
Software Requirements Specification

for

Homestay Booking Website Project

Version 2.0

Prepared by

Group 6

Vu Nguyet Hang
Duong Quoc Khanh
Nguyen Thi Thu Trang
Nguyen Van Son
Nguyen Truong Giang

22028079
22028090
22028254
22028020
19021259

22028079@vnu.edu.vn
22028090@vnu.edu.vn
22028254@vnu.edu.vn
22028020@vnu.edu.vn
19021259@vnu.edu.vn

Instructor: Assoc. Prof. Dr. Dang Duc Hanh

Course: INT2208E_23

Teaching Assistant: Kieu Van Tuyen

Date: 04/05/2024

CONTENTS

1. Introduction	3
1.1 Purpose	4
1.2 Scope	4
1.3 Intended Audience and Document Overview	4
1.4 Definitions, Acronyms and Abbreviations	4
1.5 Document Conventions	5
1.6 References	5
2. Overall Description	5
2.1 Product Perspective	5
2.2 Product Functions	6
2.3 Design and Implementation Constraints	6
2.4 Assumptions and Dependencies	8
2.5 User Classes and Characteristics	8
3. Specific Requirements	9
3.1 External Interface Requirements	9
3.2 System Features	11
3.3 Use Case Model	16
4. Other Non-functional Requirements	24
4.1 Performance Requirements	24
4.2 Safety and Security Requirements	24
4.3 Software Quality Attributes	24

REVISIONS

Version	Primary Author(s)	Description of Version	Date Completed
#1	Group 6	Initial Specification	28/03/24
#2	Group 6	Change actors, project's name Update project's persepective diagram Use case flow update	04/05/24

1. Introduction

1.1 Purpose

The purpose of this document is to describe an online homestay booking system in detail: the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate, and how the system will react to external stimuli. This document is intended to help both developers and end users understand the website's functional and nonfunctional requirements.

1.2 Scope

The software product to be produced is the Online Homestay Booking system (OHB). The OHB system will:

- Allow users to search for homestays based on various criteria such as location, price, amenities, and ratings.
- Enable users to book homestays securely and conveniently through the platform.
- Provide users with the ability to track their bookings and manage reservations.
- Allow homestay hosts/hosts to create, edit, and manage their homestay listings.
- Facilitate online payments for homestay bookings.

The OHB system will not:

- Handle the physical operations of homestays such as housekeeping, maintenance, or staffing.
- Provide transportation services to and from homestays.

With the application of the OHB system, clients can get real-time information on room availability, rates, and special offers; homestays can save on commissions and boost revenue by reducing reliance on third-party channels and manual work for staffs, etc.

1.3 Intended Audience and Document Overview

People with different skill sets are likely to use this document for their own purposes, as follows:

Developers: This group will use the document to understand the technical requirements specifications and how they will be transformed into a system design when the final software application is written.

Testers: This group will use the document during unit testing and system integration testing to understand all system features and how the system responds to external stimuli.

End users: System end users will be more interested in the overall description of the OHB, which will help them understand the final OHB product that the developers and testers will deliver and be able to propose other requirements that the development team might miss.

1.4 Definitions, Acronyms and Abbreviations

Admin	administrative user
API	application programming interface

IEEE	Institute of Electrical and Electronics Engineers
OHB	Online Homestay Booking
SRS	Software Requirements Specification

1.5 Document Conventions

Headings	Arial/16 font size/Bold
Subheadings	Arial/14 font size/Bold
Body	Times New Roman/11 font size

Blue underlined texts are referenced links, special highlighting is done by making the text bold so that important keywords can easily be differentiated.

1.6 References

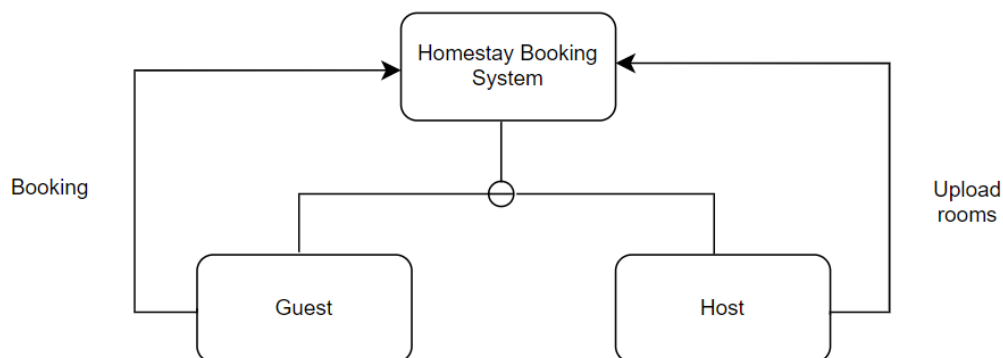
Institute of Electrical and Electronics Engineers (IEEE), Inc. (1998). IEEE Standard 830-1998, IEEE Recommended Practice for Software Requirements Specifications. New York, NY, USA: IEEE, Inc. Retrieved from: <http://www.math.uaa.alaska.edu/%7Eafkjm/cs401/IEEE830.pdf>

Software Requirements Specification template and example provided by teaching assistant.

2. Overall Description

2.1 Product Perspective

The Online Homestay Booking (OHB) system operates as an independent and self-contained product within the hospitality and travel industry. It interfaces with various components, including frontend technologies like React, backend technologies like Node.js and Express, and a MongoDB database.



2.2 Product Functions

The website will have some basic functions as follows:

- Register an account
- Log in to the website
- Edit account information

Besides, there will be specific functions for each type of user:

a) For guests

- Search and browse homestays
 - Guests can search for homestays based on location, price range, amenities, and other preferences.
 - They can browse through a comprehensive list of available homestays with detailed information.
- View homestay details: Guests can view detailed information about each homestay, including descriptions, photos, amenities, and user reviews.
- Make bookings: Guests can select their desired homestay, choose room types, specify dates, and make bookings securely through the platform.
- Manage bookings: Guests can view and manage their bookings, including modifying dates, canceling reservations (if allowed), and accessing booking confirmations.
- Search filters and sorting: Guests can use filters and sorting options to refine their search results based on criteria such as price, ratings, and proximity to landmarks.

b) For homestay hosts

- Create and manage homestay listings: Homestay hosts can create, edit, and manage listings for their properties, including adding descriptions, photos, amenities, and pricing details.
- Update availability: Homestay hosts can update the availability of rooms in real-time, ensuring accurate information is displayed to potential guests.
- Manage bookings: Homestay hosts can view and manage their booking lists, including confirming reservations, modifying dates, and processing cancellations.

2.3 Design and Implementation Constraints

Constraints for the Online Homestay Booking (OHB) system include:

a) Regulatory policies

Compliance with data privacy regulations (e.g GDPR) regarding the handling of personal data stored in the MongoDB database and payment card industry standards (PCI DSS) impose constraints on data handling, security measures, and payment processing.

b) Hardware limitations

- **Storage Space:** The website needs ample storage space to store information about homestays, bookings, user accounts, and other data. Limited storage space can restrict the amount of data that can be stored, potentially affecting the range of homestays available for booking or limiting historical data storage for analytics.
- **Processing Power:** The server's processing power is crucial for handling complex operations such as search queries, booking transactions, payment processing, and data analysis. Insufficient processing power can result in slow performance and delays in processing user requests.
- **Scalability:** The website should be designed to scale horizontally or vertically to accommodate growing user demand. Hardware limitations can restrict scalability, making it challenging to expand server capacity or upgrade hardware components to meet increasing traffic demands.

c) Interfaces to other applications

Integration with external applications and APIs for functionalities such as mapping services, payment gateways, and email communication introduces constraints related to compatibility, reliability, and data exchange protocols.

d) Parallel operation

Requirements for parallel operation, such as simultaneous access by multiple users or processing multiple transactions concurrently, may impose constraints on system scalability, resource allocation, and performance optimization.

e) Audit functions

Requirements for auditing user activities, system changes, and transaction logs impose constraints on data logging, storage, and retrieval mechanisms to ensure compliance with audit trail requirements.

f) Control functions

Requirements for user access control, permissions management, and authorization mechanisms impose constraints on user authentication, session management, and role-based access control functionalities.

g) Higher-order language requirements

The use of React for frontend development, MongoDB for database interactions, and Node.js with Express for backend development imposes constraints on the selection of programming languages, frameworks, and libraries, requiring expertise in JavaScript and related technologies.

h) Signal handshake protocols

Requirements for signal handshake protocols, such as XON-XOFF or ACK-NACK, impose constraints on data transmission, error detection, and flow control mechanisms to ensure reliable communication between system components.

i) **Reliability requirements**

Requirements for system reliability, uptime, and fault tolerance impose constraints on system architecture, redundancy measures, and disaster recovery mechanisms to minimize service disruptions and data loss.

j) **Criticality of the application**

Requirements for the criticality of the application, such as mission-critical or non-critical, impose constraints on system design, testing, and deployment strategies to meet performance, availability, and security objectives.

k) **Safety and security considerations**

Requirements for safety and security, including data encryption, access control, and vulnerability assessments, impose constraints on system architecture, configuration, and operational practices to mitigate risks and protect against unauthorized access or data breaches.

2.4 Assumptions and Dependencies

- **Availability of third-party APIs:** The SRS assumes the availability and continued support of third-party APIs for functionalities such as payment processing, mapping services, and email communication. Any changes or discontinuation of these APIs may require adjustments to the software requirements.
- **Internet connectivity:** It is assumed that users will have reliable internet connectivity to access the online homestay booking system. Dependence on internet connectivity for system functionality implies that any disruptions or limitations in connectivity may affect the software requirements.
- **Scalability requirements:** It is assumed that the OHB system will need to accommodate a growing user base and increasing transaction volumes over time. Any changes in expected scalability requirements may influence the software requirements related to performance, capacity planning, and system architecture.
- **Availability of hosting infrastructure:** The SRS assumes the availability of hosting infrastructure for deploying and hosting the OHB system, including servers, storage, and network resources. Changes in hosting arrangements or infrastructure limitations may impact the software requirements, particularly in terms of deployment options and scalability.

2.5 User Classes and Characteristics

Guest: The person is a validated user of the system, with the intention of reserving accommodation from a homestay host through the platform. They interact with multiple features including signing up, viewing their

account, logging in, exploring available homestays, examining homestay details, making immediate bookings, checking booking specifics, proceeding to payment, inputting delivery details and payment method, completing the transaction, confirming bookings, reviewing previous reservations, providing feedback, canceling bookings, and logging out.

Host: The individual is an authenticated user of the service, aiming to promote homestays via the system. Hosts utilize a range of platform functions such as registering, accessing their account, logging in, uploading homestay information, editing homestay details, and monitoring reservations.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Figures below will present the hypothetical early phase user interfaces of the core functions of the website.

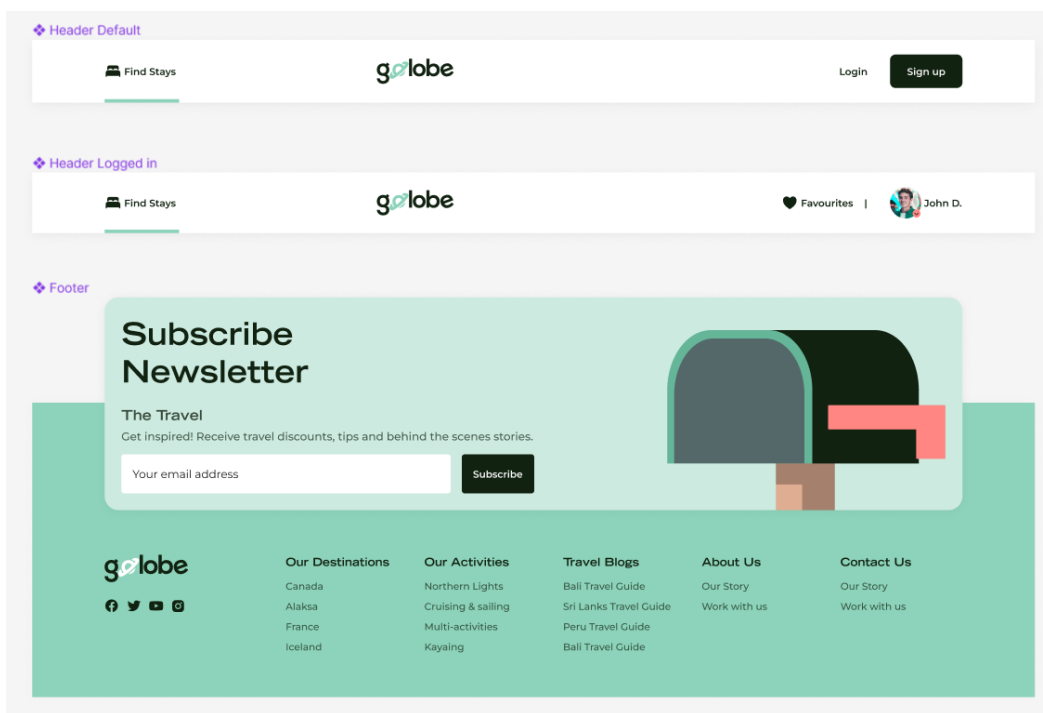



Figure 3.1: Header and footer



Login

Login to access your GoLobe account

john.doe@gmail.com

.....

☐ Remember me

[Forgot Password](#)

Login

Don't have an account? [Sign up](#)






Figure 3.2: Login





Sign up

Let's get you all st up so you can access your personal account.

john.doe@gmail.com

john.doe@gmail.com

john.doe@gmail.com

john.doe@gmail.com

.....

.....

☐ I agree to all the [Terms](#) and [Privacy Policies](#)

Create account

Already have an account? [Login](#)

Figure 3.3: Sign up

3.1.2 Hardware Interfaces

For stakeholders:

- Devices such as desktop computers, laptops, smartphones, tablets, etc.
- Printer
- Approximately 100MB of free hard drive space
- Minimum 128 MB RAM

For hosting:

- 20 – 30GB available disk space
- Memory minimum of 4GB RAM

3.1.3 Software Interfaces

- Payment gateways through Stripe's API
- Geolocation API
- Web Browsers (Chrome, FireFox, Safari, Microsoft Edge, etc)
- Database: MongoDB
- Operating System supporting the above browsers.

3.1.4 Communication Interfaces

The system will be available as a website and will be operational using standard web browsers (Safari, Google Chrome, Firefox, Microsoft Edge). Users will connect through a secured encrypted connection over internet.

3.2 System Features

This section of the document describes the functional requirements of the product by system features: the major services provided by the product. This section is written primarily for the developers and describes in technical terms the details of the functionality of the product.

3.2.1 Login

Description	Allows users to authenticate themselves to access the system.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. User navigates to the login page. 2. System displays the login form. 3. User enters their credentials and submits the form. 4. System validates the credentials. 5. If valid, system grants access; if invalid, system displays an error message.
Functional Requirements	<p>REQ_01: User shall only be able to log in with valid credentials.</p> <p>REQ_02: The system shall authenticate login credentials using hashed password encryption mechanisms.</p>

3.2.2 Register

Description	Allows users to create a new account on the system.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. User navigates to the registration page. 2. System displays the registration form. 3. User fills out the form with required information. 4. User submits the form. 5. System validates the information. 6. If valid, system creates a new user account; if invalid, system displays error messages.
Functional Requirements	<p>REQ_03: User shall provide necessary information for registration.</p> <p>REQ_04: The system shall validate the format of the provided email address.</p> <p>REQ_05: The system shall securely store the user's password using hashing techniques.</p>

3.2.3 Change password

Description	Allows users to change their password.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. User clicks on the "Change Password" link. 2. System displays the update password form. 3. User enters their new password and submits the form. 4. System verifies the valid password. 5. If valid, system changes the user password and notifies the users. If invalid, system displays error messages.
Functional Requirements	<p>REQ_06: User shall have the option to change their password for security problems.</p> <p>REQ_07: The system shall verify the valid password for changing password reset.</p> <p>REQ_08: The system shall send a notification to the user's email address.</p>

3.2.4 Edit profile

Description	Allows users to edit their profile information.
Priority	Medium
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. User navigates to the profile editing page. 2. System displays the current user information in editable fields. 3. User modifies the desired information. 4. User submits the changes. 5. System validates the changes and updates the user profile.
Functional Requirements	<p>REQ_9: User shall be able to edit their profile information.</p> <p>REQ_10: The system shall display the current user information for editing.</p> <p>REQ_11: The system shall validate the edited information before updating the user profile.</p>

3.2.5 Search rooms

Description	Allows guests to search for available homestay rooms based on specified criteria.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. Guest enters search criteria. 2. System retrieves matching rooms. 3. System displays search results to the user.
Functional Requirements	<p>REQ_12: Guest shall be able to specify search criteria such as destination, check-in/out dates, and number of guests.</p> <p>REQ_13: The system shall retrieve and display available rooms matching the search criteria.</p> <p>REQ_14: The system shall provide filters to refine search results (e.g., by price, amenities).</p> <p>REQ_15: Guest shall be able to view detailed information about each room in the search results.</p>

3.2.6 Book rooms

Description	Allows guests to book homestay rooms after selecting from available options.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. Guest selects a room for booking. 2. System displays the booking form. 3. Guest provides necessary booking information. 4. Guest confirms the booking. 5. System verifies the booking details and processes the payment. 6. System confirms the booking and sends a confirmation email to the user.
Functional Requirements	<p>REQ_16: Guest shall provide necessary booking information such as personal details and payment info.</p> <p>REQ_17: The system shall verify booking details before processing payment.</p> <p>REQ_18: The system shall handle payment securely and provide confirmation upon successful booking.</p> <p>REQ_19: The system shall update room availability status after successful booking.</p>

3.2.7 View my bookings

Description	Allows guests to view their homestay room reservation details.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. Guest selects the "View My Booking" option from the menu. 2. System retrieves the booking details associated with the guest's account. 3. System displays the booking details to the guest, including booking ID, room details, total cost, and booking status. 4. Guest reviews the booking details. If confirmed, system cancels the reservation and updates availability.

Functional Requirements	<p>REQ_20: The system shall provide a "View My Booking" option in the guest's menu.</p> <p>REQ_21: The system shall retrieve and display booking details associated with the user's account, including booking ID, room details (type, number of guests, check-in/out dates), total cost, and booking status.</p> <p>REQ_22: The system shall handle cases where the guest has no bookings by displaying a message indicating this.</p> <p>REQ_23: The system shall ensure that only logged-in guests can access the "View My Booking" feature.</p>
-------------------------	---

3.2.8 View guests' bookings

Description	Allows hosts to view bookings detail made by guest for each room
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. Host accesses the room management section. 2. System displays a list of all rooms. 3. Host selects a room to view bookings details. 4. System allows host to view booking details from guest.
Functional Requirements	<p>REQ_24: Host shall be able to view detailed information about each booking.</p> <p>REQ_25: The system shall retrieve and display booking details, including guest's name, room details (type, number of guests, check-in/out dates).</p> <p>REQ_26: The system shall handle cases where the provided booking ID or room ID is invalid or does not exist by displaying an appropriate error message.</p> <p>REQ_27: The system shall ensure that only authorized</p>

3.2.9 Add rooms

Description	Allows hosts to add new room, including edit details such as room type, availability, pricing, and amenities.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. Host accesses the room management section. 2. System displays the option to add a new room. 3. Host inputs new room details. 4. Host submits the new room information. 5. System validates the input and adds the new room to the system.
Functional Requirements	<p>REQ_28: Host shall be able to add new rooms with details such as type, availability, pricing, and amenities.</p> <p>REQ_29: The system shall display an option for adding a new room.</p> <p>REQ_30: Host shall provide necessary room information when adding a new room.</p> <p>REQ_31: The system shall validate the input when adding room information.</p> <p>REQ_32: The system shall update room information in real-time upon adding a new room.</p>

3.2.10 Remove rooms

Description	Allows hosts to remove existing rooms from the system.
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. Host accesses the room management section. 2. System displays a list of existing rooms. 3. Host selects a room to remove. 4. System prompts host for confirmation. 5. If confirmed, system removes the room from the system.
Functional Requirements	<p>REQ_33: Host shall be able to remove existing rooms from the system.</p> <p>REQ_34: The system shall display a list of existing rooms for removal.</p> <p>REQ_35: Host shall be able to select a room for removal.</p> <p>REQ_36: The system shall prompt host for confirmation before removing the room.</p> <p>REQ_37: If confirmed, the system shall remove the room from the system.</p>

3.2.11 Modify rooms detailed information

Description	Allows hosts to modify existing room information, including details such as room type, availability, pricing, and amenities, ...
Priority	High
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. Host accesses the room management section. 2. System displays a list of existing rooms. 3. Host selects a room to modify. 4. System displays the current room information in editable fields. 5. Host modifies the desired information. 6. Host submits the changes. 7. System validates the changes and updates the room information.
Functional Requirements	<p>REQ_38: Host shall be able to modify existing room information.</p> <p>REQ_39: The system shall display a list of existing rooms for modification.</p> <p>REQ_40: Host shall be able to select a room for modification.</p> <p>REQ_41: The system shall display the current room information for editing.</p> <p>REQ_42: Host shall be able to provide necessary room information when modifying a room.</p> <p>REQ_43: The system shall validate the input when modifying room information.</p> <p>REQ_44: The system shall update room information in real-time upon modifying a room.</p>

3.2.12 Log out

Description	Allows users to securely log out of their account.
Priority	Medium
Stimulus/Response Sequences	<ol style="list-style-type: none"> 1. User navigates to the log out option. 2. System ends the user session and clears authentication tokens.

	3. System redirects the user to the homepage or login page.
Functional Requirements	<p>REQ_45: User shall be able to log out of their account securely.</p> <p>REQ_46: The system shall end the user session upon log out.</p> <p>REQ_47: The system shall clear any authentication tokens associated with the user session.</p> <p>REQ_48: Upon successful log out, the system shall redirect the user to the homepage or login page.</p>

3.3 Use Case Model

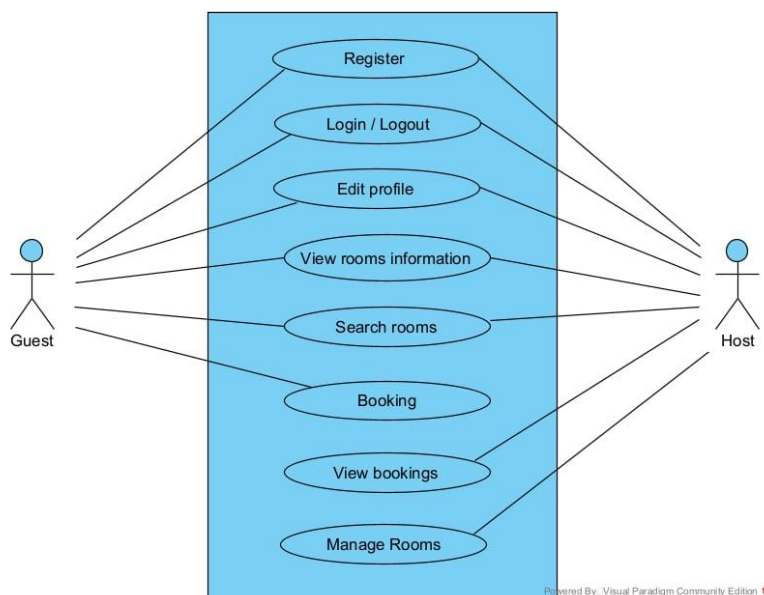
3.3.1 Actors

- Guest: who interacts with the system to search for available rooms, make reservations, view booking details.
- Host: who is responsible for managing the homestay's booking operations. They can add and modify rooms, handle bookings from guests.

3.3.2 Overview

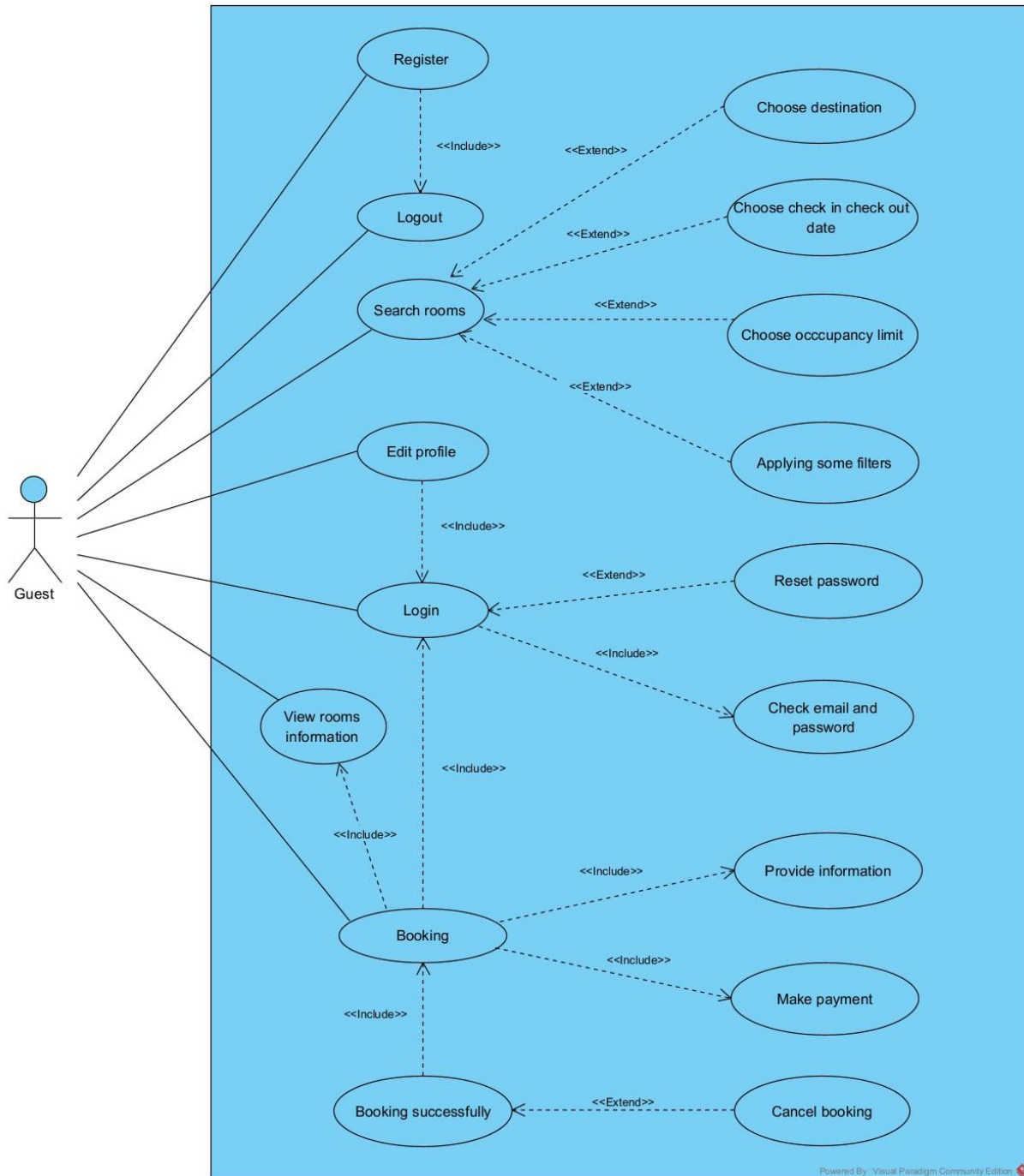
- Login: This function is intended to authenticate users when interacting with the system aims to provide rights and scope of system access.
- Registration: To access and use the system, users first need to register an account clause.
- Functional groups for managing, searching, and booking rooms.

3.3.3 General diagram

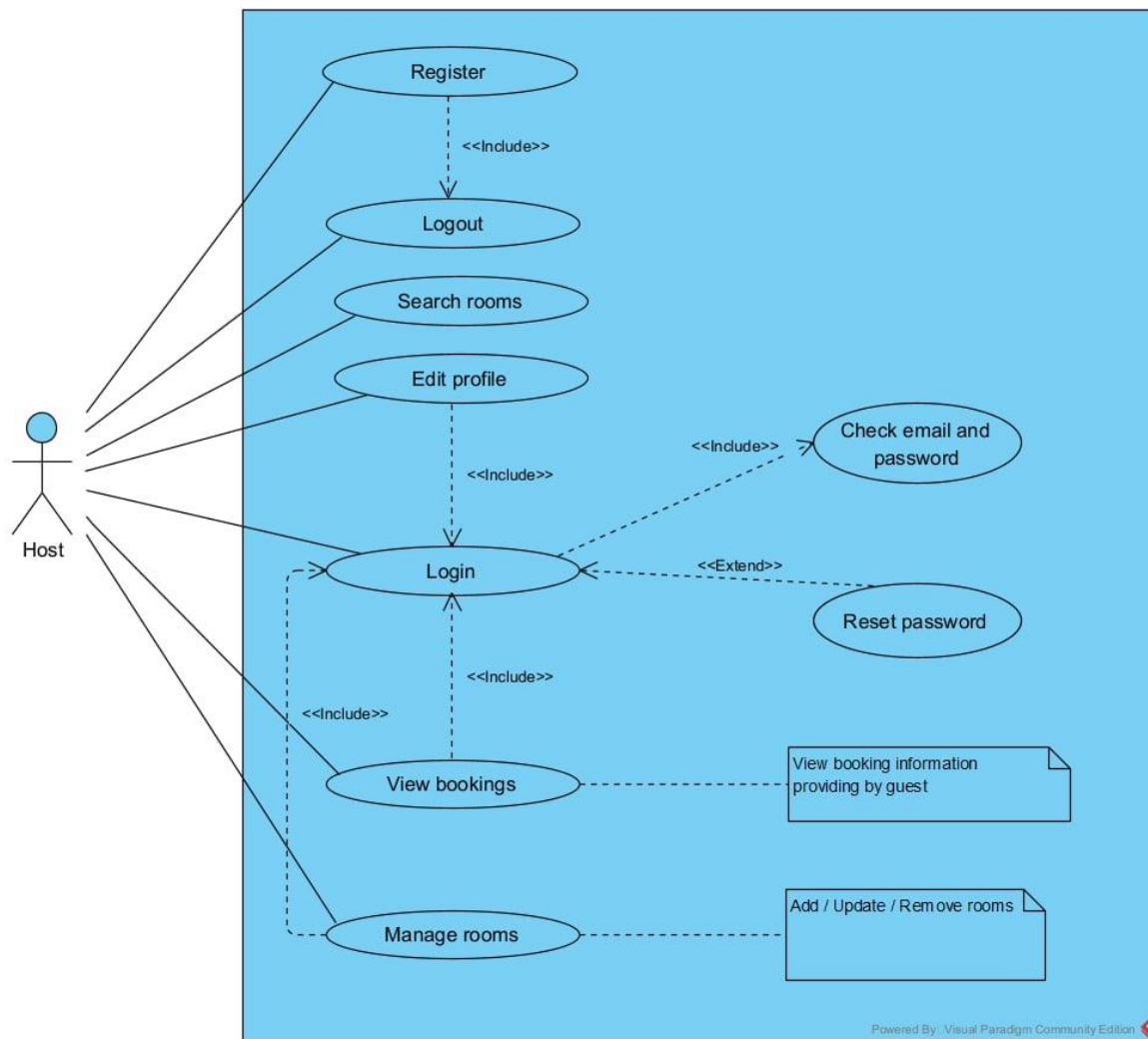


3.3.4 Detailed use case diagram

3.3.4.1 “Guest” use case diagram



3.3.4.2 “Host” use case diagram



3.3.5 Use case specification

3.3.5.1 Summary of use cases

ID	Name	Description	Actors	Trigger	Pre-condition	Post-condition
UC01	Login	Describes the process of a user logging into their account on the website	Guest, Host	User navigates to the login page	User has a registered account	User successfully logs into their account

UC02	Change password	Describes how a user can change their password	Guest, Host	User clicks on the "Change Password" in "Account" page	User is logged into their account	User successfully changes their password
UC03	Register	Describes the process of a guest registering for a new account on the website	Guest, Host	User navigates to the registration page	Users are logged out from their account	Guest successfully registers for an account and receives a confirmation email
UC04	Edit profile	Describes how a user can edit their profile information	Guest, Host	User navigates to the profile editing section	Users are logged into their account	User's profile information is successfully updated
UC05	Search room	Describes how a guest searches for available rooms based on specific criteria	Guest	Guest navigates to the search page	Guest is logged into their account	Guest views search results and available rooms
UC06	Book room	Describes how a user books a room on the website	Guest	Guest selects a room to book	Guest is logged into their account and the room is available	Room is successfully booked and guest receives confirmation
UC07	Manage room	Describes how a host manages rooms on the website, including add,	Host	Host navigates to the room management section	Host is logged into their account	Rooms information is successfully updated

		remove, or modify rooms.				
UC08	View guest's booking	Describes how a host views the booking details made by guests	Host	The host wants to view the booking details made by guests	Host is logged into their account	The host successfully views the booking details made by guests
UC09	View my booking	Describes how a guest views their booking details for the room	Guest	Guest wants to view their booking details	Guest is logged into their account	Guest successfully views their booking details

3.3.5.2 Flow of use cases

UC01:

- Basic Flow:
 1. User navigates to the login page.
 2. User fills their login form with email and password.
 3. User submits the login form.
 4. System verifies the provided credentials.
 5. System grants access to the user's account.
 6. Use case ends.
- Alternative Flow:
 1. Step 4a. If the provided credentials are incorrect, system displays an error message prompting the user to enter valid credentials.
- Exception Flow:
 1. Step 3a. If there are errors in the login form, system prompts the guest to correct the errors and resubmit the form.

UC02:

- Basic Flow:
 1. User navigates to the "Change Password" section in their account settings.
 2. System prompts the user to enter their current password and the new password.
 3. User enters their current password and the new password, then confirms the new password.

4. System verifies the entered passwords meet the password requirements (e.g., minimum length, special characters).
 5. System updates the user's password with the new one.
 6. System displays a message confirming that the password has been successfully changed.
 7. Use case ends.
- Alternative Flow:
 1. Step 4a. If the entered passwords do not meet the requirements, system displays an error message indicating the password requirements are not met, user re-enters the new password following the requirements, flow returns to step 3 of the basic flow.
 - Exception Flow: None

UC03:

- Basic Flow:
 1. User navigates to the registration page.
 2. User fills out the registration form with required details such as name, email, and password.
 3. User submits the registration form.
 4. System verifies the provided information.
 5. System creates a new user account.
 6. System sends a confirmation email to the registered email address.
 7. Use case ends.
- Alternative Flow:
 1. Step 4a. If the provided email is already registered, system prompts the user to log in or reset their password.
- Exception Flow:
 1. Step 5a. If there are errors in the registration form, system prompts the user to correct the errors and resubmit the form.

UC04:

- Basic Flow:
 1. User navigates to the profile editing section.
 2. User selects the option to edit their profile.
 3. User updates the desired profile information such as name, email, or password.
 4. User confirms the changes.

5. System verifies the updated information and applies the changes.
 6. User receives a confirmation message for the profile update.
 7. Use case ends.
- Alternative Flow:
 1. Step 3a. If the user chooses to change their password, system prompts for the current password before allowing the change.
 - Exception Flow:
 1. Step 5a. If there are errors in the updated information, system prompts the user to correct the errors and resubmit the form.

UC05:

- Basic Flow:
 1. Guest navigates to the search page.
 2. Guest selects destination, check-in and check-out dates, number of guests, and any additional filters.
 3. Guest submits the search criteria.
 4. System retrieves and displays available rooms matching the search criteria.
 5. Guest views detailed information about available rooms.
 6. Use case ends.
- Alternative Flow:
 1. Step 4a. If no rooms match the search criteria, system displays a message indicating no results found.
- Exception Flow:
 1. Step 3a. If there are errors in the search criteria, system prompts the guest to correct the errors and resubmit the form.

UC06:

- Basic Flow:
 1. Guest selects a room to book from the search results.
 2. Guest provides necessary booking information such as personal details, payment information, and any special requests.
 3. Guest confirms the booking.
 4. System verifies the booking details and processes the payment.
 5. System confirms the booking and sends a confirmation email to the user.
 6. Use case ends.

- Alternative Flow:
 1. Step 4a. If payment processing fails, system prompts the guest to try again or use a different payment method.
- Exception Flow:
 1. Step 2a. If there are errors in the booking form, system prompts the guest to correct the errors and resubmit the form.

UC07:

- Basic Flow:
 1. Host navigates to the room management section.
 2. Host selects an option to add, modify, or remove rooms, provides necessary room information such as room type, availability, pricing, and amenities.
 3. Host confirms the changes.
 4. System updates the room information accordingly.
 5. Use case ends.
- Alternative Flow:
 1. Step 3a. If the host chooses to remove a room, system prompts for confirmation before proceeding.
- Exception Flow:
 1. Step 2a. If there are errors in the room information form, system prompts the host to correct the errors and resubmit the form.

UC08:

- Basic Flow:
 1. The host selects the "View Guest Booking" option from the room management view.
 2. System retrieves the booking details associated with the provided booking ID and room ID.
 3. System displays the booking details to the host.
 4. Use case ends.
- Alternative Flow:
 1. Step 3a. If the booking ID or room ID is invalid or does not exist, system displays an error message indicating that the booking details could not be found, flow returns to step 1 of the basic flow.
- Exception Flow:

1. Step 2a. If the hotel manager cancels the operation: system cancels the view booking operation, use case ends without displaying any booking details.

UC09:

- Basic Flow:
 1. Guest selects the "View My Booking" option from the menu.
 2. System retrieves the booking details associated with the guest's account.
 3. System displays the booking details to the guest.
 4. Guest reviews the booking details.
 5. Use case ends.
- Alternative Flow:
 1. Step 3a. If the guest has no bookings, system displays a message indicating that the user has no bookings and use case ends.
- Exception Flow: None

4. Other Non-functional Requirements

4.1 Performance Requirements

- At the peak, system should be able to scale to 1,000+ users concurrently.
- The system data shall be backed up every night (full back-up) with a cycle of 30 days.
- The system should have a small turnaround time in displaying the user's search results.

4.2 Safety and Security Requirements

- The system should encrypt all user authentication data.
- The system should destroy all of the user's session data after logout.

4.3 Software Quality Attributes

Availability: The system should always be available for access by users. The website's uptime should always be greater than its downtime.

Interoperability: The system should be easy to integrate with other existing systems for the exchange of data. This should be considered during the design phase of the website.

Maintainability: The system should be easy to maintain. All the system documents should be easy to understand so that other teams can improve the website. The codebase for the system should follow the right programming standards and should also be thoroughly annotated to allow other developers to understand what it does.

Correctness: The system should be able to display the right information to the intended users. All system-computed statistics on the dashboard should be tested, to ensure that correct information regarding facilities is displayed and exported by end users.