Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Hotel Management System

CSE 3224

Information System Design & Software Engineering Lab

Submitted By:

Md Junaeid Bhuiyan	16.01.04.055
Afranul Haque Afran	16.01.04.059
Nazmus Sakib	16.01.04.072
Md Ruhul Amin Rakib	16.01.04.074

Project Motivation

Hotel management system is mainly a desktop-based application. This project will be developed to maintain a hotel in a digitalized way. Hotel management system is mainly a desktop-based application. This project will be developed to maintain a hotel in a digitalized way. In this report we are going to describe the USE CASE & CLASS DIAGRAM of our project. Analyzing our project Hotel Management System, we have gathered the necessary behaviors, actor and interactions between these actors and finally a use case diagram has been created. We have also developed a class diagram for our project. We've tried to identify the necessary classes and interactions from our project requirements analyzed before.

Primary Actors

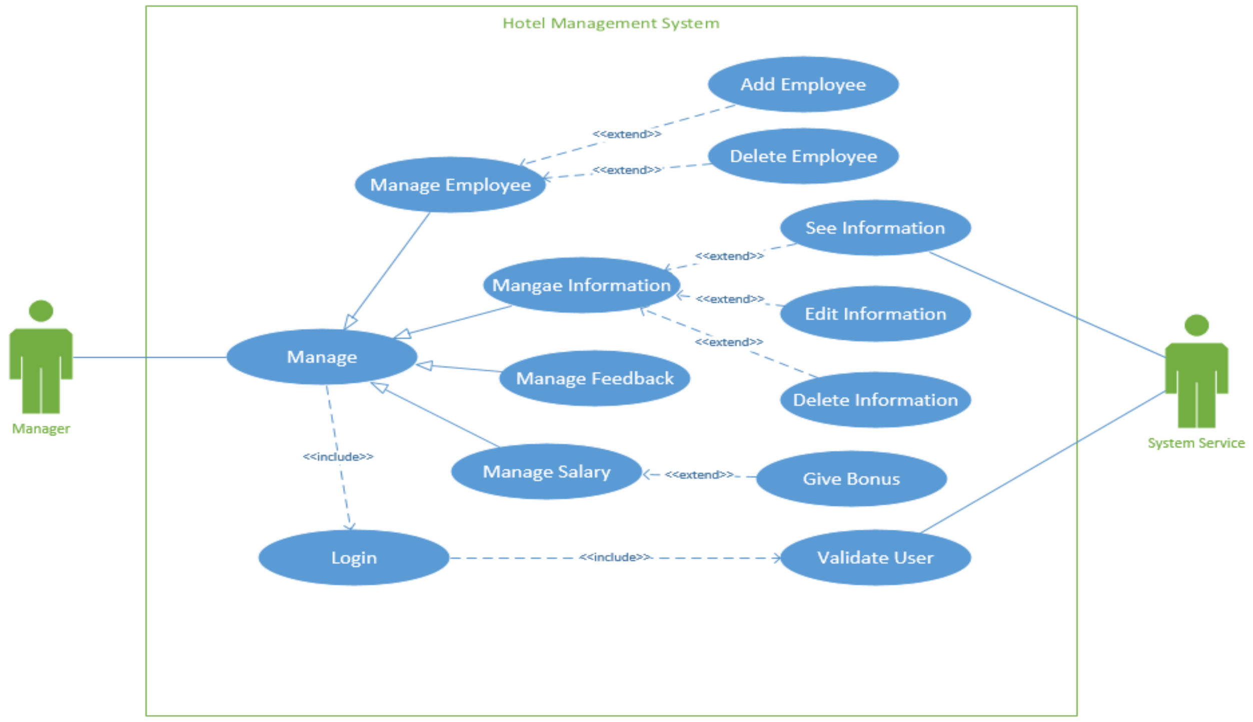
- **Manager:** A manager is a one kind of user in the system. He manages all the employee by adding employee, editing their info, deleting an employee, giving salary and bonuses to the employee. He can also see the customer feedback to improve their issues. Manager must logged in before he does something.
- **Receptionist:** A receptionist is another type of user in the system. First, he must be logged in to the system. He can book a room by choosing room, choosing staying duration, choosing discount and after taking payment. He can take payment in cash, check or by online payment such as Bikash. He can check the booking list, check-in list, checkout list and also the room information of the hotel. Room information is extended by checking available room, all room info and adding room when a new room is added or a room got finished its upgradation and blocking a room when a room is busy in upgradation. He can also order food for a guest and take feedback from the guest about their service and environment.
- **Employee:** Employee is another kind of user in our system. He can check is personal information as well as can edit his phone number and email information. Also, he can see his salary statement in this section. As other primary actors, employee must be logged first in this section.

Secondary Actors

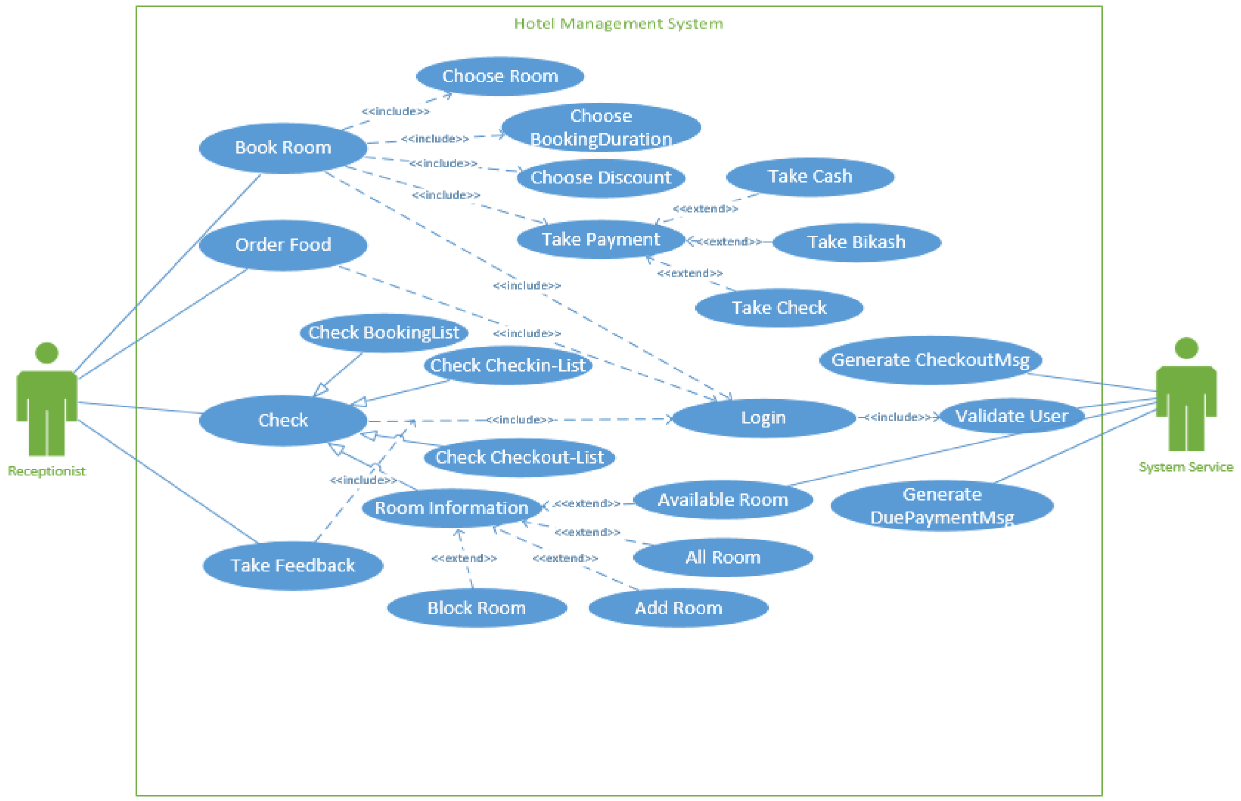
- **System Service:** The system service validates all the primary actors when they try to log in to the system so that specific user can logged in to the specific allowed section. The system service also generates message to the receptionist about the checkout time of a guest and also the due payment of a guest in the time of checkout.

Use Case Diagram

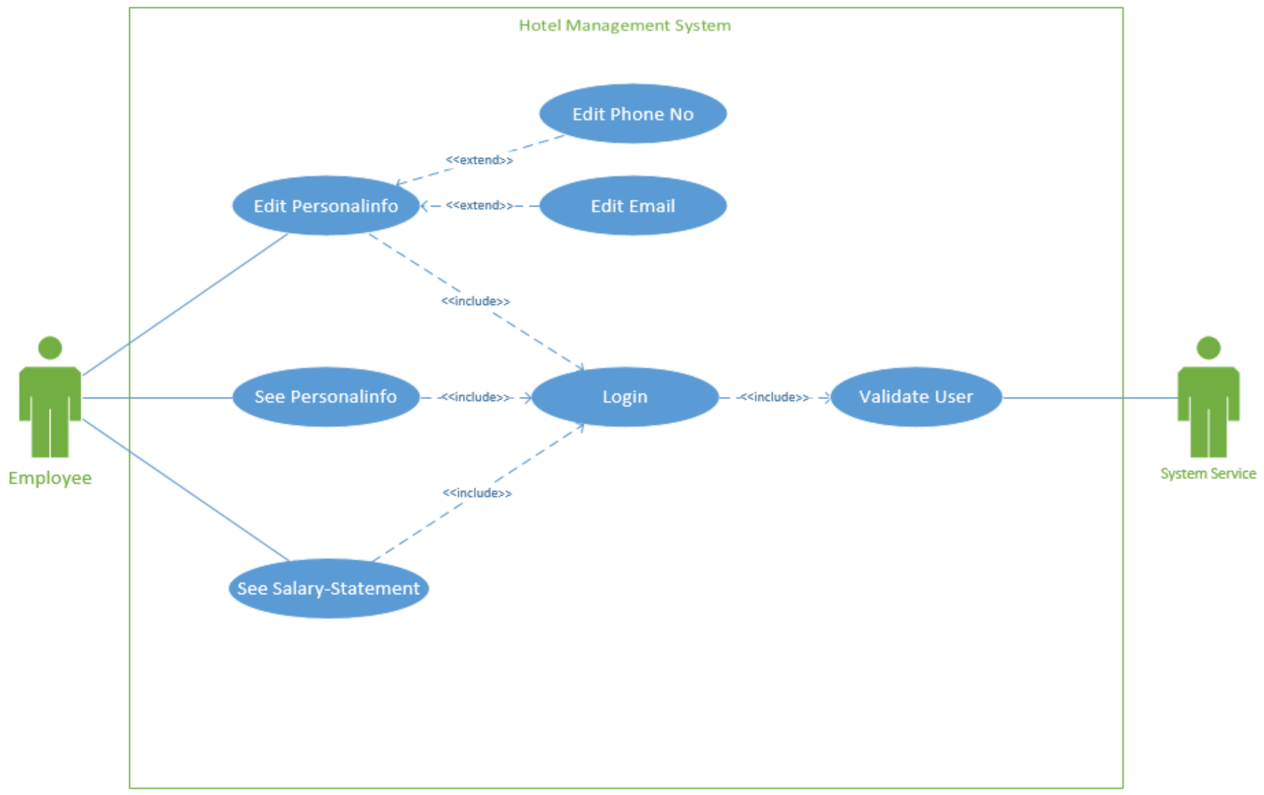
According to Manager:



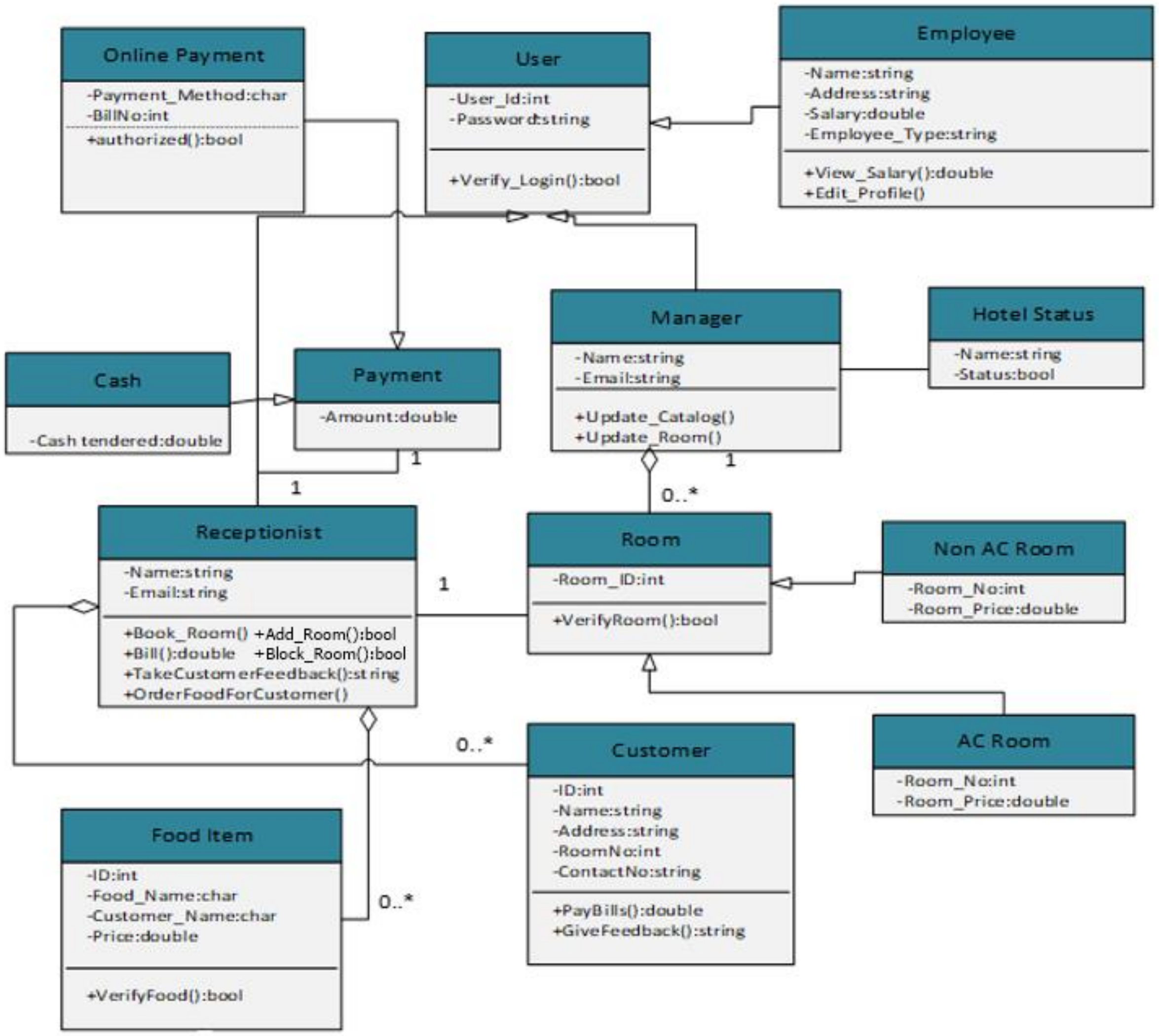
According to Receptionist:



According to Employee:



Class Diagram:



List of attributes and methods with visibility:

User Class:

Attributes:

- -user_id: int
- -password: string

Methods:

- +verifyLogin ()

Manager Class:

Attributes:

- -Name: string
- -Email: string

Methods:

- +Update_Catalog()
- +Update_Room()

Employee Class:

Attribute:

- -Name: string
- -Address: string
- -Salary: double
- -EmployeeType: string

Methods:

- +View_Salary(): double
- +Edit_Profile()

Receptionists Class:

Attribute:

- -Name: string
- -Email: string

Method:

- +Book_Room()
- +Bill(): double
- +TakeCustomerFeedback(): string
- +OrderFoodForCustomer()

Customer Class:

Attribute:

- -ID: int

- -Name: string
- -Address: string
- -RoomNo: int
- -ContactNo: int

Method:
- +PayBills(): double
- +GiveFeedback(): string

Hotel Status Class:

Attribute:
- -Name: string
- -Status : string

Refund Class:

Attribute:

- -Room_id: int

Method:
- +VerifyRoom(): bool

AC Room Class:

Attribute:
- -RoomNo: int
- -RoomPrice : double

Non AC Room Class:

Attribute:
- -RoomNo: int
- -RoomPrice: double

Food Item Class:

Attribute:
- -ID : int
- -FoodName : string
- -CustomerName : string
- -Price : double

Method:
+VerifyFood: bool

Payment Class:

Attribute:
- -Amount : double

Cash Class:

Attribute:
- -CashTendered: double

Online Payment Class:

Attribute:

- -PaymentMethod : string
- - BillNo: int

Method:

+Authorized()

Conclusion:

Use cases capture the functional requirements of a system and tell us what the system should do. From this use diagram we can describe the interactions between the different actors and the system that how the system is going to be used. More over from the class diagram we can see classes and relationship among them.