

# **PROPERTY TALASH**



**Project/Thesis ID. 2023: 111**

**Session: BSc. Spring 2023**

**Project Supervisor: M. Nouman Noor**

**Submitted By**

**Aena Qadeer**

**Zain Shabbir**

---

**Computer Science**

**HITEC University Taxila**

## **Certification**

---

This is to certify that **Aena Qadeer, 19-CS-005** and **Zain Shabbir, 19-CS-091** have successfully completed the final project **Property Talash**, at the **HITEC University**, to fulfill the partial requirement of the degree **BSCS**.

**Project Supervisor**

M. Nouman Noor

Lecturer

Department of Computer Science, HITEC University Taxila

## Project Title (Property Talash)

### Sustainable Development Goals

SDG No	Description of SDG	SDG No	Description of SDG
SDG 1	No Poverty	SDG 9	Industry, Innovation, and Infrastructure ✓
SDG 2	Zero Hunger	SDG 10	Reduced Inequalities
SDG 3	Good Health and Well Being	SDG 11	Sustainable Cities and Communities
SDG 4	Quality Education	SDG 12	Responsible Consumption and Production
SDG 5	Gender Equality	SDG 13	Climate Change
SDG 6	Clean Water and Sanitation	SDG 14	Life Below Water
SDG 7	Affordable and Clean Energy	SDG 15	Life on Land ✓
SDG 8	Decent Work and Economic Growth	SDG 16	Peace, Justice and Strong Institutions
		SDG 17	Partnerships for the Goals



<b>Range of Complex Problem Solving</b>			
	<b>Attribute</b>	<b>Complex Problem</b>	
1	Range of conflicting requirements	Involve wide-ranging or conflicting technical, engineering and other issues.	
2	Depth of analysis required	Have no obvious solution and require abstract thinking, originality in analysis to formulate suitable models.	
3	Depth of knowledge required	Requires research-based knowledge much of which is at, or informed by, the forefront of the professional discipline and which allows a fundamentals-based, first principles analytical approach.	
4	Familiarity of issues	Involve infrequently encountered issues	
5	Extent of applicable codes	Are outside problems encompassed by standards and codes of practice for professional engineering.	
6	Extent of stakeholder involvement and level of conflicting requirements	Involve diverse groups of stakeholders with widely varying needs.	
7	Consequences	Have significant consequences in a range of contexts.	
8	Interdependence	Are high level problems including many component parts or sub-problems	
<b>Range of Complex Problem Activities</b>			
	<b>Attribute</b>	<b>Complex Activities</b>	
1	Range of resources	Involve the use of diverse resources (and for this purpose, resources include people, money, equipment, materials, information and technologies).	
2	Level of interaction	Require resolution of significant problems arising from interactions between wide ranging and conflicting technical, engineering or other issues.	
3	Innovation	Involve creative use of engineering principles and research-based knowledge in novel ways.	
4	Consequences to society and the environment	Have significant consequences in a range of contexts, characterized by difficulty of prediction and mitigation.	
5	Familiarity	Can extend beyond previous experiences by applying principles-based approaches.	

## Abstract

---

Property Talash is a groundbreaking app built with Flutter that aims to tackle the prevalent issue of the absence of reliable property review platforms. In contrast to existing options, our app introduces a novel approach by giving the power to the locals, allowing them to express their genuine opinions on communities and culture. This unique feature enables users to gain valuable insights and make well-informed decisions when searching for homes. One of the key advantages of Property Talash is its ability to streamline the process of accessing property advice from experienced through community. Users can easily connect with these professionals to seek guidance and expert recommendations regarding the properties available in their desired area. This direct interaction ensures that users receive accurate and personalized information, enhancing their overall property hunting experience. Moreover, our app provides a comprehensive platform for exploring community reviews. Users can delve into detailed feedback and impressions shared by individuals who are intimately familiar with the locality. This valuable information sheds light on various aspects such as amenities, safety, convenience, and the overall quality of life in different neighborhoods. By leveraging these reviews, users can assess and compare different homes to find the one that best aligns with their specific needs and preferences. The convenience and reliability offered by Property Talash make it a go-to solution for property seekers. With a user-friendly interface and intuitive features, our app simplifies the process of finding the ideal home. By combining the expertise of local dealers and the genuine insights of community reviews, Property Talash revolutionizes the way people make decisions about properties, making it an essential tool in the real estate market.

## **Undertaking**

---

I certify that the project **PROPERTY TALASH** is our own work. The work has not, in whole or in part, been presented elsewhere for assessment. Where material has been used from other sources it has been properly acknowledged/ referred.

Aena Qadeer

19-CS-005

Zain Shabbir

19-CS-091

## **Acknowledgement**

---

We would also like to thank **M. Nouman Noor** from **HITEC University Taxila** for his help and guidance throughout this project. He has been a constant source of guidance throughout the course of this project.

We are also thankful to our friends and families whose silent support led us to complete our project.

## Table of contents

### Contents

Certification.....	2
Abstract.....	5
Undertaking.....	6
Acknowledgement .....	7
Chapter 1.....	1
1. Introduction .....	1
1.1. Project Overview .....	1
1.2. Project Scope .....	2
1.2.1. Computer/ Laptop Interface .....	2
1.2.2. Mobile Interface .....	2
1.2.3. Problem Statement .....	2
1.3. Proposed Solution .....	2
1.3.1. Proposed System Component .....	2
1.3.1.1. Administration management .....	2
1.4. Proposed System Output.....	3
1.4.1. User management .....	3
1.4.2. Developer management.....	3
1.4.3. Main features of proposed solution .....	3
1.4.3.1. Efficiency.....	3
1.4.3.2. User friendly interface.....	3
1.4.3.3. Minimum redundancy .....	3
1.4.3.4. Facilitated data inputs .....	3
1.5. Data security and integrity .....	3
1.6. Technical innovation .....	3
1.7. Existing System .....	3
Chapter 2.....	4



<b>2. Software requirements specifications .....</b>	<b>4</b>
<b>2.1. Purpose .....</b>	<b>4</b>
<b>2.2. Scope.....</b>	<b>4</b>
<b>2.3. Proposed Solution Overview .....</b>	<b>4</b>
<b>2.3.1. Modules of the proposed solution .....</b>	<b>4</b>
<b>2.3.2. Proposed Solution .....</b>	<b>4</b>
<b>2.4. Specification requirements .....</b>	<b>5</b>
<b>2.4.1. Functional Requirement.....</b>	<b>5</b>
<b>2.4.2. Non-Functional Requirement .....</b>	<b>5</b>
<b>2.5. Advantages .....</b>	<b>6</b>
<b>2.5.1. Advantages for the Users.....</b>	<b>6</b>
<b>2.5.2. Advantages for the Collaborated Companies .....</b>	<b>6</b>
<b>2.6. Product functions .....</b>	<b>6</b>
<b>2.6.1. Functional requirements explanation .....</b>	<b>6</b>
<b>2.7. General requirements .....</b>	<b>6</b>
<b>2.7.1. Software requirements .....</b>	<b>6</b>
<b>2.7.2. Hardware requirements .....</b>	<b>7</b>
<b>2.8. Use case diagrams.....</b>	<b>7</b>
<b>2.8.1 User end .....</b>	<b>7</b>
<b>2.8.2 Admin end.....</b>	<b>9</b>
<b>Chapter 3.....</b>	<b>11</b>
<b>3.1. Use case diagram .....</b>	<b>11</b>
<b>3.2. Class diagram .....</b>	<b>12</b>
<b>3.3. Activity diagram.....</b>	<b>13</b>
<b>3.3.1. Login.....</b>	<b>13</b>
<b>3.3.2. Signup .....</b>	<b>14</b>
<b>3.3.3. Main screen.....</b>	<b>15</b>
<b>3.4. Sequence diagram.....</b>	<b>16</b>
<b>3.4.1. Login.....</b>	<b>16</b>
<b>3.4.2. Signup .....</b>	<b>17</b>
<b>3.4.3. Main screen.....</b>	<b>18</b>
<b>3.6. Data dictionary .....</b>	<b>19</b>

3.6.1. User.....	19
3.6.2. City .....	19
3.6.3. City has a collection name society.....	20
3.6.4. Each society has a collection of category.....	21
3.6.5. Admin has two documents of pending and approved reviews.....	21
3.6.6. Pending reviews .....	22
3.6.7. Approved reviews.....	22
3.6.8. This is storage where we have stored images .....	23
3.6.9. Authentication where authenticate users are shown .....	23
Chapter 4.....	24
<b>4. PROJECT MANAGEMENT .....</b>	<b>24</b>
4.1. Project deliverables .....	24
4.2. Risk Management.....	24
4.3. Purpose.....	25
4.4. Risk management functions .....	25
Chapter 5.....	26
<b>5. IMPLEMENTATION.....</b>	<b>26</b>
5.1. Operating environment .....	26
5.1.1. Operating system .....	26
5.1.2. Flutter .....	26
5.1.2.1. Flutter features .....	26
5.1.2.2. System requirements for flutter .....	26
5.1.3. Firebase database server .....	26
5.1.4. Dart.....	27
5.1.4.1. Dart features.....	27
5.1.4.2. System requirements for dart .....	27
Chapter 6.....	28
<b>6. Software Testing: .....</b>	<b>28</b>
6.1. Deriving test case specifications: .....	28
6.2. Testing Environment .....	28
6.2.1. Hardware Requirements .....	28
6.2.2. Software Requirements .....	28

6.2.3. Testing Identifications .....	28
6.3. Testing procedure.....	29
6.4. Test Cases.....	29
6.4.1. Test Case for Sign up module.....	29
6.4.1.1. Sign up as a User.....	29
6.4.2. Test Case for Login Module .....	31
6.4.2.1. Log in as a User .....	31
6.4.3. Test Case for User account module .....	33
6.4.3.1. Update user information.....	33
6.4.4. Test Case for Search Module.....	35
6.4.5 Test Case for Submit review module .....	37
6.4.6 Test Case for Add to Favorite Module .....	39
6.4.7 Test Case for Sign out module.....	41
6.4.8 Test Case for Add society record.....	43
6.4.9. Test Case for Delete society record.....	44
6.4.10. Test Case for Pending Review .....	46
6.4.11. Test Case for Approved Review.....	48
Chapter 7.....	50
7. Project Display Screens .....	50
7.1. Mobile Application.....	50
7.1.1. Splash screen .....	50
7.1.2. Main screen .....	50
7.1.3. Favourite .....	51
7.1.4. Profile setting.....	52
7.1.5. Society information.....	52
7.1.6. Add review .....	53
7.1.7. Search city .....	54
7.1.8. Search society .....	54
7.1.9. Side menu bar.....	55
7.1.10. Login .....	56
7.1.11. Reset password .....	56
7.1.12. Sign up.....	57

<b>7.2.</b>	<b>Admin Panel .....</b>	<b>58</b>
7.2.1.	Dashboard .....	58
7.2.2.	Societies Record.....	58
7.2.3.	Add Record .....	59
7.2.4.	User Record.....	60
7.2.5.	Approved Reviews .....	60
7.2.6.	Pending Reviews.....	61

## List of Figures

Figure 2.8.1 (a): User .....	7
Figure 2.8.1 (b): Login.....	8
Figure 2.8.1 (c): Search society in desired city .....	8
Figure 2.8.1 (d): Post a review .....	9
Figure 2.8.2 (a): Maintenance .....	10
Figure 2.8.2 (b): Verification .....	10
Figure 3.1: Use case diagram of whole system .....	11
Figure 3.2: Class diagram of whole system .....	12
Figure 3.3.1: Activity diagram of login .....	13
Figure 3.3.2: Activity diagram of sign up .....	14
Figure 3.3.3: Activity diagram of main screen .....	15
Figure 3.4.1: Sequence diagram of log in.....	16
Figure 3.4.2: Sequence diagram of sign up.....	17
Figure 3.4.3: Sequence diagram of main screen.....	18
Figure 3.6.1: User.....	19
Figure 3.6.2: City .....	19
Figure 3.6.3 (a): City collection name .....	20
Figure 3.6.3 (b): Each society within city collection.....	20
Figure 3.6.4 : Each society with collection name.....	21
Figure 3.6.5: Admin.....	21
Figure 3.6.6: Pending reviews.....	22
Figure 3.6.7: Approved reviews.....	22
Figure 3.6.8: Storage .....	23
Figure 3.6.9: Authentication process.....	23
Figure 7 .1.1: Splash Screen.....	50
Figure 7.1. 2: Main screen.....	51
Figure 7 .1.3: Favourite .....	51
Figure 7.1.4: Profile setting.....	52
Figure 7.1.5: Society information .....	53
Figure 7.1.6: Add review .....	53
Figure 7.1.7: Search city.....	54
Figure 7.1.8: Search society .....	55
Figure 7.1.9: Side menu bar .....	55
Figure 7.1.10: Login.....	56
Figure 7.1.11: Reset password .....	57
Figure 7.1.12: Sign up.....	57
Figure 7.2.1: Admin Dashboard .....	58
Figure 7.2.2: Societies Record.....	59
Figure 7.2.3: Add Society Record.....	59
Figure 7.2.4: User Record .....	60
Figure 7.2.5: Approved reviews Record .....	61
Figure 7.2.6: Pending reviews Record .....	61

## List of tables

Table 4.1 (Project deliverable) .....	24
Table 6.4.1.1 ( Sign up testing ).....	31
Table 6.4.2.1 ( Login testing ).....	32
Table 6.4.3.1 ( User information update testing ) .....	35
Table 6.4.4 ( Search module testing ) .....	36
Table 6.4.5 ( Submit review module testing ).....	39
Table 6.4.6 ( Add to favorite module testing ).....	41
Table 6.4.7 ( Sign out testing ) .....	42
Table 6.4.8 ( Add society record module testing ).....	44
Table 6.4.9 ( Delete society record module testing ).....	46
Table 6.4.10 ( Pending review module testing ) .....	48
Table 6.4.11 ( Approved review module testing ).....	49

# Chapter 1

## 1. Introduction

Property Talash is a Flutter-based app for reviewing properties. According to our research, there isn't a platform that provides real reviews of properties. We made the decision to provide this platform to the entire public as a result. Only locals will express thoughts about their community or culture. Our endeavor has to do with property. Property dealers must now be contacted in order to advise people about the properties in the area with better facilities. Our app allows users to explore reviews of the communities they are interested in and decide which Property best meets their needs. Thus, this process will be streamlined and made easier by our app.

Our problem is special or important in certain respects because there isn't a legitimate property platform that delivers a review like this one. One has a choice.

### 1.1. Project Overview

Our investigation indicates that there isn't a website that offers genuine reviews of properties. As a result, we decided to make this platform available to the whole public. Only residents will voice opinions on their neighborhood or culture.

The user of our application was able to post remarks regarding the location or society in question. The public benefits from it and we obtain real reviews in this way.

- User must have valid email, phone number and password to create their account and to Login.
- If user want to save his/her searched society, they can add them to their favorites.
- The user can search for their desired society by selecting their city and society name from the drop down menu in the search bar which shows them the society result or if they only select a city, all societies that are present in that city will be shown.
- The user can see verified reviews and rating of any society given by others that currently live in that society.
- If users want to give a review and rating, he/she must have to upload their CNIC/ Bill picture to verify their location before they give a review about any society and If the verification fails user will not allow giving a review.
- At the backend admin manage to authenticate user and also approve reviews given by authenticated user.

## **1.2. Project Scope**

The main purpose of Property Talash is to provide proper and authentic reviews about societies. The software product that we will be building is a Real Estate application, known as Property Talash. User has a chance to check a verified review about any society which will help them to choose which society is suitable according to their needs.

### **1.2.1. Computer/ Laptop Interface**

The user will be able to access our website through his/her computer or laptop.

### **1.2.2. Mobile Interface**

The user will also be able to access our website by using his/her Android Mobile phone's web browsers.

### **1.2.3. Problem Statement**

According to our research, there is no such platform that provides authentic reviews about any society, instead, people must conduct their own research into the reputation and available resources in their desired society.

## **1.3. Proposed Solution**

- First, we will authenticate the user by checking his/her live location or by uploading CNIC/ Bill. After checking the user, we allowed the user to write his comments about their respective place/society. This is how we get genuine reviews and public get benefit from it.
- The highest rated societies on the relevant city search will be suggested by a recommendation system. A profanity filter will be added to the system to examine sensor strings for offensive words.

### **1.3.1. Proposed System Component**

Some of the proposed system components are as follows.

#### **1.3.1.1. Administration management**

The majority of system control is in the hands of ADMIN. The administrator will be in charge of checking and verifying user and review information in accordance with system specifications. The administrator will have full access to add user reviews, post reviews, and update reviews for new users. Admin can also keep checks on the user information for security purpose and he/she can manage the modifications according to the system and keep user update about it by managing notifications.



## **1.4. Proposed System Output**

After our system is successfully finished, we will be able to work with many organizations, including zameen.com, grana.com, and other property brand organizations that require our project type application for their business. We ought to enable user account creation, reviews of their gas or electricity bills, and the ability to look up and publish reviews of other societies in accordance with requirements.

### **1.4.1. User management**

The user will be able to get access to our system by using any browser. He/she can create an account and see reviews about societies. The necessary information regarding user will be maintained and recorded by our system.

### **1.4.2. Developer management**

Developer will be able to deploy the system on web browser. He/she can update and modify the system in future. The information about developer will be maintained here.

### **1.4.3. Main features of proposed solution**

#### **1.4.3.1. Efficiency**

The system is more efficient since it responds to the user quickly and uses fewest resources possible.

#### **1.4.3.2. User friendly interface**

The system's user interface is straightforward but attractive. The user can simply grasp it.

#### **1.4.3.3. Minimum redundancy**

All data will be stored in different tables having minimum chances of redundancy.

#### **1.4.3.4. Facilitated data inputs**

Simple interfaces will be provided to the user to interact with the system

## **1.5. Data security and integrity**

The database for the application will be cloud-based because it offers the highest level of data protection and is very user-friendly.

## **1.6. Technical innovation**

A user can register on our system by entering his credentials. Immediately following his registration, this app will reply to him.

## **1.7. Existing System**

All current systems have a tonne of features, but in this system we give users a good platform to post reviews of societies they have already visited, and on this platform admin-approved reviews are displayed so users can learn more about the desired society. This is a better way for users to gather information and make her own decisions.

## Chapter 2

### 2. Software requirements specifications

#### 2.1. Purpose

A document that outlines the characteristics of a project, piece of software, or application is known as a software requirements specification (SRS). Simply said, an SRS document is a project documentation that must be created before a project or application is started. It helps to get a brief description about the product the manufactured product. The targeted audience is technical experts.

#### 2.2. Scope

The software product that we will be building is a Real Estate application, known as Property Talash. User has a chance to check a verified review about any society which will help them to choose which society is suitable according to their needs.

### 2.3. Proposed Solution Overview

#### 2.3.1. Modules of the proposed solution

Module of this system is as follows:

- Signup.
- Login.
- Post reviews.
- Admin's.
- Registration.
- Favorites.
- Search bar.
- UI.
- Real time.
- Authentication.
- Verify user account.
- Verify review before posting.

#### 2.3.2. Proposed Solution

- First, we will authenticate the user by checking his/her live location or by uploading CNIC/ Bill. After checking the user, we allowed the user to write his comments about their respective place/society. This is how we get genuine reviews and public get benefit from it.

- The highest rated societies on the relevant city search will be suggested by a recommendation system. A profanity filter will be added to the system to examine sensor strings for offensive words..

## 2.4. Specification requirements

### 2.4.1. Functional Requirement

- User must have valid email, phone number and password to create their account and to Login.
- If user want to save his/her searched society, they can add them to their favorites.
- The user can search for their desired society by selecting their city and society name from the dropdown menu in the search bar which shows them the society result or if they only select a city, all societies that are present in that city will be shown.
- The user can see verified reviews and rating of any society given by others that currently live in that society.
- If users want to give a review and rating, he/she must have to upload their CNIC/ Bill picture to verify their location before they give a review about any society and If the verification fails user will not allow giving a review.
- At the backend admin manage to authenticate user and also approve reviews given by authenticated user.
- If user only want to search society in any city they do it without creating account but if they want to perform further actions like user want to give review, add searched society to their favorites they must have to verify their account first.
- **Bottom navigation menu:** There is a bottom navigation menu from where user can navigate to 3 different pages i.e. Home, Favorite and Profile.
- **Home:** It helps to navigate back to Main Screen from where user started.
- **Favorite:** It helps user to check their marked societies.
- **Profile:** It helps user to update their profile information.
- **Side menu:** There is a side menu from where user navigate to 5 different pages:
  - i. **Home:** It helps to navigate back to Main Screen from where user started.
  - ii. **Favorite:** It helps user to check their marked societies.
  - iii. **Sign in/ Create Account:** If don't user have an account they to sign up first after this they will be able to login or if they an account already they just have to login to their account.
  - iv. **Contact us:** If user has any query they can contact us through our email or phone number.
  - v. **About us:** In this user get a brief introduction about us.

### 2.4.2. Non-Functional Requirement

- Secure access of confidential data (User's data).
- Efficient system design so that system will not get slow if many users will be active at same time.
- Flexible architecture so that any future extension will be easily manageable.

- Response time of our system is fast.
- User support will be available for 24/7.

## 2.5. Advantages

### 2.5.1. Advantages for the Users

- It can allow users to save their favorite societies.
- It can also provide real time data of the societies.
- It also provides authenticated reviews about the society which is given by the people that currently live in that society which help other users to make a good decision on behalf of rating and reviews of the society.

### 2.5.2. Advantages for the Collaborated Companies

- It can help companies to track and analyze user behavior, preferences and feedback that can help them to improve their services.
- It can also provide a platform for companies to advertise their properties and reach a larger audience.

## 2.6. Product functions

### 2.6.1. Functional requirements explanation

- **Sign up:** User must have to provide their valid email, phone number and password to create their account.
- **Login:** User can login by entering their email and password. In future, they can also directly login from their Facebook id, google account and apple account.
- **Search:** The user may use the search bar to find their preferred society by choosing their city and society name from the dropdown menu, which displays the society result, or by choosing only a city, which displays all societies that are located in that city.
- **Add Review & Rating:** The user has access to verified evaluations of every society provided by individuals who currently reside there. Before reviewing and rating any society, users must submit a photo of their CNIC or Bill to prove their identity and address. The user will be prevented from leaving a review if the verification fails.
- **Favorite:** Users can add their marked searches in favorite section.
- **Authentication:** At the backend admin manage to authenticate user and also approve reviews given by authenticated user.
- **Profile:** Users are free to modify or update the information they've already entered.

## 2.7. General requirements

### 2.7.1. Software requirements

- Database server: Cloud Firestore.
- Client: Any Android, IOS and Website users.
- Development tools: Android Studio, Flutter framework.

- Programming language: Dart.

### 2.7.2. Hardware requirements

- Processor: Intel Core i7
- RAM: 8 GB
- Hard Disk / SSD: 1 TB.

## 2.8. Use case diagrams

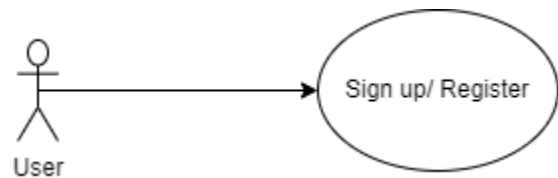
### 2.8.1 User end

**Register:**

**Email:**

xyz@gmail.com

**Phone number:** 03325665304



**Figure 2.8.1 (a): User**

**Password:** 12345678

**Confirm password:** 12345678

**Actors:** User

**Pre- condition:**

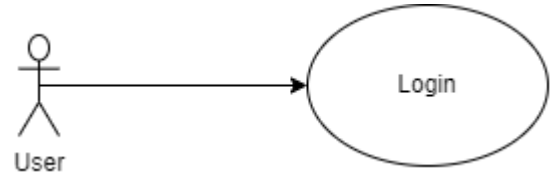
If user wants to use the services of the application, user must have to register their account by entering valid email, phone number, password to create their account.

**Post-condition:**

The user account will be validated, and the registration will be successful.

**Description:**

If a user wants to utilize the application's services, he or she must create an account by filling out the appropriate data, and after verification, the registration will be successful.

**Login****Email:** xyz@gmail.com**Password:** 12345678**Figure 2.8.1 (b): Login****Actors:** User**Pre-condition:**

After successfully creating an account, the user must provide a valid email address and password in order to use the application's services. Once these entries have been verified, the user is then able to access all of the application's features.

**Post-condition:**

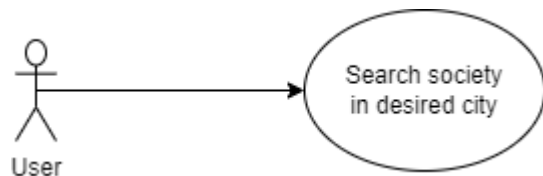
Once the verification process will complete, the user will be logged in.

**Description:**

In order to use the features offered by the application, the user must login using their email address and password. These fields are checked on the client side for potential validation problems when a user fills out the required forms and clicks the login button. The database will then be searched using the entered data to see if the user registered an account prior to logging in. If the submitted data successfully matches any entries in the database, the user's login session will begin.

**Search**

- User can do searches in two ways:

**Figure 2.8.1 (c): Search society in desired city**

- The city and society name must be chosen from the drop-down box if the user wants to limit their search to a certain society in the city they have chosen.
- User just has to choose city from the drop-down menu if they want to see all the societies that are relevant to their chosen city.

**Actors:** User**Pre-condition:**

User must check what their requirement is whether they want to get specific society result or all the society result in their targeted city.

After making decision user have to select city and society from drop down menu.

**Post-condition:**

The returned results are displayed once the user chooses the city and society from the drop-down menu.

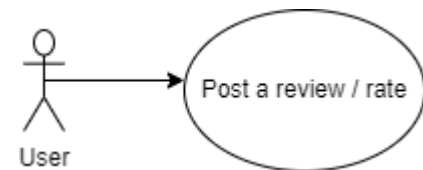
**Description:**

If a user wishes to restrict their search to a certain society in the city they have selected, they may choose the city and then the society name from the drop-down box. If a user wants to view all the relevant societies for their selected city, they need only select the city from the drop-down menu.

**Review and Rating:**

**Actors:** User

**Pre-condition:**



**Figure 2.8.1 (d): Post a review**

User must have to create account and login to their account. After this user search society in their targeted city and select society from drop-down menu. When society details will be shown, user selects the block / phase of that society. User will click on “Post a review textbox” to write review and give rating according to different aspects like security, cleanliness etc. After this user click on “Post” button.

**Post-condition:**

When user clicks on post button, user first has to verify their location and after successful verification, user review is posted publicly.

**Description:**

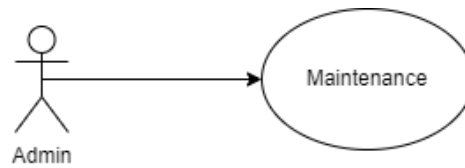
Users must first make an account and log in before they can evaluate or rate their society. If a person already has an account, they should log in to it. The user then chooses city and society. Users may then choose their block, click the "Post a review" textbox, and rate the results based on the many aspects of that society. User reviews are submitted publicly following successful location verification when user clicks the "Post" button.

**2.8.2 Admin end**

**Maintenance:**

Admin do the following:

- Checking system's working
- Database maintenance
- Environment menu



**Figure 2.8.2 (a): Maintenance**

**Actors:** Admin

**Pre-condition:**

Admin will check system's performance if any defect that will affect the system's performance.

**Post-condition:**

If any problem/ defect occur, admin will resolve the issue and check whole system including database work perfectly or not.

**Description:**

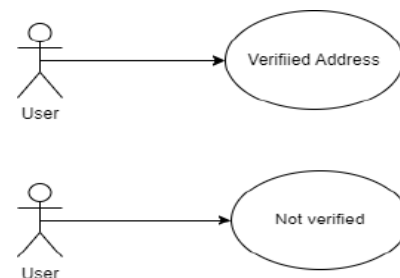
If there is a flaw that may impair the system's performance, the administrator will verify its performance. If a problem or error is found, the administrator will fix it and check to see if the entire system, including the database, is functioning properly.

**Authentication**

**Actors:** Admin

**Pre-condition:**

For verification user will have to upload picture of CNIC/Bill.



**Figure 2.8.2 (b): Verification**

**Post-condition:**

Admin will verify the address of user according to the society address if both matches then user will be verified.

**Description:**

The user will need to provide a photo of their CNIC or bill for verification. If the user's address and the society's address match, the administrator will verify the user.



## Chapter 3

## 3.1. Use case diagram

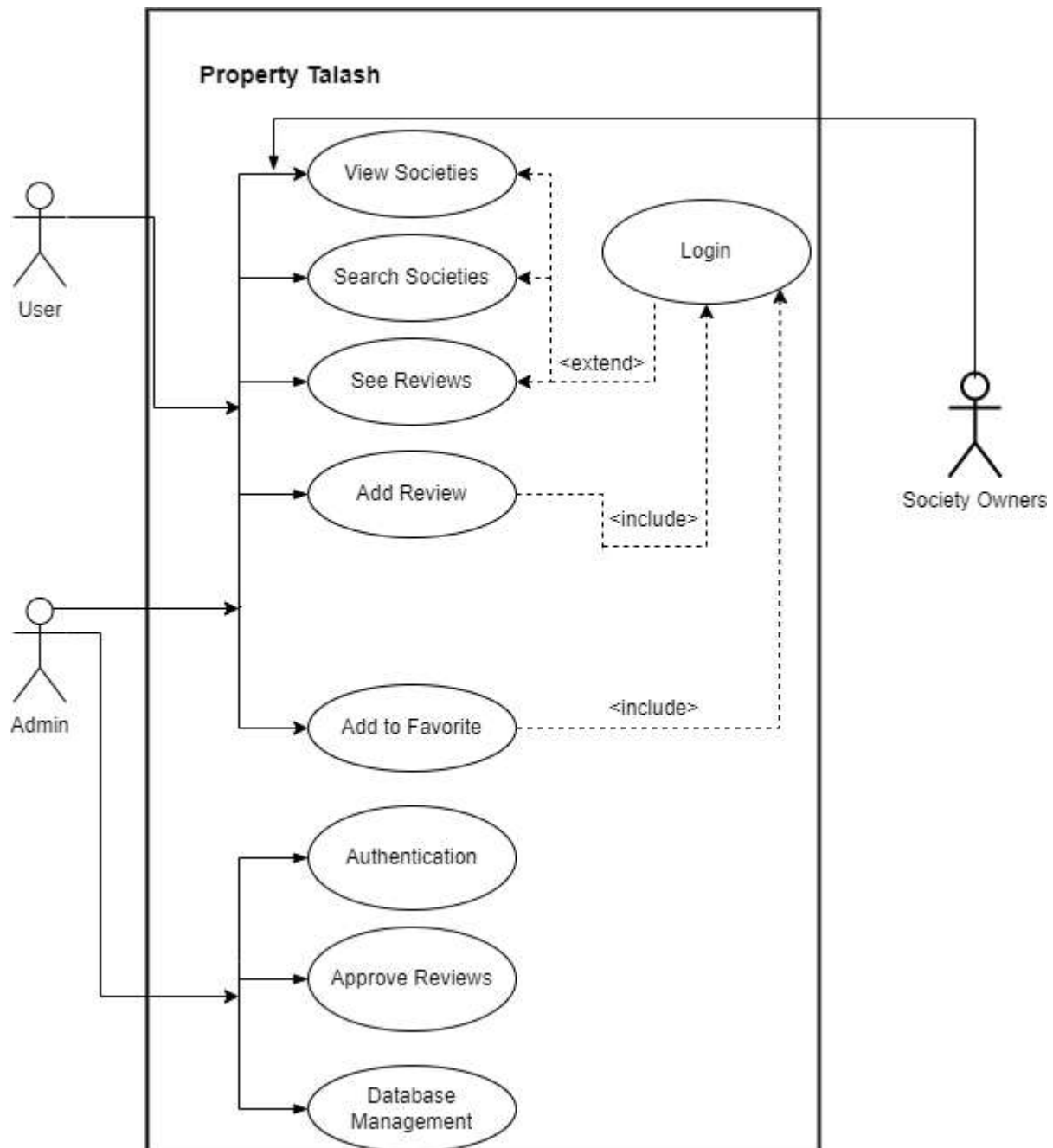


Figure 3.1: Use case diagram of whole system

### 3.2. Class diagram

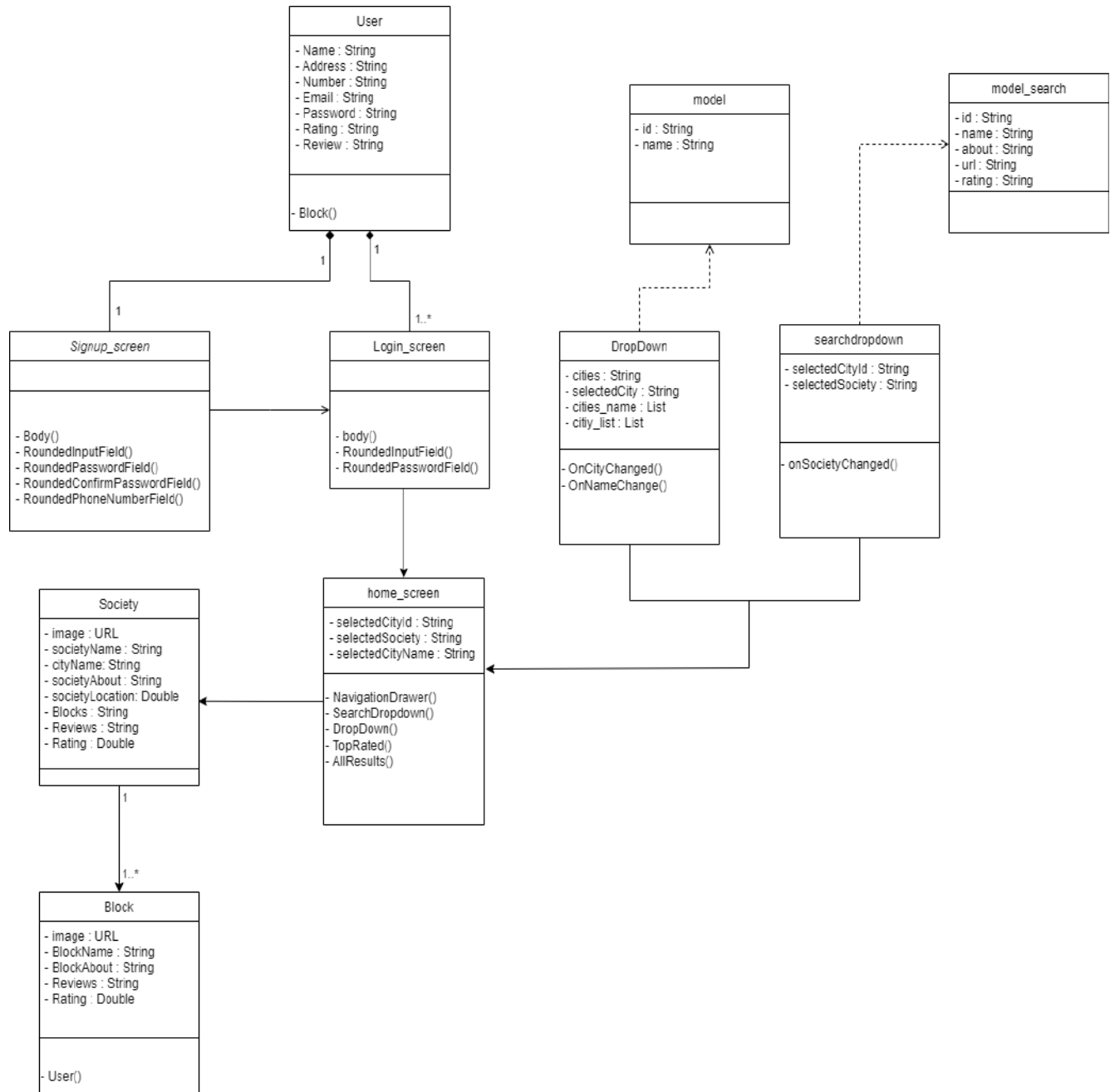


Figure 3.2: Class diagram of whole system

### 3.3. Activity diagram

#### 3.3.1. Login

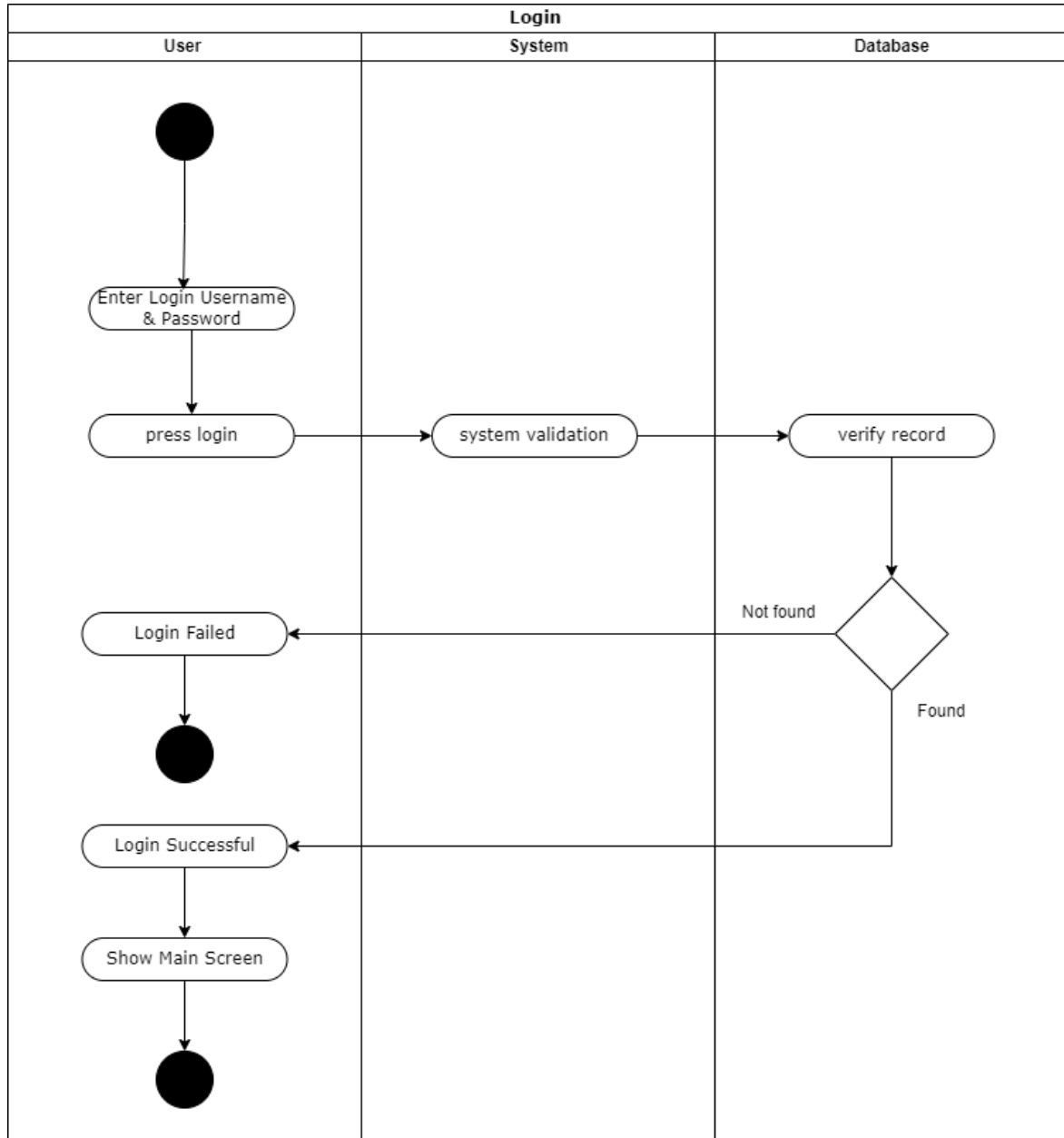


Figure 3.3.1: Activity diagram of login

### 3.3.2. Signup

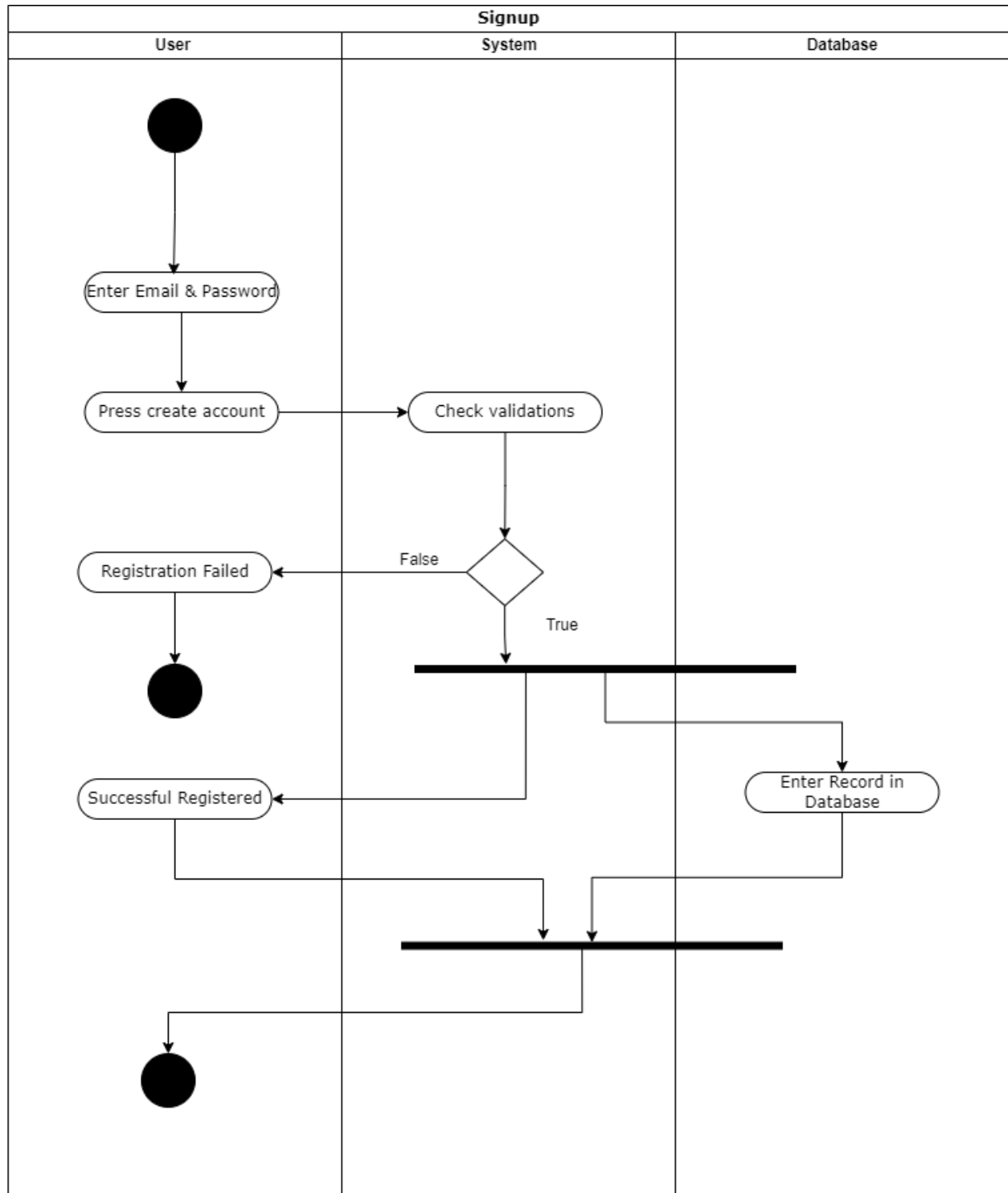


Figure 3.3.2: Activity diagram of sign up

### 3.3.3. Main screen

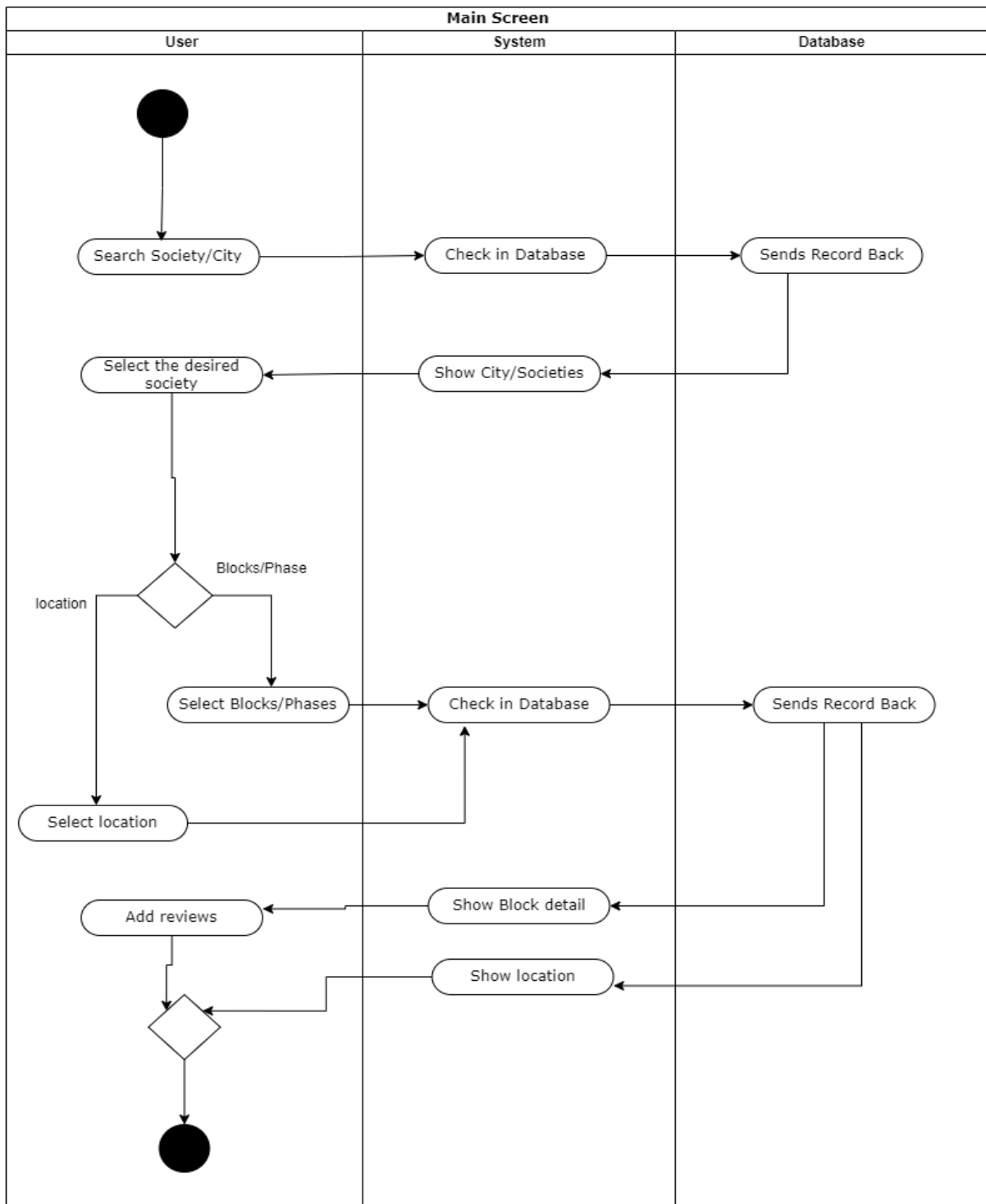


Figure 3.3.3: Activity diagram of main screen

### 3.4. Sequence diagram

#### 3.4.1. Login

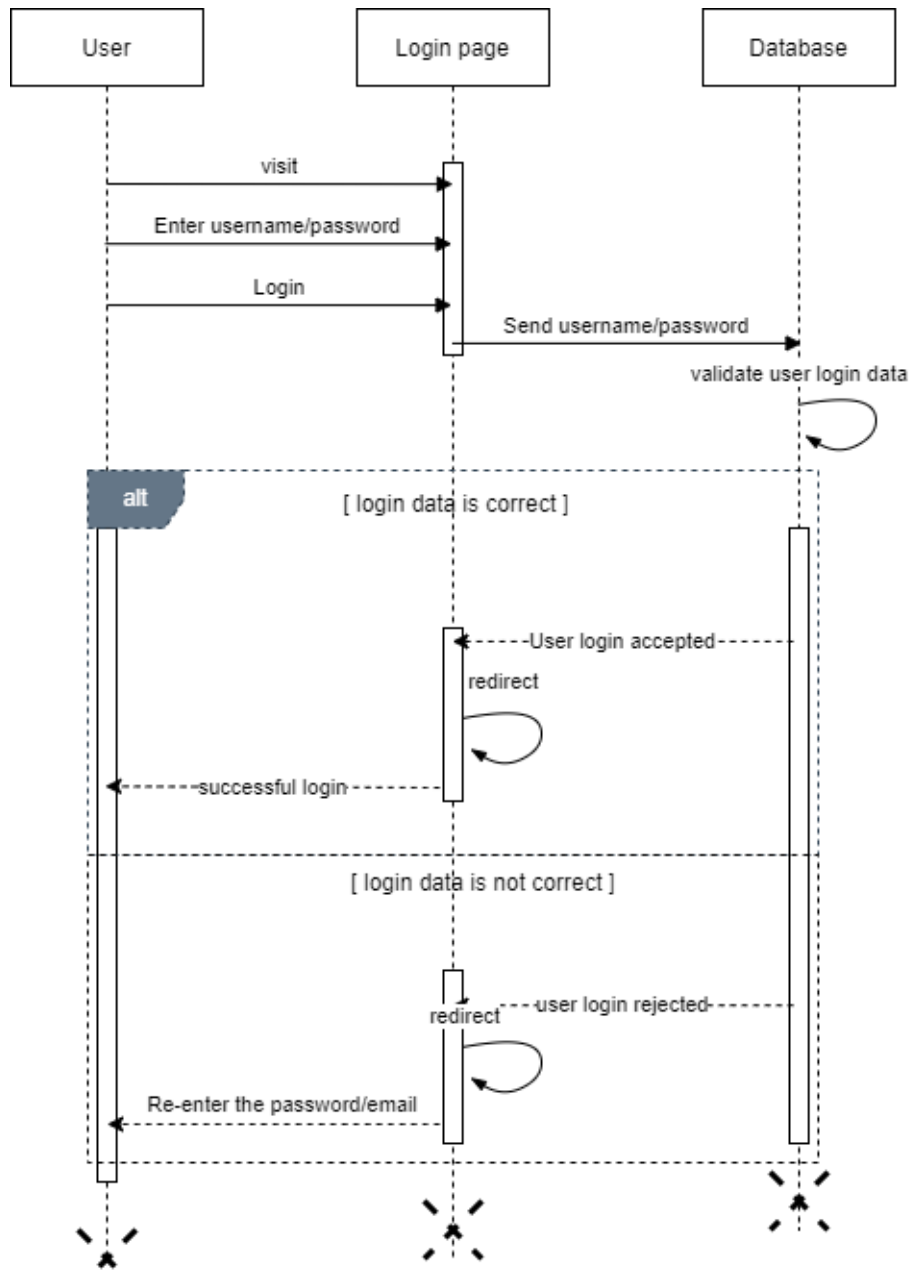


Figure 3.4.1: Sequence diagram of log in

### 3.4.2. Signup

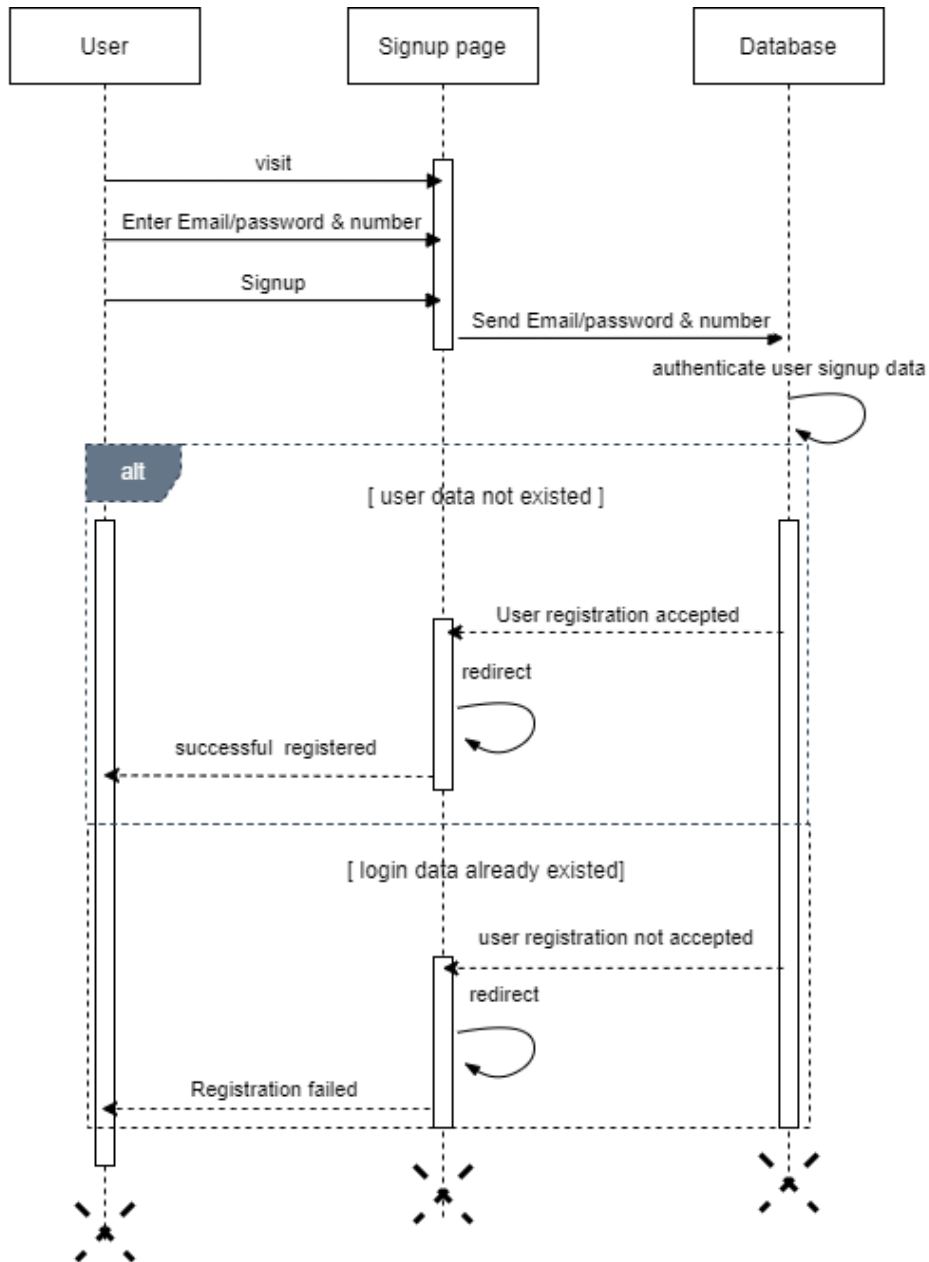


Figure 3.4.2: Sequence diagram of sign up

### 3.4.3. Main screen

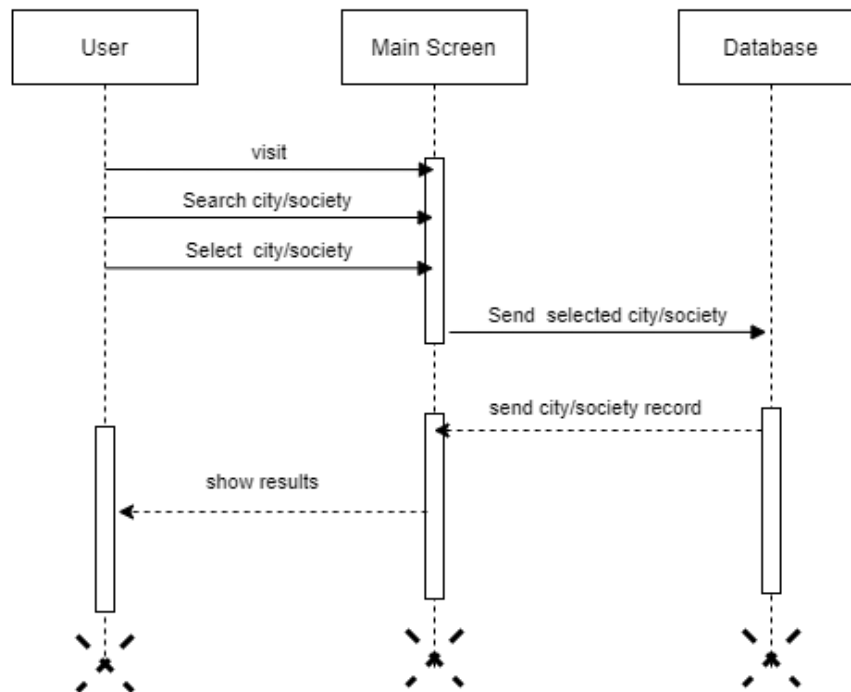


Figure 3.4.3: Sequence diagram of main screen



### 3.6. Data dictionary

#### 3.6.1. User

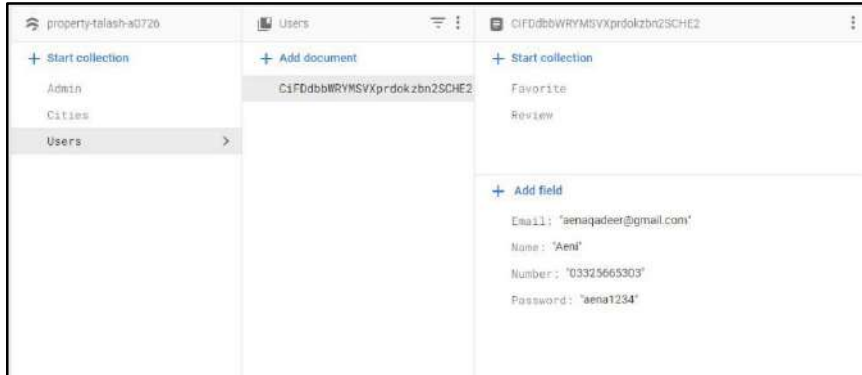


Figure 3.6.1: User

#### 3.6.2. City

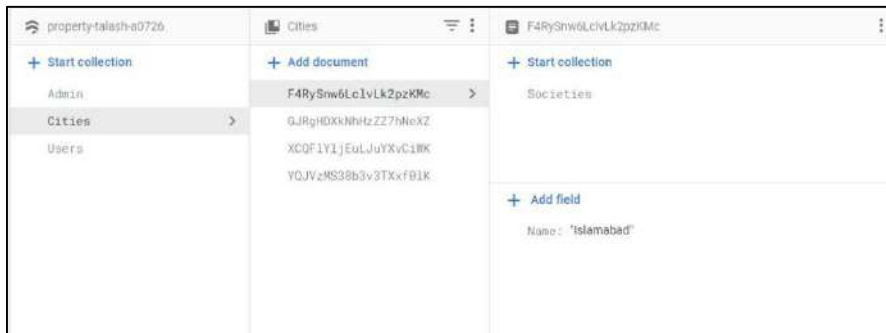


Figure 3.6.2: City

### 3.6.3. City has a collection name society

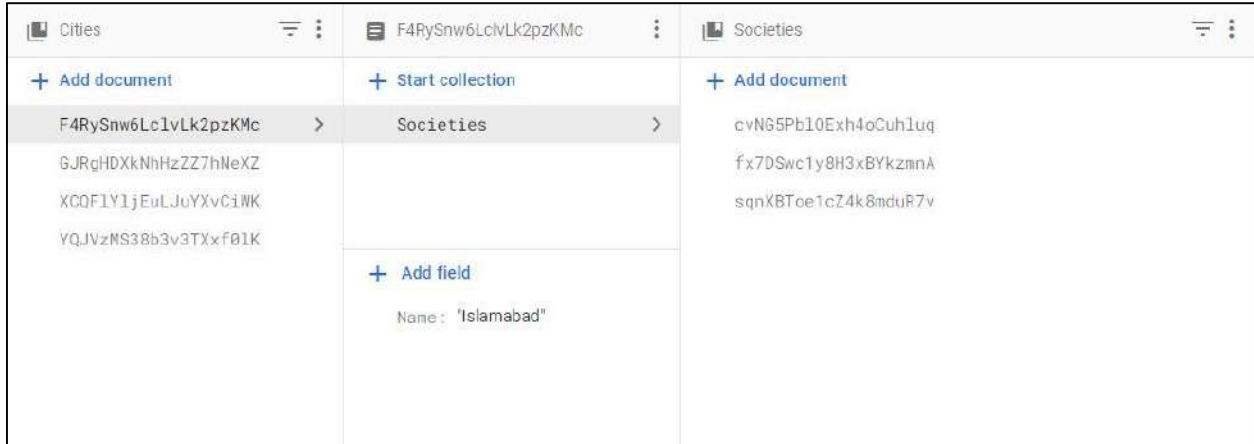


Figure 3.6.3 (a): City collection name

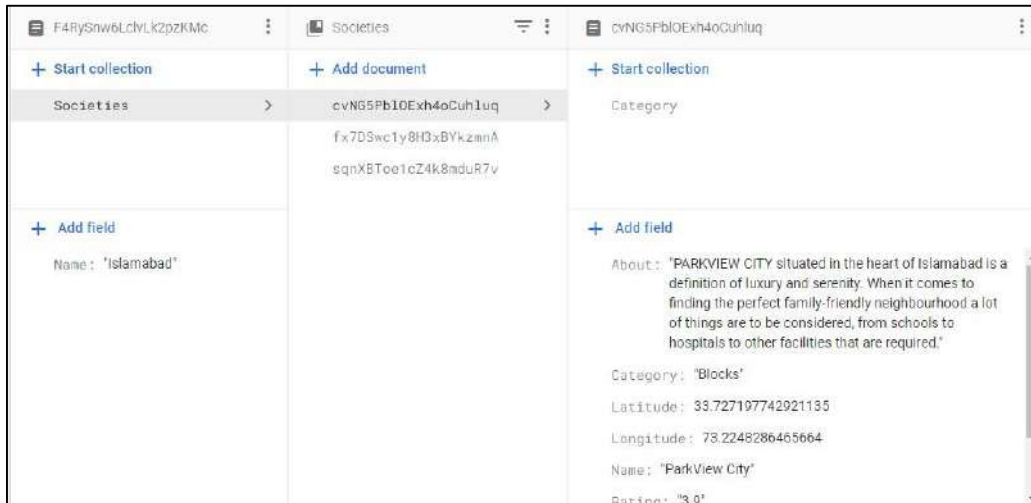


Figure 3.6.3 (b): Each society within city collection

### 3.6.4. Each society has a collection of category

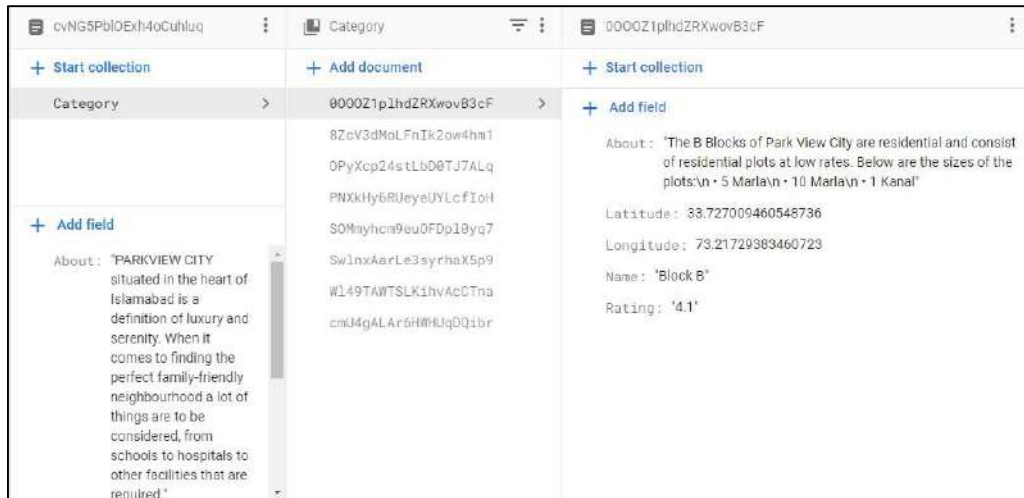


Figure 3.6.4 : Each society with collection name

### 3.6.5. Admin has two documents of pending and approved reviews

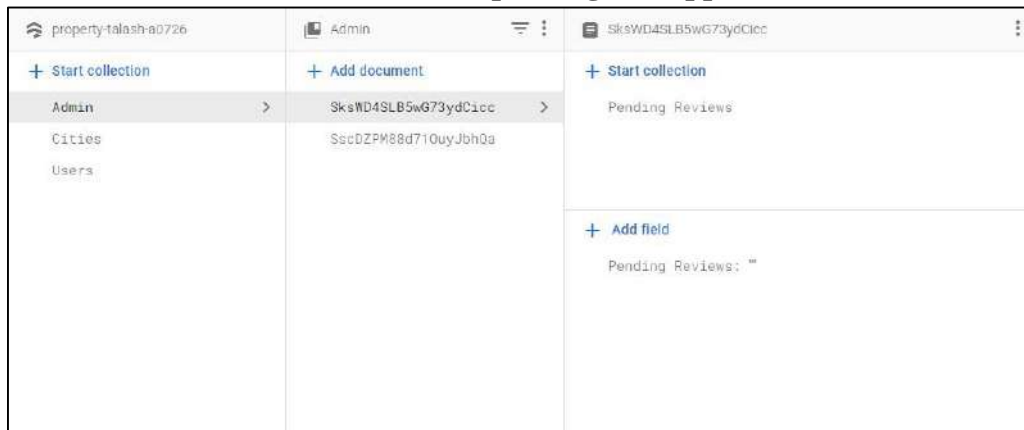


Figure 3.6.5: Admin

### 3.6.6. Pending reviews

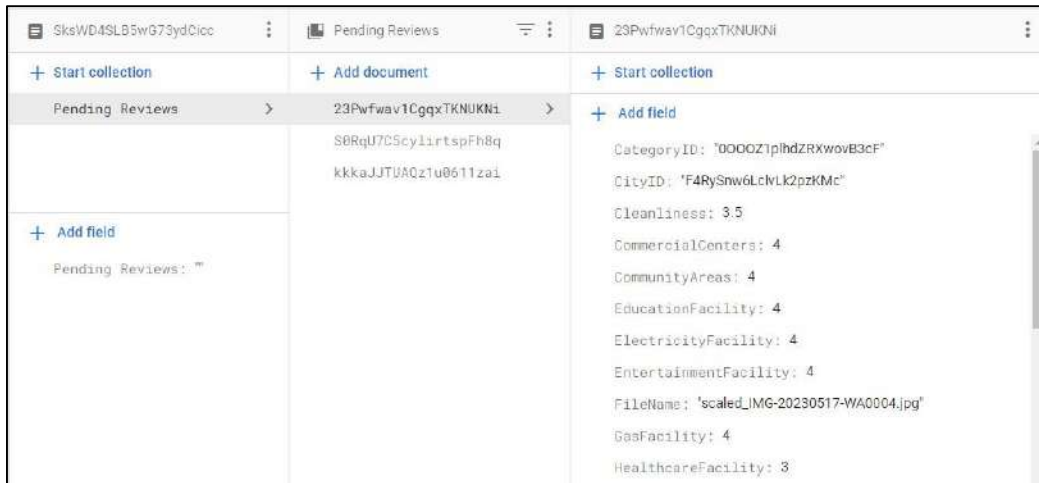


Figure 3.6.6: Pending reviews

### 3.6.7. Approved reviews

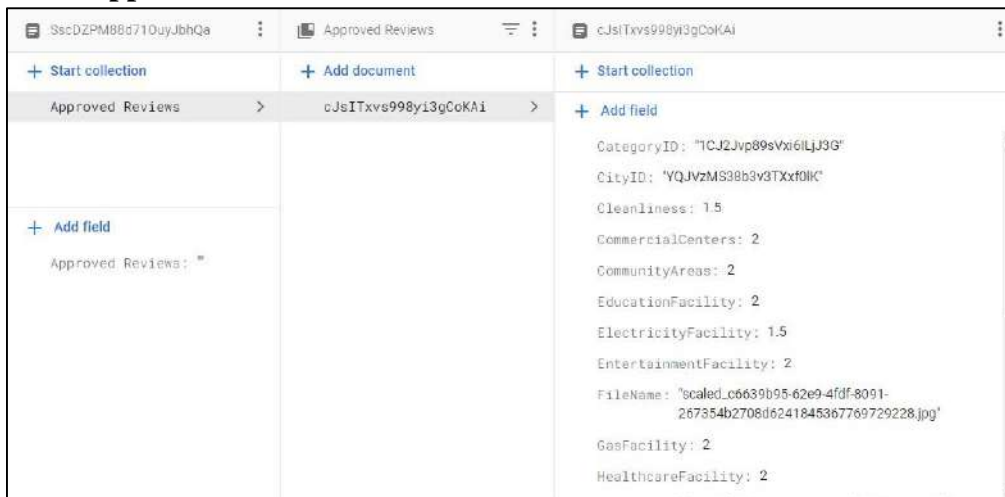


Figure 3.6.7: Approved reviews

### 3.6.8. This is storage where we have stored images

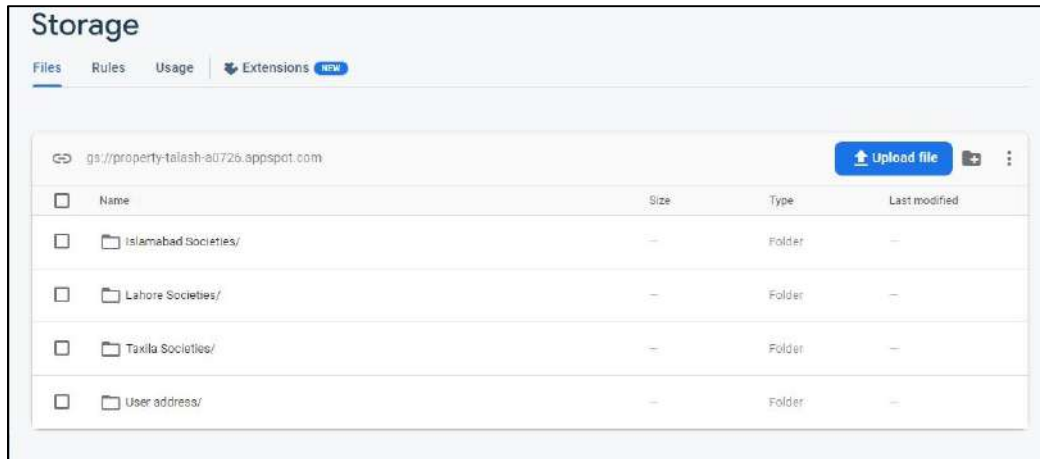


Figure 3.6.8: Storage

### 3.6.9. Authentication where authenticate users are shown

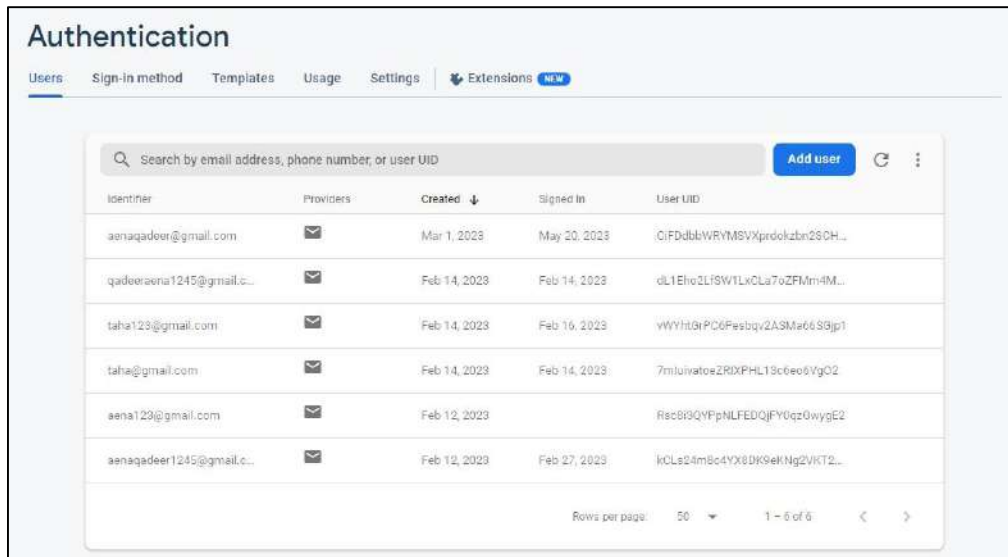


Figure 3.6.9: Authentication process

## Chapter 4

### 4. PROJECT MANAGEMENT

#### 4.1. Project deliverables

Here is the list of project deliverables:

Deliverables	Time
Proposal	1st Month
Required Specification	2nd Month
Design	3rd Month
Development & Implementation	4th 5th & 6th Month
Testing & Debugging	7th Month
Coded review	8th Month
Documentation	9th Month

**Table 4.1 (Project deliverable)**

#### 4.2. Risk Management

There are numerous considerations for security, maintainability, and usability while creating any software. Risk management is a crucial issue in this context. The goal of the risk management plan is to stop data loss and stop system crashes and abuse. To guard against insecure methods, necessary precautions have been made. At the time of user contact, the system offers user help in the form of procedures like email authentication and strong passwords for safe accounts. They also received caution warnings when they entered any incorrect credentials. The system will fully preserve their privacy, and strong security measures will be taken to guard against attacks on the system database.

### **4.3. Purpose**

Risk management's main goal is to protect systems against all types of security attacks, including hacking, sniffing and spoofing. By exercising caution, the user's accounts will remain protected. Making ensuring user privacy is protected is another goal of risk management. One of our key goals is to protect our system against SQL injection attacks, thus we have taken steps to make sure that no one can access the data on the back end of our system or the logic that powers it.

### **4.4. Risk management functions**

High security measures are taken in our system when it comes to posting reviews on social media, such as authenticated review emails and billing for user verification. There is a strict prohibition of special characters in reviews and emails because only authentic users can access their accounts and if a user forgets their password, they can do so by following the system setup link at the backend database is secured with multiple implementation processes so that system is secure.

## Chapter 5

### 5. IMPLEMENTATION

#### 5.1. Operating environment

Operating environment for the **Property Talash** is as listed below:

- Operating system: Windows
- Platform: Flutter
- Database: cloud Fire store
- Firebase database server

**Timescales:** Almost 6 to 8 months are required for the completion of this system.

**Testing:** Unit testing, integration testing and system testing will be performed

##### 5.1.1. Operating system

**Windows:** Easy to use and contains advanced features.

##### 5.1.2. Flutter

Flutter is Google's open-source UI toolkit for building native mobile, web, and desktop applications with a single codebase, featuring hot reload for fast development and a rich set of pre-built widgets for creating visually appealing interfaces.

###### 5.1.2.1. Flutter features

Flutter is an open-source UI toolkit developed by Google that enables cross-platform development for mobile, web, and desktop applications. It offers features like hot reload for fast development iterations, a reactive framework for automatic UI updates, a wide range of pre-built and customizable widgets, access to native features, high-performance rendering, and robust testing and debugging tools. With Flutter, developers can create visually appealing and native-looking applications on multiple platforms, saving time and resources while enjoying an active community and growing ecosystem.

###### 5.1.2.2. System requirements for flutter

To develop Flutter applications, you need a 64-bit operating system (Windows 7 SP1 or later, macOS, or Linux), at least 400 MB of free disk space, a 64-bit processor, 4 GB or preferably 8 GB of RAM, the Flutter SDK and Dart SDK installed, and an IDE like Visual Studio Code or Android Studio. Additional dependencies may be required for specific platforms.

##### 5.1.3. Firebase database server

**Database:** We'll use cloud fire store database.



#### **5.1.4. Dart**

We will use dart to developing this system.

##### **5.1.4.1. Dart features**

System requirements for flutter: Dart is a versatile and powerful programming language used by Flutter. It supports object-oriented programming with strong typing, offers JIT and AOT compilation for fast development cycles and optimized production performance, includes built-in support for asynchronous programming, manages memory automatically through garbage collection, allows code reuse with mixins, provides isolates for concurrent execution, has a rich set of libraries and packages, and offers excellent tooling and IDE support. Dart's features enable developers to write efficient, maintainable code and build high-performance applications with Flutter.

##### **5.1.4.2. System requirements for dart**

To develop applications using Dart programming language, you need a system with a supported operating system such as Windows, macOS, or Linux. The specific hardware requirements are relatively modest, typically a 64-bit processor, sufficient disk space for the Dart SDK and development tools, and a minimum of 4 GB of RAM. Additionally, you'll need to download and install the Dart SDK, which includes the Dart runtime and command-line tools. IDE support is available for popular editors like Visual Studio Code and IntelliJ IDEA, but it's not mandatory. Dart is designed to run efficiently on a variety of systems, making it accessible for developers with a range of hardware configurations.

## Chapter 6

### 6. Software Testing:

#### 6.1. Deriving test case specifications:

The specifics and requirements of each particular test case are described in test case specifications, often known as test case specifications documents or test case specifications sheets. These specs offer an organized method for preparing for and carrying out tests during software testing. The test strategy for our system's testing phase is described below.

#### 6.2. Testing Environment

##### 6.2.1. Hardware Requirements

- **Memory:** 8 GB
- **OS:** Windows 10
- **CPU:** Intel Core i7 64 bit

##### 6.2.2. Software Requirements

- Web browser
- Internet

##### 6.2.3. Testing Identifications

Every module will be tested in a certain manner that we have designed. After creating a list of all the modules that needed testing, we created individual test cases for each module. The following are some of the main test cases:

#### 1) User

- Signup module
- Login module
- User account module
  - a) Update the user information module
- Search module
- Submit review module
- Add to favorite
- Sign out module

## 2) Admin

- Add society record
- Delete society record
- Update society record
- Pending review
- Approved review

### 6.3. Testing procedure

A way for informing us that we will test each test case is the test procedure. Additionally, we keep track of the length of time it takes to test each module and the impact each test case has on each module.

### 6.4. Test Cases

#### 6.4.1. Test Case for Sign up module

##### 6.4.1.1. Sign up as a User

<b>Tested By</b>	<b>Aena Qadeer</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	01
<b>Test case Name</b>	Sign up testing
<b>Test case description</b>	This test case examines the user sign-up procedure to confirm that users can successfully set up accounts in the system. This test is designed to confirm that the sign-up process works as expected and that creating a user account is simple and error-free.
<b>Item to be tested</b>	
	Fill in the necessary boxes with accurate data, including your email address, password, and phone number.
	Confirm the password by re-entering it in the appropriate field.

	Click on the "Sign Up" button to initiate the account creation process. The system will now allow the user to log in.
<b>Specification Input</b>	Valid user data, including a special email address that isn't already linked to any other accounts, password, and phone number.
<b>Expected Result</b>	<p>The user sign-up procedure is the main emphasis of the test case, which verifies that the user may successfully establish an account without running into any problems or warning messages. It checks to see if the data given in the required fields are accepted appropriately and if the confirmation password is a match for the password that was submitted for successful password confirmation. Furthermore, the user's optional information should be accepted without any problems. The account creation process should begin immediately upon clicking the "Sign Up" button, without any delays or mistakes. After signing up, the user should be effortlessly transferred to the proper page without experiencing any difficulties or unusual behaviors.</p> <p>Finally, when using the newly established account credentials to log in, the login procedure should be successful and error-free, demonstrating that the account creation process was carried out as intended.</p>
<b>Actual Output</b>	When a user enters information in a field that is needed, that information is accepted as long as the confirmation password matches the password that was entered. Additionally, the optional data provided by the user need to be accepted without any issues. After clicking the "Sign Up" button, the process of creating an account should start instantly and without any errors. The user is seamlessly transported to the appropriate page after

	<p>signing up without encountering any issues or strange behavior.</p> <p>Finally, the login process is successful and error-free when using the newly created account credentials. This will show that the account creation process was completed as planned.</p>
--	--

Table 6.4.1.1 ( Sign up testing )

## 6.4.2. Test Case for Login Module

### 6.4.2.1. Log in as a User

<b>Tested By</b>	<b>Aena Qadeer</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	02
<b>Test case Name</b>	Login testing
<b>Test case description</b>	This test case examines the user login procedure and confirms that a user can access their account. This test is used to make sure that the login process works as intended and that users can safely access their accounts.
<b>Item to be tested</b>	
	Navigate to the application's login page.
	To access the user's account, provide the right email address and password
	To begin the login procedure, click the "Login" button.

	Verify that the user is successfully logged in and directed to the home page.
<b>Specification Input</b>	Valid user data, including email and password associated with the user's account.
<b>Expected Result</b>	<p>There shouldn't be any problems guiding the visitor to the login page. The user's account's connected username or email address should be approved without any mistakes.</p> <p>The user's account should accept the proper password without any issues.</p> <p>The login procedure should start immediately after clicking the "Login" button, without any glitches or delays.</p> <p>The user should be led to the correct page without any problems or unusual behavior following a successful login.</p> <p>Without running into any permission problems, the user should be able to carry out various tasks and travel between various pages within the application.</p>
<b>Actual Output</b>	Accessing the login page successfully, accurately entering the username or email address and password, starting the login process quickly after clicking the "Login" button, and receiving a redirect to the proper page following a successful login, the User is also able to move between different application pages and carry out a variety of operations without running into authorization problems.

Table 6.4.2.1 ( Login testing )

### 6.4.3. Test Case for User account module

#### 6.4.3.1. Update user information

<b>Tested By</b>	<b>Aena Qadeer</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	03
<b>Test case Name</b>	User information update testing
<b>Test case description</b>	To make sure that the changes are correctly stored and reflected in the system, this test case focuses on changing the user's account information. The goal of this test is to ensure that the account update capability works as anticipated and enables users to successfully edit their information.
<b>Item to be tested</b>	
	Go to the user profile or account settings page.
	Find the area or fields where user data, such as name, email, and phone may be updated.
	Change one or more user information fields with the needed and appropriate adjustments.
	To make the changes effective, click on the "Update" button.
	Check to see if the modifications were properly saved without any issues or warnings.
	Verify that the user account settings or profile page accurately displays the changed user information.

<b>Specification Input</b>	Changes that are legitimate and wanted are made to user information fields like name, email, phone, and address.
<b>Expected Result</b>	<p>The profile page or account settings should be easily accessible to the user.</p> <p>The user information fields must be editable and open to changes without any limitations or problems.</p> <p>Valid modifications to the user information fields should be accepted without any warnings or problems.</p> <p>The changes should be immediately applied after clicking the "Update" button, without any glitches or delays.</p> <p>The user information should be successfully stored whenever any modifications are made without any problems.</p> <p>The changed user information should appropriately appear on the user account settings or profile page, indicating that the update was successful.</p> <p>After the update, all pertinent parts and features in the program should appropriately use the updated user data.</p>
<b>Actual Output</b>	Users may modify user information fields without any limits or issues and can make legitimate changes to the needed fields without running into any problems or validation messages, after successfully navigating to the user account settings or profile page. The changes need to take effect right away after pressing the "Update" button, without any hiccups or mistakes. The database was also updated. The real result would be regarded as accurate if the user information changes were successfully stored



	without any problems or mistakes. The user account settings or profile page ought appropriately to reflect the modified data, indicating that the update was successful.
--	--

**Table 6.4.3.1 ( User information update testing )**

#### 6.4.4. Test Case for Search Module

<b>Tested By</b>	<b>Aena Qadeer</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	04
<b>Test case Name</b>	Search module testing
<b>Test case description</b>	This test case focuses on the functionality of the search module, particularly the search for cities and the societies they are affiliated with. The goal of this test is to ensure that the search module effectively locates cities and societies by properly retrieving and displaying pertinent results depending on user input.
<b>Item to be tested</b>	
	Find the search module.
	Select the city name in the search field.
	After selecting the city name, select the society
	Check to see if the search results provide pertinent information that matches the search term you typed.
<b>Specification Input</b>	Suitable city names for searches.  Valid society names or search terms for the city you've chosen.

<b>Expected Result</b>	<p>The search module needs to be easily accessible to the user.</p> <p>The search box should allow a legitimate city name selection without any issues.</p> <p>The search area should accept your choice of a legitimate society name without any problems.</p> <p>Results have to be shown on the main screen by the supplied city and its society.</p>
<b>Actual Output</b>	<p>The search field needs to work properly, enabling the user to type a valid city name without running into any problems or errors. The input is acknowledged and correctly processed.</p> <p>The user's selection of an authorized society name also is accepted by the search area without any issues. The information is accepted and correctly processed for the user to search for certain societies in the chosen city.</p> <p>The search results, which provide pertinent data depending on the provided city and its society, are shown on the home screen. The outcomes must precisely correspond to the user's search criteria and show the related cities and societies as necessary.</p>

**Table 6.4.4 ( Search module testing )**

### 6.4.5 Test Case for Submit review module

<b>Tested By</b>	<b>Aena Qadeer</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	05
<b>Test case Name</b>	Submit review module testing
<b>Test case description</b>	This test case focuses on the Submit Review module, in particular the features that let users post reviews, score societies using specified criteria, and upload images of their CNICs or utility bills to verify their addresses. This test aims to confirm that users are capable of correctly submitting reviews, rating societies fairly, and supplying the required supporting evidence for address verification.
<b>Item to be tested</b>	
	Access the application's Submit Review section.
	Find the review textbox where people may enter their reviews.
	In the text box, write a review on the society in which you share your opinions.
	Make that the review has been accepted without any hiccups or limitations.
	Look for the part on rating criteria, where people may assess the society using the criteria supplied (such as facilities, security, and cleanliness).
	By choosing suitable values for each parameter, rate the society.

	Verify sure there are no faults or limits on the rating values.
	Use the given file upload feature to locate the file upload section where customers can upload a photo of their CNIC or bill for address verification.
	Check to see whether the file upload went through without any problems or errors.
<b>Specification Input</b>	<p>Review content that is appropriate for expressing opinions about society.</p> <p>Amounts that are appropriate for each aspect based on the characteristics supplied (for example, cleanliness, security, and amenities).</p> <p>A true photocopy of the user's CNIC or utility bill for address confirmation.</p>
<b>Expected Result</b>	<p>The user ought to have no trouble using the Submit Review module.</p> <p>Users should be able to input their reviews into the review textbox without running into any problems or limitations.</p> <p>The entered review must be acknowledged and stored appropriately, without any changes or data loss.</p> <p>Users should be able to rate the society using the specified parameters in the rating parameter section without any problems or limitations.</p> <p>The chosen rating values have to be accepted and saved appropriately, without any changes or data loss.</p> <p>Users should be able to upload a photo of their CNIC or bill for address verification</p>

	<p>without any problems or limitations using the file upload area.</p> <p>Click on the “Submit” button to upload the review to the database without any error.</p> <p>Without any alterations or information loss, the uploaded photo must be precisely kept and linked to the particular review.</p>
<b>Actual Output</b>	<p>The user can input their review without running into any problems, and the review is stored correctly. The rating feature operates as intended, accepting and accurately storing the chosen rating values. The user has no trouble uploading their CNIC or bill photo using the file upload tool, assuring successful saving. By clicking on the “Submit” button, upload the review to the database without any error.</p> <p>All contributed information, including reviews, ratings, and uploaded images, must appropriately reflect the relevant society. Any variations from these anticipated results, such as submission errors or inaccurate data association, would point to potential flaws or problems with the Submit Review module.</p>

Table 6.4.5 ( Submit review module testing )

#### 6.4.6 Test Case for Add to Favorite Module

<b>Tested By</b>	<b>Aena Qadeer</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	06
<b>Test case Name</b>	Add to favorite module testing
<b>Test case description</b>	The Add to Favorites module, in particular the feature that enables users to add societies to

	their favorite area, is the subject of this test case. By selecting the Add to Favorites icon, users may store societies for later access or reference. The goal of this test is to confirm that users can successfully add societies to their favorites in this manner.
<b>Item to be tested</b>	
	Use the application to access the desired society.
	Look for the society-related Add to Favorites symbol.
	To add the society to the user's favorites, click the Add to Favorite icon.
	Check to see whether the Add to Favorites symbol changes to show that the society has been successfully added.
	Navigate to the user's favorite section within the application.
	Check to see if the society you added in Step 3 appears in the user's favorites list.
	By selecting the symbol once again, the user can delete society from their favorite list.
<b>Specification Input</b>	Societies inside the application that users may bookmark.  Access to the Add to Favorites module is valid for this user account.
<b>Expected Result</b>	The user should have no trouble entering the targeted society.  The user should be able to click the Add to Favorites symbol if it is prominently placed and visible.

	<p>The society should be added to the user's favorites after clicking the Add to Favorites icon.</p> <p>The Add to Favorites icon should modify its look to visibly indicate that the society has been added.</p> <p>It should be possible to access the user's preferred part without experiencing any problems.</p> <p>The user's favorites section ought to appropriately reflect the society that was added in Step 3.</p> <p>Users should be able to add and see various societies in their favorites area thanks to the Add to Favorites capability, which ought to behave consistently for a variety of societies.</p>
<b>Actual Output</b>	<p>The society is added to the user's favorites. To show that society has been added, the symbol visually changes. The added society has appropriately shown when accessing the user's favorite area. Users can add and see different societies in their favorites area since the functionality ought to perform consistently for a variety of civilizations.</p>

**Table 6.4.6 ( Add to favorite module testing )**

#### **6.4.7 Test Case for Sign out module**

<b>Tested By</b>	<b>Zain Shabbir</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	07
<b>Test case Name</b>	Sign out testing

<b>Test case description</b>	In particular, the capability that enables users to log out of their accounts is the subject of this test case, which concentrates on the Sign Out module. This test checks to make sure users can log out of the application safely and successfully.
<b>Item to be tested</b>	
	Go to the side menu and click on the “Logout” Button.
	Verify that the user has been successfully logged out and is being sent to the login or home screen of the program.
<b>Specification Input</b>	Reliable user account information.
<b>Expected Result</b>	<p>Go to the side menu and click on the “Logout” button.</p> <p>The logout procedure should begin when the user clicks on the “Logout” button.</p> <p>Denying them access to sites and functionality that need authentication.</p> <p>After a successful sign-out, the user should be routed to the home screen.</p> <p>Access should be denied if the user tries to access any authenticated or restricted pages or features.</p> <p>When attempting to access restricted areas, the user should be required to log in once more, signifying a successful logout.</p>
<b>Actual Output</b>	The user is logging out safely and blocking access to restricted areas.

Table 6.4.7 ( Sign out testing )



### 6.4.8 Test Case for Add society record

<b>Tested By</b>	<b>Zain Shabbir</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	08
<b>Test case Name</b>	Add society record module testing
<b>Test case description</b>	The feature that enables an admin user to add a new society record to the system is the main subject of this test case. This test is intended to confirm that the admin user can successfully add a society record with valid information, ensuring that it is correctly preserved in the system for later usage.
<b>Item to be tested</b>	
	Within the admin panel or dashboard, use the Add Society Record feature.
	Include all necessary facts for the new society record, including the name of the society, its location, and any other pertinent information.
	Make sure the mandatory fields are correctly verified and do not permit any empty or incorrect inputs.
	To add the society entry to the database, submit the form.
	Make that the system correctly records the new society record, maintaining the supplied information without any changes.
	Verify that the system's access to and presentation of the newly added society record, whether in a society listing or search feature, are proper.

<b>Specification Input</b>	The name, address, phone number, and any other pertinent information are valid for the new society record.
<b>Expected Result</b>	<p>The admin interface should make it simple to use the Add Society Record capability.</p> <p>All essential fields should be included on the form for adding a new society entry, and they should be properly identified and labeled.</p> <p>Validation of the required fields is necessary to make sure that incomplete or incorrect entries are not permitted.</p> <p>The new society record should be added to the database or storage as soon as the form is submitted.</p> <p>The system must accurately store the new society record, maintaining the supplied data without any changes or data loss.</p> <p>The newly added society record must be viewable and properly presented inside the system to be used in appropriate contexts, such as society listings or search functionality.</p>
<b>Actual Output</b>	After admin successfully adds and stores the new society record, appropriately preserving and displaying the updated record inside the system, and adding and storing the new society record successfully.

**Table 6.4.8 ( Add society record module testing )**

#### **6.4.9. Test Case for Delete society record**

<b>Tested By</b>	<b>Zain Shabbir</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	09

<b>Test case Name</b>	Delete society record module testing
<b>Test case description</b>	The feature that enables an admin user to remove a society record from the database is the main subject of this test case. This test is designed to make sure the admin user can successfully delete a society record, ensuring that it is deleted from the database and is no longer visible.
<b>Item to be tested</b>	
	Within the admin panel or dashboard, use the Delete Society Record feature.
	Pick the society record you want to remove from a list or use the search tool.
	Check to see that the selected society record is accurately presented, including relevant facts such as the society name, etc.
	By clicking the "Delete" button, confirm the deletion action.
	Verify the system deletes the chosen society's record from the database or storage.
<b>Specification Input</b>	Current society record(s) in the system are available for deletion.
<b>Expected Result</b>	<p>Within the admin panel, it should be simple to access the Delete Society Record capability.</p> <p>To ensure precise identification of the record, the user should be able to choose the society record to be destroyed from the search feature.</p> <p>The chosen society record should appear appropriately, including all necessary data for verification.</p>

	<p>The system should delete the chosen society record from the database or storage when clicking on the “Delete” button.</p> <p>The system should correctly delete the chosen society record, ensuring it can't be accessed or seen any longer.</p> <p>It is important to appropriately manage and erase all relevant information or references connected to the deleted society record from the system.</p>
<b>Actual Output</b>	After the admin deletes the record, the system successfully deletes the chosen society records from the database. That deleted record is no longer viewable or accessible inside the system.

**Table 6.4.9 ( Delete society record module testing )**

#### 6.4.10. Test Case for Pending Review

<b>Tested By</b>	<b>Zain Shabbir</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	11
<b>Test case Name</b>	Pending review module testing
<b>Test case description</b>	The capability that enables an admin user to accept pending reviews provided by users is the main topic of this test case. This test aims to confirm that the admin user can successfully examine and approve pending reviews, ensuring that they are authorized and made visible to users in the review section.

<b>Item to be tested</b>	
	Check out the list of reviews that need to be approved.
	Make that the appropriate elements, such as the review content, rating, and user information, are presented alongside the pending reviews.
	Check each pending review's content to see if it complies with the rules and regulations.
	Click on the "Approved" button to approve the pending reviews that satisfy the requirements and move that reviews to the approved review section in the database.
<b>Specification Input</b>	Pending reviews waiting for approval in the system.
<b>Expected Result</b>	<p>A list of reviews that are pending approval should be available to the admin user.</p> <p>The substance of the reviews, their ratings, and user information should all be shown alongside the pending reviews.</p> <p>To guarantee adherence to rules and regulations, the admin user should evaluate the content of each pending review.</p> <p>By choosing the proper approval option, the admin user should be able to approve the pending reviews that satisfy the requirements.</p> <p>Users should be able to read and access the authorized reviews by seeing them posted in the review area.</p>
<b>Actual Output</b>	The system should correctly save the approval status for each review and indicate it as authorized when the outstanding reviews have been approved. Users can read

	and access the authorized reviews by seeing them posted in the review area. The authorized reviews appropriately show the review text, and rating to ensure that they are accurately portrayed.
--	---

**Table 6.4.10 ( Pending review module testing )**

#### 6.4.11. Test Case for Approved Review

<b>Tested By</b>	<b>Zain Shabbir</b>
<b>Tested Type</b>	Unit Testing
<b>Test case No.</b>	12
<b>Test case Name</b>	Approved review module testing
<b>Test case description</b>	This test case focuses on the feature that allows users to browse authorized reviews in the review area. The goal of this test is to ensure users can access and read the authorized reviews by confirming the system accurately identifies and displays them.
<b>Item to be tested</b>	
	View the list of reviews that are displayed in the Review Section.
	Make sure that only authorized reviews should be shown, excluding pending and rejected reviews.
<b>Specification Input</b>	Approved reviews are available in the system.
<b>Expected Result</b>	It should be easy to find and visible, with the Review Section shown clearly.  Only accepted reviews, not any that are pending or denied, should be included in the list of reviews that are presented in the review section.

	The accepted reviews must include all pertinent data, including review text, and a rating.
<b>Actual Output</b>	Users can read the authorized reviews once they reach the Review Section, with the system appropriately displaying them. Any pending or rejected reviews are not included in the list; only authorized reviews are included. The essential elements, such as review content, and rating is included in the authorized reviews to ensure they are appropriately represented.

**Table 6.4.11 ( Approved review module testing )**

## Chapter 7

### 7. Project Display Screens

#### 7.1. Mobile Application

##### 7.1.1. Splash screen

This is the splash screen of our application when user opens the application first this splash screen will be displayed



Figure 7 .1.1: Splash Screen

##### 7.1.2. Main screen

This is the main screen of our application when application is opened this screen will be displayed after splash screen where all societies will be visible to user.





Figure 7.1. 2: Main screen

### 7.1.3. Favourite

This is the favourite screen where user can see there saved societies which they have marked as favourite.



Figure 7 .1.3: Favourite

#### 7.1.4. Profile setting

This is the profile settings screen where users can see their details and can update their details if required and can delete their account.



**Figure 7.1.4: Profile setting**

#### 7.1.5. Society information

When a user selects any society on the Main screen, this screen will be displayed containing society details. All the society information with location and reviews will be displayed on this screen.

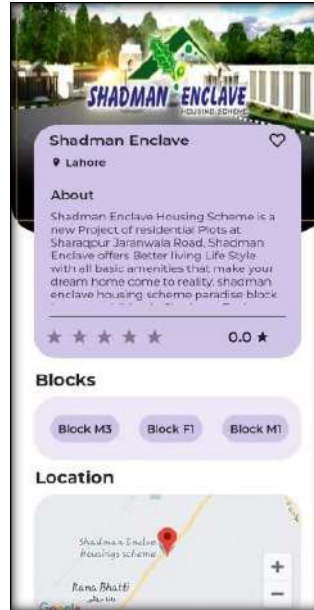


Figure 7.1.5: Society information

### 7.1.6. Add review

This is the add review screen where users can fill the desired ratings and submit their reviews about their society.

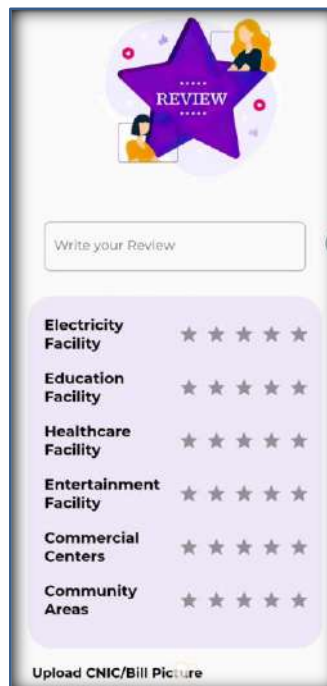


Figure 7.1.6: Add review

### 7.1.7. Search city

This is the search bar which is on the Main screen where user can select the desired city from the list of cities displayed in drop-down.



Figure 7.1.7: Search city

### 7.1.8. Search society

After selecting any city user have to select the desired society in that city which user wants to see and it will lead user to the selected society details.



Figure 7.1.8: Search society

### 7.1.9. Side menu bar

This is the side menu bar of application where some options are given like Home screen button, Login/Logout buttons, Contact us page and About us page user can click and go to desired page.

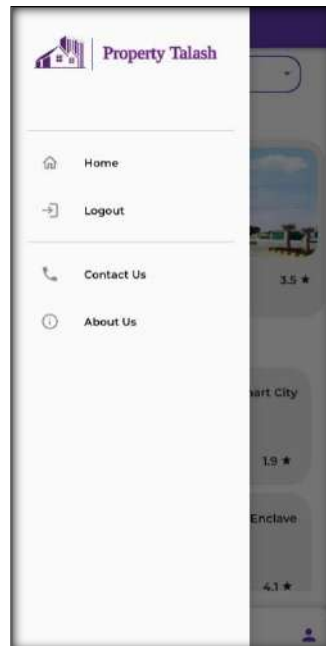


Figure 7.1.9: Side menu bar

### 7.1.10. Login

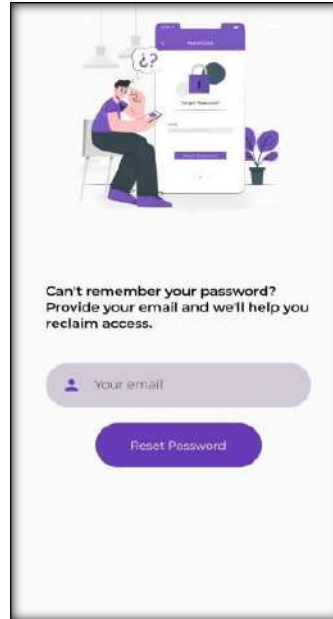
This is the Login screen where user can enter there login details and login to there accounts.



**Figure 7.1.10: Login**

### 7.1.11. Reset password

This is the Reset password screen if user forgot there passwords they can reset by entering there registered email .



**Figure 7.1.11: Reset password**

### 7.1.12. Sign up

This is the Sign up screen user can create there accounts by filling up the desired data required.



**Figure 7.1.12: Sign up**

## 7.2. Admin Panel

### 7.2.1. Dashboard

This is Dashboard the main screen of admin panel where some main details are showing like Total users, Pending reviews, Approved reviews, Total cities and societies add in this application.

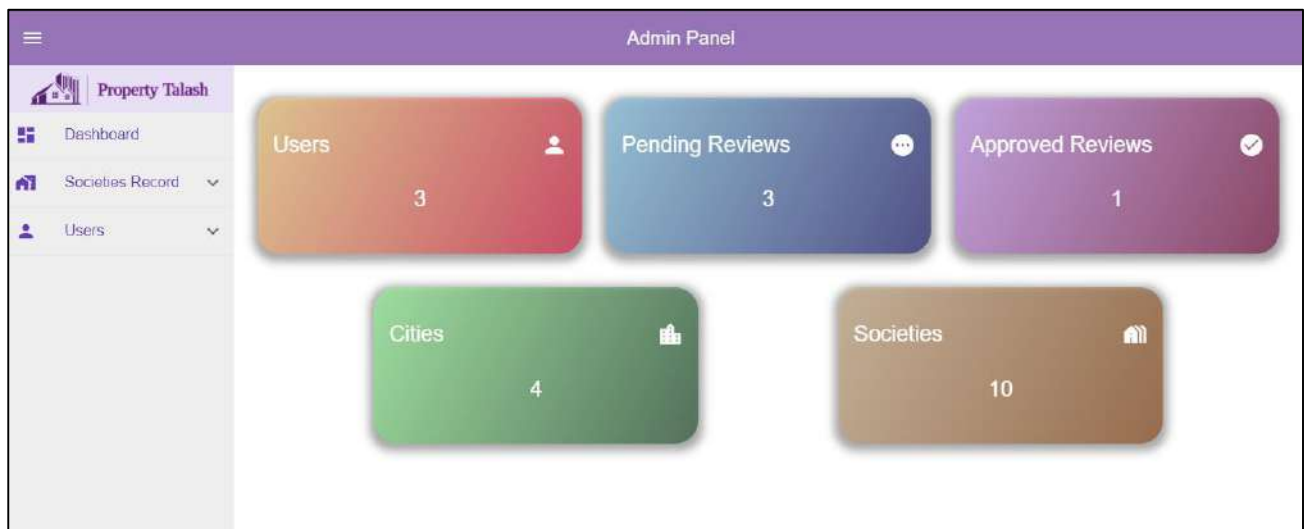


Figure 7.2.1: Admin Dashboard

### 7.2.2. Societies Record

This is the societies record section where all cities and there respective societies record are displayed which search bars. Admin can search a specific city or a specific society using search bar and admin can also delete any society.



Row Number	City ID	Society ID	Society Name	Society About
1	F4RySnw6LclvLk2pzKMc	cvNG5P6lOEh4oCuhluq	ParkView City	PARKVIEW CITY situated in
2	F4RySnw6LclvLk2pzKMc	fx7DSwc1y8H3xBYkzminA	B-17 Multi Garden	B17 Islamabad is created by
3	F4RySnw6LclvLk2pzKMc	sqnXBToe1cZ4k8mduR7V	Blue World City	Blue World City, the flagship

**Figure 7.2.2: Societies Record**

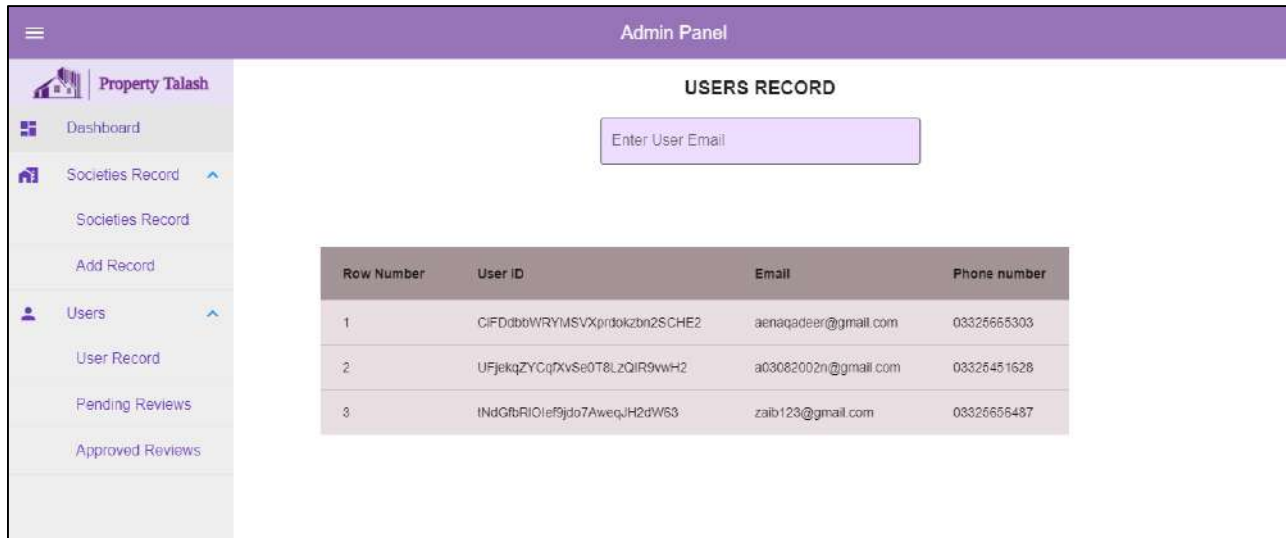
### 7.2.3. Add Record

This is the Add Record screen here admin can add new cities and societies record in the application.

**Figure 7.2.3: Add Society Record**

### 7.2.4. User Record

This is the User record screen where all the users and their details are displayed to the admin with a search bar where admin can search a specific user with its email.

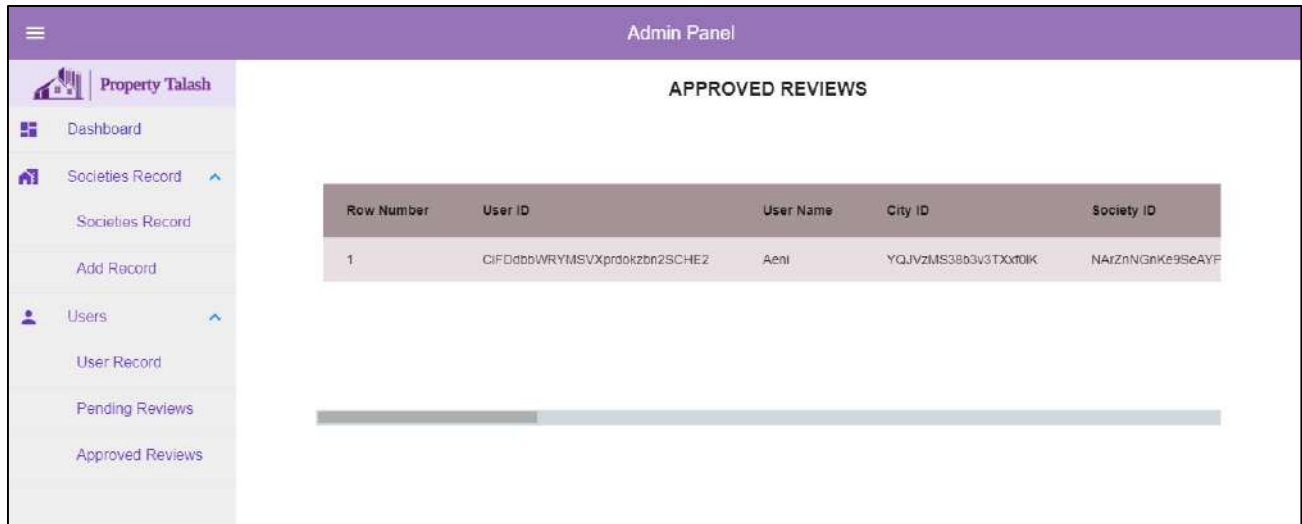


Row Number	User ID	Email	Phone number
1	CiFDdbbWRYMSVXprtkzbn2SCHE2	aenaqadeer@gmail.com	03325655303
2	UFjekqZYCqPvSe0T8LzQIR9vwH2	a03082002n@gmail.com	03325451628
3	INdGfbRIOlef9jdo7AweqjH2dW53	zaib123@gmail.com	03325655487

**Figure 7.2.4: User Record**

### 7.2.5. Approved Reviews

This is the Approved reviews screen where admin can see all the approved reviews of the users

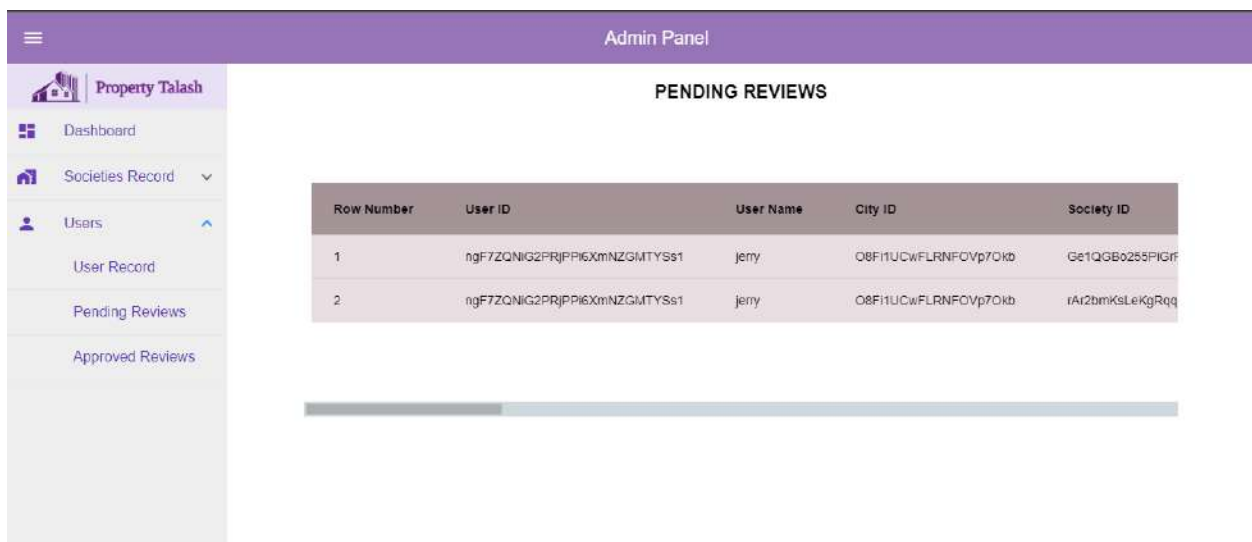


Row Number	User ID	User Name	City ID	Society ID
1	CIFDdbbWRYMSVXprdkzbn2SCHE2	Aenl	YQJVzMS3853v3TXxt0IK	NArZnNGnKe9SeAYF

Figure 7.2.5: Approved reviews Record

### 7.2.6. Pending Reviews

This is the Pending review screen where all the users submitted reviews are displayed which are still waiting for the admin's approval. Admin can approve or disapprove the review if its offensive.



Row Number	User ID	User Name	City ID	Society ID
1	ngF7ZQNIG2PRjPPi6XmNZGMTYs1	jerry	O8F1UCwFLRNFOVp7Okb	Ge1QGB0255PIGrf
2	ngF7ZQNIG2PRjPPi6XmNZGMTYs1	jerry	O8F1UCwFLRNFOVp7Okb	rAr2bmKsLeKgRqq

Figure 7.2.6: Pending reviews Record