SEP1 Project Report

V-Class Group 3:

Andrei CHIRTOACA (239929), Dana Rusu (239944), KOnstantina Blazhukova (240232) & Tamas Boda (239839)

SuperVisors:

Ole Ildsgaard Hougaard & Alexis Claire Walhovd

2015

**Abstract**

*The Deer Alley Hotel owner was hesitant to use equipment like computers and other modern conveniences, but because the world is always evolving there was a lack of booking system.*

*The Deer Alley Hotel System is a program that registers booking for time ahead depending on the desired time period and room or suite type. Besides just making booking, the user can edit or cancel it.*

*One of the main tasks of the program is to be able to check-in all the guests that arrive at the hotel. The user searches for the booking by the name and if one is found, then it proceeds to adding all of the other guests, that are going to stay at the hotel. The user registers the guests by inputting information as: name, phone number, email, home address, nationality and passport number. The check-out part of the program is used to erase the guests that stayed at the hotel from the system. At the check-out the system also calculates the price for the whole stay, including the extra services, if there are any.*

*The system’s graphic interface is designed to be easy for usage and at the same time to have a nice, well-organized vision.*

*Although some of our methods do not work as good as we wanted, they still fulfil the requirements and allow the system to perform well and execute the main functionalities.*

*.*

Table of Contents

**2. Introduction3**

**3. Analysis**

3.1 Functional and non-functional requirements4

3.2 Use case diagram5

3.3 Use case descriptions5

3.4 Activity diagram…………………………………………………………………………………………………………..6

**4. Design**

4.1 The Deer Alley Hotel System Design7

4.2 GUI design7-8

4.3 UML class diagrams design 9

**5.Implementation**10-12

**6. Design**13-15

**7. Results**15

**8. Discussion**16

**9. Conclusion**17

**10. Appendices**17

**2. Introduction**

The Deer Alley Hotel is a small hotel located in a large Central European town. The owner of the hotel has always been hesitant to use modern equipment, like computers, in order to administrate it.

However, the lack of a change, of something new, was demanded, so the owner decided to reach us to help him with that problem.

The Deer Alley booking system is a program that is able to store information about bookings and check-ins. It contains five different options: make a booking, edit a booking, cancel a booking, make a check-in and make a check-out.

In the make a booking compartment, the system is able to check the availability of the rooms/suites by inputting the time period of the requested stay. When the available room is found, the user inputs the name of the guest that is making the booking.

Edit a booking allows the system to make changes for a specific booking, that is searched by the name of the person, who requested the booking.

Option number three is “Cancel a Booking”. The user searches for the booking by the name of the guest and removes the booking from the system.

“Make a Check-In” is the option of the system that adds all of the guests that are going to stay in specific room/suite, that has been already booked. It adds this specific information: name, home address, phone number, nationality, email, birth date and passport number.

In the “Check-Out” option, the system allows the user to add services and calculate the total price for the stay. In the end, the general information including stay period and price is shown in an “INVOICE” frame, and then it is removed from the system.

**System purpose**

The purpose of the system is to ease the work of the employees and to make sure that the hotel is not losing information about bookings and its guests.

**Name of the system**

The Deer Alley Hotel Booking System.

**3. Analysis**

**3.1 Functional and non-functional requirements**

**Functional requirements**

1. The user should be able to make a booking

* The system should store guest’s name, arrival and departure dates, type of booked room and room number.

1. The user should be able to make a check-in

* The system should store all guests’ names, home addresses, phone numbers, e- mails, passport numbers, dates of birth, arrival date and expected departure date.
* The system should set the rooms to unavailable.

1. The user should be able to check for available rooms

* The system must check available rooms by price and time period.
* The system must display the number of the available room.

1. The user should be able to search for a booking or check-in

* The system should find booking or check-in by guest name.
* The system should display the booking/check-in information.

1. The user should be able to add an extra bed and extra-services such as: laundry, service room and mini-bar items.

6. The system should be able to do a check-out and calculate the total price for the guest’s stay period.

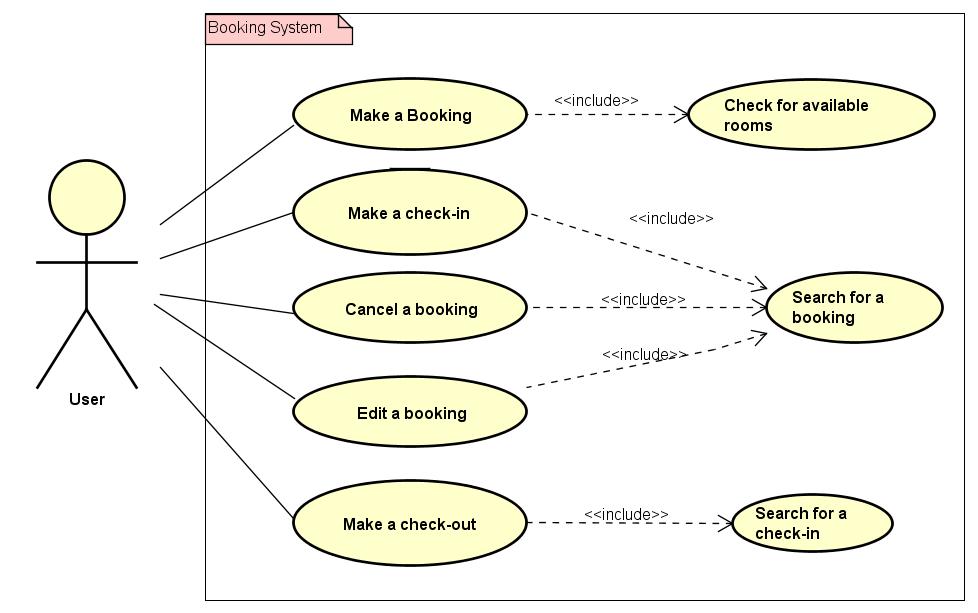
7. The user should be able to edit booking information.

8. The user should be able to cancel a booking.

**Non-functional requirements**

1. The system must contain bookings and check-ins.
2. The system should allow the user to choose from five options – “Make a Booking”, “Make a Check-In”, “Edit a Booking”, “Make a Check-Out” and “Cancel a Booking”.
3. The system must be written in Java.
4. The system must be user-friendly.

**3.2 Use case diagram**

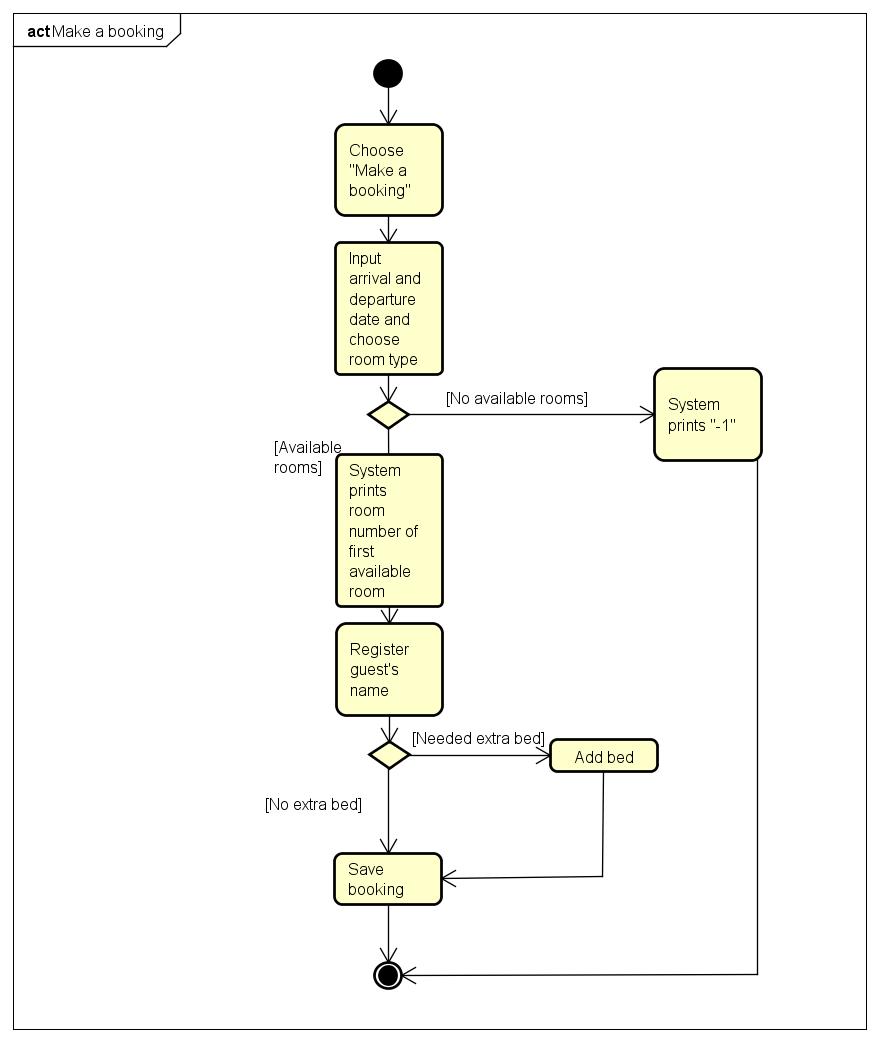
****

**3.3 Use case descriptions**

1. **Make a booking -** The user is making a booking by checking the availability for the preferred room type and time period. If there are any available rooms, the user registers the guest’s name and sets the room to unavailable for the certain dates.
2. **Make a check-in -** The user searches for the guest’s booking by name. If the booking is found, he registers every guest’s information and saves it.
3. **Cancel a booking -** The user searches for the booking by name and removes it from the system.
4. **Change a booking -** The user searches for the booking by name and changes the registered information.
5. **Make a check-out -** The user searches for a check-in by name. If the check-in is found, its information is displayed – this includes name, arrival and departure dates, room number, price and extra bed. If there are extra services, the user chooses them and the price is automatically added to the total price.

***(See all of the Activity diagrams in Appendices A)***

**3.4 Activity diagram**

****

The most important function of the system is to make a booking, because this is actually when you check the available rooms by certain parameters and occupy the right one. This diagram shows the process of making a booking step by step.

***(See all of the Activity diagrams in Appendices B)***

**4.Design**

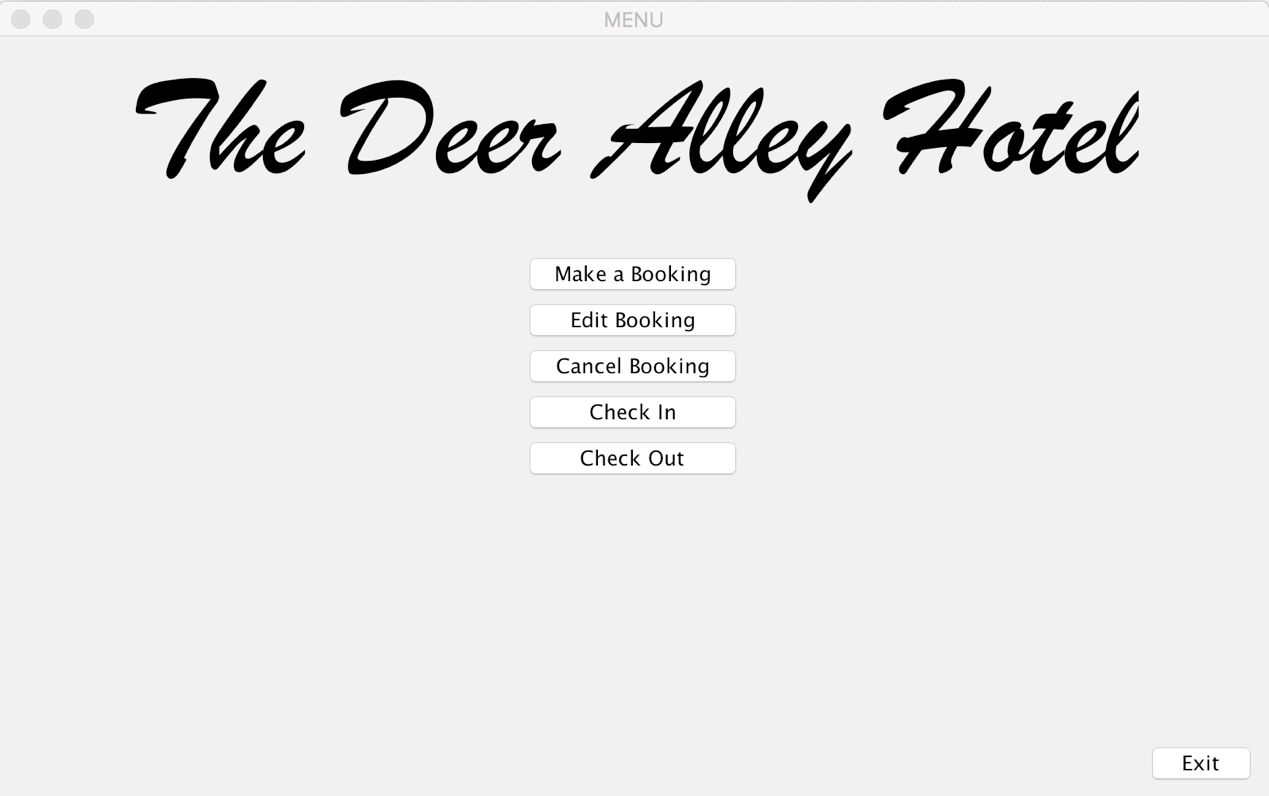
**4.1 The Deer Alley Hotel System design**

The booking system consists of Java classes and GUI. The purpose of the Java classes is to make the program function properly as it was initially planned. In the classes, important objects and methods are created, likewise storing and getting information from the user interface.

The GUI also consists of multiple Java classes, which implement different buttons, labels and text fields in consideration of making the system more organized and user-friendly.

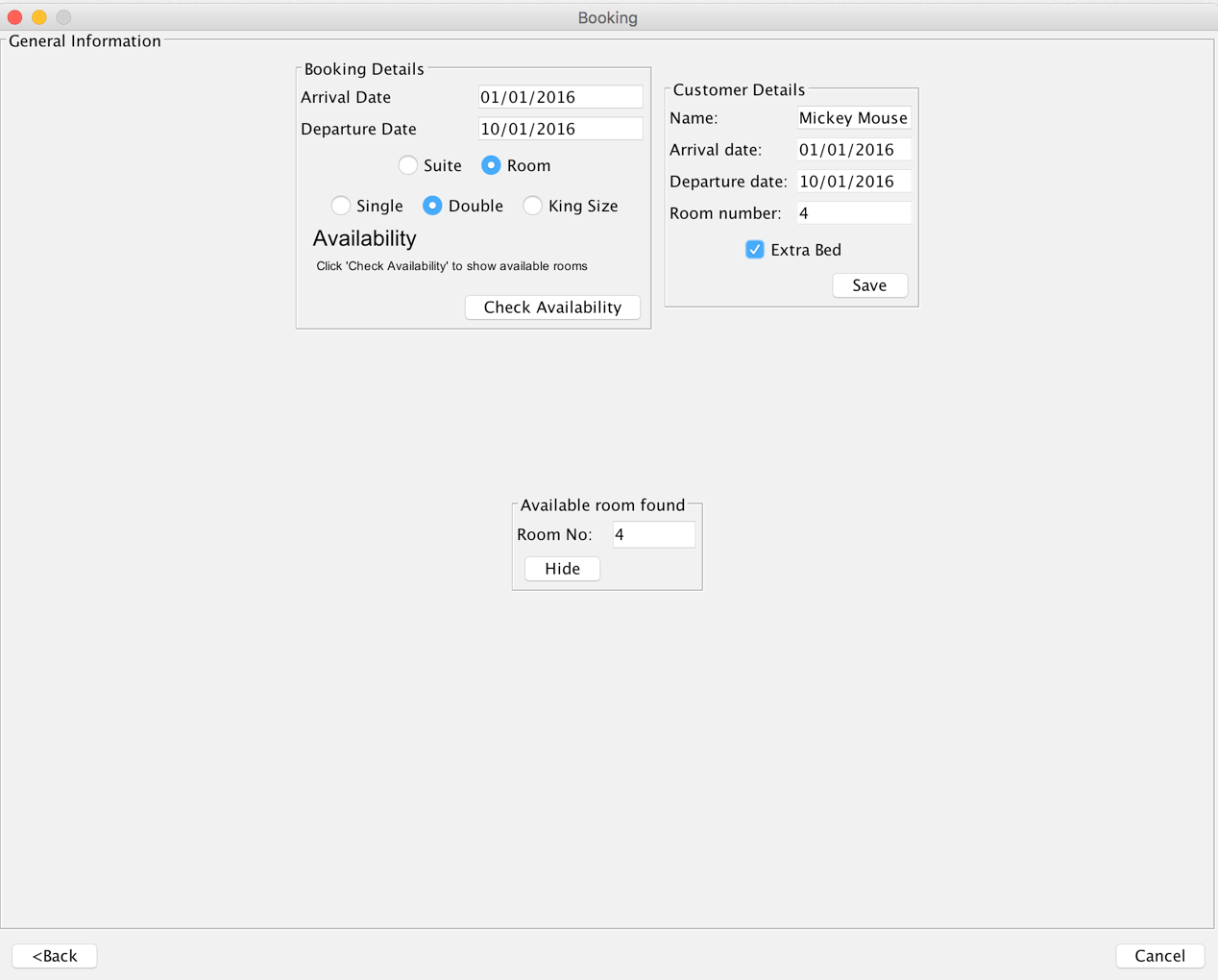
**4.2 GUI Design**

The Graphical User Interface for the Deer Alley Hotel Booking System is a balanced combination between user-friendliness and practicability.



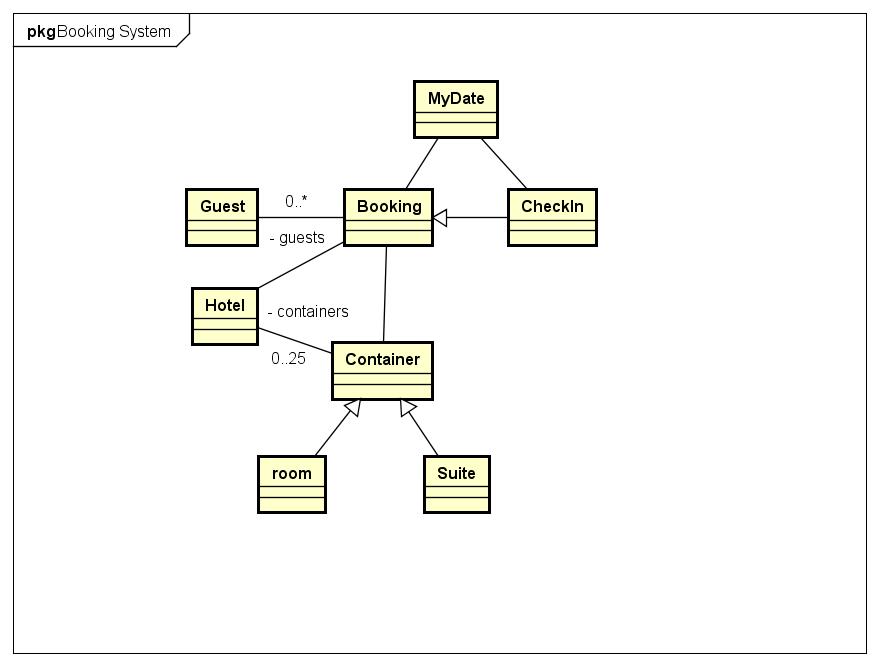
The MENU is the main frame of the GUI, containing the name of the Hotel and 5 buttons which allow you to do everything that you need in order to handle an entire hotel.

Behind each button a new frame stands, from which you can administer precisely your bookings, check-ins and guest information.

Making a booking is as simple as this:

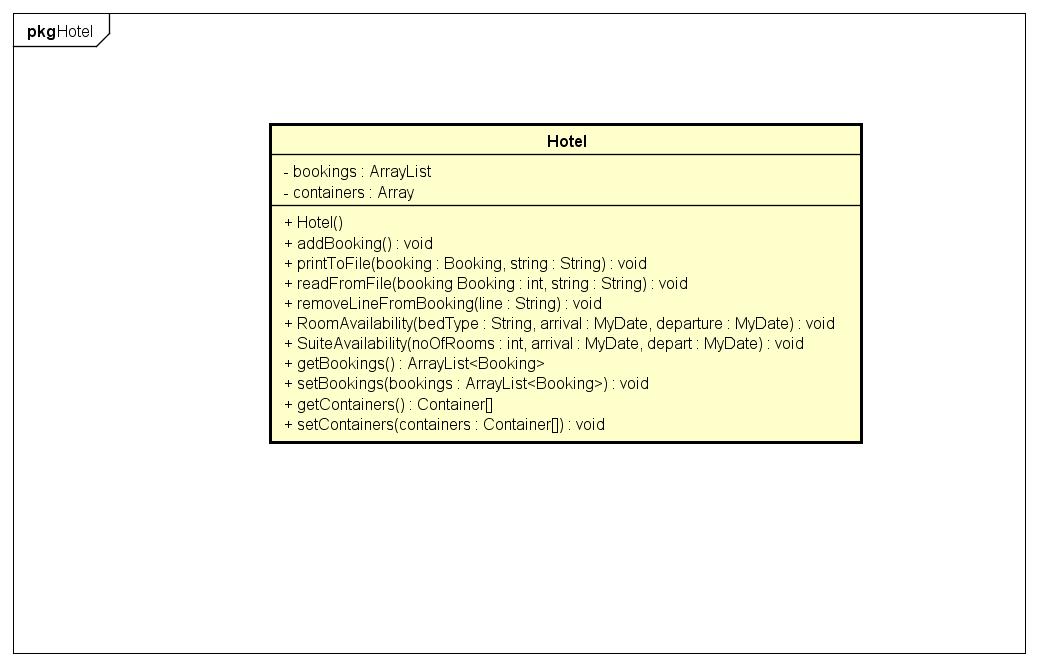
Each frame contains a *”Back”* and a *”Close”* button, which when clicked, get you back to the MENU of the System.

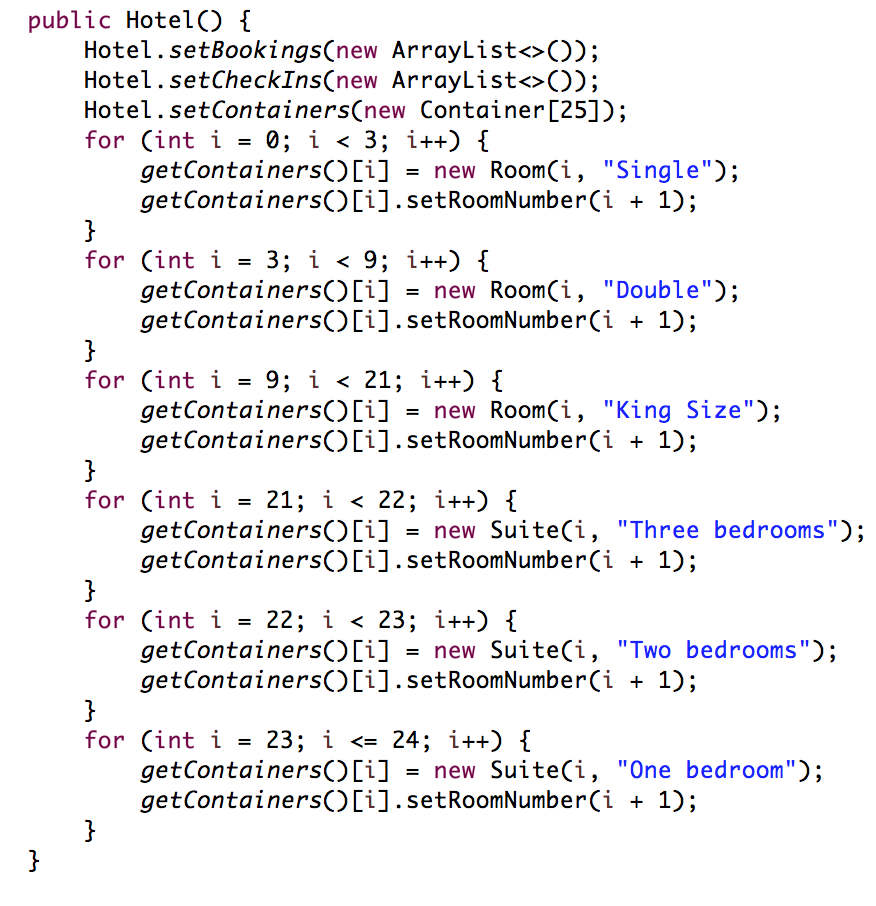
* 1. **UML class diagrams design**

****

The picture above shows the Java Classes and how they interact between each other. The Hotel class is the most important class, used to store the containers with the room and suite objects, as well as the availability method and methods which read and remove from text files. The Container class is used to store all the methods which are used by both Room and Suite classes. The Booking class stores a container object, which is used in order to access the room and suite objects, two MyDate objects and an array list of guests. The CheckIn class extends Booking, because it uses the registered information in the booking. It is also used for calculating the total price in the end. The MyDate class is simply used for creating new date objects with minutes, hours, days, months and years as parameters. The Guest class is used for creating objects which will store the guest’s information. It has two constructors because different guest information is saved in the booking and in the check in.

1. **Implementation**

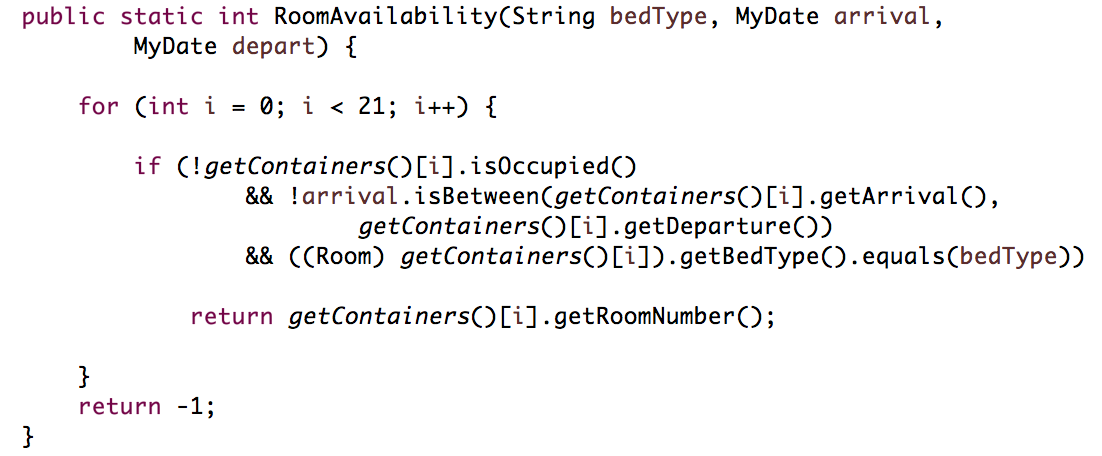


This is the Hotel class diagram. It shows the array list and array attributes, which are used to store the booking objects and the container objects made.

In the constructor, the array list of bookings is created and, with a for loop, the numbers of all suites and rooms are being set.



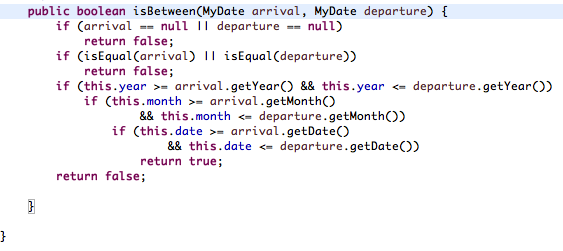
Here we also have the remove line method, where by using a while loop we copy all the lines from one text file, except the one we don’t need, to another.

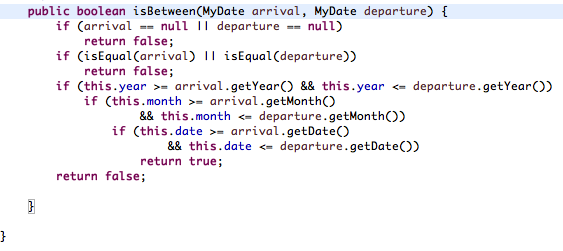


Another important method is the roomAvailability method, which checks the available rooms by the type of bed preferred by the guest and the desired stay period. In the end if a room is found the method returns the room number and if not it returns “-1”.



In the CheckIn class the totalPrice method is created, it calculates the price for the rooms, based on the stay period. Here for calculating the exact number of days the guests stayed, joda time library is used because, in our opinions it is more accurate and easier to use than the Date and Calendar classes of Java.





In the MyDate class we also have two important methods, which are called isEqual and isBetween. The isBetween method is used in the roomAvailability method, in order to check if a room is unoccupied for the preferred time period. In the isEqual we compare the date we already have with the argument of the method. The method called isBetween checks if the date we already have is between the date which we pass to the method.

1. **Test**

**6.1 Make a booking**

Making a booking is one of the most important options in the The Deer Alley Hotel System, it allows the user to register new bookings for a certain time period. It not only registers the arrival and departure dates, but also room or suite type and the guest’s name.

**Testing procedure:**

The way we test our program is by the GUI. When the user enters the program they choose the “Make a booking” button and a new frame opens up. Here the user is able to input the desired time period of the stay and preferred room/suite type. The user then presses the button “Check availability” and the program displays the first suitable room, if there is any. In the end the user types the name of the guest in the “Customer details” panel, adds extra bed if needed and saves the booking.

**Test results:**

The test was successful and a new booking was created.

**Errors:**

The system is not able to make more than one booking for the same person at once.

**6.2 Edit a booking**

Edit a booking is the second option of the system. Here the system allows the user to make changes in an already existing booking. The changes that they may make are: change arrival and departure date, room/suite type and add/remove extra bed or change guest’s name.

**Testing procedure:**

For this option the user chooses the “Edit a booking” option. A new window appears where the user searches for an already existing booking, by inputting the name of the guest that made the booking. When the system finds the booking, it allows them to change the information mentioned above.

**Test results:**

The test was successfully performed.

**Errors:**

In the “Edit a booking” option we are not conquering any errors.

**6.3 Cancel a booking**

Cancel a booking is the third option of the system. It allows the user to cancel an existing booking.

**Testing procedure:**

The user chooses “Cancel a booking” button. The user searches for the existing booking by inputting the name of the person that requested the booking. When the booking is found, the user presses the “Cancel” button and the system removes it.

**Test results:**

The test was successfully performed and the old booking is erased from the system.

**Errors:**

If the user made more than one booking for the same person, they should be deleted manually, because we have no option to delete more than one booking at once.

**6.4 Making a check-in**

Check-in is the fourth option in the system. This option allows the user to register guests to the system.

**Testing procedure:**

From the main frame if the user chooses the “Check-in” button, the frame for searching a booking appears. If the booking is found, a window shows up and displays the information about it. After pressing the “Add guest” button, the user is able to write down additional information as: phone number, email, home address, birth date, passport number, nationality and name. This procedure can be repeated if necessary.

**Test results:**

The check-in procedure is working as planned and the information is registered successfully. The booking is erased from the system once the “Check-in” button is pressed.

**Errors:**

This option operates without any errors.

**6.5 Making a check-out**

Check-out: the fifth and last option from our system. It allows the user to remove the guests’ information from the system and calculate the price for the whole stay.

**Testing procedure:**

The user clicks on the “Check-out” button. A search for check-in frame appears, here the user inputs the name of the guest that they want to check-out. After the guest is found, the system displays the information about the stay of that particular guest: name, arrival and departure dates, room number and price. The user can edit the price - if there are any used services. Adding services is done by pressing the “services” button where the user can choose from several options as: laundry, minibar or service room. After the user made the choice they press the “Add” button and the total price is updated automatically. In the end the user clicks “See info” button and a new window with the whole information is displayed. By pressing the “check-out” button the guest is removed from the system.

**Test results:**

This option does not return any significant errors.

**Errors:**

The system is not able to check out more than one person at once, meaning that all the guests should be removed manually. Also the system removes only the name of the person, not all the other information from the check-in.

1. Results

The Deer Alley Hotel Booking system was developed to facilitate the work of the hotel. Its purpose is to make booking, edit a booking or cancel one, add all guests to the hotel, check them out and print the price.

The problems that we are conquering are:

1. The system not being able to book more than one room/suite at once (in the Make a Booking option)

2. Removing all of the guests from a room at once (Check-out option)

Solutions for the problems:

1. The user can make 2 or more different booking for the same person.

2. Remove all the guests manually.

The GUI for the system was designed to be user friendly and well-organized as well as allowing the user to choose from all the functions the customer wanted.

1. Discussion

As the project started, we looked carefully at the requirements from the customer, we tried to follow them as accurate as possible. At the beginning we had more classes, that we removed as the project was pursuing, because we came up with better options.

We implemented some classes so the system works best and does justice to what it needs to be done. Throughout the project we came up with more ideas, like making a class for the GUI(FunctionalityList), which we call in the other classes when we need a method from it. We decided to do it this way, because it will look polished and more professional in our opinion, and it is easier to understand the purpose of it’s existence. We had some problems with removing the guest information from the file at the check-out, but we managed to make it happen by doing more research and thinking what would be best for the system. The extra-bed request was a struggle for us at the beginning, because we didn’t know when exactly it should be added, however we managed to add it when a booking is made and so it adds the price at the check-out. By making this changes we successfully managed to fulfil the wishes of the customer. Even though we encountered some problems, together as a team we handled them the best we could so the system implements all the conditions.

1. Conclusion

The Deer Alley Hotel Booking system was developed to facilitate the work of the hotel. Its purpose is to make booking, edit a booking or cancel one, add guests, check them out and print the price. This system has the function to ease the work of the employees of the hotel, by providing them with better equipment than just a notebook and a pen, where they can easily commit mistakes or lose information. As mentioned earlier, all the requests from the owner of the hotel were fully implemented in this booking system. It was essential that the system is doing all the tasks indicated so the hotel is not losing any guests and information about them. It was important for us to test the system several times, so we would not miss any errors that may appear during the execution. The testing was done very carefully and for every step, so we would not leave any leaks in the system. By doing this we were ensured that the system is working properly and we are not losing more time on finding the error at the end in a long code. In conclusion, we can state that the system is working accordingly to the requests and is user friendly

1. **Appendices**

Appendix A – Use Case Description

Appendix B – Activity Diagrams

Appendix C – Class Diagrams  
Appendix D – User Guide