

CONTENTS

ABSTRACT.....	v
TABLE OF FIGURES.....	vi
CHAPTER 1.....	1
1.1 INTRODUCTION 1.2 OBJECTIVE	
CHAPTER 2.....	4
2.1 LITERATURE SURVEY 2.2 PROPOSED IDEA 2.3	
MERITS AND DEMERITS 2.4 SUMMARY	
CHAPTER 3.....	8
3.1 METHODOLOGY 3.2 HARDWARE AND	
SOFTWARE 3.3 BLOCK DIAGRAM 3.4 PROGRAM	
CHAPTER 4.....	23
4.1 CIRCUIT CONNECTIONS 4.2 TESTING AND	
RESULTS 4.3 CONCLUSION 4.4 REFERENCES	

ABSTRACT

In an effort to enhance vehicle security and prevent unauthorized access, we present the development of an antitheft vehicle system utilizing a fingerprint sensor integrated with an Arduino microcontroller. The primary objective of this project is to replace traditional key-based entry methods with a more secure and user-friendly biometric authentication system.

The system comprises a fingerprint sensor that scans and verifies the user's fingerprint, an Arduino microcontroller for processing and controlling the inputs and outputs, and an immobilizer circuit to prevent engine start-up in case of unauthorized access attempts. Upon successful fingerprint recognition, the Arduino signals the immobilizer circuit to permit engine ignition, ensuring that only registered users can operate the vehicle. This innovative approach not only heightens security but also simplifies access management for vehicle owners.

The project explores the design, implementation, and testing of the system, highlighting its effectiveness in real-world scenarios. The results demonstrate a significant reduction in the likelihood of vehicle theft, offering a promising solution for modern vehicle security challenges.

TABLE OF FIGURES

SNO	FIGURE NAME	PGNO
1.	FLOW CHART	9
2.	ARDUINO	10
3.	FINGERPRINT SENSOR	11
4.	BLOCK DIAGRAM	14
5.	CIRCUIT CONNECTIONS	23
6.	SENSOR INITILISATION	23
7.	FINGERPRINT ENROLLMENT	23
8.	FINGERPRINT SCANNER	24
9.	DELETE FINGERPRINT	24
10.	DISPLAYING CONTENT	24
11.	FINGERPRINT NOT MATCHED	24
12.	MESSAGE FROM GSM	24

CHAPTER -1 1.1 INTRODUCTION

In this report, we propose a biometric vehicle ignition vehicle system. Here we introduced fingerprint module and GSM module for the secure purpose. User can first register by using fingerprint module. Then user just needs to scan finger to start vehicle without any key, so by user no need to carry any keys. The system only allows authorized user to start the vehicle. The system can register many users. When in the scanning mode, the system checks if user is authorized or not. If the user is unauthorized the system sends notification to owner of vehicle. The notification is send by the GSM technology which we introduced in this system. In this system fingerprint module and GSM module are both connected to Arduino. The Arduino board is connected to computer and then uploads code to proper work of the system.

In India, one of the most pressing problems is automobile security. One of the primary difficulties confronting emerging countries is auto theft protection. To preserve and secure the autos, several strategies have been attempted and tested. Embedded computing is a new technology that is being utilised to improve and enhance vehicle security against theft. RFID cards are used to start a car using radiofrequency identification (RFID).

The approach, however, failed due to the possibility of losing or stealing the card. Fingerprint recognition is an excellent biometric technique since it allows for precise digital identification. This technology has been widely used to protect automobiles and identity fraud due to its high-speed matching algorithms and quick integration. The proposed system gives a brief explanation about the biometric security system for controlling of the vehicle theft. With the daily increase of Internet of Things (IoT) devices, which have reached tens of billions these days. The use of IoT is growing exponentially and its use can be seen across various

fields. There has been a lot of theft of vehicles faced by the automobile owners. Many control methods exist in order to reduce these thefts, but they lack the efficient methodology which includes monitoring and tracking of the automobile.

So, designing a system which ensures a high security of the vehicle is a priority. Ultimately the lacking of these essential requirements in the present system has led to the idea of designing of a system which ensures the fulfillment of parameters in order to increase the accuracy and effectiveness of the system. Considering another fact of increased number in the problem of losing vehicle keys frequently by the owners, has led to the emergence of the idea of biometric fingerprint scanning to start the vehicle.

1.2 OBJECTIVE

1.Enhance Vehicle Security:Develop a secure system that prevents unauthorized access and theft by utilizing biometric fingerprint authentication.

2.User-Friendly Interface:Design an intuitive and easy-to-use interface for vehicle owners to register and manage their fingerprints.

3.Integration with Existing Systems: Ensure seamless integration of the fingerprintsensor and Arduino with the vehicle's existing ignition and immobilizer systems.

4.Reliable Authentication:Achieve a high level of accuracy in fingerprint recognition to minimize false acceptances and rejections.

5. Robust Performance: Test the system under various conditions to ensure consistent and reliable performance in real-world scenarios.

6. Rapid Response Time: Optimize the system to provide quick response times for fingerprint authentication and vehicle startup.

7. Power Efficiency: Design the system to operate efficiently with minimal power consumption to avoid draining the vehicle's battery.

8. Scalability: Allow for easy addition and management of multiple authorized users' fingerprints.

9. Security Against Tampering: Implement security measures to prevent tampering and unauthorized attempts to bypass the fingerprint authentication.

10. Documentation and User Guide: Provide comprehensive documentation and a user guide to assist users in installing, using, and troubleshooting the system.

These objectives will guide in creating a robust and effective anti-theft vehicle system that leverages the capabilities of fingerprint sensors and Arduino technology.

CHAPTER-2

2.1 LITERATURE SURVEY

Here's a brief literature survey on anti-theft vehicle systems using fingerprint sensors and Arduino:

1. Anti-Theft System For Vehicles Using Fingerprint Sensor by Joel Sachin and Kiran Rana Gill

This paper focuses on preventing car theft using microcontrollers and GSM modules. It highlights the integration of biometric technology (fingerprint sensors) with GSM and GPS for real-time tracking and active notifications to the vehicle owner. The system ensures multiple layers of security to prevent theft.

2. Automobile Security System Using Fingerprint Sensor and Arduino by Pavandeep Kaur et al.

This research presents an advanced vehicle protection device that uses an Arduino and fingerprint sensor. The system is designed to provide maximum protection for low-range vehicles. It emphasizes the uniqueness of fingerprints as a security mechanism and integrates various components like an LCD monitor and relay switch.

3. Fingerprint Authentication System for Vehicles Using Arduino by Dr. Laya Tojo et al.

This paper discusses the development of a fingerprint-based vehicle security system. It combines advanced technology to enhance both security and safety features of vehicles. The system

uses fingerprint sensors interfaced with Arduino and includes features like GSM and GPS for added security.

4. Fingerprint Sensor Based Anti-Theft System for Vehicles by S. Selvaraju

This paper explores the use of micro controllers and GSM modules to prevent automobile burglary. The system uses a fingerprint sensor to capture and compare fingerprints, ensuring only authorized users can access the vehicle.

5. Keyless Automobile Entry System by Rajeev Velikkal

This research focuses on creating a key less entry system using Arduino. The system uses simple programming languages to design a secure and user-friendly vehicle entry system.

2.2 PROPOSED IDEA

The proposed project aims to develop an advanced anti-theft vehicle-system that leverages biometric fingerprint authentication and Arduino technology. This system will enhance vehicle security by ensuring only authorized users can access and operate the vehicle, thus preventing theft and unauthorized use.

2.3 MERITS AND DEMERITS

Merits:

1. Enhanced Security: Fingerprint authentication significantly enhances vehicle security by ensuring that only authorized individuals can access and operate the vehicle.
2. Biometric Precision: The use of unique biometric data (fingerprints) reduces the likelihood of unauthorized access compared to traditional key-based systems.

3. User-Friendly: The system is easy to use, as users only need to place their finger on the sensor to authenticate and start the vehicle.
4. Convenience: Eliminates the need to carry keys or remember passcodes, making it a convenient solution for vehicle owners.
5. Integration Potential: Can be integrated with other security systems, such as GSM and GPS, for real-time tracking and notifications.
6. Scalability: The system can easily be scaled to accommodate multiple users by storing multiple fingerprint templates.
7. Quick Response: Offers rapid authentication and vehicle startup, enhancing the overall user experience.
8. Reduced Theft Risk: Significantly reduces the risk of vehicle theft by providing a robust defense mechanism against unauthorized access.

Demerits:

1. Cost: The initial cost of implementing a fingerprint-based anti-theft system may be higher compared to traditional keybased systems.
2. Power Consumption: Continuous operation of the fingerprint sensor and Arduino may consume more power, potentially impacting the vehicle's battery life.
3. Sensor Accuracy: In certain conditions (e.g., dirty or wet fingers), the fingerprint sensor may experience difficulties in accurately capturing and verifying fingerprints.

4. **Complex Installation:** Integrating the system with the vehicle's existing ignition and immobilizer systems may require technical expertise and can be complex.
5. **Maintenance:** Regular maintenance and updates may be needed to ensure the system's reliability and security.
6. **False Rejections:** There is a potential for false rejections, where the system fails to recognize an authorized fingerprint, causing inconvenience to the user.
7. **Tampering Risks:** Although the system enhances security, there is still a risk of tampering with the hardware components if not properly secured.
8. **Dependency on Fingerprints:** In cases where the user's fingerprint is damaged or altered (e.g., cuts, burns), the system may fail to recognize the authorized user.

2.4 SUMMARY

Biometric fingerprint sensors offer a higher level of security compared to traditional key-based systems. Fingerprint authentication provides a user-friendly and convenient method for vehicle access. Combining fingerprint sensors with other technologies like GSM and GPS enhances overall vehicle security. Fingerprint sensors demonstrate high accuracy in capturing and verifying fingerprints, minimizing false acceptances and rejections.

Issues such as cost, power consumption, sensor accuracy in varying conditions, and complexity of installation and maintenance are identified as potential challenges. The literature survey underscores the potential of fingerprint sensor-based anti-theft vehicle systems to provide a reliable and efficient solution

for modern vehicle security challenges. It also highlights the need for continued innovation and improvement to address existing challenges and enhance system performance.

CHAPTER-3

3.1 METHODOLOGY

A method is proposed for putting the vehicle in motion using a fingerprint control. All past work relating to fingerprint biometric security acts on the ignition system. This security mechanism, however, is easily circumvented. Direct manipulation of the ignition wires kept behind the wheel takes less effort to start the car. Rather of protecting ignition, we've implemented a security feature that allows only fingerprintauthenticated individuals to start the vehicle.

Fingerprint authentication is performed using the R307 sensor module. Fingerprint Enrolment/Addition and Fingerprint Recognition are the two sub modules of the Fingerprint Sensor Module. The fingerprint enrolment/addition module enrolls and stores all users who are permitted to operate the vehicle. This sub-module allows legitimate users' fingerprints to be enrolled in the database. The Fingerprint Module R307 sensor contains a micro switch with three buttons to help with enrolling. The first button is used to collect the fingerprint's 3D image. The other two options are for adding and removing fingerprints from the database. The input fingerprint is compared to one or more templates in the database by the fingerprint matching sub-module. Pattern matching and minutiae-based matching are the fingerprint matching approaches employed. Before a user can operate a vehicle, his or her fingerprint is compared to one in the database. The signals from the fingerprint sensor Module R307 are sent to the Arduino Uno board's microprocessor ATmega328.

The microcontroller sends the desired signal to start the car once the scanned fingerprint matches the one stored in the database. The state of the system is displayed on an LCD display. It also shows when fingerprints are being added, erased, or when authentication is successful.

The Arduino is used as the microcontroller board for processing the input signal. It reads the state of input, which can be 0 or 1. Input is fed to the system with the help of fingerprint sensor. When the input fingerprint is given the Arduino reads the state of input.

If the fingerprint matches with the preset one then the ignition will be turned on. In case of mismatch of the fingerprint, the system asks for an authentication from the owner(pin). Once the authentication is given Arduino sends a signal to the relay which in turn initiates the ignition to start the vehicle.

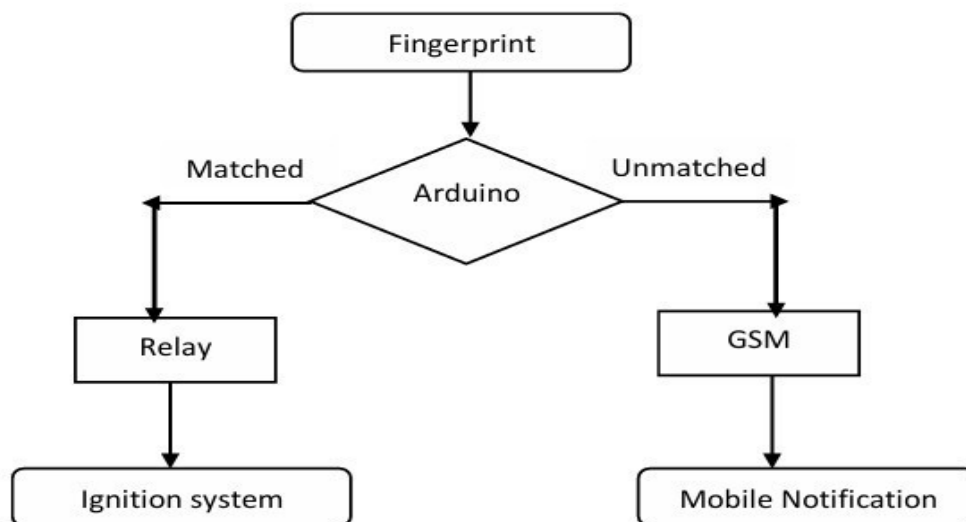


Fig 1: Flowchart for Biometric Vehicle Ignition System

3.2 HARDWARE AND SOFTWARE

The Hardware requirements are as follows:

3.2.1 Arduino Uno

The Arduino Uno is a widely-used microcontroller board based on the ATmega328P chip. It is open-source and designed to make electronics more accessible to everyone, from hobbyists to professionals. its key features are :

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Input Voltage: 7-12V (recommended)
- Digital I/O Pins: 14 (of which 6 can be used as PWM outputs)
- Analog Input Pins: 6
- Flash Memory: 32 KB (ATmega328P) of which 0.5 KB is used by the boot loader
- SRAM: 2 KB
- EEPROM: 1 KB
- Clock Speed: 16 MHz

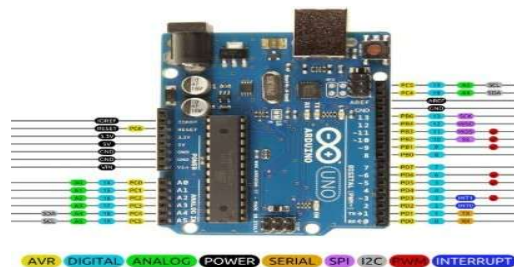


FIG 2: ARDUINO UNO

It's popular for projects ranging from simple LED blinkers to complex robotics and IoT devices. The Arduino Uno is particularly favored for its user-friendly nature and extensive

community support, making it an ideal choice for those new to electronics and programming.

3.2.2 Fingerprint Sensor R-305

Fingerprint processing includes two major parts: fingerprint enrolment and fingerprint matching (the matching can be 1:1 or 1:N). While enrolling, the user needs to enter his/her finger two times. The system will process the finger images and then generate a template of the finger based on the processing results and store the corresponding template.

While matching, the user enters the finger again through the optical sensor and the system will generate a template of the finger and compare it with available templates of the available finger library. For 1:1 matching, the system will then compare the live finger with the specific template designated in the given Module; for 1:N matching, or searching, the system will then search the whole finger library for the available matching finger. In both such circumstances, the system will then return the matching result, as either success or failure.



FIG 3: FINGERPRINT SENSOR

3.2.3 GSM MODULE SIM800L

The SIM800L GSM Module is a compact and versatile device designed for use in embedded systems. Here are some key features:

- Microcontroller: Uses the SIM800L chip from Simcom.
- Operating Voltage: 3.4V to 4.4V.
- Connectivity: Supports quad-band GSM/GPRS networks (850/900/1800/1900 MHz), making it compatible with most GSM networks worldwide.
- Data Transfer: Supports GPRS Class 10 for data transfer rates up to 85.6 kbps.
- Communication: Can send and receive SMS messages, make and receive phone calls, and connect to the internet via GPRS.
- Power Consumption: Low power consumption, making it suitable for battery-powered devices.
- Size: Compact size, ideal for IoT projects, home automation, and remote monitoring.

The SIM800L module is commonly used in applications such as vehicle tracking, remote monitoring, home automation, and industrial control systems.

3.2.4 5-PIN RELAY

A 5-pin relay is an electromechanical switch used to control a high-power circuit with a low-power signal. ### Pin

Configuration

- Coil End 1 (A1): One end of the coil, used to trigger the relay.
- Coil End 2 (A2): The other end of the coil, typically connected to ground.
- Common (COM): Connects to one end of the load (the circuit you want to control).

- Normally Closed (NC): Connects to the other end of the load when the relay is not activated.
- Normally Open (NO): Connects to the other end of the load when the relay is activated.

When a current flows through the coil (A1 and A2), it creates a magnetic field that moves the switch inside the relay.

When the relay is not activated, the circuit is completed between COM and NC.

When the relay is activated, the circuit is completed between COM and NO.

3.2.5 ARDUINO IDE

Arduino IDE is a GUI based Software that supports all the Arduino based microcontrollers. It is a cross platform application written in the programming language . It is an opensource Software (IDE) that makes it very easy to write code and also upload it to the board.

It runs on various operatingsystems Windows, Mac OS X and Linux. It originated from the IDE for the languages such as Processing and Wiring. A program written with the IDE for Arduino is called a "sketch". The Arduino IDE supports the languages such as C and C++ using special rules to organize the code. The Arduino IDE supplies a software library called Wiring from the Wiring project, which provides a lot of common input and output procedures.

3.3 BLOCK DIAGRAM

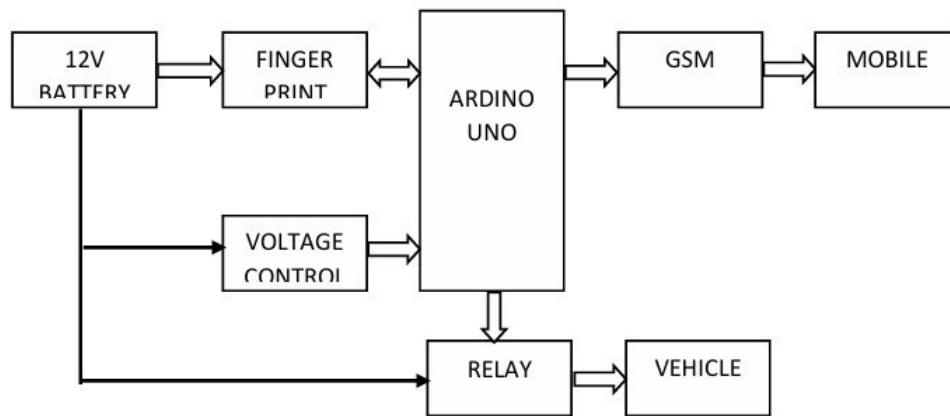


FIG 4 BLOCK DIAGRAM

3.3 PROGRAM

3.3.1 Libraries

- `#include <Adafruit_Fingerprint.h>`: This library is used to interface with the fingerprint sensor.
- `#include <SoftwareSerial.h>`: This library allows you to use software-based serial communication on any digital pins.
- `#include <LiquidCrystal.h>`: This library is used to control the LCD display.

3.3.2 Pin Definitions

- Buttons:
 - ENROLL_BUTTON_PIN (pin 6) for enrolling a fingerprint.
 - DELETE_BUTTON_PIN (pin 7) for deleting a fingerprint.
- Relay:
 - RELAY_PIN (pin A0) to control a relay.
- GSM Module:

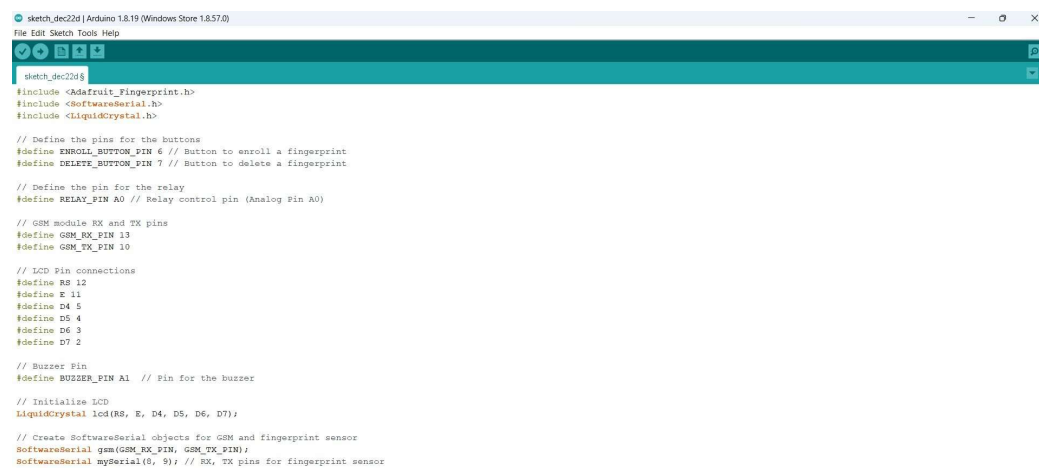
GSM_RX_PIN (pin 13) and GSM_TX_PIN (pin 10) for GSM communication.

- LCD: RS (pin 12), E (pin 11), D4 (pin 5), D5 (pin 4), D6 (pin 3), and D7 (pin 2) for LCD control.
- Buzzer:BUZZER_PIN (pin A1) to control a buzzer.

3.3.4 Object Initialization:

- LCD Display: Initialized using the LiquidCrystal object.
- GSM Module: Uses SoftwareSerial for communication.

3.3.5 CODE



```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

// Define the pins for the buttons
#define ENROLL_BUTTON_PIN 6 // Button to enroll a fingerprint
#define DELETE_BUTTON_PIN 7 // Button to delete a fingerprint

// Define the pin for the relay
#define RELAY_PIN A0 // Relay control pin (Analog Pin A0)

// GSM module RX and TX pins
#define GSM_RX_PIN 13
#define GSM_TX_PIN 10

// LCD Pin connections
#define RS 12
#define E 11
#define D4 5
#define D5 4
#define D6 3
#define D7 2

// Buzzer Pin
#define BUZZER_PIN A1 // Pin for the buzzer

// Initialize LCD
LiquidCrystal lcd(RS, E, D4, D5, D6, D7);

// Create SoftwareSerial objects for GSM and fingerprint sensor
SoftwareSerial gsm(GSM_RX_PIN, GSM_TX_PIN);
SoftwareSerial mySerial(8, 9); // RX, TX pins for fingerprint sensor
```



```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d
// Initialize the fingerprint sensor object using the SoftwareSerial object
Adafruit_Fingerprint finger = Adafruit_Fingerprint(mySerial);

// Array to keep track of stored fingerprint IDs (1 = stored, 0 = empty)
bool storedIds[128] = {false}; // Index 0 not used, valid IDs are 1-127

// Variables to keep track of button states and modes
bool enrollMode = false;
bool deleteMode = false;
uint8_t nextEnrollID = 1; // Next available ID for enrollment

void setup() {
  Serial.begin(9600); // Initialize serial communication
  while (!Serial); // Wait for serial to initialize
  delay(100);

  // Set up the LCD
  lcd.begin(16, 2);
  lcd.print(F("Initializing..."));

  // Initialize GSM module communication
  gsm.begin(9600);
  delay(3000); // Wait for GSM module to initialize
  gsm.println("AT");
  waitForGSMResponse("OK");

  // Set SMS mode to Text
  gsm.println("AT+CMGF=1");
  waitForGSMResponse("OK");

  Serial.println("GSM Module Initialized.");
}
```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$
// Set up the fingerprint sensor
finger.begin(57600);
if (finger.verifyPassword()) {
  Serial.println(F("Fingerprint sensor found!"));
  updateStoredIds(); // Initialize the stored IDs array
  displayFingerprintStatus(); // Display initial status
  lcd.clear();
  lcd.print(F("Place Finger"));
} else {
  lcd.clear();
  lcd.print(F("Sensor not found"));
  Serial.println(F("Fingerprint sensor not found!"));
  while (1) {
    delay(2);
  }
}

// Initialize the button pins as inputs
pinMode(ENROLL_BUTTON_PIN, INPUT);
pinMode(DELETE_BUTTON_PIN, INPUT);

// Set up the relay pin as output
pinMode(RELAY_PIN, OUTPUT);
digitalWrite(RELAY_PIN, LOW); // Ensure the relay is initially off

// Set up the buzzer pin as output
pinMode(BUZZER_PIN, OUTPUT);
digitalWrite(BUZZER_PIN, LOW); // Ensure the buzzer is initially off
}
```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$
void loop() {
  // Continuously scan for fingerprints
  if (finger.getImage() == FINGERPRINT_OK) {
    int p = finger.image2Tz(1); // Convert image to template
    if (p != FINGERPRINT_OK) {
      lcd.clear();
      lcd.print(F("Failed to convert image"));
      Serial.println(F("Failed to convert image"));
      return;
    }

    // Try to identify the fingerprint using fast search
    p = finger.fingerFastSearch();
    if (p == FINGERPRINT_OK) {
      lcd.clear();
      lcd.print(F("Fingerprint ID: "));
      lcd.print(finger.fingerID);
      Serial.print(F("Fingerprint matched with ID: "));
      Serial.println(finger.fingerID);

      // Turn on the relay (motor on)
      digitalWrite(RELAY_PIN, HIGH);
    } else {
      lcd.clear();
      lcd.print(F("Fingerprint not found"));
      Serial.println(F("Fingerprint not matched!"));

      // Turn off the relay (motor off)
      digitalWrite(RELAY_PIN, LOW);

      // Sound the buzzer for 10 seconds with 1-second intervals
      soundBuzzer();
    }
  }
}
```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$
// Send SMS alert
sendSMS("+919328649703", "Wrong authentication! Someone is trying to take your vehicle.");
}
delay(2000); // Show result for 2 seconds
}

// Check if the Enroll button is pressed
if (digitalRead(ENROLL_BUTTON_PIN) == HIGH) {
  if (!enrollMode) {
    enrollMode = true;
    deleteMode = false;
    nextEnrollID = findNextAvailableID();
    lcd.clear();
    lcd.print(F("Enroll Mode"));
    lcd.setCursor(0, 1);
    lcd.print(F("ID: "));
    lcd.print(nextEnrollID);
    Serial.print(F("Entering Enroll Mode. Next ID: "));
    Serial.println(nextEnrollID);
    delay(1000);
  } else {
    enrollMode = false;
    enrollFingerprint(nextEnrollID); // Enroll a new fingerprint
  }
}

// Check if the Delete button is pressed
if (digitalRead(DELETE_BUTTON_PIN) == HIGH) {
  if (!deleteMode) {
    deleteMode = true;
    enrollMode = false;
    lcd.clear();
  }
}
```

```

sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$
    enrollMode = false;
    lcd.clear();
    lcd.print(F("Delete Mode"));
    Serial.println(F("Entering Delete Mode"));
    delay(1000);
  } else {
    deleteMode = false;
    deleteLastFingerprint(); // Delete the last stored fingerprint
  }
}

// Function to sound the buzzer for 10 seconds with 1 second delay between each second
void soundBuzzer() {
  for (int i = 0; i < 10; i++) { // Loop for 10 seconds
    digitalWrite(BUZZER_PIN, HIGH); // Turn buzzer on
    delay(1000); // Wait for 1 second
    digitalWrite(BUZZER_PIN, LOW); // Turn buzzer off
    delay(1000); // Wait for 1 second
  }
}

// Function to find the next available ID
uint8_t findNextAvailableID() {
  for (int i = 1; i <= 127; i++) {
    if (!toredids[i]) {
      return i;
    }
  }
  return 0; // No available ID
}

```

```

sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$

// Function to enroll a fingerprint
void enrollFingerprint(uint8_t id) {
  lcd.clear();
  lcd.print(F("Enrolling ID: "));
  lcd.print(id);
  Serial.print(F("Enrolling ID #"));
  Serial.println(id);

  while (!getFingerprintEnroll(id)) {
    delay(100);
  }

  displayFingerprintStatus();
}

uint8_t getFingerprintEnroll(uint8_t id) {
  int p = -1;
  lcd.clear();
  lcd.print(F("Place finger"));
  Serial.println(F("Waiting for valid finger"));

  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
      case FINGERPRINT_OK:
        Serial.println(F("Image taken"));
        lcd.clear();
        lcd.print(F("Image taken"));
        break;
      case FINGERPRINT_NOFINGER:
        Serial.println(F("No finger detected"));
        break;
    }
  }
}

```

```

sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$

    break;
    case FINGERPRINT_NOFINGER:
      Serial.println(F("No finger detected"));
      break;
    case FINGERPRINT_PACKETRECEIVED:
      Serial.println(F("Communication error"));
      break;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println(F("Imaging error"));
      break;
    default:
      Serial.println(F("Unknown error"));
      break;
  }
}

p = finger.image2Tz();
if (p != FINGERPRINT_OK) {
  Serial.println(F("Image conversion failed"));
  return p;
}

lcd.clear();
lcd.print(F("Remove finger"));
Serial.println(F("Remove finger"));
delay(2000);

p = 0;
while (p != FINGERPRINT_NOFINGER) {
  p = finger.getImage();
}

lcd.clear();

```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$

lcd.clear();
lcd.print(F("Place again"));
Serial.println(F("Place same finger again"));

p = -1;
while (p != FINGERPRINT_OK) {
  p = finger.getImage();
}

p = finger.image2Ts(2);
if (p != FINGERPRINT_OK) {
  Serial.println(F("Image conversion failed"));
  return p;
}

Serial.println(F("Creating model..."));

p = finger.createModel();
if (p != FINGERPRINT_OK) {
  Serial.println(F("Failed to create model"));
  return p;
}

p = finger.storeModel(id);
if (p == FINGERPRINT_OK) {
  storedIds[id] = true;
  Serial.println(F("Stored!"));
  lcd.clear();
  lcd.print(F("Stored ID: "));
  lcd.print(id);
  delay(2000);
  displayFingerprintStatus();
}
```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$

} else {
  Serial.println(F("Error storing fingerprint"));
  return p;
}

return true;
}

// Function to delete the last stored fingerprint
void deleteLastFingerprint() {
  lcd.clear();
  lcd.print(F("Deleting Finger"));
  Serial.println(F("Ready to delete a fingerprint!"));

  // Find the last stored ID
  uint8_t id = 0;
  for (int i = 127; i >= 1; i--) {
    if (storedIds[i]) {
      id = i;
      break;
    }
  }

  if (id == 0) { // No stored IDs
    lcd.clear();
    lcd.print(F("No IDs to delete"));
    return;
  }

  Serial.print(F("Deleting ID #"));
  Serial.println(id);
}
```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d$

int p = finger.deleteModel(id);
if (p == FINGERPRINT_OK) {
  storedIds[id] = false;
  Serial.println(F("Deleted!"));
  lcd.clear();
  lcd.print(F("Deleted ID: "));
  lcd.print(id);
  delay(2000);
  displayFingerprintStatus();
  nextEnrollID = findNextAvailableID(); // Update the next enroll ID
} else {
  Serial.println(F("Error deleting fingerprint"));
  lcd.clear();
  lcd.print(F("Delete failed"));
  delay(1000);
}

// Function to update stored IDs array and count
void updateStoredIds() {
  for (int i = 1; i <= 127; i++) {
    if (finger.loadModel(i) == FINGERPRINT_OK) {
      storedIds[i] = true;
    } else {
      storedIds[i] = false;
    }
  }
}
```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d

// Function to display fingerprint storage status
void displayFingerprintStatus() {
  Serial.println(F("\n== Fingerprint Storage Status =="));
  updateStoredIds(); // Update the stored IDs array and count
  uint8_t fingerprintCount = 0;
  for (int i = 1; i <= 127; i++) {
    if (storedIds[i]) {
      fingerprintCount++;
    }
  }
  Serial.print(F("Total stored fingerprints: "));
  Serial.println(fingerprintCount);
  lcd.clear();
  lcd.print(F("Stored FPs: "));
  lcd.print(fingerprintCount);
  if (fingerprintCount > 0) {
    Serial.println(F("\nStored fingerprint IDs:"));
    lcd.setCursor(0, 1); // Display stored IDs
    for (int i = 1; i <= 127; i++) {
      if (storedIds[i]) {
        Serial.print(F("ID #"));
        Serial.println(i);
        lcd.clear();
        lcd.print(F("Stored FPs: "));
        lcd.print(fingerprintCount);
        lcd.setCursor(0, 1);
        lcd.print(F("ID: "));
        lcd.print(i);
        delay(1000);
      }
    }
  } else {

```

```
sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d

} else {
  Serial.println(F("No fingerprints stored"));
  lcd.setCursor(0, 1);
  lcd.print(F("No stored FPs"));
}
Serial.println(F("===== \n"));
delay(2000);

// Function to send an SMS
void sendSMS(String phoneNumber, String message) {
  Serial.println("Sending SMS...");

  // Send the AT command to start SMS
  gsm.print("AT+CMGF=1");
  gsm.print(phoneNumber);
  gsm.println("");
  delay(100); // Wait for the GSM module to respond

  // Send the message text
  gsm.print(message);
  delay(100); // Wait for the GSM module to respond

  // End the message with Ctrl+Z (ASCII 26)
  gsm.write(26);
  delay(5000); // Wait for the SMS to be sent

  if (gsm.available()) {
    String response = gsm.readString();
    Serial.println("GSM Response: ");
    Serial.println(response);
  }
}
```

```

sketch_dec22d | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

sketch_dec22d.g
}

// Function to wait for a specific response from GSM module
void waitForGSMResponse(String expectedResponse) {
    String response = "";
    long timeout = millis() + 5000; // 5 seconds timeout

    while (millis() < timeout) {
        if (gsm.available()) {
            char c = gsm.read();
            response += c;
        }

        if (response.indexOf(expectedResponse) != -1) {
            Serial.println("Response: " + response);
            return;
        }
    }

    Serial.println("Response timeout or unexpected response: " + response);
}

// Function to read an ID number from the Serial input
uint8_t readnumber(void) {
    uint8_t num = 0;

    while (num == 0) {
        while (!Serial.available()); // Wait for input
        num = Serial.parseInt(); // Parse the input number
    }

    return num;
}
}

```

3.3.6 PROGRAM FUNCTIONING

1. setup() Function:

- **Serial Communication:** Initializes the serial communication at 9600 baud rate.
- **LCD Display:** Initializes and displays "Initializing...".
- **GSM Module:** Sets up communication. Sends initial commands to the GSM module. Sets SMS mode to Text.
- **Fingerprint Sensor:** Initializes the sensor. Verifies the password. Updates the stored IDs array. Displays initial status on the LCD.
- **Pins Configuration:** Sets the button pins as inputs. Sets the relay and buzzer pins as outputs and ensures they are initially off.

2. loop() Function

- **Fingerprint Scanning:** Continuously scans for fingerprints. If a fingerprint is found: Converts the image to a template. Identifies the fingerprint using a fast search. If matched, displays the fingerprint ID, turns on the relay (motor on). If not matched, displays a message, turns off the relay (motor off), sounds the buzzer, and sends an SMS alert.

- Enroll Button: If pressed, enters enroll mode and displays the next available ID. If already in enroll mode, calls the function to enroll a new fingerprint.
- Delete Button Handling: Checks if the delete button is pressed. If the button is pressed for the first time, it enters "Delete Mode". If already in delete mode and the button is pressed again, it exits delete mode and calls the deleteLastFingerprint function.
- Sound Buzzer Function: Sounds the buzzer for 10 seconds with a 1-second interval. Uses a loop to turn the buzzer on and off every second for a total duration of 10 seconds.
- Find Next Available ID: Finds the next available fingerprint ID. Loops through the stored Ids array to find the first empty slot and returns the ID. Returns 0 if no ID is available.
- EnrollFingerprintFunction: Enrolls a new fingerprint. Displays the enrolling message and calls `getFingerprintEnroll` function to handle the actual enrollment process. Updates the display with the fingerprint status.
- Get Fingerprint Enroll Function: Handles the detailed process of enrolling a fingerprint. Waits for the finger to be placed on the sensor. Converts the image to a template. Prompts the user to remove and place the finger again. Converts the second image to another template. Creates a model from the two templates. Stores the model with the given ID and updates the status.
- Delete Last Fingerprint Function: Deletes the last stored fingerprint. Finds the last stored fingerprint ID. Deletes the fingerprint model with the found ID. Updates the `storedIds` array and display with the new status.

- **Update Stored IDs Array:** This function updates the array stored IDs to keep track of which fingerprint IDs are currently stored in the fingerprint sensor. It loops through IDs 1 to 127. If a fingerprint model is successfully loaded for a given ID (FINGERPRINT_OK), it marks that ID as true (stored). Otherwise, it marks the ID as false (empty).
- **Display Fingerprint Storage Status:** Displays the status of stored fingerprints on the Serial Monitor and the LCD. Calls `updateStoredIds()` to refresh the stored IDs array. Counts the number of stored fingerprints. Displays the total count and each stored ID on the Serial Monitor. Shows the same information on the LCD.
- **Send SMS:** Sends an SMS to the specified phone number with the provided message. Sends the 'AT+CMGS' command with the phone number. Sends the message text. Ends the message with Ctrl+Z (ASCII 26). Waits for the GSM module's response and prints it on the Serial Monitor.
- **Wait for GSM Response:** Waits for a specific response from the GSM module. Sets a timeout of 5 seconds. Continuously reads characters from the GSM module and builds the response string. Checks if the response contains the expected response string. Prints the response or a timeout message on the Serial Monitor.
- **Read an ID Number from Serial Input:** Reads an ID number from the Serial input. Waits until a number is available on the Serial input. Parses and returns the input number.

CHAPTER-4 4.1 CIRCUIT CONNECTIONS

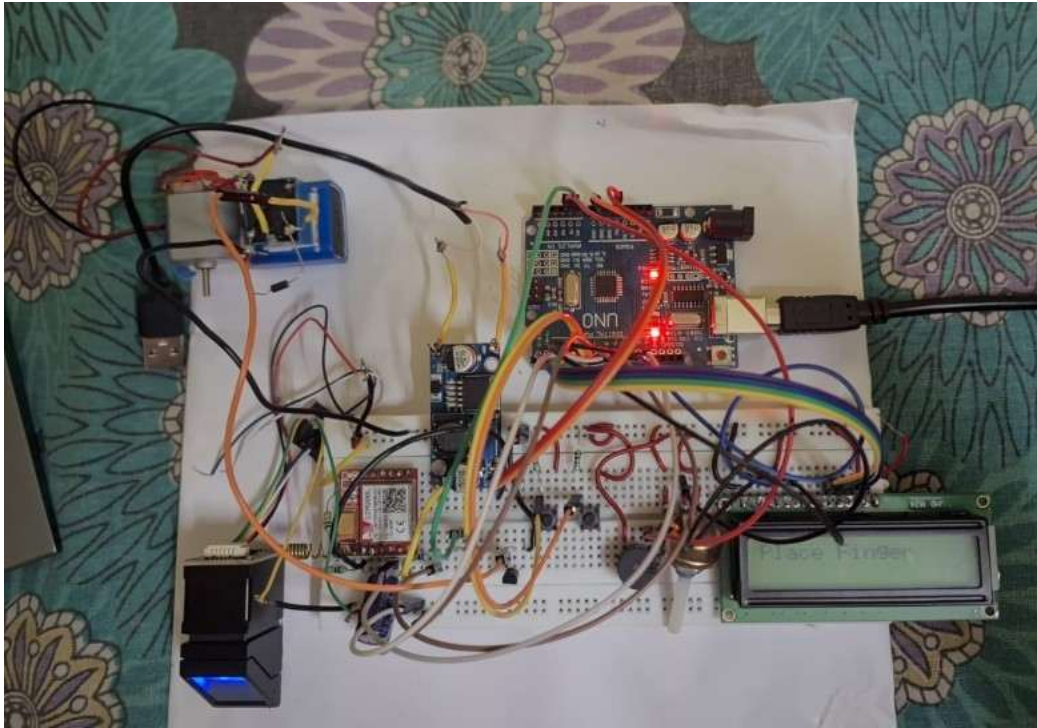


FIG 5 CIRCUIT CONNECTION

4.2 TESTING AND RESULTS



FIG 6 INITIALISING SENSOR



FIG 7 ENROLLING FINGERPRINT



FIG 8 READY TO SCAN



FIG 9 DELETING FINGERPRINT



FIG 10 DISPLAYING CONTENT



FIG 11 FINGERPRINT NOT
MATCHED



FIG 12 MESSAGE FROM GSM

4.3 CONCLUSION

The proposed system is pointed towards working fair and square of safety in automobiles. It can be concluded that the theft of the automobiles can be reduced/controlled by making use of methodologies based on IOT and biometric fingerprint technology. The major advantage being the high security and quick intimation to the owners.

Using the fingerprint sensor module the input is given to the circuit. The next step is based on the result of the two cases: if the given input fingerprint matches, the relay in the circuit ignites the spark plug to power up the engine. If the fingerprint doesn't match, a message is sent to the owner indicating that the given input fingerprint doesn't match with the pre-fed input data. Later

if the owner wants to grant permission for the usage of vehicle by a third person, he is supposed to send back a password which in-turn starts the engine. The system also consists of LCD display in the circuitry to indicate whether the fingerprint is matched or not.

The major application of the project is to reduce the thefts of the automobiles by the advancement of the security system using biometrics. Recent advancements in the field of biometrics for security purpose makes this design commercially possible to implement for real-time usage. The system accuracy is more due to the biometric (which is unique for an individual and can't be hacked) criteria used for security purpose. The proposed system is cost effective compared to other systems that are available in the market. The proposed system can be interfaced with any existing vehicle.

4.4 REFERENCES

- [1]. Sadagopan, Vinoth Kumar, UpendranRajendran, and Albert Joe Francis. "Anti-theft control system design using embedded system." Proceedings of International Conference on Vehicular Electronics and Safety, 2011 IEEE.
- [2]. Pawar, Mahesh R., and Imdad Rizvi. "IoT based embedded system for vehicle security and driver surveillance." 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT). IEEE, 2018.
- [3]. Manjunath, T. K., Andrews SamrajMaheswari, and Chidaravalli Sharmila. "Locking and Unlocking of Theft Vehicles Using CAN." Proceedings of 2013 International Conference on Green High Performance Computing. 2013. [4]. Mukhopadhyay, Debajyoti, et al. "An attempt to develop an iot based vehicle security system." 2018 IEEE International

Symposium on Smart Electronic Systems (iSES)(Formerly iNiS). IEEE, 2018.

[5]. Ramadan, Montaser N., Mohammad A. Al-Khedher, and Sharaf A. Al-Kheder. "Intelligent anti-theft and tracking system for automobiles." *International Journal of Machine Learning and Computing* 2.1 (2012).

[6]. Jesudoss, A., R. Vybhavi, and B. Anusha. "Design of smart helmet for accident avoidance." 2019 International Conference on Communication and Signal Processing (ICCSP). IEEE, 2019