A PROJECT REPORT

ON

"SMART SOCIETY APP"

Submitted to

UNIVERSITY OF MUMBAI

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER ENGINEERING

BY

HONPODE ARAFAT LIYAQAT ALI SABIHA	18DCO04
MAHADKAR SALMAN JAVED FARZANA	18DCO10
SHAIKH ADIL REHMATULLA SAJIDA	18DCO16
KHAN SAIF NADEEM YASMEEN	16DCO59

UNDER THE GUIDANCE OF
PROF. MUKHTAR ANSARI

DEPARTMENT OF COMPUTER ENGINEERING
Anjuman-I-Islam's Kalsekar Technical Campus
SCHOOL OF ENGINEERING & TECHNOLOGY
Plot No. 2 3, Sector - 16, Near Thana Naka,
Khandagaon, New Panvel - 410206

2020-2021

AFFILIATED TO

UNIVERSITY OF MUMBAI

A PROJECT II REPORT
ON

"SMART SOCIETY APP"

Submitted to

UNIVERSITY OF MUMBAI

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN COMPUTER ENGINEERING

BY

HONPODE ARAFAT LIYAQAT ALI SABIHA	18DCO04
MAHADKAR SALMAN JAVED FARZANA	18DCO10
SHAIKH ADIL REHMATULLA SAJIDA	18DCO16
KHAN SAIF NADEEM YASMEEN	16DCO59

UNDER THE GUIDANCE OF
PROF. MUKHTAR ANSARI

DEPARTMENT OF COMPUTER ENGINEERING
Anjuman-I-Islam's Kalsekar Technical Campus
SCHOOL OF ENGINEERING & TECHNOLOGY
Plot No. 2 3, Sector - 16, Near Thana Naka,
Khandagaon, New Panvel - 410206

2020-2021
AFFILIATED TO

UNIVERSITY OF MUMBAI

Anjuman-i-Islam's Kalsekar Technical Campus

Department of Computer Engineering
SCHOOL OF ENGINEERING & TECHNOLOGY
Plot No. 2 3, Sector - 16, Near Thana Naka,
Khandagaon, New Panvel - 410206

CERTIFICATE

This is certify that the project entitled

"SMART SOCIETY APP"

submitted by

HONPODE ARAFAT LIYAQAT ALI SABIHA	**18DCO04**
MAHADKAR SALMAN JAVED FARZANA	**18DCO10**
SHAIKH ADIL REHMATULLA SAJIDA	**18DCO16**
KHAN SAIF NADEEM YASMEEN	**16DCO59**

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Engineering) at ***Anjuman-I-Islam's Kalsekar Technical Campus, Navi Mumbai*** under the University of MUMBAI. This work is done during year 2020-2021, under our guidance.

Date: / /

PROF. MUKHTAR ANSARI
Project Supervisor

PROF. KALPANA BODKE
Project Coordinator

PROF. TABREZ KHAN
HOD, Computer Department

DR. ABDUL RAZAK HONNUTAGI
Director

External Examiner

Acknowledgements

I would like to take the opportunity to express my sincere thanks to my guide **Prof. MUKHTAR ANSARI**, Assistant Professor, Department of Computer Engineering, AIKTC, School of Engineering, Panvel for his invaluable support and guidance throughout my project research work. Without his kind guidance & support this was not possible.

I am grateful to him/her for his timely feedback which helped me track and schedule the process effectively. His/her time, ideas and encouragement that he gave is help me to complete my project efficiently.

We would like to express deepest appreciation towards **DR. ABDUL RAZAK HONNUTAGI**, Director, AIKTC, Navi Mumbai, **Prof. TABREZ KHAN**, Head of Department of Computer Engineering and **Prof. KALPANA R. BODKE**, Project Coordinator whose invaluable guidance supported us in completing this project.

At last we must express our sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped me directly or indirectly during this course of work.

HONPODE ARAFAT LIYAQAT ALI SABIHA
MAHADKAR SALMAN JAVED FARZANA
SHAIKH ADIL REHMATULLA SAJIDA
KHAN SAIF NADEEM YASMEEN

Project I Approval for Bachelor of Engineering

This project entitled ***S̈MART SOCIETY APP¨*** by ***HONPODE ARAFAT LIYAQAT ALI SABIHA, MAHADKAR SALMAN JAVED FARZANA, SHAIKH ADIL REHMAT-ULLA SAJIDA, KHAN SAIF NADEEM YASMEEN*** is approved for the degree of ***Bachelor of Engineering in Department of Computer Engineering.***

Examiners

1. .....
2. .....

Supervisors

1. .....
2. .....

Chairman

.....

Declaration

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Honpode Arafat Liyaqat Ali Sabiha
Roll Number: 18DCO04

Mahadkar Salman Javed Farzana
Roll Number: 18DCO10

Shaikh Adil Rehamattulla Sajida
Roll Number: 18DCO16

Khan Saif Nadeem Yasmeen
Roll Number: 16DCO59

ABSTRACT

Apart from individual expenses, living in a housing society involves upkeep, maintenance, and upgrade of areas which are not directly under any individual's ownership. This requires not only payments to the society complex or resident welfare association but also adequate communication and transparency between the members and management. Payments to the society complex for maintenance, in a usual set-up is done by writing cheques or in cash. Communication and transparency between members and society is maintained through annual reports of finances and meetings. However, the current system of maintaining housing society requires tremendous man power, time and human interaction. For example: notifying members about power shedding, water shortage, bill payment for maintenance requires human interaction. Given the current situation of COVID-19 such interaction is to be avoided. Apart from the pandemic situation the usual methods of maintenance of a society are still cost inefficient, time consuming and lack transparency between society members and management. It can also be inefficient due to delay in addressing of complaints and grievances due to varied reasons

So, we are here with an android application that will provide assistance to society members and committee. This application contains different features that will overcome the drawbacks of maintaining housing societies via traditional methods. Few of the features are as follows:- Notice: This feature will update the notices. Complaints/Grievance: The feature will result in early redressal making it more efficient. Online Maintenance bill payment: This feature will avoid human interaction and give the liberty to pay from any where and any time with ease, Gate Pass: This feature will provide an extra security measure to the society and Contacts: This feature will contain contact information of important individuals such as committee members, security guards will help in quick redressals.

Keywords And Glossary

Keywords :
Firebase, Society, User, Admin, Committee, Member, Database, Management.

Glossary :

A:

Android - Android is a mobile operating system based on a modified version of Linus kernel and other open source software, designed primarily for touch screen mobile devices such as smartphones and tablets.

API - a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.

Android Studio - Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains IntelliJ IDEA software and designed specifically for Android development.

Android Application - Android App is a software designed to run on a Android device or emulator. The term also refers to an APK file which stands for Android package.

Analysis - detailed examination of the elements or structure of something.

Authentication - Authentication is the process of recognizing a user's identity. It is the mechanism of associating an incoming request with a set of identifying credentials. The credentials provided are compared

to those on a file in a database of the authorized user's information on a local operating system or within an authentication server.

B:

Building - a structure with a roof and walls, such as a house or factory.

C:

Committee - a group of people appointed for a specific function by a larger group and typically consisting of members of that group.

Contact - a number assigned to a telephone line for a specific phone or set of phones (as for a residence) that is used to call that phone. — called also phone number.

Complaint(s) - a statement that something is unsatisfactory or unacceptable.

D:

Database - a structured set of data held in a computer, especially one that is accessible in various ways.

F:

Firebase - Firebase is a platform developed by Google for creating mobile and web applications.

Flat - flat is a self-contained housing unit (a type of residential real estate) that occupies only part of a building, generally on a single story.

G:

Grievance - a real or imagined cause for complaint, especially unfair treatment.

J:

Java - Java is a class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible.

M:

Maintenance - the process of preserving a condition or situation or the state of being preserved.

Management - the process of dealing with or controlling things or people.

N:

Notification - the action of notifying someone or something.

Notice - notification or warning of something, especially to allow preparations to be made.

R:

Redressal - remedy or compensation for a wrong or grievance.

V:

Validation - the action of checking or proving the validity or accuracy of something.

Verification - the process of establishing the truth, accuracy, or validity of something.

X:

XML - XML stands for Extensible Markup Language. In Android we use xml for designing our layouts.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Purpose

The current system of maintaining housing society requires tremendous man power, time and human interaction. For example: notifying members about power shedding, water shortage, bill payment for maintenance requires human interaction. Given the current situation of COVID19 such interaction is to be avoided. Apart from the pandemic situation the usual methods of maintenance of a society are still cost inefficient,time consuming,demand more labour and lack transparency between society members and management. In the terms of expenditure,expenses on stationary such as paper,printer,ink for the printing of notices,receipts etc. Time consuming in the terms of personally writing checks,payment through cash. Demand more labour in terms of an individual is tasked to collect checks and cash for maintenance and record keeping another individual is tasked to deliver receipts and notices to individual apartment. Lack of transparency between society members and management in terms of lack of up-to-date recording keeping of finances used for maintaining housing society, reduces transparency as there is usually one annual report of finances. It can also be inefficient due to delay in addressing of complaints and grievances due to varied reasons.

1.2 Project Scope

- This application will overcome the drawbacks of maintaining housing societies via traditional methods.
- Time efficient in terms of users will be able to pay maintenance bill from any place -any time, quick delivery of information by uploading notices on the app and early redressal of complaint(s).
- Cost effectiveness by reduced use of stationary.
- Reduced effort as members will not be required to personally deposit cash/cheque(s) for maintenance and personnel to hand-deliver notices.
- Increased transparency between society members and management as financial transactions for maintenance of the society will be regularly updated.

1.3 Project Goals and Objectives

1.3.1 Goals

- Online Maintenance Payment will avoid human interaction and give the liberty to pay from any where and any time with ease. User can pay maintenance bill through online payment methods like Google Pay, Paytm, PhonePe, etc.
- Complaints feature will result in quick redressal, making it more efficient. Users can register complaints by specifying the level of critical, category and describing the issue. Where as admin (committee member) can resolve the complaints and mark the complaints status accordingly. Users can track the complaints status as well.
- Updates of notices will reduce man power, making it cost efficient and reducing human interaction. Users can view the notices, where as admin can add notices.

- Contact information about important individuals such as committee members, security guards will help in quick redressals.

1.3.2 Objectives

The objectives of creating an app to develop a new system that is time efficient, cost effective, reduces manual labour and increases the transparency.This will be achieved through the following points:

- Time efficiency
- Cost effectiveness
- Reduced labour
- Increased transparency between society members and management

1.4 Organization of Report

- Chapter 1: Gives a brief introduction about our project.
- Chapter 2 : Describes the literature review of the existing papers and the description about the application.
- Chapter 3 : Discuss about the project planning and different roles and capability of the team members. Also talks about the budget of the project.
- Chapter 4 : Describe the brief description of the srs and the other requirements of the project.
- Chapter 5 : Shows the system design, functional requirements and different diagram of the project.
- Chapter 6 : Shows Implementation of the app and coding.
- Chapter 7 : Shows the different testings performed and the problems faced. It also shows snapshots of the current working application.

- Chapter 8 : Describes the closure to the book and tries to conclude the work in the project and also mentions the future scopes as to where it would be used Chapter.

Chapter 2

Literature Survey

2.1 Housing Society Management

The objective of this paper was to comfort its users with easily understandable as well as essential functionalities. Here, both managing committee and residents have the same application installed with the maintenance generation and financial report generation features disabled at the resident side. This is achieved by maintaining separate login type for both types of users. Both types of users have similar rights over remaining features like viewing and posting notices on the notice board, adding and getting notified by calendar events and accessing society member contacts and the miscellaneous contacts. This application is implemented to help manage the affairs of a housing society by requiring the committee member to enter and save minimal amount of information. It will allow the members of the housing society to access information about a society, its residents and the managing committee on the go. Thus, this application provides a virtual tour of the society.

2.1.1 Advantages of Paper

a. Easy GUI

b. Posting Notices

c. Access Member Contacts

2.1.2 Disadvantages of Paper

a. Society residents can't pay the Maintenance Bill via online payment methods.

b. There is no complaint status related option in the application, and if the complaint has been resolved or not.

c. Not so attractive GUI.

2.1.3 How to overcome the problems mentioned in Paper

In our app, we will provide better GUI. In our app society residents can pay Maintenance Bill online via digital payment methods. In our app we have additional feature such as Complaint status, where the society member can know the status of his/her complaint. In our app we have Gate Pass feature, which will provide extra security to the society

2.2 Society Management Application On Android

The objective of this paper was to develop an android application for housing society management to reduce the human efforts and errors to increase crystal clear transparency between society members and management. It also helps to reduce the time and efforts for manual communication in society by providing the notifications and important information to the society members in the reliable and transparent way.

2.2.1 Advantages of Paper

a. Reduce time and manual efforts

b. Provides information in a transparent way

2.2.2 Disadvantages of Paper

a. There is no Gate Pass feature.

b. Not so attractive GUI.

2.2.3 How to overcome the problems mentioned in Paper

In our app, we will provide better GUI. In our app we have Gate Pass feature, which will provide extra security to the society.

2.3 Implementation Of Society Management System

The application provides a hassle free means of communication to conduct and regulate the day to day tasks in the society using affordable, easily available and customizable android technology using push notifications. User can access these services through first registration and then login. The purpose of this project development is to enable the residents and administrative people to play their role effectively and in a smarter but disciplined way. This may improve the interaction between society members and higher authorities.

2.3.1 Advantages of Paper

a. Uses push notification

b. Improves interaction

2.3.2 Disadvantages of Paper

a. No payment method.

b. Multiple reminders can create disturbance to some people.

2.3.3 How to overcome the problems mentioned in Paper

In our app society residents can pay Maintenance Bill online via digital payment methods. Our app provides timeline for society events to visualize the society in a lively way for better user experience.

2.4 Survey Existing System

Housing Society management App In Housing Society Management App both managing committee and residents have the same application installed with the maintenance generation and financial report generation features disabled at the resident side. This is achieved by maintaining separate login type for both types of users. Both types of users have similar rights over remaining features like viewing and posting notices on the notice board, adding and getting notified by calendar events and accessing society member contacts and the miscellaneous contacts. This application is implemented to help manage the affairs of a housing society by requiring the committee member to enter and save minimal amount of information. This app will allow the members of the housing society to access information about a society, its residents and the managing committee on the go.

Figure 2.1: HSM App Main Application Home Page

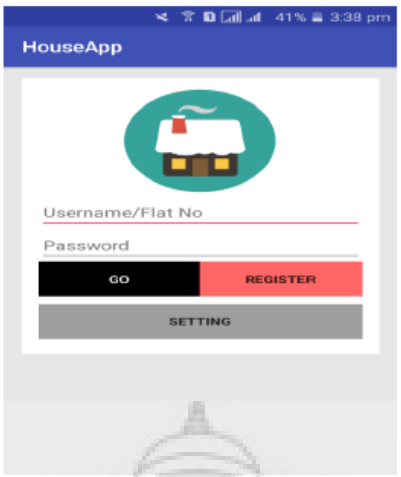


Figure 2.2: HSM App Application Home Page

2.5 Summary of Literature Review

SR.NO	Parameter	Paper-I	Paper-II	Paper-III	Proposed System
1.	Payment of maintenance bill online via digital payment methods	Not Present	Present	Not Present	Present
2.	Complaints	Not Present	Not Present	Not Present	Present
3.	Gate Pass	Not Present	Not Present	Not Present	Present
4.	Notice	Present	Present	Present	Present

Table 2.1: Comparison of Literature Paper existing System with our Proposed System

2.6 Market Potential and Competitive Advantage

In our smart society app, society residents can pay Maintenance Bill online via digital payment methods. Our app has an additional feature of complaint status, where the member can know the status of his/her complaint.The app also has a Gate Pass feature, which will provide extra security to the society. The notice feature will help the society members to keep track of all updates and events within the apartment complex. Occupancy/Vacancy status of the flats Will be updated on the application.

2.7 Technical Review

Technologies being used for the development of this application includes:

- Android Studio
- Java, XML
- Firebase database server
- Firebase for client side validation
- PHP

2.7.1 Java

JAVA is a programming language which is used in Android App Development. It is class based and object oriented programming whose syntax is influenced by C++. The primary goals of JAVA is to be simple, object-oriented, robust, secure and high level. JAVA application runs on JVM (JAVA Virtual Machine) but Android has it's own virtual machine called Dalvik Virtual Machine (DVM) optimized for mobile devices. Java is a high-level programming language developed by Sun Microsystems. It was originally designed for developing programs for set-top boxes and handheld devices, but later became a popular choice for creating web applications. Android applications are developed using the Java language.

Figure 2.3: Java

As of now, that's really your only option for native applications. Java is a very popular programming language developed by Sun Microsystems (now owned by Oracle). Developed long after C and C++, Java incorporates many of the powerful features of those powerful languages while addressing some of their drawbacks. Still, programming languages are only as powerful as their libraries. These libraries exist to help developers build applications.

Some of the Java's important core features are:

- It's easy to learn and understand
- It's designed to be platform-independent and secure, using virtual machines
- It's object-oriented

Android relies heavily on these Java fundamentals. The Android SDK includes many standard Java libraries (data structure libraries, math libraries, graphics libraries, networking libraries and everything else you could want) as well as special Android libraries that will help you develop awesome Android applications.

2.7.2 XML

XML stands for Extensible Markup Language. It is a text-based markup language derived from Standard Generalized Markup Language (SGML). XML tags identify the data and are used to store and organize the data, rather than specifying how to display it like HTML tags, which are used to display the data. XML is not going to replace HTML in the near future, but it introduces new possibilities by adopting many successful features of HTML

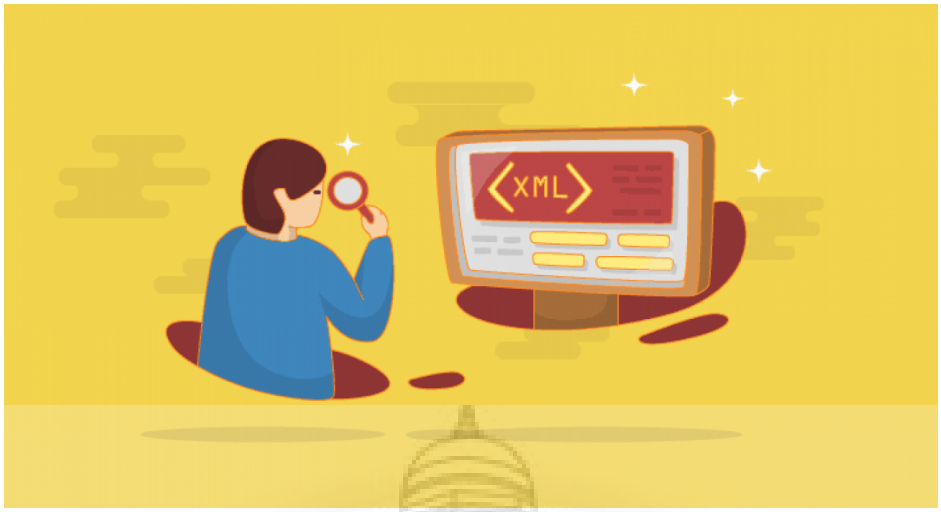


Figure 2.4: XML

Different XML Files Used in Android: In Android there are several xml files used for several different purposes. Below we define each and every one.

1. Layout XML Files: Layout xml files are used to define the actual UI(User interface) of our application. It holds all the elements(views) or the tools that we want to use in our application.

Like the TextView's,Button's and other UI elements. Location in Android Studio: You will find out this file inside the res folder and inside it there is another folder named layout where you will get all the layout files for their respective activities or fragments.

2. Manifest xml File(Mainfest.xml): This xml is used to define all the components of our application. It includes the names of our application packages, our Activities, receivers, services and the permissions that our application needs. For Example – Suppose we need to use internet in our app then we need to define Internet permission in this file. Location in Android Studio: It is located inside app ¿ manifests folder

3. Strings xml File(strings.xml): This xml file is used to replace the Hard-coded strings with a single string. We define all the strings in this xml file and then access them in our app(Activity or in Layout

XML files) from this file. This file enhance the reusability of the code.

4. Styles xml File(styles.xml): This xml is used to define different styles and looks for the UI(User Interface) of application. We define our custom themes and styles in this file

5. Drawable xml Files: These are those xml files that are used to provide various graphics to the elements or views of application. When we need to create a custom UI we use drawable xml files. Suppose if we need to define a gradient color in the background of Button or any custom shape for a view then we create a Drawable xml file and set it in the background of View.

6. Color xml File (colors.xml): This file is used to define the color codes that we used in our app. We simply define the color's in this file and used them in our app from this file.

7. Dimension xml File(dimens.xml): This xml file is used to define the dimensions of the View's. Suppose we need a Button with 50dp(density pixel) height then we define the value 50dp in dimens.xml file and then use it in our app from this file

2.7.3 Firebase

Firebase is a Backend-as-a-Service — BaaS — that started as a YC11 startup and grew up into a next-generation app-development platform on Google Cloud Platform.

Figure 2.5: Firebase

There are various sevices offered online such as storage, online processing, realtime database, authorisation of user etc. Google developed a platform called Firebase that provide all these online services. It also gives a daily analysis of usage of these services along with the details of user using it.

To simplify, it can be said that Firebase is a mobile and web application development platform. It provides services that a web application or mobile application might require. Anyone can easily include firebase to there application and it will make their online work way easier than it was used to be.

2.7.4 Android

Android is a mobile operating system (OS) based on the Linux kernel and currently developed by Google. With a user interface based on direct manipulation, Android is designed primarily for touch screen mobile devices such as Smartphone and tablet computers, with specialized user interfaces for televisions (Android TV), cars (Android Auto), and wrist watches (Android Wear).

Figure 2.6: Android

The OS uses touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, and a virtual keyboard. Despite being primarily designed for touch screen input, it also has been used in game consoles, digital cameras, regular PCs (e.g. the HP Slate 21) and other electronics.

Android is popular with technology companies which require a ready-made, low-cost and customizable operating system for high-tech devices. Android's open nature has encouraged a large community of developers and enthusiasts to use the open-source code as a foundation for community-driven projects, which add new features for advanced users or bring Android to devices which were officially released running other operating systems. The operating system's success has made it a target for patent litigation as part of the so-called "Smartphone wars" between technology companies

2.7.5 Android Studio

Figure 2.7: Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as :

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices Apply Changes to push code and resource changes to your running app without restarting your app
- Code templates and GitHub integration to help you build common app features and import sample code

2.7.6 Advantages of Technology

Java :

- Java is easy to learn. Java was designed to be easy to use and is therefore easy to write, compile, debug, and learn than other programming languages.
- Java is object-oriented. This allows you to create modular programs and reusable code.
- Java is platform-independent.

XML :

- XML is a very popular and widely-used format.
- It helps to provide separation of the UI from the code logic.
- Generating XML output is easier than writing direct code, making it easier to have drag-and-drop UI tools to generate interfaces for android apps.

Firebase :

- Reliable Extensive Databases.
- Fast Safe Hosting.
- Provides A Free Start to Newbies.
- Google Analytics.
- Firebase Cloud Messaging for Cross-Platform.
- Free Multi-Platform Firebase Authentication.

- Firebase Testing Services to Improve App Quality.
- Increment in Revenues with App Indexing API.

Android :

- Universal Chargers.
- More Phone Choices Are a Clear Advantage of Android.
- Removable Storage and Battery.
- Access to the Best Android Widgets.
- Better Hardware.
- Better Charging Options are Another Android Pro.
- Infrared.

Android Studio :

- Code and iterate faster than ever.

- Fast and feature-rich emulator.
- Code with confidence.
- Testing tools and frameworks.
- Configure builds without limits.
- Optimized for all Android devices.
- Create rich and connected apps.

2.7.7 Reasons to use this Technology

Java :
Java has platform independent feature so it is used for android development. Thus android developers to choose java as there is already a good base of java programmers are available that can help in creating, improving android applications plus with many libraries and tools of java make developers life easier.

XML :
XML stands for Extensible Markup Language. Much like HTML (or HyperText Markup Language), XML is also a markup language. In Android we use XML for designing our layouts because XML is lightweight language so it doesn't make our layout heavy.

Firebase :
Firebase is a platform developed by Google for creating mobile and web applications. Firebase is a Backend-as-a-Service (Baas). It provides with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google's infrastructure.

Android :

- Zero/negligible development cost. The development tools like Android SDK, JDK, and Eclipse IDE etc.
- Open Source.
- Multi-Platform Support.
- Multi-Carrier Support.
- Open Distribution Model

Android Studio :

- Market share.
- Profitability.
- Low barrier of entry.
- Google Play Store.
- Java.
- Portability. Native Android apps are developed using the Java programming language, and can easily be ported to other mobile operating systems like Blackberry, Symbian and Ubuntu.

Chapter 3

Project Planning

3.1 Members and Capabilities

SR. No	Name of Member	Capabilities
1	Arafat Honpode	Front-End and Back-End(Android App)
2	Adil Shaikh	Front-End and Back-End(Android App)
3	Salman Mahadkar	UI Design and Front-End(Android App)
4	Saif Khan	UI Design

Table 3.1: Table of Capabilities

Work Breakdown Structure :

1. All of the members are equally important in developing the project.
2. We work on a different part of the project based on one's capability.
3. Firstly we came up with documentation, And based on the documentation we set our goal and created a blueprint.
4. We then started going hands-on with the project to develop it according to the flow as decided earlier.

3.2 Roles and Responsibilities

SR. No	Name of Member	Role	Responsibilities
1	Arafat Honpode	Team Leader	Front and Back-End Development, Firebase integration.
2	Adil Shaikh	Member	Front and Back-End Development, Firebase integration.
3	Salman Mahadkar	Member	UI Design, Front-End Development, Documentation.
4	Saif Khan	Member	UI Design, Documentation.

Table 3.2: Table of Responsibilities

3.3 Assumptions and Constraints

- User of the app Should know how to use browser and Internet
- User should know how to deal with application.
- User should know working of application.
- Society members needs to register to enjoy smart services.

3.4 Project Management Approach

- Planning of project.
- Defining the scope of the project.
- Estimation of time and It's management.
- Creating Gantt Charts and properly assigning tasks to members.
- Reporting the progress of project with the guide

3.5 Ground Rules for the Project

- Properly planning and gathering relevant information is very important.
- Developing a Blueprint of the project and work accordingly.
- All the members should report to the guide whenever required.
- Setting up small goals every week.
- Achieving the small goal within that span of time.
- Keeping tracks of the progress towards project.
- Participate in meeting.
- Inform the leader about unavailability.

3.6 Project Budget

- It is a light project.
- Cost of the project is very low and efficient.
- Free version of Android studio is available.
- Firebase is open source

3.7 Project Timeline

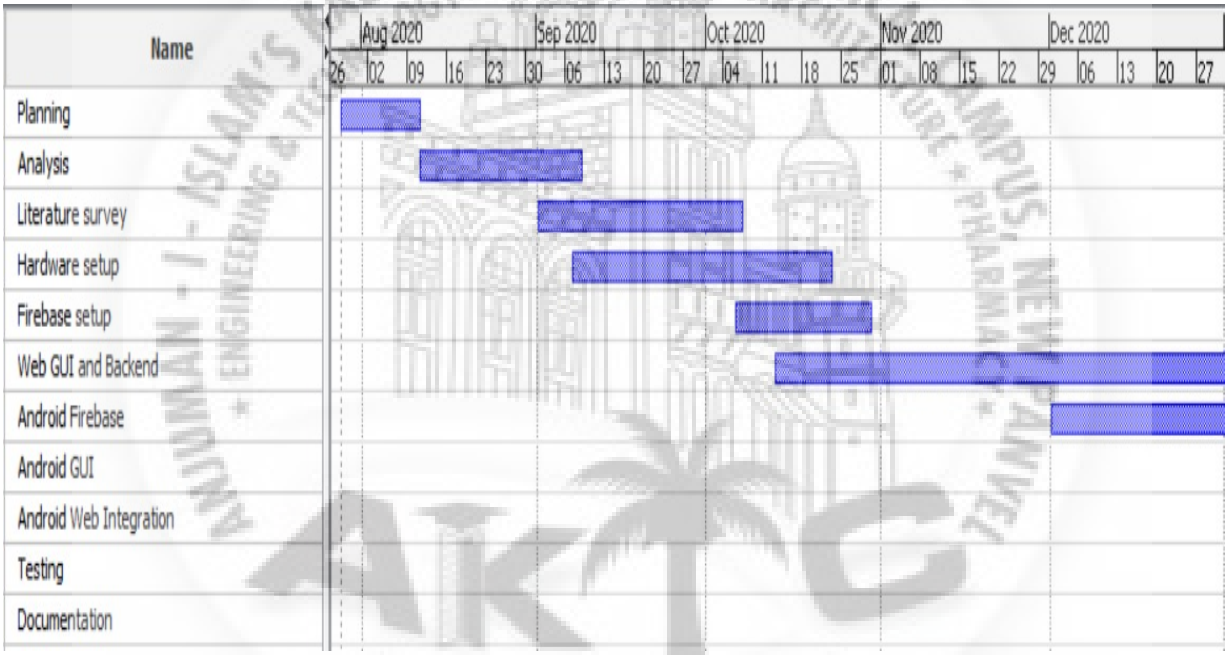


Figure 3.1: Project Timeline.

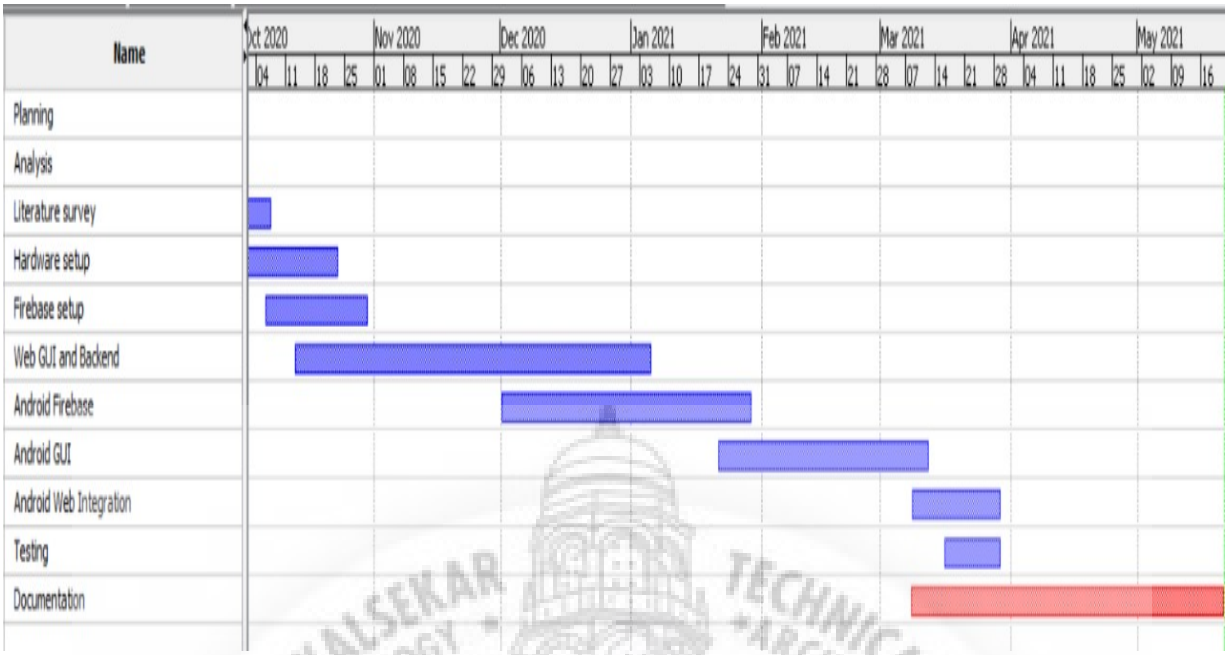


Figure 3.2: Project Timeline Continuation.

Chapter 4

Software Requirements Specification

4.1 Overall Description

This section gives a scope description and overview of everything included in this SRS document. It provides all the detail about SRS including its different functions, role of particular users, requirement of different hardware and software required.

4.1.1 Product Perspective

The system consists of user and admin. User will be interacting using android application and admin will be managing the system using admin panel/admin dashboard. The user is the society member that can make maintenance payments, read notices, access contact information of essential personnel, register complaint(s) and track the complaint(s) status, etc.

4.1.2 Product Features

The admin is the committee member that can add notices, add contact information of essential personnel, resolve the complaint(s), mark the complaint(s) status accordingly and can also make maintenance payments, etc
The users module involves Login/Sign Up, Join your building, Profile, Maintenance payment, Register Complaint(s), Track Complaint(s), Contact information of essential personnel and View Notice(s), etc. The admin module involves Login/Sign Up,create new Building, Profile,

Maintenance payment,Mark the complaint(s)status, Add Contact information of essential personnel and Add Notice(s), etc

4.1.3 User Classes and Characteristics

The project is an Android Application for society. User of project includes society members and others. An app is created to handle both the users data by super-user (admin). All the user should have knowledge of Internet and should have knowledge about how to use an android phone. Admin should know how to use

4.1.4 Software Requirements

- Operating System: Linux or Windows(7 and above)
- Adroid Studio
- FireBase

4.1.5 Hardware Requirements

- PC with 4 GB RAM.
- 2 GB of available disk space.
- 1280 x 800 minimum screen resolution.
- 2.3 GHz Fast processor.

4.1.6 Design and Implementation Constraints

The product is made using android studio hence, only android phone users can use this application. User may access the product using any android device. The information of all the users, notices, member data must be stored in database. Internet connectivity is the main source to use the product. Admin should use correct username and password to manage databse.

4.2 System Features

- Login/Sign Up: If User or Admin are already registered, they can login through Google API or Email-Id and if user or admin is not register they can register/Sign Up through Google API or Email-Id, if the user or admin is registering using email-id they will have to fill the fields like first name, last name, Email-id,password,phone number. After clicking on register/Sign Up, the user or admin have to confirm the registration of account by clicking the link sent in the email-id.
- Join your Building: User can join the specific building by clicking the Join your Building button, then entering building unique code.
- Create new Building: Admin can create a new building by clicking Create new Building button, the admin will specify the no of. flats, floors and wings.
- Profile: User or admin can see his/her profile where he/she can see details like name, phone no, email-id, building name, etc. User or Admin can change his/her password.
- Maintenance Payment: User or Admin can pay his/her maintenance bill through online payment methods like Google Pay, Paytm, PhonePe, etc.
- Register Complaint(s): User can register the complaint(s) by specifying the level of critical, category.
- Complaint(s) status: User can track his/her status of the complaint(s), where admin resolve the complaint(s) and mark the complaint(s) status accordingly.
- View Notice(s): User or admin can view the Notices(s) of society.
- Add Notice(s): Admin can add notice(s) of society.
- View Contacts: User or Admin can view the Contact information of essential personnel.

- Add Contacts: Admin can add the Contact information of essential personnel

Functional Requirements

- REQ-1: Users are limited to Android handsets.
- REQ-2: Admin is limited to Linux or Windows(7 and above).
- REQ-3: Access to the Databases.
- REQ-4: Access to Internet.

4.3 External Interface Requirements

4.3.1 User Interfaces

Android provides a variety of pre-built UI components such as structured layout objects and UI controls that allow to build the graphical user interface for app. Android also provides other UI modules for special interfaces such as dialogs, notifications,navigation drawer, and menus. The navigation drawer consist of user profile details which can be easily updated,booking history of services. The notification pop-up when user request for the services and also messages are sent to the respected users. The GUI is very simple. Home page is nothing but the firebase console which contain all users data

4.3.2 Hardware Interfaces

This application works on android handset with API 19 and above(kitkat) version.

4.3.3 Software Interfaces

Since this application is a mobile application, it will only need an Android kitkat version API 19 or higher in order to perform. Database is maintained in Firebase.

4.3.4 Communications Interfaces

The application uses internet to communicate with user.In case of any difficulties while using application user can contact through whatsApp or call. On Admin side the product is a light web, there is no such large communication in the system. Only Databases access, that also done locally.

4.4 Nonfunctional Requirements

4.4.1 Performance Requirements

Performance of overall system is very efficient and well optimize. The time taken to show various announcements on notice board section would take a sec. Process and everything is well organized. The messages related to any transaction or other will be delivered to the users in a very short time.

4.4.2 Safety Requirements

Login and sign up must be authenticated for the pre-existing users. Data of every user should maintain.

4.4.3 Security Requirements

Sign In: Only registered user can access his/her account.
Sign Up: No duplicate of the data of the user should be there.

Chapter 5

System Design

5.1 System Requirements Definition

System requirement definitions specify what the system should do, its functionality and its essential and desirable system properties. The techniques applied to elicit and collect information in order to create system specifications and requirement definitions involve consultations, interviews, requirements workshop with customers and end users. The objective of the requirements definition phase is to derive the two types of requirement:

5.1.1 Functional requirements

They define the basic functions that the system must provide and focus on the needs and goals of the end users.

Use-case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

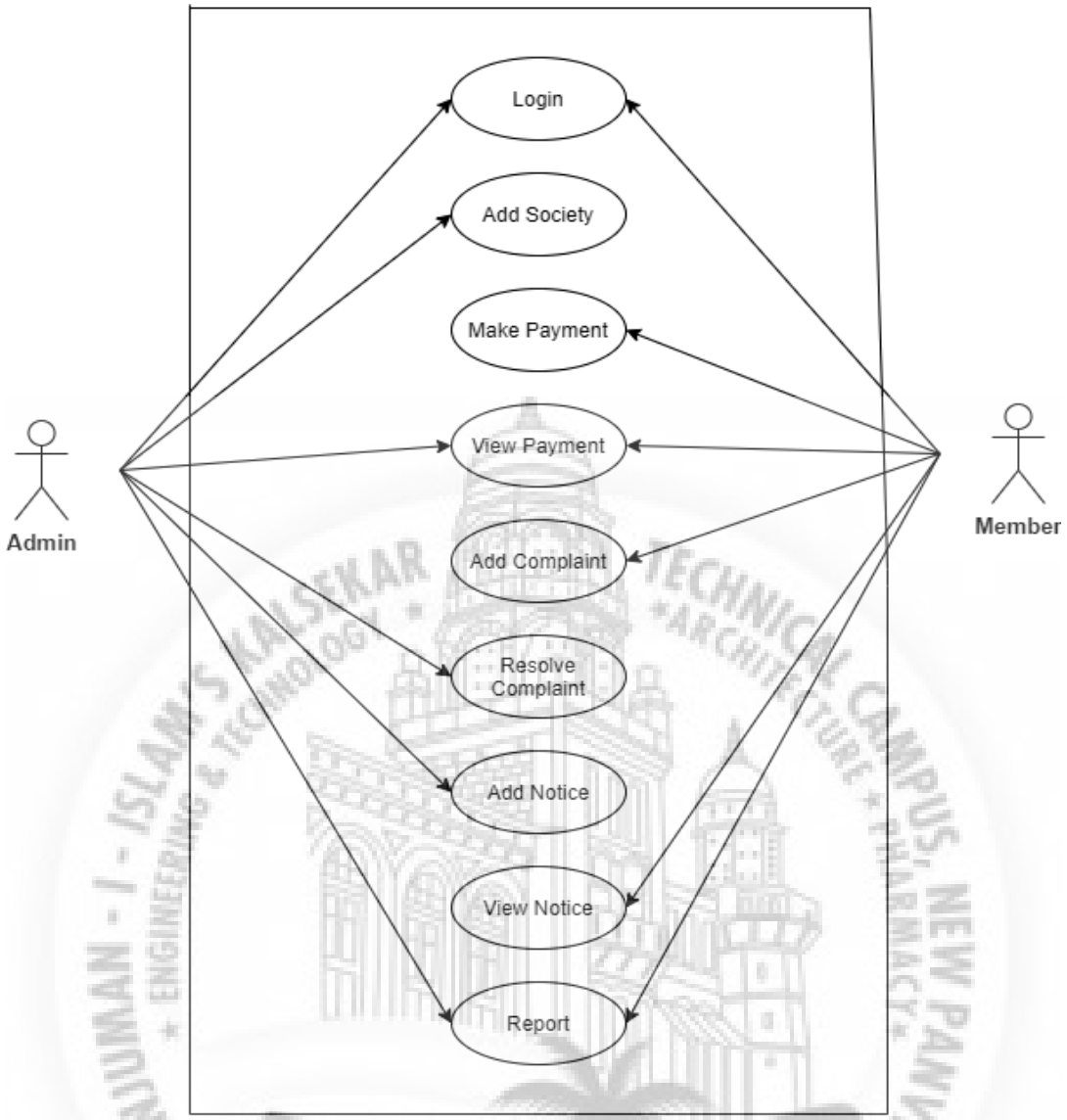


Figure 5.1: Use Case of 'Smart Society App'.

Data-flow Diagram

A data-flow diagram is a way of representing a flow of a data of a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. Given below is Level 0 Level 1 and Level 2 DFD of system.

Figure 5.2: DFD Level 0

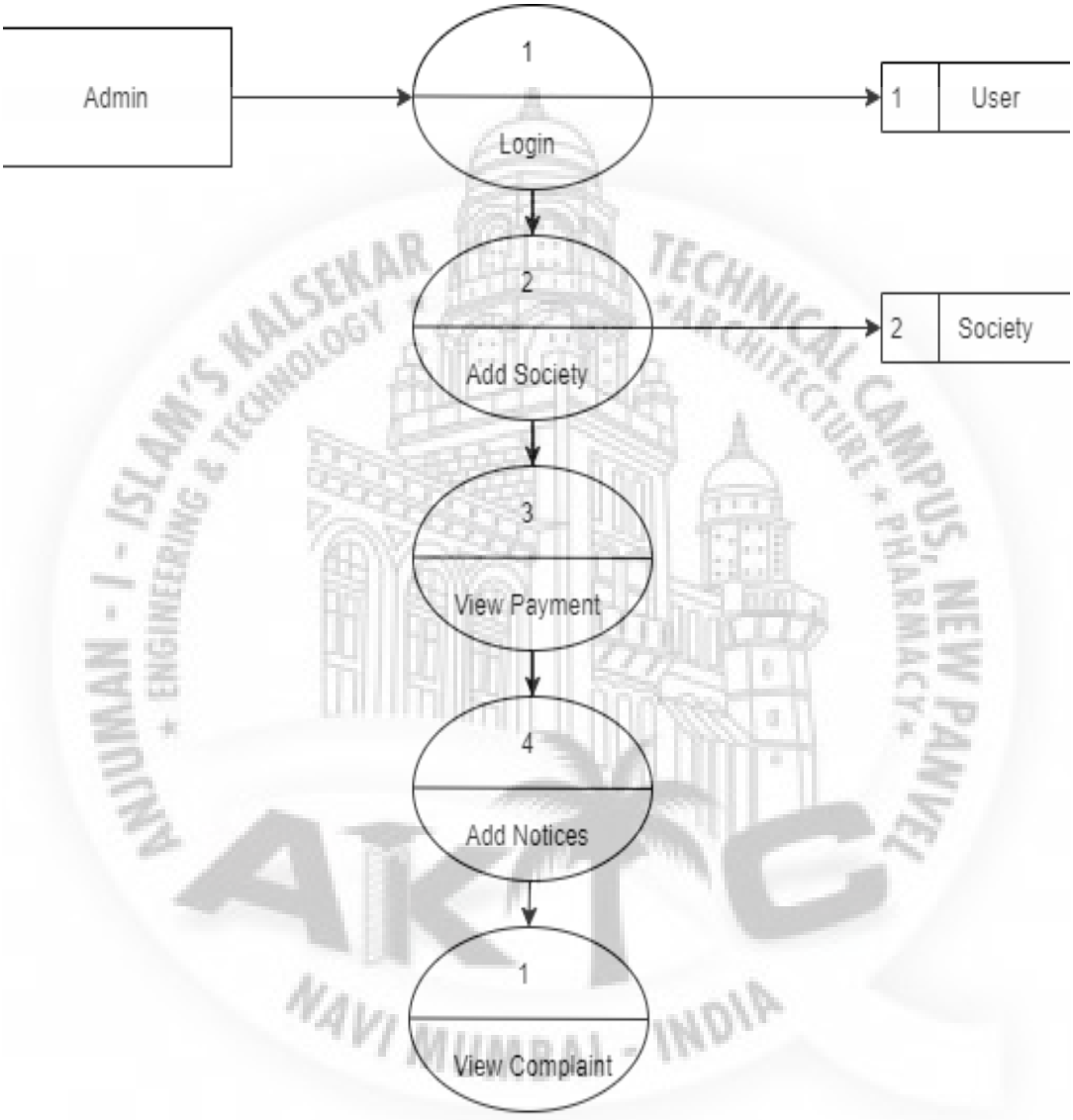


Figure 5.3: DFD Level 1 for Admin

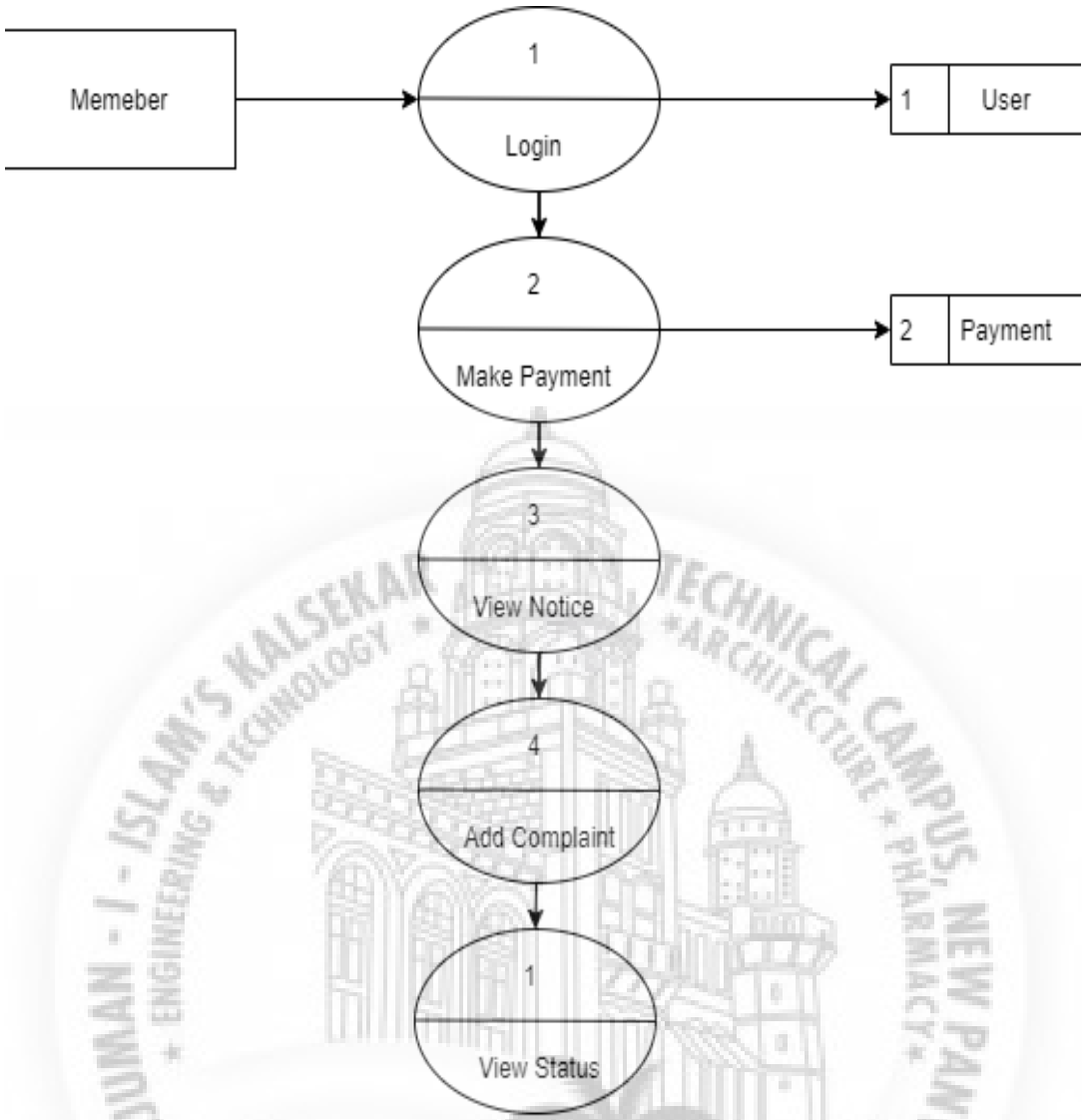


Figure 5.4: DFD Level 1 for User

5.1.2 System requirements (non-functional requirements)

These are non-functional system properties such as availability, performance and safety etc. They define functions of a system, services and operational constraints in detail.

- Usability - Application implementation is feasible using technologies that are
- accessible to the end-users.

- Portability - The interfaces are compatible with Web View and Mobile view.

- Performance Efficiency -Application is able to perform well in a proper time constraint.

- Multi User System -Application is able to consider the presence of more than one user in the same environment. All the features of the system operates properly for all users and provides proper transparency

- Time Efficiency - Time taken for the executing of system is less.

5.2 System Architecture Design

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

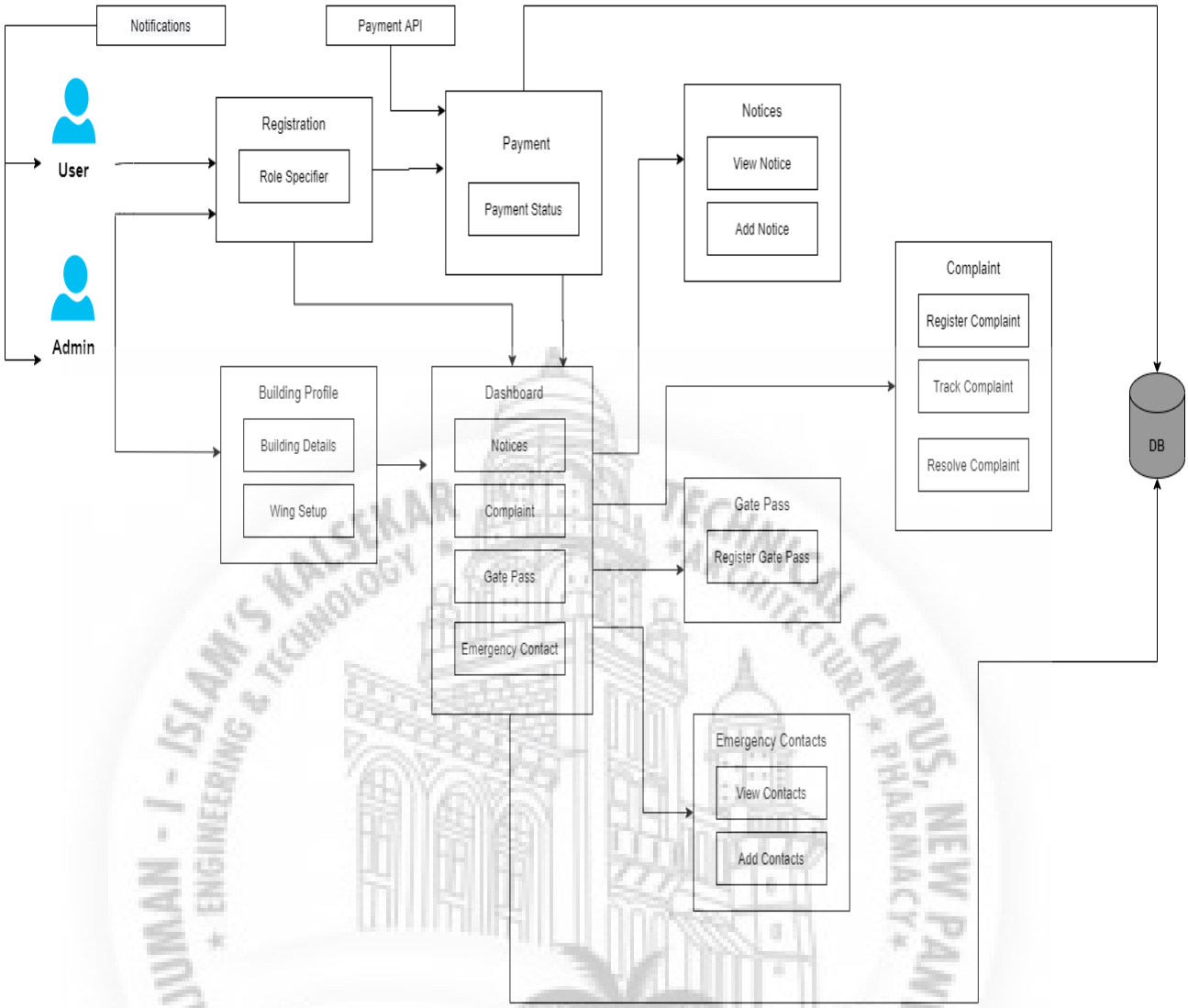


Figure 5.5: System Architecture

5.3 Sub-system Development

5.3.1 FlowChart

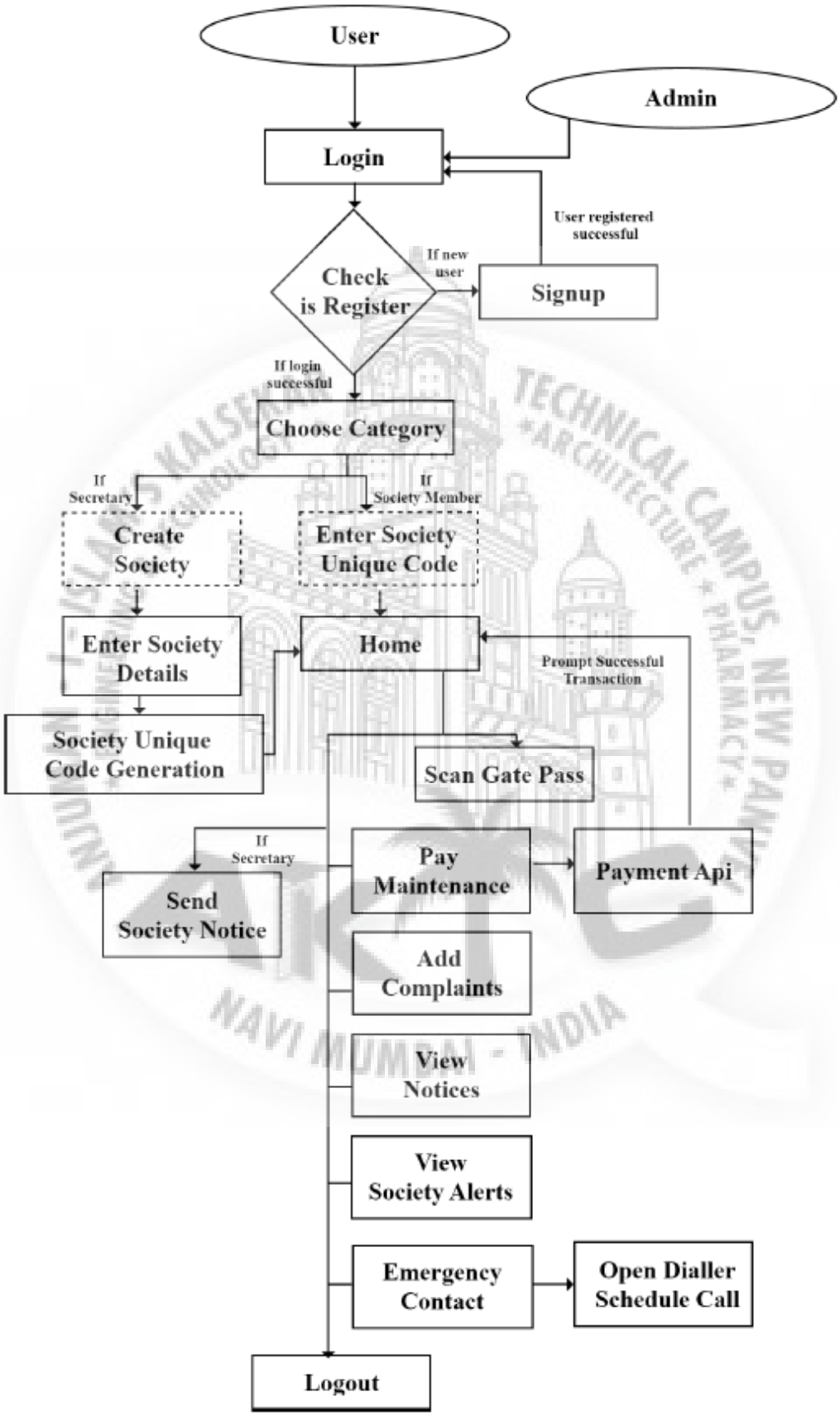


Figure 5.6: Flow-chart for 'smart society app'.

5.3.2 Activity Diagram

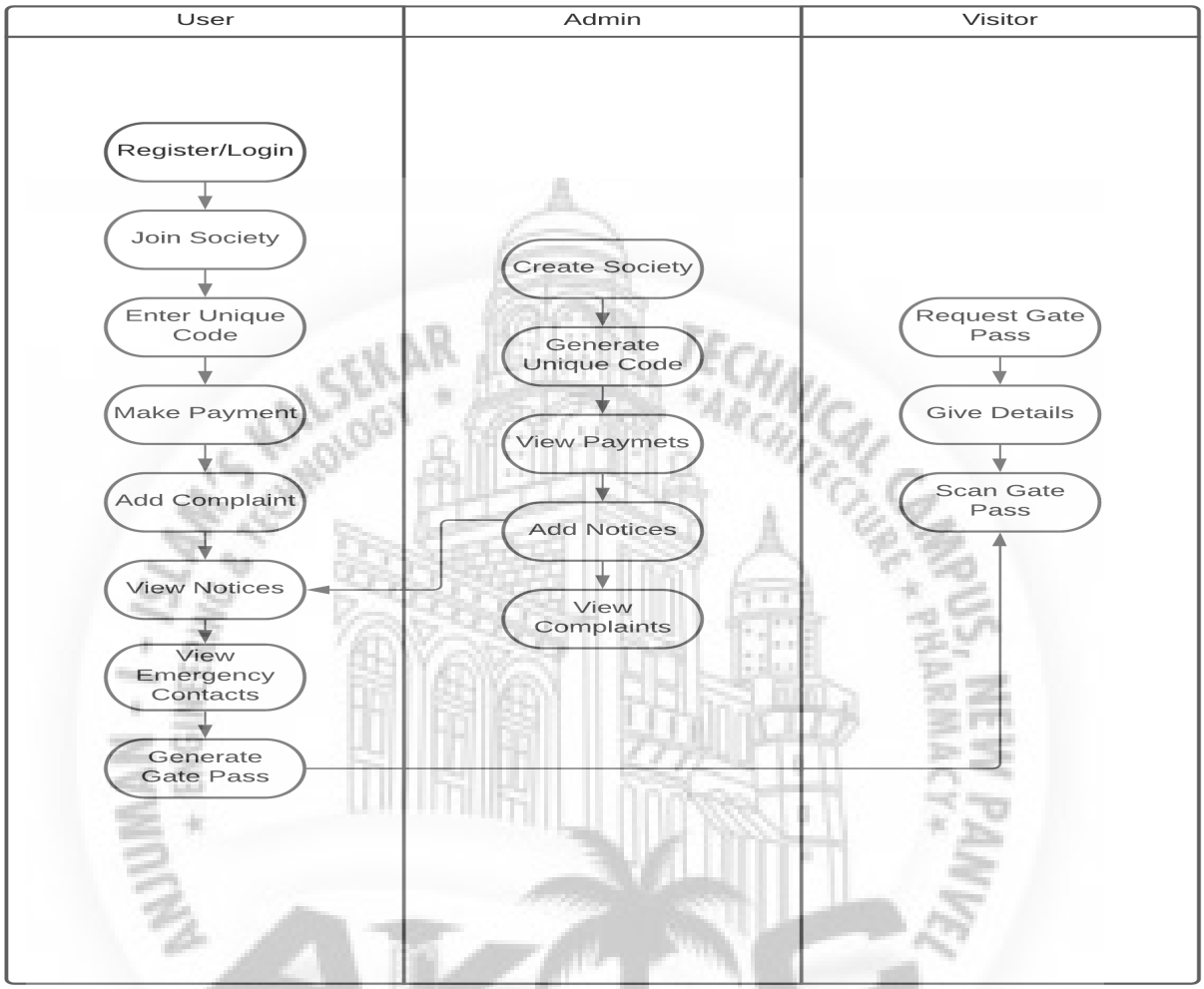


Fig : Activity Diagram for Smart Society App

Figure 5.7: Activity Diagram.

5.4 Systems Integration

System integration (SI) is an engineering process or phase concerned with joining different subsystems or components as one large system. It ensures that each integrated subsystem functions as required. Different Sub-Modules Integrated in one full System. SI is

also used to add value to a system through new functionalities provided by connecting functions of different systems.

5.4.1 Class Diagram

A class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations, and the relationships among objects.

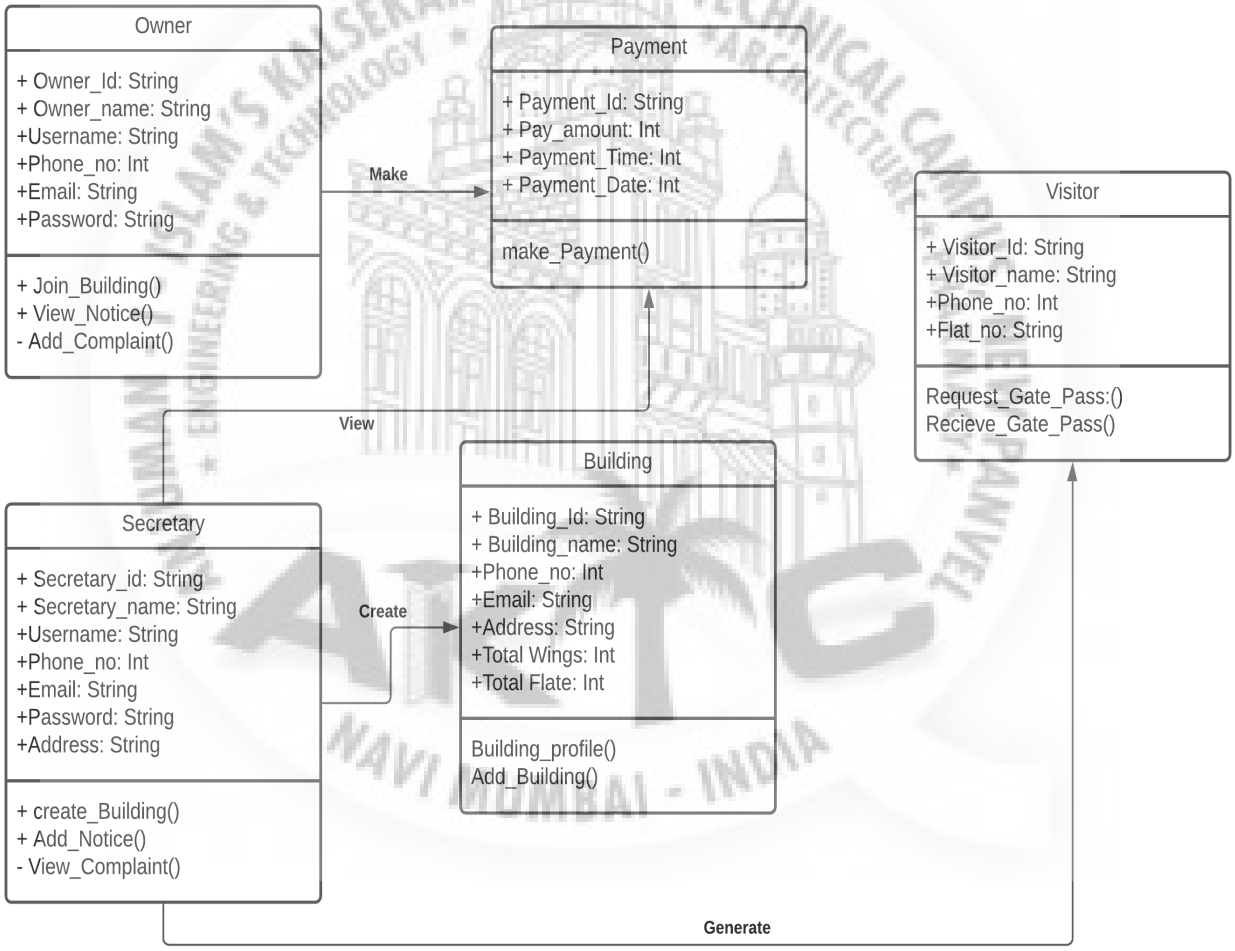


Fig : Class Diagram for Smart Sosciety App

Figure 5.8: Class Diagram.

5.4.2 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

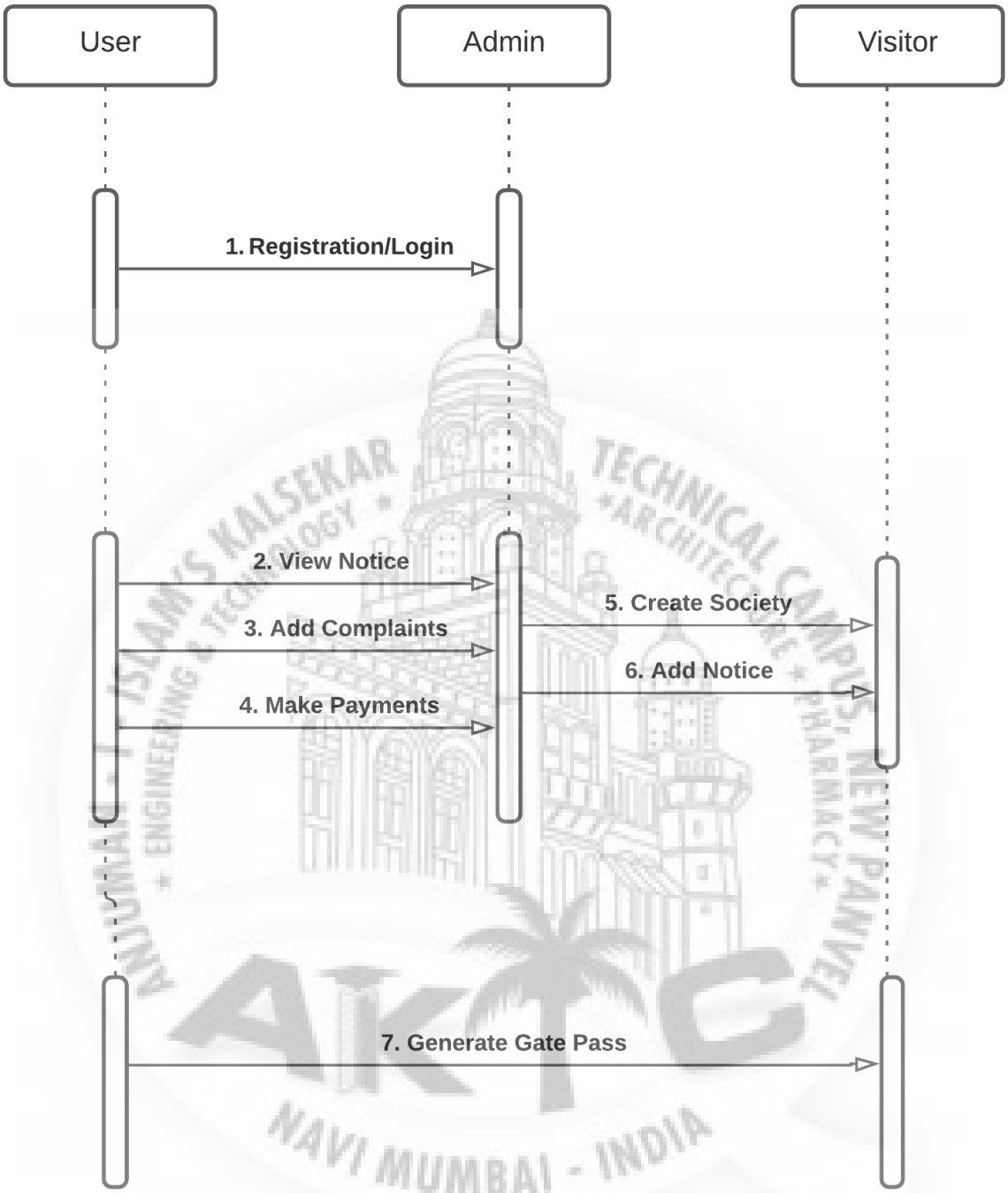


Fig: Sequence Diagram for Smart Society App

Figure 5.9: Sequence Diagram.

5.4.3 Component Diagram

Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.

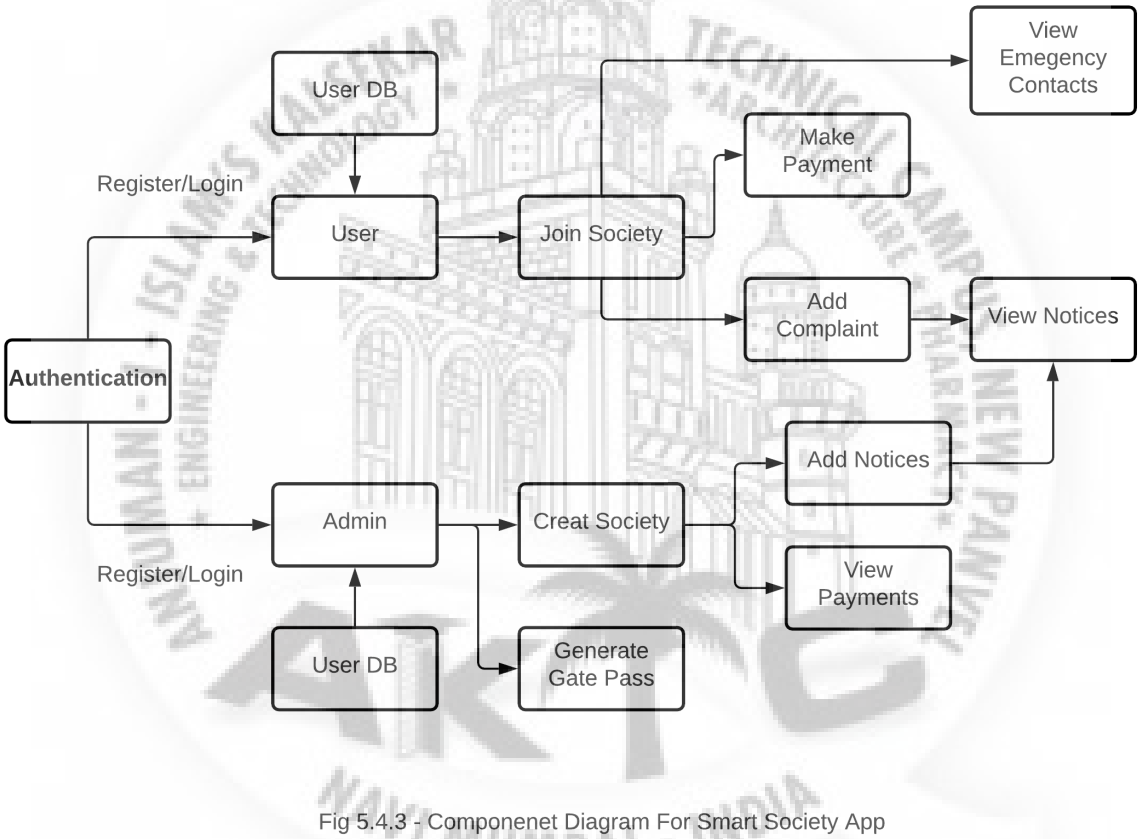


Figure 5.10: Component Diagram

Chapter 6

Implementation

6.1 Module 1 - Admin View (Society Secretary)

In this module we have build different features for the Admin (Society Secretary). Where the Admin can do various tasks like Adding Notices for the society members, Viewing Complaints of the society members, able to call Emergency Contacts in the society, Generate Gate Pass for the visitors and to see the Visitors List who are entering in the society .

Choose Category Activity.java

Here the admin will select or click on the Create Society Button.

```
package com.example.mysmartsociety;

public class ChooseCategoryActivity extends AppCompatActivity {


    private FirebaseDatabase db = FirebaseDatabase.getInstance();
    private DatabaseReference root = db.getReference().child("society");

    EditText uniqueCode;
    CategoryPreference cPrefrence;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_choose_category);
        cPrefrence = CategoryPreference.getInstance(ChooseCategoryActivity.this)
            ;
        uniqueCode = findViewById(R.id.edt_unique_code);

    }
    public void On_Society_Member(View view) {

        String code = uniqueCode.getText().toString();

        if (CheckInternetConnectivity.isInternet(ChooseCategoryActivity.this)) {
        root.orderByChild("Society_unique_code").equalTo(code).
            addListenerForSingleValueEvent(new ValueEventListener() {
                @Override
```

```
27                public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
28                    if(dataSnapshot.exists()) {
29
30                        for (DataSnapshot snapshot : dataSnapshot.getChildren())
                              {
31                            Map <String, String> map = (Map)snapshot.getValue();
32
33                            String Suniquecode = map.get("Society_unique_code");
34                            String Sname = map.get("Society_name");
35                            String Sadd = map.get("Society_add");
36                            String Sflats = map.get("Society_flats");
37                            String Swings = map.get("Society_wings");
38                            String manager = map.get("Society_manager");
39                            String secretory = map.get("Society_secretory");
                                                        cPrefrence.saveData("
                                  c_unique_code",Suniquecode);
40                            cPrefrence.saveData("c_name",Sname);
41                            cPrefrence.saveData("c_add",Sadd);
42                            cPrefrence.saveData("c_flats",Sflats);
43                            cPrefrence.saveData("c_wings",Swings);
44                            cPrefrence.saveData("c_manager_no",manager);
45                            cPrefrence.saveData("c_secretory_no",secretory);
46                        }
47
48                        cPrefrence.saveData("c_user_type","Society Member");
49                        startActivity(new Intent(ChooseCategoryActivity.this,
                              HomeActivity.class));
50                        ToastUtils.showToastShort(ChooseCategoryActivity.this, "
                              You have successfully verified    ");
51                        finish();
52                    } else{
53                        ToastUtils.showToastShort(ChooseCategoryActivity.this, "
                              Invalid Unique Code    ");
54                    }
55                }
56                @Override
57                public void onCancelled(@NonNull DatabaseError databaseError) {
58                    ToastUtils.showToastShort(ChooseCategoryActivity.this, "
                          Error:" + databaseError);
59                }
60            });
61        } else {
62            ToastUtils.showToastLong(ChooseCategoryActivity.this, "No Internet
                  Connection!!!");
63        }
64    }
65    public void On_Create_Society(View view) {
66        startActivity(new Intent(getApplicationContext(), CreateSocietyActivity.
              class));
67    }
68
69    @Override
70    public void onBackPressed() {
71        Clear_login_Instance();
72        ToastUtils.showToastShort(ChooseCategoryActivity.this,"Process cancelled
              !");
73        finish();
74    }
75
76    /* @Override
77     protected void onDestroy() {
```

```
78          Clear_login_Instance();
79          super.onDestroy();
80      }*/
81
82      public void Clear_login_Instance() {
83          if (CheckInternetConnectivity.isInternet(ChooseCategoryActivity.this)) {
84                  FirebaseAuth.getInstance().signOut();
85                 // ToastUtils.showToastShort(ChooseCategoryActivity.this, "You
                      have successfully Logout!");
86                  this.finish();
87          } else {
88              ToastUtils.showToastLong(ChooseCategoryActivity.this, "No Internet
                    Connection!!!");
89          }
90      }
91  }
```

CreateSocietyActivity.java

Here the admin will fill-up the required details for the new society.

```
1   package com.example.mysmartsociety;
2
3   public class CreateSocietyActivity extends AppCompatActivity {
4
5       EditText societyName, societyAdd, societyFlats, societyWings, ManagerNo,
            SecretoryNo;
6       LinearLayout dialog_layout;
7       TextView UniqueId;
8
9       private FirebaseDatabase db = FirebaseDatabase.getInstance();
10      private DatabaseReference root = db.getReference().child("society");
11      private FirebaseAuth mAuth;
12
13      ProgressBar uploadProgress;
14      CategoryPreference cPrefrence;
15
16      @Override
17      protected void onCreate(Bundle savedInstanceState) {
18          super.onCreate(savedInstanceState);
19          setContentView(R.layout.activity_create_society);
20
21          cPrefrence = CategoryPreference.getInstance(CreateSocietyActivity.this);
22
23          uploadProgress  = findViewById(R.id.progress_circular);
24          dialog_layout = findViewById(R.id.dialog_layout);
25          UniqueId = findViewById(R.id.unique_code);
26
27          societyName = findViewById(R.id.edt_society_name);
28          societyAdd = findViewById(R.id.edt_add);
29          societyFlats = findViewById(R.id.edt_flats);
30          societyWings = findViewById(R.id.edt_wings);
31          ManagerNo = findViewById(R.id.edt_manager_no);
32          SecretoryNo = findViewById(R.id.edt_secretory_no);
33
34          // Initialize Firebase Auth
35          mAuth = FirebaseAuth.getInstance();
36      }
37
38      public void On_save(View view) {
```

```
39
40          if (CheckInternetConnectivity.isInternet(CreateSocietyActivity.this)) {
41              if(societyName.getText().toString().isEmpty()){
42                  societyName.setError("Please enter society name!");
43              }else if(societyAdd.getText().toString().isEmpty()){
44                  societyAdd.setError("Enter Society Address!");
45              }else if(societyFlats.getText().toString().isEmpty()){
46                  societyFlats.setError("Enter total flats!");
47              }else if(societyWings.getText().toString().isEmpty()){
48                  societyWings.setError("Enter wings");
49              }else if(ManagerNo.getText().toString().isEmpty()){
50                  ManagerNo.setError("Enter Manager Phone Number!");
51              }else if(SecretoryNo.getText().toString().isEmpty()){
52                  SecretoryNo.setError("Enter Secretory Phone Number!");
53              }
54              else {
55                  uploadProgress.setVisibility(View.VISIBLE);
56
57                  String email = "";
58                  if (mAuth.getCurrentUser() !=null)
59                      email = mAuth.getCurrentUser().getEmail();
60
61                  String Sname = societyName.getText().toString();
62                  String Sadd = societyAdd.getText().toString();
63                  String Sflats = societyFlats.getText().toString();
64                  String Swings = societyWings.getText().toString();
65                  String manager = "+91"+ ManagerNo.getText().toString();
66                  String secretory = "+91" + SecretoryNo.getText().toString();
67
68                  String i = GenerateRandomString.randomString(6);
69                  UniqueId.setText(i);
70
71                  cPrefrence.saveData("c_unique_code",i);
72                  cPrefrence.saveData("c_name",Sname);
73                  cPrefrence.saveData("c_add",Sadd);
74                  cPrefrence.saveData("c_flats",Sflats);
75                  cPrefrence.saveData("c_wings",Swings);
76                  cPrefrence.saveData("c_manager_no",manager);
77                  cPrefrence.saveData("c_secretory_no",secretory);
78                  cPrefrence.saveData("c_email",email);
79
80                  HashMap<String , String> userMap = new HashMap<>();
81                  userMap.put("Society_name" , Sname);
82                  userMap.put("Society_add",Sadd);
83                  userMap.put("Society_flats" , Sflats);
84                  userMap.put("Society_wings",Swings);
85                  userMap.put("Society_manager" , manager);
86                  userMap.put("Society_secretory",secretory);
87                  userMap.put("Society_unique_code",i);
88                  userMap.put("email",email);
89
90                  root.child(Sname).setValue(userMap).addOnCompleteListener(new
                        OnCompleteListener<Void>() {
91                      @Override
92                      public void onComplete(@NonNull Task<Void> task) {
93                          uploadProgress.setVisibility(View.GONE);
94                          dialog_layout.setVisibility(View.VISIBLE);
95                          ToastUtils.showToastShort(CreateSocietyActivity.this,"
                                Society Created successfully");
96                      }
97                  });
```

```
            }
        } else {
            ToastUtils.showToastLong(CreateSocietyActivity.this, "No Internet
                Connection!!!");
        }
    }

    public static class GenerateRandomString {

        public static final String DATA = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
            ;
        public static Random RANDOM = new Random();

        public static String randomString(int len) {
            StringBuilder sb = new StringBuilder(len);

            for (int i = 0; i < len; i++) {
                sb.append(DATA.charAt(RANDOM.nextInt(DATA.length())));
            }
            return sb.toString();
        }
    }

    public void On_Got_it(View view) {
        dialog_layout.setVisibility(View.GONE);
        cPrefrence.saveData("c_user_type","Secretory");
        startActivity(new Intent(CreateSocietyActivity.this,HomeActivity.class))
            ;
        this.finish();
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
    }
}
```

HomeActivity.java

Here Admin Dashboard will be shown.

```
package com.example.mysmartsociety;

public class HomeActivity extends AppCompatActivity {
    private static final String TAG = HomeActivity.class.getName();
    private FirebaseAuth mAuth;
    FirebaseUser user;
    GoogleSignInClient mGoogleSignInClient;

    LinearLayout dialog_layout;
    TextView UserEmail, UserType;

    CategoryPreference cPrefrence;
    String UserTypeValue;
    Button Notice, Complaints;

    private FirebaseDatabase db = FirebaseDatabase.getInstance();
    private DatabaseReference visitorsRef;

    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
        mAuth = FirebaseAuth.getInstance();
        FirebaseMessaging.getInstance().subscribeToTopic("all");
        dialog_layout = (LinearLayout) findViewById(R.id.dialog_layout);
        UserEmail = findViewById(R.id.user_email);
        UserType = findViewById(R.id.user_type);

        cPrefrence = CategoryPreference.getInstance(HomeActivity.this);

        UserTypeValue = cPrefrence.getData("c_user_type");
        UserType.setText(UserTypeValue);

        //if logged in as Secretory
        if(UserTypeValue.equals("Secretory")){
            Notice = findViewById(R.id.btn_notices);
            Complaints = findViewById(R.id.btn_complaints);

            Notice.setText("View Notices");
            Complaints.setText("View Complaints");
        }
    }

    private void updateUI(FirebaseUser user) {
        // hideProgressBar();
        if (user != null) {
            UserEmail.setText(user.getEmail());
        } else {
            UserEmail.setText(null);
        }
    }

    @Override
    public void onStart() {
        super.onStart();
        // Check if user is signed in (non-null) and update UI accordingly.
        FirebaseUser currentUser = mAuth.getCurrentUser();
        updateUI(currentUser);
    }
    //dialog open
    public void On_Open_Dialog(View view) {
        if (dialog_layout.getVisibility() == View.GONE) {
            dialog_layout.setVisibility(View.VISIBLE);
        }
    }
    //dialog close
    public void On_Inner_Dilaog_Layout(View view) {
        if (dialog_layout.getVisibility() == View.VISIBLE) {
            dialog_layout.setVisibility(View.GONE);
        }
    }

    public void On_Logout(View view) {
        if (CheckInternetConnectivity.isInternet(HomeActivity.this)) {
            if (mAuth.getCurrentUser() != null) {
                FirebaseAuth.getInstance().signOut();
                UserEmail.setText(null);
                cPrefrence.clearData();
                ToastUtils.showToastShort(HomeActivity.this, "You have
                    successfully Logout!");
```

```
80                  this.finish();
81                  startActivity(new Intent(HomeActivity.this,LoginActivity.class))
                        ;
82              }else if(user != null){
83                  mGoogleSignInClient.signOut().addOnCompleteListener(this,
84                          new OnCompleteListener<Void>() {
85                              @Override
86                              public void onComplete(@NonNull Task<Void> task) {
87                                  UserEmail.setText(null);
88                              }
89                          });
90                  this.finish();
91                  startActivity(new Intent(HomeActivity.this,LoginActivity.class))
                        ;
92                  ToastUtils.showToastShort(HomeActivity.this, "You have
                        successfully Logout!");
93              }
94              else {
95                  ToastUtils.showToastShort(HomeActivity.this, "Unable to Logout!"
                        );
96              }
97          } else {
98              ToastUtils.showToastLong(HomeActivity.this, "No Internet Connection
                    !!!");
99          }
100     }
101
102     public void On_Notification(View view) {
103         startActivity(new Intent(HomeActivity.this, NotificationActivity.class))
                ;
104     }
105
106     public void On_Visitor_List(View view) {
107         startActivity(new Intent(HomeActivity.this, VisitorListActivity.class));
108     }
109
110     //society details
111     public void On_Society_Details(View view) {
112         On_Inner_Dilaog_Layout(view);
113         startActivity(new Intent(HomeActivity.this, MySocietyDetailsActivity.
                class));
114     }
115
116     //notice btn
117     public void On_Notices(View view) {
118         if(UserTypeValue.equals("Secretory")){
119             startActivity(new Intent(HomeActivity.this, NoticesActivity.class ))
                    ;
120         }else{
121             startActivity(new Intent(HomeActivity.this, ViewAllNoticesActivity.
                    class ));
122         }
123
124     }
125
126     //complaints btn
127     public void On_Add_Complaints(View view) {
128         if(UserTypeValue.equals("Secretory")){
129             startActivity(new Intent(HomeActivity.this, AddComplaintsActivity.
                    class));
130         }else{
```

```
            startActivity(new Intent(HomeActivity.this, ViewComplaintsActivity.
                class));
        }
    }
    //Pay btn
    public void On_Pay(View view){
        startActivity(new Intent(HomeActivity.this, PayActivity.class));
    }

    //emergency btn
    public void On_Emergency_Contact(View view) {
        startActivity(new Intent(HomeActivity.this, EmergencyActivity.class));
    }

    //Scan gate pass
    public void On_Scan_Gatepass(View view) {
        IntentIntegrator intentIntegrator = new IntentIntegrator(HomeActivity.
            this);
        intentIntegrator.setDesiredBarcodeFormats(intentIntegrator.QR_CODE_TYPES
            );
        intentIntegrator.setBeepEnabled(true);
        intentIntegrator.setCameraId(0);
        intentIntegrator.setPrompt("SCANNING GATE-PASS...");
        intentIntegrator.setBarcodeImageEnabled(false);
        intentIntegrator.initiateScan();
    }

    //after scanning the QR Code this method get the result and check
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data
        ) {
        IntentResult Result = IntentIntegrator.parseActivityResult(requestCode,
            resultCode, data);
        if (Result != null) {
            if (Result.getContents() == null) {
                //if scanning process cancelled by the user
                ToastUtils.showToastShort(HomeActivity.this, "Process Cancelled!
                    ");
            } else {
                //check society member using unique code if is valid or not
                try {
                    Gson g = new Gson();
                    JsonObject object = g.fromJson(Result.getContents(),
                        JsonObject.class);

                    ToastUtils.showToastShort(HomeActivity.this, "You have
                        Successfully Verified");
                    visitorsRef = db.getReference().child("visitors"+"/"+object.
                        get("Society_unique_code").getAsString());

                    String id = visitorsRef.push().getKey();
                    object.addProperty("id" , id);
                    object.addProperty("checkInTimestamp" , System.
                        currentTimeMillis());

                    HashMap<String, Object> mapObj = new Gson().fromJson(
                            object, new TypeToken<HashMap<String, Object>>() {}.
                                getType()
                    );
                    visitorsRef.child(id).setValue(mapObj);

```

```
                    Intent intent = new Intent(HomeActivity.this,
                        VisitorDetailActivity.class);
                    intent.putExtra("data", mapObj);
                    startActivity(intent);
                }
                catch (Exception ex) {
                    ex.printStackTrace();
                    ToastUtils.showToastShort(HomeActivity.this, "Sorry, unable
                        to verify!");
                }
            }
        } else {
            super.onActivityResult(requestCode, resultCode, data);
        }
    }

    @Override
    public void onBackPressed() {
        if (dialog_layout.getVisibility() == View.VISIBLE) {
            dialog_layout.setVisibility(View.GONE);
        } else {
            finish();
        }
    }

    public void On_generate_Gatepass(View view) {
        startActivity(new Intent(HomeActivity.this, CreateVisitorActivity.class)
            );
    }
}
```

NoticeActivity.java

Here Admin will be able to Add Notices and View the Notices as well.

```
package com.example.mysmartsociety;

public class NoticesActivity extends AppCompatActivity {

    private static final int CHOOSE_IMAGE = 1;
    //TextView viewGallery;
    ImageView imgPreview;
    EditText imgDescription;
    ProgressBar uploadProgress;
    private Uri imgUrl;

    private StorageReference mStorageRef;
    private DatabaseReference mDatabaseRef;

    private StorageTask mUploadTask;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_notices);

        uploadProgress = findViewById(R.id.uploadProgress);

        imgDescription = findViewById(R.id.imgDescription);
        imgPreview = findViewById(R.id.imgPreview);
```

```
26
27          mStorageRef = FirebaseStorage.getInstance().getReference("
                notices_uploads");
28          mDatabaseRef = FirebaseDatabase.getInstance().getReference("
                notices_uploads");
29
30      }
31
32      public void On_Choose_Image(View view) {
33          Intent intent = new Intent();
34          intent.setType("image/*");
35          intent.setAction(Intent.ACTION_GET_CONTENT);
36          startActivityForResult(intent, CHOOSE_IMAGE);
37      }
38
39      public void On_Upload_Image(View view) {
40          if (CheckInternetConnectivity.isInternet(NoticesActivity.this)) {
41              if(imgDescription.getText().toString().isEmpty()){
42                  imgDescription.setError("Please enter notice description!");
43              }else {
44                  uploadImage();
45              }
46          } else {
47              ToastUtils.showToastLong(NoticesActivity.this, "No Internet
                    Connection!!!");
48          }
49      }
50
51      @Override
52      protected void onActivityResult(int requestCode, int resultCode, @Nullable
            Intent data) {
53          super.onActivityResult(requestCode, resultCode, data);
54          if (requestCode == CHOOSE_IMAGE && resultCode == RESULT_OK && data !=
                null && data.getData() != null) {
55              imgUrl = data.getData();
56              Picasso.get().load(imgUrl).into(imgPreview);
57          }
58      }
59
60      private String getFileExtension(Uri uri) {
61          ContentResolver contentResolver = getContentResolver();
62          MimeTypeMap mimeTypeMap = MimeTypeMap.getSingleton();
63          return mimeTypeMap.getExtensionFromMimeType(contentResolver.getType(uri)
                );
64      }
65
66      private void uploadImage() {
67          if (imgUrl != null) {
68              ToastUtils.showToastLong(NoticesActivity.this, "Uploading notice in
                    progress...");
69              final StorageReference fileReference = mStorageRef.child(System.
                    currentTimeMillis() + "." + getFileExtension(imgUrl));
70
71              mUploadTask = fileReference.putFile(imgUrl)
72                      .addOnSuccessListener(new OnSuccessListener<UploadTask.
                            TaskSnapshot>() {
73                          @Override
74                          public void onSuccess(UploadTask.TaskSnapshot
                                taskSnapshot) {
75                              Handler handler = new Handler();
76                              handler.postDelayed(new Runnable() {
```

```
                                @Override
                                public void run() {
                                    uploadProgress.setProgress(0);
                                }
                            }, 500);
                            fileReference.getDownloadUrl().addOnSuccessListener(
                                new OnSuccessListener<Uri>() {
                                @Override
                                public void onSuccess(Uri uri) {
                                    Upload upload = new Upload(imgDescription.
                                        getText().toString().trim(), uri.
                                        toString());
                                    String uploadID = mDatabaseRef.push().getKey
                                        ();
                                    mDatabaseRef.child(uploadID).setValue(upload
                                        );
                                    ToastUtils.showToastLong(NoticesActivity.
                                        this, "Notice uploaded successfully");
                                    imgPreview.setImageResource(R.drawable.
                                        ic_image_watermark);
                                    imgDescription.setText("");
                                }
                            });
                        }
                    })
                    .addOnFailureListener(new OnFailureListener() {
                        @Override
                        public void onFailure(@NonNull Exception e) {
                            ToastUtils.showToastLong(NoticesActivity.this, "
                                Error: " + e.getMessage());
                        }
                    })
                    .addOnProgressListener(new OnProgressListener<UploadTask.
                        TaskSnapshot>() {
                        @Override
                        public void onProgress(UploadTask.TaskSnapshot
                            taskSnapshot) {
                            double progress = (100.0 * taskSnapshot.
                                getBytesTransferred() / taskSnapshot.
                                getTotalByteCount());
                            uploadProgress.setProgress((int) progress);
                        }
                    });
        } else {
            ToastUtils.showToastShort(NoticesActivity.this, "Please choose image
                first!");
        }
    }

    public void On_View_All_Notices(View view) {
        startActivity(new Intent(NoticesActivity.this, ViewAllNoticesActivity.
            class));
    }

    public void On_Go_Back(View view) {
        super.getClass();
        this.finish();
    }

    @Override
    public void onBackPressed() {
```

```
        super.onBackPressed();
    }
}
```

ViewComplaintsActivity.java

Here Admin will be able to view the Notices.

```
package com.example.mysmartsociety;

public class ViewComplaintsActivity extends AppCompatActivity {

    private RecyclerView recyclerView;
    private FirebaseDatabase db = FirebaseDatabase.getInstance();
    private DatabaseReference root = db.getReference().child("member_complaints");
    private ComplaintsAdapter adapter;
    private ArrayList<ComplaintUpload> list;

    ProgressBar mProgressBar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_view_complaints);

        mProgressBar = findViewById(R.id.progress_circular);
        recyclerView = findViewById(R.id.recycler_view);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));

        list = new ArrayList<>();
        adapter = new ComplaintsAdapter(this ,list );

        recyclerView.setAdapter(adapter);

        if (CheckInternetConnectivity.isInternet(ViewComplaintsActivity.this)) {

            root.addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot snapshot) {

                    for (DataSnapshot dataSnapshot : snapshot.getChildren()){
                        ComplaintUpload model = dataSnapshot.getValue(
                            ComplaintUpload.class);
                        list.add(0, model);
                    }
                    mProgressBar.setVisibility(View.GONE);
                    adapter.notifyDataSetChanged();
                }

                @Override
                public void onCancelled(@NonNull DatabaseError error) {
                    mProgressBar.setVisibility(View.GONE);
                    ToastUtils.showToastShort(ViewComplaintsActivity.this,"Error
                        "+ error);
                }
            });
        } else {
            ToastUtils.showToastLong(ViewComplaintsActivity.this, "No Internet
```

```
                Connection!!!");
        }

    }

    //TODO remaining delete/ resolved complaint button for admin

    public void On_Go_Back(View view) {
        super.getClass();
        this.finish();
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
    }

}
```

6.2 Module 2 - User View (Society Member) :

In this module we have build different features for the User (Society Member). Where the User can do various tasks like Viewing Notices of the society, Adding Complaints, able to call Emergency Contacts in the society, Generate Gate Pass for the visitors and to see the Visitors List who are entering in the society.

HomeActivity.java

Here User Dashboard will be shown.

```
package com.example.mysmartsociety;

public class HomeActivity extends AppCompatActivity {

    private static final String TAG = HomeActivity.class.getName();

    private FirebaseAuth mAuth;
    FirebaseUser user;
    GoogleSignInClient mGoogleSignInClient;

    LinearLayout dialog_layout;
    TextView UserEmail, UserType;

    CategoryPreference cPrefrence;
    String UserTypeValue;
    Button Notice, Complaints;

    private FirebaseDatabase db = FirebaseDatabase.getInstance();
    private DatabaseReference visitorsRef;
```

```
20
21      @Override
22      protected void onCreate(Bundle savedInstanceState) {
23          super.onCreate(savedInstanceState);
24          setContentView(R.layout.activity_home);
25
26          mAuth = FirebaseAuth.getInstance();
27
28          FirebaseMessaging.getInstance().subscribeToTopic("all");
29
30          dialog_layout = (LinearLayout) findViewById(R.id.dialog_layout);
31          UserEmail = findViewById(R.id.user_email);
32          UserType = findViewById(R.id.user_type);
33
34          cPrefrence = CategoryPreference.getInstance(HomeActivity.this);
35
36          UserTypeValue = cPrefrence.getData("c_user_type");
37          UserType.setText(UserTypeValue);
38
39          //if logged in as Secretory
40          if(UserTypeValue.equals("Secretory")){
41              Notice = findViewById(R.id.btn_notices);
42              Complaints = findViewById(R.id.btn_complaints);
43
44              Notice.setText("View Notices");
45              Complaints.setText("View Complaints");
46          }
47      }
48
49      private void updateUI(FirebaseUser user) {
50          // hideProgressBar();
51          if (user != null) {
52              UserEmail.setText(user.getEmail());
53          } else {
54              UserEmail.setText(null);
55          }
56      }
57
58      @Override
59      public void onStart() {
60          super.onStart();
61          // Check if user is signed in (non-null) and update UI accordingly.
62          FirebaseUser currentUser = mAuth.getCurrentUser();
63          updateUI(currentUser);
64      }
65
66      //dialog open
67      public void On_Open_Dialog(View view) {
68          if (dialog_layout.getVisibility() == View.GONE) {
69              dialog_layout.setVisibility(View.VISIBLE);
70          }
71      }
72
73      //dialog close
74      public void On_Inner_Dilaog_Layout(View view) {
75          if (dialog_layout.getVisibility() == View.VISIBLE) {
76              dialog_layout.setVisibility(View.GONE);
77          }
78      }
79
80      public void On_Logout(View view) {
```

```
81          if (CheckInternetConnectivity.isInternet(HomeActivity.this)) {
82              if (mAuth.getCurrentUser() != null) {
83                  FirebaseAuth.getInstance().signOut();
84                  UserEmail.setText(null);
85                  cPrefrence.clearData();
86                  ToastUtils.showToastShort(HomeActivity.this, "You have
                        successfully Logout!");
87                  this.finish();
88                  startActivity(new Intent(HomeActivity.this,LoginActivity.class))
                        ;
89              }else if(user != null){
90                  mGoogleSignInClient.signOut().addOnCompleteListener(this,
91                          new OnCompleteListener<Void>() {
92                              @Override
93                              public void onComplete(@NonNull Task<Void> task) {
94                                  UserEmail.setText(null);
95                              }
96                          });
97                  this.finish();
98                  startActivity(new Intent(HomeActivity.this,LoginActivity.class))
                        ;
99                  ToastUtils.showToastShort(HomeActivity.this, "You have
                        successfully Logout!");
100             }
101             else {
102                 ToastUtils.showToastShort(HomeActivity.this, "Unable to Logout!"
                        );
103             }
104         } else {
105             ToastUtils.showToastLong(HomeActivity.this, "No Internet Connection
                    !!!");
106         }
107     }
108
109     public void On_Notification(View view) {
110         startActivity(new Intent(HomeActivity.this, NotificationActivity.class))
                ;
111     }
112
113     public void On_Visitor_List(View view) {
114         startActivity(new Intent(HomeActivity.this, VisitorListActivity.class));
115     }
116
117     //society details
118     public void On_Society_Details(View view) {
119         On_Inner_Dilaog_Layout(view);
120         startActivity(new Intent(HomeActivity.this, MySocietyDetailsActivity.
                class));
121     }
122
123     //notice btn
124     public void On_Notices(View view) {
125         if(UserTypeValue.equals("Secretory")){
126             startActivity(new Intent(HomeActivity.this, NoticesActivity.class ))
                    ;
127         }else{
128             startActivity(new Intent(HomeActivity.this, ViewAllNoticesActivity.
                    class ));
129         }
130
131     }
```

```
132
133     //complaints btn
134     public void On_Add_Complaints(View view) {
135         if(UserTypeValue.equals("Secretory")){
136             startActivity(new Intent(HomeActivity.this, AddComplaintsActivity.
                    class));
137         }else{
138             startActivity(new Intent(HomeActivity.this, ViewComplaintsActivity.
                    class));
139         }
140     }
141     //Pay btn
142     public void On_Pay(View view){
143         startActivity(new Intent(HomeActivity.this, PayActivity.class));
144     }
145
146     //emergency btn
147     public void On_Emergency_Contact(View view) {
148         startActivity(new Intent(HomeActivity.this, EmergencyActivity.class));
149     }
150
151     //Scan gate pass
152     public void On_Scan_Gatepass(View view) {
153         IntentIntegrator intentIntegrator = new IntentIntegrator(HomeActivity.
                this);
154         intentIntegrator.setDesiredBarcodeFormats(intentIntegrator.QR_CODE_TYPES
                );
155         intentIntegrator.setBeepEnabled(true);
156         intentIntegrator.setCameraId(0);
157         intentIntegrator.setPrompt("SCANNING GATE-PASS...");
158         intentIntegrator.setBarcodeImageEnabled(false);
159         intentIntegrator.initiateScan();
160     }
161
162     //after scanning the QR Code this method get the result and check
163     @Override
164     protected void onActivityResult(int requestCode, int resultCode, Intent data
            ) {
165         IntentResult Result = IntentIntegrator.parseActivityResult(requestCode,
                resultCode, data);
166         if (Result != null) {
167             if (Result.getContents() == null) {
168                 //if scanning process cancelled by the user
169                 ToastUtils.showToastShort(HomeActivity.this, "Process Cancelled!
                        ");
170             } else {
171                 //check society member using unique code if is valid or not
172                 try {
173                     Gson g = new Gson();
174                     JsonObject object = g.fromJson(Result.getContents(),
                            JsonObject.class);
175
176                     ToastUtils.showToastShort(HomeActivity.this, "You have
                            Successfully Verified");
177                     visitorsRef = db.getReference().child("visitors"+"/"+object.
                            get("Society_unique_code").getAsString());
178
179                     String id = visitorsRef.push().getKey();
180                     object.addProperty("id", id);
181                     object.addProperty("checkInTimestamp", System.
                            currentTimeMillis());
```

```
182
183                     HashMap<String, Object> mapObj = new Gson().fromJson(
184                             object, new TypeToken<HashMap<String, Object>>() {}.
                                getType()
185                     );
186                     visitorsRef.child(id).setValue(mapObj);
187
188                     Intent intent = new Intent(HomeActivity.this,
                            VisitorDetailActivity.class);
189                     intent.putExtra("data", mapObj);
190                     startActivity(intent);
191                 }
192                 catch (Exception ex) {
193                     ex.printStackTrace();
194                     ToastUtils.showToastShort(HomeActivity.this, "Sorry, unable
                            to verify!");
195                 }
196             }
197         } else {
198             super.onActivityResult(requestCode, resultCode, data);
199         }
200     }
201
202     @Override
203     public void onBackPressed() {
204         if (dialog_layout.getVisibility() == View.VISIBLE) {
205             dialog_layout.setVisibility(View.GONE);
206         } else {
207             finish();
208         }
209     }
210
211     public void On_generate_Gatepass(View view) {
212         startActivity(new Intent(HomeActivity.this, CreateVisitorActivity.class)
                );
213     }
214 }
```

NoticeActivity.java

Here User will be able to View the Notices.

```
1  package com.example.mysmartsociety;
2
3  public class NoticesActivity extends AppCompatActivity {
4
5      private static final int CHOOSE_IMAGE = 1;
6      //TextView viewGallery;
7      ImageView imgPreview;
8      EditText imgDescription;
9      ProgressBar uploadProgress;
10     private Uri imgUrl;
11
12     private StorageReference mStorageRef;
13     private DatabaseReference mDatabaseRef;
14
15     private StorageTask mUploadTask;
16
17     @Override
18     protected void onCreate(Bundle savedInstanceState) {
```

```
19          super.onCreate(savedInstanceState);
20          setContentView(R.layout.activity_notices);
21
22          uploadProgress = findViewById(R.id.uploadProgress);
23
24          imgDescription = findViewById(R.id.imgDescription);
25          imgPreview = findViewById(R.id.imgPreview);
26
27          mStorageRef = FirebaseStorage.getInstance().getReference("
                notices_uploads");
28          mDatabaseRef = FirebaseDatabase.getInstance().getReference("
                notices_uploads");
29
30      }
31
32      public void On_Choose_Image(View view) {
33          Intent intent = new Intent();
34          intent.setType("image/*");
35          intent.setAction(Intent.ACTION_GET_CONTENT);
36          startActivityForResult(intent, CHOOSE_IMAGE);
37      }
38
39      public void On_Upload_Image(View view) {
40          if (CheckInternetConnectivity.isInternet(NoticesActivity.this)) {
41              if(imgDescription.getText().toString().isEmpty()){
42                  imgDescription.setError("Please enter notice description!");
43              }else {
44                  uploadImage();
45              }
46          } else {
47              ToastUtils.showToastLong(NoticesActivity.this, "No Internet
                    Connection!!!");
48          }
49      }
50
51      @Override
52      protected void onActivityResult(int requestCode, int resultCode, @Nullable
            Intent data) {
53          super.onActivityResult(requestCode, resultCode, data);
54          if (requestCode == CHOOSE_IMAGE && resultCode == RESULT_OK && data !=
                null && data.getData() != null) {
55              imgUrl = data.getData();
56              Picasso.get().load(imgUrl).into(imgPreview);
57          }
58      }
59
60      private String getFileExtension(Uri uri) {
61          ContentResolver contentResolver = getContentResolver();
62          MimeTypeMap mimeTypeMap = MimeTypeMap.getSingleton();
63          return mimeTypeMap.getExtensionFromMimeType(contentResolver.getType(uri)
                );
64      }
65
66      private void uploadImage() {
67          if (imgUrl != null) {
68              ToastUtils.showToastLong(NoticesActivity.this, "Uploading notice in
                    progress...");
69              final StorageReference fileReference = mStorageRef.child(System.
                    currentTimeMillis() + "." + getFileExtension(imgUrl));
70
71              mUploadTask = fileReference.putFile(imgUrl)
```

```
72                      .addOnSuccessListener(new OnSuccessListener<UploadTask.
                            TaskSnapshot>() {
73                          @Override
74                          public void onSuccess(UploadTask.TaskSnapshot
                                taskSnapshot) {
75                              Handler handler = new Handler();
76                              handler.postDelayed(new Runnable() {
77                                  @Override
78                                  public void run() {
79                                      uploadProgress.setProgress(0);
80                                  }
81                              }, 500);
82                              fileReference.getDownloadUrl().addOnSuccessListener(
                                    new OnSuccessListener<Uri>() {
83                                  @Override
84                                  public void onSuccess(Uri uri) {
85                                      Upload upload = new Upload(imgDescription.
                                            getText().toString().trim(), uri.
                                            toString());
86                                      String uploadID = mDatabaseRef.push().getKey
                                            ();
87                                      mDatabaseRef.child(uploadID).setValue(upload
                                            );
88                                      ToastUtils.showToastLong(NoticesActivity.
                                            this, "Notice uploaded successfully");
89                                      imgPreview.setImageResource(R.drawable.
                                            ic_image_watermark);
90                                      imgDescription.setText("");
91                                  }
92                              });
93                          }
94                      })
95                      .addOnFailureListener(new OnFailureListener() {
96                          @Override
97                          public void onFailure(@NonNull Exception e) {
98                              ToastUtils.showToastLong(NoticesActivity.this, "
                                    Error: " + e.getMessage());
99                          }
100                     })
101                     .addOnProgressListener(new OnProgressListener<UploadTask.
                            TaskSnapshot>() {
102                         @Override
103                         public void onProgress(UploadTask.TaskSnapshot
                                taskSnapshot) {
104                             double progress = (100.0 * taskSnapshot.
                                    getBytesTransferred() / taskSnapshot.
                                    getTotalByteCount());
105                             uploadProgress.setProgress((int) progress);
106                         }
107                     });
108         } else {
109             ToastUtils.showToastShort(NoticesActivity.this, "Please choose image
                        first!");
110         }
111     }
112
113     public void On_View_All_Notices(View view) {
114         startActivity(new Intent(NoticesActivity.this, ViewAllNoticesActivity.
                class));
115     }
116
```

```
    public void On_Go_Back(View view) {
        super.getClass();
        this.finish();
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
    }
}
```

ViewComplaintsActivity.java

Here User will be able to View Complaints.

```
package com.example.mysmartsociety;

public class ViewComplaintsActivity extends AppCompatActivity {

    private RecyclerView recyclerView;
    private FirebaseDatabase db = FirebaseDatabase.getInstance();
    private DatabaseReference root = db.getReference().child("member_complaints"
        );
    private ComplaintsAdapter adapter;
    private ArrayList<ComplaintUpload> list;

    ProgressBar mProgressBar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_view_complaints);

        mProgressBar = findViewById(R.id.progress_circular);
        recyclerView = findViewById(R.id.recycler_view);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));

        list = new ArrayList<>();
        adapter = new ComplaintsAdapter(this ,list );

        recyclerView.setAdapter(adapter);

        if (CheckInternetConnectivity.isInternet(ViewComplaintsActivity.this)) {

            root.addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot snapshot) {

                    for (DataSnapshot dataSnapshot : snapshot.getChildren()){
                        ComplaintUpload model = dataSnapshot.getValue(
                            ComplaintUpload.class);
                        list.add(0, model);
                    }
                    mProgressBar.setVisibility(View.GONE);
                    adapter.notifyDataSetChanged();
                }

                @Override
                public void onCancelled(@NonNull DatabaseError error) {
```

```
                    mProgressBar.setVisibility(View.GONE);
                    ToastUtils.showToastShort(ViewComplaintsActivity.this,"Error
                        "+ error);
                }
            });
        } else {
            ToastUtils.showToastLong(ViewComplaintsActivity.this, "No Internet
                Connection!!!");
        }
    }

    //TODO remaining delete/ resolved complaint button for admin

    public void On_Go_Back(View view) {
        super.getClass();
        this.finish();
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
    }
}
```

6.3 Module 3 - Visitor View

In this module we have implemented Gate Pass. Where the visitor will be able to see the QR Code which is been generated and shared by the user to the visitor. The visitor has to show the QR Code to the watchman. Then the watchman will scan the QR Code and check its details whether its genuine or bogus. Once details are verified then the visitor will be allowed to enter the society.

CreateVisitorActivity.java

Here Admin or User can create Gate Pass by filling-up the required details.

```
package com.example.mysmartsociety;

public class CreateVisitorActivity extends AppCompatActivity {
    EditText visitorName, roomno, visitorReason, visitorNo;
    ImageView imageView;

    CategoryPreference cPrefrence;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_visitor);
```

```
13
14          cPrefrence = CategoryPreference.getInstance(CreateVisitorActivity.this);
15
16          visitorName=findViewById(R.id.edt_visitor_name);
17          roomno=findViewById(R.id.edt_roomno);
18          visitorReason=findViewById(R.id.edt_visitor_reason);
19          visitorNo=findViewById(R.id.edt_visitor_no);
20          imageView = findViewById(R.id.imageView);
21      }
22
23      public void onVisitor(View view) {
24          if(!CheckInternetConnectivity.isInternet(CreateVisitorActivity.this)) {
25              Toast.makeText(this, "Not Internet", Toast.LENGTH_SHORT).show();
26              return;
27          }
28          if (visitorName.getText().toString().isEmpty()) {
29              visitorName.setError("Please enter Visitor name");
30              return;
31          }
32          if (roomno.getText().toString().isEmpty()) {
33              roomno.setError("Please Enter Room no");
34              return;
35          }
36          if (visitorReason.getText().toString().isEmpty()) {
37              visitorReason.setError("Please Enter Reason");
38              return;
39          }
40          if (visitorNo.getText().toString().isEmpty()) {
41              visitorNo.setError("Please Enter Phone Number");
42              return;
43          }
44          JsonObject object = new JsonObject();
45          object.addProperty("visitorName", visitorName.getText().toString());
46          object.addProperty("roomno", roomno.getText().toString());
47          object.addProperty("visitorReason", visitorReason.getText().toString());
48          object.addProperty("visitorNo", visitorNo.getText().toString());
49          object.addProperty("Society_unique_code", cPrefrence.getData("
                c_unique_code"));
50          object.addProperty("email", cPrefrence.getData("c_email"));
51
52          String text = object.toString();
53
54          MultiFormatWriter multiFormatWriter = new MultiFormatWriter();
55          try {
56              BitMatrix bitMatrix = multiFormatWriter.encode(text, BarcodeFormat.
                    QR_CODE,200,200);
57              BarcodeEncoder barcodeEncoder = new BarcodeEncoder();
58              Bitmap bitmap = barcodeEncoder.createBitmap(bitMatrix);
59              imageView.setImageBitmap(bitmap);
60          } catch (WriterException e) {
61              e.printStackTrace();
62          }
63      }
64  }
```

VisitorDetailActivity.java

Here Details of the visitor will be stored.

```
1  package com.example.mysmartsociety;
```

```
public class VisitorDetailActivity extends AppCompatActivity {
    private static final String TAG = VisitorDetailActivity.class.getName();


    TextView detailName, detailRoomno, detailReason, detailNo;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_visitor_detail);

        detailName=findViewById(R.id.edt_visitor_detail_name);
        detailRoomno=findViewById(R.id.edt_detail_roomno);
        detailReason=findViewById(R.id.edt_visitor_detail_reason);
        detailNo=findViewById(R.id.edt_visitor_detail_no);

            HashMap<String, String> societyMap = (HashMap<String, String>)
                getIntent().getSerializableExtra("data");
            Log.d("HashMapTest", societyMap.toString());
        if (societyMap.containsKey("visitorReason")) {
            detailReason.setText(societyMap.get("visitorReason").toString());
        }
        if (societyMap.containsKey("visitorName")) {
            detailName.setText(societyMap.get("visitorName").toString());
        }
        if (societyMap.containsKey("roomno")) {
            detailRoomno.setText(societyMap.get("roomno").toString());
        }
        if (societyMap.containsKey("visitorNo")) {
            detailNo.setText(societyMap.get("visitorNo").toString());
        }
    }
    public void On_Go_Back(View view) {
        this.finish();
    }
}
```

VisitorListActivity.java

Here List of the visitors in the society will be shown.

```
package com.example.mysmartsociety;

public class VisitorListActivity extends AppCompatActivity {

    private final static String TAG = VisitorListActivity.class.getName();

    CategoryPreference cPrefrence;

    private FirebaseDatabase db = FirebaseDatabase.getInstance();
    private DatabaseReference visitorsRef;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_visitor_list);

        cPrefrence = CategoryPreference.getInstance(VisitorListActivity.this);

```

```
        ArrayList<Map<String, Object>> visitorList = new ArrayList<>();

        RecyclerView recyclerView = findViewById(R.id.myListView);
        recyclerView.setLayoutManager(new LinearLayoutManager(
            getApplicationContext()));
        recyclerView.setItemAnimator(new DefaultItemAnimator());

        VisitorListAdapter adapter = new VisitorListAdapter(
            getApplicationContext(), visitorList);
        recyclerView.setAdapter(adapter);

        visitorsRef = db.getReference().child("visitors"+"/"+cPrefrence.getData(
            "c_unique_code"));
        visitorsRef.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                for (DataSnapshot postSnapshot: dataSnapshot.getChildren()) {
                    ObjectMapper oMapper = new ObjectMapper();
                    Map<String, Object> map = oMapper.convertValue(postSnapshot.
                        getValue(), Map.class);
                    visitorList.add(map);
                }
                Log.d(TAG, "onDataChange: "+visitorList.size());
                adapter.notifyDataSetChanged();
            }

            @Override
            public void onCancelled(DatabaseError databaseError) {
                // Getting Post failed, log a message
                Log.w("TAG", "loadPost:onCancelled", databaseError.toException()
                    );
                // ...
            }
        });
    }

    public void On_Go_Back(View view) {
        this.finish();
    }
}
```

Chapter 7

System Testing

System testing is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements.

7.1 Test Cases and Test Results

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	User Registration	All Valid Input	User Registered Successfully/ Unsuccessfully	Success/ Failed
T02	User Login	Username and Password Required	User Login Successfully/ Unsuccessfully	Success/ Failed
T03	Create Building	Data fetch and Create Building	Generate Unique Code	Successfully Created Building
T04	Join Building	Data fetch and Join building	Enter Unique Code	Successfully Join the Building
T05	Add Notice	Input Notice Details	Show Notice To User	Successfully show Notices
T06	Add Complaint	Input Complaint Details	Show Complaint to Admin	Successfully Show Notices

T07	Make Payment	Data Fetch and Make Payment	Successfully show Payment option	Successful Make Payment
T08	Gate Pass	All Valid Input	User Generate Gate Pass	Successfully Generated Gate Pass
T09	Scan Gate Pass	Turn On Scanner	Scan QR Code	Successfully Scan QR Code

7.2 Test Cases

Title: User registration – Successfully register a new user
Description: A new user should be able to successfully register themselves.

Precondition: The user has given valid credentials.
*Assumption:*A supported Android version is being used.

Test Steps:

1. Click 'Sign Up' button.
2. Enter valid credentials in the field.
3. Click 'Register' button

Expected Result: User should be successfully registered on the App.
Actual Result: User is successfully registered

Title: User login – Successful login in App.
Description:A registered user should be able to successfully login in App.

Precondition: The User is pre-registered
*Assumption:*A supported Android version is being used.

Test Steps:

1. Click 'SignIn' option.
2. Enter gmail id and password
3. Click 'Login' button

Expected Result: User should be successfully logged in and redirected to home page. **Actual Result:** User is redirected to home page.

Title: Add Notice– Successfully Add the Notice.
Description:A registered Admin should be able to Add and view Notice.

Precondition: The User is pre-registered
*Assumption:*A supported Android version is being used.

Test Steps:

1. Click on Add Notice.
2. Type the notice Details.
3. Add Image if necessary.

Expected Result: The Notice should be displayed to user **Actual Result:** Uploaded Notice is displayed successfully.

Title: Create Building – Successfully Created Building.
Description:A registered Admin should be able to click on Create New Building

Precondition: The user must be logged in with their registered details.

*Assumption:*A supported Android version is being used.

Test Steps:

1. Click on Create Building.
2. Enter Building Details.
3. Generate Unique Code.

Expected Result: A Building Should be successfully created . **Actual Result:** The Building is Created successfuly.

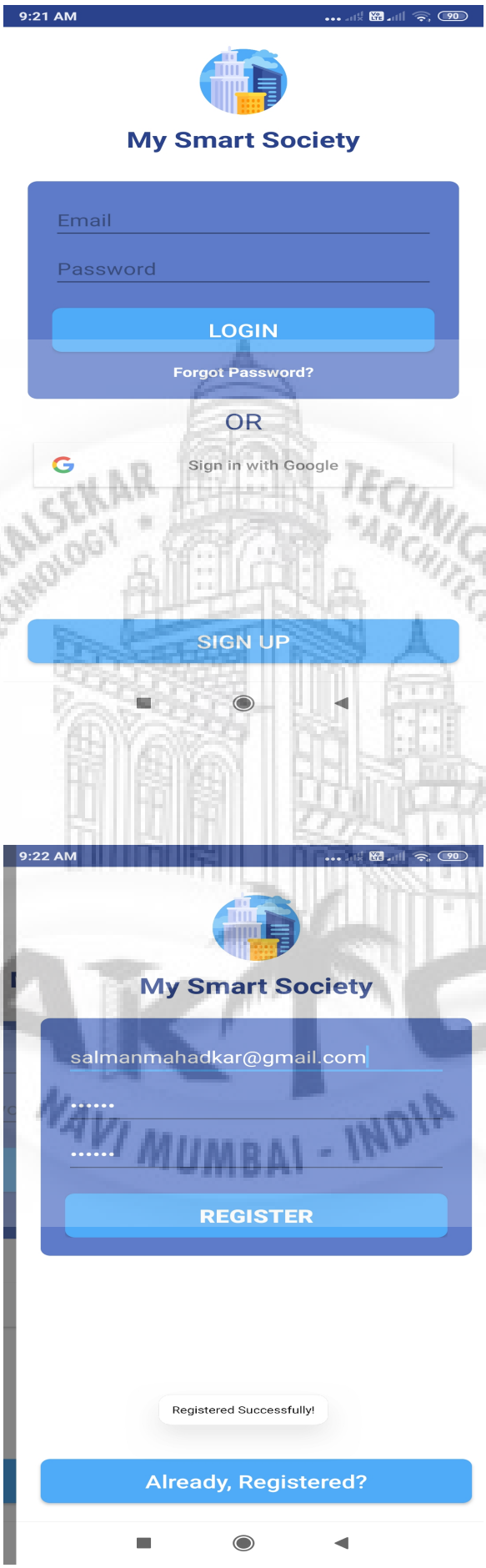
7.2.1 Software Quality Attributes

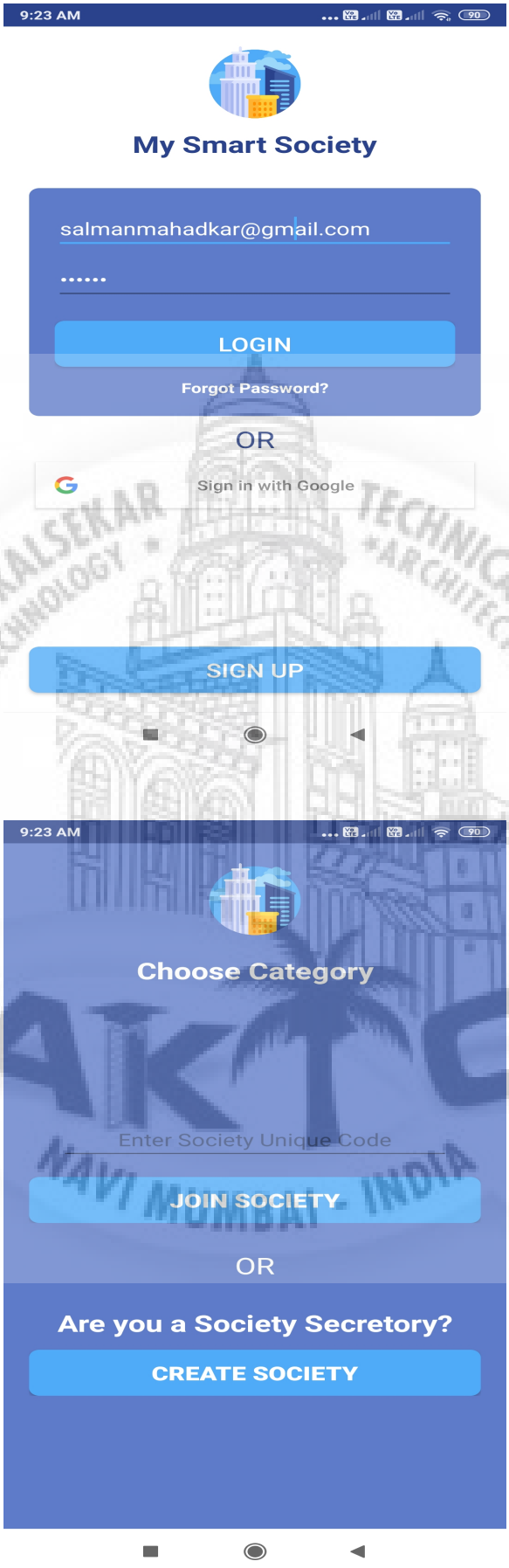
- AVAILABILITY: The system should not be down, whenever the user use the system the specific data should be available to the user.
- CORRECTNESS: As per the user search the the correct should be shown to the user like at time for searching the the similar type of app the system should show all the similar startup.
- MAINTAINABILITY: The administration of the system will maintain the system with effective updates though on air update if needed

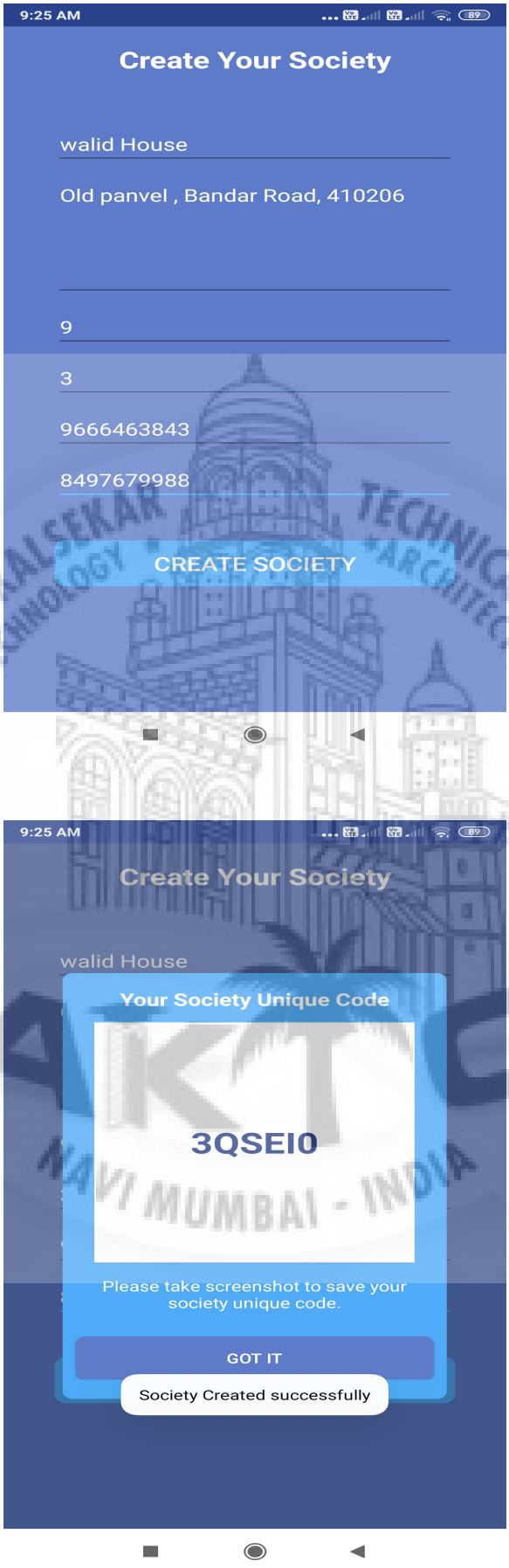
Chapter 8

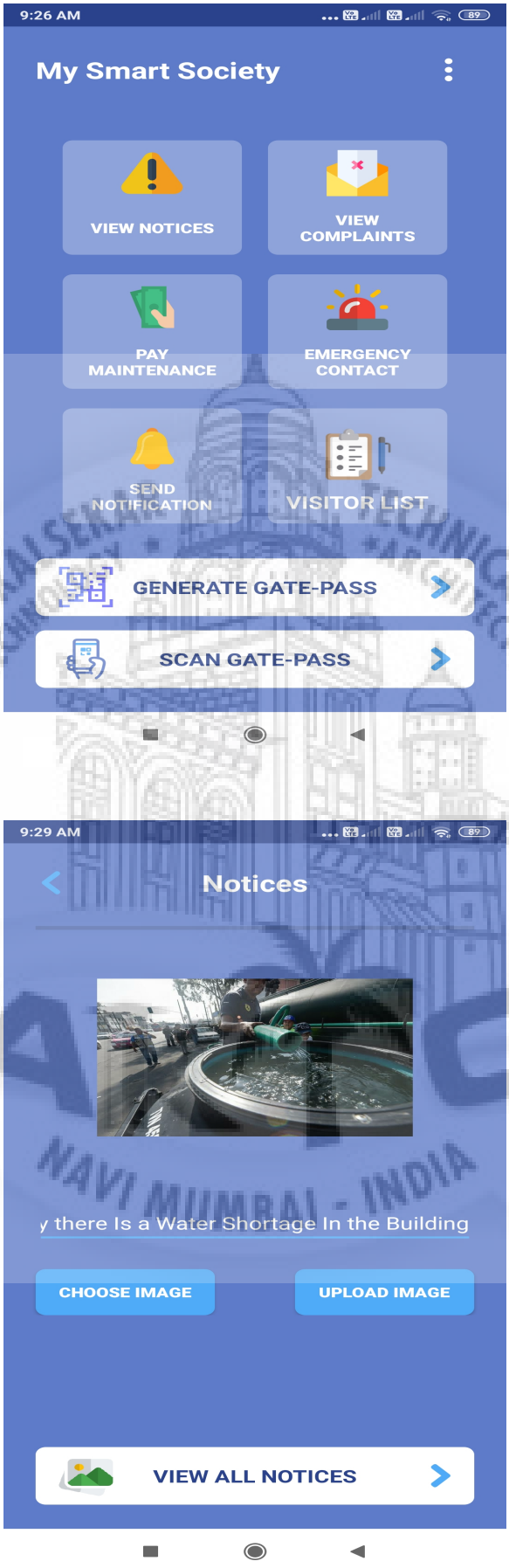
Screenshots of Project

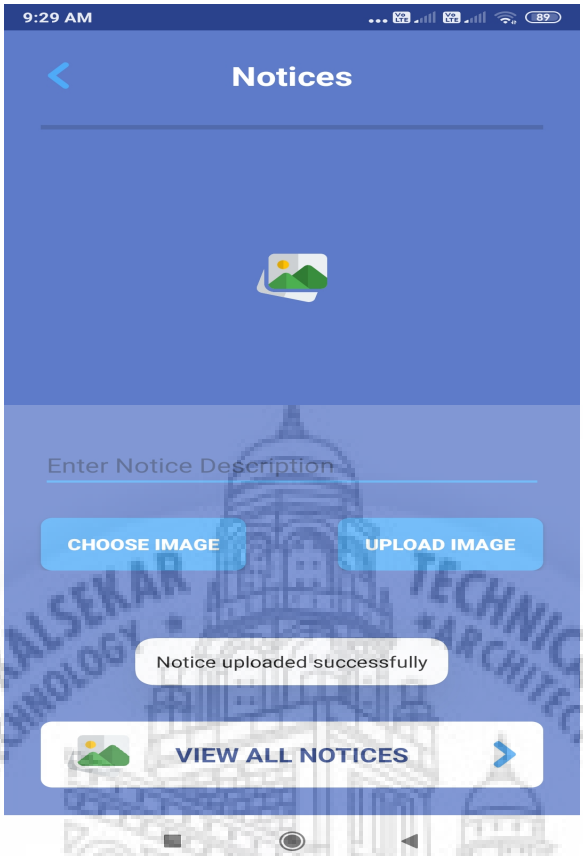
8.1 Admin View


9:21 AM
My Smart Society
Email
Password
LOGIN
Forgot Password?
OR
Sign in with Google
SIGN UP
9:22 AM
My Smart Society
salmanmahadkar@gmail.com
REGISTER
Registered Successfully!
Already, Registered?

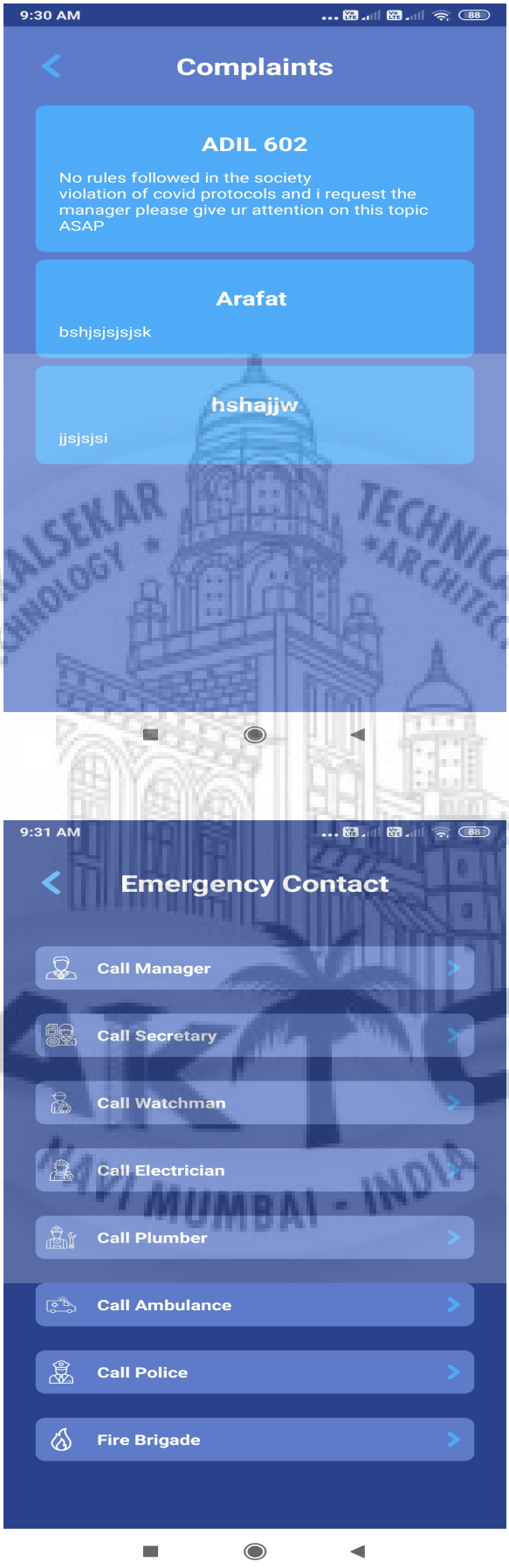

9:23 AM
My Smart Society
salmanmahadkar@gmail.com
.....
LOGIN
Forgot Password?
OR
Sign in with Google
SIGN UP
9:23 AM
Choose Category
Enter Society Unique Code
JOIN SOCIETY
OR
Are you a Society Secretory?
CREATE SOCIETY

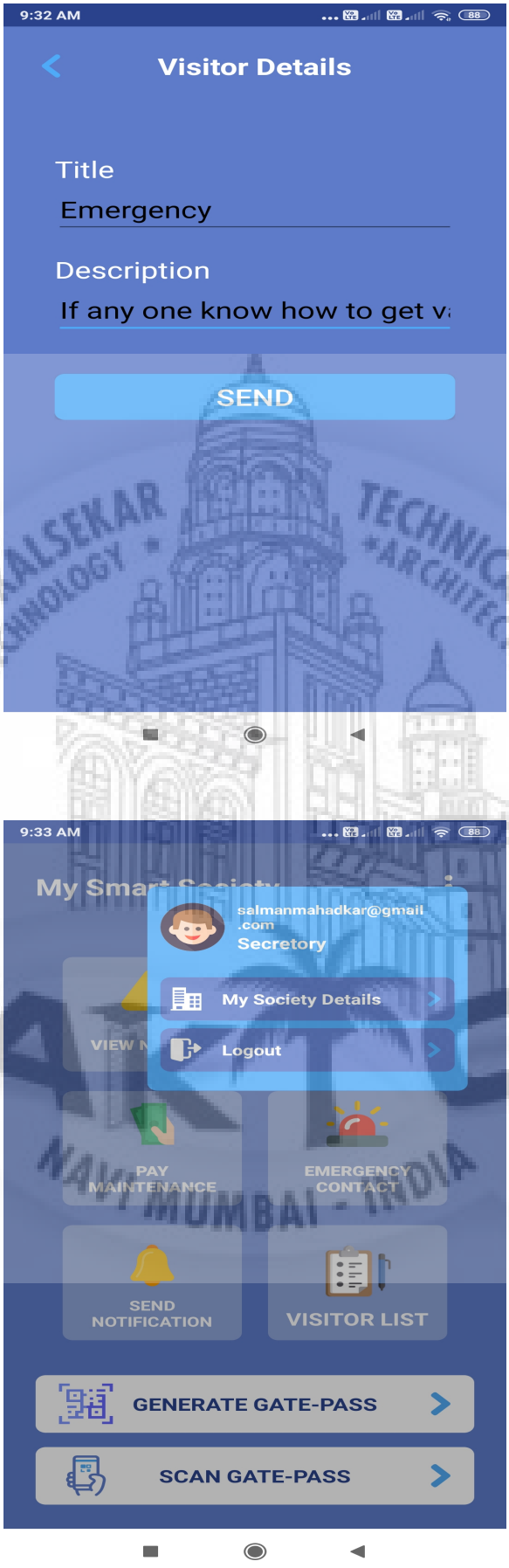

9:25 AM
Create Your Society
walid House
Old panvel , Bandar Road, 410206
9
3
9666463843
8497679988
CREATE SOCIETY
9:25 AM
Create Your Society
walid House
Your Society Unique Code
3QSEI0
Please take screenshot to save your society unique code.
GOT IT
Society Created successfully


9:26 AM
My Smart Society
VIEW NOTICES
VIEW COMPLAINTS
PAY MAINTENANCE
EMERGENCY CONTACT
SEND NOTIFICATION
VISITOR LIST
GENERATE GATE-PASS
SCAN GATE-PASS
9:29 AM
Notices
y there Is a Water Shortage In the Building
CHOOSE IMAGE
UPLOAD IMAGE
VIEW ALL NOTICES

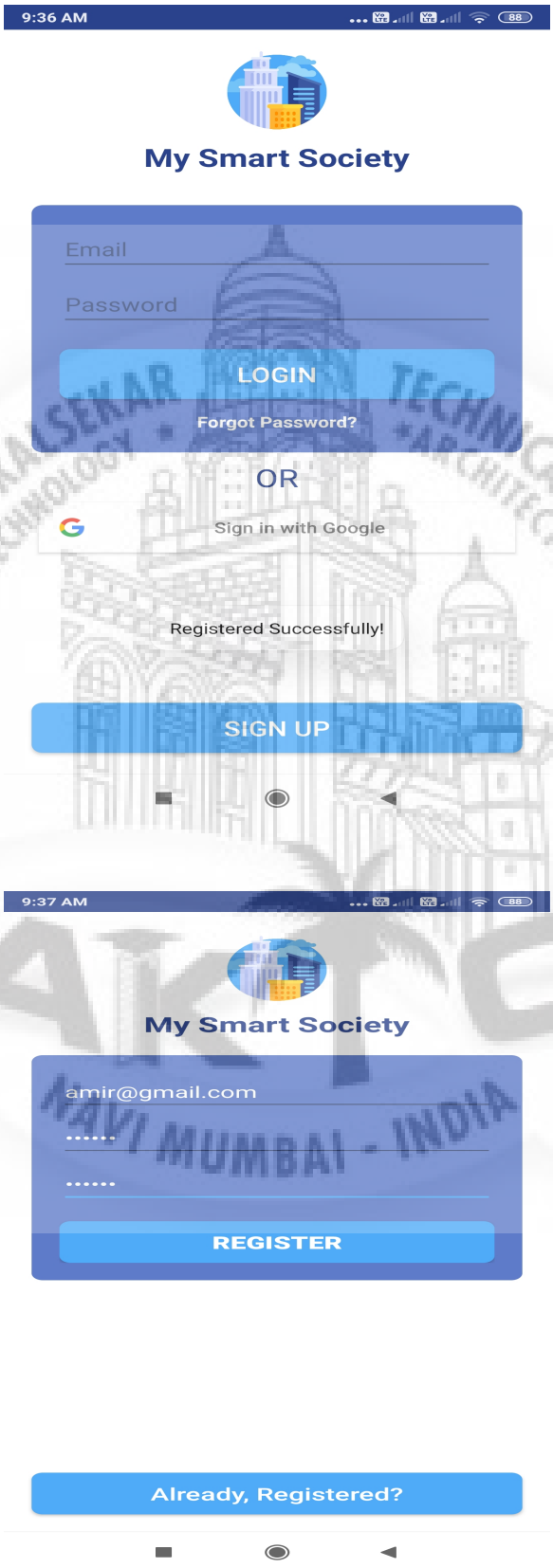

9:29 AM
Notices
Enter Notice Description
CHOOSE IMAGE
UPLOAD IMAGE
Notice uploaded successfully
VIEW ALL NOTICES

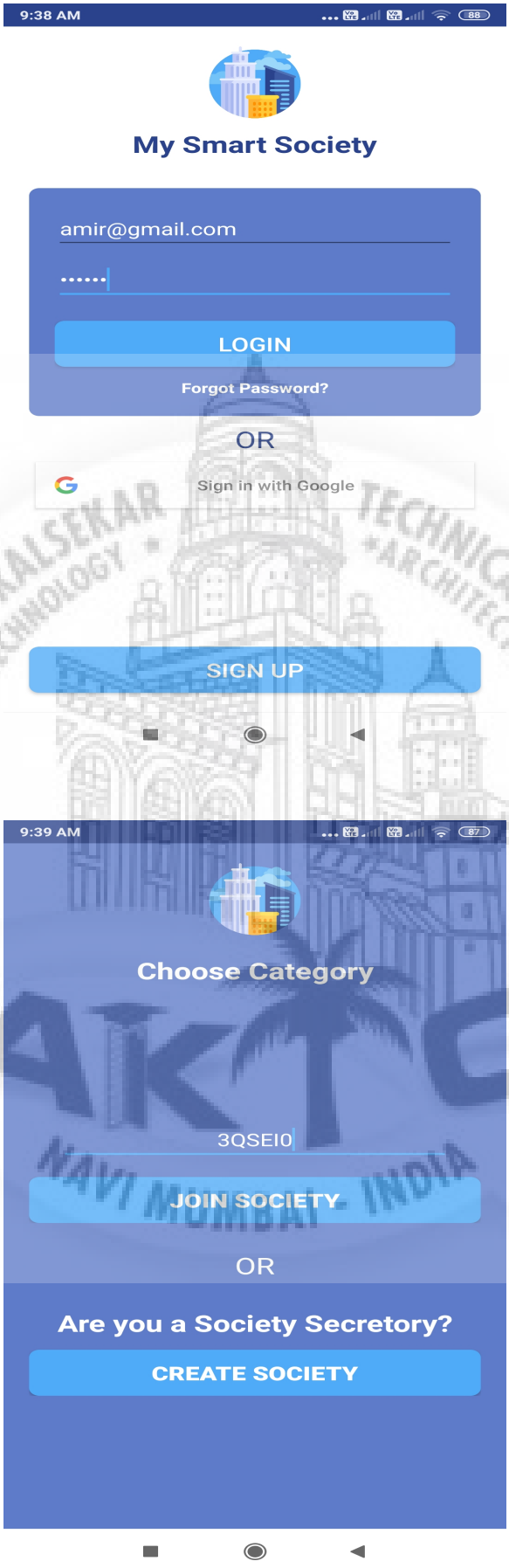

9:29 AM
All Notices
Notice: Today there Is a Water Shortage In the Building
Notice: Placement
PrepInsta
Wipro WILP
May, 2021
Eligibility:
BCA/ B.Sc (2021)
Service Agreement:
On joining, candidates would be required to sign a service agreement for 60 months
Apply Now
Notice: Test

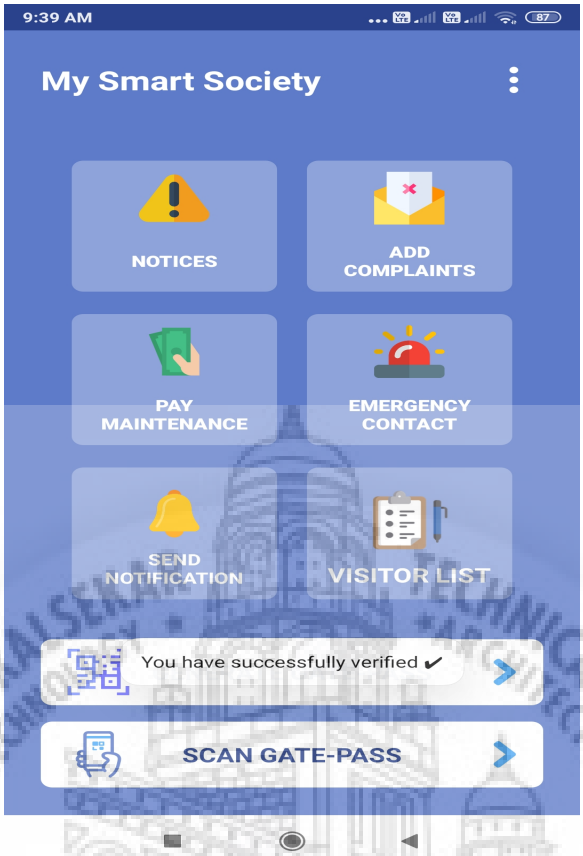

9:30 AM
Complaints
ADIL 602
No rules followed in the society violation of covid protocols and i request the manager please give ur attention on this topic ASAP
Arafat
bshjsjsjsjsk
hshajjw
jjsjsjsi
9:31 AM
Emergency Contact
Call Manager
Call Secretary
Call Watchman
Call Electrician
Call Plumber
Call Ambulance
Call Police
Fire Brigade


9:32 AM
Visitor Details
Title
Emergency
Description
If any one know how to get v
SEND
9:33 AM
My Smart Society
salmanmahadkar@gmail
.com
Secretory
My Society Details
Logout
VIEW N
PAY
MAINTENANCE
EMERGENCY
CONTACT
SEND
NOTIFICATION
VISITOR LIST
GENERATE GATE-PASS
SCAN GATE-PASS

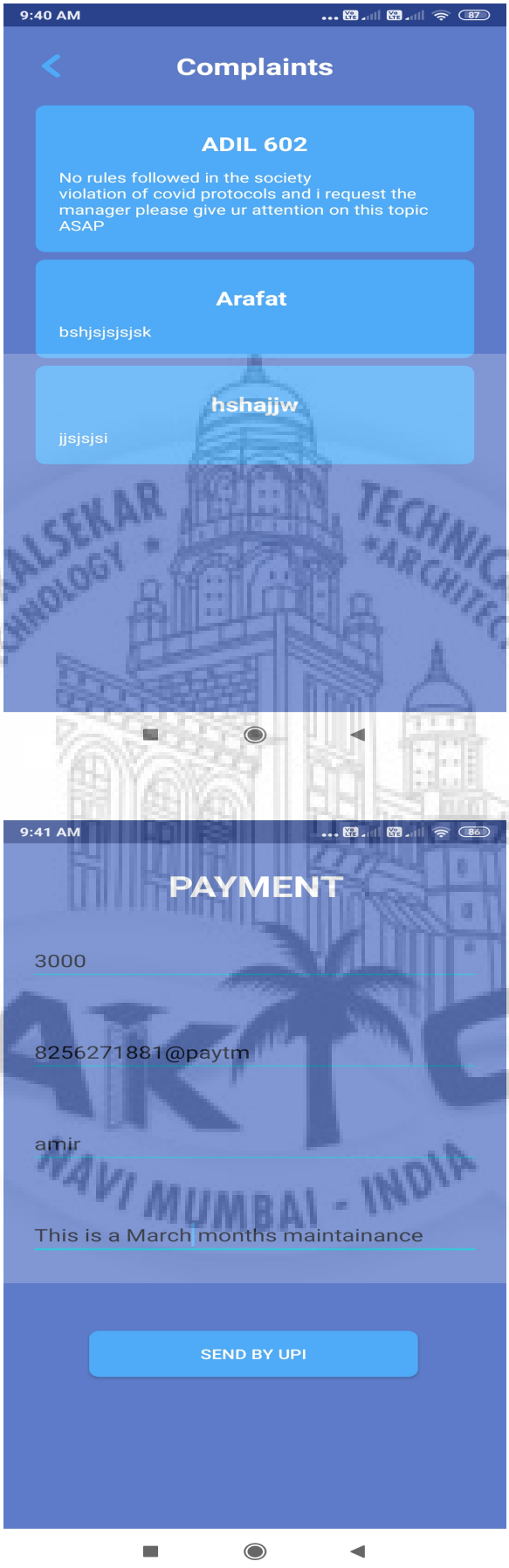
8.2 User View

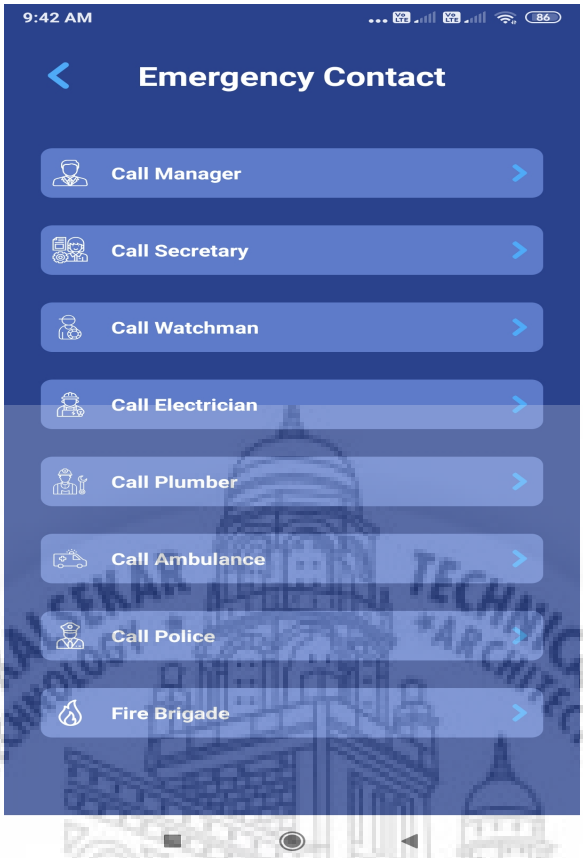

9:36 AM
My Smart Society
Email
Password
LOGIN
Forgot Password?
OR
Sign in with Google
Registered Successfully!
SIGN UP
9:37 AM
My Smart Society
amir@gmail.com
REGISTER
Already, Registered?


9:38 AM
My Smart Society
amir@gmail.com
.....
LOGIN
Forgot Password?
OR
Sign in with Google
SIGN UP
9:39 AM
Choose Category
3QSEI0
JOIN SOCIETY
OR
Are you a Society Secretory?
CREATE SOCIETY

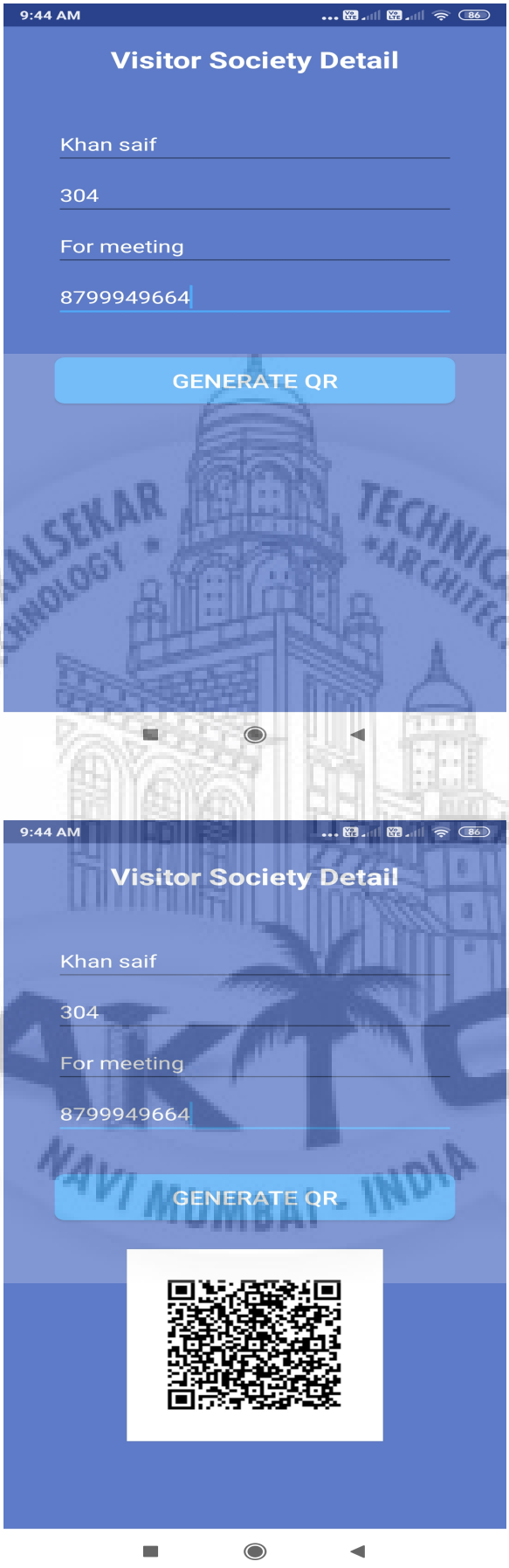

9:39 AM
My Smart Society
NOTICES
ADD COMPLAINTS
PAY MAINTENANCE
EMERGENCY CONTACT
SEND NOTIFICATION
VISITOR LIST
You have successfully verified ✔
SCAN GATE-PASS


9:39 AM
All Notices
Notice: Today there Is a Water Shortage In the Building
Notice: Placement
Wipro WILP
May, 2021
Eligibility:
BCA/ B.Sc (2021)
Service Agreement:
Apply Now
Please wait loading all notice...
Notice: Test


9:40 AM
Complaints
ADIL 602
No rules followed in the society violation of covid protocols and i request the manager please give ur attention on this topic ASAP
Arafat
bshjsjsjsjsk
hshajjw
jjsjsjsi
9:41 AM
PAYMENT
3000
8256271881@paytm
amir
This is a March months maintainance
SEND BY UPI

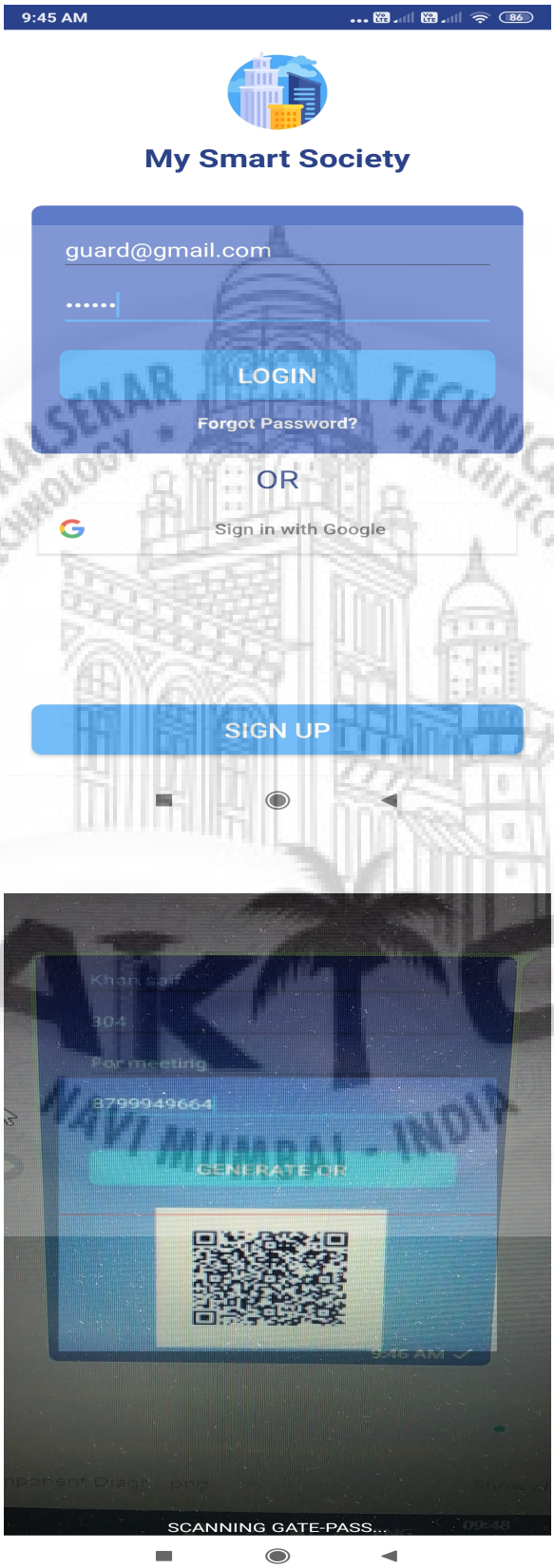

9:42 AM
Emergency Contact
Call Manager
Call Secretary
Call Watchman
Call Electrician
Call Plumber
Call Ambulance
Call Police
Fire Brigade

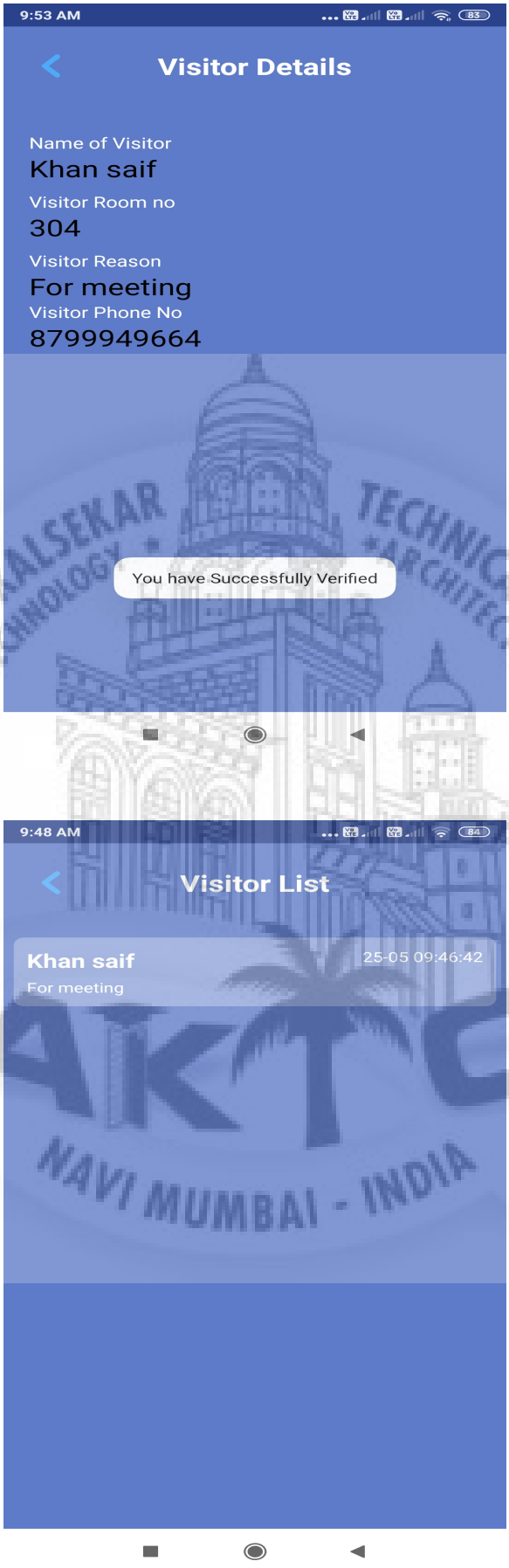

9:43 AM
Visitor Society Detail
Name of the Visitor
Room No
Reason
Visitor Phone Number
GENERATE QR


9:44 AM
Visitor Society Detail
Khan saif
304
For meeting
8799949664
GENERATE QR
9:44 AM
Visitor Society Detail
Khan saif
304
For meeting
8799949664
GENERATE QR


9:44 AM
My Smart Society
amir@gmail.com
Society Member
My Society Details
Logout
NOT
PAY MAINTENANCE
EMERGENCY CONTACT
SEND NOTIFICATION
VISITOR LIST
GENERATE GATE-PASS
SCAN GATE-PASS

8.3 Visitor View




9:53 AM
Visitor Details
Name of Visitor
Khan saif
Visitor Room no
304
Visitor Reason
For meeting
Visitor Phone No
8799949664
You have Successfully Verified
9:48 AM
Visitor List
Khan saif
25-05 09:46:42
For meeting

Chapter 9

Conclusion and Future Scope

9.1 Conclusion

Smart Society App provides various features that will overcome the drawbacks of traditional method of housing society maintenance. The main focus of the project is to reduce human efforts. The app will help in managing housing society in a systematic, organized and well-documented way.

Here, both managing committee and residents have the same application installed with the ability to do different tasks like payment of maintenance bill, viewing notices, etc. Both type of users have different rights like committee member can add contacts of essential personals, posting notices on the notice board whereas users can only view. Our smart society application is implemented to help manage the affairs of a housing society. The concept of data mining and artificial intelligence would be worked upon as a future work for our project.

9.2 Future Scope

- Use of Artifical Intelligence and Machine Learning.
- Auto-payment of Maintenance bill.
- Society members can chat, in application itself.

References

Websites:
https://developer.android.com/guide/index.html

https://stackoverflow.com/

https://parseplatform.org/

https://www.vogella.com/tutorials/android.html

[1] Shantanu Kudale, Chandan Amarnani, Harshal Sawakare, Shubhankar Kok-ate, Sujata Kadu. "Housing Society Management", INTERNATIONAL JOURNAL OF SCIENTIFIC TECHNOLOGY RESEARCH VOLUME 7, ISSUE 5, MAY 2018.

[2] Rahul Bhagwat, Aashay Bharadwaj, Vivek Harsode, Anurag Chawake, Mrs. Deepali Bhanage. "Society Management Application on Android", International Research Journal of Engineering and Technology Volume: 05 Issue: 05, May-2018.

[3] Rutuja Vatharkar, Pratiksha Patil, Swati Sonar, Prof. Shivganga Gavhane. "IMPLEMENTATION OF SOCIETY MANAGEMENT SYSTEM: SOCIET-ALES", International Research Journal of Science Technology Volume: 06 Issue: 02, April-2016.

[4] Saurabh Malgaonkar, Vivek Maurya, Mukul Kulkarni, Gurtej Singh Majithia, "Multipurpose Android Based Mobile Notifier", ICAECC,2014,p 1-4.

[5] SHAO Guo-hong, “Application Development Research Based on Android Platform”,ICICTA,2014,pp-579-582.