# Table Of Content

# 

[**Table Of Content 6**](#_y04l5rd95wao)

[**Existing system 7**](#_yhlv1arz5m0p)

[Movie Ticketing System 7](#_rfyroj88dklq)

[Functions 7](#_rql46gv1k38g)

[**Problems of existing system 8**](#_vt2x6pn74bgc)

[**Software Quality attributes of the project. 9**](#_k53fcemkhoj4)

[**Software Process Model 11**](#_41rdnqudhuyu)

[**Project Plan and Schedule 15**](#_rj1jxucxgnmx)

[**Software Requirements Specification 16**](#_ruiuwz3h4esb)

[Functional Requirements 16](#_6797uga4m1jp)

[Non-Functional Requirements 18](#_ds6lag57wtxl)

[**System Architectural Design 19**](#_65sw5gcmb5qh)

[Client-Server Model (Monolithic Architecture) 19](#_pninkcph0a2m)

[**UI Design & good user interface design principles 22**](#_4mg86cipip0z)

[1. Consistency 22](#_jpccbynkkwif)

[2. Simplicity 24](#_26tzf38wlq5i)

[3. Recoverability 25](#_z83hg5arcnqz)

[4. User Guidance 27](#_pusip9wzljdq)

[5. Familiarity 28](#_khd8ir84hbri)

[**Test Cases 30**](#_hxjcnddq3wvw)

[**Software Maintenance 58**](#_pt4wj9d3x8p2)

[1. Corrective Maintenance 58](#_vfig6c6x622k)

[2. Adaptive Maintenance 58](#_kjd0mvg9bpez)

[3. Perfective Maintenance 59](#_bi9ifba2nhj7)

[**Reference 61**](#_hqej9qyvhbj)

[**Appendix 63**](#_cn9emid0qw87)

[First Screen : Log In / Sign Up Page 63](#_p8koa2n7zms2)

[Seat Selection Page 64](#_sb7p68c8658b)

[Review Summary/Payment Page 65](#_hhu369g8hfym)

[Movie Listing Page 67](#_kmtvbxpys4by)

[Member Page 68](#_7b6tsis0ym70)

# 

# Existing system

## Movie Ticketing System

* A digital platform that allows customers to access the services of a business, reserve seats and buy tickets. This system provides details such as what time a movie will be played, what seats are available, movie previews and more.

## Functions

1. **Ticket selection**.

Users can buy the ticket online and also check for the tickets availability online. So, they can buy tickets efficiently without queuing up at the counter. Users can select the cinema’s location, play time and date.

1. **Payments and Confirmation**.

It is refundable. Accept online payment and credit card payment but without cash payment. The system allows collecting points. When making the payment, the system will send a secure pin by sms to confirm the payment. After payment, the system will send buyers a confirmed ticket details with a ticket in qr code type.

1. **Seat selection**.

Enable users to pick seats within the cinema hall. Users can experience different seats such as 2D, 3D, 4DX, Dolby Atmos. Our cinema has twin seats, single seat, wheelchair seat and premium seat. Users can check for the seat availability online.

1. **Cancel Booking.**

Buyers can cancel the booking one day before the reserved date. Buyers might get a 50% payment refund in the buyer's bank account. They should fill up the cancellation form with an appropriate reason for cancellation. They must fill up their bank account details to get the refund. Our system will explain the cancellation rules during the booking process

# Problems of existing system

1. **Old poor system performance**

Processing users' requests and payment transactions take a long time. For instance, when there are a large number of customers using the system at the same time, the current system will be slowed down and cause the processing time to be longer than it originally takes that makes the customer fade up.

1. **Current user interface is hard to understand and used by the users.**

The design is too complex. That page is hard to navigate, lacked structure, and made finding what you were looking for almost impossible. The form to fill in personal details isn’t user-friendly, the submit button is not highly visible and there are no error messages displayed when users encounter errors.

1. **Weak current system’ security**

There is a risk of leakage of user personal information . The current system relies on simple password authentication without enforcing strong password policies or additional security layers like multi-factor authentication (MFA). This makes it easier for attackers to gain unauthorized access through common attacks such as password guessing.

1. **Cross-Platform Voucher Integration Issue**

Users get vouchers or coupons from the other platform but could not apply it to the existing system. They have to ask for help from the online customer service or go to a physical store to get help.

# 

# 

# Software Quality attributes of the project.

1. **Acceptability**

The new system should have an easy-to-navigate interface that enhances the user's experience, making the process of booking tickets seamless and straightforward. For example, guiding the user through selecting a movie, choosing a showtime, picking seats, and completing the purchase. Progress indicators show users how far they are in the booking process and how many steps remain. Moreover, the user interface is clean, essential elements are highlighted and readable fonts are used so that it is easy to understand and use by users. Acceptability ensures that the system is embraced by users and meets their expectations for usability and ease of use. It is important because a user-friendly interface enhances user satisfaction, encourages repeated usage, reduces user errors, and ultimately contributes to the success of the application.

1. **Effectiveness**

The new system should quickly respond to user actions, with short processing times for pages and transactions, enhancing the overall user satisfaction. It must efficiently handle high-demand periods (when tickets for a popular movie are released), without facing system slowdown. Effectiveness ensures that the system performs optimally, responding quickly to user actions and handling high-demand periods without degradation in performance. Effectiveness is important as users expect fast response times and smooth interactions. Effective systems improve user experience, minimize frustration, and enhance overall productivity that leads to an increase of a company's economy.

1. **Dependability and Security**

The new system should perform what it should be performed in terms of reliability. For instance, the new system should support a large number of users without facing system shutdown. In addition, the new system should ensure user data, especially payment and personal information, is securely stored and transmitted. For example, the new system should verify user identities and ensure they can only access features relevant to their roles. Dependability and security ensure that the system operates reliably and protects user data from unauthorized access or breaches. It is important because users need assurance that their personal and financial information is safe. Dependable and secure systems build trust, maintain reputation, and comply with regulatory requirements. Thus, no leakage of confidential information will occur.

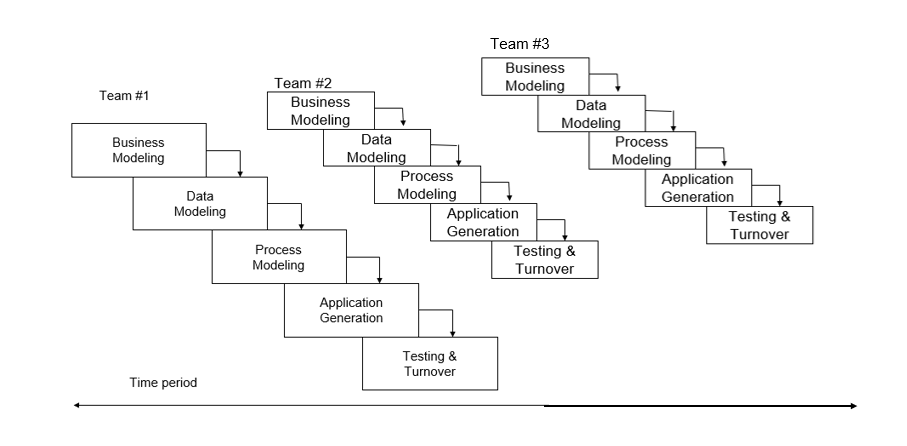
1. **Maintainability**

The new system should be easy to maintain. For example, modularity is fundamental to ensuring that the system is easy to understand and modify. By splitting the system into independent modules, developers can more easily locate and solve problems without digging into the entire system. Secondly, clear and readable code is key to maintainability. Well-named variables, clear comments, and consistent coding style make the code easy to read and understand, speeding up the maintenance process.​ Maintainability ensures that the system can be easily modified, updated, and repaired over its lifecycle. Maintainability is important because the software evolves over time to accommodate new features, fix bugs, and adapt to changing requirements. Maintainable systems reduce development costs, facilitate collaboration among developers, and extend the lifespan of the application.

1. **Functional**

The new system should perform the functions that it is required to. For example, handling users' actions (buying tickets, canceling booking etc.) efficiently. Moreover, new functions such as redeeming vouchers from other platforms should exist to improve users’ satisfaction. Functional correctness ensures that the system performs its intended tasks accurately and efficiently. Functional is important because users rely on software to accomplish specific tasks or goals. Functional systems meet user needs, fulfill business requirements, and deliver value to stakeholders.

# Software Process Model



**Recommendation: Rapid Application Development Model (RAD):**

RAD is an adaptive software development model based on prototyping and quick feedback with less emphasis on specific planning. It is a type of incremental process model in which there is an extremely short development cycle. In general, the RAD approach prioritizes development and building a prototype, rather than planning. Several teams complete different modules, then integrate it as a whole. For instance, different teams are in charge of different modules, and after carrying out a thorough testing, all the modules are integrated to produce a functional system.

**Justification:**

1. **The application is modularized.** The proposed movie ticketing system is broken down from a big task into a few small subtasks (modules). For instance, different teams work in parallel to complete their own modules and integrate them as a whole over a time period. By doing it this way, the company can save development time and allow for faster delivery of the system.
2. **Sufficient human resources .** As the company has sufficient human resources, the developers can be divided into different teams to complete different modules parallelly. This allows the new system to be produced rapidly and solve the current existing system problems. This helps in improving customer satisfaction and the company's economy.
3. **The proposed system is considered a small project.** As the size of the proposed system is small and required to be produced in a short duration to solve the existing problems, RAD is the most suitable to be used for the new proposed system. The proposed system is not complex, only consists of the simple process of a general ticketing system.
4. **Might require the need of changing requirements.** As our company receives feedback constantly from customers about the problems of the system, the system may need to be changed or added new functionality to improve the customer satisfaction. Moreover, when there are changes in the business environment, RAD allows for changes to be integrated quickly and efficiently. With rapid application development, developers can quickly make multiple iterations and updates to the software without starting from scratch.
5. **Urgent to use the system.** RAD time taken is short and reliable. This process model helps our company to earn profit in a short time as the existing problems need to be solved quickly. Rapid application development makes sense for quick development which can change direction on a dime.

| **Phase** | **Team #1 (Front-End Developers Team)** | **Team #2 (Back-End Developers Team)** | **Team #3 (Database Administrators Team)** |
| --- | --- | --- | --- |
| **Business Modeling** | They have a basic understanding of the business requirements to ensure that the user interface they develop aligns with the overall objectives of the project. | They have to understand business requirements and translate them into technical specifications for the back-end systems. | They have to understand data requirements and define data models that align with business objectives. |
| **Data Modeling** | They need to understand the data structures and formats required by the back-end systems to properly display and manipulate data on the user interface. | They ensure that the database structure supports efficient data storage, retrieval, and manipulation. | They are responsible for designing and optimizing the database schema, tables, entities, and relationships. |
| **Process Modeling** | They contribute to the creation of wireframes, prototypes, and user flow diagrams that represent the user experience and interface design. | They define APIs, business rules, and data processing workflows to support the functionality of the application. | They contribute to process modeling by understanding data flows and dependencies within the system. |
| **Application Generation** | They translate design mockups and wireframes into interactive and responsive web interfaces that users interact with by using HTML, CSS, Javascript, etc. | Back-end developers write server-side code using programming languages such as Python, Java, or Node.js. | They work closely with back-end developers to integrate database functionality into the application. |
| **Testing & Turnover** | They ensure that the user interface functions correctly across different devices, browsers, and screen sizes. | Back-end developers participate in unit testing, integration testing, and system testing to verify the functionality, performance, and reliability of the back-end systems. | They may assist in testing activities by performing data validation, integrity checks, and backup and recovery tests. |

# Project Plan and Schedule

# Software Requirements Specification

## **Functional Requirements**

1. **Movie Listing Module**
   1. The system should display all available movies according to the dates with the time in each cinema.
   2. Each movie listing should include details such as title, genre, language, synopsis and type(3D,2D).
2. **Member Function Module**
   1. It contains password recovery that allows users to reset their passwords in case they forget them, through a secure process such as email verification or security questions.
   2. It enables account verification that implements measures to verify user accounts to reduce fraudulent activities, such as email, phone number verification or password.
   3. The system provides two-factor authentication. For example, some methods like SMS verification or authenticator apps.
   4. It also contains account privacy settings like allowing users to control the visibility of their profile information, payment method and booking history.
   5. The system should enable users to deactivate or temporarily suspend their accounts if needed, with an option to reactivate them later.
3. **Payment Processing Module**
4. This module contains payment gateway integration and support. It is the capability to process online payments through various payment gateway providers, credit cards, debit cards, e-wallets, and direct bank transfers.
5. It provides payment processing countdown time that is required to process a payment and update the payment status. It is a countdown timer on the payment process in ecommerce so the system can manage serious customer’s and cancel dummy quotations so the seat booking will give priority to first come first serve basis.
6. The system also has email notification to send email notifications to customers and merchants regarding the status of a payment.
7. Payment history is also available to show how customers have paid in their accounts over the length of your credit.
8. **Ticket Booking Module**
   1. Users should be able to book tickets for selected seats, date and time.
   2. The system should calculate the total price based on the number and type of tickets.
   3. After successful payment, users should receive a booking confirmation with details such as booking ID, movie details, seat numbers and showtime.
   4. Users should be able to cancel their booking tickets before the movie showtime.
9. **Seat Selection Module** 
   1. Users should be able to select their seats for a specific movie showtime.
   2. A user account can select one or more seats for a specific movie showtime.
   3. The system should display all available seats and booked seats.

## 

## **Non-Functional Requirements**

1. **Product Requirements:**
2. **Efficiency Requirements:**

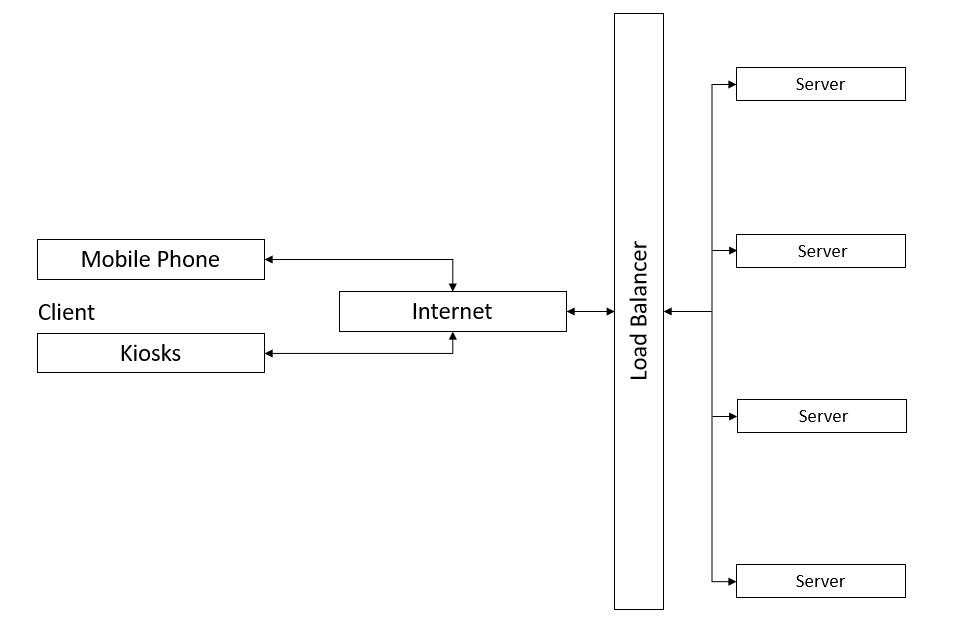
**I. Performance Requirements:** The system should be able to handle 3000 requests per second without any downtime. The system should only have at most 10ms latency during handling every request.

**II. Space Requirements:** The system should be able to store at least 10TB of important details such as customer information, payment information, etc.

1. **Dependability Requirements:** The system should support 5000 users at a time without facing system shutdown. For example, the system should provide accurate and reliable information regarding movie schedules, seat availability, and booking status.
2. **Security Requirements:** The system should be updated at all times to ensure all vulnerability of the system has been patched. For example, sensitive user information, such as payment details and personal data, should be encrypted to protect against unauthorized access or data breaches.
3. **Usability Requirements:** Every page’s design should be consistent. Shortcut button should be designed to ease customer booking. Data validation should be implemented to ensure the customer does not enter the wrong data. For example, Customer Name should not include numbers, Phone Number should be at least 10 characters and so on.
4. **Audit trail:** The system should have an audit trail to track and record important details of every transaction such as customer details, payment details, etc.

# System Architectural Design

## **Client-Server Model (Monolithic Architecture)**



* The client-server model is a network architecture where client devices, such as computers, smartphones, or IoT devices, interact with server computers to access resources, services, or data.

1. **Clients**: Users who want to book tickets for movies. They use mobile phones or kiosks at physical stores to browse available movies, select showtimes, select seats and purchase tickets.
2. **Internet**: The network infrastructure that connects clients to the system. It enables users to access service from anywhere with an internet connection, allowing them to browse movie listings, select seats, and complete transactions.
3. **Load Balancer**: The movie ticketing system uses load balancer to efficiently handle incoming traffic from users. The load balancer acts as a gateway, receiving requests from clients and distributing them across the available servers. For example, if multiple users are trying to book tickets at the same time, the load balancer ensures that their requests are evenly distributed among the four servers to maintain optimal performance and prevent any single server from becoming overwhelmed.
4. **Server**: The movie ticketing system runs on four servers, each hosting instances of the monolithic application responsible for handling various aspects of the ticket booking process. These servers work together to process user requests, manage movie listings and showtimes, handle seat selections, and facilitate payment transactions. Having multiple servers ensures reliability, allowing the system to handle increasing numbers of users and maintain high availability even during peak times, such as when popular movies are released.

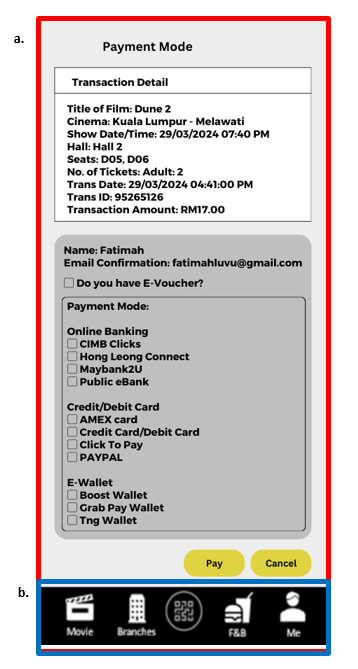
**Process:**

1. A user accesses the movie ticketing system through a client interface, such as a mobile phone to browse available movies, showtimes, and cinema locations, selecting the movie they want to watch and the showtime.
2. The client sends a request over the internet to the system's load balancer. This request includes information about the user's movie selection, showtime, and any other relevant details.
3. The load balancer receives the request and determines which server should handle it. It evenly distributes incoming requests among the four servers, ensuring efficient performance.
4. The selected server receives the user's request and processes it within the monolithic application running on that server. The server retrieves movie information from the database, checks seat availability for the selected showtime, and handles processes related to ticket pricing, promotions, or discounts.
5. Based on the user's request and the system's processing, the server generates a response, which includes details about available seats, pricing information, and any additional instructions for the user.
6. The server sends the response back to the client over the internet through the load balancer.
7. The client receives the response from the server and displays the relevant information to the user. If seats are available and the user decides to proceed with the ticket purchase, they can select their desired seats, enter payment information, and confirm the booking.
8. The client sends the ticket purchase request, including payment details, back to the load balancer. The load balancer sends the request to one of the servers for transaction processing.
9. The server securely handles the payment transaction, processing the user's payment details through a payment gateway or other payment processing service. Once the payment is confirmed, the server updates the database to reflect the purchased tickets, reserves the selected seats, and generates a confirmation for the user.
10. The server sends a confirmation response back to the client, confirming the successful purchase and providing the user with a ticket code. The client displays the booking confirmation to the user, completing the ticketing process.

## 

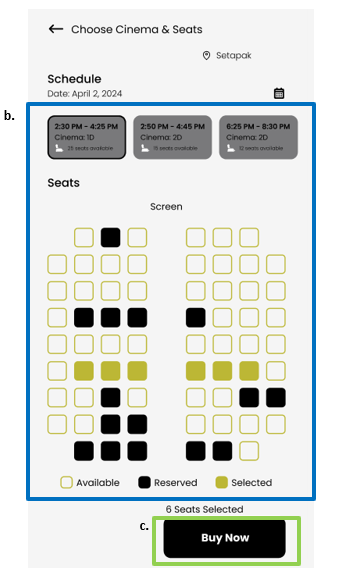
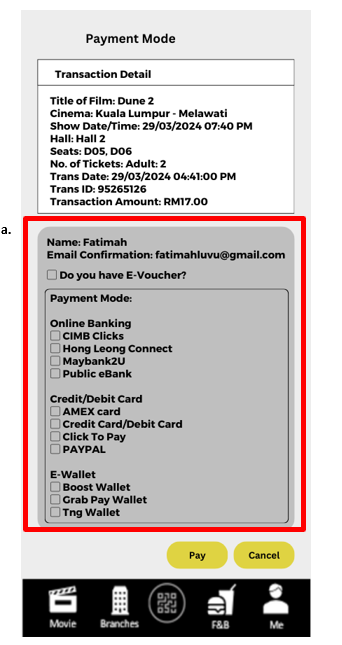
# UI Design & good user interface design principles

## Consistency

* Consistency is a fundamental principle in user interface (UI) design that emphasizes creating a uniform and predictable experience throughout a product or application. It involves maintaining coherence in various elements such as layout, visual design, interactions, terminology, and functionality. 

1. Consistent color schemes (Light Grey Background), typography, button styles, and layout across all screens to provide a coherent experience. Consistent color scheme and layout help establish a clear visual, guiding users' attention to important elements and content. This improves readability, making it easier for users to consume information.
2. Shortcut buttons are maintained at the bottom of every page to ease customer usage of the application. These buttons are designed to ease user interactions and improve efficiency by providing direct access to commonly used or important functionalities.

## Simplicity



* Simplicity in design refers to the concept of reducing complexity and focusing on clarity, ease of use, and minimalism to achieve a more effective and enjoyable user experience. It involves stripping away unnecessary elements and features to create interfaces that are intuitive, straightforward, and efficient.

1. All the available payment methods had been listed down so that users can choose it easily by seeing it straightforwardly without searching through a long dropdown list. This is important as users need to pay for their booking in a given duration. Thus, users can select their payment method easily in a short time.
2. The design is simple as users only need to select the date and time, and their preferred seats. This design is to ease users’ booking and improve customer satisfaction.
3. The ‘Buy Now’ button is designed big and clear so that customers can proceed to the payment page easily.

## Recoverability

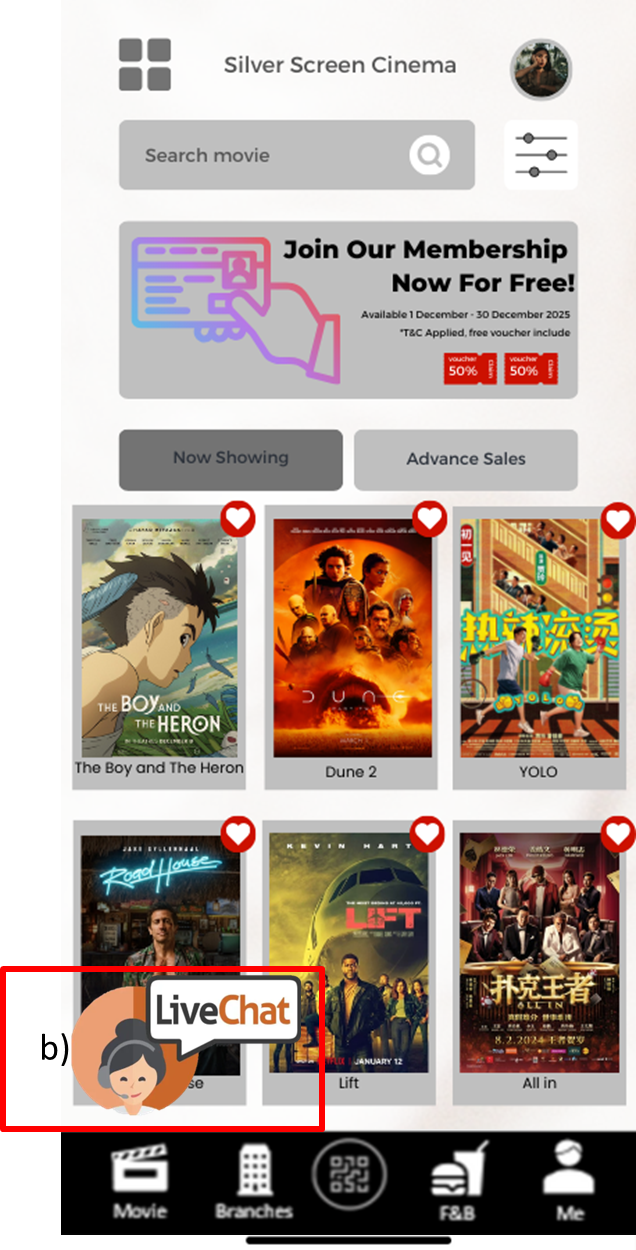
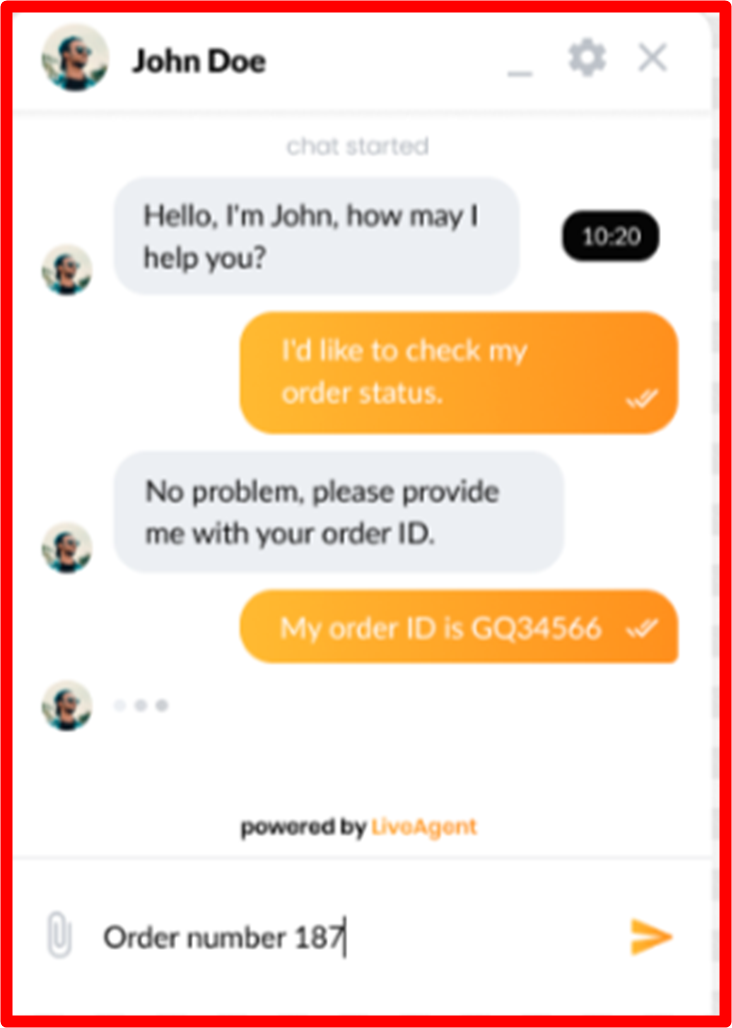
## 

* The principle of recoverability concerns the ability of users to reverse the effect of errors.

1. Users are allowed to cancel the payment if they realize that they had chosen the wrong movie/seats/date/time. This is to allow users to make any changes to their selections.

b & c. Users are allowed to go back to the previous page if they decide to change their selections. This is to allow users to make any changes to their selections.

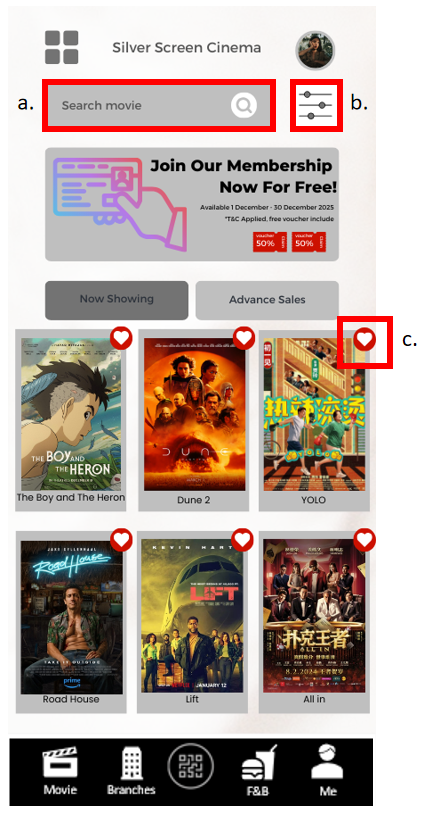
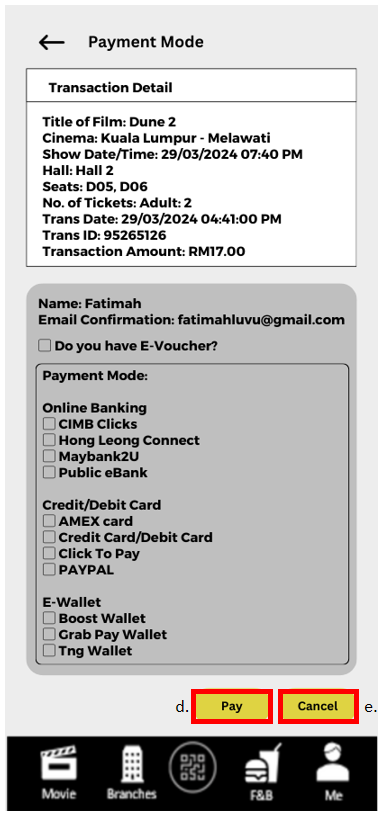
## User Guidance



* User guidance in UI (User Interface) design is essential for creating intuitive and efficient user experiences. Effective user guidance helps users understand how to interact with an application or website, ensuring they can achieve their goals with ease and satisfaction.

b. This function allows users to ask questions whatever they are confused about. A staff will be connected to the chat and will answer and guide them through the live chat.

## Familiarity

* Familiarity principle is an UI that contains Common UI Elements and controls. For example, use familiar buttons for actions such as "Search", "Select Seats", "Pay Now". This helps users recognize and understand the purpose of each element based on their prior experience. These user interfaces are similar to interfaces they are already familiar with, a desirable user interface should be intuitive, easy to navigate, and conform to the user's mental model of the system.

1. When a user performs a search, they typically have a clear idea of what they are looking for, it helps them quickly find it by using keywords. This search function is common across many different types of interfaces, including web browsers, operating systems, and software applications.
2. Filter function allows users to refine their search results and get the filtered range of data based on criteria you define. It quickly narrows down a large set of data to a smaller set that is more relevant to the user's needs. It can also help to improve the user’s efficiency and reduce the overall frustration of the results that users are finding.
3. The add to favourite function allows users to like or save the movie they like for future reference, it is similar to the shopping cart function. Their data will store in different pages in a list layout once they are saved. Users can reduce their time to navigate or research back for the information.
4. Pay function is a button that can lead users to external payment pages to complete the payment process. It also allows users to seamlessly and securely accept payments on the website with a small snippet.
5. Cancel function is a button that enables the user to quickly exit an operation which is payment mode without committing to any action. After clicking the cancel button, it will lead users back to the main page to fallback to safety. Cancel button could also mean canceling the current action that has not yet taken place before it has been reserved.

## 

# Test Cases

|  |  |
| --- | --- |
| Test Case ID : SSC-C011 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date:19/4/2024 |
| Module Name : Movie Listing Module | Test Executed By: |
| Test Title : Display Available Movies with details and showtimes | Test Execution Date: |
| Description :to check movies sorted by date,with details of each movie including title,genre,language,synopsis,and type along with the show timings in each cinema |  |

|  |
| --- |
| Pre-condition: User is logged into the system and navigated to the Movie Listing Page |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | To check movies display according to the dates | Movies date | Movies sorted by date in descending order | - | - |
| 2 | To check movies title display correctly | Movies title | Movie title presented clearly | - | - |
| 3 | To check movies genre display correctly | Movies genre | Movie genre  Presented clearly | - | - |
| 4 | To check movies language display correctly | Movies language | Movie language  Presented clearly | - | - |
| 5 | To check movies synopsis display correctly | Movies synopsis | Movie synopsis  Presented clearly | - | - |
| 6 | To check movies type display correctly | Movies type | Movie type  Presented clearly | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C021A | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 19/4/2024 |
| Module Name : Member Function Module | Test Executed By: |
| Test Title : Verify Password Recovery | Test Execution Date: |
| Description : To verify that users can reset their passwords securely through email verification or security questions. |  |

|  |
| --- |
| Pre-condition: User is registered in the system. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Press ‘Forgot Password’ Button | - | User should see options to reset password via email verification or security questions. | - | - |
| 2 | Choose the email verification option and initiate password reset. | [waikitlim0831@gmail.com](mailto:waikitlim0831@gmail.com) (valid email) | User should receive an email with instructions to reset the password. | - | - |
| 3 | Follow the instructions in the email to reset the password. | - | User should be able to reset the password successfully and log in with the new password. | - | - |

# 

|  |  |
| --- | --- |
| Test Case ID : SSC-C021B | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 19/4/2024 |
| Module Name : Member Function Module | Test Executed By: |
| Test Title : Verify Password Recovery | Test Execution Date: |
| Description : To verify that users can reset their passwords securely through email verification or security questions. |  |

|  |
| --- |
| Pre-condition: User is registered in the system. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Press ‘Forgot Password’ Button | - | User should see options to reset password via email verification or security questions. | - | - |
| 2 | Choose the email verification option and initiate password reset. | [waikitlim0821@gmail.com](mailto:waikitlim0821@gmail.com)  (invalid email) | Error message should be prompted, displaying ‘email/user not found’ | - | - |

# 

|  |  |
| --- | --- |
| Test Case ID : SSC-C022 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 19/4/2024 |
| Module Name : Member Function Module | Test Executed By: |
| Test Title : Verify Account Verification | Test Execution Date: |
| Description : To verify that user accounts are verified to reduce fraudulent activities, using methods such as email, phone number verification, or password. |  |

|  |
| --- |
| Pre-condition: User is registered in the system. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Log in to the user account. | [waikitlim0831@gmail.com](mailto:waikitlim0831@gmail.com) (valid email), 123456789a (valid password) | User should see a prompt for account verification if not already verified. | - | - |
| 2 | Choose the verification method (email, phone, password). | - | User should be able to complete the verification process successfully. | - | - |
| 3 | Verify that account is marked as verified in the system. | - | Account status should be updated to "verified". | - | - |

# 

# 

|  |  |
| --- | --- |
| Test Case ID : SSC-C023 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 19/4/2024 |
| Module Name : Member Function Module | Test Executed By: |
| Test Title : Verify Two-Factor Authentication | Test Execution Date: |
| Description : To verify that the system provides two-factor authentication using methods like SMS verification or authenticator apps. |  |

|  |
| --- |
| Pre-condition: User is registered and logged into the system. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to the two-factor authentication settings. | - | User should see options to enable two-factor authentication. | - | - |
| 2 | Choose the desired authentication method (SMS, authenticator app). | - | User should be able to successfully enable two-factor authentication. | - | - |
| 3 | Test the two-factor authentication process. | Test phone number: 01234567890 | User should receive a code via the SMS and successfully authenticate. | - | - |

# 

# 

|  |  |
| --- | --- |
| Test Case ID : SSC-C024 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 19/4/2024 |
| Module Name : Member Function Module | Test Executed By: |
| Test Title : Verify Account Privacy Settings | Test Execution Date: |
| Description : To verify that users can control the visibility of their profile information, payment method, and booking history. |  |

|  |
| --- |
| Pre-condition: User is logged into the system. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to the account privacy settings page. | - | User should see options to control visibility settings. | - | - |
| 2 | Adjust privacy settings for profile information, payment method, and booking history. | - | Settings should be updated accordingly. | - | - |
| 3 | Verify that selected settings reflect in the user profile. | - | Profile information, payment method, and booking history should only be visible as per the selected settings. | - | - |

# 

|  |  |
| --- | --- |
| Test Case ID : SSC-C025 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 19/4/2024 |
| Module Name : Member Function Module | Test Executed By: |
| Test Title : Verify Account Deactivation | Test Execution Date: |
| Description : To verify that users can deactivate or temporarily suspend their accounts, with an option to reactivate them later. |  |

|  |
| --- |
| Pre-condition: User is logged into the system. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to the account settings page. | - | User should see an option to deactivate or suspend the account. | - | - |
| 2 | Choose the option to deactivate the account. | - | User should receive a confirmation message for deactivation. | - | - |
| 3 | Attempt to log in with the deactivated account credentials. | - | User should not be able to log in, and a message indicating account deactivation should be displayed. | - | - |
| 4 | Reactivate the account using the reactivation option. | - | User should be able to log in successfully after reactivation. | - | - |

# 

# 

|  |  |
| --- | --- |
| Test Case ID : SSC-C031 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 19/4/2024 |
| Module Name : Payment Processing Module | Test Executed By: |
| Test Title : Support for Various Payment Methods | Test Execution Date: |
| Description : To verify that the payment processing module integrates with various payment gateway providers, credit cards, debit cards, e-wallet, and direct bank transfer. | |

|  |
| --- |
| Pre-condition: User has clicked the ‘Checkout & Pay’ button after reviewing the summary page to proceed to the payment mode page. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to the payment mode. | - | Users should see the options for payment methods. | - | - |
| 2 | Select a payment method. | Test payment method: Tng-Wallet | Users should see one and only one ticked box in the payment mode box. | - | - |
| 3 | Submit the payment through the corresponding payment page. | - | Jump to the corresponding page of the payment gateway or application to proceed to the next step of the payment process. | - | - |
| 4 | Verify that the payment is processed successfully. | - | Users should receive an email of the ticket’s qr code. | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C032 | Test Designed By: |
| Test Priority (Low/Medium/High) : Low | Test Designed Date: 19/4/2024 |
| Module Name : Payment Processing Module | Test Executed By: |
| Test Title : Payment Processing Countdown Timer | Test Execution Date: |
| Description : To verify that the payment processing countdown timer updates payment status within the specified time. | |

|  |
| --- |
| Pre-condition: User has selected at least one seat for booking. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to the payment review summary. | - | Users should see a countdown timer above the ticket booking summary. | - | - |
| 2 | Start the payment processing countdown timer. | - | The timer should countdown for 10 minutes, the review summary page will automatically disappear if there is no action for checkout and pay after 10 minutes. | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C033 | Test Designed By: |
| Test Priority (Low/Medium/High) : Medium | Test Designed Date: 19/4/2024 |
| Module Name : Payment Processing Module | Test Executed By: |
| Test Title : Email notification for payment status | Test Execution Date: |
| Description : To verify that email notifications are sent to users if the ticket booking is successful. | |

|  |
| --- |
| Pre-condition: The user has successfully paid for the ticket booking. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Payment Status | Payment status: FAIL | The users should return back to the main page of the merchant page. | - |  |
| 2 | Check the email inbox of the order | - | The users should receive an email notification and ticket’s qr code. | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C034 | Test Designed By: |
| Test Priority (Low/Medium/High) : Low | Test Designed Date: 19/4/2024 |
| Module Name : Payment Processing Module | Test Executed By: |
| Test Title : Payment History | Test Execution Date: |
| Description : To verify that payment history is available to show how customers have paid in their accounts over time. | |

|  |
| --- |
| Pre-condition: The user has successfully paid for the ticket booking. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to the ‘My Ticket’ page | - | Users should see the new ticket’s qr code and subsequent with all of the previous ones. | - | - |
| 2 | Verify that payment history records are displayed. | - | Users should see all the details about all the tickets and their qr codes. | - | - |
| 3 | Verify that the payment history records are accurate and up-to-date. | - | Users should see the accurate and up-to-date ticket booking history. | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C041 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 22/4/2024 |
| Module Name : Ticket Booking Module | Test Executed By: |
| Test Title : Successfully Book Tickets | Test Execution Date: |
| Description : to verify that a user can successfully book tickets for a movie by selecting a movie, showtime, available seats, ticket types, and quantities. | |

|  |
| --- |
| Pre-condition: User is already login |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to booking page | - | Bookings page load successfully | - | - |
| 2 | Select date | Date: 19/4/2024 | Date are displayed correctly | - | - |
| 2 | Select movie and showtime | Movie: “Titanic”, Time: 8.00p.m. | Movie and showtime details displayed correctly | - | - |
| 3 | Choose seats | Seats: A1,A2,A3 | Seats are successfully selected | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C042 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 22/4/2024 |
| Module Name : Ticket Booking Module | Test Executed By: |
| Test Title : Booking confirmation is successfully sent to the user. | Test Execution Date: |
| Description : to verify that the system can successfully booking confirmation after payment | |

|  |
| --- |
| Pre-condition: User is successfully pay the ticket |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Send booking ID to users | Booking ID: B1001 | Booking ID is successfully sent to users | - | - |
| 2 | Send movie details to users | Movie details: “Titanic” | Movie details is successfully  Sent to users | - | - |
| 3 | Send booked seats number to users | Booked Seats: A1,A2,A3 | Able to display the booked seats to users. | - | - |
| 4 | Send date and movie showtime to users | Date: 19/4/2024 Time: 8.00p.m. | Date and time are correct | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C043 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 22/4/2024 |
| Module Name : Ticket Booking Module | Test Executed By: |
| Test Title : Successfully cancel booking | Test Execution Date: |
| Description : to verify that the system can successfully cancel booking | |

|  |
| --- |
| Pre-condition: Cancellation must be done before the movie showtime |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Navigate to the user’s booking history | - | Booking history is displayed | - | - |
| 2 | Choose a booking to cancel | Booling ID: B1001 | Booking details are displayed | - | - |
| 3 | Cancel the booking | - | Booking is successfully canceled | - | - |
| 4 | Verify booking cancellation confirmation | - | Cancellation confirmation is received | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C051 | Test Designed By: |
| Test Priority (Low/Medium/High) : Medium | Test Designed Date: 22/4/24 |
| Module Name : Seat Selection Module | Test Executed By: |
| Test Title : Verity Seats Selected | Test Execution Date: |
| Description : To verify users are able to select seats for a specific movie showtime. | |

|  |  |
| --- | --- |
| Tsst Scenario | Users are successful in select seats for specific movie showtime. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Select for specific movie showtime | - | Show all the seats which available, reserved | - | - |
| 2 | Select the seats | - | Show the seats that chosen | - | - |

|  |  |
| --- | --- |
| Test Case ID : SSC-C052 | Test Designed By: |
| Test Priority (Low/Medium/High) : Medium | Test Designed Date: 22/4/24 |
| Module Name : Seat Selection Module | Test Executed By: |
| Test Title : Verity Seats Selection More Than One | Test Execution Date: |
| Description : To verify an user account are able to select more than one seat for a specific movie showtime. | |

|  |  |
| --- | --- |
| Tsst Scenario | Users are successful in selecting more than one seat for a specific movie showtime. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Select for specific movie showtime | - | Show all the seats for the specific movie showtime | - | - |
| 2 | Select the 3 seats | - | Show the 3 seats that chosen | - | - |
| 3 | Click to buy | - | Turn to the payment page | - | - |

# 

|  |  |
| --- | --- |
| Test Case ID : SSC-C053 | Test Designed By: |
| Test Priority (Low/Medium/High) : High | Test Designed Date: 22/4/24 |
| Module Name : Seat Selection Module | Test Executed By: |
| Test Title : Verity Available and Booked Seats | Test Execution Date: |
| Description : To verify users are able to see the seats that are available and booked for a specific movie showtime. | |

|  |  |
| --- | --- |
| Tsst Scenario | Users are successful in noticing the seats are available and booked for specific movie showtime. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step | Test Step | Test Data | Expected Results | Status | Comments |
| 1 | Select for specific movie showtime | - | Show all the seats which are available, booked | - | - |
| 2 | Select the available seats | - | Show the seats that chosen | - | - |
| 3 | Select the seats that are booked | - | Show the booked seats can’t be chosen | - | - |

# 

# 

# Software Maintenance

## Corrective Maintenance

* **The system will be keep an eye out to identify any problems, bugs,or unusual behavior** that could indicate problems.Set up alert systems and logging to detect and notify users of any issues occurring.When the systems logs a large number of failed transactions, users will able to notice a high volume of error notifications indicating that payment processing failures upon monitoring.
* **Address issues that require corrective maintenance** due to bug reports that users send in and analyze errors logs. For instance, to solve issues due to generating wrong ticket information, wrong calculation in total price or member points is not collected.
* **Fix errors as soon as possible** to prevent further disruptions to the system.To make sure that the impacted modules undergo sufficient regression testing in order to prevent new errors from occurring.In order to resolve errors, a retry mechanism will be designed through collaboration with related departments to address the errors.
* **After errors have been fixed, it will have extensive testing** to ensure that the system is stable and doesn't introduce new errors.Unit tests and integration tests will be run.If the errors affect system performance, performance tests also will be run. For example, staff will conduct extensive testing with simulated peak loads after fixing the errors to verify that the system can handle the requirements smoothly.

## Adaptive Maintenance

* **To compile software with a new environment (for use with different browsers or operating systems).** As the current system is only available for mobile phones and kiosks, adaptive maintenance can be carried out so that the system can be used through different client devices. This is to test and update the system to ensure compatibility with the latest versions of web browsers (Chrome, Firefox, Safari, Edge) and operating systems (Windows, macOS, iOS, Android).
* **To adapt to the changes in the external environment (changes in government laws and regulations).** Governments and regulatory bodies often introduce new laws that impact software systems. Adaptive maintenance ensures compliance with these changes to avoid legal issues. For example, Movie ticketing system need to comply with tax regulations (Sales and Service Tax (**SST**)). Adaptive maintenance might involve updating ticket prices to reflect changes in SST rates, ensuring accurate tax reporting in the system's backend to comply with Malaysian tax authorities and creating receipts and invoices that meet tax rules for customers who buy tickets online or at kiosks..
* **To implement security measures**, to enhance the security posture of the system and protect user data and transactions from cyber threats. For example, stay updated with the latest security patches and updates for all software components, including the operating system, web server, database server and any third-party libraries or frameworks used in the system.
* **To integrate and optimize new payment gateways involves adding or updating payment methods to meet standard and user preferences,** offering users more payment choices, enhances system flexibility and competitiveness and meets the payment needs of different user groups. This helps enhance user satisfaction by providing a smoother and more tailored payment experience.

## Perfective Maintenance

* **Users may see the need for new features or requirements** that they would like to see in the software to make it the best tool available for their needs. When the organization receives feedback from customers, new functionalities can be implemented based on the feedback to ensure high customer satisfaction.
* **To adjust software by adding new features as necessary and removing features.** For example, new features to ease ticket booking, notifications when popular movies are released and removing non commonly used features. This can help to improve customer satisfaction and prevent customer overwhelming due to lots of buttons or features.
* **To enhance the functionality like in-app purchases, location-based services for finding nearby cinema and rating for the specific movie that is showing now.** This improves the user experience of the mobile app version of the system. For example, it streamlines the checkout process to make it convenient and user-friendly, with options for storing payment information securely for future purchases. It also utilizes the GPS or geolocation technology to provide accurate cinema listings, along with directions and travel times from the user's location. Moreover, it provides a real-time movie rating by Integrate user-generated ratings and reviews, along with critic reviews, to give users insights into the quality and popularity of each movie.
* **To implement the seat selection interface and also add the features like seat recommendations based on user preferences or previous selections to make sure a better visualization for users to select the seat and effectively**. This ensures a seamless and intuitive booking experience across different screen sizes and resolutions. For example, it implements an algorithm that analyzes user preferences, such as seat location (e.g., front, middle, back), aisle access, or proximity to exits and ensures real-time updates of seat availability to make sure the recommendations are correct.

# Reference

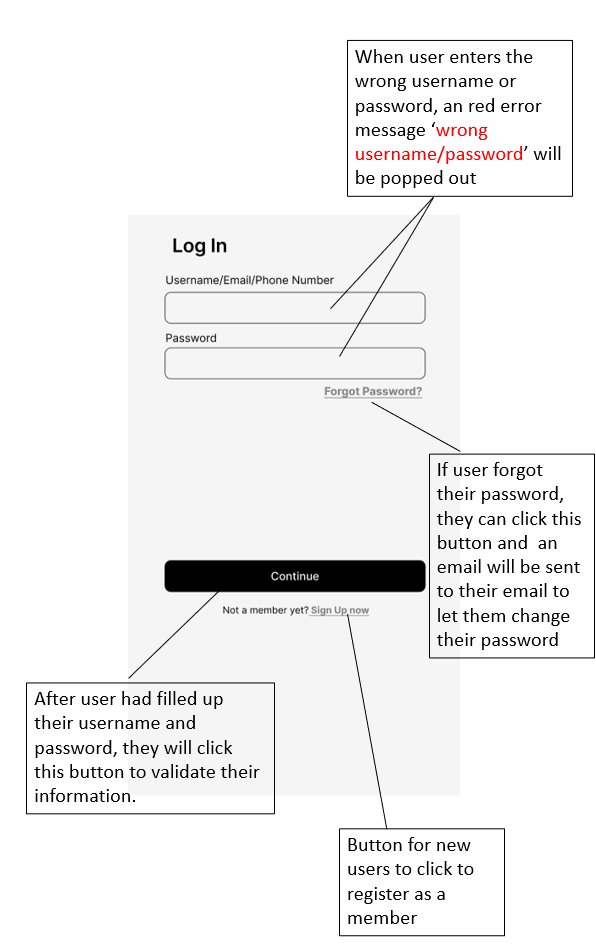
* Reference listAnon, (2018). Online movie ticket booking system | Process Street. [online] Available at:

<https://www.process.st/templates/online-movie-ticket-booking-system/#:~:text=The%20system%20retrieves%20a> [Accessed 26 Apr. 2024].

* Asana (2022). How to Write a Software Requirement Document (With Template) • Asana. [online] Asana. Available at: <https://asana.com/resources/software-requirement-document-template>.
* Codoid (2019). The Basics of Software Quality Attributes. [online] Codoid. Available at: <https://codoid.com/software-testing/the-basics-of-software-quality-attributes/>.
* filmgrail.com. (n.d.). Movie Ticket Software Essentials. [online] Available at: <https://filmgrail.com/blog/movie-ticket-software-essentials/> [Accessed 26 Apr. 2024].
* geeksforgeeks (2018). Software Engineering | Rapid application development model (RAD) - GeeksforGeeks. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/software-engineering-rapid-application-development-model-rad/>.
* GeeksforGeeks. (2018). Software Engineering | Software Maintenance - GeeksforGeeks. [online] Available at: <https://www.geeksforgeeks.org/software-engineering-software-maintenance/>.
* kissflow (2022). Rapid Application Development (RAD) | Definition, Steps & Full Guide. [online] kissflow.com. Available at: <https://kissflow.com/application-development/rad/rapid-application-development/>.
* Nechvolod, A. (2022). 12 software architecture quality attributes and their types. [online] Syndicode - Custom Software Development Company. Available at: <https://syndicode.com/blog/12-software-architecture-quality-attributes/>.
* Ozkaya, M. (2023). When to use Monolithic Architecture. [online] Design Microservices Architecture with Patterns & Principles. Available at: <https://medium.com/design-microservices-architecture-with-patterns/when-to-use-monolithic-architecture-57c0653e245e>.
* Quora. (2019). What are some major issues or problems with existing methods of managing information systems (IS) projects? [online] Available at: <https://www.quora.com/What-are-some-major-issues-or-problems-with-existing-methods-of-managing-information-systems-IS-projects> [Accessed 26 Apr. 2024].
* Thales (n.d.). The 4 Types of Software Maintenance-What is Software Maintenance. [online] cpl.thalesgroup.com. Available at: <https://cpl.thalesgroup.com/software-monetization/four-types-of-software-maintenance>.
* Trymata (2023). What is User Interface (UI) Design? Definition, Principles, Examples and Best Practices. [online] Trymata. Available at: <https://trymata.com/blog/2023/11/14/what-is-user-interface-ui-design/>.
* webnetpk.com. (n.d.). Cinema Management Solution | Webnet Pakistan. [online] Available at: <https://webnetpk.com/cinema-management-software> [Accessed 26 Apr. 2024].
* Wong, E. (2019). Principle of Consistency and Standards in User Interface Design. [online] The Interaction Design Foundation. Available at: [https://www.interaction-design.org/literature/article/principle-of-consistency-and-standards-in-user-interface-design.](https://www.interaction-design.org/literature/article/principle-of-consistency-and-standards-in-user-interface-design)

# Appendix

## First Screen : Log In / Sign Up Page



# 

# 

## Seat Selection Page

## 

## Review Summary/Payment Page

## 

## 

## 

## Movie Listing Page

# 

# 

# 

## Member Page

