islington college
(इस्लिङ्टन कलेज)

LONDON METROPOLITAN UNIVERSITY

CS5054NI Advanced Programming and Technologies

50% Group Coursework

Submission: Final Submission

Academic Semester: Spring Semester 2025

Credit: 15 credit semester long module

Group Name:			
SN	**Student Name**	**College ID**	**University ID**
1	Hirdesh Chauhan (L2N5)	NP01NT4A230194	23047525
2	Nikita Bhandari (L2N5)	NP01NT4A230092	23047392
3	Anjan Sainju (L2N5)	NP01NT4A230158	23048519
4	Samir Kumar Singh (L2N5)	NP01NT4A220062	22066994
5	Abhishek Shah (L2N5)	NP01NT4A230186	23047519

GitHub Link:

Assignment Due Date: 15th May 2025

Assignment Submission Date: 15th May 2025

Similarity Index Report

Document 8.docx

Islington College,Nepal

Document Details

Submission ID

trn:oid:::3618:94467900

Submission Date

May 6, 2025, 3:13 AM GMT+5:45

Download Date

May 6, 2025, 3:15 AM GMT+5:45

File Name

Document 8.docx

File Size

22.0 KB

23 Pages

3,277 Words

18,779 Characters

11% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

- 33 Not Cited or Quoted 11%
  Matches with neither in-text citation nor quotation marks
- 2 Missing Quotations 1%
  Matches that are still very similar to source material
- 0 Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

4%	Internet sources
0%	Publications
10%	Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Acknowledgement

I would like to express my sincere gratitude to everyone who supported me throughout the completion of this project.

First and foremost, I am deeply thankful to Mr. Sugat Man Shakya, my lecturer at Islington College, for his invaluable guidance, constructive feedback, and continuous encouragement during this project. His expertise and insight have been instrumental in shaping the outcome of my work.

I would also like to extend my heartfelt appreciation to Mr. Jaganath Paudyal, my tutor, for his support, timely supervision, and helpful suggestions that greatly assisted me throughout the research and development phases.

A special thanks goes to Islington College, affiliated with London Metropolitan University, for providing me with the necessary academic environment and resources to undertake this project successfully.

Lastly, I would like to acknowledge my family, friends, and peers for their constant motivation and support during the challenging times of this academic journey.

Abstract

This study outlines the creation of an RFID-enabled attendance management solution leveraging ESP32 microcontroller capabilities. The platform aims to optimize and digitize student attendance tracking in academic environments, targeting common issues such as human error, time-consuming processes, and fraudulent check-ins. Students use RFID-enabled identifiers that interface with the ESP32 hardware, enabling instant digital registration. Real-time cloud-based documentation through Google Sheets connectivity ensures secure remote access to attendance records. Supplementary components include a visual interface for system updates and auditory alerts for user interaction, improving operational transparency.

By transitioning from paper-based logs to automated data capture, the system improves administrative precision while serving as an adaptable, budget-friendly substitute for conventional approaches. The work details the system’s technical architecture, prototype development, validation procedures, and operational outcomes, confirming its practical value in modernizing institutional record-keeping processes. Future expansion could incorporate multi-factor authentication protocols and API integrations with existing campus management platforms.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

IoT stands for the Internet of Things, where the network of physical objects is embedded with sensors, software, and other technologies that can be accessible remotely through remotely. And its purpose is to connect and exchange data over the Internet with other devices and systems. In the 21st century, one of the most important and growing technologies has become IoT over the past few years (Oracle.com, 2024).

In educational institutions and workplaces, manual attendance tracking can be time consuming and prone to errors such as proxy attendance and inaccurate record-keeping. The **IoT-Based Smart Card Attendance System** automates the attendance process using **RFID technology, ESP32 microcontroller, and a cloud-based database** or **google sheet** to ensure accuracy and efficiency.

1.1 Current Scenario

Recently, the number of IoT devices is expected to continue its rapid growth rapidly with the estimation of tens of billions of IoT devices in use over the next few years. This growth will be driven by increased development across industries as well as the development of new use cases and applications. IoT is important for business for several reasons. The core benefits of IoT are improved efficiency, data-driven decision-making, cost savings, and enhanced user experiences (IBM, 2023).

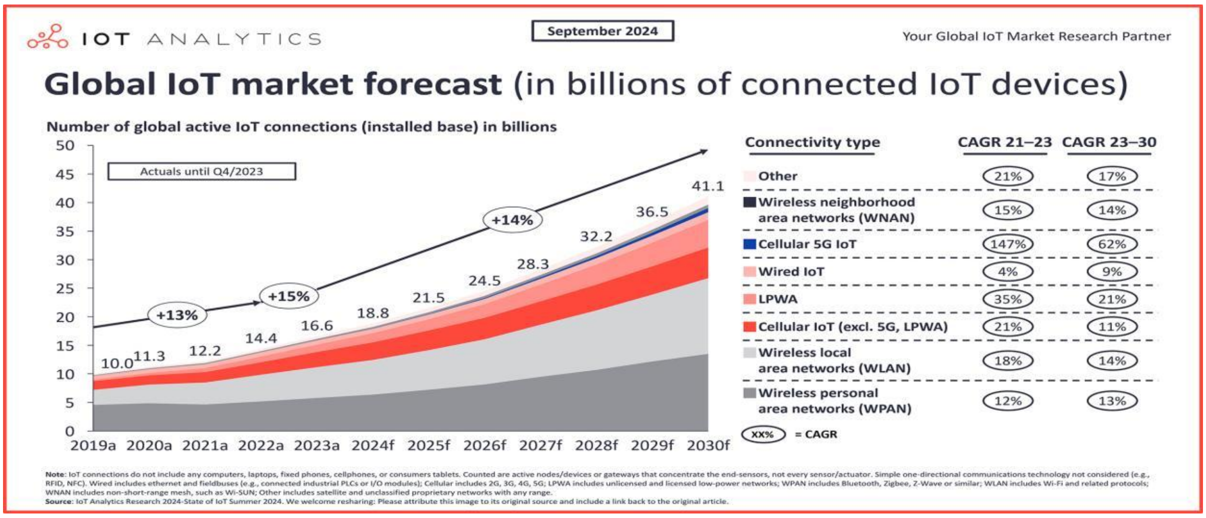

Figure 1: Global IoT Market (Sinha, 2024).

1.2 Problem Statement

In Nepal, most of the schools and colleges have a manual register-based attendance system, though they have identity cards. The traditional attendance system often suffers from:

- Time inefficiency
- Error-prone nature
- Difficult to maintain records
- Limited Accessibility

Furthermore, **biometric scanners** can be slow and unhygienic, especially in post-pandemic environments.

1.3 Project as a Solution

The RFID-Enabled Automated Attendance System offers an innovative technological solution to overcome the limitations of conventional attendance methods, including human errors, inefficient time management, and fraudulent attendance practices. Through its automated RFID identification and cloud-based data management capabilities, the system delivers a dependable, time-saving, and tamper-proof approach to monitor attendance instantly, serving as an optimal solution for both academic institutions and corporate settings.

1.4 Aim and Objectives

This project aims to develop a Smart Card Attendance System that utilizes RFID technology to automate, ensure data accuracy, time efficiency, and real-time tracking.

The objective of this IoT project is:

- To develop a Smart Card Attendance System using ESP32 microcontroller.
- To replace the traditional attendance system with biometric scanners and records.
- To use a cloud-based database or goggle sheet for collecting attendance records.

2. Background

This advanced attendance management system employs IoT technology to revolutionize traditional attendance tracking in educational and corporate environments. At its core, the solution combines RFID smart cards, an ESP32 microcontroller, and cloud computing to create a seamless digital attendance process. Everyone is issued a personalized RFID card containing a unique digital identifier. When the card interacts with the RFID reader, electromagnetic induction activates the tag, transmitting its unique code wirelessly. The ESP32 then processes this information, comparing it against pre-registered credentials in the system database.

2.1 Expected Outcomes and Deliverables

The proposed automated attendance management system harnesses RFID and cloud computing technologies to revolutionize traditional attendance monitoring processes. By implementing this solution, institutions can achieve enhanced precision in recordkeeping, instantaneous data updates, and protected administrator access from any location.

The project outcomes will encompass a functional prototype incorporating RFID scanning components and ESP32 microcontroller functionality, seamlessly connected to cloud-based data storage or google sheet

The architecture of the Smart Card Attendance System has given below:

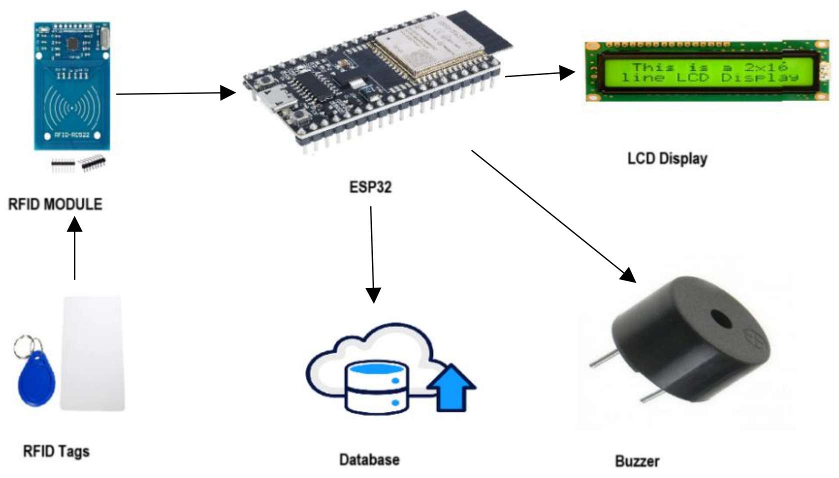

Figure 2: Figure of System Architecture.

2.2 System Overview

2.2.1 Block Diagram

A block diagram is a simplified graphical representation that outlines a system's fundamental components and their interrelationships using labeled rectangles and connecting lines. This high-level visualization method deliberately omits intricate technical details, instead emphasizing core operational functions and data pathways. By presenting systems in this abstracted format, stakeholders across various expertise levels can quickly grasp architectural concepts like how airport signage conveys essential travel information without displaying complex aviation mechanics. The technique effectively balances clarity with comprehensiveness, enabling efficient communication of system designs while preventing information overload (GeeksforGeeks, 2024).

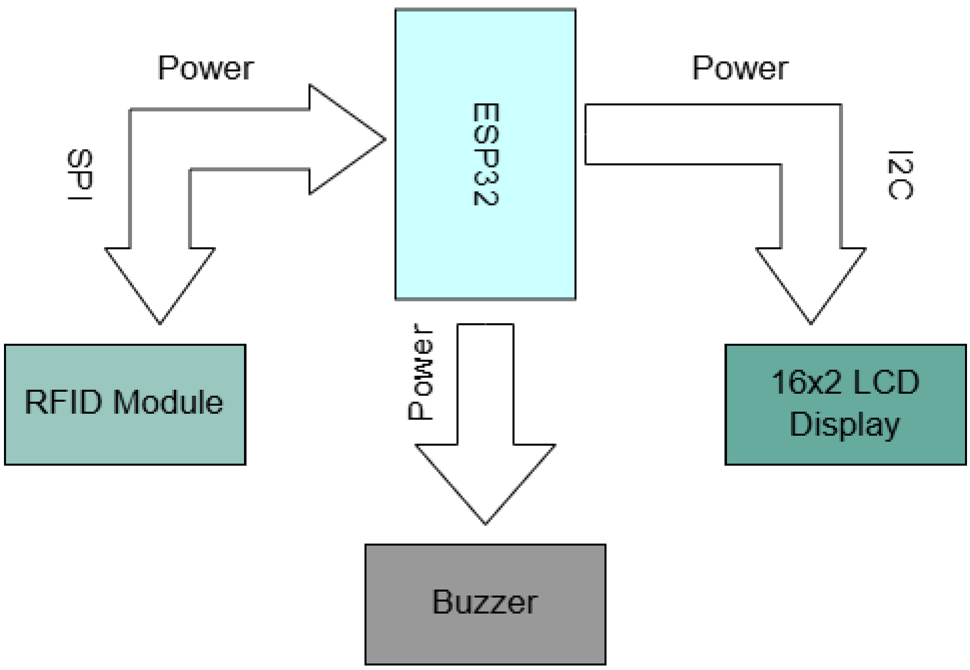

Figure 3: Figure of Block Diagram.

2.2.2 Flowchart

The flowchart of Smart Card Attendance System is

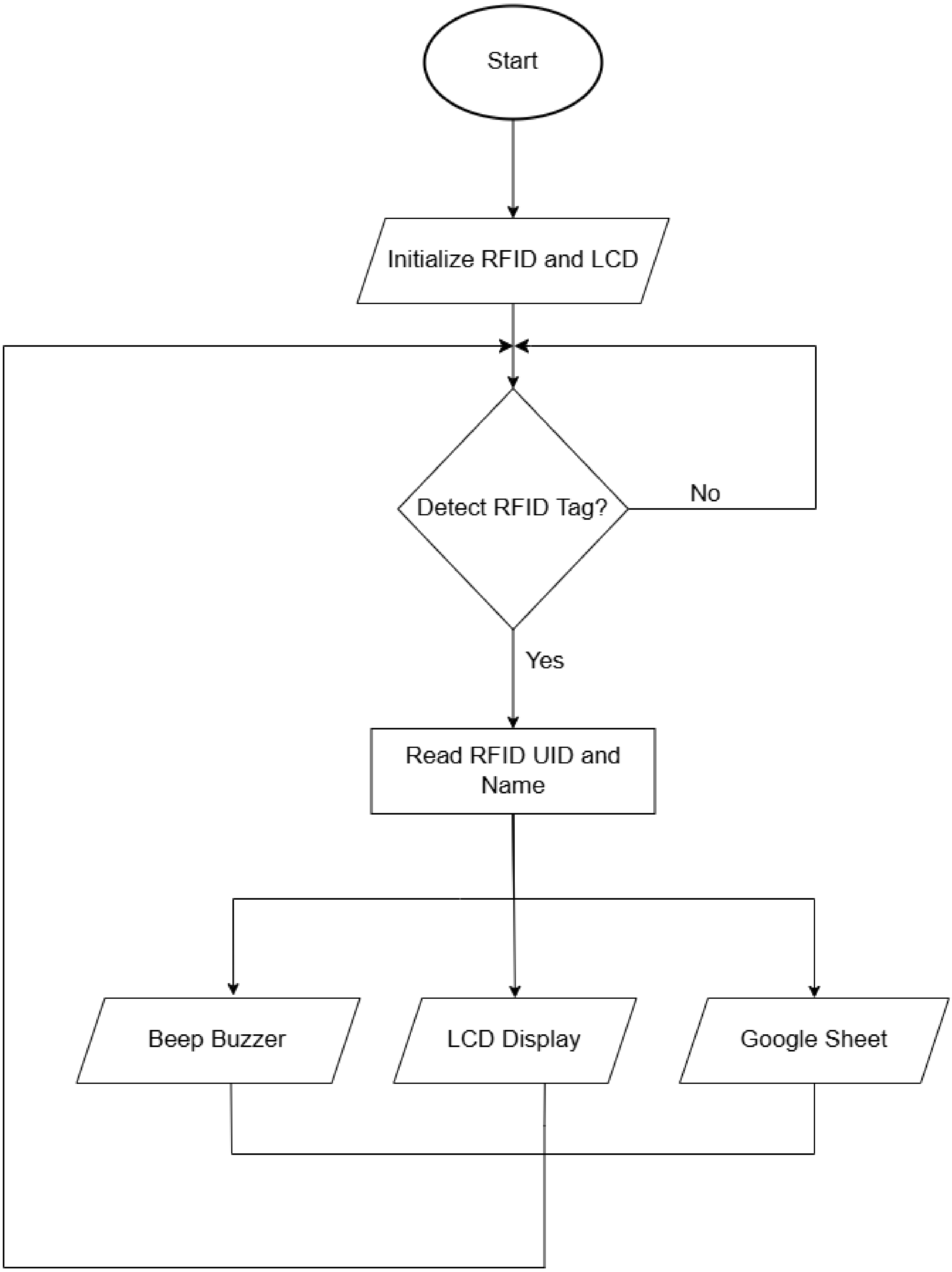

Figure 4: Flowchart of Smart Card Attendance System.

2.2.3 Circuit Diagram

A circuit schematic, often called an electrical blueprint or technical layout, visually simplifies complex electrical systems through symbolic representations. These diagrams act as essential guides for developing, assembling, and servicing electrical infrastructure and devices. By incorporating universally recognized symbols or component-specific visuals, they outline a circuit's architecture, clearly mapping how parts interconnect. This standardized approach allows engineers and maintenance personnel to quickly interpret component placement and operational dependencies, facilitating efficient troubleshooting and system modifications. Unlike photographs or detailed illustrations, schematic designs prioritize clarity over realism, distilling circuits into logical arrangements that highlight functional relationships rather than physical appearances (BYJUS, 2019).

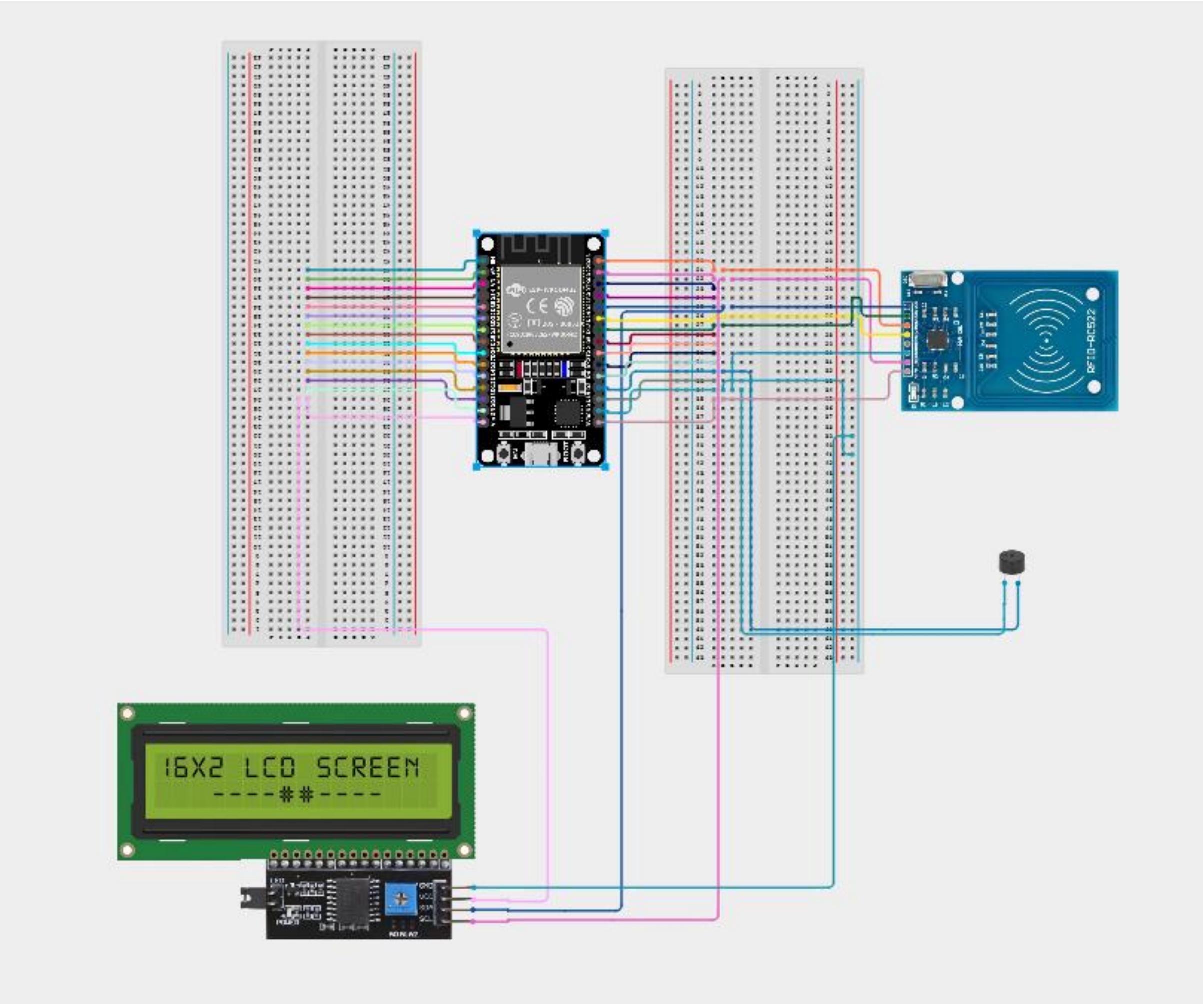

Figure 5: Circuit diagram of Smart Card Attendance System.

2.3 Requirement Analysis

2.3.1 Hardware Components

- RFID Module (RC522)

  RFID (Radio Frequency Identification) Module such as RC522 is a wireless communication operated at 13.56MHz electromagnetic field, which helps to communicate with RFID tags (ISO/IEC 14443A) protocol. This is best suited for short-range wireless communication and is easy to use. It is a low-cost device that consumes less power, best for embedded systems, and has a wide range of applications (Robotix, 2020).

Figure 6: Figure of RFID Module (Robotix, 2020).

- ESP32

  The ESP32 is an integrated circuit designed to enable wireless connectivity (Wi-Fi and optional Bluetooth) in embedded systems, particularly for Internet of Things (IoT) applications. Although the term technically refers only to the microprocessor itself, manufacturers commonly use "ESP32" to describe both the standalone chip and the pre-assembled modules/development boards that incorporate it (Hübschmann, 2020).

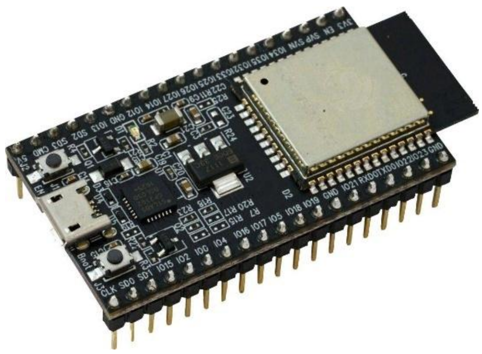

Figure 7: Figure of ESP32 (Hübschmann, 2020).

- RFID Cards

  RFID card is a smart card that has a microchip and antenna embedded, where unique identification data like ID numbers is stored, and encryption is done. It uses radio frequency technology operated at 13.56 MHz for communication with the RFID reader (Zheng, 2025).

Figure 8: Figure of RFID Cards (Zheng, 2025).

- Jumper Wires

  Jumper wires are simple, flexible wires which have connector pins at each end, which are used for connection between two points on a breadboard or development board like the ESP32. IT has three types, which are Male to Male, Male to Female, and Female to Female (Hemmings, 2016).

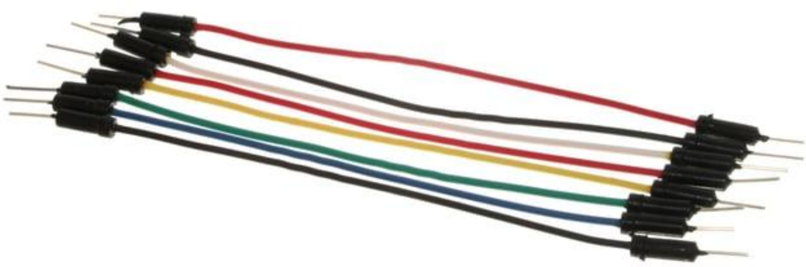

Figure 9: Figure of Jumper Wires (Hemmings, 2016).

- LCD Display (16x2)

  This is a flat panel that displays a module that can show 2 lines with 16 characters each. It is based on the Hitachi HD44780 driver and uses liquid crystal to expose images. LCDs have a thin film transistor that controls the brightness on the screen (Hongguang, 2023).

Figure 10: Figure of LCD Display (Hongguang, 2023).

- Breadboard

  A breadboard serves as a fundamental tool for circuit prototyping and experimentation. It enables users to assemble electronic components and establish connections without requiring soldering (Crowell, 2019).

Figure 11: Figure of Breadboard (Crowell, 2019).

- Buzzer

  Audible signaling devices, such as buzzers or beepers, can operate using electromechanical, piezoelectric, or purely mechanical mechanisms. Their primary purpose is to transform electrical audio signals into audible sound waves (Agarwal, 2021).

Figure 12: Figure of Buzzer (Agarwal, 2021)

- Summary Table

  The summary table shows that the sensors and actuators used in this IoT project are of which types of components and signal.

Components	**Roles**	**Types of Components**	**Signal Types**
RFID Module	Sensor	Active	Digital
RFID Tag	Identification	Passive	Digital
Buzzer	Actuator (Feedback)	Active	Digital
LCD Display	Actuator (Display)	Active	Digital

Table 1: Table of sensors and actuators.

2.3.2 Software Components

- Arduino IDE

  The Arduino IDE (Integrated Development Environment) serves as a comprehensive programming workspace featuring a specialized text editor for code composition, an output display panel, debugging terminal, and an interface with intuitive function buttons and dropdown menus. This software platform establishes direct communication with Arduino microcontroller boards to transfer compiled instructions and facilitate bidirectional data exchange (Arduino.cc, 2025).

- MS Word

  Microsoft Word stands as a dominant application in document creation, recognized for its user-friendly interface and versatile functionality. The software simplifies tasks ranging from basic correspondence to complex reports, offering comprehensive editing and layout features. Its intuitive design supports users in generating polished, organized materials for academic, professional, or personal purposes. With robust formatting options and collaborative capabilities, MS Word remains a preferred solution for efficient written communication across industries (GeeksforGeeks, 2016).

- Google Sheets

  Google Sheets is an online spreadsheet tool that allows individuals and teams to build, edit, and manage data-driven documents with live sharing capabilities. While it includes standard functions like organizing, adding, and removing data fields, its standout feature is real-time multi-user collaboration (Chai, 2021).

- C Programming

  C programming was developed in the early 190s by Dennis Ritchie. This is a foundational language that is also used for embedded programming, including microcontroller projects like Arduino. In Arduino development, simple versions of C/C++ are used, which later help to control the hardware components.

3. Development

3.1 Planning and Design

After receiving the assignment, our team quickly gathered for a group discussion to explore potential ideas. After weighing several options, we settled on building a Smart Card Attendance System, aiming to simplify and secure the attendance process in environments like schools, offices, or colleges. We chose the ESP32 as our main controller, as it could efficiently process inputs from smart cards and manage the system's operations. To streamline our workflow, we divided tasks based on each member's abilities and interests, which helped us stay focused and complete the project more effectively.

Throughout the project, we encountered a few difficulties, as expected in any collaborative effort. However, by maintaining regular communication and approaching problems with patience and teamwork, we were able to work through them successfully. When it came to designing the system, we prioritized ease of use, adaptability, and reliability to ensure it could be implemented in real-world settings. Every component we included had a distinct role in making the system functional. Additionally, we created a detailed circuit diagram that guided us through the building phase and helped keep our implementation process smooth and organized.

3.2 Resource Collection

While working on our Smart Card Attendance System, we gathered several key components including an RFID card, RFID module, LCD display, ESP32, and a buzzer. The ESP32, LCD display, and buzzer were available from the college, which allowed us to keep the costs lower. On the other hand, the RFID card and module were not provided, so we took the initiative to purchase them ourselves. Each part was chosen thoughtfully to ensure it would support the core functions of the system and work seamlessly with the rest of the setup.

To bring our system to life, we relied on a few essential software tools. We used Arduino IDE for writing the code and uploading it to the ESP32 board. For visualizing the circuit and creating system diagrams, we used cirkitdesigner and draw.io, which helped us to better document the project. To make sure the hardware and software components communicated properly, we installed necessary libraries like ESP32 (for board support) and I2C (to manage LCD communication). These libraries were added directly from the built-in manager within the Arduino IDE.

3.3 System Development

3.3.1 Phase 1: RFID Module connection with ESP32

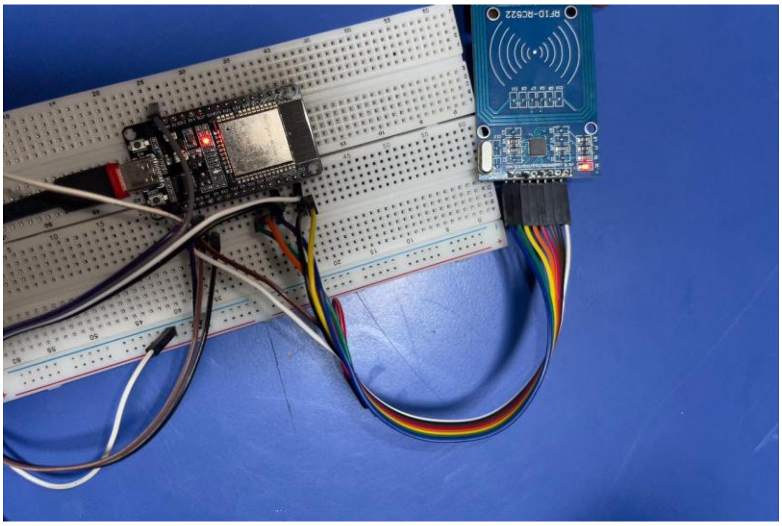

Figure 13: Figure shows the connection of ESP32 and RFID Module.

In the first phase of the IoT project, we connect the ESP32 with RFID Module in breadboard with the help of jumper wires as we can see in the figure 13 where the connection has been made and light is on.

The wire connections of ESP32 and RFID Module on breadboard are:

RFID Module	ESP32	Descriptions
GND	GND	Ground
3.3V	3.3V	Power Supply
RST	D22	Reset Pin
SDA	D21	SPI Slave Select
SCK	D18	SPI Clock
MOSI	D23	SPI Master Out Slave In
MISO	D19	SPI Master In Slave Out

Table 2: Table shows the connection of ESP32 and RFID Module.

3.3.2 Phase 2: LCD Display connection with ESP32

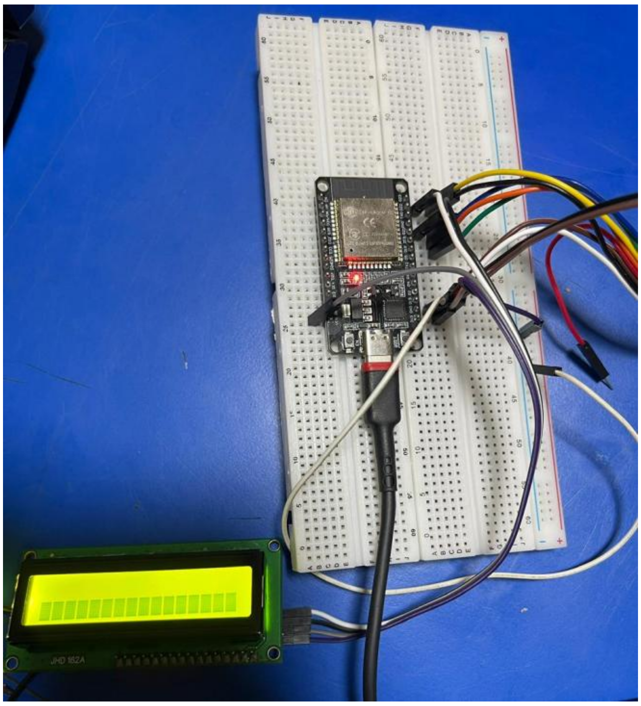

Figure 14: Figure shows the connection of LCD Display with ESP32.

In the second phase of the IoT project, we have connected the LCD screen with ESP32 with the help of jumper wires on breadboard as we can see in the figure 14 where the connection has been made and display is light on.

The wire connections of ESP32 and LCD Display on breadboard are:

LCD Display	ESP32	Descriptions
GND	GND	On breadboard of negative pin 36
VCC	VIN	Power Supply
SDA	D21	I2C Data Line
SCL	D22	I2C Clock Line
MOSI	D23	SPI Master Out Slave In
MISO	D19	SPI Master In Slave Out

Table 3: Table shows the connection of ESP32 and LCD Display.

3.3.3 Phase 3: Buzzer connection with ESP32

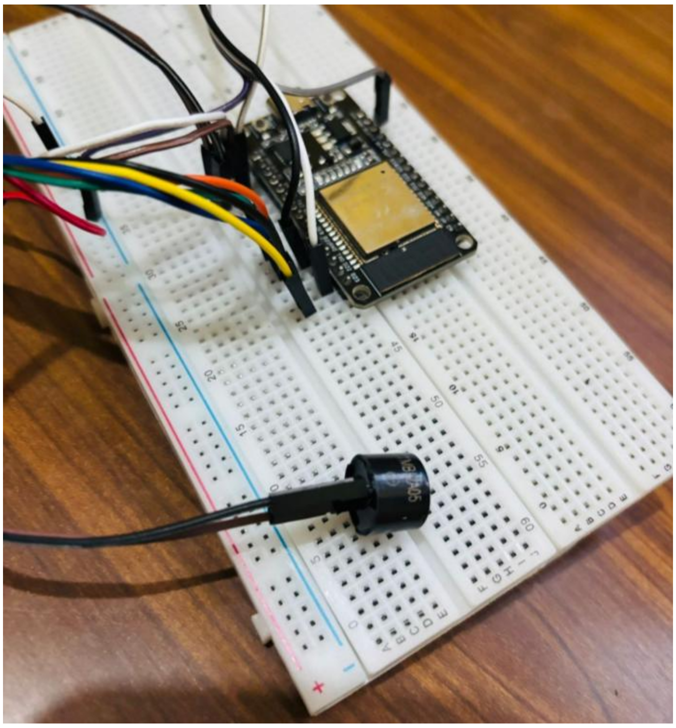

Figure 15: Figure shows the connection of Buzzer with ESP32.

In this phase 3 of the IoT project, we have connected the Buzzer with ESP32 with the help of jumper wires on breadboard as we can see in the figure where the connection has been made and produces beep sound.

The wire connections of ESP32 and Buzzer on breadboard are:

LCD Display	**ESP32**	**Descriptions**
GND	GND	On breadboard of negative pin 34
VCC	D15	Signal pin to turn buzzer ON/OFF
SDA	D21	I2C Data Line

Table 4: Table shows the connection of ESP32 and Buzzer.

4. Results and Findings

4.1 Results

At the end of the project, a fully functional RFID Attendance System was designed, which makes the recording process more efficient and accurate. A RFID module (RC522) is integrated with the Adrino Uno microcontroller which detects and reads data from the RFID tags and cards assigned to the individual users. When the tags or card is scanned the name and logs it along with the current date and time. This attendance data is stored in the centralized database Google Sheet, ensuring easy access for the future. The project also has a display unit which is 16x2 LCD where feedback like confirmation message is sent after successful scan. The buzzer also makes an alert after a successful scan.

After the project was completed, all the errors were manually checked. The code was compiled and we scanned the RFID card and tags and a value Screenshots of the hardware setup and set it up as Shristi. After the scan the buzzer made an alert sound. The serial monitors show 302 request authentication success, blog was read successfully, and this is redirected to a google sheet form **http://script.google.com/** along with successful name, time and date of the attendance.

4.2 Testing

4.2.1 Testing 1

Objective	To compile the code and upload it to ESP32 without any errors.
Action	• Write all the code in Arduino IDE. • Connect the ESP32 to the laptop. • Compile the code. • Upload the code to ESP32.
Expected Result	The code should be uploaded successfully without any errors.
Actual Result	The code was uploaded successfully without any errors.
Conclusion	The test was successful.

Table 5: Testing of compilation of code without any errors.

```
Writing at 0x00097299... (50 %)
Writing at 0x0009c9a7... (52 %)
Writing at 0x000a2375... (54 %)
Writing at 0x000a7f9c... (57 %)
Writing at 0x000ad507... (59 %)
Writing at 0x000b286f... (61 %)
Writing at 0x000b7a74... (64 %)
Writing at 0x000bd1c8... (66 %)
Writing at 0x000c2951... (69 %)
Writing at 0x000c7cdc... (71 %)
Writing at 0x000cd099... (73 %)
Writing at 0x000d2926... (76 %)
Writing at 0x000d88b9... (78 %)
Writing at 0x000de2b3... (80 %)
Writing at 0x000e5859... (83 %)
Writing at 0x000eee68... (85 %)
Writing at 0x000f4403... (88 %)
Writing at 0x000f94e1... (90 %)
Writing at 0x000ff071... (92 %)
Writing at 0x0010492b... (95 %)
Writing at 0x00109d88... (97 %)
Writing at 0x0010fbf3... (100 %)
Wrote 1054336 bytes (676094 compressed) at 0x00010000 in 13.0 seconds (effective 649.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

Figure 16: Figure shows the compilation of code.

4.2.2 Testing 2

Objective	To check whether the data is entered in the Google sheet or not.
Action	• Connect ESP32 to the laptop. • Compile the code. • Upload code to Arduino. • Take the authorized tag or card near the RFID module.
Expected Result	Registered data should be stored on Google sheet along with name, date and time.
Actual Result	Registered data was stored on Goggle sheet along with name, date and time.
Conclusion	The test was successful.

Table 6: Testing of entering data into Google sheet after scanning the card.

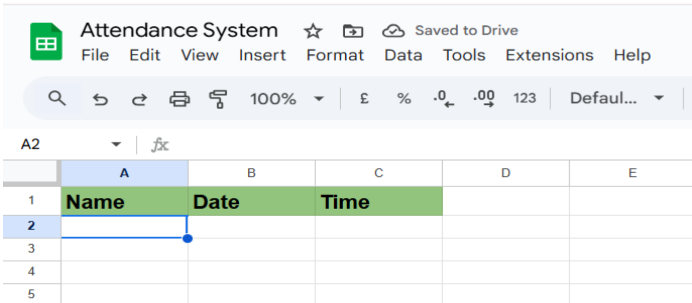

Figure 17: Figure shows the Google sheet before scanning the card.

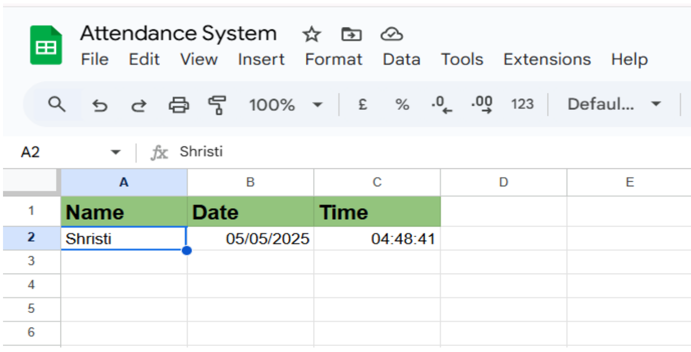

Figure 18: Figure shows the Google sheet after scanning the card.

4.2.3 Testing 3

Objective	To connect the Wi-Fi in ESP32 and display on serial point.
Action	• Connect Arduino to the laptop. • Compile the code. • Upload code to Arduino. • Open Serial Monitor.
Expected Result	Wi-Fi should be connected to ESP32 and should be displayed on serial monitor.
Actual Result	Wi-Fi was connected and displayed in serial monitor.
Conclusion	The test was successful.

Table 7: Testing of the connection of ESP32 with Wi-Fi.

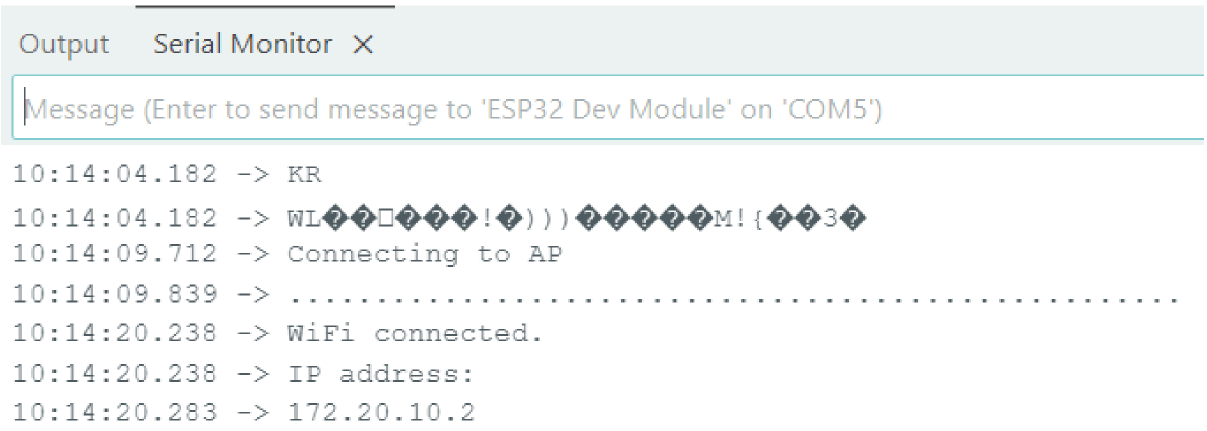

Figure 19: Figure shows that ESP32 is connected to Wi-Fi.

4.2.4 Testing 4

Objective	To check the credential and authentication of cardholders on serial monitor.
Action	• Connect Arduino to the laptop. • Compile the code. • Upload code to Arduino. • Take the authorized tag or card near the RFID module. • Open Serial Monitor.
Expected Result	The credential and authentication of cardholders should display on serial monitor i.e. “Authentication success”.
Actual Result	The credential and authentication of cardholders was displayed on serial monitor i.e. “Authentication success”.
Conclusion	The test was successful.

Table 8: Testing of RFID card credentials on serial monitor.

```
10:54:51.013 -> Reading last data from RFID...
10:54:51.058 -> Authentication success
10:54:51.078 -> Block was read successfully
10:54:51.110 -> Last data in RFID: Alex□□□□□□□□□□□□
10:54:51.888 -> https://script.google.com/macros/s/AKfycbx05CmVhb0si4hVZKxiLawAJr3Fas_iCqozcedKVCUb39GnQz6zoNZrMFE969qu9svD/exec?name=Alex
10:54:52.016 -> [HTTPS] begin...
10:54:52.016 -> [HTTPS] GET...
10:54:57.408 -> [HTTPS] GET... code: 302
```

Figure 20: Figure shows that cardholder credential displayed on serial monitor.

4.2.5 Testing 5

Objective	To check if the LCD works and displays the necessary message or not.
Action	• Connect Arduino to the laptop. • Compile the code. • Upload code to Arduino. • Take the authorized tag or card near the RFID module. • Check whether the necessary output in the LCD is or not.
Expected Result	The LCD display should display the message “Scan Your Card”.
Actual Result	The LCD display was not able to display the message “Scan Your Card”.
Conclusion	The test was unsuccessful.

Table 9: Testing of LCD Display.

Figure 21: Figure shows that LCD is not displaying the message.

4.2.6 Testing 6

Objective	To check if the LCD works and displays the necessary message or not.
Action	• Connect Arduino to the laptop. • Compile the code. • Upload code to Arduino. • Take the authorized tag or card near the RFID module. • Check whether the necessary output on the LCD is or is not.
Expected Result	The LCD display should display the message “Scan your card”, “Thank You!”, and “Data Recorded”.
Actual Result	The LCD display displayed all the messages.
Conclusion	The test was successful.

Table 10: Successful Testing of LCD display.

Figure 22: Figure shows that LCD is displaying “Scan Your Card”.

Figure 23: Figure shows that LCD is displaying “Thank You!”.

Figure 24: Figure shows that LCD is displaying “Data Recorded”.

5. Future Works

In this chapter, the future works that can be added later in the project which make it better, sustainable and usable are expected to be stated in points or paragraphs (with some overview of how can it be done can you also provide a list of future work or things that can be used to enhance the user experience and overall project quality.

- Facial Recognition Integration

  RFID can be combined with facial recognition technology, which helps with two-factor authentication and prevents proxy or fraudulent attendance entries. Tools like OpenCV and Python can be used for development.
- Mobile Application Integration

  A cross-platform mobile app (Android/iOS) can be created for students or employees for the attendance history with timely notifications and view summaries. This connection to mobile apps and use of cloud databases like Firebase, attendance data can be synced promptly with minimal effort.
- Mobile Notification (SMS/Email Alerts)

  New features like automatic alerts to the employees or parents when the respective person misses attendance. This can be done with SMS APIS or email services, the system can send alerts when the attendance is marked or when a user is absent.
- Dynamic web dashboard

  A dynamic web dashboard enhances the RFID attendance system by offering real-time tracking, visual reports, and user management. It simplifies data analysis with charts and sends alerts for discrepancies, making the system more efficient and user-friendly.

6. Conclusion

The RFID-driven Smart Attendance System, built around the ESP32 microcontroller, has emerged as a transformative tool for modernizing attendance tracking. This solution effectively displaces conventional manual approaches with an efficient, precise, and intuitive method. By harnessing RFID technology for identification and coupling the ESP32's functionality with Google Sheets integration, the system enables instantaneous data capture and encrypted cloud-based record-keeping. Interactive elements like the LCD screen, which offers real-time visual confirmations, and the buzzer's auditory cues elevate user engagement, ensuring seamless interaction with the platform.

Upgrading from the ESP8266 to the ESP32 architecture unlocked superior computational performance, expanded input/output adaptability, and native wireless connectivity, strengthening the system's reliability and scalability. This initiative not only met its technical objectives but also underscored the practical value of IoT frameworks in solving real-world logistical challenges.

Future iterations could expand functionality through biometric verification layers for enhanced security, push notifications for immediate status updates, and customizable dashboards for administrative oversight. Collectively, this project exemplifies how embedded systems can drive automation in educational and corporate environments, advancing the larger vision of digitized operational workflows.

7. References

Agarwal, T. (2021, July 26). *Buzzer : Working, Types, Circuit, Advantages & Disadvantages*. From ElProCus - Electronic Projects for Engineering Students: https://www.elprocus.com/buzzer-working-applications/

Arduino.cc. (2025). From Arduino.cc: https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics/

BYJUS. (2019, August 25). *Circuit Diagram*. From BYJUS: https://byjus.com/physics/circuit-diagram/

Chai, W. (2021). *Google Sheets*. From WhatIs: https://www.techtarget.com/whatis/definition/Google-Spreadsheets#:~:text=Google%20Sheets%20is%20a%20web,data%20online%20in%20real%20time.

Crowell, G. (2019, February 14). *What is a Breadboard?* From CircuitBread: https://www.circuitbread.com/ee-faq/what-is-a-breadboard

GeeksforGeeks. (2016, June 25). *Introduction to Microsoft Word*. From GeeksforGeeks: https://www.geeksforgeeks.org/introduction-to-microsoft-word/

GeeksforGeeks. (2024, July 26). *What is block diagram in software engineering?* From GeeksforGeeks: https://www.geeksforgeeks.org/what-is-block-diagram-in-software-engineering/#what-is-a-block-diagram

Hemmings, M. (2016). *What is a Jumper Wire?* From Sparkfuneducation.com: https://blog.sparkfuneducation.com/what-is-jumper-wire

Hongguang. (2023, August 30). *TFT vs LCD Display: which is better? | Hongguang Display*. From Hongguangdisplay.com: https://www.hongguangdisplay.com/blog/tft-vs-lcd-display-which-is-better/

Hübschmann, I. (2020, August 28). *ESP32 for IoT: A Complete Guide*. From Nabto: https://www.nabto.com/guide-to-iot-esp-32/

IBM. (2023, May 12). *Internet of Things*. From Ibm.com: https://www.ibm.com/think/topics/internet-of-things#:~:text=Growth%3A%20The%20number%20of%20IoT,new%20use%20cases%20and%20applications.

Oracle.com. (2024). *Accelerate Your Operations with IOT*. From Oracle.com: https://www.oracle.com/internet-of-things/

Robotix. (2020). *An introduction to a RFID*. From Technology Robotix Society, IIT Kharagpur: https://www.robotix.in/tutorial/auto/RFID/#:~:text=The%20RC522%20RFID%20Reader%20%20module,maximum%20data%20rate%20of%2010Mbps.

Sinha, S. (2024, September 03). *State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally*. From IoT Analytics: https://iot-analytics.com/number-connected-iot-devices/

Zheng. (2025, January 03). *Common RFID Hotel Card Issues & Solutions: What Hotel Operators Need to Know*. From RFID Card: https://www.rfidcard.com/what-is-rfid-technology/?srsltid=AfmBOoonHoKld_7uNe8FvUMG-pYak7f1XaKNeR-89IOviI8DMFfZKtIg

8. Appendix

8.1 Source Code

8.1.1 Source Code of RFID Tags

```
#include <SPI.h>

#include <MFRC522.h>

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

constexpr uint8_t RST_PIN = 22;

constexpr uint8_t SS_PIN = 21;

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

MFRC522 mfrc522(SS_PIN, RST_PIN);

MFRC522::MIFARE_Key key;

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

/* Set the block to write data */

int blockNum = 2;

/* This is the actual data which is

going to be written into the card */

byte blockData [16] = {"Shristi"};  // Change the name you want to store in RFID Tag

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

/* Create array to read data from Block */

/* Length of buffer should be 2 Bytes

more than the size of Block (16 Bytes) */

byte bufferLen = 18;

byte readBlockData[18];

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

MFRC522::StatusCode status;

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


void setup()

{

  //-----------------------------------------
```

```
  //Initialize serial communications with PC
  Serial.begin(9600);
  //----------------------------------------
  //Initialize SPI bus
  SPI.begin();
  //----------------------------------------
  //Initialize MFRC522 Module
  mfrc522.PCD_Init();
  Serial.println("Scan a RFID Tag to write data...");
  //----------------------------------------
}

/****************************************************************************************************
*****
 * Writ() function

****************************************************************************************************
****/
void loop()
{
  //---------------------------------------------------------------------------
  /* 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;
  }
  //---------------------------------------------------------------------------
  /* 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;}
//------------------------------------------------------------------------------
Serial.print("\n");
Serial.println("**Card Detected**");
/* Print UID of the Card */
Serial.print(F("Card UID:"));
for (byte i = 0; i < mfrc522.uid.size; i++){
  Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
  Serial.print(mfrc522.uid.uidByte[i], HEX);
}
Serial.print("\n");
/* Print type of card (for example, MIFARE 1K) */
Serial.print(F("PICC type: "));
MFRC522::PICC_Type piccType = mfrc522.PICC_GetType(mfrc522.uid.sak);
Serial.println(mfrc522.PICC_GetTypeName(piccType));
//------------------------------------------------------------------------------
 /* Call 'WriteDataToBlock' function, which will write data to the block */
 Serial.print("\n");
 Serial.println("Writing to Data Block...");
 WriteDataToBlock(blockNum, blockData);
 //------------------------------------------------------------------------------
 /* Read data from the same block */
 Serial.print("\n");
 Serial.println("Reading from Data Block...");
 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.print("\n");
  Serial.print("Data in Block:");
  Serial.print(blockNum);
  Serial.print(" --> ");
  for (int j=0 ; j<16 ; j++){
    Serial.write(readBlockData[j]);
  }
  Serial.print("\n");
  //---------------------------------------------------------------------------
}

/****************************************************************************************************
*****
 * Writ() function
****************************************************************************************************
****/
void WriteDataToBlock(int blockNum, byte blockData[])
{
  //---------------------------------------------------------------------------
  /* Authenticating the desired data block for write 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 Write: ");
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  //---------------------------------------------------------------------------
  else {
    Serial.println("Authentication success");
  }
```

```
  //---------------------------------------------------------------------------
  /* Write data to the block */
  status = mfrc522.MIFARE_Write(blockNum, blockData, 16);
  if (status != MFRC522::STATUS_OK) {
    Serial.print("Writing to Block failed: ");
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  else
  {Serial.println("Data was written into Block successfully");}
  //---------------------------------------------------------------------------
}

/****************************************************************************************************
*****
 * ReadDataFromBlock() function
****************************************************************************************************
****/
void ReadDataFromBlock(int blockNum, byte readBlockData[])
{
  //---------------------------------------------------------------------------
  /* 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 {
    Serial.println("Block was read successfully");
  }
  //---------------------------------------------------------------------------
}
```

8.1.2 Source Code of RFID Attendance

```
// Smart Attendance System with Google Sheets and LCD Display (ESP32 Version)

#include <SPI.h>
#include <MFRC522.h>
#include <Arduino.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <WiFiClientSecure.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

//----------------------------------------
// Pin configuration for ESP32
```

```
#define RST_PIN   22  // Change to your ESP32 wiring
#define SS_PIN    21
#define BUZZER    15
//-----------------------------------------

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

int blockNum = 2;
byte bufferLen = 18;
byte readBlockData[18];

String card_holder_name;
const String sheet_url =
"https://script.google.com/macros/s/AKfycbxeNwXpzLaorddxQv5X3hTHfopsq86r5j9b
KSc4hszdIFTjkp3pKHaebo0bFr9CKE9x/exec?name=";  // Replace with your actual
Google Apps Script URL

//#define WIFI_SSID "janakbasnet01"
//#define WIFI_PASSWORD "CLB43515D6"

#define WIFI_SSID "Islington College"
#define WIFI_PASSWORD "I$LiNGT0N2025"

// LCD setup (try 0x27 if 0x3F doesn't work)
LiquidCrystal_I2C lcd(0x27, 16, 2);

/*******************************************************************************************
*****
 * setup() function
```

```
*********************************************************************************************
****/
void setup()
{

  Serial.begin(9600);
  lcd.init();
  lcd.begin(16, 2);  // <-- FIXED: Provide cols and rows
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("  Initializing  ");
  for (int a = 5; a <= 10; a++) {
    lcd.setCursor(a, 1);
    lcd.print(".");
    delay(500);
  }

  Serial.println("\nConnecting to AP");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(200);
  }

  Serial.println("\nWiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  pinMode(BUZZER, OUTPUT);
```

```
  SPI.begin();  // Defaults: SCK=18, MISO=19, MOSI=23
}


/*******************************************************************************************
*****
 * loop() function
*******************************************************************************************
****/
void loop()
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(" Scan your Card ");



  mfrc522.PCD_Init();
  if (!mfrc522.PICC_IsNewCardPresent()) return;
  if (!mfrc522.PICC_ReadCardSerial()) return;

  Serial.println("\nReading last data from RFID...");
  ReadDataFromBlock(blockNum, readBlockData);

  Serial.print("Last data in RFID: ");
  for (int j = 0; j < 16; j++) {
    Serial.write(readBlockData[j]);
```

```
}
lcd.init();
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Thank You!");


Serial.println();

// Buzzer signal
for (int i = 0; i < 2; i++) {
  digitalWrite(BUZZER, HIGH);
  delay(200);
  digitalWrite(BUZZER, LOW);
  delay(200);
}

if (WiFi.status() == WL_CONNECTED) {
  WiFiClientSecure client;
  client.setInsecure();  // Disable SSL check (use only for testing)

  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, card_holder_name)) {
    Serial.print(F("[HTTPS] GET...\n"));
```

```
      int httpCode = https.GET();

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

        lcd.clear();
        lcd.setCursor(0, 1);
        lcd.print(" Data Recorded ");
        delay(500);
      } 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");
    }
  }

  mfrc522.PICC_HaltA();
  mfrc522.PCD_StopCrypto1();
  delay(1500);
}

/*********************************************************************************************
*****
 * ReadDataFromBlock() function
```

```
*********************************************************************************************
****/
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");
  }

  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");
  }
}
```

8.1.3 Source Code of Google Sheet

```
var ss = SpreadsheetApp.openById('1xMzbNNGem3G2DtqTuYL-
HiGu0aUUtWijh0ROvQbaBl8');

var sheet = ss.getSheetByName('Sheet1');
```

```
var timezone = "Asia/Kathmandu";


function doGet(e){
  Logger.log( JSON.stringify(e) );
  if (e.parameter == 'undefined') {
    return ContentService.createTextOutput("Received data is undefined");
  }
  var Curr_Date = new Date();
  var Curr_Time = Utilities.formatDate(Curr_Date, timezone, 'HH:mm:ss');
  var name = stripQuotes(e.parameters.name);
  var nextRow = sheet.getLastRow() + 1;
  sheet.getRange("A" + nextRow).setValue(name);
  sheet.getRange("B" + nextRow).setValue(Curr_Date);
  sheet.getRange("C" + nextRow).setValue(Curr_Time);

  return ContentService.createTextOutput("Card holder name is stored in column A");
}


function stripQuotes( value ) {
  return value.toString().replace(/^["']|['"]$/g, "");
}

function doPost(e) {
  var val = e.parameter.value;

  if (e.parameter.value !== undefined){
    var range = sheet.getRange('A2');
    range.setValue(val);
```

```
  }
}
```

8.2 Work Distribution Plan

We have a project of five people to do this project.

Student Name	Role	Contribution
Hirdesh Chauhan	• Code the program and execution. • Documentation of the project.	20%
Nikita Bhandari	• Debug and review the code. • Documentation of the project.	20%
Anjan Sainju	• Making presentation slides. • Hardware connections.	20%
Samir Kumar Singh	• Hardware Connections • Documentation of the projects.	20%
Abhishek Shah	• Making presentation slides. • Hardware connections.	20%