์ชัน DELETE**

รูปแบบของฟังก์ชัน DELETE เป็นฟังก์ชันที่ใช้สำหรับการลบข้อมูลที่ต้องการจากภายในฐานข้อมูลโดยการลบจะต้องระบุ id ของข้อมูลเช่นเดียวกับการ GET POST และ PUT มีรูปแบบการเขียนและการสร้างดังรูป



รูปที่ 3.21 ตัวอย่างโค้ดการเรียกใช้ฟังก์ชัน DELETE ของ API

**บทที่ 4**

**การทดลองและผลการทดลอง**

**4.1 กล่าวนำ**

ในส่วนของบทนี้จะกล่าวถึงการทดลองและผลการทดลองของการใช้งานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยเพื่อทดสอบการทำงานของระบบโดยจะแบ่งการทดลองออกเป็นส่วนต่าง ๆ ซึ่งประกอบด้วย การทดลองและผลของการทดลองการใช้งานระบบจัดการข้อมูลการแจ้งเตือนรับประทานยาของผู้ป่วย การทดลองและผลการทดลองการติดต่อรับส่งข้อมูลกับแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยโดยใช้บริการของ API

**4.2 การทดลองการทำงานของระบบจัดการข้อมูล**

การทำงานของระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยมีหน้าที่หลักคือ ใช้กำหนดสิทธิ์ในการเข้าถึงข้อมูลต่าง ๆ ในระบบให้กับผู้ใช้งาน ใช้เพิ่มผู้ใช้งานระบบหากเป็นผู้ดูแล และใช้ในการ ลบ แก้ไข อัพเดต และเรียกดูข้อมูลต่าง ๆ ของผู้ป่วยโดยได้ทำการทดลองดังนี้

**4.2.1 การทดลองเข้าสู่ระบบของผู้ดูแล**

**4.2.1.1 ขั้นตอนการทดลอง**

1. รันเซิร์ฟเวอร์ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. เข้าสู่หน้า Login ผ่าน localhost : 3200 บนหน้าเว็บเบราว์เซอร์

3. ใส่ชื่อผู้ดูแลและรหัสผ่านในช่อง Login และ Password

4. ทำการกดปุ่ม Login เพื่อเข้าใช้งานระบบ

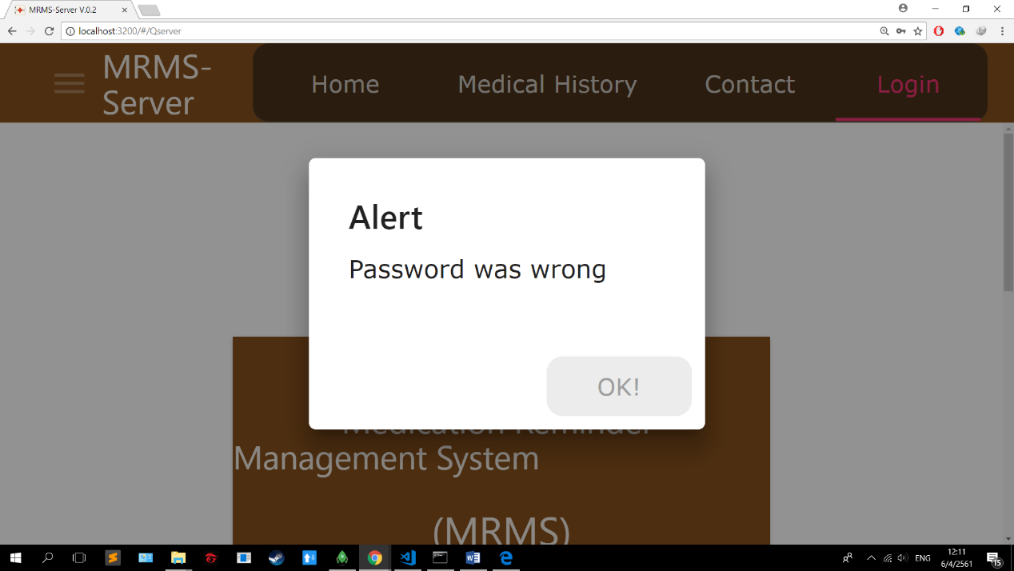
5. สังเกตและบันทึกผลลงในตารางที่ 4.1 โดย

- ถ้าชื่อผู้ใช้และรหัสผ่านถูกต้องจะสามารถเข้าใช้งานระบบได้

- ถ้าชื่อผู้ใช้และรหัสผ่านถูกต้อง ไม่ถูกต้องจะไม่สามารถเข้าใช้งานระบบได้ซึ่งจะแจ้งเตือนให้ผู้ใช้งานทราบดังรูปที่ 4.1

**ตารางที่ 4.1** ผลการทดลองการเข้าสู่ระบบของผู้ดูแล

|  |  |  |
| --- | --- | --- |
| **User name** | **Password** | **ผลลัพธ์** |
| admin | 1234 | สามารถเข้าใช้งานได้ |
| admin | admin | ไม่สามารถเข้าใช้งานได้ |



**รูปที่ 4.1** การแจ้งเตือนเมื่อใส่รหัสผ่านผิด

**4.2.1.2 ผลการทดลอง**

จากการทดลอง ถ้าผู้ดูแลระบบ Login เข้าสู่ระบบโดยใส่ชื่อผู้ใช้และรหัสผ่านได้ถูกต้อง จะสามารถเข้าใช้งานระบบได้ แต่ถ้าใส่ผู้ใช้หรือรหัสผ่านช่องใดช่องหนึ่งผิด จะมีการแจ้งเตือนให้ผู้ดูแลระบบทราบเพื่อใส่ชื่อผู้ใช้และรหัสผ่านใหม่ โดยผู้ดูแลระบบสามารถเข้าสู่หน้าเว็บแอปพลิเคชันได้ทุกหน้ารวมถึงหน้าการเพิ่มผู้ใช้งานทั่วไปได้

**4.2.2 การทดลองเพิ่มผู้ใช้งานโดยผู้ดูแลระบบ**

**4.2.2.1 ขั้นตอนการทดลอง**

1. รันเซิร์ฟเวอร์ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. เข้าสู่หน้า Login ผ่าน localhost : 3200 บนหน้าเว็บเบราว์เซอร์

3. ใส่ชื่อผู้ดูแลระบบและรหัสผ่านในช่อง Login และ Password

4. ทำการกดปุ่ม Login เพื่อเข้าใช้งานระบบ

5. ในแถบ sidebar menu ให้เพิ่มผู้ใช้งานระบบโดยกดปุ่ม Add User โดย

- ในหัวข้อ Role ที่ช่อง role level user เลือกเครื่องหมายถูกหน้าเมนู setting ออกและทำการกดบันทึก

- ในหัวข้อ Add or Remove User ให้ตั้งชื่อผู้ใช้งานและรหัสผ่านเป็น user รหัสผ่าน 4567 และเลือก role level เป็น user

6. กด save และ logout ออกจากระบบ

7. เข้าสู่ระบบด้วยชื่อ user รหัสผ่าน 4567 ที่ทำการสร้างไว้

8. สังเกตการเข้าถึงแต่ละเมนูของหน้าเว็บแอปพลิเคชัน

9. บันทึกผลลงในตารางที่ 4.2

**ตารางที่ 4.2** ผลการทดลองการเข้าใช้งานเมนูตั้งค่าของผู้ใช้งานทั่วไป

|  |  |
| --- | --- |
| **ชื่อผู้ใช้** | **การเข้าใช้งานเมนูตั้งค่า** |
| admin | สามารถใช้งานเมนูการตั้งค่าผู้ใช้งานและเมนูอื่น ๆ ได้ |
| User2 | ไม่สามารถใช้งานเมนูการตั้งค่าผู้ใช้งานแต่เข้าใช้เมนูอื่น ๆ ได้ |

**4.2.2.2 ผลการทดลอง**

จากการทดลองเข้าสู่ระบบโดยชื่อผู้ใช้งานที่เป็น user พบว่า สามารถเข้าใช้งานเมนูเพื่อไปยังหน้า Home Medical History และ Contact ได้แต่ไม่สามารถเข้าสู่หน้าการตั้งค่าผู้ใช้งานได้เพราะถูกกำหนดสิทธิ์ในการเข้าถึงให้ไม่สามารถเข้าได้โดยผู้ดูแลระบบ

**4.2.3 การทดลองแก้ไขข้อมูลผู้ป่วยในระบบ**

**4.2.3.1 ขั้นตอนการทดลอง**

1. รันเซิร์ฟเวอร์ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. Login เข้าสู่ระบบ

3. ไปที่หน้าเว็บเพจ Medical History เลือกฟิลด์ข้อมูลที่ต้องการแก้ไขโดยการใส่ข้อมูลลงไป

4. ในกรณีที่ต้องการลบข้อมูลผู้ป่วย ให้กดลบที่ไอคอนถังขยะ

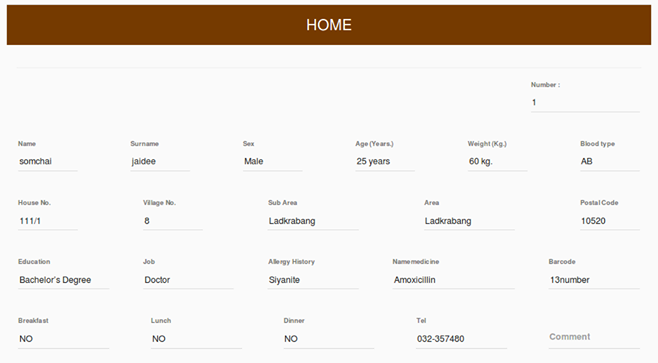
5. กดปุ่ม Save เพื่อบันทึกข้อมูล

6. รีเฟรชหน้าเบราว์เซอร์และทำการเข้าสู่ระบบใหม่อีกครั้ง

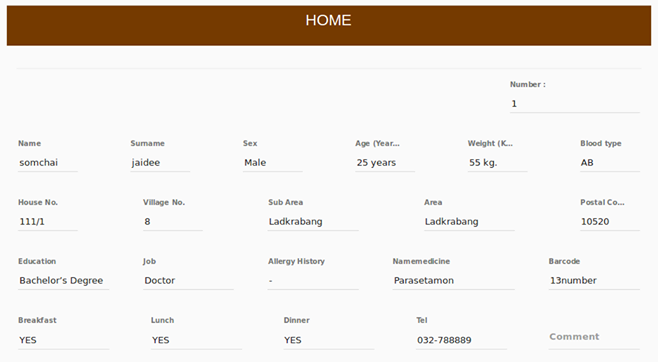
7. สังเกตความเปลี่ยนแปลงและบันทึกผลที่ได้

**4.2.3.2 ผลการทดลอง**

จากการทดลองแก้ไขข้อมูลผู้ป่วยพบว่า เมื่อทำการแก้ไขข้อมูลลงไปในฟิลด์ที่ต้องการโดยใส่ข้อมูลเพิ่ม หรือลบข้อมูลที่มีอยู่ในฟิลด์เดิมออกและทำการบันทึก ข้อมูลดังกล่าวจะถูกเขียนทับโดยการอัพเดตใหม่แทนข้อมูลเดิม และถ้ากดปุ่มลบ ข้อมูลผู้ป่วยคนนั้นจะหายไป ดังรูปที่ 4.2 และ 4.3



**รูปที่ 4.2** ข้อมูลในฐานข้อมูลผู้ป่วยก่อนการแก้ไข



**รูปที่ 4.3** ข้อมูลในฐานข้อมูลผู้ป่วยหลังการแก้ไข

**4.2.4 การทดลองเรียกดูข้อมูลผู้ป่วยในระบบ**

**4.2.4.1 ขั้นตอนการทดลอง**

1. รันเซิร์ฟเวอร์ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. Login เข้าสู่ระบบ

3. ทดลองแก้ไขข้อมูลผู้ป่วยในหน้า Medical History และกดบันทึก

4. รีเฟรชหน้าเว็บเพจในเบราว์เซอร์

5. รีเฟรชหน้าเบราว์เซอร์และทำการเข้าสู่ระบบใหม่อีกครั้ง

6. สังเกตความเปลี่ยนแปลงและบันทึกผลที่ได้

**4.2.4.2 ผลการทดลอง**

จากการทดลองเรียกดูข้อมูลผู้ป่วยในระบบพบว่า เมื่อ Login เข้าสู่ระบบจะเข้าสู่หน้า Home ซึ่งจะแสดงข้อมูลต่าง ๆ ที่อยู่ในระบบทั้งหมดให้ผู้ใช้งานเห็น เมื่อข้อมูลในฐานข้อมูลมีการเปลี่ยนแปลงหรือถูกแก้ไขและบันทึกลงในระบบ ข้อมูลดังกล่าวจะถูกดึงออกมาแสดงบนหน้า Home ซึ่งเห็นได้จากการรีเฟรชหน้าเบราว์เซอร์และเข้าสู่ระบบใหม่อีกครั้ง

**4.3 การทดลองติดต่อรับส่งข้อมูลกับแอปพลิเคชันโดยใช้ API**

**4.3.1 การทดลองใช้ฟังก์ชัน GET POST PUT DELETE**

**4.3.1.1 ขั้นตอนการทดลอง**

1. รันเซิร์ฟเวอร์ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย

2. Login เข้าสู่ระบบ

3. ในฝั่งของแอปพลิเคชัน (ARC) ให้ทำการเรียกใช้งาน API เพื่อเพิ่มข้อมูล ดึงข้อมูล อัพเดตข้อมูลและลบข้อมูลของผู้ป่วย ซึ่งทำตามขั้นตอนต่อไปนี้

3.1 การเพิ่มข้อมูลโดยการเรียกใช้ฟังก์ชัน POST

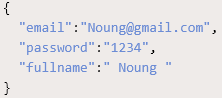
รูปแบบที่ 1

- ทำการรันเซิร์ฟเวอร์เพื่อทำการเรียกใช้ Path API

- เปิดโปรแกรม ARC เพื่อทดลองเพิ่มข้อมูลผู้ป่วยผ่านการ POST

- ทำการเลือก Method POST แล้วทำการใส่ localhost:3200/users ลงไปในช่อง Request URL

- ในส่วนของ Body ให้พิมข้อมูล JSON ตามรูปที่ 4.5 และกดปุ่ม send



**รูปที่ 4.4** ตัวอย่าง JSON FORM ของ Method POST รูปแบบที่ 1

- สังเกตและบันทึกผลลงในตารางที่ 4.3

รูปแบบที่ 2

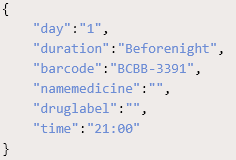
- ทำการรันเซิร์ฟเวอร์เพื่อทำการเรียกใช้ Path API

- เปิดโปรแกรม ARC เพื่อทดลองเพิ่มข้อมูลผู้ป่วยผ่านการ POST

- ทำการเลือก Method POST แล้วทำการใส่ localhost:3200/users/

:userID/dates ลงไปใน Request URL

- ในส่วนของ Body ให้พิมข้อมูล JSON ตามรูปที่ 4.6 และกดปุ่ม send



**รูปที่ 4.5** ตัวอย่าง JSON FORM ของ Method POST รูปแบบที่ 2

- สังเกตและบันทึกผลลงในตารางที่ 4.3

**ตารางที่ 4.3** ตารางบันทึกผลการเพิ่มข้อมูลโดย Method POST รูปแบบที่ 1 และ 2

|  |  |  |
| --- | --- | --- |
| **รูปแบบ Method POST** | **การกรอกข้อมูล** | **ผลลัพธ์** |
| แบบที่ 1 | กรอกถูกต้อง | สามารถเพิ่มข้อมูลได้ |
| แบบที่ 2 | กรอกไม่ครบหรือกรอกผิด | ไม่สามารถเพิ่มข้อมูลได้ |

3.2 การดึงข้อมูลโดยการเรียกใช้ฟังก์ชัน GET

รูปแบบที่ 1

- ทำการรันเซิร์ฟเวอร์เพื่อทำการเรียกใช้ Path API

- เปิดโปรแกรม ARC เพื่อทดลองเพิ่มข้อมูลผู้ป่วยผ่านการ GET

- ทำการเลือก Method GET แล้วทำการใส่ localhost:3200/users ลงไปในช่อง Request URL และกดปุ่ม send

- สังเกตและบันทึกผลลงในตารางที่ 4.4

รูปแบบที่ 2

- ทำการรันเซิร์ฟเวอร์เพื่อทำการเรียกใช้ Path API

- เปิดโปรแกรม ARC เพื่อทดลองเพิ่มข้อมูลผู้ป่วยผ่านการ GET

- ทำการเลือก Method GET แล้วทำการใส่ localhost:3200/users/

:userID ลงไปในช่อง Request URL และกดปุ่ม send

- สังเกตและบันทึกผลลงในตารางที่ 4.4

**ตารางที่ 4.4** ตารางบันทึกผลการเพิ่มข้อมูลโดย Method GET รูปแบบที่ 1 และ 2

|  |  |  |
| --- | --- | --- |
| **รูปแบบ Method GET** | **การกรอกข้อมูล** | **ผลลัพธ์** |
| แบบที่ 1 | กรอกถูกต้อง | สามารถดึงข้อมูลได้ |
| แบบที่ 2 | กรอกไม่ครบหรือกรอกผิด | ไม่สามารถดึงข้อมูลได้ |



**รูปที่ 4.6** ตัวอย่างข้อมูลที่ถูกดึงจากฐานข้อมูลโดยการใช้ Method GET

3.3 การแก้ไขข้อมูลโดยการเรียกใช้ฟังก์ชัน PUT

- ทำการรันเซิร์ฟเวอร์เพื่อทำการเรียกใช้ Path API

- เปิดโปรแกรม ARC เพื่อทดลองเพิ่มข้อมูลผู้ป่วยผ่านการ GET

- ทำการเลือก Method GET แล้วทำการใส่ localhost:3200/users/:userID

ลงไปในช่อง Request URL

- ในส่วนของ Body ให้พิมข้อมูล JSON ตามรูปที่ 4.7 และกดปุ่ม send



**รูปที่ 4.7** ตัวอย่างของข้อมูลในรูป JSON FORM

- สังเกตและบันทึกผลลงในตารางที่ 4.5

**ตารางที่ 4.5** ตารางบันทึกผลการอัพเดตข้อมูลโดย Method PUT

|  |  |  |
| --- | --- | --- |
| **การกรอกข้อมูล** | **ผลลัพธ์** | **การแจ้งเตือนการทำงาน** |
| กรอกถูกต้อง | สามารถอัพเดตข้อมูลได้ | https://scontent.fbkk8-1.fna.fbcdn.net/v/t34.0-12/30019751_2019596328360595_1336968905_n.png?_nc_cat=0&oh=d232ea22ead706a1ec08fc50afca8c07&oe=5ACA8AE0 |
| กรอกไม่ครบหรือกรอกผิด | ไม่สามารถอัพเดตข้อมูลได้ |  |

3.4 การแก้ไขข้อมูลโดยการเรียกใช้ฟังก์ชัน DELETE

- ทำการรันเซิร์ฟเวอร์เพื่อทำการเรียกใช้ Path API

- เปิดโปรแกรม ARC เพื่อทดลองเพิ่มข้อมูลผู้ป่วยผ่านการ GET

- ทำการเลือก Method GET แล้วทำการใส่ localhost:3200/users/:userID

ลงไปในช่อง Request URL

- ในส่วนของ Body ให้พิมพ์เครื่องหมาย { } และกดปุ่ม send

- สังเกตและบันทึกผลลงในตารางที่ 4.6

**4.3.1.2 ผลการทดลอง**

จากการทดลองใช้งาน API ติดต่อรับส่งข้อมูลกับแอปพลิเคชันพบว่า ฟังก์ชัน POST สามารถใช้เพื่อเพิ่มข้อมูลลงไปยังระบบได้โดยข้อมูลจะถูกสร้างเป็น ID ขึ้นภายในฐานข้อมูล ฟังก์ชัน GET สามารถดึงข้อมูลตาม ID หรือคีย์ของข้อมูลที่ต้องการขึ้นมาแสดงได้โดย ฟังก์ชัน PUT สามารถแก้ไขข้อมูลเดิมโดยต้องมีการระบุ ID เพื่อระบุปลายทางที่ต้องการแก้ไขได้และฟังก์ชัน DELETE สามารถใช้ลบข้อมูลตาม ID ที่เลือกได้ โดยทั้ง 4 ฟังก์ชันจะทำงานได้ก็ต่อเมื่อ รูปแบบกรอกข้อมูลในลักษณะที่เป็น JSON ซึ่งมีคีย์ตรงกับโครงสร้างของฐานข้อมูล

**ตารางที่ 4.6** ตารางบันทึกผลการลบข้อมูลโดย Method DELETE

|  |  |  |
| --- | --- | --- |
| **การกรอกข้อมูล** | **ผลลัพธ์** | **การแจ้งเตือนการทำงาน** |
| กรอกถูกต้อง | ข้อมูลถูกลบออกจากฐานข้อมูล | https://scontent.fbkk8-1.fna.fbcdn.net/v/t34.0-12/30019751_2019596328360595_1336968905_n.png?_nc_cat=0&oh=d232ea22ead706a1ec08fc50afca8c07&oe=5ACA8AE0 |
| กรอกไม่ครบหรือกรอกผิด | ไม่สามารถลบข้อมูลออกจากฐานข้อมูลได้ |  |

**บทที่ 5**

**บทสรุป**

**5.1 สรุป**

ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยสร้างขึ้นเพื่อใช้เป็นฐานข้อมูลสำหรับจัดการกับข้อมูลที่มีการรับส่งจากแอปพลิเคชันแจ้งเตือนการรับประทานของผู้ป่วย

การทำงานของระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยแบ่งได้เป็น 2 ส่วนคือ ส่วนติดต่อกับผู้ใช้งานหรือหน้าเว็บแอปพลิเคชัน สร้างโดยใช้ภาษา HTML CSS และ AngularJS ประกอบไปด้วย หน้าเว็บเพจการเข้าสู่ระบบ ใช้ในการยืนยันตัวตนเพื่อเข้าใช้งานซึ่งแบ่งผู้ใช้งานได้เป็น 2 กลุ่มคือ ผู้ดูแลระบบและผู้ใช้งานทั่วไป หน้าเว็บเพจการตั้งค่าผู้ใช้งาน สำหรับการแก้ไข อัพเดต ลบ หรือใช้กำหนดสิทธิ์การเข้าถึงให้กับผู้ใช้งานโดยผู้ดูแล หน้าเว็บเพจแสดงข้อมูลของผู้ป่วย สำหรับแสดงข้อมูลผู้ป่วยที่อยู่ในฐานข้อมูล หน้าเว็บเพจสำหรับการแก้ไข อัพเดต ลบข้อมูลผู้ป่วยสำหรับจัดการกับข้อมูลผู้ป่วยในฐานข้อมูล และหน้าเว็บเพจสำหรับติดต่อผู้ดูแลระบบเพื่อเป็นช่องทางในการติดต่อแก้ไขปัญหาที่เกิดหรือให้คำแนะนำเพื่อนำไปปรับปรุงให้ระบบทำงานได้มีประสิทธิภาพยิ่งขึ้น ในส่วนประมวลผลและจัดการข้อมูล สร้างขึ้นโดยใช้ Node.js เป็นเซิร์ฟเวอร์และใช้ MongoDB เป็นฐานข้อมูล

การรับส่งข้อมูลระหว่างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยและแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยจะใช้ API เป็นตัวจัดการโดยใช้ฟังก์ชันพื้นฐานคือ ฟังก์ชัน GET สำหรับดึงข้อมูลผู้ป่วยในฐานข้อมูลไปใช้งาน ฟังก์ชัน POST สำหรับเพิ่มข้อมูลผู้ป่วยลงไปยังฐานข้อมูล ฟังก์ชัน PUT สำหรับการอัพเดตข้อมูลผู้ป่วย และฟังก์ชัน DELETE สำหรับการลบข้อมูลผู้ป่วยในฐานข้อมูล

จากการทดลองรับส่งข้อมูลระหว่างระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยและแอปพลิเคชันแจ้งเตือนการรับประทานยาของผู้ป่วยโดยใช้ API พบว่า สามารถรับส่งข้อมูลได้ซึ่งข้อมูลดังกล่าวจะถูกนำไปจัดการผ่านหน้าเว็บแอปพลิเคชันโดยการเรียกดูข้อมูล แก้ไขข้อมูล อัพเดตข้อมูลและลบข้อมูล ตามกลุ่มผู้ใช้งานซึ่งเป็นไปตามจุดประสงค์ที่กำหนดไว้

**5.2 ปัญหาและวิธีการแก้ไขปัญหา**

จากการสร้างและทดลองใช้ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย พบว่ามีปัญหาเกิดขึ้นหลายประการ ซึ่งสามารถสรุปได้ดังนี้

1. ฝั่งแอปพลิเคชันไม่สามารถนำข้อมูลจากฐานข้อมูลไปใช้งานได้เนื่องจากโครงสร้างของฐานข้อมูลที่สร้างไว้ไม่ตรงกับวิธีการดึงข้อมูลของแอปพลิเคชันคือ แอปพลิเคชันจะมีการดึงข้อมูลในรูปแบบของ JSON ที่เป็น Object และ Object แต่โครงสร้างข้อมูลที่สร้างไว้จะอยู่ในรูปแบบของ JSON ที่เป็น Object ตามด้วย Array

**วิธีการแก้ไข** ปรับเปลี่ยนโครงสร้างของฐานข้อมูลใหม่ให้เป็นรูปแบบของ JSON ที่เป็น Object และ Object ทั้งหมด

2. การบันทึกข้อมูลที่มีการแก้ไขหรืออัพเดตข้อมูลผู้ป่วยในฐานข้อมูลไม่สามารถบันทึกทับข้อมูลที่อยู่ใน ID ของผู้ป่วยที่ต้องการได้เพราะโครงสร้างของฐานข้อมูลเปลี่ยนไป

**วิธีการแก้ไข** เปลี่ยนวิธีการบันทึกของฟังก์ชันบันทึกข้อมูลให้สามารุเขียนทับโครงสร้างข้อมูลที่เป็น Object ตามด้วย Object ได้

3. การเพิ่มข้อมูลหรือจัดการกับข้อมูลในฐานข้อมูลบางตัวจากฝั่งแอปพลิเคชันไม่สามารถทำได้เพราะไม่มีคีย์สำหรับรองรับค่าของข้อมูลที่ต้องการ

**วิธีการแก้ไข** เพิ่มคีย์ที่สามารถรองรับการใช้งานในการเพิ่ม ลบ แก้ไข อัพเดต เรียกดูข้อมูลต่าง ๆของผู้ป่วยให้สามารถทำงานร่วมกันได้ทั้งทางฝั่งแอปพลิเคชันและฐานข้อมูล

**5.3 แนวทางการพัฒนา**

ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยควรมีการปรับแก้ไขให้สามารถทำงานได้มีประสิทธิภาพมากขึ้นดังนี้

1. ในส่วนการแสดงผลข้อมูลของผู้ป่วย เพื่อให้แสดงรายละเอียดและสามารถแก้ไขปรับปรุงข้อมูลได้รวดเร็วควรแก้ไขให้สามารถแสดงข้อมูลตามการค้นหาโดยระบุชื่อผู้ป่วยหรือคำที่สามารถระบุถึงผู้ป่วยคนนั้นได้

2. หน้าการจัดการข้อมูลของผู้ป่วย ในส่วนของยาควรพัฒนาให้มีการแสดงผลข้อมูลของยาที่อยู่ในระบบและเวลาในการกินยาตัวนั้นของผู้ป่วยให้สามารถเลือกเป็นรายการว่าผู้ป่วยคนนั้นรับประทานยาตัวนี้หรือยัง โดยข้อมูลนี้ต้องมีการอัพเดตเมื่อผู้ป่วยที่ใช้งานแอปพลิเคชันมีการติดต่อกับฐานข้อมูลในระบบ เพื่อความสะดวกต่อการจัดการกับข้อมูลได้ง่ายยิ่งขึ้น

3. ถ้ามีการนำระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วยไปใช้งานกับผู้ป่วยจริง ๆ ต้องมีการกำหนดให้ระบบสามารถจัดการกับข้อมูลของผู้ป่วยที่เข้ามาในแต่ละวันให้มีการบันทึกอัตโนมัติได้

**บรรณานุกรม**

จตุรพัชร์ พัฒนทรงศิวิลัย. **พัฒนาเว็บแอปพลิเคชันด้วย JavaScript**. กรุงเทพฯ : ซีเอ็ดยูเคชัน. 2559

AdaCode.IO. 2558. MEAN Stack คืออะไร?. [online]. Available :

<http://www.comscicafe.com/article/109/introduction-to-mean-tack#.Wsjo3IhubDc>

Adisak Tisanon. 2555. **การออกแบบระบบฐานข้อมูล**. [online]. Available :

http://kruoong.blogspot.com/2011/12/blog-post.html

Artak H.Andrey Smorkalov. n.d. **SQL vs. NoSQL Databases: What’s the Difference?.**

[online]. Available :

<https://www.upwork.com/hiring/data/sql-vs-nosql-databases-whats-the-ifference/>

[Banyapon Poolsawasd](http://www.daydev.com/author/nomad). 2558. **สร้าง Single Page Application ด้วย Node JSและExpress JS.**

[online]. Available :

<http://www.daydev.com/developer/single-page-application-node-js-and-express->

js.html

[Chai Phonbopit](https://www.facebook.com/phonbopit). 2558. **Node.js คืออะไร ? + เริ่มต้นใช้งาน Node.js**. [online]. Available :

<https://devahoy.com/posts/getting-started-with-nodejs/>

[Chai Phonbopit](https://www.facebook.com/phonbopit).2558. **MongoDB คืออะไร? + สอนวิธีใช้งานเบื้องต้น**. [online]. Available :

https://devahoy.com/posts/getting-started-with-nodejs/

Designil. n.d. **7 กฎการออกแบบ UI Design ให้สวยงาม สำหรับ Non-Designer**. [online].

Available : <https://www.designil.com/7-rules-beautiful-ui-design.html>

Kiattisak Anoochitarom. 2556. **Understanding Event-driven and Non-Blocking I/O in**

**Node.js**. [Online]. Available :  
http://www.macbaszii.com/2013/12/understanding-event-driven-and-non.html

Kunapat Pairat. 2560. **สร้าง RESTful API ด้วย Express-Express 101**. [online]. Available :

<https://medium.com/@aofleejay/restful-api-express-express/-101-ee37cc4952b4>

Monmai. 2560. **การออกแบบ UX-UI**. [online]. Available :

http://www.monmai.net/ux-ui/

Mospichit. 2559. **Front-End และ Back-End คืออะไร**. [online]. Available :

<https://blog.sogoodweb.com/Article/Detail/54053>

Patcharin Nammontree. n.d. **พื้นฐานการออกแบบฐานข้อมูล**. [online]. Available :

<https://sites.google.com/site/aridsaraonline/phun-than-kar-xxkbaeb-than-khxmul>

Somkiat Puisungnoen. 2559. **สรุปโครงสร้างข้อมูลของ NoSQL Database**. [online].

Available : http://www.somkiat.cc/types-of-nosql-database/

Somkiat Puisungnoen. 2554. **ทำความเข้าใจการออกแบบ schema ของ MongoDB**.

[Online]. Available : http://www.somkiat.cc/how-to-design-schema-of-mongodb/

TechTalkThai.2015. **รู้จักกับ SQL NoSQL และ NewSQL**. [Online]. Avalable :

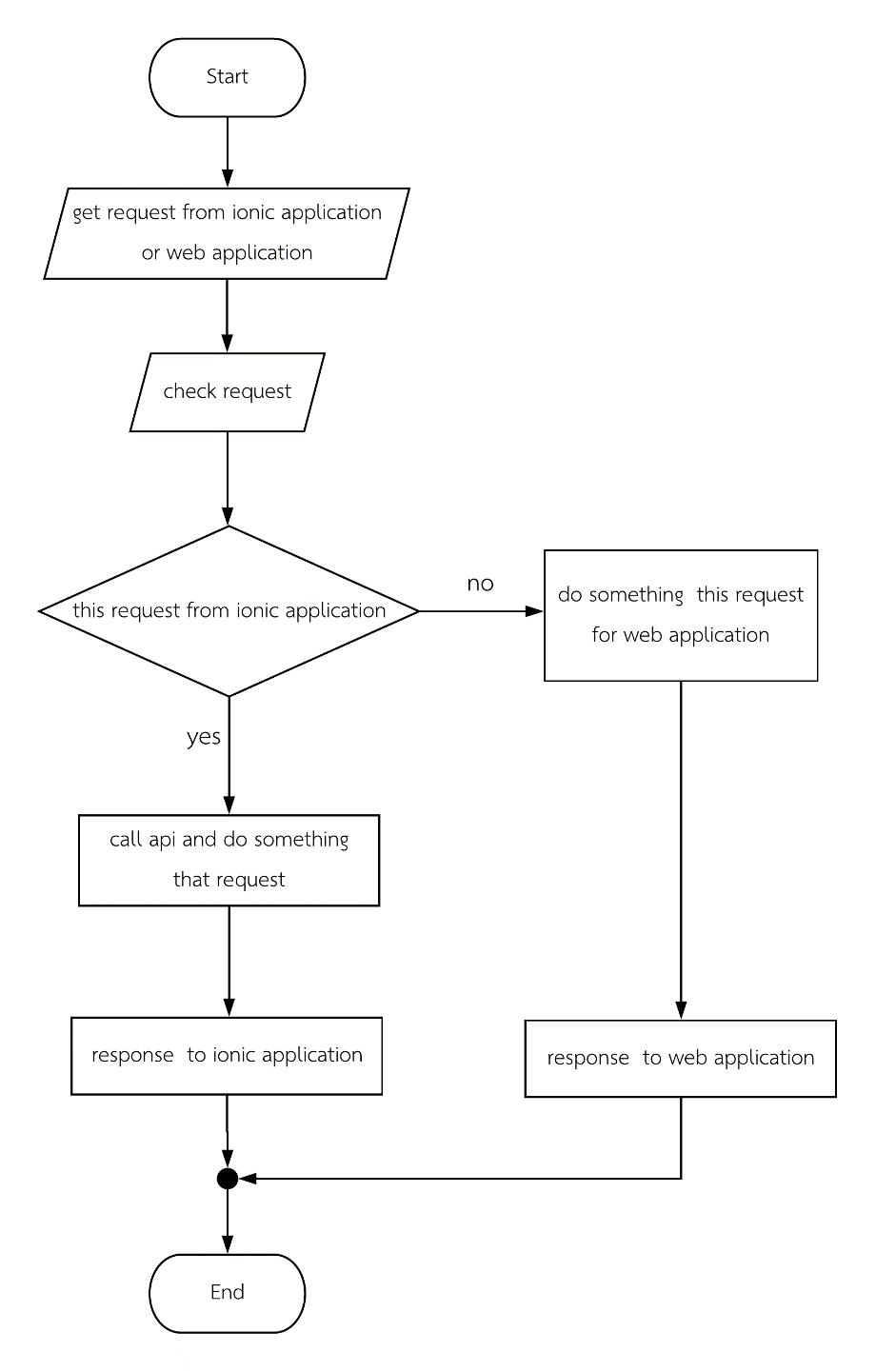
h<ttps://www.techtalkthai.com/introduce-sql-nosql-and-newsql-as-choices-of-d>atabase-technology/

Teerasej Jiraphatchandej. n.d. **ครบวงจร! รู้จักกับ MEAN: สร้างเว็บแอพด้วย JavarScript ครบ-**

**สูตร**. [Online]. Avalable : http://nextflow.in.th/2014/getting-start-mean/

**ภาคผนวก ก**

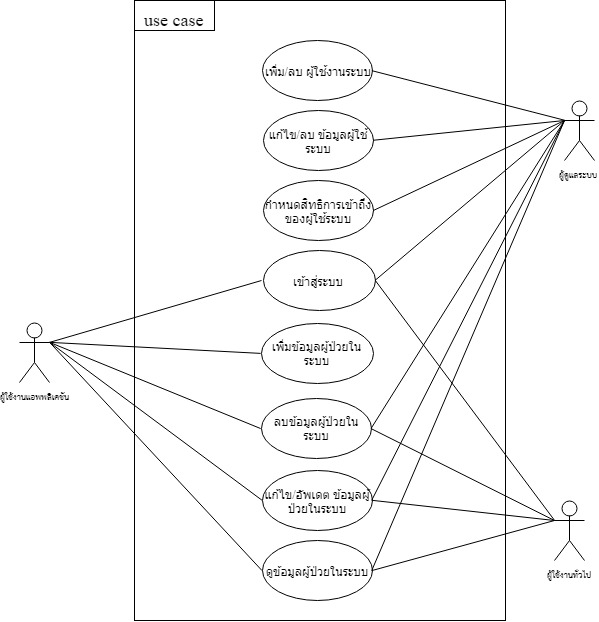
**ผังงาน**



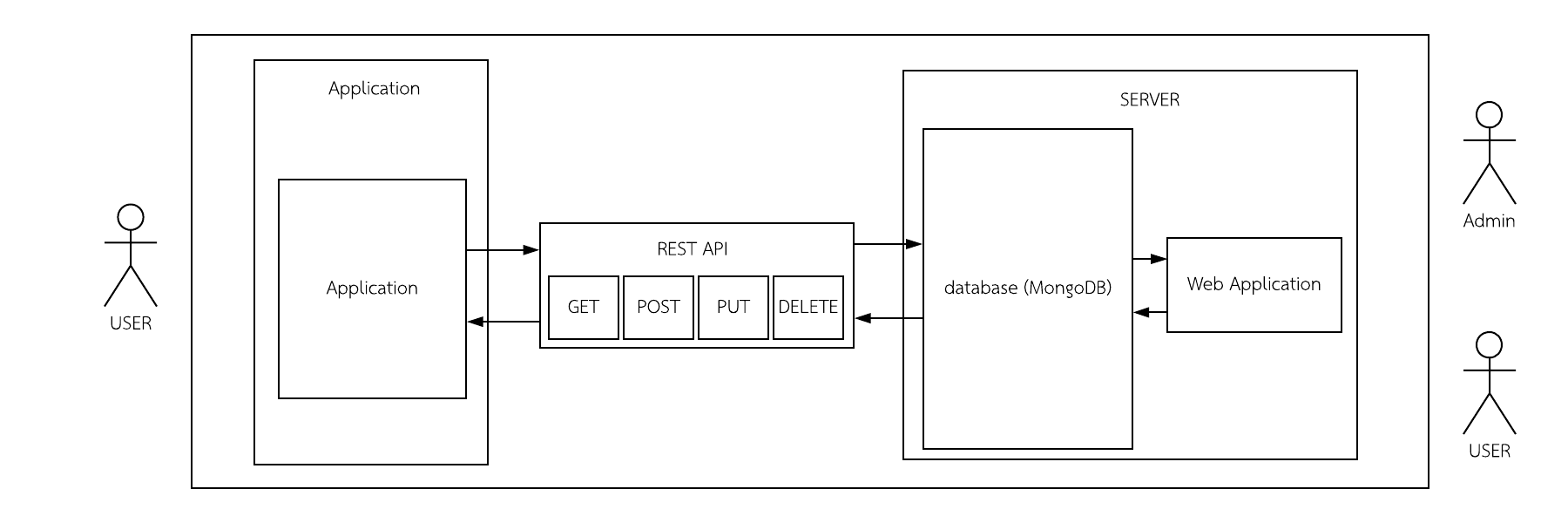
รูปที่ ก.1 ผังงานการทำงานของเซิร์ฟเวอร์กับผู้ใช้งานผ่านหน้าเว็บแอปพลิเคชัน

**ภาคผนวก ข**

**แผนผังการใช้งาน**



**รูปที่ ข.1** แผนผังภาพรวมการใช้งานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย



**รูปที่ ข.2** แผนผังการรับส่งข้อมูลระหว่างแอปพลิเคชันและเซิร์ฟเวอร์โดยการใช้งาน API

**ภาคผนวก ค**

**คู่มือการใช้งาน**

**คู่มือการใช้งาน**

**ระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย**

**( MRMS )**

**ภาควิชาครุศาสตร์วิศวกรรม**

**คณะครุศาสตร์อุตสาหกรรมและเทคโนโลยี**

**สถาบันเทคโนโลยีพระจอมแกล้าเจ้าคุณทหารลาดกระบัง**

**ปีการศึกษา 2560**

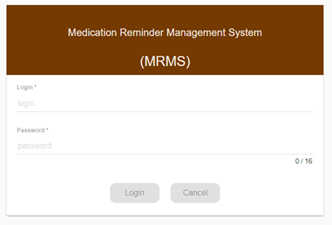
**การเข้าใช้งานระบบจัดการข้อมูลแจ้งเตือนการรับประทานยาของผู้ป่วย**

**1. การเข้าใช้งานสำหรับผู้ดูแลระบบ**

ผู้ดูแลระบบจะมีบทบาทในการเข้าถึงและจัดการข้อมูลต่าง ๆ ที่อยู่ในระบบตั้งแต่การเรียกดูข้อมูลผู้ป่วยในฐานข้อมูล การแก้ไขข้อมูลผู้ป่วย การอัพเดตข้อมูลผู้ป่วย การลบข้อมูลผู้ป่วย ตลอดจนการเพิ่มผู้ใช้งานและกำหนดสิทธิ์ให้กับผู้ใช้งานดังกล่าวซึ่งการทำงานต่าง ๆ มีขั้นตอนดังนี้

1.1 ทำการรันเซิร์ฟเวอร์

1.2 เข้าสู่หน้า Login โดยจะมีช่องสำหรับใส่ชื่อผู้ใช้และรหัสผ่านดังรูปที่ ค.1



**รูปที่ ค.1** ฟิลด์สำหรับกรอกข้อมูล login

1.3 ใส่ชื่อและรหัสผ่านของผู้ดูแลระบบในช่อง Login และ Password

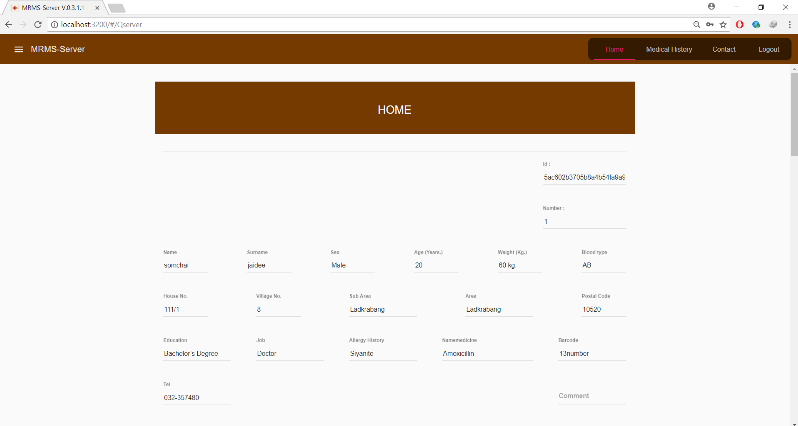
1.4 กดปุ่ม Login เพื่อเข้าสู่ระบบหรือ Cancel เพื่อลบข้อมูลทั้งหมดที่กรอกและใส่ข้อมูลใหม่

1.5 เมื่อกด Login จะเข้าสู่หน้า HOME สำหรับดูข้อมูล

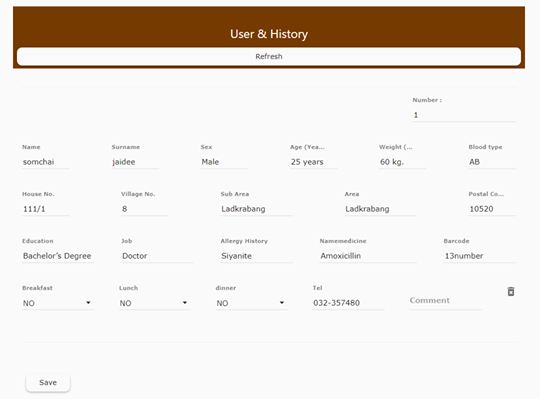
1.6 ผู้ใช้งานสามารถดูข้อมูลประวัติต่าง ๆ ทั้งหมดของผู้ป่วยได้ในหน้า HOME ดังรูปที่ ค.2

1.7 ในหน้า Medical History สามารถทำการการแก้ไขข้อมูลผู้ป่วยโดยข้อมูลจะถูกบันทึกลงไปในฐานข้อมูลเมื่อทำการกดปุ่ม save ดังรูปที่ ค.3

1.8 หน้า Contact เป็นหน้าหรับแจ้งปัญหาหรือให้คำแนะนำโดยการติดต่อผ่าน email หรือ facebook ที่ได้ทำการแจ้งไว้ ดังรูปที่ ค.4

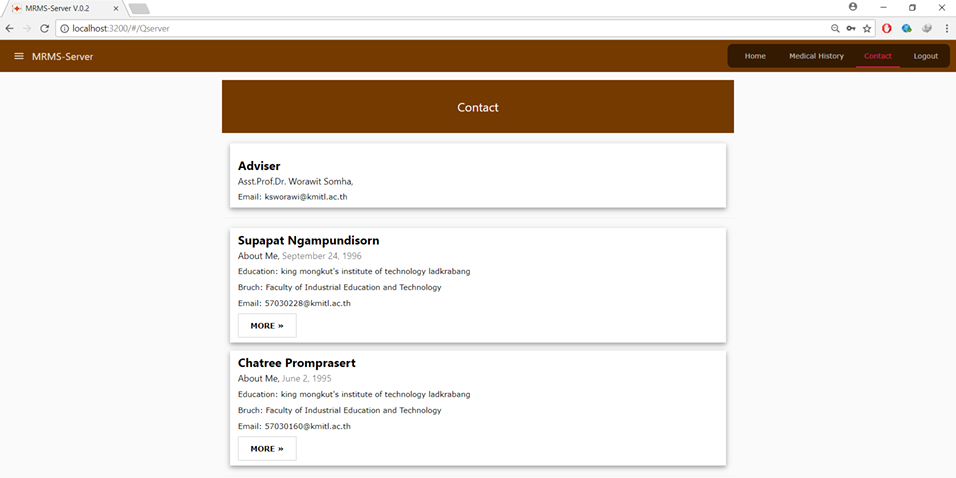


รูปที่ ค.2 หน้า HOME





**รูปที่ ค.3** การกด save ข้อมูลผู้ป่วย



**รูปที่ ค.4** หน้า Contact

1.9 ผู้ดูและระบบสามารถเพิ่มผู้ใช้งานระบบได้ โดยการกดที่ sidebar menu และเลือก Set User ดังรูปที่ ค.5





รูปที่ ค.5 การเข้าหน้าตั้งค่าผู้ใช้งาน

1.10 ในหัวข้อ Role ที่ Role name User นำเครื่องหมายถูกหน้าเมนูที่ไม่ต้องการให้ผู้ใช้งานได้ออกและกดไอคอนบันทึก

1.11 เพิ่มผู้ใช้งานโดยใส่ชื่อ-นามสกุล ในช่อง First name และ Last name

1.12 กำหนดรหัสผ่านและยืนยันรหัสผ่านให้กับผู้ใช้งาน ถ้าไม่กำหนดระบบจะทำการกำหนดให้อัตโนมัติเป็น 1234 จากนั้นกด save

1.13 ผู้ดูแลระบบและผู้ใช้งานสามารถออกจากระบบโดยการกดแถบเมนู logout

**2. การเข้าใช้งานสำหรับผู้ใช้งานทั่วไป**

การเข้าใช้งานสำหรับผู้ใช้ทั่วไปจะถูกกำหนดให้เข้าถึงได้เฉพาะบางหน้าเพจหรือเมนูเท่านั้นซึ่งขึ้นอยู่กับผู้ดูแลระบบจะเป็นผู้กำหนดให้เช่นไม่สามารถเข้าสู่เมนูตั้งค่าเพื่อเพิ่มผู้ใช้งานระบบโดยจะขึ้นเป็นสีเทาคือไม่สามารถกดได้ แต่มีการเข้าใช้งานหน้าเพจอื่น ๆ ได้ โดยมีการเข้าใช้งานดังนี้

2.1 ทำการรันเซิร์ฟเวอร์

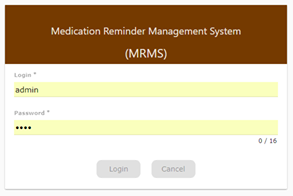
2.2 ใส่ชื่อและรหัสผ่านของผู้ดูแลระบบในช่อง Login และ Password ดังรูปที่ ค.6

2.3 กดปุ่ม Login เพื่อเข้าสู่ระบบหรือ Cancel เพื่อลบข้อมูลทั้งหมดที่กรอกและใส่ข้อมูลใหม่

2.4 เมื่อกด Login จะเข้าสู่หน้า HOME สำหรับดูข้อมูล

2.5 ผู้ใช้งานทั่วไปสามารถเข้าสู่แถบเมนูเพื่อไปยังหน้าเพจอื่นหรือ Logout ออกจากระบบได้

2.6 ในแถบ sidebar menu จะไม่สามารถกดได้โดยจะขึ้นเป็นสีเทาดังรูปที่ ค.7



**รูปที่ ค.6** การกรอกข้อมูลโดยใช้ชื่อและรหัสผ่านขงผู้ดูแลระบบ



**รูปที่ ค.7** เปรียบเทียบการเข้าใช้งานเมนูการตั้งค่าของผู้ดูแลและผู้ใช้ระบบทั่วไป

การใช้งาน API รับส่งข้อมูลผู้ป่วย

ในฝั่งของแอปพลิเคชัน เมื่อต้องการเพิ่ม ลบ แก้ไข อัพเดตข้อมูลของผู้ป่วยลงไปยังฐานข้อมูล สามารถทำได้โดยการเรียกใช้งาน Method ของ API ซึ่งการใช้งานมีดังนี้

**การใช้งาน Method POST**

1. ทำการรันเซิร์ฟเวอร์

2. เรียกใช้งาน Method POST ผ่าน URL localhost:3200/P1 โดย P1 เป็นชื่อของฐานข้อมูล

3. ในฝั่งของแอปพลิเคชันทำการเพิ่มข้อมูลตามรูปแบบ JSON FORM ของฐานข้อมูล

4. กด send เพื่อทำการเพิ่มข้อมูลลงในฐานข้อมูล

**การใช้งาน Method GET**

1. ทำการรันเซิร์ฟเวอร์

2. เรียกใช้งาน Method GET ผ่าน URL localhost:3200/G1 โดย G1 เป็นชื่อของฐานข้อมูล

3. ถ้าต้องการเพียงแค่ข้อมูลบางอย่างให้ทำการระบุ path/G2 ต่อจาก path/G1 โดย G2 หมายถึง ID ของผู้ใช้งานตามรูปตัวอย่างที่ ค. 8

4. กด send เพื่อทำการดึงข้อมูลที่ต้องการ



**รูปที่ ค.8** การเรียกใช้งาน Method GET

**การใช้งาน Method PUT**

1. ทำการรันเซิร์ฟเวอร์

2. เรียกใช้งาน Method PUT ผ่าน URL localhost:3200/PU1/PU2 โดย PU1 เป็นชื่อของฐานข้อมูลและ PU2 เป็น ID ที่ต้องการอัพเดตข้อมูล

3. ในฝั่งของแอปพลิเคชัน ทำการแก้ไขข้อมูลตามรูปแบบ JSON FORM ของฐานข้อมูล

4. กด send เพื่อทำการอัพเดตข้อมูลที่ต้องการลงไปยังฐานข้อมูล

**การใช้งาน Method DELETE**

1. ทำการรันเซิร์ฟเวอร์

2. เรียกใช้งาน Method DELETE ผ่าน URL localhost:3200/DE1/DE2 โดย DE1 เป็นชื่อของฐานข้อมูลและ DE2 เป็น ID ที่ต้องการลบข้อมูล

3. ในฝั่งของแอปพลิเคชัน ทำการแก้ไขข้อมูลตามรูปแบบ JSON FORM ของฐานข้อมูลโดยการพิมพ์เครื่องหมาย { }

4. กด send เพื่อทำการลบข้อมูลที่ต้องการออกจากฐานข้อมูล

**ประวัติผู้แต่ง**



**ชื่อ-สกุล** นาย ชาตรี พรหมประเสริฐ

**วัน เดือน ปีเกิด** 2 มิถุนายน พ.ศ. 2538

**ภูมิลำเนา** บ้านเลขที่ 42 หมู่ 6 ตำบลบ้านฆ้อง อำเภอโพธาราม จังหวัดราชบุรี 70120

**ประวัติการศึกษา**

ประถมศึกษา โรงเรียนสมศักดิ์วิทยา จังหวัดราชบุรี

มัธยมศึกษาตอนต้น โรงเรียนอนุบาลโพธาราม จังหวัดราชบุรี

มัธยมศึกษาตอนปลาย โรงเรียนโพธาวัฒนาเสนี จังหวัดราชบุรี

ปริญญาตรี ครุศาสตร์อุตสาหกรรมบัณฑิต (ครุศาสตร์วิศวกรรม)

สาขาวิชาครุศาสตร์วิศวกรรม

ครุศาสตร์อุตสาหกรรมและเทคโนโลยี สจล.

**คติพจน์** จงเดินหน้าต่อไป

**ประวัติผู้แต่ง**



**ชื่อ-สกุล** นาย ศุภภัทร งามพันธุ์ดิศร

**วัน เดือน ปีเกิด** 24 กันยายน พ.ศ. 2539

**ภูมิลำเนา** 146/1 หมู่ 8 ตำบลเขาสามยอด อำเภอเมือง

จังหวัดลพบุรี 15000

**ประวัติการศึกษา**

ประถมศึกษา โรงเรียนเมืองละโว้ จังหวัดลพบุรี

มัธยมศึกษาตอนต้น โรงเรียนอัสสัมชัญคอนแวนต์ จังหวัดลพบุรี

มัธยมศึกษาตอนปลาย โรงเรียนพิบูลวิทยาลัย จังหวัดลพบุรี

ปริญญาตรี ครุศาสตร์อุตสาหกรรมบัณฑิต (ครุศาสตร์วิศวกรรม)

สาขาวิชาครุศาสตร์วิศวกรรม

ครุศาสตร์อุตสาหกรรมและเทคโนโลยี สจล.

**คติพจน์** ความอดทนและความพยายามคือสิ่งที่นำไปสู่ความสำเร็จ