Theatre Booking System

22TikhomirV699

2020

Contents

[Investigation 2](#_Toc96940498)

[Desk based research 2](#_Toc96940499)

[Stakeholders 2](#_Toc96940500)

[Limitations of current system 2](#_Toc96940501)

[Design 3](#_Toc96940502)

[Sub programs 3](#_Toc96940503)

[Task List 3](#_Toc96940504)

[Algorithms 3](#_Toc96940505)

[User Interface 3](#_Toc96940506)

[UML 3](#_Toc96940507)

[Data Dictionary 3](#_Toc96940508)

[Software Development 5](#_Toc96940509)

[Developmental Testing 5](#_Toc96940510)

[Testing 0](#_Toc96940511)

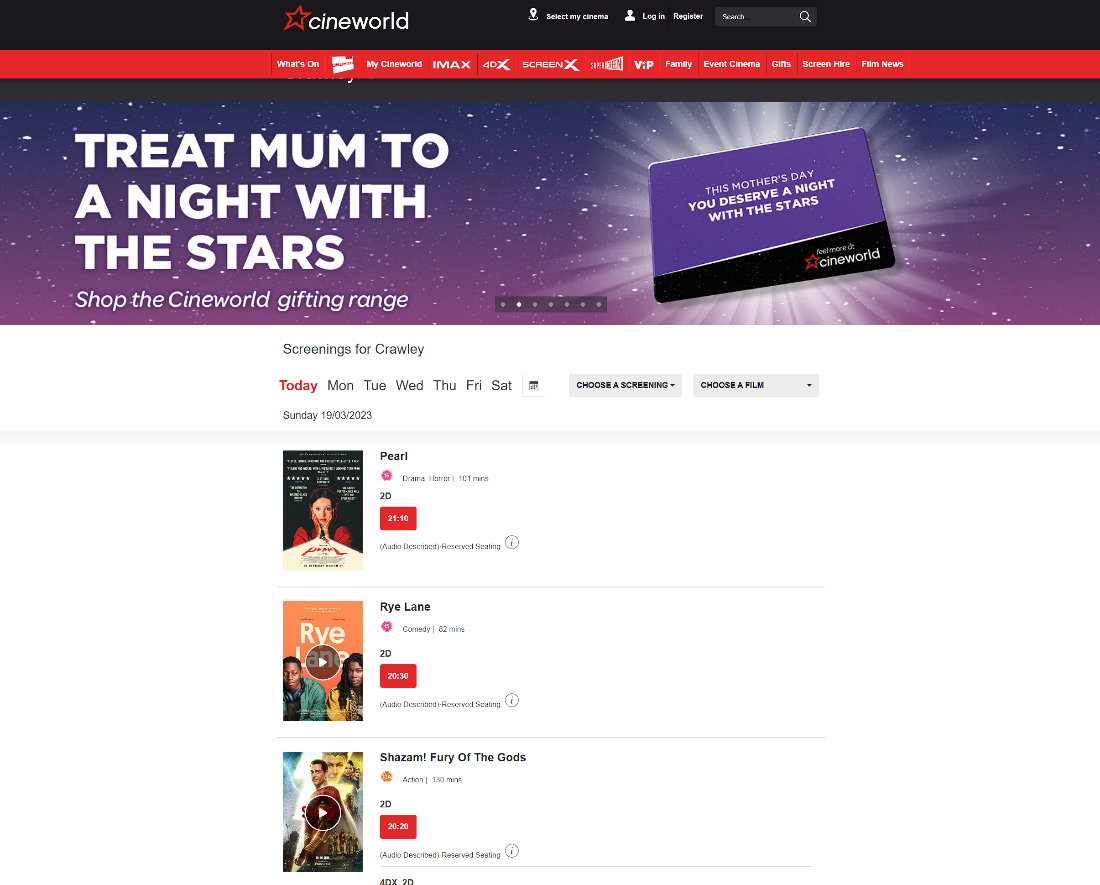
[Evaluation 0](#_Toc96940512)

## Investigation

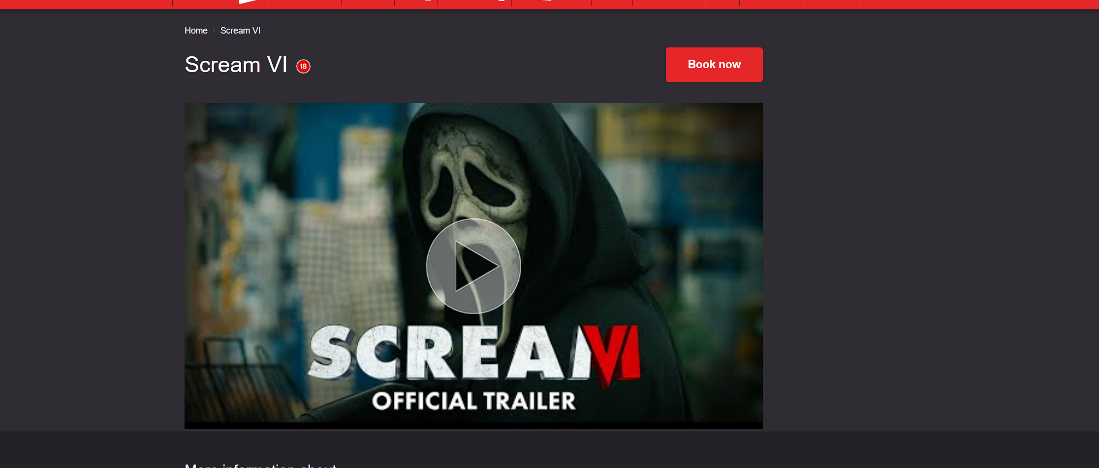
### Desk based research

Use this to look online for similar solutions, you should print screen your findings identifying any features you like and will be trying to incorporate into your solution as well as features you think don’t work.

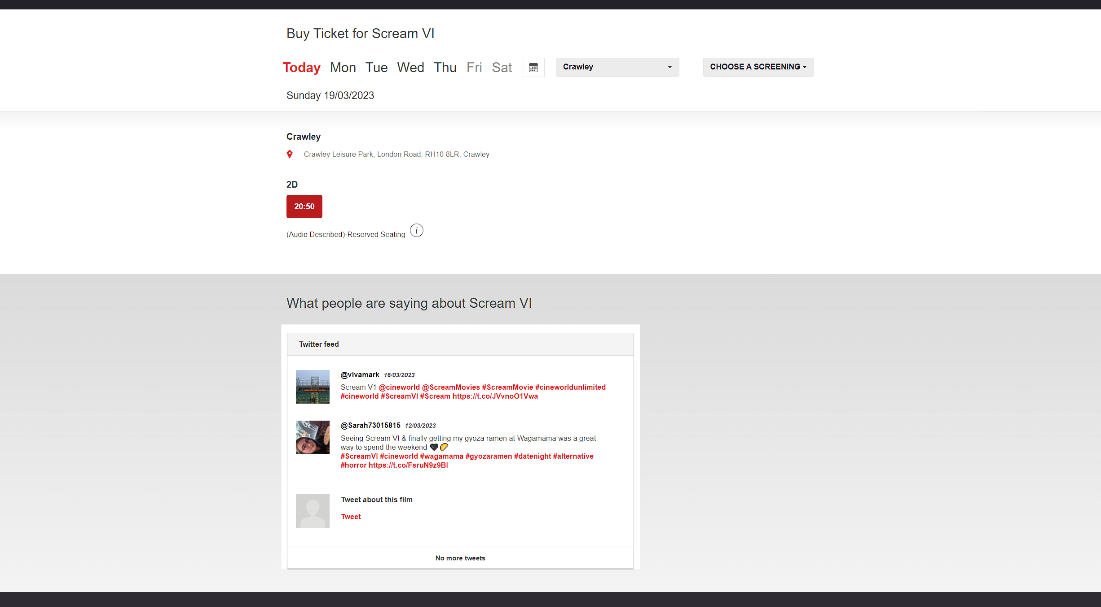
Cineworld is an online ticketing service for movie theatre patrons. The site has a clean and modern design, with its main page featuring an interactive map that allows users to quickly locate a nearby theatre. The map is overlaid with a search bar and a grid of movie posters that link to more information about the movies. Users can easily navigate to the “Buy Tickets” page, where they can select a theatre and showtime and purchase tickets. The layout is simple and intuitive, with helpful features such as a “Seat Selection” button that allows customers to pick the exact seats they want.



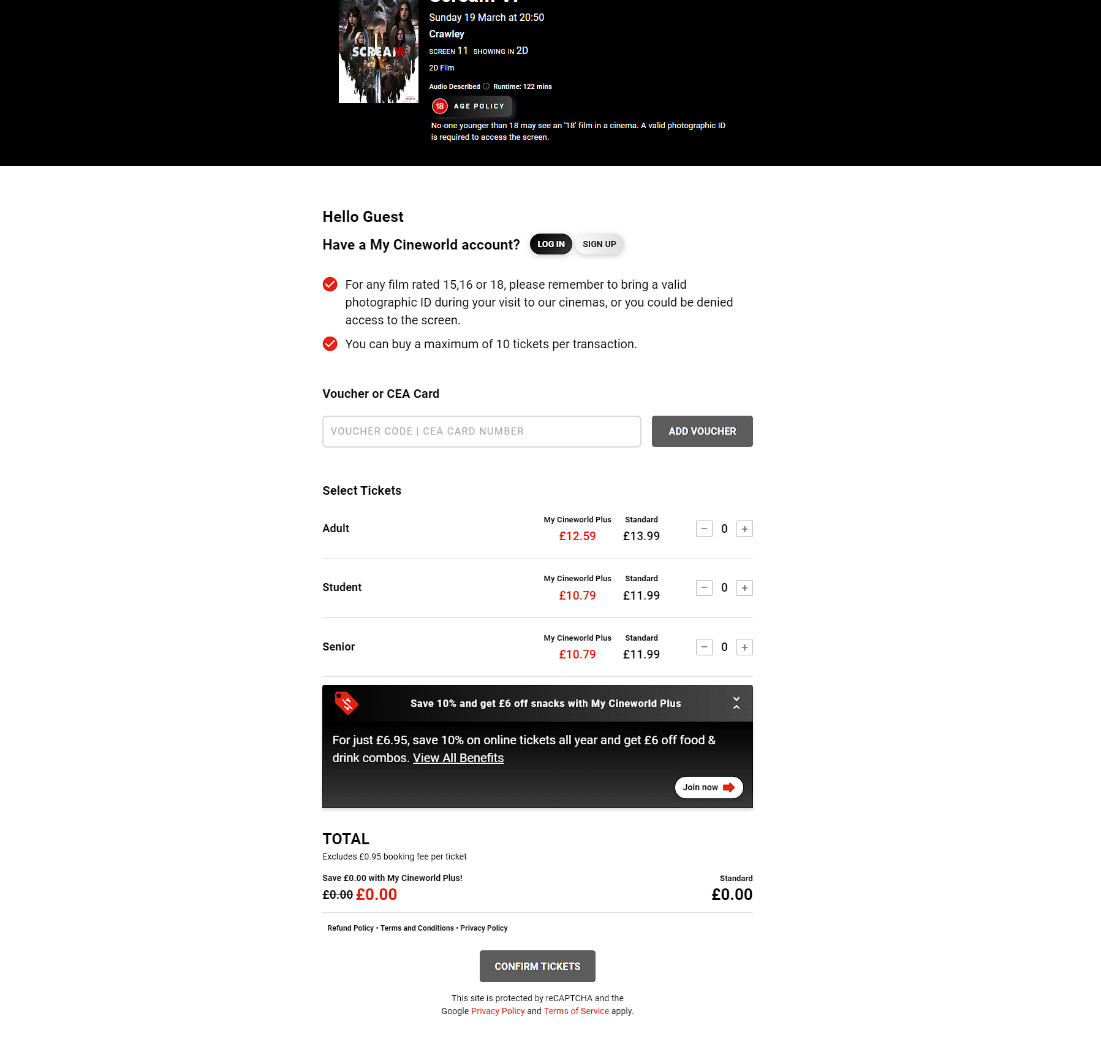
There design includes a lot of bright colours which would lure you into trying to buy a ticket. The whole selection of whats available is shown and this makes the user feel acustommed to the website and feel in power to chose a certain film. This creates a consume mindset which I would like to recreate in my project.



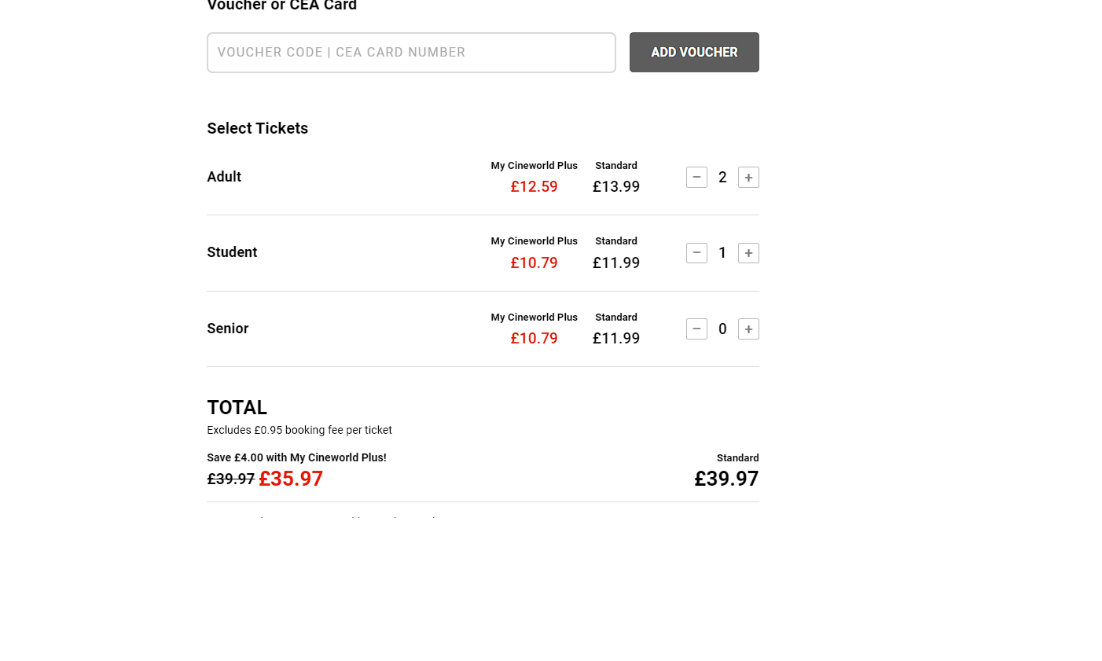
Then booking the tickets is very easy. You click onto the film you would like to watch and then click onto book now. Cineworld shows a trailer of the movie, which in our case could be the preview of the show. The button is made red so that it easily seen. This again makes the user be able to navigate the interface.



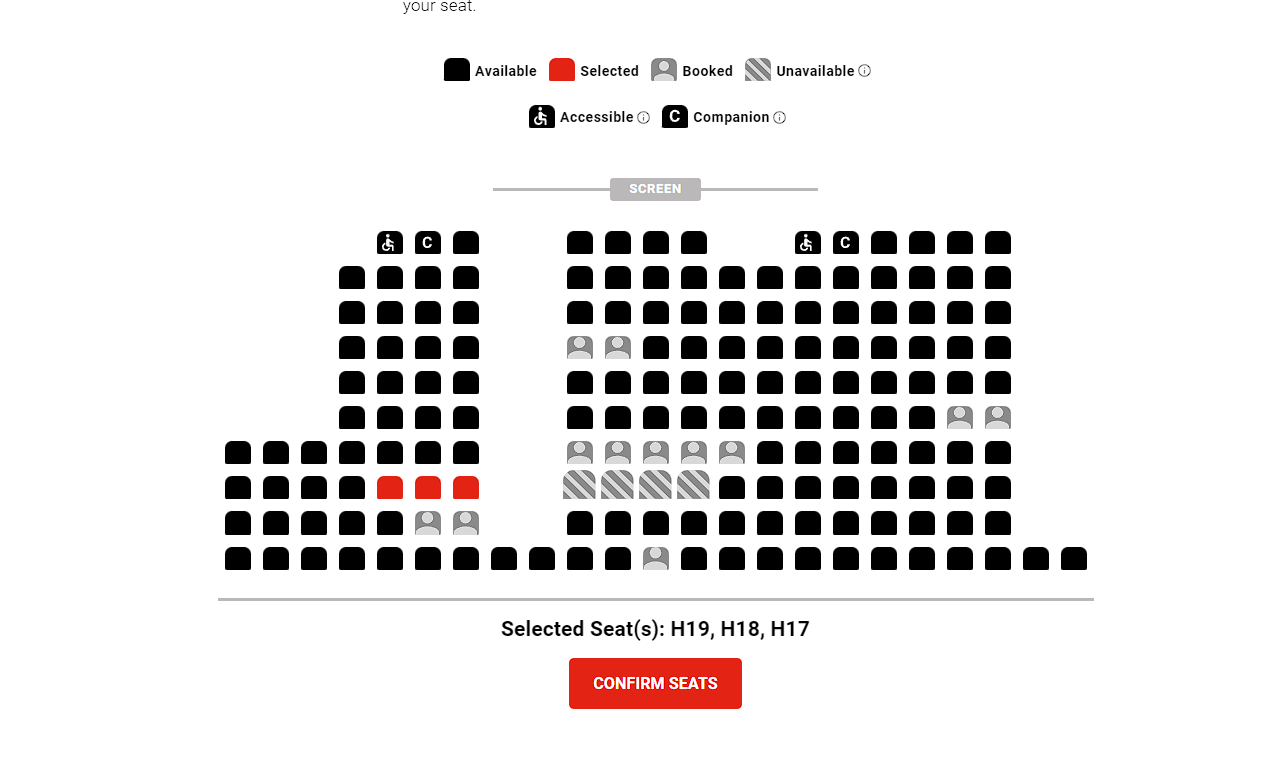
They include a time system which we can implement for our screenings of the play. This will be done with the days of the week as the show will only run for a couple of days. I would again like to point out that the interface is simplistic in a nature to not confuse the user of what is happening.



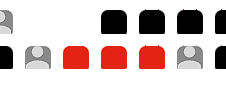
Cineworld then shows us the pricing of the tickets and allows us to choose the tickets we need and want. There is a login that pops up as you try to book the tickets. There is a counter at the bottom to show how much the tickets will cost. This is a good feature to include as it will show the user the amount, they are spending so that there are no calculations or nasty surprises at the end. I would include this in my interface by making the counter in a separate window which would show the price at any given point until the checkout. There is a confirm seat button which just adds verification to the process as you have to make sure you want to buy the tickets you selected.



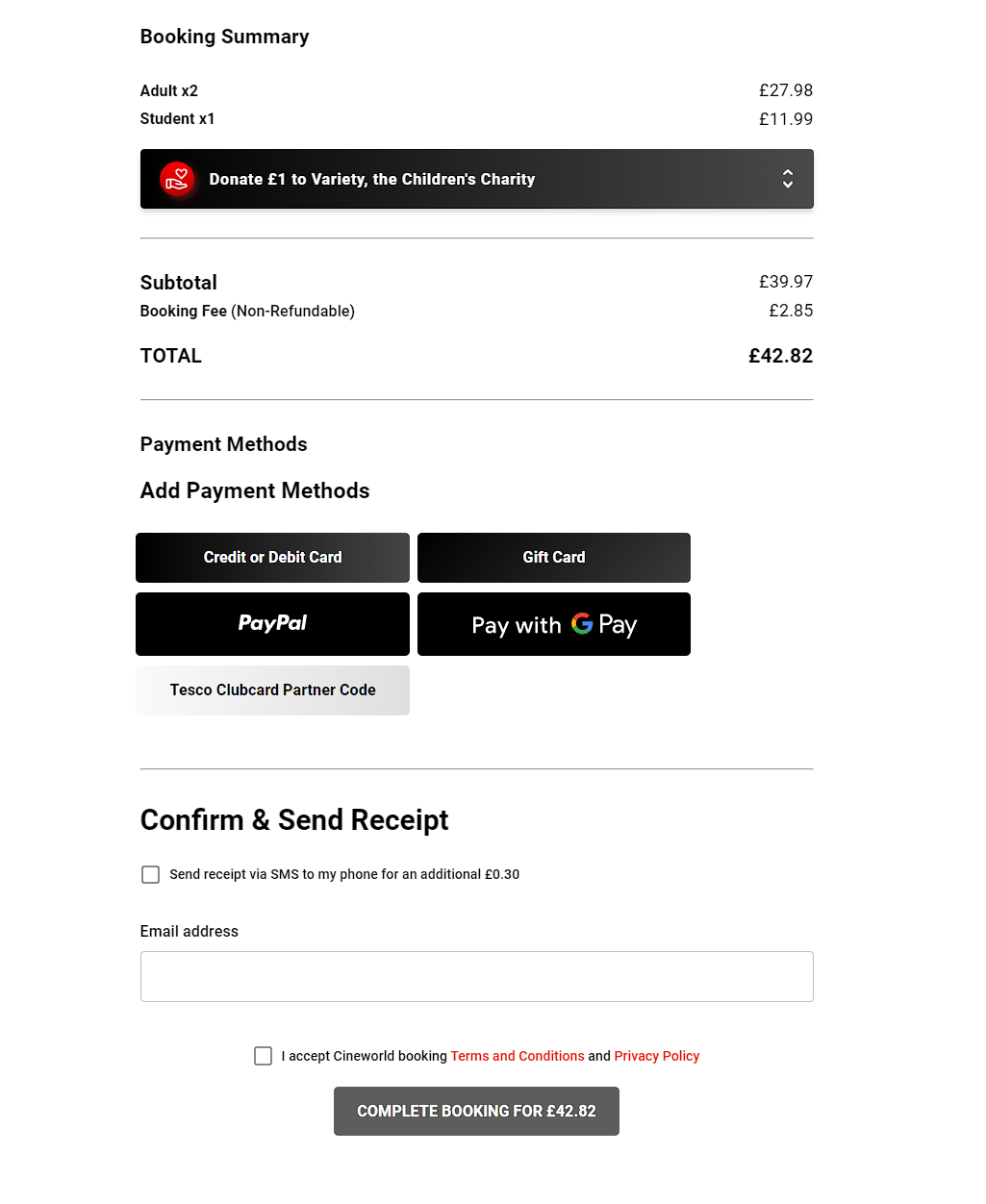
After choosing your tickets you have a counter of the tickets you chose, and the price is calculated at the bottom. They promote a discount by having a membership. This could be done in my interface, but I do not see a point of a discount on a school play. The counter feature is very handy though and will be included.



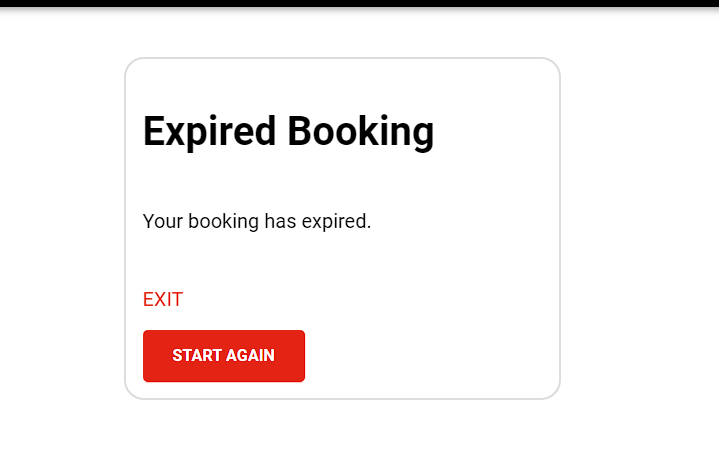
You then have a seat map where you can choose your seating arrangement. This should be done with my interface as it is a necessary step in the program. The seats also have validation (shown below):



The interface did not allow me to pick the seat with a one space gap. This type of validation is particularly important as it means that some of the members of audience will not be split up. What I do not like is that you can pick the seat between the rows so you can split up the people. This can be a way to make a lot of single spaces as most rows are even numbers.



Finally, there is a checkout page where you can confirm and pay for your tickets. Your earlier presented options are once again displayed so that you can check that the tickets you chose are the right ones for you. There is a confirm and send button which will email a confirmation to you. This will be included in my interface.



One bit I do not like is that there is a time limit for your tickets, so it forces you of your booking as the time runs out. This could be annoying as you must fit within a certain frame of time so that your booking does not expire.

### Stakeholders

What are the [stakeholders](https://www.bbc.co.uk/bitesize/guides/z4gcd2p/revision/1#:~:text=Stakeholders%20are%20individuals%2C%20groups%20or,They%20include%3A&text=Workers%20who%20want%20to%20earn,quality%20products%20at%20reasonable%20prices.) and what are their requirements for the solution, these should be as diverse as possible and will have different requirements.

Students (Actors) – The students need it for looking at how many people are coming and to see whether they will be performing on certain days, people may not have booked.

Director of play – Needs to know if people book so that they can then make the play.

Audience – Will need to know available days and seats. See bookings, and make sure that there are no overbookings.

Finance team – Will need to know how many people are booked on to calculate profits and see how much to charge people and who to charge.

Security team – Need to know turnout of people so that they can maintain a safe environment by having enough security in comparison to people.

Special Guests – need to have seats reserved and prepaid for them.

### Limitations of current system

The current system is paper based with the performing arts teachers taking notes about any tickets sold. This makes the system slow as it must be checked by paper. Then you have the whole idea of the system not being online. This means that there would be a need for a receptionist type worker which would receive calls from the people who want to book and write in their details. This makes the entire process inefficient as you could only do this in certain working hours, and you cannot check the bookings or seats as they aren’t online. This adds a limitation of the system as the users cannot see if they want certain seats and cannot check their bookings without calling the receptionist. Getting the tickets becomes difficult as the system to obtain them may be by post or through the reception at Collyer’s. This makes the process long as the ticket collection either causes queues or if lost in post, missing tickets.

## Design

### Sub programs

Login function:

This will mean that the user can log on to access special features. The feature will open more options to the user so a totally different gui will have to be presented to the user. This will mean that the data is safe from normal people as they cannot access certain functions. This will need validation so that there will not be a sql injection type attack.

Validation for all input:

This will mean that there wont be any error full statements that can be inserted eg sql commands. This makes the interface secure.

Home Page:

A menu bar at the top of the page featuring options for 'Book Ticket', 'Search Customer', 'View Performance Revenue', 'Block Seats' and 'Save Data'.

This should link to all the allocated buttons to traverse the interface.

Will include tabs in the top of the interface and the user can use the tabs to navigate through the interface. There will be an order so that the user cannot jump ahead and get features that they have not yet selected. Eg the actual booking of the seats before selecting the seats that they need.

Book Ticket Page

A form featuring fields for the customer’s name, type, phone number, performance date, seat booked, and price paid.

This should also have a seat map so that customers can click onto the seat they want. This will be designed with buttons that can have different states.

Seat map:

The seat map will include a fully custom seating plan where the user can select and confirm the tickets that they want. This will be colour coded so that there wont be any confusion. I will also add some seats that are blocked off by a certain hashy pattern which will make the interface look better.

Block tickets:

Only allowed to users with special privilege.

A form featuring fields for the performance date and seat number to be blocked.

This will only be allowed on the seat maps and for users with clearance so once again a login system is implemented.

Search Customer Page:

again a special user privilege.

A search bar to enter a customer name and a display area to show the customer's booked tickets and corresponding performances.

Can only be accessed by certain people. Will have a logon button that can filter through the teachers that have access to the Customers data for security reasons.

View Performance Revenue Page:

A display area to show the total number of tickets sold, tickets remaining available, and total revenue for all performances.

Also has a logon system so that the data is kept secure.

Linked to SQL server to see the inputted data from earlier.

Save Data:

Each button that says confirm or enter will save all relevant data to an external file, this will save email and phone and customer detail to SQL file so that the data can be accessed later. This can then be used for showing the confirmation or to send the recipient a copy of the tickets.

Ticket distribution:

Will send out tickets via email. Will need to edit certain elements of the ticket eg the name the seat number etc. This will make the ticket look professional as the users name and their seat number will be shown. This can be done by processing certain info from the sql file that we stuffed the data to earlier.

### Task List

For each task (sub program) you need to identify the inputs, outputs and processing

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Input | Output | Processing |
| Home page | Clicking onto the header that you want: eg a book ticket header. | Will take you to the designated page that you want to go to. |  |
|  |  |  |  |
|  |  |  |  |

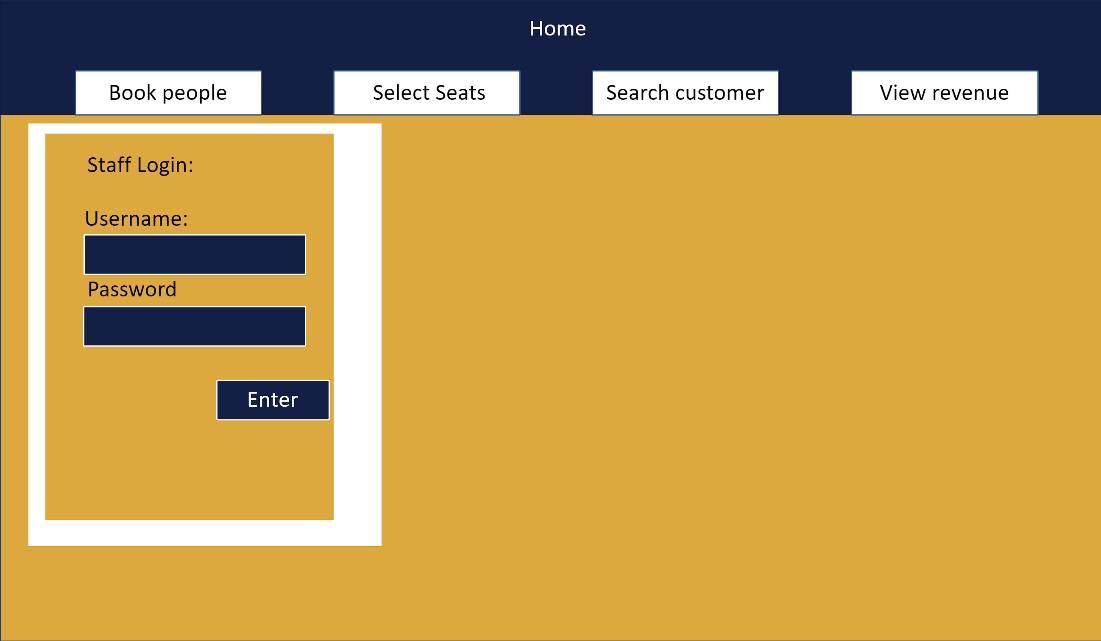
### Algorithms

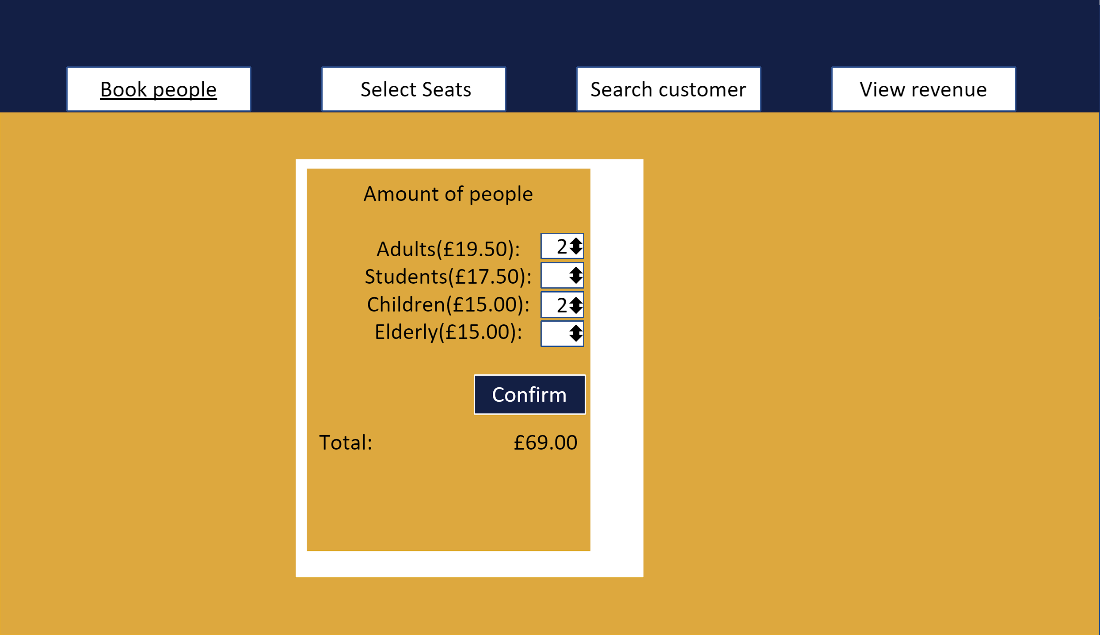
Each subtask should have its own algorithm written in pseudocode, these should be clear enough for a third party (fellow students) to follow, consider if someone in the class could make your system from these designs.

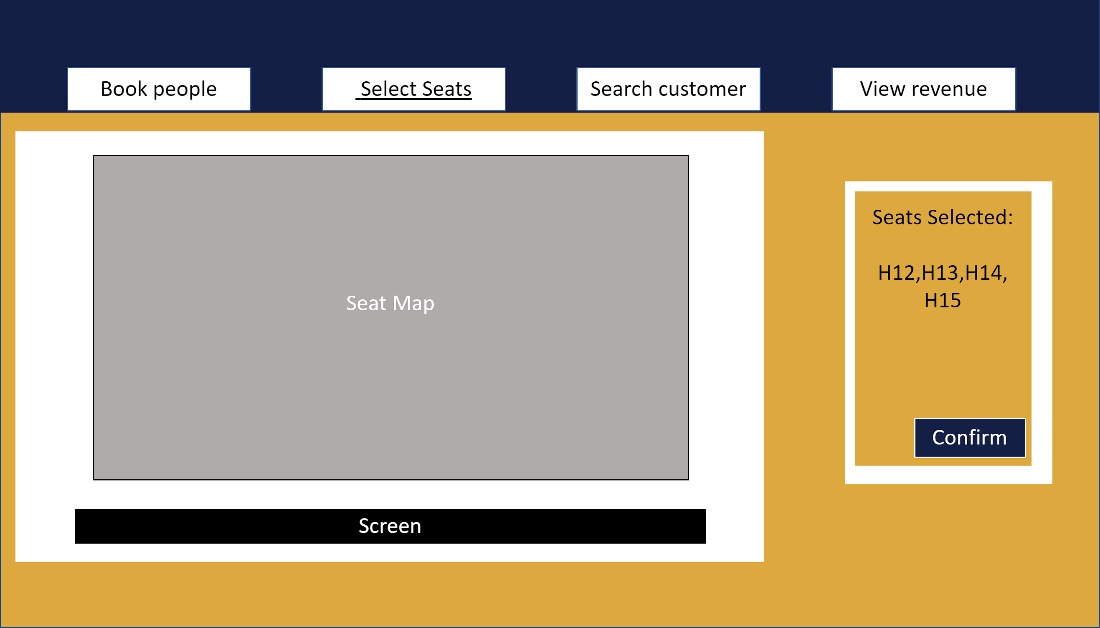
These should be lots of separate algorithms, I do not need to see any algorithms for the user interface as these are designed above.

### User Interface

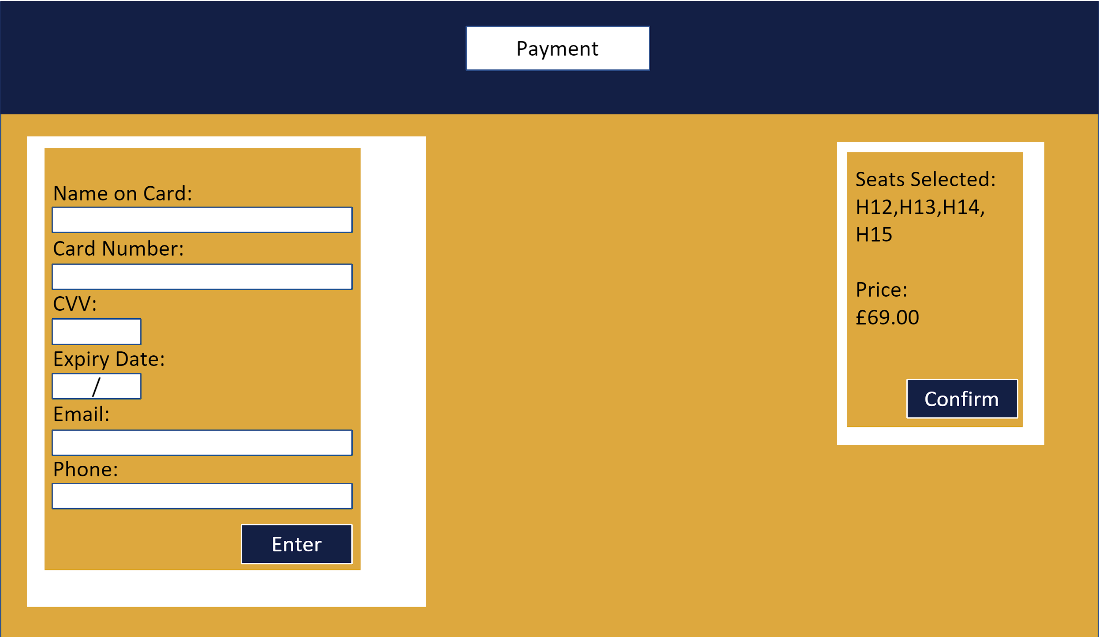
Draw annotated designs of your user interface. These should be detailed enough that another person could make your system and it will look as you envision.

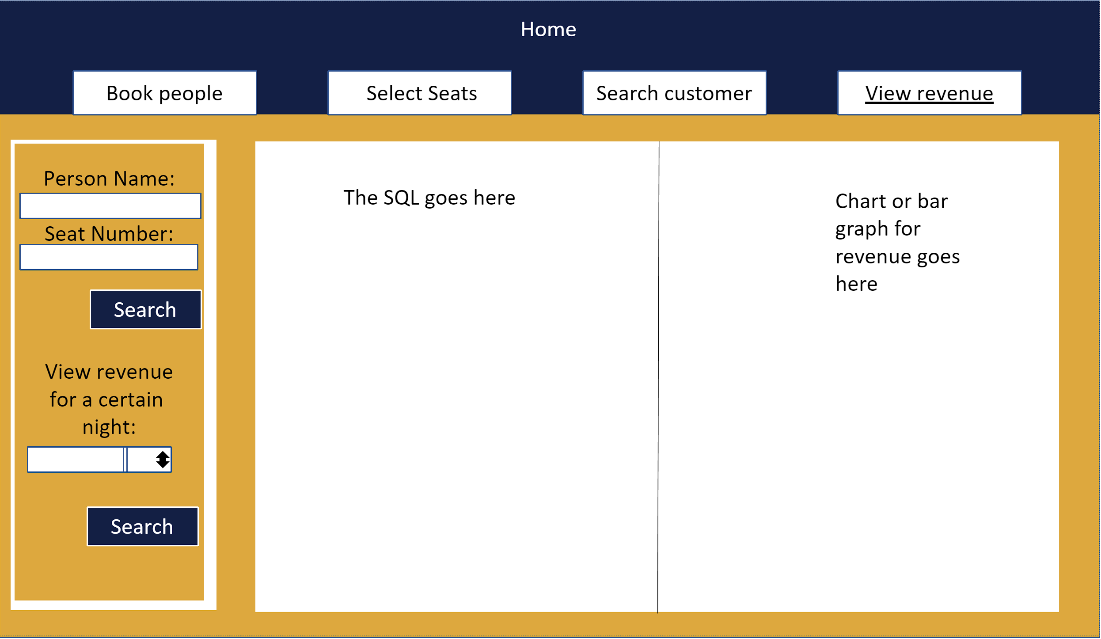






Added inbetween select seats and search customers





### UML

Include all your UML here, this could be an ERD, DFD or class diagram

Graphical user interface

Description automatically generated with medium confidence

### Data Dictionary

Fill in the table for each of your entities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute/Field | Data type | Validation | Key | Comments |
| PerformanceID | Int | Varchar(100) | Primary | Key that will link to booking so that the night can be selected |
| Seats\_Available | Int | Varchar(600) | Null | This will show the user how many seats available for each night. |
| Date | Date | Varchar(10) | Null | DD-MM-YYYY |
| Type\_of\_Seat | str | Presence check() | Null | This is to show what type of seat the user is selceting eg normal, disabled, blocked etc. |
| BookingID | Int | Varchar(600) | Primary | This will be the seat number which is assigned to the button that the user presses onto. |
| PerformanceID | Int | Varchar(100) | Secondary | This links the performance to the BookingID |
| CustomerID | int | Varchar(100) | Secondary | This links the Customer table to the booking ID and thus also to the PerformanceID |
| Seats\_booked | int | Varchar(10) | Null | This will assign the number of seats that the person booked. |
| CustomerID | Int | Varchar(600) | Primary | This is the unique code of the customer |
| Fname | str | Varchar(20) | Null | This is the customer's name so that they can be referenced on the booking confirmation. |
| Sname | str | Varchar(20) | Null | This is the surname so that it can be referenced on the booking confirmation. |
| Phone | int | Varchar(11) | Null | This is the phone number so the customer can be contacted. |
| Email | str | Varchar(30), PresenceCheck() | Null | This is so that the customers verification can be sent through. |
|  |  |  |  |  |
|  |  |  |  |  |

## Software Development

All your code should be annotated with suitable comments and doc strings, use <http://hilite.me/> to format your code as this retains the colour coding and lets you copy and paste the code in the correct format.

## Developmental Testing

Identify at least 5 different problems you had and your solutions, there should be before and after images.

## Testing

Complete the testing table, you must comprehensively test your solution using a range of testing criteria. Testing should include:

* Valid data – data with expected outcomes that should be accepted by any validation you have included
* Invalid data – data that should be rejected with an error message
* Erroneous data – symbols etc in text fields that could break everything
* Extreme data – only used for range checks, data that should be accepted but is on the edges of acceptable.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test | Test type | Test data | Expected result | Actual result | Commentary |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## Evaluation

You should write a detailed evaluation covering the following points

* How effective the language including libraries etc you chose is for solving this and what features you used.
* Compared the system you have made with those you researched for your desk based research.
* Identified strengths and weaknesses of your system with potential improvements, you don’t have to program these so you can be imaginative
* Your own strengths and weaknesses, consider each stage of the process
* Identify any personal changes you would make for the future