Software Requirements Specification

for

Real-Estate Broker

Version 1.0

Prepared by

Group: 14 **Group Name: Dangling Pointers**

Chinmay Hiran Pillai	**200298**	**chinmay20@iitk.ac.in**
Shivang Pandey	**200941**	**shivangp20@iitk.ac.in**
Armeet Singh Luthra	**200185**	**armeet20@iitk.ac.in**
Arth Banka	**200191**	**arth20@iitk.ac.in**
Vaishali Rawat	**201080**	**vaishalir20@iitk.ac.in**
Utpal Dwivedi	**211132**	**utpaldwi21@iitk.ac.in**
Praveen Raj	**210763**	**praveenr21@iitk.ac.in**
Abhishek Kumar	**210039**	**akumar21@iitk.ac.in**
Pravallika Mudunuru	**220814**	**pmudunuru22@iitk.ac.in**
Gaddam Shiva Leela	**220392**	**gaddamshiv22@iitk.ac.in**

Course: CS253

Mentor TA: *Vaibhav*

Date: 26/01/2024

Index

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Chinmay Hiran Pillai, Shivang Pandey, Armeet Luthra, Arth Banka, Vaishali Rawat, Abhishek Kumar, Praveen Raj, Utpal Dwivedi, Pravallika Mudunuru, Gaddam Shivaleela	Basic SRS for Real-Estate Broker	26/1/24

1 Introduction

1.1 Product Scope

Our website is an innovative real estate platform specifically designed to streamline and empower the buying and selling of properties. Our primary focus is on connecting savvy property owners and property seekers, eliminating the hurdles and uncertainties often associated with property transactions.

Our platform empowers you to:

- Discover the perfect plot: Explore a curated selection of property options, with detailed information on location, size, availability, and potential development.
- Compete with confidence: Place bids or auctions your own plots in a transparent and secure environment, ensuring fair market pricing and maximizing your profit potential.
- Stay informed, stay ahead: Watchlist your desired properties, receive updates on bidding activity, and track market trends with dynamic pricing that adjusts based on demand.
- Manage your portfolio: Easily access your transaction history, track your invested funds, and control your financial activity with a dedicated portfolio section.
- Expand reach: Leverage currency conversion features to expand your reach and deal in foreign currency.

Our platform will initially focus on facilitating the buying and selling of properties for its users. The platform will prioritize user experience, security, and transparency, ensuring a smooth and reliable journey for all users.
We are excited to bring our platform to life and empower individuals to take control of their real-estate investments.

1.2 Intended Audience and Document Overview

This document caters to a multi-faceted audience:

- Development Team: Comprised of students within the Dangling Pointers project team, responsible for translating requirements into functional software.
- Testing Team: Composed of the students tasked with evaluating adherence to requirements and ensuring software quality.
- Documentation Team: The Dangling Pointers student-led group responsible for generating clear and concise technical documentation.
- Users: Encompasses the people, who will ultimately utilize the developed software.
- Course Instructor and TA Mentor: Serve as key stakeholders providing valuable feedback and guidance throughout the project lifecycle.

The document structure outlines a comprehensive roadmap for the project, encompassing:

- Section 2: Project Overview: Delves into the overall vision, functionality, and design principles of the software.
- Section 3: User Interfaces and Specifications: Details the interaction paradigms for users, hardware and software interfaces, and elaborates on the functional requirements of the software.
- Section 4: Non-Functional Requirements: Focuses on critical aspects beyond functionality, including safety, performance limitations, and software quality attributes.

This comprehensive approach aims to equip all stakeholders with a clear understanding of the project's objectives, technical specifications, and expected outcomes.

1.3 Definitions, Acronyms and Abbreviations

API (Application Programming Interface): A set of routines and protocols that allow other software to interact with the specified system.

Backend*:* The server with which the website communicates with in order to query and store data.

Broker*:* The web application that acts as an intermediary for buying and selling a property.

Database: The central store of data that the backend server communicates with to read and write data as per the website's needs.

Functional Requirement: A specific statement describing what the software must do. Examples: "The system shall allow users to log in using their username and password."

Non-Functional Requirement: A requirement that defines qualities of the software beyond its functionality, like performance, usability, security, etc. Examples: "The system shall respond to user actions within 2 seconds."

Property: Land, apartment or house which can be traded on the platform.

SRS (Software Requirements Specification): A detailed document outlining the functional and non-functional requirements of a software system.

UI (User Interface): The elements and features that users interact with, such as buttons, menus, and screens.

Use Case Diagram: A use case diagram is a roadmap for a user journey, detailing the steps they take to achieve a specific goal within a system. It's essentially a story from the user's perspective, showing how they interact with the system and what happens in response.

1.4 Document Conventions

The document adopts Arial with a size of 11 points as the default typeface, excluding headings. Italic styling denotes placeholders for variables, to be substituted with contextually appropriate values. Headings and subheadings utilize bold typography for emphasis. Subheadings nested within a heading designated as H are numbered as H.x, where x represents their ordinal position among siblings. This recursive numbering scheme persists for further nested subheadings

1.5 References and Acknowledgments

References Used:

https://www.geeksforgeeks.org/software-requirement-specification-srs-format/

https://www.utdallas.edu/~chung/RE/Presentations07S/Team_1_Doc/Documents/SRS4.0.doc

https://jelvix.com/blog/software-requirements-specification

2 Overall Description

2.1 Product Overview

The website we're proposing is a simple platform to buy and sell properties online. It's akin to a broker application in that it allows the user to buy and place bids on and/or auction off and sell properties in a convenient fashion.

The idea is to integrate dynamic pricing and showcase the present state of the market by giving consumers the power to compare and watch out for the best deals while the sellers can maximise their profits in much the same way. Features like currency conversion would ensure accessibility and translation to a wide consumer base.

One would create an account by giving a username, password, mailing id and primary contact information. Upon successful registration, they would be taken to the product page where various properties will be listed along with details like location, sizing, number of available units, et all.

Consequently, they would be able to bid on plots/ place their properties for sale. A watchlist feature will ensure that the consumer stays updated about the status of particular plots he/she might be interested in. An order book shall also be maintained showcasing all the presently bid upon units of land. In addition to this, every user would have a portfolio which would showcase the properties one has bought/sold. Each consumer can access a funds page which would indicate the present state of their funds according to their purchasing/ selling activity and this can be added to (via a payment) or withdrawn from. Dynamic pricing shall be implemented by using the number of property units sold to gauge the price of the remaining units and will be updated every day.

2.2 Product Functionality

The primary functionality that we want to achieve is convenient buying and selling of properties:

- There will be a product page wherein one can browse through the plots that are currently available for sale (this would have basic details about the property such as location, sizing, number of units sold, etc.)
- The user will be able to place bids and/ or buy the product at the best available price
- The user can also auction off and sell a property
- A watchlist feature will allow the consumer to keep track of/ view the fluctuating prices of certain properties.
- A portfolio will be maintained for each consumer listing their bought and sold properties.
- An order book will show the current buy and sell bids on a specific property
- Currency conversion shall be implemented to ensure smooth transactions between a widely varied consumer base.
- Funds for each user would also be maintained for ease of payment, this will incorporate addition and subtraction thereof whenever there's a final purchase or sale

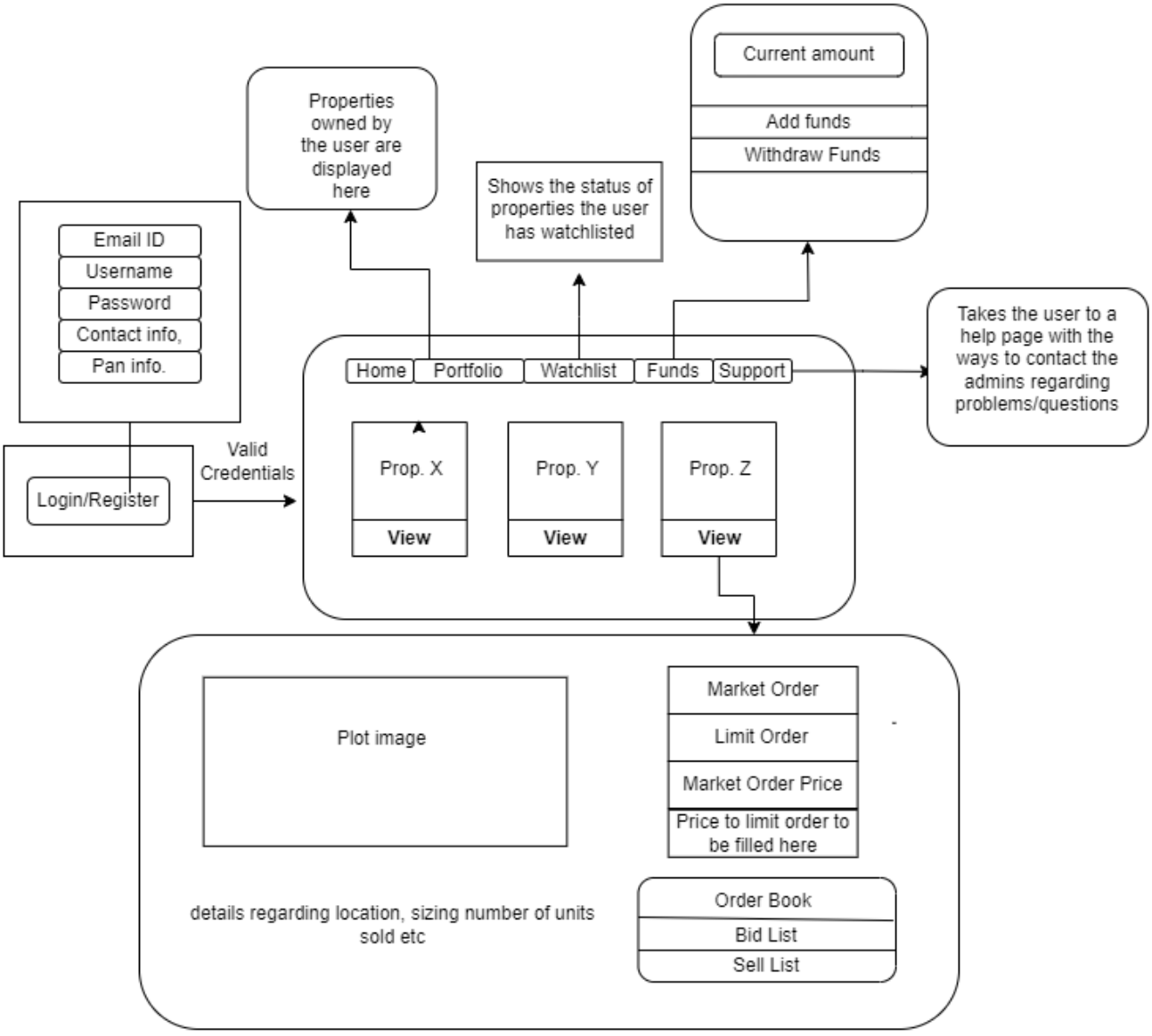

Fig. (2.1.1)- Product Overview and general functionality

2.3 Design and Implementation Constraints

- Users must register with their PAN card number and undergo eKYC so that they can be verified and avoid fraud.
- In some cases, quality and information of the property listings may or may not be accurate. So, we suggest users to check personally, before buying.
- Limited availability of property listings in certain geographical areas.
- Sometimes, workability of the website can be affected by third party apps like Paytm.

2.4 Assumptions and Dependencies

2.4.1 Assumptions

- Users are assumed to use a PC or mobile with a browser and stable internet connection.
- Users visit the website with the intent to buy or sell properties.
- The volatility and liquidity in real-estate is assumed to be low enough that any price changes within 1 minute are negligible and can be ignored.

2.4.2 Dependencies

- The user's identity and PAN card number will be verified through eKYC software.
- Paytm payment gateway for is utilised for adding funds into the broker from bank account.
- The broker depends on its database which stores all user and broker data.
- The broker utilises the HTTPS protocol to send requests and receive responses from the backend server and database.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

1. **Login/Register Page:** The web application will first prompt the user to register an account with the broker if they haven't already and login once registration is complete.
2. **Home Page:** Once logged in, the application will display all the properties that are on sale to the user.
3. **Product Pages:** Displays details and best buy and sell price for the property.
    - **a.** *Market Order Button:* Allows the user to place a buy/sell market order on the property.
    - **b.** *Limit Order Button and Price Field:* Allows the user to place a buy/sell limit order(bid) on the property at a particular price.
    - ***c.*** *Order Book***:** Displays the 5 highest buy bids and the 5 lowest sell bids on the property.
    - ***d.*** *Watchlist Button:* Adds the current property to watchlist.
4. **Navigation Bar:** Allows the user to navigate to the Home, Portfolio, Watchlist, Funds a Support pages.
5. **Portfolio Page:** Displays all the properties owned by the user.
6. **Watchlist Page:** Displays all the properties watchlisted by the user.
7. **Funds Page:** Displays the user's available funds.
    a. *Add Funds: Allows the user to add funds into the broker from bank account.*
    b. *Withdraw Funds: Allows the user to withdraw funds from the broker to bank account.*
8. **Support Page:** Provides the user with important contact information to address issues and complaints.

The users will first land on the login page from where they can click on the register button to go to the register page. Here, they will register an account with the broker. Then they will be redirected back to the login page. After logging in, they will be brought to the home page that displays all the listed properties. Clicking on any property will open the property's page with details regarding the property and the option to place a market or limit order.

The navigation bar on top can be used to navigate to all the pages. Portfolio page will give details regarding all the properties owned by the user. The watchlist page will allow the user to view all the properties watchlisted by them. The funds page will show the user the funds they have transferred onto the broker. It also allows them to add or withdraw funds from the broker. The support page provides the user with the contact information to reach out customer support for queries or resolving issues.

Fig 3.1.1 - Login Page

Fig 3.1.2 - Register Page

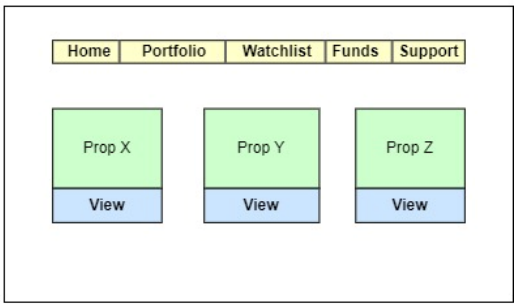

Fig 3.1.3 - Home Page

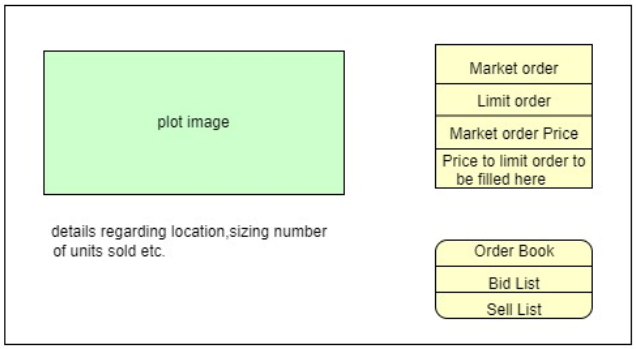

Fig 3.1.4 - Property Page

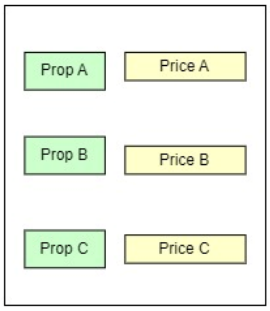

Fig 3.1.5 - Watchlist Page

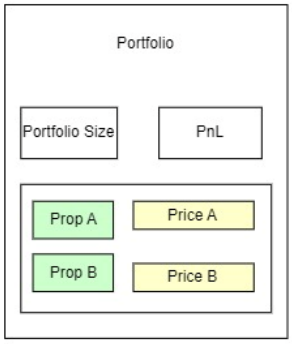

Fig 3.1.6 - Portfolio Page

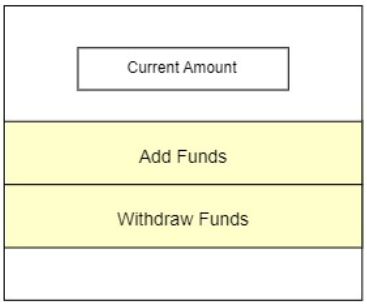

Fig 3.1.7 - FundsPage

Fig 3.1.8 - Customer Support Page

3.1.2 Hardware Interfaces

3.1.2.1 Physical Interfaces

1. Computer system with internet access and browser at admin terminal to add new properties after verifying property ownership
2. Any PC or mobile with internet access and browser for customers to access broker application to buy/sell properties.
3. Mobile to verify payment OTP from bank to broker while adding funds.

3.1.2.2 Logical Interfaces

1. The data transfer between the frontend client and backend server will take place through HTTPS protocol using RESTful API.

3.1.3 Software Interfaces

1. Payment APIs and bank software to add/withdraw funds from broker to bank account.
2. eKYC API to authorise customer to buy and sell properties.
3. All property data will be stored in an ACID compliant database. Each user will only have access to their own data and the properties for sale.

3.2 Functional Requirements

3.2.1 Property Display

- *The application must display a comprehensive list of all available properties.*
- *Users should be able to navigate to a page providing further details by clicking on their desired property.*

3.2.2 eKYC

- *The broker must only allow users to interact with the application after eKYC procedure*

3.2.3 Market Order Option

- *Users must be allowed to execute market orders to buy/sell at the current best available sell/buy price.*
- *If funds are available, the system must automatically subtract them on buy.*
- *If the required amount of funds is not available, the system must provide a 'no funds' alert. The buy order must not go through unless the required amount of funds are present.*
- *On a successful sell, the system must add funds to the user's account.*

3.2.4 Limit Order Option

- *The platform should enable users to place buy/sell limit orders at specified prices.*
- *The bids must get added onto the order book.*
- *If the required amount of funds is not available, the system must provide a 'no funds' alert. A buy order must not go through unless the required amount of funds are present.*
- *The system must automatically subtract funds when the buy order goes through.*
- *On a successful sell, the system must add funds to the user's account.*

3.2.5 Order Book

- *The system should display the current best 5 buy and best 5 sell bids for a specific property.*

3.2.6 Property Details

- *The platform should show the location, size, price and other necessary details about the properties.*
- *The price of every property must be updated every minute so as the avoid stale data.*

3.2.7 Portfolio Overview

- *The system must display all properties that the user has purchased.*
- *The platform should show the quantity and relevant details for each owned item.*

3.2.8 Watchlist Properties

- *The system should allow users to add any property they are interested in to a watchlist.*
- *The platform should have a page that shows only the properties watchlisted by the user.*
- *The platform should display the current prices of the items on the watchlist.*

3.2.9 Funds

- *The platform must show the current balance of funds in the user's account.*
- *The system must enable users to add funds to their account.*
- *The platform should redirect users to a payment portal to adding funds to their account.*
- *The system must allow users to withdraw funds from their account.*

3.2.10 New Properties

- *The system should allow the admin to add new properties to the list of available products.*

3.2.11 Customer Support

- *The system should provide basic contact information for customer support that users can reach out for assistance or queries.*

3.3 Use Case Model

3.3.1 Use Case #1 (UC1)

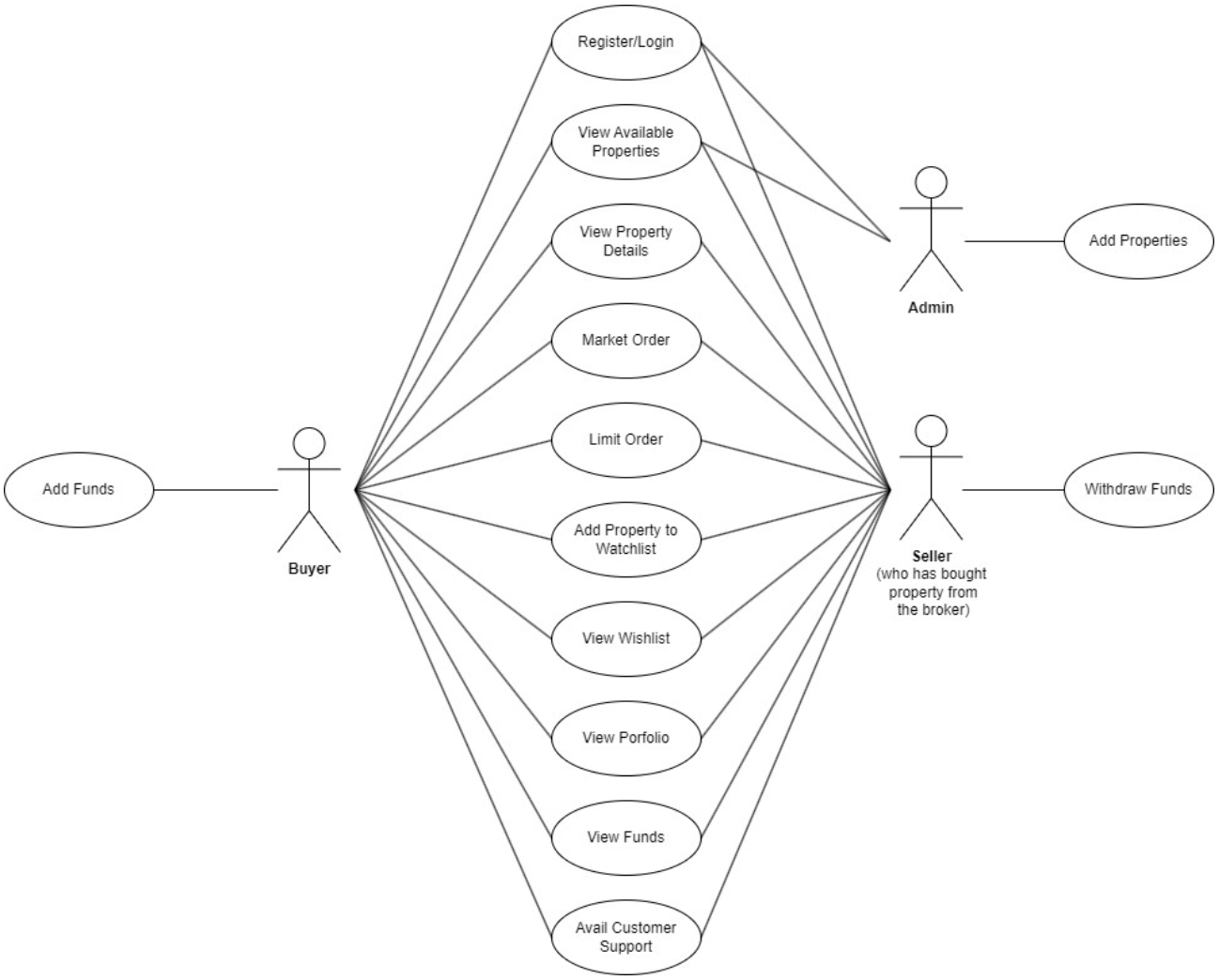

Fig: (3.3.1) - Use Case Diagram of the system

Author – Chinmay Hiran Pillai

Purpose – The use case gives a high-level overview of all the actors in the system and interaction between them.

Requirements Traceability – Traced in the lines of user interface requirements and functional requirements.

Priority – This use case has high priority since it involves all the high-level requirements required for the system to function.

Preconditions – The admin needs to add properties onto the broker application before a user can buy it.

Post conditions - The property is transferred from the seller's portfolio to the buyer's portfolio

Actors – Buyer, Seller and Admin

Exceptions -

1. Payment gateway to add funds may fail
2. Server or Database may fail to send required information in time.

Includes - UC1 is a stand-alone use case diagram.

Notes - Here seller refers only to those sellers who have previously bought and hence currently own properties within their broker account.

3.3.2 Use Case #2 (UC2)

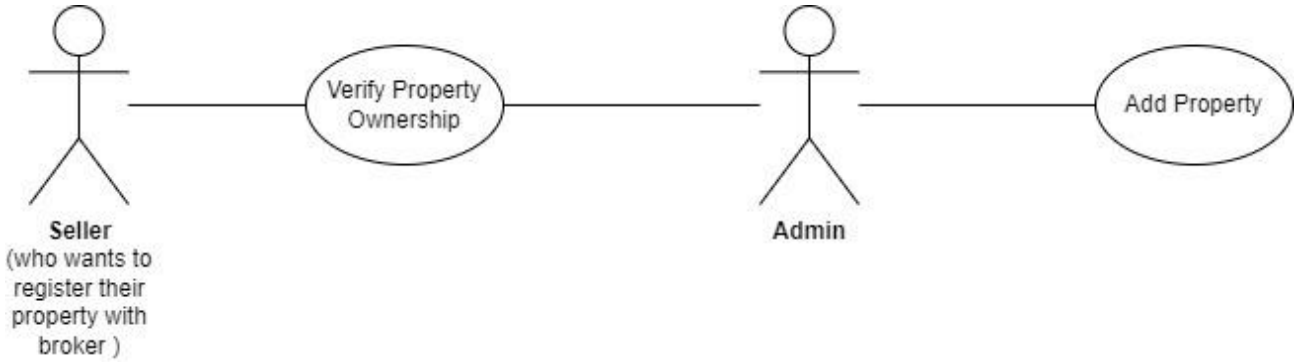

Fig. (3.3.2) - Use Case diagram for adding new properties into broker

Author – Chinmay Hiran Pillai

Purpose – The use case gives a high-level overview of all the actors involved in adding new properties to the broker.

Requirements Traceability – Traced in the lines of functional and non-functional requirements.

Priority – This use case has medium priority since initially it is essential to add properties into the broker but once the application is functional, adding more properties isn't a critical requirement.

Preconditions – The seller must have the proper documentation to prove ownership of the property.

Post conditions – The property will be added onto the broker's available properties list.

Actors – Seller and Admin

Exceptions -

1. The seller doesn't have ownership of the property they are trying to sell.
2. The seller has fake documents showing ownership of the property.

Includes - UC2 is a stand-alone use case diagram.

Notes - Here seller refers only to those sellers who do not have their properties on the application yet and would like to sell their properties on through the broker.

4 Other Non-functional Requirements

4.1 Performance Requirements

4.1.1 Property Listing Page Load Time:
The system should respond to requests for property listings within a maximum of 3 seconds as faster page load times improve user satisfaction and engagement.

4.1.2 Property Purchase Transaction processing time:
Property transactions, including purchase and resale, should be processed within 5 seconds to enhance the overall user experience, especially during high traffic periods.

4.1.3 Concurrent Users
Property Listing:
The system should support at least 50 concurrent users browsing property listings without a significant degradation in performance.
Property Purchase Transaction:
The system should handle at least 10 concurrent property purchase transactions without a significant degradation in performance.

4.1.4 Database Response Time:
Database queries must return results within 2 seconds, ensuring efficient retrieval of property and user data and contribute to a responsive and seamless user interface.

4.1.5 Currency Conversion Update Frequency:
Currency conversion rates must be updated daily to ensure accurate and real-time dynamic currency conversion. Regular updates prevent discrepancies in currency values, providing users with accurate pricing information.

4.1.6 Atomicity in Database:
The database must enforce atomicity to prevent simultaneous purchases of the same unit by different buyers. In case of multiple people placing order for the same property then as soon as one enters the payment gateway other orders will be blocked till the order is completed and the status update. This would ensure data integrity and prevents conflicts in property ownership.

4.1.7 Cross-device compatibility
The application should be fully functional on accessing from multiple devices such as mobile, tablet or laptop of different resolutions. This is to ensure user experience is good.

4.2 Safety and Security Requirements

4.2.1 User Authentication
Login Security:
Users must authenticate using a valid email address and password. Passwords must be securely stored using standard encryption techniques. Failed login attempts should be limited to 3 within a 5-minute period to prevent brute force attacks.

4.2.2 Data Protection

Personal Information:

Personal information such as email addresses and PAN card details (must be stored securely. Use HTTPS to encrypt data during transmission between the client and the server.

4.2.3 User Authorization

Role-Based Access:

Buyers/Sellers and admin should have distinct roles with specific permissions.

4.2.4 Session Management

Cookie-based authentication:

Implement cookie-based session management to ensure that user sessions expire after a reasonable period of inactivity, but not expire after refreshing, or after logging in.

4.2.5 Secure Payment Processing:

To Ensure secure handling of payment information during transactions. For this integration with reputable payment gateways and encryption of payment is required.

Rationale

These basic safety and security requirements are intended to safeguard user data, ensure secure access to the platform, and prevent unauthorized access. The focus is on fundamental security practices to protect user accounts and sensitive information.

4.3 Software Quality Attributes

4.3.1 Usability

User Interface:

The user interface should be simple and intuitive, allowing authorised users to navigate the platform and access features without confusion

Error Handling:

Clear and concise error messages should be provided to users in case of input errors or system issues. This would guide users on how to rectify the issue or seek assistance.

4.3.2 Reliability

The platform should aim for 24x7 availability during standard operating hours. Based on the limited number of users as well as limited database size, we can ensure the reliability and stability of the application through failsafe implementations.

4.3.3 Maintainability

Code should be adequately documented with a modular architecture to facilitate ease of understanding and future maintenance.

4.3.5 Interoperability

The systems will interact with a database that will in turn interact with other systems. This will be ensured by building our software with a common standard and using the same Tech stack across multiple systems.

4.3.4 Robustness

If the connection between the user and system is broken prior to an order being confirmed, the platform shall enable the user to recover the incomplete order.

4.3.5 Testability

It will be ensured by passing valid input and invalid input to check the reliability of the software. For example, on entering invalid password, what kind of error response we get will be tested. It is to be made sure that even after entering invalid input we should not be granted access. For all testing (response times, page load time, authentication and atomicity) at least 98% accuracy is required. This is to be tested by sending 100 requests and calculating the % of desired response obtained.

Rationale

These quality attributes aim to enhance the user experience, ensure the reliability of the platform, facilitate future maintenance, and optimize system performance. Usability testing, availability targets, and maintainability practices are crucial for a successful prototype that aligns with user expectations and industry standards.

5 Other Requirements

5.1 Legal Requirements:

5.1.1 Property Ownership:

- *The admin must verify the legal ownership of the property using government verified documents before listing them on the application.*

5.1.2 eKYC Software:

- *The software utilised for eKYC process must be from a reputable and government verified source.*

Appendix A – Data Dictionary

TERMS	DESCRIPTION
API (Application Programming Interface)	A set of routines and protocols that allow other software to interact with the specified system.
Backend	The server with which the website communicates with in order to query and store data.
Broker	The web application that acts as an intermediary for buying and selling a property.
Database	The central store of data that the backend server communicates with to read and write data as per the website's needs.
Functional Requirement	A specific statement describing what the software must do. Examples: "The system shall allow users to log in using their username and password."
Non-Functional Requirement	A requirement that defines qualities of the software beyond its functionality, like performance, usability, security, etc. Examples: "The system shall respond to user actions within 2 seconds."
Property	Land, apartment or house which can be traded on the platform.
SRS (Software Requirements Specification)	A detailed document outlining the functional and non-functional requirements of a software system.
UI (User Interface)	The elements and features that users interact with, such as buttons, menus, and screens.
Use case	A use case is a roadmap for a user journey, detailing the steps they take to achieve a specific goal within a system. It's essentially a story from the user's perspective, showing how they interact with the system and what happens in response.
UX (User Experience)	The overall experience a user has while interacting with the software.

Appendix B – Group Log

Meeting 1: 9 Jan 7pm-8pm
Discussed the various ideas for the project pitched by the team members and shortlisted 3 ideas – course review site, gym membership portal, e-commerce website.
Meeting 2: 12 Jan 4pm-5pm
Discussed the tech stack with the group members and the features feasible to implement in project timeframe.
Meeting 3: 15 Jan 8:30pm-10pm
Met with the startup founder (Dharambir) to get an overview of his project idea and brief outline features.
Meeting 4: 19 Jan 10pm-12am
Met with the startup (Dharambir) to finalise the features and implementation details in full. The expectations from the start-up were unrealistic high, given the limited time of the project. Subsequently, we requested the professor to change our project and allot us a new TA,
Meeting 5: 20 Jan 6pm-7pm
Met with the new TA (Vaibhav) to discuss our project, functional requirements and implementation details, finalising all the details.
Meeting 6: 22 Jan 9pm-10pm
Met with the group to start working on the requirement documentation. Division of work was done and sections were divided among the team members.
Meeting 7: 25 Jan 10pm-11pm
The group met up to review their contributions to the requirement document and ensure consistency between all the sections.