

**COEN 6312**

**Model Driven Software Engineering**

**Deliverable-4**

**Project-Team**

|  |  |
| --- | --- |
| Name | ID No. |
| **Ganesh Santhar** | **40010625** |
| BhanuPrakash Ramineni | 27107838 |
| Jithin James | 27420854 |
| Rambabu Kunchala | 27262957 |
| Rakhi Ubriani | 27396333 |

**Index**

[**1** **Introduction** 2](#_Toc446363534)

[**2** **State Diagram** 2](#_Toc446363535)

[**2.1** **State Machine for Credit /Debit/ Interac Class** 2](#_Toc446363536)

[**2.2** **State Machine for Account Class** 3](#_Toc446363537)

[**2.3** **State Machine for Flight Class** 4](#_Toc446363538)

[**2.4** **State Machine for Member Points Class** 5](#_Toc446363539)

[**3** **Action Specification** 6](#_Toc446363540)

[**3.1** **Action Specification for Operation Login()** 6](#_Toc446363541)

[**3.2** **Action Specification for Operation CreateTransaction()** 7](#_Toc446363542)

[**3.3** **Action Specification for Operation ViewBooking ()** 8](#_Toc446363543)

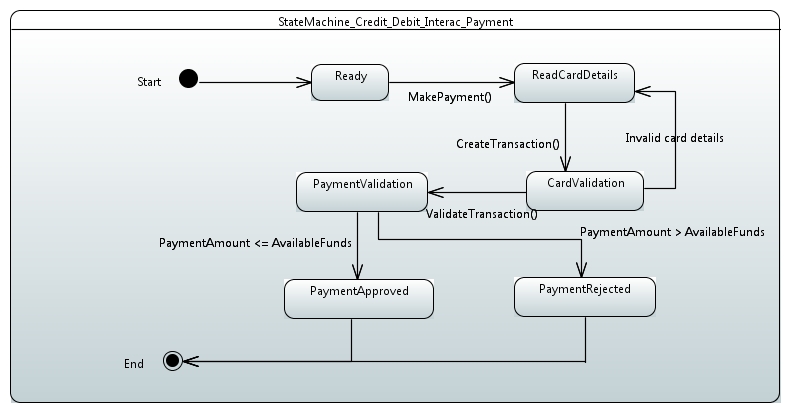
[**References** 8](#_Toc446363544)

# **Introduction**

This document emphasizes on the various state machines that are presented in the flight management system. The state diagram represents the behavior of an object as states in its lifecycle. The transition of the states of an object occurs with receiving a signal.

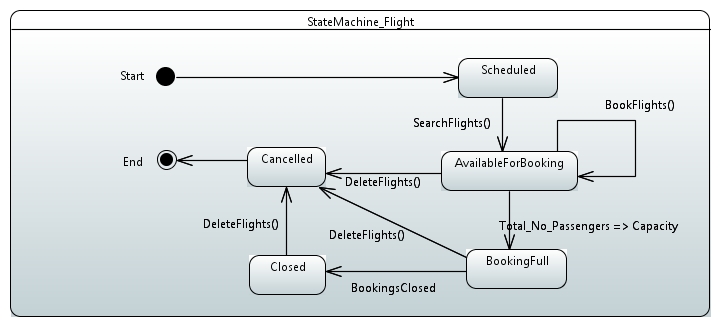
# **State Diagrams**

## **State Machine for Credit /Debit/ Interac Class**



1. The User entered Card details will be read by system and make available for validation.
2. If the Card details are valid, then the payment amount will be validated based on the available funds in the Card.
3. If the Card details are Invalid, then the system will prompt the user to re-enter the Card details.
4. In the Payment Validation, the Payment will be approved if the available funds in the Card are sufficient for Payment amount. If not, then the Payment will be rejected.

## **State Machine for Flight Class**



1. The flights are in Scheduled State and made Available for booking State based on the Search flights signal.
2. The object remains in Available for booking state upon the Book Flights signal.
3. The Booking full state is entered if the flight capacity is equal or exceeded by the Total no of passengers booked on booking the flight
4. The Cancelled state is entered only if the Delete Flights signal is given. i.e. the flight made deleted or cancelled by the service provider.
5. The Closed state is entered when the Booking Closed signal is given from the Booking Full state

# **Action Specification**

## **Action Specification for Operation Login()**

This Method performs below actions

* Gets Username and Password as input.
* Verifies the input with the User account database(members), upon successful authentication provides the access into account.

public static void login ()

{

System.out.println(" --------------------------------------------------------------- ");

System.out.println(" --------------------- LOGIN TO FlyinTravel ----------------------------- ");

System.out.println(" --------------------------------------------------------------- ")

System.out.print("\n =========== Please Enter the Username: ");

String user = read.GetString();

System.out.print("\n =========== Please Enter the Password: ");

String pass = read.GetString();

Static Vector Useraccounts = new vector (); // Vector contains User account details. Populated by other methods we are just using it.

// creating temporary Member class object to search in User accounts vector

Member Temp = new Member ();

for (int i=0; i<Useraccounts.size();i++)

{

Temp = (Member) Useraccounts.elementAt(i);

if(Temp.Username.equals(user) && Temp.Password.equals(pass))

{

System.out.println(" Login successful ");

System.out.println(" welcome to FlyinTravel ");

view (); // view account details

}

else {

System.out.println(" Login unsuccessful ---- Try again");

}

}

}

## **Action Specification for Operation CreateTransaction()**

This Method performs below actions

* Reads the Card number, CVV, Expiry date and Name on card as input.
* Sets the read input to the attributes.
* Bundles Card details and Payment amount in string buffer to pass on to validation.

public String [] createtransactions ()

{

String Transaction [];

System.out.print("\n =========== Please Enter your Name on card: ");

String name = read.GetString();

System.out.print("\n =========== Please Enter the CardNumber: ");

int cardnum = read.GetInt();

System.out.print("\n =========== Please Enter the CVV 3 digits in the back of the card: ");

int cvv = read.GetInt();

System.out.print("\n =========== Please Enter the Expiry Date MMDD format: ");

int exdate = read.GetInt();

System.out.print("\n =========== Printing the payment amount: " + Amount); // from parent class Payment

this.CardNumber = cardnum; // Setting the card number

this.Nameoncard = name; // Setting the card Name

this.CVV = cvv; // Setting the cvv number

this.ExpiryDate = exdate; // Setting the expiry date

return Transaction = new String [] {String.valueof(CardNumber), Nameoncard, String.valueof(CVV), String.valueof(ExpiryDate), String.valueof(Amount)}

// above command puts all the read input as one single buffer and returns to caller

}

# **References**

1. Dr. Abdelwahab Hamou-Lhadj Notes on Modelling with State Diagrams.