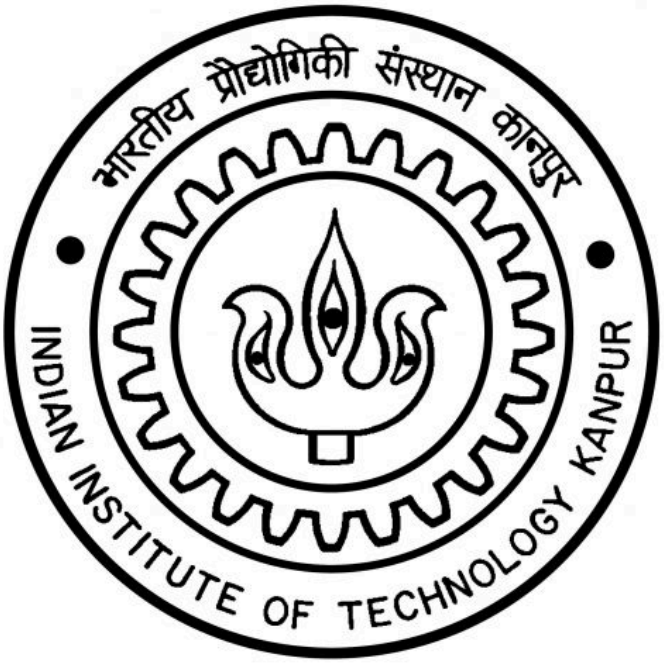
EE 381A: Electronic Circuits Laboratory



Project Report

GPS + Fingerprint Based Door Locking System

Section	C
Table Number	11
Day	Tuesday
Group Members	1. Lavesh Gupta (**210562**) 2. Parv Chhabra (**210706**) 3. Pranav Susana (**210739**)

Q1. What problem are you trying to solve, and why is it important / interesting?
Ans: The problem We're trying to solve with our GPS+Fingerprint based door locking system is to enhance security and convenience for door access control. This is crucial because it addresses several common concerns that many people face:

1. **Security**: By incorporating fingerprint authentication, we're adding a high level of security since each fingerprint is unique, making it difficult for unauthorised individuals to gain access.

2. **Convenience**: Being able to lock and unlock the door using a mobile device adds a significant level of convenience, especially in situations where carrying physical keys might be cumbersome or inconvenient.

3. **Automation**: Automatically locking the door when the mobile device goes out of range adds an extra layer of security and peace of mind, reducing the risk of leaving the door unlocked accidentally.

Overall, our system not only provides a more secure method of access control but also streamlines the process for users, making it both important and interesting for home or office security applications.

Q2. What are the existing solutions? Describe a few of them and list any shortcomings in them. Is your solution approach unique in some way?
Ans: Existing solutions for door locking systems vary in their technology and features. Some common solutions include traditional key-based locks, keypad locks, RFID card-based systems, and Bluetooth-enabled locks. Here's a brief overview of each with their shortcomings:

1. **Traditional Key-Based Locks**: These locks are simple and reliable but can be lost or duplicated, posing a security risk. They also lack features like remote access and activity tracking.

2. **Keypad Locks**: Keypad locks allow users to enter a code for access. However, these codes can be easily guessed or shared, compromising security. They also don't offer the convenience of remote locking/unlocking.

3. **RFID Card-Based Systems**: These systems use RFID cards for access control. While more secure than keys or codes, RFID cards can still be lost or stolen. Managing multiple cards for different users can also be cumbersome.

4. **Bluetooth-Enabled Locks**: Bluetooth locks allow for keyless entry using a smartphone. While convenient, they may have limited range and can be vulnerable to Bluetooth hacking or signal interference.

Shortcomings in these existing solutions include security vulnerabilities, lack of

convenience features like remote access, and potential issues with managing access for multiple users.

Our solution combines GPS and fingerprint technology, offering a unique approach to door locking systems. By integrating GPS, we can automate locking based on the proximity of the mobile device, adding an extra layer of security. The fingerprint authentication ensures only authorized individuals can unlock the door, enhancing overall security. Additionally, our system provides the convenience of remote access control via a mobile app, addressing the shortcomings of traditional and Bluetooth-based locks.

Q3. What resources do you require to complete the project? Give a breakup of tasks that you need to accomplish week by week to complete the project.
Ans: Below is the complete list of components required -

S.No.	Component Name	Quantity
1.	DC 12V Solenoid Electromagnetic Cabinet Door Lock	1
2.	Arduino Nano	1
3.	HC05 Bluetooth Module	1
4.	Relay Module Single Channel	1
5.	Breadboard	1
6.	Jumper Wires	10
7.	LED	2
8.	GPS Module Ublox Neo-6M	1
9.	USB Cable	1

Week Wise Breakup of Tasks is as follows:

Week 1:
1.1) We began by researching and acquiring all the necessary components for our GPS+Fingerprint based door locking system, including the GPS module, Arduino board, Bluetooth module, solenoid lock, and other required parts.
1.2) To ensure a smooth start, we familiarized ourselves with Arduino programming, Bluetooth communication, GPS integration, solenoid lock control, and other relevant technologies.
1.3) As a preliminary test, we wrote and uploaded a simple Arduino program that blinked an LED light on and off. Additionally, we used Arduino code to measure and determine the appropriate resistor value for our circuit.

Week 2:
2.1) The focus was on building the circuit on a breadboard, incorporating the GPS module, Bluetooth module, and solenoid lock. We conducted thorough testing of these connections to ensure they functioned correctly.
2.2) We also conducted independent tests on the solenoid lock to ensure it worked as expected.
2.3) For control testing, we developed an Arduino program that allowed manual control of the solenoid lock using a pushbutton or switch.
2.4) Additionally, we wrote an Arduino program to read fingerprint data transmitted from smartphone over Bluetooth. We extensively tested Bluetooth communication to ensure accurate transmission of fingerprint data.

Week 3:
3.1) We developed an Android app capable of connecting to the Bluetooth module on our Arduino board and transmitting fingerprint and GPS location data..
3.2) Within the app, we wrote code to verify fingerprint data against authorised prints and control the solenoid lock based on authentication results.
3.3) We also integrated the GPS functionality into the app, allowing it to detect when the mobile device reached a threshold range from the door. This triggered automatic locking if the door was left unlocked.
3.4) The final step for this week was comprehensive testing of the entire system. We scanned fingerprints using the app, verified authentication, and observed how the solenoid lock responded based on authentication results. Additionally, we tested the automatic locking feature triggered by the GPS module.

EC Lab Project Report

Project Objective:

The objective of our project is to develop a robust and secure GPS+Fingerprint based door locking system that allows users to lock and unlock their doors using a mobile app with fingerprint authentication. Additionally, the system automatically locks the door if the mobile device goes out of a predefined range, ensuring enhanced security and convenience for users in controlling access to their premises.

Resources Used:

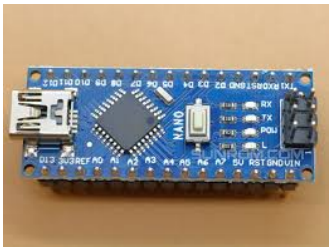
1. Arduino Nano
2. GPS Module
3. Dual Channel Relay module
4. Solenoid lock
5. Bluetooth module – HC-05

6. 12V DC power supply
7. Jumper wires
8. Breadboard
9. USB 2.0 Cable Type A/B

Description of each component:

1) Arduino Nano:

The Arduino Nano is a compact microcontroller board based on the ATmega328P chip, offering ample processing power and I/O capabilities. It operates at 5 volts, can be powered via USB or external supply, and features digital and analog I/O pins, PWM pins, and serial communication. Its mini USB port facilitates programming and communication with a computer, making it ideal for space-constrained projects like robotics, IoT devices, and embedded systems.

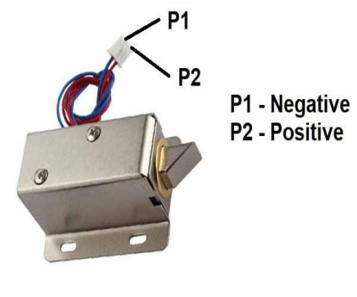


2) Solenoid Lock:

A 12V DC solenoid lock is an electromechanical device that is commonly used to secure doors, cabinets, and other access control points. When an electric current is applied to the solenoid, a magnetic field is generated, which pulls a metal plunger into the coil, thus locking the mechanism.

Here are some specifications of a typical 12V DC solenoid lock:

i) Operating Voltage: 12V DC
ii) Operating Current: 400mA to 500mA
iii) Operating Temperature: -20°C to 55°C
iv) Material: Stainless Steel
v) Holding Force: 600lbs (approx. 272kg)
vi) Lock Type: Fail-Secure
vii) Lock Mechanism: Solenoid
viii) Mounting Type: Surface Mount
ix) Dimensions: 150mm x 28mm x 28mm

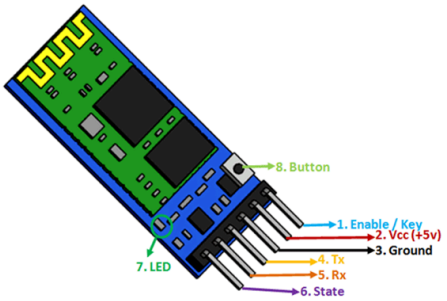


3) Bluetooth module – HC-05

The HC-05 Bluetooth module is a popular module that allows wireless communication between devices over Bluetooth. It uses the Bluetooth 2.0+EDR (Enhanced Data Rate) protocol and can be configured as either a master or a slave device.

Here are some specifications of the HC-05 module:

i) Bluetooth Version: Bluetooth 2.0+EDR
ii) Frequency Band: 2.4 GHz ISM band
iii) Modulation: GFSK (Gaussian Frequency Shift Keying)
iv) Transmit Power: Class 2, up to 4dBm
v) Sensitivity: -84dBm at 0.1% BER
vi) Range: Up to 10 meters (Class 2)
vii) Operating Voltage: 3.3V DC to 6V DC
viii) Current Consumption: <30mA (at 3.3V DC)
ix) Interface: UART (Universal Asynchronous Receiver/Transmitter)
x) Baud Rate: Default 9600 baud, configurable up to 1382400 baud
xi) Dimensions: 28mm x 15mm x 2.35mm



4) GPS MODULE

GPS module receives signals from satellites to determine precise geographic coordinates(latitude, longitude, and altitude) using trilateration. It provides accurate position data, interfaces with microcontrollers via UART or I2C, and is used in navigation systems for vehicles, drones, smartphones, and other devices.

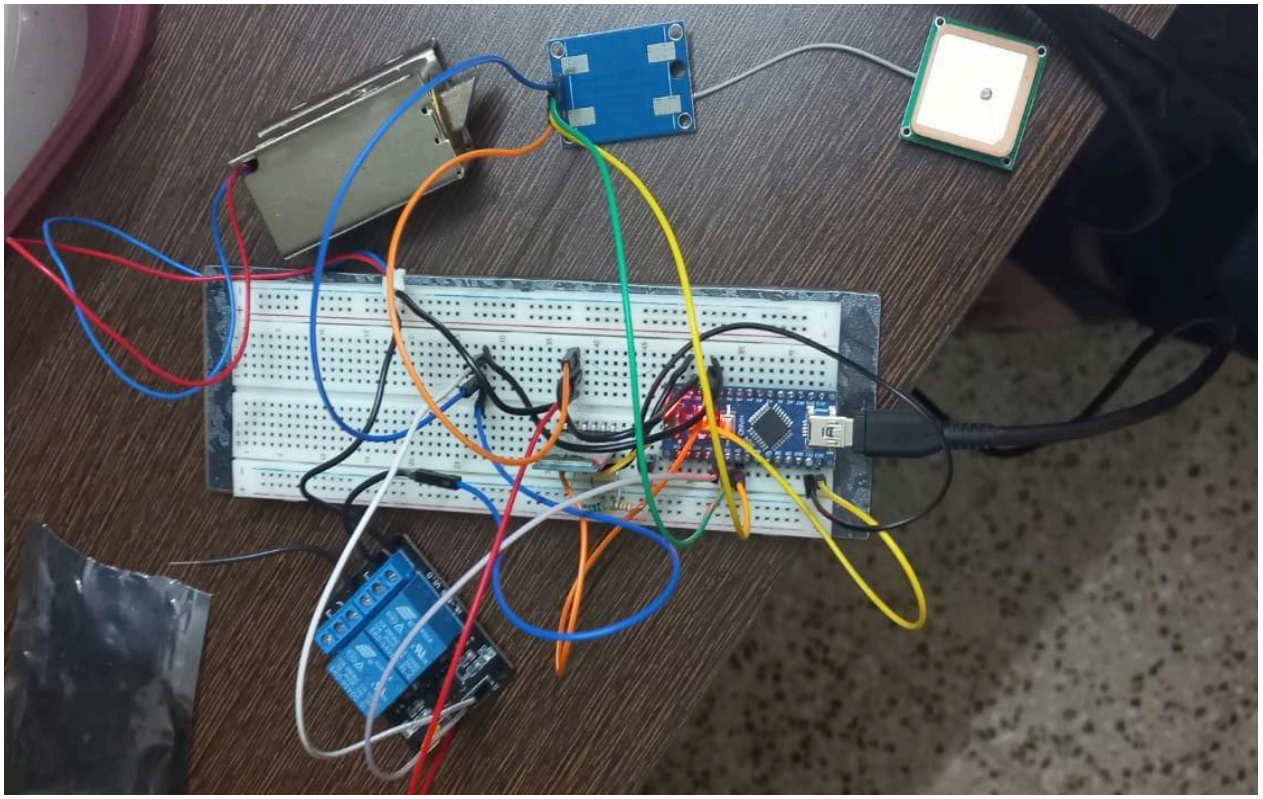
5) DOUBLE CHANNEL RELAY MODULE

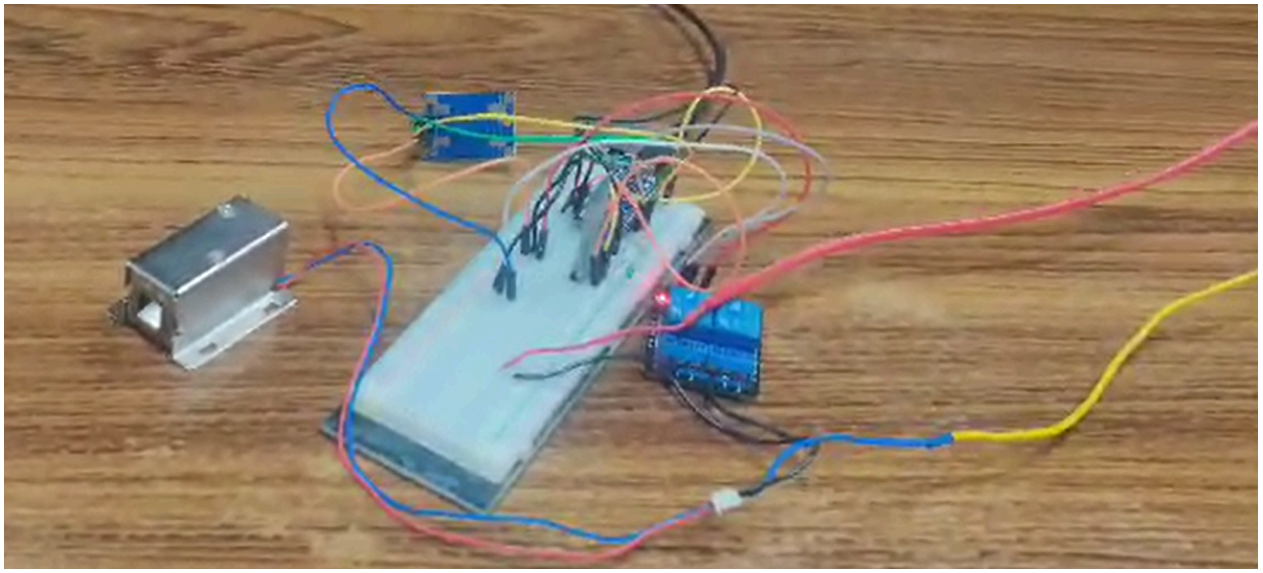
A double channel relay module switches two electrical circuits using electromechanical relays controlled by a low-voltage signal, commonly used in automation, robotics, and industrial control. It offers isolation, various voltage/current ratings, and reliable switching for diverse applications.

Implementation

a) **Connections:**

The TX and RX terminals were used by the HC05 and GPS module to send binary data(1 or 0) to the
Arduino nano depending on the fingerprint recognized.
If the fingerprint recognized is correct, the Bluetooth module sends 1 to the Arduino Nano. Whenever 1 is received by the Nano, it sets pin 12(D12) and pin 11(D11) to HIGH and LOW, respectively, and relayPIN (D2) to high.
If the distance between the mobile and the gps module connected to the door lock goes above 8m, the bluetooth module sends 1 to Arduino Nano and it sets pin 3(D3) to HIGH and pin 4(D4) to LOW, and relayPIN (D2) to high.





b) Programming the setup:

```
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include <math.h>

SoftwareSerial bluetoothSerial(12, 11);
SoftwareSerial gpsSerial(3,4);
const int relayPin = 2;
double x,y,rad;
double threshold = 8.0;
double RADIUS_EARTH = 6371000;

double haversine(double lat1, double lon1, double lat2, double lon2) {
```

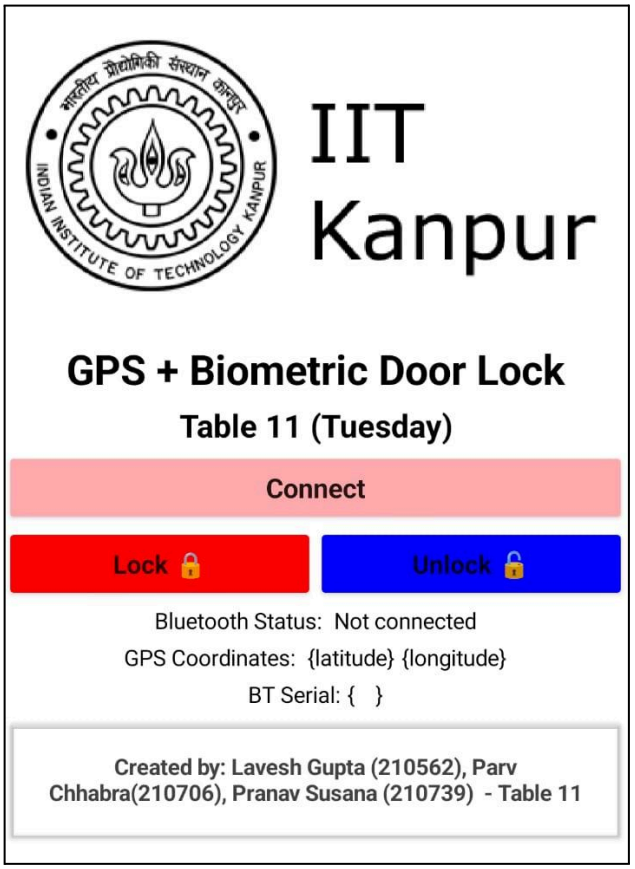
```
  // Convert latitude and longitude from degrees to radians
  lat1 = toRadians(lat1);
  lon1 = toRadians(lon1);
  lat2 = toRadians(lat2);
  lon2 = toRadians(lon2);

  // Haversine formula
  double dlat = lat2 - lat1;
  double dlon = lon2 - lon1;
  double a = pow(sin(dlat / 2), 2) + cos(lat1) * cos(lat2) * pow(sin(dlon / 2), 2);
  double c = 2 * asin(sqrt(a));
  double d = RADIUS_EARTH * c;

  return d;
}

double toRadians(double degree) {
  return degree * M_PI / 180.0;
}

void setup() {
  Serial.begin(9600);
  bluetoothSerial.begin(9600);
  // gpsSerial.begin(GPSBaud);
  pinMode(relayPin, OUTPUT);
}

void loop() {
  if (bluetoothSerial.available()){
    while (gpsSerial.available() > 0){
    if (gpsSerial.find("$GPRMC")) {
      float latitude = parseDegree();
      char lat = gpsSerial.read();
      if (lat == 'S') {
        latitude = -latitude;
      }

      float longitude = parseDegree();
      char lon = gpsSerial.read();
      if (lon == 'W') {
        longitude = -longitude;
      }
    }
    }
```

```
  String a = bluetoothSerial.readString(); // Read the incoming data from Bluetooth
  if(a.length()==1){
    char receivedChar = a.charAt(0);
    if(receivedChar =='1'){
    unlockDoor();
    delay(1000);
    }
    else if (receivedChar=='0'){
    lockDoor();
    delay(1000);
    }
 delay(1000);
 }
  else{
  int commaIndex = a.indexOf(',');
  if (commaIndex >= 0) {
    String latitudeStr = a.substring(0, commaIndex);
    String longitudeStr = a.substring(commaIndex + 1);

    // Convert the latitude and longitude strings to float values
    double latitude = latitudeStr.toDouble();
    double longitude = longitudeStr.toDouble();
    Serial.print(latitude, 6);
    Serial.print(", ");
    Serial.println(longitude, 6);



    double distance = haversine(lat1, lon1, latitude, longitude);

    if(distance > threshold){
      lockDoor();
    }

  }
 }
}
}

void unlockDoor(){
 digitalWrite(relayPin, LOW);
}
```

```
// Function to unlock the door
void lockDoor() {
  digitalWrite(relayPin, HIGH); // Turn ON the relay
}

float parseDegree() {
  float degree = gpsSerial.parseFloat();
  float minute = gpsSerial.parseFloat();
  return degree + minute / 60.0;
}
```

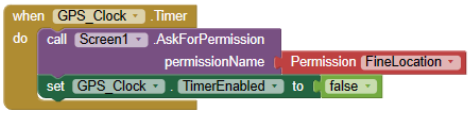
Mobile App:

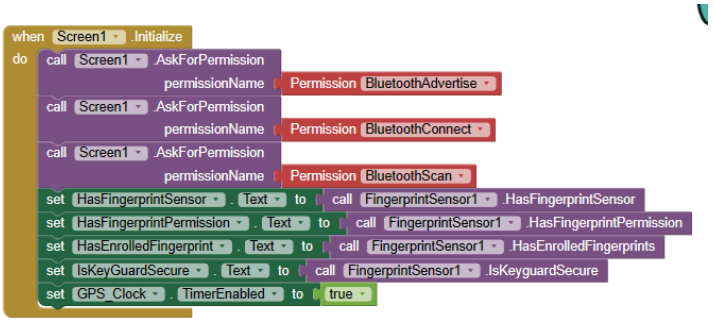
- The app was created using MIT App Inventor which has inbuilt drag-and-drop blocks-based interface.
- The logic for sending data to the HC-05 Bluetooth module based on fingerprint detection was implemented using blocks in the 'Blocks' screen.
- The Fingerprint Sensor extension allowed detection and authentication of fingerprint input. The 'BlueTooth Client' component was used to establish connection and send data to HC-05 Bluetooth Module.

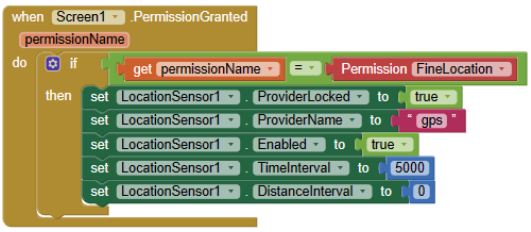


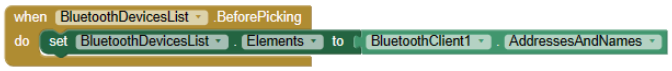
- When a correct fingerprint was detected, and the lock button was pressed, the app sent '1' to the bluetooth module, and when the unlock button was pressed, the app sent '0' to the bluetooth module. In case of incorrect fingerprint, no action was taken, and authentication error was shown.
- Along with this, the app also acquired GPS location through Location Sensor component, parsed it and sent the latitude and longitude values in real time through Bluetooth, , as and when the location is changed, which was further processed by the code running in Arduino to calculate the distance between lock and phone in real time.

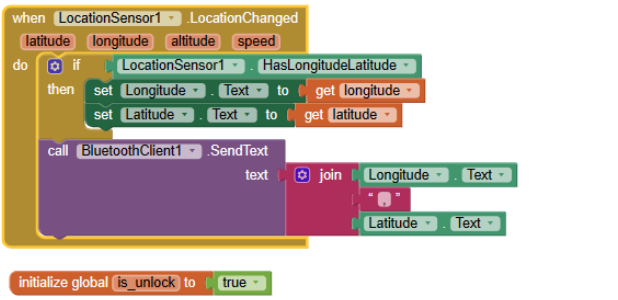
Code for the mobile app:

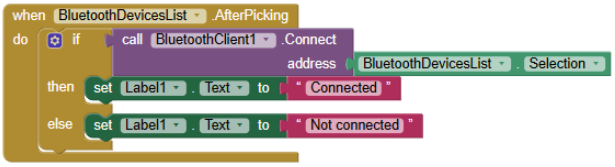


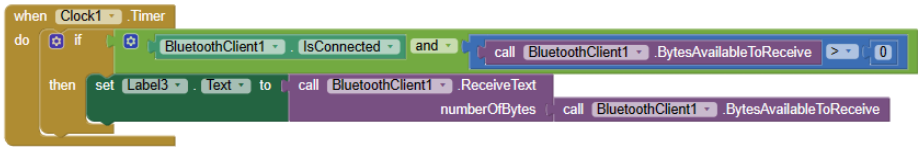


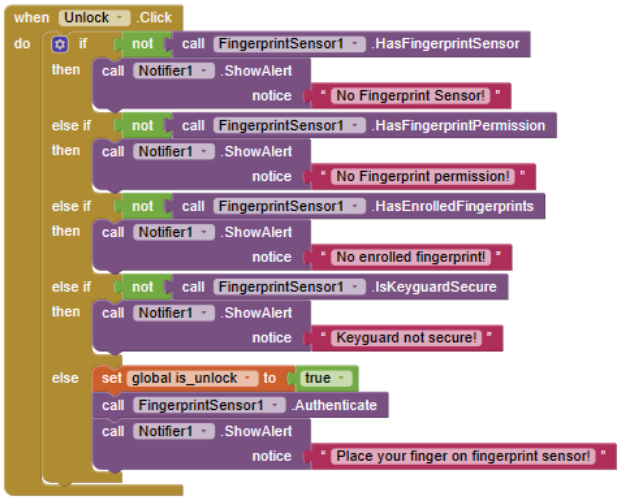


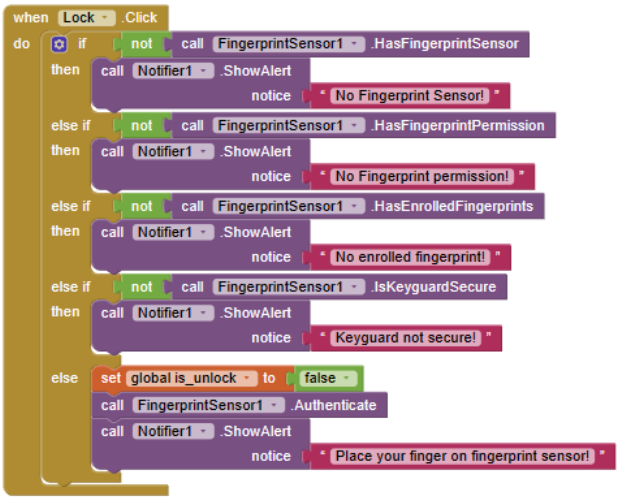


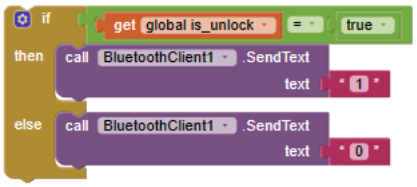


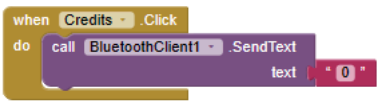


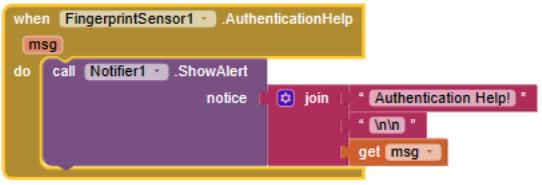


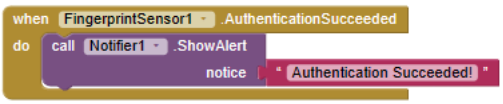












Testing & Workflow:

- Initially, we verified the functionality of the Arduino Nano by executing a blink code having 1000ms delay, provided as a sample in the Arduino IDE.

```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage
level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage
LOW
  delay(1000);                      // wait for a second
}
```

- Following this, we proceeded with the experimentation:
    - We integrated a potentiometer with the Arduino to exhibit resistance values.
    - The Solenoid Lock was evaluated by connecting it directly to a 12 V power source, resulting in its unlocking upon power supply.
    - To test the Relay module, we connected an LED with a resistor and programmed it to alternate the relay's ON and OFF states in a loop with a 5-second delay.
    - The HC05 Bluetooth module was assessed similarly, with an LED blinking in response to data from the Bluetooth. During compilation and upload, we ensured to disconnect the serial pins and reconnect them only after successful upload. The module responded correctly to commands sent from the mobile app.
    - We confirmed that the GPS module was receiving location perfectly, and was able to parse the data in NMEA format using TinyGPS++ library. However, sometimes the GPS module was not able to connect to the satellites when present indoors.

- With all modules confirmed operational, we constructed the entire circuit on a breadboard, supplied it with 12 V voltage, and uploaded the code onto the Nano. Subsequently, the Bluetooth module was paired with the mobile app, which was used for fingerprint scanning and recognition. Upon successful recognition, the door was locked and unlocked based on which button was pressed. Along with this the door got locked, if the mobile device went outside the lab i.e. when the distance got more than the threshold. Thus, our GPS + fingerprint-based door lock system was successfully implemented.

Results:

The Final project can be seen working in the following video -
https://drive.google.com/file/d/1GSwGyroT3FT8EvXCa-pMfBBxcZIPKLmT/view?usp=sharing