



**Bilkent University**

Department Of Computer Engineering

# **CS 319 Project :**

## **Analysis and Design Report**

*AutoHotel*

*Hotel Automation Software System*

**Group 3-7**

İrem Hergüner	21201077
Halit Onur Karabaş	20901806
Çağlar Alim Ağlar	21101255

Can Alkan

CS319

Object Oriented Software Engineering

# **Contents**

<b>1. Introduction.....</b>	<b>4</b>
<b>2. Requirement Analysis.....</b>	<b>5</b>
2.1. Overview.....	5
2.2. Functional Requirements.....	5
2.3. Non-Functional Requirements.....	6
2.4. Constraints.....	7
2.5. Scenarios.....	7
2.6. Use Case Model.....	12
2.7. User Interface.....	14
<b>3. Analysis.....</b>	<b>23</b>
3.1. Object Model.....	23
3.1.1. Domain Lexicon.....	23
3.1.2. Class Diagram.....	24
3.2. Dynamic Models.....	26
3.2.1. State Chart.....	26
3.2.2. Sequence Diagram.....	30
<b>4. Design.....</b>	<b>35</b>
4.1 Design Goals.....	35
4.2 Sub-System Decomposition.....	38
4.3 Architectural Patterns.....	41
4.4 Hardware/Software Mapping.....	42
4.5 Addressing Key Concerns.....	43
4.5.1 Persistent Data Management.....	43
4.5.2 Access Control and Security .....	44
4.5.3 Global Software Control.....	44
4.5.4 Boundary Conditions.....	44
<b>5. Conclusion.....</b>	<b>46</b>

## **List Of Figures**

Table 1 - Receptionist scenario

Table 2- Room Keeper scenario

Table 3- Accountant scenario

Table 4- Manager scenario

Table 5- Kitchen Chef scenario

Table 6- Bartender scenario

Table 7- Waiter scenario

Table 8- Security scenario

Figure 1- Use case Diagram

Figure 2- Opening Page

Figure 3- Receptionist Interface

Figure 4- Room Reservation Page

Figure 5- Room Information Page

Figure 6- Meal Plan Page

Figure 7- View of Room Status

Figure 8- View of Employee Information

Figure 9- View of Economic Status

Figure 10- Accountant Interface

Figure 11- Kitchen Chef Interface

Figure 12- Room Keeper Interface

Figure 13- Bartender Interface

Figure 14- Waiter Interface

Figure 15- Security Interface

Figure 16- Class Diagram

Figure 17- State Chart Diagram for Receptionist

Figure 18- State Chart Diagram for Manager

Figure 19- State Chart Diagram for Accountant

Figure 20- State Chart Diagram for Bartender

Figure 21- State Chart Diagram for Kitchen Chef

Figure 22- State Chart Diagram for Room Keeper

Figure 23- State Chart Diagram for Security

Figure 24- State Chart Diagram for Waiter

Figure 25- Sequence Diagram for Receptionist

Figure 26- Sequence Diagram for Manager

Figure 27- Sequence Diagram for Accountant

Figure 28- Sequence Diagram for Room Keeper

Figure 29- Sequence Diagram for Kitchen Chef

Figure 30- Sequence Diagram for Bartender

Figure 31- Sequence Diagram for Waiter

Figure 32- Sequence Diagram for Security

Figure 33- Diagram for the Subsystem Decomposition

Figure 34- Layers of System

Figure 35- Deployment Diagram

# **Analysis and Design Report**

*AutoHotel : Hotel Automation Software System*

## **1) Introduction**

This project is an automation software system for the use of hotel employees. We are designing this system in order to be a desktop application on hotel's computers. With this project, each employee can see and set the information of rooms and customers according to their job description in order to maintain their works efficiently. So, the users of this software system is hotel employees, not customers. All adjustments are made by considering that every employee has access to a computer in order to enter informations according to their jobs into the system. With this system, hotels do not have any missing and incorrect document about operation of hotel in order to eliminate fraud. Besides this, AutoHotel can keep and store all information about hotel in one place.

In this project, we aimed to create an automation system that all workers can communicate easily and efficiently between themselves. For this reason, we will try to maintain a system which is able to use easily by all workers without any training. However, to avoid any possible problem about software system, there will be help page in order to guide employees about their abilities of access, usage manual of system and information about handling possible situations. For these reasons, our program has functions to provide every need of an hotel by reducing the complexity of the program and providing a user friendly environment.

## **2) Requirement Analysis**

### **2.1 Overview**

There are workers in hotels which are mainly receptionists, room keepers, managers, accountants and kitchen chef. Receptionist can make a reservation with the help of this software, managers can control operations, accountants can calculate economic issues, room keepers can see dirty and clean rooms and chef in kitchen can see order of customers that comes from room service. This system aims to ease the communication among hotel employees and improve efficiency of service.

### **2.2 Functional Requirements**

- Users of this program are hotel employees, not customers.
- This program will be designed with considering the all employees have computer access to make changes on rooms and customers information.
- Each user types which are receptionist, manager, accountant, room keeper and kitchen chef have their own interface according to their worker ID and password on opening page.
- Receptionist will be able to access customer and room informations, make and cancel reservations depending on capacity of hotel.
- Manager will be able to access worker informations, set shifts, pay salaries and control the economic status.
- Accountant will be able to access calcualte workers salaries, pay hotel taxes, calculate economic status and report it to the manager and set currency.

- Room keeper will be able to access room informations that if they are empty, full, clean or dirty; get minibar status, laundry need and ordered room service.
- Kitchen chef will be able to access customers meal order detail to prepare them, update the their bill and manage the hotel meal plan.

## **2.3 Non-functional Requirements**

- System's user interface is designed according to employees types that means each employee can make changes only on their job related informations.
- System is user friendly that every employee -even not related with higher technology- can easily manage.
- We will use online databases to keep the information of customer, hotel, room and worker.
- System defines its technical details which are ease of use, ease of learning, extensibility, portability, modifiability and consistency in design goal part

### **2.3.1 User-friendly Interface**

AutoHotel system is user friendly in many ways:

- The program is easy to install since it is desktop application and do not need any external database system. It can keep informations in text files.
- The system is intuitive that it can be work well with good GUI.
- Software is efficient in a way that users can complete their job, features are well located, not maste work time while manage the system.
- It is a menu-driven programs, employees are able to find their ways with toolbars unlikely than command-driven programs.

### **2.3.2 Extensibility**

Extensibility means system ability about future growth according to requirements. Good extensibility feature of software system can be measured with effort while adding new functionalities to software.

Our project supports the white-box extensibility that modified with source code. It is flexible and less restrictive. This project is an object-oriented software that supports inheritance and dynamic binding. So with extensions, original system cannot be affected by changings.

As an example, during extensions developers can add new worker types without changing original source code because there are inheritance between worker class and worker types which are receptionist, manager, accountant, room keeper and kitchen chef. In order to add new worker type, the extension does not affect these classes under favour of object-oriented software structure and specifically glass-box extensibility.

## **2.4 Pseudo Requirements**

- The program will be implemented in Java.
- This program will implement for desktop computers.

## **2.5 Scenarios**

There will be different scenarios for different users. There will be six different type of user which are receptionist, cleaner, manager, accountant and kitchen workers. Our program will detect users by their login information. These users will be have usernames and passwords, and these usernames will contain the information about their job description. Therefore, these scenarios show us how can these different users use this program depend on their job description.



### 2.5.1 Scenarios for Receptionists

**Participating Actor :** Receptionist

**Entry Condition :** Actor logins with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will see all the rooms and detailed information about them
2. Actor will push Check-In button for customers
3. Actor will enter informations about customer and room if customer does not have reservation
4. Actor will push Check-out button for leaving customer

**Alternative Flow of Events:**

1. Actor will push Make Reservation button for customers who wants to reserve room and enter customers information
2. Actor will choose Menu bar to see the information about meals for workers and customers
3. Actor will choose Room Service bar to enter customers' room services information in order to send Kitchen Chef.

Table 1: Receptionist Scenario

### 2.5.2 Scenarios for Room Keeper

**Participating Actor :** Room Keeper

**Entry Condition :** Actor logins with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will see all the rooms with detailed informations about rooms
2. Actor will enter Laundry price and update customer's bill if there is a laundry
3. Actor will fill minibar and update customer's bill if customer consume minibar

**Alternative Flow of Events:**

1. Actor will push Set Clean button after cleaning it and changes room status from dirty to clean

Table 2: Room Keeper Scenario

### 2.5.3 Scenarios for Accountant

**Participating Actor :** Accountant

**Entry Condition :** Actor logins with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will choose type of a worker
2. Choose an employee
3. Actor will see employee's informations and calculate his salary depends on his job and shifts.

**Alternative Flow of Events:**

1. Actor choose the Tax bar and calculate the tax to send Manager.
2. Actor choose the Economy bar and calculate the economical status of Hotel to send Manager.
3. Actor choose the Economy bar and set the currency

Table 3: Accountant Scenario

### 2.5.4 Scenarios for Manager

**Participating Actor :** Manager

**Entry Condition :** Actor logins with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will choose Employees bar and Actor will see Employees in order depending on their job type
2. Actor can pay employees' salary and set their shifts.
3. Actor will choose the Economy bar and see the detailed informations about hotel's economic status

**Alternative Flow of Events:**

1. Actor will choose a room
2. Actor will see detailed information about room and customer
3. Actor will see occupancy rate of hotel

Table 4: Manager Scenario

### 2.5.5 Scenarios for Kitchen Chef

**Participating Actor :** Kitchen Chef

**Entry Condition :** Actor logins with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will get meal orders automatically from reception or waiter
2. Actor will see information about meal
3. Actor will send order and enter information

**Alternative Flow of Events:**

1. Actor will choose Meal list bar to create meal list for workers and guests

Table 5: Kitchen Chef Scenario

### 2.5.6 Scenarios for Bartender

**Participating Actor :** Bartender

**Entry Condition :** Actor logins with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will get beverage order automatically from reception, waiter or himself
2. Actor will see information about beverage
3. Actor will send order and enter information

**Alternative Flow of Events:**

1. Actor choose Beverage List bar to create beverage list for guests

Table 6: Bartender Scenario

### 2.5.7 Scenarios for Waiter

**Participating Actor :** Waiter

**Entry Condition :** Actor logs in with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will see rooms and information if they have an order
2. Actor will create an order for a room and enter information

**Alternative Flow of Events:**

1. Actor will choose menu bar to give information about meals and beverages to guests

Table 7: Room Keeper Scenario

### 2.5.8 Scenarios for Security

**Participating Actor :** Security

**Entry Condition :** Actor logs in with his ID and password

**Exit Condition :** Actor pushes Logout Button

**Main Flow of Events:**

1. Actor will enter informations about left luggage
2. Actor will delete a left luggage from list

**Alternative Flow of Events:**

1. Actor will choose the Contact Police bar to inform police

Table 8: Security Scenario

## **2.6 Use-Case Model**

In AutoHotel Project, there will be eight different users of the program and they have different screens according to their login informations and inputs about customers, hotel and employees. We use this use case model for requirements elicitation and analysis to represent the functionality of hotel automation system. The diagram represents the behaviour of the project from external view which is actor's view. This diagram also clarify boundary of the system. The actors which receptionist, security, bartender, waiter, accountant, manager, room keeper and kitchen chef are outside of the boundary in diagram. And use cases are inside the boundary. Each rectangle defines subsystems and dashed arrows defines the interactions between them.

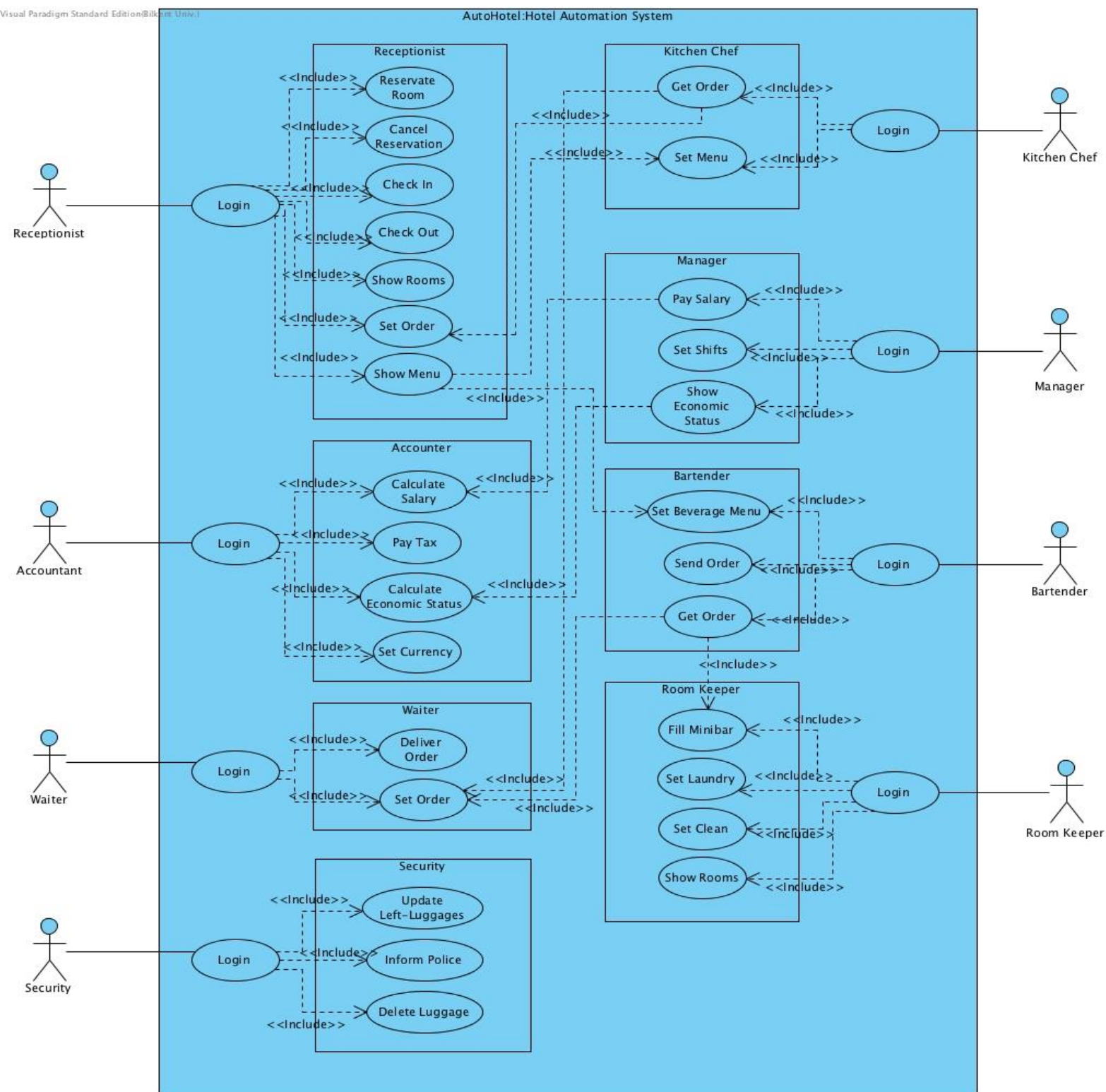
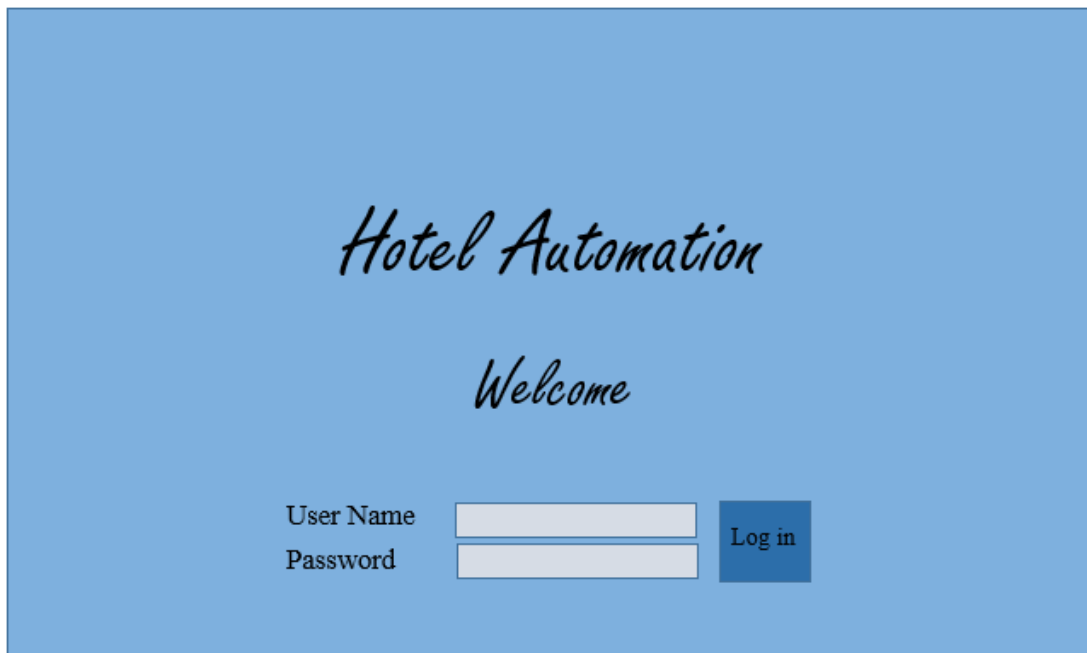


Figure 1: Use case model of whole system

## 2.7 User Interface

In this section, there are user interface prototypes. They do not provide much information about functionality and source codes but represent interfaces for most users. When the application runs, all users will see the welcoming screen with two text fields for username and password, and a login button. After login, the system



direct the users to different pages according to their authorisation.

Figure 2: Opening Page

### 2.7.1 Receptionist Interface

Receptionists will be directed to the screen below. This is main control screen of receptionist. In the Rooms tab, a receptionist can see informations about all the rooms. To make an easier access to different groups of rooms, there are four buttons as follows, all rooms, reserved rooms, occupied rooms and empty rooms. Receptionists can see informations about each room, and with 'check in', 'check out' and 'make reservation' buttons, they can change the status of the selected room according to customer request. Only the receptionist has the rights to change

reservations.

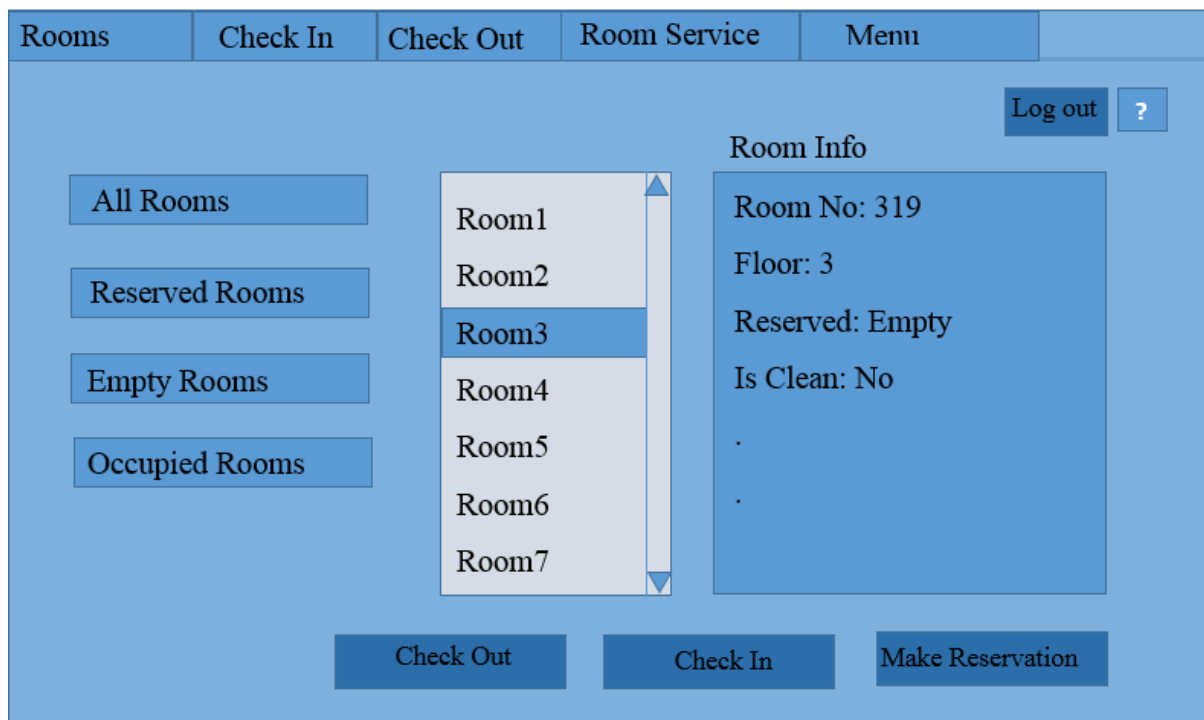


Figure 3: Receptionist Interface

In 'Check in' tab, customer check in can be made by the receptionists. Receptionists will enter the customer info, and choose a room type depends on customer's request. According to chosen room type, list of available rooms will be displayed on the screen. Receptionist will be able to choose a room from the list, and by clicking the 'make reservation' button, receptionist can take the customer's information and store them into the system.

Here is a reservation page that customer's information is stored. Customer informations are needed to provide a better hotel service. Beside this, if customer reenter the system in further dates, system can remember customer information to ease reservation. In 'Check Out' tab, customer check out will be made by the receptionists.



Rooms	Check In	Check Out	Room Service	
Name <input type="text"/> Surname <input type="text"/> TC <input type="text"/> Gender <input type="text"/> Birth Date <input type="text"/> Over Night Stay <input type="text"/> Check in Date <input type="text"/> Check out Date <input type="text"/> Room Type <input type="text"/>			<div>Available Rooms</div> <div>Room1 Room2 <b>Room3</b> Room4 Room5 Room6 Room7</div>	<div>Room Info</div> <div>Room3 Third Floor For two Clean . .</div> <div>Make Reservation</div>

Figure 4: Room Reservation Interface

Receptionists will be able to search according to customer name or room number. After search, customer info and room info will be displayed on the screen. By clicking the 'checkout' button receptionist is able to end the checkout process.

Rooms	Check In	Check Out	Room Service
<div>Search with Name <input type="text"/></div> <div>Room Number <input type="text"/></div> <div>Search</div>			<div>Log out</div>
<div> Customer :  Room :  Check in Date :  Check out Date :  Over Night Stay :  Total Charge : </div> <div>Check out</div>			

Figure 5: Room Information Page

In the menu tab, receptionists will only see the daily menu. Receptionist cannot change meal plan, it is kitchen chef's duty. They will just inform the customers about the menu if it's needed.

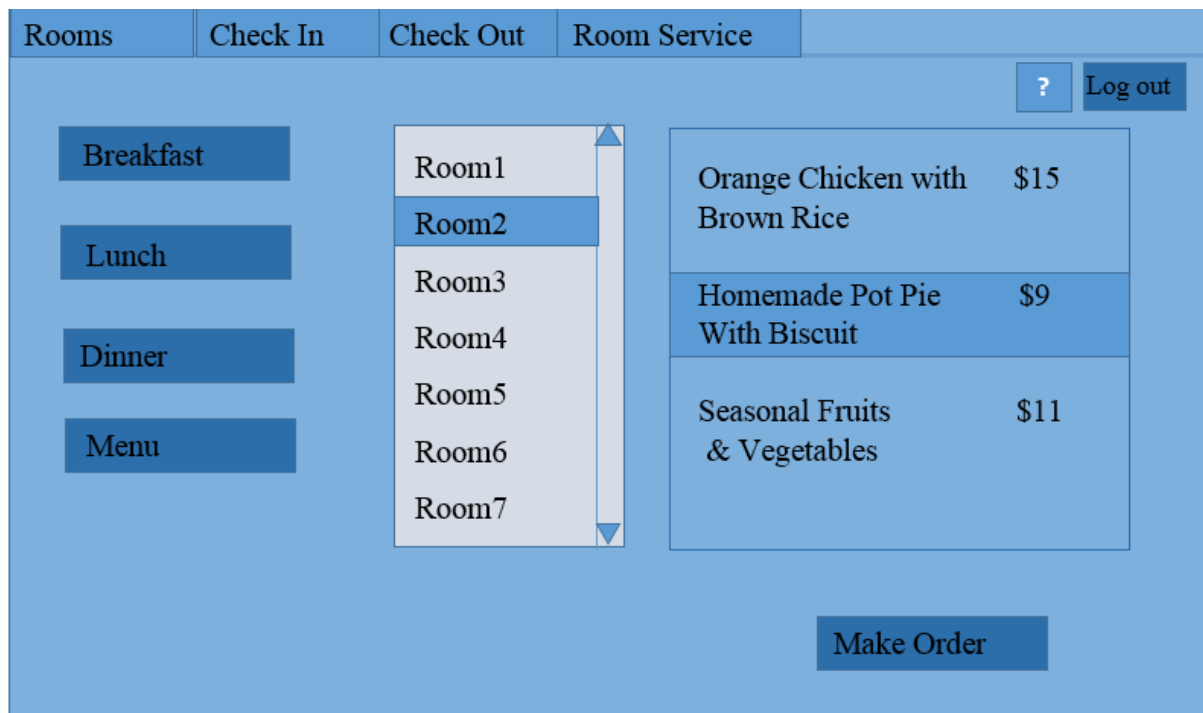


Figure 6: View of Meal Plan

### 2.7.2 Manager Interface

Manager will be directed to the screen below where he can see the general informations of the hotel and the rooms. The Hotel tab is similar with 'Rooms' tab where in the receptionist interface but the difference is that the manager can not change anything.

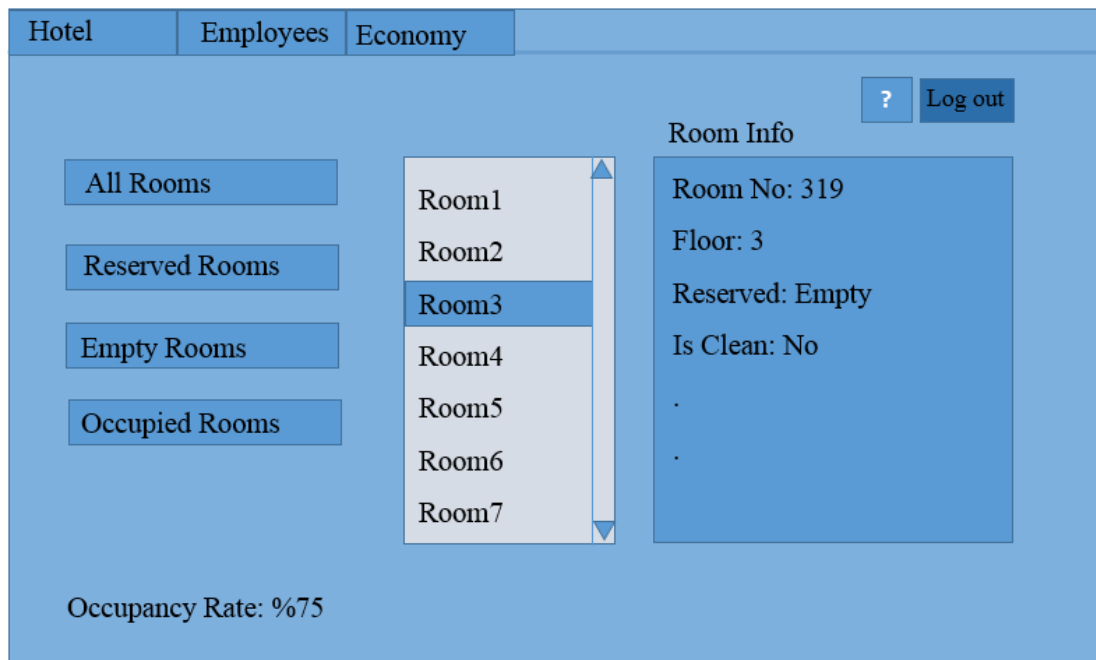


Figure 7: View of Room Status

In 'Employees' tab, manager is able to see employees from different departments. The manager can check the informations about the employees, set their shifts, and pay their salaries.

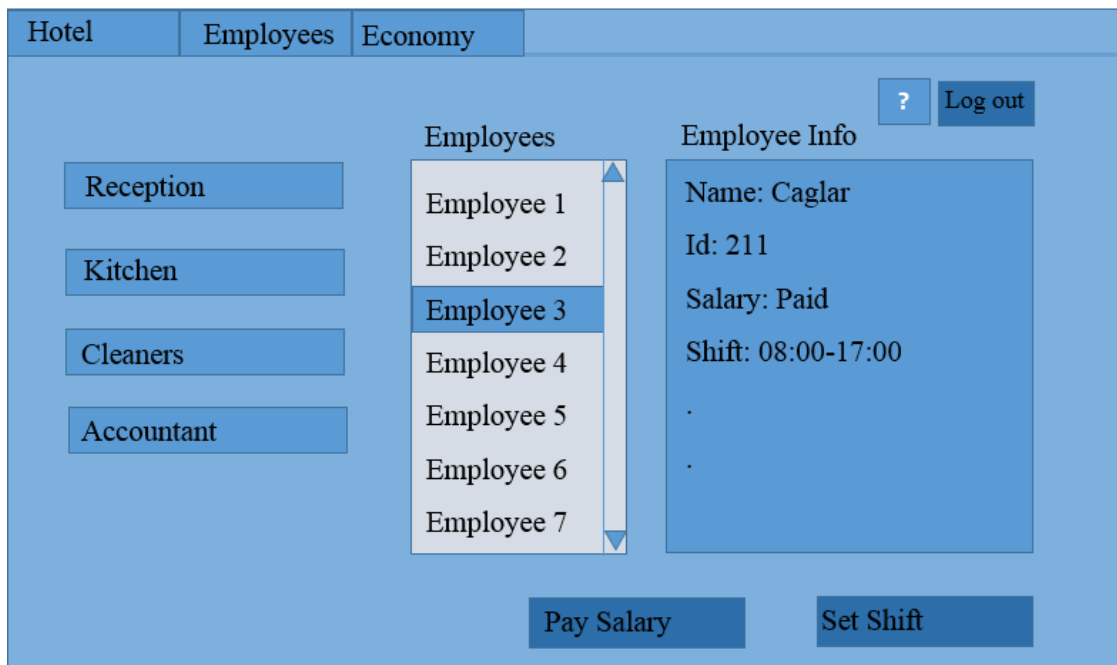
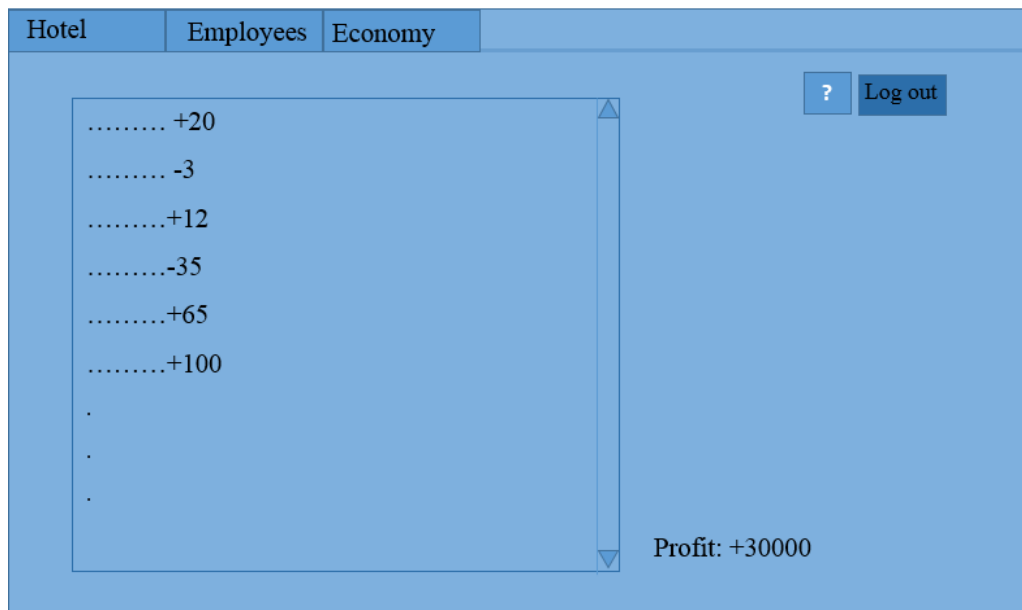


Figure 8: View of Employee Information

In the 'Economy' tab, the manager can see the economic status of the hotel,



which is organized by the accountant.

Figure 9: View of Economic Status

### 2.7.3 Accountant Interface

Accountant will see the screen below where he can see the employees from different departments and calculate their salaries according to their shifts.

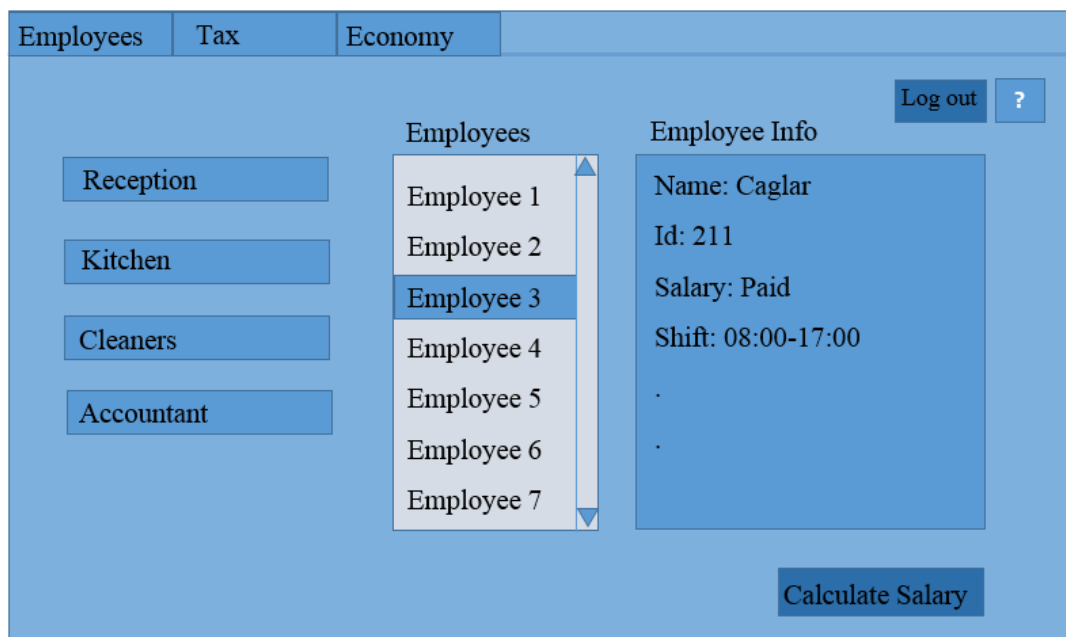


Figure 10: Accountant Interface

### 2.7.4 Kitchen Interface

Employees from the kitchen department will be directed to the screen below where they can enter the price of ordered meals after they prepare them. The meals are ordered by the customers as a room service.

Rooms Meal List

Room1  
Room2  
Room3  
Room4  
Room5  
Room6  
Room7

Room No: 319  
Floor: 3  
Food Order:  
Chicken with plums  
.  
.

Log out ?

Ok

Figure11: Kitchen Chef Interface

### 2.7.5 Room Keeper Interface

Employees from the cleaning department which are room keepers will be directed to the screen below where they can enter customers' laundry price, change the cleaning status of the room and change the packness status of the minibar.

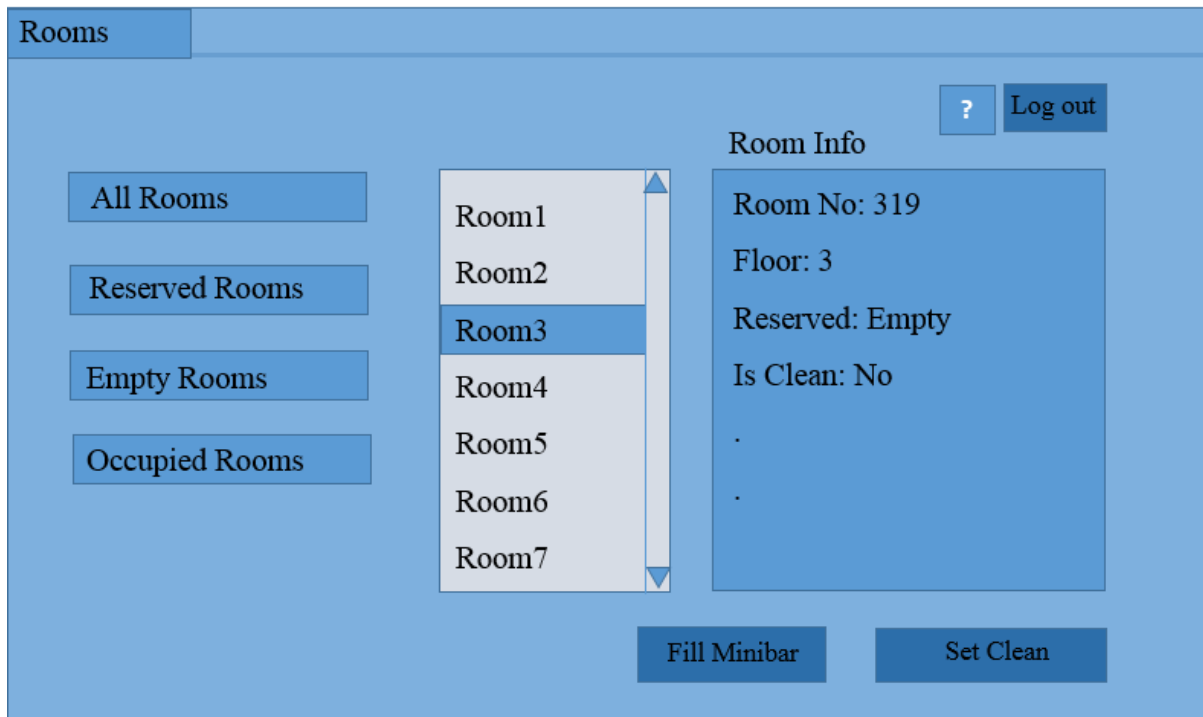


Figure12: Room Keeper Interface

### 2.7.6 Bartender Interface

Bartenders will be directed to the screen below where they will be able to see the customers' order and change the beverage list.

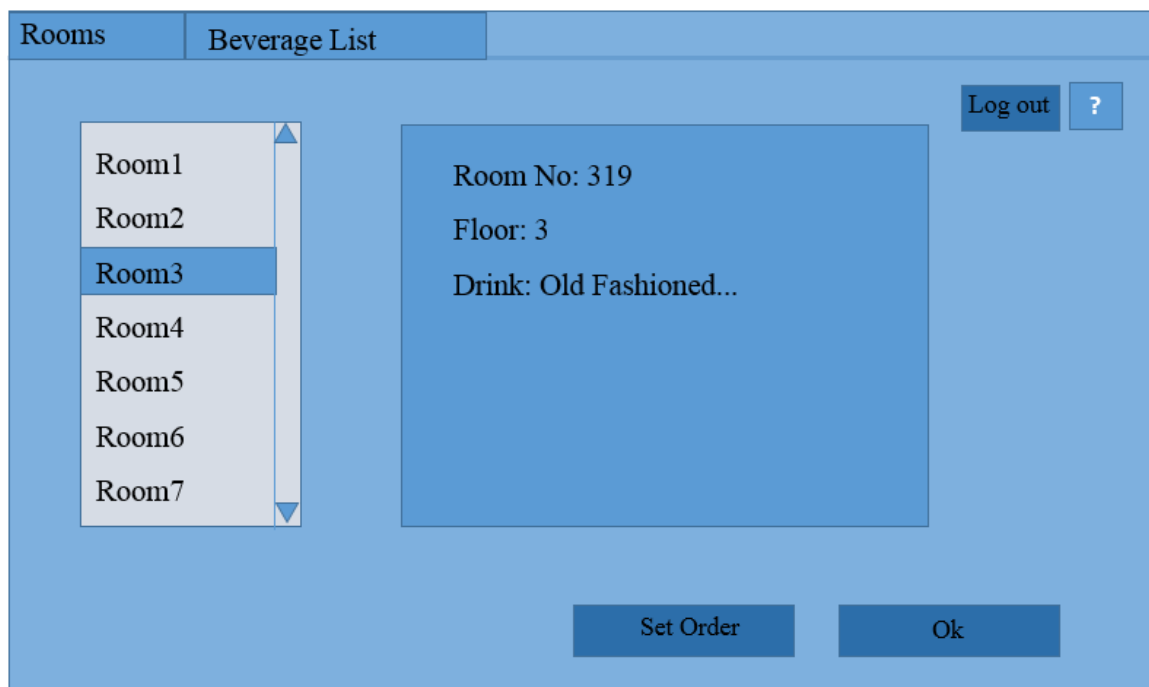


Figure13: Bartender Interface

### 2.7.7 Waiter Interface

Waiters will be directed to the screen below where they will be able to see the customers' order and set orders of the customers. They also will be able to see the Menu but they won't be able to change it.

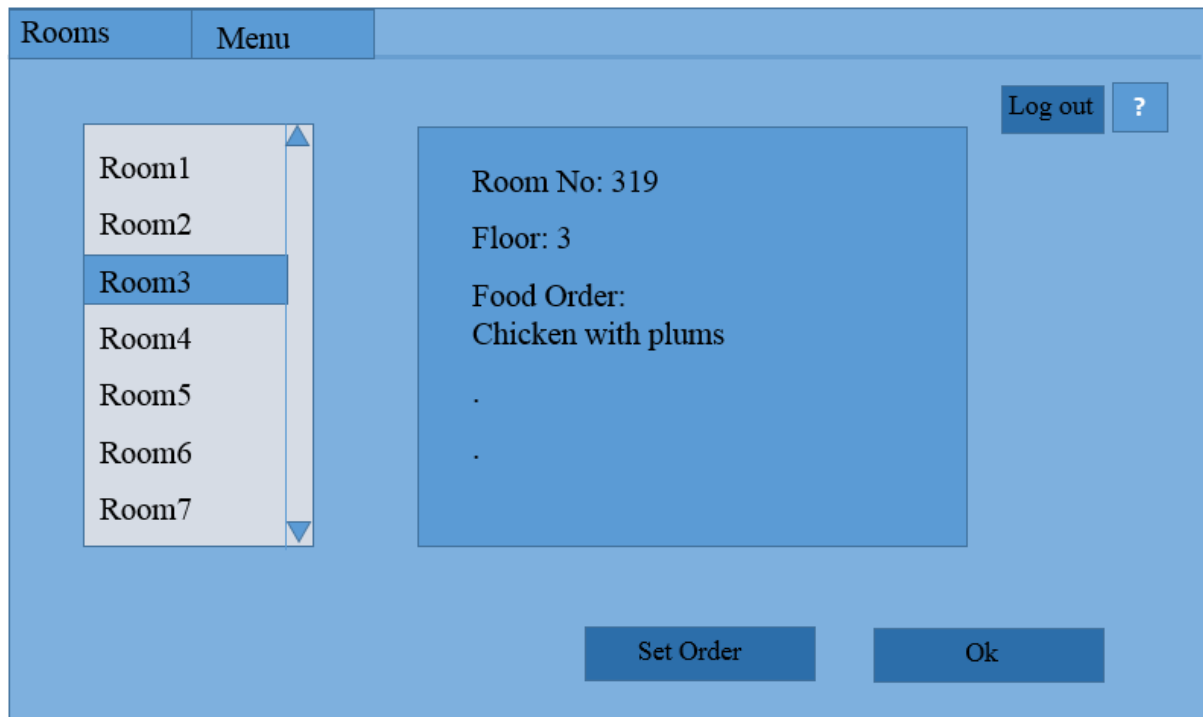


Figure 14: Waiter Interface

### 2.7.8 Security Interface

Workers from security department will be directed to the screen below where they will be able to add left luggage. They will be able to search and delete luggage.

The interface has a blue header with two tabs: "Left Luggage" (selected) and "Contact Police". In the top right corner, there is a help icon (?) and a "Log out" button.

On the left side, there is a "Search Luggage" section with a text input field and an "Ok" button. Below this is a form with five labels and corresponding input fields: "Name", "Location", "Guest", "Date", and "Founder".

On the right side, there is a "Luggage Info" section containing a list of details: "Suitcase", "Third Floor Room 301", "Customer X", "13.12.2015", and "Cleaner X".

At the bottom right, there are two buttons: "Delete Luggage" and "Add Left Luggage".

Figure 15: Security Interface

## 3) Analysis

### 3.1 Object Model

#### 3.1.1 Domain Lexicon

In AutoHotel system, "Shift" class identifies employees working period. It is separated into two which are early riser and working in evening.

"BeverageList" class identifies drinks which hotel have in its stocks and also it is used to list in drink menu.

Hotel have laundry services for customers if they want to clean up their clothes.

Hotel have left luggage service which stores the information of left luggages with found date when customers forget them in hotel.



### 3.1.2 Class Diagram

AutoHotel software system have 22 classes, 5 of them are enumeration class which have fixed attributes. Hotel class is main class that can be considered as controller class. It has interaction with all classes and BeverageList, MealList, Worker, Room and LeftLuggage classes are part of Hotel. Main incidents actualise on Room and Worker classes since all employee types are kind of Worker. Customer is part of Room because in hotel, there could not be a customer without room and other employees should not learn details about customer. Thus customer class is embedded into room class and not to attend class interaction too much.

Waiter, KitchenChef and Bartender are kind of KitchenWorker, they do not have common operations but their id will start with same special number according to KitchenWorkers to make operations more easily. We will give special starting numbers for each employee type in their id numbers to make search and retrieval operations easily.

In class diagram, nearly each class have set and get methods. Get method search and retrieval required information from database with its parameter. Set method make changes on databases with proper parameter.

HelpForEmployee class include texts for each employee that give information about details of their job, usage of system and what to do in the resulting possible errors.

The Room class do not have impact on worker class, it just stores customer and room information for hotel management.

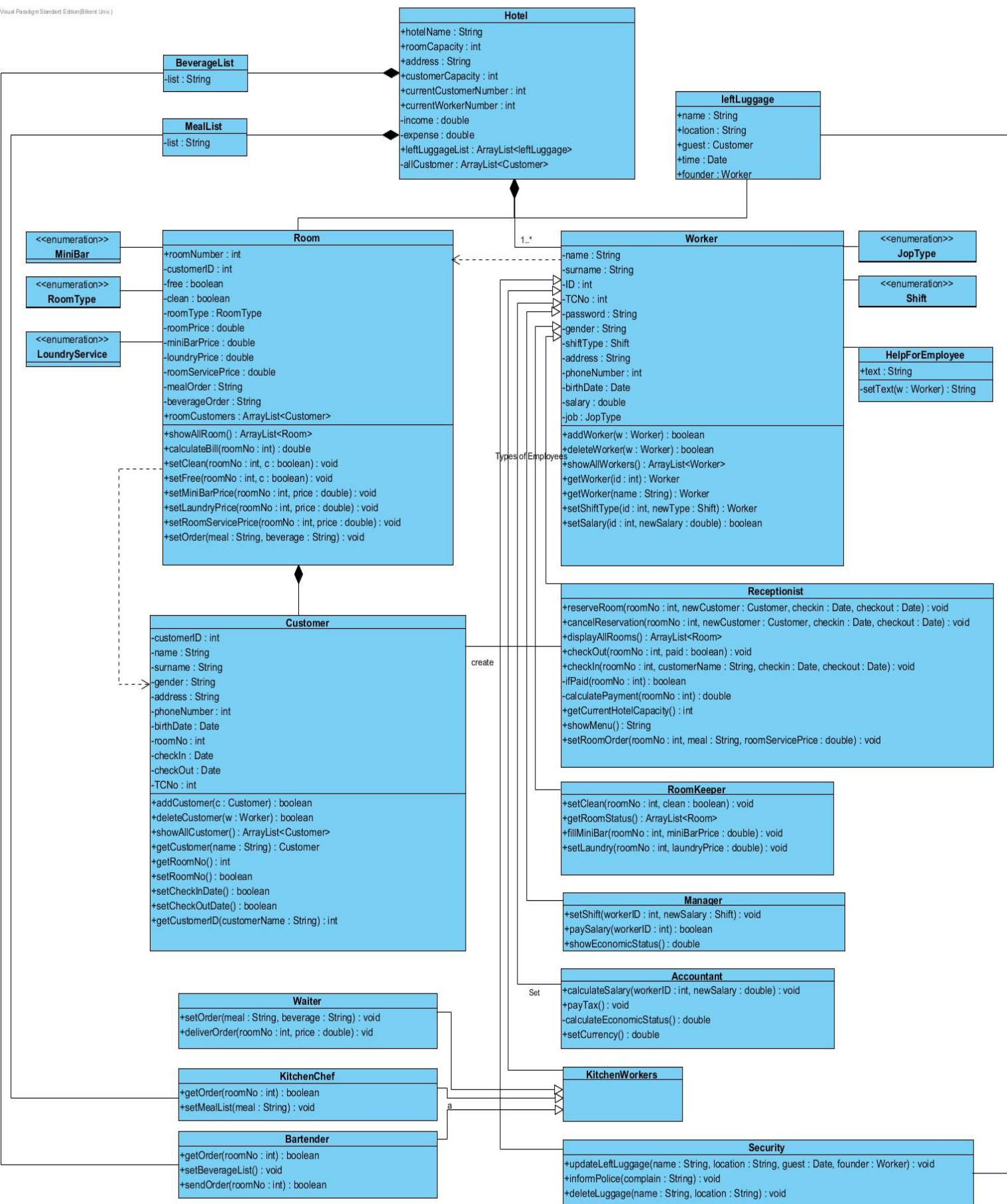


Figure 16: Class Diagram

## 3.2 Dynamic Models

### 3.2.1 State Chart Diagram

In this section, there are State Chart diagrams that are used to formalize the dynamic behaviour of individual objects in the system. They start with initial condition after login step and states of the objects are shown with the events that affect them.

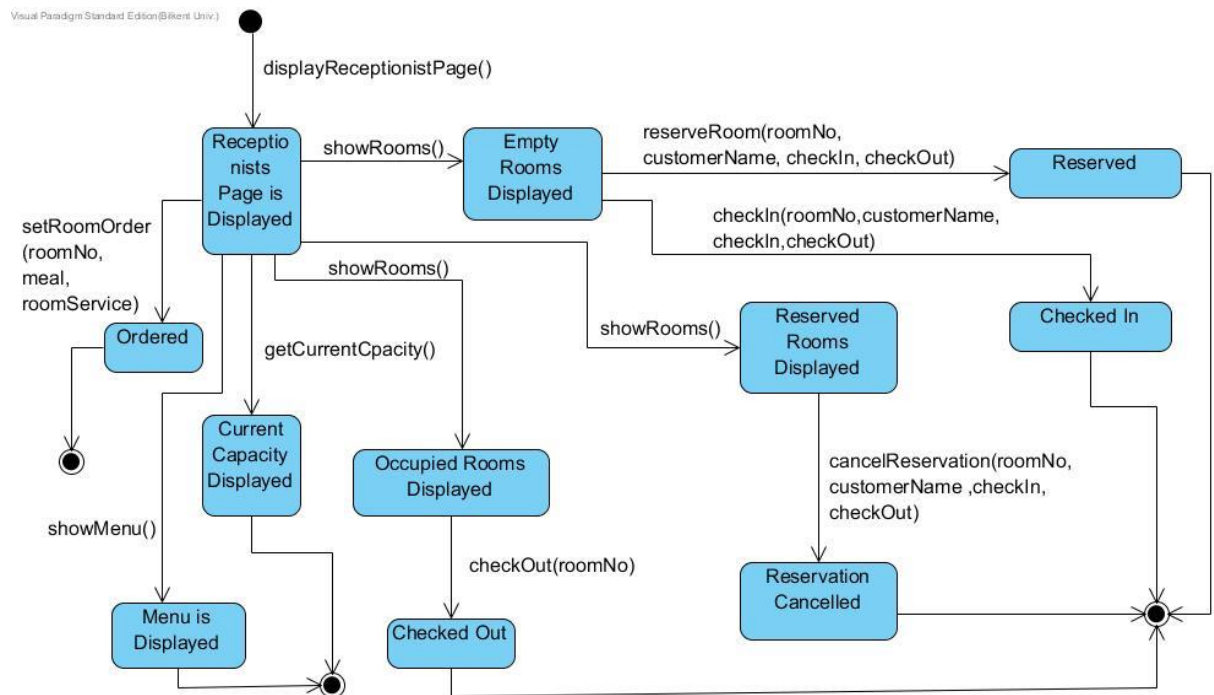


Figure 17: State chart diagram for Receptionist

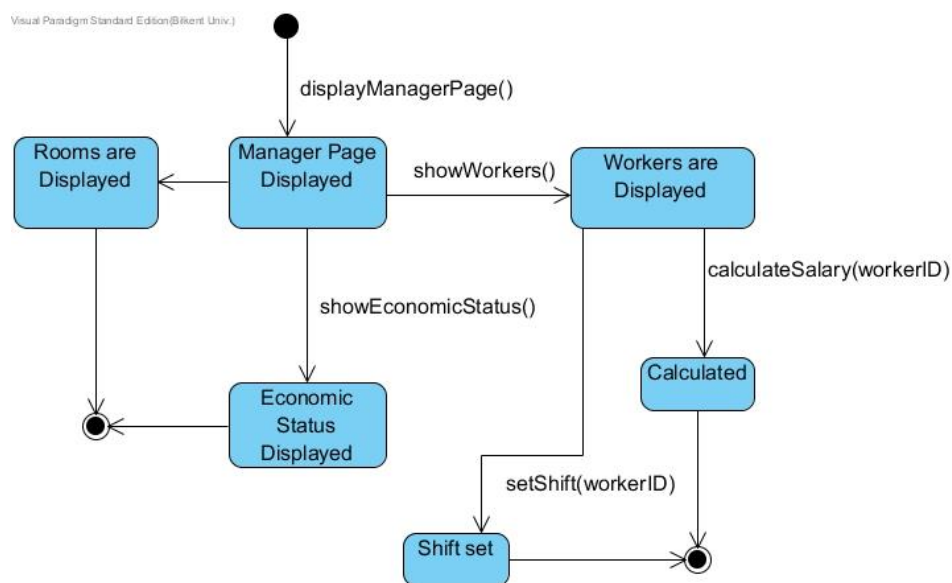


Figure 18: State chart diagram for Manager

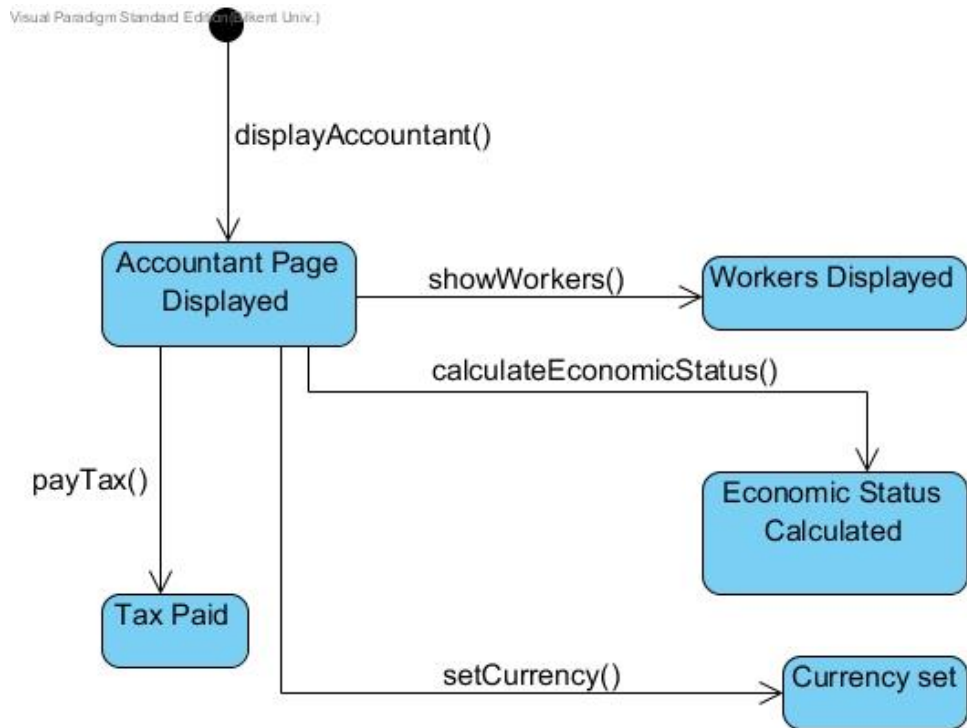


Figure 19: State chart diagram for Accountant

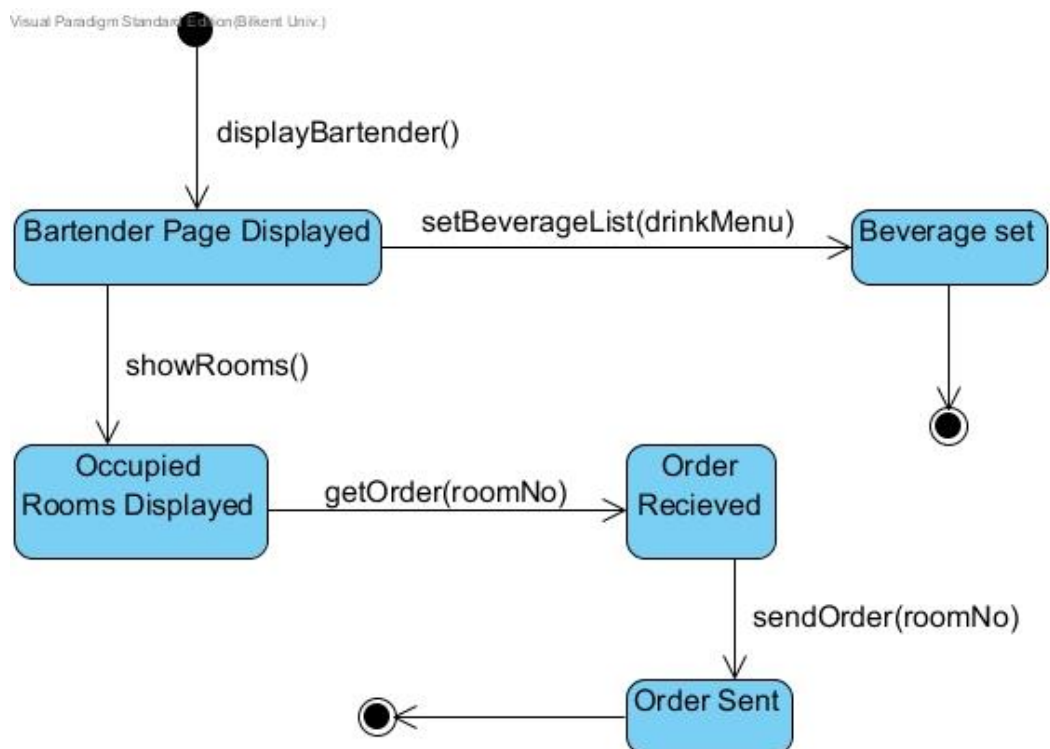


Figure 20: State chart diagram for Bartender

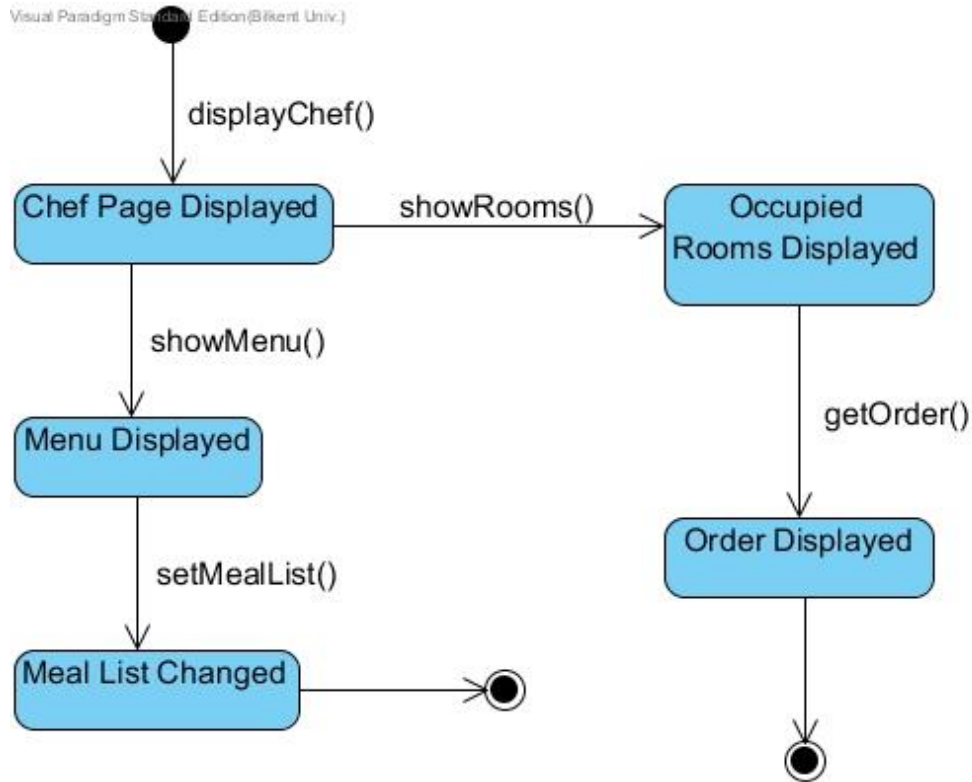


Figure 21: State chart diagram for Chef

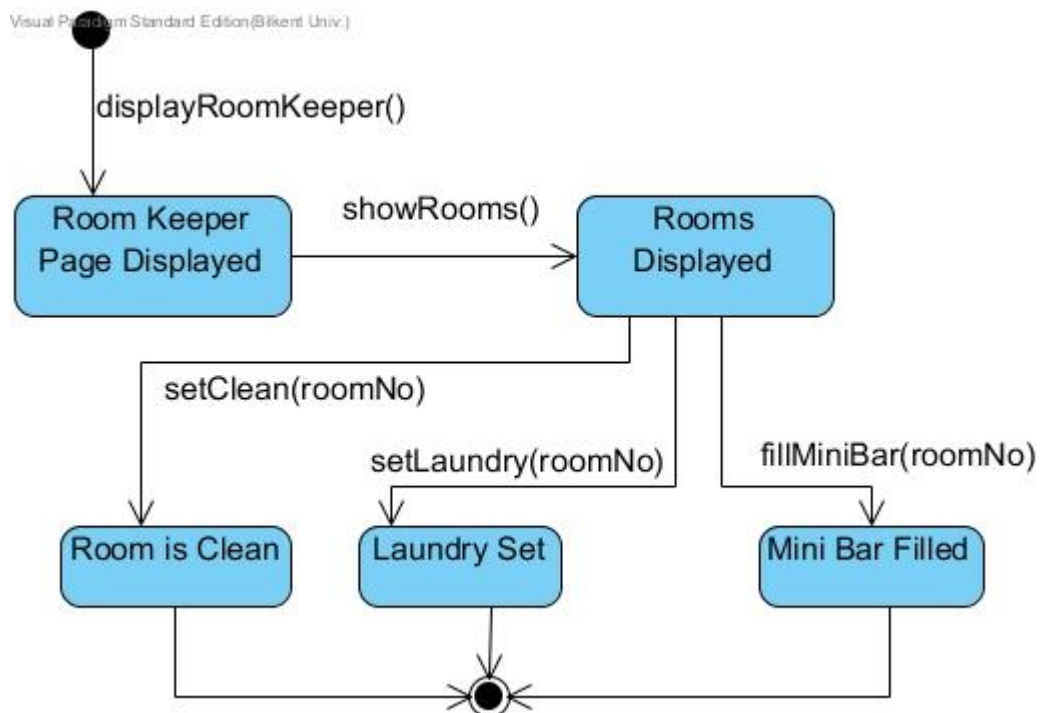


Figure 22: State chart diagram for Room Keeper

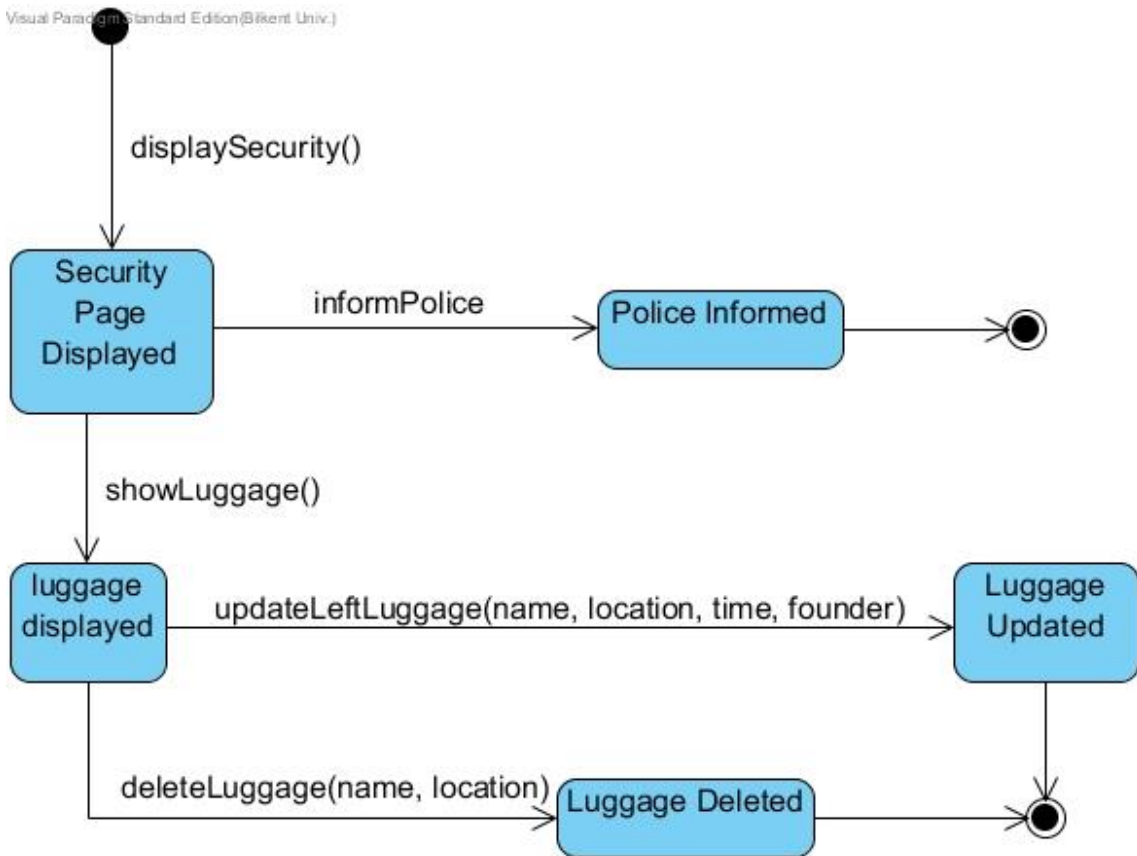


Figure 23: State chart diagram for Security

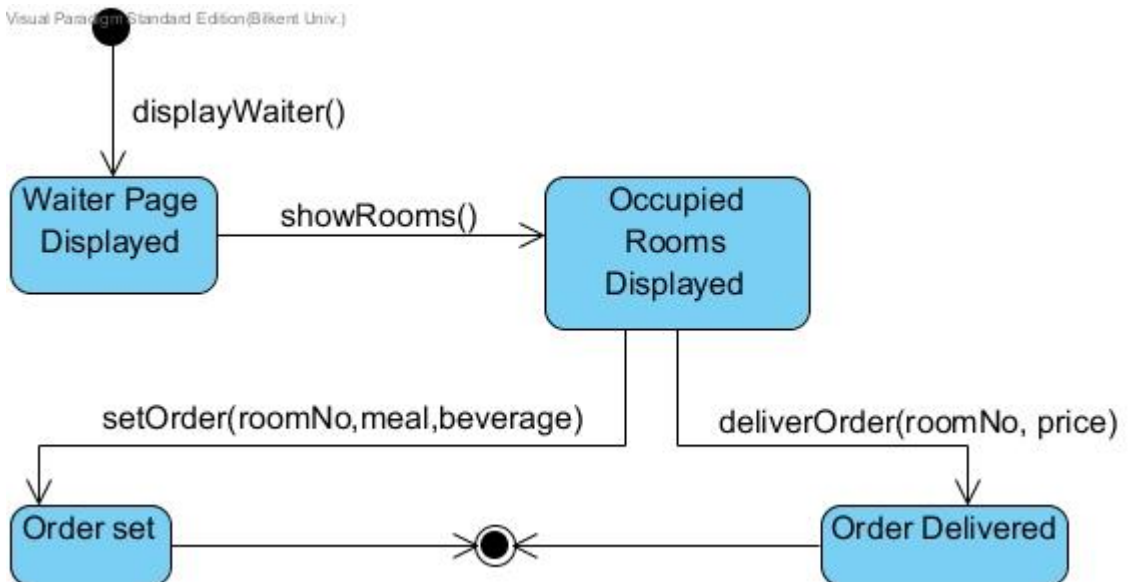


Figure 24: State chart diagram for Waiter

### **3.2.2 Sequence Diagram**

In this section, there are sequence diagrams that are used to formalize the dynamic behaviour of the system and to visualize the communication among objects.

In this diagrams, left most column represents the actor who initiates the use case.

Labeled arrows represents the functions that actor or object send s to other object which functions are shown more clearly in class diagram.

In receptionist sequence diagram there are 2 combined fragments that are for if - else conditions that activity change among them.



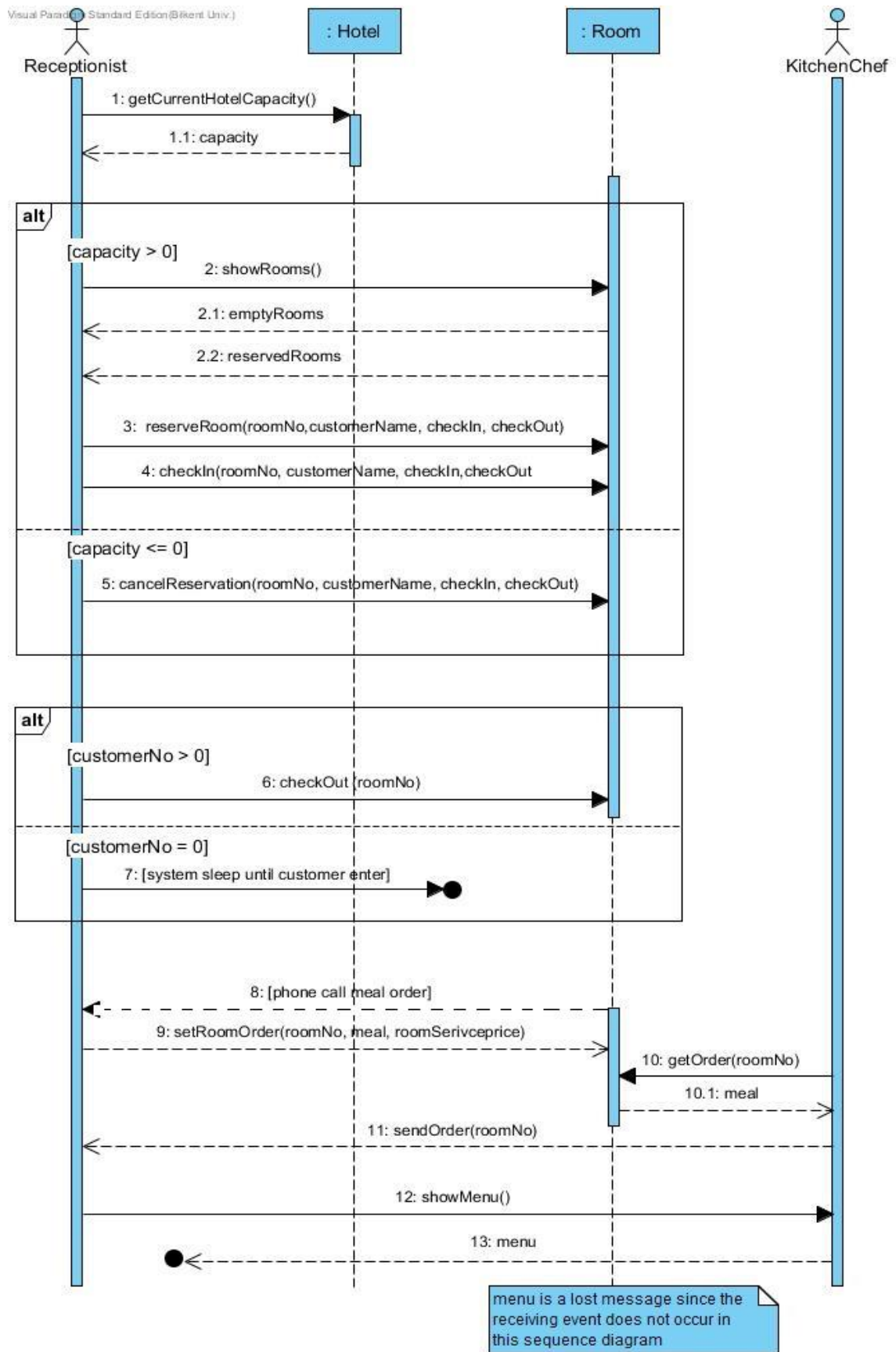


Figure 25: Sequence diagram for Receptionist



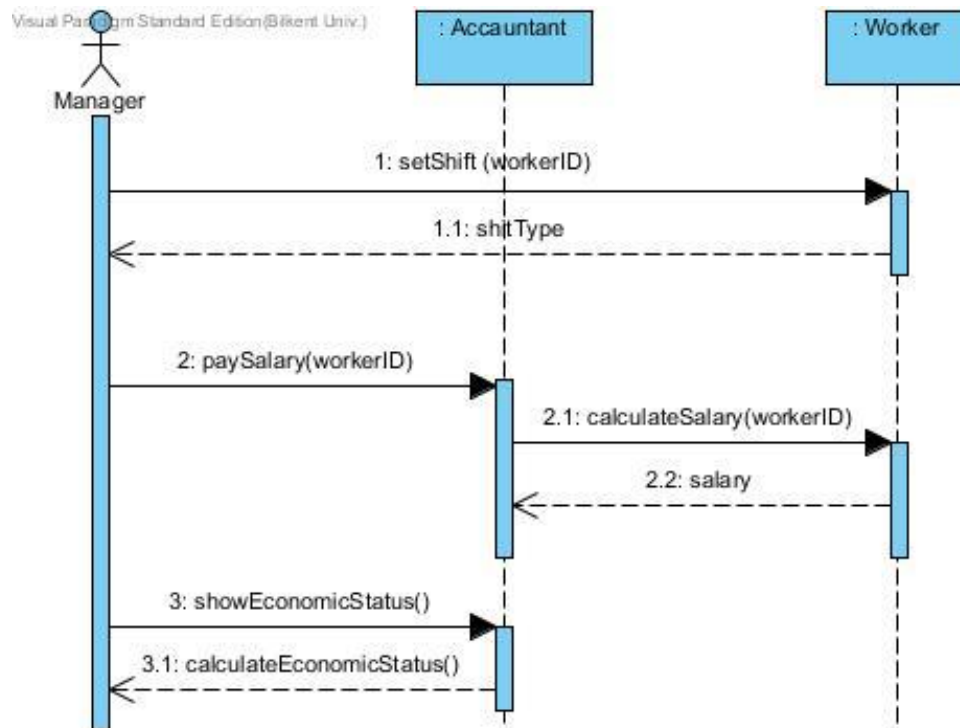


Figure 26: Sequence diagram for the Manager use case

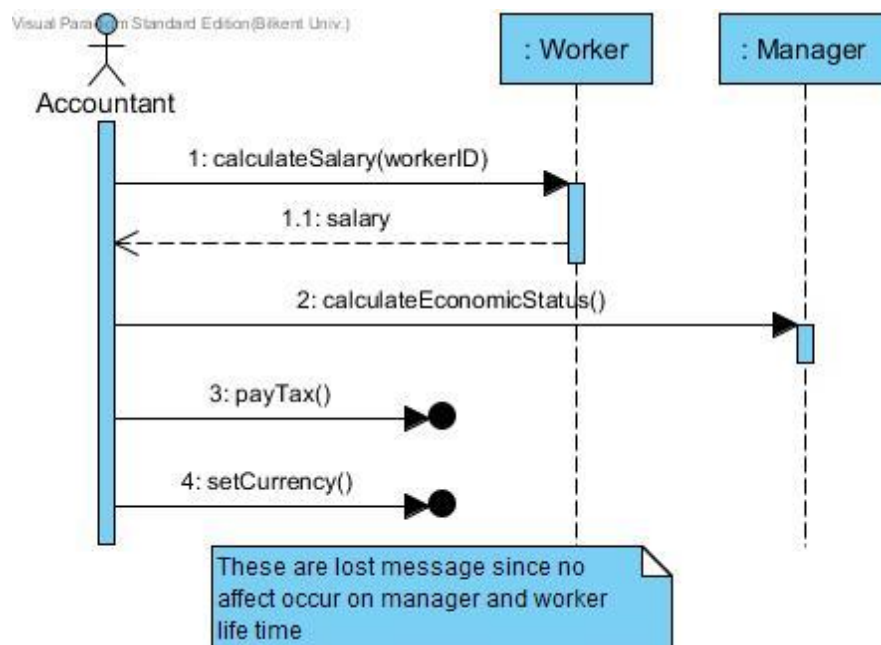


Figure 27: Sequence diagram for the Accountant use case

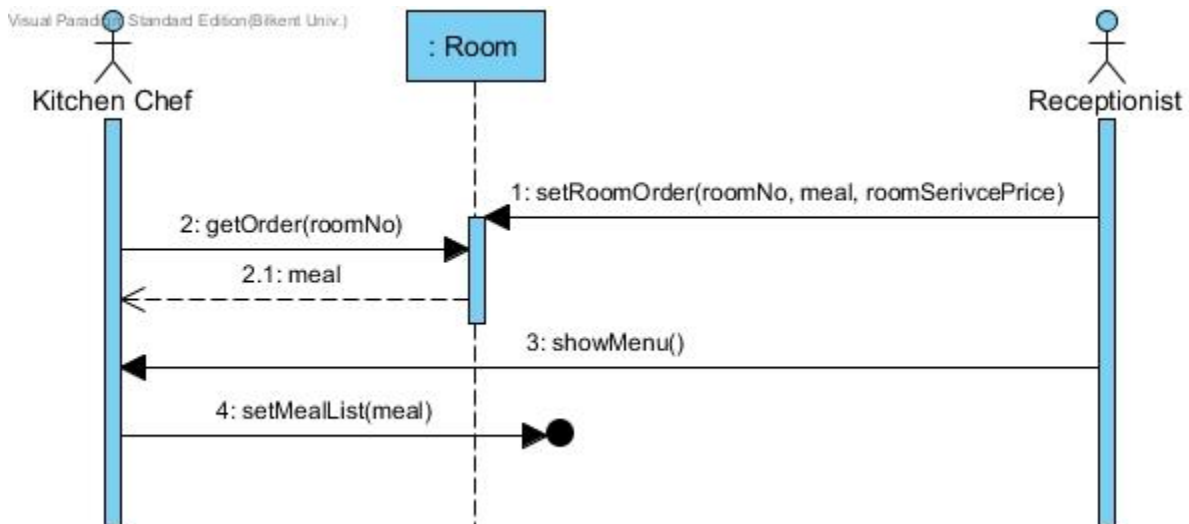
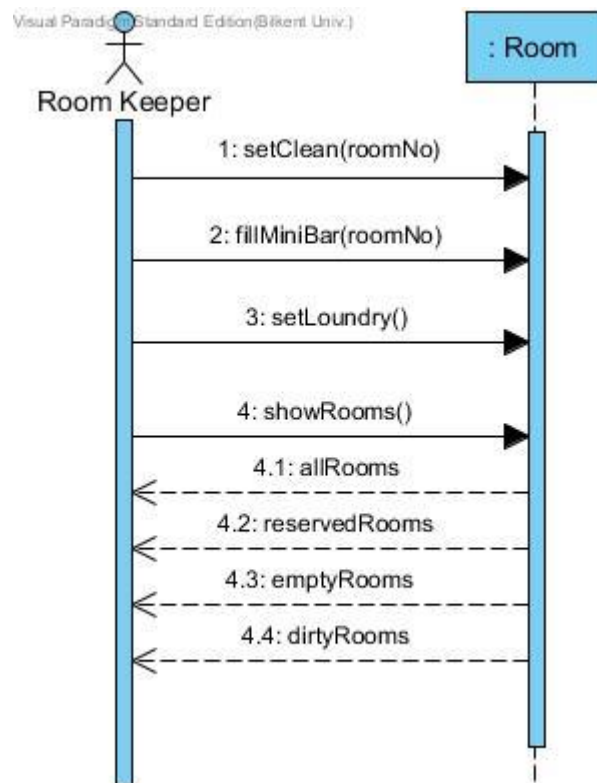


Figure 28: Sequence diagram for the Room Keeper use case

Figure 29: Sequence diagram for the Kitchen Chef use case

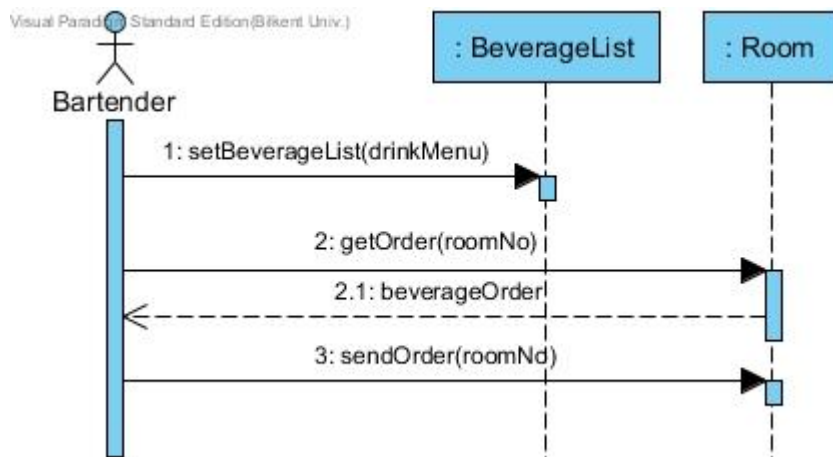


Figure 30: Sequence diagram for the Bartender use case

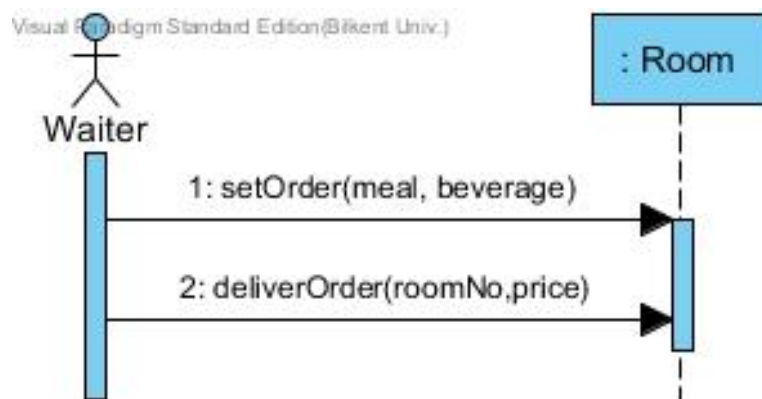


Figure 31: Sequence diagram for the Waiter use case

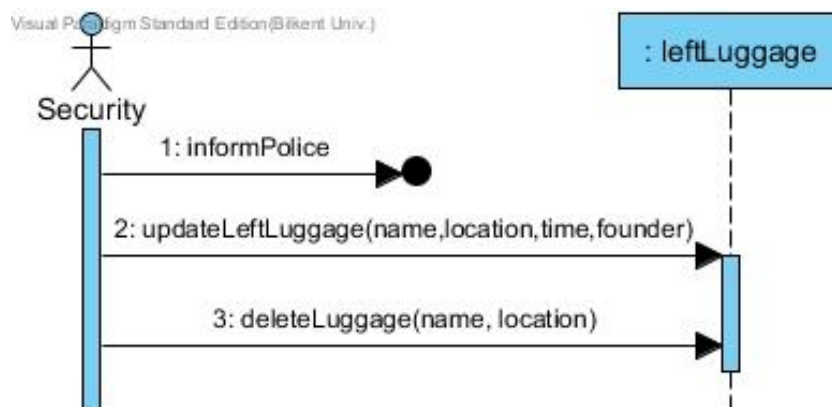


Figure 32: Sequence diagram for the Security use case

## **4) Design**

Auto Hotel is a hotel automation software system that keeps track of almost everything going on in the hotel. It provides several features for different employees, but in the system everything comes together to complete the all functionality of the hotel. With a simple and modest interface design, it provides an easy usage to the users. Auto Hotel aims to provide an easy and efficient way of controlling and keeping the track of all events happening in the hotel. Besides this, AutoHotel prevent any unregistered interaction about customers between employees. AutoHotel have 3-layer software architecture style that are presentation tier, business and data tier. Presentation tier is the user interface of employees see, business tier is control part of whole back-end system and data tier is online database that holds the hotel information in safety.

### **4.1 Design Goals**

During this design phase, several elements will impact and shape the design of the automation system. Some of these elements are typically non-negotiable or finite resources, such as time, money, and manpower. Others, such as available technologies, knowledge, and skill that we learned in department, are dynamic and will vary throughout the development of system. So before programing the application, it is very important to determine design goals, which are going to shape the application. Therefore, we build our project on non-functional requirements of our system, which is explained in the analysis report.

#### **4.1.1 End User Criteria:**

**Ease of Use:** Auto Hotel is a hotel automation software system, and because of that, it is designed to be used by many different departments. Each department has different interfaces and different data accesses. That means there are many buttons, text fields, combo boxes and data. But this means difficulty for non-technical people such as hotel employees. Therefore, Auto Hotel is designed to extinguish all this complexity and offer a simple and efficient control of data. For all users, reachable options are grouped under the meaningful names that are represented with different tabs. Under each tab, choices are clearly stated. To increase the readability and to make it user friendly, interfaces designed in a way that they do not contain too many colors and symbols. Besides this, AutoHotel offers specific layouts according to job type of employees.

**Ease of Learning:** Since the user does not have any knowledge about the AutoHotel, it is essential to obtain information about the use of the application. To inform and help the user, there are help buttons that provides information about the system and guides the user. Any user can easily use the AutoHotel by following the instructions provided by the help button. The information in help buttons also varies according to job type of employees.

#### **4.1.2 Maintenance Criteria:**

**Extensibility:** Hotels are dynamic places that offer new options to the customers. In time they have future growth, newer, or sometimes they get smaller. Therefore, to support and easily settle those changes, Auto Hotel should support the extensibility. AutoHotel is designed to support adding functionalities and changes in hotel features like new rooms, new workers, new departments, etc. It is suitable to

add new functionalities and modify existing structures because of its hierarchic structure. We try to aim minimize dependency between irrelevant classes and collect common things in interfaces and super classes.

**Portability:** AutoHotel is a desktop application that can be used in different environments. AutoHotel will be installed to the computers and the tablets in the hotel. It holds the information in online database so it does not require installation of database. For room keepers, it is required to use tablets because of transportation while visit the rooms. Therefore AutoHotel may be required to enhance.

**Modifiability:** It is important that future changes will be relatively easy to implement, since this decreases maintenance cost of software system. Corrections, improvements or adaptations of the AutoHotel to changes can be handled by 3-layer software architecture style and with extensibility feature.

**Consistency:** Consistency used for distributed system for shared data stores. The data consistency model specifies same data could be achieved and updated from different systems instantaneously. In our case, several employees take information from different computers at the same time about hotel. Therefore information should be achievable consistently. AutoHotel system eliminate this problem with using online database. When each user login the system, these datas will be loaded from database and whenever object is created, this information will be renew.

**Concurrency:** Many users can access data at the same time in AutoHotel software system and they can executable it simultaneously. Since our program is desktop application, each user can download and install this system into their computer to make changes simultaneously.

#### **4.1.3 Performance Criteria:**

**Response Time:** To provide a quick service to the customers of the hotel, Auto Hotel should respond the command very fast. Since different users access different data, and it is not a game that contains too much visual data to process, the application will respond to the user almost immediately.

#### **4.1.4 Trade Offs:**

**Ease Of Use and Ease of Learning vs. Functionality:** Since the hotels are crowded and there are too many things to be handled by different users, it is essential to make customers of the hotel happy. Therefore Auto Hotel is designed to be simple to avoid complex functionalities and increase the usability of the application, so that things should be done easily in a very short time of period for the customer satisfaction. In other words, priority of the usability is higher than the functionality.

**Performance vs. Memory:** In the hotels, there are many different departments and all these departments have too many data. As a matter of fact, hotel automation systems are based on data management. Auto Hotel, stores many important data such as customers' info, expenses, room info etc. Therefore the most important part of the system is keeping all the data safe. Auto Hotel saves big amount of data. Therefore it is expected to have a slightly slower application. In other words, memory is important than the high speed.

## **4.2 Subsystem Decomposition**

Under this title, we will try to show the collection of classes, associations, operations, events and constraints which are related themselves from different

layers. This decomposition is a type of 3-Tier Architecture and it consists of Interface, Application and Storage layers.

Actors will login and see the page according to their jobs as in shown Interface subsystem. Then, changes in user interfaces making by employees will manipulate databases through the objects on application subsystem. Therefore, we can realize that actors will interact with Interface layer, then the changes they made on this layer will make operations on object in the Application layer. Moreover, these operations will update Storage of the program and there will be 4 tables for storage datas which are customer, worker, room and hotel.



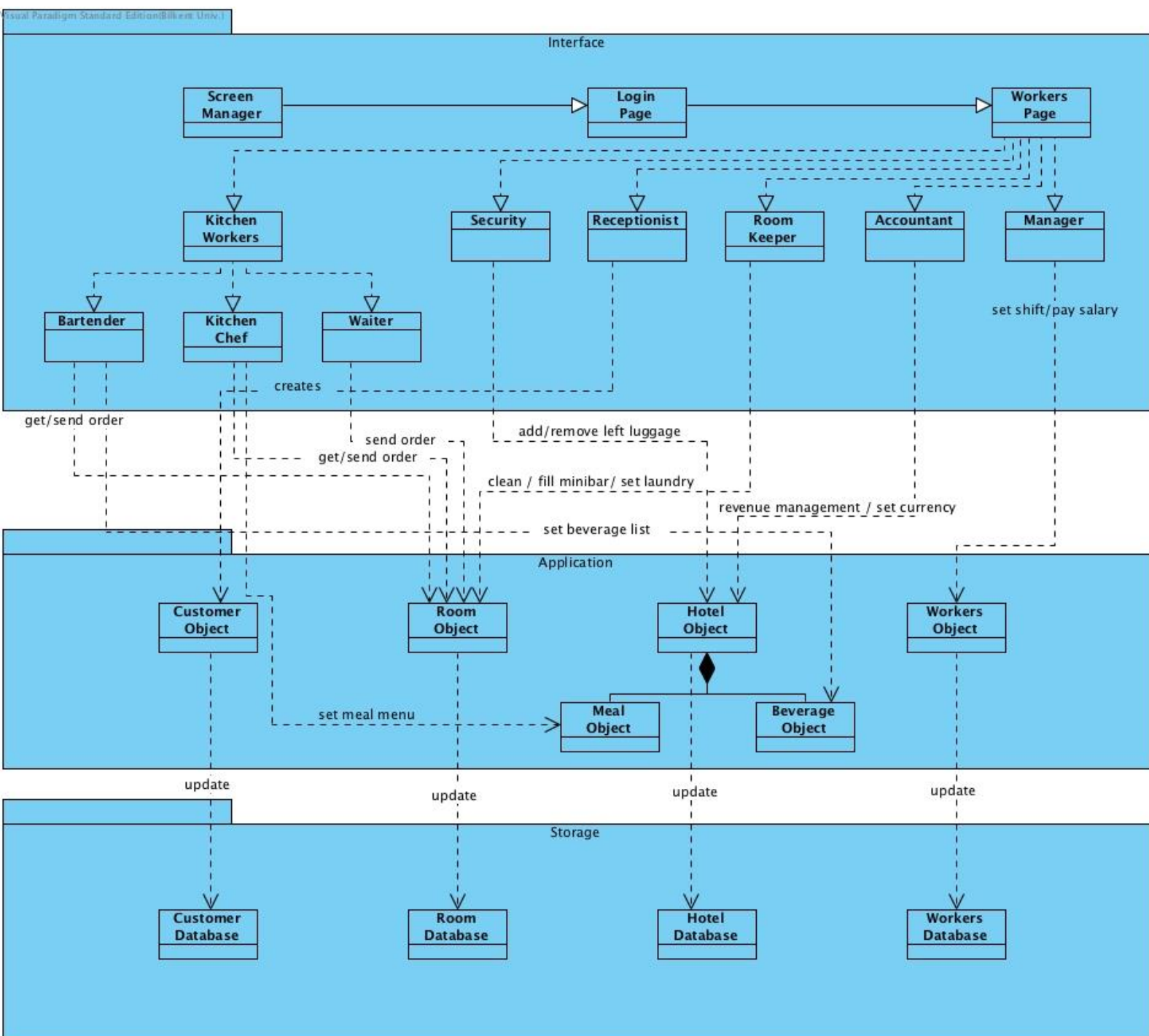
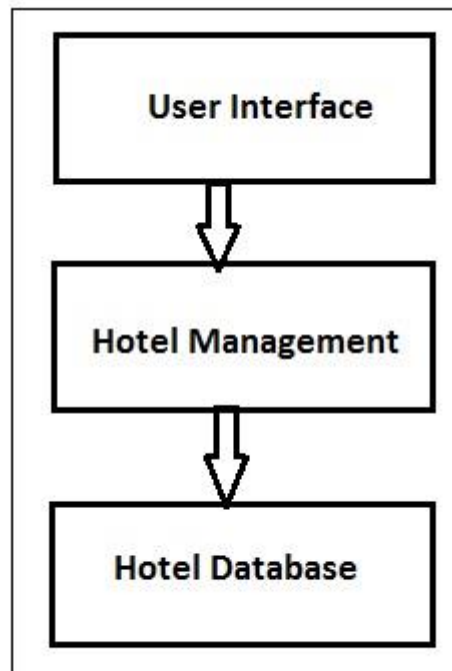


Figure 33: Diagram for the Subsystem Decomposition

## 4.3 Architectural Patterns

### 4.3.1 3-Layer Architecture Style

Our software system is 3-layer architecture. This system uses many layers for allocating the different responsibilities of a software subsystem and provides a model by which developers can create flexible and reusable applications. Application consists of 3 hierarchically ordered subsystems which are user interface, middleware and a database system. In interface subsystem, there are different pages according to users. In middleware subsystem, application, there are connections between user interface and database. This subsystem is required for updating databases and getting information from database. Our layer decomposition also proposes the closed architectural style, in which a layer can only access to the layer below it.



Therefore, 3-layer architecture is suitable for AutoHotel software system.

Figure 34: Layers Of System

#### **4.3.2 Model-View- Controller**

In this architectural style, the main approach is classifying the subsystems into three parts, called model, view and controller. The model, hotel database which is online database, stores data that is retrieved according to commands from the controller and displayed in the view. Hotel management constitute the controller in this design and can send commands to the model to update the hotel's states. It can also send commands to its associated view to change the information presented to employees from the database. User interface constitute the view part of MVC design, which shows data comes from controller to users which are employees in AutoHotel. For this reason, MVC system design is also suitable for our project.

#### **4.4 Hardware/Software Mapping**

One of the main design goals of the AutoHotel project is platform-independence which means implemented codes on developer's machine can be used on another machine without (or with minimal) changes. Thus, most subsystems have few specific hardware requirements. Exceptions to this are the database and some users access to system. The Database subsystem will be online therefore no specific hardware platform is required. Some employees' difference access to system is caused from their mobility such as room keeper and waiter.

The mobility problem can be solved by using tablets that are available with Android software. However in demo presentation of AutoHotel system, there will be no implementation for Android based AutoHotel system. Since our project requires the Java Runtime Environment because we implement this project by using Java. Also, our program requires a keyboard and a mouse for every actor in order to interact

with the program.

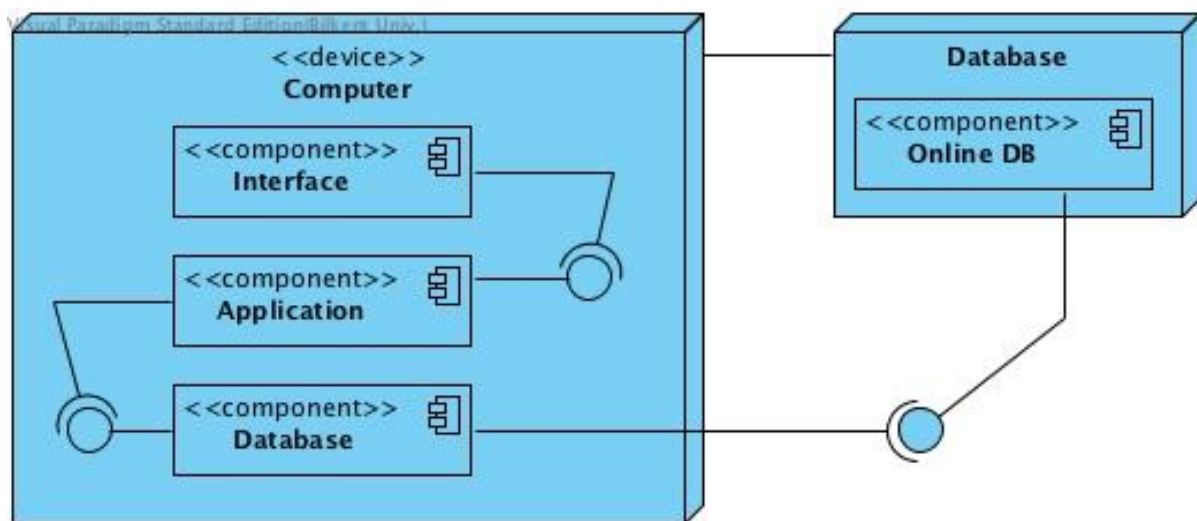


Figure 35: Deployment Diagram

## 4.5 Addressing Key Concerns

### 4.5.1 Persistent Data Management

AutoHotel Software will be stored data on online database because we need to access the proper real-time data in all computers in the hotel network as we stated in consistency part. When a worker makes an operation on objects, it will update the related database through objects, then this change will be shown in all computers. We will have four different type database table for four different objects which are customer, hotel, room and worker. These objects attributes will be column of database tables. When new object attributes is stored, database will be create new row to store them. The key point is object ids. Main search algorithms use this ids to find in database tables. There are also search algorithms that take name and room number as parameters in order to find specific element.

Our data will be downloaded from our online database when an actor logs in the program. After that, if there is a change on database and new object is created, the software get these updates automatically from our online database, then send these updates to the screen of related actor.

### **4.5.2 Access Control and Security**

In our automation program, we will implement an authentication system, we will use the worker database to keep information of workers' username and passwords. This database will be online and also contains all the information about workers. According to our design, workers will reach their related screens by entering their login information. Therefore, we will overcome the configuration of access control by using security system. An actor cannot change his username or password by himself, this change can be made by manager only, therefore, by this way, we expect to resist most of the security breaches.

### **4.5.3 Global Software Control**

In AutoHotel software system, it is aimed establishing consistency among [data](#) from a source to a target and also it is aimed concurrency which means that many users can access data at the same time. Subsystems synchronize with using two-way file synchronization, updated files are copied in both directions. It is required for the purpose of keeping the two or more computers' datas identical to each other. We aimed the continuous harmonization of the data over time with updated data which is circulating among object periodically.

### **4.5.4 Boundary Conditions**

The program will give an error if the data which is downloaded from online database is missing or corrupted. This situation can be happened when there is interruption in network connection.

This program is an automation software, therefore, actors will use the program all their shift long. When the shifts are end, actors will logout and close their screens. Then, login screen will be appeared for the new employee who will work at next shift.

The program give an error when manager could not pay salaries of employees with sending email to bank. This situation can also be happened when there is interruption in network connection.

Employees enter their passwords incorrectly only for 3 times in order to provide safety. After this situation, program will be locked and employees should be take new password from accountant.

## 5) Conclusion

When we were making our analysis and design report, we used Visual Paradigm to prepare models and it was very useful tool to create our diagrams.

During analysis process, we learned how to start to create this project efficiently instead of jumping directly to writing code. We deeply discuss the structure of project, user communities and requirements of them, class diagrams, which storage technique will be useful and how user interface will be efficient. Also we improved our skills while decide the structure of classes and objects, visual representation of them and handle possible logic mistakes without any implementation. As we were creating this report, we focused on requirements of our users and tried to create a program which realistic, user-friendly and implementable. When we create our diagrams, we always care about implementation in order to create a realistic and meaningful analysis report.

After our analysis report, we focused on our design and during this process, we determined our functional requirements and dependencies of our classes. Thanks to this process, we learned different architectural design models and understand how software layers separate between them. We learned how mapping our software into different hardware devices.