Test Approach

About

As part of the testing assessment, we are going to test the fictitious e-commerce eShopOnWeb application. It provides customer an online shopping experience. It provides various functionalities starting from viewing the list of products to adding it to cart and successfully checkout along with payment option. It provides admin and non-admin login functionalities. All these functionalities are considered while testing the application.

Selenium, in conjunction with Java is used for the automation of the scenarios. It is a powerful open-source testing framework that allows to automate the web application tests. It offers flexibility to write script in various programming languages, including Java, C#, and Python. Selenium can operate across different browsers and operating systems.

For API testing, we preferred Karate framework. It is a very powerful API test-automation tool whose syntax is language neutral and can be used with several programming languages including Java.

**Scenarios:**

In the process of designing the scenario, the key functionalities of the application have been considered. The primary pages selected for testing encompass the Login, Catalog, Cart, and Checkout pages. These pages have been chosen as they represent crucial user interaction points within the application.

1. **Login:**

Login is one of the key features in any application. It helps user in sign in into the application and authenticates them successfully based on their credentials. While testing this feature, both admin and non-admin users are considered. Also, the positive and negative scenario such as login with valid or invalid credentials are verified here. For data driven testing, Scenario Outline is used which allows the same test to be executed with various data sets.

**Approach:** BDD Framework with Cucumber, JUnit and Selenium Java for testing. For positive scenario, valid email id and password is used for login successfully. For negative scenario, invalid email id is used.

**Scenario 1:** A non-admin user with valid email and password should be login into the eShopOnWeb application successfully. In this case, we have used below data, and it can be extended with various examples.

|  |  |
| --- | --- |
| Email | Password |
| demouser@microsoft.com | Pass@word1 |

**Scenario 2:** A user with invalid email and password should not be allowed to login into the eShopOnWeb application. Below is the data which we have considered for testing.

|  |  |
| --- | --- |
| Email | Password |
| fail@microsoft.com | Pass@word1 |

**Scenario 3:** An admin user with valid email and password should be login into the eShopOnWeb application successfully. We have used [admin@microsoft.com](mailto:admin@microsoft.com) as valid admin email for this scenario.

|  |  |
| --- | --- |
| Email | Password |
| admin@microsoft.com | Pass@word1 |

1. **Catalog:**

As part of the Catalog Page feature, the objective is to efficiently browse products, seamlessly add them to the cart, and effortlessly search for specific items, thereby streamlining and accelerating the shopping experience.

**Approach:** BDD Framework with Cucumber, JUnit and Selenium Java for testing. For data driven testing, Scenario Outline is used which allows the same test to be executed with various data sets.

**Scenario 1:** User should be able to view the products in the catalog page. All the products should be visible is verified in this scenario.

**Scenario 2:** Add User should be able to see the list of products and select a particular product and add it to the cart. Product should be successfully added to the cart. This scenario provides flow of end-to-end selection and adding of product to cart.

**Scenario 3:** User should be able to filter the product based on the brand name and product type. This scenario is considered for data driven testing, so we used Scenario outline. In this case we have used .NET and Mug as brand name and product type to get the filtered product. This can be extended with various examples.

|  |  |  |
| --- | --- | --- |
| Brand Name | Product Type | Filtered Product |
| .NET | Mug | .NET Black & White Mug |

1. **Cart:**

When any product gets added to basket it appears into the cart page. As part of testing of this feature, after adding the product whether it appears into the cart page or not is verified. Also, the quantity of item is modified to see if the cart value gets updated or not. Along with this, the checkout functionality is enabled for the user is also checked here.

**Approach:** BDD Framework with Cucumber, JUnit and Selenium Java for testing. For checking the cart value gets updated the quantity of the product is changed.

**Scenario 1:**  User should be able to select any product from the cart and add it to basket and same should be displayed in the cart page.

**Scenario 2:** User should be able to update the quantity of the product in the cart page and same quantity should be reflected into the cart page. We have used the example to update the quantity to “2” and verified the product count.

**Scenario 3:** User should be able to successfully checkout the product from the cart page after adding the product. Here, functionality to checkout is enabled for user or not is verified.

1. **Checkout:**

Once the product gets added to cart next step is to check out and make payment. In this feature, we have tested the user is able to check out the product and complete the payment successfully. Once the payment is made the order is complete.

**Approach:** BDD Framework with Cucumber, JUnit and Selenium Java for testing. For authentication the approach of data-driven testing is done using scenario outline.

**Scenario:** User should be able to review the items in the cart and complete the payment. In the course of checkout if user has not signed in yet, then he/she will be redirected to the login page to get authenticated and after that, payment can be made.