EE04-แบบขอสอบป้องกัน

นักศึกษา ☐ ปกติ ☐ สมทบ

การสอบป้องกันโครงงานวิศวกรรมไฟฟ้า

หลักสูตรสาขาวิชาวิศวกรรมไฟฟ้า

****

ชื่อโครงการ

ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT

IoT-Based Cattle Farm Management System

ผู้จัดทำ

นายยุทธนา เอียดมาก รหัสนักศึกษา 164404130053 โทร. 098-2911687

นายกฤตพล ยิ้มละมัย รหัสนักศึกษา 164404130079 โทร. 082-7061368

ความเห็นอาจารย์ที่ปรึกษาร่วม	**ความเห็นอาจารย์ที่ปรึกษาหลัก**
ได้ตรวจโครงงานแล้ว เห็นควร	ได้ตรวจโครงงานแล้ว เห็นควร
☐ อนุมัติให้สอบได้ ☐ ปรับปรุงเนื้อหาหรือรูปแบบใหม่	☐ อนุมัติให้สอบได้ ☐ ปรับปรุงเนื้อหาหรือรูปแบบใหม่
ลงชื่อ ..... (.....)	ลงชื่อ ..... (.....)

สาขาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลศรีวิชัย

ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT

IoT-Based Cattle Farm Management System

ยุทธนา เอียดมาก

Yuttana Aeadmak

กฤตพล ยิ้มละมัย

Krittapon Yimlamai

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

หลักสูตรสาขาวิชาวิศวกรรมไฟฟ้า

สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลศรีวิชัย

A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE BACHELOR DEGREE IN ELECTRICAL ENGINEERING

FACULTY OF ENGINEERING

RAJAMANGALA UNIVERSITY OF TECHNOLOGY SRIVIJAYA

2567

ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT

IoT-Based Cattle Farm Management System

ยุทธนา เอียดมาก

Yuttana Aeadmak

กฤตพล ยิ้มละมัย

Krittapon Yimlamai

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

หลักสูตรสาขาวิชาวิศวกรรมไฟฟ้า

สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลศรีวิชัย

A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE BACHELOR DEGREE IN ELECTRICAL ENGINEERING

FACULTY OF ENGINEERING

RAJAMANGALA UNIVERSITY OF TECHNOLOGY SRIVIJAYA

2567

ปริญญานิพนธ์ : ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT

ชื่อ : นายยุทธนา เอียดมาก

: นายกฤตพล ยิ้มละมัย

สาขาวิชา : วิศวกรรมไฟฟ้า

ประธานที่ปรึกษา

.....

(ดร.สลักจิตร แบลนชาร์ด)

กรรมการสอบ

.....ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์พิทักษ์ สถิตวรรธนะ)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์เกียติศักดิ์ ทองอ่อน)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ธีระพงษ์ ฉิมเพชร)

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลศรีวิชัย สงขลา อนุมัติให้ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมไฟฟ้า

.....

(ผู้ช่วยศาสตราจารย์ชาญณรงค์ พงศ์รักธรรม)

หัวหน้าหลักสูตร/ประธานหลักสูตร

Project Report Title : IoT-Based Cattle Farm Management System

Name : Mr. Yuttana Aeadmak

: Mr. Krittapon Yimlamai

Major Field : Electrical Engineering

Major Advisor

.....

(Mrs.Salakjit Blanchard)

Examining Committee

.....Chairperson

(Asst.Prof.Phitak Sathitwantana)

.....Committee

(Asst.Prof.Kiattisak Tongon)

.....Committee

(Asst.Prof. Teerapong Chimphet)

Accepted by the Faculty of Engineering, Rajamangala University of Technology Srivijaya in Partial Fulfillment of the Requirements for the Degree of Bachelor of Electrical Engineering.

.....

(Asst.Prof. Channarong Phongraktham)

Chief Curriculum/Head of Curriculum

หัวข้อปริญญานิพนธ์ : ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ **IoT**
โดย : นายยุทธนา เอียดมาก
: นายกฤตพล ยิ้มละมัย
อาจารย์ที่ปรึกษา : ดร.สลักจิตร แบลนชาร์ด
สาขาวิชา : วิศวกรรมไฟฟ้า
ปีการศึกษา : 2567

บทคัดย่อ

โครงการนี้มีวัตถุประสงค์เพื่อพัฒนาระบบควบคุมและติดตามสถานะฟาร์มปศุสัตว์ด้วยเทคโนโลยี Internet of Things (IoT) โดยใช้บอร์ด Raspberry Pi 4 เป็นศูนย์กลางในการควบคุมรีเลย์ เปิด-ปิดอุปกรณ์ไฟฟ้า และเชื่อมต่อกล้อง USB เพื่อแสดงภาพสดผ่านหน้าเว็บ นอกจากนี้ยังมีระบบประมวลผลภาพสำหรับการตรวจจับและนับจำนวนวัวภายในคอกพร้อมแสดงผลและส่งการแจ้งเตือนผ่านหน้าเว็บแอปพลิเคชัน เมื่อจำนวนวัวต่ำกว่าที่กำหนดระบบสามารถทำงานตามตารางเวลาโดยอัตโนมัติ บันทึกวิดีโอจากกล้องในรูปแบบ .mp4 และลบไฟล์เก่าอัตโนมัติเมื่อเกิน 7 วัน ผลการทดสอบแสดงให้เห็นว่าระบบมีความแม่นยำและเสถียร สามารถควบคุมอุปกรณ์และประมวลผลภาพได้อย่างมีประสิทธิภาพเหมาะสมสำหรับการใช้งานจริงในฟาร์ม และสามารถต่อยอดเป็นระบบ Smart Farm ได้ในอนาคต

คำสำคัญ : ระบบรับ-ส่งภาพ, กล้องนับจำนวน, ระบบไร้สาย

Project Report Title	: IoT-Based Cattle Farm Management System
Name	: Mr. Yuttana Aeadmak
	: Mr. Krittapon Yimlamai
Advisor	: Mrs.salakjit Blanchard
Major Field	: Electrical Engineering
Academic Years	: 2023

Abstract

This project aims to develop a livestock farm monitoring and control system using Internet of Things (IoT) technology. A Raspberry Pi 4 board is used as the central controller to operate relays for turning electrical devices on and off, as well as to connect a USB camera for live video streaming via a web interface. Additionally, the system includes image processing capabilities to detect and count the number of cows in the pen, display results, and send alerts through the web application when the number of cows falls below a predefined threshold. The system can operate automatically according to a set schedule, record video from the camera in .mp4 format, and automatically delete old files after 7 days. Test results show that the system is accurate and stable, capable of controlling devices and processing images efficiently. It is suitable for real-world farm applications and has the potential to be developed into a Smart Farm system in the future.

Keywords: image transmission system, cow counting camera, wireless system

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์จาก ดร.สลักจิตร แบลนชาร์ด ซึ่งเป็นอาจารย์ที่ปรึกษา ที่กรุณาให้คำปรึกษาข้อชี้แนะอันมีคุณค่าและก่อให้เกิดประโยชน์ในการดำเนินโครงงาน ตลอดจนแนะแนวทางการแก้ปัญหาต่าง ๆ ในการดำเนินโครงงานอย่างดียิ่ง อีกทั้งความเมตตาในการช่วยเหลือด้านต่าง ๆ ตลอดโครงงานนี้ ผู้จัดทำปริญญานิพนธ์ขอกราบขอบพระคุณอาจารย์ด้วยความซาบซึ้งใจและเคารพอย่างสูง

ขอกราบขอบพระคุณบิดา มารดา และครอบครัว ที่ให้คำปรึกษาในเรื่องต่าง ๆ ตลอดจนให้กำลังใจที่ดีเสมอมา จนทำให้ศึกษาเล่าเรียนจนสำเร็จการศึกษา

ขอขอบคุณเพื่อน ๆ นักศึกษาคณะวิศวกรรมศาสตร์สาขาวิชาวิศวกรรมไฟฟ้าที่ให้ข้อมูล และให้คำปรึกษาในด้านต่าง ๆ

ผู้จัดทำปริญญานิพนธ์ขอขอบพระคุณคณาจารย์และบุคลากรในหลักสูตรสาขาวิศวกรรมไฟฟ้าทุกท่านที่ได้กรุณาให้ความรู้ และคำปรึกษาแนะนำด้านการออกแบบโครงงานชุดนี้ให้ปริญญานิพนธ์มีความเรียบร้อยสมบูรณ์

สุดท้ายนี้ ขอขอบพระคุณ บิดา มารดา ญาติพี่น้องคณาจารย์ทุกท่าน ตลอดจนเพื่อนๆที่คอยเป็นกำลังใจ จนกระทั่งคณะผู้จัดทำมีโอกาสทำปริญญานิพนธ์ฉบับนี้ได้สำเร็จจึงขอขอบคุณทุกท่านทุกฝ่ายและขอประกาศเป็นคุณงามความดี มา ณ ที่นี้ด้วย

ยุทธนา เอียดมาก

กฤตพล ยิ้มละมัย

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

บทนี้จะกล่าวถึงความสำคัญและที่มาของหัวข้อโครงงาน วัตถุประสงค์ของโครงงานขอบเขตของโครงงาน ประโยชน์ที่คาดว่าจะได้รับ ขั้นตอนและวิธีการดำเนินการของโครงงาน รวมถึงแผน ของโครงงาน ประโยชน์ที่คาดว่าจะได้รับ ขั้นตอนและวิธีการดำเนินการของโครงงาน รวมถึงแผน

1.1 ความสำคัญและที่มาของหัวข้อโครงงาน

ในปัจจุบันเทคโนโลยีสารสนเทศ และการสื่อสารมีบทบาทในการพัฒนาอุตสาหกรรมการเกษตร โดยเฉพาะอย่างยิ่งในด้านการเลี้ยงสัตว์ ซึ่งจำเป็นต้องมีการจัดการที่มีประสิทธิภาพ ทั้งในเรื่องของการให้อาหาร การดูแลสุขภาพ และการควบคุมสิ่งแวดล้อมในฟาร์ม อย่างไรก็ตาม ฟาร์มเลี้ยงโคเนื้อจำนวนมากยังคงดำเนินการในรูปแบบดั้งเดิม ซึ่งต้องอาศัยแรงงานมนุษย์จำนวนมาก มีความเสี่ยงต่อข้อผิดพลาดในการดูแลและมีต้นทุนในการดำเนินงานสูง

ด้วยเหตุนี้ การนำระบบ IoT (Internet of Things) มาประยุกต์ใช้ในฟาร์มจึงเป็นแนวทางที่ช่วยเพิ่มประสิทธิภาพในการบริหารจัดการฟาร์มได้อย่างมีระบบ โดยสามารถตรวจสอบ ควบคุม และแจ้งเตือนสภาพแวดล้อม หรือกิจกรรมต่าง ๆ ภายในฟาร์มได้แบบเรียลไทม์ผ่านอินเทอร์เน็ต เช่น การเปิด-ปิดไฟหรือพัดลมอัตโนมัติ การตรวจสอบจำนวนโคในโรงเรือน หรือแม้กระทั่งการเฝ้าระวังกล้องวงจรปิดผ่านอุปกรณ์เคลื่อนที่

ระบบควบคุมฟาร์มผ่าน IoT ไม่เพียงแต่ช่วยลดภาระงานของแรงงานภายในฟาร์มเท่านั้น แต่ยังเพิ่มความแม่นยำในการบริหารจัดการ ทำให้สามารถดูแลสัตว์เลี้ยงได้อย่างทั่วถึง ลดอัตราการสูญเสีย และเพิ่มผลผลิตทางเศรษฐกิจได้ นอกจากนี้ยังเป็นแนวทางในการพัฒนาฟาร์มเข้าสู่เกษตรอัจฉริยะ (Smart Farm) ซึ่งเป็นเป้าหมายสำคัญของการพัฒนาเกษตรกรรมไทยในยุคปัจจุบันดังนั้น โครงงานนี้จึงมีความสำคัญในฐานะเป็นต้นแบบของระบบควบคุมฟาร์มเลี้ยงโคเนื้อ โดยใช้ต้นทุนต่ำ ติดตั้งง่าย และสามารถประยุกต์ใช้ได้จริงในระดับฟาร์มครัวเรือนหรือเชิงพาณิชย์

1.2 วัตถุประสงค์

1.2.1 เพื่อออกแบบระบบควบคุมอุปกรณ์ไฟฟ้าภายในฟาร์มเลี้ยงโคเนื้อผ่านเทคโนโลยี IoT

1.2.2 เพื่อเพิ่มประสิทธิภาพในการจัดการฟาร์มด้วยระบบควบคุมและตรวจสอบระยะไกล

1.2.3 เพื่อลดการพึ่งพาแรงงานคนและลดข้อผิดพลาดในการดำเนินงาน

1.2.4 เพื่อทดลองระบบนับจำนวนโคผ่านกล้องและแจ้งเตือนเมื่อจำนวนไม่เป็นไปตามที่กำหนด

1.3 ขอบเขตของโครงงาน

1.3.1 ออกแบบและสร้างระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT ประกอบด้วย กล่องควบคุมจำนวน 1 กล่อง และ 1 แอปพลิเคชัน

1.3.2 ระบบสามารถสั่งเปิด-ปิด และตั้งเวลาควบคุม 8 เอาท์พุต

1.3.3 ระบบสามารถรับสัญญาณจากกล้องไม่น้อยกว่า 1 กล้อง และแสดงภาพบนแอปพลิเคชันได้และสามารถเรียกดูภาพย้อนหลังได้ไม่น้อยกว่า 1 สัปดาห์

1.3.4 สามารถนับจำนวนวัวภายในคอก และแจ้งเตือนเมื่อผิดปกติ

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 ได้ระบบควบคุมอุปกรณ์ในฟาร์มที่สามารถสั่งงานและตรวจสอบได้จากระยะไกล

1.4.2 ลดภาระแรงงานและความผิดพลาดจากการควบคุมด้วยมนุษย์

1.4.3 เพิ่มประสิทธิภาพด้านเวลา ความสะดวก และความปลอดภัยในการจัดการฟาร์ม

1.4.4 ตรวจสอบจำนวนโคได้แบบอัตโนมัติ ลดความเสี่ยงจากการสูญหาย

1.5 ขั้นตอนและวิธีการดำเนินการวิจัย

1.5.1 จัดหาหัวข้อโครงงาน

1.5.2 ศึกษาค้นคว้าหาขอมูลเกี่ยวกับโครงงาน

1.5.3 เข้าปรึกษาและพูดคุยกับอาจารย์ที่ปรึกษาโครงงาน

1.5.4 ออกแบบโครงสร้างชิ้นงาน

1.5.5 สอบหัวข้อโครงงาน

1.5.6 ต่อวงจรการทำงานของบอร์ด raspberry pi 4

1.5.7 เขียนโปรแกรมไมโครคอนโทรลเลอร์

1.5.8 ทดสอบการทำงาน

1.5.9 รายงานความก้าวหน้า

1.5.10 ปรับปรุงและทดสอบการทำงานอีกครั้ง

1.5.11 จัดทำปริญญานิพนธ์

1.5.12 สอบจบโครงงาน

1.6 แผนการดำเนินงาน

ตารางที่ 1-1 แผนการดำเนินงาน

ระยะเวลาในการดำเนินงาน	ต.ค. 67	พ.ย. 67	ธ.ค. 67	ม.ค. 68	ก.พ. 68	มี.ค. 68	เม.ย. 68	พ.ค. 68	มิ.ย. 68
1.กำหนดหัวข้อโครงงาน									
2.ออกแบบระบบควบคุม และโครงสร้างชิ้นงาน									
3.จัดทำโครงงาน นำเสนอและขออนุมัติ									
4.ศึกษาอุปกรณ์ และจัดซื้ออุปกรณ์ที่นำมาทำโครงงาน									
5.จัดทำโครงสร้างระบบ									
6.ฝึกเขียนโปรแกรมการทำงาน									
7.ทดสอบการทำงาน									
8.แก้ไขงานผิดพลาด									
9.ประเมินผลที่ได้ และสรุปผล									
10.จัดทำปริญญานิพนธ์									

บทที่ 2
งานวิจัย และทฤษฎีบทที่เกี่ยวข้อง

2.1 งานวิจัยที่เกี่ยวข้อง

ณัฐพล ธนเชวงสกุล, ธีระพงษ์ ฤทธิ์มาก, ปรีชา โคตะพัฒน์ (2560) กล่าวไว้ว่า ความต้องการสินค้าปศุสัตว์ยังมีแนวโน้มเพิ่มขึ้น โดยเฉพาะความต้องการของ ตลาดต่างประเทศที่มีความเชื่อมั่นในมาตรฐานการผลิตและคุณภาพสินค้าปศุสัตว์ของไทย

Thitisak Pothong et al. [5] ได้ทำวิจัยเรื่อง การพัฒนาระบบฟาร์มอัจฉริยะสำหรับเกษตรกรยุคใหม่ด้วย ซอฟต์แวร์รหัสเปิด และอินเทอร์เน็ตของสรรพสิ่ง มีวัตถุประสงค์คือ ผลิตชุดควบคุมสำหรับตรวจวัดสภาพแวดล้อม เช่น อุณหภูมิ และความชื้นในอากาศ รวมไปถึงความชื้นในดินเพื่อให้บุคคลทั่วไปสามารถใช้งานได้ง่ายสะดวกสบาย ประหยัดเวลา ผ่านระบบเครือข่ายอินเทอร์เน็ต (Internet) ด้วยเทคโนโลยี 3G, 4G หรือ WiFi และควบคุมอุปกรณ์ต่าง ๆ ผ่านมือถือสมาร์ทโฟน ผลลัพธ์ที่ได้ข้อมูลที่ส่งจากเครือข่ายเซนเซอร์ไร้สายเป็นข้อมูลสภาพแวดล้อมเป็นปัจจุบัน และข้อมูลมีการเปลี่ยนแปลงอยู่ตลอด

พชรพล จันทาทอง , สุวิมล มรรควิบูลย์ชัย (2566) ได้ทำวิจัยเรื่อง ความต้องการของระบบโดยระบบที่พัฒนาขึ้นมีการทำงานเป็นไปอย่างมีประสิทธิภาพ และมีความถูกต้องในเรื่องของการควบคุมการเปิด-ปิดหลอดไฟ พัดลม และให้อาหารผ่านแอปพลิเคชัน สามารถดูการแจ้งเตือนอุณหภูมิ ผ่านแอปพลิเคชันไลน์ได้ทำให้ผู้ใช้งานมีความสะดวกมากขึ้น สอดคล้องกับ ชัยเดช อินทร์ชัยศรี,ปิยะณัฐ ประสมศร (2561) กล่าวไว้ว่า การจัดการฟาร์มโคนมมีการเปลี่ยนแปลงอย่างรวดเร็วในช่วง 2-3 ปีที่ผ่านมา โดยมีการนำเทคโนโลยีที่มีความแม่นยำสูงมาใช้ในการจัดการโคเพิ่มขึ้นอย่างมาก การปรับเปลี่ยนระบบการจัดการด้วยเทคโนโลยีที่มีความแม่นยำสูงนี้ช่วยให้การจัดการฟาร์มโคนมและตัวโคมีประสิทธิภาพในการผลิต

2.2 ทฤษฎีที่เกี่ยวข้อง

2.2.1 โคพันธุ์บราห์มัน

โคสายพันธุ์อเมริกันบราห์มัน เป็นโคขนาดกลาง เพศผู้มีน้ำหนักตัวประมาณ 800 – 900 กิโลกรัม และมีพ่อพันธุ์บางตัวน้ำหนักโตเต็มที่มากถึง 1,800 กิโลกรัม ในประเทศไทยก็ยังมีให้เห็น ส่วนเพศเมีย จะมีน้ำหนักมาตรฐาน 500 – 700 กิโลกรัม และมากที่สุดถึง 800 กิโลกรัม โดยแม่โคจะให้นมลูก

เมื่อน้ำหนักแรกเกิดปานกลาง (30 – 32 กิโลกรัม) และน้ำหนักลูกเมื่อหย่านมค่อนค้างน้อย (220 – 230 กิโลกรัม)

รูปที่ 2-1 โคพันธุ์บราห์มัน

(ที่มา: https://taradko.com/-brahman/)

ลักษณะของโคบราห์มัน เป็นโคที่มีเขาขึ้นชันและงุ้ม มีตระโหนก ซึ่งเป็นกล้ามเนื้อที่ต่อมาจากกล้ามเนื้อไหล่ทั้งเพศผู้และเพศเมีย ส่วนหูนั้นจะยาวชี้ลงข้างล่างจนถึงริมผีปาก แต่จะสั้นกว่าพันธุ์อินดูบราซิล มีหนังหลวม เหนียงใต้คางจะใหญ่ หนอกใหญ่และหย่อน ขณะที่ผิวหนังใต้ท้องก็จะหย่อน พู่หางสีดำ ส่วนขานั้นค่อนข้างยาว และจะมีกล้ามเนื้อตรงขาหลังมาก โคนขาใหญ่ มีร่างกายล่ำสัน ลำตัวมีความลึกมาก และยาวได้สัดส่วน หน้าผากยาว คอสั้น ส่วนอกกว้างลึก หลังค่อนข้างตรงเป็นโคที่เลี้ยงดูง่าย ต้องการดูแลน้อย เหมาะสำหรับเลี้ยงในเขตร้อน เพราะมีความทนทานต่อสภาพอากาศร้อนได้ดี ทนต่อโรคและแมลงได้ดี แต่ไม่ทนต่อสภาพอากาศหนาวเย็น อีกทั้งแม่โคยังเลี้ยงลูกเก่ง ให้น้ำนมพอสมควร ใช้เป็นทั้งโคพันธุ์เนื้อและโคงาน มีการเจริญเติบโตค่อนข้างช้า มีความสมบูรณ์พันธุ์ (Fertility) ค่อนข้างต่ำ เปอร์เซ็นของซากต่ำ เมื่อเทียบกับโคสายพันธุ์อื่น (ชาโรเล่ย์ แองกัส แบรงกัส) มีนิสัยระวังภัยสูง เชื่องช้าและขี้เล่น ถ้าอยู่ในลักษณะคอกหรือโรงเรือน เพศเมียผสมพันธุ์ได้เมื่อมีอายุราว 1.8-2 ปี ซึ่งค่อนข้างช้ากว่าโคเนื้อพันธุ์ยุโรป แม่พันธุ์นั้นสามารถให้ลูกไปได้จนถึงอายุ 15 ปี ปกติใช้เป็นพันธุ์พื้นฐานในการปรับปรุงพันธุ์ ลูกผสมที่ได้จะมีลักษณะดีเด่นกว่าพ่อแม่ (Hybrid Vigor) กล้ามเนื้อของโคพันธุ์บราห์มัน สามารถขยับตัวได้เพื่อไล่แมลง และผิวหนังบริเวณคอมีการขับสารสีเหลืองออกมาซึ่งเชื่อว่าเป็นกลิ่นที่สามารถไล่แมลงได้ตามธรรมชาติ

2.2.2 กล้องวงจรปิด หรือ CCTV (Closed-Circuit Television)

กล้องวงจรปิด เป็นระบบกล้องถ่ายภาพวิดีโอที่สามารถส่งสัญญาณภาพไปยังจุดแสดงผลหรือบันทึกข้อมูลโดยเฉพาะเจาะจง ซึ่งต่างจากการแพร่ภาพแบบสาธารณะ กล้องวงจรปิดมักถูกนำมาใช้ในด้านการรักษาความปลอดภัย การเฝ้าระวัง และการตรวจสอบพฤติกรรมต่าง ๆ ทั้งในบ้านพัก อาศัยโรงงานอาคารหรือแม้กระทั่งในฟาร์มเลี้ยงสัตว์

ความละเอียดของกล้อง (Resolution) ความละเอียดเป็นปัจจัยสำคัญที่ส่งผลต่อความคมชัดของภาพ โดยทั่วไปจะวัดเป็นพิกเซล เช่น 720p (HD), 1080p (Full HD), หรือ 4K (Ultra HD) ยิ่งมีความละเอียดสูง ยิ่งเหมาะสำหรับการตรวจจับรายละเอียด เช่น ใบหน้าหรือป้ายทะเบียนรถ เทคโนโลยีอินฟราเรด (Infrared: IR) กล้องบางรุ่นติดตั้งหลอด IR LED เพื่อให้สามารถบันทึกภาพในที่มืดหรือกลางคืนได้ เรียกว่ากล้องอินฟราเรด (IR Camera) ซึ่งเหมาะสำหรับการติดตั้งในพื้นที่ห่างไกลหรือในฟาร์มที่มีแสงสว่างน้อย

2.2.3 ไมโครคอนโทรลเลอร์ (Microcontroller)

ไมโครคอนโทรลเลอร์ (Microcontroller Unit: MCU) คือวงจรอิเล็กทรอนิกส์ที่รวมเอาหน่วยประมวลผลกลาง (CPU), หน่วยความจำ (Memory) และวงจรสำหรับเชื่อมต่ออุปกรณ์อินพุต/เอาต์พุต (I/O) ไว้ในชิปเดียวกัน โดยทั่วไปไมโครคอนโทรลเลอร์จะถูกฝังอยู่ในระบบฝังตัว (Embedded System) ซึ่งหมายถึงระบบที่ออกแบบมาเฉพาะสำหรับงานใดงานหนึ่ง เช่น ระบบควบคุมเครื่องใช้ไฟฟ้า อุปกรณ์อิเล็กทรอนิกส์ ระบบอัตโนมัติในโรงงาน เครื่องมือแพทย์ หรือแม้กระทั่งหุ่นยนต์

ไมโครคอนโทรลเลอร์มีหน้าที่หลักในการรับข้อมูลจากอุปกรณ์อินพุต เช่น เซนเซอร์ ปุ่มกด หรือโมดูลอื่น ๆ จากนั้นจึงทำการประมวลผลข้อมูลที่ได้รับตามคำสั่งที่โปรแกรมไว้ในหน่วยความจำ และส่งผลลัพธ์ออกไปยังอุปกรณ์เอาต์พุต เช่น ไฟ LED จอแสดงผล มอเตอร์ หรือโมดูลสื่อสารต่าง ๆ

องค์ประกอบหลักของไมโครคอนโทรลเลอร์

1. หน่วยประมวลผลกลาง (CPU) เป็นสมองของไมโครคอนโทรลเลอร์ ทำหน้าที่คำนวณประมวลผล และควบคุมการทำงานต่าง ๆ ตามโปรแกรมที่กำหนดไว้
2. หน่วยความจำ (Memory) ROM ทำหน้าที่เก็บโปรแกรมถาวรที่ผู้พัฒนาเขียนขึ้น RAM ทำหน้าที่เก็บข้อมูลชั่วคราวระหว่างการทำงาน เช่น ค่าตัวแปร หรือผลลัพธ์การคำนวณ

3. พอร์ตอินพุต/เอาต์พุต (I/O Ports) ใช้ในการเชื่อมต่อกับอุปกรณ์ภายนอก เช่น เซนเซอร์ สวิตช์ มอเตอร์ หรือจอแสดงผล
4. ตัวจับเวลา (Timers/Counters) สำหรับกำหนดเวลาการทำงาน เช่น หน่วงเวลา หรือการนับเหตุการณ์
5. โมดูลสื่อสาร (Communication Modules) เช่น UART, SPI, I2C สำหรับการติดต่อกับอุปกรณ์ภายนอกหรือไมโครคอนโทรลเลอร์ตัวอื่น
6. แหล่งจ่ายพลังงาน (Power Supply) โดยทั่วไปไมโครคอนโทรลเลอร์จะทำงานที่แรงดัน 3.3V หรือ 5V

2.2.4 เอชทีเอ็มแอล (HTML: Hypertext Markup Language)

เอชทีเอ็มแอล คือภาษาเครื่องหมาย (Markup Language) หลักที่ใช้ในการสร้าง และกำหนดโครงสร้างเนื้อหาของเว็บเพจ หรือข้อมูลอื่น ๆ ที่สามารถเข้าถึงได้ผ่านเว็บเบราว์เซอร์ในปัจจุบัน โดย HTML ใช้การจัดระเบียบเนื้อหาผ่านการใช้แท็ก (tags) ต่าง ๆ เช่น ข้อความ, รูปภาพ, ลิงก์, ตาราง และองค์ประกอบอื่น ๆ ที่ปรากฏบนหน้าเว็บไซต์

HTML มีบทบาทสำคัญในการกำหนดโครงสร้าง และความหมายของเนื้อหาบนหน้าเว็บ เช่น การระบุว่าข้อความใดควรแสดงเป็นหัวข้อ หรือส่วนใดที่ต้องการเป็นลิงก์ไปยังหน้าหรือเว็บไซต์อื่น ๆ อย่างไรก็ตาม HTML ไม่ได้มีหน้าที่ในการจัดการเรื่องการออกแบบ หรือรูปแบบการแสดงผล ซึ่งจะเป็นหน้าที่ของ CSS (Cascading Style Sheets) และการกำหนดพฤติกรรมหรือการทำงานของหน้าเว็บนั้น ๆ จะเป็นการควบคุมโดย JavaScript

HTML ได้รับการพัฒนาเป็นเวอร์ชันต่าง ๆ โดยเวอร์ชันปัจจุบันที่ได้รับความนิยมคือ HTML5 ซึ่งมีความสามารถ รองรับการฝังวิดีโอและเสียงโดยไม่ต้องใช้ปลั๊กอิน รองรับแท็กใหม่สำหรับโครงสร้างหน้า เช่น <header>, <footer>, <article>, <section> รองรับการทำงานบนอุปกรณ์พกพาและหน้าจอขนาดต่าง ๆ

2.2.5 Flask

Flask คือเฟรมเวิร์ก (Framework) สำหรับการพัฒนาเว็บแอปพลิเคชัน (Web Application) ที่เขียนด้วยภาษา Python โดยมีลักษณะเป็น “Micro Web Framework” ซึ่งหมายความว่า Flask มีแกนกลางที่มีขนาดเล็ก ไม่บังคับโครงสร้างโครงการที่ซับซ้อน และไม่ผูกติดกับไลบรารีเฉพาะใด ๆ นักพัฒนาจึงสามารถออกแบบโครงสร้างโค้ดและเลือกใช้ไลบรารีเสริม (Extension) ตามความเหมาะสมของงานได้อย่างอิสระ ทำให้ Flask เหมาะกับทั้งงานขนาดเล็กไปจนถึงโครงการขนาดใหญ่ในระดับองค์กร แม้ว่า Flask จะเป็น micro framework แต่ก็มีความสามารถที่เพียงพอต่อการพัฒนาเว็บแอปพลิเคชันแบบเต็มรูปแบบ โดยเฉพาะเมื่อทำงานร่วมกับไลบรารีเสริม เช่น การจัดการผู้ใช้ (User Login), ฐานข้อมูล (Database), แบบฟอร์ม (Form Validation) หรือ RESTful API

คุณสมบัติของ Flask

1. Routing Flask รองรับการกำหนดเส้นทาง (Routing) ได้อย่างยืดหยุ่น ผ่านการใช้ decorator @app.route() ซึ่งช่วยกำหนดว่า เมื่อผู้ใช้เข้าถึง URL ใด ระบบจะเรียกใช้ฟังก์ชันใด ตัวอย่างเช่น การเข้าถึง /home สามารถให้เรียกใช้ฟังก์ชันแสดงหน้าแรก หรือหน้าแสดงข้อมูลได้โดยตรง
2. Template Rendering Flask มีโหมด Debug ที่ช่วยให้นักพัฒนาสามารถเห็นข้อผิดพลาดของโปรแกรมได้ทันทีขณะรันเว็บแอป โดยจะแสดงรายละเอียด error และ stack trace บนหน้าเว็บ อีกทั้งยังสามารถตั้งค่าให้ระบบรีโหลดแอปอัตโนมัติเมื่อมีการแก้ไขโค้ด
3. Template Rendering Flask ใช้ Jinja2 ซึ่งเป็น engine สำหรับการประมวลผลเทมเพลต HTML ที่ทรงพลัง ช่วยให้นักพัฒนาสามารถฝังตัวแปร คำสั่งลูป และเงื่อนไขลงใน HTML ได้โดยตรง เพิ่มความยืดหยุ่นในการแสดงผล เช่น การสร้างตารางจากข้อมูล หรือแสดงข้อความตามเงื่อนไขที่กำหนด
4. Static Files Flask ช่วยให้สามารถจัดการไฟล์คงที่ เช่น CSS, JavaScript, และรูปภาพ ได้อย่างง่ายดายผ่านโฟลเดอร์ static/ ซึ่งจะถูกใช้งานร่วมกับเทมเพลต HTML ในการสร้างส่วนติดต่อผู้ใช้ที่มีความสมบูรณ์

5. Request / Response Flask รองรับ HTTP Methods มาตรฐาน ได้แก่ GET, POST, PUT, DELETE และสามารถจัดการข้อมูลที่รับเข้ามา เช่น จากแบบฟอร์ม (form), URL (args), และ JSON ได้อย่างยืดหยุ่นผ่านโมดูล request และ response

รูปที่ 2-2 Flask

(ที่มา: urlkub.co/DMzrxw)

2.2.6 YOLO (You Only Look Once)

YOLO คืออัลกอริธึมการตรวจจับวัตถุ (Object Detection Algorithm) ที่มีความสามารถในการตรวจจับวัตถุหลายประเภทภายในภาพนิ่งหรือวิดีโอได้อย่างแม่นยำและรวดเร็วใน“ครั้งเดียว”โดยไม่ต้องแบ่งภาพเป็นหลายส่วนเหมือนอัลกอริธึมรุ่นเก่า

YOLO ถูกออกแบบมาให้สามารถตรวจจับวัตถุได้แบบ Real-Time (ความเร็วสูง) เหมาะสำหรับงานที่ต้องประมวลผลภาพทันที เช่น กล้องวงจรปิด, ยานยนต์ไร้คนขับ, หุ่นยนต์ และระบบฟาร์มอัจฉริยะ

หลักการทำงานของ YOLO

1. YO LO รับอินพุตเป็นภาพทั้งภาพ โดยขนาดที่นิยมใช้คือ 416x416 หรือ 640x640 พิกเซล เพื่อให้สามารถทำงานร่วมกับโครงข่ายประสาทเทียมได้อย่างมีประสิทธิภาพ
2. ภาพจะถูกแบ่งออกเป็นกริด เช่น 13x13 หรือ 19x19 ขึ้นอยู่กับเวอร์ชันของ YOLO และความละเอียดที่ใช้ แต่ละกริดจะเป็นตัวแทนพื้นที่ย่อยของภาพ
3. YOLO จะทำการตรวจสอบว่าในช่องกริดนั้นมีวัตถุอยู่หรือไม่ โดยพยากรณ์ค่าความมั่นใจที่เรียกว่า Objectness Score หรือ Confidence จากนั้นจะทำการคำนวณค่าพิกัดของกรอบครอบวัตถุ (Bounding Box) ได้แก่ ค่าตำแหน่ง x และ y รวมถึงความกว้างและความสูงของ

กรอบ (w, h) พร้อมทั้งทำการจำแนกประเภทของวัตถุ (Class Label) เช่น cow, person หรือ car โดยใช้ฟังก์ชัน Softmax หรือ Sigmoid ในการจำแนกประเภทวัตถุทั้งหมด ซึ่งขั้นตอนเหล่านี้จะถูกประมวลผลผ่านโครงข่ายประสาทเทียมแบบ Convolutional Neural Network (CNN)

4. YOLO จะรวมกรอบทั้งหมดที่พยากรณ์ได้ แล้วใช้เทคนิค Non-Maximum Suppression (NMS) เพื่อลดจำนวนกรอบซ้ำซ้อน โดยจะเลือกกรอบที่มีค่าความมั่นใจสูงที่สุดเท่านั้น

2.2.7 OpenCV

OpenCV (Open Source Computer Vision Library) คือไลบรารีโอเพนซอร์สสำหรับประมวลผลภาพและวิดีโอ (Image & Video Processing) ที่เขียนด้วยภาษา C/C++ แต่สามารถใช้งานผ่านภาษา Python, Java และภาษาอื่น ๆ ได้อย่างสะดวก โดย OpenCV ถูกพัฒนาขึ้นครั้งแรกโดยบริษัท Intel ในปี ค.ศ. 1999 และเปิดเผยเป็นโอเพนซอร์สในปี 2000 โดยมีจุดประสงค์เพื่อส่งเสริมการวิจัยด้านการประมวลผลภาพ (Computer Vision) และการประยุกต์ใช้กับระบบจริง เช่น หุ่นยนต์ กล้องอัจฉริยะ ระบบจดจำใบหน้า ฯลฯ

แม้ว่า OpenCV จะถูกพัฒนาด้วยภาษา C/C++ เป็นหลัก แต่ก็สามารถใช้งานได้อย่างสะดวกผ่านภาษาโปรแกรมอื่น ๆ เช่น Python, Java และ MATLAB ซึ่งช่วยให้นักพัฒนาและนักวิจัยสามารถเข้าถึงและใช้งานไลบรารีได้ง่ายขึ้น โดยเฉพาะในโครงการต้นแบบหรืองานที่ต้องการการประมวลผลแบบเรียลไทม์

OpenCV มีฟังก์ชันที่หลากหลาย ครอบคลุมตั้งแต่การประมวลผลเบื้องต้นไปจนถึงขั้นสูง ตัวอย่างเช่น การอ่านและแสดงผลภาพหรือวิดีโอ การแปลงรูปแบบสีระหว่างระบบสีต่าง ๆ เช่น RGB, Grayscale, และ HSV การตรวจจับใบหน้า วัตถุ หรือคุณลักษณะเฉพาะ (Feature Detection) ของภาพ การวิเคราะห์การเคลื่อนไหว และการประมวลผลวิดีโอแบบเรียลไทม์ ซึ่งเป็นสิ่งจำเป็นในระบบเฝ้าระวังความปลอดภัย และหุ่นยนต์

มีฟังก์ชันหลากหลาย ครอบคลุมงานด้าน Computer Vision เช่น

1. อ่านและแสดงผลภาพและวิดีโอ (Read/write/display images and video)
2. แปลงรูปแบบสี (เช่น RGB ↔ Grayscale ↔ HSV)
3. การตรวจจับใบหน้า วัตถุ หรือลักษณะเฉพาะ (Feature Detection)

4. การประมวลผลวิดีโอแบบเรียลไทม์ (Real-time video processing)
5. การตรวจจับการเคลื่อนไหว (Motion Detection)
6. การแปลงภาพ เช่น การหมุน, ซูม, ตัดขอบ และบิดเบี้ยว
7. การประมวลผลภาพขั้นสูง เช่น การทำ Filter, Edge Detection (Canny), Thresholdin
8. เชื่อมต่อกับกล้อง (USB, Pi Camera, IP Camera ฯลฯ)
9. การวิเคราะห์วัตถุร่วมกับโมเดล AI เช่น YOLO, TensorFlow, OpenVINO

2.2.8 ภาษา Python

Python เป็นภาษาระดับสูง (High-level Programming Language) ที่ได้รับความนิยมอย่างแพร่หลายในการพัฒนาโปรแกรมทุกประเภท ไม่ว่าจะเป็นด้านวิทยาศาสตร์ข้อมูล (Data Science), ปัญญาประดิษฐ์ (AI), การประมวลผลภาพ, เว็บแอปพลิเคชัน รวมถึงระบบสมองกลฝังตัว (Embedded Systems) บนบอร์ดไมโครคอนโทรลเลอร์หรือคอมพิวเตอร์ขนาดเล็ก เช่น Raspberry Pi

ภาษา Python ถูกพัฒนาโดย Guido van Rossum ในปี ค.ศ. 1991 โดยมีเป้าหมายเพื่อสร้างภาษาที่ใช้งานง่าย มีไวยากรณ์ที่ชัดเจน และส่งเสริมการเขียนโปรแกรมเชิงโครงสร้างและเชิงวัตถุ (OOP) ปัจจุบัน Python มีหลายเวอร์ชัน โดยเวอร์ชันที่นิยมใช้คือ Python 3.x

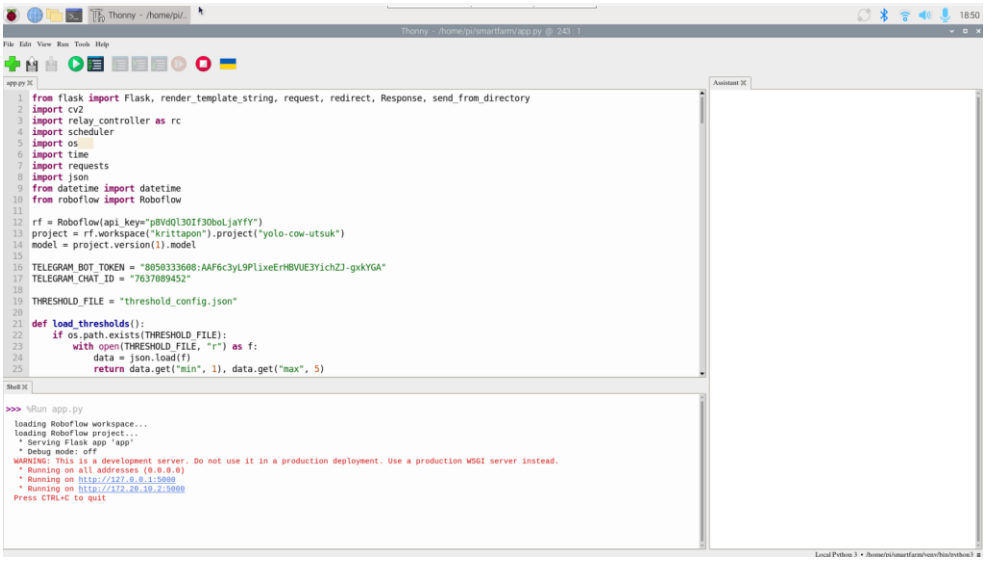
คุณลักษณะสำคัญของภาษา Python

1. เป็นภาษาที่อ่านง่าย คล้ายภาษาอังกฤษ
2. ใช้การจัดย่อหน้า (Indentation) แทนการใช้วงเล็บเปิดปิด
3. รองรับการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming)
4. มีไลบรารีจำนวนมาก เช่น OpenCV, NumPy, Flask, Requests, GPIO Zero ฯลฯ
5. สามารถทำงานข้ามแพลตฟอร์มได้ (Cross-Platform)
6. มีชุมชนนักพัฒนาขนาดใหญ่และเอกสารประกอบมากมาย

โครงสร้างพื้นฐานของภาษา Python

1. ตัวแปร (Variables) ใช้เก็บข้อมูลชนิดต่าง ๆ เช่น ตัวเลข, ข้อความ, ลิสต์ ฯลฯ
2. คำสั่งควบคุมการไหล (Control Flow) เช่น if, for, while
3. ฟังก์ชัน (Functions) ใช้สำหรับแยกโค้ดออกเป็นหน่วยย่อยที่สามารถเรียกใช้งานซ้ำได้

4. โมดูลและไลบรารี (Modules and Libraries) ช่วยให้การพัฒนาโปรแกรมซับซ้อนง่ายขึ้น เช่น Flask สำหรับเว็บ, OpenCV สำหรับกล้อง, Telepot สำหรับ Telegram



รูปที่ 2-3 ภาษา Python

2.2.9 VLC (VLC media player)

VLC เป็นโปรแกรมเล่นสื่อมัลติมีเดียแบบโอเพนซอร์สที่ได้รับความนิยมอย่างแพร่หลายทั่วโลก โดยได้รับการพัฒนาและดูแลโดย VideoLAN Project จุดเด่นของ VLC คือความสามารถในการรองรับการเล่นไฟล์มัลติมีเดียหลากหลายรูปแบบโดยไม่จำเป็นต้องติดตั้งโค้ดเสริมหรือปลั๊กอินเพิ่มเติม ทำให้ผู้ใช้สามารถเปิดใช้งานไฟล์วิดีโอและเสียงได้ทันที ไม่ว่าจะเป็นไฟล์ประเภท MP4, MKV, AVI, MP3, FLAC และอีกมากมาย VLC ยังมีความสามารถในการทำงานข้ามแพลตฟอร์มอย่างยอดเยี่ยม โดยรองรับระบบปฏิบัติการหลายระบบ เช่น Windows, macOS, Linux, Android และ iOS นอกจากนี้ยังมีอินเทอร์เฟซที่ใช้งานง่าย เหมาะทั้งสำหรับผู้ใช้ทั่วไปและนักพัฒนาที่ต้องการปรับแต่งหรือใช้งานขั้นสูง

คุณสมบัติที่ทำให้ VLC แตกต่างจากโปรแกรมเล่นสื่ออื่น ๆ คือการรองรับฟีเจอร์ขั้นสูง เช่น การสตรีมวิดีโอผ่านเครือข่าย (Streaming) ซึ่งช่วยให้ผู้ใช้สามารถรับส่งข้อมูลวิดีโอจากคอมพิวเตอร์หนึ่งไปยังอุปกรณ์อื่นในเครือข่ายได้แบบเรียลไทม์, การบันทึกวิดีโอ (Recording) จากแหล่งต่าง ๆ เช่น กล้อง USB หรือกล้อง IP, การแปลงรูปแบบไฟล์ (File Conversion) เพื่อให้เหมาะสมกับอุปกรณ์หรือการ

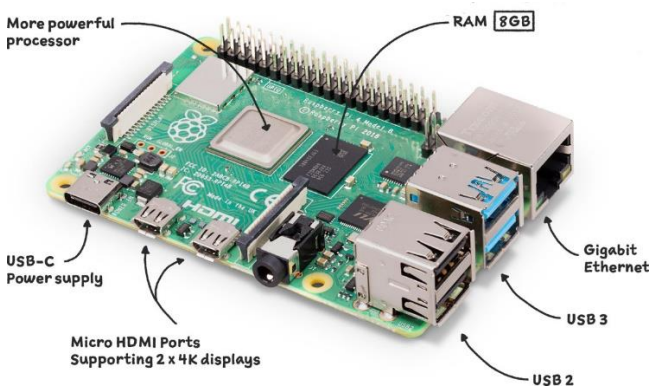
จัดเก็บ และการควบคุมการเล่นสื่อที่ยืดหยุ่น เช่น การปรับความเร็วในการเล่น, การเลือกเสียงพากย์หรือซับไตเติลที่ต้องการ, การใช้ฟิลเตอร์ภาพและเสียง, การครอบตัด (Crop), การซูมภาพ และการหมุนวิดีโอ ในบริบทเฉพาะของ “การแสดงผลย้อน” หรือ “การเล่นย้อนกลับ” (Reverse Playback) VLC มีความสามารถในการเล่นวิดีโอย้อนจากท้ายไปต้น ซึ่งเป็นคุณสมบัติที่มีประโยชน์ในงานวิเคราะห์วิดีโอ เช่น การตรวจสอบเหตุการณ์ย้อนหลัง, การวิเคราะห์การเคลื่อนไหวของวัตถุ หรือใช้ในการพัฒนาระบบประมวลผลภาพ โดยแม้ว่าฟีเจอร์นี้จะยังไม่สามารถเปิดใช้งานได้โดยตรงจากอินเทอร์เฟซแบบกราฟิก (GUI) ทั่วไปของโปรแกรม แต่ผู้ใช้สามารถเรียกใช้งานได้ผ่านวิธีการเฉพาะ เช่น การใช้คำสั่งใน Command Line, การใช้สคริปต์ หรือการตั้งค่าระดับลึกในตัว

2.3 อุปกรณ์ที่ใช้ในโครงงาน

2.3.1 Raspberry Pi 4

บอร์ดคอมพิวเตอร์ขนาดเล็ก (Single Board Computer) ที่มีขนาดเล็กเท่าบัตรเครดิต แต่มีความสามารถในการทำงานคล้ายคอมพิวเตอร์ทั่วไป เหมาะสำหรับผู้ที่ต้องการเรียนรู้การเขียนโปรแกรม, ทำงานกับอุปกรณ์อิเล็กทรอนิกส์, หรือพัฒนาโปรเจคต่างๆ ที่ต้องการความสามารถในการประมวลผล

Raspberry Pi 4 เป็นบอร์ดคอมพิวเตอร์ขนาดเล็กที่มีประสิทธิภาพสูง มาพร้อมหน่วยประมวลผลแบบ Quad-core ARM Cortex-A72 ความเร็ว 1.5GHz และมีหน่วยความจำให้เลือกหลายขนาด ได้แก่ 2GB, 4GB และ 8GB LPDDR4 รองรับการเชื่อมต่อผ่าน Wi-Fi, Bluetooth 5.0 และ Gigabit Ethernet มีพอร์ต USB 2.0 และ USB 3.0 สำหรับเชื่อมต่ออุปกรณ์เสริม รวมถึงพอร์ต Micro HDMI จำนวน 2 ช่อง รองรับการแสดงผลความละเอียดสูงสุดระดับ 4K นอกจากนี้ยังมีช่องเชื่อมต่อ GPIO จำนวน 40 ขา สำหรับควบคุมอุปกรณ์อิเล็กทรอนิกส์ภายนอก เหมาะสำหรับการพัฒนาโครงการ IoT ระบบควบคุมอัตโนมัติ และการเรียนรู้ด้านเทคโนโลยีสารสนเทศและการเขียนโปรแกรม ระบบใช้ไฟเลี้ยงผ่านพอร์ต USB-C ขนาด 5V/3A และรองรับการติดตั้งระบบปฏิบัติการผ่าน microSD card



รูปที่ 2-4 Raspberry Pi 4

(ที่มา: https://inex.co.th/home/wp-content)

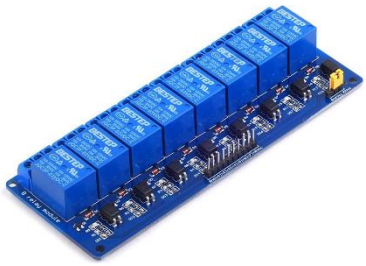
คุณสมบัติทางเทคนิคของ Raspberry Pi 4

1. หน่วยประมวลผล: Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit 1.5GHz
2. หน่วยความจำ RAM: 2GB, 4GB, หรือ 8GB LPDDR4 (ขึ้นอยู่กับรุ่น)
3. หน่วยความจำเก็บข้อมูล: ใช้ microSD card เป็นระบบปฏิบัติการและพื้นที่เก็บข้อมูล
4. พอร์ต USB: 2x USB 3.0 ,2x USB 2.0
5. พอร์ตวิดีโอ: 2x Micro HDMI รองรับวิดีโอ 4K
6. พอร์ตเครือข่าย: Gigabit Ethernet
7. Wi-Fi: 802.11ac (2.4GHz และ 5GHz)
8. Bluetooth: 5.0
9. GPIO (General Purpose Input Output): 40 ขา สำหรับเชื่อมต่อเซ็นเซอร์ รีเลย์ และอุปกรณ์อิเล็กทรอนิกส์
10. กล้องและจอภาพ: รองรับ CSI (Camera Serial Interface) และ DSI (Display Serial Interface)

2.3.2 Relay Module

รีเลย์ (Relay) เป็นอุปกรณ์อิเล็กทรอนิกส์ชนิดหนึ่งที่ทำหน้าที่เป็นสวิตช์ควบคุมการเปิด-ปิดวงจรไฟฟ้า โดยอาศัยหลักการทำงานของแม่เหล็กไฟฟ้า เมื่อจ่ายกระแสไฟฟ้าเข้าไปที่ขดลวดของรีเลย์ จะเกิดสนามแม่เหล็กดึงหน้าสัมผัสให้เปลี่ยนสถานะ ทำให้สามารถควบคุมวงจรไฟฟ้าอีกชุดหนึ่งได้โดยไม่ต้องเชื่อมต่อทางตรง

รีเลย์มีหลายประเภท เช่น รีเลย์แบบ 1 ช่อง (1 Channel), 2 ช่อง, 4 ช่อง และ 8 ช่อง โดยสามารถเลือกใช้ตามจำนวนอุปกรณ์ที่ต้องการควบคุม



รูปที่ 2-5 Relay Module

(ที่มา: https://inex.co.th/home/wp-content)

2.3.3 กล้อง USB (USB Camera)

กล้อง USB หรือ USB Camera เป็นอุปกรณ์รับภาพที่สามารถเชื่อมต่อกับคอมพิวเตอร์หรือบอร์ดควบคุม เช่น Raspberry Pi ผ่านพอร์ต USB โดยไม่ต้องใช้ไดรเวอร์เพิ่มเติมในระบบปฏิบัติการที่รองรับ กล้องชนิดนี้สามารถใช้งานสำหรับถ่ายภาพนิ่งและวิดีโอในความละเอียดที่หลากหลาย

กล้อง USB ทำงานโดยส่งข้อมูลภาพผ่านพอร์ต USB ไปยังระบบประมวลผล ซึ่งสามารถนำไปแสดงผลแบบเรียลไทม์หรือใช้ร่วมกับระบบวิเคราะห์ภาพ เช่น การตรวจจับวัตถุ การนับจำนวน หรือการเฝ้าระวังความปลอดภัย

อุปกรณ์ชนิดนี้เหมาะสำหรับใช้งานในโครงงานด้าน IoT, ระบบกล้องวงจรปิด, การประมวลผลภาพ และระบบอัตโนมัติ เนื่องจากมีราคาย่อมเยา ติดตั้งง่าย และสามารถใช้งานร่วมกับซอฟต์แวร์หลายประเภท เช่น OpenCV หรือ Motion

รูปที่ 2-6 USB Camera

(ที่มา: https://inwfile.com/s-dx/odsmvk.png)

2.3.4 พาวเวอร์ปลั๊ก

พาวเวอร์ปลั๊ก หรือปลั๊กไฟฟ้า เป็นอุปกรณ์ที่ใช้สำหรับเชื่อมต่อเครื่องใช้ไฟฟ้ากับแหล่งจ่ายไฟ โดยทั่วไปจะมีสายไฟพร้อมปลั๊กเสียบที่ปลายด้านหนึ่ง และเต้ารับ (Socket) หรือช่องเสียบอุปกรณ์ไฟฟ้าอีกด้านหนึ่ง พาวเวอร์ปลั๊กมีทั้งแบบเต้ารับเดี่ยว และแบบพ่วงหลายช่อง พร้อมด้วยสวิตช์ควบคุมการเปิด-ปิดแต่ละช่อง

วัสดุที่ใช้ผลิตพาวเวอร์ปลั๊กมักเป็นพลาสติกทนความร้อน และมีฉนวนหุ้มเพื่อความปลอดภัย บางรุ่นมีระบบป้องกันไฟกระชากหรือฟิวส์นิรภัยในตัว เพื่อป้องกันความเสียหายต่ออุปกรณ์ที่เชื่อมต่อ พาวเวอร์ปลั๊กมีความจำเป็นอย่างยิ่งในการติดตั้งระบบไฟฟ้าในโครงงานต่าง ๆ โดยเฉพาะเมื่อมีการใช้หลายอุปกรณ์ร่วมกัน เช่น กล้อง, รีเลย์, หลอดไฟ หรืออุปกรณ์ IoT

รูปที่ 2-7 พาวเวอร์ปลั๊ก

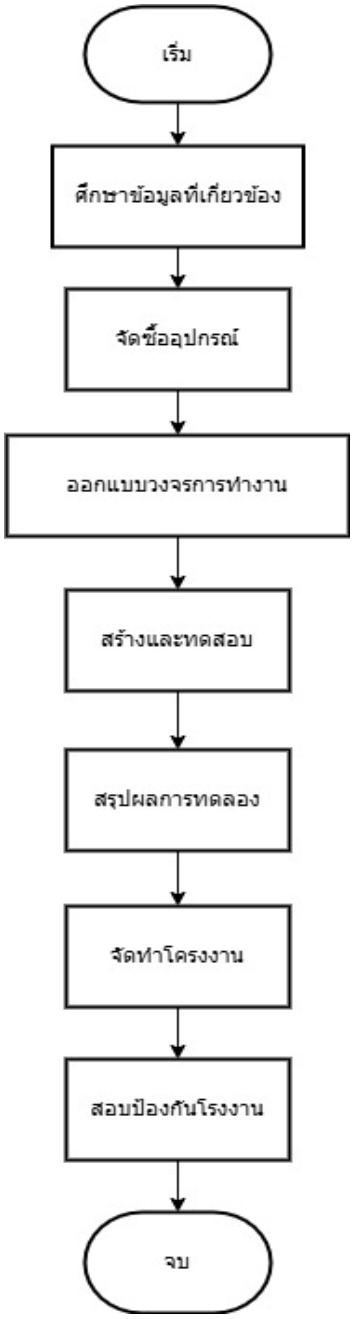
(ที่มา: https://inwfile.com/s-dx/odsmvk.png)

บทที่3
วิธีการดำเนินงาน

ในบทนี้จะกล่าวถึงกระบวนการวางแผน ออกแบบ และดำเนินการพัฒนาโครงงานระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านเทคโนโลยี IoT ซึ่งรวมถึงการออกแบบระบบ การจัดหาวัสดุอุปกรณ์ การเขียนโปรแกรม และการทดสอบระบบ เพื่อให้ระบบสามารถทำงานได้อย่างมีประสิทธิภาพ ตอบสนองต่อความต้องการของผู้ใช้งาน และสามารถนำไปประยุกต์ใช้ได้จริง

โดยการทำโครงงานครั้งนี้มีวัตถุประสงค์เพื่อพัฒนา ระบบตรวจนับวัวเข้าและออกคอกพร้อมการควบคุมอุปกรณ์ผ่านเว็บแอปพลิเคชันแบบเรียลไทม์ โดยใช้บอร์ด Raspberry Pi เป็นหน่วยประมวลผลหลัก ซึ่งสามารถเชื่อมต่อกับกล้องหลายตัวเพื่อรับภาพจากพื้นที่จริง แล้วนำมาประมวลผลด้วยเทคนิคการตรวจจับวัตถุด้วยโมเดลปัญญาประดิษฐ์ (Artificial Intelligence: AI) เพื่อระบุและนับจำนวนวัว โดยใช้เทคนิคการตรวจจับการเคลื่อนที่ผ่านเส้น (Line Crossing Detection) นอกจากนี้ ระบบยังสามารถควบคุมรีเลย์ไฟฟ้าแบบตั้งเวลาและตอบสนองต่อเงื่อนไขที่ผู้ใช้กำหนดได้ โดยมีการแสดงข้อมูลและแจ้งเตือนผ่านหน้าเว็บแอปพลิเคชันแบบทันที (real-time web interface)

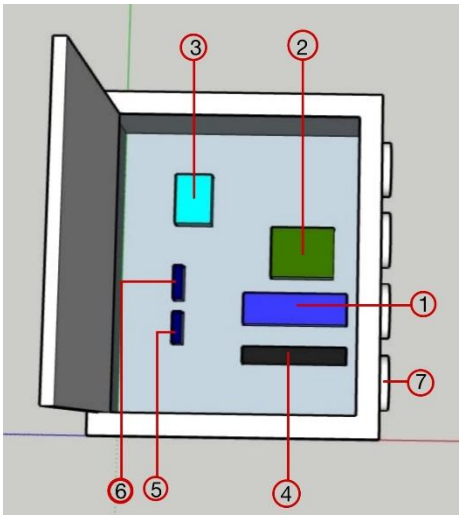
การดำเนินโครงงานสามารถเขียนแผนผังขั้นตอนการดำเนินงานได้แสดงดังภาพที่ 3-1



รูปที่ 3-1 แผนผังขั้นตอนการดำเนินโครงงาน

3.1 การออกแบบชุดระบบควบคุมสมาร์ทฟาร์ม

3.1.1 ออกแบบตู้ควบคุมด้วย Raspberry pi ขนาด 25 x 35 เซนติเมตร ความหนา 15 เซนติเมตร ดังภาพที่ 3-2

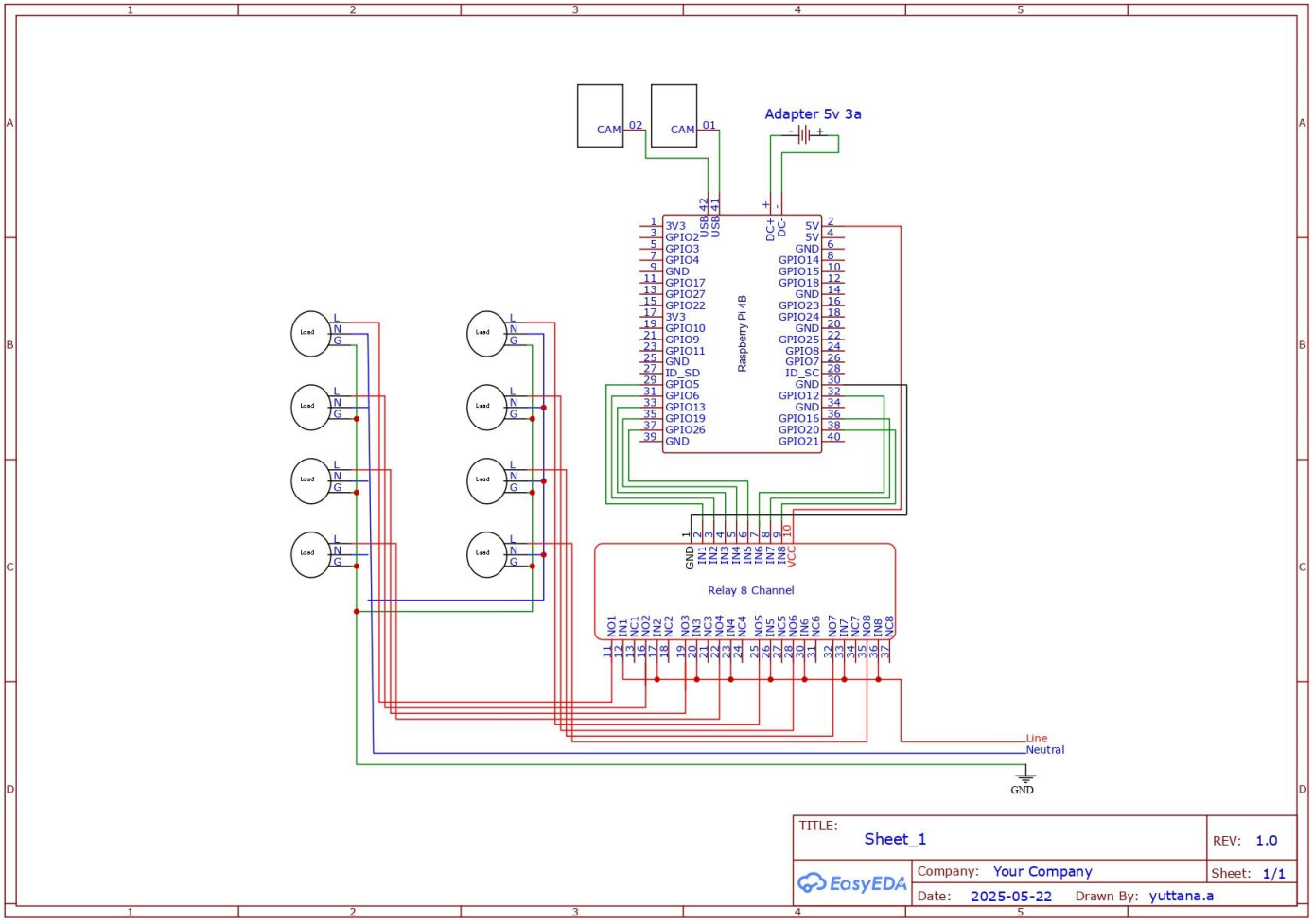


รูปที่ 3-2 แบบจำลองตู้ควบคุม

ตารางที่ 3-1 อุปกรณ์ภายในตู้ควบคุม

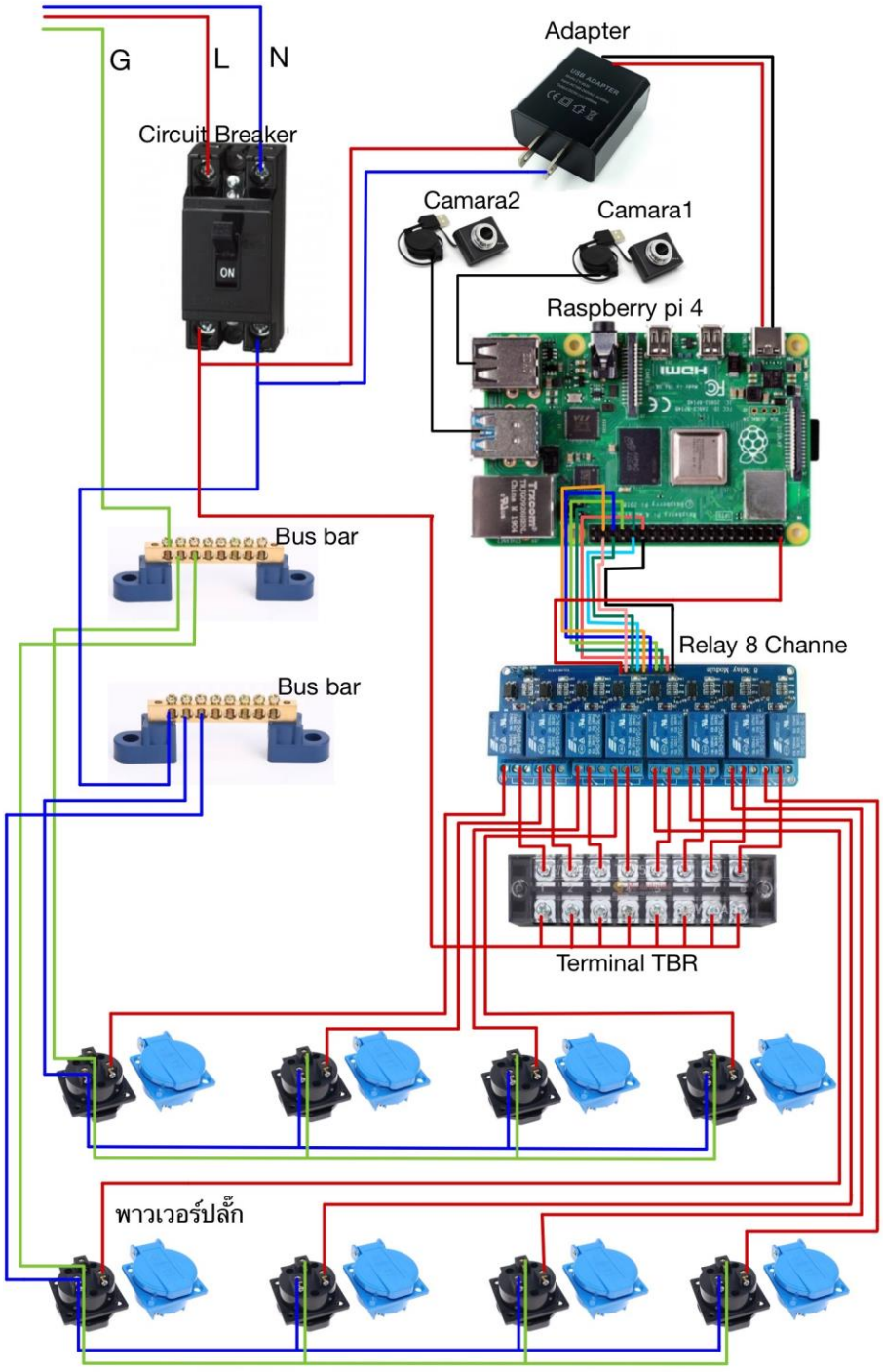
หมายเลข	ชื่ออุปกรณ์
1	Relay 8 Channel Relay 5V relay Active High / LOW Relay Module Shield 250V/10A
2	Raspberry Pi 4 Model B 4GB RAM
3	Circuit breaker 220v 30a NANO
4	Terminal TBR-10 600V 10A 8ช่อง
5	Bus barขนาด 9x6
6	Bus barขนาด 9x6
7	Sumoglobal พาวเวอร์ปลั๊ก 2สาย 3ขา 16A (ตัวเมีย) รุ่น P1-313U SUMO Type Thai

3.1.2 วงจรการเชื่อมต่ออุปกรณ์ในระบบ



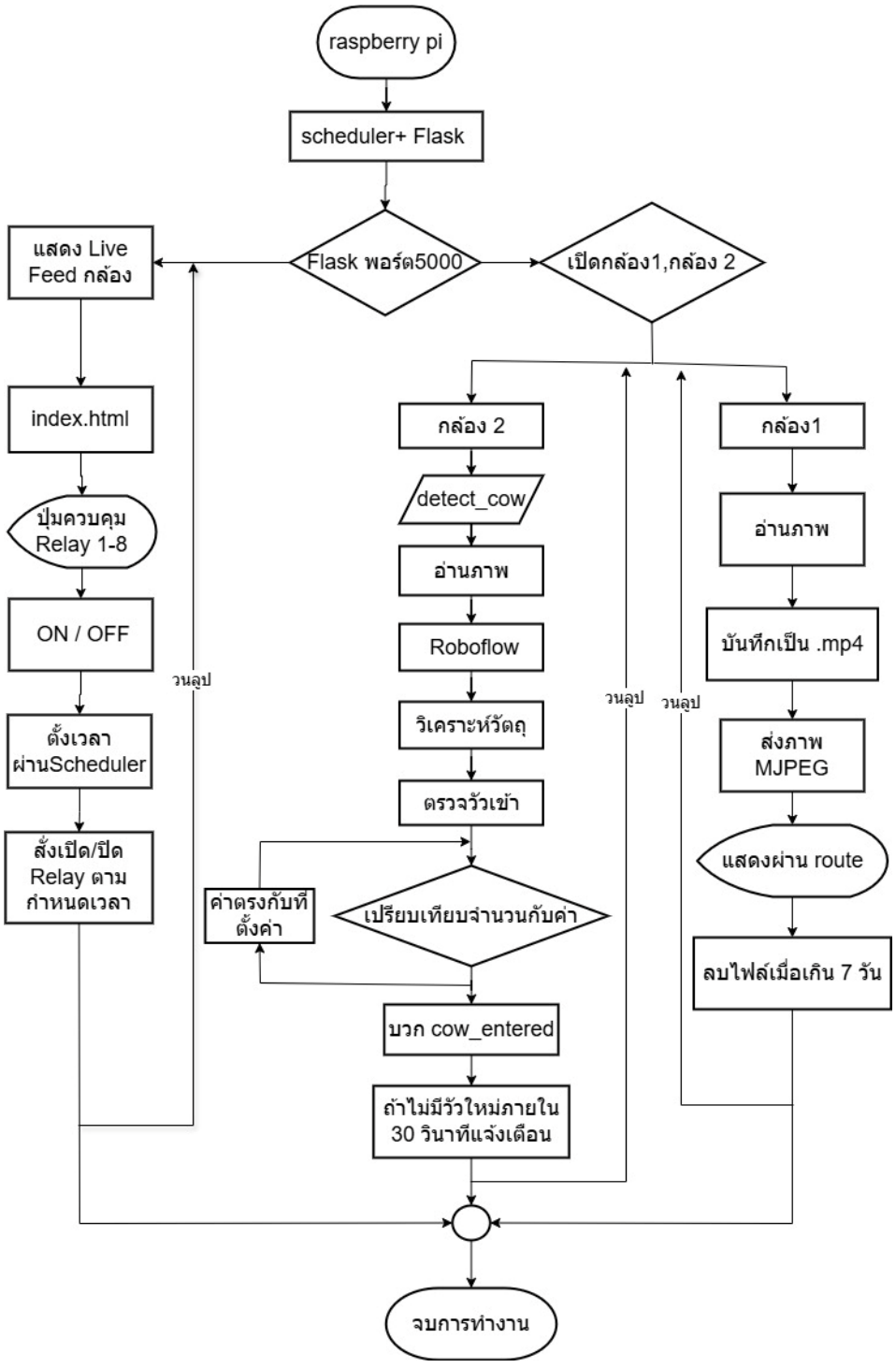
รูปที่ 3-3 วงจรการเชื่อมต่ออุปกรณ์ในระบบ

3.1.3 วงจรการต่อใช้งานอุปกรณ์จริง



รูปที่ 3-4 แบบจำลองตู้ควบคุม

3.2 flowchart แสดงการทำงานของโปรแกรม



รูปที่ 3-5 flowchart แสดงการทำงานของโปรแกรม

ระบบนี้ถูกออกแบบให้ทำงานบนบอร์ด Raspberry Pi โดยมีวัตถุประสงค์หลักในการควบคุมการเปิด หรือ ปิดรีเลย์ และตรวจนับจำนวนวัวที่ผ่านเข้ามาภายในฟาร์ม โดยอาศัยกล้องสองตัวและเทคโนโลยีวิเคราะห์ภาพผ่าน Roboflow ซึ่งทั้งหมดนี้ควบคุมผ่านเว็บแอปที่เขียนด้วย Flask และสามารถตั้งเวลาเปิด หรือ ปิดอุปกรณ์ได้ด้วย Scheduler

เมื่อเริ่มต้นระบบ Raspberry Pi จะโหลดโมดูลต่าง ๆ ที่จำเป็น Flask, Scheduler, โมดูลควบคุมกล้อง และโมดูลควบคุมรีเลย์ จากนั้น Flask จะเริ่มทำงานที่พอร์ต 5000 และเปิดให้ผู้ใช้เข้าถึงผ่านเว็บเบราว์เซอร์ กล้องทั้งสองตัวจะถูกเปิดใช้งานโดยที่กล้องตัวที่ 1 ทำหน้าที่แสดงภาพสด (Live Feed) พร้อมกับบันทึกวิดีโอในรูปแบบ .mp4 ส่วนกล้องตัวที่ 2 ใช้สำหรับตรวจนับวัวโดยทำงานร่วมกับ Roboflow เพื่อวิเคราะห์ภาพและแยกแยะวัตถุในส่วนของกล้องที่ 2 จะมีเธรดเฉพาะที่ทำงานชื่อว่า detect_cow ซึ่งทำงานแบบวนลูปตลอดเวลา โดยจะอ่านภาพจากกล้องทุก ๆ ระยะและส่งภาพที่ผ่านการปรับขนาดเข้าไปยัง Roboflow เพื่อให้ระบบช่วยวิเคราะห์วัตถุในภาพ จากนั้นจะตรวจจับตำแหน่งของวัวที่ปรากฏขึ้น หากพบว่ามีวัวเดินผ่านเข้ามาในพื้นที่ตรวจจับ ระบบจะบันทึกไว้ในตัวแปร cow_entered และบันทึกเวลาที่วัวเข้าครั้งล่าสุดระบบจะเปรียบเทียบจำนวนวัวที่นับได้กับค่าเกณฑ์ที่กำหนดไว้ในไฟล์ cow_thresholds.json หากจำนวนวัวน้อยกว่าค่าที่ตั้งไว้ และไม่มีวัวใหม่ภายในระยะเวลา 30 วินาที ระบบจะทำการแจ้งเตือนผ่านหน้าเว็บโดยแสดงข้อความเตือนเพื่อให้ผู้ดูแลระบบทราบว่ามีความผิดปกติเกิดขึ้นในระบบนับวัวขณะเดียวกัน กล้องที่ 1 จะทำการบันทึกวิดีโอลงไฟล์ .mp4 และแสดงภาพแบบ MJPEG ผ่านหน้าเว็บ โดยผู้ใช้สามารถดูภาพสดจากกล้องทั้งสองผ่านหน้าเว็บที่เสิร์ฟจาก Flask (index.html) ภายในหน้านี้ยังมีปุ่มควบคุมรีเลย์ 1–8 ที่สามารถเปิด หรือ ปิดได้แบบ Manual และยังสามารถตั้งเวลาให้เปิดหรือปิดรีเลย์แบบอัตโนมัติได้ผ่าน Scheduler ซึ่งทำงานโดยตรวจสอบเวลาทุกนาที หากเวลาปัจจุบันตรงกับเวลาที่ตั้งไว้ ระบบจะสั่งให้เปิดหรือปิดรีเลย์ตามกำหนดโดยอัตโนมัติผู้ใช้สามารถดูการแจ้งเตือนผ่านหน้าจอเว็บได้แบบเรียลไทม์ โดยจะมีฟังก์ชันลบไฟล์วิดีโออัตโนมัติเมื่อไฟล์นั้นมีอายุเกินกว่า 7 วัน เพื่อป้องกันไม่ให้พื้นที่จัดเก็บเต็ม ระบบทั้งหมดนี้ทำงานแบบวนลูปอย่างต่อเนื่อง

3.3 การคำนวณหาค่าจากการวัด

3.3.1 ข้อผิดพลาดร้อยละ

ข้อผิดพลาดร้อยละ (Percentage Error) หรือข้อผิดพลาดเปอร์เซ็นต์ หมายถึง ความแตกต่างระหว่างค่าที่ประมาณหรือค่าที่วัดได้กับค่าที่แน่นอนหรือค่าที่ทราบ โดยแสดงเป็นเปอร์เซ็นต์ ข้อผิดพลาดร้อยละมีความสำคัญในการประเมินความแม่นยำ และความถูกต้องของการวัดในงานวิจัยทางวิทยาศาสตร์ และยังเป็นเครื่องมือสำคัญในการควบคุมคุณภาพในกระบวนการผลิต โดยการเบี่ยงเบนจากค่าที่คาดหวังอาจชี้ให้เห็นถึงข้อบกพร่องที่อาจเกิดขึ้นในกระบวนการผลิตหรือการทดลองต่าง ๆ ได้

สูตรการคำนวณข้อผิดพลาดเปอร์เซ็นต์มีดังนี้

$$\text{Relative error} = \left| \frac{X_{mea} - X_t}{X_t} \right|$$

$$\text{เปอร์เซ็นต์ Error} = \text{Relative error} \times 100$$

โดย X_t คือค่าจริง *(*True value*)*

X_{mea} คือค่าที่ได้จากการวัด (Measure value)

3.3.2 ค่าเฉลี่ย

การหาค่าเฉลี่ยเป็นกระบวนการทางสถิติที่ใช้ในการคำนวณค่ากลางของชุดข้อมูล โดยนำผลรวมของข้อมูลทั้งหมดมาหารด้วยจำนวนข้อมูลที่มีอยู่ เพื่อให้ได้ค่าที่สะท้อนแนวโน้มของข้อมูลในภาพรวม

ข้อมูลไม่แจกแจงความถี่จะมีลักษณะเป็นตัวๆ คือ $X_1+X_2+...+X_n$ดังนั้น ค่าเฉลี่ยเลขคณิต คือ

$$\bar{X} = \frac{X_1+X_2+...+X_n}{n}$$

โดย $\bar{X}$ คือค่าเฉลี่ย (เอ็กซ์บาร์)

$X_1+X_2+...+X_n$ คือจำนวนข้อมูลทั้งหมดรวมกัน

n คือจำนวนของข้อมูล

3.3.3 ทฤษฎีบทพีทาโกรัส (Pythagorean Theorem)

ทฤษฎีบทพีทาโกรัสใช้ในกรณีของสามเหลี่ยมมุมฉาก โดยระบุว่า ผลรวมของกำลังสองของด้านประกอบมุมฉากทั้งสอง เท่ากับกำลังสองของด้านตรงข้ามมุมฉาก (ด้านตรงข้ามคือตัวเส้นทแยง หรือ “ระยะเฉียง” จากกล้องไปยังจุดตรวจจับ)

สูตรการคำนวณหาสามเหลี่ยมมุมฉากมีดังนี้

$$L = \sqrt{H^2 + D^2}$$

โดย L คือระยะเฉียงจากกล้องถึงพื้นหรือวัตถุที่ต้องการตรวจจับ

H คือความสูงของกล้องจากพื้น

D คือระยะห่างตามแนวราบจากตำแหน่งกล้องถึงตำแหน่งวัตถุ

3.3.4 ฟังก์ชันตรีโกณมิติ (Trigonometry)

ฟังก์ชันตรีโกณมิติ เป็นฟังก์ชันทางคณิตศาสตร์ที่เกี่ยวข้องกับมุมและด้านของสามเหลี่ยม โดยเฉพาะสามเหลี่ยมมุมฉาก โดยการคำนวณใช้หลักตรีโกณมิติจากความสูงของกล้องและระยะในแนวราบ เพื่อควบคุมให้ภาพครอบคลุมจุดที่วัตถุจะผ่าน

สูตรการคำนวณฟังก์ชันตรีโกณมิติมีดังนี้

$$\theta = tan^{-1}\left(\frac{H}{D}\right)$$

โดย θ คือมุมก้มของกล้องจากแนวระนาบ

H คือความสูงของกล้องจากพื้น

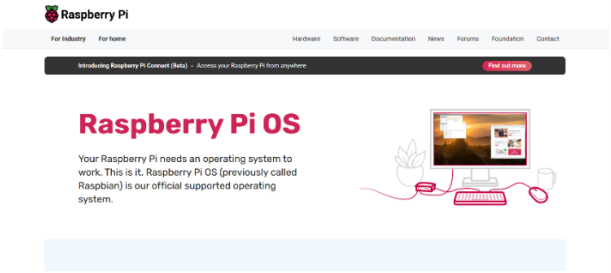
D คือระยะห่างแนวนอนจากกล้องถึงวัตถุเป้าหมาย

3.4 วิธีการติดตั้งและเตรียมระบบปฏิบัติการ

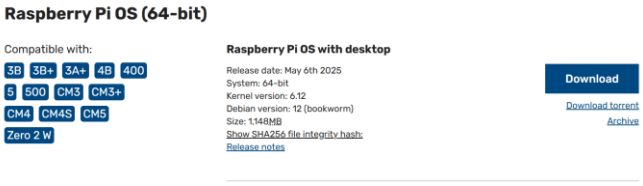
3.4.1. การเลือกระบบปฏิบัติการ ระบบปฏิบัติการที่ใช้ในโครงงานนี้คือ Raspberry Pi OS เวอร์ชัน 64-bit แบบ Lite ซึ่งเป็นระบบปฏิบัติการอย่างเป็นทางการที่พัฒนาโดย Raspberry Pi Foundation โดยมีพื้นฐานจาก Debian Linux

3.4.2 การติดตั้งระบบปฏิบัติการ เริ่มจากการดาวน์โหลดไฟล์ Image ของ Raspberry Pi OS Lite (64-bit) จากเว็บไซต์ทางการของ Raspberry Pi จากนั้นใช้โปรแกรม Raspberry Pi Imager ซึ่งเป็นเครื่องมือที่พัฒนาโดย Raspberry Pi Foundation เช่นกัน ในการเขียนระบบปฏิบัติการลงบน microSD Card โดยเลือกไฟล์ Image ที่ดาวน์โหลดมา และเลือก microSD Card ที่ต้องการใช้งาน

หลังจากเขียนไฟล์ระบบปฏิบัติการเรียบร้อยแล้ว มีการเตรียมไฟล์เพิ่มเติมเพื่อให้ Raspberry Pi สามารถใช้งานได้อย่างสะดวก ได้แก่ การสร้างไฟล์ชื่อ ssh เพื่อเปิดใช้งานการเชื่อมต่อ SSH โดยอัตโนมัติ โดยไฟล์ทั้งหมดจะถูกวางไว้ในพาร์ทิชัน boot ของ microSD Card ก่อนที่จะนำไปเสียบใช้งานกับบอร์ด Raspberry Pi เมื่อเปิดเครื่องและบูตเข้าสู่ระบบปฏิบัติการสำเร็จแล้ว สามารถเชื่อมต่อเข้าสู่ Raspberry Pi ผ่านโปรแกรม PuTTY เพื่อเริ่มต้นกระบวนการตั้งค่าระบบต่อไป



รูปที่ 3-6 www.raspberrypi.com



รูปที่ 3-7 Raspberry Pi OS

รูปที่ 3-8 หน้าจอ Desktop Raspberry Pi

3.5 การสร้างเว็บแอปพลิเคชัน

3.5.1 การสร้าง Backend ด้วย Flask โดยในส่วนของ Backend ของเว็บแอป ใช้ภาษา Python ร่วมกับเฟรมเวิร์ก Flask เป็นตัวกลางในการจัดการคำร้องขอ (HTTP Request) และตอบกลับ (HTTP Response) รวมถึงควบคุมการประมวลผลทางด้านตรรกะและการติดต่อกับฮาร์ดแวร์ โดย Flask ทำหน้าที่ดังนี้

3.5.1.1 ควบคุมรีเลย์แบบเรียลไทม์ โดยใช้เส้นทาง (Route) ที่สามารถรับคำสั่งจากผู้ใช้เพื่อเปิดหรือปิดรีเลย์แต่ละช่องผ่าน Web Interface และส่งคำสั่งควบคุมไปยัง GPIO ของ Raspberry Pi โดยใช้ไลบรารี RPi.GPIO

3.5.1.2 ตั้งเวลาโดยกำหนดช่วงเวลาที่ต้องการให้รีเลย์แต่ละตัวเปิดหรือปิดได้ผ่านหน้าเว็บ โดย Flask จะจัดเก็บข้อมูลเหล่านี้ไว้ในไฟล์ JSON และมี Scheduler ทำหน้าที่เช็คเวลาและสั่งงานรีเลย์อัตโนมัติ

3.5.1.3 การเชื่อมต่อกับกล้อง Flask โดยจะประสานงานกับ OpenCV และ FFmpeg เพื่อเชื่อมต่อกับกล้องที่เชื่อมต่อกับ Raspberry Pi ทั้ง 2 ตัว และทำการประมวลผลภาพแบบเรียลไทม์ รวมถึงใช้ pipe ของ FFmpeg เพื่อลด latency ในการส่งภาพสดขึ้นหน้าเว็บ

3.5.1.4 ระบบนับจำนวนวัว ใช้โมเดล YOLOv5 ที่รันผ่าน Roboflow เพื่อตรวจจับวัตถุในเฟรมภาพจากกล้องตัวที่ 2 เมื่อวัตถุเคลื่อนที่ผ่านเส้นที่กำหนด ระบบจะตรวจจับและนับจำนวนวัวที่เข้า-ออก และนำค่าดังกล่าวมาแสดงบนหน้าเว็บแบบเรียลไทม์

3.5.1.5 ระบบแจ้งเตือน เมื่อจำนวนวัวที่นับได้ต่ำกว่าหรือเกินกว่าค่าที่กำหนดไว้ ระบบจะส่งข้อมูลแจ้งเตือนแบบ Pop-up Notification ไปยังหน้าเว็บหลังจากนั้นเป็นเวลา 30 วินาที

3.5.2 การสร้าง Frontend ด้วย HTML, CSS และ JavaScript ส่วนของ Frontend ถูกพัฒนาด้วยภาษา HTML สำหรับโครงสร้างหน้าเว็บ, CSS สำหรับการจัดรูปแบบการแสดงผล และ JavaScript สำหรับการโต้ตอบกับผู้ใช้และการอัปเดตข้อมูลแบบ Real-time

3.5.2.1 แผงควบคุม (Dashboard) มีการจัดวางองค์ประกอบต่าง ๆ เช่น ปุ่มควบคุมรีเลย์, ตารางแสดงเวลาเปิด และปิด การตั้งค่าค่าจำนวนวัวขั้นต่ำ หรือสูงสุด การแสดงจำนวนวัวที่นับได้ และภาพจากกล้องแบบสด

3.5.2.2 แสดงภาพกล้องแบบเรียลไทม์ ใช้ <img> tag ที่เชื่อมต่อกับ stream ของ Flask ผ่าน URL เฉพาะแต่ละกล้อง (เช่น /video_feed_camera1, /video_feed_camera2) ซึ่งภาพจะอัปเดตต่อเนื่องโดยใช้ MJPEG

3.5.2.3 การควบคุมแบบ Ajax เมื่อผู้ใช้คลิกเปิดหรือปิดรีเลย์ หรือเปลี่ยนค่าการตั้งเวลา ระบบจะส่งคำสั่งไปยัง Flask Backend โดยไม่ต้องโหลดหน้าใหม่ ผ่าน JavaScript และการใช้งาน AJAX

3.5.2.4 การแสดงผลจำนวนวัวและแจ้งเตือน ใช้ JavaScript ร่วมกับ WebSocket หรือการเช็คค่าผ่าน AJAX ทุก ๆ วินาที เพื่อแสดงจำนวนวัวเข้า/ออกแบบเรียลไทม์ และแจ้งเตือนทันทีหากเกินหรือต่ำกว่าค่าที่กำหนด

3.5.3 การจัดการโค้ดและความปลอดภัย โครงสร้างโค้ดของเว็บแอปพลิเคชันถูกจัดแยกเป็นไฟล์ และในด้านความปลอดภัย ได้มีการจำกัดการเข้าถึงเฉพาะ IP ที่อยู่ในเครือข่ายภายใน

3.5.3.1 app.py เป็นไฟล์หลักที่ทำหน้าที่รวมระบบทั้งหมด ทั้งควบคุมกล้อง รีเลย์ การประมวลผลภาพ และการให้บริการหน้าเว็บ

3.5.3.2 โฟลเดอร์ templates/ สำหรับเก็บไฟล์ HTML

3.5.3.3 โฟลเดอร์ static/ สำหรับเก็บไฟล์ CSS, JS และภาพประกอบ

3.5.3.4 ไฟล์ config.json หรือฐานข้อมูล SQLite สำหรับเก็บค่าการตั้งค่าจากผู้ใช้

3.6.4 การทดสอบระบบและการใช้งานจริง ได้มีการติดตั้งและรันบน Raspberry Pi โดยใช้คำสั่ง flask run --host=0.0.0.0 เพื่อให้สามารถเข้าถึงจากอุปกรณ์อื่น ๆ ภายในเครือข่าย เช่น คอมพิวเตอร์ โทรศัพท์มือถือ หรือแท็บเล็ต ระบบได้รับการทดสอบในสถานการณ์จริง โดยทดสอบการควบคุมรีเลย์แบบเรียลไทม์ การตั้งเวลา การแสดงภาพกล้องสด การนับจำนวนวัว และการแจ้งเตือน พบว่าทุกฟังก์ชันทำงานได้อย่างถูกต้อง และมีประสิทธิภาพ

3.6 การตรวจนับจำนวนวัว

3.6.1 การฝึกสอนโมเดล ใช้โมเดล YOLOv5 ที่ผ่านการฝึกสอน (training) โดยใช้ dataset ของวัวในฟาร์มซึ่งประกอบด้วยภาพวัวในหลากหลายมุมมองและสภาวะแวดล้อม เพื่อเพิ่มความแม่นยำในการตรวจจับในสภาพแวดล้อมจริง โดยใช้บริการ Roboflow สำหรับการจัดการ dataset และการฝึกสอนโมเดลผ่าน cloud

3.6.1.1 ทำการเก็บภาพจากกล้องในคอกวัว

3.6.1.2 ทำการ annotate (ตีกรอบวัตถุ) โดยกำหนด label ว่าเป็น “yolo-cow”

3.6.1.3 อัปโหลด dataset เข้าสู่ Roboflow และตั้งค่า pre-processing โดยปรับขนาดภาพเป็น 512x512 พิกเซล

3.6.1.4 ฝึกสอนโมเดล YOLOv5 ผ่าน Roboflow โดยกำหนดค่าความเชื่อมั่น (confidence threshold) ไว้ที่ 30

3.6.1.5 จากนั้นทำการ export โมเดล สำหรับใช้งานร่วมกับไลบรารี PyTorch

3.6.2 การนำโมเดลมาใช้งาน โดยจะนำโมเดลที่ฝึกเสร็จแล้วมาใช้ร่วมกับกล้องตัวที่ 2 ซึ่งติดตั้งไว้ที่ทางเข้า/ออกของคอกวัว โดยภาพที่ได้จากกล้องจะถูกปรับขนาดเป็น 512x512 และส่งเข้าระบบตรวจจับของ YOLOv5 และระบบจะทำการตรวจจับ bounding box ที่ครอบวัว พร้อมทั้งระบุตำแหน่งพิกัด (x, y) ของวัตถุแต่ละตัว เพื่อนำไปวิเคราะห์พฤติกรรมการเดินผ่านเส้น (line crossing) ว่ามีวัวเดินเข้า หรือออกจากคอกหรือไม่ โดยหลักการคือ หากพิกัด y ของจุดศูนย์กลางวัตถุในเฟรมก่อนหน้า (prev_centroids) อยู่เหนือเส้นที่กำหนดไว้ (line_y) และในเฟรมปัจจุบันจุดเดียวกันเคลื่อนลงมาต่ำกว่าเส้น แสดงว่าวัวได้ “เดินผ่านเข้า” คอก ซึ่งระบบจะทำการเพิ่มตัวแปร cow_entered

3.7 หลักการทำงานของกล้อง

3.7.1 กล้องตัวที่ 1 ใช้สำหรับแสดงภาพ Live Feed ผ่านหน้าเว็บ และบันทึกวิดีโอย้อนหลัง โดยจะทำหน้าที่จับภาพวิดีโอแบบสด (Live Video) ซึ่งจะใช้ไลบรารี OpenCV ในการรับภาพจากกล้องและนำมาประมวลผลทีละเฟรม แล้วแปลงภาพให้เป็นรูปแบบ MJPEG เพื่อส่งไปยังหน้าเว็บผ่าน Flask Web Server ทำให้สามารถเปิดหน้าเว็บเพื่อดูภาพสดจากกล้องได้ตลอดเวลา และระบบจะทำการบันทึกวิดีโอลงในโฟลเดอร์ที่ชื่อว่า recordings โดยระบบจะแยกไฟล์วิดีโอตามวัน เช่น “12-05-2025.mp4” และจะมีฟังก์ชันอัตโนมัติในการลบวิดีโอที่มีอายุเกิน 7 วันเพื่อประหยัดพื้นที่บน SD Card ของ Raspberry Pi

3.7.2 กล้องตัวที่ 2 ทำการติดตั้งไว้ที่ทางเข้าและออกของคอกวัว ทำงานโดยระบบจะใช้โมเดล YOLOv5 ที่รันในเครื่อง ตรวจจับวัวในแต่ละเฟรม ซึ่งจะวิเคราะห์ภาพด้วย YOLO และ Roboflow เมื่อได้ภาพจากกล้องในแต่ละเฟรม ระบบจะส่งภาพนั้นไปยัง Roboflow API ซึ่งทำหน้าที่ใช้โมเดล YOLO ที่ผ่านการฝึกฝนมาแล้วสำหรับตรวจจับวัวโดยเฉพาะ Roboflow จะวิเคราะห์ภาพและส่งข้อมูลกลับมาในรูปแบบ JSON โดยมีการกำหนดเส้นเสมือน (Virtual Line) บนเฟรมวิดีโอ เช่น เส้นแนวนอนตรงประตูคอก และระบบจะตรวจสอบตำแหน่งจุดกึ่งกลางของวัว (center point) ว่าเคลื่อนที่ข้ามเส้นจากด้านหนึ่งไปอีกด้านหรือไม่ เมื่อวัวเดินข้ามเส้น ระบบจะระบุว่าเป็น "เข้า" หรือ "ออก" โดยพิจารณาจากทิศทางของการเคลื่อนไหว (พิกัด Y ก่อนและหลังข้ามเส้น) ทำให้จำนวนวัวปัจจุบันจะถูกอัปเดตแบบเรียลไทม์ บนหน้าเว็บในภาพ Live ของกล้องตัวที่2 และเมื่อวัวได้เข้าไปในคอกหมด ระบบจะทำการแจ้งเตือนจำนวนวัวทั้งหมดที่เดินผ่านเข้าคอกไป ในลักษณะ Pop-up บนหน้าเว็บแอปพลิเคชันหลังผ่านไป 30วินาที

3.8 หลักการทำงานของดีเลย์

3.8.1 การควบคุมรีเลย์แบบเรียลไทม์ ผู้ใช้สามารถควบคุมการทำงานของรีเลย์ โดยจะแสดงปุ่มควบคุมรีเลย์ทั้งหมด 8 ช่อง (Relay 1 ถึง Relay 8) ผ่านปุ่มควบคุมบนหน้าเว็บ ซึ่งปุ่มสามารถแสดงสถานะว่า เปิด หรือ ปิด แบบเรียลไทม์ โดยเปลี่ยนสีและข้อความ ดังนั้นเมื่อผู้ใช้คลิกที่ปุ่มเปิด หรือ ปิด ระบบจะส่งคำสั่งผ่าน HTTP Request ไปยัง Flask Server ซึ่งจะส่งสัญญาณผ่าน GPIO ไปยังรีเลย์โมดูลและเปลี่ยนสถานะของรีเลย์นั้น ๆ ตามที่สั่ง

3.8.2 ระบบสามารถกำหนดเวลาการทำงานของรีเลย์อัตโนมัติแต่ละช่องได้ผ่านหน้าเว็บ โดยผู้ใช้สามารถเลือกช่วงเวลาเริ่มต้นและสิ้นสุดล่วงหน้าได้ ซึ่งผู้ใช้สามารถเลือกการตั้งค่าได้ดังนี้ เวลาเริ่มต้น (เปิด), เวลาสิ้นสุด (ปิด) และวันที่ทำงาน (ครั้งเดียว หรือ ทุกวัน) หลังจากนั้นระบบจะเก็บข้อมูลการตั้งค่า

เหล่านี้ไว้ในฐานข้อมูล SQLite โดยระบบจะมีฟังก์ชัน Scheduler ที่ทำงานแบบ Background คอยตรวจสอบเวลาปัจจุบันและสั่งงานรีเลย์ให้เปิดหรือปิดตามเวลาที่กำหนด

3.8.3 ติดตั้งระบบควบคุมและเชื่อมต่อโหลดกับเอาต์พุตของตู้ควบคุม โดยนำชุดอุปกรณ์ทั้งหมดไปติดตั้งในพื้นที่เลี้ยงวัว โดยเริ่มจากการติดตั้งตู้ควบคุมที่บรรจุบอร์ดควบคุมรีเลย์และอุปกรณ์ที่เกี่ยวข้อง ซึ่งได้มีการเดินสายเอาต์พุตจากรีเลย์เพื่อต่อเข้ากับโหลดจริง ทั้งนี้เพื่อทดสอบความสามารถในการควบคุมการเปิด และปิดอุปกรณ์ผ่านทางหน้าเว็บแอปพลิเคชันในสภาพแวดล้อมที่ใช้งานจริง

รูปที่ 3-9 อุปกรณ์ภายในตู้

รูปที่ 3-10 ตรวจสอบการทำงาน

3.9 การทดสอบการควบคุมรีเลย์ผ่านเว็บแอป

การควบคุมรีเลย์ผ่านหน้าเว็บโดยใช้สามารถสั่งเปิด และปิดอุปกรณ์ไฟฟ้าที่เชื่อมต่อกับรีเลย์ได้ การทดสอบเริ่มจากการเปิดหน้าเว็บแอปพลิเคชันที่รันอยู่บน Raspberry Pi โดยใช้เบราว์เซอร์ที่อยู่ในเครือข่ายเดียวกัน หลังจากเข้าสู่หน้าเว็บ ให้กดปุ่มควบคุม ON หรือ OFF ของแต่ละช่องรีเลย์ ซึ่งบนหน้าเว็บจะแสดงสถานะของรีเลย์แบบเรียลไทม์

เมื่อกดปุ่ม ON หรือ OFF ระบบจะส่งคำสั่งผ่าน Flask Backend ไปควบคุม GPIO บน Raspberry Pi เพื่อเปลี่ยนสถานะของรีเลย์ จากนั้นให้สังเกตอุปกรณ์ที่เชื่อมต่ออยู่ เช่น หลอดไฟหรือพัดลม นอกจากนี้ยังมีการกดสลับสถานะอย่างต่อเนื่องหลายครั้งเพื่อทดสอบความแม่นยำและความรวดเร็วในการตอบสนองของระบบ

3.10 การทดสอบการตั้งเวลาเปิด และปิดรีเลย์

ระบบมีฟังก์ชันสำหรับตั้งเวลาเปิด และปิดรีเลย์ล่วงหน้าเพื่อความสะดวกในการควบคุมอุปกรณ์อัตโนมัติ โดยได้ดำเนินการกรอกเวลาเปิด (ON) และเวลาปิด (OFF) ลงในแบบฟอร์มหน้าเว็บ พร้อมเลือกหมายเลขรีเลย์ที่ต้องการควบคุม และเลือกโหมดเป็น “ทุกวัน” หรือ “ครั้งเดียว” ตามต้องการ จากนั้นกดปุ่ม “Add Schedule”ระบบจะบันทึกรายการตั้งเวลาและแสดงในตาราง “Time Tasks” บนหน้าเว็บ เมื่อถึงเวลาที่ตั้งไว้ ระบบจะเปลี่ยนสถานะของรีเลย์โดยอัตโนมัติ โดยสามารถสังเกตอุปกรณ์ที่เชื่อมต่อ ว่ามีการเปิดหรือปิดตามเวลาที่กำหนดไว้

3.11 การทดสอบการแสดงภาพจากกล้องตัวที่ 1

กล้องตัวแรกถูกออกแบบมาให้แสดงภาพสดแบบเรียลไทม์ และบันทึกวิดีโอเก็บไว้ย้อนหลัง 7 วัน ทดสอบโดยการเข้าสู่หน้าเว็บเพื่อดูภาพสดจากกล้องตัวที่ 1 ตรวจสอบความต่อเนื่องของภาพ ประเมินคุณภาพของวิดีโอ และการแสดงผลลื่นไหลในดูผ่านอุปกรณ์มือถือหรือคอมพิวเตอร์ที่เชื่อมกับเครือข่ายเดียวกัน

ในส่วนของการบันทึกวิดีโอ ระบบจะทำการสร้างไฟล์วิดีโอแยกตามวันที่และเวลา และจัดเก็บไว้ในโฟลเดอร์ recordings โดยอัตโนมัติ ทำการตรวจสอบการบันทึกวิดีโอย้อนหลัง โดยเข้าไปยังโฟลเดอร์ที่

เก็บไฟล์ recording และตรวจสอบชื่อไฟล์ว่าตรงกับวันที่ที่บันทึก รวมถึงทดสอบระบบลบไฟล์อัตโนมัติเมื่อมีอายุมากกว่า 7 วัน และสามารถเปิดเล่นย้อนหลังได้

3.12 การทดสอบการแสดงภาพจากกล้องตัวที่ 2

กล้องตัวที่ 2 ทำหน้าที่ตรวจนับการเข้า และออกของวัวในคอกผ่านเทคนิค Line Crossing Detection ทดสอบโดยเข้าสู่หน้าเว็บและตรวจสอบภาพสดจากกล้องซึ่งมีการแสดงเส้นตรวจจับ จากนั้นทำการเคลื่อนวัตถุ คือ วัว ให้ผ่านเส้นที่กำหนด และสังเกตการเพิ่มค่าการนับโดยจะแสดงตัวเลข IN บนภาพแบบเรียลไทม์ นอกจากนี้ ยังตรวจสอบหน้าข้อมูล /cow_event ว่ามีการแสดงเวลาของวัวตัวสุดท้ายที่ผ่านเข้าคอก และจำนวนวัวที่นับได้ทั้งหมด พร้อมการแจ้งเตือนผ่าน Popup บนหน้าเว็บเตือนกรณีไม่มีวัวตัวใหม่เข้าเกิน 30 วินาที

3.13 การทดสอบการตั้งค่า และอ่านค่าจำนวนวัวขั้นต่ำ และสูงสุด

ตั้งค่าเกณฑ์ขั้นต่ำและสูงสุดของจำนวนวัวในคอก เพื่อแจ้งเตือนเมื่อจำนวนวัวอยู่ในช่วงที่ผิดปกติ โดยการทดสอบกรอกค่า Min และ Max cows ผ่านแบบฟอร์ม “Set Cow” บนหน้าเว็บ จากนั้นกด “Update Threshold” เพื่อบันทึกค่า โดยพบว่าค่าที่ตั้งไว้ถูกรักษาไว้ในระบบ และเมื่อจำนวนวัวที่นับได้ในระบบมีค่ามากกว่าค่าสูงสุด หรือมีจำนวนน้อยกว่าค่าต่ำสุด สังเกตการแสดงแจ้งเตือนข้อความ alert เป็นข้อความบนหน้าเว็บ

3.14 การทดสอบการรีเซ็ตค่าการนับวัว

ในการทดสอบเมื่อต้องการเริ่มต้นการนับจำนวนใหม่ ให้ทำการกดปุ่ม “Reset Cow Entry” บนหน้าเว็บเพื่อรีเซ็ตค่าจำนวนวัวทั้งหมด โดยทำการกดปุ่มดังกล่าว แล้วสังเกตค่าบนหน้าเว็บ ว่าถูกรีเซ็ตเป็นศูนย์ โดยไม่มีการเก็บค่าค้าง หรือแสดงผลผิดพลาด แล้วจากนั้นระบบสามารถเริ่มต้นนับใหม่จากวัวตัวต่อไปที่เดินผ่านเส้นได้

รูปที่ 3-11 ติดตั้งระบบในฟาร์ม

รูปที่ 3-12 องศาการติดตั้งกล้อง

บทที่4
การทดสอบและการใช้งานระบบ

ในบทที่ผ่านมากล่าวถึงระบบควบคุมฟาร์มด้วย Raspberry Pi 4 โดยสามารถควบคุมรีเลย์ 8 ช่อง ผ่านเว็บ ตั้งเวลาเปิด และปิดได้ และแสดงภาพสดจากกล้องผ่านหน้าเว็บ พร้อมทั้งระบบตรวจนับวัว รวมไปถึงระบบจะแจ้งเตือนให้สามารถทำงานได้ ตามขอบเขตที่กำหนดไว้ บทนี้จะกล่าถึงการนำระบบที่ได้ออกแบบและสร้างไปทดลองใช้งานจริงและบันทึกผล การทดสอบ ซึ่งจะทำให้ทราบถึง ผลการทำงานของระบบ และข้อบกพร่อง เพื่อจะนำข้อบกพร่องมาแก้ไขและพัฒนาปรับปรุงต่อไป

4.1 การทดสอบการควบคุมรีเลย์ผ่านหน้าเว็บ

ในการทดสอบขั้นตอนการควบคุมรีเลย์ผ่านระบบเว็บแอป ได้ดำเนินการเปิดเว็บแอปพลิเคชันผ่านเว็บเบราว์เซอร์บนอุปกรณ์ที่เชื่อมต่ออยู่ในเครือข่ายเดียวกับบอร์ดควบคุม (Raspberry Pi 4) โดยหน้าเว็บดังกล่าวแสดงปุ่มสำหรับควบคุมการเปิด–ปิดรีเลย์จำนวน 8 ช่อง ได้ทำการกดปุ่มควบคุม ON ของรีเลย์แต่ละช่อง และสังเกตผลการเปลี่ยนแปลงของอุปกรณ์ที่เชื่อมต่อกับรีเลย์โดยใช้หลอดไฟ เพื่อประเมินความถูกต้องในการส่งคำสั่งผ่านหน้าเว็บ ไปยังอุปกรณ์ควบคุมจริง นอกจากนี้ยังได้ทำการกดสลับสถานะของรีเลย์ซ้ำหลายครั้งในระยะเวลาต่อเนื่อง เพื่อตรวจสอบความแม่นยำและความเสถียรของระบบว่ามีการตอบสนองอย่างรวดเร็ว ไม่มีอาการหน่วงหรือขัดข้องระหว่างการสั่งงานซ้ำ

รูปที่ 4-1 ควบคุมรีเลย์ผ่านหน้าเว็บ

จากการทดสอบพบว่า ระบบสามารถควบคุมการเปิด–ปิดของรีเลย์แต่ละช่องผ่านหน้าเว็บได้อย่างมีประสิทธิภาพ โดยคำสั่งจากปุ่ม ควบคุมสามารถแปลผลได้อย่างถูกต้องและแม่นยำ อุปกรณ์ที่เชื่อมต่อกับรีเลย์โดยใช้หลอดไฟ ทำงานสอดคล้องกับคำสั่งที่ได้รับโดยไม่มีความล่าช้า ดังตารางที่ 4-1

รูปที่ 4-2 ทดสอบการเปิดและปิดรีเลย์โดยใช้หลอดไฟ

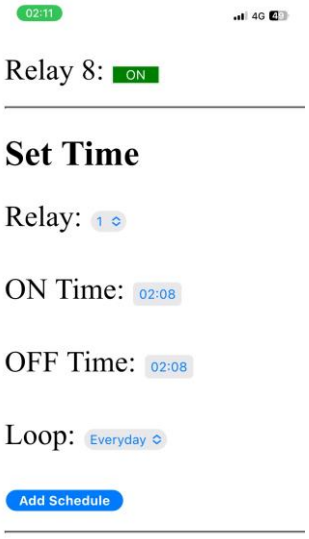
ตารางที่ 4-1 แสดงการทำงานของระบบควบคุมการเปิดและปิดของรีเลย์

เอาต์พุต	การทดสอบ					ผลการทดสอบ
	ครั้งที่1	ครั้งที่2	ครั้งที่3	ครั้งที่4	ครั้งที่5	
ตัวที่1	✓	✓	✓	✓	✓	ทำงานได้ดี
ตัวที่2	✓	✓	✓	✓	✓	ทำงานได้ดี
ตัวที่3	✓	✓	✓	✓	✓	ทำงานได้ดี
ตัวที่4	✓	✓	✓	✓	✓	ทำงานได้ดี
ตัวที่5	✓	✓	✓	✓	✓	ทำงานได้ดี
ตัวที่6	✓	✓	✓	✓	✓	ทำงานได้ดี
ตัวที่7	✓	✓	✓	✓	✓	ทำงานได้ดี
ตัวที่8	✓	✓	✓	✓	✓	ทำงานได้ดี

จากตารางที่ 4-1 การทดสอบการตั้งเวลาเปิดและปิดรีเลย์พบว่ารีเลย์ทุกตัวตั้งแต่ตัวที่ 1 ถึงตัวที่ 8 สามารถตอบสนองต่อคำสั่งควบคุมผ่านเว็บได้อย่างถูกต้องทุกครั้ง โดยมีเครื่องหมายถูก แสดงว่ารีเลย์ตอบสนองต่อคำสั่งได้สำเร็จในแต่ละรอบการทดสอบทั้ง 5 ครั้ง และไม่มีกรณีที่ระบบไม่ตอบสนองหรือตอบสนองผิดพลาดการทำงานของระบบควบคุมรีเลย์มีความ แม่นยำ เสถียร และต่อเนื่อง ช่วยให้สามารถใช้งานในระบบควบคุมอุปกรณ์ไฟฟ้าได้

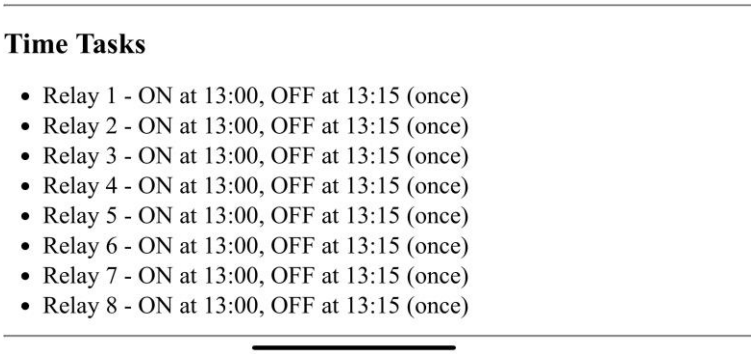
4.2 การทดสอบการตั้งเวลาเปิดและปิดรีเลย์

ในการทดสอบระบบการตั้งเวลาเปิด หรือ ปิดรีเลย์ ได้ดำเนินการผ่านหน้าเว็บแอป โดยสามารถกรอกเวลาที่ต้องการเปิด หรือ ปิด สำหรับรีเลย์แต่ละช่อง พร้อมทั้งเลือกรูปแบบการทำงานว่าต้องการให้ทำงานเพียงครั้งเดียว (Once) หรือ ทำซ้ำทุกวัน (Everyday) เมื่อกรอกข้อมูลครบถ้วนแล้ว กดปุ่ม “Add Schedule” เพื่อเพิ่มตารางเวลาการทำงาน จากนั้นสามารถตรวจสอบว่ารายการตั้งเวลาที่เพิ่มเข้ามาปรากฏในส่วน Time Tasks ของหน้าเว็บอย่างถูกต้อง และรอให้ระบบทำงานตามช่วงเวลาที่กำหนดไว้โดยอัตโนมัติ



รูปที่ 4-3 ระบบการตั้งเวลาเปิดและปิดรีเลย์

ผลจากการทดสอบพบว่า ระบบสามารถทำงานตามเวลาที่ตั้งไว้ได้อย่างถูกต้อง โดยรีเลย์แต่ละช่องสามารถเปลี่ยนสถานะได้ตรงตามเวลาที่ตั้งไว้ทั้งในโหมดทำงานครั้งเดียวและทำซ้ำทุกวัน อุปกรณ์ที่เชื่อมต่อ โดยใช้หลอดไฟ เปิดและปิดได้ตามคำสั่งโดยไม่มีความผิดพลาดหรือความล่าช้า ดังตารางที่ 4-2 และช่วยลดความจำเป็นในการควบคุมแบบแมนนวล



รูปที่ 4-4 ทดสอบการตั้งเวลาเปิดและปิดรีเลย์โดยใช้หลอดไฟ

ตารางที่ 4-2 แสดงการทำงานของระบบการตั้งเวลาเปิดและปิดรีเลย์

เอาต์พุต	ทดสอบการทำงาน				ผลการทดสอบ
	ครั้งเดียว			ทุกวัน	
	5นาที	1ชั่วโมง	6ชั่วโมง		
ตัวที่1	✓	✓	✓	✓	ทำงานตรงตามเวลา
ตัวที่2	✓	✓	✓	✓	ทำงานตรงตามเวลา
ตัวที่3	✓	✓	✓	✓	ทำงานตรงตามเวลา
ตัวที่4	✓	✓	✓	✓	ทำงานตรงตามเวลา
ตัวที่5	✓	✓	✓	✓	ทำงานตรงตามเวลา
ตัวที่6	✓	✓	✓	✓	ทำงานตรงตามเวลา
ตัวที่7	✓	✓	✓	✓	ทำงานตรงตามเวลา
ตัวที่8	✓	✓	✓	✓	ทำงานตรงตามเวลา

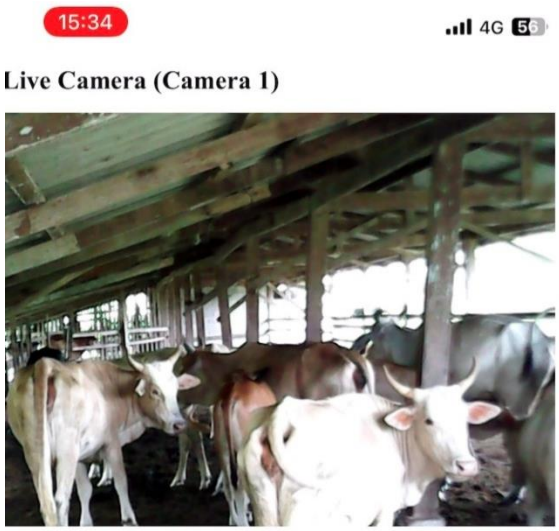
จากตารางที่ 4-2 การทดสอบระบบควบคุมการเปิด และปิดรีเลย์ผ่านฟังก์ชันตั้งเวลาอัตโนมัติของเว็บแอป พบว่าระบบสามารถทำงานได้อย่างมีประสิทธิภาพและตรงตามเวลาที่กำหนดในทุกช่วงการทดสอบ ตั้งเวลาแบบทำงานครั้งเดียว (One-Time) หรือ แบบทำงานซ้ำทุกวัน (Everyday) โดยได้มีการกำหนดระยะเวลาทดสอบที่ 5 นาที, 1 ชั่วโมง 6 ชั่วโมง และ ทำงานซ้ำทุกวัน เพื่อประเมินความแม่นยำของระบบในทุกเงื่อนไขการใช้งานผลจากตารางที่ 4-2 แสดงให้เห็นอย่างชัดเจนว่ารีเลย์ทุกช่องสามารถทำงานได้ตรงตามเวลาที่กำหนด โดยมีเครื่องหมายถูกแสดงถึงความสำเร็จในการทำงานของแต่ละช่วงเวลา

4.3 การทดสอบกล้องตัวที่ 1

ในการทดสอบระบบแสดงภาพสดและการบันทึกวิดีโอของกล้องตัวที่ 1 ได้ดำเนินการผ่านหน้าเว็บแอป เข้าสู่หน้าแสดงผลภาพจากกล้องแบบเรียลไทม์ เพื่อตรวจสอบความต่อเนื่องของการแสดงผล รวมถึงคุณภาพของภาพที่ได้รับผ่านระบบเครือข่าย

นอกจากนี้ ยังได้ตรวจสอบการทำงานของระบบบันทึกวิดีโอโดยอัตโนมัติ โดยไฟล์วิดีโอจะถูกจัดเก็บไว้ในโฟลเดอร์ recordings ภายในระบบ และตั้งชื่อไฟล์ตามวันที่และเวลาที่เริ่มบันทึกอย่างเป็นระบบ เพื่อให้ง่ายต่อการตรวจสอบย้อนหลัง

ในขั้นตอนสุดท้าย ได้มีการทดสอบฟังก์ชันการลบไฟล์วิดีโอโดยอัตโนมัติเมื่ออายุของไฟล์เกินกว่า 7 วัน เพื่อป้องกันการใช้พื้นที่เก็บข้อมูลมากเกินไป ในการจัดเก็บข้อมูลในระยะยาว



รูปที่ 4-5 แสดงภาพจากกล้องแบบเรียลไทม์

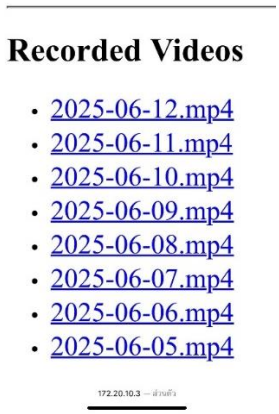
จากการทดสอบพบว่า ระบบสามารถแสดงภาพสดจากกล้องตัวที่ 1 ได้อย่างต่อเนื่อง โดยมีความหน่วงของสัญญาณ (Delay) อยู่ที่ประมาณ 1–3 วินาที ซึ่งอยู่ในเกณฑ์ที่ยอมรับได้ ภาพที่แสดงมีความชัดเจน และไม่พบอาการกระตุกหรือหยุดชะงัก

ระบบสามารถบันทึกวิดีโอลงในโฟลเดอร์ recordings ได้อย่างถูกต้อง โดยไฟล์แต่ละรายการมีชื่อที่อ้างอิงตามวันที่และเวลาที่เริ่มบันทึกอย่างชัดเจน และระบบสามารถลบไฟล์วิดีโอที่มีอายุเกิน 7 วันได้โดยอัตโนมัติ ระบบบันทึกวิดีโอจากกล้อง Camera 1 มีขนาดไฟล์เฉลี่ยวันละประมาณ 2.7 กิกะไบต์ ดังนั้นหากบันทึกต่อเนื่องเป็นเวลา 7 วัน จะใช้พื้นที่รวมประมาณ 18.9 กิกะไบต์ ซึ่งยังไม่เกินขนาดความจุของ MicroSD Card ที่ใช้ในระบบ 64 กิกะไบต์ ทำให้สามารถจัดเก็บวิดีโอย้อนหลังได้โดยไม่กระทบต่อพื้นที่อื่นของระบบ

ตารางที่ 4-3 แสดงการทำงานของระบบแสดงภาพสดและการบันทึกวิดีโอของกล้องตัวที่ 1

การทดสอบ	Live มองเห็นภาพวัวตอนกลางวัน	Live มองเห็นภาพวัวตอนกลางคืน	บันทึกภาพย้อนหลัง	ความรีเลย์ (วินาที)
ครั้งที่1	✓	✓	✓	1
ครั้งที่2	✓	✓	✓	2
ครั้งที่3	✓	✓	✓	2
ครั้งที่4	✓	✓	✓	3
ครั้งที่5	✓	✓	✓	1

จากตารางที่ 4-3 แสดงการทดสอบกล้องตัวที่ 1 กล้องสามารถแสดงภาพสดได้อย่างต่อเนื่องทั้งในช่วงเวลากลางวันและกลางคืน ซึ่งในช่วงกลางวัน ภาพที่ได้มีความคมชัด รายละเอียดของวัวและสภาพแวดล้อมปรากฏชัดเจน ช่วยให้ผู้ใช้งานสามารถตรวจสอบสถานการณ์ภายในฟาร์มได้อย่างแม่นยำ ส่วนในช่วงเวลากลางคืน แม้ว่าคุณภาพของภาพจะลดลงบ้าง โดยเฉพาะความคมชัดและรายละเอียดบางส่วนที่อาจเบลอเล็กน้อย แต่ยังสามารถแยกแยะวัวและการเคลื่อนไหวได้อยู่ในระดับที่ยอมรับได้ จึงถือว่าไม่เป็นอุปสรรคต่อการใช้งานจริงในสภาพแวดล้อมฟาร์มในด้านของการบันทึกภาพย้อนหลัง (Recording) ระบบได้มีการจัดเก็บวิดีโอจากกล้องลงในโฟลเดอร์ recordings อย่างถูกต้องและเป็นระบบ โดยใช้การตั้งชื่อไฟล์ตามวันที่ เพื่อให้สามารถค้นหาและเรียกดูข้อมูลย้อนหลังได้อย่างสะดวกและรวดเร็ว ที่สำคัญ ระบบยังรองรับการจัดการพื้นที่จัดเก็บด้วยการลบไฟล์วิดีโอที่มีอายุเกินกว่า 7 วันโดยอัตโนมัติ ซึ่งช่วยลดภาระในการจัดการข้อมูลและทำให้ระบบสามารถทำงานได้อย่างต่อเนื่องโดยไม่สิ้นเปลืองพื้นที่เก็บข้อมูลในส่วนของค่าความล่าช้าในการแสดงผล จากการทดสอบพบว่าระบบมีค่าหน่วงเวลาประมาณ 1–3 วินาที ซึ่งอยู่ในระดับที่ยอมรับได้สำหรับการใช้งานจริง โดยไม่ส่งผลต่อการตัดสินใจหรือการติดตามพฤติกรรมของวัวแบบเรียลไทม์

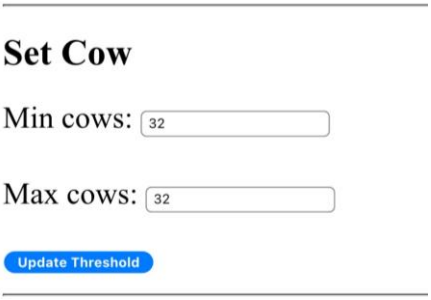


รูปที่ 4-6 แสดงภาพรายการบันทึกวิดีโอย้อนหลัง

4.4 การทดสอบการตั้งค่า และอ่านค่าจำนวนวัว

ในการทดสอบฟังก์ชันการตั้งค่าเกณฑ์จำนวนวัวขั้นต่ำ (Min) และสูงสุด (Max) ที่ระบบสามารถตรวจสอบได้ ได้ดำเนินการตั้งค่าผ่านแบบฟอร์ม “Set Cow” บนหน้าเว็บแอป โดยผู้ใช้งานกรอกค่าจำนวนวัวขั้นต่ำและสูงสุดตามความเหมาะสมของฟาร์ม โดยได้มีการตั้งค่าไว้ที่ 32 ตัว แล้วกดปุ่ม “Update Threshold” เพื่อยืนยันการตั้งค่าใหม่ จากนั้นโหลดหน้าเว็บใหม่เพื่อตรวจสอบการอัปเดตค่าเกณฑ์

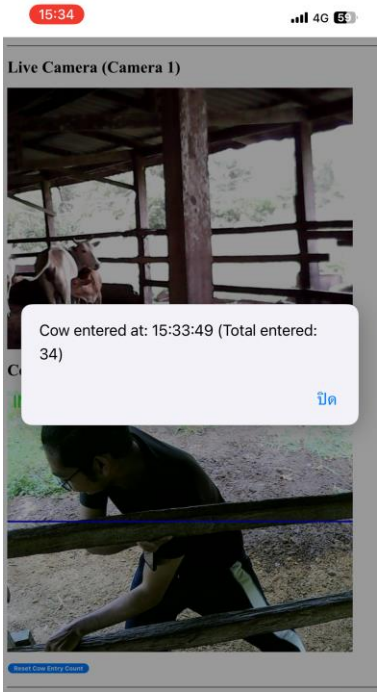
เมื่อระบบเริ่มทำงานโดยตรวจสอบจำนวนวัวจากกล้องว่าระบบจะเปรียบเทียบค่าที่ตรวจจับได้กับค่าขอบเขตที่ตั้งไว้ และจะแสดงการแจ้งเตือน (Alert) บนหน้าเว็บโดยอัตโนมัติ หากจำนวนวัวอยู่ต่ำกว่าค่าขั้นต่ำ หรือเกินกว่าค่าสูงสุดที่กำหนดไว้ เพื่อให้ผู้ใช้งานรับทราบและดำเนินการตรวจสอบหรือแก้ไขสถานการณ์ได้ทันท่วงที



รูปที่ 4-7 การตั้งค่าเกณฑ์จำนวนวัวขั้นต่ำ (Min) และสูงสุด (Max)

จากการทดสอบฟังก์ชันการตั้งค่าและตรวจสอบจำนวนวัวภายในระบบ พบว่าระบบสามารถทำงานได้อย่างมีประสิทธิภาพ โดยสามารถอัปเดตค่าขอบเขตขั้นต่ำและสูงสุดของจำนวนวัว (Min/Max) ได้ผ่านแบบฟอร์ม “Set Cow” บนหน้าเว็บ เมื่อผู้ใช้งานทำการตั้งค่าและกด “Update Threshold” ระบบจะบันทึกค่าที่ตั้งไว้ และเมื่อโหลดหน้าเว็บใหม่ ค่าเหล่านั้นจะยังคงปรากฏอยู่ตามที่กำหนดไว้ โดยไม่เกิดข้อผิดพลาดหรือการสูญหายของข้อมูลแต่อย่างใด

นอกจากนี้ ระบบยังสามารถตรวจสอบจำนวนวัวที่อยู่ในพื้นที่แบบเรียลไทม์ และเมื่อจำนวนวัวเกินหรือต่ำกว่าขอบเขตที่กำหนดไว้ ระบบจะแสดงการแจ้งเตือน เมื่อไม่มีวัวตัวใหม่เดินผ่านเส้น Line ภายในระยะเวลา 30 วินาที เพื่อแสดงว่าวัวตัวสุดท้ายได้ผ่านเข้าไปแล้ว และไม่มีการเคลื่อนไหวใหม่ภายในช่วงเวลาที่กำหนด



รูปที่ 4-8 ฟังก์ชันการแจ้งเตือน

4.5 การทดสอบรีเซ็ตค่าการนับวัว

ในการทดสอบฟังก์ชันการรีเซ็ตค่าการนับจำนวนวัว ได้ดำเนินการโดยเข้าสู่หน้าเว็บแอปที่ใช้ควบคุมระบบ จากนั้นกดปุ่ม “Reset Cow Entry Count” ซึ่งแสดงอยู่บนหน้าเว็บในส่วนที่แสดงจำนวนวัวที่ตรวจนับได้ ระบบจะต้องรีเซ็ตค่าจำนวนวัวที่นับได้กลับเป็นศูนย์ทันทีภายหลังจากการกดปุ่ม

หลังจากดำเนินการรีเซ็ตแล้ว ได้ตรวจสอบค่าจำนวนวัวที่แสดงผลบนหน้าเว็บเพื่อเปรียบเทียบกับสถานะของระบบจริง พบว่าค่าการนับจำนวนวัวถูกรีเซ็ตกลับเป็นศูนย์ตามที่ตั้งไว้ และระบบยังคงสามารถนับจำนวนวัวใหม่ได้ต่อไปโดยไม่มีปัญหา

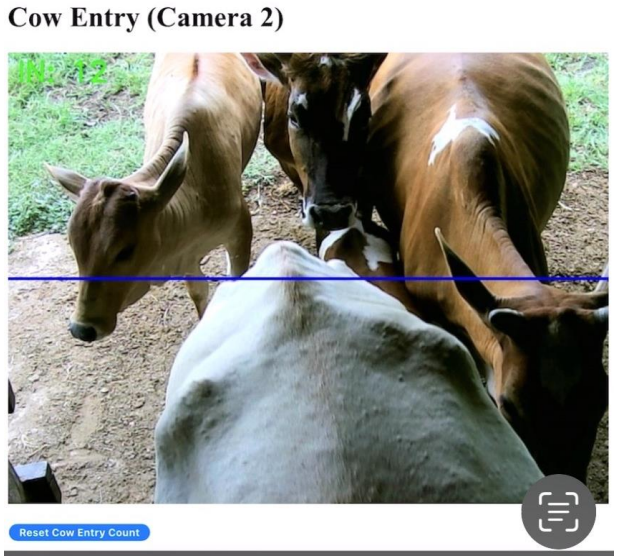
รูปที่ 4-9 ฟังก์ชันการรีเซ็ตค่าการนับจำนวนวัว

จากผลการทดสอบระบบสามารถดำเนินการรีเซ็ตค่าการนับวัวได้อย่างถูกต้องและรวดเร็ว โดยค่าจำนวนวัวที่แสดงผลหลังจากกดปุ่ม “Reset Cow Entry Count” กลับเป็นศูนย์ทันที และไม่พบข้อผิดพลาด หรือความล่าช้าในการอัปเดต

4.6 การทดสอบกล้องตัวที่ 2

4.6.1 การตั้งค่าระบบและการตรวจสอบการแสดงผลภาพในการทดสอบระบบกล้องตัวที่ 2 ที่ใช้สำหรับตรวจนับวัวที่เดินผ่านเข้าสู่พื้นที่เลี้ยง ระบบได้ถูกตั้งค่าให้แสดงภาพสดผ่านหน้าเว็บ พร้อมแสดงเส้น Line บนภาพเพื่อใช้ในการตรวจจับการเคลื่อนผ่านของวัว เมื่อตรวจพบว่าวัวเดินผ่านเส้น Line ดังกล่าว ระบบจะเพิ่มค่าการนับจำนวนวัวเข้า (IN) โดยจะแสดงผลแบบเรียลไทม์บนภาพจากกล้อง

4.6.1 การทดสอบการทำงานของระบบนับและแจ้งเตือนเพื่อทดสอบความถูกต้องของการทำงาน ได้ทำการเคลื่อนวัวให้เดินผ่านเส้น Line โดยสังเกตค่าการนับวัวที่ปรากฏ และตรวจสอบว่าระบบสามารถอัปเดตจำนวนวัวเข้าได้ถูกต้องตามสถานการณ์จริง



รูปที่ 4-10 การแสดงผลภาพในการทดสอบระบบกล้องตัวที่ 2

จากผลการทดสอบระบบตรวจจับและนับจำนวนวัวที่เดินผ่านกล้อง พบว่าระบบสามารถนับจำนวนวัวได้ ซึ่งถือว่าอยู่ในระดับที่สามารถนำไปใช้งานจริงได้ อย่างไรก็ตามยังพบความคลาดเคลื่อนบางประการที่ส่งผลต่อความแม่นยำในการนับ โดยมีปัจจัยสำคัญ ได้แก่ กรณีที่วัวหลายตัวเดินเข้าพร้อมกันในช่วงเวลาเดียวกัน ทำให้ระบบไม่สามารถแยกแยะตัววัวได้ชัดเจน อาจเกิดการนับผิดพลาดอีกปัจจัยคือมุมมองของกล้องจากการทดลองพบว่า การติดตั้งกล้องในมุมเฉียงประมาณ 45 องศาให้ผลลัพธ์ดีที่สุด

ช่วยให้ระบบมองเห็นรูปร่างและการเคลื่อนไหวของวัวได้ชัดเจน นอกจากนี้ ความเร็วในการเดินของวัวก็มีผลต่อความแม่นยำ หากวัวเดินผ่านกล้องด้วยความเร็วสูง ระบบอาจไม่สามารถประมวลผลได้ทัน ทำให้เกิดการนับพลาด ระบบมีความสามารถเพียงพอสำหรับการใช้งานจริง หากมีการจัดการปัจจัยด้านมุมกล้อง ความเร็ว และความหนาแน่นของวัวอย่างเหมาะสม

ในการทดสอบระบบนับและแจ้งเตือนพบว่า ระบบสามารถตรวจจับและนับจำนวนวัวที่เดินผ่านเส้น Line โดยค่าการนับที่ปรากฏบนหน้าจอสอดคล้องกับจำนวนวัวที่ผ่านจริงในแต่ละรอบการทดสอบ ระบบสามารถอัปเดตข้อมูลจำนวนวัวเข้าแบบเรียลไทม์

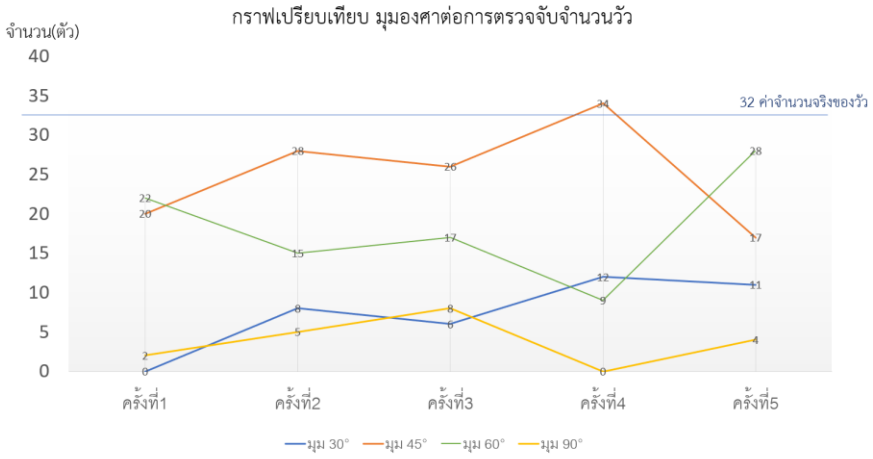
ตารางที่ 4-4 แสดงการทำงานของกล้องตัวที่ 2

มุมกล้อง (องศา)	ครั้งที่	จำนวนจริง (ตัว)	จำนวนตรวจนับได้ (ตัว)	ค่าเฉลี่ยนับได้ (ตัว)	ค่าความผิดพลาด (%)
30	1	32	0	7	78.12
	2	32	8		
	3	32	6		
	4	32	12		
	5	32	11		
45	1	32	20	25	21.87
	2	32	28		
	3	32	26		
	4	32	34		
	5	32	17		
60	1	32	22	18	43.75
	2	32	15		
	3	32	17		
	4	32	9		
	5	32	28		

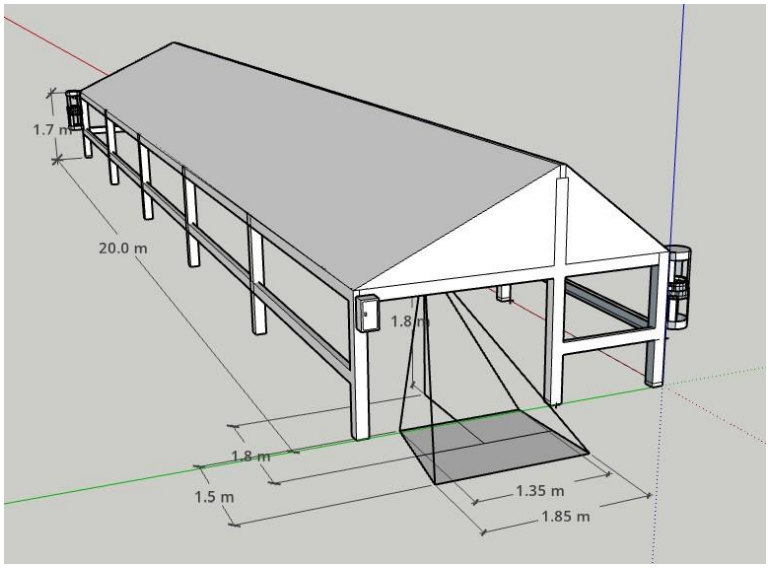
90	1	32	2	4	87.5
	2	32	5		
	3	32	8		
	4	32	0		
	5	32	4		

จากตารางแสดงผลการทดสอบประสิทธิภาพของระบบตรวจจับวัวในมุมกล้องต่าง ๆ ได้แก่ มุม 30°, 45°, 60° และ 90° โดยพิจารณาจากจำนวนวัวที่ตรวจจับได้ ภาพในมุม 45° ให้ผลลัพธ์ดีที่สุด ระบบสามารถตรวจจับวัวได้ใกล้เคียงกับจำนวนจริงในทุกครั้งที่ทดสอบ โดยเฉพาะในการทดสอบครั้งที่ 4 ที่ตรวจจับได้ถึง 34 ตัว จากทั้งหมด 32 ตัว แสดงให้เห็นถึงประสิทธิภาพสูงสุดของระบบในมุมนี้ มุม 30° และ 90° ให้ผลลัพธ์ต่ำที่สุด โดยเฉพาะมุม 90° ที่ในการทดสอบบางครั้งตรวจจับได้เพียง 0–2 ตัว

มุม 60° ให้ผลลัพธ์ระดับปานกลาง แม้บางครั้งจะตรวจจับได้จำนวนวัวค่อนข้างดี เช่น ครั้งที่ 1 (22 ตัว) และครั้งที่ 5 (28 ตัว) แต่มีความแปรปรวนสูงในแต่ละรอบจากผลการทดสอบดังกล่าวสามารถสรุปได้ว่า มุมกล้องที่เหมาะสมที่สุดสำหรับการติดตั้งเพื่อตรวจจับการเดินผ่านของวัวคือ มุม 45° ซึ่งให้ความแม่นยำในการตรวจจับ ดังรูปที่ 4-12



รูปที่ 4-11 แสดงการทำงานของกล้องตัวที่ 2 สำหรับตรวจนับวัวเข้า



รูปที่ 4-12 ตำแหน่งการติดตั้งกล้องตรวจจับวัว

4.7 สรุปและวิเคราะห์ผลการทดลอง

ระบบควบคุมรีเลย์ผ่านหน้าเว็บ สามารถสั่งงานเปิด-ปิดอุปกรณ์ไฟฟ้าได้ทันทีผ่าน Web Application ที่พัฒนาขึ้นโดยใช้ Flask บนบอร์ด Raspberry Pi 4 ผลการทดลองแสดงให้เห็นว่า ระบบสามารถรับคำสั่งได้รวดเร็ว ไม่มีความล่าช้าหรือข้อผิดพลาด และสามารถใช้งานผ่านอุปกรณ์ในเครือข่ายเดียวกันได้โดยไม่มีปัญหาใด ๆ

ต่อมาในส่วนของ ระบบตั้งเวลารีเลย์ ได้ทำการทดลองทั้งแบบตั้งเวลาเปิดและปิดเพียงครั้งเดียว และแบบทำซ้ำทุกวัน โดยตั้งระยะเวลาทดสอบที่หลากหลาย ตั้งแต่ 5 นาที ไปจนถึง 6 ชั่วโมง ผลลัพธ์ที่ได้แสดงให้เห็นว่า ระบบสามารถสั่งเปิดและปิดรีเลย์ได้ตรงตามเวลาที่กำหนดทุกกรณี ระบบตรวจสอบเวลาและสั่งการอย่างแม่นยำโดยอัตโนมัติผ่าน Scheduler

สำหรับ กล้องตัวที่ 1 ซึ่งทำหน้าที่แสดงภาพแบบ Live Feed ผ่านหน้าเว็บ พร้อมกับการบันทึกวิดีโอลงในรูปแบบ .mp4 และจัดเก็บไฟล์โดยแยกตามวันที่ พบว่าระบบสามารถแสดงภาพสดได้ราบรื่น มีความคมชัดเพียงพอในการใช้งานทั้งกลางวันและกลางคืน แม้ในสภาพแสงน้อย ภาพจะไม่คมชัดเท่ากลางวัน แต่ยังสามารถใช้งานได้จริง นอกจากนี้ระบบยังมีการจัดการพื้นที่จัดเก็บอย่างมีประสิทธิภาพ โดยลบไฟล์วิดีโอที่มีอายุเกิน 7 วันโดยอัตโนมัติ

ส่วนที่สำคัญที่สุดของโครงงาน คือ ระบบตรวจนับจำนวนวัว ผ่านกล้องตัวที่ 2 โดยใช้เทคโนโลยี YOLOv5 และ Roboflow API ในการตรวจจับวัตถุภายในภาพ จากการทดสอบมุมกล้อง พบว่ามุม 45 องศาให้ผลดีที่สุด โดยตรวจจับวัวได้แม่นยำ มีความล่าช้าน้อย และตรวจนับได้ถูกต้อง ส่วนมุม 30 และ 90 องศา ให้ผลลัพธ์ที่ไม่น่าพึงพอใจ โดยเฉพาะมุม 90 องศาที่แทบไม่สามารถตรวจจับวัวได้เลย ขณะที่มุม 60 องศาให้ผลอยู่ในระดับปานกลางและไม่คงที่ สะท้อนให้เห็นว่ามุมกล้องเป็นปัจจัยสำคัญที่ส่งผลต่อประสิทธิภาพของระบบตรวจจับภาพ

นอกจากนี้ยังมีการโดยผู้ใช้สามารถกำหนดค่าผ่านหน้าเว็บ ระบบจะตรวจนับวัวแบบเรียลไทม์จากกล้องตัวที่ 2 หากจำนวนวัวในคอกต่ำกว่าหรือเกินกว่าค่าที่กำหนด ระบบจะทำการแจ้งเตือนทันทีผ่านหน้าเว็บด้วยข้อความ Pop-up การแจ้งเตือนจะหายไปโดยอัตโนมัติ ผลการทดสอบแสดงให้เห็นว่าระบบสามารถอ่านค่า เปรียบเทียบ และแจ้งเตือนได้อย่างแม่นยำ

ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT ที่พัฒนาขึ้นนี้สามารถทำงานได้ตรงตามเป้าหมายและวัตถุประสงค์ที่ตั้งไว้ ระบบมีความแม่นยำ เสถียร และพร้อมใช้งานจริงในฟาร์ม มีความสามารถในการควบคุมอุปกรณ์ไฟฟ้าแบบเรียลไทม์ แสดงภาพสด ตรวจนับจำนวนวัว ตรวจสอบค่าที่ตั้งไว้ พร้อมระบบแจ้งเตือนและจัดเก็บข้อมูลอย่างเป็นระบบ

บทที่ 5
สรุปผลและข้อเสนอแนะ

จากการดำเนินการจัดทำเรื่องระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT ได้รับการออกแบบและพัฒนาขึ้นโดยมีวัตถุประสงค์เพื่อยกระดับการจัดการฟาร์มโคเนื้อให้มีความทันสมัยและมีประสิทธิภาพมากยิ่งขึ้น โดยการนำเทคโนโลยี Internet of Things (IoT) มาผสานกับระบบควบคุมอัตโนมัติ การประมวลผลภาพจากกล้อง และการแสดงผลข้อมูลผ่านเว็บแอปพลิเคชันในแบบเรียลไทม์ สามารถควบคุมและตรวจสอบการทำงานของฟาร์มได้สะดวก

5.1 สรุปผลการดำเนินงาน

จากการศึกษาและทำการทดลองโครงงานระบบควบคุมในฟาร์มเลี้ยงโคเนื้อผ่านระบบ IoT ทั้งหมดถูกพัฒนาขึ้นโดยใช้บอร์ด Raspberry Pi 4 เป็นศูนย์กลางในการควบคุม ซึ่งสามารถเชื่อมต่อกับอุปกรณ์ไฟฟ้าต่าง ๆ ผ่านโมดูลรีเลย์ และรองรับการเขียนโปรแกรมด้วยภาษา Python โดยใช้เฟรมเวิร์ก Flask สำหรับพัฒนาเว็บแอปพลิเคชัน ระบบสามารถควบคุมการเปิดหรือปิดอุปกรณ์ผ่านหน้าเว็บได้ทันที และยังมีฟังก์ชัน ตั้งเวลาอัตโนมัติ ทั้งแบบครั้งเดียวและแบบทำซ้ำทุกวัน ผู้ใช้งานสามารถกำหนดเวลาได้ตามความต้องการ ซึ่งระบบจะทำงานผ่าน Scheduler โดยตรวจสอบเวลาและสั่งการอย่างแม่นยำในทุกกรณีที่ทดสอบ

ในด้านระบบกล้อง ได้ติดตั้งกล้องสองตัว โดยกล้องตัวแรกทำหน้าที่ แสดงภาพสด และบันทึกวิดีโอในรูปแบบ .mp4 ไฟล์จะถูกจัดเก็บลงในหน่วยความจำและลบอัตโนมัติเมื่อครบ 7 วัน เพื่อรักษาพื้นที่จัดเก็บให้เหมาะสมสำหรับการใช้งานระยะยาว กล้องตัวที่สองถูกนำมาใช้ในระบบ ตรวจนับจำนวนวัว โดยใช้เทคนิคการตรวจจับวัตถุด้วยโมเดล YOLOv5 และประมวลผลร่วมกับแพลตฟอร์ม Roboflow ในการวิเคราะห์ภาพที่ได้รับแบบเรียลไทม์ จากการทดสอบพบว่า มุมกล้องที่ 45 องศาให้ผลลัพธ์ดีที่สุด ทั้งด้านความแม่นยำในการตรวจจับ ความชัดเจนของวัตถุ และการตอบสนองที่รวดเร็ว นอกจากนี้ระบบยังรองรับการตั้งค่า จำนวนวัวขั้นต่ำและสูงสุดในคอก ผ่านหน้าเว็บ หากจำนวนวัวที่ตรวจนับได้ไม่อยู่ในช่วงที่กำหนด ระบบจะทำการแจ้งเตือนผ่านหน้าจอแบบ Pop-up โดยทันที

5.2 ปัญหาและอุปสรรค

5.2.1 วัวหลายตัวเดินเข้าพร้อมกัน ทำให้ระบบตรวจจับไม่สามารถแยกวัวแต่ละตัวได้ชัดเจน ส่งผลให้เกิดการนับผิดพลาด

5.2.2 ข้อจำกัดด้านความสูงและตำแหน่งติดตั้งกล้องในพื้นที่จริงบริเวณคอกวัว

5.2.3 ภาพจากกล้องมีความล่าช้าเล็กน้อยเมื่อส่งมายังหน้าเว็บ ซึ่งอาจทำให้การนับวัวในเวลาจริง

5.2.4 การทดสอบระบบตรวจนับวัวทำได้เพียงวันละ 2 ครั้ง เนื่องจากวัวจะเข้าและออกจากคอกในช่วงเช้าและเย็น

5..2.5 ความเร็วในการเดินของวัว เมื่อวัวเดินเร็วหรือวิ่งผ่านกล้อง ระบบอาจประมวลผลภาพไม่ทัน ทำให้การตรวจนับขาดหายหรือคลาดเคลื่อน

5.2.6 ความเร็วของอินเทอร์เน็ต ระบบบางส่วนต้องใช้การเชื่อมต่ออินเทอร์เน็ต เช่น การใช้ Roboflow API หากอินเทอร์เน็ตช้าจะส่งผลต่อการประมวลผล

5.3 ข้อเสนอแนะ

5.3.1 ควรเลือกตำแหน่งในการติดตั้งกล้องที่มีมุมมองชัดเจนและสูงพอที่จะเห็นบริเวณภายในฟาร์ม เพื่อให้สามารถบันทึกภาพได้อย่างครบถ้วน

5.3.2 ควรเลือกตำแหน่งติดตั้งกล้องนับจำนวนให้ครอบคลุมมัมกล้องบริเวณทางเข้าและออกอย่างชัดเจน เพื่อให้การตรวจจับมีความแม่นยำ

5.4 แนวทางในการพัฒนาโครงงาน

5.4.1 ปรับแต่งหน้าเว็บให้มีความสวยงามและใช้งานง่าย เพื่อให้ผู้ใช้งานสามารถเข้าถึงข้อมูลได้สะดวก

5.4.2 เพิ่มจำนวนกล้องให้ครอบคลุมทั้งบริเวณภายนอกและภายในฟาร์ม

5.4.3 เพิ่มข้อมูลภาพวัวเพื่อฝึกโมเดล YOLO ผ่าน Roboflow ให้ระบบสามารถเรียนรู้และตรวจจับวัวได้ดียิ่งขึ้น

บรรณานุกรม

[1] Raspberry Pi Foundation. (2567). Raspberry Pi. สืบค้นจาก https://www.raspberrypi.com

[2] Roboflow. (2567). Roboflow: Train and Deploy Computer Vision Models. สืบค้นจาก https://www.roboflow.com

[3] อภิสิทธิ์ มานะกิจ. (2565). ระบบฟาร์มอัจฉริยะด้วย Internet of Things (IoT) และการประมวลผลภาพ. วิทยานิพนธ์วิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี.

[4] สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. (2564). การพัฒนาระบบตรวจสอบวัวในคอกเลี้ยงด้วย AI. งานประชุมวิชาการวิศวกรรมไฟฟ้าแห่งชาติ.

[5] สุรเชษฐ์ พูลสวัสดิ์. (2566). การสตรีมวิดีโอจาก Raspberry Pi ด้วย OpenCV และ Flask. เอกสารประกอบการอบรมวิชาชีพเทคโนโลยีสารสนเทศ.

[6] วิรัช ศรีสำอาง. (2565). การควบคุมอุปกรณ์ไฟฟ้าด้วยรีเลย์และเว็บเซิร์ฟเวอร์. โครงงานวิศวกรรมไฟฟ้า มหาวิทยาลัยขอนแก่น.

ภาคผนวก ก
งบประมาณ

งบประมาณ

ลำดับที่	รายการ	จำนวน(ชิ้น)	ราคา(บาท)
1	บอร์ด raspberry pi 4 model b 4gb	1	2,350
2	Micro SD Card 64 Gb	1	290
3	ตู้สวิตซ์บอร์ด รุ่น CTBS 25x35x15	1	662
4	พาวเวอร์ปลั๊ก รุ่น P1-313TH	8	56
5	พาวเวอร์ปลั๊ก รุ่น P1-0123TH	1	63
6	เบรกเกอร์ Nano 30 A	1	40
7	Terminal TBR-10 600V 10A 8ช่อง	1	15
8	Bus bar ขนาด 9x6	2	21
9	Relay 8 Channel Relay 5V	1	185
10	สายไฟ	1	300
11	กล้อง USB	1	256
12	กล้อง USB Webcam Hoco Di06	1	600
13	หางปลา	1	100
14	น๊อตยึดบอร์ด	8	40
15	น็อตเจาะเหล็ก	1	45
16	Adapter ureen	1	195
17	สาย USB-C to USB-A	1	95
18	อุปกรณ์อื่น ๆ	-	200
รวมเป็นเงิน			5,926

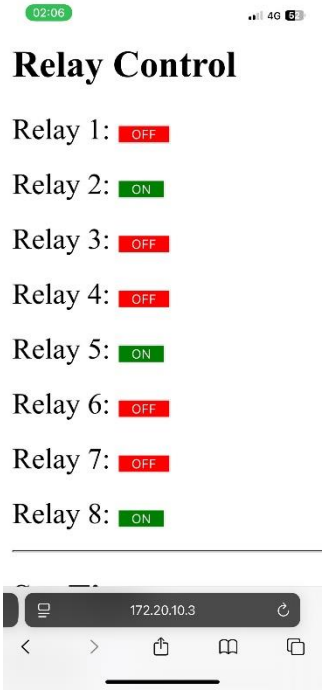
ภาคผนวก ข
คู่มือแนะนำการใช้งาน

คู่มือแนะนำการใช้งานเว็บเซิร์ฟเวอร์ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อ

1. วิธีการเชื่อมต่ออุปกรณ์เพื่อเข้าสู่หน้าเว็บเซิร์ฟเวอร์ระบบควบคุมในฟาร์มเลี้ยงโคเนื้อ

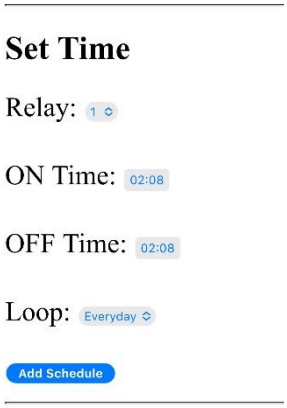
1.1 เชื่อมต่ออุปกรณ์สมาร์ทโฟนเข้ากับเครือข่ายอินเทอร์เน็ตผ่าน Wi-Fi ตรวจสอบให้แน่ใจว่าอุปกรณ์เชื่อมต่อเข้ากับเครือข่ายเดียวกันกับที่เว็บเซิร์ฟเวอร์ตั้งอยู่

1.2 เปิดเว็บเบราว์เซอร์กรอก IP Address: 172.20.10.3 เพื่อเข้าสู่หน้าเว็บเซิร์ฟเวอร์



ภาพที่ 1 คือการตั้งค่าเปิด และปิดดีเลย์ทั้ง 8 ตัว แบบ manual

เมื่อผู้ใช้เปิดระบบขึ้นมาจะเห็นสถานะปุ่มไอคอนดีเลย์ทุกตัวจะเป็น OFF และมีสีแดง เมื่อผู้ใช้ต้องการจะเปิดเอาต์พุตตัวไหน ให้เลือกดีเลย์ที่ตรงกับตำแหน่งเอาต์พุตตัวนั้น โดยทำการคลิกไปที่ปุ่มไอคอนตัวนั้น สถานะปุ่มไอคอนดีเลย์จะเปลี่ยนเป็น ON และมีสีเขียว

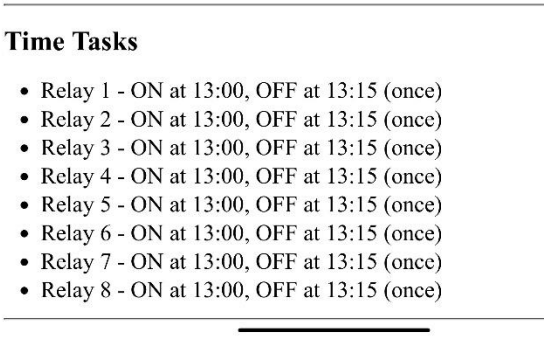


ภาพที่ 2 คือการตั้งค่าการทำงาน ของดีเลย์ แบบ Auto

ผู้ใช้สามารถกำหนดการทำงานของดีเลย์ หรือเอาต์พุตแต่ละตัวได้

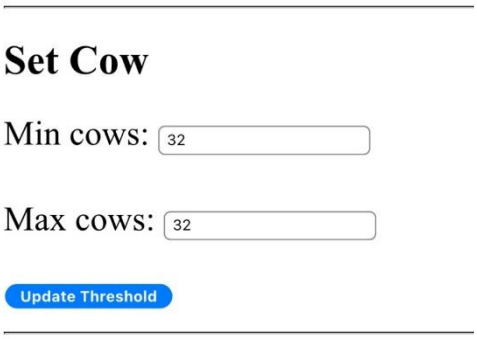
วิธีการปรับตั้ง

1. เลือกลำดับดีเลย์ หรือตำแหน่งเอาต์พุต
2. ตั้งเวลาเปิดการทำงาน (ON Time)
3. ตั้งเวลาปิดการทำงาน (OFF Time)
4. เลือกรูปแบบการทำงาน โดยจะมี 2 สถานะ คือ ให้ทำงานเพียงครั้งเดียว (Once) หรือ ให้ทำงานทุกวัน (Everyday) ตามเวลาที่ตั้งไว้



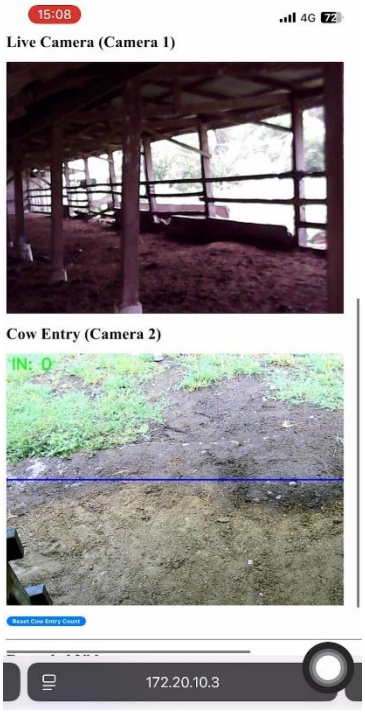
ภาพที่ 3 คือรายการแสดงการทำงานของดีเลย์

โดยผู้ใช้สามารถเข้ามาตรวจดูรายการทำงานย้อนหลัง ของเอาต์พุต หรือดีเลย์แต่ละตัวได้



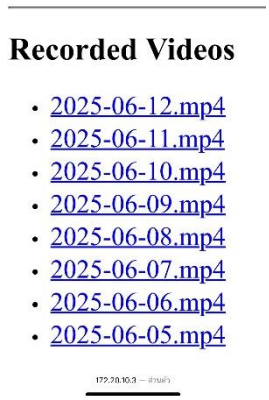
ภาพที่ 4 การตั้งค่าจำนวนวัวที่มีอยู่จริงในฟาร์ม

ผู้ใช้สามารถตั้งค่าจำนวนวัว โดยกรอกค่าจำนวนวัวที่มีอยู่จริงในคอก ไปที่ตำแหน่ง Min cows และ Max cows หลังจากนั้นทำการกดอัพเดทค่า



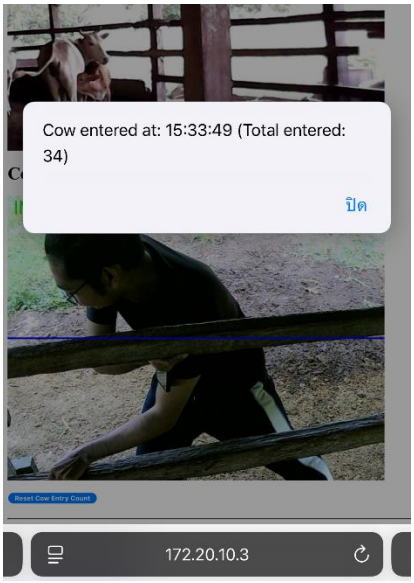
ภาพที่ 5 คือวีดีโอไลฟ์สดจากกล้องตัวที่ 1 และกล้องตัวที่ 2

โดยในตำแหน่งกล้องตัวที่ 2 ผู้ใช้สามารดูจำนวนวัวที่เดินผ่านเข้าไปในคอกได้จากมุมซ้ายบนของวีดีโอ ซึ่งจะแสดงจำนวนแบบเรียลไทม์



รูปที่ 6 คือรายการไฟล์วีดีโอที่ได้ทำการบันทึกย้อนหลัง เป็นระยะเวลา 7วัน

ผู้ใช้สามารถกดเข้าไปดูวีดีโอที่ได้ทำการบันทึกไว้ได้ โดยทำการกดเข้าไปในวันที่ต้องการดู จะมีวีดีโอแสดงขึ้นมาทันที



รูปที่ 7 คือการแจ้งเตือนเวลาวัวตัวสุดท้าย และจำนวนวัวทั้งหมดที่นับได้

โดยผู้ใช้จะเห็นการแจ้งเตือนเด้งขึ้นมาหน้าเว็บแอป หลังเวลาวัวตัวสุดท้ายผ่านเข้าคอกไป 30 วินาที

ภาคผนวก ค
SOURCE CODE

Source Code ที่เขียนลงบอร์ด บอร์ด raspberry pi 4

```
//import โมดูลสำคัญจาก Flask ที่ใช้สร้าง web application
from flask import Flask, render_template_string, request, redirect, Response,
send_from_directory, jsonify
//เรียกใช้โมดูล OpenCV
import cv2
//เรียกใช้โมดูลควบคุมรีเลย์
import relay_controller as rc
//เรียกใช้โมดูลจัดการตารางเวลาการเปิด-ปิดรีเลย์
import scheduler
//ใช้สำหรับจัดการไฟล์และโฟลเดอร์
import os
//ใช้จับเวลา, หน่วงเวลา, และคำนวณเวลาที่ผ่านไป
import time
//สำหรับเรียก API ผ่าน HTTP
import requests
//ใช้จัดการข้อมูล JSON (โหลด/บันทึก)
import json
//ใช้จัดการวันที่และเวลา (เวลาปัจจุบัน, แสดงผลเป็น string
from datetime import datetime
//ใช้เชื่อมต่อ Roboflow API เพื่อตรวจจับวัตถุ
from roboflow import Roboflow
//ใช้สร้าง Thread สำหรับรันฟังก์ชันคู่ขนาน เช่น ตรวจจับวัวจากกล้อง 2
from threading import Thread
//เรียกซ้ำ (ซ้ำกับด้านบน) ⟶ ควรลบหนึ่งบรรทัดออก
from datetime import datetime
```

```
//ตั้งค่าและเปิดใช้งานกล้อง
cow_count_camera2 = 0
last_frame_camera2 = None
last_cow_in_time = None
last_cow_out_time = None
notified_all_in = False
notified_all_out = False
last_enter_time = None
notified_final_cow = False
//โหลดโมเดลตรวจจับวัตถุจาก Roboflow Cloud โดยใช้ API
rf = Roboflow(api_key="tyMzuIe6ngXF8zf2F7dj")
project = rf.workspace("cowfollow").project("cow-yolo-gutmp")
model = project.version(2).model
//กำหนดไฟล์สำหรับเก็บ threshold
THRESHOLD_FILE = "cow_thresholds.json"
//เริ่มต้น Flask และ Scheduler
app = Flask(__name__)
scheduler.start_scheduler()
//เตรียมโฟลเดอร์เก็บวิดีโอ
os.makedirs("recordings", exist_ok=True)
//เปิดกล้อง กล้อง 1 – สำหรับ Live + บันทึกวิดีโอ
# Camera 1: Live + Recording
camera1 = cv2.VideoCapture(0)
if not camera1.isOpened():
print("Unable to open camera 1")
exit()
```

```
//เปิดกล้อง กล้อง 2 – สำหรับนับวัวเข้า/ออก
# Camera 2: Cow Entry/Exit Counting
camera2 = cv2.VideoCapture(2)
if not camera2.isOpened():
print("Unable to open camera 2")
exit()
//เก็บข้อมูลจำนวนวัวเข้า-ออก และตรวจจับตำแหน่งวัวในเฟรม
# Cow count variables
cow_entered = 0
cow_exited = 0
expected_cows = 5  # Set expected cow count
//ตำแหน่งกลางของวัตถุในเฟรมก่อนหน้า
prev_centroids = []
//ตรวจจับวัวจากกล้อง 2 โดยใช้โมเดล Roboflow
//ประกาศตัวแปร global ที่ใช้ร่วม
def detect_cow_from_camera2():
global cow_count_camera2, last_frame_camera2, cow_entered, prev_centroids
global last_cow_in_time, last_enter_time, notified_final_cow
//กำหนดตำแหน่งเส้นแนวนอน
line_y = 240
//ลูปรันตรวจจับวัวตลอดเวลา
while True:
if last_frame_camera2 is not None:
//ย่อขนาดภาพให้เหมาะกับโมเดล
resized = cv2.resize(last_frame_camera2, (512, 512))
// ส่งภาพให้โมเดล Roboflow ตรวจจับวัว
```

```
 try:
results = model.predict(resized, confidence=30, overlap=20).json()
//นับจำนวนวัว + หาตำแหน่งวัว
cow_count_camera2 = sum(1 for r in results['predictions'] if r['class'] == 'cow')
current_centroids = [(int(obj["x"]), int(obj["y"])) for obj in results["predictions"]]
//ตรวจวัว "เดินเข้า" (จากตำแหน่งก่อนหน้าถึงปัจจุบัน)
for curr in current_centroids:
for prev in prev_centroids:
if prev[1] < line_y and curr[1] >= line_y:
cow_entered += 1
last_cow_in_time = datetime.now().strftime("%H:%M:%S")
 last_enter_time = time.time()
notified_final_cow = False
break
 //อัปเดตค่าตำแหน่งวัวสำหรับเฟรมต่อไป
prev_centroids = current_centroids
//ตรวจจับข้อผิดพลาดและจัดการ "แจ้งเตือนสุดท้าย"
except Exception as e:
print("Roboflow detection error:", e)
if last_enter_time and not notified_final_cow:
time_since_last_enter = time.time() - last_enter_time
if time_since_last_enter >= 30:
notified_final_cow = True
//รอ 0.2 วินาทีก่อนวนลูปรอบถัดไป
time.sleep(0.2)
//การโหลดและบันทึกค่า "เกณฑ์วัวขั้นต่ำและสูงสุด"
```

```
def load_thresholds():
if os.path.exists(THRESHOLD_FILE):
with open(THRESHOLD_FILE, "r") as f:
data = json.load(f)
return data.get("min", 1), data.get("max", 5)
return 1, 5
//บันทึกค่าความคาดหวังจำนวนวัวขั้นต่ำและสูงสุด ลงในไฟล์ cow_thresholds.json
def save_thresholds(min_val, max_val):
with open(THRESHOLD_FILE, "w") as f:
json.dump({"min": min_val, "max": max_val}, f)
//โหลดค่า "min" และ "max" มาเก็บในตัวแปร cow_min_threshold
cow_min_threshold, cow_max_threshold = load_thresholds()
//ตัวแปรที่เกี่ยวข้องกับการแจ้งเตือน
last_sent_time = 0
alert_interval = 60
//ฟังก์ชันลบวิดีโอบันทึกเก่า
def delete_old_recordings():
folder = "recordings"
now = time.time()
cutoff = now - (7 * 86400)
for filename in os.listdir(folder):
file_path = os.path.join(folder, filename)
if os.path.isfile(file_path):
if os.path.getmtime(file_path) < cutoff:
try:
 os.remove(file_path)
```

```
print(f"Deleted file: {file_path}")
except Exception as e:
print(f" Could not delete file {file_path}: {e}")
//อ่านภาพจากกล้อง 1
def generate_frames_camera1():
//ตัวแปร out จะเก็บ VideoWriter object เพื่อใช้บันทึกวิดีโอ
out = None
//วนลูปอ่านภาพจากกล้อง 1  ถ้าอ่านไม่สำเร็จให้ออกจากลูป
while True:
success, frame = camera1.read()
if not success:
break
//สร้าง VideoWriter เมื่อได้เฟรมแรก
if out is None:
height, width = frame.shape[:2]
filename = datetime.now().strftime("recordings/%Y-%m-%d.mp4")
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter(filename, fourcc, 20.0, (width, height))
//เขียนภาพจากกล้องลงไฟล์ .mp4
out.write(frame)
//แปลงภาพเป็น JPEG และส่งเป็น Live Stream
_, buffer = cv2.imencode('.jpg', frame)
frame_bytes = buffer.tobytes()
yield (b'--frame\r\n'
b'Content-Type: image/jpeg\r\n\r\n' + frame_bytes + b'\r\n')
```

```
//ดึงภาพจากกล้อง 2
def generate_frames_camera2()
//การตั้งค่าตัวแปรชี้เก็บ เฟรมล่าสุดจากกล้อง 2
global last_frame_camera2
//วนลูปไม่สิ้นสุดเพื่อดึงภาพจากกล้อง 2
while True:
success, frame = camera2.read()
if not success:
continue
//ทำสำเนาภาพที่อ่านได้เก็บไว้ในตัวแปร
 last_frame_camera2 = frame.copy()
//วาดเส้นแนวนอนตรงกลางภาพ
cv2.line(frame, (0, frame.shape[0]//2), (frame.shape[1], frame.shape[0]//2), (255, 0, 0), 2)
//แสดงจำนวนวัวที่เข้า ข้อความบนภาพว่า IN: <จำนวนวัวที่เข้า>
cv2.putText(frame, f"IN: {cow_entered}", (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1.0, (0, 255, 0), 2)
//แปลงภาพเป็น JPEG แล้วส่งออกแบบ MJPEG
ret, buffer = cv2.imencode('.jpg', frame)
frame = buffer.tobytes()
yield (b'--frame\r\n'
b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
//หน้าเว็บหลักของแอป Flask
@app.route('/')
def index():
html = '''
<html>
```

```
<head><title>SmartFarm Monitor</title></head>
<h1>Relay Control</h1>
<form method="post" action="/control">
//ส่วนของปุ่ม Relay ควบคุมรีเลย์ 8 ช่อง เป็น Jinja2 Template
{% for i in range(8) %}
 <p>
Relay {{i + 1}}:
{% if relay_status[i] %}
<button name="relay" value="{{i}}" style="background-color: green; color:
white;">ON</button>
{% else %}
<button name="relay" value="{{i}}" style="background-color: red; color:
white;">OFF</button>
{% endif %}
</p>
{% endfor %}
</form>
//ส่วนนี้คือฟอร์มสำหรับตั้งเวลาการทำงานของรีเลย์ (Schedule)
<hr>
<h2>Set Time</h2>
<form method="post" action="/schedule">
Relay:
<select name="relay">
{% for i in range(8) %}
<option value="{{i}}">{{i + 1}}</option>
{% endfor %}
```

```
</select><br><br>
ON Time: <input type="time" name="on_time" required><br><br>
OFF Time: <input type="time" name="off_time" required><br><br>
Loop:
<select name="loop">
<option value="everyday">Everyday</option>
<option value="once">Once</option>
</select><br><br>
<button type="submit">Add Schedule</button>
</form>
//ส่วนนี้คือฟอร์มสำหรับตั้งค่าขั้นต่ำและขั้นสูงของจำนวนวัว (Thresholds
<hr>
<h2>Set Cow</h2>
<form method="post" action="/set_threshold">
Min cows: <input type="number" name="min" value="{{ cow_min }}"><br><br>
Max cows: <input type="number" name="max" value="{{ cow_max }}"><br><br>
<button type="submit">Update Threshold</button>
</form>
//ส่วนนี้คือแสดงรายการงานที่ตั้งเวลาไว้ (Scheduled Tasks) สำหรับรีเลย์แต่ละตัว
<hr>
<h2>Time Tasks</h2>
<ul>
{% for sched in schedules %}
<li>Relay {{sched.relay | int + 1}} - ON at {{sched.on_time}}, OFF at {{sched.off_time}}
({{sched.loop}})</li>
{% endfor %}
```

```
</ul>
//ส่วนนี้คือแสดงภาพกล้อง1ถ่ายทอดสด พร้อมปุ่มรีเซ็ตนับจำนวนวัวที่เข้ามา
<hr>
<h2>Live Camera (Camera 1)</h2>
<img src="/video_feed" width="640" height="480">
<h2>Cow Entry (Camera 2)</h2>
<img src="/counter_feed" width="640" height="480">
//ส่วนนี้คือปุ่มรีเซ็ตนับจำนวนวัวที่เข้ามา
<h2>
<form action="/reset_cow_entered" method="post">
<button type="submit">Reset Cow Entry Count</button>
</form>
//ส่วนนี้คือแสดงรายการวิดีโอที่บันทึกไว้ (Recorded Videos)
<hr>
<h2>Recorded Videos</h2>
<ul>
 {% for video in videos %}
<li><a href="/recordings/{{ video }}">{{ video }}</a></li>
{% endfor %}
</ul>
//โค้ด JavaScript สำหรับตรวจจับเหตุการณ์วัวเข้าใหม่โดยดึงข้อมูลจากเซิร์ฟเวอร์
<script>
let prevIn = 0;
let finalCowNotified = false;
function pollCowEvent() {
fetch('/cow_event')
```

```
.then(response => response.json())
.then(data => {
const entered = data.cow_in;
if (entered > prevIn) {
  prevIn = entered;
finalCowNotified = false;
}
//ตรวจสอบว่ามีเหตุการณ์วัวตัวสุดท้ายเข้ามา
if (data.final_cow_triggered && !finalCowNotified) {
alert("Cow entered at: " + data.last_in + " (Total entered: " + entered + ")");
finalCowNotified = true;
}
//แสดงจำนวนวัวที่เข้าล่าสุดแบบเรียลไทม์ใน elemen
document.getElementById("cow_status").innerText =
Entered: ${entered} | Exited: ${data.cow_out}`;
});
}
//หน้าเว็บอัปเดตข้อมูลวัวแบบเรียลไทม์ทุก 2 วินาที
setInterval(pollCowEvent, 2000);
//แสดงหน้าเว็บ ตารางเวลารีเลย์ วิดีโอที่บันทึก สถานะรีเลย์ และการแจ้งเตือนเมื่อนับวัวเกินหรือต่ำกว่าค่าที่ตั้งไว้
 </script>
</body>
</html>
'''
schedules = scheduler.load_schedules()
```

```
videos = sorted(os.listdir("recordings"), reverse=True)
alert = ""
current_cows = cow_entered - cow_exited
if current_cows < cow_min_threshold or current_cows > cow_max_threshold:
alert = f"Cow count out of range! Current: {current_cows}, Expected:
{cow_min_threshold}-{cow_max_threshold}"
relay_status = [rc.get_status(i) for i in range(8)]
return render_template_string(
html,
schedules=schedules,
videos=videos,
cow_in=cow_entered,
relay_status=relay_status,
alert=alert,
cow_min=cow_min_threshold,
cow_max=cow_max_threshold,
cow_count_camera2=cow_count_camera2)
//URL /video_feed ให้บริการภาพจาก กล้อง 1
@app.route('/video_feed')
def video_feed():
return Response(generate_frames_camera1(), mimetype='multipart/x-mixed-replace;
boundary=frame')
//แสดงภาพจาก กล้อง 2 ที่ใช้ตรวจจับวัวเข้า-ออก
@app.route('/counter_feed')
def counter_feed():
```

```
return Response(generate_frames_camera2(), mimetype='multipart/x-mixed-replace;
boundary=frame')
//backend สำหรับการควบคุมรีเลย์ (relay) แบบเปิด/ปิดจากหน้าเว็บ โดยใช้ Flas
@app.route('/control', methods=['POST'])
def control():
index = int(request.form['relay'])
if rc.get_status(index):
rc.turn_off(index)
else:
rc.turn_on(index)
return redirect('/')
//ฟังก์ชันนี้ใช้สำหรับ ตั้งเวลาการทำงานของรีเลย์ ผ่านเว็บอินเทอร์เฟซ โดยผู้ใช้สามารถระบุเวลาที่จะ
เปิด/ปิดแต่ละรีเลย์ พร้อมเลือกว่าจะทำซ้ำทุกวันหรือครั้งเดียว
@app.route('/schedule', methods=['POST'])
def schedule():
on_time = request.form['on_time']
off_time = request.form['off_time']
relay = request.form['relay']
loop = request.form['loop']
schedules = scheduler.load_schedules()
schedules.append({
"relay": relay,
"on_time": on_time,
"off_time": off_time,
"loop": loop})
scheduler.save_schedules(schedules)
```

```
return redirect('/')
//ฟังก์ชันนี้ใช้สำหรับ กำหนดช่วงจำนวนวัวขั้นต่ำและสูงสุด ที่ระบบจะใช้ในการแจ้งเตือนว่ามีวัวเข้า/ออก
@app.route('/set_threshold', methods=['POST'])
def set_threshold():
global cow_min_threshold, cow_max_threshold
cow_min_threshold = int(request.form['min'])
cow_max_threshold = int(request.form['max'])
save_thresholds(cow_min_threshold, cow_max_threshold)
return redirect('/')
//ฟังก์ชัน @app.route('/cow_event') นี้เป็น API endpoint ที่ส่งข้อมูลแบบ JSON ไปให้ JavaScript
ฝั่งหน้าเว็บ เพื่ออัปเดตสถานะการเข้า-ออกของวัวแบบเรียลไทม์
@app.route('/cow_event')
def cow_event():
return jsonify({
'cow_in': cow_entered,
'cow_out': cow_exited,
 'last_in': last_cow_in_time,
'last_out': last_cow_out_time,
'final_cow_triggered': notified_final_cow })
//ฟังก์ชันนี้คือ route สำหรับ รีเซ็ตตัวนับจำนวนวัวที่เข้า (cow_entered) ให้กลับเป็นศูนย์จากหน้าเว็บ
โดยใช้ปุ่มกดใน HTML ที่คุณกำหนดไว้
@app.route('/reset_cow_entered', methods=['POST'])
def reset_cow_entered():
global cow_entered
cow_entered = 0
return redirect('/')
```

```
//ฟังก์ชันนี้คือดาวน์โหลดไฟล์วิดีโอบันทึก (recordings)
@app.route('/recordings/<filename>')
def download_recording(filename):
return send_from_directory('recordings', filename)
//เรียกฟังก์ชันลบไฟล์ที่เก่ากว่า 7 วันในโฟลเดอร์ recordings
if __name__ == '__main__':
delete_old_recordings()
//เริ่มการตรวจจับวัวจากกล้อง 2 ใน background (thread แยก)
Thread(target=detect_cow_from_camera2, daemon=True).start()
//รันเว็บเซิร์ฟเวอร์ Flask บน IP ใช้พอร์ต 5000
 app.run(host='0.0.0.0', port=5000)
```

ประวัติผู้จัดทำปริญญานิพนธ์

ประวัติผู้แต่ง

ชื่อ	: นายยุทธนา เอียดมาก
สาขา	: ภาควิชาวิศวกรรมไฟฟ้า
วัน-เดือน-ปีเกิด	: วันที่ 24 สิงหาคม 2545
สถานที่เกิด	: จังหวัดนครศรีธรรมราช
ที่อยู่	: 223 หมู่ที่ 6 ตำบลสวนขัน อำเภอช้างกลาง จังหวัดนครศรีธรรมราช 80250
ประวัติการศึกษา	: ประกาศนียบัตรวิชาชีพ วิทยาลัยเทคนิคนครศรีธรรมราช ปี 2564

ประวัติผู้แต่ง

ชื่อ	: นายกฤตพล ยิ้มละมัย
สาขา	: ภาควิชาวิศวกรรมไฟฟ้า
วัน-เดือน-ปีเกิด	: วันที่ 28 กันยายน 2541
สถานที่เกิด	: จังหวัดสงขลา
ที่อยู่	: 109/4 หมู่ที่ 1 ตำบลท่าบอน อำเภอระโนด จังหวัดสงขลา 90140
ประวัติการศึกษา	: มัธยมศึกษาปีที่6 โรงเรียนมหาวชิราวุธ ปี 2560