VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnanasangama, Belagavi-590018

A Micro Project Report

on

ATTENDANCE SYSTEM

Submitted as part of additional knowledge in Embedded Controller subject

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND INSTRUMENTATION ENGINEERING

Submitted by

N S VINAY	1BI21EI022
PRAJWAL H R	1BI21EI024

Under the Guidance of the Course Coordinator

Mr. ANJAN KUMAR B S

Assistant Professor, Dept of EIE, BIT

Department of Electronics and Instrumentation Engineering

Bangalore Institute of Technology

K R Road, V V Pura, Bengaluru-560004

TABLE OF CONTENT

1. Introduction:

The IoT-based Attendance System using NodeMCU and RFID is a project that leverages the power of Internet of Things (IoT) technology to streamline and automate the process of tracking attendance. Traditional methods of taking attendance can be time-consuming and prone to errors. This project addresses these challenges by combining the NodeMCU, an ESP8266-based microcontroller with built-in Wi-Fi capabilities, and Radio-Frequency Identification (RFID) technology.

2. Components Required:

The components required for Attendance System set-up are as follows:

1. NodeMCU ESP8266
2. MFRC522 RFID Card Module
3. 5V Buzzer
4. Solderless Bread Board
5. Jumper wires

3. Circuit Diagram:

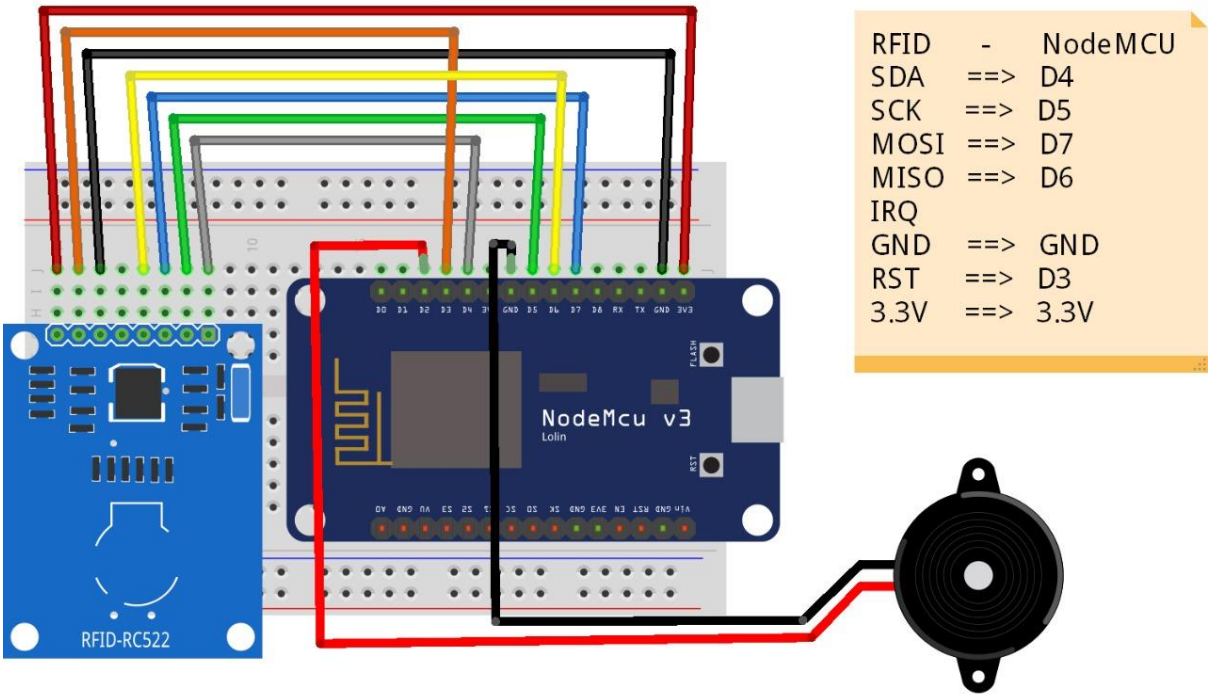

Table 1.1

4. Setup:

i. Hardware Setup:

- Take up NodeMCU (ESP8266) and RFID Module (MFRC522), and connect them to each other using a breadboard and jumper wires. Follow the below table 1.1 for the connections.
- A buzzer is connected to NodeMCU to recognize the updation of the attendance (RFID tag) when it is scanned through an RFID reader and the pins description is explained in table 1.2.
- The connections are done as per the circuit diagram.

MFRC522 Pin	NodeMCU Pin (ESP8266)	Description
3.3V	3.3V	Power Supply (3.3V)
RST	D0(GPIO16)	Reset Pin
GND	GND	Ground (0V)
MISO	D6(GPIO12)	SPI Master Input/Slave Output (Data from MFRC522 to NodeMCU)
MOSI	D7(GPIO13)	SPI Master Output/Slave Input (Data from NodeMCU to MFRC522)
SCK	D5(GPIO14)	SPI Clock
SDA(SS)	D8(GPIO15)	SPI Slave Select (Chip Select)

- Connect the USB A-B cable from NodeMCU to PC.

Table 1.2

ii. Software Setup:

- Install Arduino IDE Software. Open a new tab and write the code.
- Once the code is ready, go to tools in the menu bar, select board>>uno board and port>>respective COM port.
- Compile the code and then upload it to the NodeMCU board.
- Here we have interfaced a Google sheet using code with the Node MCU, which helps to collect the Data of the attendee (which will be already fed in the RIFD tag in Hex code).
- Open the Google sheet, and bring one RFID tag in contact with its reader. The data inside the tag will be recorded in the sheets with the date and time.

5.Code:

```
#include <SPI.h>
#include <MFRC522.h>
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#include <WiFiClientSecureBearSSL.h>

#define RST_PIN  D3
#define SS_PIN   D4
#define BUZZER   D2

MFRC522 mfrc522(SS_PIN, RST_PIN);
MFRC522::MIFARE_Key key;
ESP8266WiFiMulti WiFiMulti;
MFRC522::StatusCode status;

int blockNum = 2;
byte bufferLen = 18;
byte readBlockData[18];
String data2;
const String data1 =
"https://script.google.com/macros/s/AKfycbyaB2sYxNQVRF92JFG5HQihwUbfNFQboFr
2frUIZA5MQCqu7WPBUzCYSkyh1cn0BHm3KQ/exec?name=";
void setup()
{
    Serial.begin(9600);
    Serial.println();
  Serial.println();
  Serial.println();
  for (uint8_t t = 4; t > 0; t--)
  {
    Serial.printf("[SETUP] WAIT %d...\n", t);
    Serial.flush();
    delay(1000);
  }
  WiFi.mode(WIFI_STA);
    /* Put your WIFI Name and Password here */
  WiFiMulti.addAP("NordCE2", "Dheeraj$02");
   pinMode(BUZZER, OUTPUT);
  /* Initialize SPI bus */
  SPI.begin();
}
void loop()
{
```

```
  mfrc522.PCD_Init();
  /* Look for new cards */
  /* Reset the loop if no new card is present on RC522 Reader */
 if ( ! mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }

  if ( ! mfrc522.PICC_ReadCardSerial())
  {
    return;
  }
  Serial.println();
  Serial.println(F("Reading last data from RFID..."));
  ReadDataFromBlock(blockNum, readBlockData);
     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, HIGH);
  delay(200);
  digitalWrite(BUZZER, LOW);
  delay(200);
  digitalWrite(BUZZER, HIGH);
  delay(200);
  digitalWrite(BUZZER, LOW);
  if ((WiFiMulti.run() == WL_CONNECTED))
  {
    std::unique_ptr<BearSSL::WiFiClientSecure>client(new
BearSSL::WiFiClientSecure);
    client->setInsecure();
    data2 = data1 + String((char*)readBlockData);
    data2.trim();
    Serial.println(data2);
        HTTPClient https;
    Serial.print(F("[HTTPS] begin...\n"));
    if (https.begin(*client, (String)data2)
    {
         Serial.print(F("[HTTPS] GET...\n"));
           int httpCode = https.GET();

      if (httpCode > 0)
      {
           Serial.printf("[HTTPS] GET... code: %d\n", httpCode);
       }
```

```
        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;
   }

   status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A,
 blockNum, &key, &(mfrc522.uid));

   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");
   }
 }
```

6. Working:

The Attendance System using IoT NodeMCU and RFID, specifically the MFRC522 module, operates by seamlessly integrating hardware components and software to automate the attendance tracking process. It begins with the enrollment of participants who are issued RFID (Radio Frequency Identification) cards or tags containing unique identification information(in HEX code). When a participant approaches the MFRC522 reader, the reader wirelessly communicates with the RFID tag, extracting its unique identifier. The NodeMCU, equipped with Wi-Fi capabilities, serves as the central processing unit. It receives the RFID data from the reader and cross-references it with a pre-established database of participants, instantly identifying the individual. Subsequently, the system records the attendance data either locally on the NodeMCU or remotely on a central server, depending on its connectivity status. The collected data can be accessed through a user-friendly interface, Here we have interfaced with a Google sheet using some code, which helps to collect the Data and store it for future use.

7. Advantages:

- Real-time: The system provides real-time updates, reducing the need for manual data entry and updates.
- Accuracy: RFID technology ensures accurate tracking of attendance, minimizing errors.
- Efficiency: Automating the attendance process saves time and reduces administrative workload.
- Remote Management: Cloud-based storage enables easy access to attendance records from anywhere.

8. Disadvantages:

- Security Concerns: RFID technology, if not properly secured, can be vulnerable to cloning or unauthorized access. Ensuring the security of RFID cards/tags and the communication between the NodeMCU and the server is crucial.
- Cost: Implementing an IoT-based attendance system with NodeMCU and RFID technology can be relatively expensive.
- Dependence on Technology: Relying on technology for attendance tracking can become a single point of failure. If the system experiences technical issues or malfunctions, it can disrupt attendance tracking entirely.

9. Outcomes:

This project enlightened us about testing board like nodeMCUesp8266, how the real world application such as Automated Attendance system works

There were many challenges we faced while working on this project. Identifying a original nodeMCUesp8266 board and accessing the RFID tag was a new learning. We gathered different programs across internet. Analysed, altered and combined many codes to get the output we wanted.

We faced few issues while working this project. Our RFID tag used to not scan when we introduced it to the reader .After proper soldering it started to work and power the RFID properly, then our nodeMCUesp8266 board was connecting to WI-FI ,it was resolved after setting up our board to work in 2.4GHz and update the data read from the RFID tag to the linked Google sheets.

10. Reference:

- YouTube link
  https://youtu.be/97samwAA-TA

- Google sheet link
  https://docs.google.com/spreadsheets/d/1IyM5_R0Aj3RVuJtRDoEzpCSO2jnJrQPQR0jo2dM665Y/edit?usp=sharing