Software Requirements Specification

for

PropBidder

Version 1.0 approved

Prepared by Qi En, Spencer, Yong Khang, Atul, Naveen, Shreya

SCSB Team 4

15/04/2025

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document outlines the features, functionalities, and technical requirements of the **PropBidder Web Application**. Its purpose is to provide a comprehensive understanding of the system's objectives, scope, and specifications for all stakeholders involved. **PropBidder** is a web-based platform designed to facilitate seamless interaction between property owners and property agents, enabling them to find the best matches based on commission preferences and service quality. The system ensures a transparent and competitive environment, allowing property owners to make informed decisions when selecting agents. This document serves as a reference throughout the development, testing, and maintenance phases of the project. It includes detailed descriptions of system features, external interface requirements, and non-functional requirements, ensuring that the project objectives are clearly defined and met.

1.2 Document Conventions

This Software Requirements Specification (SRS) follows specific conventions to ensure clarity, consistency, and ease of use across all sections.

1.2.1 Font and Formatting

- **Header 1:** Font – *Times New Roman*, Size – 18 pt
- **Header 2:** Font – *Times New Roman*, Size – 14 pt
- **Header 3:** Font – *Times New Roman*, Size – 12 pt
- **Header 4:** Font – *Times New Roman*, Size – 11 pt
- **Body Content:** Font – *Times New Roman*, Size – 12 p

1.2.2 Structure and Style

- **Section Headings** follow a hierarchical decimal numbering structure (e.g., 1, 1.1, 1.2, 2.1.1) to reflect the document's organization and to improve navigation.
- **Priority Levels** for requirements are denoted as **High**, **Medium**, or **Low**, and indicate their relative importance to the overall system. Unless otherwise specified, sub-requirements inherit the priority of their parent requirement.
- **Requirement Identifiers** follow the format **REQ-#** (e.g., REQ-01, REQ-02) to facilitate traceability and referencing throughout the development lifecycle.
- **Bolded terms and phrases** are used to emphasize key concepts, major components, and critical actions for enhanced readability.

These conventions are intended to support uniformity and reduce ambiguity, making the document accessible and comprehensible to developers, testers, and stakeholders alike.

1.3 Intended Audience and Reading Suggestions

The document is intended to the following stakeholders:

Developers: To allow them to understand system functionalities, database interactions, and API integrations.

Project Managers: To track feature implementation and overall project scope.

End-Users: To review all system capabilities and ensure alignment with expectations.

QA Testers: To validate requirements through test cases and system testing to ensure optimality.

Regulatory Bodies: To ensure compliance with Singapore's housing and data privacy regulations.

Recommended reading order:

1. **Introduction** (Overview of the system)
2. **Overall Description** (System context and user needs)
3. **System Features** (Detailed functional specifications)
4. **External Interface Requirements** (Integration details)
5. **Non-Functional Requirements** (Performance, security, and usability constraints)

1.4 Product Scope

The Property Agent Bidding Platform is an online marketplace that connects property sellers with real estate agents through a competitive bidding system. Unlike traditional property listing websites, this platform allows sellers to set a maximum commission rate and enables agents to bid for the opportunity to represent the property. The system promotes transparency by displaying agent profiles, including experience, past transactions, client reviews, and commission trends.

1.5 References

Singapore Government Open Data Portal (data.gov.sg) – Source for property transaction data and pricing trends.

2. Overall Description

2.1 Product Perspective

The **PropBidder Platform** is a self-contained, web-based application designed to **streamline and modernize the property agent selection process** through a transparent, commission-based bidding system. Rather than replacing existing property listing websites, PropBidder complements them by optimizing **agent selection based on competitive pricing and service quality**.

Sellers can post their property requirements along with a maximum acceptable commission rate. Real estate agents can then **submit bids** to represent the property, competing based on their proposed commission, transaction history, and reviews. The platform facilitates **data-driven decision-making** by providing sellers with detailed agent profiles and historical commission trends.

2.1.1 Key Integrations and Components

The platform incorporates multiple layers and integrates various services:

- **External APIs**
  - Government APIs such as **HDB Resale API**, **URA datasets**, and **OTP SMS API** for real-time verification and property price insights.
- **Data Layer**
  - Internal databases to store:
    - Agent profiles and reviews
    - Bidding history
    - Property and pricing data
- **Application Logic Layer**
  - Handles all backend functionalities including:
    - User authentication
    - Commission bidding
    - 2FA verification
    - Past price analysis
    - Profile and preference management
- **User Interface Layer**
  - Role-based dashboards for:
    - **Sellers** to post listings and evaluate agents
    - **Agents** to view opportunities and submit bids

2.1.2 System Architecture

The system architecture diagram of the PropBidder web application is illustrated in the diagram below. It captures the interaction flow between **users**, **UI components**, **backend logic**, **data handling**, and **external services**:

2.2 Product Functions

The **PropBidder Web Application** must provide the following core functions to support seamless interaction between property sellers and real estate agents

2.2.1 General User (Unauthenticated)

- **View Sold Properties**
  Visitors can access a public homepage displaying recently sold properties.
- **Display Seller and Agent Information**
  Each sold property card includes the seller's and agent's names for transparency.

2.2.2 Registration & Verification

- **Multi-Role Registration**
  New users can register as either a **Seller** or an **Agent**.
- **Agent Verification with ACEAS API**
  Agents must submit a valid registration number, which is verified in real time via the **ACEAS API**.
- **OTP Verification via Twilio**
  Both agents and sellers must verify their phone number by entering a one-time password (OTP) sent via **Twilio SMS API**.
- **Secure Account Creation**
  Account creation only succeeds after both ACEAS (for agents) and OTP (for all) validations pass.

2.2.3 Seller Functions

- **Seller Dashboard Access**
  - Logout
  - Navigation bar with "List Property" and "View Your Listed Properties"
- **List a property**
  - Input: Flat Type, Town, Street Name, Floor Area, Years Remaining, Max Commission
  - Fetch past resale prices via API to help estimate a suitable listing price
  - Submit listing with custom price
- **Manage Listings**

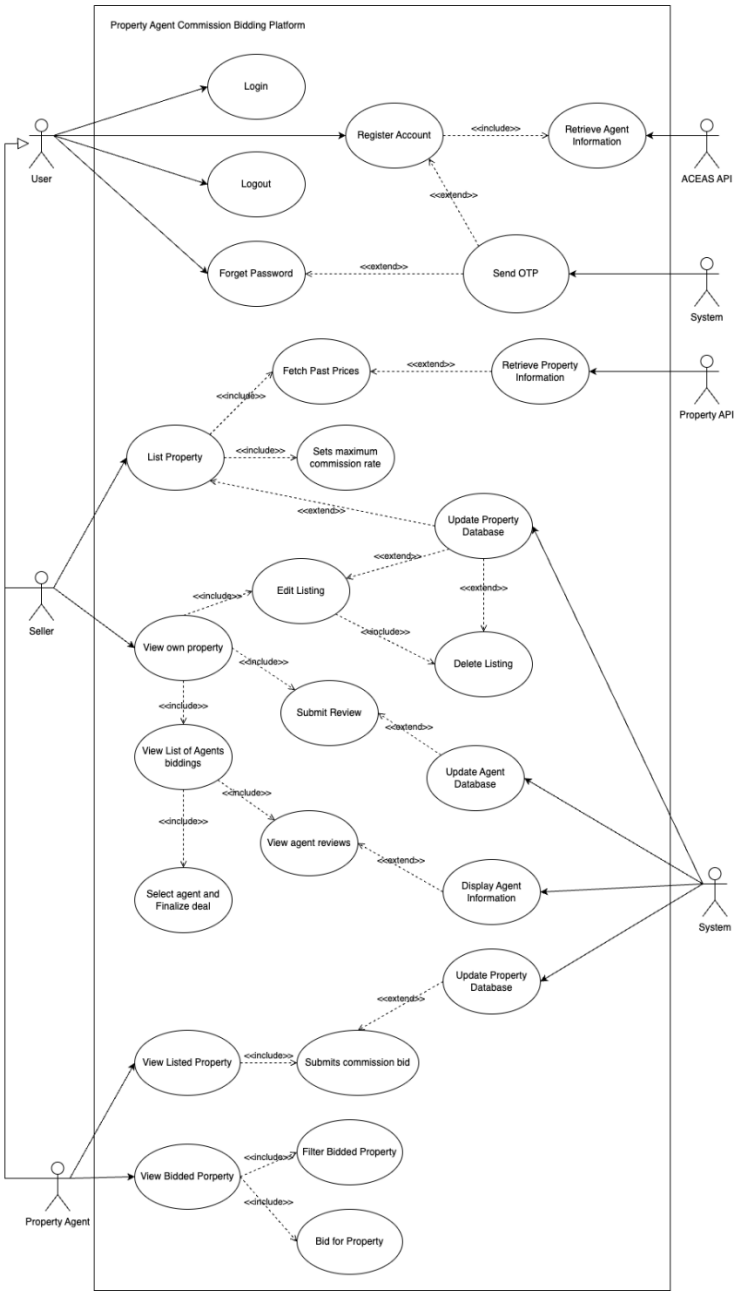
  - View all properties listed by the seller
  - View all bids received for each property
  - Access full agent profiles tied to each bid
  - Accept a bid to finalize agent selection
- **Post-Sale Agent Review**
  - After a sale, leave a review for the selected agent

2.2.4 Agent Functions

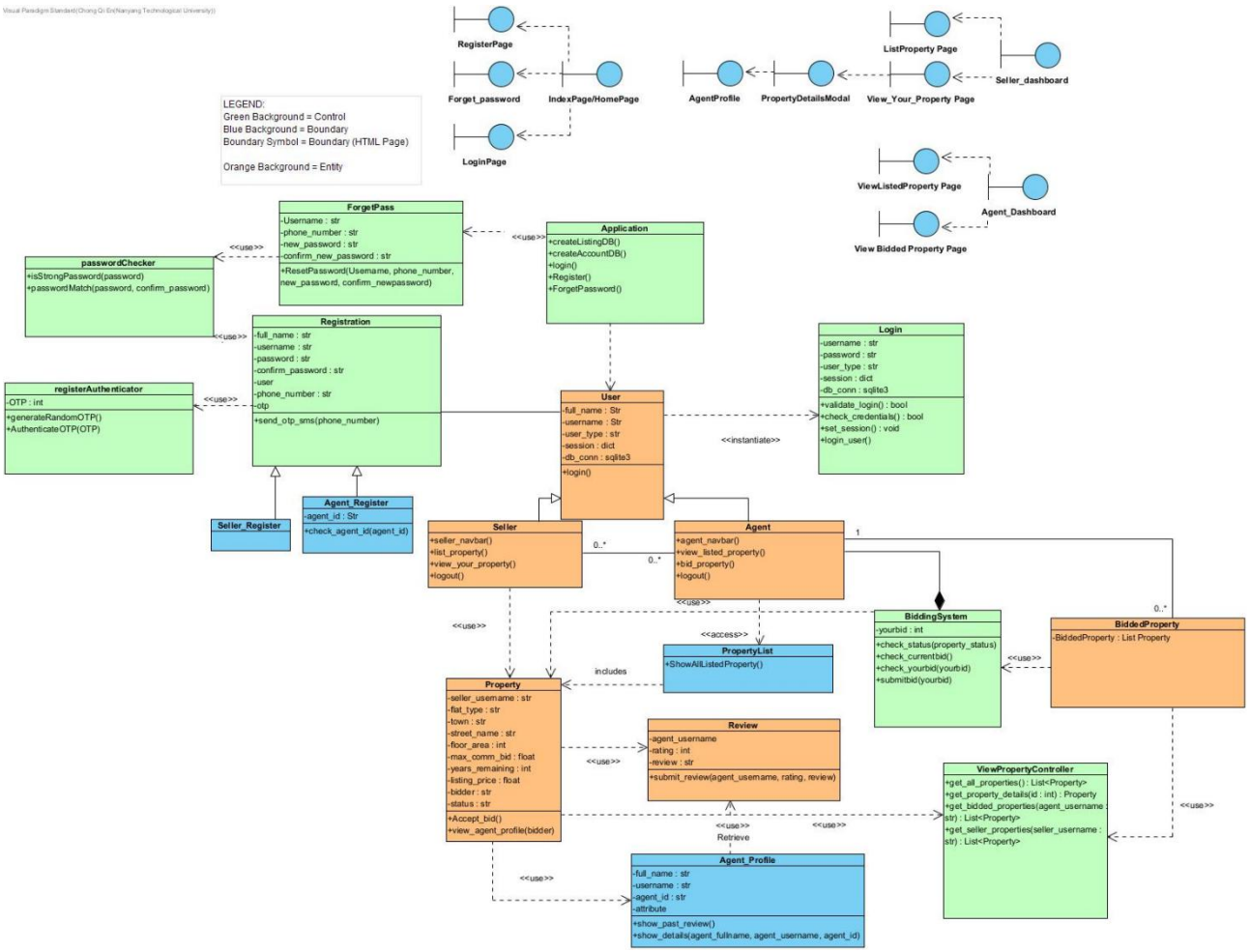
- **Agent Dashboard Access**
  - Logout
  - Navigation bar with “View Listed Properties” and “View Bidded Properties”
- **View Listings**
  - Browse all *active* property listings posted by sellers
  - View full property details, including room type, floor area, listing price, and max commission
- **Submit & Manage Bids**
  - Submit a commission bid below the seller’s maximum
  - View all properties the agent has bid on, with bid status (Active/Sold)
  - View current lowest bid and determine if they are leading
  - If not leading, use “Rebid” to enter a more competitive commission offer
- **Filter Bids**
  - Use filters to display only Active or Sold properties in the *View Bidded Properties* section

2.3 User Classes and Characteristics

The use case diagram and class diagram are shown below to provide a visual representation of the functionalities.



Use Case Diagram



Class Diagram

2.4 Operating Environment

Hardware Platform:
The application will be designed to run on typical user devices such as desktop computers, laptops, and mobile devices. It will be optimized for:

- Desktop and Laptops: Windows, macOS, and Linux platforms.

Operating System:
The application will be fully compatible with the following operating systems for both development and deployment:

- Development Environment: Windows 10/11, macOS, and Linux distributions (Ubuntu, Fedora, etc.)

- Server-side (for deployment): Linux-based systems (Ubuntu preferred) for backend deployment, as Flask and other Python-based services work well on these systems.
- Client-side (for use by end-users): Cross-platform support for Windows, macOS, and major browsers (Chrome, Firefox, Safari, Edge).

Web Server & Deployment:

- Web Server: The application will be hosted on a Flask-based backend, which can be deployed on a Linux server running Apache or Nginx.
- Database Server: The system will use MongoDB for storing non-relational data (e.g., user preferences, transaction history) and PostgreSQL or SQLite for relational data (e.g., pricing trends, flat data).

Software Components:

- Backend: Python 3.x (using Flask as the framework), along with libraries such as:
- PyMongo for MongoDB interaction
- SQLite3 for database management (local storage)
- Flask-Login for user authentication
- Flask-WTF for web forms and validation

- Frontend: HTML5, CSS3, and JavaScript (with frameworks/libraries like React or Vue.js if future dynamic interactions are needed).

- Database:
- MongoDB for storing unstructured data such as trends and user preferences.
- PostgreSQL for structured data such as pricing and flat information.

- API Integrations: The app will connect to public government data APIs (e.g., from data.gov.sg) for resale price trends and other relevant information

Other Tools:

- Version Control: GitHub for version control, collaboration, and deployment processes.

2.5 Design and Implementation Constraints

2.5.1 Regulatory and Compliance Requirements

- All agent registration and bidding mechanisms must comply with the **Estate Agents Act** in Singapore
- System must comply with local data protection laws for user information such as the General Data Protection Regulation and Singapore's Personal Data Protection Act (PDPA).

2.5.2 Performance Constraint

- API calls to external services (e.g., HDB, ACEAS) must have timeouts and error handling mechanisms in place to prevent blocking. The system enforces a timeout of 60 seconds for such calls.

2.6 User Documentation

https://github.com/softwarelab3/2006-SCSB-T4/tree/main

2.7 Assumptions and Dependencies

The following assumptions and dependencies are considered for the development and operation of the **PropBidder** website. These factors are not guaranteed and could impact the project if they are incorrect, not shared, or change during the development process.

2.7.1 Stable Access to Third-Party APIs

- Assumes consistent and reliable access to APIs such as data.gov.sg, ACEAS, and Twilio
- SMS API usage is assumed to fall within free/limited tier plans, with only one virtual number allocated for OTP delivery. Hence, only 1 number (Spencer's phone number) is able to receive the SMS OTP.

2.7.2 Data Availability and Accuracy

- Assumes regular and accurate updates to resale flat data provided by government sources
- The system depends on the availability of fields such as Date, Town, Flat Type, Block, Street Name, Flat Model, Remaining Lease, and Resale Price
- Inaccuracies or lack of data availability may affect the platform's ability to suggest relevant past-price comparisons

2.7.3 Open-Source Library Availability

- Depends on open-source Python libraries for backend operations (Flask, PyMongo, etc.) and frontend compatibility with JavaScript libraries

2.7.4 Client-Side Requirements

- Assumes users access the application using modern, standards-compliant browsers (e.g., Chrome, Firefox, Safari, Edge)

2.7.5 Third-Party API Rate Limits and Costs

- Assumes current usage remains within the bounds of free-tier quotas; future expansion may require budgeting for higher API usage limits

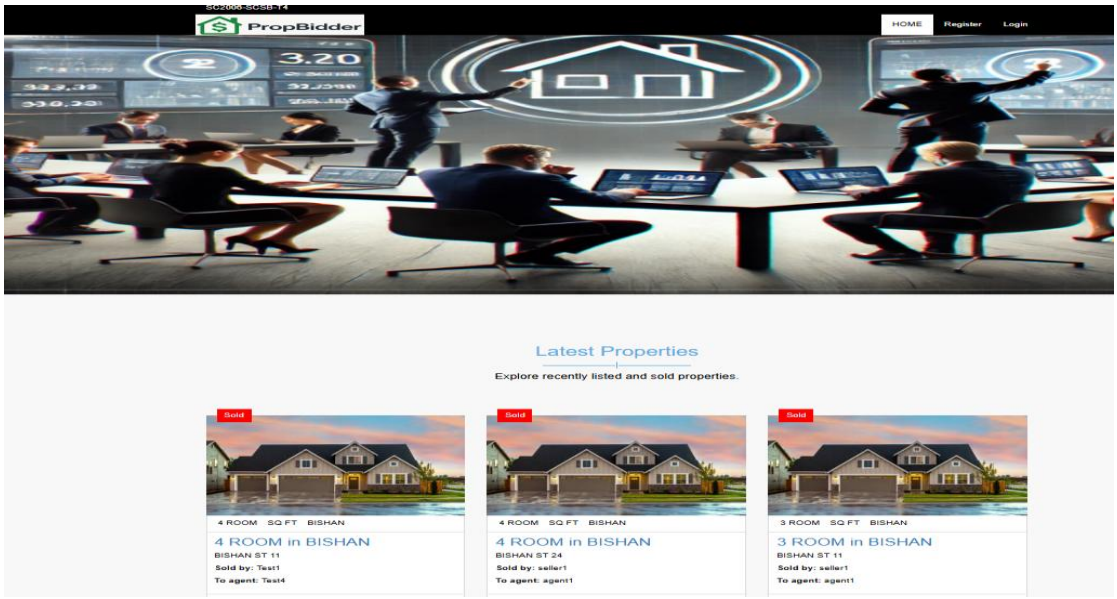
2.7.6 Future Platform Integration

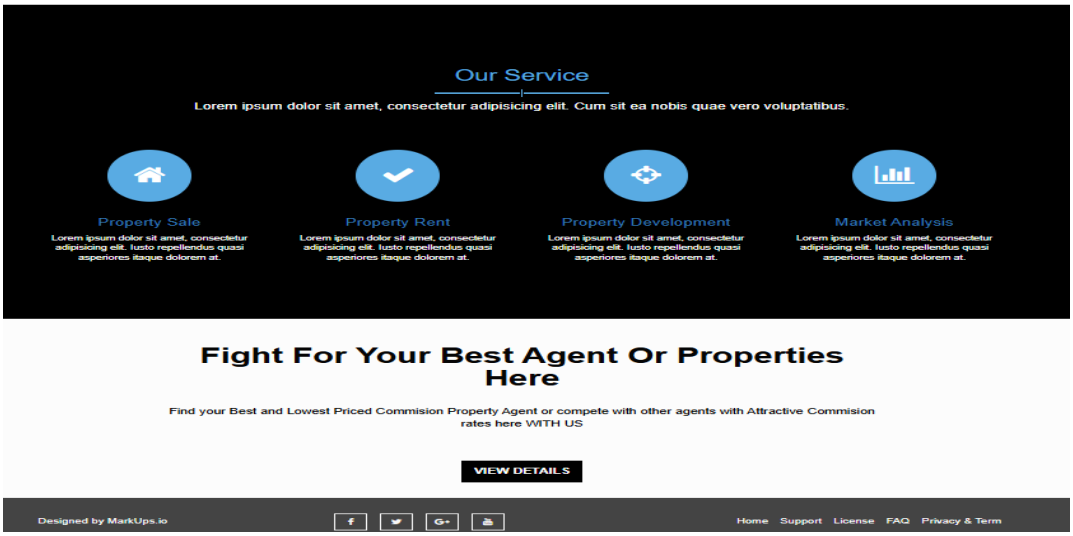
- Potential future integration with real estate platforms is expected but not immediately required

3. External Interface Requirements

3.1 User Interfaces

The PropBidder user interface is designed to provide users a clear and structured experience, offering direct access to all primary features through an intuitive layout. The choice of a dark theme header with contrasting white sections and calming blue highlights offers both aesthetic appeal and navigational clarity





3.1.1 Navigation Bar

The homepage and unauthenticated user views feature a standardized navigation bar at the top header, which includes links to key pages such as Home, Register, and Login. Once authenticated, the navigation bar adapts to the user's role:

- **Sellers** see options like List Property and View Your Listed Properties. This layout allows sellers to post new listings, view their listed properties, and manage agent selections.

- **Agents** see options like View Listed Property and View Bidded Property. This setup helps agents explore property listings and track or revise their bids.

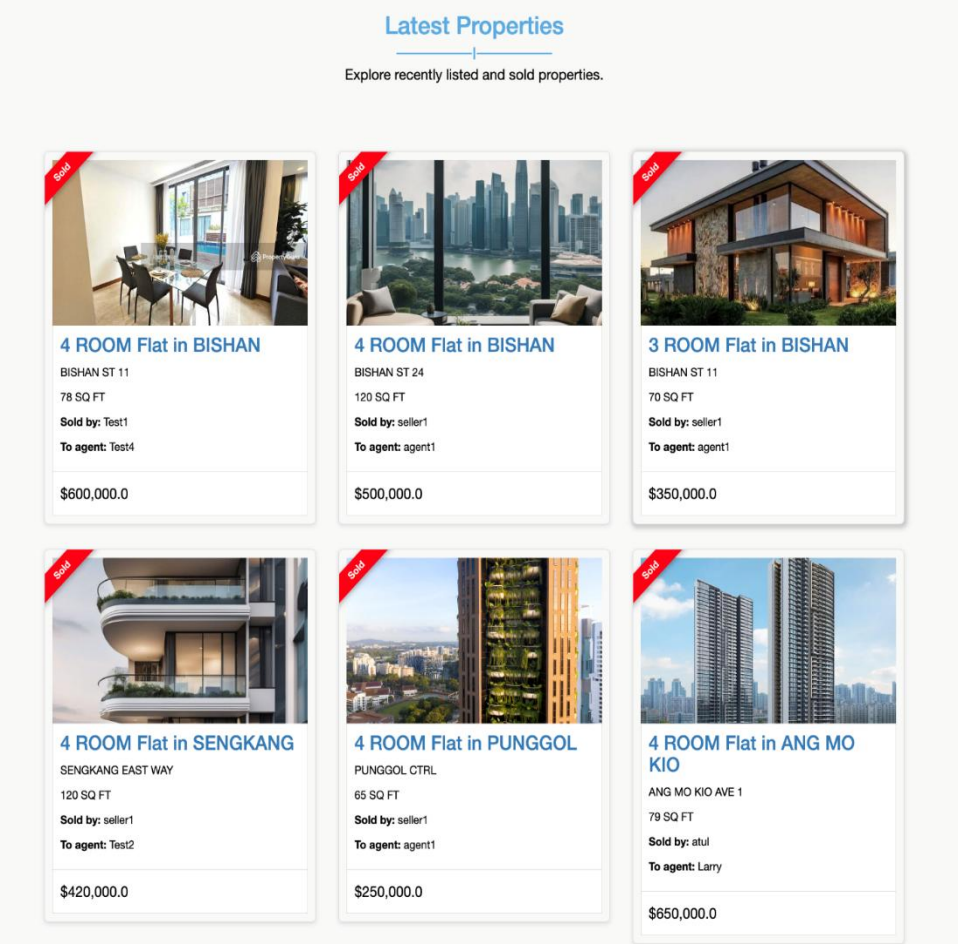
This role-specific navigation ensures that users can access the page's most relevant to their functions. The PropBidder logo in the header is clickable and always routes the user back to the homepage. at the top header.

3.1.2 Property Showcase Grid

A section labeled **"Latest Properties"** displays a grid of recently sold properties for all user types (unauthenticated, agents, and sellers). Each property card contains:

- Flat type, location, street name
- Seller and agent names
- Sold status (with red badge)
- A thumbnail image of the property
- Final price

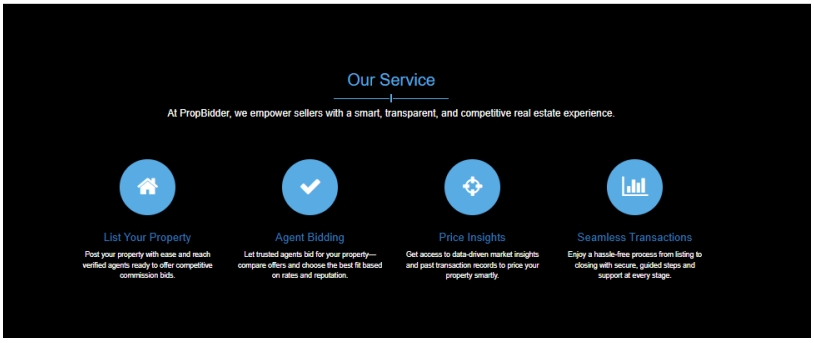
This section enhances transparency and encourages user trust.



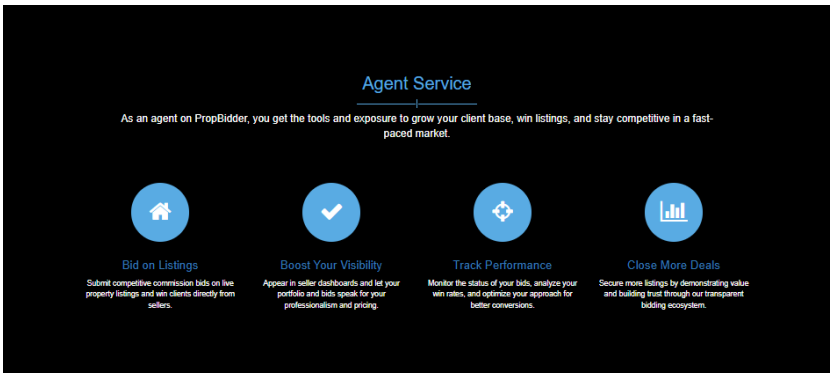
3.1.2.1 Service Highlights

Below the property grid is the **"Our Service"** section, customized according to user roles:

- **For Sellers**:
  - **List Your Property**: Post your property and reach verified agents ready to bid competitively.
  - **Agent Bidding**: Compare commission bids and choose the most attractive offers.
  - **Price Insights**: Get access to data-driven market analysis and past transaction prices.
  - **Seamless Transactions**: Navigate from listing to sale with guided and secure steps.



- **For Agents**:
  - **Bid on Listings**: Submit commission bids on live properties listed by sellers.
  - **Boost Your Visibility**: Appear in seller dashboards with your portfolio and competitive rates.
  - **Track Performance**: Monitor bid performance, win rates, and refine strategies.
  - **Close More Deals**: Build trust and convert leads through transparent bidding.

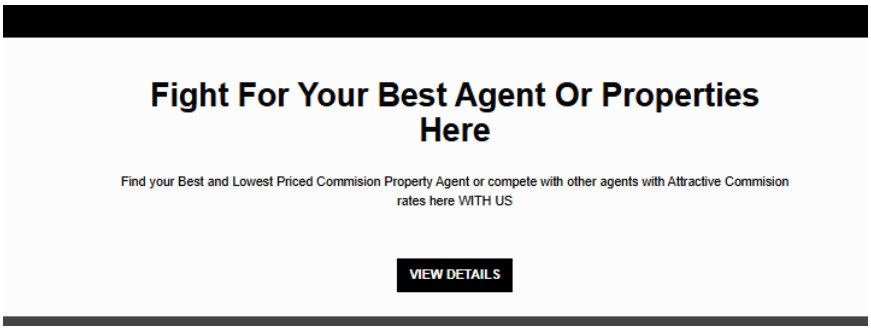


Each service is paired with a circular blue icon and a concise supporting description, maintaining visual consistency and clarity.

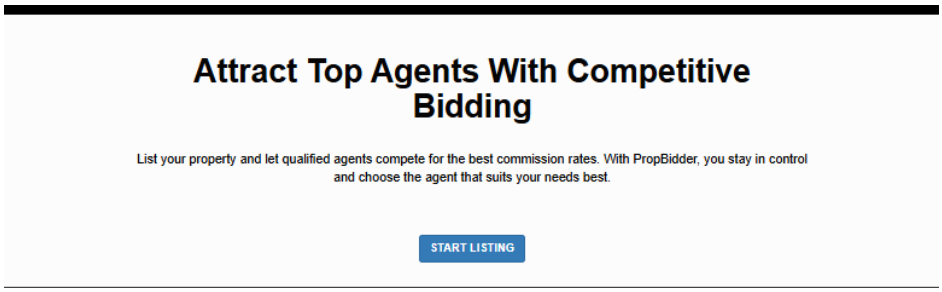
3.1.3 Call to Action

The homepage includes a central call-to-action section tailored by user type:

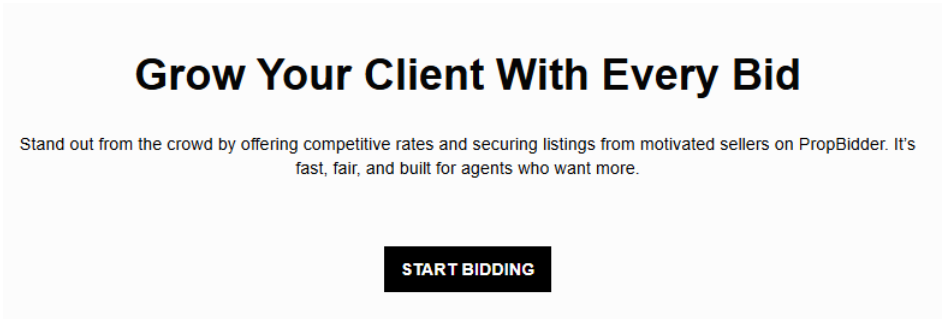
For Unauthenticated Users: The CTA reads *"Fight For Your Best Agent Or Properties Here"*, encouraging them to either register or login to begin engaging with listings or bidding. The button is labeled **View Details**.



- **For Sellers**: The CTA reads *"Attract Top Agents With Competitive Bidding"* and encourages users to post listings to receive commission offers. It is accompanied by a button labeled **Start Listing**.



- **For Agents**: The CTA reads *"Grow Your Client With Every Bid"* and motivates agents to submit competitive rates to win listings. It includes a button labeled **Start Bidding**

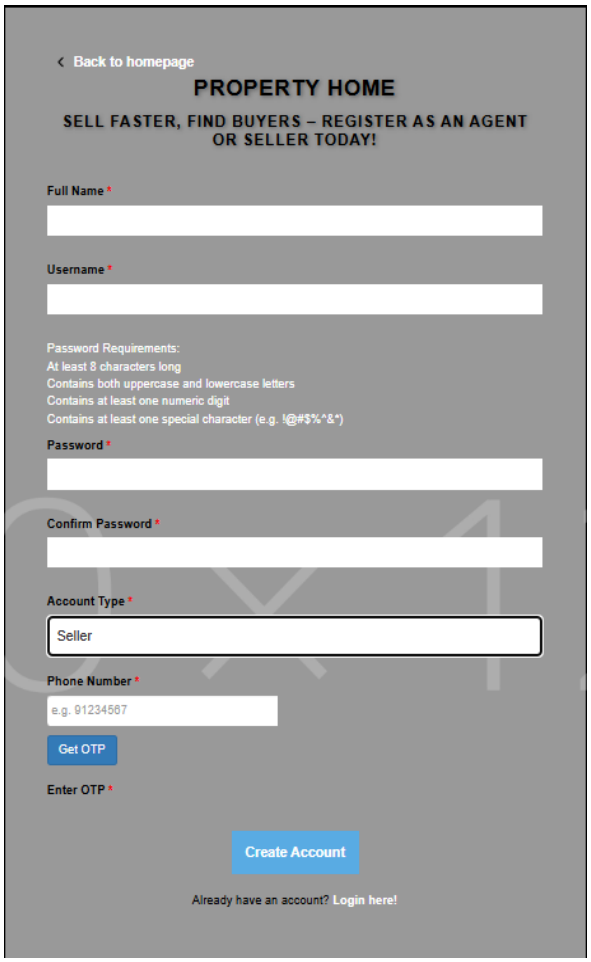


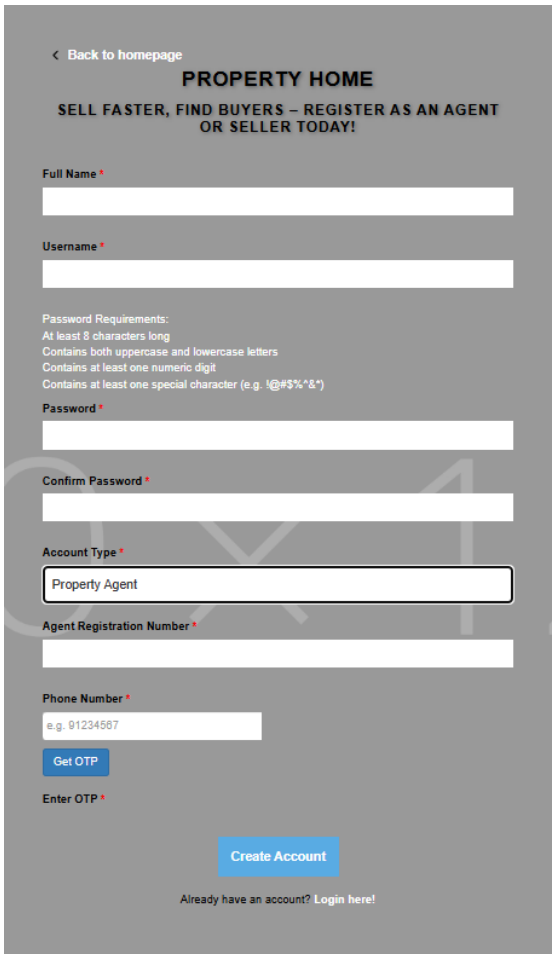
These CTAs are designed to immediately guide users toward action based on their respective goals.

3.1.4 Registration Page

The registration process includes tailored forms for agents and sellers:

- **Sellers** fill in full name, username, password, confirm password, phone number, and OTP verification.
- **Agents** complete the same fields, with an additional required entry for their **Agent Registration Number**, which is verified against the external agent registry through API.





Password criteria are clearly stated to ensure account security. OTP verification is enforced using SMS-based validation for added authentication using Twilio API.

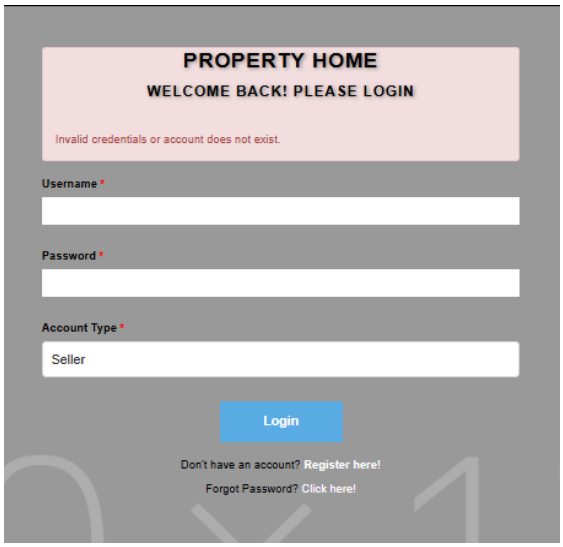
3.1.5 Login Page

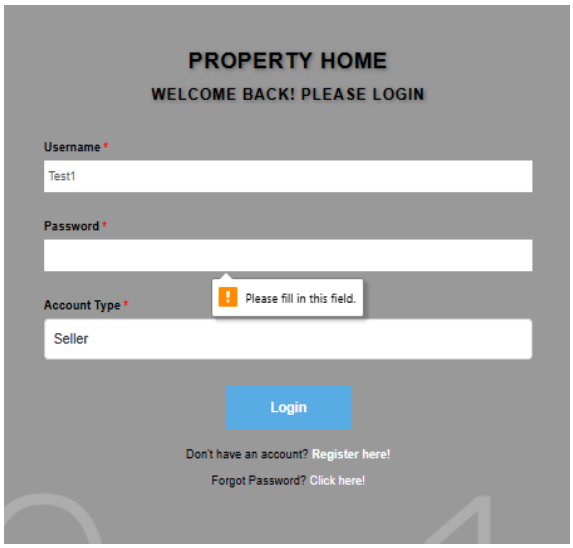
The login page allows both sellers and agents to securely access their accounts. It consists of:

- Username and password input

- Account type selection dropdown
- Links to register for new users and reset password for existing ones

This layout supports quick access and role-specific routing based on successful login credentials. Validation messages guide users if they enter incorrect information, and upon successful login, users are redirected to their respective dashboards.



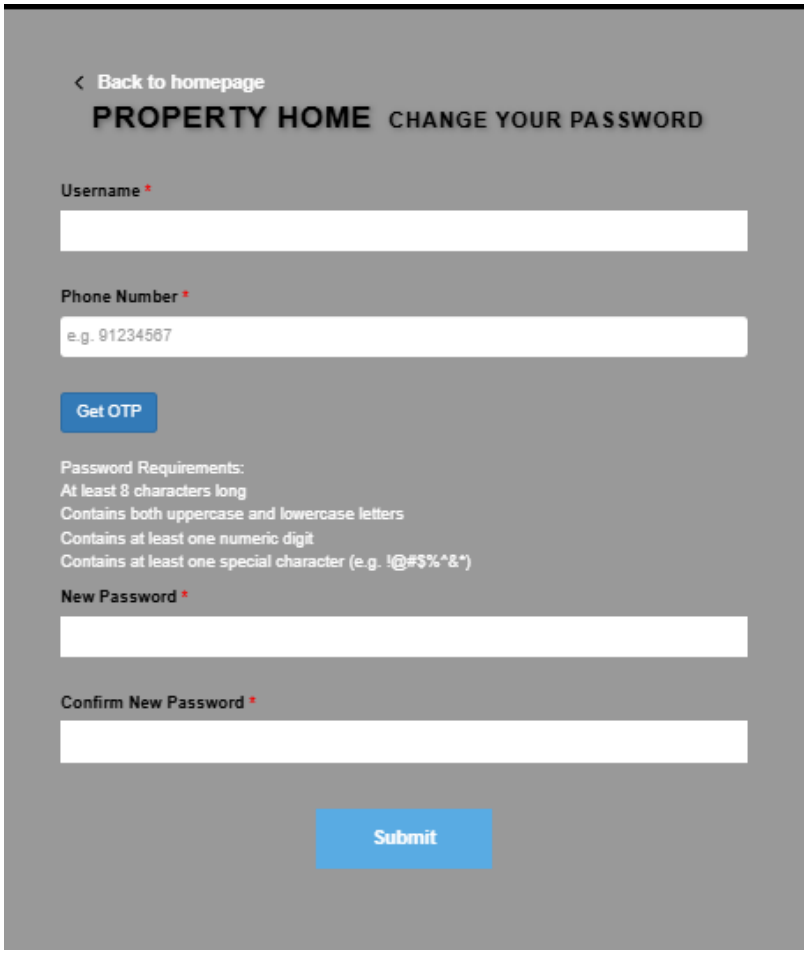


3.1.5.1 Forget Password Page

The **Forget Password Page** allows users to reset their password securely. Key features include:

- Required fields: **Username**, **Phone Number**, and OTP entry
- **New Password** and **Confirm New Password** fields with password rule guidance
- A **Submit** button to complete the reset

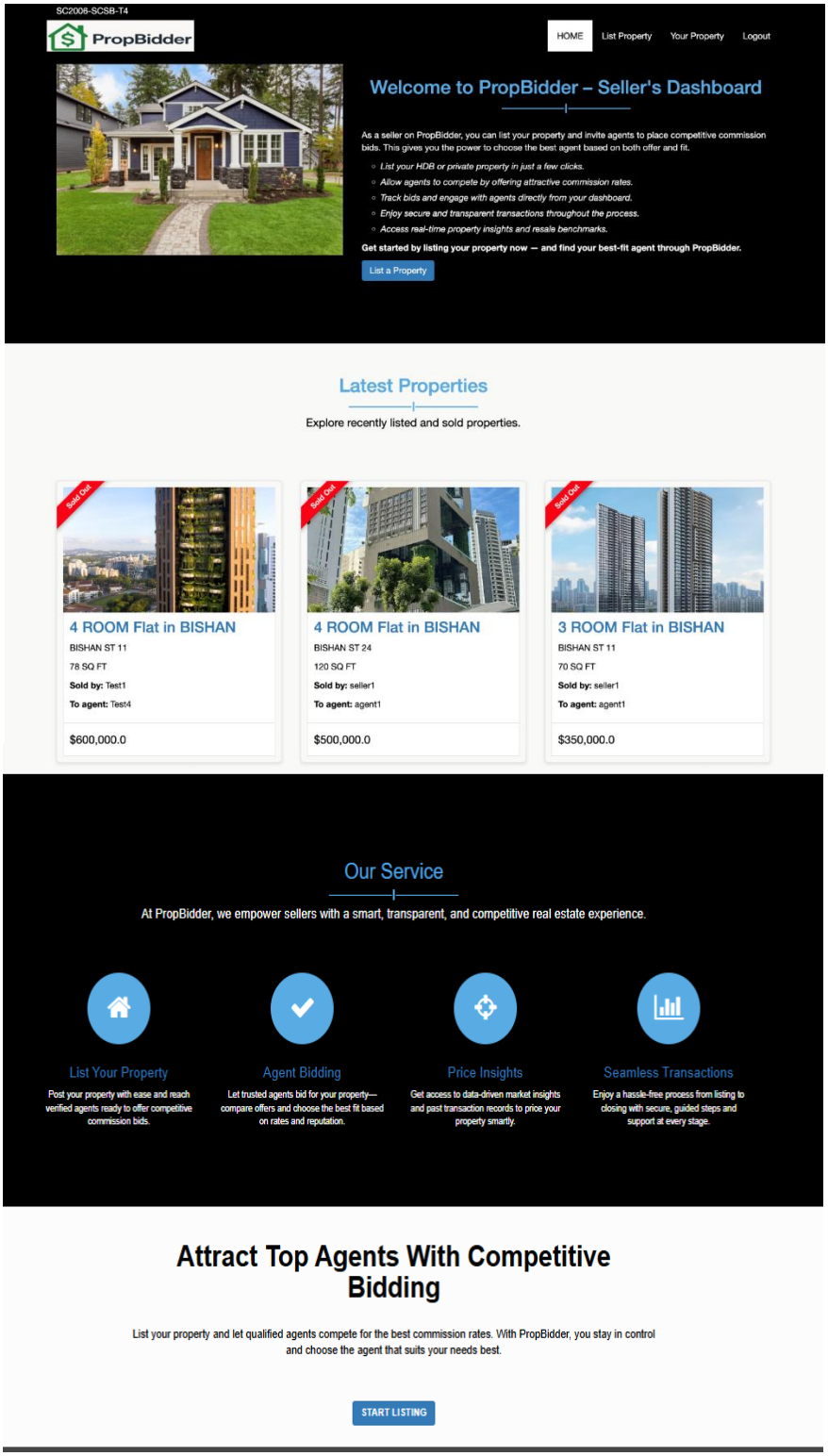
This page ensures secure and user-friendly password recovery, with validation for both OTP and password criteria.



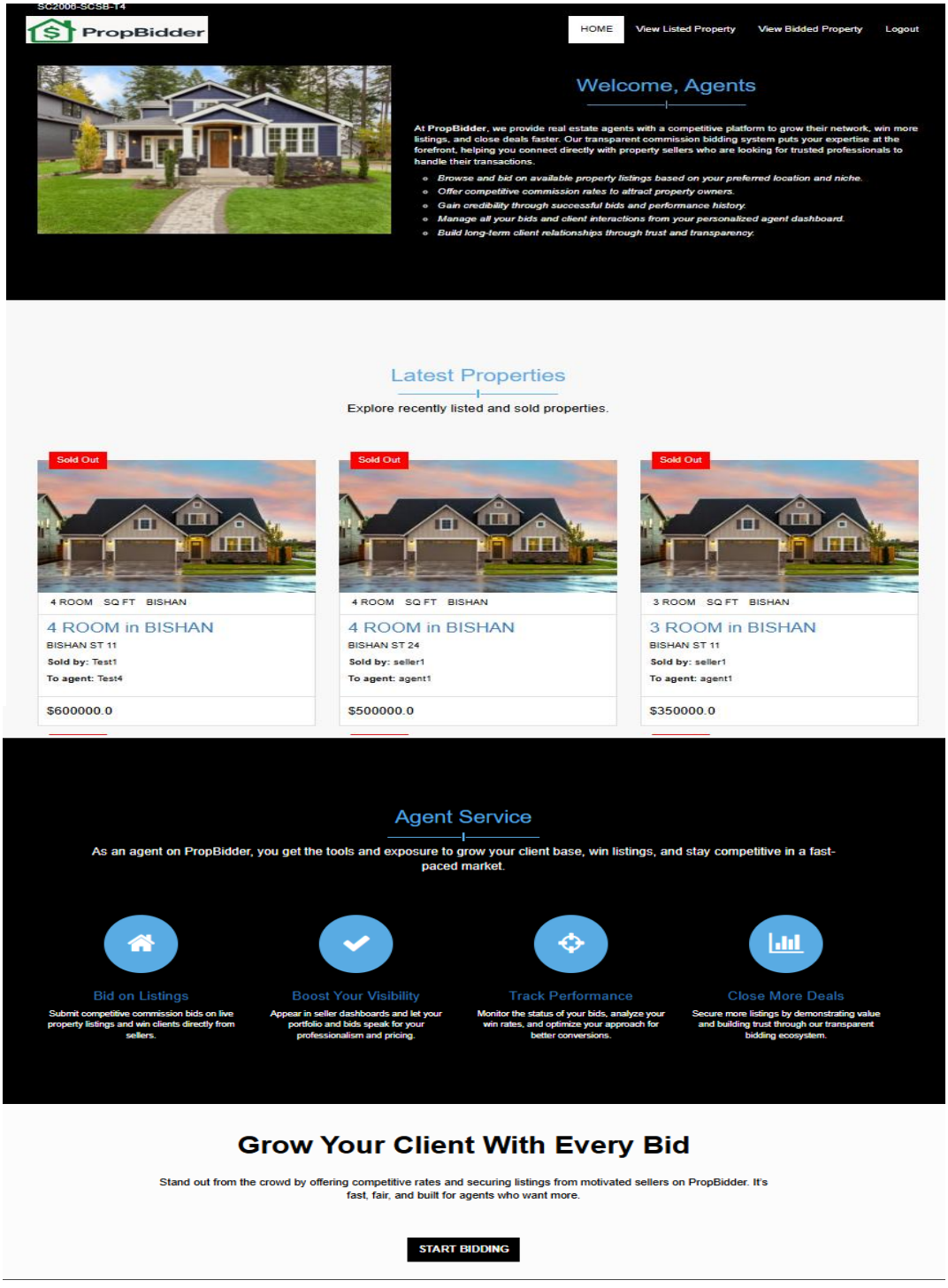
3.1.6 Agent and Seller Dashboard

Once logged in, users are routed to their respective dashboards. Each role has a unique dashboard and welcome section:

- **Seller Dashboard**: Includes personalized greeting and onboarding steps to list properties, receive agent bids, and make data-informed decisions.



- **Agent Dashboard**: Welcomes agents with tools to bid, monitor performance, and increase client conversion through better visibility.

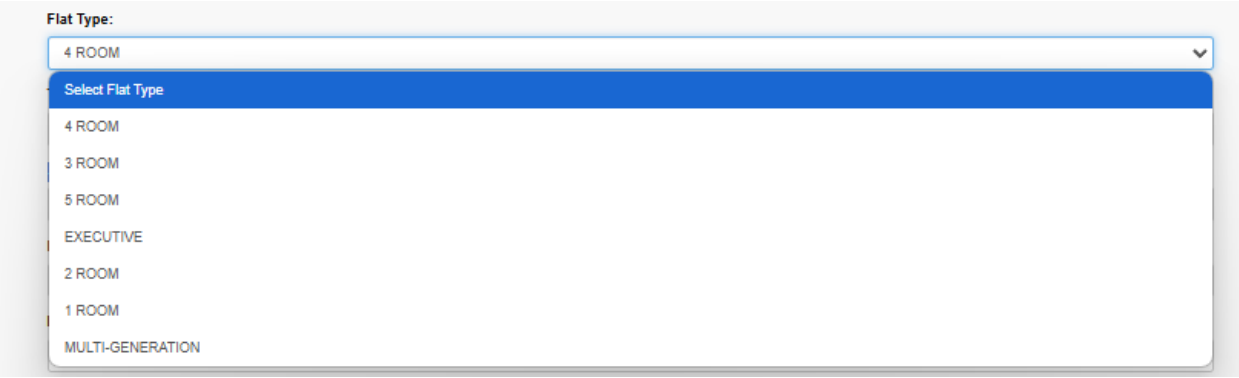


Visual and textual elements are role-aligned and action-oriented.

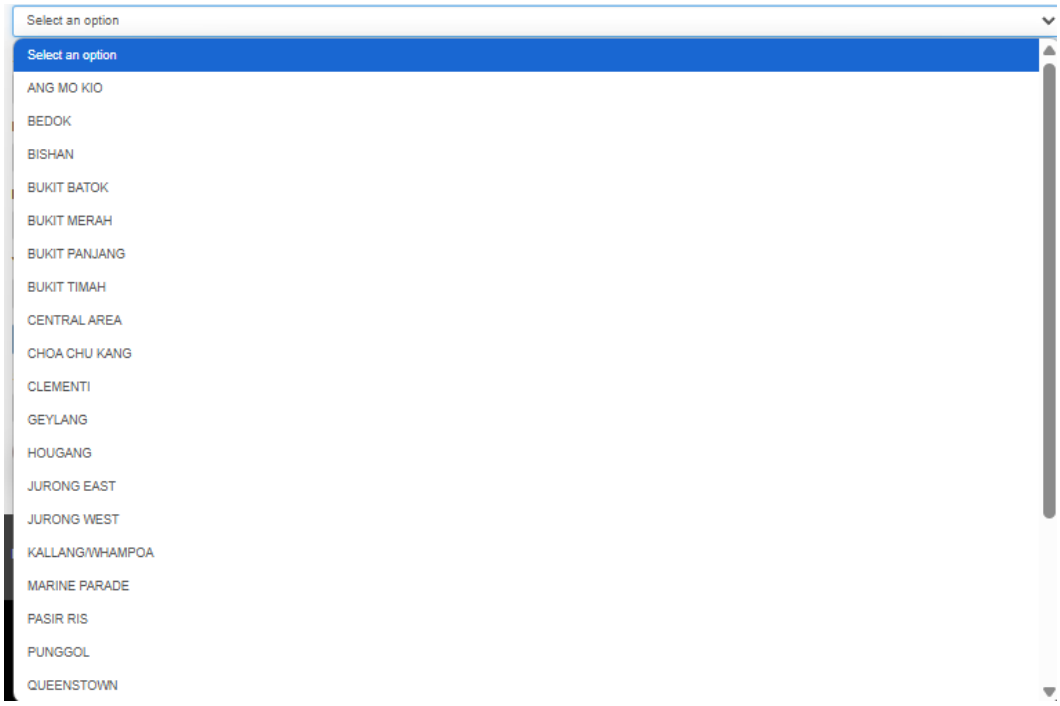
3.1.7 Listing Property page (Seller's Functionality)

Sellers can access the **List Your Property** page through their dashboard navigation. This page provides a structured form that enables users to submit detailed information about their property, including:

- **Flat Type** (dropdown selection)



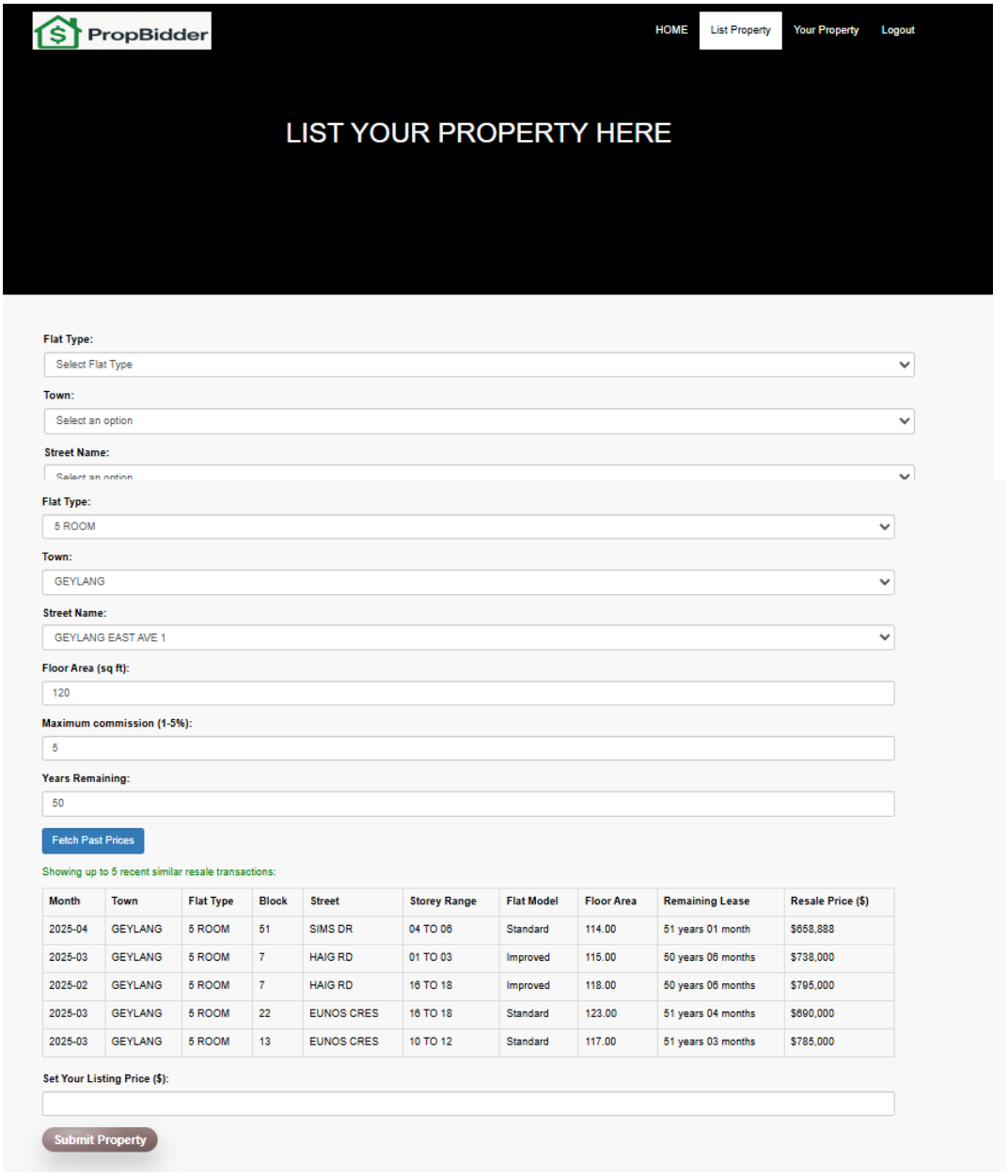
- **Town** and **Street Name** (dropdowns based on HDB dataset) , Town will be auto populated to the corresponding town of the Street Name to prevent mix-up of Town and Street Name.



- **Floor Area (sq ft)**
- **Maximum Commission (1–5%)**
- **Years Remaining on Lease**
- **Fetch Past Prices,** which shows you the closest results (up to five) based on similiarity of your listings

- **Set Listing Price**, which you can enter the listing price of the property after ing Past price.

The form enforces basic validations (e.g., required fields, value limits) and includes a button to **Fetch Past Prices**, enabling sellers to make informed pricing decisions based on recent transactions. Upon submission, the listing becomes visible to agents for bidding.



3.1.8 Your Property Page (Seller Functionality)

Once a seller has submitted a property listing through the **List Your Property** form, they are redirected to the **Your Property** page. This page acts as a centralized dashboard for sellers to monitor and manage their listed properties.

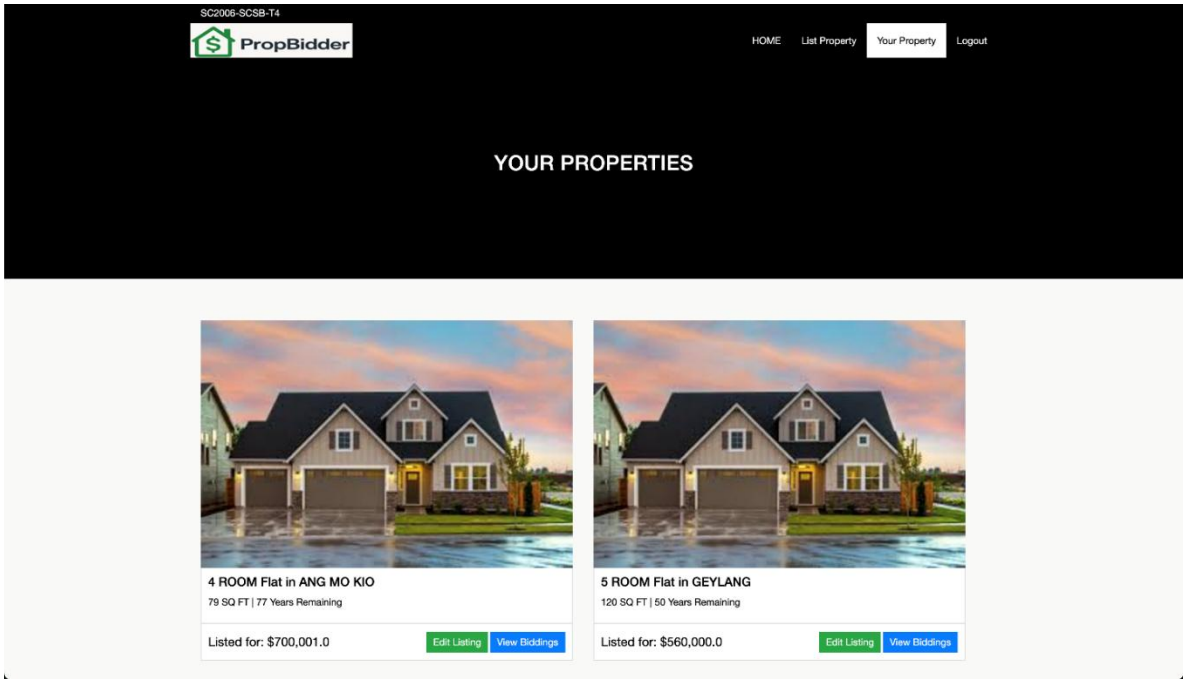
Each property is presented in a card-based layout and includes:

- A **thumbnail image** of the property
- **Flat type** (e.g., 4 ROOM Flat)
- **Location** (e.g., BISHAN, GEYLANG)
- **Floor Area** (in square feet)
- **Years Remaining** on lease
- **Listed Price** (e.g., $700,000)
- A **View Biddings** button to see agents who have placed commission bids
- An **Edit Listing** button to edit the details of their listing
- A **Review Agent** button to leave a review to the agent who won the bidding

This page allows sellers to:

- Edit details of their own listings
- Review details of their active listings
- Track the status of submitted properties
- Compare bidding offers from agents
- Select an agent based on their proposed commission

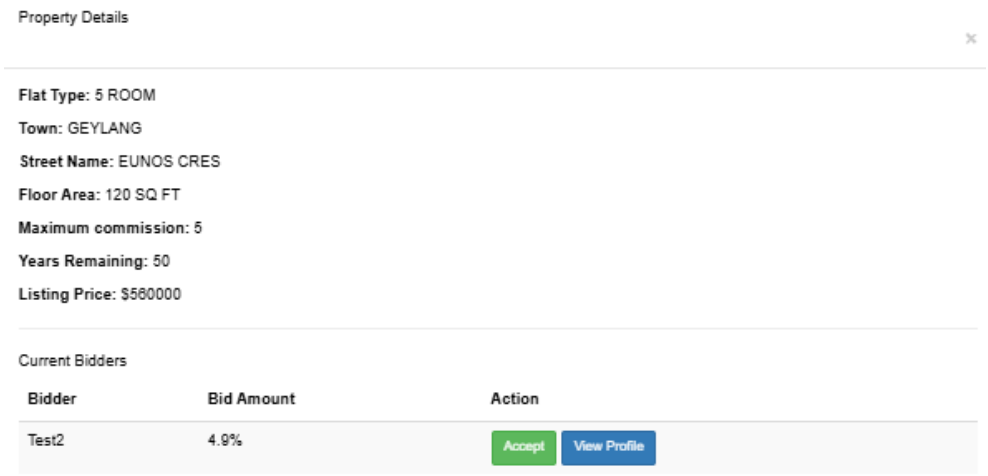
The layout is optimized for readability and ease of navigation, ensuring that sellers can quickly evaluate and act on incoming bids.



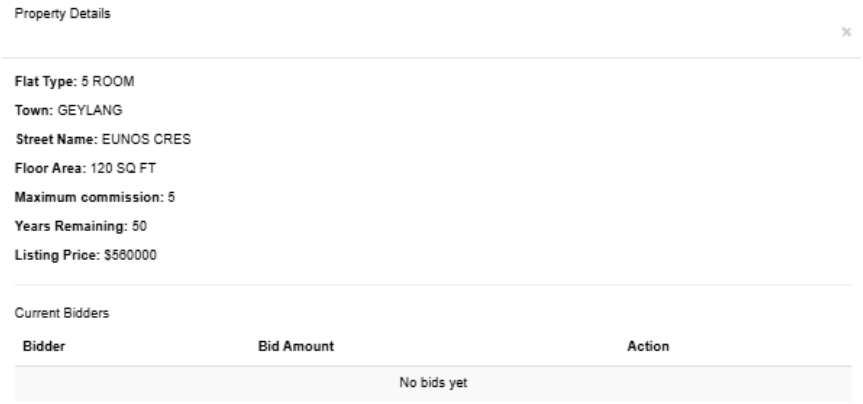
3.1.8.1 View Biddings

The bidding modal displays the following:

- Detailed listing information (flat type, town, street name, area, max commission, years remaining, and listing price)
- A bidder table listing:
    - **Bidder name**
    - **Bid Amount (%)**
    - **Action buttons** to **Accept** the bid or **View Profile** of the agent



- If there are no bids, a message stating "No bids yet" is shown.



This view supports seller decision-making by clearly presenting agent offers in a structured, interactive format.

The layout is optimized for readability and ease of navigation, ensuring that sellers can quickly evaluate and act on incoming bids.

3.1.8.2 View Profile of bidder

When a seller clicks **View Profile** from within the bidding modal, they are redirected to a detailed agent profile page. This page provides insights into the agent's identity and performance.

Displayed agent information includes:

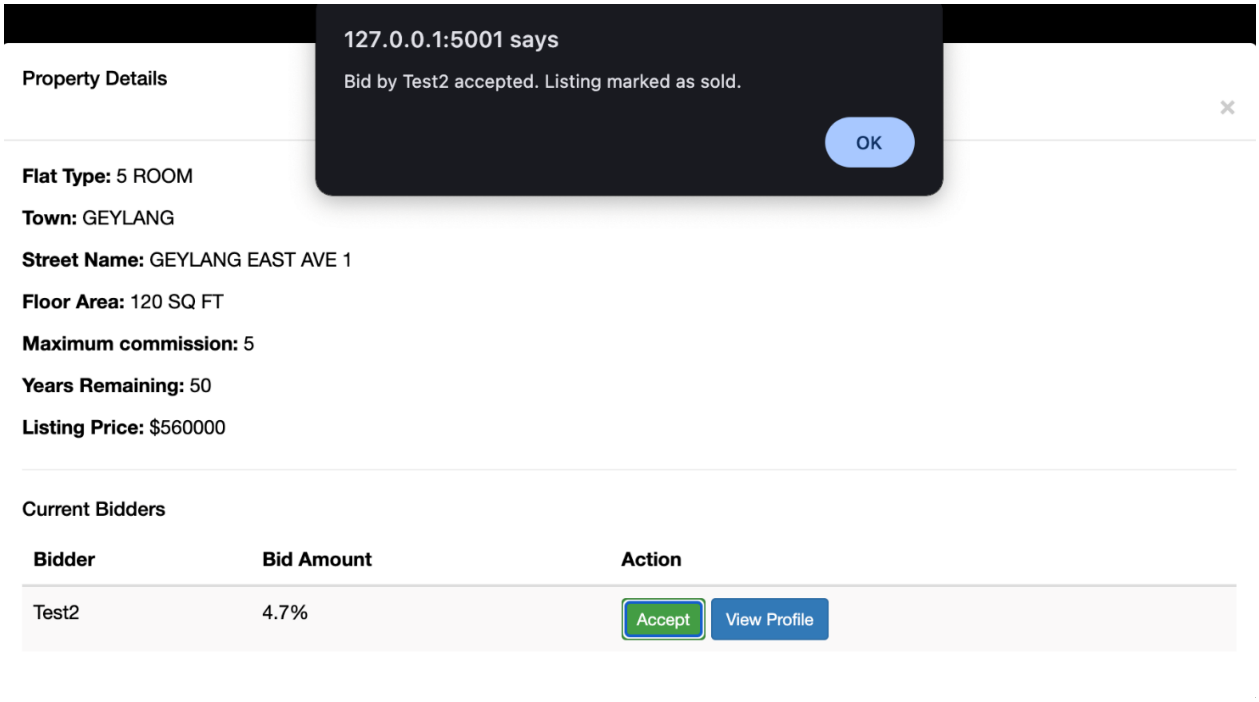
- **Full Name**
- **Username**

- **Agent ID**
- **Review Section**, listing:
    - **Rating** (1 to 5 scale)
    - **Written Review** from past sellers
    - **Seller Identity** associated with each past review

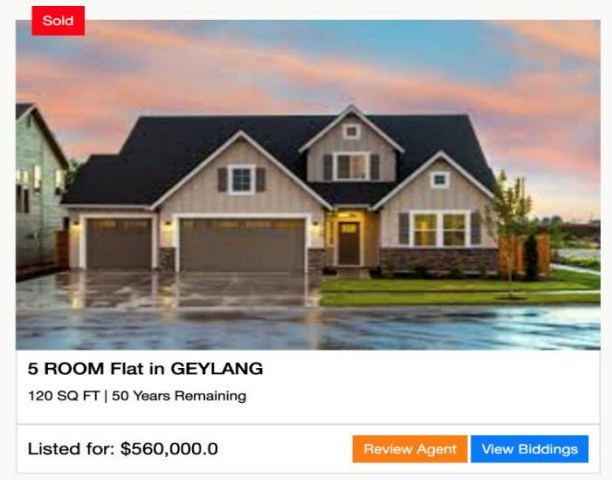
This profile helps sellers assess agent credibility and performance before accepting a bid. The clean and professional layout focuses on clarity and verification, aiding trust-based decision making.

3.1.8.3 Accept Bidding

After the seller clicks the “Accept” button, an alert will show up saying that the bid has been accepted, and the listing is marked as sold. Since we are not able to stimulate the whole process of an agent selling the property (agent contacts seller, goes to seller's house to vet, finds a buyer, sign contract etc.), here we assume that the process is over once seller clicks the “Accept” button.



Once the process is over, seller will be able to see a “Review Agent” button on their listing, and they will not be able to edit their listing again. Their listing will also be marked with a “Sold” tag.

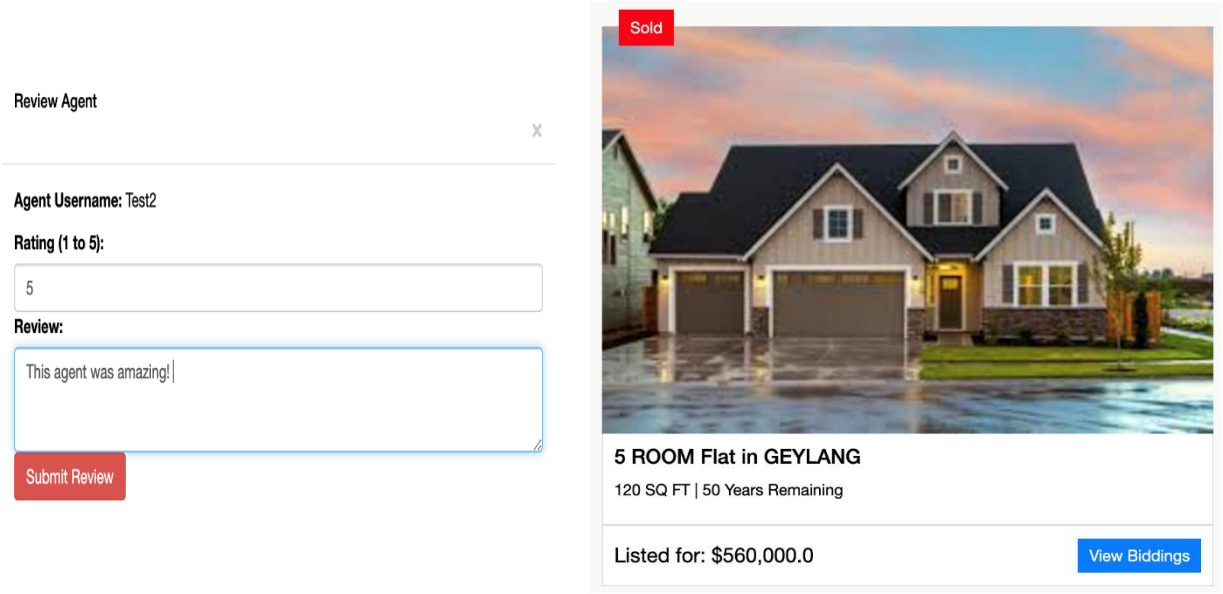


3.1.8.4 Review Agent

When seller clicks on the "Review Agent" button, a modal view will pop out, where it will display:

- **Agent Username**
- **Rating** (Input field)
- **Review** (Text field)
- **Submit Review** button

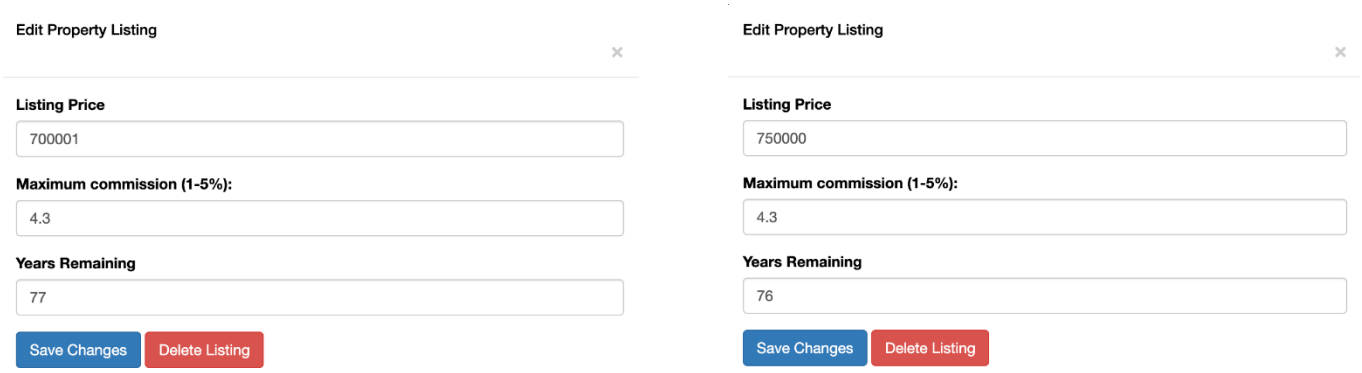
Here, sellers will be able to leave a rating and review for the agent. Once they "Submit Review", an alert will pop out saying that they have completed the review process. Once they have left a review, they will only be able to view their past biddings.



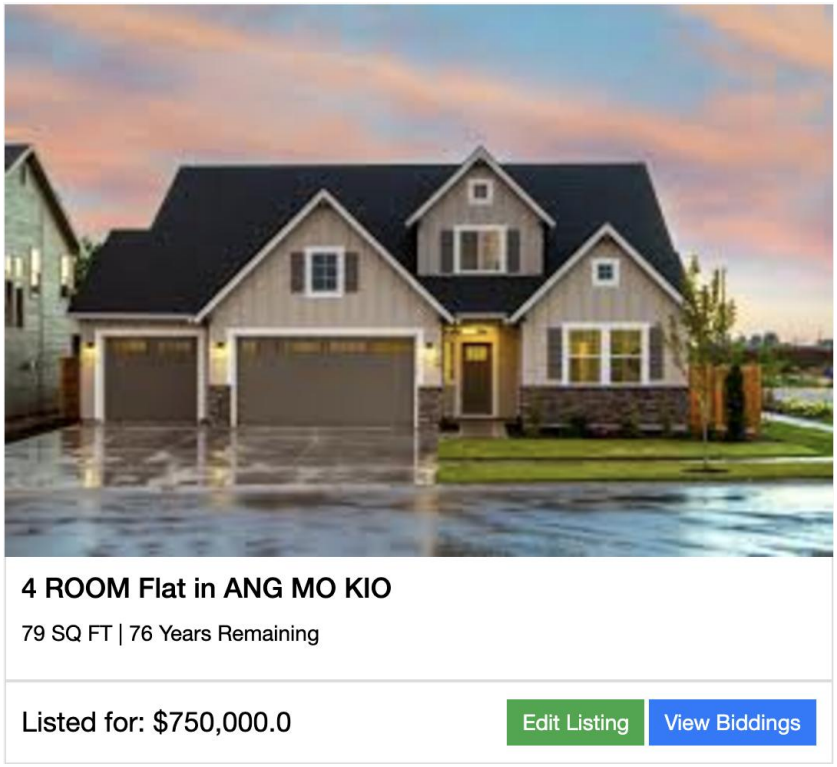
3.1.8.5 Edit Listing

The Edit Listing modal include:

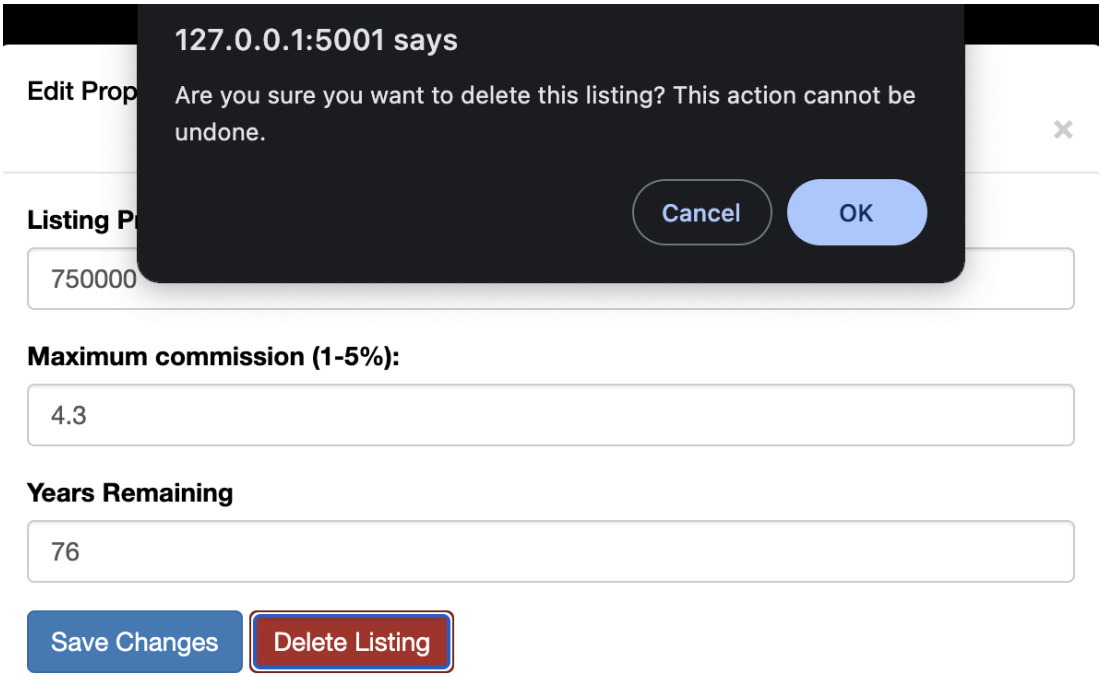
- **Listing Price** (placeholder populated by current value)
- **Maximum commission** (placeholder populated by current value)
- **Years Remaining** (placeholder populated by current value)
- **Save Changes** button to save the changes made by seller
- **Delete Listing** button to delete the entire listing off the site



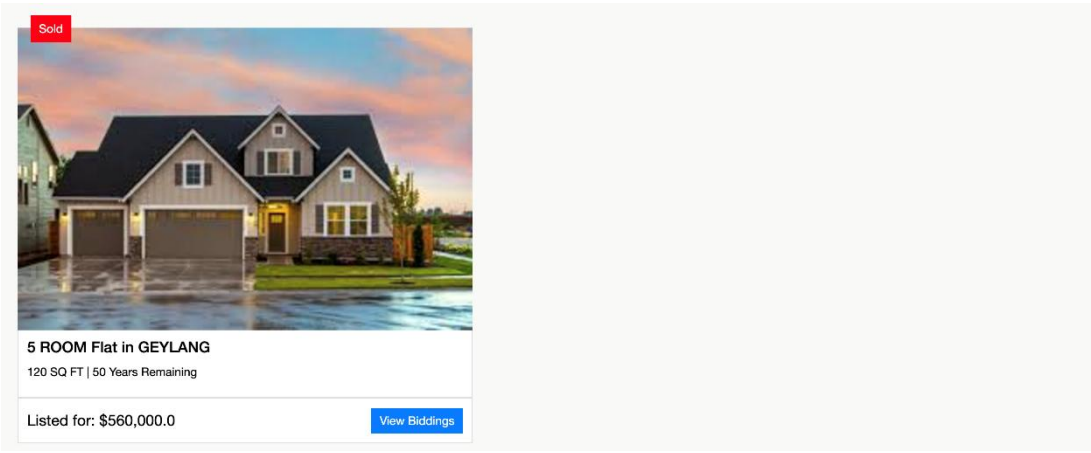
Once the seller has made their input on the changes that they want to make (image on the right) they can then proceed to press “Save Changes” and the changes will be reflected in the listing.



3.1.8.6 Delete Listing



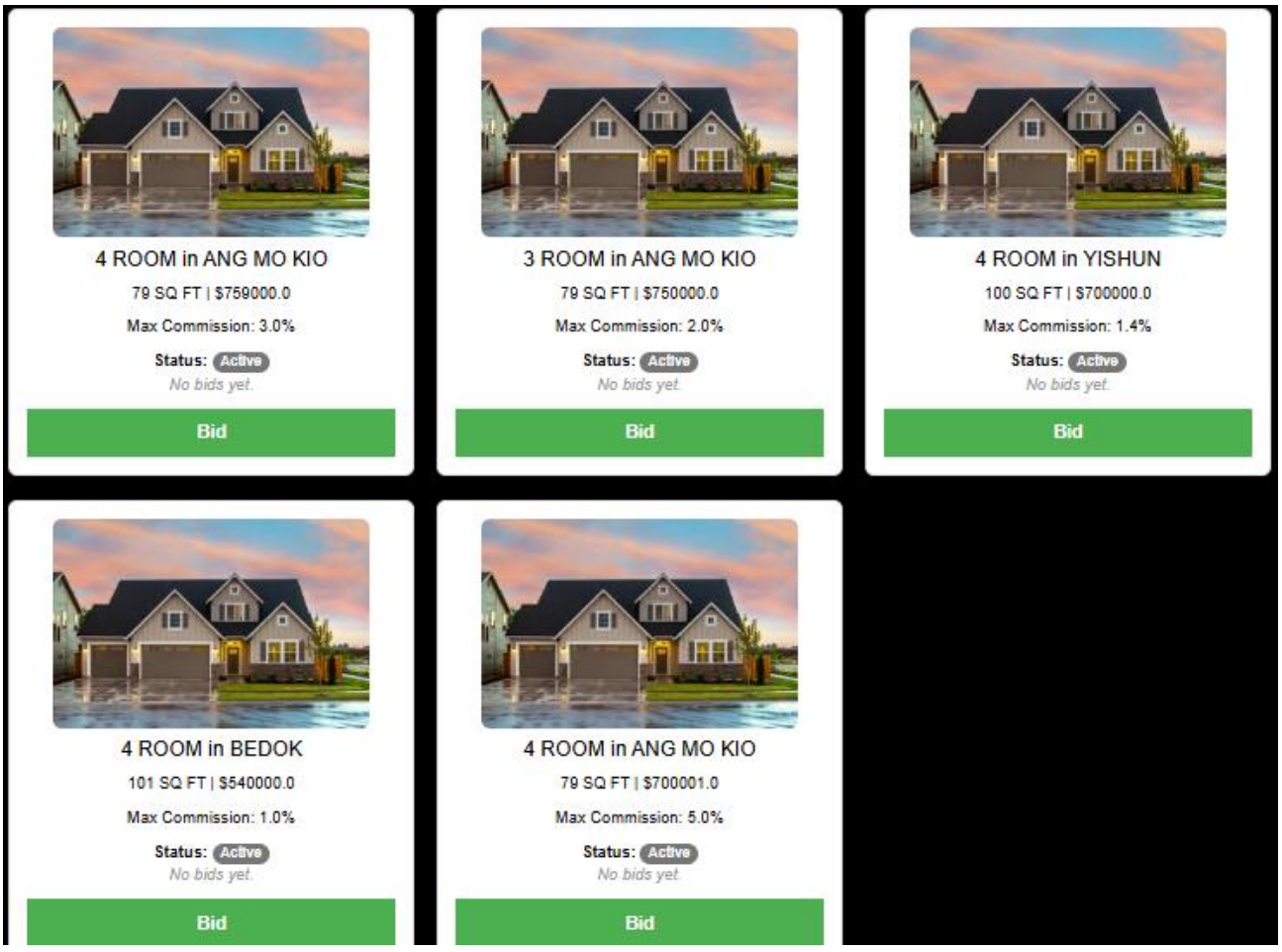
When the seller clicks on “Delete Listing” an alert will show up to prompt the seller again to make sure that the seller did not mis-click. Once they click “OK” to confirm. The listing will be deleted from the view.



3.1.9 View Listed Properties (Agent's Functionality)

The **View Listed Property** page provides agents with a comprehensive overview of all properties currently open for bidding. Each listing is displayed in a consistent card layout and includes:

- A **thumbnail image** of the property
- **Flat type** (e.g., 3 ROOM, 4 ROOM)
- **Floor Area (SQ FT)**
- **Listing Price**
- **Maximum Commission** allowed by the seller
- **Status** (e.g., Active)
- **Bidding status** (e.g., "No bids yet")
- A **Bid** button for agents to submit their commission offer



This page empowers agents to:

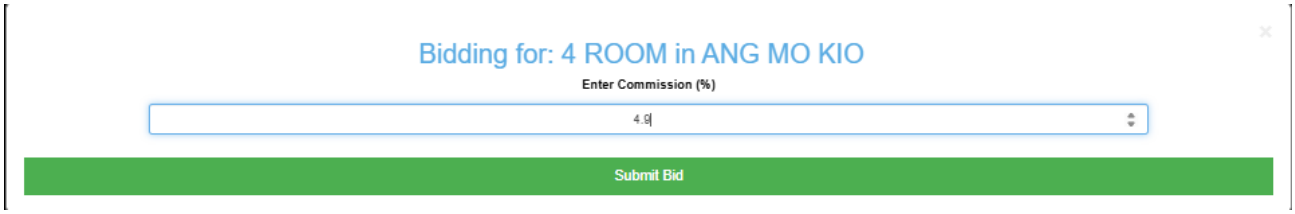
- Evaluate available listings based on their preferred regions and property specifications
- Submit bids directly through the interface
- Monitor active opportunities for client acquisition

The design ensures quick visual scanning and intuitive interactions, allowing agents to promptly act on properties that align with their service goals.

3.1.9.1 Bidding Interface (Agent's Functionality)

When an agent selects a property to bid on, a modal window appears, displaying:

- The **property name** and location (e.g., "Bidding for: 4 ROOM in ANG MO KIO")
- A **form field** to enter the proposed commission bid (as a percentage)
- A **Submit Bid** button to finalize the submission



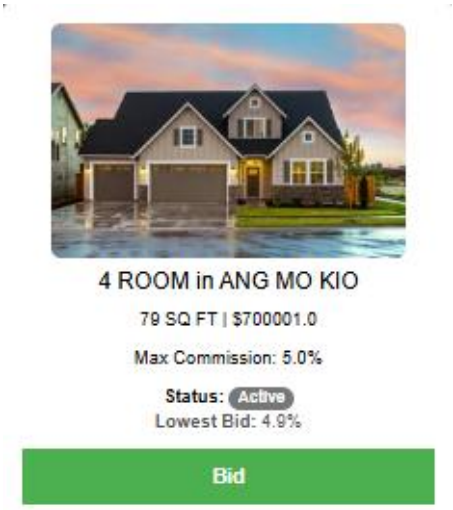
Key functionalities include:

- Input validation ensures commission bids do not exceed the maximum limit set by the seller
- The agent's bid is submitted to the backend and reflected immediately upon confirmation
- If a bid already exists for that agent on the same property, it will be **updated**

Once submitted:

- The agent's bid appears on the **View Bidded Property** page
- The property card updates to show **"Lowest Bid: X%"** if applicable

- The bidding button remains accessible for bid revisions until the property is marked as **sold**



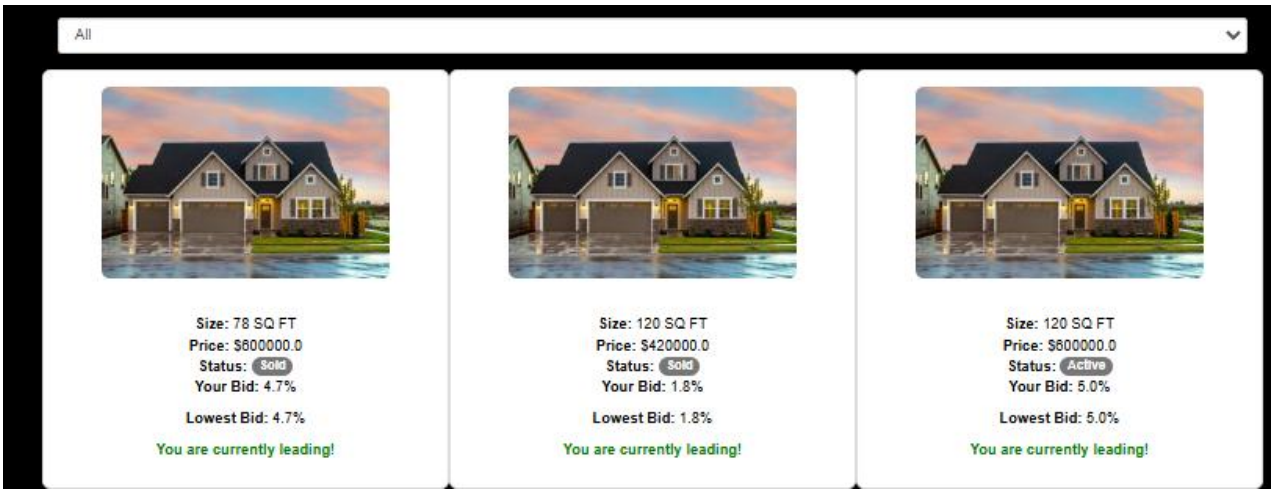
This ensures agents are kept informed of the latest bid ranges and can remain competitive in their offers.

3.1.10 View Bidded Property (Agent Functionality)

The **View Bidded Property** page allows agents to track all properties they have submitted bids for. This view helps agents manage their bidding activity and understand their current standing in ongoing commission competitions.

Each property card contains:

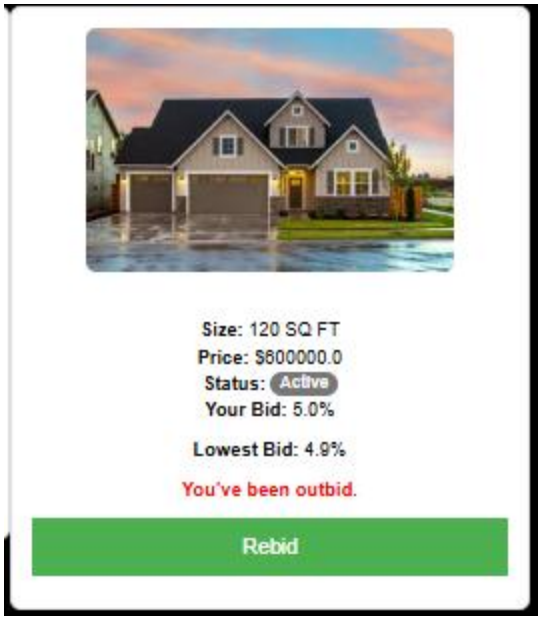
- **Thumbnail image** of the listed property
- **Flat type** and **size** in square feet
- **Listing Price**
- **Status**: Active or Sold
- **Your Bid**: The agent's submitted commission
- **Lowest Bid**: The lowest current bid by all agents
- **Status Indicator**:
    - If the agent's bid is the lowest: *"You are currently leading!"* (highlighted in green)
    - If another agent has underbid: *"You've been outbid."* (highlighted in red)



3.1.10.1 Rebid Functionality:

If the agent is outbid and the listing is still active:

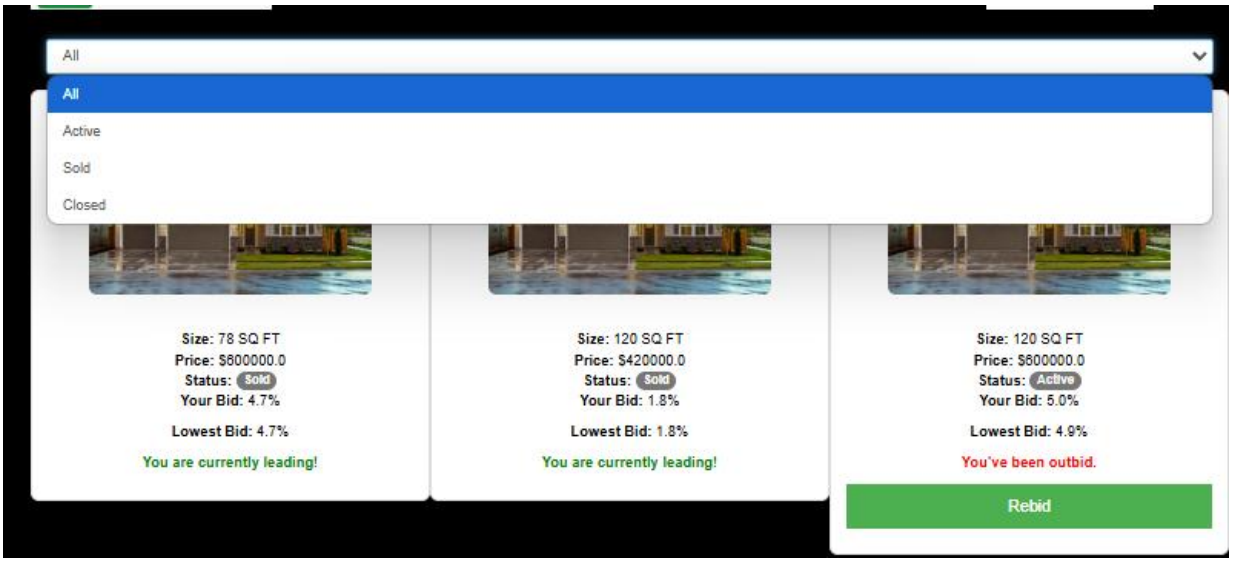
- A **Rebid** button appears beneath the status indicator
- Agents can click it to open a modal, allowing them to submit a revised bid within the seller's allowed maximum commission

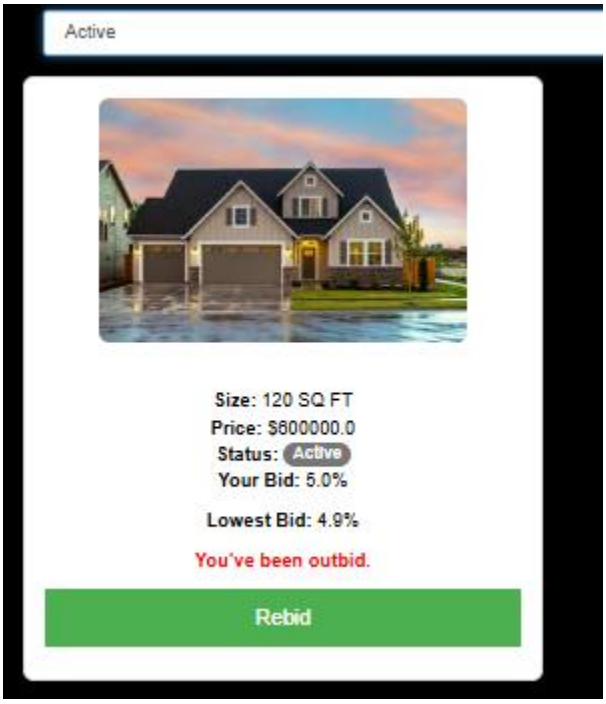


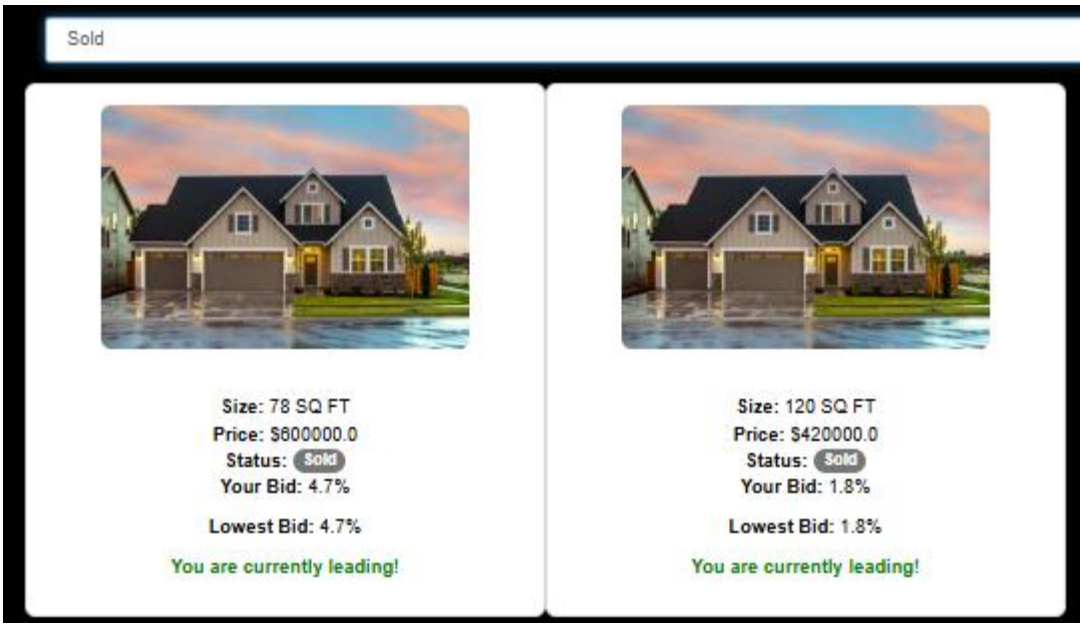
3.1.10.2 Filters:

At the top of the page, agents can filter their bid history by:

- **All**
- **Active** – showing only ongoing listings still open to bids
- **Sold** – showing listings that have concluded







This functionality helps agents:

- Stay competitive by tracking whether their bids are still leading
- Adjust their bids in real-time to maintain an edge
- Track outcomes for past deals and performance insights

3.2 Hardware Interfaces

The system is a web-based application with no specialized hardware requirements.

3.2.1 Client Devices

- Supported Devices:
  Laptops, desktops, tablets, and smartphones with modern browsers (Chrome, Firefox, Safari, Edge).
- Logical Interface:
  Users interact via web browsers using standard input/output peripherals (keyboard, mouse, touchscreen).
- Data and Control Interactions:
  - User inputs (e.g., registration forms, OTP entry) are submitted as HTTP requests.
  - Flask processes the requests and returns HTML-rendered pages or JSON data for API endpoints.

3.2.2 Server Hardware

- Development Environment:
  Application runs on localhost (127.0.0.1:5000) using standard operating systems like Windows, macOS, or Linux.

- Deployment Environment:
  Application is intended to be hosted on a cloud server (e.g., AWS EC2).
  Uses SQLite in development and PostgreSQL in production.
  No dedicated hardware; relies on scalable cloud resources.

3.3 Software Interfaces

3.3.1 Backend

- Framework:
  Flask (Python 3.10+)
- Databases:
  - Development: SQLite (accounts dB, listings.db)
  - Production: PostgreSQL (scalable for multi-user support)
- Libraries Used:
  - sqlite3 – for database operations
  - Twilio – for sending OTP via SMS
  - requests – for validating agent ID with external APIs

3.3.2 Frontend

- Templates:
  HTML with Jinja2 for server-side rendering
- Static Files:
  CSS and JavaScript used for form styling and client-side validation

3.3.3 External Services

- Component: Twilio API
  Purpose: Send OTP via SMS
  Data Flow: Phone number + OTP → Twilio → SMS delivery
- Component: data.gov.sg API
  Purpose: Validate agent registration IDs
  Data Flow: Agent ID → API → Validation result

3.3.4 Data Sharing

- Session Storage:
  Flask's session object used to temporarily store OTPs during verification.
- Database Tables:
  - Users: Stores registration info (username, hashed password, phone number, etc.)
  - Listings: Stores property listing information (unrelated to registration, but part of the same application)

3.4 Communications Interfaces

3.4.1 Protocols

- HTTP/HTTPS:
  All client-server communications occur via HTTP (development) or HTTPS (production).
- RESTful Endpoints:
  - /send_otp (POST): Sends OTP using Twilio API
  - /create_account (POST): Processes user registration

3.4.2 Security

- Encryption:
  HTTPS is used to secure all data in transit.
- Sensitive Data Handling:
  - Passwords are currently stored in plaintext (should be hashed in production).
  - OTPs are session-based and expire after 10 minutes.

3.4.3 Data Formats

- Form Data:
  application/x-www-form-url encoded used for form submissions
- API Responses:
  JSON format, e.g., {"success": true, "message": "OTP sent"}

3.4.4 Third-Party Integration

- Twilio:
  - Authentication: Uses Account SID and Auth Token (configured in config.py)
  - Rate Limiting: Application handles SMS failures gracefully (covered in Test Case #16)
- data.gov.sg API:
  - Endpoint accessed via GET requests to validate agent registration numbers

4. System Features

4.1 Authentication Features

The Authentication features ensures that new users can register, verify their identity through secure validation steps and gain authorized access to the platform

4.1.1 Registration

4.1.1.1 Description and Priority

The Registration feature allows new users to be able to create new accounts as Sellers or Agents. It ensures verification via OTP sent by Twilio API and cross-verification for Agent roles through the ACEAS API.

Priority: High – Critical for ensuring secure account registration and verifying the authenticity of agent's identity.

4.1.1.2 Stimulus/Response Sequence

User Action: User fills in registration form with full name, Username, Password, Confirmation of password, phone number and select a role (Seller/Agent).

System Response:

- The system checks for validity of the form, if user selects Agent, validate Agent ID via ACEAS API before Sending an OTP to the provided phone number.
- In the event of invalid form inputs, error message will prompt the user to reenter the form.

User Action: User enters the OTP that was sent to the phone number and register an account

System Response: System creates and stores the user account under User's database.

4.1.1.3 Functional Requirements

REQ-1: The system shall allow users to register by providing necessary information full name, username, password, confirmation password, phone number, role, AgentID if Agent).

REQ-2: The system shall send OTP Verification via Twilio API to all users during registration.

REQ-3: The system shall verify Agent IDs in real-time with the ACEAS API before creating Agent's accounts.

REQ-4: The system shall prevent account creation until both OTP and Agent ID (for agents) are validated.

4.1.2 Login

4.1.2.1 Description and Priority

The login feature allows existing users in the database to securely access their accounts by validating their roles, username and password credentials.

Priority: High – Secure login is essential to prevent unauthorized access to personal data, sensitive property information, and bidding records. Without a proper login, the security and integrity of the platform would be at risk. Potentially leading to misuse of features and user distrust

4.1.2.2 Stimulus/Response Sequences

User Action: User inputs registered username, password and role on the login page.

System Response:
- System checks for validity of the credentials including the role, and redirect Sellers to the Seller Dashboard and Agents to the Agent Dashboard.
- In the event of invalid credentials, error message will pop up saying invalid credentials.

4.1.2.3 Functional Requirements

REQ-1: The system shall allow users to login with a valid account type, username and password that is stored within the database.

REQ-2: The system shall redirect sellers to the Seller Dashboard page upon successful login

REQ-3: The system shall redirect agents to the Agent Dashboard page upon successful login

REQ-4: The system shall reject invalid login attempts and display appropriate error messages without revealing whether the username or password was incorrect.

4.1.3 Forgot Password

4.1.3.1 Description and Priority

Forgot Password allows users who have forgotten their account credentials to reset their passwords securely through phone number verification using an OTP (One-Time Password) system.

Priority: High - Password recovery is crucial for maintaining accessibility without compromising account security. A secure and simple password reset process ensures users can regain access without creating vulnerabilities in the authentication system.

4.1.3.2 Stimulus/Response Sequences

User Action: User clicks on the "Forgot Password" link on the login page and submits their registered phone number.

System Response:
- Sends an OTP to the user's phone number via Twilio API.
- Verifies OTP entry.

User Action: User enters the OTP and a new password.

System Response:
- If OTP is valid, the system updates the password and notifies the user of a successful reset.
- If OTP is invalid, the system prompts the user to retry.

4.1.3.3 Functional Requirements

REQ-1: The system shall allow users to request password reset through the "Forgot Password" option on the Login Page.

REQ-2: The system shall send a secure OTP to the registered phone number using the Twilio SMS API.

REQ-3: The system shall verify the OTP input by the user.

REQ-4: Upon successful OTP verification, the system shall allow the user to set a new password.

REQ-5: The system shall update the user's password in the database securely.

REQ-6: The system shall display a message saying User password has been successfully changed.

4.2 Seller's Feature

4.2.1 List Property

4.2.1.1 Description and Priority

The List Property feature allows sellers to input detailed information about their property, such as flat type, town, street name, floor area, years remaining, maximum commission rate, and listing price. Sellers can also retrieve up to 5 recent similar resale transactions via API to assist in setting a competitive price.

Priority: High - Sellers need to create property listings for agents to bid on. Without listings, the core platform functionality cannot proceed.

4.2.1.2 Stimulus/Response Sequences

User Action: Seller enters property listing page.

System Response: Dynamically validates and loads town-street via API.

User Action: Seller enters property details such as flat type, town, street name, floor area, years remaining and maximum commission rate.

System Response: Dynamically populate town to ensure that the street name is in the correct town.

User Action: Seller clicks "Fetch Past Prices."

System Response: Displays up to 5 recent resale prices using API data.

User Action: Seller enters listing price after considering from the recent data and submit listing.

System Response: Saves listing and redirects to "Your Properties" page.

4.2.1.3 Functional Requirements

REQ-1: The system shall allow sellers to input flat type, town, street, floor area, lease years, max commission, and listing price.

REQ-2: The system shall allow the form to extract the town and street name from external API.

REQ-3: The system shall allow the dynamic consistency between town and street name from external API.

REQ-4: The system shall allow sellers to fetch up to 5 past resale prices through external API.

REQ-5: The system shall store property listing upon submission and show success message.

REQ-6: The system shall validate the form to ensure that all the forms are properly filled in with the allowed parameters (e.g. Between 1-5% Commission Rate, above $1 Listing Price, above 1 Remaining Lease Years)

4.2.2 Manage Listings and View agent's profile

4.2.2.1 Description and Priority

Sellers can view, edit, delete, and manage their property listings, as well as review incoming bids from agents.

Priority: High - Sellers must be able to manage their active listings to respond flexibly to market activity and offers.

4.2.2.2 Stimulus/Response Sequences

User Action: Seller views "Your Properties" page.

System Response: Displays listed properties with action buttons to edit listing.

User Action: Seller edits or deletes a listing.

System Response: Updates or removes the listing in the database.

User Action: Seller views agent bids and agent profiles.

System Response: Displays all bids sorted by lowest commission first and opens up agent's profile displaying past reviews/transactions of agent.

4.2.2.3 Functional Requirements

REQ-1: The system shall allow sellers to view all their listed properties.

REQ-2: The system shall allow sellers to edit or delete properties (only if no bid is accepted yet).

REQ-3: The system shall allow sellers to view all bids on a property.

REQ-4: The system shall display agent profile details linked to each bid.

4.2.3 Accept Bids and Review Agents

4.2.3.1 Description and Priority

Once a seller accepts a bid, the property is marked as sold. The seller can then leave a rating and review for the selected agent.

Priority: High - Finalizing deals and ensuring post-sale feedback maintains marketplace trust and quality control.

4.2.3.2 Stimulus/Response Sequences

User Action: Seller clicks "Accept" on an agent's bid.

System Response: Marks the property as sold; disables the accept button and a review agent button will appear.

User Action: Seller clicks on the review agent button and submits rating and review after transaction.

System Response: Links review to the agent's profile.

4.2.3.3 Functional Requirements

REQ-1: The system shall mark the property as sold after accepting a bid.

REQ-2: The system shall show the sold properties onto the portion of Sold Properties in the Home page, Seller's Dashboard and Agent's Dashboard

REQ-3: The system shall allow sellers to submit a 1 to 5-star rating and written review.

REQ-4: Associate seller reviews with verified transactions only.

REQ-5: The system shall save the reviews onto the agent's profile so that another seller will be able to see.

4.3 Agent's Feature

4.3.1 View Available Listed Properties

4.3.1.1 Description and Priority

Agents can browse all active properties listed by sellers, view property details, and identify opportunities to bid.

Priority: High - Agents must be able to access property data to participate competitively.

4.3.1.2 Stimulus/Response Sequences

User Action: Agent navigates to "View Listed Properties."

System Response: Displays all active properties with max commission, floor area, flat type, etc.

4.3.1.3 Functional Requirements

REQ-1: Allow agents to browse all active property listings.

REQ-2: Display detailed property information: listing price, flat type, max commission, floor area, etc.

4.3.2 Submit and Manage Bids

4.3.2.1 Description and Priority

Agents can submit commission bids on properties below the maximum allowed rate. If outbid, agents can remain competitive.

Priority: High - Enables dynamic bidding and competition, essential for the seller selection process.

4.3.2.2 Stimulus/Response Sequences

User Action: Agent enters bid amount.

System Response: Validates that bid is under the seller’s max commission.

User Action: Agent is notified if they are leading or outbid.

System Response: Updates property listing with lowest bid.

4.3.2.3 Functional Requirements

REQ-1: Allow agents to submit commission bids lower than seller's max commission.

REQ-2: Prevent submission of bids exceeding the maximum commission.

REQ-3: Notify agent if outbid, allowing rebid functionality.

REQ-4: The system shall allow agents to submit bids in 2 decimal places.

4.3.3 View and Filter Bidded Properties

4.3.3.1 Description and Priority

Agents can monitor all the properties they have bid on, and filter between active and sold properties.

Priority: Medium - Helps agents manage their bids but is not critical for initial platform operation.

4.3.3.2 Stimulus/Response Sequences

User Action: Agent navigates to "View Bidded Properties."

System Response: Displays all properties with agent's bid status (Active/Sold).

User Action: Agent uses dropdown to filter properties.

System Response: Dynamically updates property list based on selected filter.

4.3.3.3 Functional Requirements

REQ-1: Allow agents to view all properties they have bidded on.

REQ-2: Allow agents to filter properties based on bidding status (Active/Sold).

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system should retrieve and display property listings and agent profiles within 3 seconds to ensure an optimal user experience.
- The commission bidding leaderboard should update in real-time, with no more than a 2-second delay between bid submission and display.
- The system should process and store agent commission bids within 2 seconds of submission.
- Search and filtering operations for agents and properties should not exceed 4 seconds, even under peak load.
- Notifications to agents about being shortlisted should be delivered within 2 seconds to maintain competitive bidding dynamics.

5.2 Safety Requirements

- The system shall provide disclaimers informing users that commission offers are non-binding until a formal agreement is signed, preventing misunderstandings or disputes.
- The system shall alert users about potential risks of choosing agents solely based on the lowest commission rate, encouraging consideration of experience and reviews.
- Automated backups shall be performed daily to prevent data loss, with backups stored securely and retained for at least 30 days.
- The system shall implement data validation checks to prevent submission of invalid bids (e.g., negative commission rates or excessively low values).
- In case of system failure or downtime, the system shall ensure no data corruption or loss of ongoing bids or transactions.
- The system shall comply with relevant data protection regulations (e.g., PDPA in Singapore) to safeguard user information and prevent unauthorized access.

5.3 Security Requirements

- For account creation, the system shall require user authentication through secure methods (e.g., email verification and two-factor authentication) for all users, including sellers, property agents, and buyers.
- For property agents, their account creation will involve an additional step where their credentials will be checked against the list of agents provided from ACEAS

- All user data, including login credentials and preferences, must be encrypted using industry-standard encryption methods.
- The system must follow Singapore's Personal Data Protection Act (PDPA) guidelines to ensure proper handling of user data.

5.4 Software Quality Attributes

- Usability: The system shall provide an intuitive and user-friendly interface, allowing new users to navigate and perform actions (e.g., placing a bid, shortlisting agents) within 3 clicks or taps.
- Availability: The system shall maintain an uptime of at least 99.9%, ensuring continuous access for users except during scheduled maintenance.
- Reliability: The system shall ensure accurate bid processing and data consistency, even in case of system failures or interruptions.
- Maintainability: The system should be modular to facilitate easy updates and enhancements.
- Interoperability: The system shall integrate seamlessly with third-party APIs (e.g., for property data) while maintaining data consistency.
- Scalability: The system should be designed to handle a growing number of users and an expanding database of resale flat transactions without significant degradation in performance.
- Testability: The system shall support automated testing to verify core functionalities before deployment.
- Portability: The system shall be compatible with all major web browsers (Chrome, Firefox, Safari, and Edge)

5.5 Business Rules

- Users must create an account and be authenticated before submitting reviews or ratings on flats.
- Only verified users can submit more than one review per flat to prevent spam or manipulation.
- Flat recommendations must be based on publicly available government data from data.gov.sg and should be updated biannually.
- The system should not allow recommendations to be influenced by paid promotions to maintain fairness and transparency.
- Users can compare up to 5 different flats at once in the price comparison feature.

- The system should adhere to all data privacy and security regulations set by Singaporean authorities.

6. Other Requirements

6.1 Legal and Regulatory Requirements

The PropBidder platform complies with Singapore's regulatory frameworks relevant to real estate and data privacy. This includes integration with the ACEAS API to validate agents under the Estate Agents Act, and adherence to the Personal Data Protection Act (PDPA) through the secure handling of user information.

6.2 API Integration Constraints

The system utilizes third-party APIs, including the ACEAS API and the Twilio SMS API, to enable agent verification and OTP-based user authentication. Due to Twilio's free-tier limitations, OTP delivery is currently restricted to a single phone number.

6.3 Data Accuracy Assumptions

The system assumes that the datasets retrieved from government APIs are complete and accurate. The validity of suggestions, listings, and bid recommendations is contingent on the reliability of these data sources.

6.4 Accessibility Considerations

The PropBidder platform is designed to be accessible and usable by a wide range of users. While full WCAG compliance has not been implemented, the user interface adheres to several best practices for accessibility:

- Clear labeling and descriptions in all input fields.
- Sufficient color contrast for readability across light and dark elements.
- Simple tab navigation to accommodate keyboard users. Future enhancements may include screen reader compatibility and compliance with WCAG 2.1 Level AA guidelines to ensure inclusivity for users with visual or motor impairments. The current architecture allows for easy extensibility to incorporate these accessibility upgrades without major structural changes.1 Level AA guidelines to ensure inclusivity for users with visual or motor impairments.

6.5 Internationalization Requirements

While the current version of the PropBidder system is limited to English and localized for use in Singapore, its architecture supports future internationalization. All UI text elements are modular and can be externalized for language translation. The system has been designed with the potential to support:

- Multi-language interfaces through internationalization (i18n) standards.
- Location specific formatting for dates, currencies, and addresses.
- Future expansion to agent verification APIs and property data sources in other countries, subject to regional legal compliance.

Appendix A: Glossary

SR NO	TERM	DEFINITION
1	Resale Flat	A previously owned Housing and Development Board (HDB) flat that is being sold by its current owner rather than directly from HDB.
2	Homebuyers	Individuals or couples actively searching for resale flats.
3	User Account Management	A system feature that allows users to manage their account credentials, housing preferences (budget, flat size, amenities, etc.), and saved flats.
4	Remaining Lease	The number of years left on the flat's lease before it expires. The system accounts for this while recommending flats as it affects affordability and resale value.
5	Commute Tolerance	The maximum travel time or distance the user is willing to commute to and from their workplace. The system considers this while recommending flats.
6	Amenities	Facilities and services available near the resale flat such as: schools, polyclinics, gyms, supermarkets.
7	Resale Price Trends	The fluctuation of property prices over time which enables users to estimate whether a flat pricing is justifiable.
8	Agent	A real estate professional who submits commission bids to represent sellers in property transactions.

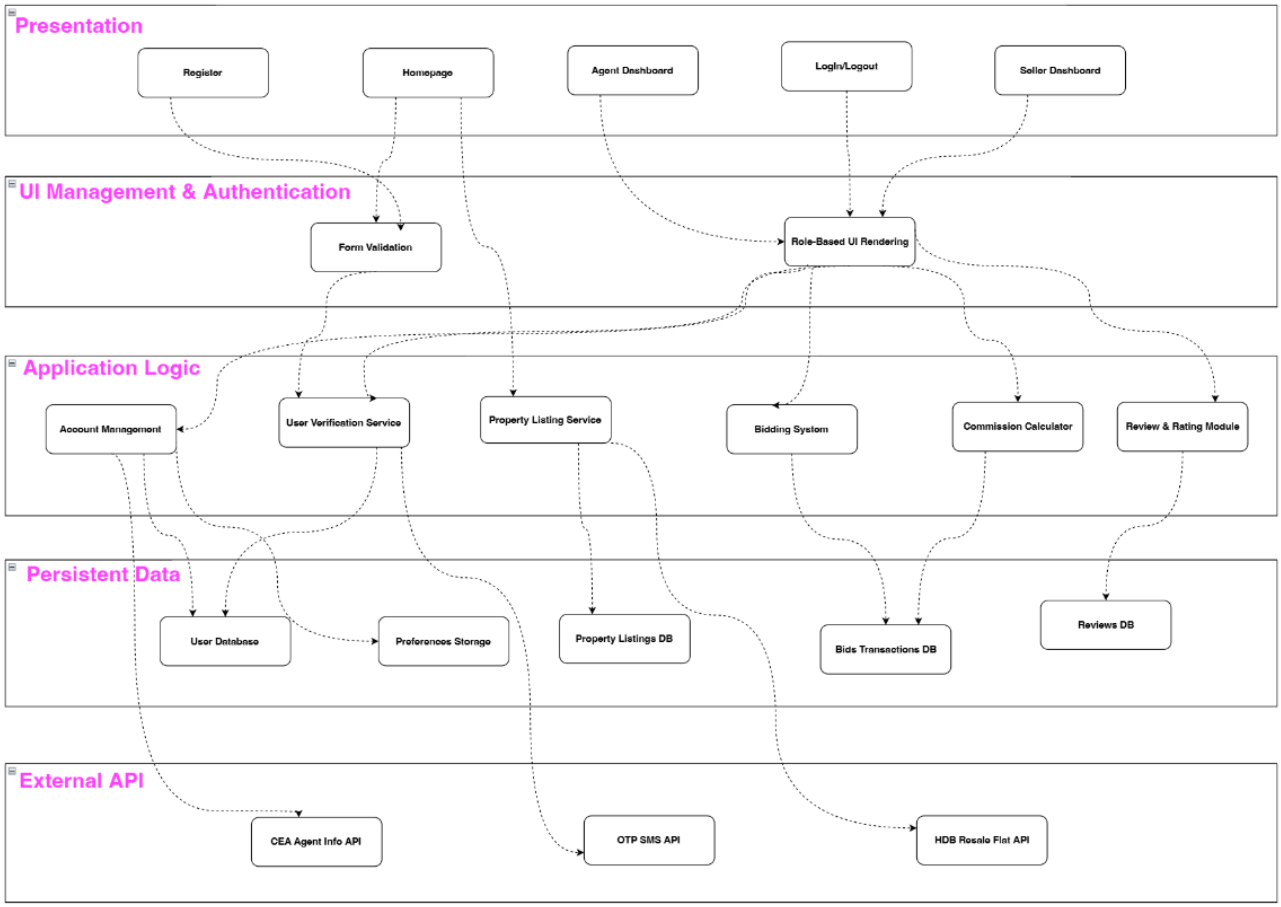
9	Bid Processing	A system that ensures quick processing and updating of property bids in real-time.
10	Verified Transactions	A method to provide credibility for a completed transaction by a user upon which the user will be able to submit a review.
11	User Authentication	A process that ensures the accurate identity of a user and doesn't allow for impersonation. It is implemented by using login credentials and security verification methods.
12	Seller	A user who lists their property and accepts commission bids from agents.
13	Bid	A commission rate offered by an agent in response to a property listing.
14	Commission	A percentage of the property's sale price that an agent requests for their service.
15	Dashboard	A role-based interface provided to agents and sellers with access to core system functionalities.
16	ACEAS API	A government API used to validate agent registration numbers during account creation.
17	Twilio API	A third-party service used to send OTPs (One-Time Passwords) via SMS for secure registration and password reset.
18	OTP (One-Time Password)	A 6-digit code sent to a user's phone for verification during registration or password recovery.
19	Listing	A posted property entry created by a seller including details such as location, flat type, and listing price.
20	Review and Rating	Feedback submitted by a seller after a property is marked sold, including a star rating (1–5) and written comment about the agent.
21	MongoDB	A NoSQL database used to store unstructured data like trends and preferences.
22	PostgreSQL / SQLite	Relational databases used for storing structured data such as user accounts and property listings.
23	Flask	A lightweight Python web framework used for backend development.

24	Frontend	The client-facing portion of the web application, built with HTML, CSS, and JavaScript.
25	Backend	The server-side component responsible for handling logic, APIs, database access, and security.
26	Session Storage	Temporary storage (e.g., for OTPs) used during registration or password reset flows.
27	PDPA (Personal Data Protection Act)	A Singaporean data protection regulation that governs how personal information must be collected, used, and secured.
28	Role-Based UI	A user interface that adapts to the logged-in user's role (agent or seller), showing only relevant functionalities.
29	Agent Profile	A dedicated page displaying an agent's name, reviews, rating, and past transaction history.
30	Active/Sold Filter	A toggle used by agents to filter their bidded properties based on their current status (still open or completed).

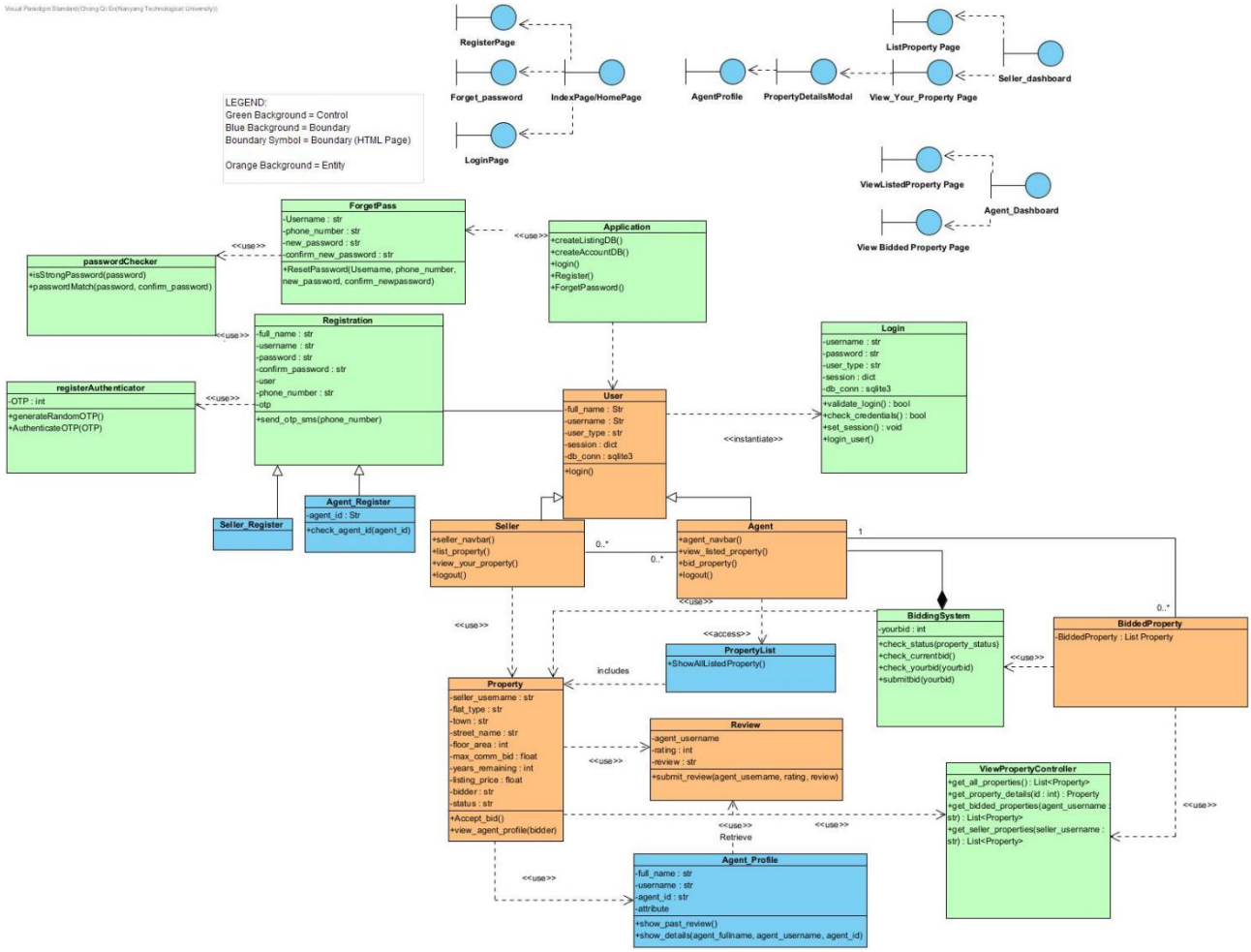
Appendix B: Analysis Models

Please Refer to Lab 3 Deliverable Folder if the image is unclear.

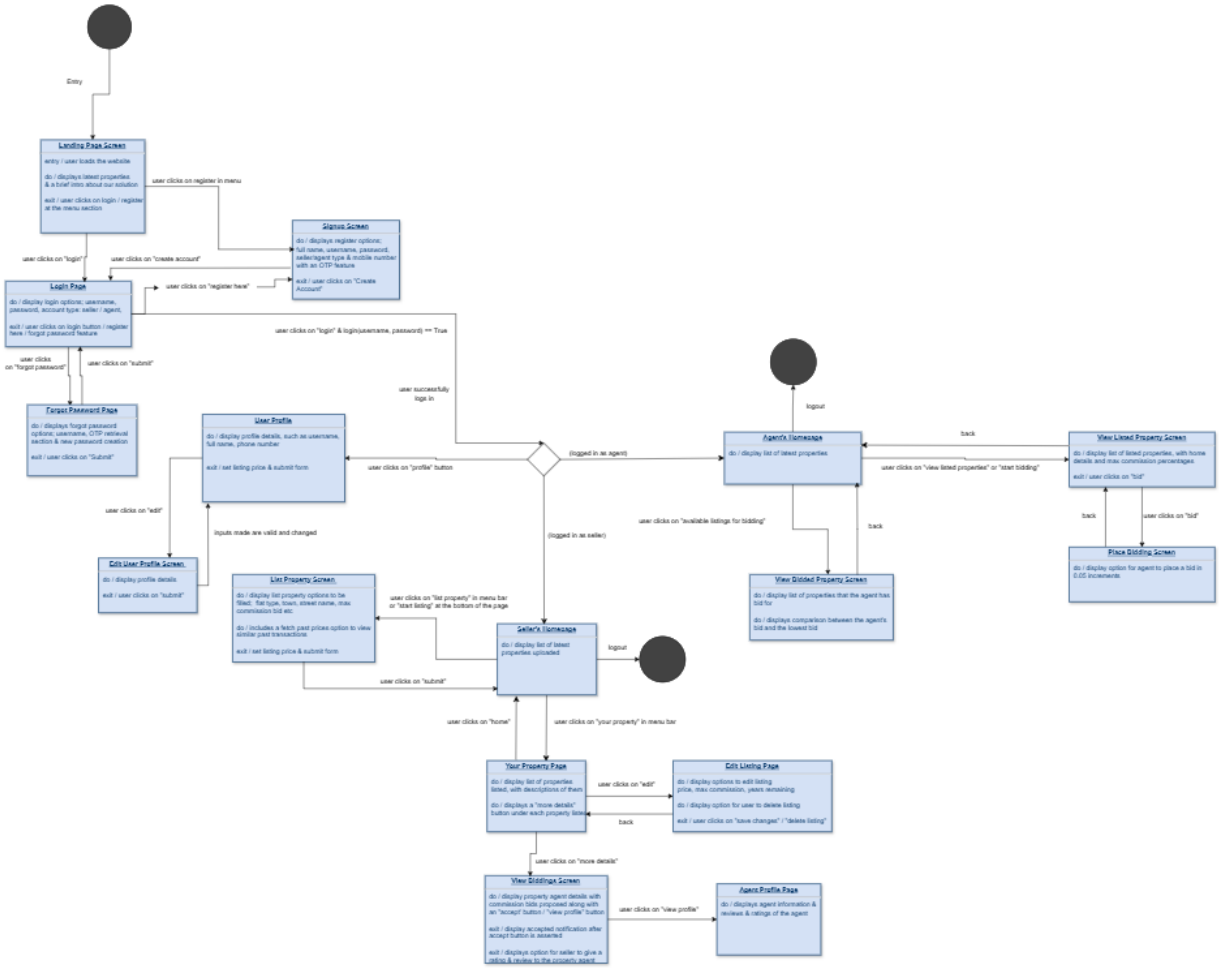
System Architecture Diagram:



Class Diagram:

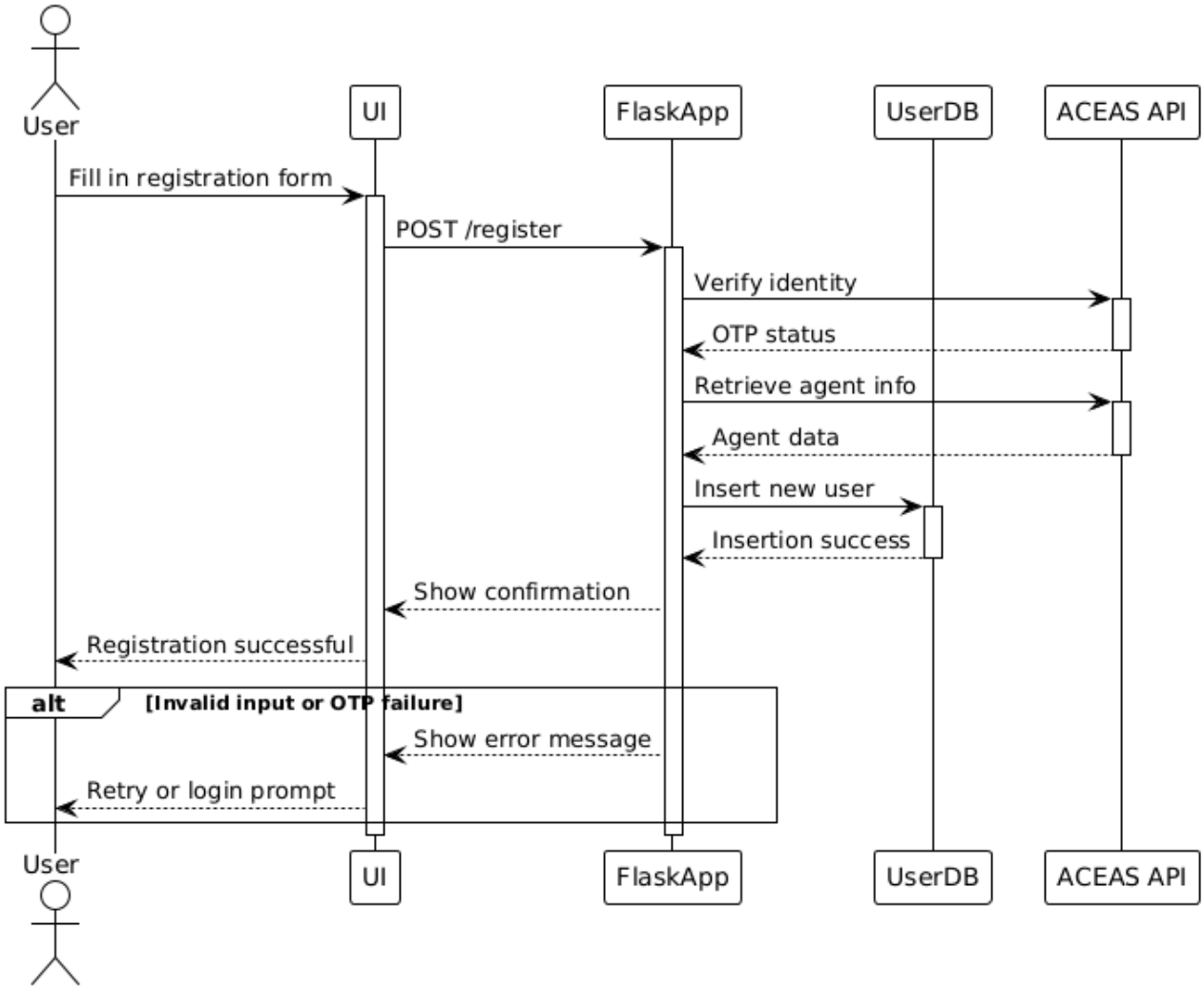


Dialog Map:

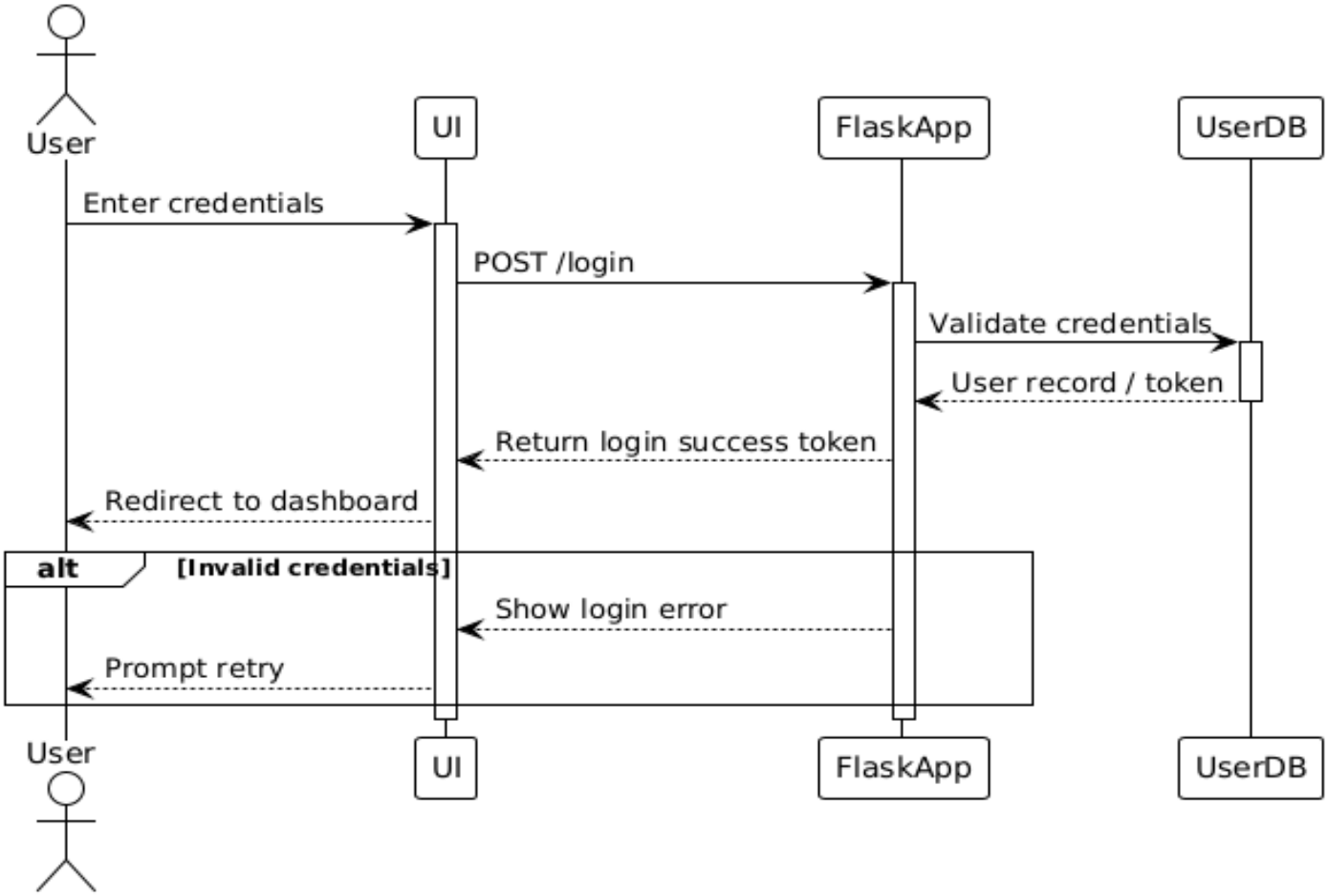

Entry
Landing Page Screen
entry / user loads the website
do / displays latest properties & a brief intro about our solution
exit / user clicks on login / register at the menu section
user clicks on register in menu
Signup Screen
do / displays register options; full name, username, password, seller/agent type & mobile number with an OTP feature
exit / user clicks on "Create Account"
user clicks on "login"
user clicks on "create account"
Login Page
do / display login options; username, password, account type: seller / agent,
exit / user clicks on login button / register here / forgot password feature
user clicks on "register here"
user clicks on "login" & login(username, password) == True
user clicks on "forgot password"
user clicks on "submit"
Forgot Password Page
do / displays forgot password options; username, OTP retrieval section & new password creation
exit / user clicks on "Submit"
user successfully logs in
User Profile
do / display profile details, such as username, full name, phone number
exit / set listing price & submit form
user clicks on "profile" button
(logged in as agent)
Agent's Homepage
do / display list of latest properties
logout
back
View Listed Property Screen
do / display list of listed properties, with home details and max commission percentages
exit / user clicks on "bid"
user clicks on "view listed properties" or "start bidding"
back
user clicks on "bid"
Place Bidding Screen
do / display option for agent to place a bid in 0.05 increments
user clicks on "edit"
inputs made are valid and changed
Edit User Profile Screen
do / display profile details
exit / user clicks on "submit"
(logged in as seller)
user clicks on "available listings for bidding"
back
View Bidded Property Screen
do / display list of properties that the agent has bid for
do / displays comparison between the agent's bid and the lowest bid
List Property Screen
do / display list property options to be filled: flat type, town, street name, max commission bid etc
do / includes a fetch past prices option to view similar past transactions
exit / set listing price & submit form
user clicks on "list property" in menu bar or "start listing" at the bottom of the page
Seller's Homepage
do / display list of latest properties uploaded
logout
user clicks on "submit"
user clicks on "home"
user clicks on "your property" in menu bar
Your Property Page
do / display list of properties listed, with descriptions of them
do / displays a "more details" button under each property listed
user clicks on "edit"
back
Edit Listing Page
do / display options to edit listing price, max commission, years remaining
do / display option for user to delete listing
exit / user clicks on "save changes" / "delete listing"
user clicks on "more details"
View Biddings Screen
do / display property agent details with commission bids proposed along with an "accept" button / "view profile" button
exit / display accepted notification after accept button is asserted
exit / displays option for seller to give a rating & review to the property agent
user clicks on "view profile"
Agent Profile Page
do / displays agent information & reviews & ratings of the agent

Sequence Diagram:

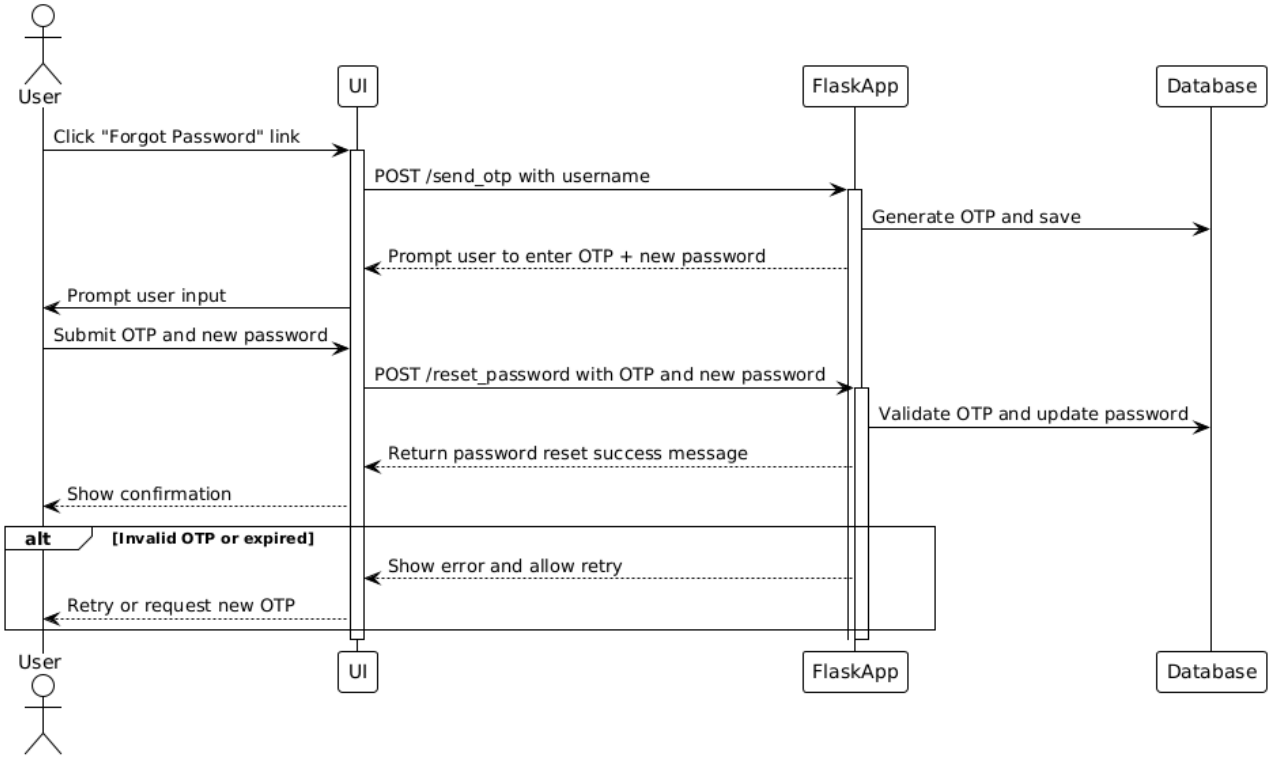
U1: Register Account



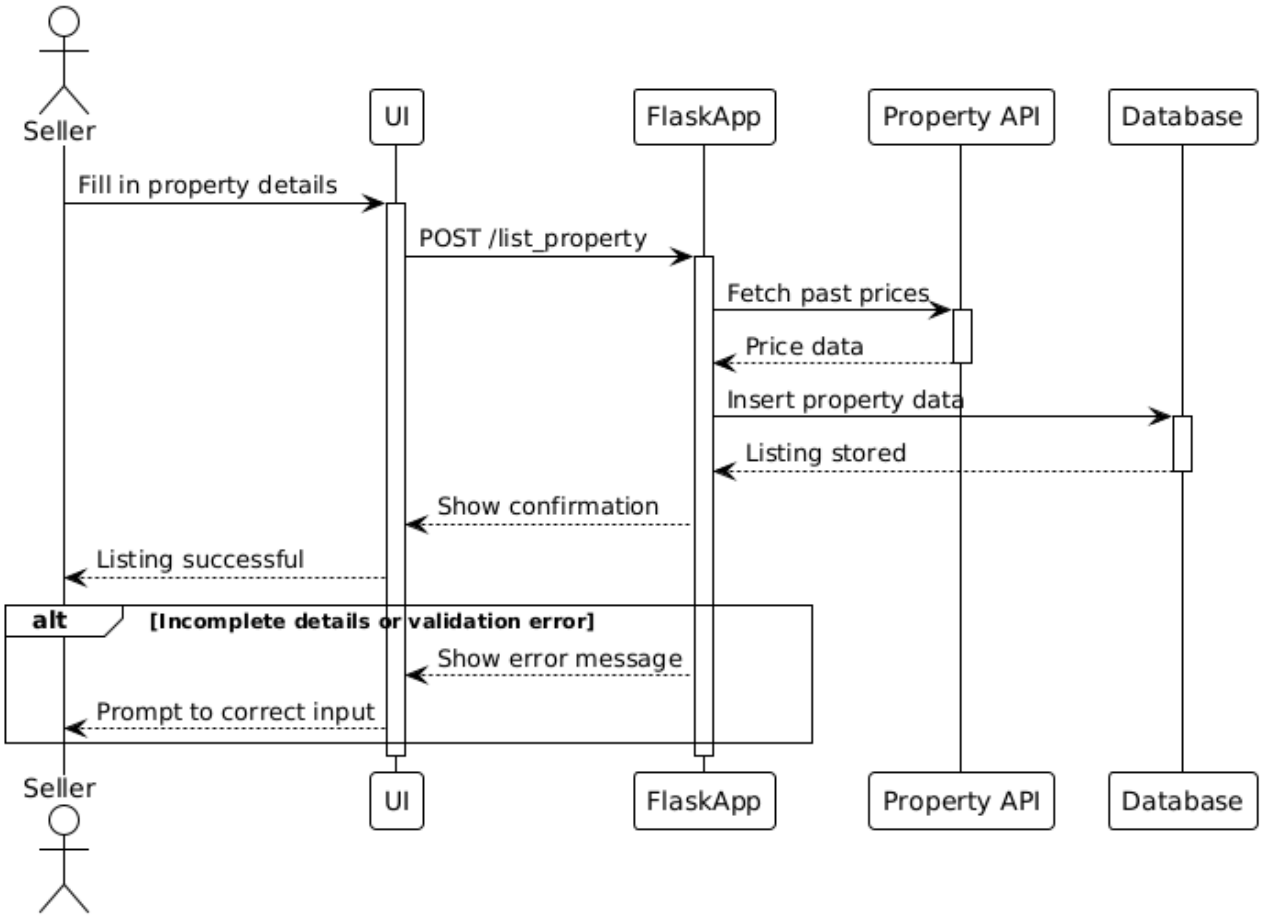
U2: Login


User
UI
FlaskApp
UserDB
Enter credentials
POST /login
Validate credentials
User record / token
Return login success token
Redirect to dashboard
alt
[Invalid credentials]
Show login error
Prompt retry
User
UI
FlaskApp
UserDB

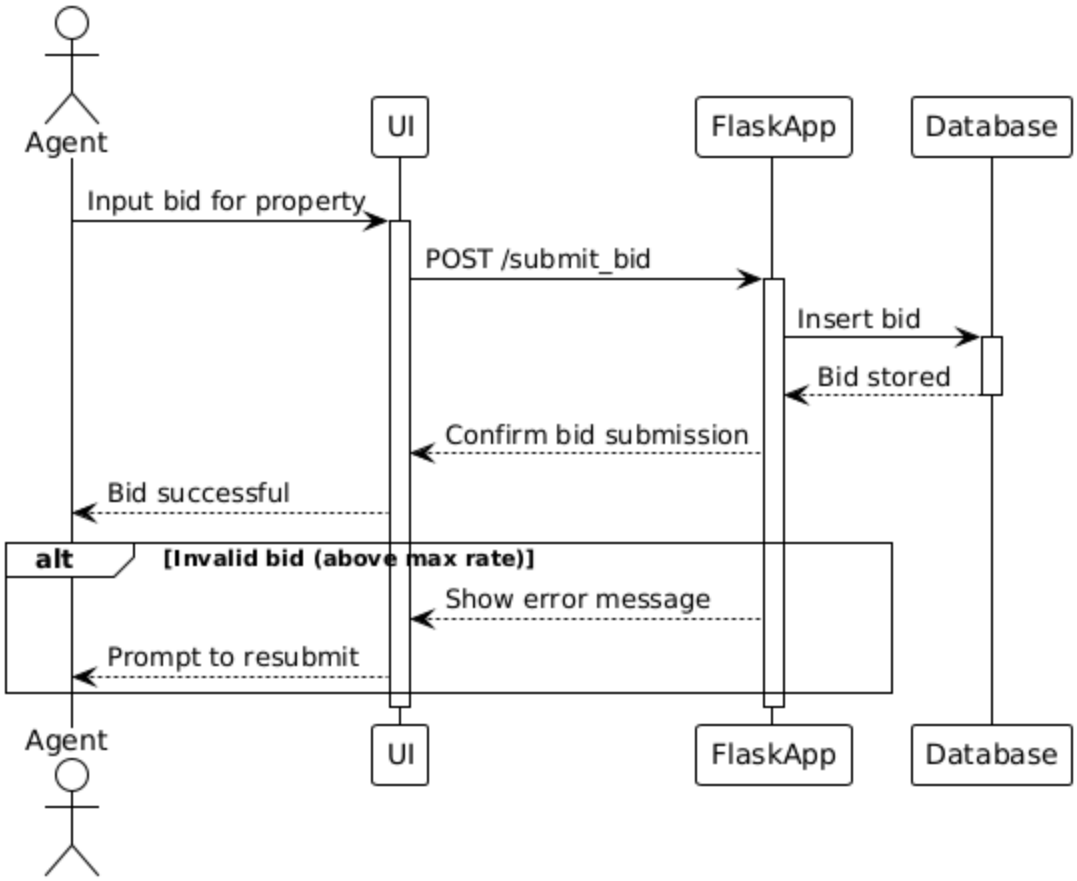
U3: Forgot Password



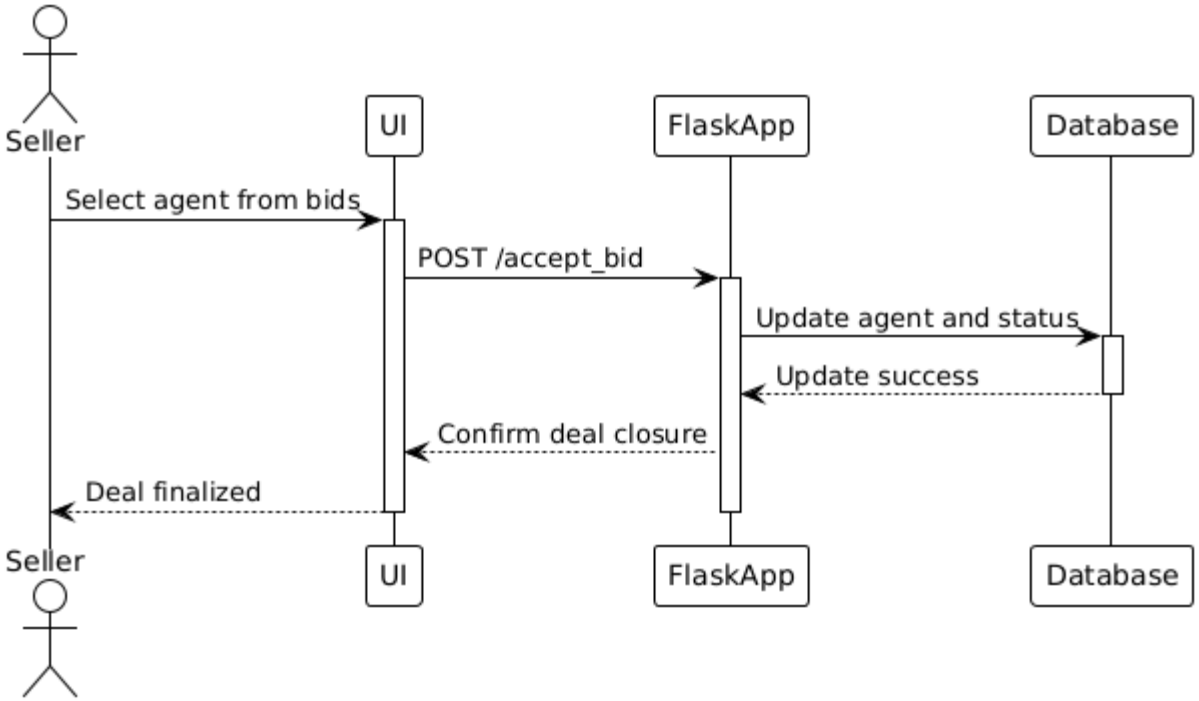
U4: List Property



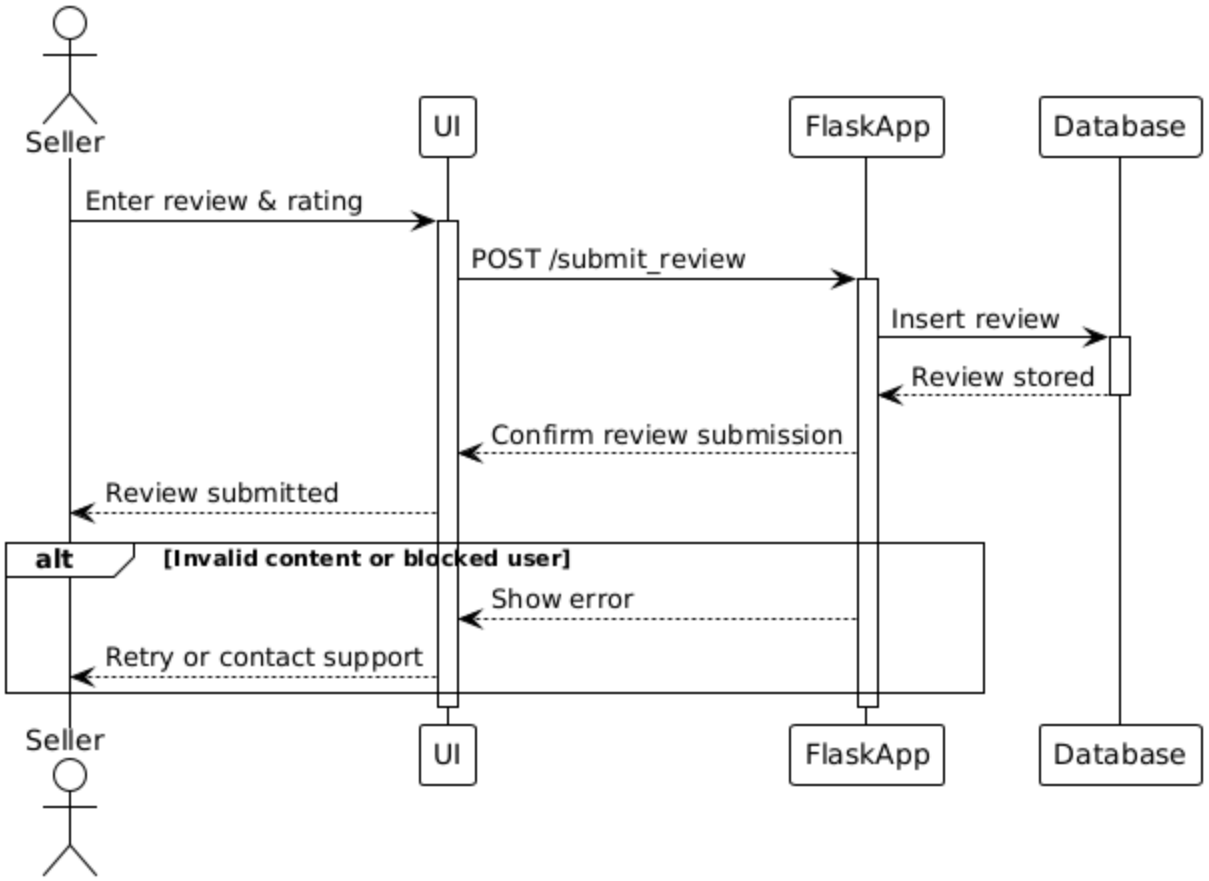
U5: Submit Commission Bid



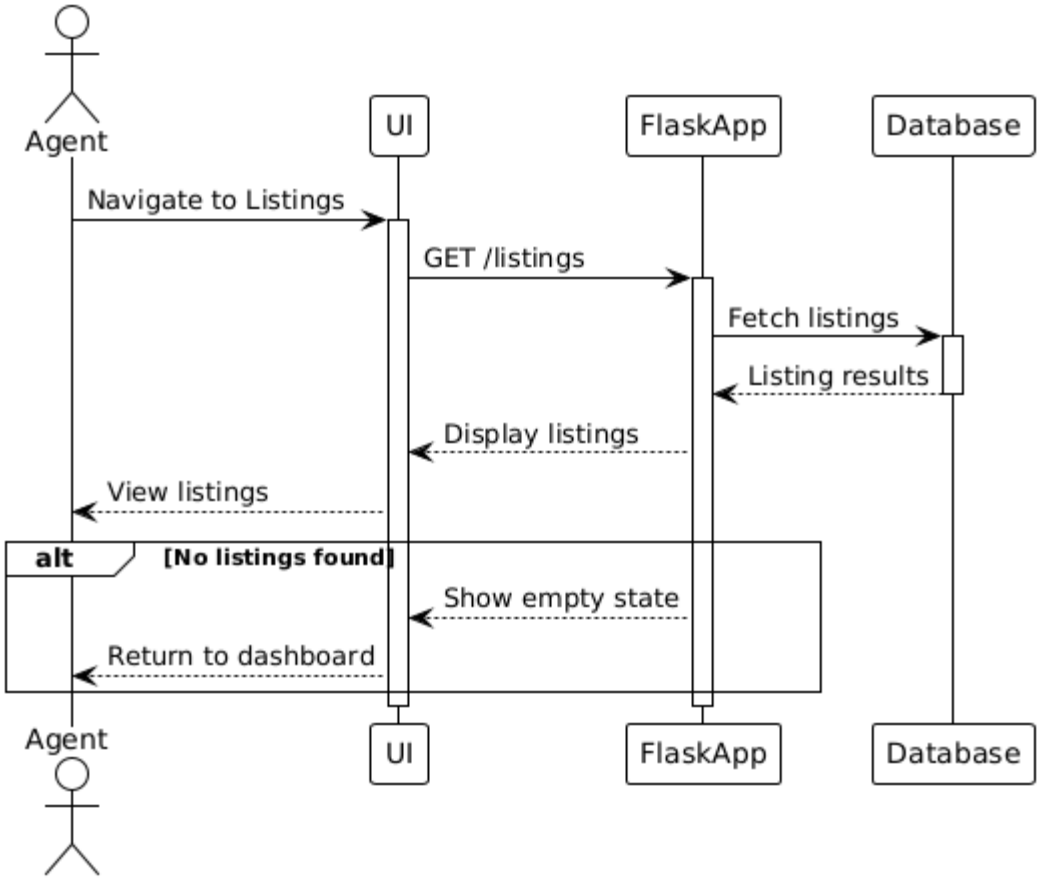
U6: Select Agent and Finalize Deal



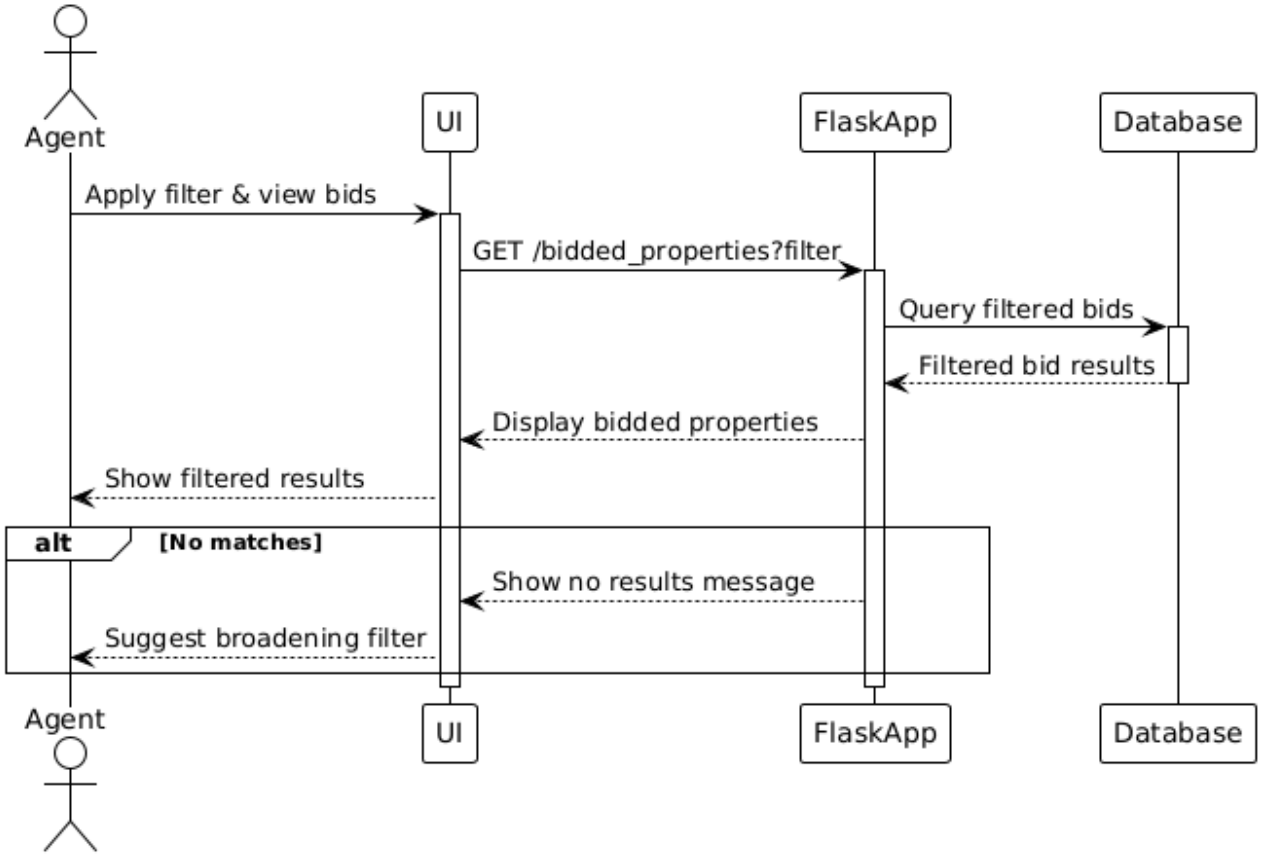
U7: Submit Rating & Review



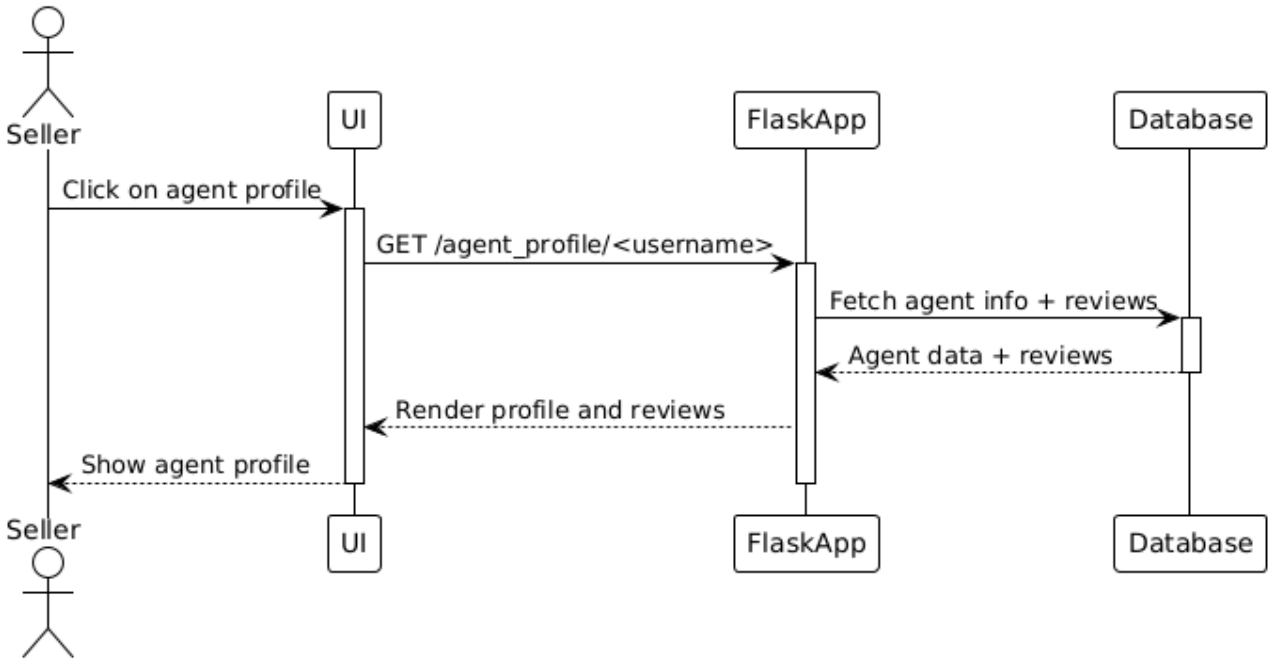
U8: View Listed Properties



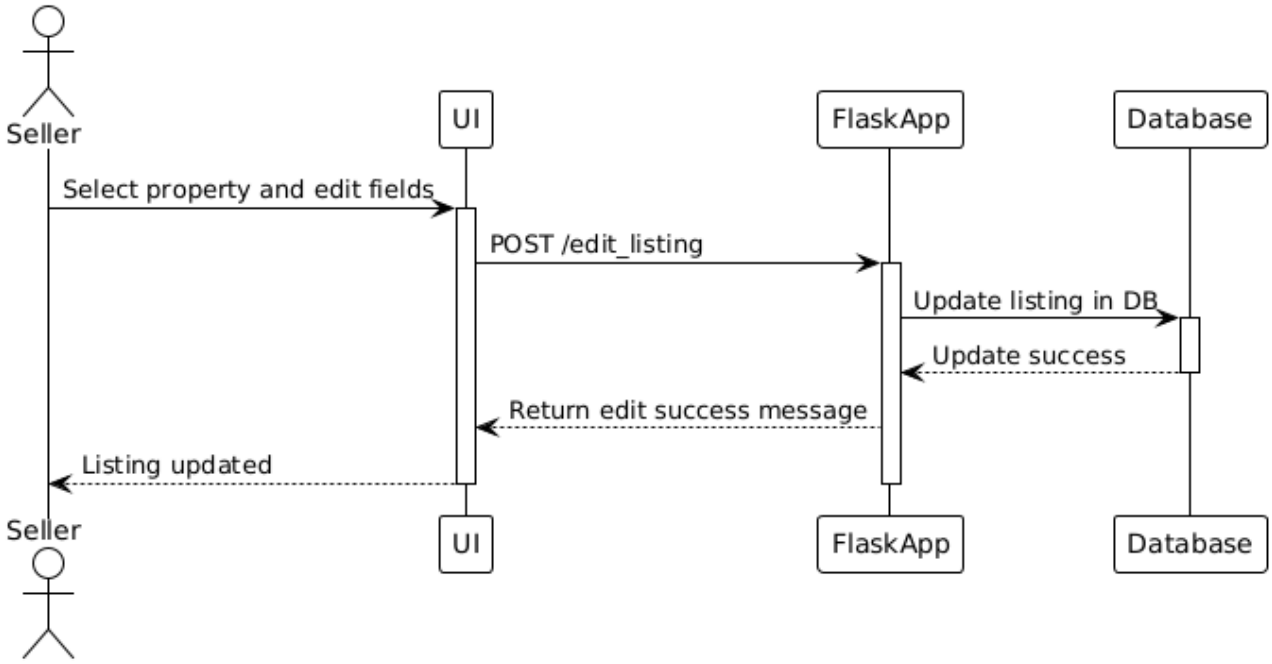
U9: View Bidded Property & Filter



U10: View Agent Profile & Reviews



U11: Edit Listings



Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

Source: http://www.frontiernet.net/~kwiegers/process_assets/srs_template.doc

Appendix D: Test Cases

Note: Black Box Testing and White Box Testing will be focused on the Account Creation, in particular the Register Account Use Case

Black Box Testing

I. AuthController (AuthRoutes)

Class to Test: auth_routes.py (Flask Blueprint)

The AuthRoutes Flask Blueprint handles user authentication workflows, primarily focusing on:

a. Registration:

- Creates seller/agent accounts with validation for phone numbers (+65 format), password strength (8+ chars with mixed case, numbers, and special characters), and unique usernames.
- For agents, verifies CEA license IDs against a government API from data.gov.sg
- Link: https://data.gov.sg/datasets/d_07c63be0f37e6e59c07a4ddc2fd87fcb/view?dataExplorerPage=4

b. OTP Verification:

- Integrates with Twilio to send SMS OTPs (or mocks for testing) and validates user-submitted codes.
- Currently only works for one phone number due to API free trial account limitations

c. Session Management:

- Temporarily stores OTPs in session and clears them post-verification.

d. Data Persistence:

- Saves validated user data to SQLite (accounts.db) with role-based fields (e.g., agent_id).

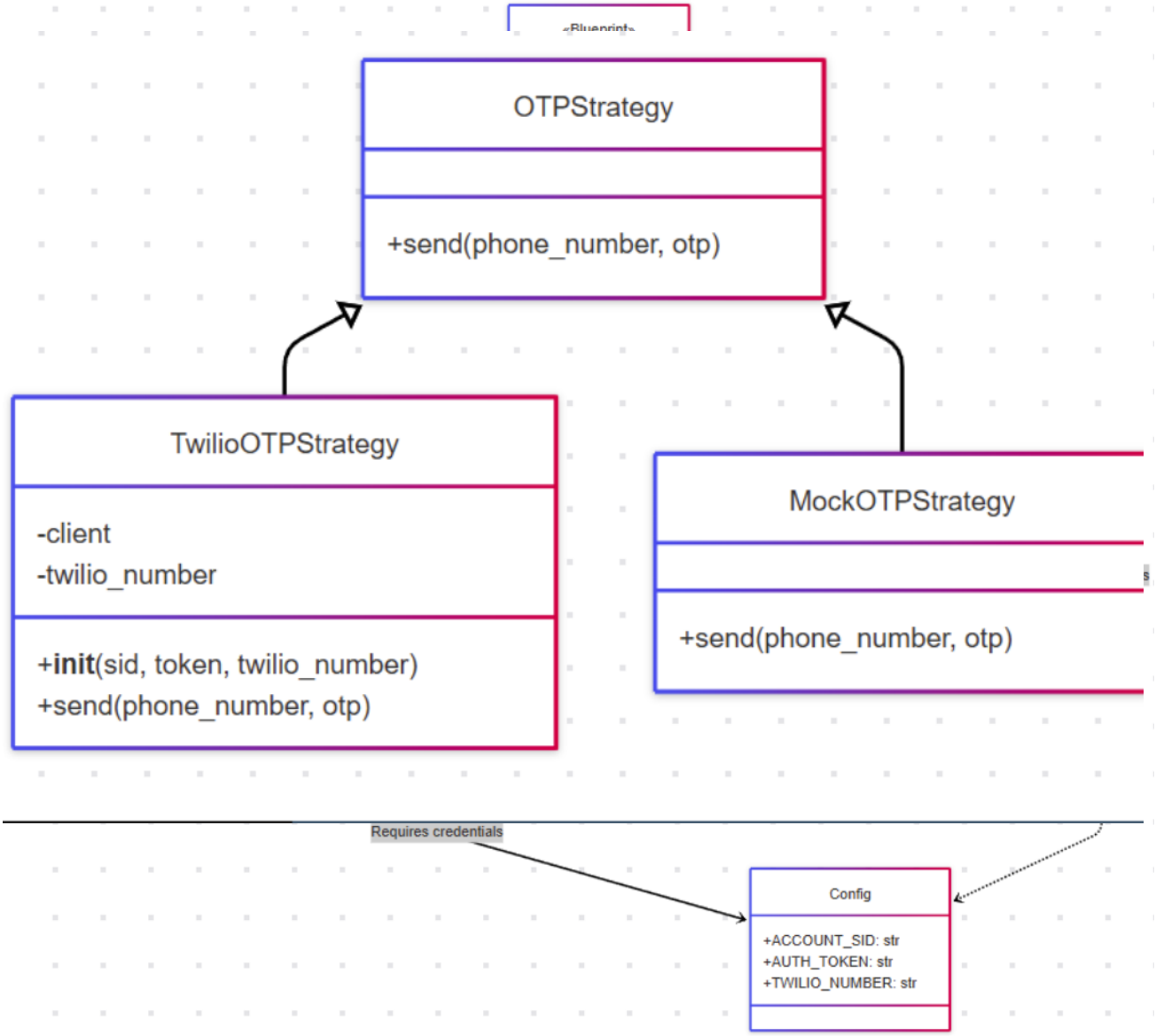
Key Features:

- Auto-formats Singaporean phone numbers (e.g., 91234567 → +6591234567).
- Uses the Strategy Pattern for OTP services (refer to images below or refer to 2006-SCSB-T4/HomeProperty/services/otp_service.py)
- Enforces strict input validation before database operations.

Example Flow:
User submits form → OTP sent → Validates inputs → Creates account → Redirects

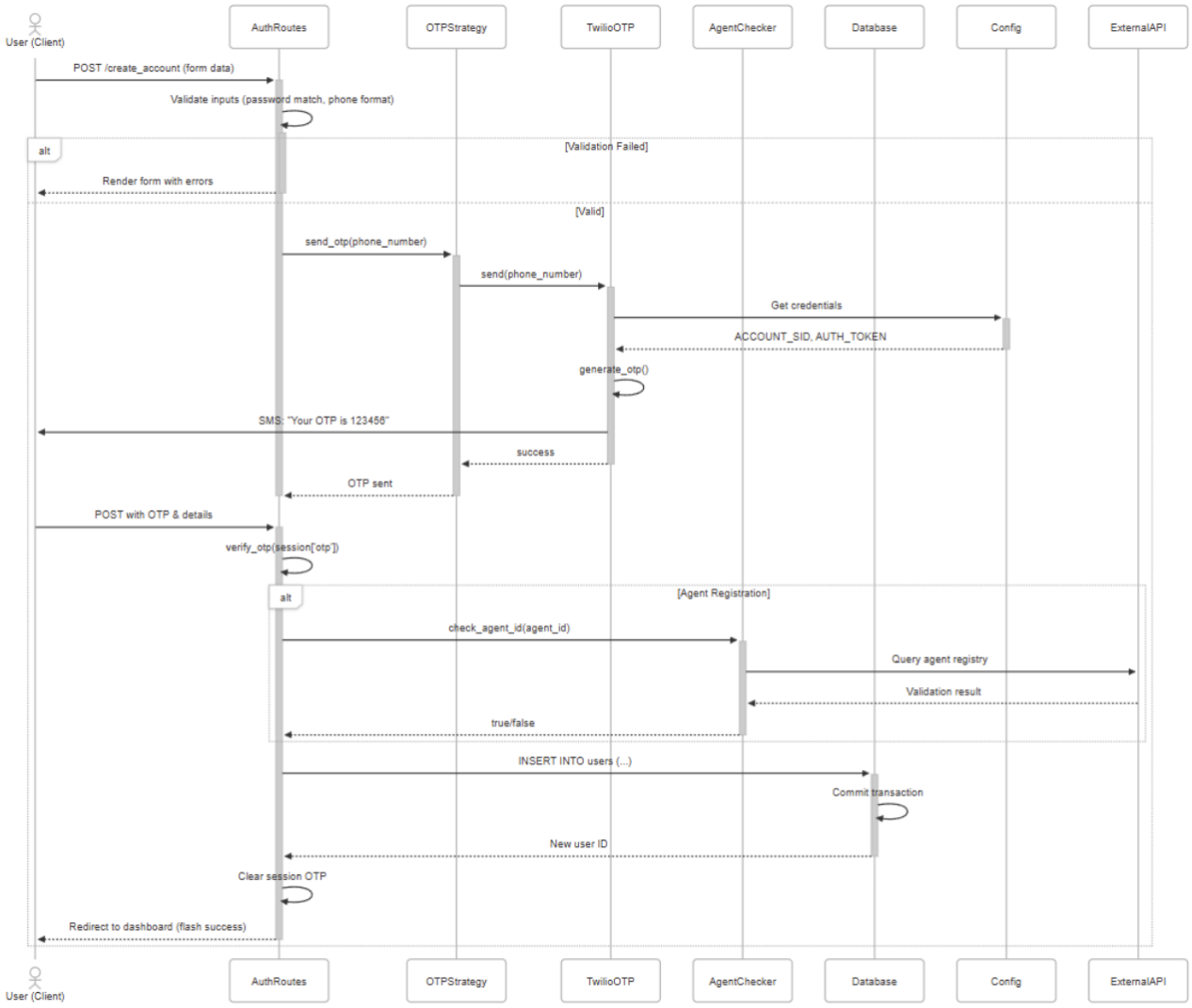
Class Diagram:

Refer to Lab 4 Folder if image is unclear



OTP Strategy (included in Class Diagram):

Sequence diagram:
Refer to Lab 4 Folder if image is unclear



Equivalence Class and Boundary Value Testing (Under Black Box Testing)
Registration Function
Valid Equivalence Class:
Full name, username, password, phone number, and user type with correct formats
Password: 8+ characters with uppercase, lowercase, number, and special character
Phone number: 8 digits with or without +65 prefix (system auto-formats)
OTP: 6-digit code matching sent value
Invalid Equivalence Class:
Missing required fields (name, username, password, etc.)
Passwords with incorrect formats (missing rules)
Invalid phone numbers (≠8 digits, wrong prefix)
Incorrect/expired OTP codes
If role selected is Seller:

Valid: No agent ID required
Invalid: Agent ID field filled (should be stored as NULL)
If role selected is Agent:
Valid: CEA agent ID registered in government database
Invalid: Invalid/empty agent ID
Boundary Value Testing

Phone Number:
Lower bound: 7 digits (9123456) → Invalid
Upper bound: 8 digits (91234567) → Valid

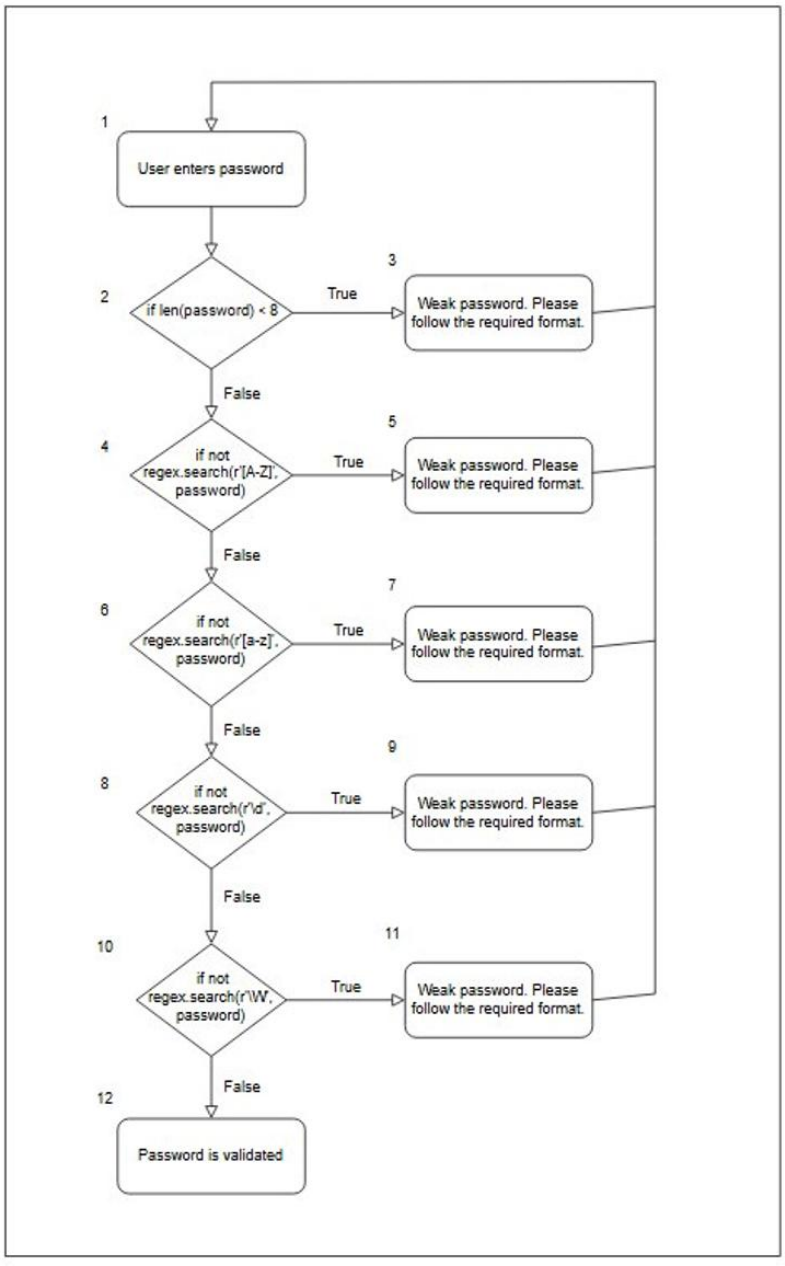
Password Length:
Lower bound: 7 characters (Apple1!) → Invalid
Upper bound: 8 characters (Apple1!@) → Valid
OTP Code:
Lower bound: 100000 → Valid if correct
Upper bound: 999999 → Valid if correct

Test Case #	Test Input	Expected Output	Actual Output (Tick means same as Expected Output)
1	Valid seller account (unique username, strong password, verified phone)	Account created successfully, redirect to login	✓
2	Valid agent account (valid agent ID, unique username, strong password, verified phone)	Account created successfully, redirect to login	✓
3	Duplicate username (no 2 accounts can have same username regardless of account type)	Error: "Username already taken"	✓
4	Password without uppercase letter	Error: "Password must contain uppercase letter"	✓
5	Password without lowercase letter	Error: "Password must contain lowercase letter"	✓
6	Password without number	Error: "Password must contain number"	✓
7	Password without special character	Error: "Password must contain special character"	✓
8	Phone number without +65 prefix	System auto-prepends +65, OTP sent	✓
9	Invalid OTP entered	Error: "Invalid OTP"	✓
10	Agent account with invalid agent ID	Error: "Invalid agent registration number"	✓
11	Agent account with empty agent ID	Error: "Agent registration number required"	✓

12	Missing required field (e.g. empty username)	Error highlighting missing field	✓
13	Password and confirm password don't match	Error: "Passwords do not match"	✓
14	Seller account with agent ID field filled	Agent ID stored as NULL in database	✓
15	Multiple OTP requests for same number	Only latest OTP is valid	✓
16	Valid credentials but Twilio SMS failure	Error: "Failed to send OTP" with retry option	✓
17	Session timeout during OTP verification	Prompt to restart verification process	✓
18	Existing user forgot password flow	Successfully resets password after OTP verification	✓
19	Password with 7 characters (just below min)	Error: "Password must be at least 8 characters long..."	✓
20	Password with exactly 8 characters (min valid)	Account created successfully	✓
21	OTP = 100000 (lower boundary)	OTP accepted if correct	✓
22	OTP = 999999 (upper boundary)	OTP accepted if correct	✓
23	Username with exactly max allowed length (e.g., 20 characters)	Account created successfully	✓
24	Username exceeding max allowed length (e.g., 21 characters)	Error: "Username too long"	✓

6.1.1 White Box Testing

I. Strong Password Check



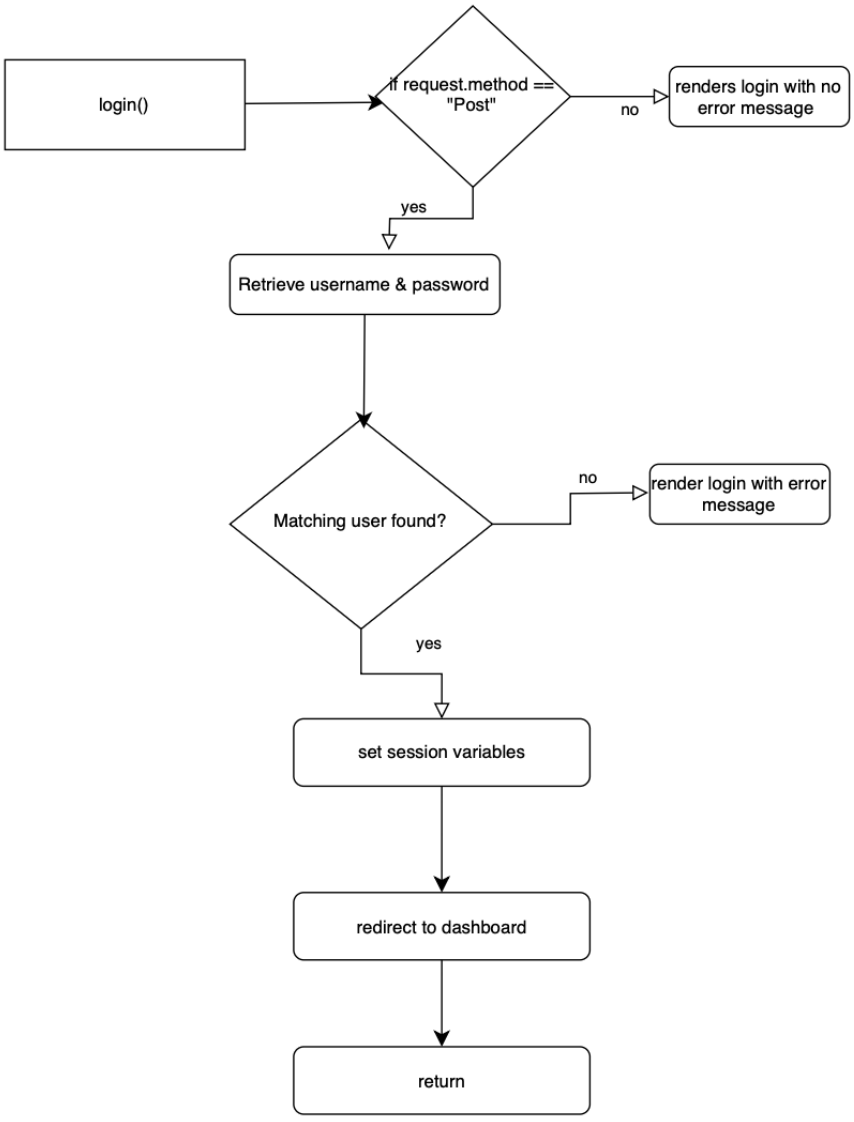
Test Cases and Results

These test cases validate the password strength logic, which requires that a password must:

- Be at least **8 characters long**
- Contain **at least one uppercase letter**
- Contain **at least one lowercase letter**
- Contain **at least one digit**
- Include **at least one special character** (e.g., !, @, #)

No.	Test Input	Expected Output	Actual Output	Pass?
1	Password : Short1! (Too short, 7 chars)	Weak password. Please follow the required format.	Weak password. Please follow the required format.	Yes
2	Password : longpassword1! (No uppercase)	Weak password. Please follow the required format.	Weak password. Please follow the required format.	Yes
3	Password : LONGPASSWORD1!(No lowercase)	Weak password. Please follow the required format.	Weak password. Please follow the required format.	Yes
4	Password : StrongPass! (No digit)	Weak password. Please follow the required format.	Weak password. Please follow the required format.	Yes

II. Login



Test Cases and Results

These test cases verify the login logic by checking for:

- Valid credentials and account type
- Invalid credentials
- Missing required fields

No.	Test Input	Expected Output	Actual Output	Pass?
1	"Username: Larry20, Password: Password21@, Account Type: seller" (in DB)	Redirect to / seller dashboard	Redirect to / seller dashboard	Yes
2	"Username: alice, Password: Alice123!, Account Type: seller" (not in DB)	"Invalid credentials or account does not exist"	"Invalid credentials or account does not exist"	Yes
3	"Username: Larry20, Account Type: Property Agent"	"Fill out this field"	"Fill out this field"	Yes