**Object-oriented concepts**

**BSc (Hons) in Information Technology**

Topic : **Movie Booking System**

Group no : **MLB_08.01_07**

Campus : **Malabe**

Submission Date : **20/05/2022**

We declare that this is our own work, and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

| Registration No | Name | Contact Number |
|---|---|---|
| IT21166488 | Maleesha K.L.D.D. S | +94 766598143 |
| IT21164026 | Wijayasooriya W.A.K. A | +94 704476158 |
| IT21164194 | D.M.W.B. Thesarana Dissanayake | +94 769997546 |
| IT21164644 | D.N Pathirathna | +94 759348378 |
| IT21163272 | Amanda Yasathri | +94 762661514 |

## 1) User Requirements

01) As an Unregistered customer first, he needs to register to the system by providing details such as name, email, username, and password.

02) Registered users can log in to the system by providing his/her valid login username/email and password.

03) Users can search for a movie and can view movie details.

04) As registered users they will be able to book tickets.

05) Movies shown in theatres have a payment option and it can be done through credit or debit card.

06) When a customer makes a payment, payment is validated in every transaction.

07) The payment details are stored in the system.

08) Users can make feedback and inquiries if needed.

09) The users can make all inquiries through the contact us page. The system also allows connecting across the page in three ways. They are the map, "Meet us" section & "Pitch us" sections. The contact us page is responsible for storing all inquiry information and responding to customers.

10) Managers can manage feedback, inquiries, and users.

11) In the system, the manager manages every customer, analyses customer feedback, and creates every report.

12) The manager can generate reports such as the list of movies (Now showing & Coming soon), the list of members & income reports.

13) System administrator can update all details regarding movies such as add movies, delete movies, movie showtime and prices. Etc.

## 2) Noun & verb Analysis

**Noun – Red || Verb – Blue**

01) As an Unregistered customer first, he/she needs to register to the system by providing details such as name, email, username, and password.

02) Registered users can log in to the system by providing his/her valid login username/email and password.

03) Users can search for a movie and can view movie details.

04) As registered users they will be able to book tickets.

05) Movies shown in theatres have a payment option and it can be done through credit or debit card.

06) When a customer makes a payment, payment is validated in every transaction.

07) The payment details are stored in the system.

08) Users can make feedback and inquiries if needed.

09). The users can make all inquiries through the contact us page. The system also allows connecting across the page in three ways. They are the map, "Meet us" section & "Pitch us" sections. The contact us page is responsible for storing all inquiry information and responding to customers.

10) Managers can manage feedback, inquiries and users.

11) In the system, the manager manages every customer, analyses customer feedback, and creates every report.

12) The manager can generate reports such as the list of movies (now showing & coming soon), the list of members & income reports.

13) System administrator can update all details regarding movies such as add movies, delete movies and update movies showtime and prices. Etc.

**Redundant:**

- Unregistered Customer
- User
- Registered user

**Outside the scope**

- System
- Theatre
- The map
- Meet us
- Pitch us

**Event Operation:**

- Search
- Add
- View
- Delete
- Update

**Meta Language:**

- He
- She
- His/her
- They

**Attributes:**

- Name
- Email
- Username
- Password
- Credit Card
- Debit Card

**Classes:**

- Customer
- Payment
- Contact
- Manager
- Feedback
- Ticket
- Movie
- Now showing movie
- Coming Soon movie
- Admin
- Report

### 3) CRC Cards

| **Class Name:** Customer | |
|---|---|
| **Responsibility** | **Collaborations** |
| Register to the system | |
| Log in to the system | |
| Book tickets | Ticket |
| Make payment | Payment |
| Edit profile details | |
| Store customer details | |
| Validate customer details | |

| **Class Name:** Payment | |
|---|---|
| **Responsibility** | **Collaborations** |
| Store all details of payment | |
| Display payment details | |
| Calculate total amount | Ticket |
| Verify payment | Card |

| **Class Name:** Contact | |
|---|---|
| **Responsibility** | **Collaborations** |
| Store inquiry details | |
| Display inquiry details | |

| **Class Name:** Manager | |
|---|---|
| **Responsibility** | **Collaborations** |
| Manage customers | Customer |
| Analyse feedback | Feedback |
| Create reports | Report |

| **Class Name:** Feedback | |
|---|---|
| **Responsibility** | **Collaborations** |
| Store feedback details | |
| Display feedback | |

| **Class Name:** Ticket | |
|---|---|
| **Responsibility** | **Collaborations** |
| Display Movie name | Movie |
| Display movie date | |
| Display movie time | |
| Display ticket price | |
| Display seat number | |

| **Class Name:** Movie | |
|---|---|
| **Responsibility** | **Collaborations** |
| Keep records of “Now showing “movies | |
| Keep records of “Coming Soon “movies | |
| Display movie | |

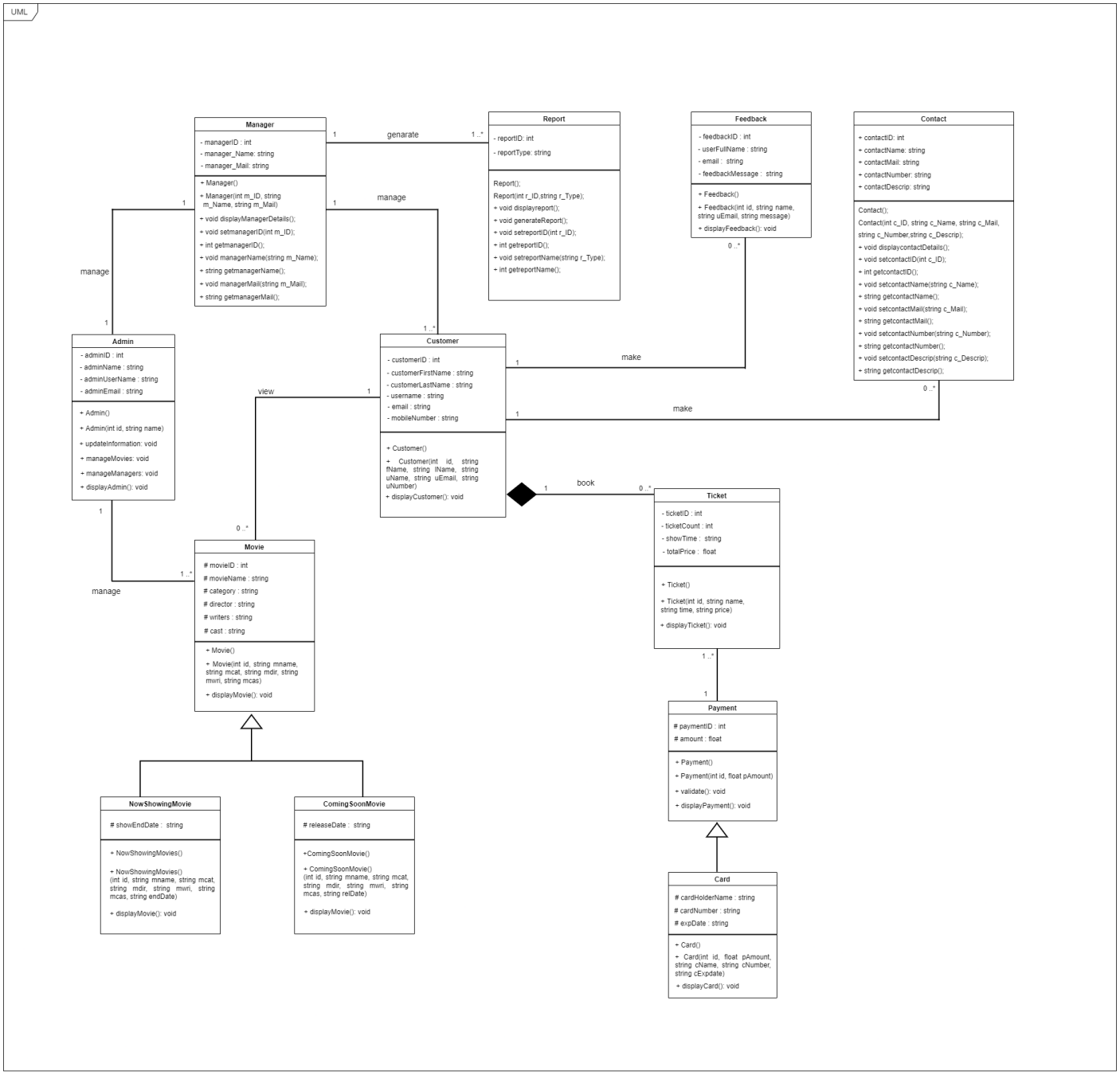
| **Class Name:** Now showing | |
|---|---|
| **Responsibility** | **Collaborations** |
| Store details of now showing movies | Movie |
| Display now showing movies | |

| **Class Name:** Coming Soon | |
|---|---|
| **Responsibility** | **Collaborations** |
| Store details of coming soon movies | Movie |
| Display coming soon movies | |

| **Class Name:** Admin | |
|---|---|
| **Responsibility** | **Collaborations** |
| Keep records of login details | |
| Update/ Edit information | |
| Manage Managers | Manager |
| Add/ Remove/ Update movies | Movie |

| **Class Name:** Report | |
|---|---|
| **Responsibility** | **Collaborations** |
| Report of movies | Movie |
| Report of members | Customer |
| Report of income | Payment |
| Display report | |

# Exercise: 1 – Class Diagram



## Exercise: 2 – Coding Part C++

```
#include <iostream>
#include <string>
using namespace std;

// Class Admin
class Admin {
private:
        int adminID;
        string adminName;
        string adminUsername;
        string adminEmail;
public:
        Admin();
        Admin(int id, string name, string username, string email);
        void updateInformation();
        void manageMovies();
        void manageManagers();
        void displayAdminDetails();
};

Admin::Admin(){}

Admin::Admin(int id, string name, string username, string email)
{
        adminID = id;
        adminName = name;
        adminUsername = username;
        adminEmail = email;
}

void Admin::updateInformation(){}

void Admin::manageMovies(){}

void Admin::manageManagers(){}

void Admin::displayAdminDetails()
{
        cout << "Admin ID: " << adminID << endl;
        cout << "Admin Name: " << adminName << endl;
        cout << "Admin Username: " << adminUsername << endl;
        cout << "Admin Email: " << adminEmail << endl << endl;
}

// Class Manager
class Manager {
private:
        int managerID;
        string manager_Name;
        string manager_Mail;
public:
        Manager();
        Manager(int m_ID, string m_Name, string m_Mail);
        void displayManagerDetails();
```

```
};

Manager::Manager(){}

Manager::Manager(int m_ID, string m_Name, string m_Mail)
{
        managerID = m_ID;
        manager_Name = m_Name;
        manager_Mail = m_Mail;
}

void Manager::displayManagerDetails()
{
        cout << "Manager ID: " << managerID << endl;
        cout << "Manager Name: " << manager_Name << endl;
        cout << "Manager Mail: " << manager_Mail << endl << endl;
}

// Class Report
class Report{
private:
    int reportID;
    string reportType;
public:
    Report();
    Report(int id, string type);
    void displayReport();
};

Report::Report(){}

Report::Report(int id, string type){
    reportID = id;
    reportType = type;
}

void Report::displayReport(){
    cout<<"Report ID: "<<reportID<<endl;
    cout<<"Report Type: "<<reportType<<endl<<endl;
}


// Class Movie
class Movie
{
protected:
    int movieID;
    string movieName;
    string category;
    string director;
    string writers;
    string cast;

public:
    Movie();
    Movie(int id, string name, string cat, string dir, string wri, string
cas);
```

```
    void displayMovieDetails();
};

class NowShowingMovie: public Movie {
protected:
    string showEndDate;
public:
    NowShowingMovie();
    NowShowingMovie(int id, string name, string cat, string dir, string
wri, string cas, string endDate);
    void displayMovieDetails();
};

class ComingSoonMovie : public Movie {
protected:
    string releaseDate;
public:
    ComingSoonMovie();
    ComingSoonMovie(int id, string name, string cat, string dir, string
wri, string cas, string rDate);
    void displayMovieDetails();
};

Movie::Movie()
{
}

Movie::Movie(int id, string name, string cat, string dir, string wri,
string cas) {
    movieID = id;
    movieName = name;
    category = cat;
    director = dir;
    writers = wri;
    cast = cas;
};

void Movie::displayMovieDetails() {
    cout << "Movie ID: " << movieID << endl;
    cout << "Movie Name: " << movieName << endl;
    cout << "Movie Category : " << category << endl;
    cout << "Movie Director: " << director << endl;
    cout << "Movie Writers: " << writers << endl;
    cout << "Movie Cast: " << cast << endl << endl;
};

NowShowingMovie::NowShowingMovie()
{
}

NowShowingMovie::NowShowingMovie(int id, string name, string cat, string
dir, string wri, string cas, string endDate)
    :Movie(id, name, cat, dir, wri, cas)
{
    showEndDate = endDate;
}
```

```
void NowShowingMovie::displayMovieDetails()
{
    cout << "Movie ID: " << movieID << endl;
    cout << "Movie Name: " << movieName << endl;
    cout << "Movie Category : " << category << endl;
    cout << "Movie Director: " << director << endl;
    cout << "Movie Writers: " << writers << endl;
    cout << "Movie Cast: " << cast << endl;
    cout << "Show endDate: " << showEndDate << endl << endl;
}

ComingSoonMovie::ComingSoonMovie()
{
}

ComingSoonMovie::ComingSoonMovie(int id, string name, string cat, string
dir, string wri, string cas, string rDate)
    :Movie(id, name, cat, dir, wri, cas)
{
    releaseDate = rDate;
}

void ComingSoonMovie::displayMovieDetails()
{
    cout << "Movie ID: " << movieID << endl;
    cout << "Movie Name: " << movieName << endl;
    cout << "Movie Category : " << category << endl;
    cout << "Movie Director: " << director << endl;
    cout << "Movie Writers: " << writers << endl;
    cout << "Movie Cast: " << cast << endl;
    cout << "Movie Release Date: " << releaseDate << endl << endl;
}

// Class Customer
class Customer {
private:
        int customerID;
        string customerFirstName;
        string customerLastName;
        string username;
        string email;
        string mobileNumber;
public:
        Customer();
        Customer(int id, string firstName, string lastName, string uname,
string cEmail, string number);
        void setCustomerID(int id);
        int getCustomerID();
        void displayCustomer();
};

Customer::Customer()
{
}

Customer::Customer(int id, string firstName, string lastName, string uname,
string cEmail, string number)
```

```
{
        customerID = id;
        customerFirstName = firstName;
        customerLastName = lastName;
        username = uname;
        email = cEmail;
        mobileNumber = number;
}

void Customer::setCustomerID(int id)
{
        customerID = id;
}

int Customer::getCustomerID()
{
    return customerID;
}

void Customer::displayCustomer()
{
        cout << "Customer ID: " << customerID << endl;
        cout << "Customer First Name: " << customerFirstName << endl;
        cout << "Customer Last Name: " << customerLastName << endl;
        cout << "Customer Username: " << username << endl;
        cout << "Customer Email: " << email << endl;
        cout << "Customer Mobile Number: " << mobileNumber << endl << endl;
}

// Class Feedback
class Feedback {
private:
        int feedbackID;
        string fullName;
        string email;
        string feedbackMessage;
        Customer* cu;
public:
        Feedback();
        Feedback(int id, string name, string cEmail, string message,
Customer *pcu);
        void displayFeedback();
};

Feedback::Feedback()
{
};

Feedback::Feedback(int id, string name, string cEmail, string message,
Customer* pcu)
{
        feedbackID = id;
        fullName = name;
        email = cEmail;
        feedbackMessage = message;
        cu = pcu;
};
```

```
void Feedback::displayFeedback()
{
        cout << "Feedback ID: " << feedbackID << endl;
        cout << "Customer Full Name: " << fullName << endl;
        cout << "Customer Email: " << email << endl;
        cout << "Message: " << feedbackMessage << endl;
        cout << "Customer ID: " << cu->getCustomerID() << endl << endl;
};

// Class Contact
class Contact{
private:
    int contactID;
    string contactName;
    string contactMail;
    string contactNumber;
    string contactMessage;
    Customer* cu;
public:
    Contact();
    Contact(int id, string c_Name, string c_Mail, string c_Number, string
c_Messgae, Customer* pcu);
    void displayContact();
};

Contact::Contact(){}

Contact::Contact(int id, string c_Name, string c_Mail, string c_Number,
string c_Messgae, Customer* pcu){
    contactID = id;
    contactName = c_Name;
    contactMail = c_Mail;
    contactNumber = c_Number;
    contactMessage = c_Messgae;
    cu = pcu;
}

void Contact::displayContact(){
    cout << "Contact ID: " << contactID << endl;
        cout << "Customer Name: " << contactName << endl;
        cout << "Customer Email: " << contactMail << endl;
        cout << "Customer Number: " << contactNumber << endl;
        cout << "Message: " << contactMessage << endl;
        cout << "Customer ID: " << cu->getCustomerID() << endl << endl;
}

// Class Ticket
class Ticket{
private:
    int ticketID;
    int ticketCount;
    string showTime;
    float price;
    Customer* cu;
public:
    Ticket();
```

```
    Ticket(int id, int tcount, string stime, float totprice, Customer*
pcu);
    void displayTicketDetails();
};

Ticket::Ticket(){}

Ticket::Ticket(int id, int tcount, string stime, float totprice, Customer*
pcu){
    ticketID = id;
    ticketCount = tcount;
    showTime = stime;
    price = totprice;
    cu = pcu;
}

void Ticket::displayTicketDetails(){
    cout<<"Ticket ID: "<<ticketID<<endl;
    cout<<"Number of Tickets: "<<ticketCount<<endl;
    cout<<"Show Time: "<<showTime<<endl;
    cout<<"Price: Rs "<<price<<endl;
    cout <<"Customer ID: " << cu->getCustomerID() << endl << endl;
}

// Class Payment
class Payment{
protected:
   int paymentID;
   float totalPrice;
public:
    Payment();
    Payment(int id, float price);
    void validate();
    void dispayPayment();
};

Payment::Payment(){}

Payment::Payment(int id, float price){
    paymentID = id;
    totalPrice = price;
}

void Payment::validate(){}

void Payment::dispayPayment(){
    cout<<"Payment ID: "<<paymentID<<endl;
    cout<<"Total Price: Rs "<<totalPrice<<endl<<endl;
}

// Class Card
class Card:public Payment{
protected:
    string cardHolderName;
    string cardNumber;
    string expirationDate;
public:
```

```
    Card();
    Card(int id, float price, string name, string number, string date);
    void displayCardDetails();
};

Card::Card(){}

Card::Card(int id, float price, string name, string number, string date)
    :Payment(id, price)
{
    cardHolderName = name;
    cardNumber = number;
    expirationDate = date;
}

void Card::displayCardDetails(){
    cout<<"Payment ID: "<<paymentID<<endl;
    cout<<"Total Price: "<<totalPrice<<endl;
    cout<<"Name on the card: "<<cardHolderName<<endl;
    cout<<"Card Number: "<<cardNumber<<endl;
    cout<<"Expiration Date: "<<expirationDate<<endl<<endl;
}

int main(void){

    cout << "-- Admin Details --" << endl;
        Admin ad001 = Admin(1, "Diwan Sachidu", "diwansachidu",
"diwansachidu@gmail.com");
        ad001.displayAdminDetails();

        cout << "-- Manager Details -- " << endl;
        Manager ma001 = Manager(1, "Kasun Perera", "kasun@gmail.com");
        ma001.displayManagerDetails();

        cout<< "-- Report Details--"<<endl;
        Report r1 = Report(001,"Income Report");
    Report r2 = Report(002, "List of Members");
    Report r3 = Report(003, "List of Movies");

    r1.displayReport();
    r2.displayReport();
    r3.displayReport();

        cout << "-- Movie Details --" << endl;
    Movie m1 = Movie(1,"Sonic the Hedgehog 2", "Action", "Jeff Fowler",
"Josh Miller,Patrick Casey,John Whittington", "Ben Schwartz,Idris
Elba,Colleen O'Shaughnessey");
    m1.displayMovieDetails();

    cout << "-- Now Showing Movie Detials --" << endl;
    NowShowingMovie n1 = NowShowingMovie(2, "Turning Red", "Animation",
"Domee Shi", "Rosana Sullivan,Searit Huluf,Sarah Streicher", "Rosalie
Chiang,Sandra Oh,Ava Morse", "2022-05-30");
    n1.displayMovieDetails();

    cout << "-- Comimg Soon Movie Detials --" << endl;
```

```
    ComingSoonMovie c1 = ComingSoonMovie(3, "Jurassic World Dominion",
"Adventure", "Colin Trevorrow", "Colin Trevorrow,Derek Connolly,Emily
Carmichael", "Sam Neill,Laura Dern,Jeff Goldblum", "2022-07-10");
    c1.displayMovieDetails();

        cout << "-- Customer Details --" << endl;
        Customer cu001 = Customer(1, "Kasun", "Chamara", "kasunchamara",
"kasunchamara@gmail.com", "0769841521");
        cu001.displayCustomer();

        cout<<"-- Feedback Details --"<< endl;
        Feedback fe001 = Feedback(1, "Sadaruwan Bandara",
"sadaruwan@gmail.com", "Demo Message", &cu001);
        fe001.displayFeedback();

        cout<<"-- Contact Details --"<< endl;
        Contact co001 = Contact(1, "Sadaruwan Bandara",
"sadaruwan@gmail.com", "0766894512", "Demo Message", &cu001);
        co001.displayContact();

        cout<<"-- Ticket Details --"<<endl;
        Ticket ti001 = Ticket(1, 02, "09:00 AM", 700, &cu001);
        ti001.displayTicketDetails();

        cout<<"-- Payment Details --"<<endl;
        Payment pa001 = Payment(1, 700);
        pa001.dispayPayment();

        cout<<" -- Card Details-- "<<endl;
        Card ca001 = Card(1, 700, "Kasun Chamara", "0769841521", "04/23");
        ca001.displayCardDetails();

        return 0;
}
```

## Individual Contribution

**As IT21166488 Maleesha K.L.D.D.S**

- Drew CRC cards for “Admin” and “Movie”.
- Identified the relationships between the “Admin” class and the “Movie” class.
- Identified inheritance classes of “Movie” Class.
- Drew class diagram for “Admin” and “Movie” with relationships to other classes.
- Coding the “Admin” class, “Movie” class, “Now Showing Movie” class and “Coming Soon Movie” class (Now Showing Movie and Coming Soon Movie are inheritance classes of the movie).

**As IT21164026 Wijayasooriya W.A.K.A**

- I created the CRC Cards for “Manager”,” Contact “and” Report” classes.
- I designed the class diagram for “Manager”,” Contact “and” Report” and identified relationships among those classes.
- I did all the c++ coding parts regarding “Manager”,” Contact “and” Report” classes.

**AS IT21164194 D.M.W.B. Thesarana Dissanayake**

- Created the CRC Card for the “Feedback” class.
- Created the class diagram for “Feedback”.
- Identified the relationship between the “Customer” and “Feedback” classes.
- Implemented the coding for the “Feedback” class.

**As IT21163272 Amanda Yasathri**

- Created CRC Card for the “Ticket” class.
- Have drawn the class diagram for the “Ticket”.
- Have created the code for the “Ticket”.

**AS IT21164644 D.N. Pathirathna**

- Designed the CRC Cards for the Customer class, Payment class and card class.
- Have drawn the Class diagram for the Customer class, Payment class and card class.
- Have implemented the codes for the Customer class, Payment class and card class.