Name-Ritesh Shende

202202378@vupune.ac.in

9022524700

Code:

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class LoginTest {

    public static void main(String[] args) {
        // Set the path to the ChromeDriver executable
        System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");

        // Initialize WebDriver
        WebDriver driver = new ChromeDriver();

        // Navigate to the application URL
        driver.get("https://www.conqt.com");

        // Find the username and password fields and login button
        WebElement usernameField = driver.findElement(By.id("username"));
        WebElement passwordField = driver.findElement(By.id("password"));
```

```
        WebElement loginButton = driver.findElement(By.id("login-button"));

        // Enter valid username and password
        usernameField.sendKeys("your_username");
        passwordField.sendKeys("your_password");

        // Click on the login button
        loginButton.click();

        // Verify successful login (assertions or checks based on your application behavior)
        // Example: Assert.assertEquals(driver.getCurrentUrl(), "expected_url_after_login");

        // Close the browser
        driver.quit();
    }
```

OUTPUT:

Login Successful

```
Login successful. Redirected to the dashboard.
```

Login Failed

```
Login failed or redirected to unexpected page.
```

Explanation:

1. Setting WebDriver and ChromeDriver Path:
    - Ensure that you have downloaded the ChromeDriver executable and set the path correctly using System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");.
2. Navigating to Application URL:
    - Use driver.get("https://www.conqt.com"); to navigate to the login page of the application.
3. Locating Elements:
    - Find the username and password fields using driver.findElement(By.id("username")); and driver.findElement(By.id("password"));.
4. Entering Credentials:
    - Use .sendKeys("your_username"); and .sendKeys("your_password"); to enter the username and password into their respective fields.
5. Clicking on Login Button:
    - Use loginButton.click(); to simulate clicking the login button.
6. Verifying Successful Login:
    - After clicking login, wait for a brief period (Thread.sleep(2000);) to allow the login process to complete. This is essential in real-world scenarios where redirects or asynchronous operations might occur.
    - Verify the login success by checking the current URL (driver.getCurrentUrl();). Adjust the expected URL (https://www.conqt.com/dashboard) according to your application's actual behavior.

Test cases

Test Case ID	Test Case Description	Test Steps	Expected Results	Pass Criteria	Fail Criteria
TC-001	Login Functionality	1. Navigate to the login page. 2. Enter valid username and password. 3. Click on the "Login" button.	User should be logged in successfully. User should be redirected to the dashboard/home page. User data and features are displayed correctly.	User logs in without errors. User lands on the correct page/dashboard. Correct user data and features are displayed.	Login fails with valid credentials. User is not redirected to the expected page. Incorrect data or features displayed after login.
TC-002	Invalid Login	1. Navigate to the login page. 2. Enter invalid username and/or password. 3. Click on the "Login" button.	User should not be logged in. User should stay on the login page. Error message should be displayed.	User cannot log in with invalid credentials. User remains on the login page. Error message is shown for invalid input.	User logs in with invalid credentials. User is redirected to unexpected page. No error message displayed for invalid input.
TC-003	Navigation Between Pages	1. Navigate to different pages/modules. 2. Verify navigation links/buttons. 3. Click on navigation links/buttons.	User should be able to navigate between pages/modules smoothly.	Navigation links/buttons work correctly. User can move between pages/modules without errors.	Navigation links/buttons do not work. User cannot move between pages/modules as expected.
TC-004	Data Input Validation	1. Enter data into input fields (e.g., forms). 2. Submit the form or move focus away from the input field.	Input data should be validated according to specified rules (e.g., required fields, format).	Input fields accept valid data. Error messages are shown for invalid data.	Input fields accept invalid data. No validation occurs for required fields/format.

Test Case ID	Test Case Description	Test Steps	Expected Results	Pass Criteria	Fail Criteria
TC-005	Performance - Response Time	1. Perform actions that require server responses (e.g., loading a page, submitting a form).	Response time should be within acceptable limits (e.g., < 5 seconds).	Actions complete within expected response time. Server response is fast and consistent.	Actions take longer than expected. Server response time is inconsistent or exceeds acceptable limits.
TC-006	Security - Authentication	1. Attempt to log in with unauthorized credentials. 2. Access restricted areas without proper authentication.	Unauthorized access should be denied. Proper error messages should be displayed.	Unauthorized access is denied. Error messages indicate unauthorized attempt.	Unauthorized access is granted. No error message displayed for unauthorized attempt.
TC-007	Compatibility - Browser Compatibility	1. Test application functionality on different web browsers (e.g., Chrome, Firefox, Safari).	Application should work correctly on each tested browser.	Application functions as expected on all tested browsers. UI layout is consistent across browsers.	Application does not function as expected on one or more browsers. UI layout is inconsistent on certain browsers.
TC-008	UI Layout and Design	1. Navigate through different screens. 2. Check alignment and visibility of UI elements.	UI elements should be properly aligned and visible. UI should be aesthetically pleasing.	UI elements are aligned correctly. UI is visually appealing and intuitive.	UI elements are misaligned. UI is cluttered or visually unappealing.
TC-009	Usability - Ease of Use	1. Perform common tasks (e.g., search, filter, update profile). 2. Evaluate ease of task completion.	Tasks should be intuitive and easy to complete. User should not encounter confusion or frustration.	Tasks are completed without difficulty. User finds the system easy to navigate and use.	Tasks are difficult to complete. User encounters confusion or frustration during task execution.

Test Case ID	Test Case Description	Test Steps	Expected Results	Pass Criteria	Fail Criteria
TC-010	Error Handling	1. Intentionally trigger errors (e.g., invalid input, network timeout).	Error messages should clearly indicate the issue and suggest corrective actions.	Error messages are informative and actionable. User can recover from errors easily.	Error messages are unclear or misleading. User cannot understand how to resolve the issue.
TC-011	Mobile Responsiveness	1. Access the application using mobile devices (e.g., smartphones, tablets).	Application should display correctly and function properly on mobile devices.	UI elements should be responsive and adjust to different screen sizes. User experience on mobile is seamless.	UI elements are not responsive. User experience on mobile devices is poor or inconsistent.
TC-012	Offline Functionality	1. Disable internet connectivity. 2. Attempt to access and use the application.	Application should provide appropriate feedback or offline functionality (if applicable).	User should be informed of the lack of internet connectivity. Offline features should be available and functional.	User can access features normally despite internet disconnection. No offline functionality is provided.
TC-013	Data Integrity	1. Perform operations that involve data creation, update, and deletion.	Data should be accurate and consistent after performing operations.	Data changes are reflected correctly in the system. No data loss or corruption occurs.	Data inconsistencies or inaccuracies occur. Data loss or corruption is observed after operations.
TC-014	Integration with Third-Party Services	1. Test integration with external APIs or services (if applicable).	Data exchange and functionality should work seamlessly with third-party services.	Integration with third-party services is successful. Data is exchanged correctly and in real-time.	Integration fails with third-party services. Data exchange is incomplete or inconsistent with external APIs.
TC-015	Accessibility - Screen Readers	1. Use screen reader tools (e.g., NVDA, VoiceOver) to	Application should be accessible to	Screen reader tools can navigate through UI elements. UI	Screen reader tools cannot interpret UI

Test Case ID	Test Case Description	Test Steps	Expected Results	Pass Criteria	Fail Criteria
		navigate and interact with the application.	users with disabilities. Screen reader tools should interpret UI elements correctly.	is labeled and structured for accessibility.	elements. UI is not properly labeled or structured for accessibility.

In conclusion, creating effective software test cases is crucial for ensuring the quality and reliability of applications like "conqt technology" or any other software system. Test cases help verify that all functionalities work as expected, user interactions are smooth, and the application performs well under various conditions.

Key points to consider when designing test cases include:

1. **Clarity and Specificity:** Each test case should be clearly defined with specific steps, expected outcomes, and criteria for passing or failing.
2. **Coverage:** Test cases should cover different aspects such as functionality, usability, performance, security, compatibility, and more, depending on the application's requirements.
3. **Relevance:** Test cases should be relevant to the user's perspective and business requirements, ensuring that tested functionalities meet user expectations.
4. **Automation:** Where possible, automate repetitive test cases to improve efficiency and reliability in testing processes.
5. **Adaptability:** Test cases should be adaptable to changing requirements and updates in the software, ensuring continuous testing and validation.