



ESP32-Based Fingerprint Attendance System with Real-Time Google Sheets Integration

Prepared by: Aditya Raj (COMETFWC015)

Contents

| 1 | Project Overview | 1 | | | |
|----|---|------------------|--|--|--|
| 2 | Hardware Components | 2 | | | |
| 3 | Software Stack | 2 | | | |
| 4 | Functional Operation 4.1 Enrollment | 2 2 3 3 | | | |
| 5 | Google Apps Script: Web API for Google Sheets | | | | |
| 6 | ESP32 Firmware Code (Main logic) | 4 | | | |
| 7 | Key Implementation Details | | | | |
| 8 | Testing and Usage Workflow | | | | |
| 9 | Improvements and Notes | 5 | | | |
| 10 | Appendix | 5 | | | |

1 Project Overview

This project implements a WiFi-enabled fingerprint-based attendance solution using an ESP32, a R307 (or compatible) fingerprint sensor, and real-time logging to a Google Sheet using a Google Apps Script Web App.

- Enrollment: New users can be added with fingerprint, name, and ID via Serial.
- Attendance: Each scan logs attendance to a Google Sheet with timestamp.
- Security: Uses WPA2-Enterprise (802.1X) for campus/corporate WiFi.
- Persistence: User database and attendance serial stored in ESP32 EEPROM.
- Feedback: Audible buzzer signals successful scan or sheet sync status.

2 Hardware Components

- ESP32 Dev Board: NodeMCU-ESP32 (or similar).
- Fingerprint Sensor: R307 or Adafruit-compatible module (UART).
- Buzzer: Basic 3.3V active or passive buzzer (GPIO4).
- Cables: Appropriate jumper wires.
- Power: Micro-USB or regulated 3.3V/5V supply.

3 Software Stack

- Arduino IDE (ESP32 core v2+).
- Adafruit Fingerprint Sensor Library.
- WiFi.h, HTTPClient.h, EEPROM.h.
- Google Apps Script for receiving and logging data in Google Sheets.

4 Functional Operation

4.1 Enrollment

- 1. User starts enrollment via Serial command 'E'.
- 2. System prompts for a numeric ID, name, and Comet ID.
- 3. Fingerprint is scanned twice; model is created and saved to the sensor.
- 4. User info is mapped to fingerprint ID and stored in ESP32 EEPROM.

4.2 Attendance Logging

- 1. User places a registered finger on the sensor.
- 2. ESP32 searches for and matches the template.
- 3. System fetches corresponding name and ID from EEPROM.
- 4. Details (name, ID, timestamp) are POSTed as JSON to Google Apps Script endpoint.
- 5. Buzzer feedback indicates result.

4.3 Administrative Serial Commands

- P | Print all currently stored users
- E | Start new user enrollment
- C | Clear all stored user and attendance data

5 Google Apps Script: Web API for Google Sheets

Script Code

Listing 1: Google Apps Script Web Endpoint

```
// Paste this code into your Google Apps Script editor
   function doPost(e) {
2
    try {
3
       var sheet = SpreadsheetApp.getActiveSpreadsheet().getSheets()[0];
      var data = JSON.parse(e.postData.contents);
6
      var name = data.name;
      var cometId = data.id;
      var time = data.time;
      // Generate serial number based on current last row.
11
      // Assumes first row is header, serial starts at 1 for first data row.
12
      var lastRow = sheet.getLastRow();
13
      var serial = (lastRow < 1) ? 1 : lastRow;</pre>
14
15
      // Append data, serial in first column, then name, cometId, time
16
      sheet.appendRow([serial, name, cometId, time]);
17
18
      return ContentService.createTextOutput(JSON.stringify({result:"success"}))
19
             .setMimeType(ContentService.MimeType.JSON);
20
    }
21
     catch (error) {
22
      return ContentService.createTextOutput(
23
        JSON.stringify({result:"error", message:error.toString()}))
24
```

How to Deploy:

- 1. Copy and paste the above code into the script editor on your Google Sheet (Extensions \rightarrow Apps Script).
- 2. Deploy as a Web App with access to "Anyone, even anonymous" (or as per your security needs).
- 3. Copy the Deployment URL and paste it into your ESP32 code as WEB_APP_URL.

6 ESP32 Firmware Code (Main logic)

Main features: Enrollment, fingerprint search, EEPROM storage, HTTP POST to Sheets, buzzer feedback, WPA2-Enterprise WiFi.

Listing 2: ESP32 Project Source Code (Main)

```
#include <Adafruit_Fingerprint.h>
   #include <WiFi.h>
   #include <HTTPClient.h>
   #include <HardwareSerial.h>
   #include <EEPROM.h>
   #include <time.h>
   #include "esp_wpa2.h"
10
12
13
14
15
16
^{17}
18
19
```

(For the full working code see: here.)

7 Key Implementation Details

- User info stored and retrieved safely in EEPROM with null-terminated strings.
- Mapping fingerprint ID to user index in arrays for flexible linkage.

- Attendance serial is auto-incremented and persistent but not sent to Google Sheets (handled server-side).
- Buzzer feedback for successful scans and specific server response codes like HTTP 302.
- NTP time synchronization provides accurate time stamps for attendance entries.

8 Testing and Usage Workflow

- 1. Power up the ESP32 device; it connects to WiFi and fingerprint sensor initializes.
- 2. Use serial commands: P print all users, E enroll new user, C clear all stored data.
- 3. For attendance, place a registered finger on the sensor.
- 4. The ESP32 detects the fingerprint, matches the user, and posts data to Google Sheets.
- 5. Buzzer indicates success or HTTP 302 responses with specific buzz patterns.
- 6. Monitor Google Sheet to confirm attendance logs are received.

9 Improvements and Notes

- Use ARRAYFORMULA or Apps Script to auto-generate serial numbers in Google Sheets.
- Consider EEPROM alternatives (SPIFFS or LittleFS) for scalable user storage.
- Upgrade WiFi code to use esp_eap_client.h for future WPA2 enhancements.
- Secure Google Apps Script endpoints and ESP32 HTTPS communication for production.

10 Appendix

Sample Google Sheet Layout

| 1 | Serial Name | Comet ID | Time |
|---|----------------|-------------|---------------------|
| 2 | | | |
| 3 | 1 Aditya Raj | COMETFWC015 | 2025-07-30 16:05:23 |