Department of Engineering

The Hong Kong Institute of Vocational Education (IVE)

MBS4522 IoT Applications

Lab 2: Smart Car Park (v1.1)

Instructions:

Please follow the detailed lab manual and complete all of them. You need to submit a lab report that shows the screenshot results of major steps and answer questions if any. Please also include your copyable source code (e.g. C, C++, Arduino) at the end of the lab report. Here are some instructions:

1. Write down all your answers/screenshot to a PDF file. (Do not copy the problems/lab descriptions) Please ensure the answers/screenshot is readable and clear enough.
2. Although discussion is permitted, the solution must be written by yourself. Plagiarism will not be tolerated. Otherwise, you will get zero mark.
3. Due at 17:30pm, April 30, 2023. Late submission is capped at 70% of the original marks. It will be zero mark if it lates 48 hours or above.

Note:

1. If you are facing a problem, please try to deal it by yourself first. Debugging, problem-solving skill and self-learning are the most important learning points. Take the chance, do not give up easily. Otherwise, you learn nothing!
2. Do not forget that Google, Stack overflow, GitHub are your friends. Try it before you give up or ask someone.

Software tools:

1. Visual Studio Code (VS code)
2. PlatformIO IDE (an extension of VS code)
3. Node-RED
4. MQTT mosquitto

Hardware tools:

1. ESP8266 x1
2. RFID–RC522 x1
3. 0.96” OLED x1
4. Router x1

Lab 2 Report

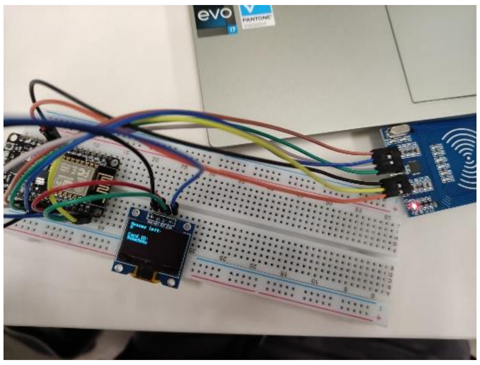
Group: D

Name: Yu Hoi Lam, Cheung Tsz Chun Noddy, Lai Ho, Lai Ka Ming

ID: 220135612, 220171174, 220668951, 220140474

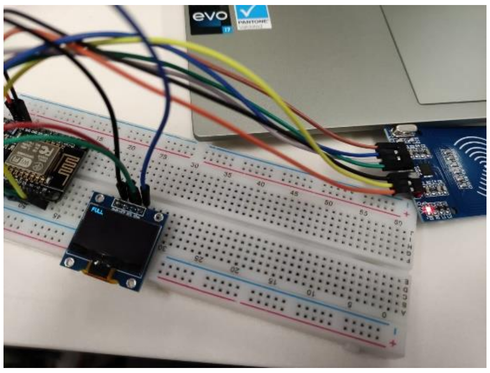
Task:

a) Build the record system. (Code a program only which includes below functions) (Total: 65%)
   i. The default parking lot is 5. Display the remaining parking spaces on the OLED screen. When recognizing a new ID, it should subtract one. If it gets the same ID as before, it means the car is leaving. Then, the remaining number needs to be added by one. (20%) When the number is zero, it should show "FULL". (10%) (Total: 30%)
   ii. When user tap the card, the RFID card reader get the card ID and print it on serial monitor. (20%)
   iii. Show the card ID on OLED screen. Do not show the ID and keep "FULL" on display if it is full. (15%)



When the Parking system is not full, it shows remaining parking spaces and RFID card reader gets its card ID and print it out on serial monitor.

In this picture, since a card is tapped on the RFID card reader, the space count subtracts one and shows "4" remaining space, and shows the card ID on screen.



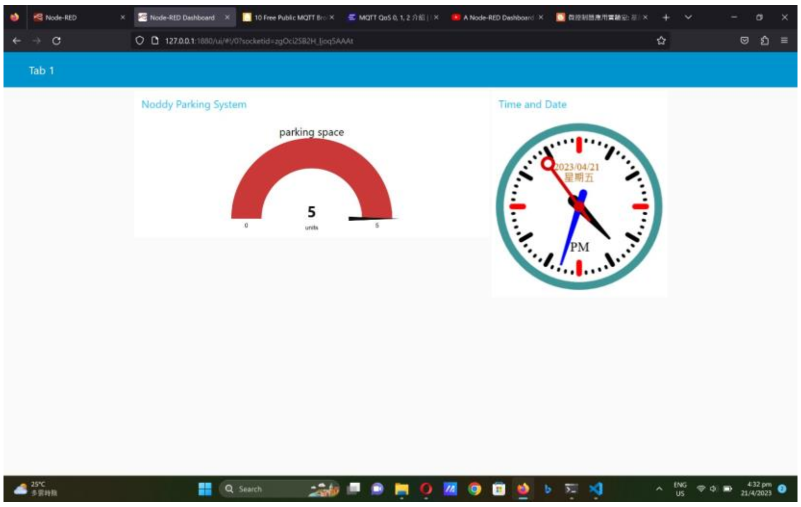
When the Parking system is full, it shows "FULL" on screen and does not show ID. Even a ID tapped the RFID sensor, it keeps displaying full and does not record the ID of new cards tapped after full state.

Github Code: https://github.com/noddycheung/IoT-lesson-2223/blob/main/lab%202%20Q1

Source Code:

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SPI.h>
#include <MFRC522.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#define SS_PIN 15
#define RST_PIN 16
MFRC522 rfid(SS_PIN, RST_PIN);

const int maxParkingSpaces = 5;
int remainingSpaces = maxParkingSpaces;
bool isFull = false;
#include <set>
std::set<String> parkedCars;

void setup() {
  Serial.begin(9600);
  SPI.begin();
  rfid.PCD_Init();
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  display.println("Noddy Parking System");
  display.display();
  delay(1000);
  display.clearDisplay();
}

void loop() {
  if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial()) {
    return;
  }

  String cardID = "";
  for (byte i = 0; i < rfid.uid.size; i++) {
    cardID += String(rfid.uid.uidByte[i], HEX);
  }

  if (parkedCars.find(cardID) != parkedCars.end()) {
    parkedCars.erase(cardID);
    if (isFull) {
      isFull = false;
    }
    remainingSpaces++;
  } else if (!isFull) {
    parkedCars.insert(cardID);
    remainingSpaces--;
    if (remainingSpaces == 0) {
      isFull = true;
    }
  }

  display.clearDisplay();
  display.setCursor(0, 0);

  if (isFull) {
    display.println("FULL");
  } else {
    display.println("Spaces left: ");
    display.println(remainingSpaces);
    display.println();
    display.println("Card ID: ");
    display.println(cardID);
  }

  display.display();

  Serial.print("Card ID: ");
  Serial.println(cardID);

  rfid.PICC_HaltA();
  rfid.PCD_StopCrypto1();
}
```

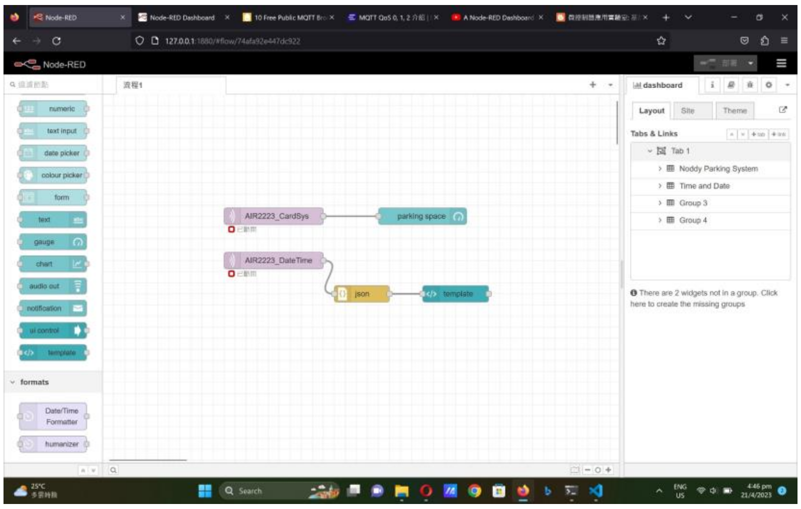
b) Using Node-RED to build a website. It should fulfil below functions or requirements. (Total: 35%)
   i. Communication with ESP8266 and related sensors by MQTT
   ii. Display the real time remaining parking spaces on website (20%)
   iii. Display the live time and date on website (10%)
   iv. Nice outlook of website (5%)



It synchronise parking space count and time and date on the dashboard.

For example, when remaining parking slot is 5, it shows 5 on dashboard.

We used reference online to make the clock that shows date and current time.
Reference: https://youtu.be/AINXJ2wN--0



The connection is as above.

Github Code: https://github.com/noddycheung/IoT-lesson-2223/blob/main/Q2%20wifi%20attempt

Source Code:

```
#include <Wire.h>

#include <Adafruit_GFX.h>

#include <Adafruit_SSD1306.h>

#include <SPI.h>

#include <MFRC522.h>
```

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

// Change the credentials below, so your ESP8266 connects to your router
const char* ssid = "Linksys03688";
const char* password = "ivelwl2022";



// Change the variable to your Raspberry Pi IP address, so it connects to your MQTT broker
const char* mqtt_server = "test.mosquitto.org";
//For example
//const char* mqtt_server = "192.168.1.106";

// Initializes the espClient. You should change the espClient name if you have multiple ESPs
running in your home automation system
WiFiClient espClient;
PubSubClient client(espClient);



#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#define SS_PIN 15
#define RST_PIN 16
MFRC522 rfid(SS_PIN, RST_PIN);

// Timers auxiliar variables
long now = millis();
long lastMeasure = 0;
```

```
const int maxParkingSpaces = 5;
int remainingSpaces = maxParkingSpaces;
bool isFull = false;
#include <set>
std::set<String> parkedCars;

// This functions connects your ESP8266 to your router
void setup_wifi() {
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("WiFi connected - ESP IP address: ");
  Serial.println(WiFi.localIP());
}

// This function is executed when some device publishes a message to a topic that your ESP8266 is subscribed to
// Change the function below to add logic to your program, so when a device publishes a message to a topic that
// your ESP8266 is subscribed you can actually do something
void callback(String topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
  }
  Serial.println();

  // Feel free to add more if statements to control more GPIOs with MQTT

  Serial.println();
}

// This functions reconnects your ESP8266 to your MQTT broker
// Change the function below if you want to subscribe to more topics with your ESP8266
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    /*
     YOU MIGHT NEED TO CHANGE THIS LINE, IF YOU'RE HAVING PROBLEMS WITH MQTT
MULTIPLE CONNECTIONS
     To change the ESP device ID, you will have to give a new name to the ESP8266.
     Here's how it looks:
       if (client.connect("ESP8266Client")) {
     You can do it like this:
       if (client.connect("ESP1_Office")) {
     Then, for the other ESP:
       if (client.connect("ESP2_Garage")) {
      That should solve your MQTT multiple connections problem
    */
    if (client.connect("ESP8266Client", "", "")) {
      Serial.println("connected");
      // Subscribe or resubscribe to a topic
      // You can subscribe to more topics (to control more LEDs in this example)
      client.subscribe("AIR2223_CardSys");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}


void setup() {
  Serial.begin(9600);

  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);

  SPI.begin();
  rfid.PCD_Init();

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  display.println("Noddy Parking System");
  display.display();
  delay(1000);
  display.clearDisplay();
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  if(!client.loop())
    client.connect("ESP8266Client");

  now = millis();

  if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial()) {
    return;
  }

  String cardID = "";
  for (byte i = 0; i < rfid.uid.size; i++) {
    cardID += String(rfid.uid.uidByte[i], HEX);
  }

  if (parkedCars.find(cardID) != parkedCars.end()) {
    parkedCars.erase(cardID);
    if (isFull) {
      isFull = false;
    }
    remainingSpaces++;
  } else if (!isFull) {
```

```
  parkedCars.insert(cardID);
  remainingSpaces--;

  if (remainingSpaces == 0) {
    isFull = true;
  }
}

display.clearDisplay();
display.setCursor(0, 0);

if (isFull) {
  display.println("FULL");
} else {
  display.println("Spaces left: ");
  display.println(remainingSpaces);
  display.println();
  display.println("Card ID: ");
  display.println(cardID);
}

display.display();

Serial.print("Card ID: ");
Serial.println(cardID);

rfid.PICC_HaltA();
rfid.PCD_StopCrypto1();


  //Publishes Temperature and Humidity values
```

```
    client.publish("AIR2223_CardSys", String(remainingSpaces).c_str());
}
```

c) Complete and sign the form. All team members need to agree the weighting and sign.

Name	1. Yu Hoi Lam	2. Cheung Tsz Chun, Noddy	3. Lai Ka Ming	4. Lai Ho
ID	1. 220135612	2. 220171174	3. 220140474	4. 220668951
Workload	Hardware Programming Flowchart Node-Red Layout Debug Lab Report	Hardware Programming debug Node-Red Welding of RFID card sensor and OLED monitor Lab Report	Hardware Node-Red	Node-Red
Weighting (Amount is 100%)	40	40	15	5
Signature	[illegible]	張子俊	[illegible]	Ho
Remark	Hardware testing on part 1 Adding core functions on car parking system Node-Red Layout	Hardware testing on part 1 and 2 Connection and adding extra functions on Node-Red Showing ID number on LED monitor	Attempt on hardware and connection	Attempt on connection
Adjustment (For faculty use)				

~~END~~