Software Design & Analysis Project Document

for

The Best of Hostels

- **Daniyal Imran (20I-0940)**
- **Faizan Pervaz (20I-0565)**

FAST National University of Computer and Emerging Sciences

November 27,2022

Table of Contents

1. Introduction

1.1 Purpose

The java-based application is for a hostel user driven menu. One has to go to a hostel and has to manually get their services whereas, we aim to make it feasible by implementing the whole process into an application where everyone will have access to this application.

1.2 Product Scope

The application's main purpose is to provide users the interface to have their booking in a hostel where different hostels will be available, and User can choose one upon his needs. It can be used a business strategy by any Hostel Owner, who can book his hostel on our application and manage everything from just an application.

1.3 Title

The aim of this project is to provide a hostel management system that will help manage the whole management of the hostel. The immediate solution we created by a java application with a database with user friendly UI.

1.4 Objectives

The application is going to provide a hostel management system, which will be used by the hostels and their employees and users. The system will have a user-friendly interface that allows users to easily check the information such as room bookings, payment collections, transport, and mess, etc.

1.5 Problem Statement

The major reason for choosing the project is that the management system can help to improve the efficiency of the hostels. It will also help in reducing manual work and increasing productivity. The problem addressed by this project is to automate the office work of a hostel so that it can be done electronically instead of manually.

This will help in reducing manual work, which is one of the main reasons for failure of most of the businesses. The operation cost would also be reduced due to automation of any business process.

Another major reason for choosing the project is that it will help us to reduce our costs and time. The problem addressed by this project is, in general, that we have a lot of problems with office automation which has led to a lot of manual work. It will benefit the students who need to stay in hostels, and they will no longer have to carry around their own dossiers. The database system will also help them with their other tasks such as storing important information and making payments, etc.

2. Overall Description

2.1 Product Perspective

This project is a follow-on member of a product family. This product has been developed as an upgrade to existing systems, and it has been designed with the same features and capabilities as its predecessor with some additional features. The software design and analysis describe a component of a larger system, the hostel management system.

The hostel management system includes an online booking module, an online payment module, and an online reservation module. Each of these modules requires user input from each other, so it is important to define how they interact with one another as well as with external systems (e.g., Google Maps).

Each module has its own interface with the database that stores user data and settings. However, there are interfaces between each module and other components within the larger hostel management system: for example, there are interfaces between the booking module and the reservation module so that users can make requests for accommodations during their stay at hostel.

2.2 Product Functions

The Hostel management system is a software that can help hostelers to manage their hostels, especially the standard services such as booking and reservation, accounting, marketing, and other related operations. It can also help them connect with the customers and business partners. It has a database that stores all the information like rooms, rates, rates per room, etc. Different functions which a user can perform,

1. Managing Rooms
2. Managing Mess
3. Booking Transport
4. User can order Cleaning Services
5. User can order Laundry Services
6. User can get his own Internet Services
7. User can give payment through the application

2.3 List of Use Cases

- Manage Users
- Internet Services
- Laundry Services
- Payment
- Cleaning Services
- Manage hostels
- Book room
- Book mess
- Book transport

2.4 Extended Use Cases

Use Case Section:	1
Use Case Name:	User Registration or Login (Faizan Pervaz – 20I0565)
Scope:	Hostel Management System
Level:	User Goal
Primary Actor	User
Stakeholders and Interests:	User, Hostel Owner, HMP Owner
Main Success Scenario:	

Actor Actions	System Response
The User opens the application and presents the interface.	
	The system waits for input
The User clicks option off logging in or signup Accordingly	
	If systems get log in option, it gets input of id and password from user and checks it from database, is it correct and allows user to log in otherwise system registers the user as a new user and asks to input name, password, email, age.
User enters the Name, Password, email, age in case of registration and Password and Id in case of log in.	
	System saves the information provided by user in the database and generates a unique id number to the user
User gets the Unique Id on screen.	

Extensions:	System Detects an error User is not logging in due to database issue

Pre-conditions:	User must be in the Application
Post conditions:	User will be registered or logged in

Use Case Section:	2
Use Case Name:	Manage Users (Daniyal Imran – 20I0940)
Scope:	Hostel Management System
Level:	System Goal
Primary Actor	Admin
Stakeholders and Interests:	User, Hostel Owner, HMP Owner
Main Success Scenario:	

Actor Actions	System Response
Admin clicks on option of manage users from the interface.	
	System checks the admin from it's database And allows admin to manage users
Admin can view the users (customers) from The database and change their information as Well as delete user from it.	
	System will make changes and save to it's database accordingly

Extensions:	System Detects an Error Admin is not in database
Pre-conditions:	Application Must be Open and Admin should be there

Post conditions:	Admin can manage users

Use Case Section:	3
Use Case Name:	Manage Hostels (Faizan Pervaz – 20I0565)
Scope:	Hostel Management System
Level:	User Goal
Primary Actor	Admin
Stakeholders and Interests:	User, Hostel Owner, HMP Owner
Main Success Scenario:	

Actor Actions	System Response
Admin clicks on the option of manage hostels in the application interface	
	System checks the admin from it's database and allows admin to manage hostels
Admin now has the option to delete, change and update the hostels. Admin has hostel's information and hostel owners information	
	System will get the updated the hostel and update it in the database

Extensions:	System Detects an Error Admin is not in database
Pre-conditions:	Application Must be Open and Admin should be there

Post conditions:	Admin can manage users

Use Case Section:	4
Use Case Name:	Book Room (Faizan Pervaz – 20I0565)
Scope:	Hostel Management System
Level:	User Goal
Primary Actor	User
Stakeholders and Interests:	User, Hostel Owner, HMP Owner.

Main Success Scenario:

Actor Actions	System Response
The user opens the application. The home screen appears to the user.	
	The system opens its interface to the user and waits for input.
User searches for hostels by specifying city and area.	
	The application gives options for the user to select from one of them. Users can either book themselves in a specific hostel's transport.
Users select the option to book the hostel rooms.	
	The specific hostel room will be booked and seat/bed number will be assigned to the user and user details will be stored in the database against the hostel room number.

1. Extensions:

1. Updating error
1. Error in taking possession of the room

2.

➢ Pre-conditions:	User must be login.
Post conditions:	The user has booked the room

<table>
<tr><td>Use Case Section:</td><td>5</td></tr>
<tr><td>➢ Use Case Name:</td><td>Book Mess (Faizan Pervaz – 20I0565)</td></tr>
<tr><td>Scope:</td><td>Hostel Management System</td></tr>
<tr><td>➢ Level:</td><td>User Goal</td></tr>
<tr><td>➢ Primary Actor</td><td>User</td></tr>
<tr><td>Stakeholders and Interests:</td><td>User, Hostel Owner, HMP Owner.</td></tr>
<tr><td>Main Success Scenario:</td><td>
<table>
<tr><th colspan="2">Actor Actions</th><th colspan="2">System Response</th></tr>
<tr><td colspan="2">User opens the application.
The home screen appears to the user.</td><td></td><td></td></tr>
<tr><td colspan="2"></td><td></td><td>The system opens its interface to the user and waits for input.</td></tr>
<tr><td colspan="2">User searches for hostel’s mess by specifying city and area.</td><td></td><td></td></tr>
<tr><td colspan="2"></td><td></td><td>The application gives options for the user to select from one of them. Users can either book themselves in a specific hostel’s room or mess or transport.
.</td></tr>
<tr><td colspan="2">The user selects the option to book the hostel Mess.</td><td></td><td>The specific hostel Mess will be booked and the Id number will be assigned to the user and user details will be stored in the database against the hostel Mess</td></tr>
</table>
</td></tr>
<tr><td>Extensions:</td><td>Swapping Error
System Detects Error</td></tr>
</table>

Pre-conditions:	User must be login.
Post conditions:	The user has booked the Mess

Use Case Section:	6
➢ Use Case Name:	Book Transport (Faizan Pervaz – 20I0565)
➢ Scope:	Hostel Management System
Level:	User Goal
Primary Actor	User
Stakeholders and Interests:	User, Hostel Transport Owner, HMP Owner.

Main Success Scenario:

Actor Actions	System Response
The user opens the application. The home screen appears to the user.	
	The system opens its interface to the user and waits for input.
User searches for hostel's transports by specifying city and area.	
	The application gives options for the user to select from one of them. Users can either book themselves in a specific hostel's room or mess or transport.
User selects the option to book the hostel transport facility.	
	The specific hostel transport facility will be booked and the Id number will be assigned to the user and user details will be stored in the database against the hostel Transport.

Extensions:	Updating error System Detects error

➢ Pre-conditions:	User must be login.
➢ Post conditions:	The user has booked the Transport.

Use Case Section:	7
Use Case Name:	Payment (Daniyal Imran – 20I0940)
➢ Scope:	Hostel Management System
Level:	User Goal
Primary Actor	User
Stakeholders and Interests:	User, Hostel Owner, HMP Owner.
Main Success Scenario:	

Actor Actions	System Response
	In the month's end application generates the total bill and sends a notification to the user along with a bill to pay the bill.
The user wants to pay the bill and then he clicks the option of payment	
	The application opens too many options of payment for the user. In application, you can pay from a bank account, cash, easy paisa, or Jazz Cash.
The user selects the cash payment option.	
	Application will send the details of the hostel owner whom he must give payment by cash.
User will give the cash to the hostel owner.	
	After payment confirmation user payment status will be updated.
User has successfully paid.	

Extensions:	Updating error

Pre-conditions:	User must be login
Post conditions:	The user has paid successfully

Use Case Section:	8
Use Case Name:	Social Discounts (Faizan Pervaz – 20I0565)
Scope:	Hostel Management System
Level:	User Goal
Primary Actor	User
Stakeholders and Interests:	User, Hostel Owner, HMP Owner.

Main Success Scenario:

Actor Actions	System Response
The user opens the application. The home screen appears to the user.	
	System opens its interface to the user and waits for input
User wants to get a social discount. Users will click on the social discount option.	
	The application will open many options to the user.
Suppose the user clicks on the community service option	
	The application will show the details of all the community service organizations
Users will choose one of the options of a community service organization and book themselves in that community service for the weekend.	
	The application will check for that organization's seat availability. After confirmation notification will be sent to the user that their seat is booked.

Extensions:	Confirmation error System detects error

Pre-conditions:	User must be login
Post conditions:	The user has received the discount.

<table>
<tr><td>Use Case Section:</td><td>9</td></tr>
<tr><td>➢ Use Case Name:</td><td>Internet Services (Daniyal Imran – 20I0940)</td></tr>
<tr><td>➢ Scope:</td><td>Hostel Management System</td></tr>
<tr><td>➢ Level:</td><td>User Goal</td></tr>
<tr><td>Primary Actor</td><td>User</td></tr>
<tr><td>Stakeholders and Interests:</td><td>User, Hostel Owner, HMP Owner</td></tr>
<tr><td>Main Success Scenario:</td><td>
<table>
<tr><th colspan="2">Actor Actions</th><th colspan="2">System Response</th></tr>
<tr><td colspan="2">The user opens the application. The home screen appears to the user.</td><td colspan="2"></td></tr>
<tr><td colspan="2"></td><td colspan="2">The system opens its interface to the user and waits for input.</td></tr>
<tr><td colspan="2">The user wants to get internet service. The hostel is providing WIFI. but its signal strength is not good. The user will click on the internet service options.</td><td colspan="2"></td></tr>
<tr><td colspan="2"></td><td colspan="2">The application will open many options to the user. there will be plenty of options for internet service providers (ISP) like PTCL, Nayatel, and Transworld etc.</td></tr>
<tr><td colspan="2">The user clicks on one of the options.</td><td colspan="2"></td></tr>
<tr><td colspan="2"></td><td colspan="2">Application will show the details of all its subscription plans. The user will select one of them.</td></tr>
<tr><td colspan="2">The application will book their subscription plan and ISP workers will come and install the modem in the user’s room. The user won’t pay the modem installation fee (The application will handle it).</td><td colspan="2">User will get the internet service they desire.</td></tr>
</table>
</td></tr>
<tr><td>➢ Extensions:
➢</td><td>Limited Options
System detects an error</td></tr>
</table>

Pre-conditions:	User must be login
Post conditions:	The user has acquired internet.

Use Case Section:	10
Use Case Name:	Cleaning Service (Daniyal Imran – 20I0940)
Scope:	Hostel Management System
Level:	User Goal
Primary Actor	user
Stakeholders and Interests:	User, Hostel Owner, HMP Owner
Main Success Scenario:	

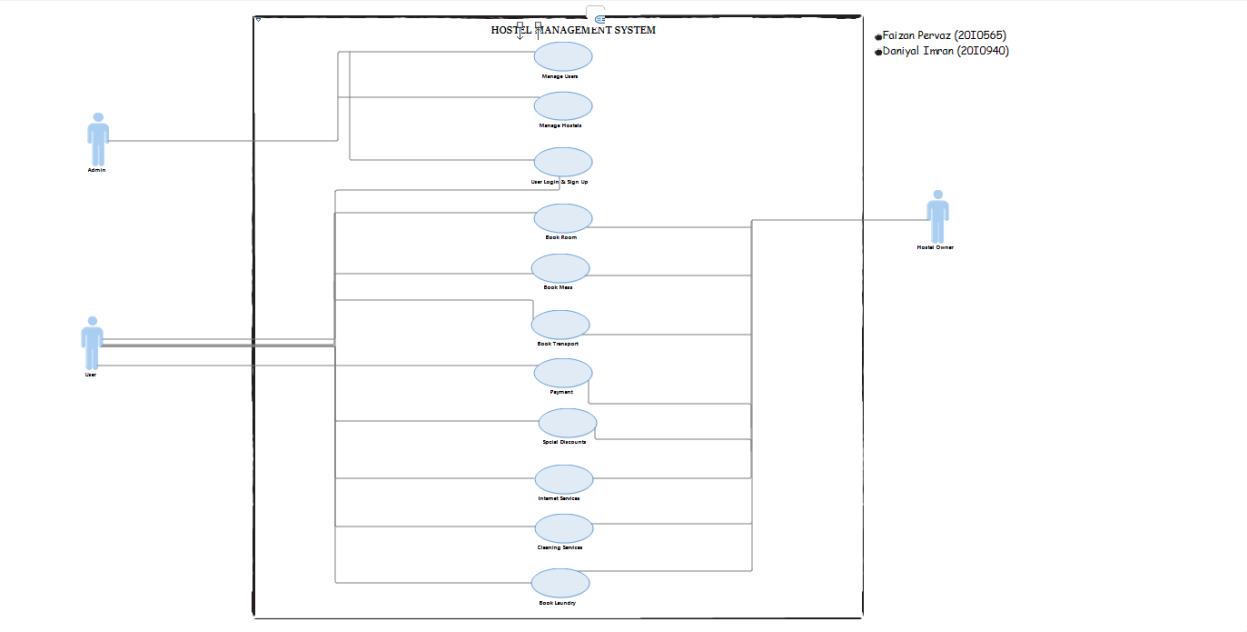
Actor Actions	System Response
The user opens the application. The home screen appears to the user.	
	The system opens its interface to the user and waits for input
The user wants to get a cleaning service. The hostel is providing a cleaning service. The user will click on the cleaning service option.	
	The application will get a cleaning request from the user and send a notification to the hostel owner about the cleaning service along with the user and its room details.

Extensions:	Availability issues System Detects Error
Pre-conditions:	User must be login
Post conditions:	The user’s room has been cleaned.

<table>
<tr><td>Use Case Section:</td><td>11</td></tr>
<tr><td>Use Case Name:</td><td>Book Laundry (Daniyal Imran – 20I0940)</td></tr>
<tr><td>Scope:</td><td>Hostel Management System</td></tr>
<tr><td>Level:</td><td>User Goal</td></tr>
<tr><td>Primary Actor</td><td>User</td></tr>
<tr><td>Stakeholders and Interests:</td><td>User, Hostel Owner, HMP Owner</td></tr>
<tr><td>Main Success Scenario:</td><td>
<table>
<tr><th>Actor
Actions</th><th>System Response</th></tr>
<tr><td>The user opens the application. The home screen appears to the user.</td><td></td></tr>
<tr><td></td><td>The system opens its interface to the user and waits for input</td></tr>
<tr><td>The user wants to get his laundry done. The hostel is providing a laundry service.
The user will click on the Do Laundry Service option.</td><td></td></tr>
<tr><td></td><td>The application will get a Laundry request from the user and send a notification to the hostel owner about the Laundry service along with the user and its room details.</td></tr>
<tr><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>Extensions:</td><td>Laundry Not Available

System Detects an Error</td></tr>
<tr><td>Pre-conditions:</td><td>User must be login</td></tr>
<tr><td>Post conditions:</td><td>User has acquired laundry</td></tr>
</table>

2.5 Use Case Diagram



3. Other Nonfunctional Requirements

3.1 Performance Requirements

The application will be able to handle a large number of users. It will also be able to handle the high volume of user requests that take place on the system like get transport, mess, room, internet services, etc. The application should be able to use a database for storing data about user's reservations, etc.

Performance requirements for this project are as follows:

The system should be able to handle an unlimited user at any given time

The system should have a minimum response time of 0.5 seconds from a user's request to the action being taken

3.2 Safety Requirements

Safeguards or actions that must be taken:

Users can resolve their queries by contacting the support staff on the given email.

The product should have an easy-to-use interface, which allows users to navigate through the features in an intuitive way, especially when they need help or support from other people who are using the system. The interface should not require too much time or effort from users when they want access to information they need at any given moment of time; otherwise, this will affect their productivity negatively

3.3 Security Requirements

Hostel Management System is a java-based application developed in Java and based on MySQL database. It has a set of access control rules and procedures that help to manage the hostels. The security requirements are as follows:

1. The application should have a mechanism to authenticate the users by using username and password to access the application.

2. The application should have a mechanism to ensure that no unauthorized person can modify any of the data in the database, even if they are able to get access to it by using an account with limited privileges or an unauthorized person gets access to it through hacking or other attacks on an open network, or through some other means not covered by this product's security policy.

3.4 Software Quality Attributes

After careful consideration, we have identified the following quality attributes that are important to either the customers or the developers. We will be focusing on adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability.

Adaptability: A product that is adaptable allows its users to easily modify it as they see fit. To be useful in a specific setting or situation with a particular data set requires an adaptable software product. Only the Admin can modify all the user functionalities. The Users cannot modify another user's information.

Availability: It is important for a software product to be available since it will facilitate the users' ability to access it. Availability should not be confused with portability since both refer to how easy it is for someone else to use your product in their context. It is a java-based application on my own computer hence, it will not be portable right now, but we aim to make it web based.

Correctness: The most important aspect of any software product is its correctness since it determines whether it will work properly without any errors being introduced into the system. Therefore, correctness is considered an important quality attribute for a software product because if there are errors present then there will be problems in using this product which could lead to major issues later down the road when attempting to use this product again or rely on it in some way or another. Not everything is perfect, but we have made countless pieces of information to do it.

3.5 Business Rules

The application is a Java-based application which will be run on a server. It will be used for the management of hostels, which are places where students can stay during their stay in college. The application will have a database for storing all the information about the hostels and their users.

The business rules are as follows:

-Only Users can book rooms.

-Only students can check out rooms from the database.

-Only the Admin has access to everything

3.6 Operating Environment

The Hostel Management System is developed in Java. The Hostel Management System will run on Windows Operating System version 10,11, which is installed on a PC with a Pentium 4 processor and 2 gigabytes of RAM. Hostel Management System will use SQL Database, which can be downloaded from the SQL official website. The Hostel Management System has been tested on Windows 11, but it should work fine on Windows 10 as well.

3.7 User Interfaces

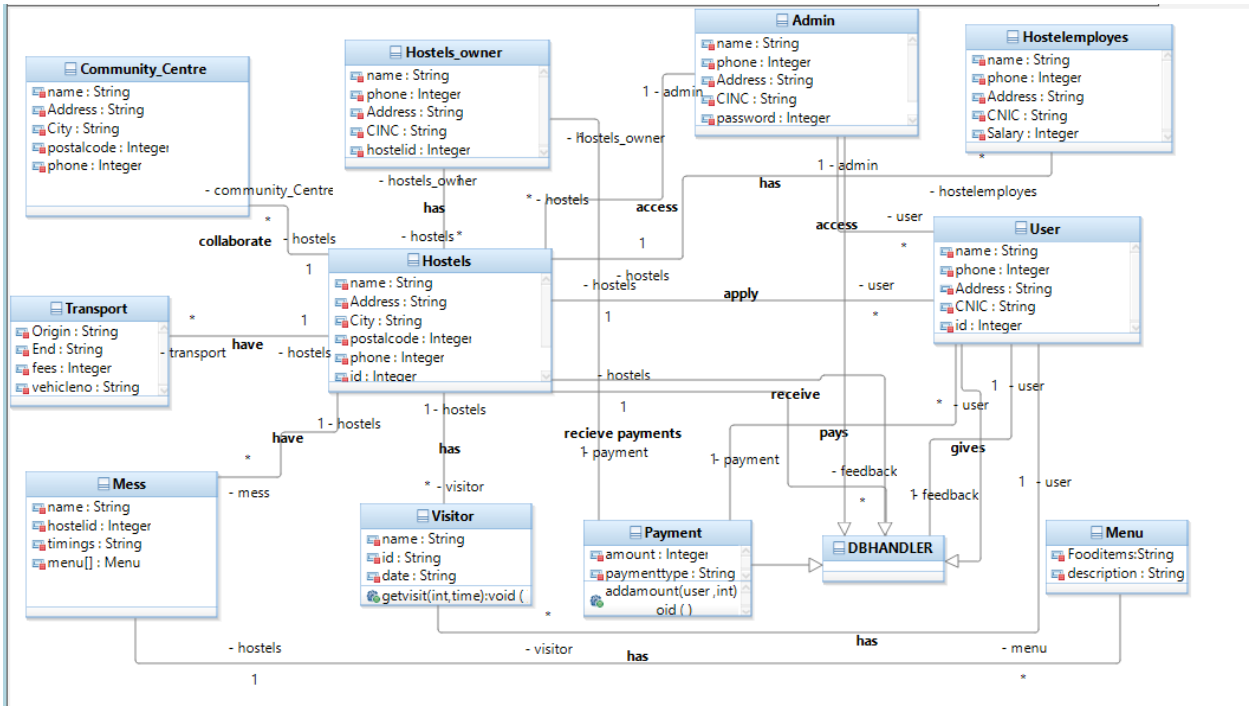
The application has a Java based user interface with a database that is used to manage the hostel and bookings. The user interface can be accessed through this java application.

Each screen has multiple dropdowns, buttons and other functions that are needed for the users to use the application according to their needs. These functions are located at different places on each screen, so it becomes easier for the user to perform their tasks.

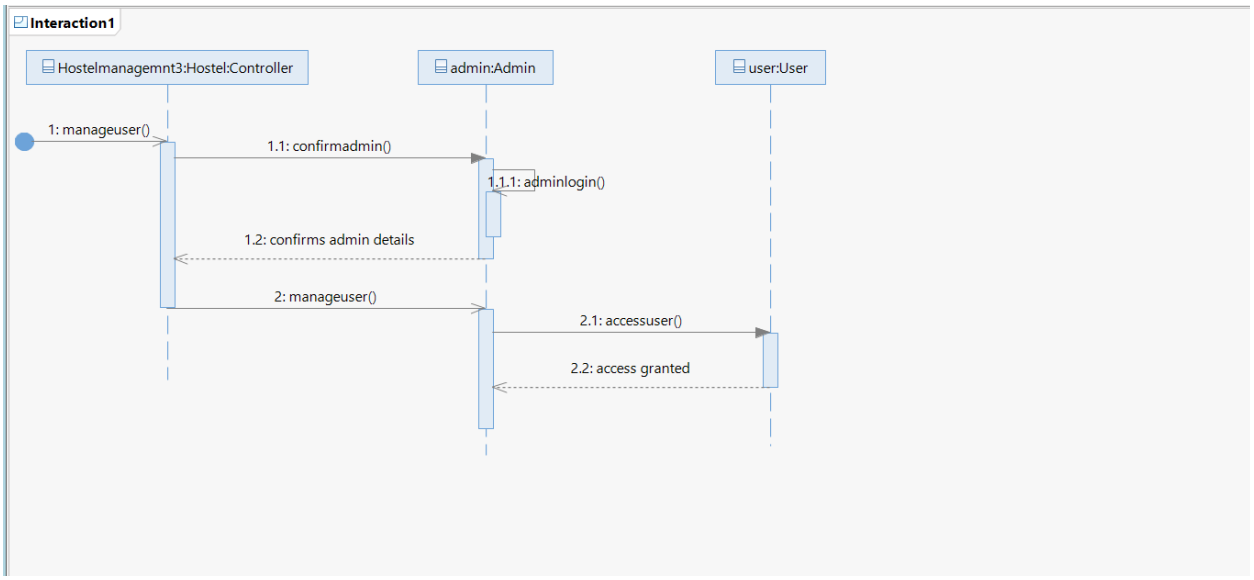
All screens have standard buttons which are used for various purposes.

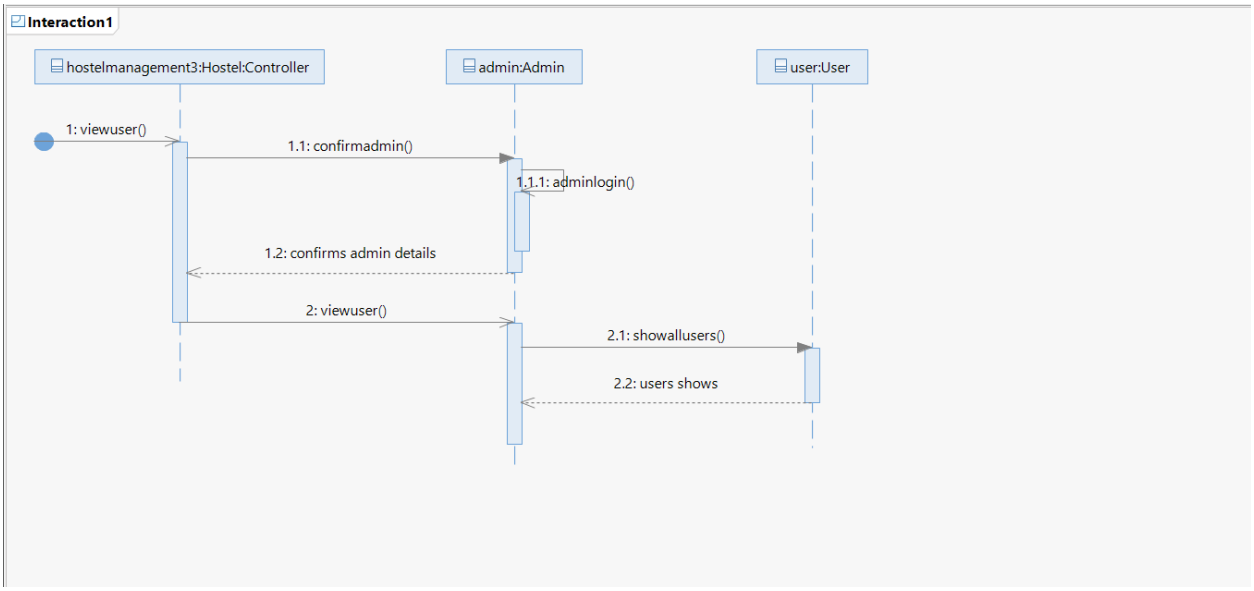
Like in Admin, User can use the buttons to access, delete or update the user. In the User Roles, logged in as User, buttons are there to get transport, mess, room, cleaning services and laundry, etc. In the Payment option dropdown button is there to select which bank. And many other functionalities are implemented.

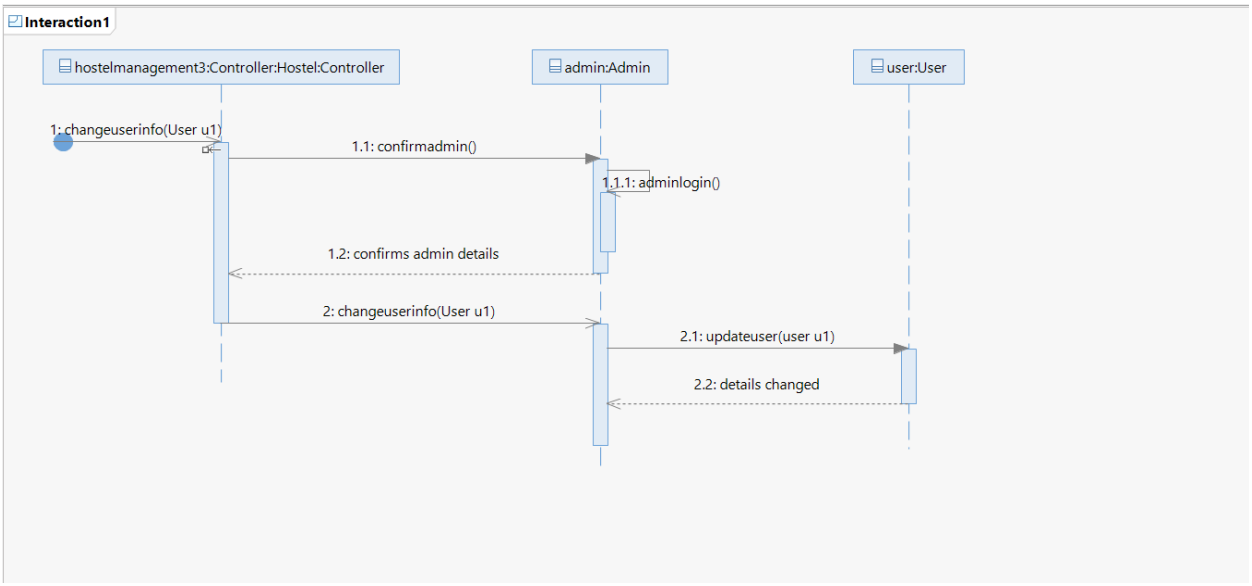
4. Domain Model

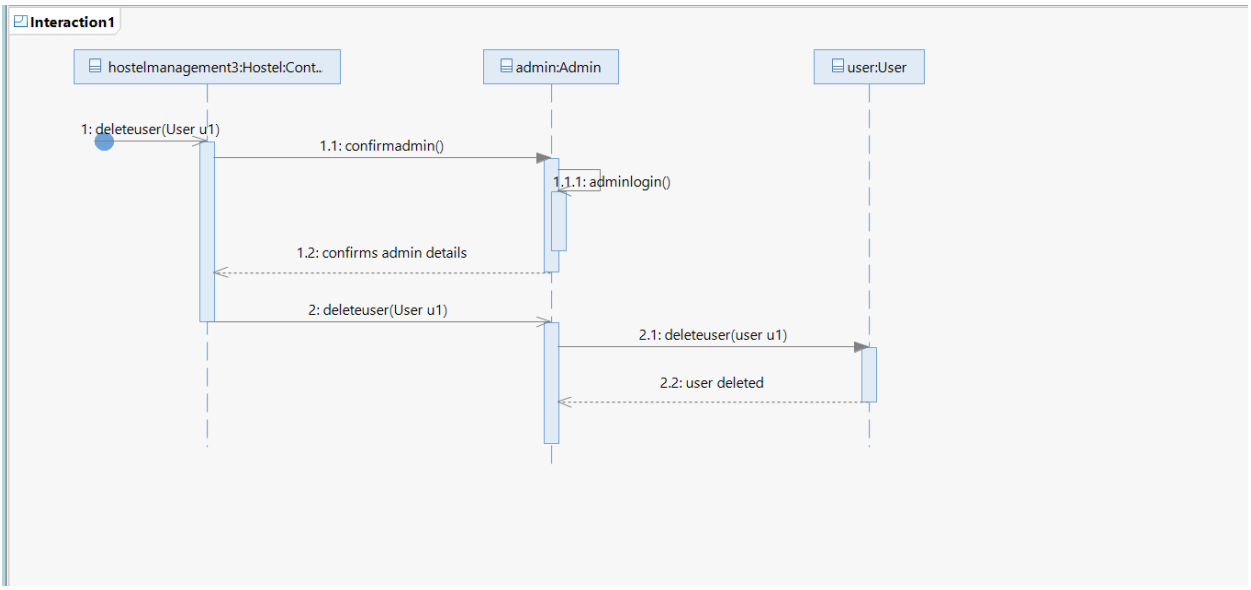


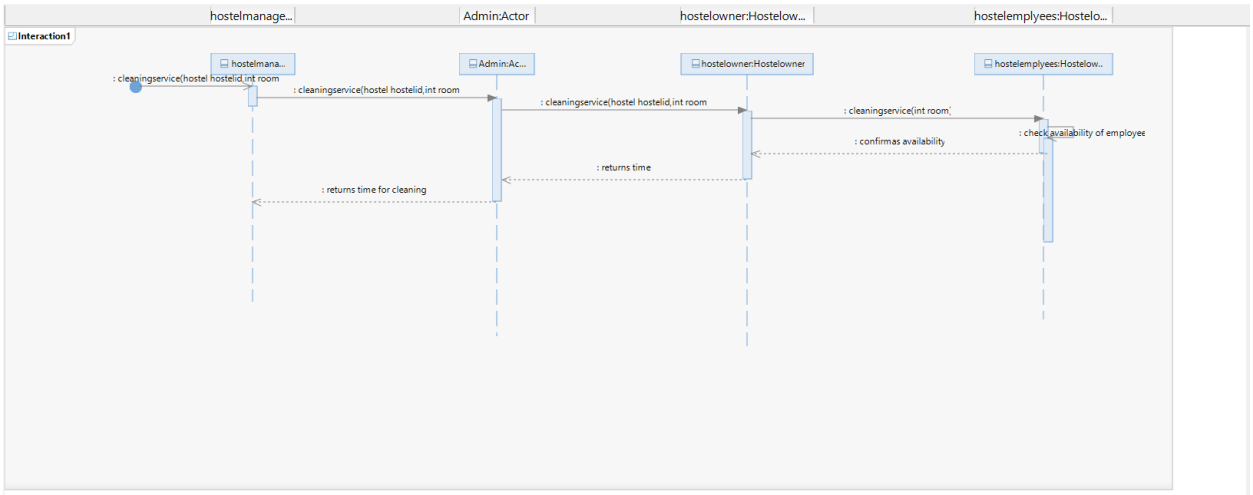
5. Sequence Diagram

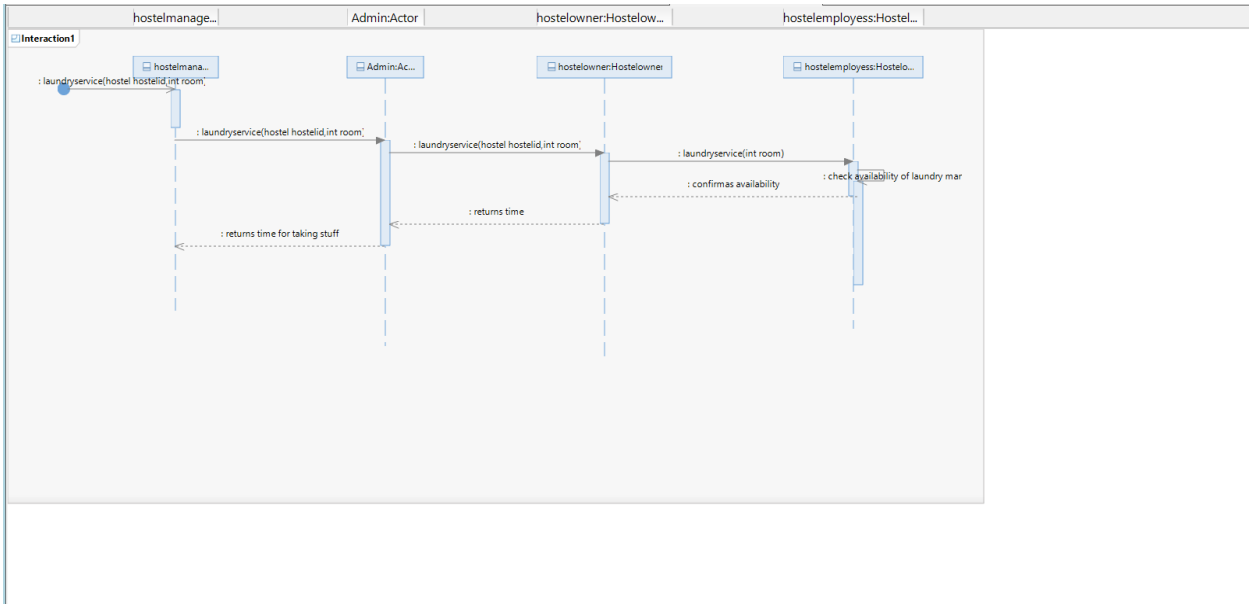


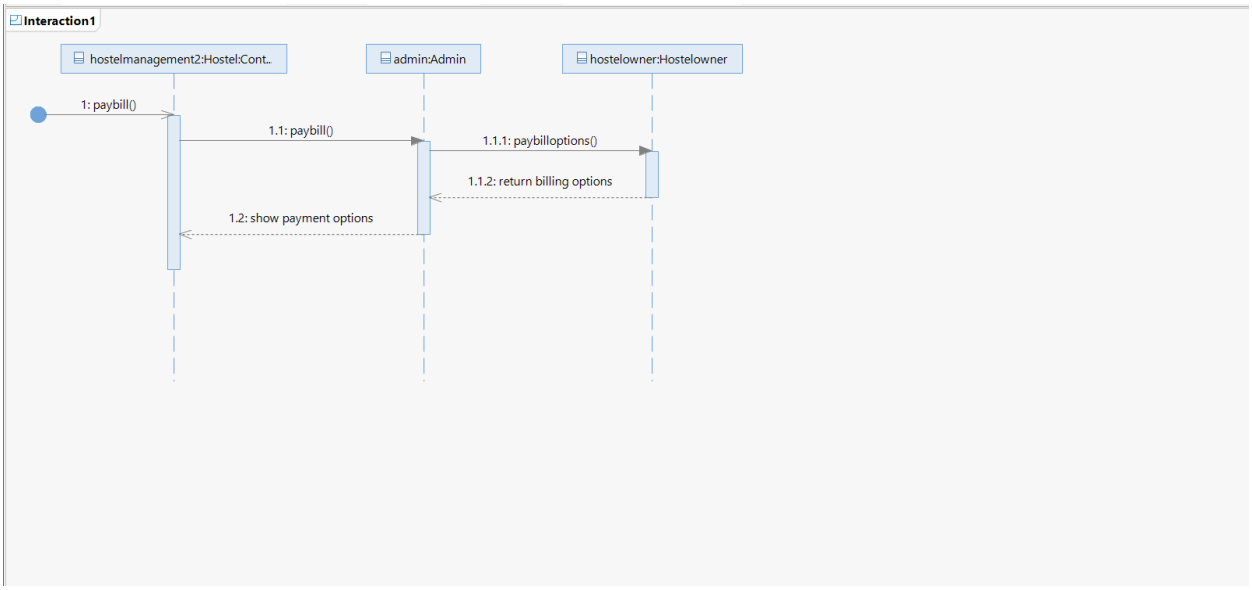


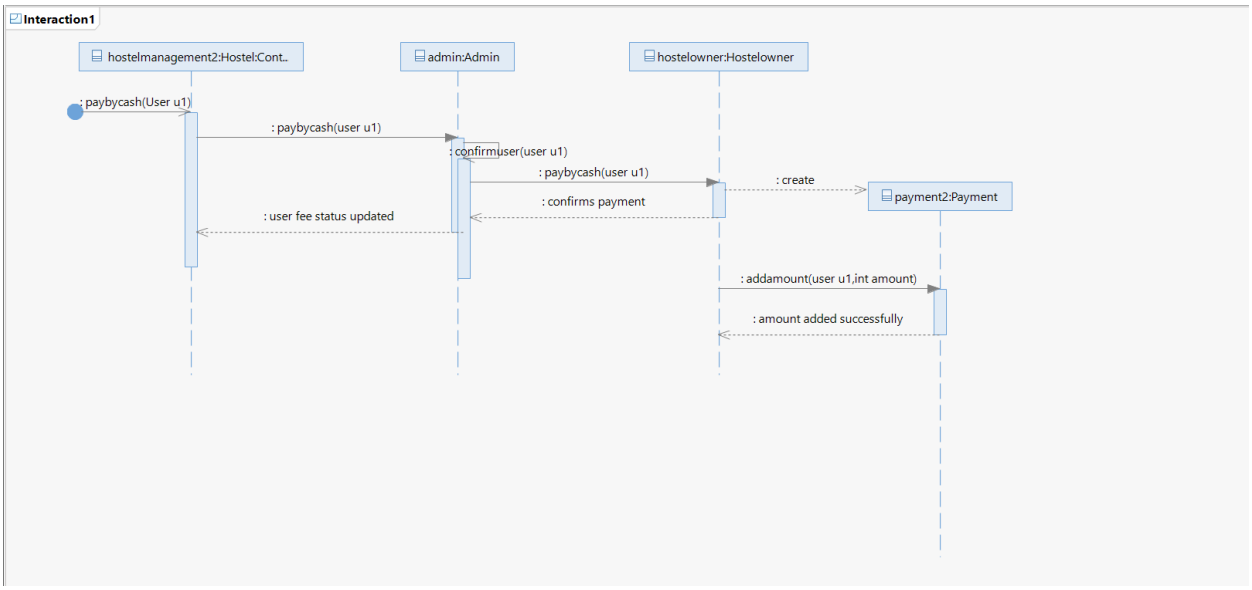

Interaction1
hostelmanagement3:Controller:Hostel:Controller
admin:Admin
user:User
1: changeuserinfo(User u1)
1.1: confirmadmin()
1.1.1: adminlogin()
1.2: confirms admin details
2: changeuserinfo(User u1)
2.1: updateuser(user u1)
2.2: details changed

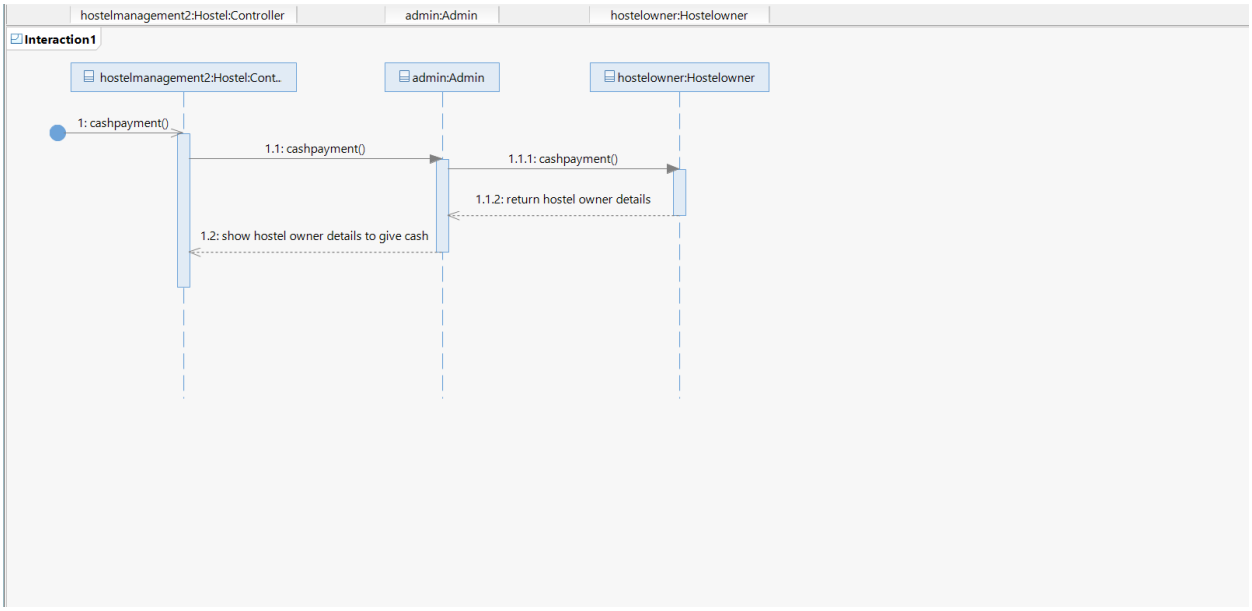

Interaction1
hostelmanagement3:Hostel:Cont..
admin:Admin
user:User
1: deleteuser(User u1)
1.1: confirmadmin()
1.1.1: adminlogin()
1.2: confirms admin details
2: deleteuser(User u1)
2.1: deleteuser(user u1)
2.2: user deleted

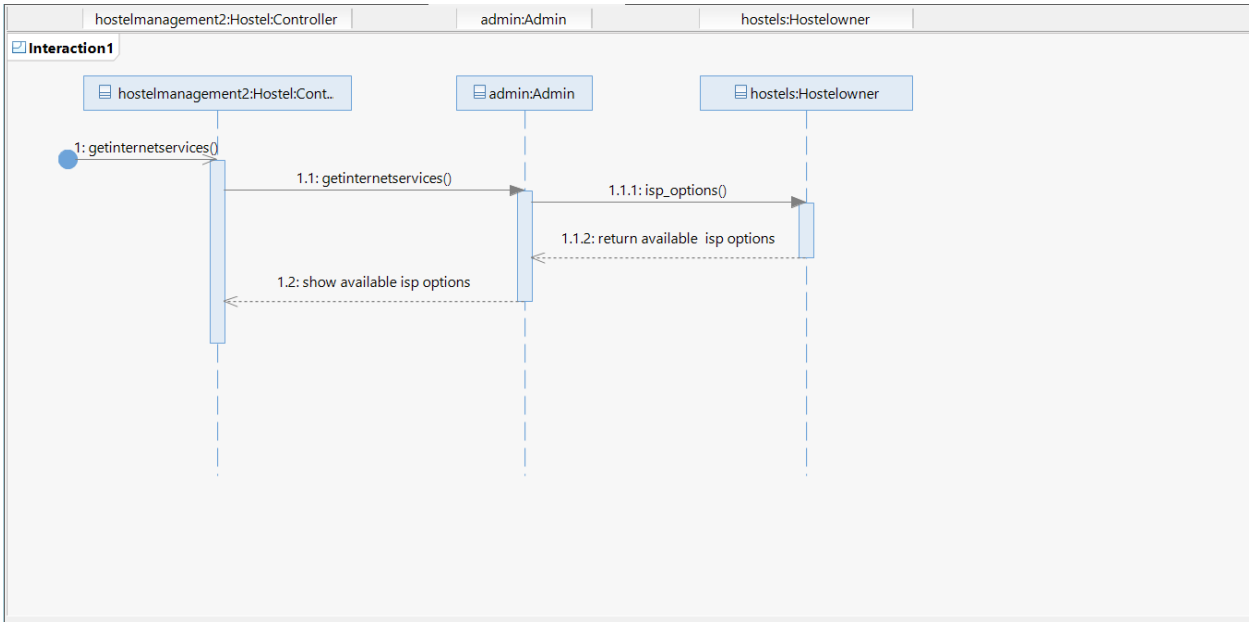

hostelmanage...
Admin:Actor
hostelowner:Hostelow...
hostelemplyees:Hostelo...
Interaction1
hostelmana...
Admin:Ac...
hostelowner:Hostelowner
hostelemplyees:Hostelow..
: cleaningservice(hostel hostelid,int room
: cleaningservice(hostel hostelid,int room
: cleaningservice(hostel hostelid,int room
: cleaningservice(int room)
: check availability of employee
: confirmas availability
: returns time
: returns time for cleaning

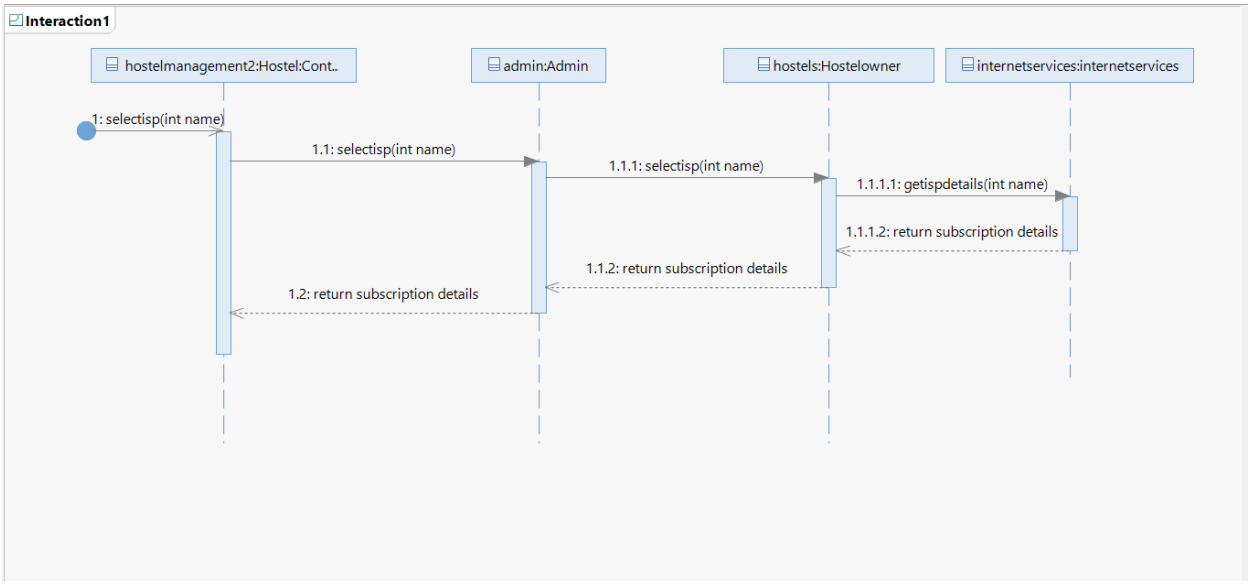

hostelmanage...
Admin:Actor
hostelowner:Hostelow...
hostelemployess:Hostel...
Interaction1
hostelmana...
Admin:Ac...
hostelowner:Hostelowner
hostelemployess:Hostelo...
: laundryservice(hostel hostelid,int room)
: laundryservice(hostel hostelid,int room)
: laundryservice(hostel hostelid,int room)
: laundryservice(int room)
: check availability of laundry mar
: confirmas availability
: returns time
: returns time for taking stuff

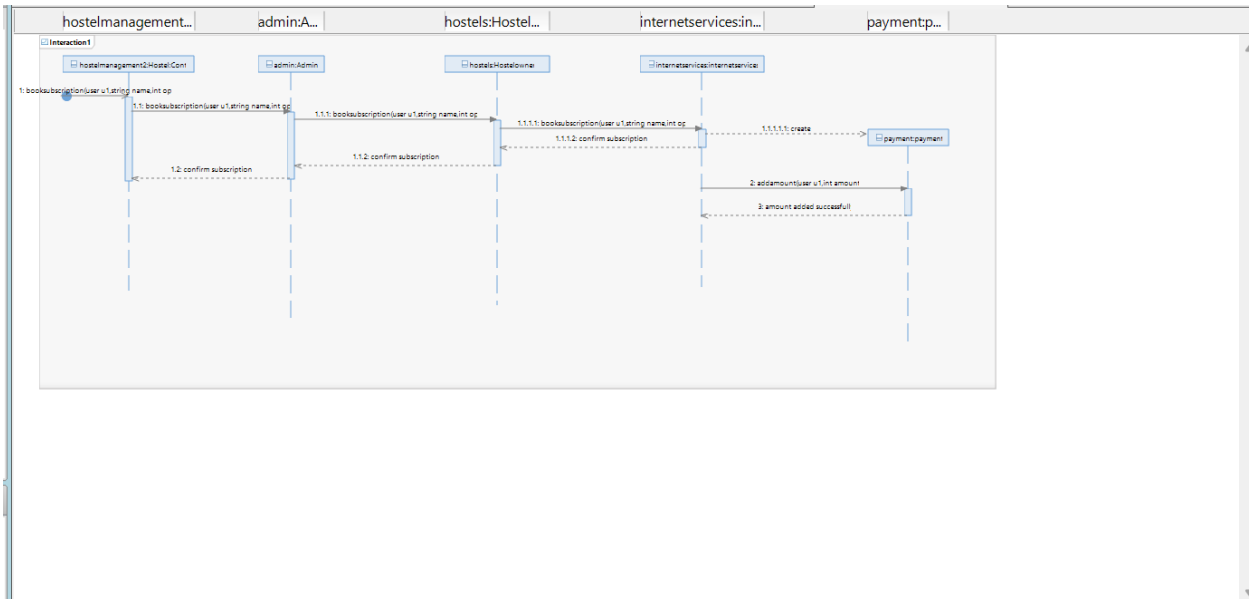

Interaction1
hostelmanagement2:Hostel:Cont...
admin:Admin
hostelowner:Hostelowner
1: paybill()
1.1: paybill()
1.1.1: paybilloptions()
1.1.2: return billing options
1.2: show payment options

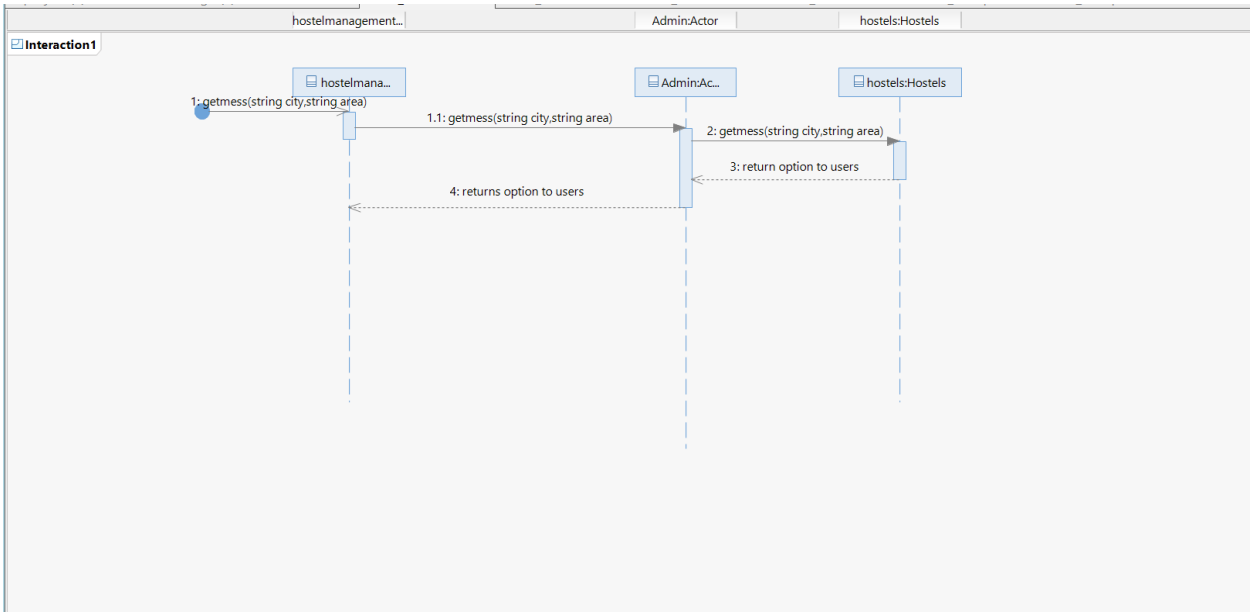

Interaction1
hostelmanagement2:Hostel:Cont..
admin:Admin
hostelowner:Hostelowner
payment2:Payment
: paybycash(User u1)
: paybycash(user u1)
: confirmuser(user u1)
: paybycash(user u1)
: create
: confirms payment
: user fee status updated
: addamount(user u1,int amount)
: amount added successfully

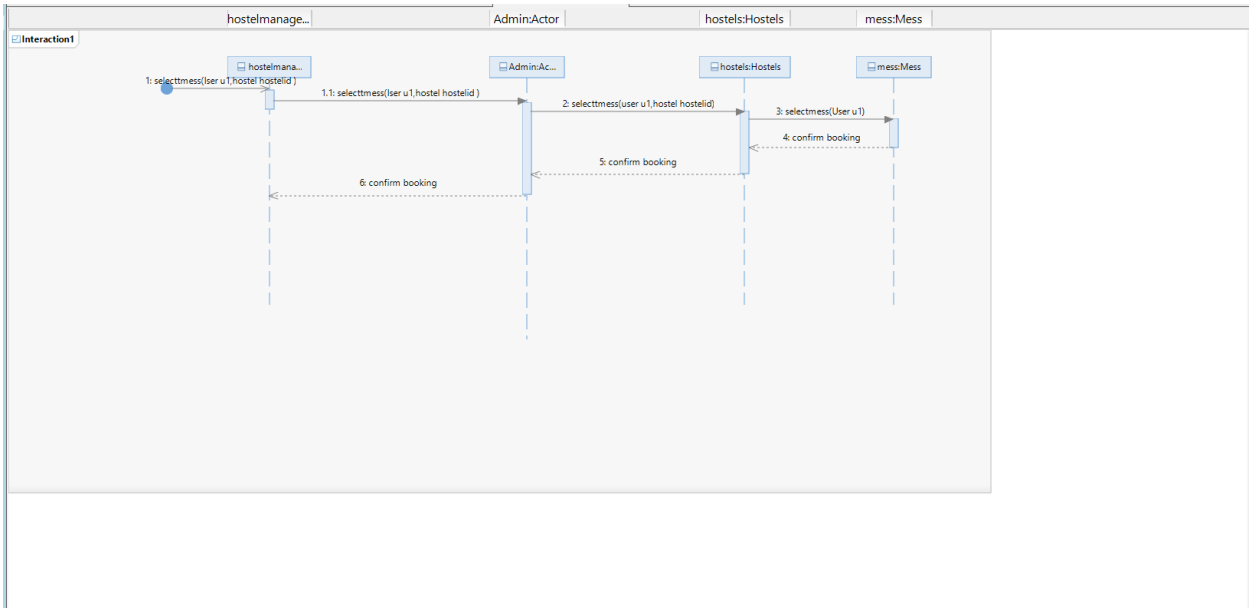

hostelmanagement2:Hostel:Controller
admin:Admin
hostelowner:Hostelowner
Interaction1
hostelmanagement2:Hostel:Cont..
admin:Admin
hostelowner:Hostelowner
1: cashpayment()
1.1: cashpayment()
1.1.1: cashpayment()
1.1.2: return hostel owner details
1.2: show hostel owner details to give cash

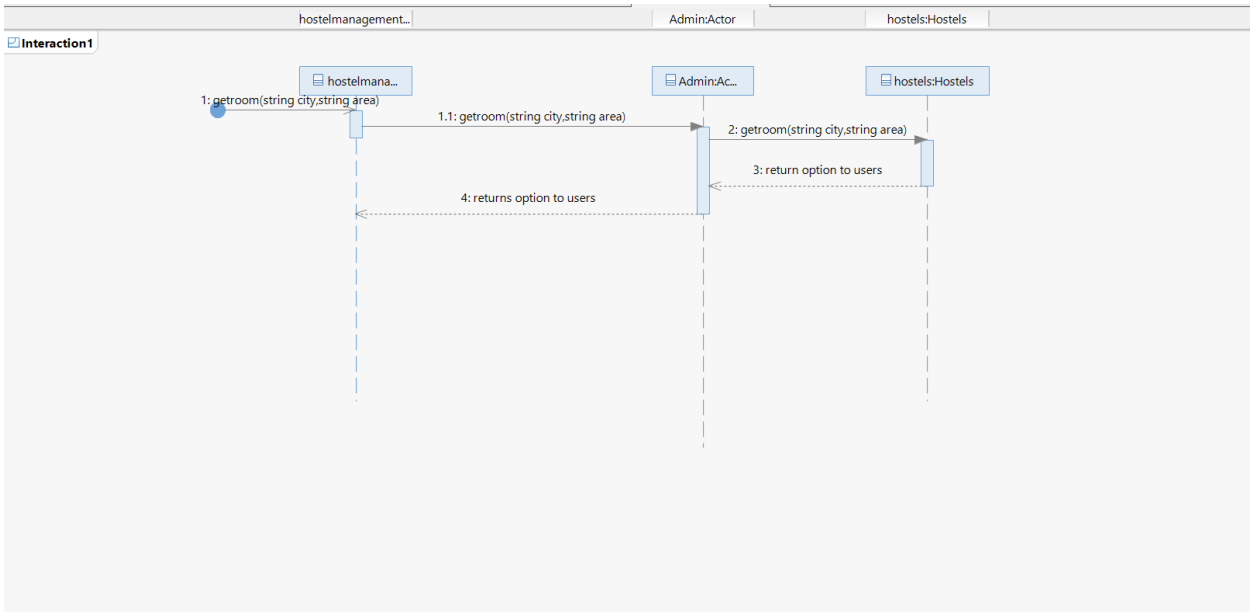

hostelmanagement2:Hostel:Controller
admin:Admin
hostels:Hostelowner
Interaction1
hostelmanagement2:Hostel:Cont..
admin:Admin
hostels:Hostelowner
1: getinternetservices()
1.1: getinternetservices()
1.1.1: isp_options()
1.1.2: return available isp options
1.2: show available isp options

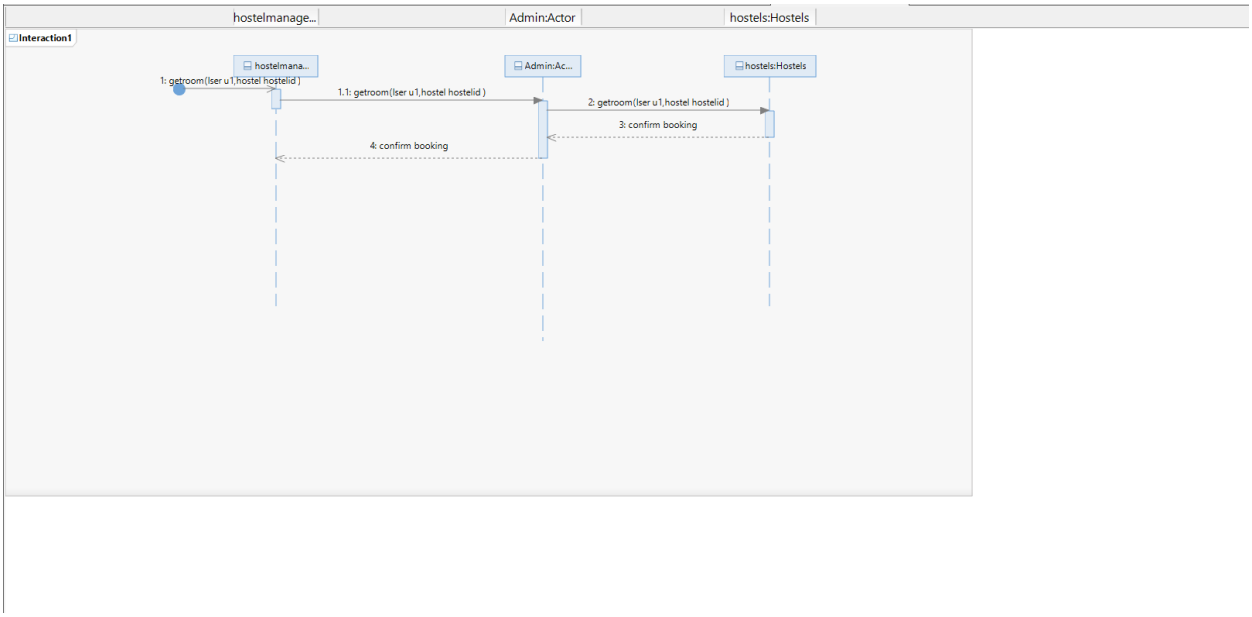

Interaction1
hostelmanagement2:Hostel:Cont..
admin:Admin
hostels:Hostelowner
internetservices:internetservices
1: selectisp(int name)
1.1: selectisp(int name)
1.1.1: selectisp(int name)
1.1.1.1: getispdetails(int name)
1.1.1.2: return subscription details
1.1.2: return subscription details
1.2: return subscription details

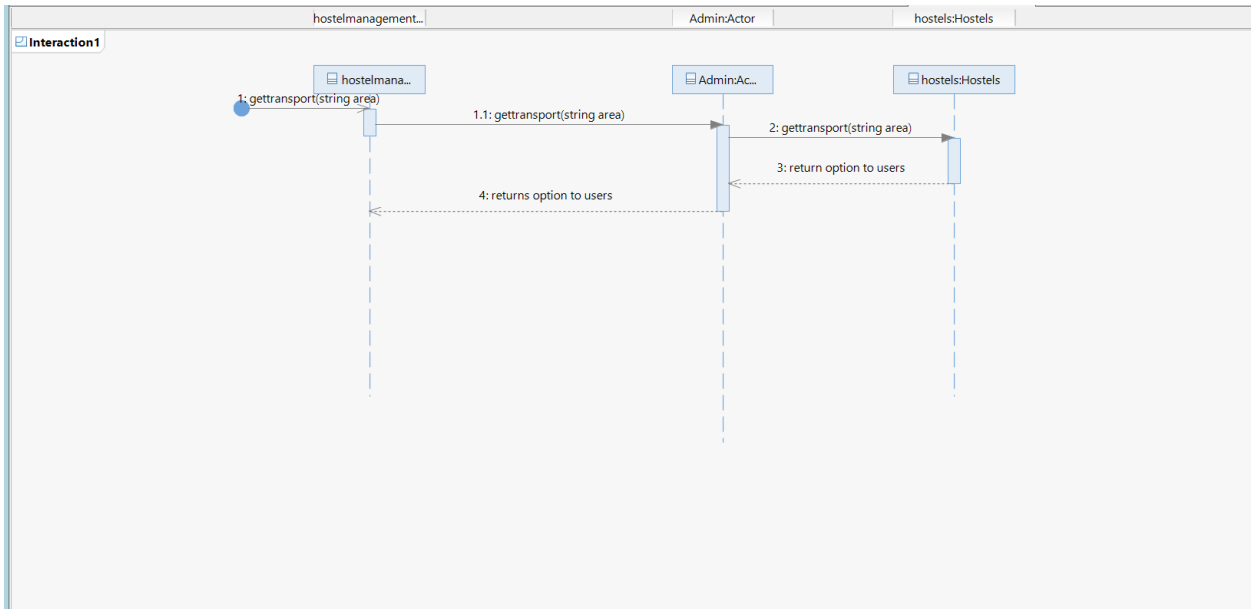

hostelmanagement...
admin:A...
hostels:Hostel...
internetservices:in...
payment:p...
Interaction1
hostelmanagement2:Hostel:Cont
admin:Admin
hostels:Hostelowner
internetservices:internetservice
1: booksubscription(user u1,string name,int op
1.1: booksubscription(user u1,string name,int op
1.1.1: booksubscription(user u1,string name,int op
1.1.1.1: booksubscription(user u1,string name,int op
1.1.1.1.1: create
payment:payment
1.1.1.2: confirm subscription
1.1.2: confirm subscription
1.2: confirm subscription
2: addamount(user u1,int amount
3: amount added successfull

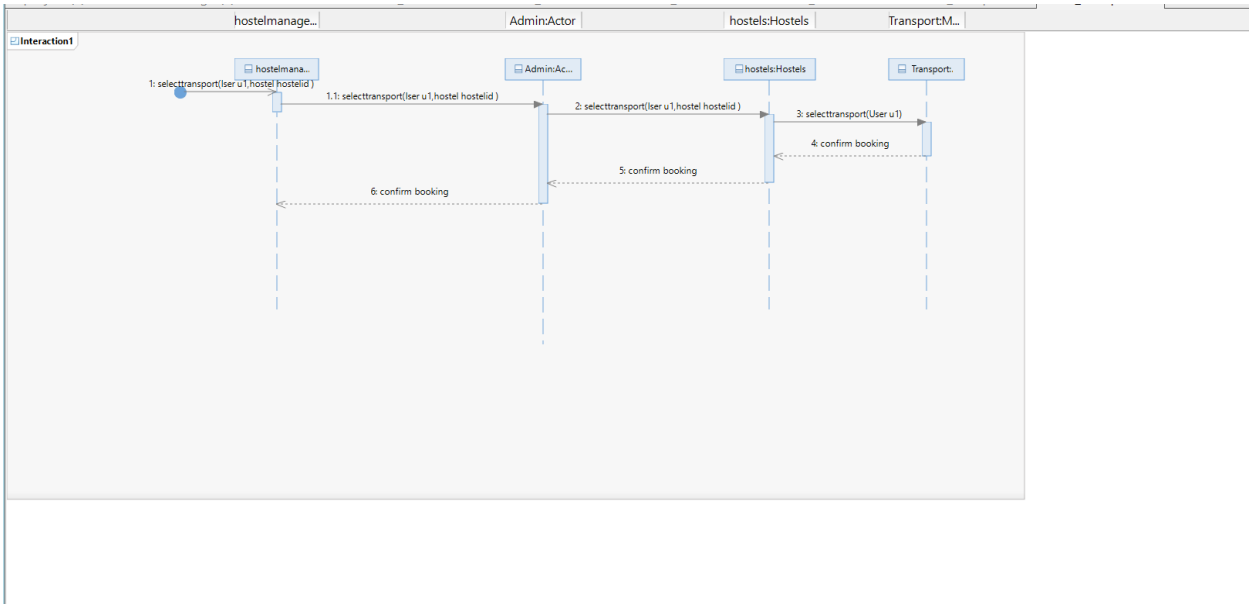

hostelmanagement...
Admin:Actor
hostels:Hostels
Interaction1
hostelmana...
Admin:Ac...
hostels:Hostels
1: getmess(string city,string area)
1.1: getmess(string city,string area)
2: getmess(string city,string area)
3: return option to users
4: returns option to users

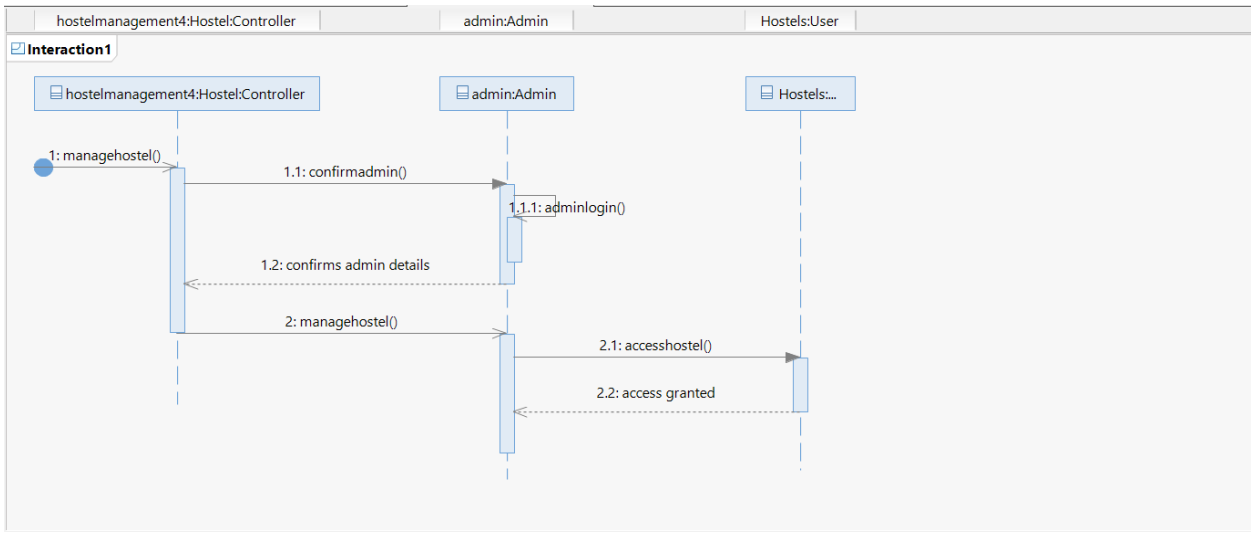

hostelmanage...
Admin:Actor
hostels:Hostels
mess:Mess
Interaction1
hostelmana...
Admin:Ac...
hostels:Hostels
mess:Mess
1: selecttmess(Iser u1,hostel hostelid)
1.1: selecttmess(Iser u1,hostel hostelid)
2: selecttmess(user u1,hostel hostelid)
3: selectmess(User u1)
4: confirm booking
5: confirm booking
6: confirm booking

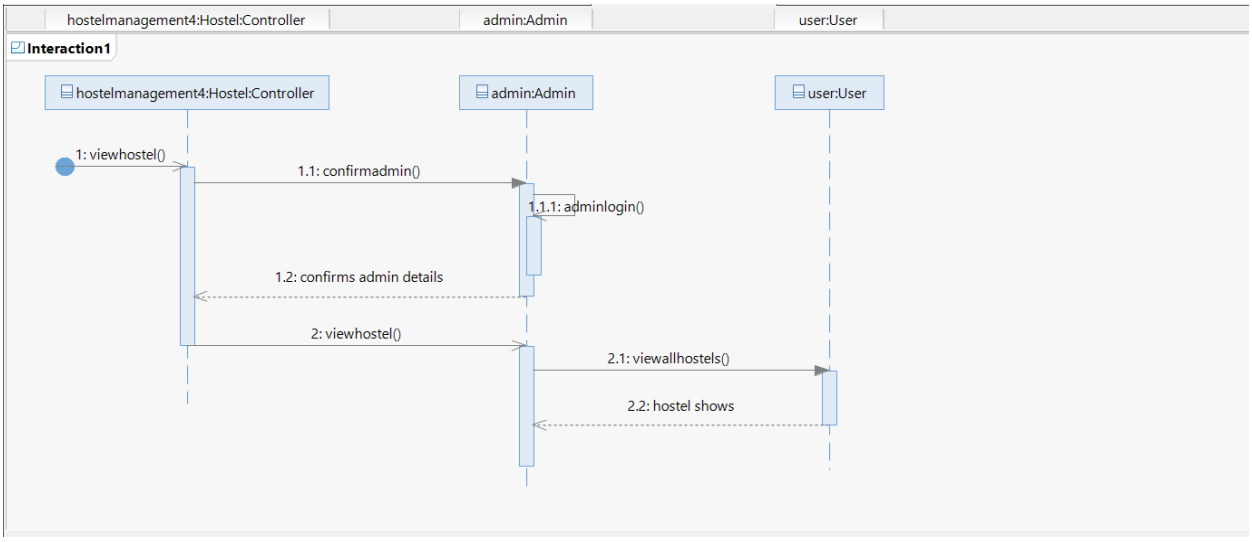

hostelmanagement...
Admin:Actor
hostels:Hostels
Interaction1
hostelmana...
Admin:Ac...
hostels:Hostels
1: getroom(string city,string area)
1.1: getroom(string city,string area)
2: getroom(string city,string area)
3: return option to users
4: returns option to users

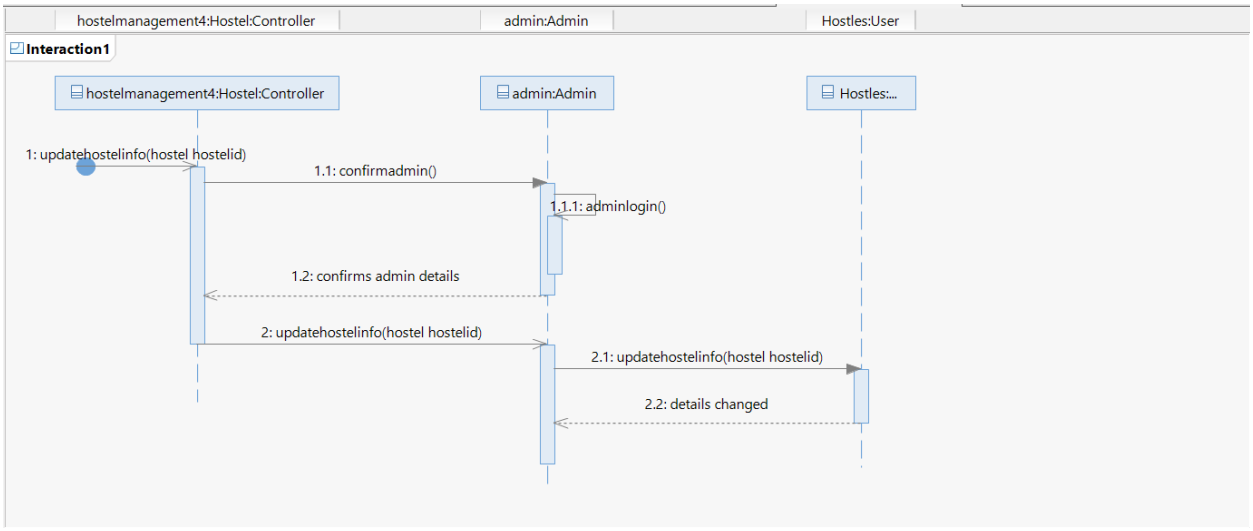

hostelmanage...
Admin:Actor
hostels:Hostels
Interaction1
hostelmana...
Admin:Ac...
hostels:Hostels
1: getroom(lser u1,hostel hostelid)
1.1: getroom(lser u1,hostel hostelid)
2: getroom(lser u1,hostel hostelid)
3: confirm booking
4: confirm booking

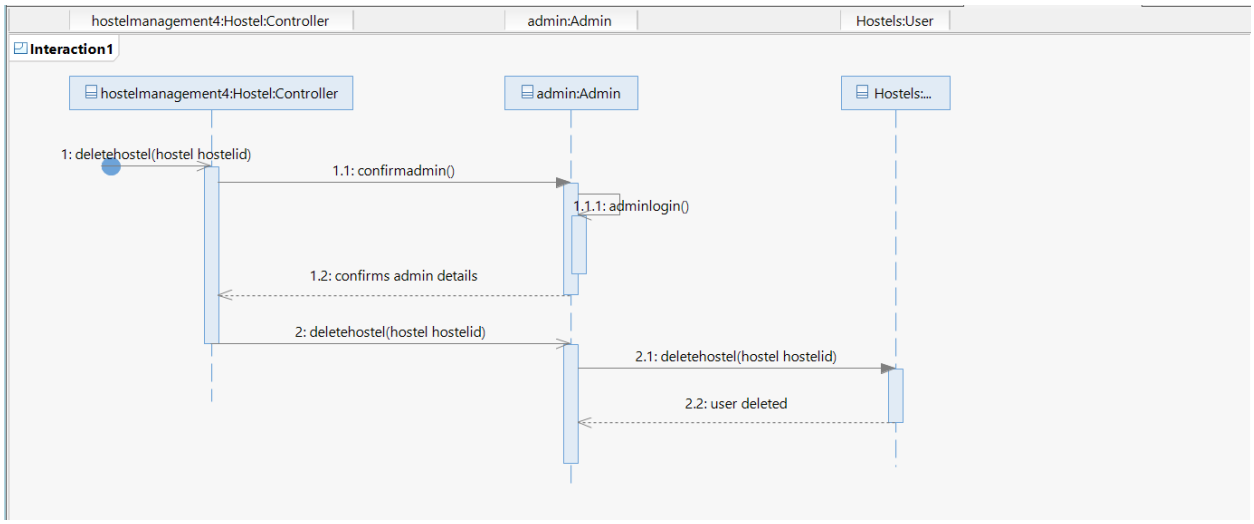

hostelmanagement...
Admin:Actor
hostels:Hostels
Interaction1
hostelmana...
Admin:Ac...
hostels:Hostels
1: gettransport(string area)
1.1: gettransport(string area)
2: gettransport(string area)
3: return option to users
4: returns option to users


hostelmanage...
Admin:Actor
hostels:Hostels
Transport:M...
Interaction1
hostelmana...
Admin:Ac...
hostels:Hostels
Transport:
1: selecttransport(lser u1,hostel hostelid)
1.1: selecttransport(lser u1,hostel hostelid)
2: selecttransport(lser u1,hostel hostelid)
3: selecttransport(User u1)
4: confirm booking
5: confirm booking
6: confirm booking

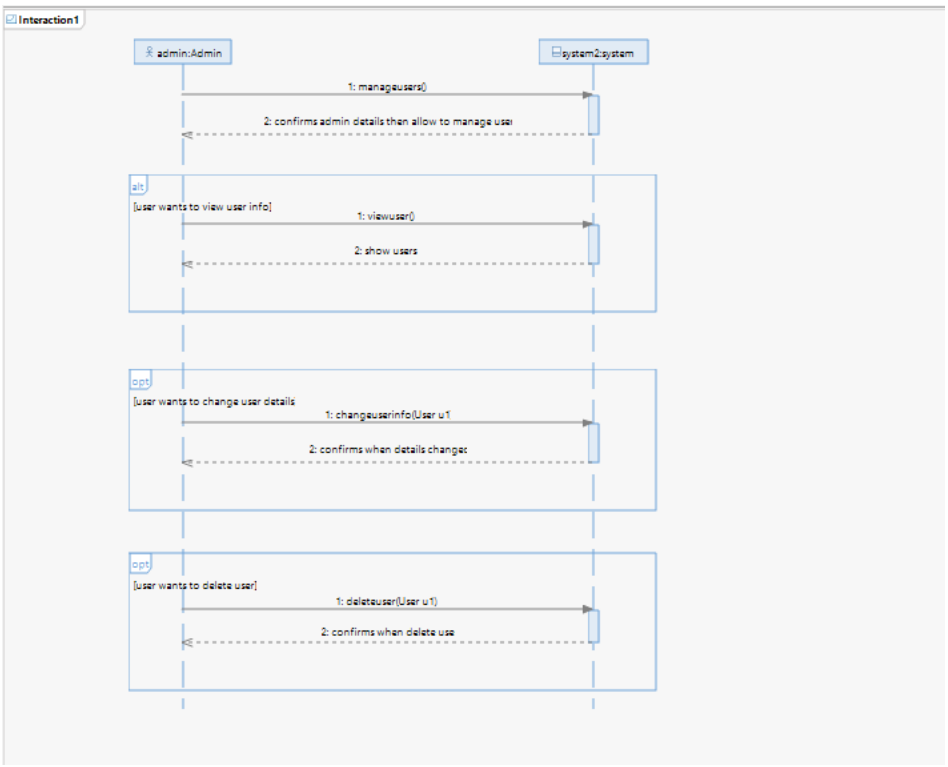

hostelmanagement4:Hostel:Controller
admin:Admin
Hostels:User
Interaction1
hostelmanagement4:Hostel:Controller
admin:Admin
Hostels:...
1: managehostel()
1.1: confirmadmin()
1.1.1: adminlogin()
1.2: confirms admin details
2: managehostel()
2.1: accesshostel()
2.2: access granted

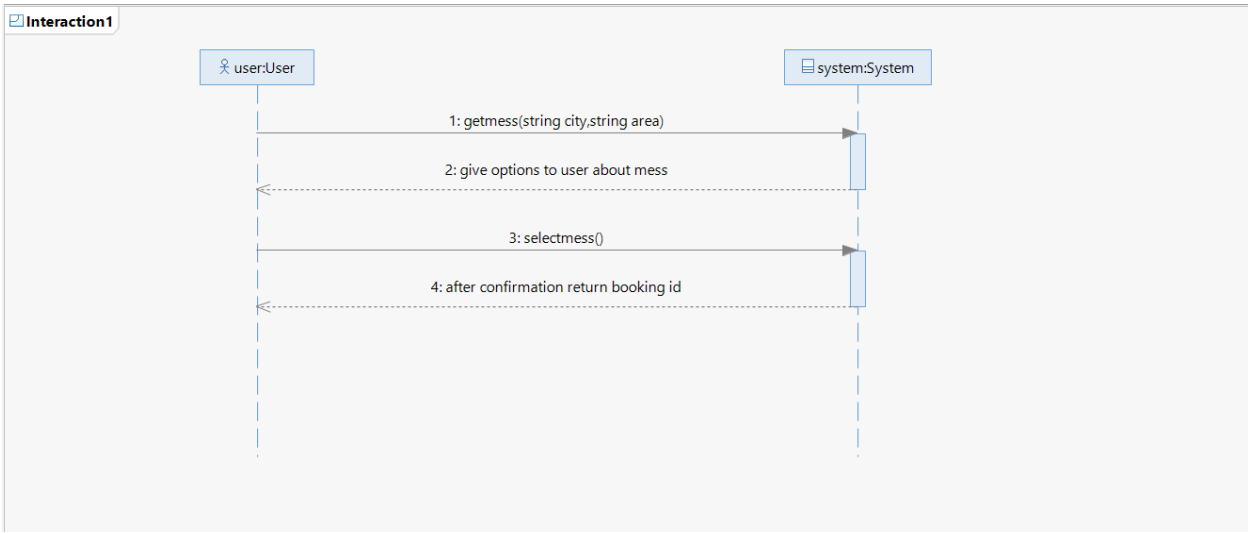

hostelmanagement4:Hostel:Controller
admin:Admin
user:User
Interaction1
hostelmanagement4:Hostel:Controller
admin:Admin
user:User
1: viewhostel()
1.1: confirmadmin()
1.1.1: adminlogin()
1.2: confirms admin details
2: viewhostel()
2.1: viewallhostels()
2.2: hostel shows

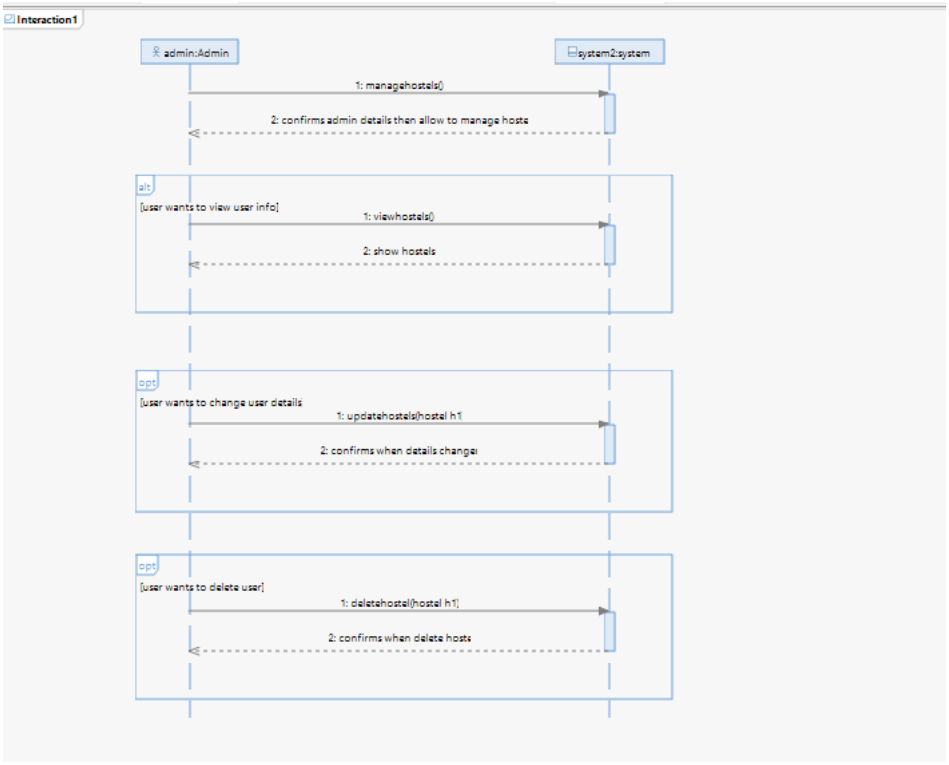

hostelmanagement4:Hostel:Controller
admin:Admin
Hostles:User
Interaction1
hostelmanagement4:Hostel:Controller
admin:Admin
Hostles:...
1: updatehostelinfo(hostel hostelid)
1.1: confirmadmin()
1.1.1: adminlogin()
1.2: confirms admin details
2: updatehostelinfo(hostel hostelid)
2.1: updatehostelinfo(hostel hostelid)
2.2: details changed

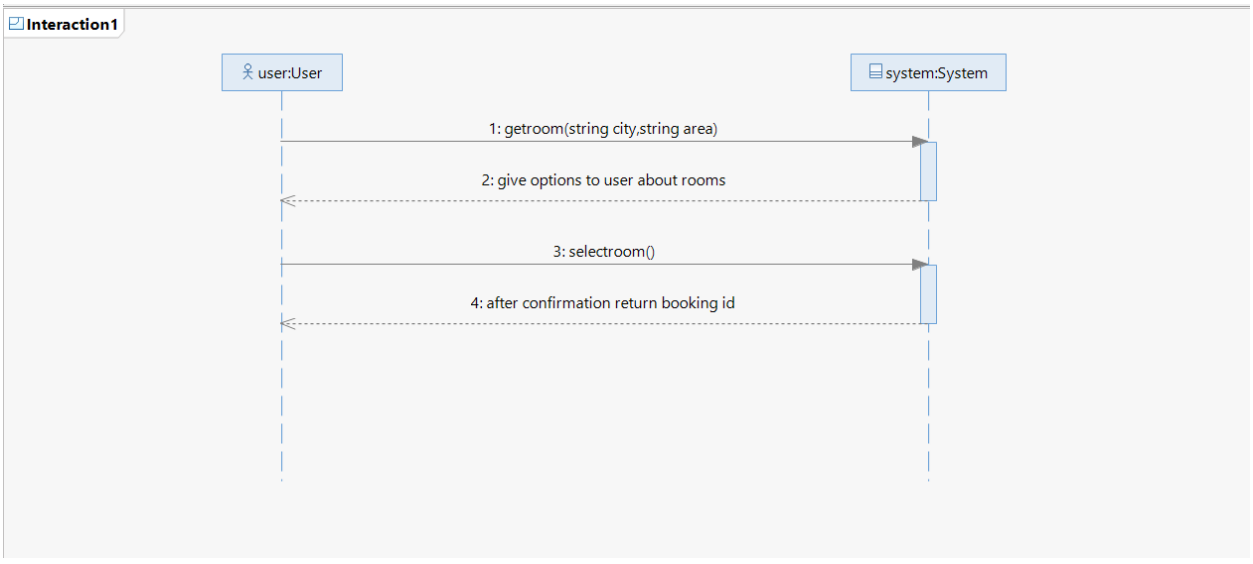

hostelmanagement4:Hostel:Controller
admin:Admin
Hostels:User
Interaction1
hostelmanagement4:Hostel:Controller
admin:Admin
Hostels:...
1: deletehostel(hostel hostelid)
1.1: confirmadmin()
1.1.1: adminlogin()
1.2: confirms admin details
2: deletehostel(hostel hostelid)
2.1: deletehostel(hostel hostelid)
2.2: user deleted

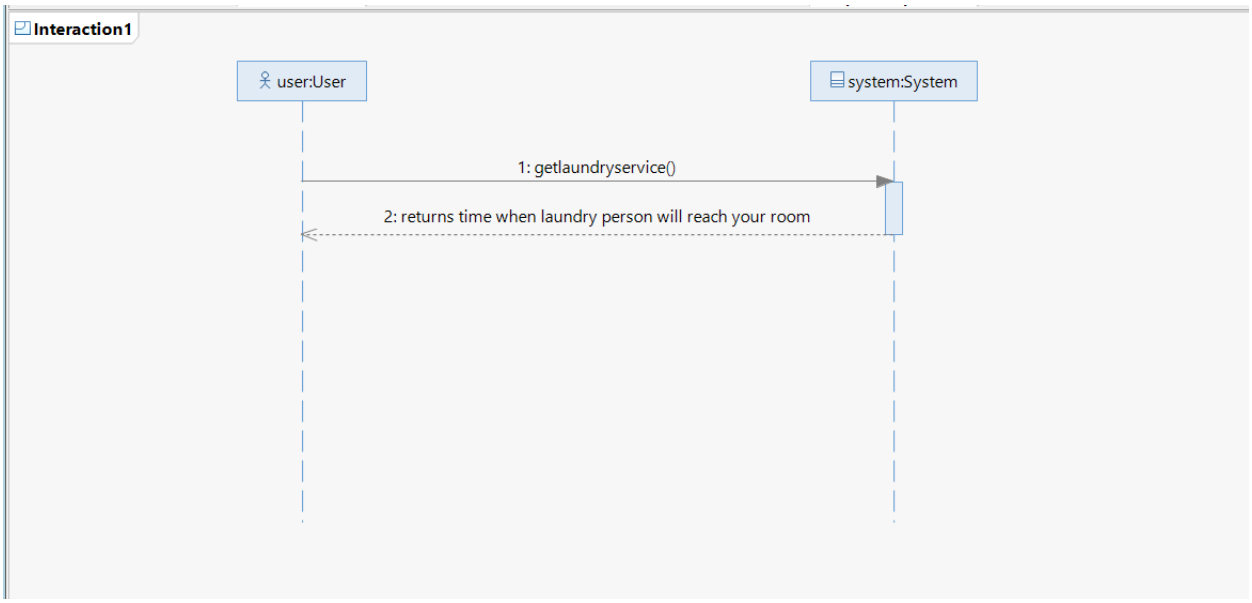
6. System Sequence Diagram

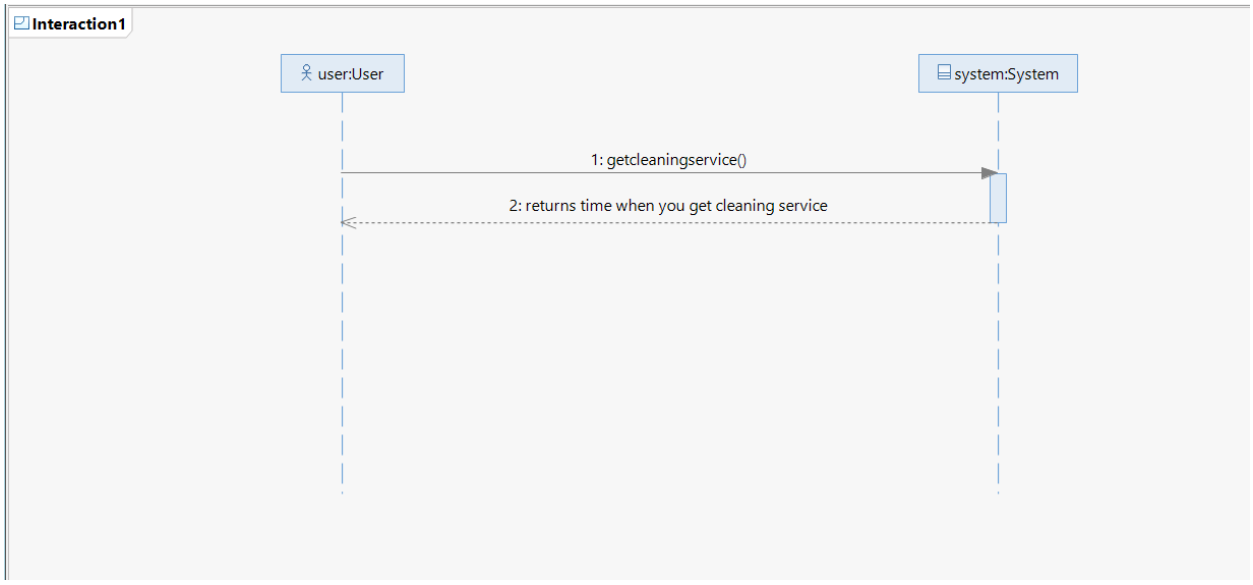


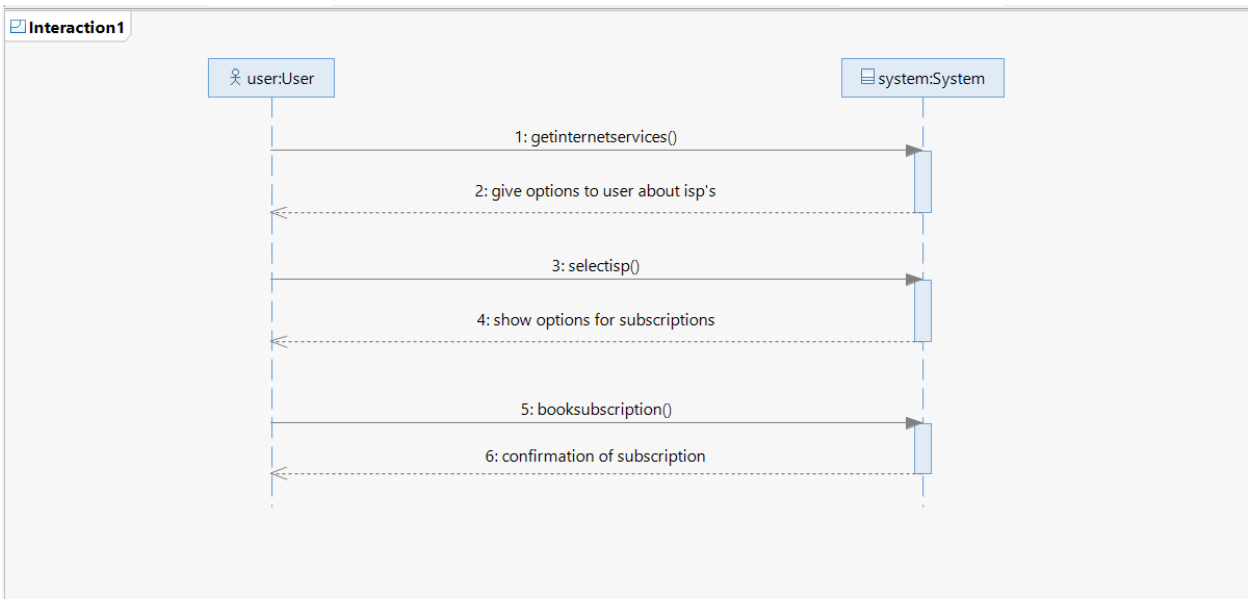


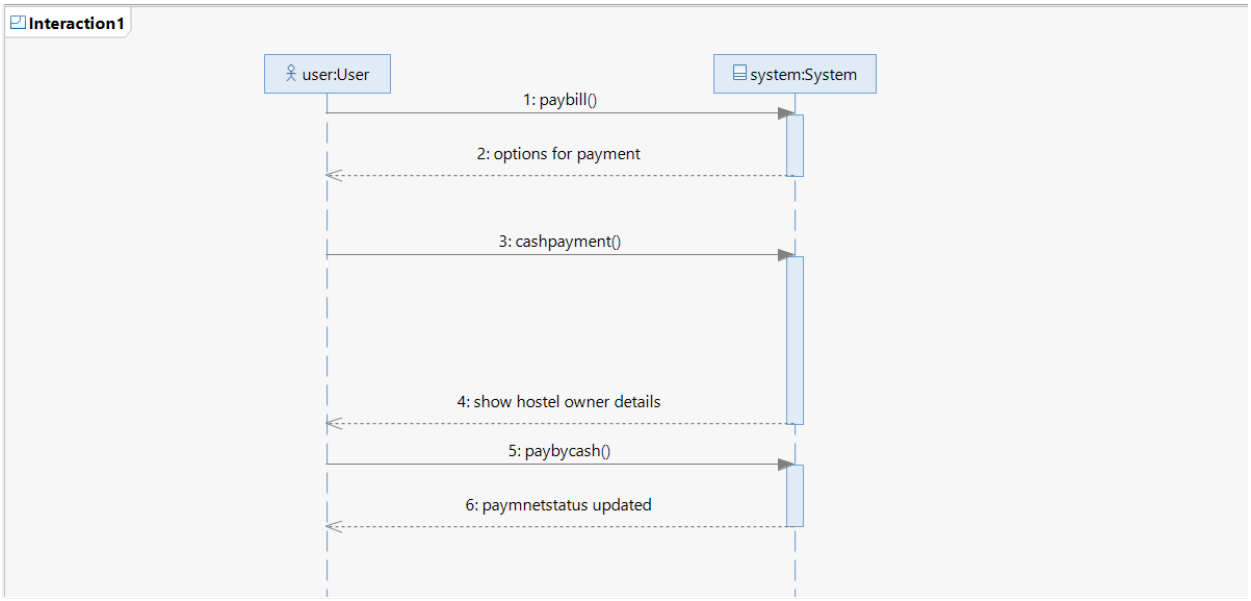

Interaction1
admin:Admin
system2:system
1: managehostels()
2: confirms admin details then allow to manage hoste
alt
[user wants to view usar info]
1: viewhostels()
2: show hostels
opt
[user wants to change usar details
1: updatehostels(hostel h1
2: confirms when details change
opt
[user wants to delete usar]
1: deletehostel(hostel h1]
2: confirms when delete hoste

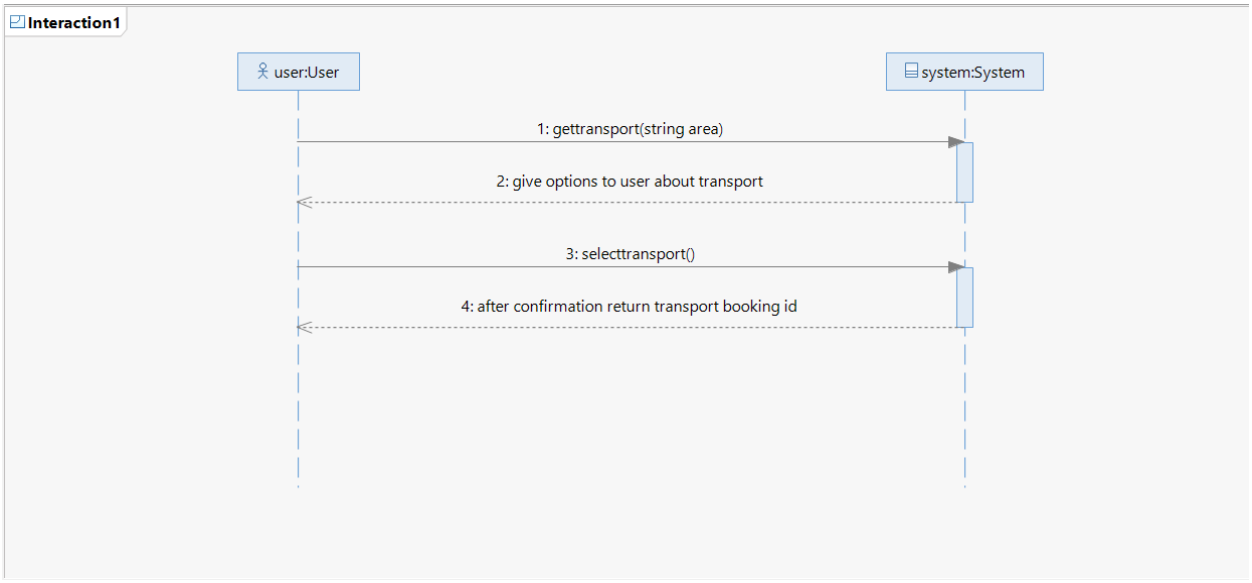

Interaction1
user:User
system:System
1: getroom(string city,string area)
2: give options to user about rooms
3: selectroom()
4: after confirmation return booking id


Interaction1
user:User
system:System
1: getlaundryservice()
2: returns time when laundry person will reach your room


Interaction1
user:User
system:System
1: getcleaningservice()
2: returns time when you get cleaning service


Interaction1
user:User
system:System
1: getinternetservices()
2: give options to user about isp's
3: selectisp()
4: show options for subscriptions
5: booksubscription()
6: confirmation of subscription


Interaction1
user:User
system:System
1: paybill()
2: options for payment
3: cashpayment()
4: show hostel owner details
5: paybycash()
6: paymnetstatus updated


Interaction1
user:User
system:System
1: gettransport(string area)
2: give options to user about transport
3: selecttransport()
4: after confirmation return transport booking id

7. Class Diagram