#include <ESP8266WiFi.h>

#include <ESP8266WebServer.h>

#include <Adafruit\_Fingerprint.h>

#include <SoftwareSerial.h>

#include <WiFiClientSecure.h>

#include <LiquidCrystal\_I2C.h>

#include <ESP8266Firebase.h>

#include "HTTPSRedirect.h"

const char\* ssid = "12346";

const char\* password = "66666666";

const char\* host = "script.google.com";//url connect to Google Apps Script

const int httpsPort = 443;//Port

String GAS\_ID = "AKfycbwZ53oZGJGLHyxY9ndAANT-UjdCp8E-KNcjYMHuEpWkLSr\_liz3aD5l5SfhzCLvyrBe5A";//Key Google Apps Script

// Enter command (insert\_row or append\_row) and your Google Sheets sheet name (default is Sheet1):

String payload\_base = "{\"command\": \"insert\_row\", \"sheet\_name\": \"Sheet1\", \"values\": ";

String payload = "";

// Google Sheets setup (do not edit)

String url = String("/macros/s/") + GAS\_ID + "/exec";

HTTPSRedirect\* client = nullptr;

#define REFERENCE\_URL "iotproject-470fe-default-rtdb.asia-southeast1.firebasedatabase.app" // Your Firebase project reference url

#define RX\_PIN D4 // Chân RX của cổng nối tiếp mềm

#define TX\_PIN D3 // Chân TX của cổng nối tiếp mềm

Firebase firebase(REFERENCE\_URL);

SoftwareSerial mySerial(RX\_PIN, TX\_PIN);

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

ESP8266WebServer server(80);

LiquidCrystal\_I2C lcd(0x27, 16, 2);

int status = 0;

int FingerID = 0;

char webpage[] PROGMEM = R"=====(

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Add User Form</title>

<style>

</style>

</head>

<body>

<div class="form-style-5 slideInDown animated">

<div class="alert">

<label id="alert"></label>

</div>

<form onsubmit="sendData(); return false;">

<fieldset>

<legend><span class="number">1</span> User Fingerprint ID:</legend>

<label>Enter Fingerprint ID between 1 & 127:</label>

<input type="number" name="fingerid" id="fingerid" placeholder="User Fingerprint ID...">

<button type="button" name="fingerid\_add" class="fingerid\_add" onclick="enrollFingerprint()">Add Fingerprint

ID</button>

</fieldset>

<fieldset>

<legend><span class="number">2</span> User Info</legend>

<input type="text" name="name" id="name" placeholder="User Name...">

<input type="text" name="number" id="number" placeholder="Serial Number...">

<input type="email" name="email" id="email" placeholder="User Email...">

</fieldset>

<button type="submit" class="user\_add">Add User</button>

</form>

<button type="button" class="attendance\_check" onclick="checkAttendance()">Attendance Check</button>

<form onsubmit="deleteFingerprint(); return false;">

ID fingerprint: <input type="text" name="txtInput" id="iddelete" value="" />

<input type="submit" name="btSubmit" value="Delete" />

</form>

<script>

var baseUrl = '/'; // Đường dẫn cơ bản, không cần địa chỉ IP

function deleteFingerprint() {

var fingerid = document.getElementById('iddelete').value;

if (!fingerid) {

console.log("Please enter a fingerprint ID.");

return;

}

var xhr = new XMLHttpRequest();

xhr.onreadystatechange = function () {

if (xhr.readyState == 4 && xhr.status == 200) {

var response = xhr.responseText;

console.log(response);

}

};

var deleteUrl = baseUrl + 'deleteFingerprint?fingerid=' + fingerid;

xhr.open('GET', deleteUrl, true);

xhr.send();

}

function sendData() {

var fingerid = document.getElementById('fingerid').value;

var name = document.getElementById('name').value;

var number = document.getElementById('number').value;

var email = document.getElementById('email').value;

var xhr = new XMLHttpRequest();

xhr.onreadystatechange = function () {

if (xhr.readyState == 4 && xhr.status == 200) {

var response = xhr.responseText;

console.log(response);

}

};

var url = baseUrl + 'senddata?fingerid=' + fingerid + '&name=' + name + '&number=' + number + '&email=' + email;

xhr.open('GET', url, true);

xhr.send();

}

function checkAttendance() {

var xhr = new XMLHttpRequest();

xhr.onreadystatechange = function () {

if (xhr.readyState == 4) {

if (xhr.status == 200) {

var response = xhr.responseText;

console.log(response);

} else {

console.log("Error: " + xhr.status);

}

}

};

var checkAttendanceUrl = baseUrl + 'checkAttendance?action=attendance';

xhr.open('GET', checkAttendanceUrl, true);

xhr.send();

}

</script>

</div>

</body>

</html>

)=====";

void handleRoot() {

server.send(200, "text/html", webpage);

}

void setup() {

lcd.init(); // initialize the lcd

lcd.init();

lcd.backlight();

Serial.begin(115200);

WiFi.begin(ssid, password);

Serial.print("Connecting");

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

Serial.print(".");

delay(500);

}

Serial.println("");

Serial.print("Successfully connected to : ");

Serial.println(ssid);

lcd.println("Success connect");

Serial.print("IP address: ");

Serial.println(WiFi.localIP());

Serial.println();

mySerial.begin(57600);

Serial.println("Waiting for the fingerprint sensor...");

if (finger.verifyPassword()) {

Serial.println("Found fingerprint sensor!");

} else {

Serial.println("Did not find fingerprint sensor :(");

while (1);

}

delay(1000);

server.on("/", HTTP\_GET, handleRoot);

server.on("/senddata", HTTP\_GET, handleSendData);

server.on("/deleteFingerprint", HTTP\_GET, handleDeleteFingerprint);

server.on("/checkAttendance", HTTP\_GET, handleCheckAttendance);

server.begin();

}

void loop() {

server.handleClient();

}

void handleDeleteFingerprint() {

String fingerid = server.arg("fingerid");

int id = fingerid.toInt();

Serial.print("Received Data: ");

Serial.print("Fingerprint ID: " + fingerid);

deleteFingerprint(id);

deleteData(fingerid);

}

void handleCheckAttendance() {

String action = server.arg("action");

if (action.equals("attendance")) {

int id = getFingerprintID();

while(id <= 0){

id = getFingerprintID();

delay(1000);

}

getData(id);

}

}

void handleSendData() {

String fingerid = server.arg("fingerid");

String name = server.arg("name");

String number = server.arg("number");

String email = server.arg("email");

int id = fingerid.toInt();

// Xử lý dữ liệu nhận được từ ESP8266

Serial.print("Received Data: ");

Serial.print("Fingerprint ID: " + fingerid);

Serial.print(", Name: " + name);

Serial.print(", Number: " + number);

Serial.print(", Email: " + email);

Serial.println();

enrollFingerprint(id);

sendData(fingerid, name, number, email);

// Thực hiện các xử lý khác nếu cần

// Phản hồi cho ESP8266 (nếu cần)

server.send(200, "text/plain", "Data received successfully");

}

void deleteData(String fingerid) {

String path ="/Enroll/" + fingerid;

firebase.deleteData(path);

}

void sendData(String fingerid, String name, String number, String email) {

String path = "/Enroll/" + fingerid;

firebase.pushString(path, fingerid);

firebase.pushString(path + "/name", name);

firebase.pushString(path + "/number", number);

firebase.pushString(path + "/email", email);

}

void getData(int fingerId) {

String path = "/Enroll/" + String(fingerId) + "/"; // Đường dẫn đến Firebase dựa trên fingerId

String name = firebase.getString(path + "name");

String name1 = extractValue(name);

String number = firebase.getString(path + "number");

String number1 = extractValue(number);

String email = firebase.getString(path + "email");

String email1 = extractValue(email);

Serial.print("Data from firebase: ");

Serial.println("fingerid=" + String(fingerId) + ", name=" + name1 + ", number=" + number1 + ", email="+email1);

Serial.println();

connectToWiFi();

++status;

updatesheet(fingerId, name1, number1, email1);

}

String extractValue(String input) {

// Tìm vị trí của dấu hai chấm

int colonIndex = input.indexOf(':');

// Tìm vị trí của dấu ngoặc kép bắt đầu và kết thúc

int startQuoteIndex = input.indexOf('"', colonIndex);

int endQuoteIndex = input.indexOf('"', startQuoteIndex + 1);

// Trích xuất chuỗi con từ vị trí bắt đầu đến kết thúc

String result = input.substring(startQuoteIndex + 1, endQuoteIndex);

return result;

}

void updatesheet(int fingerid, String name, String number, String email) {

String status1;

if(status == 1) {

status1 = "Check in";

}

if(status == 2) {

status1 = "Check out";

status = 0;

}

static bool flag = false;

while (!flag) {

client = new HTTPSRedirect(httpsPort);

client->setInsecure();

flag = true;

client->setPrintResponseBody(true);

client->setContentTypeHeader("application/json");

}

if (client != nullptr) {

if (!client->connected()) {

client->connect(host, httpsPort);

}

}

else {

Serial.println("Error creating client object!");

}

// Create json object string to send to Google Sheets

payload = payload\_base + "\"" + String(fingerid) + "," + name + "," + number + "," + email + "," + status1 + "\"}";

// Publish data to Google Sheets

Serial.println("Publishing data...");

Serial.println(payload);

if (client->POST(url, host, payload)) {

// do stuff here if publish was successful

}

else {

// do stuff here if publish was not successful

Serial.println("Error while connecting");

}

// a delay of several seconds is required before publishing again

delay(5000);

}

void enrollFingerprint(int fingerId) {

//int fingerId = -1;

Serial.println("Place your finger on the sensor...");

while (true) {

int p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.print(".");

delay(500);

continue;

default:

Serial.println("Unknown error");

return;

}

p = finger.image2Tz(1);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

continue;

default:

Serial.println("Unknown error");

return;

}

Serial.println("Remove your finger...");

delay(2000);

p = 0;

while (p != FINGERPRINT\_NOFINGER) {

p = finger.getImage();

}

Serial.println("Place your finger again...");

p = -1;

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

}

p = finger.image2Tz(2);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

continue;

default:

Serial.println("Unknown error");

return;

}

Serial.println("Creating model...");

p = finger.createModel();

if (p == FINGERPRINT\_OK) {

Serial.println("Prints matched!");

} else {

Serial.println("Fingerprints did not match");

continue;

}

p = finger.storeModel(fingerId);

if (p == FINGERPRINT\_OK) {

Serial.println("Fingerprint enrolled successfully!");

Serial.print("Finger ID: ");

Serial.println(fingerId);

return;

} else {

Serial.println("Error storing fingerprint");

}

}

}

int getFingerprintID() {

uint8\_t p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println("No finger detected");

return 0;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return -2;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

return -2;

default:

Serial.println("Unknown error");

return -2;

}

// OK success!

p = finger.image2Tz();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return -1;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return -2;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return -2;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return -2;

default:

Serial.println("Unknown error");

return -2;

}

// OK converted!

p = finger.fingerFastSearch();

if (p == FINGERPRINT\_OK) {

Serial.println("Found a print match!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return -2;

} else if (p == FINGERPRINT\_NOTFOUND) {

Serial.println("Did not find a match");

return -1;

} else {

Serial.println("Unknown error");

return -2;

}

// found a match!

Serial.print("Found ID #"); Serial.print(finger.fingerID);

Serial.print(" with confidence of "); Serial.println(finger.confidence);

return finger.fingerID;

}

uint8\_t deleteFingerprint(int id) {

uint8\_t p = -1;

p = finger.deleteModel(id);

if (p == FINGERPRINT\_OK) {

Serial.println("Deleted!");

lcd.clear();

// lcd.setCursor(0, 0);

lcd.print("Deleted");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

lcd.clear();

// lcd.setCursor(0, 0);

lcd.print("Communication error");

return p;

} else if (p == FINGERPRINT\_BADLOCATION) {

Serial.println("Could not delete in that location");

lcd.clear();

// lcd.setCursor(0, 0);

lcd.print("Could not delete");

lcd.setCursor(0,1);

lcd.print("in that location");

return p;

} else if (p == FINGERPRINT\_FLASHERR) {

Serial.println("Error writing to flash");

lcd.clear();

// lcd.setCursor(0, 0);

lcd.print("Error writing");

lcd.setCursor(0,1);

lcd.print("to flash");

return p;

} else {

Serial.print("Unknown error: 0x"); Serial.println(p, HEX);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Unknown error: 0x");

return p;

}

return p;

}

void connectToWiFi() {

WiFi.mode(WIFI\_OFF); //Prevents reconnection issue (taking too long to connect)

delay(1000);

WiFi.mode(WIFI\_STA);

Serial.print("Connecting to ");

Serial.println(ssid);

lcd.clear();

// lcd.setCursor(0, 0);

lcd.print("Connecting to ");

lcd.setCursor(1, 0);

lcd.print(ssid);

WiFi.begin(ssid, password);

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

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("Connected");

lcd.clear();

// lcd.setCursor(0, 0);

lcd.print("Connected, IP: ");

lcd.setCursor(0, 1);

lcd.print(WiFi.localIP());

Serial.print("IP address: ");

Serial.println(WiFi.localIP()); //IP address assigned to your ESP

}