### **User Story**

**Requirement-1:**

Implement Log In functionality.

| **Epic** | **User Story** | **Acceptance Criteria** |
| --- | --- | --- |
| As a **User, I Want** to be able to securely login to the system **so that** my information can only be accessed by me. | As a new **user**, **I want** to receive a mail from admin team username and temporary password **so that** I can get the login credentials. | Given that I am a new user, I need to get a mail to my valid organization mail. |
| As a **registered user**, **I want** to log in with my username and temporary password **so that** the system can authenticate me and I can trust it. | 1. Given that I am a registered user and logged out, if I go to the log in page and enter my username and password and click on Log in, then the data associated to my user should be accessible. 2. Given that I am a registered user and logged out, if I go to the log in page and enter my username but an incorrect password and click on Log in, then log in fails with an error message that specifies that the username or password was wrong. |

### **Functional Validations for Login**

1. Verify that cursor is focused on the “Username” text box on the page load (login page)
2. Verify that Enter/Tab key works as a substitute for the Sign-in button
3. Verify that the User is able to Login with Valid Credentials
4. Verify that the User is not able to Login with an invalid Username and invalid Password
5. Verify that the User is not able to Login with a Valid Username and invalid Password
6. Verify that the User is not able to log in with an invalid Username and Valid Password
7. Verify that the User is not able to log in with a blank Username or Password
8. Verify that the reset button clears the data from all the text boxes in the login form
9. Verify that the login credentials, mainly password stores in a database in an encrypted format
10. Verify that clicking on the browser back button after successful login should not take the User to log out mode
11. Verify that validation message is displayed in case of exceeding the character limit of the Username and Password fields
12. Verify that validation message is displayed in case of entering special character in the Username and password fields
13. Verify that the timeout of the login session (Session Timeout)
14. Verify that the logout link is redirected to login/home page
15. Verify that User is redirected to appropriate page after successful login
16. Verify that the User is redirected to the Forgot password page when clicking on the Forgot Password link
17. Verify that spaces should not be allowed before any password characters attempted

### **Security Test Cases for Login Page**

1. Verify that clicking on the browser back button after successful logout should not take the User to a logged-in mode
2. Verify that the password is in encrypted form (masked format) when entered in the password field.
3. Verify the password can be copy-pasted. System should not allow users to copy paste password.
4. Verify that encrypted characters in the “Password” field should not allow deciphering if copied
5. Verify whether the login form is revealing any security information by viewing the page source
6. Verify that the login page is vulnerable to SQL injection.

### **Performance Test Cases for Login Page**

1. Verify that how much time the application is taking to load the home page after entering the valid user name and password in the login page.

**Requirement-2:**

Add User.

| **Epic** | **User Story** | **Acceptance Criteria** |
| --- | --- | --- |
| As an admin, I want to be able to add a new user to the system so that they can book office seats. | As an admin, I want to be able to create a new user account with a Organization e-mail and temporary password, so that they can access the system and book office seats. | 1. Given that, the admin should be able to access the "Create New User" feature from the dashboard. 2. The admin should be able to enter a username and a temporary password for the new user. 3. The system should ensure that the password meets the minimum-security requirements, such as a minimum length or complexity |
| As an admin, I want to be able to enter the user's personal information such as their employee id so that we can contact them if necessary. | 1. The admin should be able to access the user's personal information section from the user account page. 2. The system should validate that the email address entered is in a valid format. |
|  | As an admin, I want to be able to assign the new user a role or permission level, so that they have the appropriate level of access to the system. | 1. The admin should have access to a user management system that allows them to create new user accounts and assign roles or permission levels to those accounts.   The user management system should have a clear and intuitive interface that allows the admin to easily navigate and assign roles or permission levels.  The admin should be able to assign roles or permission levels based on the user's job responsibilities or the level of access required for their position.  The roles or permission levels should be clearly defined and documented to ensure that the admin understands the level of access being granted to the new user.  The user's role or permission level should be visible in the user management system and should be easily updated if needed. |
|  | As an admin, I want to be able to send an email or notification to the new user, welcoming them to the system and providing instructions on how to log in and use it. | 1. The admin should have access to a messaging system that allows them to send emails or notifications to new users. 2. The messaging system should be integrated with the user management system, so that the admin can select the new user to send the message to. 3. The message should be clear and concise, welcoming the user to the system and providing instructions on how to log in and use it. 4. The message should include any relevant links or resources that the user may need to access the system, such as a login page or user guide. |
|  | As an admin, I want to be able to view and manage the list of users in the system, including the ability to edit the details. | 1. The admin should have access to a user management system that displays a list of all users in the system, including their name, username, role, and other relevant details. 2. The user management system should have a clear and intuitive interface that allows the admin to easily navigate and manage the list of users. 3. The user management system should allow the admin to search, filter, and sort the list of users based on various criteria such as name, username, role, and date of creation. 4. The user management system should provide appropriate error messages and warnings if the admin tries to edit a user account in a way that exceeds their level of authority or violates any system rules or policies. |

### **Functional Validations for Add User**

1. Verify that enter functionality is working properly or not.
2. Ensure that all required fields, such as name, email, and phone number, are filled in before allowing the user to submit the form.
3. Ensure that the email address provided by the user is unique and not already associated with an existing account.
4. Ensure that the user's chosen password meets any relevant password requirements, such as minimum length, complexity, or special characters.
5. Ensure that the user's role or permission level is properly assigned based on their job responsibilities or level of access needed.
6. Send a confirmation email to the user to confirm their account creation and provide instructions on how to access the seat booking app.
7. Implement restrictions on the number of accounts a user can create, to prevent spam or misuse of the system.
8. Ensure that all input provided by the user is validated, to prevent any malicious input from disrupting the system.

### **Security Test Cases for Add User**

1. Ensure that only authorized users with proper credentials are able to add new users to the system.
2. Ensure that users are assigned the appropriate role and level of access based on their job responsibilities, and that they cannot exceed their authorized level of access.
3. Ensure that password policies are enforced, such as minimum length, complexity, and expiration, to prevent weak passwords from being used.
4. Ensure that input provided by the user is validated, to prevent any malicious input from disrupting the system or introducing security vulnerabilities.
5. Ensure that all sensitive data, such as passwords and user information, are properly encrypted and stored securely.
6. Ensure that APIs used to add new users are properly secured, with appropriate access controls and validation of input parameters.

### **Performance Test Cases for Add User**

1. Measure the response time of the system when adding a new user, and ensure that it meets the acceptable performance criteria, such as under a certain threshold, typically measured in milliseconds.
2. Test the system's ability to handle large volumes of user additions by simulating multiple users adding new users simultaneously and ensuring that the system can handle the load without degrading performance.
3. Test the system's ability to handle high levels of user additions, such as during peak periods or special events, by simulating a large number of user additions and monitoring the system's performance under stress.
4. Monitor the system's resource utilization, such as CPU and memory usage, during user addition operations, and ensure that the system does not become overloaded or run out of resources.
5. Test the system's ability to scale up or down based on changes in user demand, such as by adding or removing servers or increasing capacity and ensure that performance remains consistent.
6. Test the system's ability to handle errors or exceptions that may occur during the user addition process, such as network timeouts or database errors, and ensure that performance is not significantly impacted.

**Requirement-3**

Verify Booking-

| **Epic** | **User Story** | **Acceptance Criteria** |
| --- | --- | --- |
| As a **Admin**, **I** **Want** to verify the seat booking **so that**  admin can confirm the details of employee. | As a **Admin**, **I want** an option to verify booking in dashboard **so** **that** I can verify the details. | The dashboard should have an option for the Admin to navigate to verify details page |
| As a **Admin**, **I want**  an option to scan the QR code/enter the booking id **so** **that** I can scan or enter the details to verify. | 1. The Admin should have an option to enter a booking ID or scan a QR code to retrieve the booking details. 2. If the Admin enters a booking ID, the system should display the corresponding booking details, including the user name, contact information, date and time of the event, seat location. |
|  | As a **Admin**, **I** **want** to change the status of the booking id along with timestamp and date **so** **that** I can have the updated list. | 1. The Admin should be able to mark a booking as verified or rejected based on the displayed details. 2. If a booking is marked as rejected, the reason for rejection should be specified and saved in the system. |

### **Functional Validations for verification of seat booking**

1. Verify that the system displays the correct booking details, including the user's name and contact information, date and time, seat location.
2. Verify that the booking details can be retrieved by entering the booking ID or scanning the QR code.
3. Verify that the system provides a way for the Admin to contact the user in case of any issues with the booking details.
4. Verify that appropriate security measures are in place to ensure that only authorized Admin users can access the booking details.
5. Verify that the seat reservation status is updated after scan/enter the booking id.

### **Security Test Cases for verification of seat booking**

1. Verify that admin can only perform actions they are authorized to perform by checking whether the system is enforcing role-based access control (RBAC) and whether admin can access only the data they are authorized to view.
2. Verify that the system is handling errors and exceptions correctly by checking whether the system is providing appropriate error messages to users and that these messages do not reveal sensitive information.

### **Performance Test Cases for verification of seat booking**

1. Verify that how much time is taking to scanning page/entering the booking id after clicking the scan button in dashboard.
2. Verify that how much time is taking to scan and getting the detail of seat booking .

**Requirement-4:**

Implement Admin user should be able to generate various reports on need basis.

functionality

| **Epic** | **User Story** | **Acceptance Criteria** |
| --- | --- | --- |
| As an **Admin, I Want** to be able to generate various reports on need basis **so that** booking details can only be accessed by me. | As an Admin, **I want** to generate various reports on Number of employees booked the seat with the details of Emp ID, Emp Name, Floor, Timing, Food req,designation. | 1. Given that I am an admin, when I go to the dashboard page I should be able see current day data like number of employee available and food count. 2. Given that I am a admin, I need and option for download. |
| As an Admin, **I want** to generate various reports on Weekly and Monthly report - Employee wise how many times have come to office | 1. Given that I am an admin, I should have an option manage employee seat booking data in the dashboard. 2. Given that I am a admin after choosing the option manage employee I should get complete employee list in organization 3. Given that I am a admin, I should be able to generate Weekly and Monthly report of employee till date and download. |

**Functional Validations for Admin generating reports**

1. Verify that admin dashboard is displayed when admin logged-in to application using admin credentials.
2. Verify that UI and all features navigation buttons are designed user-friendly
3. Verify that Enter key works as a substitute for the features button
4. Verify that the Admin is able to navigate to proper page. As mentioned in dashboard Ui
5. Verify that the Admin is able view the current days Number of employees booked with the details of Emp ID, Emp Name, Floor, Timing, Food req etc..
6. Verify that the Admin is able to generate and download the data on-click.
7. Verify that the Admin is able navigate again to dashboard from current page.
8. Verify that the Admin is able view choose option Weekly and Monthly and customized date of Employee wise how many times have come to office report

### **Security Test Cases for Admin generating reports**

1. Verify that the admin is required to provide valid credentials, such as a username and password, to access the system and generate the seat booking report. Ensure that unauthorized access is not possible.
2. Check that the admin has the necessary permissions to generate the seat booking report. Make sure that only authorized users can access the report and that access is restricted to the necessary data.
3. Verify that the seat booking report is not leaked or displayed to unauthorized users. Ensure that the report is not stored in an insecure location and that access is limited to authorized users only.
4. Test for proper error handling and validation of user inputs. Ensure that meaningful error messages are displayed to the user when input validation fails, and that system errors are logged and reported to the appropriate parties.

### **Performance Test Cases for Admin generating reports**

1. Test the system's ability to handle a large number of concurrent requests for generating and downloading seat booking reports. This can be done by simulating multiple users generating and downloading reports at the same time.
2. Test the time taken by the system to generate and download seat booking reports for a single user. This can be done by measuring the time taken from the moment the user initiates the report generation process until the report is downloaded

**Requirement-5:**

Implement Seat Booking functionality.

| **Epic** | **User Story** | **Acceptance Criteria** |
| --- | --- | --- |
| As a **User, I Want** to be able to book a seat. | As a User, **I want** to book my desired seat using the application. | User can access the seat booking features easily from their dashboard after login. |
| As a User **I want** to book my seat on daily or weekly basis. I can choose shift time and I have options to opt for lunch at office. | 1. User can view the availability of all seats in the office, including desks and other shared spaces. 2. User can see a clear and detailed map of the office space, including the location of each available seat. 3. User can book a seat for daily or weekly basis on custom dates. And should have an option to choose shift time like 9am to 6 pm or 6pm to 2pm etc. 4. User can have an option to opt for lunch at office Yes or No. |

### **Functional Validations for Seat booking**

1. The First step in functional validation is to test the user interface of the seat booking system. This involves checking that the interface is user-friendly, easy to navigate and provides clear instructions for users to book seat.
2. Verify that enter functionality is working properly or not.
3. Verify that Enter key works as a substitute for the Seat booking button.
4. Verify that clicking on the browser back button after successful seat booking should not take the User to log out.
5. Test the seat booking functionality which involves verifying that users are able to book seats, cancel seats and view availability in real time.
6. Testing notification and confirmation messages after seat booking.
7. Application should be able to handle errors gracefully and provide clear messages to users when something goes wrong.
8. Application should be able to handle the availability of seat like booked seat can not be access by another user.

### **Security Test Cases for Seat booking**

1. Verify that clicking on the browser back button after successful seat booking should not cancel the booked seat.
2. Verify that all actions taken in the seat booking feature are logged. Check that the logs are properly secured and not accessible by unauthorized personnel.
3. Verify that sensitive information such as employee id, social security number are not exposed in any way.
4. Verify that the system prevents multiple seats booking attempts within a short frame from same user.

### **Performance Test Cases for Seat booking**

1. Verify that how much time the application is taking to book the seat after entering the valid inputs.
2. Verify that the system can handle many users simultaneously trying to book seats.
3. Verify that the system can handle peak loads and still function properly.

**Requirement-6:**

Profile update **-**

| **Epic** | **User Story** | **Acceptance Criteria** |
| --- | --- | --- |
| As a **User, I Want** to modify profile. | As a user, I want to be able to modify my profile information easily so that I can keep account up-to-date and ensure that my personal information is accurate and relevant. | As a user I expect to easily access profile setting option from dashboard. |
| I want to be able to access my profile settings through a clear and intuitive interface, which allows me to view and edit my basic account information, such as email address, phone number and password.  In addition, I want to be able to upload or change my profile picture easily, and I want to be able to preview any changes that I make before they are saved to my account.  Overall, I want the profile modification process to be quick and easy, with clear information and instructions and minimal steps required | 1. As a user I want to be able to view and edit basic account information. 2. As a user I want to be able to change profile picture easily. 3. As a user I expect that information that I given is secured and protected with security measures. 4. As modification made to the profile are accurately reflected in account and saved without any errors or issues. |

### **Functional Validations for Profile Modification**

1. Verify that the user has the appropriate permissions to modify their profile.
2. Check that the user’s input is valid and follows any required format or character limitations.
3. Ensure that the changes made by the user do not cause any conflicts or inconsistencies with existing data in the system.

### **Security Test Cases for Profile Modification**

1. Authentication testing: - Verify that the user is properly authenticated before allowing access to the profile modification feature.
2. Authorization testing: - Verify only authorized employee can modify their own profiles. Attempt to access and modify the profile of another employee and ensure that it is not allowed.
3. Input validation testing: - Verify that input fields accept only expected and valid data.
4. Access control testing: - Verify that only the intended fields are accessible by the user for modification. Ensure that fields that are not supposed to modified are protected.
5. Data privacy testing: - Ensure that sensitive information such as employee ID are not exposed in any way. Ensure that such information is encrypted during transmission and storage.
6. Session management testing: - Verify that sessions expire after a certain amount of time or after logout.
7. HTTPS testing: - Verify that the application is only accessible via HTTPS protocol and that HTTPS is properly configured.
8. Multi-factor authentication: - Verify that the system uses multi-factor authentication for critical operations, such as password changes or profile modification.

### **Performance Test Cases for Profile Modification**

1. Verify that how much time the application is taking to load the profile page after updating the information in the profile page.

**Requirement-7**

Modify Seat Booking

| **Epic** | **User Story** | **Acceptance Criteria** |
| --- | --- | --- |
| As a **User, I Want** to be able to modify the seat booking details so that I can accommodate my change of plans and preferences. | As an existing **user**, **I want** to modify the shift timings for my seat booking **so that** the system can modify the shift timings as per my requirement. | Given that I am an existing user, logged in to the web app and have a seat booking, when I go to the modify bookings page, it should list my bookings and I should be able to select a booking to modify, then I should be able to modify the specifics of the booking. |
| As an existing **user**, **I want** to change the seat that I have booked **so that** the system can book a new seat for me and de-allocate the previous one. | Given that I am a registered user and logged in, if I go to the modify booking page and select a booking from the list of existing bookings, I should be able to modify one or more specifics from the following:   1. Shift Timings 2. Seat 3. Lunch preference   Given that I am a registered user and logged out, the web app displays a message to login and redirects me to the login page.  Given that I am a registered user and logged in, if I go to the modify bookings page and select a booking from the list of existing bookings, and select a seat that is already booked by someone, then selection fails with an error message that says “ booked seat in not available ”. |
| As an existing **user**, **I want** to change the preference of having lunch at office **so that** the system can account for my change of preference. |

### **Functional Validations for Modifying the Seat Booking**

1. Verify that the bookings drop down list is filled with the user’s booking in any.
2. Verify that the web app displays, “you have no bookings “, if the user does not have any seat bookings.
3. Verify that Enter key works as a substitute for the Sign-in button.
4. Verify that Tab key works as a substitute for the focussing the cursor to the different selectable.
5. Verify that Arrow up and down key works as a substitute for selecting the items from drop down lists using mouse pointer.
6. Verify that the User is not able to modify the seat booking if the user is not logged in.
7. Verify that the User is not able to modify the seat booking if the user is selecting a seat that is already booked by someone else.
8. Verify that none of the drop-down options are selected by default when the modify bookings page loads up.
9. Verify that the User can modify all the parameters that are allowed using the modify seat bookings page.
10. Verify that the User is not able to change booking specifics for past bookings.
11. Verify that the User is not able to modify the seat booking specifics without selecting the required specifics.
12. Verify that the reset button clears all the data from all the selections.
13. Verify that a message which confirms that the changes in booking detail specifics are done is displayed in the case the user is successfully able to modify the seat booking specifics.
14. Verify that the unacceptable/invalid changes cannot be done through an API call to the backend from outside of the web app.
15. Verify that validation message is displayed in the case when User leaves the required selections to empty values.
16. Verify that the modify seat bookings link is redirected to login/home page if the user is not logged in.

### **Security Test Cases for Modifying the Seat Booking**

1. Verify whether the modify seat bookings page is revealing any security information by viewing the page source.
2. Verify that the modify seat bookings page is not vulnerable to SQL injection, through which sensitive data can be leaked, or invalid API calls to modify seat bookings do not actually modify the seat bookings

**Performance Test Cases for Modifying the Seat Booking**

1. Verify that how much time the web application is taking to load the modify seat bookings page when the user is logged in.
2. Verify that how much time the web application is taking to confirm that the modifications in the seat bookings are done successfully.
3. Verify that how much time the web application is taking to display the message asking the user to sign in and redirecting the user to login page, when the user is not logged in.