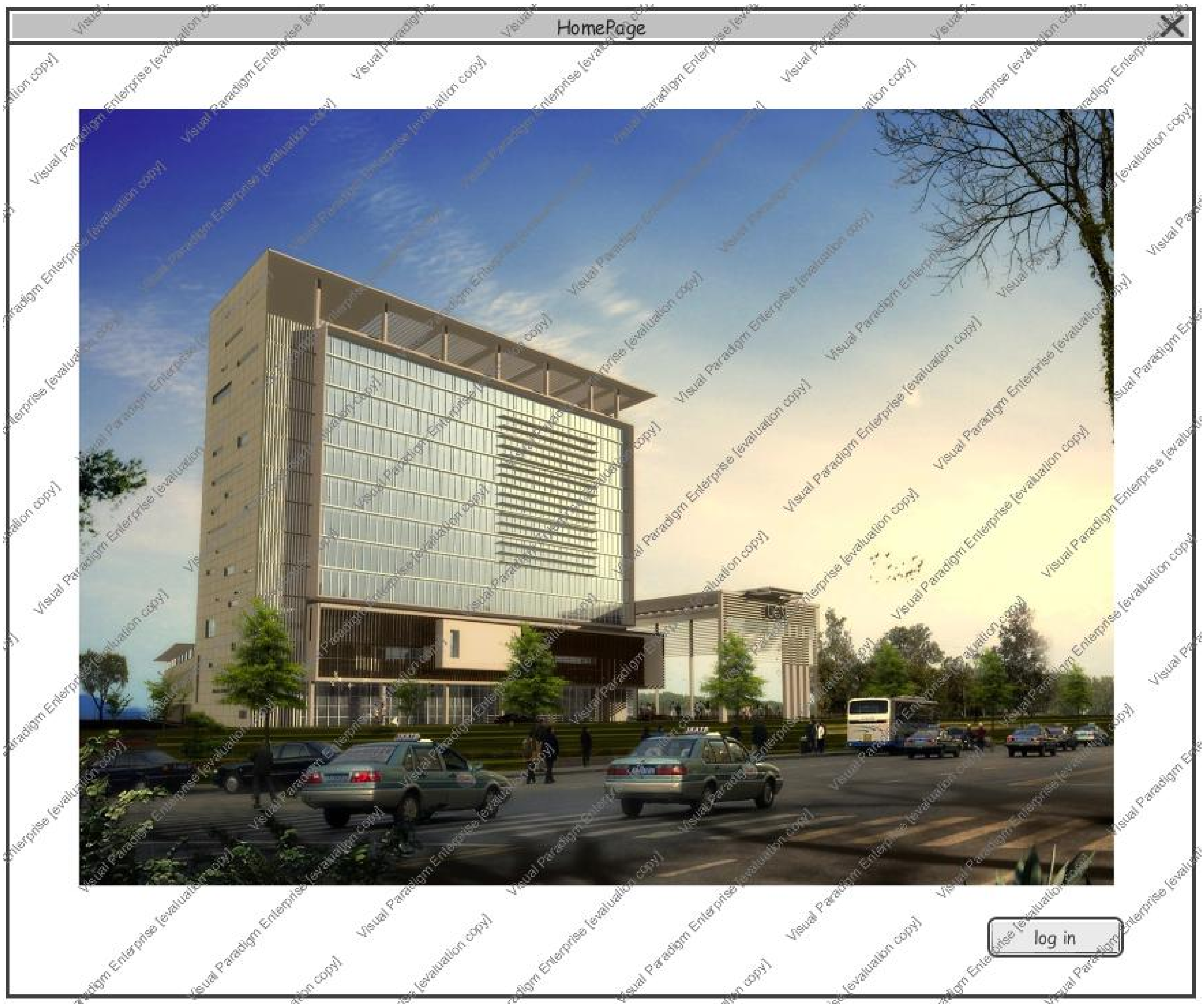
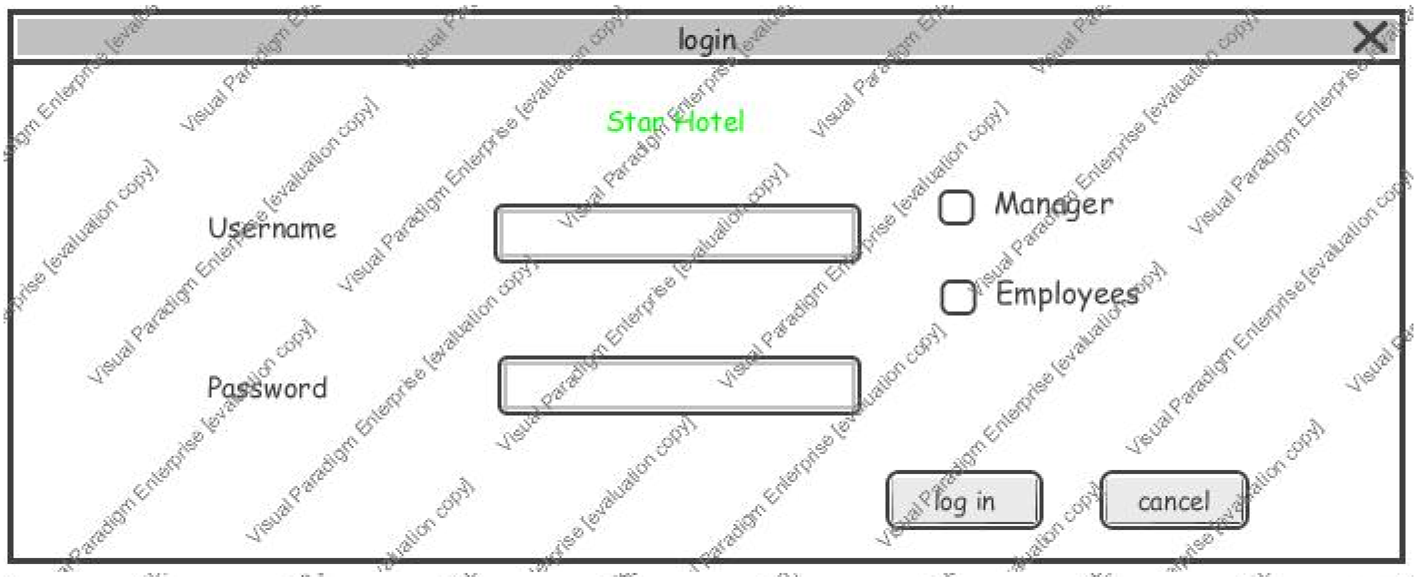
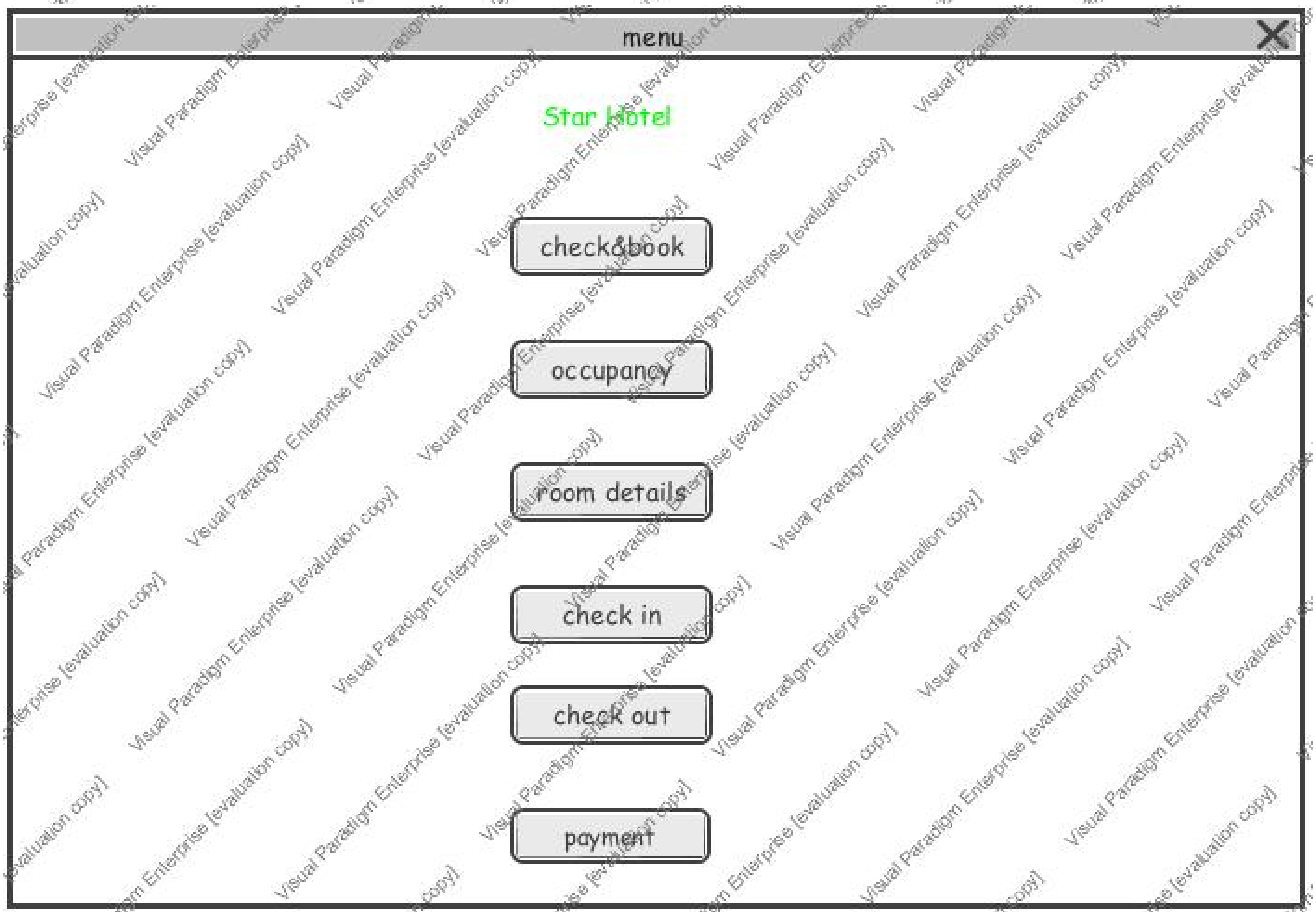
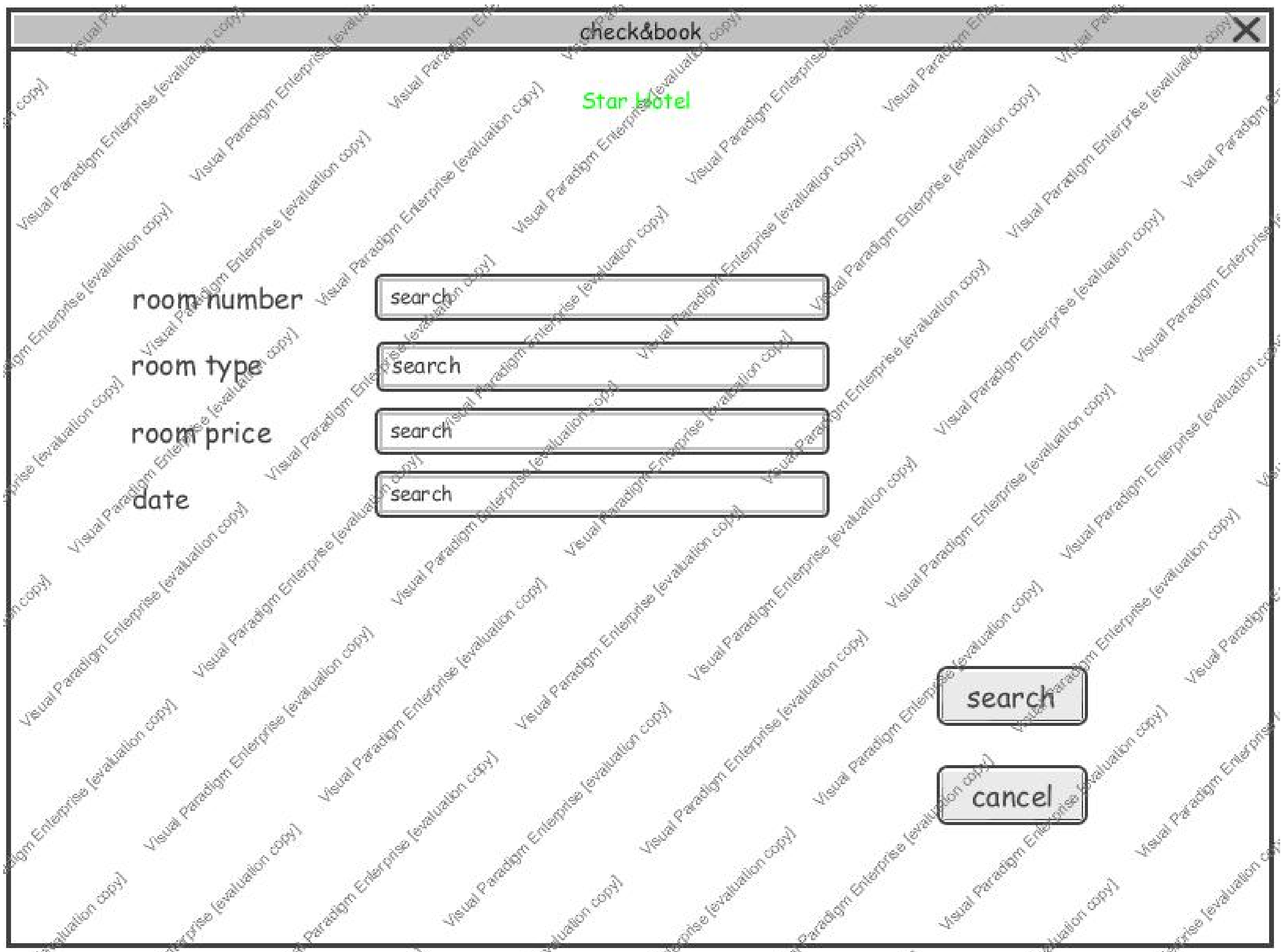
## User Interface Design:



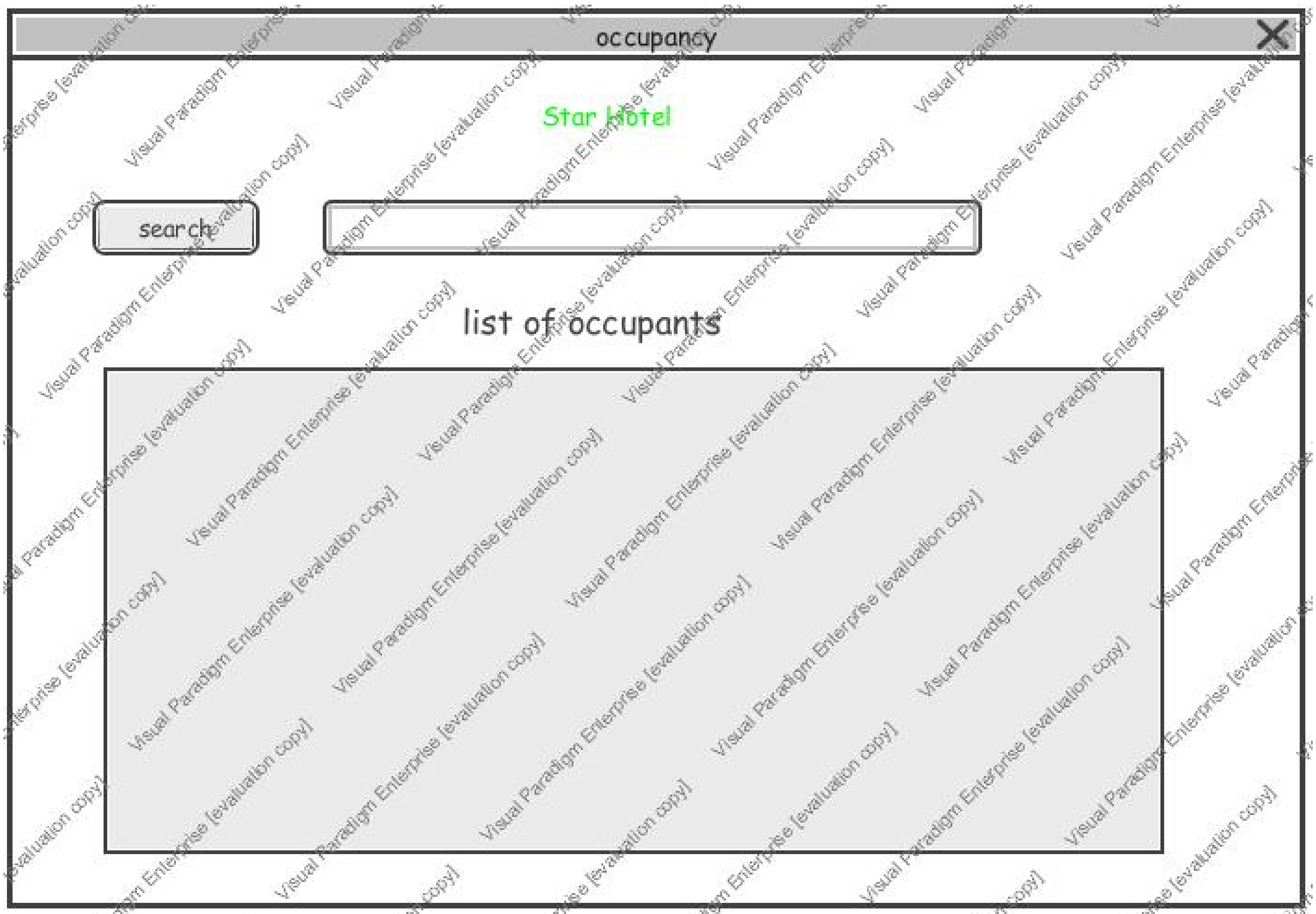

login
Star Hotel
Username
Password
Manager
Employees
log in
cancel


menu
Star Hotel
check&book
occupancy
room details
check in
check out
payment

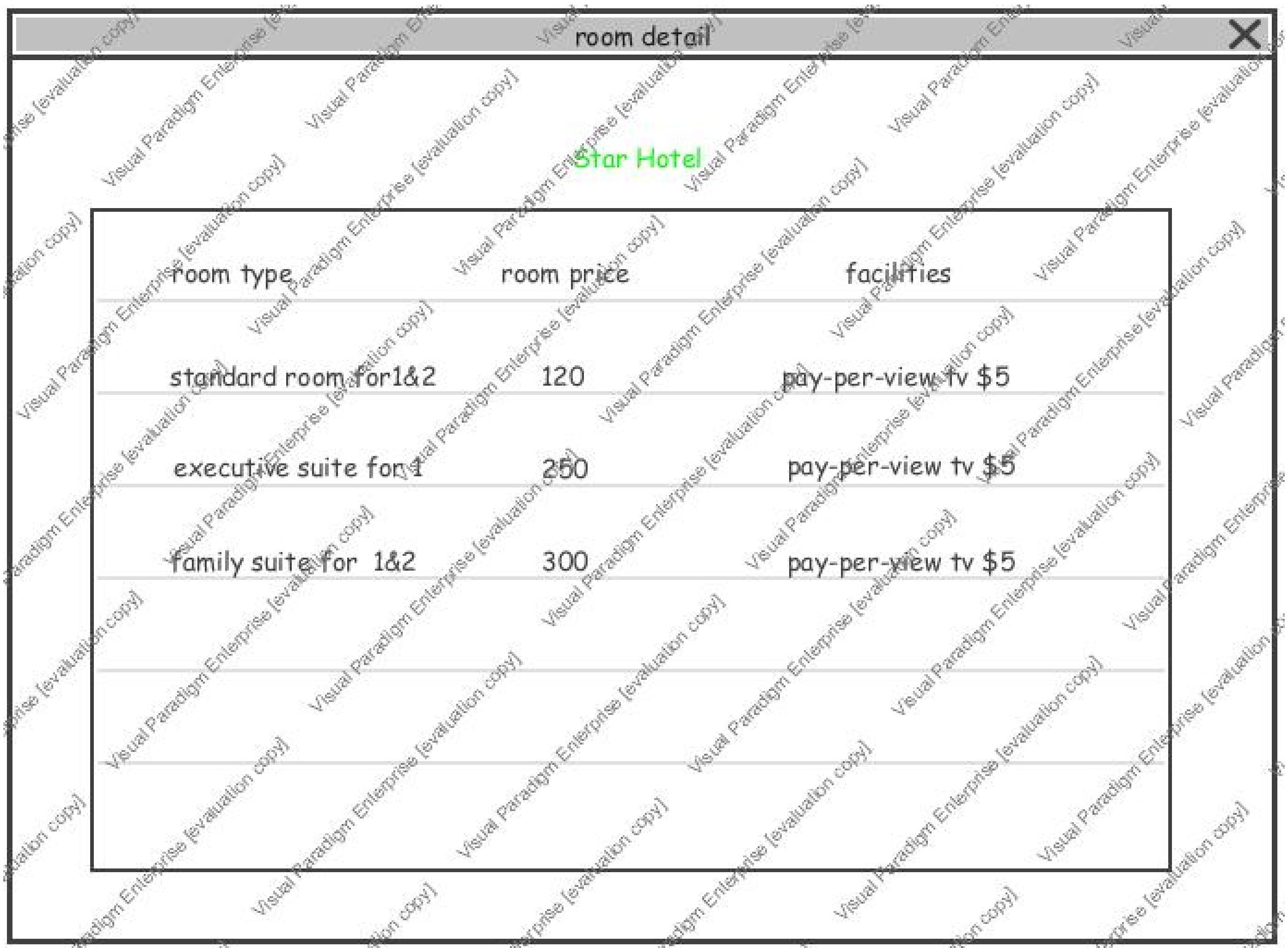
Annotation


check&book
Star Hotel
room number
search
room type
search
room price
search
date
search
search
cancel

Annotation



Annotation

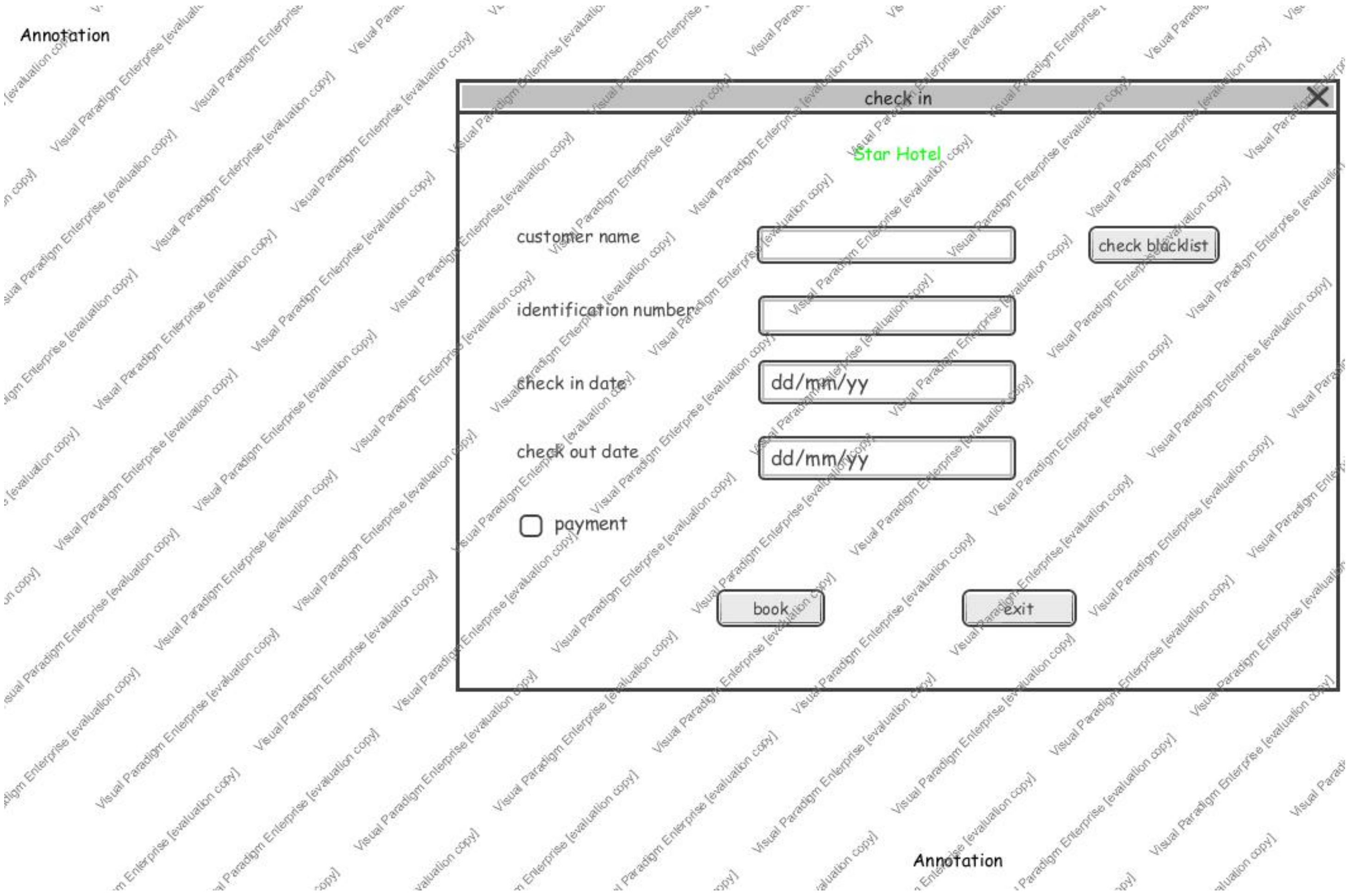


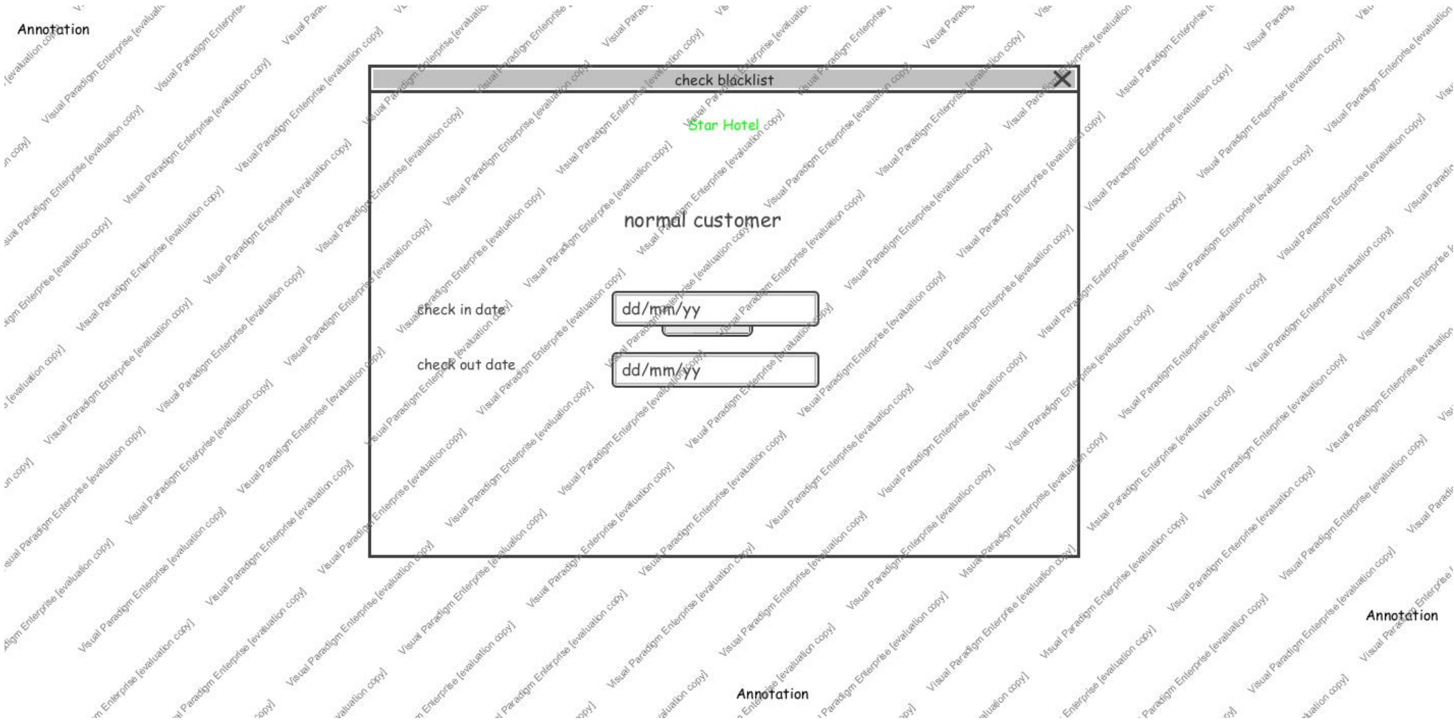
room detail

Star Hotel

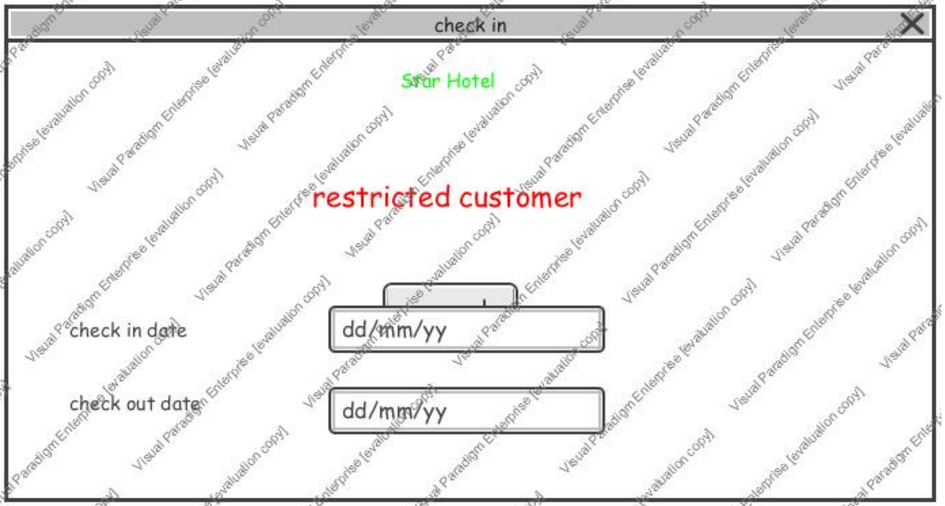
| room type | room price | facilities |
|---|---|---|
| standard room for1&2 | 120 | pay-per-view tv $5 |
| executive suite for 1 | 250 | pay-per-view tv $5 |
| family suite for 1&2 | 300 | pay-per-view tv $5 |

Annotation


Annotation
check in
Star Hotel
customer name
check blacklist
identification number
check in date
dd/mm/yy
check out date
dd/mm/yy
payment
book
exit
Annotation

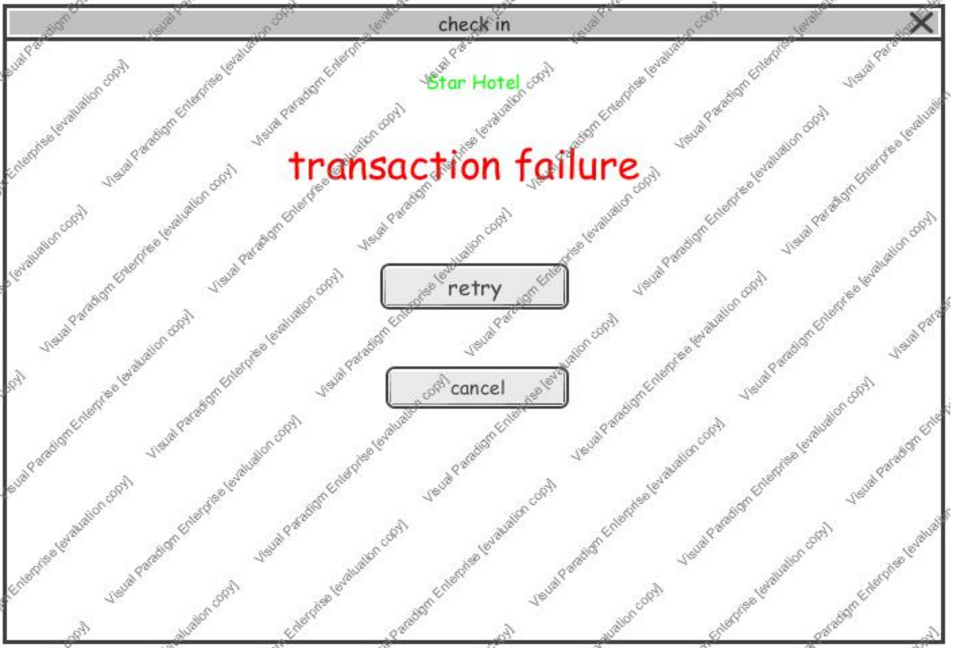

Annotation
check blacklist
Star Hotel
normal customer
check in date
dd/mm/yy
check out date
dd/mm/yy
Annotation
Annotation

Annotation



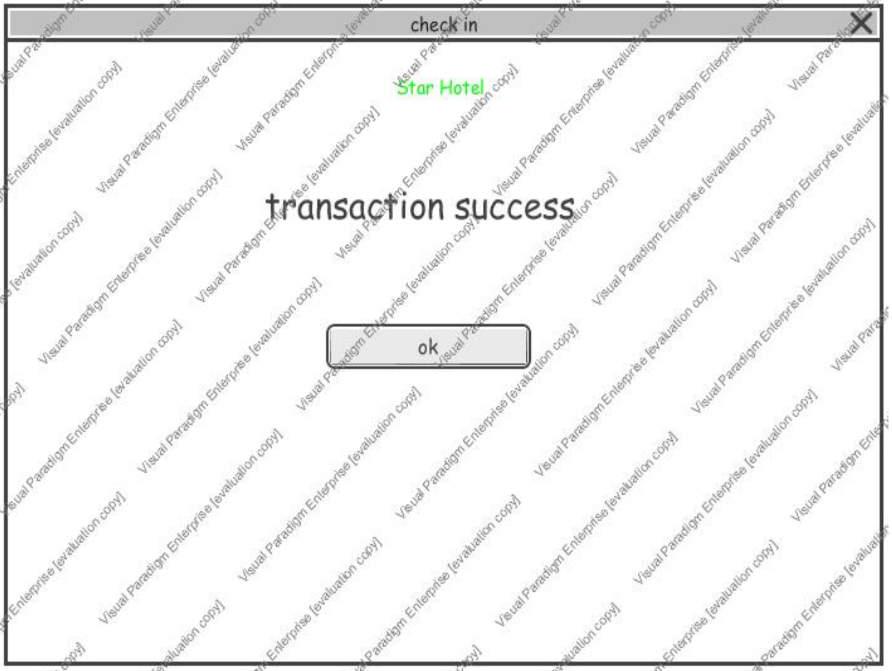
Annotation

Annotation



Annotation

Annotation



Annotation

## Entity-Relationship diagram:

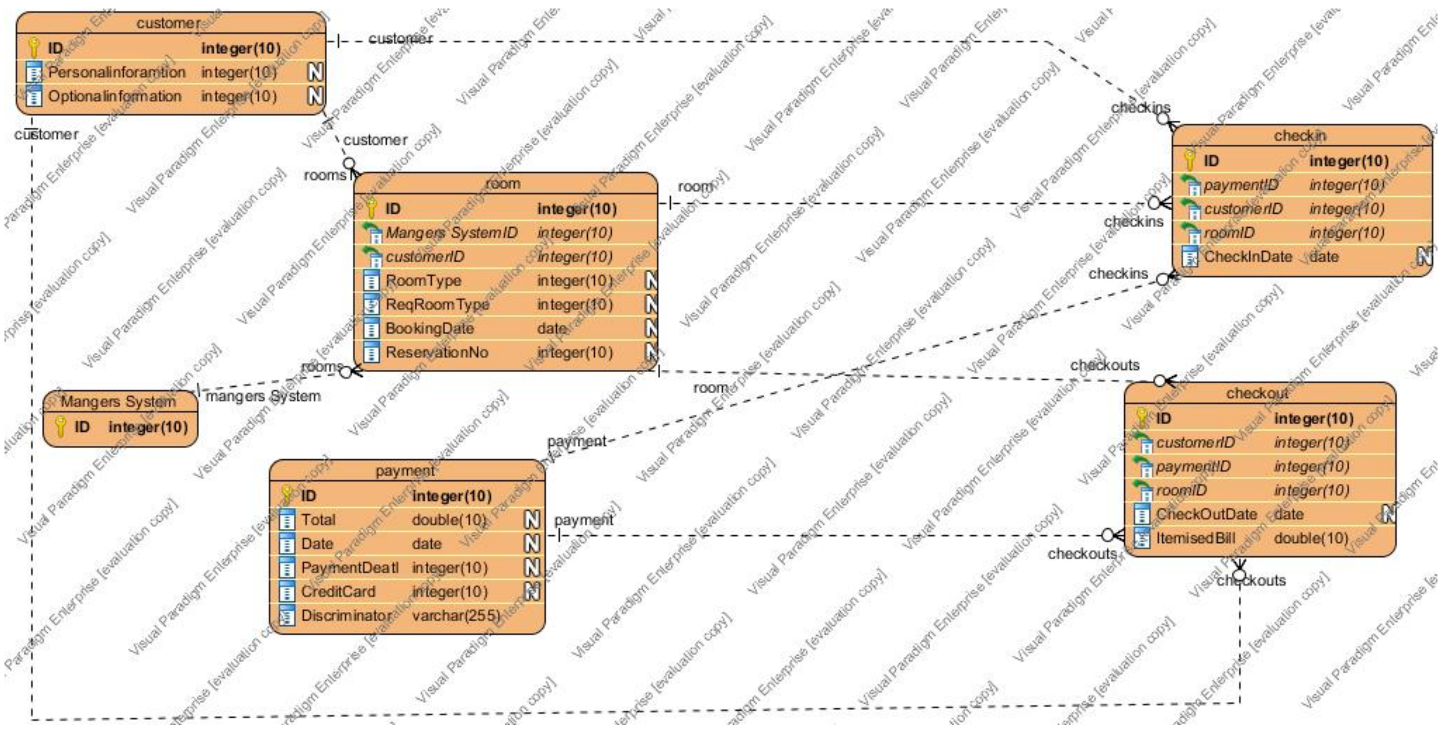
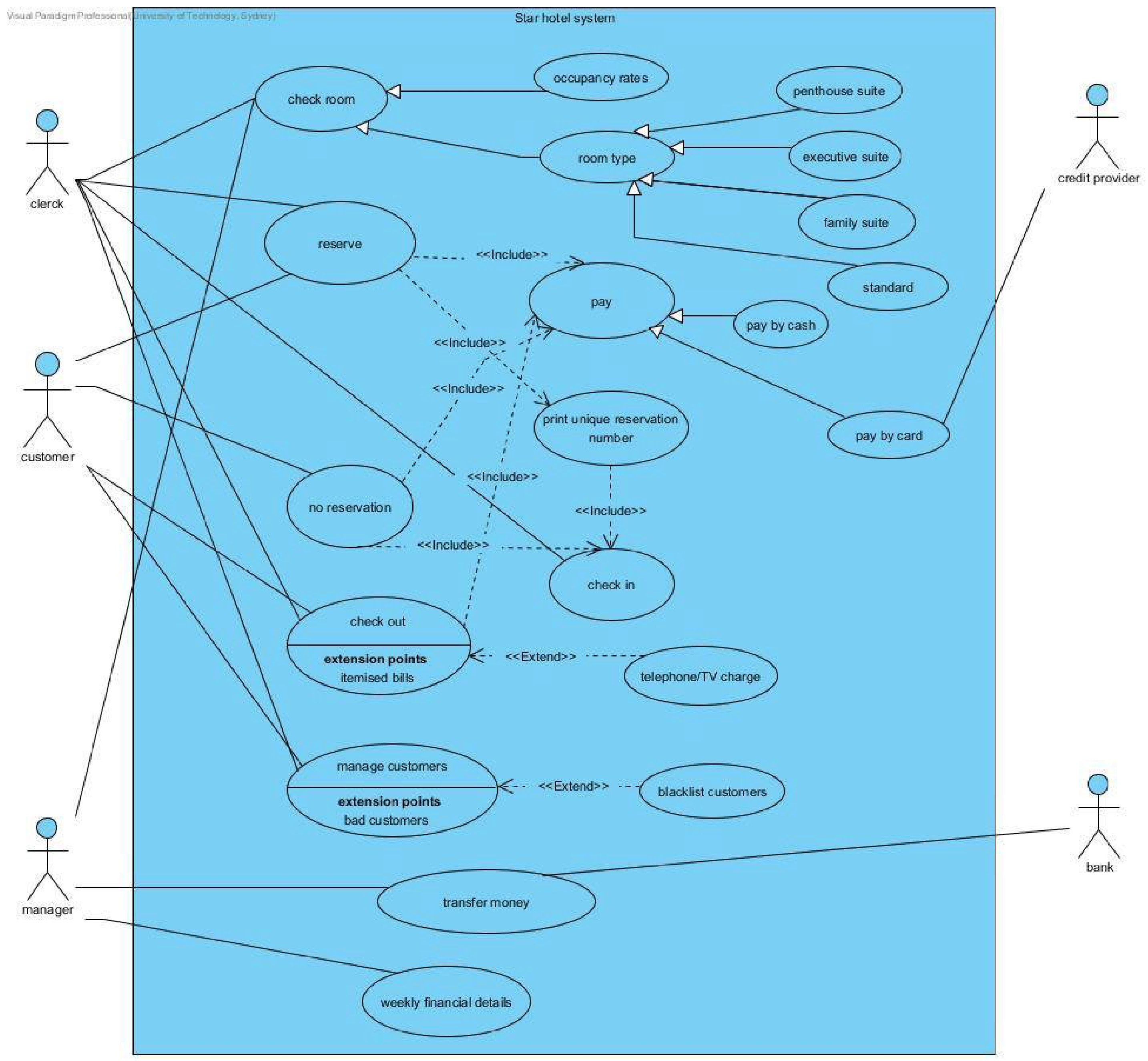
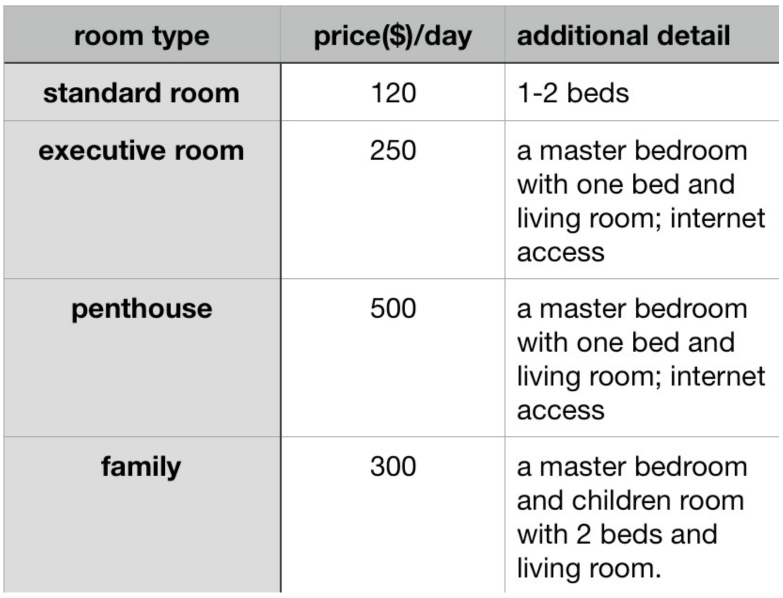


**Figure: ERD Diagram**

# Star hotel system
# Use case diagram



| | |
|---|---|
| **Use Case UC1** | Check room |
| **Goal in Context** | A customer interests in detail information about rooms such as what types of room, price, facilities and vacancy. |
| **Scope & Level** | Star hotel booking system, Secondary Task |
| **Preconditions** | None |
| **Postconditions** | The customer receives the suitable room information and makes a decision about whether to book room or not. |
| **Actors** | Clerk : Primary |
| **Trigger** | The customer enquires suitable room information from the hotel. |
| **Description** | 1. The customer enquires Star hotel to know suitable room information by phone, email or in person with the clerk.<br>2. Clerk enters hotel system to check the type of room required by the customer and gives feedback to the customer.<br>3. The customer can ask for an own preference room or searches an available suitable room of the hotel.<br>4. Clerk checks the available room required and answers customer's quires.<br>5. The customer receives the results. |
| **Extensions** | 1a. Clerk misses the customer's enquiry.<br>2a. Customer does not satisfy with given room information.<br>4a. Customer does not satisfy with given room information. |
| **Other information** | 1. The hotel has 50 standard rooms, 20 executive suites, 2 penthouse suites and 8 family suites. Every room has a bathroom, a telephone, cable TV, pay-preview TV and etc. The categories and price are shown below.<br> |

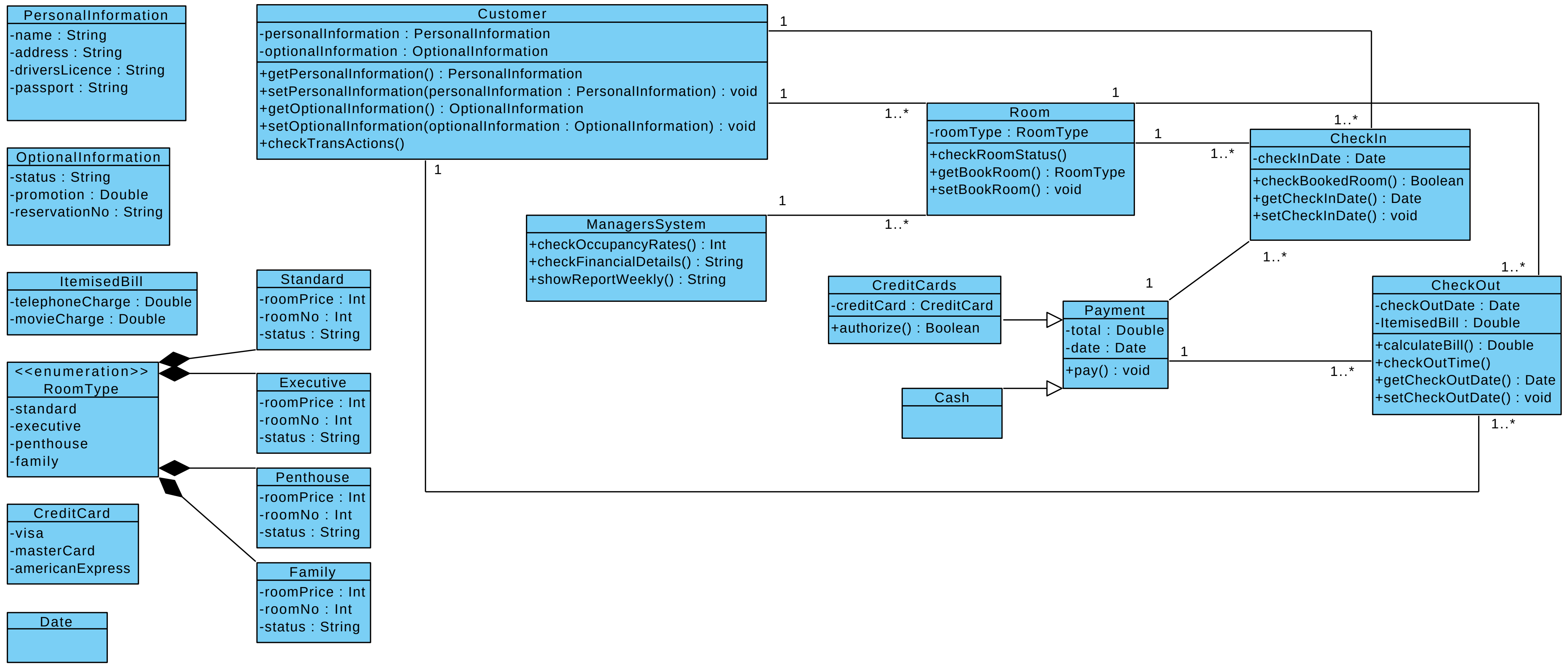
| room type | price($)/day | additional detail |
|---|---|---|
| **standard room** | 120 | 1-2 beds |
| **executive room** | 250 | a master bedroom with one bed and living room; internet access |
| **penthouse** | 500 | a master bedroom with one bed and living room; internet access |
| **family** | 300 | a master bedroom and children room with 2 beds and living room. |

| | |
|---|---|
| **Unresolved Issues** | None |

| | |
|---|---|
| **Use Case UC2** | Reserve |
| **Goal in Context** | Customer want to book the room in Star hotel. |
| **Scope & Level** | Star hotel booking system, Primary Task |
| **Preconditions** | The customer gets proper room information from the hotel. |
| **Postconditions** | The customer completes booking information. |
| **Actors** | Customer, Clerk : Primary<br>Bank : Secondary |
| **Trigger** | The customer gains satisfying information of the hotel room. |
| **Description** | 1. The customer gains satisfying information of the hotel room and wants to book the room.<br>2. Clerk receives customer’s booking apply and asks customer to provide customer’s credit card detail.<br>3. The customer supplies credit card information.<br>4. The details are validated.<br>5. Clerk asks customer to pay 10% advance deposit by credit card.<br>6. The customer pays for reservation deposit.<br>7. Clerk gives customer a unique reservation identifier.<br>8. The customer receives the unique reservation identifier and completes booking information. |
| **Extensions** | 3a. The customer can not provide his/her credit card information.<br>4a. Customer’s credit card information are invalidated. |
| **Other information** | Unique reservation identifier consists of the room number 3 digits, followed by start date of the reservation (6 digits in the form ddmmyy). |
| **Unresolved Issues** | None |

| | |
|---|---|
| **Use Case UC3** | Check in |
| **Goal in Context** | Customer wants to check in Star hotel. |
| **Scope & Level** | Star hotel booking system, Primary Task |
| **Preconditions** | 1. Customer has not booked for Star hotel room yet : none.<br>2. The customer booked for Star hotel room successfully. |
| **Postconditions** | The customer gets the room key and enter the hotel room. |
| **Actors** | Customer, Clerk : Primary<br>Bank : Secondary |
| **Trigger** | The customer arrives at the hotel. |
| **Description** | 1. Customer arrives at the hotel.<br>2. Clerk asks customers' name, reservation number if the customer has already booked for the room or asks for customer's credit card and identification for non-reservation.<br>3. The customer provides the identifier number or credit card.<br>4. Customer's information is validated by clerk.<br>5. Clerk asks customer to sign room booking form.<br>6. The Customer fills his/her information into the form.<br>7. Clerk gives the room key to the customer. |
| **Extensions** | 3a. Customer provides wrong identifier information to clerk.<br>4a. Customer's information is invalidated or customer who is on hotel's blacklist can not get inside. |
| **Other information** | 1. The identification data consists of customer's name, gender, and address: street number and name, the suburb and post code.<br>2. A regular customer gets 5% discount in their final charge.<br>3. The booking form shows room number and period, the customer's identification data and payment details. |
| **Unresolved Issues** | None |

| | |
|---|---|
| **Use Case UC4** | Check out |
| **Goal in Context** | A customer wants to check out Star hotel. |
| **Scope & Level** | Star hotel booking system, Primary Task |
| **Preconditions** | Customer wants to check out. |
| **Postconditions** | The customer pays for room payment and leave the hotel room. |
| **Actors** | Customer, Clerk : Primary<br>Bank : Secondary |
| **Trigger** | The customer arrives at the hotel receptionist. |
| **Description** | 1. Customer arrives at the hotel receptionist before 11am.<br>2. Clerk gives the customers itemized bill and asks customer to pay for the room.<br>3. The customer pays for the payment.<br>4. Clerk gives payment receipt to the customer.<br>5. The customer leaves the hotel. |
| **Extensions** | 3a. Customer leaves without paying.<br>5a. Customer wants to extend the stay. |
| **Other information** | 1. Itemized bill consists of telephone charge and pay-per-view TV.<br>1. A general audience movie costs $5 and an adult movie costs $10 each.<br>2. The phone bill is not more than $200.<br>2. Limit of Penthouse suite is $5000 being only paid by credit card, if the amount owing is over $2000. |
| **Unresolved Issues** | None |


PersonalInformation
-name : String
-address : String
-driversLicence : String
-passport : String
OptionalInformation
-status : String
-promotion : Double
-reservationNo : String
ItemisedBill
-telephoneCharge : Double
-movieCharge : Double
<<enumeration>>
RoomType
-standard
-executive
-penthouse
-family
CreditCard
-visa
-masterCard
-americanExpress
Date
Customer
-personalInformation : PersonalInformation
-optionalInformation : OptionalInformation
+getPersonalInformation() : PersonalInformation
+setPersonalInformation(personalInformation : PersonalInformation) : void
+getOptionalInformation() : OptionalInformation
+setOptionalInformation(optionalInformation : OptionalInformation) : void
+checkTransActions()
Standard
-roomPrice : Int
-roomNo : Int
-status : String
Executive
-roomPrice : Int
-roomNo : Int
-status : String
Penthouse
-roomPrice : Int
-roomNo : Int
-status : String
Family
-roomPrice : Int
-roomNo : Int
-status : String
ManagersSystem
+checkOccupancyRates() : Int
+checkFinancialDetails() : String
+showReportWeekly() : String
Room
-roomType : RoomType
+checkRoomStatus()
+getBookRoom() : RoomType
+setBookRoom() : void
CheckIn
-checkInDate : Date
+checkBookedRoom() : Boolean
+getCheckInDate() : Date
+setCheckInDate() : void
CreditCards
-creditCard : CreditCard
+authorize() : Boolean
Cash
Payment
-total : Double
-date : Date
+pay() : void
CheckOut
-checkOutDate : Date
-ItemisedBill : Double
+calculateBill() : Double
+checkOutTime()
+getCheckOutDate() : Date
+setCheckOutDate() : void
1
1
1..*
1
1..*
1
1..*
1
1..*
1
1..*
1..*
1
1..*
1
1..*
1
1..*

## Java code:

### 1) CheckIn Class

```
package assignment3;

import java.util.ArrayList;
import java.util.Scanner;

public class CheckIn {

	private String checkin;



	public static void main(String[] args)
	{
		CheckIn checkin1;
		String namecustomerlist;
		int result;
		String payment;

		Customer x;
		x= new Customer("test");

		checkin1 = new CheckIn("newcheckin");
		result=checkin1.ProcessCheckIn();

		if(result==1)
		{
		Scanner ind = new Scanner(System.in);
		System.out.println("Enter customername :");
		namecustomerlist=ind.next();
		x.addcCustomerInformation(namecustomerlist);
		System.out.println("Checkin has been successful");
		return;
		}
		if(result==0)
		{
			Scanner in= new Scanner(System.in);
			 System.out.println("Enter payment result:(yes/no)");
			 payment= in.next();
			 if(payment.equals("no"))
					{
					System.out.println("Payment has not been paid");
					return;

					}
		else if(payment.equals("yes"))
					{
```

```
                                        System.out.println("Payment has paid");
                                        Scanner ind = new Scanner(System.in);
                                        System.out.println("Enter customername :");
                                        namecustomerlist=ind.next();
                                        x.addcCustomerInformation(namecustomerlist);
                                        System.out.println("Checkin has been successful");
                                        return;
                                        }
                }
                else
                {
                        System.out.println("Checkin has not been successful");
                        return;
                }

        }

        public CheckIn(String initialName)
        {
        checkin = initialName;
        }


        public int ProcessCheckIn()
        {
                String reservationnumber;
                int i;
                int result=0;
                Customer customer1;
                String customercheckin;


                customer1 = new Customer("customer1");

                customer1.addcCustomerInformation("15102018307"); //assume that customer reservation
number has been added

                customer1.customerCheck();


                Scanner in= new Scanner(System.in);
                System.out.println("Search for reservationno:");
                reservationnumber = in.next();

                for(i=0;i<customer1.customerlist.size();i++)
```

```
                    {
                              customercheckin=customer1.getCustomerInformation();

                              if((reservationnumber.equals("15102018307")) ||
(reservationnumber.equals(customercheckin)))
                              {
                                        System.out.println("reservation has been found");
                                        result =1;
                                return result;
                              }
                    }
                    System.out.println("reservation number has not been found");
                              return result;

          }

}
```

## 2) Customer Class

```
package assignment3;

import java.util.ArrayList;
import java.util.Scanner;

public class Customer {
        private ArrayList<Customer> blacklists = new ArrayList<Customer>();
        public ArrayList<Customer> customerlist = new ArrayList<Customer>();
        private String nameofblacklist;
        private String namecustomer;
        private String name;


        public void customerCheck()
        {


                String creditcardno;
                Scanner in= new Scanner(System.in);
                System.out.println("Enter creditcardNo or customer name");
                creditcardno = in.next();



                if ((creditcardno.equals("123456789"))||(creditcardno.equals("Adam"))) // can add more conditions
                {
                        System.out.println("this customer's in hotel blacklist");
                   System.exit(0);
                }
                System.out.println("This customer is no in hotel blacklist");


        }


        public void addBlacklist()
        {
        Customer x;
        String nameblacklist;
        Scanner inc = new Scanner(System.in);
        System.out.println("Enter blacklistname :");
        nameblacklist=inc.next();
        x= new Customer(nameblacklist);
        blacklists.add(x);
        }

        public Customer(String initialName)
        {
                namecustomer=initialName;
        }

        public void addcCustomerInformation(String name)
        {
                Customer x;

                x= new Customer(name);
                customerlist.add(x);
                        }

        public String getCustomerInformation() {
                return name;
        }



}
```

## 3) Payment Class

```
package assignment3;


import java.util.Scanner;

public class Payment {
	String payment;

	public Payment(String initialName)
	{
	payment = initialName;
	}

	public static void main(String[] args)
	{
		Payment x;

		x= new Payment("newpayment");
		x.paymentdetail();


	}

	public void paymentdetail()
	{
		int value;
		String creditcardinvalid ="1234567890"; //assume this creditcard number is invalid
		Scanner in= new Scanner(System.in);
		System.out.println("Pay by cash press 1");
		System.out.println("Pay by card press 2");
		value= in.nextInt();
		String x;


		switch(value)
	{
		case 1: System.out.println("pay by cash");
				Scanner inx= new Scanner(System.in);
				System.out.println("Enter payment result:(yes/no)");
				payment= inx.next();
				if(payment.equals("no"))
				{
				System.out.println("Payment has not been paid");
				System.exit(0);
				}
```

```
                    else if(payment.equals("yes"))
                    {
                    System.out.println("Payment has been paid");
                    }
                    break;

            case 2: System.out.println("pay by card");
                    Scanner ina= new Scanner(System.in);

                    System.out.println("Enter credit card number for checking validation");
                    x=ina.next();
                    if(x.equals(creditcardinvalid))
                    {
                        System.out.println("credit card Invalid: unsuccessful");
                        System.exit(0);
                    }
                    else
                    {
                        System.out.println("Payment has been paid");
                        System.out.println("successful");
                    }
                    break;

            default: System.out.println("wrong number"); break;

        }
        }
}
```

## 4) Room Class

```
package assignment3;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.Scanner;
import java.util.Calendar;
import java.time.LocalDateTime;

public class Room {
        private String name;
        private int prices;
        private char status;
        private int roomNo;
        private Room a;
        public ArrayList<Room> rooms = new ArrayList<Room>();



public Room(String initialName)
        {
        name = initialName;
        }


public static void main(String[] args)
{
Room room1;
String name1;
int check;
int roomnomain;
int recno;  //rec = recieve
int recbookno;
char resultfrombooking;

String standard ="standard"; //for assume static parameter
String executive = "executive";
String penthouse = "penthouse";
String family = "family";
char A='A';
char F='F'; // for assume static parameter
```

```
room1 = new Room("hotel");
room1.addRoomName(standard);
room1.addRoomName(executive);
room1.addRoomName(family);
room1.addRoomName(penthouse);


room1.enterPrice(standard,120);
room1.enterPrice(executive,250);
room1.enterPrice(family,300);
room1.enterPrice(penthouse,500);

room1.setStatus(standard,A);
room1.setStatus(executive,A);
room1.setStatus(family,A);
room1.setStatus(penthouse,F);

room1.setRoomNumber(standard,17);
room1.setRoomNumber(executive,307);
room1.setRoomNumber(family,602);
room1.setRoomNumber(penthouse,701);


Scanner in= new Scanner(System.in);
System.out.println("Press 1 to show all of rooms ");
System.out.println("Press 2 to search rooms ");
recno= in.nextInt();
if(recno==1)
{
        room1.process();
}

if(recno==2)
{
Scanner inx= new Scanner(System.in);
System.out.println("Enter searched room :");
name1= inx.next();
check=room1.checkRoom(name1);
roomnomain =check; //compared parameter of roomNo
if(check!=0)
        {check=1;} // for let system know what roomNo is (Y)

switch(check)
        {
                case 0: System.out.println("The room is full");
                break;
```

```
            case 1: System.out.println("The room is available");
            Scanner ina= new Scanner(System.in);
            System.out.println("press 1 for book");
            System.out.println("press 2 for exit");
            recbookno=ina.nextInt();
            if(recbookno!=1)
            {
                System.exit(0);
            }

            if(recbookno==1)
            {
            resultfrombooking=room1.bookRoom(roomnomain);
            if (resultfrombooking == 'S')
            {
                System.out.println("process has been successful");
            }
            }
            break;

            default: System.out.println("ERROR"); break;
        }
}
if((recno!=1)&&(recno!=2)) {
        System.out.println("please enter 1 or 2");
        }

}
public void process() {
        Room t;
        int i;
        System.out.println("   name  |"+"room No|"+"price|"+"Availibity ");
        for(i=0;i<rooms.size();i++)
        {
        t=rooms.get(i);
        t.name=t.getName();
        t.prices=t.getRoomPrice();
        t.roomNo=t.getRoomNo();
        System.out.println(t.name+" |   "+t.roomNo+"  | "+t.prices+" | "+t.status);
}}

public void addRoomName(String name)
{
Room x;
x = new Room(name);
```

```
rooms.add(x);
}


public String getName(){
return name;
}

public void enterPrice(String name, int price)
{
Room t;
int i;
for(i=0;i<rooms.size();i++)
        {
        t=rooms.get(i);
        if(t.getName().equals(name))
                {
                t.prices = price;

                return;

                }
        }
}

public void setStatus(String name, char stat)
{
        int i;
        Room t;
                for(i=0;i<rooms.size();i++)
                        {
                                t=rooms.get(i);
                                if(t.getName().equals(name))
                                {
                                        t.status = stat;

                                        return;
                                }
                        }
}

public void setRoomNumber(String name, int no)
{
        int i;
        Room t;
                for(i=0;i<rooms.size();i++)
```

```
                        {
                                t=rooms.get(i);
                                if(t.getName().equals(name))
                                        {
                                                t.roomNo = no;
                                                return;
                                        }
                        }
}

public char getRoomStatus(){
return status;
}

public int getRoomPrice(){

return prices;
}

public int getRoomNo(){
return roomNo;
}

public int checkRoom(String name1)
{       Room t;
        int roomnoo1;
        int i;
        char st;
                for(i=0;i<rooms.size();i++)
                {
                        t=rooms.get(i);
                        if(t.getName().equals(name1))
                        {

                                st=t.status;
                                System.out.println(t.name+"   "+t.roomNo+"  "+t.prices+"   "+t.status);
                                {
                                        if(st == 'A')
                                        {
                                                roomnoo1=t.getRoomNo();
                                                return roomnoo1;
                                        }
                                        if(st == 'F')
                                        {   roomnoo1= 0;
                                                return roomnoo1;
                                        }
                                }
                        }
                }
                 System.out.println("Wrong spelling of the room");
                System.exit(0);
                return 0;
}

public char bookRoom(int roomnomain)
{ int creditnumber;
  String payment;
  Customer x;
  char result; //F or S
  Room t;
  String combine;

  ArrayListCustomerInformation z;
  z= new ArrayListCustomerInformation("New");

  int i;
  String a1;
  String reservationno;
  x= new Customer("Customer1");

  x.customerCheck();
  char resultofbooking='U';

  Scanner in= new Scanner(System.in);
  System.out.println("Enter payment result:(yes/no)");
  payment= in.next();
  if(payment.equals("no"))
                {
                System.out.println("Payment has not been paid");
                System.exit(0);
                }
  else if(payment.equals("yes"))
                {
                        for(i=0;i<rooms.size();i++)
                                {
                                        t=rooms.get(i);
                                        if(t.getRoomNo()==roomnomain)
                                                {
                                                a1=t.getName();
                                                t.setStatus(a1,'F');
                                                String numberconverted=Integer.toString(roomnomain);
                                                DateFormat df = new SimpleDateFormat("YMMDD");
                                                Date dateobj = new Date();
                                        System.out.println("the reservationNo is :"+df.format(dateobj)
+""+numberconverted);

                                        combine=df.format(dateobj)+numberconverted; //reservationno
                                        x.addcCustomerInformation(combine);
                                        resultofbooking='S';
                                        }
                              }
                    }
 else
          {
               System.out.print("wrong word!");
               System.exit(0);
          }
 return resultofbooking;
}

 }
```

## User Interface Design:

**Entity-Relationship diagram:**

## Appendix A: Individual contribution to the assignment

**Fill in and submit this form with the last part of the assignment in week 13.**

The group mark is multiplied by an individual weighting to calculate the individual mark.

One way to allocate marks is to give each member of the group an initial weighting of 100. If a member of the group has contributed more than the others, then that person's weight is increased and the weights on the other members are decreased so the total weight is always 100*n, where n is the number of people in the group (normally three). No person can score more than 60 marks for the assignment; any marks above 60 will be ignored.

The following table **must be filled in and signed by every member of the group**, and submitted with the final part of the solution **in week 13**. No individual mark will be given until this form has been signed and submitted.

## Group number:

| **Student id, name** | **Weight** | **Signature** |
|---|---|---|

| | | |
|---|---|---|
| | | |
| | | |
| | | |