# HOME RENTAL APP USING ANDROID APPLICATION DEVELOPMENT

*A min-project report submitted in partial fulfillment of the Academic requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING**
**IN**
**INFORMATION TECHNOLOGY**

**By**

| | |
|---|---|
| **MARRI RAHUL** | **(2451-20-737-313)** |
| **PONAM SHIVANI** | **(2451-20-737-315)** |
| **B. CHANDRA SHEKAR** | **(2451-20-737-317)** |

***Under the guidance of***
**Ms. P. AMBABHAVANI**
**Assistant Professor ITD**

**DEPARTMENT OF INFORMATION TECHNOLOGY**
**MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE**
**(An Autonomous Institution)**
**(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)**
**Nadergul, Saroornagar Mandal, Hyderabad-501510**

**2022-2023**

**(An Autonomous Institution)**
**(Affiliated to Osmania University, Hyderabad. Recognized by AICTE)**
**Nadergul, Saroornagar Mandal, Hyderabad-501510**

## DEPARTMENT OF INFORMATION TECHNOLOGY

## CERTIFICATE

This is to certify that the mini project work entitled "**HOME RENTAL APP USING ANDROID APPLICATION DEVELOPMENT**" is a bonafide work carried out by **Mr. Marri Rahul (2451-20-737-313), Mr. Ponnam Shivani (2451-20-737-316), Mr. BEJUGAM CHANDRA SHEKAR (2451-20-737-317)** in partial fulfillment of the requirements for the award of degree of **Bachelor of Engineering** in **Information Technology** from **Maturi Venkata Subba Rao (M.V.S.R.) Engineering College,** affiliated to OSMANIA UNIVERSITY, Hyderabad, during the Academic Year 2022-23. Under our guidance and supervision.

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Signature of Project Coordinator Signature of Guide

Signature of Head, ITD Signature of External Examiner

# DECLARATION

This is to certify that the work reported in the present mini-project entitled "**HOME RENTAL APP USING ANDROID APPLICATION DEVELOPMENT**" is a record of bonafide work done by us in the Department of Information Technology, M.V.S.R. Engineering College, and Osmania University. This report is based on the project work done entirely by us and not copied from any other source.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

| **Roll Number** | **Student Name** | **Signature** |
|---|---|---|
| 2451-20-737-313 | M. RAHUL | |
| 2451-20-737-316 | P. SHIVANI | |
| 2451-20-737-317 | B. CHANDRA SHEKAR | |

# ACKNOWLEDGEMENT

We with extreme jubilance and deepest gratitude, would like to thank our guide **Ms. P. AMBABHAVANI Assistant Professor ITD**, Department of Information Technology, Maturi Venkata Subba Rao (MVSR) Engineering College, for her constant encouragement to us to complete our work in time.

With immense pleasure, we record our deep sense of gratitude to our beloved Head of the department **Dr.K.VenuGopal Rao** Dean-Academics & HOD, Department of Information Technology, Maturi Venkata Subba Rao Engineering College, for permitting and providing facilities to carry out this project.

We would like to extend our gratitude to **K.Devaki** and **D.Muninder** Mini-Project-I coordinators, Department of Information Technology, Maturi Venkata Subba Rao Engineering College, for their valuable suggestions and timely help during the course of the project.

Finally, we express, from the bottom of our heart and deepest gratitude to the entire faculty, my parents and family for the support, dedication, comprehension and love.

**MARRI RAHUL(2451-20-737-313)**

**PONAM SHIVANI(2451-20-737-316)**

**BEJUGAM CHANDRA SHEKAR(2451-20-737-317)**

## MVSR Engineering College

## Department of Information Technology

**COURSE NAME: MINI PROJECT I**

**COURSE CODE: PW 654 IT**

### VISION

To impart technical education to produce competent and socially responsible engineers in the field of Information Technology.

### MISSION

M1. To make the teaching-learning process effective and stimulating.

M2. To provide adequate fundamental knowledge of sciences and Information Technology with positive attitude.

M3. To create an environment that enhances skills and technologies required for industry.

M4. To encourage creativity and innovation for solving real world problems.

M5. To cultivate professional ethics in students and inculcate a sense of responsibility towards society

### PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

The Bachelor’s program in Information Technology is aimed at preparing graduates who will:

I. Apply knowledge of mathematics and Information Technology to analyze, design and implement solutions for real world problems in core or in multidisciplinary areas.

II. Communicate effectively, work in a team, practice professional ethics and apply knowledge of computing technologies for societal development.

III. Engage in Professional development or postgraduate education to be a life-long learner.

## PROGRAM OUTCOMES (POs)

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. Problem analysis: Identity, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.
3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## PROGRAM SPECIFIC OUTCOMES (PSOS):

(1) Hardware design: An ability to analyze, design, simulate and implement computer hardware/software and use basic analog/digital circuits, VLSI design for various computing and communication system applications.

(2) Software design: An ability to analyze a problem, design algorithm, identify and define the computing requirements appropriate to its solution and implement the same.

## COURSE OBJECTIVES:

1. To enhance practical & Professional skills.

2.To familiarize the tools and techniques of symmetric literature survey and documentation.

3. To expose students to industry practices and teamwork.

4. To encourage students to work with innovative and entrepreneurial ideas.

## COURSE OUTCOMES:

On successful completion of this course students will be able to:

1. Define a problem of the recent advancements with applications towards society.

2. Outline requirements and perform requirement analysis for solving the problem.

3. Design and develop a software and/or hardware-based solution within the scope of project using contemporary technologies and tools.

4. Test and deploy the applications for use.

5. Develop the Project as a team and demonstrate the application, with effective written and oral communications.

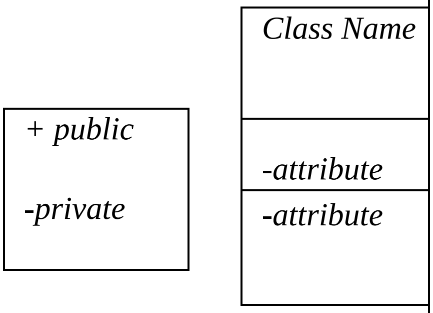
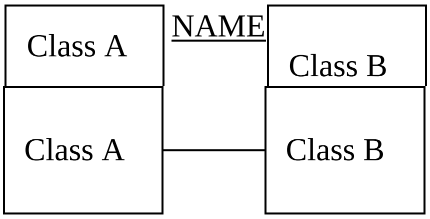
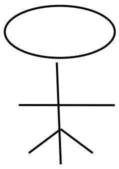
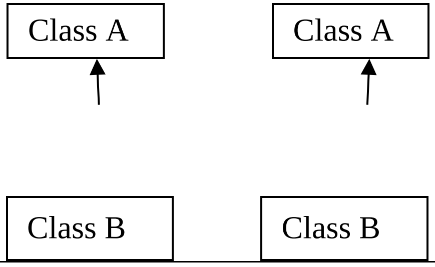
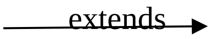
# ABSTRACT

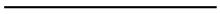
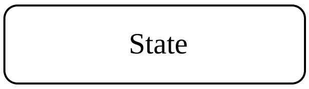
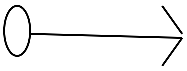
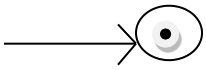
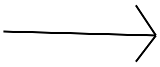
This project aims to develop a mobile application for house rental services using Android development. The app will serve as a platform for property owners to list their rental properties, and for potential renters to search for available house for rent, view photos and details, and book rentals. The app will include features such as user authentication, property listing and management, rental booking between property owners and renters. The development process will involve designing and implementing a user-friendly interface, and testing the app for functionality, usability, and performance. The ultimate goal of the project is to create a reliable and efficient solution for individuals seeking rental properties and property owners looking to list their properties in a seamless and streamlined manner.

## LIST OF FIGURES

# LIST OF TABLES

# LIST OF SYMBOLS

| S.NO | NOTATION NAME | NOTATION | DESCRIPTION |
|---|---|---|---|
| 1. | Class |   + public<br>-private<br>Class Name<br>-attribute<br>-attribute | Represents a collection of similar entities grouped together. |
| 2. | Association |  | Association represents static relationships between classes. Role represents the way the two classes see each other. |
| 3. | Actor |  | It aggregates several classes into a single class. |
| 4. | Aggregation |  | Interaction between the system and external environment |
| 6. | Relation (extends) |  | Extends relationship is used when one use case is similar to another use case but does a bit more. |

| | | | |
|---|---|---|---|
| 7. | Communication |  | Communication between various use cases. |
| 8. | State |  | State of the processes. |
| 9. | Initial State |  | Initial state of the object |
| 10. | Final state |  | Final state of the object |
| 11. | Control flow |  | Represents various control flow between the states |

## TABLE OFCONTENTS

# CHAPTER 1

## INTRODUCTION

In the fast-paced digital era, where convenience and flexibility are paramount, a revolutionary solution has emerged to address the complexities of the housing market – the Home Rental App. This application transforms the way individuals search for, connect with, and secure their ideal rental properties, all at the touch of a button. Users can explore a wide array of available rental properties, ranging from apartments to spacious houses, tailored to suit diverse preferences and budgets. The app's intuitive interface empowers users to set personalized search criteria, including location, price range, property size, and desired amenities. Advanced filtering options ensure that each search yields relevant results, saving valuable time and energy. Finding a new home can be a daunting task, With the help of a home rental app, you can easily search for and apply for rentals all from your smartphone. Home rental app offer a variety of features that can make the rental process easier and more efficient. These features include Search for listings by location, price, number of bedrooms and bathrooms, and other criteria**.** You can search for and book rentals from anywhere, at any time. Compare listings side-by-side to see which one is the best fit for you. This app can help you save time and money by streamlining the rental process and can connect you with other renters and owners in your area. In this app Property owners can showcase their listings with detailed descriptions, high-quality images, and essential amenities. This app provides a comprehensive platform for property owners to list their properties, manage bookings with potential renters.

## 1.1 PROBLEM STATEMENT

In today's fast-paced world, finding a suitable rental property can be a daunting and time-consuming task for individuals and families alike. The traditional methods of browsing through classified ads, contacting multiple agents, and visiting properties in person can lead to frustration, wasted time, and often inadequate results. Moreover, the lack of transparency and trustworthiness in the rental market can leave renters feeling uncertain about their choices. To address these challenges, we aim to develop a comprehensive Home Rental App that revolutionizes the rental property search process, making it efficient, convenient, and reliable. The primary objective of this app is to empower users to find their ideal rental homes seamlessly, while also ensuring that property owners can connect with potential tenants more effectively.

## 1.2 OBJECTIVES

- Facilitate easy search for rental properties.
- Connect owners and renters on the platform.
- Provide a seamless application and approval process for renters.
- Track rental trends and occupancy rates.
- Enable owners to list multiple properties under one account.

## 1.3 MOTIVATON

The motivation behind developing a home rental app stems from the growing demand for convenient and efficient solutions in the real estate and housing industry. With urbanization on the rise and an increasing number of people seeking rental properties, there is a need for a platform that can streamline the process of finding, renting, and managing homes.

## 1.4 EXISTING SYSTEM

Existing systems on Home rental app that allows users to search for and book homes to rent .It could offer features to help users find the perfect rent house for their needs. It has a user-friendly interface that allows users to filter their search by location, price, amenities, and other criteria. Once a renter finds a house they're interested in, they can contact the owner to enquire about availability and make a booking.

### DRAWBACKS OF EXISTING SYSTEM

- Fraudulent listings
- Inconsistent Listing Information
- Untrusted Transactions
- Limited Flexibility

## 1.5 PROPOSED SYSTEM

The proposed system has been developed with the idea to create an interface Between owners and renters to ensure authenticity and credibility. It also allows users to see inside the homes before they book them. This would help to reduce the need for in-person viewings, which can be time-consuming and inconvenient. This apps could improve communication between owners and renters, making it easier to resolve issues and ensure a smooth rental experience. This could be done through features such as live chat and messaging. This app could integrate with payment processors, making it easier for owners and renters to pay and be paid. This would help to streamline the rental process and reduce the risk of fraud.

## 1.6 SCOPE

- A mobile app which creates the interface between the owner and the renter.
- This project is to create a mobile app that can streamline the house rental process for both renters and property owners.
- The ultimate goal is to provide a reliable and efficient solution for individuals seeking rental properties and property owners.

# CHAPTER 2

## LITERATURE SURVEY

| S.NO | NAME | YEAR | ALGORITHM/TECHNIQUE USED | ADVANTAGES | LIMITATIONS |
|---|---|---|---|---|---|
| 1 | Magic Bricks Buy, Rent Property | Dec.2011 | Search Algorithm, Sorting Algorithms, Data Filtering and Processing Algorithms | • Wide Property Selection<br>• Advanced Search Filters<br>• Property Details and Media | • Data Accuracy<br>• Limited Availability<br>• Competitive Market |
| 2 | 99acres Buy/Rent/Sell Property | Dec 17,2013 | Machine learning algorithms, Natural language processing algorithms, Filtering algorithm, | • Advanced Search Filters<br>• User-Friendly Interface<br>• Notifications and Alerts | • Real-Time Availability<br>• Reliance on User Inputs<br>• Can be expensive, Fake listings |

| | | | | | |
|---|---|---|---|---|---|
| 3 | No Broker Property Rent & Sale | Jun 26, 2015 | Recommendation Algorithm Ranking Algorithm, Filtering Algorithm, Geocoding Algorithm, Payment algorithm | • No brokerage<br>• Large Inventory<br>• Easy to use, Secure and Reliable | • Not available in all areas<br>• Some fake listings<br>• Can be spammy<br>• Not as user-friendly as some other Apps |

# CHAPTER 3

# SYSTEM REQUIREMENTS SPECIFICATIONS

## 3.1 HARDWARE REQUIREMENTS

| | | |
|---|---|---|
| ● Processor | : | 200Mhz Processor |
| ● Hard Disk | : | 50MB |
| ● RAM | : | 2GG |

## 3.2 SOFTWARE REQUIREMENTS

| | | |
|---|---|---|
| ● Operating System | : | Android 8 |
| ● Programming Language | : | Java |
| ● Database | : | Firebase |

# CHAPTER 4

# SYSTEM DESIGN

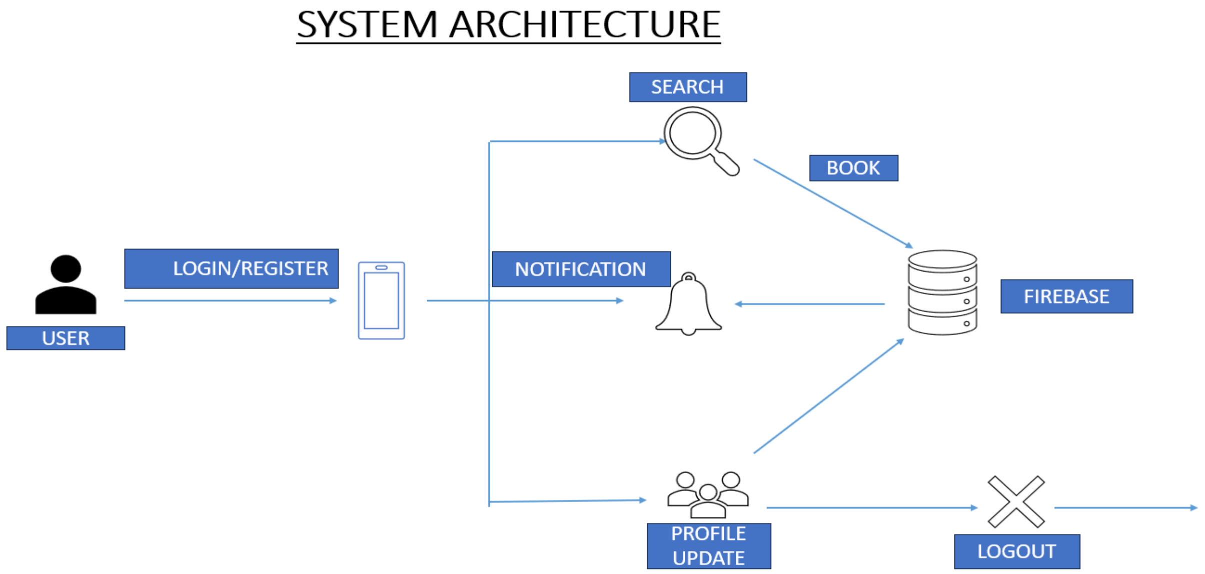


**Fig-4.1 SYSTEM ARCHITECTURE**

## 4.1.1 Architecture Description

**User Interface (UI) Components:**

**Login/Register Page:** Allows users to either log in to their existing accounts or create new ones.

**Homepage:** Displays relevant information, featured properties, and navigation options.

**Search Page:** Enables users to search for rental properties based on various criteria.

**Property Details Page:** Shows detailed information about a selected property and allows users to book it.

**Notifications:** Provides users with real-time notifications regarding their account and rental activities.

**Logout Functionality:** Allows users to log out from their accounts securely**.**

**Database:** Firebase Realtime Database can be employed to store property listings, user profiles, and booking information. Property details like location, price, amenities, and availability dates will be stored in the database.

## 4.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: A Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

### 4.2.1 Use Case Diagram

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

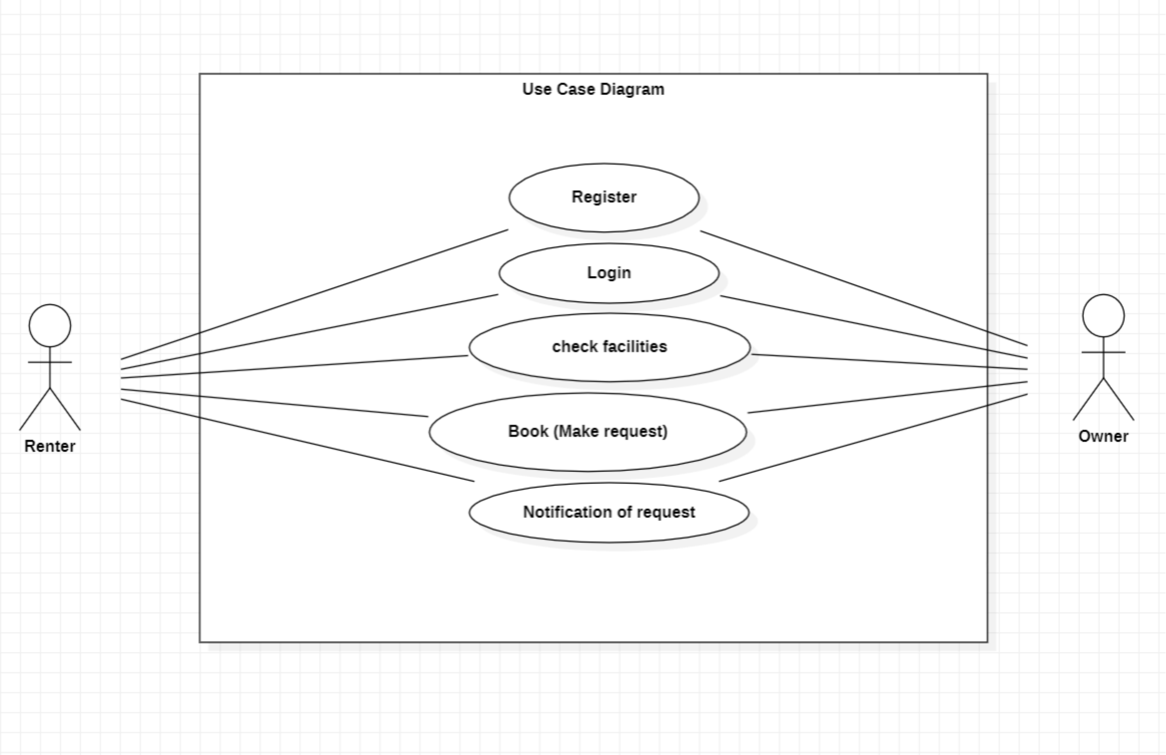


**Fig-4.2.1 USE CASE DIAGRAM**

## 4.2.2 Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains which information.

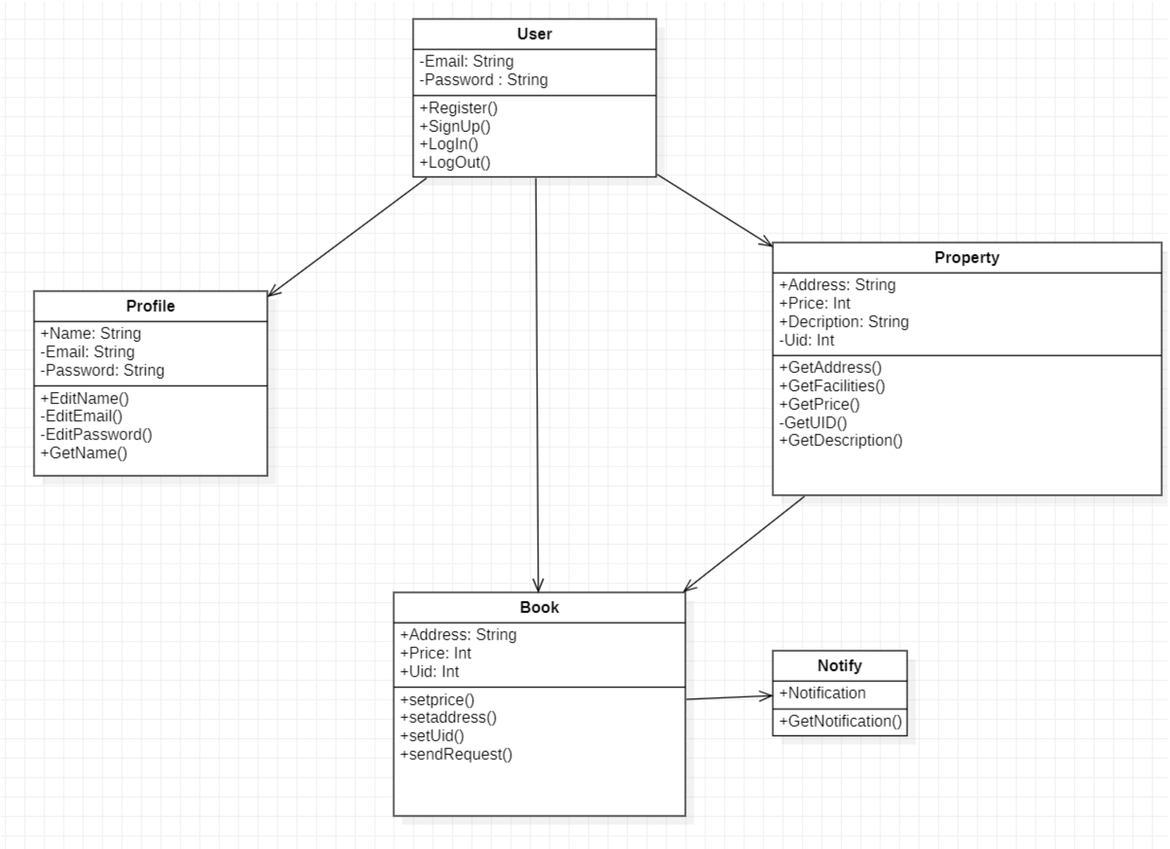


**Fig-4.2.2 CLASS DIAGRAM**

### 4.2.3 Sequence Diagram

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

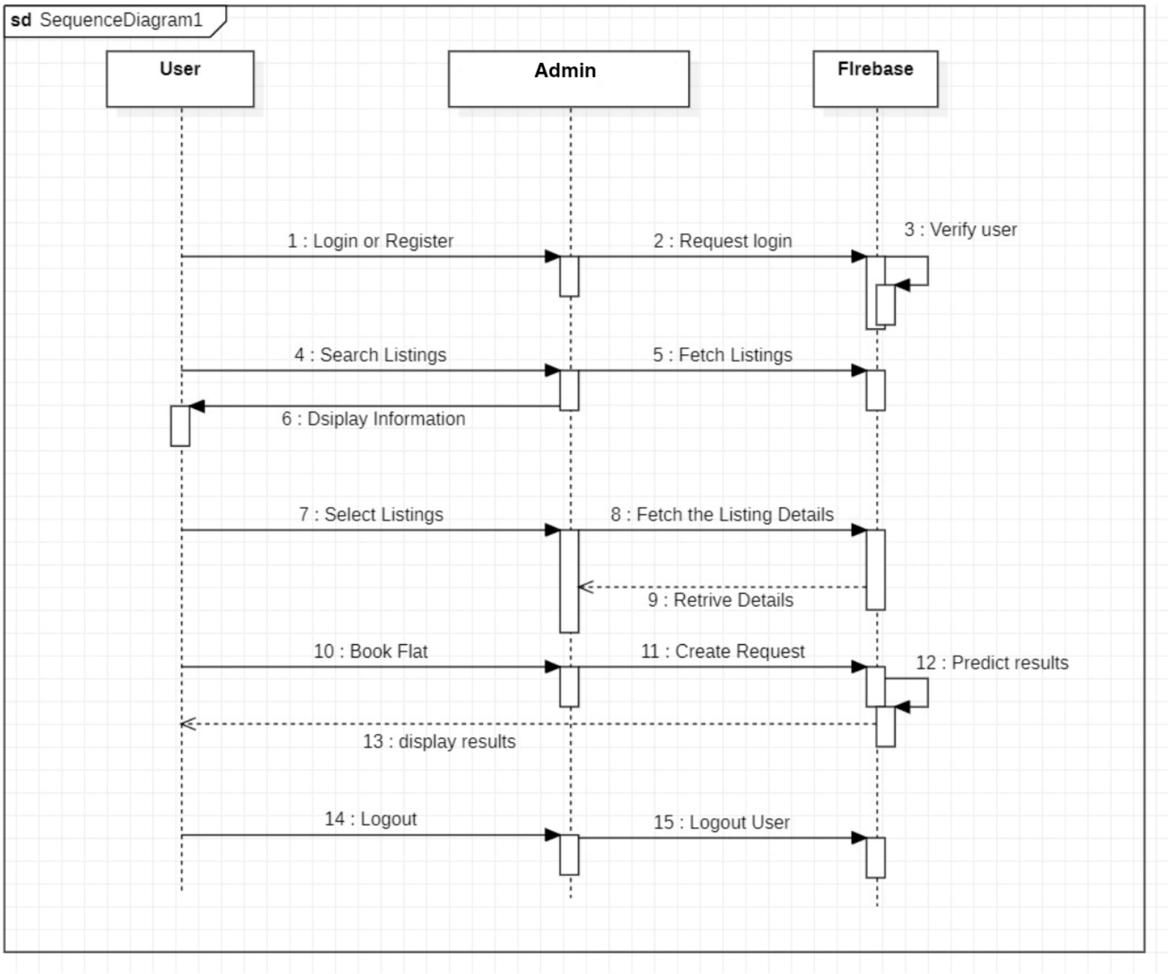


**Fig-4.2.3 SEQUENCE DIAGRAM**

### 4.2.6 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

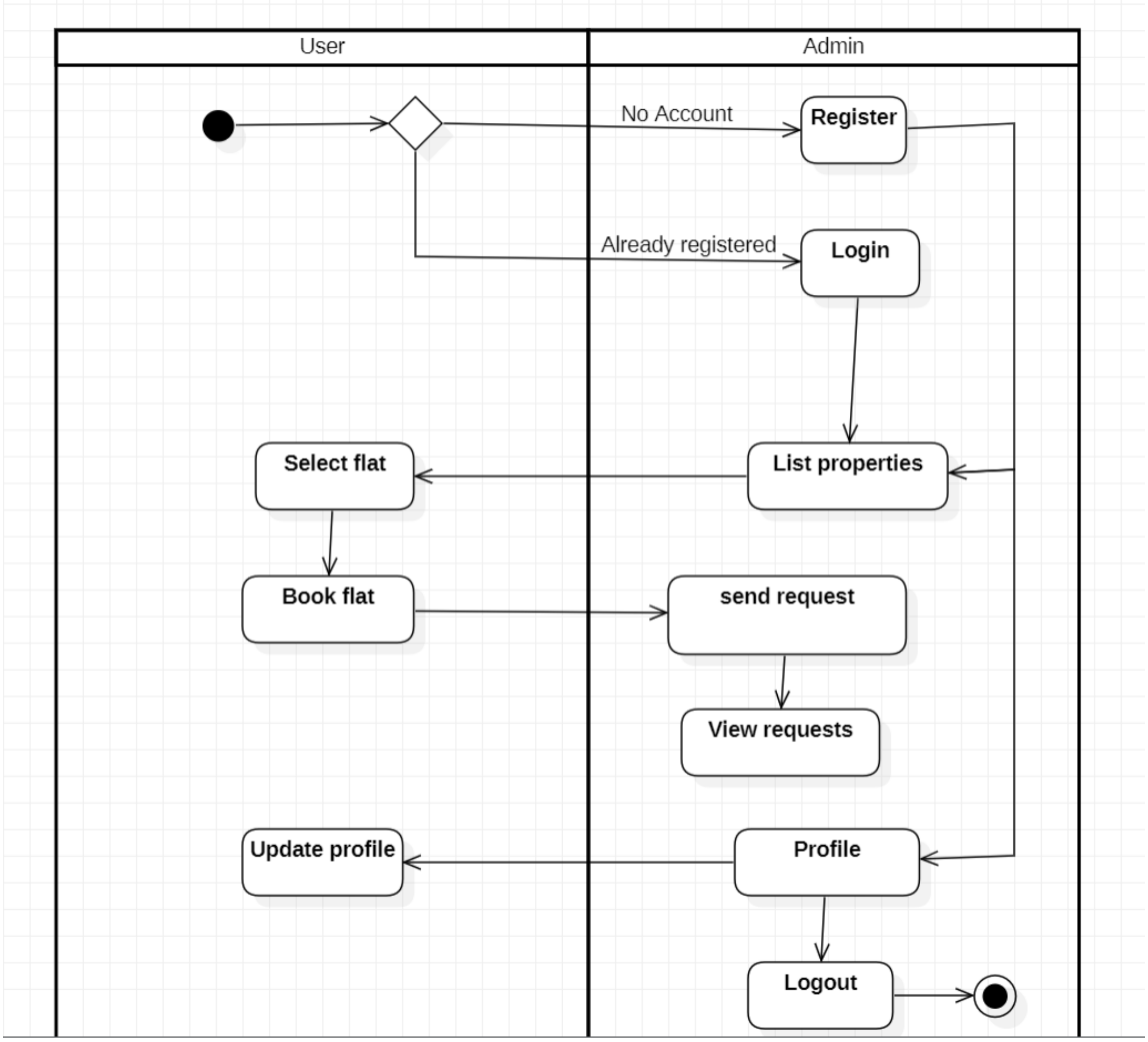


**Fig-4.2.4 ACTIVITY DIAGRAM**

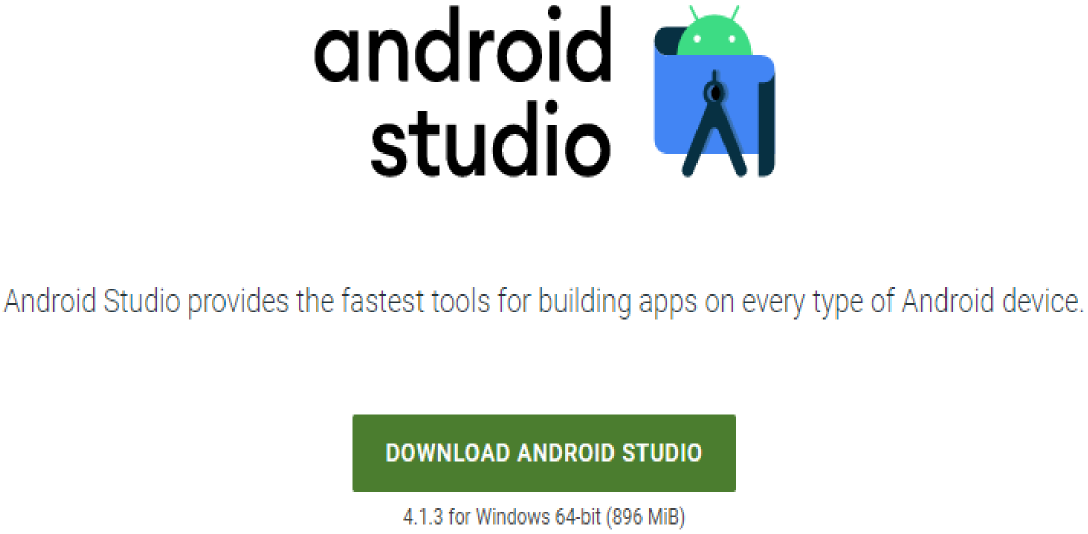
# CHAPTER 5

# IMPLEMENTATION &WORKING METHODOLOGY

## 5.1 ENVIRONMENTAL SETUP

Installing Android Studio:

1. To download and install Android Studio visit the official website of Android Studio https://developer.android.com/studio and choose your version.

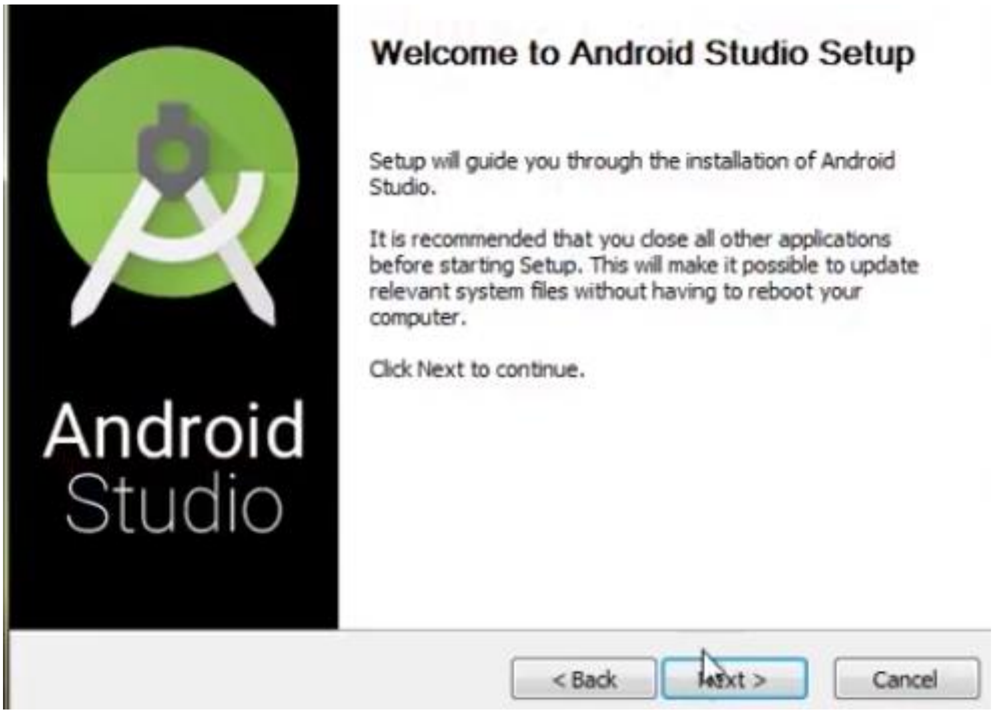


Android Studio provides the fastest tools for building apps on every type of Android device.

4.1.3 for Windows 64-bit (896 MiB)

**FIG-5.1 ANDROID STUDIO INSTALLATION**

2. Once the download is complete, run the exe to install Android Studio. Now click on Next.

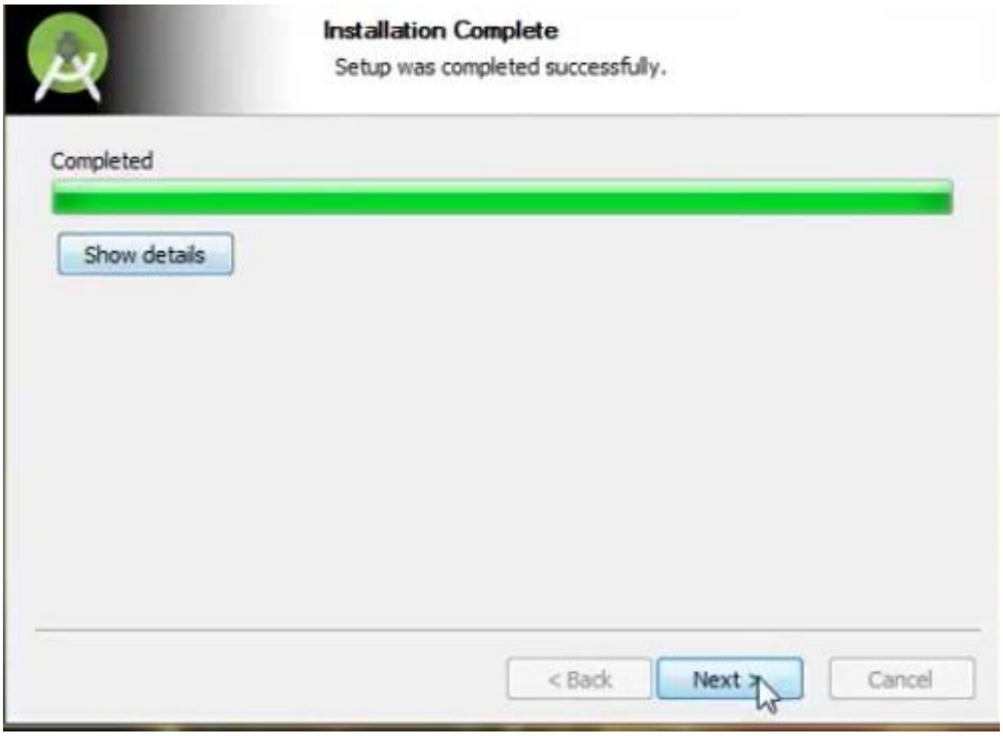


Welcome to Android Studio Setup

Setup will guide you through the installation of Android Studio.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

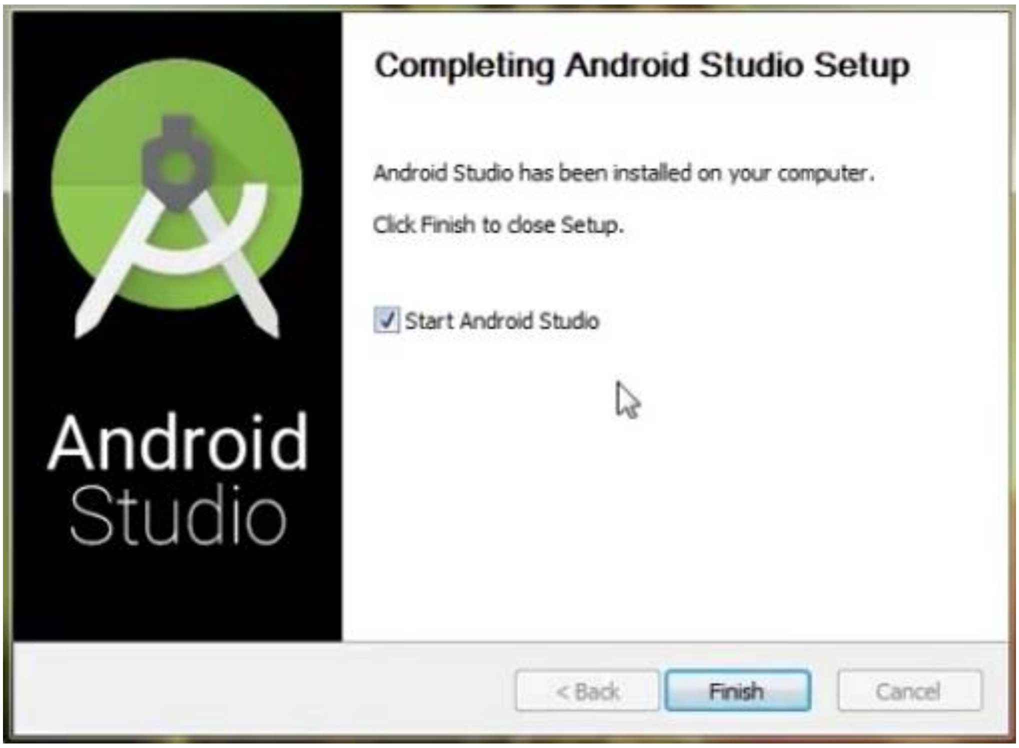


Installation Complete

Setup was completed successfully.

Completed

3. You can see Android Studio installing at this point.



Completing Android Studio Setup

Android Studio has been installed on your computer.

Click Finish to close Setup.

☑ Start Android Studio

4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Finish".

**1.USER :**

**1.1 ACCOUNT CREATION :** the user has to enter their email-ID, password and name then the account will be created.

**1.2 SELECT PROPERTY :** the user needs to select a property among the available properties.

**1.3 BOOK FLAT BUTTON:** the user can press the book flat button to send the request to the owner of the house.

**1.4 ENTER DETAILS:** the user needs to enter his details in order to confirm the request and click on the send request button to send the request to the owner of the property.

**1.2 VIEW REQUESTS**: the user can see all of his requests made by him and decisions of the owners for his requests in the notifications section.

**2. ADMIN :**

**2.1 INTERFACE:** create user interface for displaying all the properties which are uploaded by the owner and also facilitates the option to see the user requests and owner decision.

**2.2 HOME PAGE:** the home page is the area where the user can be able to see all the featured properties which are uploaded by the owners.

**2.3 HOME DETAILS PAGE:** the home details page will open when the user clicks on a property in the home page. Here the user can see the complete details of the property and can able to book the flat.

**2.4 NOTIFICATIONS PAGE:** in the notifications page the user can able to see all of his requests made for the property and the correspondent decision from the owner regarding the request.

**2.2 PROFILE PAGE**: the page can be used to see the details of the user and from here the user can be able to logout if he wants.

**3. SYSTEM:**

**3.1 USER AUTHENTICATION:** whenever a user tries to login to the app the system will check the credentials and verify them if everything is correct then it will allow the user to login else it will generate an error message.

**3.2 DATA STORAGE**: the data which is sent by the users and the owners are stored in the system(firebase) in systematic order.

**3.3 DATA RETRIVAL:** the date which is stored in the system will be sent back to the app when the data is required and it is fetched by using the methods.

**3.4 MANAGING REQUEST:** the requests which are sent by the user will be sent to the system from where it will be transferred to the owner of the corresponding property and then the decision will be again transmitted back to the user

## 5.3 MODULE DISCRIPTION

- XML
- Java
- Firebase
- Gradle

### Xml

- XML (Extensible Markup Language) is a widely used markup language for structuring and storing data. In the context of Android development, XML is commonly used in Android Studio for defining user interface layouts, handling resources, and configuring app components.
- Android Studio provides a visual editor to create and modify XML files, allowing you to drag and drop UI elements, set properties, and preview the resulting layout. However, understanding the underlying XML structure is crucial for effective Android development.
- XML is just one component of Android development, and you'll need to work with Java or Kotlin code to handle the logic and functionality of your app.

### Java

- Java is one of the primary programming languages used for Android app development in Android Studio. Android Studio provides a Java development environment and a set of libraries and APIs specifically designed for Android development.

- Android Studio provides a set of libraries and APIs (Application Programming Interfaces) that allow you to interact with various Android system components. These components include activities, services, content providers, broadcast receivers, and more. You'll need to learn how to use these components and their corresponding Java classes to build functional Android applications.

- Android Studio provides a range of development tools to assist you in writing Java code. These tools include code completion, code analysis, debugging, refactoring, and more.

### Firebase

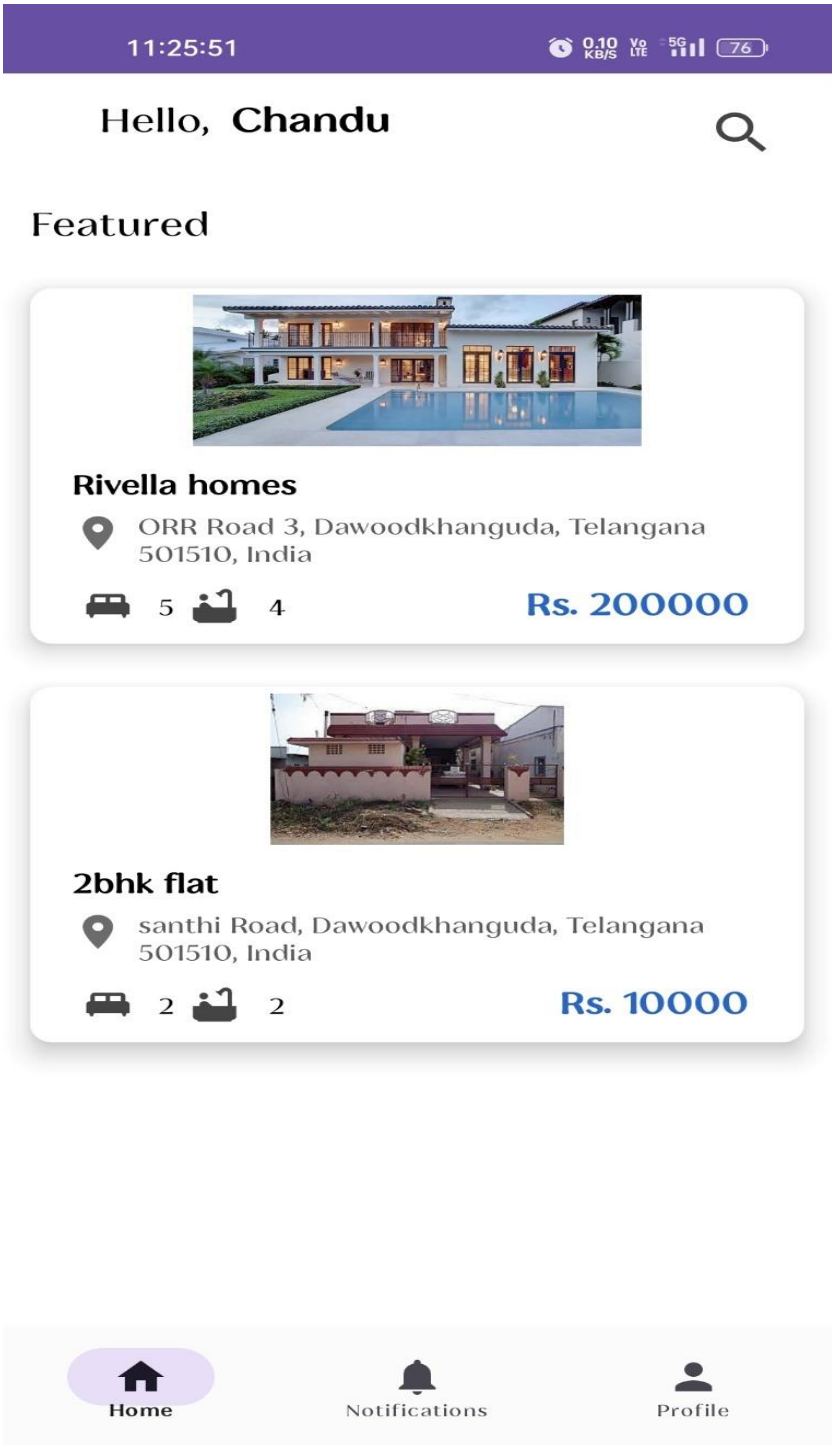
- Firebase is a powerful platform provided by Google that offers a wide range of services and tools for developing and managing mobile and web applications.
- Firebase provides backend infrastructure and functionality as a service, allowing developers to focus on building their applications without having to manage server infrastructure. Firebase is commonly used in Android Studio projects to add features such as real-time database, authentication, cloud storage, cloud messaging, and more.
- The Firebase console is a web-based interface provided by Firebase for managing your Firebase projects. Through the console, you can configure and monitor various Firebase services, view analytics data, send notifications, manage authentication, and perform other administrative tasks.

## Gradle

- Gradle is the build system used by Android Studio to compile, build, and package Android projects. It is a powerful and flexible build tool that allows you to define and manage dependencies, configure project settings, and automate various build tasks.
- Gradle simplifies dependency management by allowing you to specify external libraries and dependencies required by your app.
- Gradle provides a plugin system that extends the functionality of the build system. Android Studio uses the Android Gradle Plugin, which adds Android-specific tasks and features to Gradle. The plugin handles tasks such as compiling code, packaging the APK, generating resource files, and more. Other plugins can be added to enable additional functionality, such as code quality checks or code generation.

# CHAPTER 6
# RESULTS

## 6.1 Home Page



Hello, **Chandu**

Featured

**Rivella homes**

ORR Road 3, Dawoodkhanguda, Telangana 501510, India

5 4 Rs. 200000

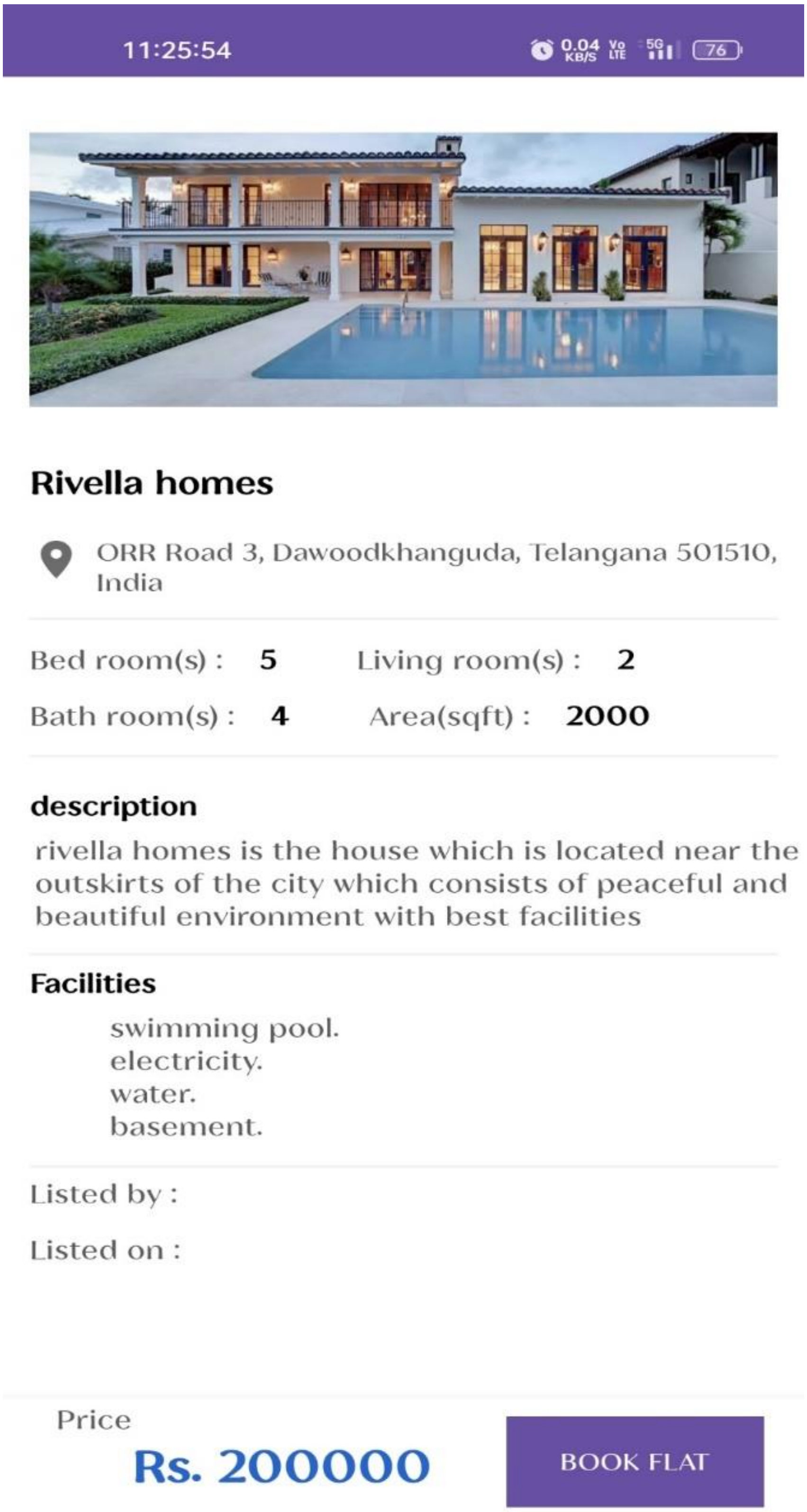
**2bhk flat**

santhi Road, Dawoodkhanguda, Telangana 501510, India

2 2 Rs. 10000

FIG-6.1 HOMEPAGE

## 6.2 Home details



### Rivella homes

ORR Road 3, Dawoodkhanguda, Telangana 501510, India

Bed room(s) : 5 Living room(s) : 2

Bath room(s) : 4 Area(sqft) : 2000

**description**

rivella homes is the house which is located near the outskirts of the city which consists of peaceful and beautiful environment with best facilities

**Facilities**

swimming pool.
electricity.
water.
basement.

Listed by :

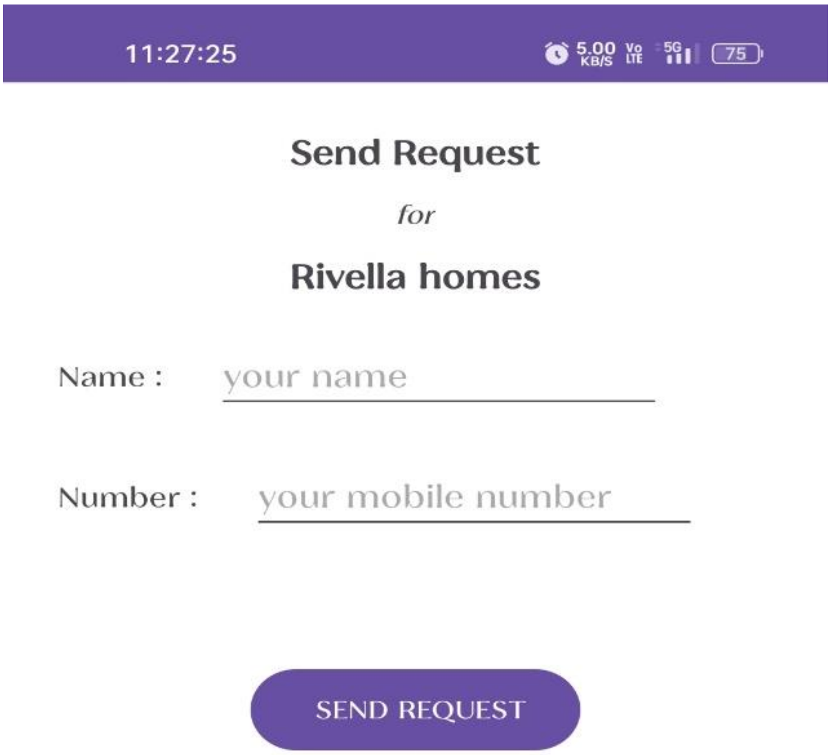
Listed on :

Price

Rs. 200000

FIG-6.2 HOME DETAILS

### 6.3 Send request



Send Request

*for*

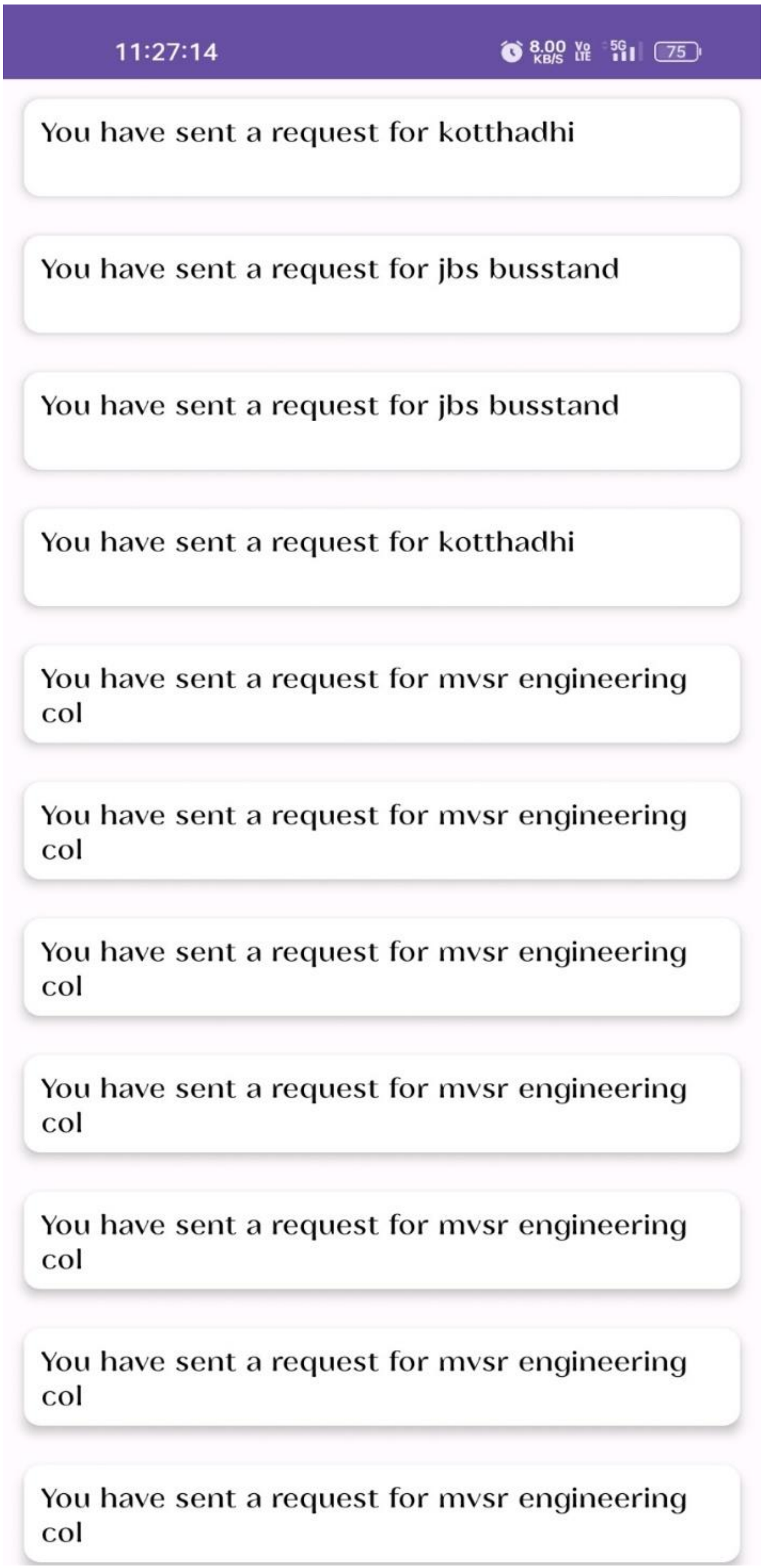
Rivella homes

Name : your name

Number : your mobile number

FIG-6.3 SEND REQUEST

### 6.4 View requests page



You have sent a request for kotthadhi

You have sent a request for jbs busstand

You have sent a request for jbs busstand

You have sent a request for kotthadhi

You have sent a request for mvsr engineering col

You have sent a request for mvsr engineering col

You have sent a request for mvsr engineering col

You have sent a request for mvsr engineering col

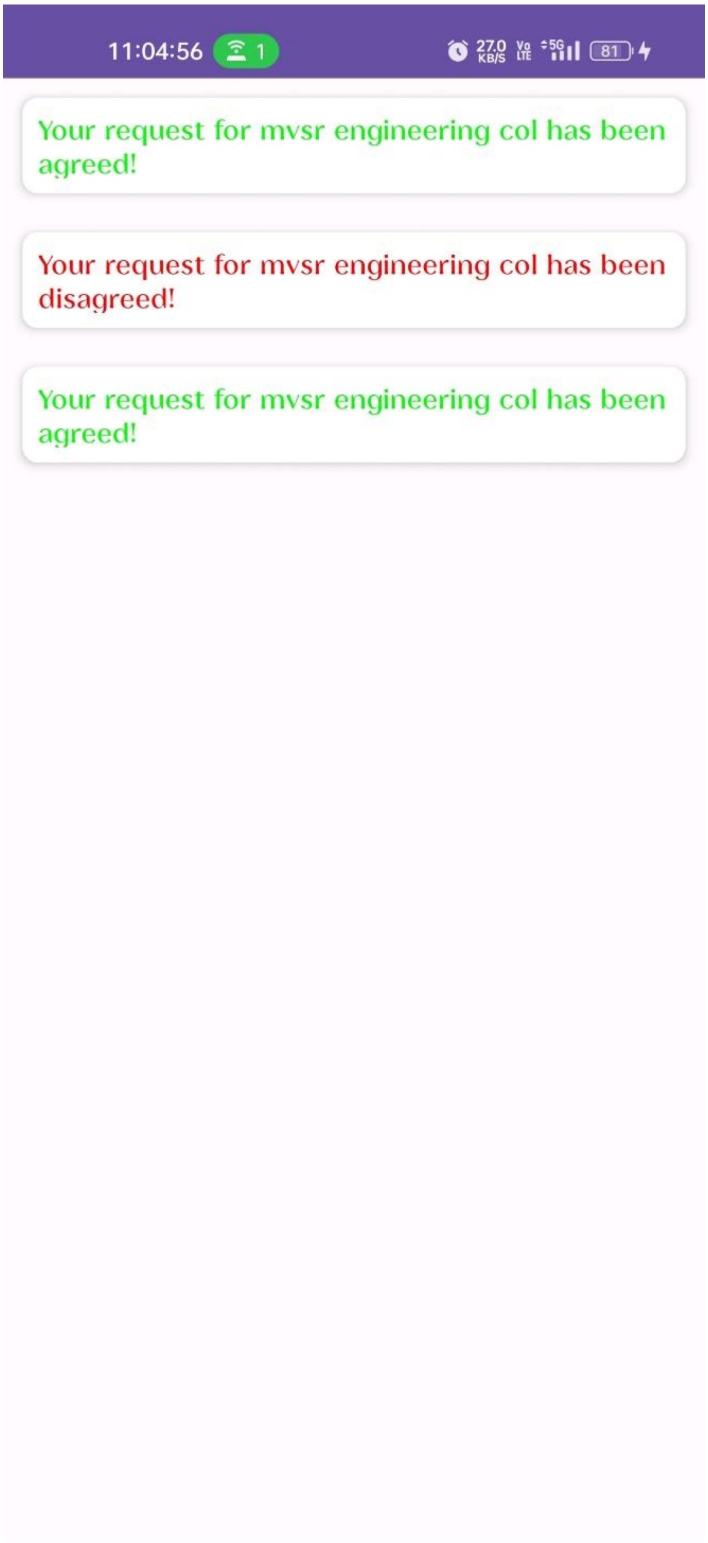
You have sent a request for mvsr engineering col

You have sent a request for mvsr engineering col

You have sent a request for mvsr engineering col

FIG-6.4 VIEW REQUESTS PAGE

### 6.5 Decisions Page



11:04:56

Your request for mvsr engineering col has been agreed!

Your request for mvsr engineering col has been disagreed!

Your request for mvsr engineering col has been agreed!

FIG-6.5 DECISIONS PAGE

# CHAPTER 7
# TESTS

## 7.1 TESTING

### 7.1.1 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

- All field entries must work properly.
- Every button must work properly.
- Pages must be activated from the given options.
- The entry screen, messages and responses must not be delayed.

**Features to be tested**

- Verify that the entries are of the correct format.
- Verify that all the properties are listed correctly.
- Verify that the request from the user is going to the owner of the house.
- All buttons and options should take the user to the correct page.

### 7.1.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

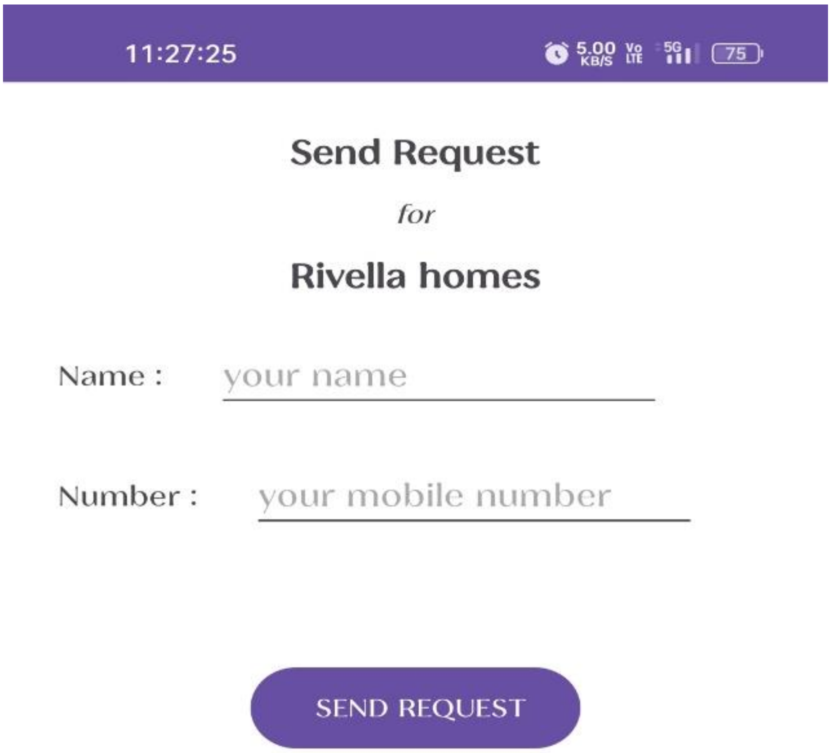
### 7.1.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

## 7.2TEST CASES

- SENDING REEQUEST



11:27:25

Send Request

*for*

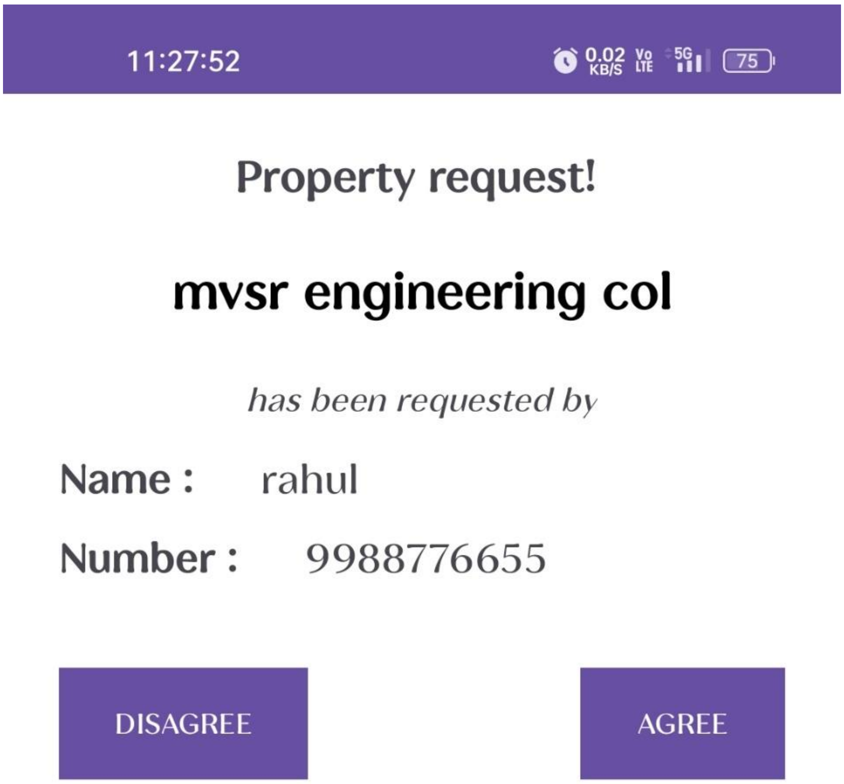
Rivella homes

Name : your name

Number : your mobile number

FIG-7.2.1 SENDING REQUEST

- ACCEPTING REQUEST



Property request!

mvsr engineering col

*has been requested by*

Name : rahul

Number : 9988776655

FIG-7.2.2 ACCEPTING REQUEST

# CHAPTER 8
# CONCLUSION AND FUTURE ENHANCEMENTS

In this work, we are able to develop a Home rental app which acts as an interface between the seller and renter. In the proposed method we made sure that using the app owner of the house can list his property to the app and then the property is available for everyone to book. Once a property is listed to the app every user who is using the app can see the property which is listed by the owner and the interested user can send the book request to the owner and they can discuss the deal. Hence, we are concluding that in this project we are trying to create an easy and user-friendly interface between the user and the owner which makes the buy and sell of the property seamless.

# REFERENCES

- O Singh, A Lakhan and J Gupta, "Implementation of an Android Application for Management of a Housing Society", *International Journal of Engineering and Computer Science*, vol. 4, no. 6, June 2015.
- Sonya R. Manalu ,Aswin Wibisurya, Natalia Chandra and Alan Patrick Oedijanto, "Development and Evaluation of Mobile Application for Room Rental Information With Chat and Push Notification", *2016 International Conference on Information Management and Technology (ICIMTech)*, pp. 7-11, 2016
- Kirmani, J.A., Yousuf, A. and Bhat, S.M., 2017. Rental Housing Management System. International Journal of Computer Science and Mobile Computing, 6(7), pp.1-4.

# APPENDIX

## Source code:

```
package com.codeinfinity.homescope;

import android.content.Intent;
import android.os.Bundle;
import androidx.cardview.widget.CardView;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class FavFragment extends Fragment {

    FirebaseAuth auth = FirebaseAuth.getInstance();
    FirebaseUser user = auth.getCurrentUser();
    String currentUserId = user.getUid();

    public FavFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {
        View = inflater.inflate(R.layout.fragment_fav, container,
false);

        CardView request = view.findViewById(R.id.requestCard);
        CardView decision = view.findViewById(R.id.decisionCard);

        request.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(getActivity(),
MyRequests.class));
            }
        });

        decision.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(getActivity(),
DecisionActivity.class));
            }
        });


        return view;

    }


package com.codeinfinity.homescope;

import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;
import androidx.recyclerview.widget.LinearLayoutManager;

import com.bumptech.glide.Glide;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.util.ArrayList;
import java.util.List;

public class HomeDetailsActivity extends AppCompatActivity {

    private String propertyName, uid;
    public String priceTextView;

    public String descriptionTextView;
    public String addressTextView;

    public String homeAreas;
    public String beds;
    public String baths, facilities, dates;
    String living;
    String imagePath;

    Button bookFlat;

    FirebaseAuth auth = FirebaseAuth.getInstance();
    FirebaseUser user = auth.getCurrentUser();

    String currentUserId = user.getUid();

    public ImageView;
    ProgressBar;
    TextView name, desc, address, price, area, bed, bath, livingRoom,
facility, nameUser, dateUpload, sent;
    ImageView image;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home_details);


        name = findViewById(R.id.nameHome);
        image = findViewById(R.id.homeDetailsImage);
        desc = findViewById(R.id.descriptionHome);
        address = findViewById(R.id.addressFlat);
        price = findViewById(R.id.priceHome);
        area = findViewById(R.id.areaValue);
        bed = findViewById(R.id.bedNoTxt);
        bath = findViewById(R.id.bathNoTxt);
        livingRoom = findViewById(R.id.livingNoTxt);
        facility = findViewById(R.id.facilitiesText);
        nameUser = findViewById(R.id.userNameText);
        dateUpload = findViewById(R.id.dateText);
        bookFlat = findViewById(R.id.homeRegister);
        progressBar = findViewById(R.id.progress);
        sent = findViewById(R.id.sentText);

        DatabaseReference userRef =
FirebaseDatabase.getInstance().getReference("sellerDetails").child(curre
ntUserId).child("name");
        userRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                String user = snapshot.getValue(String.class);
                nameUser.setText(user);
            }

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

            }
        });

        Intent = getIntent();
        if (intent != null) {

            propertyName = intent.getStringExtra("property_name");
            priceTextView = intent.getStringExtra("property_price");
            descriptionTextView =
intent.getStringExtra("property_description");
            addressTextView = intent.getStringExtra("property_address");
            homeAreas = intent.getStringExtra("property_area");
            beds = intent.getStringExtra("property_beds");
            baths = intent.getStringExtra("property_baths");
            living = intent.getStringExtra("property_living");
            facilities = intent.getStringExtra("property_facilities");
            imagePath = intent.getStringExtra("image_path");
            dates = intent.getStringExtra("property_date");
            uid = intent.getStringExtra("property_uid");


            Glide.with(this)
                    .load(imagePath)  //Assuming you have a
getImageUrl() method in your Property class that returns the image URL
                    .into(image);
        }
        name.setText(propertyName);
        desc.setText(descriptionTextView);
        address.setText(addressTextView);
        price.setText("Rs.  " + priceTextView);
        facility.setText(facilities);
        area.setText(homeAreas);
        bed.setText(beds);
        dateUpload.setText(dates);
        bath.setText(baths);
        livingRoom.setText(living);


        bookFlat.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent1 = new Intent(HomeDetailsActivity.this,
SendRequestActivity.class);
                intent1.putExtra("homeName", propertyName);
                intent1.putExtra("homePrice", priceTextView);
                intent1.putExtra("sellerUid", uid);
                startActivity(intent1);





            }
        });








    }
}


package com.codeinfinity.homescope;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.os.Handler;
import android.view.MenuItem;
import android.view.View;
import android.widget.ProgressBar;

import
com.google.android.material.bottomnavigation.BottomNavigationView;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
```

```
public class MainActivity extends AppCompatActivity {

    private BottomNavigationView bottomNavigationbuyer;

    FirebaseAuth auth = FirebaseAuth.getInstance();
    FirebaseUser user = auth.getCurrentUser();


    String uid = user.getUid();

    FirebaseDatabase database = FirebaseDatabase.getInstance();

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


        bottomNavigationbuyer = findViewById(R.id.bottomNavBarBuyer);

        getSupportFragmentManager().beginTransaction()
                .replace(R.id.fragment_container, new HomeFragment())
                .commit();


        bottomNavigationbuyer.setOnNavigationItemSelectedListener(
                new
BottomNavigationView.OnNavigationItemSelectedListener() {
                    @Override
                    public boolean onNavigationItemSelected(@NonNull
MenuItem item) {
                        if (item.getItemId() == R.id.homeMenu) {

getSupportFragmentManager().beginTransaction()
                                    .replace(R.id.fragment_container,
new HomeFragment())
                                    .commit();
                        } else if (item.getItemId() == R.id.favMenu) {

getSupportFragmentManager().beginTransaction()
                                    .replace(R.id.fragment_container,
new FavFragment())
                                    .commit();
                        } else if (item.getItemId() == R.id.profileMenu)
{

getSupportFragmentManager().beginTransaction()
                                    .replace(R.id.fragment_container,
new ProfileFragment())
                                    .commit();
                        }

                        return true;
                    }
                });
}
}
```