**Connections**

**Train:**

* RFID v3.3 to ESP8266 3.3v
* RFID RST to ESP8266 D4
* RFID GND to ESP8266 GND
* RFID MISO to ESP8266 D6
* RFID MOSI to ESP8266 D7
* RFID SCK to ESP8266 D5
* RFID SDA to ESP8266 D8
* OLED VDD to ESP8266 3.3v
* OLED GND to ESP8266 GND
* OLED SCK to ESP8266 D1
* OLED SDA to ESP8266 D2

**Bus:**

* RFID v3.3 to ESP8266 3.3v
* RFID RST to ESP8266 D4
* RFID GND to ESP8266 GND
* RFID MISO to ESP8266 D6
* RFID MOSI to ESP8266 D7
* RFID SCK to ESP8266 D5
* RFID SDA to ESP8266 D8
* OLED VDD to ESP8266 3.3v
* OLED GND to ESP8266 GND
* OLED SCK to ESP8266 D1
* OLED SDA to ESP8266 D2

**Register office:**

* RFID v3.3 to ESP8266 3.3v
* RFID RST to ESP8266 D4
* RFID GND to ESP8266 GND
* RFID MISO to ESP8266 D6
* RFID MOSI to ESP8266 D7
* RFID SCK to ESP8266 D5
* RFID SDA to ESP8266 D8
* OLED VDD to ESP8266 3.3v
* OLED GND to ESP8266 GND
* OLED SCK to ESP8266 D1
* OLED SDA to ESP8266 D2

Note: Connect Vin and GND of all 3 nodemcu together respectively

**Code:**

**Train:**

#include <SPI.h>

#include <MFRC522.h>

#include <Arduino.h>

#include <U8g2lib.h>

#include <ESP8266WiFi.h>

#include <Firebase\_ESP\_Client.h>

#include "addons/TokenHelper.h"

#include "addons/RTDBHelper.h"

#ifdef U8X8\_HAVE\_HW\_SPI

#include <SPI.h>

#endif

#ifdef U8X8\_HAVE\_HW\_I2C

#include <Wire.h>

#endif

constexpr uint8\_t RST\_PIN = D4; // Configurable, see typical pin layout above

constexpr uint8\_t SS\_PIN = D8; // Configurable, see typical pin layout above

MFRC522 rfid(SS\_PIN, RST\_PIN); // Instance of the class

MFRC522::MIFARE\_Key key;

U8G2\_SSD1306\_128X64\_NONAME\_F\_SW\_I2C u8g2(U8G2\_R0, /\* clock=\*/ D1, /\* data=\*/ D2, /\* reset=\*/ U8X8\_PIN\_NONE); // All Boards without Reset of the Display

#define WIFI\_SSID "123456789"

#define WIFI\_PASSWORD "123456789"

#define API\_KEY "AIzaSyBuz9awICx9wnE44zeVrgGy\_JSiI51JfcE"

#define DATABASE\_URL "https://smart-billboards-using-iot-default-rtdb.firebaseio.com/"

FirebaseData fbdo;

FirebaseAuth auth;

FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;

bool signupOK = false;

String intValue;

void setup() {

Serial.begin(115200);

SPI.begin(); // Init SPI bus

rfid.PCD\_Init(); // Init MFRC522

u8g2.begin();

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

Serial.print("Connecting to Wi-Fi");

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

Serial.print(".");

delay(300);

}

Serial.println();

Serial.print("Connected with IP: ");

Serial.println(WiFi.localIP());

Serial.println();

config.api\_key = API\_KEY;

config.database\_url = DATABASE\_URL;

if (Firebase.signUp(&config, &auth, "", "")){

Serial.println("ok");

signupOK = true;

}

else{

Serial.printf("%s\n", config.signer.signupError.message.c\_str());

}

config.token\_status\_callback = tokenStatusCallback; //see addons/TokenHelper.h

Firebase.begin(&config, &auth);

Firebase.reconnectWiFi(true);

}

void loop() {

if (!rfid.PICC\_IsNewCardPresent())

return;

if (rfid.PICC\_ReadCardSerial()) {

String tag;

for (byte i = 0; i < 4; i++) {

tag += rfid.uid.uidByte[i];

}

Serial.println("Detected Card UID: " + tag);

// Check if the detected UID is in the list of authorized UIDs

delay(100);

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Verifying"); // write something to the internal memory

u8g2.sendBuffer();

delay(100);

if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 1000 || sendDataPrevMillis == 0)){

sendDataPrevMillis = millis();

if (Firebase.RTDB.setString(&fbdo, "smarttravel/check1", tag)){

Serial.println("PATH: " + fbdo.dataPath());

Serial.println("TYPE: " + fbdo.dataType());

}

else {

Serial.println("Failed REASON: " + fbdo.errorReason());

}

// transfer internal memory to the display

delay(4000);

if (Firebase.RTDB.getString(&fbdo, "/smarttravel/access1"))

{

intValue = fbdo.stringData();

String mySubString = intValue.substring(2, 3);

Serial.println(intValue);

Serial.println(mySubString);

if (mySubString == "a")

{

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Access Approved"); // write something to the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(5000);

u8g2.clearBuffer(); // clear the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(100);

}

else if (mySubString == "b")

{

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Access Denied"); // write something to the internal memory

u8g2.drawStr(0,40,"Please check your"); // write something to the internal memory

u8g2.drawStr(0,50,"balance"); // write something to the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(5000);

u8g2.clearBuffer(); // clear the internal memory

u8g2.sendBuffer();

delay(100);

}

delay(100);

}

else {

Serial.println(fbdo.errorReason());

}

delay(100);

}

rfid.PICC\_HaltA();

rfid.PCD\_StopCrypto1();

}

}

**Bus:**

#include <SPI.h>

#include <MFRC522.h>

#include <Arduino.h>

#include <U8g2lib.h>

#include <ESP8266WiFi.h>

#include <Firebase\_ESP\_Client.h>

#include "addons/TokenHelper.h"

#include "addons/RTDBHelper.h"

#ifdef U8X8\_HAVE\_HW\_SPI

#include <SPI.h>

#endif

#ifdef U8X8\_HAVE\_HW\_I2C

#include <Wire.h>

#endif

constexpr uint8\_t RST\_PIN = D4; // Configurable, see typical pin layout above

constexpr uint8\_t SS\_PIN = D8; // Configurable, see typical pin layout above

MFRC522 rfid(SS\_PIN, RST\_PIN); // Instance of the class

MFRC522::MIFARE\_Key key;

U8G2\_SSD1306\_128X64\_NONAME\_F\_SW\_I2C u8g2(U8G2\_R0, /\* clock=\*/ D1, /\* data=\*/ D2, /\* reset=\*/ U8X8\_PIN\_NONE); // All Boards without Reset of the Display

#define WIFI\_SSID "123456789"

#define WIFI\_PASSWORD "123456789"

#define API\_KEY "AIzaSyBuz9awICx9wnE44zeVrgGy\_JSiI51JfcE"

#define DATABASE\_URL "https://smart-billboards-using-iot-default-rtdb.firebaseio.com/"

FirebaseData fbdo;

FirebaseAuth auth;

FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;

bool signupOK = false;

String intValue;

void setup() {

Serial.begin(115200);

SPI.begin(); // Init SPI bus

rfid.PCD\_Init(); // Init MFRC522

u8g2.begin();

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

Serial.print("Connecting to Wi-Fi");

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

Serial.print(".");

delay(300);

}

Serial.println();

Serial.print("Connected with IP: ");

Serial.println(WiFi.localIP());

Serial.println();

config.api\_key = API\_KEY;

config.database\_url = DATABASE\_URL;

if (Firebase.signUp(&config, &auth, "", "")){

Serial.println("ok");

signupOK = true;

}

else{

Serial.printf("%s\n", config.signer.signupError.message.c\_str());

}

config.token\_status\_callback = tokenStatusCallback; //see addons/TokenHelper.h

Firebase.begin(&config, &auth);

Firebase.reconnectWiFi(true);

}

void loop() {

if (!rfid.PICC\_IsNewCardPresent())

return;

if (rfid.PICC\_ReadCardSerial()) {

String tag;

for (byte i = 0; i < 4; i++) {

tag += rfid.uid.uidByte[i];

}

Serial.println("Detected Card UID: " + tag);

// Check if the detected UID is in the list of authorized UIDs

delay(100);

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Verifying"); // write something to the internal memory

u8g2.sendBuffer();

delay(100);

if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 1000 || sendDataPrevMillis == 0)){

sendDataPrevMillis = millis();

if (Firebase.RTDB.setString(&fbdo, "smarttravel/check2", tag)){

Serial.println("PATH: " + fbdo.dataPath());

Serial.println("TYPE: " + fbdo.dataType());

}

else {

Serial.println("Failed REASON: " + fbdo.errorReason());

}

// transfer internal memory to the display

delay(4000);

if (Firebase.RTDB.getString(&fbdo, "/smarttravel/access2"))

{

intValue = fbdo.stringData();

String mySubString = intValue.substring(2, 3);

Serial.println(intValue);

Serial.println(mySubString);

if (mySubString == "a")

{

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Access Approved"); // write something to the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(5000);

u8g2.clearBuffer(); // clear the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(100);

}

else if (mySubString == "b")

{

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Access Denied"); // write something to the internal memory

u8g2.drawStr(0,40,"Please check your"); // write something to the internal memory

u8g2.drawStr(0,50,"balance"); // write something to the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(5000);

u8g2.clearBuffer(); // clear the internal memory

u8g2.sendBuffer();

delay(100);

}

delay(100);

}

else {

Serial.println(fbdo.errorReason());

}

delay(100);

}

rfid.PICC\_HaltA();

rfid.PCD\_StopCrypto1();

}

}

**Register office:**

#include <SPI.h>

#include <MFRC522.h>

#include <Arduino.h>

#include <U8g2lib.h>

#include <ESP8266WiFi.h>

#include <Firebase\_ESP\_Client.h>

#include "addons/TokenHelper.h"

#include "addons/RTDBHelper.h"

#ifdef U8X8\_HAVE\_HW\_SPI

#include <SPI.h>

#endif

#ifdef U8X8\_HAVE\_HW\_I2C

#include <Wire.h>

#endif

constexpr uint8\_t RST\_PIN = D4; // Configurable, see typical pin layout above

constexpr uint8\_t SS\_PIN = D8; // Configurable, see typical pin layout above

MFRC522 rfid(SS\_PIN, RST\_PIN); // Instance of the class

MFRC522::MIFARE\_Key key;

const int buzzerPin = D3; // Replace with the actual pin connected to the buzzer

U8G2\_SSD1306\_128X64\_NONAME\_F\_SW\_I2C u8g2(U8G2\_R0, /\* clock=\*/ D1, /\* data=\*/ D2, /\* reset=\*/ U8X8\_PIN\_NONE); // All Boards without Reset of the Display

#define WIFI\_SSID "123456789"

#define WIFI\_PASSWORD "123456789"

#define API\_KEY "AIzaSyBuz9awICx9wnE44zeVrgGy\_JSiI51JfcE"

#define DATABASE\_URL "https://smart-billboards-using-iot-default-rtdb.firebaseio.com"

FirebaseData fbdo;

FirebaseAuth auth;

FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;

bool signupOK = false;

String intValue;

void setup() {

Serial.begin(115200);

SPI.begin(); // Init SPI bus

rfid.PCD\_Init(); // Init MFRC522

pinMode(buzzerPin, OUTPUT);

u8g2.begin();

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

Serial.print("Connecting to Wi-Fi");

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

Serial.print(".");

delay(300);

}

Serial.println();

Serial.print("Connected with IP: ");

Serial.println(WiFi.localIP());

Serial.println();

config.api\_key = API\_KEY;

config.database\_url = DATABASE\_URL;

if (Firebase.signUp(&config, &auth, "", "")){

Serial.println("ok");

signupOK = true;

}

else{

Serial.printf("%s\n", config.signer.signupError.message.c\_str());

}

config.token\_status\_callback = tokenStatusCallback; //see addons/TokenHelper.h

Firebase.begin(&config, &auth);

Firebase.reconnectWiFi(true);

}

void loop() {

if (!rfid.PICC\_IsNewCardPresent())

return;

if (rfid.PICC\_ReadCardSerial()) {

String tag;

for (byte i = 0; i < 4; i++) {

tag += rfid.uid.uidByte[i];

}

Serial.println("Detected Card UID: " + tag);

// Check if the detected UID is in the list of authorized UIDs

delay(100);

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Welcome"); // write something to the internal memory

u8g2.drawStr(0,30,"process in App"); // write something to the internal memory

u8g2.sendBuffer();

delay(100);

if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 1000 || sendDataPrevMillis == 0)){

sendDataPrevMillis = millis();

if (Firebase.RTDB.setString(&fbdo, "smarttravel/add", tag)){

Serial.println("PATH: " + fbdo.dataPath());

Serial.println("TYPE: " + fbdo.dataType());

}

else {

Serial.println("Failed REASON: " + fbdo.errorReason());

}

// transfer internal memory to the display

delay(4000);

if (Firebase.RTDB.getString(&fbdo, "/smarttravel/result"))

{

intValue = fbdo.stringData();

String mySubString = intValue.substring(2, 3);

Serial.println(intValue);

Serial.println(mySubString);

if (mySubString == "a")

{

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"User Added"); // write something to the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(5000);

u8g2.clearBuffer(); // clear the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(100);

}

else if (mySubString == "b")

{

u8g2.clearBuffer(); // clear the internal memory

u8g2.setFont(u8g2\_font\_ncenB08\_tr); // choose a suitable font

u8g2.drawStr(0,10,"Amount added"); // write something to the internal memory

u8g2.sendBuffer(); // transfer internal memory to the display

delay(5000);

u8g2.clearBuffer(); // clear the internal memory

u8g2.sendBuffer();

delay(100);

}

delay(100);

}

else {

Serial.println(fbdo.errorReason());

}

delay(100);

}

rfid.PICC\_HaltA();

rfid.PCD\_StopCrypto1();

}

}