Automating the Star Health Application Using Selenium

Githublink-https://github.com/Utkarsh9991/Testing.git

***Feature File:***

The feature file orchestrates a comprehensive suite of behavior tests for the Star Health application, specifically tailored for the Chrome browser. The primary scenario, denoted by the "Validate the Star Health Buy Now flow" outline, meticulously guides through the intricate steps necessary to ensure the seamless functioning of the Buy Now feature. These steps encompass launching the application, handling pop-ups, asserting the home page title, interacting with the Buy Now button, inputting user details, navigating through dropdown menus, and validating entered information. The feature file also provides a structured example table, offering test data for various scenarios.

***Test Runner (TestRunner.java):***

The Test Runner class serves as the orchestrator, configuring and executing Cucumber tests seamlessly. It intricately specifies the location of the feature file, the package containing step definitions (glue), and employs various reporting plugins, including HTML and Extent reports. This setup ensures a robust reporting mechanism for test execution outcomes.

***Step Definitions (StepDefinition.java):***

Step Definitions, the heart of the automation suite, encapsulate the actual logic and implementation of the steps articulated in the feature file. These Java methods interact seamlessly with Page Objects, orchestrating actions such as launching the application, clicking buttons, entering user information, and asserting expected outcomes. The step definitions provide a clear bridge between the high-level feature descriptions and the low-level automation actions.

***Page Objects (HomePage.java, BuyNowpage.java):***

Page Objects, akin to digital replicas of web pages, encapsulate the interactions with the application's user interface. HomePage incorporates methods to retrieve the homepage title and click on the Buy Now button, while BuyNowpage includes functionality for entering user information, interacting with dropdowns, and performing validations. The Page Object Model (POM) design pattern ensures maintainability, scalability, and readability of the automation suite.

***Base Test (BaseTest.java):***

The Base Test class lays the foundation by initializing the WebDriver, specifically the ChromeDriver, ensuring a clean slate for each test scenario. Additionally, the class incorporates an @AfterAll method, responsibly closing the browser post-test execution. This encapsulation guarantees a pristine testing environment, contributing to the reliability and reproducibility of the automated tests