



รายงาน

เรื่อง เครื่องให้อาหารสัตว์ Happy Pet ด้วยระบบ IoT

จัดทำโดย

นางสาว รัชฎาภรณ์ วุฒิรุ่งเรืองสกุล รหัสนักศึกษา 6687063

นางสาว เอมิสรา อนธรรมงคล รหัสนักศึกษา 6687074

นักศึกษาชั้นปีที่ 2 คณะเทคโนโลยีสารสนเทศและการสื่อสาร

เสนอ

ผู้ช่วยศาสตราจารย์ ดร. วิจิตันนท์ ตันติธรรม

รายงานฉบับนี้เป็นส่วนหนึ่งในรายวิชา ITDS282 Internet of Thing (IoT)

ภาคเรียนที่ 2 ปีการศึกษา 2567

มหาวิทยาลัยมหิดล

คำนำ

รายงานฉบับนี้เป็นส่วนหนึ่งของวิชา ITDS282 Internet of Things (IoT) โดยมีวัตถุประสงค์เพื่อศึกษาและพัฒนาอุปกรณ์สมาร์ทโฮม (Smart Home Device) ที่สามารถนำเทคโนโลยี IoT มาประยุกต์ใช้ในชีวิตประจำวันได้อย่างมีประสิทธิภาพ ขณะผู้จัดทำได้เลือกพัฒนา เครื่องให้อาหารสัตว์อัตโนมัติ ซึ่งออกแบบมาเพื่อช่วยเจ้าของสัตว์เลี้ยงในการดูแลให้อาหารได้อย่างสะดวกสบายและตรงเวลา แม้ในช่วงที่ไม่อยู่บ้าน

โครงการนี้ใช้เซนเซอร์ตรวจจับความเคลื่อนไหว (PIR Sensor) เพื่อรับรู้การเข้ามาใกล้ของสัตว์เลี้ยง และใช้มอเตอร์เซอร์โว (Servo Motor) เป็นกลไกในการเปิด-ปิดช่องจ่ายอาหาร โดยสามารถตรวจสอบและสั่งการผ่านแอปพลิเคชันหรือแพลตฟอร์มออนไลน์ได้

การดำเนินโครงการครั้งนี้ทำให้คณะผู้จัดทำได้เรียนรู้กระบวนการพัฒนาอุปกรณ์ IoT ตั้งแต่การออกแบบระบบ การเลือกใช้อุปกรณ์ การเขียนโปรแกรมควบคุม ไปจนถึงการทดสอบและแก้ไขปัญหาที่เกิดขึ้น ซึ่งเป็นประสบการณ์สำคัญในการนำไปประยุกต์ใช้ในการทำงานจริงต่อไป

คณะผู้จัดทำหวังว่ารายงานฉบับนี้จะเป็นประโยชน์ต่อผู้อ่าน และขอขอบคุณอาจารย์ผู้สอนที่ให้โอกาสในการทำโครงการครั้งนี้ รวมถึงค่าแนะนำที่มีคุณค่าตลอดระยะเวลาการเรียน

ธัญญารัตน์ วุฒิรุ่งเรืองสกุล

เขมิสรา อนธรรมงคล

สารบัญ

หัวข้อ	หน้า
บทที่ 1 บทนำ	
- ที่มาและความสำคัญ	1
- วัตถุประสงค์ของโครงการ	2
- ขอบเขตของการศึกษาค้นคว้า	2
บทที่ 2 ตัวอย่างการใช้งาน scenarios	3
บทที่ 3 การทำงานของระบบ	5
บทที่ 4 อุปกรณ์และเทคโนโลยีที่เกี่ยวข้อง	
- อุปกรณ์ที่ใช้ในการพัฒนา	7
- เทคโนโลยีที่ใช้ในการพัฒนา	10
บทที่ 5 ผลการศึกษา	
- การต่อบอร์ดจริง	11
- Code สำหรับอัปโหลดโปรแกรมไปยังบอร์ด ESP32	12
- การทำงานของ Flow Node-RED	17
- การทำงานของ Blynk	23
- สรุปผลการศึกษา	25
วิดีโอนำเสนอ	26

บทที่ 1

บทนำ

1. ที่มาและความสำคัญ

ในปัจจุบัน สัตว์เลี้ยงมีบทบาทสำคัญในชีวิตมนุษย์ ทั้งในฐานะเพื่อนคลายเหงา และเป็นสมาชิกหนึ่งของครอบครัว จากสถิติของ American Pet Products Association (APPA) พบว่าในปี 2023 ชาวอเมริกันกว่า 66% ของครัวเรือนมีสัตว์เลี้ยงอย่างน้อยหนึ่งตัว (APPA, 2023) และดังให้เห็นถึงแนวโน้มที่เพิ่มขึ้นทั่วโลกในการเลี้ยงสัตว์ ทั้งนี้ การดูแลสุขภาพและพฤติกรรมของสัตว์เลี้ยงเป็นสิ่งสำคัญ โดยเฉพาะอย่างยิ่งในด้านการให้อาหารที่ต้องมีความสม่ำเสมอและเหมาะสมตามเวลา

แต่เจ้าของสัตว์เลี้ยงจำนวนมากไม่สามารถให้อาหารสัตว์เลี้ยงได้อย่างสม่ำเสมอ นำไปสู่ปัญหาทางสุขภาพ เช่น น้ำหนักตัวที่ผิดปกติ หรือปัญหาพฤติกรรมต่าง ๆ อันเนื่องมาจากความเครียดและความทิฐิ ขณะผู้จัดทำจึงทำการพัฒนาเครื่องให้อาหารสัตว์อัตโนมัติ เพื่อตอบโจทย์ความต้องการดังกล่าว โดยช่วยให้สัตว์เลี้ยงได้รับอาหารในเวลาที่เหมาะสม

โดยในโครงงานนี้ได้นำเทคโนโลยีเซ็นเซอร์ตรวจจับการเคลื่อนไหว (Motion Sensor) มาประยุกต์ใช้ โดยเซ็นเซอร์จะทำหน้าที่ตรวจจับการเคลื่อนไหวในบริเวณใกล้เคียงกับตัวเครื่อง และส่งการให้เปิดไฟสัญญาณแสดงสถานะเมื่อมีการเคลื่อนไหวใกล้ๆ ทำให้สัตว์เลี้ยงสามารถสังเกตเห็นเครื่องให้อาหารได้ง่ายขึ้น เพิ่มความสะดวกในการใช้งานโดยเฉพาะในเมืองใหญ่ ทั้งยังช่วยประหยัดพลังงานเนื่องจากไฟจะทำงานเฉพาะเมื่อมีความเคลื่อนไหว

การออกแบบระบบลักษณะนี้สอดคล้องกับแนวโน้มการใช้เทคโนโลยีอัจฉริยะในชีวิตประจำวัน (Smart Living) ที่เน้นการใช้พลังงานอย่างมีประสิทธิภาพและสร้างความสะดวกสบายสูงสุด (Smart Home and Building Association, 2023) อีกทั้งยังสอดคล้องกับหลักการของ Internet of Things (IoT) ที่มุ่งเน้นการเชื่อมต่ออุปกรณ์ต่าง ๆ เพื่อตอบสนองความต้องการของผู้ใช้งานอย่างชาญฉลาด

ด้วยเหตุนี้ การพัฒนาเครื่องให้อาหารสัตว์อัตโนมัติที่ใช้เซ็นเซอร์ตรวจจับการเคลื่อนไหวและระบบไฟแสดงสถานะ จึงมีความสำคัญอย่างยิ่งทั้งในด้านการยกระดับคุณภาพชีวิตของสัตว์เลี้ยงและการใช้เทคโนโลยีเพื่อส่งเสริมความสะดวกสบายให้แก่เจ้าของสัตว์

2. วัตถุประสงค์ของโครงงาน

- เพื่อออกแบบและพัฒนาเครื่องให้อาหารสัตว์ที่สามารถจ่ายอาหารได้แบบอัตโนมัติเมื่อมีการเคลื่อนไหว
- เพื่อประยุกต์ใช้เซ็นเซอร์ตรวจจับการเคลื่อนไหว (Motion Sensor) และการเปิดปิดไฟแสดงสถานะเมื่อมีการเคลื่อนไหวใกล้เครื่อง
- เพื่อส่งเสริมการใช้เทคโนโลยี Internet of Things (IoT) และแนวคิด Smart Living ในการดูแลสัตว์เลี้ยง

3. ขอบเขตการศึกษาค้นคว้า

1. ด้านฮาร์ดแวร์

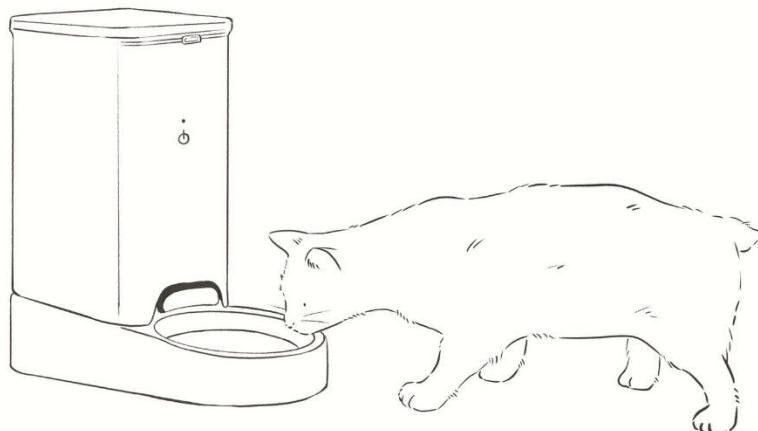
- ใช้บอร์ด ESP32 เป็นตัวควบคุมหลักของระบบ
- ใช้ PIR Sensor ในการตัวจับการเคลื่อนไหว
- ใช้ Servo ในการควบคุมการเปิด-ปิดช่องจ่ายอาหาร
- ใช้ LED ในการแสดงผลสถานะของเครื่อง

2. ด้านซอฟต์แวร์

- โปรแกรมควบคุมการทำงานของระบบบนบอร์ด ESP32 โดยใช้ภาษา C/C++ ผ่าน Arduino IDE
- ใช้อ�플ิเคชัน Blynk สำหรับควบคุมสั่งการ แสดงสถานะ และการแจ้งเตือนของเครื่อง

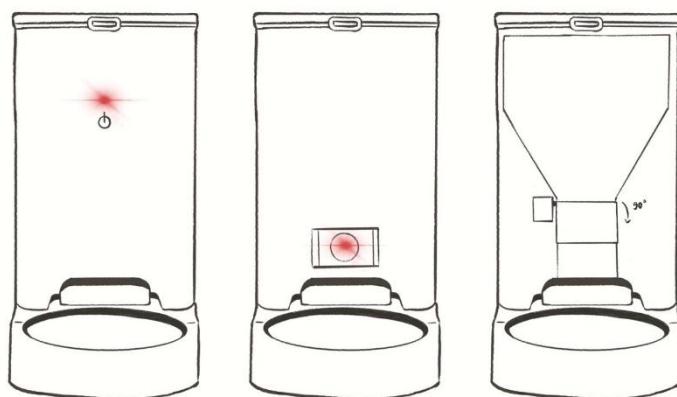
บทที่ 2

ตัวอย่างการใช้งาน Scenarios



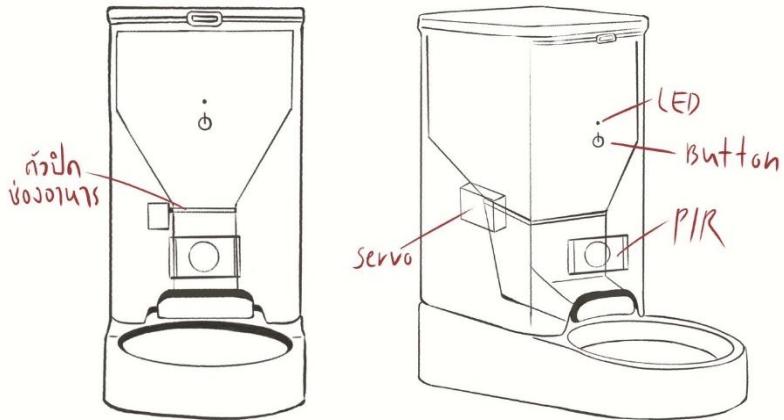
(รูปภาพที่ 1 : ภาพจำลองเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติ Happy Pet)

เครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติจะมีเซนเซอร์สำหรับตรวจสอบการเคลื่อนไหว โดยมีเงื่อนไขคือหากมีการเคลื่อนไหวใกล้ๆ เครื่อง เช่น สัตว์เลี้ยงเดินผ่านบริเวณเซนเซอร์ของตัวเครื่อง ไฟด้านขวาจะติด และเครื่องจะปล่อยอาหารออกมาเป็นเวลา 1 วินาทีและหยุด ถ้าหากไม่มีการเคลื่อนไหวบริเวณเครื่องไฟจะไม่ติดและเครื่องจะไม่ปล่อยอาหารออกมา



(รูปภาพที่ 2 : ภาพจำลองเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติ Happy Pet ขณะตรวจจับความเคลื่อนไหวได้)

นอกจากนี้ ตัวเครื่องยังมีปุ่มสำหรับเปิดและปิดระบบการทำงาน โดยทางระบบถูกปิดอยู่ ถึงแม้จะมีการเคลื่อนไหวบริเวณเซ็นเซอร์ ตัวเครื่องจะไม่ทำงานใด ๆ ทั้งสิ้น ทั้งไฟและกลไกการปล่อยอาหารจะไม่ทำงาน จนกว่าผู้ใช้จะกดปุ่มเพื่อระบบอิกซ์ริง ซึ่งช่วยให้สามารถควบคุมการทำงานของเครื่องได้ตามต้องการและเพิ่มความปลอดภัยในการใช้งาน

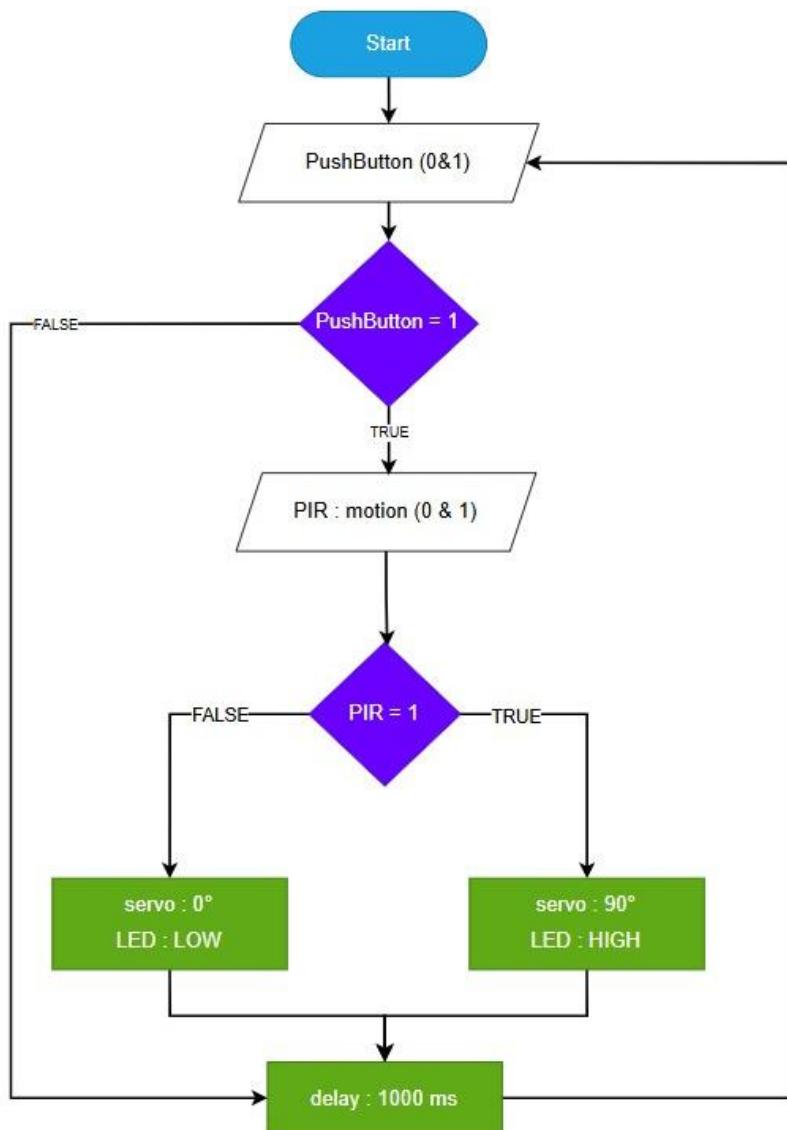


(รูปภาพที่ 3 : ภาพจำลองเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติ Happy Pet พร้อมส่วนประกอบต่างๆ)

บทที่ 3

การทำงานของระบบ

เพื่อให้เข้าใจขั้นตอนการทำงานของระบบได้อย่างชัดเจนและเป็นลำดับ คณะกรรมการฯทำจำนำเสนอภาพรวมของการทำงานของระบบผ่านแผนผังลำดับขั้น (Flowchart) ซึ่งจะแสดงกระบวนการตั้งแต่เริ่มต้นจนถึงสิ้นสุด รวมถึงการตัดสินใจและการทำงานในแต่ละส่วนของระบบอย่างเป็นระบบ ดังนี้



(รูปภาพที่ 4 : Flowchart การทำงานของระบบ)

อธิบายการทำงานของระบบ

1. รับค่าจากปุ่ม โดยแทนค่าให้หากมีการกดปุ่มเป็น 1 และหากไม่มีการกดปุ่มเป็น 0
2. หากค่าจากปุ่มเป็น 1 จะเข้าเงื่อนไข True
 - 2.1. รับค่าการขยับจาก PIR Sensor โดยแทนค่าให้หากมีการขยับเป็น 1 และหากไม่มีการขยับเป็น 0
 - 2.2. หากค่าจาก PIR Sensor เป็น 1 จะเข้าเงื่อนไข True
 - 2.2.1. Servo หมุนขน 90° และ LED ติด
 - 2.2.2. Delay 1 วินาที และกลับไปรับค่าใหม่
 - 2.3. หากค่าจาก PIR Sensor เป็น 0 จะเข้าเงื่อนไข False
 - 2.3.1. Servo ไม่หมุนขนและ LED ไม่ติด
 - 2.3.2. Delay 1 วินาที และกลับไปรับค่าใหม่
3. หากค่าจากปุ่มเป็น 0 จะเข้าเงื่อนไข False
 - 3.1. Delay 1 วินาที และกลับไปรับค่าใหม่

บทที่ 4

อุปกรณ์และเทคโนโลยีที่เกี่ยวข้อง

เพื่อให้เข้าใจถึงหน้าที่และความสมมั่นใจของอุปกรณ์ที่ใช้ในการสนับสนุนการทำงานของระบบอย่างมีประสิทธิภาพ คณะกรรมการจัดทำจึงจะนำเสนออุปกรณ์ที่มีความสำคัญและเทคโนโลยีที่ถูกนำมาใช้ในการพัฒนาระบบทั้งหมด โดยจะอธิบายถึงลักษณะการทำงาน และบทบาทของแต่ละอุปกรณ์ภายในระบบ ดังนี้

1. อุปกรณ์ที่ใช้ในการพัฒนา

1.1. บอร์ด ESP32

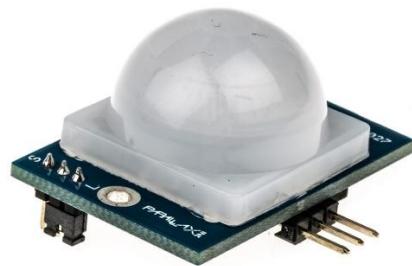
เป็นหน่วยประมวลผลหลักของระบบ ทำหน้าที่ควบคุมการทำงานของอุปกรณ์ทุกชิ้น เช่น อ่านค่าจากเซ็นเซอร์ตรวจจับการเคลื่อนไหว ตรวจสอบสถานะของปุ่ม และสั่งให้ LED และเซอร์วิโมเตอร์ทำงานตามเงื่อนไขที่กำหนด



(รูปภาพที่ 5 : บอร์ด ESP32)

1.2. PIR Sensor

ใช้สำหรับตรวจจับการเคลื่อนไหวของสัตว์เลี้ยงบริเวณหน้าเครื่อง หากมีการเคลื่อนไหว เช่น สัตว์เลี้ยงเดินผ่าน เซนเซอร์จะส่งสัญญาณไปยัง ESP32 เพื่อสั่งให้เครื่องให้อาหารทำงาน



(รูปภาพที่ 6 : PIR Sensor)

1.3. Servo Motor

ใช้ควบคุมกลไกการเปิด–ปิดช่องปล่อยอาหาร เมื่อมีการเคลื่อนไหวและระบบเปิดอยู่ ตัวเซอร์โวจะหมุนเพื่อเปิดช่องให้อาหารไหลออกมานานเวลา 2 วินาที และหมุนกลับเพื่อปิดช่อง



(รูปภาพที่ 7 : Servo Motor)

1.4. หลอดไฟ LED

ใช้แสดงสถานะว่ามีการตรวจพบการเคลื่อนไหวหรือไม่ โดยเมื่อเซ็นเซอร์ตรวจพบการเคลื่อนไหว ไฟ LED จะติดขึ้น เพื่อแสดงว่ากำลังมีการทำงานของระบบให้อาหาร



(รูปภาพที่ 8 : หลอดไฟ LED)

1.5. Button

เป็นปุ่มกดสำหรับเปิดหรือปิดระบบโดยผู้ใช้งาน หากผู้ใช้กดปุ่มเพื่อปิดระบบ ตัวเครื่องจะไม่ตอบสนองต่อการเคลื่อนไหวของสัตว์เลี้ยง และจะไม่ปล่อยอาหารออคมา จนกว่าจะเปิดระบบอีกครั้ง



(รูปภาพที่ 9 : หลอดไฟ LED)

1.6. Resistor

ใช้ร่วมกับวงจรของปุ่มเพื่อช่วยป้องกันสัญญาณรบกวนหรือ false trigger โดยเฉพาะเมื่อต่อปุ่มกับขา Input ของ ESP32 ที่ใช้การตั้งค่าเป็น INPUT_PULLUP ทำให้ปุ่มทำงานได้อย่างเสถียร



(รูปภาพที่ 10 : Resistor)

2. เทคโนโลยีที่ใช้ในการพัฒนา

2.1. โปรแกรม Arduino ide

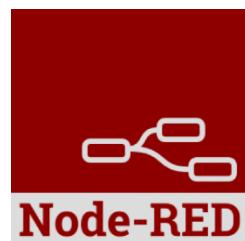
เป็นโปรแกรมสำหรับเขียนโค้ดและอปโหลดโปรแกรมไปยังบอร์ด ESP32 โดยใช้ภาษา C/C++ ผ่านไลบรารีต่าง ๆ ที่รองรับการควบคุมเซ็นเซอร์, มอเตอร์, ไฟ LED และอุปกรณ์อื่น ๆ ในระบบ ช่วยให้สามารถพัฒนาและทดสอบฟังก์ชันของเครื่องให้อาหารสัตว์เลี้ยงได้อย่างสะดวก



(รูปภาพที่ 11 : โปรแกรม Arduino ide)

2.2. Node-RED

เป็นเครื่องมือพัฒนาแอปพลิเคชันแบบ visual programming ที่ทำงานผ่าน web browser โดยสามารถใช้สร้าง Dashboard สำหรับแสดงผลและควบคุมระบบจากระยะไกลผ่านอินเทอร์เน็ต เช่น แสดงสถานะการเคลื่อนไหวหรือควบคุมการเปิด-ปิดระบบให้อาหาร



(รูปภาพที่ 12 : Node-RED)

2.3. Blynk

เป็นแพลตฟอร์ม IoT ที่ช่วยให้สามารถควบคุมและมอนิเตอร์อุปกรณ์จากมือถือได้อย่างง่ายดาย โดยในโครงการนี้สามารถนำมาใช้สร้างอินเทอร์เฟซบนสมาร์ทโฟนเพื่อเปิด-ปิดระบบให้อาหาร หรือดูสถานะของเซ็นเซอร์และการทำงานของอุปกรณ์ต่าง ๆ ได้แบบเรียลไทม์



(รูปภาพที่ 13 : Blynk)

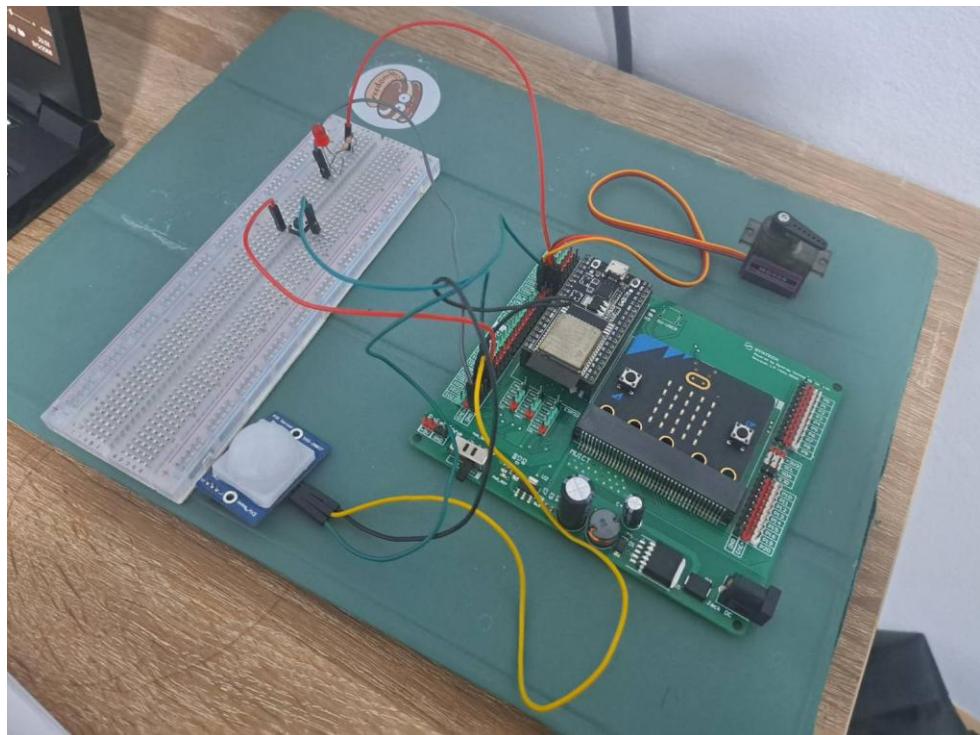
บทที่ 5

ผลการศึกษา

จากการศึกษาและพัฒนาเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติ โดยใช้เซนเซอร์ตรวจจับความเคลื่อนไหว (PIR Sensor) เพื่อรับรู้การเข้ามาใกล้ของสัตว์เลี้ยง และใช้มอเตอร์เซอร์โว (Servo Motor) เป็นกลไกในการเปิด-ปิดช่องจ่ายอาหาร ขณะผู้จัดทำขอนำเสนอผลการศึกษา ดังนี้

1. การต่อบอร์ดจริง

ในการต่อบอร์ด ESP32 กับอุปกรณ์อื่น ขณะผู้จัดทำใช้การต่อตระหง่านแต่ละอุปกรณ์กับบอร์ด โดยใช้ต่อ PIR Sensor, Servo Motor, LED, Button ที่ GPIO 35, 32, 33 และ 4 ตามลำดับ และมีการใช้ตัวต้านทานต่อกับ LED ตั้งภาพและ [วิดีโอการใช้งาน](#)



(รูปภาพที่ 14 : การต่อบอร์ด ESP32 กับอุปกรณ์ต่างๆ)

2. Code สำหรับอัปโหลดโปรแกรมไปยังบอร์ด ESP32

โปรแกรมนี้เขียนขึ้นเพื่อควบคุมการทำงานของเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติ โดยใช้บอร์ด ESP32 เป็นตัวควบคุมหลัก ร่วมกับเซนเซอร์ตรวจจับการเคลื่อนไหว (PIR Sensor), เซอร์โวมอเตอร์ (Servo Motor), หลอดไฟแสดงสถานะ (LED) และปุ่มสำหรับเปิด-ปิดระบบ โดยโปรแกรมยังรองรับการเชื่อมต่อกับระบบ IoT ผ่านโปรโตคอล MQTT เพื่อให้สามารถควบคุมหรืออ่อนนิเตอร์สถานะของเครื่องจากระยะไกลได้

Code
#include <WiFi.h> #include <PubSubClient.h> #include <ESP32Servo.h> // === CONFIG ===== const char* ssid = "ชื่อ WiFi"; const char* password = "รหัส WiFi"; const char* mqtt_server = "IP เครื่อง"; const int mqtt_port = 1883; const char* mqtt_topic = "esp32/motion"; const char* mqtt_servo_topic = "esp32/servo"; const char* mqtt_control_topic = "esp32/control"; // ===== const int PIR_PIN = 35; const int LED_PIN = 33; const int SERVO_PIN = 32; const int BUTTON_PIN = 4; // ปุ่มจริงที่ต่อ GPIO 4 WiFiClient espClient; PubSubClient client(espClient); Servo myServo; bool enabled = false; bool lastButtonState = HIGH;

```

unsigned long lastDebounceTime = 0;
const unsigned long debounceDelay = 50;
void callback(char* topic, byte* payload, unsigned int length) {
    String message;
    for (unsigned int i = 0; i < length; i++) {
        message += (char)payload[i];
    }

    if (String(topic) == mqtt_control_topic) {
        if (message == "1") {
            enabled = true;
            Serial.println("  ระบบเปิดจาก MQTT");
        } else {
            enabled = false;
            Serial.println("  ระบบปิดจาก MQTT");
            myServo.write(0);
            digitalWrite(LED_PIN, LOW);
        }
    }
}

void setup_wifi() {
    delay(10);
    Serial.println("Connecting to WiFi... ");
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("\nWiFi connected. IP address: ");
    Serial.println(WiFi.localIP());
}

```

```

}

void reconnect() {
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        String clientId = "ESP32Client-" + String(random(0xffff), HEX);
        if (client.connect(clientId.c_str())) {
            Serial.println("connected");
            client.subscribe(mqtt_control_topic);
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            delay(5000);
        }
    }
}

void setup() {
    Serial.begin(115200);
    pinMode(PIR_PIN, INPUT);
    pinMode(LED_PIN, OUTPUT);
    pinMode(BUTTON_PIN, INPUT_PULLUP);

    myServo.setPeriodHertz(50);
    myServo.attach(SERVO_PIN, 500, 2500);
    myServo.write(0);
    digitalWrite(LED_PIN, LOW);

    setup_wifi();
    client.setServer(mqtt_server, mqtt_port);
    client.setCallback(callback);
    client.subscribe(mqtt_control_topic);
}

```

```

    }

void loop() {
    if (!client.connected()) {
        reconnect();
    }
    client.loop();

    // O บุ่ม toggle เปิด/ปิดระบบ
    int reading = digitalRead(BUTTON_PIN);
    if ((reading != lastButtonState) && (millis() - lastDebounceTime > debounceDelay)) {
        lastDebounceTime = millis();
        if (reading == LOW) {
            enabled = !enabled;
            Serial.print(" O บุ่มถูกกด → ");
            Serial.println(enabled ? "เปิดระบบ" : "ปิดระบบ");
            if (!enabled) {
                // ระบบปิด → ปิด LED/Servo และไม่ต้องสนใจ PIR
                myServo.write(0);
                digitalWrite(LED_PIN, LOW);
                Serial.println(" ⚡ ระบบปิด → IGNORE PIR");
            }
        }
    }
    lastButtonState = reading;
    // ตรวจจับ PIR เสมอ แต่จะใช้ผลเฉพาะตอน enabled เท่านั้น
    int pirState = digitalRead(PIR_PIN);
    if (enabled) {
        if (pirState == HIGH) {
            Serial.println(" 🚶 ตรวจพบการเคลื่อนไหว → Servo = 90");
            myServo.write(90);
            digitalWrite(LED_PIN, HIGH);
        }
    }
}

```

```

client.publish(mqtt_topic, "1");
client.publish(mqtt_servo_topic, "90");
} else {
    Serial.println("🔴 ไม่มีการเคลื่อนไหว → Servo = 0");
    myServo.write(0);
    digitalWrite(LED_PIN, LOW);
    client.publish(mqtt_topic, "0");
    client.publish(mqtt_servo_topic, "0");
}
} else {
    // ระบบปิด: ไม่ควรทำอะไรเมื่อ PIR จะแจ้ง HIGH
    // (ถ้ามีสิ่งใดควบคุมเพิ่มเมื่อปิดก็ใส่ตรงนี้)
}
delay(1000);
}

```

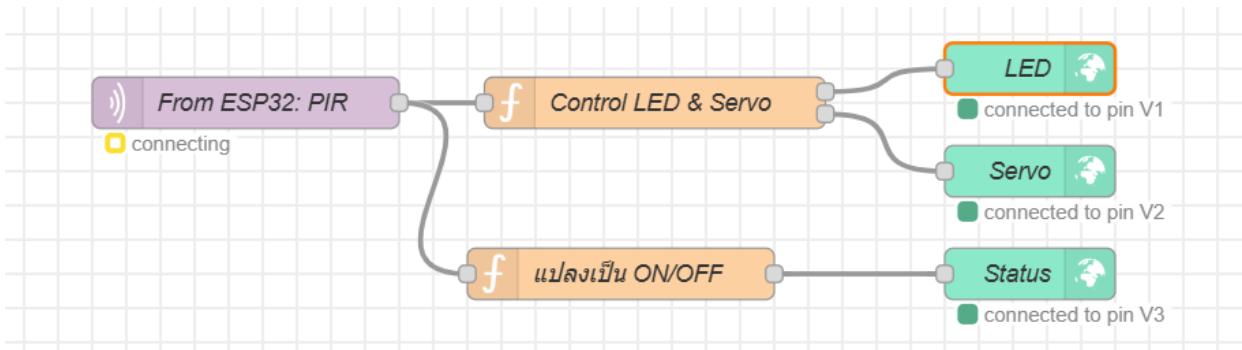
Description

ในขั้นตอนแรกของโปรแกรม จะมีการเชื่อมต่อ WiFi ด้วยชื่อและรหัสผ่านที่กำหนดไว้ จากนั้นจึงเชื่อมต่อกับ MQTT Broker โดยใช้ IP และพอร์ตที่ระบุ เมื่อเชื่อมต่อสำเร็จ โปรแกรมจะสมัคร (subscribe) กับหัวข้อ esp32/control เพื่อรับคำสั่งเปิดหรือปิดระบบจากภายนอก เช่น Node-RED หรือแอป Blynk เมื่อได้รับข้อมูลความจาก MQTT หากข้อมูลเป็น “1” ระบบจะถูกเปิดใช้งาน และหากเป็น “0” ระบบจะปิดการทำงานทันที รวมถึงสั่งให้เซอร์โวมอเตอร์กลับตำแหน่งเดิม (ปิดช่องอาหาร) และดับไฟ LED

ในส่วนของฟังก์ชันหลัก loop() โปรแกรมจะทำงานอย่างต่อเนื่องเพื่อตรวจสอบสถานะของปุ่มที่ผู้ใช้กดเพื่อเปิดหรือปิดระบบ หากผู้ใช้กดปุ่ม ระบบจะทำการเปลี่ยนสถานะจากเปิดเป็นปิดหรือจากปิดเป็นเปิด โดยใช้เทคนิคการหน่วงเวลา (debounce) เพื่อหลีกเลี่ยงการอ่านค่าซ้ำจากการกดปุ่มเพียงครั้งเดียว หากระบบถูกปิดอยู่ โปรแกรมจะไม่ตอบสนองต่อสัญญาณจากเซนเซอร์ตรวจจับความเคลื่อนไหว

เมื่อระบบเปิดอยู่ โปรแกรมจะตรวจสอบค่าอินพุตจากเซนเซอร์ PIR ว่ามีการเคลื่อนไหวเกิดขึ้นบริเวณใกล้ตัวเครื่องหรือไม่ หากตรวจพบการเคลื่อนไหว โปรแกรมจะสั่งให้เซอร์โวมอเตอร์หมุนไปยังตำแหน่ง 90 องศา ซึ่งหมายถึงการเปิดช่องปล่อยอาหาร พร้อมกับเปิดไฟ LED เพื่อแสดงสถานะการทำงาน และส่งข้อมูลผ่าน MQTT เพื่อแจ้งให้ระบบภายนอกทราบว่ามีการเคลื่อนไหวเกิดขึ้นและเครื่องกำลังทำงาน หากไม่ตรวจพบการเคลื่อนไหว ระบบจะสั่งให้เซอร์โวมอเตอร์หมุนกลับที่ตำแหน่ง 0 องศา และปิดไฟ LED

3. การทำงานของ Flow Node-RED



(รูปภาพที่ 15 : Flow Node-RED)

Flow.json

```
[  
 {  
   "id": "8aee7dd8cd533d18",  
   "type": "tab",  
   "label": "Flow 3",  
   "disabled": false,  
   "info": ""  
,  
 {  
   "id": "5b2fdfac69426b60",  
   "type": "mqtt in",  
   "z": "8aee7dd8cd533d18",  
   "name": "From ESP32: PIR",  
   "topic": "esp32/motion",  
   "qos": "2",  
   "datatype": "auto-detect",  
   "broker": "mqtt1",  
   "nl": false,  
   "rap": false,  
   "inputs": 0,
```

```

    "x": 150,
    "y": 240,
    "wires": [
        [
            "2e94f117d59b98cf",
            "e32c36f86956ec02"
        ]
    ],
    "id": "2e94f117d59b98cf",
    "type": "function",
    "z": "8aee7dd8cd533d18",
    "name": "Control LED & Servo",
    "func": "let pir = parseInt(msg.payload);\n// สำหรับ Widget ที่รับค่าตัวเลข เช่น LED หรือ\nServo\nlet led = { payload: pir, pin: 1 };           // V1\nlet servo = { payload: pir === 1 ? 90 : 0, pin: 2 }; // V2\n\nreturn [led, servo];\n",
    "outputs": 2,
    "timeout": "",
    "noerr": 0,
    "initialize": "",
    "finalize": "",
    "libs": [],
    "x": 390,
    "y": 240,
    "wires": [
        [
            "63db134f8a262116"
        ],
        [
    
```

```
        "1d64b95eb29cf14d"
    ],
],
},
{
  "id": "63db134f8a262116",
  "type": "blynk-iot-out-write",
  "z": "8aee7dd8cd533d18",
  "name": "LED",
  "pin": "1",
  "pinmode": 0,
  "client": "d3e8d91f4d85366f",
  "x": 610,
  "y": 220,
  "wires": []
},
{
  "id": "1d64b95eb29cf14d",
  "type": "blynk-iot-out-write",
  "z": "8aee7dd8cd533d18",
  "name": "Servo",
  "pin": "2",
  "pinmode": 0,
  "client": "d3e8d91f4d85366f",
  "x": 610,
  "y": 280,
  "wires": []
},
{
  "id": "8865b048869e4aa5",
```

```

    "type": "blynk-iot-out-write",
    "z": "8aee7dd8cd533d18",
    "name": "Status",
    "pin": "3",
    "pinmode": 0,
    "client": "d3e8d91f4d85366f",
    "x": 610,
    "y": 340,
    "wires": []
},
{
    "id": "e32c36f86956ec02",
    "type": "function",
    "z": "8aee7dd8cd533d18",
    "name": "ແປລງເປັນ ON/OFF",
    "func": "let pir = msg.payload.toString().trim();\nlet label = pir === \"1\" ? \"ON\" : \"OFF\";\n\nreturn {\n    payload: label,\n    pin: 3\n};",
    "outputs": 1,
    "timeout": "",
    "noerr": 0,
    "initialize": "",
    "finalize": "",
    "libs": [],
    "x": 370,
    "y": 340,
    "wires": [
        [
            "8865b048869e4aa5"
        ]
    ]
}

```

```
},
{
  "id": "mqtt1",
  "type": "mqtt-broker",
  "name": "Local MQTT",
  "broker": "192.168.26.8",
  "port": "1883",
  "clientid": "",
  "autoConnect": true,
  "usetls": false,
  "protocolVersion": 4,
  "keepalive": 15,
  "cleansession": true,
  "autoUnsubscribe": true,
  "birthTopic": "",
  "birthQos": "0",
  "birthPayload": "",
  "birthMsg": {},
  "closeTopic": "",
  "closePayload": "",
  "closeMsg": {},
  "willTopic": "",
  "willQos": "0",
  "willPayload": "",
  "willMsg": {},
  "userProps": "",
  "sessionExpiry": ""
},
{
  "id": "d3e8d91f4d85366f",
```

```

    "type": "blynk-iot-client",
    "name": "Blynk",
    "path": "blynk.cloud",
    "key": "VV303OaYc_uRpymyfk80VzorRu9xjDrJ",
    "tmpl": "TMPL6XuaZNchM",
    "dbg_all": false,
    "dbg_log": false,
    "dbg_prop": false,
    "dbg_sync": false,
    "dbg_low": false,
    "dbg_pins": "",
    "multi_cmd": false,
    "enabled": false
}
]

```

ในการพัฒนาเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติ ได้มีการนำเทคโนโลยี Node-RED มาใช้เพื่อควบคุมและแสดงผลข้อมูลจากอุปกรณ์ที่เชื่อมต่อกับบอร์ด ESP32 โดยระบบ Node-RED ที่ใช้นี้ทำหน้าที่เป็นตัวกลางในการรับข้อมูลจาก ESP32 ผ่านโปรโตคอล MQTT และแสดงสถานะต่าง ๆ ผ่านแอป Blynk บนโทรศัพท์มือถือ ใน Flow ที่สร้างขึ้น ประกอบด้วย Node หลัก ๆ ได้แก่ mqtt in, function, และ blynk-iot-out-write โดยเริ่มจาก Node แรกคือ mqtt in ที่ใช้รับข้อมูลจาก ESP32 ซึ่งถูกส่งมาทางหัวข้อ (topic) ชื่อ esp32/motion เมื่อมีการเคลื่อนไหวของสัตว์เลี้ยงเกิดขึ้นบริเวณเซ็นเซอร์ PIR ที่เชื่อมต่อกับ ESP32 ค่าที่ส่งมาจะเป็น "1" แต่หากไม่มีการเคลื่อนไหว ค่าจะเป็น "0" Node นี้จะส่งข้อมูลไปยัง Node ถัดไปคือ function ที่ชื่อว่า "Control LED & Servo"

ใน Node "Control LED & Servo" จะมีการแปลงค่าที่ได้จากเซ็นเซอร์เพื่อใช้กับ Widget บนแอป Blynk โดยค่าที่ได้จะถูกแยกออกเป็นสองชุดข้อมูล ชุดแรกใช้ควบคุมหลอดไฟ LED (Pin V1) เพื่อแสดงสถานะว่ามีการเคลื่อนไหวหรือไม่ และชุดที่สองใช้ควบคุมเซอร์โวมอเตอร์ (Pin V2) โดยกำหนดให้เมื่อมีการเคลื่อนไหว เชอร์โวจะหมุนไปที่ตำแหน่ง 90 องศา และเมื่อไม่มีการเคลื่อนไหว จะหมุนกลับที่ตำแหน่ง 0 องศา

นอกจากนี้ ข้อมูลจาก mqtt in ยังถูกส่งไปยัง Node function อีกด้วยหนึ่งชื่อว่า "แปลงเป็น ON/OFF" ซึ่งทำหน้าที่แปลงค่าที่ได้เป็นข้อความ "ON" หรือ "OFF" เพื่อแสดงบน Widget สถานะบน Blynk (Pin V3) ให้ผู้ใช้งานทราบว่าสถานะปัจจุบันของเซนเซอร์เป็นอย่างไร

ผลลัพธ์ทั้งหมดจาก Node function จะถูกส่งไปยัง Node blynk-iot-out-write เพื่อส่งข้อมูลขึ้นไปแสดงผลบนแอป Blynk ตาม Pin ที่กำหนดไว้ ได้แก่ V1 สำหรับ LED, V2 สำหรับ Servo และ V3 สำหรับข้อความสถานะ โดยระบบนี้ช่วยให้ผู้ใช้งานสามารถเห็นสถานะของเครื่องให้อาหารแบบเรียลไทม์ผ่านโทรศัพท์มือถือได้อย่างสะดวก และยังสามารถนำไปใช้ร่วมกับระบบอัตโนมัติอื่น ๆ ได้อีกด้วย

4. การทำงานของ Blynk

ระบบ Blynk ที่ใช้ในโครงงานเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติ มีบทบาทในการแสดงผลสถานะและการทำงานของอุปกรณ์ที่เชื่อมต่อกับบอร์ด ESP32 ผ่านอินเทอร์เน็ต

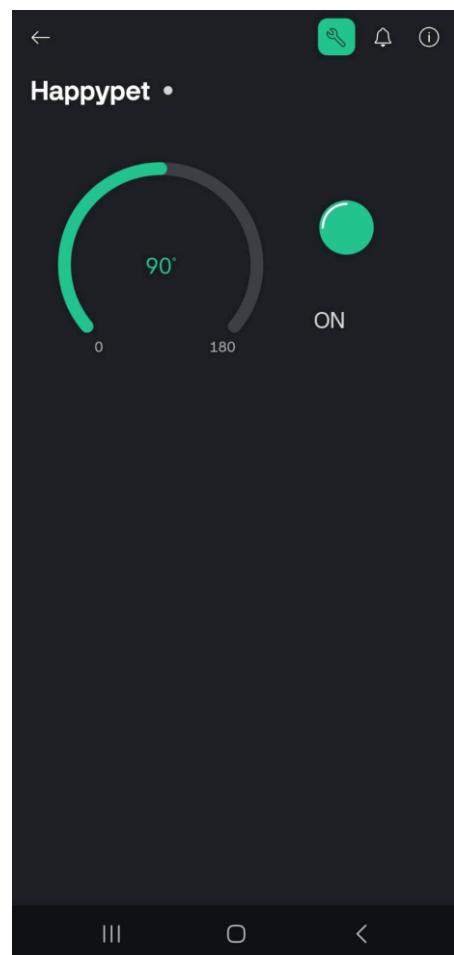
ในโครงการนี้ ระบบ Blynk ได้รับข้อมูลจาก ESP32 ผ่าน Node-RED ซึ่งทำหน้าที่เป็นตัวกลางในการประมวลผลและส่งข้อมูลเข้าแอป Blynk โดยมีการใช้งาน Virtual Pin (VPin) 3 ช่อง ได้แก่:

- V1 ใช้สำหรับควบคุมการแสดงสถานะของ LED ว่ามีการตรวจพบการเคลื่อนไหวหรือไม่ หากมีการเคลื่อนไหว หลอดไฟจะติดเป็นสีเขียว หากไม่มีจะดับหรือเปลี่ยนสถานะ
- V2 ใช้แสดงค่ามุมของเซนเซอร์รวมอเตอร์ผ่าน Gauge Widget โดยจะแสดงเป็นองศาระหว่าง 0 ถึง 180 องศา ซึ่งในระบบนี้จะใช้เพียงค่าระหว่าง 0 กับ 90 องศา เพื่อแสดงสถานะการเปิด–ปิดช่องให้อาหาร
- V3 ใช้แสดงข้อความว่าเซนเซอร์ตรวจพบการเคลื่อนไหวหรือไม่ โดยแสดงข้อความว่า "ON" เมื่อตรวจพบ และ "OFF" เมื่อไม่มีการเคลื่อนไหว

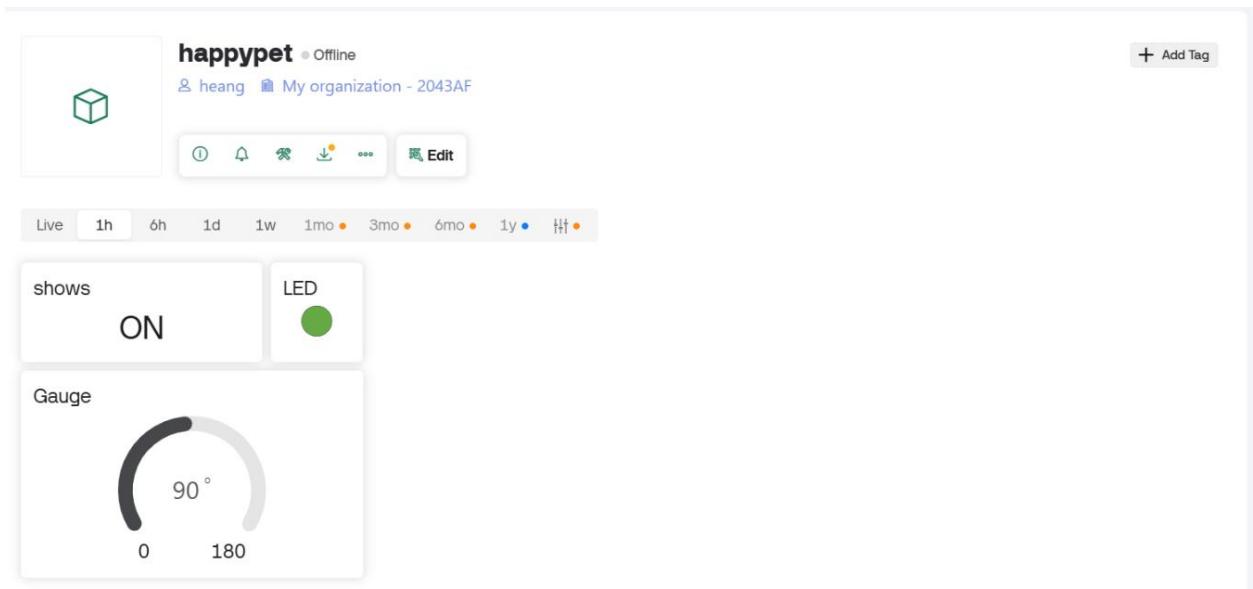
Datastreams

ID	Name	Pin	Color	Data Type	Units	Is Raw	Min	Max
2	LED	V1	Green	Integer		false	0	1
3	Gauge	V2	Red	Integer	°	false	0	180
4	shows	V3	Blue	String		false		

(รูปภาพที่ 16 : Datastream)



(รูปภาพที่ 17 : Dashboard บนแอปพลิเคชัน Blynk)



(รูปภาพที่ 18 : Dashboard บนเว็บไซต์ Blynk)

จากภาพที่ และ จะเห็นได้ว่ามีการแสดงผลค่าต่างๆ อย่างชัดเจน โดยข้อความ “shows” แสดงสถานะการตรวจจับ (ON/OFF), ส่วน Widget LED สีเขียวจะแสดงสถานะของระบบ และ Gauge จะแสดงองค์การหมุนของเซอร์วิโอเตอร์ที่กำลังทำงานอยู่

ข้อมูลทั้งหมดที่แสดงบน Blynk ถูกส่งผ่านทาง Node-RED ซึ่งรับข้อมูลจาก ESP32 ผ่าน MQTT และแปลงเป็นรูปแบบที่สามารถส่งเข้าสู่ Virtual Pin ของ Blynk ได้ ทำให้ระบบสามารถทำงานได้อย่างสอดคล้องกันระหว่างอาร์ดเวย์และแอปพลิเคชัน

การนำ Blynk มาใช้ในโครงการนี้ช่วยเพิ่มความสะดวกให้กับผู้ใช้งานอย่างมาก เพราะสามารถดูสถานะของเครื่องให้อาหารสัตว์เลี้ยงได้ทุกที่ทุกเวลา ผ่านอินเทอร์เน็ตโดยไม่จำเป็นต้องอยู่ใกล้ตัวเครื่อง อีกทั้งยังสามารถพัฒนาให้มีการควบคุมระยะไกลเพิ่มเติมในอนาคต เช่น การสั่งปล่อยอาหารด้วยตนเองผ่านปุ่มในแอปได้อีกด้วย

5. สรุปผลการศึกษา

จากการศึกษาและพัฒนาเครื่องให้อาหารสัตว์เลี้ยงอัตโนมัติในครั้งนี้ พบว่าสามารถนำเซนเซอร์ตรวจจับการเคลื่อนไหว (PIR Sensor), มอเตอร์เซอร์วิโอ (Servo Motor), ไฟ LED และปุ่มควบคุม มาทำงานร่วมกันผ่านบอร์ด ESP32 ได้อย่างมีประสิทธิภาพ โดยระบบสามารถตรวจจับการเคลื่อนไหวของสัตว์เลี้ยงบริเวณใกล้ตัวเครื่อง และทำการปล่อยอาหารโดยอัตโนมัติพร้อมแสดงสถานะด้วยไฟ LED

นอกจากนี้ยังได้เพิ่มปุ่มเปิด–ปิดระบบ เพื่อให้ผู้ใช้งานสามารถควบคุมการทำงานได้ตามต้องการ ซึ่งช่วยเพิ่มความปลอดภัยและประหยัดพลังงานในกรณีที่ไม่ต้องการให้เครื่องทำงานตลอดเวลา รวมถึงสามารถนำเทคโนโลยี IoT เช่น Node-RED และ Blynk มาประยุกต์ใช้ในการควบคุมและตรวจสอบสถานะของเครื่องจากระยะไกลได้อีกด้วย

จากการทดลองพบว่าระบบสามารถทำงานได้อย่างถูกต้องตามเงื่อนไขที่กำหนด และสามารถนำไปพัฒนาต่ออยอดสูงระบบให้อาหารอัตโนมัติที่ชาญฉลาดและมีความสามารถในการเข้ามาร่วมตอกย้ำได้ในอนาคต

วิดีโอการนำเสนอ

[วิดีโอการนำเสนอ ITDS282 GR11 Project : Happy Pet](#)