ถังขยะอัตโนมัติ

จัดทำโดย

6304062630016 กฤษญา โมรา

ตอนเรียนที่ 1

เสนอ

รองศาสตราจารย์ ดร.กอบเกียรติ สระอุบล

โครงการนี้เป็นส่วนหนึ่งของวิชา INTERNET OF THINGS (040613375)

ภาคเรียนที่ 2 ปี การศึกษา 2565

ภาควิชาวิทยาการคอมพิวเตอร์และสารสนเทศ คณะวิทยาศาสตร์ประยุกต์

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

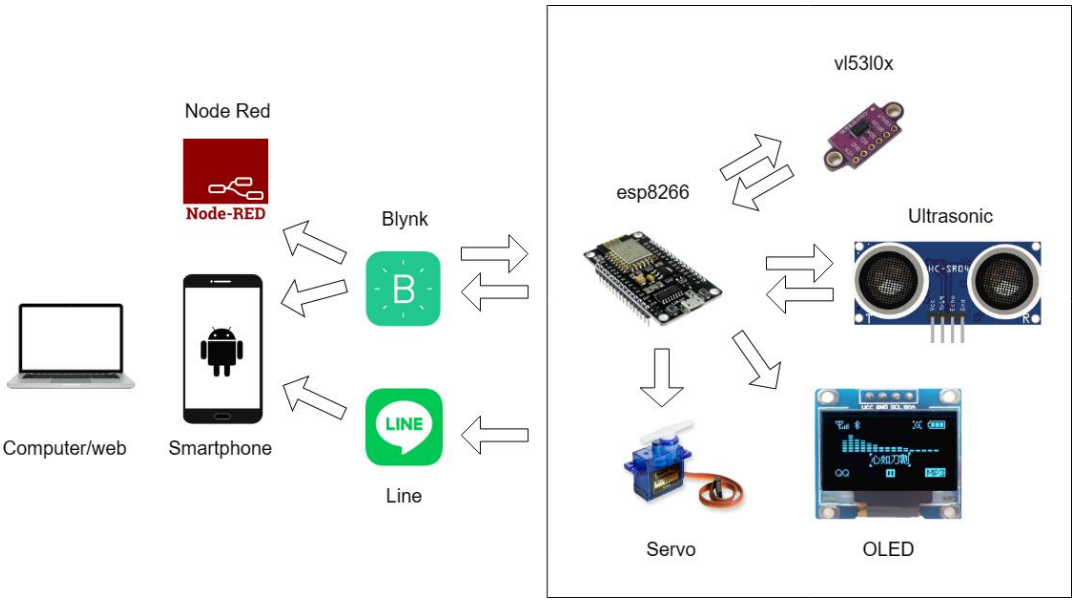
Description

ถังขยะอัตโนมัติ เมื่อมีสิ่งใดอยู่ในระยะตรวจจับของ Ultrasonic sensor ตัวเซนเซอร์จะวัดระยะทางระหว่างสิ่งนั้น ถ้าผลลัพธ์ของการคำนวณที่เกิดจากการวัดมีระยะน้อยกว่าหรือเท่ากับ 10 เซนติเมตร ฝาของถังขยะจะเปิดโดยอัตโนมัติโดยที่ไม่จำเป็นต้องเอื้อมมือเปิดฝากถังขยะด้วยตนเอง โดยทำเพียงยืนอยู่หน้าถังขยะเพียงเท่านั้น จะช่วยลดการสัมผัสสิ่งสกปรกโดยไม่จำเป็น และเมื่อพ้นระยะตรวจจับของเซนเซอร์ฝาถังก็จะปิดลง

มีหน้าจอ OLED แสดงความสูงของถังขยะ , ความสูงของขยะที่มีอยู่ในถังใช้ VL53L0X ในการวัดและแสดงสถานะการเปิดปิดของฝาถังว่า Open หรือ Close และใช้ Line Notify เพื่อให้มีการแจ้งเตือนของไลน์แอปพลิเคชันเมื่อมีความสูงของขยะที่มีอยู่ในถังมากกว่าหรือเท่ากับ 160 มิลลิเมตร ไลน์จะส่งข้อความว่า ถังขยะเต็ม

ใช้ Service/Cloud คือ Blynk และ Node Red โดยจะส่งความสูงของขยะที่มีอยู่ในถังขึ้นไปสร้างเป็น Dashboard เพื่อแสดงผลบนเว็บไซต์และมือถือ ใช้ Node Red ในการส่งข้อมูลจาก Blynk เพื่อทำ MQTT ส่งข้อความเป็นความสูงของขยะที่มีอยู่ในถัง

สถาปัตยกรรมระบบ



เมื่อมีสิ่งใดอยู่หน้า Ultrasonic Sensor ในระยะทางที่น้อยกว่าหรือเท่ากับ 10 เซนติเมตร เซนเซอร์จะส่งสัญญาณไปยังบอร์ด esp8266 เพื่อสั่งให้ Servo Motor หมุนไปที่ 180 องศาก็คือการเปิดฝาถังขยะ ถ้าไม่มีสิ่งใดอยู่หน้าระยะตรวจจับของเซนเซอร์บอร์ด esp8266 จะสั่งให้ Servo Motor หมุนกลับมาที่ 15 องศา เป็นการปิดฝาของถังขยะ

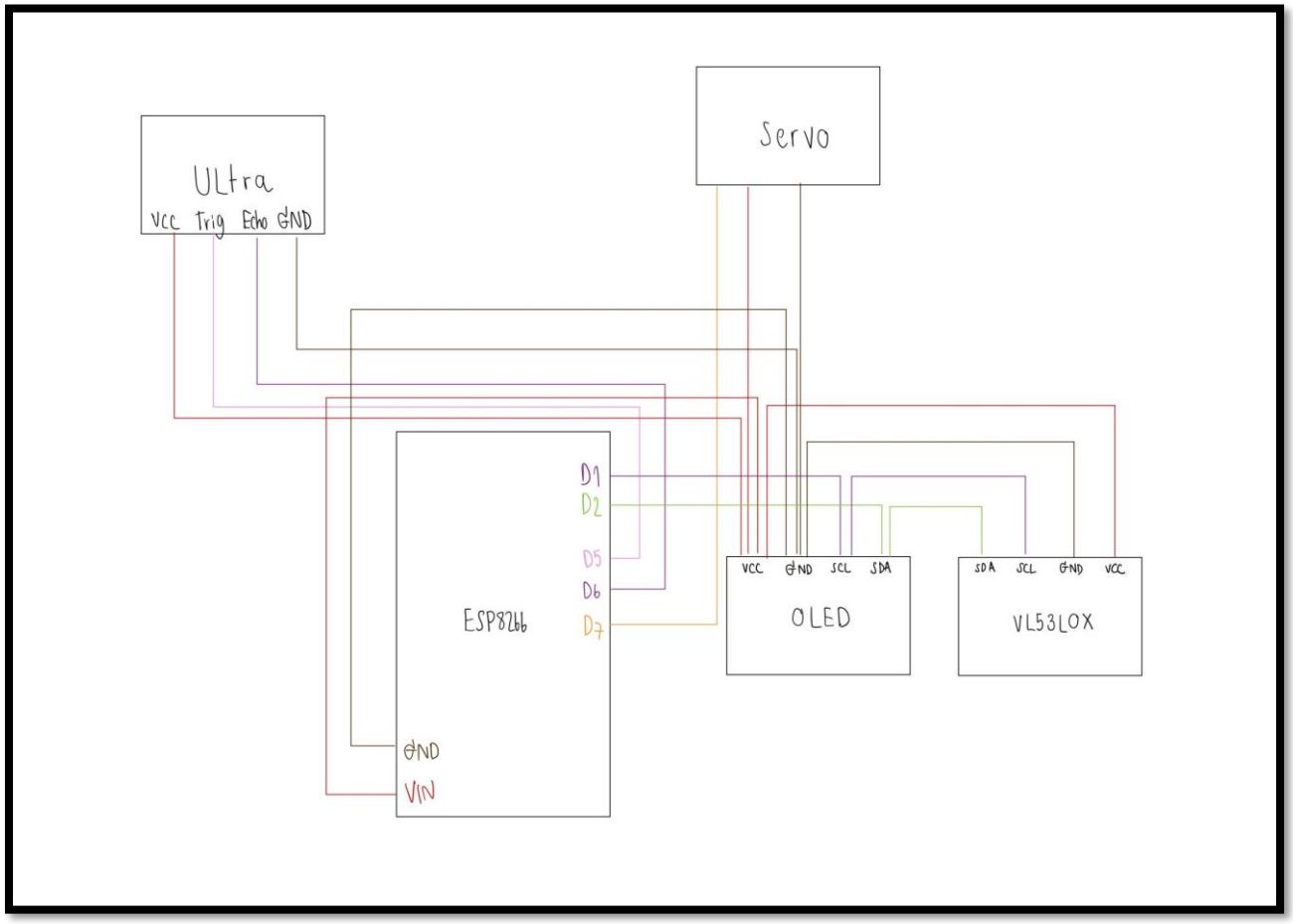
ใช้ VL53L0X V2 Laser Ranging Distance Sensor สื่อสารแบบ I2C ซึ่งเป็นเซนเซอร์วัดระยะทางเช่นกัน แต่ภายในโปรเจกต์นี้จะนำมาวัดความสูงของขยะที่มีอยู่ภายในถังแล้วส่งไปยังบอร์ด esp8266

OLED สื่อสารแบบ I2C จะรับข้อมูลจาก esp8266 เพื่อแสดงผลลัพธ์ทางหน้าจอ คือ จะแสดงความสูงของถังขยะเป็นหน่วยมิลลิเมตร , แสดงความสูงที่อยู่ภายในถังขยะเป็นหน่วยมิลลิเมตร และแสดงสถานะการเปิดปิดของฝากถังว่า Open หรือ Close

Esp8266 จะเชื่อมกับ Line Application โดยใช้ Line Notify ในการแจ้งเตือนข้อความทางไลน์ เมื่อมีความสูงของขยะในถังมากกว่าหรือเท่ากับ 160 มิลลิเมตร จะแจ้งเตือนข้อความว่า Status: ถังขยะเต็ม โดยจะแจ้งเตือนเพียงครั้งเดียว จนกว่าความสูงของถังขยะจะน้อยกว่า 160 มิลลิเมตร และเมื่อเกินค่าที่กำหนดไว้ ไลน์จึงจะทำการแจ้งเตือนอีกครั้ง

ใช้ Blynk และ Node Red เป็น Server/Cloud โดยจะใช้ Blynk มาเป็น Service ในการสร้าง Dashboard โดย esp8266 จะส่งค่าความสูงของขยะที่มีอยู่ในถังจาก VL53L0X ขึ้นไปแสดงบนหน้าเว็บและแอปพลิเคชันบนมือถือของ Blynk ส่วน Node Red จะเชื่อมกับ Blynk โดยดึงค่าความสูงของขยะที่มีอยู่ในถังมาทำเป็น MQTT ในการรับส่งข้อความระหว่างอุปกรณ์

วงจร



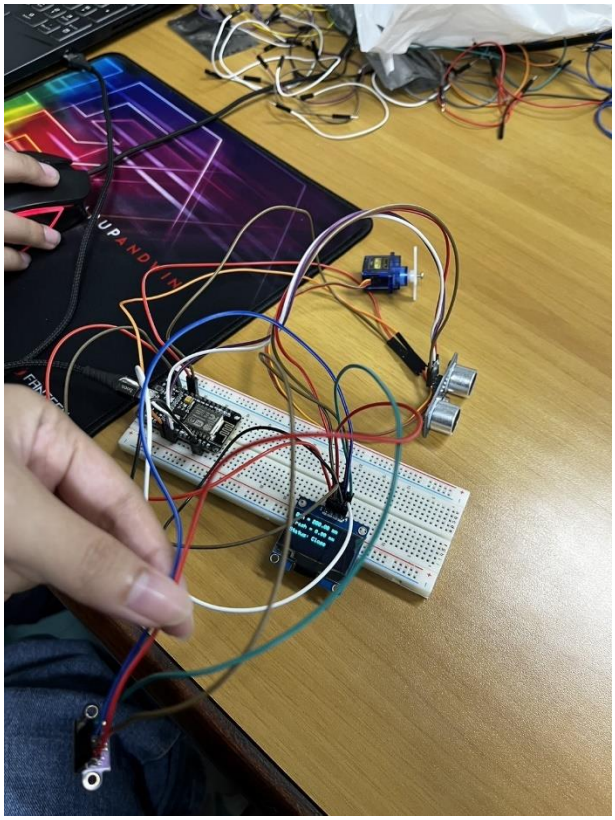
Ultra : VCC -> VCC ของ OLED , Trig -> D5 , Echo -> D6 , GND - > GND ของ OLED

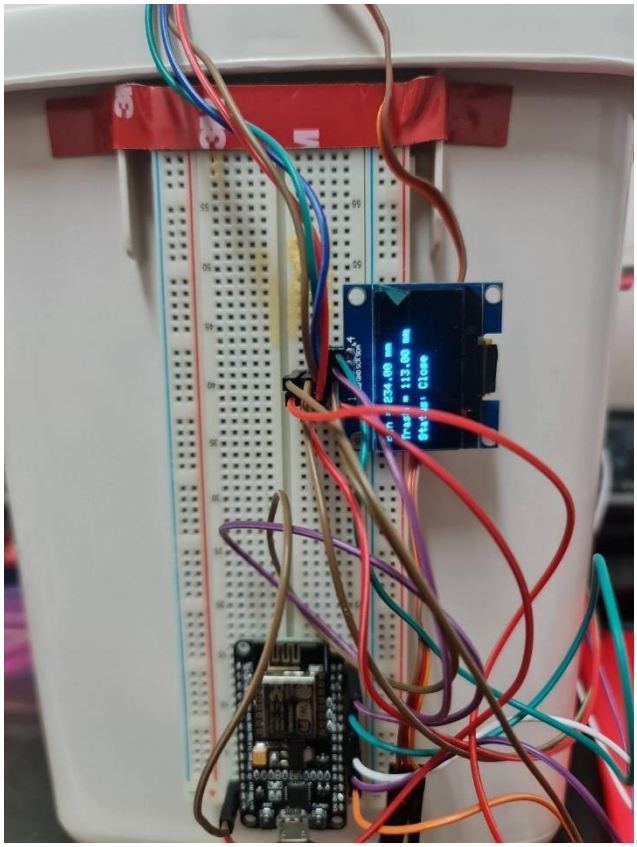
Servo : สายสีส้ม -> D7 , สายสีแดง -> VCC ของ OLED , สายสีน้ำตาล -> GND ของ OLED

VL53L0X : SDA -> SDA ของ OLED , SCL -> SCL ของ OLED ,
GND -> GND ของ OLED , VCC -> VCC ของ OLED

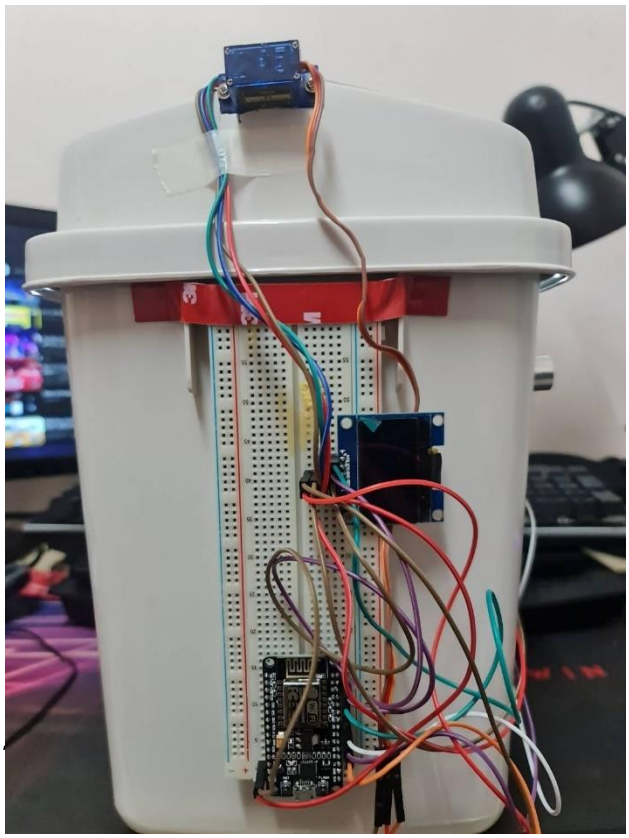
OLED : VCC -> VIN , GND -> GND , SCL -> D1 , SDA -> D2

วงจรที่นำไปต่อจริง

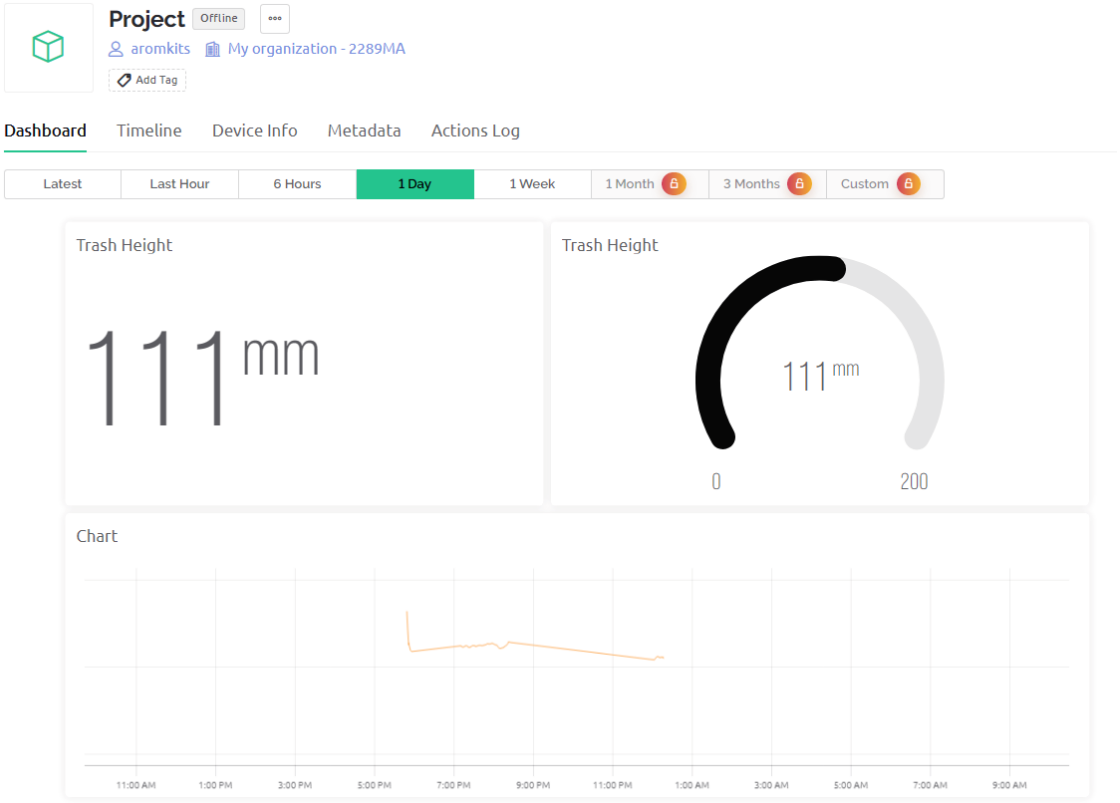


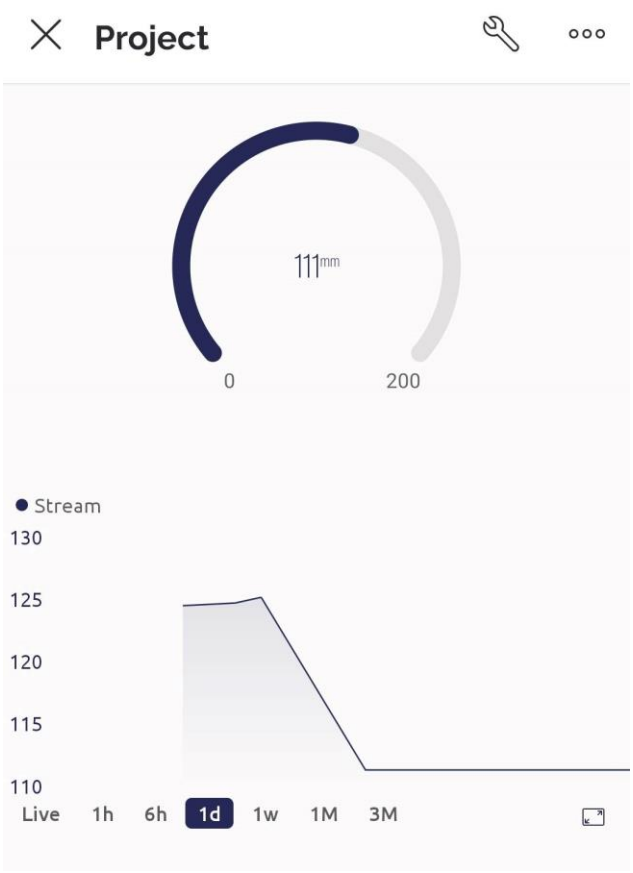


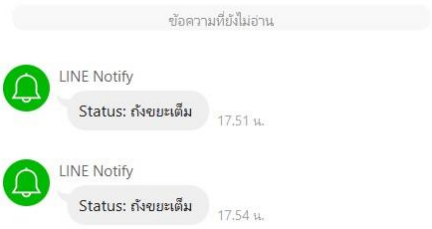
ภาพรวมของถังขยะ

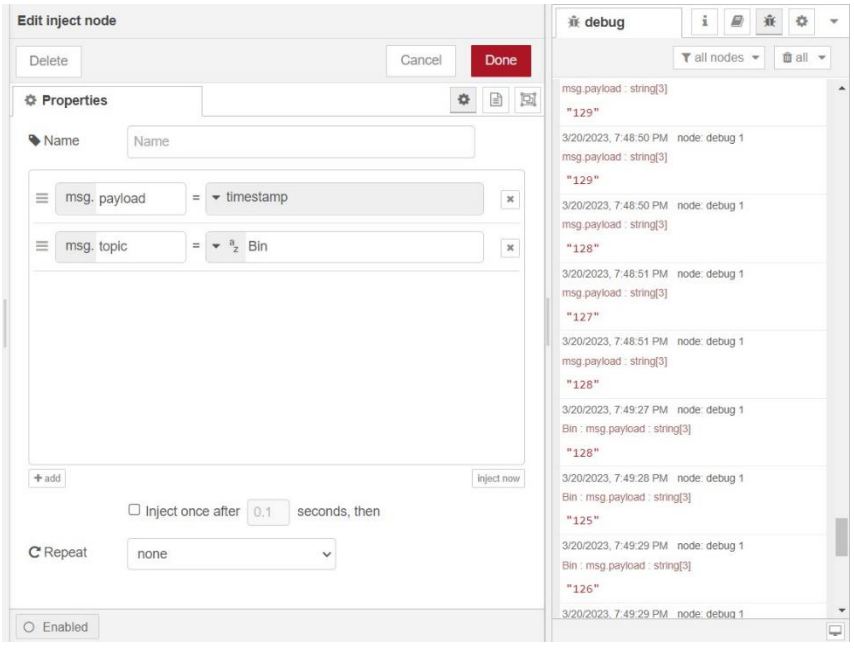


ภาพของ Dashboard , MQTT , Line









โค้ด

library

```
#include <ESP8266WiFi.h> — 1
#include <BlynkSimpleEsp8266.h> — 2

#include <Adafruit_SH110X.h> — 3
Adafruit_SH1106G display = Adafruit_SH1106G(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#include <TridentTD_LineNotify.h> — 4
#include "Adafruit_VL53L0X.h" — 5
Adafruit_VL53L0X laser = Adafruit_VL53L0X ();

#include<Servo.h> — 6
Servo servo1;
```

1. เป็น library WiFi ของ ESP8266 (ESP8266WiFi.h) แล้วกำหนดค่า SSID และรหัสผ่านของเครือข่าย WiFi ที่ต้องการเชื่อมต่อเก็บไว้ที่ตัวแปร ssid และ password ตามลำดับ
2. เป็น library เพื่อใช้งาน Blynk
3. เป็น library ของจอ OLED
4. เป็น library เพื่อใช้ Line Notify
5. เป็น library ของ VL53L0X
6. เป็น library ของ Servo Motor

Ultrasonic Seneor

```
void ultrasonic()
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;

  Serial.println(String(distance) + " CM");

}
```

1. จะเป็นการตั้งค่าการจ่ายกระแสไฟให้กับขา trigPin
2. จะอ่านค่าที่ขา echo เป็น pulse เก็บไว้ใน duration แล้วนำไปคำนวณเป็นเซนติเมตร

โค้ดการทำงานหลัก

```
void loop()
{
  Blynk.run();

  ultrasonic();
  display.clearDisplay();
  display.setTextColor(SH110X_WHITE);
  display.setCursor(0, 0);
  display.print("Bin = " + String(height_Bin)+" mm");
  display.setCursor(0, 16);
  display.print("Trash = "+String (trash) +" mm");

  if (distance <= 10)
  {
    servo1.write(180);
    display.setCursor(0, 32);
    display.print("Status: Open");
  }
  else
  {
    servo1.write(15);
    display.setCursor(0, 32);
    display.print("Status: Close");
  }
```

1. เชื่อมต่อกับ Blynk
2. เรียกใช้ฟังก์ชัน ultrasonic
3. เป็นการตั้งค่าการแสดงผลบนหน้าจอ OLED
4. ถ้า distance ที่ได้จาก ultrasonic มีค่าน้อยกว่าหรือเท่ากับ 10 เซนติเมตร ให้ Servo หมุนไป 180 องศา และแสดงข้อความทางหน้าจอ OLED ว่า Open
5. ถ้า distance ที่ได้จาก ultrasonic มากกว่า 10 เซนติเมตรให้ Servo หมุนไป 0 องศา และแสดงข้อความทางหน้าจอ OLED ว่า Close

```
laser.startRangeContinuous();

if(laser.isRangeComplete())
{
  trash = laser.readRange();
  if(trash > 234)
  {
    trash = 0;
  }
  else if(trash == 0)
  {
    trash = 0;
  }
```

1. เรียกใช้ VL53L0X เพื่อที่จะอ่านค่า
2. ถ้า VL53L0X อ่านค่าได้สำเร็จ จะเอาค่าที่อ่านได้ไปเก็บไว้ในตัวแปร trash โดยจะอ่านค่าได้เป็นหน่วยมิลลิเมตร กำหนดเงื่อนไขถ้า trash เกิน 234 มิลลิเมตร นั่นก็คือความสูงของถังที่ใช้ทำโปรเจกต์ ให้ trash เก็บค่า 0 แสดงว่าไม่มีขยะอยู่ในถัง ถ้า trash วัดค่าได้ 0 ก็ set เป็น 0 เช่นเดียวกัน

```
  else
  {
    trash = height_Bin - trash;
    if(trash >= 160 && !alerted)
    {
      LINE.notify("ถังขยะเต็ม");
      alerted = true;
    }
    else if(trash < 160 && alerted)
    {
      alerted = false;
    }
  }
  Serial.println("trash: " + String(trash));
}

Blynk.virtualWrite(V0, trash);
display.display();
delay(100);
```

1. ในปีกกา else คือถ้ามีถังขยะอยู่ในถังและ VL53L0X วัดค่าได้ จะนำความสูงของถังขยะที่กำหนดไว้ตายตัวมาลบกับ trash ที่วัดค่าได้ เพื่อให้ได้ความสูงของขยะที่อยู่ในถัง
2. ถ้าความสูงของขยะในถังมากกว่าหรือเท่ากับ 160 และตัวแปร alerted เป็น false จะส่งข้อความแจ้งเตือนทางไลน์ว่า ถังขยะเต็ม
3. ถ้าความสูงของขยะในถังน้อยกว่า 160 และตัวแปร alerted เป็น true จะ set ให้ alerted เป็น false
4. นำ trash ส่งผ่าน PIN V0 เพื่อใช้แสดงผล dashboard บน Blynk

โครงสร้างข้อมูล

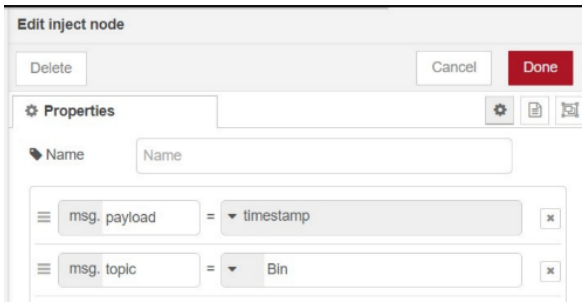
ไฟล์ JSON ที่ได้จากการ Export จาก Node Red


```
[
    {
        "id": "0cdeb150a1bfbe61",
        "type": "inject",
        "z": "cfac41f32aef7b1e",
        "name": "",
        "props": [
            {
                "p": "payload"
            },
            {
                "p": "topic",
                "vt": "str"
            }
        ],
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "topic": "Bin",
        "payload": "",
        "payloadType": "date",
        "x": 350,
        "y": 380,
        "wires": [
            [
                "339668b1a7efa442"
            ]
        ]
    }
]
```


Data รับส่งในระบบ

Blynk จะส่งค่า trash ให้กับ Node Red ผ่าน PIN V0 ซึ่ง Node Red จะรับค่าจาก V0 และนำค่าที่ได้ส่งเข้า MQTT ที่มี Topic ชื่อว่า Bin

MQTT Topic



IOT Platform/Service

Features

Blynk

เป็น IoT Cloud ซึ่งถูกพัฒนามาจากภาษา Java สามารถทำงานกับระบบปฏิบัติการได้หลากหลาย โดยแม่เครื่องแม่ข่ายพัฒนาแบบ Open-Source ซึ่งสามารถนำไปใช้ประโยชน์ได้มากมาย รวมไปถึงการใช้งานประกอบการสร้างนวัตกรรมเพื่อการค้าด้วย

จุดเด่น/ข้อดี มีดังนี้ คือ ผู้ใช้สามารถออกแบบและสร้างหน้าจอแสดงผล UI บนเว็บหรือโทรศัพท์มือถือได้โดยใช้ Widgets ต่างๆ ที่ Blynk มีให้ , รองรับการเชื่อมต่อกับอุปกรณ์ IoT ที่มีการเชื่อมต่ออินเทอร์เน็ตได้ , การควบคุมอุปกรณ์ IoT ผ่านเว็บหรือโทรศัพท์มือถือ , รองรับการส่งการแจ้งเตือนและการแจ้งเตือนสถานะไปยังอุปกรณ์ของผู้ใช้ , ใช้ง่าย , มีความยืดหยุ่น เพราะ Blynk รองรับการเชื่อมต่อกับอุปกรณ์ IoT หลายแบบ และรองรับการเชื่อมต่อผ่านหลาย ๆ โปรโตคอล, มีการเข้ารหัสข้อมูลที่ส่งระหว่างอุปกรณ์ IoT และแอพพลิเคชั่นผ่านโครงสร้างแบบ SSL/TLS ทำให้ข้อมูลปลอดภัยจากการถูกดักจับและปลอมแปลง , สามารถบันทึกประวัติการใช้งานอุปกรณ์ IoT ได้ ซึ่งช่วยให้ผู้ใช้สามารถติดตามการใช้งานอุปกรณ์ได้อย่างง่ายดาย

ข้อจำกัด คือ Blynk เป็นแพลตฟอร์ม IoT แบบ cloud-based ซึ่งหมายความว่าต้องเชื่อมต่อกับบริการ cloud ของ Blynk เพื่อใช้งาน และอาจทำให้มีปัญหาเมื่อเซิร์ฟเวอร์ของ Blynk ล่มหรือมีปัญหาในการเชื่อมต่อกับอินเทอร์เน็ต , ในแพลตฟอร์มฟรีของ Blynk จะมีข้อจำกัดในการใช้งาน เช่น การ Export Report ต้องเสียเงิน ทำให้ผู้ใช้ต้องจ่ายค่าบริการหากต้องการใช้ฟีเจอร์เพิ่มเติมหรือสร้างโปรเจกต์ใหม่โดยเฉพาะกับการใช้งานในระยะยาว

Node Red

มีการรองรับ Message Queuing Telemetry Transport (MQTT) ซึ่งเป็นเป็น Protocol ที่ออกแบบมาเพื่อการเชื่อมต่อแบบ M2M (machine-to-machine) คือ อุปกรณ์ติดต่อหรือสื่อสารกับอุปกรณ์

จุดเด่น/ข้อดี มีดังนี้ คือ ตัวสร้าง Flow ที่ใช้งานง่าย , รองรับการทำงานแบบ real-time , ฟรีและเป็น Open Source , รองรับหลายแพลตฟอร์ม , มีการรองรับ MQTT , รองรับการเชื่อมต่อกับหลายฐานข้อมูล และสามารถนำข้อมูลมาแสดงผลในรูปแบบต่างๆ ได้ เช่น Dashboard หรือแผงควบคุม , รองรับการใช้งาน Node.js

ข้อจำกัด คือ หากต้องการสร้าง Flows ที่มีความซับซ้อนอาจจะทำให้ต้องมีจำนวนของ Nodes และ Connection มาก ทำให้การจัดการและดู Flows ของตัวเองยากขึ้น , การแสดงผลของ Node-Red อาจจะไม่สามารถทำได้ตามที่ผู้ใช้งานต้องการ เนื่องจากมีการจำกัดในการแสดงผลที่สามารถปรับแต่งได้น้อย , การติดตั้งและการใช้งาน Node-Red อาจจะยากสำหรับผู้ที่ไม่มีความเชี่ยวชาญเฉพาะทางในเรื่องนี้ , หากต้องการประมวลผลข้อมูลที่มีปริมาณมาก อาจจะต้องใช้เวลานานในการประมวลผล เนื่องจาก Node-Red ใช้งานบนเว็บเบราว์เซอร์ซึ่งอาจมีข้อจำกัดในการประมวลผลข้อมูลที่มาก

Space

Blynk

แบบฟรีจะไม่สามารถดู Dashboard แบบรายเดือนได้ Custom ไม่ได้ Download Report และ Ping ไม่ได้ มีการจำกัด Device และ Widgets บางตัวไม่สามารถใช้งานได้

แต่แบบเสียเงินจะมีแบบ Package คร่าว ๆ ดังนี้

Blynk Plus: ราคาเริ่มต้น $4.99/เดือน , สามารถเชื่อมต่อกับบอร์ด IoT ได้ไม่จำกัด, สามารถสร้างโปรเจ็กต์ IoT ได้ไม่จำกัด , จำนวนผู้ใช้งานไม่จำกัด , ฟังก์ชันเสริมเพิ่มเติม เช่น การส่งแจ้งเตือนผ่านอีเมล

Blynk Pro: ราคาเริ่มต้น $42/เดือน , มีความสามารถทั้งหมดของ Blynk Plus , สามารถสร้างและใช้งาน Blynk App แบบ Private ได้ , สามารถเชื่อมต่อกับฐานข้อมูลได้ , สามารถสร้างและใช้งาน Widget ที่เฉพาะเจาะจงได้

Blynk Business: ราคาเริ่มต้น $499/เดือน , มีความสามารถทั้งหมดของ Blynk Pro , สามารถเชื่อมต่อกับแพลตฟอร์มอื่น ๆ เช่น Azure, AWS, Google Cloud เป็นต้น , สามารถเชื่อมต่อกับฐานข้อมูลขนาดใหญ่ได้ , สามารถติดตั้งบนเครื่องเซิร์ฟเวอร์ขององค์กรได้ , มีฟังก์ชันการจัดการผู้ใช้งานและการกำหนดสิทธิ์การเข้าถึงเพิ่มเติม

Node Red

การใช้งาน Node-RED ในรูปแบบเสียเงิน หรือ Enterprise Edition จะได้รับการสนับสนุนจากผู้ให้บริการต่าง ๆ และสามารถใช้งานได้ด้วยฟีเจอร์ที่เพิ่มเติม เช่น

การรองรับและการจัดการกับระบบขนาดใหญ่: Node-RED Enterprise สามารถรองรับการติดตั้งบนเครื่องเซิร์ฟเวอร์หรือคลาวด์ และรองรับการทำงานในระบบขนาดใหญ่ โดยเพิ่มความสามารถในการจัดการส่วนต่อประสานกับระบบปฏิบัติการและดาต้าเบส

การเชื่อมต่อและการทำงานร่วมกับพื้นฐานสมัยใหม่: Node-RED Enterprise มีการรองรับการเชื่อมต่อและการทำงานร่วมกับพื้นฐานสมัยใหม่ เช่น Docker และ Kubernetes ซึ่งช่วยให้ง่ายต่อการสร้างและการจัดการบนระบบ

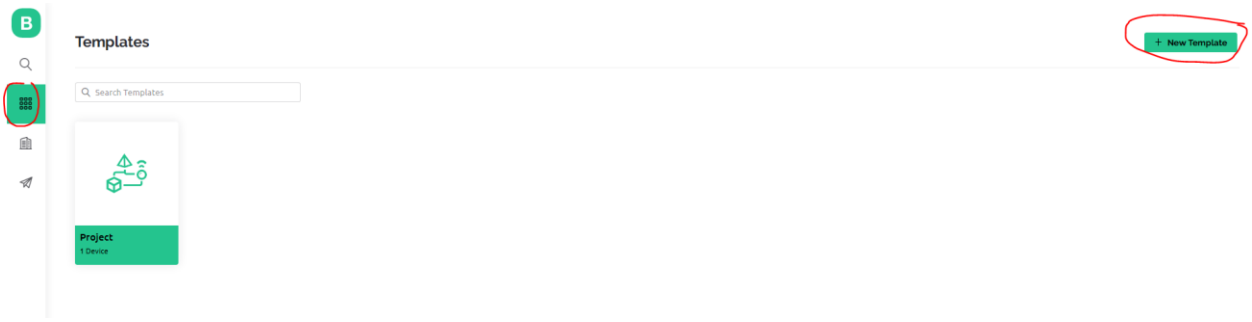
การปรับแต่งและการควบคุมการเข้าถึง: Node-RED Enterprise มีความยืดหยุ่นในการปรับแต่งและการควบคุมการเข้าถึง เช่น การสร้างสิทธิ์การเข้าถึงเพื่อป้องกันการเข้าถึงของผู้ไม่พึงประสงค์ และการกำหนดระดับการเข้าถึง

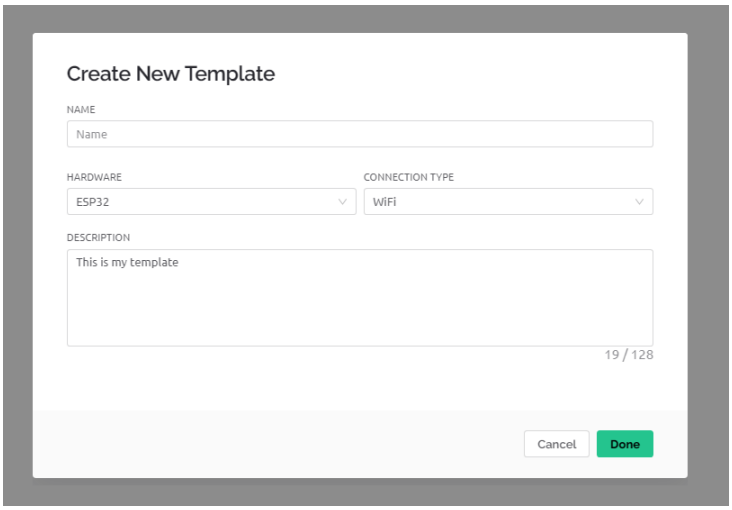
การตรวจสอบและการเข้าถึงข้อมูล: Node-RED Enterprise มีความสามารถในการตรวจสอบและการเข้าถึงข้อมูลอย่างมีประสิทธิภาพ และมีการรองรับการจัดการการให้บริการในพื้นที่ที่ปลอดภัย

โดย Node-RED Enterprise มีการเสนอราคาเฉพาะตามความต้องการ

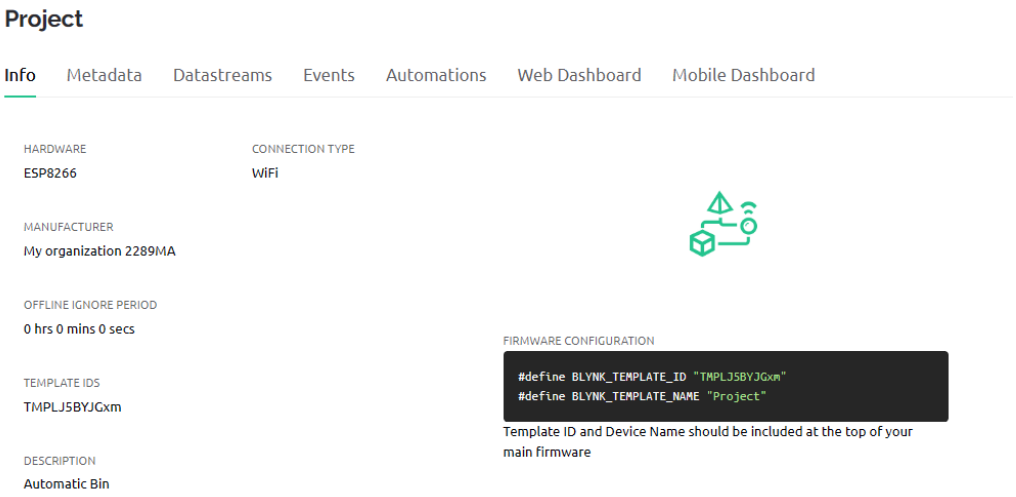
ขั้นตอนการ Setup/Config

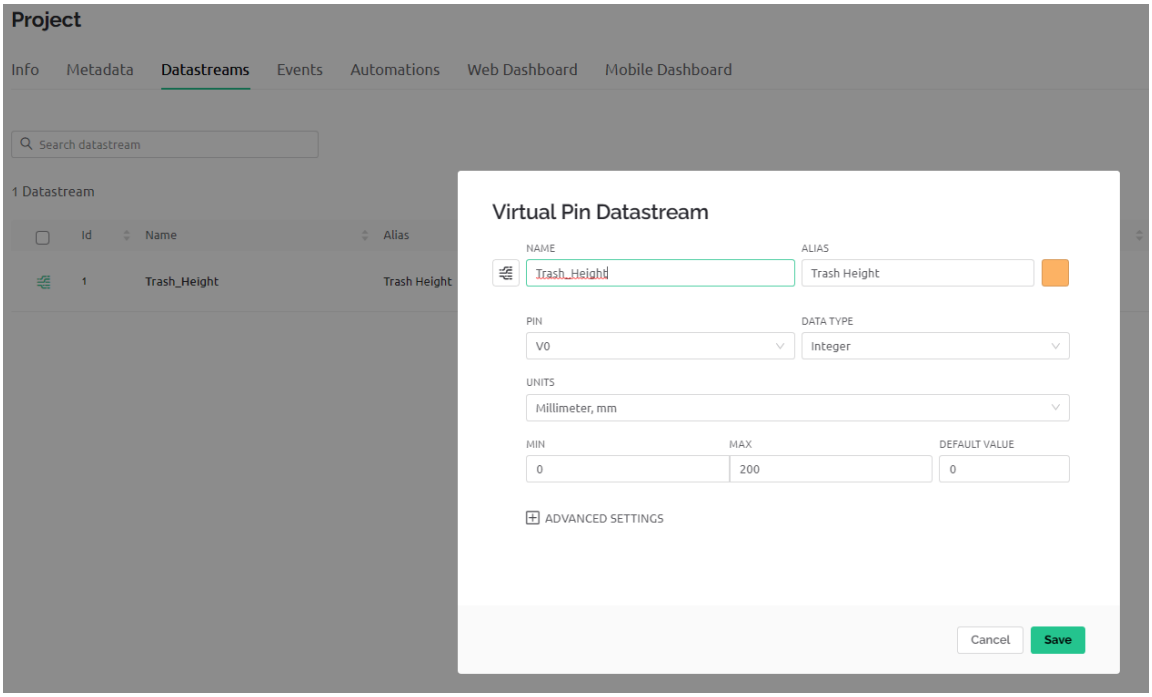
Blynk



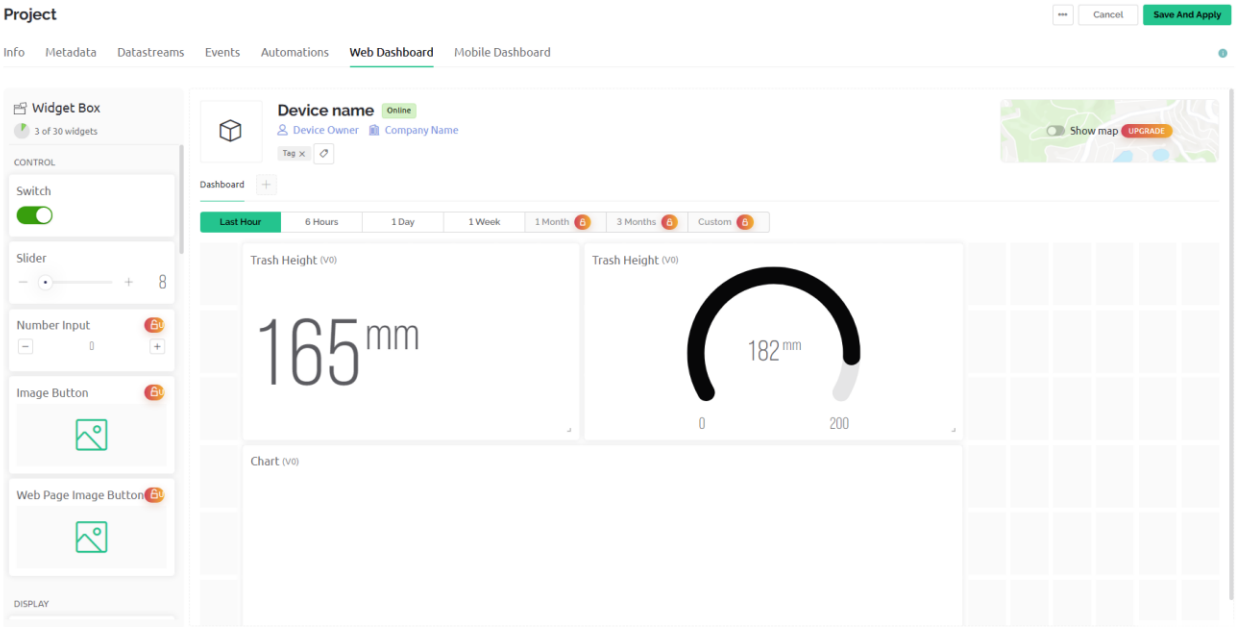


1. ต้อง Create New Template ก่อน แล้วกรอกข้อมูลให้ครบถ้วน

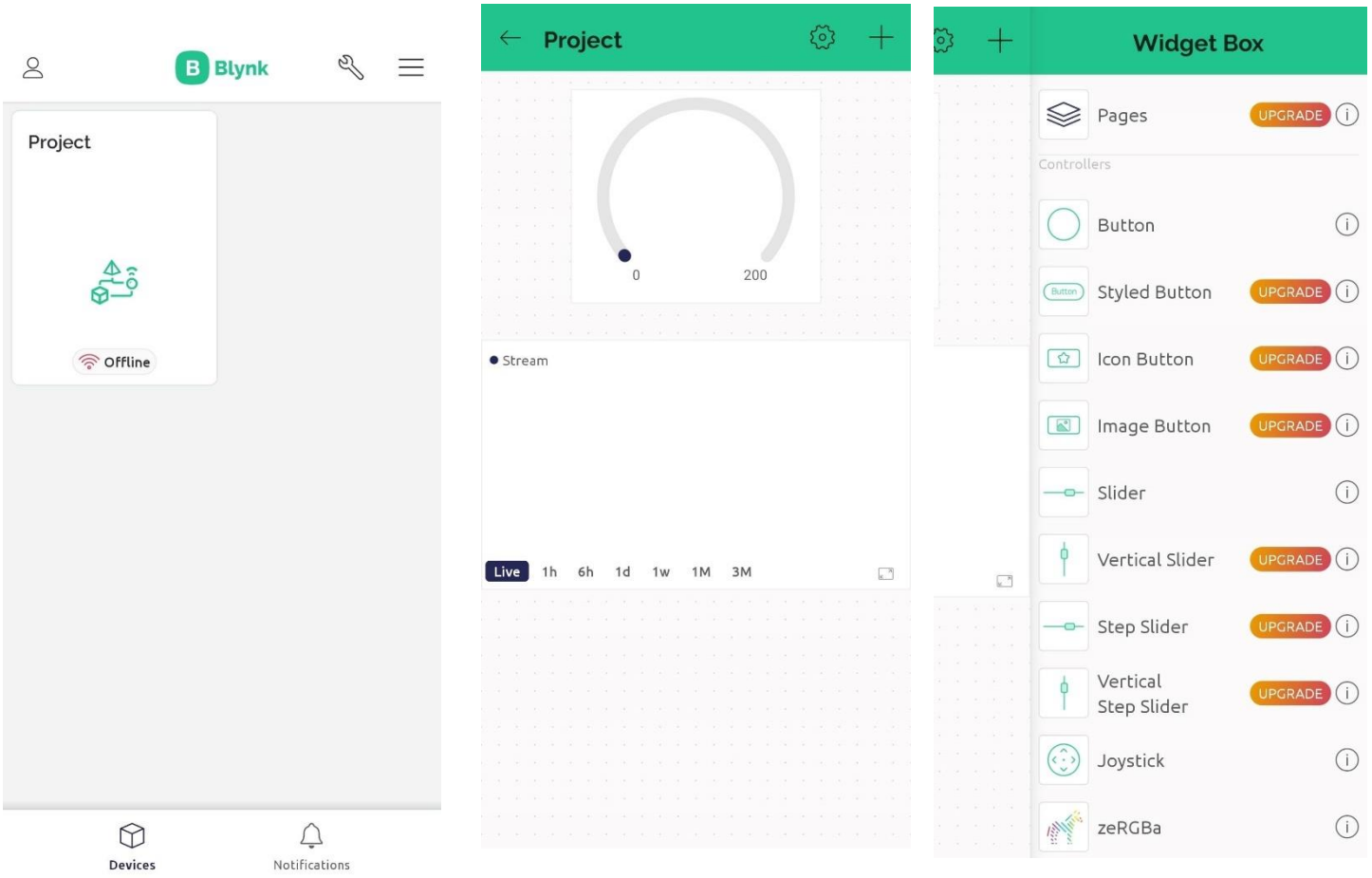




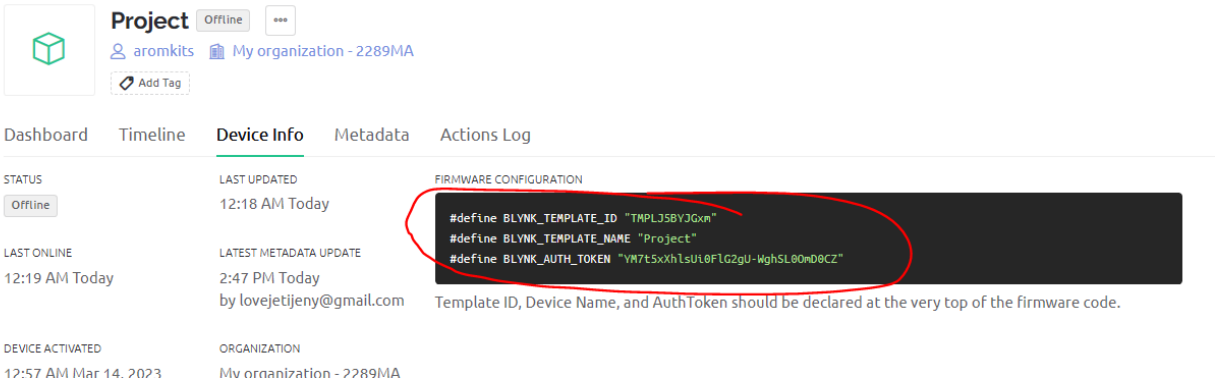
2. มาในหน้าของ Datastreams เพื่อใส่ PIN ข้อมูลที่เราจะรับจากโค้ดของเรา



3. ทำการแก้ไข Dashboard ตามต้องการ



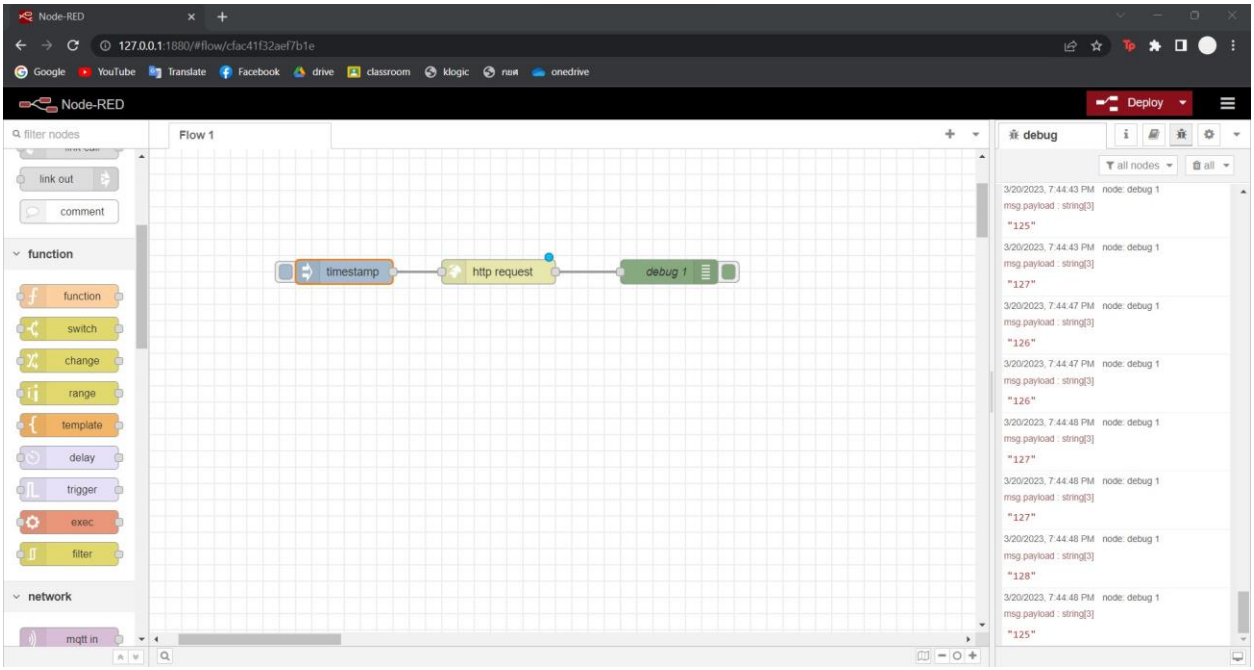
4. ส่วนในมือถือให้โหลด App Blynk แล้ว Add New Device รันบอร์ดเพื่อให้บอร์ดปล่อย Wifi และเชื่อมกับบอร์ดจะได้ Device ตัวเดียวกับที่อยู่ในหน้าเว็บของเราก็คือจะได้ Project มา จากนั้นให้ปรับแต่ง Dashboard ตามต้องการ



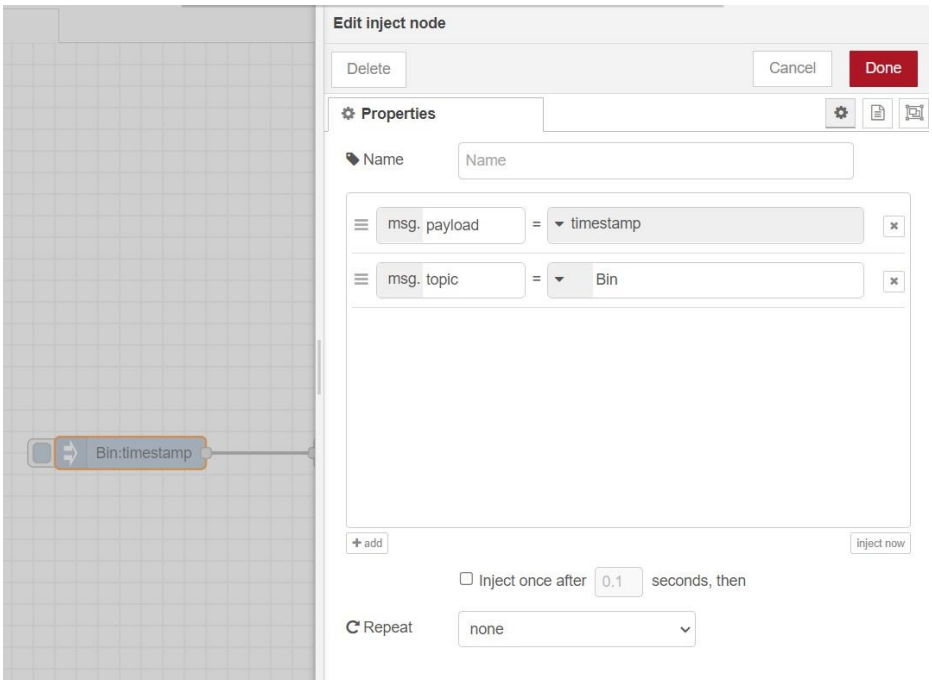
5. เมื่อทำ Dashboard เสร็จแล้วให้ไปหน้า Device info แล้วเอาข้อมูลที่วงไว้ไปใส่ในโค้ด เพื่อเชื่อมต่อระหว่าง esp8266 กับ Blynk

Node Red

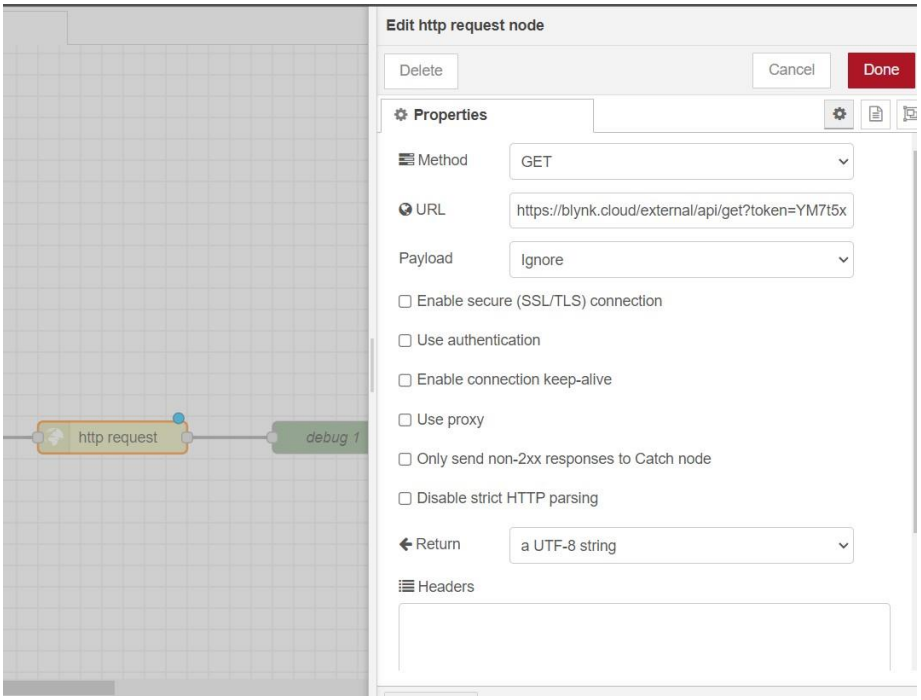
ต้องเปิดหน้า Command Prompt ก่อน แล้วติดตั้ง Node Red โดย npm install -g --unsafe-perm node-red แล้วจากนั้นให้รันคำสั่ง node-red คำสั่งนี้จะให้ URL ของ Node Red มา จากนั้นทำการ Copy URL ไปรันบนหน้าเว็บและปรับแต่งตามต้องการ ต้องเปิดหน้า Command ที่รัน node-red ไว้ตลอด ไม่งั้นจะไม่สามารถใช้งานได้



1. ลาก inject , http request , debug มาใส่

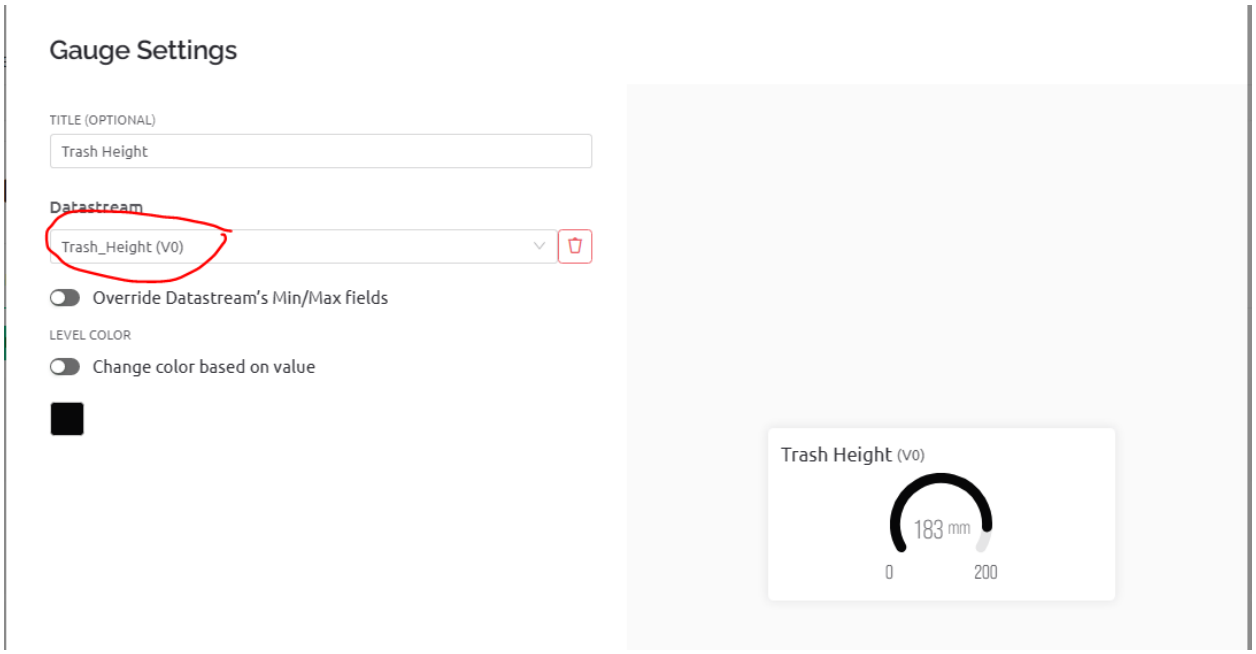


2. แก้ไข inject โดยใส่ topic ชื่อว่า Bin



3. แก้ไข URL ให้เชื่อมกับ Blynk โดยใช้เป็น token และ pin ของ Blynk จากนั้นกด Deploy ก็จะสามารถรับส่งข้อความโดย MQTT ได้

การรับส่งข้อมูล



ใน Blynk เรา set Datastreams ไว้ว่าให้ PIN V0 รับข้อมูล จากนั้นในโค้ดของเราจะใส่ Blynk.virtualWrite(V0, trash); เพื่อบอกว่าจะส่ง ค่าใน trash ผ่าน PIN V0 ไปให้ Blynk ส่วนใน Node Red ก็จะรับ URL ของ Blynk มา โดยใช้ URL รูปแบบนี้

GET https://{server_address}/external/api/get?token={token}&{pin}

Get Datastream value

This endpoint allows you to get the stored value of the Datastream by pin type and pin.
Example:

```
https://blynk.cloud/external/api/get?
token=Rps15JICmtRVbFyS_95houlLbm6xIQ2L&v1
```