SMART APARTMENT SECURITY SYSTEM

SafeNet

DESIGN MANUAL

Table of Content

1. Introduction

For years, the need to protect one's property has become one of people's main concerns. The sense of security and protection is one of those feelings that makes us comfortable and complements quality living. Although there are existing solutions for home security like video surveillance cameras, alarms etc., they are very expensive, and the installation process is also not easy. Most of the existing security system solutions address the needs of the people who are living in private houses. But people who live in apartments in urban areas have some added needs to be satisfied with a particular security system. Therefore, addressing all these issues along with some added unique features we have come up with the idea of a Smart Apartment Security System. Smart Apartment security system is a system that secure entry points, like doors and windows, as well as interior spaces in an apartment from a burglary or a fire by notifying the owners and security officer at the apartment gate whenever a threat has been identified through a mobile application.

2. Prerequisites

- Basic understanding of microcontroller programming with arduino framework
- Web development with HTML
- Mobile development with Flutter
- Backend development with NodeJS

3. Safenet

3.1 Overview

Safenet is a smart apartment security system that secure entry points, like doors and windows, as well as interior spaces in an apartment from a burglary or a fire by notifying the owners and security officer at the apartment gate whenever a threat has been identified through a mobile application.

It consists of four main parts.

1. Security system with sensors
   The security system is used to identify threats in the entry points and interior spaces of an apartment through sensors. It will consist of wireless sensors (door and window sensors, motion sensors, smoke detector), a siren and a smart door lock with RFID control access for added security.

2. Smart Door Lock
   Smart Door lock is for added security for front door. It uses RFID access control and password access control to access the front door from outside.

3. Mobile application for house owners and security officer
   Mobile application consists of two interfaces (for house owner and security officer). It is used to view and control the security system.

4. Web application for administration
   Admin of the security system has a admin web application where he has access to delete users, add, update and delete apartments, security officers and sensors for system.

3.2 Hardware components Required

Electronic Components

- ESP 32 Development board
- PIR Motion sensor module
- Flame sensor module
- Buzzer
- Relay board module
- Membrane switch keypad
- RFID card reader module
- I2C module
- LCD display
- Electric solenoid door lock
- Magnetic door reed switch
- Li-ion 1800mah battery

3.3 ESP -32 Development board

Wireless connectivity WiFi: 150.0 Mbps data rate with HT40

Bluetooth: BLE (Bluetooth Low Energy) and Bluetooth Classic

Processor: Tensilica Xtensa Dual-Core 32-bit LX6 microprocessor, running at 160 or 240 MHz

Memory:

- ROM: 448 KB (for booting and core functions)
- SRAM: 520 KB (for data and instructions)
- RTC fas SRAM: 8 KB (for data storage and main CPU during RTC Boot from the deep-sleep mode)
- RTC slow SRAM: 8KB (for co-processor accessing during deep-sleep mode)
- eFuse: 1 Kbit (of which 256 bits are used for the system (MAC address and chip configuration) and the remaining 768 bits are reserved for customer applications, including Flash-Encryption and Chip-ID)
- Embedded flash: flash connected internally via IO16, IO17, SD_CMD, SD_CLK, SD_DATA_0 and SD_DATA_1 on ESP32-D2WD and ESP32-PICO-D4.
    - 0 MiB (ESP32-D0WDQ6, ESP32-D0WD, and ESP32-S0WD chips)
    - 2 MiB (ESP32-D2WD chip)
    - 4 MiB (ESP32-PICO-D4 SiP module)

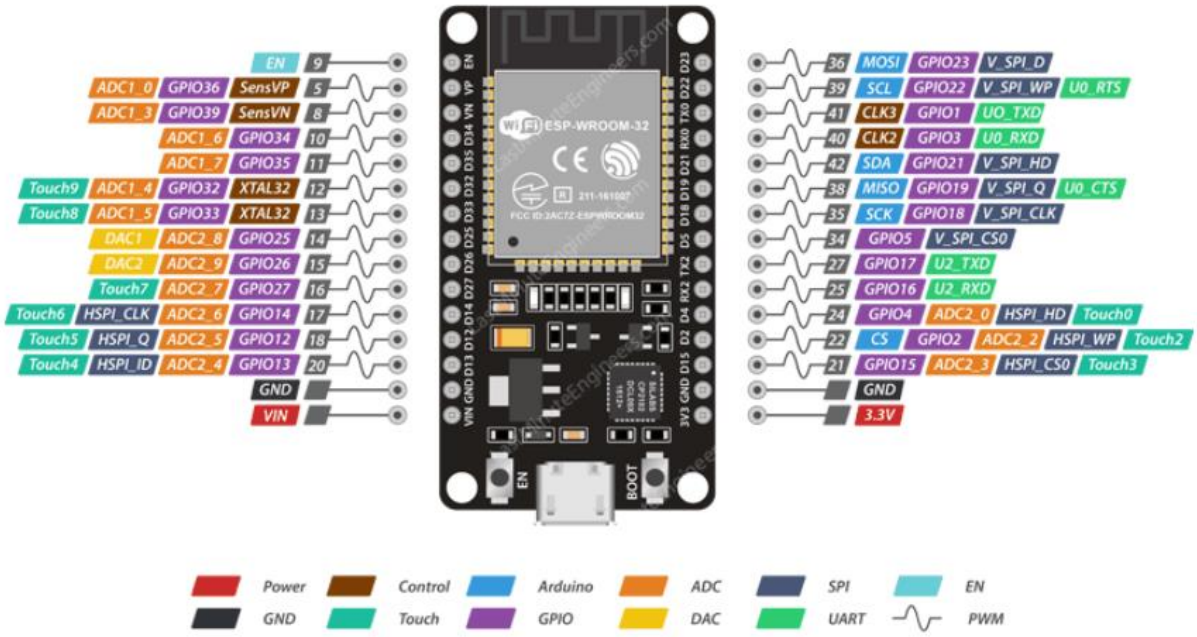
Low Power: ensures that you can still use ADC conversions, for example, during deep sleep.

Peripheral Input/Output:

- peripheral interface with DMA that includes capacitive touch
- ADCs (Analog-to-Digital Converter)
- DACs (Digital-to-Analog Converter)
- I²C (Inter-Integrated Circuit)
- UART (Universal Asynchronous Receiver/Transmitter)
- SPI (Serial Peripheral Interface)
- I²S (Integrated Interchip Sound)
- RMII (Reduced Media-Independent Interface)
- PWM (Pulse-Width Modulation)

Security: hardware accelerators for AES and SSL/TLS

3.4 ESP 32 pinout diagram

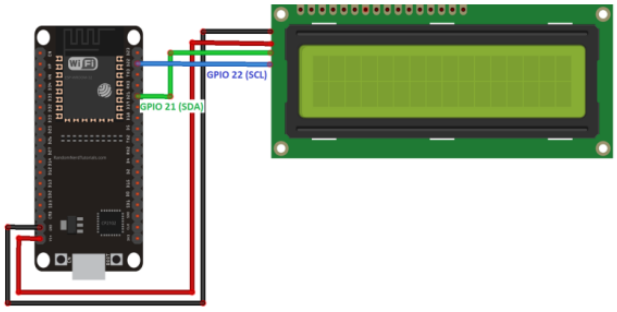


3.5 LCD Display

The 16x02 LCD display and the I2S port expander for the LCD are used to display the information to the user. I2C port expander uses I2C protocol to communicate between the microcontroller and the LCD I2C expander module. This method was used to reduce the number of GPIO interfaces used by the LCD.

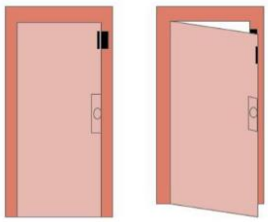
● Library used - LiquidCrystal_I2C

Following diagram shows how the LCD with I2C expander can be interfaced with the ESP32 micro controller.



The VCC and the GND pins of the I2C port should be connected with the 5V and the GND respectively and the SCL and SDA should be connected with the D22
and D21 respectively.

3.6 Sensors



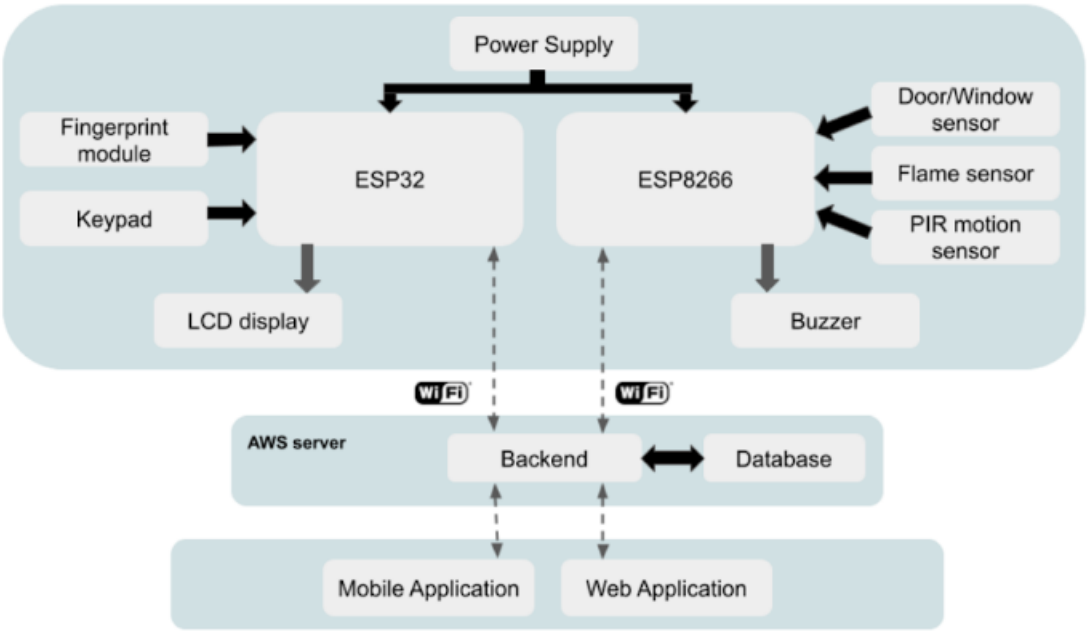
Magnetic Reed Switch Sensor

- Interfacing - ESP8266 Wifi module
- Accuracy - 0.125 inches(about 3mm)
- Calibration - Needed
- Power - operating voltage (3.3V-5V)
- Cost - Rs. 150/-

PIR motion sensor

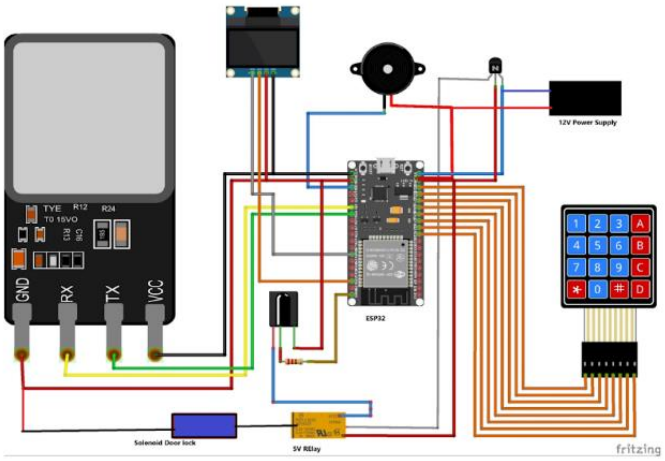
- Interfacing - ESP8266 Wifi module
- Accuracy - 93%
- Calibration - Needed
- Power - 3.3V-5V
- Cost - Rs.220/-

3.7 Block diagram

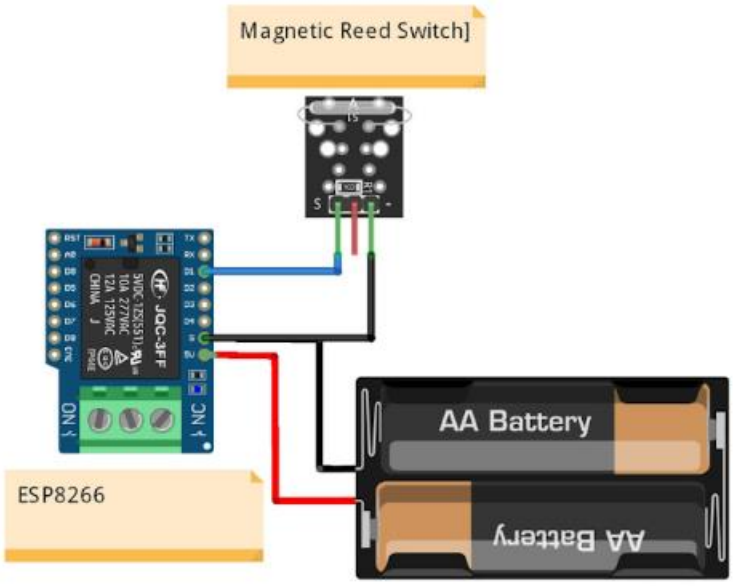


3.8 Circuit Diagram

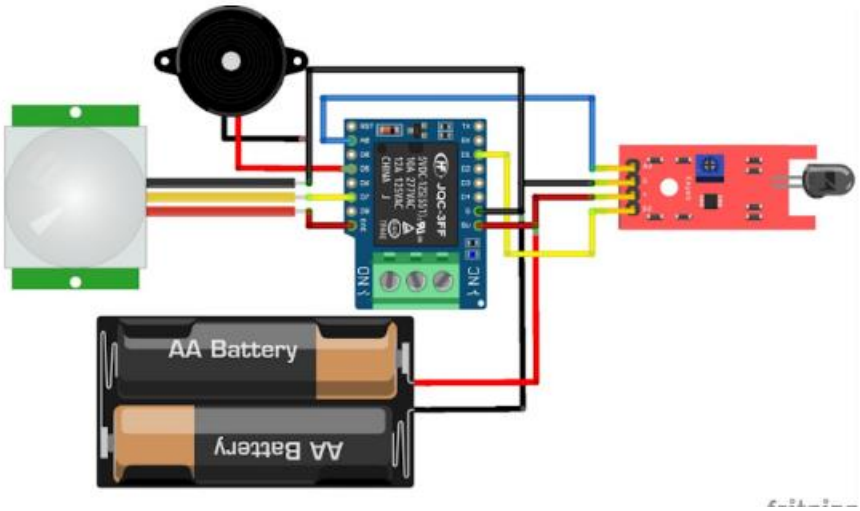
Door Lock circuit



Door/Window sensor circuit

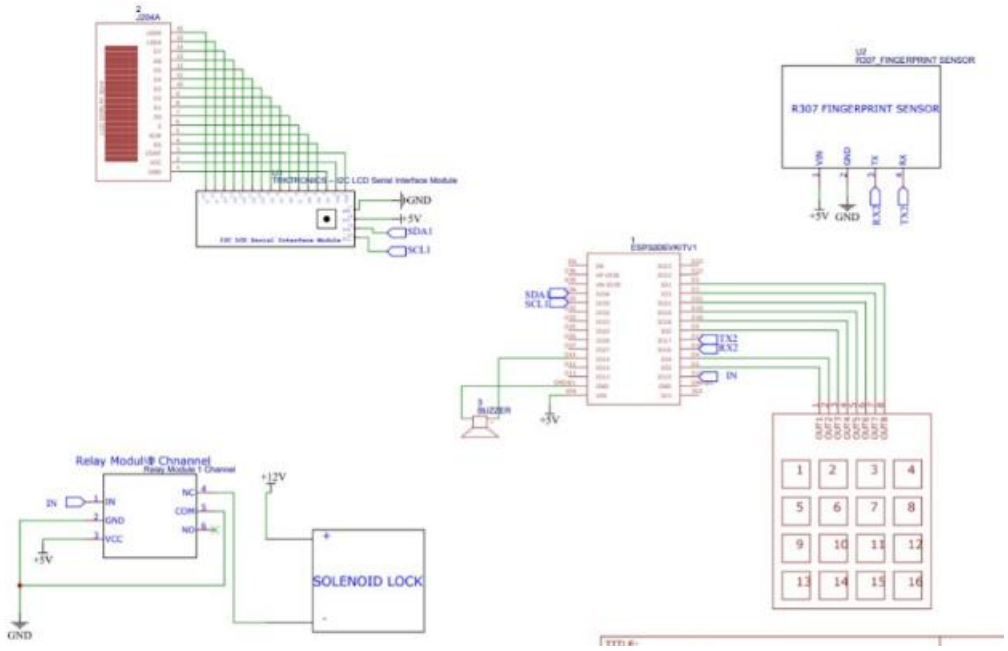


Flame & PIR sensor module circuit

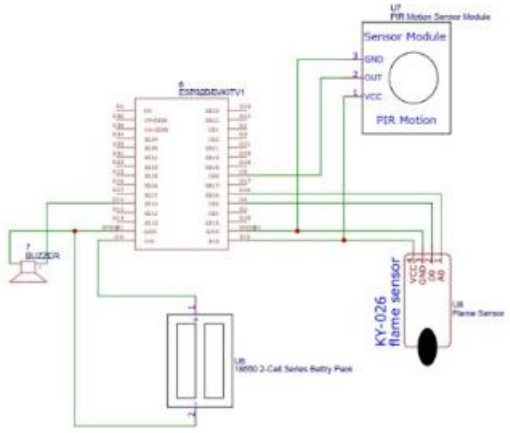


3.9 Schematic diagrams

Door Lock



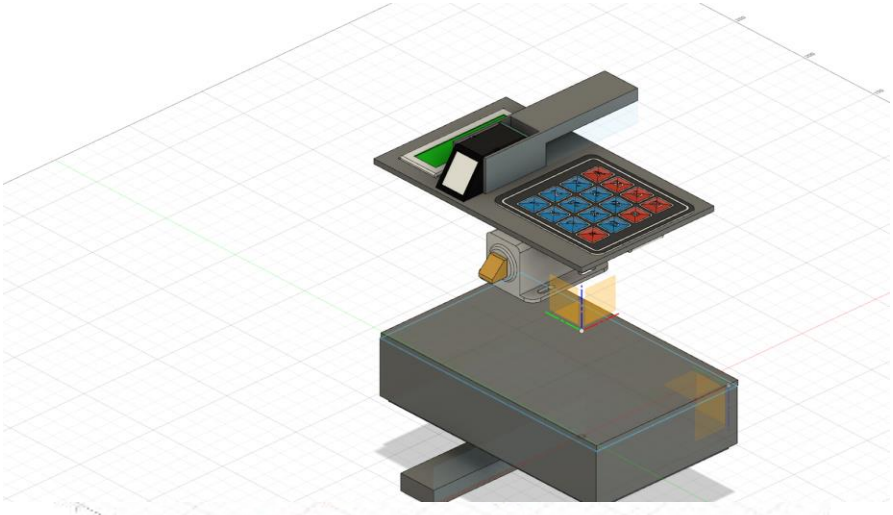
Sensor Module

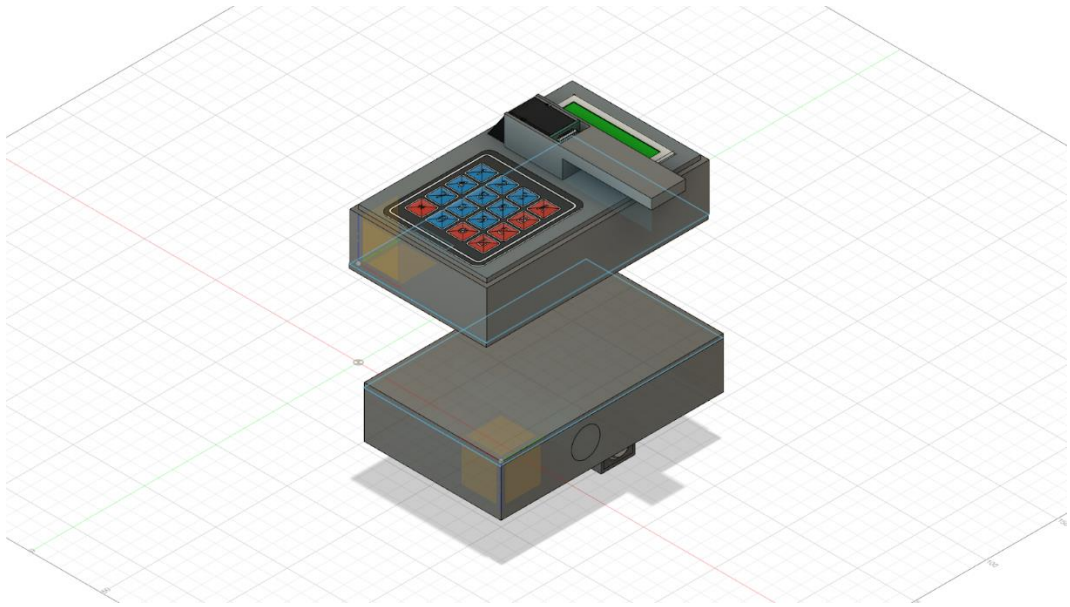


3.10 Power

3.7V 18000mah Lithium ion batteries are used for door lock and sensor modules. For door lock 4 batteries are needed and for sensor modules 2 batteries are needed.

3.11 Hardware designs





5. Technology

Front-end development

Flutter

As the front end development we use flutter as our front end development software in this development process the expectations are to provide user friendly and interesting interface with enhanced UI and UX experience to the client.Other than that cross platform support is required for us to maintain the target clients by giving them access to our services independant their mobile os type.

Backend development

NodeJs

In the backend server requires to handle a larger amount of requests. Therefore the scalability and sustainability if uncaught error exceptions happend are critical points under choosing a optimum backend server. For those requirements, Node js would be the best solution for this project. Therefore the backend server side requests managements are done by the NodeJs.

Cloud Deployment

AWS

Due to financial issues met in the process of developing the physical server of maintaining cost. The team sole decision and the academic guidance as well based on cloud services. Therefore the cloud server deployment was decieded. In that case the EC2 instance of AWS server support is selected.

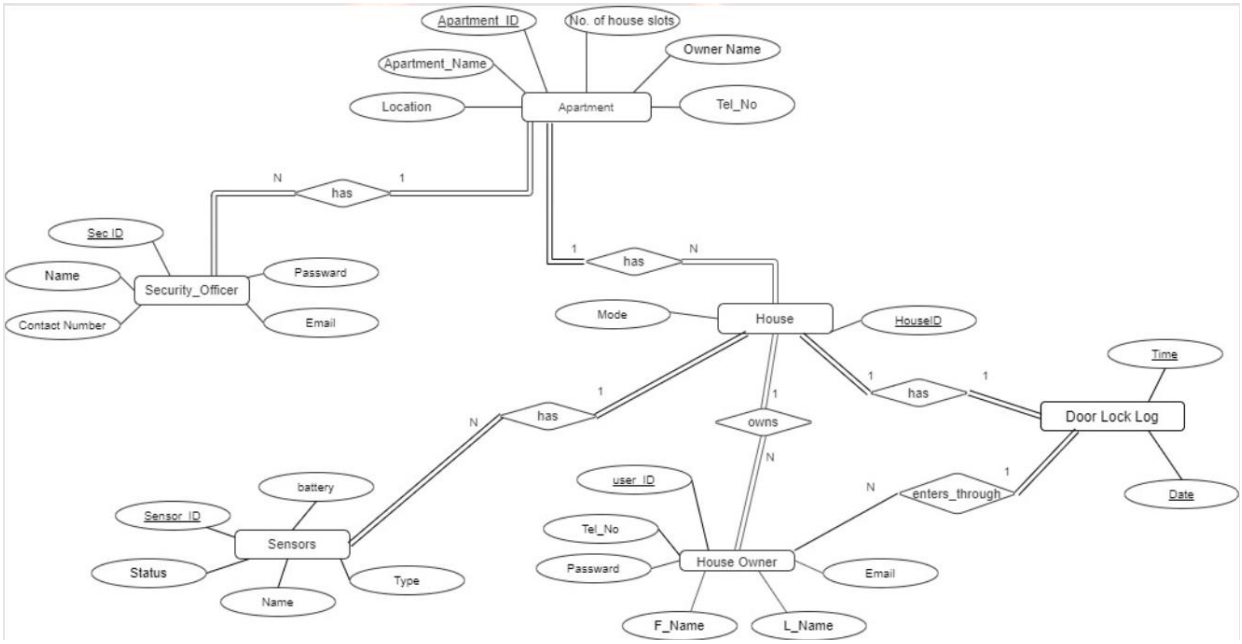
Database Development

MySQL

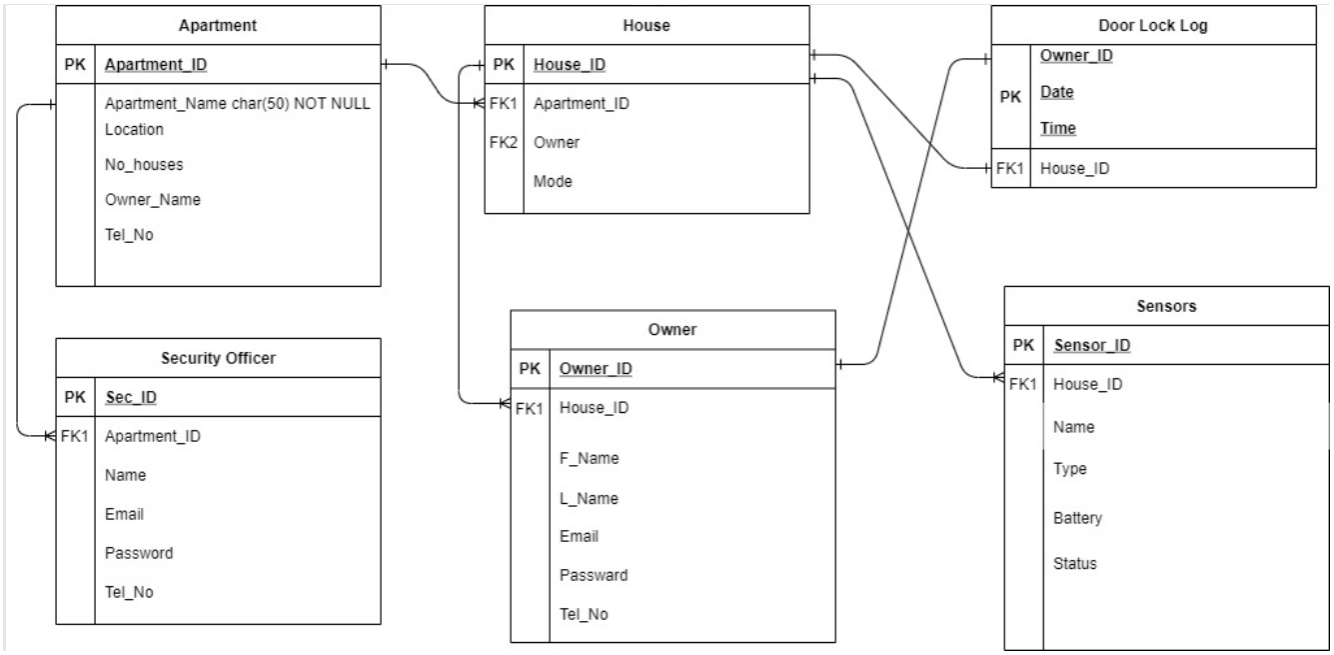
For storing data puroposes' requirements were the structured schema with the same type of data that have the ability to contain all the details of the target market when the product able to cover the whole market itself. Also the data base should be able to scalable from a smaller database. For these requirements are fulfil when the chosen selection was Mysql. Because the mysql have the ability contain 1000 different tables and all our reqiurements are fulfilled under 1000 tables. Also a table can record 21 million records with the space under 1GB. The maximum row table will be our user table but it will not exceed 21 million because the targeted ordience is less.

6. Diagrams

6.1 ER Diagram

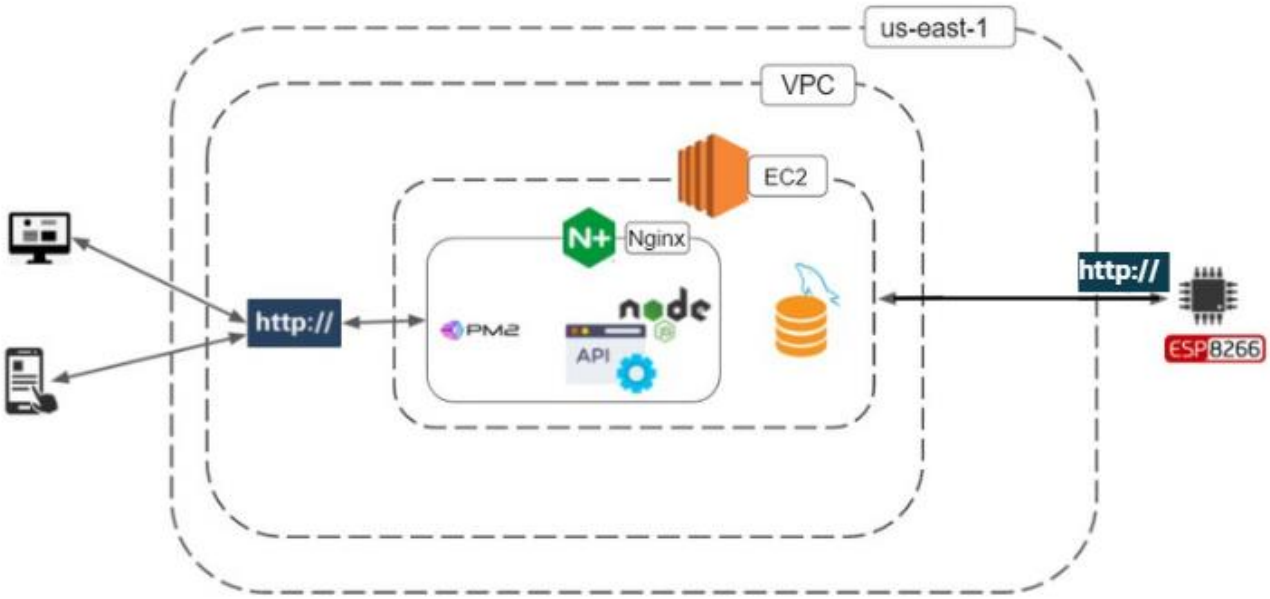


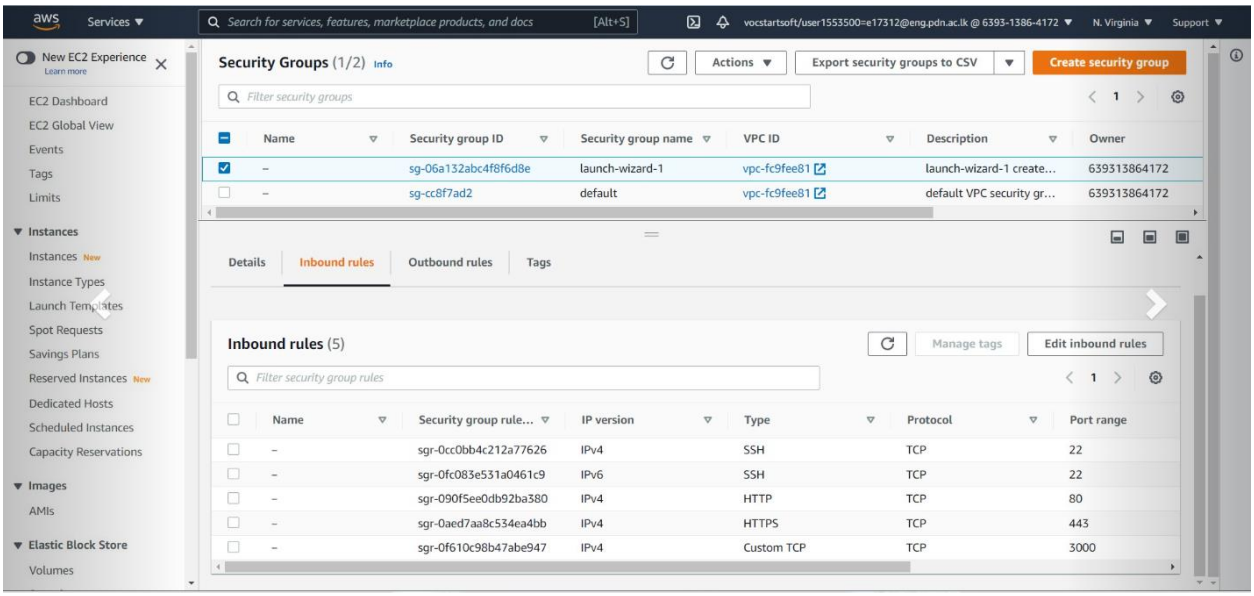
6.2 Relational Schema

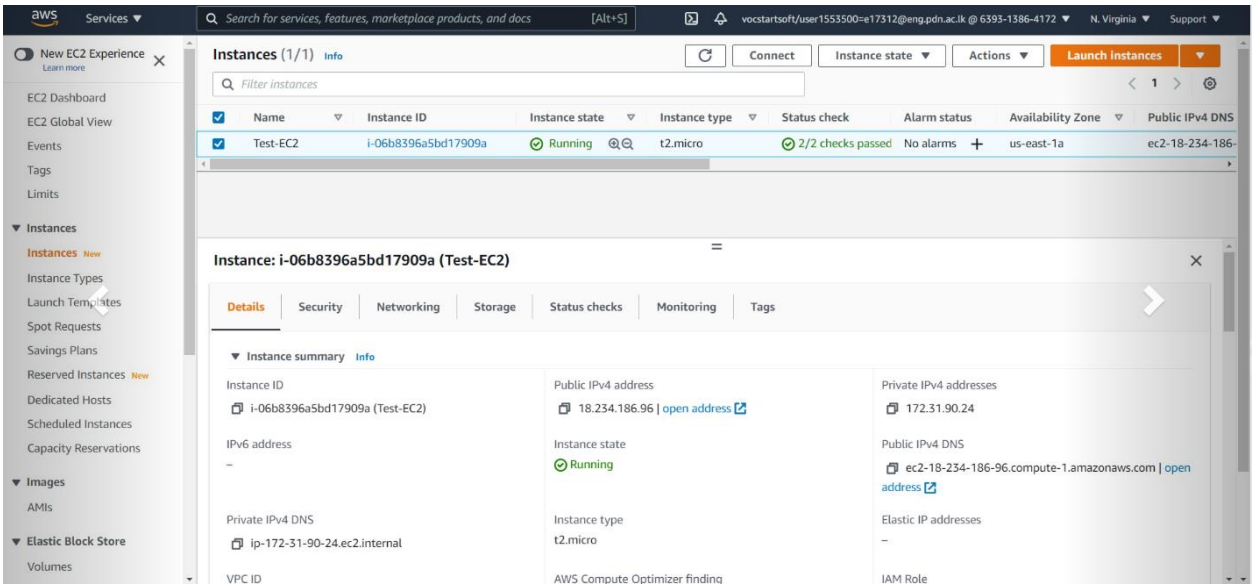


7. Cloud Deployment

- Resources are deployed in AWS cloud
- Used a single vpc to host the resources.
- A single linux server is used to deploy the backend and the database.
- Server is currently secured with proper security group rules and firewall rules.
- Pm2 is used as the process manager for the nodejs backend processes.
- Nginx is used as the reverse proxy.
- MariaDB (MySql) is used as the database and it is also deployed on the same server.
- For the POC version we went with this single server design for simplicity and the cost concern.
- However, to scale this we can easily offload the database to AWS RDS so that the backend can be scaled separately and RDS takes care of data security, integrity and backups.



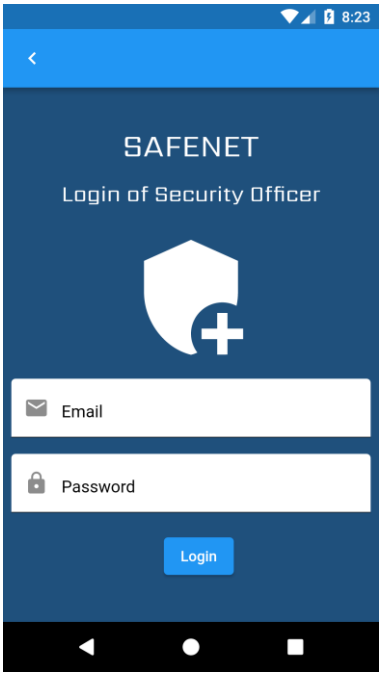

Security Groups (1/2) Info
Actions
Export security groups to CSV
Create security group
Filter security groups
Name
Security group ID
Security group name
VPC ID
Description
Owner
sg-06a132abc4f8f6d8e
launch-wizard-1
vpc-fc9fee81
launch-wizard-1 create...
639313864172
sg-cc8f7ad2
default
vpc-fc9fee81
default VPC security gr...
639313864172
Details
Inbound rules
Outbound rules
Tags
Inbound rules (5)
Manage tags
Edit inbound rules
Filter security group rules
Security group rule...
IP version
Type
Protocol
Port range
sgr-0cc0bb4c212a77626
IPv4
SSH
TCP
22
sgr-0fc083e531a0461c9
IPv6
SSH
TCP
22
sgr-090f5ee0db92ba380
IPv4
HTTP
TCP
80
sgr-0aed7aa8c534ea4bb
IPv4
HTTPS
TCP
443
sgr-0f610c98b47abe947
IPv4
Custom TCP
TCP
3000

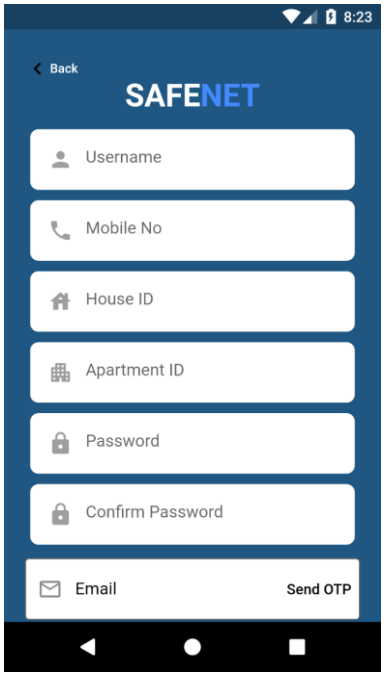

Instances (1/1) Info
Connect
Instance state
Actions
Launch instances
Filter instances
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Test-EC2
i-06b8396a5bd17909a
Running
t2.micro
2/2 checks passed
No alarms
us-east-1a
Instance: i-06b8396a5bd17909a (Test-EC2)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance summary Info
Instance ID
i-06b8396a5bd17909a (Test-EC2)
Public IPv4 address
18.234.186.96 | open address
Private IPv4 addresses
172.31.90.24
IPv6 address
Instance state
Running
Public IPv4 DNS
ec2-18-234-186-96.compute-1.amazonaws.com | open address
Private IPv4 DNS
ip-172-31-90-24.ec2.internal
Instance type
t2.micro
Elastic IP addresses
VPC ID
AWS Compute Optimizer finding
IAM Role

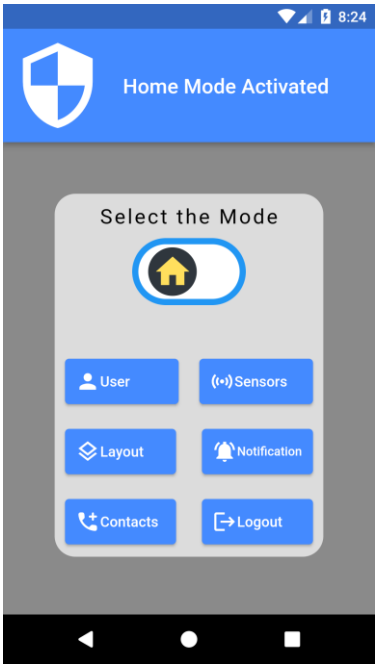
8. Mobile Application

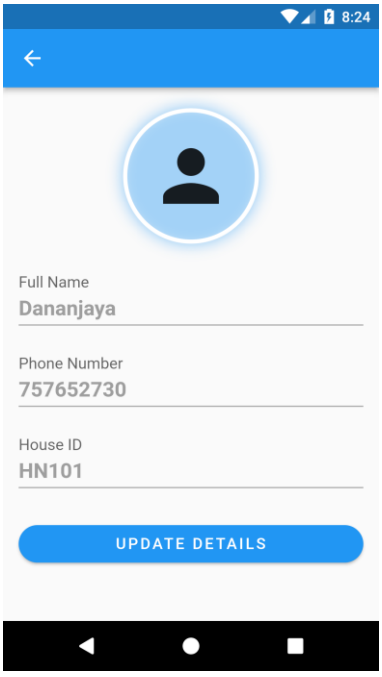
Home owner interface

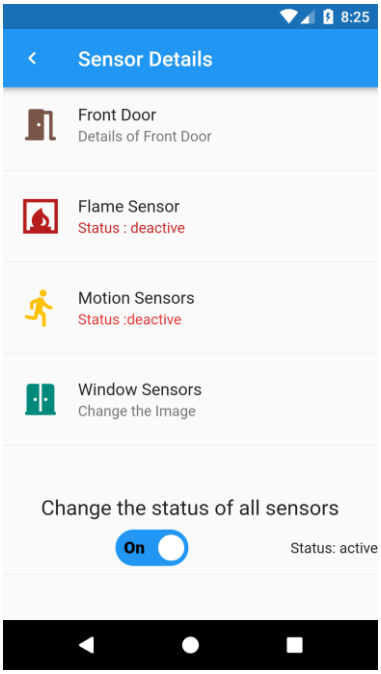
Mobile Application has two interfaces. House owners can change the mode between Home and Away. Only in Away mode users get notifications. From Users button users can update their details. From sensor button can view all the status of sensors. From contact details button can contact emergency contacts.

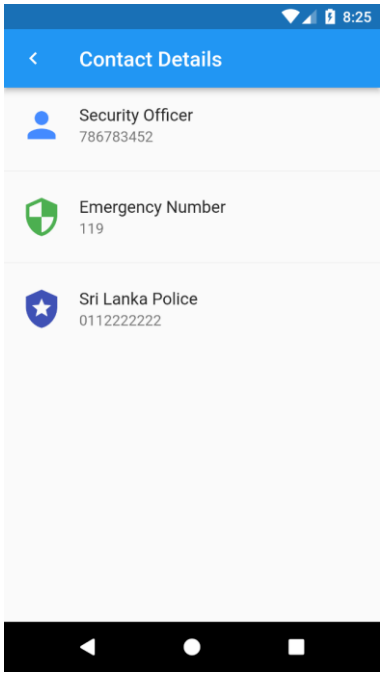






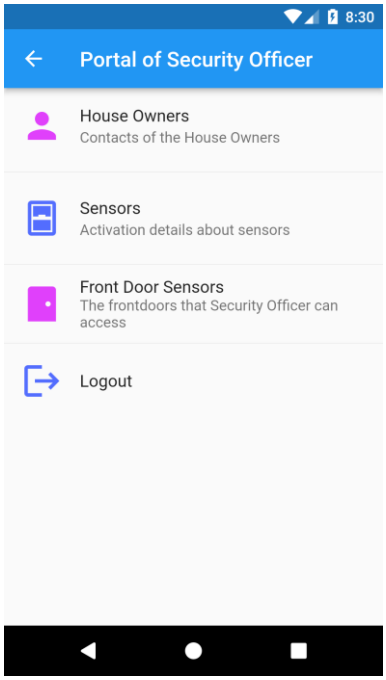


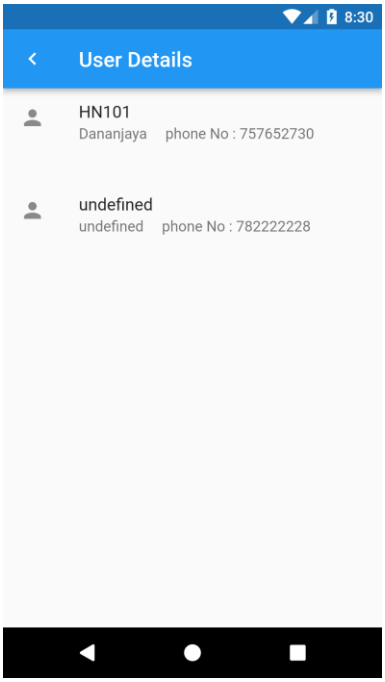


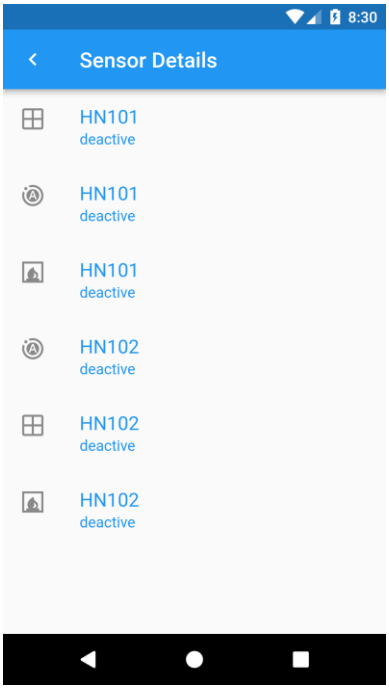


Security Officer Interface

From the interface of security officer, security officer can see the details of Users with respect to their house id. And also, he can view the house numbers he is given access to and all the sensors activated at the moment.







9. Failure Handling

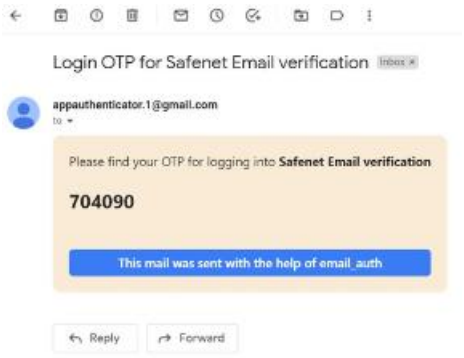
- **Power failures for door lock –** Using a UPS for router for uninterrupted Wi-Fi
- **When unexpected failures happen in application –** Server actions logging and monitoring using applications like Sentry.io

10. Security Enhancement

Security is the most important thing in the IT world right now. In our solution, we have deeply considered the security point of view of our system.

We have developed our system with the help of the following techniques to enhance the application security in our solution.

Two factor Authentication



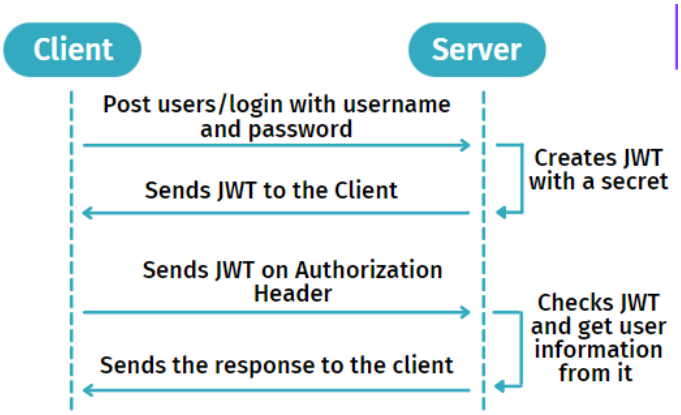
Used in Authentication is used in the Registration of the Mobile Application. When Email is provided, OTP is sent to email address and user has to Enter that OTP for the registration.

User Input Validations

This prevents improperly formed data from entering an information system. Because it is difficult to detect a malicious user who is trying to attack software, applications should check and validate all input entered into a system specially in Resgistration and Login. Input validation as a mitigation strategy for both SQL injection and XSS.

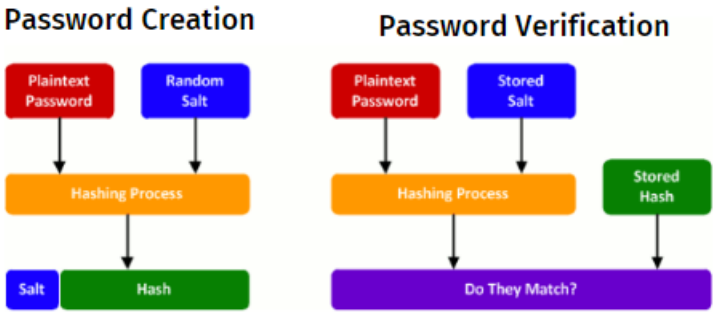
JWT Tokens

Our application use JSON Web Tokens (JWT) to allow the client to indicate its identity for further exchange after authentication. It is a self-contained way for securely transmitting information between **parties.**



Password Hashing

Password hashing is used in our application to verify the integrity of passwords, sent during login, against the stored hash so that actual passwords never has to be stored. A salt is added to the hashing process to force their uniqueness, increase their complexity without increasing user requirements, and to mitigate password attacks like hash tables



AWS EC2 Security Groups

A custom security group is assigned to the instance where our application is deployed which allows only http/tcp connections on port 80. Security groups offer protection at the ports and protocol access level

11. Testing

Test	Reason for testing	Procedure	Results and findings
User Input Validation for Login and Register page	Prevents improperly formed data from entering an information system. Prevents injection attacks, buffer overflows.	API testing using JEST and SUPERTEST	App successfully giving access for correctly entered records and giving error messages for violating validations.
Testing Authorization	To check whether it is securely transmitting information between parties.	API testing using JEST and SUPERTEST	App successfully giving permission when token is provided and denying access when token is missing
Load Testing	To determine a system's behavior under anticipated peak load conditions	Using Artillery	In a duration of 20 seconds, if 10 new arrivals per seconds came, (20 requests per each new arrival) the mean responses/second is 195.69
Validate_login Unit test	To check whether validate_login function work as expected	Using Flutter tests	Validate_login function works as expected
Widget testing in mobile application	To verify that every widget's UI looks and behaves as expected	Using Flutter driver	Widgets behave as expected.