UNIVERSITI KEBANGSAAN MALAYSIA
The National University of Malaysia

TTKK2023

Object-Oriented Software Engineering

Title:

Roomly

Lecturer's Name:

Group 4

Group Members:

NUR AINA SABIHAH BINTI ALIAS	A202607
NUR HAIFA BINTI ALIMAN	A202198
HUMAIRA' BINTI HAMIZAT	A202167
SITI NURKHALIDAH BINTI MOHD YUSOF	A202331
FARAH NABIHA BINTI ROSLEE	A202096

Table of Contents

CHAPTER 1 INTRODUCTION

CHAPTER 2 SOFTWARE REQUIREMENTS

CHAPTER 3 DESIGN AND IMPLEMENTATION

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In this day and age, the lodging and travel industries have been revolutionized by platforms that make it convenient for users to book accommodations. The way people discover and offer temporary stay has been completely transformed by websites like Airbnb, which connect hosts and guests. Despite having extensive features, Airbnb can become overwhelming for those users that are looking for a more straightforward, targeted platform for their most basic accommodation needs. We suggest creating a streamlined version of Airbnb to fill this gap, concentrating on essential features to appeal to people who want simple booking procedures free of needless complications.

The goal of this project is to design and create a useful software program that exemplifies object-oriented concepts. Through the program, hosts will be able to offer their lodgings, and guests will be able to quickly and easily browse and reserve these listings. The suggested solution guarantees scalability, maintainability, and user-friendliness by utilising object-oriented design and an intuitive interface.

1.2 PROBLEM STATEMENT

While Airbnb has made significant strides in providing a convenient and user-friendly platform for short-term rentals, certain limitations persist that affect the overall experience of both guests and hosts. The problem encountered by Airbnb is the **limited interaction between guests and hosts**. One of the core aspects of short-term rentals is the ability for guests to connect directly with hosts to ask questions about the property, its amenities or the surrounding area. However, communication tools on existing platforms are often restricted, leaving guests with unresolved queries or delayed responses. This limited interaction can deter guests from proceeding with bookings or lead to unmet expectations when they arrive at the property.

Apart from that is **lack of transparency in payment and fees**. Guests often discover additional fees for services like cleaning, taxes or platform charges only at the final stages of booking. This lack of upfront transparency can result in frustration, eroding trust between the platform and its users. Similarly, hosts may find themselves unprepared for unexpected deductions from their payouts due to fees they were unaware of. While **strict cancellation policies** are necessary to protect hosts, they can sometimes feel overly rigid for guests who face unforeseen circumstances. Guests who need to cancel within a short timeframe often forfeit the majority, if not all, of their booking fee. This lack of flexibility can discourage users from booking altogether, especially during uncertain times such as global disruptions or personal emergencies. By addressing these issues, Airbnb can create a more user-friendly, transparent, and reliable platform for both hosts and guests.

1.3 PROPOSED SOLUTION

The proposed software will feature a secure messaging system for hosts and guests, enabling seamless communication within the application. This system will facilitate real-time conversations to clarify details and enhance customer service. Key features include a direct messaging interface, structured communication options, the ability to handle special requests, and file-sharing capabilities. These features will allow hosts and guests to exchange messages with notifications for prompt responses and share images or documents, such as identity verification, when needed.

Additionally, the software will provide a transparent pricing breakdown designed to foster trust by offering a detailed analysis of costs before finalizing a booking. The pricing breakdown will include:

- **Detailed cost display**: Showcasing the base price, service fees, taxes, and optional add-ons.
- **Estimated total**: Highlighting the total amount payable by guests, clearly distinguishing between refundable and non-refundable components.

Lastly, the system will clearly outline cancellation policies and refund terms. These policies will be visible on the booking page, and an automated refund feature will calculate and process refunds based on the timing of the cancellation and the applicable terms. For host protection, the system will ensure that hosts receive a portion of the booking fee for last-minute cancellations to mitigate potential losses from missed opportunities.

CHAPTER 2

SOFTWARE REQUIREMENTS

2.1 INTRODUCTION

Software requirements are the descriptions of what the system should do; the services that it provides and the constraints on its operation. It serves as the foundation for any application development process, which outlines the features and limitations that make up the system. The software requirements for this project outline what the streamlined lodging platform needs to accomplish in order to successfully satisfy user expectations. We can make sure that the platform meets the fundamental demands of hosts and travellers while upholding the highest standards of usability, performance, and scalability by classifying requirements into functional, non-functional, and technical components. This methodical approach guarantees congruence between design goals and user requirements by offering a clear development roadmap.

There are 3 types of requirements, which are functional requirements, quality requirements and constraints.

2.2 FUNCTIONAL REQUIREMENTS

2.2.1 USER REQUIREMENTS

1. Guest

The system shall allow guests to sign up, log in, and log out.

The system shall enable guests to search for available properties.

The system shall allow guests to view property details, including descriptions, images, and prices.

The system shall enable guests to book rental properties.

The system shall support guests in making payments for bookings.

The system should allow guests to leave reviews and ratings after a stay.

The system should enable guests to communicate with property hosts via messaging.

2. Host

The system shall allow hosts to sign up, log in, and log out.

The system shall enable hosts to add, update, and remove rental listings.

The system shall allow hosts to accept, reject, or modify booking requests.

The system shall provide detailed information about upcoming and previous bookings for hosts.

The system shall display summaries of earnings from completed bookings to hosts.

The system should enable hosts to view and respond to messages.

The system shall allow hosts to set property availability for specific dates.

2.2.2 System Requirements

1. Account Management

The system shall support user authentication, including sign-up, login, and logout functionality for both guests and hosts.

The system shall validate user credentials during login.

The system shall allow users to reset passwords securely.

2. Property Listings

The system shall allow hosts to create, update, and delete property listings.

The system shall enable guests to search for properties based on filters such as location, price, and availability.

The system shall display property details, including descriptions, images, prices, and availability.

3. Booking Management

The system shall enable guests to make bookings for available properties.

The system shall allow hosts to accept, reject, or modify booking requests.

The system should provide notifications for booking status updates.

4. Payment Processing

The system shall support secure payment transactions for bookings.

The system should generate receipts for completed transactions.

5. Ratings and Reviews

The system should allow guests to submit ratings and reviews after completing a stay.

The system shall display average ratings and reviews for each property.

6. Messaging System

The system shall allow guests and hosts to exchange messages regarding bookings.

The system shall provide notifications for new messages.

2.2.3 Non-Functional Requirements

1. Performance

The system shall handle up to 1000 concurrent users without performance degradation.

The system shall respond to user requests within 3 seconds under normal load.

2. Security

The system shall encrypt sensitive data, including passwords and payment information.

The system shall implement multi-factor authentication for enhanced security.

3. Scalability

The system shall support scalability to accommodate future growth in the number of users and properties listed.

4. Availability

The system shall maintain 99.9% uptime to ensure availability for users.

5. Usability

The system shall provide a user-friendly interface optimized for both desktop and mobile devices.

2.3 QUALITY REQUIREMENTS

2.3.1 Functionality

The system shall accurately process user registration and login attempts within 10 seconds.

The system shall correctly display property listings based on search criteria within 5 seconds.

The system shall enable secure and efficient booking and payment processing, with transactions completed within 2 minutes.

The system shall facilitate clear and timely communication between hosts and guests, with response times to messages not exceeding 24 hours.

The system shall accurately calculate and process refunds within 48 hours of a cancellation request.

The user shall be able to complete the registration process within 5 minutes.

The user shall be able to search and book a property within 10 minutes.

2.3.2 Reliability

The system shall have a minimum uptime of 99.9% during peak hours (6 AM to midnight).

The system shall recover from system failures within 15 minutes.

The system shall maintain data integrity and security, with no data loss or breaches.

2.3.3 Usability

The system shall have a user-friendly interface with an average task completion time of 5 minutes for experienced users.

The system shall provide clear and concise instructions and error messages.

The user shall be able to easily navigate the website and app.

2.3.4 Performance

The system shall respond to user requests within 2 seconds on average.

The system shall handle peak loads of 10,000 concurrent users without significant performance degradation.

The user shall experience fast load times and minimal delays.

2.3.5 Security

The system shall protect user data and sensitive information using strong encryption techniques.

The system shall implement robust authentication and authorization mechanisms.

The system shall be regularly scanned for vulnerabilities and updated with security patches within 48 hours of a vulnerability report.

2.3.6 Maintainability

The system shall be well-documented with clear and concise documentation.

The system shall be modular and maintainable, with changes to one module having minimal impact on other modules.

The system shall adhere to industry-standard coding practices and guidelines.

2.3.7 Portability

The user shall be able to use this application on multiple platforms (e.g., web, mobile) without significant modifications.

2.3.8 Scalability

The system shall be able to handle increased user load and data volume without compromising performance.

2.3.9 Interoperability

- The system shall integrate seamlessly with third-party services (e.g., payment gateways, mapping services).

2.4 CONSTRAINTS

User Constraints

1. Users must complete the login process using verified emails to enhance security and prevent fraudulent bookings.
2. Users must complete bookings within a day session to prevent reservation conflicts. Incomplete booking will be automatically canceled.

System Constraints

1. The system must undergo maintenance every 6 months, with prior user notification at least 48 hours in advance.
2. The system must log errors and user activities with timestamps for troubleshooting and operational transparency.
3. During maintenance, it may involve temporary downtime during which bookings and payment may be unavailable.

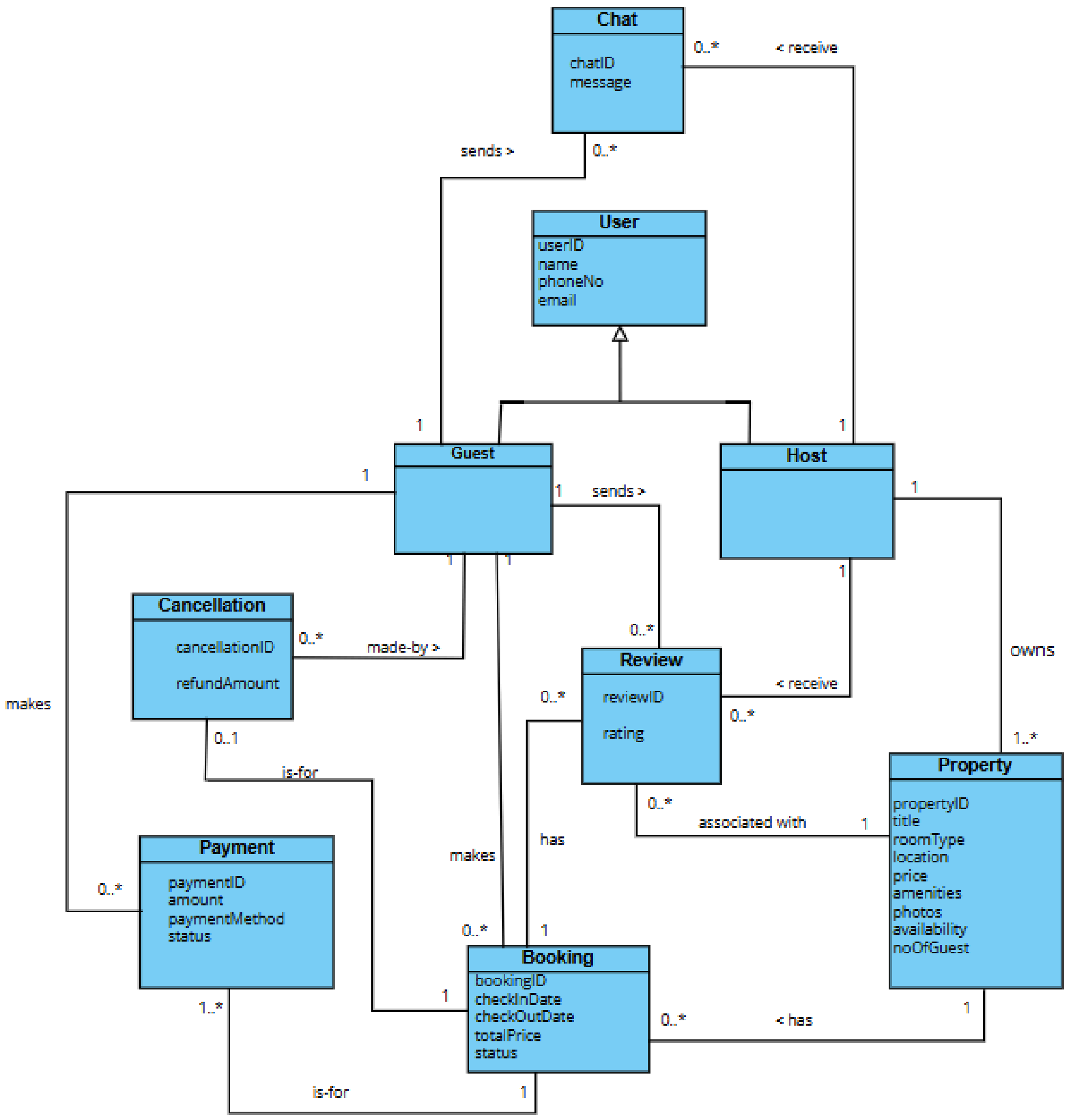
Third-Party Constraints

1. The system must integrate with a third-party payment gateway for transactions processing, using encryption protocols (SSL/TLS).
2. The system must support multiple payment methods, including credit/debit cards, e-wallet and bank transfers.

Flexibility Constraints

1. Cancellation policies must allow refunds only if cancellations occur at least 2 days before the actual check-in date.
2. Refunds will be processed within 5-7 business days.
3. Exceptions for refund may only be applied in case of medical emergencies or system-related issues.

2.5 DOMAIN MODEL



User represent individuals who interact with the system and have attributes of userID, name, phoneNo and email. Guest is a subclass of User, which represents user that make bookings. Host is also a subclass of User, which represents user that host and manage property. Property represents accommodation units, such as rooms or apartments and has attributes such as propertyID, title, roomType, location, price, amenities, photos, availability and noOfGuest. Booking represents reservations made by guests and has attributes of bookingID, checkInDate, checkOutDate, totalPrice and status. Payment represent financial transactions related to bookings and the attributes are paymentID, amount, paymentMethod and status.

Cancellation represents cancellation of bookings and has attributes of cancellationID and refundAmount. Review represents feedback provided by guests about their stays and the attributes are reviewID and rating. Chat represents communication between guests and hosts and the attributes includes chatID and message.

Guests can send zero more reviews and Review is sent by exactly one Guest. Guest makes zero or many Payment, and Payment is sent by exactly one Guest. Guest send zero or more Chat, and Chat sent by exactly one Guest. Guest make zero or more Booking, and Booking is made by exactly one Guest.

Host owns one or more Property, and Property is owned by exactly one Host. Host receives zero or more Review, and Review received by exactly one Host. Host receives zero or more Chat, and Chat is received by exactly one Host.

Cancellation is made by exactly one Guest, and Guest can make zero or more Cancellation. Cancellation is for exactly one Booking, and Booking has zero or one Cancellation.

Payment is for exactly one Booking, and Booking has one or more Payment.

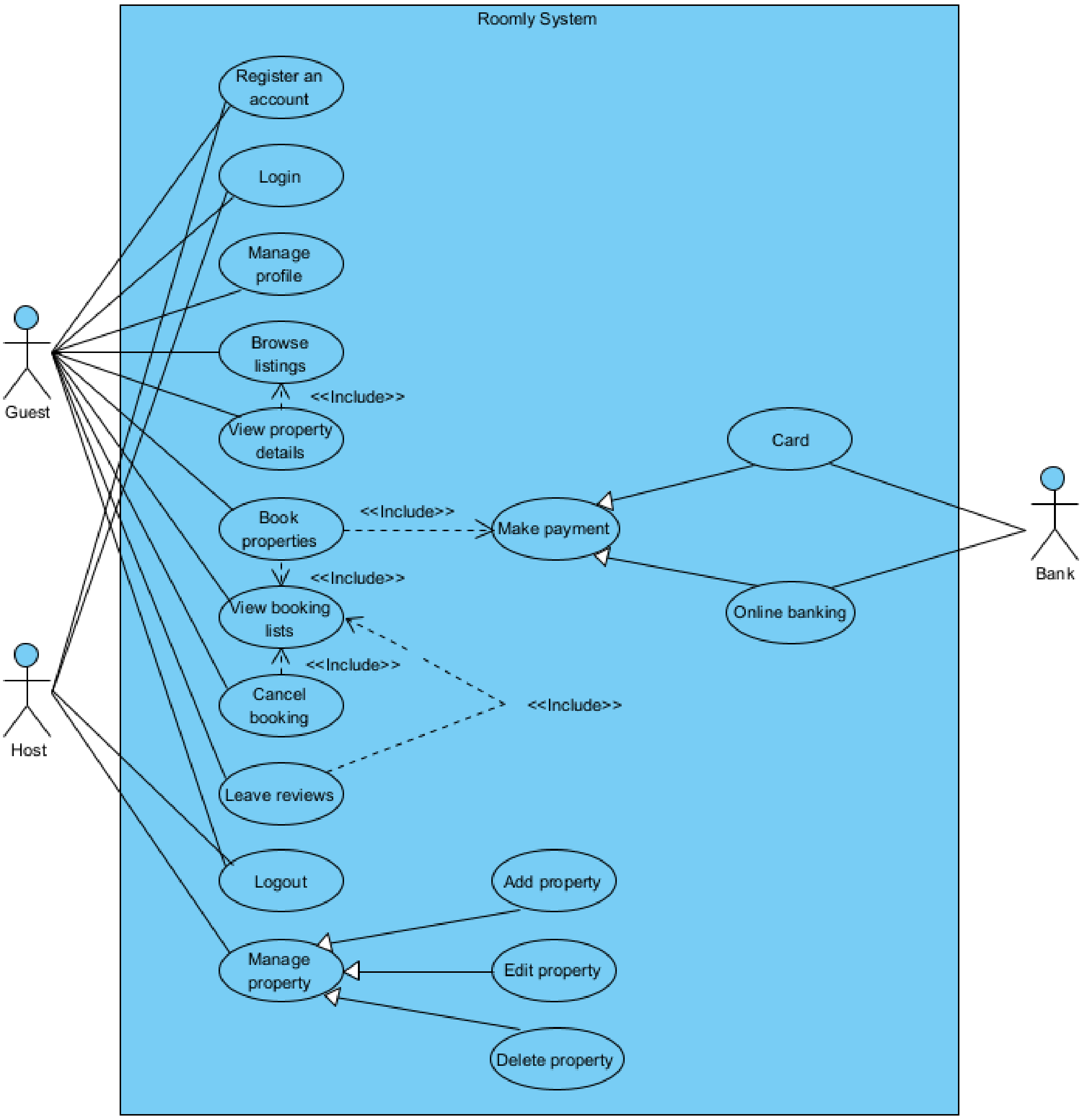
Booking has zero or more Review, and Review has exactly one Booking.

Review is associated with exactly one Property, and Property has zero or more Review.

Property has zero or more Booking, and Booking is for exactly one Booking.

2.6 USE CASE

2.6.1 USE CASE DIAGRAM



The use case diagram represents the functionalities of the Roomly System, which enables **guests** to browse and book properties and allows **hosts** to manage their property listings. It also incorporates the interaction with the **bank** for payment processing.

The diagram helps in identifying the key actors, their interactions with the system, and the major functionalities provided by the Roomly System.

2.6.2 USE CASE SPECIFICATIONS

ID:	UC-1	
Title:	Register an account	
Description:	This use case describes the process where a user creates a new account in the Roomly system to gain access to its features.	
Primary Actor:	Guests, Host	
Precondition:	-	
Postcondition:	The guest and host account is created and ready for login.	
Main Success Scenario:	Actor	System
	1. Guest and Host selects "Register an account."	
		2. System prompts the guest and host to input their details.
	3. Guest and user enter details.	
	4. Guest and user submits the registration form.	
		5. The system validates the input and creates the account.
		6. System sends a confirmation email to the guest and host.
Alternative Scenarios:	Actor	System
	3a. Guest and host provide invalid or incomplete input in the form —3a1. System prompts for correction of input.	2a. The email is registered —2a1. System displays an error message and asks the guest and host to refill the form.

ID:	UC-2	
Title:	Login	
Description:	This use case allows users to log into the Roomly System.	
Primary Actor:	Guest, Host	
Precondition:	Guest and Host must be logged into their account.	
Postcondition:	Guest and Host are authenticated and granted access to their account.	
Main Success Scenario:	Actor	System
	1. The guest and host selects “Login”. 3. Guest and Host enters their credentials	2. The system prompts for Guest and Host to enter an email and password. 4. The system validates the credentials. 5. If valid, the Guest and Host are granted their access into their Roomly account.
Alternative Scenarios:	Actor	System
		4a. Credentials are incorrect. —4a1. System displays an error message. 4b. Account is locked —4b1. System notifies the Guest and Host.

ID:	UC-3	
Title:	Manage profile	
Description:	Allows Guest and Host to update their account details like name, email, password and profile picture.	
Primary Actor:	Guest, Host	
Precondition:	Guests and hosts must be logged into the system.	
Postcondition:	Guest and host profile information are updated successfully.	
Main Success Scenario:	Actor	System
	1. The Guest and Host selects “Manage profile”. 3. The Guest and Host updates the desired fields.	2. The system displays the current profile details. 4. The system updates and saves the new information.
Alternative Scenarios:	Actor	System
		4a. The inputs are invalid —4a1. System prompts for corrections of information.

ID:	UC-4	
Title:	Browse listings	
Description:	This use case allows a guest to view available properties for booking.	
Primary Actor:	Guest	
Precondition:	-	
Postcondition:	The Guest and Host view the list of properties for booking.	
Main Success Scenario:	Actor	System
	1. The Guest and Host selects “Browse listings”. 3. The Guest and Host can filter or sort the listings.	2. The system displays a list of properties.
Alternative Scenarios:	Actor	System

ID:	UC-5	
Title:	View property details	
Description:	This use case allows a guest to view detailed information about a chosen property.	
Primary Actor:	Guest	
Precondition:	The property must exist in the system.	
Postcondition:	The guest views the property details.	
Main Success Scenario:	Actor	System
	1. The guest selects a property from the listings.	2. The system displays property details like price, amenities, and reviews
Alternative Scenarios:	Actor	System

ID:	UC-6	
Title:	Book properties	
Description:	This use case allows guests to book a property.	
Primary Actor:	Guest	
Precondition:	The guest must be logged in and the chosen property is available.	
Postcondition:	The property is booked.	
Main Success Scenario:	Actor	System
	1. The Guest selects “Book properties” after viewing property details. 3. The guest completes the payment process.	2. The system prompts for payment. 4. The booking is confirmed.
Alternative Scenarios:	Actor	System
		3a. Payment fails —3a1. System notifies the guest.

ID:	UC-7	
Title:	Make payment	
Description:	Guests complete payment after making a property booking.	
Primary Actor:	Guest	
Precondition:	The guest must have selected a property to book.	
Postcondition:	The payment is completed, and the booking is confirmed.	
Main Success Scenario:	Actor	System
	2. Guest selects a payment method and provides necessary details.	1. System displays options for payment. Card or online banking? 3. System processes the payment through the bank. 4. System confirms the payment.
Alternative Scenarios:	Actor	System
		3a. Payment fails. —3a1. System notifies the guest.

ID:	UC-8	
Title:	View booking lists	
Description:	Guests view their past and current bookings.	
Primary Actor:	Guest	
Precondition:	Guests must be logged into the system.	
Postcondition:	Guests are able to view their booking history.	
Main Success Scenario:	Actor	System
	1. The Guest selects “View booking list”.	2. System displays the list of bookings.
Alternative Scenarios:	Actor	System

ID:	UC-9	
Title:	Cancel booking	
Description:	Allow guests to cancel a booking.	
Primary Actor:	Guest	
Precondition:	Guests must be logged into the system and have an active booking. The cancellation policy must allow cancellation.	
Postcondition:	Booking is canceled.	
Main Success Scenario:	Actor	System
	1. The guest selects a booking to cancel.	2. The system confirms the cancellation. 3. The system processes any refunds if applicable.
Alternative Scenarios:	Actor	System
		2a. Cancellation is not allowed. —2a1. The system notifies the guest.

ID:	UC-10	
Title:	Leave reviews	
Description:	Allow guests to leave a review for a property after their stay.	
Primary Actor:	Guest	
Precondition:	The guest must be logged into the system and has completed a stay at a property.	
Postcondition:	The review is added to the property details.	
Main Success Scenario:	Actor	System
	1. The Guest selects “Leave reviews”. 3. The Guest submits the review.	2. The system displays a form for the review.

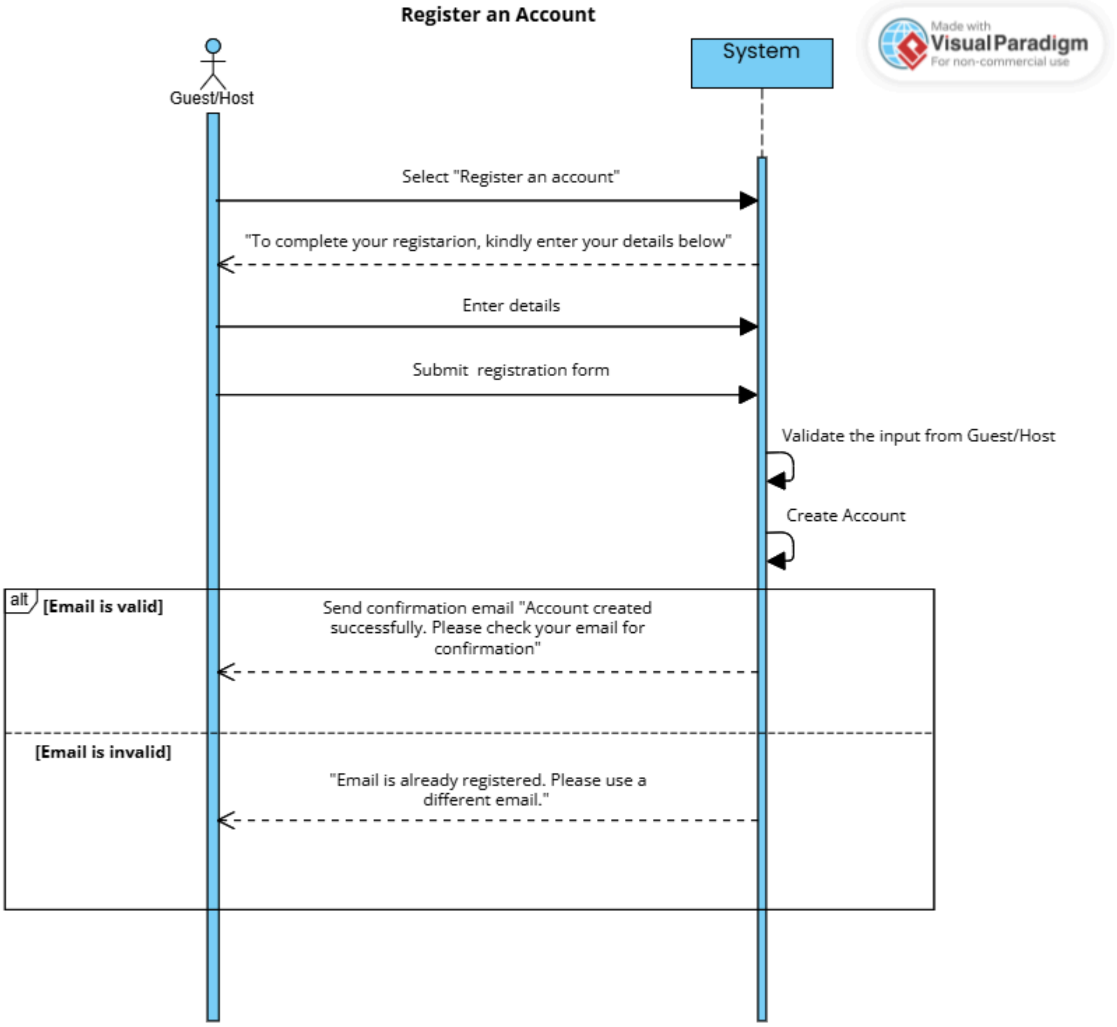
Alternative Scenarios:	Actor	System

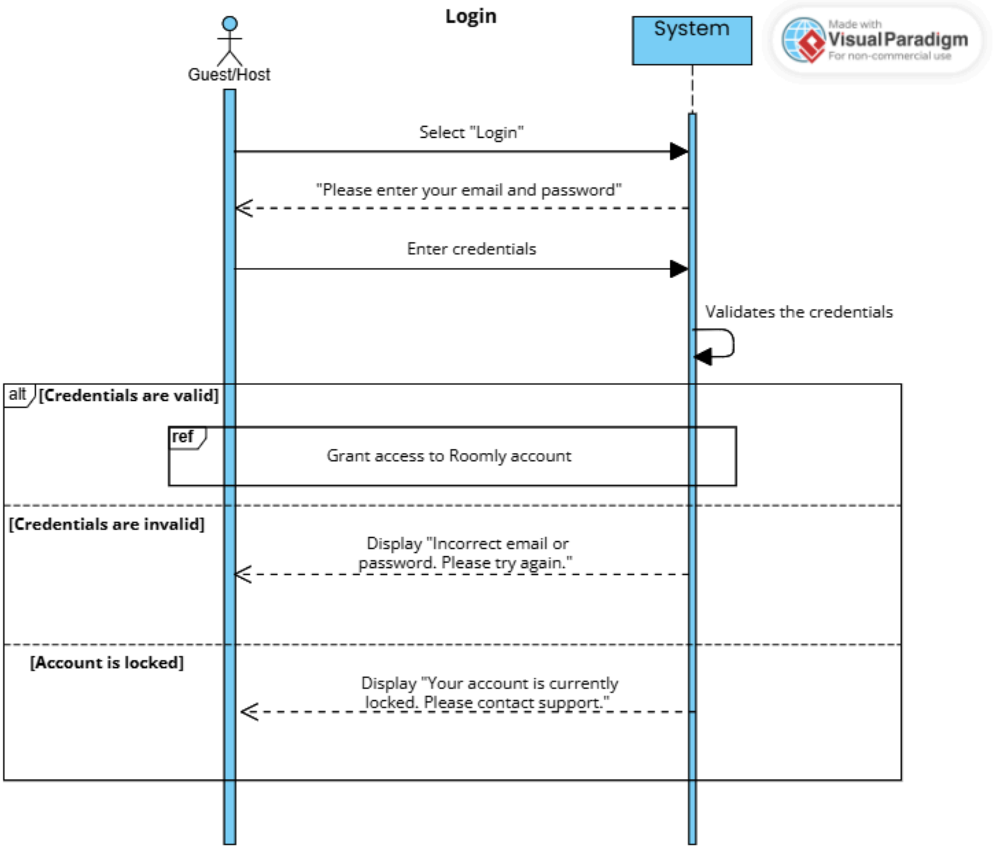
ID:	UC-11	
Title:	Logout	
Description:	Enables guests and hosts to log out of the system to end their session securely.	
Primary Actor:	Guest, Host	
Precondition:	Guest and Host are logged into the system.	
Postcondition:	The guest and host are logged out and the session is terminated.	
Main Success Scenario:	Actor	System
	1. Guest and Host selects "Logout".	2. System terminates the session and redirects the Guest and Host back to the login page.
Alternative Scenarios:	Actor	System
		2a. The session has already expired. —2a1. System displays a message informing the guest and user that the session has timed out.

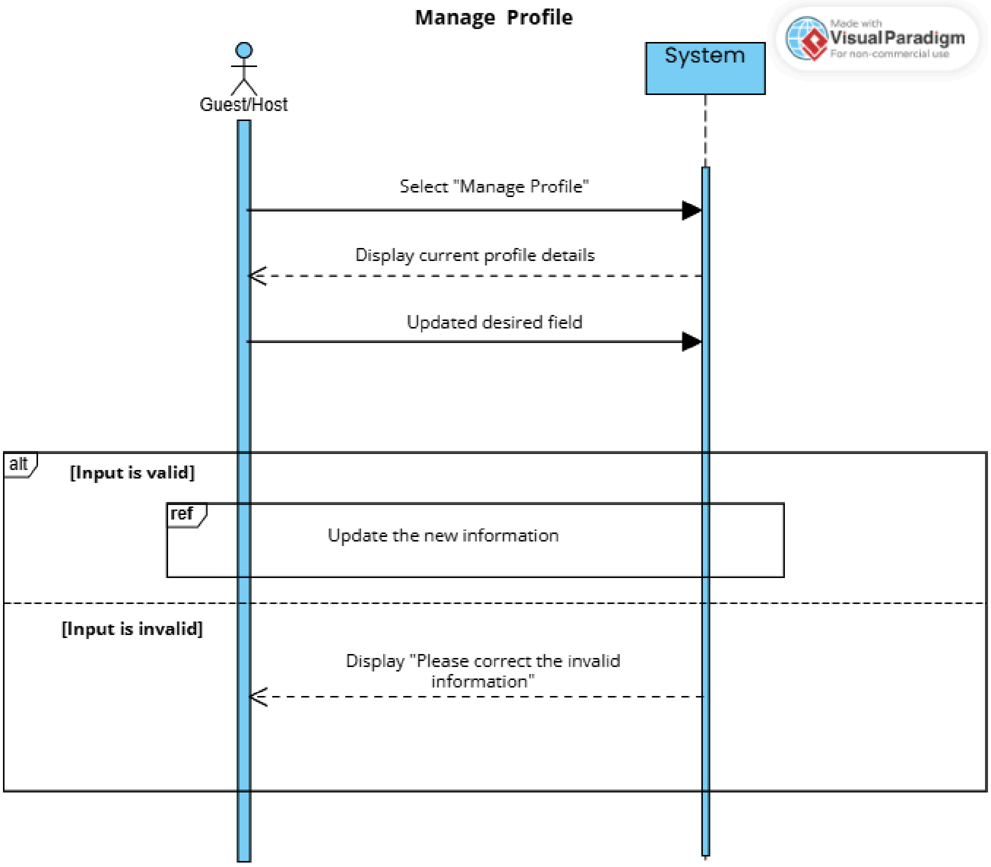
ID:	UC-12	
Title:	Manage property	
Description:	This use case allows a host to manage their property listings by adding, editing, or deleting the property listings.	
Primary Actor:	Host	
Precondition:	Host must be logged in.	
Postcondition:	The property information is successfully updated in the system.	
Main Success Scenario:	Actor	System
	1. The host selects "Manage Property".	

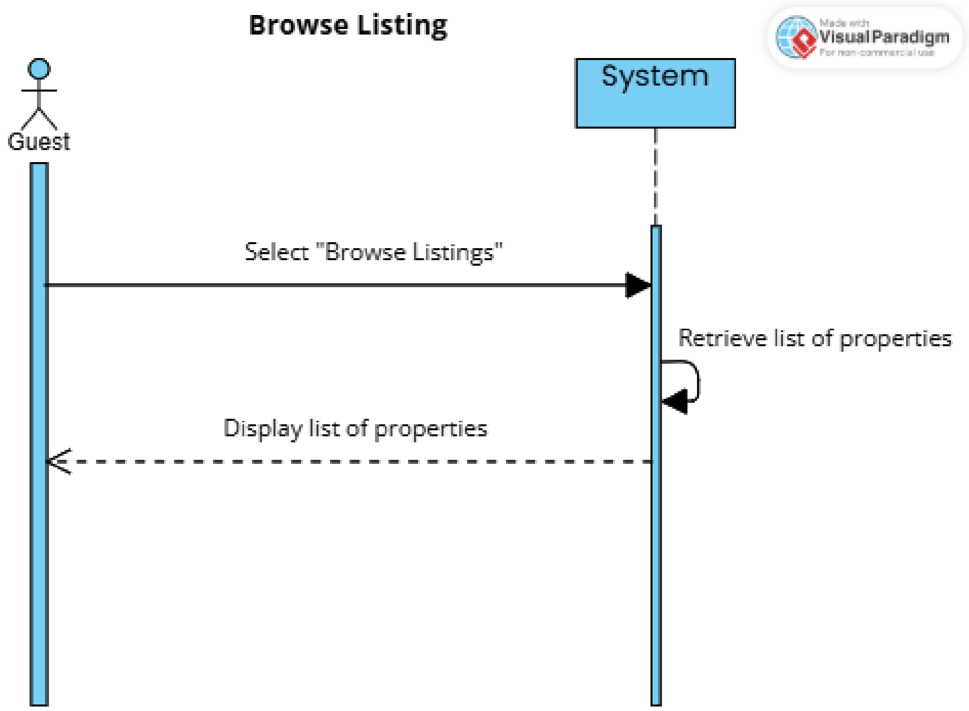
		2. System displays options: Add Property, Edit Property, Delete Property.
Alternative Scenarios:	Actor	System
	2a. Host selects “Add Property”. —2a1. Host provides property details.	—2a2. System validates the details and saves the new added property listings. —2a3. System confirms the property addition to the host.

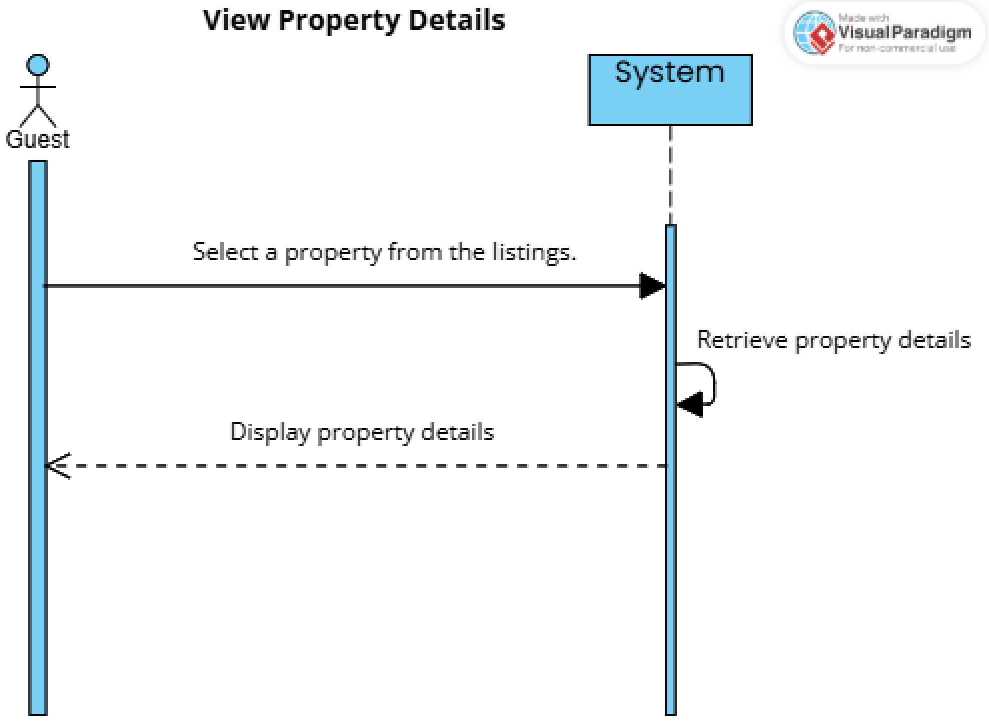
2.6.3 SYSTEM SEQUENCE DIAGRAMS


Register an Account
Guest/Host
System
Made with Visual Paradigm For non-commercial use
Select "Register an account"
"To complete your registarion, kindly enter your details below"
Enter details
Submit registration form
Validate the input from Guest/Host
Create Account
alt
[Email is valid]
Send confirmation email "Account created successfully. Please check your email for confirmation"
[Email is invalid]
"Email is already registered. Please use a different email."

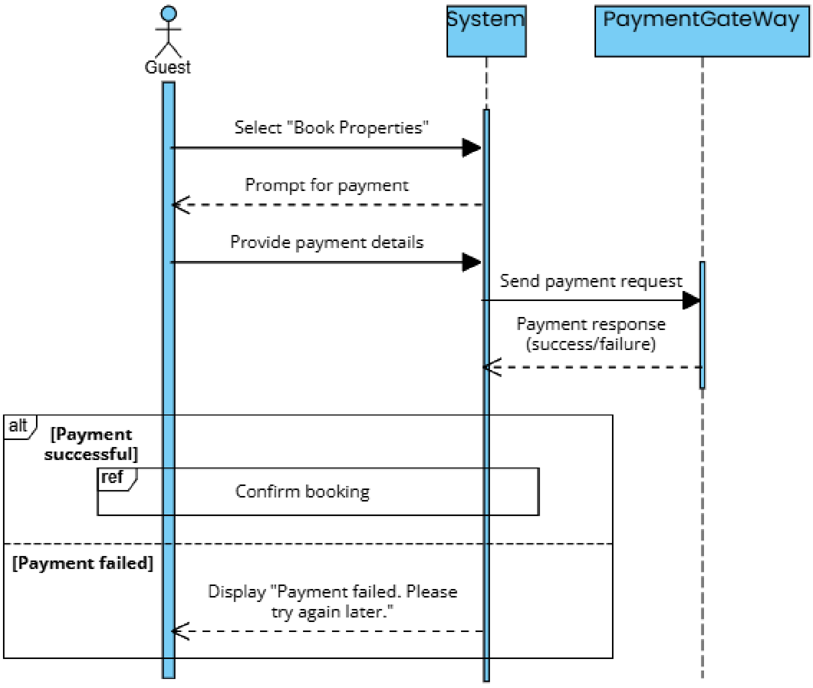

Login
Guest/Host
System
Made with Visual Paradigm For non-commercial use
Select "Login"
"Please enter your email and password"
Enter credentials
Validates the credentials
alt
[Credentials are valid]
ref
Grant access to Roomly account
[Credentials are invalid]
Display "Incorrect email or password. Please try again."
[Account is locked]
Display "Your account is currently locked. Please contact support."


Manage Profile
Guest/Host
System
Made with Visual Paradigm For non-commercial use
Select "Manage Profile"
Display current profile details
Updated desired field
alt
[Input is valid]
ref
Update the new information
[Input is invalid]
Display "Please correct the invalid information"

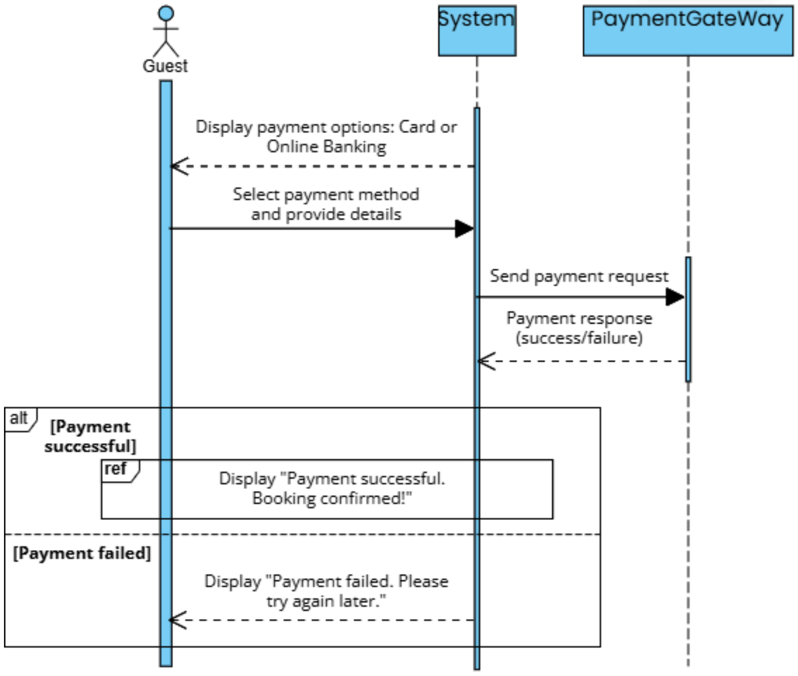

Browse Listing
Guest
System
Select "Browse Listings"
Retrieve list of properties
Display list of properties


View Property Details
Guest
System
Select a property from the listings.
Retrieve property details
Display property details

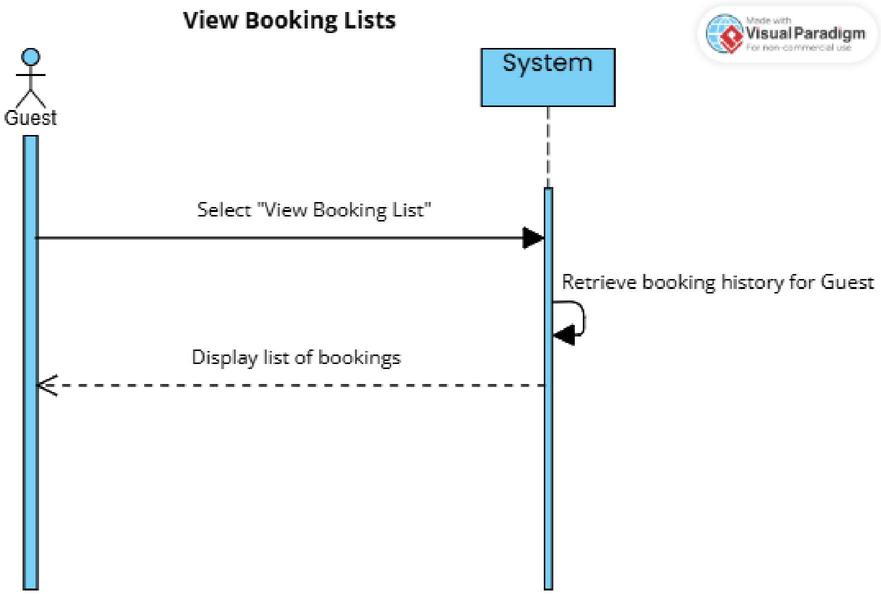
Book Properties



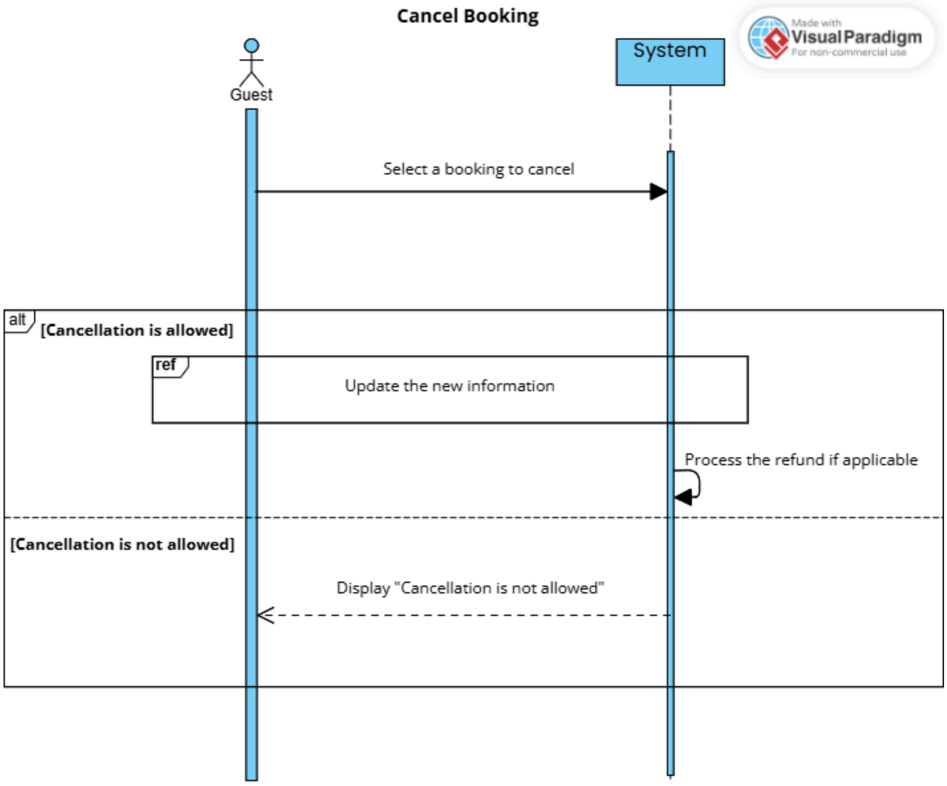
Make Payment


Guest
System
PaymentGateWay
Display payment options: Card or Online Banking
Select payment method and provide details
Send payment request
Payment response (success/failure)
alt
[Payment successful]
ref
Display "Payment successful. Booking confirmed!"
[Payment failed]
Display "Payment failed. Please try again later."

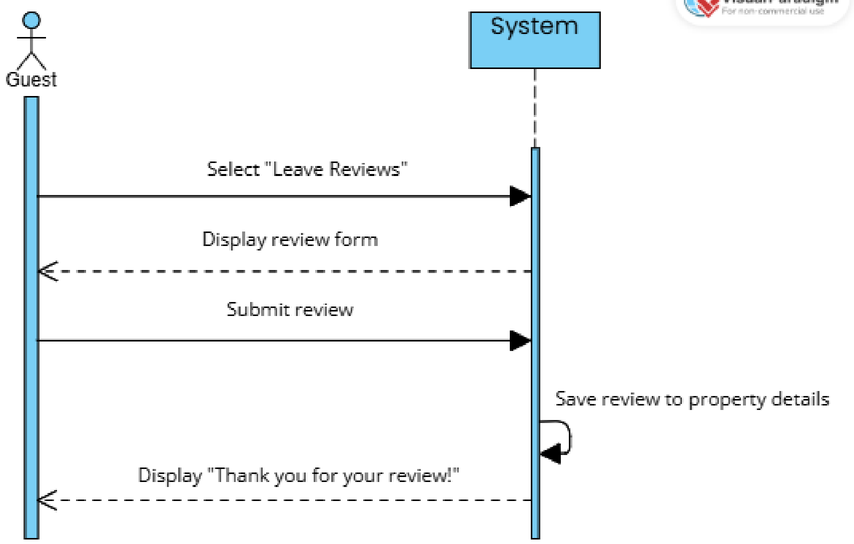
View Booking Lists


System
Guest
Select "View Booking List"
Retrieve booking history for Guest
Display list of bookings

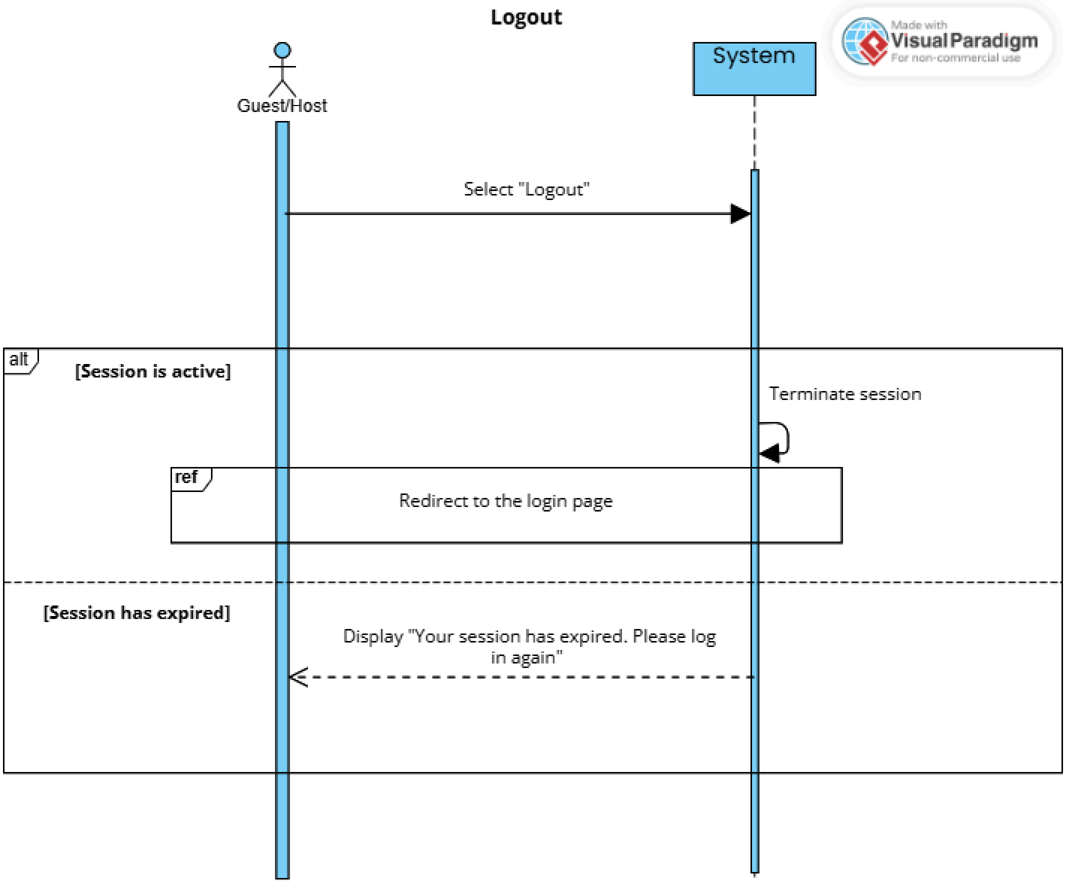
Cancel Booking

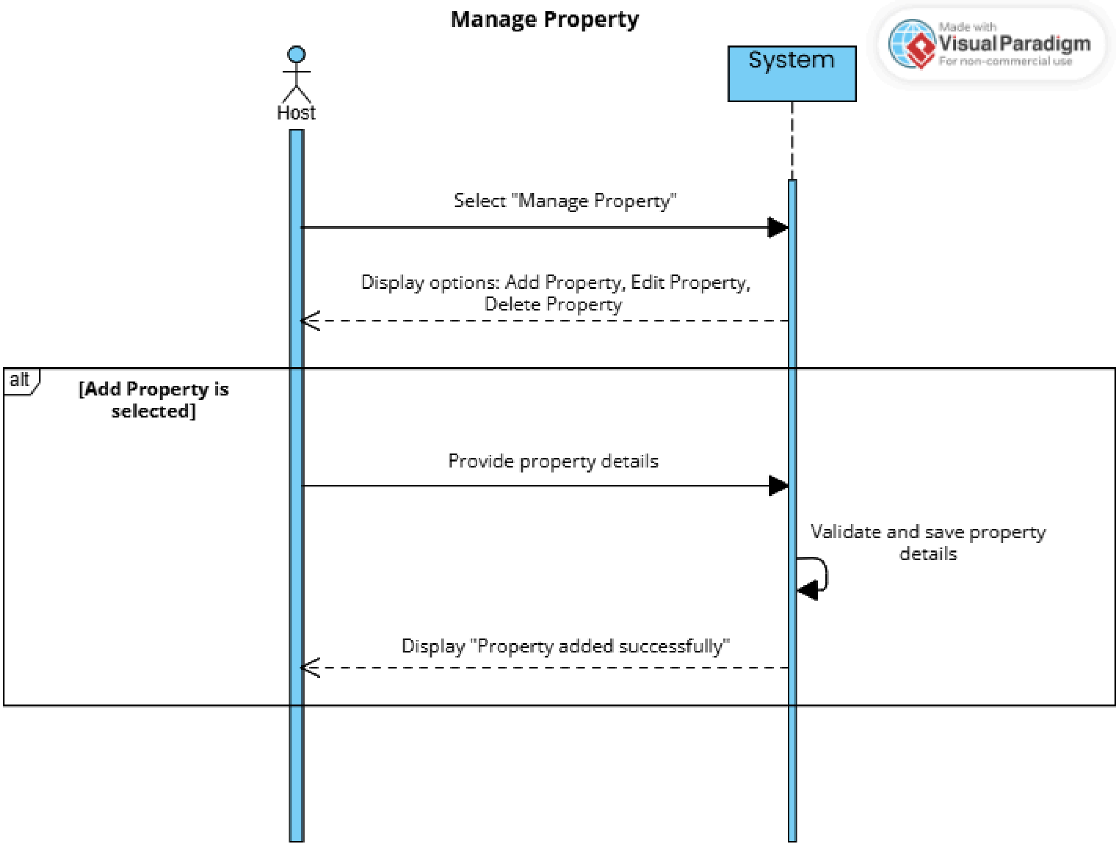

Guest
System
Select a booking to cancel
alt
[Cancellation is allowed]
ref
Update the new information
Process the refund if applicable
[Cancellation is not allowed]
Display "Cancellation is not allowed"

Leave Review


Guest
System
Select "Leave Reviews"
Display review form
Submit review
Save review to property details
Display "Thank you for your review!"

Logout




Manage Property
Made with Visual Paradigm For non-commercial use
Host
System
Select "Manage Property"
Display options: Add Property, Edit Property, Delete Property
alt
[Add Property is selected]
Provide property details
Validate and save property details
Display "Property added successfully"

CHAPTER 3

DESIGN AND IMPLEMENTATION

3.1 INTRODUCTION

This chapter provides an in-depth look at the design and implementation of the Roomly System, a platform that streamlines short-term rental services that function similarly to Airbnb. This system is designed to offer a seamless experience for both guests and hosts, ensuring smooth property bookings, secure transactions and efficient management tools for administrators to oversee the whole platform operations.

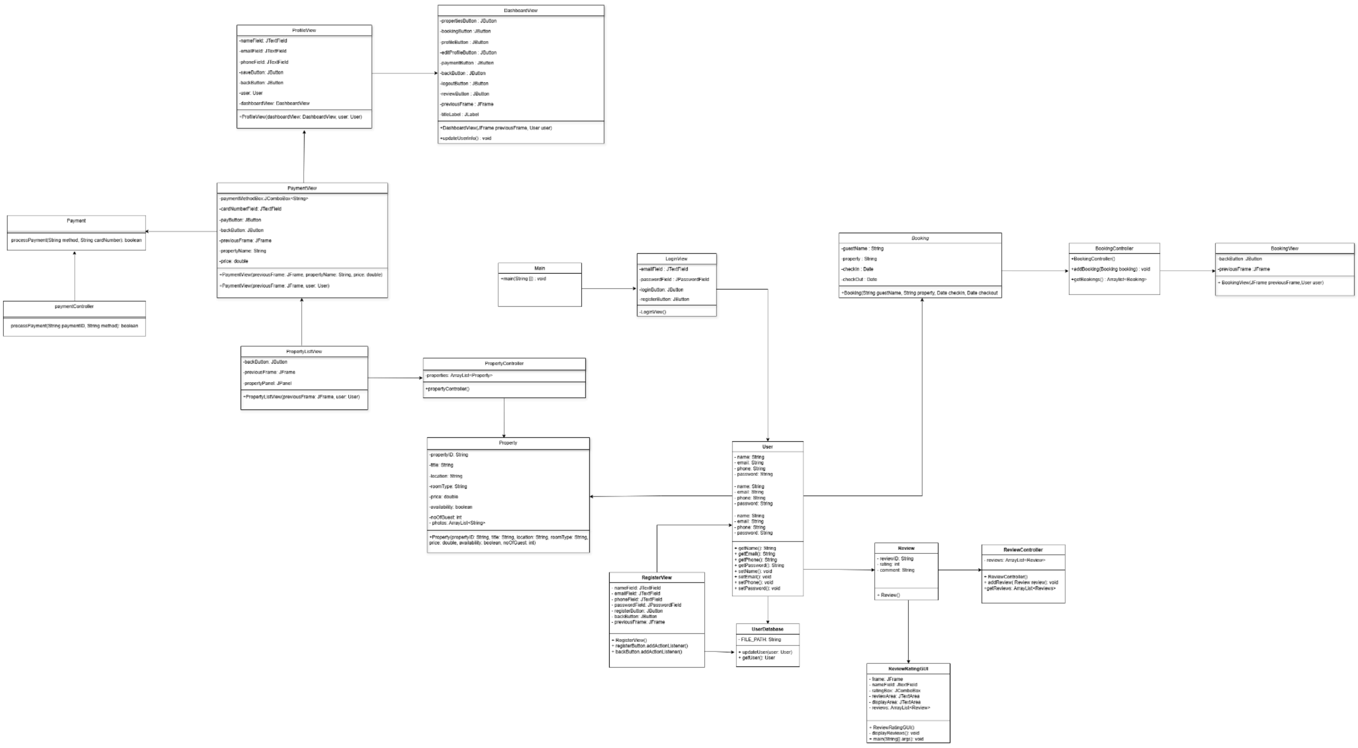
In the design phase, system requirements and user needs are thoroughly examined and transformed into a structured architectural plan. This stage includes the creation of UML (Unified Modeling Language) diagrams like class diagrams, sequence diagrams and use case diagrams. These visual representations are essential for understanding the system's structure and workflow, ensuring all components integrate effectively to the functional requirements.

The class diagrams play such a vital role in defining the system's foundation by identifying core classes, their properties, methods and relationships. From class diagrams itself, it can provide a structured guide to developers and illustrate how different modules interact to form a cohesive system. Key components of the Roomly system are Guest, Host, Property, Reservation, Payment and Review that are defined to showcase their interconnections to support processing and user feedback.

The implementation phase focuses more on bringing the conceptual design into a fully functional system through coding, integration and testing. Each system component is developed incrementally to ensure reliability and usability. For the user interface, it's designed to be intuitive to enable guests to browse and book properties effortlessly while allowing hosts to manage their listings properties and handle the reservations. Additionally, administrative functionalities are incorporated to oversee the whole platform operations.

The chapter delves into the deeper of both design and implementation by providing insights into UML diagrams, system architecture and functional outputs. By exploring these aspects, users will gain a better understanding of how Roomly is structured and operates. The detailed analysis highlights not only the technical parts but also the significance of a well-planned design and the development process in creating a dependable and user-friendly rental platform.

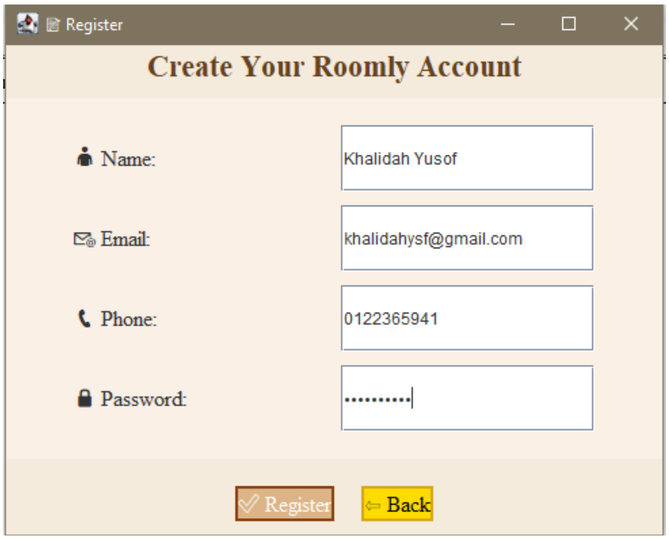
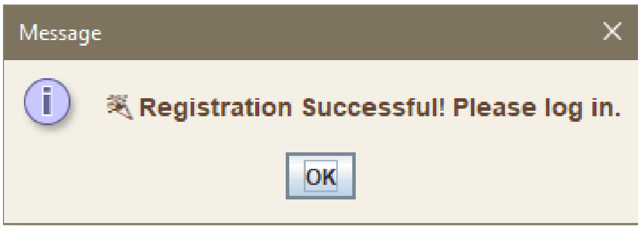
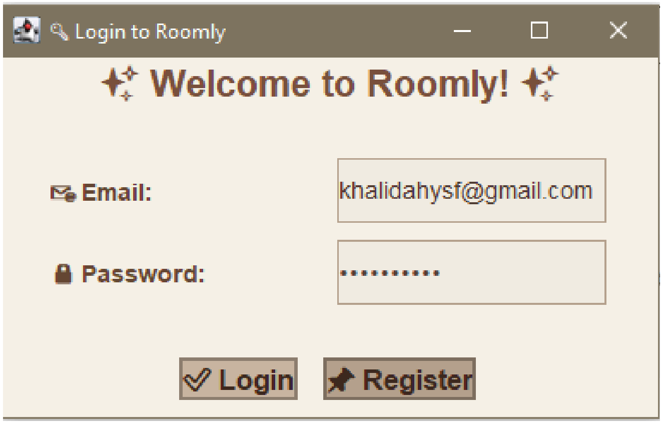
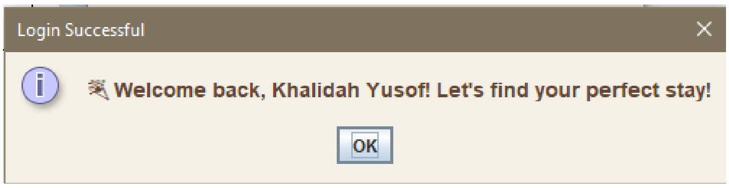
3.2 CLASS DIAGRAM

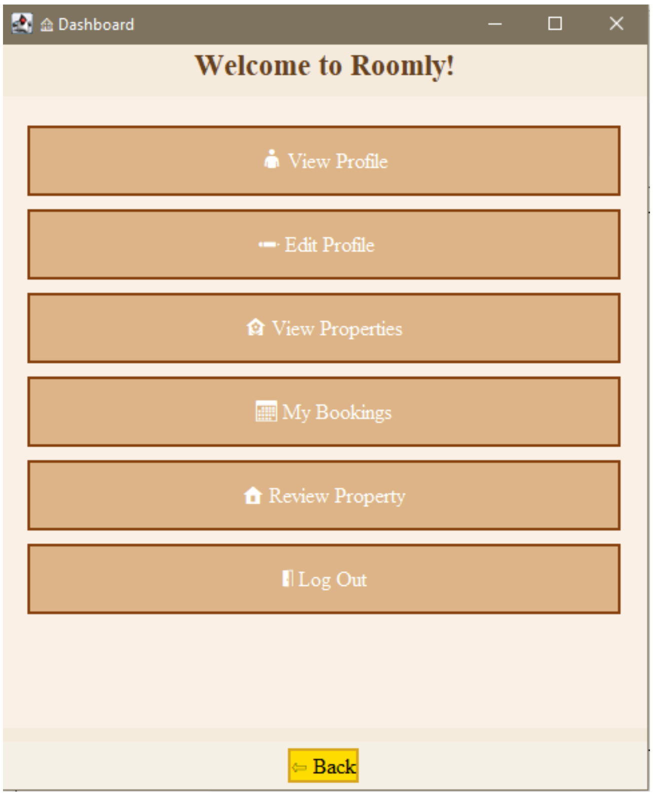
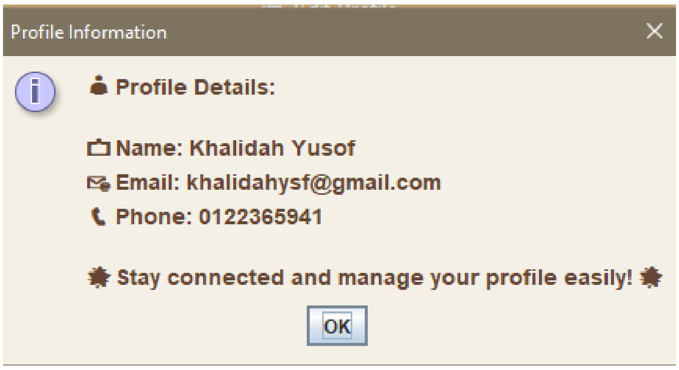


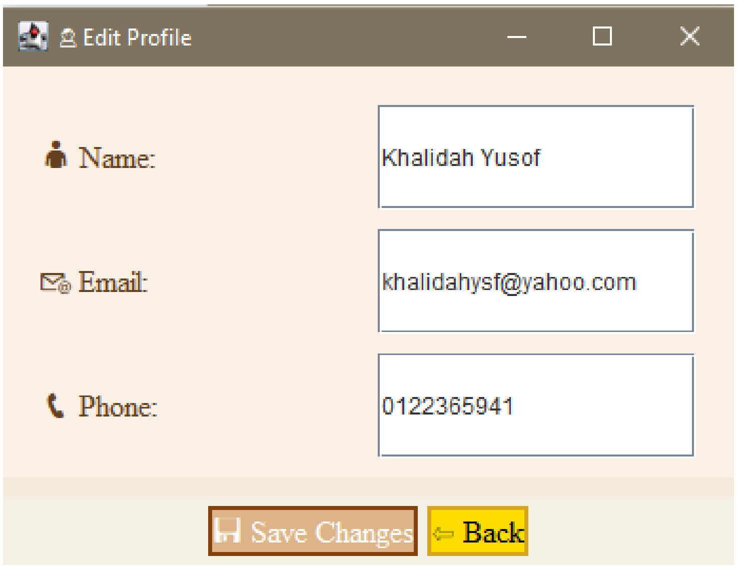
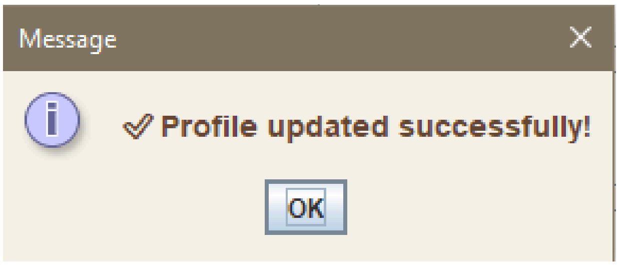
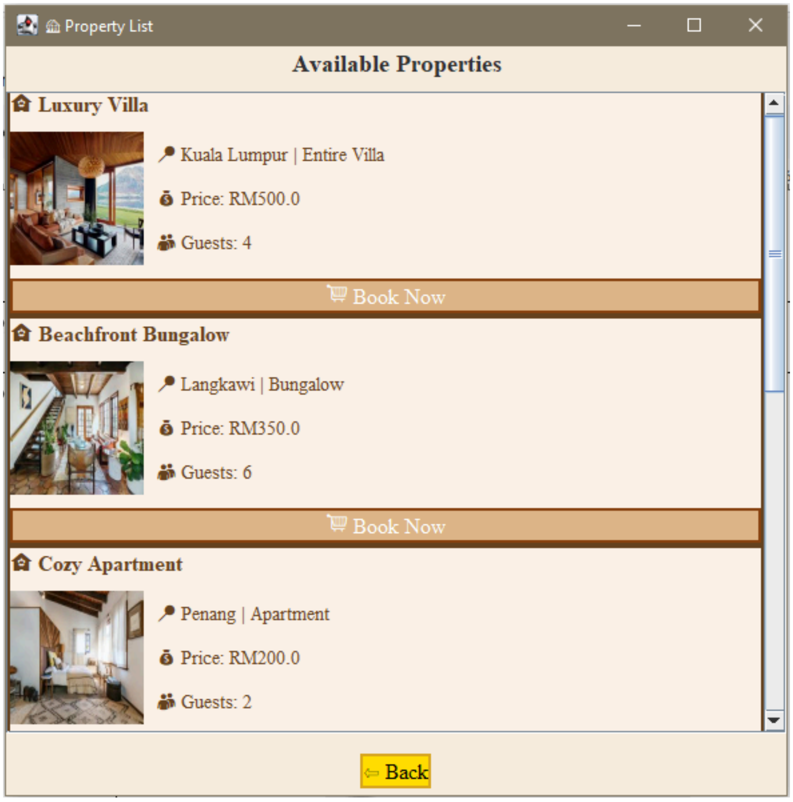
https://drive.google.com/file/d/1E55lYkXJQGbQiu91nHZ3A7U1vRvvjjTu/view?usp=sharing

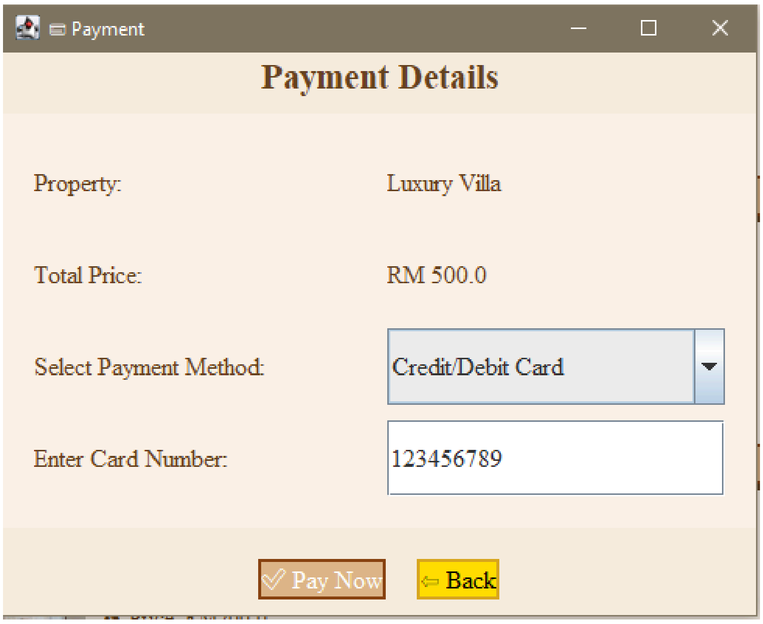
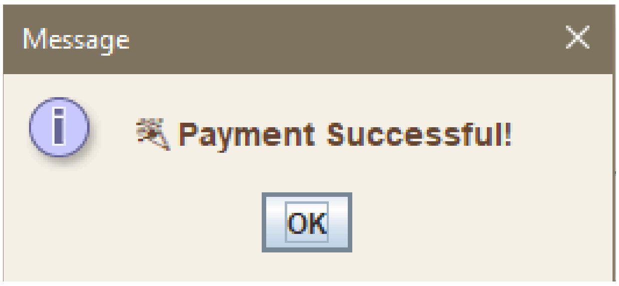
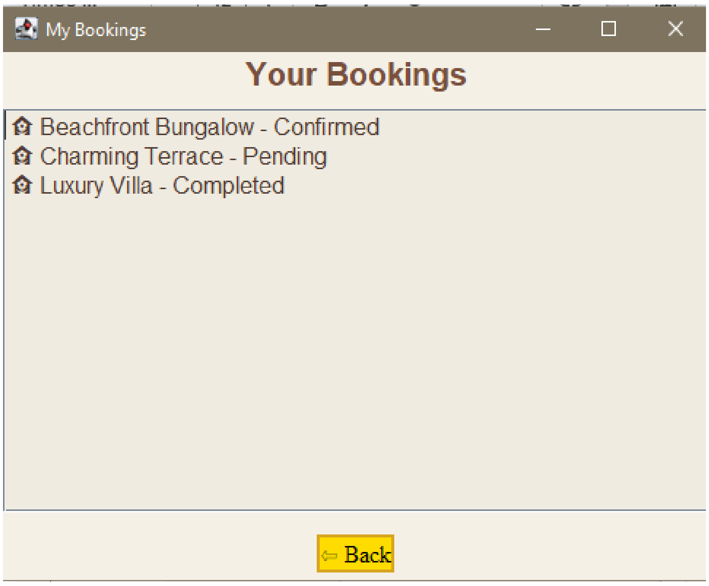
The class diagram represents a Property Booking System following the Model-View-Controller (MVC) architecture, ensuring modularity and scalability. The Model layer consists of key entities: User that stores user details, Property represents available properties, Booking that manages reservations with check-in/check-out dates, Review stores user feedback, and Payment handles transactions. The Controller layer includes BookingController, ReviewController, PropertyController, and PaymentController, responsible for handling business logic such as adding bookings, processing payments, and retrieving reviews. The View layer contains UI components like LoginView, PropertyListView, BookingView, and PaymentView, which interact with controllers to display data and capture user input. The relationships between these classes allow users to register, browse properties, make bookings, process payments and leave reviews, ensuring a smooth booking experience.

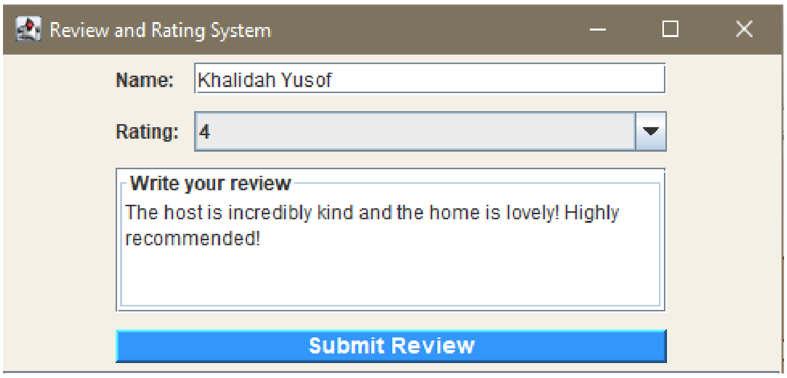
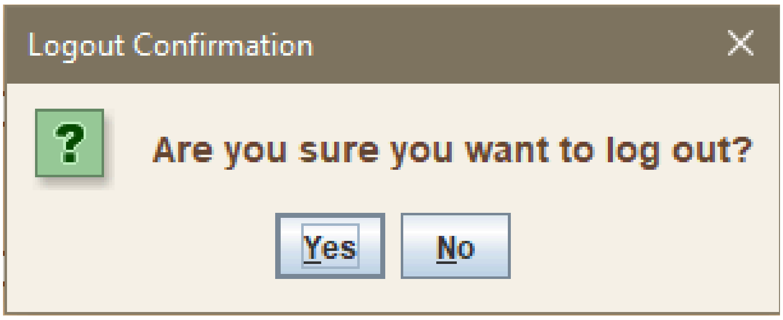
3.3 SAMPLE OUTPUT

No	Screen	Output
1	Register	 Create Your Roomly Account Name: Khalidah Yusof Email: khalidahysf@gmail.com Phone: 0122365941 Password: ••••••••••  Registration Successful! Please log in.
2	Login	 Welcome to Roomly! Email: khalidahysf@gmail.com Password: •••••••••••  Welcome back, Khalidah Yusof! Let's find your perfect stay!

3	Dashboard	 Dashboard Welcome to Roomly! View Profile Edit Profile View Properties My Bookings Review Property Log Out Back
4	Profile Information	 Profile Details: Name: Khalidah Yusof Email: khalidahysf@gmail.com Phone: 0122365941 Stay connected and manage your profile easily!

5	Edit Profile	 Edit Profile Name: Khalidah Yusof Email: khalidahysf@yahoo.com Phone: 0122365941  Message ✔ Profile updated successfully!
6	Property List	 Property List Available Properties Luxury Villa Kuala Lumpur | Entire Villa Price: RM500.0 Guests: 4 Beachfront Bungalow Langkawi | Bungalow Price: RM350.0 Guests: 6 Cozy Apartment Penang | Apartment Price: RM200.0 Guests: 2

7	Payment	 Payment Details Property: Luxury Villa Total Price: RM 500.0 Select Payment Method: Credit/Debit Card Enter Card Number: 123456789  Payment Successful!
8	Booking	 Your Bookings Beachfront Bungalow - Confirmed Charming Terrace - Pending Luxury Villa - Completed

9	Review and Rating	 Name: Khalidah Yusof Rating: 4 Write your review The host is incredibly kind and the home is lovely! Highly recommended!
10	Logout	 Are you sure you want to log out?

3.4 PROJECT DEMO

https://drive.google.com/file/d/1CcJaKth-olyRZd8El-CXEzp2yAZP4T7o/view?usp=drive_link