Software Requirements Specification

for

<Project>

Version 1.0 approved

Prepared by <author>

<organization>

<05/02/2025>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This document specifies the software requirements for the **Resale Flat Recommendation System**, a web-based platform that is aimed at assisting both couples and investors in evaluating and comparing resale flats based on affordability, appreciation, and area preferences. The system utilizes publicly available government data to provide real-time insights into resale flat prices, historical trends, and personalized recommendations. The document outlines the functional and non-functional requirements necessary for the system's operation.

<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>

1.2 Document Conventions

Headings follow a hierarchical numbering structure (1.1, 1.2, 2.1, etc.).

Priority levels are categorized as **High, Medium, or Low** to indicate their importance.

Functional requirements are denoted with "**REQ-#**" identifiers.

Bolded words and phrases highlight **key concepts, system components, and critical actions** within the document for better readability and emphasis.

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

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)

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>

1.4 Product Scope

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>

- *Simplify the flat selection process by allowing users to filter flats based on budget, location, remaining lease, commute tolerance, and amenities.*
- *Provide insights using historical resale price data and plotting it onto a graph*
- *Allow easier user decision-making with side-by-side comparisons of different flats based off criteria above*

1.5 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

2. Overall Description

2.1 Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

Designed to replace or complement existing property listing platforms like PropertyGuru. Unlike traditional property platforms, which primarily focus on listing properties, this system uses publicly available government data (e.g., from data.gov.sg) to provide personalized, data-driven recommendations for resale flats.

Limitations of existing property platforms include:

- *Lack of personalized recommendations based on user preferences (e.g., budget, commute tolerance, proximity to amenities).*
- *Limited historical data analysis to help users understand price trends and make informed decisions.*
- *Absence of side-by-side comparisons of resale flats based on user-defined criteria.*

While the system operates independently, it is designed to complement or replace existing property platforms by offering advanced features that are not currently available. It does not rely on other systems for its core functionality but integrates with external APIs to enhance its capabilities.

2.2 Product Functions

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

- *Recommendation of flats based off preferences*
- *Comparison feature between flats*
- *User Account Management to store preferences*

2.3 User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

- *Homebuyers*

- *Description: Individuals or couples looking to purchase a resale flat, possibly for the first time.*

- *Characteristics:*
- *Frequency of Use: Moderate to high (frequent interaction as they explore various options before purchase).*
- *Subset of Functions Used: Account Login, flat recommendation, flat comparison*
- *Technical Expertise: Basic to moderate (may not be tech-savvy but comfortable using simple tools).*
- *Security Levels: Standard access to all features.*
- *Experience: Little to no experience in real estate transactions*
- *Education Level: General public with varying levels of education*

2.4 Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

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.*

- *Mobile Devices: Android and iOS devices*

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

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

2.6 User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

2.7 Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

3. External Interface Requirements

3.1 User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

3.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

3.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

3.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 Resale Flat Recommendation

<Don't really say "System Feature 1." State the feature name in just a few words.>

4.1.1 Description and Priority

This feature analyzes historical resale price data to provide users with insights into price trends, helping them compare resale flats effectively. It evaluates available flats based on user-defined criteria, including budget, flat size, floor level, remaining lease, commute tolerance for workplace locations, and proximity to facilities (e.g., clinics, gyms).

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

User Action: Inputs preferences such as budget, flat size, floor level, remaining lease, workplace location, commute tolerance, and desired nearby facilities. System retrieves relevant past transaction data from the preexisting database.

System Response: Retrieves relevant past transaction data from the preexisting database.

System Response: Analyzes the data to generate price trend visualizations and provides flat recommendations that match the filtered criteria.

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

REQ-1: The system shall allow users to input at least one of the following preferences via an interactive form: budget, flat size, floor level, remaining lease, workplace locations, commute tolerance, and preferred nearby facilities.

REQ-2: The website shall retrieve and process historical resale price data from the existing database.

REQ-3: The website shall provide graphical visualization of price trends based on user-selected inputs.

REQ-4: The system must be able to filter results based on the user's previously input preferences.

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should

be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:
REQ-2:

4.2 Resale Flat Comparison

4.2.1 Description and Priority

Allows users to compare resale flat factors across different flats, such as locations, flat types, and time periods.
Medium Priority: Essential for users to make informed decisions.

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.2.2 Stimulus/Response Sequences

Users select multiple locations or flat types for comparison.

The system retrieves and processes relevant resale price data.

The system displays a side-by-side comparison of flats attributes such as size, distance from workplace, price, etc.)

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.2.3 Functional Requirements

REQ-1: The system shall allow users to select multiple flats for comparison.

REQ-2: The system shall retrieve and display all the information of the selected flats and compare between each other.

REQ-3: The system should highlight the information that is better in respect of the selected flats such as Higher Remaining Lease, Floor Areas, Flat Type etc.

4.3 User Account & Preference Storage (which includes reviews and ratings)

4.3.1 Description and Priority

Allows users to save their preferred locations, flat types, and budgets for future searches.

The system can provide more tailored recommendations.

It keeps reviews tied to verified users, reducing spam or manipulation.

Medium Priority: Enhances user experience and personalization.
r

Priority: Medium.

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.3.2 Stimulus/Response Sequences

The user creates an account and saves preferred locations, flat types, and budget preferences.

System stores these preferences securely.

The system provides personalized flat recommendations and allows users to access their saved preferences anytime.

System ensures that reviews and ratings are linked to verified users.

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.3.3 Functional Requirements

REQ-1: The system shall allow users to create an account and save housing preferences (budget, locations, flat types, etc.).

REQ-2: The system shall provide personalized recommendations based on saved preferences.

REQ-3: The system must allow users to key in reviews and rating to the recommended flats

REQ-3: The system shall ensure that reviews and ratings are tied to verified users to prevent spam or manipulation.

REQ-4: The system shall allow users to retrieve and modify their saved preferences at any time.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

- *The system should be able to retrieve and display resale flat recommendations within 3 seconds for an optimal user experience.*
- *The resale price trend visualizations should be generated and displayed within 3 seconds after user input submission.*
- *The system should ensure that searches and data retrieval operations do not exceed 5 seconds, even under peak load.*
- *The user-submitted review and rating system should process and store reviews within 2 seconds of submission.*

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

- *The system shall prevent unauthorized modifications or deletions of resale flat data to ensure data integrity.*
- *The system should handle failed transactions gracefully and provide appropriate error messages to users.*
- *The website should provide warnings to users about potential limitations or inaccuracies in resale flat predictions based on historical data trends.*
- *The system should have automated backups performed daily to prevent data loss.*

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication

requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

- *All user data, including login credentials and preferences, must be encrypted using industry-standard encryption methods.*
- *User authentication shall be enforced for submitting reviews, with email or phone verification required for account registration.*
- *The system must follow Singapore's Personal Data Protection Act (PDPA) guidelines to ensure proper handling of user data.*

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

- *Usability: The website should have an intuitive UI, ensuring that first-time users can complete a flat search in less than 2 minutes.*
- *Reliability: The system should have an uptime of at least 99.5% to ensure availability for users.*
- *Maintainability: The system should be modular to facilitate easy updates and enhancements.*
- *Interoperability: The system should be compatible with modern web browsers, including Chrome, Firefox, Safari, and Edge.*
- *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.*

5.5 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the 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

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

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