# Final Report

## Project Name

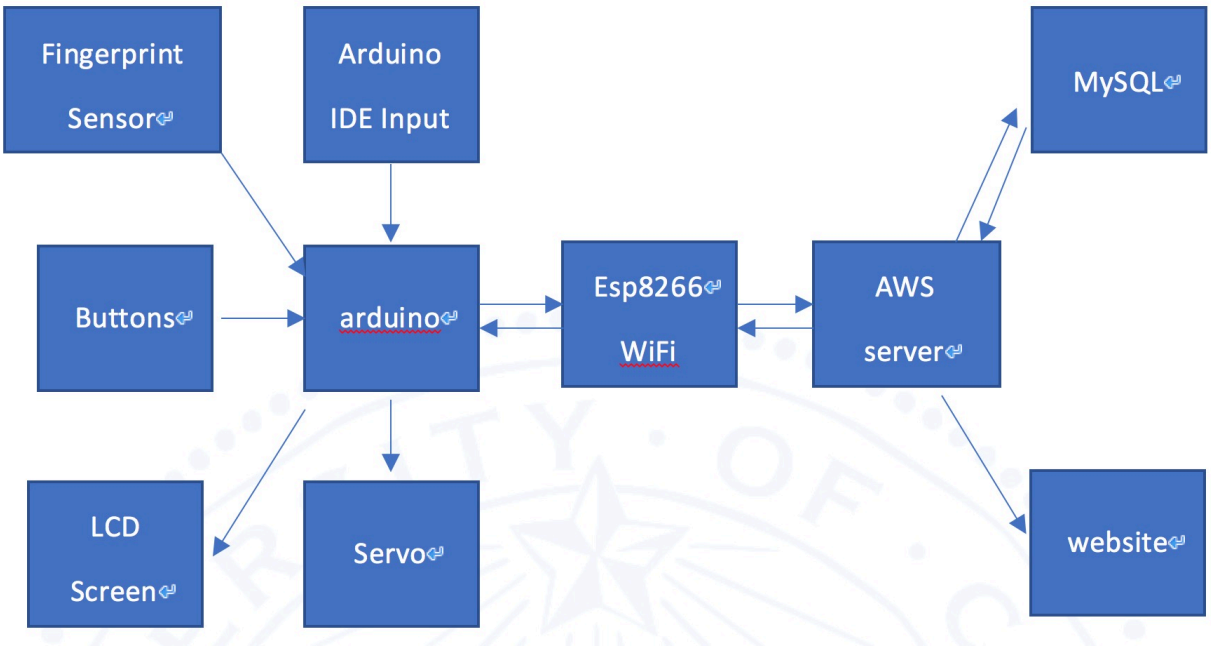
Fingerprint Lock System

## Group Member:

Member1: Wenbo Zhao 75887267

Member2: Xiaowei Tan 69203272

## Overview




Based on the procedure above, we developed our fingerprint lock system. To make it more similar to a practical lock. We define three mode for our system. One is the ENROLL_MODE, the other is the DETECT_MODE, the third is the IDLE_MODE. To switch between the three mode, we use two buttons. If we don't press any button, the mode is IDLE_MODE. If we press first button, it switches to ENROLL_MODE and guide the user to enroll a fingerprint, then it will switch back to IDLE_MODE. If we press the second button, it switches to DETECT_MODE and let the user try to open the lock and then switches back to IDLE_MODE.

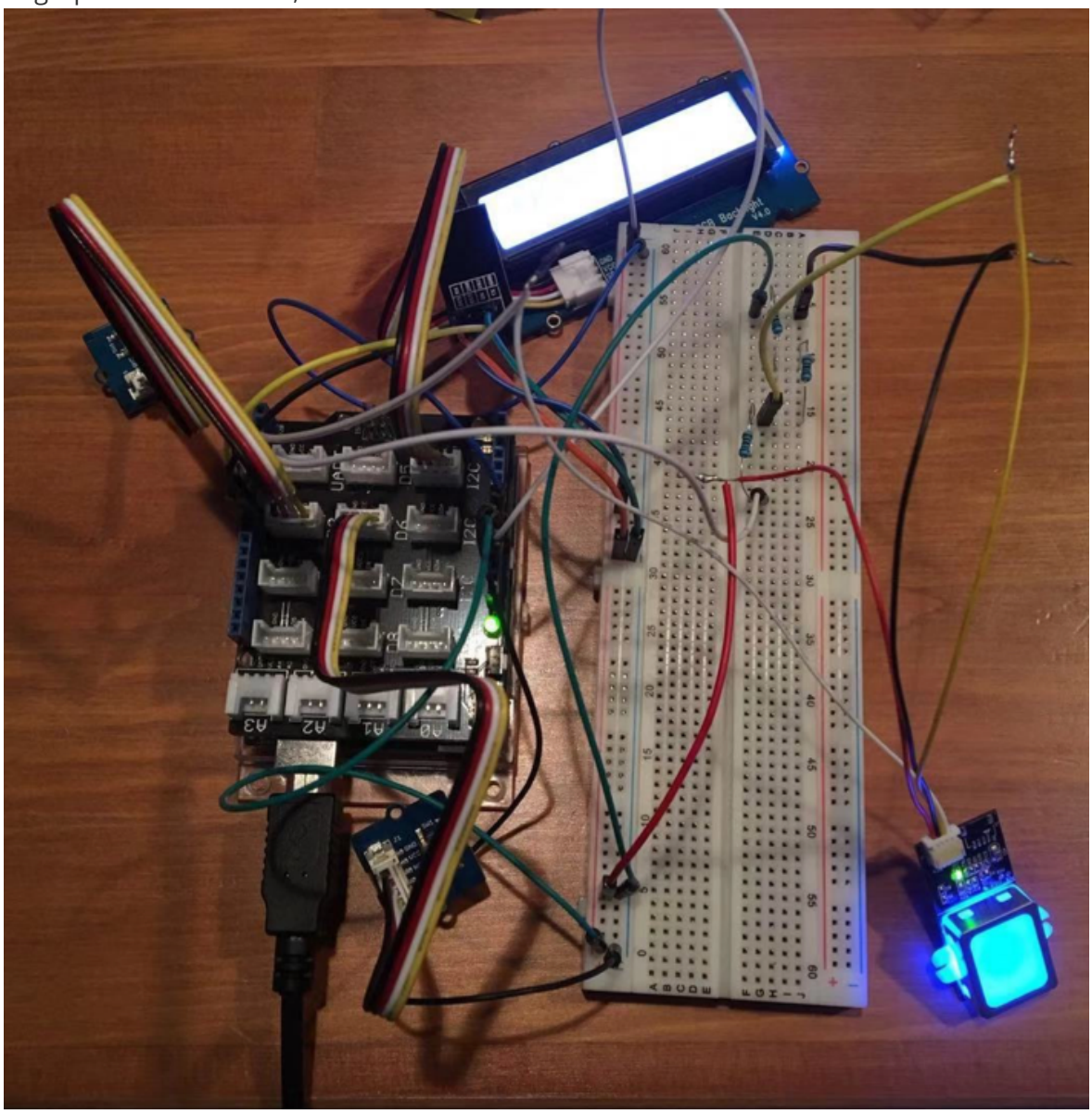
Although the fingerprint sensor itself has the database and can implement a basic prototype, it can't support info aggregation and intuitive display. Which means that our lock system saves comprehensive info like user name, fingerprint enrolled time except for the fingerprint.

Also, when someone open the lock, the LCD screen will show “<somebody’s name>, welcome back!” by accessing the AWS. The other amazing function is that our system will record ever log info, which can help the admin check who have enrolled the finger and who open the lock. That makes the whole system more humanized.

## Front End

- Circuit Connection

  Circuit connections is a pretty difficult part in this project, since we have many sensors and actuators in this project. What need to be noticed is that the fingerprint sensor needs both 3.3 and 5v. we need resistances to get proper voltage. (Figure 2). Also, the key of the fingerprint sensor is TTL, which is not the same to most sensors.



- Data Transmission

  In our project, we use esp8266 Wi-Fi module to send post or get requests as fingerprint sensor cannot using with esp8266 thing dev. For using esp8266 Wi-Fi, we use the AT commands to connect to Wi-Fi and send requests.

When connecting to Wi-Fi, we use such codes:

```
esp8266.begin(115200);
sendCommand("AT",5,"OK");
sendCommand("AT+CWMODE=3",5,"OK");
sendCommand("AT+CWJAP=\"6DC030\",\"H2ULAEG361198\"",5,"OK");
```

When sending a post request, we use such codes:

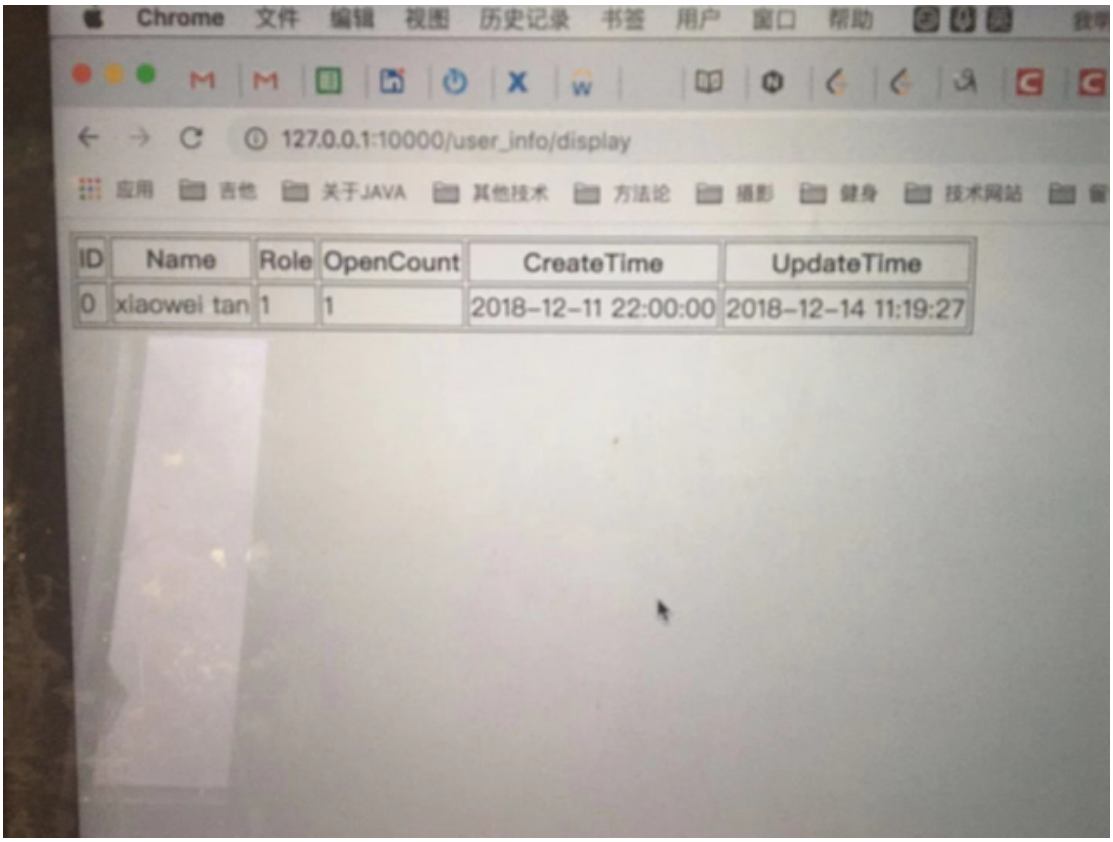
```
sendCommand("AT+CIPMUX=0",5,"OK");
sendCommand("AT+CIPSTART=\"TCP\",\"192.168.0.56\",20000",1,"OK");
sendCommand("AT+CIPSEND="+String(postRequest.length()),1,">");
esp8266.println(postRequest);
delay(1500);
sendCommand("AT+CIPCLOSE",1,"OK");
```

When sending a get request, we use such codes:

```
sendCommand("AT+CIPMUX=0",5,"OK");
sendCommand("AT+CIPSTART=\"TCP\",\"192.168.0.56\",20000",1,"OK");
sendCommand("AT+CIPSEND="+String(getRequest.length()),1,">");
delay(1000);
esp8266.println(getRequest);
delay(1500);
sendCommand("AT+CIPCLOSE",1,"OK");
```

- Three Modes
  - Enroll Mode

    In the ENROLL_MODE, the LCD screen will first tell you to do the admin check, which means the admin must agree with this enroll (we consider the first enrolled fingerprint as admin). Then the system will tell you to enroll the fingerprint for three times (needs to be same or enrollment will fail) (Figure 3). After that, you will be asked to write your name. Since then we have collected the data we need, the system uses esp8266 Wi-Fi module to upload the data to the cloud database and append the create time. You can also access our website to see the details of enrolled info (Figure 4). After everything is done, the LCD will show “success”.


- Detect Mode

  In the DETECT_MODE, the system will tell you to press your finger. If the fingerprint is not saved before, the detect is failed. If the fingerprint is checked, however, the fingerprint sensor will use the id it obtained to send a get request to our server, which is followed by the database query. After we get name attribute of the fingerprint, it returns to the Arduino and show “welcome home!” on the LCD screen (Figure 5). In the DECTECT_MODE, we will also use esp8266 Wi-Fi module for communicated but the direction is from the

cloud to the Arduino.


- Idle Mode

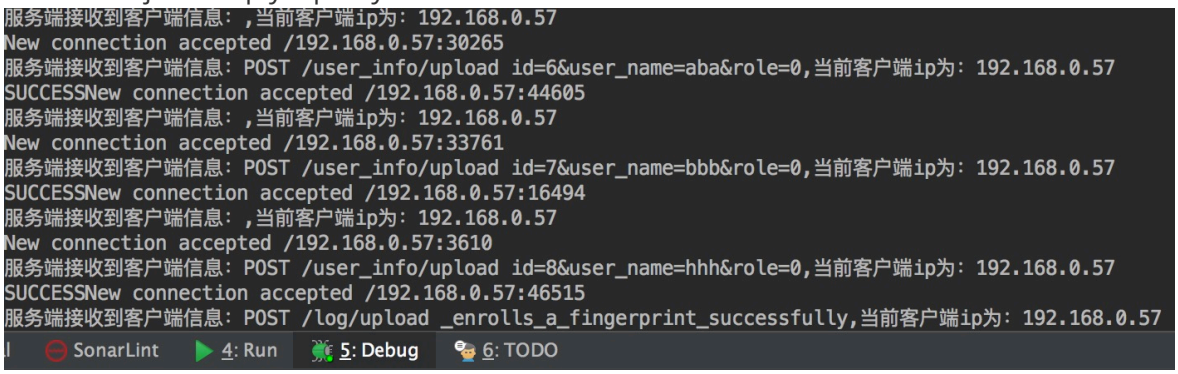
  Our system does nothing in this mode. If not having this mode, the system will be too busy and easily meet problems.

## Back End

There are three parts of the backend system.

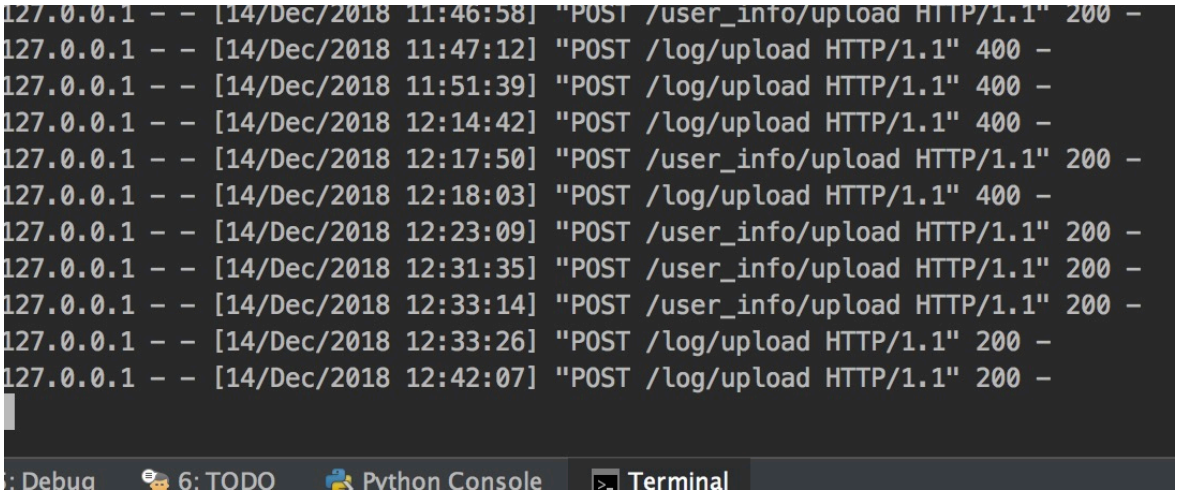
- Java Proxy Service

  We write a backend service in Java to accept TCP connections from ESP8266 Wifi Module and redirect the requests from ESP8266 to a Python service. It has little processing logic inside and just simply a proxy.




- Python Service

We write a backend service in Python to receive http requests from Java service complete the processing logic, including maintain the data storing in MySQL and render the front end pages.

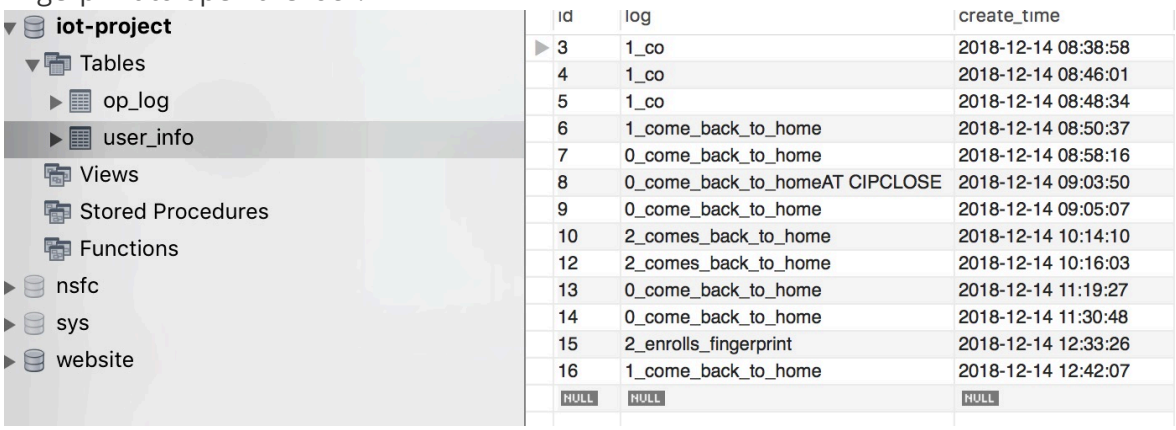



- MySQL

  We use MySQL as the database in our project. There are two tables, user_info and log.

  Table user_info contains the information about the user, including fingerprint id, name, times that have opened the lock, create time and update time.

  Table log contains the user operation logs, including newly enrolling fingerprint and check fingerprint to open the lock.




| id | log | create_time |
|---|---|---|
| 3 | 1_co | 2018-12-14 08:38:58 |
| 4 | 1_co | 2018-12-14 08:46:01 |
| 5 | 1_co | 2018-12-14 08:48:34 |
| 6 | 1_come_back_to_home | 2018-12-14 08:50:37 |
| 7 | 0_come_back_to_home | 2018-12-14 08:58:16 |
| 8 | 0_come_back_to_homeAT CIPCLOSE | 2018-12-14 09:03:50 |
| 9 | 0_come_back_to_home | 2018-12-14 09:05:07 |
| 10 | 2_comes_back_to_home | 2018-12-14 10:14:10 |
| 12 | 2_comes_back_to_home | 2018-12-14 10:16:03 |
| 13 | 0_come_back_to_home | 2018-12-14 11:19:27 |
| 14 | 0_come_back_to_home | 2018-12-14 11:30:48 |
| 15 | 2_enrolls_fingerprint | 2018-12-14 12:33:26 |
| 16 | 1_come_back_to_home | 2018-12-14 12:42:07 |
| NULL | NULL | NULL |

## Challenges Met In Project

- We don't get the key of the fingerprint sensor and bought it on the website. But the key cannot be used since on side socket is not corrected. Also, to use this sensor, we need resistances to get different voltage. TA helps us solved these problems.
- As our esp8266 thing board cannot connect to the Arduino board, we first considered to use Bluetooth. However, we can't get the data from NRF apps on our phone and transfer it to the cloud. Finally, we found that esp8266 Wi-Fi module for Arduino can solve such problem, so we went back and use Wi-Fi architecture.
- When doing cloud part, we found that most tutorial for IOT are showing a chart on cloud. Our situation, however, is to show a form. We decided to develop our own web project at last.
- As our project is complex, the Arduino always alert us that our memory is limited and will

cause unstable running. We then reduce global variable definition, free string as soon as finish using and cut off useless print in serial to save memory.

## Future Optimization

- We build our own database to save extra info while fingerprint sensor has its own database. But the tiny database can only accommodate 200 fingerprints. We consider to not using that small database at all and save all data on the cloud.
- Fingerprint lock is usually combining with the other way to unlock like using password or face ID. We plan to add face ID in the future cause sometimes the fingerprint is not working like your hands are wrapped with water.