The BUCKS Centre for the Performing Arts

By

Felix Saraiva

CO554: Object Oriented Programming

bucks new university

School of Business, Law and Computing

Department of Computing

Object Oriented Programming: CW1

Module Title:	Object Oriented Programming	**Module Code:**	CO554
Assignment No/Title:	CW1	**Assessment Weighting:**	100%
Submission Date:	Friday 25th January 2019 by 14:00 Week 17	**Target Feedback Date:**	+ 2 Weeks
Module Leader:	Mike Everett	**Degree/Foundation:**	Degree
Student ID:	21713532	**Student Name:**	Felix Monteiro Saraiva
Course:	Software Engineering		

Index:

CASE STUDY

The Bucks Centre for the Performing Arts (BCPA), an entertainment venue, wants to allow customers to order tickets through the Internet. This new Online Ticketing System (OTS) must allow the customer to view a list of upcoming events, or view scheduled shows by date, select seat(s) from a seating chart, hold the seat(s) while they complete their selection, and purchase the selected seats.

The BCPA has contracts with several ticket agents at various ticket outlets. These contracts define the agent commissions and the terms and conditions for the sale of tickets. The contract is the agent's authorisation to use the OTS. Associated with the contract is a sales agreement that defines the seats that are assigned to the agent to sell. One agent may not sell seats from another agent's assigned seats. The seat assignments apply to a set of seats for the specified date range, not for specific shows.

The venue manager is responsible for managing promotions for each show. A promotion defines the pricing structure for seats in a show. A pricing structure must accommodate differences for adult, student, child, and senior citizen seating. Discounts are defined per show. A promotion can be unique to each showing of an event. For example, the promotion for a Saturday matinee may be different than the promotion for the Saturday evening show. A promotion can be specific to seats within a show. A promotion may also be reused for many shows for numerous events. The system must be capable of displaying the price for each seat on the seating chart Assigning seats to promotions must be dynamic; that is, seats may be redefined into different promotions if a show sells either better or worse than anticipated.

The system must allow the venue manager to cancel, reschedule, or add events and shows, and to allow changes to the maximum-seats per-customer value for each show.

A consumer will access the OTS via the World Wide Web. The user interface will be implemented with an OO language applications; that is, without browsers and hypertext mark-up language (HTML).

Consumers must provide a valid sign on and password. Then they must provide or verify their customer profile information. The customer profile includes address information for mailing the tickets. This information is also used to target customers for special promotions. The system must keep this customer information on file so that returning consumers can use their existing sign on and password, and avoid re-entering the information.

Consumers are then presented with the choice between selecting a show using a list of upcoming events or a list of shows for a given date range. Once Consumers select a show, they are offered the choice of interactively selecting a seat(s) or having the system select the best available seat(s) for a price range.

When users select interactive seat selection, they are presented with a floor chart of the Concert Hall. The seating chart is coloured according to the status of the seats for

each show; for example, available, held, or sold. Selecting a seat places, it on hold so that no one else can select it while the users complete their transactions.

Deselecting a seat removes the hold and makes the seat available again for other users. Users can select up to the maximum allowed seats per customer set for the show by the venue manager.

When users select automatic seat selection, they must provide a price range and the number of seats desired. The system will then attempt to select the "best" seats available. Once the attempt is completed, the system will either display the resulting seating chart with the selected seats highlighted, or an appropriate message. Users can then either accept the selection and change the criteria, or switch to interactive seat selection.

When consumers select a seat, the system will "hold" the seat so that it will appear unavailable to subsequent customers. After the consumers pay for the seats, the system will mark the seat(s) reserved and generate a ticket(s). If consumers choose not to purchase the seat(s), then the system will remove the hold, thus making the seat(s) available again.

In a transaction, consumers can purchase a single ticket or multiple tickets at varied prices. For some shows, volume discounts are available. For example, ticket purchases of £100 or more might receive a 10 % discount, or buying 6 or more tickets might qualify the consumer for a 15 % discount. In all cases, each ticket must be tracked separately, with its associated price and applied discount and seat assignment.

Credit card will be the only form of currency accepted, so the system must have the ability to validate a card number and accept or reject the purchase. For this case study, assume that all credit card purchases are approved.

Ticket agents interact with the OTS using the World Wide Web. After signing onto the application as an agent, the agent interacts with the system on behalf of the customer. Once agents provide the customer profile information, the same initial choices of event selection by upcoming events or date range are displayed, Agents use the same features for seat selection as the consumer, with one additional feature; agents are able to see only the seats assigned to them. Agents can also see the total number of tickets sold for the currently displayed show or all shows for a date range.

Once the seats are placed in a hold state, an internal clock that sounds an alarm after five minutes and prompts users about continuing the transaction. The alarm then sounds every minute for three minutes, after which time all "held" seats are released if the transaction is not completed. This same feature applies to the consumer.

Read the BUCKS Centre for the Performing Arts case study below. As with any real-world problem statement, you will find that the information is not always presented in the most logical manner. Information on any given topic may be scattered across discussions of related topics.

Introduction

This assignment runs in parallel with my initial investigation into the Bucks Center for the Performing Arts case study, reproduced before. I was required to use the example sub-model supplied, as base for demonstrating my individual ability to partially implement an Object Orientated (OO) system.

In this project I must build a C++ console application that simulates a real-life system that sells tickets. Also, are used schemas, diagrams, pseudocode, and testing plans to increase the quality of the system.

This program is organized as a collection of cooperative, dynamic objects, each of which is an instance of some class. Object composition will be used so objects will contain other objects in their instance variables and objects are polymorphic this means that objects of different types are accessed through the same interface. Each type can provide its own, independent implementation of this interface. The classes are organized into a hierarchy. Also, this program binds together the data and functions that manipulate the data, using the concept of encapsulation.

Requirements

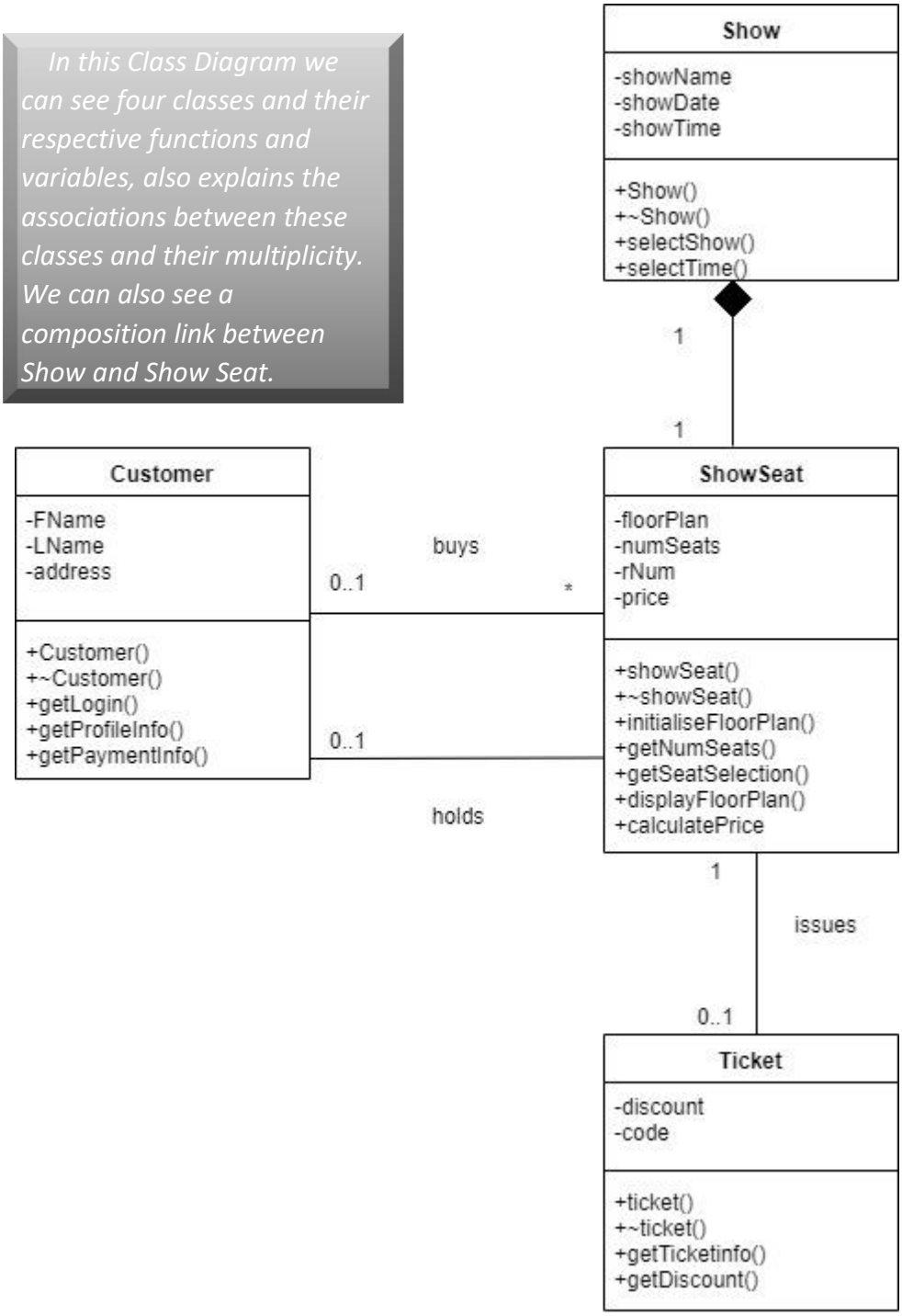
In order to produce a good quality program, it must follow a simple and clear Requirement List so the developer can always keep track of all the Functions that the program must have. Also, it is useful to keep in mind the Non-Functions which represent a major importance to the program.

Here we can see a Table with the Functions and non-Function of this project:

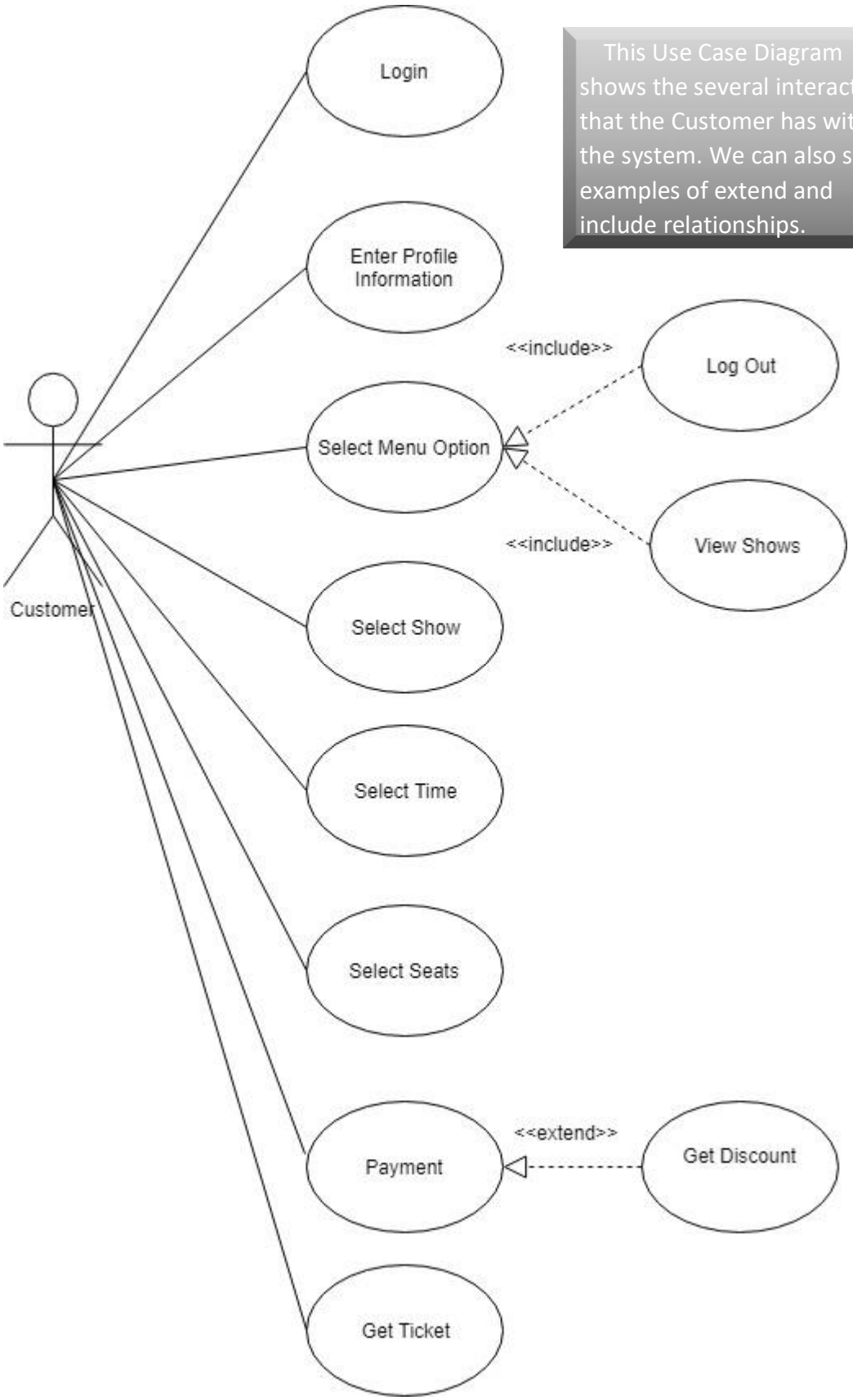
	Functions	***Non-Function***
1.	Allow user to Login	Usability
2.	Allow User to enter profile information	Security
3.	Allow user view shows, buy tickets or log out	Accessibility
4.	Allow user to select a show	Performance
5.	Displays Sessions	Capacity
6.	Allow user to select a session	Extensibility
7.	Allow user to select number of seats	
8.	Displays theatre room	
9.	Allow user to select row and column of the seats	
10.	Show price of the tickets based on the spot	
11.	Displays the discount options	
12.	Allow user to choose a discount if he possesses one	
13.	Print ticket with all the information	

Class Diagram

In this Class Diagram we can see four classes and their respective functions and variables, also explains the associations between these classes and their multiplicity. We can also see a composition link between Show and Show Seat.

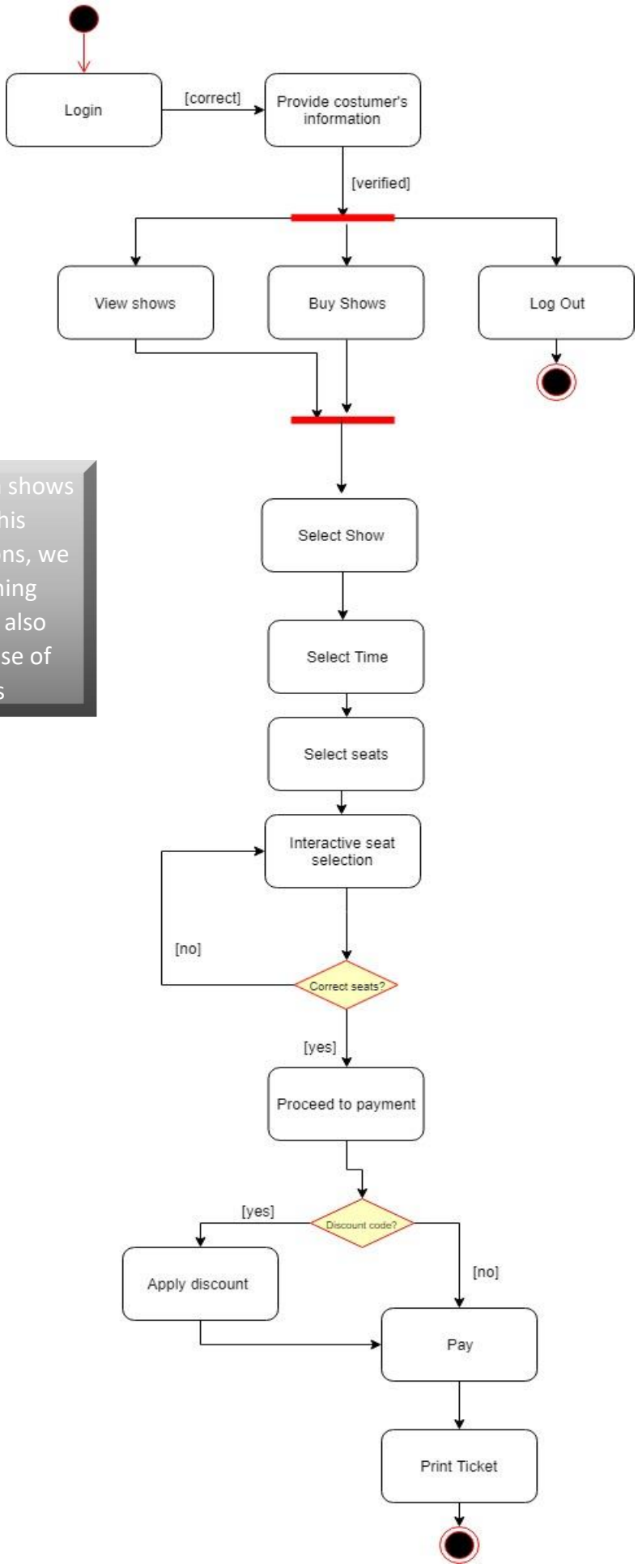


Use Case Diagram



This Use Case Diagram shows the several interactions that the Customer has with the system. We can also see examples of extend and include relationships.

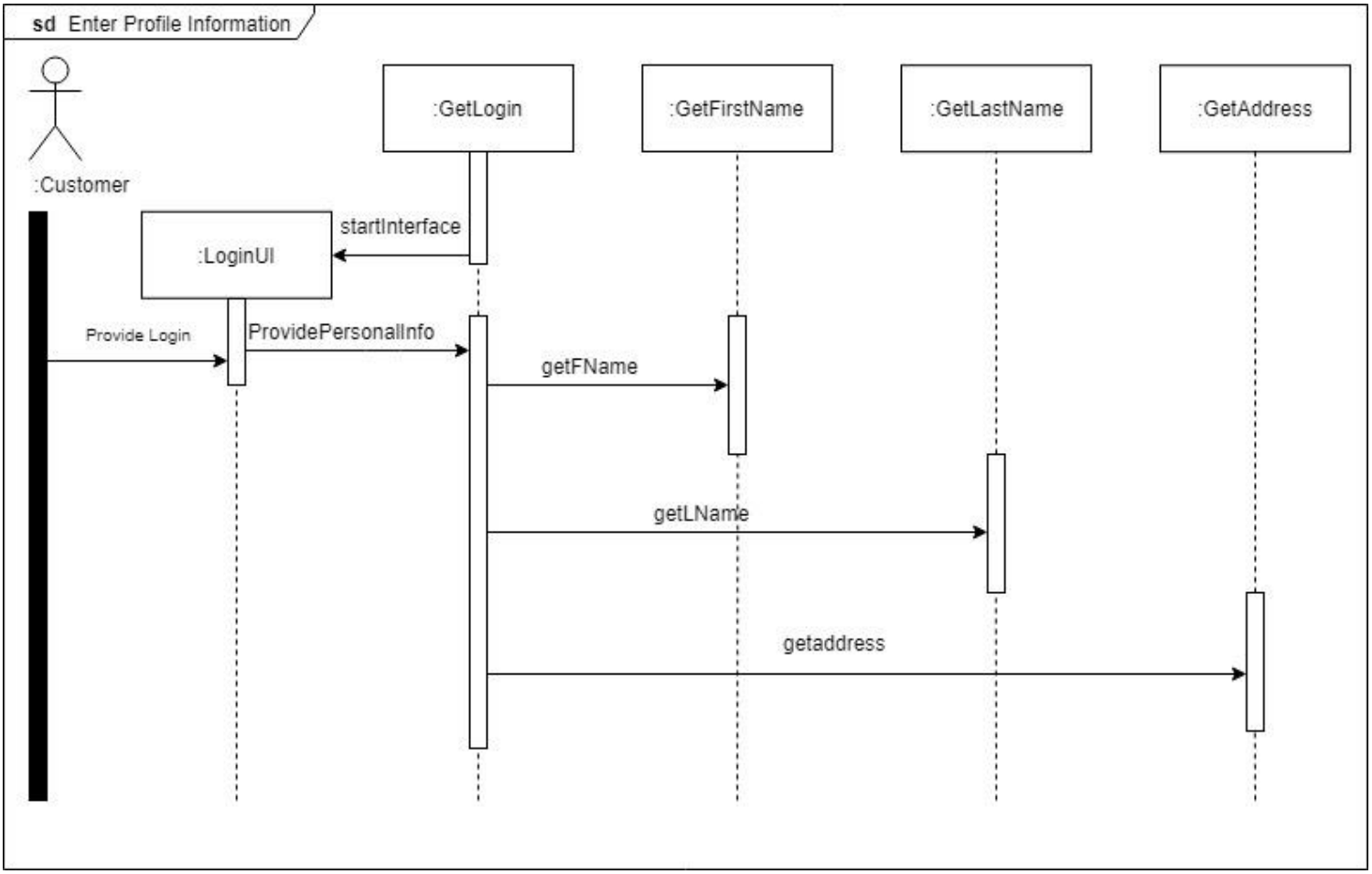
Activity Diagram



This Activity diagram shows the complete path of this system and all its options, we can see what is happening and when. And we can also understand the response of system if the user gives

Sequence Diagram

This sequence diagram represents the sequence of the use case 'Enter Profile Information', it starts with the Login because in this project both of this use cases are connected and share the same User Interface.



Action & Response Diagram

In this table we intend to clarify the actions of the user and the responses of the system to these actions. With this we can build a more rich and organized Pseudocode.

Actor: Customer	***System response***
1.None	2.Display login prompt
3.Enter Username and Password	4.Validates
5.None	6.Displays Profile Information prompt
7.Enters Profile Information	8.Opens Main Menu
9.Chooses Buy Tickets Option	10.Displays Shows
11.Chooses a Show	12.Display Sessions
13.Chooses a Session	14.Display the number of seats desired prompt
15.Enters Number of Seats desired	16.Displays Select seats prompt
17.Enter a column and row number for each seat	18.Displays Discount prompt
19.Enters Discount	20.Displays Payment information prompt
21.Enters Payment Information	22.Validates
23.None	24.Prints Ticket

Pseudocode

By using a pseudocode that sets out a detailed plan of the algorithms within the system, it will be easier to program.

Program: CW1

```
Include iostream, string,customer.h, ticket.h, show.h, showSeat.h, Windows.h, stdio.h
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

Function:Main()

    Initialise variable String a,b,c,f,g,h
    Initialise variable Int d
    Initialise variable Double e,l
    Initialise variable Char ch, terminator
    Initialise object showSeat SEAT
    Initialise object Customer CUST
    Initialise object Show SHOW
    Initialise object ticket TICK

    Call CUST getLogin()
    Call CUST getProfileInfo(f,g,h)
    Clear screen
    Do
        Print main menu layout
        Print 1. option buy tickets for upcoming shows
        Print 2. View shows
        Print 3. log out
        Prompt user to choose a menu choice number
        Enter input into ch variable
        While ch is not valid
            Prompt user to select a valid choice number
            Enter input into ch variable
        End while
        If input =2
            Print all the shows with their dates and times
            Wait for 10 seconds
            Clear screen
        End if
        If input=3
            Exit
        End if
    While input is not 1
    Do
        Call SHOW selectShow(a,b)
        Call SHOW selectTime(c)
        Do
            Prompt user if they are happy with their choice
            Enter input into ch variable
        While is not valid
        Clear buffer
        Clear screen
```

```
    While ch is valid
    Call SEAT initialiseFloorPlan()
    d= SEAT getNumSeats()
    e=SEAT getSeatSelection()
    print final price without discount
    wait 3 seconds
    clear screen
    initialize l variable
    l= TICK getDiscount(e,l)
    wait 2 seconds
    Call CUST getPaymentInfo()
    Clear screen
    Call TICK getTicketinfo(f,g,h,a,b,c,e,d,l)
    Clear buffer
    Pause system
    Return Exit_Sucess
End Main Function
```

Class: customer

Include iostream, iomanip, string,Windows.h,time.h

//

Declare customer class

Declare public variables:

```
    customer()
    ~customer()
    void getLogin()
    void getProfileInfo()
    void getPaymentInfo()
```

Declare protected variables:

```
    string FName
    string LName
    string address
```

//

Function:customer()

```
    Initialize FName
    Initialize LName
    Initialize address
```

End function

//

Function: ~customer()

End function

//

Function:getLogin()

```
    Declare string username variable
    Declare string password variable
    Prompt user to enter username
    Enter input into username variable
While username length less than 10 char
    Prompt user to re-enter username
    Enter input into username variable
    Prompt user to enter password
    Enter input into password variable
While password length less than 10 char
    Prompt user to re-enter password
```

```
    Enter input into password variable
End function
////////////////////////////////////////////////////////////////////////////////////////////////////
Function:getProfileInfo()
    Prompt user to enter First Name
    Enter input into FName variable
    Prompt user to enter Last Name
    Enter input into LName variable
    Prompt user to enter address
     Enter input into address variable
End function
////////////////////////////////////////////////////////////////////////////////////////////////////
Function:getPaymentInfo()
    Declare string Busername variable
    Declare string Bpassword variable
    Prompt user to enter Bank username
    Enter input into Busername variable
While Busername length less than 10 char
    Prompt user to re-enter Bank username
    Enter input into Busername variable
    Prompt user to enter Bank password
    Enter input into Bpassword variable
While Bpassword length less than 10 char
    Prompt user to re-enter Bank password
    Enter input into Bpassword variable
End function
```

Class: show

```
Include iostream, iomanip, string
////////////////////////////////////////////////////////////////////////////////////////////////////
Declare public variables:
    Show()
    ~Show()
    selectShow()
    selectTime()
Declare protected variables:
    string showName
    string showDate
    string showTime
////////////////////////////////////////////////////////////////////////////////////////////////////
Function:Show ()
    Initialize showName
    Initialize string showDate
    Initialize showTime
End function
////////////////////////////////////////////////////////////////////////////////////////////////////
Function: ~Show()
End function
////////////////////////////////////////////////////////////////////////////////////////////////////
Function: selectShow(string& showName, string &showDate)
     Declare char ch variable
     Declare char terminator variable
     Clear screen
     Output a list of shows
     Prompt user to enter a choice number
```

Enter choice into ch variable
While choice input is valid
Prompt user to enter valid choice number
Enter choice into ch variable
End while
Switch(ch variable)
1: enter “Star wars: the Musical” into showName variable, and “20/05/2013” into showDate variable
2: enter “Les Miserables” into showName variable, and “21/05/2013” into showDate variable
3: enter “The Phantom of the Opera” into showName variable, and “22/05/2013” into showDate variable
end switch
enter showName reference variable into showName class variable
enter showDate reference variable into showDate class variable
End function
//
Function: selectTime(string& showTime)
Declare char ch variable
Declare char terminator variable
Clear screen
Output a list of times
Prompt user to enter a choice number
Enter choice into ch variable
While choice input is valid
Prompt user to enter valid choice number
Enter choice into ch variable
End while
Switch(ch variable)
1: enter “1pm show” into showTime variable
2: enter “7pm show” into showTime variable
end switch
enter showTime reference variable into showTime class variable
End function

Class: showSeat

Include iostream, iomanip, string, conio.h
//
Declare public variables:
showSeat()
~showSeat()
initialiseFloorPlan()
getNumSeats()
getSeatSelection()
Declare protected variables:
Char floorplan [7][6]
Int numSeats
Double price
Declare private variables:
displayFloorPlan(int r, int c)
calculatePrice(double price)
//
Function: showSeat()
For int r=0 to 7

For int c=0 to 6

```
                floorPlan[r][c]=0
            end for
    end for
    initialize numSeats variable
    initialize rNum variable
    initialize price variable
End function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function: ~showSeat()
End Function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function: initializeFloorPlan()
    For int r=0 to 7

            For int c=0 to 6
                floorPlan[r][c]='A' character
            end for
    end for
End function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function: getNumSeats()
    Do
            Prompt user to enter number of seats
            Enter input into numSeats variable
    While input is not a number between 1 and 8
    Return numSeats
End function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function: getSeatSelection()
    Declare char ch variable
    Declare and initialize int r variable
    Declare and initialize int c variable
    Call displayFloorPlan(r,c)
    Do
        For int i into numSeats
            Declare char terminator variable
            Do
                Prompt user to enter column number between 1 and 7
                Enter input into r variable
            While input is column number is invalid
            Enter r variable into rNum
            Do
                Prompt user to enter column number between 1 and 6
                Enter input into c variable
            While input is column number is invalid
            While floorPlan[r][c]='H'
                Do
                    Prompt user to re-enter a row number between 1 and 7
                    Enter input into r variable
                While r is less than 1 or bigger than 7
                Do
                    Prompt user to re-enter a column number between 1 and 6
                    Enter input into c variable
                While c is less than 1 or bigger than 6
            floorPlan[r][c]='H'
            call calculatePrice(price)
```

```
        End for
        Clear screen
        Call displayFloorPlan(r,c)
        Do
                Prompt user to decide whether or not they are happy with their selection
                Enter input into ch variable
        While choice input is invalid
        If choice is n/N
                Call initialiseFloorPlan()
                Prompt user to reselect their seats
                Re-initialise price variable
        End if
    While choice is y/Y
    Return price
End Function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function: displayFloorPlan(int r, int c)
    Declare char terminator variable
    Print column numbers from 1 to 6
    For r int=0 to 7
        Prompt r+1
        For c int=0 to 6
            Print floorplan[r][c]
        End for
    Print floor plan key
End function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function: calculatePrice(double price)
    If rNum is between 1 and 4
        Add 20 to price
    End if
    If rNum is between 5 and 7
        Add 30 to price
    End if
End function
```

Class: ticket

```
Include iostream, iomanip, string
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Declare public variables:
    ticket()
    ~ticket ()
    getTicketinfo(string &FName, string &LName, string &address, string & showName, string
    &showDate, string &showTime,double price,int numSeats, double discount)
    getDiscount(double price,double discount)
    declare char ch variable
Declare private variables:
    Declare and initialize discount variable
    Declare char code variable
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function:ticket
End Function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function:~ticket
End Function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
Function: getTicketinfo(string &FName, string &LName, string &address, string & showName, string
&showDate, string &showTime,double price,int numSeats, double discount)
    Declare char terminator variable
    Print ticket layout
    Print showName
    Print showDate
    Print showTime
    Print numSeats
    Print price with discount
    Print Ticket Purchaser Names
    Print Ticket purchaser address
End Function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
Function: getDiscount(double price, double discount)
 Do
    Print 3 discount options //(10%,30%,50%)
    Prompt user to choose a discount option
    Enter input into ch variable
    While input is invalid
        Prompt user to select a valid discount choice
        Enter input into ch variable
    End while
    If input =1
        Discount=price*0.9
        Prompt user to enter discount coupon code
        Enter input into code variable
        Print processing information…
        Wait 3 seconds
        Print code accepted
    End if
    If input =2
        Discount=price*0.7
        Prompt user to enter discount coupon code
        Enter input into code variable
        Print processing information…
        Wait 3 seconds
        Print code accepted
    End if

    If input =3
        Discount=price*0.5
        Prompt user to enter discount coupon code
        Enter input into code variable
        Print processing information…
        Wait 3 seconds
        Print code accepted
    End if

    If input =4
        Discount=price
        Print Thank you
    End if
  While input is equal to 1, 2 and 3
       Return discount
      Return price
End Function
```

Implemented Code

Cw1.cpp

```
#include "pch.h"
#include <iostream>
#include <string>
#include "customer.h"
#include "ticket.h"
#include "show.h"
#include "showSeat.h"
#include <Windows.h>
#include <stdio.h>


int main()
{
       system("color 8f");//changes the color of the system

       /////////////////////////////////////////////////////////////
       //Initial variable declarations
       string a,
              b,
              c,
              f,
              g,
              h
              ;

       int d;
       double e, l;//discount

       char ch,
              terminator;

       /////////////////////////////////////////////////////////////
       //Objects
       showSeat SEAT;
       Customer CUST;
       Show SHOW;
       ticket TICK;
       /////////////////////////////////////////////////////////////
       //Deals with getting customer's information

       CUST.getLogin();
       CUST.getProfileInfo(f,g,h);

       /////////////////////////////////////////////////////////////
       //Displays Main Menu, and allows customer to choose option

       system("CLS");
       do
       {
              cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ MAIN
MENU ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n" << endl;

              cout << "1. Buy tickets for upcoming shows" << endl;
              cout << "2. View Shows" << endl;
              cout << "3. Log out\n" << endl;
              cout << "Please enter a menu choice number: ";
```

```
            cin.get(ch);

            while (ch != '1' && ch != '2' && ch != '3')
            {
                    cin.clear();
                    cin.ignore(100, '\n');// ensures buffer is completrly clear
(if, say, the user has previosly input a long string)
                    cout << "Please select a valid menu choice number: ";
                    cin.get(ch);
            }
            if (ch == '2')
            {
                    cout << "\n";
                    cout << "THE SHOWS WILL BE DISPLAYED FOR 10 SECONDS!\n";
                    cout << "1. Star Wars: The Musical (20/05/2013)      1pm &
7pm" << endl;
                    cout << "2. Les Miserables (21/05/2013)              1pm &
7pm" << endl;
                    cout << "3. The Phantom of the Opera (22/05/2013)    1pm &
7pm" << endl;

                    Sleep(10000); //Sleeps for 10000 ms or 10 sec

                    system("CLS");
            }
            if (ch == '3')
            {
                    return EXIT_SUCCESS;
            }
        } while (ch!='1');


        ////////////////////////////////////////////////////////////////////////////
//////////////////////////
        //Deals with selecting a show
        do
        {
            SHOW.selectShow(a, b);
            SHOW.selectTime(c);

            do
            {
                    cout << "\nAre you happy with yout choice (Y=Yes, N=No)?: ";
                    cin.get(ch);

            } while (ch!='Y' && ch!='y' &&ch!='N' && ch!='n');
            cin.get(terminator);//clears buffer
            system("CLS");
        } while (ch=='N' || ch=='n');
        ////////////////////////////////////////////////////////////////////////////
//////////////////////////
        //Deals with selecting seat and calculating price of seats

        SEAT.initialiseFloorPlan();
        d = SEAT.getNumSeats();
        e=SEAT.getSeatSelection();
        cout << " Your final price is: " << (char)156 << e << endl;
        cin.get(terminator);//clears buffer
        Sleep(3000); //Sleeps for 3000 ms or 3 sec
        system("CLS");
        ////////////////////////////////////////////////////////////////////////////
//////////////////////////
```

```
        //Deals with discounts, payment and ticket generation

        l = 0;
        l = TICK.getDiscount(e, l);
        Sleep(2000);
        system("CLS");
        CUST.getPaymentInfo();
        system("CLS");
        TICK.getTicketinfo(f,g,h,a,b,c,e,d,l);
        cin.get(terminator);//clears buffer

        ////////////////////////////////////////////////////////////////////////////
///////////////////////////
        system("pause");
        return EXIT_SUCCESS;





}
```

customer.h

```
#include <iostream>
#include <iomanip>
#include <string>
#include <Windows.h>
#include <time.h>

using namespace std;

class Customer
{
public:
	Customer();
	~Customer();
	void getLogin();
	void getProfileInfo(string &FName, string &LName, string &address);
	void getPaymentInfo();

protected:
	string FName;
	string LName;
	string address;

};
////////////////////////////////////////////////////////////////////////////////
////////////////////
//constructor
Customer::Customer()
{

	FName = "";
	LName = "";
	address = ""; //initialise variables

}
////////////////////////////////////////////////////////////////////////////////
////////////////////
//destructor
Customer::~Customer()
{
}
////////////////////////////////////////////////////////////////////////////////
////////////////////
//Deals with the Customer's Login
void Customer::getLogin()
{
	string username;
	string password;



	cout << "\n~~~
```

```
       while (username.length()>10)
       {
              cout << "Your username should be no more than 10 characters long."
<< endl;
              cout << "Please re-enter yout username";
              getline(cin, username);
       }

       cout << "Enter password: ";
       getline(cin, password);

       while (password.length() > 10)
       {
              cout << "Your password should be no more than 10 characters long."
<< endl;
              cout << "Please re-enter yout username";
              getline(cin, password);
       }
}
/////////////////////////////////////////////////////////////////////////////////
///////////////////
//Gets the Customers Personal Information
void Customer::getProfileInfo(string &FName, string &LName, string &address)
{



       cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ENTER PROFILE
INFORMATION ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n" << endl;
       cout << "As you are a new customer, you must provide your profile
information.\n";
       cout <<
"_____________________________________________________________________\n";
       cout << "Enter your first name: \n";
       getline(cin, FName);


       cout << "Enter your last name: \n";
       getline(cin, LName);


       cout << "Enter your address: \n";
       getline(cin, address);



}
/////////////////////////////////////////////////////////////////////////////////
///////////////////
//Deals with the transaction
void Customer::getPaymentInfo()
{
       string Busername;
       string Bpassword;

       cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ENTER PAYMENT
INFORMATION ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n\n" << endl;

       cout << "Enter Bank username: ";
       cin >> Busername;

       while (Busername.length() > 10)
```

```
        {
                cout << "Your username should be no more than 10 characters long."
<< endl;
                cout << "Please re-enter yout username";
                cin >> Busername;
        }

        cout << "\nEnter Bank password: ";
        cin >> Bpassword;

        while (Bpassword.length() > 10)
        {
                cout << "Your password should be no more than 10 characters long."
<< endl;
                cout << "Please re-enter yout username";
                cin >> Bpassword;
        }

        cout << "Processing information..." << endl;
        Sleep(3000); //Sleeps for 3000 ms or 3 sec
        cout << "Transaction accepted!" << endl;
}
```

show.h

```
#include <iostream>
#include <iomanip>
#include <string>

using namespace std;

class Show
{
public:
	Show();
	~Show();
	void selectShow(string & showName, string &showDate);
	string selectTime(string &showTime);

protected:
	string showName;
	string showDate;
	string showTime;
};

/////////////////////////////////////////////////////////////////////////////////
///////////////////
//constructor
Show::Show()
{
	showName = "";
	showDate = "";
	showTime = "";
}
/////////////////////////////////////////////////////////////////////////////////
///////////////////
//destructor
Show::~Show()
{
}

/////////////////////////////////////////////////////////////////////////////
//customer selects upcoming show
void Show::selectShow(string & showName, string &showDate)
{
	char ch;
	char terminator;

	system("CLS");

	cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT AN
UPCOMING SHOW ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n" << endl;

	cout << "1. Star Wars: The Musical (20/05/2013)" << endl;
	cout << "2. Les Miserables (21/05/2013)" << endl;
	cout << "3. The Phantom of the Opera (22/05/2013)" << endl;

	cin.clear();
	cin.ignore(100, '\n');// ensures buffer is completrly clear (if, say, the
user has previosly input a long string)
	cout << "Please select a show number: ";
	cin.get(ch);

	while (ch != '1'&& ch != '2'&& ch != '3')
	{
```

```
		cin.clear();
		cin.ignore(100, '\n');
		cout << "Please select a valid show number: ";
		cin.get(ch);
	}

	switch (ch)
	{
	case '1':showName = "Star Wars: The Musical", showDate = "20/05/2013";
		break;
	case '2':showName = "Les Miserables", showDate = "21/05/2013";
		break;
	case '3':showName = "The Phantom of the Opera", showDate = "22/05/2013";
		break;

	}

	this->showName = showName;
	this->showDate = showDate;//enters reference variable into class variable

	cin.get(terminator);//clears buffer
}

/////////////////////////////////////////////////////////////////////////////////
///////////////////
//customer selects 1pm/7pm showing
string Show::selectTime(string &showTime)
{
	char ch;
	char terminator;

	system("CLS");

	cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT SHOW
TIME ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n" << endl;

	cout << "1. 1pm show" << endl;
	cout << "2. 7pm show" << endl;



	cout << "Please select a show time number (1/2): \n";
	cin.get(ch);

	while (ch != '1'&& ch != '2')
	{
		cin.clear();
		cin.ignore(100, '\n');
		cout << "Please select a valid show number: ";
		cin.get(ch);
	}

	switch (ch)
	{
	case '1':showTime = "1pm show";
		break;
	case '2':showTime = "7pm show";
		break;


	}
```

```
        this->showTime = showTime;


        cin.get(terminator);//clears buffer
        return showTime;

}
```

showSeat.h

```
#include <iostream>
#include <iomanip>
#include <string>
#include<conio.h>

using namespace std;

class showSeat
{
public:
	showSeat();
	~showSeat();
	void initialiseFloorPlan();
	int getNumSeats();
	double getSeatSelection();


protected:
	char floorPlan[7][6];
	int numSeats;
	int rNum;
	double price;

private:
	void displayFloorPlan(int r, int c);
	void calculatePrice(double price);
};
////////////////////////////////////////////////////////////////////////////////
//////////////////
//constructor
showSeat::showSeat()
{
	for (int  r = 0; r < 7; r++)
	{
		for (int c = 0; c < 6; c++)
		{
			floorPlan[r][c] = 0;
		}
	}

    numSeats =0 ;
	rNum = 0;
	price=0 ;
}
////////////////////////////////////////////////////////////////////////////////
//////////////////
//destructor
showSeat::~showSeat()
{

}
//////////////////////////////////////////////////////
//Starts the Floor plan
void showSeat::initialiseFloorPlan()
{
	for (int r = 0; r < 7; r++)
	{
		for (int c = 0; c < 6; c++)
		{
			floorPlan[r][c] = 'A'; //character
```

```
                }

        }

}
/////////////////////////////////////////////////////////////////
//Asks the user to choose number of seats
int showSeat::getNumSeats()
{
        cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT
SEATS ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n" << endl;
        do
        {
                cout << "Please enter the number of seats desired (max:8): "<<endl;
                cin >> numSeats;

        } while (numSeats<1 || numSeats>8);

        return numSeats;
}
//////////////////////////////////////////////////////////////////////////////////
/////////////////
//Get the user to select the seats they want
double showSeat::getSeatSelection()
{
        char ch;
        int r = -1;
        int c = -1;
        displayFloorPlan(r,c);
        do
        {
                for (int i = 0; i < numSeats; i++)
                {
                        char terminator;
                        do
                        {
                                cout << "Please enter a row number between 1 and 7:
";

                                cin >> r;
                                r = (r - 1);
                        } while (r<0 || r>7);
                        rNum = r;
                        do
                        {
                                cout << "Please enter a column number between 1 and
6: ";

                                cin >> c;
                                c = (c - 1);
                        } while (c<0 || c>6);




                        while (floorPlan[r][c] == 'H')
                        {
                                cout << "Seat is unavailable.\n ";
                                do
                                {
                                        cout << "Please re-enter a row number between
1 and 7: ";

                                        cin >> r;
```

```cpp
                } while (r < 1 || r > 7);
                do
                {
                    cout << "Please re-enter a column number between 1 and 6: ";

                    cin >> c;

                } while (c < 1 || c > 6);

            }

            floorPlan[r][c] = 'H';
            calculatePrice(price);



        }

        system("CLS");
        displayFloorPlan(r, c);


        do
        {
            cout << "Please check your seat.\n";
            cout << "Are you happy with your selection? (Y/N) ";
            cin >> ch;

        } while (ch != 'Y' && ch != 'y' && ch != 'N' && ch != 'n');

        if ( ch!='Y' && ch!='y')
        {
            initialiseFloorPlan();
            cout << "Please reselect your seats. ";
            price = 0;
        }

    } while (ch=='n' || ch=='N');


    return price;


}


//////////////////////////////////////////////////////////////////////////////////////////////////////////
//Displays the Floor Plan

void showSeat::displayFloorPlan(int r, int c)
{
    char terminator;

    cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT SEATS INTERACTIVELY ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n" << endl;



    cout<< "                                               |  1  |  2  |  3  |  4  |  5  |  6  |";
```

```
        for (int r = 0; r < 7; r++)
        {

                cout <<"\n\n"<<setw(42)<< (r+1)<<"|";
                for (int c = 0; c < 6; c++)
                {

                        cout <<"    "<< floorPlan[r][c]<<" ";
                }
        }
        cout << "\n                                                        ~~~~~~~~~~~ Key
~~~~~~~~~~~\n ";
        cout << "                                                           A= Availeble
seat\n";
        cout << "                                                             H= Held
seat\n";



}
////////////////////////////////////////////////////////////////////////////////
//////////////////
//Calculates the price of the tickets
void  showSeat::calculatePrice(double price)
{

        if (rNum >= 0 && rNum <= 4)
        {
                price = price + 20;
        }

        if (rNum >= 5 && rNum <= 7)
        {
                price = price + 30;

        }
        this->price = price;
}
```

ticket.h

```
#include <iostream>
#include <iomanip>
#include <string>


using namespace std;

class ticket
{
public:
       ticket();
       ~ticket();
       void getTicketinfo(string &FName, string &LName, string &address, string &
showName, string &showDate, string &showTime,double price,int numSeats, double
discount);
       double getDiscount(double price,double discount);
       char ch;
private:
       double discount = 0;
       char code;
};
//////////////////////////////////////////////////////////////////////////////////
///////////////////
//constructor
ticket::ticket()
{


}
//////////////////////////////////////////////////////////////////////////////////
///////////////////
//destructor
ticket::~ticket()
{
}
//////////////////////////////////////////////////////////////////////////////////
/////////////////////////////
//Gets and prints the information of the ticket
void ticket::getTicketinfo(string &FName, string &LName, string &address, string
& showName, string &showDate, string &showTime, double price, int numSeats,double
discount)
{
       char terminator;
       cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ YOUR
TICKETS ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n\n" << endl;
       cout <<
"=================================================================================
========================================\n" << endl;
       cout << "========================================== The Bucks Center for
the Performing Arts ====================================\n" << endl;
       cout << "===================================================== Enjoy the
show ===================================================\n\n" << endl;

       cout << "Show: "<< showName<<endl;
       cout << "Date: " << showDate<<endl;
       cout << "Time: " << showTime<<endl;
       cout <<
"=================================================================================
========================================\n" << endl;
```

```
        cout << "Number of Seats: " <<numSeats<<endl;
        cout << "Total cost:  "<<(char)156 <<(discount)<< endl;
        cout << "Ticket Purchaser: " << FName << " " << LName<<endl;
        cout << "Postal address: " << address<<endl;
        cout <<
"================================================================================
========================================\n\n" << endl;
        cout <<
"================================================================================
========================================\n" << endl;
}
/////////////////////////////////////////////////////////////////////////////////
/////////////////////////////
//Asks the User to use any discount in case of having one
double ticket::getDiscount(double price,double discount)
{

        char terminator;



        do
        {
                cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
DISCOUNT ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n" << endl;

                cout << "1. 10% Discount" << endl;
                cout << "2. 30% Discount" << endl;
                cout << "3. 50% Discount\n" << endl;
                cout << "4. Do not have any discount code.\n" << endl;
                cout << "Please choose a discount: ";
                cin.get(ch);

                while (ch != '1' && ch != '2' && ch != '3' && ch!='4')
                {
                        cin.clear();
                        cin.ignore(100, '\n');// ensures buffer is completrly clear
(if, say, the user has previosly input a long string)
                        cout << "Please select a valid discount choice number: ";
                        cin.get(ch);
                }
                if (ch == '1')
                {

                        discount = price* 0.9;
                        cout << "Please enter your discount coupon code: " << endl;
                        cin >> code;

                        cout << "Processing information..." << endl;
                        Sleep(3000); //Sleeps for 3000 ms or 3 sec
                        cout << "Code accepted!" << endl;

                }

                if (ch == '2')
                {
                        discount = price* 0.7;
                        cout << "Please enter your discount coupon code: " << endl;
                        cin >> code;

                        cout << "Processing information..." << endl;
                        Sleep(3000); //Sleeps for 3000 ms or 3 sec
```

```
                cout << "Code accepted!" << endl;
            }
            if (ch == '3')
            {
                discount = price* 0.5;
                cout << "Please enter your discount coupon code: " << endl;
                cin >> code;

                cout << "Processing information..." << endl;
                Sleep(3000); //Sleeps for 3000 ms or 3 sec
                cout << "Code accepted!" << endl;

            }
            if (ch=='4')
            {
                discount = price;
                cout << "Thank you!" << endl;
            }



        } while (ch == '1'&& ch == '2' && ch == '3');

        return discount;
        return price;
}
```

Testing

In order to ensure that the program meets each of the functional and non-functional requirements, it must be rigorously tested. This will also highlight any errors that occur when the program runs.

Functional Requirements Testing

Login & Profile Information

Test No.	*Input(s)*	*Output*		*Successful*
		Expected Output	*Actual Output*	
1	*Enter 'thunder' as username*	*Input is accepted user is prompt to enter password*	*Input is accepted user is prompt to enter password*	✓
2	*Enter 'marks' as password*	*Input is accepted user is prompt to enter Personal Info*	*Input is accepted user is prompt to enter password*	✓
3	*Enter 'bigfriendorange' as username*	*Input is not accepted its bigger than 10 char prompt user to re-enter username*	*Input is not accepted its bigger than 10 char prompt user to re-enter username*	✓
4	*Enter 'Felix' as first name*	*Input accepted and stored into FName variable*	*Input accepted and stored into FName variable*	✓
5	*Enter 'Saraiva' as last name*	*Input accepted and stored into LName variable*	*Input accepted and stored into LName variable*	✓
6	*Enter 'High street' as address*	*Input accepted and stored into address variable*	*Input accepted and stored into address variable*	✓

Selecting Seats

Test No.	*Input(s)*	*Output*		*Successful?*
		Expected Output	*Actual Output*	
1	*Enter '9' as seat amount*	*Input is not accepted, prompt to re-enter*	*Input is not accepted, prompt to re-enter*	✓
2	*Enter '8' as seat amount*	*Input is accepted*	*Input is accepted*	✓
3	*Enter '1' as seat amount*	*Input is accepted*	*Input is accepted*	✓
4	*Enter '8' as row number*	*Input is not accepted, prompt to re-enter*	*Input is not accepted, prompt to re-enter*	✓
5	*Enter '7' as row number*	*Input is not accepted, prompt to re-enter*	*Input is not accepted, prompt to re-enter*	✓
6	*Enter '7' as row number and enter 6 as a column number*	*Input is accepted character 'H' is placed in seat row 7, column 6*	*Input is accepted character 'H' is placed in seat row 7, column 6*	✓
7	*Enter '5' as row number and enter '5' as a column number*	*Input is accepted character 'H' is placed in seat row 5, column 5*	*Input is accepted character 'H' is placed in seat row 5, column 5*	✓
8	*Enter 'y' when asked if choice is acceptable*	*Total price is displayed, user is prompt to enter discount if they have one*	*Total price is displayed, user is prompt to enter discount if they have one*	✓
9	*Enter 'N' when asked if choice is acceptable*	*Array is reset, all 'H' characters are removed, seat selection function is called again*	*Array is reset, all 'H' characters are removed, seat selection function is called again*	✓

Calculating price & Discount

Test No.	*Input(s)*	*Output*		*Successful?*
		Expected Output	*Actual Output*	
1	*Enter '2' as seat amount, choose seats (1,1) and (1,7)*	*Price expected 50$*	*Price expected 50$*	✓
2	*Enter '3' as seat amount, choose seats (1,1), (1,2) and (1,3)*	*Price expected 60$*	*Price expected 60$*	✓
3	*Enter '2' as seat amount, choose seats (1,1) and (1,7)) and use 10% discount*	*Price expected 45$*	*Price expected 45$*	✓
4	*Enter '2' as seat amount, choose seats (1,1) and (1,7)) and use 50% discount*	*Price expected 25$*	*Price expected 25$*	✓

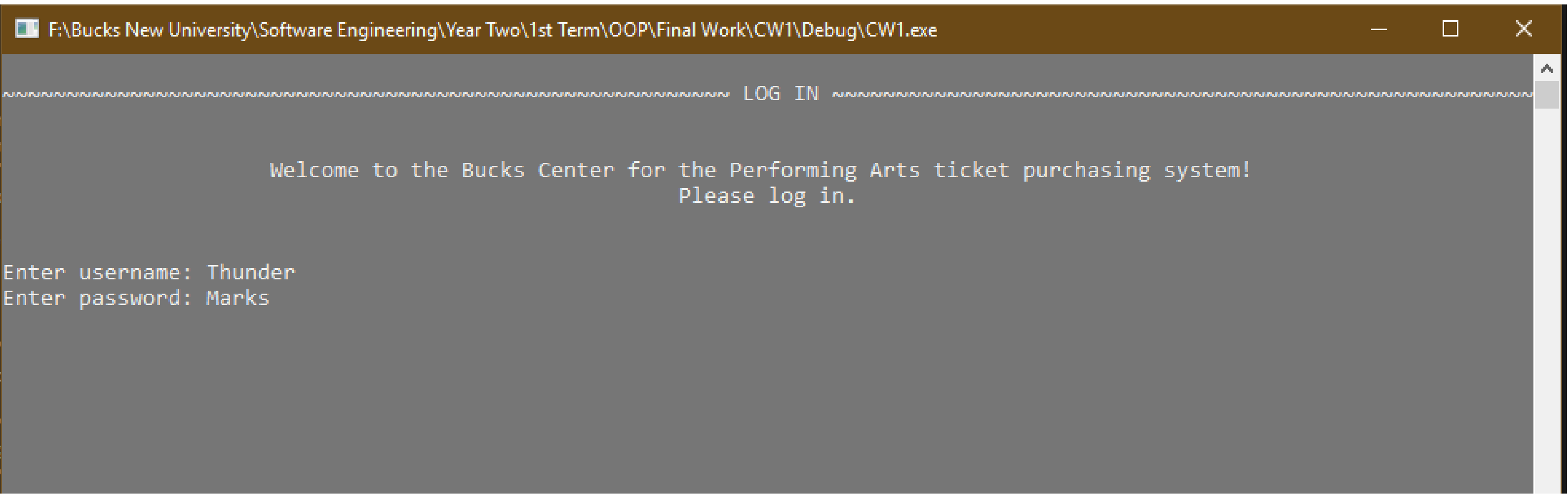
Non-Functional Requirements Testing

Screen	*The Interface is not unappealing*	*The Interface is easy to use*	*No errors occur while this section of the program runs*	*No Invalid data types are accepted*	*All process response times are short*	*It is easy to modify the code*
Log In	✓	✓	✓	✓	✓	✓
Enter Profile Information	✓	✓	✓	✓	✓	✓
Select an Upcoming Show	✓	✓	✓	✓	✓	✓
Main Menu	✓	✓	✓	✓	✓	✓
View Shows	✓	✓	✓	✓	✓	✓
Select seats Interactively	✓	✓	✓	✓	✓	✓
Enter Payment Information	✓	✓	✓	✓	✓	✓
Enter Discount	✓	✓	✓	✓	✓	✓
Your Ticket	✓	✓	✓	✓	✓	✓

Final Functional Program

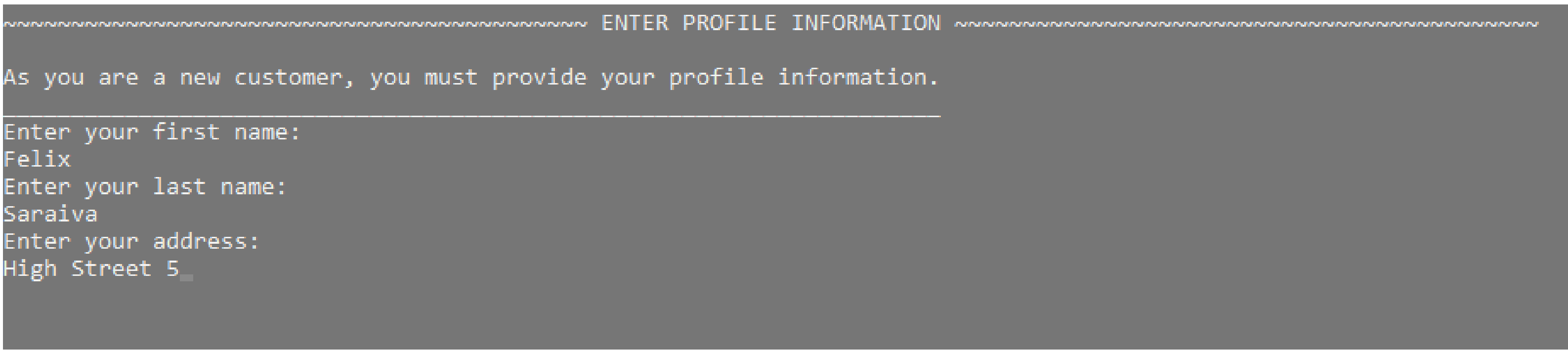
Screenshots

Log in



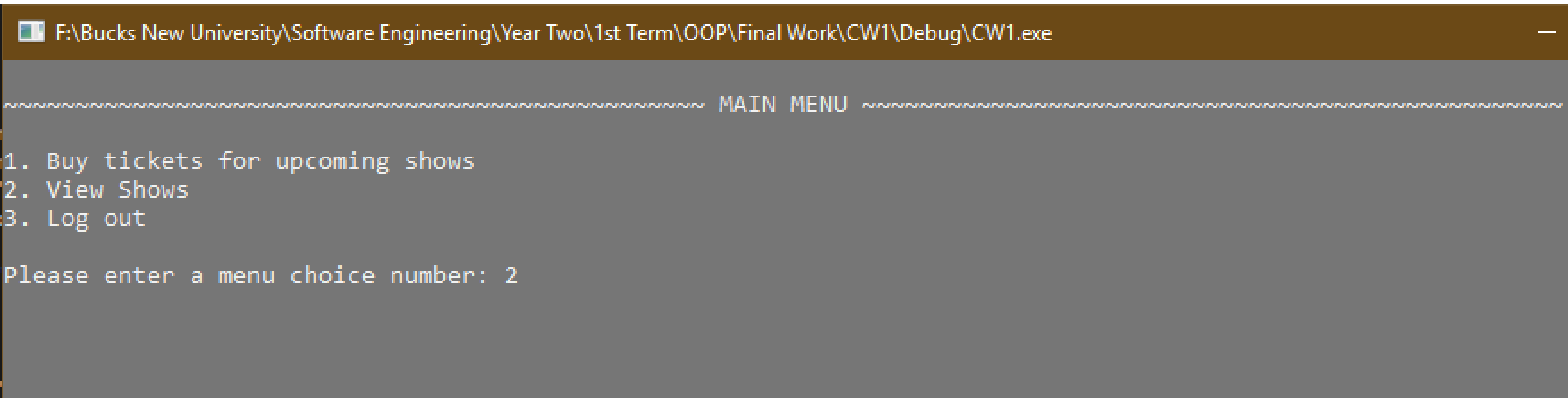
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ LOG IN ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


                     Welcome to the Bucks Center for the Performing Arts ticket purchasing system!
                                                         Please log in.


Enter username: Thunder
Enter password: Marks
```

Enter Profile Information



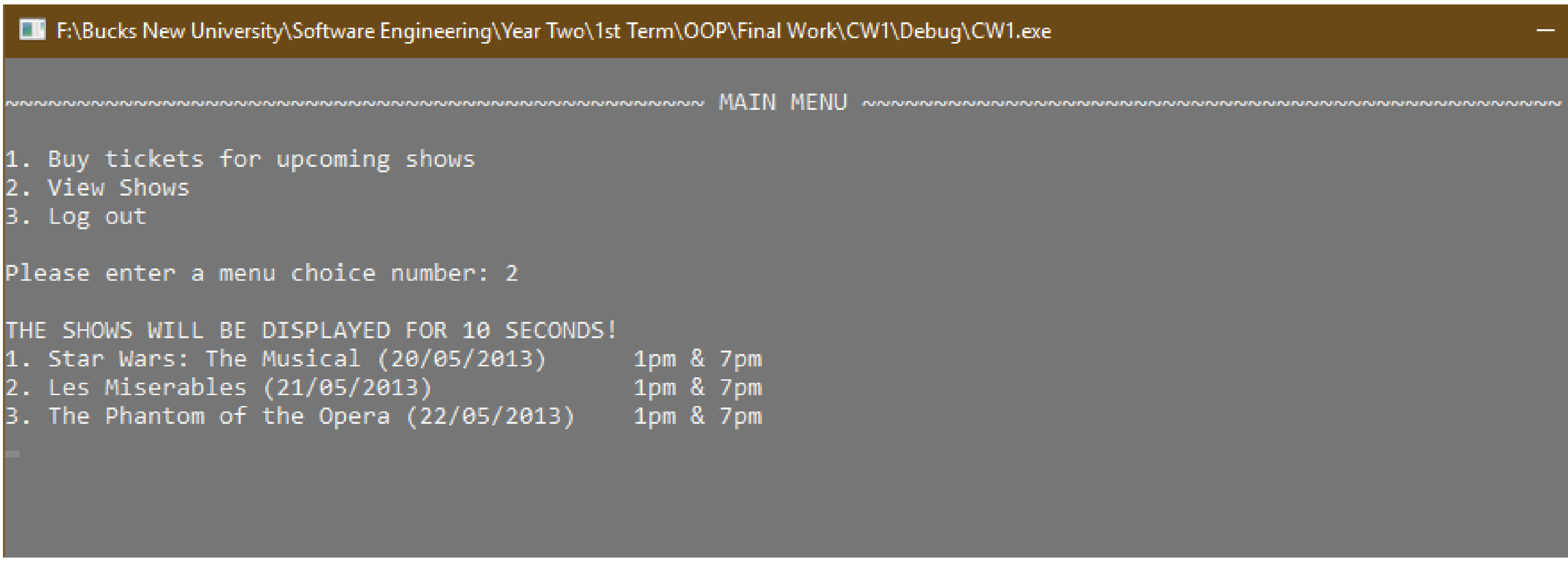
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ENTER PROFILE INFORMATION ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

As you are a new customer, you must provide your profile information.
_____________________________________________________________________
Enter your first name:
Felix
Enter your last name:
Saraiva
Enter your address:
High Street 5_
```

Main Menu



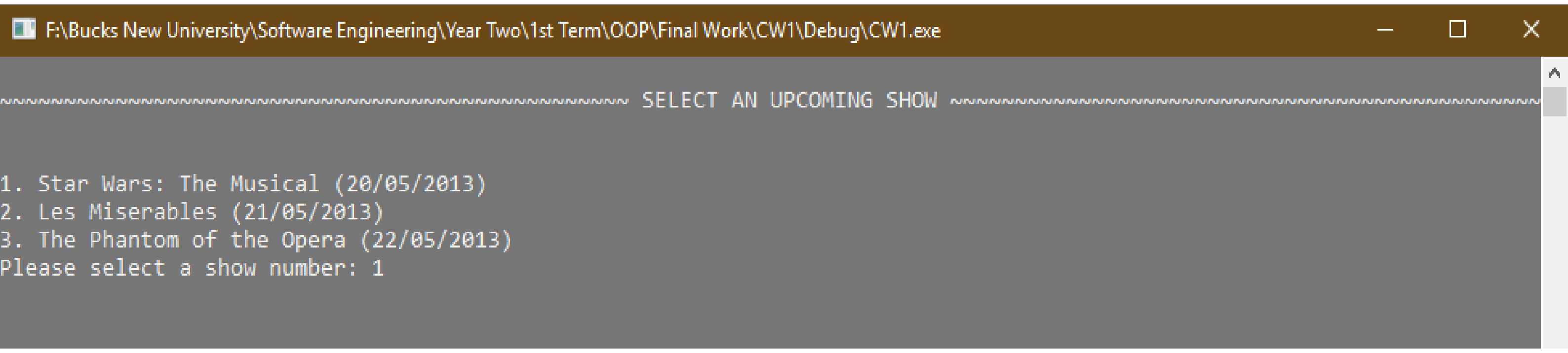
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ MAIN MENU ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1. Buy tickets for upcoming shows
2. View Shows
3. Log out

Please enter a menu choice number: 2
```

View Shows



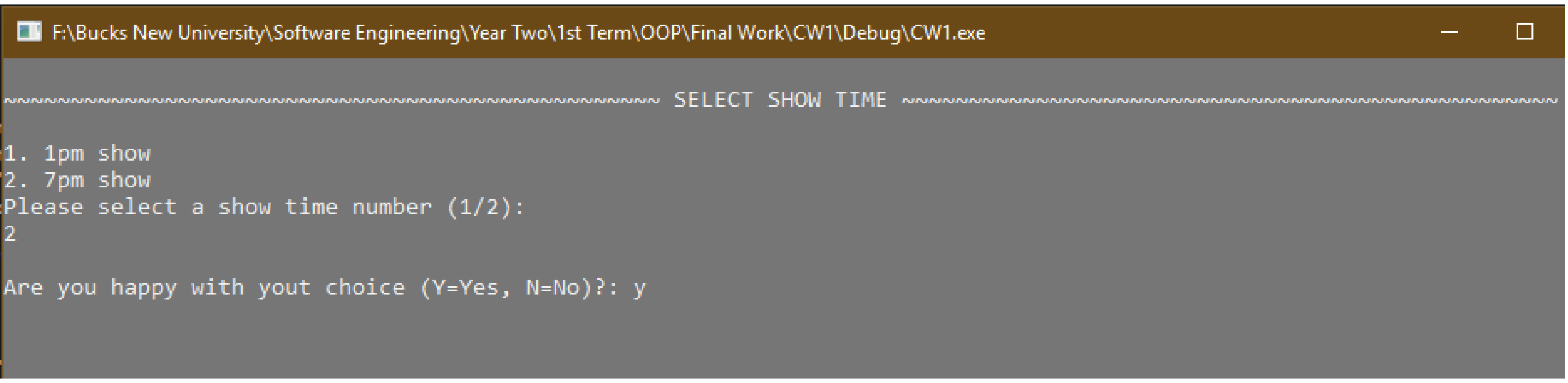
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ MAIN MENU ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1. Buy tickets for upcoming shows
2. View Shows
3. Log out

Please enter a menu choice number: 2

THE SHOWS WILL BE DISPLAYED FOR 10 SECONDS!
1. Star Wars: The Musical (20/05/2013)      1pm & 7pm
2. Les Miserables (21/05/2013)              1pm & 7pm
3. The Phantom of the Opera (22/05/2013)    1pm & 7pm
```

Select an Upcoming Show



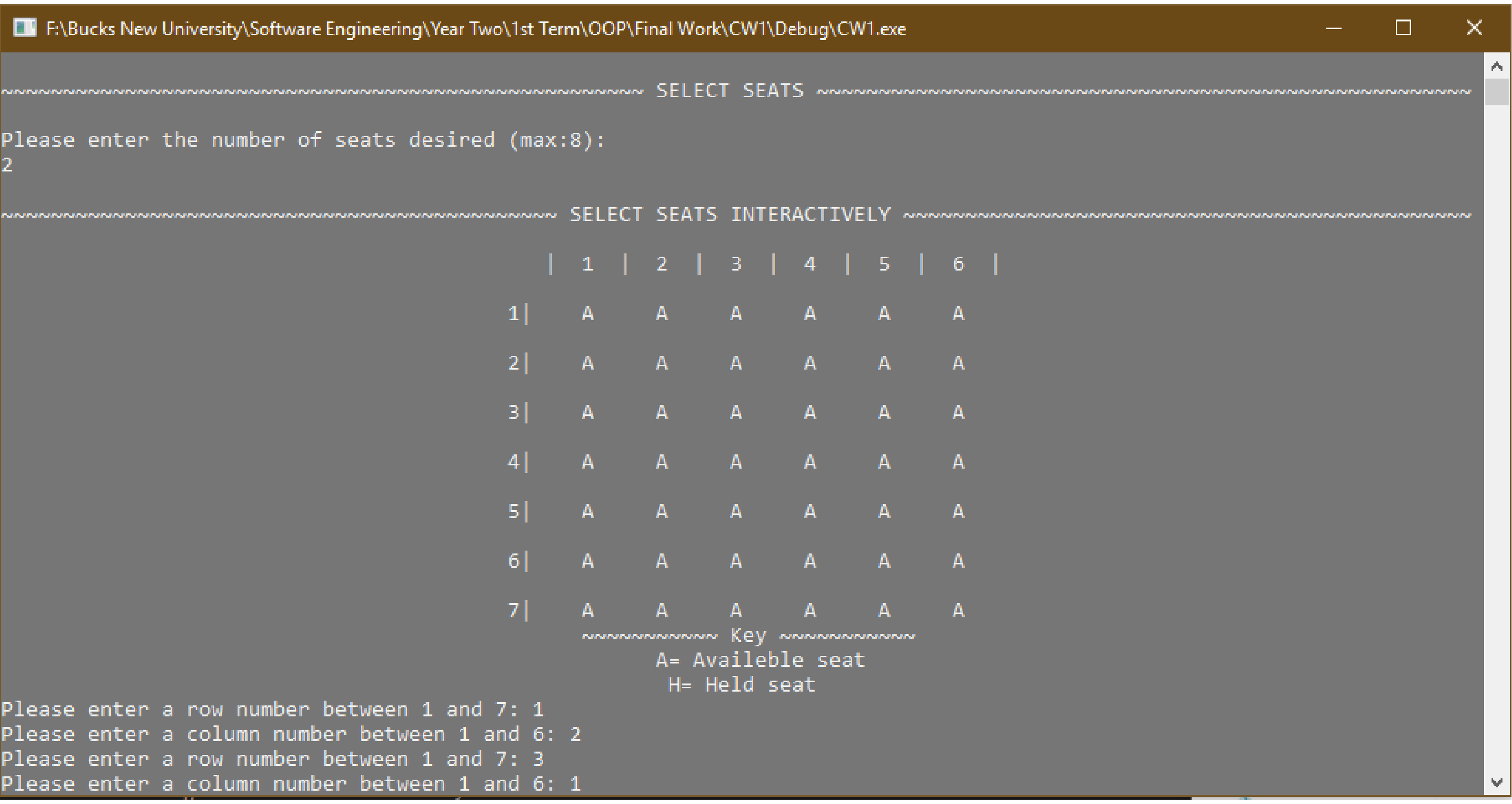
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT AN UPCOMING SHOW ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


1. Star Wars: The Musical (20/05/2013)
2. Les Miserables (21/05/2013)
3. The Phantom of the Opera (22/05/2013)
Please select a show number: 1
```

Select Show Time

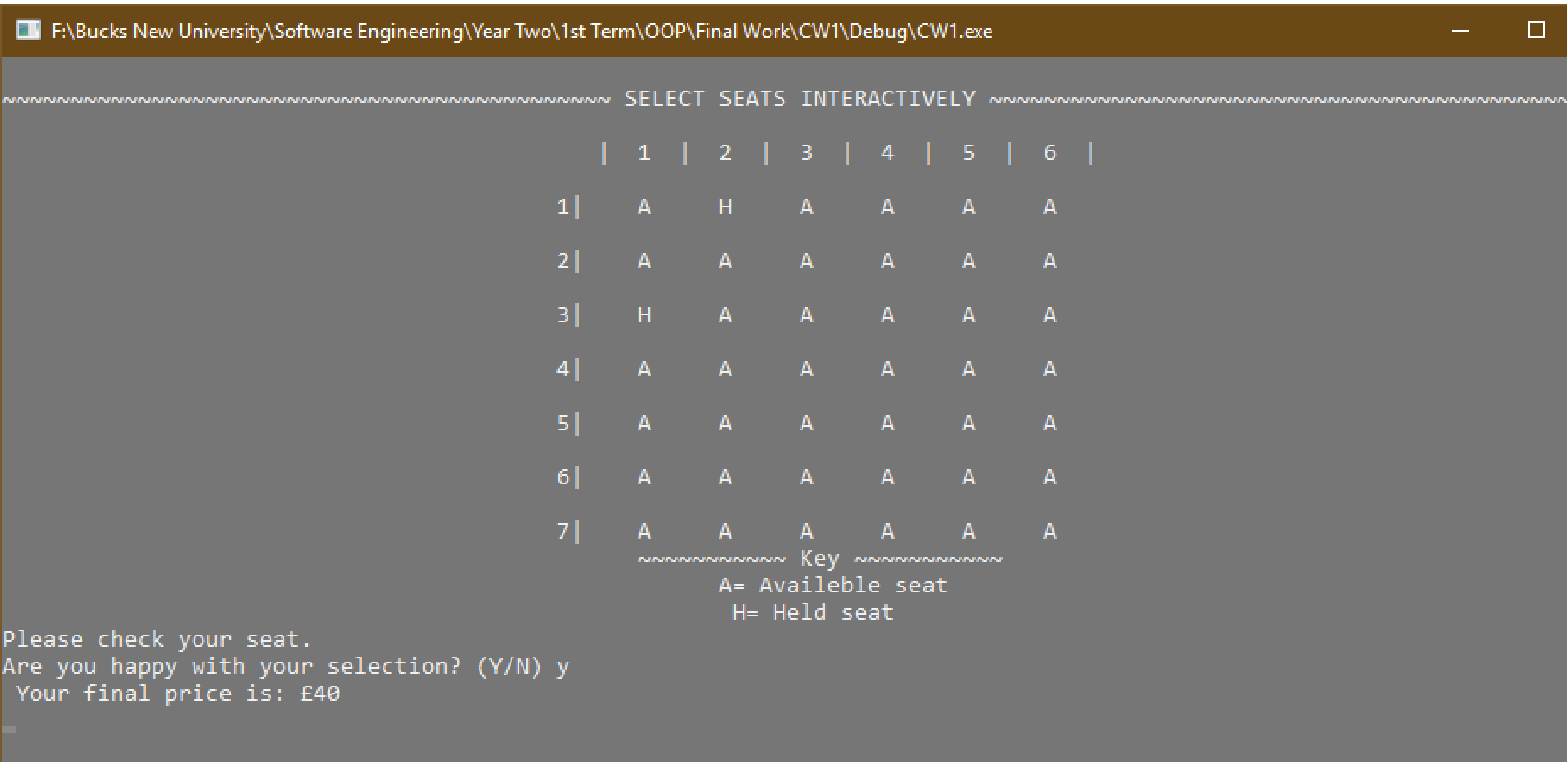


```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT SHOW TIME ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1. 1pm show
2. 7pm show
Please select a show time number (1/2):
2

Are you happy with yout choice (Y=Yes, N=No)?: y
```

Select Seats

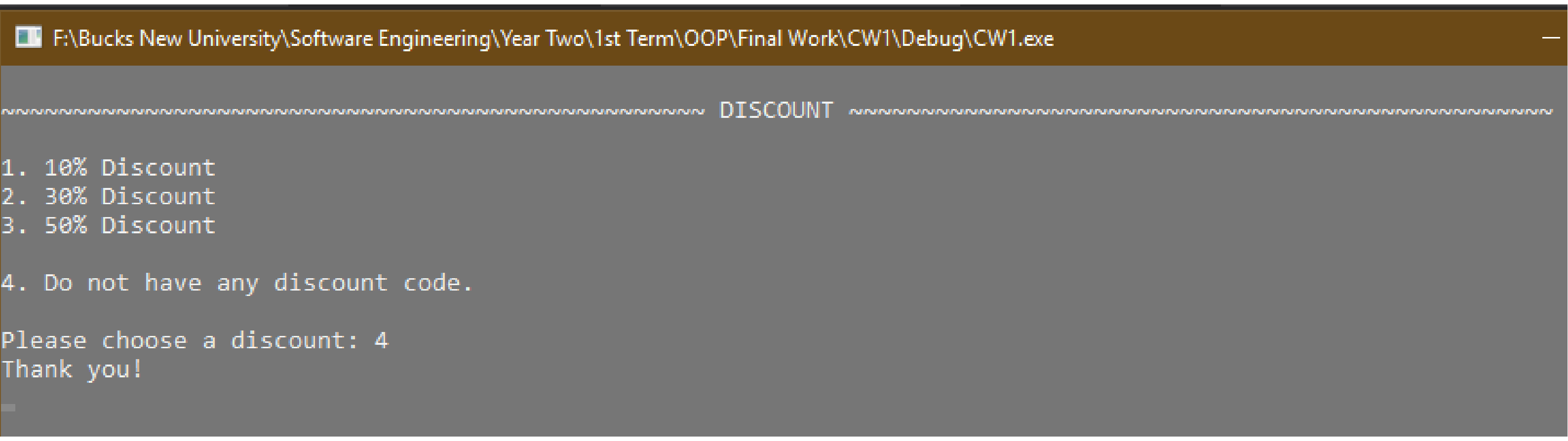


```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT SEATS ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Please enter the number of seats desired (max:8):
2

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT SEATS INTERACTIVELY ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

                                            |  1  |  2  |  3  |  4  |  5  |  6  |

                                         1|    A     A     A     A     A     A

                                         2|    A     A     A     A     A     A

                                         3|    A     A     A     A     A     A

                                         4|    A     A     A     A     A     A

                                         5|    A     A     A     A     A     A

                                         6|    A     A     A     A     A     A

                                         7|    A     A     A     A     A     A
                                               ~~~~~~~~~~~ Key ~~~~~~~~~~~
                                                     A= Availeble seat
                                                      H= Held seat
Please enter a row number between 1 and 7: 1
Please enter a column number between 1 and 6: 2
Please enter a row number between 1 and 7: 3
Please enter a column number between 1 and 6: 1
```



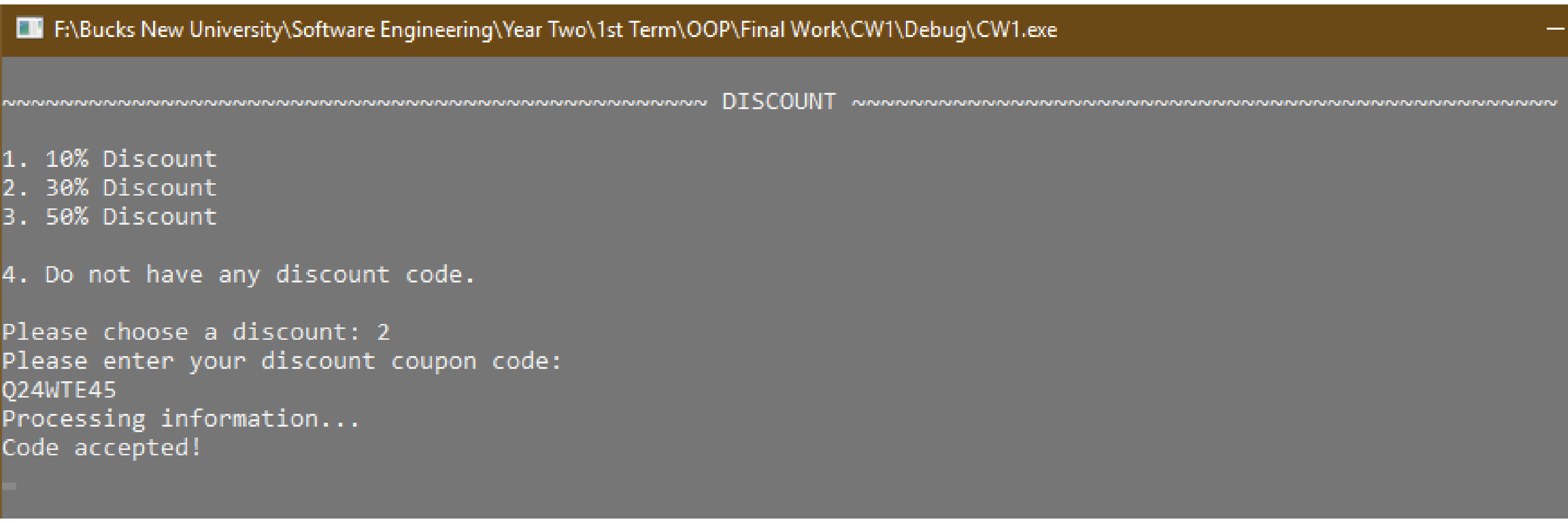
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ SELECT SEATS INTERACTIVELY ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

                                            |  1  |  2  |  3  |  4  |  5  |  6  |

                                         1|    A     H     A     A     A     A

                                         2|    A     A     A     A     A     A

                                         3|    H     A     A     A     A     A

                                         4|    A     A     A     A     A     A

                                         5|    A     A     A     A     A     A

                                         6|    A     A     A     A     A     A

                                         7|    A     A     A     A     A     A
                                               ~~~~~~~~~~~ Key ~~~~~~~~~~~
                                                     A= Availeble seat
                                                      H= Held seat
Please check your seat.
Are you happy with your selection? (Y/N) y
 Your final price is: £40
```

No Discount



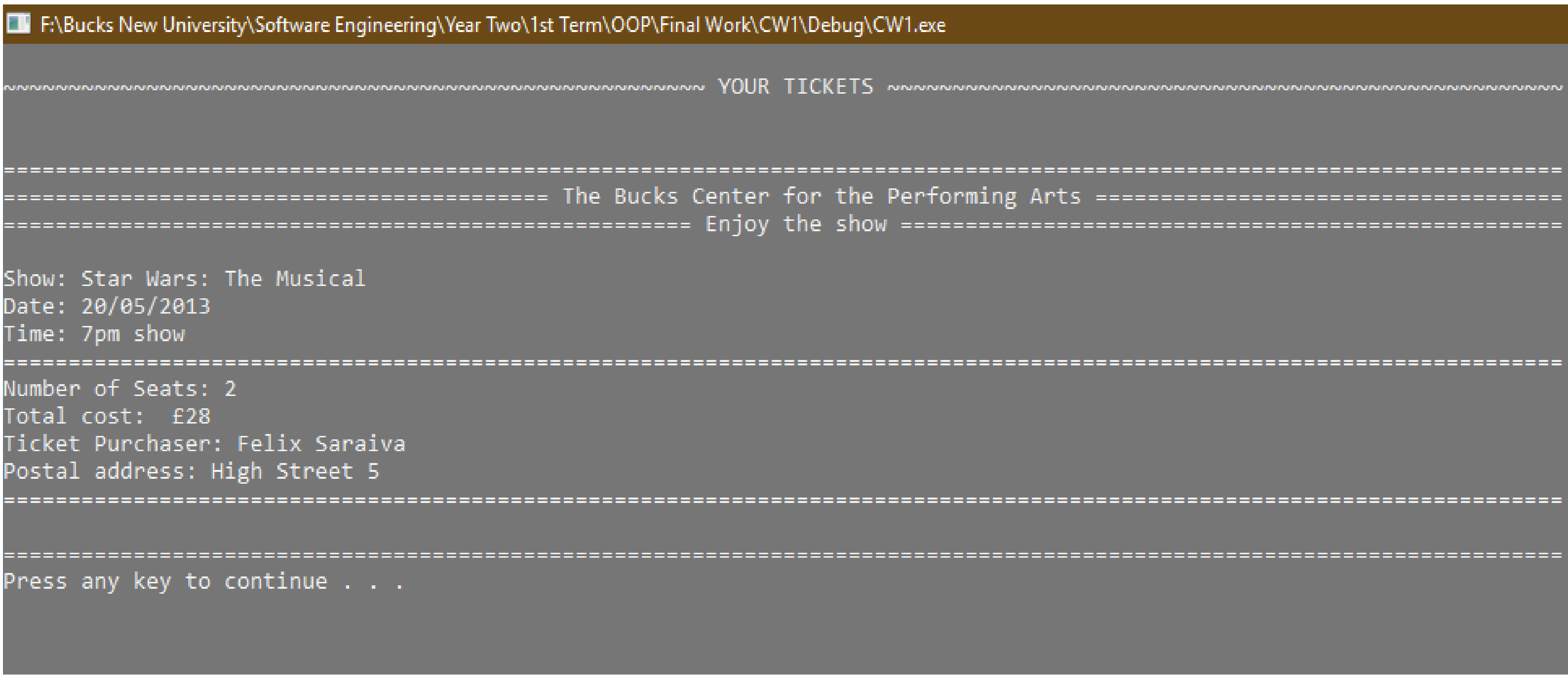
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ DISCOUNT ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1. 10% Discount
2. 30% Discount
3. 50% Discount

4. Do not have any discount code.

Please choose a discount: 4
Thank you!
```

With Discount



```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ DISCOUNT ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1. 10% Discount
2. 30% Discount
3. 50% Discount

4. Do not have any discount code.

Please choose a discount: 2
Please enter your discount coupon code:
Q24WTE45
Processing information...
Code accepted!
```

Your Ticket



```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ YOUR TICKETS ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


========================================================================================================================
========================================== The Bucks Center for the Performing Arts ====================================
===================================================== Enjoy the show ===================================================

Show: Star Wars: The Musical
Date: 20/05/2013
Time: 7pm show
========================================================================================================================
Number of Seats: 2
Total cost:  £28
Ticket Purchaser: Felix Saraiva
Postal address: High Street 5
========================================================================================================================

========================================================================================================================
Press any key to continue . . .
```

Conclusion

After analysing each part of the case study, I have been given I started to plan the best way to proceed into developing the course work.

Starting with a simple Requirement List of the Functions and Non-Functions that my system should have, followed by a class diagram, an activity diagram and a sequence diagram, so I could see how my system would interact and the best way to proceed in the different features.

In order to start planning the pseudocode, I found useful to do an action and response diagram, a simple table that allows me to see how the system will behave depending on the actions of the user. With all this material I was ready to start the pseudocode. Found some troubles but with research, and by sharing ideas with colleges always found an answer.

After the pseudocode done, it was time to start coding the system based on the same. It was a straight forward system and with the help of the pseudocode it was easier to not lose my self. Although I came across some problems.

My first issue was to start working with strings since it was a new method for me. In lectures we always used char so I had to ask for help and my tutor, Mike Everett, provide me with an extraction of a chapter about strings where I could find all the answers for my problems. After that I end up stuck with some problems but nothing that some extra reading and extra revising did not solve.

The most challenging part of this work for me was the implementation of the discount function, moving the price from one class to another and to apply a different set of discounts to it was really interesting, I end up stuck with so many challenging problems that made me research and discover more and more about C++ and OOP.

And the way to achieve that understanding was by, using all the information provided during the semester and online searching, designing Diagrams, planning every step of the way and use my previous OOAD group work to take ideas and find answers.

Developing this project helped me to improve my coding and planning ability.

After finishing the project, I acknowledge a better understanding of how systems work, how they are designed and what difficulties are encountered when designing them. I also learn what I can do improve my System and its design, and what mistakes should and must be avoided in order to have a fully functional System.

References

Everett, M. (2018-2019). *Assignment Brief.* High Wycombe: Bucks New University.

Everett, M. (2019). *Appendix A: Source Code.* High Wycombe: Bucks New University.

Everett, M. (n.d.). *Object-Oriented Principles and Design CW2: The Bucks centre for the Performing Arts.* High Wycombe: Bucks New University.

Felix Saraiva, Vitor Cardoso, Daniel Morgan (2018). *The BUCKS Centre for the Performing Arts OOAD.* High Wycombe: Bucks New University.

wikipedia. (2019, January 20). *Object-oriented_programming*. Retrieved from wikipedia: https://en.wikipedia.org/wiki/Object-oriented_programming