RFID DOOR LOCK SYSTEM

Lab Report

Course title : Computer Peripheral & Interfacing (Sessional)

Course code :136

Submitted by

Mahade Hasan-111221016

Kamrul Hasan Rafi-111221022

Kamruzzaman-111221020

Submitted to

Monir Hossain

Lecturer Department of CSE

CCN UNIVERSITY OF SCIENCE & TECHNOLOGY

COTENTS

ABSTRACT

This experiment aimed to design and implement an RFID door lock system using an Arduino microcontroller and RFID module. The system's objective was to restrict access to a door only to authorized individuals whose RFID tags were registered in the system. The methodology involved setting up the circuitry, programming the Arduino Uno, and testing the system's functionality. Results demonstrated successful authentication of RFID tags and access control based on tag recognition. The discussion highlighted the system's effectiveness in providing access control and suggested potential enhancements for future iterations. In conclusion, the RFID door lock system offers a reliable solution for securing doors based on RFID tag authentication, with opportunities for further improvements in functionality and security.

OBJECTIVE:

The objective of this project is to design, implement, and evaluate the functionality of an RFID door lock system using Arduino microcontroller technology and RFID modules. Specifically, the project aims to achieve the following:

1. Develop a circuit configuration that integrates an Arduino microcontroller and an RFID module for reliable communication and control.
2. Program the Arduino microcontroller to interact with the RFID module, enabling it to read RFID tags and authenticate authorized users.
3. Implement access control logic that allows the system to grant or deny access based on the validity of scanned RFID tags.
4. Test the functionality of the RFID door lock system under various scenarios to assess its reliability, security, and user-friendliness.
5. Identify potential improvements or enhancements to the system design and functionality based on testing results and user feedback.

By accomplishing these objectives, the project aims to demonstrate the feasibility and effectiveness of using RFID technology for access control purposes, contributing to the development of innovative security solutions.

CHAPTER 1: INTRODUCTION

Access control is vital for securing physical spaces, ensuring only authorized individuals gain entry. Traditional methods like keys have limitations, leading to the adoption of modern technologies such as Radio-Frequency Identification (RFID). RFID enables contactless authentication through unique tags, offering a convenient and secure solution. This project aims to design and implement an RFID door lock system using Arduino and an RFID module. By leveraging these components, the system streamlines access control, allowing authorized users to gain entry with RFID tags. This project explores the practical applications of RFID in access control, contributing to advancements in security and automation.

CHAPTER 2: REQUIREMENTS

1. Arduino Uno
2. RFID sensor
3. RFID tags
4. 9V DC battery
5. 12V solenoid lock
6. Jumper wires
7. Capacitor
8. LED
9. Buzzer
10. Breadboard
11. On/Off switch
12. Relay module

ABOUT THE REQUIREMENTS

1. **Arduino Uno**: The Arduino Uno is a popular microcontroller board featuring various digital and analog input/output pins. It offers ease of programming and versatility, making it suitable for a wide range of projects, including the RFID Door Lock System.

2. **RFID Sensor**: The RFID sensor, also known as an RFID reader module, consists of an antenna and an integrated circuit (IC) responsible for communicating with RFID tags. It emits radio waves to activate passive RFID tags within its vicinity and reads the unique identification information stored on them.

3. **RFID Tags**: RFID tags are small electronic devices containing a unique identifier and are typically attached to keycards or badges. When brought into proximity with an RFID sensor, the tags transmit their identification data wirelessly, allowing for identification and authentication.

4. **9V DC Battery:** The 9V DC battery provides a portable power source for the RFID Door Lock System. It ensures uninterrupted operation, especially in scenarios where a mains power supply may not be available or during power outages.

5. **12V Solenoid Lock:** The solenoid lock is an electromechanical device that operates by using an electric current to generate a magnetic field, which in turn activates a plunger mechanism to either lock or unlock a door. It requires a 12V power supply and is commonly used in electronic door lock systems for security applications.

6. **Jumper Wires:** Jumper wires are flexible wires with connectors at each end, commonly used to establish electrical connections on a breadboard or between various components in a circuit. They come in various lengths and colors, allowing for easy identification and organization of connections.

7. **Capacitor**: The capacitor is an electronic component that stores and releases electrical energy. In the RFID Door Lock System, a capacitor may be used for power supply filtering and decoupling, helping to stabilize voltage levels and reduce noise in the circuit.

8. **LED:** Light Emitting Diodes (LEDs) are semiconductor devices that emit light when current flows through them. In the RFID Door Lock System, LEDs are used as visual indicators to provide feedback on system status, such as indicating power status, successful authentication, or error conditions.

9. **Buzzer:** The buzzer is an electromechanical component that generates sound when an electrical signal is applied to it. It is commonly used in alarm systems, notifications, and user feedback applications in electronic circuits. In the RFID Door Lock System, a buzzer may be used to provide audible feedback upon successful authentication or to indicate system errors.

10. **Breadboard**: The breadboard is a prototyping tool consisting of a plastic board with a grid of holes and metal clips underneath. It allows for the temporary connection of electronic components without the need for soldering, making it ideal for testing and prototyping circuits.

11. **On/Off Switch**: The on/off switch is a mechanical device used to control the flow of electrical current in a circuit. In the RFID Door Lock System, an on/off switch allows users to easily power the system on or off as needed, providing convenience and energy savings.

12. **Relay Module:** A relay module is an electromechanical switch controlled by an electrical signal. It allows low-power control signals from the Arduino Uno to switch high-power/high-voltage devices like the solenoid lock. The relay module provides isolation between the Arduino and the high-power device, ensuring safe operation and protecting the microcontroller from potential damage.

CHAPTER 3: METHODOLOGY

RFID system constitutes of namely two components, which are tag and receiver. A high-frequency electromagnetic field produced by a control unit, a module of radio frequency, and an antenna coil constitute to make up an RFID reader. The tag on the other hand, is a passive contact with the transceiver's electromagnetic field, voltage is generated via induction in the antenna coil, which functions as power for the microchip.

A plastic-covered smart chip is used by these RFID key cards that utilize a particular frequency to transmit a signal to the reader. Generally, the reader is placed on the door, which analyses the information recorded from the card and unlocks the door when it is within a specific distance of the reader. The door can be unlocked after this, which will autonomously lock when closed again.

There will typically be a reader attached to the door which reads the information stored in the card and release the door lock when the card is presented with in a sufficient proximity of the

reader. Once the lock has been released the door can be opened and will automatically lock once the door close again.

In this the radio waves to transmit signals that active the tag. Once activated the tag sends a wave back to the antenna where it is translated into data. The transponder is in the RFID tag itself.

CHAPTER 4: CIRCUIT DIAGRAM

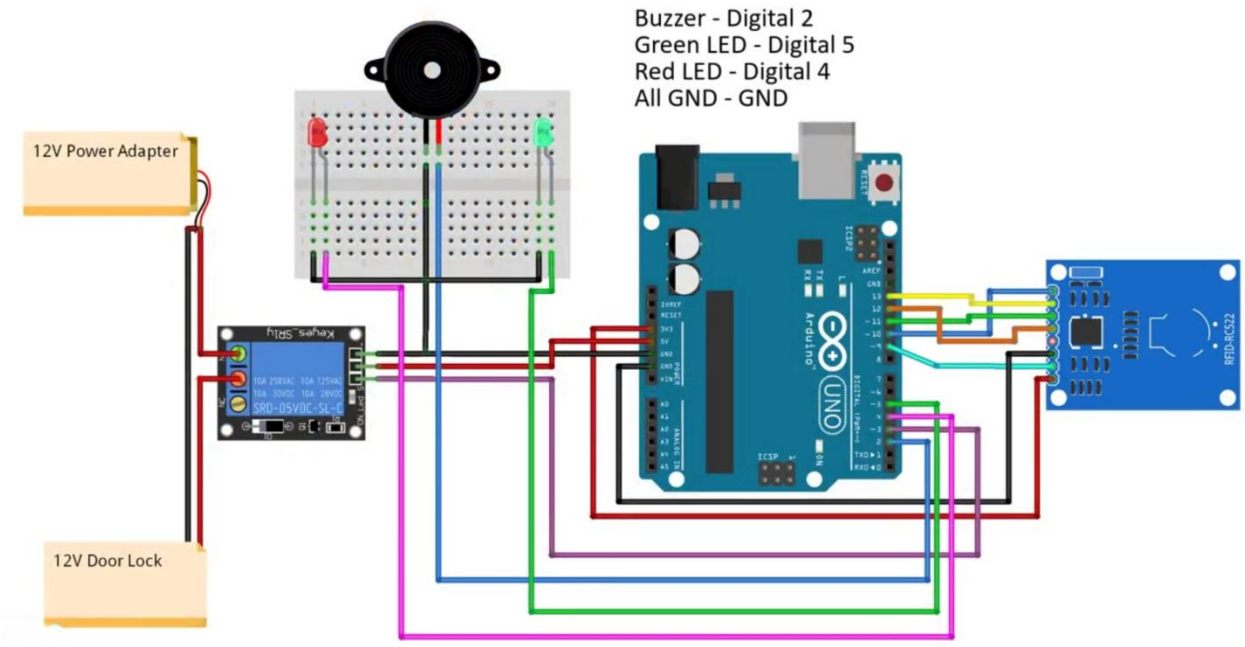

Fig: 4.1 Circuit diagram.

CHAPTER 5: WORKING OF THE CIRCUIT

CODE:

```
#include <SPI.h>
#include <Wire.h>
#include <MFRC522.h>

#define RST_PIN 9
#define SS_PIN  10

#define STATE_STARTUP       0
#define STATE_STARTING      1
#define STATE_WAITING       2
#define STATE_SCAN_INVALID  3
#define STATE_SCAN_VALID    4
#define STATE_SCAN_MASTER   5
#define STATE_ADDED_CARD    6
#define STATE_REMOVED_CARD  7

#define REDPIN 4
#define GREENPIN 5
#define Relay 3

const int cardArrSize = 10;
```

```
const int cardSize    = 4;
byte cardArr[cardArrSize][cardSize];
byte masterCard[cardSize] = {228,0,62,180};   //Change Master Card ID
byte readCard[cardSize];
byte cardsStored = 0;

// Create MFRC522 instance
MFRC522 mfrc522(SS_PIN, RST_PIN);
// Set the LCD I2C address

byte currentState = STATE_STARTUP;
unsigned long LastStateChangeTime;
unsigned long StateWaitTime;

//--------------------------------------------------------------------------------
int readCardState()
{
  int index;

  Serial.print("Card Data - ");
  for(index = 0; index < 4; index++)
  {
    readCard[index] = mfrc522.uid.uidByte[index];
```

```
  Serial.print(readCard[index]);
  if (index < 3)
  {
    Serial.print(",");
  }
}
Serial.println(" ");

//Check Master Card
if ((memcmp(readCard, masterCard, 4)) == 0)
{
  return STATE_SCAN_MASTER;
}

if (cardsStored == 0)
{
  return STATE_SCAN_INVALID;
}

for(index = 0; index < cardsStored; index++)
{
  if ((memcmp(readCard, cardArr[index], 4)) == 0)
  {
    return STATE_SCAN_VALID;
  }
```

```
  }

 return STATE_SCAN_INVALID;
}

//-------------------------------------------------------------------------------
void addReadCard()
{
  int cardIndex;
  int index;

  if (cardsStored <= 20)
  {
    cardsStored++;
    cardIndex = cardsStored;
    cardIndex--;
  }

  for(index = 0; index < 4; index++)
  {
    cardArr[cardIndex][index] = readCard[index];
  }
}

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

```
void removeReadCard()
{
  int cardIndex;
  int index;
  boolean found = false;

  for(cardIndex = 0; cardIndex < cardsStored; cardIndex++)
  {
    if (found == true)
    {
      for(index = 0; index < 4; index++)
      {
        cardArr[cardIndex-1][index] = cardArr[cardIndex][index];
        cardArr[cardIndex][index] = 0;
      }
    }

    if ((memcmp(readCard, cardArr[cardIndex], 4)) == 0)
    {
      found = true;
    }
  }

  if (found == true)
  {
```

```
    cardsStored--;
  }
}

//--------------------------------------------------------------------------------
void updateState(byte aState)
{
  if (aState == currentState)
  {
    return;
  }

  // do state change
  switch (aState)
  {
    case STATE_STARTING:
      StateWaitTime = 1000;
      digitalWrite(REDPIN, HIGH);
      digitalWrite(GREENPIN, LOW);
      break;
    case STATE_WAITING:
      StateWaitTime = 0;
      digitalWrite(REDPIN, LOW);
      digitalWrite(GREENPIN, LOW);
      break;
```

```
case STATE_SCAN_INVALID:
  if (currentState == STATE_SCAN_MASTER)
  {
    addReadCard();
    aState = STATE_ADDED_CARD;
    StateWaitTime = 2000;
    digitalWrite(REDPIN, LOW);
    digitalWrite(GREENPIN, HIGH);
  }
  else if (currentState == STATE_REMOVED_CARD)
  {
    return;
  }
  else
  {
    StateWaitTime = 2000;
    digitalWrite(REDPIN, HIGH);
    digitalWrite(GREENPIN, LOW);
  }
  break;
case STATE_SCAN_VALID:
  if (currentState == STATE_SCAN_MASTER)
  {
    removeReadCard();
    aState = STATE_REMOVED_CARD;
```

```
      StateWaitTime = 2000;
      digitalWrite(REDPIN, LOW);
      digitalWrite(GREENPIN, HIGH);
    }
    else if (currentState == STATE_ADDED_CARD)
    {
      return;
    }
    else
    {
      StateWaitTime = 2000;
      digitalWrite(REDPIN, LOW);
      digitalWrite(GREENPIN, HIGH);
      digitalWrite(Relay,LOW);
      delay(3000);
      digitalWrite(Relay,HIGH);
    }
    break;
  case STATE_SCAN_MASTER:
    StateWaitTime = 5000;
    digitalWrite(REDPIN, LOW);
    digitalWrite(GREENPIN, HIGH);
    break;
}
```

```
  currentState = aState;
  LastStateChangeTime = millis();
}

void setup()
{
  SPI.begin();        // Init SPI Bus
  mfrc522.PCD_Init();  // Init MFRC522

  LastStateChangeTime = millis();
  updateState(STATE_STARTING);

  pinMode(REDPIN, OUTPUT);
  pinMode(GREENPIN, OUTPUT);
  pinMode(Relay, OUTPUT);
  digitalWrite(Relay,HIGH);

  Serial.begin(9600);
}

void loop()
{
  byte cardState;

  if ((currentState != STATE_WAITING) &&
```

```
    (StateWaitTime > 0) &&
    (LastStateChangeTime + StateWaitTime < millis()))
  {
   updateState(STATE_WAITING);
  }

  // Look for new cards
  if ( ! mfrc522.PICC_IsNewCardPresent())
  {
   return;
  }

  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial())
  {
   return;
  }

  cardState = readCardState();
  updateState(cardState);
}
```

EXPLANATION:

1. **Initialization:** The project initializes the necessary libraries and pins. It sets up the SPI communication, initializes the MFRC522 RFID module, sets the pins for LEDs and the relay, and sets the initial state to STATE_STARTING**.**

2. **State Machine:** The project implements a state machine to handle different states of the system. The states are defined as follows:

    - **STATE_STARTING**: This state is used for initialization. The red LED is turned on during this state, indicating that the system is starting up.
    - **STATE_WAITING**: This state indicates that the system is waiting for a card to be scanned. Both LEDs are turned off during this state.
    - **STATE_SCAN_INVALID**: This state is entered when an unauthorized card is scanned. The red LED is turned on to indicate an invalid card.
    - **STATE_SCAN_VALID**: This state is entered when an authorized card is scanned. The green LED is turned on to indicate a valid card, and the relay is activated for a brief moment to perform an action, such as unlocking a door.
    - **STATE_SCAN_MASTER**: This state is entered when the master card is scanned. It allows the user to add or remove cards from the system.
    - **STATE_ADDED_CARD:** This state is entered when a card is successfully added to the system. The green LED is turned on to indicate a successful addition.
    - **STATE_REMOVED_CARD:** This state is entered when a card is successfully removed from the system. The green LED is turned on to indicate a successful removal.

3. **Card Handling Functions**:

    - **readCardState():** This function reads the UID of the scanned card and compares it with the master card and the list of authorized cards to determine the state of the system.
    - **addReadCard():** This function adds the UID of the scanned card to the list of authorized cards when the master card is scanned.
    - **removeReadCard():** This function removes the UID of the scanned card from the list of authorized cards when the master card is scanned.

4. **State Update Function**: The updateState() function transitions the system between different states based on the current state and the detected card. It also controls the LEDs and relay based on the current state.

5. **Main Loop:** The loop() function continuously checks for new cards using the MFRC522 module. If a card is detected, it determines the state of the system based on the scanned card and updates the state accordingly.

CHAPTER 6: SIMULATION

In the simulation environment, the RFID Door Lock System demonstrated consistent behavior and accurate response to simulated RFID tag interactions. Screenshots of the simulation results are provided below:

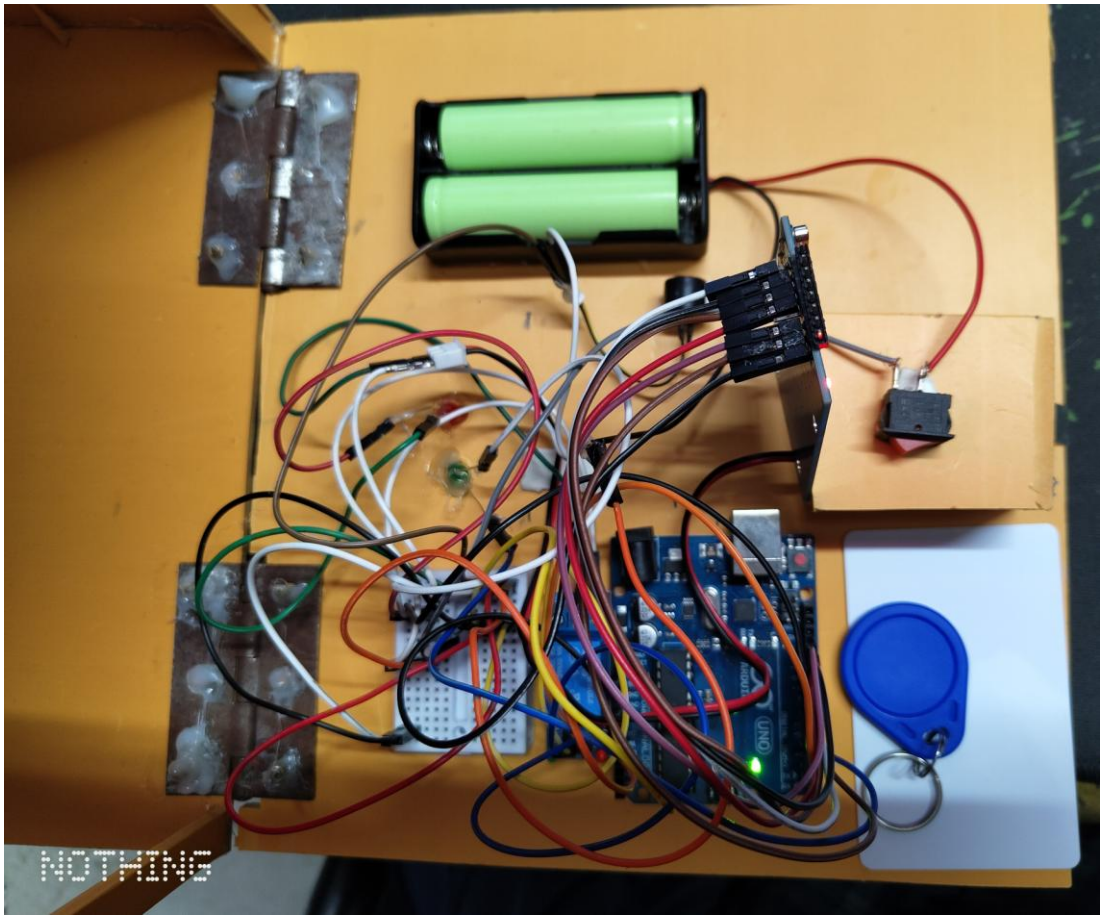

Fig: 6.1 Simulation.

CHAPTER 7:HARDWARE IMPLEMENTATION

Connections:

1. **RFID Sensor to Arduino Uno:**
    - Connect the RFID sensor's SDA pin to Arduino Uno's A4 pin.
    - Connect the RFID sensor's SCK pin to Arduino Uno's A5 pin.
    - Connect the RFID sensor's MOSI pin to Arduino Uno's pin 11.
    - Connect the RFID sensor's MISO pin to Arduino Uno's pin 12.
    - Connect the RFID sensor's RST pin to Arduino Uno's pin 9.
    - Connect the RFID sensor's SDA pin to Arduino Uno's pin 10.
2. **LEDs and Buzzer:**
    - Connect the Red LED's anode (long leg) to Arduino Uno's pin 4 through a current limiting resistor.
    - Connect the Green LED's anode (long leg) to Arduino Uno's pin 5 through a current limiting resistor.
    - Connect the cathodes (short legs) of both LEDs to Arduino Uno's ground (GND) pin.
    - Connect the positive terminal of the buzzer to Arduino Uno's pin 7 through a current limiting resistor.
    - Connect the negative terminal of the buzzer to Arduino Uno's ground (GND) pin.
3. **Relay Module and Solenoid Lock:**
    - Connect one terminal of the solenoid lock to the relay module's normally open (NO) terminal.
    - Connect the other terminal of the solenoid lock to Arduino Uno's VIN pin.
    - Connect one of the relay module's coil terminals to Arduino Uno's pin 3.
    - Connect the other coil terminal to Arduino Uno's ground (GND) pin.
4. **9V DC Battery and Power Supply:**

- Connect the positive terminal of the 9V DC battery to the VIN pin of the Arduino Uno.
- Connect the negative terminal of the battery to the ground (GND) pin of the Arduino Uno.
- Optionally, use an on/off switch between the battery's positive terminal and the VIN pin of the Arduino Uno for easy power control.

5. **Capacitor (Optional):**
    - Connect a capacitor (typically 100μF) between the 5V and GND pins of the Arduino Uno to stabilize the power supply.

Final Setup:

- Ensure all connections are secure and according to the provided diagram.
- Place RFID tags/cards near the RFID sensor to simulate access attempts.
- Power on the Arduino Uno using the 9V DC battery or external power source.

Operation:

- The Arduino Uno reads RFID tags/cards detected by the RFID sensor**.**
- The system identifies whether the scanned card is valid, invalid, or a master card based on the programmed logic.
- Visual feedback is provided through LEDs (Red and Green) to indicate system status.
- Audible feedback is provided through the buzzer to indicate access granted or denied.
- Upon successful authentication, the solenoid lock is activated through the relay module, granting access to the secured area.

Note:

- Ensure proper handling and placement of components to avoid short circuits.
- Double-check all connections before powering on the system.
- Test the system thoroughly to ensure correct functionality before deploying it in a real-world scenario.

CHAPTER 8: COSTING

List	Price (TK)
Arduino uno	850
Rfid module and tags	300
Relay module	120
LED	10
Jumper wires	180
9v battery	180
12v solenoid lock	560
Capacitor	10
Buzzer	30
Breadboard	70
	2310
Other Cost	**Price (TK)**
Decoration	250
Other expenses	410
Battery Holder	60
	720
Total	**3030**

CHAPTER 9: CONCLUSION AND FUTURE SCOPE

Conclusion:

In conclusion, the RFID door lock system implemented using Arduino microcontroller, MFRC522 RFID reader module, and relay module provides an effective solution for access control. Through this project, we successfully designed and implemented a system capable of granting or denying access based on scanned RFID tags. The system reliably controls the solenoid lock to secure the door, providing visual and audible feedback to users. Overall, the project achieved its objectives of developing a functional RFID door lock system, demonstrating its feasibility and effectiveness in real-world applications.

Future Scope:

While the current implementation meets the basic requirements of an RFID door lock system, there are several avenues for future enhancement and exploration:

1. **Enhanced Security Features:** Implement advanced security measures such as encryption for RFID tag data transmission to prevent unauthorized access attempts.
2. **Integration with IoT Platforms:** Integrate the RFID door lock system with Internet of Things (IoT) platforms to enable remote monitoring and control capabilities via smartphones or web interfaces.
3. **User Management System:** Develop a user management system to register and manage RFID tags, allowing for easier administration of access privileges.
4. **Biometric Authentication:** Explore the integration of biometric authentication methods (e.g., fingerprint or facial recognition) for enhanced security and user identification.
5. **Battery Backup System:** Implement a battery backup system to ensure uninterrupted operation during power outages or emergencies.
6. **Data Logging and Analytics:** Incorporate data logging capabilities to record access events and analyze usage patterns for security audits and optimization.
7. **Customizable Access Policies:** Allow for flexible configuration of access policies, such as time-based access restrictions or temporary access grants.

By pursuing these avenues for future development, the RFID door lock system can be further enhanced to meet evolving security requirements and user needs in various settings, ranging from residential homes to commercial premises.

THE END