Appendices

Table of contents

1. User guide to the system…………………….……………………………………………2

1.1 Main window…...…………………………………………………………...2

1.1.1 Search bar……………………………………………………………3
1.1.2 Buttons…....………………………………………………………....4
1.1.3 Organizer…………………………………………………………….5
1.1.4 Information box…………………………………………………...6

1.2 Booking……………………………………………………………………..7

1.3 Edit……………………………………………………..…………….…..8

1.4 Check-in ……………………………………………………………………9

1.5 Check-out…………………………………………..…………....10

2. Selected Java code…………………………………………………………….11

3. Analysis documentation…………………………………..………………..12

3.1 Use Case diagram…………………………………..…………………12

3.2 Use Case descriptions…………………………………....………………13

3.3 Activity diagrams…………………………………………………………..17

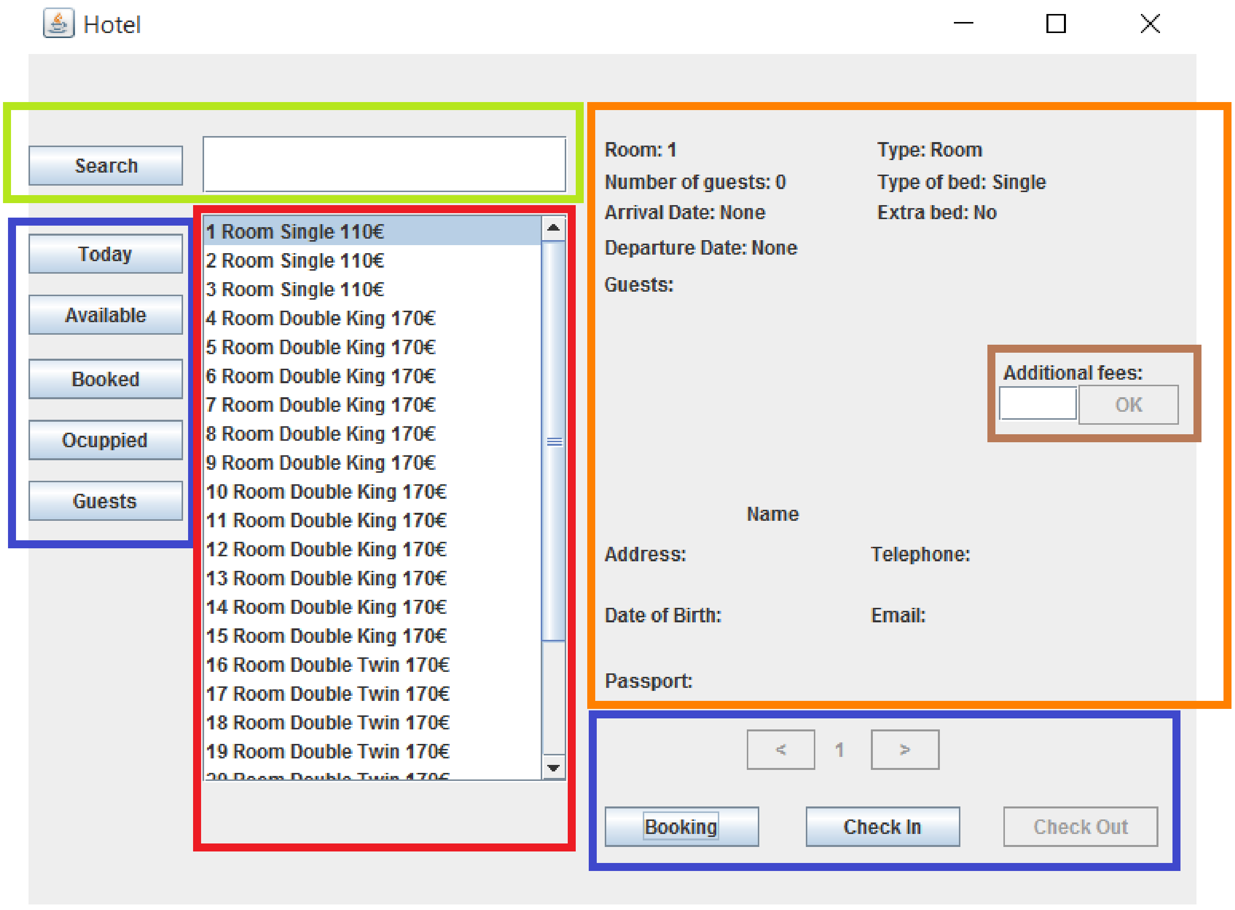
4. Design documentation…………………………………..………………….25

4.1 Sequence diagram……………………………………..…………….25

4.2 UML class diagram…………………………………...….…………….26

1. User guide to the system

1.1 Main window



1 Room Single 110€
2 Room Single 110€
3 Room Single 110€
4 Room Double King 170€
5 Room Double King 170€
6 Room Double King 170€
7 Room Double King 170€
8 Room Double King 170€
9 Room Double King 170€
10 Room Double King 170€
11 Room Double King 170€
12 Room Double King 170€
13 Room Double King 170€
14 Room Double King 170€
15 Room Double King 170€
16 Room Double Twin 170€
17 Room Double Twin 170€
18 Room Double Twin 170€
19 Room Double Twin 170€

Room: 1 Type: Room
Number of guests: 0 Type of bed: Single
Arrival Date: None Extra bed: No
Departure Date: None
Guests:

Name
Address: Telephone:
Date of Birth: Email:
Passport:

The main window is the first thing you see when you run the program.
Almost every action is executed from this window.

Green- Search box. (More info on page 3).

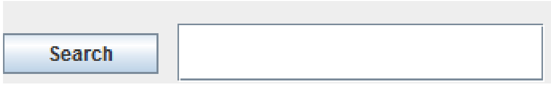
Blue- Action buttons. (More info on page 4).

Red- Organizer. (More info on page 5).

Orange- Information box. (More info on page 6).

Brown- Additional fees. (More info on page 6).

1.1.1 Search bar



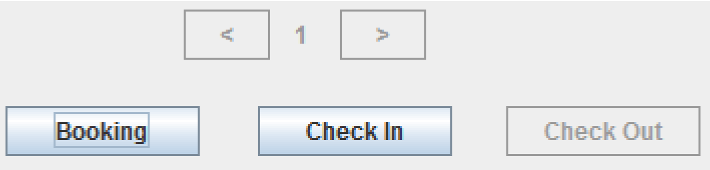
You can search for a room simply by entering a room number, name of a guest or a date into the text field and then pressing the “Search” button.
The result will be shown in the Organizer under this Search bar. If there are no results, you will be informed with an information message.

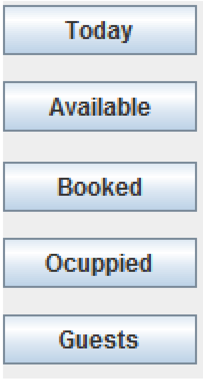
1. You can search by room number. For example: “12”, the result will be only one room with number “12”.

2. You can search by name. For example: “John” or “John Smith”, the result will be all guests with this name and also all bookings on this name.

3. You can search for a room by date. For example: “15/11/2015-22/11/2015”, the result will be all rooms that are free for the duration of these dates.

1.1.2 Buttons





“Today” button – Show a list of all of today’s Check in and Check outs.

“Available” button – Show a list of all available rooms for today.

“Booked” button – Show a list of all bookings.

“Occupied” button – Show a list of all occupied rooms.

“Guest” button – Show a list of all guests.

“Booking” button – Opens a new window for booking. In the organizer, room has to be selected. (More info on page 9).

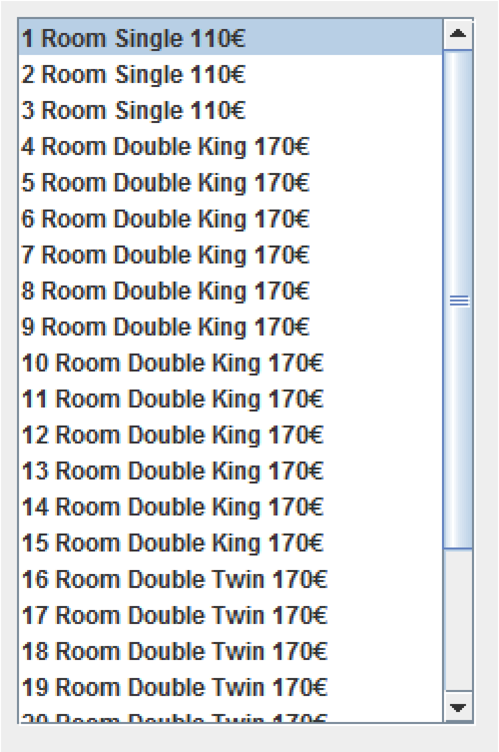
“Check In” button – Opens a new window for Check In. Available only if an available room or a booking with today’s arrival is selected. (More info on page 11).

“Check Out” button – Opens a new window for Check out. Available only if an occupied room is selected. (More info on page 12).

“Edit” button – Opens a new window for Edit. Available only if a booking is selected. (More info on page 10).

“<” And “>” buttons – Available only if a room contains more than 1 guest. You can switch between guests and their information using these arrows.

1.1.3 Organizer

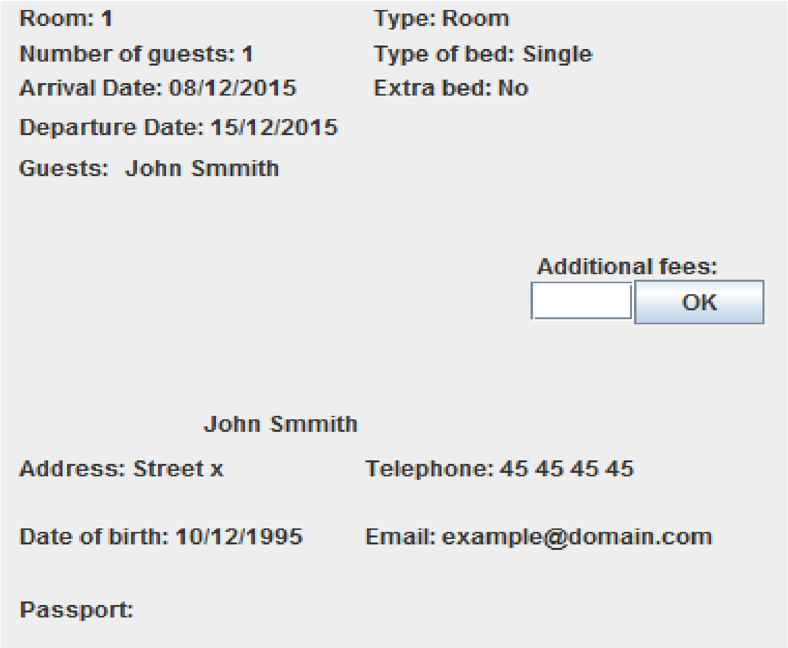


1 Room Single 110€
2 Room Single 110€
3 Room Single 110€
4 Room Double King 170€
5 Room Double King 170€
6 Room Double King 170€
7 Room Double King 170€
8 Room Double King 170€
9 Room Double King 170€
10 Room Double King 170€
11 Room Double King 170€
12 Room Double King 170€
13 Room Double King 170€
14 Room Double King 170€
15 Room Double King 170€
16 Room Double Twin 170€
17 Room Double Twin 170€
18 Room Double Twin 170€
19 Room Double Twin 170€
20 Room Double Twin 170€

The Organizer is a list of results from the search bar and the action buttons. This organizer can show lists of guests, bookings, rooms and more.

You can select a room, a booking or a guest from this list and all the information connected to this room or guest is shown in the information box (on the right side).

1.1.4 Information box

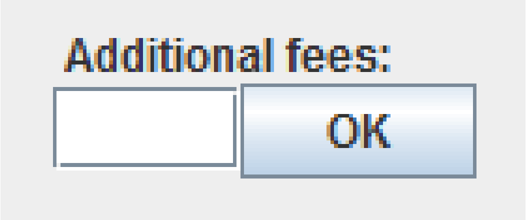

Room: 1
Type: Room
Number of guests: 1
Type of bed: Single
Arrival Date: 08/12/2015
Extra bed: No
Departure Date: 15/12/2015
Guests: John Smmith

John Smmith
Address: Street x
Telephone: 45 45 45 45
Date of birth: 10/12/1995
Email: example@domain.com
Passport:

In the information box you can see all the information about the rooms and the guests.
Each room has a room number, a type, the guest's names and arrival and departure dates.

Then there is the "Additional fees" bar. More information below.

And the Guest's information where you can find the name, the contact information and the address.



If the guests use additional services such as the minibar, you can add a price for that service and that price will be added to the Check Out total price.

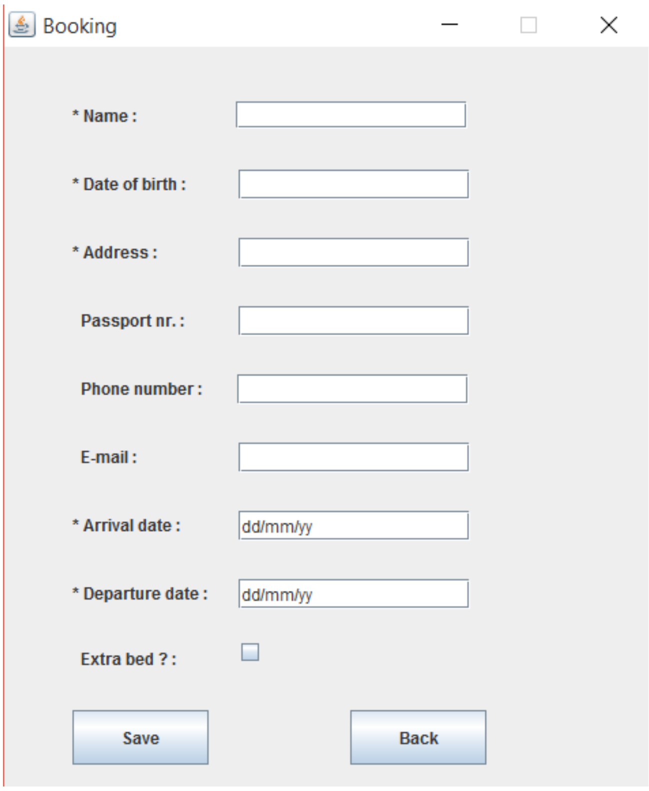
If you write for example: "50", 50 Euros will be added.

Also you can write: "-50", 50 Euros will be removed from the total price. (Discounts)

1.2 Booking

If you press the “Booking” button, a new window will appear.
There you need to put information about the booker.
Fields marked with * mark are necessary, the others are optional.

Dates have to be in format ”dd/mm/yyyy”.


Booking

* Name :

* Date of birth :

* Address :

Passport nr. :

Phone number :

E-mail :

* Arrival date : dd/mm/yy

* Departure date : dd/mm/yy

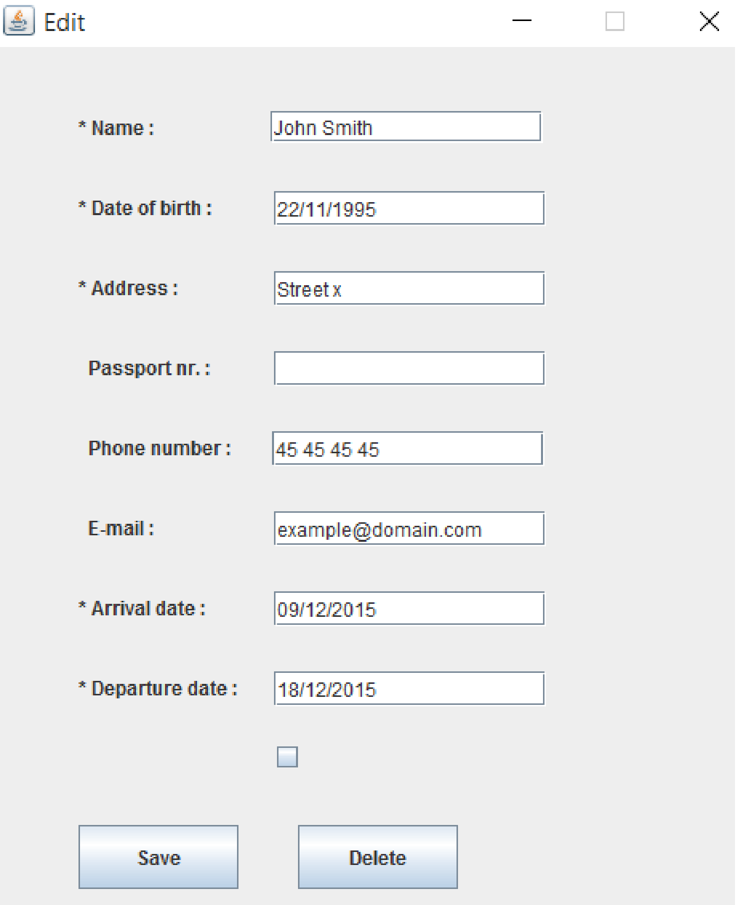
Extra bed ? : ☐

With the button “Save”, you will save this booking in the database.
With the button “Back”, you will return to the main window and nothing is saved.

If guest does not show up until 18:00 of the arrival day, the booking will be removed.

1.3 Edit

If you have booked room and you want to edit or delete it, you can select it from booked rooms list and click the button “Edit”.
A new window will appear:

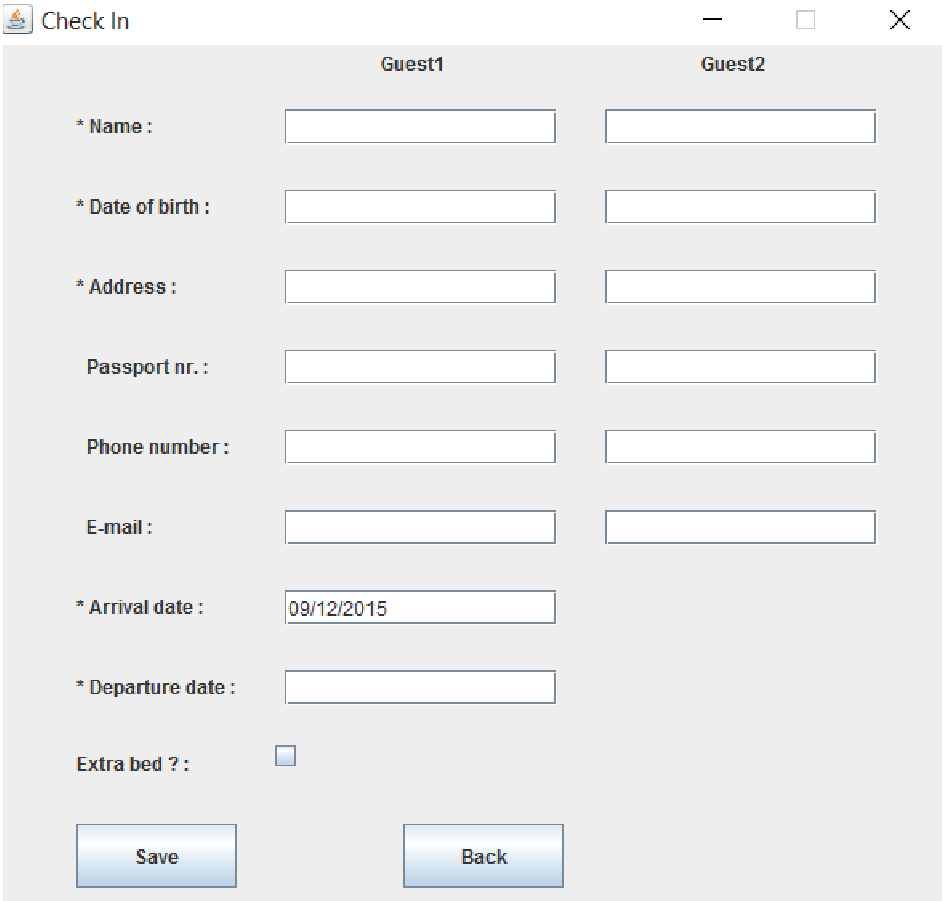


* Name :	John Smith
* Date of birth :	22/11/1995
* Address :	Street x
Passport nr. :	
Phone number :	45 45 45 45
E-mail :	example@domain.com
* Arrival date :	09/12/2015
* Departure date :	18/12/2015
	□

Text fields are already filled with “Old” information. You can change them and click on the button “Save” to save this booking with the new information or you can press “Delete” to delete this booking.

1.4 Check-in

If you want to Check In a booking or an available room, you can click on the button ”Check In” and a new window will appear:



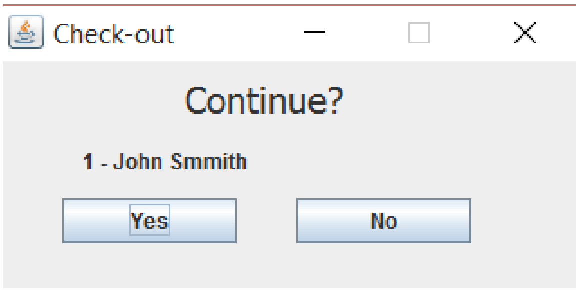
Check In

	Guest1	Guest2
* Name :		
* Date of birth :		
* Address :		
Passport nr. :		
Phone number :		
E-mail :		
* Arrival date :	09/12/2015	
* Departure date :		
Extra bed ? :	☐	

If you have selected an available room (Not booked), all fields except the arrival date will be blank. Else, if you have selected booking, Guest1 will be filled automatically with the information from booking.
Arrival date is also filled out automatically.
When you click “Save”, the room will be checked and booking will be removed.
When you click “Back”, you will return to the Main window and the room will still be only in a booked state.

1.5 Check-out

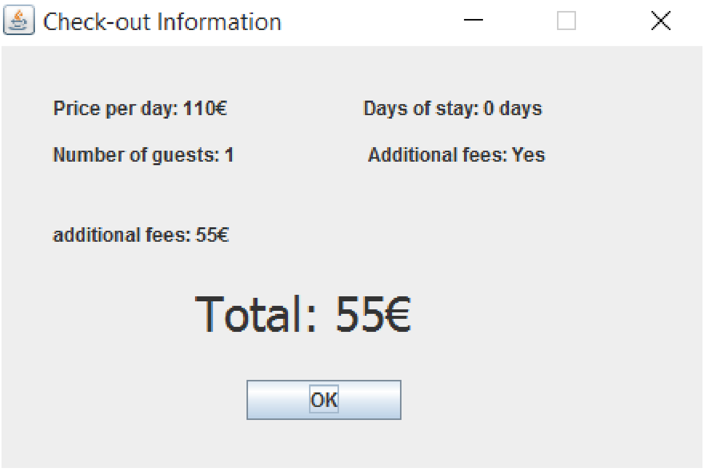
When you want to Check Out guests you can click “Check Out” button. A window with confirmation will appear:



Continue?

1 - John Smmith

When you click on “No” you will be returned to the Main menu. Else if you click on “yes”, there will be a new window with information about the duration of stay and the total price:



Price per day: 110€

Days of stay: 0 days

Number of guests: 1

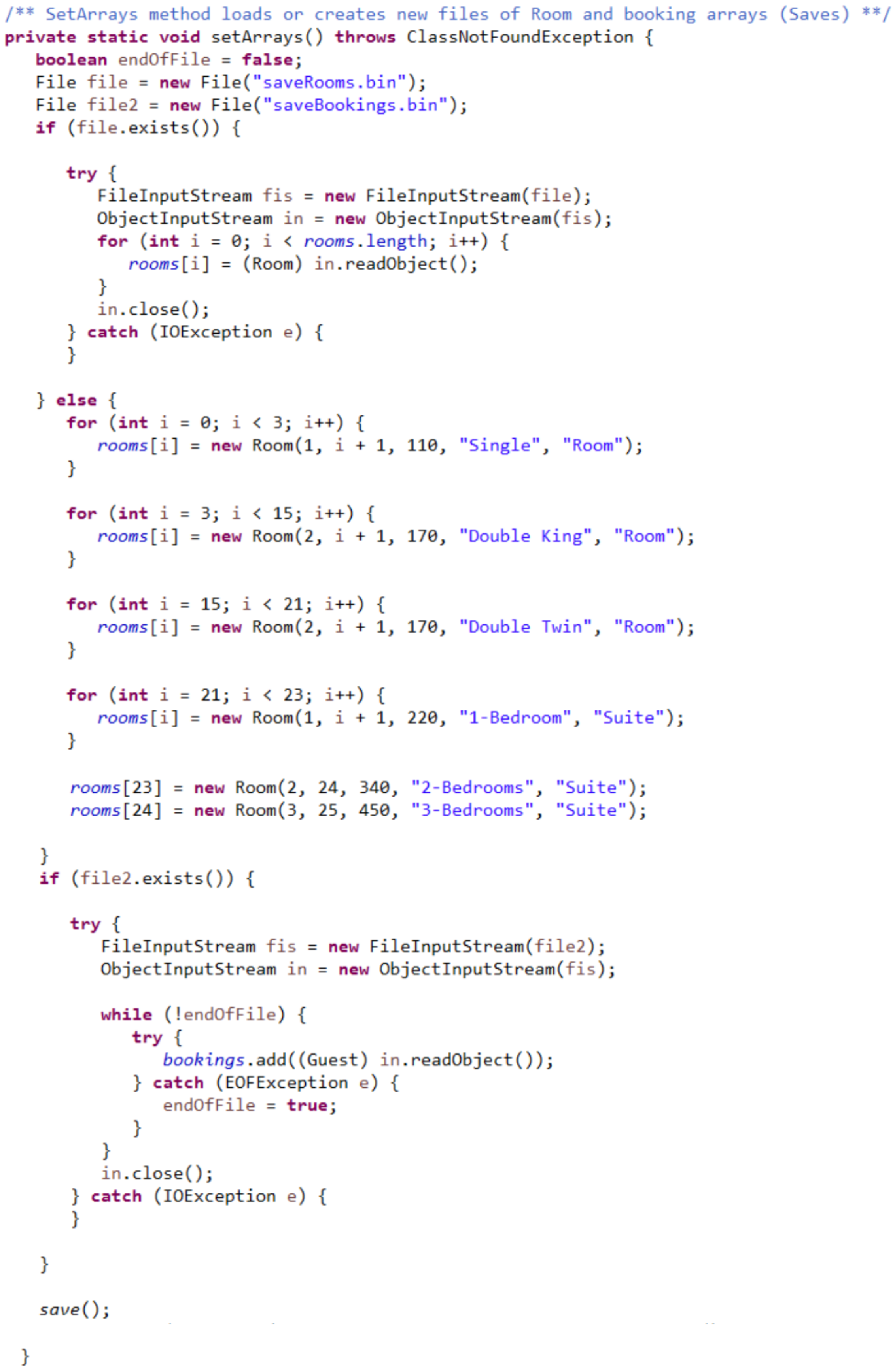
Additional fees: Yes

additional fees: 55€

Total: 55€

After you click OK, the room will be empty and you will return to the Main window.

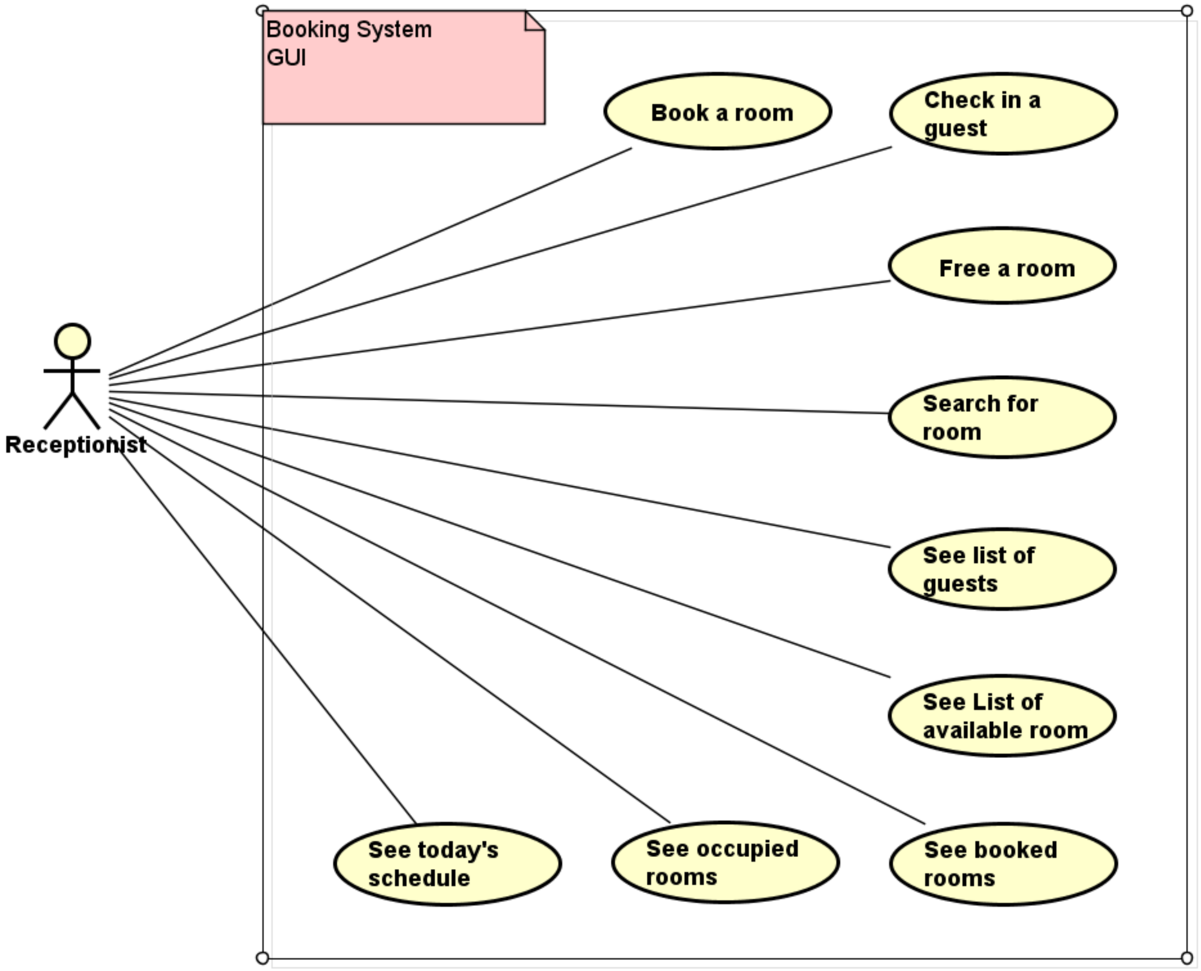
2. Selected Java code



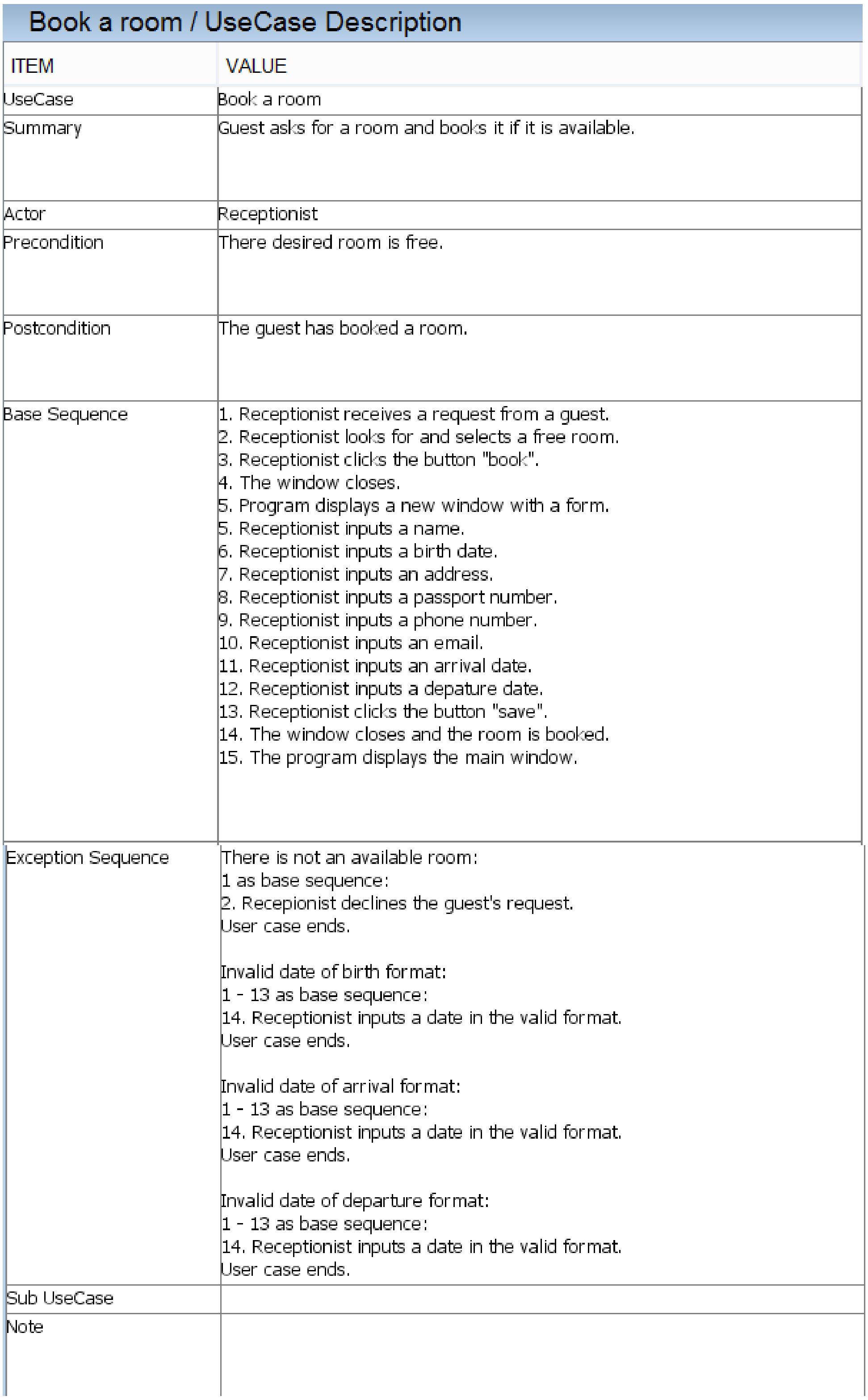
```java
/** SetArrays method loads or creates new files of Room and booking arrays (Saves) **/
private static void setArrays() throws ClassNotFoundException {
   boolean endOfFile = false;
   File file = new File("saveRooms.bin");
   File file2 = new File("saveBookings.bin");
   if (file.exists()) {

      try {
         FileInputStream fis = new FileInputStream(file);
         ObjectInputStream in = new ObjectInputStream(fis);
         for (int i = 0; i < rooms.length; i++) {
            rooms[i] = (Room) in.readObject();
         }
         in.close();
      } catch (IOException e) {
      }

   } else {
      for (int i = 0; i < 3; i++) {
         rooms[i] = new Room(1, i + 1, 110, "Single", "Room");
      }

      for (int i = 3; i < 15; i++) {
         rooms[i] = new Room(2, i + 1, 170, "Double King", "Room");
      }

      for (int i = 15; i < 21; i++) {
         rooms[i] = new Room(2, i + 1, 170, "Double Twin", "Room");
      }

      for (int i = 21; i < 23; i++) {
         rooms[i] = new Room(1, i + 1, 220, "1-Bedroom", "Suite");
      }

      rooms[23] = new Room(2, 24, 340, "2-Bedrooms", "Suite");
      rooms[24] = new Room(3, 25, 450, "3-Bedrooms", "Suite");

   }
   if (file2.exists()) {

      try {
         FileInputStream fis = new FileInputStream(file2);
         ObjectInputStream in = new ObjectInputStream(fis);

         while (!endOfFile) {
            try {
               bookings.add((Guest) in.readObject());
            } catch (EOFException e) {
               endOfFile = true;
            }
         }
         in.close();
      } catch (IOException e) {
      }

   }

   save();

}
```

3. Analysis documentation

3.1 Use Case diagram

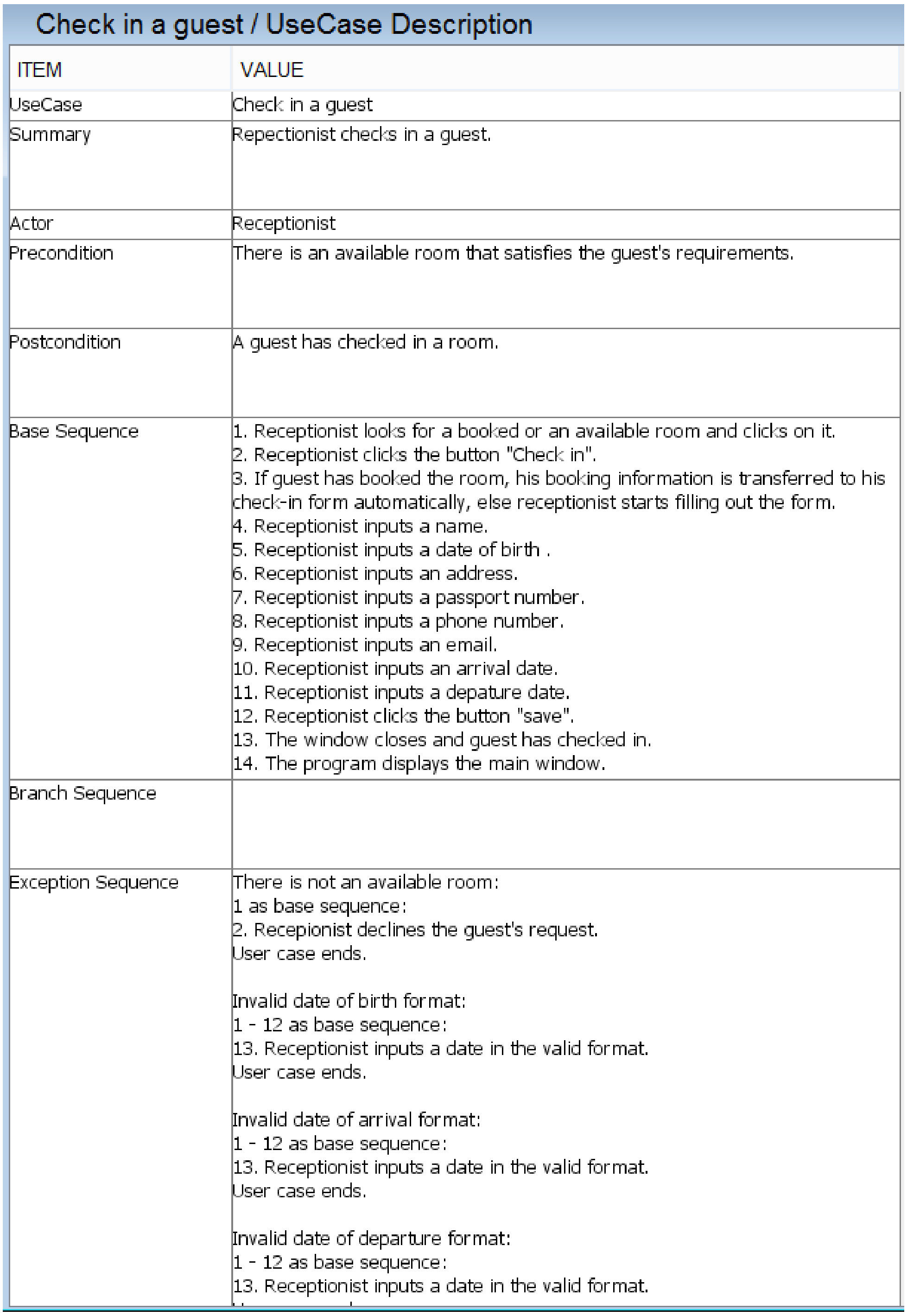


3.2 Use Case descriptions



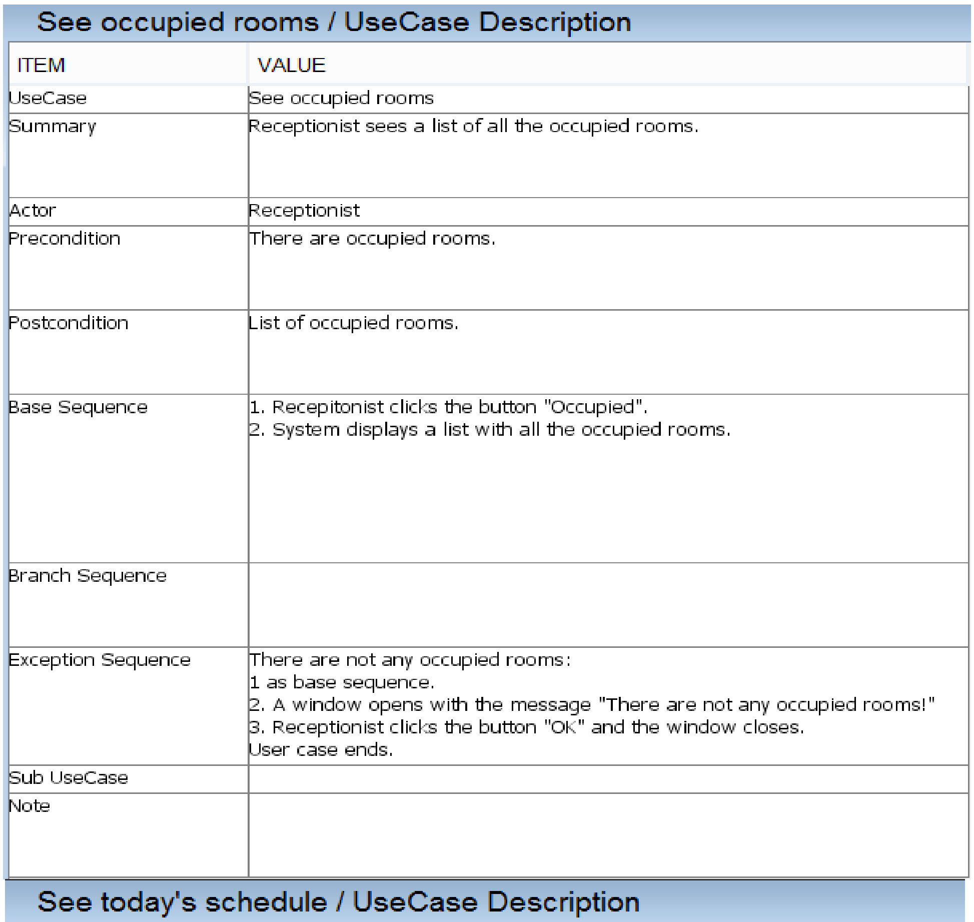
Book a room / UseCase Description

ITEM	VALUE
UseCase	Book a room
Summary	Guest asks for a room and books it if it is available.
Actor	Receptionist
Precondition	There desired room is free.
Postcondition	The guest has booked a room.
Base Sequence	1. Receptionist receives a request from a guest. 2. Receptionist looks for and selects a free room. 3. Receptionist clicks the button "book". 4. The window closes. 5. Program displays a new window with a form. 5. Receptionist inputs a name. 6. Receptionist inputs a birth date. 7. Receptionist inputs an address. 8. Receptionist inputs a passport number. 9. Receptionist inputs a phone number. 10. Receptionist inputs an email. 11. Receptionist inputs an arrival date. 12. Receptionist inputs a depature date. 13. Receptionist clicks the button "save". 14. The window closes and the room is booked. 15. The program displays the main window.
Exception Sequence	There is not an available room: 1 as base sequence: 2. Recepionist declines the guest's request. User case ends. Invalid date of birth format: 1 - 13 as base sequence: 14. Receptionist inputs a date in the valid format. User case ends. Invalid date of arrival format: 1 - 13 as base sequence: 14. Receptionist inputs a date in the valid format. User case ends. Invalid date of departure format: 1 - 13 as base sequence: 14. Receptionist inputs a date in the valid format. User case ends.
Sub UseCase	
Note	



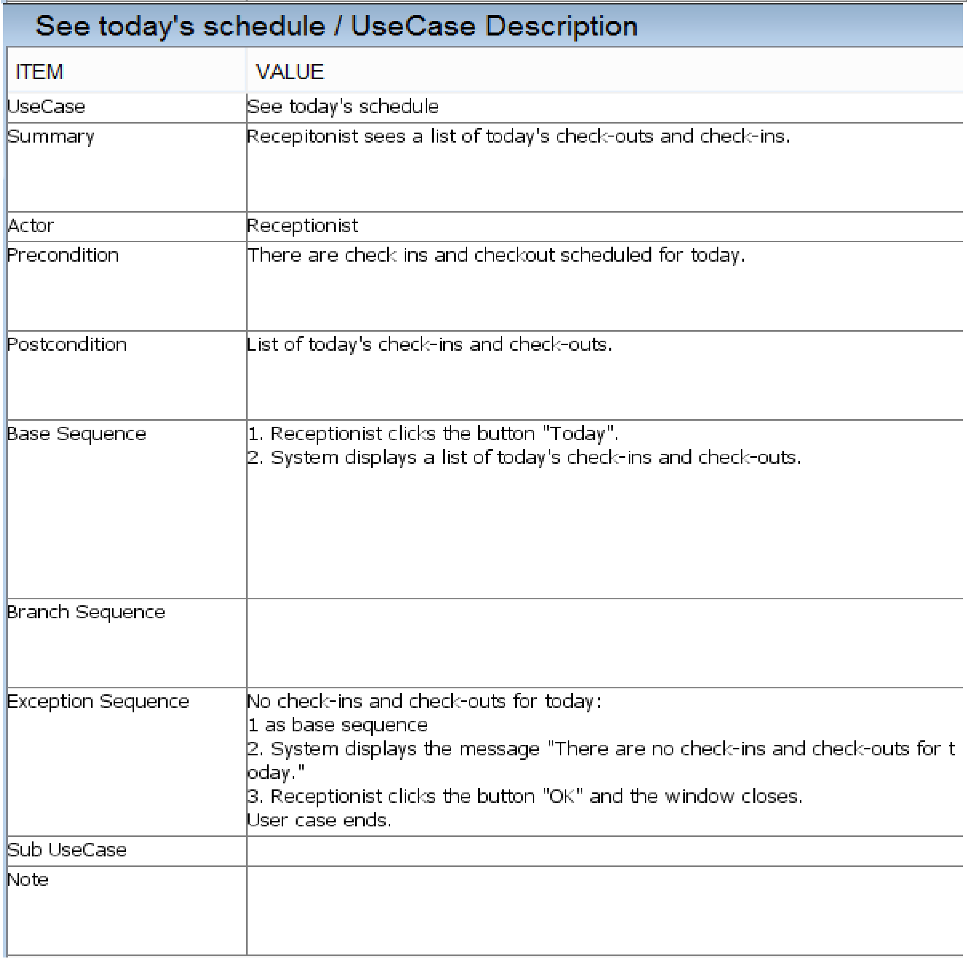
Check in a guest / UseCase Description

ITEM	VALUE
UseCase	Check in a guest
Summary	Repectionist checks in a guest.
Actor	Receptionist
Precondition	There is an available room that satisfies the guest's requirements.
Postcondition	A guest has checked in a room.
Base Sequence	1. Receptionist looks for a booked or an available room and clicks on it. 2. Receptionist clicks the button "Check in". 3. If guest has booked the room, his booking information is transferred to his check-in form automatically, else receptionist starts filling out the form. 4. Receptionist inputs a name. 5. Receptionist inputs a date of birth . 6. Receptionist inputs an address. 7. Receptionist inputs a passport number. 8. Receptionist inputs a phone number. 9. Receptionist inputs an email. 10. Receptionist inputs an arrival date. 11. Receptionist inputs a depature date. 12. Receptionist clicks the button "save". 13. The window closes and guest has checked in. 14. The program displays the main window.
Branch Sequence	
Exception Sequence	There is not an available room: 1 as base sequence: 2. Recepionist declines the guest's request. User case ends. Invalid date of birth format: 1 - 12 as base sequence: 13. Receptionist inputs a date in the valid format. User case ends. Invalid date of arrival format: 1 - 12 as base sequence: 13. Receptionist inputs a date in the valid format. User case ends. Invalid date of departure format: 1 - 12 as base sequence: 13. Receptionist inputs a date in the valid format.



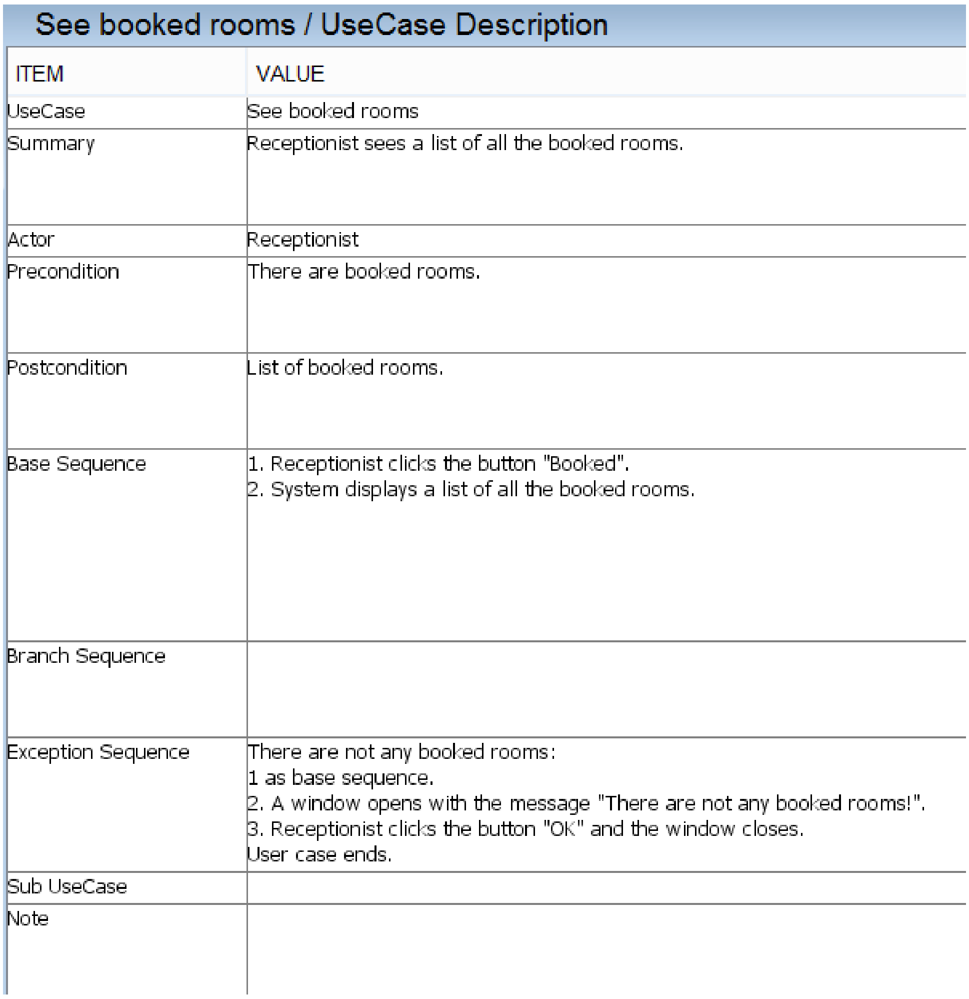
See occupied rooms / UseCase Description

ITEM	VALUE
UseCase	See occupied rooms
Summary	Receptionist sees a list of all the occupied rooms.
Actor	Receptionist
Precondition	There are occupied rooms.
Postcondition	List of occupied rooms.
Base Sequence	1. Recepitonist clicks the button "Occupied". 2. System displays a list with all the occupied rooms.
Branch Sequence	
Exception Sequence	There are not any occupied rooms: 1 as base sequence. 2. A window opens with the message "There are not any occupied rooms!" 3. Receptionist clicks the button "OK" and the window closes. User case ends.
Sub UseCase	
Note	



See today's schedule / UseCase Description

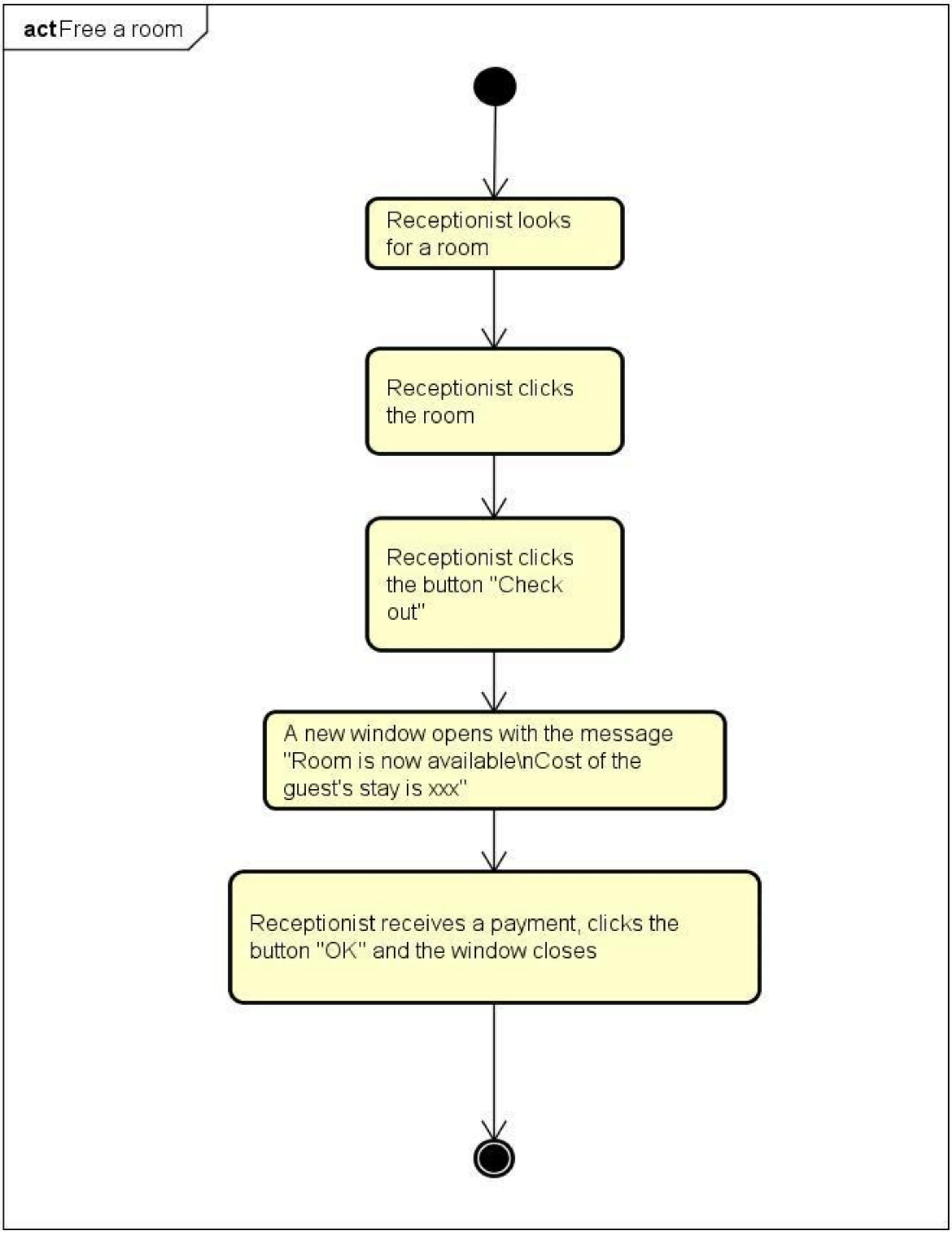
ITEM	VALUE
UseCase	See today's schedule
Summary	Recepitonist sees a list of today's check-outs and check-ins.
Actor	Receptionist
Precondition	There are check ins and checkout scheduled for today.
Postcondition	List of today's check-ins and check-outs.
Base Sequence	1. Receptionist clicks the button "Today". 2. System displays a list of today's check-ins and check-outs.
Branch Sequence	
Exception Sequence	No check-ins and check-outs for today: 1 as base sequence 2. System displays the message "There are no check-ins and check-outs for today." 3. Receptionist clicks the button "OK" and the window closes. User case ends.
Sub UseCase	
Note	

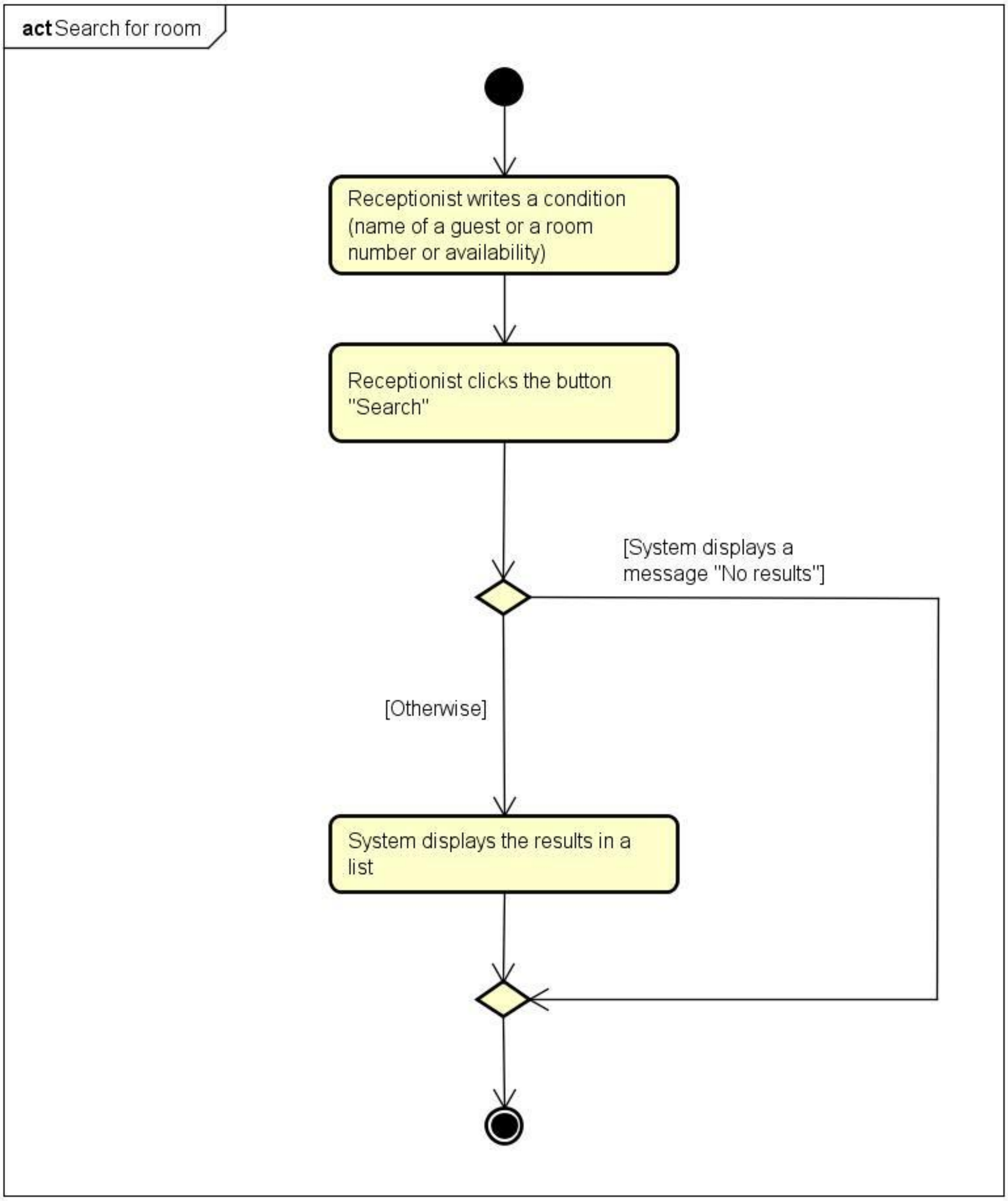


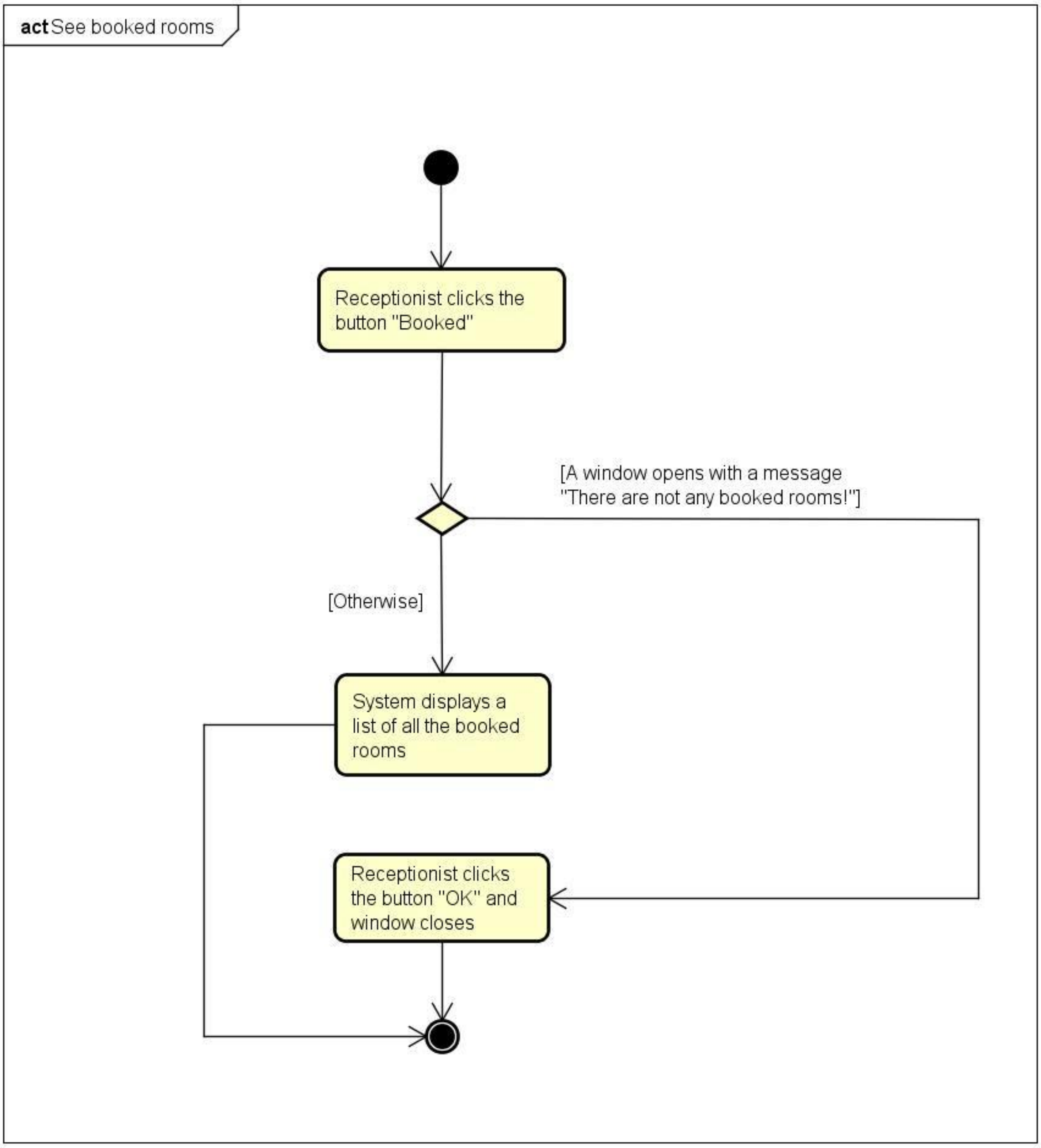
See booked rooms / UseCase Description

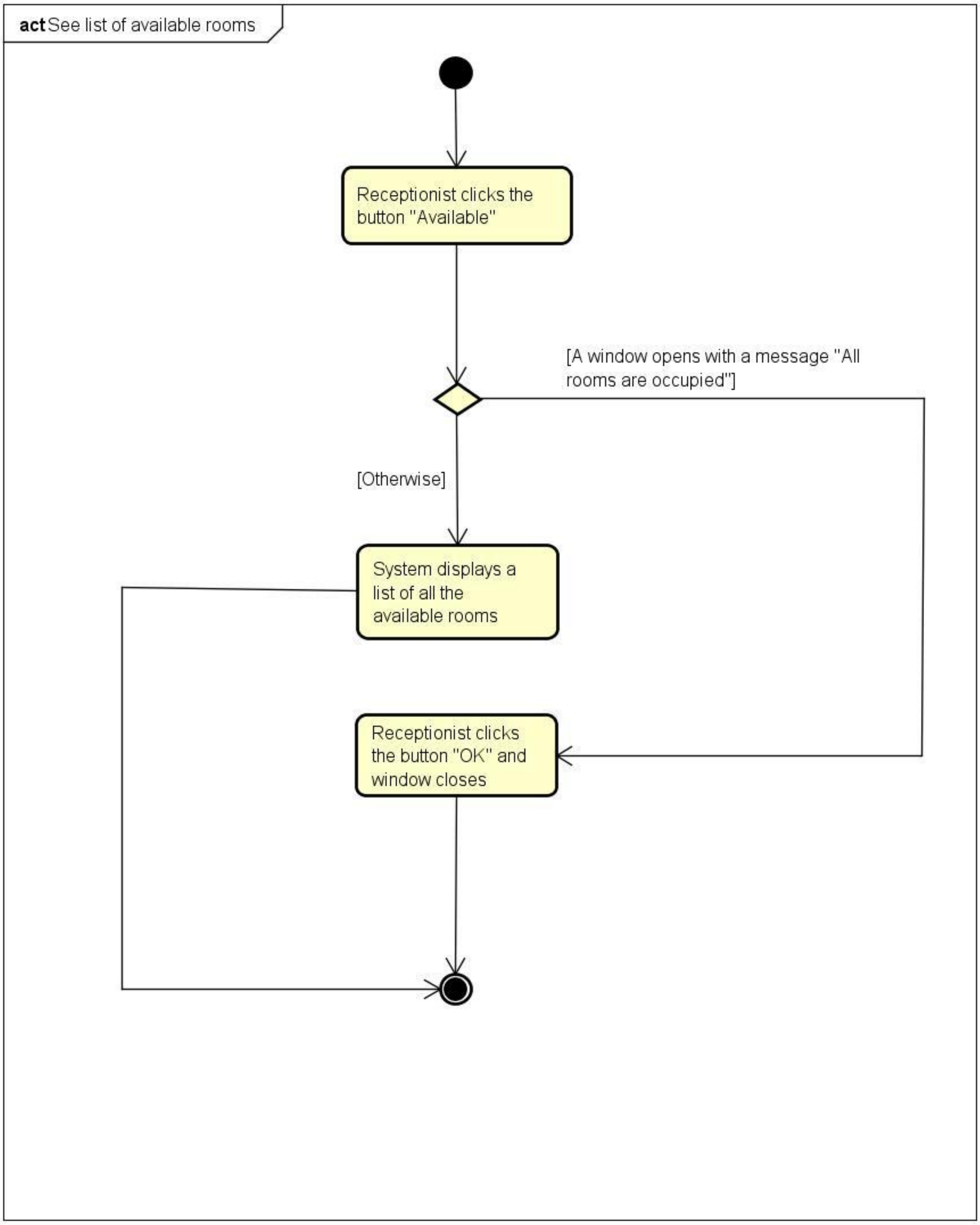
ITEM	VALUE
UseCase	See booked rooms
Summary	Receptionist sees a list of all the booked rooms.
Actor	Receptionist
Precondition	There are booked rooms.
Postcondition	List of booked rooms.
Base Sequence	1. Receptionist clicks the button "Booked". 2. System displays a list of all the booked rooms.
Branch Sequence	
Exception Sequence	There are not any booked rooms: 1 as base sequence. 2. A window opens with the message "There are not any booked rooms!". 3. Receptionist clicks the button "OK" and the window closes. User case ends.
Sub UseCase	
Note	

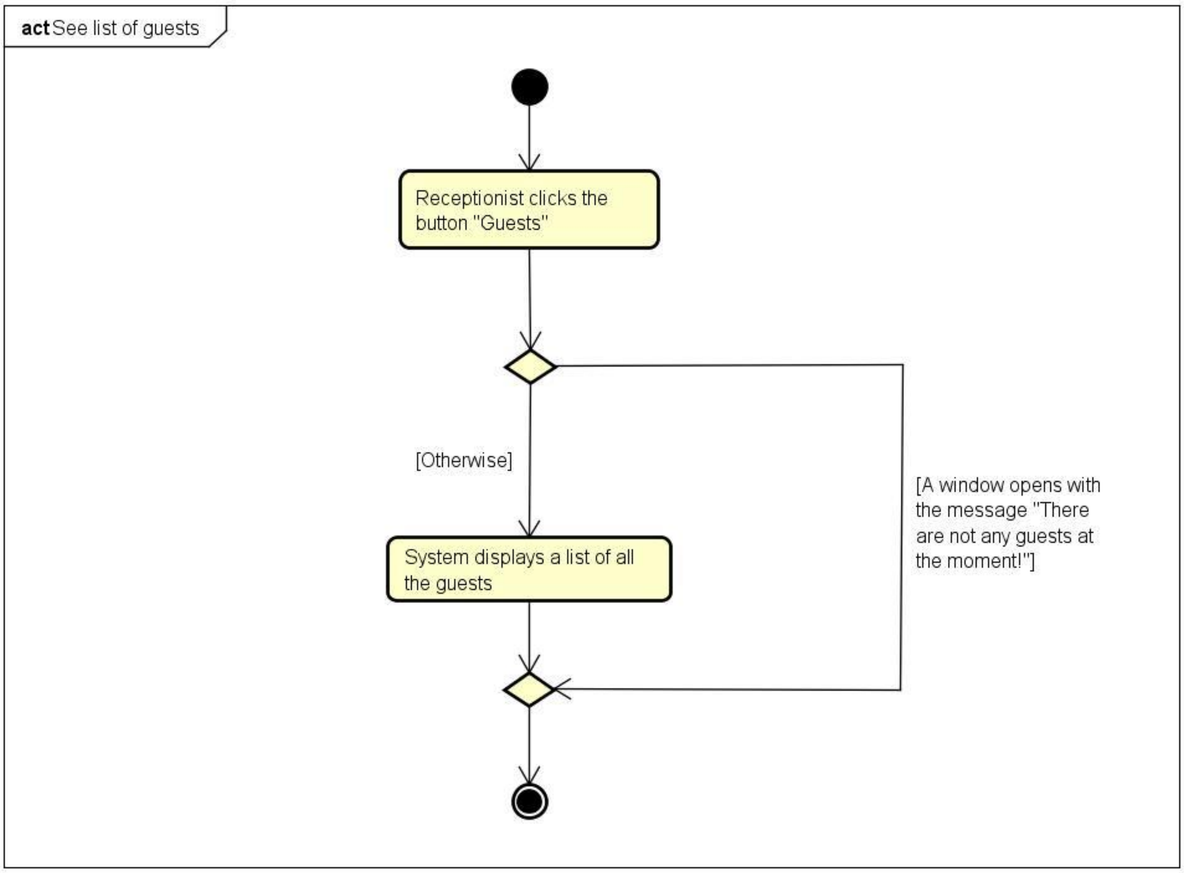
3.3 Activity diagrams

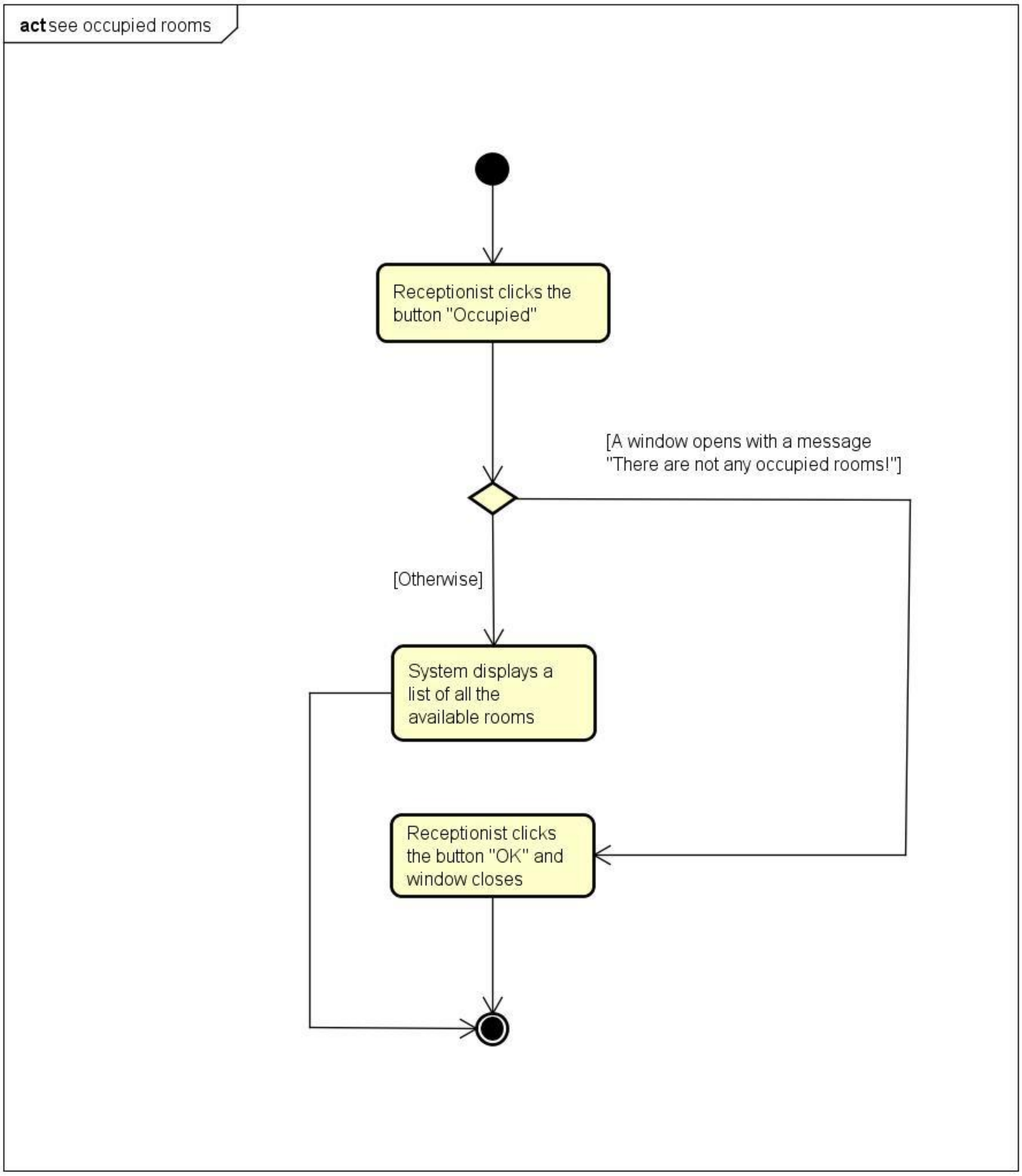

actFree a room
Receptionist looks for a room
Receptionist clicks the room
Receptionist clicks the button "Check out"
A new window opens with the message "Room is now available\nCost of the guest's stay is xxx"
Receptionist receives a payment, clicks the button "OK" and the window closes

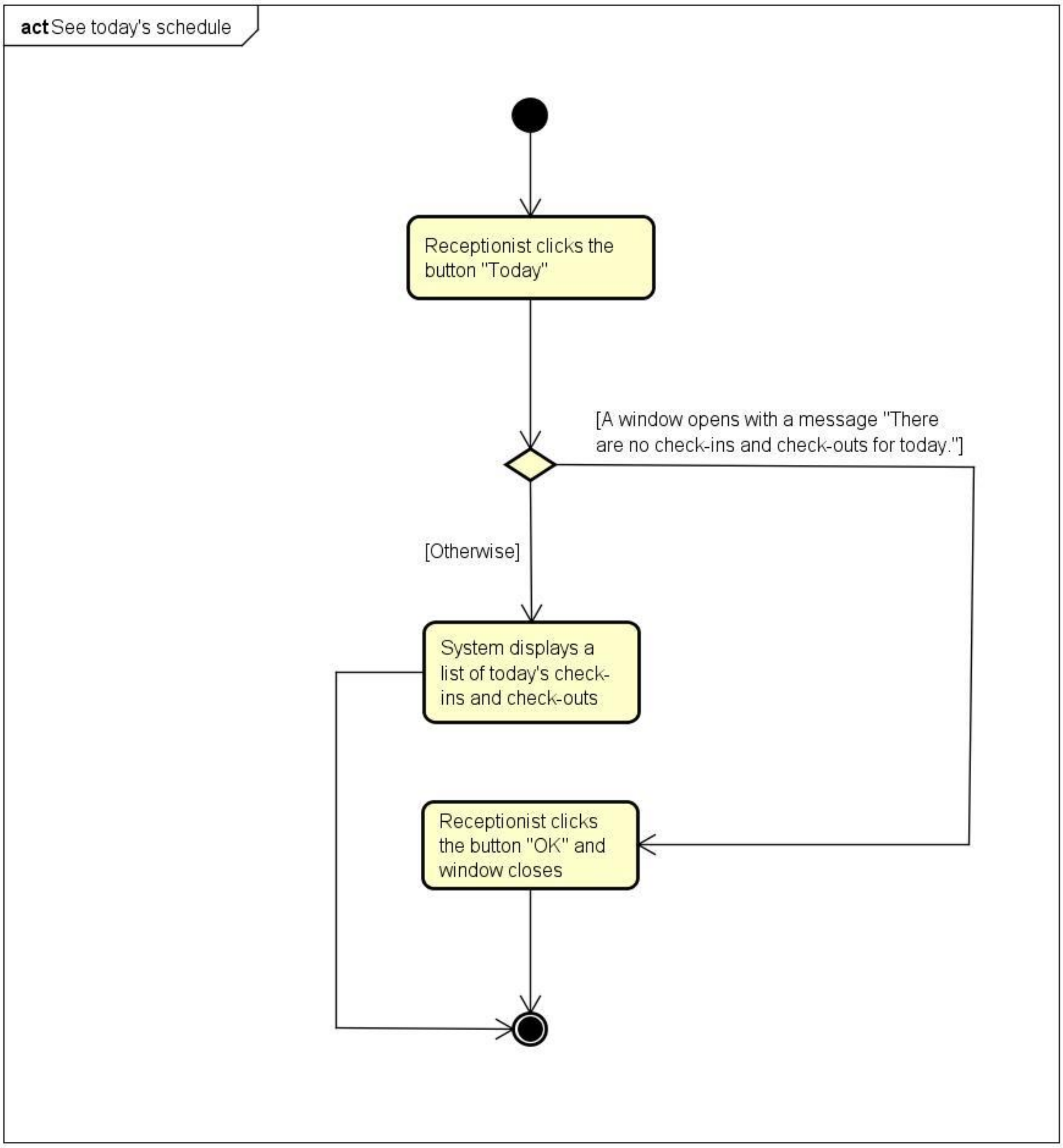

act Search for room
Receptionist writes a condition (name of a guest or a room number or availability)
Receptionist clicks the button "Search"
[System displays a message "No results"]
[Otherwise]
System displays the results in a list

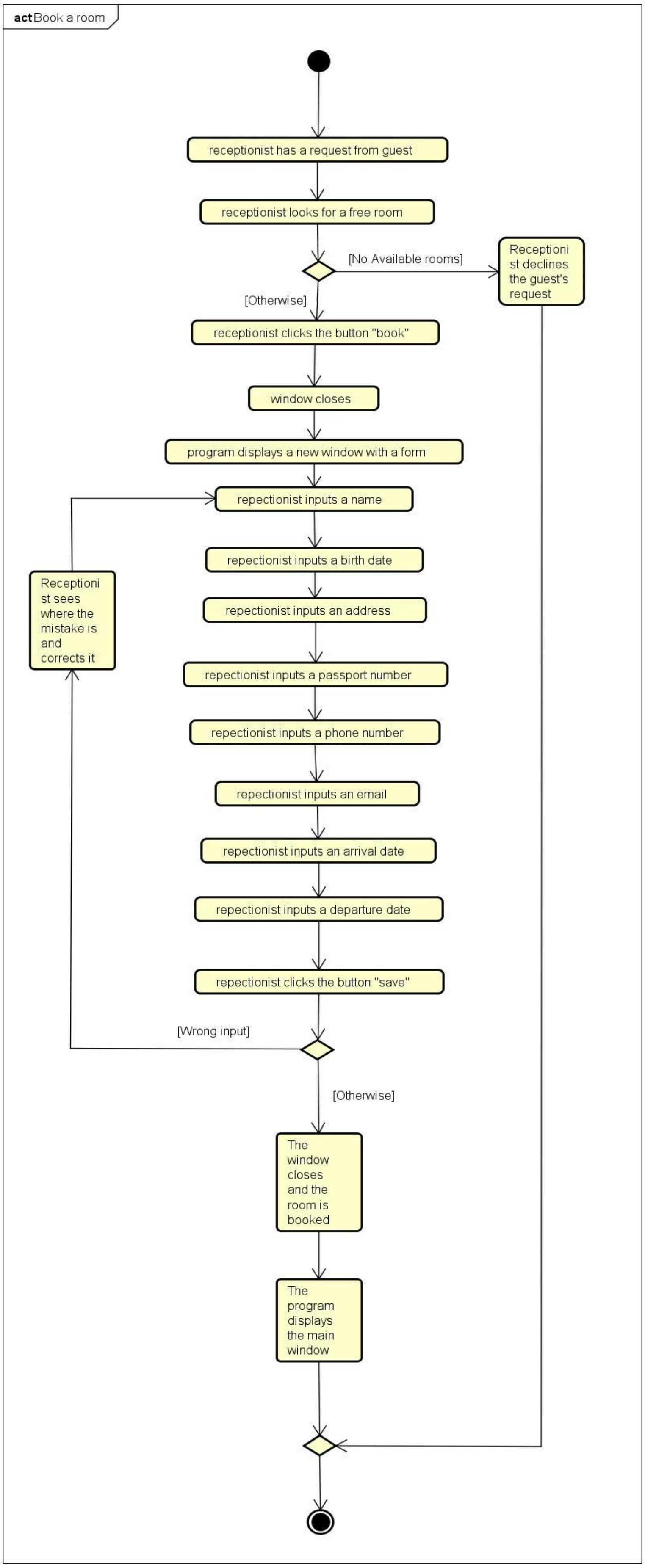

act See booked rooms
Receptionist clicks the button "Booked"
[A window opens with a message "There are not any booked rooms!"]
[Otherwise]
System displays a list of all the booked rooms
Receptionist clicks the button "OK" and window closes


act See list of available rooms
Receptionist clicks the button "Available"
[A window opens with a message "All rooms are occupied"]
[Otherwise]
System displays a list of all the available rooms
Receptionist clicks the button "OK" and window closes


act See list of guests
Receptionist clicks the button "Guests"
[Otherwise]
[A window opens with the message "There are not any guests at the moment!"]
System displays a list of all the guests

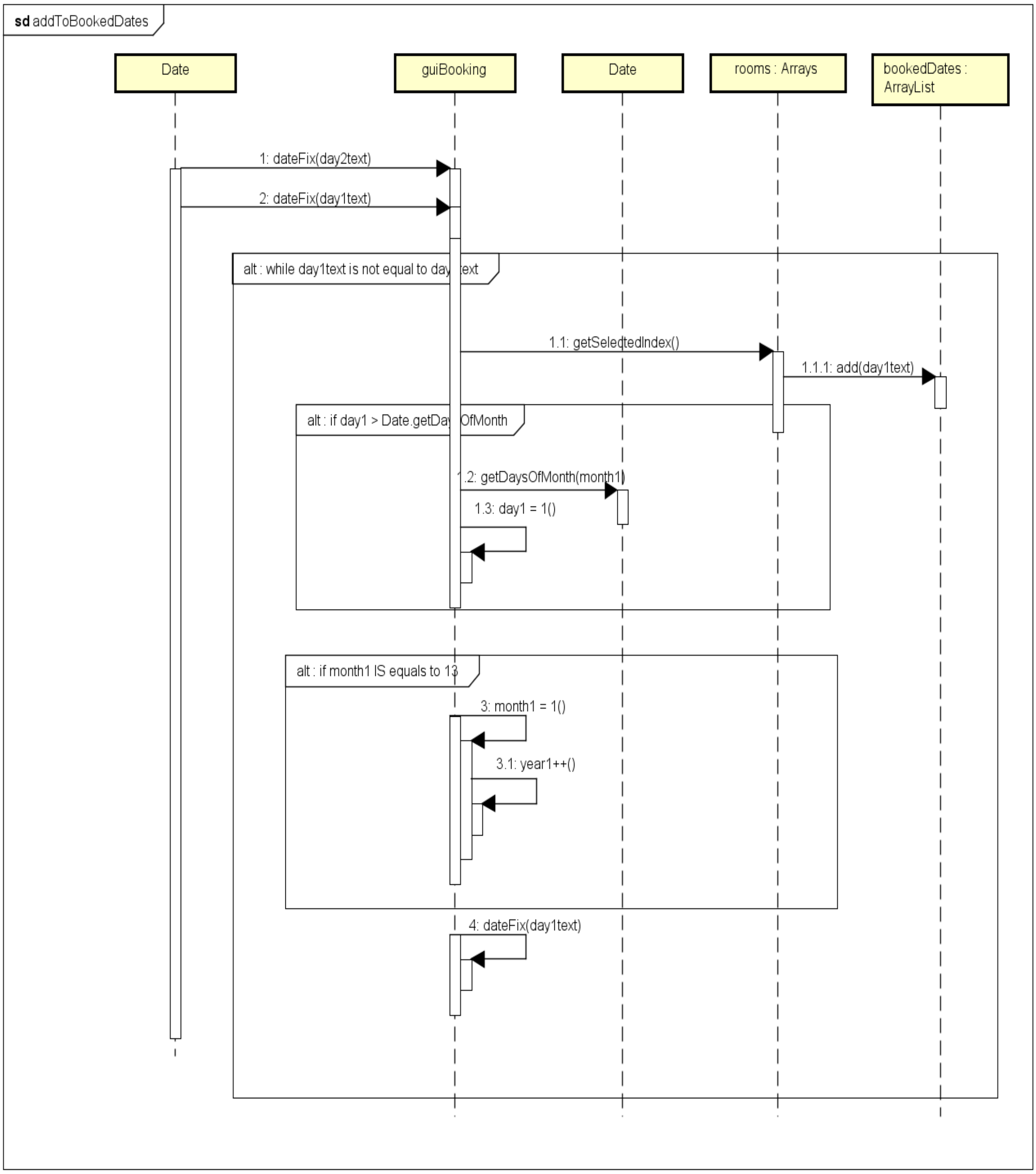

act see occupied rooms
Receptionist clicks the button "Occupied"
[A window opens with a message "There are not any occupied rooms!"]
[Otherwise]
System displays a list of all the available rooms
Receptionist clicks the button "OK" and window closes


act See today's schedule
Receptionist clicks the button "Today"
[A window opens with a message "There are no check-ins and check-outs for today."]
[Otherwise]
System displays a list of today's check-ins and check-outs
Receptionist clicks the button "OK" and window closes

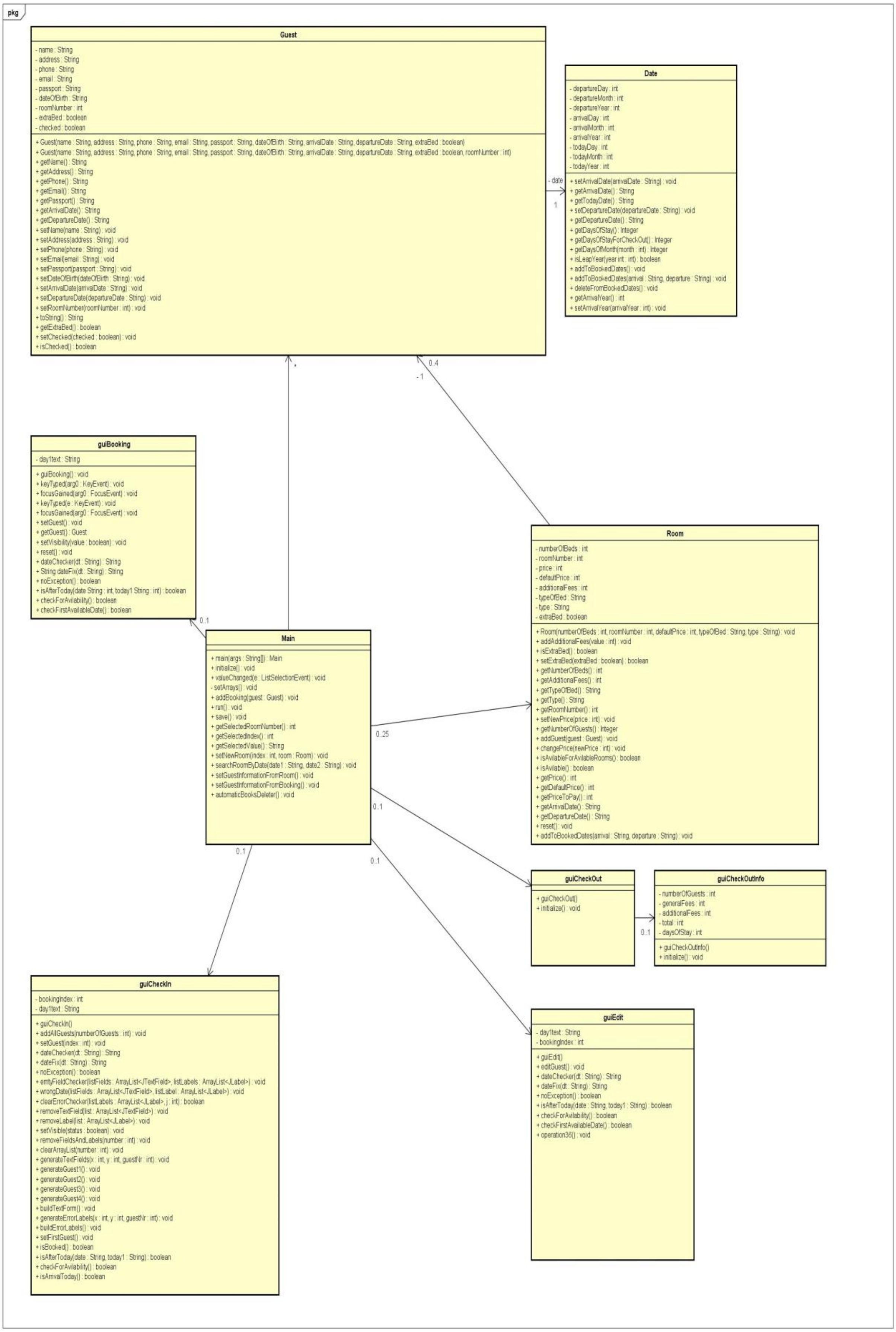

act Book a room
receptionist has a request from guest
receptionist looks for a free room
[No Available rooms]
Receptionist declines the guest's request
[Otherwise]
receptionist clicks the button "book"
window closes
program displays a new window with a form
repectionist inputs a name
repectionist inputs a birth date
repectionist inputs an address
repectionist inputs a passport number
repectionist inputs a phone number
repectionist inputs an email
repectionist inputs an arrival date
repectionist inputs a departure date
repectionist clicks the button "save"
[Wrong input]
Receptionist sees where the mistake is and corrects it
[Otherwise]
The window closes and the room is booked
The program displays the main window

4 Design documentation

4.1 Sequence diagram



4.2 UML class diagram


pkg
Guest
- name : String
- address : String
- phone : String
- email : String
- passport : String
- dateOfBirth : String
- roomNumber : int
- extraBed : boolean
- checked : boolean
+ Guest(name : String, address : String, phone : String, email : String, passport : String, dateOfBirth : String, arrivalDate : String, departureDate : String, extraBed : boolean)
+ Guest(name : String, address : String, phone : String, email : String, passport : String, dateOfBirth : String, arrivalDate : String, departureDate : String, extraBed : boolean, roomNumber : int)
+ getName() : String
+ getAddress() : String
+ getPhone() : String
+ getEmail() : String
+ getPassport() : String
+ getArrivalDate() : String
+ getDepartureDate() : String
+ setName(name : String) : void
+ setAddress(address : String) : void
+ setPhone(phone : String) : void
+ setEmail(email : String) : void
+ setPassport(passport : String) : void
+ setDateOfBirth(dateOfBirth : String) : void
+ setArrivalDate(arrivalDate : String) : void
+ setDepartureDate(departureDate : String) : void
+ setRoomNumber(roomNumber : int) : void
+ toString() : String
+ getExtraBed() : boolean
+ setChecked(checked : boolean) : void
+ isChecked() : boolean
- date
1
Date
- departureDay : int
- departureMonth : int
- departureYear : int
- arrivalDay : int
- arrivalMonth : int
- arrivalYear : int
- todayDay : int
- todayMonth : int
- todayYear : int
+ setArrivalDate(arrivalDate : String) : void
+ getArrivalDate() : String
+ getTodayDate() : String
+ setDepartureDate(departureDate : String) : void
+ getDepartureDate() : String
+ getDaysOfStay() : Integer
+ getDaysOfStayForCheckOut() : Integer
+ getDaysOfMonth(month : int) : Integer
+ isLeapYear(year int : int) : boolean
+ addToBookedDates() : void
+ addToBookedDates(arrival : String, departure : String) : void
+ deleteFromBookedDates() : void
+ getArrivalYear() : int
+ setArrivalYear(arrivalYear : int) : void
*
0..4
- 1
guiBooking
- day1text : String
+ guiBooking() : void
+ keyTyped(arg0 : KeyEvent) : void
+ focusGained(arg0 : FocusEvent) : void
+ keyTyped(e : KeyEvent) : void
+ focusGained(arg0 : FocusEvent) : void
+ setGuest() : void
+ getGuest() : Guest
+ setVisibility(value : boolean) : void
+ reset() : void
+ dateChecker(dt : String) : String
+ String dateFix(dt : String) : String
+ noException() : boolean
+ isAfterToday(date String : int, today1 String : int) : boolean
+ checkForAvilability() : boolean
+ checkFirstAvailableDate() : boolean
0..1
Room
- numberOfBeds : int
- roomNumber : int
- price : int
- defaultPrice : int
- additionalFees : int
- typeOfBed : String
- type : String
- extraBed : boolean
+ Room(numberOfBeds : int, roomNumber : int, defaultPrice : int, typeOfBed : String, type : String) : void
+ addAdditionalFees(value : int) : void
+ isExtraBed() : boolean
+ setExtraBed(extraBed : boolean) : boolean
+ getNumberOfBeds() : int
+ getAdditionalFees() : int
+ getTypeOfBed() : String
+ getType() : String
+ getRoomNumber() : int
+ setNewPrice(price : int) : void
+ getNumberOfGuests() : Integer
+ addGuest(guest : Guest) : void
+ changePrice(newPrice : int) : void
+ isAvilableForAvilableRooms() : boolean
+ isAvilable() : boolean
+ getPrice() : int
+ getDefaultPrice() : int
+ getPriceToPay() : int
+ getArrivalDate() : String
+ getDepartureDate() : String
+ reset() : void
+ addToBookedDates(arrival : String, departure : String) : void
Main
+ main(args : String[]) : Main
+ initialize() : void
+ valueChanged(e : ListSelectionEvent) : void
- setArrays() : void
+ addBooking(guest : Guest) : void
+ run() : void
+ save() : void
+ getSelectedRoomNumber() : int
+ getSelectedIndex() : int
+ getSelectedValue() : String
+ setNewRoom(index : int, room : Room) : void
+ searchRoomByDate(date1 : String, date2 : String) : void
+ setGuestInformationFromRoom() : void
+ setGuestInformationFromBooking() : void
+ automaticBooksDeleter() : void
0..25
0..1
0..1
0..1
guiCheckOut
+ guiCheckOut()
+ initialize() : void
0..1
guiCheckOutInfo
- numberOfGuests : int
- generalFees : int
- additionalFees : int
- total : int
- daysOfStay : int
+ guiCheckOutInfo()
+ initialize() : void
guiCheckIn
- bookingIndex : int
- day1text : String
+ guiCheckIn()
+ addAllGuests(numberOfGuests : int) : void
+ setGuest(index : int) : void
+ dateChecker(dt : String) : String
+ dateFix(dt : String) : String
+ noException() : boolean
+ emtyFieldChecker(listFields : ArrayList<JTextField>, listLabels : ArrayList<JLabel>) : void
+ wrongDate(listFields : ArrayList<JTextField>, listLabel : ArrayList<JLabel>) : void
+ clearErrorChecker(listLabels : ArrayList<JLabel>, j : int) : boolean
+ removeTextField(list : ArrayList<JTextField>) : void
+ removeLabel(list : ArrayList<JLabel>) : void
+ setVisible(status : boolean) : void
+ removeFieldsAndLabels(number : int) : void
+ clearArrayList(number : int) : void
+ generateTextFields(x : int, y : int, guestNr : int) : void
+ generateGuest1() : void
+ generateGuest2() : void
+ generateGuest3() : void
+ generateGuest4() : void
+ buildTextForm() : void
+ generateErrorLabels(x : int, y : int, guestNr : int) : void
+ buildErrorLabels() : void
+ setFirstGuest() : void
+ isBooked() : boolean
+ isAfterToday(date : String, today1 : String) : boolean
+ checkForAvilability() : boolean
+ isArrivalToday() : boolean
guiEdit
- day1text : String
- bookingIndex : int
+ guiEdit()
+ editGuest() : void
+ dateChecker(dt : String) : String
+ dateFix(dt : String) : String
+ noException() : boolean
+ isAfterToday(date : String, today1 : String) : boolean
+ checkForAvilability() : boolean
+ checkFirstAvailableDate() : boolean
+ operation36() : void