
Software Requirements Specification

for

Hotel Booking Website Project

Version 1.0

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: 29/03/2004

CONTENTS

1. Introduction.....	1
1.1 Purpose	1
1.2 Scope	1
1.3 Intended Audience and Document Overview	1
1.4 Definitions, Acronyms and Abbreviations	1
1.5 Document Conventions	2
1.6 References	2
2. Overall Description	2
2.1 Product Perspective	2
2.2 Product Functions	3
2.3 Design and Implementation Constraints.....	3
2.4 Operating Environment	4
2.5 Assumptions and Dependencies	4
2.6 User Classes and Characteristics	5
3. Specific Requirements.....	5
3.1 External Interface Requirements	5
3.2 System Features	8
3.3 General use case diagram	8
3.4 Detailed use case diagram	9
3.5 Use case specification.....	10
4. Other Non-functional Requirements.....	16
4.1 Performance Requirements.....	16
4.2 Safety and Security Requirements.....	16
4.3 Software Quality Attributes.....	16

REVISIONS

Version	Primary Author(s)	Description of Version	Date Completed
#1	Team 6	Initial Specification	28/03/24

1. Introduction

1.1 Purpose

The purpose of this document is to describe an online hotel 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 Hotel Booking system (OHB). The OHB system will:

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

The OHB system will not:

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

With the application of the OHB system, clients can get real-time information on room availability, rates, and special offers; hotels 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 Hotel 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

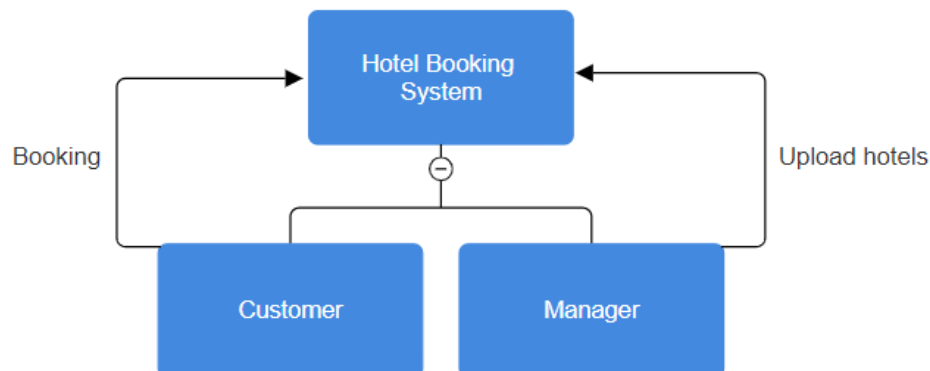
Institute of Electrical and Electronics Engineers (IEEE), Inc. (1998). IEEE Standard 830-1998, IEEE Recommended Practice for Software Requirements Specifications. New York, NY, USA: IEEE, Inc. Retrieved from: <http://www.math.uaa.alaska.edu/%7Eafkjm/cs401/IEEE830.pdf>

Software Requirements Specification template and example provided by teaching assistant.

2. Overall Description

2.1 Product Perspective

The Online Hotel 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

2.3 Design and Implementation Constraints

Constraints for the Online Hotel 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 hotels, bookings, user accounts, and other data. Limited storage space can restrict the amount of data that can be stored, potentially affecting the range of hotels 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 Operating Environment

2.5 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 hotel 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.

2.6 User Classes and Characteristics

Guest: The person is a validated user of the system, with the intention of reserving accommodation from a hotel manager through the platform. They interact with multiple features including signing up, viewing their account, logging in, exploring available hotels, examining hotel 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.

Manager: The individual is an authenticated user of the service, aiming to promote hotels via the system. Managers utilize a range of platform functions such as registering, accessing their account, logging in, uploading hotel information, editing hotel 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.

HOTEL BOOKING WEBSITE PROJECT

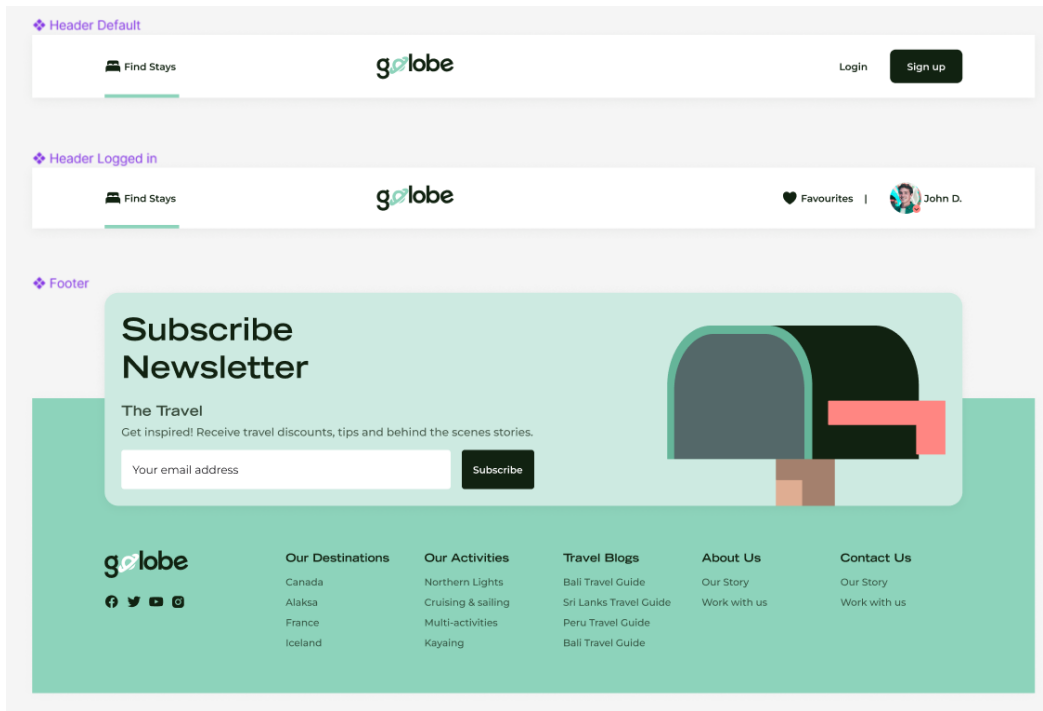


Figure 3.1: Header and footer

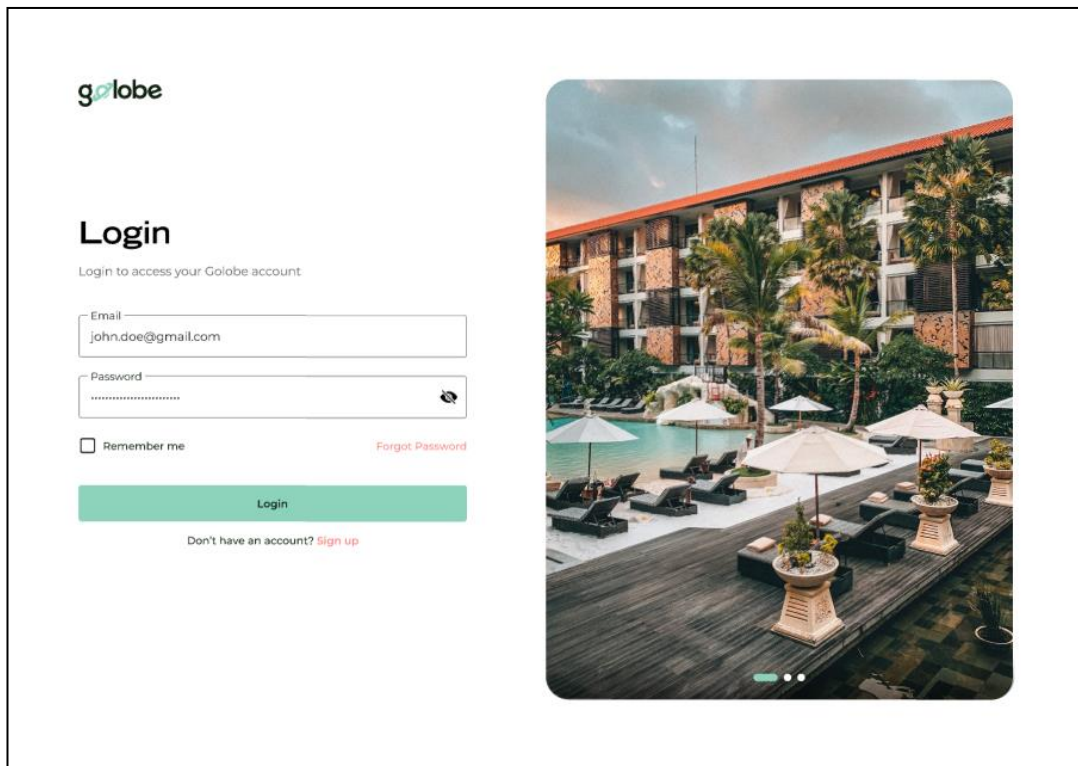
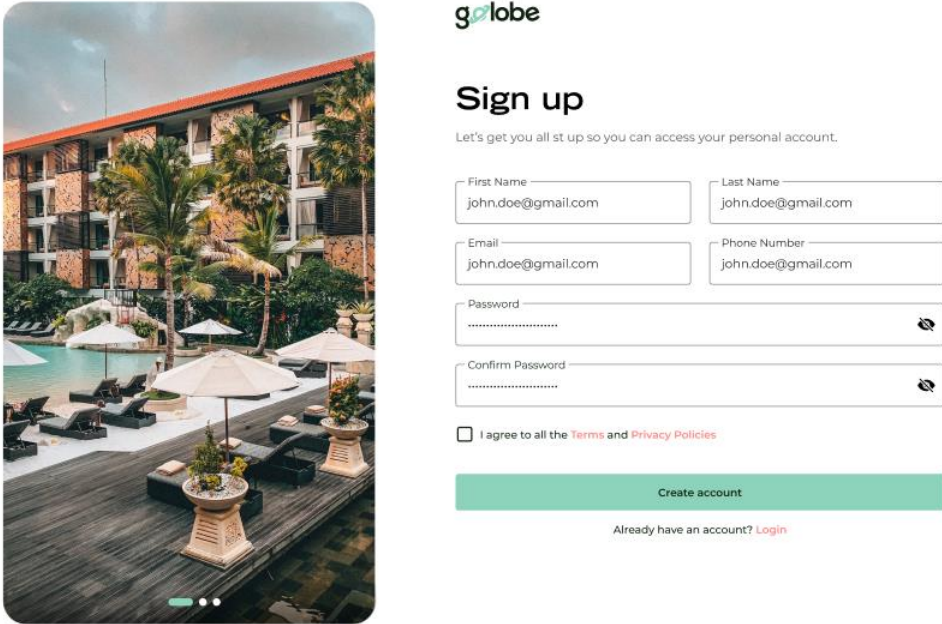


Figure 3.2: Login



golobe

Sign up

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

First Name
john.doe@gmail.com

Last Name
john.doe@gmail.com

Email
john.doe@gmail.com

Phone Number
john.doe@gmail.com

Password
.....

Confirm Password
.....

☐ 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.

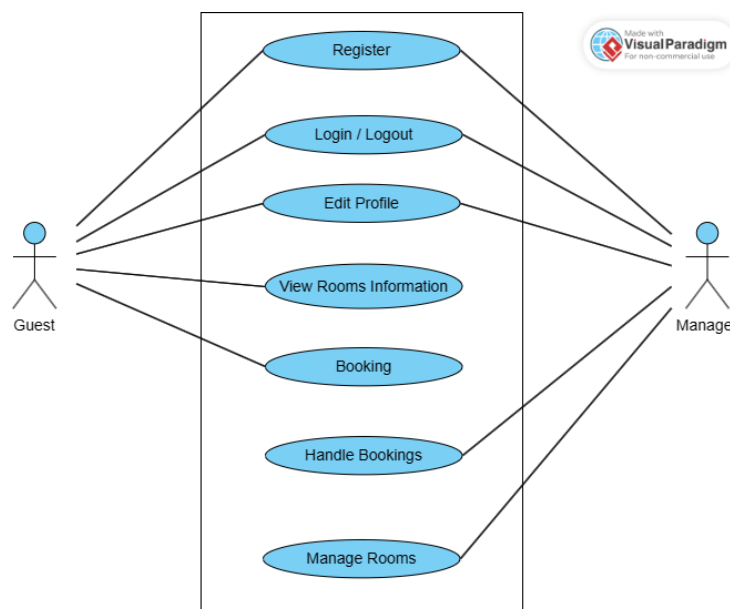
3.2.1 Actors

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

3.2.2 Overview

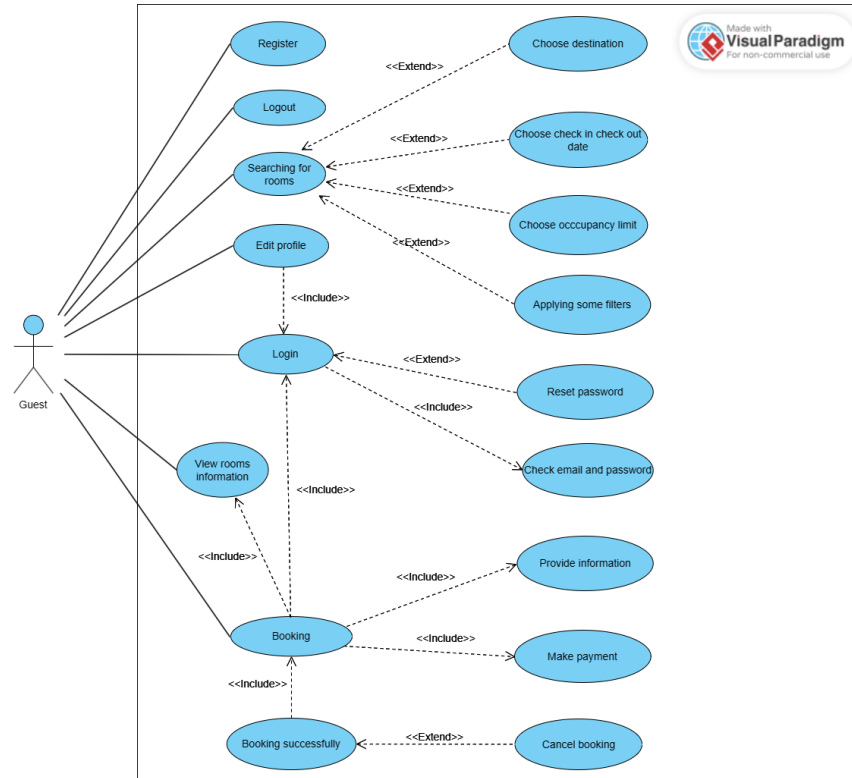
- Login: This function is intended to authenticate users when interacting with the system aims to provide rights and scope of system access.
- Registration: To access and use the system, users first need to register an account clause.
- Functional groups for managing, searching, and booking rooms.

3.3 General use case diagram

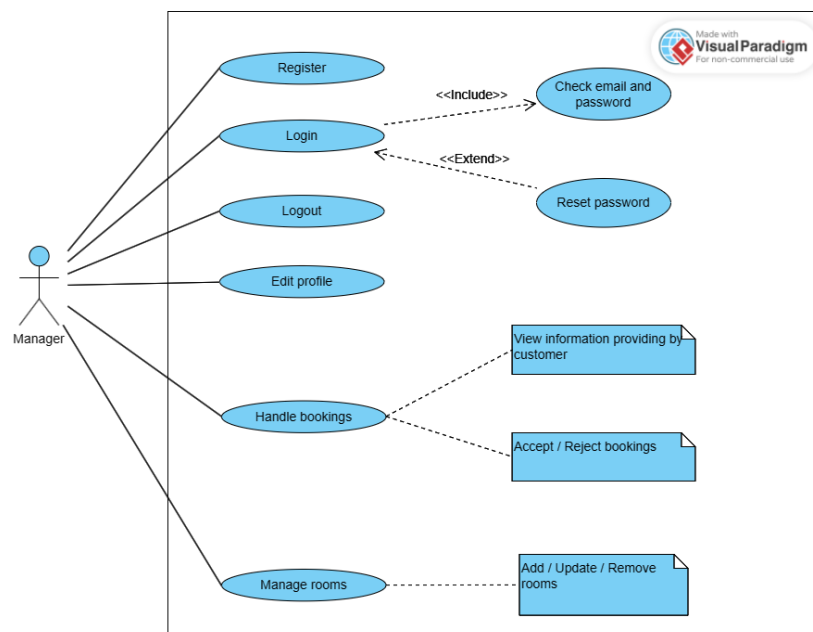


3.4 Detailed use case diagram

3.4.1 “Guest” use case diagram



3.4.2 “Manager” use case diagram



3.5 Use case specification

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, Manager	User navigates to the login page	User has a registered account	User successfully logs into their account
UC02	Reset password	Describes how a user can reset their password if they forget it	Guest, Manager	User clicks on the "Forgot Password?" link on the login page	User has a registered account	User's password is successfully reset, and they receive a confirmation email
UC03	Register	Describes the process of a guest registering for a new account on the website	Guest, Manager	User navigates to the registration page	None	Guest successfully registers for an account and receives a confirmation email
UC04	Edit profile	Describes how a user can edit their profile information	Guest, Manager	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	Booking room	Describes how a user books a room on the website	Guest	Guest selects a room to book	Guest is logged into their account	Room is successfully booked and guest receives confirmation
UC07	Manage room	Describes how a manager manages rooms on the website, including add, remove, or modify rooms.	Manager	Manager navigates to the room management section	Manager is logged into their account	Rooms information is successfully updated
UC08	Handle booking	Describes how a manager can accept or reject booking requests made by guests	Manager	Manager receives a new booking request from guest	Manager is logged into their account	Booking request is either accepted or rejected, and guest is notified accordingly
UC09	Cancel booking	Describes how a user cancels a previously room reservation	Guest	The guest decides to cancel a booked hotel room reservation	The guest is logged into their account and has at least reservations	Booked room is successfully canceled. The guest receives confirmation of the cancellation via email.

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 enters their registered email address.
 2. User submits the password reset request.
 3. System sends a password reset link to the user's email address.
 4. User clicks on the password reset link in the email.
 5. User enters a new password and confirms it.
 6. System verifies the new password and updates it.
 7. User receives a confirmation email for the password reset.
 8. Use case ends.
- Alternative Flow:
 1. Step 4a. If the provided email address is not found in the system, system displays an error message prompting the guest to enter a valid email address.
- Exception Flow:
 1. Step 7a. If the new password does not meet the system's password requirements, system prompts the user to choose a different password.

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. Manager navigates to the room management section.
 2. Manager selects an option to add, modify, or remove rooms, provides necessary room information such as room type, availability, pricing, and amenities.
 3. Manager confirms the changes.
 4. System updates the room information accordingly.
 5. Use case ends.
- Alternative Flow:
 1. Step 3a. If the manager 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 manager to correct the errors and resubmit the form.

UC08:

- Basic Flow:
 1. Manager receives a new booking request notification.
 2. Manager navigates to the bookings section of their account.
 3. Manager reviews the details of the booking request, including guest information and room availability.
 4. Manager selects the option to accept or reject the booking request.
 5. System updates the booking status accordingly.
 6. System sends a notification to the guest informing them of the booking status.
 7. Use case ends.
- Alternative Flow:
 1. Step 4a. If the manager decides to reject the booking, system prompts for a reason for rejection before proceeding.
- Exception Flow:
 1. Step 2a. If there are errors in the room information form, system prompts the manager to correct the errors and resubmit the form.

UC09:

- Basic Flow:
 1. User navigates to the bookings or reservations section of their account on the website.
 2. User selects the booked hotel room reservation they wish to cancel.
 3. User selects the option to cancel the booking.
 4. System prompts the user to confirm the cancellation.
 5. User confirms the cancellation.
 6. System updates the booking status to "canceled" and removes the reservation from the user's account.
 7. System sends a confirmation email to the user.
 8. Use case ends.
- Alternative Flow:

1. Step 4a. If the cancellation request is made within the cancellation window (e.g: 24 hours before check-in), system displays a message indicating the cancellation window has passed and user may not proceed with the cancellation.
- Exception Flow:
 1. Step 4c. If there is an error processing the cancellation request, system displays an error message indicating technical difficulties and user may try again later or contact customer support for assistance.

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.