Automating the Star Health Application Using Selenium

Scenario 1 Cucumber + JUnit

Automating the Star Health Buy Now flow using Cucumber and JUnit.

Step 1: Create a Maven Project

- 1. Open your IDE (Eclipse/IntelliJ).
- 2. Create a new Maven project With Maven-Archtype-Quickstart.
- 3. Provide a group and artifact ID.
- 4. Click "Finish."

Step 2: Add Dependencies in POM.XML

- 1. Open the pom.xmlfile.
- 2. Add dependencies for Cucumber, JUnit, Selenium, and required libraries.

```
<dependencies>
       <dependency>
               <groupId>junit
               <artifactId>junit</artifactId>
               <version>4.11</version>
               <scope>compile</scope>
       </dependency>
       <!-- https://mvnrepository.com/artifact/io.cucumber/cucumber-java -->
       <dependency>
               <groupId>io.cucumber</groupId>
               <artifactId>cucumber-java</artifactId>
               <version>7.10.1
       </dependency>
       <!-- https://mvnrepository.com/artifact/io.cucumber/cucumber-java -->
       <dependency>
               <groupId>io.cucumber</groupId>
               <artifactId>cucumber-junit</artifactId>
               <version>7.10.1
               <scope>compile</scope>
       </dependency>
       <!-- https://mvnrepository.com/artifact/io.cucumber/cucumber-java -->
```

```
<dependency>
                <groupId>io.cucumber</groupId>
                <artifactId>cucumber-core</artifactId>
                <version>7.10.1
        </dependency>
        <dependency>
                <groupId>org.seleniumhq.selenium/groupId>
                <artifactId>selenium-java</artifactId>
                <version>4.11.0</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
        <dependency>
                <groupId>io.github.bonigarcia/groupId>
                <artifactId>webdrivermanager</artifactId>
                <version>5.4.1</version>
        </dependency>
        <dependency>
                <\!groupId\!\!>\!tech.grasshopper<\!/groupId\!\!>
                <artifactId>extentreports-cucumber7-adapter</artifactId>
                <version>1.7.0</version>
        </dependency>
</dependencies>
```

Step 3: Create a Cucumber Feature File : starHealth.feature

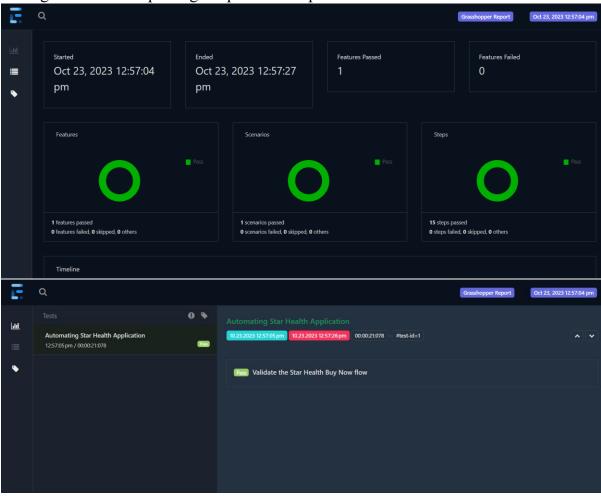
```
1 Feature: Automating Star Health Application
    Scenario Outline: Validate the Star Health Buy Now flow
      Given User launches the Star Health application with "<URL>"
      And User waits for the Welcome to Star Health pop-up and closes it And User validates the Star Health home page title using a JUnit assertion
      And User clicks on the Buy Now button
And User types Name as "<FullName>"
      And User types Phone as "<PhNo>"
      And User types the PIN as "<PIN>"
      And User clicks on I need health insurance from the drop-down menu And User selects the option "<OptionPlan>"
      And User sees the Plan for My Family page
      And User validates that the mobile number is the same as the previously entered number using a JUnit assertio And User clicks on the Star Health logo
      And The Application should redirect to the home page
      And User closes the child tab
      And User navigates back to the parent tab
                                          | FullName | PhNo
                                                                      PIN
                                                                                | OptionPlan |
```

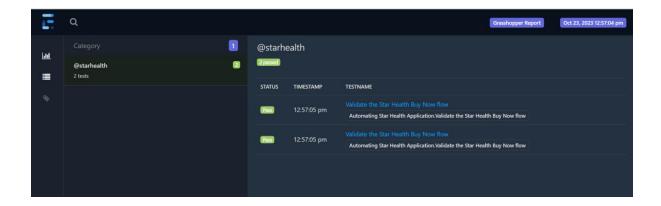
Step 5: Parameterize Test Data

- 1. Parameterize test data using the Cucumber Table.
- 2. Define examples for the scenario.

Step 6: Use Extend Reporting

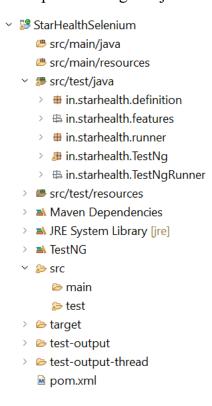
1. Configure Extend Reporting for pass/fail steps.





Step 7: Implement Page Object Model

- 1. Create page objects for the web elements.
- 2. Implement Page Object Model design pattern for a clean structure.



Step 8: Implement Step Definitions

- 1. Create a step definition file.
- 2. Implement step definitions using Selenium and JUnit assertions.

```
👄 eclipse-workspace - StarHealthSelenium/src/test/java/in/starhealth/definition/StarHealth.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
public class StarHealth {
    public static WebDriver driver = new ChromeDriver();
  LearnCucumber
                                                             @Given("User launches the Star Health application with {string}")
public void user_launches_the_Star_Health_application_with{String url) {
    // Initialize WebDriver, open the URL, and other setup
    driver_manage().window().max_mize();
    driver_manage().deleteAllCookies();
  ➢ Pom-Framework
  StarHealthSelenium
     src/main/javasrc/main/resources
                                                                   driver.get(url);

→ 

⊕ in.starhealth.definition

                                                             @And("User waits for the Welcome to Star Health pop-up and closes it")
public void user_waits_for_welcome_popup_and_closes_it() {
           StarHealth.java
       > # in.starhealth.features
                                                                    WebElement popup = driver.findElement(By.xpath("/html/body/div[5]/div/div[2]/div/div[2]/button"));
if (popup.sblisplayed()) {
    popup.click();

→ 

in.starhealth.runne

         > ② StartHealthRunner.java
           testna.xml
      > # in.starhealth.TestNg
> # in.starhealth.TestNgRunner
                                                                  System.out.println("User waits for the Welcome to Star Health pop-up and closes it");
     src/test/resources
     ■ Maven Dependencies

■ JRE System Library [jre]
                                                             @And("User validates the Star Health home page title using a JUnit assertion")
public void user_validates_home_page_title() {
   String expectedTitle = "Star Health Insurance: Medical, Accident and Travel insurance policies";
   String actualTitle = driver.getTitle();
   Assert.assertEquals(expectedTitle, actualTitle);
     ■ TestNG

    test

    > 🍅 target
                                                              @And("User clicks on the Buy Now button")
public void user_clicks_buy_now_button() {
    WebElement buyNowButton = driver.findElement(By.xpath("//*[@id=\"_next\"]/div/div/header/div[2]/div[2]/div/but
    buyNowButton.click();

    test-output

     🟭 Q 🥨 🖿 🔘 🗯 📜 🍀 💁 🕴 🧶 💌 🖨 🤣
```

Step 9: Create a Test Runner

- 1. Create a test runner class to run the Cucumber tests.
- 2. Configure the runner to execute scenarios from starHealth.feature.

```
🍔 eclipse-workspace - StarHealthSelenium/src/test/java/in/starhealth/runner/StartHealthRunner.java - Eclipse IDE
                                                                                                                             a X
File Edit Source Refactor Navigate Search Project Run Window Help
Q 📳 🐉
                     - -
₽ Package Explorer × む JUnit
               🖹 🖇 🖁 🚪 1 package in.starhealth.runner;
> 🛂 AmazonTestCucumber
                             3*import org.junit.runner.RunWith; □
> 🕏 JavaProject
> 🍃 Junit-Learning
                            7 @RunWith(Cucumber.class)
                           8 @CucumberOptions(features="C:\\Users\\karth\\eclipse-workspace\\StarHealthSelenium\\src\\test\\java\\in\\starhealth\\fe
> 🎏 LearnCucumber
> > Pom-Framework
                                                    glue="in.starhealth.definition",
> > Selenium-Learn
                                                    plugin= {"html:target/Cucumberreport.html",
                                                    "pretty",
StarHealthSelenium
                                                    "com.aventstack.extentreports.cucumber.adapter.ExtentCucumberAdapter:",
  src/main/java
                                                    "timeline:test-output-thread/"
  src/main/resources
                           14
 17 public class StartHealthRunner {
    > 🖟 StarHealth.iava
```

Step 10: Run Tests with Maven

- 1. Use Maven to run the Cucumber tests.
- 2. Verify test results

.

Step 11: Set Up Jenkins Integration

- 1. Create a FreeStyle Jenkins project.
- 2. Configure Jenkins to run this Cucumber project.
- 3. Trigger the Jenkins job and observe the results.

Scenario 2 TestNG + Selenium:

Automating various tasks using TestNG and Selenium.

Step 1: Create a testng.xml File

- 1. In the Maven project, create a testng.xml file.
- 2. Define a TestNG suite and test cases.

Step 2: Implement TestNG Test

1. Create a Java class for the TestNG test.

- 2. Implement test methods and assertions.
- 3. Use Extend Reporting for pass/fail steps.

Step 3: Configure TestNG Annotations

- 1. Use @BeforeClass and @BeforeSuite for setup.
- 2. Use @AfterClass and @AfterSuite for cleanup.

Step 4: Parameterize Data

1. Use parameterized data for test inputs, possibly from an external source like testNG.xml or Excel.

Step 5: Page Object Model

- 1. Implement the Page Object Model for a clean structure.
- 2. Create page objects for web elements.

Step 6: Run TestNG Scenarios

- 1. Run TestNG scenarios using the testNG.xml file.
- 2. Ensure tests execute as expected.

Step 7: Parallel Execution

- 1. Create another TestNG test in testng.xml.
- 2. Configure it to run in parallel with the first test.
- 3. Execute both tests concurrently.

Step 8: Git Version Control

1. Upload the entire code to a Git repository.

Step 9: Jenkins Integration

1. Configure Jenkins to fetch the code from Git.

- 2. Set up Jenkins to execute the TestNG tests through Jenkins jobs.
- 3. Run the Jenkins job and observe results.

GitHub repository: https://github.com/Karthick-Office/Project01.git

Automating the Star Health Application Using Selenium Project is present under the "Selenium Phase 2 End Project" folder in the My GitHub repository