G.Guna Sundeep – 19R11A04M2 – ECE2E

#include <WiFi.h>

#include <PubSubClient.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, -1);

String command;

String data="";

void callback(char\* topic, byte\* payload, unsigned int payloadLength);

const char\* ssid = "Excell";

const char\* password = "sunny@79";

#define ORG "z0yo2t"

#define DEVICE\_TYPE "ESP32"

#define DEVICE\_ID "12345"

#define TOKEN "gunsun123"

const char publishTopic[] = "iot-2/evt/Data/fmt/json";

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char topic[] = "iot-2/cmd/home/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;

WiFiClient wifiClient;

PubSubClient client(server, 1883, callback, wifiClient);

int publishInterval = 5000;

long lastPublishMillis;

void publishData();

void setup() {

Serial.begin(115200);

Serial.println();

if(!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

Serial.println("SSD1306 allocation failed");

for(;;);

}

wifiConnect();

mqttConnect();

}

void loop() {

if (millis() - lastPublishMillis > publishInterval)

{

publishData();

lastPublishMillis = millis();

}

if (!client.loop()) {

mqttConnect();

}

}

void wifiConnect() {

Serial.print("Connecting to "); Serial.print(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.print("WiFi connected, IP address: ");

Serial.println(WiFi.localIP());

}

void mqttConnect() {

if(!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

Serial.println("SSD1306 allocation failed");

for(;;);

}

if (!client.connected())

{

Serial.print("Reconnecting MQTT client to "); Serial.println(server);

while (!client.connect(clientId, authMethod, token)) {

Serial.print(".");

delay(500);

}

initManagedDevice();

Serial.println();

}

}

void initManagedDevice() {

if (client.subscribe(topic)) {

Serial.println("subscribe to cmd OK");

} else {

Serial.println("subscribe to cmd FAILED");

}

}

void callback(char\* topic, byte\* payload, unsigned int payloadLength) {

Serial.print("callback invoked for topic: ");

Serial.println(topic);

for (int i = 0; i < payloadLength; i++) {

command+= (char)payload[i];

}

Serial.println("data: "+ command);

control\_func();

command= "";

}

void control\_func()

{

if(command== "Hi")

{

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(3, 25);

display.println("Hi!");

display.display();

}

else if(command== "Hru")

{

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(3, 25);

display.println("How are you?");

display.display();

}

else if(command== "fine")

{

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(3, 25);

display.println("I'm fine!");

display.display();

}

else if(command== "demo")

{

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(3, 25);

display.println("LED demo");

display.display();

}

else if(command== "esp")

{

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(3, 25);

display.println("Output from ESP");

display.display();

}

else

{

Serial.println("......no commands have been subscribed..........");

}

}

void publishData()

{

String payload = "{\"d\":{\"Message\":";

payload += command;

payload += "}}";

Serial.print("\n");

Serial.print("Sending payload: "); Serial.println(payload);

if (client.publish(publishTopic, (char\*) payload.c\_str())) {

Serial.println("Publish OK");

} else {

Serial.println("Publish FAILED");

}

}

