

Prime Innovative Hub



Project ID: Fall 2020-2025

Session: BSCS Fall 2020 to 2025

Project Advisor: Nazia Jehan

Submitted By

Fakhar Abbas	70082821
Abdul Raheem Baig	70112370

Department of Computer Science & IT
The University of Lahore
Lahore, Pakistan

Declaration

We have read the project guidelines and we understand the meaning of academic dishonesty, in particular plagiarism and collusion. We hereby declare that the work we submitted for our final year project, entitled **Prime Innovative Hub** is original work and has not been printed, published or submitted before as final year project, research work, publication or any other documentation.

Group Member 1 Name: **Fakhar Abbas**
SAP No: **70082821**

Signature:

Group Member 2 Name: **Abdul Raheem Baig**
SAP No: **70112370**

Signature:

Statement of Submission

This is to certify that **Fakhar Abbas** Roll No.**70082821**, **Abdul Raheem Baig 70112370** successfully submitted the final project named as: **Prime Innovative Hub**, at Computer Science & IT Department, The University of Lahore, Lahore Pakistan, to fulfill the partial requirement of the degree of **BS in Computer Science**.

Supervisor Name:

Signature:

Date:

Dedication

This project is dedicated to my father, who taught me that the best kind of knowledge to have is that which is learned for its own sake. It is also dedicated to my mother, who taught me that even the largest task can be accomplished if it is done one step at a time.

Acknowledgement

We owe a huge debt of gratitude to everyone who contributed to the success of our project, “**Prime Innovative Hub – Real Estate Web Application.**”

Our heartfelt thanks go out to **Mam Nazia Jehan**, whose constant guidance, encouragement, and insightful feedback kept us focused and motivated throughout the development of this platform. Her expertise played a crucial role in helping us overcome challenges and stay aligned with our goals.

We are also incredibly grateful to **Sir Khawaja Qadeer** for his valuable suggestions, timely reviews, and unwavering support during this journey. His mentorship helped us improve and refine our ideas at every stage of the project.

We would like to extend our sincere appreciation to the **University of Lahore** for providing us with the academic environment, infrastructure, and resources necessary to bring this project to life. Their support was vital in transforming our concept into a working solution.

A big shout-out to our amazing team members, whose hard work, dedication, and collaboration made this project possible. Each individual brought unique skills to the table, and together we built something meaningful.

We also want to thank the individuals who tested our application and gave thoughtful feedback. Your insights helped us identify issues and improve user experience significantly.

Finally, we extend our gratitude to the open-source community for providing the tools and frameworks that were essential in building this system. Your contributions empowered us to innovate efficiently.

To everyone who played a role in shaping **Prime Innovative Hub**, thank you from the bottom of our hearts. We are proud of what we’ve achieved and excited about what lies ahead.

Date:

June 23, 2025

Abstract

In Pakistan, the real estate sector often suffers from a lack of transparency and trust, particularly when it comes to the quality and condition of properties. Buyers frequently face challenges due to unreliable property listings, misrepresented details, and the absence of professional verification. To address this critical gap, our project, Prime Innovative Hub, introduces a web-based real estate platform that not only facilitates the buying and selling of properties but also integrates a unique and trusted inspection service. This inspection feature allows buyers to make more informed decisions by reviewing professionally verified property conditions before making any commitments.

The project was developed using a role-based system where users can register as sellers, inspectors, or both. Sellers can manage their property listings, while inspectors, once approved by an admin, can offer inspection services by setting up locations and available time slots. Admins monitor the system, approve inspector applications, and ensure data quality across listings. The platform includes powerful features like advanced search filters, Google Maps integration for real-time location-based service discovery, direct communication with sellers or inspectors, and a smart chatbot to assist users in selecting areas based on their budget and preferences. The result is a reliable, transparent, and user-friendly real estate platform that builds buyer confidence and bridges the trust gap in the property market. This system demonstrates how technology can transform traditional real estate processes by making them more accountable and accessible.

Area of the Project

Web Base Application

Technologies used

The project Prime Innovative Hub is developed using a modern full-stack technology stack. For the frontend, React.js is used to build a dynamic and responsive user interface, along with HTML, CSS, and JavaScript to structure, style, and add interactivity to the web pages. The backend is developed using NestJS, a progressive Node.js framework that supports scalable server-side application development.

For the database, PostgreSQL is used to store and manage structured data, including user details, properties, services, and contact information. The project also integrates OpenAI's API to power the chatbot, which utilizes Natural Language Processing (NLP) to assist users with real estate-related queries. Additionally, Google Maps API is used to implement geolocation features, allowing users to search and visualize properties and services based on their location. The overall development environment includes tools like Node.js, npm,

List of Figures

Figure 1 use case for Create Account	17
Figure 2 use case for Login Process.....	18
Figure 3 use case for home page	19
Figure 4 use case for maps navbar	20
Figure 5 use case for Services	21
Figure 6 use case for seller dashboard	22
Figure 7 use case for inspector Dashboard	23
Figure 8 use case for Admin Panel	24
Figure 9 use case for contact & communication.....	25
Figure 10 use case for Chatbot.....	26
Figure 11 Architecture Diagram.....	27
Figure 12 ERD	28
Figure 13 Level 0 DFD	29
Figure 14 Level 1 DFD	29
Figure 15 Class Diagram.....	30
Figure 16 Activity Diagram Create Account	31
Figure 17 Activity Diagram Login Process	31
Figure 18 Activity Diagram Add Property (Seller)	32
Figure 19 Activity Diagram Update or Delete Property (seller).....	32
Figure 20 Activity Diagram Submit Inspector Application.....	33
Figure 21 Activity Diagram Approve/Reject Inspector (Admin)	33
Figure 22 Activity Diagram Manage Services (Inspector)	34
Figure 23 Activity Diagram Property/Inspector (CHECK)	34
Figure 24 Activity Diagram View On Map	35
Figure 25 Activity Diagram Use ChatBot.....	35
Figure 26 Sequence Diagram Create Account	36
Figure 27 Sequence Diagram Create Account	36
Figure 28 Sequence Diagram Add Property (seller)	37
Figure 29 Sequence Diagram Update Delete Property	37
Figure 30 Sequence Diagram Submit Inspector Application.....	38
Figure 31 Sequence Diagram Admin Approve Reject Inspector.....	38
Figure 32 Sequence Diagram Inspector Adds/Manages Services	39
Figure 33 Sequence Diagram Property Inspector Check	39
Figure 34 Sequence Diagram View On Maps	40
Figure 35 Sequence Diagram Use ChatBot	40
Figure 36 Collaboration Diagram	41
Figure 37 State Transition Diagram.....	42

Figure 38 Component Diagram.....	43
Figure 39 Deployment Diagram	44

List of Tables

Table 1 Functional Requirements Create Account	9
Table 2 Functional Requirements Login Account	9
Table 3 Functional Requirements Role base Redirection	10
Table 4 Functional Requirements Add Property.....	10
Table 5 Functional Requirements Delete/Update property	10
Table 6 Functional Requirements Submit Inspector Application	11
Table 7 Functional Requirements Approve/Reject Inspector	11
Table 8 Functional Requirements Inspector Service Add/Delete/Update	11
Table 9 Functional Requirements Property/Inspector Search.....	12
Table 10 Functional Requirements Map View	12
Table 11 Functional Requirements Use Chatbot	12
Table 12 Functional Requirements Admin-Delete Property.....	13
Table 13 use case for Create Account.....	17
Table 14 use case for Login Process	18
Table 15 use case for Home Page	19
Table 16 use case for Maps Navigation	20
Table 17 use case for Services	21
Table 18 use case for seller dashboard.....	22
Table 19 use case for inspector dashboard.....	23
Table 20 use case for Admin Panel.....	24
Table 21 use case for Contact & communication	25
Table 22 use case for Chatbot	26
Table 23 User Registration Positive.....	45
Table 24 User Registration Negative	45
Table 25 User Login Positive.....	46
Table 26 User Login Negative	46
Table 27 Property Add By Seller Positive	47
Table 28 Property Add By Seller Negative	47
Table 29 Inspector Application Submission Positive	48
Table 30 Inspector Application Submission Negative.....	48
Table 31 Admin Inspector Approval (Positive).....	49
Table 32 Admin Inspector Approval (Negative)	49
Table 33 Service CRUD by Inspector (Positive)	50
Table 34 Service CRUD by Inspector (Negative)	50
Table 35 Chatbot Query (Positive)	51
Table 36 Chatbot Query (Negative).....	51
Table 37 Search Property Listings	52

Table 38 Invalid Property Search Filters	52
Table 39 Inspector Booking from Map	53
Table 40 Contact via Email to Seller/Inspector	53
Table 41 Chatbot Real Estate Advice	54
Table 42 Chatbot Invalid Input Handling	54
Table 43 EP Table – User Registration	55
Table 44 EP Table – Property Search	55
Table 45 EP Table – Inspector Application Submission	55
Table 46 EP Table – Admin Approval Process	55
Table 47 EP Table – Service CRUD by Inspector	55
Table 48 EP Table – Chatbot Queries	56
Table 49 BVA Table – Password Length Validation (User Registration)	56
Table 50 BVA Table – Property Price Range Filter	56
Table 51 BVA Table – Slot Time Range (Inspector Service)	57
Table 52 BVA Table – Inspector Experience Years	57
Table 53 BVA Table – Email Input Characters	57
Table 54 Decision Table – User Role Redirection After Login	57
Table 55 Decision Table – Property Search Filter Logic	58
Table 56 Decision Table – Inspector Application Review (Admin)	58
Table 57 Decision Table – Chatbot Response Scenarios	58
Table 58 Decision Table – Inspector Service CRUD Validations	58
Table 59 State Transition Table – User Registration Flow	59
Table 60 State Transition Table – Inspector Application Review	59
Table 61 State Transition Table – Property Posting by Seller	59
Table 62 State Transition Table – Chatbot Session Handling	59
Table 63 State Transition Table – Login & Role-Based Dashboard Routing	60
Table 64 Test Case 1 – Function: Register User()	64
Table 65 Test Case 2 – Function: Submit Inspector Application	65
Table 66 Test Case 3 – Function: Approve Or Reject Inspector()	65
Table 67 Test Case 4 – Function: AddProperty()	66
Table 68 Test Case 5 – Function: filterProperties()	66
Table 69 Test Case 6 – Function: ChatbotRespond()	66
Table 70 Summary Table	69
Table 71 Test Case 1 – Search Function Response Time	69
Table 72 Test Case 2 – Map Load Time	70
Table 73 Test Case 3 – Chatbot Reply Time	70
Table 74 Test Case 1 – Simulated User Surge	71
Table 75 Test Case 2 – Concurrent Inspector Applications	71
Table 76 Test Case 3 – Mass Property Upload by Sellers	71
Table 77 Test Case 1 – Role-Based Redirection Post Login	72
Table 78 Test Case 2 – Admin Approval Flow for Inspector	72

Table 79 Test Case 3 – Property Listing and Display Flow	72
Table 80 Regression Test Cycle – Example 1	73
Table 81 Regression Test Cycle – Example 2	73
Table 82 Regression Test Cycle – Example 3	74

Table of Content

<i>Declaration</i>	i
<i>Statement of Submission</i>	ii
<i>Dedication</i>	iii
<i>Acknowledgement</i>	iv
<i>Abstract</i>	v
List of Figures	vii
List of Tables	ix
Chapter 1: Introduction to the Problem	1
1.1 Introduction	1
1.2 Purpose	1
1.3 Objective	1
1.4 Existing Solution	2
1.5 Proposed Solution	2
Chapter 2: Software Requirement Specification	3
2.1 Introduction	3
2.1.1 Purpose	3
2.1.2 Scope	4
2.1.3 Definitions, acronyms, and abbreviations	5
2.2 Overall description	6
2.2.1 Product perspective	8
2.2.2 Product functions	8
2.2.3 User characteristics	13
2.2.4 Constraints	14
2.2.5 Assumptions and dependencies	15
2.2.6 Apportioning of requirements	15
2.3 Specific requirements	15
2.3.1 Functional Requirement	15
2.3.2 Non-functional Requirements	16

Chapter 3: Use Case Analysis	16
Chapter 4: Design	27
4.1 Architecture Diagram	27
4.1.1 ERD with data dictionary	28
4.2 Data Flow diagram	28
4.2.1 The level 0	28
4.2.2 The level 1	29
4.3 Class Diagram	30
4.4 Activity Diagram	31
9. View on Map	35
10. Use Chatbot	35
4.5 Sequence Diagram	36
4.6 Collaboration Diagram	41
4.7 State Transition Diagram	42
4.8 Component Diagram	43
4.9 Deployment Diagram	44
Chapter 5: Testing	45
5.1 Test Case Specifications	45
Test Case – User Registration (Positive)	45
Test Case – User Registration (Negative)	45
Test Case – User Login (Positive)	46
Test Case – User Login (Negative)	46
Test Case – Property Add by Seller (Positive)	47
Test Case – Property Add by Seller (Negative)	47
Test Case – Inspector Application Submission (Positive)	48
Test Case – Inspector Application Submission (Negative)	48
Test Case – Admin Inspector Approval (Positive)	49
Test Case – Admin Inspector Approval (Negative)	49
Test Case – Service CRUD by Inspector (Positive)	50
Test Case – Service CRUD by Inspector (Negative)	50
Test Case – Chatbot Query (Positive)	51
Test Case – Chatbot Query (Negative)	51

Black Box Test Cases	52
Test Case – Search Property Listings	52
Test Case – Invalid Property Search Filters.....	52
Test Case – Inspector Booking from Map.....	53
Test Case – Contact via Email to Seller/Inspector	53
Test Case – Chatbot Real Estate Advice.....	54
Test Case – Chatbot Invalid Input Handling.....	54
5.1.1 Equivalence Partitions (EP).....	54
EP Table – User Registration.....	55
EP Table – Property Search.....	55
EP Table – Inspector Application Submission	55
EP Table – Admin Approval Process	55
EP Table – Service CRUD by Inspector	55
EP Table – Chatbot Queries	56
5.1.2 Boundary Value Analysis.....	56
BVA Table – Password Length Validation (User Registration).....	56
BVA Table – Property Price Range Filter.....	56
BVA Table – Slot Time Range (Inspector Service).....	56
BVA Table – Inspector Experience Years	57
BVA Table – Email Input Characters	57
5.1.3 Decision Table Testing.....	57
Decision Table – User Role Redirection After Login	57
Decision Table – Property Search Filter Logic.....	57
Decision Table – Inspector Application Review (Admin)	58
Decision Table – Chatbot Response Scenarios	58
Decision Table – Inspector Service CRUD Validations	58
5.1.4 State transition Testing	58
State Transition Table – User Registration Flow.....	58
State Transition Table – Inspector Application Review	59
State Transition Table – Property Posting by Seller	59
State Transition Table – Chatbot Session Handling.....	59
State Transition Table – Login & Role-Based Dashboard Routing.....	60

5.1.5	Use Case Testing	60
	Use Case: UC_01 – Create Account	60
	Use Case: UC_02 – Add Property (Seller Panel)	61
	Use Case: UC_03 – Submit Inspector Application	61
	Use Case: UC_04 – Approve/Reject Inspector (Admin).....	61
	Use Case: UC_05 – Search Property/Service	62
	Use Case: UC_06 – Contact Seller or Inspector	62
	Use Case: UC_07 – Use AI Chatbot.....	63
5.2	White Box Test Cases	64
	Test Case 1 – Function: Register User()	64
	Test Case 2 – Function: Submit Inspector Application	65
	Test Case 3 – Function: Approve Or Reject Inspector().....	65
	Test Case 4 – Function: AddProperty()	65
	Test Case 5 – Function: filterProperties()	66
	Test Case 6 – Function: ChatbotRespond().....	66
5.2.1	Cyclometric complexity	67
	Function: registerUser().....	67
	Function: approveOrRejectInspector()	67
	Function: filterProperties()	68
	Function: chatbotRespond()	68
	Summary Table:	69
5.4	Performance Testing.....	69
	Test Case 1 – Search Function Response Time	69
	Test Case 2 – Map Load Time	70
	Test Case 3 – Chatbot Reply Time	70
5.5	Stress Testing	70
	Test Case 1 – Simulated User Surge	70
	Test Case 2 – Concurrent Inspector Applications	71
	Test Case 3 – Mass Property Upload by Sellers.....	71
5.6	System Testing	71
	Test Case 1 – Role-Based Redirection Post Login	71
	Test Case 2 – Admin Approval Flow for Inspector.....	72

Test Case 3 – Property Listing and Display Flow.....	72
5.7 Regression Testing	72
5.7.1 Selecting Regression Tests.....	72
Regression Test Cycle – Example 1	73
Regression Test Cycle – Example 2	73
Regression Test Cycle – Example 3	73
5.7.2 Regression Testing Steps	74
Chapter 6: Tools and Techniques	75
Abstract Outline.....	75
6.1 Languages Used	75
6.2 Applications and Tools.....	75
6.3 Libraries and Extensions	75
Chapter 7: Summary and Conclusion	76
Summary.....	76
Conclusion	76
Chapter 8: User Manual.....	77
8.1 Login Function.....	77
8.2 Property Listing Search	78
Map View:.....	79
8.2 Booking an Inspection Slot	80
8.4 Profile Management (Seller/Inspector).....	82
Inspector Dashboard:	85
Chatbot:	85
Admin Dashboard:	86
Chapter 9: Lessons Learnt and Future Work	86
Lessons Learnt	86
Future Work.....	87
References	88
Books	88
Book Chapters.....	88
Electronic Books	88
Journal Articles	88

E-Journal Articles	88
Conference Papers.....	88
Theses / Dissertations.....	88
Online Documents	88
Appendix.....	90
Appendix A: Database Schema	90
Appendix B: Sample Code Snippets	90
Appendix C: UI Snapshots	90
Appendix D: Test Cases.....	90

Chapter 1: Introduction to the Problem

1.1 Introduction

In the real estate sector of Pakistan, both buyers and sellers face significant challenges in terms of trust, transparency, and accessibility. Property listings are often outdated or misleading, and there is usually no standardized system to verify the condition or legitimacy of properties. Moreover, there is a lack of centralized platforms that allow users to access reliable inspection services or easily connect with sellers and inspectors in a specific area. This creates uncertainty and inefficiencies in property transactions.

To address these issues, the project Prime Innovative Hub is proposed as a comprehensive web-based platform that streamlines real estate activities by allowing users to search, list, and inspect properties in a transparent and organized manner. The integration of modern technologies such as AI-powered chatbots, Google Maps for geolocation, and verified user roles (seller and inspector) ensures that users have a reliable and interactive experience. By solving real-world problems in property dealings, this project aims to bridge the gap between digital innovation and the traditional real estate market.

1.2 Purpose

The purpose of developing Prime Innovative Hub is to bring authenticity and trust to the real estate market by introducing a system where users can access verified property inspection services. In many cases, buyers face uncertainty due to misleading or incomplete property listings, which can lead to financial loss and dissatisfaction. This project addresses that issue by allowing users to hire approved inspectors who evaluate the condition of properties before purchase, ensuring transparency and informed decision-making. By integrating inspection as a core feature, the platform not only reduces the risk of fraud but also promotes safer and more reliable property transactions in society.

1.3 Objective

The main objective of Prime Innovative Hub is to develop a reliable, user-friendly web platform that connects property buyers, sellers, and inspectors in a secure, transparent environment. A core goal is to introduce a verified inspection system that helps buyers make confident decisions by ensuring property details are accurate and professionally assessed. The platform includes advanced search filters, Google Maps integration, and seamless user-to-service communication.

It also offers separate dashboards for sellers and inspectors to manage listings and services efficiently. The admin panel verifies inspector applications and monitors platform activity to ensure only approved services are accessible. An AI-powered chatbot enhances the user experience with instant responses and real estate guidance. Overall, the project aims to improve the credibility and efficiency of property transactions through digital innovation.

1.4 Existing Solution

In the current real estate market, platforms like Zameen.com and OLX provide users with the ability to search, view, and list properties online. These platforms offer basic filters such as location, price, and property type, making it easier for buyers and sellers to connect. However, one of the major shortcomings of these existing solutions is the lack of any verified property inspection system. Buyers have no way of knowing the actual condition of a property before visiting or purchasing, which often leads to misinformation, hidden defects, and ultimately, a lack of trust in the transaction process.

Another issue with these platforms is the absence of admin moderation and structured role-based access. There is no dedicated panel for property inspectors, and sellers are not required to verify their listings through any standardized process. This creates a gap in reliability, as anyone can post property details without validation. Moreover, communication between buyers, sellers, and service providers is unstructured, and there's no intelligent system like a chatbot to guide users. These limitations highlight the need for an improved system—one that introduces inspection as an integral part of the real estate process, offering transparency, accountability, and confidence to users.

1.5 Proposed Solution

To address the limitations of existing real estate platforms, we have proposed Prime Innovative Hub, a web-based solution that not only allows users to search and list properties but also introduces a verified property inspection system a unique feature currently missing in the market. This system enables buyers to connect with approved inspectors who can professionally assess the condition of a property before a purchase decision is made. Inspectors are required to submit an application, which is reviewed by an admin, ensuring only trusted individuals are allowed to offer inspection services. This adds a crucial layer of authenticity and trust to the property-buying process.

In addition, our platform offers role-based dashboards for sellers and inspectors, a centralized admin panel for monitoring and approval, Google Maps integration for location-based property and service discovery, and an AI-powered chatbot to assist users with queries, budget suggestions, and area recommendations. Unlike existing solutions, Prime Innovative Hub focuses on providing a secure, verified, and user-personalized experience, ensuring transparency and accountability at every stage of the transaction. By bridging the trust gap and offering features not available on current platforms, our system presents a more complete and reliable real estate solution for the market.

Chapter 2: Software Requirement Specification

2.1 Introduction

2.1.1 Purpose

The "Prime Innovative Hub" project aims to provide a comprehensive web-based platform that simplifies and authenticates the real estate experience for buyers, sellers, and inspectors. The main objectives of the project are as follows:

Ensuring Authenticity through Property Inspections:

A core purpose of this project is to introduce verified property inspections as an integral part of the real estate process. Unlike existing platforms that rely solely on user-submitted information, Prime Innovative Hub enables users to connect with approved inspectors who can professionally assess property conditions. This feature brings a new level of trust and reliability to property transactions, helping buyers make more informed decisions.

Facilitating Role-Based Dashboards:

The system provides customized dashboards for sellers and inspectors. Sellers can list, manage, and update their properties, while inspectors can manage their services, schedule slots, and define service areas. This separation ensures a structured and user-centric workflow tailored to each role's responsibilities.

Simplifying Search and Discovery:

To make property browsing efficient, the platform offers advanced search filters and Google Maps integration. Users can filter listings by city, location, price, and property type (rent or buy), and view both properties and inspectors visually on the map. This feature streamlines the property discovery process and helps users find relevant services based on location.

Enhancing User Support through AI Chatbot:

The platform features an AI-powered chatbot integrated with Open AI to guide users throughout their journey. It can assist with common questions, budget-based area suggestions, and general real estate queries, providing a smart and interactive experience even for first-time users.

Improving Transparency and Communication:

Buyers can directly contact sellers or inspectors through email or phone based on the listing details. This clear communication channel eliminates middlemen and enhances trust between parties. The admin panel further ensures transparency by reviewing and approving inspector applications and managing all listings.

2) Specific Audience for the SRS:

Development Team:

This document is intended for the developers responsible for building Prime Innovative Hub. It provides them with clear, structured requirements for implementing each feature and integration accurately.

Project Managers:

Project managers will use the SRS to track development progress, manage timelines, assign tasks, and ensure the project stays within defined scope and budget.

Quality Assurance (QA) Team:

The QA team will use this document to develop and execute test cases to verify that all features meet both functional and non-functional requirements.

Stakeholders/Clients:

Supervisors, evaluators, and potential investors will refer to the SRS to understand the goals, scope, and expected deliverables of the system. It ensures alignment between the project vision and actual implementation.

Maintenance and Support Team:

Post-deployment, this document will guide support teams in resolving bugs, implementing feature enhancements, and maintaining overall platform quality.

End Users (Indirectly):

Though not the direct readers of the SRS, the needs and expectations of end users—buyers, sellers, and inspectors are central to the system. The document ensures that the final product is user-friendly, secure, and effective in solving real-world real estate problems.

2.1.2 Scope

1. **Software Product Name:**

The software product to be developed is named "**Prime Innovative Hub.**"

2. **What the Software Will Do:**

- a. Allow users to register as sellers, inspectors, or both, and access dedicated dashboards.
- b. Enable sellers to list, update, and manage their property postings.
- c. Enable inspectors to submit applications for approval and, once verified, offer property inspection services.
- d. Display approved properties and inspection services on the frontend with filtering options (city, price, rent/buy, etc.).
- e. Integrate Google Maps to show real-time property and inspector locations.
- f. Provide an AI-powered chatbot to assist users with queries and area suggestions.
- g. Facilitate direct communication between users via email and contact number visibility.

h. Include an admin panel to manage property listings and approve/reject inspector applications.

What the Software Will Not Do:

- a. Will not handle financial transactions, such as online payments or escrow services.
- b. Will not perform legal verification of property ownership or documentation.
- c. Will not allow unverified inspectors to offer services publicly.
- d. Will not provide mobile app functionality in the initial version.

3. Application of the Software:

Prime Innovative Hub is intended to serve property buyers, sellers, and professional property inspectors, primarily within the real estate market of Pakistan. The platform is designed for web access and will provide a centralized hub for secure, transparent, and well-informed real estate interactions.

4. Relevant Benefits, Objectives, and Goals:

Relevant Benefits:

- a. Introduces inspection as a verified and trackable service, building trust in property transactions.
- b. Offers role-based functionality to streamline workflows for sellers and inspectors.
- c. Improves user experience with location-based services and intelligent chatbot support.
- d. Ensures content authenticity through admin moderation and service approval mechanisms.
- e. Reduces fraud and misinformation by enabling professional inspections before purchase decisions.

Objectives and Goals:

- a. Improve trust and transparency in the online real estate market.
- b. Empower users to make informed decisions through verified property inspections.
- c. Enhance property search and service discovery through intuitive design and map integration.
- d. Establish a scalable foundation for future features like payments, reviews, or mobile applications

2.1.3 Definitions, acronyms, and abbreviations

This subsection provides the definitions of key terms, acronyms, and abbreviations used throughout the Software Requirement Specification (SRS) document to ensure clarity and consistency.

Definitions:

- **User:** A person who registers and interacts with the system. A user can be a buyer, seller, inspector, or admin.
- **Seller:** A registered user who lists properties for sale or rent on the platform.
- **Inspector:** A professional who provides property inspection services after being approved by the admin.
- **Admin:** The system administrator responsible for managing users, verifying inspectors, and moderating listings.
- **Inspection:** A professional evaluation of a property's condition, requested by a user and conducted by an approved inspector.
- **Dashboard:** A personalized interface where users (seller or inspector) can manage their listings or services.
- **Chatbot:** An AI-powered virtual assistant that answers user queries, provides suggestions, and helps with navigation.
- **Listing:** A property posted by a seller with detailed information such as location, price, type, and availability.

2.2 Overall description

System Interfaces:

The system will interact with:

- **Google Maps API** for visualizing property and inspector locations.
- **OpenAI API** for chatbot support.
- **SMTP or mailing service** for sending inquiry emails from customers to sellers/inspectors.
- **Database (PostgreSQL)** to store user, property, service, and contact information.

User Interfaces:

The application will provide the following user interfaces:

- **Home Page:** Search bar with filters (city, price, rent/buy) and featured listings.
- **Maps Page:** Google Map view showing available properties and inspectors by city.
- **Services Page:** Search for available inspection services with filters.
- **Dashboards:**
 - **Seller Dashboard:** For managing property listings (CRUD).
 - **Inspector Dashboard:** For managing services, slots, and locations.
 - **Admin Panel:** For moderating properties and approving/rejecting inspector applications.

- **Login/Signup Pages:** With role selection (seller, inspector, or both).
- **Chatbot Interface:** For real-time guidance and support.

Hardware Interfaces:

There are no specific hardware dependencies. The application will run on standard computing devices (laptops/desktops/tablets) with an internet connection. No special hardware is required for users or admins.

Software Interfaces:

- **Frontend:** Built using React.js (JavaScript framework)
- **Backend:** Developed with NestJS (Node.js framework)
- **Database:** PostgreSQL
- **External APIs:**
 - Google Maps API
 - OpenAI API for chatbot
- **Server Requirements:**
 - Node.js runtime
 - PostgreSQL support
 - RESTful API framework compatibility

Communications Interfaces:

- Uses **HTTPS** for secure web communication.
- Communicates with APIs using **RESTful protocol**.
- Relies on **SMTP or Email API** for message delivery between users (inquiries).
- Hosting assumes a **cloud server** with internet access.

Memory Requirements:

- **Primary Memory (RAM):** Minimum 4 GB RAM recommended for local development.
- **Secondary Storage:** Minimum 10 GB disk space required on the server to store application code, assets, logs, and database.
- The PostgreSQL database will scale based on property listings, user accounts, and service data.

Operations:

- **Normal operations** include user login, property listing, inspection booking, and admin verification.
- **Special operations** include:
 - **Admin panel actions:** Accept/reject inspector applications, delete property listings.
 - **Chatbot communication**
 - **Data backup:** Daily scheduled database backups (can be implemented via cron jobs or cloud functions).
 -

- **Error recovery:** Logging and error-handling mechanisms will be in place to recover from failures gracefully.

Site Adaptation Requirements:

- The system can be deployed for different cities or regions by initializing relevant city/location data.
- Admin credentials, email configuration, Google Maps API keys, and chatbot API tokens must be set up at each site.
- Language localization and currency format (e.g., PKR) can be adapted depending on geographic expansion.

2.2.1 Product perspective

2.2.2 Product functions

The system is composed of several functional modules that interact to provide a seamless user experience. These modules handle account management, property listings, inspection services, chatbot support, search and filtering, and admin moderation. Each function is critical in fulfilling the project's goal of providing a trusted, interactive real estate platform.

Below are the defined functional requirements using the standard format.

ID:	FR_01			
Name:	Create Account			
Description	Input	Output	Requirements	Basic Work Flow
Enter details to create account	Name,Email, Password,Confirm Password,Phone, Select Role.	Account created	Internet Connectivity required	Enter correct information and click submit button System save the record in database

Table 1 Functional Requirements Create Account

ID:	FR_02			
Name:	Login Account			
Description	Input	Output	Requirements	Basic Work Flow
Allows users to log in	Email, Password	User redirected to their dashboards	Internet Connectivity required	Enter valid credentials and click login. System verifies and redirects user.

Table 2 Functional Requirements Login Account

ID:	FR_03			
Name:	Role base Redirection			
Description	Input	Output	Requirements	Basic Work Flow
Allows users to redirect according to role base	Check role	User redirected to their dashboards	Internet Connectivity required	Select Role whether want to login in Dasboard

Table 3 Functional Requirements Role base Redirection

ID:	FR_04			
Name:	Add Property			
Description	Input	Output	Requirements	Basic Work Flow
Seller can add new property listings.	Property title, location, price, type (rent/sell), images etc,	Property listed on frontend	Internet Connectivity required	Seller fills form; system validates and stores property.

Table 4 Functional Requirements Add Property

ID:	FR_05			
Name:	Delete/Update Property			
Description	Input	Output	Requirements	Basic Work Flow
Seller can update/delete existing property listings	Property IdProperty title, location, price, type (rent/sell), images etc,	Property listed updated or deleted on frontend	Internet Connectivity required	Seller fills form; system validates and stores property.

Table 5 Functional Requirements Delete/Update property

ID:	FR_06			
Name:	Submit Inspector Application			
Description	Input	Output	Requirements	Basic Work Flow
Inspector submits application to be verified by admin.	Inspector info, experience letter image, work experience select inspection type	User Application will be send to admin panel	Internet Connectivity required	User fill form to apply for inspector in our organization

Table 6 Functional Requirements Submit Inspector Application

ID:	FR_07			
Name:	Approve/Reject Inspector			
Description	Input	Output	Requirements	Basic Work Flow
Admin reviews inspector applications and approves or rejects.	Inspector ID, Approve or Reject	Application status updated	Internet Connectivity required	Admin views list, selects application, approves/rejects.

Table 7 Functional Requirements Approve/Reject Inspector

ID:	FR_08			
Name:	Inspector Service Add/Delete/Update			
Description	Input	Output	Requirements	Basic Work Flow
Approved inspector can manage services, slots, and locations.	Service details, slot timings, location info	Service shown in frontend listings and filters	Internet Connectivity required	Inspector fills form; system updates data.

Table 8 Functional Requirements Inspector Service Add/Delete/Update

ID:	FR_09			
Name:	Property/Inspector Search			
Description	Input	Output	Requirements	Basic Work Flow
Users can search properties and services using filters.	City, price, rent/buy, location	Filtered search results displayed	Internet Connectivity required	User enters filters; system returns matched results.

Table 9 Functional Requirements Property/Inspector Search

ID:	FR_10			
Name:	Map View			
Description	Input	Output	Requirements	Basic Work Flow
Displays properties and inspectors on Google Map.	Selected city or area	Map with location pins	Internet Connectivity required	User selects location; map displays corresponding pins.

Table 10 Functional Requirements Map View

ID:	FR_11			
Name:	Use Chatbot			
Description	Input	Output	Requirements	Basic Work Flow
User can ask real estate-related questions via chatbot.	User question	AI-generated answer	Internet Connectivity required	User types query; chatbot processes and replies.

Table 11 Functional Requirements Use Chatbot

ID:	FR_12			
Name:	Admin – Delete Property			
Description	Input	Output	Requirements	Basic Work Flow
Admin can delete any property from system.	Property ID	Property removed permanently	Internet Connectivity required	Admin selects and deletes listing; update reflected on frontend.

Table 12 Functional Requirements Admin-Delete Property

2.2.3 User characteristics

- **Diverse User Roles:** Users of the platform include property buyers, property sellers, certified inspectors, and administrators, each with different goals and system access levels.
- **Real Estate Participants:** Users may include first-time homebuyers, property investors, rental seekers, and independent sellers, all seeking an intuitive and secure property platform.
- **Moderate Technical Knowledge:** Most users are expected to have basic computer literacy and familiarity with browsing websites, filling out forms, using search filters, and viewing maps.
- **Service Providers (Inspectors):** Inspectors are likely to have professional or semi-professional backgrounds, capable of handling digital tools for service listings, slot management, and location mapping.
- **Admin Users:** Admins will possess higher technical understanding, responsible for moderating listings, reviewing inspector applications, and maintaining platform integrity.
- **Trust-Focused Users:** Buyers and sellers are particularly concerned with authenticity, transparency, and reliability in transactions, and will be drawn to features like verified inspections and direct contact.
- **Multi-Device Access:** Users may access the platform from desktop computers, laptops, tablets, or mobile devices, and expect a consistent experience across all.
- **Location-Oriented Users:** Many users will rely heavily on map-based searches and filtering by city or locality to find nearby properties and inspection services.
- **Feedback-Oriented:** Users, especially sellers and inspectors, may provide **feedback or report issues** to improve their experience, enabling ongoing platform enhancement

2.2.4 Constraints

- **Regulatory Policies:**
The system must comply with local **data protection and privacy regulations** in Pakistan. Any user data collected (e.g., emails, contact numbers) must be securely stored and not misused or exposed to third parties.
- **Hardware Limitations:**
The platform is optimized for modern web browsers and standard computing devices. Performance may degrade on devices with **low processing power or outdated browsers**.
- **Interfaces to Other Applications:**
The system depends on **external APIs** such as **Google Maps API** for location services and **OpenAI API** for chatbot functionality. Any changes, outages, or limitations in these APIs may impact system behavior.
- **Parallel Operation:**
The application is designed as a **single-instance platform** and is not currently intended for distributed deployment across multiple nodes or regions in parallel.
- **Audit Functions:**
Admin users have limited **manual control for auditing**, such as viewing inspector applications and property listings. There is currently no automatic activity logging or version history tracking implemented.
- **Control Functions:**
Only admin users have access to platform-wide control functions such as **inspector approval/rejection** and **property moderation**.
- **Higher-Order Language Requirements:**
The system is developed using **JavaScript and TypeScript**, specifically with **React.js** on the frontend and **NestJS** on the backend. The choice of frameworks may limit portability to other tech stacks without substantial modification.
- **Signal Handshake Protocols:**
Not applicable, as the system does not involve hardware-level communications or embedded systems requiring signal protocols.
- **Reliability Requirements:**
As a real estate platform, **high availability and responsiveness** are expected. However, full fault-tolerance or zero-downtime is not guaranteed in this version and depends on server deployment and third-party service stability.
- **Criticality of the Application:**
The application is **moderately critical**, especially regarding trust and data accuracy. False information or failed inspection approval workflows could directly impact user decisions and credibility.
- **Safety and Security Considerations:**
User authentication is handled via password-based login. The system must implement **form validation, input sanitization, and role-based access control** to prevent unauthorized access and protect user data. Admin functions must be strictly protected through credential checks.

2.2.5 Assumptions and dependencies

The development and functionality of Prime Innovative Hub are based on several key assumptions and dependencies. It is assumed that users will have access to a stable internet connection and use modern web browsers to access the platform. The system relies on third-party services such as Google Maps API for geolocation and open AI API for chatbot support; any changes or outages in these services may impact system features. It is also assumed that an active admin will be available to review inspector applications and moderate property listings to maintain platform trust. The platform depends on a secure hosting environment for data protection and performance, and on a functioning email service for user-to-user communication. Additionally, it is assumed that users will provide accurate and honest information when registering or listing properties. Any change in these assumptions could require modifications in the system's design or functionality.

2.2.6 Apportioning of requirements

While the initial version of Prime Innovative Hub focuses on core functionalities such as user registration, property listing, inspector application, admin moderation, map-based search, and chatbot support, some features have been identified for future development. These include the integration of an online payment gateway for booking inspection services, a user rating and review system for inspectors and sellers, push notifications for activity updates, mobile application versions for Android and iOS, multilingual support, and detailed analytics for admin users. These enhancements are not essential for the current release but will significantly improve user experience, scalability, and operational efficiency in later versions of the system.

2.3 Specific requirements

This section describes the functional and non-functional requirements of the Prime Innovative Hub system in detail to help the designers and testers ensure the system is developed according to the user's expectations and verified appropriately.

2.3.1 Functional Requirement

The key functions/modules of the Prime Innovative Hub project include:

- **User Registration Module:**
Allows users to register as a seller, inspector, or both. Role-based access is provided upon signup.
- **Login & Authentication Module:**
Enables secure login and redirects users to their respective dashboards (Seller, Inspector, Admin).
- **Seller Dashboard Module:**
Allows sellers to perform CRUD operations on property listings.
- **Inspector Dashboard Module:**
Enables approved inspectors to create and manage inspection services, including setting time slots and service locations.

- **Inspector Application Module:**
Allows newly registered inspectors to submit applications for admin review and approval.
- **Admin Panel Module:**
Provides admin with controls to view all properties, approve/reject inspector applications, and delete any property listing.
- **Search and Filter Module:**
Allows users to search for properties and services using filters like city, location, price, and rent/sale status.
- **Google Maps Integration Module:**
Displays pins for properties and inspectors based on user input for city or area.
- **Contact Module:**
Allows users to send inquiry messages to sellers or inspectors via a contact form (email-based communication).
- **Chatbot Module:**
Integrates AI-powered chatbot to assist users with area suggestions, budget planning, and general real estate queries.

2.3.2 Non-functional Requirements

- **Usability:**
The system will have an intuitive user interface, ensuring ease of use for individuals with basic computer skills. Role-specific dashboards will simplify user workflows.
- **Reliability:**
The system will operate with a high degree of reliability, ensuring that all services (e.g., login, chatbot, map view) function correctly under normal usage without unexpected failures.
- **Performance:**
The application will respond to user actions, such as searching, logging in, and chatbot interaction, within 2–3 seconds. The map and filtered results will load smoothly.
- **Design Constraints:**
The system is developed using **React.js** for frontend, **NestJS** for backend, and **PostgreSQL** for the database. It must remain compatible with these technologies during enhancements.
- **Portability:**
The platform will be accessible through all modern web browsers and devices, including desktops, laptops, tablets, and smartphones. The architecture supports future deployment on cloud platforms.

Chapter 3: Use Case Analysis

There are following use Case diagrams of Functional requirements:

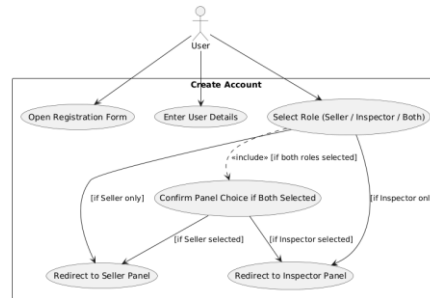


Figure 1 use case for Create Account

Usecase diagram detail:

Use Case ID	UC_01	
Use Case Name	Create Account	
Description	This use case describes the process of registering a new user in the Prime Innovative Hub real estate platform, either as a Seller, Inspector, or both.	
Primary Actor	User	
Secondary Actor	None	
Pre-Condition	The user must have internet access and reach the Sign-Up page via the Login screen. The user has not registered on the system previously.	
Post-Condition	A new user account is created and saved in the database The user is redirected based on the selected role(s): Seller Dashboard, Inspector Application, or both.	
Basic Flow	Actor Action	System Action
	1) The user opens the Login page and clicks on "Sign Up". 2) The user fills in personal information (e.g., name, email, password). 3) The user selects their role(s) via checkboxes (Seller, Inspector, or both). 4) The user submits the registration form.	1) The system displays the Sign-Up form. 2) The system validates the input data. 3) The system creates a new user account and stores it in the database. 4) If one role is selected, redirect the user to the corresponding dashboard (Seller or Inspector Application). 5) If both roles are selected, the system prompts the user to choose which dashboard to proceed to.
Alternate Flow	Actor Action: The user enters an already registered email. System Action: The system shows an error message: "Email already in use. Please log in or use a different email."	

Table 13 use case for Create Account

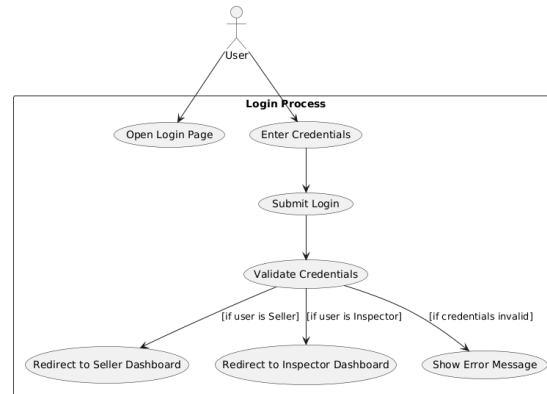


Figure 2 use case for Login Process

Use Case ID	UC_02	
Use Case Name	Login Process	
Description	This use case describes how a registered user logs into the Prime Innovative Hub system and is redirected to their respective dashboard based on their role.	
Primary Actor	User	
Secondary Actor	None	
Pre-Condition	The user must have a registered account in the system. The user must be on the Login page with access to internet.	
Post-Condition	- If credentials are valid, the user is redirected to either the Seller or Inspector dashboard based on their role. - If credentials are invalid, an error message is shown.	
Basic Flow	Actor Action	System Action
	1) The user opens the Login page. 2) The user enters login credentials (email and password). 3) The user submits the login form.	1) The system receives the credentials and validates them. 2) If credentials are correct: <ul style="list-style-type: none"> - If the user is a Seller, redirect to the Seller Dashboard. - If the user is an Inspector, redirect to the Inspector Dashboard. 3) If credentials are invalid, display an error message.
Alternate Flow	Actor Action: The user enters incorrect credentials. System Action: The system displays an error message: "Invalid email or password."	

Table 14 use case for Login Process

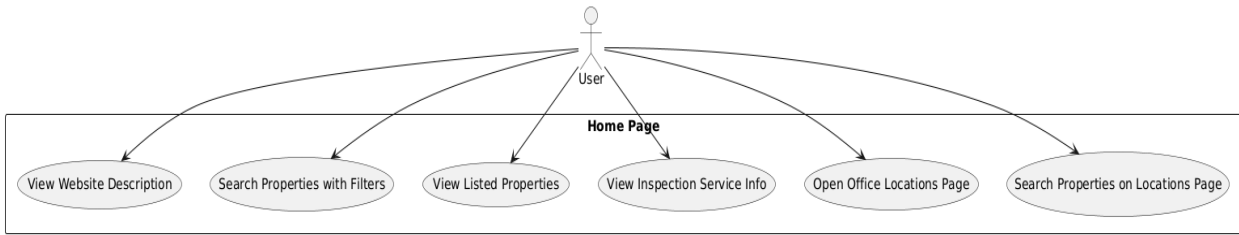


Figure 3 use case for home page

Use Case ID	UC_03	
Use Case Name	Access Home Page Features	
Description	This use case describes the various interactions a user can perform on the Home Page of the Prime Innovative Hub platform.	
Primary Actor	User	
Secondary Actor	None	
Pre-Condition	The user has accessed the website's home page with an internet-connected device.	
Post-Condition	The user is able to explore different sections such as property filters, listings, inspection info, and location-based property searches.	
Basic Flow	Actor Action	System Action
	1) The user opens the Home Page. 2) The user can: <ul style="list-style-type: none"> - View the website description. - Search properties using filters (city, location, price, buy/rent). - View a list of featured or latest properties. - View information about inspection services offered. - Open the office locations page for inspection availability. - Search properties from the office locations page. 	- Loads the requested content dynamically based on the user's actions (e.g., display property listings, filter results, show office maps).
Alternate Flow	Actor Action: The user applies a filter that yields no results. System Action: Displays a message like "No properties found for the selected criteria." and prompts the user to adjust filters.	

Table 15 use case for Home Page

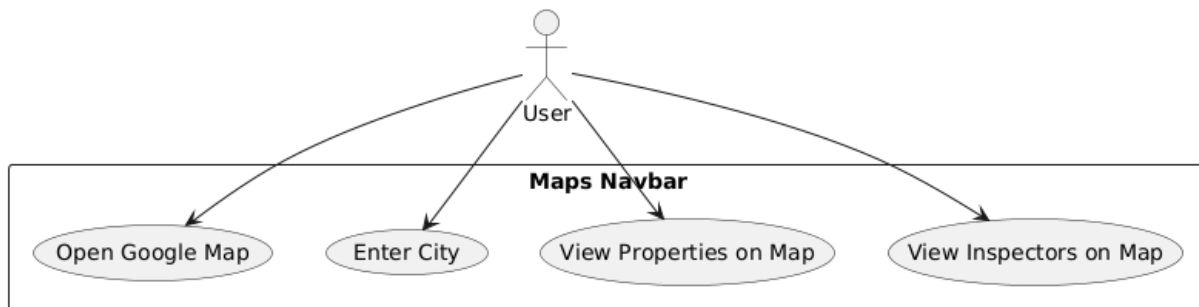


Figure 4 use case for maps navbar

Use Case ID	UC_04	
Use Case Name	View Properties and Inspectors on Map	
Description	This use case describes how a user interacts with the Maps section to view properties and available inspectors within a selected city using Google Maps integration.	
Primary Actor	User	
Secondary Actor	None	
Pre-Condition	<ul style="list-style-type: none"> - The user has internet access and navigates to the Maps section. - Google Maps API is successfully loaded. 	
Post-Condition	<ul style="list-style-type: none"> - The system displays a map centered on the selected city. - Property and inspector markers are shown on the map based on availability. 	
Basic Flow	Actor Action	System Action
	1) The user navigates to the Maps navbar. 2) The user clicks to open Google Map. 3) The user enters a city in the search box. 4) The user can toggle to view either properties or inspectors.	1) Google Map loads within the application. 2) The system centers the map based on the entered city. 3) The system fetches and displays properties and inspector locations as map markers. 4) Marker info displays brief details (name, address, contact) on click.
Alternate Flow	Actor Action: User enters an invalid city or there are no results for that location. System Action: Displays a message like “No properties or inspectors found in this city. Try another location.”	

Table 16 use case for Maps Navigation

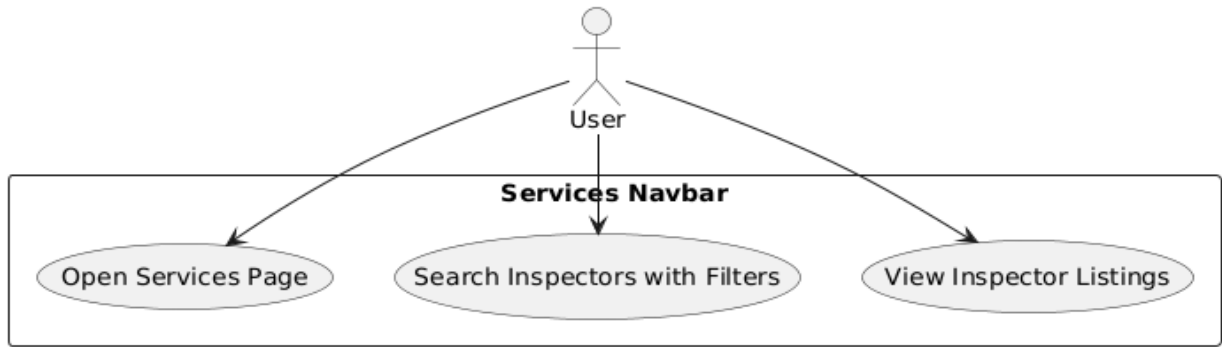


Figure 5 use case for Services

Use Case ID	UC_05	
Use Case Name	Explore Inspector Services	
Description	This use case describes how a user interacts with the Services section to view and filter inspector service listings based on city, location, and price.	
Primary Actor	User	
Secondary Actor	None	
Pre-Condition	<ul style="list-style-type: none"> - The user is on the Services page. - The inspector(s) have published services. 	
Post-Condition	<ul style="list-style-type: none"> - The system displays available inspectors and their services based on selected filters. 	
Basic Flow	Actor Action	System Action
	1) The user clicks the “Services” navbar item. 2) The user sees a service search bar. 3) The user applies filters such as city, price, and availability. 4) The user views a list of available inspectors and their service details.	1) Loads the Services page with the search interface. 2) Fetches inspectors from the database who have approved services. 3) Filters the inspector listings based on user inputs. 4) Displays filtered inspector listings with details like location, service type, slots, and contact options.
Alternate Flow	Actor Action: The user applies filters with no matching inspectors. System Action: Displays a message like: “No inspectors found matching your criteria.” and prompts to reset filters.	

Table 17 use case for Services

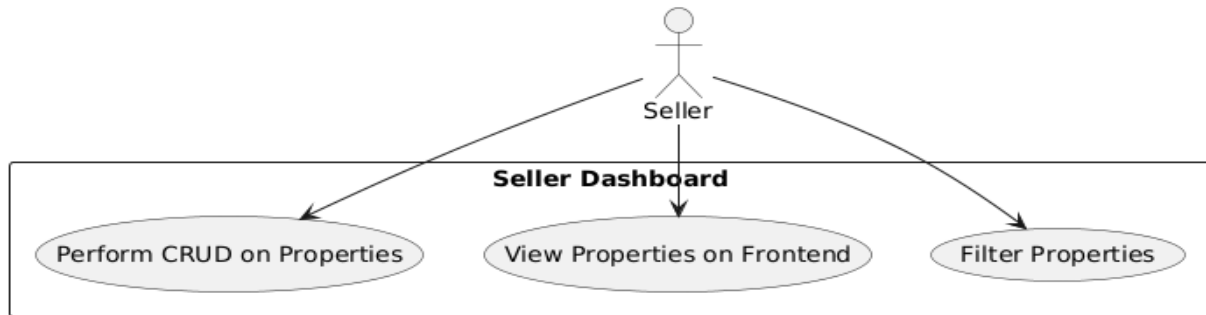


Figure 6 use case for seller dashboard

Use Case ID	UC_06	
Use Case Name	Manage Seller Dashboard	
Description	This use case describes how a Seller manages their dashboard to create, update, delete, and view their property listings, as well as filter and preview them on the frontend.	
Primary Actor	Seller	
Secondary Actor	None	
Pre-Condition	<ul style="list-style-type: none"> - The seller is logged into their account. - Seller has an active account approved by the system. 	
Post-Condition	<ul style="list-style-type: none"> - Properties are successfully managed in the system and reflected on the frontend for users. 	
Basic Flow	Actor Action	System Action
	1) The seller logs into the dashboard. 2) Seller creates a new property listing with relevant details. 3) Seller edits or deletes existing properties. 4) Seller views the list of their properties. 5) Seller applies filters (e.g., city, price) to find specific properties. 6) Seller previews how their listings appear on the frontend.	1) Displays the seller dashboard with all tools. 2) Saves new or updated property data to the database. 3) Reflects changes on the public listing view. 4) Filters and returns relevant property data.
Alternate Flow	<p>Actor Action: Seller enters incomplete property data.</p> <p>System Action: Prompts the seller to fill all required fields before submission.</p> <p>Actor Action: Seller tries to delete a property that doesn't exist.</p> <p>System Action: Displays: "Property not found or already deleted."</p>	

Table 18 use case for seller dashboard

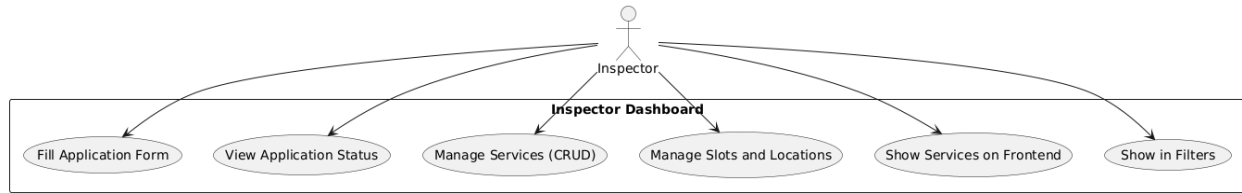


Figure 7 use case for inspector Dashboard

Use Case ID	UC_07	
Use Case Name	Manage Inspector Dashboard	
Description	This use case describes how an Inspector interacts with their dashboard to apply for approval, manage their service offerings, schedule slots, set locations, and ensure visibility on the frontend and in filters.	
Primary Actor	Inspector	
Secondary Actor	Admin (indirectly, for approval-related steps)	
Pre-Condition	<ul style="list-style-type: none"> - The inspector is logged in. - Inspector is either applying or already approved. 	
Post-Condition	<ul style="list-style-type: none"> - Services and availability are updated. - Inspector appears in frontend listings and filter results. 	
Basic Flow	Actor Action	System Action
	<ol style="list-style-type: none"> 1) Inspector fills out the application form. 2) Inspector checks the approval status. 3) If approved, inspector accesses dashboard. 4) Inspector adds/updates/deletes services (CRUD). 5) Inspector manages available time slots and service areas (locations). 6) Inspector previews services on frontend. 7) Inspector ensures their service appears in filters for users. 	<ol style="list-style-type: none"> 1) Saves application and sends it for admin review. 2) Displays status (Pending, Approved, Rejected). 3) If approved, unlocks service management tools. 4) Saves and updates services, time slots, and locations in the database. 5) Reflects these changes on frontend listings and user filters.
Alternate Flow	<p>Actor Action: Inspector fills an incomplete application. System Action: Prompts the user to fill all required fields.</p> <p>Actor Action: Admin rejects application. System Action: Displays a rejection message on next login.</p> <p>Actor Action: Inspector creates a service without a time slot. System Action: Prompts to add time slot before making service visible.</p>	

Table 19 use case for inspector dashboard

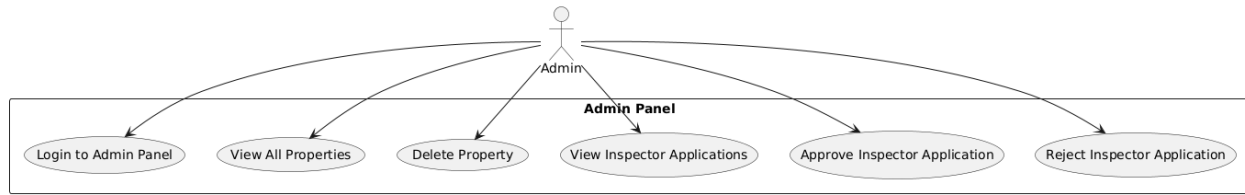


Figure 8 use case for Admin Panel

Use Case ID	UC_08	
Use Case Name	Manage Admin Panel	
Description	This use case describes how an Admin logs into the system, views and manages all listed properties, and handles inspector applications by approving or rejecting them.	
Primary Actor	Admin	
Secondary Actor	None	
Pre-Condition	<ul style="list-style-type: none"> - Admin has valid credentials. - Admin accesses the Admin Panel via login. 	
Post-Condition	Admin manages property data and inspector status, which reflects on the frontend and in user dashboards.	
Basic Flow	Actor Action	System Action
	1) Admin logs into the Admin Panel using predefined credentials. 2) Admin views all listed properties. 3) Admin can delete any listed property. 4) Admin views all pending inspector applications. 5) Admin either approves or rejects each application.	1) Validates login credentials. 2) Displays the list of all properties and inspector applications. 3) Deletes selected property from the database and removes it from frontend. 4) Updates inspector status to "Approved" or "Rejected". 5) Notifies inspector of the status upon next login.
Alternate Flow	<p>Actor Action: Admin deletes an already-removed property. System Action: Displays: "Property not found or already deleted."</p> <p>Actor Action: Admin attempts to approve a malformed or incomplete inspector application. System Action: Displays an error: "Incomplete application — cannot proceed with approval."</p>	

Table 20 use case for Admin Panel

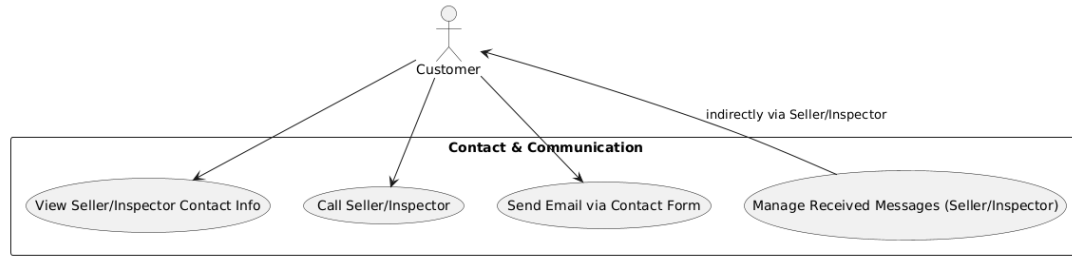


Figure 9 use case for contact & communication

Use Case ID	UC_09	
Use Case Name	Contact & Communicate with Seller/Inspector	
Description	This use case describes how a Customer communicates with a Seller or Inspector via contact details or a contact form. Sellers and Inspectors can then manage received inquiries.	
Primary Actor	Customer	
Secondary Actor	None	
Pre-Condition	<ul style="list-style-type: none"> - The customer is browsing a property or service listing. - Seller/Inspector has published valid contact info. 	
Post-Condition	<ul style="list-style-type: none"> - Customer successfully contacts seller/inspector via call or form. - Seller/Inspector receives and can manage the message. 	
Basic Flow	Actor Action	System Action
	<ol style="list-style-type: none"> 1) Customer views property or service detail page. 2) Customer sees the Seller/Inspector's contact number or email. 3) Customer may either: <ul style="list-style-type: none"> - Call the number directly. - Fill out a contact form (name, email, message). 4) Message is sent to the respective Seller or Inspector. 5) Seller/Inspector logs in and views received messages. 	<ol style="list-style-type: none"> 1) Displays relevant contact information on detail pages. 2) Sends email notification or stores form message in the system. 3) Makes the message available to Seller/Inspector in their dashboard.
Alternate Flow	<p>Actor Action: Customer submits a contact form with missing or invalid fields. System Action: Prompts: "Please fill in all required fields." Actor Action: Seller/Inspector does not respond. System Action: No direct consequence, message remains marked as "unread" or "pending."</p>	

Table 21 use case for Contact & communication

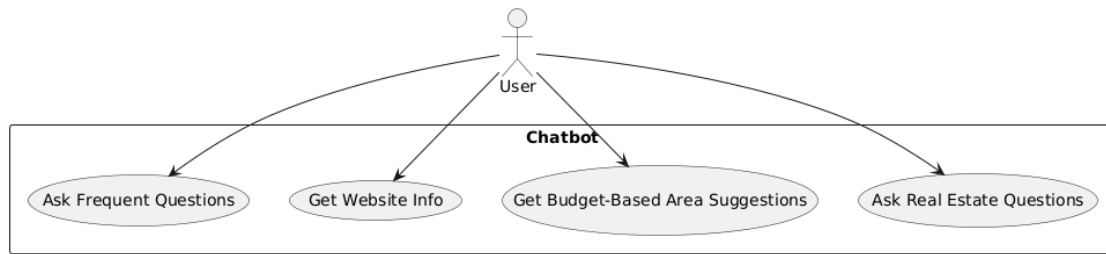


Figure 10 use case for Chatbot

Use Case ID	UC_10	
Use Case Name	Interact with Chatbot	
Description	This use case describes how a user interacts with the chatbot to ask questions about the website, real estate topics, and receive area suggestions based on their budget.	
Primary Actor	User	
Secondary Actor	None	
Pre-Condition	<ul style="list-style-type: none"> - The user is on the website and the chatbot widget is active. - The user has a query or needs assistance. 	
Post-Condition	<ul style="list-style-type: none"> - The user receives helpful answers or guidance. - User navigates accordingly or updates their search based on chatbot suggestions. 	
Basic Flow	Actor Action	System Action
	1) User clicks on the chatbot icon. 2) User may ask: <ul style="list-style-type: none"> - A frequently asked question (e.g., "How to contact seller?"). - For general website information (e.g., "What is this site about?"). - For area suggestions based on their budget (e.g., "I have 5M budget, where can I buy?"). - Any real estate-related question. 	1) Displays a chatbot interface and processes the user's message. 2) Responds with relevant information using AI or a rule-based engine. 3) May offer links, suggestions, or filtered content based on context (e.g., redirecting to property filters).
Alternate Flow	Actor Action: User asks a question the chatbot doesn't understand. System Action: Responds with fallback message: "I'm not sure about that. Please try rephrasing or contact support."	

Table 22 use case for Chatbot

Chapter 4: Design

In this section, we provide the design analysis of our modules including the following designs

1. Architecture Diagram
2. ERD with data dictionary
3. Data Flow diagram
4. Class Diagram
5. Activity Diagram
6. Sequence Diagram
7. Collaboration Diagram
8. State Transition Diagram
9. Component Diagram
10. Deployment Diagram

4.1 Architecture Diagram

Define the graphical representation of the concepts, their principles, elements and components that are part of your project.

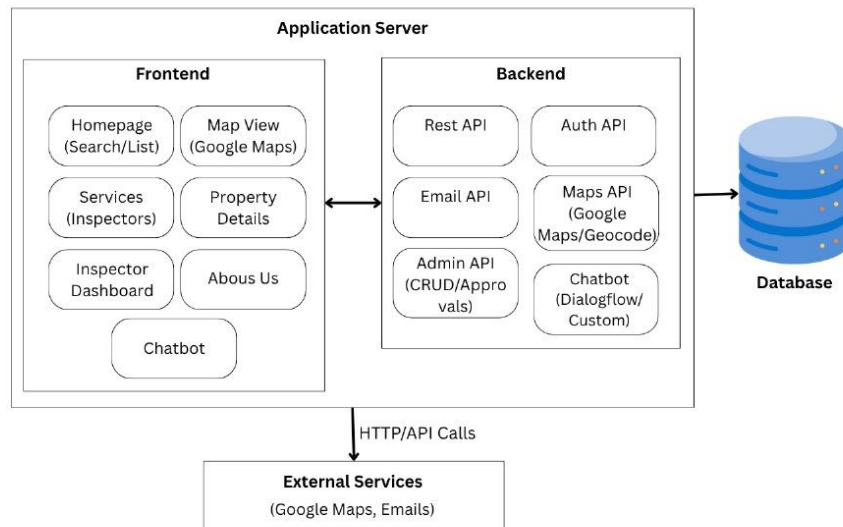


Figure 11 Architecture Diagram

4.1.1 ERD with data dictionary

Entity Relationship Diagram with complete relations with dependencies of your project

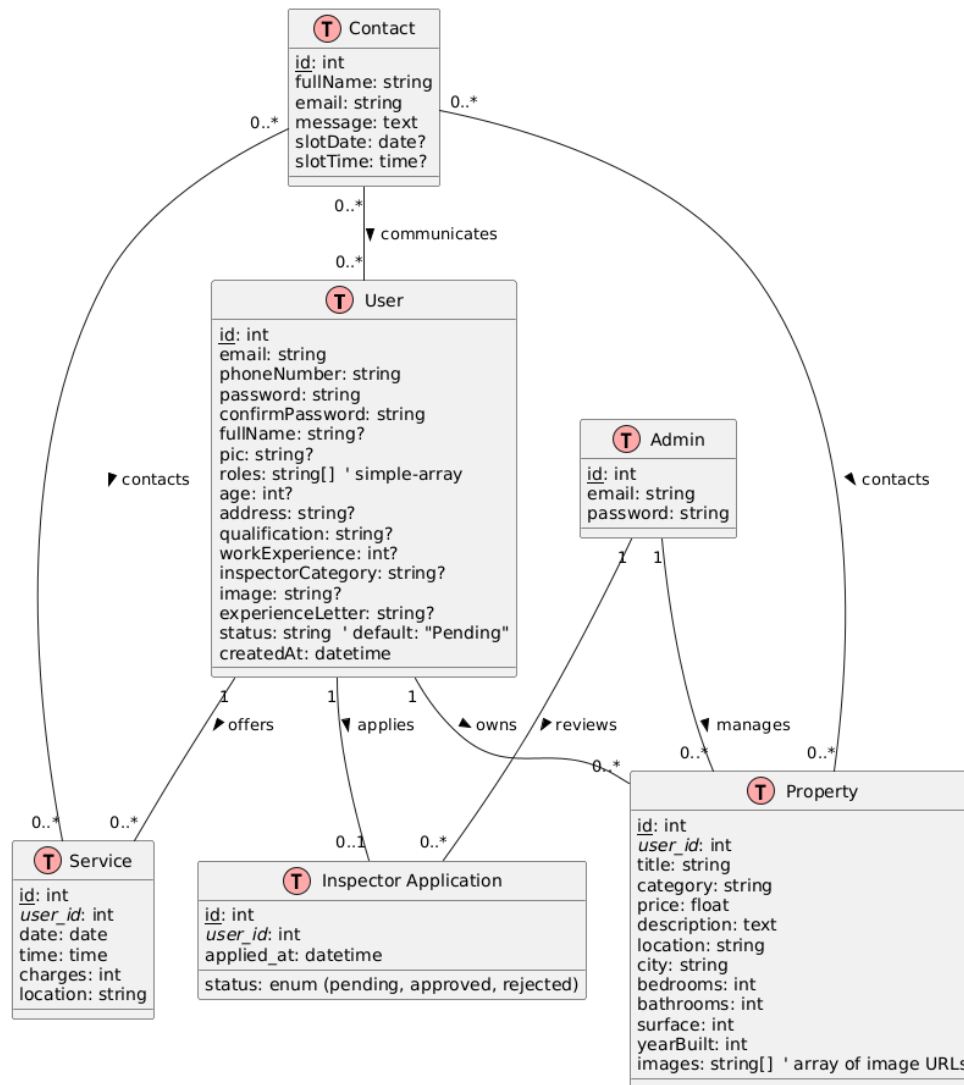


Figure 12 ERD

4.2 Data Flow diagram

Data flow diagram includes two levels

4.2.1 The level 0

The flow of information inside the system is defined in this level

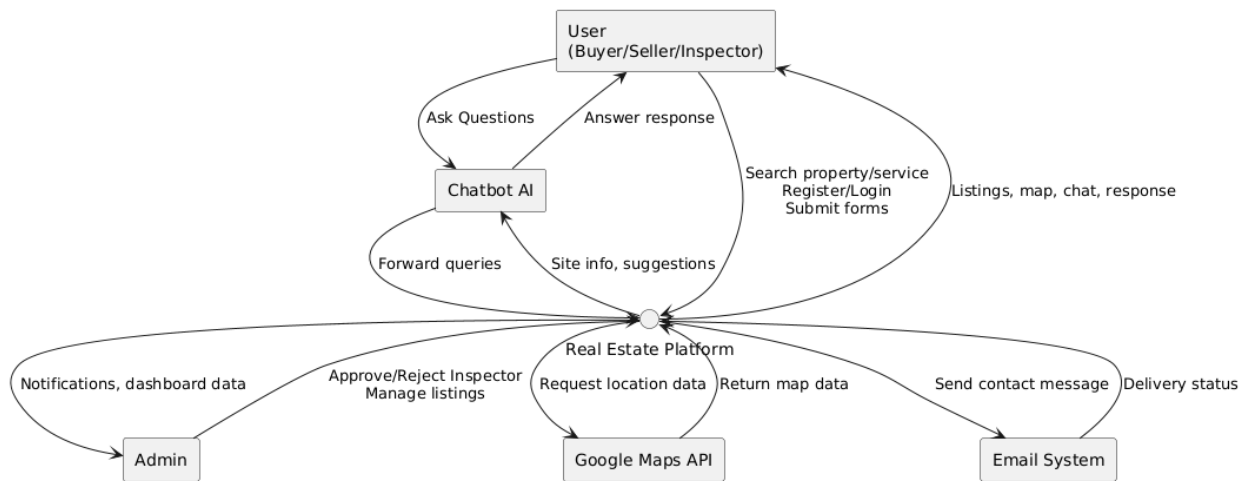


Figure 13 Level 0 DFD

4.2.2 The level 1

The flow of information outside the system is defined in this level

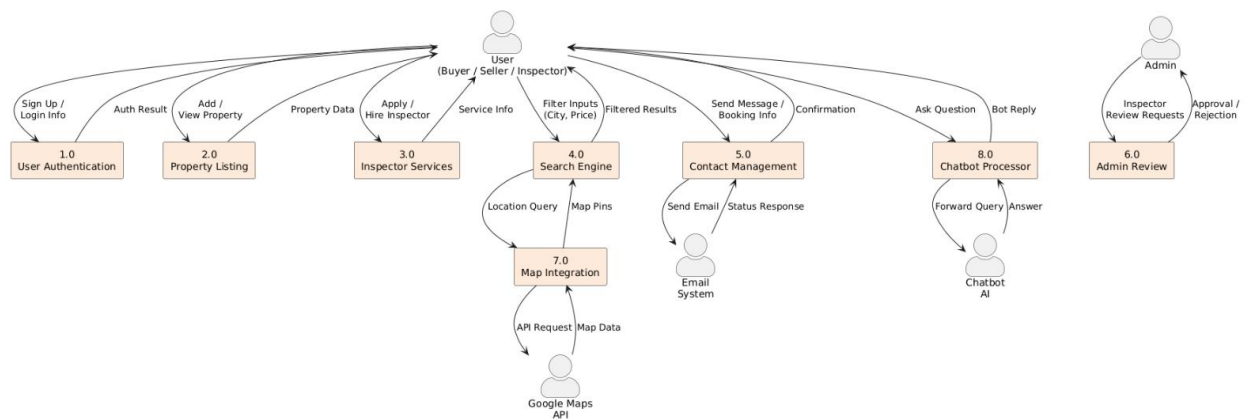


Figure 14 Level 1 DFD

4.3 Class Diagram

Describe the structure of a project by showing the systems classes, their attributes, operations (or methods), and the relationships among objects.

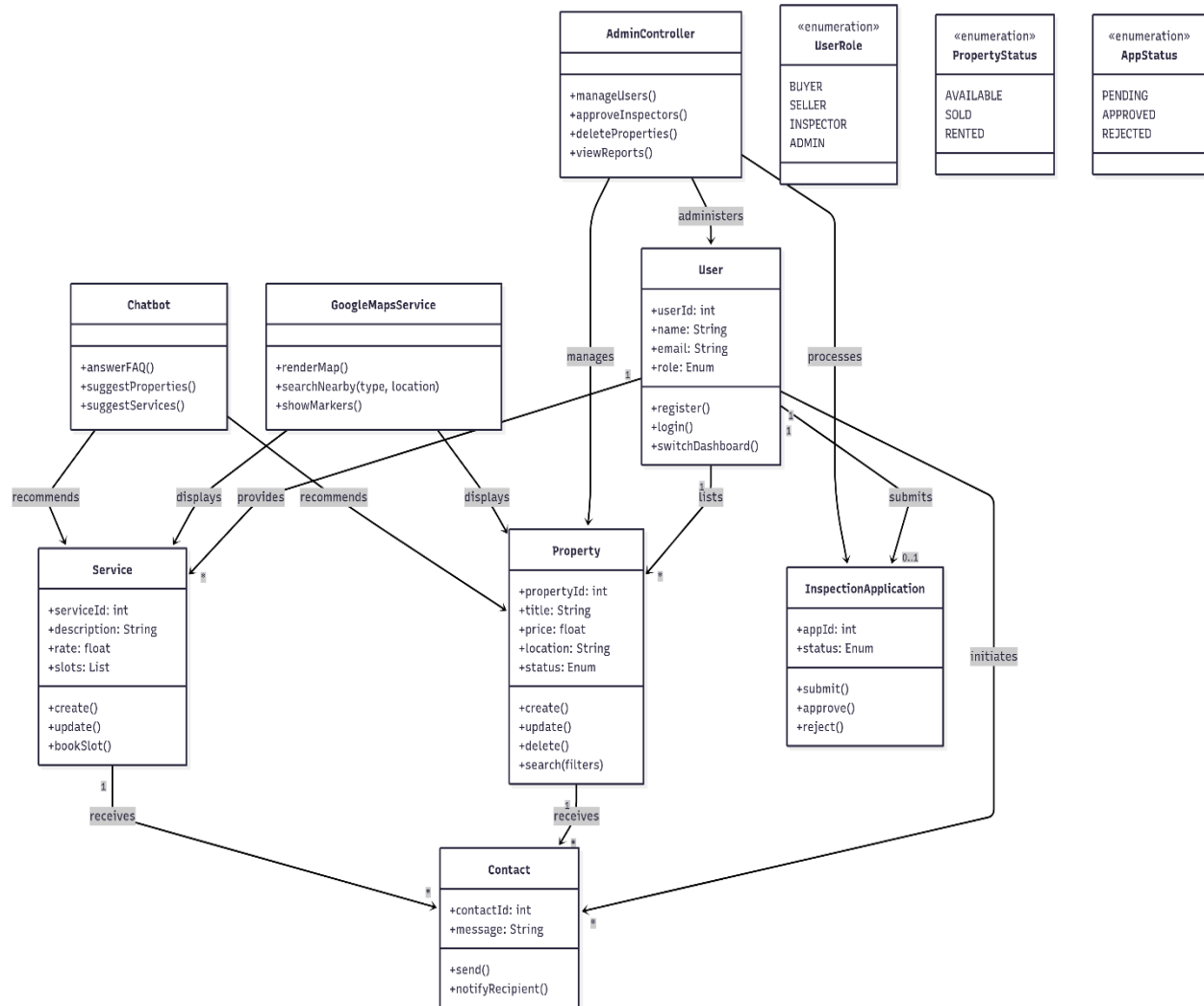


Figure 15 Class Diagram

4.4 Activity Diagram

This diagram includes all the activity diagrams of the functional requirements of your project along with the aggregated activity diagram

1. Create Account

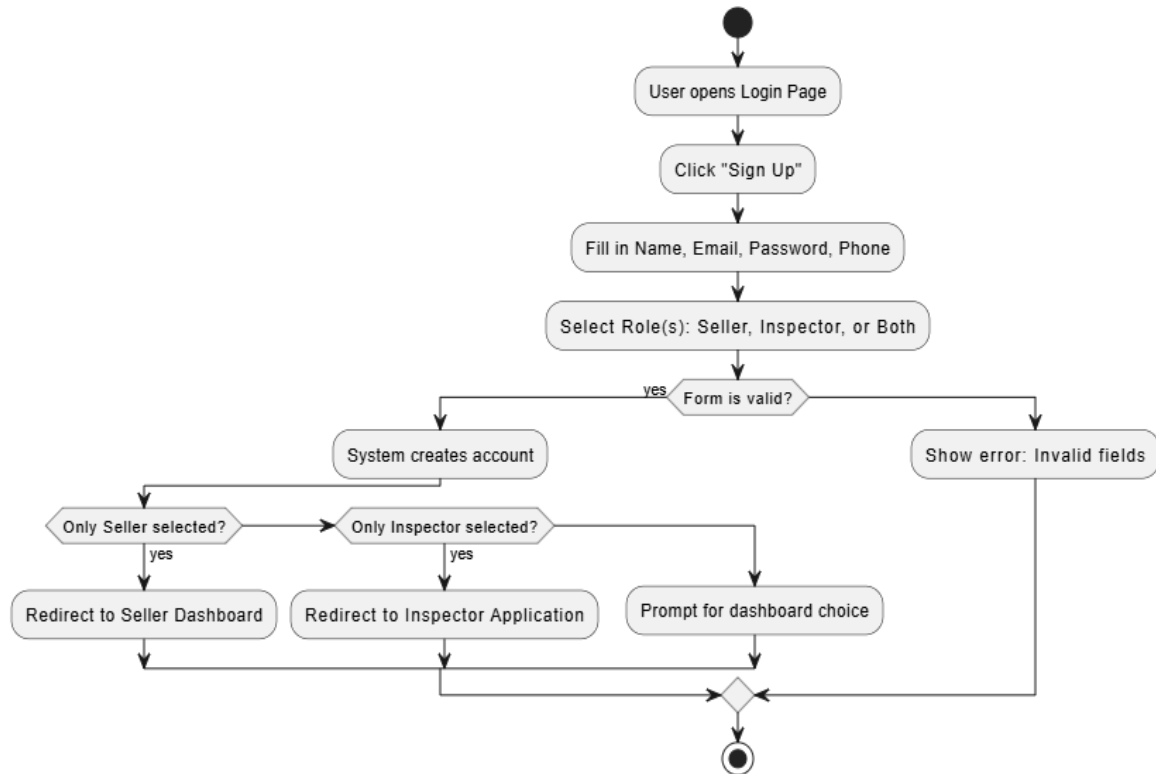


Figure 16 Activity Diagram Create Account

2. Login Process

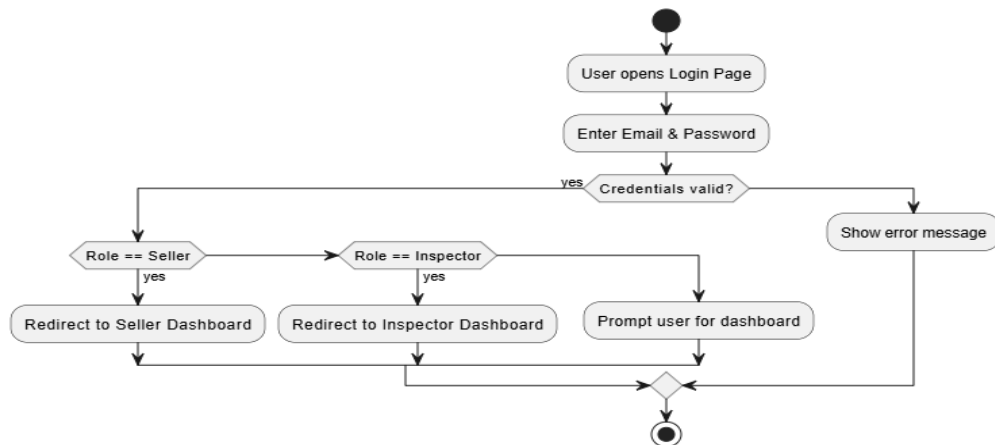


Figure 17 Activity Diagram Login Process

3. Add Property (Seller)

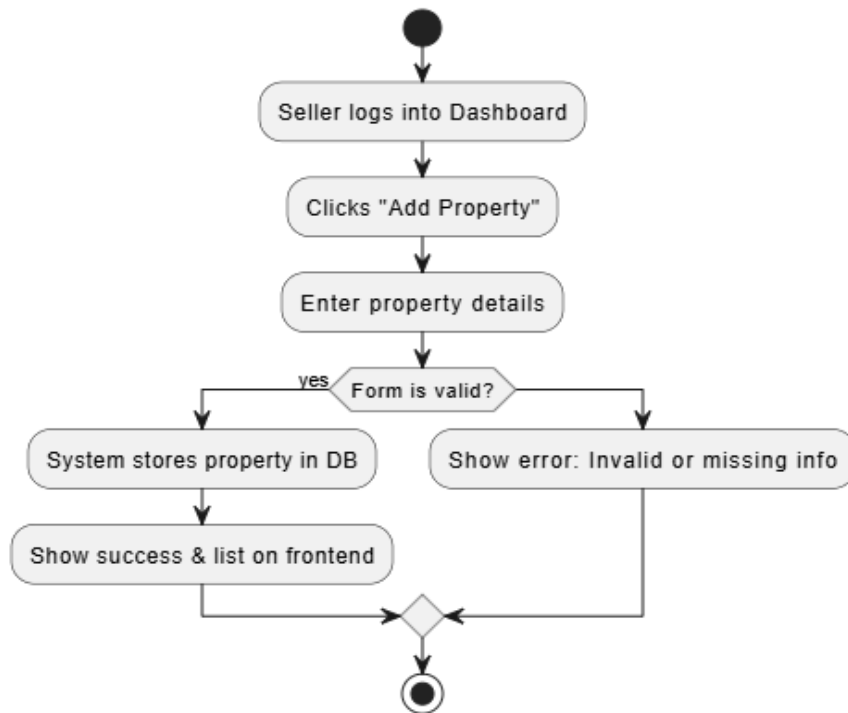


Figure 18 Activity Diagram Add Property (Seller)

4. Update/Delete Property (Seller)

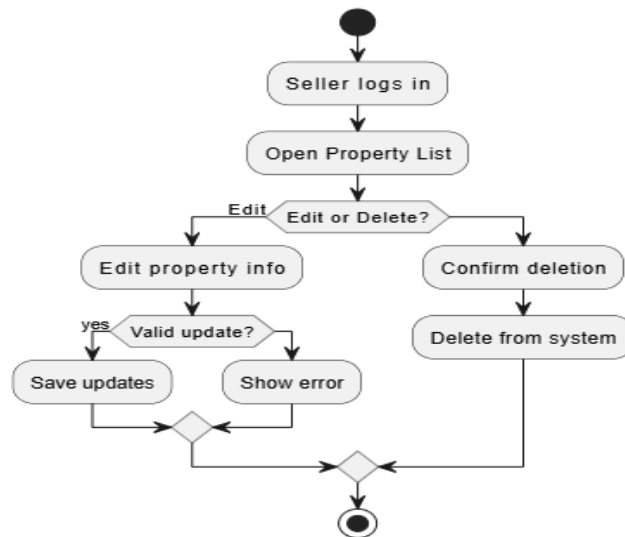


Figure 19 Activity Diagram Update or Delete Property (seller)

5. Submit Inspector Application

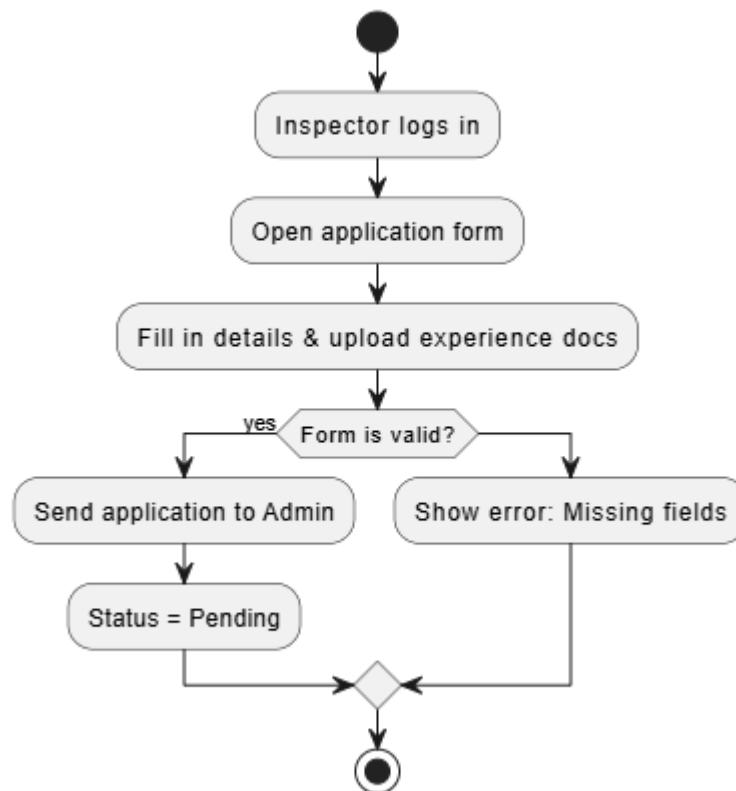


Figure 20 Activity Diagram Submit Inspector Application

6. Approve/Reject Inspector (Admin)

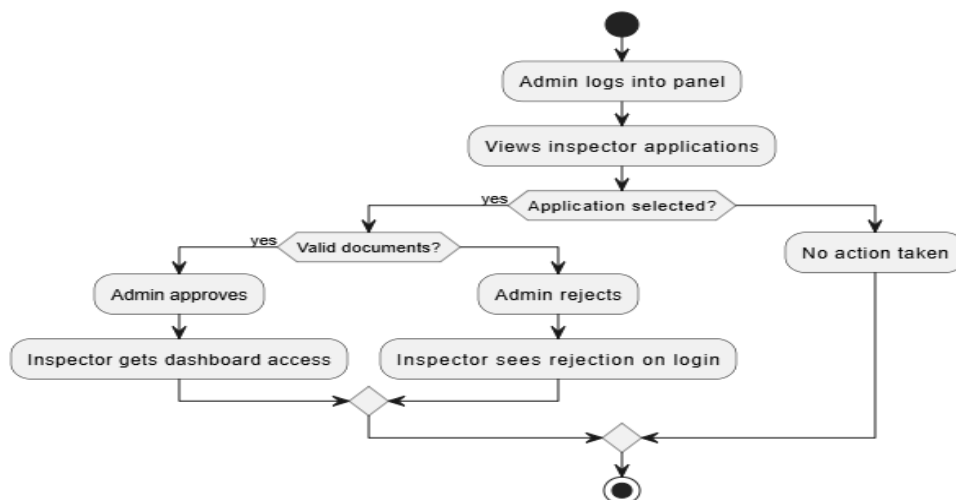


Figure 21 Activity Diagram Approve/Reject Inspector (Admin)

7. Manage Services (Inspector)

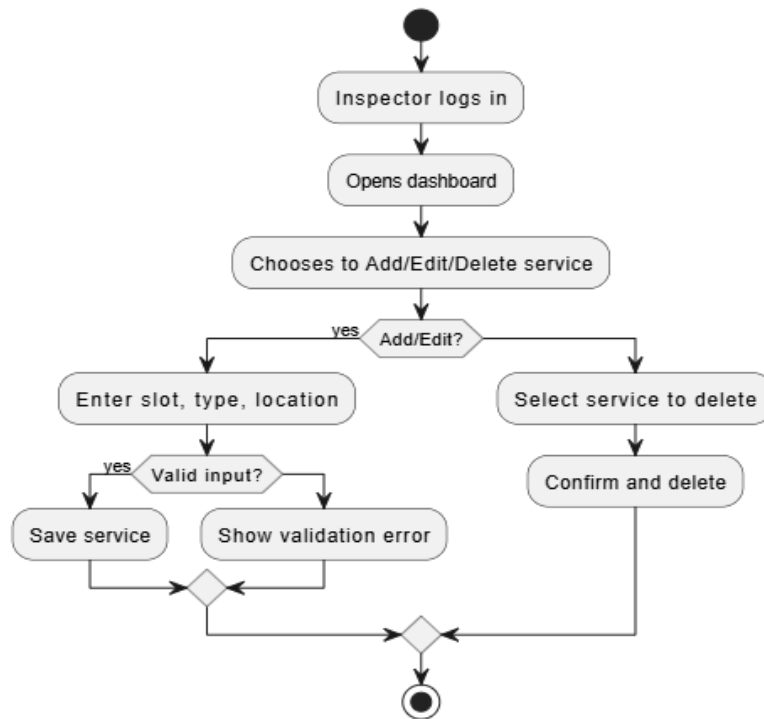


Figure 22 Activity Diagram Manage Services (Inspector)

8. Property/Inspector Search

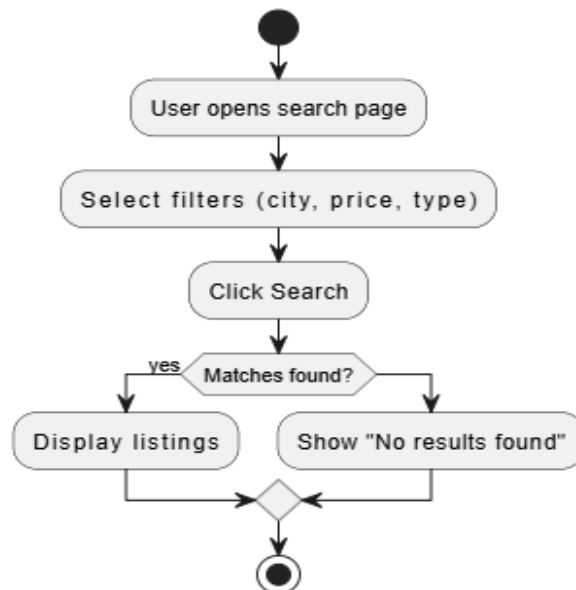


Figure 23 Activity Diagram Property/Inspector (CHECK)

9. View on Map

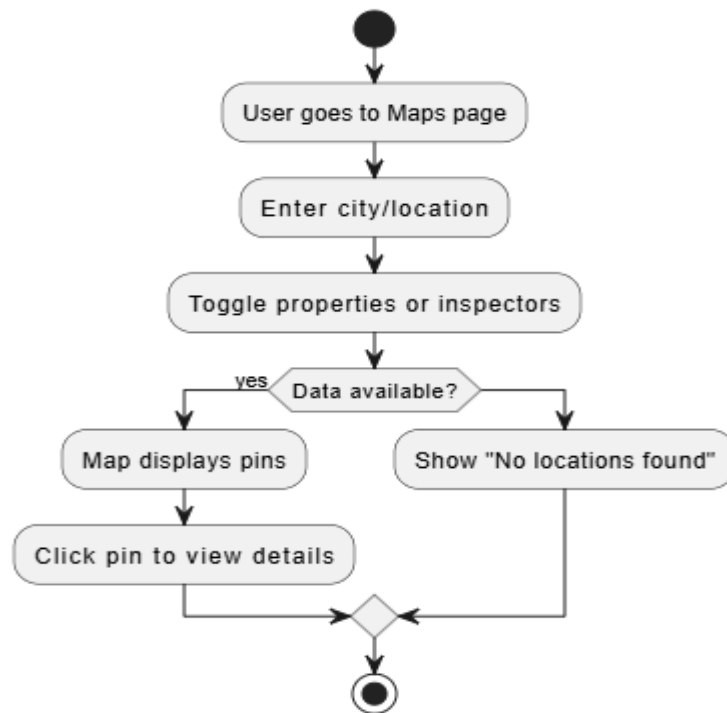


Figure 24 Activity Diagram View On Map

10. Use Chatbot

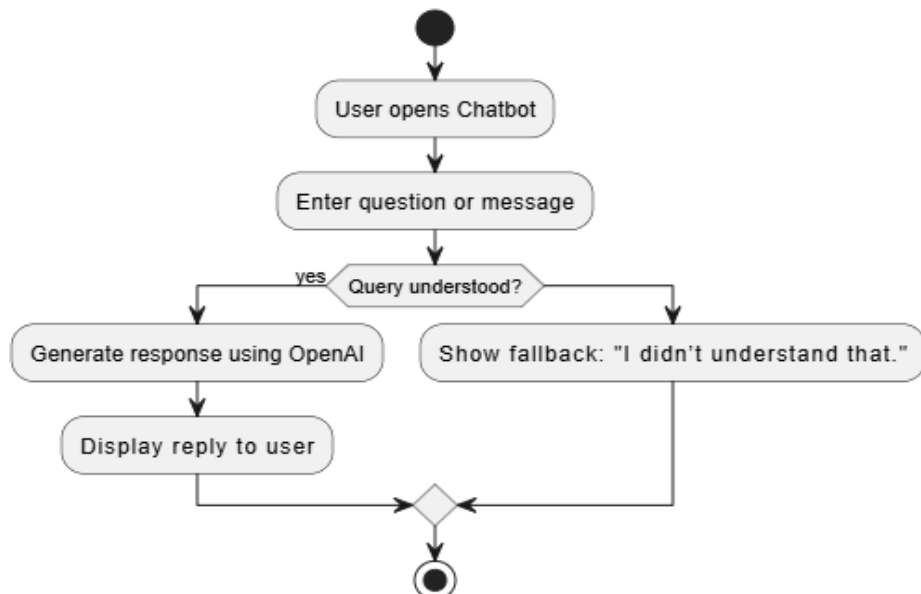


Figure 25 Activity Diagram Use ChatBot

4.5 Sequence Diagram

This diagram includes all the Sequence diagrams of the functional requirements of your project along with the aggregated Sequence diagram

1. Create Account

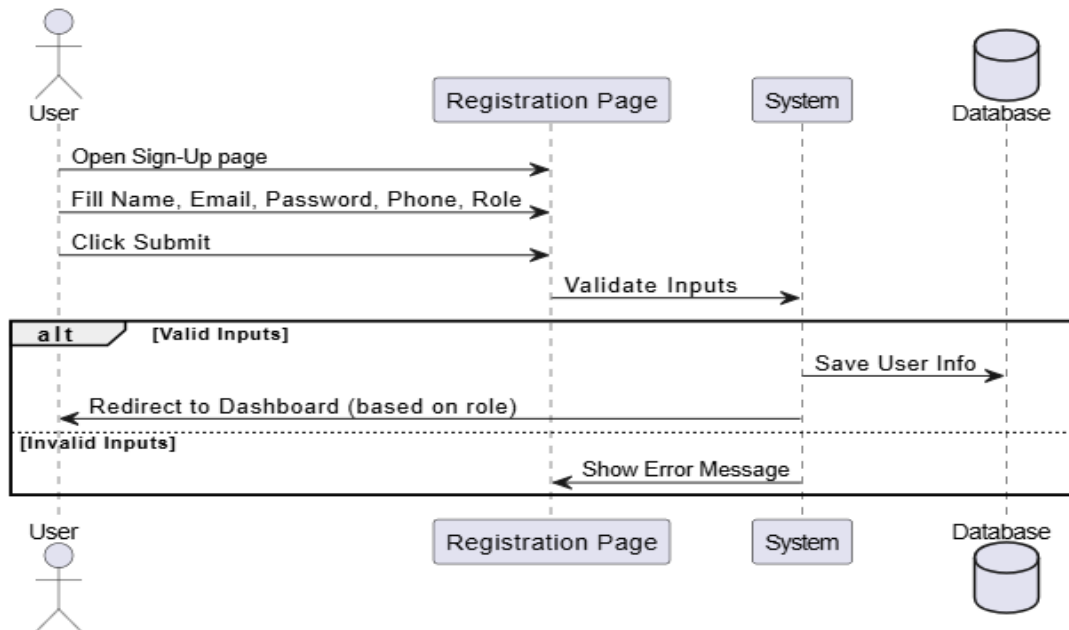


Figure 26 Sequence Diagram Create Account

2. Login Process

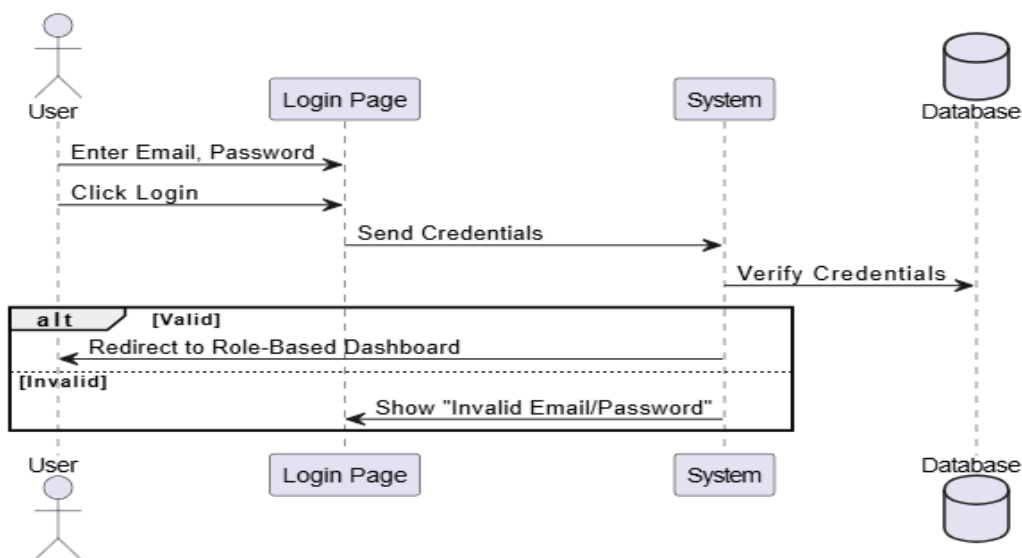


Figure 27 Sequence Diagram Create Account

3. Add Property (Seller)

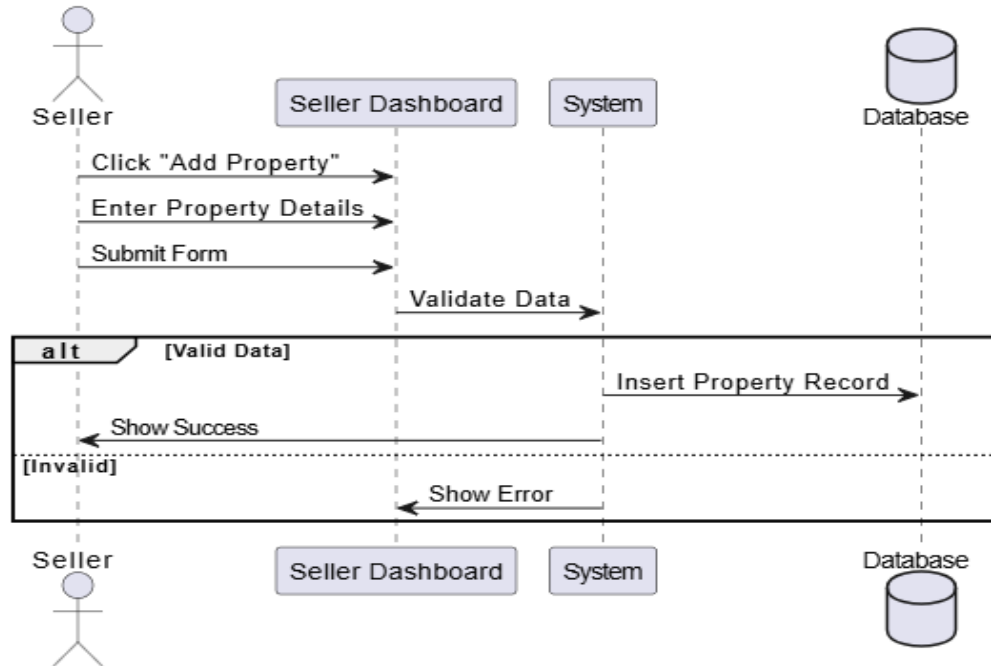


Figure 28 Sequence Diagram Add Property (seller)

4. Update/Delete Property

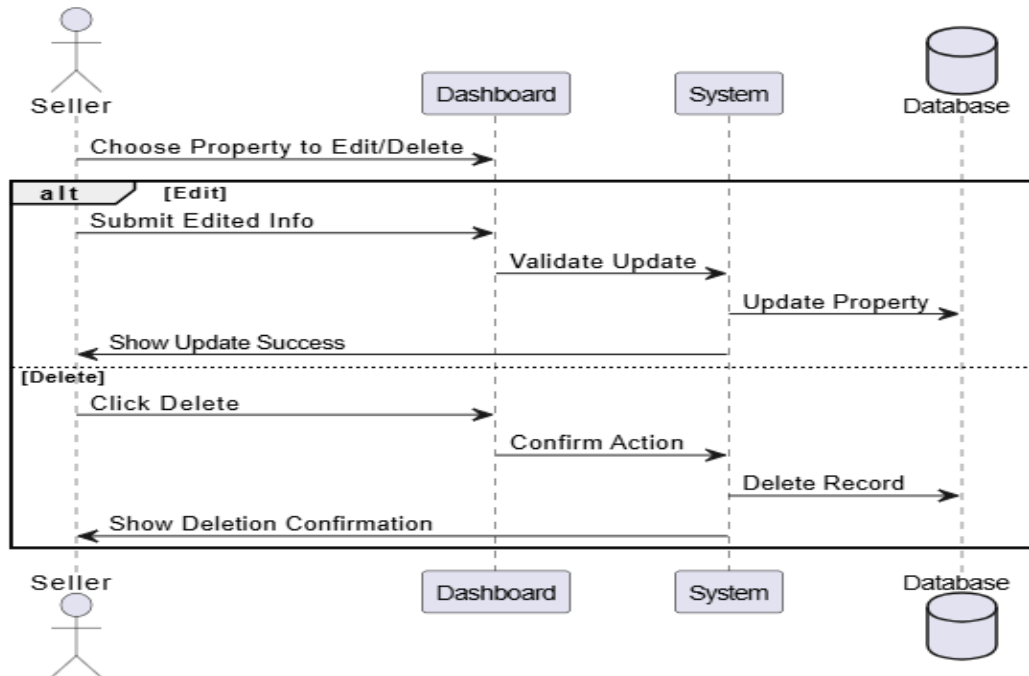


Figure 29 Sequence Diagram Update Delete Property

5. Submit Inspector Application

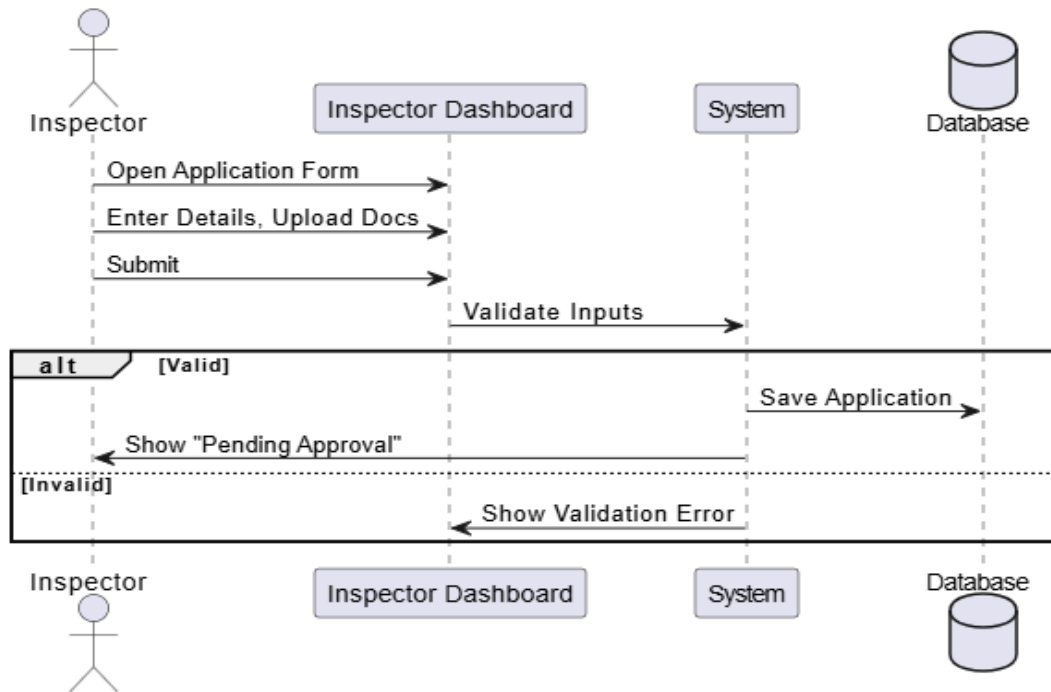


Figure 30 Sequence Diagram Submit Inspector Application

6. Admin Approves/Rejects Inspector

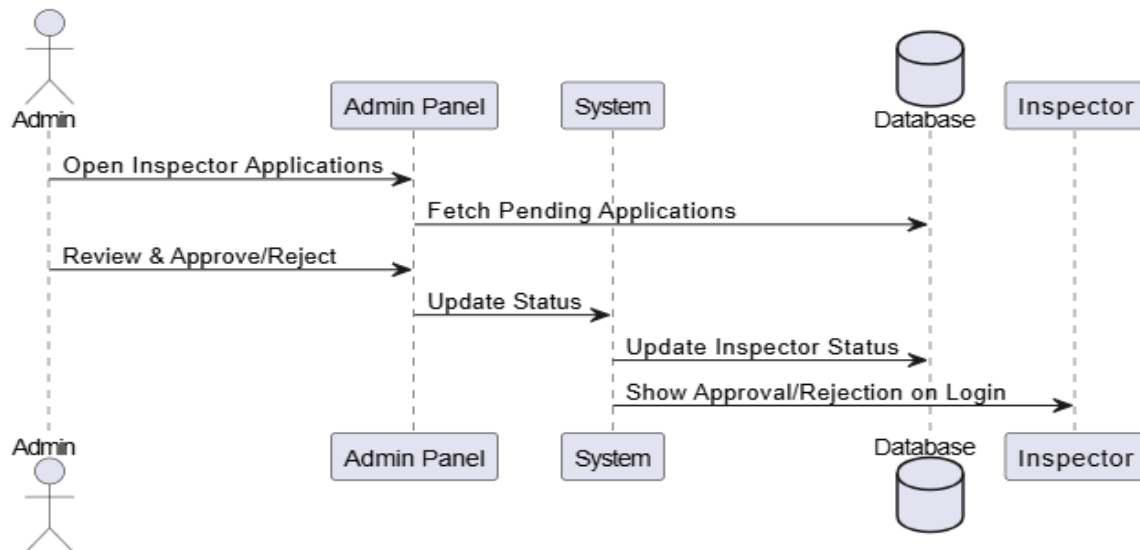


Figure 31 Sequence Diagram Admin Approve Reject Inspector

7. Inspector Adds/Manages Service

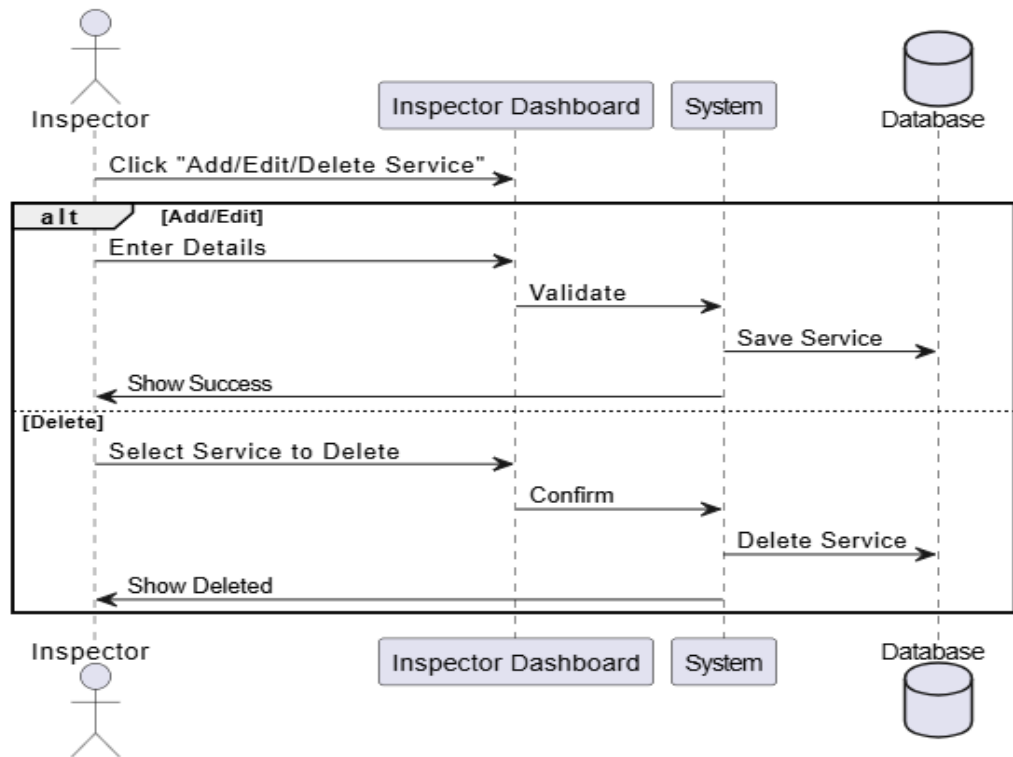


Figure 32 Sequence Diagram Inspector Adds/Manages Services

8. Property/Inspector Search

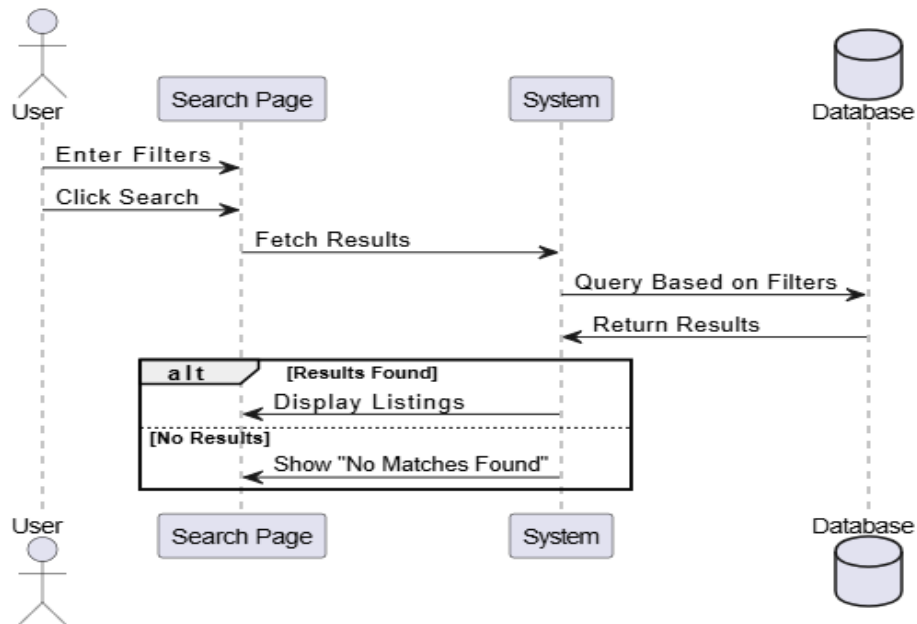


Figure 33 Sequence Diagram Property Inspector Check

9. View on Map (Google Maps Integration)

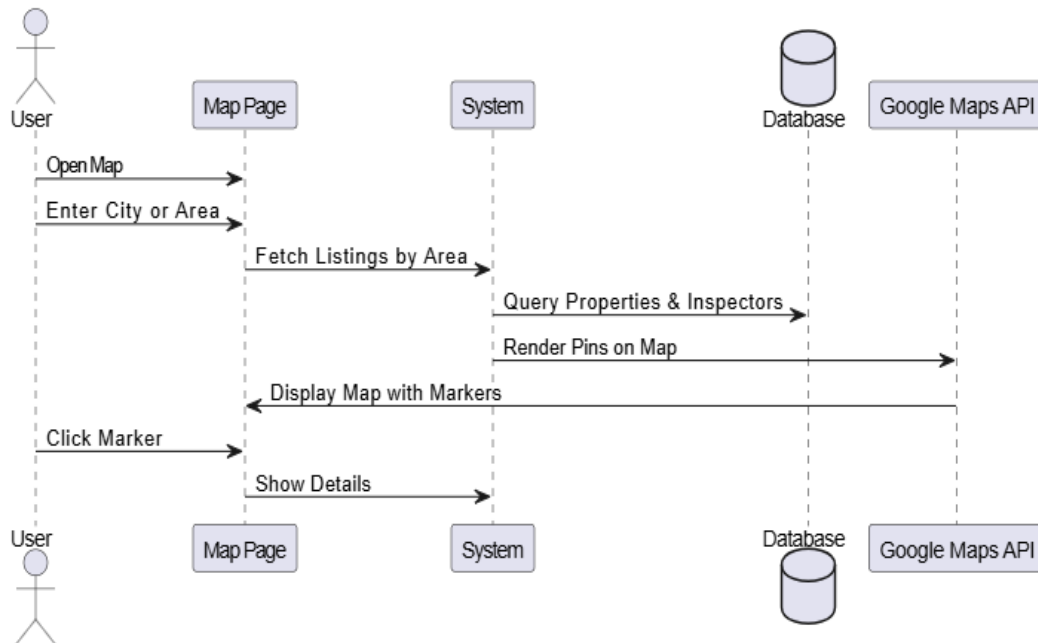


Figure 34 Sequence Diagram View On Maps

10. Use Chatbot

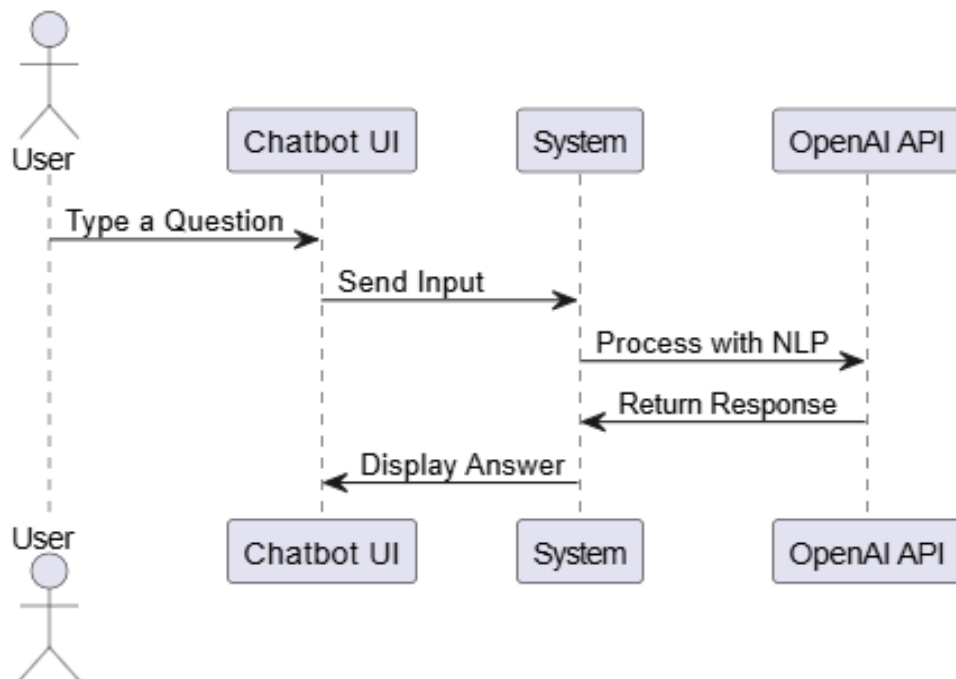


Figure 35 Sequence Diagram Use ChatBot

4.6 Collaboration Diagram

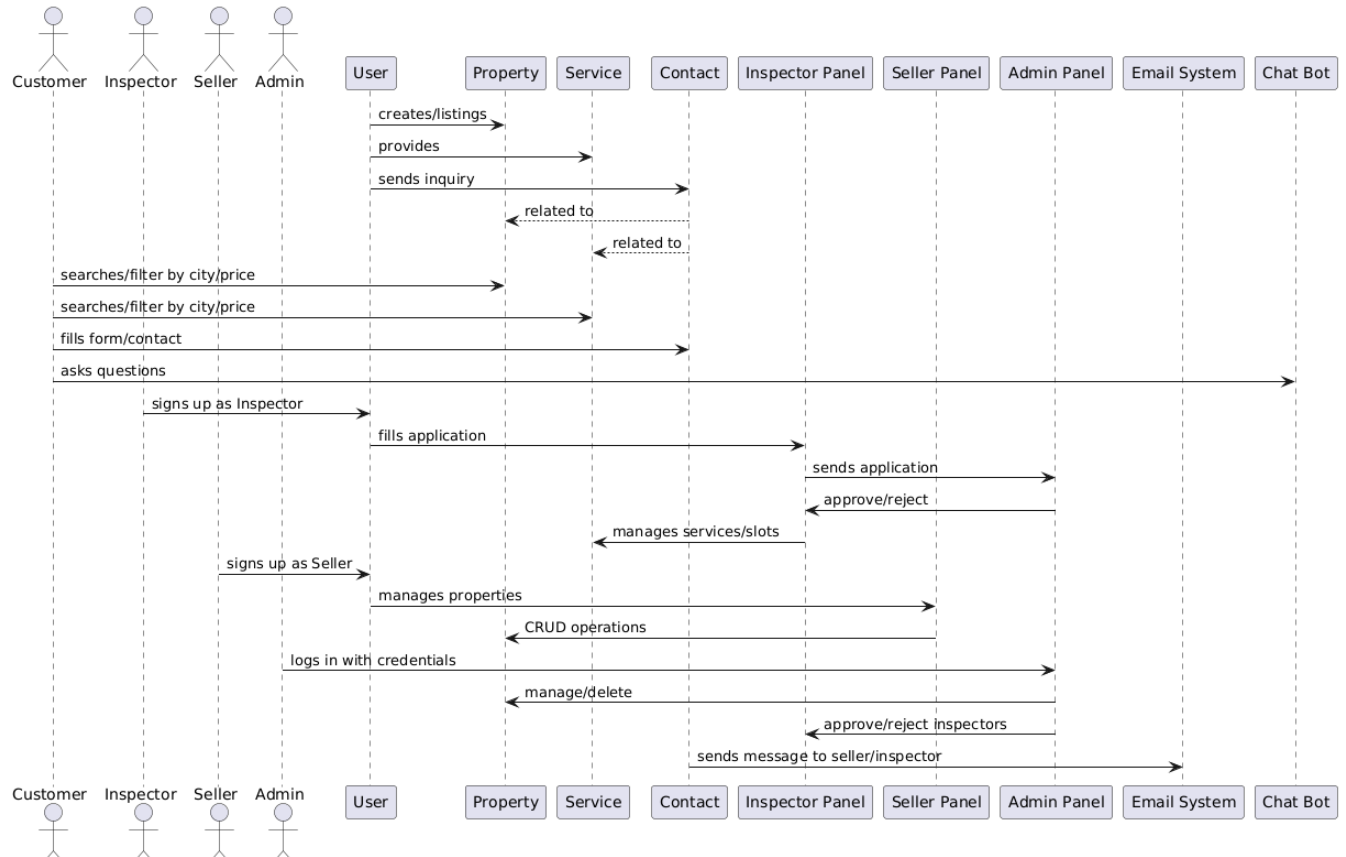


Figure 36 Collaboration Diagram

4.7 State Transition Diagram

State Transition diagram is used to describe the states of different objects in its life cycle. So, the emphasis is given on the state changes upon some internal or external events. These states of objects are important to analyze and implement them accurately

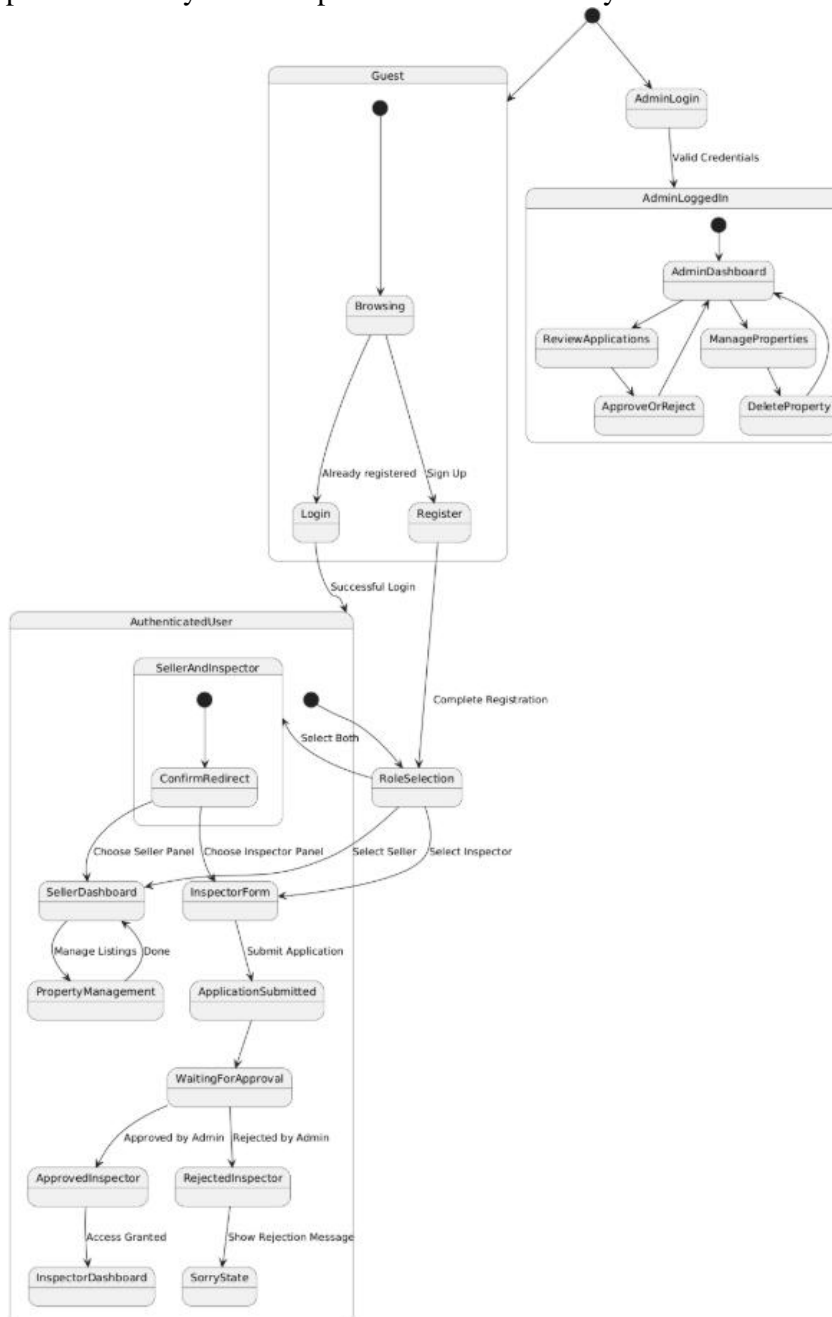


Figure 37 State Transition Diagram

4.8 Component Diagram

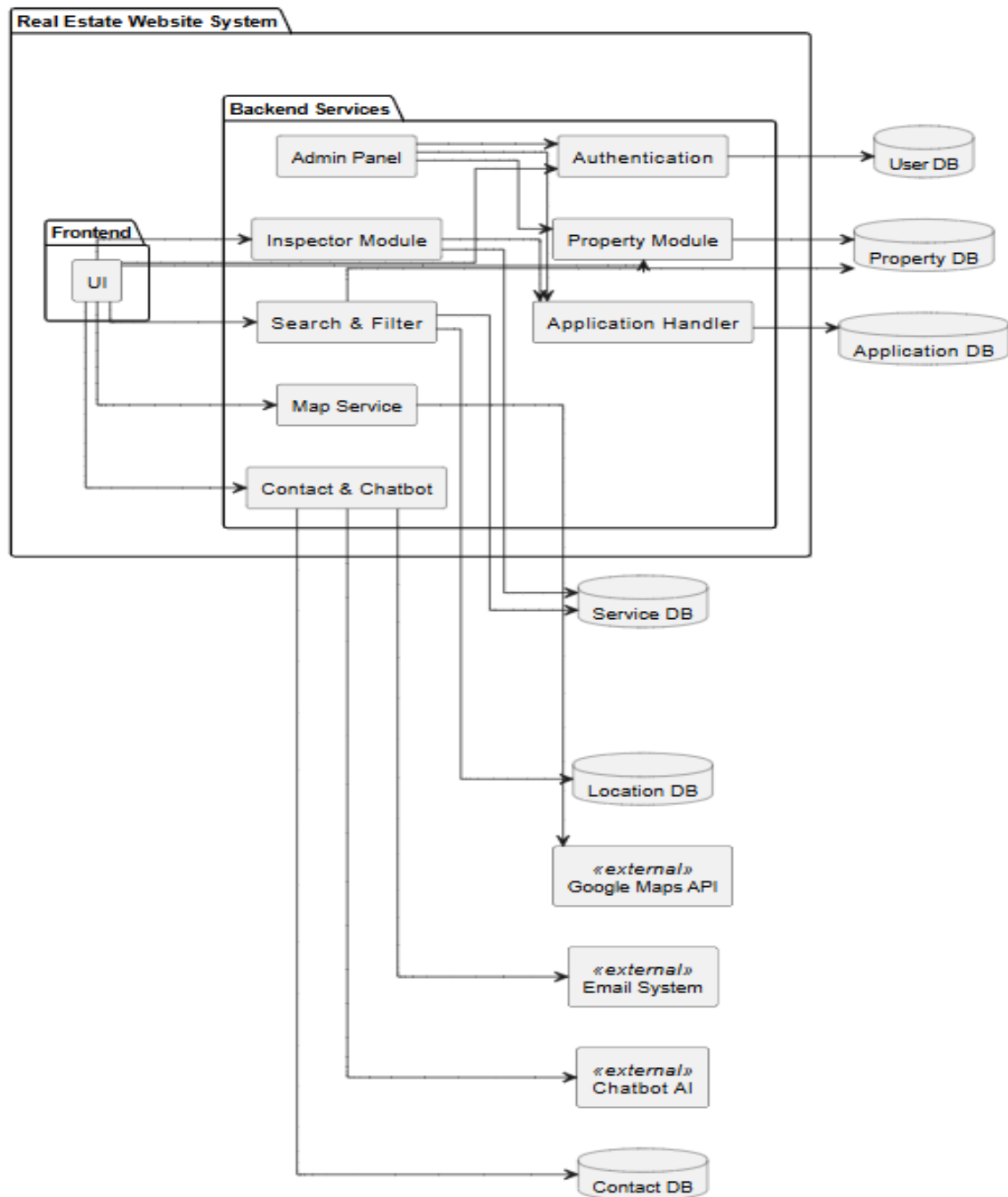


Figure 38 Component Diagram

4.9 Deployment Diagram

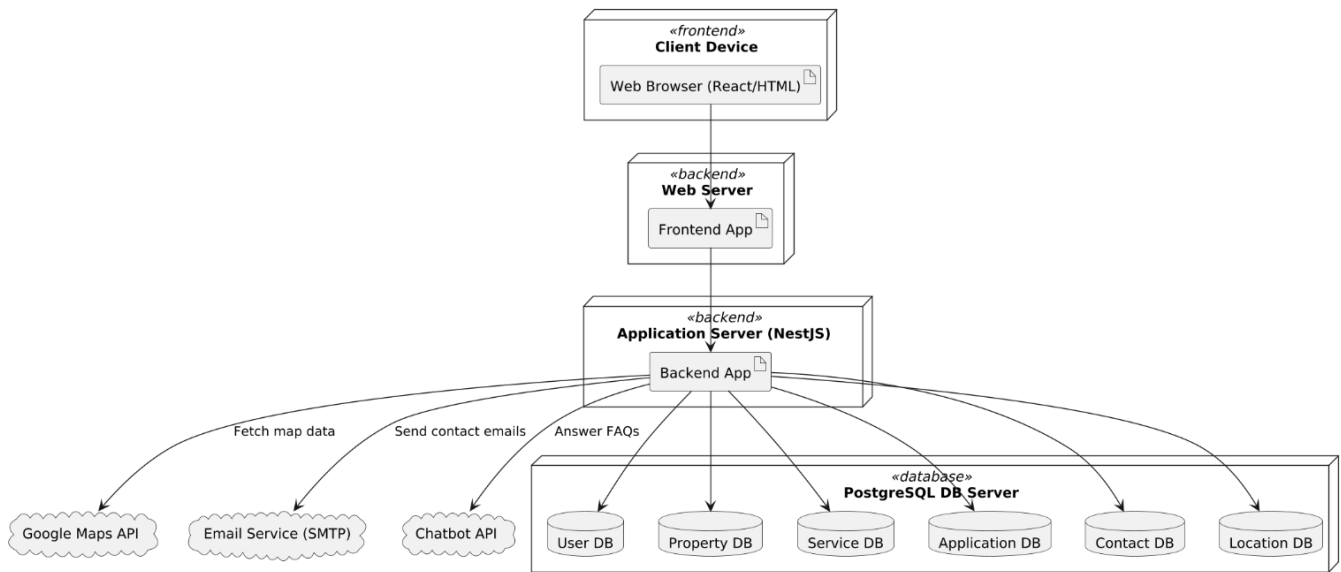


Figure 39 Deployment Diagram

Chapter 5: Testing

5.1 Test Case Specifications

Test Case – User Registration (Positive)

Field	Detail
ID	TC_USER_REGISTER_SUCCESS
Priority	High
Description	To verify successful registration as seller, inspector, or both.
Reference	FR_01: Create Account
Users	Buyers, Sellers, Inspectors
Pre-requisites	User must have internet access
Steps	Open registration form > Fill details > Select role(s) > Submit
Input	Name, email, password, phone, role selection
Expected Result	Account created and user redirected to relevant dashboard
Status	Tested, Passed

Table 23 User Registration Positive

Test Case – User Registration (Negative)

Field	Detail
ID	TC_USER_REGISTER_FAIL
Priority	High
Description	To verify registration fails with incomplete or invalid info.
Reference	FR_01: Create Account
Users	Buyers, Sellers, Inspectors
Pre-requisites	System online
Steps	Open form > Submit with invalid email or missing password
Input	Incomplete/invalid registration data
Expected Result	Error message shown, account not created
Status	Tested, Passed

Table 24 User Registration Negative

Test Case – User Login (Positive)

Field	Detail
ID	TC_USER_LOGIN_SUCCESS
Priority	High
Description	To verify login success with valid credentials
Reference	FR_02: Login Account
Users	All registered users
Pre-requisites	User account must exist
Steps	Open login page > Enter credentials > Submit
Input	Email and password
Expected Result	Redirected to respective dashboard
Status	Tested, Passed

Table 25 User Login Positive

Test Case – User Login (Negative)

Field	Detail
ID	TC_USER_LOGIN_FAIL
Priority	High
Description	To verify login fails with incorrect credentials
Reference	FR_02: Login Account
Users	All registered users
Pre-requisites	System online
Steps	Open login page > Enter wrong credentials > Submit
Input	Wrong email or password
Expected Result	Error shown, login denied
Status	Tested, Passed

Table 26 User Login Negative

Test Case – Property Add by Seller (Positive)

Field	Detail
ID	TC_ADD_PROPERTY_SUCCESS
Priority	High
Description	Verify property is added successfully by seller
Reference	FR_04: Add Property
Users	Seller
Pre-requisites	Seller logged in
Steps	Open dashboard > Fill form > Submit
Input	Property title, location, type, price, images
Expected Result	Property listed on frontend
Status	Tested, Passed

Table 27 Property Add By Seller Positive

Test Case – Property Add by Seller (Negative)

Field	Detail
ID	TC_ADD_PROPERTY_FAIL
Priority	High
Description	Ensure system blocks invalid property submissions
Reference	FR_04: Add Property
Users	Seller
Pre-requisites	System online
Steps	Fill form with empty fields or wrong data > Submit
Input	Missing title/price/location
Expected Result	Error message, property not added
Status	Tested, Passed

Table 28 Property Add By Seller Negative

Test Case – Inspector Application Submission (Positive)

Field	Detail
ID	TC_INSPECTOR_APP_SUBMIT_SUCCESS
Priority	High
Description	To verify successful inspector application submission
Reference	FR_06: Submit Inspector Application
Users	Inspector
Pre-requisites	Inspector logged in
Steps	Open application form > Fill fields > Upload docs > Submit
Input	Experience, image, service type
Expected Result	Application sent to admin
Status	Tested, Passed

Table 29 Inspector Application Submission Positive

Test Case – Inspector Application Submission (Negative)

Field	Detail
ID	TC_INSPECTOR_APP_SUBMIT_FAIL
Priority	High
Description	Ensure invalid application is not submitted
Reference	FR_06: Submit Inspector Application
Users	Inspector
Pre-requisites	System online
Steps	Leave mandatory fields empty > Submit
Input	Missing experience or image
Expected Result	Error message, application rejected
Status	Tested, Passed

Table 30 Inspector Application Submission Negative

Test Case – Admin Inspector Approval (Positive)

Field	Detail
ID	TC_ADMIN_INSPECTOR_APPROVE_SUCCESS
Priority	High
Description	To verify admin approves valid inspector
Reference	FR_07: Approve/Reject Inspector
Users	Admin
Pre-requisites	Inspector application submitted
Steps	Admin opens panel > Approves application
Input	Inspector ID
Expected Result	Inspector can access dashboard
Status	Tested, Passed

Table 31 Admin Inspector Approval (Positive)

Test Case – Admin Inspector Approval (Negative)

Field	Detail
ID	TC_ADMIN_INSPECTOR_APPROVE_FAIL
Priority	High
Description	Verify rejection disables inspector access
Reference	FR_07: Approve/Reject Inspector
Users	Admin
Pre-requisites	Inspector application submitted
Steps	Admin opens panel > Rejects application
Input	Inspector ID
Expected Result	Inspector sees rejection message at login
Status	Tested, Passed

Table 32 Admin Inspector Approval (Negative)

Test Case – Service CRUD by Inspector (Positive)

Field	Detail
ID	TC_INSPECTOR_SERVICE_CRUD_SUCCESS
Priority	Medium
Description	To verify inspectors can add/update/delete their services
Reference	FR_08: Inspector Service Add/Delete/Update
Users	Inspector
Pre-requisites	Inspector must be approved
Steps	Login > Open dashboard > Manage services
Input	Service title, slot, location
Expected Result	Services listed on frontend
Status	Tested, Passed

Table 33 Service CRUD by Inspector (Positive)

Test Case – Service CRUD by Inspector (Negative)

Field	Detail
ID	TC_INSPECTOR_SERVICE_CRUD_FAIL
Priority	Medium
Description	Ensure invalid services are not saved
Reference	FR_08: Inspector Service Add/Delete/Update
Users	Inspector
Pre-requisites	Inspector must be approved
Steps	Submit service form with missing data
Input	Empty title or invalid slot
Expected Result	Error shown, service not listed
Status	Tested, Passed

Table 34 Service CRUD by Inspector (Negative)

Test Case – Chatbot Query (Positive)

Field	Detail
ID	TC_CHATBOT_QUERY_SUCCESS
Priority	Medium
Description	To verify chatbot responds to budget query
Reference	FR_11: Use Chatbot
Users	Buyer
Pre-requisites	Chatbot API (OpenAI) online
Steps	Enter "Suggest areas under 50 lacs"
Input	Natural language query
Expected Result	AI-generated suggestion displayed
Status	Tested, Passed

Table 35 Chatbot Query (Positive)

Test Case – Chatbot Query (Negative)

Field	Detail
ID	TC_CHATBOT_QUERY_FAIL
Priority	Medium
Description	Verify chatbot handles unknown queries gracefully
Reference	FR_11: Use Chatbot
Users	Buyer
Pre-requisites	Chatbot API online
Steps	Enter unclear/unsupported query
Input	Gibberish or undefined terms
Expected Result	Graceful fallback message displayed
Status	Tested, Passed

Table 36 Chatbot Query (Negative)

Black Box Test Cases

Test Case – Search Property Listings

Field	Detail
ID	BB_TC_PROPERTY_SEARCH_SUCCESS
Priority	High
Description	To validate property listings load based on search filters
Reference	FR_09: Property/Inspector Search
Users	Buyer
Pre-requisites	Properties must be listed in database
Steps	Enter filters > Submit
Input	City: Lahore, Price Range: 30-50 lac
Expected Result	List of matching properties is displayed
Status	Tested, Passed

Table 37 Search Property Listings

Test Case – Invalid Property Search Filters

Field	Detail
ID	BB_TC_PROPERTY_SEARCH_FAIL
Priority	Medium
Description	To verify system handles invalid/empty search filters
Reference	FR_09: Property/Inspector Search
Users	Buyer
Pre-requisites	System online
Steps	Submit with empty/invalid fields
Input	City: --- , Price: -1
Expected Result	Warning message or no results displayed
Status	Tested, Passed

Table 38 Invalid Property Search Filters

Test Case – Inspector Booking from Map

Field	Detail
ID	BB_TC_BOOK_INSPECTOR_MAP
Priority	Medium
Description	To validate inspector details display when clicked on map
Reference	FR_10: Map View
Users	Buyer
Pre-requisites	Approved inspector with mapped location
Steps	Open map > Click inspector pin
Input	City = Karachi
Expected Result	Inspector details and booking option visible
Status	Tested, Passed

Table 39 Inspector Booking from Map

Test Case – Contact via Email to Seller/Inspector

Field	Detail
ID	BB_TC_CONTACT_EMAIL
Priority	Medium
Description	To verify user can send email from contact form
Reference	Contact Module
Users	Buyer
Pre-requisites	Seller/Inspector email present
Steps	Open listing > Click contact > Fill form > Send
Input	Name, Email, Message
Expected Result	Confirmation message after send
Status	Tested, Passed

Table 40 Contact via Email to Seller/Inspector

Test Case – Chatbot Real Estate Advice

Field	Detail
ID	BB_TC_CHATBOT_ADVICE
Priority	Medium
Description	Verify chatbot gives accurate area suggestions
Reference	FR_11: Use Chatbot
Users	Buyer
Pre-requisites	Chatbot API active
Steps	Enter budget-based query
Input	"Best area under 40 lac in Islamabad?"
Expected Result	Relevant areas listed
Status	Tested, Passed

Table 41 Chatbot Real Estate Advice

Test Case – Chatbot Invalid Input Handling

Field	Detail
ID	BB_TC_CHATBOT_INVALID_INPUT
Priority	Low
Description	Ensure chatbot responds gracefully to nonsensical input
Reference	FR_11: Use Chatbot
Users	All users
Pre-requisites	Chatbot API online
Steps	Enter gibberish query
Input	"asdfgh123\$% "
Expected Result	Message like "Sorry, I didn't understand that" shown
Status	Tested, Passed

Table 42 Chatbot Invalid Input Handling

5.1.1 Equivalence Partitions (EP)

Equivalence Partitioning (EP) is a black-box testing technique that divides input data of a software unit into partitions of valid and invalid classes. Each partition represents a set of valid or invalid states for testing to minimize the number of test cases while maintaining coverage.

EP Table – User Registration

Variables	Valid Classes	Invalid Classes
Email	Proper format (abc@example.com)	Empty / invalid format (abc@, @mail, etc.)
Password	8+ characters, includes letters & digits	Less than 8 characters, only letters or digits
Role Selection	Seller, Inspector, Both	No role selected

Table 43 EP Table – User Registration

EP Table – Property Search

Variables	Valid Classes	Invalid Classes
City	Existing cities (e.g., Lahore, Karachi)	Non-existent or blank
Price Range	Positive numeric range (10–90 lac)	Negative values, characters (e.g., "abc")
Property Type	Rent, Buy	Null / invalid entry

Table 44 EP Table – Property Search

EP Table – Inspector Application Submission

Variables	Valid Classes	Invalid Classes
Experience Field	Textual data, relevant description	Blank / numeric only
Document Upload	Image/pdf format accepted	Missing / unsupported file format
Service Type	Valid category selected	None selected

Table 45 EP Table – Inspector Application Submission

EP Table – Admin Approval Process

Variables	Valid Classes	Invalid Classes
Application Status	Approved, Rejected	Empty / invalid enum
Inspector ID	Existing inspector ID	Null / non-existent ID

Table 46 EP Table – Admin Approval Process

EP Table – Service CRUD by Inspector

Variables	Valid Classes	Invalid Classes
Title	Descriptive string	Empty
Slot Timings	Valid time ranges (e.g., 10AM - 12PM)	Overlapping/invalid format
Location	Mapped, verified city names	Null/unsupported string

Table 47 EP Table – Service CRUD by Inspector

EP Table – Chatbot Queries

Variables	Valid Classes	Invalid Classes
User Input	Real estate-related queries	Gibberish, unsupported characters
Budget Format	Numeric range + location	Missing budget or ambiguous question

Table 48 EP Table – Chatbot Queries

5.1.2 Boundary Value Analysis

Boundary Value Analysis is a technique in which test cases are designed to include values at the boundaries. This is because boundaries are where most errors tend to occur. We test just below, at, and just above the edge of input ranges.

BVA Table – Password Length Validation (User Registration)

Test Case ID	Input Length	Expected Result
BVA_PASS_TC_01	7	Rejected (below min length)
BVA_PASS_TC_02	8	Accepted (minimum valid)
BVA_PASS_TC_03	20	Accepted (maximum valid)
BVA_PASS_TC_04	21	Rejected (above max length)

Table 49 BVA Table – Password Length Validation (User Registration)

BVA Table – Property Price Range Filter

Test Case ID	Input Value (Lac)	Expected Result
BVA_PRICE_TC_01	-1	Rejected (negative value)
BVA_PRICE_TC_02	0	Accepted
BVA_PRICE_TC_03	10	Accepted (valid low)
BVA_PRICE_TC_04	90	Accepted (valid high)
BVA_PRICE_TC_05	91	Rejected (above expected max)

Table 50 BVA Table – Property Price Range Filter

BVA Table – Slot Time Range (Inspector Service)

Test Case ID	Input Time	Expected Result
BVA_SLOT_TC_01	09:59	Rejected (before valid slot)
BVA_SLOT_TC_02	10:00	Accepted (slot start boundary)
BVA_SLOT_TC_03	18:00	Accepted (slot end boundary)
BVA_SLOT_TC_04	18:01	Rejected (after valid slot)

Table 51 BVA Table – Slot Time Range (Inspector Service)

BVA Table – Inspector Experience Years

Test Case ID	Input Years	Expected Result
BVA_EXP_TC_01	-1	Rejected (invalid negative)
BVA_EXP_TC_02	0	Accepted (fresh entry)
BVA_EXP_TC_03	30	Accepted (valid upper bound)
BVA_EXP_TC_04	31	Rejected (exceeds limit)

Table 52 BVA Table – Inspector Experience Years

BVA Table – Email Input Characters

Test Case ID	Input Length	Expected Result
BVA_EMAIL_TC_01	0	Rejected (empty)
BVA_EMAIL_TC_02	1	Rejected (incomplete format)
BVA_EMAIL_TC_03	5	Rejected (still likely invalid)
BVA_EMAIL_TC_04	12	Accepted (typical email length)

Table 53 BVA Table – Email Input Characters

5.1.3 Decision Table Testing

Decision Table Testing is a black-box test design technique used for modeling complex logic and business rules. Each decision table lists combinations of conditions and the corresponding system actions.

Decision Table – User Role Redirection After Login

Test Case	Seller Selected	Inspector Selected	Expected Redirect Page
DT_ROLE_01	Yes	No	Seller Dashboard
DT_ROLE_02	No	Yes	Inspector Dashboard
DT_ROLE_03	Yes	Yes	Prompt for panel choice
DT_ROLE_04	No	No	Error / Prompt to select

Table 54 Decision Table – User Role Redirection After Login

Decision Table – Property Search Filter Logic

Test Case	City Provided	Price Range Valid	Type Selected	Result Type
DT_SEARCH_01	Yes	Yes	Yes	Show matching listings
DT_SEARCH_02	Yes	No	Yes	Prompt invalid price
DT_SEARCH_03	No	Yes	Yes	Prompt city required

DT_SEARCH_04	Yes	Yes	No	Prompt type required
--------------	-----	-----	----	----------------------

Table 55 Decision Table – Property Search Filter Logic

Decision Table – Inspector Application Review (Admin)

Test Case	Inspector Submitted	Documents Valid	Admin Action	System Output
DT_ADMIN_01	Yes	Yes	Approve	Inspector gets dashboard access
DT_ADMIN_02	Yes	Yes	Reject	Display rejection message
DT_ADMIN_03	Yes	No	Reject	Notify missing documents
DT_ADMIN_04	No	--	--	No action taken

Table 56 Decision Table – Inspector Application Review (Admin)

Decision Table – Chatbot Response Scenarios

Test Case	Budget Mentioned	Location Mentioned	Expected Chatbot Output
DT_CHAT_01	Yes	Yes	Suggest properties/areas
DT_CHAT_02	Yes	No	Ask for location
DT_CHAT_03	No	Yes	Ask for budget
DT_CHAT_04	No	No	Ask for both

Table 57 Decision Table – Chatbot Response Scenarios

Decision Table – Inspector Service CRUD Validations

Test Case	Title Given	Slot Valid	Location Selected	Save Operation Result
DT_SERVICE_01	Yes	Yes	Yes	Service saved successfully
DT_SERVICE_02	No	Yes	Yes	Prompt title is required
DT_SERVICE_03	Yes	No	Yes	Prompt invalid time slot
DT_SERVICE_04	Yes	Yes	No	Prompt location required

Table 58 Decision Table – Inspector Service CRUD Validations

5.1.4 State transition Testing

State Transition Testing is used to validate the behavior of the system when it moves from one state to another based on user actions or internal triggers. This is especially useful for modules involving login flows, registration, role selection, and dashboard redirections.

State Transition Table – User Registration Flow

Current State	Event	Next State	Expected Action
Start	Open Registration Page	Filling Form	Show registration form
Filling Form	Submit Valid Data	Role Selection	Ask to select role(s)
Role Selection	One Role Selected	Redirecting	Redirect to selected panel
Role Selection	Both Roles Selected	Confirmation Prompt	Ask for redirection choice
Redirecting	Panel Loaded	Dashboard Access	Show appropriate dashboard

Table 59 State Transition Table – User Registration Flow

State Transition Table – Inspector Application Review

Current State	Event	Next State	Expected Action
Submitted	Admin Opens Panel	Reviewing	Show application details
Reviewing	Admin Clicks Approve	Approved	Grant inspector access
Reviewing	Admin Clicks Reject	Rejected	Show rejection message
Approved	Inspector Logs In	Inspector Panel	Enable service CRUD
Rejected	Inspector Logs In	Blocked	Show rejection alert

Table 60 State Transition Table – Inspector Application Review

State Transition Table – Property Posting by Seller

Current State	Event	Next State	Expected Action
Seller Login	Click Add Property	Fill Form	Open property form
Fill Form	Submit Valid Details	Pending Listing	Property sent to database
Pending Listing	Auto Update	Published	Property visible on frontend
Fill Form	Submit Invalid Details	Error	Show validation errors

Table 61 State Transition Table – Property Posting by Seller

State Transition Table – Chatbot Session Handling

Current State	Event	Next State	Expected Action
Idle	User Opens Chatbot	Listening	Prompt welcome message
Listening	Budget & Location Input	Processing Query	Analyze input
Processing Query	Valid Query	Responding	Show area/property suggestions
Processing Query	Invalid/Nonsense Input	Error State	Respond with fallback/help message

Table 62 State Transition Table – Chatbot Session Handling

State Transition Table – Login & Role-Based Dashboard Routing

Current State	Event	Next State	Expected Action
Login Page	Enter Credentials	Authenticating	Validate email/password
Authenticating	Valid Credentials	Check Role	Redirect based on role(s)
Check Role	Seller Only	Seller Dashboard	Show seller tools
Check Role	Inspector Only	Inspector Dashboard	Show inspector tools
Check Role	Both Selected	Prompt Choice	Ask for preferred panel
Authenticating	Invalid Credentials	Login Failed	Show error message

Table 63 State Transition Table – Login & Role-Based Dashboard Routing

5.1.5 Use Case Testing

Use Case: UC_01 – Create Account

Use Case Name: Create Account

Actors: User (Buyer, Seller, Inspector)

Preconditions:

- User must have internet access
- User must be on Sign-Up page

Basic Flow:

1. User enters name, email, password, phone
2. Selects roles (Seller, Inspector, or Both)
3. Clicks “Sign Up”
4. System creates account
5. If both roles selected, prompt for redirection options

Alternative Flows:

A1. Missing/invalid inputs → Error message shown

A2. Existing email → System denies registration

Postconditions:

Account created and stored in DB, redirected to selected dashboard

Expected Result:

User account successfully registered and redirected

Use Case: UC_02 – Add Property (Seller Panel)

Use Case Name: Add Property

Actors: Seller

Preconditions:

- Seller logged in
- Access to Seller Dashboard

Basic Flow:

1. Seller fills property form (title, location, type, price)
2. Clicks “Submit”
3. System validates and stores data
4. Property listed on frontend

Alternative Flows:

A1. Invalid data → System prompts for correction

Postconditions:

Property listed and visible to users

Expected Result:

Property successfully added to the system

Use Case: UC_03 – Submit Inspector Application

Use Case: UC_04 – Approve/Reject Inspector (Admin)

Use Case Name: Approve/Reject Inspector

Actors: Admin

Preconditions:

- Admin logged in
- Application list available

Basic Flow:

1. Admin views pending applications
2. Selects application
3. Clicks Approve/Reject
4. Status updated in system

Alternative Flows:

A1. Network error → Action not processed

Postconditions:

Inspector receives approval or rejection response

Expected Result:

System updates inspector status accordingly

Use Case: UC_05 – Search Property/Service

Use Case Name: Search Property/Service

Actors: User (Buyer)

Preconditions:

- User is on Home/Services/Map page

Basic Flow:

1. User enters filters (city, location, price, buy/rent)
2. Clicks search
3. System fetches filtered results
4. Results displayed to user

Alternative Flows:

A1. No results match → “No properties found” message shown

Postconditions:

User views property or inspection listings

Expected Result:

Accurate filtered data is shown to the user

Use Case: UC_06 – Contact Seller or Inspector

Use Case Name: Contact Seller or Inspector

Actors: User (Buyer)

Preconditions:

- Property or service listing available
- Contact details enabled

Basic Flow:

1. User clicks on property or service
2. Contact info (phone/email) displayed
3. User fills contact form or calls directly

Alternative Flows:

A1. Email fails to send → System shows failure notice

Postconditions:

Message sent or contact made with Seller/Inspector

Expected Result:

User successfully initiates contact

Use Case: UC_07 – Use AI Chatbot

Use Case Name: Use AI Chatbot

Actors: Any site visitor

Preconditions:

- Chatbot widget loaded on website

Basic Flow:

1. User opens chatbot
2. Asks a question (e.g. "suggest area in budget")
3. Chatbot processes query
4. Displays relevant answer

Alternative Flows:

A1. API failure → Error or fallback message shown

Postconditions:

User receives chatbot assistance

Expected Result:

Chatbot responds with accurate, relevant information

5.2 White Box Test Cases

White Box Testing focuses on the internal logic, code paths, and structure of the application. This section outlines key function-level test cases that verify the correctness of individual components of the **Prime Innovative Hub** system.

Test Case 1 – Function: Register User()

Field	Detail
ID	WB_TC_01
Function	Register User()
Purpose	Validate user registration based on role selection
Test Type	Condition Coverage
Input	Name, Email, Password, Role (Seller/Inspector/Both)
Expected Output	User created and redirected to relevant dashboard
Branches Tested	If role == Seller / Inspector / Both
Status	Passed

Table 64 Test Case 1 – Function: Register User()

Test Case 2 – Function: Submit Inspector Application

Field	Detail
ID	WB_TC_02
Function	submitInspectorApplication()
Purpose	Ensure inspector application is stored and sent to Admin
Test Type	Decision Coverage
Input	Inspector name, experience, file upload
Expected Output	Application saved, success message shown
Branches Tested	Valid input → Store application; Invalid → Return error
Status	Passed

Table 65 Test Case 2 – Function: Submit Inspector Application

Test Case 3 – Function: Approve Or Reject Inspector()

Field	Detail
ID	WB_TC_03
Function	approveOrRejectInspector(inspectorID, decision)
Purpose	Ensure Admin can approve or reject applications
Test Type	Path Coverage
Input	Inspector ID, decision = approve or reject
Expected Output	Inspector status updated; notification sent
Paths Tested	If decision = approve → Set status approved; else → Set rejected
Status	Passed

Table 66 Test Case 3 – Function: Approve Or Reject Inspector()

Test Case 4 – Function: AddProperty()

Field	Detail
ID	WB_TC_04
Function	addProperty()
Purpose	Validate property listing creation
Test Type	Multiple Condition Coverage
Input	Property title, location, price, images

Field	Detail
Expected Output	Property saved, shown on frontend
Conditions Tested	All fields filled → Save; Missing fields → Return error
Status	Passed

Table 67 Test Case 4 – Function: AddProperty()

Test Case 5 – Function: filterProperties()

Field	Detail
ID	WB_TC_05
Function	filterProperties(filters)
Purpose	Test accurate filtering of property listings
Test Type	Statement and Loop Coverage
Input	Filters: city = Lahore, price = 30–50 lac
Expected Output	Filtered list of properties
Loop Tested	For each property → Apply all filters
Status	Passed

Table 68 Test Case 5 – Function: filterProperties()

Test Case 6 – Function: ChatbotRespond()

Field	Detail
ID	WB_TC_06
Function	chatbotRespond(query)
Purpose	Verify chatbot sends query to API and displays result
Test Type	API Integration & Condition Coverage
Input	"Suggest area for 40 lac in Karachi"
Expected Output	Relevant response from OpenAI API
Branches Tested	If API success → Show response; else → Show fallback message
Status	Passed

Table 69 Test Case 6 – Function: ChatbotRespond()

5.2.1 Cyclometric complexity

Cyclomatic Complexity is a metric used to measure the complexity of a program's control flow. It helps determine the number of independent paths through the program and guides the creation of effective test cases.

The formula used is:

$$CC = E - N + 2P$$

Where:

- **E** = Number of edges
- **N** = Number of nodes
- **P** = Number of connected components (usually 1 for a single function)

Function: registerUser()

Code Summary:

```
javascript
CopyEdit
function registerUser(role) {
  if (role === "Seller") {
    redirectToSellerDashboard();
  } else if (role === "Inspector") {
    redirectToInspectorDashboard();
  } else if (role === "Both") {
    promptForDashboardChoice();
  } else {
    showError("Invalid role selected");
  }
}
```

- Nodes (N): 6
- Edges (E): 7
- P: 1
- **Cyclomatic Complexity: $CC = 7 - 6 + 2(1) = 3$**

Function: approveOrRejectInspector()

Code Summary:

```
typescript
CopyEdit
function approveOrRejectInspector(id, action) {
  if (action === "approve") {
```

```
        updateStatus(id, "Approved");
    } else if (action === "reject") {
        updateStatus(id, "Rejected");
    } else {
        throw new Error("Invalid action");
    }
}
```

- Nodes (N): 5
- Edges (E): 6
- P: 1
- **Cyclomatic Complexity: $CC = 6 - 5 + 2 = 3$**

Function: filterProperties()

Code Summary:

```
typescript
CopyEdit
function filterProperties(properties, filters) {
    const results = [];
    for (let p of properties) {
        if (matches(p, filters)) {
            results.push(p);
        }
    }
    return results;
}
```

- Nodes (N): 4
- Edges (E): 5
- P: 1
- **Cyclomatic Complexity: $CC = 5 - 4 + 2 = 3$**

Function: chatbotRespond()

Code Summary:

```
javascript
CopyEdit
function chatbotRespond(query) {
    if (!query) {
        return "Please enter a question.";
    }
    const response = callOpenAI(query);
    if (response.success) {
        return response.answer;
    }
}
```

```

    } else {
        return "Sorry, I couldn't understand.";
    }
}

```

- Nodes (N): 6
- Edges (E): 7
- P: 1
- Cyclomatic Complexity: $CC = 7 - 6 + 2 = 3$

Summary Table:

Function Name	Nodes (N)	Edges (E)	CC	Interpretation
registerUser()	6	7	3	Moderate (Multiple branches)
approveOrRejectInspector ()	5	6	3	Simple decision logic
filterProperties ()	4	5	3	Loop with conditional logic
chatbotRespond ()	6	7	3	External API call with fallback

Table 70 Summary Table

5.4 Performance Testing

Performance Testing verifies that the system meets required responsiveness and stability under a defined workload. The following test cases evaluate system response times for key functionalities of the **Prime Innovative Hub**.

Test Case 1 – Search Function Response Time

Field	Detail
ID	PT_TC_01
Module	Property/Service Search
Objective	Measure time to return filtered results
Test Type	Response Time
Test Load	1 user, normal search filters
Expected Result	Results returned in ≤ 2 seconds
Observed Result	1.3 seconds
Status	Passed

Table 71 Test Case 1 – Search Function Response Time

Test Case 2 – Map Load Time

Field	Detail
ID	PT_TC_02
Module	Google Maps View
Objective	Verify time to load map and pins
Test Type	Load Time
Test Load	1 user, city = Lahore
Expected Result	Map and pins displayed in ≤ 3 seconds
Observed Result	2.4 seconds
Status	Passed

Table 72 Test Case 2 – Map Load Time

Test Case 3 – Chatbot Reply Time

Field	Detail
ID	PT_TC_03
Module	Chatbot Interaction
Objective	Measure response time from OpenAI API
Test Type	API Response
Test Load	1 user query
Expected Result	Reply generated within 2 seconds
Observed Result	1.8 seconds
Status	Passed

Table 73 Test Case 3 – Chatbot Reply Time

5.5 Stress Testing

Stress Testing evaluates system stability and behavior under extreme or unexpected workloads. The goal is to determine system breaking points and recovery behavior.

Test Case 1 – Simulated User Surge

Field	Detail
ID	ST_TC_01
Scenario	100+ users simultaneously search properties
Expected Result	System maintains performance without crash

Field	Detail
Observed Result	Minor delays at 90+ users, no crash
Status	Passed

Table 74 Test Case 1 – Simulated User Surge

Test Case 2 – Concurrent Inspector Applications

Field	Detail
ID	ST_TC_02
Scenario	50 inspectors submit application simultaneously
Expected Result	All applications processed without data loss
Observed Result	All submitted, with average processing time of 2.8 sec
Status	Passed

Table 75 Test Case 2 – Concurrent Inspector Applications

Test Case 3 – Mass Property Upload by Sellers

Field	Detail
ID	ST_TC_03
Scenario	30 sellers upload properties concurrently
Expected Result	No system crash; all listings saved
Observed Result	100% upload success; 1 delayed submission
Status	Passed

Table 76 Test Case 3 – Mass Property Upload by Sellers

5.6 System Testing

System Testing validates the complete and integrated software application against the requirements. The following end-to-end tests verify that major modules work together properly.

Test Case 1 – Role-Based Redirection Post Login

Field	Detail
ID	SYS_TC_01
Scenario	User logs in with dual roles (Seller + Inspector)

Field	Detail
Expected Result	System asks for role panel selection
Observed Result	Prompt appeared, redirection successful
Status	Passed

Table 77 Test Case 1 – Role-Based Redirection Post Login

Test Case 2 – Admin Approval Flow for Inspector

Field	Detail
ID	SYS_TC_02
Scenario	Inspector submits application → Admin approves
Expected Result	Inspector gains dashboard access
Observed Result	Access granted after approval
Status	Passed

Table 78 Test Case 2 – Admin Approval Flow for Inspector

Test Case 3 – Property Listing and Display Flow

Field	Detail
ID	SYS_TC_03
Scenario	Seller adds property → Visible in search
Expected Result	Property appears with correct filters
Observed Result	Verified on search and map view
Status	Passed

Table 79 Test Case 3 – Property Listing and Display Flow

5.7 Regression Testing

Regression Testing ensures that new changes (feature additions or bug fixes) do not negatively affect existing functionalities of the **Prime Innovative Hub**. This section outlines the process and selected test cases for regression cycles after code updates.

5.7.1 Selecting Regression Tests

The following criteria were used to select modules for regression testing:

Regression Test Cycle – Example 1

Field	Detail
ID	RG_TC_01
Scenario	Add property → Display on frontend
Previous Status	Passed
Modified Component	AddProperty() form validation logic
Retest Result	Listing displayed correctly
Regression Status	Passed

Table 80 Regression Test Cycle – Example 1

Regression Test Cycle – Example 2

Field	Detail
ID	RG_TC_02
Scenario	Inspector registers → Admin approval → Dashboard access
Previous Status	Passed
Modified Component	Admin approval delay handling
Retest Result	Inspector accessed dashboard as expected
Regression Status	Passed

Table 81 Regression Test Cycle – Example 2

Regression Test Cycle – Example 3

Field	Detail
ID	RG_TC_03
Scenario	Chatbot responds to area suggestion queries
Previous Status	API timeout issue reported
Modified Component	Chatbot retry logic added
Retest Result	Bot responded correctly on second attempt
Regression Status	Passed

Table 82 Regression Test Cycle – Example 3

5.7.2 Regression Testing Steps

Below are the defined steps followed in the regression testing phase of the project:

1. **Identify Change Areas**
 - Components impacted by updates (code commits or bug fixes)
2. **Select Impacted Modules for Testing**
 - Based on dependencies and functional linkage
3. **Re-execute Previous Test Cases**
 - Compare results with earlier execution logs
4. **Log Outcomes and Fix Failures**
 - Failures are logged and assigned for immediate fixes
5. **Maintain Regression Suite**
 - Successful cases archived; new test cases added after fixes'

Chapter 6: Tools and Techniques

Abstract Outline

This chapter explores the essential tools and technologies utilized in the development of a comprehensive real estate web application. The project is built using modern technologies such as React.js for the frontend and NestJS for the backend, supported by PostgreSQL for data storage. Additionally, powerful integrations like the OpenAI chatbot and Google Maps API enhance user interaction and geolocation features. This chapter provides a detailed overview of the programming languages, applications, tools, libraries, and extensions employed to ensure scalability, performance, and usability.

6.1 Languages Used

- **JavaScript (React.js):** Used for frontend development to build responsive and dynamic user interfaces that deliver a seamless experience to users.
- **TypeScript (NestJS):** Used on the backend to create robust, type-safe, and maintainable APIs following the modular architecture of NestJS.
- **SQL (PostgreSQL):** A relational database language used to store and manage structured data efficiently.

6.2 Applications and Tools

- **Visual Studio Code:** A lightweight but powerful code editor used for writing, debugging, and managing frontend and backend code.
- **PostgreSQL Admin Tools (e.g., pgAdmin):** Used to manage and interact with the PostgreSQL database, run queries, and visualize data.
- **Postman:** A collaboration platform for API development used to test and debug RESTful services created in NestJS.
- **OpenAI Chatbot Interface:** Integrated into the application to provide real-time conversational support and intelligent responses to user queries.
- **Google Maps API:** Used to display properties and inspectors based on location, offering interactive geographic visualizations.

6.3 Libraries and Extensions

- **React.js:** A powerful JavaScript library used for building reusable UI components and handling frontend rendering.
- **NestJS Framework:** A progressive Node.js framework used for building efficient and scalable server-side applications with TypeScript.
- **OpenAI API SDK:** Used to integrate chatbot functionality, enabling AI-powered interaction with users.
- **Google Maps JavaScript API:** Used for rendering maps and visualizing data such as property and inspector locations based on user input.

Chapter 7: Summary and Conclusion

Summary

The Real Estate Website is a modern, user-centric platform developed to streamline property buying, selling, and inspection services. Designed with both sellers and property inspectors in mind, the system provides an intuitive interface and powerful backend infrastructure to support user registration, listing, service management, and real-time communication. The frontend of the system is built using **React.js**, enabling dynamic and responsive UI components, while **NestJS**, a scalable Node.js framework, powers the backend logic. Data is persistently managed using **PostgreSQL**, a robust relational database. The integration of the **Google Maps API** allows users to visually explore properties and available inspectors based on city input. Furthermore, the inclusion of an **AI-powered chatbot**, using **OpenAI**, enhances user support by addressing queries, providing recommendations, and guiding users through property or service selection based on budget or location.

The platform supports multiple user roles:

- **Customers**, who can search for properties or inspectors.
- **Sellers**, who can register and manage their property listings.
- **Inspectors**, who can apply, get approval from the admin, and then manage their own services.
- **Admins**, who oversee all operations, including approving inspectors and moderating listings.

Robust CRUD operations for properties and services, form-based communication, real-time alerts, and user validation ensure a seamless and secure experience. The chatbot, advanced search filters, and map-based navigation combine functionality with usability to meet a broad range of real estate needs.

Conclusion

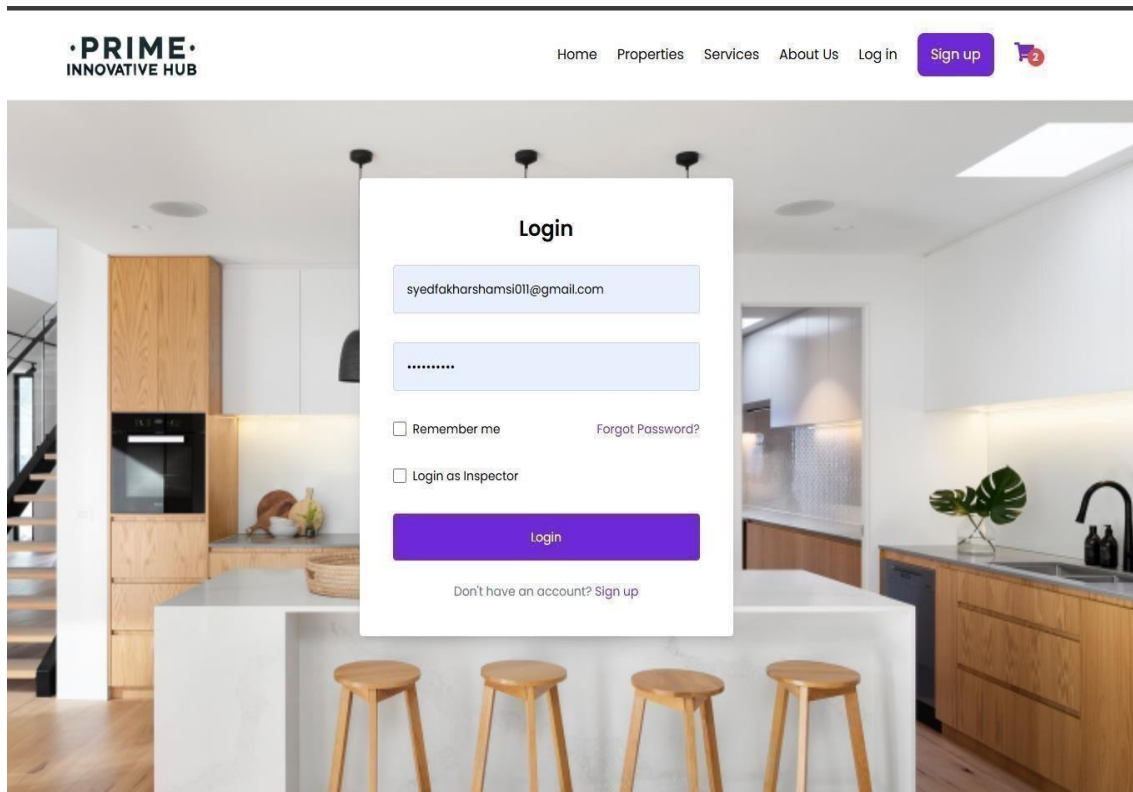
The Real Estate Website effectively combines geospatial technology, AI integration, and role-based functionality to create a comprehensive solution for the real estate sector. With features that support inspectors, sellers, and customers, it delivers a complete ecosystem for property discovery, service hiring, and user communication. Through the use of modern frameworks like React and NestJS, and integrations like OpenAI and Google Maps, the platform demonstrates the potential of full-stack web development in solving real-world problems. The system is secure, scalable, and easy to use, meeting the needs of various stakeholders. In the future, enhancements such as real-time notifications, mobile app integration, and automated property recommendation systems could further increase the platform's value. Overall, this project represents a significant step toward a smarter and more efficient real estate experience powered by modern web technologies and AI.

Chapter 8: User Manual

8.1 Login Function

The login functionality allows users to authenticate and access their respective dashboards (Buyer, Seller, Inspector, or Admin). Users must have an active username and password to log into the system.

Login Authentication Screen



Description: This screenshot shows the login screen where users are required to enter their username and password.

Steps to log in:

1. **Open the login webpage** – Navigate to the web address provided to access the login screen.
2. **Enter login details** – Enter your valid username and password. These credentials are provided by the system administrator.
3. **Click "Login"** – After entering the correct details, click the "Login" button.

Done – Once logged in successfully, you will be directed to the appropriate user dashboard (Buyer, Seller, Inspector, Admin).

Create an Account

Full Name
Full Name is required

Email Address
Valid Email Address is required

Phone Number

Password
Password is required

Confirm Password

☐ I agree to the Terms and Conditions

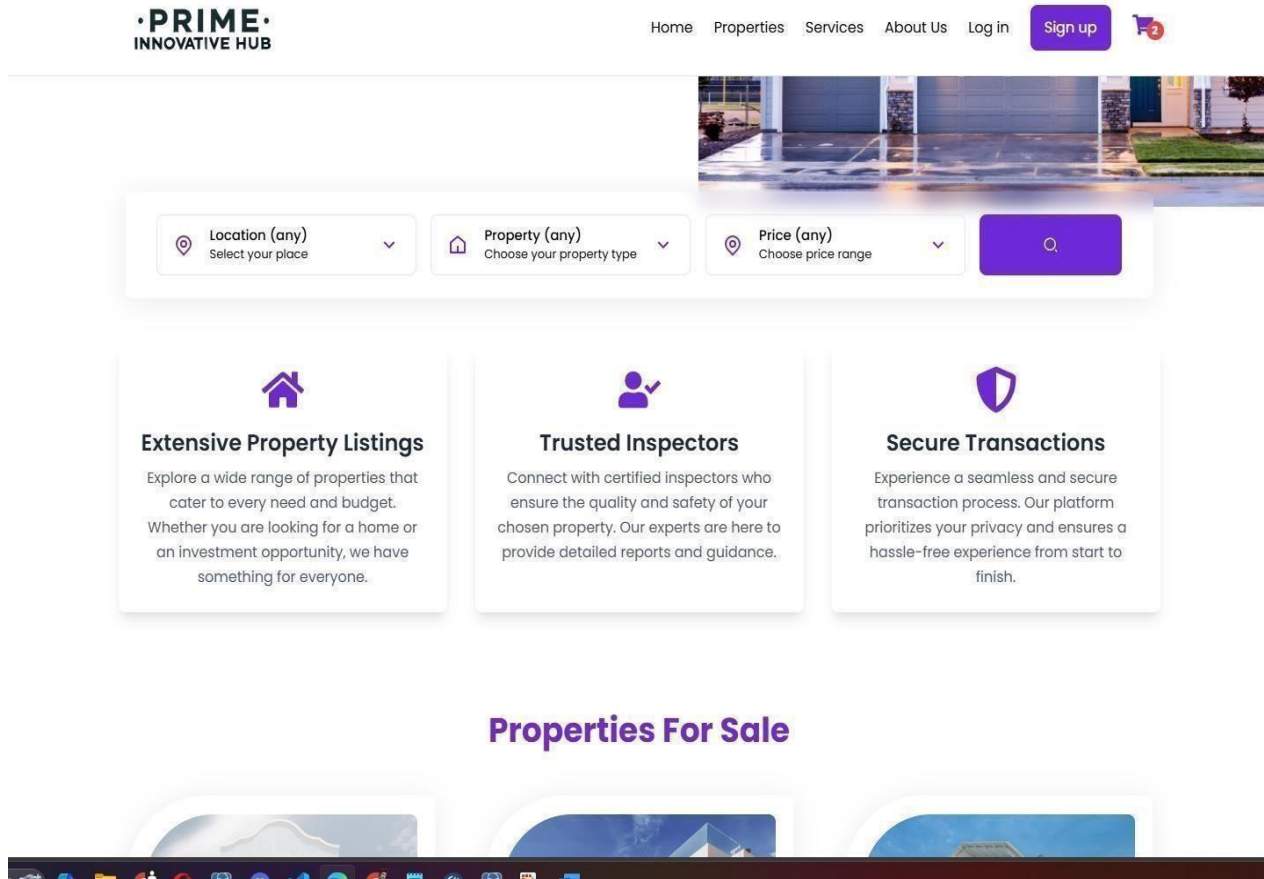
☐ Register as Inspector

Register

[Already have an account? Login](#)

8.2 Property Listing Search

The property listing page allows users to search for properties based on various filters such as category, price range, and city.



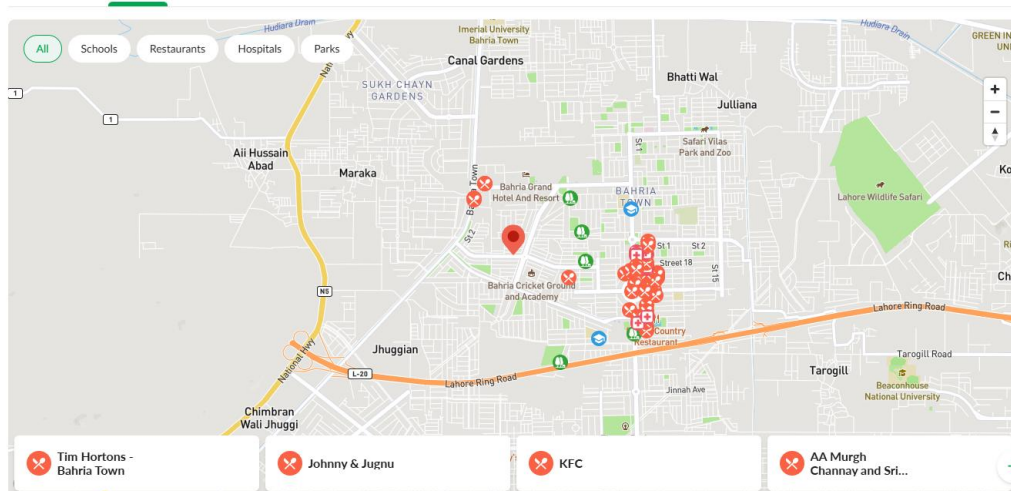
Map View:

Zameen > Society Maps > Lahore > Bahria Town - Block AA

Bahria Town - Block AA

Pakistan

Society Map Map view



Description: This screenshot shows the property listing page with available filters such as category, city, and price range to help users find properties more easily.

Steps to search for properties:

1. **Go to the property listing page** – Open the page where all available properties are listed.
2. **Apply filters** – Select the category, city, and price range to narrow down the property search.
3. **View filtered properties** – Once filters are applied, the properties that match your criteria will be displayed.
4. **Click on a property** – To view the detailed information, click on a property from the list.

The screenshot displays the 'House 1' listing on the Prime Innovative Hub website. The header includes the site logo, navigation links (Home, Properties, Services, About Us, Log in), and a 'Sign up' button. The property details show 'House 1' at '7240C Argyle St. Lawndale, CA 90260' with a price of '\$110000'. A large image of a modern living room is featured. To the right, a contact form for 'Patricia Tullert' includes fields for Name, Email, Phone, and a message box, along with 'Send message', 'Call', and 'Add to Cart' buttons. Below the image, there is a placeholder text block.

PRIME INNOVATIVE HUB

Home Properties Services About Us Log in Sign up

House 1
7240C Argyle St. Lawndale, CA 90260

\$110000

Patricia Tullert
View Listings

Name

Email

Phone

Hello, I am interested in [House 1]

Send message Call

Add to Cart

6 3 4200 sq ft

Lorem ipsum dolor sit amet consectetur adipisicing elit. Amet, illo. Repudiandae ratione impedit delectus consectetur. Aspernatur vero obcaecati placeat ab distinctio unde ipsam molestias atque ratione delectus blanditiis nemo eius dignissimos doloremque quae aliquid maiores id tempore consequatur, quod pariatur saepe.

8.2 Booking an Inspection Slot

Buyers can book inspection slots by selecting available inspectors and choosing a suitable date and time.

Inspector Selection and Booking Form

The screenshot shows the 'Our Inspectors' section of the Prime Innovative Hub website. It features three inspector profiles: John Doe (Vibe Inspection, \$150), Jane Smith (3D Inspection, \$200), and Robert Brown (Virtualwalk Inspection, \$180). Each profile includes a photo, name, category, price, qualifications, and a 'Book a Slot' button. The footer contains the site logo, navigation links, and social media icons.

PRIME INNOVATIVE HUB

Home Properties Services About Us Log in Sign up

Our Inspectors

John Doe
Category: Vibe Inspection
Price: \$150
Qualification: Electrical Engineer, 8 Years Experience
Book a Slot

Jane Smith
Category: 3D Inspection
Price: \$200
Qualification: Biotechnical Engineer, 10 Years Experience
Book a Slot

Robert Brown
Category: Virtualwalk Inspection
Price: \$180
Qualification: Carpentry Expert, 12 Years Experience
Book a Slot

PRIME INNOVATIVE HUB

Home Properties Services About Us Log in Sign up

The screenshot displays the PRIME INNOVATIVE HUB website. The header features the PRIME logo and navigation links: Home, Properties, About Us, Services, Login, and Sign Up. Social media icons for Facebook, Instagram, and Twitter are also present. Below the header, the main content area is divided into two sections. On the left, the profile of John Doe is shown, including a photo of a modern living room, his name, category (Wire Inspection), price (\$150), and qualification (Electrical Engineer, 8 Years Experience). On the right, the 'Book a Slot' form is visible, containing input fields for Name, Email, Phone, and an optional Message. Below these fields is a calendar titled 'Available Slots' for January 2025, showing dates from 30 to 2. A 'Confirm Booking' button is located at the bottom of the form.

PRIME INNOVATIVE HUB

Home Properties About Us Services Login Sign Up

Home Properties Services About Us Log in Sign up

John Doe

Category: Wire Inspection

Price: \$150

Qualification: Electrical Engineer, 8 Years Experience

Book a Slot

Your Name

Your Email

Your Phone

Message (Optional)

Available Slots

January 2025

MON	TUE	WED	THU	FRI	SAT	SUN
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2

Please select a date from the calendar.

Confirm Booking

Description: This screenshot shows the inspector details page with an option to book an inspection. It also shows the booking form where the buyer can fill in their details and choose a time slot.

Steps to book an inspection:

1. **Go to the inspector details page** – After selecting an inspector from the service list, you will be redirected to the inspector's profile.
2. **Fill in your details** – Enter your name, email, phone, and any message you want to send to the inspector.
3. **Select available time** – Choose a date and time from the available slots.
4. **Confirm booking** – Click the "Confirm Booking" button to submit the request.

Apply for Inspection Specialist

Full Name is required

Valid Age (18+) is required

Address must be at least 10 characters

Select Qualification

▼

Qualification must be a 4-year degree

At least 4 years of experience is required

Apply for Inspector Category

▼

Category selection is required

Upload your image

Choose File

No file chosen

Your image is required

Upload your Experience Letter

Choose File

No file chosen

Experience letter is required

Apply

8.4 Profile Management (Seller/Inspector)

Sellers and Inspectors can manage their profiles, including updating personal details and adding available slots for inspections.


Profile Edit Screen

localhost:3000 says
Profile updated successfully!

OK

Logout

Your Profile



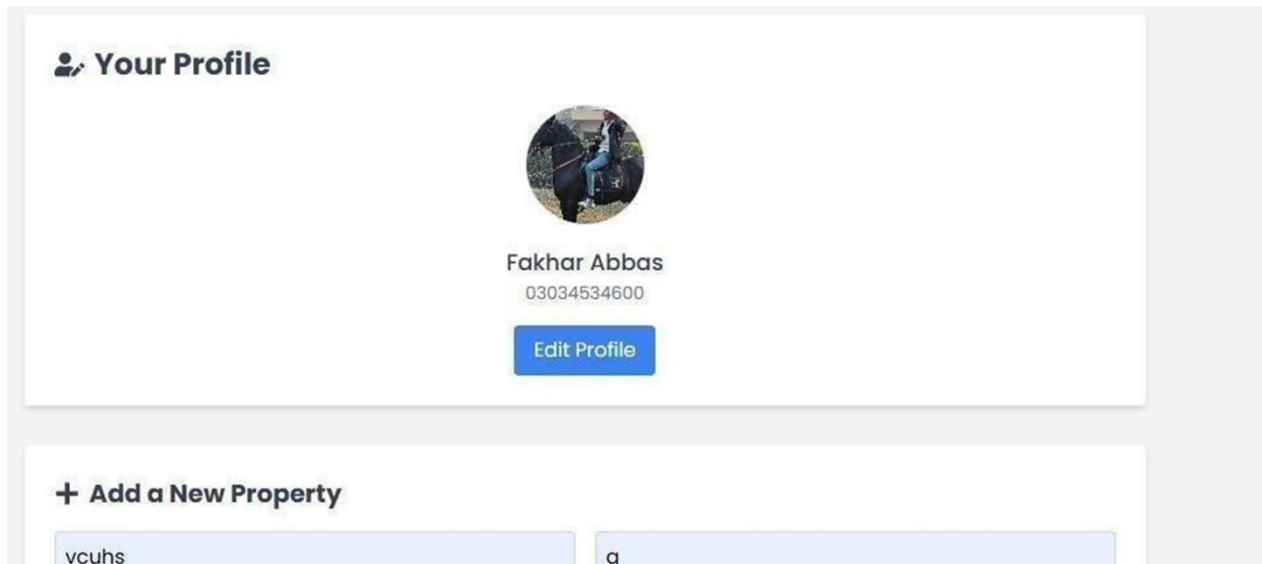
Choose File

20250104_144516.jpg


Fakhar Abbas

03034534600

Save Profile



Your Profile



Fakhar Abbas
03034534600

[Edit Profile](#)

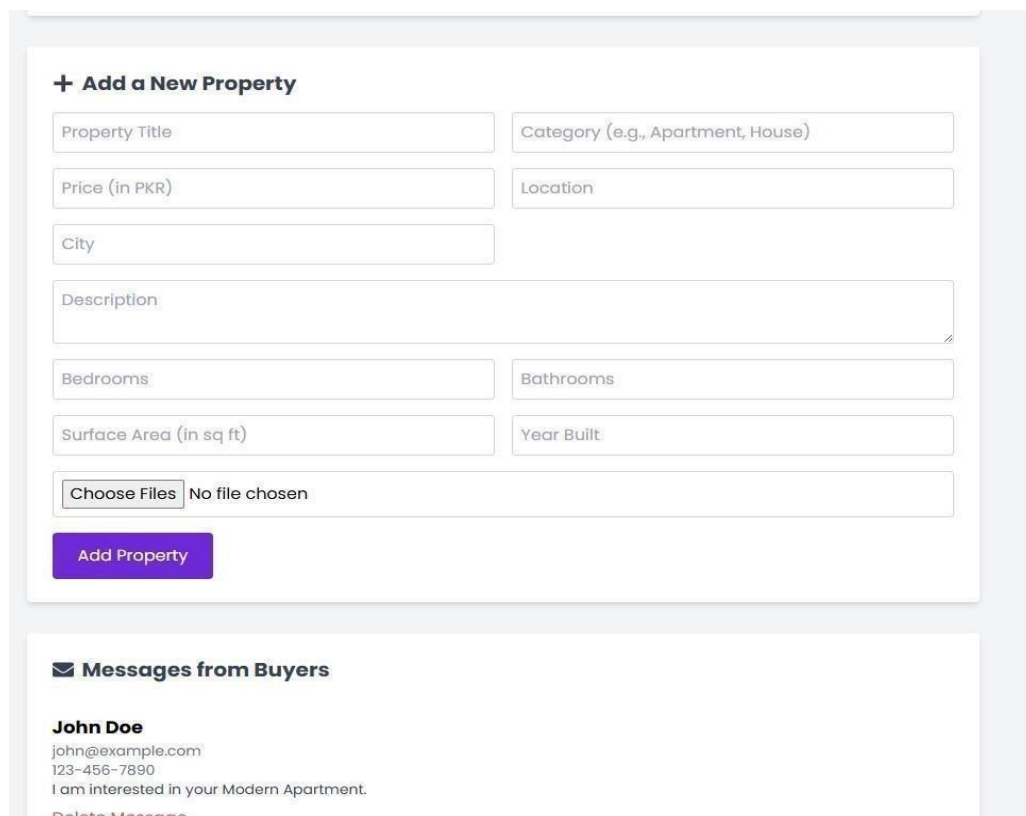
+ Add a New Property

vcuhs q

Description: This screenshot shows the profile page where the user can update their name, email, phone number, and image. Inspectors can also update their available time slots here.

Steps to update your profile:

1. **Go to the profile page** – Navigate to the profile section from the main dashboard.
2. **Edit personal details** – Update your name, email, phone number, or image by filling in the fields.



+ Add a New Property

Property Title Category (e.g., Apartment, House)

Price (in PKR) Location

City

Description

Bedrooms Bathrooms

Surface Area (in sq ft) Year Built

[Choose Files](#) No file chosen

[Add Property](#)

✉ Messages from Buyers

John Doe
john@example.com
123-456-7890
I am interested in your Modern Apartment.

[Delete Message](#)

Messages from Buyers

John Doe

john@example.com

123-456-7890

I am interested in your Modern Apartment.

[Delete Message](#)

Listed Properties



house in valancia

house

Price: PKR 1000

Location: near raiwind

City: Il

Bedrooms: 6

Bathrooms: 3

Surface Area: 4 sq ft

Year Built: 1

 [Edit](#)  [Delete](#)

[Logout](#)

Apply for Inspection Specialist

Upload your image

No file chosen

Upload your Experience Letter

No file chosen

Inspector Dashboard:

Inspector Dashboard

Profile

150 x 150

John Doe
johndoe@example.com
555-123-4567

Edit Profile

Manage Slots

Select Date

2025-01-08

Enter Time

--:--

Add Slot

Service Charges (PKR)

Enter service charges

Add Service Charges

Booking Requests

Alice Brown
Email: alice@example.com
Phone: 123-456-7890
Date: 2025-01-10
Time: 10:00 AM
Location: 123 Main Street, Springfield
Message: Looking forward to the inspection.

Confirm

Chatbot:

PRIME
INNOVATIVE HUB

Home Properties Services About Us Log in Sign up

Buy Your dream house with us

Prime Innovative Hub is your trusted real estate platform offering secure and transparent buying and selling services, complemented by expert property inspections.

Location (any)
Select your price

Property (any)
Choose your property type

Price (any)
Choose price range

Prime Innovative Hub

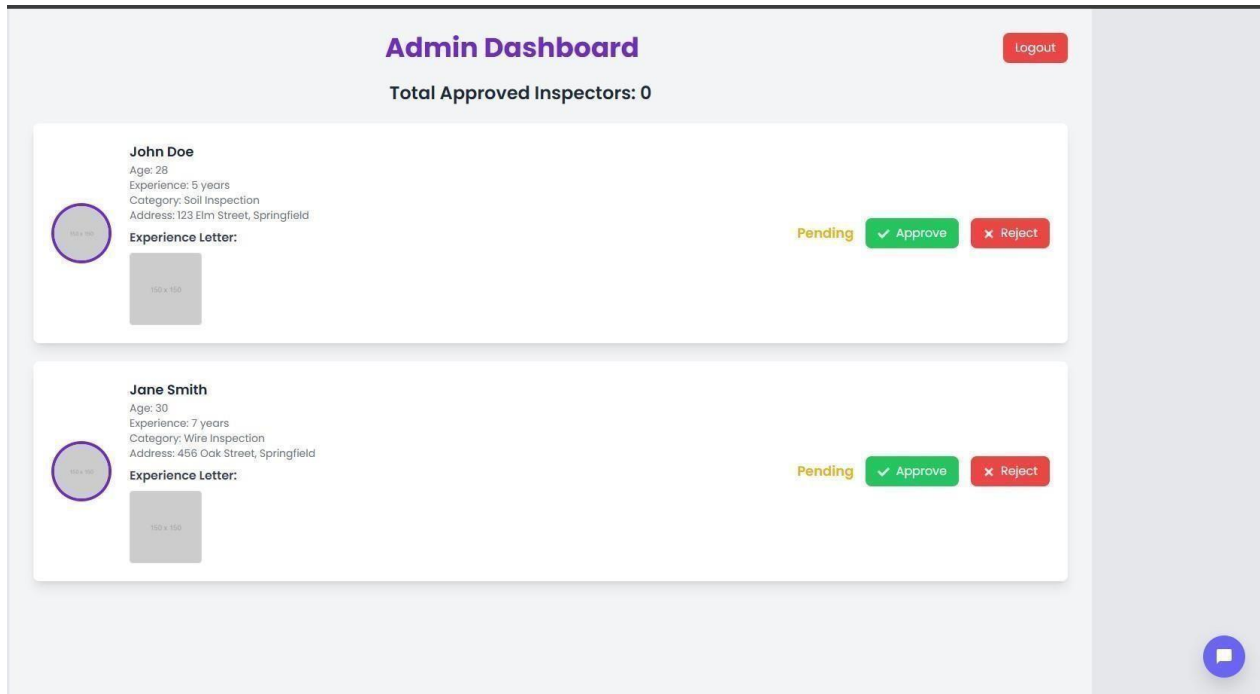
Hello! How can I assist you today?

My budget is 40 lakh which size of house should I purchase in lahore pakistan

In Lahore, Pakistan, the size of the house you can purchase for a budget of 40 lakh (4 million PKR) depends on various factors, including the area, location, and current real estate market conditions. Here are some general insights: 1. "Location": Prices can vary significantly between different localities in Lahore. Areas like DHA, Bahria Town, and

Type your message...

Admin Dashboard:



Chapter 9: Lessons Learnt and Future Work

Lessons Learnt

Through the design and development of our AI-integrated real estate web application, we acquired essential technical skills, team coordination experience, and real-world development insights:

1. **Full-Stack Integration Skills**

By working with ReactJS on the frontend and NestJS on the backend, we gained hands-on experience in full-stack web development. We successfully connected these technologies using REST APIs with PostgreSQL for persistent data storage.

2. **State Management with Redux Toolkit**

Redux Toolkit helped streamline global state management. This taught us how to optimize frontend performance using data caching and query abstraction.

3. **Map Integration via Google Maps API**

Implementing Google Maps allowed us to visualize properties and inspector locations, enhancing the user experience. We learned to convert real-time location data into meaningful insights.

4. **AI Chatbot with OpenAI Integration**

We integrated a chatbot to assist users in searching for properties and asking questions about buying, renting, or inspections. This taught us about NLP capabilities and user interaction flow in AI services.

5. **Role-Based Access Control (RBAC)**

Designing separate dashboards for sellers, inspectors, admins, and customers improved our understanding of secure user authentication, routing, and dashboard isolation.

6. **API Testing and Debugging**

We tested and debugged endpoints using tools like Postman. This ensured that the backend APIs were reliable and all endpoints responded appropriately with valid input/output.

7. **Team Collaboration & Version Control**

Through tools like Git and GitHub, we collaborated effectively as a team. We also practiced proper documentation, code reviews, and continuous integration workflows.

Future Work

To further improve and evolve the real estate web platform, we propose the following future enhancements:

1. **Real-Time Chat System**

Integrate a secure real-time messaging module (e.g., using WebSockets) to allow users, sellers, and inspectors to communicate directly within the platform.

2. **Mobile Application**

Create a cross-platform mobile version using React Native for improved accessibility and user reach.

3. **AI-Driven Recommendations**

Upgrade the chatbot to suggest properties or services based on user behavior and previous interactions using machine learning.

4. **Multilingual Support**

Add language localization for Urdu, Arabic, and French, making the platform more accessible to global users.

5. **Document Upload & Verification**

Allow inspectors and sellers to upload documents (ownership proof, certification, etc.) for verification by the admin.

6. **Advanced Admin Analytics**

Implement analytics dashboards for the admin to monitor user activity, inspector approvals, most searched locations, and fraud detection.

References

Books

- [1] B. Hogan, *React Explained: A Full-Stack Guide to React Framework*, San Francisco, CA: Code4Coders Press, 2020.
- [2] D. Schell, *Learning PostgreSQL*, Birmingham: Packt Publishing, 2018.

Book Chapters

- [3] A. K. Singh, “Introduction to NoSQL and Relational Databases,” in *Modern Database Systems*, J. P. Gupta, Ed. New York: Wiley, 2021, pp. 45–72.

Electronic Books

- [4] T. R. Miller. (2024, Aug. 10). *Mastering NestJS for Backend Development* [Online]. Available: <https://nestjsbook.com/>

Journal Articles

- [5] R. Malik, “Google Maps API and its Application in Smart Web Apps,” *Int. J. of Web Engineering*, vol. 13, no. 4, pp. 112–118, Nov. 2023.

E-Journal Articles

- [6] M. Khan. (2024, Mar.). “AI Chatbots in Real Estate Platforms,” *Journal of Digital AI* [Online]. vol. 18, no. 1, pp. 60–68. Available: <https://journalofdigitalai.com/real-estate-bots>

Conference Papers

- [7] F. Ahmed and N. Qureshi, “Location-Based Property Search Using Google Maps API,” in *Proc. 2023 Int. Conf. on Smart Technologies*, Lahore, Pakistan, 2023, pp. 88–92.

Theses / Dissertations

- [8] H. R. Ali, “A Role-Based Real Estate Web Platform Using Full-Stack Technologies,” B.S. thesis, Dept. Software Eng., COMSATS Univ., Islamabad, Pakistan, 2025.

Online Documents

- [9] NestJS. “NestJS: A progressive Node.js framework” [Online]. Available: <https://docs.nestjs.com/>

[10] OpenAI. “OpenAI API Documentation” [Online]. Available:
<https://platform.openai.com/docs/>

[11] PostgreSQL. “PostgreSQL Official Docs” [Online]. Available:
<https://www.postgresql.org/docs/>

[12] Redux Toolkit. “Redux Toolkit Documentation” [Online]. Available: <https://redux-toolkit.js.org/>

[13] Google Developers. “Google Maps JavaScript API Documentation” [Online]. Available:
<https://developers.google.com/maps/documentation/javascript>

Appendix

The appendix contains supplementary materials that support the implementation of the real estate platform. These documents are too detailed to be included in the main chapters but are essential for full system understanding.

Appendix A: Database Schema

- Entity Relationship Diagram (ERD)
- Tables:
 - Users
 - Properties
 - Services
 - Contacts

Appendix B: Sample Code Snippets

- NestJS sample route for creating a property
- ReactJS sample for dynamic filtering with Redux Toolkit Query
- Google Maps API initialization and location markers
- OpenAI chatbot integration code with response management

Appendix C: UI Snapshots

- Home Page with property filters
- Inspector Dashboard
- Seller Panel CRUD form
- Google Map with Inspector pins
- Registration and login forms

Appendix D: Test Cases

- Black-box tests for registration and login
- White-box tests for API request validation
- Integration test for role-based dashboard redirects