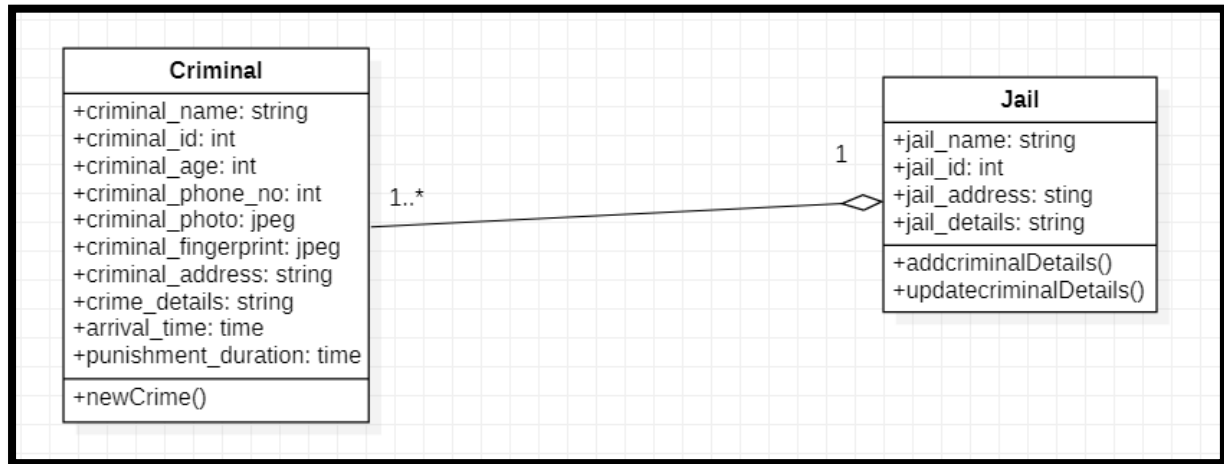**Name:Vishal Salvi**
Uid :2019230069

Q 1a) Criminals live in Jail. Find the relation between the classes Criminal and Jail. **1Mark**

Aggregation relationship:
Aggregation is a special type of association that models a whole- part relationship between aggregate and its parts.
The class of Jail is made up of one or more criminal. Here, the Jail class will remain even if the Criminal is not available.

Q 1b) A client who runs a small hotel chain wants you to design a piece of software to handle the day-to-day operations of the business. You receive this description.

We own several hotels and need to keep track of pricing, availability, and reservations in each building. Customers may call ahead to place reservations; they may also simply walk in to get a room (which is pretty much like making a reservation that starts immediately). The customer's reservation will specify how long the room is reserved. Once the customer finishes the stay, we charge payment for the room. We need to make sure that appointments never overlap, of course.
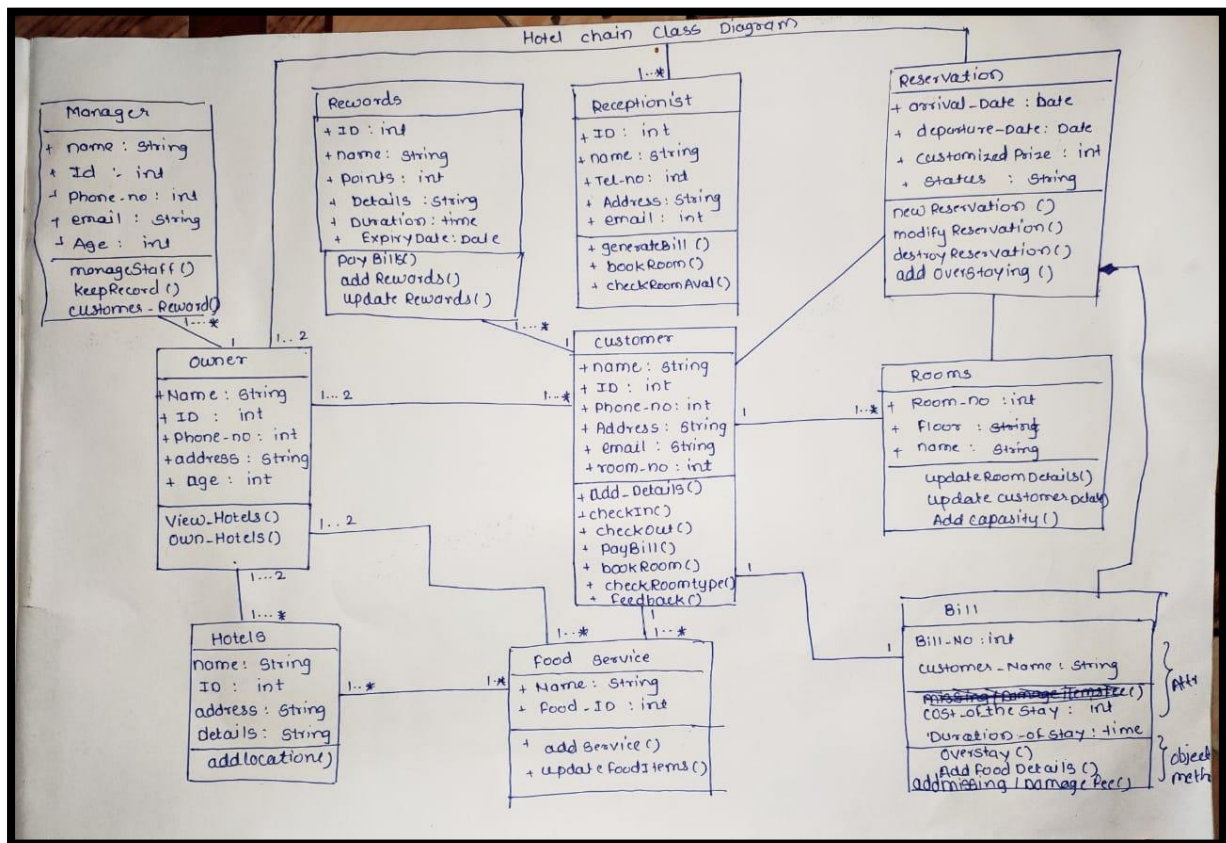
We also provide services during the customer's stay: pay-per-view movies, delivery service, and so on. When we make a bill for the customer, we must keep track of the cost of the stay, the duration of the stay (as customers may leave before or after their reservation is complete), and any services that the customer consumed. Different hotels offer different services, so we need to keep track of the offerings at each hotel. Note that prices may change over time, so we need some way of keeping track of what the customer paid during their stay regardless of what prices are now.

We also charge for overstaying a reservation. If a customer is still staying in a room past the end of a reservation, we charge the regular rate for the room plus a ten percent fee. Customers are allowed to extend their reservations, though, so this fee only applies if the customer overstays the reservation without asking us to extend it. In rare cases, we will also charge fees

for missing or damaged items – some customers seem to have a fondness for our towels – and we bill these fees in the same way that we bill services.

We also have a customer rewards program to encourage customer loyalty. Customers may have a rewards account which allows them to accumulate rewards points. One point is earned for each dollar the customer spends. Customers may pay their bills using either points or money but not both. Rooms and services have prices in terms of points as well as in terms of money. Points do not last forever; they expire two years after they are earned if they have not been spent. This means that we need to know not only how many points each customer has but also when they were earned. Remember that different amounts of points can be earned at different times.

Draw a UML class diagram which illustrates a domain model for the above application. Make sure to include appropriate relationships on the diagram, notation indicating the multiplicity of these relationships ("1..*", etc.), and methods and attributes for entities.

Hotel chain Class Diagram

Manager
- + name : String
- + Id : int
- + phone-no : int
- + email : String
- + Age : int

- manageStaff ()
- keepRecord ()
- customes-Reward()

Records
- + ID : int
- + name : String
- + points : int
- + Details : String
- + Duration : time
- + ExpiryDate : Date

- pay Bill()
- add Rewards()
- update Rewards ()

Receptionist
- + ID : int
- + name : String
- + Tel-no : int
- + Address : String
- + email : int

- + generateBill ()
- + bookRoom ()
- + checkRoomAval ()

Reservation
- + arrival-Date : Date
- + departure-Date : Date
- + Customized Prize : int
- + Status : String

- new Reservation ()
- modify Reservation ()
- destroy Reservation ()
- add Overstaying ()

Owner
- +Name : String
- + ID : int
- + phone-no : int
- + address : String
- + age : int

- View-Hotels ()
- own-Hotels ()

Customer
- + name : String
- + ID : int
- + phone-no : int
- + Address : String
- + email : String
- + room-no : int

- + Add-Details ()
- + checkinc()
- + checkout ()
- + payBill ()
- + bookRoom ()
- + checkRoomtype()
- + feedback()

Rooms
- + Room-no : int
- + Floor : String
- + name : String

- update RoomDetails()
- update customerpay
- Add capasity ()

Hotels
- name : String
- ID : int
- address : String
- Details : String

- add location()

Food Service
- + Name : String
- + Food-ID : int

- + add Service ()
- + update foodItems()

Bill
- Bill-No : int
- customer-Name : String
- Actual prize/Item Bill()   { Att
- cost of the stay : int
- Duration-of stay : time

- overstay ()
- Add Food Details ()   } Object
- add missing Damage fee()   } meth

Multiplicities: 1..*, 1..2, 1..2, 1..*, 1..*, 1..*, 1..*, 1, 1..*