Software Requirements and Design Document

for

Hotel Management System

Prepared by M. Hamdan Rauf, Abdullah Ilyas, Abdul Qudoos

TechBrothers

07-06-2024

Contents

1. Introduction

1.1 Purpose

The purpose of this document is to collect, analyze, and define high-level needs and features of the Hotel Management System. It focuses on the capabilities needed by the stakeholders, and the target users, and why these needs exist. The details of how the Hotel Management System fulfils these needs are detailed in the use-case and supplementary specifications.

1.2 Product Scope

This Vision Document applies to the Hotel Management System (HMS), which will be developed by our implementation team. The team will develop this MVC system to interface with existing Hotel Management Systems. The (HMS) monitors, controls, and coordinates a wide variety of hotel services such as room booking and reservations, guest check-in and check-out processes, housekeeping schedules, in-room dining services, facility bookings, guest billing and payment processing, and customer service and feedback mechanisms. The system supports local remote access through land-line phones, cell phones or via the internet.

1.3 Title

The Hotel Management System (HMS) provides visitor experiences and improves hotel operations, offering a profitable business opportunity. It provides easy check-in and check-out procedures, tailored environmental controls, and automated room allocations. The HMS improves passenger attraction and retention through combining with dynamic booking and pricing solutions. The easy coordination of food, facility reservations, and housekeeping results in operational efficiencies. With its scalable design, it supports both third-party integrations and current IT infrastructures, making it suitable for a range of hotel kinds and sizes. By providing real-time insights to hotel workers and management, this solution boosts client pleasure and loyalty.

1.4 Objectives

For	Hotel operators and owners who wants to smooth the operations, guests, tourists
Who	needs a complete management system that improves guest experience, performance, and revenue improvement. This system must go besides simple reservation and check-in/check-out features.
The Hotel Management System (HMS)	Is a software and hardware product

That	offers flexible check-in and check-out procedures, effective housekeeping and maintenance scheduling, in-room dining management, dynamic room booking and pricing, unique guest experience management.
Unlike	Traditional hotel management system that has many issues like minimum flexibility, and also have lack of real-time check-in/check-out.
Our product	monitors, controls, and coordinates a wide variety of hotel services such as room booking and reservations, guest check-in and check-out processes, housekeeping schedules, in-room dining services, facility bookings, guest billing and payment processing, and customer service and feedback mechanisms. The system supports local remote access through land-line phones, cell phones or via internet.

1.5 Problem Statement

The problem of	still maintaining high room rates, offering amazing guest experiences, and improving performance in a hotel sector that is highly profitable and growing quickly.
affects	All types of hotel owners and operators, but especially those in the luxury and mid-range categories want to make their hotels stand out from the competition and raise profits.
the impact of which is	increasing operational cost, difficulty providing unique guest services on a regular basis, and challenges using data to inform profitable choices can all have a negative impact on the bottom line by reducing visitor satisfaction and loyalty.
a successful solution would be	a flexible, scalable Hotel Management System (HMS) that works in accordance with current hotel operations. Dynamic room booking and pricing, automatic guest service adjustments, proper utilisation of resources for facilities and cleaning, and large data analytics for important decisions are just a few of the features that this system would provide.

2. Overall Description

2.1 Product Perspective

The Hotel Management System (HMS) is a comprehensive software program designed to improve hotel and housing facilities' performance and user fun. With mobile access and direct contact, it offers individual customer experience in addition to easy reservation, guest service, room assignment, and payment management. Through changing pricing tools, the system ensures appropriate room use and income while improving cleaning and maintenance duties for better service quality. With its deep analytics, the HMS offers useful data that helps with decisions. Scalability in design allows for different hotel sizes and easily interacts with third-party platforms and current infrastructure, making it an essential element for improving hotel operations and fulfilling business goals.

2.2 Product Functions

- Booking & Reservation Management
- Maintenance Request Management
- Inventory Management
- Analyzing Reports (using History mechanism)
- Feedback Collection

2.3 List of Use Cases

Implemented:

- Search Room Availability
- Book Room
- View Booking History
- Manage Rooms
- Handle Walk-in Bookings
- Generate Invoice
- Modify Booking
- Manage Staff Accounts
- Manage Prices

Rest UCs:

- Request for Special Accommodation
- Process Payment
- Manage Staff Account
- Oversee Hotel Operations
- Resolve Booking Issues
- Manage Personal Profile
- Online Check-in
- Request Room Service
- Manage Inventory

2.4 Extended Use Cases

Implemented:

Use Case ID: UC01	
Use Case Name: Search for Room Availability	
Scope: Hotel Management System	
Level: User Goal	
Primary Actor: Guest	
Stakeholders and Interests: **Guest:** Wants to easily find available rooms that meet their criteria for stay dates, number of guests, and room preferences. **Hotel Management:** Interested in maximizing room occupancy and providing a seamless booking experience to increase customer satisfaction and revenue. **Customer Service:** Aims to provide support for guests using the system and resolve any issues or queries related to room availability.	
Pre-Condition: Room inventory, including details of room types, availability, and restrictions, is up to date.	
Post-Condition: The user has the option to proceed with booking one of the available rooms.	
Main Success Scenario:	
User Action	**System Response**

1. User accesses the hotel booking system. 3. Users input their stay dates, number of guests, and room preferences.	2. System displays the room search interface. 4. System validates the input and searches the room inventory for matches. 5. System displays a list of available rooms that match the criteria.

Extensions:

3a. User enters invalid date range:

- System displays an error message and requests the user to input a valid date range.

3b. User selects a few guests that exceeds the maximum room capacity:

- The system displays an error message and advises on the maximum capacity or suggests booking multiple rooms.

5a. No rooms are available for the selected criteria:

- System informs the user that no rooms are available for the selected dates and criteria.
- System may suggest alternative dates or room types.

Special Requirements:

- The system should be able to handle high volumes of searches, especially during peak booking seasons.

- The room search and availability check should be performed in real-time to ensure accuracy.

Technology and data variation list:

- The system should be accessible from multiple devices, including desktops, laptops, tablets, and smartphones.
- Room preferences may include room type (e.g., single, double, suite), amenities (e.g., Wi-Fi, air conditioning), and accessibility options.

Frequency: Potentially multiple times a day, depending on the size of the hotel and the number of users accessing the system.

Open Issues:

- How does the system handle request for group bookings, which may involve complex requirements and multiple rooms?
- What integrations are required with other hotel management systems, such as payment processing and customer relationship management (CRM) systems, to streamline the booking process?

Use Case ID: UC02

Use Case Name: Book a Room

Scope: Hotel Management System

Level: User Goal

Primary Actor: Guest

Stakeholders and Interests:

Guest: Wants to easily book a selected room using personal and payment information with the assurance of security and privacy.

Hotel Management: Interested in efficiently converting room availability searches into bookings to maximize occupancy and revenue.

Payment: Ensures secure and reliable processing of payment transactions.

Pre-Condition:

- User has successfully searched for and identified a room to book.
- The hotel booking system is operational and accessible.

Post-Condition: The room availability is updated in the hotel's inventory system to reflect the booking.

Main Success Scenario:

User Action	**System Response**
1. User selects a room from the available options.	2. System displays the room booking form.
3. User enters personal information (dates).	4. System validates the information and show total price.
5. Click on checkIN room	6. Send a room booking request to manager

Extensions:

6a. Payment information is invalid, or transaction fails:

- System displays an error message and requests the user to enter valid payment information again.

6b. User wants to modify the booking details (e.g., dates, number of guests) after entering payment information:

- The system allows the user to modify the booking details before final confirmation.
- System updates the payment amount if necessary and confirms the new details with the user.

Special Requirements:

- The system must comply with data protection regulations to safeguard personal and payment information.
- Payment processing should be secure and integrate with widely accepted payment gateways.

Technology and data variation list:

- Users can complete the booking process on various devices, including desktops, laptops, tablets, and smartphones.

• Payment methods might include credit cards, debit cards, online payment platforms, and direct bank transfers.
Frequency: Multiple times daily
Open Issues: • How will the system handle last-minute cancellations or modifications to the booking by the user?

Use Case ID: UC03	
Use Case Name: View Booking History	
Scope: Hotel Management System	
Level: User Goal	
Primary Actor: Guest	
Stakeholders and Interests: **Registered User:** Aims to access and review their past and current bookings for reference and tracking. **Hotel Management:** Interested in providing a user-friendly interface for guests to view their booking history.	
Pre-Condition: The user is authenticated and logged into the hotel management system.	
Post-Condition: The user successfully views their booking history, including details and status.	
Main Success Scenario:	
User Action	**System Response**

1. User selects the "Booking History" option from the user dashboard. 3. User selects a specific booking or applies filters for specific time periods or booking status.	2. System retrieves and displays a list of the user's past and current bookings, including reservation dates, room types, and booking status. 4. System presents detailed information for the selected booking, including payment details, room preferences, and any associated notes.
5. User reviews the booking details and may choose to print a confirmation or contact customer service for further assistance.	

Extensions:

2a. No bookings match the selected criteria.

System informs the user that no bookings were found and suggests adjusting the filters or checking back later.

4a. User selects "Print Confirmation."

System generates a printable version of the booking confirmation and prompts the user to download or print the document.

Special Requirements:

- The booking history section should be easily accessible and navigable within the hotel management system.
- User interface design should prioritize user experience, allowing for a seamless and intuitive process.

Technology and data variation list:

- The system should support various filters, such as date range, booking status, and room type, to enhance the user's ability to find specific bookings.
- Booking details may include reservation ID, check-in and check-out dates, room type, total cost, and payment status.

Frequency: Users may access their booking history as needed, particularly before upcoming stays or for record-keeping purposes.

Open Issues:

- How does the system handle cancellations or modifications to bookings, and what information is available to the user in such cases?
- Are there any legal or compliance considerations for the collection and storage of guest booking information within the hotel management system?

Use Case ID: UC09

Use Case Name: Manage Rooms

Scope: Hotel Management System

Level: User Goal

Primary Actor: Receptionist

Stakeholders and Interests:

Receptionist: Wants to efficiently assign rooms to guests according to their preferences and booking details to ensure a smooth check-in experience.

Guests: Expect their room assignments to align with their preferences and booking details, contributing to their overall satisfaction with their stay.

Hotel Management: Interested in maximizing room occupancy and providing guests with suitable accommodations to enhance their experience.

Pre-Condition: The Receptionist is logged into the hotel's system.

Post-Condition: The guest is assigned a room according to their preferences and booking details, and the room assignment is recorded in the system

Main Success Scenario:

Receptionist Action	**System Response**
1. The Receptionist accesses the room assignment section in the hotel management system. 3. The Receptionist add new room.	2. The system verifies the Receptionist's credentials and grants access to the room assignment section. 4. The system verify details and add room.
6. The Receptionist edit existing room 7. The Receptionist delete the room.	7. The system verify details and update room 8. The system confirms and delete room.

Extensions:

If there are no available rooms that match the guest's preferences or booking details, the Receptionist may need to explore alternative options or offer upgrades if possible.

If the Receptionist encounters difficulties during the room assignment process, such as technical issues or conflicting bookings, they may need to seek assistance from a supervisor or system administrator.

Special Requirements:

- The system should provide real-time updates on room availability to ensure accurate room assignments.
- Proper validation checks should be implemented to ensure that room assignments align with guest preferences and booking details.
- User-friendly interfaces should be implemented to facilitate easy navigation and selection of available rooms.

Technology and data variation list:

• The system should be accessible via a web interface or a dedicated software application. • Room availability and guest booking details may vary based on factors such as seasonal demand and special events.

Frequency: This use case occurs frequently throughout the day as guests check-in and new bookings are made.

Open Issues:

• How does the system handle situations where a guest requests a room change after the initial assignment?

Use Case ID: UC10
Use Case Name: Handle Walk-in Bookings
Scope: Hotel Management System
Level: User Goal
Primary Actor: Receptionist
Stakeholders and Interests: **Receptionist:** Aims to efficiently create new bookings for walk-in guests to ensure a seamless check-in experience. **Guests:** Expect to be provided with suitable accommodation upon arrival, even without prior reservations. **Hotel Management:** Interested in maximizing room occupancy and revenue by accommodating walkin guests effectively.
Pre-Condition: The Receptionist is logged into the hotel's system.
Post-Condition: A new booking is created for the walk-in guest, and the room assignment is recorded in the system.
Main Success Scenario:

Receptionist Action	**System Response**

1. The Receptionist receives a walk-in guest inquiry and accesses the booking creation section. 3. The Receptionist add guest details. 5. The Receptionist select guest and receptionist. 7. The Receptionist Enter Date and Click on check In.	2. The system verifies the Receptionist's credentials and grants access to the booking creation section. 4. The system add guest details in system. 6. The system retrieve details. 8. The system creates a new booking record for the walk-in guest and assigns the selected room.

Extensions:

If there are no available rooms, the Receptionist may need to inform the guest and explore alternative options such as nearby hotels.

If the walk-in guest has specific preferences or requirements, the Receptionist may need to search for suitable rooms or offer alternatives if available.

Special Requirements:

- The system should provide real-time updates on room availability to facilitate prompt handling of walk-in bookings.
- Proper validation checks should be implemented to ensure that new bookings are created accurately and efficiently.
- User-friendly interfaces should be implemented to facilitate easy input of guest information and preferences.

Technology and data variation list: • The system should be accessible via a web interface or a dedicated software application. • Room availability and guest preferences may vary based on factors such as seasonal demand and special events.
Frequency: This use case occurs sporadically throughout the day as walk-in guests arrive at the hotel.
Open Issues: • How does the system handle situations where multiple walk-in guests arrive simultaneously, and there is limited room availability?

Use Case ID: UC06
Use Case Name: Generate and Send Invoice
Scope: Hotel Management System
Level: User Goal
Primary Actor: Receptionist
Stakeholders and Interests:

Receptionist: To generate Invoices and handle customer payments through system. **Guests:** Expect to receive invoices detailing charges and services provided. **Hotel Management:** Interested in maintaining transparent and accurate financial records	
Pre-Condition: The guest have chosen desired room and Recipient is logged into system	
Post-Condition: An invoice is generated and sent to the guest.	
Main Success Scenario:	
Manger Action	**System Response**

1. The Receptionist opens the invoicing section in the hotel management system to generate the invoice generation process.	2. The system verifies the Receptionist's credentials and give access to the invoicing section.
3. The Receptionist searches deatils.	4. The system displays relevant booking and charge details for the Receptionist to review.
5. The Receptionist triggers the system to generate the invoice.	6. The system Generates invoice and display it.

Extensions:

5 A. Discount Application

5.1 The Receptionist has the option to apply discounts to specific charges on the invoice.

5.2 The system recalculates the total amount based on applied discounts and displays the revised charges for confirmation.

9 A. Invoice Preview

9.1 Before finalizing the invoice generation, the Receptionist can preview the invoice to ensure accuracy and completeness.

9.2 The system provides a detailed preview, allowing the Receptionist to verify all charges, services, and guest details before initiating the generation process.

Special Requirements:

- The system must ensure data security and only allow authorized users (such as the Manager) to perform staff account management tasks.
- Proper validation checks should be implemented to ensure the accuracy and completeness of the entered information.
- Audit logs should be maintained to track account creation, updates, and deletions for accountability purposes.

Technology and data variation list:

- The system should provide secure and reliable methods for generating, sending, and storing invoices.
- Invoice content may vary based on room rates, additional services utilized, and any applicable taxes or discounts.

Frequency: This use case occurs regularly, either at the time of booking confirmation or during the check-out process.

Open Issues:

- How does the system handle cases where the guest disputes charges or requests modifications to the invoice?
- What measures are in place to ensure the security and privacy of guest financial information during the invoicing process?
- How does the system handle scenarios where the guest prefers a printed copy of the invoice, and what is the process for providing it during check-out?

Use Case ID: UC07
Use Case Name: Manage Staff account

Scope: Hotel Management System
Level: User Goal
Primary Actor: Manager
Stakeholders and Interests: **Manager:** Wants to efficiently manage staff accounts and ensure proper role assignment and permissions. **Staff Members:** Want their accounts to be accurately created, updated, or deleted based on their roles and responsibilities. **System Administrator:** Needs to ensure that the system functions properly and securely handle account management operations.
Pre-Condition: The Manager is logged into the hotel's system.

Post-Condition: The staff accounts are managed according to the Manager's actions.	
Main Success Scenario:	
Manger Action	**System Response**
1. The Manager selects the option to manage staff accounts.	2. The system displays a list of existing staff accounts along with options to create, update, or delete accounts.
3. The Manager selects the option to create a new staff account.	4. The system prompts the Manager to enter the necessary details such as name, email, role, and permissions for the new account.
5.The Manager selects an existing staff account to update.	6. System Response: The system displays the current details of the selected account and allows the Manager to modify them as needed.
7.The Manager selects an existing staff account to delete.	8.The system prompts the Manager to confirm the deletion, then removes the selected account from the system.
Extensions: **6b.** If the Manager cancels the creation, update, or deletion process, the system returns to the main menu without making any changes. If there are errors in the input data during account creation or update, the system provides appropriate error messages and prompts the Manager to correct the information.	
Special Requirements:	

• The system must ensure data security and only allow authorized users (such as the Manager) to perform staff account management tasks. • Proper validation checks should be implemented to ensure the accuracy and completeness of the entered information. • Audit logs should be maintained to track account creation, updates, and deletions for accountability purposes.
Technology and data variation list: • The system should be accessible via a web interface. • The types of roles and permissions assigned to staff accounts may vary based on the hotel's organizational structure and requirements
Frequency: This use case occurs regularly as the hotel's staffing needs change, typically on a weekly or monthly basis.
Open Issues: • How to ensure that Managers are adequately trained to use the system for staff account management effectively?

Use Case ID: UC07
Use Case Name: Manage Rooms Pricing by analyzing trend
Scope: Hotel Management System
Level: User Goal
Primary Actor: Manager
Stakeholders and Interests: **Receptionist:** Wants to efficiently assign rooms to guests according to their preferences and booking details to ensure a smooth check-in experience. **Guests:** Expect their room assignments to align with their preferences and booking details, contributing to their overall satisfaction with their stay. **Hotel Management:** Interested in maximizing room occupancy and providing guests with suitable accommodations to enhance their experience.
Pre-Condition: The Receptionist is logged into the hotel's system.

<table>
<tr><td colspan="2">Post-Condition: The guest is assigned a room according to their preferences and booking details, and the room assignment is recorded in the system</td></tr>
<tr><td colspan="2">Main Success Scenario:</td></tr>
<tr><td>Receptionist Action</td><td>System Response</td></tr>
<tr><td>1. The Manager accesses the room assignment section in the hotel management system.

3. The Manager select the option of report of overall hotel rooms.</td><td>2. The system verifies the Receptionist's credentials and grants access to the room assignment section.

4. The system retrieves and displays:
and guest booking details. -
- List of available rooms based on current availability and guest preferences.
- Overall hotel occupancy rate (percentage).
- Number of available rooms.
- Occupancy rate for each room (number of times booked).
- Revenue generated by each room.</td></tr>
<tr><td>5. The Manager selects a room and analyze the room details .

7. The Manager update the room pricing accordingly.</td><td>8. The system updates the pricing details and confirms to manager.</td></tr>
<tr><td colspan="2">Extensions:</td></tr>
</table>

Special Requirements:

- The system should provide real-time updates on room availability to ensure accurate room assignments.
- Proper validation checks should be implemented to ensure that room assignments align with guest preferences and booking details.
- User-friendly interfaces should be implemented to facilitate easy navigation and selection of available rooms.

Technology and data variation list:

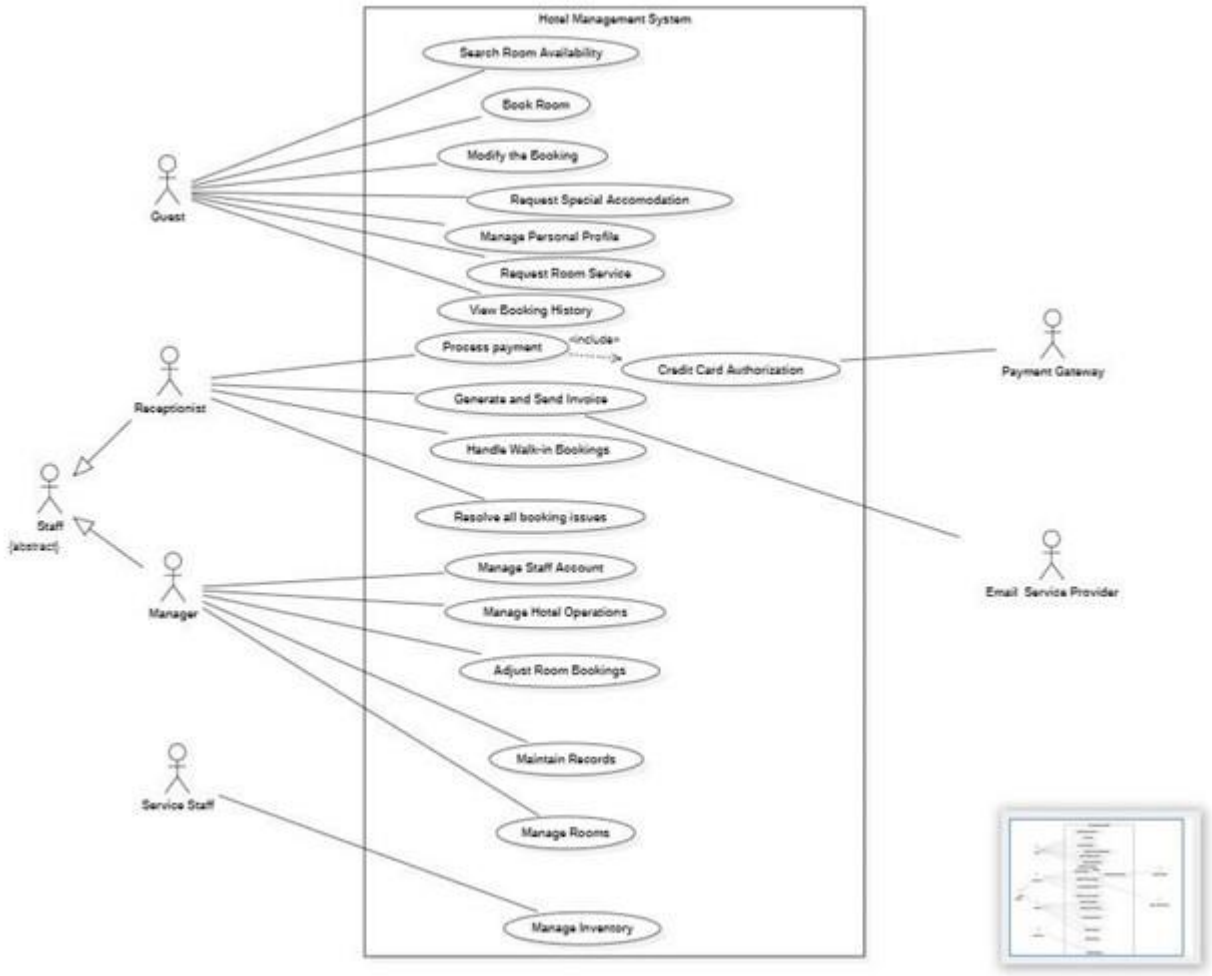
- The system should be accessible via a web interface or a dedicated software application.
- Room availability and guest booking details may vary based on factors such as seasonal demand and special events.

Frequency: This use case occurs frequently throughout the day as guests check-in and new bookings are made.

Open Issues:

- How does the system handle situations where a guest requests a room change after the initial assignment?

2.5 Use Case Diagram



3. Other Nonfunctional Requirements

3.1 Performance Requirements

- System responds quickly to user requests or changes in the environment after performing authentications.

3.2 Safety Requirements

- Customer Care users can only monitor the system and manually place a medical alert emergency request for an ambulance if any guest causes injury or health issue. (it's safety requirement as well)

3.3 Security Requirements

- Security for the HMS includes authentication, access control, data integrity, and data privacy.
- Authentication of the user is by identifier and password.

3.4 Software Quality Attributes

- Reusability
- Maintainability
- Robustness
- Correctness
- Usability

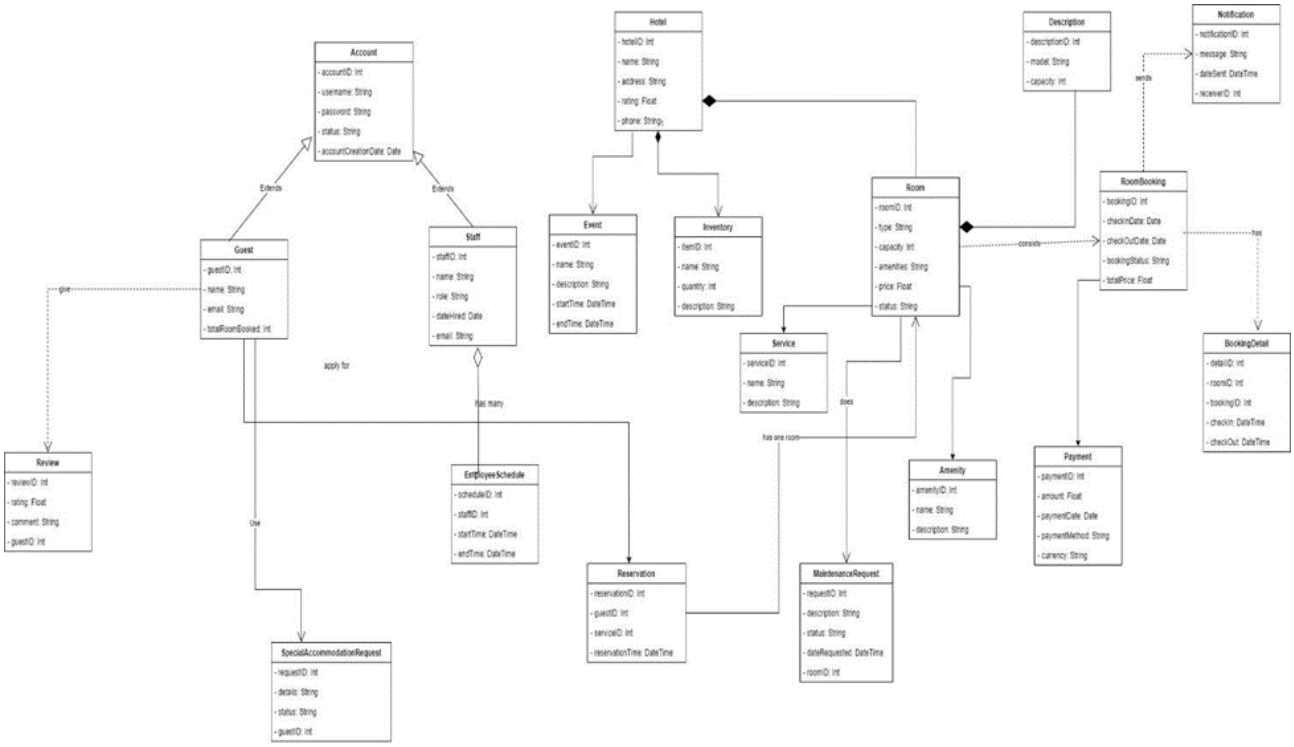
3.5 Operating Environment

It is not deployed but running on:

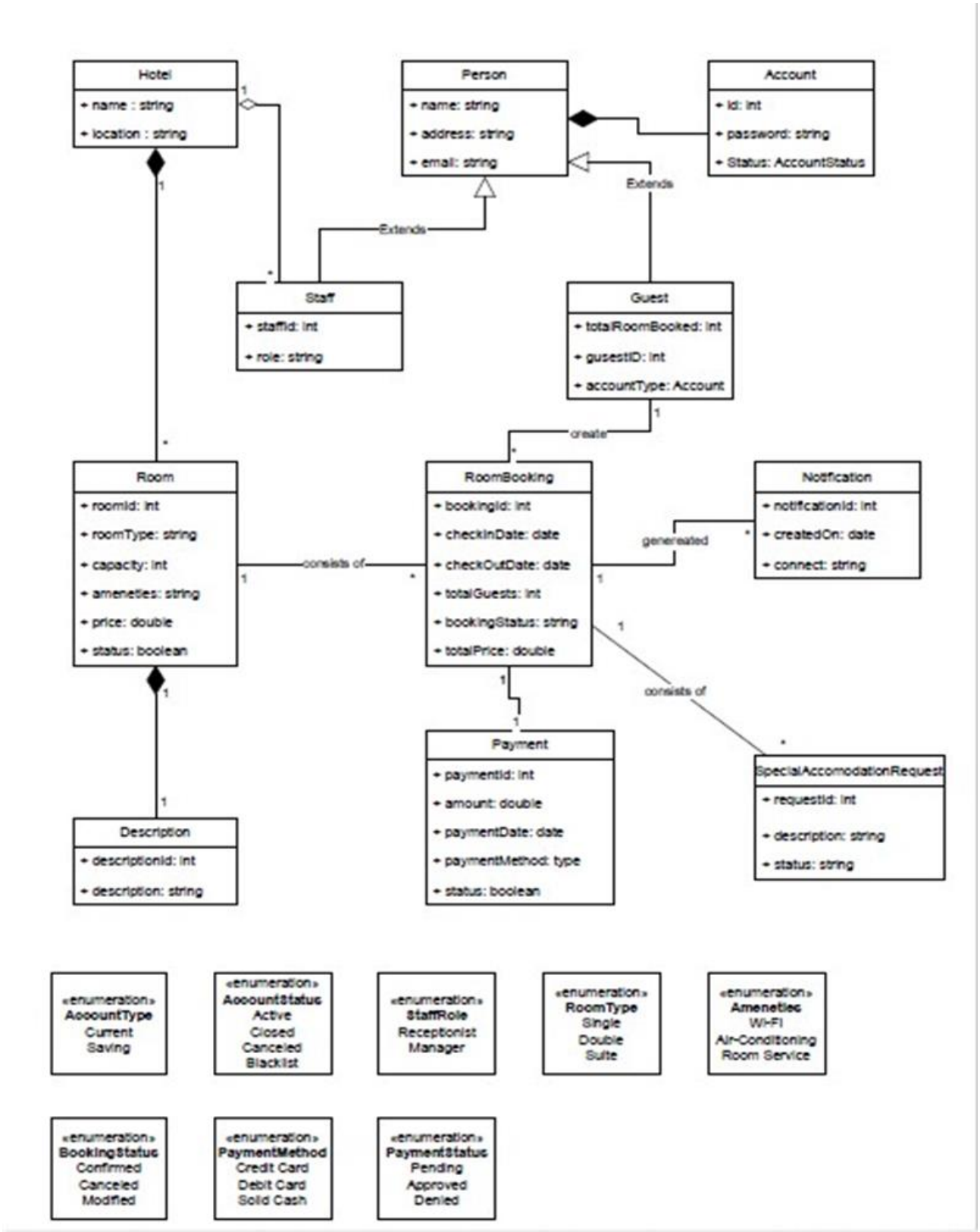
- Server using Apache Tomcat v9.0
- Eclipse 2023-12 build
- Backend Java SE-17
- DBMS → MY SQL
- Front-End → JSP + CSS

4. Domain Model

Updated:


Account
- accountID: Int
- username: String
- password: String
- status: String
- accountCreationDate: Date
Hotel
- hotelID: Int
- name: String
- address: String
- rating: Float
- phone: String
Description
- descriptionID: Int
- model: String
- capacity: Int
Notification
- notificationID: Int
- message: String
- dateSent: DateTime
- receiverID: Int
sends
Extends
Extends
Guest
- guestID: Int
- name: String
- email: String
- totalRoomBooked: Int
Staff
- staffID: Int
- name: String
- role: String
- dateHired: Date
- email: String
Event
- eventID: Int
- name: String
- description: String
- startTime: DateTime
- endTime: DateTime
Inventory
- itemID: Int
- name: String
- quantity: Int
- description: String
Room
- roomID: Int
- type: String
- capacity: Int
- amenities: String
- price: Float
- status: String
RoomBooking
- bookingID: Int
- checkInDate: Date
- checkOutDate: Date
- bookingStatus: String
- totalPrice: Float
consists
has
give
apply for
has many
Service
- serviceID: Int
- name: String
- description: String
does
has one room
BookingDetail
- detailID: Int
- roomID: Int
- bookingID: Int
- checkIn: DateTime
- checkOut: DateTime
Review
- reviewID: Int
- rating: Float
- comment: String
- guestID: Int
Use
EmployeeSchedule
- scheduleID: Int
- staffID: Int
- startTime: DateTime
- endTime: DateTime
Amenity
- amenityID: Int
- name: String
- description: String
Payment
- paymentID: Int
- amount: Float
- paymentDate: Date
- paymentMethod: String
- currency: String
Reservation
- reservationID: Int
- guestID: Int
- serviceID: Int
- reservationTime: DateTime
MaintenanceRequest
- requestID: Int
- description: String
- status: String
- dateRequested: DateTime
- roomID: Int
SpecialAccommodationRequest
- requestID: Int
- details: String
- status: String
- guestID: Int

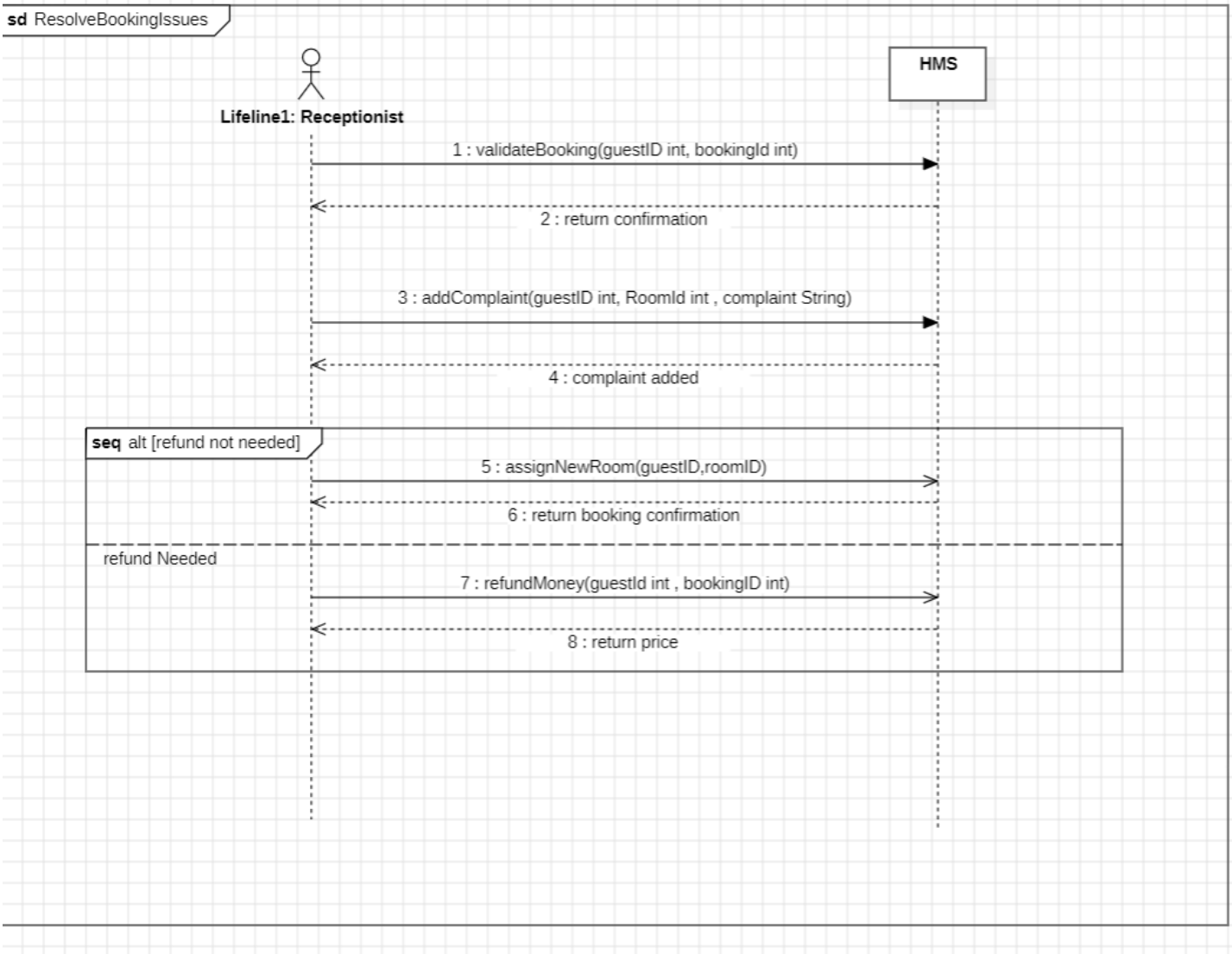
Previous:


Hotel
+ name : string
+ location : string
Person
+ name: string
+ address: string
+ email: string
Account
+ id: int
+ password: string
+ Status: AccountStatus
Extends
Extends
Staff
+ staffId: int
+ role: string
Guest
+ totalRoomBooked: int
+ guestID: int
+ accountType: Account
create
Room
+ roomId: int
+ roomType: string
+ capacity: int
+ amenetIes: string
+ price: double
+ status: boolean
consists of
RoomBooking
+ bookingId: int
+ checkInDate: date
+ checkOutDate: date
+ totalGuests: int
+ bookingStatus: string
+ totalPrice: double
generated
Notification
+ notificationId: int
+ createdOn: date
+ connect: string
consists of
Description
+ descriptionId: int
+ description: string
Payment
+ paymentId: int
+ amount: double
+ paymentDate: date
+ paymentMethod: type
+ status: boolean
SpecialAccomodationRequest
+ requestId: int
+ description: string
+ status: string
«enumeration»
AccountType
Current
Saving
«enumeration»
AccountStatus
Active
Closed
Canceled
Blacklist
«enumeration»
StaffRole
Receptionist
Manager
«enumeration»
RoomType
Single
Double
Suite
«enumeration»
Amenetiec
WI-FI
Air-Conditoning
Room Service
«enumeration»
BookingStatus
Confirmed
Canceled
Modified
«enumeration»
PaymentMethod
Credit Card
Debit Card
Solid Cash
«enumeration»
PaymentStatus
Pending
Approved
Denied

5. System Sequence Diagram and Sequence Diagram

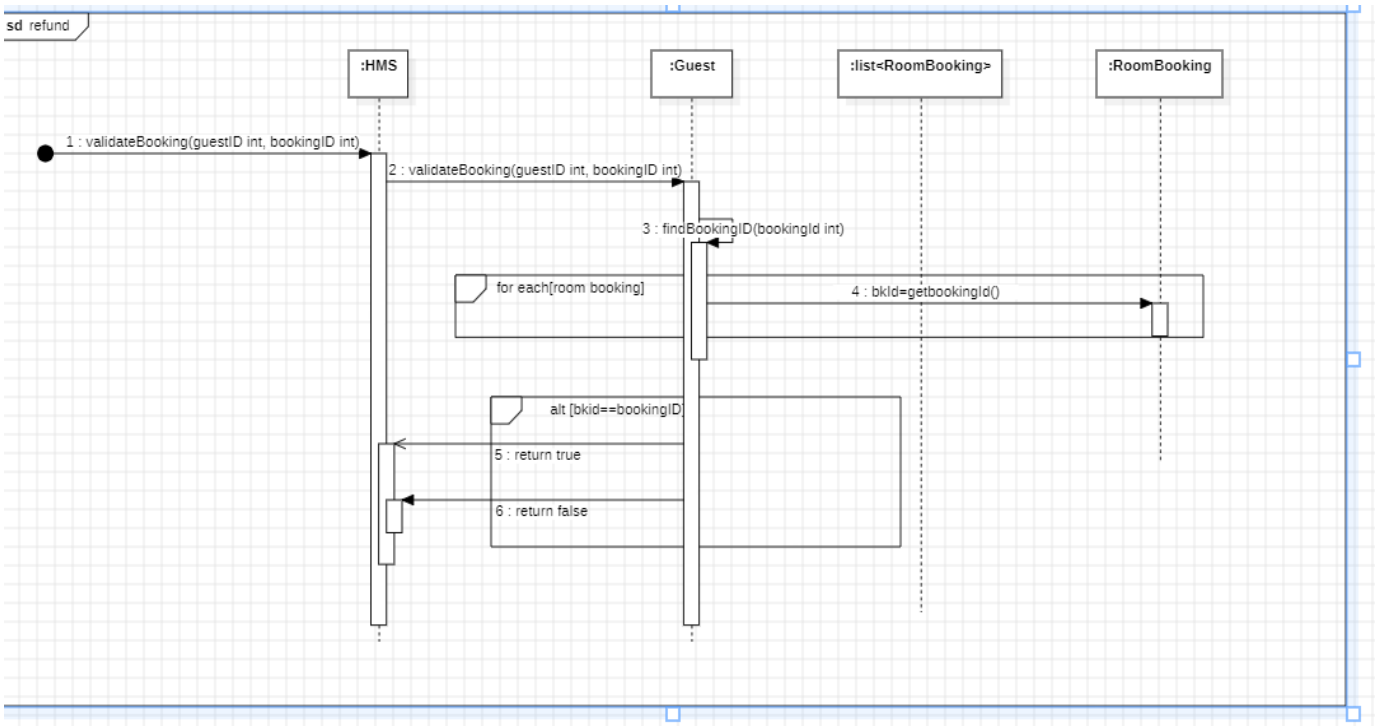
1. Resolve Booking Issues

SSD

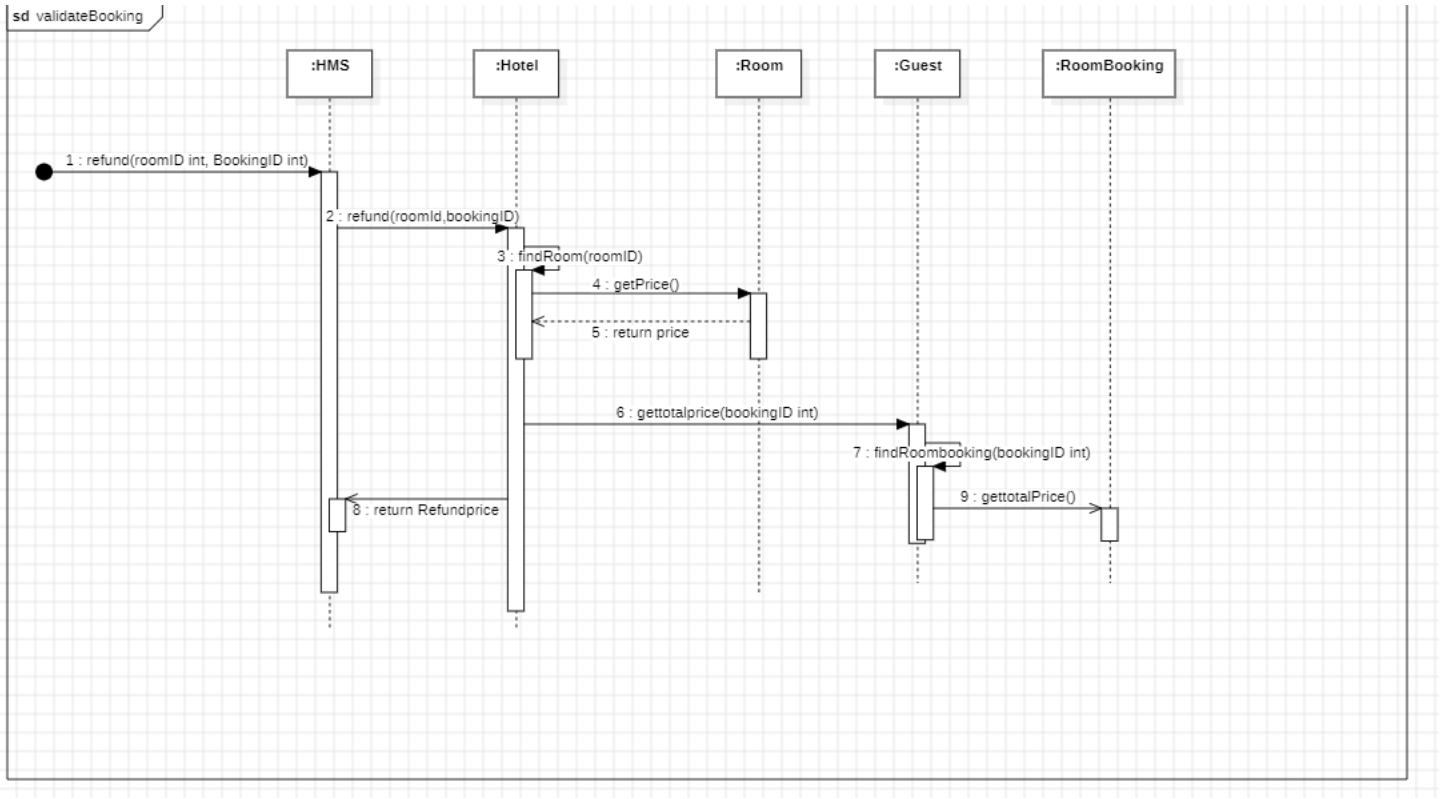


SDs

1:

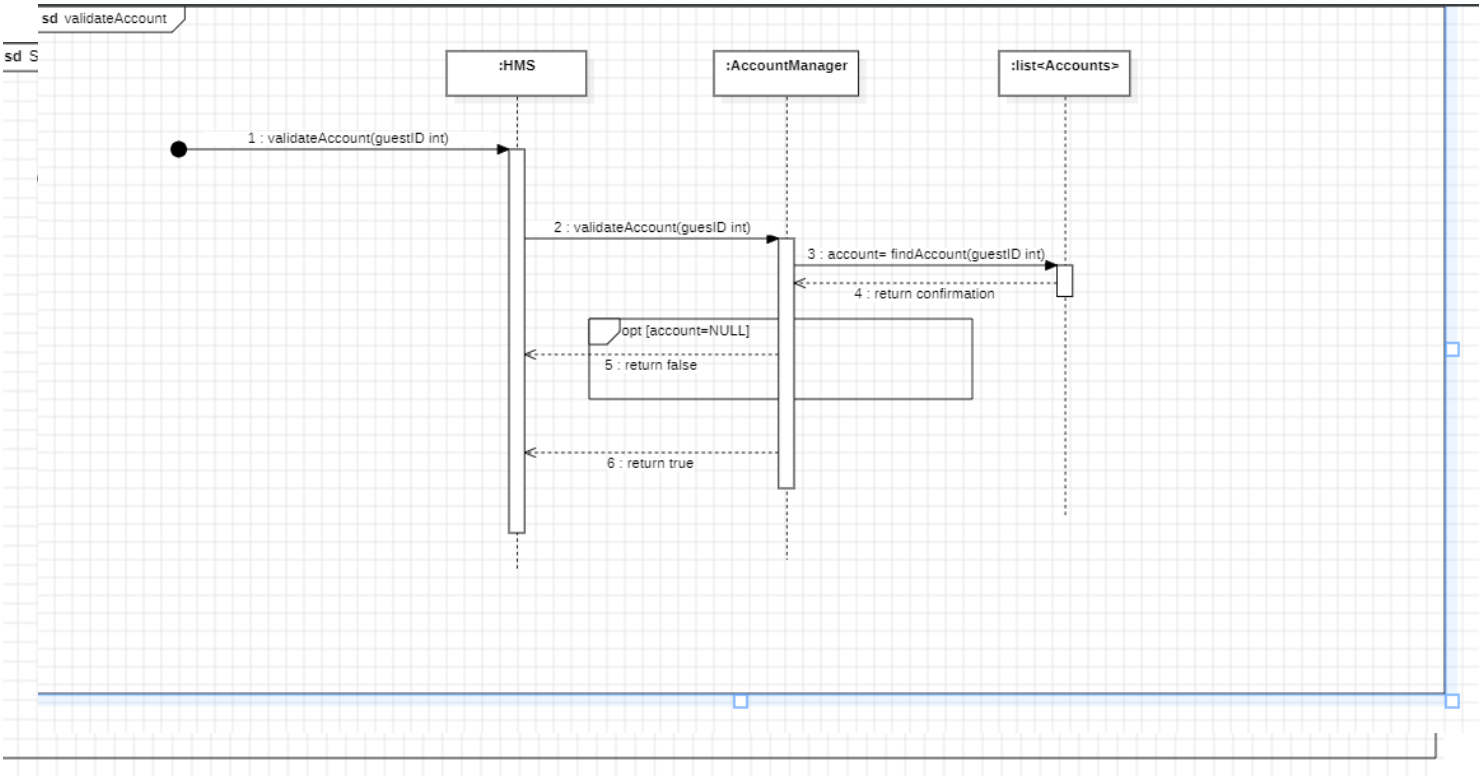

sd refund
:HMS
:Guest
:list<RoomBooking>
:RoomBooking
1 : validateBooking(guestID int, bookingID int)
2 : validateBooking(guestID int, bookingID int)
3 : findBookingID(bookingId int)
for each[room booking]
4 : bkId=getbookingId()
alt [bkid==bookingID]
5 : return true
6 : return false

2:


sd validateBooking
:HMS
:Hotel
:Room
:Guest
:RoomBooking
1 : refund(roomID int, BookingID int)
2 : refund(roomId,bookingID)
3 : findRoom(roomID)
4 : getPrice()
5 : return price
6 : gettotalprice(bookingID int)
7 : findRoombooking(bookingID int)
8 : return Refundprice
9 : gettotalPrice()

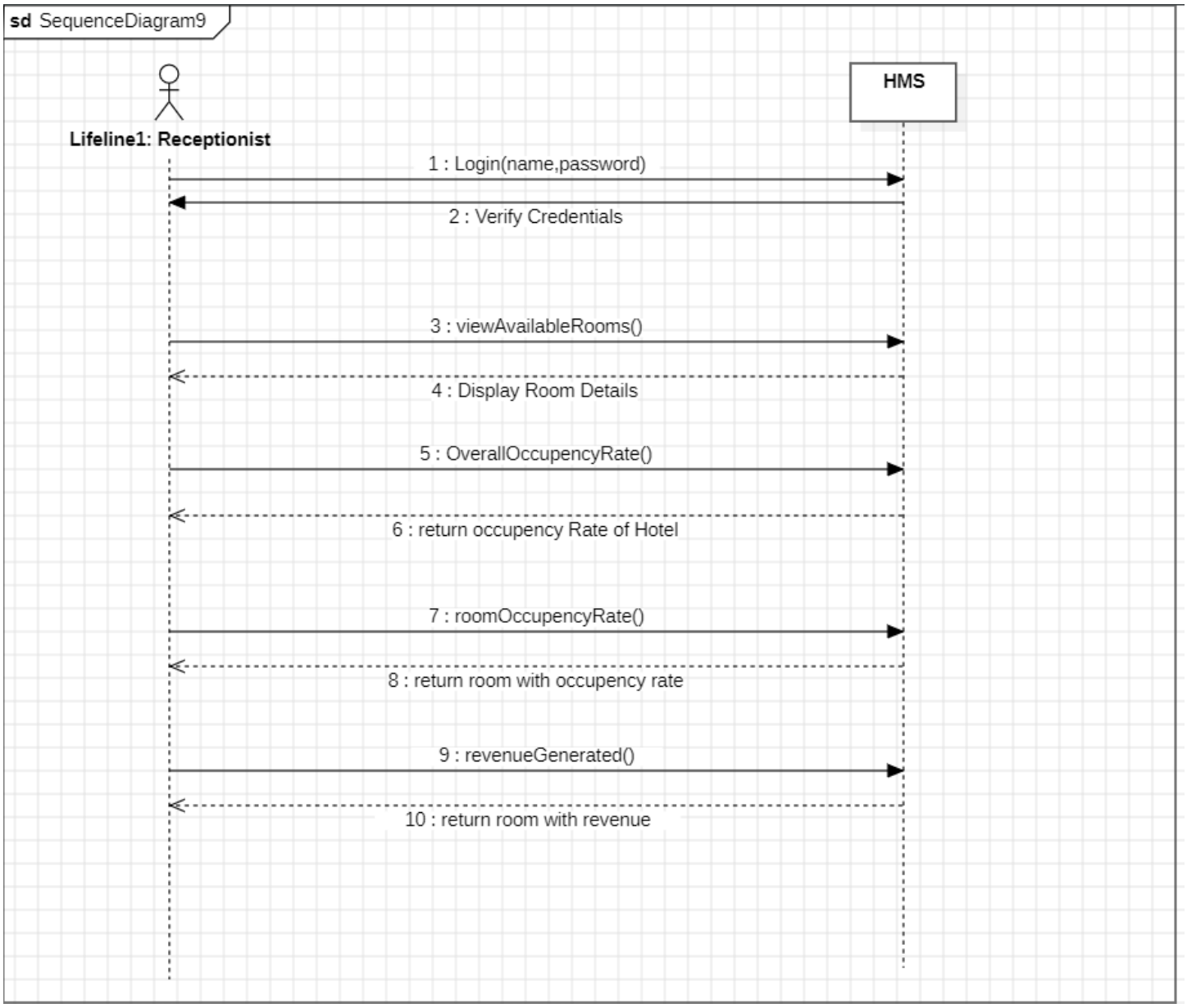
3:

4:



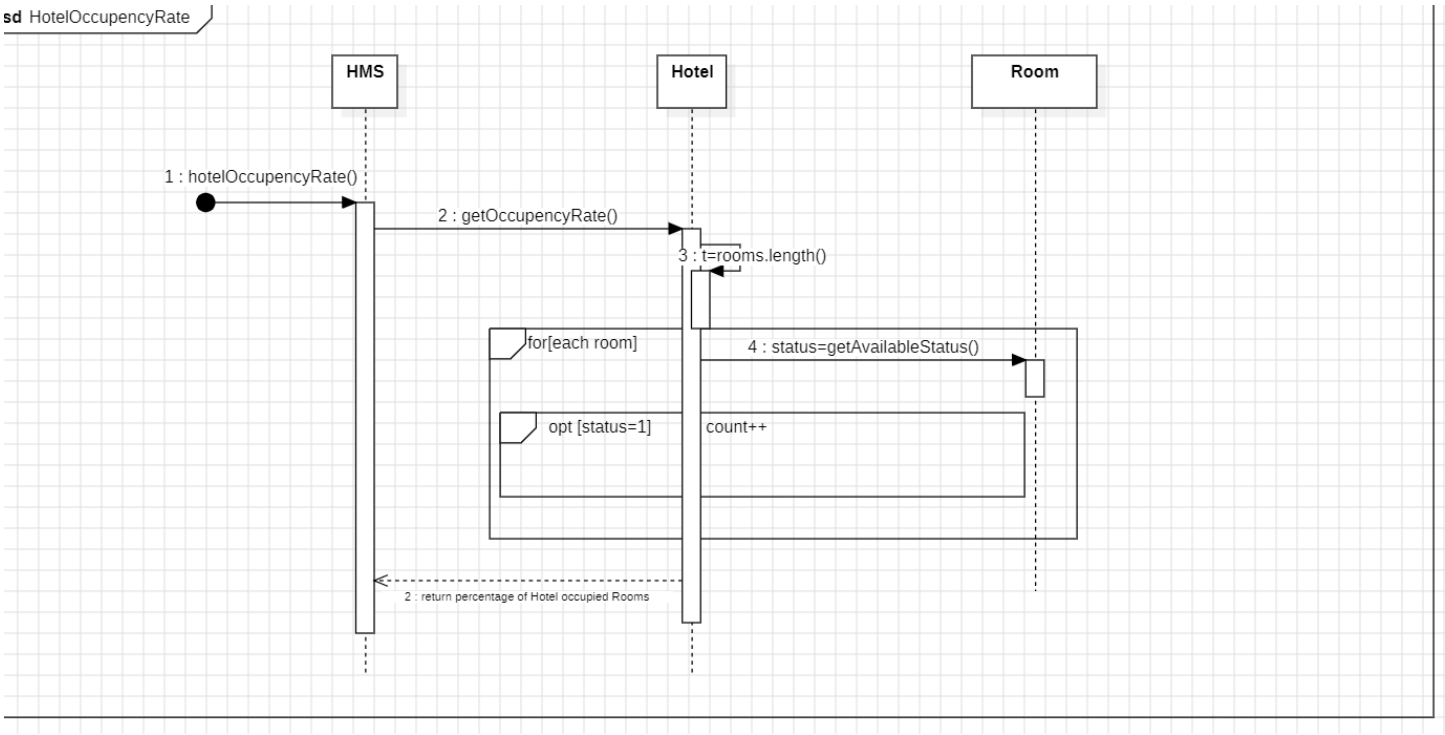
2. Manage Room

SSD:

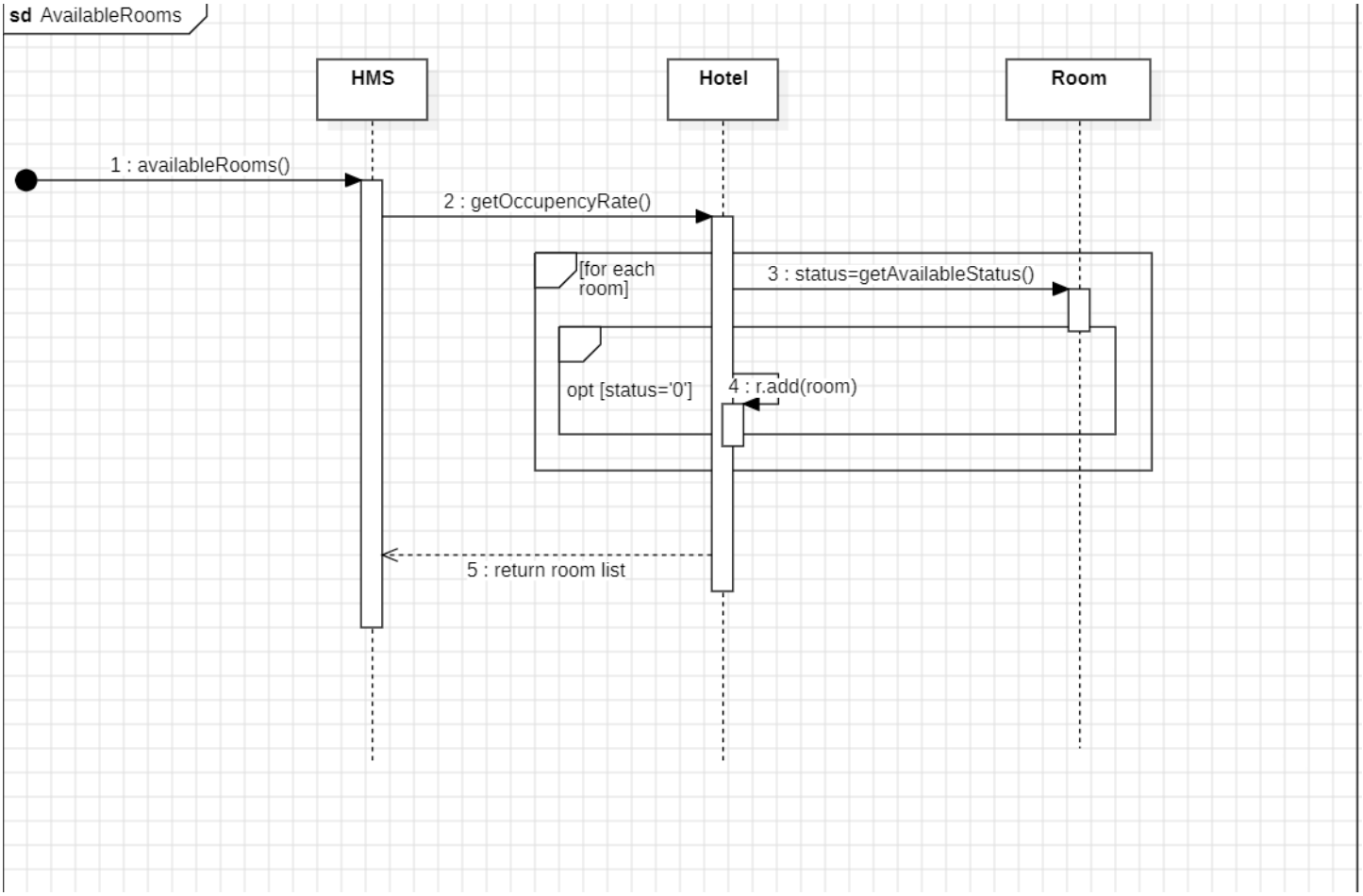

sd SequenceDiagram9
Lifeline1: Receptionist
HMS
1 : Login(name,password)
2 : Verify Credentials
3 : viewAvailableRooms()
4 : Display Room Details
5 : OverallOccupencyRate()
6 : return occupency Rate of Hotel
7 : roomOccupencyRate()
8 : return room with occupency rate
9 : revenueGenerated()
10 : return room with revenue

SDs

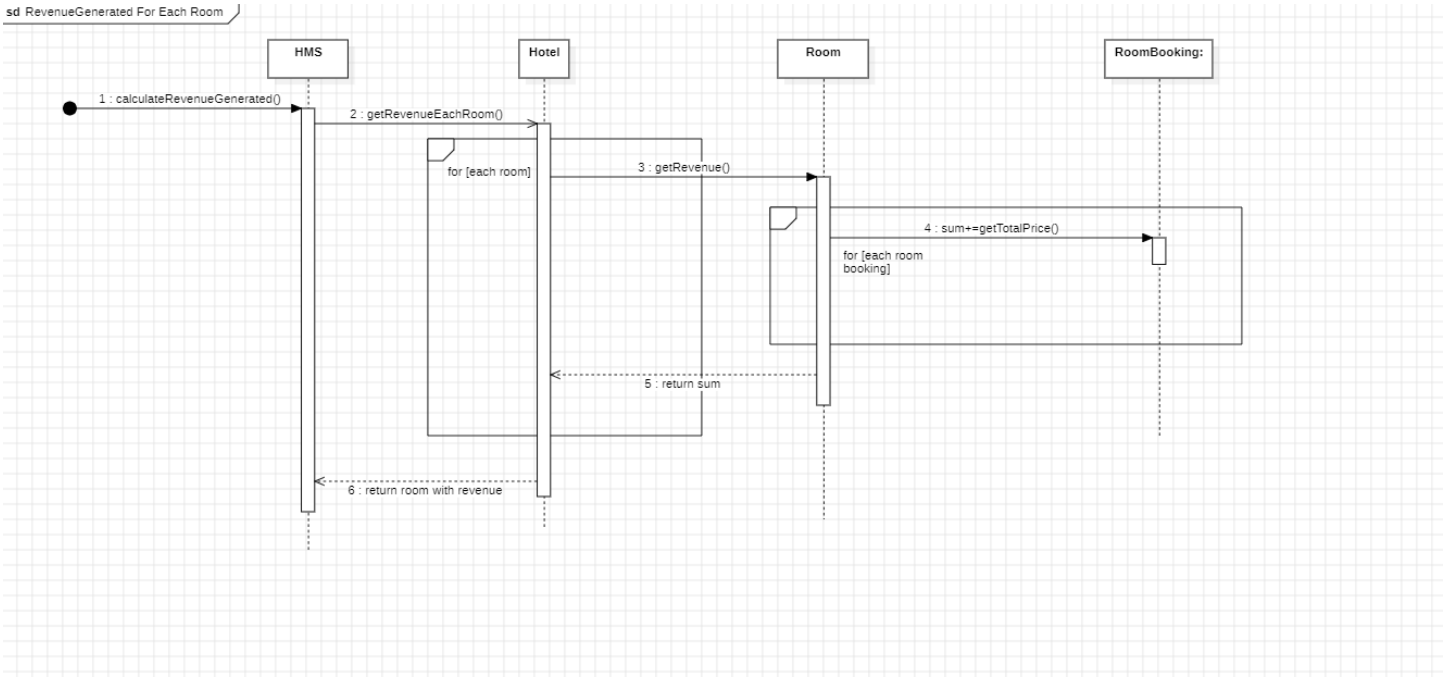
1:


sd HotelOccupencyRate
HMS
Hotel
Room
1 : hotelOccupencyRate()
2 : getOccupencyRate()
3 : t=rooms.length()
for[each room]
4 : status=getAvailableStatus()
opt [status=1]
count++
2 : return percentage of Hotel occupied Rooms

2:

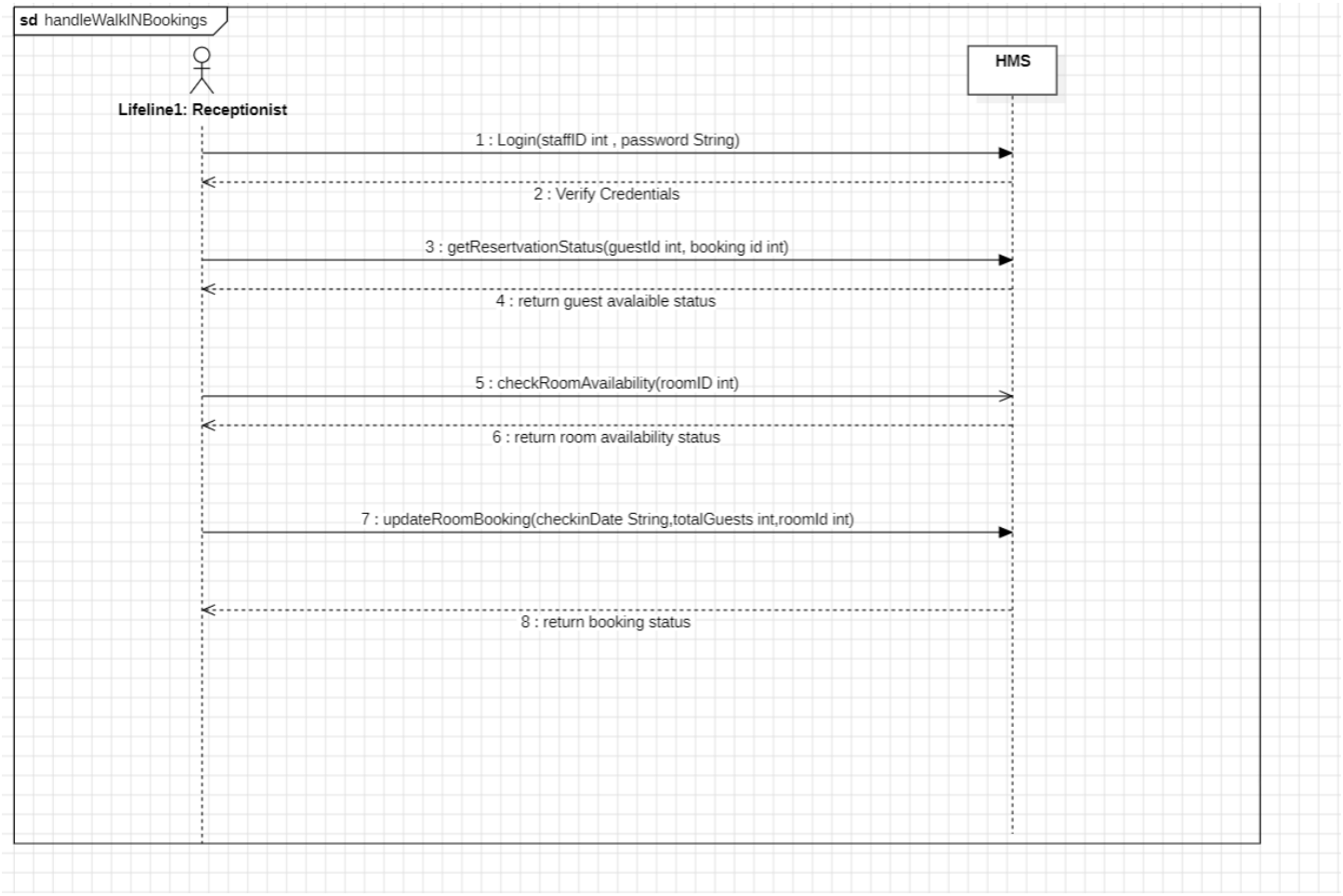

sd AvailableRooms
HMS
Hotel
Room
1 : availableRooms()
2 : getOccupencyRate()
[for each room]
3 : status=getAvailableStatus()
opt [status='0']
4 : r.add(room)
5 : return room list

3:



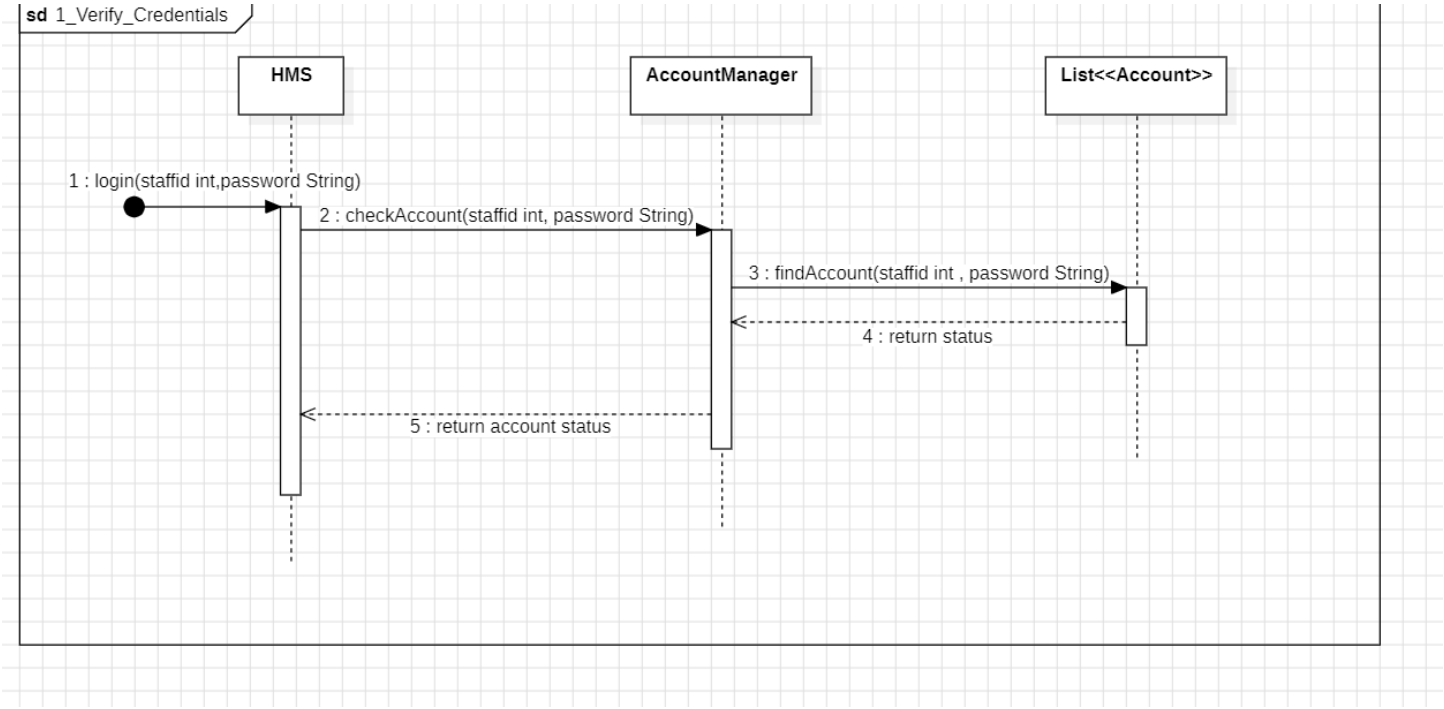
3. Handle Walk in bookings

SSD

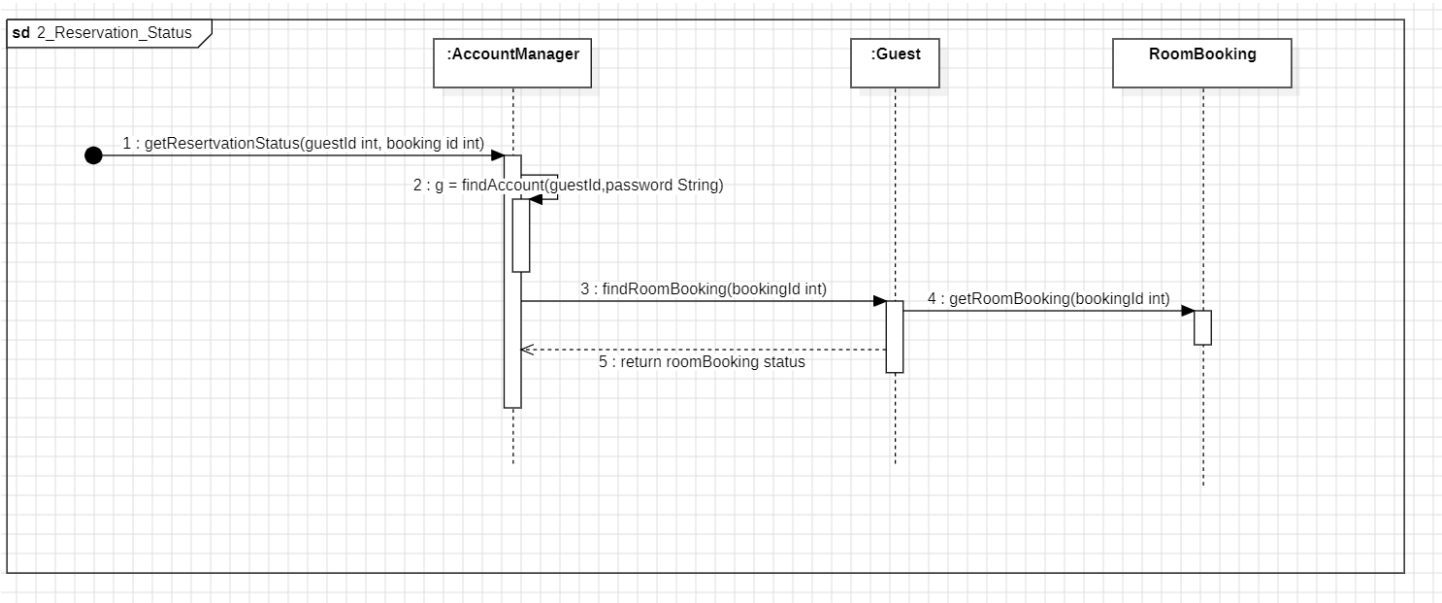

sd handleWalkINBookings
Lifeline1: Receptionist
HMS
1 : Login(staffID int , password String)
2 : Verify Credentials
3 : getResertvationStatus(guestId int, booking id int)
4 : return guest avalaible status
5 : checkRoomAvailability(roomID int)
6 : return room availability status
7 : updateRoomBooking(checkinDate String,totalGuests int,roomId int)
8 : return booking status

SDs

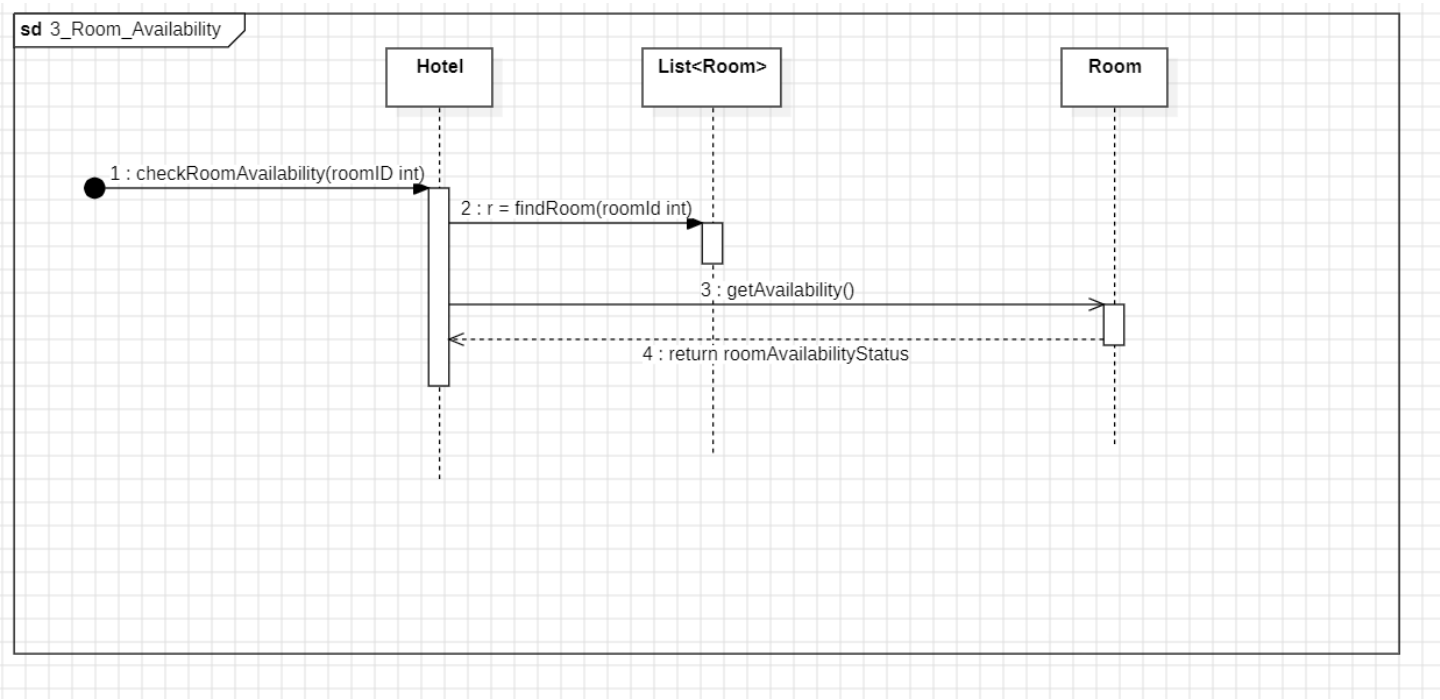
1.

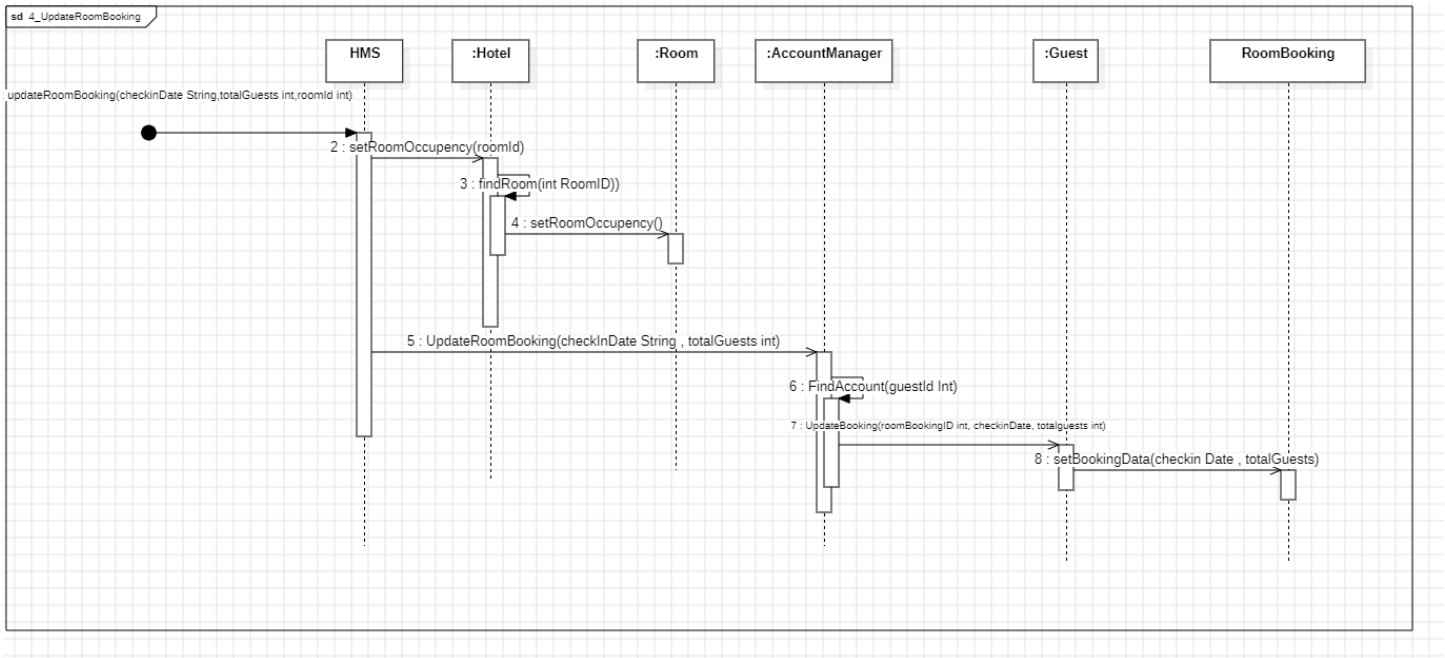

sd 1_Verify_Credentials
HMS
AccountManager
List<<Account>>
1 : login(staffid int,password String)
2 : checkAccount(staffid int, password String)
3 : findAccount(staffid int , password String)
4 : return status
5 : return account status

2.


sd 2_Reservation_Status
:AccountManager
:Guest
RoomBooking
1 : getResertvationStatus(guestId int, booking id int)
2 : g = findAccount(guestId,password String)
3 : findRoomBooking(bookingId int)
4 : getRoomBooking(bookingId int)
5 : return roomBooking status

3.

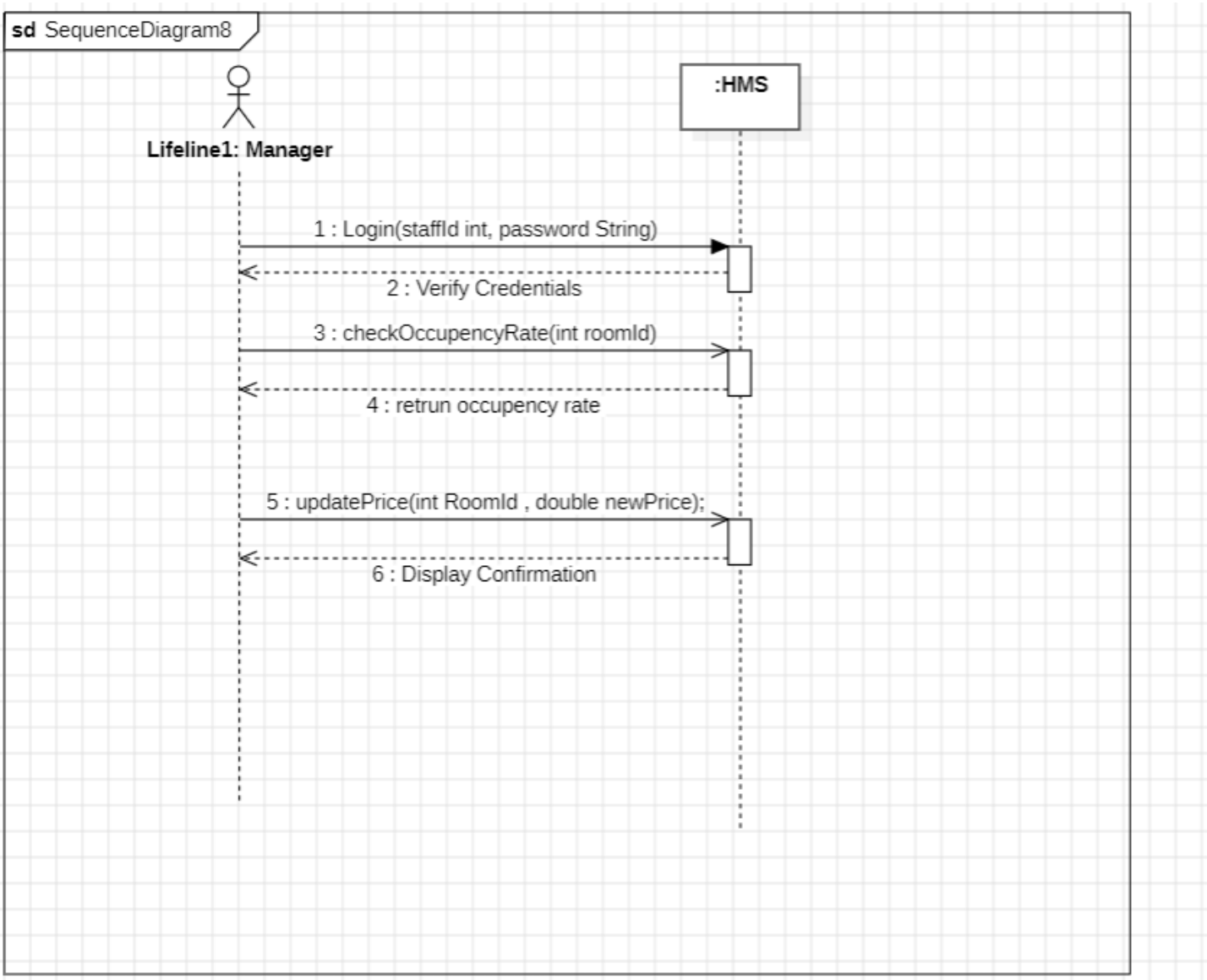

sd 3_Room_Availability
Hotel
List<Room>
Room
1 : checkRoomAvailability(roomID int)
2 : r = findRoom(roomId int)
3 : getAvailability()
4 : return roomAvailabilityStatus


sd 4_UpdateRoomBooking
HMS
:Hotel
:Room
:AccountManager
:Guest
RoomBooking
updateRoomBooking(checkinDate String,totalGuests int,roomId int)
2 : setRoomOccupency(roomId)
3 : findRoom(int RoomID))
4 : setRoomOccupency()
5 : UpdateRoomBooking(checkInDate String , totalGuests int)
6 : FindAccount(guestId Int)
7 : UpdateBooking(roomBookingID int, checkinDate, totalguests int)
8 : setBookingData(checkin Date , totalGuests)

4.

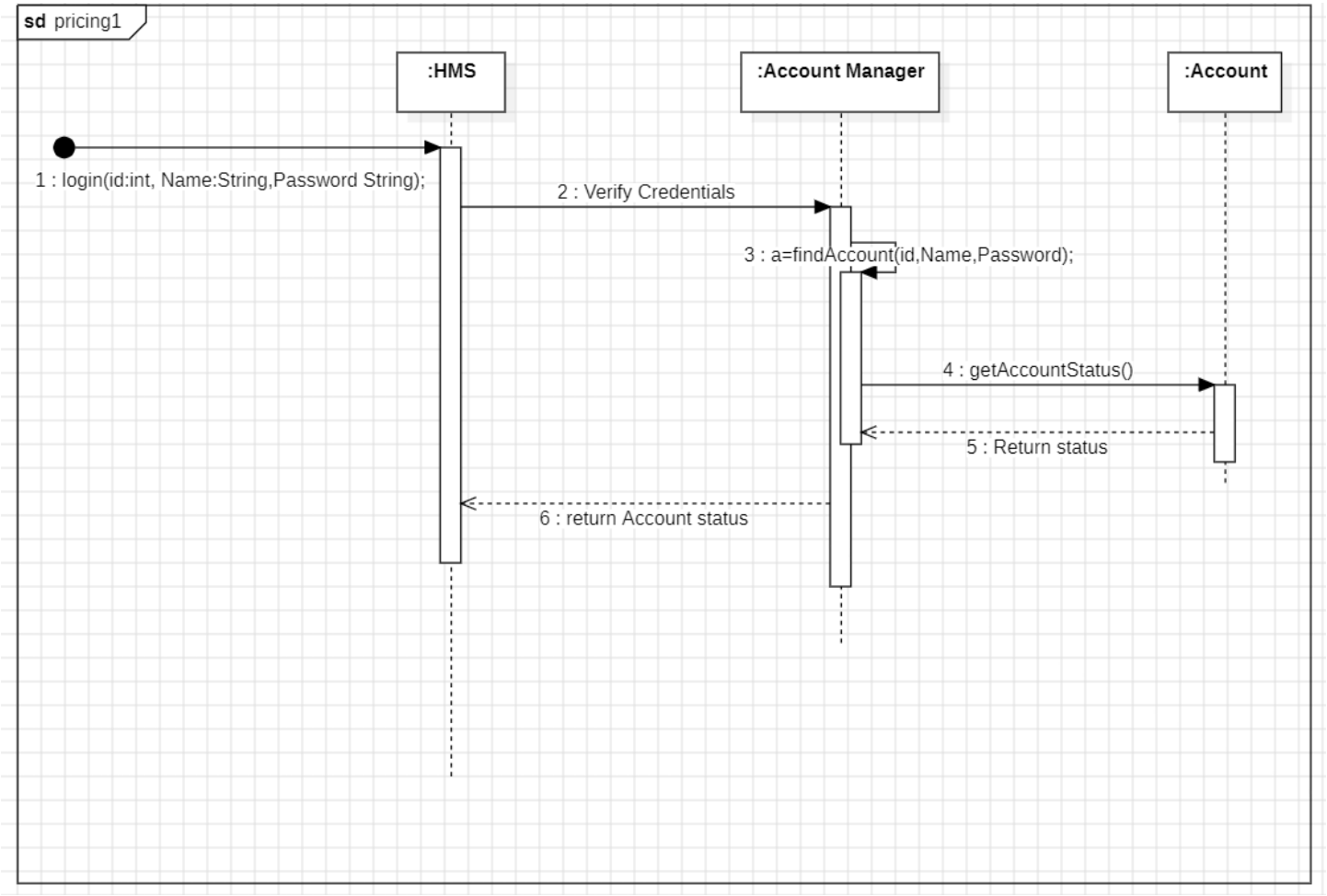
4. Manage Price

SSD

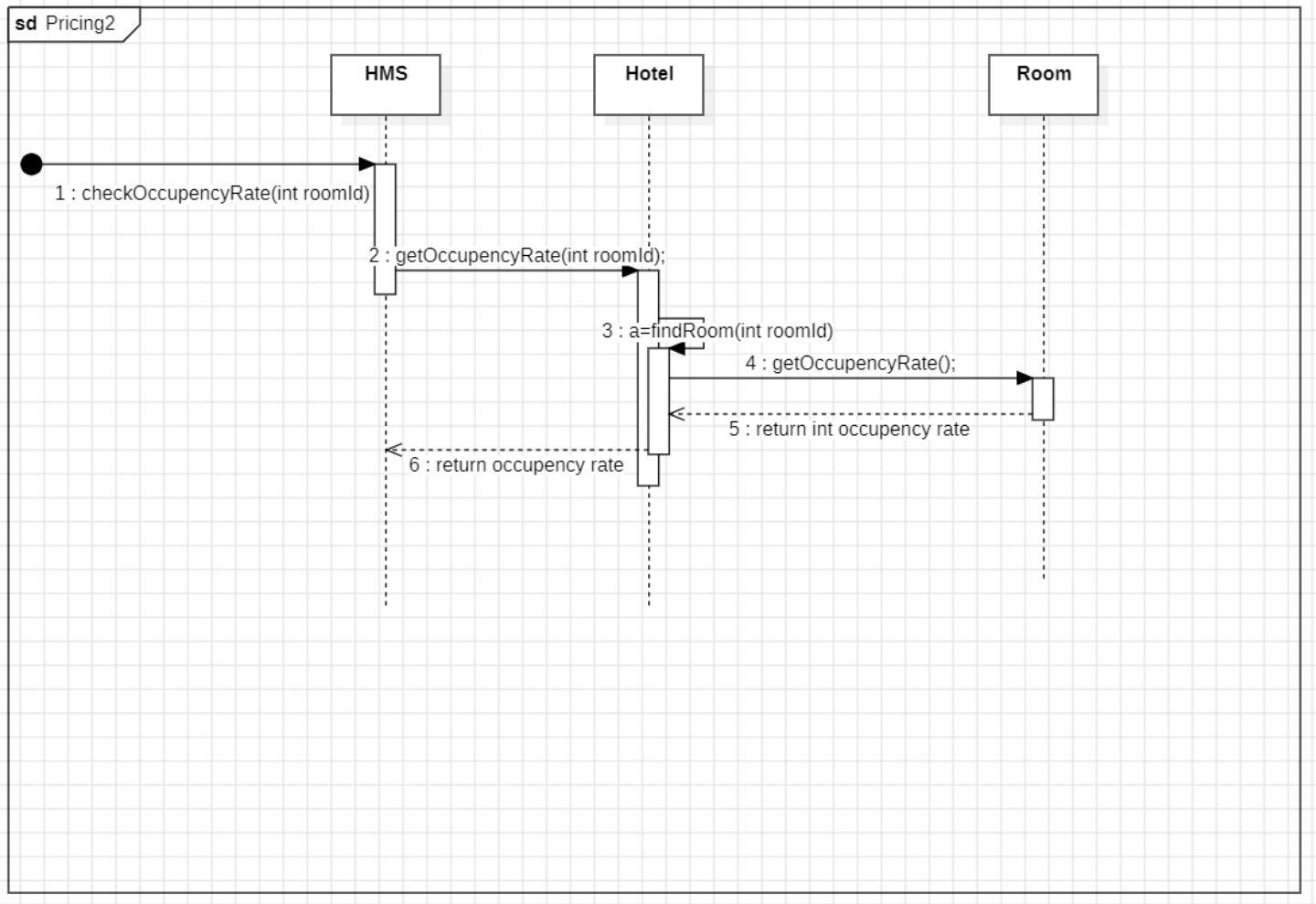


SSDs

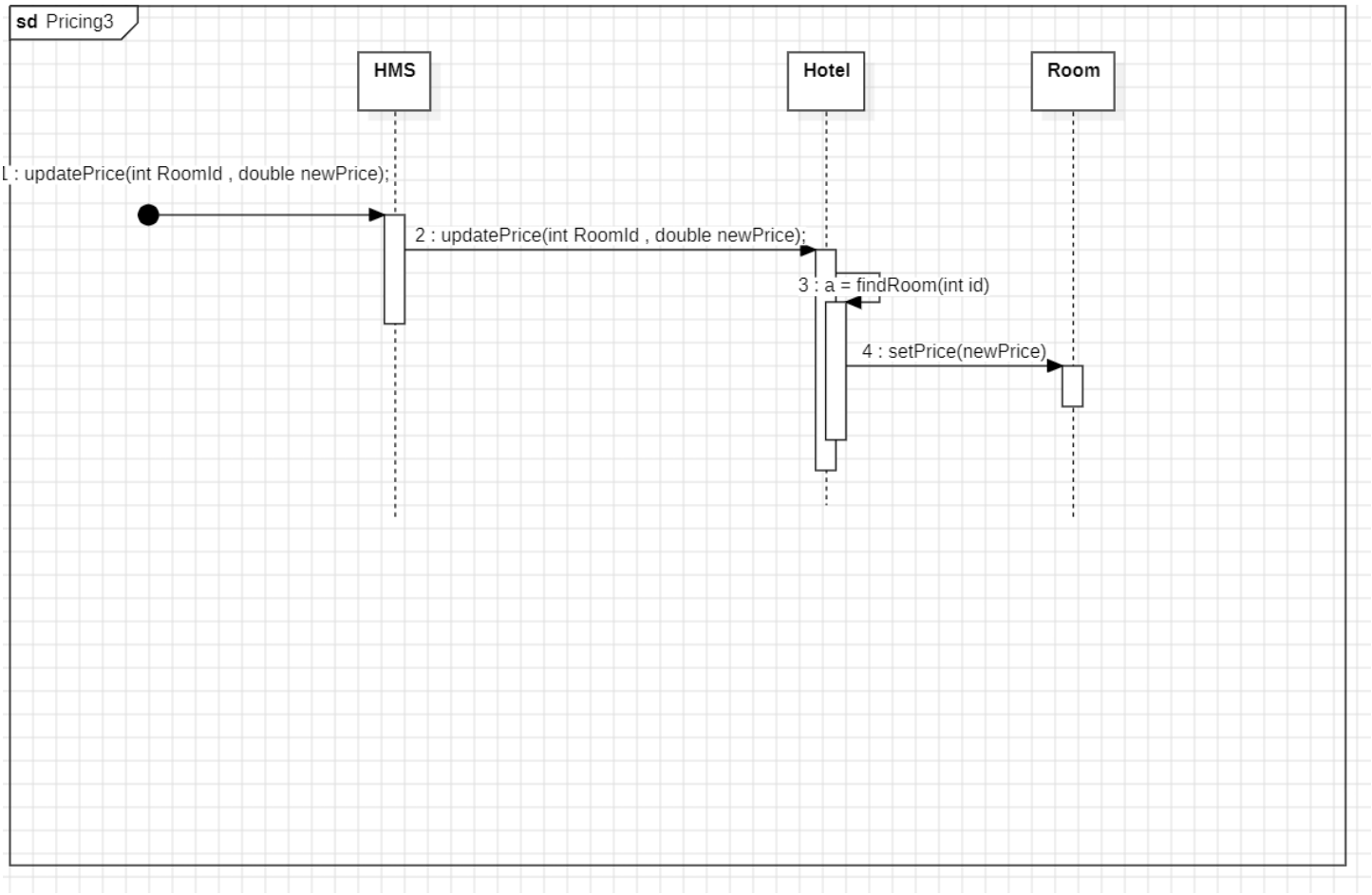
1.


sd pricing1
:HMS
:Account Manager
:Account
1 : login(id:int, Name:String,Password String);
2 : Verify Credentials
3 : a=findAccount(id,Name,Password);
4 : getAccountStatus()
5 : Return status
6 : return Account status

2.

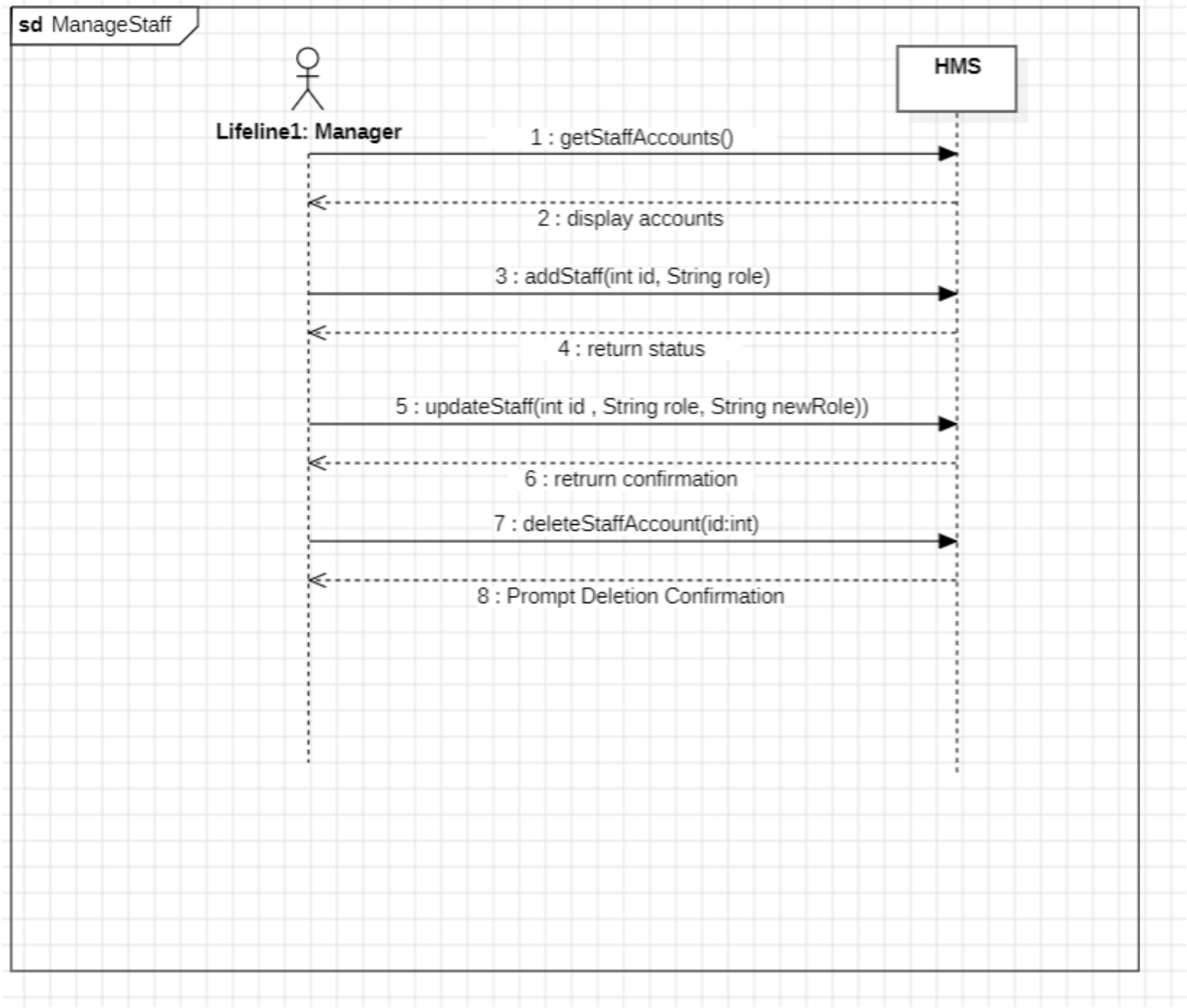

sd Pricing2
HMS
Hotel
Room
1 : checkOccupencyRate(int roomId)
2 : getOccupencyRate(int roomId);
3 : a=findRoom(int roomId)
4 : getOccupencyRate();
5 : return int occupency rate
6 : return occupency rate

3.

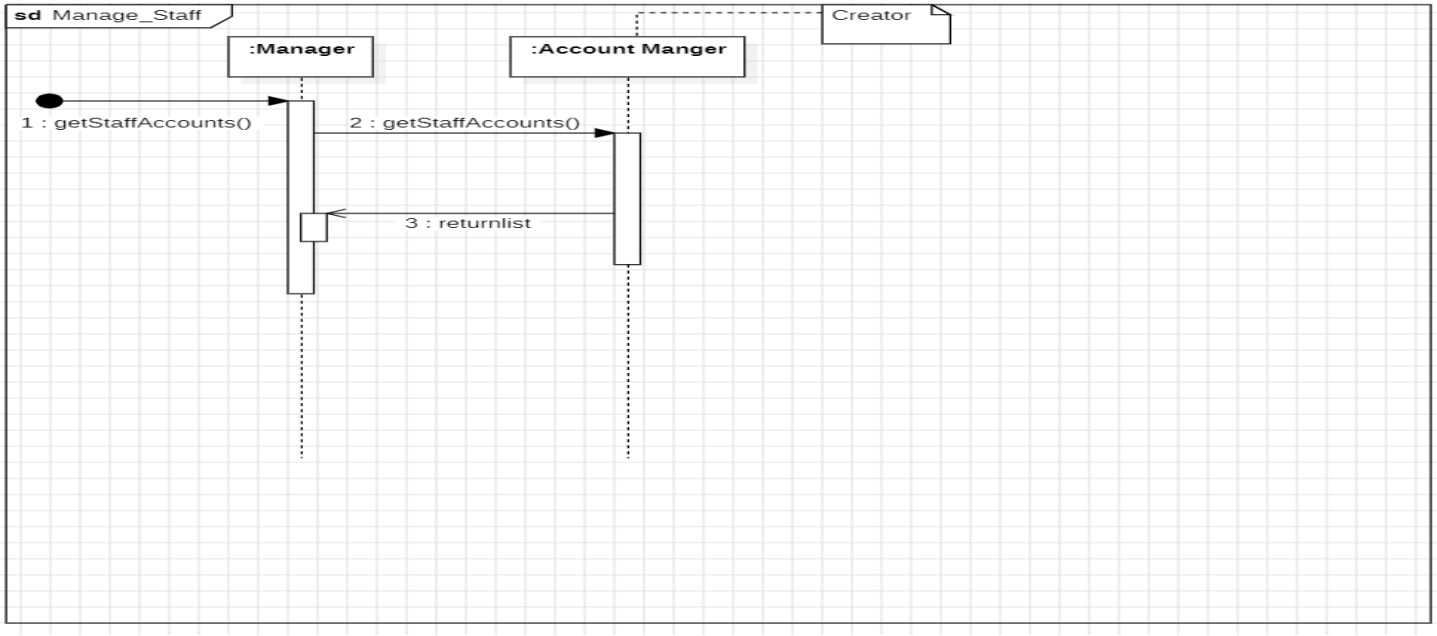


5. Manage Staff

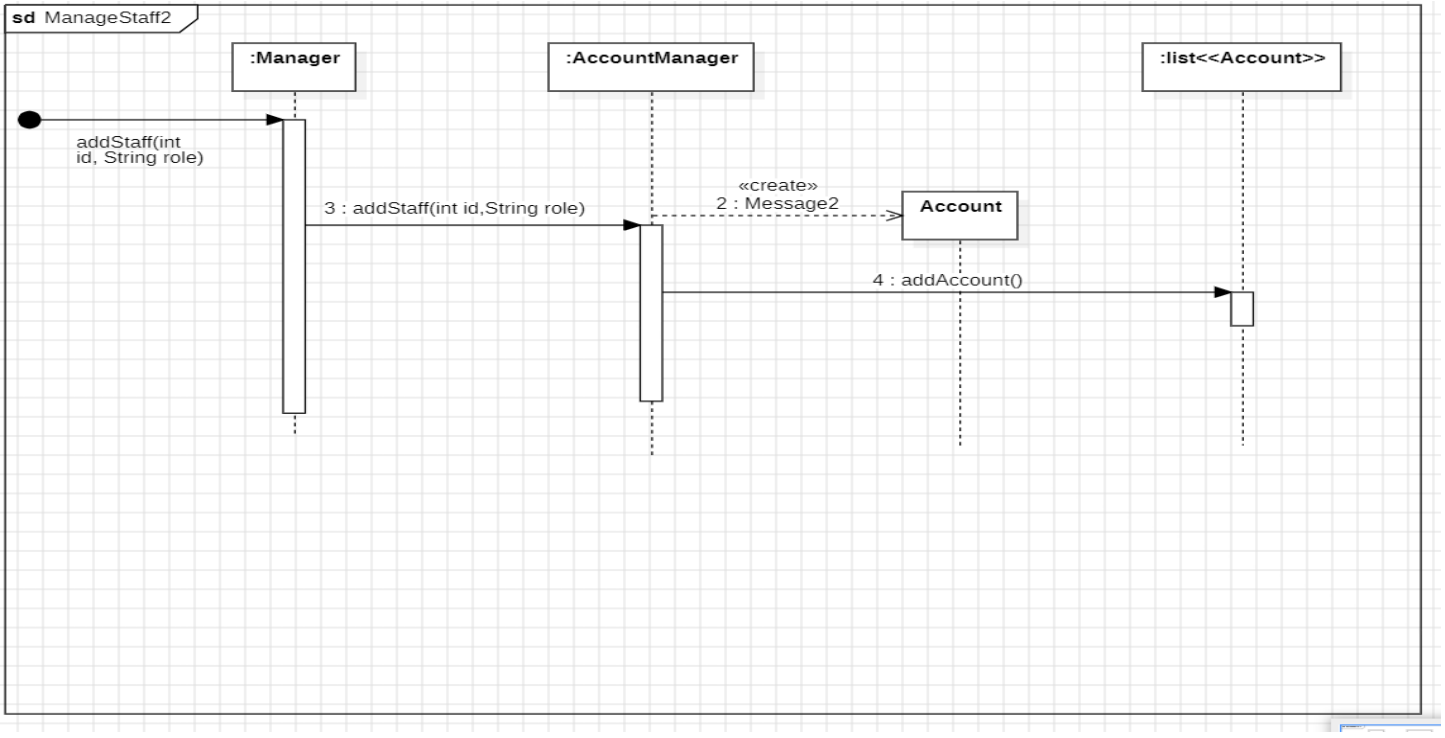
SSD:


sd ManageStaff
Lifeline1: Manager
HMS
1 : getStaffAccounts()
2 : display accounts
3 : addStaff(int id, String role)
4 : return status
5 : updateStaff(int id , String role, String newRole))
6 : retrurn confirmation
7 : deleteStaffAccount(id:int)
8 : Prompt Deletion Confirmation

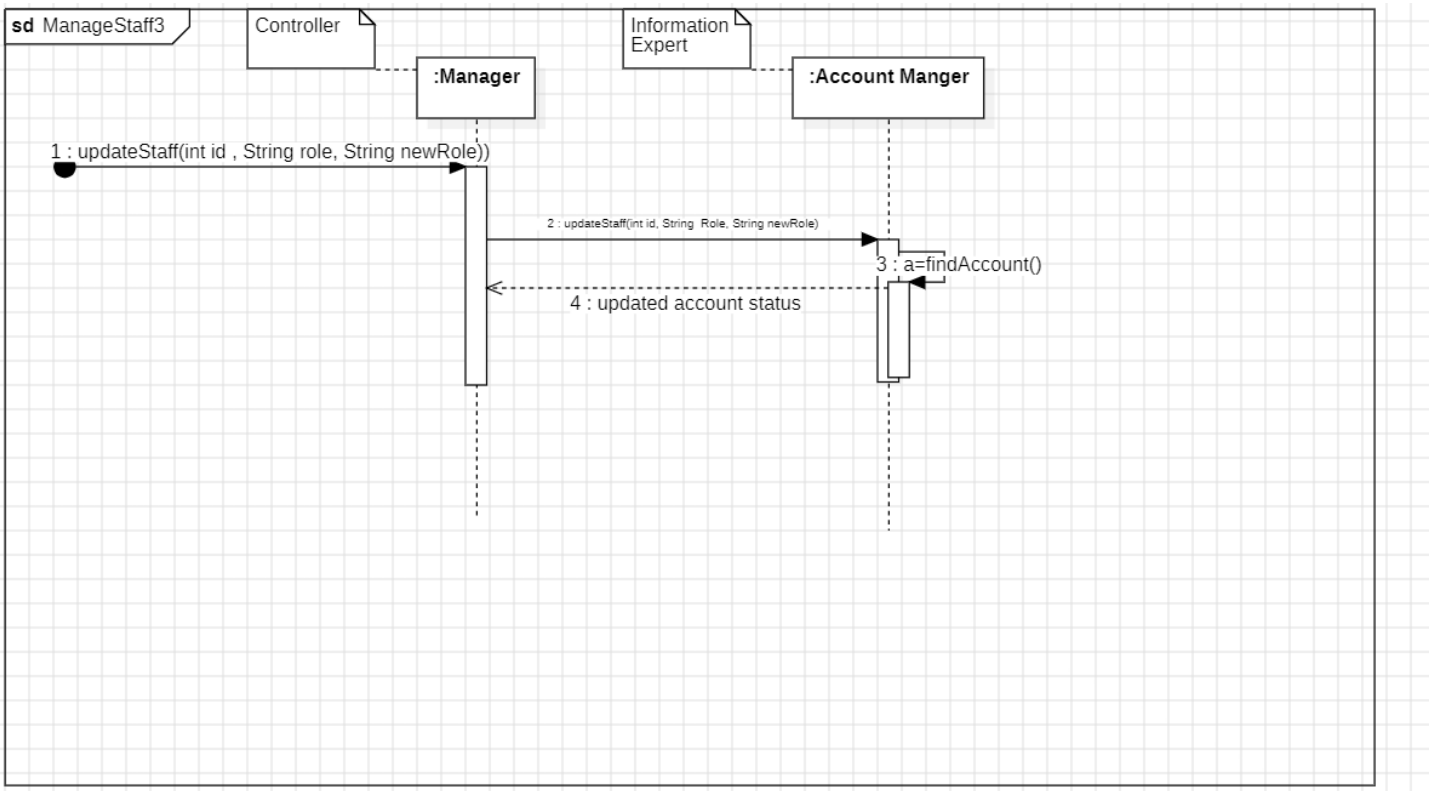
SDs
1:


sd Manage_Staff
:Manager
:Account Manger
Creator
1 : getStaffAccounts()
2 : getStaffAccounts()
3 : returnlist

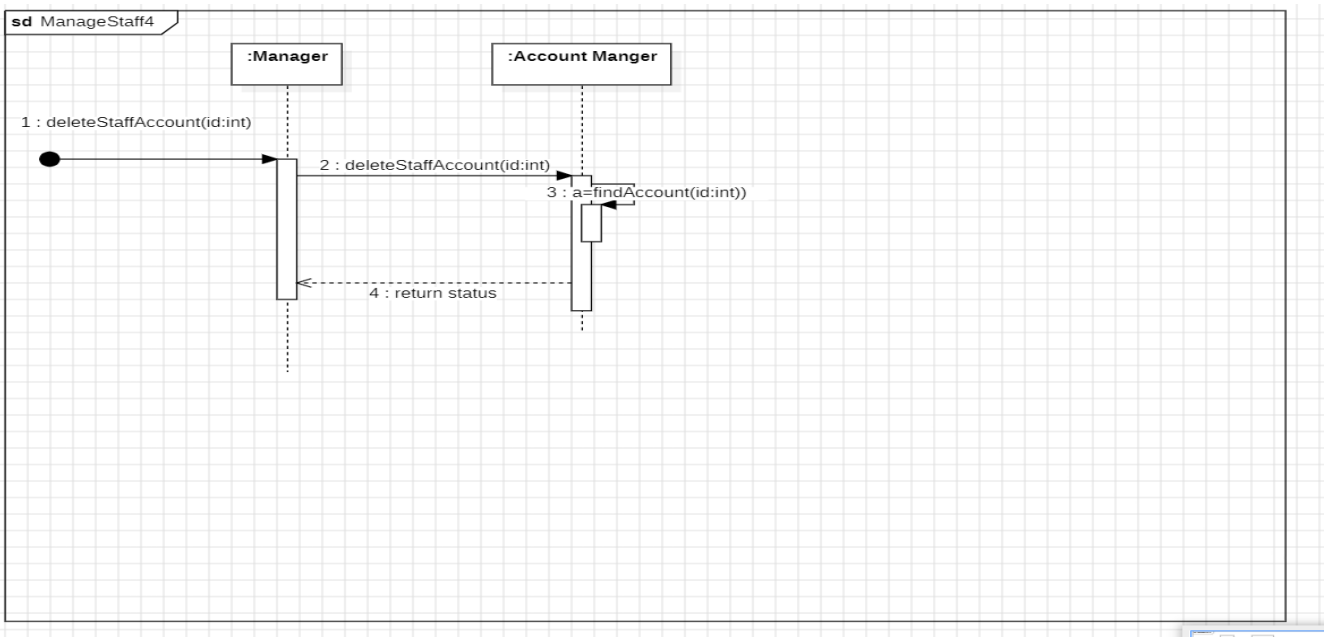
2.


sd ManageStaff2
:Manager
:AccountManager
:list<<Account>>
addStaff(int
id, String role)
3 : addStaff(int id,String role)
«create»
2 : Message2
Account
4 : addAccount()

3.

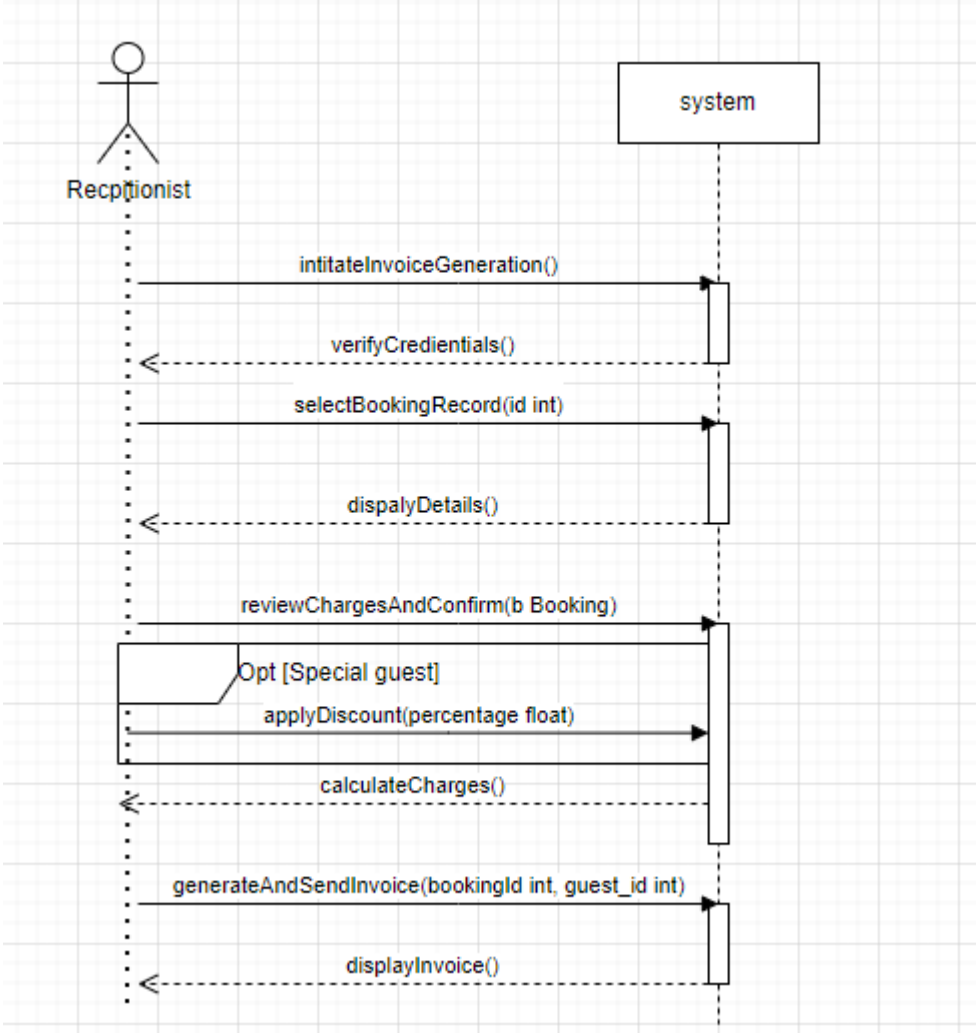

sd ManageStaff3
Controller
:Manager
Information Expert
:Account Manger
1 : updateStaff(int id , String role, String newRole))
2 : updateStaff(int id, String Role, String newRole)
3 : a=findAccount()
4 : updated account status

4.

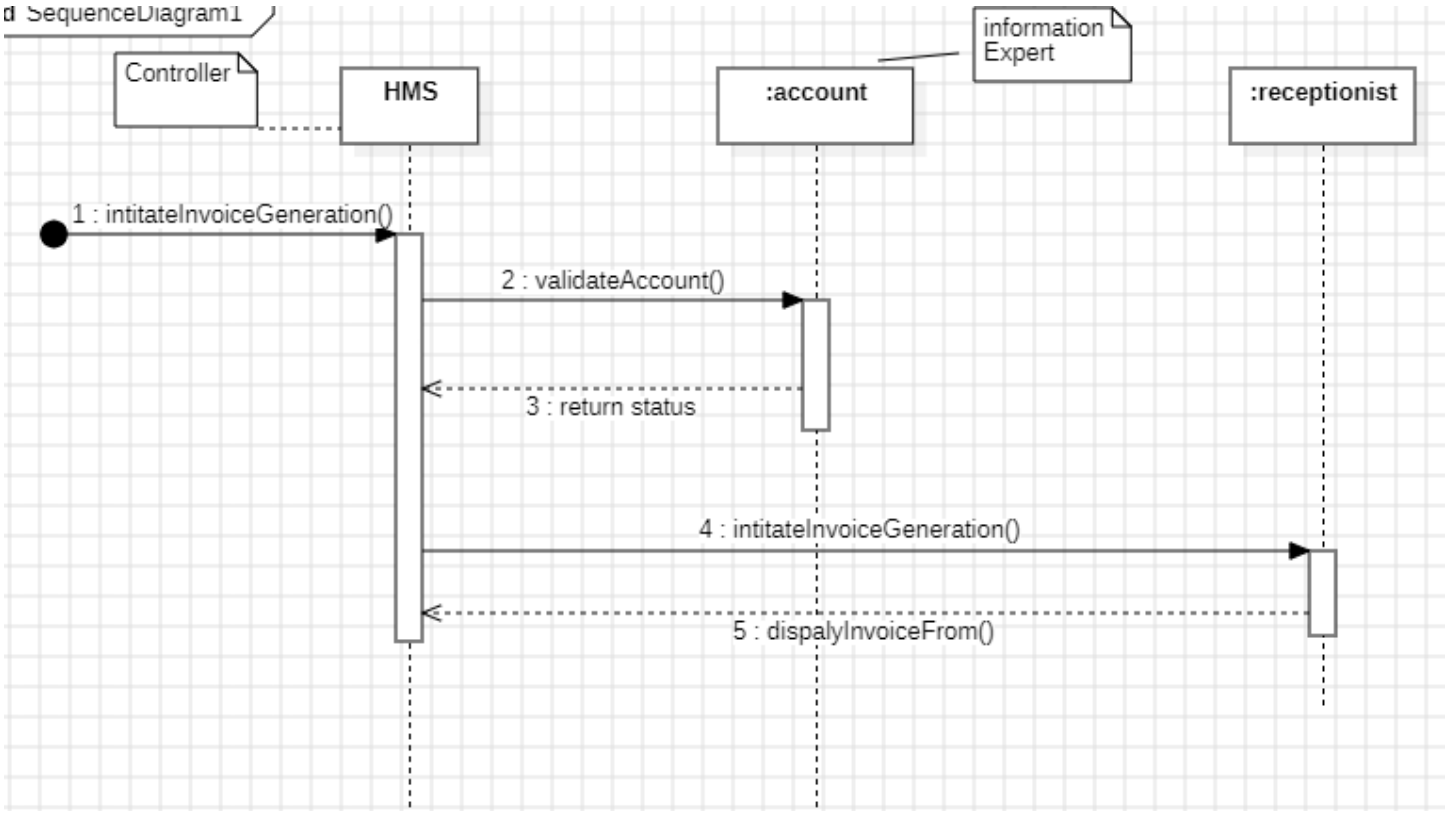


6. Generate and Send Invoice

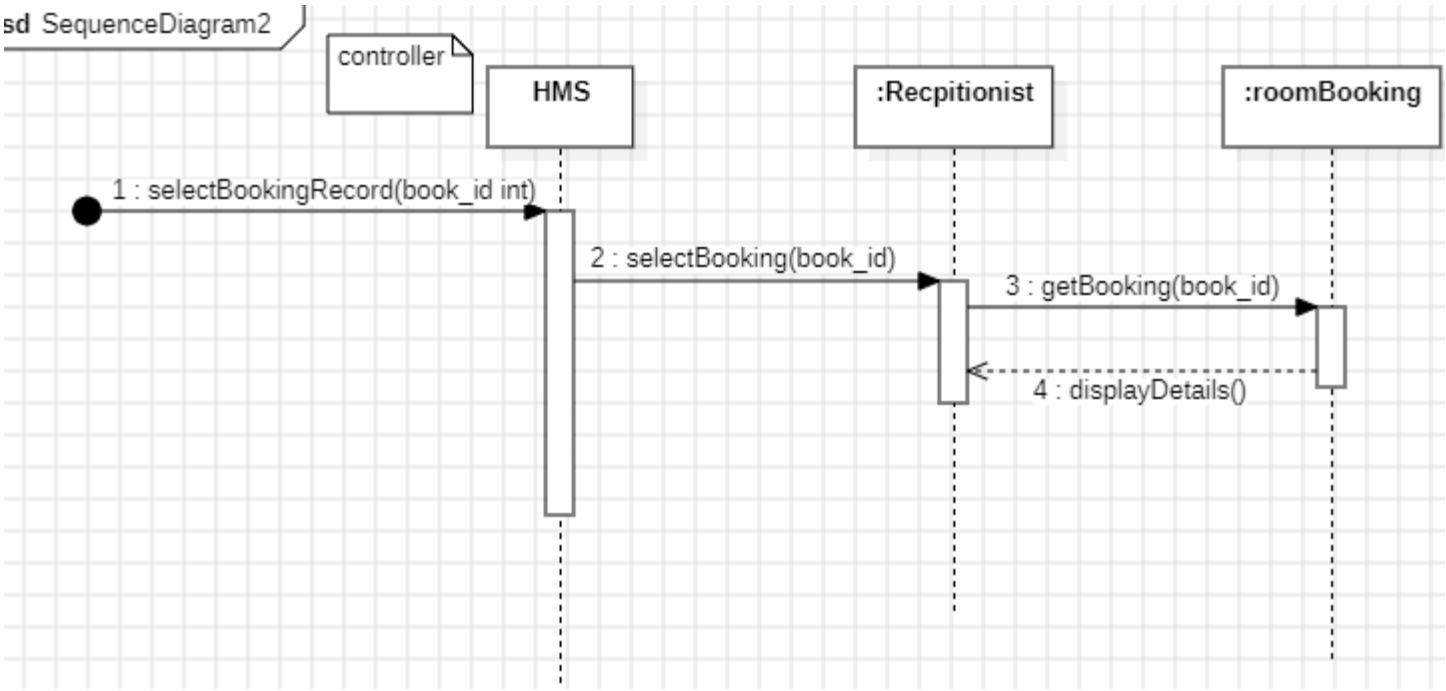
SSD


system
Recptionist
intitateInvoiceGeneration()
verifyCredientials()
selectBookingRecord(id int)
dispalyDetails()
reviewChargesAndConfirm(b Booking)
Opt [Special guest]
applyDiscount(percentage float)
calculateCharges()
generateAndSendInvoice(bookingId int, guest_id int)
displayInvoice()

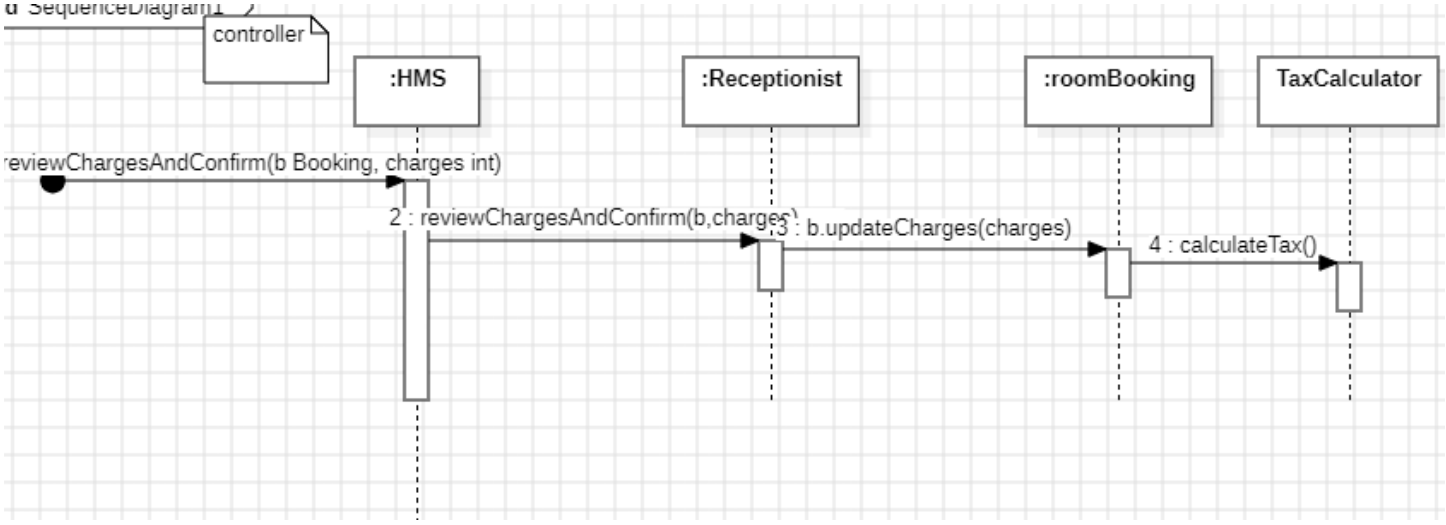
SD 1


d SequenceDiagram1
Controller
HMS
:account
information Expert
:receptionist
1 : intitateInvoiceGeneration()
2 : validateAccount()
3 : return status
4 : intitateInvoiceGeneration()
5 : dispalyInvoiceFrom()

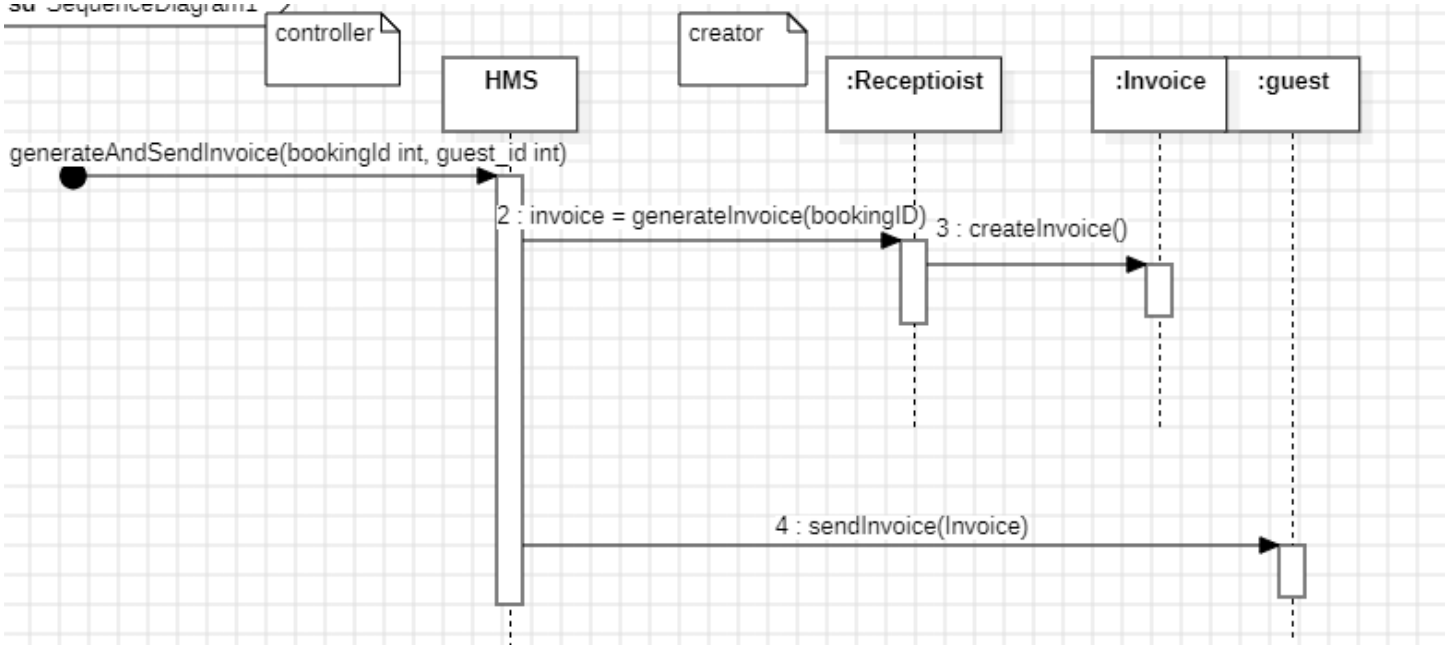
2


sd SequenceDiagram2
controller
HMS
:Recpitionist
:roomBooking
1 : selectBookingRecord(book_id int)
2 : selectBooking(book_id)
3 : getBooking(book_id)
4 : displayDetails()

3

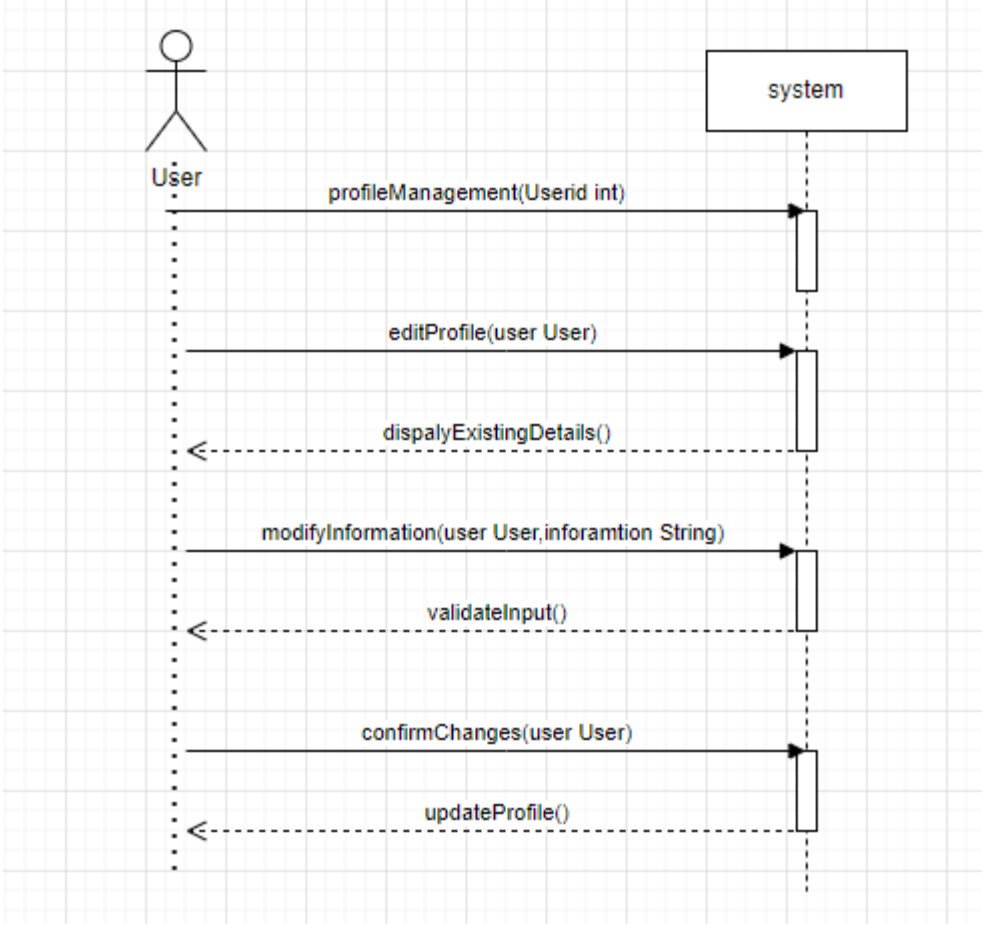


4

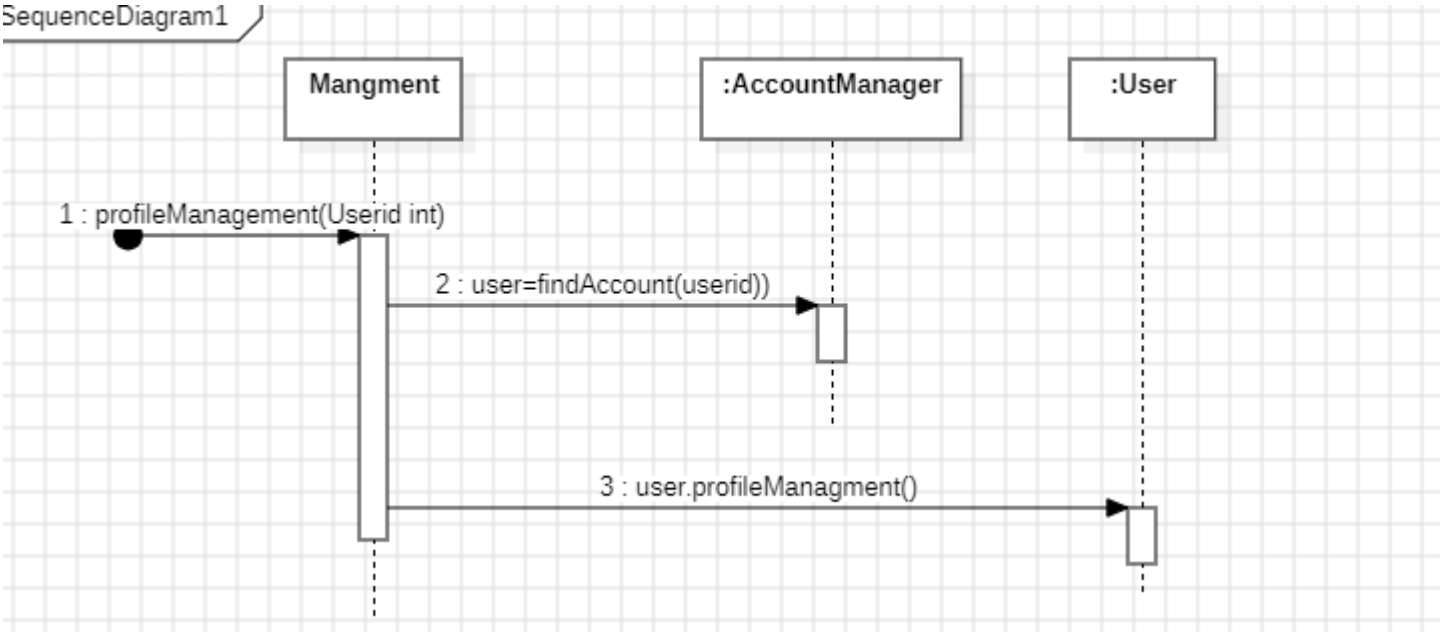


7. Manage Personal Profile

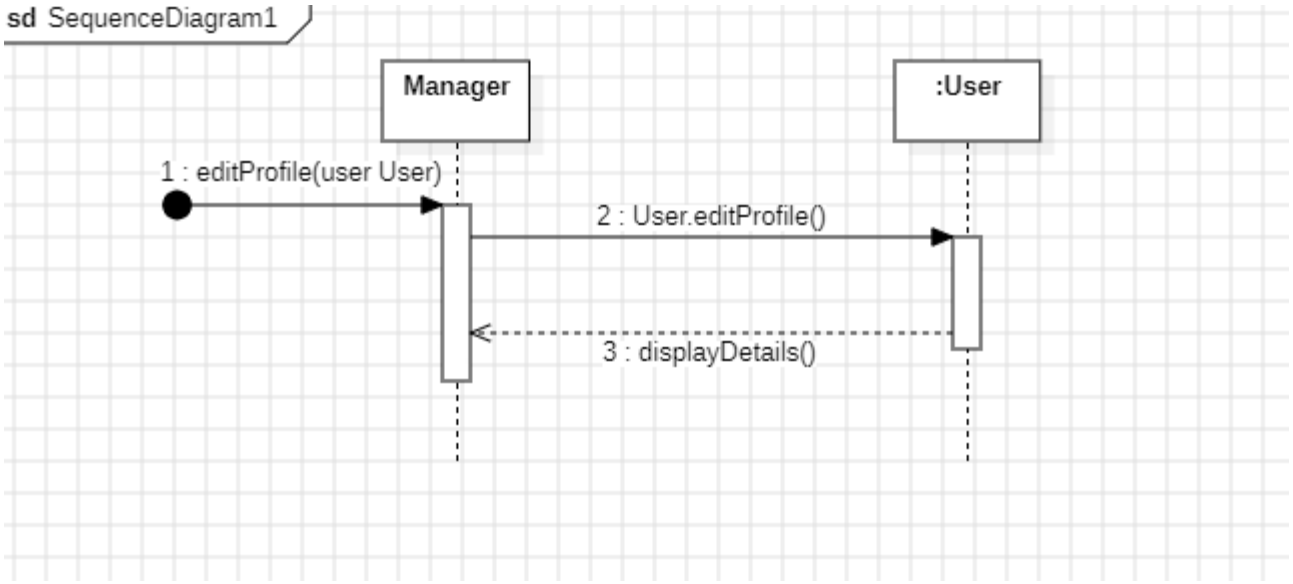
SSD


system
User
profileManagement(Userid int)
editProfile(user User)
dispalyExistingDetails()
modifyInformation(user User,inforamtion String)
validateInput()
confirmChanges(user User)
updateProfile()

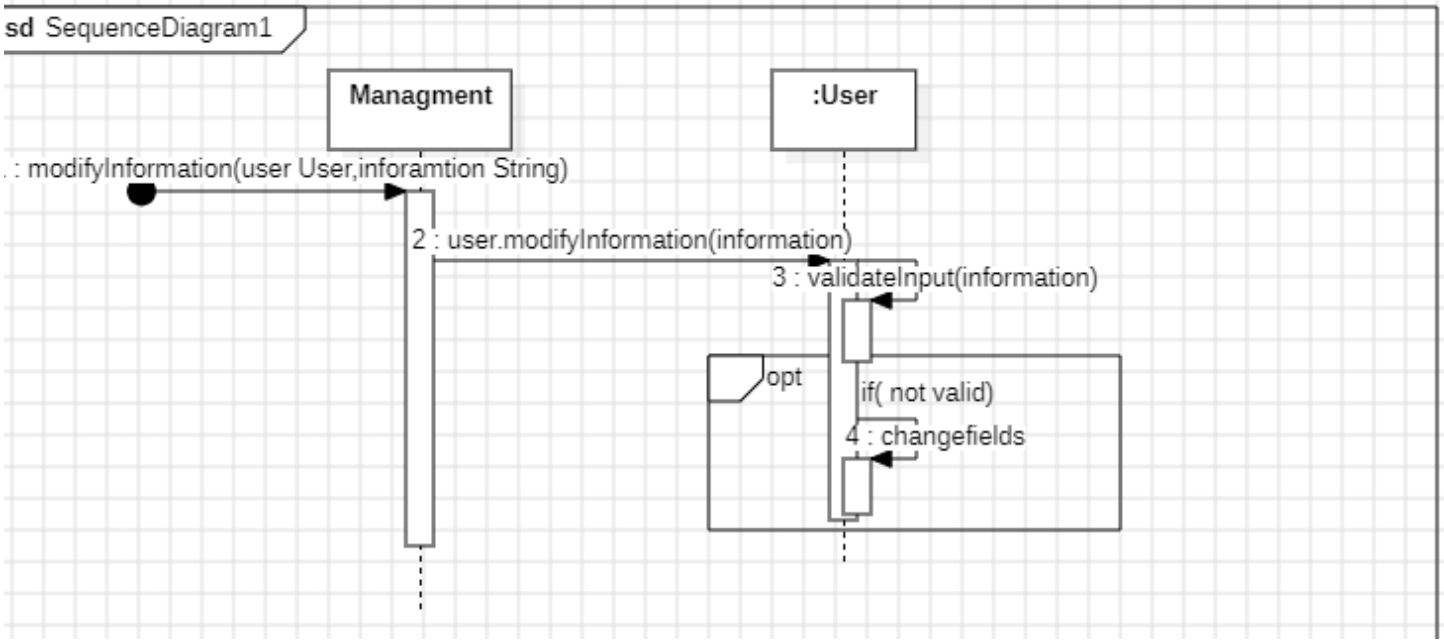
SD 1


SequenceDiagram1
Mangment
:AccountManager
:User
1 : profileManagement(Userid int)
2 : user=findAccount(userid))
3 : user.profileManagment()

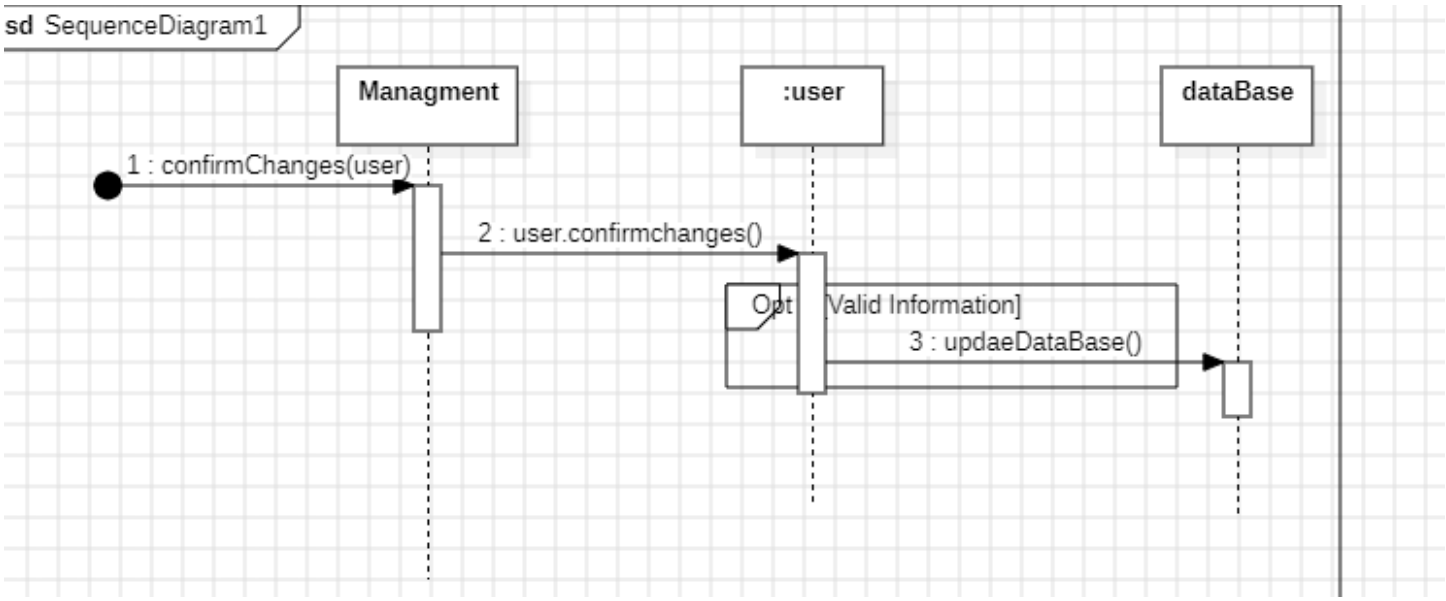
2


sd SequenceDiagram1
Manager
:User
1 : editProfile(user User)
2 : User.editProfile()
3 : displayDetails()

3

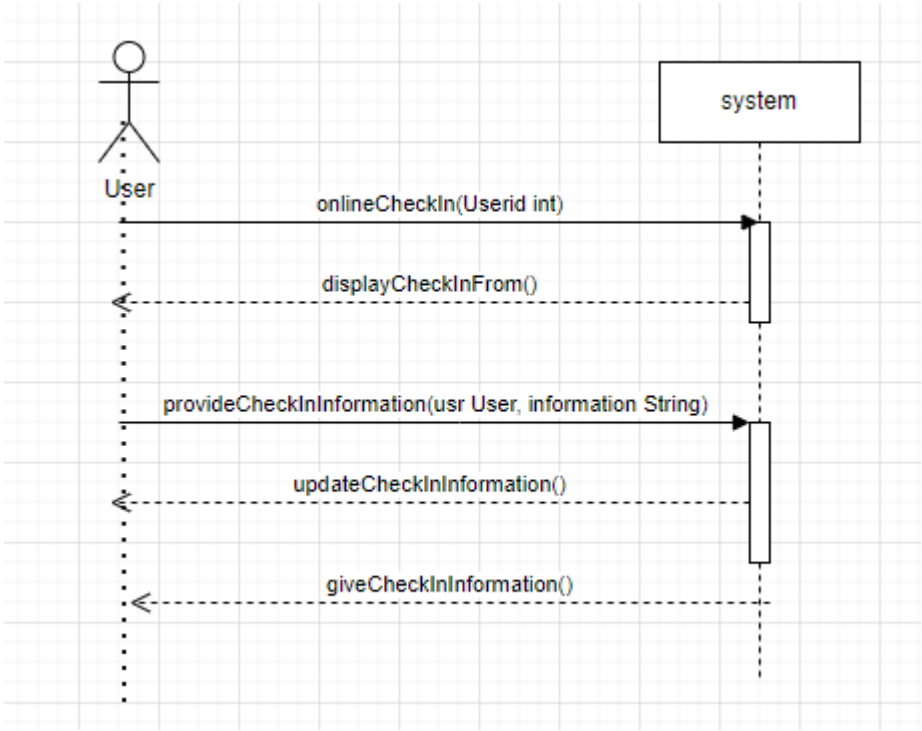

sd SequenceDiagram1
Managment
:User
: modifyInformation(user User,inforamtion String)
2 : user.modifyInformation(information)
3 : validateInput(information)
opt
if(not valid)
4 : changefields

4

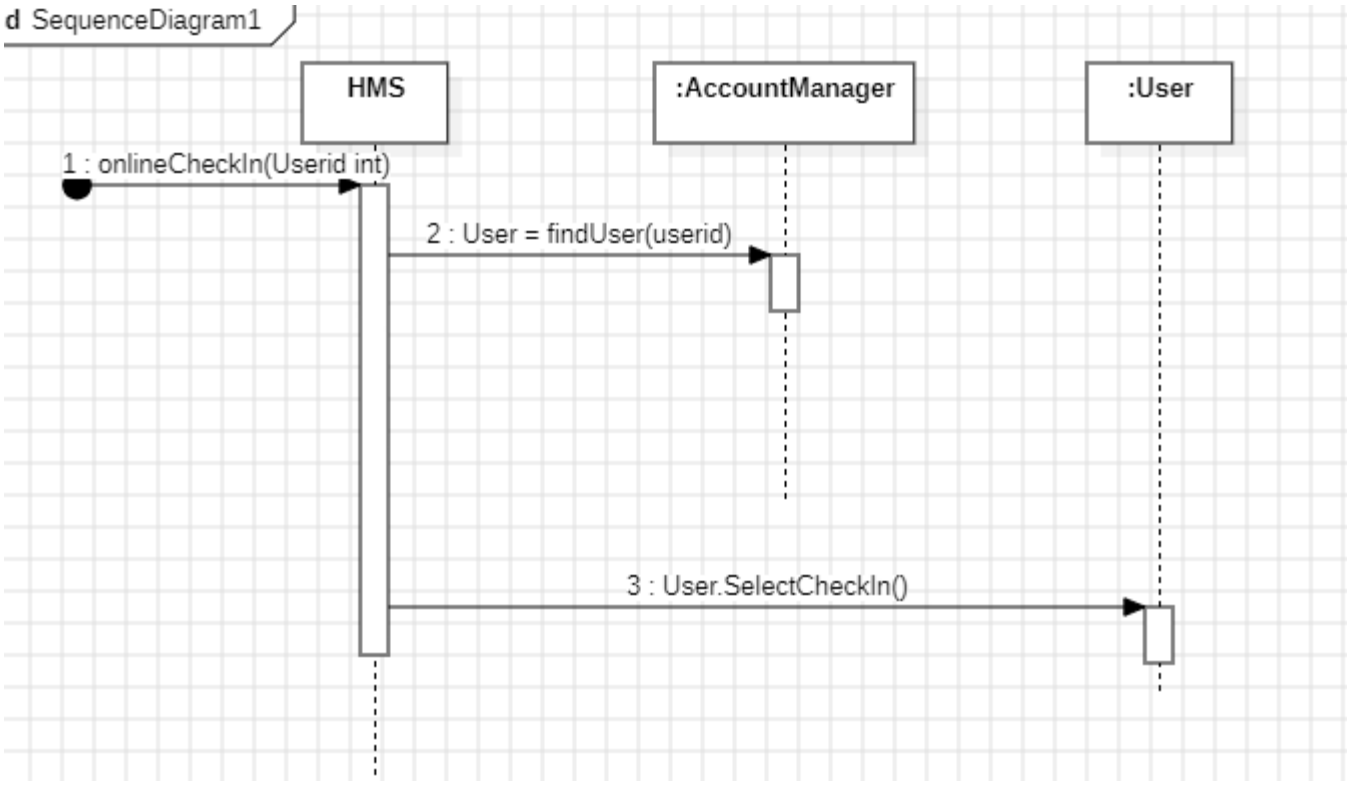


8. Online Check-in

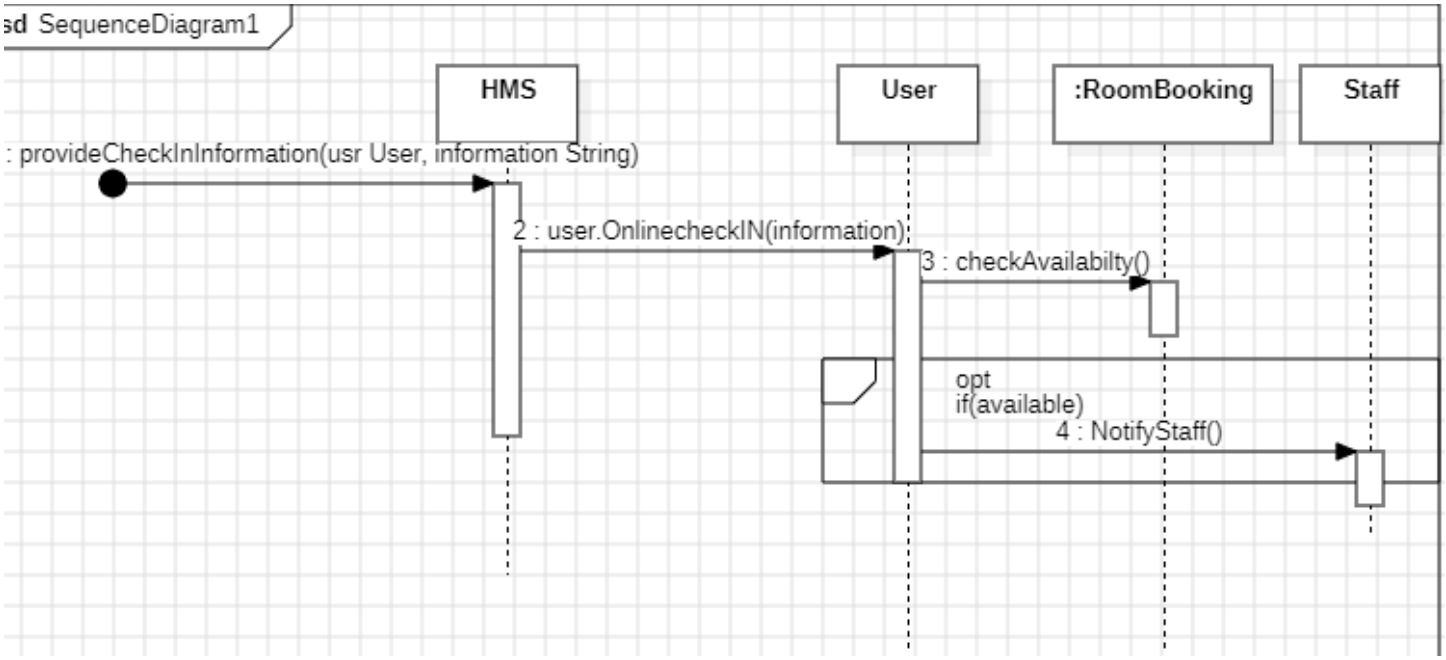
SSD



SD 1

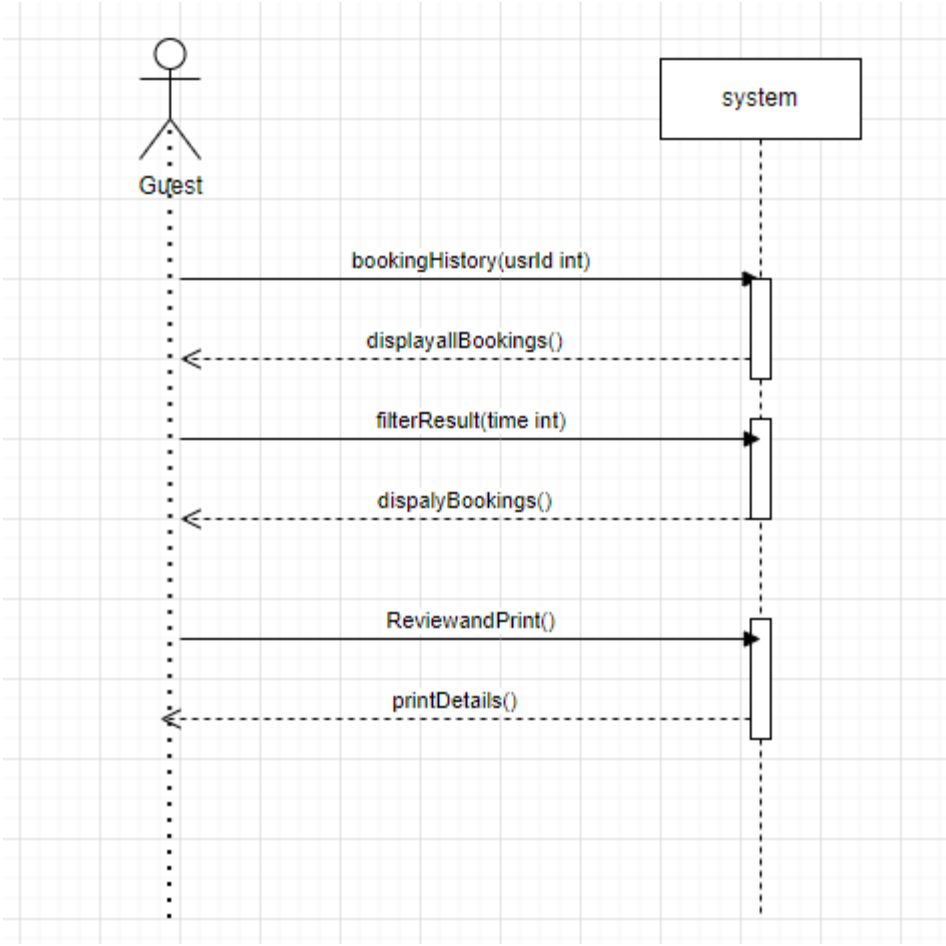


2

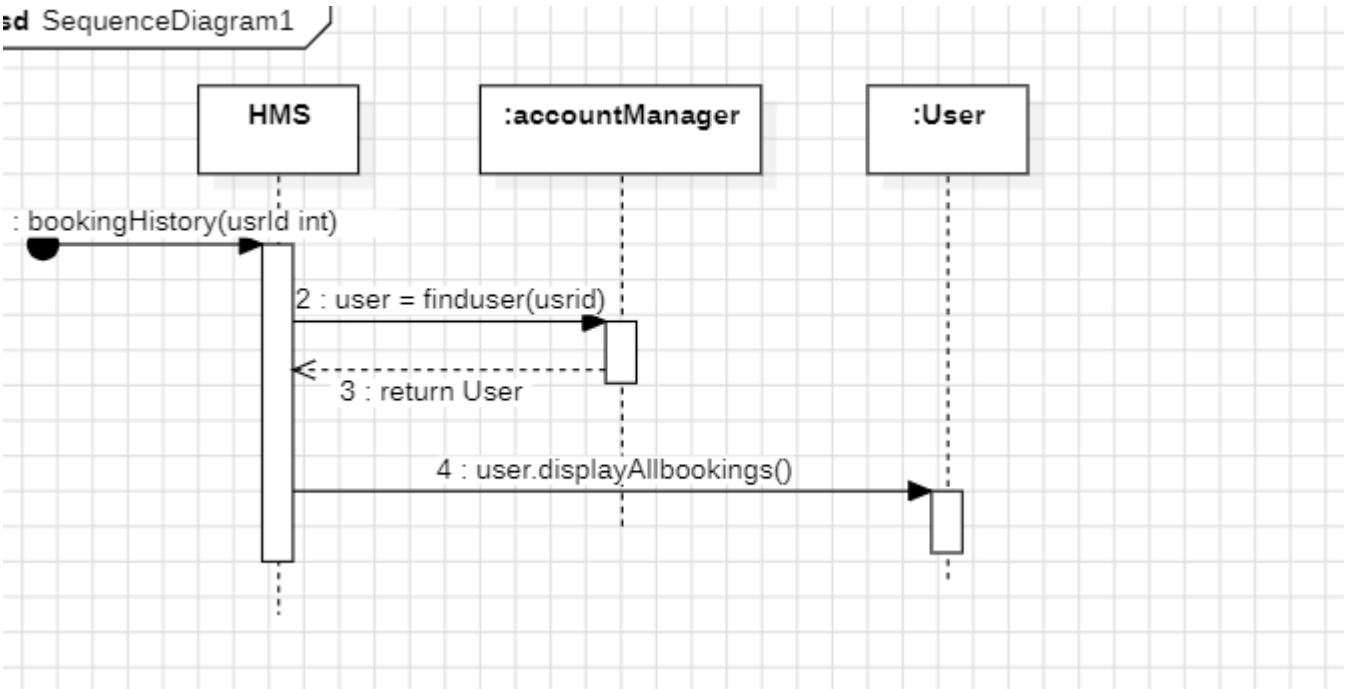


9. View Booking History

SSD

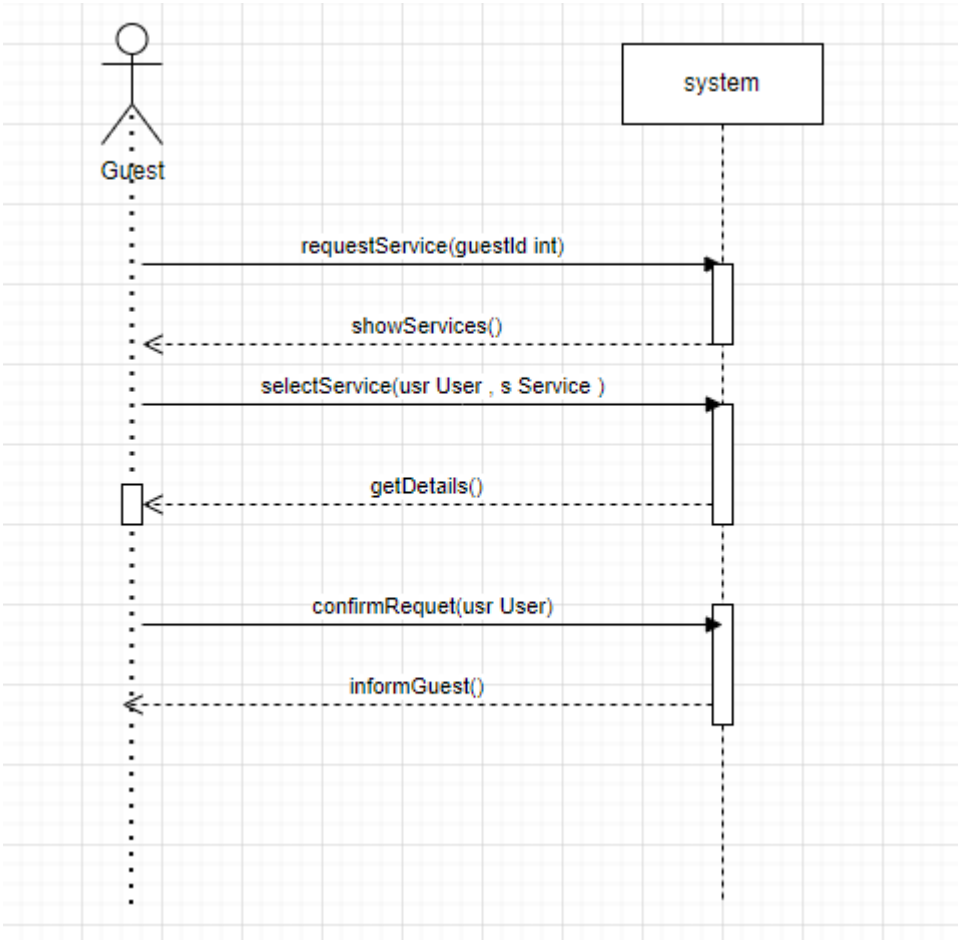

system
Guest
bookingHistory(usrId int)
displayallBookings()
filterResult(time int)
dispalyBookings()
ReviewandPrint()
printDetails()

1

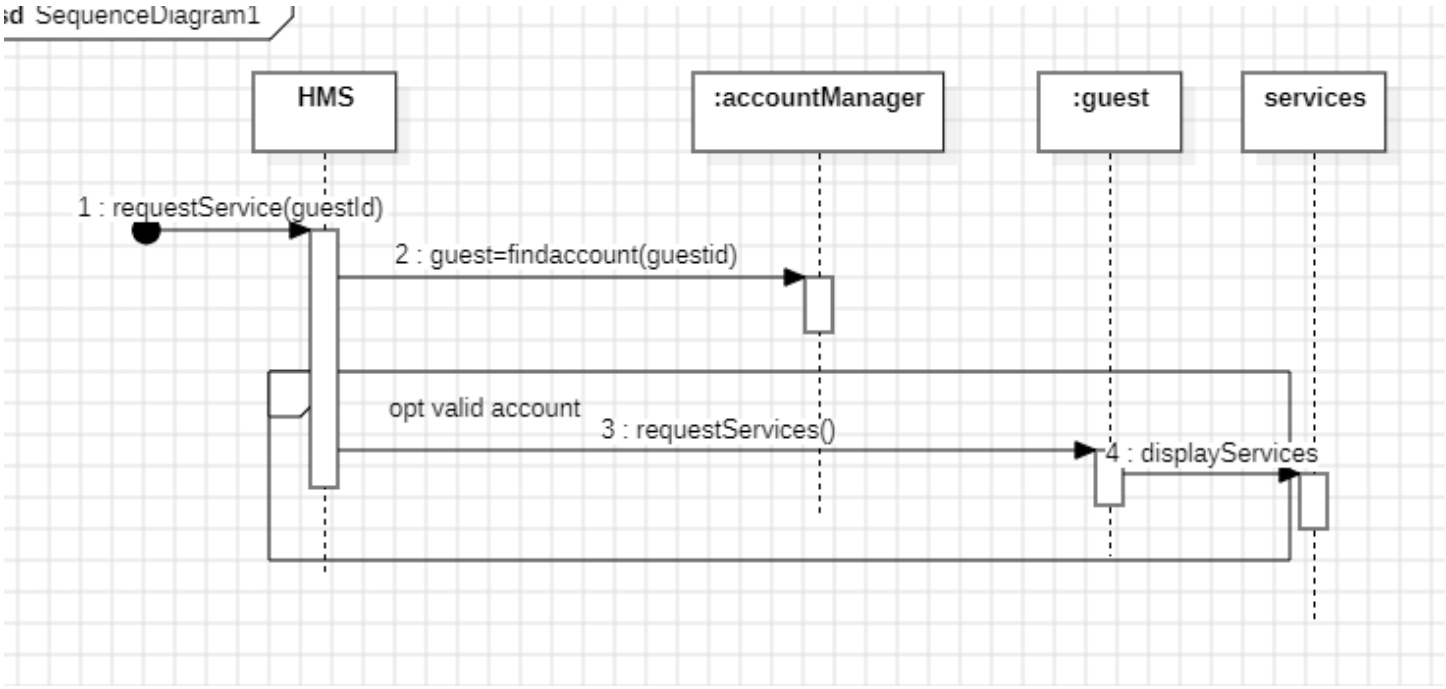

sd SequenceDiagram1
HMS
:accountManager
:User
: bookingHistory(usrId int)
2 : user = finduser(usrid)
3 : return User
4 : user.displayAllbookings()

10. Request Room Services

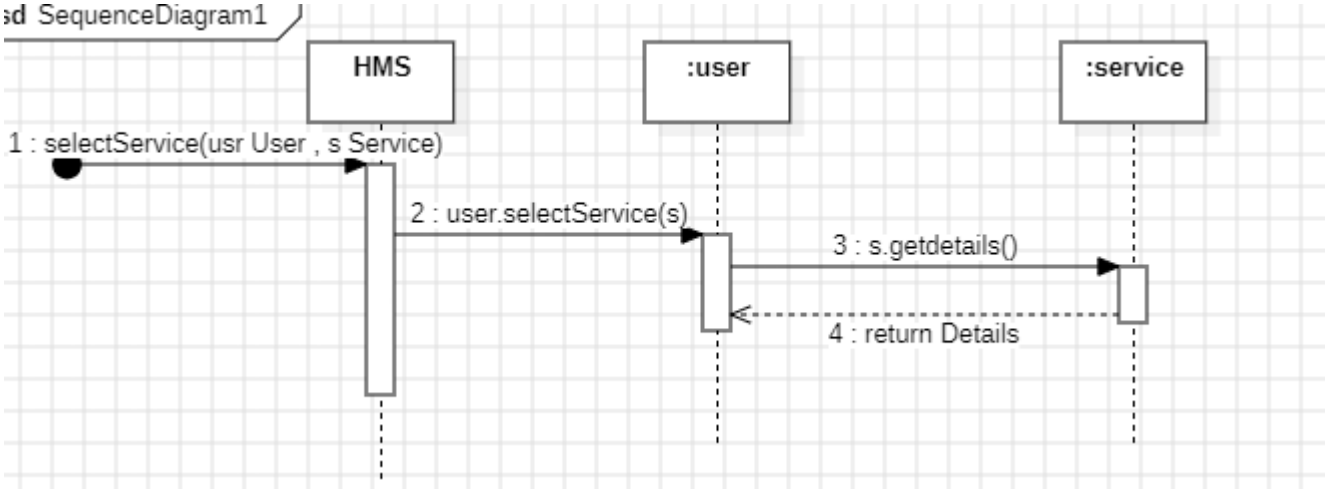
SSD



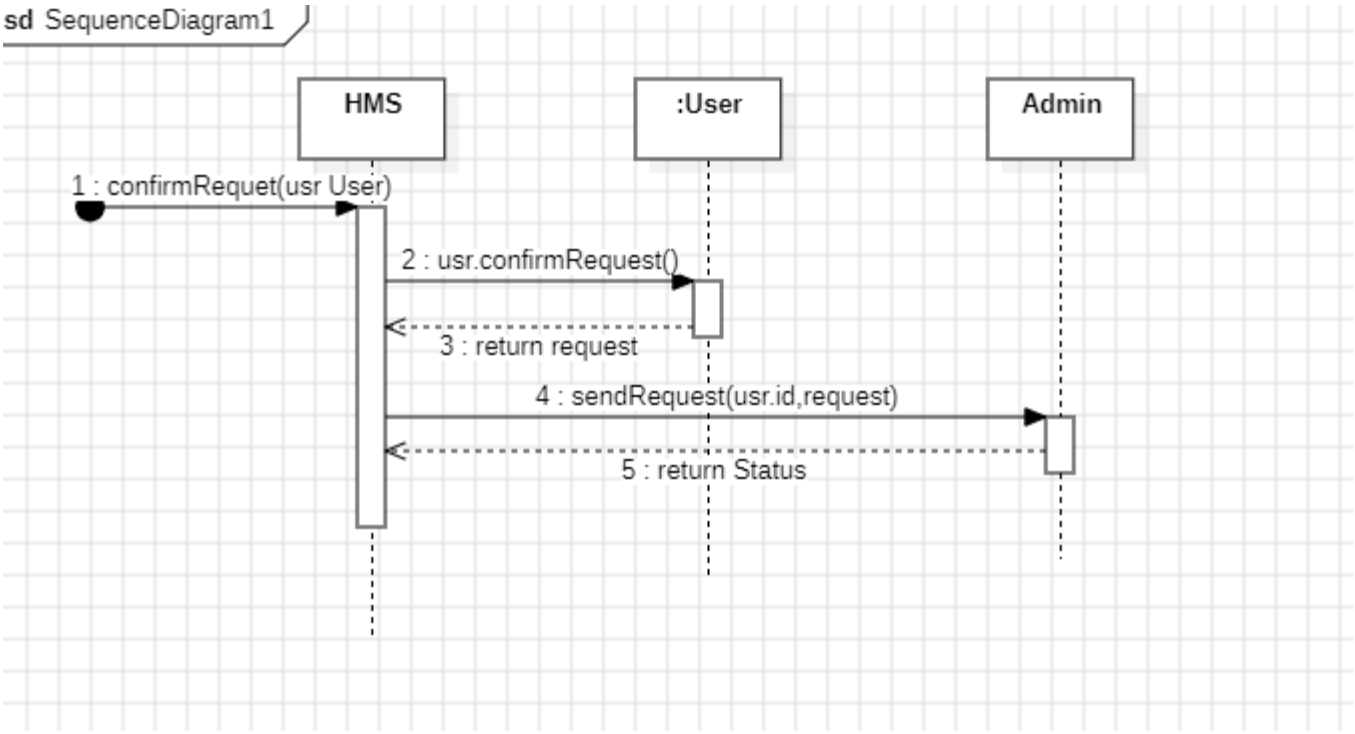
SD 1


sd SequenceDiagram1
HMS
:accountManager
:guest
services
1 : requestService(guestId)
2 : guest=findaccount(guestid)
opt valid account
3 : requestServices()
4 : displayServices

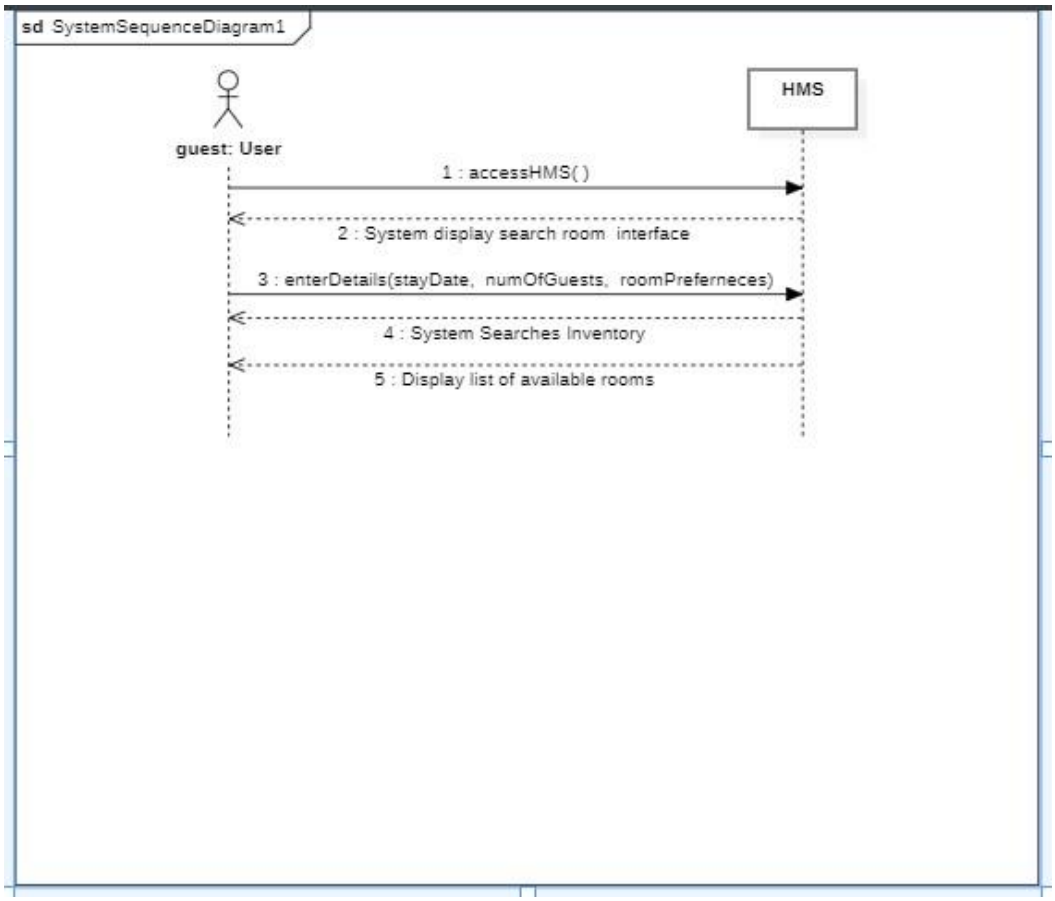
2


sd SequenceDiagram1
HMS
:user
:service
1 : selectService(usr User , s Service)
2 : user.selectService(s)
3 : s.getdetails()
4 : return Details

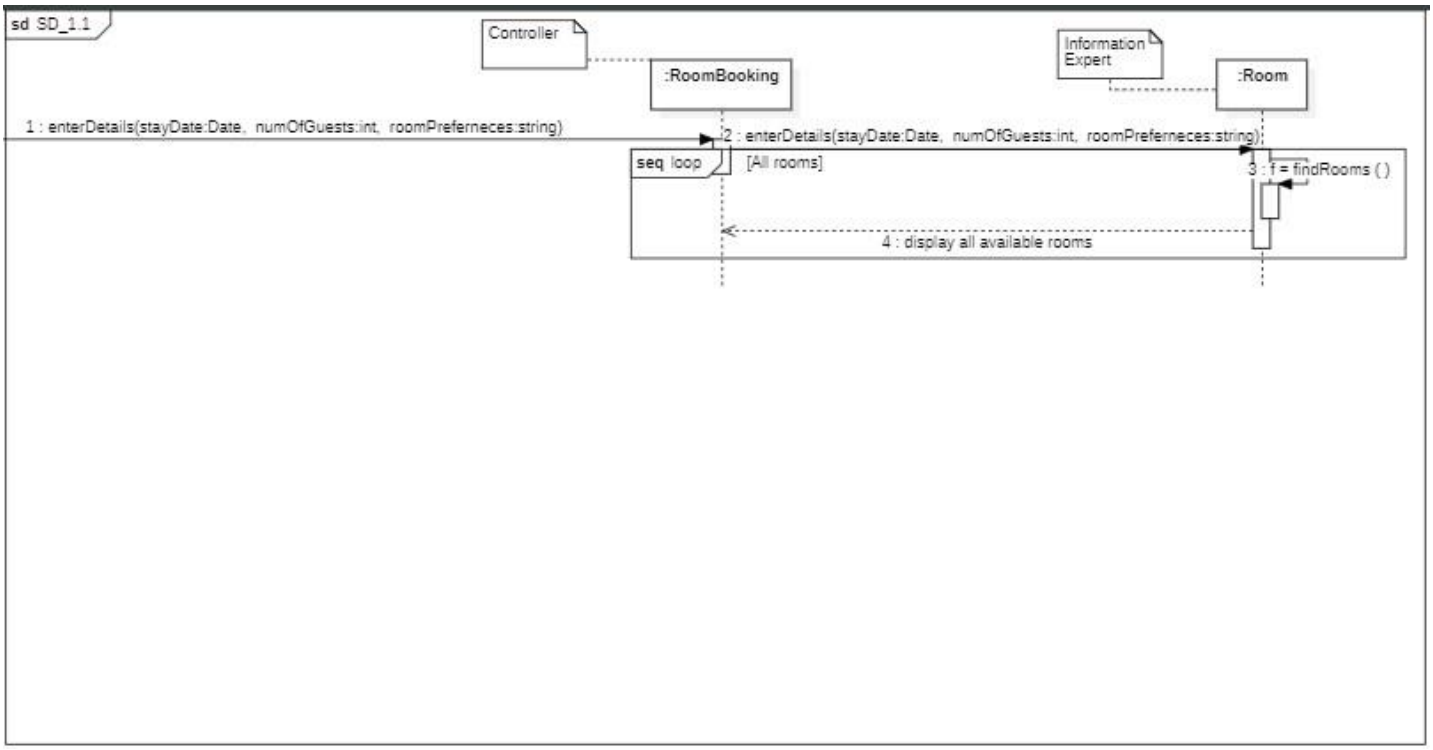
3



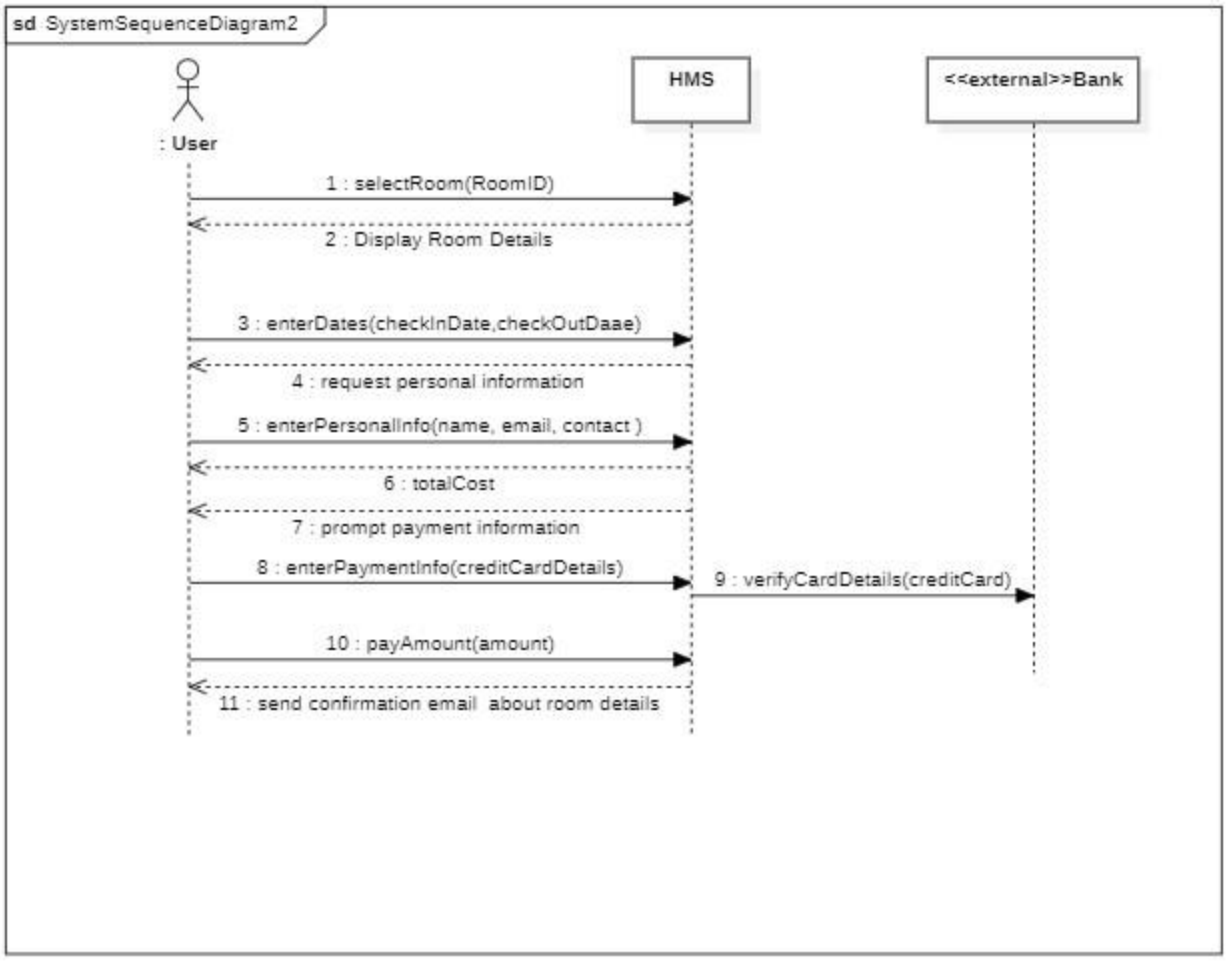
11. Search for Room Availability


sd SystemSequenceDiagram1
guest: User
HMS
1 : accessHMS( )
2 : System display search room interface
3 : enterDetails(stayDate, numOfGuests, roomPreferneces)
4 : System Searches Inventory
5 : Display list of available rooms

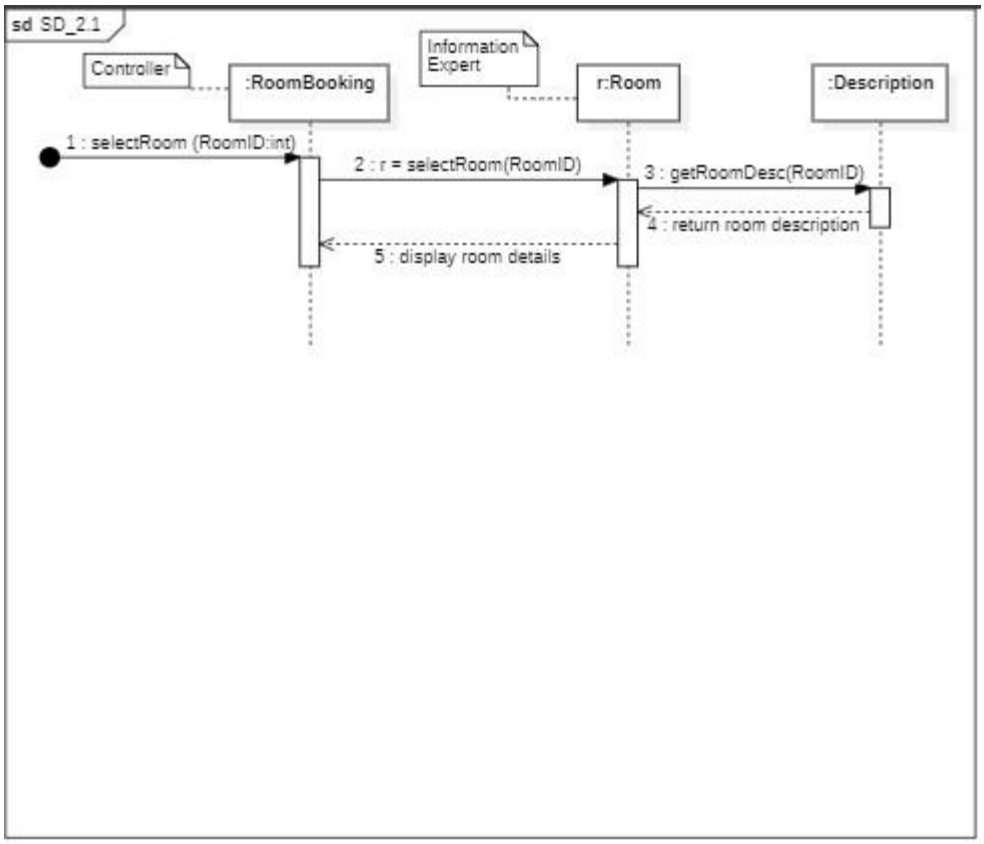
SD 1



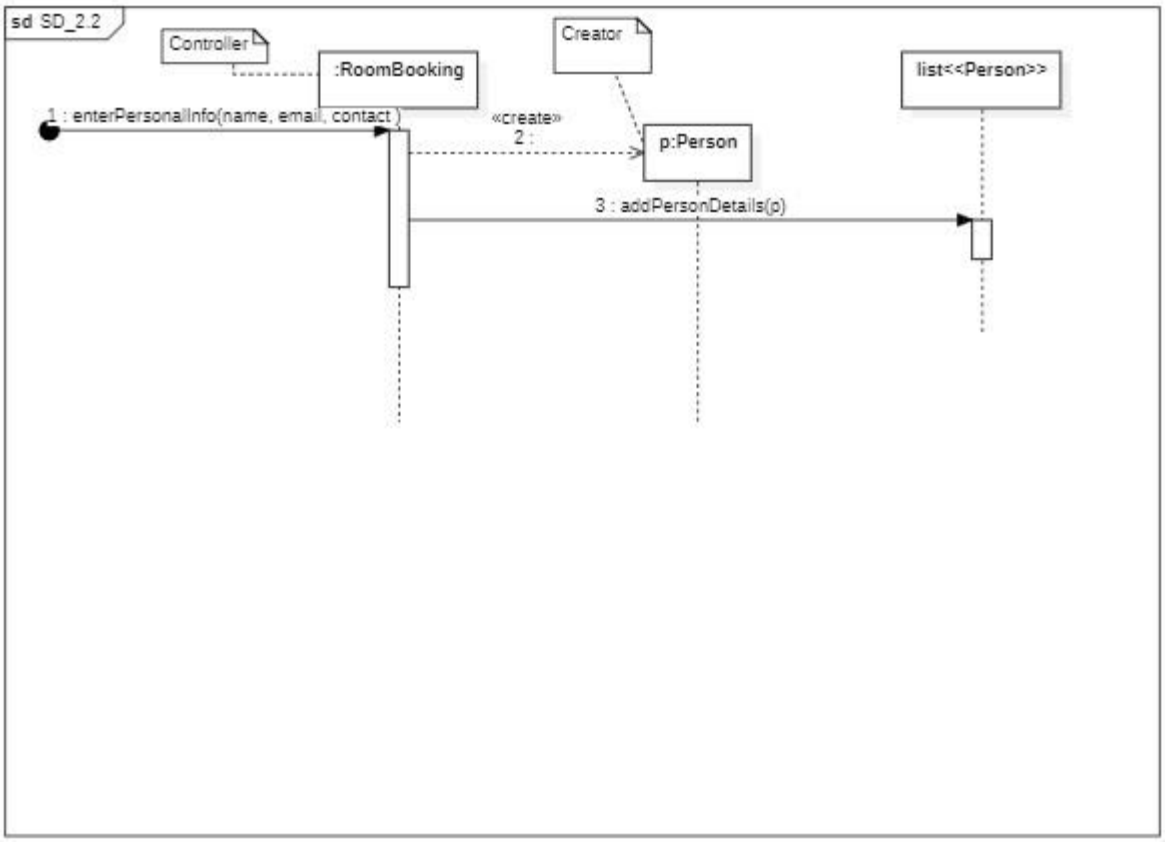
12. Book a Room


sd SystemSequenceDiagram2
: User
HMS
<<external>>Bank
1 : selectRoom(RoomID)
2 : Display Room Details
3 : enterDates(checkInDate,checkOutDaae)
4 : request personal information
5 : enterPersonalInfo(name, email, contact)
6 : totalCost
7 : prompt payment information
8 : enterPaymentInfo(creditCardDetails)
9 : verifyCardDetails(creditCard)
10 : payAmount(amount)
11 : send confirmation email about room details

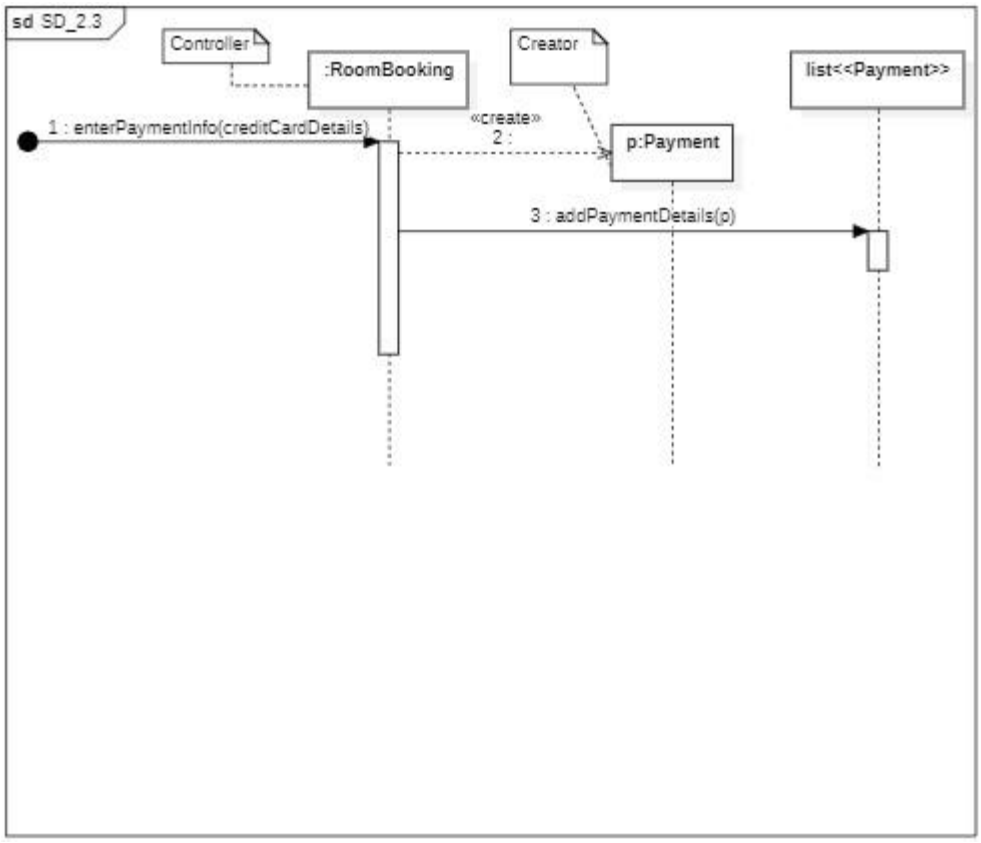
SD 1


sd SD_2.1
Controller
:RoomBooking
Information Expert
r:Room
:Description
1 : selectRoom (RoomID:int)
2 : r = selectRoom(RoomID)
3 : getRoomDesc(RoomID)
4 : return room description
5 : display room details

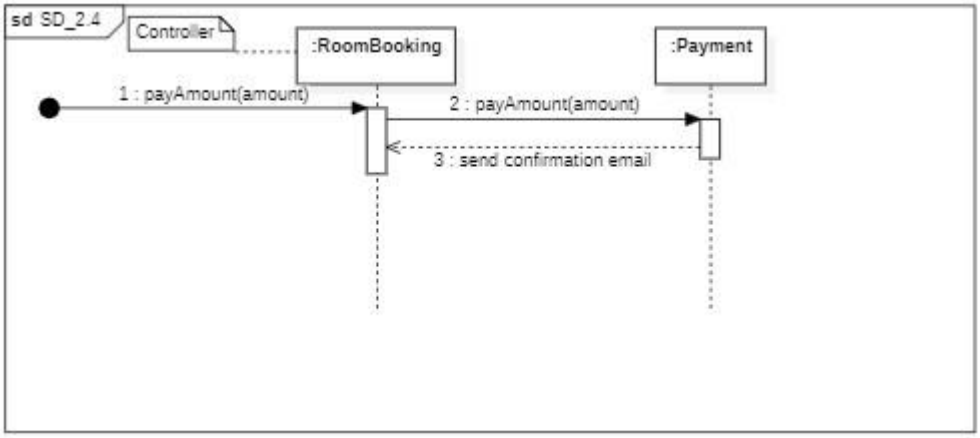
2


sd SD_2.2
Controller
:RoomBooking
Creator
list<<Person>>
1 : enterPersonalInfo(name, email, contact)
«create»
2 :
p:Person
3 : addPersonDetails(p)

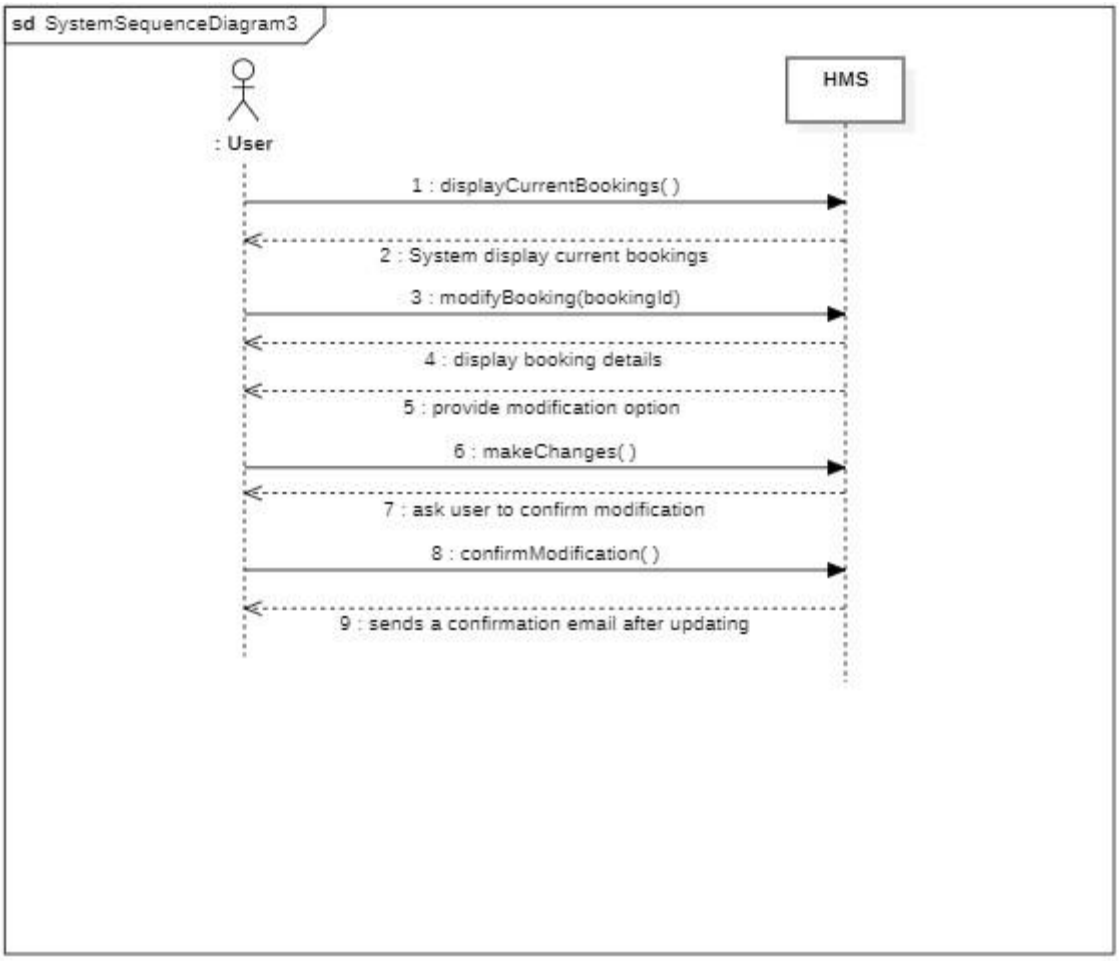
3



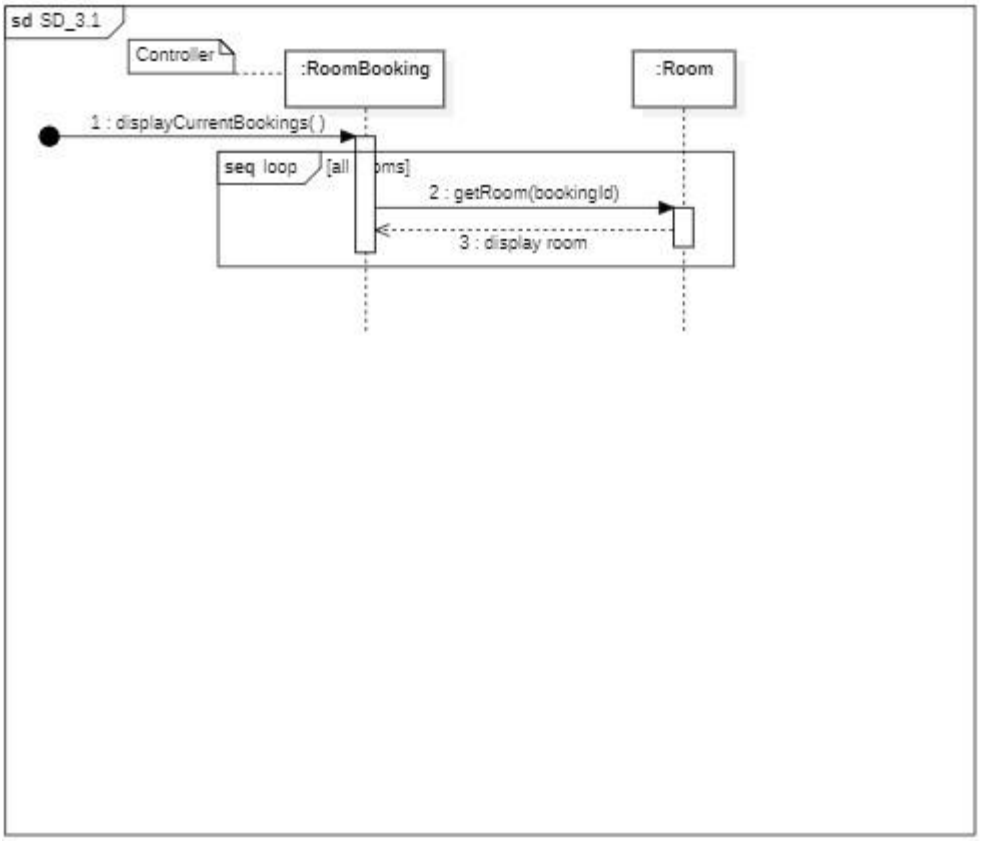
4



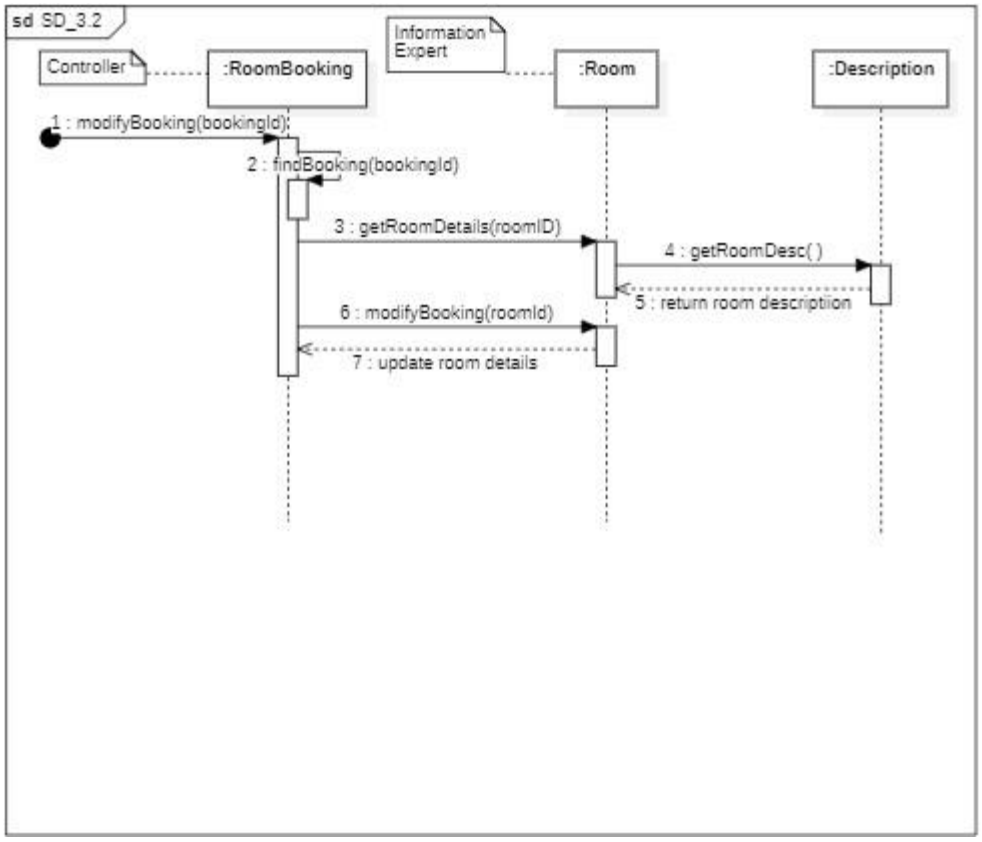
13. Modify a Booking


sd SystemSequenceDiagram3
: User
HMS
1 : displayCurrentBookings( )
2 : System display current bookings
3 : modifyBooking(bookingId)
4 : display booking details
5 : provide modification option
6 : makeChanges( )
7 : ask user to confirm modification
8 : confirmModification( )
9 : sends a confirmation email after updating

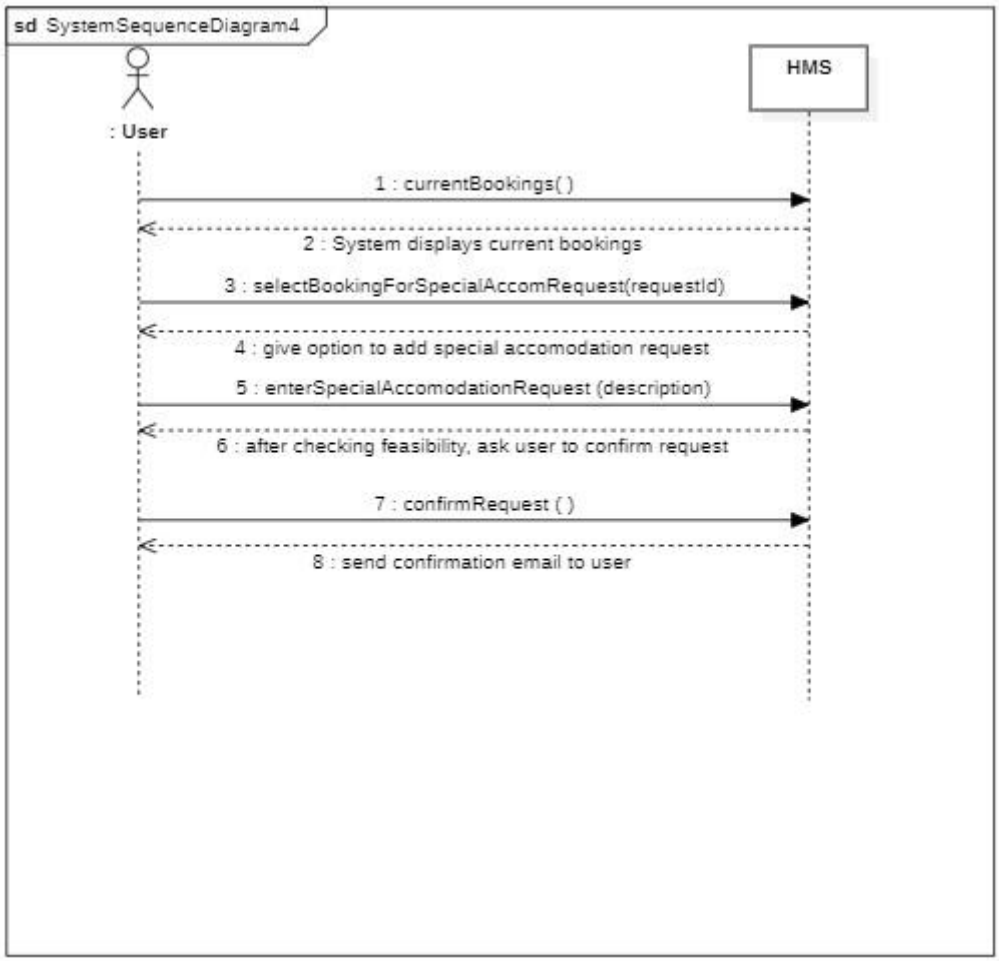
SD 1


sd SD_3.1
Controller
:RoomBooking
:Room
1 : displayCurrentBookings( )
seq loop
2 : getRoom(bookingId)
3 : display room

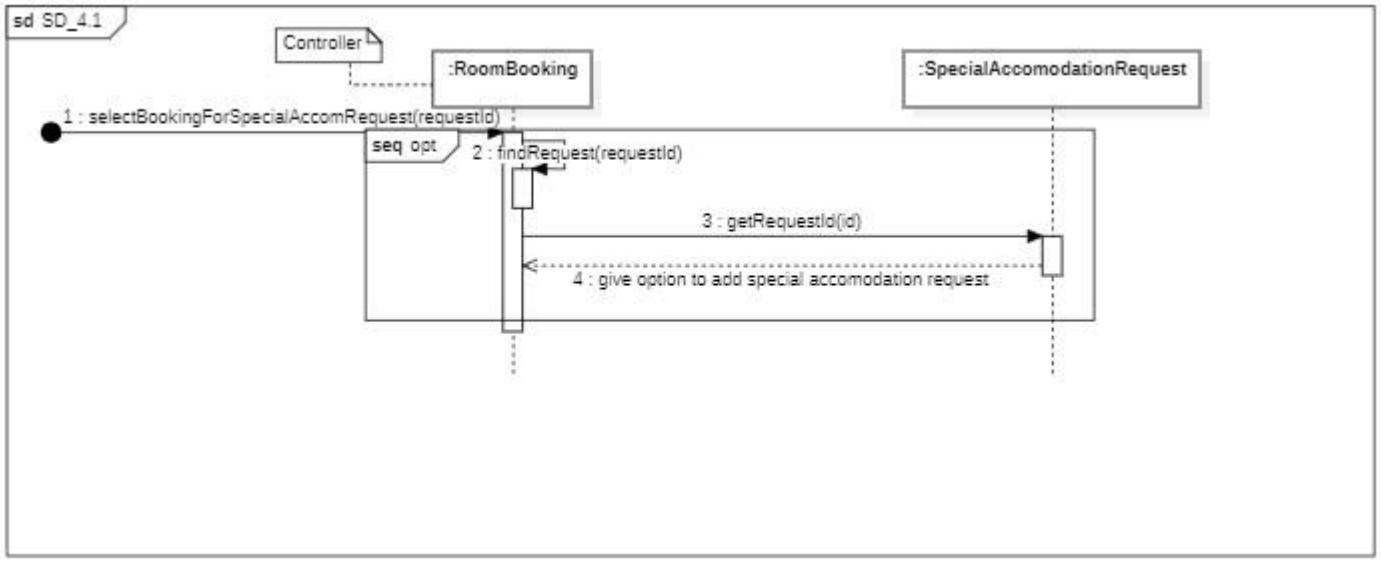
2



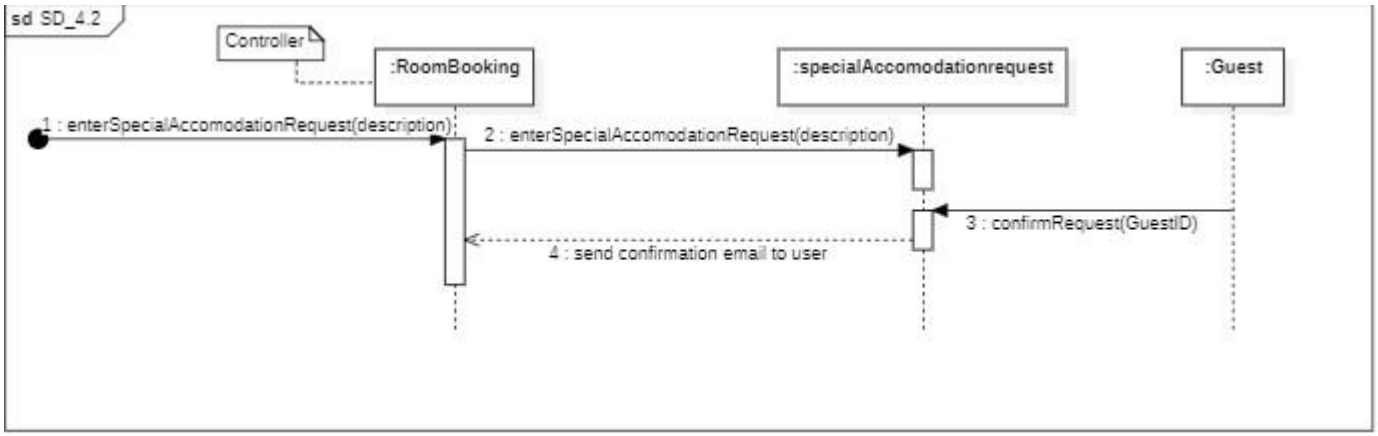
14. Request for Special Accommodations


sd SystemSequenceDiagram4
: User
HMS
1 : currentBookings( )
2 : System displays current bookings
3 : selectBookingForSpecialAccomRequest(requestId)
4 : give option to add special accomodation request
5 : enterSpecialAccomodationRequest (description)
6 : after checking feasibility, ask user to confirm request
7 : confirmRequest ( )
8 : send confirmation email to user

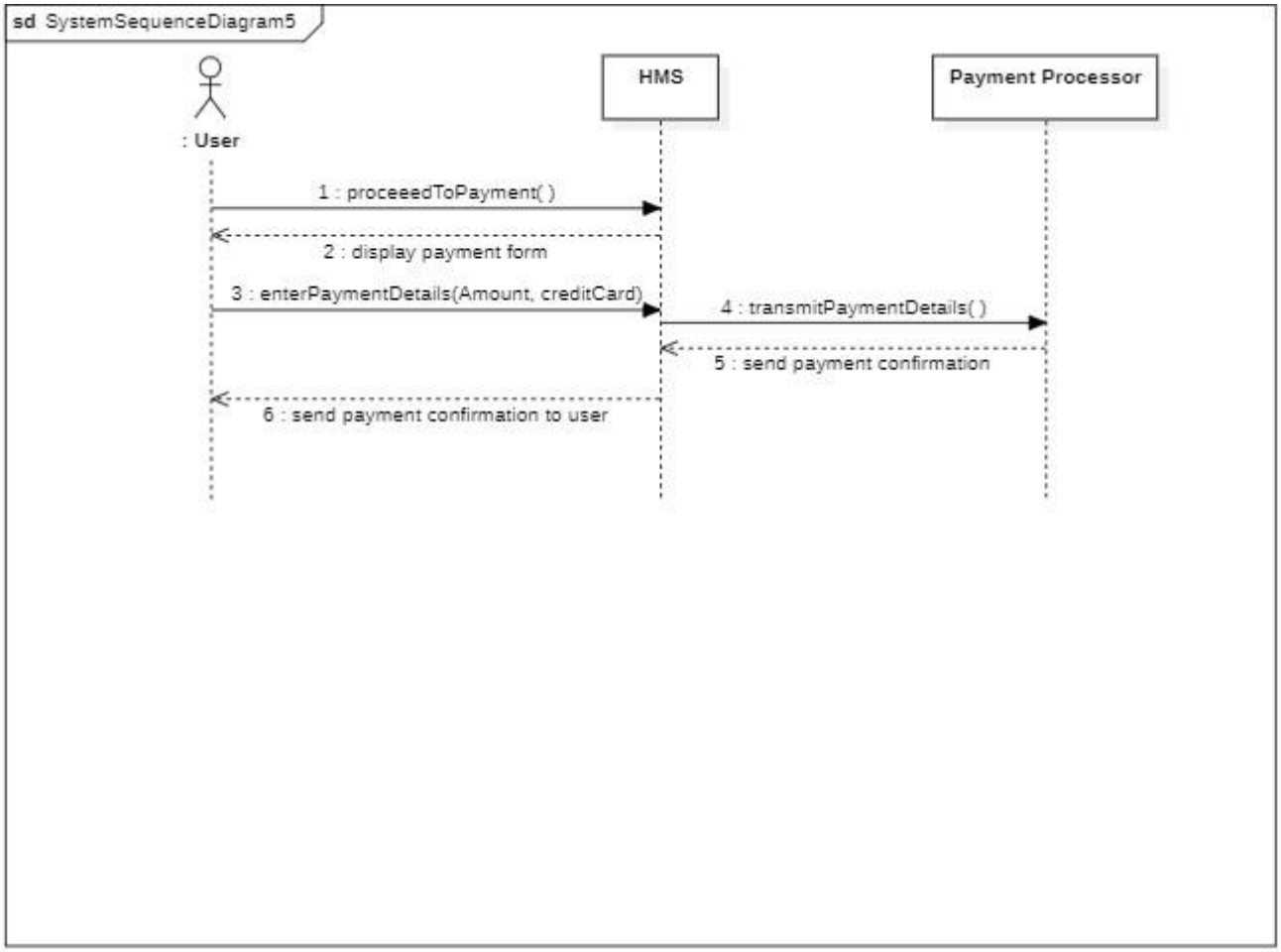
SD 1


sd SD_4.1
Controller
:RoomBooking
:SpecialAccomodationRequest
1 : selectBookingForSpecialAccomRequest(requestId)
seq opt
2 : findRequest(requestId)
3 : getRequestId(id)
4 : give option to add special accomodation request

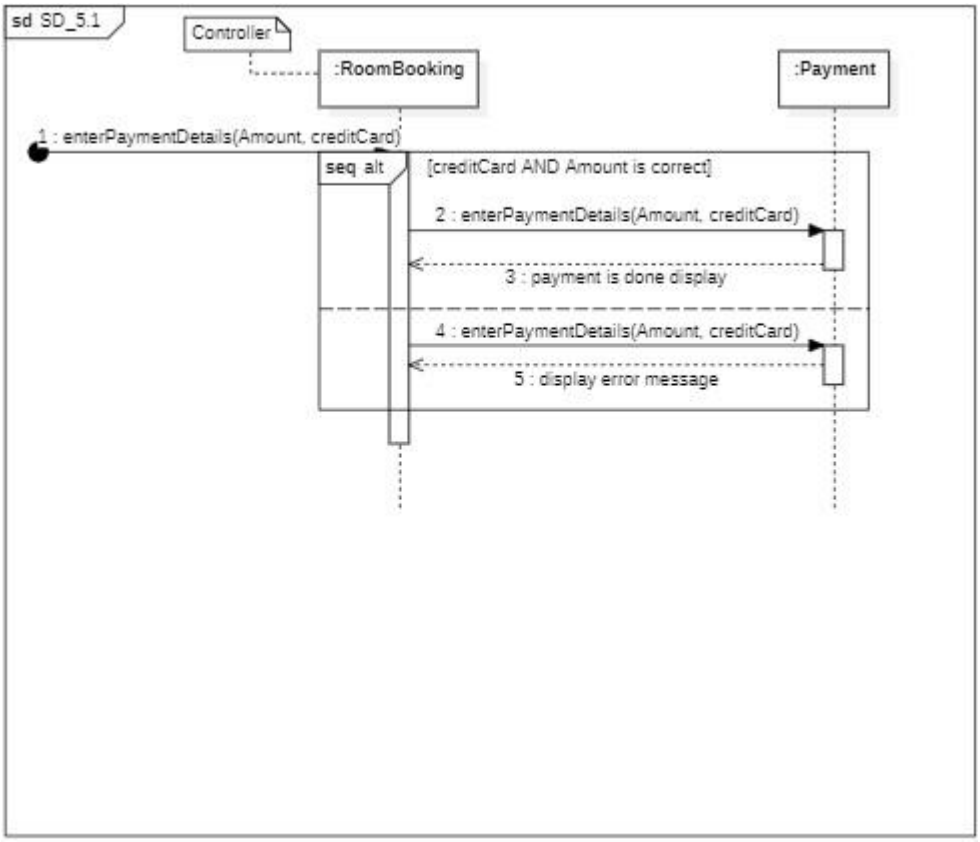
2



15. Process Payment

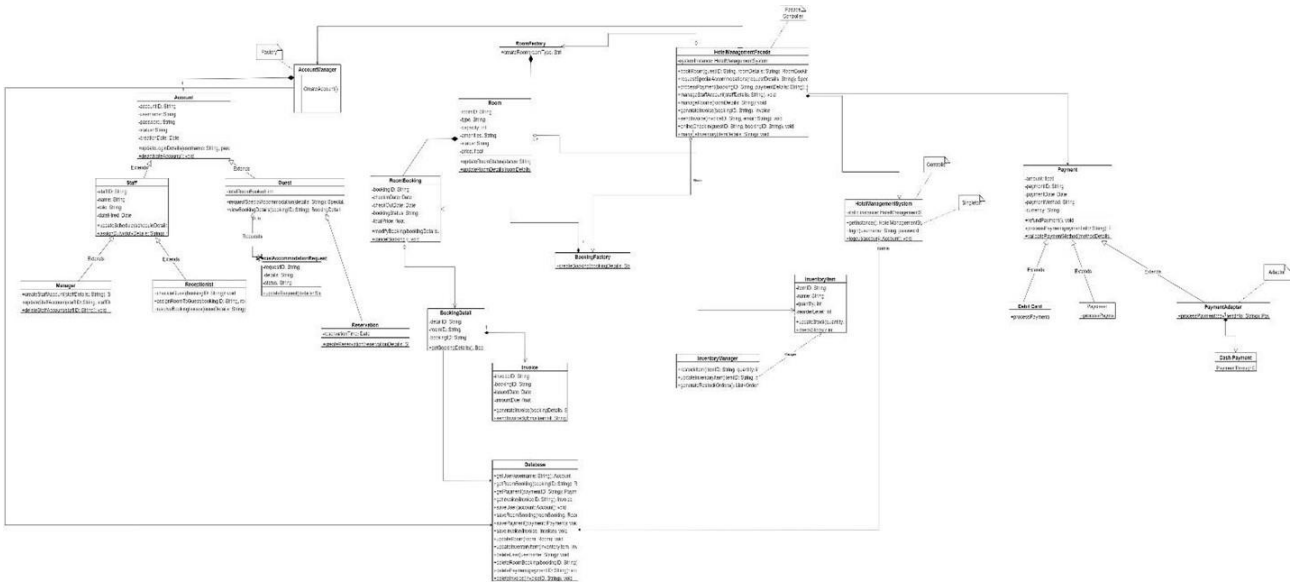


SD 1

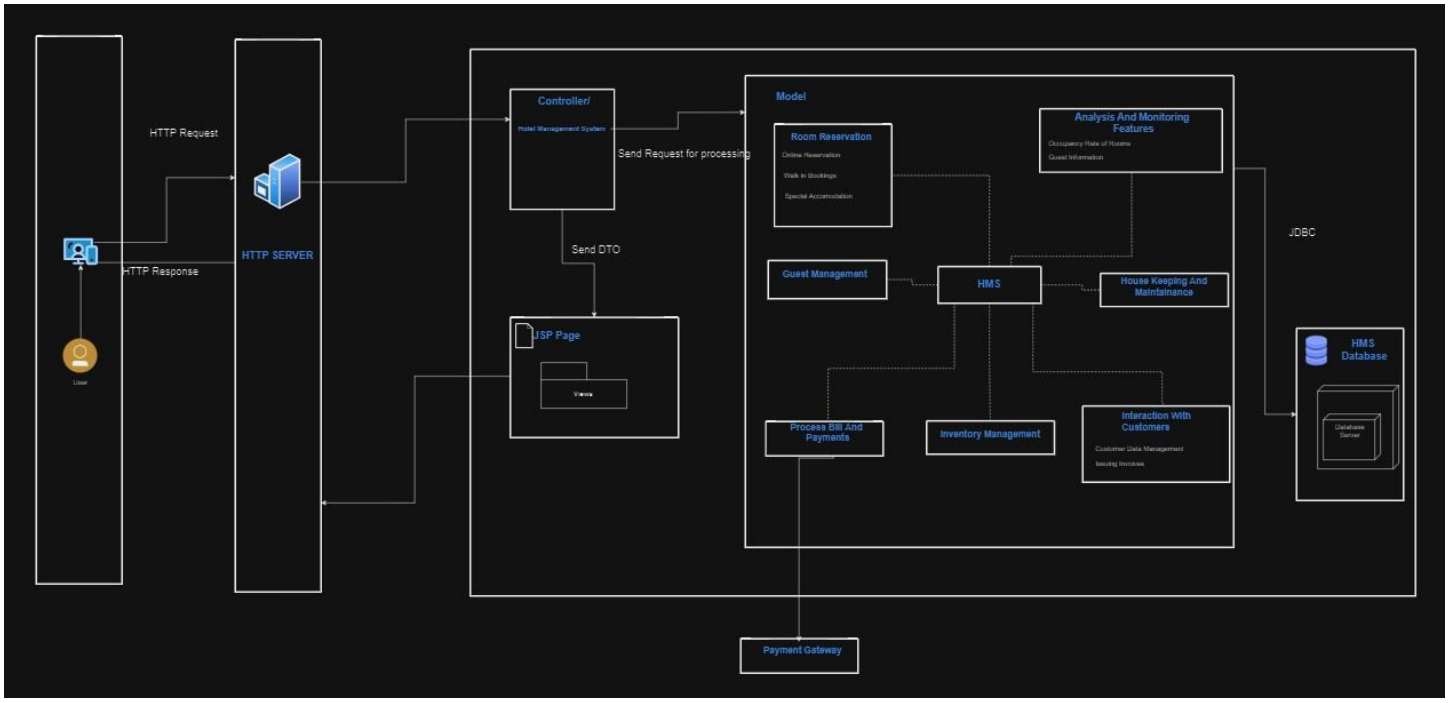


Implemented

6. Class Diagram

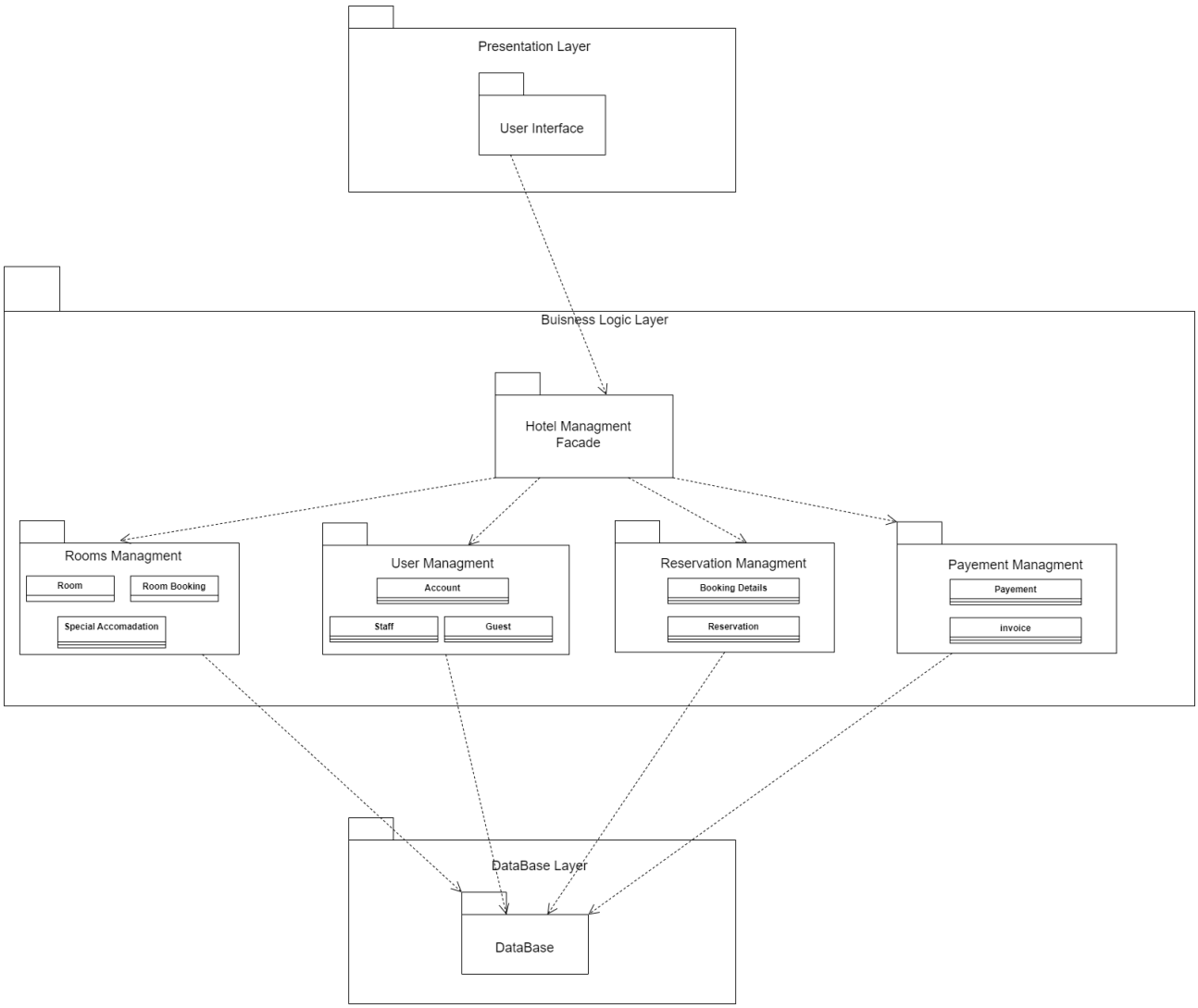


7. High-level Architecture

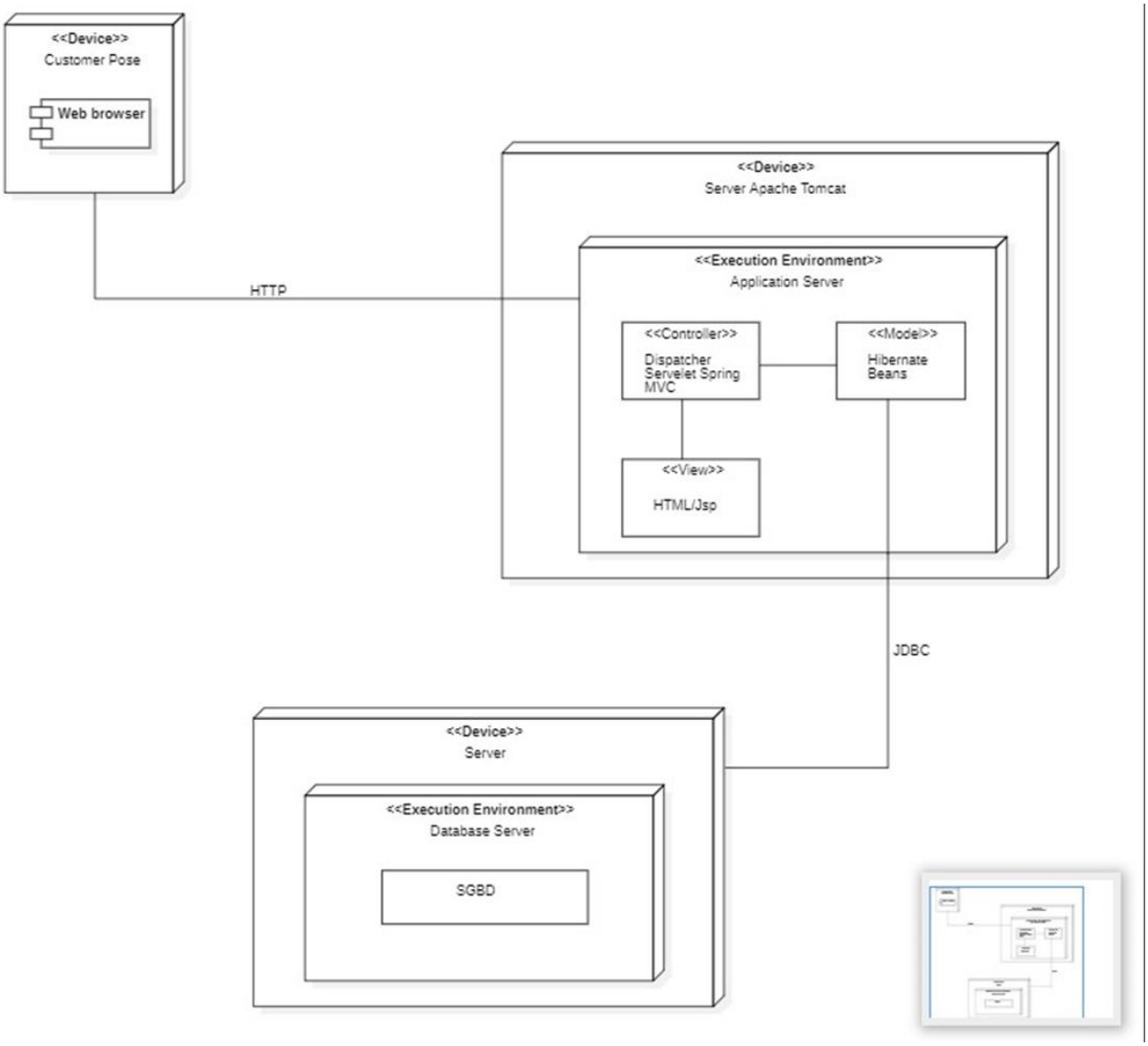


●

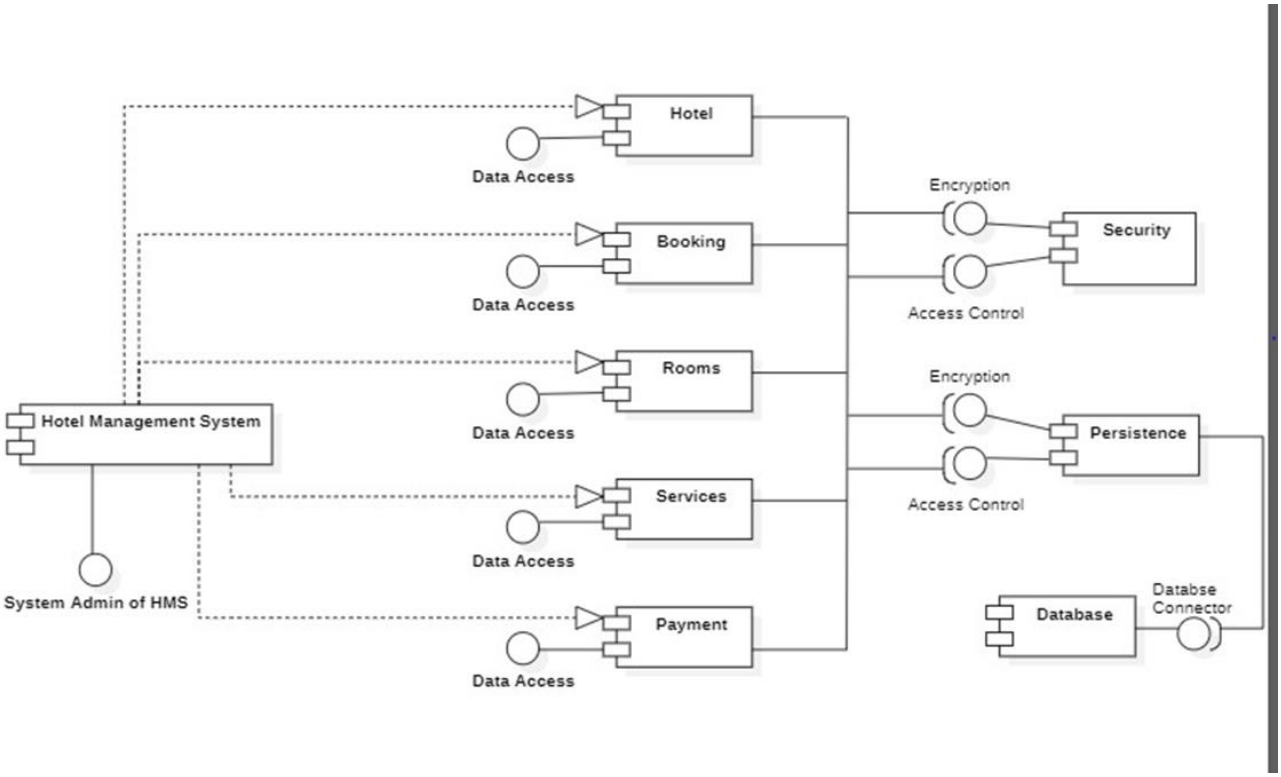
7.1 Package Diagram


Presentation Layer
User Interface
Buisness Logic Layer
Hotel Managment Facade
Rooms Managment
Room
Room Booking
Special Accomadation
User Managment
Account
Staff
Guest
Reservation Managment
Booking Details
Reservation
Payement Managment
Payement
invoice
DataBase Layer
DataBase

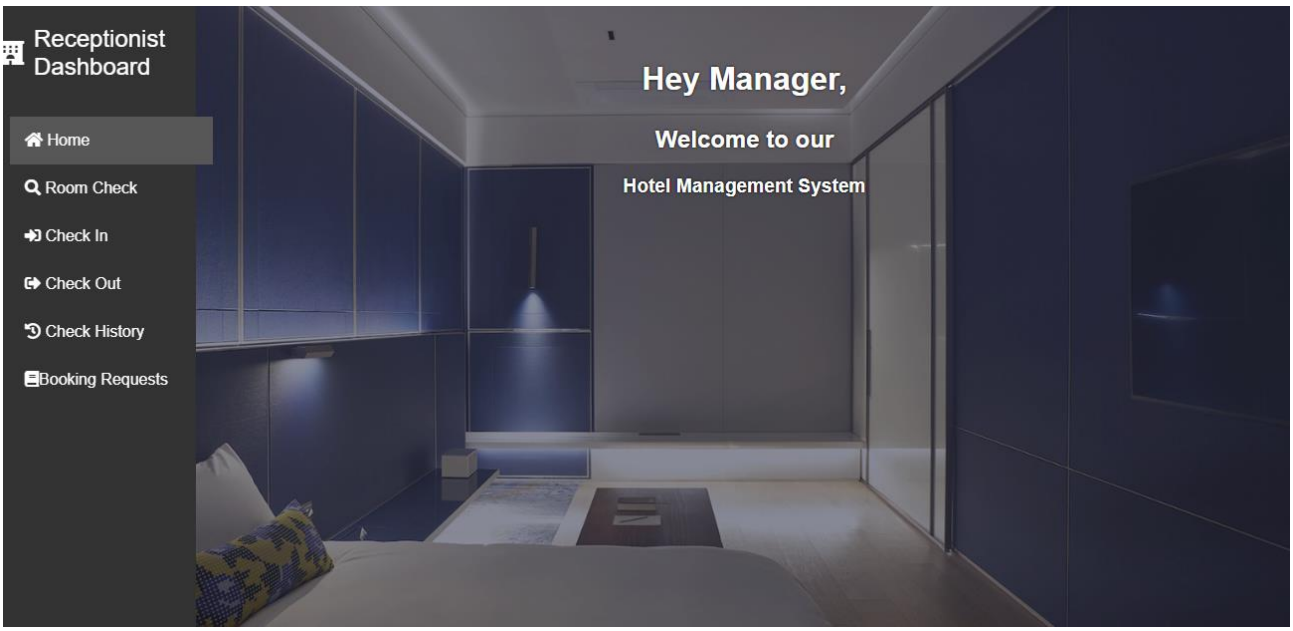
7.2 Deployment Diagram

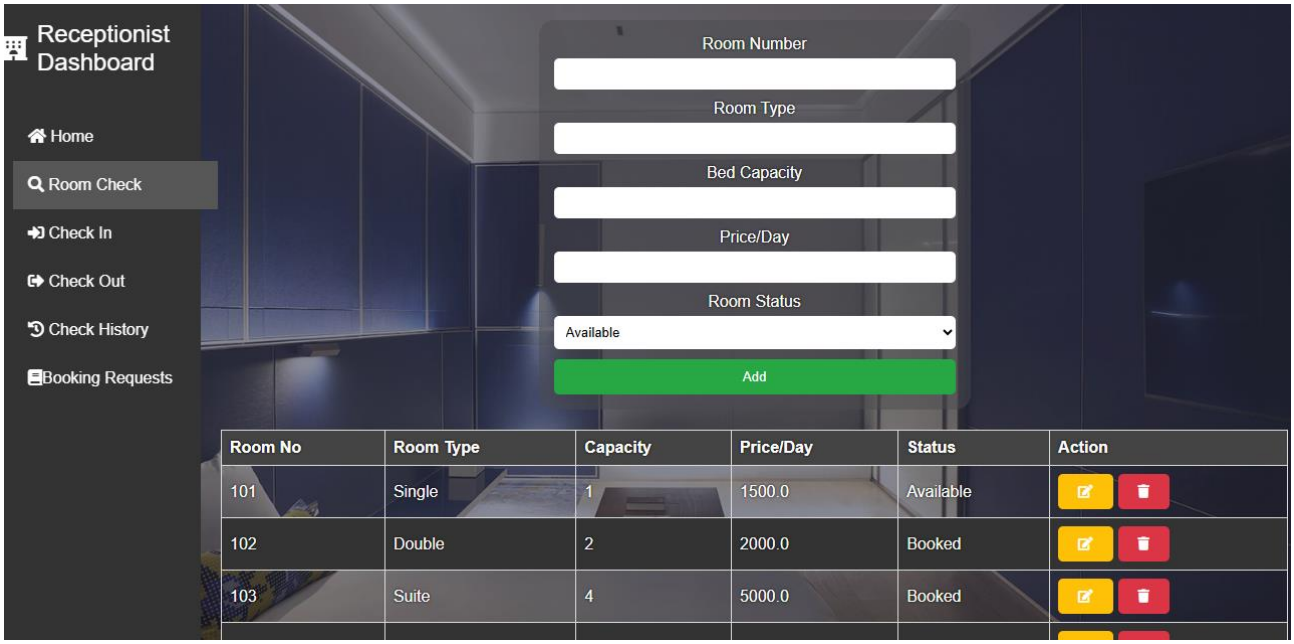


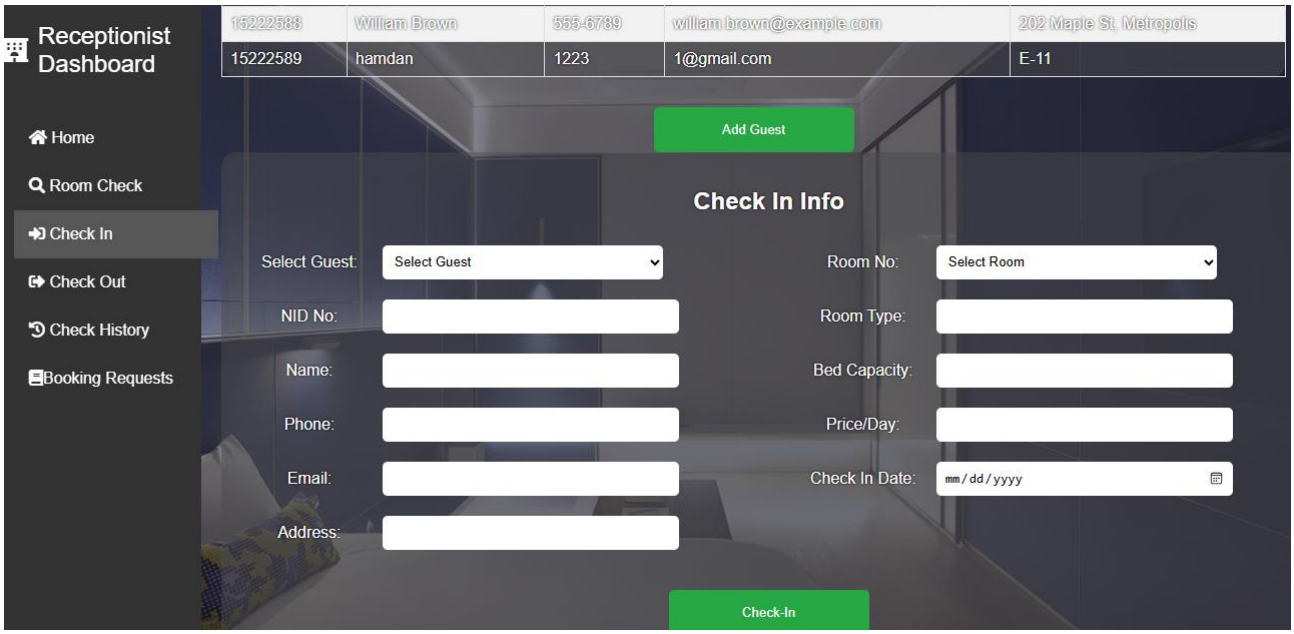
7.3 Component Diagram

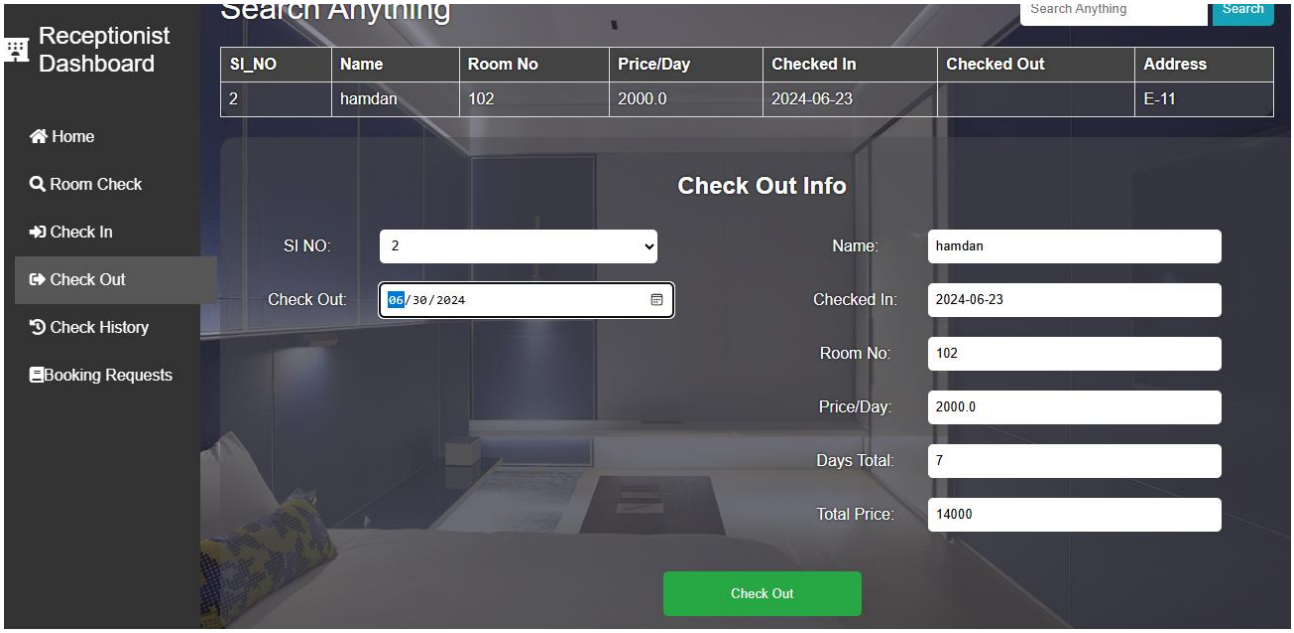


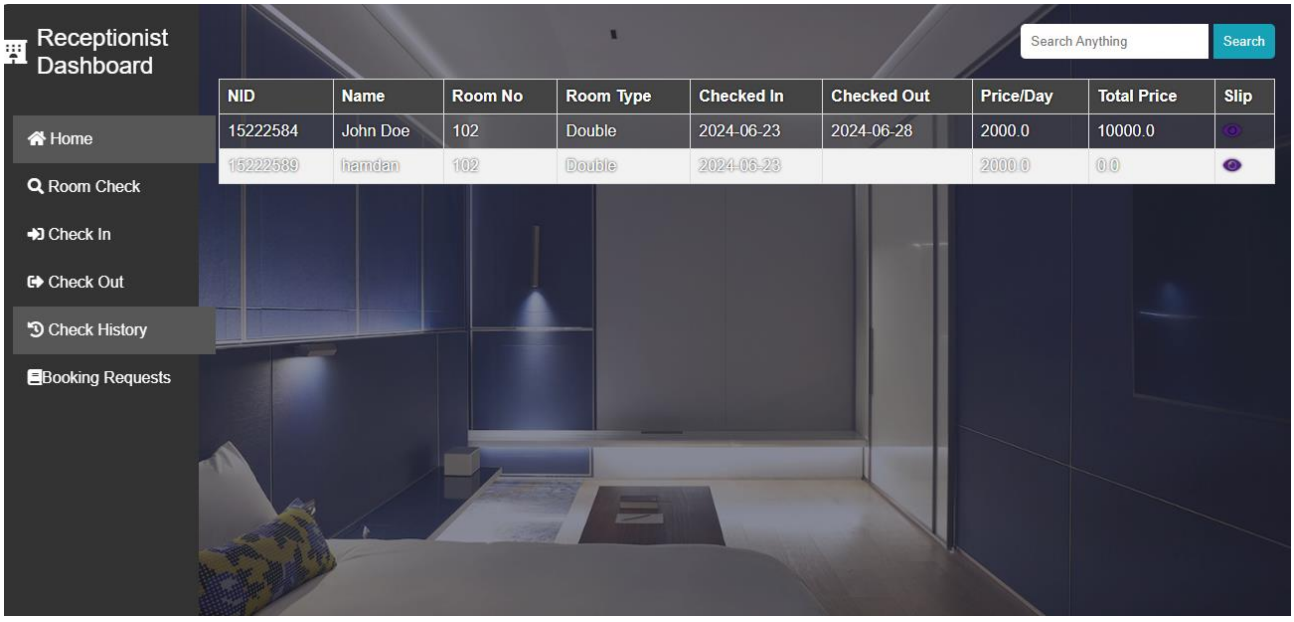
8. Screen Shots

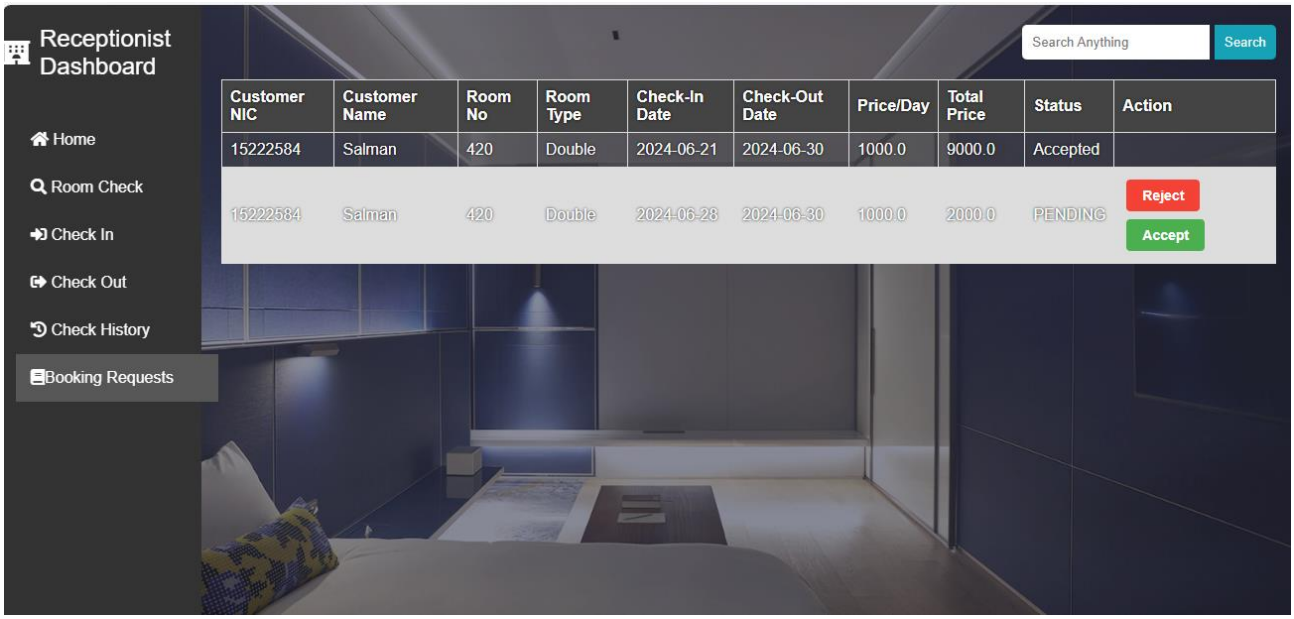


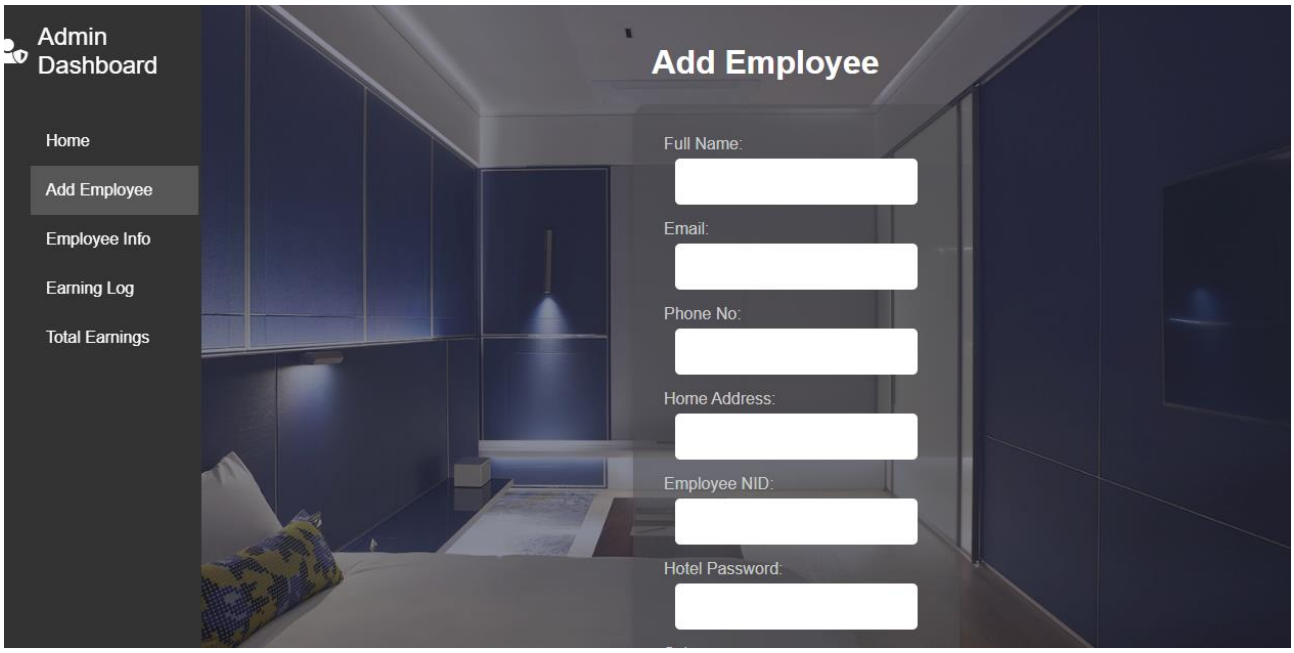

Receptionist Dashboard
Home
Room Check
Check In
Check Out
Check History
Booking Requests
Room Number
Room Type
Bed Capacity
Price/Day
Room Status
Available
Add
Room No
Room Type
Capacity
Price/Day
Status
Action
101
Single
1
1500.0
Available
102
Double
2
2000.0
Booked
103
Suite
4
5000.0
Booked

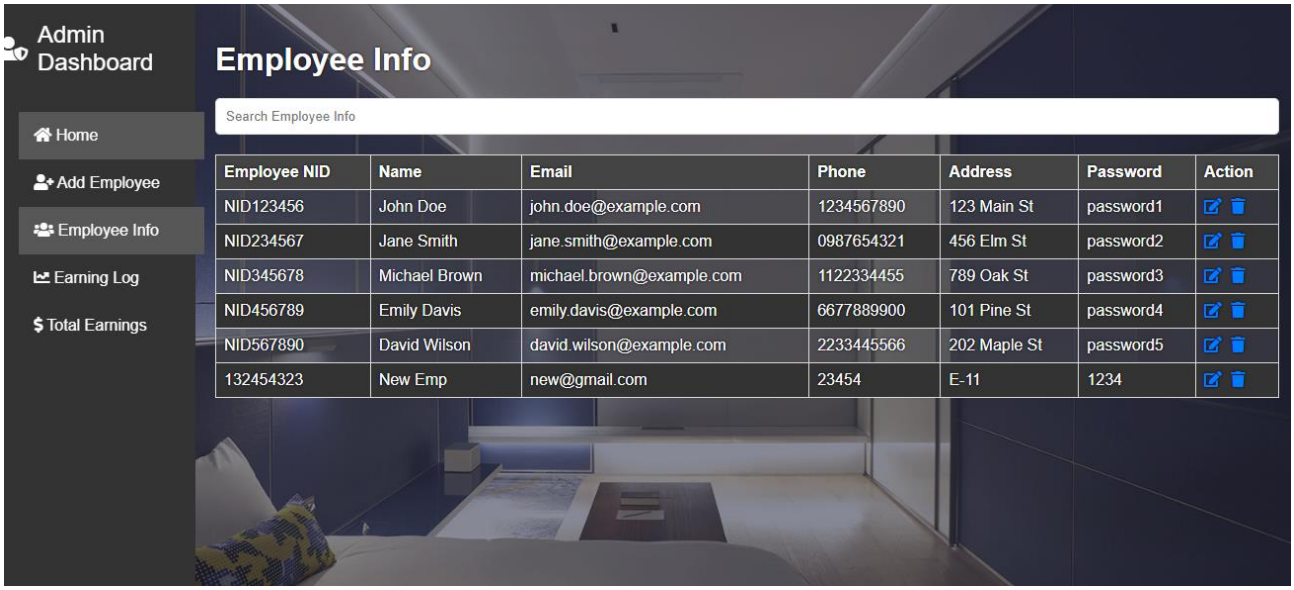

Receptionist Dashboard
Home
Room Check
Check In
Check Out
Check History
Booking Requests
15222588
William Brown
555-6789
william.brown@example.com
202 Maple St, Metropolis
15222589
hamdan
1223
1@gmail.com
E-11
Add Guest
Check In Info
Select Guest:
Select Guest
NID No:
Name:
Phone:
Email:
Address:
Room No:
Select Room
Room Type:
Bed Capacity:
Price/Day:
Check In Date:
mm/dd/yyyy
Check-In

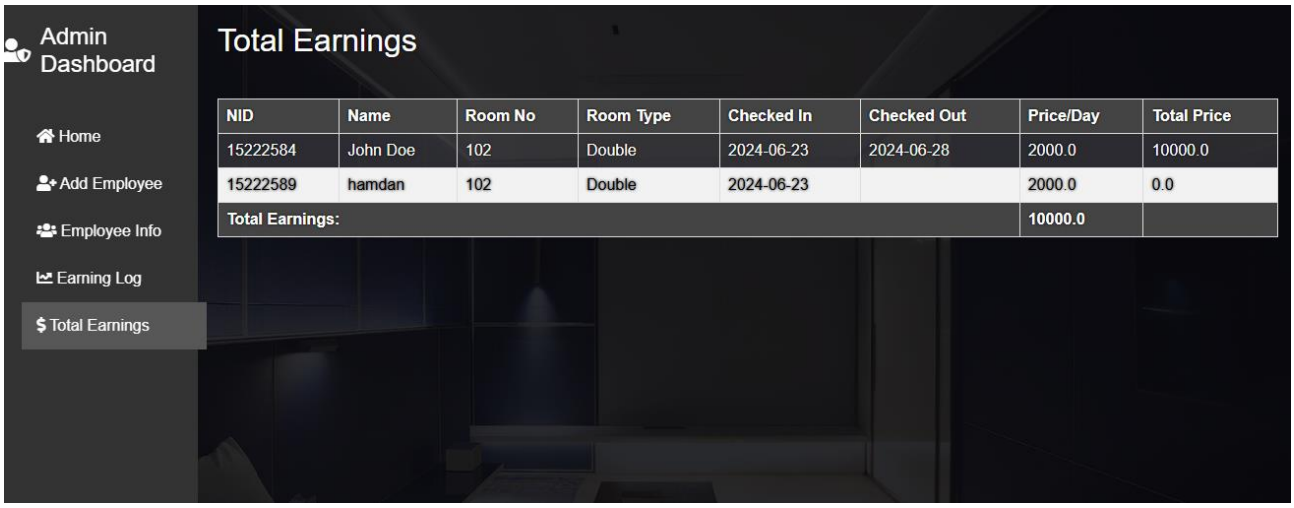

Receptionist Dashboard
Home
Room Check
Check In
Check Out
Check History
Booking Requests
Search Anything
Search
SI_NO Name Room No Price/Day Checked In Checked Out Address
2 hamdan 102 2000.0 2024-06-23 E-11
Check Out Info
SI NO: 2
Check Out: 06/30/2024
Name: hamdan
Checked In: 2024-06-23
Room No: 102
Price/Day: 2000.0
Days Total: 7
Total Price: 14000
Check Out

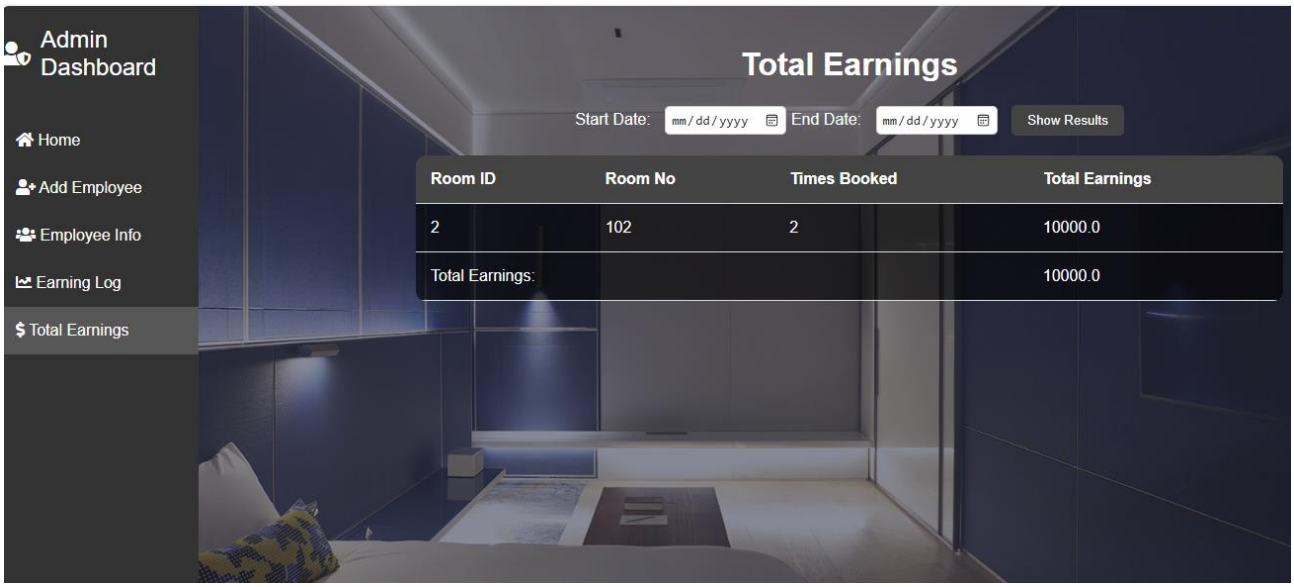

Receptionist Dashboard
Home
Room Check
Check In
Check Out
Check History
Booking Requests
Search Anything
Search
NID Name Room No Room Type Checked In Checked Out Price/Day Total Price Slip
15222584 John Doe 102 Double 2024-06-23 2024-06-28 2000.0 10000.0
15222589 hamdan 102 Double 2024-06-23 2000.0 0.0

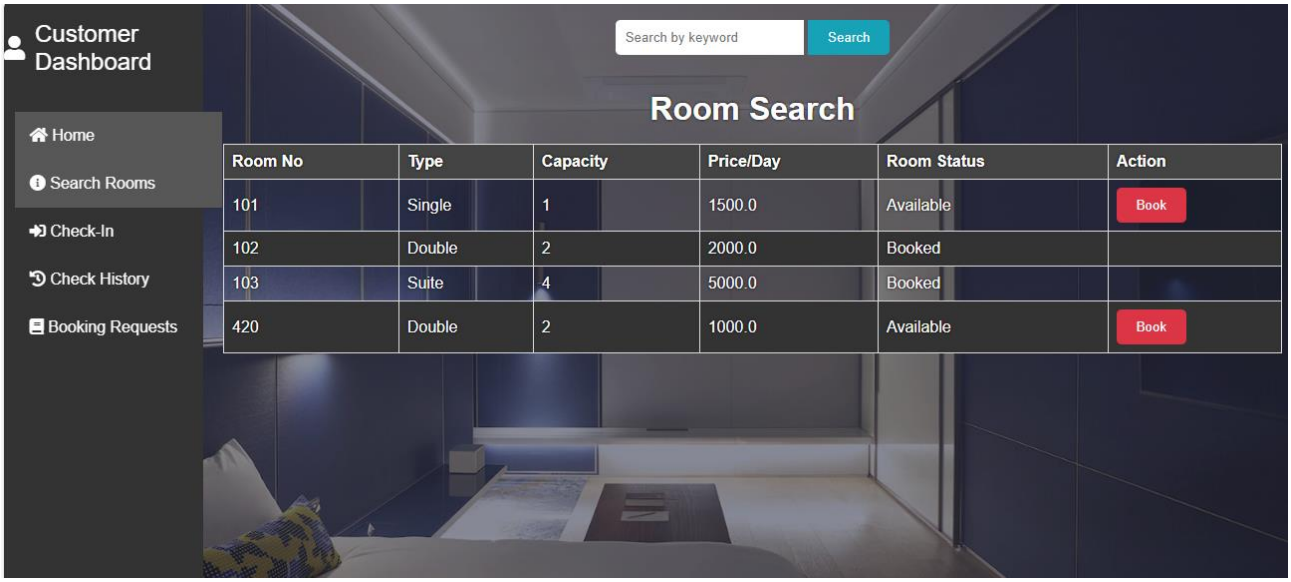

Receptionist Dashboard
Home
Room Check
Check In
Check Out
Check History
Booking Requests
Search Anything
Search
Customer NIC
Customer Name
Room No
Room Type
Check-In Date
Check-Out Date
Price/Day
Total Price
Status
Action
15222584
Salman
420
Double
2024-06-21
2024-06-30
1000.0
9000.0
Accepted
15222584
Salman
420
Double
2024-06-28
2024-06-30
1000.0
2000.0
PENDING
Reject
Accept

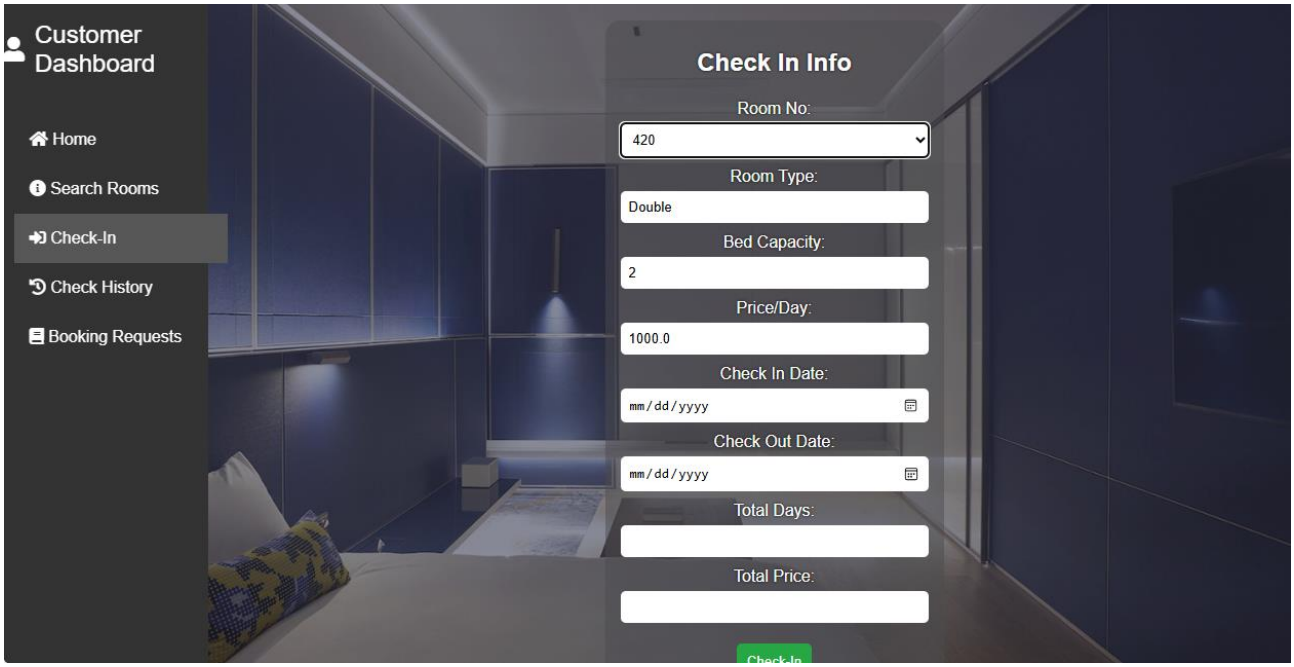

Admin Dashboard
Home
Add Employee
Employee Info
Earning Log
Total Earnings
Add Employee
Full Name:
Email:
Phone No:
Home Address:
Employee NID:
Hotel Password:

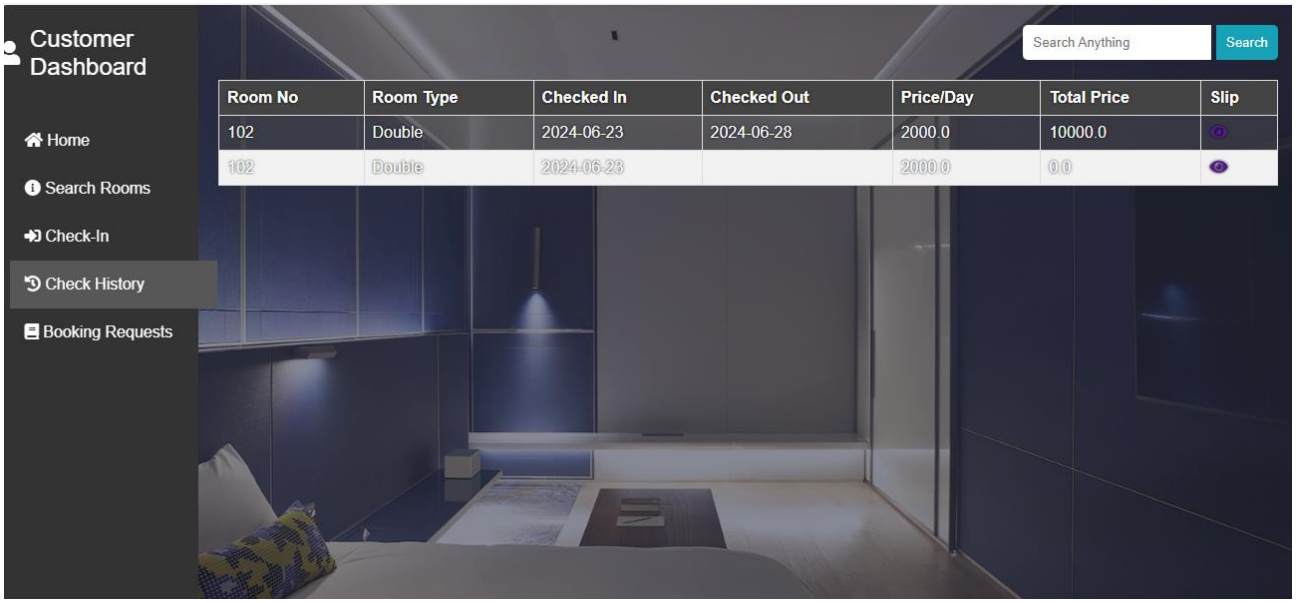

Admin Dashboard
Home
Add Employee
Employee Info
Earning Log
Total Earnings
Employee Info
Search Employee Info
Employee NID | Name | Email | Phone | Address | Password | Action
NID123456 | John Doe | john.doe@example.com | 1234567890 | 123 Main St | password1
NID234567 | Jane Smith | jane.smith@example.com | 0987654321 | 456 Elm St | password2
NID345678 | Michael Brown | michael.brown@example.com | 1122334455 | 789 Oak St | password3
NID456789 | Emily Davis | emily.davis@example.com | 6677889900 | 101 Pine St | password4
NID567890 | David Wilson | david.wilson@example.com | 2233445566 | 202 Maple St | password5
132454323 | New Emp | new@gmail.com | 23454 | E-11 | 1234

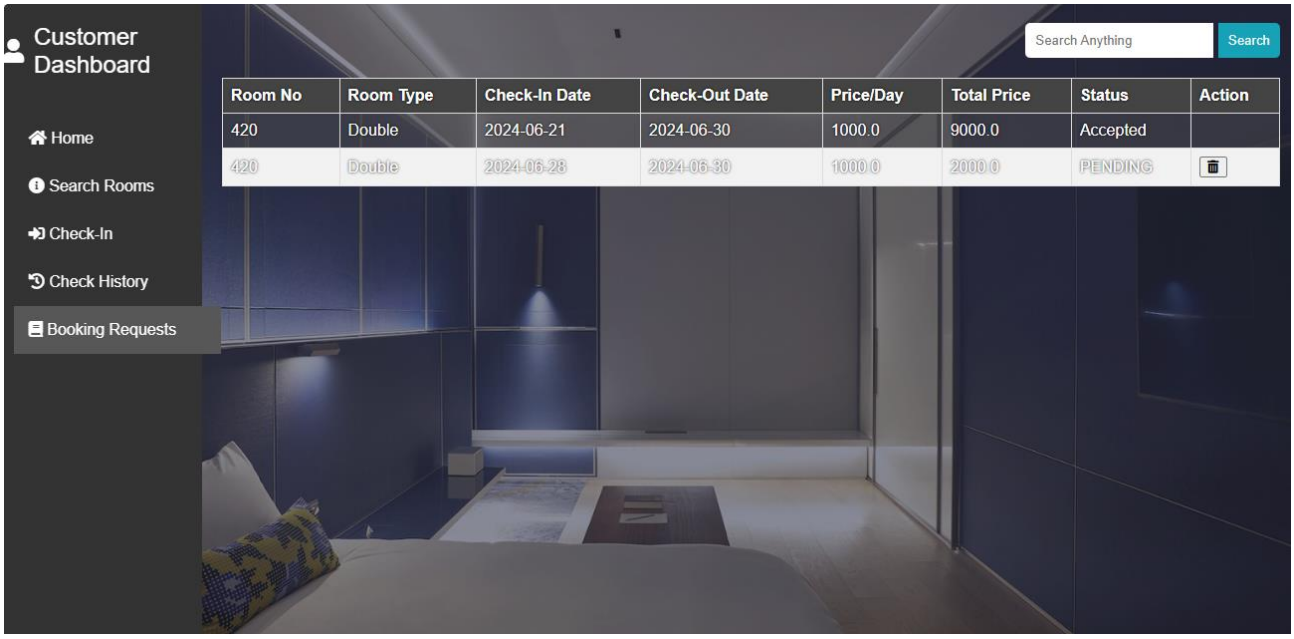

Admin Dashboard
Home
Add Employee
Employee Info
Earning Log
Total Earnings
Total Earnings
NID | Name | Room No | Room Type | Checked In | Checked Out | Price/Day | Total Price
15222584 | John Doe | 102 | Double | 2024-06-23 | 2024-06-28 | 2000.0 | 10000.0
15222589 | hamdan | 102 | Double | 2024-06-23 | | 2000.0 | 0.0
Total Earnings: | 10000.0


Admin Dashboard
Home
Add Employee
Employee Info
Earning Log
$ Total Earnings
Total Earnings
Start Date: mm/dd/yyyy End Date: mm/dd/yyyy
Show Results
Room ID
Room No
Times Booked
Total Earnings
2
102
2
10000.0
Total Earnings:
10000.0


Customer Dashboard
Home
Search Rooms
Check-In
Check History
Booking Requests
Search by keyword
Search
Room Search
Room No
Type
Capacity
Price/Day
Room Status
Action
101
Single
1
1500.0
Available
Book
102
Double
2
2000.0
Booked
103
Suite
4
5000.0
Booked
420
Double
2
1000.0
Available
Book


Customer Dashboard
Home
Search Rooms
Check-In
Check History
Booking Requests
Check In Info
Room No:
420
Room Type:
Double
Bed Capacity:
2
Price/Day:
1000.0
Check In Date:
mm/dd/yyyy
Check Out Date:
mm/dd/yyyy
Total Days:
Total Price:


Customer Dashboard
Home
Search Rooms
Check-In
Check History
Booking Requests
Search Anything
Search
Room No
Room Type
Checked In
Checked Out
Price/Day
Total Price
Slip
102
Double
2024-06-23
2024-06-28
2000.0
10000.0



9. Work Division Table

Member	Work
Abdullah Ilyas	Creating and Setup Spring MVC project Use cases 4-6
Abdul Qadus	Updating diagrams, Data Base setup Use case 7-9
M. Hamdan Rauf	Documentation Use case 1-3