Students' Auditorium Management Software(SAMS)

Test Results Document

Gurram Dhanunjay(22CS10029)
Sai Deepak Reddy Mara(22CS10066)
Eswara Aditya(22CS10023)

TABLE OF CONTENTS

1. Black Box Testing

All the black box tests are done assuming that the database is connected. Else in SQL Exception is thrown and a stack trace is printed.

1.1 BBT for Login

This method prompts for the input of UserID and Password of the employee. It verifies the same with the database.

Type	Case	Expected Output	Actual Output	Result
Invalid	Any field is left empty	Unable to log in	A message appears “Fill out this field”	Pass
Invalid	An Invalid Username is entered	Unable to log in	Displays a pop up saying”Invalid Credentials”	Pass
Invalid	Username is valid, Password is Invalid	Unable to log in	Displays a pop up saying”Invalid Credentials”	Pass
Valid	Appropriate Log in details of Show Manager are given	Show manager is able to login	Show manager dashboard is opened.	Pass
Valid	Appropriate Log in details of Account Clerk are given	Account clerk is able to login	Account clerk dashboard is opened.	Pass
Valid	Appropriate Log in details of Sales Person are given	Sales Person is able to login	Sales Person dashboard is opened.	Pass
Valid	Appropriate Log Info. of Spectators are given.	Spectators is able to login	Spectators dashboard is opened.	Pass

1.2 BBT for Adding a Sales Person

This method prompts to enter the sales ID, name, username and password of the sales person to be added.. It updates the database accordingly.

Type	Case	Expected Output	Actual Output	Result
Invalid	Any field is empty	No field entry	A message appears: "Please fill out this field"	Pass
Valid	Appropriate entries in all the fields	Sales person account is created	Redirects to sales person list Showing the added sales person	Pass

1.3 BBT for Changing Password

This prompts the user to enter a new string to update/change the password. Accordingly, database is updated.

Type	Case	Expected Output	Actual Output	Result
Invalid	An empty string is password is said to be updated or confirm password is not entered	Password is not changed	The page reloads and the password is not changed	Pass
Valid	Any non-empty string is entered	Password is changed	Redirected to login page and password is updated	Pass

1.4 BBT for Adding a show details

It prompts the manager to enter the name of the show, the prices for balcony and ordinary seats and also the balcony and ordinary seats available for booking, No of shows and Start and end times of the show

Type	Case	Expected Output	Actual Output	Result
Invalid	Any of the fields is empty	Error Message	Shows “Fill out this field” message	Pass
Invalid	Strings are entered in any of the integer field like ticket prices and seats available	Error message	Show “Enter a number” message	Pass
Invalid	Seats prices are floating point numbers.	Error Message	Shows “Enter a valid value” message	Pass
Invalid	Balcony seats have lower price than Ordinary Seat	Error message	Shows message “Balcony seat price should be more than ordinary seat price”	Pass
Invalid	Dates and timings are not entered	No Change in database	Error message prompting user to fill all rows.	Pass
Valid	Every field is entered with appropriate values	Show is listed in the database	Redirects to shows list showing the update shows list.	Pass

1.5 BBT for Adding a date and time for a show

User is prompted to enter the date for a show and select timings for that show and on that day. Database is updated accordingly.

Type	Case	Expected Output	Actual Output	Result
Invalid	Date is selected but time is not selected	No change in database	Show details are not updated in the database and shows message “Fill out this message”	Pass

Invalid	Start time is later than end time	No change in database	Shows message “End time must be after start time”	Pass
Invalid	Selected Date is in the past.	No change in database	Shows message “Cannot add shows for past dates”	Pass
Valid	All fields are appropriately filled	Accepts the show details	Database is updated with these show details	Pass

1.6 BBT for Displaying Balance Sheets

The Show Manager is prompted to select either a show or a year to view the balance sheets.

Type	Case	Expected Output	Actual Output	Result
Invalid	Check balance Sheets for a year when there are no shows	No balance sheets displayed	A message displaying ”No Data Available”	Pass
Invalid	Check balance sheet for a show when there are no shows	No balance sheet is displayed	In the first place no such show is displayed in the list to select	Pass
Valid	A valid year is chosen	Balance sheet should be shown	All the expenditures and sales incomes of that year is displayed.	Pass
Valid	A valid show is chosen	Balance sheet is displayed	All the expenditures including logistics, artist payments and sales income are displayed.	Pass

1.7 BBT for Selecting a show

Users are prompted to first select a show name and then show date and finally show time. Show time can't be selected without selecting the show date.

Type	Case	Expected Output	Actual Output	Result
Valid	A show is selected from the list of displayed shows	Show is selected	That show is selected and proceeds for further usage	Pass

1.8 BBT for Booking a Ticket

Users are prompted to first select a show , show details like its name, timing are shown to select ,further we can select the salesperson and the seat we want to book

Type	Case	Expected Output	Actual Output	Result
Valid	All options are properly selected (there is no possibility of incorrect input since the input is selected from list and not entered)	Ticket is booked	Database is updated and the booking of the show is updated and can be seen in my_bookings	Pass

1.9 BBT for Canceling a Ticket

Users are shown their bookings in my bookings page and can cancel by clicking the cancel button.

Type	Case	Expected Output	Actual Output	Result
Valid	The cancel button is clicked for their booked tickets	Ticket is canceled, seat is updated empty, and refund is issued accordingly.	Database is updated for seats and transaction and refund is issued.	Pass

1.10 BBT for Entering Expenditures

Accounts clerk is prompted to select a show and enter the expenditures for that show like payments to artists, logistics, etc. Database is updated accordingly.

Type	Case	Expected Output	Actual Output	Result
Invalid	All or any fields are empty	Database is not updated	Shows Message “Fill out this field” And Database is not updated’	Pass
Invalid	String is entered in expenses field	Error message	Shows message “Enter a number”	Pass
Valid	All fields are appropriately filled	Expenditures are updated in the database	Expenditures are added for that particular show and database is updated.	Pass

1.12 BBT for Querying Seats

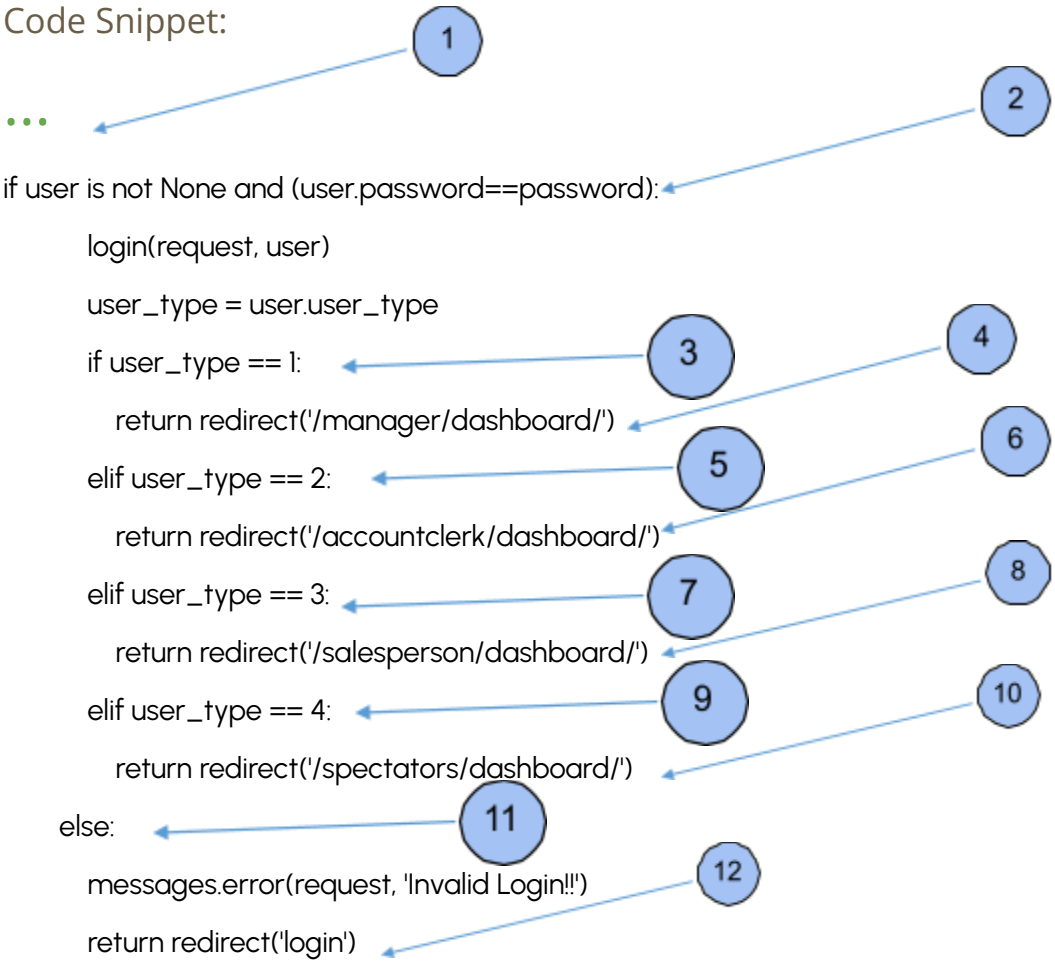
The user is prompted to select a show to see seat availability.

Type	Case	Expected Output	Actual Output	Result
Valid	Show is selected properly	See the seats available	Available seats are displayed as a list	Pass

2. White Box Testing

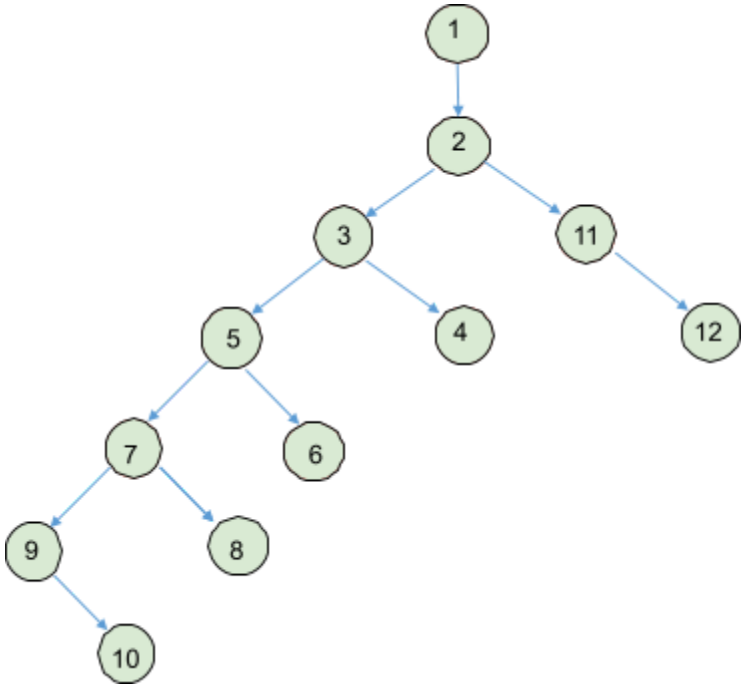
2.1 WBT for Login

Code Snippet:




```
...
if user is not None and (user.password==password):
        login(request, user)
        user_type = user.user_type
        if user_type == 1:
            return redirect('/manager/dashboard/')
        elif user_type == 2:
            return redirect('/accountclerk/dashboard/')
        elif user_type == 3:
            return redirect('/salesperson/dashboard/')
        elif user_type == 4:
            return redirect('/spectators/dashboard/')
    else:
        messages.error(request, 'Invalid Login!!')
        return redirect('login')
...
```

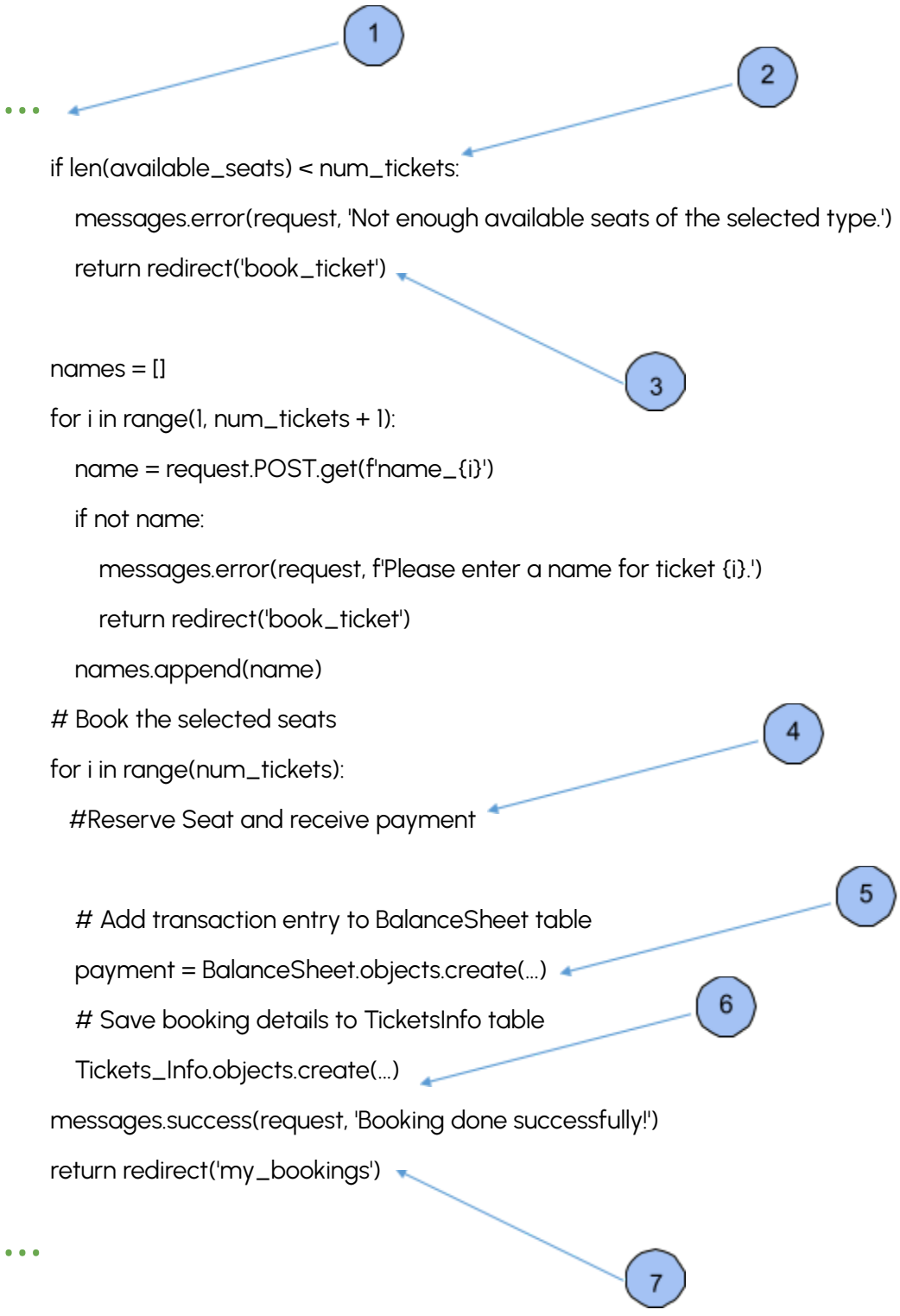
Path Diagram



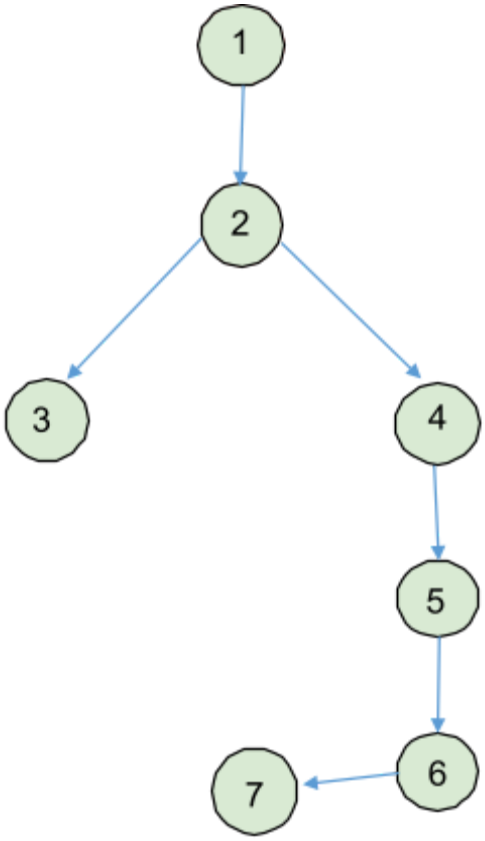
Path - 1	1-2-11-12
Path - 2	1-2-3-4
Path - 3	1-2-3-5-6
Path - 4	1-2-3-5-7-8
Path - 5	1-2-3-5-7-9-10
Path - 1 Expected Result	Invalid Credentials
Path - 2 Expected Result	Show Manager logins
Path - 3 Expected Result	Accounts Clerk logins
Path - 4 Expected Result	Sales Person logins
Path – 5 Expected Result	Spectator logins

2.2. WBT for Ticket Booking

Code Snippet:



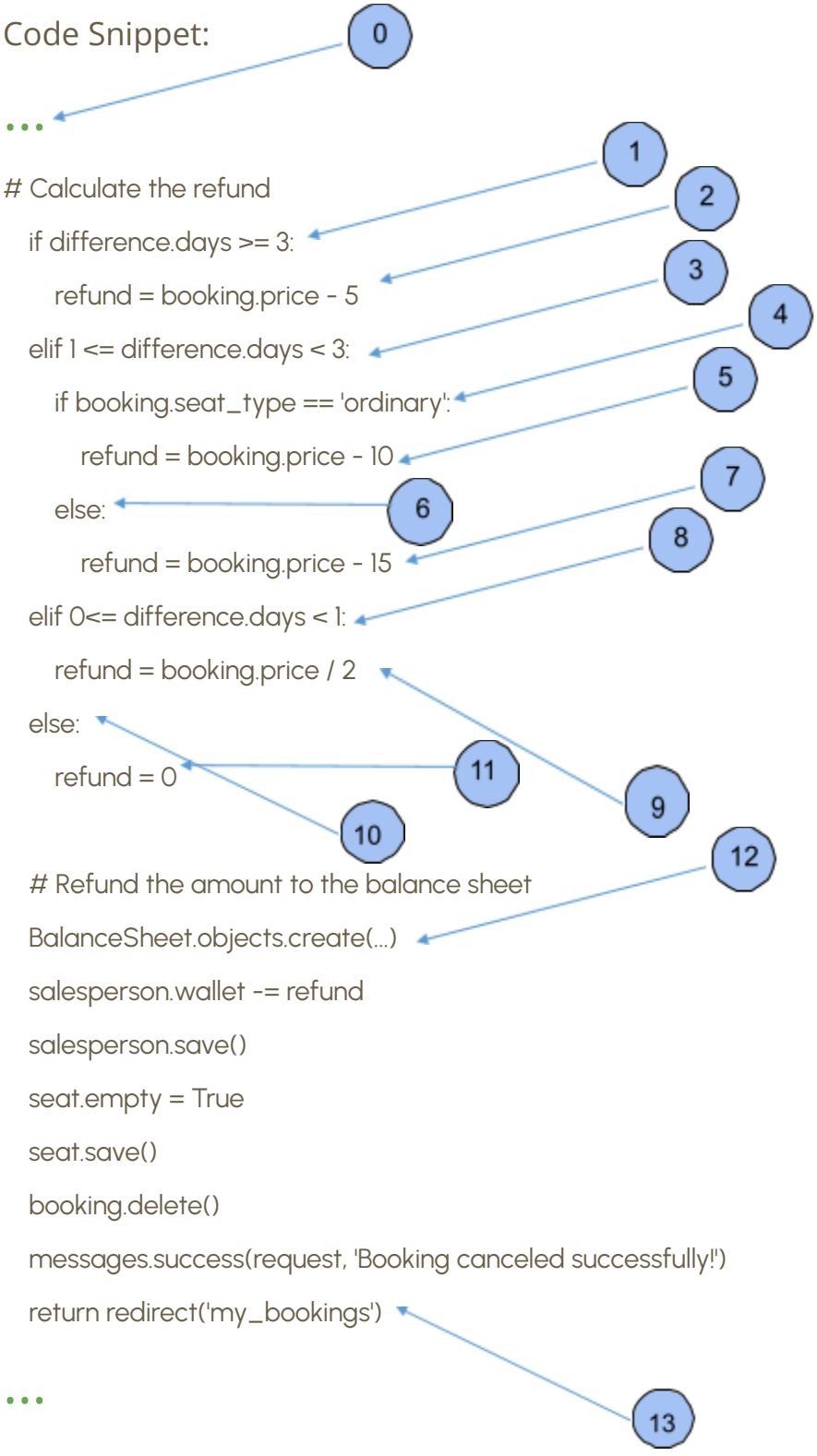
Path Diagram



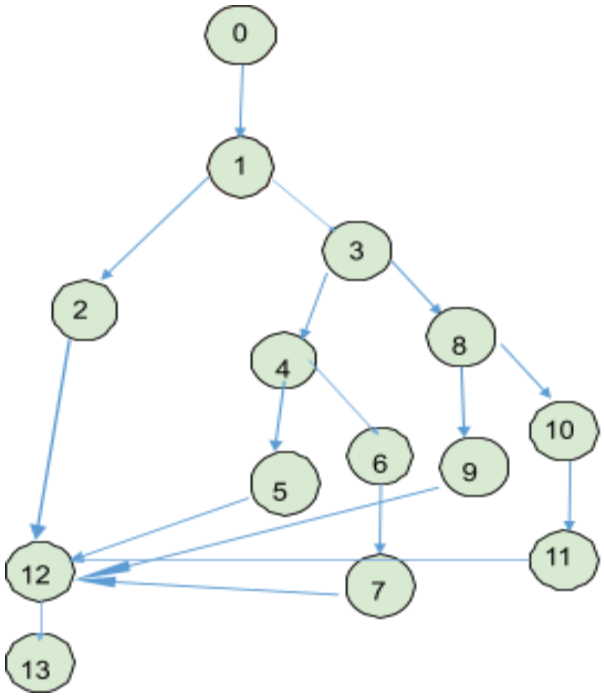
Path - 1	1-2-3
Path - 2	1-2-4-5-6-7
Path - 1 Expected Result	Required seats aren't available, prompting user to choose less no. of seats.
Path - 2 Expected Result	Required seats are available, tickets are booked, payments are received and the database is updated accordingly.

2.3. WBT for Ticket Cancellation

Code Snippet:



Path Diagram



Path - 1	0-1-2-12-13
Path - 2	0-1-3-4-5-12-13
Path - 3	0-1-3-4-6-7-12-13
Path - 4	0-1-3-8-9-12-13
Path - 5	0-1-3-8-10-11-12-13
Path - 1 Expected Result	Refund before 3 days
Path - 2 Expected Result	Refund between 1 & 3 days(Ordinary)
Path - 3 Expected Result	Refund between 1 & 3 days(Balcony)
Path - 4 Expected Result	Refund on Show Day
Path – 5 Expected Result	Refund after show(i.e, no refund)

3. User Interface Testing

After running the code, we get a display of Sign-In. If the spectators want to create an account, they will choose to do Sign-Up(This feature is only available to the spectators).

3.1 Login Window

What is tested?	All the components with which the user interacts with.
Inputs	• User enters Username and their password • User presses Sign-In Button
Expected Result(s)	If the user enters correct details then the Sign-In would be successful. Else an error message will be displayed.
Effective Result(s)	All results match according to our expected results.

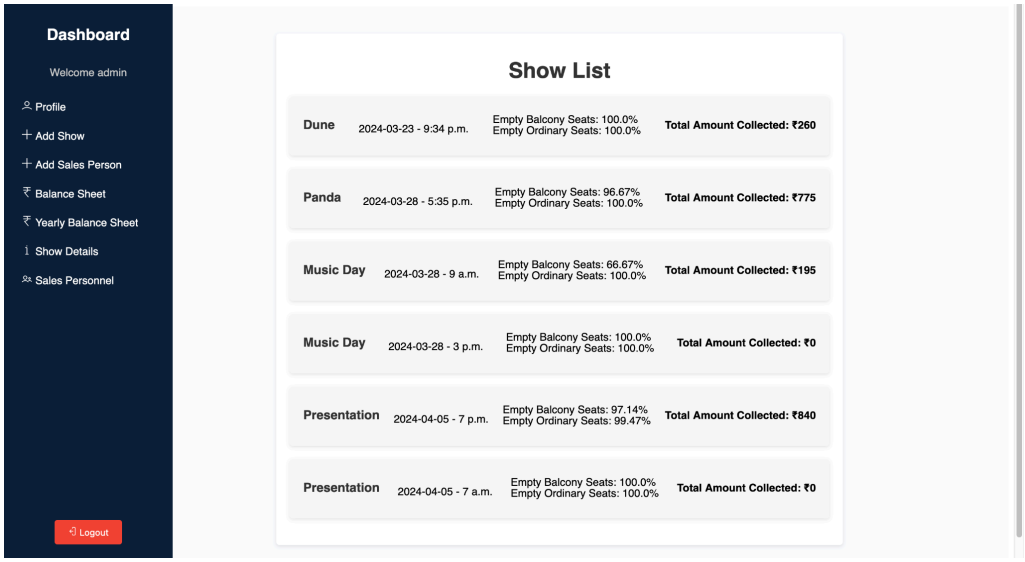
3.2 Sign-Up Window

This feature is only available to the Spectator. The window appears as follows

What is tested?	All the components with which the user interacts with.
Inputs	• User enters their name, User ID, their password and then confirms it. • User presses the Register button.
Expected Result(s)	Account successfully gets created and the corresponding box appears.
Effective Result(s)	• All results match our expected results.

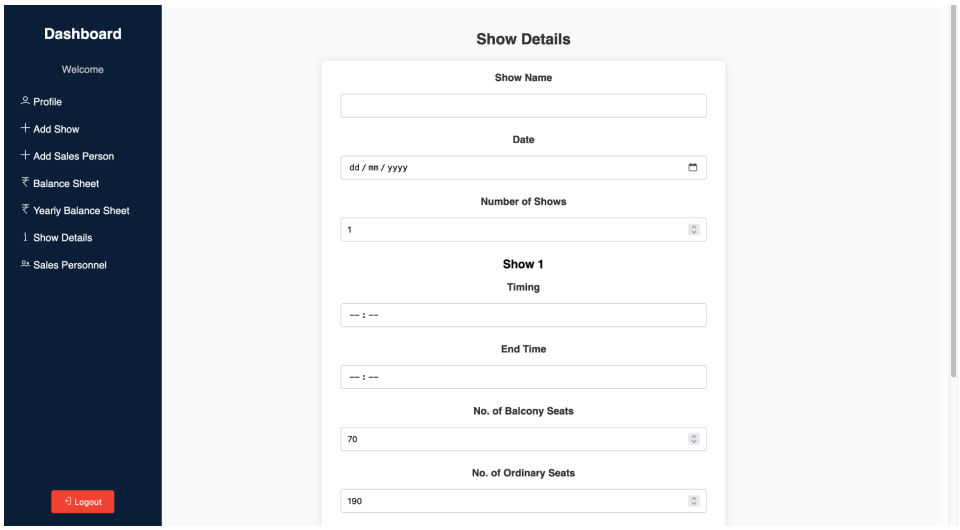
3.2 Show Manager Dashboard/Show Details Window

After the manager logins successfully, the following will be displayed



What is tested?	All the components with which the user interacts.
Inputs	• Profile button is pressed. • Add Show button is pressed. • Add Sales Person button is pressed. • Balance sheet button is pressed. • Yearly Balance Sheet is pressed. • Show Details button is pressed. • Sales Personnel is pressed. • Logout button is pressed.
Expected Result(s)	• Every button will correspond according to its functionality.

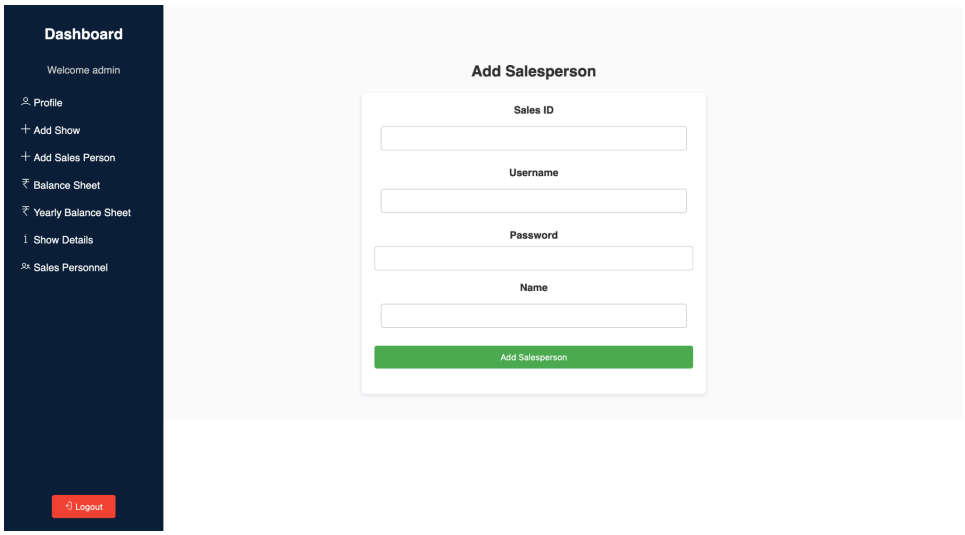
3.4 Add Show Window



What is tested?	All the components with which the user interacts.
Inputs	• Name of the show is entered • Date of the Show as well as the Start time and the End time is also added. • Number of Balcony seats and Ordinary seats as well as their prices can be added • Submit button is pressed
Expected Results	• Show will be created successfully according to the given prompts.
Effective Results	All results match the our expected results

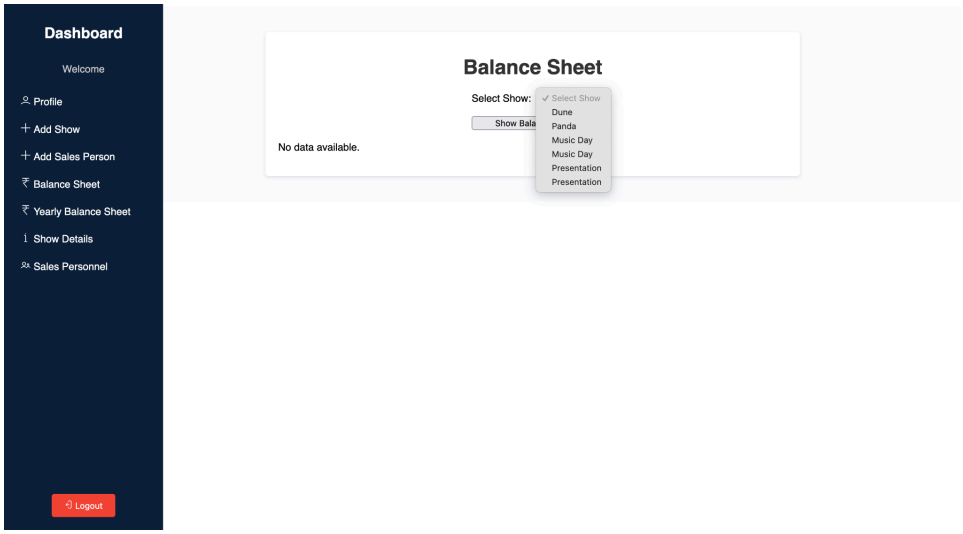
3.5 Add Sales Person Window

The manager can add a sales person. If the user hits the Add Sales Person the following will be displayed:-



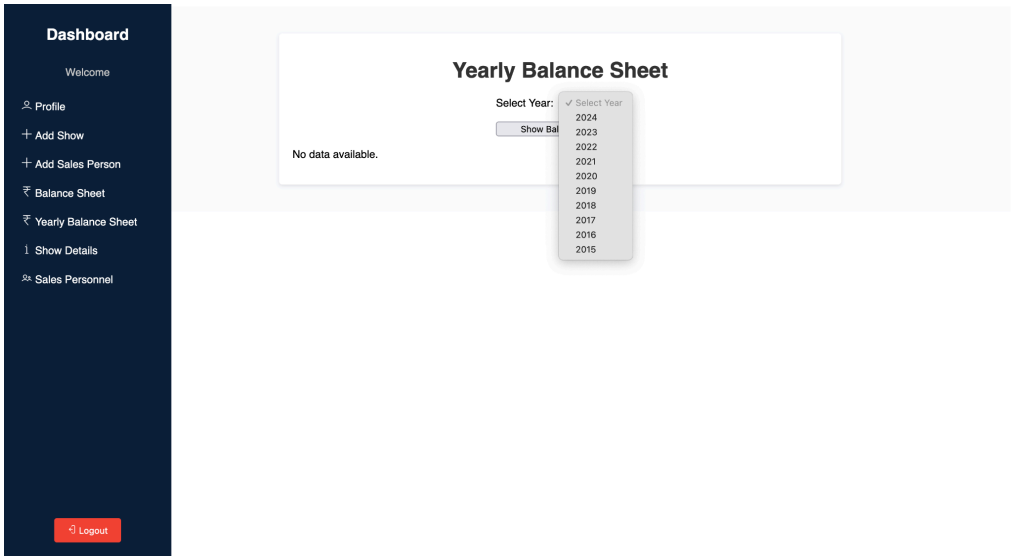
What is tested?	All the components with which the user interacts.
Inputs	• Sales ID is entered • Username, Password and Name are entered • Add Sales Person is pressed
Expected Results	• Sales Person will be added successfully.
Effective Results	All results match to the expected results

3.6 Balance Sheet Window



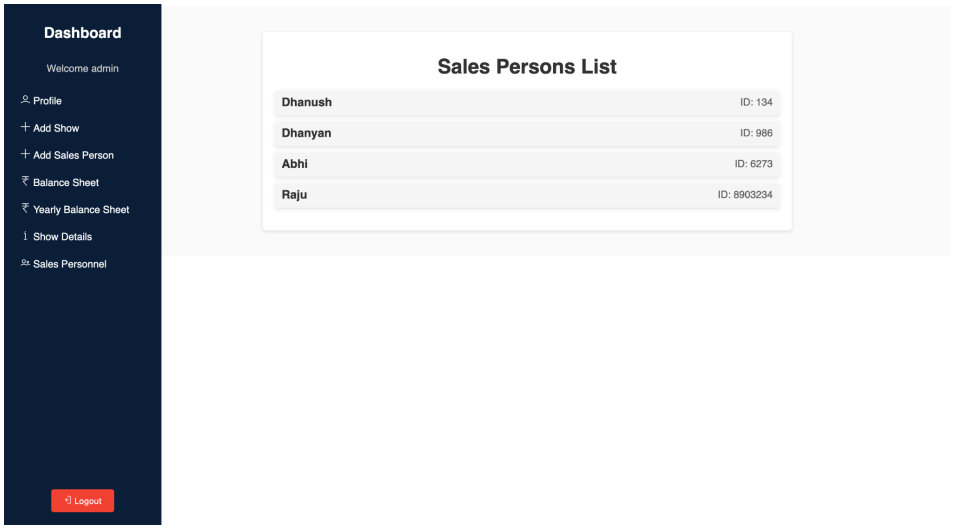
What is tested?	All the components with which the user interacts.
Inputs	• A list will be displayed if the user presses Select Show.
Expected Results	When the user hits the Show Balance Sheet, the Balance Sheet will be displayed successfully.
Effective Results	All results match to our expectations.

3.7 Yearly Balance Sheet Window



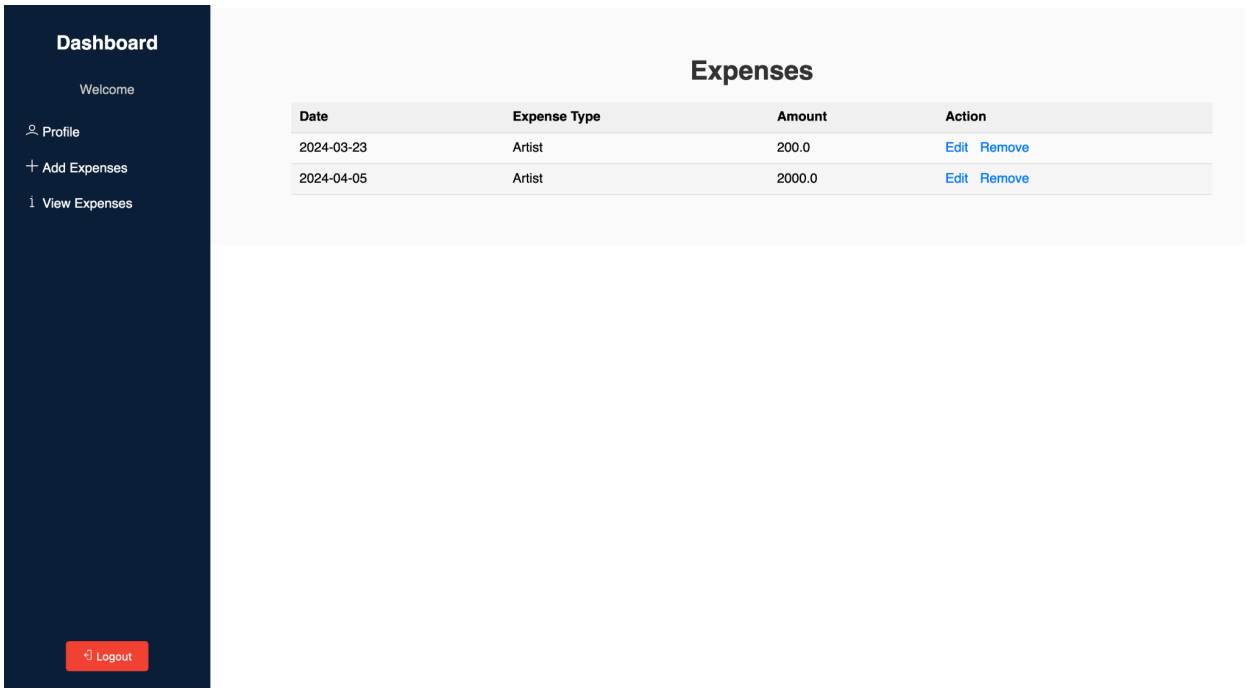
What is tested?	All the components with which the user interacts.
Inputs	• The year is selected from the list which appears when the user hits the Select Year. • Show Balance Sheet is hit
Expected Results	The Yearly Balance Sheet appears as per our expectation.
Effective Results	All results match to our expectations.

3.8 Sales Personnel Window



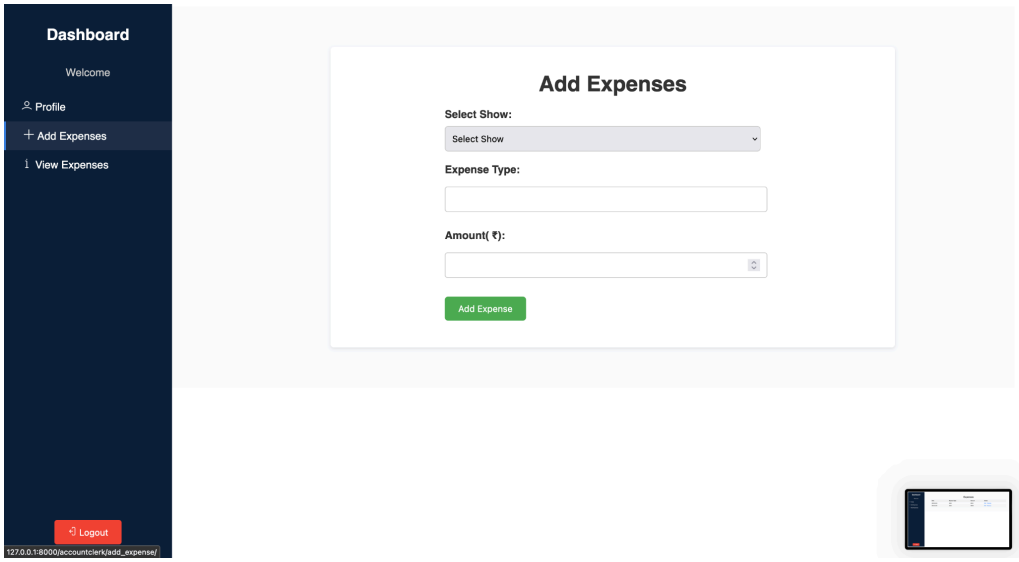
What is tested?	All the components with which the user interacts with.
Inputs	None
Expected Results	Shows the list of Sales Person with their Sales ID.
Effective Results	All results match the expected results

3.9 Account Clerk Dashboard/View Expenses



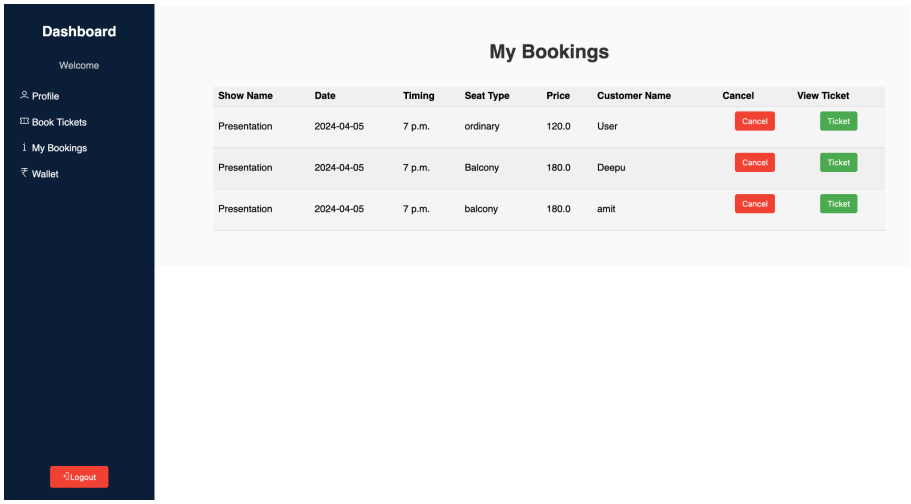
What is tested?	All the components with which the user interacts.
Inputs	• Profile button can be pressed • Add Expenses can be pressed • View Expenses can be pressed. • Edit or Remove expenses can be pressed.
Expected Results	Displays the next window accordingly as per our wish.
Effective Results	All results match our expectations.

3.10 Add Expenses Window



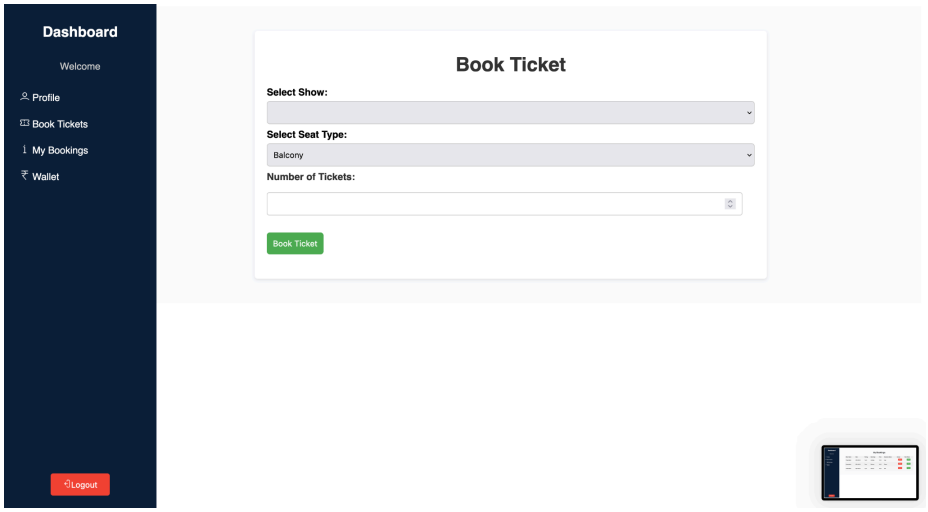
What is tested?	All the components with which the user interacts.
Inputs	• Show can be selected from the list that appears. • Expense type can be entered. • Amount can be entered
Expected Results	Expenses can be added as per the user's wish.
Effective Results	All results match our expectations.

3.11 Sales Person Dashboard/ My Bookings Window



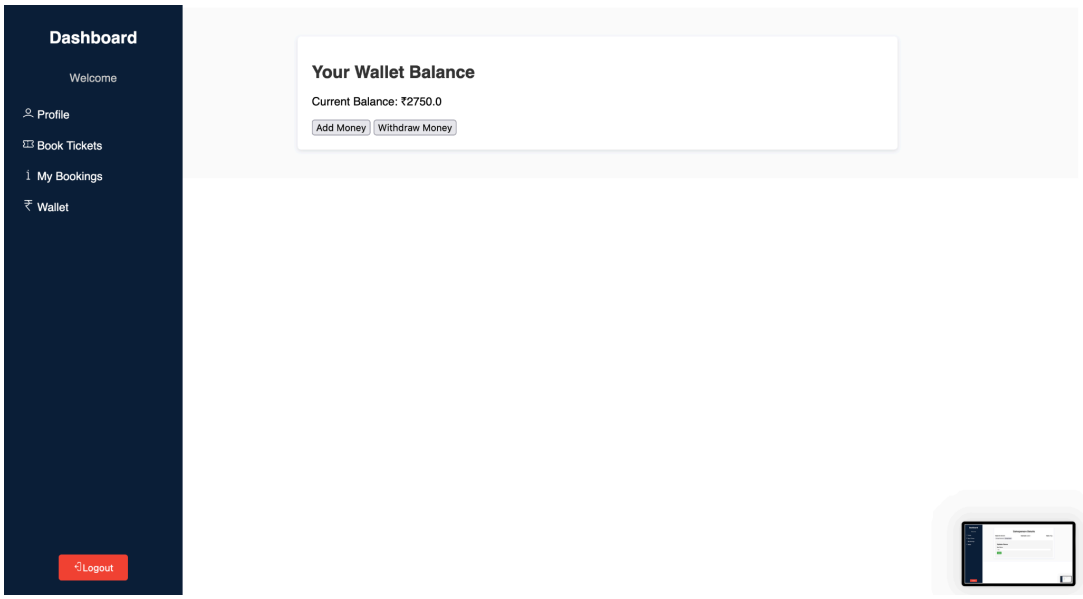
What is tested?	All the components with which the user interacts.
Inputs	• The user can cancel a ticket or view a ticket. • Book Tickets button can be pressed. • My Bookings button can be pressed. • Wallet button can be pressed.
Expected Results	The corresponding windows are displayed as per the user's decision.
Effective Results	All results match our expectations.

3.12 Book Tickets Window



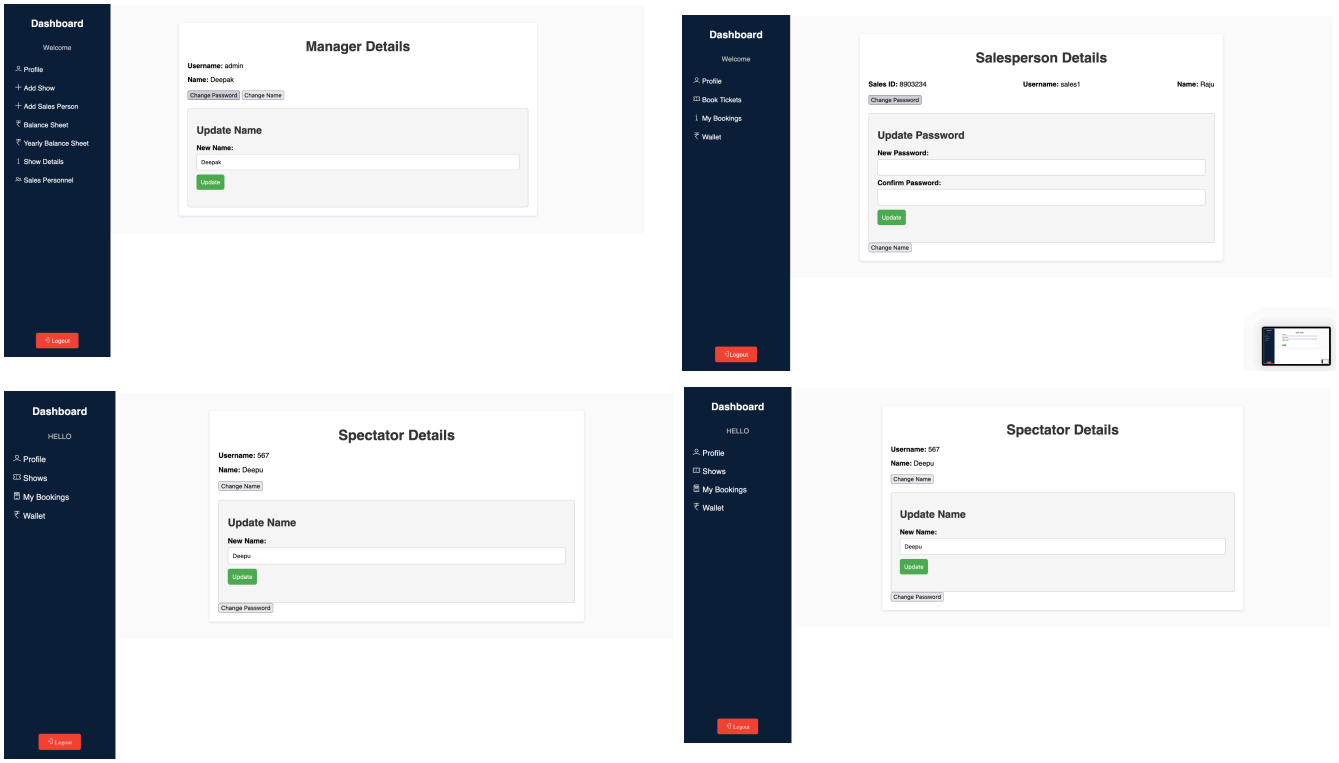
What is tested?	All the components with which the user interacts.
Inputs	• The corresponding show can be selected by the user. • Seat type can be selected (Either Balcony/Ordinary) • Number of Tickets can be entered • Book Ticket button can be pressed.
Expected Results	The ticket gets booked successfully.
Effective Results	All results match our expectations.

3.13 Wallet Window



What is tested?	All the components with which the user interacts.
Inputs	• Add Money button can be pressed. • Withdraw Money button can be pressed.
Expected Results	Money is added when the user hits Add Money button else the money can be withdrawn if the Withdraw Money is selected.
Effective Results	All results match our expectations.

3.14 Profile Window



What is tested?	All the components with which the user interacts.
Inputs	• The user of every type can change their password and name as per their wish, it can be done by hitting the Change Password or Change Name button.
Expected Results	The username's and password's can be changed according to the users willing.
Effective Results	All results match our expectations.

4. Entry and Exit criteria

This part explains the main rules for starting, pausing, restarting, and finishing testing in each phase. If there are any differences in how features or components are tested, those will be noted in their own test plans. Also, when a problem is reported during bug tracking, it will be sorted based on how much it impacts the project.

4.1 Unit Testing

Unit Testing checks the code itself for mistakes like typos, logic errors, or problems with specific parts of the code. Unit tests are created to make sure the program works correctly.

4.1.1 Black Box Phase

Black box testing means testing the software by trying out different inputs to make sure it gives the correct outputs, just like a regular user would do. We'll also use methods like Error Guessing and Boundary Value Analysis to figure out how many test cases we need to thoroughly check the code.

4.1.1.1 Black Box Entry Criteria

Before commencing Black Box testing, it's imperative to have the component specifications and user interface requirements finalized. Here's the checklist for the beginning of the Black Box stage:

- All functions such as Booking Ticket, Canceling ticket, viewing Balance sheets, and managing shows must either be coded or have placeholders called stubs in place.
- We will determine which Black Box Testing Methods to utilize from the outset. Primarily, we'll employ Error Guessing and Boundary Value Analysis.
- Error Guessing involves experimenting with various scenarios such as entering random text in search fields, inputting letters in number fields, restarting the server and application, and inputting incorrect information in different fields.
- Boundary Value Analysis involves testing scenarios like adding show timings that overlap with existing shows or attempting to add a show on a day that has already passed.

4.1.1.2 Black Box Exit Criteria

The Black Box Exit Criteria tells us what must be finished to move out of the Black Box phase. To successfully exit this phase, we need a 100% success rate. Here's what needs to be done when exiting the Black Box stage:

- The application should not produce any results when tested with random or nonsensical input.
- If a show time is entered that already exists in the database, an error message should be displayed.
- When the server is under high stress, its response time should not slow down.
- Any bugs found in the code should be fixed as much as possible.

4.1.2 White Box Phase

The White Box criteria are used to examine the internal structure of the program and find any errors within it. This helps in categorizing defects and evaluating the overall quality of the software.

4.1.2.1 White Box Entry Criteria

Upon entering the White Box stage, the focus was on ensuring that individual major features functioned correctly, even if they weren't tested together. The design and user interface were deemed stable. Here's what occurred during the entry into the White Box stage:

- Unit tests were generated for as many functions as possible.
- The White Box Testing Methods to be utilized were determined from the outset, primarily concentrating on unit testing and detecting memory leaks.

4.1.2.2 White Box Exit Criteria

Upon completion of the White Box stage, the Student Auditorium Management Software demonstrated stable performance. White Box testing persisted until meeting the criteria for Black Box testing or reaching the next project milestone. Exiting the White Box phase

necessitated achieving a 100% success rate. Here's the status of the product upon exiting the White Box Stage:

- All functions, including ticket booking, cancellation, salesperson management, balance sheet viewing, and show management, were implemented, operational, and rigorously tested.
- Test cases were meticulously crafted based on Control Flow diagrams for all functions.
- The graphical interface underwent thorough review and was deemed satisfactory and stable. No further alterations to dialog boxes or other interface elements were planned.
- Minor changes were permissible but required coordination with Development and Test Engineers.

4.2 Integration Testing

Integration Testing will involve integrating two modules: the Graphic User Interface module and the Controller (back-end). These modules contain a combination of stubs, drivers, and fully functional code.

4.2.1 Integration Test Entry Criteria

Both modules, the Graphic User Interface, and the Controller (back-end) are ready for integration.

Each module contains a mix of stubs, drivers, and fully functional code.

Test cases for integration are prepared and ready for execution.

4.2.2 Integration Test Exit Criteria

The integration testing phase is considered complete when all test cases have been executed and passed successfully.

Any issues or defects found during integration testing are resolved, and retesting is performed to ensure their resolution.

Integration test reports are generated, documenting the results and any issues encountered during testing.

The integrated system is deemed stable and ready for further testing phases or deployment.

4.3 System Testing

The System Test criteria are used to classify defects and evaluate the overall quality of the product. This phase involves combining and testing all components of the Controller and Graphical User Interface together as a unified system. System testing examines various aspects including functionality, performance, reliability, installation, behavior under special conditions, and stress testing.

4.4 Shipping or Live Release

The testing for the Controller and server was simplified by merging all testing stages into two phases: Function Complete and Regression testing. This approach adheres to the release criteria.

4.4.1 Shipping/Live Release Criteria

To move into the final stages, we need to make sure of two things:

We've identified all issues with the product, whether they're fixed, documented, postponed, or dealt with in some way.

We've finished and double-checked regression testing on all issues and the entire product.

5.Deliverables

- Source Code of Software

- Functional Specifications of Software:
  These outline the detailed description of the functions and capabilities of the software, including user interactions, input/output behaviors, and system responses.

- Test Plan Document:
  This comprehensive document outlines the objectives, criteria, standards, schedule, assignments, and tools to be used for testing the software. It serves as a roadmap for the testing process.

- Unit Testing Plan:
  This plan details the approach for testing individual units or components of the

software.

- Integration Plan:
  This plan outlines the strategy for combining individual units or components of the software into larger modules or systems, ensuring they interact and function together seamlessly.

- System Testing Plan:
  This plan describes the methodology for testing the integrated system as a whole, ensuring it meets the specified requirements and functions correctly in its intended environment.

6.Environmental Needs

The software, developed using Django, is compatible with Windows, MacOS and Linux platforms. It relies on MySQL for database connectivity, ensuring its proper functioning across operating systems.