โครงงาน สาขา วิทยาศาสตร์และเทคโนโลยี

Smart Home Monitor

โดย

นายพัฒนพล สุธรรม	ชั้น	ม.3/1 เลขที่ 19
เด็กชายภาณุวัฒน์ เทียนทอง	ชั้น	ม.3/1 เลขที่ 21
เด็กชายรัชพล บุญทร	ชั้น	ม.3/1 เลขที่ 22

ครูที่ปรึกษา

นายพิทักษ์ วงษ์รีย์

โรงเรียนพิริยาลัยจังหวัดแพร่ ตำบลในเวียง อำเภอเมือง จังหวัดแพร่

สำนักงานเขตพื้นที่การศึกษามัธยมศึกษาแพร่

สำนักงานคณะกรรมการการศึกษาขั้นพื้นฐาน

กระทรวงศึกษาธิการ

โครงงาน สาขา วิทยาศาสตร์และเทคโนโลยี

Smart Home Monitor

โดย

นายพัฒนพล สุธรรม ชั้น ม.3/1 เลขที่ 19

เด็กชายภาณุวัฒน์ เทียนทอง ชั้น ม.3/1 เลขที่ 21

เด็กชายรัชพล บุญทร ชั้น ม.3/1 เลขที่ 22

ครูที่ปรึกษา

นายพิทักษ์ วงษ์รีย์

โครงงานนี้เป็นส่วนหนึ่งของการเรียนในรายวิชาการโปรแกรม 5 ปีการศึกษา 2568

ตามหลักสูตร สถาบันส่งเสริมการสอนวิทยาศาสตร์และเทคโนโลยี (สสวท.)

ชื่อโครงงาน Smart Home Monitor

ผู้จัดทำโครงงาน 1. นายพัฒนพล สุธรรม ชั้น ม.3/1 เลขที่ 19

3. เด็กชายภาณุวัฒน์ เทียนทอง ชั้น ม.3/1 เลขที่ 21

3. เด็กชายรัชพล บุญทร ชั้น ม.3/1 เลขที่ 22

โรงเรียน พิริยาลัยจังหวัดแพร่

ครูที่ปรึกษา นายพิทักษ์ วงษ์รีย์

บทคัดย่อ

โครงงานนี้มีวัตถุประสงค์เพื่อออกแบบและพัฒนาระบบ Smart Home Monitor ที่สามารถตรวจสอบและเฝ้าระวังสภาพแวดล้อมภายในบ้านได้แบบเรียลไทม์ โดยใช้บอร์ดไมโครคอนโทรลเลอร์ ESP32 ร่วมกับ ESP32-CAM และเซ็นเซอร์หลายชนิด ได้แก่ DHT22 สำหรับตรวจวัดอุณหภูมิและความชื้น, MQ-2 สำหรับตรวจจับแก๊สไวไฟ, PIR สำหรับตรวจจับการเคลื่อนไหว, ปุ่มกดสำหรับนับจำนวนคน และโมดูล LCD สำหรับแสดงผลลัพธ์ทันที ระบบสามารถแจ้งเตือนผู้ใช้งานผ่านแอปพลิเคชัน Telegram

ผลการทดสอบเบื้องต้นพบว่าระบบสามารถตรวจจับเหตุการณ์ต่าง ๆ และส่งการแจ้งเตือนได้ภายในเวลาประมาณไม่กี่วินาที ทั้งยังสามารถสั่งงานผ่าน Telegram เพื่อขอข้อมูลสถานะปัจจุบัน เช่น จำนวนคน ค่าอุณหภูมิ ความชื้น และปริมาณก๊าซ รวมถึงสั่งให้ ESP32-CAM ถ่ายภาพและส่งกลับมาได้ ระบบดังกล่าวจึงเหมาะสมที่จะนำไปประยุกต์ใช้เป็นต้นแบบของระบบเฝ้าระวังภายในบ้านหรือสำนักงาน โดยในอนาคตสามารถปรับปรุงด้านความแม่นยำของการตรวจจับและเพิ่มการประมวลผลภาพเพื่อให้รองรับการใช้งานจริงได้ดียิ่งขึ้น

กิตติกรรมประกาศ

โครงงานคอมพิวเตอร์เรื่อง Smart Home Monitor จัดทำขึ้นเพื่อศึกษาเกี่ยวกับการทำระบบดูแลบ้าน โดยได้รับการสนับสนุนจาก คุณครูพิทักษ์ วงษ์รีย์ คุณครูที่ปรึกษาที่ได้ให้คำแนะนำปรึกษา และให้ข้อมูลต่างๆ ขอขอขอบพระคุณที่ได้ให้คำปรึกษา

อนึ่ง คณะผู้จัดทำหวังว่า โครงงานวิทยาศาสตร์ เรื่อง Smart Home Monitor จะมีประโยชน์อยู่ไม่น้อย จึงขอมอบส่วนดีทั้งหมดนี้ให้แก่เหล่าครูบาอาจารย์ที่ได้ประสิทธิ์ประสาทวิชาจนทำให้ผลงานวิจัยเป็นประโยชน์ต่อผู้เกี่ยวข้องและขอมอบความกตัญญูกตเวทิตาคุณแด่บิดา มารดา และผู้มีพระคุณทุกท่านตลอดจนเพื่อนๆ ที่คอยให้ความช่วยเหลือและกำลังใจ

คณะผู้จัดทำ

18 กุมภาพันธ์ 2568

สารบัญ

สารบัญรูป

บทที่ 1
บทนำ

ที่มาและความสำคัญ

ในปัจจุบัน เทคโนโลยีสมาร์ทโฮม (Smart Home) และระบบอินเทอร์เน็ตของสรรพสิ่ง (Internet of Things: IoT) ได้เข้ามามีบทบาทสำคัญในชีวิตประจำวันของผู้คนมากขึ้น เนื่องจากช่วยเพิ่มความสะดวกสบาย ความปลอดภัย และการตรวจสอบสภาพแวดล้อมภายในบ้านได้อย่างต่อเนื่อง เจ้าของบ้านสามารถรับข้อมูลและควบคุมอุปกรณ์ต่าง ๆ ได้จากระยะไกลผ่านเครือข่ายอินเทอร์เน็ต ทำให้การอยู่อาศัยมีความมั่นใจและปลอดภัยมากยิ่งขึ้น

อย่างไรก็ตาม ระบบสมาร์ทโฮมที่มีจำหน่ายทั่วไปมักมีราคาสูง และอาจไม่ตอบโจทย์ความต้องการของผู้ใช้งานบางกลุ่ม อีกทั้งยังไม่สามารถปรับแต่งหรือเพิ่มเติมฟังก์ชันการทำงานได้ตามสถานการณ์จริง ดังนั้น การพัฒนาโครงงานต้นแบบระบบ Smart Home Monitor จึงมีความสำคัญ เนื่องจากเป็นการออกแบบและสร้างระบบตรวจสอบและควบคุมบ้านที่มีความยืดหยุ่น ต้นทุนไม่สูง และสามารถปรับปรุงเพิ่มเติมได้ตามความต้องการของผู้ใช้

ดังนั้น โครงงานนี้จึงมีความสำคัญทั้งในด้านการเสริมสร้างความปลอดภัยภายในบ้าน การเฝ้าระวังสภาพแวดล้อม และการอำนวยความสะดวกแก่เจ้าของบ้าน อีกทั้งยังเป็นการประยุกต์ใช้เทคโนโลยี IoT เพื่อพัฒนาเป็นต้นแบบของระบบสมาร์ทโฮมที่มีความทันสมัย ปรับใช้ได้จริง และสามารถต่อยอดสู่การพัฒนาระบบบ้านอัจฉริยะที่สมบูรณ์ยิ่งขึ้นในอนาคต

วัตถุประสงค์

1. เพื่อออกแบบและพัฒนาระบบ Smart Home Monitor ที่สามารถตรวจสอบและแจ้งเตือนข้อมูลต่าง ๆ ภายในบ้านได้
2. เพื่อพัฒนาระบบควบคุมระบบระยะไกลผ่าน Telegram

ตัวแปร

ตัวแปรต้น : ค่าที่ตรวจวัดจากเซ็นเซอร์ต่าง ๆ ได้แก่ อุณหภูมิ ความชื้น และก๊าซ
การเคลื่อนไหวที่ตรวจจับได้จากกล้อง ESP32-CAM
สถานะการเข้าออกของบุคคล (จากเซ็นเซอร์สัมผัส)

ตัวแปรตาม : การแจ้งเตือนผ่าน Telegram (ข้อความ, ภาพถ่าย, สัญญาณเตือนเสียง/ไฟ)
การควบคุมสถานะอุปกรณ์ต่าง ๆ (ไฟ, สัญญาณเสียง)

ตัวแปรควบคุม : สภาพแวดล้อมการทดสอบ เช่น พื้นที่ภายในบ้านหรือห้องทดลอง
อุปกรณ์และซอฟต์แวร์ที่ใช้ เช่น บอร์ด ESP32, ESP32-CAM, Telegram Bot

นิยามศัพท์เฉพาะ

Smart Home Monitor หมายถึงระบบที่สร้างขึ้นเพื่อใช้ในการตรวจสอบและเฝ้าระวังสภาพแวดล้อมภายในบ้าน โดยอาศัยเทคโนโลยีไมโครคอนโทรลเลอร์และอินเทอร์เน็ตของสรรพสิ่ง (IoT) เพื่อรวบรวมข้อมูลจากเซ็นเซอร์และกล้อง พร้อมทั้งแจ้งเตือนผู้ใช้งานผ่านเครือข่ายอินเทอร์เน็ต

เทคโนโลยี Internet of Things (IoT) คือเทคโนโลยีที่เชื่อมต่ออุปกรณ์ต่างๆเข้ากับระบบอินเทอร์เน็ตเพื่อให้สามารถส่งข้อมูลระหว่างกันและควบคุมจากระยะไกล

การแจ้งเตือนผ่าน Telegram คือการส่งข้อความหรือข้อมูลแจ้งเตือนไปยังผู้ใช้งานผ่านแอปพลิเคชัน Telegram โดยอาศัยเทคโนโลยีการเชื่อมต่อระหว่างระบบควบคุม กับ Telegram Bot API เพื่อส่งข้อมูลในรูปแบบข้อความ รูปภาพ หรือข้อมูลอื่นๆ ในเวลาจริง (real-time) เพื่อแจ้งเตือนเหตุการณ์หรือสถานะที่สำคัญ

ขอบเขตของการศึกษา

1. โครงงานนี้มุ่งเน้นการพัฒนา ต้นแบบระบบสมาร์ทโฮม (Home Smart Monitor) โดยใช้อุปกรณ์ไมโครคอนโทรลเลอร์ ESP32 และ ESP32-CAM เป็นหลัก
2. ระบบสามารถตรวจวัดข้อมูลจากเซ็นเซอร์ ได้แก่ อุณหภูมิ ความชื้น และก๊าซ รวมถึงตรวจจับการเคลื่อนไหวและจำนวนบุคคลที่เข้าออกบ้านได้
3. การแจ้งเตือนและควบคุมระบบจะกระทำผ่านแอปพลิเคชัน Telegram เท่านั้น ไม่ได้พัฒนาเป็นแอปพลิเคชันเฉพาะของโครงงาน
4. ระบบต้นแบบนี้ออกแบบสำหรับใช้งานภายในบ้านหรือพื้นที่ขนาดเล็ก ไม่ครอบคลุมการใช้งานเชิงพาณิชย์หรือพื้นที่ขนาดใหญ่
5. โครงงานนี้เน้นการพัฒนาและทดสอบในระดับต้นแบบ (Prototype) โดยไม่ได้ครอบคลุมถึงการออกแบบเชิงอุตสาหกรรมหรือการผลิตจริง

ประโยชน์ที่คาดว่าจะได้รับ

1. ได้รับความรู้จากการประยุกต์ใช้เทคโนโลยี IoT ในการนำมาสร้างระบบ
2. เจ้าของบ้านสามารถรับข้อมูลและควบคุมระบบจากระยะไกลผ่านแอปพลิเคชัน Telegram ทำให้สามารถเฝ้าระวังบ้านได้ตลอดเวลา ไม่ว่าจะอยู่ในสถานที่ใด
3. สามารถช่วยลดความเสี่ยงจากอันตรายที่อาจเกิดขึ้นในบ้าน เช่น ก๊าซรั่ว อุณหภูมิสูงเกินกำหนด หรือความเคลื่อนไหวที่ผิดปกติ
4. ผู้จัดทำโครงงานได้รับความรู้และทักษะในการประยุกต์ใช้เทคโนโลยี IoT การเขียนโปรแกรมไมโครคอนโทรลเลอร์ การเชื่อมต่อเซ็นเซอร์ และการทำงานร่วมกับแพลตฟอร์มออนไลน์
5. สามารถต่อยอดโครงงานไปสู่การพัฒนาเป็นระบบสมาร์ทโฮมที่มีความสมบูรณ์ยิ่งขึ้น และอาจขยายผลไปใช้เชิงพาณิชย์ได้ในอนาคต
6. เป็นตัวอย่างในการประยุกต์ใช้เทคโนโลยีเพื่อการแก้ปัญหาในชีวิตประจำวัน และช่วยกระตุ้นให้เกิดการเรียนรู้ด้านนวัตกรรมและการพัฒนาระบบอัจฉริยะ

บทที่ 2
เอกสารและงานวิจัยที่เกี่ยวข้อง

การทำโครงงานเรื่อง Smart Home Monitor คณะผู้จัดทำได้ศึกษาแนวคิดและเอกสารที่เกี่ยวข้อง โดย นำเสนอตามลำดับ ดังนี้

1.Internet of Things หรือ IoT คืออะไร

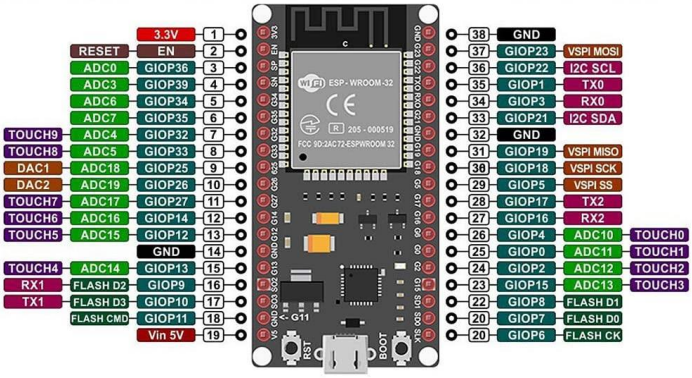
มัทนา วิบูลย์ยะศักดิ์ (ม.ป.ป.) ได้ให้ความหมายของ Internet of Things (IoT) ไว้ว่าเป็นการที่อุปกรณ์อิเล็กทรอนิกส์ต่างๆ สามารถเชื่อมโยงหรือส่งข้อมูลถึงกันได้ด้วยอินเทอร์เน็ต โดยไม่ต้องป้อนข้อมูล การเชื่อมโยงนี้ง่ายจนทำให้เราสามารถสั่งการควบคุมการใช้งานอุปกรณ์อิเล็กทรอนิกส์ต่างๆ ผ่านทางเครือข่ายอินเตอร์เน็ตได้ ไปจนถึงการเชื่อมโยงการใช้งานอุปกรณ์อิเล็กทรอนิกส์ต่างๆ ผ่านทางเครือข่ายอินเตอร์เน็ตเข้ากับการใช้งานอื่นๆ จนเกิดเป็นบรรดา Smart ต่างๆ ได้แก่ Smart Device, Smart Grid, Smart Home, Smart Network, Smart Intelligent Transportation ทั้งหลายที่เราเคยได้ยินนั่นเอง ซึ่งแตกต่างจากในอดีตที่อุปกรณ์อิเล็กทรอนิกส์เป็นเพียงสื่อกลางในการส่งและแสดงข้อมูลเท่านั้น

กล่าวได้ว่า Internet of Things นี้ได้แก่การเชื่อมโยงของอุปกรณ์อัจฉริยะทั้งหลายผ่านอินเทอร์เน็ตที่เรานึกออก เช่น แอปพลิเคชัน แว่นตากูเกิลกลาส รองเท้าวิ่งที่สามารถเชื่อมต่อข้อมูลการวิ่ง ทั้งความเร็ว ระยะทาง สถานที่ และสถิติได้

นอกจากนั้น Cloud Storage หรือ บริการรับฝากไฟล์และประมวลผลข้อมูลของคุณผ่านทางออนไลน์ หรือเราเรียกอีกอย่างว่า แหล่งเก็บข้อมูลบนก้อนเมฆ เป็นอีกสิ่งหนึ่งที่เราใช้งานบ่อยๆแต่ไม่รู้ว่าเป็นหนึ่งในรูปแบบของ Internet of Things สมัยนี้ผู้ใช้นิยมเก็บข้อมูลไว้ในก้อนเมฆมากขึ้น เนื่องจากมีข้อดีหลายประการ คือ ไม่ต้องกลัวข้อมูลสูญหายหรือถูกโจรกรรม ทั้งยังสามารถกำหนดให้เป็นแบบส่วนตัวหรือสาธารณะก็ได้ เข้าถึงข้อมูลได้ทุกที่ทุกเวลาด้วยอุปกรณ์อิเล็กทรอนิกส์ใดๆผ่านเครือข่ายอินเตอร์เน็ต แถมยังมีพื้นที่ใช้สอยมากมีให้เลือกหลากหลาย ช่วยเราประหยัดค่าใช้จ่ายได้อีกด้วย เนื่องจากเราไม่ต้องเสียเงินซื้ออุปกรณ์จัดเก็บข้อมูล เช่น ฮาร์ดไดร์ฟ หรือ Flash drive ต่างๆ

เทคโนโลยี Internet of Things มีประโยชน์ในหลายด้านทั้งเรื่องการเก็บข้อมูลที่แม่นยำและเป็นปัจจุบัน ช่วยลดต้นทุน แถมยังช่วยเพิ่มผลผลิตของพนักงานหรือผู้ใช้งานได้ แม้ว่าแนวโน้มของ IoT มีแต่จะเพิ่มขึ้นด้วยคุณาประโยชน์ตามที่ได้กล่าวมาแล้ว แต่ประโยชน์ใดๆนั้นก็มาพร้อมกับความเสี่ยง เพราะความท้าทายในการรักษาความปลอดภัยของเครือข่ายใหม่ที่เกิดขึ้นนั้น จะผลักดันให้ผู้เชี่ยวชาญมีการรับมือทางด้านความปลอดภัยมากขึ้น ในทางตรงกันข้ามแฮกเกอร์หรือผู้ไม่หวังดีก็ทำงานหนักเพื่อที่จะเข้าควบคุม โจมตีเครือข่าย หรือเรียกค่าไถ่ในช่องโหว่ที่ IoT มีอยู่ ฉะนั้นผู้เชี่ยวชาญด้านความปลอดภัยทาง IoT จึงจำเป็นต้องพัฒนามาตรการ และระบบรักษาความปลอดภัยไอทีควบคู่กันไป เพื่อให้ธุรกิจและการใช้งาน IoT สามารถขับเคลื่อนต่อไปได้

2. ไมโครคอนโทรลเลอร์ ESP32



รูป 1 ไมโครคอนโทรลเลอร์ ESP32 DEVKIT V1

(ที่มา: GROBOTRONICS, ม.ป.ป.)

ศุภรัตน์ แย้มครวญ (2566) ได้อธิบายบอร์ด อีเอสพี32 (ESP32) ดังรูป 1 ไว้ว่าเป็นบอร์ดไมโครคอนโทรลเลอร์ราคาไม่สูง ใช้พลังงานต่ำ และพร้อมกับการเชื่อมต่อ Wi-Fi และ Bluetooth ซึ่งเป็นการพัฒนาจากรุ่นก่อนหน้า ESP8266 โดยใช้ระบบ System-on-chip (SoC) ที่ถูกพัฒนาจาก Espressif Systems

ด้วยองค์ประกอบของ ESP32 ที่มาพร้อมกับสายอากาศในตัวเอง (Antenna Integration), RF balun, การขยายกำลังส่ง, การขยายสัญญาณรบกวนต่ำ, ตัวกรอง, และโมดูลการจัดการพลังงาน โดยทั้งหมดนี้จะใช้พื้นที่บนบอร์ดวงจรพิมพ์น้อยมาก โดยบอร์ด ESP32 นี้ใช้กับชิป Wi-Fi และ Bluetooth แบบคู่ 2.4 กิกะเฮิร์ต และมีเทคโนโลยีการผลิตที่มีพลังงานต่ำของ TSMC ที่ 40 นาโนเมตร ที่มีคุณสมบัติทางพลังงานที่ใช้ต่ำ และมีระบบ RF ซึ่งเป็นเทคโนโลยีที่น่าเชื่อถือ และสามารถปรับขนาดใช้งานได้หลากหลายในแอปพลิเคชันต่างๆ โดยเฉพาะด้านอินเทอร์เน็ตของสรรพสิ่ง (IoT) ESP32 เป็นไมโครคอนโทรลเลอร์ที่ได้รับความนิยมสูงในการพัฒนาอุปกรณ์ IoT เนื่องจากมีคุณสมบัติที่มีประสิทธิภาพด้านพลังงาน และด้วยความสามารถในการเชื่อมต่อ Wi-Fi Bluetooth อย่างคล่องตัว ทำให้นักพัฒนาผสามารถเชื่อมต่อไปยังเครือข่ายไร้สาย สื่อสารกับอุปกรณ์อื่นๆ ได้อย่างง่ายดาย

3. ไมโครคอลโทรลเลอร์ ESP32 Camera

เป็นบอร์ดที่ใช้ชิป ใช้ลักษณะการต่อเหมือนกับบอร์ด ESP32 แต่มาพร้อมกับกล้อง OV2640 และมีช่องเสียบ SD Card ในตัว สามารถเชื่อมต่อ WiFi+Bluetooth เพื่อการควบคุมระยะไกลได้

4. Touch Sensor

th.element14 (ม.ป.ป.) เซนเซอร์สัมผัสคืออุปกรณ์ที่ตรวจจับและบันทึกการสัมผัสหรือการจับถือทางกายภาพของอุปกรณ์ และ/หรือ วัตถุ เซนเซอร์จะช่วยให้อุปกรณ์หรือวัตถุตรวจจับการสัมผัสหรือการอยู่ใกล้ของมนุษย์หรือผู้ใช้งานได้ อุปกรณ์อินพุตการตรวจจับสัมผัสมอบความเป็นไปได้อันหลากหลายเพื่อการใช้งานที่สะดวกสบาย และแทนที่ปุ่มกดเชิงกลและสวิตช์เพื่อกำจัดการสึกหรอทางกลอีกด้วย อุปกรณ์สามารถปรับแต่งเป็นทั้งตัวเลื่อนแบบง่ายๆ ล้อหมุน หรือแม้กระทั่งทัชแพดสำหรับอินเตอร์เฟซผู้ใช้ที่ใช้งานง่าย

เซนเซอร์สัมผัสจะทำงานเมื่อวัตถุหรือบุคคลมีการสัมผัสทางกายภาพ เซนเซอร์สัมผัสหรือเซนเซอร์ไวสัมผัสจะตอบสนองต่อการสัมผัส แรงหรือน้ำหนักกดได้อย่างรวดเร็ว และยังใช้เทคโนโลยีตรวจจับสัมผัสแบบ Capacitive หรือ Resistive ได้ด้วย

เทคโนโลยีการตรวจจับสัมผัสแบบ Capacitive จะใช้คัปปลิ้ง Capacitive ที่จะตรวจจับและวัดทุกสิ่งที่นำกระแสหรือมีไดอิเล็กตริกที่ต่างจากอากาศ จอสัมผัสแบบ Capacitive ตรวจจับและระบุตำแหน่งที่สัมผัสตามแรงกระตุ้นทางไฟฟ้าในร่างกายมนุษย์ ซึ่งปกติแล้วจะเป็นปลายนิ้ว จอสัมผัสแบบ Capacitive จึงไม่ต้องใช้แรงกระทำใดๆ ที่พื้นผิวของหน้าจอ

เทคโนโลยีจอสัมผัสแบบ Capacitive ได้รับความนิยมสูงและมีความทนทาน อีกทั้งยังมีการใช้งานที่หลากหลาย นอกจากนี้จอสัมผัสแบบ Capacitive ยังมีความใสอย่างมาก ซึ่งมีความโปร่งใสถึง 90 เปอร์เซ็นต์ ด้วยความคมชัดที่สูงกว่าเทคโนโลยี Resistive จึงมีการใช้งานกับสมาร์ทโฟนอย่างแพร่หลาย

5. DHT22

cybertice (2568) ได้กล่าวเกี่ยวกับ DHT22 ไว้ว่าเป็นโมดูลเซนเซอร์วัดความชื้นและอุณหภูมิในตัวเดียว มีความแม่นยำสูง มีตัวต้านทาน Pull up มาแล้วสามารถต่อขาทดลองได้เลยไม่ต้องต่อเพิ่ม ถ้าต้องการความถูกต้องแม่นยำในการวัดอุณหภูมิและความชื้น แนะนำตัวนี้เลย DHT22 High Accuracy Digital Temperature and Humidity Sensor DHT22 ใช้สำหรับวัดอุณหภูมิและความชื้น ออกแบบมาให้วัดได้แม่นยำกว่ารุ่น DHT11 ใช้ง่ายสามารถนำ DHT22 ไปเปลี่ยนแทน DHT11 ได้เลยเพราะโค้ด Arduino DHT22 เขียนเหมือนกัน

Accuracy humidity +-2%RH(Max +-5%RH); temperature +-0.2Celsius

Resolution or sensitivity humidity 0.1%RH; temperature 0.1Celsius

Repeatability humidity +-1%RH; temperature +-0.2Celsius

Humidity hysteresis +-0.3%RH

Long-term Stability +-0.5%RH/year

Sensing period Average: 2s

Interchangeability fully interchangeable

Features DHT22:

3.3-6V Input

1-1.5mA measuring current

40-50 uA standby current

Humidity from 0-100% RH

-40 - 80 degrees C temperature range

+-2% RH accuracy

+-0.5 degrees C

6. PIR Sensor

adafruit (2568) ได้อธิบายว่า PIR Sensor เป็นเซนเซอร์ตรวจจับความเคลื่อนไหวแบบอินฟราเรดพาสซีฟ (Passive Infrared Motion Sensor: PIR) เป็นอุปกรณ์อิเล็กทรอนิกส์ที่ใช้ในการตรวจจับการเคลื่อนไหวของสิ่งมีชีวิต โดยอาศัยการเปลี่ยนแปลงของพลังงานรังสีอินฟราเรดที่ปล่อยออกมาจากร่างกายมนุษย์หรือสัตว์ เซนเซอร์ชนิดนี้ประกอบด้วย pyroelectric sensor ซึ่งมีคุณสมบัติในการตรวจจับความเข้มของรังสีอินฟราเรด เมื่อมีการเคลื่อนที่ผ่านบริเวณที่ตรวจจับ จะทำให้ปริมาณรังสีอินฟราเรดที่ตกกระทบกับเซนเซอร์สองส่วนไม่เท่ากัน เกิดการเปลี่ยนแปลง (differential change) ซึ่งถูกแปลงเป็นสัญญาณไฟฟ้าส่งออกมา

เพื่อเพิ่มประสิทธิภาพในการตรวจจับ เซนเซอร์ PIR มักติดตั้งร่วมกับ เลนส์ Fresnel ที่ทำหน้าที่รวมและกระจายแสงอินฟราเรดเข้าสู่ตัวเซนเซอร์ ทำให้สามารถตรวจจับการเคลื่อนไหวได้ในระยะที่ไกลขึ้นและมุมมองที่กว้างขึ้น (โดยทั่วไปประมาณ 6 เมตร ในมุมตรวจจับ 110° × 70°)

คุณสมบัติทั่วไป

1. ขนาดเล็ก ราคาถูก และใช้พลังงานต่ำ
2. ให้สัญญาณดิจิทัลเป็นเอาต์พุต (ค่าระดับสูง/ต่ำ) เมื่อมีหรือไม่มีการเคลื่อนไหว
3. มีความทนทานสูงและไม่สึกหรอง่าย
4. รองรับแรงดันไฟฟ้าอินพุตทั่วไป 5V – 12V (ภายในมีตัวปรับแรงดัน)
5. ใช้งานง่าย สามารถเชื่อมต่อกับไมโครคอนโทรลเลอร์ เช่น Arduino หรือ ESP32 ได้โดยตรง

ข้อจำกัดของเซนเซอร์ PIR

แม้ว่า PIR จะมีประสิทธิภาพสูงในการตรวจจับการเคลื่อนไหว แต่ก็มีข้อจำกัดบางประการ เช่น ไม่สามารถบอกจำนวนคนในพื้นที่ได้อย่างแม่นยำ ไม่สามารถวัดระยะห่างของวัตถุจากเซนเซอร์ได้โดยตรง และอาจเกิดการตรวจจับผิดพลาดจากสัตว์เลี้ยงหรือความร้อนจากสิ่งแวดล้อม

การประยุกต์ใช้งาน

เซนเซอร์ PIR นิยมนำไปใช้ในระบบอัตโนมัติและระบบรักษาความปลอดภัยภายในบ้าน เช่น ไฟส่องสว่างอัตโนมัติ ระบบแจ้งเตือนผู้บุกรุก หรือการควบคุมอุปกรณ์ไฟฟ้าเมื่อมีผู้ใช้งานอยู่ในพื้นที่ นอกจากนี้ยังถูกประยุกต์ใช้ในโครงการด้าน IoT (Internet of Things) เพื่อสร้างระบบ Smart Home ที่สามารถตรวจจับการเคลื่อนไหวและส่งข้อมูลไปยังผู้ใช้งานได้แบบเรียลไทม์

7. Gas Sensor

winsen-sensor (2568) ได้บอกไว้ว่าเซนเซอร์ MQ-2 เป็น เซนเซอร์ก๊าซกึ่งตัวนำ (Semiconductor Gas Sensor) ที่ใช้ ดีบุกออกไซด์ (SnO_2) เป็นวัสดุไวต่อแก๊ส ซึ่งมีค่าการนำไฟฟ้าต่ำในบรรยากาศอากาศสะอาด เมื่อมีแก๊สไวไฟ เช่น โพรเพน (C_3H_8), มีเทน (CH_4), ไฮโดรเจน (H_2), ควันบุหรี่ หรือไอระเหยของแอลกอฮอล์ เข้ามาสัมผัสพื้นผิวของ SnO_2 จะทำให้การนำไฟฟ้าของเซนเซอร์เพิ่มสูงขึ้นตามปริมาณความเข้มข้นของแก๊ส การเปลี่ยนแปลงนี้สามารถวัดค่าได้เป็นสัญญาณไฟฟ้า เพื่อบอกระดับความเข้มข้นของแก๊สในอากาศ

เซนเซอร์ MQ-2 มีวงจรข้างในที่ต้องใช้แรงดันเลี้ยงสองส่วน คือ

1. Heater Voltage (VH) ~ 5V ใช้ให้ความร้อนกับตัวเซนเซอร์เพื่อให้ทำงานได้ในสภาวะเหมาะสม
2. Circuit Voltage (VC) สำหรับตรวจวัดค่าความต้านทานที่เปลี่ยนแปลง

คุณสมบัติทั่วไป

1. ตรวจจับแก๊สไวไฟได้หลายชนิด เช่น โพรเพน มีเทน แก๊สหุงต้ม ควัน และไอระเหยแอลกอฮอล์
2. มีช่วงการตรวจจับกว้าง: ประมาณ 300 – 10,000 ppm
3. มีความไวสูงต่อโพรเพนและควัน
4. ใช้พลังงานต่ำ (การใช้ฮีตเตอร์ ≤ 950 mW)
5. อายุการใช้งานยาวนานและราคาถูก
6. ให้สัญญาณแอนะล็อกเอาต์พุต (แรงดันไฟฟ้า) ที่สามารถนำไปแปลงเป็นค่าความเข้มข้นของแก๊สได้ง่าย

ข้อควรระวังในการใช้งาน

1. ไม่ควรให้เซนเซอร์สัมผัสกับ ไอซิลิโคน (เช่น ซิลิโคนกาว, น้ำยาอุดรอย) เพราะจะทำให้ความไวต่อแก๊สเสียไปอย่างถาวร
2. ไม่ควรใช้ในสภาวะแก๊สกัดกร่อนเข้มข้น (เช่น H_2S, SO_2, Cl_2) เพราะจะทำให้โครงสร้างเซนเซอร์สึกกร่อนและเสื่อมสภาพ
3. หลีกเลี่ยงการสัมผัสน้ำ ความชื้นสูงจัด หรือการเกิดน้ำค้างบนเซนเซอร์
4. หากเก็บไว้นานโดยไม่ใช้งาน จำเป็นต้อง เปิดใช้งาน (aging) นานหลายชั่วโมงก่อนนำมาใช้งานจริง เพื่อให้ค่ามีความเสถียร
5. ควรใช้ในสภาพแวดล้อมทั่วไปที่มีอุณหภูมิห้องและความชื้นปกติ เพื่อความแม่นยำในการตรวจจับ

8. Passive Buzzer

handsontec (ม.ป.ป.) ได้อธิบายว่าบัซเซอร์แบบพาสซีฟ (Passive Buzzer) เป็นอุปกรณ์แปลงสัญญาณไฟฟ้าให้เป็นเสียง โดยอาศัยหลักการสร้างคลื่นเสียงจากการสั่นสะเทือนของแผ่นไดอะแฟรมภายในตัวบัซเซอร์ อุปกรณ์ชนิดนี้ไม่สามารถสร้างเสียงได้ด้วยตัวเอง แต่จะต้องอาศัยสัญญาณกระแสสลับ (AC) หรือสัญญาณพัลส์ความถี่สูงที่ส่งเข้ามาจากภายนอก เช่น สัญญาณ PWM (Pulse Width Modulation) เพื่อบังคับให้แผ่นไดอะแฟรมสั่นตามความถี่ของสัญญาณไฟฟ้า

ดังนั้น เสียงที่ได้จากบัซเซอร์พาสซีฟจึงขึ้นอยู่กับความถี่ของสัญญาณไฟฟ้าที่จ่ายเข้าไป เช่น หากป้อนสัญญาณ PWM ที่มีความถี่ 1.5 – 2.5 kHz บัซเซอร์จะสร้างเสียงที่มีความถี่ใกล้เคียงกับค่านั้น ผู้ใช้สามารถปรับความถี่เพื่อสร้างโทนเสียงที่แตกต่างกันได้

คุณสมบัติทั่วไป

1. ช่วงความถี่การทำงาน: ประมาณ 1.5 – 2.5 kHz
2. แรงดันไฟฟ้าที่รองรับ: 3 – 5V
3. ขนาดเล็ก กินไฟน้อย
4. ขาเชื่อมต่อมาตรฐานระยะ 2.54 มม. สามารถใช้งานร่วมกับบอร์ดไมโครคอนโทรลเลอร์ได้สะดวก
5. มีความทนทานและสามารถใช้งานได้ในช่วงอุณหภูมิกว้าง (-20 ถึง +70 °C)

ข้อแตกต่างจากบัซเซอร์แบบแอคทีฟ (Active Buzzer)

Passive Buzzer: ต้องป้อนสัญญาณความถี่จากภายนอก (เช่น PWM) ถึงจะเกิดเสียง ⟶ สามารถสร้างเสียงหลายโทนได้

Active Buzzer: มีวงจรสร้างความถี่ในตัวเอง เพียงจ่ายไฟตรงก็สามารถส่งเสียงได้ทันที แต่จะได้เพียงเสียงโทนเดียว

9. การแจ้งเตือนผ่าน Telegram

การแจ้งเตือนผ่าน Telegram โดยใช้ ESP32 เป็นการนำความสามารถของไมโครคอนโทรลเลอร์ ESP32 มาผสานกับแพลตฟอร์มการสื่อสาร Telegram ซึ่งช่วยให้สามารถส่งข้อความแจ้งเตือนหรือข้อมูลจากเซ็นเซอร์ไปยังผู้ใช้งานได้แบบเรียลไทม์

Telegram เป็นแอปพลิเคชันส่งข้อความที่รองรับการสร้างบอทเพื่ออัตโนมัติในกระบวนการส่งข้อความ โดยกระบวนการนี้ประกอบด้วย 5 ขั้นตอนหลักดังนี้

1. การสร้างบอทใน Telegram
    - เปิดแอป Telegram และค้นหา "BotFather" ซึ่งเป็นบอทที่ใช้สำหรับสร้างและจัดการบอทอื่น ๆ
    - เริ่มการสนทนากับ BotFather และพิมพ์คำสั่ง /newbot เพื่อสร้างบอทใหม่
    - ตั้งชื่อและกำหนดชื่อผู้ใช้ (username) ให้กับบอท โดยชื่อผู้ใช้ต้องลงท้ายด้วยคำว่า "bot" เช่น MyESP32Bot
    - หลังจากสร้างบอทเสร็จสิ้น BotFather จะส่ง โทเค็น (Token) ซึ่งเป็นรหัสสำหรับเข้าถึงบอท
2. การรับ Chat ID ของผู้ใช้
    - ค้นหาและเริ่มการสนทนากับบอทชื่อ "IDBot" ใน Telegram
    - พิมพ์คำสั่ง /getid เพื่อรับ Chat ID ของคุณ ซึ่งจำเป็นสำหรับการส่งข้อความไปยังผู้ใช้
3. การตั้งค่า ESP32
    - ติดตั้งไลบรารี Universal-Arduino-Telegram-Bot ใน Arduino IDE โดยสามารถดาวน์โหลดได้จาก GitHub
    - ตั้งค่า Wi-Fi SSID และรหัสผ่าน รวมถึงระบุโทเค็นและ Chat ID ที่ได้รับในขั้นตอนก่อนหน้าในโค้ดของ ESP32

4. การเขียนโปรแกรมเพื่อส่งข้อความ

ใช้ฟังก์ชันจากไลบรารี Universal-Arduino-Telegram-Bot เพื่อเชื่อมต่อ ESP32 กับเซิร์ฟเวอร์ Telegram และส่งข้อความ ตัวอย่างโค้ดมีดังนี้:

```
#include <WiFi.h>
#include <UniversalTelegramBot.h>
#include <WiFiClientSecure.h>
const char* ssid = "Your_SSID";
const char* password = "Your_PASSWORD";
#define BOT_TOKEN "Your_Bot_Token"
#define CHAT_ID "Your_Chat_ID"
WiFiClientSecure client;
UniversalTelegramBot bot(BOT_TOKEN, client);
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("WiFi connected!");
}
void loop() {
  bot.sendMessage(CHAT_ID, "Hello from ESP32!", "");
  delay(10000); // ส่งข้อความทุก 10 วินาที
}
```

5. การติดตั้งและอัปโหลดโค้ด

- เชื่อมต่อ ESP32 กับคอมพิวเตอร์ผ่านสาย USB
- เปิด Arduino IDE และเขียนโค้ดตามตัวอย่าง
- ติดตั้งไลบรารี Universal-Arduino-Telegram-Bot หากยังไม่ได้ติดตั้ง
- อัปโหลดโค้ดไปยัง ESP32
- ตรวจสอบว่า ESP32 เชื่อมต่อกับ Wi-Fi และสามารถส่งข้อความไปยัง Telegram ได้สำเร็จ

(ElectronicWings, ม.ป.ป.)

บทที่ 3
อุปกรณ์และวิธีการพัฒนานวัตกรรม

อุปกรณ์และเครื่องมือที่ใช้การออกแบบการสร้าง

1. วัสดุอุปกรณ์

1. แผ่นไม้อัด
2. สายไฟ
3. สว่าน
4. สกรู
5. คีมตัด
6. ไขควงหัวแฉก
7. แผ่นพลาสวูด

2. อุปกรณ์อิเล็กทรอนิกส์

1. ไมโครคอนโทรลเลอร์ ESP32 DEVKIT V1
2. ไมโครคอนโทรลเลอร์ ESP32 Camera
3. จอ LCD I2C 16 x 2
4. Touch Sensor
5. DHT22
6. PIR Sensor
7. MQ-2 Gas Sensor
8. Passive Buzzer
9. หลอดไฟ LED 3 สี
10. ตัวต้านทาน 1k

วิธีการดำเนินการ

1. ศึกษาการสร้างระบบ Smart Home Monitor

1.1. ศึกษาข้อมูลเกี่ยวกับการทำระบบ Smart Home และทำการออกแบบภาพร่างอุปกรณ์ที่ต้องใช้

1.2. การออกแบบและพัฒนาระบบ Smart Home Monitor

1. การออกแบบวงจร วัสดุอุปกรณ์และเซ็นเซอร์ที่ต้องใช้
2. ตัดแผ่นพลาสวูดให้ได้ขนาดสำหรับใช้เป็นฐานวางของอุปกรณ์
3. นำเซ็นเซอร์ บอร์ด จอ ติดกาวสองหน้าและทำการติดตั้งบนแผ่นพลาสวูด
4. ต่อสายไฟกับวงจรตามแบบที่วางไว้

1.3. การติดตั้งระบบ IoT

1. เขียนโค้ดสำหรับควบคุมระบบโดยใช้ภาษา C++ ในโปรแกรม Arduino IDE
2. ตั้งค่าการเชื่อมต่อ IoT ผ่าน WiFi เพื่อเชื่อมต่อกับ Telegram
3. ทดสอบระบบการส่งข้อมูลจากเซนเซอร์ไปยังแพลตฟอร์ม IoT

1.4. การทดสอบระบบและปรับปรุง

1. ทดสอบการทำงานของเซ็นเซอร์ ระบบ และการแสดงผลต่างๆ
2. ตรวจสอบระบบ IoT เช่น การแจ้งเตือนแก๊สรั่ว หรือมีผู้บุกรุกขณะที่ไม่มีคนในบ้าน
3. แก้ไขปัญหาที่พบในระบบ และปรับปรุงให้การทำงานเสถียรมากที่สุด

1.5. ประเมินประสิทธิภาพของระบบ และการแจ้งเตือนของระบบ

บทที่ 4
ผลการดำเนินงาน

จากการดำเนินงานตามขั้นตอนที่วางแผนไว้ คณะผู้จัดทำโครงงานได้ออกแบบและพัฒนาระบบ Smart Home Monitor ที่สามารถตรวจสอบสภาพแวดล้อมภายในบ้าน พร้อมทั้งจัดเก็บข้อมูลและแจ้งเตือนผู้ใช้งานเมื่อเกิดความผิดปกติ โดยผลการดำเนินงานสามารถสรุปได้ดังนี้

1. จากวัตถุประสงค์ข้อที่ 1 เพื่อออกแบบและพัฒนาระบบ Smart Home Monitor ที่สามารถตรวจสอบและแจ้งเตือนข้อมูลต่าง ๆ ภายในบ้านได้ ได้ผลการทดลองออกมาดังนี้

1. ระบบสามารถอ่านค่า อุณหภูมิและความชื้น จากเซ็นเซอร์ DHT22 ได้อย่างถูกต้อง
2. ระบบสามารถตรวจจับการเคลื่อนไหวด้วยเซ็นเซอร์ PIR และส่งคำสั่งไปยัง ESP32-CAM ให้ทำการถ่ายภาพ
3. ระบบสามารถตรวจวัดค่าก๊าซ/ควันด้วยเซ็นเซอร์ MQ-2 และส่งสัญญาณเตือนด้วย บัซเซอร์และไฟ LED
4. ระบบสามารถนับจำนวนบุคคลที่เข้าออกบ้านเพื่อเปลี่ยนโหมดการทำงาน เช่น โหมดรักษาความปลอดภัยเมื่อไม่มีคนอยู่ในบ้าน
5. ข้อมูลและภาพถ่ายจากกล้องสามารถส่งแจ้งเตือนไปยังผู้ใช้งานผ่าน แอปพลิเคชัน Telegram ได้แบบเรียลไทม์

2. จากวัตถุประสงค์ข้อที่ 2 เพื่อพัฒนาระบบควบคุมระบบระยะไกลผ่าน Telegram ได้ผลการทดลองคือระบบสามารถควบคุมระยะไกลผ่าน Telegram ด้วยคำสั่งต่างๆได้

จากผลการดำเนินงานทั้งหมด พบว่าระบบ Smart Home Monitor ที่พัฒนาขึ้นสามารถทำงานได้ครบถ้วนตามวัตถุประสงค์ที่กำหนดไว้ ทั้งด้านการตรวจสอบ แจ้งเตือน และควบคุมระบบ เพื่อเพิ่มความปลอดภัยและความสะดวกสบายในการดูแลบ้าน

บทที่ 5
สรุปผลการดำเนินการ/อภิปรายผลการดำเนินการ

สรุปผลการดำเนินงาน

จากการดำเนินโครงงาน “Smart Home Monitor” คณะผู้จัดทำสามารถพัฒนาระบบต้นแบบที่ทำงานได้ครบถ้วนตามวัตถุประสงค์ที่กำหนดไว้ โดยระบบสามารถตรวจวัดและติดตามข้อมูลสำคัญภายในบ้าน ได้แก่ อุณหภูมิ ความชื้น การเคลื่อนไหว และปริมาณก๊าซ พร้อมทั้งนับจำนวนบุคคลที่เข้าออกบ้าน ข้อมูลดังกล่าวถูกนำมาใช้ในการวิเคราะห์สถานการณ์และควบคุมการแจ้งเตือนผู้ใช้งานผ่านทางแอปพลิเคชัน Telegram

นอกจากนี้ ระบบยังสามารถควบคุมระยะไกลผ่าน Telegram ได้ เพื่อให้ผู้ใช้สามารถควบคุมได้อย่างสะดวก และระบบถ่ายภาพจาก ESP32-CAM เพื่อบันทึกหรือส่งไปยังผู้ใช้งานเมื่อเกิดเหตุผิดปกติ จึงสรุปได้ว่าระบบ Smart Home Monitor ที่พัฒนาขึ้นสามารถตอบสนองต่อวัตถุประสงค์ที่ตั้งไว้ทั้งด้านการเฝ้าติดตาม แจ้งเตือน และบันทึกข้อมูลเพื่อการตรวจสอบย้อนหลัง

อภิปรายผลการทดลอง

จากการดำเนินโครงงานพบว่า ระบบ Smart Home Monitor มีความสามารถในการทำงานที่สอดคล้องกับการใช้งานจริง กล่าวคือ สามารถช่วยเพิ่มความปลอดภัยและความสะดวกสบายแก่ผู้ใช้งานในบ้านได้ โดยการแจ้งเตือนผ่าน Telegram ทำให้ผู้ใช้สามารถรับข้อมูลแบบเรียลไทม์
อย่างไรก็ตาม จากการทดสอบระบบยังพบข้อจำกัดบางประการ เช่น

1. ความเสถียรของการเชื่อมต่อ Wi-Fi อาจส่งผลต่อการส่งข้อมูลและการแจ้งเตือน
2. คุณภาพของภาพจาก ESP32-CAM ขึ้นอยู่กับสภาพแสงและการตั้งค่า ทำให้บางครั้งได้ภาพที่ไม่ชัดเจน

ดังนั้น ในอนาคตสามารถพัฒนาระบบเพิ่มเติมได้ เช่น การเชื่อมต่อกับแอปพลิเคชัน Smart Home อื่น ๆ การใช้ Cloud Database ที่มีประสิทธิภาพสูงขึ้น รวมถึงการเพิ่มเซ็นเซอร์ประเภทอื่นเพื่อให้ครอบคลุมการเฝ้าติดตามสภาพแวดล้อมมากยิ่งขึ้น

ข้อเสนอแนะ

คณะผู้จัดทำโครงงานมีข้อเสนอแนะสำหรับผู้ที่ต้องการพัฒนาระบบ Smart Home Monitor ได้ดังนี้

1. ควรเพิ่มระบบสำรองการสื่อสาร เช่น การใช้โมดูล SIM (4G/5G) หรือการเข้ารหัสข้อมูล เพื่อเพิ่มความน่าเชื่อถือและความปลอดภัยของระบบในกรณีที่สัญญาณอินเทอร์เน็ตภายในบ้านขัดข้องหรือถูกดักจับข้อมูล
2. ควรพัฒนาอินเทอร์เฟซสำหรับผู้ใช้งาน เช่น Dashboard หรือ Mobile Application เพื่อให้สามารถติดตามและควบคุมการทำงานของระบบได้ง่ายและสะดวกมากขึ้น
3. ควรปรับปรุงคุณภาพของการตรวจวัดเซ็นเซอร์ โดยการสอบเทียบ (Calibration) อย่างสม่ำเสมอ เพื่อให้ข้อมูลที่ได้มีความถูกต้องและน่าเชื่อถือมากขึ้น

เอกสารอ้างอิง

adafruit. (2568). **PIR Motion Sensor.** [ออนไลน์]. เข้าถึงได้จาก: https://cdn-learn.adafruit.com/. (วันที่สืบค้นข้อมูล: 5 กันยายน 2568).

cybertice. (2568). **DHT22 AM2302 Module โมดูลวัดอุณหภูมิและความชื้น Temperature and Humidity Sensor Module พร้อมสายไฟ.** [ออนไลน์]. เข้าถึงได้จาก: https://cybertice.com/. (วันที่สืบค้นข้อมูล: 5 กันยายน 2568).

ElectronicWings. (ม.ป.ป.). **Send a Telegram message using ESP32.** [ออนไลน์]. เข้าถึงได้จาก: https://www.electronicwings.com/. (วันที่สืบค้นข้อมูล: 25 มกราคม 2568).

GROBOTRONICS. (ม.ป.ป.). **ESP32 Development Board - DEVKIT V1.** [ออนไลน์]. เข้าถึงได้จาก: https://grobotronics.com/. (วันที่สืบค้นข้อมูล: 1 ธันวาคม 2567).

handsontec. (ม.ป.ป.). **Passive Buzzer Module.** [ออนไลน์]. เข้าถึงได้จาก: https://www.handsontec.com/. (วันที่สืบค้นข้อมูล: 5 กันยายน 2568).

th.element14. (ม.ป.ป.). **เซนเซอร์สัมผัส.** [ออนไลน์]. เข้าถึงได้จาก: https://th.element14.com/. (วันที่สืบค้นข้อมูล: 5 กันยายน 2568).

winsen-sensor. (2568). **Flammable Gas Sensor.** [ออนไลน์]. เข้าถึงได้จาก: https://www.winsen-sensor.com/. (วันที่สืบค้นข้อมูล: 5 กันยายน 2568).

มัทนา วิบูลย์ยะศักดิ์. (ม.ป.ป.). **ทำความรู้จักกับ Internet of Things.** [ออนไลน์]. เข้าถึงได้จาก: https://www.aware.co.th/th/. (วันที่สืบค้นข้อมูล: 25 มกราคม 2568).

ศุภรัตน์ แย้มครวญ. (2566). **ESP32 คืออะไร.** [ออนไลน์]. เข้าถึงได้จาก: https://byter.in.th/esp32/what-is-esp32/. (วันที่สืบค้นข้อมูล: 1 ธันวาคม 2567).

ภาคผนวก

ภาคผนวก ก แสดงโค้ดโปรแกรมที่ใช้พัฒนาโครงงาน Smart Home Monitor บนบอร์ด ESP32 โดยใช้ภาษา C++ ผ่าน Arduino IDE

1. โค้ดการทำงานของบอร์ด ESP32

```
#include <Arduino.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <DHT.h>
#include <LiquidCrystal_I2C.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include <UrlEncode.h>
#define IN_SWITCH_PIN 12
#define OUT_SWITCH_PIN 13
#define GAS_SENSOR_PIN 34
#define PIR_SENSOR_PIN 14
#define RED_LED_PIN 4
#define YELLOW_LED_PIN 2
#define GREEN_LED_PIN 15
#define BUZZER_PIN 18
#define DHT22_PIN 5
#define RX_PIN 16
#define TX_PIN 17
#define ADC_RESOLUTION 4095
String botToken = "8216612749:AAE_0eLRSXB5_kN-YEXyfhh2lmXBIkBTg74";
String chatID = "7969041356";
const char* googleScriptURL =
"https://script.google.com/macros/s/AKfycbzb6h6ei0SyeSQP1Sag8hLeZc_iStKUBuwNqvCAc-
dPuyUUveXW3So-PdeZQO5apO4h/exec";
DHT dht(DHT22_PIN, DHT22);
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

```
void connectWiFi();
void checkWifi();
void sendMessage(String message);
void sendDataToGoogleSheets();
void checkTelegramMessages();
const char* ssid = "กระจายบุญ";
const char* password = "25222524";
int numberOfPeople = 0;
int gasValue = 0;
bool pirState = false;
bool SurveillanceMode = false;
bool nightMode = false;
float temperature = 0.0;
float humidity = 0.0;
unsigned long lastTriggerTime = 0;
unsigned long cooldown = 10000;
unsigned long previousMillis = 0;
const unsigned long interval = 60000;
unsigned long lastBotTime = 0;
const unsigned long botInterval = 10000;
long lastUpdateId = 0;
void setup() {
  Serial.begin(9600);
  Serial2.begin(9600);
  Serial2.setPins(RX_PIN, TX_PIN);
  delay(1000);

  dht.begin();

  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Home Assistant");
```

```
  pinMode(IN_SWITCH_PIN, INPUT);
  pinMode(OUT_SWITCH_PIN, INPUT);
  pinMode(GAS_SENSOR_PIN, INPUT);
  pinMode(PIR_SENSOR_PIN, INPUT);
  pinMode(RED_LED_PIN, OUTPUT);
  pinMode(YELLOW_LED_PIN, OUTPUT);
  pinMode(GREEN_LED_PIN, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  connectWiFi();
  sendMessage("Smart Home Monitor is Ready");
  lcd.clear();
  lcd.setCursor(0, 0);
}
void loop() {
  if (digitalRead(IN_SWITCH_PIN) == HIGH) {
    numberOfPeople++;
    tone(BUZZER_PIN, 1000);
    delay(200);
  }
  if (digitalRead(OUT_SWITCH_PIN) == HIGH) {
    numberOfPeople--;
    if (numberOfPeople < 0) {
      numberOfPeople = 0;
    }
    tone(BUZZER_PIN, 800);
    delay(200);
  }
  if (numberOfPeople == 0) {
    SurveillanceMode = true;
  } else {
    SurveillanceMode = false;
  }
```

```
pirState = digitalRead(PIR_SENSOR_PIN);
gasValue = analogRead(GAS_SENSOR_PIN);
temperature = dht.readTemperature();
humidity = dht.readHumidity();
if (pirState) {
  unsigned long now = millis();
  if (now - lastTriggerTime > cooldown) {
    lastTriggerTime = now;
    if (nightMode || SurveillanceMode) {
      digitalWrite(RED_LED_PIN, HIGH);
      digitalWrite(GREEN_LED_PIN, LOW);
      tone(BUZZER_PIN, 500);
      delay(200);
      noTone(BUZZER_PIN);
      digitalWrite(RED_LED_PIN, LOW);
      Serial2.println("TAKE_PHOTO_AND_SEND");
    } else {
      digitalWrite(RED_LED_PIN, LOW);
      digitalWrite(GREEN_LED_PIN, HIGH);
      delay(200);
      digitalWrite(GREEN_LED_PIN, LOW);
      Serial2.println("TAKE_PHOTO");
    }
  }
}
else {
  digitalWrite(RED_LED_PIN, LOW);
  tone(BUZZER_PIN, 0);
}
if (nightMode || SurveillanceMode) {
  digitalWrite(YELLOW_LED_PIN, HIGH);
}
```

```
else {
    digitalWrite(YELLOW_LED_PIN, LOW);
  }
  if (gasValue > 3000) {
    sendMessage("Gas leak detected! Value: " + String(gasValue));
    tone(BUZZER_PIN, 1200);
    digitalWrite(RED_LED_PIN, HIGH);
    digitalWrite(GREEN_LED_PIN, LOW);
    delay(1000);
    tone(BUZZER_PIN, 0);
    digitalWrite(RED_LED_PIN, LOW);
  }
  checkWifi();
  if (millis() - lastBotTime > botInterval) {
    checkTelegramMessages();
    lastBotTime = millis();
  }
  lcd.setCursor(0, 0);
  lcd.printf("In:%d ", numberOfPeople);
  lcd.setCursor(8, 0);
  lcd.printf("Gas:%4d", gasValue);
  lcd.setCursor(0, 1);
  lcd.printf("T:%2.1fC H:%2.1f%%", temperature, humidity);
}
void sendMessage(String message) {
  if(WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String encodedMessage = urlEncode(message);
    String url = "https://api.telegram.org/bot" + botToken +
                 "/sendMessage?chat_id=" + chatID +
                 "&text=" + encodedMessage;
    http.begin(url);
```

```
    int httpResponseCode = http.GET();
    if(httpResponseCode > 0) {
      Serial.println("ส่งข้อความสำเร็จ: " + String(httpResponseCode));
      Serial.println(http.getString());
    } else {
      Serial.println("ไม่สามารถส่งข้อความได้, รหัสข้อผิดพลาด: " + String(httpResponseCode));
    }
    http.end();
  } else {
    Serial.println("ไม่ได้เชื่อมต่อ Wi-Fi");
  }
}

void checkWifi(){
  if (WiFi.status() != WL_CONNECTED) {
    WiFi.begin(ssid, password);
  }
  else if (WiFi.status() == WL_CONNECTED){
  }
}
void connectWiFi() {
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 20) {
    WiFi.begin(ssid, password);
    digitalWrite(YELLOW_LED_PIN, HIGH);
    delay(250);
    digitalWrite(YELLOW_LED_PIN, LOW);
    delay(250);
    Serial.print(".");
    attempts++;
   }}
```

```
  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("\nWi-Fi connected!");
  } else {
    Serial.println("\nFailed to connect.");
    delay(1000);
  }
}
void checkTelegramMessages() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Checking Msg...");
  lcd.setCursor(0, 1);
  lcd.print("Wait a moment");
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String url = "https://api.telegram.org/bot" + botToken + "/getUpdates?offset=" +
String(lastUpdateId + 1);
    http.begin(url);
    int httpResponseCode = http.GET();
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println("Response: " + response);
      StaticJsonDocument<2048> doc;
      DeserializationError error = deserializeJson(doc, response);
      if (!error) {
        JsonArray result = doc["result"].as<JsonArray>();
        for (JsonObject msg : result) {
          lastUpdateId = msg["update_id"].as<long>();
          String text = msg["message"]["text"].as<String>();
          String chat_id = msg["message"]["chat"]["id"].as<String>();
```

```
          lcd.clear();
          lcd.setCursor(0, 0);
          lcd.print(text);
          lcd.setCursor(0, 1);
          lcd.print("Sending...");
          if (text == "/status") {
            String reply = "People: " + String(numberOfPeople) +
                           "\nGas: " + String(gasValue) +
                           "\nTemp: " + String(temperature) + "C" +
                           "\nHumidity: " + String(humidity) + "%";
            sendMessage(reply);
          }
          else if (text == "/nightmode") {
            nightMode = !nightMode;
            sendMessage("Night mode toggled: " + String(nightMode ? "ON" : "OFF"));
          }
        }
      }
    }
    http.end();
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Success");
  lcd.clear();
  lcd.setCursor(0, 0);
}
```

2.โค้ดการทำงานของบอร์ด ESP Camera

```
#include <Arduino.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <HTTPClient.h>
#include "esp_camera.h"
#include "SD_MMC.h"
#define RX_PIN 3
#define TX_PIN 1
String botToken = "8216612749:AAE_0eLRSXB5_kN-YEXyfhh2lmXBIkBTg74";
String chatID = "7969041356";
const char* ssid = "กระจายบุญ";
const char* password = "25222524";
const char * photoPrefix = "/project_mr.Pitak/photo_";
int photoNumber = 0;
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22
```

```
void checkWifi();
void connectWiFi();
void takePhoto();
String sendPhotoToTelegram(String token, String chat_id);
void savePhotoToSD(camera_fb_t * fb);
void setup() {
  Serial.begin(115200);
  Serial2.begin(9600, SERIAL_8N1, RX_PIN, TX_PIN);
  Serial.println("ESP32-CAM ready to receive commands...");
  connectWiFi();
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
```

```
  if (psramFound()) {
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }
  #if defined(CAMERA_MODEL_ESP_EYE)
  pinMode(13, INPUT_PULLUP);
  pinMode(14, INPUT_PULLUP);
  #endif
  esp_err_t err = esp_camera_init( & config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  sensor_t * s = esp_camera_sensor_get();
  if (s -> id.PID == OV3660_PID) {
    s -> set_vflip(s, 1);
    s -> set_brightness(s, 1);
    s -> set_saturation(s, -2);
  }
  s -> set_framesize(s, FRAMESIZE_QVGA);
  #if defined(CAMERA_MODEL_M5STACK_WIDE) ||
defined(CAMERA_MODEL_M5STACK_ESP32CAM)
  s -> set_vflip(s, 1);
  s -> set_hmirror(s, 1);
```

```
  #endif
  Serial.println("Initialising SD card");
  if (!SD_MMC.begin()) {
    Serial.println("Failed to initialise SD card!");
    return;
  }
  uint8_t cardType = SD_MMC.cardType();
  if (cardType == CARD_NONE) {
    Serial.println("SD card slot appears to be empty!");
    return;
  }
}
void loop() {
  if (Serial2.available()) {
    String command = Serial2.readStringUntil('\n');
    command.trim();
    if (command.length() > 0) {
      Serial.print("Received command: ");
      Serial.println(command);
      if (command == "TAKE_PHOTO") {
        Serial.println("Received request: TAKE_PHOTO");
        takePhoto();
      }
      else if (command == "TAKE_PHOTO_AND_SEND") {
        Serial.println("Received request: TAKE_PHOTO_AND_SEND");
        sendPhotoToTelegram(botToken, chatID);
      }
      else {
        Serial.println("Unknown command");
      }
    }
  }
  checkWifi();}
```

```
void checkWifi(){
  if (WiFi.status() != WL_CONNECTED) {
    WiFi.begin(ssid, password);
  }
  else if (WiFi.status() == WL_CONNECTED){
  }
}
void connectWiFi() {;
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 20) {
    WiFi.begin(ssid, password);
    digitalWrite(2, HIGH);
    delay(250);
    digitalWrite(2, LOW);
    delay(250);
    Serial.print(".");
    attempts++;
  }
  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("\nWi-Fi connected!");
  } else {
    Serial.println("\nFailed to connect.");
    delay(1000);
  }
}
```

```
void takePhoto(){
  camera_fb_t * fb = NULL;
  fb = esp_camera_fb_get();
  if (!fb) {
    Serial.println("Camera capture failed");
    return;
  }
  savePhotoToSD(fb);
  esp_camera_fb_return(fb);
}
String sendPhotoToTelegram(String token, String chat_id) {
  const char* myDomain = "api.telegram.org";
  camera_fb_t * fb = esp_camera_fb_get();
  if (!fb) {
    Serial.println("Camera capture failed");
    return "Camera capture failed";
  }
  WiFiClientSecure client;
  client.setInsecure();
  if (!client.connect(myDomain, 443)) {
    Serial.println("Connection to Telegram failed");
    esp_camera_fb_return(fb);
    return "Connection failed";
  }
```

```
  String boundary = "ESP32CAM";
  String head = "--" + boundary + "\r\n"
                "Content-Disposition: form-data; name=\"chat_id\"\r\n\r\n" +
                chat_id + "\r\n--" + boundary +
                "\r\nContent-Disposition: form-data; name=\"photo\"; filename=\"esp32-
cam.jpg\"\r\n"
                "Content-Type: image/jpeg\r\n\r\n";
  String tail = "\r\n--" + boundary + "--\r\n";

  uint16_t imageLen = fb->len;
  uint16_t totalLen = head.length() + tail.length() + imageLen;
  client.println("POST /bot" + token + "/sendPhoto HTTP/1.1");
  client.println("Host: " + String(myDomain));
  client.println("Content-Length: " + String(totalLen));
  client.println("Content-Type: multipart/form-data; boundary=" + boundary);
  client.println();
  client.print(head);
  uint8_t *fbBuf = fb->buf;
  size_t fbLen = fb->len;
  for (size_t n = 0; n < fbLen; n += 1024) {
    if (n + 1024 < fbLen) client.write(fbBuf, 1024);
    else client.write(fbBuf, fbLen % 1024);
    fbBuf += 1024;
  }
  client.print(tail);
  savePhotoToSD(fb);
  esp_camera_fb_return(fb);
  String response = "";
  while (client.connected()) {
    if (client.available()) {
      response = client.readStringUntil('\n');
      break;
    }}
```

```
  client.stop();
  Serial.println("Telegram response: " + response);
  return response;
}
void savePhotoToSD(camera_fb_t * fb){
  String photoFileName = photoPrefix + String(photoNumber) + ".jpg";
  fs::FS & fs = SD_MMC;
  Serial.printf("Picture file name: %s\n", photoFileName.c_str());
  File file = fs.open(photoFileName.c_str(), FILE_WRITE);
  if (!file) {
    Serial.println("Failed to open file in writing mode");
  } else {
    file.write(fb -> buf, fb -> len);
    Serial.printf("Saved file to path: %s\n", photoFileName.c_str());
    ++photoNumber;
  }
  file.close();
}
```

ภาคผนวก ข การเชื่อมต่อบอร์ด ESP32

ตาราง 1 การเชื่อมต่อบอร์ด ESP32

อุปกรณ์	**ขาเชื่อมต่อ ESP32**	**หมายเหตุ**
ปุ่ม IN	GPIO 12	ตรวจคนเข้า
ปุ่ม OUT	GPIO 13	ตรวจคนออก
MQ-2 Gas Sensor	GPIO 34 (Analog)	อ่านค่าแก๊ส
PIR Sensor	GPIO 14	ตรวจจับการเคลื่อนไหว
DHT22	GPIO 5	อุณหภูมิ/ความชื้น
LED แดง	GPIO 4	สัญญาณอันตราย
LED เหลือง	GPIO 2	Night/Surveillance mode
LED เขียว	GPIO 15	ปลอดภัย
Buzzer	GPIO 18	แจ้งเตือนเสียง
Serial2 (TX)	GPIO 17 ⟶ RX ESP32-CAM	ส่งคำสั่งถ่ายรูป
Serial2 (RX)	GPIO 16 ⟵ TX ESP32-CAM	รับข้อมูล/สถานะ

ตาราง 2 การเชื่อมต่อบอร์ด ESP Camera

อุปกรณ์	**ขาเชื่อมต่อ ESP32**	**หมายเหตุ**
Serial (TX)	GPIO 1	ส่งข้อมูลกลับ ESP32
Serial (RX)	GPIO 3	รับคำสั่งจาก ESP32
Camera	GPIO 0 – 39 (ตาม Config ที่โค้ด)	ใช้ถ่ายรูป
SD_MMC	GPIO 0 – 39 (ตาม Config ที่โค้ด)	ใช้บันทึกภาพ

ภาคผนวก ค คำอธิบายการทำงานระบบ

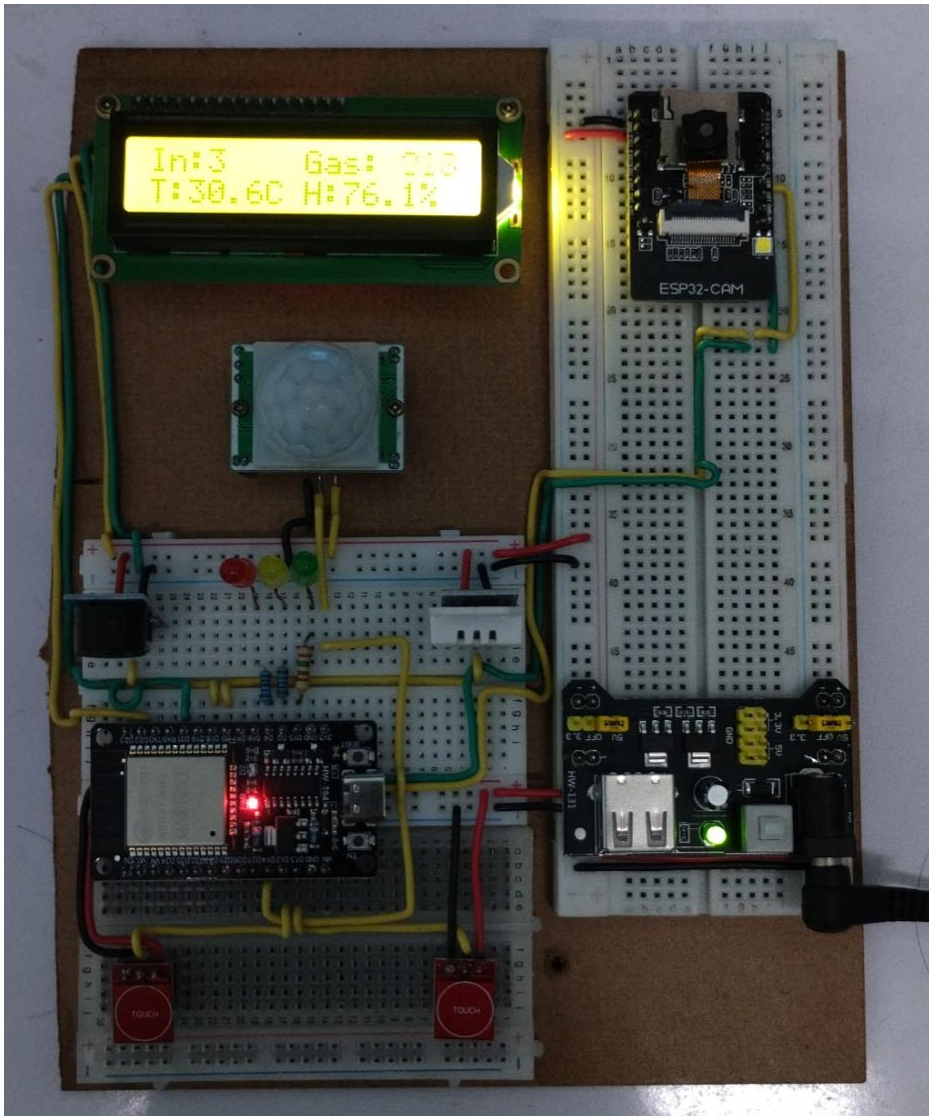
ESP32 (บอร์ดหลัก)

1. อ่านค่าเซ็นเซอร์ DHT22, PIR, MQ-2
2. คำนวณจำนวนคนเข้า–ออก
3. แจ้งเตือนผ่าน Telegram เมื่อพบค่าเกิน threshold
4. ส่งคำสั่ง "TAKE_PHOTO" หรือ "TAKE_PHOTO_AND_SEND" ไปยัง ESP32-CAM ผ่าน Serial2

ESP32-CAM

1. รอรับคำสั่งจาก ESP32
2. ถ่ายภาพและบันทึกลง SD Card
3. ส่งภาพขึ้น Telegram ถ้าได้รับ "TAKE_PHOTO_AND_SEND"

ภาคผนวก ง รูปชิ้นงาน



ภาคผนวก จ การแจ้งเตือน และการควบคุม