DIT945 H14 Model Driven Software Development
Department of Computer Science and Engineering
University of Gothenburg

Final Report

Hotel Management System

Group 17

Ale Lotström
Andy Dang
Georgij Marchulija
Shireen Yusur
Sinan Abdulwahhab
Yacoub Sattar
Yazen Raad

1. Introduction
    - The purpose and objective
2. Scope
    - Definitions
    - Explicit features that are excluded
3. Project Overview
    - Product Functions
    - System Functional requirements
    - User Specific Requirements
        - Customer:
        - Manager
        - Guest
    - Non-functional requirements
4. Management
    - Team member responsibilities
    - Division of labor:
        - Ale Lotström:
        - Andy Dang:
        - Georgij Marchulija :
        - Shireen Yusur :
        - Sinan Abdulwahhab:
        - Yacoub Sattar:
        - Yazen Raad:
5. Early Stage Mindmap
6. Concepts:
7. Domain Model
8. Vocabulary:
9. Use case diagram:
    - Use cases description
10. Responsibilities of group members
11. Complete Use-Cases.
    - Textual description
        - Create Booking
        - Add / Remove / Change room:
        - Checkout and pay
        - Change Room
        - Cancel Booking
12. Business rules:
13. System Operations:
14. Activity diagram

1. Introduction

The hotel management system (HMS) is designed to serve both end-users and management based on the software requirements specification. The latter gives a clear picture of what is expected of the hotel management system that is to be modeled. Moreover, a clear knowledge and perception of the system and its functionalities will allow the correct software to be developed, and it will provide the target-user of what they need. Thus, the software requirement specification provides the functional requirement for the system, and that enables the HMS to be designed.

1.1. The purpose and objective

The aim of this project is to design and develop a hotel management system for running a hotel business and make it efficient and reusable for different hotels.

The objectives of the HMS is to provide a system that can be used by different hotels. The system will be able to manage different services to take care of all customers and guests in quick manner. Also, the system should be user appropriate, explicit, easy to use and follow, and lastly have an overall high subjective satisfaction from the target user perspective.

2. Scope

The system serves multiple end users, each has different privileges as how they are able to use the system. These privileges include normal access for customers, where the they can perform actions such as book, cancel or pay in advance for a room or a service. A customer becomes a guest when they physically check in to the hotel system. Bookings can be made either on the hotel's website, on the phone or at the reception at the hotel. Other privileges such as managing rooms, services and payments, are only allowed for special users, namingly the manager. A manager of the hotel manages the booking system, including the availability of rooms and what services are provided. The guest checks in and checks out of the hotel and the system registers a guest to a specific room number at check in. There is also a banking component that verifies a customer or guest payment.

Our motivation for this system is that it is flexible. This is a simple hotel management system that only includes basic functions needed for a hotel, so it can be used in any hotel. We want to focus on first creating a good core and then adding additional functions if we have time while still making the system flexible.

2.1. Definitions

We have chosen to group all meals, entertainment, special facilities, laundry etc. as types of services, as they all work on the same basis: It is something added to the total cost of a guests stay.

A booking includes both rooms and all available services that a specific customer chooses.

By customer information we refer to the first name, last name, personal number, phone number, address, city, zip code, email address.

2.2. Explicit features that are excluded

There are several features that we have chosen not to include:

- Traveling services such as hotel shuttles, taxis and airport pickups.
- Safety and security systems such as alarms, camera surveillance, elevator services etc.
- In-house amenities such as telephone, lighting and heating systems.
- Economical factors such as hotel revenue and employee salaries.
- Specifics of services i.e. a certain meal price.
- Statistics of hotel occupancy over time.
- The employees which are in charge of providing services.
- The note of "Not used" in some of the system requirements that mean excluded requirement. We decided as a group to not use some of the features or requirements that will not impact the HSM at all since it is not high priority.
- The note of "Not implemented" in some of the system requirements that mean excluded. We decided as a group to not implement some of the system requirements that are not high priority for the system hand-in. We want to finish the fundamental requirements, rather than trying to implement additional functions regarding the time-consuming process. For instance, we excluded the "Services" requirements since it not impacts the Hotel System. However, these features are still mentioned in the system requirements because we believe they are still important to be used in the future.
- Managerial branch e.g employee.

3. Project Overview

This section presents the software requirements specification of the HMS from a general high level perspective.

3.1. Product Functions

This section presents the functional requirements and the non-functional requirements. We have chosen to divide the requirements into system specific ones that the system itself is responsible for, and then we have chosen user specific requirements that each user (Manager, Customer, Guest) deal with individually. We are not including requirements relating to the list of excluded features listed above.

3.1.1. System Functional requirements

The functional requirements define the essential actions that the system must perform.

The system shall:

1. Calculate the total cost of a booking.
2. Calculate the total cost for a guest bill.
3. Store customer information (Not implemented)..
4. Store the number of occupants per room.
5. Display the default room rate (Not implemented). .
6. Store room numbers.
7. Establish payment transactions with banks.
8. Store payment history (Not implemented).
9. Create receipts for payment transactions (Not implemented).
10. Track room availability.
11. Track service availability (Not used).
12. Store bookings.
13. Log errors (Not implemented).
14. Store employee accounts.
15. Log employee availability (Not used).
16. Offer different employee privileges (Not used).
17. Generate a unique confirmation number per booking.
18. Assign room number to guests.
19. Display whether or not the room is guaranteed.
20. Automatically cancel non-guaranteed the reservations if the customer has not provided their credit card number by 6:00 pm on the check-in date.
21. Mark the guaranteed rooms as “must pay” after 6:00 pm on the check-in date.

22. Prevent customer from cancelling booking after 10 AM of check-in date.
23. Store the expected check-in date and time.
24. Store the expected checkout date and time.
25. Charge the guest for an extra night if they check-out after 11:00 a.m.
26. Store customer/guest feedback (Not used).
27. Validate customer information (Not implemented).

3.1.2. User Specific Requirements

Within the system, there are user specific requirements that are as follows:

a. Customer:

The customer shall be able to:

1.1. Search different types of rooms.
1.2. View available services (Not used).
1.3. Book a room online.
1.4. Book a room over the phone.
1.5. Book a room at reception.
1.6. Cancel booking
1.7. Re-book.
1.8. Add services to a booking (Not used).
1.9. Add personal information when booking.
1.10. Add payment method when booking.
1.11. Give feedback.
1.12. Pay in advance.
1.13. Choose payment method.
1.14. Book a room for third party.

b. Manager

The manager shall be able to:

1.1. Login with unique user id and password.
1.2. View availability of rooms.
1.3. View availability of services.
1.4. Add room availability.
1.5. Remove room availability.
1.6. Add new service.
1.7. Remove service.
1.8. Change price of a room.

1.9. Change type of a room.
1.10. Change price of a service.
1.11. Create employee accounts.
1.12. Set privileges for employee accounts (Not implement).

c. Guest

The guest shall be able to:

1.15. Check in at the reception.
1.16. Check out at the reception.
1.17. Request services during stay (Not used).
1.18. Request to change room.
1.19. Request to extend stay.
1.20. Request to re-book.
1.21. Pay upon check-in.
1.22. Pay upon check-out.

3.1.3 Non-functional requirements

1. Payment transactions have to be secure.
2. Guest/customer information should be confidential.
3. Interfaces should be easy to use.
4. New manager should be able to use the system after 20 minutes of familiarization.
5. Customers should feel comfortable using the system in less than a minute.
6. System should not crash more than once per 5000 hours.
7. The downtime should not be more than 3 minutes 5000 hours.
8. The hotel's website should work on major (i.e FireFox, Chrome, IE, Safari) browsers.
9. The login data should be verified within 3 seconds.
10. The load time for user interface display shall take no longer than two seconds.

4. Management

4.1 Team member responsibilities

1. Each member is responsible for delivering in all work areas, papers and development.
2. Deliver his/her task on time once we divide all the work.
3. Invest up to 16 hours each week for the weekly assignments.
4. Attend meetings on time.
5. Communicate effectively with other team members.
6. Take responsibility for the project as a whole.
7. Have fun.

4.2 Division of labor:

We have divided the requirements into seven categories in which they relate to: Payment, Check in and Check out, Booking, Login/Accounts, Rooms/Services, Storing and finally the remaining requirements which did not fit within the previous categories. Each member is going to be in charge of a different category but this is a preliminary division and it is subject to change since the amount of implementation have not fully been considered.

1. **Ale Lotström:**

Payment System:

Calculate the total cost of a booking.
Calculate the total cost for a guest bill.
Establish payment transactions with banks.
Store payment history.
Create receipts for payment transactions.
Generate a unique confirmation number per booking.
Assign room number to guests.

Customer:

Pay in advance

Non-functional:

Payment transactions have to be secure.

2. **Andy Dang:**

Check in and check out Guests:

Check in at the reception.
Check out at the reception.
Request services during stay.

Request to change room.
Request to stay longer than booked.
Pay upon check-in.
Pay upon check-out.

3. **Georgij Marchulija :**

Booking System:

Prevent guest from cancelling booking after 10 AM of check-in date.

Customer:

Book a room online.
Book a room over the phone.
Book a room at reception.
Add services to a booking.
Add personal information when booking.
Add payment method when booking.
Give feedback.
Choose payment method.
Book a room for third party.

Non-functional:

Guest information should be confidential.

4. **Shireen Yusur :**

Login/accounts Manager:

Login with unique user id and password.
Create employee accounts .
Set privileges for employee accounts (Not implemented).

Non-functional:

The login data should be verified within 3 seconds.

5. **Sinan Abdulwahhab:**

Rooms/services System:

Display the default room rate.
Track room availability.
Track service availability.

Customer:

Search different types of rooms.
View available services.

Manager:

View availability of rooms.
View availability of services.
Add room availability.
Remove room availability.
Add new service.
Remove service.
Change price of a room.
Change type of a room.
Change price of a service.

6. **Yacoub Sattar:**

The remaining requirements System:

Offer different employee privileges (Not used).
Display whether or not the room is guaranteed.
Automatically cancel non-guaranteed bookings if the guest has not provided their credit card number by 6:00 pm on the check-in date.
Mark the guaranteed rooms as "must pay" after 6:00 pm on the check-in date.

Non-functional:

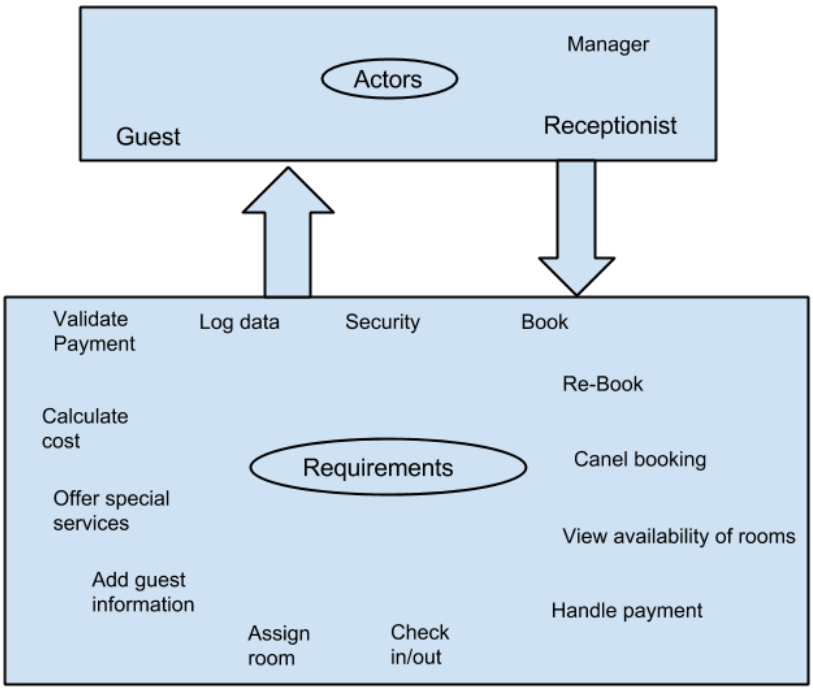
Interfaces should be easy to use (Not used).
New manager should be able to use the system after 20 minutes of familiarization.
Guests should feel comfortable using the system in less than a minute.
System should not crash more than once per 5000 hours.
The downtime should not be more than 3 minutes 5000 hours.
The hotel's website should work on major (i.e FireFox, Chrome, IE, Safari) browsers (Not used).
The load time for user interface display shall take no longer than two seconds.

7. **Yazen Raad:**

Storing System:

Store customer information.
Store the number of occupants per room.
Store room numbers.
Store bookings.
Log errors.
Store employee accounts.
Log employee availability (Not implemented).
Store the expected check-in date and time.
Store the expected checkout date and time.

5. Early Stage Mindmap



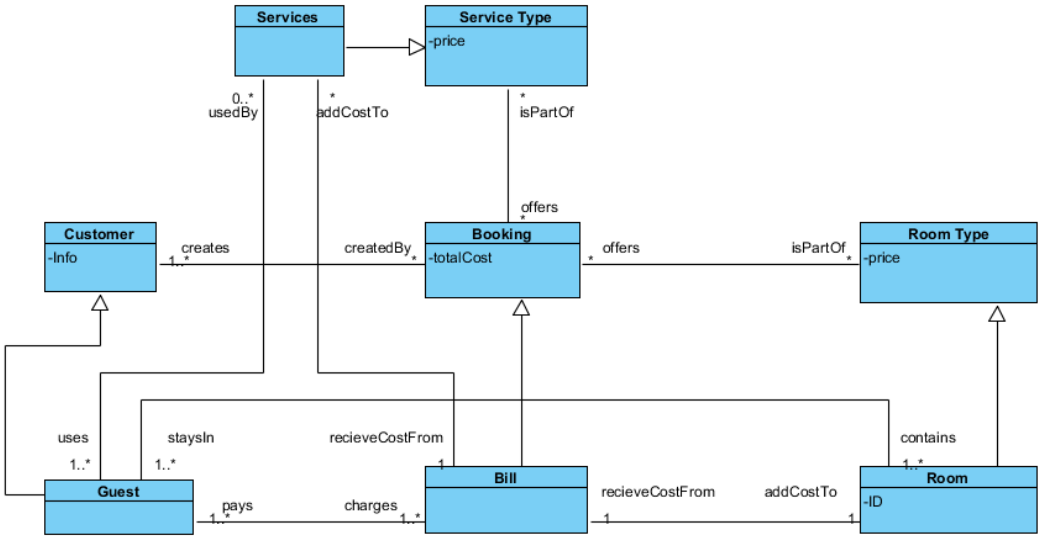
6. Concepts:

‘Customer’ interacts with all booking procedures of rooms and services and is able to pay in advance for a booking.
‘Guest’ interacts with the system by checking in, checks out, assigned to a room, stays in a room or pay his bill.

‘RoomType’ and **‘ServiceType’** are separate concepts because one of them can be booked or used without the other. They are both non-physical and part of the larger concept ‘Booking’ because they are usually booked together and multiple services and rooms adds to the total cost of a specific booking.
‘Room’ and ‘Service’ are the actual physical room and services used during a stay by a guest. They both add to the cost of a guest’s total ‘Bill’.
‘Manager’ is responsible for adding and removing both rooms and services to the system as well as changing e.g. price for them.

Domain Model
We have decided to keep the domain model simple, without extended use of attributes and operations in order to focus it on the main relations of the concepts. Some things that are missing here is the factor of time, meaning a guest only stays in a room for a certain amount of time. Also, the availability attribute of room carries much more importance than what is shown in this domain model. Further details will be added in the class diagram. We have decided not to use a receptionist at all because we consider it to simply be an actor between the guest and the system that has no added functionality, it is just a transition phase that we consider to be part of the system. We have also chosen not to include the manager and his ability to add/remove rooms in the domain model, as it is not essential for the booking use-case and we want to keep our domain model focused and simplistic.

7. Domain Model



8. Vocabulary:

System - a software which is used to manage bookings, guests, special services an rooms` availability of the hotel.

User/Actor - anyone using the system (i.e. customer, guest, manager).

Manager - hotel employee which has access to the system with CRUD privileges.

CRUD - an acronym for Create, Read, Update, and Delete.

Customer- person who has paid for booking, or is an associate to who made a booking for a guest.

Guest- person who stays at the hotel.

Employee- a person employed by hotel management for wages.

Cleaner - employee of the hotel responsible for cleanliness of the rooms (Not used).

Room- accommodation provided by hotel in exchange for money.

Booking - an act of reserving accommodation and service by customer.

Re-booking - an act of reserving accommodation and services by a customer after such action has already been performed.

Cost- amount of money necessary to complete a booking.

Payment method- option available for guest to deposit money for booking or additional services.

Services - different services that hotel provides for its guests for extra fee, such as, cleaning, laundry, breakfast, ironing, cab and others of the kind (Not used).

Free Services - services which are included in the booking fee, such as cleaning (Not used).

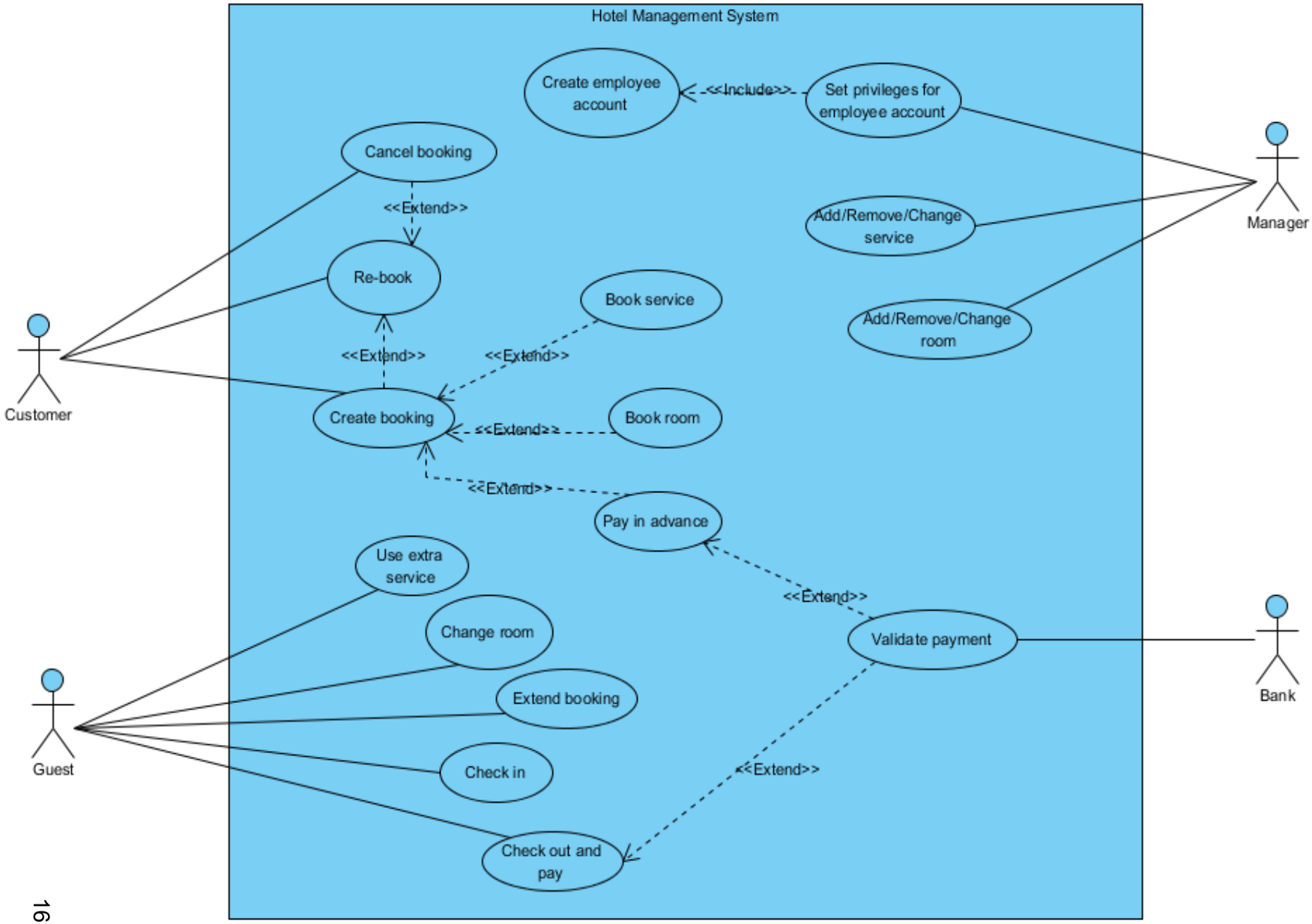
Cancellation- an act of removing a reserved accommodation by guest.

Guaranteed reservation- reservation with valid credit card assigned to it.

Room Type- ~~property of the room such as having a special view, smoking/non smoking, disabled friendly, pets friendly or single/double-bed.~~
We decided to go with some simpler room types. With the way our code works, this would take a lot of time to code. There are now regular, economy, family and luxury rooms instead. This would make it simpler. You could have floors with rooms of one room type.

9. Use case diagram:

In the use case diagram we have four actors Customer, Guest, Manager and Bank. The customer can only create a booking online, while the guest can request from the system to create a booking for the guest in any other way. In the same way the customer can only pay online while the guest can request from the system to make the payment for the guest at the check out.


Hotel Management System
Create employee account
<<Include>>
Set privileges for employee account
Cancel booking
<<Extend>>
Re-book
Add/Remove/Change service
Manager
Book service
Add/Remove/Change room
<<Extend>>
<<Extend>>
Customer
Create booking
<<Extend>>
Book room
<<Extend>>
Pay in advance
Use extra service
<<Extend>>
Change room
Validate payment
Bank
Extend booking
Guest
Check in
<<Extend>>
Check out and pay

9.1 Use-case descriptions

Name : Create booking
ID : 001
Actor : Customer
Goal : Create a booking of a room
Priority : High
Description : The customer enters search criteria into the system. The system searches for the rooms matched by criteria and displays them on the website. User selects desired room and system creates the booking for the guest.

Name : Book room
ID : 002
Actor : Customer
Goal : Book a room
Priority : High
Description : The customer enters search criteria into the system. The system searches for the rooms matched by criteria and displays them on the website. User selects desired room and system books the room for the customer.

Name : Book service
ID : 003
Actor : Customer
Goal : Adding extra services.
Priority : High
Description : The customer enters the hotel website into the system. The system searches for the rooms matched by criteria and displays them on the website. User selects desired room and system books the room for the customer.

Name : Cancel booking
ID : 004
Actor : Customer
Goal : Cancel an already made booking
Priority : High
Description : A customer wants to cancel a booking he has made. The customer provides the information about the booking. the system checks to cancel the booking.

Name : Extend booking for guest
ID : 005
Actor : Guest
Goal : Extends a guest's staying at the hotel.
Priority : Low
Description : A guest wants to extend his staying at the hotel by a time. provide his information to the to check for availability.

Name : Re-book
ID : 006
Actor : Customer
Goal : Re-book a booking for a customer.
Priority : Medium
Description : A customer wants to re-book his booking at the hotel. The system checks for room availability to see if the room is available at the wanted time. If available, the system re-books the booking.

Name : Check in
ID : 007
Actor : Guest
Goal : Check-in a guest
Priority : High
Description : The guest arrives at the hotel reception and requests to check in.

Name : Check out / Pay
ID : 008
Actor : Guest
Goal : Check-out a guest and make payment
Priority : High
Description : The guest has stayed at the hotel and is ready to leave. the system checks the guest out and provide him with the bill.

Name : Change room for a guest
ID : 009
Actor : Guest
Goal : Change room for a guest.
Priority : Low
Description : The guest wants another room. System changes room for the guest if a room is available.

Name : Change service
ID : 010
Actor : Manager
Goal : Change the specifics of services
Priority : Low
Description : Manager accesses system`s User-Interface and changes specifics of services.

Name : Add service
ID : 011
Actor : Manager
Goal : Add a service
Priority : Low
Description : Manager accesses system`s User Interface and adds new services.

Name : Remove service
ID : 012
Actor : Manager
Goal : Remove a service
Priority : Low
Description : Manager accesses system`s User Interface and removes services.

Name : Change room
ID : 013
Actor : Manager
Goal : Change room
Priority : Low
Description : Manager accesses system`s User Interface and changes the price of the room

Name : Add room
ID : 014
Actor : Manager
Goal : Add a room in the system
Priority : Low
Description : Manager accesses system`s User Interface and adds rooms to the system.

Name : Remove room
ID : 015
Actor : Manager
Goal : Remove a room in the system
Priority : Low
Description : Manager accesses system`s User Interface and removes rooms to the system.

Name : Create employee account
ID : 016
Actor : Manager
Goal : Create account for hotel employees
Priority : Medium
Description : Manager accesses system`s User Interface and creates account for a new employee

Name : Set privileges for employee account
ID : 017
Actor : Manager
Goal : Set privileges for employee account
Priority : Low
Description : Manager accesses system`s User Interface and grants privileges to employee accounts according to their job position

Name : Validate payment
ID : 018
Actor : Bank
Goal : Validate payment
Priority : Medium
Description : Bank checks if the credit card information is correct and if there is sufficient amount of money on the account attached to the card.

Name : Pay in advance
ID : 019
Actor : Customer
Goal : Make a payment in advance.
Priority : Medium
Description : The customer pays in advance.

10. Responsibilities of group members

1. **Ale Lotström.**
   Use-Case name: Validate Payment
   Use-Case name: Pay in Advance
   Use-Case name: Receive Payment

2. **Andy Dang:**
   Use-Case name: Check in guest
   Use-Case name: Check out guest
   Use-Case name: Create employee account

3. **Georgij Marchulija :**
   Use-Case name: Add/Remove services
   Use-Case name: Create Booking
   Use-Case name: Choose payment method

4. **Shireen**
   Use-Case name: Set privileges for employee accounts
   Use-Case name: Change service price
   Use-Case name: Change the room for a guest

5. **Sinan Abdulwahhab:**
   Use-Case name: Search room type
   Use-Case name: Change room price
   Use-Case name: Change type of a room

6. **Yacoub Sattar:**
   Use-Case name: Re-book
   Use-Case name: Book service
   Use-Case name: Book Room

7. **Yazen Raad:**
   Use-Case name: Check service availability
   Use-Case name: Change service
   Use-Case name: Change type of the room
   Use-Case name: Add/Remove room

11. Complete Use-Cases.

Textual description

Create booking use-case starts when a customer decides to book a room at hotel. Customer can access system online through website or via reception (which we consider as a part of the system). Customer provides his preferences of room type, check-in and check-out dates. After system displays search results, customer checks the room that fits him and proceeds to booking interface. There he/she adds personal details and is asked to agree on terms of conditions of the hotel. After agreeing on conditions, customer is redirected to choose payment method where he/she can choose whether to attach credit card details to the booking and create guaranteed-reservation, or, to pay on spot. After payment method is set, system confirms reservation and generates confirmation number, which is sent to customer's email.

- Create Booking
- System creates a booking of the room and services for a guest
- Customer
- Preconditions: -
- Main flow:
    1. Customer accesses the system and enters desired check-in and check-out dates and adds room type and services.
    2. Assume: System filters the results and returns available options by displaying on the User-Interface and calculate total costs
    3. System generates new interface with empty fields which require customer to provide personal information
    4. Customer fills in the empty fields with his personal information such as Name, Last name, date of birth, address and email.
    5. Assume: System confirms that all fields are filled, Name and Last name have only latin letters, email field value has a correct format and the age is above 18 years.
    6. System displays "Accept terms and Conditions" message with a clickable check-box
    7. Assume: Customer accepts "service terms and conditions"
    8. System stores the customer's information.
    9. System displays payment options on the website.
    10. Assume: Customer chooses "pay now" option.
    11. System redirects the customer to the bank's system to validate payment.
    12. Assume: the validation of the customers info with the bank is valid.
    13. System displays booking confirmation.
    14. Assume: the customer confirms the booking.
    15. System stores selected room as "Booked" in the system to prevent double-booking.
    16. System generates unique confirmation number and sends the receipt to the customer.
    17. Customer receives the receipt and use case ends.

Alternative flows:

2a: No room type available in a given period of time, customer changes the preferences:

1. System displays “Dates are booked, try another date” message on the user-interface
2. Start from action step 1

5a: Customer does not meet age requirements:

1. System displays “You do not meet age requirement” message on the user-interface next to “date of birth” field.
2. System stores the correctly filled in information.
3. Start from action step 4

5c: Customer forgets to fill one of the required fields

1. System displays message “Information missing” message on the user-interface next to empty field
2. System stores the correctly filled in information.
3. Start from action step 4

5d: Customer has unsupported values in name or last name

1. System displays message “Illegal value, please use latin alphabet” message on the user-interface next to Name or Last Name field
2. System stores the correctly filled in information.
3. Start from action step 4.

5e: Customer provides wrong email format

1. System displays message “wrong email” message on the user-interface next to email field.
2. System stores the correctly filled in information.
3. Start from action step 4.

7a: Customer does not tick checkbox and clicks next

1. System displays message “Please Accept our service terms and conditions to proceed”.
2. Start from action step 6.

10:. Customer chooses “pay upon check-out” option:

1. Start from action step 13.

12a: Payment is invalid.

Start from action step 9.

14a: Customer does not confirm booking

1. Use case ends.

- Postconditions:

  If customer enters valid check in/check out dates and they are available and the customer information was provided correctly and the customer passes the age restriction and payment was successful or customer chooses to pay upon check-out and customer confirms booking then a room is booked.

  Else a room is not booked.

- Remove room:
- System removes a room.
- Manager
- Preconditions: -
- Main flow:
  1. Manager logs into the system`s user interface.
  2. Assume: Manager login information is valid.
  3. Assume: System displays an overview of all rooms with room numbers.
  4. System highlights whether rooms are occupied, booked or empty.
  5. Manager selects a room and presses ‘delete’ button on the user-interface.
  6. Assume: room is not booked or occupied.
  7. System displays confirmation message.
  8. Assume: Manager chooses ‘confirm’ option on the user-interface.
  9. System marks selected room as permanently unavailable.
  10. System displays updated overview of rooms.

2a: Login information is invalid.
  1. Start from action step 1.

3a: No rooms have been added.
  1. System displays empty with message “please add rooms”.
  2. Start use case “add rooms”.

6a: Room is occupied or booked.
  1. System ignores deletion and displays “Room can not be removed”.
  2. Start from action step 4.

8a: Manager chooses to cancel.
  1. Start from action step 4.

- Postconditions:

  If manager successfully logs in and there are rooms in the system and at least one room is not occupied or booked and manager presses deletes and confirms deletion then a room is removed.

  Else no rooms are removed.

- Checkout and pay
- The guest checks out of the hotel and pays the total bill
- Guest
- Preconditions: -
- Main flow:
  1. The guest requests to check-out.
  2. System requests guest`s room number.
  3. Guest provides room number to the system.
  4. Assume: room number is valid.
  5. System checks whether the guest paid for the booking or not.
  6. Assume: the guest haven't paid for the booking in advance.
  7. The system calculates the total bill and displays it for the guest.
  8. Assume: the guest pays the bill with card.
  9. The system requests payment validation from bank.
  10. Assume: the payment transaction is successful.
  11. The system registers payment and provides a receipt to the guest.
  12. The system checks out the guest and the use case ends.

- Alternative Flows:

  4a. The room number is not valid:
  1. The system displays an error.
  2. Start from action step 3.

  4b. Guest is unsure of room number:
  1. Guest provides first and last name.
  2. Start from action step 3.

  6a. The guest already paid beforehand and there are no added service costs:
  1. Start from action step 11.

  6b. The guest already paid beforehand and there are added service costs:
  1. Start from action step 7.

8a. The guest pays with cash:

1. Start from action step 11.

10a. The payment transaction is unsuccessful:

1. The system displays an error.
2. Start from action step 9.

- Postconditions:

  If the guest is checked in and provides the correct room number and check-out date is correct and payment is valid or total bill was paid in advance then the guest is checked out.

  Else guest is not checked out.

- Change Room
- The system changes the room for a guest
- Guest,
- Preconditions: -
- Main flow:
  1. Guest requests to change room
  2. System requests current room number of the guest
  3. Guest provides room number.
  4. Assume: room number is valid.
  5. System requests desired room type from the guest
  6. Guest provides description of the room type of his preference.
  7. System checks availability of the new room.
  8. Assume: system has found matching room.
  9. System marks new room as “occupied” and assigns it to the guest.
  10. System marks previously assigned room to guest as “available” and use case ends.

  Alternative Flows:

  4.a: System could not find room number

  1. Start from action step 2

  8.a: System could not find matching room

  1. Start from action step 5

- Postcondition:

  If the current room number is valid and the new room of choice is available then the guest changes room.

  Else guest remains in the same room.

- Cancel Booking
- A customer wishes to cancel a booking.
- Actor: Customer
- Precondition: -

- Main Flow:
    1. The customer requests cancellation of the booked room
    2. The system requests a confirmation number
    3. The customer provides confirmation number of the booking
    4. Assume: System found matching confirmation number
    5. The system asks customer to confirm his First Name and Last Name
    6. The customer provides his First Name and Last Name.
    7. Assume: System confirms matched First Name and Last Name with confirmation number.
    8. Assume: Guarantee booking was selected
    9. The system checks for check-in date, and current date and time.
    10. Assume: Cancellation is requested before 10 AM of the check-in date
    11. The system sends refund notice with customer details to bank
    12. The system cancels the booking
    13. The system marks the room as “available”
    14. Use case ends

- Alternative Flow:

    4a: System did not find matching confirmation number
    1. System display “confirmation number was not found”.
    2. Start from action step 2

    6a: System did not find matching First Name and Last Name
    1. System display “ The name did not match”.
    2. Start from action step 5

    8a: Non-Guarantee booking was selected
    1. Start from action step 11

    10a: Cancellation is requested at/after 10 AM of check-in date
    1. The use-case ends.

- Postcondition**:**
  If customer has a booking and requests cancellation and provides correct confirmation ID and verifies his first and last name and the booking is guaranteed and the time is before 10 AM of check in date, then booking is cancelled.

  Else the booking is not cancelled.

- Check in
- A guest checks into the hotel
- Actor: Guest
- Precondition: -

- Main Flow:
  1. Guest request from the reception to check in.
  2. The system requests a confirmation number.
  3. The customer provides confirmation number of the booking.
  4. Assume: System found a matching confirmation number.
  5. The system compares the current date with the check in date in the booking.
  6. Assume: The current date is the same as the check in date in the booking.
  7. System assign a free physical room to the guest.
  8. System assign the physical room as occupied.
  9. Use case ends.

- Alternative flow 1:
  4: System can not find a matching number.
  1. System displays that the confirmation number was not found.
  2. Start from step 3.

- Alternative flow 2:
  6: The current date is not the same as the check in date in the booking.
  1. System displays that you can't check in before the check in date in the booking.
  2. Use case ends.

- Postcondition**:**
  If customer has a booking and provides correct confirmation number, customer is checked in.

12. Business rules:

1. A guest only stays in a room for a certain amount of time. When booking, a check-in date and check-out date is always specified.
2. There is a maximum amount of rooms. Bookings can only be made for the maximum amount of rooms at any given time.
3. A guest cannot check in to a room that is not booked. Any check-in is perceived by a booking. Even though a guest may book and check-in in person at the same time, the same principle applies.
4. A customer can book a room for a guest that is not himself. For example a company can book for their employees.
5. The age of the customer must be 18 or above in order to book. If a customer is under 18, someone else needs to book for him/her.

13. System Operations:

System Operations:

validateCustomerInfo(customer):bool - this operation checks if the information provided by customer is valid. It ensures that for example birth date is in logical range (e.x. it cannot be 2025), or that name does not contain illegal symbols such as #N@me.

checkAgeRestriction(customer):bool - this operation checks is customer is over 18 years old.

storeCustomerInfo(customer) - this operation receives customer information as a parameter and stores it

generateConfirmationNumber(customer):int - this operation generates unique confirmation number

registerPayment(payment) - registers a payment into the system.

createReceipt(confirmationNumber): - this operation creates a recept after the payment is confirmed

storeBookingInfo(confirmationNumber, customer): - this operation takes two parameters and assignms confirmation number to customer information.

checkRoomAvalibility():bool -this operation checks if there are any rooms not registered to anyone, and returns a boolean value.

checkServiceAvalibility():bool -this operation checks if the requested services are available for booking , and returns a boolean value.

retrieveBookingInfo(confirmationNumber):string - retrieves information about a booking.

retrieveCustomerInfo(confirmationNumber):customer - retrieves information about a customer.

isPaid(roomNumber):bool - check if a room has been paid for in advance.

calculateBookingCost(roomType[], serviceType[]):double - calculates the cost of a booking with a certain room type and services.

calculateTotalBill(confirmationNumber, service[]):double - calculates the total cost a customer has to pay

setAvailabilityOfRoom(roomNumber):bool - this operation changes the availability of the room to “available” or “unavailable”

setRoomNumber(guest, int):int - this operation assigns a room number to the guest.

getRoomNumber(guest):int - gets the room number of a room where the guest is staying.

Customer Operations:

createBooking(roomType[], serviceType[], checkInDate, checkOutDate):int - this operations receives 2 parameters from the customer, which are desired room type and service type. It creates reservation.

cancelBooking(confirmationNumber): - this method receives confirmation number, checks for a matching number in the system and cancels booking.

re-Book(confirmationNumber, new roomType, new serviceType, new checkInDate, new

checkOutDate):int - this method receives confirmation number, desired room type and services type. It validates confirmation number with existing booking, then it cancels existing booking, creates new booking and generates new confirmation number.

payInAdvance(): - this operation redirects the customer to the banking system.

confirmBooking(customer, booking):bool - this operation will assign booking information to customer

addInformation(firstName, lastName, personalNumber, address, zipCode, city, state, country, email, phoneNumber): - this operation stores personal information of customer

changeInformation(customer, firstName, lastName, personalNumber, address, zipCode, city, state, country, email, phoneNumber): - this operation alters personal information of the customer

Guest Operations:

checkIn(firstName, lastName): - Checks in the guest with the given first name and last name.

checkOut(roomNumber, payment): - Checks out a guest, after receiving the guest's room number as parameters.

changeRoom(roomNumber, new roomType): - Changes a room for a guest, after taking the guest's room number and room type as parameters.
addService(guest, service[]): - this operation assigns array of services to the guest

Managerial Operations:

addRoom(price, type, number): - Adding a room to the hotel, provide price type number and system returns conformation.

removeRoom(number): - Remove a room from the hotel, system returns conformation.

changePriceOfRoom(number): - Change the price of a room which was registered in the system, provide the room that needs to be changed.

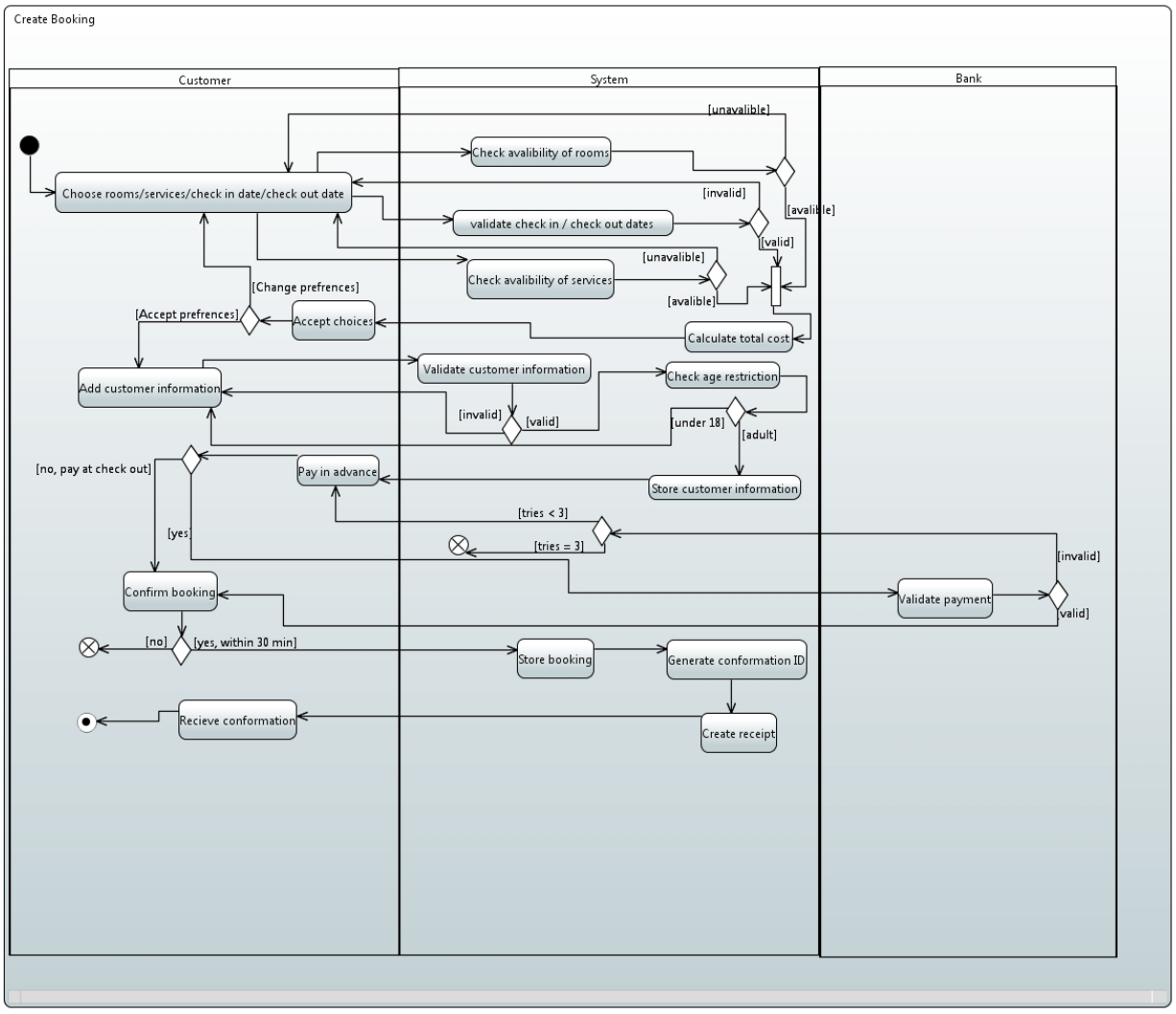
changeTypeOfRoom(number): - Change the type of the room which was registered in the system, provide the room that needs to be changed.

changeRoomNr(room): - Change the number of the room which was registered in the system, provide the room that needs to be changed.

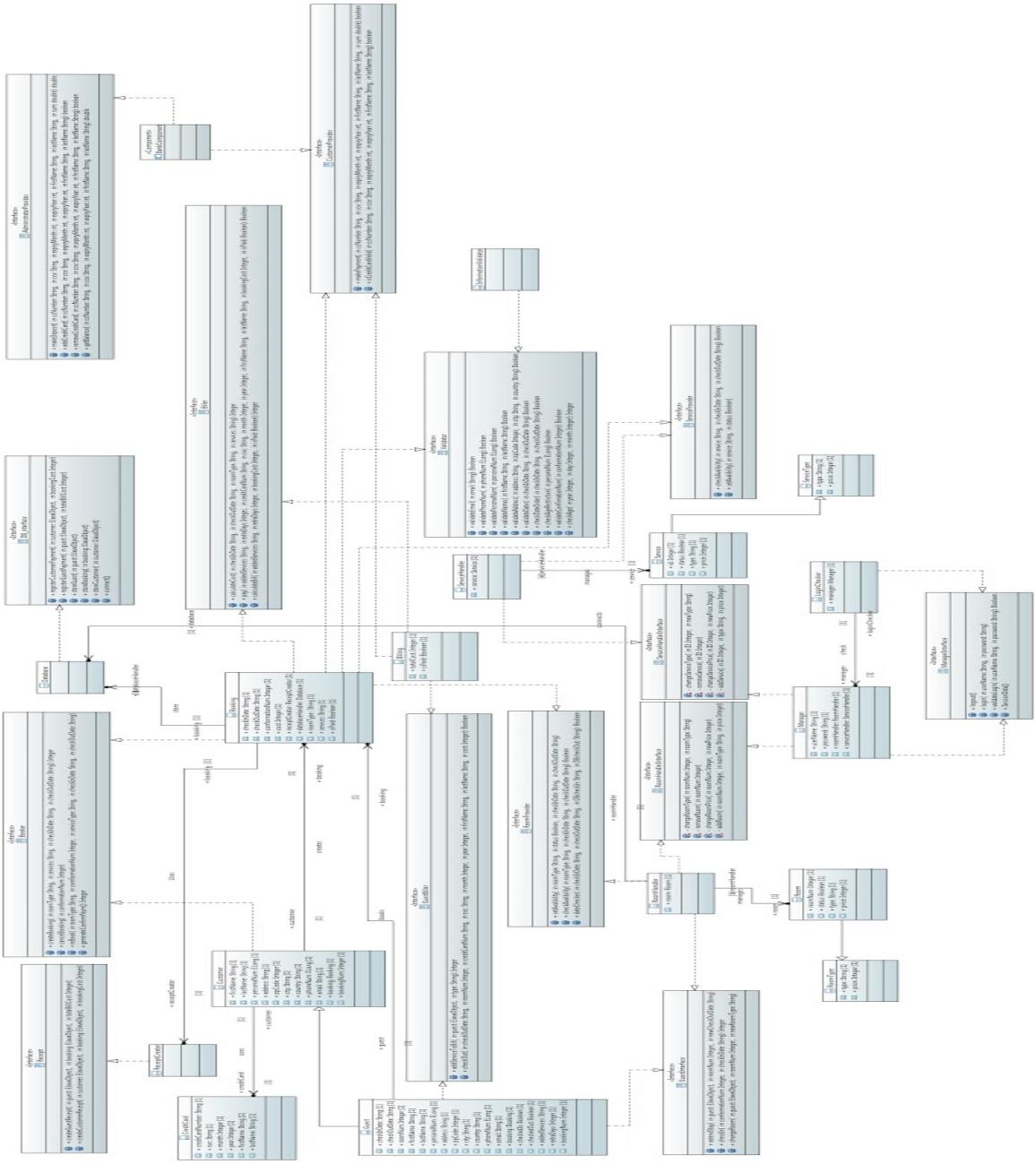
CreateEmpAcc(privilege, accName, password): - create employee account for the system and set the level of privilege.

login(userName, passWord): - this operation checks if username and password match and it creates login session
logout(): - kills current login session

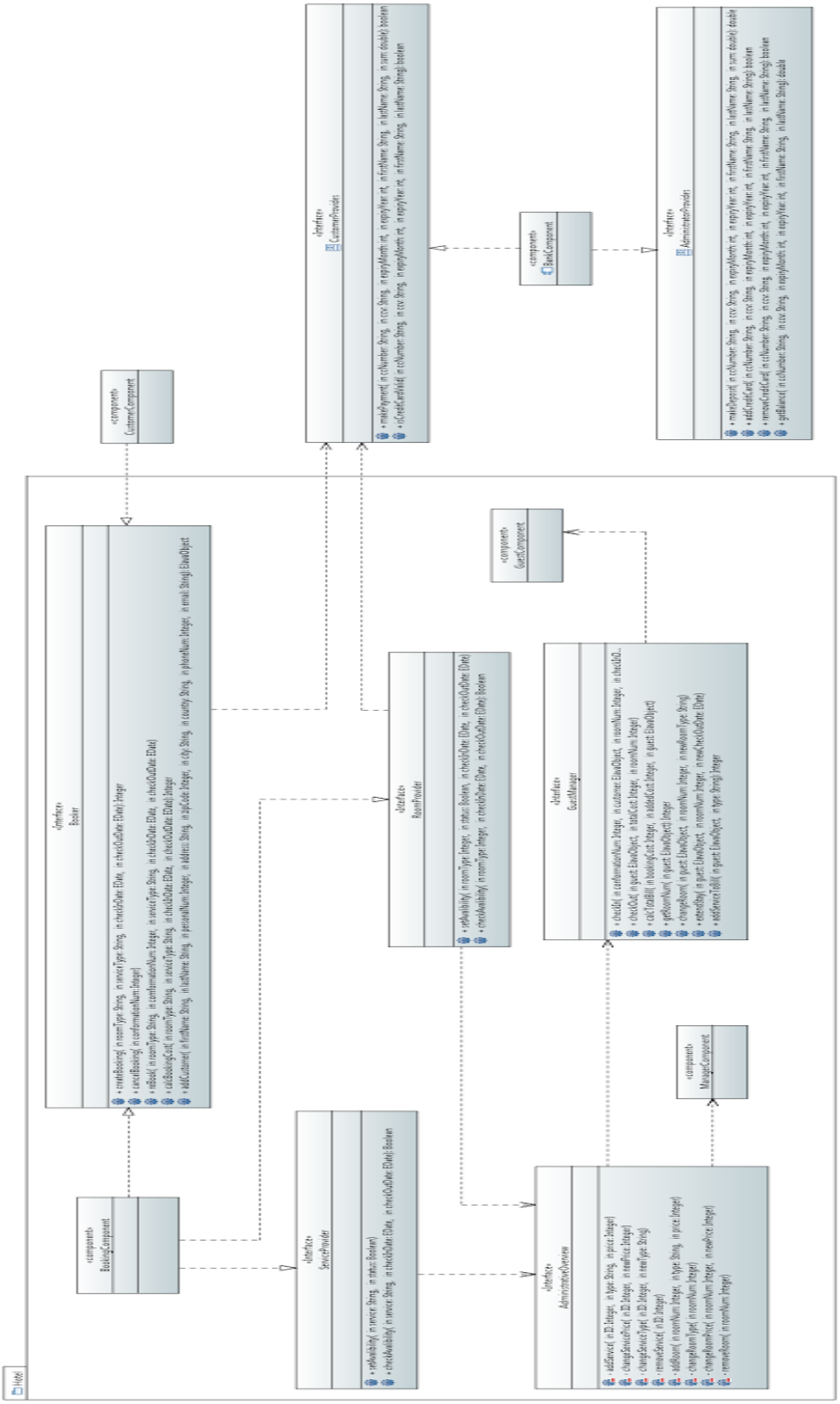
14. Activity diagram



Class Diagram

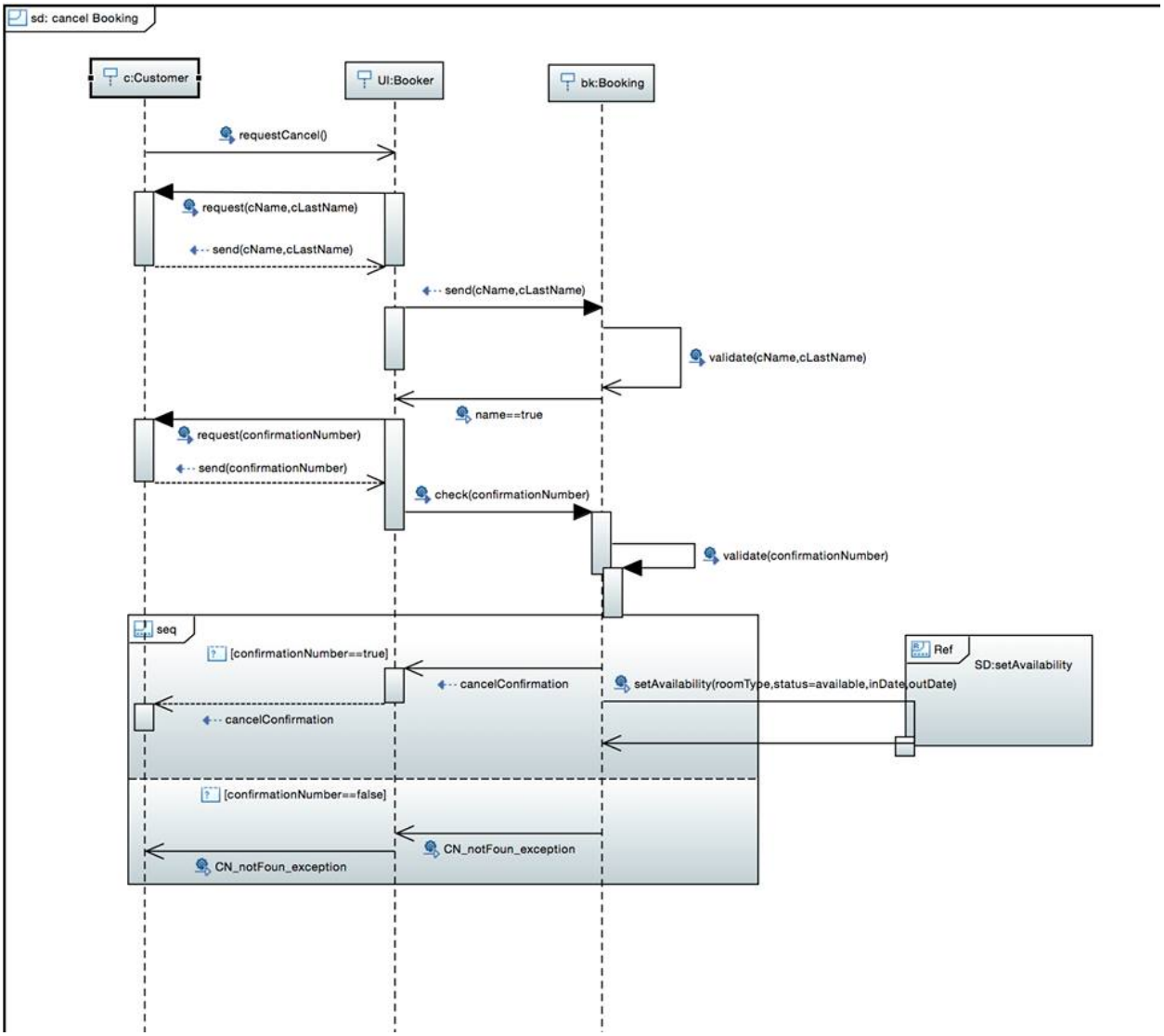


Component Diagram

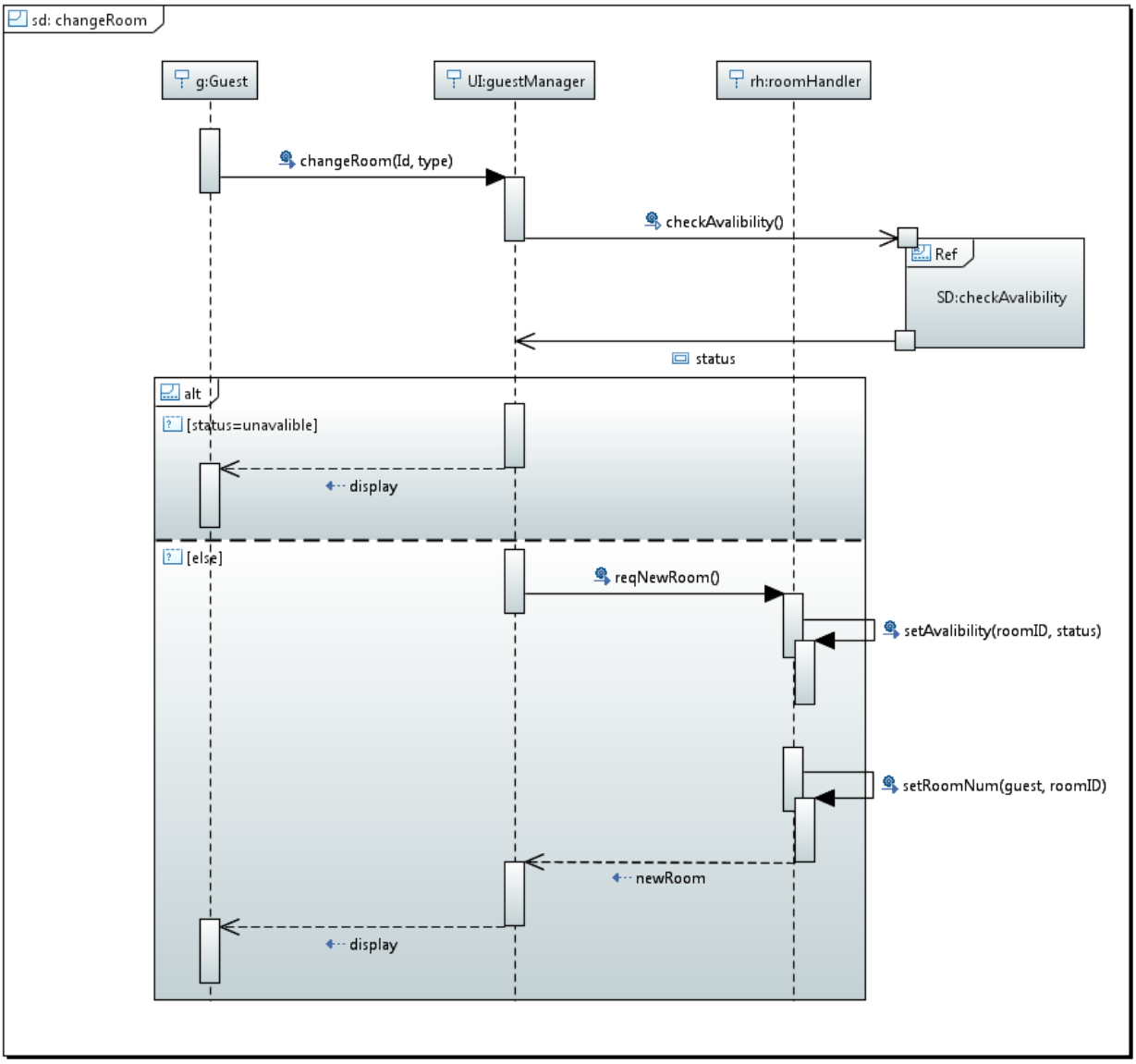


Sequence Diagrams :

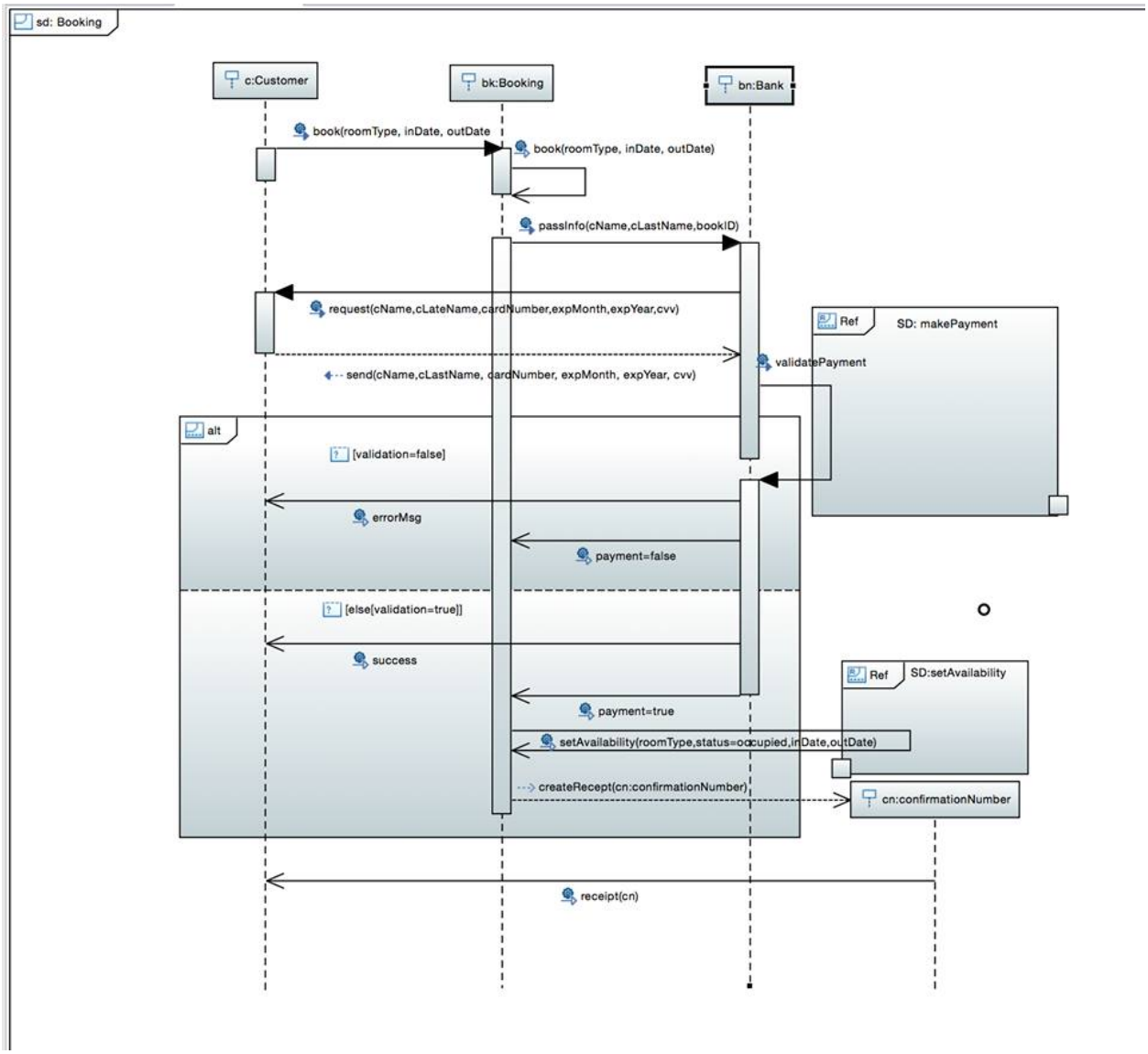
1. **SD:booking**



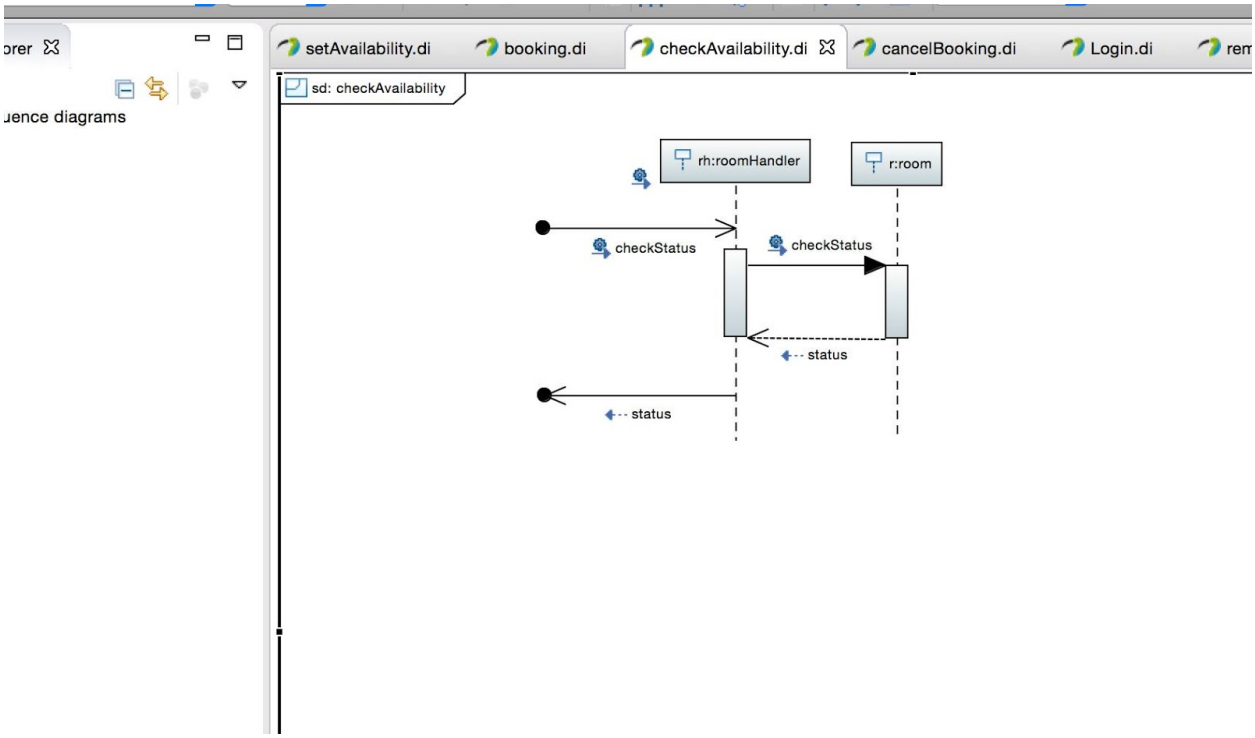
2. SD:cancelBooking



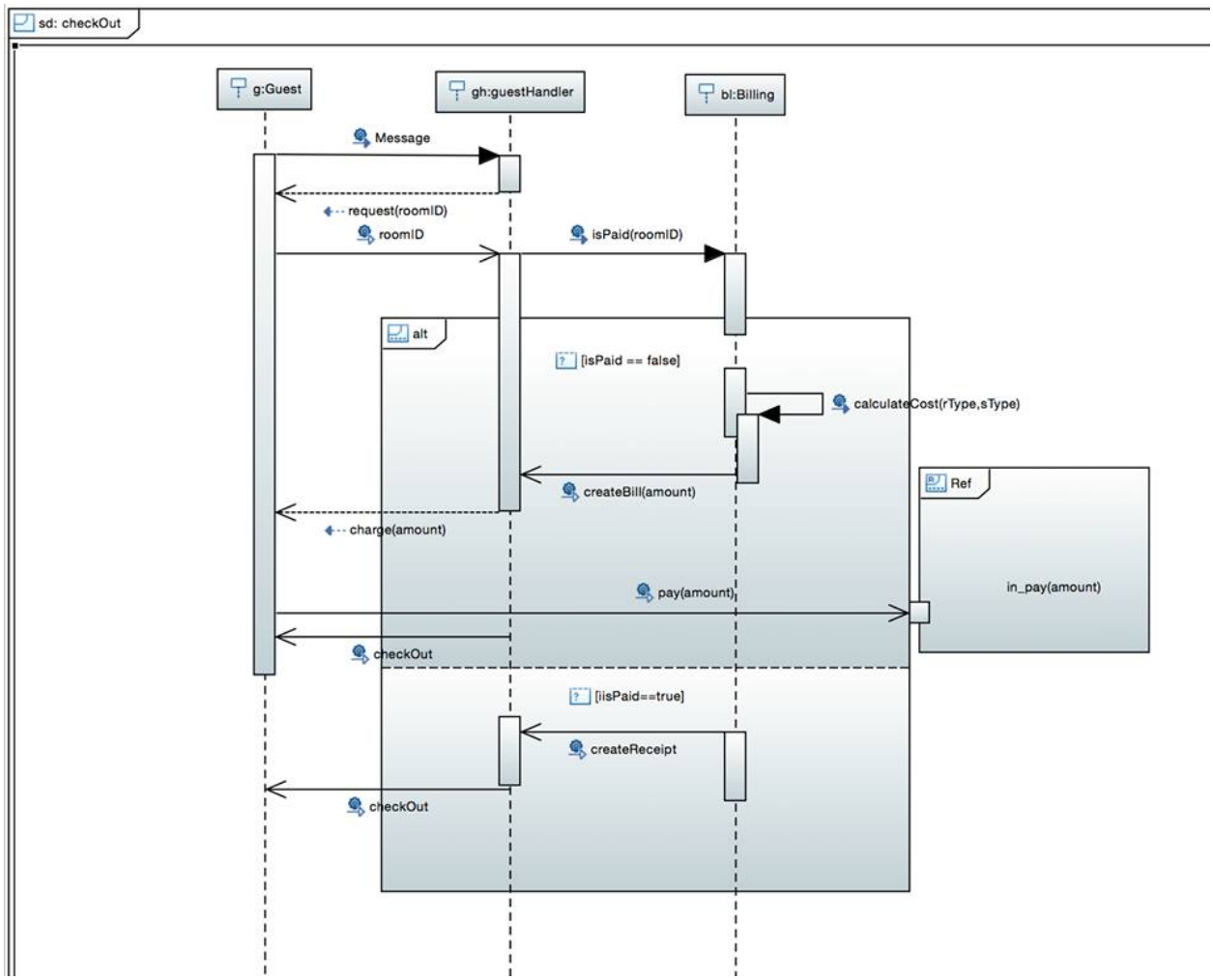
3. SD:changeRoom



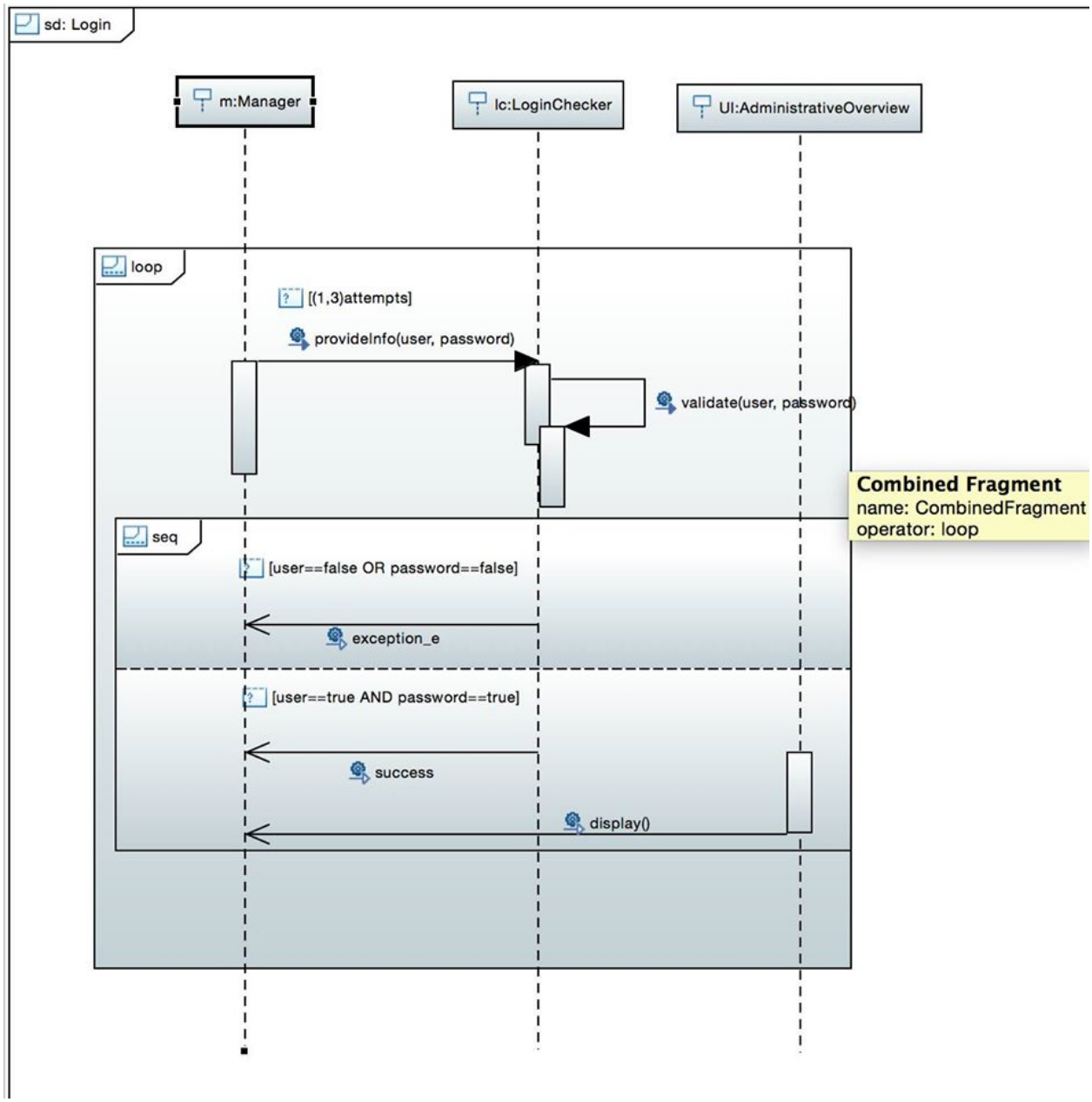
4. SD:checkAvalibility



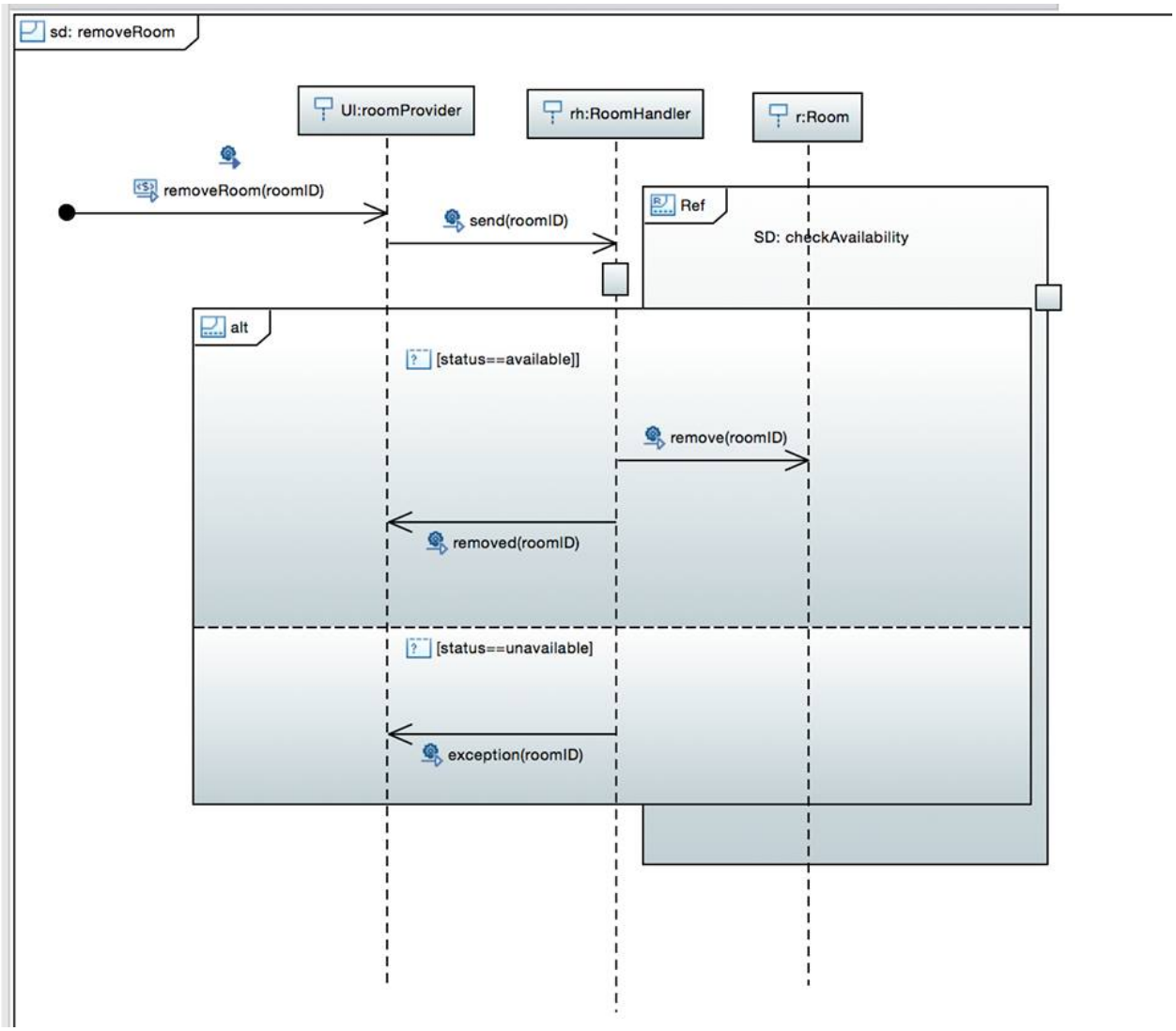
5. SD:checkOut



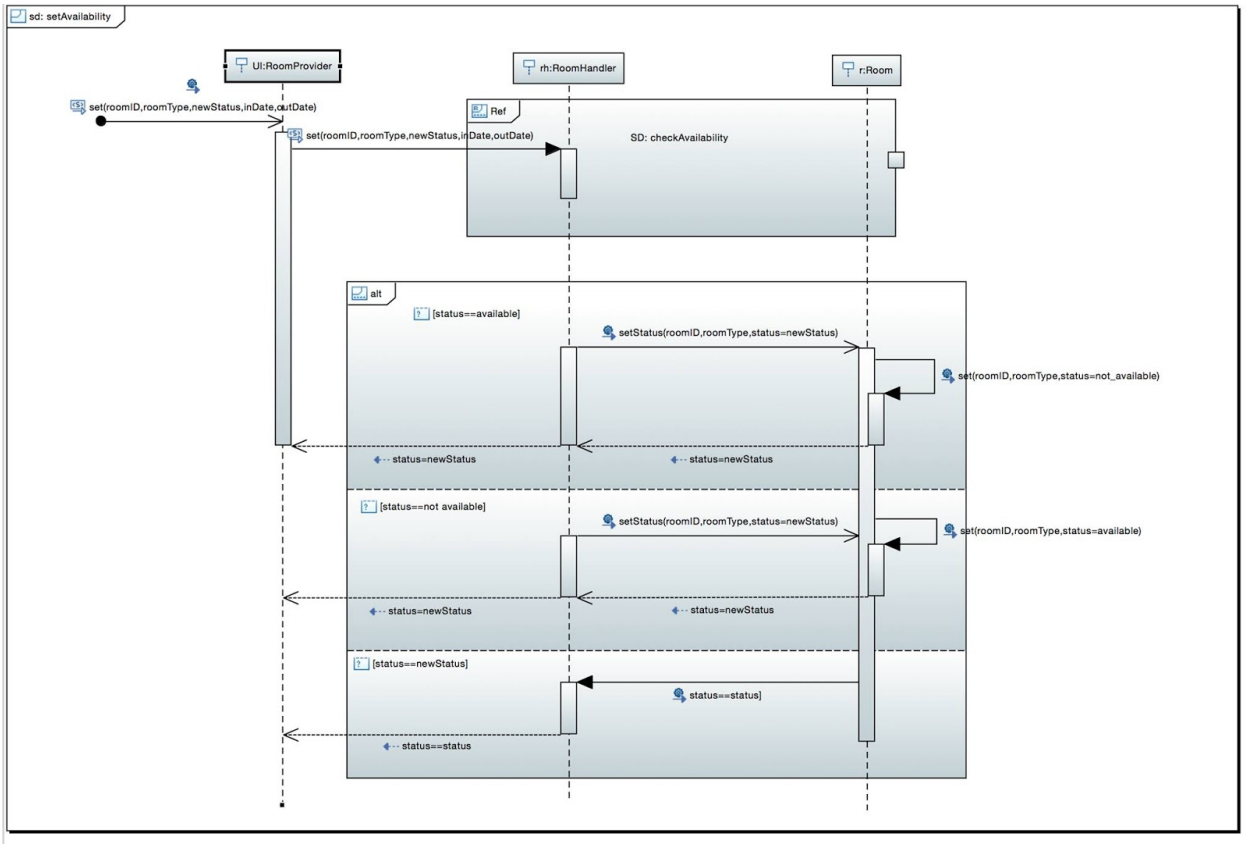
6. SD:login


sd: Login
m:Manager
lc:LoginChecker
UI:AdministrativeOverview
loop
[(1,3)attempts]
provideInfo(user, password)
validate(user, password)
Combined Fragment
name: CombinedFragment
operator: loop
seq
[user==false OR password==false]
exception_e
[user==true AND password==true]
success
display()

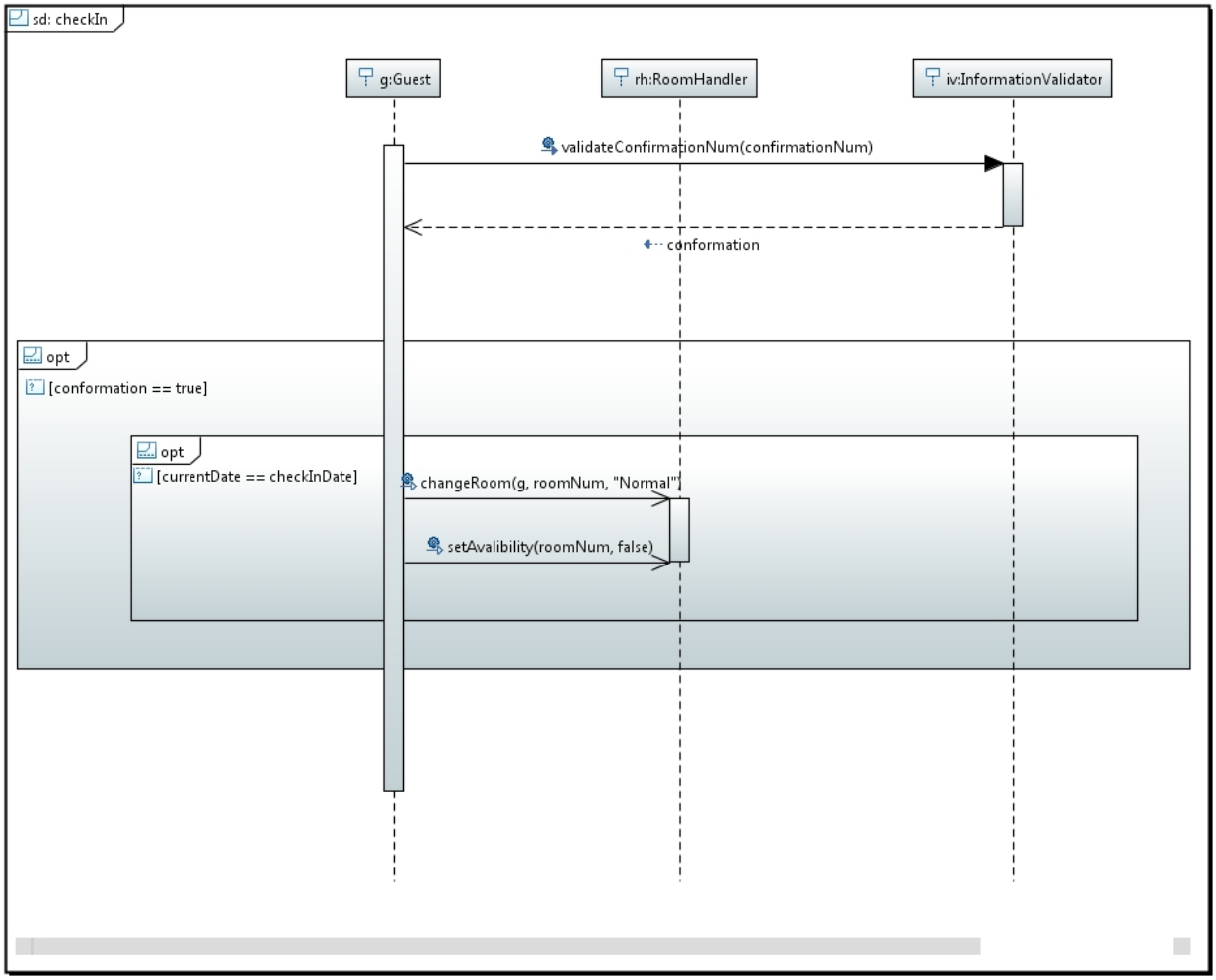
7. SD:removeRoom



8. SD:setAvalibility



9. SD:Check in


sd: checkIn
g:Guest
rh:RoomHandler
iv:InformationValidator
validateConfirmationNum(confirmationNum)
conformation
opt
[conformation == true]
opt
[currentDate == checkInDate]
changeRoom(g, roomNum, "Normal")
setAvalibility(roomNum, false)

15. Division of classes

This section shows the responsibilities of who should use the classes below to implement the code.

Ale

InformationValidator

Andy

Customer
Guest

George

RoomHandler
ServiceHandler
Room
RoomType
Service
ServiceType

Sinan

Booking

Shireen

LoginChecker
Manager

Yazen

Billing
ReceiptCreator

Yacoub

InformationChecker
checkDateOrder()
checkAgeRestriction

Halftime review of (Group16) done by (group17)

2.1 Domain model

1. No concepts need to be removed.
2.
    - ‘CreditCard’ can be an attribute of ‘Customer’. It does not have significant responsibilities to be considered a concept of its own.
    - Same goes for ‘BookingPrice’ and ‘Date’. They are too trivial to be concepts and can be attributes of ‘Booking’.
    - ‘Payment’ could be considered an association between ‘Guest’ and ‘Billing’, and/or ‘Customer’ and ‘Booking’. It feels a bit too weak to be a concept of its own since it depends on those other concepts.
    - We agree that ‘Check-In’ could be considered a concept but since you have ‘Check-out’ as an association class we feel like those to have to be consistent either way you chose. So we would suggest making ‘Check-In’ an association class between ‘BookingRoom’ and ‘Guest’. ‘Guest’ could in order include ‘BookingNumber’ as an attribute.
3. No concepts are missing.
4. Associations names are clear but there are no role names at all. Adding some could clarify some relationships between concepts.
5.
    - According to the model there has to be at least one ‘Guest’ staying in at least one ‘BookingRoom’. For the room it makes sense but there should be able to be zero guests.
    - Also, same goes for ‘Customer’, there can be 0..* customers using the system.
6.
    - The biggest problem here is that ‘Customer’ has no association with ‘Guest’ which doesn’t correspond to reality since ‘Guest’ previously was a ‘Customer’. Your model would be more logical if ‘Guest’ inherits everything that ‘Customer’ contains.
    - Also, it is a bit unclear that ‘BookingRoom’ contains a room number because a ‘Customer’ most likely only books a type of room and ‘Guest’ is the one that has a relationship with a specific room number. It might be a good idea to split ‘BookingRoom’ in to a booking part and add ‘Room’ as a physical part, alternatively changing so that when ‘Customer’ books, it books ‘RoomType’ and only ‘Guest’ interacts with ‘BookingRoom’.
7. The meaning of all names are clear.
8. - Answered in use-case section.
9. The domain model may include some unnecessary concepts but generally, it gives a very good and clear description of the problem domain.

2.2 Vocabulary

The vocabulary is displayed in a good manner.
There are no proper descriptions for some of the element. All of the element have roles, attributes described and some have the association described. The elements are in pair with the domain model and there are none which are not.

2.3 Use-cases

1. All FR are covered by a use-case. If not, is there a description or motivation of which ones were excluded?
No Requirements were provided.

2. Looking at the use cases, what FR should be added?
No Requirements were provided

3. The use-cases have clear goals?
Other than the 5 Complete use-cases there are no use-cases to check their goals.

4.The use-cases have informative names?
Other than the 5 complete use-cases there are no use-cases to check their informative names

5. For each complete use-case, the goal is fulfilled by the main flow?

a. the multiplicity in the domain model suggests that one Customer can make more than one Booking, yet the goal of the Book Room use-case implies that the Customer creates one booking for his staying at the hotel. Also at step 5 it says select rooms meaning he/she can select more.

6. All action steps are written correctly? (e.g. according to the action block approach).

a. action steps are not written correctly in the book room main flow
    i. Step 5-6 can be put together into one step.
    ii. Then there should be a system response like moving to the add information section
    iii. then Step 7 continues
    iv. then Step 8
    v. then validate information (Missing, similar names or long phone number etc)
    vi. then Assume info is valid.
    vii. then Step 9
    viii. Step 11, better use Assume: customer selects pay now and continue with step 12
    ix. remove step 15
    x. Step 17 should be on the Banks system which is out of your system (Different page or such, you got the customer's card number now if you get the password then that's a violation of security sort of)

b. action steps are written correctly in the Check In main flow

c. action steps are not written correctly in the Checkout main flow
    i. usually from room number check out information appears, kinda much to write each time the customer's name, room number, start date and end date.

d. action steps are written correctly in the Add Room main flow

7. All action steps are reasonable for an essential use case
8. What important action steps are missing? covered at 6.
9. What important alternative flows are missing? all of the alternative flows for their use-cases seem covered.
10. The language of the domain model is used in the use-cases? seems so, in the use-cases that are provided.
11. What use-cases do you not consider appropriate? All the provided use-cases seem appropriate.
12. General: No important brief use cases are missing? All of the brief use-cases are missing.

2.4 Business Rules

1. Yes they do, they give specific descriptions of scenarios that are very possible but are hard to include in the domain model.
2. Contexts are given nicely.
3. No they can not. Multiplicities could restrict the maximum number of rooms in a booking to 10, but it would not cover that you can book more over phone or email.

3. Required report contents so far

- Textual description of the system.
  No.
- List of requirements with responsibilities and motivation.
  No.
- Domain model with concepts, attributes, associations, multiplicities and roles.
  yes, but no roles.
- Motivations for the requirements and the domain model requirements (Mind-maps interviews etc).
  No.
- Vocabulary.
  Yes.
- Use-case diagram (use-cases and actors).
  Yes.
- Brief descriptions of all use-cases with priorities.
  No.
- Business rules.
  Yes.

- At least five complete use-cases.
  Yes.
- A list of system operations.
  No.
- An activity diagram and a textual description of the booking use case
  Yes.