

ONLINE TOUCHLESS ATTENDANCE SYSTEM (OTAS)

By

GABRIEL ELDOSE

INTRODUCTION

The COVID pandemic has affected the livelihood of people in different forms by making it harder for them to live. One of the sections which were affected adversely by this pandemic was the daily waged workers. The marking of attendance was the main problem for them because the formation of the crowd at a headquarters to mark their attendance caused the spread of infection. The usage of the Online RFID attendance system helps the people working for daily wages to mark their attendance accurately. It enables them to mark their attendance at their respective sites of work without directly reporting to headquarters. This helps in decreasing the crowding for marking attendance and thereby reduces the spread of COVID.

The Attendance system has enormous benefits as it can be developed at an affordable rate. With it maintaining daily attendance on Google Spreadsheets without any Human control. The monthly attendance record of staff can be assembled in a single place and format. RFID is being used actively in retail, healthcare, and other sectors to monitor workers. Since the workers in these sectors are large in number, hard to handle and their work can be performed by others in case of absenteeism; there the attendance mechanism is of trivial significance. The System provides more accurate identification. This system can be used to take attendance for workers in working places. Its ability to uniquely identify each person based on their RFID tag type of ID card and upload the data directly to the server will make the process of taking the attendance easier, faster, and secure as compared to the conventional method. It Identifies candidates in seconds. The System is less tedious, cost-efficient, easy to carry and use, hard to adulterate, and modest. This Online RFID attendance system reduces the chance of COVID Infection spreading among workers in different sectors, also the time can be saved.

MATERIALS USED IN THIS PROJECT

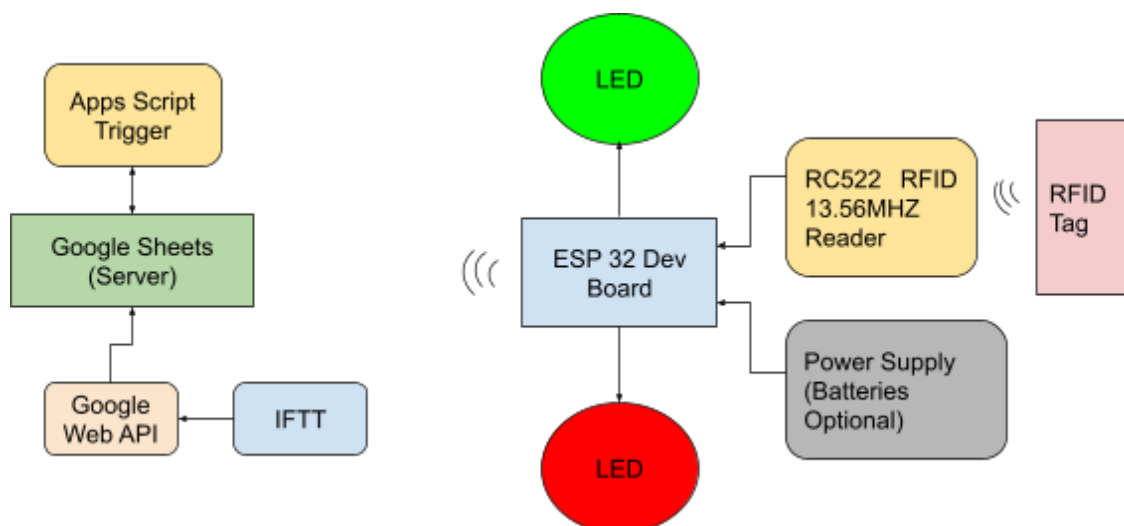
HARDWARE COMPONENTS

- ESP32 Dev board
- RC522 RFID 13.56MHZ Reader Writer Module
- RFID Cards
- Light Emitting Diode (LEDs)
- USB cable
- Connecting Wires
- Breadboard
- Li-ion Battery (Optional)

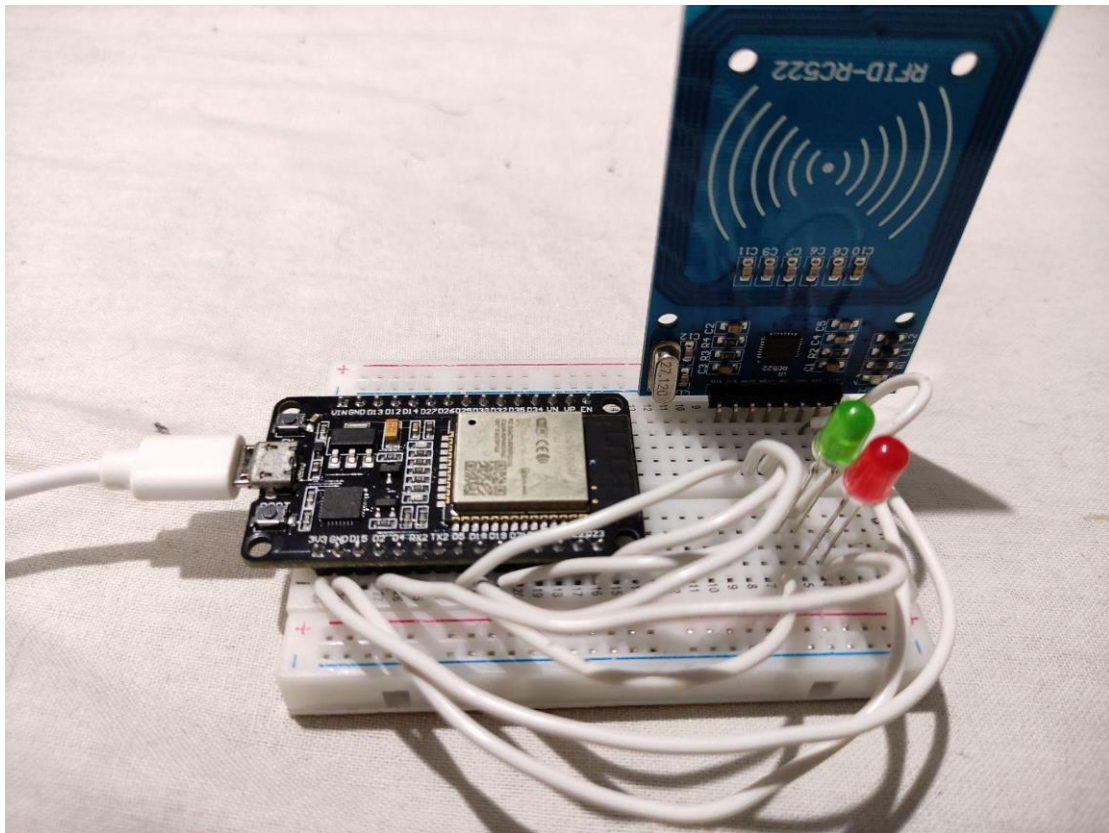
SOFTWARE APPS & ONLINE SERVICES

- Google Spreadsheet
- Arduino IDE
- Google Apps Script
- IFTTT Platform

BLOCK DIAGRAM



WORKING PRINCIPLE

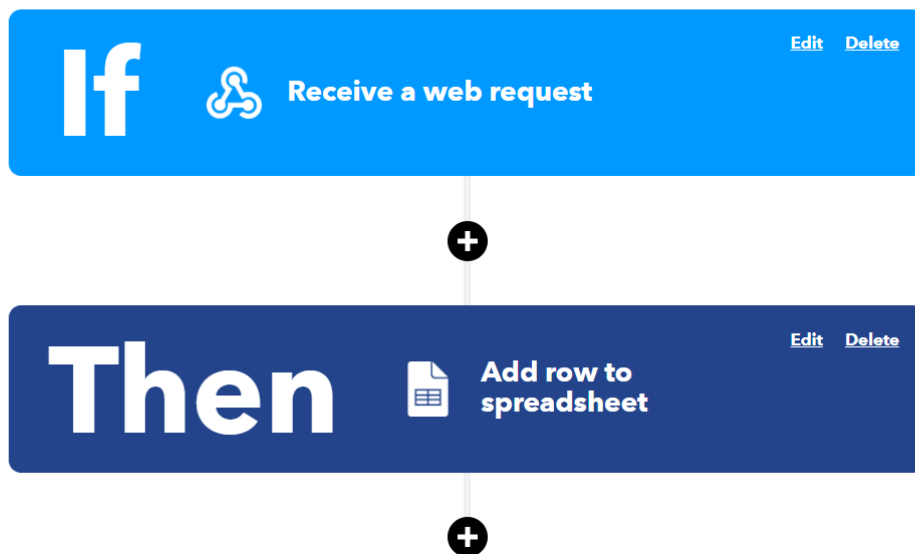


The Attendance system mainly consists of 4 hardware components, ESP32 is the brain or microcontroller of the system. RC522 is a MFRC522 based RFID Reader Module. RC522 is a highly integrated RFID card reader which works on Non-Contact 13.56MHz Communication, and it is designed by NXP as a low power consumption, low cost and compact size read and write chip. The module uses 3.3V power supply. The Reader Module is connected to the ESP32 board using SPI Communication. The system also contains 2 LEDs as indicators. All the wiring is done on the BreadBoard. Programming part of the Microcontroller is done in Arduino IDE Software using external open-source libraries. The connection or data transfer between the ESP32 and Google Spreadsheet is done using the IFTTT platform. It connects to the ESP32 board using Webhook API services with an authentication code and then webhook establishes connection to the specified Spreadsheet API. Hence IFTTT helps in automating the services by creating an Applet. The system is powered by a 5V USB power supply. It can also be powered using a Li-ion battery for external usage.

```
Final_RFID
1 //libraries
2 #include <SPI.h> //https://www.arduino.cc/en/reference/SPI
3 #include <MFRC522.h> //https://github.com/miguelbalboa/rfid
4 //Constants
5 #define SS_PIN 5
6 #define RST_PIN 0
7 #include <WiFi.h>
8 #include <Wire.h>
9 // Replace with your SSID and Password
10 const char* ssid = "GBK3";
11 const char* password = "password";
12
13 // Replace with your unique IFTTT URL resource
14 const char* resource = "/trigger/rfid_readings/with/key/mLEaWgqWhwc3Nz-lu8e9R3MPSvqP2zJ5ydcFEsa";
15
16 // How your resource variable should look like, but with your own API KEY (that API KEY below is just an example):
17 //const char* resource = "/trigger/rfid_readings/with/key/nA5jOphL3d-204N3k64-1A7gT1N0rxhJ5eqy3";
18
19 // Maker Webhooks IFTTT
20 const char* server = "maker.ifttt.com";
21
22 //Parameters
23 const int ipaddress[4] = {103, 97, 67, 25};
24
25 //Variables
26 byte nuidPICC[4] = {0, 0, 0, 0};
27 MFRC522::MIFARE_Key key;
28 MFRC522 rfid = MFRC522(SS_PIN, RST_PIN);
29
30 void setup() {
31   pinMode(4, OUTPUT);
32   pinMode(2, OUTPUT);
33   //Init Serial USB
34   Serial.begin(115200);
35   Serial.println(F("Initialize System"));
36   initWifi();
37   //init rfid D8,D5,D6,D7
38 }
```

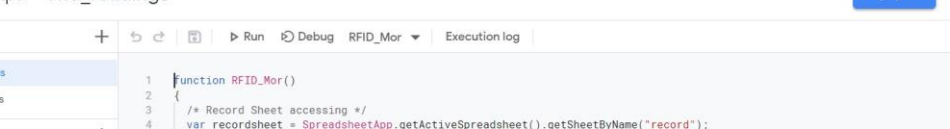
ESP32 Code in Arduino IDE

Edit Applet



RFID Applet in IFTTT Platform

Initially, when the ESP32 is powered up it tries to connect to a particular WiFi network of 2.4GHz band with a predefined Username and Password. Hence the User has to set this beforehand. The Peculiarity of this system is that, to establish a connection between the system and online server, a mobile hotspot is sufficient. When the system gets connected to the specified WiFi network, the green LED starts blinking 5 times. Otherwise, the red LED blinks 5 times. After that, when the



The screenshot shows the Google Apps Script editor interface. The top bar includes the 'Apps Script' logo, the project name 'rfid_readings', and a 'Deploy' button. Below the top bar is a sidebar with icons for Files, Libraries, and Services. The main area displays the code for the 'RFID_Mor.js' file. The code is a JavaScript function named 'RFID_Mor()' that interacts with a Google Sheet. It uses the 'SpreadsheetApp' and 'Utilities' classes. The function iterates through rows of data in a sheet named 'recordsheet' and updates values in a sheet named 'datasheet' based on specific conditions. The code is as follows:

```
1 function RFID_Mor()
2 {
3   /* Record Sheet accessing */
4   var recordsheet = SpreadsheetApp.getActiveSpreadsheet().getSheetByName("record");
5   /* 'Data' Sheet accessing */
6   var datasheet = SpreadsheetApp.getActiveSpreadsheet().getSheetByName("data");
7   var rule = SpreadsheetApp.newDataValidation().requireDate();
8   var sum;
9   var date= Utilities.formatDate(new Date(), SpreadsheetApp.getActive().getSpreadsheetTimeZone(), "MM")
10
11 for(var i=2;datasheet.getRange(i,1).getValue()!="";i++)
12 { var count=0;
13   for(var j=2;recordsheet.getRange(j,3).getValue()!="";j++)
14   {
15     if(recordsheet.getRange(j,3).getValue() == datasheet.getRange(i,1).getValue())
16     {
17       count++;
18     }
19     if(count==1)
20     {
21       if(date==1)
22       {
23         sum=datasheet.getRange(i,3).getValue();
24         sum=sum+1;
25         datasheet.getRange(i,3).setValue(sum);
26         break;
27       }
28       else if(date==2)
29       {
30         sum=datasheet.getRange(i,4).getValue();
31         sum=sum+1;
32         datasheet.getRange(i,4).setValue(sum);
33         break;
34       }
35     }
36   }
37 }
```

Apps Script

rfid_readings

Triggers

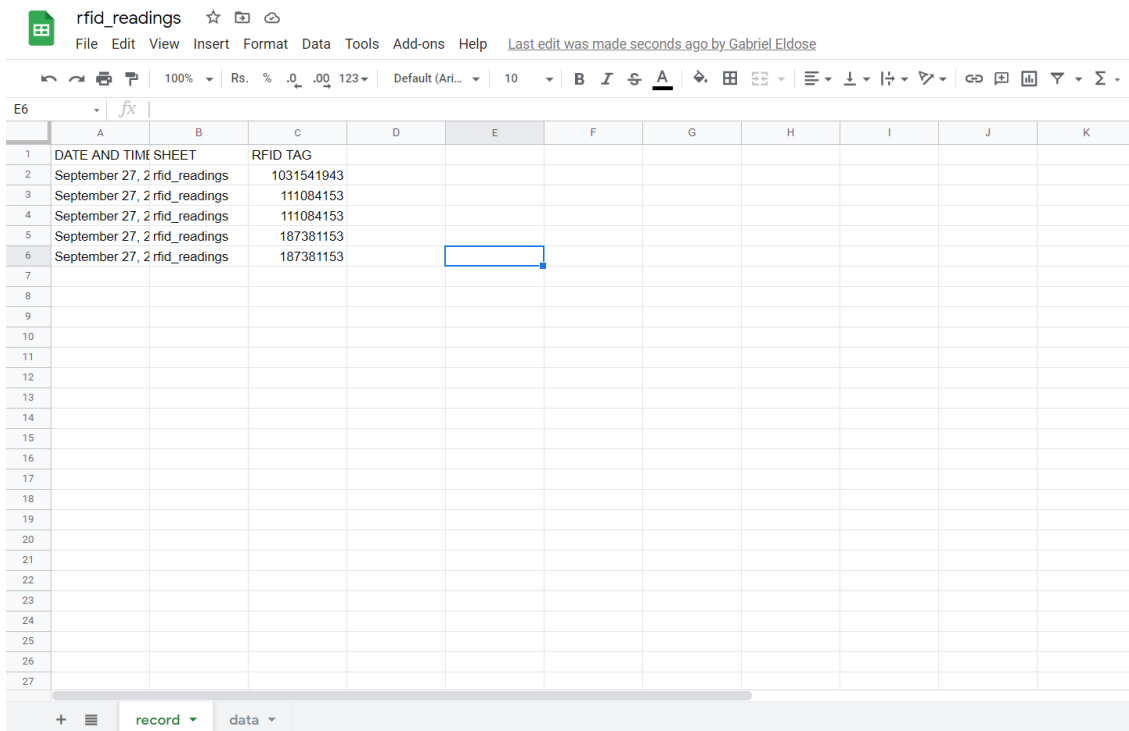
+ Add a filter

Owned by	Last run	Deployment	Event	Function
Me	Sep 26, 2021, 5:10:12 PM	Head	Time-based	RFID_Eve
Me	Sep 27, 2021, 9:07:33 AM	Head	Time-based	RFID_Mor

When the attendance data is transmitted to the Spreadsheet server it gets stored there, then using Google Apps Script the data is manipulated. In our case, we have two sheets inside a Spreadsheet. One sheet is for storing every day entry and exit data of employees and another sheet for storing information about the count of the number of days worked by employees individually. The Apps Script is set to trigger at a certain time as per the user requirement. In our case we have set it for morning 9 AM and in the evening 5 PM. When the entry of employees is recorded in the sheet, the script triggers and data is manipulated as programmed after that in the evening the exit of employees is also recorded in the sheet and then script triggers. The script is triggered twice in order to verify that the persons entered in the morning are also present while exiting the workplace. If any of the employees forgets to record their entry and exit, then that will not be considered as work day in the server.

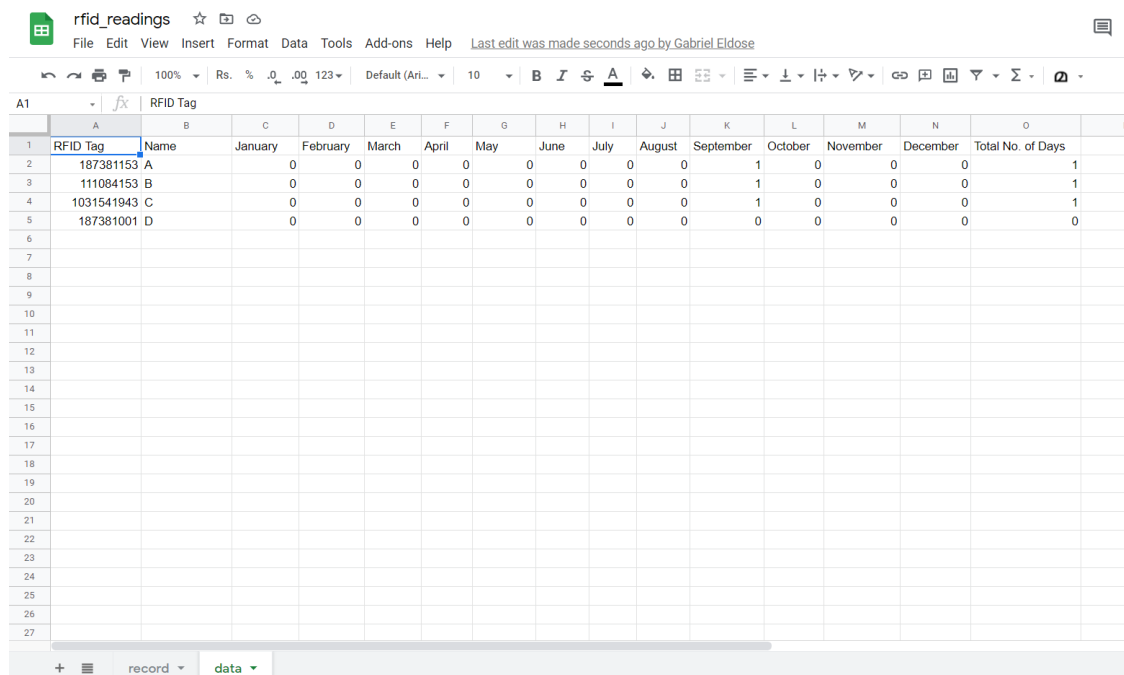
Hence, if needed to change, then the admin has the permission to edit the server data. Also If employees try to swipe their RFID tag or ID more than once at entry or exit, then also the data will be considered as a single entry or exit for that day. If a new employee has joined and has to be added to the server, then the admin can register the name and RFID tag number to the server. Then the script will automatically start counting the number of days worked by the employee.

The link to the code and library is available in [github](#).



	A	B	C	D	E	F	G	H	I	J	K
1	DATE AND TIME SHEET		RFID TAG								
2	September 27, 2 rfid_readings		1031541943								
3	September 27, 2 rfid_readings		111084153								
4	September 27, 2 rfid_readings		111084153								
5	September 27, 2 rfid_readings		187381153								
6	September 27, 2 rfid_readings		187381153								
7											
8											
9											
10											
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											
25											
26											
27											

Record sheet in Spreadsheet Server



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	RFID Tag	Name	January	February	March	April	May	June	July	August	September	October	November	December	Total No. of Days	
2	187381153 A		0	0	0	0	0	0	0	0	1	0	0	0	1	
3	111084153 B		0	0	0	0	0	0	0	0	1	0	0	0	1	
4	1031541943 C		0	0	0	0	0	0	0	0	1	0	0	0	1	
5	187381001 D		0	0	0	0	0	0	0	0	0	0	0	0	0	
6																
7																
8																
9																
10																
11																
12																
13																
14																
15																
16																
17																
18																
19																
20																
21																
22																
23																
24																
25																
26																
27																

Data sheet in Spreadsheet Server

BUDGET ESTIMATION

HARDWARE COMPONENTS	COST
ESP32 Board	₹500
RC522 RFID READER	₹150
LED	₹10
PCB & CASING	₹250
Li-ion Battery (Optional)	₹90
TOTAL	₹1000

Hardware Costs approximately less than or equal to ₹1000 including a 1500mAh Li-ion battery (18650) which gives about nearly 4hrs of continuous usage since the current consumption of Hardware varies between 300-500mA.

On the Software side, IFTTT Standard package provides free usage of applets which includes Webhook and Google Sheet API. Also 3 applets can be created per user, but for this system a single applet is only required. Google spreadsheet is free to use and Spreadsheet API provides 100 requests per 100 seconds per user for free which is sufficient for this system. Also, Google Apps Script is free and open to use.

CONCLUSION

RFID technology is evolving and the applications of RFID technology are vast. The proposed approach provides a method to mark the attendance of workers without physical contact and recording. Everything is automated. So the system will be helpful at tough times like this pandemic. This project is to simplify the attendance recorder system by using Radio Frequency Identification (RFID) technology and easily calculate the wages of the employees. The end users, i.e. the workers just need to swipe their card over the RFID reader and automatically their attendance or presence will be marked. Since the system is portable and easy to carry, it can be transported to anywhere. This system can shift the paradigm of monitoring everyone and helps to conserve time and effort.