Detailed Manual Tests
LNU Scheduler

Authentication and Authorization

Test Case 1: Verify Login and Register Button Visibility

Objective: This test case verifies if an unregistered user can see the Login and Register buttons on the landing page.

Procedure:

1. Open a web browser and navigate to the application's landing page.
2. Verify that the user is not automatically logged in (e.g., check for a username or profile picture not being present).

Checklist:

1. Login Button: A button labeled "Login" or similar is present on the landing page.
2. Register Button: A button labeled "Register" or similar is present on the landing page.

Test Case 2: Access Registration Functionality

Objective: This test case verifies that an unregistered user can access the registration functionality.

Procedure:

1. Open a web browser and navigate to the application's landing page.
2. Verify the presence of a "Register" button.
3. Click on the "Register" button.

Checklist:

1. The application redirects the user to a dedicated registration page.

Test Case 3: Successfully Create a New Account

Objective: This test case verifies that a user can successfully create a new account through the registration process.

Procedure:

1. Open a web browser and navigate to the application's landing page.
2. Click on the "Register" button.
3. Fill in the registration form with valid data.
4. Submit the registration form.

Checklist:

1. The application creates a new user account.
2. The application redirects the user to a confirmation page or logs them in automatically.

Test Case 4: Registration Form Fields

Objective: This test case verifies that the registration page includes the necessary fields for user information.

Procedure:

1. Access the registration page.

Checklist:

1. The registration form includes a field for "First Name."
2. The registration form includes a field for "Last Name."
3. The registration form includes a field for "E-mail."
4. The registration form includes a field for "Password."

Test Case 5: Registration Process Usability

Objective: This test case verifies the user-friendliness of the registration process.

Procedure:

1. Access the registration page.

Checklist:

1. The registration form layout is clear and easy to understand.
2. Labels for each field are descriptive and guide the user on what information to provide.

3. The registration process has a clear flow and guides the user towards completion.

Test Case 6: Login with Valid Credentials

Objective: This test case verifies that a user can successfully log in with the credentials created during registration.

Procedure:

1. Navigate to the application's landing page.
2. Verify the user is not logged in.
3. Click on the "Login" button.

Checklist:

1. The application redirects the user to the login page.
2. The user can enter the registered email address in the designated field.
3. The user can enter the registered password in the designated field.
4. Clicking the "Log in" button successfully logs the user into the application.
5. The application redirects the user to a relevant page after login (e.g., dashboard, profile).

Test Case 7: Login Page Functionality

Objective: This test case verifies the presence of essential elements on the login page.

Procedure:

1. Navigate to the application's landing page.
2. Click on the "Login" button.

Checklist:

1. The login page includes a field for "E-mail" or similar (e.g., username).
2. The login page includes a field for "Password."
3. The login page offers a "Remember me" option to remember login credentials.

4. The login page provides a "Sign up" button or link for new user registration.
5. The login page offers a "Forgot password" functionality to recover lost credentials.

Test Case 8: Login Feedback

Objective: This test case verifies that the user receives clear feedback about the login attempt.

Procedure:

1. Navigate to the application's landing page.
2. Click on the "Login" button.

Checklist:

1. When a user enters invalid credentials (e.g., incorrect email or password) and attempts to log in, the application displays an error message indicating the login failure.
2. The error message is clear and informative, guiding the user on how to resolve the issue (e.g., suggesting a password reset).
3. Upon successful login, the application provides a confirmation message or redirects the user without any error messages.

Test Case 9: Login Error with Sign Up Prompt

Objective: This test case verifies that the user is prompted to sign up after encountering a login error.

Procedure:

1. Navigate to the application's landing page.
2. Click on the "Login" button.
3. Enter invalid credentials (e.g., incorrect email or password).
4. Submit the login form.

Checklist:

1. The application displays an error message indicating the login failure.

2. The error message includes a clear call to action or link for signing up for a new account.

Test Case 10: Remember Me Functionality

Objective: This test case verifies that the "Remember me" option allows users to remain logged in after leaving the website.

Procedure:

1. Navigate to the application's landing page.
2. Click on the "Login" button.
3. Enter valid credentials.
4. Select the "Remember me" checkbox.
5. Submit the login form.
6. Close the web browser and website completely.
7. Re-open the web browser and navigate to the application again.

Checklist:

1. The application automatically logs the user in without requiring them to re-enter credentials

Test Case 11: Sign Up Button Functionality

Objective: This test case verifies that clicking the "Sign up" button on the login page redirects the user to the registration page.

Procedure:

1. Navigate to the application's landing page.
2. Click on the "Login" button.
3. Click the "Sign up" button

Checklist:

1. The login page displays a "Sign up" button or link.
2. Clicking the "Sign up" button redirects the user to a dedicated registration page.

Test Case 12.1: Forgot Password Functionality

Objective: This test case verifies that the "Forgot password" button initiates the password recovery process.

Procedure:

1. Navigate to the application's landing page.
2. Click on the "Login" button.
3. Locate the "Forgot password" button or link.

Checklist:

1. The "Forgot password" button or link is visible on the login page.
2. Clicking the "Forgot password" button prompts the user to enter their registered email address or username.

Test Case 12.2: Forgot Password Functionality

Objective: This additional step verifies that the user receives an email with password reset instructions.

Procedure:

1. Follow steps 1-3 from Test Case 12.1.
2. Enter a valid registered email address.
3. Submit the form to initiate password recovery.

Checklist:

1. An email with password reset instructions is sent to the provided email address.

Test Case 13: User Logout Functionality

Objective: This test case verifies that a registered user can log out of the application.

Procedure:

1. Log in to the application using valid credentials
2. After successful login, a "Logout" button or link is displayed.

3. Click the "Logout".

Checklist:

1. The application redirects the user to the login page or landing page.
2. The user is not authorized anymore

User Settings

Test Case 14: Access Settings Page

Objective: This test case verifies that a registered user can access the settings page to manage their account preferences.

Procedure:

1. Log in to the application using valid credentials.

Checklist:

1. The application interface provides a clear way to access user settings (e.g., profile menu, settings button).
2. Clicking the designated element successfully redirects the user to the settings page.

Test Case 15: Notification Settings Accessibility

Objective: This test case verifies that the settings page allows users to modify notification settings.

Procedure:

1. Log in to the application using valid credentials.
2. Click on the Settings button.
3. Observe the opened page.

Checklist:

1. The settings page includes a dedicated section for managing notification preferences.

Test Case 16: Notification Option Availability

Objective: This test case verifies that users can choose their preferred notification methods on the settings page.

Procedure:

1. Follow steps 1 & 2 from Test Case 13.
2. Locate the notification settings section.
3. Log in to the application using valid credentials.
4. Click on the Settings button.
5. Observe the opened page.

Checklist:

1. The notification settings section provides options to enable or disable notifications for different communication methods (e.g., email, phone number).
2. Each notification channel has a clear label indicating the notification method (e.g., "Email Notifications," "Phone Number Notifications").

Test Case 17: Access Change Password Functionality

Objective: This test case verifies that a registered user can access the functionality to change their password on the settings page.

Procedure:

1. Log in to the application using valid credentials.
2. Click on the Settings button.
3. Observe the opened page.
4. Locate the notification settings section.

Checklist:

1. The settings page includes a section or option for changing the user's password.

Test Case 18: Change Password with Old Password

Objective: This test case verifies that a user can change their password by entering their old password on the settings page.

Procedure:

1. Log in to the application using valid credentials.
2. Click on the Settings button.

3. Observe the opened page.
4. Locate the password change functionality.

Checklist:

1. The password change functionality requires the user to enter their current password for verification.
2. Upon successful password change, the application displays a confirmation message or redirects the user to the updated settings page.

Test Case 19: Access Change Email Functionality

Objective: This test case verifies that a registered user can access the functionality to change their email address on the settings page.

Procedure:

1. Log in to the application using valid credentials.
2. Click on the Settings button.
3. Observe the opened page.

Checklist:

1. The settings page includes a section or option for changing the user's email address.
2. The email change functionality requires the user to enter their current password for verification.
3. Upon successful email change, the application displays a confirmation message or redirects the user to the updated settings page.

Test Case 20: Access Event Template Creation

Objective: This test case verifies that a registered user can access the functionality to create event templates on the settings page.

Procedure:

1. Log in to the application using valid credentials.
2. Click on the Settings button.
3. Observe the opened page.

Checklist:

1. The settings page includes a section or option for creating event templates.

Test Case 21: Select Event Templates

Objective: This test case verifies that a registered user can access and select existing event templates on the settings page.

Procedure:

1. Log in to the application using valid credentials.
2. Click on the Settings button.
3. Observe the opened page.

Checklist:

1. The user can view a list of their existing event templates within the event template management section.
2. Each event template has a clear identifier or name for easy selection.
3. The user interface allows the user to select a desired event template (e.g., checkboxes, radio buttons).

Test Case 22: Account Deletion Functionality

Objective: This test case verifies that a registered user can initiate account deletion through the settings page.

Procedure:

1. Log in to the application using valid credentials.
2. Click on the Settings button.
3. Observe the opened page.

Checklist:

1. The settings page includes a section or option for account deletion.
2. Clicking the account deletion option presents a clear confirmation message or dialog box to the user.

3. The confirmation message clearly explains the consequences of account deletion (e.g., data loss, and profile removal).
4. Upon confirmation of account deletion, the application logs the user out and redirects them to the landing page or a relevant confirmation page.

Note: Additional testing might be required to verify account deletion irreversibility, data deletion processes, and potential restrictions on account deletion.

Schedule Management

Test Case 23: View Existing Schedules on the Main Page

Objective: This test case verifies that a registered user can see a list of their existing schedules on the main application page.

Procedure:

1. Log in to the application using valid credentials.

Checklist:

1. The main application page displays a section or list containing the user's existing schedules.
2. Each schedule entry provides a clear and concise way to identify the schedule (e.g., name, description, date).

Test Case 24: Edit Schedule Name

Objective: This test case verifies that a user can modify the name of their existing schedules directly on the main page.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to modify on the main page.

Checklist:

1. The user interface offers a way to edit the name of the selected schedule directly on the main page (e.g., click-to-edit functionality, inline edit mode).
2. Upon saving the changes, the updated schedule name is reflected in the list.

Test Case 25: Delete Schedule from Main Page

Objective: This test case verifies that a registered user can remove existing schedules from the main application page.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to delete on the main page.

Checklist:

1. The user interface provides a clear way to delete the selected schedule (e.g., delete button, confirmation prompt).
2. Clicking the delete option prompts the user for confirmation to avoid accidental deletion.
3. Confirming deletion successfully removes the schedule from the list on the main page.

Test Case 26: Schedule Delete Confirmation

Objective: This test case verifies that a user receives a confirmation popup before deleting a schedule.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to delete on the main page.
3. Initiate the deletion process using the designated option.

Checklist:

1. A confirmation popup appears before permanently deleting the schedule.
2. The confirmation popup clearly displays the name of the schedule to be deleted (optional, but recommended for clarity).
3. The confirmation popup offers two distinct buttons: "Delete" and "Cancel."
4. Clicking "Cancel" closes the popup and leaves the schedule unchanged.

Test Case 27: Create a New Schedule on the Main Page

Objective: This test case verifies that a registered user can create new schedules directly on the main application page.

Procedure:

1. Log in to the application using valid credentials.

Checklist:

1. The main application page provides a clear way to initiate the creation of a new schedule (e.g., "Create New Schedule" button, "+" button).
2. Clicking the designated element opens a dedicated interface or form for creating a new schedule.

Event Management

Test Case 28: Access Schedule Details

Objective: This test case verifies that a registered user can access the details of a created schedule on the schedule page.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to access on the main page.

Checklist:

1. The application allows users to navigate to individual schedule pages.
2. The schedule page displays the name of the selected schedule.

Test Case 29: View Events in Schedule

Objective: This test case verifies that the schedule page displays existing events within a selected schedule.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to access on the main page.

Checklist:

1. The schedule page includes a section or list that displays the events associated with the chosen schedule.
2. Each event entry provides a clear and concise way to identify the event (e.g., name, date, time).

Test Case 30: Access Event Creation on Schedule Page

Objective: This test case verifies that a registered user can add new events to a schedule while on the schedule page.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to access on the main page.

Checklist:

1. The schedule page offers a clear way to initiate the creation of a new event within that specific schedule (e.g., "Add Event" button, "+" button).
2. Clicking the designated element opens a dedicated interface or form for creating a new event within the chosen schedule.

Test Case 31: Event Creation Form

Objective: This test case verifies that the event creation page includes essential fields for defining event details.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to access on the main page.
3. Locate a button to create a new event.

Checklist:

1. The event creation form includes a field for entering the event date.
2. The event creation form includes a field for entering the event time.
3. The event creation form includes a field for specifying the event duration.
4. The event creation form includes a field for entering the event location (optional).
5. The event creation form includes a field for providing a description of the event.
6. The event creation form includes a field for adding relevant links to the event (optional).
7. The event creation form includes a checkbox or option for setting event notifications (optional, but functionality should be clear).

Test Case 32: Event Creation Guidance

Objective: This test case verifies that the event creation process offers clear instructions or tooltips for each field.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule you want to access on the main page.
3. Locate a button to create a new event.

Checklist:

1. Each event creation field offers informative text or tooltips that explain the purpose and expected input format (e.g., date format, time selection method).

Test Case 33: Event Notifications

Objective: This test case verifies that users receive notifications for events with the "Notify me" checkbox selected.

Procedure:

1. Log in to the application using valid credentials.
2. Create a new schedule (optional).
3. Access the newly created schedule.
4. Initiate the creation of a new event within the schedule.
5. Set a future date and time for the event.
6. Select the "Notify me" checkbox during event creation.
7. Save or submit the new event.

Checklist:

1. The user receives a notification via their chosen method (e.g., email, pop-up) closer to the event time.

Test Case 34: Access Event Modification

Objective: This test case verifies that a user can access the functionality to modify an existing event.

Procedure:

1. Log in to the application using valid credentials.
2. Access the schedule containing the event you want to modify.

Checklist:

1. The schedule page offers a clear way to edit or modify existing events within the schedule (e.g., edit button, pencil icon).
2. Clicking the designated element opens an interface or form for modifying the details of the selected event.

Test Case 35: Modify Event Details

Objective: This test case verifies that a user can modify the details of an existing event.

Procedure:

1. Log in to the application using valid credentials.
2. Access the schedule containing the event you want to modify.

Checklist:

1. The event modification form allows editing previously entered information for the event (e.g., date, time, description).
2. Upon saving the changes, the updated event details are reflected on the schedule page.

Test Case 36: Delete Event from Schedule

Objective: This test case verifies that a user can delete events from a schedule on the schedule page.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule with an event that you want to delete on the main page.
3. Access the event you want to delete.

Checklist:

1. The schedule page offers a clear way to delete existing events (e.g., delete button, trash can icon).
2. Clicking the designated element initiates the deletion process for the selected event.

Test Case 37: Event Delete Confirmation

Objective: This test case verifies that a user receives a confirmation popup before deleting an event.

Procedure:

1. Log in to the application using valid credentials.
2. Locate a schedule with an event that you want to delete on the main page.
3. Access the event you want to delete.
4. Initiate the deletion process using the designated option.

Checklist:

1. A confirmation popup appears before permanently deleting the event.
2. The confirmation popup clearly displays the name or details of the event to be deleted (optional, but recommended for clarity).
3. The confirmation popup offers two distinct buttons: "Delete" and "Cancel."
4. Clicking "Cancel" closes the popup and leaves the event unchanged.

Non-Functional

Test Case 38: Application Loading Speed

Objective: This test case verifies that the application loads within an acceptable time frame.

Procedure:

1. Launch the application using a representative device and network connection.
2. Measure the time it takes for the application to become fully usable (e.g., time to interactive).

Checklist:

1. The application loading time meets the target benchmark (ideally less than 1 second).

Note: Performance testing tools can be used to automate this process and measure loading times across different devices and network conditions.

Test Case 39: Application Usability After Load

Objective: This test case verifies that the application remains usable after loading.

Procedure:

1. Launch the application using a representative device and network connection.
2. Measure the time it takes for the application to become fully usable (e.g., time to interactive).

Checklist:

1. After loading, the application displays content and elements correctly.
2. Users can interact with the application features without encountering performance issues (e.g., slow response times, lag).

Test Case 40: System Scalability

Objective: This test case verifies that the system can handle a high number of concurrent users.

Procedure:

1. Simulate a scenario with increasing numbers of concurrent users accessing the application (e.g., using load testing tools).

Checklist:

1. The system maintains acceptable performance levels (e.g., response times, stability) up to 500 concurrent users.

Note: Scalability testing often requires specialized tools and expertise.

Test Case 41: System Response Time

Objective: This test case verifies that the system responds to user actions within an acceptable timeframe.

Procedure:

1. Launch the application and perform typical user actions (e.g., login, navigation, data entry).
2. Measure the time it takes for the system to respond to each action (e.g., page load times, form submissions).

Checklist:

1. The system's response time for user actions meets the target benchmark (ideally less than 100 milliseconds).

Note: Performance testing tools can be used to automate this process and measure response times across different functionalities.

Test Case 42: System Error Handling

Objective: This test case verifies that the system gracefully handles errors and unexpected situations.

Procedure:

1. Launch the application and perform actions that might trigger errors (e.g., entering invalid data, performing unsupported actions).

Checklist:

1. The system displays informative error messages that guide users towards resolving the issue.
2. The application avoids crashing or becoming unresponsive due to errors.
3. Users can continue using the application's functionalities after encountering and resolving errors (if applicable).

Test Case 43: System Recovery After Errors

Objective: This test case verifies that the system can recover from errors without requiring a full restart.

Procedure:

1. Launch the application and perform actions that might trigger errors (e.g., entering invalid data, performing unsupported actions).
2. Attempt to recover from the error using appropriate actions (e.g., correcting input, retrying the action).

Checklist:

1. The system allows users to recover from errors and continue using the application without a complete shutdown.
2. Data entered before the error is preserved whenever possible.

Test Case 44: System Uptime Monitoring

Objective: This test case verifies that the system maintains a high uptime percentage.

Procedure:

1. Set up system monitoring tools to track uptime and downtime periods.

Checklist:

1. The system uptime remains at or above 90% over a defined monitoring period.

Note: Uptime monitoring is typically an ongoing process that requires dedicated tools and configurations.

Test Case 45: System Downtime Recovery

Objective: This test case verifies that the system can be restored after a critical failure within an acceptable timeframe.

Procedure:

1. Simulate a critical system failure scenario (if possible within your testing environment).
2. Measure the time it takes for the system administrators to identify, diagnose, and recover from the failure.

Checklist:

1. The system downtime after a critical failure does not exceed 6 hours.

Test Case 46: Average Downtime Recovery

Objective: This test case verifies that the system can be restored from failures within an average timeframe.

Procedure:

1. Simulate a series of non-critical system failures (if possible within your testing environment) and track the recovery times.

Checklist:

1. The average downtime for recovering from failures falls within the target range of 2 to 6 hours.

Note: Simulating and testing system failures often requires specialized environments and expertise.

Test Case 47: Dynamic User Interface Updates

Objective: This test case verifies that the user interface reflects changes made by the user in real-time.

Procedure:

1. Launch the application and perform actions that modify data or settings (e.g., editing text fields, selecting options).

Checklist:

1. The user interface updates automatically to reflect the changes made by the user without requiring additional actions (e.g., page refresh).
2. The updated information is visually clear and easy for the user to identify.

Test Case 48: Real-time Feedback on Actions

Objective: This test case verifies that users receive immediate feedback on their actions.

Procedure:

1. Launch the application and perform actions that might require feedback (e.g., saving data, submitting forms).

Checklist:

1. The application provides clear indications of successful or unsuccessful actions through visual cues or messages (e.g., success messages, error messages, loading indicators).
2. The feedback helps users understand the outcome of their actions and guides them further if necessary.

Test Case 49: Intuitive User Interface

Objective: This test case verifies that the user interface is easy to understand and navigate without requiring additional instructions.

Procedure:

1. Recruit testers with varying levels of technical experience (if possible).
2. Observe how users interact with the application for the first time.

Checklist:

1. The layout and organization of the interface are clear and logical.
2. Users can find the functionalities they need without difficulty.
3. Users can complete tasks successfully without requiring tutorials or help guides (within reason).

Test Case 50: Error Prevention Through Interface Design

Objective: This test case verifies that the user interface design minimizes the possibility of user errors.

Procedure:

1. Launch the application and analyze the design elements for data entry, selection, and actions.

Checklist:

1. The interface uses clear labels and instructions to guide user input.
2. The design discourages users from making invalid selections or entering incorrect data (e.g., dropdown menus, input validation).
3. The application provides clear prompts or warnings before users perform irreversible actions.

Test Case 51: Browser Compatibility

Objective: This test case verifies that the application functions correctly across different web browsers.

Procedure:

1. Launch the application using a variety of popular web browsers (e.g., Google Chrome, Microsoft Edge, Opera, Mozilla Firefox, Safari).

Checklist:

1. The application layout, functionalities, and content render correctly across all chosen browsers.
2. Users can interact with the application features and perform actions without encountering errors or compatibility issues.

Test Case 52: Responsive Design

Objective: This test case verifies that the application uses responsive design to adapt to different screen sizes.

Procedure:

1. Launch the application on a variety of devices with different screen sizes (e.g., desktops, laptops, tablets, smartphones).

Checklist:

1. The application layout adjusts automatically to fit the screen size of the device being used.
2. Content remains readable and interactive across all tested devices.
3. Users can access features and functionalities without encountering layout issues.

Test Case 53: Usability on Different Devices

Objective: This test case verifies that the application remains usable across various devices with different screen sizes and input methods.

Procedure:

1. Launch the application on a variety of devices with different screen sizes (e.g., desktops, laptops, tablets, smartphones).

Checklist:

1. Users can perform typical tasks using the appropriate input methods for each device (e.g., mouse, touch screen).
2. Buttons, menus, and other interactive elements are sized appropriately for easy interaction on all devices.

Test Case 54: Testability

Objective: This test case verifies that the application is designed and implemented to facilitate efficient testing.

Procedure:

1. Analyze the application's code structure, documentation, and logging mechanisms.

Checklist:

1. The application code is well-structured, modular, and documented, allowing for easier test case development and maintenance.
2. The application provides logging or tracing mechanisms to aid in debugging and identifying issues during testing.
3. Test data can be easily generated or manipulated to support various testing scenarios.

Test Case 55: Continuous Integration Implementation

Objective: This test case verifies that the development process leverages continuous integration (CI) for efficient deployment.

Procedure:

1. Investigate the development environment and tools used for building and deploying the application.

Checklist:

1. The development team utilizes a CI pipeline that automates code building, testing, and deployment processes.
2. The CI pipeline integrates seamlessly with version control systems to trigger deployments upon code changes.
3. The CI process minimizes downtime during deployments by performing them rapidly and efficiently.

Test Case 56: System Documentation

Objective: This test case verifies that the system is accompanied by proper documentation for maintenance and troubleshooting purposes.

Procedure:

1. Review the available system documentation (e.g., user manuals, technical guides, administration manuals).

Checklist:

1. The documentation is comprehensive and up-to-date, covering system functionalities, configuration options, and troubleshooting procedures.
2. The documentation is clear, concise, and easy to understand for system administrators with varying levels of technical expertise.

Test Case 57: User Authorization and Access Control

Objective: This test case verifies that the system enforces user authorization and access control mechanisms.

Procedure:

1. Create user accounts with different permission levels within the system.
2. Attempt to access functionalities and modify data using accounts with varying permissions.

Checklist:

1. The system restricts unauthorized access to sensitive functionalities and data.
2. Users can only perform actions and access information based on their assigned permissions.
3. The system employs a secure authentication mechanism to verify user identities before granting access.

Test Case 58: Minimum Password Length

Objective: This test case verifies that the system enforces a minimum password length for user accounts.

Procedure:

1. Attempt to create a new user account with a password that does not meet the minimum length requirement.

Checklist:

1. The system prevents users from creating passwords that are shorter than the defined minimum length.
2. The system enforces password length requirements during account creation and password changes.

Test Case 59: Password Complexity Requirements

Objective: This test case verifies that the system enforces password complexity requirements.

Procedure:

1. Attempt to create a new user account with a password that does not meet the complexity requirements (e.g., lacking digits or letters, if applicable in your testing environment).

Checklist:

1. The system enforces password complexity requirements that include a combination of character types (e.g., uppercase letters, lowercase letters, numbers, symbols).
2. The system prevents users from creating passwords that do not adhere to the defined complexity criteria.

Test Case 60: User Permission Management

Objective: This test case verifies that system administrators maintain control over user access permissions.

Procedure:

1. Create user accounts with different permission levels within the system.

Checklist:

1. System administrators have exclusive control over assigning, modifying, and revoking user permissions.
2. Regular users cannot modify their own permissions or the permissions of other users.

Test Case 61: Permission Change Audit Trail

Objective: This test case verifies that the system tracks changes made to user access permissions.

Procedure:

1. Log in as a system administrator and modify user permissions within the system.

Checklist:

1. The system logs all modifications made to user permissions, including details about who made the change, what permissions were modified, and when the change occurred.
2. Audit trails provide a record of permission changes for security and troubleshooting purposes.

Test Case 62: Encrypted Communication

Objective: This test case verifies that communication between the server and clients is encrypted.

Procedure:

1. Analyze the application traffic using network monitoring tools.

Checklist:

1. Communication between the server and clients utilizes a secure protocol (e.g., HTTPS) that encrypts data in transit.
2. Sensitive data, such as passwords and user information, is protected from eavesdropping and man-in-the-middle attacks during transmission.