GOOGLE SCRIPT CODE

```
var timeZone="CST";
var dateTimeFormat="dd/MM/yyyy HH:mm:ss";
var logSpreadSheetId="";

function sendEmail(message, id) {
    var subject = 'Something wrong with ' + id;
    MailApp.sendEmail(emailAddress, subject, message);

}

function doGet(e) {
   var access="-1";
      var text='Welcome';
      var name='Place your card';
  var json;
  var error="idk";
    Logger.log(JSON.stringify(e)); // view parameters
    var result = 'Ok'; // assume success
    if (e.parameter == 'undefined') {
        result = 'No Parameters';
    } else {

        var uid = '';
        var onlyPing=false;
        var id = 'Attendance';
        var error = '';
        for (var param in e.parameter) {

            var value = stripQuotes(e.parameter[param]);

            switch (param) {
                case 'uid':
                    uid = value;
                    break;
                case 'id':
                    id = value;
                    break;

                default:
                    result = "unsupported parameter";
            }
        }


      var sheet=SpreadsheetApp.getActive().getActiveSheet();

      var data = sheet.getDataRange().getValues();
    if (data.length == 0)
        return;
    for (var i = 0; i < data.length; i++) {

        if (data[i][0] ==uid)
```

```
        {
          name=data[i][1];
          access=data[i][2];
          text=data[i][3];
          break;
        }

    }


      addLog(uid,id,name,access);


       }
//     json = {
//    'access':access,
//    'name': name,
//    'text':text,
//    'error':error}

     result=(access+":"+name+":"+text);
  return ContentService.createTextOutput(result);
//  return ContentService.createTextOutput(JSON.stringify(json)
).setMimeType(ContentService.MimeType.JSON);
}


function addLog(uid,entrance, name,result) {


    var spr=SpreadsheetApp.openById(logSpreadSheetId);
    var sheet = spr.getSheets()[0];
    var data = sheet.getDataRange().getValues();

    var pos = sheet.getLastRow() + 1;

    var rowData = [];
  rowData[0] = Utilities.formatDate(new Date(), timeZone, dateTimeFormat);
  rowData[4]=entrance;
  rowData[1] = uid;
  rowData[2] = name;
  rowData[3] = result;
    var newRange = sheet.getRange(pos, 1, 1, rowData.length);
    newRange.setValues([rowData]);


}



/**
 * Remove leading and trailing single or double quotes
 */
function stripQuotes(value) {
    return value.replace(/^["']|['"]$/g, "");
}
```

ARDUINO CODE

```
#include
<SPI.h>
          #include <MFRC522.h>
          #include <ESP8266WiFi.h>
          #include <WiFiClientSecure.h>
          #include <LiquidCrystal_I2C.h>
          LiquidCrystal_I2C lcd(0x27, 16, 2);
          #define SS_PIN D4
          #define RST_PIN D0    // Configurable, see typical pin layout
          above
          MFRC522 mfrc522(SS_PIN, RST_PIN);  // Create MFRC522 instance
          #define BUZZ_PIN D8
          #define GATE_PIN D3
          const char* host = "script.google.com";
          const int httpsPort = 443;
          const char* fingerprint  = "46 B2 C3 44 9C 59 09 8B 01 B6 F8 BD
          4C FB 00 74 91 2F EF F6"; // for https
          //***********Things to change******************
          const char* ssid = "";
          const char* password = "";
          String GOOGLE_SCRIPT_ID = ""; // Replace by your GAS service id
          const String unitName = "Attendance"; // any name without
          spaces and special characters
          //***********Things to change******************
          uint64_t openGateMillis = 0;
          WiFiClientSecure client;
          void LcdClearAndPrint(String text)
          {
            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print(text);
          }
```

```
void Siren()
{
  for (int hz = 440; hz < 1000; hz++) {
    tone(BUZZ_PIN, hz, 50);
    delay(5);
  }
  for (int hz = 1000; hz > 440; hz--) {
    tone(BUZZ_PIN, hz, 50);
    delay(5);
  }
  digitalWrite(BUZZ_PIN, LOW);
}
void Beep()
{
  for (int i = 0; i < 1000; i++)
  {
    analogWrite(BUZZ_PIN, i);
    delayMicroseconds(50);
  }
  digitalWrite(BUZZ_PIN, LOW);
}
void Beep2()
{
  tone(BUZZ_PIN, 1000, 30);
  delay(300);
  digitalWrite(BUZZ_PIN, LOW);
}
void setup() {
  pinMode(GATE_PIN, OUTPUT);
  pinMode(BUZZ_PIN, OUTPUT);
  digitalWrite(GATE_PIN, LOW);
  digitalWrite(BUZZ_PIN, LOW);

  Serial.begin(921600);
```

```
  lcd.begin(); // Init with pin default ESP8266 or ARDUINO
// lcd.begin(0, 2); //ESP8266-01 I2C with pin 0-SDA 2-SCL
// Turn on the blacklight and print a message.
  lcd.backlight();
  LcdClearAndPrint("Loading");
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  Serial.println("Started");
  Serial.print("Connecting");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  // Initialize serial communications with the PC
  while (!Serial);    // Do nothing if no serial port is opened
(added for Arduinos based on ATMEGA32U4)
  SPI.begin();      // Init SPI bus
  mfrc522.PCD_Init();   // Init MFRC522
  delay(4);       // Optional delay. Some board do need more
time after init to be ready, see Readme
  mfrc522.PCD_DumpVersionToSerial();  // Show details of PCD -
MFRC522 Card Reader details
  Serial.println(F("Scan PICC to see UID, SAK, type, and data
blocks..."));
  LcdClearAndPrint("Ready");
}
byte readCard[4];
void HandleDataFromGoogle(String data)
{
  int ind = data.indexOf(":");
  String access = data.substring(0, ind);
  int nextInd = data.indexOf(":", ind + 1);
  String name = data.substring(ind + 1, nextInd);
```

```
  String text = data.substring(nextInd + 1, data.length());
  Serial.println(name);
  LcdClearAndPrint(name);
  lcd.setCursor(0, 1);
  lcd.print(text);
  if (access=="-1")
  {
    lcd.print(" " + String("denied"));
    Siren();
    LcdClearAndPrint("Ready");
  }
  else if(access=="any")
  {

    lcd.print(" " + String("go in"));
    OpenGate();
  }
  else if (access=="fridge")
  {

    lcd.print(" " + String("take it"));
    OpenGate();
  }
}
void OpenGate()
{
  openGateMillis = millis()+5000;
  digitalWrite(GATE_PIN, HIGH);
  Beep();
  delay(100);
  Beep();
}
void CloseGate()
{
```

```
  openGateMillis = 0;
  digitalWrite(GATE_PIN, LOW);
  Beep2();
  LcdClearAndPrint("Ready");
}
void loop() {
  if (openGateMillis > 0 && openGateMillis < millis())
  {
    CloseGate();
  }
  if (!mfrc522.PICC_IsNewCardPresent()) {
    return;
  }
  // Select one of the cards
  // Reset the loop if no new card present on the
sensor/reader. This saves the entire process when idle.
  if (!mfrc522.PICC_ReadCardSerial()) {
    return;
  }
  Serial.println(F("Scanned PICC's UID:"));
  String uid = "";
  for (uint8_t i = 0; i < 4; i++) {  //
    readCard[i] = mfrc522.uid.uidByte[i];
    Serial.print(readCard[i], HEX);
    uid += String(readCard[i],HEX);
  }
  Serial.println("");
  Beep();
  LcdClearAndPrint("Please wait...");
  String data = sendData("id=" + unitName + "&uid=" +
uid,NULL);
  HandleDataFromGoogle(data);
  mfrc522.PICC_HaltA();
}
```

```
String sendData(String params, char* domain) {
  //google scripts requires two get requests
  bool needRedir = false;
  if (domain == NULL)
  {
    domain=(char*)host;
    needRedir = true;
    params = "/macros/s/" + GOOGLE_SCRIPT_ID + "/exec?" +
params;
  }

  Serial.println(*domain);
  String result = "";
  client.setInsecure();
  Serial.print("connecting to ");
  Serial.println(host);
  if (!client.connect(host, httpsPort)) {
    Serial.println("connection failed");
    return "";
  }
  if (client.verify(fingerprint, domain)) {
  }
  Serial.print("requesting URL: ");
  Serial.println(params);
  client.print(String("GET ") + params + " HTTP/1.1\r\n" +
    "Host: " + domain + "\r\n" +
    "Connection: close\r\n\r\n");
  Serial.println("request sent");
  while (client.connected()) {
    String line = client.readStringUntil('\n');
    //Serial.println(line);
    if (needRedir) {
    int ind = line.indexOf("/macros/echo?user");
    if (ind > 0)
```

```
    {
      Serial.println(line);
      line = line.substring(ind);
      ind = line.lastIndexOf("\r");
      line = line.substring(0, ind);
      Serial.println(line);
      result = line;
    }
    }
    if (line == "\r") {
      Serial.println("headers received");
      break;
    }
  }
  while (client.available()) {
    String line = client.readStringUntil('\n');
    if(!needRedir)
    if (line.length() > 5)
      result = line;
    //Serial.println(line);

    }
  if (needRedir)
    return sendData(result, "script.googleusercontent.com");
  else return result;

}
```