APRIL 19, 2018

BOWEN THEATRE

PRESENTED BY: STEVE TEECE

TABLE OF CONTENTS

BOWEN THEATRE

PREAMBLE

The Bowen Theatre is a small purpose-built theatre. It has four rows and six seats in each row.
They require a system to keep track of bookings. When customer books a seat they can request a specific seat using the row and seat number. Bowen Theatre Company only presents a single show or performance at a time. They do not run a show more than once, so the date and time of the show are not required when taking a booking.

BASIC PROGRAM REQUIREMENTS

The program for Bowen Theatre's Reservation/Booking system needs to perform the following basic functions:

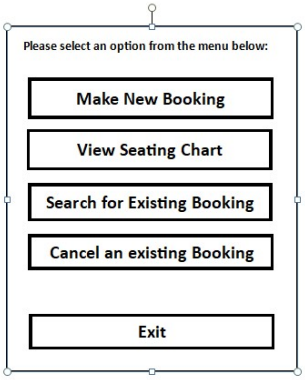
1. Accept and store bookings from people wanting seats at the theatre
2. Display seating chart so customers can select seat(s) they would like to book
3. Store ticket information, being person's name, the booking type (Adult or Concession), and the booked seat
4. Cater for a single person to book more than one seat
5. Calculate the total cost of the booking, considering Adult and Concession price differences
6. It is possible for a single booking to contain multiple tickets, with different ticket types (Adult or Concession) with in the one booking.
7. Be able to look up a seat, and view any bookings for that seat
8. Be able to Cancel an existing booking
9. Bookings are to be made for only a single event at the theatre at a time. Therefore, date of the show/booking does not need to be recorded.
10. Tickets are pre-printed, and a seat number hand written on at time of making the booking. It is therefore not required for the program to print any tickets or booking information.
11. The program should be easy to use and intuitive, with some help (contextual help or tool tips) provided for basic user functions
12. The program should be targeted at the basic computer user and should not require any advanced knowledge to operate.

Program Design

SCREEN DESIGNS

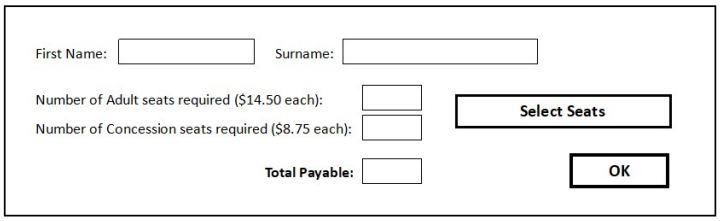
MAIN MENU SCREEN

Main Menu Screen wireframe



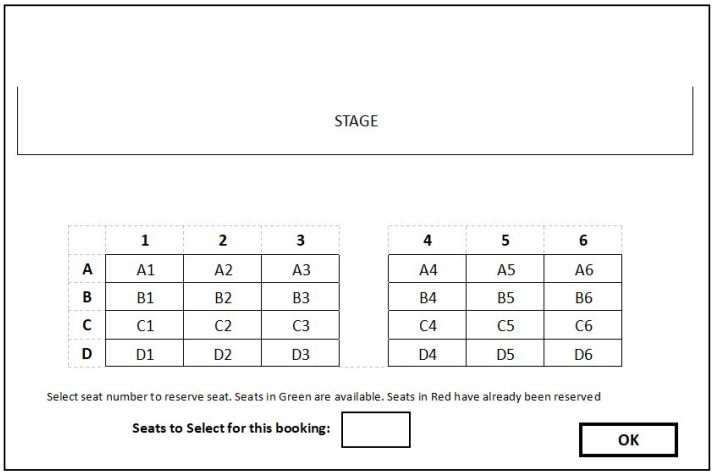
MAKE BOOKING SCREEN

Make Booking Screen wireframe



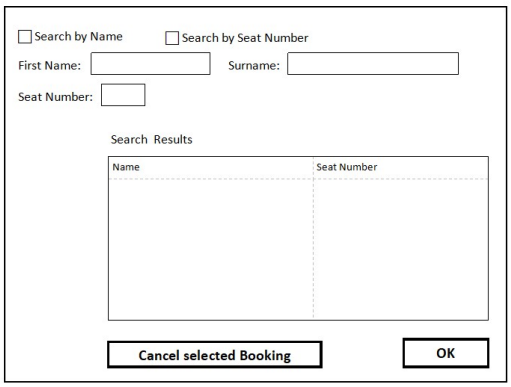
SEAT SELECTION SCREEN

Seat Selection Screen wireframe



CURRENT BOOKINGS SEARCH SCREEN

Search Screen wireframe



DATA STRUCTURES

ENUMERATIONS

Ticket Type Enumeration:

```
TicketType
        Adult = 1
        Concession = 2
End TicketType
```

TICKET DATA STRUCTURE

Structure of a single Ticket object

```
Ticket (built as a Structure)
        FirstName (string)
        LastName (string)
        SeatNumber (String)
        TicketType (Enumeration)
```

SEATS DATA STRUCTURE

Structure for Seating Chart (2-Dimensional Array)

```
        Seats(Row as Integer, Column as Integer)
```

BOOKING STRUCTURE

As more than one ticket can be reserved per booking, a separate temporary variable dimension array will be used. When the user selects the number of tickets required, the Booking array will be resized to accommodate the required tickets

```
Booking() type of Ticket
```

BOOKINGS FILE DATA STRUCTURE

Booking file will be stored as a comma separated text file (.csv file) and each line contains a single Ticket record in the format:

```
FirstName,LastName,SeatNumber,TicketType
```

PROGRAM LOGIC

MAIN MENU LOGIC

```
Start Program
Load data from file into Tickets array [Sub Routine]
Mark seats already reserved [Sub Routine]
        Display Menu Screen
        If action = new booking
                Create Booking [Sub Routine]
        If action = Search for Booking
                Search Booking [Sub Routine]
        If action = Cancel Booking
                Search Booking [Sub Routine]
                CancelBooking [Subroutine]
        If action = Exit
                Exit Application
End Program
```

MAKE BOOKING LOGIC

```
Create New Booking
Display Make Booking form
Get person's name
Get number of Adult tickets
Get number of Concession tickets
If Select Seats action
        Select Seats [Function] passing number of seats needed
Calculate total owing for ticket purchase
Display total owing
If Action is OK
        Write booking info to file [Sub Routine] passing ticket info
Close Window
```

SEARCH BOOKINGS LOGIC

```
If CancelBooking is true, show CancelBooking button
If CancelBooking is false, hide CancelBooking button
Display Search screen
Get Search Type (by Name or By Ticket)
If Search type is by Name
        Hire Ticket Number search fields
Display Name fields
        Get name from User
        Call Search [function] passing in name info
        Populate Search Results list with returned data
If Search type is by Ticket Number
        Hide Name fields
        Show Ticket Search fields
        Call search [function] passing in ticket information
        Populate search results list with returned data
If Action is CancelBooking
        Call CancelBooking subroutine, passing the ticket selected
If action is OK
        Close search window
```

PROGRAM FUNCTIONS LOGIC

LOADDATA SUBROUTINE

```
If reservations file exists
         While there is data left in the file to read
                Read next line from file
                Inflate CSV record to a Ticket [InflateTicket Function]
                Convert Seat Number to Row/Column [ConvertSeat Function]
                Mark seat(row,col) as reserved [MarkSeatsReserved Subroutine]
                Add read ticket to Reservations()
        End While
Else File not Found
        Create new Reservations file
        Mark all seats as Available
End If
```

MARKRESERVEDSEATS SUBROUTINE

```
Start Subroutine (input seat(row,col)
        Change seat availability to Reserved (change colour to red)
End Subroutine
```

CREATEBOOKING SUBROUTINE

```
Start Subroutine (inputs FirstName, LastName, TicketType, SeatNumber)
        For each ticket in Bookings()
                Mark seat as reserved [MarkReservedSeat Subroutine]
                Add Ticket data to Reservations()
        End For
End Subroutine
```

CANCELBOOKING SUBROUTINE

```
Start CancelBooking sub routine (input of ticket)
        Call ConvertSeat function, passing in Ticket Number
        Remove Ticket from Reservations array
        Call SaveData sub routine
End sub routine
```

SAVEDATA SUBROUTINE

```
Start Subroutine
        Open file in Overwrite mode
        For each Item in Reservations()
                Deflate Reservation to CSV Line {DeflateReservation Function]
                Write line to file
        End for
        Close file
End Subroutine
```

SEARCH FUNCTION

```
Start Search Function (accepts firstname and lastname, or Ticket Number)
        For each item in Reservations array
                Check ticket info against passed search parameters
                If Match found, display information in search results list
        End for
End Search
```

INFLATETICKET FUNCTION

```
Start InflateTicket (input of single line of data separated by commas)
        Create temp array to hold record information
        Split inputted string on ","
        Return contents of temp as Ticket to caller
End Function
```

DEFLATETICKET FUNCTION

```
Start DeflateTicket Function (input of single Ticket)
        Create temp string to hold data as CSV
        For each property of Ticket
                Add property text to temp
                If not last property
                        Add ",” to temp
                End If
        End For
        Return temp as string to caller
End Function
```

CONVERTSEAT FUNCTION

```
Start ConvertSeat Function (accepts SeatNumber or seat(row,col))
        If input type is SeatNumber
                Get first character of seatnumber
                Turn character into row number (A=0, B=1, C=2, D=4)
                Return SeatNumber(row,col) to caller
        End If
        If input type is seat(row,col)
                Create temp string variable to hold ticket name
                Turn row into letter (0=A, 1=B, 2=C, 3=D)
                Return temp to caller as string (temp=letter+col)
        End if
End Function
```

SELECTSEATS FUNCTION

```
Start SelectSeat Function (accepts number of seats required)
        While seats required not 0
                Get seat(row,col) user clicked on
                Change seat colour to Reserved
                ConvertSeat function (passing seat(row,col)
                Return Seat Number to caller as string
                Decrement seats required counter
        End While
End Function
```

TEST DATA

RESERVATIONS FILE

A Bookings.csv file is included in this project to act as test data. The file will be copied to the Output directory on compilation if no newer file already exists in the location.

Format of the test file:

```
FirstName,LastName,SeatNumber,TicketType
```

Data included in the test file:

```
Cersie,Lannister,A2,3
Denerys,Targarian,C1,3
Denerys,Targarian,C2,3
Denerys,Targarian,C3,3
Homer,Simpson,C4,3
Homer,Simpson,C5,3
Homer,Simpson,C6,3
Ned,Flanders,D2,3
Homer,Simpson,D4,3
Ned,Stark,D5,3
Ned,Stark,D6,3
```

REFERENCES

1. ListView.FindItemWithText Method (String)
   https://msdn.microsoft.com/en-us/library/y3h4x385(v=vs.110).aspx
   https://docs.microsoft.com/en-au/dotnet/api/system.windows.forms.listview.finditemwithtext?view=netframework-4.7.1

2. ListView.FullRowSelect Property
   https://msdn.microsoft.com/en-us/library/system.windows.forms.listview.fullrowselect(v=vs.110).aspx

3. ListView.SelectedItems Property
   https://msdn.microsoft.com/en-us/library/system.windows.forms.listview.selecteditems(v=vs.110).aspx

4. Add tooltip control dynamically
   https://stackoverflow.com/questions/15751055/add-tooltip-control-dynamically