SOFTWARE REQUIREMENTS SPECIFICATION

For

Boarding House and Apartment Locator System

Prepared by:

Eleceed Compiler

Kiah Karim

Kriz John Ragadio

Vince Cyrus Salvador

Submitted in fulfillment requirements for Integrative Programming and Technologies 2 (IT305)

August 17, 2025

Table of Contents

1. Introduction

The Boarding House and Apartment Locator System is a web-based platform designed to simplify the search for rental properties. Its primary goal is to create a centralized and efficient marketplace for both property owners and renters, reducing the time and effort typically involved in the search and listing process.

1.1 Purpose

The purpose of this system is to serve as a comprehensive database and search engine for available boarding houses and apartments. It aims to connect property owners who wish to rent their properties with individuals seeking accommodation. For renters, the system provides a convenient way to filter and find suitable properties based on their specific location. For property owners, it offers a simple and effective way to list, manage, and advertise their available units, reaching a wider audience of potential tenants. The intended users are primarily students, professionals, and families looking for rental housing, as well as property managers and individual landlords.

1.2 Intended Audience

The client for this system is a property management company specializing in rental properties. The primary users of the system are renters (students, professionals, and families) looking for boarding houses and apartments and property owners/landlords who want to list their properties.

1.3 Product Scope

The Boarding House and Apartment Locator System will include the following key functions:

- Property Listing Management: Property owners can create, edit, and remove property listings.
- Search and Filter: Users can search for properties based on location, price range, number of bedrooms, amenities, and other criteria.
- User Profiles: Renters can create profiles to save their favorite listings and contact information. Property owners can manage their listings and view inquiries.
- Inquiry System: A built-in messaging or inquiry system will allow potential renters to contact property owners directly.
- Image and Video Support: Listings will support the upload of photos and videos to provide a visual representation of the properties.

The system's boundaries are limited to the functions of a rental property locator. It does not include features for handling financial transactions, such as rent payments or security deposits. It also does not manage legal contracts or tenant-landlord disputes. Its core function is strictly to facilitate the connection and communication between renters and property owners.

1.4 Definitions, Acronyms, and Abbreviations

API – Application Programming Interface

Rental Property – Refers to apartments, boarding houses, bed spacers, or rooms available for rent.

User – A person who is searching for or posting a rental property.

Listing – A specific entry or advertisement for a property in the system, which includes details like location, price, description, and photos.

2. Overall Description

2.1 User Characteristics

The Boarding House and Apartment Locator System is designed for two primary user groups: renters and property owners.

- Renters: This group includes students, professionals, and families seeking rental accommodation. They are expected to have basic computer literacy and be familiar with using search engines and online forms. They should be able to navigate a website. No specialized technical skills are necessary. The system's UI will be intuitive to accommodate a broad range of users.
- Property Owners/Landlords: This group consists of individuals and property management companies who wish to list and manage their rental properties. They are expected to have moderate computer skills, including the ability to create accounts, fill out detailed forms, upload high-resolution images, and respond to inquiries. They need to be comfortable with managing their property listings and updating information as needed. The system will provide clear instructions and a user-friendly dashboard to simplify these tasks.

2.2 Constraints

The system is subject to several constraints to ensure its functionality and legality. Users must have a stable internet connection and access to a modern web browser on any device, as the platform is entirely web-based and requires no special software installation. It must be compatible with popular browsers like Chrome, Firefox, and Safari and feature a responsive design for optimal viewing on desktops, tablets, and mobile phones. Furthermore, the system is constrained by legal requirements, needing to comply with data privacy regulations such as GDPR, and must also adhere to fair housing laws to prevent discrimination in property listings. Lastly, it must be able to handle a high volume of users and search queries without a noticeable drop in performance, ensuring quick and efficient delivery of search results.

2.3 Assumptions and Dependencies

Assumptions:

- Users have reliable internet access.
- Property owners will provide accurate and up-to-date information about their listings.
- The system will not be responsible for verifying the authenticity of property listings or the financial capabilities of renters.
- All communication and negotiations about rent, contracts, and deposits occur directly between the renter and property owner, outside of the system.

Dependencies:

- Mapping Service API: The system will rely on a third-party mapping service API (e.g., Google Maps, Mapbox) to display property locations and provide navigation.
- Payment Gateway (Optional, for premium features): If the system introduces premium features for property owners, it will depend on an external payment gateway (e.g., Gcash, PayPal, PayMaya).

3. Specific Requirements

3.1 Functional Requirements

The system must include the following key functions:

- **Property Listing Management**: Property owners must be able to create, edit, and remove their property listings.
- **Search and Filter**: Users need to be able to search for properties based on location, price range, number of bedrooms, amenities, and other criteria. The system's interface demonstrates this capability, allowing users to filter by property type (e.g., apartment or boarding house) and location (e.g., Barangay Balintawak).
- **User Profiles**: Renters should be able to create profiles to save their favorite listings and contact information. Property owners can also manage their listings and view inquiries through their profiles.
- **Inquiry System**: The system will have a built-in messaging or inquiry system to allow potential renters to contact property owners directly.
- **Image and Video Support**: Listings must support the upload of photos and videos to provide a visual representation of the properties.

- **Login/Signup**: The system must have a secure login/signup interface to allow users to register and access their accounts. It should distinguish between owners and renters, providing each with relevant features.
- **Detailed Property View**: When a user selects a listing, the system must display a detailed pop-up with essential property information, such as the owner, location, price, and description.
- **Add Listing**: Property owners must have a feature to post new rental properties, enabling them to fill out a form with details like the property name, type, location, price, and description.

3.2 Non-Functional Requirements

The non-functional requirements focus on the system's performance, constraints, and dependencies.

- **Performance**: The system must be able to handle a high volume of users and search queries without a noticeable drop in performance, ensuring quick and efficient delivery of search results.
- **Usability**: The user interface must be intuitive to accommodate users with basic computer literacy. For property owners, the system should provide clear instructions and a user-friendly dashboard to simplify tasks like managing listings and responding to inquiries.
- **Reliability**: The system assumes that users have reliable internet access and that property owners will provide accurate and up-to-date information about their listings.

- **Legal & Security**: The system must comply with data privacy regulations like GDPR and fair housing laws to prevent discrimination. It must also ensure that only authorized users can manage or inquire about listings.
- **Compatibility**: The platform is entirely web-based and must be compatible with popular modern web browsers such as Chrome, Firefox, and Safari. It also needs to have a responsive design for optimal viewing on desktops, tablets, and mobile phones.

3.3 External Interface Requirements

User Interfaces

The system features a web-based user interface designed for intuitive navigation and ease of use for both renters and property owners.

- **Available Listings Interface**: Displays all available properties in a clean, card-based layout, allowing for quick comparison and navigation. Users can search for and filter properties based on type (e.g., apartment, boarding house) and location.
- **Login / Signup Pop-up**: A pop-up interface for user authentication and registration. It distinguishes between **Owners** and **Renters**, ensuring each user group has access to relevant features.
- **Detailed Listing View**: A pop-up that appears when a user selects a property, showing essential details such as the owner, location, price, and a description.
- **Add Listing Interface**: A form-based interface for property owners to input details and publish new listings.

Hardware Interfaces

The system is entirely web-based and does not require any special hardware installation. Users only need a device with a modern web browser and a stable internet connection. The responsive design ensures optimal viewing on desktops, tablets, and mobile phones.

Software Interfaces

The system relies on software interfaces for functionality and potential future features.

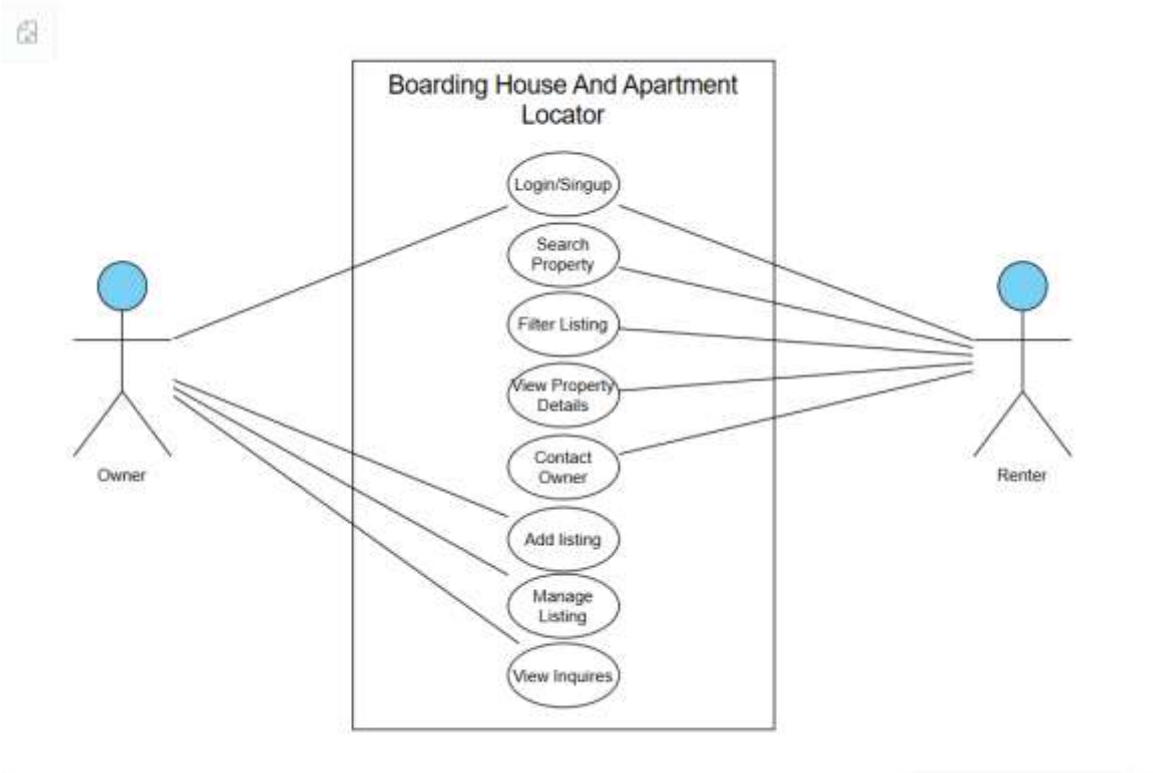
- **Mapping Service API**: The system depends on a third-party mapping service API (e.g., Google Maps, Mapbox) to display property locations and provide navigation.
- **Payment Gateway (Optional)**: If premium features are introduced for property owners, the system will depend on an external payment gateway like Gcash, PayPal, or PayMaya to handle transactions.

Communication Protocols

The system is a web-based platform. It requires a stable internet connection and uses standard web protocols (e.g., HTTPS) for communication between the user's browser and the server. This ensures the secure and efficient delivery of search results and other data.

3.4 System Models

This section presents a high level, visual representation of the system's functions and the way users interact with them. A case diagram is the most suitable model for this purpose as it illustrates the interactions between the primary users, the Renter and the Property Owner and the system itself.



- **Renter**:
    - **Search for Property**: Users can search for properties based on various criteria.
    - **Filter Listings**: Users can narrow down
    - **View Property Details**: Users can click on a listing to see detailed information about the property.
    - **Contact Owner**: Users can contact property owners directly through an inquiry system.
    - **Login/Signup**: Users can access their accounts or register.
- **Property Owner**:
    - **Login/Signup**: Owners can register and log into their accounts.
    - **Add Listing**: Owners can post new rental properties.
    - **Manage Listings**: Owners can create, edit, and remove property listings.
    - **View Inquiries**: Owners can view inquiries from potential renters.

3.5 Database Design

Relationships

1. **users → listings** (1:N)
    - One user (owner) can create multiple listings
    - One listing belongs to one user (owner)
2. **users → reviews** (1:N)
    - One user (renter) can write multiple reviews
    - One review is written by one user (renter)
3. **listings → reviews** (1:N)
    - One listing can have multiple reviews
    - One review is for one listing
4. **users → favorites** (1:N)
    - One user can favorite multiple listings
    - One favorite entry belongs to one user
5. **listings → favorites** (1:N)
    - One listing can be favorited by multiple users
    - One favorite entry points to one listing
6. **users → bookings** (1:N)
    - One user (renter) can make multiple bookings
    - One booking is made by one user (renter)
7. **listings → bookings** (1:N)
    - One listing can have multiple bookings
    - One booking is for one listing
8. **users → reservations** (1:N)
    - One user (renter) can make multiple reservations

    - One reservation is made by one user (renter)
9. **listings** → **reservations** (1:N)
    - One listing can have multiple reservations
    - One reservation is for one listing

These entities and relationships form the foundation of the database design that supports all the functional requirements described in the document, including property listing management, search and filtering, user profiles, and the inquiry system.

Constraints

1. **Key Uniqueness: Primary Keys (PK)** (id) identify every record, and **Unique Keys** enforce non-duplication for users.username and for the combination of renter_id/listing_id in the favorites table.

2. **Data Validation: NOT NULL** ensures critical fields are never empty. **ENUM** constraints restrict values for roles and statuses (e.g., 'owner', 'pending'). **CHECK** constraints validate data logic, like ensuring ratings are 1-5 and booking start dates precede end dates.

3. **Referential Integrity (FK Actions):**

- **$ON DELETE RESTRICT** on the **bookings** table prevents the deletion of a user or listing if they are tied to an existing booking, safeguarding crucial historical transaction data.

3.6 Implementation

Technology and Development Stack

- **Application Type:** Modern, responsive **web application**.
- **Development Environment:**

    - **Local Server Stack: XAMPP**.
    - **Code Editor: Visual Studio Code**.
- **Core Technology Stack: Full-stack JavaScript solution**.

Backend (Server-Side)

- **Runtime Environment: Node.js**.
- **Framework: Express.js**.
- **Dependency Management: NPM (Node Package Manager)**

Frontend (Client-Side)

- **Core Technologies: HTML5, CSS3, and JavaScript**.

Database and Administration

- **Database System: MySQL**.
- **Database Administration: phpMyAdmin**.

Key External Integrations

- **Location and Navigation: Google Maps JavaScript API**.
- **Future Monetization (Payment Gateway Options): GCash, PayPal, or PayMaya**.