***OOAD-Assignment***

Group Number : 04

Name of the Assignment: Airline reservation system

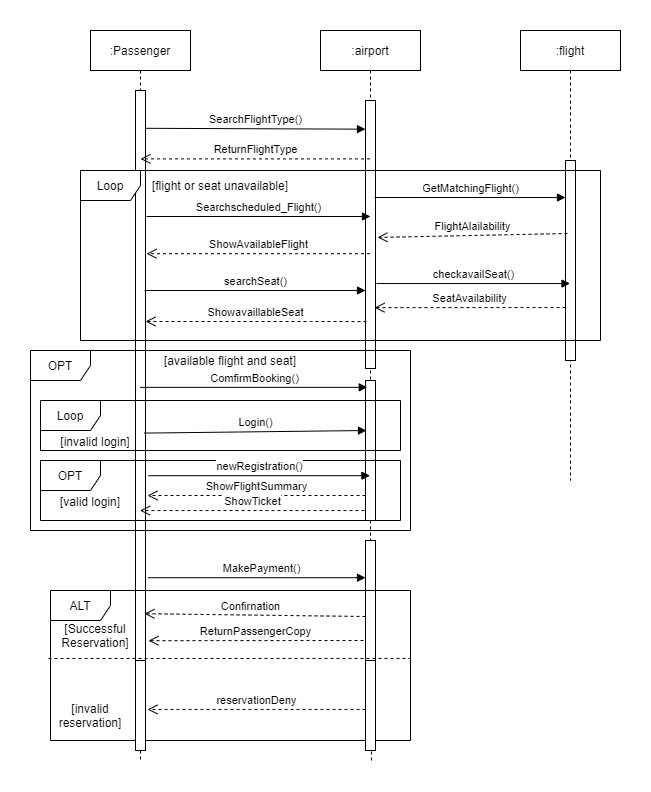
Name of the Course:Object oriented Analysis and design.

Group Members’ Name and ID:

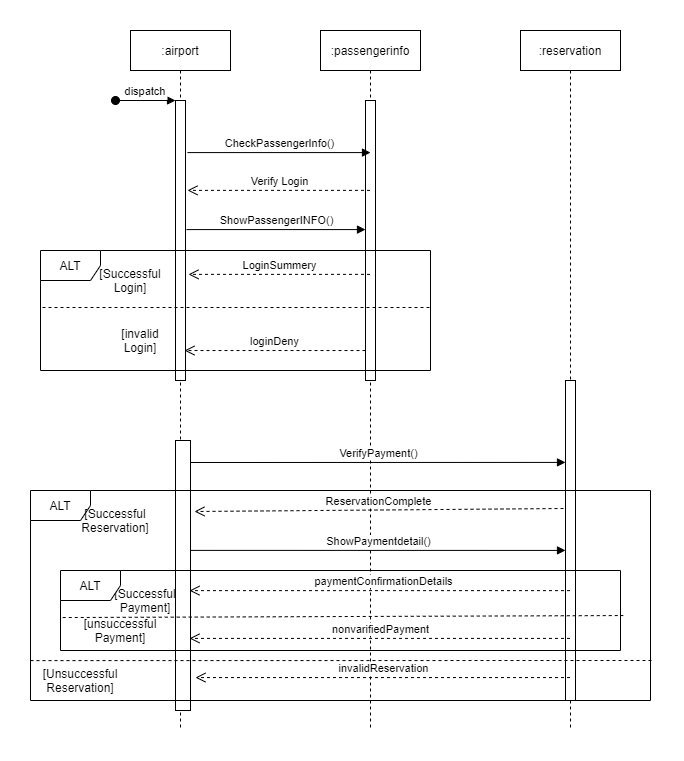
* Zafrin Sultana (19-39345-1) (D section)
* Rifatul azim (19-39388-1) (D section)
* B.M.Mahir Shahriar (19-40538-1) (F section)

Brief Description: In an airline reservation system, A customer will search for flight in airport server. To know the availability the customer have to select the details of the flight so that he/she can be sure about confirming. First he/she will see the destination and select one. Then he/she will be asked to select the preferable date and time .After selecting, the server will see if there is any flight available of that particular time. If yes then server will ask for further details and if not the server will deny. For the available flight, customer will select the flight and check no of seat which are available. Then server asks if the customer is interested to go further for reservation. Then comes the booking process. To confirm the flight the customer have to login to the server as a passenger. After verifying the information of the customer by system, the customer will select flight details and then system will show the amount to pay. If the customer wants to confirm the reservation he/she have to pay the bill otherwise the ticket will be cancelled. To pay the amount the customer will enter the credit card info and and verify it .If the information is nor verified it will ask again to enter correct details. After verifying the customer will pay the bill and confirm the reservation.

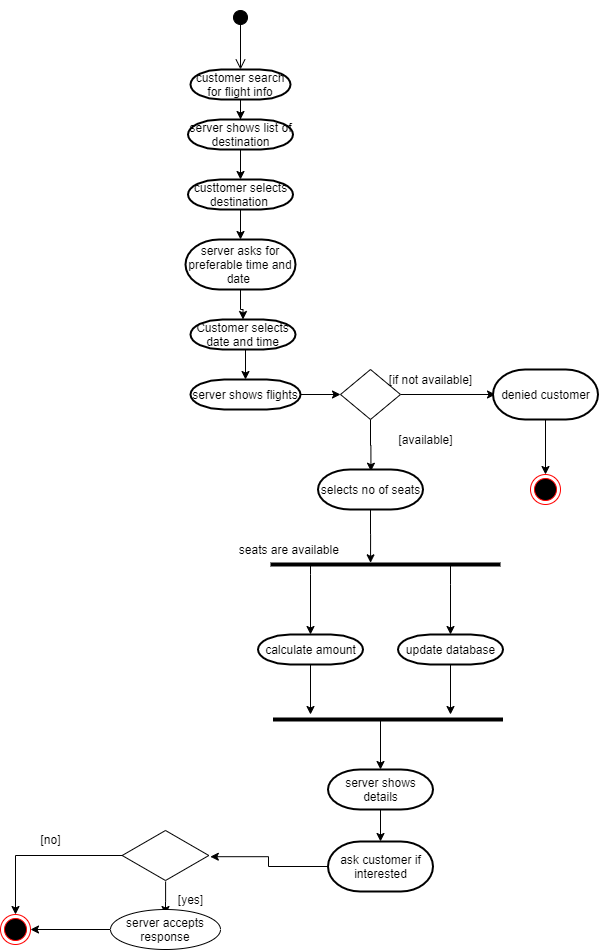
Sequence diagram for “Select flight”:



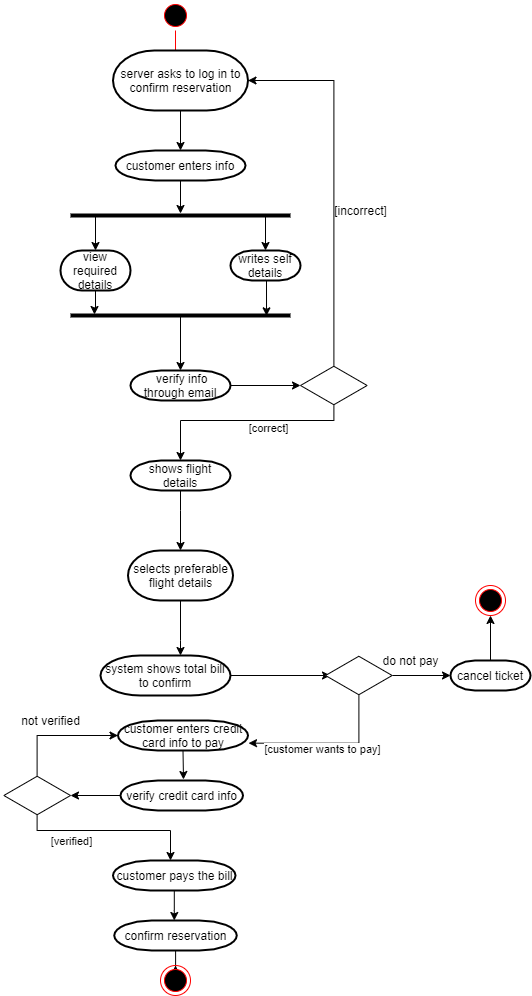
Sequence diagram for “confirm payment”:



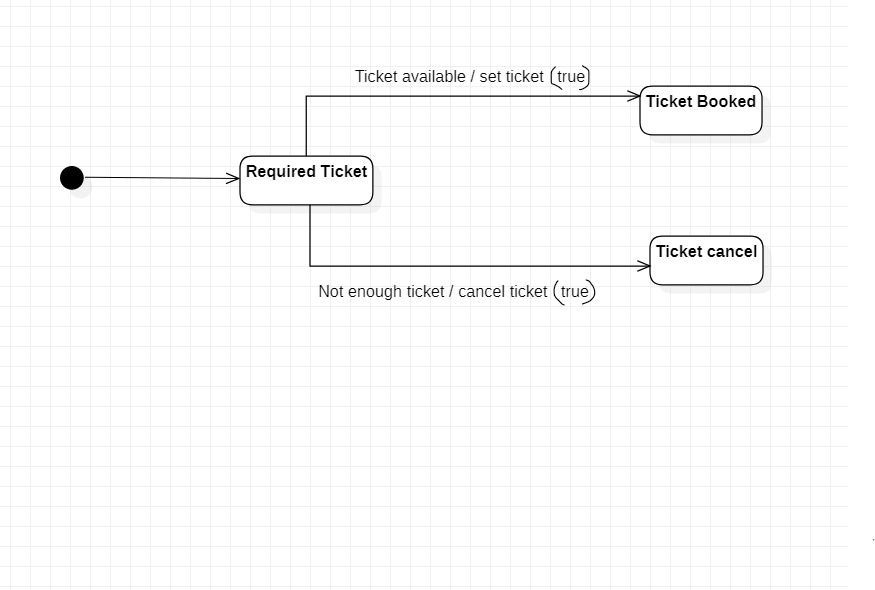
Activity diagram for “Checking flight info”:



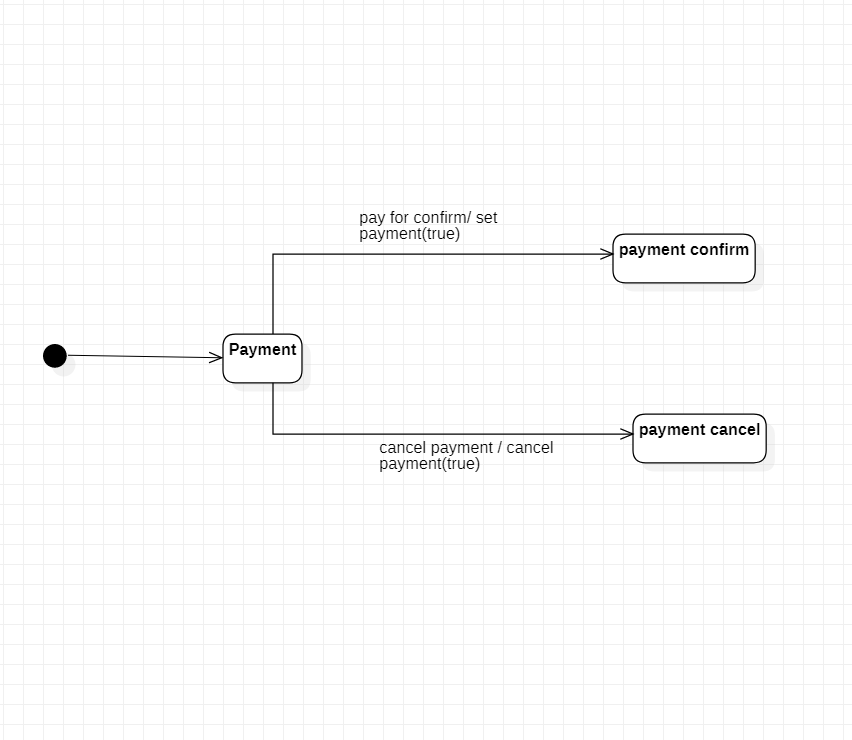
Activity Diagram for “Confirm Reservation”:



Statechart Diagram for “ticket” object:



Statechart Diagram for “payment” object:



**Calculation of the LCOM**

The ‘Customer’class:

CLASS Customer

-passenger\_name:String

-passenger\_id\_no:integer

-no\_of\_tickets:integer

-destination:String

-travel\_date:date

travel(destination,travel\_date)

payment(passenger \_name, passenger\_id\_no,no\_of\_tickets)

booking(name,Id\_no,time)

As, LCOM = |P|-|Q|, if |P| > |Q|, otherwise 0

For the Customer class :

Pairs:

(travel,payment),(payment,booking),(booking,travel)

P = 1(Non-Cohesive pairs)

Q = 2 (Cohesive pairs)

Q > P

So,for ‘Customer’ class LCOM = 0

**The Class ‘Flight’:**

CLASS Flight\_info

-Flight\_type:String

-flight\_code:integer

-destination:String

-travel\_date:date

-passenger\_name:String

-passenger\_id\_no:integer

-no\_of\_seat:integer

travel(destination,travel\_date)

addPassenger(passenger\_name, passenger\_id\_no,flight\_code,no\_of\_seat)

removePassenger(passenger\_name, passenger\_id\_no,flight\_code)

getFlight(flight\_code,travel\_date,destination)

As, LCOM = |P|-|Q|, if |P| > |Q|, otherwise 0

For the ‘Flight\_info’ class :

Pairs:

(travel,addPassenger),(addPassenger,removePassenger),(remove passanger,getFlight),(getFlight,travel),(travel,removePassenger),(addPassenger,getFlight),

P = 1(Non-Cohesive pairs)

Q = 5 (Cohesive pairs)

Q > P

So,for ‘Flight\_info’ class LCOM = 0

**Decision**: Two methods of a class can be considered “cohesive” if the set of instance variables of *the class* that they access have some elements in common.So, The larger the number of similar **methods**, the more **cohesive** the class, For both Customer and Flight\_info class, We got the value of LCOM is **zero**.As on both Classes the number of **cohesive pairs(P)** were greater than the **non-cohesive pair (Q),** The LCOM became zero. Because The **larger** the number of cohesive methods, the more cohesive the class will be, and the LCOM metric will be **lower .**

On the other hand,a high LCOM value may indicate that the methods are trying to do different things and operate on different data entities. Which means the class can be parted into different classes and keep only those methods who are working together.

As the Customer and Flight\_info class have Low LCOM value it means we don’t need to partition the method and its okay to have those method in that class because the methods are well connected.