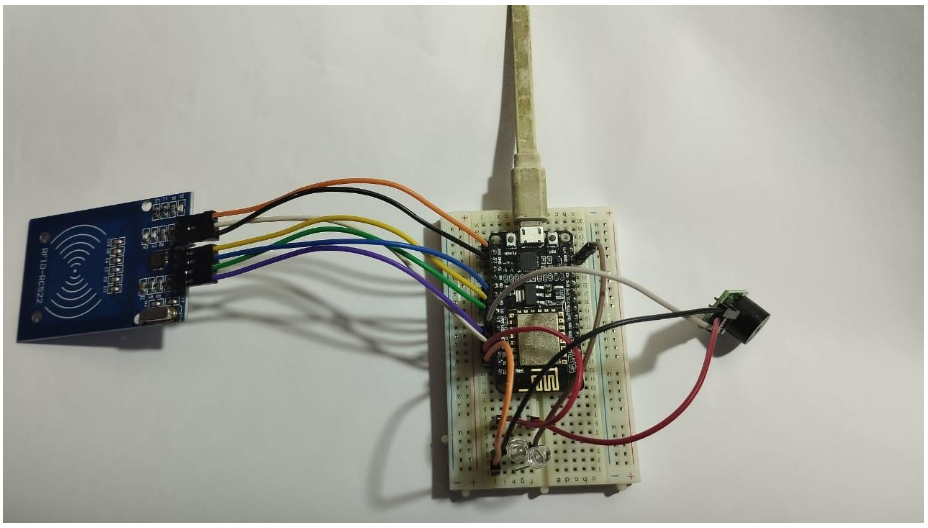
भारतीय प्रौद्योगिकी संस्थान गुवाहाटी
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

CS 578: Internet of Things

Hands On Project – I

TITLE: Smart Attendance Monitoring System



Team 14:

1.	Saarthak Sarkar	- 200103094
2.	Kashish Ranjan	- 200104043
3.	Rikita Jatav	- 200122048

Objectives:

The primary objective of this project is to design, develop, and implement a cost-effective and efficient smart attendance logging system using **NodeMCU** and **RFID (Radio-Frequency Identification)** sensors. The system will automate the process of recording attendance in various environments such as schools, workplaces, and events, enhancing accuracy and convenience.

- Key Features and Functionality: The project aims to achieve the following key features and functionalities:
    - **RFID-based Identification**: Utilize RFID sensors to uniquely identify individuals through RFID cards or tags, allowing for quick and reliable attendance tracking.
    - **Wireless Connectivity:** Implement Wi-Fi capabilities of NodeMCU to connect to a central server or cloud platform for real-time data storage and access.
    - **User-friendly Interface:** Create a user-friendly web-based or mobile application interface to view and manage attendance records, providing administrators with real-time insights and attendance reports
    - **Notification and Alerts:** Enable automatic notifications or alerts for both administrators and attendees, ensuring timely updates and reminders.

Implemented Attributes:

1.NodeMCU : Used as main controller board with Wifi module.

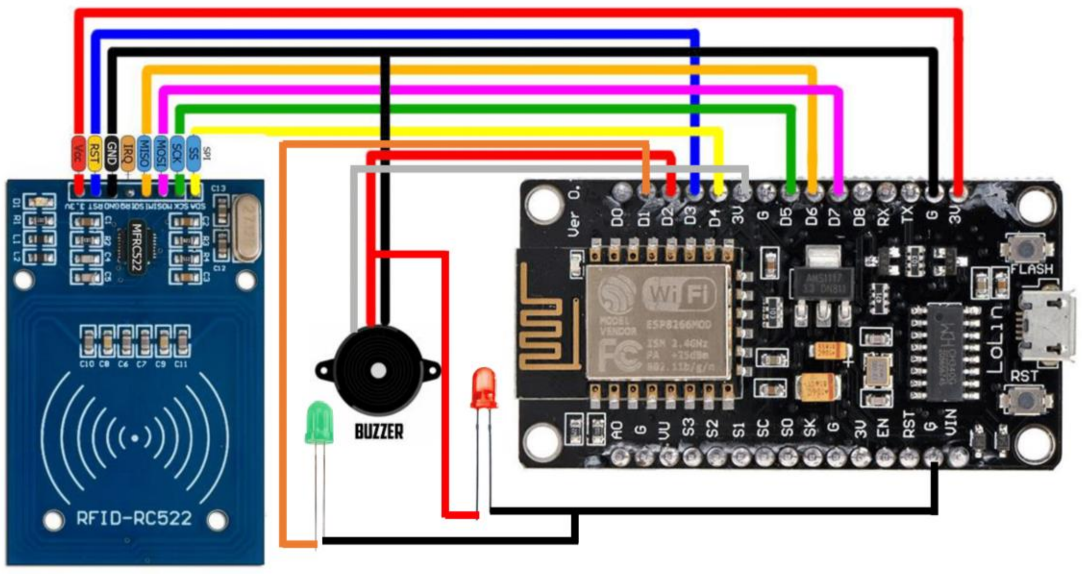
2. RFID Sensor and Authentication: To read the RFID id card & authenticate.

3. Buzzer and LEDs: Used as actuator to produce output on receiving input.

4. WiFi Connectivity : Used for connectivity among sensors, actuator & cloud.

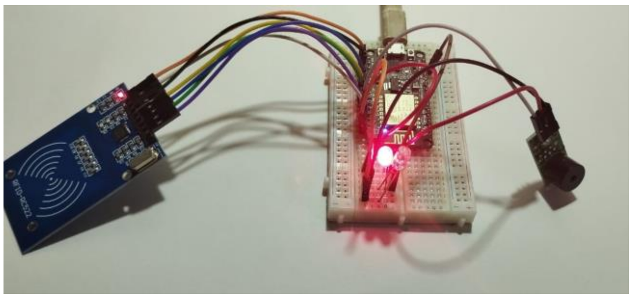
5. Cloud based Application (Google Sheets): Used to store data

Configuration Diagram:

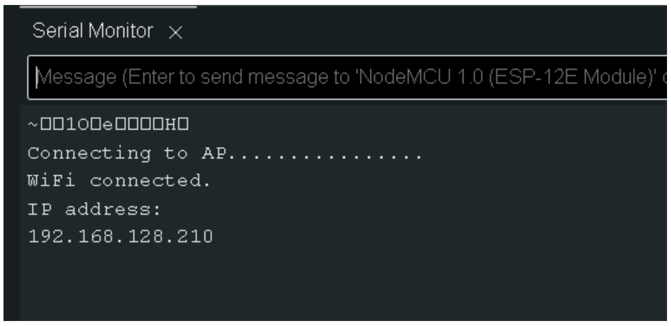


Procedure:

1. **1.** Firstly, at normal time, the light remains red which signifies that the system is ready to scan a card.

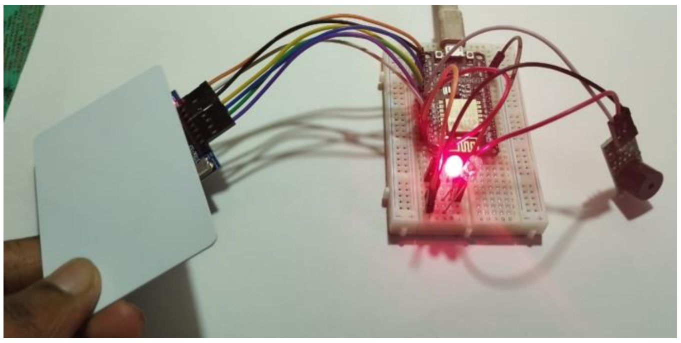


2. **2.** **NodeMCU** connects to WiFi signal of the router (here smartphone).

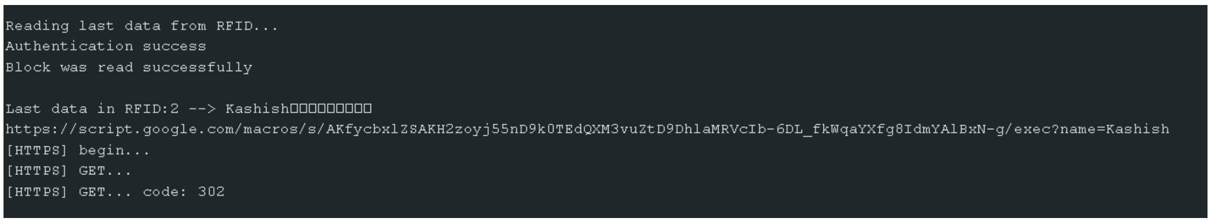

Serial Monitor ×
Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)'

```
~□□1o□e□□□□H□
Connecting to AP...............
WiFi connected.
IP address:
192.168.128.210
```

3. Then we need to scan the RFID attendance card, there will be a short double beep with green light, signifies the card has been scanned.



4. The card gets scanned and sends it to cloud based application (here we used Google Sheets).



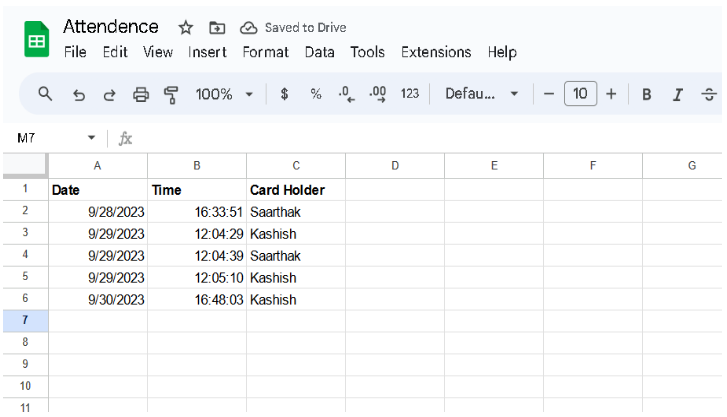
```
Reading last data from RFID...
Authentication success
Block was read successfully

Last data in RFID:2 --> Kashish□□□□□□□□□
https://script.google.com/macros/s/AKfycbxlZSAKH2zoyj55nD9k0TEdQXM3vuZtD9DhlaMRVcIb-6DL_fkWqaYXfg8IdmYAlBxN-g/exec?name=Kashish
[HTTPS] begin...
[HTTPS] GET...
[HTTPS] GET... code: 302
```

Sample Output:

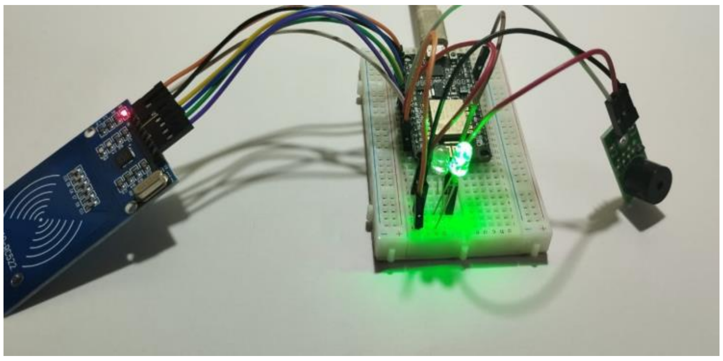
After successfully sending the data to the cloud, the data gets stored in the google sheets created for attendance tracking. The sheet gives us name of the person scanning the card, date and time at which he enters the class.

When its done, the cloud sends instruction to NodeMCU which further instructs the buzzer (actuator) for a long beep and green light, which tells attendance has been stored and it is ready for scanning the next card.

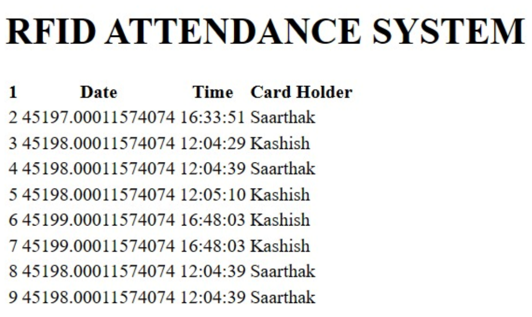


	A	B	C
1	Date	Time	Card Holder
2	9/28/2023	16:33:51	Saarthak
3	9/29/2023	12:04:29	Kashish
4	9/29/2023	12:04:39	Saarthak
5	9/29/2023	12:05:10	Kashish
6	9/30/2023	16:48:03	Kashish

The new entry is entered in the excel sheet with date, time and name. We can visualize the data for multiple students and easily compare the attendance using charts.

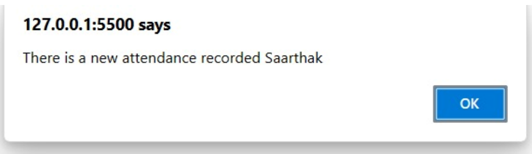


A prolonged beep is buzzed along with green light is given as the output after receiving instruction from the cloud that the attendance has been saved and next person can scan the card.



RFID ATTENDANCE SYSTEM

1	Date	Time	Card Holder
2	45197.00011574074	16:33:51	Saarthak
3	45198.00011574074	12:04:29	Kashish
4	45198.00011574074	12:04:39	Saarthak
5	45198.00011574074	12:05:10	Kashish
6	45199.00011574074	16:48:03	Kashish
7	45199.00011574074	16:48:03	Kashish
8	45198.00011574074	12:04:39	Saarthak
9	45198.00011574074	12:04:39	Saarthak

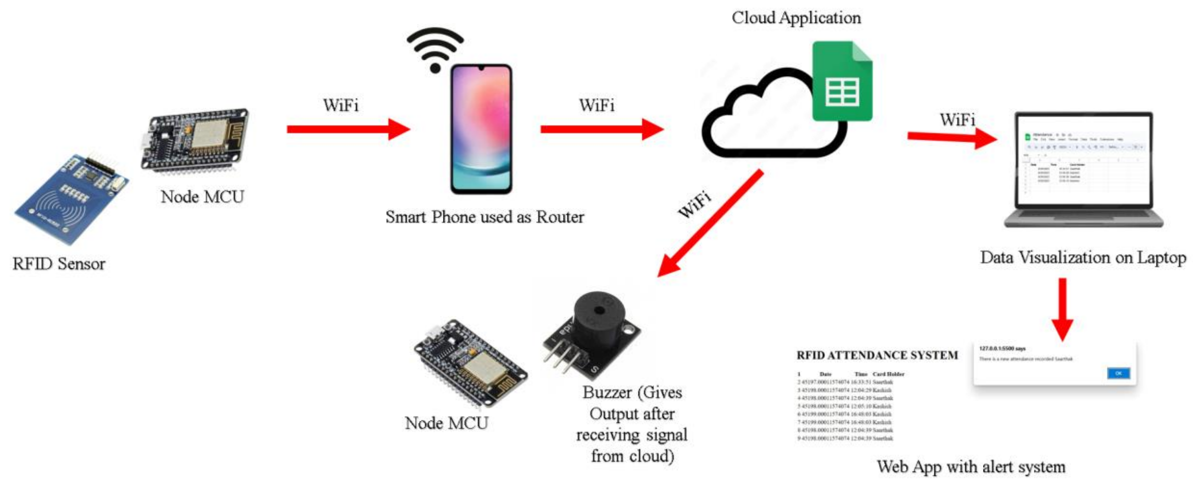


127.0.0.1:5500 says

There is a new attendance recorded Saarthak

A web app is created where the administrator can get real time data and receives a notification/ alert when a new attendance is taken.

Data Flow Diagram:


Cloud Application
WiFi
WiFi
WiFi
WiFi
Node MCU
RFID Sensor
Smart Phone used as Router
Data Visualization on Laptop
Node MCU
Buzzer (Gives Output after receiving signal from cloud)
Web App with alert system
OK

RFID ATTENDANCE SYSTEM

1	Date	Time	Card Holder
2	45197.00011574074	16:33:51	Saarthak
3	45198.00011574074	12:04:29	Kashish
4	45198.00011574074	12:04:39	Saarthak
5	45198.00011574074	12:05:10	Kashish
6	45199.00011574074	16:48:03	Kashish
7	45199.00011574074	16:48:03	Kashish
8	45198.00011574074	12:04:39	Saarthak
9	45198.00011574074	12:04:39	Saarthak

127.0.0.1:5500 says
There is a new attendance recorded Saarthak

Written Codes:

The main code which is used for scanning, uploading and receiving data.

```
#include <SPI.h>

#include <MFRC522.h>

#include <Arduino.h>

#include <ESP8266WiFi.h>

#include <ESP8266HTTPClient.h>

#include <WiFiClient.h>

#include <WiFiClientSecureBearSSL.h>

//-----------------------------------------

#define RST_PIN  D3

#define SS_PIN   D4

#define BUZZER   D2

#define LED      D1

//-----------------------------------------

MFRC522 mfrc522(SS_PIN, RST_PIN);

MFRC522::MIFARE_Key key;

MFRC522::StatusCode status;
```

```
//----------------------------------------
/* Be aware of Sector Trailer Blocks */
int blockNum = 2;
/* Create another array to read data from Block */
/* Legthn of buffer should be 2 Bytes more than the size of Block (16 Bytes) */
byte bufferLen = 18;
byte readBlockData[18];
//----------------------------------------
String card_holder_name;
const String sheet_url =
"https://script.google.com/macros/s/AKfycbxlZSAKH2zoyj55nD9k0TEdQXM3vuZtD9DhlaMRVcIb-
6DL_fkWqaYXfg8IdmYAlBxN-g/exec?name=";
//----------------------------------------
const uint8_t fingerprint[20] = {0xbb, 0xb9, 0x27, 0xfb, 0x7d, 0xf3, 0xa7, 0x1a, 0x57, 0xcc, 0x23, 0xf8,
0x42, 0xe9, 0x10, 0xbe, 0x59, 0x7e, 0x1f,  0xd4};
//----------------------------------------
#define WIFI_SSID "saarthak"
#define WIFI_PASSWORD "sarkar11"
//----------------------------------------
void setup()
{
  /* Initialize serial communications with the PC */
  Serial.begin(9600);
  //Serial.setDebugOutput(true);
  //------------------------------------------------
  //WiFi Connectivity
  Serial.println();
  Serial.print("Connecting to AP");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(200);
  }

 Serial.println("");
```

```
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  //------------------------------------------------
  /* Set BUZZER as OUTPUT */
  pinMode(BUZZER, OUTPUT);
  pinMode(LED, OUTPUT);
  //------------------------------------------------
  /* Initialize SPI bus */
  SPI.begin();
  //------------------------------------------------
}
 void loop()
{
  //------------------------------------------------
  /* Initialize MFRC522 Module */
  mfrc522.PCD_Init();
  /* Look for new cards */
  /* Reset the loop if no new card is present on RC522 Reader */
  if ( ! mfrc522.PICC_IsNewCardPresent()) {return;}
  /* Select one of the cards */
  if ( ! mfrc522.PICC_ReadCardSerial()) {return;}
  /* Read data from the same block */
  //------------------------------------------------
  Serial.println();
  Serial.println(F("Reading last data from RFID..."));
  ReadDataFromBlock(blockNum, readBlockData);
  /* If you want to print the full memory dump, uncomment the next line */
  //mfrc522.PICC_DumpToSerial(&(mfrc522.uid));

  /* Print the data read from block */
  Serial.println();
  Serial.print(F("Last data in RFID:"));
Serial.print(blockNum);
Serial.print(F(" --> "));
for (int j=0 ; j<16 ; j++)
{
  Serial.write(readBlockData[j]);
}
Serial.println();
//------------------------------------------------
digitalWrite(BUZZER, LOW);
digitalWrite(LED, HIGH);
delay(200);
digitalWrite(BUZZER, HIGH);
digitalWrite(LED, LOW);
delay(400);
digitalWrite(BUZZER, LOW);
digitalWrite(LED, HIGH);
delay(200);
digitalWrite(BUZZER, HIGH);
digitalWrite(LED, LOW);
//------------------------------------------------


if (WiFi.status() == WL_CONNECTED) {
  //-------------------------------------------------------------------------------
  std::unique_ptr<BearSSL::WiFiClientSecure>client(new BearSSL::WiFiClientSecure);
  //-------------------------------------------------------------------------------
  client->setFingerprint(fingerprint);
  // Or, if you want to ignore the SSL certificate
  //then use the following line instead:
  // client->setInsecure();
  //-----------------------------------------------------------------
  card_holder_name = sheet_url + String((char*)readBlockData);
  card_holder_name.trim();
  Serial.println(card_holder_name);
  //------------------------------------------------------------------
  HTTPClient https;
  Serial.print(F("[HTTPS] begin...\n"));
  //------------------------------------------------------------------
  if (https.begin(*client, (String)card_holder_name)){
    //------------------------------------------------------------------
    // HTTP
    Serial.print(F("[HTTPS] GET...\n"));
    // start connection and send HTTP header
    int httpCode = https.GET();
    //------------------------------------------------------------------
    // httpCode will be negative on error
    if (httpCode > 0) {
      // HTTP header has been send and Server response header has been handled
      Serial.printf("[HTTPS] GET... code: %d\n", httpCode);
      // file found at server
       digitalWrite(BUZZER, LOW);
       digitalWrite(LED, HIGH);
       delay(1000);
       digitalWrite(BUZZER, HIGH);
       digitalWrite(LED, LOW);
    }
    //------------------------------------------------------------------
    else
    {Serial.printf("[HTTPS] GET... failed, error: %s\n", https.errorToString(httpCode).c_str());}
    //------------------------------------------------------------------
    https.end();
    delay(1000);
  }

  else {
    Serial.printf("[HTTPS} Unable to connect\n");
  }
}
```

```
}
void ReadDataFromBlock(int blockNum, byte readBlockData[])
{
  for (byte i = 0; i < 6; i++) {
    key.keyByte[i] = 0xFF;
  }
  //--------------------------------------------------------------------------
  /* Authenticating the desired data block for Read access using Key A */
  status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, blockNum, &key,
&(mfrc522.uid));
  //--------------------------------------------------------------------------s
  if (status != MFRC522::STATUS_OK){
     Serial.print("Authentication failed for Read: ");
     Serial.println(mfrc522.GetStatusCodeName(status));
     return;
  }
  //--------------------------------------------------------------------------
  else {
    Serial.println("Authentication success");
  }
  //--------------------------------------------------------------------------
  /* Reading data from the Block */
  status = mfrc522.MIFARE_Read(blockNum, readBlockData, &bufferLen);
  if (status != MFRC522::STATUS_OK) {
    Serial.print("Reading failed: ");
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  //--------------------------------------------------------------------------
  else {
    Serial.println("Block was read successfully");
  }
  //--------------------------------------------------------------------------
}
```

User manual:

1. Code Upload and Card Configuration:

    - First, upload the custom-written code to the NodeMCU device, configuring it to facilitate RFID card reading and writing.
    - Within the code, provide the system with the desired name or identifier to associate with the RFID card that will be written.

2. RFID Card Initialization and Attendance Recording:

    - Initialize the RFID card for attendance recording by following these steps:
        - Load the customized code onto the NodeMCU microcontroller.
        - Proceed to scan the RFID card by bringing it into proximity with the RFID sensor.
        - Upon successful card detection, the system will generate two short, distinct beeping sounds using the buzzer to indicate card recognition.
        - Remain patient and attentive until you observe a prolonged beep, accompanied by a green indicator light. This signifies that the attendance data has been securely recorded and stored within the system.