Topic : Movie Booking System

Group no : MLB_WE_01.01_10

Campus : Malabe

Submission Date : 17/05/2022

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**
IT21247972	Kongahawatte D D	0776306494
IT21192296	W W B B Mendis	0719539375
IT21229084	Iroshan G.H.M	0773032542
IT21074622	Samaraweera S.M	0766061623
IT21210488	Kumara K.D.A.N	0713780408

System requirements for Online Book Store System

• The system should provide services for 24 hours and 7 days.

• Admin and the registered user must enter the correct username or email and password to enter the system.

• If an unregistered user wants to utilize the system, they must first register by submitting accurate information.

• Unregistered user must provide the E-mail address and the phone number.

• The registered user can re-register to the system. if there are any errors in the registration procedure.

• The administrator should be able to add and update movies to the system at any time. It should also be possible to remove it from the movie booking system.

• After each movie booking, the system should be updated.

• When a user searches for a movie, the information of the movie should be shown to the user.

• The registered user can add or remove tickets as user wish. At last total amount is displayed.

• Registered user can book movie ticket/s by entering credit or debit card details and confirm information.

• After that, registered user can check the booking details by the status displayed on the screen.

• Once the money transfer is successful, the booking details and transaction report must be added to the system as a new booking.

• Finally, the registered user can use the URL Link (download link) to download the E-ticket/s and E-ticket/s will be mailed to the E-mail address via auto generated E-mail. The phone number will also receive a verification SMS message.

Noun Analysis

- Customers can visit the online movie booking system using the website's URL.
- There are two types of customers. They are registered customers and unregistered customers.
- Unregistered customers can only browse and search movies. Registered customers can search, browse and book tickets.
- Unregistered customers can register to the system by creating an account. This is done by providing details such as name, address, NIC, email, username and a desired password).
- Registered customers can login to the system using the username and password. System then validates the entered username and password.
- Systems allows user to change password and to reset password if user has forgotten the password.
- Staff members has the ability to add new movies to the system and remove movies from the system and edit movie details.
- User can book a ticket by selecting a movie then selecting a theatre and selecting a preferred time.
- Movies can be of two types as now showing and coming soon.
- The customer has ability to create multiple bookings. System calculates total cost and shows the total price of the booking.
- Customers can then continue to the payment. Customer can select a desired payment method out of credit/debit cards, eZCash and loyalty points.
- System validates the entered payment details and a receipt and booking confirmation is sent to the user.
- Users can cancel a booking if needed by providing a valid reason. If the booking is cancelled within 24 hours of making the payment, a full refund will be given.
- If the transaction is successful, the booking and the transaction will be recorded in the database.

Identifying classes using Noun Analysis

- Customers, account, user - Redundant
- system - Out of scope
- URL - Out of scope
- registered customers - Class
- unregistered customers - Class
- movies – Class
- Booking - Class
- tickets - Redundant
- name, address, NIC, email, username, password - Attributes
- Staff members - Class
- details - Attributes
- theatre - Class
- time - Attribute
- types - Attributes
- total cost - Attributes
- payment - Class
- payment method - Attributes
- receipt - Attribute
- confirmation - Redundant
- reason - Attribute
- refund - Attribute
- transaction - Redundant
- database - Out of scope

Verb Analysis

- Customers can visit the online movie booking system using the website's URL.
- There are two types of customers. They are registered customers and unregistered customers.
- Unregistered customers can only browse and search movies. Registered customers can search, browse and book tickets.
- Unregistered customers can register to the system by creating an account. This is done by providing details such as name, address, NIC, email, username and a desired password).
- Registered customers can login to the system using the username and password. System then validates the entered username and password.
- Systems allows user to change password and to reset password if user has forgotten the password.
- Staff members has the ability to add new movies to the system and remove movies from the system and edit movie details.
- User can book a ticket by selecting a movie then selecting a theatre and selecting a preferred time.
- Movies can be of two types as now showing and coming soon.
- The customer has ability to create multiple bookings. System calculates total cost and shows the total price of the booking.
- Customers can then continue to the payment. Customer can select a desired payment method out of credit/debit cards, eZCash and loyalty points.
- System validates the entered payment details and a receipt and booking confirmation is sent to the user.
- Users can cancel a booking if needed by providing a valid reason. If the booking is cancelled within 24 hours of making the payment, a full refund will be given.
- If the transaction is successful, the booking and the transaction will be recorded in the database.

CRC Cards

Class Name: UnregisteredCustomer	
Responsibility	**Collaborators**
Create new account	
Search movies	Movie

Class Name: RegisteredCustomer	
Responsibility	**Collaborators**
Login to the system	
Search movies	Movie
Book tickets	Booking

Class Name: Payment	
Responsibility	**Collaborators**
Making New Payment	
Confirm Payment Details	RegisteredCustomer / Booking
View Booking Details	RegisteredCustomer / Booking

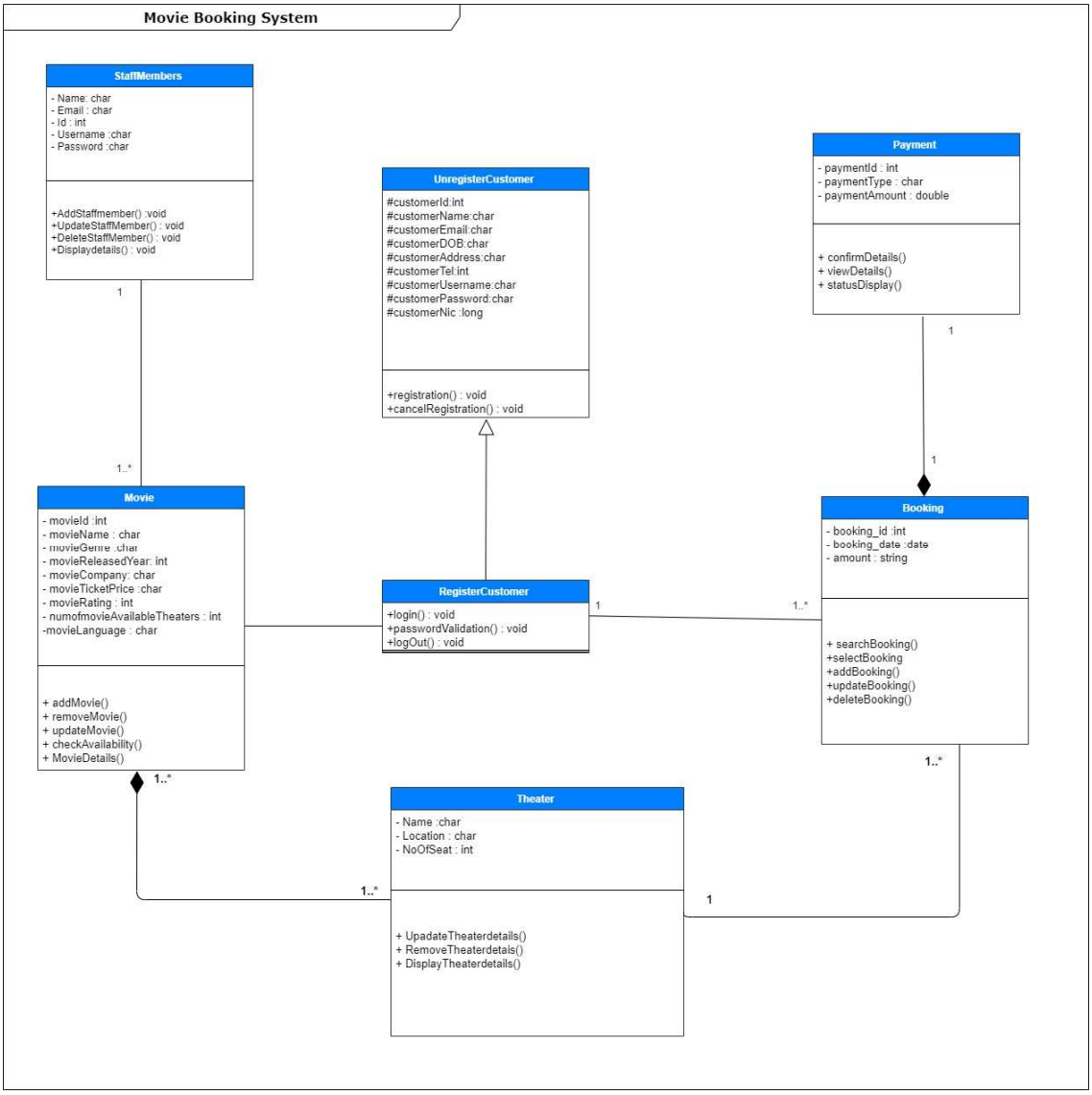
Class Name: Booking	
Responsibilities	**Collaboration**
Select booking	
Create booking	RegisteredCustomer
Confirm booking	Payment
Cancel booking	

Class Name: Theater	
Responsibilities	**Collaboration**
Accept Reservation	Booking
check theater to watch movies	Movie
Maintain Theaters details	

Class Name: StaffMember	
Responsibilities	**Collaboration**
Add and remove movies into system	Movie
Login to the system	
Maintain the system	

Class Name: Movie	
Responsibility	**Collaborators**
Adding movies to the system and removing moving from the system	
System update each booking	Booking
Showing the information of the movies to the customer.	RegisteredCustomer / UnregisteredCustomer

Class Diagram



Unregistered Customer class

UnregisteredCustomer.h

```
#include "Movie.h"
#define SIZE 2

class UnregisteredCustomer
{
private:
  //Class releationship
  Movie*movie[SIZE];

protected:
  int customerId;
  char customerName[30];
  char customerEmail[50];
  char customerDOB[10];
  char customerAddress[100];
  int customerTel;
  char username[10];
  char password[10];
  long customerNic;

public:

  UnregisteredCustomer();
```

```
  UnregisteredCustomer(int cid, const char cname[], const char cmail[], const char cdob[], const char caddress[], int ctel,const char uname[], const char pword[], int nic);


  void regisration();


  void cancelRegistration();


  ~UnregisteredCustomer();


};
```

UnregisteredCustomer.cpp

```
#include <cstring>
#include "UnregisteredCustomer.h"


using namespace std;
UnregisteredCustomer::UnregisteredCustomer()
{
  customerId = 0;
  strcpy_s(customerName, " ");
  strcpy_s(customerEmail, " ");
  strcpy_s(customerDOB, " ");
  strcpy_s(customerAddress, " ");
  customerTel = 0;
  strcpy_s(username, "");
  strcpy_s(password, "");
  customerNic=0;
}


UnregisteredCustomer::UnregisteredCustomer(int cid, const char cname[], const char cmail[], const
char cdob[], const char caddress[], int ctel, const char uname[], const char pword[], int nic)
{
  customerId = cid;
  strcpy_s(customerName, cname);
  strcpy_s(customerEmail, cmail);
  strcpy_s(customerDOB, cdob);
  strcpy_s(customerAddress, caddress);
  customerTel = ctel;
  strcpy_s(username,uname);
```

```
  strcpy_s(password,pword);
  customerNic=nic;
}
void UnregisteredCustomer::regisration()
{
}
void UnregisteredCustomer::cancelRegistration()
{
}
UnregisteredCustomer:: ~UnregisteredCustomer()
{
}
```

Registered Customer Class

RegisteredCustomer.h

```
#include "UnregisteredCustomer.h"
#include "Booking.h"
#include "Movie.h"
#define SIZE 2

class RegisteredCustomer: public UnregisteredCustomer
{
private:
  //Class Relationship
  Moive * Movie[SIZE];
  Booking *booking;

public:

  RegisteredCustomer();
  RegisteredCustomer(const char uname[25], const char cpwd[8]);
  void login();
  void passwordvalidation();
  void logout();
  ~RegisteredCustomer();
};
```

RegisteredCustomer.cpp

```
#include "RegisteredCustomer.h"
#include <cstring>

using namespace std;
RegisteredCustomer::RegisteredCustomer()
{
  strcpy_s(username, " ");
  strcpy_s(password, " ");
}
RegisteredCustomer::RegisteredCustomer(const char uname[25], const char cpwd[8])
{
  strcpy_s(username, uname);
  strcpy_s(password, cpwd);
}
void RegisteredCustomer::login()
{
 }
void RegisteredCustomer::passwordvalidation()
{
}
void RegisteredCustomer::logout()
{
}
RegisteredCustomer::~RegisteredCustomer()
{
}
```

Payment class

Payment.h

```
#include "Booking.h"
#define SIZE 2
class Payment {

private:
        int paymentId;
        char paymentType[20];
        double paymentAmount;

//Class Relationship
Booking* booking[SIZE];

public:
        Payment();
        Payment(int id, const char type[], double amount);
        void confirmDetails();
        void viewDetails();
        void statusDisplay();
};
```

Payment.cpp

```
#include "Payment.h"
#include <cstring>

Payment::Payment() {
	paymentId = 0;
	strcpy_s(paymentType, "");
	paymentAmount = 0;
	}

Payment::Payment(int id, const char type[], double amount) {
	paymentId = id;
	strcpy_s(paymentType, type);
	paymentAmount = amount;
	}

void Payment::confirmDetails()
{
}

void Payment::viewDetails()
{
}

void Payment::statusDisplay()
{
}
```

Booking Class

Booking.h

```
#include "movie.h"
using namespace std;

class Booking
{
private:
 int booking_ID;
 char booking_Date[20];
 double amount;

 Movie* movie;

public:
 Booking();
 Booking(int bID, const char bDate[], double amount);
 void searchBooking();
 void selectBooking();
 void addBooking();
 void editBooking();
 void deleteBooking();
};
```

Booking.cpp

```
#include "Booking.h"
#include <cstring>

Booking::Booking(){
  booking_ID = 0;
  strcpy_s(booking_Date, "");
  amount = 0;
}

Booking::Booking(int bID, const char bDate[], double amount){
  booking_ID = bID;
  strcpy_s(booking_Date, bDate);
  amount = amount;
}

void Booking::searchBooking()
{
}

void Booking::selectBooking()
{
}

void Booking::addBooking()
{
}
```

```
void Booking::editBooking()
{
}

void Booking::deleteBooking()
{
}
```

Theater Class

Theater.h

```
#include "Booking.h"
#include "Movie.h"

class Theater
{
private:
	char Name[20];
	char Location[15];
	int NoOfSeat;

	Booking *booking[5];
	Movie *movie[5];

public:
	Theater();
	Theater(const char name[],const char location[], int Noofseat);
	void AddTheater();
	void UpadteTheaterdeails();
	void RemoveTheaterdetails();
	void DisplayTheaterdetails();

};
```

Theater.cpp

```
#include "Theater.h"

#include <cstring>


Theater::Theater()
{

	strcpy_s(Name, "");
	strcpy_s(Location, "");
	NoOfSeat = 0;

}
Theater::Theater(const char name[], const char location[], int Noofseat)
{

	strcpy_s(Name,name);
	strcpy_s(Location,location);
	NoOfSeat = Noofseat;

}
void Theater::AddTheater()
{

}
void Theater::UpadteTheaterdeails()
{

}
void Theater::RemoveTheaterdetails()
{

}
void Theater::DisplayTheaterdetails()
{

	cout << "Theater Name - " << Name << endl;
	cout << "Theater Location - " << Location << endl;
	cout << "No of Seats - " << NoOfSeat << endl;

}
```

Staff Member Class

Staffmember.h

```
#include "Movie.h"

class Staffmember
{
private:
	char Name[20];
	char Email[30];
	int Id;
	char UserName[15];
	char Password[15];
	Movie* movie[5];



public:
	Staffmember();
	Staffmember(const char name[], const char email[], int id, const char Username[], const char
password[]);
	void AddStaffmember();
	void UpdateStaffmember();
	void DeleteStaffmember();
	void Displaydetails();

};
```

Staffmember.cpp

```
#include "Staffmember.h"

#include <cstring>



Staffmember::Staffmember()
{
	strcpy_s(Name, "");
	strcpy_s(Email, "");
	Id = 0;
	strcpy_s(UserName,"");
	strcpy_s(Password,"");

}
Staffmember::Staffmember(const char name[], const char email[],int id,const char Username[],const
char password[])
{
	strcpy_s(Name, name);
	strcpy_s(Email, email);
	Id = id;
	strcpy_s(UserName, Username);
	strcpy_s(Password, password);

}


void Staffmember::AddStaffmember()
{

}
void Staffmember::UpdateStaffmember()
{

}
void Staffmember::DeleteStaffmember()
{

}
void Staffmember::Displaydetails()
{
	cout << "Name -" << Name << endl;
	cout << "Id No -" << Id << endl;
	cout << "Email -" << Email << endl;
	cout << "UserName -" << UserName << endl;
	cout << "Password -" << Password << endl;

}
```

Movie Class

Movie.h

```
#include"unregisteredCustomer.h"
#include "RegisteredCustomer.h"
#include "Booking.h"
#define SIZE 2
class Movie {
private:
int movieId;
char movieName[15];
char movieGenre[15];
int movieReleasedYear;
char movieCompany[15];
double movieTicketPrice;
int movieRating;
int numofmovieAvailableTheaters;
char movieLanguage[15];
//Class Relationship
Booking* booking[SIZE];
UnregisteredCustomer *unregisteredcustomer[SIZE];
RegisteredCustomer *registeredcustomer[SIZE];



public:
Movie();
Movie(int M_movieId, char *M_movieName, char *M_movieGenre, int M_movieReleasedYear,
```

```
char *M_movieCompany, double M_movieTicketPrice, int M_movieRating,int
M_numofmovieAvailableTheaters, char *M_movieLanguage);

void addMovie();

void removeMovie();

void updateMovie();

void checkAvailability();

void MovieDetails();

~Movie();

};
```

Movie.cpp

```
#include "Movie.h"
#include <cstring>


Movie::Movie() {
movieId = 0;
strcpy_s(movieName, "");
strcpy_s(movieGenre, "");
movieReleasedYear = 0;
strcpy_s(movieCompany, "");
movieTicketPrice = 0.00;
movieRating = 0;
numofmovieAvailableTheaters=0;
strcpy_s(movieLanguage, "");
}
Movie::Movie(int M_movieId, char *M_movieName, char *M_movieGenre, int
M_movieReleasedYear,
char *M_movieCompany, double M_movieTicketPrice, int M_movieRating,int
M_numofmovieAvailableTheaters, char *M_movieLanguage)
{
movieId = M_movieId;
strcpy_s(movieName,M_movieName);
strcpy_s(movieGenre,M_movieGenre);
movieReleasedYear = M_movieReleasedYear;
strcpy_s(movieCompany, M_movieCompany);
movieTicketPrice = M_movieTicketPrice;
movieRating = M_movieRating;
numofmovieAvailableTheaters=M_numofmovieAvailableTheaters
strcpy_s(movieLanguage,M_movieLanguage);
```

```
}
void Movie::addMovie()
{
}
void Movie::removeMovie()
{
}
void Movie::updateMovie()
{
}
void Movie::checkAvailability()
{
}
void Movie::MovieDetails()
{
}
Movie::~Movie()
{
}
```

main.cpp

```
#include <iostream>
#include "Booking.h"
#include "RegisteredCustomer.h"
#include "UnregisteredCustomer.h"
#include "Movie.h"
#include "Payment.h"
#include "Theater.h"
#include "StaffMember"

int main()
{
  //Creating objects
  RegisteredCustomer*registeredCustomer = new RegisteredCustomer();



  UnregisteredCustomer*unregisteredCustomer = new UnregisteredCustomer();

  Movie*movie = new Movie();

  Booking*booking = new Booking();

  Payment*payment = new Payment();

  Theater*theater = new Theater();

  StaffMember*staffmember = new StaffMember();
```

```
//Function calling
registeredCustomer->login();
registeredCustomer->logout();
registeredCustomer->passwordValidation();

unregisteredCustomer->regisration();
unregisteredCustomer->cancelRegistration();

staffmember->AddStaffmember();
staffmember->UpdateStaffMember();
staffmember->DeleteStaffMember();
staffmember->Displaydetails();

movie->addMoive();
movie->removeMovie();
movie->UpdateMovie();
movie->CheckAvailability();
movie->MovieDetails();

theater->UpdateTheaterDetails();
theater->RemoveTheaterDetails();
theater->DisplayTheaterDetails();

booking->searchBooking();
booking->selectBooking();
booking->addBooking();
booking->updateBooking();
booking->deleteBooking();
```

```
    payment->confirmDetails();
    payment->viewDetails();
    payment->statusDisplay();



    //Deleting dynamic objects
    delete registeredCustomer;
    delete unregisteredCustomer;
    delete staffmember;
    delete movie;
    delete theater;
    delete booking;
    delete payment;


    return 0;
}
```

Individual Contributions

Registration No	Name	Contribution
IT21247972	Kongahawatte D D	1. UnregisteredCustomer.h 2. UnregisteredCustomer.cpp 3. RegisteredCustomer.h 4. RegisteredCustomer.cpp 5. main.cpp
IT21192296	W W B B Mendis	1. Payment.h 2. Payment.cpp 3. Verb analysis
IT21229084	Iroshan G.H.M	1. Theater.h 2. Theater.cpp 3. Staffmember.h 4. Staffmember.cpp
IT21074622	Samaraweera S.M	1. Booking.h 2. Booking.cpp 3. Noun Analysis 4. Identifying classes using noun analysis
IT21210488	Kumara K.D.A.N	1. Moive.h 2. Movie.cpp 3. System requirements