Worldline Testing Task – 2

Dhineshwaran C
WDGET2024090

Github Link: https://github.com/Dhineshwaran-C/WorldlineTask2.git

Introduction:

This documentation outlines a Selenium WebDriver automation script written in Java for testing login functionality on a web application. It utilizes the Cucumber framework for behavior-driven development (BDD) style testing. The script simulates user actions such as opening a browser, navigating to a webpage, logging in with valid and invalid credentials, and verifying login success or failure.

Code Explanation:

open_chrome_and_open_application() Method:

This method sets up the Chrome WebDriver by specifying the path to the Chrome driver executable and initializing a new instance of the WebDriver.It maximizes the browser window to ensure consistent testing environments.Then, it navigates to the specified URL of the web application under test. This action occurs when the feature file calls the 'Given User is on Home Page' step.

```
@Given("User is on Home Page")
public void open_chrome_and_open_application() {
    System.setProperty("webdriver.chrome.driver","G:\\worldline\\chromedriver-win64\\chromedriver-win64\\chromedriver.exe");
    driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://practice.expandtesting.com/");
}
```

Login_page() Method:

It initiates navigation to the login page by locating the login button on the home page and clicking it. After the click, it captures the current URL and verifies if it matches the expected URL of the login page. This ensures the correctness of the redirection. This action occurs when the feature file calls the ' When User navigate to Login Page' step.

```
@When("User navigate to Login Page")
public void login_page() {
    WebElement login = driver.findElement(By.xpath("(//a[normalize-space()='Login Form'])[1]"));
    login.click();
    String expectedURL = "https://practice.expandtesting.com/login";
    String actualURL = driver.getCurrentUrl();
    Assert.assertEquals(expectedURL,actualURL);
}
```

Enter_username_and_password() Method:

It is responsible for entering the provided username and password into their respective input fields on the login form. Locates the username and password input fields using their name attributes and populates them with the provided credentials. This action occurs when the feature file calls the ' Then User enters "<username>" and "<password>"’ step, with <username> and <password> being taken from the examples provided at the bottom of the feature file.

```java
@Then("User enters {string} and {string}")
public void enter_username_and_password(String usernames,String passwords) {
    WebElement user = driver.findElement(By.name("username"));
    user.sendKeys(usernames);
    WebElement pass = driver.findElement(By.name("password"));
    pass.sendKeys(passwords);
}
```

Valid() Method:

Takes a parameter called "cases" representing whether the test case is "Valid" or "Invalid". Prints messages based on the validity of the test case, aiding in test result interpretation. This action occurs when the feature file calls the ' And Keeping case "<Case>" as Valid ’ step, with <Case> being taken from the examples provided at the bottom of the feature file.

```java
@And("Keeping case {string} as Valid")
public void valid(String cases) {
    if(cases.equals("Valid")) {
        System.out.println("It is a Valid User");
    }
    else {
        System.out.println("It is and InValid User");
    }
}
```

Login_with_credentials() Method:

Simulates clicking the login button after entering the username and password. Locates the login button element on the login page and clicks it. This action occurs when the feature file calls the ' Then User should get logged in' step.

```java
@Then("User should get logged in")
public void login_with_credentials()  {
    WebElement l = driver.findElement(By.xpath("(//button[normalize-space()='Login'])[1]"));
    l.click();
}
```

Login_success() Method:

It waits for a brief period after clicking the login button to allow the application to process the login request. It captures the current URL and compares it with the expected URL of the secure page to verify successful login and prints an appropriate message indicating login success or failure based on the comparison result. This action occurs when the feature file calls the ' And Message displayed Login Successfully or not ' step.

```java
@And("Message displayed Login Successfully or not")
public void login_success() throws InterruptedException {
    Thread.sleep(2000);
    String expectedURL = "https://practice.expandtesting.com/secure";
    String actualURL = driver.getCurrentUrl();
    if (expectedURL.equals(actualURL)) {
            System.out.println("Login Successful");
       } else {
            System.out.println("Login Failed");
       }
}
```

Invalid() Method:

Similar to the valid() method, this method prints messages based on whether the provided case is "Valid" or "Invalid," aiding in result interpretation. This action occurs when the feature file calls the ' And Keeping case "<Case>" as InValid ’ step, with <Case> being taken from the examples provided at the bottom of the feature file.

```java
@Then("Keeping case {string} as InValid")
public void invalid(String cases) {
    if(cases.equals("Valid")) {
    System.out.println("It is a Valid User");
    }
    else {
        System.out.println("It is and InValid User");
    }
}
```

Login_again() Method:

It refreshes the page if the login attempt is unsuccessful, ensuring subsequent tests start from a consistent state and prints a message indicating the user has been redirected back to the login page. This action occurs when the feature file calls the ' And user will be asked to go back to login page if user is InValid "<Case>"' step, with <Case> being taken from the examples provided at the bottom of the feature file.

```java
@And("user will be asked to go back to login page if user is InValid {string}")
public void login_again(String cases) {
    if(cases.equals("Valid")) {
        System.out.println("User will not be asked to go back to login page");
    }
    else {
        driver.navigate().refresh();
        System.out.println("User went back to login page");
    }
}
```

Provide_valid_user() Method:

It reminds the user to provide valid credentials in case of an invalid login attempt and prints a message instructing the user to provide valid credentials, enhancing test clarity. This action occurs when the feature file calls the ' Then Provide correct credentials if user is InValid "<Case>"' step, with <Case> being taken from the examples provided at the bottom of the feature file.

```java
@Then("Provide correct credentials if user is InValid {string}")
public void provide_valid_user(String cases) {
    if(cases.equals("Valid")) {}
    else {
        System.out.println("Provide Valid Credentials");
    }
    driver.quit();
}
```

Feature File:

```
Feature: Login Action
    Description: This feature will test a LogIn and LogOut functionality

    Scenario: Login with valid Credentials
        Given User is on Home Page
        When User navigate to Login Page
        Then User enters "<username>" and "<password>"
        And Keeping case "<Case>" as Valid
        Then User should get logged in
        And Message displayed Login Successfully or not
        Then Keeping case "<Case>" as InValid
        And user will be asked to go back to login page if user is InValid "<Case>"
        Then Provide correct credentials if user is InValid "<Case>"

    Examples:
        | username          | password | Case    |
        | practice | SuperSecretPassword! | Valid   |
        | abc1@gmail.com    | dfsd2    | InValid |
```

Console Output:

```
Mar 24, 2024 2:35:23 PM cucumber.api.cli.Main run
WARNING: You are using deprecated Main class. Please use io.cucumber.core.cli.Main

Scenario: Login with valid Credentials                                  # src/test/java/Feature/loginfeature.feature:17
Starting ChromeDriver 122.0.6261.111 (9d4c1072da62b411b351a38b9ed6214ab236aa7b-refs/branch-heads/6261@{#1015}) on port 34641
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
[1711271124.801][WARNING]: This version of ChromeDriver has not been tested with Chrome version 123.
Mar 24, 2024 2:35:24 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
  Given User is on Home Page                                            # Feature.Logins.open_chrome_and_open_application()
  When User navigate to Login Page                                      # Feature.Logins.login_page()
  Then User enters "practice" and "SuperSecretPassword!"                # Feature.Logins.enter_username_and_password(java.lang.String,java.lang.String)
It is a Valid User
  And Keeping case "Valid" as Valid                                     # Feature.Logins.valid(java.lang.String)
  Then User should get logged in                                        # Feature.Logins.login_with_credentials()
Login Successful
  And Message displayed Login Successfully or not                       # Feature.Logins.login_success()
It is a Valid User
  Then Keeping case "Valid" as InValid                                  # Feature.Logins.invalid(java.lang.String)
User will not be asked to go back to login page
  And user will be asked to go back to login page if user is InValid "Valid" # Feature.Logins.go_back(java.lang.String)
  Then Provide correct credentials if user is InValid "Valid"           # Feature.Logins.provide_valid_user(java.lang.String)

Scenario: Login with valid Credentials                                  # src/test/java/Feature/loginfeature.feature:18
Starting ChromeDriver 122.0.6261.111 (9d4c1072da62b411b351a38b9ed6214ab236aa7b-refs/branch-heads/6261@{#1015}) on port 25434
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
[1711271131.533][WARNING]: This version of ChromeDriver has not been tested with Chrome version 123.
Mar 24, 2024 2:35:31 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
  Given User is on Home Page                                            # Feature.Logins.open_chrome_and_open_application()
  When User navigate to Login Page                                      # Feature.Logins.login_page()
  Then User enters "abc1@gmail.com" and "dfsd2"                         # Feature.Logins.enter_username_and_password(java.lang.String,java.lang.String)
It is and InValid User
  And Keeping case "InValid" as Valid                                   # Feature.Logins.valid(java.lang.String)
  Then User should get logged in                                        # Feature.Logins.login_with_credentials()
Login Failed
  And Message displayed Login Successfully or not                       # Feature.Logins.login_success()
It is and InValid User
  Then Keeping case "InValid" as InValid                                # Feature.Logins.invalid(java.lang.String)
User went back to login page
  And user will be asked to go back to login page if user is InValid "InValid" # Feature.Logins.go_back(java.lang.String)
Provide Valid Credentials
  Then Provide correct credentials if user is InValid "InValid"         # Feature.Logins.provide_valid_user(java.lang.String)

2 Scenarios (2 passed)
18 Steps (18 passed)
0m14.886s
```