|  |  |  |  |
| --- | --- | --- | --- |
|  |  | SQE Phase#1  22F-3664 **22f-3660** |  |
|  |  | **UI AUTOMATION TESTING** |  |
|  | | | |

|  |  |  |
| --- | --- | --- |
|  |  |  |

**Web UI Test Automation Framework Documentation**

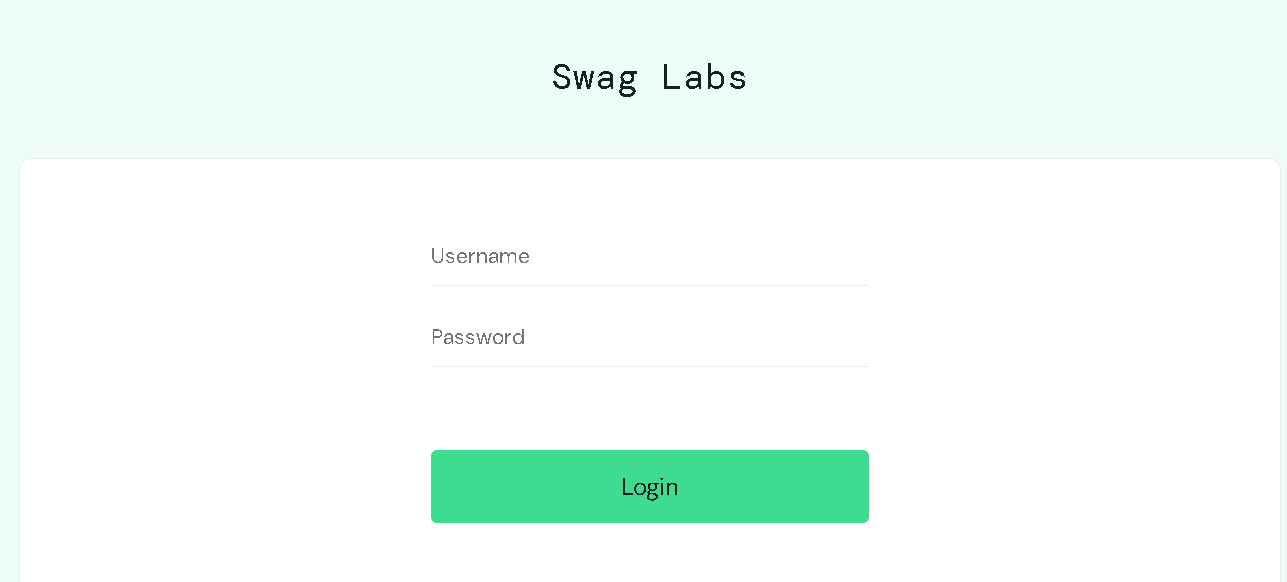
**Author: Muhammad Abdullah[3664] , USMAN ANWAR[3660]**

**Table of Contents**

1. **Introduction**
2. **Setup Instructions**
3. **Usage**
4. **Writing and Executing Test Cases**
5. **Design Decisions**
6. **Integrating with Data Sources and Generating Reports**
7. **GitHub Management**
8. **Conclusion**

**1. Introduction**

This document provides comprehensive documentation for the **Web UI Test Automation Framework**, designed for testing the **Swag Labs** website. The framework is built using Java and integrates **Selenium**, **Cucumber**, and **Allure** for effective UI testing.

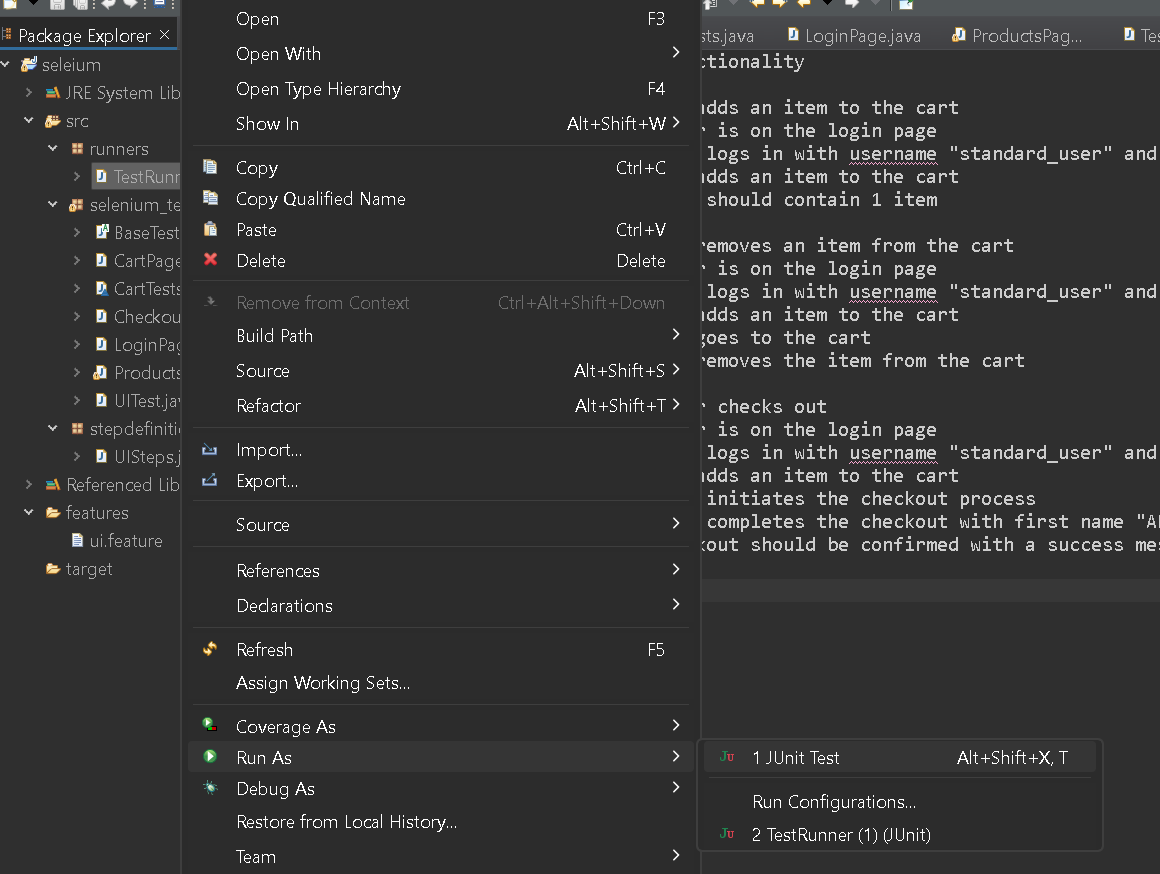
**Project Overview** The framework tests three main scenarios related to the cart functionality of the Swag Labs website, ensuring a robust user experience.  


**2. Setup Instructions**

To set up the framework, follow these steps:

1. **Install Eclipse IDE**: Download and install the latest version of Eclipse.
2. **Install Java Development Kit (JDK)**: Ensure you have JDK installed. Set the JAVA\_HOME environment variable.
3. **Create a New Java Project**: Open Eclipse, go to File > New > Java Project, and name it WebUITestAutomation.
4. **Add Required Libraries**:
   * Add **Selenium** dependencies.
   * Add **Cucumber** dependencies for BDD.
   * Add **Allure** for reporting.
5. **Configure Project Structure**: Organize the project into the following packages:
   * pages: Contains Page Object Model (POM) classes.
   * stepdefinitions: Contains step definitions for Cucumber.
   * runners: Contains test runner classes.
   * tests: Contains Gherkin feature files.

**To execute the test cases, run the Cucumber test runner class within the runners package as a JUnit Test.**

1. **Open the TestRunner class.**
2. **Right-click on the class file.**
3. **Select Run As > 1JUnit Test.  
   **

**4. Writing and Executing Test Cases**

The framework uses Gherkin syntax to define test cases. Below are the scenarios tested:

**Feature: Cart Functionality**

**Scenario 1: User Adds an Item to the Cart**

Given the user is on the login page

When the user logs in with username "standard\_user" and password "secret\_sauce"

And the user adds an item to the cart

Then the cart should contain 1 item

**Scenario 2: User Removes an Item from the Cart**

Given the user is on the login page

When the user logs in with username "standard\_user" and password "secret\_sauce"

And the user adds an item to the cart

And the user goes to the cart

And the user removes the item from the cart

**Scenario 3: User Checks Out**

Scenario: User checks out

Given the user is on the login page

When the user logs in with username "standard\_user" and password "secret\_sauce"

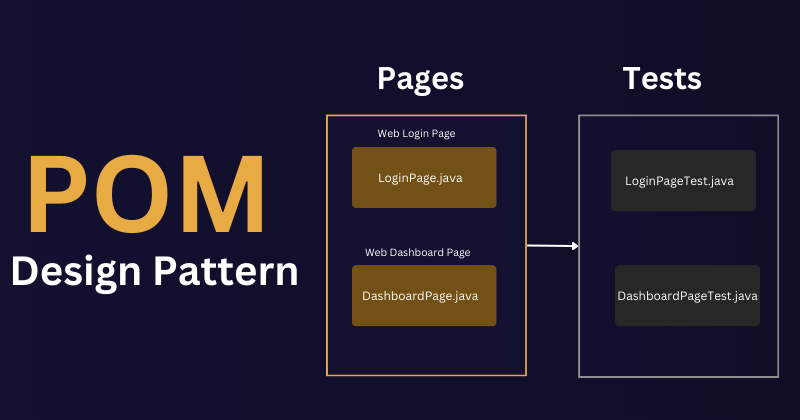
And the user adds an item to the cart

When the user initiates the checkout process

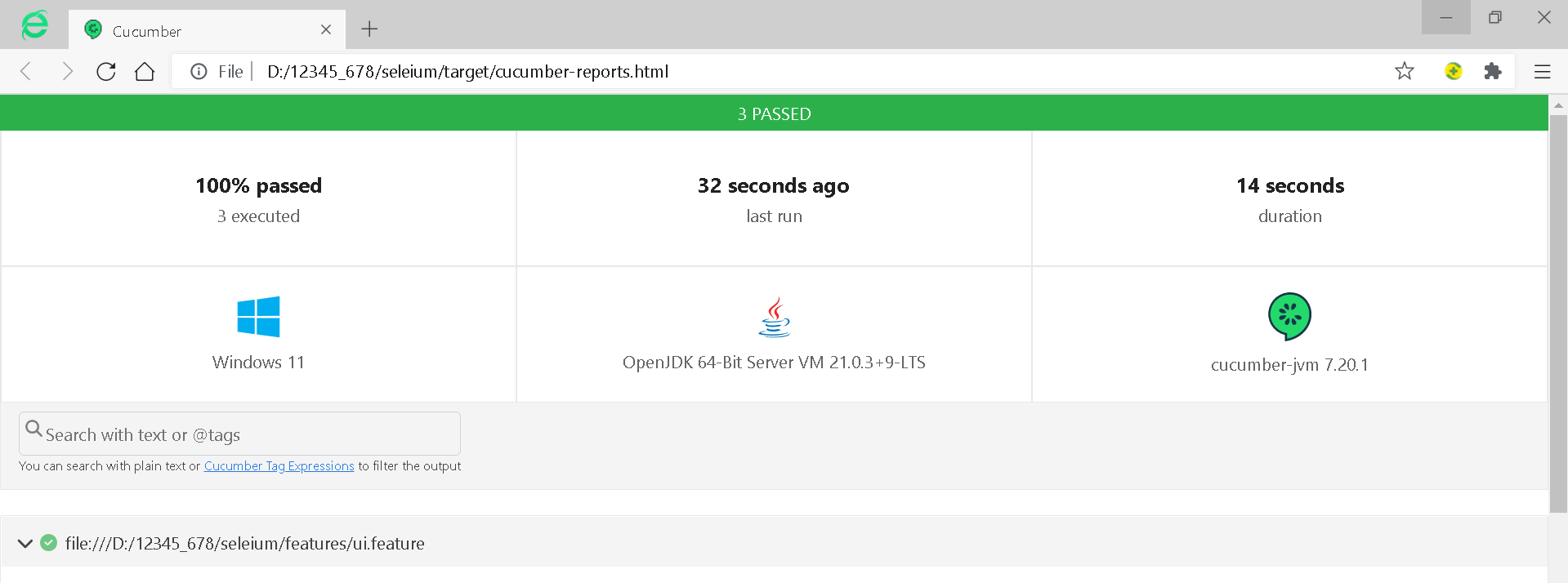
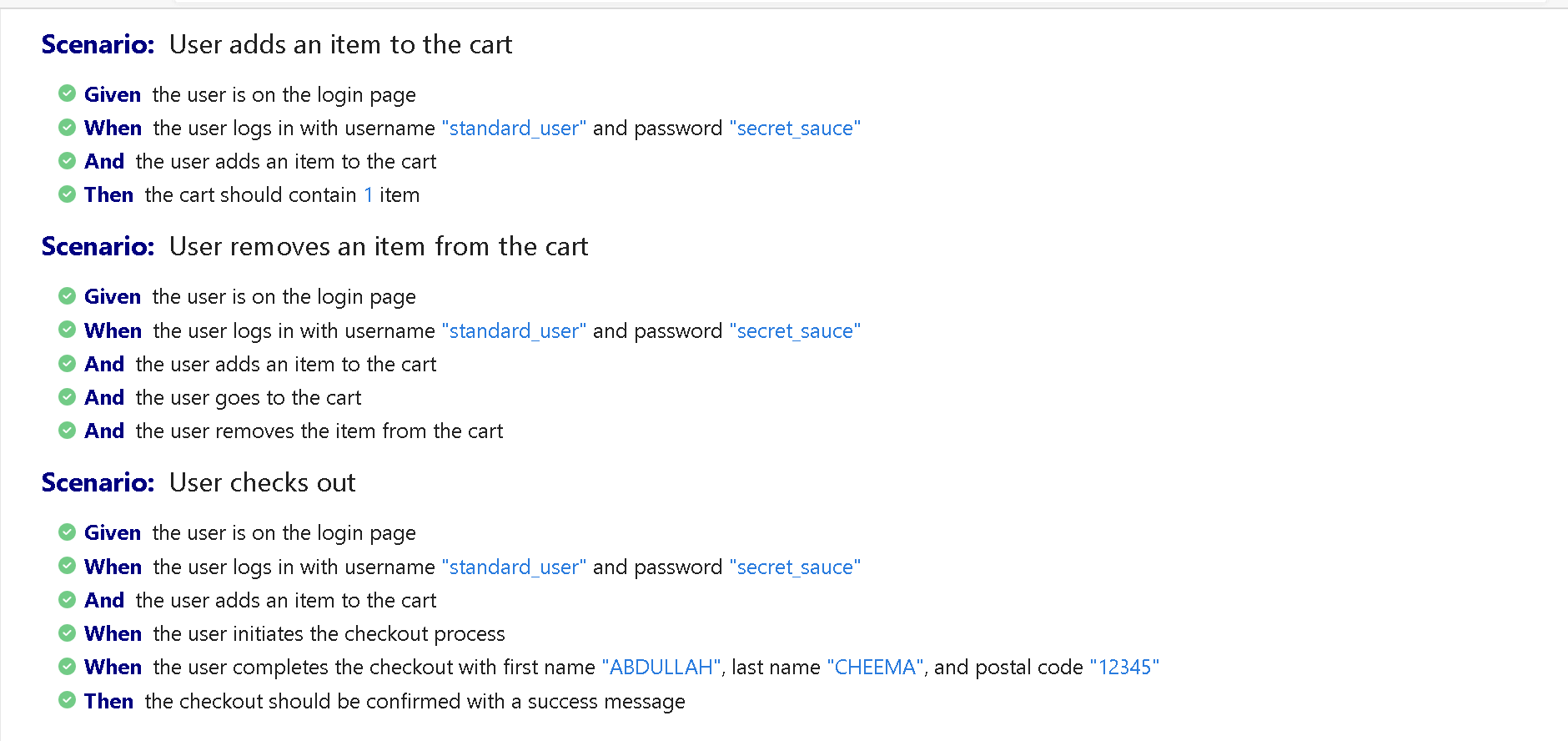
When the user completes the checkout with first name "ABDULLAH", last name "CHEEMA", and postal code "12345"

Then the checkout should be confirmed with a success message

**5. Design Decisions**

The framework utilizes the **Page Object Model (POM)** to enhance maintainability and readability. POM separates the test logic from the UI interactions, making the framework easier to understand and modify.  


**6. Integrating with Data Sources and Generating Reports**

* **Allure Reporting**: After executing the test cases, Allure generates detailed reports. To view reports, run the Allure command in the terminal after your tests have executed:  
    
    
  

**7. GitHub Management**

For version control and collaboration, GitHub is utilized:

1. **Collaboration**: One team member uploads their part of the project.
2. **Pull Requests**: Use pull requests to review and merge code.
3. **Issue Tracking**: Track bugs and enhancements using GitHub issues.

When completing the project, ensure to pull the latest changes from the repository, complete your part, and push back to GitHub.

**8. Conclusion**

The **Web UI Test Automation Framework** for the Swag Labs website provides a robust solution for automated testing. Utilizing Java, Selenium, Cucumber, and Allure, the framework ensures reliable testing of cart functionalities, contributing to a better user experience.