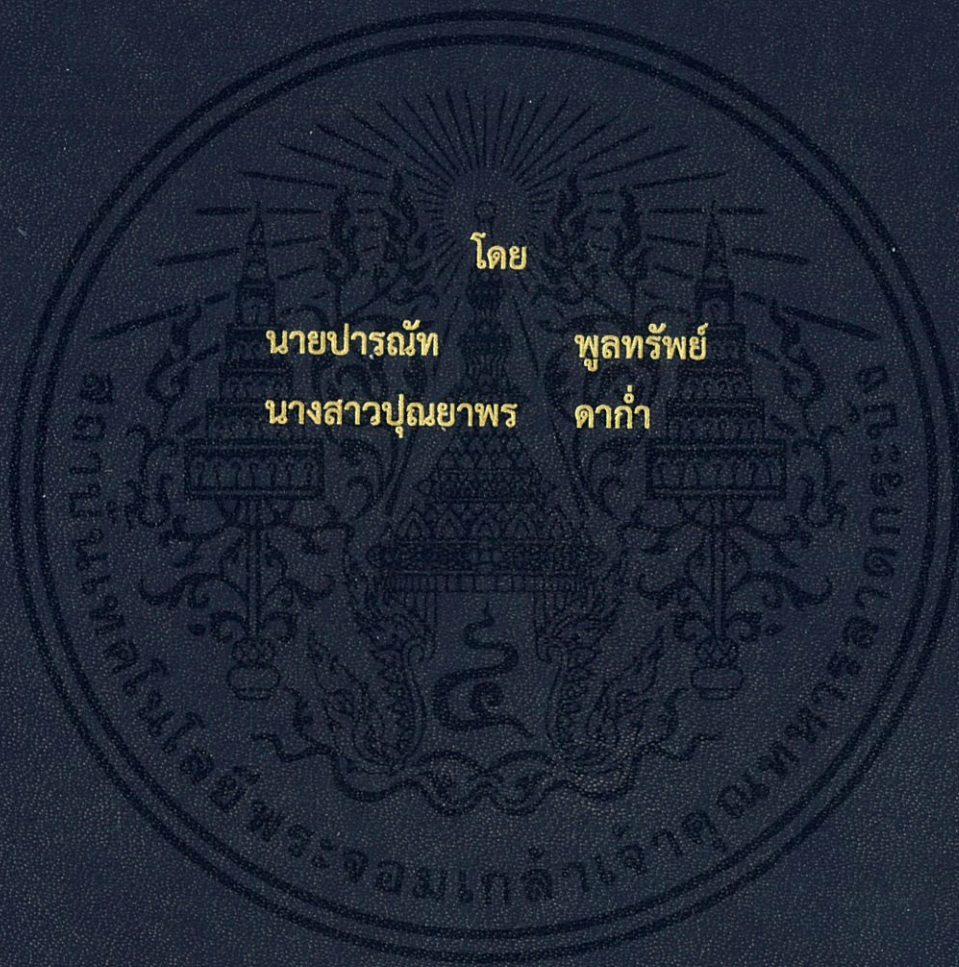


ระบบการยืนยันตัวตนการใช้อุปกรณ์ไฟฟ้าโดยการสั่งงานด้วยเสียง  
AUTHENTICATION SYSTEM CONTROLLED ELECTRICAL DEVICES  
BY VOICE CONTROL



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต  
ภาควิชาวิศวกรรมโทรคมนาคม  
คณะวิศวกรรมศาสตร์  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง  
ปีการศึกษา 2561

ระบบการยืนยันตัวตนการใช้อุปกรณ์ไฟฟ้าโดยการสั่งงานด้วยเสียง  
AUTHENTICATION SYSTEM CONTROLLED ELECTRICAL DEVICES  
BY VOICE CONTROL

โดย  
นายปารณัท พูลทรัพย์ 58010778  
นางสาวปุณยาพร ดากำ 58010800

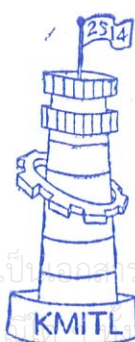
อาจารย์ที่ปรึกษา  
ดร. พีระเมศร์ โชติกวีกิจญาดา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต  
ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2561



ผ่านการตรวจรูปเล่มแล้ว

(.....)  
อาจารย์ที่ปรึกษา

22/5/62

วิศวกรรมโทรคมนาคม  
Telecommunications Engineering



ผ่านการตรวจชิ้นงานแล้ว

(.....)  
กรรมการผู้ตรวจชิ้นงาน

22/5/62

วิศวกรรมโทรคมนาคม  
Telecommunications Engineering

ปริญญานิพนธ์ปีการศึกษา 2561

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบการยืนยันตัวตนการใช้อุปกรณ์ไฟฟ้าโดยการสั่งงานด้วยเสียง

AUTHENTICATION SYSTEM CONTROLLED ELECTRICAL DEVICES BY VOICE  
CONTROL

ผู้จัดทำ

- |                        |          |
|------------------------|----------|
| 1. นายปารณัท พูลทรัพย์ | 58010778 |
| 2. นางสาวปณยาพร ดาก่ำ  | 58010800 |

.....  
(ดร. พีระเมศร์ โชติทวีกิจญาดา)

อาจารย์ที่ปรึกษา

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดี เนื่องมาจากการได้รับความอนุเคราะห์จาก ดร. พีระเมศร์ โชติทวีกิจญาติ อาจารย์ที่ปรึกษาปริญญานิพนธ์ ซึ่งท่านคอยติดตามความคืบหน้า ให้คำชี้แนะแนวความคิด และทักษะกระบวนการแก้ปัญหาต่างๆที่เกิดขึ้น รวมไปถึงความคิดเห็นซึ่งเป็นประโยชน์ต่อข้าพเจ้าอย่างยิ่ง และช่วยเพิ่มเติมทักษะในด้านต่างๆ ที่ข้าพเจ้าสามารถนำไปใช้ได้ในชีวิตประจำวัน ทั้งการทำงานเป็นกลุ่ม ฝึกการค้นคว้าหาข้อมูล และฝึกการแก้ปัญหาเฉพาะหน้าที่ข้าพเจ้าไม่คาดคิด นอกจากการชี้แนะทางทฤษฎี ยังสอนถึงการลงมือปฏิบัติในการใช้เครื่องมือต่างๆ ทางวิศวกรรม ทำให้การดำเนินการทำงานเป็นไปอย่างราบรื่นยิ่งขึ้น นอกจากนี้ขอขอบคุณพระคุณ บิดามารดา และครอบครัว ซึ่งเปิดโอกาสให้ได้รับการศึกษาเล่าเรียน ตลอดจนคอยช่วยเหลือและให้กำลังใจผู้จัดทำเสมอมาจนสำเร็จการศึกษา และขอขอบคุณเพื่อนในภาควิชาที่คอยให้ความช่วยเหลือในด้านต่างๆ ทั้งการปรึกษาปัญหาและการให้กำลังใจในการทำปริญญานิพนธ์ครั้งนี้

ท้ายที่สุดทางผู้จัดทำคาดหวังว่าปริญญานิพนธ์ฉบับนี้จะมีประโยชน์แก่ผู้ที่เข้ามาศึกษาหรือสนใจอยู่ไม่มากนักน้อย สำหรับข้อบกพร่องต่างๆ ที่อาจจะเกิดขึ้น ผู้จัดทำขออภัยและขอโทษผู้เดียว และยินดีที่จะรับฟังคำแนะนำจากทุกท่านที่ได้เข้ามาศึกษาหรือสนใจ เพื่อเป็นประโยชน์ในการพัฒนาปริญญานิพนธ์ต่อไป

ปารณัท พูลทรัพย์  
ปุณยาพร ดาก้า  
ผู้จัดทำ

ระบบการยืนยันตัวตนการใช้อุปกรณ์ไฟฟ้าโดยการสั่งงานด้วยเสียง

AUTHENTICATION SYSTEM CONTROLLED  
ELECTRICAL DEVICES BY VOICE CONTROL

โดย นายปารณัท พูลทรัพย์ 58010778

นางสาวปณยาพร ดาก่ำ 58010800

อาจารย์ที่ปรึกษา ดร. พิระเมศร์ โชติทวีกิจญาดา

### บทคัดย่อ

ในปัจจุบันมีการพัฒนาเทคโนโลยีสิ่งประดิษฐ์ต่างๆ เพื่อตอบสนองความต้องการและความสะดวกสบายของมนุษย์โดยเฉพาะด้านการดำเนินชีวิตประจำวัน ซึ่งโครงการนี้เป็นการนำเสนอระบบการยืนยันตัวตนสั่งงานอุปกรณ์ไฟฟ้าด้วยเสียงภายในที่พักอาศัย เพื่ออำนวยความสะดวกแก่ผู้ใช้งานและรักษาความปลอดภัยของระบบมากขึ้น โดยระบบจะทำการยืนยันตัวตนของผู้ใช้งาน ให้สามารถควบคุมอุปกรณ์ไฟฟ้าพื้นฐาน เช่น พัดลม, โทรทัศน์ และหลอดไฟ เป็นต้น พร้อมทั้งสามารถสั่งงานอุปกรณ์ไฟฟ้าที่อยู่ภายในแต่ละห้องของที่พักอาศัยด้วยเสียงผ่านไมโครโฟน จากนั้นไมโครคอนโทรลเลอร์จะวิเคราะห์คำสั่งเสียงแปลงเป็นตัวอักษร เพื่อประมวลผลเป็นคำสั่งในการทำงานอุปกรณ์ไฟฟ้า

### ABSTRACT

Nowadays, innovative technology has been developed to serve the needs of humans especially in daily life. This project will present voice controlled electrical devices authentication system in accommodation to facilitate the users and security of the system. The system will confirm the identity of the user to

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

control basic electrical devices such as fan, television and light bulbs, etc. It will operate voice controlled the electrical device inside each room of the accommodation through microphone. Then the microcontroller will convert speech to text to process orders in electrical equipment.



เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## สารบัญ

|                                                            | หน้า      |
|------------------------------------------------------------|-----------|
| กิตติกรรมประกาศ                                            | I         |
| บทคัดย่อ                                                   | II        |
| สารบัญ                                                     | IV        |
| สารบัญรูป                                                  | VI        |
| สารบัญตาราง                                                | VIII      |
| <br>                                                       |           |
| <b>บทที่ 1      บทนำ</b>                                   | <b>1</b>  |
| 1.1 ความเป็นมาและความสำคัญของปัญหา                         | 1         |
| 1.2 วัตถุประสงค์                                           | 1         |
| 1.3 ขอบเขตของโครงการ                                       | 2         |
| <br>                                                       |           |
| <b>บทที่ 2      ทฤษฎีและหลักการที่เกี่ยวข้อง</b>           | <b>3</b>  |
| 2.1 INTERNET OF THINGS                                     | 3         |
| 2.2 การควบคุมสถานะของอุปกรณ์ไฟฟ้า                          | 6         |
| 2.3 การแสดงผลการสั่งการและสถานะอุปกรณ์                     | 7         |
| 2.4 การแปลงเสียงเป็นตัวอักษรด้วย GOOGLE SPEECH RECOGNITION | 12        |
| 2.5 การยืนยันตัวตนด้วยอาร์เอฟไอดี                          | 13        |
| <br>                                                       |           |
| <b>บทที่ 3      การออกแบบและการจัดทำปริญญานิพนธ์</b>       | <b>20</b> |
| 3.1 การออกแบบ                                              | 20        |
| 3.2 เครื่องมือที่ใช้ในการทดลอง                             | 29        |
| 3.3 การจัดเก็บผลการทดลอง                                   | 35        |

## สารบัญ (ต่อ)

|                                                | หน้า |
|------------------------------------------------|------|
| บทที่ 4      ผลการทดลอง                        | 36   |
| 4.1 ผลการทดสอบระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง | 36   |
| 4.2 ผลการทดสอบระบบ GRAPHIC USER INTERFACE      | 43   |
| 4.3 ผลการทดสอบระบบยืนยันตัวตนด้วย RFID-RC522   | 47   |
| บทที่ 5      สรุปผลและข้อเสนอแนะ               | 52   |
| 5.1 สรุปผล                                     | 52   |
| 5.2 ข้อเสนอแนะ                                 | 52   |
| บรรณานุกรม                                     | 53   |
| ภาคผนวก      SOURCE CODE                       | 54   |

## สารบัญรูป

| รูปที่                                                                | หน้า |
|-----------------------------------------------------------------------|------|
| 2.1 การเชื่อมต่อของ WIRELESS SENSOR NETWORK                           | 3    |
| 2.2 การเชื่อมต่อ WSN NODES                                            | 4    |
| 2.3 Diagram อธิบายการเชื่อมต่อ Gateway หลายตัวเข้ากับ local network   | 5    |
| 2.4 การทำงานแต่ละ NETWORK LAYERS ของ IOT โดย IBM                      | 5    |
| 2.5 การเชื่อมต่อ Relay module กับอุปกรณ์ไฟฟ้า                         | 7    |
| 2.6 LAYER ของ GUI ในระบบ                                              | 8    |
| 2.7 โครงสร้างภาษา HTML                                                | 9    |
| 2.8 โครงสร้างของ CSS                                                  | 11   |
| 2.9 ส่วนประกอบของ RADIO FREQUENCY IDENTIFICATION                      | 14   |
| 2.10 โครงสร้างภายในป้าย (TAG)                                         | 15   |
| 2.11 หลักการทำงานของป้ายในย่านความถี่ต่ำและสูงกับ UHF                 | 16   |
| 2.12 ภาพแสดงถึงการจัดลำดับมาตรฐาน RFID                                | 17   |
| 2.13 ภาพแสดงการเชื่อมต่อแบบ SINGLE ระหว่าง MASTER กับ SLAVE           | 18   |
| 2.14 การตั้งค่าฮาร์ดแวร์โดยใช้ซีพรีจิสเตอร์เพื่อสร้างบัพเฟอร์แบบวงกลม | 19   |
| 3.1 บล็อกไดอะแกรมระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง                     | 20   |
| 3.2 แผนภาพการทำงานของระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงรองรับภาษาอังกฤษ | 21   |
| 3.3 แผนภาพการทำงานของระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงรองรับภาษาไทย    | 22   |
| 3.4 วงจรระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง                              | 23   |
| 3.5 แผนภาพการทำงานของระบบ GRAPHIC USER INTERFACE                      | 24   |
| 3.6 แผนภาพการทำงานของ GRAPHIC USER INTERFACE ควบคุมอุปกรณ์ไฟฟ้าโดยตรง | 25   |
| 3.7 แผนภาพการทำงานของระบบยืนยันตัวตนด้วย RFID                         | 27   |

## สารบัญรูป (ต่อ)

| รูปที่                                                    | หน้า |
|-----------------------------------------------------------|------|
| 3.8 วงจรระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงแบบระบุตัวตน      | 28   |
| 3.9 แผนภาพแสดงการเรียกใช้ระบบคำสั่งเสียง                  | 28   |
| 3.10 แผนภาพแสดงการเรียกใช้ระบบ GUI                        | 28   |
| 3.11 RASPBERRY PI 3 MODEL B                               | 29   |
| 3.12 ARDUINO UNO BOARD                                    | 30   |
| 3.13 ไมโครโฟน รุ่น MIC-01 ยี่ห้อ SCORPION                 | 30   |
| 3.14 หลอดไฟไส้ร้อนแบบธรรมดา                               | 31   |
| 3.15 พัดลม                                                | 31   |
| 3.16 สภาวะการทำงานของ RELAY MODULE                        | 33   |
| 3.17 รีเลย์ 4 ช่อง (4-CHANNEL RELAY MODULE 5V)            | 33   |
| 3.18 เครื่องอ่านสัญญาณ RFID โมดูล RC522                   | 34   |
| 3.19 ป้ายอิเล็กทรอนิกส์ (RFID TAG)                        | 34   |
| 4.1 ข้อความที่ระบบได้รับคำว่า “TURN OFF BEDROOM LIGHT”    | 36   |
| 4.2 ข้อความที่ระบบได้รับคำว่า “TURN ON BEDROOM LIGHT”     | 36   |
| 4.3 ข้อความที่ระบบได้รับคำว่า “TURN OFF KITCHEN ROOM FAN” | 37   |
| 4.4 ข้อความที่ระบบได้รับคำว่า “TURN ON KITCHEN ROOM FAN”  | 37   |
| 4.5 ข้อความที่ระบบได้รับคำว่า “เปิดไฟห้องนอน”             | 39   |
| 4.6 ข้อความที่ระบบได้รับคำว่า “ปิดไฟห้องนอน”              | 39   |
| 4.7 ข้อความที่ระบบได้รับคำว่า “เปิดพัดลมห้องครัว”         | 39   |
| 4.8 ข้อความที่ระบบได้รับคำว่า “ปิดพัดลมห้องครัว”          | 40   |
| 4.9 แบบแผนห้องทำการทดลองและตำแหน่งไมโครโฟน                | 41   |
| 4.10 คำสั่งเปิดหลอดไฟ “TURN ON”                           | 42   |
| 4.11 คำสั่งปิดหลอดไฟ “TURN OFF”                           | 42   |

## สารบัญรูป (ต่อ)

| รูปที่                                                           | หน้า |
|------------------------------------------------------------------|------|
| 4.12 GUI ควบคุมอุปกรณ์ไฟฟ้าโดยตรง                                | 43   |
| 4.13 GUI สั่งเปิดไฟห้องนั่งเล่นและห้องน้ำ                        | 44   |
| 4.14 GUI สั่งปิดไฟห้องนั่งเล่นและห้องน้ำ                         | 44   |
| 4.15 GUI แสดงสถานะของอุปกรณ์ไฟฟ้า                                | 45   |
| 4.16 GUI แสดงการเปลี่ยนสถานะอุปกรณ์ไฟฟ้าจากคำสั่งเสียงภาษาอังกฤษ | 46   |
| 4.17 GUI แสดงการเปลี่ยนสถานะอุปกรณ์ไฟฟ้าจากคำสั่งเสียงภาษาไทย    | 46   |
| 4.18 ข้อมูล RFID Tag และเครื่องอ่านสัญญาณ RFID                   | 47   |
| 4.19 การใช้งานห้องนั่งเล่น                                       | 48   |
| 4.20 อุปกรณ์ไฟฟ้าในห้องนั่งเล่น                                  | 48   |
| 4.21 การใช้งานห้องน้ำ                                            | 49   |
| 4.22 อุปกรณ์ไฟฟ้าในห้องน้ำ                                       | 49   |
| 4.23 การใช้งานห้องนอน                                            | 50   |
| 4.24 อุปกรณ์ไฟฟ้าห้องนอน                                         | 50   |
| 4.25 การใช้งานห้องครัว                                           | 51   |
| 4.26 อุปกรณ์ไฟฟ้าห้องครัว                                        | 51   |

## สารบัญตาราง

| ตารางที่                                                        | หน้า |
|-----------------------------------------------------------------|------|
| 2.1 ความแตกต่างระหว่างการเชื่อมต่อของ INTERNET OF THING         | 4    |
| 2.2 คำสั่งต่างๆ ของ HTML                                        | 10   |
| 3.1 ข้อมูลรายละเอียด PIN ของรีเลย์มอดูล 4 ช่อง                  | 25   |
| 3.2 โครงสร้างของรีเลย์มอดูล                                     | 26   |
| 4.1 ผลการทดสอบความถูกต้องในการแปลงเสียงพูดภาษาอังกฤษเป็นข้อความ | 38   |
| 4.2 ผลการทดสอบความถูกต้องในการแปลงเสียงพูดภาษาไทยเป็นข้อความ    | 40   |



## บทที่ 1

### บทนำ

#### 1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบัน internet มีผลต่อการดำเนินชีวิต ระบบปฏิบัติการบ้านอัจฉริยะ จึงเริ่มมีบทบาทต่อการดำเนินชีวิต โดยการเปลี่ยนแปลงทั้งหมดนี้เพื่ออำนวยความสะดวกสบายแก่ผู้ใช้งาน และสามารถบริหารจัดการกับอุปกรณ์ภายในบ้านได้อย่างครบถ้วน และยังสามารถตรวจสอบสถานะของเครื่องใช้ไฟฟ้าภายในบ้าน

ในปฏิญานพนธ์นี้ได้นำเสนอ การควบคุมการเปิด-ปิด ของอุปกรณ์ไฟฟ้าโดยการสั่งการด้วยเสียง ทำให้มีความรวดเร็วกว่าเดิม ที่ต้องเปิดหรือปิดที่สวิตซ์ไฟฟ้าด้วยตัวเอง โดยระบบนี้เพียงแค่ออกคำสั่งผ่านไมโครโฟนรับสัญญาณเสียงไป อุปกรณ์ไฟฟ้าที่ถูกสั่งการนั้นก็จะ เปิดหรือปิดตามความต้องการของเรา ทั้งนี้ยังมีการใช้งาน การยืนยันตัวตนด้วย RFID (Radio Frequency - Identification) เพื่อรักษาความปลอดภัยในการสั่งการ และ ระบุตัวตนของผู้สั่งการอีกด้วย

#### 1.2 วัตถุประสงค์

- 1) เพื่อศึกษาระบบการยืนยันตัวตนและการทำงานการสั่งการเครื่องใช้ไฟฟ้าภายในบ้านด้วยเสียง
- 2) เพื่อกำหนดตัวบุคคลที่อนุญาตให้สามารถควบคุม การเปิด-ปิดของอุปกรณ์ไฟฟ้าภายในบ้าน
- 3) เพื่อนำระบบที่ศึกษามาประยุกต์ใช้กับเครื่องใช้ไฟฟ้าภายในบ้าน
- 4) เพื่อแสดงสถานะ การทำงานของอุปกรณ์ไฟฟ้าต่างๆได้
- 5) เพื่ออำนวยความสะดวกสบายและพัฒนาการสั่งการของระบบเครื่องใช้ไฟฟ้าด้วยเสียง

#### 1.3 ขอบเขตของปฏิญานพนธ์

ในปฏิญานพนธ์นี้ได้นำเสนอระบบเปิด-ปิดอุปกรณ์ไฟฟ้าโดยการสั่งงานด้วยเสียงแบ่งออกได้เป็น 3 ขั้นตอนหลักในการจำแนกๆ ได้แก่

- 1) ออกแบบระบบการยืนยันตัวตนการสั่งงานเครื่องใช้ไฟฟ้าแต่ละห้องภายในที่พักอาศัย ซึ่งในกาออกแบบนั้นเราจะออกแบบทั้งระบบตัวสั่งการอุปกรณ์ไฟฟ้า และ ออกแบบโมเดลบ้าน เพื่อจำลองการทำงานของระบบ การสั่งการด้วยเสียง โดยเราจะแบ่งเป็นห้องต่างๆและมีอุปกรณ์ไฟฟ้า

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประจำแต่ละห้องอีกด้วย โดยในขั้นตอนนี้จะยังสั่งการโดยการเขียน HTML ขึ้นมาเพื่อสร้าง GUI เป็นหน้าแสดงสถานะและหน้าควบคุมการทำงาน เพื่อควบคุมการเปิด-ปิดของอุปกรณ์ ไฟฟ้า ซึ่งการควบคุมนั้น เราจะใช้ Relay Module ทำหน้าที่ เป็นตัวควบคุมการเปิด-ปิดของอุปกรณ์ผ่านการสั่งการผ่านโปรแกรม

2) พัฒนาระบบที่ออกแบบนั้นให้สามารถสั่งการด้วยเสียงได้ โดยการเพิ่มขั้นตอนในการรับเสียงพูดสั่งการของผู้ใช้งาน และนำเสียงพูดนั้นไปประมวลเป็นตัวอักษรด้วย Google Assistant เพื่อนำมาประมวลผลต่อในบอร์ด Raspberry Pi ซึ่งทำให้สามารถสั่งการอุปกรณ์ไฟฟ้าที่ทดลองด้วยเสียง และสั่งงานผ่านโปรแกรมที่สร้างมาข้างต้นได้ ทำให้สามารถเลือกควบคุมตามความสะดวกของเรา โดยมีให้เลือกควบคุมทั้งแบบใช้เสียงสั่งการและแบบโปรแกรมสั่งการ

3) เพิ่มความปลอดภัยให้กับระบบของเราที่ออกแบบโดยใช้ การระบุตัวตนแบบ RFID (Radio Frequency Identification) เพื่อป้องกันการเข้าใช้งานโดยไม่ได้รับอนุญาตและเพื่อความปลอดภัยในการใช้งานโดยเหตุผลหลักที่เลือกใช้การยืนยันตัวตนแบบ RFID เพราะว่าใช้ภายในบ้าน ซึ่งมีพื้นที่น้อย และเหมาะกับการใช้งานกับโครงงานอีกด้วย

## บทที่ 2

### ทฤษฎีและหลักการที่เกี่ยวข้อง

#### 2.1 Internet of Things

แนวคิด Internet of Things ถูกคิดขึ้นโดย Kevin Ashton ในปี 1999 โดยเริ่มต้นโครงการ Auto-ID Center จากเทคโนโลยี RFID Sensors ต่างๆที่จะเชื่อมต่อกันได้ ในช่วงปี ค.ศ. 2000 ได้มีอุปกรณ์อิเล็กทรอนิกส์ออกมาและใช้คำว่า Smart กับอุปกรณ์ต่างๆที่ผลิตขึ้นมาเป็น smart device, smart home, transportation เป็นต้น โดยอุปกรณ์จะมีโครงสร้างพื้นฐานที่สามารถเชื่อมต่อกับโลกอินเทอร์เน็ตได้ จนกลายเป็นแนวคิดที่อุปกรณ์เหล่านั้นสามารถสื่อสารกันได้ โดยอาศัยตัว Sensor สามารถสรุปแนวคิดนี้ได้คือ นอกจาก Smart devices ต่างๆจะเชื่อมต่ออินเทอร์เน็ตได้แล้วมันยังสามารถเชื่อมต่อไปยังอุปกรณ์ตัวอื่นได้

##### 2.1.1 A wireless sensor network

ตัวแปรสำคัญสำหรับ Internet of Things ไม่ใช่มีเพียง Internet network เท่านั้น ยังมีตัวแปรอื่นเข้ามาเกี่ยวข้องคือ Sensor node จำนวนมากที่ทำให้เกิด wireless sensor network (WSN) ทำให้อุปกรณ์ต่างๆสามารถเชื่อมต่อถึง WSN ดังรูปที่ 2.1 และตัว WSN สามารถตรวจจับ physical phenomena ในเครือข่ายที่เชื่อมต่อกับ WSN ยกอย่างคือ แสง อุณหภูมิ ความดัน เป็นต้น เพื่อรับค่าที่ตรวจจับได้ไปยังอุปกรณ์ในระบบ



รูปที่ 2.1 การเชื่อมต่อของ Wireless Sensor Network

ที่มา : <http://www.vedvil.com/news/internet-of-things-iot/>

2.1.2 การเชื่อมต่อของ internet of things

Access technology คือการเชื่อมต่อของ IoT เป็นการพัฒนาไปพร้อมกับทาง ส่วนของ Hardware คือ processors, radio และ sensors จนสามารถพัฒนาและกลายเป็น SoC (system on a chip) และยังพัฒนาส่วนซอฟต์แวร์ ไปพร้อมกัน โดยการเชื่อมต่อสำหรับ internet of things มี 3 การเชื่อมต่อหลัก คือ

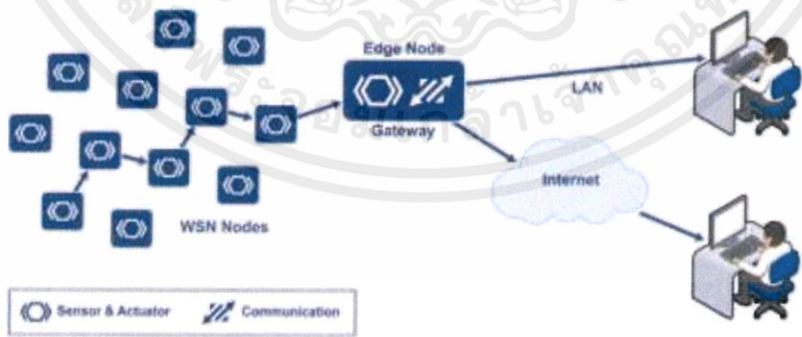
- 1. Bluetooth 4.0
- 2. IEEE 802.15.4e
- 3. WLAN IEEE 802.11 หรือ Wi-Fi

ตารางที่ 2.1 ความแตกต่างระหว่างการเชื่อมต่อของ internet of thing

|           | IEEE 802.15.4e      | Bluetooth | WLAN IEEE 802.11 |
|-----------|---------------------|-----------|------------------|
| Frequency | 868/915 MHz 2.4 GHz | 2.4 GHz   | 2.4,5.8 GHz      |
| Data Rate | 250 Kbps            | 723 Kbps  | 11-105 Mbps      |
| Power     | Very low            | Low       | High             |

2.1.3 Gateway Sensor Nodes

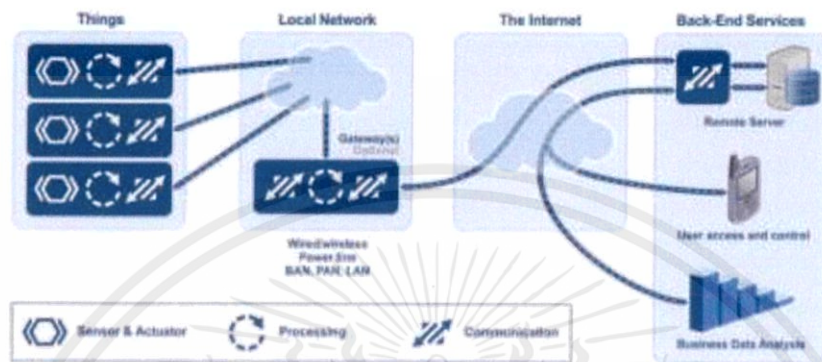
เมื่อมีโครงข่าย Sensor nodes ดังรูป 2.2 จำเป็นจะต้องมี Gateway Sensor Nodes เพื่อจะเชื่อมต่อไปยังโลกอินเทอร์เน็ตด้วย โดยตัว Gateway นี้จะทำหน้าที่เชื่อมต่อไปยัง Internet ให้อุปกรณ์ทั้งหมดในโครงข่าย Sensor nodes ทั้งหมดส่งข้อมูลเข้าสู่อินเทอร์เน็ต



รูปที่ 2.2 การเชื่อมต่อ WSN Nodes

ที่มา : <http://www.veedvil.com/news/internet-of-things-iot/>

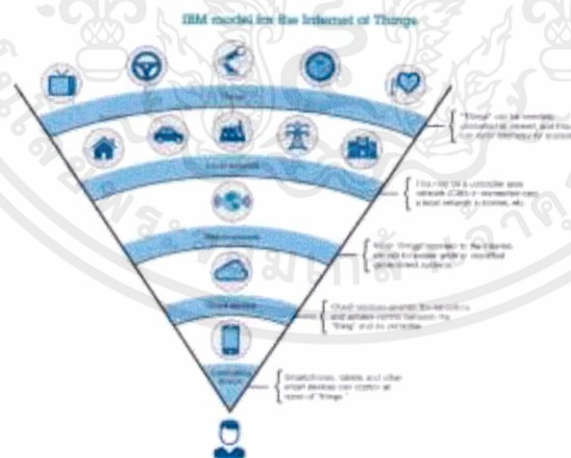
Gateway จะอยู่ภายใน Local network ซึ่งจะมีการกำหนดกันต่อไปว่า Gateway ภายใน Local network จะให้เชื่อมต่อไปยังอินเทอร์เน็ต หรืออุปกรณ์ที่เชื่อมเข้ามาใน Gateway ก็อาจจะสื่อสารกันได้เฉพาะภายใน Local network ดังรูป 2.3



รูปที่ 2.3 Diagram อธิบายการเชื่อมต่อ Gateway หลายตัวเข้ากับ local network  
ที่มา : <http://www.veedvil.com/news/internet-of-things-iot/>

#### 2.1.4 กลุ่มของ Internet of Things

สามารถจำแนกประเภทได้ 2 ประเภท ดังรูปที่ 2.4 เพื่อแบ่งการทำงานตามชั้น คือ เซิงพาณิชย์และเชิงอุตสาหกรรม



รูปที่ 2.4 การทำงานแต่ละ Network Layers ของ IoT โดย IBM  
ที่มา : <http://www.veedvil.com/news/internet-of-things-iot/>

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเชิงพาณิชย์ได้จำแนกเทคโนโลยีจาก local network ที่มีหลายเทคโนโลยีที่แตกต่างกันใน Sensor nodes โดยตัวอุปกรณ์ Internet of Thing ในกลุ่มนี้จะเชื่อมต่อแบบเครือข่ายแอดเดรสเพื่อเข้าสู่อินเทอร์เน็ต

ส่วนในภาคอุตสาหกรรมแบ่งจาก local communication ที่เป็นบลูทูธหรืออีเทอร์เน็ตทั้งแบบมีสายหรือไร้สาย โดยอุปกรณ์ Internet of Thing ในกลุ่มนี้จะสื่อสารภายในกลุ่ม Sensor nodes เดียวกันเท่านั้นหรือเป็นแบบ local devices เพียงอย่างเดียวอาจไม่ได้เชื่อมสู่อินเทอร์เน็ต

## 2.2 การควบคุมสถานะของอุปกรณ์ไฟฟ้า

### 2.2.1 รีเลย์โมดูล (Relay module)

เป็นอุปกรณ์ที่เปลี่ยนพลังงานไฟฟ้าให้เป็นพลังงานแม่เหล็ก เพื่อใช้ในการดึงดูดหน้าสัมผัสของคอนแทคให้เปลี่ยนสถานะ โดยการป้อนกระแสไฟฟ้าให้กับขดลวด เพื่อทำการปิดหรือเปิดหน้าสัมผัสเช่นเดียวกับสวิตช์อิเล็กทรอนิกส์

รีเลย์ประกอบด้วยส่วนสำคัญ 2 ส่วนหลักคือ

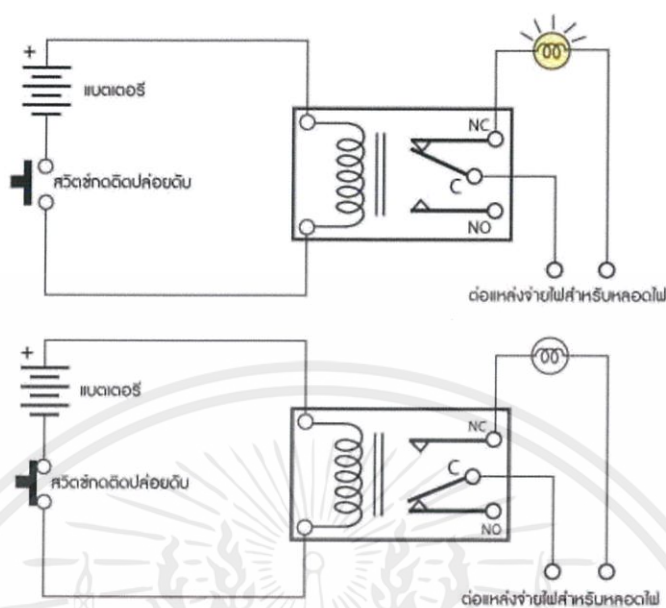
#### 2.2.1.1 ส่วนของขดลวด (coil)

ทำหน้าที่สร้างสนามแม่เหล็กไฟฟ้าให้แกนโลหะไปกระตุ้นให้หน้าสัมผัสต่อกัน ทำงานโดยการรับแรงดันจากภายนอกต่อคร่อมที่ขดลวดเหนี่ยวนำนี้ เมื่อขดลวดได้รับแรงดัน (ค่าแรงดันที่รีเลย์ต้องการขึ้นกับชนิดและรุ่นตามที่คุณผลิตกำหนด) จะเกิดสนามแม่เหล็กไฟฟ้าทำให้แกนโลหะด้านในไปกระตุ้นให้แผ่นหน้าสัมผัสต่อกัน

#### 2.2.1.2 ส่วนของหน้าสัมผัส (contact)

ทำหน้าที่เป็นสวิตช์กำหนดสถานะอุปกรณ์ จุดต่อใช้งานมาตรฐานของ Relay module ดังรูป 2.5

- 1) จุดต่อ NC (normal close) โดยปกติสถานะจุดนี้จะปิดตลอดการทำงาน
- 2) จุดต่อ NO (normal open) เป็นจุดที่ open circuit โดยปกติ
- 3) จุด C (common) เป็นจุดต่อร่วมแหล่งจ่ายกระแสไฟฟ้า



รูปที่ 2.5 การเชื่อมต่อ Relay module กับอุปกรณ์ไฟฟ้า

ที่มา : <http://www.psptech.co.th/>

## 2.3 การแสดงผลการสั่งการ และสถานะอุปกรณ์

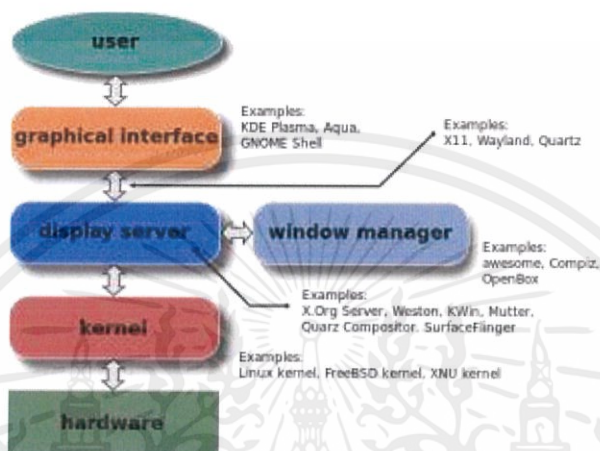
### 2.3.1 Graphical User Interface (GUI)

เป็นรูปแบบของการอินเทอร์เฟซสำหรับผู้ใช้ที่ช่วยให้ผู้ใช้สามารถโต้ตอบกับอุปกรณ์อิเล็กทรอนิกส์ ผ่านทางไอคอนกราฟิกและตัวชี้วัดภาพ เช่น สัญกรณ์รอก แทนการใช้ อินเทอร์เฟซ ซึ่งต้องใช้คำสั่งที่พิมพ์ลงบนแป้นพิมพ์คอมพิวเตอร์โดยการสั่งการ GUI จะดำเนินการ ผ่านการจัดการโดยตรงขององค์ประกอบกราฟฟิคภายใน GUI

#### 2.3.1.1 ส่วนการติดต่อผู้ใช้และการออกแบบโต้กลับ

การออกแบบองค์ประกอบภาพของ GUI เป็นส่วนสำคัญของการเขียน โปรแกรมประยุกต์ซอฟต์แวร์ในด้านปฏิสัมพันธ์ของมนุษย์กับคอมพิวเตอร์ เป้าหมายคือการเพิ่ม ประสิทธิภาพและความสะดวกในการใช้สำหรับการออกแบบตรรกะพื้นฐานของโปรแกรมที่เก็บไว้ และออกแบบที่ทำให้ใช้งานได้ มีการใช้วิธีการออกแบบโดยเน้นผู้ใช้เป็นหลักเพื่อให้แน่ใจว่าภาษา ที่นำมาใช้ในการ ออกแบบนั้นได้รับการออกแบบให้เหมาะสมกับงาน คุณลักษณะการเชื่อมต่อแบบ กราฟฟิคมองเห็นได้ของโปรแกรมบางครั้งเรียกว่าโครเมี่ยมหรือ GUI โดยปกติผู้ใช้จะมีปฏิสัมพันธ์กับ

ข้อมูลโดยการจัดการภาพซึ่งจะช่วยให้การโต้ตอบเหมาะสมกับชนิดของข้อมูลที่ต้นถือไว้เครื่องมือของอินเทอร์เฟซที่ออกแบบมาอย่างดีได้รับเลือกเพื่อสนับสนุนการดำเนินการที่จำเป็นเพื่อให้บรรลุเป้าหมายของผู้ใช้ ตัวควบคุมมุมมองแบบจำลองช่วยให้โครงสร้างแบบยืดหยุ่นซึ่งอินเทอร์เฟซเป็นอิสระและเชื่อมโยงกับฟังก์ชันแอปพลิเคชันโดยอ้อมเพื่อให้ GUI ดังรูปที่ 2.6



รูปที่ 2.6 Layer ของ GUI ในระบบ

ที่มา : [https://en.wikipedia.org/wiki/Graphical\\_user\\_interface](https://en.wikipedia.org/wiki/Graphical_user_interface)

### 2.3.2 Flask

เป็นเฟรมเวิร์ก (Framework) ที่เขียนขึ้นมาสำหรับการใช้งานในภาษา Python เพื่อใช้ในการสร้างเว็บไซต์ โดยจะทำให้ภาษา Python นั้นมีความสามารถในการจัดการกับเว็บไซต์ ซึ่งมีความสามารถคล้ายภาษา PHP Hypertext Preprocessor

โดยเฟรมเวิร์ก (Framework) หรือโครงสร้างซอฟต์แวร์เป็นขอบเขตของระบบงานที่เป็นรูปแบบที่น่ากลับมาใช้ใหม่ได้สำหรับระบบซอฟต์แวร์ ซึ่งสามารถอยู่ในรูปของคลาสนามธรรม (Abstract class) และวิธีการใช้ตัวตน (Instance) ของคลาสร่วมกันจำเพาะสำหรับซอฟต์แวร์ โครงสร้างซอฟต์แวร์ทุกโครงสร้างใช้การออกแบบเชิงวัตถุ (Object Oriented Programming)

### 2.3.3 Hyper Text Markup Language (HTML)

เป็นภาษาคอมพิวเตอร์ที่ใช้ในการแสดงผลของเอกสารบนเว็บไซต์หรือเว็บเพจ ถูกพัฒนาและกำหนดมาตรฐานโดยองค์กร World Wide Web Consortium (W3C) และจากการ

พัฒนาทางด้านซอฟต์แวร์ของ Microsoft ทำให้ภาษา HTML เป็นอีกภาษาที่ใช้เขียนโปรแกรม (HTML Application) ซึ่งจะให้ข้อมูลแสดงผลบนเว็บเบราว์เซอร์ โดยดึงข้อมูลผ่านอินเทอร์เน็ต ในรูปแบบของข้อความ รูปภาพ เสียง เป็นต้น ไฟล์ที่บันทึกจะมีนามสกุล .html หรือ .htm

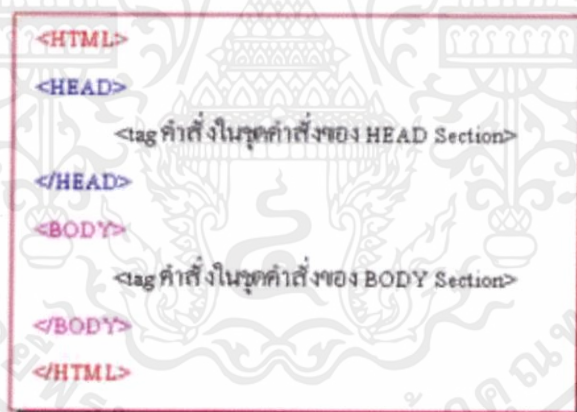
### 2.3.3.1 โครงสร้างของภาษา HTML

โครงสร้างของภาษา HTML จะประกอบด้วย 2 ส่วน ดังนี้

1) Head คือส่วนที่เป็นหัว (Header) ของหน้าเอกสารทั่วไปหรือส่วนชื่อเรื่อง (Title) ของหน้าต่างการทำงานในระบบ

2) Body คือส่วนเนื้อหาของเอกสารนั้น ซึ่งจะประกอบด้วยแท็กคำสั่งในการจัดรูปแบบหรือตกแต่งเอกสาร HTML

แท็กหรือคำสั่งที่ใช้ในภาษา HTML ประกอบด้วยเครื่องหมาย “<”, ชื่อคำสั่ง และเครื่องหมาย “>” เป็นส่วนที่ทำหน้าที่ตกแต่งข้อความเพื่อการแสดงผลข้อมูล โดยทั่วไปจะเป็นคำสั่งเปิดและปิด ซึ่งคำสั่งปิดของแต่ละคำสั่งจะมีรูปแบบเหมือนคำสั่งเปิดเพียงแต่จะมีเครื่องหมาย “/” นำหน้าคำสั่ง คำสั่งพื้นฐานของการเขียนภาษา HTML มีดังรูป 2.7



รูปที่ 2.7 โครงสร้างภาษา HTML

คำสั่ง <HTML> เป็นคำสั่งเริ่มต้นในการเขียนโปรแกรมและคำสั่ง </HTML> เป็นคำสั่งสิ้นสุดโปรแกรม HTML คำสั่งนี้จะไม่แสดงผลในโปรแกรมเว็บเบราว์เซอร์แต่ต้องเขียนเพื่อให้เกิดความเป็นระบบของงาน

คำสั่ง <HEAD> เป็นส่วนที่ใช้อธิบายเกี่ยวกับข้อมูลเฉพาะของหน้าเว็บ เช่น ชื่อเรื่องของหน้าเว็บ (Title), ชื่อผู้จัดทำเว็บ (Author), คำสำหรับการค้นหา (Keyword)

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสั่ง <BODY> เป็นส่วนเนื้อหาหลักของหน้าเว็บและเป็นส่วนการทำงานหลักของหน้าเว็บ ซึ่งการแสดงผลจะต้องใช้แท็กจำนวนมากขึ้นอยู่กับลักษณะของข้อมูลที่ต้องการนำเสนอ เช่น ข้อความ, รูปภาพ, เสียง, วิดีโอ หรือไฟล์ต่างๆ เป็นต้น

2.3.3.2 คำสั่งของ HTML

คำสั่งของ HTML เป็นส่วนที่ทำหน้าที่ตกแต่งข้อความ เพื่อการแสดงผลข้อมูล ซึ่งจะประกอบด้วยคำสั่งต่างๆ ดังตารางที่ 2.2

ตาราง 2.2 คำสั่งต่างๆ ของ HTML

| คำสั่งพื้นฐาน                                  | การใช้งาน                                       |
|------------------------------------------------|-------------------------------------------------|
| <br>                                           | ขึ้นบรรทัดใหม่                                  |
| <p> ข้อความ </p>                               | ย่อหน้าใหม่                                     |
| <CENTER> ข้อความ </CENTER>                     | จัดให้ข้อความอยู่กึ่งกลาง                       |
| คำสั่งรูปแบบตัวอักษร                           | การใช้งาน                                       |
| <B> ข้อความ </B>                               | ตัวอักษรหนา                                     |
| <I> ข้อความ </I>                               | ตัวอักษรเอน                                     |
| <U> ข้อความ </U>                               | ขีดเส้นใต้ตัวอักษร                              |
| <font size = “ตัวเลข”> ข้อความ </font>         | ขนาดตัวอักษร                                    |
| <font color = “ชื่อสี”> ข้อความ </font>        | สีตัวอักษร                                      |
| <font face = “รูปแบบ”> ข้อความ </font>         | รูปแบบตัวอักษร                                  |
| คำสั่งจุดเชื่อมโยงข้อมูล                       | การใช้งาน                                       |
| <a href = “ชื่อไฟล์.html”> ข้อมูล </a>         | สร้างลิงค์ไปยังเอกสาร ชื่อไฟล์.html             |
| <a href = “ลิงค์”> ข้อมูล </a>                 | สร้างลิงค์ไปยังเว็บไซต์อื่น                     |
| <a href = “ลิงค์”target = “blank”> ข้อมูล </a> | สร้างลิงค์ไปยังเว็บไซต์อื่น และเปิดหน้าต่างใหม่ |
| <a href = “ลิงค์”> <img src = “ชื่อgif”> </a>  | สร้างลิงค์โดยใช้รูปภาพ ชื่อ.gif เป็นตัวเชื่อม   |

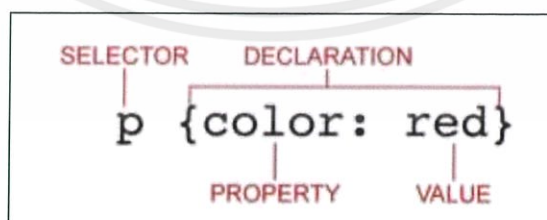
### 2.3.4 Cascading Style Sheet (CSS)

เป็นภาษาที่ใช้เป็นส่วนของการจัดรูปแบบการแสดงผลเอกสาร HTML โดยกำหนดเกณฑ์ในการระบุรูปแบบของเนื้อหาในเอกสาร ได้แก่ สีของข้อความ สีพื้นหลัง ประเภทตัวอักษร และการจัดวางข้อความ ซึ่งการกำหนดรูปแบบนี้ใช้หลักการของการแยกเนื้อหาเอกสาร HTML ออกจากคำสั่งที่ใช้ในการจัดรูปแบบการแสดงผล กำหนดรูปแบบของการแสดงผลเอกสารไม่ขึ้นอยู่กับเนื้อหาของเอกสาร เพื่อให้ง่ายต่อการจัดรูปแบบการแสดงผลลัพท์ของ HTML โดยเฉพาะในกรณีที่มีการเปลี่ยนแปลงเนื้อหาเอกสารบ่อยครั้งหรือต้องการควบคุมให้รูปแบบการแสดงผลเอกสาร HTML มีลักษณะของความสม่ำเสมอทั่วกันทุกหน้าเอกสารภายในเว็บไซต์เดียวกัน โดยกฎเกณฑ์ในการกำหนดรูปแบบเอกสาร HTML ถูกเพิ่มเข้ามาครั้งแรกใน HTML 4.0 เมื่อปีพ.ศ. 2539 ในรูปแบบของ CSS level 1 Recommendations ที่กำหนดโดยองค์กร World Wide Web Consortium (W3C)

#### 2.3.4.1 โครงสร้างและรูปแบบคำสั่งของ CSS

รูปแบบคำสั่งของ CSS ดังรูปที่ 2.8 จะประกอบไปด้วย 3 ส่วน ได้แก่

- 1) Selector เป็นส่วน HTML tag ต่างๆ ที่เราต้องการจะให้แสดงผล เช่น `<table>`, `<body>` และ `<p>` เป็นต้น นอกจากนั้นยังสามารถใช้เป็นแบบ Class name ได้
- 2) Property เป็นส่วนของคุณสมบัติในการจัดรูปแบบการแสดงผลที่เราต้องการให้แสดง เช่น Color สำหรับกำหนดสี, font-size สำหรับกำหนดขนาดตัวอักษร เป็นต้น
- 3) Value เป็นค่าที่เราต้องการกำหนดให้กับ Property ในการแสดงผล เช่น ตัวอักษรสีแดง ตัวอักษรแบบ Tahoma, ตัวอักษรขนาด 12 pt เป็นต้น



รูปที่ 2.8 โครงสร้างของ CSS

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## 2.4 การแปลงเสียงเป็นตัวอักษรด้วย Google Speech Recognition API

### 2.4.1 หลักการทำงานของระบบคำสั่งภาษาไทย

Speech recognition เป็นเทคโนโลยีที่ทำให้คอมพิวเตอร์จำเสียงมนุษย์ เพื่อแปลงเสียงมนุษย์เป็นคำสั่งให้คอมพิวเตอร์เข้าใจ การพัฒนาเทคโนโลยี Speech recognition ด้วย Google Speech Recognition API จะรองรับภาษาไทย ใช้ภาษา Python ในการพัฒนาโดย อาศัยโมดูล Speech Recognition (BSD license) รองรับทั้ง Python2, Python3 และรองรับทั้ง Linux, Mac OS และ Window หลักการทำงานของระบบรับเสียงจากไมโครโฟนซึ่งเป็นอินพุตของระบบ นำไปประมวลผลบน Cloud ซึ่งแปลงเสียงพูดเป็นข้อความภาษาไทย จึงนำค่าที่ได้มาเปรียบเทียบกับ คำสั่งในระบบ เพื่อสั่งให้อุปกรณ์ไฟฟ้าทำงาน

#### ขั้นตอนและวิธีการของระบบ

โดยปกติในการสร้าง Speech Recognition จะมีขั้นตอนการปฏิบัติอยู่ ทั้งหมด 4 ขั้นตอน ซึ่งจะพยายามอธิบายโดยสรุปได้เป็น 4 ขั้นตอน

ขั้นที่ 1 แปลงคลื่นเสียงที่มากกระทบในขั้นต้นให้เป็นตัวเลขที่เราต้องการ เพื่อทำความเข้าใจ

ขั้นที่ 2 คำนวณถึงลักษณะเฉพาะซึ่งเป็นสัญลักษณ์ของ Spectral โดย domain ที่เป็นหัวเรื่องของ Speech ลักษณะ เหล่านี้จะถูกคำนวณทุกๆ 10 msec โดยแต่ละ 10 msec จะถูกเรียกว่า “Frame”

ขั้นที่ 3 Artificial Neural Network (ANN) แบบ Multi - Layer Perceptron (MLP) จะถูกใช้เพื่อแยกชั้นของกลุ่มของลักษณะเหล่านั้นไปสู่ phonetic-based categories ในแต่ละ frame

ขั้นที่ 4 Viterbi search จะทำการจับคู่ neural network output scores กับคำศัพท์ที่ต้องการ โดย Viterbi Search ใช้กำหนดถึงความต่อเนื่องของหน่วยพื้นฐานของเสียง จากความเป็นไปได้สูงสุด โดยคำนวณผ่าน Gaussian Mixture Model ข้อดีของ Viterbi Search คือการประมวลผลข้อมูลเป็นแบบ Real time แต่ข้อเสียคือระบบจะเลือกตัด Path ที่ หนาแน่น ต่ำ เกินกว่ากำหนดไปและไม่นำกลับมาพิจารณาอีกเลย ทั้งๆ ที่บางครั้ง Path นั้นอาจจะมีค่าความ น่าจะเป็นรวมสูงขึ้นกว่า Path อื่นเมื่อ Search ถึง Word ท้ายของประโยค

## 2.5 การยืนยันตัวตนด้วยอาร์เอฟไอดี

### 2.5.1 ประวัติความเป็นมาอาร์เอฟไอดี

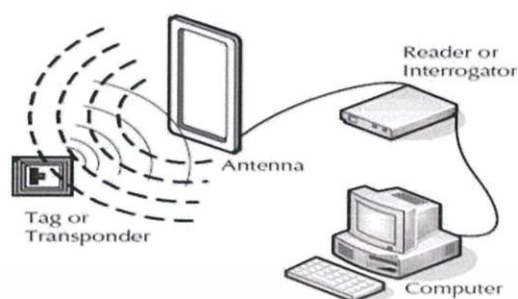
RFID (Radio Frequency Identification) ถูกพัฒนามาในยุคสงครามโลกครั้งที่สอง ซึ่งประเทศในกลุ่มพันธมิตร และกลุ่มอักษะได้มีการใช้เรดาร์ซึ่งถูกค้นพบโดย เซอร์โรเบิร์ต อเล็กซานเดอร์ วัตสัน-วัตต์ ในปี ค.ศ. 1935 ใช้ในการตรวจจับและเตือนเครื่องบินที่กำลังเข้ามา แต่ปัญหาของการใช้เรดาร์ในยุคนั้นคือไม่สามารถแยกแยะระหว่างเครื่องบินรบว่าเป็นของฝ่ายไหน ทางฝั่งเยอรมัน ได้ค้นพบว่าเมื่อนักบินหมุนตัวแล้ว จะทำให้มีการสะท้อนสัญญาณเรดาร์ที่เปลี่ยนไป จึงทำให้ทราบว่าเป็นเครื่องบินที่เข้ามาเป็นฝ่ายเยอรมัน ซึ่งเป็นจุดกำเนิดของอาร์เอฟไอดีแบบที่ขึ้นอยู่กับคุณสมบัติการสะท้อนคลื่นวิทยุ โดยไม่อาศัยเครื่องส่งวิทยุ (แบบพาสซีฟ)

ยุคแรกเริ่มของการใช้อาร์เอฟไอดีในเชิงพาณิชย์ได้แก่ระบบกันขโมยในห้างสรรพสินค้า ซึ่งสินค้าจะมีการติด RFID แบบ 1 บิต ซึ่งจะมีค่าเป็น '0' หรือ '1' เมื่อสินค้ามีการชำระเงิน ตัวบิต ก็จะถูกตั้งค่าเป็น '0' ทำให้สามารถนำออกจากร้านได้ ในกรณีที่ไม่มีชำระสินค้า เมื่อผ่านประตูเครื่องตรวจป้ายกันขโมย เมื่ออ่านค่าจากวัตถุในถุงของลูกค้ามีค่าเป็น '1' จะมีสัญญาณเตือนดังขึ้น

อาร์เอฟไอดีในย่านความถี่สูงยิ่ง UHF ได้รับความนิยมอีกครั้งในปี ค.ศ. 1999 เมื่อหน่วยงาน UCC (Uniform Code Council) หน่วยงาน EAN International บริษัท Procter & Gamble และบริษัท Gillette ได้ร่วมกันก่อตั้งศูนย์ Auto-ID ขึ้นที่สถาบันเทคโนโลยีแมสซาชูเซตส์ (MIT) ประเทศสหรัฐอเมริกาเพื่อพัฒนาแนวทางการใช้อาร์เอฟไอดีในห่วงโซ่อุปทาน (Supply Chain) ในช่วงปี ค.ศ. 1999 – 2003 Auto-ID ได้รับการสนับสนุนจากบริษัทเอกชนจำนวนมาก และได้มีการขยายศูนย์ Auto-ID ไปยังประเทศออสเตรเลีย อังกฤษ สวิตเซอร์แลนด์ ญี่ปุ่น และจีน ได้มีการพัฒนามาตรฐานใหม่เรียกว่า รหัสสินค้าอิเล็กทรอนิกส์

### 2.5.2 ส่วนประกอบของระบบอาร์เอฟไอดี

ในระบบอาร์เอฟไอดีมีองค์ประกอบหลักสองส่วน ส่วนแรกคือ ทรานสพอนเดอร์หรือ แท็ก ที่ใช้ติดกับวัตถุต่างๆที่เราต้องการ โดยป้ายนั้นจะประกอบด้วยสายอากาศและไมโครชิป ที่มีการบันทึกหมายเลข (ID) หรือข้อมูลเกี่ยวกับวัตถุชิ้นนั้นๆ ส่วนที่สอง คือ เครื่องอ่าน/เขียนข้อมูลภายในแท็ก เรียกว่า Reader ทำหน้าที่เชื่อมต่อเพื่ออ่านหรือเขียนข้อมูลลงในป้ายด้วยสัญญาณความถี่วิทยุ ดังรูปที่ 2.9



รูปที่ 2.9 ส่วนประกอบของ Radio Frequency Identification

ที่มา : <https://www.thaieasyelec.com/article-wiki/basic-electronics/rfid-basic.html>

### 2.5.2.1 ทรานสปอนเดอร์ (แท็ก)

ทำหน้าที่ส่งสัญญาณหรือข้อมูลที่บันทึกอยู่ในแท็ก โดยจะตอบสนองไปที่ตัวอ่านข้อมูล การสื่อสารระหว่างแท็กและตัวอ่านข้อมูลจะเป็นการสื่อสารกันโดยอาศัยช่องความถี่วิทยุผ่านอากาศ ซึ่งโครงสร้างภายในของป้าย (Tag) ดังรูปที่ 2.10 ประกอบด้วย 2 ส่วนหลัก ได้แก่

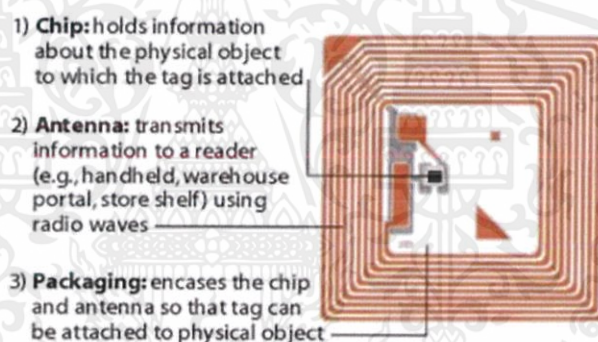
- 1) ไมโครชิป (Microchip) จะทำหน้าที่เก็บข้อมูลของวัตถุในหน่วยความจำ ซึ่งในหน่วยความจำอาจจะเป็นแบบอ่านอย่างเดียว (ROM) หรือทั้งอ่านและเขียน (RAM) ทั้งนี้ขึ้นอยู่กับความต้องการในการนำไปใช้งาน โดยปกติหน่วยความจำแบบ ROM จะเก็บข้อมูลด้วยความปลอดภัย เช่น สิทธิในการเข้าออกประตู ส่วน RAM ใช้เก็บข้อมูลชั่วคราวในระหว่างที่แท็กและเครื่องอ่านทำการติดต่อสื่อสารกัน

นอกจาก ROM และ RAM ยังมีหน่วยความจำแบบ EEPROM ที่ใช้ในการเก็บข้อมูลการสื่อสารระหว่างแท็กและเครื่องอ่าน รวมถึงข้อมูลยังคงอยู่ ถึงแม้จะไม่มีพลังงานไฟฟ้าป้อนให้แก่แท็ก

ภายในไมโครชิปยังมีส่วนที่เป็นแหล่งพลังงาน ซึ่งมีหน้าที่ในการแปลงไฟฟ้ากระแสสลับจากเสาอากาศของเครื่องอ่านมาเป็นไฟฟ้ากระแสตรง ส่วนที่ทำหน้าที่ในการแปลงสัญญาณสัญญาณ เรียกว่า Modulator ทำหน้าที่แปลงสัญญาณจากเครื่องอ่านและส่งข้อมูลกลับให้เครื่องอ่าน และส่วนลอจิก (Logic) ทำหน้าที่ในการกำหนดโปรโตคอลในการสื่อสารข้อมูลระหว่างเครื่องอ่านกับป้าย

2) เสาอากาศ (Antenna) เป็นขดลวดขนาดเล็กที่ทำหน้าที่เป็นเสาอากาศสำหรับรับหรือส่งสัญญาณคลื่นความถี่วิทยุ และสร้างพลังงานป้อนให้กับไมโครชิป เสาอากาศจะแผ่สัญญาณวิทยุจำนวนหนึ่งออกมา เพื่อกระตุ้นให้ป้ายอ่านหรือเขียนข้อมูลลงไป

เสาอากาศสามารถมีได้หลากหลายขนาดและรูปร่าง เพื่อให้เหมาะสมกับวัตถุที่จะนำไปติดตั้งและเพื่อให้เกิดประสิทธิภาพสูงสุดในการรับหรือส่งสัญญาณคลื่นความถี่วิทยุ เสาอากาศจะถูกติดไปโดยตรงกับตัวรับส่งสัญญาณให้เป็นอุปกรณ์ติดกัน เสาอากาศของป้าย RFID มีขนาดใหญ่กว่าชิปอย่างมาก ดังนั้นจะเห็นได้ว่าการออกแบบเสาอากาศของป้ายเป็นปัจจัยสำคัญมาก เนื่องจากมีผลต่อระยะการอ่านและมุมในการอ่าน ปัจจัยที่เกี่ยวข้องกับการออกแบบเสาอากาศมีหลายปัจจัย เช่น ระยะการอ่านระหว่างป้ายกับเครื่องอ่าน มุมในการอ่านระหว่างป้ายกับเครื่องอ่าน วัสดุที่ใช้ในการทำ ความเร็วในการอ่าน สภาพแวดล้อม และลักษณะเสาอากาศของเครื่องอ่าน เป็นต้น



รูปที่ 2.10 โครงสร้างภายในป้าย (Tag)

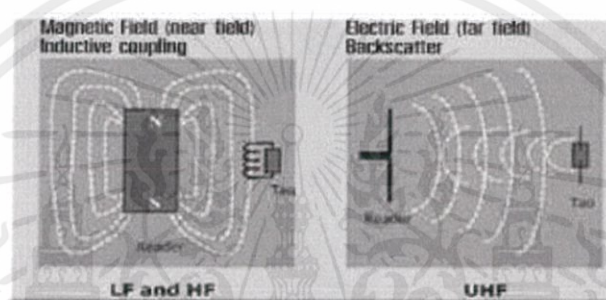
ที่มา : <https://www.thaieasyelec.com/article-wiki/basic-electronics/rfid-basic.html>

ประเภทของป้าย (Tag) จะมีความแตกต่างกันตามการใช้งาน โครงสร้างและหลักการทำงานของแต่ละประเภท โดยส่วนมากแบ่งเป็นออกเป็น 3 ประเภท ดังนี้

1) ป้ายแบบแพสซีฟ ป้ายชนิดนี้ทำงานได้โดยไม่ต้องใช้แหล่งจ่ายไฟจากภายนอก เนื่องจากทำงานในย่านคลื่นความถี่ต่ำและสูงจะใช้หลักการคู่ควบแบบเหนี่ยวนำ (Inductive coupling) ซึ่งเกิดจากการอยู่ใกล้กันของขดลวดจากเครื่องอ่านที่กำลังทำงานและสายอากาศของป้าย ทำให้เกิดการถ่ายเทพลังงานจากเครื่องอ่านไปยังป้ายผ่านสนามแม่เหล็กไฟฟ้าที่เกิดขึ้น เมื่อไมโครชิปได้รับพลังงานก็จะทำงานตามที่ได้ตั้งค่าไว้ โดยเครื่องอ่านจะรับรู้ได้จากสนามแม่เหล็กที่ส่งมาจากป้ายจำกัดจากหลักการทำงานแบบคู่ควบเหนี่ยวนำ ทำให้ระยะในการอ่าน

ข้อมูลสูงสุดประมาณ 1 เมตร แต่ทั้งนี้ขึ้นอยู่กับกำลังงานของเครื่องอ่านและคลื่นวิทยุที่ใช้ มีหน่วยความจำขนาดเล็กประมาณ 16 ถึง 1024 ไบต์ จึงทำให้ป้ายแบบแพสซีฟ มีขนาดเล็ก น้ำหนักเบา ราคาถูกกว่าป้ายแบบแอกทีฟ และมีอายุการใช้งานไม่จำกัด

ส่วนในระบบความถี่สูงยิ่ง (UHF) จะใช้หลักการคู่ควบแบบแผ่กระจาย (Propagation coupling) โดยที่สายอากาศของเครื่องอ่านจะทำการส่งพลังงานแม่เหล็กไฟฟ้าในรูปคลื่นออกมา เมื่อป้ายได้รับสัญญาณผ่านสายอากาศ จะสะท้อนกลับคลื่นที่ถูกปรับค่าตามรหัสประจำตัวไปยังเครื่องอ่าน (backscattering) ดังรูปที่ 2.11



รูปที่ 2.11 หลักการทำงานของป้ายในย่านความถี่ต่ำและสูงกับ UHF

ที่มา : <https://sites.google.com/site/thekhnoloyirfid/home/laksana-kar-thangan-rfid>

ไอซีของป้ายแบบแพสซีฟจะมีทั้งขนาดและรูปร่างเป็นแท่งหรือแผ่นขนาดเล็กจนถึงขนาดใหญ่ ซึ่งต่างก็มีความเหมาะสมกับชนิดงานที่แตกต่างกัน ส่วนโครงสร้างภายในที่เป็นไอซีของป้ายนั้น จะประกอบด้วย 3 ส่วน ได้แก่ ส่วนควบคุมการทำงานของภาครับส่งสัญญาณวิทยุ (Analog Front-End) ส่วนควบคุมภาคลอจิก (Digital Control Unit) ส่วนของหน่วยความจำ (Memory) ซึ่งอาจจะเป็นแบบ ROM หรือ EEPROM

2) ป้ายแบบแอกทีฟ ป้ายชนิดนี้ต้องอาศัยแหล่งจ่ายไฟจากภายนอก เพื่อจ่ายพลังงานให้กับวงจรภายในทำงาน โดยส่วนใหญ่อาจสามารถแบ่งตามหลักการทำงานได้เป็น ทรานสปอนเดอร์แบบแอกทีฟ ซึ่งจะทำการส่งข้อมูลออกก็ต่อเมื่อได้รับสัญญาณจากเครื่องอ่านและเครื่องบอกตำแหน่งหรือบีคอน (beacon) ซึ่งสัญญาณจะถูกปล่อยออกมาเป็นระยะ ๆ ตลอดเวลาการใช้งานของแท็กหรือทรานสปอนเดอร์แบบแอกทีฟนั้นอาจพบได้ในระบบ เช่น ระบบจ่ายเงินในทางด่วน หรือ ด้านตรวจขณะบีคอนอาจพบได้ในระบบที่ต้องการการบ่งชี้พิกัดแบบเวลาจริง (Real – time locating system, RTLS) เช่น การจัดการการขนส่งสินค้า เป็นต้น

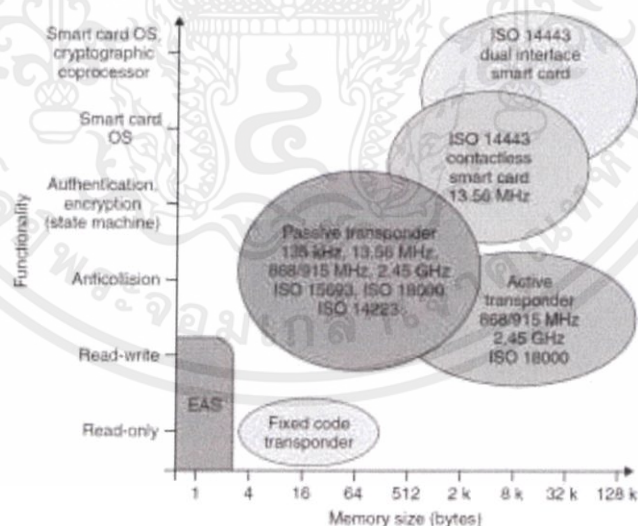
โดยแท็กแบบนี้สามารถมีหน่วยความจำภายในขนาดใหญ่ได้ถึง 1 เมกะไบต์และสามารถอ่านได้ในระยะสูงสุดประมาณ 100 เมตร ข้อเสียของแท็กแบบนี้คือ มีราคาต่อหน่วยสูง มีขนาดค่อนข้างใหญ่ และมีอายุการใช้งานที่จำกัดตามอายุของแบตเตอรี่ซึ่งจะมีอายุใช้งานประมาณ 3-7 ปี

### 2.5.2.2 เครื่องอ่าน/เขียนข้อมูล

หน้าที่ของเครื่องอ่านก็คือ การเชื่อมต่อเพื่ออ่านหรือเขียนข้อมูลลงในป้ายด้วยสัญญาณความถี่วิทยุ ภายในเครื่องอ่านจะประกอบด้วย เสาอากาศที่ทำจากขดลวดทองแดง เพื่อใช้รับส่งสัญญาณ ภาคส่งสัญญาณวิทยุ และวงจรควบคุมการอ่าน-เขียนข้อมูลซึ่งมักจะเป็นวงจรจากพวกไมโครคอนโทรลเลอร์ และส่วนของการติดต่อกับคอมพิวเตอร์ โดยลักษณะขนาด และรูปร่างของเครื่องอ่านจะอ่านแตกต่างกันไปตามประเภทของการใช้งาน

### 2.5.3 ระบบที่ใช้ประมวลผลอาร์เอฟไอดี

เป็นส่วนที่จะทำการประมวลผลข้อมูลที่ได้มาจากป้าย (Tag) หรือจะสร้างข้อมูลเพื่อส่งไปยังป้าย (Tag) หรืออาจจะเป็นที่เก็บระบบฐานข้อมูลดังรูปที่ 2.12 ทั้งนี้ขึ้นอยู่กับระบบที่เราเลือกใช้ ตัวอย่างเช่น ระบบการจัดการฟาร์มปศุสัตว์, ระบบคลังสินค้า, ระบบขนส่ง, ระบบการบริหารจัดการทรัพยากรต่างๆ เป็นต้น

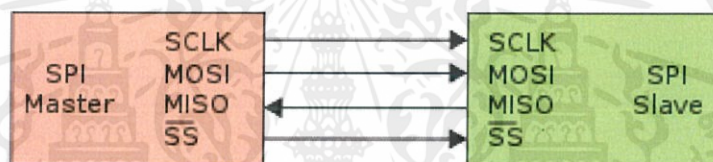


รูปที่ 2.12 ภาพแสดงถึงการจัดลำดับมาตรฐาน RFID

ที่มา : <https://www.thaieasyelec.com/article-wiki/basic-electronics/rfid-basic.html>

## 2.5.4 การเชื่อมต่อกับไมโครคอนโทรลเลอร์

Serial Peripheral Interface (SPI) เป็นการสื่อสารแบบซิงโครนัสที่ใช้สำหรับการสื่อสารระยะทางสั้น ๆ โดยเฉพาะอย่างยิ่งในระบบปิด การอินเทอร์เฟซได้รับการพัฒนา โดยไมโครโรร่า 1980 และได้กลายเป็นมาตรฐาน อุปกรณ์ SPI สื่อสารในโหมด full -duplex โดยใช้การออกแบบเป็น master-slave กับ master เดียว อุปกรณ์หลักเริ่มต้นเฟรมสำหรับการอ่านและการเขียน อุปกรณ์หลายตัวได้รับการสนับสนุนผ่านการเลือกด้วยสายเลือก (SS) แต่ละบรรทัดและอธิบายได้ว่าเป็นอินเทอร์เฟซอนุกรมแบบซิงโครนัสดังรูปที่ 2.13 แสดงให้เห็น แต่มีข้อแตกต่างจากโปรโตคอล Synchronous Serial Interface (SSI) ซึ่งเป็นโปรโตคอลการสื่อสารแบบอนุกรมสี่สายแบบซิงโครนัส SSI Protocol ใช้การส่งสัญญาณที่แตกต่างกันและให้บริการช่องทางการสื่อสาร simplex เพียงช่องเดียว



รูปที่ 2.13 ภาพแสดงการเชื่อมต่อแบบ single ระหว่าง master กับ slave  
ที่มา : [https://en.wikipedia.org/wiki/Serial\\_Peripheral\\_Interface](https://en.wikipedia.org/wiki/Serial_Peripheral_Interface)

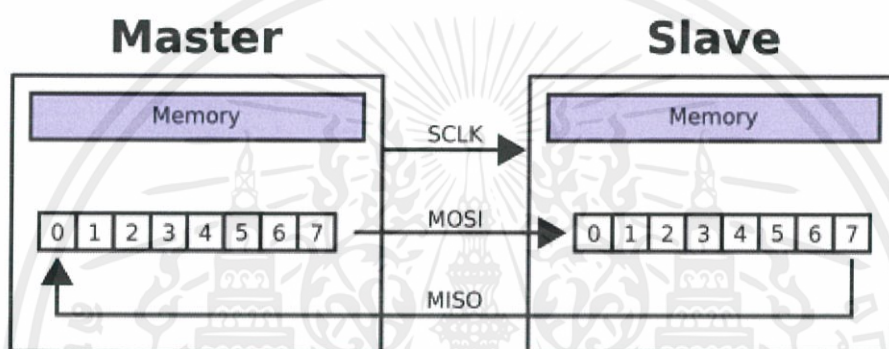
### 2.5.3.1 การทำงานของ SPI

SPI บัสสามารถทำงานกับอุปกรณ์หลักเดียวและกับอุปกรณ์อย่างน้อยหนึ่งรายการหากใช้อุปกรณ์สลาฟเดี่ยวพิน SS อาจถูกจับจ้องไปที่ลอจิกต่ำหากสลาฟอนุญาต บางครั้งต้องใช้ขอบของสัญญาณเพื่อเริ่มต้นการดำเนินการ อุปกรณ์สลาฟส่วนใหญ่มีเอาต์พุตแบบสามสถานะดังนั้นสัญญาณ MISO จึงมีความต้านทานสูง (ตัดการเชื่อมต่อแบบลอจิกคัล) เมื่อไม่ได้เลือกอุปกรณ์ อุปกรณ์ที่ไม่มีเอาต์พุต tri-state ไม่สามารถ bus segments SPI กับอุปกรณ์อื่น มีทาสคนเดียวเท่านั้นที่สามารถคุยกับ master ได้

### 2.5.3.2 การส่งข้อมูลของ SPI

การเริ่มต้นการสื่อสารบัสต้นแบบกำหนดค่า clk โดยใช้ความถี่ที่อุปกรณ์ที่ไ้รองรับ โดยขั้นตอนต่อมาต้นแบบจะเลือกอุปกรณ์สลาฟที่มีระดับตรรกะ 0 ในบรรทัดที่เลือก หากจำเป็นต้องใช้ช่วงเวลารอเช่นสำหรับการแปลงสัญญาณแอนะล็อกเป็นดิจิตอลต้นแบบ

ต้องรอน้อยกว่าเวลานั้นก่อนที่จะออกรอบสัญญาณ clk ในระหว่างรอบ clk SPI แต่ละครั้งจะมีการส่งข้อมูลเต็มรูปแบบ master จะส่งบิตบนบรรทัด MOSI และอ่าน ในขณะที่ส่งบิตบนบรรทัด MISO และ master จะทำการอ่าน ขึ้นตอนนี้จะมีแม้ว่าจะมีเพียงการถ่ายโอนข้อมูลทิศทางเดียวเท่านั้นดังรูปที่ 2.14 และการส่งสัญญาณมักจะเกี่ยวข้องกับการเปลี่ยนแปลงขนาด ที่ขนาดของคำ โดยการเชื่อมต่อในโทโพโลยี ข้อมูลมักจะถูกเลื่อนออกไปพร้อมกับบิตที่สำคัญที่สุดก่อน ที่ขอบของสัญญาณ clk ทั้งต้นแบบและเลื่อนออกเล็กน้อยและส่งออกบนสายส่งไปยังคู่ ที่ขอบสัญญาณ clk ไปที่ตัวรับสัญญาณแต่ละตัวบิตจะถูกสุ่มจากสายส่งและตั้งเป็นบิตการเปลี่ยนแปลง



รูปที่ 2.14 การตั้งค่าฮาร์ดแวร์โดยใช้พริจิสเตอร์เพื่อสร้างบัฟเฟอร์แบบวงกลม  
ที่มา : [https://en.wikipedia.org/wiki/Serial\\_Peripheral\\_Interface](https://en.wikipedia.org/wiki/Serial_Peripheral_Interface)

หลังจากที่บิตถูกเลื่อนออกและเข้าหลักและได้แลกเปลี่ยน หากต้องการแลกเปลี่ยนข้อมูลเพิ่มเติมการจะถูกโหลดใหม่และกระบวนการซ้ำ การส่งสัญญาณอาจดำเนินต่อไปตามจำนวนรอบของนาฬิกา เมื่อเสร็จแล้วผู้ควบคุมจะหยุดการสลับสัญญาณนาฬิกา

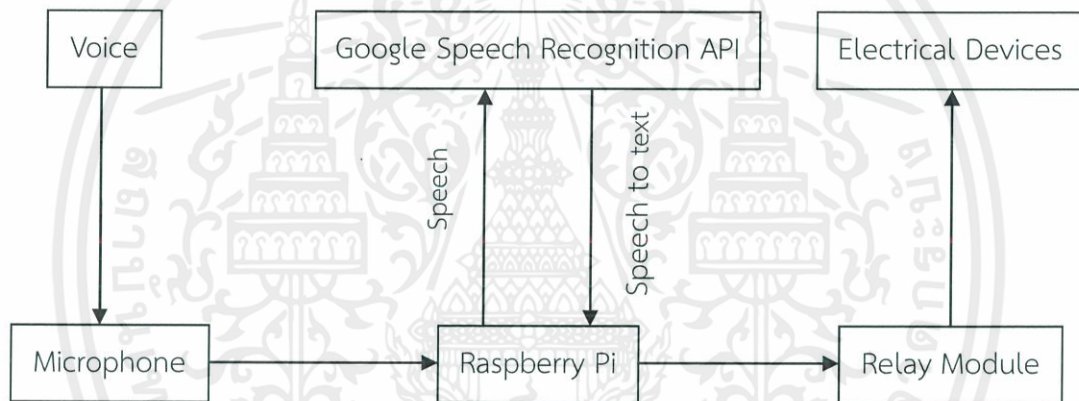
## บทที่ 3

### การออกแบบและการจัดทำปริญญานิพนธ์

#### 3.1 การออกแบบ

##### 3.1.1 การออกแบบระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง

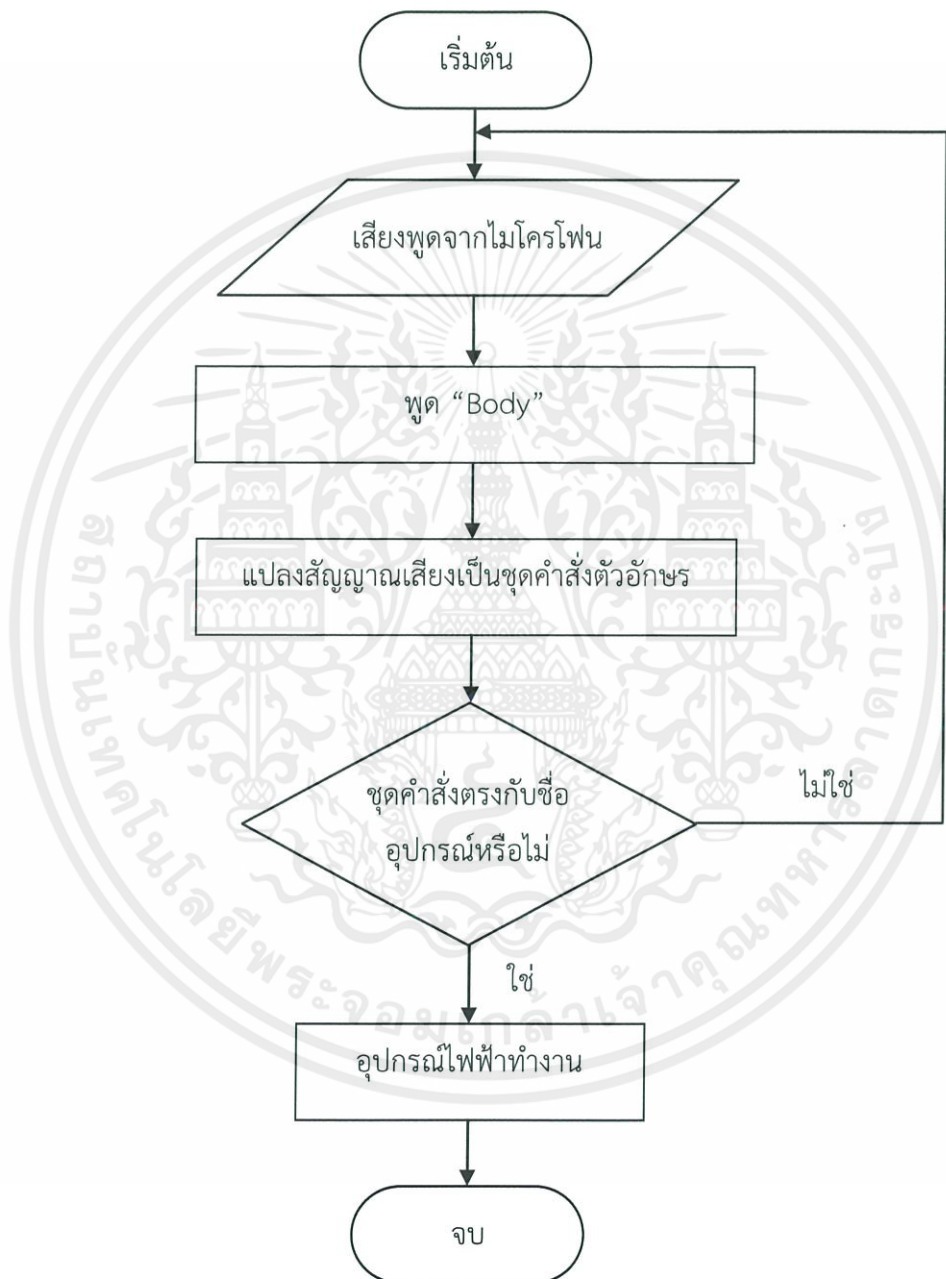
ระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงจะรับเสียงที่แปลงสัญญาณอะนาลอกเป็นสัญญาณดิจิทัลจากไมโครโฟนมาที่ Raspberry Pi เพื่อทำการประมวลผล โดยแปลงจะเสียงเป็นชุดคำสั่งตัวอักษร ซึ่งจะนำไปสั่งการรีเลย์ทำให้อุปกรณ์อิเล็กทรอนิกส์ที่เชื่อมต่อไว้ทำงาน บล็อกไดอะแกรมแสดงระบบการสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงดังรูปที่ 3.1



รูปที่ 3.1 บล็อกไดอะแกรมระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง

หลักการทำงานของระบบสั่งการด้วยเสียงจะแบ่งเป็น 2 ส่วน คือส่วนที่รองรับคำสั่งภาษาอังกฤษและภาษาไทย โดยทั้งสองส่วนจะใช้หน่วยประมวลผลแปลงจากสัญญาณเสียงเป็นข้อความ คือ Google Speech Recognition ซึ่งส่วนที่รองรับคำสั่งภาษาอังกฤษโดยระบบจะรับเสียงจากไมโครโฟนเป็นอินพุตซึ่งถูกแปลงเป็นสัญญาณดิจิทัลส่งผ่านทางพอร์ต USB มาที่ Raspberry Pi นำไปประมวลผลสัญญาณเสียงด้วย Google Speech Recognition API ซึ่งภายในคำสั่งให้อุปกรณ์ไฟฟ้าทำงาน ผู้ใช้งานต้องพูดคำว่า “Body” ระบบจึงจะสามารถรับ

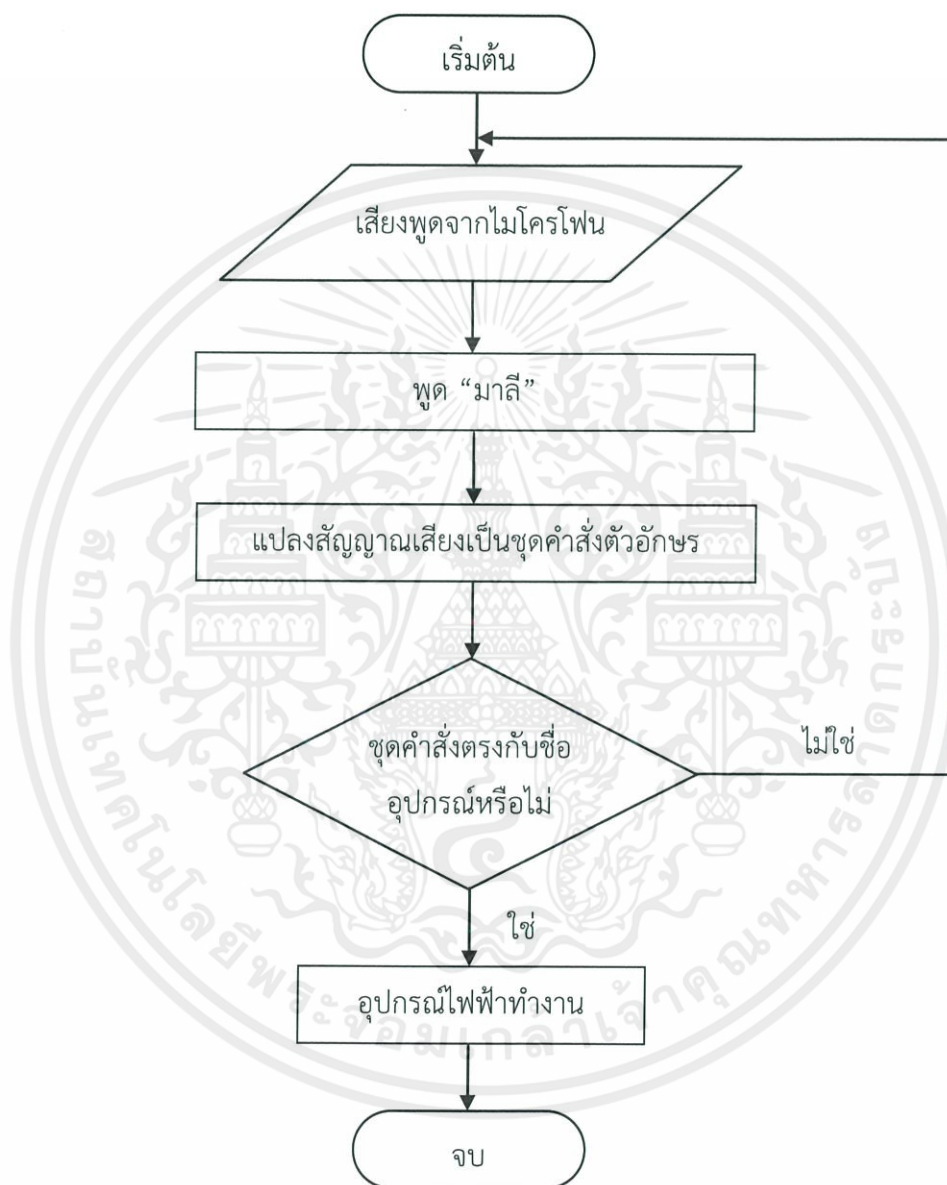
ชุดคำสั่งในการสั่งการอุปกรณ์ไฟฟ้า เมื่อรับชุดคำสั่งมาจึงทำการเปรียบเทียบกับชื่ออุปกรณ์ ถ้าชุดคำสั่งกับชื่ออุปกรณ์ตรงกันก็จะทำให้รีเลย์ควบคุมให้อุปกรณ์ทำงาน ดังรูปที่ 3.2



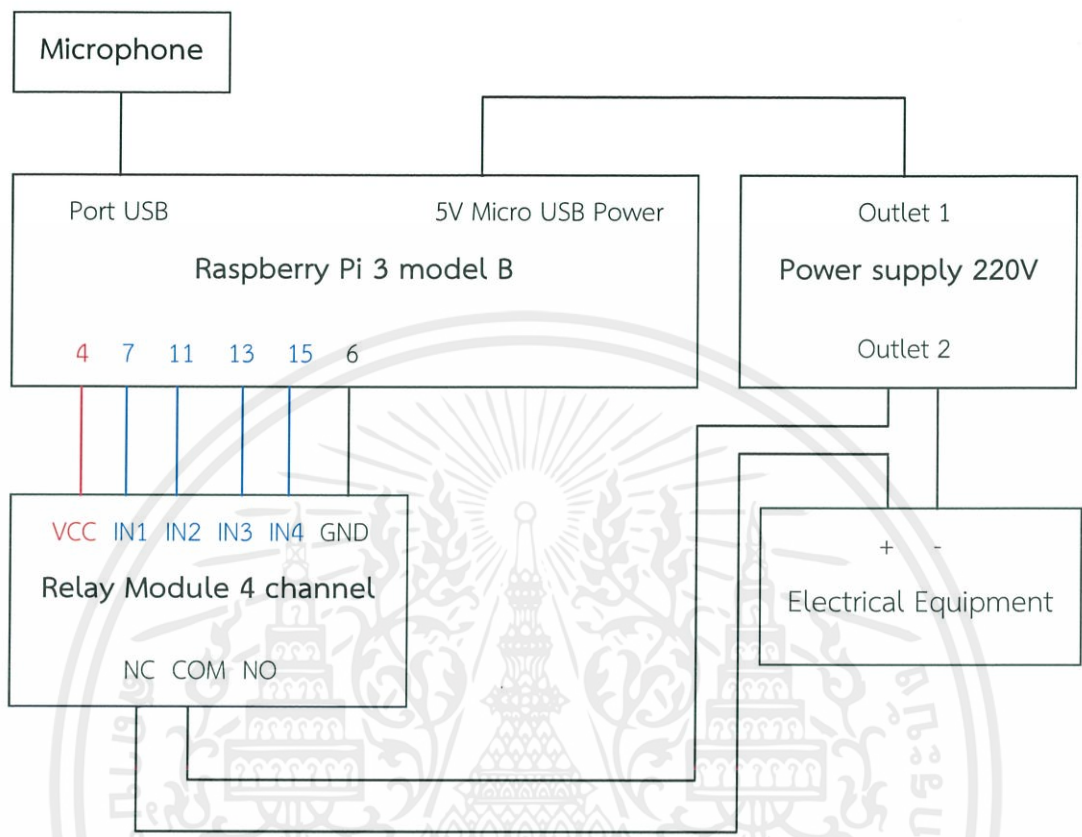
รูปที่ 3.2 แผนภาพการทำงานของระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงรองรับภาษาอังกฤษ

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนที่สองจะเป็นส่วนที่รองรับคำสั่งภาษาไทย หลักการรับเสียงและแปลงสัญญาณเป็นข้อความเช่นเดียวกับส่วนรองรับภาษาอังกฤษ แต่แตกต่างที่ภายในคำสั่งให้อุปกรณ์ไฟฟ้าทำงาน ผู้ใช้งานต้องพูดคำว่า “มาลี” ระบบจึงทำงาน ดังรูปที่ 3.3



รูปที่ 3.3 แผนภาพการทำงานของระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงรองรับภาษาไทย



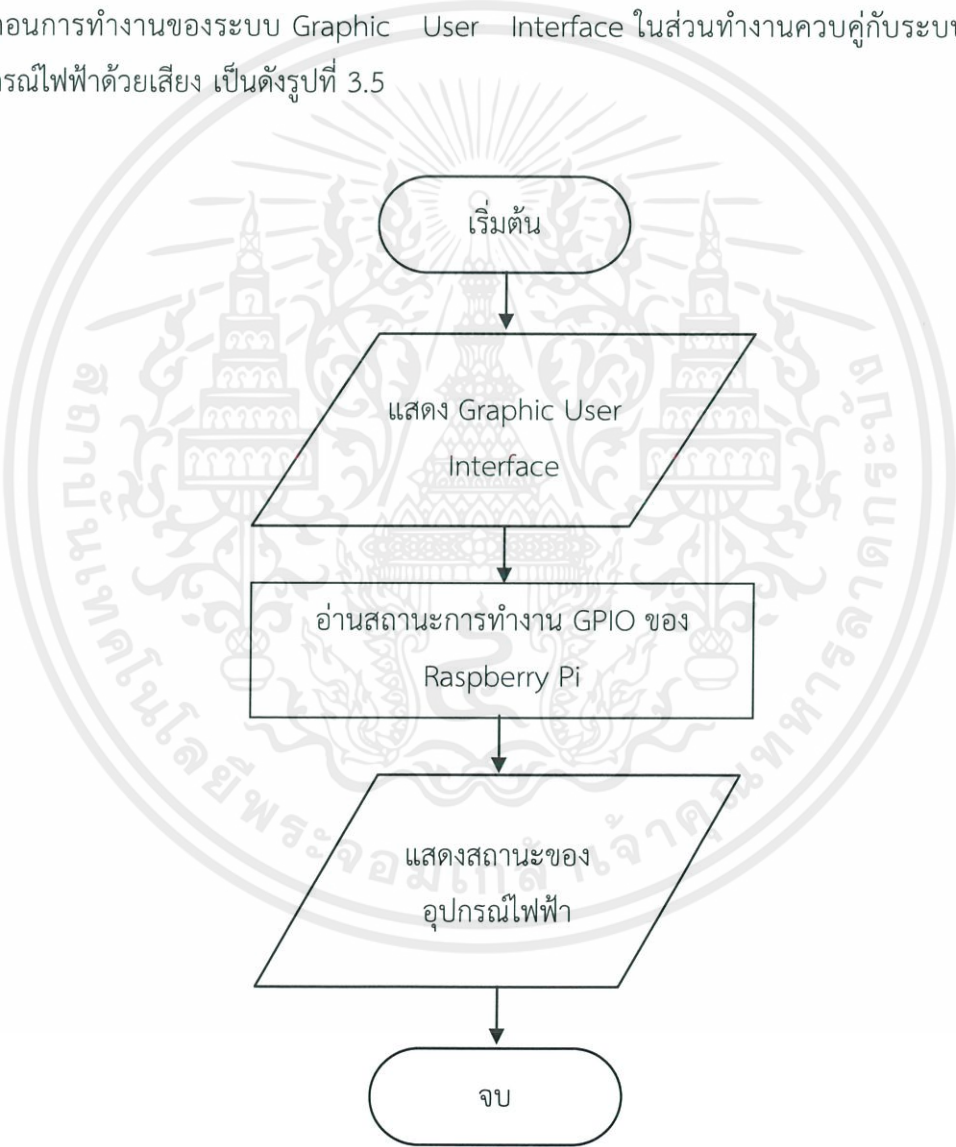
รูปที่ 3.4 วงจรระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง

จากรูปที่ 3.4 แสดงวงจรของระบบสั่งการด้วยเสียงซึ่งมี Raspberry Pi เป็นตัวประมวลผลกลาง เพื่อควบคุมการสั่งการ ใช้งานรีเลย์ผ่าน GPIO โดยรีเลย์จะทำหน้าที่คล้าย สวิตซ์ในการเปิด-ปิดอุปกรณ์ไฟฟ้า ซึ่งรีเลย์ 1 ช่อง จะควบคุมอุปกรณ์ไฟฟ้า 1 เครื่อง เป็นอิสระต่อกัน เมื่อรีเลย์ทำงาน จะทำให้อุปกรณ์ไฟฟ้าเปิดใช้งานได้ตามปกติผ่านทางขา NC ของรีเลย์มอดูล

3.1.2 การออกแบบและการทำงานของระบบ Graphic User Interface

ระบบ Graphic User Interface หรือ GUI มีหน้าที่แสดงสถานะของอุปกรณ์ไฟฟ้า ซึ่งการทำงานของ GUI จะอ่านค่าสถานะจาก GPIO ของ Raspberry Pi เมื่อ GPIO ทำงาน

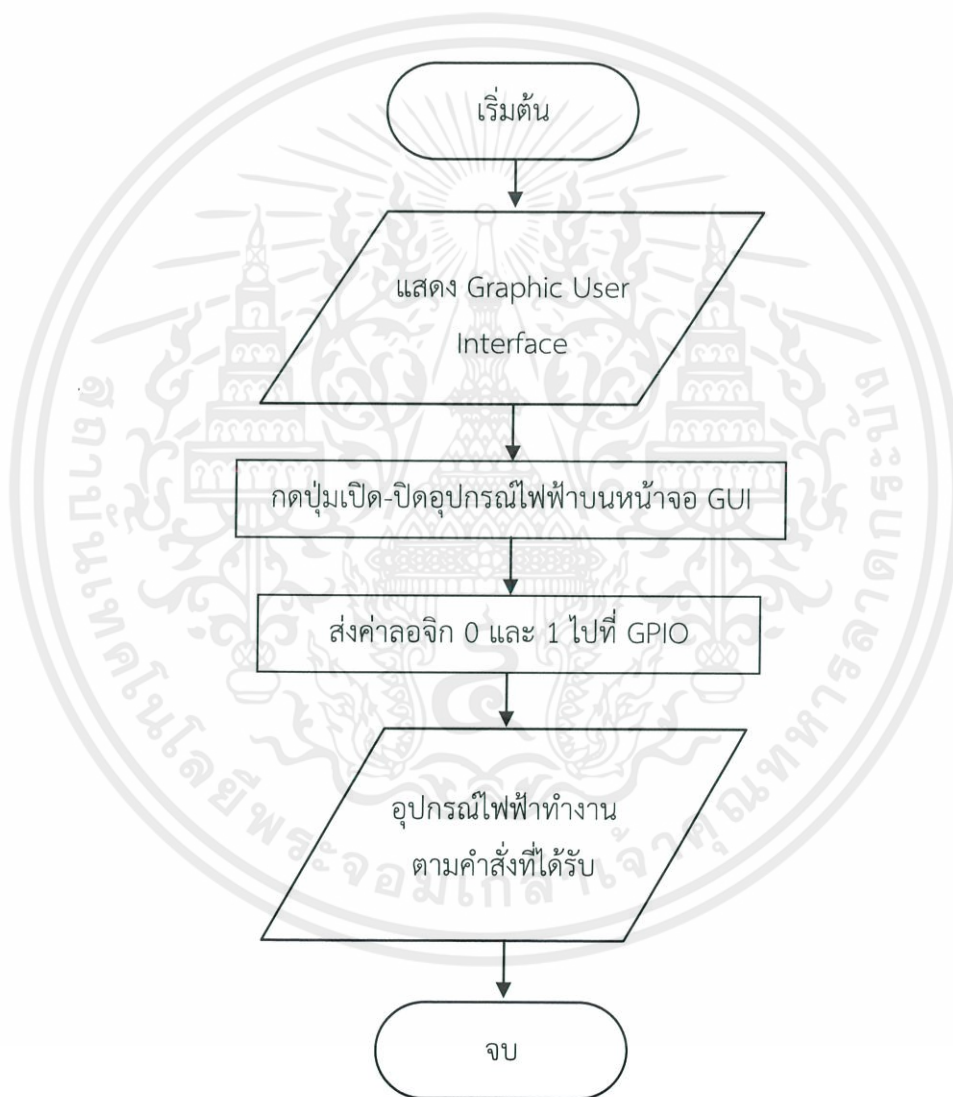
โดยระบบ Graphic User Interface จะมี 2 ส่วน คือส่วนที่จะแสดงสถานะของอุปกรณ์ไฟฟ้าถูกใช้งาน ซึ่งจะทำงานควบคู่กับระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง ขั้นตอนการทำงานของระบบ Graphic User Interface ในส่วนทำงานควบคู่กับระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง เป็นดังรูปที่ 3.5



รูปที่ 3.5 แผนภาพการทำงานของระบบ Graphic User Interface

เมื่ออ่านค่าสถานะจาก GPIO ระบบจะส่งค่าไปไฟล์ html ซึ่งเป็นหน้าเว็บไซต์หรือ GUI ในการแสดงสถานะการใช้งานของอุปกรณ์ไฟฟ้า โดยกำหนดว่า สถานะของ GPIO เป็น 1 ให้ขึ้นสถานะของอุปกรณ์ใน GUI เป็น “Turn On” แต่ถ้าสถานะของ GPIO เป็น 0 ให้ขึ้นสถานะของอุปกรณ์ใน GUI เป็น “Turn Off”

ส่วนที่สองจะเป็นส่วนที่เป็นหน้าจอ Graphic User Interface ที่ใช้ควบคุมการทำงานของอุปกรณ์ไฟฟ้าโดยตรง ซึ่งขั้นตอนการทำงานจะแสดงดังรูปที่ 3.6

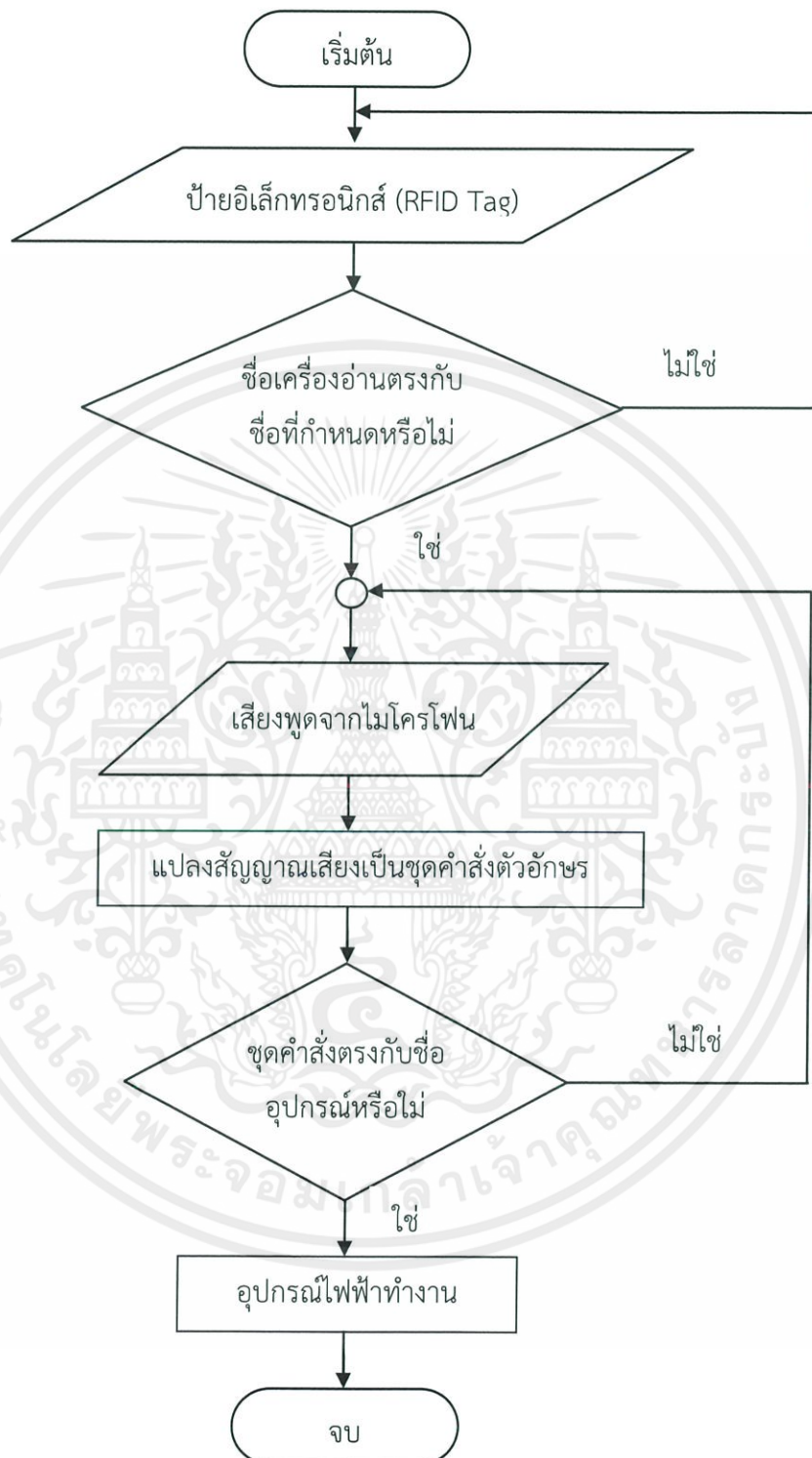


รูปที่ 3.6 แผนภาพการทำงาน Graphic User Interface ควบคุมอุปกรณ์ไฟฟ้าโดยตรง

### 3.1.3 การออกแบบระบบยืนยันตัวตนด้วย RFID

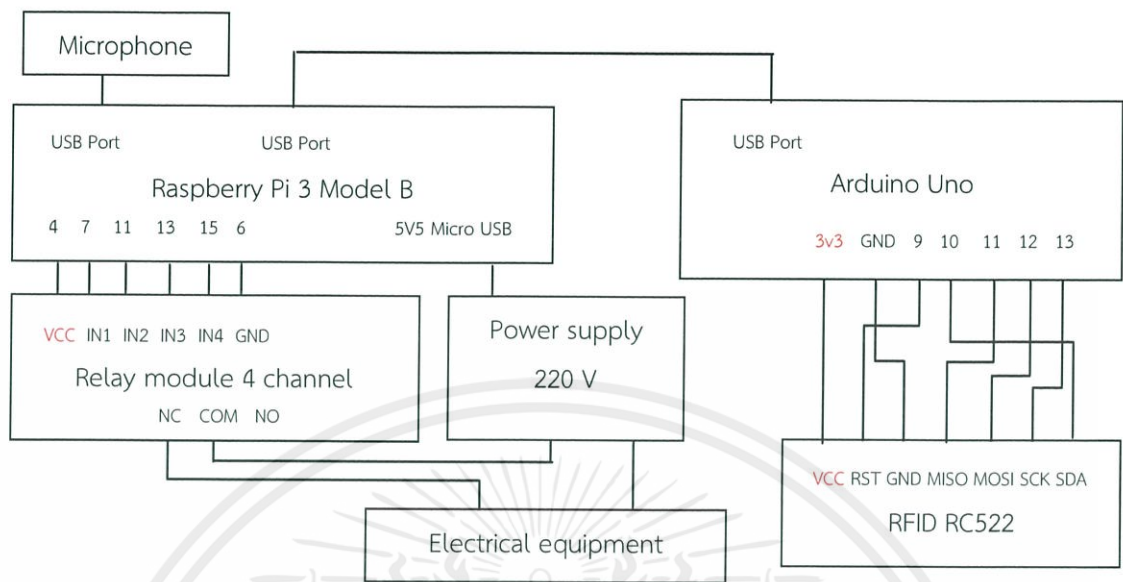
ระบบยืนยันตัวตนด้วย RFID มีหน้าที่รักษาความปลอดภัยในการใช้งานระบบควบคุมอุปกรณ์ไฟฟ้าด้วยเสียง โดยระบบยืนยันตัวตน RFID จะถูกอนุญาตให้เฉพาะผู้ที่มีป้ายอิเล็กทรอนิกส์ (RFID Tag) ในการใช้งานระบบได้ ซึ่งเครื่องอ่านสัญญาณ RFID จะเชื่อมต่อกับ Raspberry Pi และใช้วิธีสื่อสารแบบ Serial Peripheral Interface (SPI) ทำงานในรูปแบบที่ให้อุปกรณ์ตัวหนึ่งทำหน้าที่เป็น Master คือ Raspberry Pi ในขณะที่อีกตัวหนึ่งทำหน้าที่เป็น Slave คือ เครื่องอ่านสัญญาณ RFID

ในการออกแบบระบบยืนยันตัวตนด้วย RFID จะใช้เครื่องอ่านสัญญาณ RFID จำนวน 4 ตัว ซึ่งแต่ละตัวทำหน้าที่รักษาความปลอดภัยการใช้งานอุปกรณ์ไฟฟ้าในแต่ละห้อง ได้แก่ ห้องน้ำ ห้องครัว ห้องนอน และห้องนั่งเล่น เป็นต้น เมื่อผู้ใช้งานนำป้ายอิเล็กทรอนิกส์ (RFID Tag) และที่เครื่องอ่านสัญญาณ RFID หากรหัสป้ายที่เรียกว่า UID ตรงกับเงื่อนไขที่กำหนด จะทำให้ระบบคำสั่งเสียงพร้อมรับคำสั่งจากผู้ใช้งาน เพื่อการใช้งานอุปกรณ์ไฟฟ้าในแต่ละห้องที่ผู้ใช้งานนำป้ายอิเล็กทรอนิกส์ไปที่เครื่องอ่านสัญญาณ RFID ในห้องนั้น แต่ในกรณีที่รหัสป้าย UID ไม่ตรงกับเงื่อนไขที่กำหนด จะทำให้ผู้ใช้งานไม่สามารถใช้งานอุปกรณ์ไฟฟ้าในห้องนั้นได้ ซึ่งการทำงานจะแสดงดังรูปที่ 3.7 และมีวงจรที่ใช้งานดังรูปที่ 3.8



รูปที่ 3.7 แผนภาพการทำงานระบบยืนยันตัวตนด้วย RFID

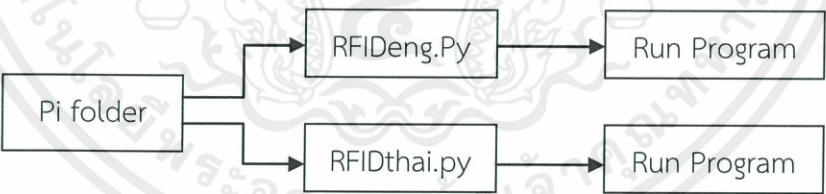
เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



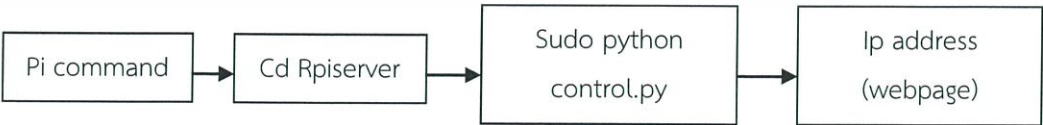
รูปที่ 3.8 วงจรระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียงแบบระบุตัวตน

3.1.4 การออกแบบการเรียกใช้ระบบคำสั่งเสียงและระบบ GUI

ระบบคำสั่งเสียงเป็นระบบหลักจะมีโปรแกรมการทำงานสำหรับรับคำสั่งภาษาอังกฤษและภาษาไทยทั้งหมด 2 โปรแกรม โดยชื่อไฟล์ว่า RFIDeng.py และ RFIDthai.py ซึ่งการเรียกใช้โปรแกรมมีขั้นตอนดังรูปที่ 3.9 ส่วนระบบควบคุมอุปกรณ์ไฟฟ้าด้วย GUI ซึ่งเป็นระบบรอง จะมีการเข้าใช้งานตามขั้นตอนดังรูปที่ 3.10



รูปที่ 3.9 แผนภาพแสดงการเรียกใช้ระบบคำสั่งเสียง



รูปที่ 3.10 แผนภาพแสดงการเรียกใช้ระบบ GUI

## 3.2 เครื่องมือที่ใช้ในการทดลอง

### 3.2.1 หน่วยประมวลผลกลาง

#### 3.2.1.1 Raspberry Pi 3 model B

เป็นบอร์ดคอมพิวเตอร์ขนาดเล็กสามารถรองรับระบบปฏิบัติการลินุกซ์ (Linux Operating System) ได้หลายระบบ มี CPU เป็น Cortex-A53 ด้วยความเร็ว 1.2 GHz ที่สัญญาณนาฬิกา และจุดเชื่อมต่อ GPIO ให้ผู้ใช้สามารถนำไปใช้ร่วมกับอุปกรณ์อิเล็กทรอนิกส์อื่นๆ ดังรูปที่ 3.11

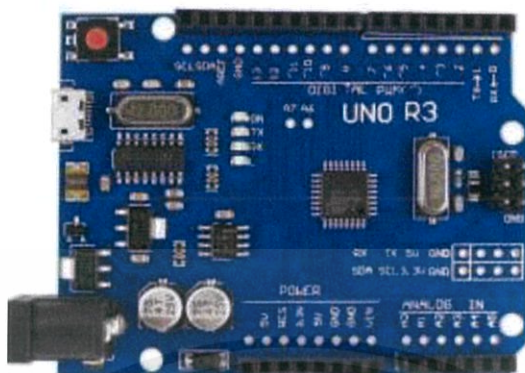


รูปที่ 3.11 Raspberry Pi 3 model B

### 3.2.2 อุปกรณ์ในระบบสั่งงานอุปกรณ์ไฟฟ้าด้วยเสียง

#### 3.2.2.1 Arduino UNO

เป็นบอร์ดไมโครคอนโทรลเลอร์ตระกูล AVR ที่มีการพัฒนาแบบ Open Source (เปิดเผยข้อมูลทั้งด้าน Hardware และ Software) ตัวบอร์ด Arduino ถูกออกแบบมาให้ใช้งานได้สะดวก สามารถดัดแปลง เพิ่มเติม พัฒนาต่อยอดทั้งตัวบอร์ดหรือโปรแกรมให้ตรงกับความต้องการการใช้งาน โดยบอร์ด Arduino ในการต่ออุปกรณ์เสริมต่างๆ สามารถต่อวงจรอิเล็กทรอนิกส์จากภายนอกแล้ว เชื่อมต่อเข้ามาที่ขา I/O ของบอร์ด หรือเพื่อความสะดวกสามารถเลือกต่อกับบอร์ดเสริม (Arduino Shield) ประเภทต่างๆ เช่น Arduino XBee เป็นต้น แล้วเขียนโปรแกรมเพื่อใช้งาน และตัวบอร์ดมีรูปแบบดังรูปที่ 3.12



รูปที่ 3.12 Arduino UNO board

### 3.2.2.2 ไมโครโฟน รุ่น MIC-01 ยี่ห้อ Scorpion

มีทำหน้าที่ในการรับเสียงเข้ามาและแปลงสัญญาณอะนาลอกเป็นสัญญาณดิจิตอลส่งผ่านทาง USB เพื่อเป็นอินพุตให้ระบบนำไปแปลงเป็นตัวอักษรในการประมวลผลเพื่อส่งงานอุปกรณ์ไฟฟ้าให้ทำงาน ดังรูปที่ 3.13

คุณสมบัติของไมโครโฟน

รับสัญญาณแบบทั่วทุกทิศทาง (Omnidirectional)

- ค่าความไวต่อเสียงของไมโครโฟน คือ  $-34 \text{ dB} \pm 3 \text{ dB}$



รูปที่ 3.13 ไมโครโฟน รุ่น MIC-01 ยี่ห้อ Scorpion

### 3.2.2.3 หลอดไฟ

เป็นอุปกรณ์ไฟฟ้าที่ใช้ในการทดลอง ซึ่งจะใช้รีเลย์มอดูลเป็นตัวควบคุมการทำงานเปิดและปิดของหลอดไฟ โดยหลอดไฟจะต่อเข้ากับรีเลย์มอดูลและแหล่งจ่ายไฟฟ้ากระแสสลับ 220 โวลต์ เพื่อเปลี่ยนตัวแทนของหลอดไฟที่ใช้ในชีวิตประจำวันที่ใช้ไฟฟ้า 220 โวลต์เท่ากัน ดังรูปที่ 3.14



รูปที่ 3.14 หลอดไฟไส้ร้อนแบบธรรมดา

### 3.2.2.4 พัดลม

เป็นอุปกรณ์ไฟฟ้าที่นำมาใช้ในการทดลอง ซึ่งมีรีเลย์มอดูลทำหน้าที่คล้ายสวิตช์ให้กับพัดลม และเป็นตัวเชื่อมให้ระหว่าง Raspberry pi กับแหล่งจ่ายไฟฟ้ากระแสสลับ 220 โวลต์ เป็นการยกตัวอย่างแทนอุปกรณ์ไฟฟ้าที่ใช้ไฟ 220 โวลต์ ดังรูปที่ 3.15



รูปที่ 3.15 พัดลม

3.2.2.5 รีเลย์ 4 ช่อง (4-Channel Relay Module 5V)

มีหน้าที่คล้ายสวิตช์อิเล็กทรอนิกส์ ซึ่งได้นำมาใช้ในการควบคุมการทำงานเปิด-ปิดอุปกรณ์ไฟฟ้าในระบบ เช่น หลอดไฟ พัดลม โดยส่งงานผ่าน GPIO ของ Raspberry Pi ซึ่งรีเลย์แต่ละช่อง เป็นอิสระต่อกันควบคุมอุปกรณ์ไฟฟ้าได้ 1 เครื่อง

คุณสมบัติของรีเลย์มอดูล

- รับแรงดันไฟฟ้ากระแสสลับได้สูงสุด 250 โวลต์ 10 แอมแปร์
- รับแรงดันไฟฟ้ากระแสตรงได้สูงสุด 30 โวลต์ 10 แอมแปร์

ตารางที่ 3.1 ข้อมูลรายละเอียด Pin ของรีเลย์มอดูล 4 ช่อง

| Name   | Description    |
|--------|----------------|
| GND    | Supply ground  |
| IN 1-4 | GPIO input     |
| VCC    | Supply Voltage |

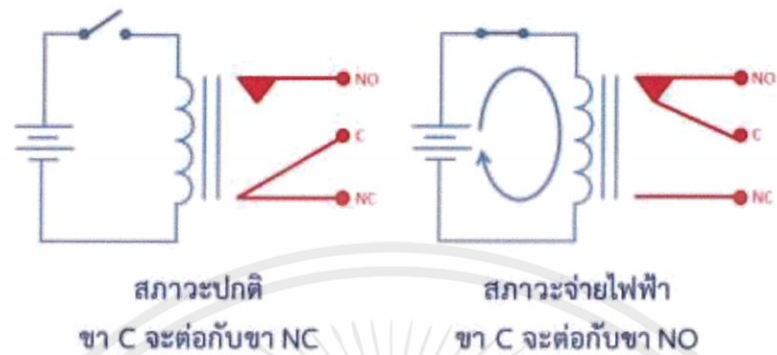
จากตารางที่ 3.1 แสดงข้อมูลรายละเอียด Pin ของรีเลย์มอดูล 4 ช่องในการเชื่อมต่อเข้ากับบอร์ดไมโครคอนโทรลเลอร์ เช่น Raspberry Pi, Arduino เป็นต้น โดยขา IN 1-4 จะถูกสั่งการหรือถูกรับคำสั่งจากบอร์ดไมโครคอนโทรลเลอร์ เพื่อให้รีเลย์ทำงานได้

ตารางที่ 3.2 โครงสร้างของรีเลย์มอดูล

| Name | Description  |
|------|--------------|
| NC   | Normal close |
| COM  | Common       |
| NO   | Normal open  |

จากตารางที่ 3.2 แสดงโครงสร้างของรีเลย์มอดูลภายใน โครงสร้างของ รีเลย์ จะประกอบไปด้วยขดลวด 1 ชุด และ หน้าสัมผัส ซึ่งในหน้าสัมผัส 1 ชุด ซึ่งจะประกอบไปด้วย หน้าสัมผัสแบบปกติปิด (Normally Close หรือ NC.) ซึ่งในสภาวะปกติขานี้จะต่อกับขาร่วม (Common) และ หน้าสัมผัสแบบปกติเปิด (Normally Open หรือ NO.) ขานี้จะต่อ

เข้ากับขาร่วม (Common) เมื่อขดลวดมีแรงดันตกคร่อมหรือกระแสไหลผ่านในปริมาณที่เพียงพอ ดังรูปที่ 3.16



รูปที่ 3.16 สภาวะการทำงานของ Relay Module

โดยส่วนที่ใช้ต่อกับอุปกรณ์ไฟฟ้าที่ใช้ คือขา NC และขา COM ซึ่งเป็นการใช้งานในสภาวะปกติ จะนำขา COM ต่อเข้ากับแหล่งจ่ายไฟกระแสสลับแรงดัน 220 โวลต์ ส่วนขา NC ต่อเข้ากับอุปกรณ์ไฟฟ้าที่ต้องการควบคุม ดังรูปที่ 3.17

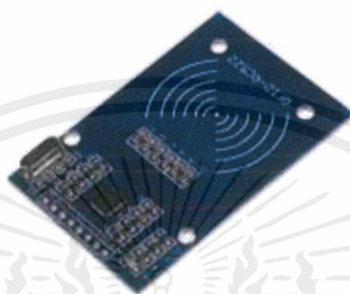


รูปที่ 3.17 รีเลย์ 4 ช่อง (4-Channel Relay Module 5V)

### 3.2.3 อุปกรณ์ในระบบยืนยันตัวตนด้วย RFID

#### 3.2.3.1 เครื่องอ่านสัญญาณ RFID โมดูล RC522

มีหน้าที่รักษาความปลอดภัยสำหรับระบบควบคุมอุปกรณ์ไฟฟ้า ด้วยเสียงจากผู้ใช้งานที่ได้รับอนุญาต และเพื่อเตรียมระบบในการเข้าใช้งานอุปกรณ์ไฟฟ้า ดังรูปที่ 3.18



รูปที่ 3.18 เครื่องอ่านสัญญาณ RFID โมดูล RC522

#### 3.2.3.2 ป้ายอิเล็กทรอนิกส์ (RFID Tag)

เป็นอุปกรณ์ยืนยันผู้เข้าใช้งานตามที่ระบบอนุญาต หากป้ายอิเล็กทรอนิกส์มีรหัสตรงตามเงื่อนไขจึงสามารถเข้าใช้งานระบบได้ ดังรูปที่ 3.19



รูปที่ 3.19 ป้ายอิเล็กทรอนิกส์ (RFID Tag)

### 3.3 การจัดเก็บผลการทดลอง

#### 3.3.1 การทดสอบระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง

1. ทดสอบระบบโดยส่งคำสั่งเสียงผ่านไมโครโฟน เพื่อสังเกตคำตัวอักษรที่ได้จากการประมวลผลของ Google Speech Recognition API ผ่าน Raspberry Pi
2. เปรียบเทียบคำสั่งที่รับมาจากการประมวลผลด้วย Google Speech Recognition API กับฐานข้อมูลคำสั่งเสียงที่ใช้สั่งการควบคุมอุปกรณ์ไฟฟ้า
3. ทดสอบระยะประสิทธิภาพของไมโครโฟนในการรับคำสั่งเสียง
4. สังเกตการทำงานอุปกรณ์ไฟฟ้ามีการทำงานตรงตามที่ได้รับคำสั่งหรือไม่
5. บันทึกผลค่าความถูกต้องของคำสั่งเสียง

#### 3.3.2 การทดสอบระบบ Graphic User Interface

1. ทดสอบระบบโดยส่งคำสั่งผ่านระบบคำสั่งเสียงให้ทำงาน GPIO ของ Raspberry Pi เพื่อสังเกตการเปลี่ยนค่าสถานะการถูกใช้งานบนหน้าเว็บไซต์
2. ทดสอบการทำงานของหน้าจอของ GUI สั่งให้อุปกรณ์ไฟฟ้าทำงานโดยตรง เพื่อสังเกตการทำงานของอุปกรณ์ไฟฟ้า
3. บันทึกผลการทำงานของการแสดงสถานะอุปกรณ์บนหน้าเว็บไซต์

#### 3.3.3 การทดสอบระบบยืนยันตัวตนด้วย RFID

1. ทดสอบความถูกต้องและแม่นยำในการตรวจจับป้ายอิเล็กทรอนิกส์ (RFID Tag) เพื่อสังเกตข้อผิดพลาดในระบบ
2. ตรวจสอบรหัสป้ายอิเล็กทรอนิกส์ (RFID Tag) ตรงตามเงื่อนไขที่กำหนดการเข้าใช้งานอุปกรณ์ไฟฟ้าในแต่ละห้องหรือไม่
3. บันทึกผลค่าความถูกต้องในการตรวจจับสัญญาณจากป้ายอิเล็กทรอนิกส์ (RFID tag)

## บทที่ 4

### ผลการทดลอง

#### 4.1 ผลการทดสอบระบบสั่งการอุปกรณ์ไฟฟ้าด้วยเสียง

##### 4.1.1 ผลการทดสอบค่าความถูกต้องของการแปลงเสียงพูดเป็นตัวอักษร

การทดสอบเสียงพูดภาษาอังกฤษจากไมโครโฟน ซึ่งเป็นอินพุตของระบบ นำไปประมวลผลด้วย Google Speech Recognition API โดยผลของคำสั่งภาษาอังกฤษสำหรับ เปิดและปิดไฟในห้องนอนที่ได้เป็นดังรูปที่ 4.1 และ 4.2

```
===== RESTART: /home/pi/Eng.py
You said: turn off bedroom light
You said: turn off bedroom light
You said: turn off bedroom light
You said: turn off bedroom light
You said: turn off the light
You said: turn off bedroom light
You said: turn off bedroom light
You said: turn off bedroom light
You said: turn off bedroom light
You said: turn off bedroom light
```

รูปที่ 4.1 ข้อความที่ระบบได้รับคำว่า “Turn off bedroom light”

```
===== RESTART: /home/pi/Eng.py
You said: turn on bedroom light
You said: turn on bedroom light
You said: turn on table light
You said: turn on bedroom light
You said: turn on bedroom light
You said: turn on bedroom light
You said: turn on the light
You said: turn on bedroom light
You said: turn on bedroom light
You said: turn on bedroom light
```

รูปที่ 4.2 ข้อความที่ระบบได้รับคำว่า “Turn on bedroom light”

ส่วนผลการทดสอบการประมวลผลด้วย Google Speech Recognition API ในคำสั่งภาษาอังกฤษของการเปิดและปิดพัดลมในห้องครัวที่ได้เป็นดังรูปที่ 4.3 และ 4.4

```
===== RESTART: /home/pi/Eng.py
You said: turn off kitchen room fan
You said: turn off kitchen room fan
You said: turn off kitchen fan
You said: turn off fan
You said: turn off kitchen room fan
You said: turn off kitchen room fan
You said: turn off kitchen floor plan
You said: 10 floor plan
You said: turn off kitchen room fan
You said: turn off kitchen floor plan
```

รูปที่ 4.3 ข้อความที่ระบบได้รับคำว่า “Turn off kitchen room fan”

```
===== RESTART: /home/pi/Eng.py
You said: turn on kitchen room fan
You said: turn on kitchen room fan
You said: turn on kitchen room
You said: turn on kitchen room
You said: turn on kitchen room fan
You said: turn on kitchen room plan
You said: turn on kitchen fan
You said: turn on kitchen room fan
You said: turn on kitchen room fan
You said: turn on kitchen table
```

รูปที่ 4.4 ข้อความที่ระบบได้รับคำว่า “Turn on kitchen room fan”

จากรูปข้างต้นได้ทำการทดสอบประมวลผลการแปลงเสียงพูดภาษาอังกฤษเป็นข้อความในคำสั่งต่างๆ คำสั่งละจำนวน 10 ครั้ง ซึ่งผลที่ได้นำไปสร้างเป็นตารางความถูกต้องของการแปลงเสียงพูดภาษาอังกฤษเป็นตัวข้อความ ดังตารางที่ 4.1

ตารางที่ 4.1 ผลการทดสอบความถูกต้องในการแปลงเสียงพูดภาษาอังกฤษเป็นข้อความ

| คำสั่ง                    | จำนวนข้อความที่ถูก | จำนวนข้อความที่ผิด | ความถูกต้องคิดเป็นเปอร์เซ็นต์ |
|---------------------------|--------------------|--------------------|-------------------------------|
| Turn off bedroom light    | 9                  | 1                  | 90%                           |
| Turn on bedroom light     | 8                  | 2                  | 80%                           |
| Turn off kitchen room fan | 5                  | 5                  | 50%                           |
| Turn on kitchen room fan  | 5                  | 5                  | 50%                           |

จากตารางข้างต้นพบว่า การทดสอบการแปลงเสียงพูดคำสั่งภาษาอังกฤษเป็นข้อความโดยใช้ Google Speech Recognition API เป็นตัวประมวลในการรับสัญญาณอินพุตเสียงพูดคำสั่งภาษาอังกฤษจากไมโครโฟนมีความถูกต้องในการแปลงเป็นข้อความทั้งหมด 27 ครั้ง จากการพูดทั้งหมด 40 ครั้ง โดยจะคิดเป็นเปอร์เซ็นต์ความถูกต้องได้ 60 เปอร์เซ็นต์ ทั้งนี้ความถูกต้องในการแปลงเสียงพูดคำสั่งภาษาอังกฤษเป็นข้อความอาจเกิดความผิดพลาดได้ขึ้นอยู่กับปัจจัยหลายอย่าง เช่น เสียงรบกวนจากสภาพแวดล้อมรอบข้าง คำที่พูดไม่ชัด การออกเสียงคำพูดที่ใกล้เคียงกัน และความเร็วในการออกเสียงคำพูด เป็นต้น

อีกส่วนของการทดสอบเสียงพูดภาษาไทยจากไมโครโฟน ซึ่งเป็นอินพุตของระบบนำไปประมวลผลด้วย Google Speech Recognition API โดยผลของคำสั่งภาษาไทยสำหรับเปิดและปิดไฟในห้องนอนซึ่งเป็นดังรูปที่ 4.5 และ 4.6

```
===== RESTART: /home/pi/thai.py
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
คุณพูดว่า: เปิดไฟห้องนอน
```

รูปที่ 4.5 ข้อความที่ระบบได้รับคำว่า “เปิดไฟห้องนอน”

```
===== RESTART: /home/pi/thai.py :
คุณพูดว่า: ห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
คุณพูดว่า: ปิดไฟห้องนอน
```

รูปที่ 4.6 ข้อความที่ระบบได้รับคำว่า “ปิดไฟห้องนอน”

ส่วนผลการทดสอบการประมวลผลด้วย Google Speech Recognition API ในคำสั่งภาษาไทยของการเปิดและปิดพัดลมในห้องครัวที่ได้เป็นดังรูปที่ 4.7 และ 4.8

```
===== RESTART: /home/pi/thai.py
คุณพูดว่า: เปิดพัดลมห้องครัว
คุณพูดว่า: เปิดพัดลมห้องครัว
คุณพูดว่า: ตรวจหัว
คุณพูดว่า: ตรวจหัว
คุณพูดว่า: เปิดพัดลมห้องครัว
คุณพูดว่า: เปิดพัดลมห้องครัว
คุณพูดว่า: เปิดพัดลมห้องครัว
คุณพูดว่า: เปิดโรงฆ่า
คุณพูดว่า: เปิดนมห้องครัว
คุณพูดว่า: เปิดพัดลมห้องครัว
```

รูปที่ 4.7 ข้อความที่ระบบได้รับคำว่า “เปิดพัดลมห้องครัว”

```
===== RESTART: /home/pi/thai.py
คุณพูดว่า: เบอรรถบางปู
คุณพูดว่า: เปิดห้องครัว
คุณพูดว่า: เปิดเตาต้มห้องครัว
คุณพูดว่า: เปิดเตาต้มห้องครัว
คุณพูดว่า: เปิดโรงฟ้าว
คุณพูดว่า: เปิดเตาต้มห้องครัว
คุณพูดว่า: เปิดเตาต้มห้องครัว
คุณพูดว่า: เปิดโรงห้องครัว
คุณพูดว่า: เปิดเตาต้มห้องครัว
คุณพูดว่า: เปิดเตาต้มห้องครัว
```

รูปที่ 4.8 ข้อความที่ระบบได้รับคำว่า “ปิดเตาต้มห้องครัว”

จากรูปข้างต้นได้ทำการทดสอบประมวลผลการแปลงเสียงพูดภาษาไทยเป็นข้อความในคำสั่งต่างๆ คำสั่งละจำนวน 10 ครั้ง ซึ่งผลที่ได้นำไปสร้างเป็นตารางความถูกต้องของการแปลงเสียงพูดภาษาไทยเป็นข้อความ ดังตารางที่ 4.2

ตารางที่ 4.2 ผลการทดสอบความถูกต้องในการแปลงเสียงพูดภาษาไทยเป็นข้อความ

| คำสั่ง             | จำนวนข้อความที่ถูก | จำนวนข้อความที่ผิด | ความถูกต้องคิดเป็นเปอร์เซ็นต์ |
|--------------------|--------------------|--------------------|-------------------------------|
| เปิดไฟห้องนอน      | 9                  | 1                  | 90%                           |
| ปิดไฟห้องนอน       | 9                  | 1                  | 90%                           |
| เปิดเตาต้มห้องครัว | 6                  | 4                  | 60%                           |
| ปิดเตาต้มห้องครัว  | 5                  | 5                  | 50%                           |

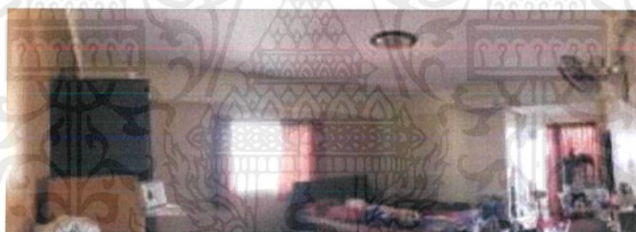
จากตารางข้างต้นพบว่า การทดสอบการแปลงเสียงพูดคำสั่งภาษาไทยเป็นข้อความโดยใช้ Google Speech Recognition API เป็นตัวประมวลในการรับสัญญาณอินพุตเสียงพูดคำสั่งภาษาไทยจากไมโครโฟนมีความถูกต้องในการแปลงเป็นข้อความทั้งหมด 29 ครั้ง จากการพูดทั้งหมด 40 ครั้ง โดยจะคิดเป็นเปอร์เซ็นต์ความถูกต้องได้ 72.5 เปอร์เซ็นต์ ทั้งนี้ความถูกต้องในการแปลงเสียงพูดคำสั่งภาษาไทยเป็นข้อความอาจเกิดความผิดพลาดได้ขึ้นอยู่กับปัจจัยหลายอย่าง เช่น เสียงรบกวนจากสภาพแวดล้อมรอบข้าง คำที่พูดไม่ชัด และความเร็วในการออกเสียงคำพูด เป็นต้น

#### 4.1.2 ผลการทดสอบระยะประสิทธิภาพในการรับเสียงของไมโครโฟน

การทดสอบวัดระยะประสิทธิภาพในการรับเสียงของไมโครโฟน ซึ่งทดลองระบบสั่งการอุปกรณ์ด้วยเสียงในห้องขนาดความกว้าง 4 เมตร ความยาว 6 เมตร โดยผู้ทดลองได้ทำการเดินไปในตำแหน่งต่างๆของห้อง และทำการสั่งการระบบให้อุปกรณ์ทำงาน ซึ่งภายในห้องจะมีไมโครโฟนวางไว้ตำแหน่งตรงกลางห้อง ดังรูปที่ 4.9



(ก)



(ข)

รูปที่ 4.9 (ก) และ (ข) แบบห้องทำการทดลองและตำแหน่งไมโครโฟน

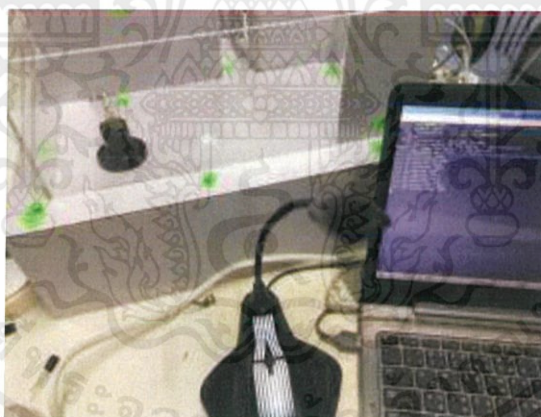
จากผลการทดลอง พบว่าเมื่อผู้ทดลองได้อยู่ในตำแหน่งต่างๆ ที่เป็นจุดสีเขียวตามรูปที่ 4.8 และได้ทำการสั่งให้ระบบทำงานในตำแหน่งละ 10 ครั้ง ซึ่งผลที่ได้เป็นไปตามผลที่ผู้ทดลองต้องการทั้งหมด 32 ครั้ง จากการสั่งงานระบบ 40 ครั้ง โดยคิดเป็นเปอร์เซ็นต์ความถูกต้อง 80 เปอร์เซ็นต์ ทั้งนี้การที่ไมโครโฟนไม่ได้รับคำสั่งหรือคำสำคัญ (keyword) เป็นอินพุตให้ระบบ ทำให้ระบบไม่สามารถทำงานเพื่อรอรับคำสั่งได้ ซึ่งความผิดพลาดได้ขึ้นอยู่กับหลายปัจจัย เช่น ประสิทธิภาพของไมโครโฟน ค่าความไวในการตอบสนองของไมโครโฟนไม่เพียงพอ ระดับความดังของเสียงพูด และเสียงรบกวนจากภายนอก เป็นต้น

### 4.1.3 ผลการทดสอบระบบควบคุมการทำงานของหลอดไฟด้วยคำสั่งเสียง

การทดสอบควบคุมการทำงานของหลอดไฟด้วยคำสั่ง “Turn on” และ “Turn off” ซึ่งจะทำให้การทำงานของ GPIO ของ Raspberry Pi ไปสั่งให้รีเลย์มอดูลทำงาน หลอดไฟจึงเปิดและปิดตามที่ต้องการ ดังรูปที่ 4.10 และ 4.11



รูปที่ 4.10 คำสั่งเปิดหลอดไฟ “Turn on”



รูปที่ 4.11 คำสั่งปิดหลอดไฟ “Turn off”

จากรูปข้างต้นเป็นการควบคุมการทำงานของหลอดไฟโดยใช้คำสั่งด้วยเสียง โดยมี Raspberry Pi เป็นตัวประมวลผลของระบบ ซึ่งผลที่ได้เป็นไปตามคำสั่งที่ต้องการของผู้ทดลองในการเปิดและปิดหลอดไฟจากเสียงพูด

## 4.2 ผลการทดสอบระบบ Graphic User Interface

### 4.2.1 ผลการทดสอบ GUI ที่ควบคุมอุปกรณ์ไฟฟ้าโดยตรง

การทดสอบ GUI ที่ใช้ควบคุมสั่งการทำงานของอุปกรณ์ไฟฟ้าโดยตรง จะเป็นการสั่งงานผ่านหน้าเว็บเพจที่สร้างจากไฟล์ HTML ซึ่งทำงานร่วมกับโปรแกรม Python สำหรับสั่งงานผ่าน GPIO ดังรูปที่ 4.12

#### Controlling Electronic Devices Using GUI



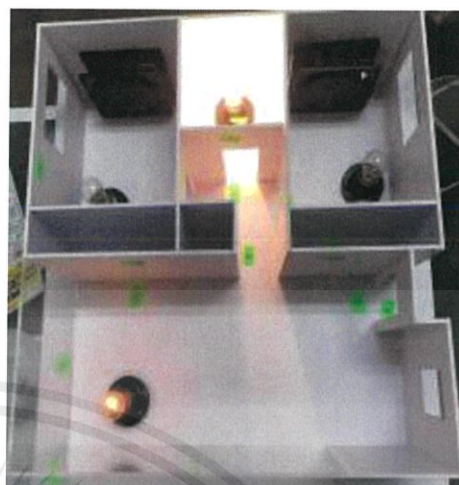
รูปที่ 4.12 GUI ควบคุมอุปกรณ์ไฟฟ้าโดยตรง

จากรูปที่ 4.12 เป็น GUI ควบคุมอุปกรณ์ไฟฟ้าโดยตรง เมื่อผู้ทดลองกดปุ่ม TURN ON ในส่วนของหลอดไฟในห้องนั่งเล่นและห้องน้ำ ปุ่มกดจะเปลี่ยนเป็น TURN OFF และสถานะอุปกรณ์นั้นจะเปลี่ยนเป็น “ON” ดังรูปที่ 4.13

### Controlling Electronic Devices Using GUI



(ก)



(ข)

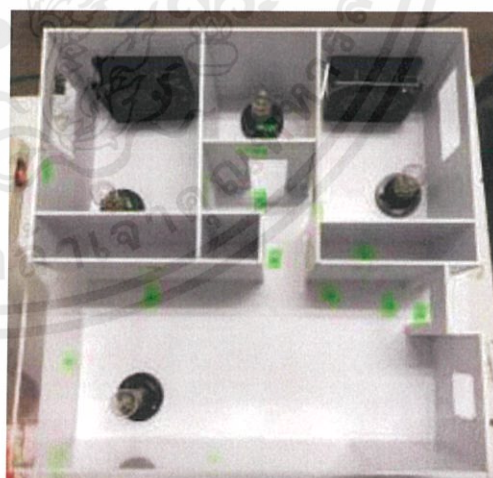
รูปที่ 4.13 (ก) และ (ข) GUI สั่งเปิดไฟห้องนั่งเล่นและห้องน้ำ

จากรูปที่ 4.13 (ก) เมื่อผู้ทดลองกดปุ่ม TURN ON หลอดไฟทั้ง 2 ห้องสถานะของหลอดไฟจะถูกเปลี่ยนเป็น “ON” และปุ่มกดจะเปลี่ยนเป็น TURN OFF ซึ่งหลอดไฟจะเปิดตามรูปที่ 4.12 (ข) ซึ่งหากผู้ทดลองต้องการจะปิดหลอดไฟสามารถกดปุ่ม TURN OFF หลอดไฟทั้ง 2 ห้องจะปิดตามที่ต้องการ และสถานะถูกเปลี่ยนเป็น “OFF” ดังรูปที่ 4.14

### Controlling Electronic Devices Using GUI



(ก)



(ข)

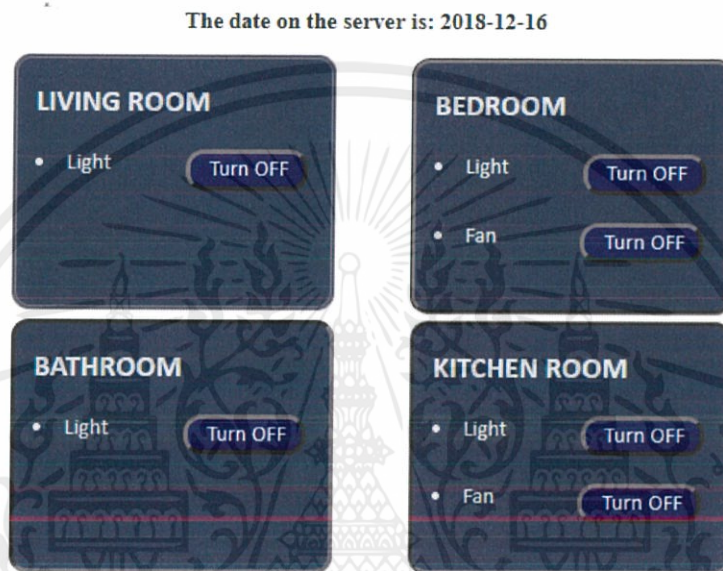
รูปที่ 4.14 (ก) และ (ข) GUI สั่งปิดไฟห้องนั่งเล่นและห้องน้ำ

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

#### 4.2.2 ผลการทดสอบ GUI แสดงสถานะอุปกรณ์ตามคำสั่งเสียง

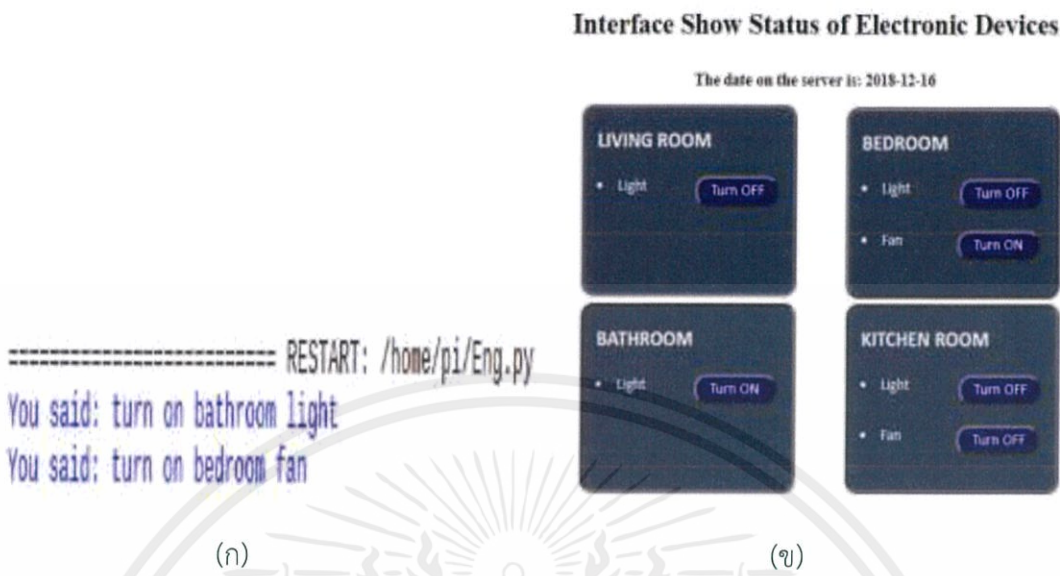
การทดสอบ GUI ในส่วนนี้จะทำงานคู่กับคำสั่งเสียง ซึ่งทำหน้าที่แสดงสถานะการใช้งานของอุปกรณ์ไฟฟ้าผ่านทางเว็บเพจที่สร้างจากไฟล์ HTML ดังรูปที่ 4.15

### Interface Show Status of Electronic Devices



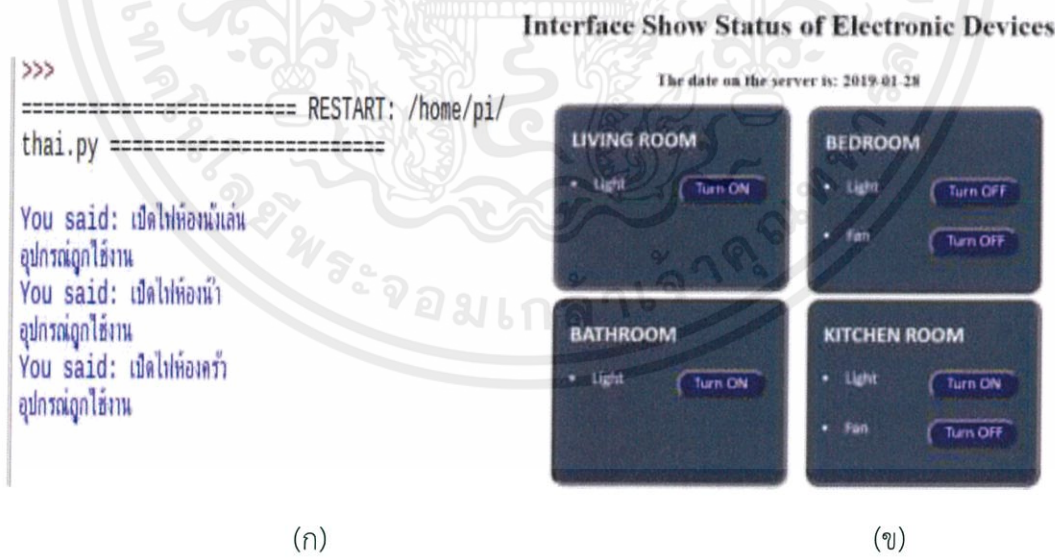
รูปที่ 4.15 GUI แสดงสถานะของอุปกรณ์ไฟฟ้า

จากรูปที่ 4.15 เป็น GUI แสดงสถานะของอุปกรณ์ ซึ่งทำงานควบคู่กับคำสั่งเสียง เมื่อมีคำสั่งให้เปิดการทำงานของอุปกรณ์ สถานะจะถูกเปลี่ยนเป็น “Turn on” หรือมีคำสั่งให้ปิดการทำงานของอุปกรณ์ สถานะจะถูกเปลี่ยนเป็น “Turn off” ดังรูปที่ 4.16



รูปที่ 4.16 (ก) และ (ข) GUI แสดงการเปลี่ยนสถานะอุปกรณ์ไฟฟ้าจากคำสั่งเสียงภาษาอังกฤษ

จากรูป 4.16 (ก) ผู้ทดลองสั่งคำสั่งเสียงภาษาอังกฤษคำว่า “turn on bedroom fan” และ “turn on bathroom light” เพื่อเปิดการใช้งานพัดลมห้องนอนและหลอดไฟห้องน้ำ ซึ่งสถานะการทำงานเปลี่ยนไปตามคำสั่ง ดังรูปที่ 4.17 (ข)



รูปที่ 4.17 (ก) และ (ข) GUI แสดงการเปลี่ยนสถานะอุปกรณ์ไฟฟ้าจากคำสั่งเสียงภาษาไทย

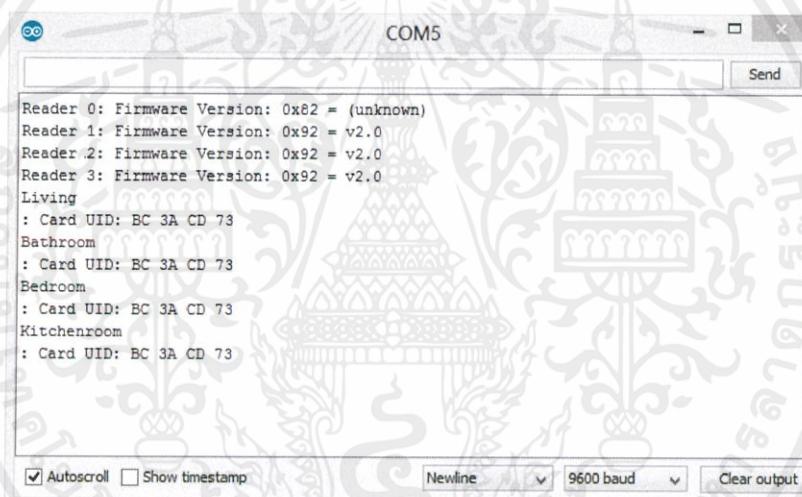
เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูป 4.17 (ก) ผู้ทดลองส่งคำสั่งเสียงภาษาไทยคำว่า “เปิดไฟห้องนั่งเล่น”, “เปิดไฟห้องน้ำ” และ “เปิดไฟห้องครัว” เพื่อเปิดการใช้งานหลอดไฟทั้ง 3 ห้อง ได้แก่ ห้องนั่งเล่น, ห้องน้ำ และห้องครัว ซึ่งสถานะการทำงานเปลี่ยนไปตามคำสั่ง ดังรูปที่ 4.17 (ข)

### 4.3 ผลการทดสอบระบบยืนยันตัวตนด้วย RFID-RC522

#### 4.3.1 ผลการทดสอบความแม่นยำการอ่านป้ายอิเล็กทรอนิกส์ (RFID Tag)

การทดสอบความแม่นยำในการอ่านป้ายอิเล็กทรอนิกส์ (RFID Tag) ซึ่งทำหน้าที่ตรวจจับป้ายอิเล็กทรอนิกส์ (RFID Tag) โดยมีเครื่องอ่านสัญญาณ RFID จำนวน 4 ตัวที่แทนการเข้าใช้งานในแต่ละห้อง ซึ่งการทดสอบจะมีข้อมูล UID ของป้ายอิเล็กทรอนิกส์ และชื่อของเครื่องอ่านที่ตรวจจับป้ายอิเล็กทรอนิกส์ ดังรูปที่ 4.18



รูปที่ 4.18 ข้อมูล RFID Tag และเครื่องอ่านสัญญาณ RFID

จากรูปที่ 4.18 ผู้ทดลองนำป้ายอิเล็กทรอนิกส์ไปแตะเครื่องอ่านสัญญาณ RFID ในแต่ละตัว จำนวน 4 ตัว โดยถ้าเครื่องอ่านสัญญาณ RFID อ่านสัญญาณ ได้แก่ Reader 0 จะแทนการเข้าใช้งานห้องนั่งเล่น, Reader 1 แทนการเข้าใช้งานห้องอาบน้ำ, Reader 2 แทนการเข้าใช้งานห้องนอนและ Reader 3 แทนการเข้าใช้งานห้องครัว เป็นต้น ซึ่งการระบุการเข้าใช้งานในแต่ละห้องจะสามารถใช้ในการสั่งงานอุปกรณ์ไฟฟ้าในแต่ละห้องได้

### 4.3.2 ผลการทดสอบความถูกต้องระบบยืนยันตัวตนสำหรับเข้าใช้งานอุปกรณ์ไฟฟ้าภายในแต่ละห้อง

การทดสอบความถูกต้องในระบบยืนยันตัวตนด้วย RFID สำหรับการเข้าใช้งานอุปกรณ์ไฟฟ้า ซึ่งระบบจะตรวจสอบจากเครื่องอ่านที่ผู้ใช้งานจึงสามารถเข้าใช้งานอุปกรณ์ไฟฟ้าในห้องนั้น และหากผู้ใช้งานออกจากห้องอุปกรณ์ทุกอย่างภายในห้องจะถูกปิดการใช้งาน จึงไม่สามารถใช้งานได้ แม้ว่าผู้ใช้งานจะสั่งให้อุปกรณ์ไฟฟ้าทำงาน

b''  
b''  
b'Living\r\n'  
Welcome to your living room  
You said: body turn on light  
turn on  
You said: body goodbye  
b': Card UID: BC 3A CD 73\r\n'  
b''  
b''

(ก)

b''  
b''  
b'Living\r\n'  
พร้อมรับคำสั่งการเข้าใช้งานห้องนั่งเล่น  
You said: มาลีเปิดไฟ  
turn on  
You said: มาลีไว้อีกกัน  
b': Card UID: BC 3A CD 73\r\n'  
b''  
b''

(ข)

รูปที่ 4.19 (ก) และ (ข) การเข้าใช้งานห้องนั่งเล่น

จากรูปที่ 4.18 ผู้ทดลองนำป้ายอิเล็กทรอนิกส์ (RFID Tag) แตะเครื่องอ่านสำหรับห้องนั่งเล่นเพื่อเข้าใช้งานอุปกรณ์ในห้อง ซึ่งรูปที่ 4.19 (ก) เป็นคำสั่งภาษาอังกฤษโดยคำสั่งต้องมีคำว่า “Body” ระบบจึงทำตามคำสั่ง ส่วนรูปที่ 4.19 (ข) เป็นคำสั่งภาษาไทยโดยคำสั่งต้องมีคำว่า “มาลี” ระบบจึงทำตามคำสั่ง ซึ่งทำให้หลอดไฟสว่าง ดังรูปที่ 4.20



รูปที่ 4.20 อุปกรณ์ไฟฟ้าในห้องนั่งเล่น

```

b''
b''
b'Bathroom\r\n'
Welcome to your bathroom
You said: body turn on light
turn on
You said: body goodbye
b': Card UID: BC 3A CD 73\r\n'
b''
b''

```

(ก)

```

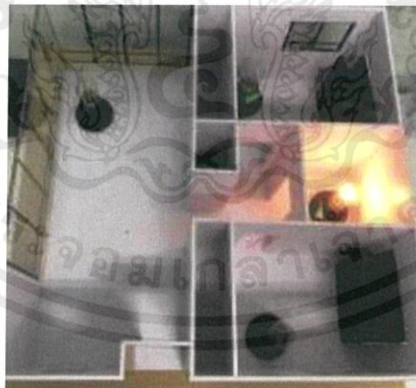
b'Bathroom\r\n'
พร้อมเข้าใช้งานห้องน้ำ
You said: มาลีเปิดไฟ
turn on
You said: มาลีไว้ออกกัน
b': Card UID: BC 3A CD 73\r\n'
b''
b''

```

(ข)

#### รูปที่ 4.21 (ก) และ (ข) การเข้าใช้งานห้องน้ำ

จากรูปที่ 4.21 ผู้ทดลองนำป้ายอิเล็กทรอนิกส์ (RFID Tag) แตะเครื่องอ่านสำหรับห้องน้ำเพื่อเข้าใช้งานอุปกรณ์ในห้อง ซึ่งรูปที่ 4.21 (ก) เป็นคำสั่งภาษาอังกฤษโดยคำสั่งต้องมีคำว่า “Body” ระบบจึงทำตามคำสั่ง หากผู้ใช้งานต้องการที่จะออกจากห้องน้ำและปิดอุปกรณ์ทุกอย่างภายในห้อง ผู้ทดลองจึงพูดว่า “Goodbye” ส่วนรูปที่ 4.21 (ข) เป็นคำสั่งภาษาไทยโดยคำสั่งต้องมีคำว่า “มาลี” ระบบจึงทำตามคำสั่ง ซึ่งทำให้หลอดไฟสว่าง ดังรูปที่ 4.22 หากผู้ใช้งานต้องการที่จะออกจากห้องน้ำและปิดอุปกรณ์ทุกอย่างภายในห้อง ผู้ทดลองจึงพูดว่า “ไว้ออกกัน”



รูปที่ 4.22 อุปกรณ์ไฟฟ้าในห้องน้ำ

```

b'Bedroom\r\n'
Welcome to your bedroom
You said: turn on fan body
turn on
You said: turn on light body
turn on
You said: goodbye body
b': Card UID: BC 3A CD 73\r\n'
b''
b''

```

(ก)

```

b''
b'Bedroom\r\n'
พร้อมรับคำสั่งการใช้งานห้องนอน
You said: มาลีเปิดพัดลม
turn on
You said: มาลีเปิดไฟ
turn on
You said: มาลีไว้ออกกัน
b': Card UID: BC 3A CD 73\r\n'
b''

```

(ข)

### รูปที่ 4.23 (ก) และ (ข) การใช้งานห้องนอน

จากรูปที่ 4.23 ผู้ทดลองนำป้ายอิเล็กทรอนิกส์ (RFID Tag) แตะเครื่องอ่านสำหรับห้องนอนเพื่อใช้งานอุปกรณ์ในห้อง ซึ่งรูปที่ 4.23 (ก) เป็นคำสั่งภาษาอังกฤษโดยคำสั่งต้องมีคำว่า “Body” ระบบจึงทำตามคำสั่ง หากผู้ใช้งานต้องการที่จะออกจากห้องนอนและปิดอุปกรณ์ทุกอย่างภายในห้อง ผู้ทดลองจึงพูดว่า “Goodbye” ส่วนรูปที่ 4.23 (ข) เป็นคำสั่งภาษาไทยโดยคำสั่งต้องมีคำว่า “มาลี” ระบบจึงทำตามคำสั่ง ซึ่งทำให้หลอดไฟสว่างและพัดลมทำงาน ดังรูปที่ 4.24 หากผู้ใช้งานต้องการที่จะออกจากห้องนอนและปิดอุปกรณ์ทุกอย่างภายในห้อง ผู้ทดลองจึงพูดว่า “ไว้ออกกัน”



รูปที่ 4.24 อุปกรณ์ไฟฟ้าห้องนอน

```

b''
b'Kitchenroom\r\n'
Welcome to your kitchenroom
You said: turn on fan body
turn on
You said: turn on light body
turn on
You said: goodbye body
b': Card UID: BC 3A CD 73\r\n'
b''

```

(ก)

```

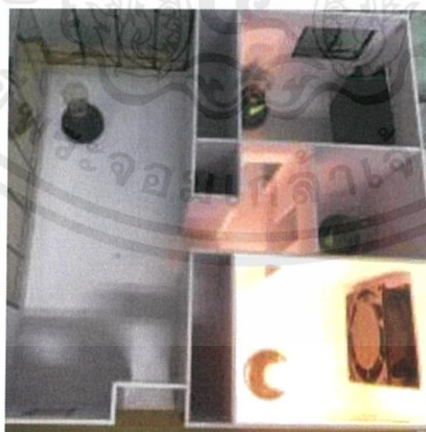
b''
b'Kitchenroom\r\n'
พร้อมเข้าใช้งานห้องครัวพร้อมเข้าใช้งานห้องครัว
You said: มาลิเปิดพัดลม
turn on
You said: มาลิเปิดไฟ
turn on
You said: มาลิไว้เจอกัน
b': Card UID: BC 3A CD 73\r\n'
b''

```

(ข)

รูปที่ 4.25 (ก) และ (ข) การเข้าใช้งานห้องครัว

จากรูปที่ 4.25 ผู้ทดลองนำป้ายอิเล็กทรอนิกส์ (RFID Tag) และเครื่องอ่านสำหรับห้องนอนเพื่อเข้าใช้งานอุปกรณ์ในห้อง ซึ่งรูปที่ 4.25 (ก) เป็นคำสั่งภาษาอังกฤษโดยคำสั่งต้องมีคำว่า “Body” ระบบจึงทำตามคำสั่ง หากผู้ใช้งานต้องการที่จะออกจากห้องนอนและปิดอุปกรณ์ทุกอย่างภายในห้อง ผู้ทดลองจึงพูดว่า “Goodbye” ส่วนรูปที่ 4.25 (ข) เป็นคำสั่งภาษาไทยโดยคำสั่งต้องมีคำว่า “มาลิ” ระบบจึงทำตามคำสั่ง ซึ่งทำให้หลอดไฟสว่างและพัดลมทำงาน ดังรูปที่ 4.26 หากผู้ใช้งานต้องการที่จะออกจากห้องน้ำและปิดอุปกรณ์ทุกอย่างภายในห้อง ผู้ทดลองจึงพูดว่า “ไว้เจอกัน”



รูปที่ 4.26 อุปกรณ์ไฟฟ้าห้องครัว

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## บทที่ 5

### สรุปผลและข้อเสนอแนะ

#### 5.1 สรุปผล

ปริญญานิพนธ์ได้นำเสนอการนำเอา Internet of Things มาประยุกต์ใช้กับการควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าไฟฟ้าภายในบ้าน จนพัฒนากลายเป็นระบบสั่งการใช้งานอุปกรณ์ไฟฟ้าโดยการสั่งงานด้วยเสียง และระบบการยืนยันตัวตนเพิ่มการรักษาความปลอดภัยในการใช้งานด้วยการเพิ่ม การยืนยันตัวตนของผู้ใช้งานด้วย RFID โดยในการทดลอง ทางผู้จัดทำได้ออกแบบการสั่งการอุปกรณ์ไว้ด้วยกันอยู่สองวิธีคือ

1) การสั่งการผ่านระบบ Graphic User Interface (GUI) เป็นการเขียน GUI ขึ้นมาจากการใช้ภาษา HTML เพื่อเป็นหน้าที่ใช้กดคำสั่ง เปิด-ปิด ของอุปกรณ์ และยัง สามารถแสดง สถานการณ์ทำงานของอุปกรณ์ว่าอุปกรณ์นั้นทำงานหรือไม่ได้ ทำงานอยู่ ภายใต้เงื่อนไขว่า GUI กับ Raspberry Pi ต้องอยู่ภายในวง LAN เดียวกัน จึงสามารถใช้งานได้

2) การสั่งการด้วยเสียง ผ่านระบบ Google speech recognition เป็นการพัฒนาจากการใช้งานแบบ GUI โดยการเพิ่มความสามารถการสั่งการด้วยเสียงด้วยการพูดผ่าน ตัวรับสัญญาณเสียง และนำไปผ่านกระบวนการแปลงจากเสียงเป็นคำพูดของ Google speech recognition เพื่อนำการแปลงเสียงที่ได้มาเป็นตัวอักษรนั้น ไปเข้าการประมวลผลของ Raspberry - Pi เพื่อไปสั่งการ Relay module และแสดงผลออกมาด้วยการ เปิดหรือปิด อุปกรณ์ที่เราสั่งการ

#### 5.2 ข้อเสนอแนะ

1) การควบคุมระยะไกลยังทำไม่ได้เนื่องจากปัญหาของ GUI กับ Raspberry Pi จำเป็นต้องอยู่ภายในวง LAN เดียวกัน ทำให้ไม่สามารถสั่งการจากภายนอกได้ ถ้าสามารถทำให้ควบคุมจากระยะไกลได้จะทำให้ สามารถเลือกวิธีการสั่งการทั้งนอกบ้านและในบ้านได้

2) ในการสั่งการด้วยเสียงจำเป็นต้องพูดให้ใกล้กับไมโครโฟนมากที่สุดเพื่อให้ Google Speech Recognition นั้นสามารถแปลงจากเสียงเป็นตัวอักษรได้แม่นยำมากขึ้น และแม้พูดใกล้ไมโครโฟนแล้ว ก็อาจเกิดความผิดพลาดจากเสียงรบกวน

3) เปลี่ยนไมโครโฟนแบบสายที่ใช้ ให้เป็นแบบไร้สายเพื่อให้สามารถเดินไปด้วย และสั่งการไปพร้อมกันได้ เพื่อแก้ปัญหาในการสั่งการด้วยเสียงและลดปัญหาสัญญาณรบกวน

## บรรณานุกรม

- [1] Marcelo Rovai. 2561. “Python Webserver with Flask and Raspberry Pi.”  
<https://medium.com/@rovai/python-webserver-with-flask-and-raspberry-pi-398423cc6f5d>.
- [2] Pypi. 2561. “SpeechRecognition2.0.0.”  
<https://pypi.org/project/SpeechRecognition/2.0.0/#data>.
- [3] Veedvil. 2561. “ทำความเข้าใจเรื่อง Internet of things.”  
<http://www.veedvil.com/news/internet-of-things-iot>.
- [4] Saixiii. 2561. “Flask คืออะไร.” <https://saixiii.com/python-flask-web-application>.
- [5] Education preecha. 2561. “คำสั่งในภาษา HTML.”  
<http://www.education.preecha.net/ht.html>.
- [6] Enjoyday.net. 2561. “โครงสร้างคำสั่ง.”  
[http://www.enjoyday.net/webtutorial/css/css\\_chapter02.html](http://www.enjoyday.net/webtutorial/css/css_chapter02.html).
- [7] Wikipedia. 2561. “Graphical user interface”  
[https://en.wikipedia.org/wiki/Graphical\\_user\\_interface](https://en.wikipedia.org/wiki/Graphical_user_interface).
- [8] Etinker. 2561. “How to control Raspberry Pi GPIO via http web server.”  
<https://www.e-tinkers.com/2018/04/how-to-control-raspberry-pi-gpio-via-http-web-server>.
- [9] Mktk1117. 2561. “Communication between Arduino and Raspberry Pi.”  
[https://github.com/mktk1117/six\\_wheel\\_robot/wiki/Communication-between-Arduino-and-Raspberry-Pi](https://github.com/mktk1117/six_wheel_robot/wiki/Communication-between-Arduino-and-Raspberry-Pi).
- [10] PSP Tech. 2561. “รีเลย์ (Relay) คืออะไร.”  
<http://www.psptech.co.th/%E0%B8%A3%E0%B8%B5%E0%B9%80%E0%B8%A5%E0%B8%A2%E0%B9%8Crelay%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3-15696.page>.
- [11] Wikipedia. 2561. “Radio-frequency identification”  
[https://en.wikipedia.org/wiki/Radio-frequency\\_identification](https://en.wikipedia.org/wiki/Radio-frequency_identification).
- [12] วิไลลักษณ์. 2561. “การทำงานของระบบ RFID”  
[http://rfiddatacom.blogspot.com/2008/09/rfid\\_2832.html](http://rfiddatacom.blogspot.com/2008/09/rfid_2832.html).



เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 1. Source code ในระบบคำสั่งเสียงภาษาอังกฤษ

```

import speech_recognition as sr          #เรียกใช้งาน speech_recognition library
import RPi.GPIO as GPIO                 #เรียกใช้งาน RPi.GPIO library
import serial                            #เรียกใช้งาน serial library
import time                             #เรียกใช้งาน time library

ser = serial.Serial('/dev/ttyUSB0',timeout=0.1)    #เก็บข้อมูลจาก USB port 0
ser.flushInput

GPIO.setmode(GPIO.BCM)                  #กำหนดรูปแบบขา GPIO ตามชนิด BCM
GPIO.setwarnings(False)

Liv = ('light')                          #กำหนดให้ Liv คือคำว่า "light"
GPIO.setup(13,GPIO.OUT)                  #กำหนดขา 13 เป็นเอาต์พุต
GPIO.output(13,0)                        #กำหนดขา 13 มีค่าเป็น 0
Bed = ('fan','light')                   #กำหนดให้ Bed มีคำว่า "light" และ "fan"
GPIO.setup(20,GPIO.OUT)                  #กำหนดขา 20 เป็นเอาต์พุต
GPIO.output(20,0)                        #กำหนดขา 20 มีค่าเป็น 0
GPIO.setup(21,GPIO.OUT)                  #กำหนดขา 21 เป็นเอาต์พุต
GPIO.output(21,0)                        #กำหนดขา 21 มีค่าเป็น 0
Bath = ('light')                         #กำหนดให้ Bath มีคำว่า "light"
GPIO.setup(19,GPIO.OUT)                  #กำหนดขา 19 เป็นเอาต์พุต
GPIO.output(19,0)                        #กำหนดขา 19 มีค่าเป็น 0
Kit = ('fan','light')                   #กำหนดให้ Kit มีคำว่า "light" และ "fan"
GPIO.setup(16,GPIO.OUT)                  #กำหนดขา 16 เป็นเอาต์พุต
GPIO.output(16,0)                        #กำหนดขา 19 มีค่าเป็น 0
GPIO.setup(26,GPIO.OUT)                  #กำหนดขา 26 เป็นเอาต์พุต
GPIO.output(26,0)                        #กำหนดขา 26 มีค่าเป็น 0

r = sr.Recognizer()                     #ฟังก์ชันการแปลงเสียงเป็นข้อความ
m = sr.Microphone()                     #ฟังก์ชันรับเสียงจากไมโครโฟน

while True:                             #สร้างลูปเพื่อรับคำสั่งตลอดเวลา
    inputValue = ser.readline()          #รับค่า RFID Reader เก็บไว้ใน inputValue
    Value = format(inputValue)           #เก็บไว้ในรูปแบบตัวอักษร
    print(Value)                         #แสดงค่าที่เก็บไว้ใน Value

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

if 'Living' in Value:                                #ถ้าค่าใน Value เป็นคำว่า 'Living'
    print("Welcome to your living room")             #แสดง "Welcome to your living room"
    while True:                                       #สร้างลูปให้รับคำสั่งเสียงวนตลอด
        with sr.Microphone() as source:
            audio = r.listen(source)                 #รับเสียงจากไมโครโฟน
            try:                                       #แปลงเสียงเป็นข้อความ
                text = (" +r.recognize_google(audio,language = "en-EN"))
                print("You said: {}".format(text))    #แสดงข้อความที่ถูกแปลง
                if Liv and 'body' in text:            #ถ้าในข้อความมีคำว่า 'Body' และ 'ไฟ'
                    if 'on' in text:                 #ถ้ามีคำว่า 'on' ในข้อความ
                        GPIO.output(13,1)            #GPIO 13 จะทำงาน
                        print("turn on")              #แสดงคำว่า "Turn on"
                    elif 'off' in text:              #ถ้ามีคำว่า 'off' ในข้อความ
                        GPIO.output(13,0)            #GPIO 13 จะไม่ทำงาน
                        print("turn off")             #แสดงคำว่า "Turn off"
                if 'body' and 'goodbye' in text:      #ถ้าข้อความมีคำว่า 'Body' และ 'goodbye'
                    GPIO.output(13,0)                #GPIO 13 จะไม่ทำงาน
                    Break                             #ออกจากการสั่งในห้องนั่งเล่น
if 'Bedroom' in Value:                               #ถ้าค่าใน Value เป็นคำว่า 'Bedroom'
    print("Welcome to your bedroom")                 #แสดง "Welcome to your bedroom"
    while True:                                       #สร้างลูปให้รับคำสั่งเสียงวนตลอด
        with sr.Microphone() as source:
            audio = r.listen(source)                 #รับเสียงจากไมโครโฟน
            try:                                       #แปลงเสียงเป็นข้อความ
                text = (" +r.recognize_google(audio,language = "en-EN"))
                print("You said: {}".format(text))    #แสดงข้อความที่ถูกแปลง
                if 'body' in text:                   #ถ้าในข้อความมีคำว่า 'Body'
                    if 'fan' and 'on' in text:      #ถ้ามีคำว่า 'fan' และ 'on' ในข้อความ
                        GPIO.output(20,1)           #GPIO 20 จะทำงาน
                        print("turn on")             #แสดงคำว่า "Turn on"
                    elif 'fan' and 'off' in text:   #ถ้ามีคำว่า 'fan' และ 'off' ในข้อความ

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

GPIO.output(20,0)      #GPIO 20 จะไม่ทำงาน
print("turn off")      #แสดงคำว่า "Turn off"
elif 'light' and 'on' in text: #ถ้ามีคำว่า 'light' และ 'on' ในข้อความ
    GPIO.output(21,1)   #GPIO 21 จะทำงาน
    print("turn on")    #แสดงคำว่า "Turn on"
elif 'light' and 'off' in text: #ถ้ามีคำว่า 'light' และ 'off' ในข้อความ
    GPIO.output(21,0)   #GPIO 21 จะไม่ทำงาน
    print("turn off")   #แสดงคำว่า "Turn off"
if 'body' and 'goodbye' in text: #ถ้าข้อความมีคำว่า 'Body' และ 'goodbye'
    GPIO.output(20,0)    #GPIO 20 จะไม่ทำงาน
    GPIO.output(21,0)    #GPIO 21 จะไม่ทำงาน
    Break               #ออกจากการสั่งในห้องนอน
if 'Bathroom' in Value:   #ถ้าค่าใน Value เป็นคำว่า 'Bathroom'
    print("Welcome to your bathroom") #แสดง "Welcome to your bathroom"
while True:              #สร้างลูปให้รับคำสั่งเสียงวนตลอด
    with sr.Microphone() as source:
        audio = r.listen(source)      #รับเสียงจากไมโครโฟน
    try:                               #แปลงเสียงเป็นข้อความ
        text = (" +r.recognize_google(audio,language = "en-EN"))
        print("You said: {}".format(text)) #แสดงข้อความที่ถูกแปลง
    if Bath and 'body' in text:         #ถ้าในข้อความมีคำว่า 'Body' และ 'ไฟ'
        if 'on' in text:               #ถ้ามีคำว่า 'on' ในข้อความ
            GPIO.output(19,1)          #GPIO 19 จะทำงาน
            print("turn on")           #แสดงคำว่า "Turn on"
        elif 'off' in text:            #ถ้ามีคำว่า 'off' ในข้อความ
            GPIO.output(19,0)          #GPIO 19 จะไม่ทำงาน
            print("turn off")          #แสดงคำว่า "Turn off"
        if 'body' and 'goodbye' in text: #ถ้าข้อความมีคำว่า 'Body' และ 'goodbye'
            GPIO.output(19,0)          #GPIO 19 จะไม่ทำงาน
            Break                     #ออกจากการสั่งในห้องนั่งเล่น
if 'Kitchenroom' in Value:            #ถ้าค่าใน Value เป็นคำว่า 'Kitchenroom'

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

print("Welcome to your kitchen room")#แสดง "Welcome to your kitchen room"
while True:                                #สร้างลูปให้รับคำสั่งเสียงวนตลอด
    with sr.Microphone() as source:
        audio = r.listen(source)            #รับเสียงจากไมโครโฟน
    try:                                    #แปลงเสียงเป็นข้อความ
        text = (" +r.recognize_google(audio,language = "en-EN"))
        print("You said: {}".format(text))    #แสดงข้อความที่ถูกแปลง
        if 'body' in text:                    #ถ้าในข้อความมีคำว่า 'Body'
            if 'fan' and 'on' in text:        #ถ้ามีคำว่า 'fan' และ 'on' ในข้อความ
                GPIO.output(16,1)            #GPIO 16 จะทำงาน
                print("turn on")              #แสดงคำว่า "Turn on"
            elif 'fan' and 'off' in text:     #ถ้ามีคำว่า 'fan' และ 'off' ในข้อความ
                GPIO.output(16,0)            #GPIO 16 จะไม่ทำงาน
                print("turn off")            #แสดงคำว่า "Turn off"
            elif 'light' and 'on' in text:    #ถ้ามีคำว่า 'light' และ 'on' ในข้อความ
                GPIO.output(26,1)            #GPIO 26 จะทำงาน
                print("turn on")              #แสดงคำว่า "Turn on"
            elif 'light' and 'off' in text:   #ถ้ามีคำว่า 'light' และ 'off' ในข้อความ
                GPIO.output(26,0)            #GPIO 26 จะไม่ทำงาน
                print("turn off")            #แสดงคำว่า "Turn off"
        if 'body' and 'goodbye' in text:     #ถ้าข้อความมีคำว่า 'Body' และ 'goodbye'
            GPIO.output(16,0)                #GPIO 16 จะไม่ทำงาน
            GPIO.output(26,0)                #GPIO 26 จะไม่ทำงาน
        Break                                #ออกจากการสั่งในห้องครัว

```

## 2. Source code ในระบบคำสั่งเสียงภาษาไทย

```

import speech_recognition as sr            #เรียกใช้งาน speech_recognition library
import RPi.GPIO as GPIO                   #เรียกใช้งาน RPi.GPIO library
import serial                              #เรียกใช้งาน serial library
import time                               #เรียกใช้งาน time library
ser = serial.Serial('/dev/ttyUSB0',timeout=0.1) #เก็บข้อมูลจาก USB port 0

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

ser.flushInput
GPIO.setmode(GPIO.BCM)           #กำหนดรูปแบบขา GPIO ตามชนิด BCM
GPIO.setwarnings(False)
Liv = ('ไฟ')                      #กำหนดให้ Liv คือคำว่า "ไฟ"
GPIO.setup(13,GPIO.OUT)           #กำหนดขา 13 เป็นเอาต์พุต
GPIO.output(13,0)                 #กำหนดขา 13 มีค่าเป็น 0
Bed = ('พัสดม', 'ไฟ')           #กำหนดให้ Bed มีคำว่า "ไฟ" และ "พัสดม"
GPIO.setup(20,GPIO.OUT)          #กำหนดขา 20 เป็นเอาต์พุต
GPIO.output(20,0)                 #กำหนดขา 20 มีค่าเป็น 0
GPIO.setup(21,GPIO.OUT)          #กำหนดขา 21 เป็นเอาต์พุต
GPIO.output(21,0)                 #กำหนดขา 21 มีค่าเป็น 0
Bath = ('ไฟ')                    #กำหนดให้ Bath มีคำว่า "ไฟ"
GPIO.setup(19,GPIO.OUT)          #กำหนดขา 19 เป็นเอาต์พุต
GPIO.output(19,0)                 #กำหนดขา 19 มีค่าเป็น 0
Kit = ('พัสดม', 'ไฟ')           #กำหนดให้ Kit มีคำว่า "ไฟ" และ "พัสดม"
GPIO.setup(16,GPIO.OUT)          #กำหนดขา 16 เป็นเอาต์พุต
GPIO.output(16,0)                 #กำหนดขา 19 มีค่าเป็น 0
GPIO.setup(26,GPIO.OUT)          #กำหนดขา 26 เป็นเอาต์พุต
GPIO.output(26,0)                 #กำหนดขา 26 มีค่าเป็น 0
r = sr.Recognizer()              #ฟังก์ชันการแปลงเสียงเป็นข้อความ
m = sr.Microphone()              #ฟังก์ชันรับเสียงจากไมโครโฟน
while True:                      #สร้างลูปเพื่อรับคำสั่งตลอดเวลา
    inputValue = ser.readline()   #รับค่า RFID Reader เก็บไว้ใน inputValue
    Value = format(inputValue)    #เก็บไว้ในรูปแบบตัวอักษร
    print(Value)                  #แสดงค่าที่เก็บไว้ใน Value
    if 'Living' in Value:         #ถ้าค่าใน Value เป็นคำว่า 'Living'
        print("Welcome to your living room") #แสดง "Welcome to your living room"
        while True:              #สร้างลูปให้รับคำสั่งเสียงวนตลอด
            with sr.Microphone() as source:
                audio = r.listen(source) #รับเสียงจากไมโครโฟน
                try:                  #แปลงเสียงเป็นข้อความ

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

text = (" +r.recognize_google(audio,language = "en-EN"))
print("You said: {}".format(text))      #แสดงข้อความที่ถูกละเปลี่ยน
if Liv and 'มาลี' in text: #ถ้าในข้อความมีคำว่า 'มาลี' และ 'ไฟ'
    if 'เปิด' in text:      #ถ้ามีคำว่า 'เปิด' ในข้อความ
        GPIO.output(13,1)  #GPIO 13 จะทำงาน
        print("turn on")   #แสดงคำว่า "Turn on"
    elif 'ปิด' in text:     #ถ้ามีคำว่า 'ปิด' ในข้อความ
        GPIO.output(13,0)  #GPIO 13 จะไม่ทำงาน
        print("turn off")  #แสดงคำว่า "Turn off"
    if 'มาลี' and 'ไว้อีกกัน' in text: #ถ้าข้อความมีคำว่า 'มาลี' และ 'ไว้อีกกัน'
        GPIO.output(13,0)  #GPIO 13 จะไม่ทำงาน
        Break              #ออกจากการสั่งในห้องนั่งเล่น
if 'Bedroom' in Value:     #ถ้าค่าใน Value เป็นคำว่า 'Bedroom'
    print("Welcome to your bedroom") #แสดง "Welcome to your bedroom"
while True:                #สร้างลูปให้รับคำสั่งเสียงวนตลอด
    with sr.Microphone() as source:
        audio = r.listen(source)    #รับเสียงจากไมโครโฟน
    try:                            #แปลงเสียงเป็นข้อความ
        text = (" +r.recognize_google(audio,language = "en-EN"))
        print("You said: {}".format(text))      #แสดงข้อความที่ถูกละเปลี่ยน
        if 'มาลี' in text:                      #ถ้าในข้อความมีคำว่า 'มาลี'
            if 'พัดลม' and 'เปิด' in text:      #ถ้ามีคำว่า 'พัดลม' และ 'เปิด' ในข้อความ
                GPIO.output(20,1)               #GPIO 20 จะทำงาน
                print("turn on")                #แสดงคำว่า "Turn on"
            elif 'พัดลม' and 'ปิด' in text:      #ถ้ามีคำว่า 'พัดลม' และ 'ปิด' ในข้อความ
                GPIO.output(20,0)               #GPIO 20 จะไม่ทำงาน
                print("turn off")               #แสดงคำว่า "Turn off"
            elif 'ไฟ' and 'เปิด' in text:        #ถ้ามีคำว่า 'ไฟ' และ 'เปิด' ในข้อความ
                GPIO.output(21,1)               #GPIO 21 จะทำงาน
                print("turn on")                #แสดงคำว่า "Turn on"
            elif 'ไฟ' and 'ปิด' in text:         #ถ้ามีคำว่า 'ไฟ' และ 'ปิด' ในข้อความ

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

GPIO.output(21,0)          #GPIO 21 จะไม่ทำงาน
print("turn off")          #แสดงคำว่า "Turn off"
if 'มาลี' and 'ไว้ใจกัน' in text: #ถ้าข้อความมีคำว่า 'มาลี' และ 'ไว้ใจกัน'
    GPIO.output(20,0)      #GPIO 20 จะไม่ทำงาน
    GPIO.output(21,0)      #GPIO 21 จะไม่ทำงาน
    Break                  #ออกจากการสั่งในห้องนอน
if 'Bathroom' in Value:    #ถ้าค่าใน Value เป็นคำว่า 'Bathroom'
    print("Welcome to your bathroom") #แสดง "Welcome to your bathroom"
    while True:            #สร้างลูปให้รับคำสั่งเสียงวนตลอด
        with sr.Microphone() as source:
            audio = r.listen(source)    #รับเสียงจากไมโครโฟน
            try:                        #แปลงเสียงเป็นข้อความ
                text = (" +r.recognize_google(audio,language = "en-EN"))
                print("You said: {}".format(text))    #แสดงข้อความที่ถูกแปลง
            if Bath and 'มาลี' in text:    #ถ้าในข้อความมีคำว่า 'มาลี' และ 'ไฟ'
                if 'เปิด' in text:        #ถ้ามีคำว่า 'เปิด' ในข้อความ
                    GPIO.output(19,1)    #GPIO 19 จะทำงาน
                    print("turn on")      #แสดงคำว่า "Turn on"
                elif 'ปิด' in text:      #ถ้ามีคำว่า 'ปิด' ในข้อความ
                    GPIO.output(19,0)    #GPIO 19 จะไม่ทำงาน
                    print("turn off")    #แสดงคำว่า "Turn off"
            if 'มาลี' and 'ไว้ใจกัน' in text:    #ถ้าข้อความมีคำว่า 'มาลี' และ 'ไว้ใจกัน'
                GPIO.output(19,0)    #GPIO 19 จะไม่ทำงาน
                Break                #ออกจากการสั่งในห้องนั่งเล่น
if 'Kitchenroom' in Value:    #ถ้าค่าใน Value เป็นคำว่า 'Kitchenroom'
    print("Welcome to your kitchen room") #แสดง "Welcome to your kitchen room"
    while True:            #สร้างลูปให้รับคำสั่งเสียงวนตลอด
        with sr.Microphone() as source:
            audio = r.listen(source)    #รับเสียงจากไมโครโฟน
            try:                        #แปลงเสียงเป็นข้อความ
                text = (" +r.recognize_google(audio,language = "en-EN"))

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

print("You said: {}".format(text))      #แสดงข้อความที่ถูกลบ
if 'มาลี' in text:                      #ถ้าในข้อความมีคำว่า 'มาลี'
    if 'พัดลม' and 'เปิด' in text:      #ถ้ามีคำว่า 'พัดลม' และ 'เปิด' ในข้อความ
        GPIO.output(16,1)              #GPIO 16 จะทำงาน
        print("turn on")                #แสดงคำว่า "Turn on"
    elif 'พัดลม' and 'ปิด' in text:      #ถ้ามีคำว่า 'พัดลม' และ 'ปิด' ในข้อความ
        GPIO.output(16,0)              #GPIO 16 จะไม่ทำงาน
        print("turn off")               #แสดงคำว่า "Turn off"
    elif 'ไฟ' and 'เปิด' in text:        #ถ้ามีคำว่า 'ไฟ' และ 'เปิด' ในข้อความ
        GPIO.output(26,1)              #GPIO 26 จะทำงาน
        print("turn on")                #แสดงคำว่า "Turn on"
    elif 'ไฟ' and 'ปิด' in text:         #ถ้ามีคำว่า 'ไฟ' และ 'ปิด' ในข้อความ
        GPIO.output(26,0)              #GPIO 26 จะไม่ทำงาน
        print("turn off")               #แสดงคำว่า "Turn off"
if 'มาลี' and 'ไว้อีกกัน' in text:      #ถ้าข้อความมีคำว่า 'มาลี' และ 'มาลี'
    GPIO.output(16,0)                  #GPIO 16 จะไม่ทำงาน
    GPIO.output(26,0)                  #GPIO 26 จะไม่ทำงาน
Break                                  #ออกจากการสั่งในห้องครัว

```

### 3. Source code สำหรับสั่งการทำงานอุปกรณ์ไฟฟ้าผ่าน GUI

```

from flask import Flask, render_template #เรียกใช้ Flask Library
import RPi.GPIO as GPIO                 #เรียกใช้งาน RPi.GPIO library
app = Flask(__name__)                   #เข้าใช้งานฟังก์ชัน Flask
GPIO.setmode(GPIO.BCM)                  #ตั้งค่า GPIO ตามรูปแบบของ BCM
GPIO.setwarnings(False)
gpio = (4,9,10,17,22,27)                #กำหนดการให้ใช้งานขาของ GPIO
for pin in gpio:                         #สร้างลูปกำหนดชนิดรูปแบบของ GPIO
    GPIO.setup(pin,GPIO.OUT)             #ให้ทุกขาที่กำหนดเป็นเอาต์พุต
@app.route('/')
def interface():                          #ส่วนบอกสถานะอุปกรณ์ไฟฟ้า
    LivL = GPIO.input(4)                 #กำหนดให้ "LivL" รับค่าที่ขา GPIO 4

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

if LivL == 1:
    LivLight = 'ON'
else:
    LivLight = 'OFF'
BedF = GPIO.input(9)
if BedF == 1:
    BedFan = 'ON'
else:
    BedFan = 'OFF'

BedL = GPIO.input(10)
if BedL == 1:
    BedLight = 'ON'
else:
    BedLight = 'OFF'
BathL = GPIO.input(17)
if BathL == 1:
    BathLight = 'ON'
else:
    BathLight = 'OFF'
KitF = GPIO.input(22)
if KitF == 1:
    KitFan = 'ON'
else:
    KitFan = 'OFF'
KitL = GPIO.input(27)
if KitL == 1:
    KitLight = 'ON'
else:
    KitLight = 'OFF'
templateData = {

```

#สร้างเงื่อนไขถ้า “LivL” มีค่าเป็น 1  
 #ให้ “LivLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “LivL” มีค่าไม่เท่ากับ 1  
 #ให้ “LivLight” มีค่าเป็นคำว่า “OFF”  
 #กำหนดให้ “BedF” รับค่าที่ขา GPIO 9  
 #สร้างเงื่อนไขถ้า “BedF” มีค่าเป็น 1  
 #ให้ “BedFan” มีค่าเป็นคำว่า “ON”  
 #ถ้า “BedF” มีค่าไม่เท่ากับ 1  
 #ให้ “BedFan” มีค่าเป็นคำว่า “OFF”  
  
 #กำหนดให้ “BedL” รับค่าที่ขา GPIO 10  
 #สร้างเงื่อนไขถ้า “BedL” มีค่าเป็น 1  
 #ให้ “BedLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “BedL” มีค่าไม่เท่ากับ 1  
 #ให้ “BedLight” มีค่าเป็นคำว่า “OFF”  
 #กำหนดให้ “BathL” รับค่าที่ขา GPIO 17  
 #สร้างเงื่อนไขถ้า “BathL” มีค่าเป็น 1  
 #ให้ “BathLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “BathL” มีค่าไม่เท่ากับ 1  
 #ให้ “BathLight” มีค่าเป็นคำว่า “OFF”  
 #กำหนดให้ “KitF” รับค่าที่ขา GPIO 22  
 #สร้างเงื่อนไขถ้า “KitF” มีค่าเป็น 1  
 #ให้ “KitFan” มีค่าเป็นคำว่า “ON”  
 #ถ้า “KitF” มีค่าไม่เท่ากับ 1  
 #ให้ “KitFan” มีค่าเป็นคำว่า “OFF”  
 #กำหนดให้ “KitL” รับค่าที่ขา GPIO 27  
 #สร้างเงื่อนไขถ้า “KitL” มีค่าเป็น 1  
 #ให้ “KitLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “KitL” มีค่าไม่เท่ากับ 1  
 #ให้ “KitLight” มีค่าเป็นคำว่า “OFF”  
 #เก็บค่าและแทนค่าใน HTML

```

'LivL' : LivLight,
'BedF' : BedFan,
'BedL' : BedLight,
'BathL' : BathLight,
'KitL' : KitLight,
'KitF' : KitFan
}

return render_template('interface1.html' , **templateData) #นำค่าที่ได้ลงใน HTML
@app.route("/<deviceName>/<action>")
def action(deviceName, action):
    if deviceName == 'LivL':
        pin = 4
    if deviceName == 'BedF':
        pin = 9
    if deviceName == 'BedL':
        pin = 10
    if deviceName == 'BathL':
        pin = 17
    if deviceName == 'KitF':
        pin = 22
    if deviceName == 'KitL':
        pin = 27
    if action == "on":
        GPIO.output(pin,1)
    if action == "off":
        GPIO.output(pin,0)
    LivL = GPIO.input(4)
    if LivL == 1:
        LivLight = 'ON'
    else:
        LivLight = 'OFF'

```

#ใส่ค่า LivLight ลง LivL ใน HTML  
 #ใส่ค่า BedFan ลง BedF ใน HTML  
 #ใส่ค่า BedLight ลง BedL ใน HTML  
 #ใส่ค่า BathLight ลง BathL ใน HTML  
 #ใส่ค่า KitLight ลง KitL ใน HTML  
 #ใส่ค่า KitFan ลง KitF ใน HTML

#ส่วนควบคุมสถานะอุปกรณ์ไฟฟ้า  
 #ถ้า deviceName เป็น LivL  
 #ให้ pin เป็น 4  
 #ถ้า deviceName เป็น BedF  
 #ให้ pin เป็น 9  
 #ถ้า deviceName เป็น BedL  
 #ให้ pin เป็น 10  
 #ถ้า deviceName เป็น BathL  
 #ให้ pin เป็น 17  
 #ถ้า deviceName เป็น KitF  
 #ให้ pin เป็น 22  
 #ถ้า deviceName เป็น KitL  
 #ให้ pin เป็น 27  
 #ให้ action เป็น on  
 #ให้ GPIO นั้นทำงาน  
 #ให้ action เป็น off  
 #ให้ GPIO นั้น ไม่ทำงาน  
 #กำหนดให้ “LivL” รับค่าที่ขา GPIO 4  
 #สร้างเงื่อนไขถ้า “LivL” มีค่าเป็น 1  
 #ให้ “LivLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “LivL” มีค่าไม่เท่ากับ 1  
 #ให้ “LivLight” มีค่าเป็นคำว่า “OFF”

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
 ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

BedF = GPIO.input(9)
if BedF == 1:
    BedFan = 'ON'
else:
    BedFan = 'OFF'

BedL = GPIO.input(10)
if BedL == 1:
    BedLight = 'ON'
else:
    BedLight = 'OFF'

BathL = GPIO.input(17)
if BathL == 1:
    BathLight = 'ON'
else:
    BathLight = 'OFF'

KitF = GPIO.input(22)
if KitF == 1:
    KitFan = 'ON'
else:
    KitFan = 'OFF'

KitL = GPIO.input(27)
if KitL == 1:
    KitLight = 'ON'
else:
    KitLight = 'OFF'

templateData = {
    'LivL' : LivLight,
    'BedF' : BedFan,
    'BedL' : BedLight,
    'BathL' : BathLight,

```

#กำหนดให้ “BedF” รับค่าที่ขา GPIO 9  
 #สร้างเงื่อนไขถ้า “BedF” มีค่าเป็น 1  
 #ให้ “BedFan” มีค่าเป็นคำว่า “ON”  
 #ถ้า “BedF” มีค่าไม่เท่ากับ 1  
 #ให้ “BedFan” มีค่าเป็นคำว่า “OFF”

#กำหนดให้ “BedL” รับค่าที่ขา GPIO 10  
 #สร้างเงื่อนไขถ้า “BedL” มีค่าเป็น 1  
 #ให้ “BedLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “BedL” มีค่าไม่เท่ากับ 1  
 #ให้ “BedLight” มีค่าเป็นคำว่า “OFF”

#กำหนดให้ “BathL” รับค่าที่ขา GPIO 17  
 #สร้างเงื่อนไขถ้า “BathL” มีค่าเป็น 1  
 #ให้ “BathLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “BathL” มีค่าไม่เท่ากับ 1  
 #ให้ “BathLight” มีค่าเป็นคำว่า “OFF”

#กำหนดให้ “KitF” รับค่าที่ขา GPIO 22  
 #สร้างเงื่อนไขถ้า “KitF” มีค่าเป็น 1  
 #ให้ “KitFan” มีค่าเป็นคำว่า “ON”  
 #ถ้า “KitF” มีค่าไม่เท่ากับ 1  
 #ให้ “KitFan” มีค่าเป็นคำว่า “OFF”

#กำหนดให้ “KitL” รับค่าที่ขา GPIO 27  
 #สร้างเงื่อนไขถ้า “KitL” มีค่าเป็น 1  
 #ให้ “KitLight” มีค่าเป็นคำว่า “ON”  
 #ถ้า “KitL” มีค่าไม่เท่ากับ 1  
 #ให้ “KitLight” มีค่าเป็นคำว่า “OFF”

#เก็บค่าและแทนค่าใน HTML  
 #ใส่ค่า LivLight ลง LivL ใน HTML  
 #ใส่ค่า BedFan ลง BedF ใน HTML  
 #ใส่ค่า BedLight ลง BedL ใน HTML  
 #ใส่ค่า BathLight ลง BathL ใน HTML

```

'KitL' : KitLight,           #ใส่ค่า KitLight ลง KitL ใน HTML
'KitF' : KitFan              #ใส่ค่า KitFan ลง KitF ใน HTML
}

return render_template('interface1.html' , **templateData) #นำค่าที่ได้ลงใน HTML
if __name__ == '__main__':
    app.run(debug=True, port=80, host='0.0.0.0')

```

#### 4. Source code สำหรับเว็บเพจแสดงสถานะและสั่งการทำงานอุปกรณ์ไฟฟ้าผ่าน GUI

```

<!doctype html>
<html>
<head>
<style>
!-----รูปแบบการตกแต่งส่วนของห้องนั่งเล่น -----!
.Living{
    border:4px double gray;           !สร้างกรอบขนาด 4 px สีเทา
    border-radius:15px;               !มุมรอบโค้ง 15 px
    position:absolute;
    top:99px;                        !เลื่อนลงจากด้านบนของหน้าเว็บ 99 px
    left:25%;                         !เลื่อนลงจากด้านซ้ายของหน้าเว็บ 25%
    width:240px;                     !กำหนดให้กรอบกว้าง 240 px
    margin:0 auto;}
.Living{
    height:260px;                    !ให้กรอบมีความสูง 260 px
    padding-left:15px;               !เลื่อนไปด้านซ้าย 15 px
    background:#34495E;              !กำหนดสีของพื้นหลัง
    font color:#ECF0F1;              !กำหนดสีของตัวอักษร
    font-family:Calibri; }           !กำหนดรูปแบบของตัวอักษร
!-----รูปแบบการตกแต่งส่วนของห้องนอน -----!
.Bedroom{
    border:4px groove gray;          !สร้างกรอบขนาด 4 px สีเทา
    border-radius:15px;              !มุมรอบโค้ง 15 px

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

position:relative;
top:0px;                !เลื่อนลงจากด้านบนของหน้าเว็บ 0 px
left:15%;                !เลื่อนลงจากด้านซ้ายของหน้าเว็บ 15%
width:240px;             !กำหนดให้กรอบกว้าง 240 px
margin:0 auto;
}

.Bedroom{
height:260px;            !ให้กรอบมีความสูง 260 px
padding-left:15px;       !เลื่อนไปด้านซ้าย 15 px
background:#34495E;      !กำหนดสีของพื้นหลัง
font color:#24437C;      !กำหนดสีของตัวอักษร
font-family:Calibri; }   !กำหนดรูปแบบของตัวอักษร
!----- รูปแบบการตกแต่งส่วนของห้องอาบน้ำ -----!

.Bathroom{
border:4px inset gray;    !สร้างกรอบขนาด 4 px สีเทา
border-radius:15px;       !มุมกรอบโค้ง 15 px
position:absolute;
top:400px;                !เลื่อนลงจากด้านบนของหน้าเว็บ 400 px
left:25%;                 !เลื่อนลงจากด้านซ้ายของหน้าเว็บ 25%
width:240px;              !กำหนดให้กรอบกว้าง 240 px
margin:0 auto;
}

.Bathroom{
height:260px;            !ให้กรอบมีความสูง 260 px
padding-left:15px;       !เลื่อนไปด้านซ้าย 15 px
background:#34495E;      !กำหนดสีของพื้นหลัง
font color:#ECF0F1;      !กำหนดสีของตัวอักษร
font-family:Calibri; }   !กำหนดรูปแบบของตัวอักษร
!----- รูปแบบการตกแต่งส่วนของห้องครัว -----!

.Kitchen{
border:4px ridge gray;    !สร้างกรอบขนาด 4 px สีเทา
border-radius:15px;       !มุมกรอบโค้ง 15 px
position:relative;

```

```

top:30px;                !เลื่อนลงจากด้านบนของหน้าเว็บ 30 px
left:25%;                !เลื่อนลงจากด้านซ้ายของหน้าเว็บ 25%
width:240px;             !กำหนดให้กรอบกว้าง 240 px
margin:0 auto;}

.Kitchen{
  height:260px;          !ให้กรอบมีความสูง 260 px
  padding-left:15px;      !เลื่อนไปด้านซ้าย 15 px
  background:#34495E;     !กำหนดสีของพื้นหลัง
  font color:#ECF0F1;     !กำหนดสีของตัวอักษร
  font-family:Calibri; }  !กำหนดรูปแบบของตัวอักษร

a:link, a:visited {
  background-color: #120377; !กำหนดสีของพื้นหลังปุ่มกด
  border:4px outset gray;    !สร้างกรอบขนาด 4 px สีเทา
  border-radius:20px;        !มุมกรอบโค้ง 20 px
  color: white;              !กำหนดให้เป็นสีขาว
  width:100px;              !กำหนดให้กรอบกว้าง 240 px
  height:25px;              !กำหนดให้กรอบสูง 25 px
  text-align: center;        !ให้ตัวอักษรอยู่กึ่งกลาง
  text-decoration: none;
  display: inline-block;}

a:hover, a:active {background-color: #2ECC71;} !กำหนดสีขณะกดปุ่ม
</style></head><body>
!----- ชื่อหัวข้อโดยกำหนดให้อยู่ตรงกลาง -----!
<h1><p align="center">Controlling Electronic Devices Using GUI</h1></p>
<font color="#ECF0F1">                !กำหนดสีตัวอักษร
<div class="Living">                  !ตกแต่งภายในกรอบของห้องนั่งเล่น
  <h2>LIVING ROOM</h2>                !หัวข้อที่ถูกกำหนดให้มีขนาดใหญ่
  <font size="4">                      !กำหนดขนาดตัวอักษรเนื้อหา
  <li>Light is currently state <b>{{LivL}}</b><p>&emsp;&emsp;&emsp;
  {% if LivL == 'OFF' %}                !ถ้า LivL มีค่าเป็น OFF
  <a href="/LivL/on">TURN ON</a>      !สั่งให้อุปกรณ์ทำงาน และปุ่มแสดง TURN ON

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

{% else %}                                !ถ้า LivL มีค่าไม่เป็น OFF
<a href="/LivL/off" class="button">TURN OFF</a> !สั่งให้ปิดไฟและปุ่มแสดง TURN OFF
{% endif %}</p></li></font> !จบสั่งงานอุปกรณ์ไฟฟ้า
</div> <!div class="Living">              !จบการตกแต่งภายในกรอบของห้องนั่งเล่น
<div class="Bedroom">                    !ตกแต่งภายในกรอบของห้องนั่งเล่น
  <h2>BEDROOM</h2>                      !หัวข้อที่ถูกกำหนดให้มีขนาดใหญ่
  <font size="4">                        !กำหนดขนาดตัวอักษรเนื้อหา
    <li>Light is currently state <b>{{BedL}}</b><p>&emsp;&emsp;&emsp;
    {% if BedL == 'OFF' %}                !ถ้า BedL มีค่าเป็น OFF
      <a href="/BedL/on">TURN ON</a> !สั่งให้อุปกรณ์ทำงาน และปุ่มแสดง TURN ON
    {% else %}                            !ถ้า BedL มีค่าไม่เป็น OFF
      <a href="/BedL/off" class="button">TURN OFF</a>!สั่งให้ปิดหลอดไฟและปุ่มแสดง TURN OFF
    {% endif %}</p></li>                  !จบสั่งงานหลอดไฟ
    <li>Fan is currently state <b>{{BedF}}</b><p>&emsp;&emsp;&emsp;
    {% if BedF == 'OFF' %}                !ถ้า BedF มีค่าเป็น OFF
      <a href="/BedF/on">TURN ON</a> !สั่งให้อุปกรณ์ทำงาน และปุ่มแสดง TURN ON
    {% else %}                            !ถ้า BedF มีค่าไม่เป็น OFF
      <a href="/BedF/off" class="button">TURN OFF</a>!สั่งให้ปิดพัดลมและปุ่มแสดง TURN OFF
    {% endif %}</p></li></font> !จบสั่งงานหลอดไฟ
  </div> <!div class="Bedroom">          !จบการตกแต่งภายในกรอบของห้องนอน
  <div class="Bathroom">                !ตกแต่งภายในกรอบของห้องอาบน้ำ
    <h2>BATHROOM</h2>                  !หัวข้อที่ถูกกำหนดให้มีขนาดใหญ่
    <font size="4">                    !กำหนดขนาดตัวอักษรเนื้อหา
      <li>Light is currently state <b>{{BathL}}</b><p>&emsp;&emsp;&emsp;
      {% if BathL == 'OFF' %}            !ถ้า BathL มีค่าเป็น OFF
        <a href="/BathL/on">TURN ON</a> !สั่งให้อุปกรณ์ทำงาน และปุ่มแสดง TURN ON
      {% else %}                          !ถ้า BedF มีค่าไม่เป็น OFF
        <a href="/BathL/off" class="button">TURN OFF</a>!สั่งให้ปิดหลอดไฟและแสดง TURN OFF
      {% endif %}</p></li></font> !จบสั่งงานหลอดไฟ

```

เอกสารนี้เป็นเอกสารที่สงวนไว้ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

|                                                                   |                                          |
|-------------------------------------------------------------------|------------------------------------------|
| </div> <!div class="Bathroom">                                    | !จบการตกแต่งภายในกรอบของห้องอาบน้ำ       |
| <div class="Kitchen">                                             | !ตกแต่งภายในกรอบของห้องครัว              |
| <h2>KITCHEN ROOM</h2>                                             | !หัวข้อที่ถูกกำหนดให้มีขนาดใหญ่          |
| <font size="4">                                                   | !กำหนดขนาดตัวอักษรเนื้อหา                |
| <li>Light is currently state <b>{{KitL}}</b><p>&emsp;&emsp;&emsp; |                                          |
| {% if KitL == 'OFF' %}                                            | !ถ้า KitL มีค่าเป็น OFF                  |
| <a href="/KitL/on">TURN ON</a>                                    | !สั่งให้อุปกรณ์ทำงาน และปุ่มแสดง TURN ON |
| {% else %}                                                        | !ถ้า KitL มีค่าไม่เป็น OFF               |
| <a href="/KitL/off" class="button">TURN OFF</a>                   | !สั่งให้ปิดหลอดไฟและแสดง TURN OFF        |
| {% endif %}</p></li>                                              | !จบสั่งงานหลอดไฟ                         |
| <li>Fan is currently state <b>{{KitF}}</b><p>&emsp;&emsp;&emsp;   |                                          |
| {% if KitF == 'OFF' %}                                            | !ถ้า KitF มีค่าเป็น OFF                  |
| <a href="/KitF/on">TURN ON</a>                                    | !สั่งให้อุปกรณ์ทำงาน และปุ่มแสดง TURN ON |
| {% else %}                                                        | !ถ้า KitF มีค่าไม่เป็น OFF               |
| <a href="/KitF/off" class="button">TURN OFF</a>                   | !สั่งให้ปิดพัดลมและปุ่มแสดง TURN OFF     |
| {% endif %}</p></li></font>                                       | !จบสั่งงานพัดลม                          |
| </div> <!div class="Kitchen"></font>                              | !จบการตกแต่งภายในกรอบของห้องครัว         |
| </form></body></html>                                             | !จบการเขียนหน้าเว็บเพจ                   |