**Problem Description:**

Hotel **Renaissance** is one of the newest hotels in Hubli. The hotel is equipped with all the general amenities and facilities , set amidst beautifully landscaped gardens and fresh water lake it proves to be an ideal dream destination for a perceptive traveler.

The hotel has well furnished rooms with inclusive meals and lip smacking dishes.

The customer requests to book a room by providing the necessary details, a unique ID will be assigned to him.The room will be allocated to the customer depending upon the availability of vacant rooms .If the customer is unhappy with the room conditions he can contact the room service .The details of all the customers will be stored which can be viewed by the manager. The customers can order food from the provided appetizing menu. Before the customer checks out, he will be asked for a feedback.

**Objects Identified:**

Customer

Room Customer

Employee

Dish

Restaurant Customer

Restaurant

Hotel

Room

Room Service

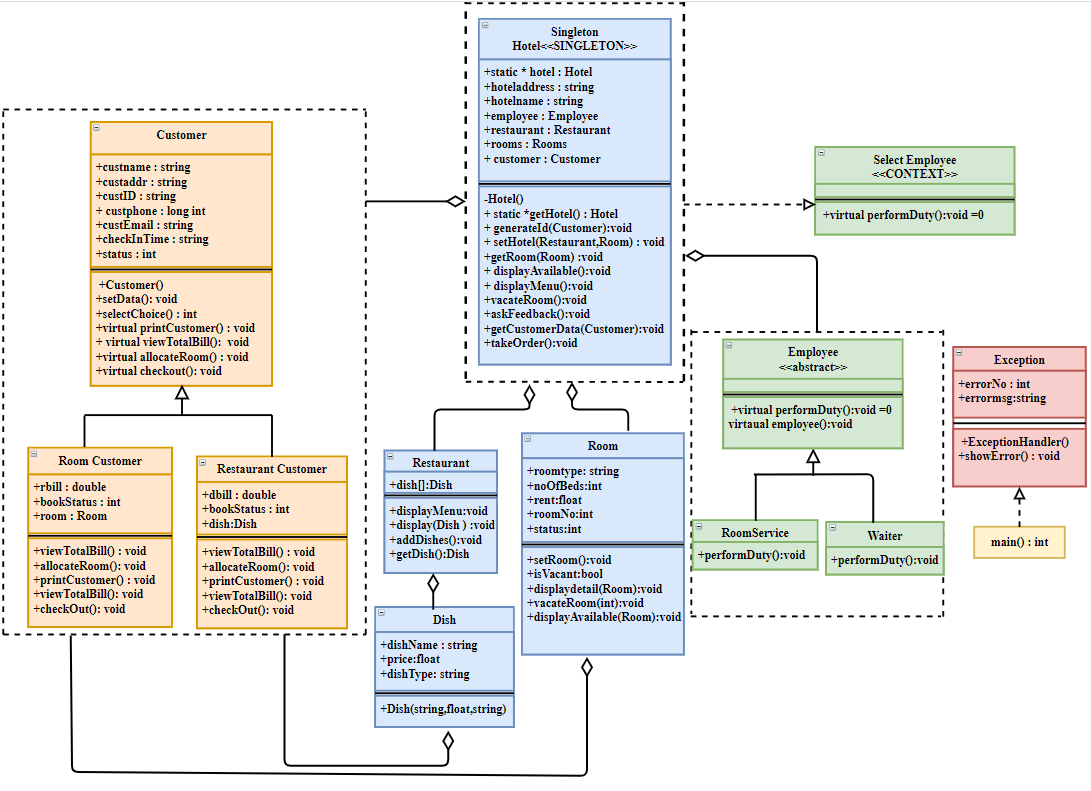
**Design Patterns used:**

Singleton

Strategy

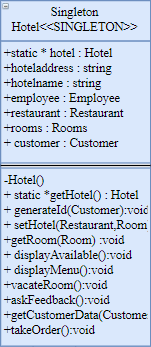
Factory

**Class Diagram:**



**Implementation Details:**

**Class : Hotel**

****

Hotel is the main class which follows singleton design pattern. It consists of

static \*hotel to create an instance of the class. The other attributes include the name and the address of the hotel.

**Functions:**

generateID() which generates a unique ID for every customer.

displayAvailable(): displays all the available rooms in the hotel.

displayMenu(): displays the menu available in the restaurant.

guestSummary(): It gives the whole summary of the details of the customer.

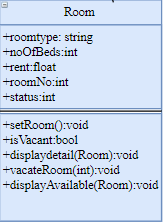
askFeedback() : It asks the feedback of the customer about the hotel service.

vacateRoom(): to vacant the room that has been allotted.

getCustomerData() : to get the data from customer.

takeOrder() : to take the order from the customer.

**Class : Room**



It has five attributes where roomType describes the type of room i.e., Deluxe, AC, Non-AC , General, Suite. The attribute status tells whether the room is occupied or no.

**Functions:**

setRoom() : to book a room.

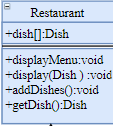
isVacant() : to know the status whether the room is booked or no.

displayDetail(): to display the details of all rooms

vacateRoom(): to vacate the room.

displayAvailable() : to display the details of the rooms which are available.

**Class : Restaurant**



It consists of array all the dishes that are available.

**Functions:**

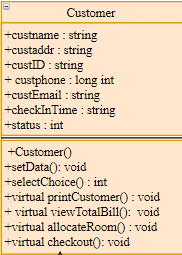
displayMenu() : to display the menu for all dishes available.

display() : To display the details of a particular dish

addDish(): to add a dish into the menu.

getDish() : to get the dish.

**Class: Customer**



The class Customer has two inherited classes restaurantCustomer and roomCustomer. It follows factory design pattern

**Functions:**

Setdata() : to enter all the customer details.

selectChoice() : To select the choice between booking a room and placing an order for a dish.

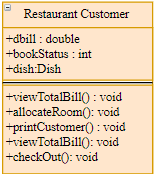
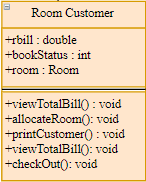
callRoomService(): To call the room service.

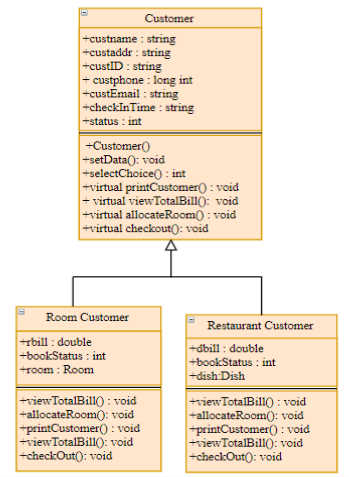
viewTotalBill() : to view the bill of the service service provided to him.

Checkout() : to checkout from the hotel.

printCustomer() : to print the customer details who is in the room.

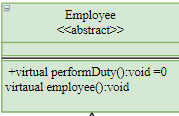
Class RestaurantCustomer: Class RoomCustomer:



The customer class follows factory design pattern.

**Class : Employee**

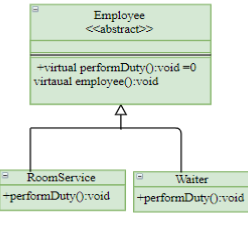


**Functions:**

performDuty() : To display all the employees working in the hotel

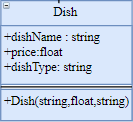
**Class: Roomservice Class : Waiter**

**** 

****

The class employee inherits two classes namely RoomService and Waiter who perform their respective duties on customer demand. It follows strategy design pattern.

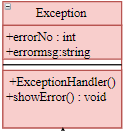
**Class : Dish**



This class has attributes such as dishname, price and dish type . The dishtype specifies the type to which the given dish belongs i.e., main course, starters, desserts , beverages etc.

The parameterized constructor creates new dishes.

**Class : Exception**

****

The following exception class comes into picture if

1. any customer asks for a room which is already occupied.
2. any customer orders for a dish which is not available at that instance of time.

In the above cases Exception Handler will throw an appropriate error.

**Code Snippet**