เสาไฟอัจฉริยะ (smart electric pole)

Final Project Report

2110366 Embedded System Laboratory

สมาชิกกลุ่ม Inbedded

6331337121 Phurin Taengsriwan

6331315321 Theerachot Dejsuwannakij

6332004321 Jinnapat Yana

6332006621 Chanathip Sombuttong

Introduction

เสาไฟฟ้าที่มีอยู่ในปัจจุบันทำหน้าที่เพียงให้แสงสว่างกับผู้คนที่เดินผ่านไปผ่านมาในเวลากลางคืนหรืออย่างมากก็สามารถให้แสงสว่างเฉพาะตอนที่มีคนเดินผ่านเพื่อเป็นการประหยัดไฟเท่านั้น ซึ่งคณะผู้จัดทำมองว่าเสาไฟ 1 ต้นสามารถทำอะไรได้มากกว่านั้น จึงได้ริเริ่มออกแบบแบบจำลองเสาไฟที่มีประสิทธิภาพที่สามารถสั่งการผ่านทางระบบอินเตอร์เน็ทได้

ผลงานชิ้นนี้สามารถทำงานได้หลากหลายรูปแบบ ซึ่งการสั่งงานทั้งหมดจะเกิดขึ้นบนเว็บไซต์ โดยเบื้องต้น เสาไฟจะตรวจสอบว่าสิ่งแวดล้อมโดยรอบมีความสว่างมากน้อยแค่ไหน (ซึ่งเราสามารถกำหนดระดับความสว่างที่ต้องการให้ไฟติดได้) ถ้ามืดกว่าที่กำหนดหลอดไฟจะติดทันที นอกจากนี้เรายังสามารถระบุเวลาที่หลอดไฟจะติดได้โดยไม่ต้องรอให้บรรยากาศโดยรอบมืดลง ผลงานชิ้นนี้ยังสามารถนับจำนวนสิ่งที่เคลื่อนที่ผ่านมันไปได้ (สามารถสั่งการให้นับหรือไม่นับได้อีกด้วย)

อุปกรณ์ที่ใช้

Microcontroller

- NUCLEO-F411RE
- NodeMCU ESP8266

Sensors

- Ultrasonic Sensor
- LDR Sensor

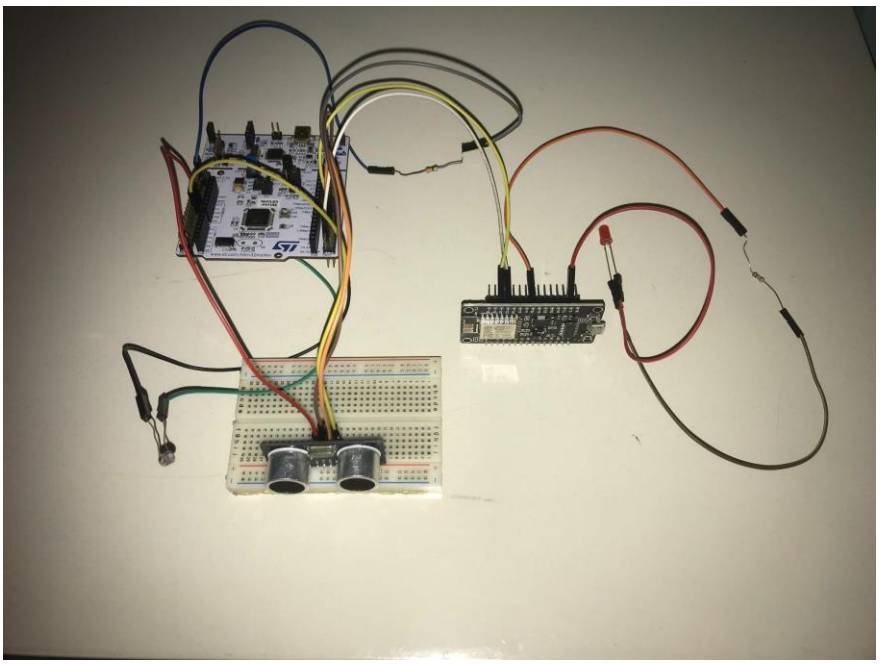
อุปกรณ์อื่นๆ

- หลอดไฟ LED
- สายไฟ
- resistor 2 ตัว

เครื่องมือที่ใช้ทำเว็บ

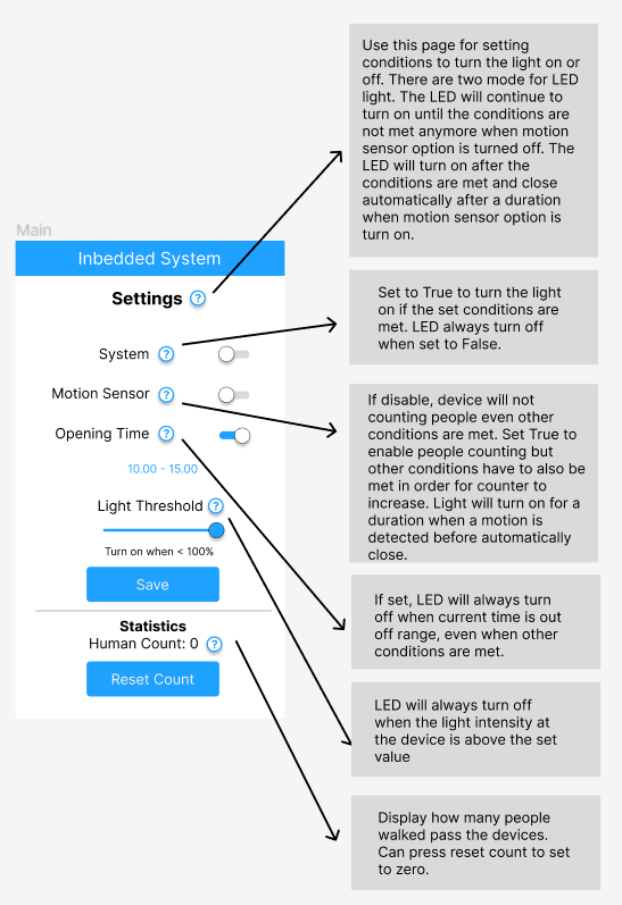
- firebase
- figma
- flutter

ภาพผลงาน



การออกแบบตัวผลงานและเว็บไซต์

1.ออกแบบหน้าเว็บ

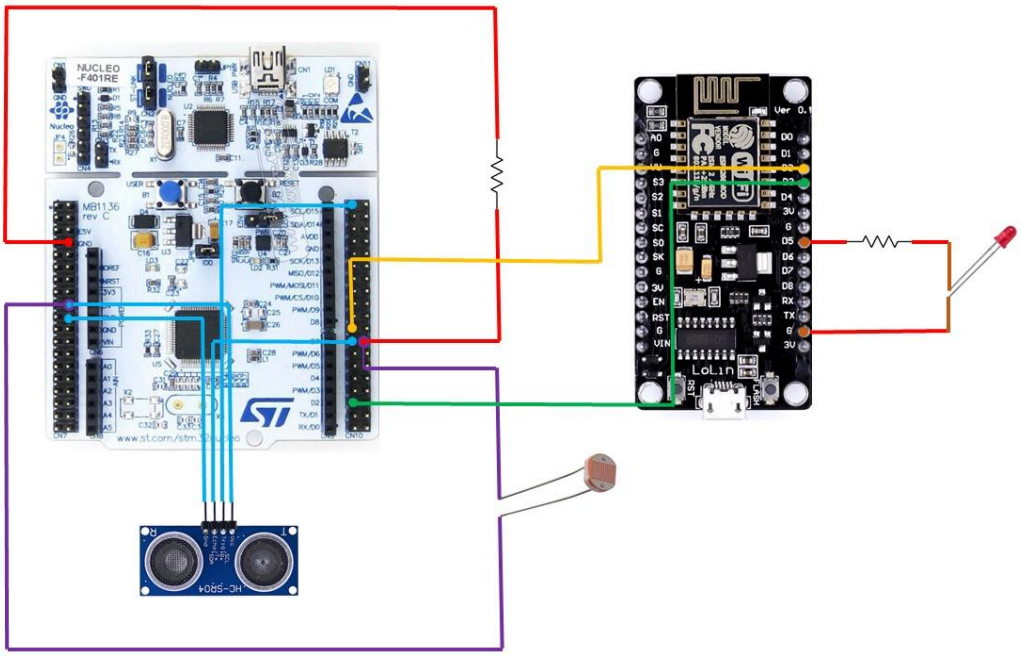


ตัวเว็บออกแบบโดยใช้ figma และพัฒนาเว็บโดยใช้ flutter เป็นเครื่องมือช่วยออกแบบซึ่งรายละเอียดของตัวเว็บเป็นไปดังภาพด้านบน โดยองค์ประกอบต่างๆในเว็บมีดังนี้

1. ตัว header จะเป็นชื่อกลุ่มคือ “Inbedded System”
2. ตัวเลือก system คือ ตัวสั่งเปิด/ปิดการทำงานทั้งหมด ถ้าเป็น true จะทำงานตามเงื่อนไข แต่ ถ้าตั้งค่าเป็น false หลอดไฟจะปิดเสมอ
3. ตัวเลือก Motion Sensor ถ้าเลือกตัวเลือกนี้จะตัวผลงานจะทำการนับจำนวนคนที่เดินผ่าน แต่ถ้าไม่เลือกก็จะไม่มีการนับเกิดขึ้น
4. ตัวเลือก Opening Time ใช้สำหรับเลือกว่าจะตั้งเวลาเปิดปิดหรือไม่ ถ้าเลือก หลอดไฟจะติดตามเวลาที่กำหนด แต่ถ้าไม่หลอดไฟจะติดเมื่อตรงกับเงื่อนไขอื่นๆ
5. แถบ Light Threshold ใช้กำหนดระดับความสว่างที่จะทำให้หลอดไฟติด
6. Block ด้านล่างจะบอกจำนวนคนที่เดินผ่านตามเงื่อนไขซึ่งสามารถ reset ค่าได้

2.ออกแบบตัวผลงาน

การออกแบบวงจรเป็นดังภาพ



ตัวบอร์ด STM32 จะเป็นตัวรับ input ทั้งหมดทั้งจาก ultrasonic sensor ซึ่งใช้สำหรับนับจำนวนคนที่เดินผ่านและจาก LDR sensor ซึ่งเป็นตัววัดความสว่าง(เพื่อเช็คว่าบรรยากาศโดยรอบมืดแล้วหรือยัง

บอร์ด STM32 และ NodeMCU จะสื่อสารโดยใช้ UART ส่งตัวค่าต่างๆที่ได้จาก STM32 มาให้ NodeMCU เพื่อประมวลผลต่อไป

NodeMCU จะเป็นบอร์ดที่ใช้ประเมินว่าควรเปิดหลอดไฟหรือยัง บอร์ดตัวนี้จะเชื่อมต่อกับเว็บที่ทำขึ้นเพื่อรับเงื่อนไขการเปิด/ปิดไฟ บอร์ดนี้ยังรับค่าจากบอร์ด STM32 เพื่อรับค่าความสว่างและประเมินว่าควรเปิดหลอดไฟหรือไม่ และถ้ามีคนเดินผ่านก็จะมีการส่งข้อมูลเพื่ออัพเดทจำนวนคนที่เดินผ่านไปยังหน้าเว็บด้วย ตัวหลอดไฟจะต่อกับบอร์ดตัวนี้เพื่อให้สามารถสั่งการเปิด/ปิดได้ทันทีเมื่อมีเงื่อนไขที่ตรงกัน

การโยงสายไฟในวงจร

สายไฟที่เชื่อมต่อกับ ultrasonic sensor

- VCC ต่อกับ +5V ของบอร์ด
- trig ต่อกับ PC9 ของบอร์ด
- echo ต่อกับ PA8 ของบอร์ด
- GND ต่อกับ GND ของบอร์ด

สายไฟที่เชื่อมต่อกับ LDR sensor

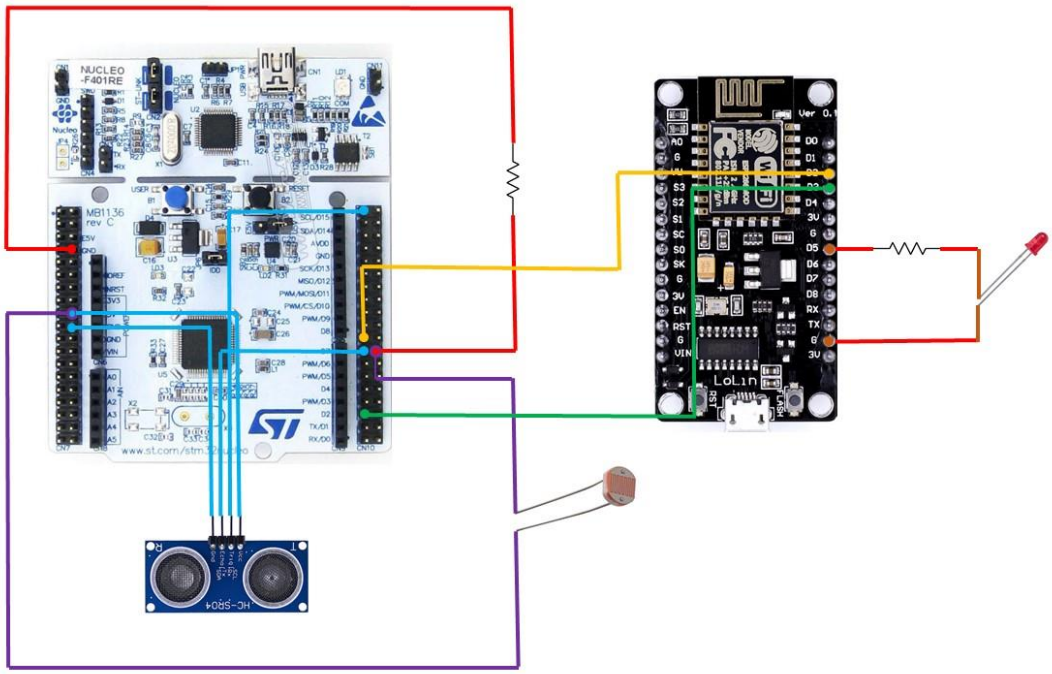
- resister ต่อกับ PB1 และ GND ของบอร์ด
- LDR ต่อกับ PB1 และ +5V ของบอร์ด

สายไฟที่ใช้สื่อสาร uart ระหว่าง 2 บอร์ด

- PA9 ของบอร์ด STM32 ต่อกับ D2 ของบอร์ด nodemcu
- PA10 ของบอร์ด STM32 ต่อกับ D3 ของบอร์ด nodemcu

สายไฟที่เชื่อมกับหลอดไฟ LED

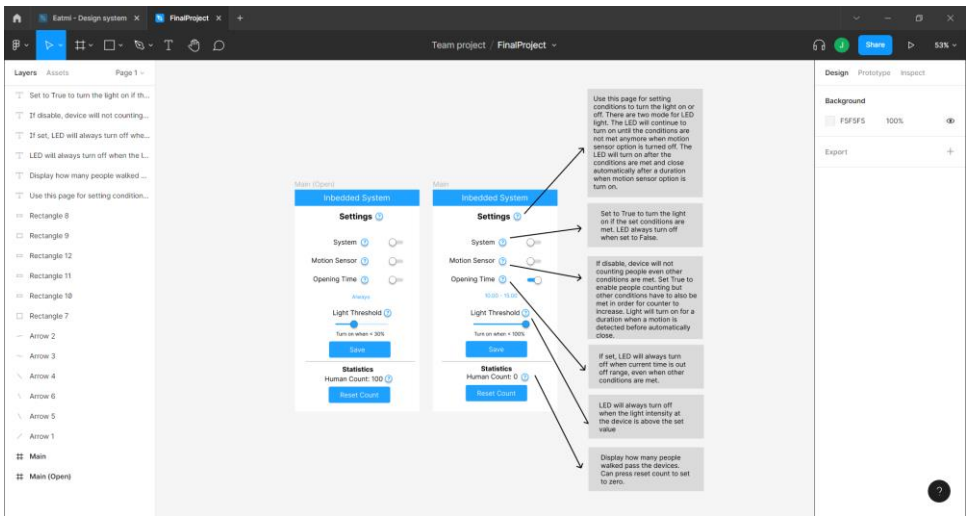
- D5 ของบอร์ดต่อกับ resister ซึ่งต่อกับหลอดไฟ LED
- LED อีกข้างต่อกับ GND



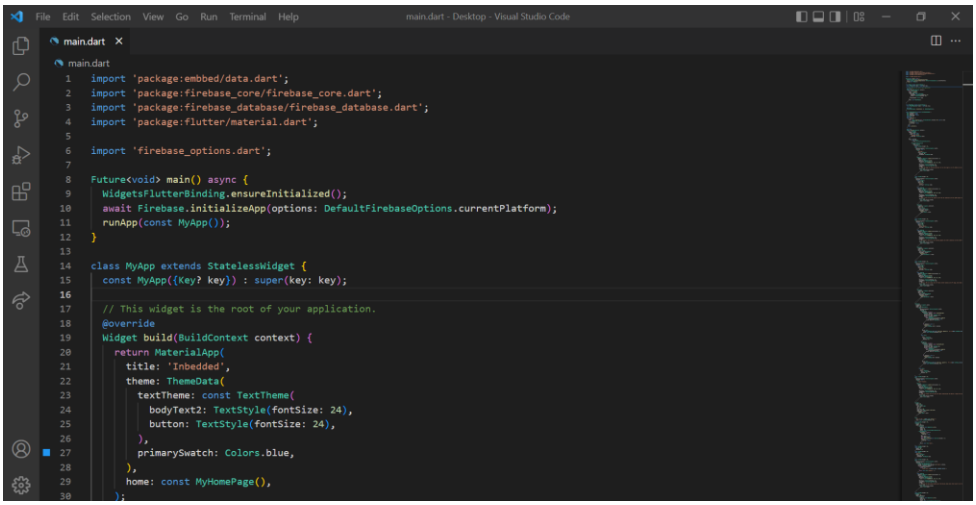
การแบ่งหน้าที่ของสมาชิกกลุ่ม

Phurin Taengsriwan frontend developer

รับหน้าที่ออกแบบ User Interface ของหน้าเว็บโดยใช้โปรแกรม Figma รวมถึงเป็นคนคิดคำอธิบายของการตั้งค่าและวิธีการใช้งานต่าง ๆ จากนั้นก็เป็นคนที่พัฒนาด้าน Front-end ของเว็บโดยใช้ภาษา Dart และ Flutter เป็นเครื่องมือในการเขียนพัฒนาหน้าเว็บตามที่ได้ออกแบบไว้ในโปรแกรม Figma โดยหน้าเว็บจะแบ่งออกเป็นสองส่วนคือส่วนการตั้งค่า (ด้านบน) และส่วนการแสดงค่าสถิติ (ด้านล่าง) เมื่อเขียนหน้าเว็บเสร็จแล้วก็รับหน้าที่ deploy หน้าเว็บไปที่ Firebase hosting รวมถึงยังเป็นคนที่ทำหน้าที่ทดสอบความถูกต้องในการทำงานของ User Interface ที่ออกแบบมาในอุปกรณ์ต่าง ๆ อย่างเช่น มือถือ คอมพิวเตอร์ แท็บเล็ต

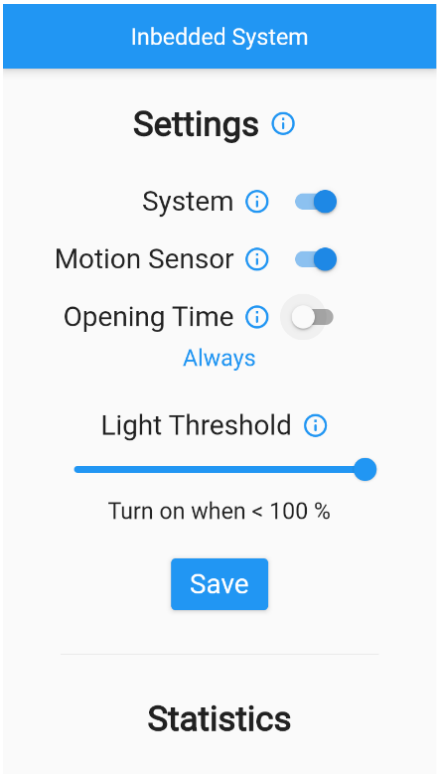


ภาพการออกแบบหน้าเว็บโดยใช้โปรแกรม Figma



```dart
import 'package:embbed/data.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';

import 'firebase_options.dart';

Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Inbedded',
      theme: ThemeData(
        textTheme: const TextTheme(
          bodyText2: TextStyle(fontSize: 24),
          button: TextStyle(fontSize: 24),
        ),
        primarySwatch: Colors.blue,
      ),
      home: const MyHomePage(),
    );
```

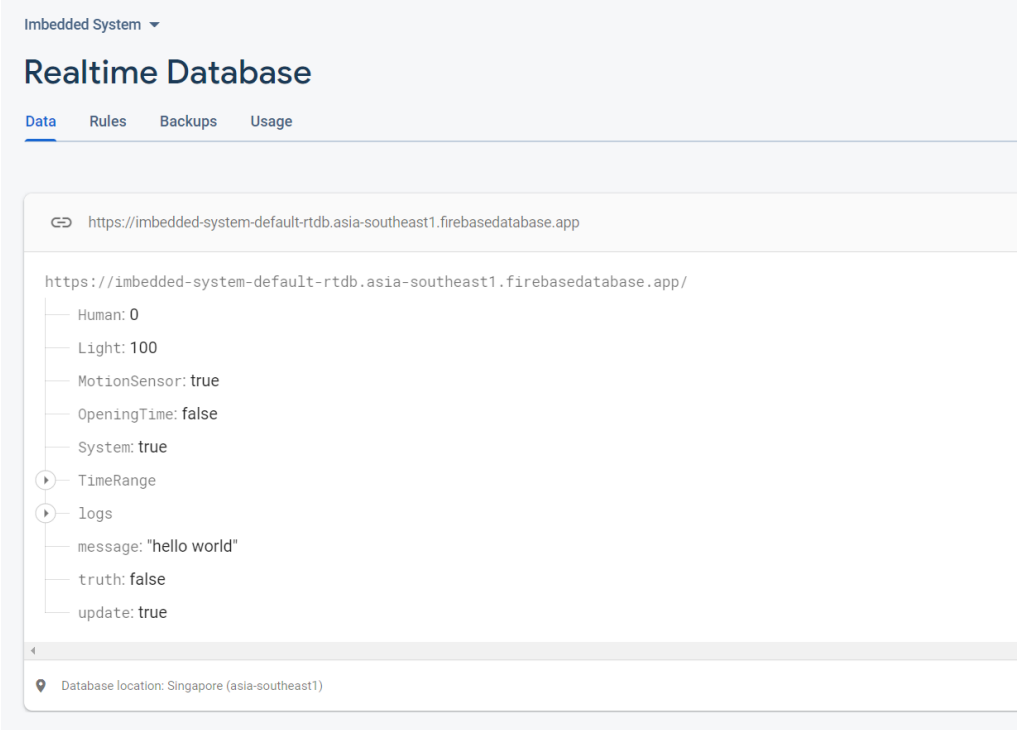
ภาพโค้ดภาษา Dart ที่ใช้ในการเขียนหน้าเว็บ



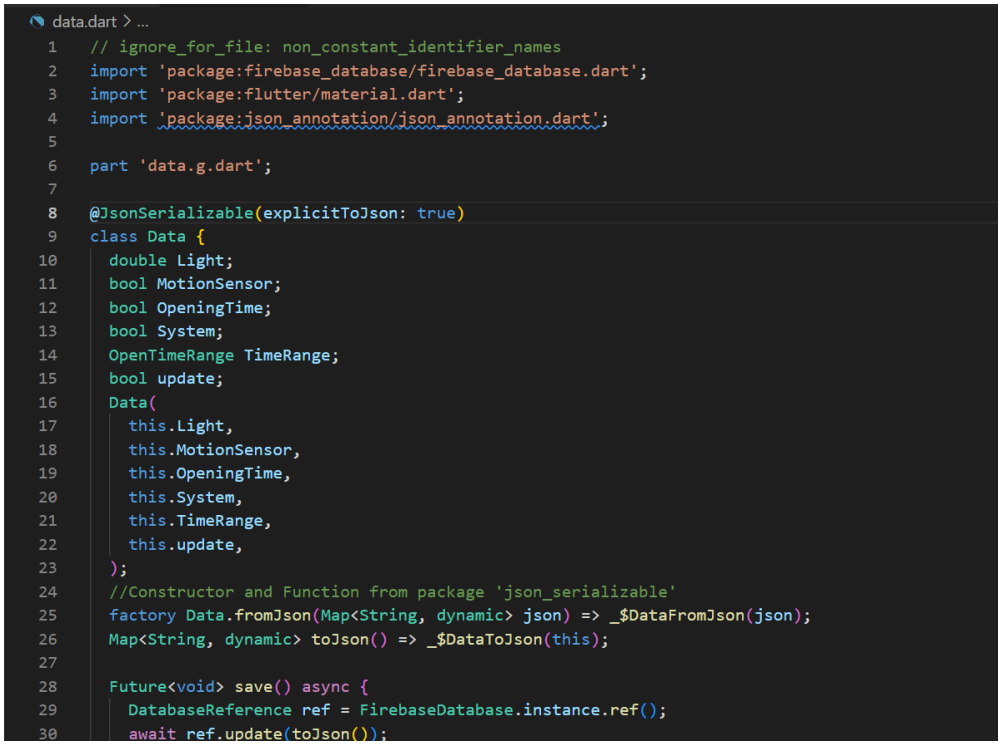
ภาพการทดสอบ User Interface ในมือถือ

Jinnapat Yana backend developer

รับหน้าที่ในการออกแบบโครงสร้างการเก็บข้อมูลในฐานข้อมูล รวมถึงการออกแบบวิธีที่จะทำให้มีการส่งข้อมูลระหว่างอุปกรณ์และเซิร์ฟเวอร์อย่างมีประสิทธิภาพมากขึ้น เช่น การออกแบบระบบ update flag เพื่อให้อุปกรณ์ไม่ต้องอ่านการตั้งค่าทั้งหมดจากฐานข้อมูลทุก ๆ ครั้ง แต่ให้อ่านเฉพาะครั้งที่ flag update เป็น true เท่านั้นเป็นต้น นอกจากนี้ก็รับผิดชอบหน้าที่ในการเชื่อมหน้าเว็บฝั่ง Front-end ให้สามารถแก้ไขและอ่านข้อมูลจากฐานข้อมูล Realtime Database ของ Firebase ได้โดยการใช้ Firebase SDK Flutter และภาษา Dart ในการพัฒนาระบบ จากนั้นก็รับหน้าที่ในการทดสอบความถูกต้องในการทำงานของระบบที่ทำหน้าติดต่อกับฐานข้อมูล



ภาพการออกแบบโครงสร้างการเก็บข้อมูลในฐานข้อมูล



ภาพการเขียนโปรแกรมเชื่อมฝั่ง Front-end กับ Realtime Database

Theerachot Dejsuwannakij Nodemcu และ UART สื่อสารระหว่าง 2 บอร์ด

ทำหน้าที่ออกแบบและคิดวิธีการในการสื่อสารข้อมูลจาก Firebase มาสู่ตัวบอร์ดโดยใช้ไลบรารี่ Firebase Arduino และเขียนโปรแกรมเชื่อมต่อ NodeMCU เข้ากับ WiFi รวมถึงคิดวิธีในการสื่อสารข้อมูลระหว่าง NodeMCU และบอร์ด STM32 เพื่อให้สามารถนำค่าจากเซ็นเซอร์ของทั้งสองบอร์ดมาประมวลผลรวมกันที่ NodeMCU นอกจากนี้ยังเป็นคนเขียนโปรแกรมเพื่อติดต่อขอข้อมูลเวลาปัจจุบันจากเซิร์ฟเวอร์ NTP และเมื่ออุปกรณ์สามารถที่จะอ่านข้อมูลจากฐานข้อมูล Realtime Database และสามารถรู้เวลาปัจจุบันได้แล้วก็ลงมือเชื่อมต่อเงื่อนไขต่าง ๆ ที่ทำให้ไฟติดดับ ซึ่งทั้งหมดจะทำการประมวลผลบนบอร์ด NodeMCU ESP8266 เมื่อเปรียบเทียบได้แล้วจึงเขียนโปรแกรมที่ใช้สื่อสารระหว่าง NUCLEO-F411RE กับ NodeMCU ESP8266 โดยใช้ UART และต่อตัวบอร์ด NodeMCU เข้ากับหลอดไฟ LED หลังจากนั้นลองทดสอบระบบร่วมกับส่วนอื่น ๆ ว่าทำงานตรงกับที่ต้องการหรือไม่

```
void receiveInfo(){
  static bool receivingInProgress = false;
  static byte index = 0;
  char startMarker = '<';
  char endMarker = '>';
  char rc;

  while (NodeSerial.available() > 0 && newData == false){
    rc = NodeSerial.read();

    //Find Start
    if (rc == startMarker){
      receivingInProgress = true;
    }
    //Read Information
    else if (rc != endMarker){
      receivedInformation[index] = rc;
      index++;
      if (index >= maxInts){
        index = maxInts - 1;
      }
    }
    //Find End
    else if (rc == endMarker){
      receivedInformation[index] = '\0';
      receivingInProgress = false;
      index = 0;
      newData = true;
    }
  }
}
```

ภาพโค้ดที่ใช้ uart สื่อสารระหว่างบอร์ด

```
void CountHuman(){
  if(Firebase.getBool("System") && InRange && Firebase.getBool("MotionSensor") && prev_d!=d){
    if(d!=0 && d<50 && 1.0*r/4100 <Firebase.getFloat("Light")){
      human++;
      delayTime = 5000;
      lastTime = millis();
    }
  }
  prev_d=d;
}
void BlinkLed(){
  if(Firebase.getBool("System") && InRange){
    if(Firebase.getBool("MotionSensor") && d!=0 && d<50 && prev_d!=d && (1.0*r/4100)*100 <Firebase.getFloat("Light")){
      digitalWrite(led,HIGH);
      human++;
      delayTime = 5000;
      lastTime = millis();
      while(Firebase.getBool("System") && Firebase.getBool("MotionSensor") && millis()-lastTime<delayTime && (1.0*r/4100)*100 <Firebase.getFloat("Light")){
        loop1();
        CountHuman();
      }
      digitalWrite(led,LOW);
    }
    else if(!Firebase.getBool("MotionSensor") && (1.0*r/4100)*100 <Firebase.getFloat("Light")){
      while((1.0*r/4100)*100 <Firebase.getFloat("Light") && Firebase.getBool("System") && checkTime()&& !Firebase.getBool("MotionSensor")){
        digitalWrite(led,HIGH);
        loop1();
      }
      digitalWrite(led,LOW);
    }
  }
```

ภาพโค้ดที่ใช้สื่อสารกับเว็บ

Chanathip Sombuttong STM32

ทำหน้าที่ในการหาวิธีรับข้อมูลของสิ่งแวดล้อมโดยรอบทั้งค่าความสว่างและตรวจสอบว่าในช่วงเวลาต่าง ๆ มีคนเดินผ่านไปผ่านมาหรือไม่ ซึ่งเลือกใช้วิธีการเก็บข้อมูลต่างๆที่ได้จาก sensor 2 ตัว คือ Ultrasonic Sensor และ LDR Sensor มาเก็บที่ บอร์ด NUCLEO-F411RE เพื่อนำข้อมูลส่งให้ NodeMCU ESP8266 ประมวลผล หลังจากนั้นได้ลองทดสอบร่วมกับส่วนอื่นๆว่าทำงานถูกต้องหรือไม่

```
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
{
    if (htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1)  // if the interrupt source is channel1
    {
        if (Is_First_Captured==0) // if the first value is not captured
        {
            IC_Val1 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1); // read the first value
            Is_First_Captured = 1;  // set the first captured as true
            // Now change the polarity to falling edge
            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1, TIM_INPUTCHANNELPOLARITY_FALLING);
        }

        else if (Is_First_Captured==1)   // if the first is already captured
        {
            IC_Val2 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);  // read second value
            __HAL_TIM_SET_COUNTER(htim, 0);  // reset the counter

            if (IC_Val2 > IC_Val1)
            {
                Difference = IC_Val2-IC_Val1;
            }

            else if (IC_Val1 > IC_Val2)
            {
                Difference = (0xffff - IC_Val1) + IC_Val2;
            }

            Distance = Difference * .034/2;
            Is_First_Captured = 0; // set it back to false

            // set polarity to rising edge
            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1, TIM_INPUTCHANNELPOLARITY_RISING);
            __HAL_TIM_DISABLE_IT(&htim1, TIM_IT_CC1);
        }
    }
}
```

ภาพโค้ดที่ใช้เก็บข้อมูลจาก ultrasonic sensor

```c
char buffer[32];
while (1)
{
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
    HCSR04_Read();
    HAL_ADC_Start(&hadc1);
      if (HAL_ADC_PollForConversion(&hadc1,1000000) == HAL_OK){
          readValue = HAL_ADC_GetValue(&hadc1);
      }
    sprintf(buffer,"<%d,%d>",readValue,Distance);
    HAL_UART_Transmit(&huart1,buffer,strlen(buffer),100);
    HAL_UART_Transmit(&huart2,buffer,strlen(buffer),100);
    HAL_Delay(1000);

}
/* USER CODE END 3 */
```

ภาพโค้ดที่ใช้เก็บข้อมูลจาก LDR light sensor