Cucumber Framework for Amazon Application

Overview:

The objective is to create an automation framework using Cucumber and Selenium WebDriver to automate basic functionalities of the Amazon application. This framework employs the Page Object Model (POM) design pattern for better code organization and maintainability.

Components:

Maven Project Setup: Utilized Maven as the build tool for dependency management. The primary dependencies include Selenium WebDriver, Cucumber, and JUnit.

Project Structure: Organized the project into a structured format with distinct packages for pages, utilities, step definitions, and hooks to maintain modularity and scalability.

WebDriver Manager: Implemented a DriverManager class to manage the WebDriver instance. This class initializes the ChromeDriver for browser interactions.

Page Object Model (POM): Created a HomePage class within the pages package. This class encapsulates the methods related to the Amazon homepage, such as navigating to the homepage and searching for products.

Step Definitions: Defined step definitions in the StepDefinitions class under the tests package. The step definitions are written using Cucumber's Gherkin syntax to represent the Given-When-Then scenarios.

Cucumber Hooks: Integrated Cucumber hooks by implementing @Before and @After annotations in the CucumberHooks class. These hooks ensure that the WebDriver instance is initialized before the test execution and closed after the test completion.

Integration with Jenkins: Established integration with Jenkins for continuous integration. Configured a Maven project in Jenkins and set up a build job to trigger the test execution using the mvn clean test command.

Execution Flow:

Jenkins triggers the build job upon a specific event or schedule.

Maven builds the project, resolving dependencies and compiling the source code.

During the execution phase, Cucumber reads the feature files and maps them to the corresponding step definitions.

The WebDriver interacts with the Amazon application, executing the defined test scenarios.

Upon test completion, the results are reported, and Jenkins provides insights into the test execution status.

Conclusion:

The implemented Cucumber framework facilitates the automation of basic functionalities for the Amazon application, ensuring reliability, maintainability, and scalability. By leveraging Selenium WebDriver and Cucumber, the framework streamlines the testing process, enabling continuous integration and delivery through Jenkins. Future enhancements can include integrating additional functionalities, implementing data-driven testing, and enhancing reporting mechanisms for comprehensive test coverage.