ECE 445

SENIOR DESIGN LABORATORY

DESIGN DOCUMENT

Design Document: Fingerprint Recognition Door Lock

Team Members

CHENGRUI WU
(cw70@illinois.edu)
HANGGANG ZHU
(hz66@illinois.edu)
HAORAN YUAN
(haorany7@illinois.edu)
LIZHUANG ZHENG
(lzheng17@illinois.edu)

TA: Guozheng He

Wednesday 27th March, 2024

Contents

1 Introduction

1.1 Problem and Solution Overview

In our Residential College, each student has own dormitory, and the dormitory door can only be opened by the student's own IC card from outside. Sometimes it is possible that you forget your IC card inside the dormitory room, or lost it somewhere by mistake, so that you must go to the front desk of the Residential College to get a temporary card, or go to the IC card service center to get a new card. And if it is in the midnight, it will be harder to get staff in touch. So it is better that students can use more methods to open the door except for swiping the IC card. We are thinking of other ways to unlock the door using other personal identification information. Even if you didn't lose your IC card, with more ways to open the door brings a little more joy in daily life.

Some popular way to unlock the door is password, facial recognition and fingerprint recognition. Considering the difficulty and portability, we decide to develop our own fingerprint recognition lock for our Residential College. However, replacing all the door lock in the Residential College is quite challenging. we propose a device which can be easily attached to the existing door lock, and turn it into a fingerprint recognition door lock, without assistance from the professional installation workers.

In addition to fingerprint recognition, we also intend to integrate other approaches to our smart door lock. Some basic functionalities include unlocking the door using Software App with the help of remote control through Wi-Fi. Besides, we will also apply Bluetooth technology to open the door lock automatically when the bonded mobile phone is approaching. In order to save energy, the device will turn into low energy mode, and we will add an infrared detection part to our device, which will wake up the device when people come back. Furthermore, we will try to implement more advanced features including unlocking the door through facial recognition, and voice recognition, which can make our device more convenient and intelligent. In general, we intend to develop a portable device with integrated ways to unlock doors, which can be managed easily through our mobile phone application and promise the security.

1.2 Visual Aid

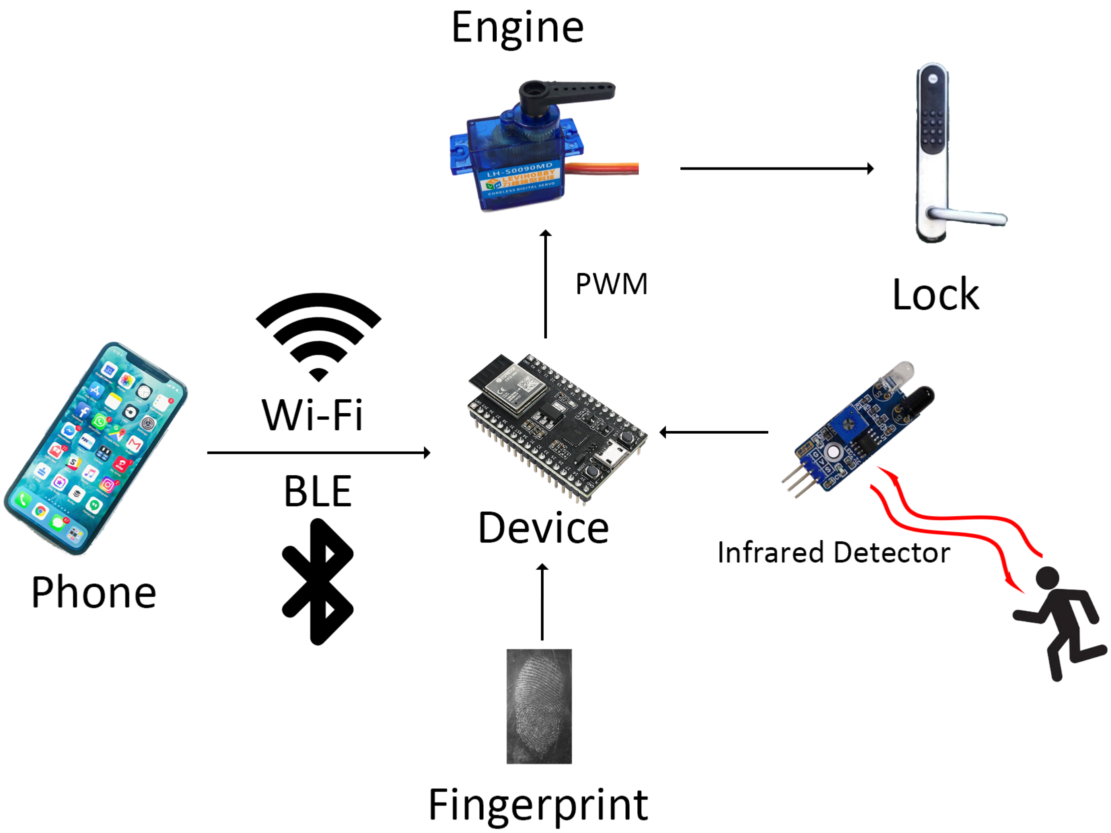

Figure 1: Visual Aid

1.3 High-level Requirements List

- Enable the authorized users to open the door lock using their fingerprints, the controller should be able to store at least 5 different fingerprints and the success rate should be above 80%. Besides, the infrared detector can wake up the device when people stand in front of the door within 1m ± 0.2m.
- Allow remote control using the software app with delay time of at most 5 seconds. And the BLE module of device can identify the neighboring mobile phones when approaching inside the range of 0.5m ± 0.1m.
- The mechanical subsystem can reliably open the door, the servo motor with a torque of at least 25 kg·cm is necessary.

2 Design

2.1 Block Diagram

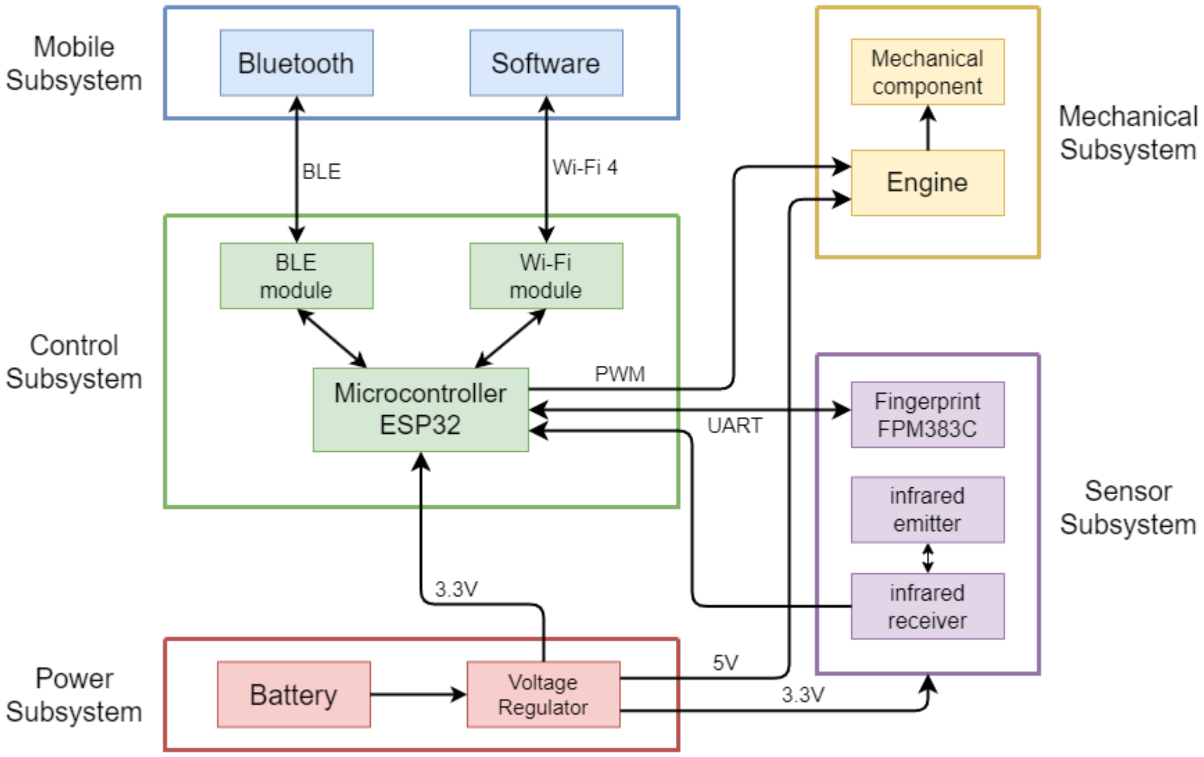

Figure 2: Block Diagram

Our device is divided into 5 subsystems: The Power Subsystem, the Sensor Subsystem, the Control Subsystem, the Mechanical Subsystem, and the Mobile Subsystem. The Power Subsystem contains a 12V lithium battery as the power supply, together with a set of AA batteries as backup batteries. The output voltages from the batteries will be regulated by the voltage regulator circuit in order to output a 3.3V voltage for the Control Subsystem and the Sensor Subsystem, and a 5V voltage for the servo motor in the Mechanical Subsystem. The Sensor Subsystem consists of a fingerprint recognition module FPM383C, and a set of infrared module including an infrared emitter and a receiver. The fingerprint recognition module compares the fingerprint received with the user fingerprint data, and communicates with the microcontroller through the UART protocol. The infrared module is set up with the concern of energy saving. When human passed by, the module wakes up the Control Subsystem and then make it to start working. The Control Subsystem uses ESP32 as the microcontroller, which accepts the signals from the

Sensor Subsystem and delivers a PWM signal to control the Mechanical Subsystem. In the Control Subsystem, ESP32 is also integrated with a Bluetooth at Low Energy (BLE) module and a Wi-Fi module, which allows the remote control from the software app in the Mobile Subsystem. The Mobile Subsystem contains a software app which communicates with the Wi-Fi module through the Wi-Fi 4 protocol, so that the user can control the door lock remotely. A server is also needed between the Wi-Fi module and the software. The cellphone with the Bluetooth function can also be seen as a part of the Mobile Subsystem, it uses BLE protocol to communicate with the BLE module, so the device can recognize the approaching mobile phones and open the door automatically. The Mechanical Subsystem contains a mechanical engine and some other mechanical components. The mechanical engine is basically a servo motor, which is controlled by the PWM signal from ESP32. For other components, a nylon thread connecting the motor with the door handle is used to pull the door handle down; and some brackets holding all the components are needed for attaching them on the door. So once the user approaches the door, the infrared module will detect the user and wake up the whole device, and then the device will wait for a signal to open a door, either from the fingerprint matching, or the remote control (Bluetooth or software app). Then the microcontroller will turn the command of opening the door into PWM signals, so that the servo motor can rotate to a particular angle, pulling the door handle with the attached nylon thread, and thus open the door.

2.2 Physical Design

For the physical design, we basically have 2 strategies, as shown in Figure 3 below. The first strategy consists of two brackets mounting on the door using bolts, considering the structure of the tiger clamp. One is above the door lock, holding batteries, voltage regulators, the microcontroller ESP32, the fingerprint recognition module and the infrared module. The other is installed below the door lock, holding the servo motor which can pull the door handle by a thread.

The second strategy consists of 3 brackets (boxes), one for the servo motor, one for the ESP32, batteries and the voltage regulator, and the rest one for the sensors. They are all sticking to the door by some strong adhesive tapes or glue. Wires go out of each box through some holes and connect each part together. Between the servo motor box and the box for the controller & power subsystems, there are two side boards touching the left and right side of the lock. They have two main functions. One is to limit the movement of the device when it is opposing the torque generated by the motor. The second function is for the path on which the wires can travel.

The size information should be further determined after more tests and experiments.

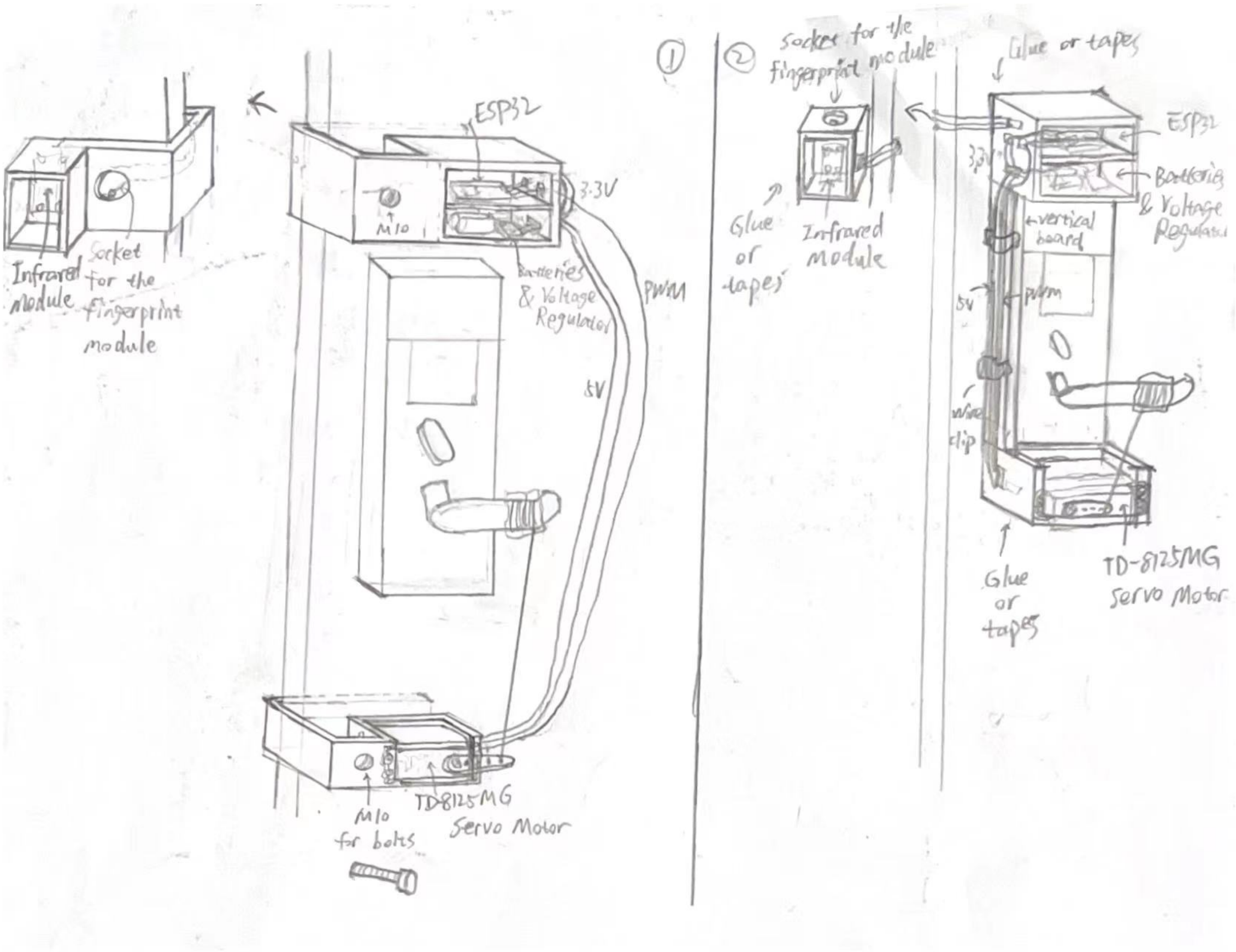

Figure 3: Physical Design

2.3 Controller Subsystem

2.3.1 Overview

This subsystem consists of a micro-controller that manages the operations of the device, including processing fingerprint data, controlling mechanical subsystem, and coordinating with the mobile app and BLE signal designed to interact with mobile phone. Besides, it will be waked up by infrared signal, which enables the device to sleep when people have left and to work when someone is approaching.

2.3.2 Requirements

We will choose an ESP32-WROOM-32 which involves Wi-Fi module and BLE module as the micro-control unit. For the sensor subsystem, it will communicate and manage the fingerprint sensor FPM383C by sending command and receive feedback through UART protocol (GPIO16&17); it will also be listening to the WAKE signal from the infrared detection module when it is sleeping through the RTC_GPIO pin (GPIO32). For the Mobile Subsystem, it will receive the message from server through Wi-Fi module when the user want to control the lock remotely; it will also evaluate the distance of connected device by reading the Received Signal Strength Indication (RSSI) of BLE signal and decide whether to open the door. For the Mechanical Subsystem, it will use PWM signal to control the behavior of the servo motor (GPIO4). The ESP32 microcontroller will be powered at +3.3V regulated by the power supply subsystem.

Requirements	Verification
1. Identify the strength of BLE signal when devices approach 0.5m ± 0.1m. 2. Can both receive and transmit over UART at a baud rate of 57600bps with FPM383C. 3. Interacting with the server with a delay of no more than 3 seconds.	1. (a) Establish a connection between the mobile phone and the ESP32 through the software app we developed. (b) Approach the ESP32 microcontroller from 1 meter away. Record the distance between them when the door was opened. Repeat this step for 3 times (c) Compare the distance we record with the target requirements (0.5m ± 0.1m) 2. (a) Run the code that set the baud rate of Serial2 (GPIO16&17) to 57600bps and send a command packet to FPM383C (we test it with the command that set LED to blue). (b) Connect Serial2 to FPM383C. (c) Ensure that Serial2 receive 0x00 (desired reply packet) and the color of LED is blue. 3. (a) Send data packet with the size range from 1byte to 1024bytes (b) Record the time we receive the reply packet.

Table 1: Requirements and Verification Table for the Controller Subsystem

2.3.3 Schematics

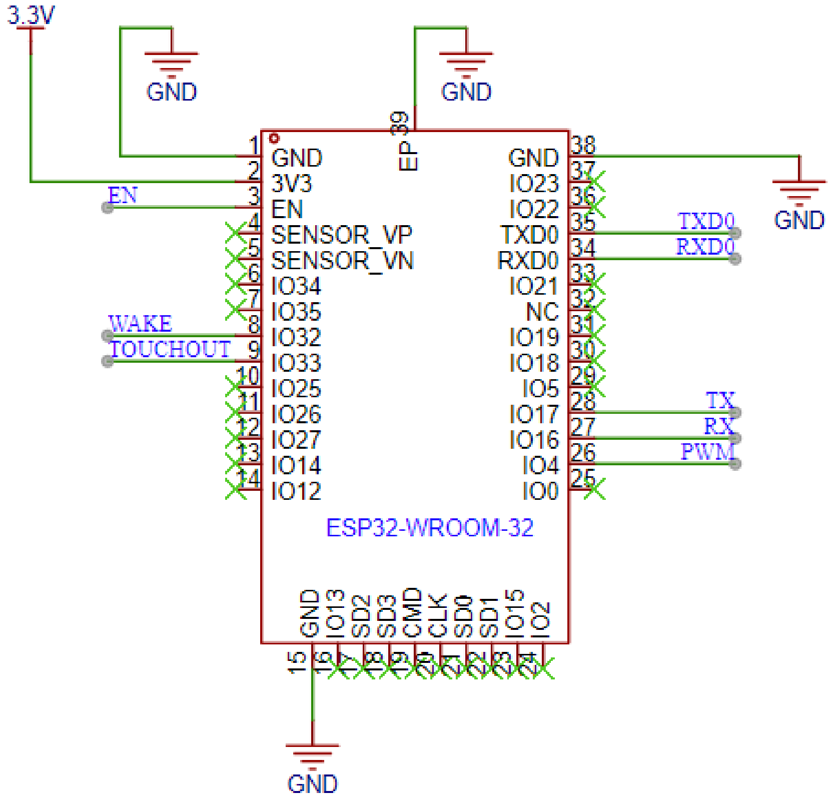

Figure 4: Microcontroller ESP32 Schematics

2.4 Sensor Subsystem

2.4.1 Overview

This component comprises an infrared detection module and a fingerprint recognition module. The infrared module signals the ESP32 board to activate the Bluetooth and WiFi modules when someone enters a predetermined range, allowing the system to conserve power by operating in a low-power state when unoccupied. The fingerprint module consolidates fingerprint scanning, storage, and identification into a single, efficient unit.

2.4.2 Requirements

This section encompasses both an infrared detection module and a fingerprint recognition module. The infrared detection module is designed to emit a signal to the ESP32 board, which activates the Bluetooth and WiFi modules upon detecting a person within a predefined proximity. This functionality ensures the system conserves energy by operating in a low-power state in the absence of individuals. We will design a circuit to generate wake up signal by precisely evaluating the approach of people. Note that the Figure 5 shown in the schematics section below is a preliminary design, more test need to be done in the future.

We utilize the FPM383C ideal fingerprint sensor as our chosen fingerprint recognition module. This module stands out due to its compact size and low power consumption, making it an ideal match for our battery-operated door lock system. It operates on a 3.3V power supply and communicates with our ESP32 micro-controller via the UART serial communication protocol. Designed with a touch-to-wake feature to minimize power usage, the module can store up to 60 fingerprint records in its flash memory. Besides, we will use AFC01-S06FCA-00 connector to connect the FPM383C and the PCB board.

Requirements	Verification
1. Able to wake up when people stand in front of the door within 1m ± 0.2m. 2. Able to collect and store the fingerprint of at least 5 users. 3. Identify the user fingerprint in less than 2 seconds after touching, with a minimum accuracy of 80%. 4. Can reply the command packet over UART at a baud rate of 57600bps.	1. (a) Approach the infrared detector from 2 meter away. Record the distance when the ESP32 is awoke. Repeat this step for 3 times (b) Compare the distance we record with the target requirements (1m ± 0.2m) 2. (a) clear the fingerprint storage in FPM383C. (b) Register 8 different fingerprint. (c) Test the above 8 fingerprint. 3. (a) Register a new fingerprint. (b) Press the sensor in different angles (0,±45°,±90°). (c) Record the result and calculate the accuracy. 4. (a) Run the code that set the baud rate of Serial2 (GPIO16&17) to 57600bps and send a command packet to FPM383C (we test it with the command that set LED to blue). (b) Connect Serial2 to FPM383C. (c) Ensure that Serial2 receives 0x00 (desired reply packet) and the color of LED is blue.

Table 2: Requirements and Verification Table for the Sensor Subsystem

2.4.3 Schematics

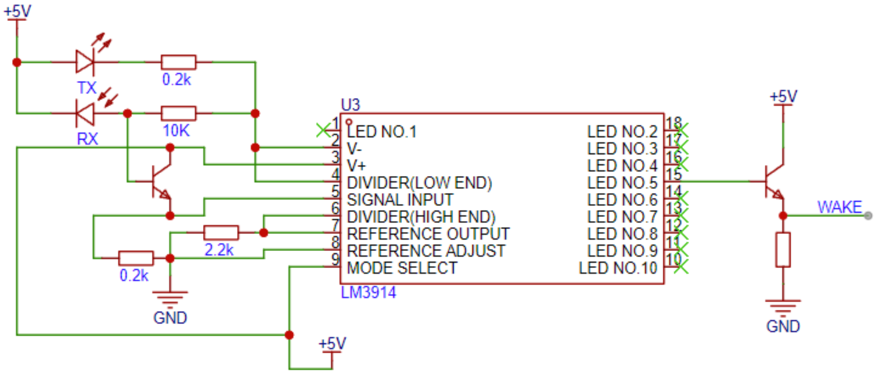

Figure 5: Infrared Module Schematics

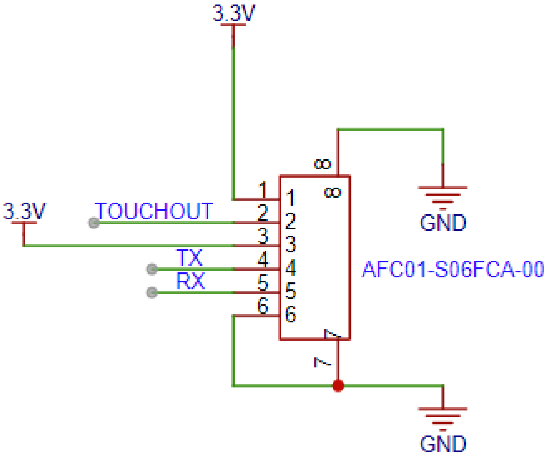

Figure 6: Fingerprint Recognition Module Schematics

2.5 Mobile Subsystem

2.5.1 Overview

Mobile Application End It's a software app containing a front-end and a back-end. The front-end will be a simple app page on the phone with buttons which allow users to unlock the door, add new fingerprints or delete fingerprints. It also contains a simple user login page. The software also provides one-to-one control to every device. The back-end uses Bluetooth to communicate with ESP32. Specifically, ESP32 will act as the client and phone will act as the server and they will communicate using RFCOMM channel[1]. Bluetooth protocol will also be implemented to connect to the BLE module of ESP32. The mobile software app is intended to be working on Android 12+.

If possible, our software app will also communicate with FPM383C fingerprint module, with a database containing data of fingerprint. It enables users to query/add/delete fingerprints through mobile phones.

Server Application End The server app, hosted on a cloud platform like Azure, facilitates secure communication between mobile devices and door locks. Developed with Flask in Python, it starts by enabling direct interaction between a single mobile and door lock, laying the groundwork for more complex functionalities. It scales up to manage multiple devices and locks through a sophisticated database that supports user registration and identity verification, ensuring personalized access.

Containerized with Docker for streamlined deployment, the app emphasizes robust security measures for data protection and secure connections, catering to the evolving needs of a connected ecosystem. This app provides a secure, scalable, and user-friendly solution for mobile-device-to-door-lock interactions.

2.5.2 Requirements

Mobile Application End This subsystem mainly interacts with the Controller Subsystem, as it provides a connection between the user and our device. Specially, we will use Wi-Fi and Bluetooth API in our back-end of ESP32 to interact with the controller. Espressif provides detailed API documentation for us to achieve wireless communication.[2]. For the front-end of our software, we will use some widely used framework such as React and Vue. The Bluetooth is used to connect to the BLE module of ESP32. The challenging part of this subsystem is to maintain a stable and secure connection between our software and our device.

A more detailed flowchart of how mobile software will interact with ESP32 is described in Figure 7. The mobile software will first handle all the jobs of establishing Bluetooth connection: setting up Bluetooth, scanning Bluetooth, pairing Bluetooth and connecting Bluetooth. These are the prerequisites for later communication. Once connection is established, mobile phone and ESP32 will have a connected BluetoothSocket and they will communicate using **InputStream** and **OutputStream**[3]. Data are read from and written into stream using **read(byte[])** and **write(byte[])**[3]. The communication protocol is also described in Figure 7. Note that it should not take too much time for the protocol to be established (within 6 seconds). In case of any accidents, after mobile software sends *UNLOCK* signal but does not receive *FINISH* signal in 10 seconds, the system is considered to fail and should report error. Additionally, in order to save power, Bluetooth connection will close automatically 60 seconds after user is not in front of door (no infrared signal) any more.

Server Application End The server application operates on a secure verification model to process requests from sources, such as mobile devices, to a destination, like door locks. Requests are queued in the Input Queue, where each request is identified by a unique ID and accompanied by a security token.

Upon a request's turn for processing, the app consults the Tokens & IDs Database to verify two critical aspects:

1. **Existence of ID**: The app checks if the ID provided in the request is recognized within the system's database. If the ID does not exist, the app rejects the request and sends a failure notification back to the source.
2. **Token Validation**: For IDs that are found in the database, the app checks if the corresponding token matches the one stored. If there is a mismatch, indicating an invalid or expired token, the request is not processed further, and a failure message is sent to the source.

Only when both the ID is verified and the token matches does the server app forward the request to the destination. Successful requests are then placed in the Output Queue, awaiting actions such as unlocking a door. This system ensures that only authenticated requests from legitimate sources are processed, maintaining the integrity and security of the server's operations.

Requirements	Verification
1. Sent packets and received packets are the same and transmitted within 0.1 second when mobile phone and ESP32 are 5 meters away. 2. Communication protocol is finished within 3 seconds. 3. Maximum waiting time for unlocking door should be 3 seconds and device is disabled after 60 seconds without use.	1. (a) Send data of different sizes (1 byte to 1024 bytes) and check that it remains the same on received side. (b) Write a test program to record message round trip time (RTT). Check if RTT is smaller than 0.2 seconds at a distance of 5 meters. 2. (a) Record time difference between *INIT* signal and *FINISH* signal for many times and compute the average. Make sure the average is within 6 seconds. 3. (a) Manually disable the motor and check that error message is sent on the mobile software after 10 seconds. (b) 60 seconds after door is unlocked, check that Bluetooth connection between phone and ESP32 is closed.

Table 3: Requirements and Verification Table for the Mobile Application

Requirements	Verification
1. Direct Interaction: The server app must enable secure communication between a single mobile device and one door lock within 2 seconds of request initiation. 2. Scalability: The app should support interactions with multiple mobile devices and door locks simultaneously, with no degradation in performance. 3. User Registration and Identity Verification: The database must support user registration and validate identities in under 5 seconds to ensure personalized access. 4. Secure Connection: All communications between mobile devices and door locks must be encrypted and secure. 5. Token and ID Validation: The server must validate tokens and IDs within 1 second before processing any requests.	1. Direct Interaction Verification: Measure the time taken from request initiation to door lock activation for a single mobile device and verify that it is within the required 2-second limit. 2. Scalability Verification: Conduct stress tests by increasing the number of mobile devices and door locks connected to the server both to 4 to ensure stable performance and reasonable response time. 3. User Registration and Identity Verification Testing: Simulate user registration processes and verify that the system authenticates identities and provides access within the 5-second threshold. 4. Secure Connection Assessment: Perform security audits to confirm that the encryption standards are met and that all data transmissions are secure. 5. Token and ID Validation Assessment: Implement automated tests that check the server's response times for token and ID validation to ensure they meet the 1-second requirement.

Table 4: Requirements and Verification Table for the Server Application

2.5.3 Flow Charts

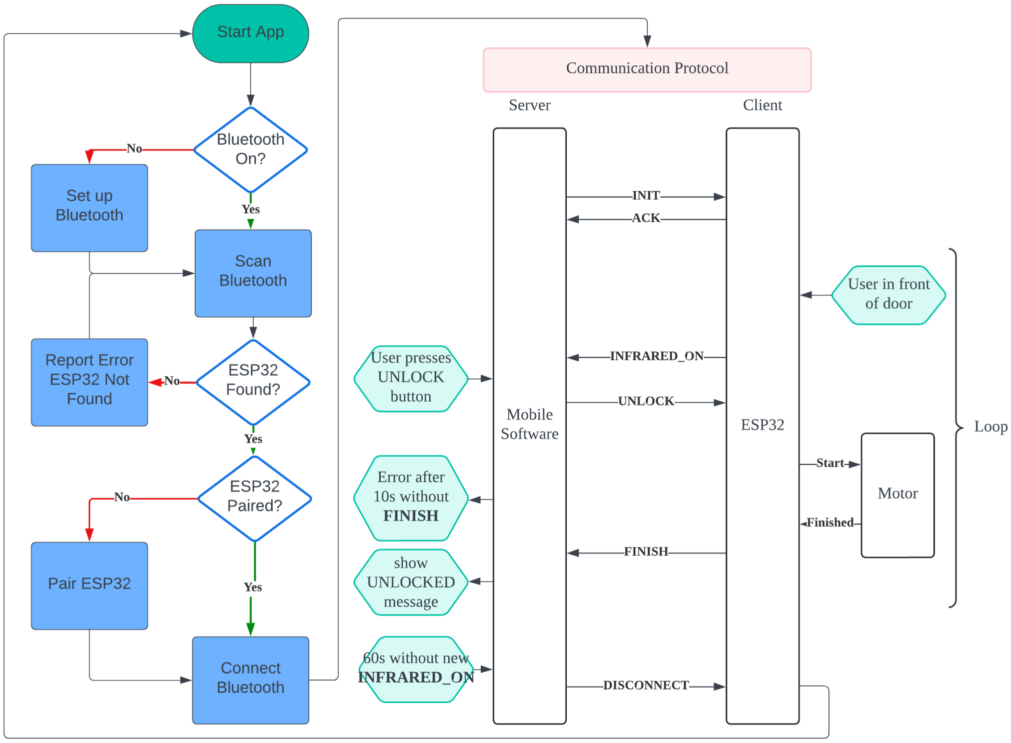

Figure 7: Mobile Software Application and its Communication with Control System

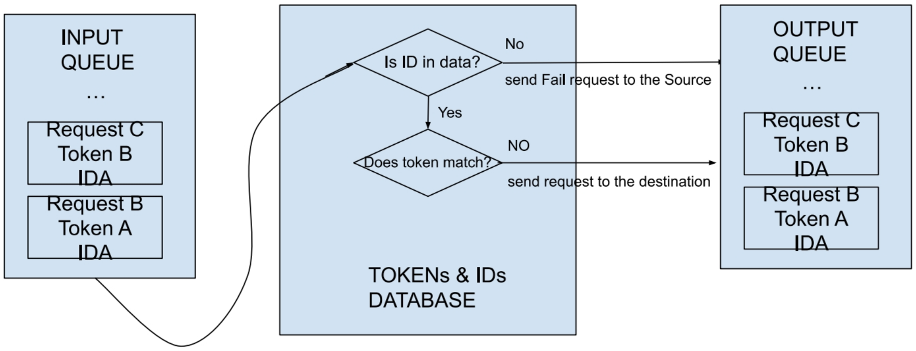

Figure 8: Server Application Flowchart

2.6 Mechanical Subsystem

2.6.1 Overview

The Mechanical Subsystem consists of some brackets, a servo motor and an mechanical actuator which can push or pull the handle of the door lock from the inner side and thus open the door. The design sketch is shown in Figure 3, section 2.2. This subsystem is installed near the lock, inside the door. The servo motor is directly wired with the Controller Subsystem, and accepts PWM signal from the micro-controller as a trigger. Then the servo motor drives the actuator, and the actuator can push or pull the door handle inside, to complete the action of opening the door.

2.6.2 Requirements

The servo motor are used to drive the actuator when it gets the signal to open the door. It is wired with the micro-controller, and uses PWM protocol to contact with the micro-controller. So it can get a PWM signal from the Controller Subsystem as a trigger. When the PWM signal requests to open the door, the servo motor will turn a particular angle and drive the actuator to move to some extent. By estimation, our door handle needs around 25N to open, consider the arm of force in about 8∼9cm, we need a servo motor with its torque above 25kg·cm. TD-8125MG digital servo motor can be a candidate. However, due to the limitation of the installation place, voltage supply, force application, we had better to use a servo motor with larger torque, which can provide larger amount of redundancy.

The actuator is directly contacted with the door handle. There are many possible designs for the actuator. One possibility is a hammer hanging above the handle inside, when the servo motor moves, the hammer can drop down and push the door handle to the appropriate angle so that we can open the door from the outside. Another strategy is to use a nylon thread or a strong hemp rope to pull the handle down from the inside, thus the servo motor should be installed at the bottom of the lock.

Requirements	Verification
1. Some brackets for holding all the device components strongly on the door. 2. A powerful enough servo motor to drive the mechanical actuator, approximately with its torque larger than 25kg·cm. 3. A reliable mechanical actuator with links strong enough to move the door handle to a angle of at least 45°.	1. (a) Install the brackets on the door, apply a force $\geq 25N$ on each bracket continuously for 3 minutes, test if the bracket can resist this impulse. (b) Put 500g weights as loads on each bracket, leave them for a day, test if the bracket can still connect reliably with the door. 2. (a) Use the PWM to drive the servo motor, then use the force gauge to test whether the motor can produce more than 25kg force at 1cm away from the axis. (b) Use the PWM to drive the servo motor, then use the nylon thread to pull the handle down, test if the motor can open the door by pulling the nylon thread. (c) Modify the position of the motor, angle of pulling, voltage input, etc. to ensure that we can open the door reliably. 3. (a) Use different materials of threads, like nylon, cotton, or hemp ropes, knot them tightly with the handle and the motor. (b) Use the PWM to drive the servo motor, and pull the handle, measure the angle using a protractor when the motor is stuck, see if it reaches 45°.

Table 5: Requirements and Verification Table for the Mechanical Subsystem

2.6.3 Schematics

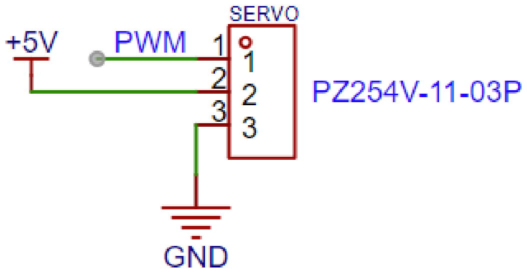

Figure 9: Servo Motor Schematics

2.7 Power Subsystem

2.7.1 Overview

The Power Subsystem contains a battery set and some voltage regulators. It is used to power up our Controller Subsystem, Mechanical Subsystem and Sensor Subsystem. We plan to use a 12V lithium battery as the power source. And in addition, a set of AA batteries can be used as the backup power source. The voltage regulators will regulate the voltage to different levels so that we can power up different parts of the device.

2.7.2 Requirements

The Power Subsystem is the crucial part for powering up other subsystems, it should provide stable power to support the normal work of the entire device. The power can be provided by a 12V battery, which can be easily recharged or replaced when it dies out. Then for the voltage regulator circuit, we can use some adjustable voltage regulators to produce different stable voltage output, like LM317 3-terminal adjustable voltage regulator[4]. The micro-controller and the fingerprint recognition subsystems need to work under 3.3V, but the 25kg·cm servo motor should work in about 4.8∼7.2V, and the ones with larger torque may need higher voltage. The voltage regulators should provide stable voltage for them and ensure that they can work normally.

Requirements	Verification
1. Provide at least 200mA, stable 3.3V power supply for the Controller Subsystem and the Sensor Subsystem. 2. Provide at least 200mA, stable power supply in the range of 4.8∼7.2V for the servo motor to work normally. 3. Can automatically switch to the backup batteries when the main battery dies out.	1. (a) Use the voltmeter to measure the output voltage of the voltage regulator, see if it is a stable 3.3V output. (b) Connect the load (the microcontroller and the sensors), use the ammeter to measure the output current, for both when they are working or sleeping. 2. (a) Use the voltmeter to measure the output voltage of the voltage regulator, see if it is a stable output in 4.8∼7.2V. (b) Connect the load (the servo motor), use the ammeter to measure the output current, for both when the motor is working or sleeping. 3. (a) Use an adjustable constant current voltage source to sweep the power from 12V down to 0, in order to simulate the situation where the main battery dies out. (b) Monitor the output voltages using the oscilloscope, see whether the output voltages are still stable, one in 3.3V, another in 4.8∼7.2V.

Table 6: Requirements and Verification Table for the Power Subsystem

2.7.3 Schematics

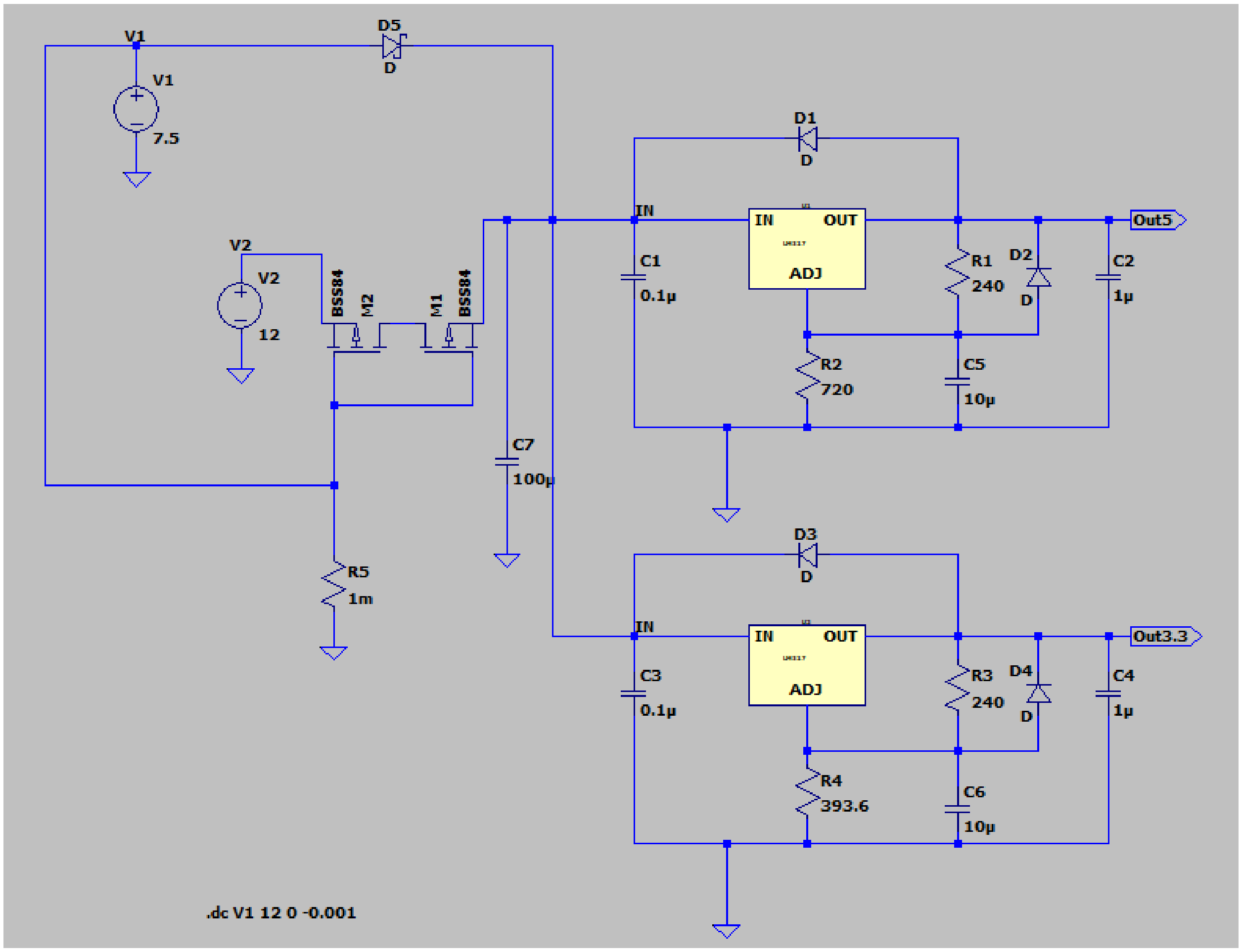

Figure 10: Voltage Source Switching Circuit and Voltage Regulator Circuit

Figure 10 shows the basic circuit design for the voltage regulators and the voltage source switching circuit. In this schematics, the left part is the voltage source switching circuit, V1 is the main lithium battery, V2 is the backup AA battery set. D5 is a Schottky diode, which has a lower forward voltage drop (about 0.15∼0.45 V) and a faster switching action, very suitable for switching the voltage source. M1 and M2 are two BSS84 PMOS transistors, they are conducted when the gates (G) are in low voltage levels. M1 is used to avoid V1 from recharging V2 (equivalent to a Schottky diode), M2 is used as the switch for V2. When the main source V1 dies out, its voltage will decrease, and then the voltage level of the gate (G) will be lowered, for M2, $V_{GS} < 0$ and then the PMOS transistors will conduct, so that the backup battery set V2 will take over and continue to supply power. Besides, $R_5 = 1M\Omega$ is to avoid short circuit of V1, C7 is for filtering.

For the right side of the circuit, we use the typical 3-terminal linear voltage regulator circuit built by LM317 as the main part of our voltage regulator. The circuit can be found in page 11 of [4]. The output voltage V_{Out} of LM317 can be calculated as shown below [4]:

$$V_{Out} = V_{Ref}(1 + \frac{R_2}{R_1}) + (I_{Adj} \times R_2) \tag{1}$$

I_{Adj} is typically 50 μA and negligible in most applications.[4] Since there is a 1.25V offset input in the adjust terminal, we have $V_{Ref} = 1.25V$[4], so the output voltage is approximately

$$V_{Out} = 1.25V \times (1 + \frac{R_2}{R_1}) \tag{2}$$

Here we want the output be $V_{Out} = 5V$ and $3.3V$, so we use two voltage regulator circuits with the pin out "Out5" and "Out3.3." The data sheet [4] recommends us to use $R_1 = R_3 = 240\Omega$, so according to Equation 2, we can solve that $R_2 = 720\Omega$ and $R_4 = 393.6\Omega \approx 400\Omega$. What is more, if we want a output of 7.2V, we need $R_2 = 1142.4\Omega$. C1 is the filter capacitor, C2 is used to improve the transient response, C5=C_{Adj} is used to improve the ripple rejection.[4] There are also two diodes in each voltage regulator circuit for protection. They both provide a low-impedance path for the capacitors to discharge, so that they can prevent them from discharging into the output port of the regulator. D2 is for C5, and D1 id for C2.[4]

2.7.4 Simulation

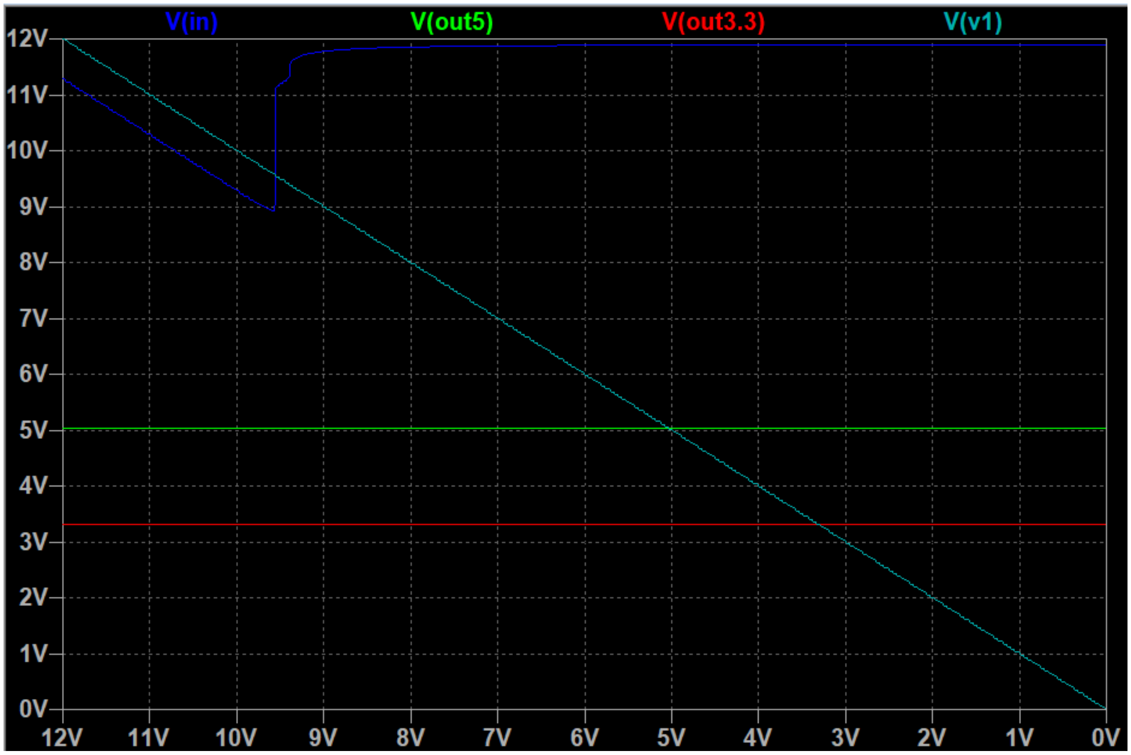

Figure 11: Simulation: V1 Sweep from 12V to 0

The simulation in LTSpice shows that, when we sweep the voltage of the main battery V1 from 12V down to 0, the input voltage V_{in} first decreases together with V1. Then at about 9.5V, the circuit successfully switched to the backup battery set, so that V_{in} returns to 12V. In this process, the 5V and 3.3V output are both stable. In fact, as long as the output needed is smaller than about 9V, this circuit will output two stable regulated voltages.

2.8 Tolerance Analysis

The common reason for failing to open the door might be that the torque provided by the servo motor was insufficient to overcome the limiting friction. To ensure the selection of a motor with adequate torque to smoothly rotate the door handle, we utilized an electronic force gauge from the lab to measure the forces involved in rotating the handle. We operated the force gauge slowly to mitigate any experimental errors caused by acceleration. As shown in the figure below, the data indicates that the peak force is approximately 24 N and the force stabilizes at 16 N, indicating that the maximum force we need is around 24 N. Considering that the distance from the attachment point to the axis of rotation is 8.5 cm, and using the formula

$$\tau = \vec{r} \times \vec{F} \tag{3}$$

we calculated the required torque to be

$$\tau = \frac{24\text{N} \cdot 8.5\text{cm}}{9.8\text{N/kg}} = 20.8\text{kg} \cdot \text{cm} < 25\text{kg} \cdot \text{cm}. \tag{4}$$

Consequently, we selected the TD-8125MG digital servo motor with a nominal torque of $25\text{kg} \cdot \text{cm}$ to fulfill our requirements.

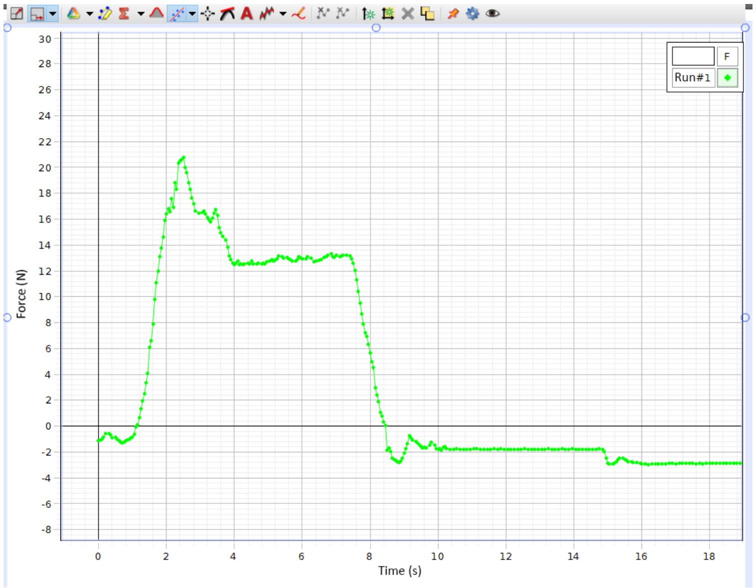

Figure 12: Force-Time Plot for Opening the Door

3 Cost and Schedule

3.1 Cost Analysis

3.1.1 Labor

We assume that each of us deserves ￥200 per hour of work, and each of us works 10 hour per week. The project takes about a semester (12 weeks) to complete, so the reasonable salary for each of us is:

$$\frac{\text{￥}200}{\text{hour}} \cdot \frac{10\text{ hour}}{\text{week}} \cdot 12\text{ weeks} \cdot 2.5 = \text{￥}60,000 \tag{5}$$

So for all 4 of us, our labor cost is about

$$4\text{ persons} \cdot \text{￥}60,000\ /\text{person} = \text{￥}240,000 \tag{6}$$

3.1.2 Parts

Part #	Mft.	Description	For	Price	Qty.	Total
TD-8125MG	Tiankongrc	25kg·cm digital servo motor	Mech.	￥55	1	￥55
ESP32	Espressif	Micro-controller	Ctrl.	￥22	1	￥22
FPM383C	Hi-Link	Fingerprint sensor	Sensor	￥22	1	￥22
LM317	Texas Instruments	3-Terminal Adjustable Regulator	Power	￥1.3	2	￥2.6
18650	Doublepow	12V lithium battery set	Power	￥35.8	1	￥35.8
LM3914	Texas Instruments	Dot/Bar Display Driver	Sensor	￥3	1	￥3
PCBs		PCB customization	Power & Ctrl.	￥10	2	￥20
Total	-	-	-	-	-	￥160.4

Table 7: Costs for Parts

3.1.3 Sum of Total Costs

The grand total cost is

$$\begin{aligned}\text{Total Costs} &= \text{Labor Costs} + \text{Parts Costs}\\ &= \text{￥}240,000 + \text{￥}160.4\\ &= \text{￥}240,160.4\end{aligned} \tag{7}$$

3.2 Schedule

Week	Tasks	Member
Jan 15, 2024	Propose the project.	All
Jan 22, 2024	Do research on the decided project.	All
Feb 26, 2024	Learn ESP32 and its modules.	Chengrui Wu
	Learn Wi-Fi, Bluetooth and ESP32.	Hanggang Zhu
	Learn fingerprint module	Haoran Yuan
	Force analysis and choosing components.	Lizhuang Zheng
Mar 4, 2024	Choose ESP32 module type.	Chengrui Wu
	Develop a toy Android App.	Hanggang Zhu
	Choose fingerprint sensor.	Haoran Yuan
	Servo motor selection, components purchasing.	Lizhuang Zheng
Mar 11, 2024	Build ESP32 develop environment and test Serial, PWM generation function of ESP32.	Chengrui Wu
	Test connection between Software and ESP32 using Bluetooth.	Hanggang Zhu
	Implement a toy backend for HTTP requests.	Haoran Yuan
	Servo motor rotation testing and torque testing.	Lizhuang Zheng
Mar 18, 2024	Test Wi-Fi module and Bluetooth module of ESP32.	Chengrui Wu
	Test connection between Software and ESP32 using Wi-Fi.	Hanggang Zhu
	Learn background knowledge for docker.	Haoran Yuan
	Power regulator circuit design and software simulation.	Lizhuang Zheng
Mar 25, 2024	Test the connection between ESP32 and FPM383C.	Chengrui Wu
	Test connection between Software and FPM383C.	Hanggang Zhu
	Rent a cloud service in Azure other cloud service provider and prepare a usable URL. Start writing server app with flask in python.	Haoran Yuan
	Mechanical part design.	Lizhuang Zheng

Week	Tasks	Member
Apr 1, 2024	Design the FSM of ESP32.	Chengrui Wu
	Test connection bugs.	Hanggang Zhu
	Implement the basic function of handling HTTP requests communication between one mobile and one door lock. Containerization via docker and deploy on cloud.	Haoran Yuan
	3D printing and mechanical part testing.	Lizhuang Zheng
Apr 8, 2024	Integrate Wi-Fi ,BLE together to control FPM383C and the open of the door.	Chengrui Wu
	Build database for recording of fingerprint.	Hanggang Zhu
	Implememt sub-function that supports multiple mobile phone and multiple door locks. Build a database system that supports user registration and personal identity verification.	Haoran Yuan
	Power regulator PCB design and fabrication.	Lizhuang Zheng
Apr 15, 2024	PCB design and infrared detecttion module test.	Chengrui Wu
	Build software front-end.	Hanggang Zhu
	Implement secure connection function and complete other security consideration.	Haoran Yuan
	Components combination and debugging.	Lizhuang Zheng
Apr 22, 2024	Test the stability of the microcontroller.	Chengrui Wu
	Test bugs on software.	Hanggang Zhu
	Further test and reliability check.	Haoran Yuan
	Test bugs on mechanical part and PCB part.	Lizhuang Zheng
Apr 29, 2024	Ensure the connection to other components	Chengrui Wu
	Fix remaining issues if any and improve performance of software.	Hanggang Zhu
	Ensure the connection to other component	Haoran Yuan
	Test and fix remaining bugs, check the power and signal reliability.	Lizhuang Zheng
May 6, 2024	Finalize the design, prepare for the demo.	All

Table 8: Schedule

4 Ethics and Safety

4.1 Ethics

We intend to do experiments on doors inside our campus Residential College, which means we need approval from Residential College. IEEE Code of Ethics request avoiding real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist.[5] So we need to corporate well with Residential College and if we can't reach an agreement, we should try to do experiments on doors which are allowed.

ACM code of ethics[6] requests that we should avoid harm. While our project doesn't involve any organic living things, we still need to do experiments on doors. If not dealt in an appropriate way, we may cause damage to the door, which include stressing too much weight on the door knob, opening or closing the door in a rude way etc. In all cases, we should take full consideration before we start to do any experiments.

ACM code of ethics[6] requests that we should be honest and trustworthy. Our device's basic functionality includes unlocking a door using fingerprint and phone. We can achieve fingerprint recognition using specialized fingerprint sensors but we should achieve that using a simple touching sensors, which means not everyone can unlock the door. Also, for the remote control, we can use Wi-Fi, Bluetooth, SIM card or any other way. But in any cases, we must be transparent about our ways to achieve remote control.

4.2 Safety

As our fingerprint recognition door lock system requires electricity to work, we must pay attention to the usage of battery and follow ECE 445 safety guideline: Any group charging or utilizing certain battery chemistries must read, understand, and follow guidelines for safe battery usage.[7]. Battery is necessary for our project as we need to attach our device to a door. As our device is intended to be small and not having too much weight, we will use lithium batteries. However, lithium batteries are substantially more flammable.[8]. So we must pay attention to the usage of battery and we will use charger from laboratory to charge the battery. All team members have attended fire extinguisher training and there's fire extinguisher in our Residential College. In anyway, when we do experiments inside laboratory or on doors, we must be careful with the battery problem.

References

[1] Google. "Bluetooth overview." (2024), [Online]. Available: https://developer.android.google.cn/develop/connectivity/bluetooth?hl=en (visited on 03/27/2024).

[2] ESPRESSIF. "Wifi api – arduino esp32 latest documentation." (2024), [Online]. Available: https://docs.espressif.com/projects/arduino-esp32/en/latest/api/wifi.html (visited on 03/07/2024).

[3] Google. "Transfer bluetooth data." (2024), [Online]. Available: https://developer.android.google.cn/develop/connectivity/bluetooth/transfer-data?hl=en (visited on 03/27/2024).

[4] Texas Instruments, *LM317 3-Terminal Adjustable Regulator*, LM317, Revised Apr. 2020, Sep. 1997.

[5] IEEE. "IEEE Code of Ethics." (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 03/07/2024).

[6] ACM. "ACM Code of Ethics and Professional Conduct." (2018), [Online]. Available: https://www.acm.org/code-of-ethics (visited on 03/07/2024).

[7] Course Staff. "ECE 445 safety guidelines." (2024), [Online]. Available: https://courses.grainger.illinois.edu/ece445zjui/guidelines/safety.asp (visited on 03/07/2024).

[8] Course Staff. "Safe Practice for Lead Acid and Lithium Batteries." (2016), [Online]. Available: https://courses.grainger.illinois.edu/ece445zjui/documents/GeneralBatterySafety.pdf (visited on 03/07/2024).