International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

RFID Based Attendance System for Department

Ms. Krupali Panchal[1*], Ms. Parthvi Jingar[2], Mr. Om Vataliya[3], Mr. Harshul Rathod[4]

Submitted: 06/12/2023 **Revised:** 17/01/2024 **Accepted:** 27/01/2024

Abstract: On the current digital era, in each and every sectors technology has become the most pioneer need. Also it has become mandatory to maintain the pace with the fast growing world that's how the time management is more important. Nowadays to reduce the human efforts and make the task smoother and easier, Internet of Things with many more sectors is in the market. With the help of Internet of Things, a proper combination of hardware and software implementation can be proved more helpful. In the class full of students or a company having good number of employees, has to adapt a technology to track the attendance of the attendees. To be helpful in this way for time management, RFID based attendance system is designed.

Keywords: *IoT, RFID, Arduino based project, Smart Attendance System Project*

1. Introduction

The manual management of attendance records can be arduous and time-consuming. Traditional attendance systems are susceptible to errors, resulting in the generation of inaccurate data and the misallocation of resources. A smart attendance system represents a modern and technologically advanced approach to tracking and managing attendance in various settings, such as schools, universities, businesses, and other organizations. This innovative system leverages cutting-edge technology to streamline the traditionally time-consuming and error-prone task of attendance management. By combining elements of IoT (Internet of Things) and data analytics, smart attendance systems offer a more efficient and accurate way to record and monitor the presence of individuals. These systems can range from simple smartphone apps to sophisticated RFID card scanning devices. Such a system not only automates the process of attendance-taking but also provides valuable insights and data for administrators and educators, facilitating informed decision-making. This system is designed with the IoT tools with the research purpose. In this system Arduino code is designed to create an RFID-based attendance system that integrates with Google Sheets. The system uses an ESP8266 board (NodeMCU), an RFID reader (MFRC522), and a Liquid Crystal Display (LCD) to read RFID tags and record attendance data in a Google Sheets spreadsheet.

2. Workflow of the System

A smart attendance system using the Internet of Things (IoT) is a sophisticated and efficient solution for tracking and managing attendance across different settings. Here's a general workflow of how such a system operates:

1. Device Enrolment: The process begins with the enrolment of users or students in the system. Each user is assigned a unique identifier, which will be the Identity Card with RFID tag.

2. Sensor Deployment: IoT sensors are strategically placed in the attendance areas, such as classrooms, laboratories, offices, or other relevant locations. These sensors can include RFID readers.

3. Data Collection: When a user wants to enter in an area, RFID readers will capture the unique identifier from the user's RFID tag enabled identity card and collect the data about user's presence.

4. Data Transmission: The collected data is then transmitted to a central IoT gateway. This can occur wirelessly using protocols like Wi-Fi.

5. Data Processing: The central server processes the data to identify users and record their attendance. The data will be stored in a Google sheet so the admin users can access it remotely from anywhere.

6. Attendance Recording: The system maintains a real-time attendance record, updating it as users enter or leave the area. This record can be accessed by authorized personnel.

7. Data Storage and Analysis: Attendance data is stored securely in a Google Sheet for future reference and analysis. Long-term data can be used for trend analysis, attendance evaluation, and generating attendance reports.

8. Notifications and Alerts: The system is configured to alert a trespassing of any user without scanning their identity card administrators. The buzzer will be

[1*,2,3,4]*Department of Computer Applications, Faculty of Science, The Maharaja Sayajirao University of Baroda Vadodara, Gujarat, India*

constantly producing an alert until the card is scanned. For the authenticate user, the mark of the attendance will be displayed on the LED screen.

9. Maintenance and Updates: Regular maintenance and updates will be taken care to ensure the system's reliability, security, and compatibility with evolving IoT technologies.

By implementing a smart attendance system using IoT, organizations can significantly improve attendance management, reduce manual errors, enhance security, and obtain valuable insights from attendance data for better decision-making.

This project combines RFID technology with Google Sheets integration to create an attendance system. The Arduino-based system uses an MFRC522 RFID reader to read RFID tags and records the attendance data in a Google Sheets spreadsheet via the ESP8266 (NodeMCU) board. Additionally, an ultrasonic sensor is employed to control the attendance system based on proximity.

1. The system connects to the WiFi network and the Google Sheets server during the setup phase.

2. The RFID reader continuously looks for new RFID cards. When a card is detected, its unique ID is read and stored.

3. The system waits for user input via the serial monitor, allowing administrators to toggle the ultrasonic sensor's functionality.

4. If the ultrasonic sensor is enabled, the system uses it to detect nearby individuals. When someone is in close proximity (within a certain range), their attendance is marked.

5. The system prepares the attendance data along with the gate/room number and sends it as a JSON payload to the Google Sheets script.

6. The Google Sheets script processes the data and updates the attendance record in the corresponding sheet.

7. Real-time feedback is displayed on the LCD, showing the user's ID and attendance status (e.g., Present).

8. The system ensures a delay before the next attendance marking to prevent multiple entries for the same individual.

For setting up and configure this system the Google Sheets script should be set up separately with the provided deployment ID and function to handle incoming data. Also modify the network credentials, Google Sheets script ID, and other settings in the code to suit the specific setup. Ensure proper hardware connections, especially for the RFID reader, LCD, and optional ultrasonic sensor.

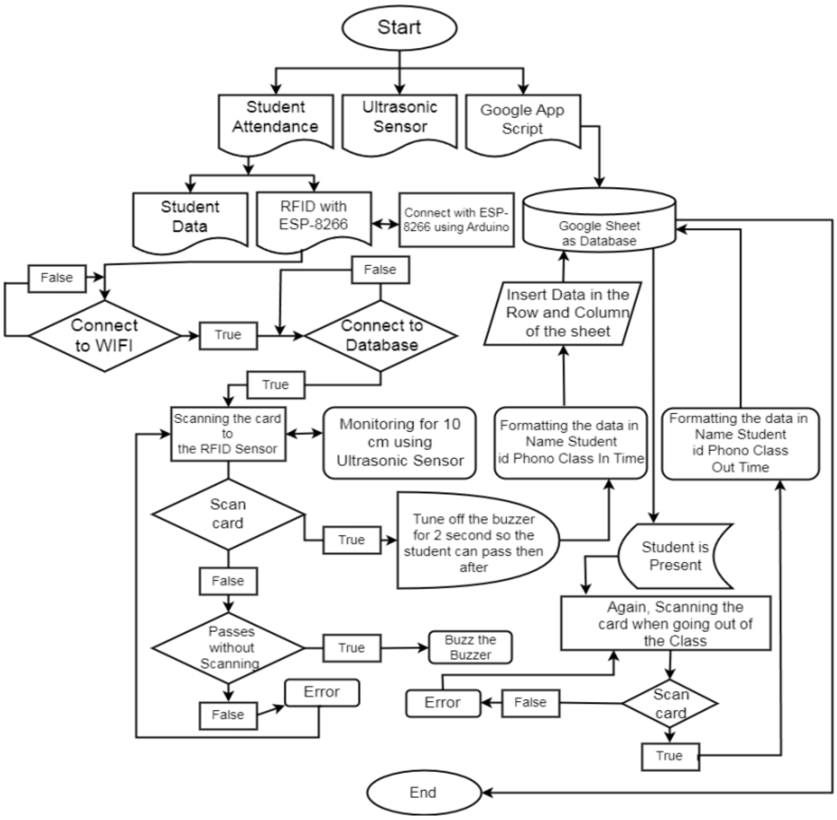

Fig 1: Flow diagram of system

In this system some IoT hardware components are used.

1.ESP8266 (NodeMCU) Board: The ESP8266 module, often used in conjunction with development boards like NodeMCU, is a popular and versatile microcontroller module designed for IoT (Internet of Things) and embedded systems projects. ESP8266 Module is the heart of the NodeMCU development board. It is a low-cost, low-power, and highly integrated system-on-a-chip (SoC) that includes a microcontroller unit (MCU) and Wi-Fi capabilities. The ESP8266 module is equipped with a Tensilica L106 32-bit RISC processor, which can run at varying clock speeds, typically around 80 MHz. It has GPIO (General Purpose Input/Output) pins for interfacing with sensors, displays, and other hardware components. One of the key features of the ESP8266 is its built-in Wi-Fi connectivity. It can connect to local Wi-Fi networks, making it suitable for IoT applications where remote communication and data exchange are essential. The ESP8266 module includes both program memory (flash memory) and RAM. The amount of memory can vary depending on the specific ESP8266 module variant. Developers must manage memory effectively to ensure stable operation. NodeMCU is a popular open-source development platform that utilizes the ESP8266 module. It provides a convenient way to program and use the ESP8266, featuring USB-to-serial communication, a voltage regulator, and GPIO pin headers for easy connections to external components. The ESP8266 ecosystem has a large and active community, which means extensive online documentation, forums, and community-contributed libraries and projects. This support network can be very helpful for developers working with the ESP8266. The NodeMCU board typically operates at 3.3V, and it provides a convenient voltage regulator that allows it to be powered via USB or an external power source. NodeMCU boards feature GPIO pins that can be used for interfacing with various sensors, actuators, and other devices. Some of these pins have additional functionalities like PWM, I2C, and UART. NodeMCU boards come with a USB-to-serial interface, making it easy to upload code to the ESP8266 module and monitor serial output.

2. MFRC522 RFID Reader Module: The MFRC522 RFID Reader Module is a popular RFID (Radio-Frequency Identification) module used to read and interact with RFID tags and cards. RFID is a generation that makes use of radio waves to become aware of and tune objects. RFID systems consist of RFID tags or cards and RFID readers. The MFRC522 module is used as the reader component. The MFRC522 operates at the 13.56 MHz frequency with power supply at 3.3V or 5V, depending on the model and the microcontroller it is connected to. The MFRC522 module typically includes an antenna, an RFID chip, and support circuitry for communication with microcontrollers or other host systems. The MFRC522 module can read RFID tags or cards within its proximity. When an RFID tag is brought near the reader's antenna, it activates a radio-frequency field and the reader communicates with the tag, exchanging data and often retrieving a unique identifier or information stored on the tag. This module supports various communication protocols, including SPI (Serial Peripheral Interface) for connecting to microcontrollers like Arduino and Raspberry Pi. The reading range of the MFRC522 is typically a few centimetres, which makes it suitable for applications where proximity-based identification is required. The MFRC522 module often provides support for authentication and security features. This means that RFID tags or cards can be secured with encryption and authentication mechanisms to prevent unauthorized access. The MFRC522 RFID Reader Module is a versatile component for RFID-based projects, and its ease of use and availability of libraries make it a popular choice for professionals working on applications that require RFID technology.

3. 16x2 I2C Liquid Crystal Display (LCD): A 16x2 I2C Liquid Crystal Display (LCD) is a popular display module used in electronics and microcontroller-based projects. The "16x2" in the name indicates the display's size. It means the LCD has 16 character positions in each of its two rows. Each position can display a character or symbol. LCDs are used for displaying alphanumeric characters, numbers, and simple graphics. They are commonly used in various electronic devices, including microcontroller-based projects, to provide visual feedback and information. The "I2C" (Inter-Integrated Circuit) interface is a serial communication protocol that simplifies the wiring and control of the LCD. With the I2C interface, you only need two wires (SDA and SCL) to connect the LCD to a microcontroller, reducing the number of required pins and making it easy to interface with various platforms. This LCD is having a built-in LED backlight, which can be controlled to adjust the display's visibility in different lighting conditions. Most 16x2 LCDs use the HD44780 controller (or compatible controllers), which simplifies the process of programming and displaying text on the screen. This controller is well-supported in libraries for various microcontroller platforms. The display typically supports a character set that includes numbers, letters (both uppercase and lowercase), symbols, and some special characters. The power supply voltage for 16x2 I2C LCDs typically ranges from 3.3V to 5V, depending on the specific module. It is critical to give the optimum voltage to ensure effective operation. To use the 16x2 I2C LCD, code is required that sends data and commands to the display via the I2C interface.

4. Buzzer: The only use of buzzer is to alert the user or the administrator that the entry has been occurred without scanning the card.

5. Ultrasonic Sensor: An ultrasonic sensor is a device that uses ultrasonic sound waves for distance measurement and object detection. It operates on the principle of

echolocation, much like how bats navigate by emitting sound waves and listening for the echoes to determine the distance to objects. Ultrasonic sensors are made up of a transmitter and a receiver. The transmitter emits high-frequency sound waves (ultrasonic waves), typically in the ultrasonic range above 20 kHz. These waves travel through the air or another medium. In this system Ultrasonic sensors are used for detecting the presence of a user in specified range.

In this module, major libraries like arduino.h, ESP8266WiFi.h, SPI.h, MFRC522.h, HTTPSRedirect.h, wire.h, liquidCrystal_I2C.h and softwareSerial.h are used. Arduino.h is a standard Arduino library. ESP8266WiFi.h is used for handling WiFi connectivity on ESP8266. SPI.h is used for SPI communication. MFRC522.h is used for the MFRC522 RFID reader. HTTPSRedirect.h is used for handling HTTPS communication. Wire.h is used for I2C communication with the LCD. LiquidCrystal_I2C.h is used for driving the I2C LCD. SoftwareSerial.h is used for software serial communication.

To record all data in Google sheet, it should be sestup properly. A Google Sheets script should be created with the deployment ID which is GScriptId. The Google Sheets script should contain a function to handle incoming data from the Arduino. The function should insert or append rows with the data provided by the Arduino.

The next important key role is of network which allows communication between all the hardware components. This setup requires WiFi network credentials (SSID and password). The code establishes a connection to the Google Sheets server using HTTPSRedirect and the provided data (not recommended to leave blank for production).

With the configuration of a network and a data collection tool, some variables are used in the code. A gate_number represents the name of the location or gate, class number or any laboratory number. A variable ssid and password to hold WiFi network credentials for connecting the ESP8266 to the network. Another variable GScriptId to access the Google Sheet script using Google Sheets script deployment ID. A payload_base is used for Base JSON string for creating the payload to be sent to Google Sheets. A payload variable is used to complete JSON string containing the data to be sent to Google Sheets. Another variable blocks [] is an array containing the block numbers of the RFID card where data is stored. The one more total_blocks variable shows a total number of blocks to be read. RST_PIN, SS_PIN, BUZZER used for pin numbers for the RFID reader's reset, slave select, and the buzzer. A key variable is an authentication key used to read the data from the RFID card. A status variable represents the status of the RFID reader's operations (e.g., authentication success/failure). An ultrasonicEnabled variable is a boolean flag to control the ultrasonic sensor's functionality (optional).

With these variables some functions are also used.

1. setup(): Initializes the necessary components, connects to WiFi and Google Sheets, and sets up the RFID reader.

2. loop(): The main loop function, reads the RFID tags, retrieves data from the RFID card, prepares the payload, and sends it to Google Sheets.

3. ReadDataFromBlock(): Reads data from the RFID card block using authentication.

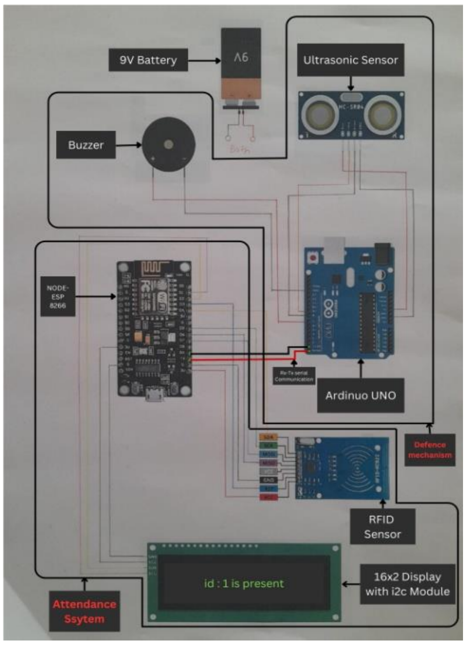

Fig 2: Components diagram

The code initializes the components and connects to the WiFi network. It establishes a connection to the Google Sheets server. It waits for an RFID tag to be scanned by the RFID reader. Upon scanning, it reads the data from the RFID card and stores it in the 'values' string. It creates a JSON payload using the 'values' and 'gate_number'. The payload is sent to the Google Sheets script via HTTPS request. If the data is published successfully, it displays the student's ID and 'Present' on the LCD and optionally toggles the ultrasonic sensor state based on the student's ID. The code uses 'SoftwareSerial' to communicate with the ultrasonic sensor, which may require additional hardware connections.

3. Ultrasonic Sensor Control with Nodemcu (ESP8266) and Serial Communication

The ESP8266 can be programmed using a variety of programming languages, including Arduino IDE (using the ESP8266 core), MicroPython, Lua, and more. This versatility allows developers to choose the language that best suits their project requirements and familiarity. The

Arduino code allows the control of an ultrasonic sensor connected to a NodeMCU (ESP8266) board. The code uses the SoftwareSerial library to establish communication between the Arduino board and the NodeMCU over serial communication (TX, RX pins). The ultrasonic sensor measures distance, and the code toggles the sensor's functionality based on commands received from the serial monitor.

To make this operational some hardware components are used which includes Arduino Board (e.g., Arduino Uno), NodeMCU (ESP8266) Board, HC-SR04 Ultrasonic Sensor and LED (connected to pin 7). A SoftwareSerial.h library is used with all these for software serial communication between Arduino and NodeMCU. Along with all these another variables are used. An ESP8266Serial variable is used to communicate SoftwareSerial object with the NodeMCU. A trigPin variable is used for Arduino pin connected to the ultrasonic sensor's trigger pin. Another variable echoPin where Arduino pin connected to the ultrasonic sensor's echo pin. A ultrasonicEnabled variable is a boolean flag to control the ultrasonic sensor's functionality.

The code sets up the serial communication for both the Arduino and NodeMCU, and the pin modes for the ultrasonic sensor and LED. In the loop function, the ultrasonicEnabled flag is checked. If it is true, the ultrasonic sensor is enabled, and the code proceeds with distance measurement. The code sends a brief HIGH pulse to the ultrasonic sensor's trigger pin and measures the echo's duration to calculate the distance. If the measured distance is less than or equal to 30 cm, the LED is turned on. The code then checks for commands from the serial monitor (input from the user) using Serial.available() and Serial.parseInt(). If the command "1" is received, the ultrasonicEnabled flag is set to false, and the ultrasonic sensor is disabled. Additionally, the LED blinks twice as feedback. If the command "0" is received, the ultrasonicEnabled flag is set to true, and the ultrasonic sensor is enabled. The LED is turned off as feedback. The loop continues to measure distance and wait for commands from the serial monitor.

4. RFID Data Writing and Reading with MFRC522 RFID Reader

This Arduino code enables the reading and writing of data to an MIFARE 1K RFID card using the MFRC522 RFID reader. The code allows the user to enter various data, here Student ID, First Name, Last Name, and PRN which a unique identifying number generated for each student, via the serial monitor and writes this data to specific blocks on the RFID card. Additionally, the code reads and displays the data stored on the card to verify successful writing.

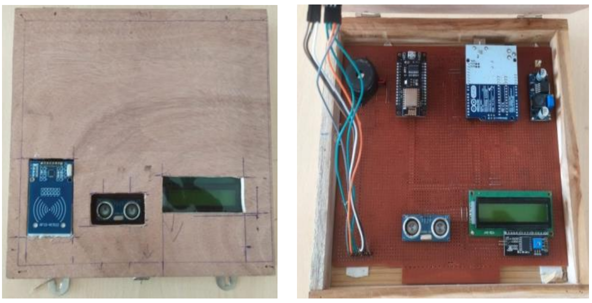

Fig 3: Prototype of working system

In this module certain hardware components, libraries, variables and functions are used. As the hardware components Arduino Uno, MFRC522 RFID Reader Module and MIFARE 1K RFID Card are used. The MFRC522 RFID (Radio-Frequency Identification) Reader Module is a popular RFID module that is often used for reading and interacting with RFID cards and tags. The MFRC522 module uses radio waves to communicate with RFID cards or tags. When an RFID card or tag is brought into close proximity to the module, it sends out radio signals to activate the card's or tag's microchip. The module then communicates with the microchip to read the stored information. The MFRC522 module typically operates at a frequency of 13.56 MHz. It uses the SPI (Serial Peripheral Interface) communication protocol to communicate with microcontrollers like Arduino, Raspberry Pi, or other microcontroller platforms. The MFRC522 module is capable of reading and interacting with various RFID cards and tags, including MIFARE cards. MIFARE is a widely used type of RFID technology, and many access control cards and public transportation cards use MIFARE technology. The typical operating range for the MFRC522 module is a few centimeters to a couple of inches. The exact range may vary depending on the specific antenna used and environmental factors. The module typically comes with an integrated antenna, but it can also be connected to an external antenna for extended range or specific applications. The libraries provide functions for reading and writing data on RFID cards or tags.

With this module two libraries are used: SPI.h - Library for SPI communication and MFRC522.h - Library for the MFRC522 RFID reader.

The variables used in this module are RST_PIN, SS_PIN: GPIO pins connected to the MFRC522 RFID reader (reset and slave select); key: An authentication key used to read/write data to the RFID card; blockNum: The block number to which data will be written or read; bufferLen: The length of the buffer to read data from the RFID card; readBlockData: An array to store data read from the RFID card; and status: Represents the status of the RFID reader's operations (e.g., authentication success/failure).

```
#include <Arduino.h>

#include <ESP8266WiFi.h>

#include <SPI.h>

#include <MFRC522.h>

#include <HTTPSRedirect.h>

#include<Wire.h>

#include<LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

#include<SoftwareSerial.h>

SoftwareSerial abc(3,1);

// Enter Google Script Deployment ID:

const char *GScriptId = "AKfycbyDcafjQZF6fesQScbiNmJ3kb3TcyoHg53Daqx35-ZV8Kf3p7O9kCe8ny9OJq04hb5d";

String gate_number = "Lab-3";

// Enter network credentials:

const char* ssid     = "BCA-Lab3";

const char* password = "c94469446a";

// Enter command (insert_row or append_row) and your Google Sheets sheet name (default is Sheet1):

String payload_base =  "{\"command\": \"insert_row\", \"sheet_name\": \"Sheet1\", \"values\": ";

String payload = "";

// Google Sheets setup

const char* host = "script.google.com";

const int   httpsPort   = 443;

const char* fingerprint = "";

String url = String("/macros/s/") + GScriptId + "/exec";

HTTPSRedirect* client = nullptr;

// Declare variables that will be published to Google Sheets

String student_id;

int blocks[] = {4,5,6,8,9};

#define total_blocks  (sizeof(blocks) / sizeof(blocks[0]))

#define RST_PIN  0  //D3

#define SS_PIN   2  //D4

#define BUZZER   4  //D2

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];

bool ultrasonicEnabled = true;


void setup() {

  Serial.begin(9600);

  delay(10);

  Serial.println('\n');

  abc.begin(9600);

  SPI.begin();

  //initialize lcd screen

  lcd.begin();

  // turn on the backlight

  lcd.backlight();

  lcd.clear();

  lcd.setCursor(0,0); //col=0 row=0

  lcd.print("Connecting to");

  lcd.setCursor(0,1); //col=0 row=0

  lcd.print("WiFi...");

  // Connect to WiFi

  WiFi.begin(ssid, password);

  Serial.print("Connecting to ");

  Serial.print(ssid); Serial.println(" ...");

    while (WiFi.status() != WL_CONNECTED) {

    delay(100);

    Serial.print(".");

  }

  Serial.println('\n');

  Serial.println("WiFi Connected!");

  //Serial.print("IP address:\t");

  Serial.println(WiFi.localIP());

  // Use HTTPSRedirect class to create a new TLS connection
```

```
client = new HTTPSRedirect(httpsPort);
client->setInsecure();
client->setPrintResponseBody(true);
client->setContentTypeHeader("application/json");
  lcd.clear();
lcd.setCursor(0,0); //col=0 row=0
lcd.print("Connecting to");
lcd.setCursor(0,1); //col=0 row=0
lcd.print("Google ");
delay(100);

Serial.print("Connecting to ");
Serial.println(host);
// Try to connect for a maximum of 5 times
bool flag = false;
for(int i=0; i<5; i++){
  int retval = client->connect(host, httpsPort);
  if (retval == 1){
   flag = true;
    String msg = "Connected. OK";
    Serial.println(msg);
    lcd.clear();
    lcd.setCursor(0,0); //col=0 row=0
    lcd.print(msg);
    delay(100);
    break;
  }
     else
    Serial.println("Connection failed. Retrying...");
   }
if (!flag){
  lcd.clear();
  lcd.setCursor(0,0); //col=0 row=0
  lcd.print("Connection fail");
      Serial.print("Could not connect to server: ");
  Serial.println(host);
  delay(1000);
  return;
  }
    delete client;    // delete HTTPSRedirect object
  client = nullptr; // delete HTTPSRedirect object
}

void ReadDataFromBlock(int blockNum, byte readBlockData[]);
void loop() {
  //Serial.println("[TEST] loop() starts");
  static bool flag = false;
  if (!flag){
    client = new HTTPSRedirect(httpsPort);
    client->setInsecure();
    flag = true;
    client->setPrintResponseBody(true);
    client->setContentTypeHeader("application/json");
  }
  if (client != nullptr){
    //when below if condition is TRUE then it takes more time then usual, It means the device
    //is disconnected from the google sheet server and it takes time to connect again

//NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
    if (!client->connected()){
      int retval = client->connect(host, httpsPort);
      if (retval != 1){
        Serial.println("Disconnected. Retrying...");
        lcd.clear();
        lcd.setCursor(0,0); //col=0 row=0
        lcd.print("Disconnected.");
        lcd.setCursor(0,1); //col=0 row=0
        lcd.print("Retrying...");
        return; //Reset the loop
      }
    }
     }
  else{Serial.println("Error creating client object!"); Serial.println("else");}
```

```
lcd.clear();
lcd.setCursor(0,0); //col=0 row=0
lcd.print("Scan your Tag");
delay(1000);
//Serial.println("[TEST] Scan Your Tag");
/* 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..."));
String values = "", data;
//creating payload - method 2 - More efficient
for (byte i = 0; i < total_blocks; i++) {
ReadDataFromBlock(blocks[i], readBlockData);
  if (i == 0) {
  data = String((char*)readBlockData);
  data.trim();
  student_id = data;
  values = "\"" + data + ",";
  int num = 0;
  if (student_id == "000") {
    // num = random(9)-1;
    abc.write("0");
   // Serial.println("0"); // Print '0' to the serial monitor
  } else {
    abc.write("1");
    // Serial.println(num);
  }
} else {
  data = String((char*)readBlockData);
  data.trim();
  values += data + ",";
 }
}
 values += gate_number + "\"}";
 //--------------------------------------------------------------
 // Create json object string to send to Google Sheets
 // values = "\"" + value0 + "," + value1 + "," + value2 + "\"}"
 payload = payload_base + values;
 //--------------------------------------------------------------
 lcd.clear();
 lcd.setCursor(0,0); //col=0 row=0
 lcd.print("Publishing Data");
 lcd.setCursor(0,1); //col=0 row=0
 lcd.print("Please Wait...");
 delay(100);
   // Publish data to Google Sheets
 Serial.println("Publishing data...");
 Serial.println(payload);

 if (client->POST(url, host, payload)) {
   // do stuff here if publish was successful
   Serial.println("[OK] Data published.");
   lcd.clear();
   lcd.setCursor(0, 0); //col=0 row=0
   lcd.print("Student ID: " + student_id);
   lcd.setCursor(0, 1); //col=0 row=0
   lcd.print("Present");
   delay(100);
   if (student_id == "000") {
     if (ultrasonicEnabled) {
       abc.write("1");   // Send '0' to Arduino to disable ultrasonic sensor
       ultrasonicEnabled = false;      } else {
       abc.write("0");   // Send '1' to Arduino to enable ultrasonic sensor
       ultrasonicEnabled = true;      }
   }
   else{
     delay(100);
```

```
    abc.write("0");
    ultrasonicEnabled = true;    }
  }
   else{
   // do stuff here if publish was not successful
   Serial.println("Error while connecting");
   lcd.clear();
   lcd.setCursor(0,0); //col=0 row=0
   lcd.print("Failed.");
   abc.write("1");
   lcd.setCursor(0,1); //col=0 row=0
   lcd.print("Try Again");
  }
  // a delay of several seconds is required before publishing again
  Serial.println("[TEST] delay(1000)");
  delay(1000);
}
void ReadDataFromBlock(int blockNum, byte readBlockData[])
{
  /* Prepare the ksy for authentication */
  /* All keys are set to FFFFFFFFFFFFh at chip delivery from the factory */
  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));
  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 {
    readBlockData[16] = ' ';
    readBlockData[17] = ' ';
    Serial.println("Block was read successfully");    }
}
```

The code sets up the serial communication, SPI bus, and MFRC522 RFID reader module. It waits for a new RFID card to be detected by calling the function 'mfrc522.PICC_IsNewCardPresent()'. When a card is detected, it reads its UID (unique identifier) and the type of card (e.g., MIFARE 1K) and prints this information to the serial monitor. The code then waits for the user to input data (Student ID, First Name, Last Name, and PRN) via the serial monitor. The input data is read until the user enters '#' to signify the end of data entry. The data is written to specific blocks on the RFID card using the 'WriteDataToBlock()' function. The code then reads and displays the data from the RFID card using the 'ReadDataFromBlock()' and 'dumpSerial()' functions to verify successful writing.

Future expansion

To make this system working more accurately and more secure, the user survey will be conducted with their feedback. Also an integration of biometric inputs of a user will provide more security to the system.

5. Conclusion

This project presents a versatile RFID-based attendance system that integrates with Google Sheets for streamlined attendance management. Additionally, the optional ultrasonic sensor control adds an extra layer of convenience and versatility to the system. With this system, accurate and real-time attendance data can be efficiently recorded and accessed, making it suitable for various educational and organizational applications.

References

[1] M. Y. I. Idris, E. M. Tamil, Z. Razak, N. M. Noor, and L.W. Km, “Smart parking system using image processing techniques in wireless sensor network

environment," Information Technology Journal, Vol 8, pp. 114–127, 2009.

[2] Sam Polniak, "RFID case study book: RFID application stories from around the globe" Abhisam software, 2007.

[3] S. Shepard, (2005), "RFID Radio Frequency Identification", (2005), USA, ISBN:0-07 144299-5.

[4] Frederick E Terman, Radio Engineers Handbook, McGraw Hill, 1943, pages 785 - 786

[5] Fei Hu, Laura Celentano, and Yang Xiao, "Errorresistant RFID-assisted wireless sensor networks for cardiac telehealthcare," Wireless Communications and Mobile Computing, Vol 9, 120 Davinder Parkash, Twinkle Kundu & Preet Kaur pp. 85–101, 2009

[6] Ray Cronin, "RFID versus Barcode," Pharmaceutical Technology, Vol. 32, pp177-178, 2008. S. P. Singh, M. Mccartney, J. Singh, And R. Clarke, "RFID Research And Testing For Packages Of Apparel, Consumer Goods And Fresh Produce In The Retail Distribution Environment," Packaging Technology and Science, Vol. 21,pp. 91–102, 2008.

[7] Kashif Ali and Hossam Hassanein, "Passive RFID for ntelligent transportation systems," 6th IEEE Consumer Communications and Networking Conference, CCNC 2009, January 10 - January 13, 2009.

[8] A. Kumar, D. Parkash, M.V. Kartikeyan, "Planer antennas for passive UHF RFID tag," Progress in Electromagnetics Research, Vol 91, pp. 95–212, 2009.

[9] Basar Oztaysi, Serdar Baysan and Fatma Akpinar, "Radio Frequency Identification, (RFID) in Hospitality," Technovation, Vol 29,pp. 618–624, 2009.

[10] Kamran Ahsan, Hanifa Shah and Paul Kingston "RFID Applications: An Introductory and Exploratory Study," IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 1, No. 3, January 2010.

[11] K. Ahsan, H. Shah, P. Kingston, "Context Based Knowledge Management in Healthcare: An EA Approach", AMCIS 2009, Available at AIS library.

[12] Elisabeth Ilie-Zudor, Zsolt Kemeny, Peter Egri, Laszlo Monostori, "RFID technology and its current application," The Modern Information Technology in the Innovation Processes of the Industrial Enterprises-MITIP 2006, ISBN 963 86586 5 7, pp.29-36.

[13] B. H. Jeong, C. Y. Cheng, V. Prabhu, and B. J. Yu, "An RFID application model for surgerypatient identification,"IEEE Symposium on Advanced Management of Information for Globalized Enterprises,AMIGE 2008, September 28 - September 29,pages 304–306, 2008.

[14] K. Ahsan, H. Shah, P. Kingston, "Role of Enterprise Architecture in healthcare IT", Proceeding ITNG2009, IEEE.

[15] Martin Brandl, Julius Grabner, Karlheinz Kellner,Franz Seifert, Johann Nicolics, Sabina Grabner, and Gerald Grabner, " A low-cost wireless sensor system and its application in dental retainers," IEEE Sensors Journal, Vol 9,pp. 255–262, 2009.

[16] Anuran Chattaraj, Saumya Bansal, and Anirudhha Chandra, "An intelligent traffic control system using RFID,"IEEE Potentials, Vol. 28, pp. 40–43, 2009.

[17] Yunus A. Kathawala and Benjamin Tueck, "The use of RFID for traffic management, "International Journal of Technology, Policy and Management, Vol. 8, pp.111–125, 2008

[18] Agarwal, S.K., Singh, A., Aniraj, V., Pandey, A., Prasad, D., Nath, V. (2020). RFID (MF-RC522) and Arduino Nano-Based Access Control System. In: Nath, V., Mandal, J. (eds) Nanoelectronics, Circuits and Communication Systems. NCCS 2018. Lecture Notes in Electrical Engineering, vol 642. Springer, Singapore. https://doi.org/10.1007/978-981-15-2854-5_50

[19] Anusha, PV., Atul, K., Kshama, PM. & Menita, C., (2016) "Web service for student attendance management system", Int J Adv Resh Sci and Engr, Vol. 5, pp319-323.

[20] Shih-Sung Lin, Min-Hsiung Hung, and Ding-Rong Lai, " Development of a RFID-based missile assembly and test management system," Chung Cheng Ling Hsueh Pao/Journal of Chung Cheng Institute of Technology, Vol. 37,pp. 185–195, 2009.

[21] Ayu, MA. & Ahmad, BI., (2014) "TouchIn: an NFC supported attendance system in a university environment", Int J Inf Educ Tech, Vol. 4, pp448-453

[22] A Review of RFID Sensors, the New Frontier of Internet of Things - PMC (nih.gov)

[23] David Grau Torrent and Carlos H. Caldas, "Methodology for automating the identification and localization of construction components on industrial projects,"Journal of Computing in Civil Engineering, Vol 23,pp. 3–13, 2009.