**AIM**: Create a Testing Framework for Sporty Shoes Website

**SOFTWARE**: Eclipse, JMeter, Postman

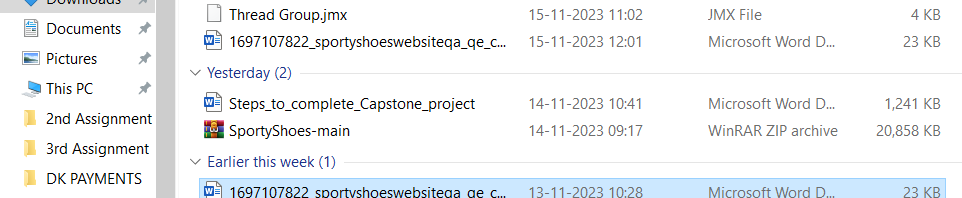
**Step 1**: Launch the Application under test.

Go to GitHub URL <https://github.com/Simplilearn-Edu/SportyShoes.git>

Click on Green button which has Code written on it. Click on Download Zip

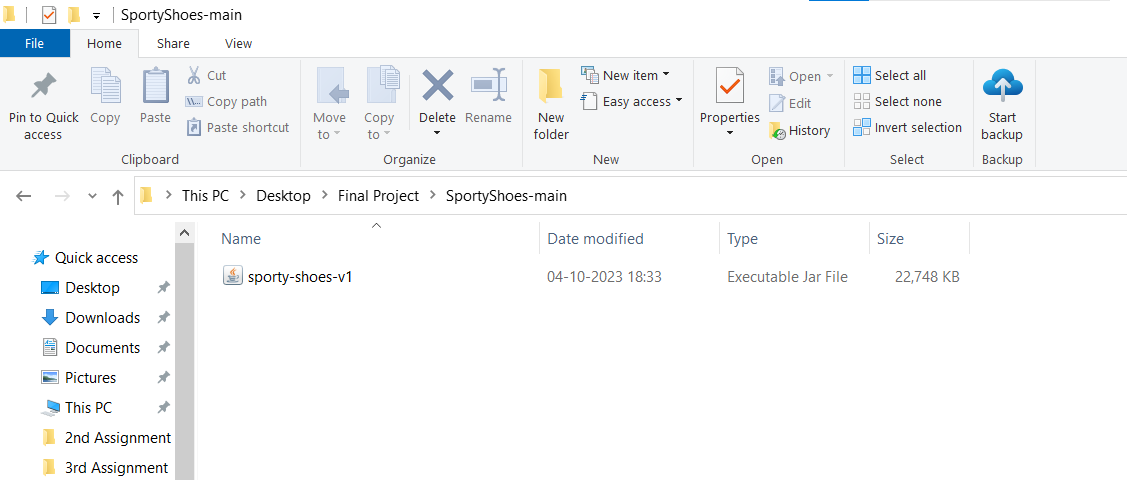
The download zip file will be in your download folder.

Extract all the files in downloads folder.



Go inside the folder Sporty Shoes-main and then again go in the folder Sporty Shoes-main.

Go inside the folder where you see the sporty-shoes-v1 jar file.



Start cmd prompt in the same folder.

A screenshot of a computer

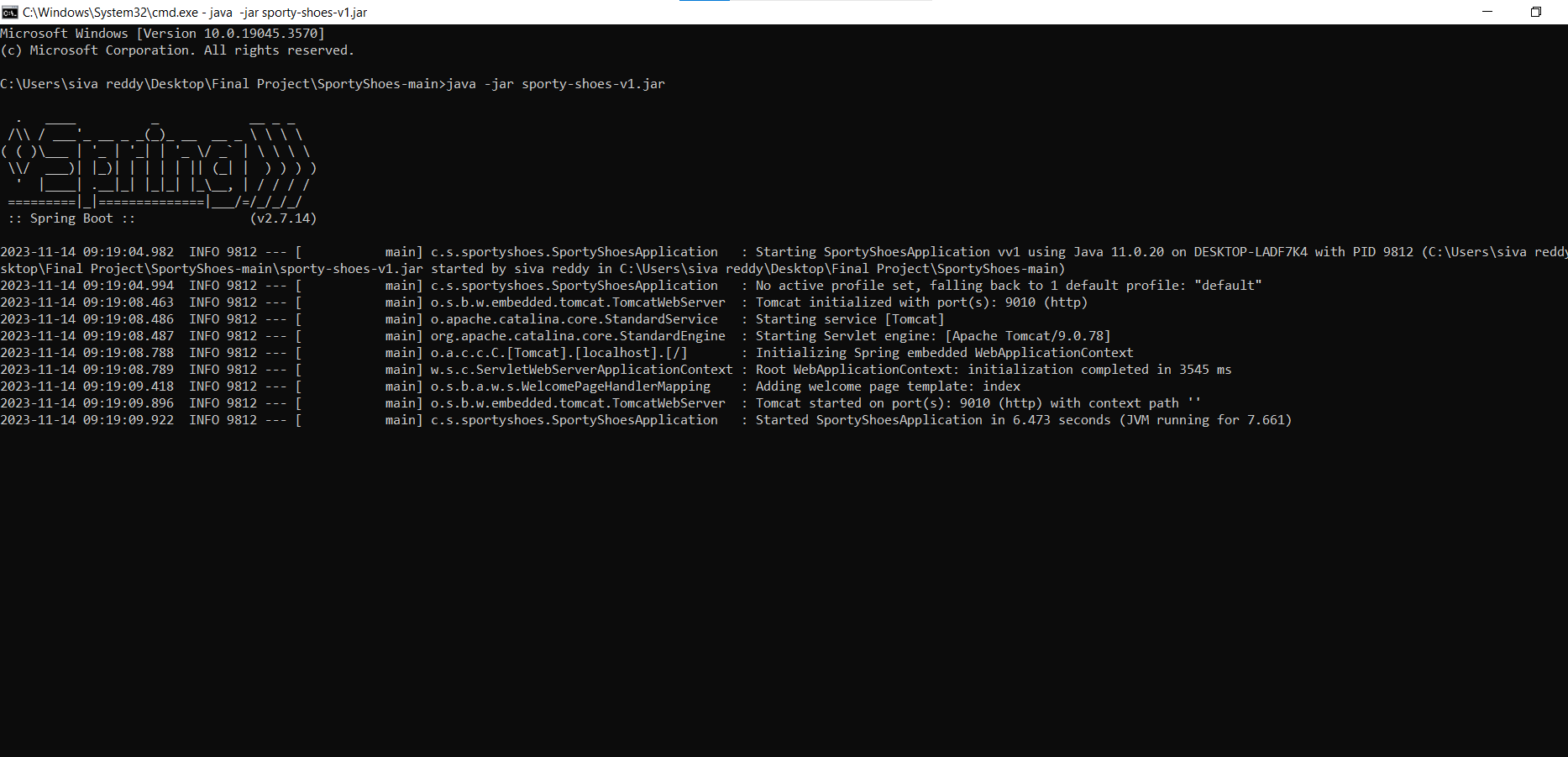
Description automatically generated

Execute the java command to run the jar file and deploy the application

java -jar sporty-shoes-v1.jar

OR

java -jar -Dserver.port=8100 sporty-shoes-v1.jar



After Execute the commands in command prompt until it minimize the window to run the postman execution.

Now we open chrome browser to run the command localhost:9010 command to open or not.

A screenshot of a computer

Description automatically generated

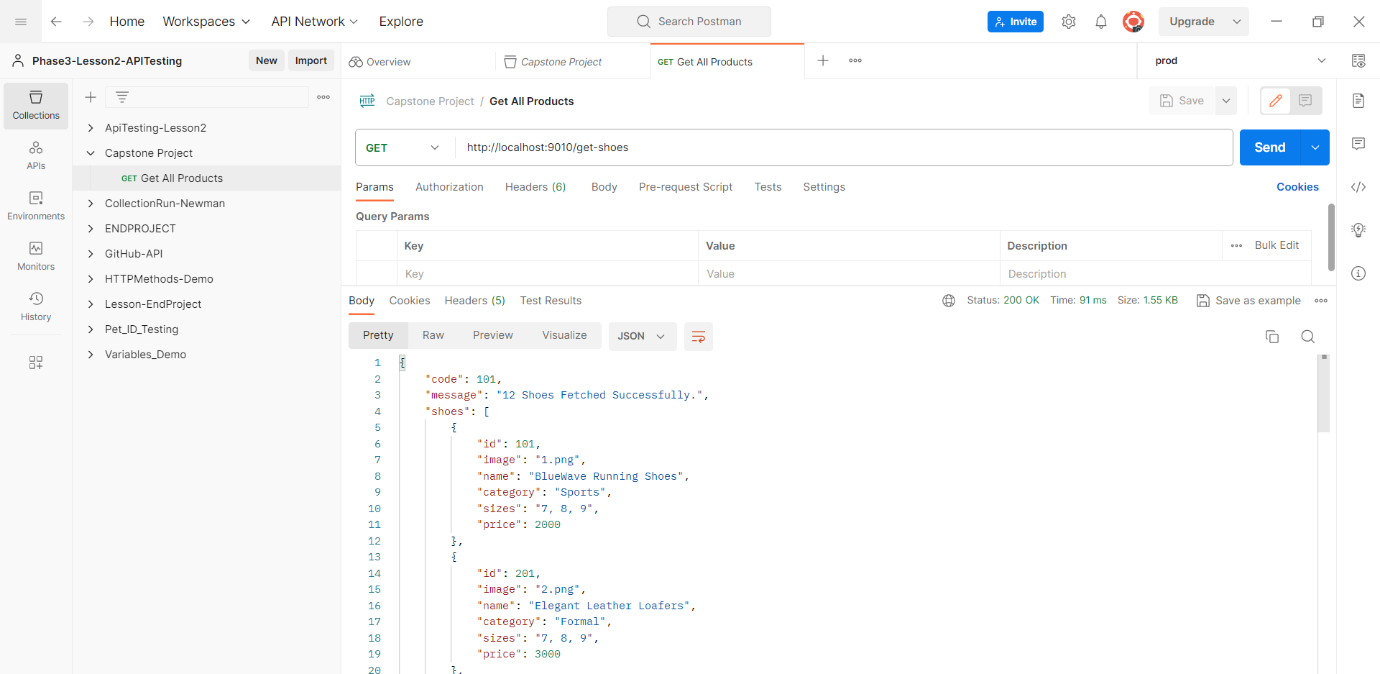
**Implementing Postman steps:**

We can open the postman to create the blank collection and its name is capstone project.

After we add a request.

**Step1:** Retrieve the list of all products in the store.

We can use command in postman to use “[**http://localhost:9010/get-shoes**](http://localhost:9010/get-shoes)”, to use ’**GET**’ method and get all products of shoes.

****

**Step2:** Retrieve the list of all registered users.

Again, we can add request to get all registered users. Now we can use command to get all registered users “[**http://localhost:9010/get-users**](http://localhost:9010/get-users)” again we can use the “**GET**” method.

**A screenshot of a computer

Description automatically generated**

**Step3:** Add Product URL Sample with POST Method.

Again, we can add request to add the product. Now we can use command to add the product “**http://localhost:9010/add-shoe?id=10001&image=www.image.com&name=SampleShoe&category=Running&sizes=12&price=1500**” we can use the “**POST**” method.

A screenshot of a computer

Description automatically generated

**Step4:** Update Product URL Sample with PUT Method.

Again, we can add request to update the product. Now we can use command to update the product “**http://localhost:9010/update-shoe?id=101&image=www.image.com&name=SampleShoe&category=Running&sizes=6,7,9&price=5000**” we can use the “**PUT**” method.

We can add the body in raw type is JSON.

**Request body** ([http://localhost:9010/update-shoe](http://localhost:9010/add-shoe))

{

"id": 101,

"name": "Updated Shoe Name",

"category": "Updated Category",

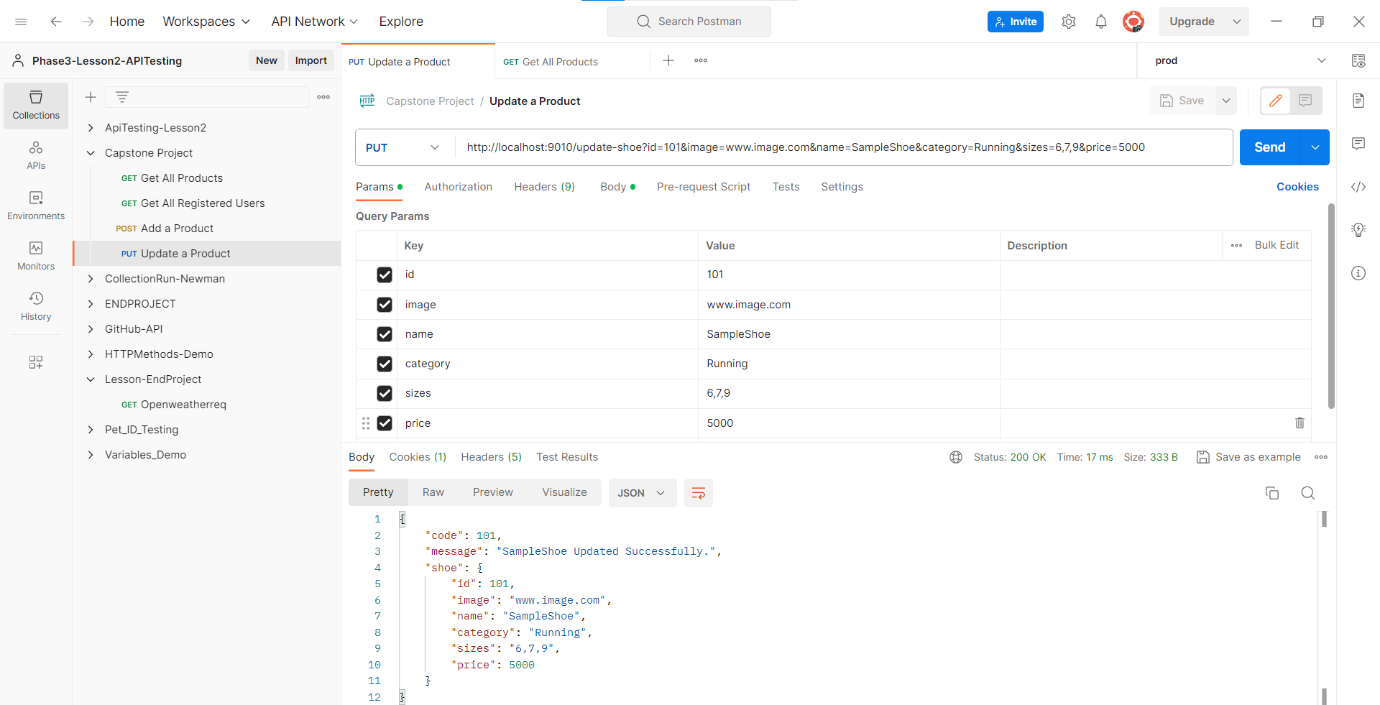
"sizes": "8,9,10",

"price": 1500

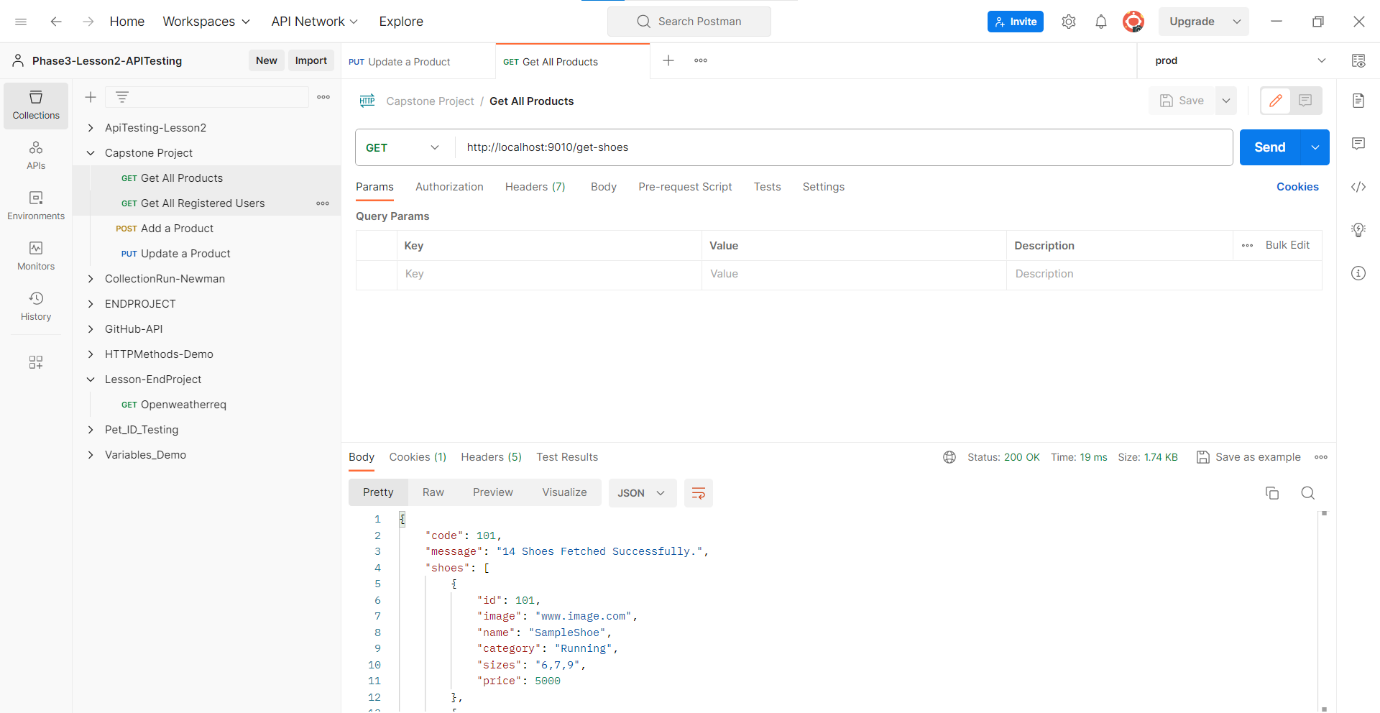
"image": "updated\_image\_url",

}

In this body cannot run in the postman its not execute the body so that we can make the change in given URL so now we can run the command it execute the command.

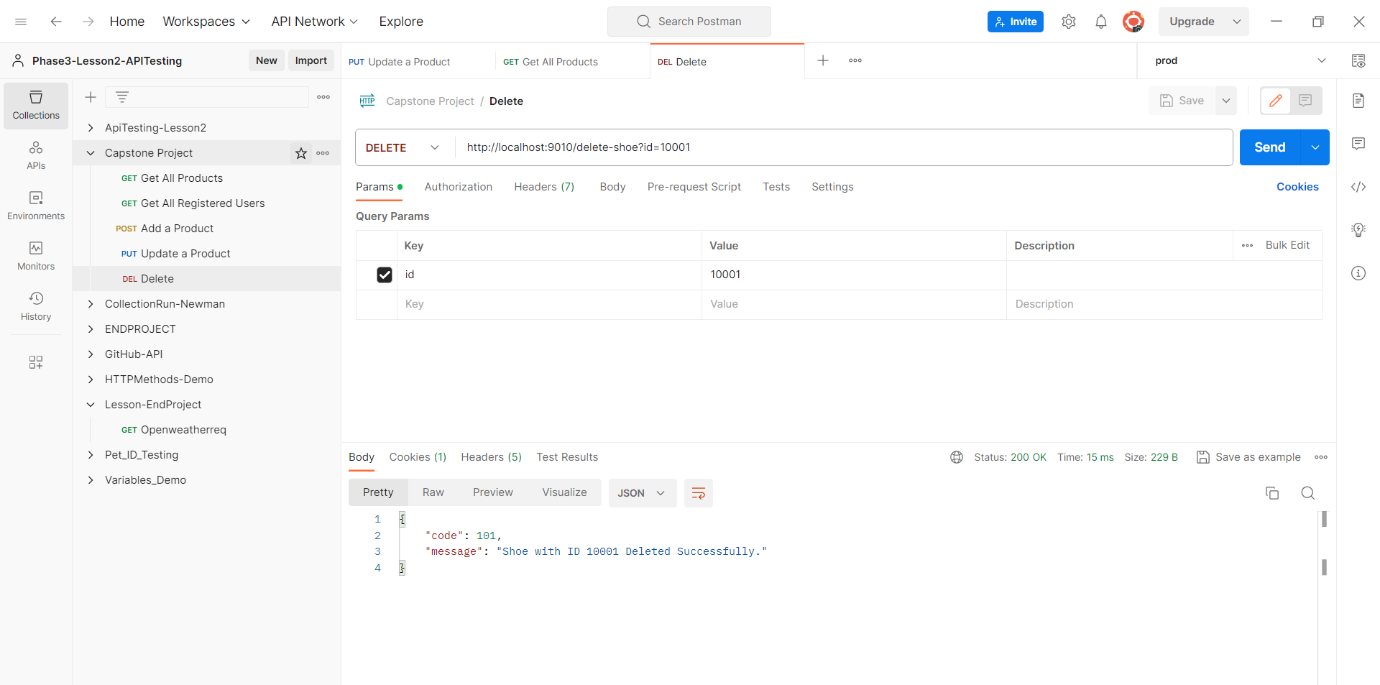


Now we can again run the get command it product details update or not check the run

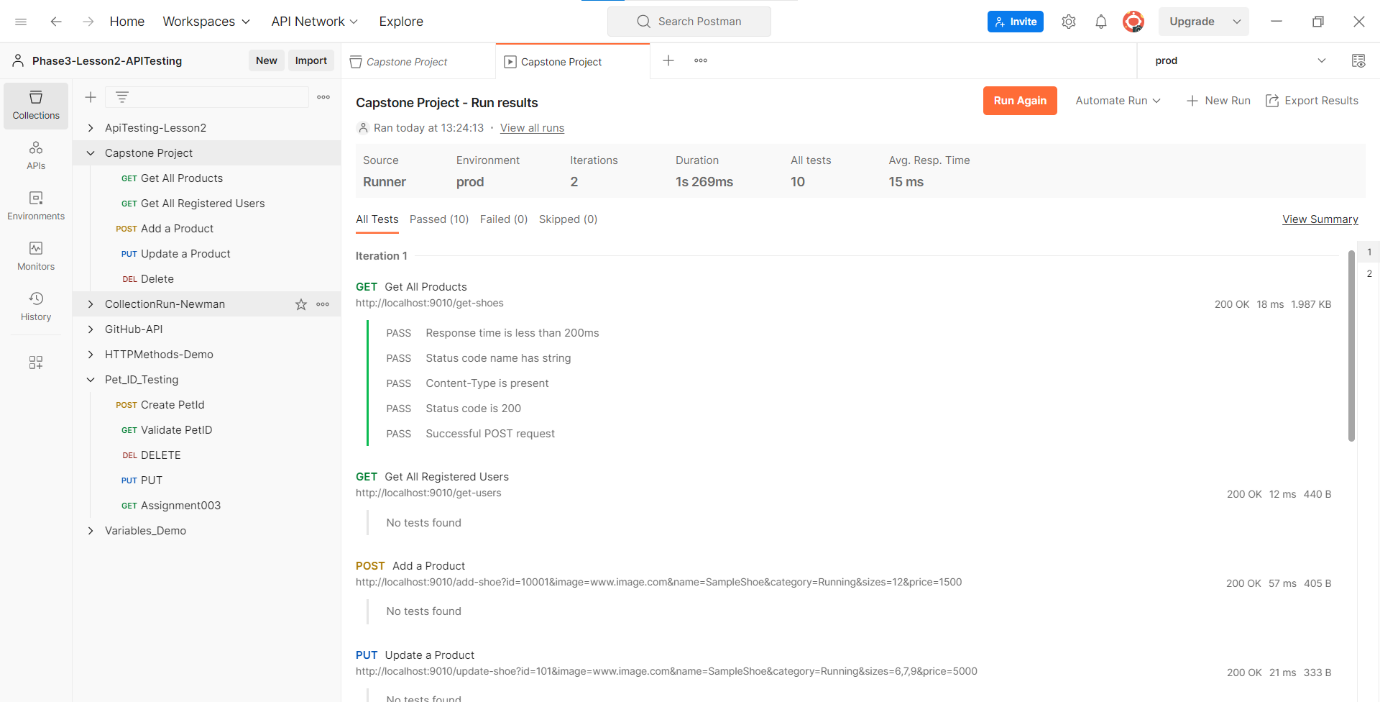
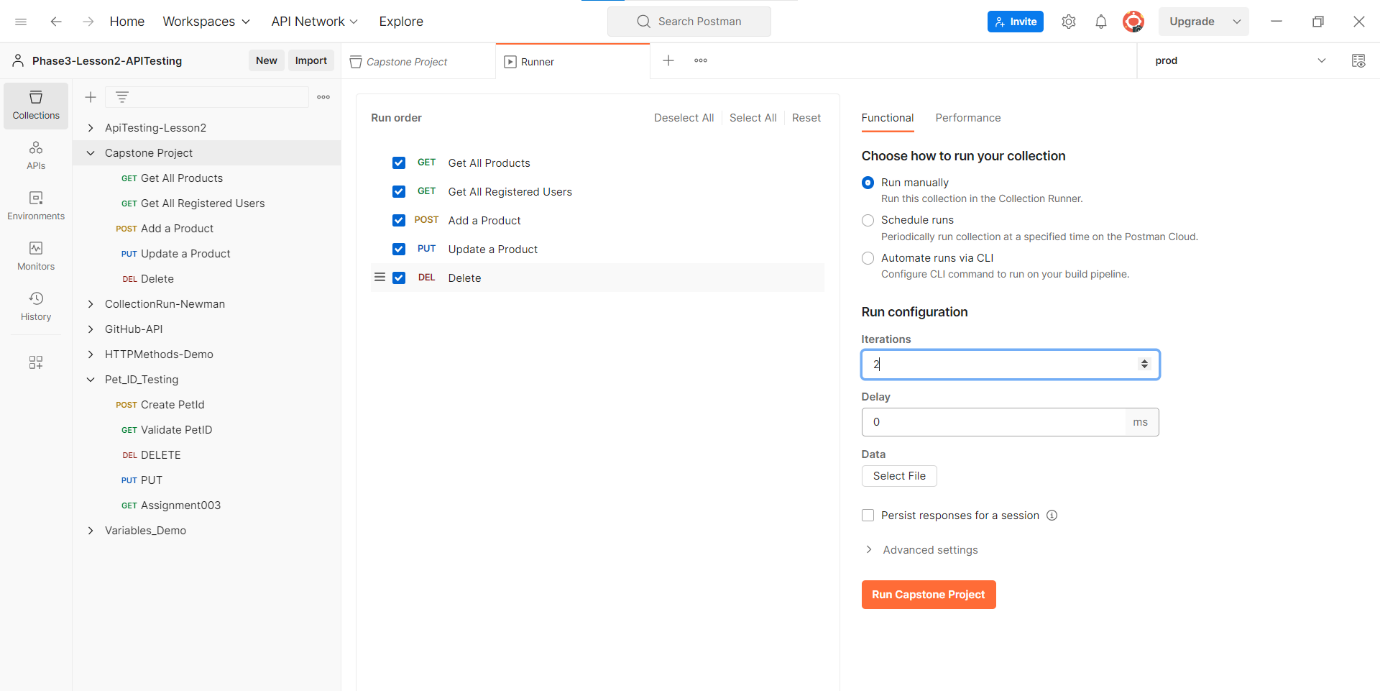


**Step5:** Delete Product URL Sample with DELETE Method

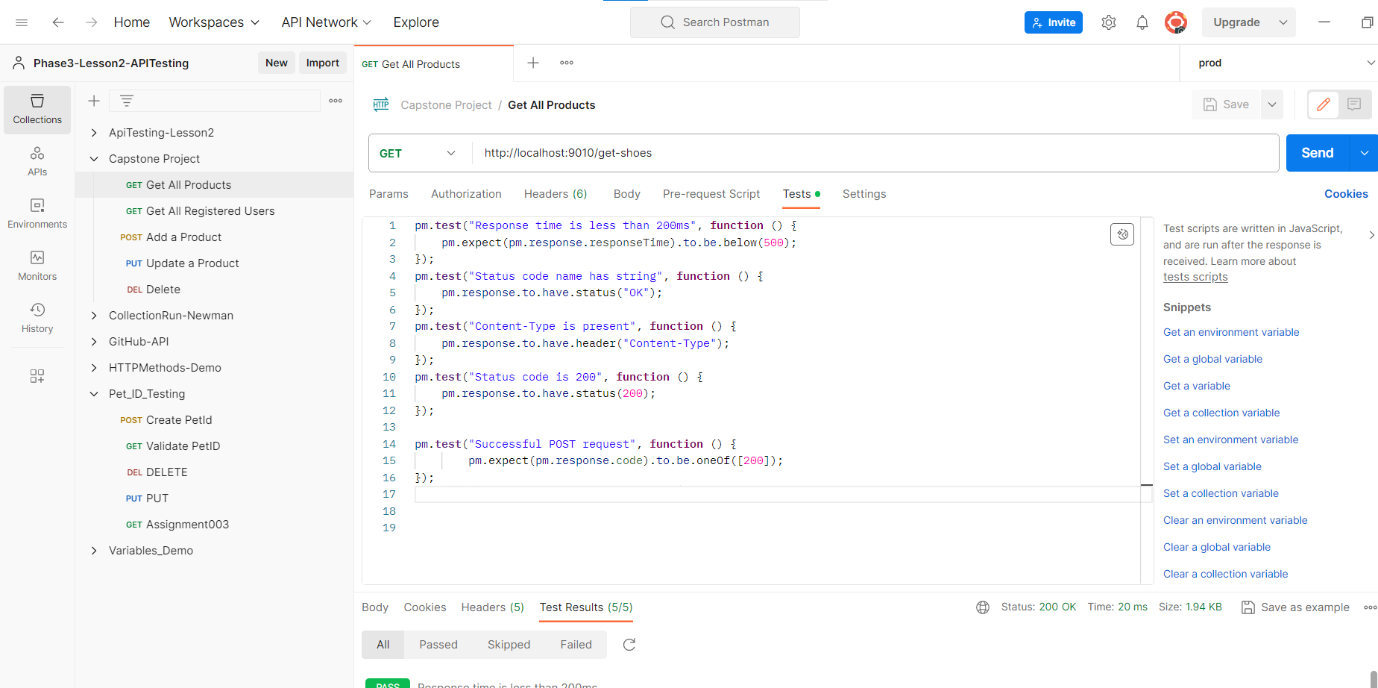
Again, we can add request to delete the product. Now we can use command to delete the product “**http://localhost:9010/delete-shoe?id=10001**” we can use the “**DELETE**” method.

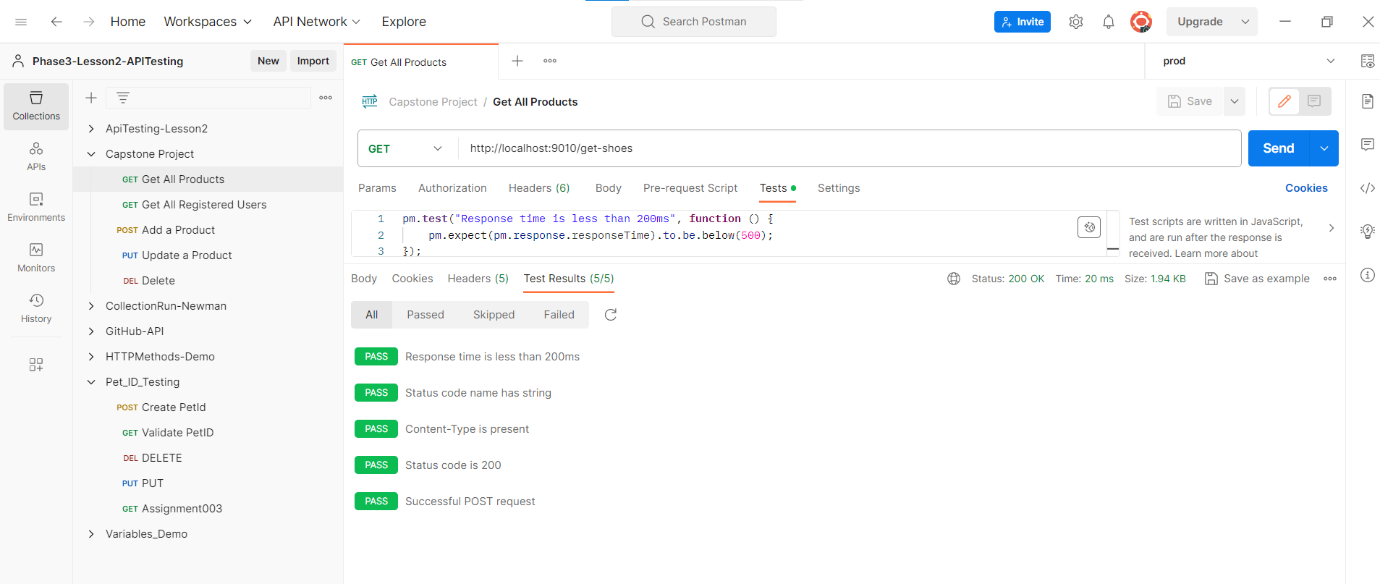
****

In postman we can run all commands now we create, add, update, delete, get all details products of shoes.



And we can run different test scripts in the postman.





**JMeter scripts to do load testing of the homepage and the product detail page.**

Now we can run the home page login and logout user to run with help of blaze meter.



A screenshot of a computer

Description automatically generated

Now we can add the run file and run it.

A screenshot of a computer

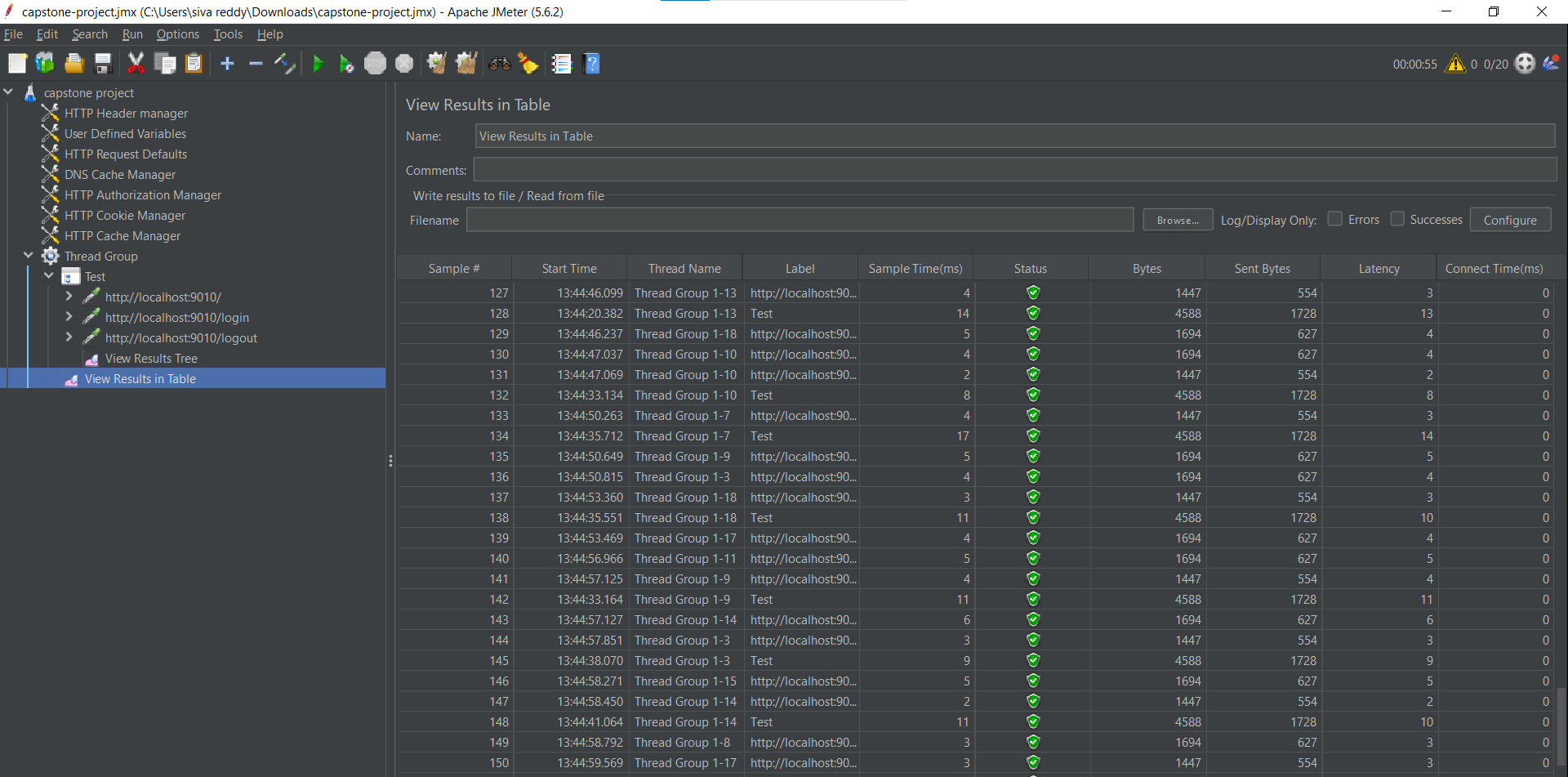
Description automatically generated

Now we can add the load in file increase the users to run the file.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

**Create Selenium scripts using TestNG to test all the pages in the web app that will automate**:

* Login page
* Registration Page
* Add Product to cart page.
* Place Order Page

First, we can add all selenium and cucumber drivers.

1. **HOME PAGE:**

**Code:**

Code for Home page

package com.sportyshoe.SeleniumCucumberScripts;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class HomePage extends TestBaseClass {

@FindBy(xpath="//div[@class='container mt-3']/descendant::p[1]")

WebElement text;

@FindBy(linkText="New User? Register Here")

WebElement registerLink;

public HomePage(WebDriver driver) {

PageFactory.initElements(driver, this);

}

public String getURL\_page()

{

String URLnew = driver.getCurrentUrl();

return URLnew;

}

public String Validate\_Text\_On\_Page()

{

String pageText = text.getText();

System.out.println(pageText);

return pageText;

}

public void click\_register\_link()

{

registerLink.click();

}

}

Code for Home Page Test

package com.sportyshoes.Tests;

import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

import com.sportyshoe.SeleniumCucumberScripts.HomePage;

import com.sportyshoe.SeleniumCucumberScripts.TestBaseClass;

import static org.testng.Assert.assertEquals;

import org.testng.Assert;

import org.testng.Assert.\*;

public class HomePageTest extends TestBaseClass {

HomePage hp;

@BeforeTest

public void start\_browser()

{

OpenBrowser("Chrome");

hp = new HomePage(driver);

}

@Test(priority='1')

public void test\_getTitle\_page()

{

String expected = "http://localhost:9010/";

String Actual = hp.getURL\_page();

Assert.assertEquals(Actual, expected);

}

@Test(priority='2')

public void Test\_Validate\_Text\_On\_Page()

{

String expected = "Powered By Simplilearn";

String actualText = hp.Validate\_Text\_On\_Page();

Assert.assertEquals(actualText, expected);

}

@Test(priority='3')

public void test\_click\_register\_link() throws InterruptedException

{

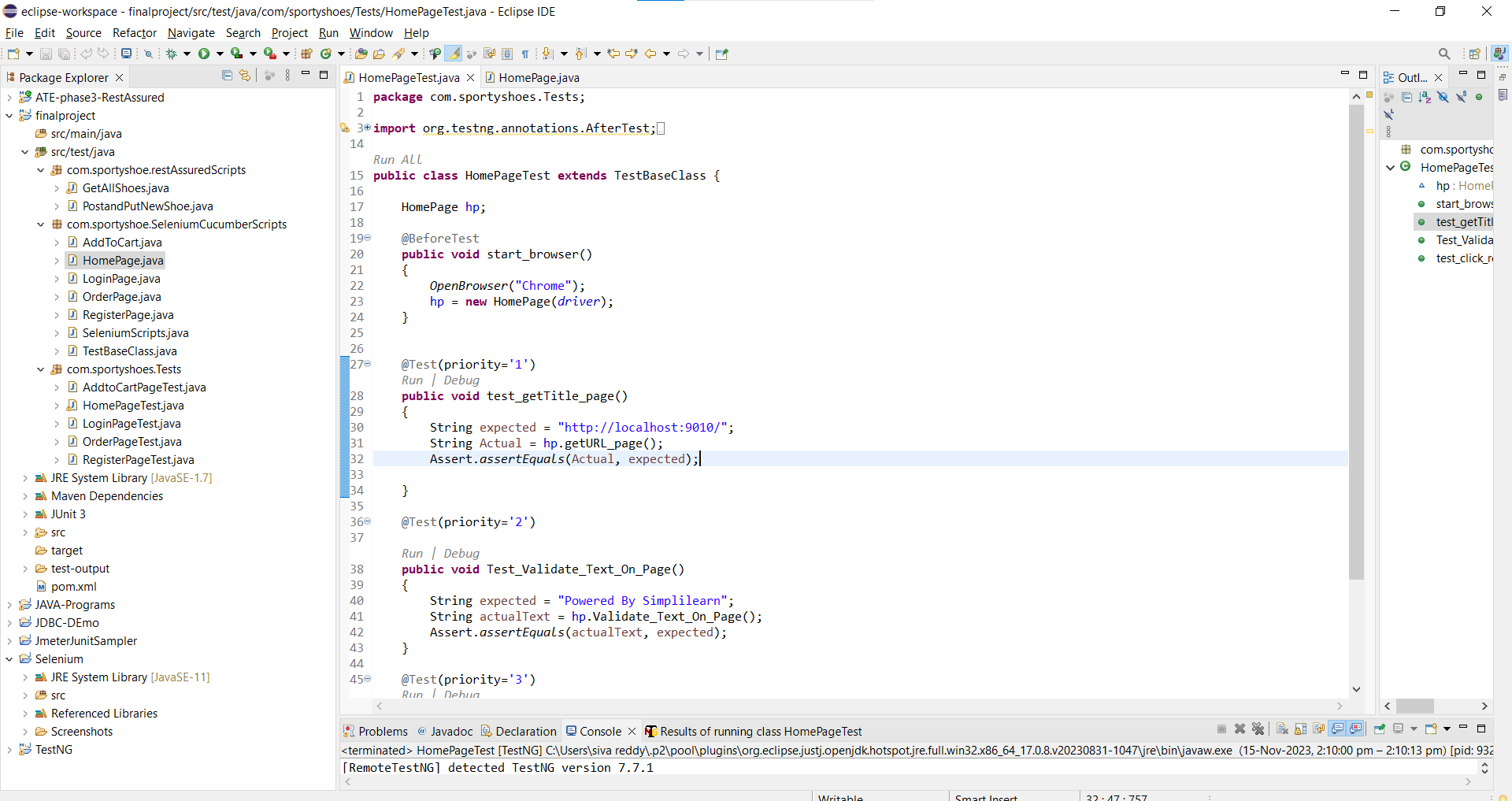
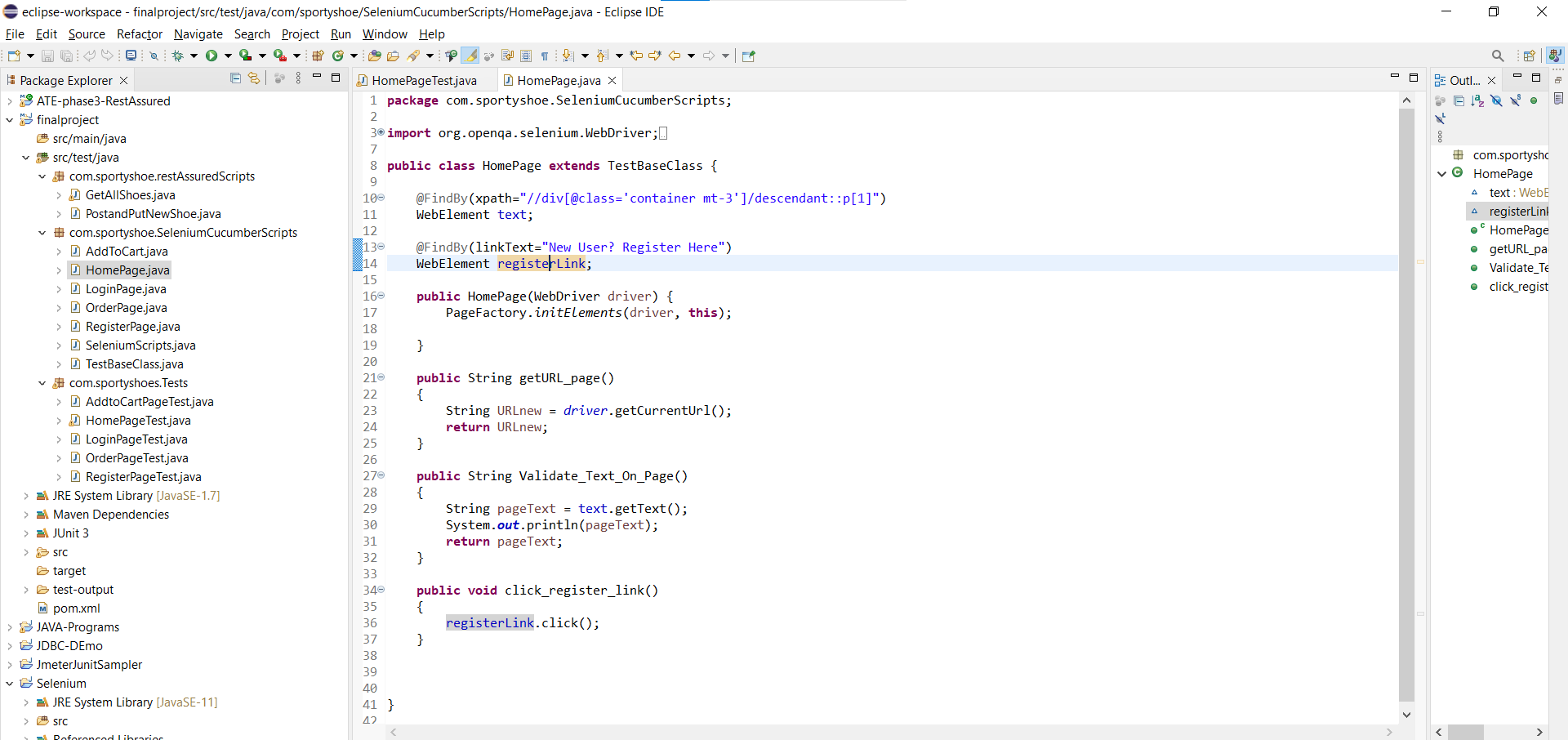
Thread.sleep(1500);

hp.click\_register\_link();

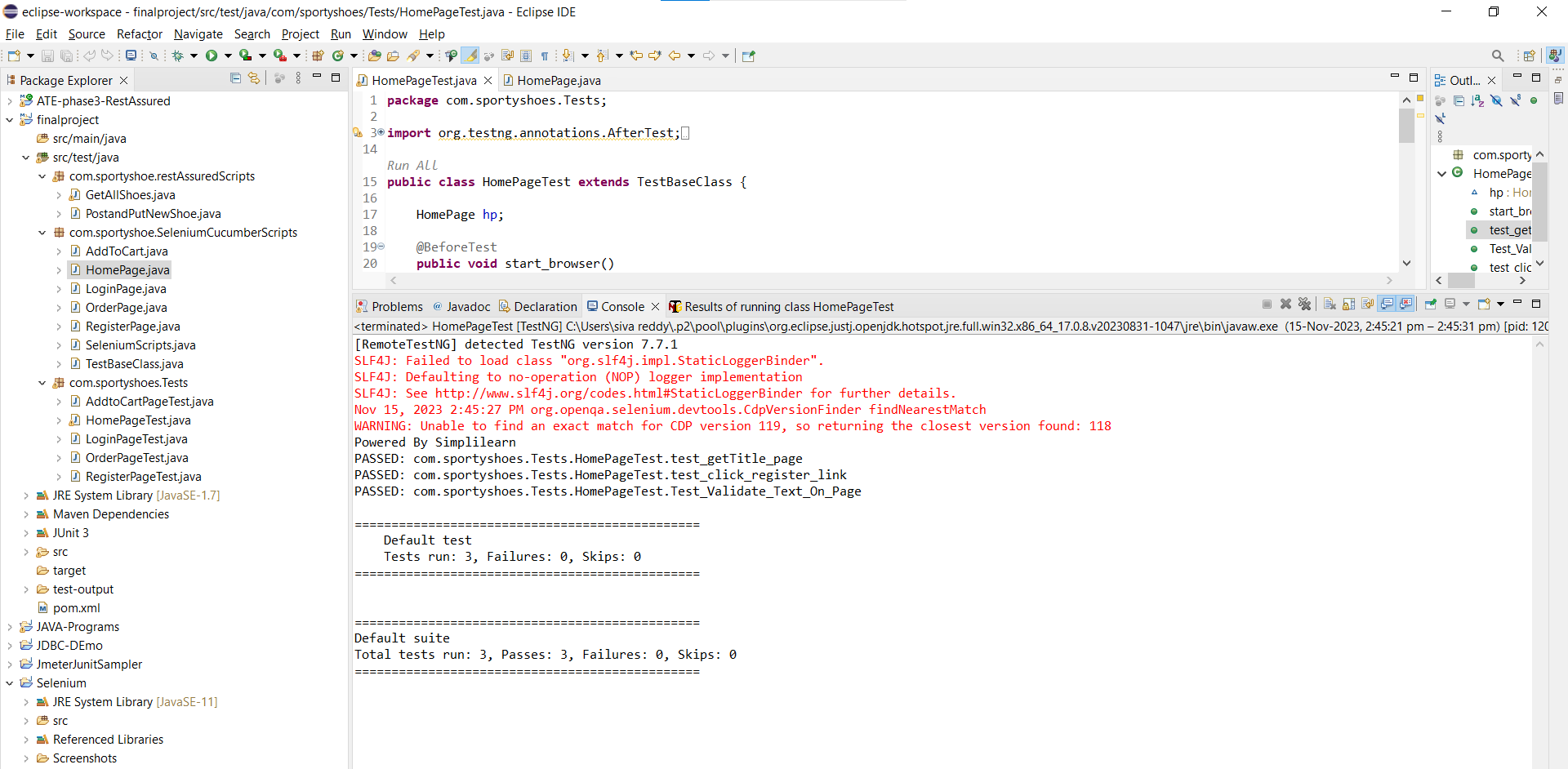
}

}

**OUTPUTS:**

**A blue and white text box

Description automatically generatedA screenshot of a computer

Description automatically generated**

1. **LOGIN PAGE:**

Code for Login page

package com.sportyshoe.SeleniumCucumberScripts;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class LoginPage {

@FindBy(xpath="//input[@id='email']")

WebElement loginEmail;

@FindBy(xpath="//input[@id='password']")

WebElement loginpassword;

@FindBy(xpath="//button[@type='submit']")

WebElement loginbtn;

@FindBy(linkText="Cart")

WebElement clickCart;

public LoginPage(WebDriver driver) {

PageFactory.initElements(driver, this);

}

public void user\_login()

{

loginEmail.sendKeys("siva@gmail.com");

loginpassword.sendKeys("siva@123");

loginbtn.click();

}

public void click\_cart()

{

clickCart.click();

}

}

Code for login page Test

package com.sportyshoes.Tests;

import org.testng.Assert;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

import com.sportyshoe.SeleniumCucumberScripts.HomePage;

import com.sportyshoe.SeleniumCucumberScripts.LoginPage;

import com.sportyshoe.SeleniumCucumberScripts.RegisterPage;

import com.sportyshoe.SeleniumCucumberScripts.TestBaseClass;

public class LoginPageTest extends TestBaseClass {

HomePage hp;

RegisterPage rp;

LoginPage lp;

@BeforeTest

public void start\_browser()

{

OpenBrowser("Chrome");

hp = new HomePage(driver);

rp = new RegisterPage(driver);

lp = new LoginPage(driver);

}

@Test(priority='1')

public void test\_login()

{

lp.user\_login();

}

@Test(priority='2')

public void test\_getTitle\_page()

{

String expected = "http://localhost:9010/login";

String Actual = hp.getURL\_page();

Assert.assertEquals(Actual, expected);

}

@Test(priority='3')

public void Test\_validate\_registration\_Text()

{

String expected = "Hello siva!";

String actualText = rp.validate\_registration\_Text();

Assert.assertEquals(actualText, expected);

}

@Test(priority='4')

public void test\_click\_cart()

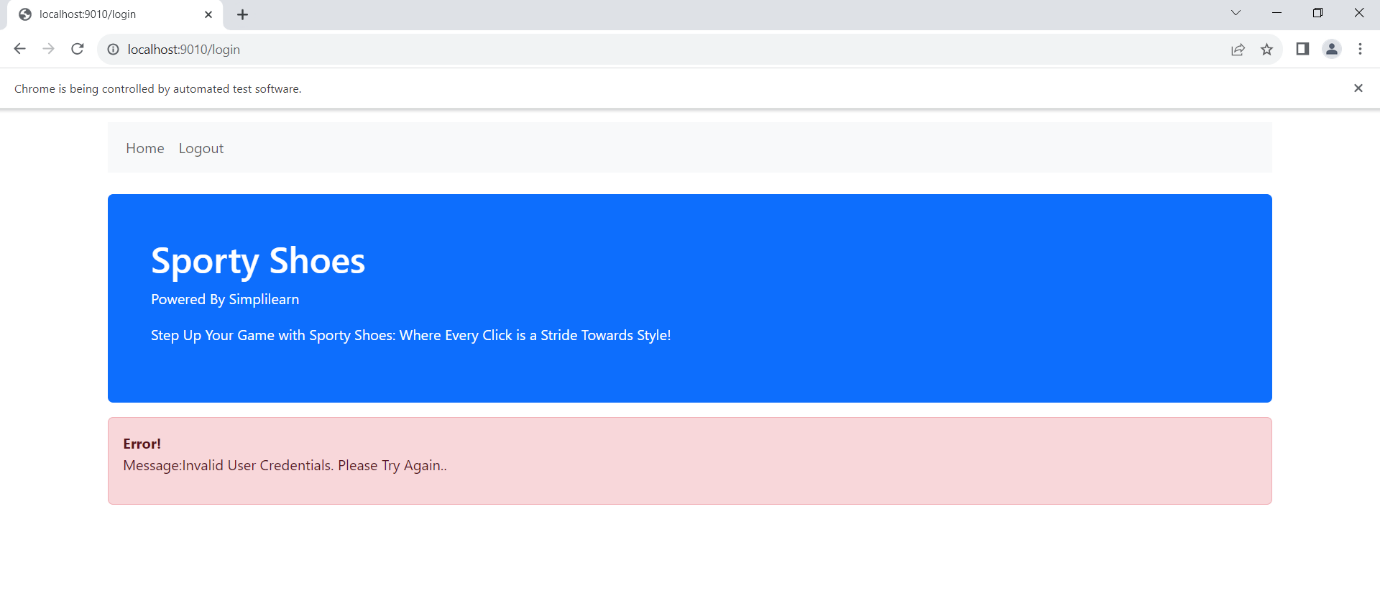
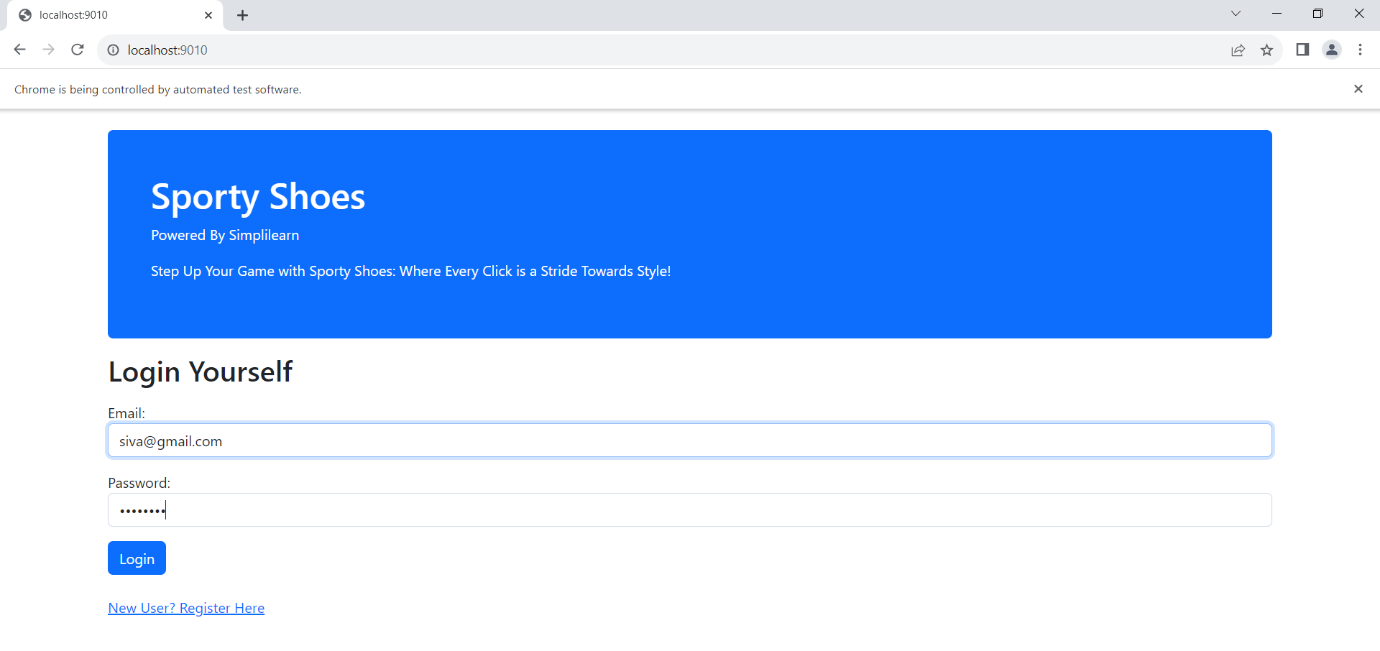
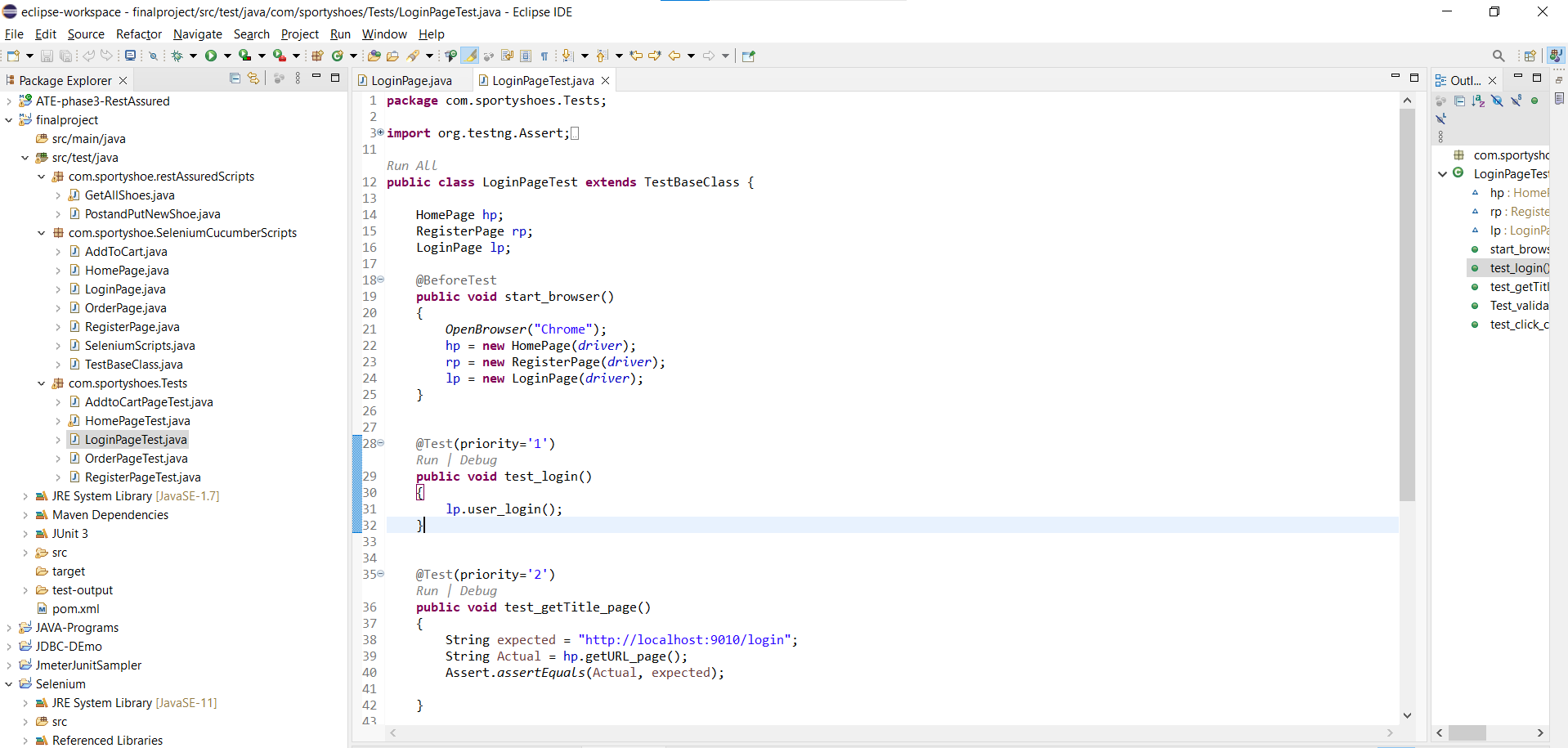
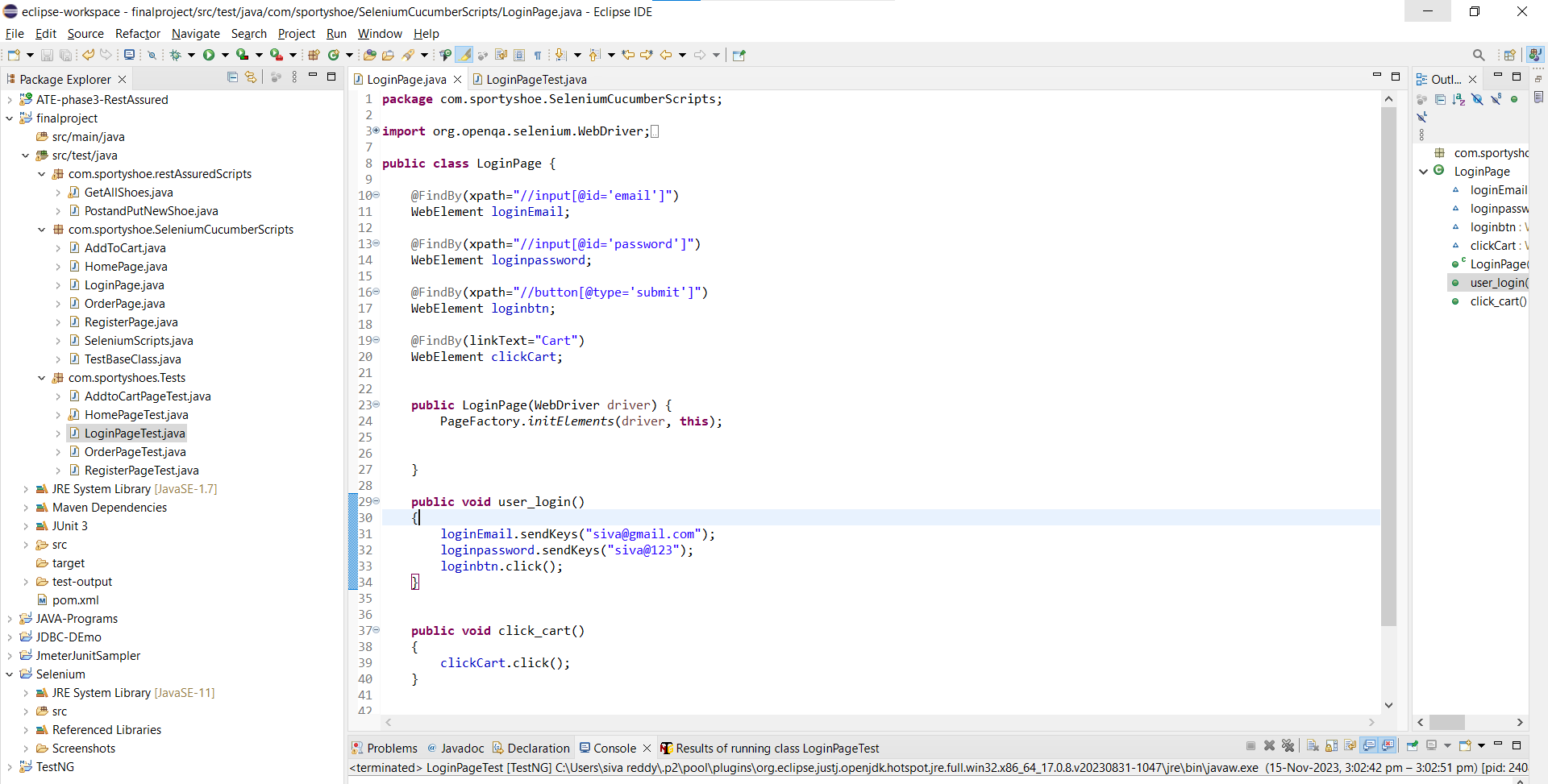
{

lp.click\_cart();

}

}

**OUTPUTS:**

****

1. **Register Page:**

Code for register page

package com.sportyshoe.SeleniumCucumberScripts;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class RegisterPage extends TestBaseClass{

@FindBy(xpath="//input[@id='name']")

WebElement registername;

@FindBy(xpath="//input[@id='email']")

WebElement registeremail;

@FindBy(xpath="//input[@id='password']")

WebElement registerpassword;

@FindBy(xpath="//button[@type='submit']")

WebElement registerBtn;

@FindBy(xpath="//div[@class='mt-4 p-5 bg-primary text-white rounded']/descendant::p[3]")

WebElement userText;

public RegisterPage(WebDriver driver) {

PageFactory.initElements(driver, this);

}

public void register\_user()

{

registername.sendKeys("siva");

registeremail.sendKeys("siva@gmail.com");

registerpassword.sendKeys("siva@123");

registerBtn.click();

}

public String validate\_registration\_URL()

{

String register\_url = driver.getCurrentUrl();

return register\_url;

}

public String validate\_registration\_Text()

{

String user\_name = userText.getText();

return user\_name;

}

}

Code for Register Page Test

package com.sportyshoes.Tests;

import static org.testng.Assert.assertEquals;

import org.testng.Assert;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

import com.sportyshoe.SeleniumCucumberScripts.HomePage;

import com.sportyshoe.SeleniumCucumberScripts.RegisterPage;

import com.sportyshoe.SeleniumCucumberScripts.TestBaseClass;

public class RegisterPageTest extends TestBaseClass {

HomePage hp;

RegisterPage rp;

@BeforeTest

public void start\_browser()

{

OpenBrowser("Chrome");

hp = new HomePage(driver);

rp = new RegisterPage(driver);

}

@Test(priority='1')

public void test\_click\_register\_link() throws InterruptedException

{

Thread.sleep(1500);

hp.click\_register\_link();

}

@Test(priority='2')

public void test\_getTitle\_page()

{

String expected = "http://localhost:9010/register";

String Actual = hp.getURL\_page();

Assert.assertEquals(Actual, expected);

}

@Test(priority='3')

public void Test\_register\_user()

{

rp.register\_user();

}

@Test(priority='4')

public void Test\_validate\_registration\_URL()

{

String expected = "http://localhost:9010/register-user";

String Actual = rp.validate\_registration\_URL();

assertEquals(Actual, expected);

}

@Test(priority='5')

public void Test\_validate\_registration\_Text()

{

String expected = "Hello siva !";

String actualText = rp.validate\_registration\_Text();

Assert.assertEquals(actualText, expected);

}

}

**OUTPUTS:**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA blue and white website with a blue and white background

Description automatically generated

1. **Add To Cart**

Code for add to cart.

package com.sportyshoe.SeleniumCucumberScripts;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class AddToCart {

@FindBy(xpath="//a[@id=\"shoe101\"]")

WebElement viewShoeBTN;

@FindBy(xpath = "//a[@id='cart101']")

WebElement addtocartBTN;

@FindBy(xpath="//div[@class='alert alert-success']/descendant::p[1]")

WebElement successText;

JavascriptExecutor executor;

public AddToCart(WebDriver driver) {

PageFactory.initElements(driver, this);

executor = (JavascriptExecutor) driver;

}

public void add\_product\_to\_cart() throws InterruptedException

{

executor.executeScript("arguments[0].click();", addtocartBTN);

}

public String validate\_success\_message()

{

String successtext = successText.getText();

return successtext;

}

}

Code for add to cart test.

package com.sportyshoes.Tests;

import org.testng.Assert;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

import com.sportyshoe.SeleniumCucumberScripts.AddToCart;

import com.sportyshoe.SeleniumCucumberScripts.HomePage;

import com.sportyshoe.SeleniumCucumberScripts.LoginPage;

import com.sportyshoe.SeleniumCucumberScripts.RegisterPage;

import com.sportyshoe.SeleniumCucumberScripts.TestBaseClass;

public class AddtoCartPageTest extends TestBaseClass {

HomePage hp;

RegisterPage rp;

LoginPage lp;

AddToCart ac;

@BeforeTest

public void start\_browser()

{

OpenBrowser("Chrome");

hp = new HomePage(driver);

rp = new RegisterPage(driver);

lp = new LoginPage(driver);

ac = new AddToCart(driver);

}

@Test(priority='1')

public void test\_login()

{

lp.user\_login();

}

@Test(priority='2')

public void test\_add\_product\_to\_cart() throws InterruptedException

{

ac.add\_product\_to\_cart();

}

@Test(priority='3')

public void test\_validate\_success\_message()

{

String expected = "Message:Shoe BlueWave Running Shoes Added Successfully to Cart";

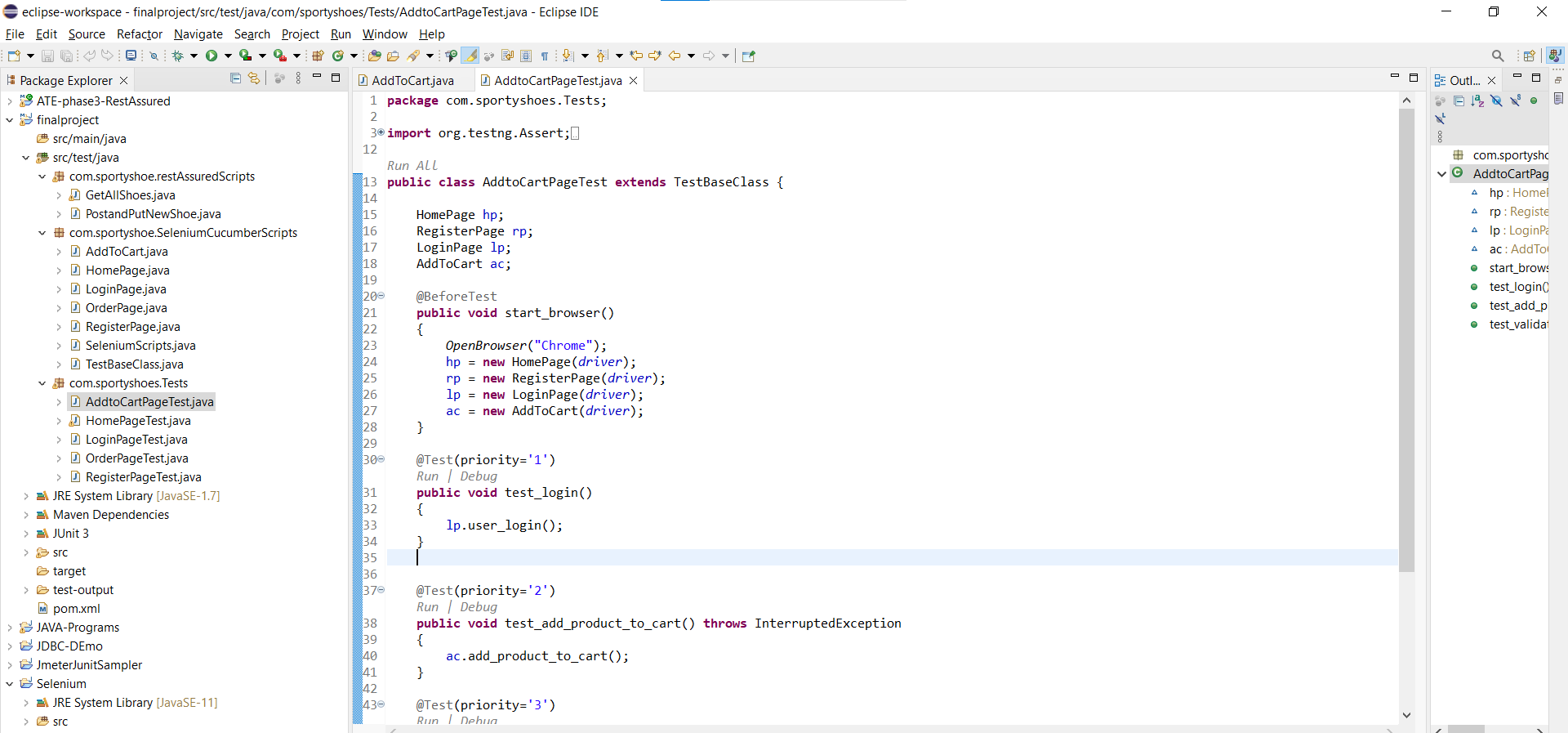
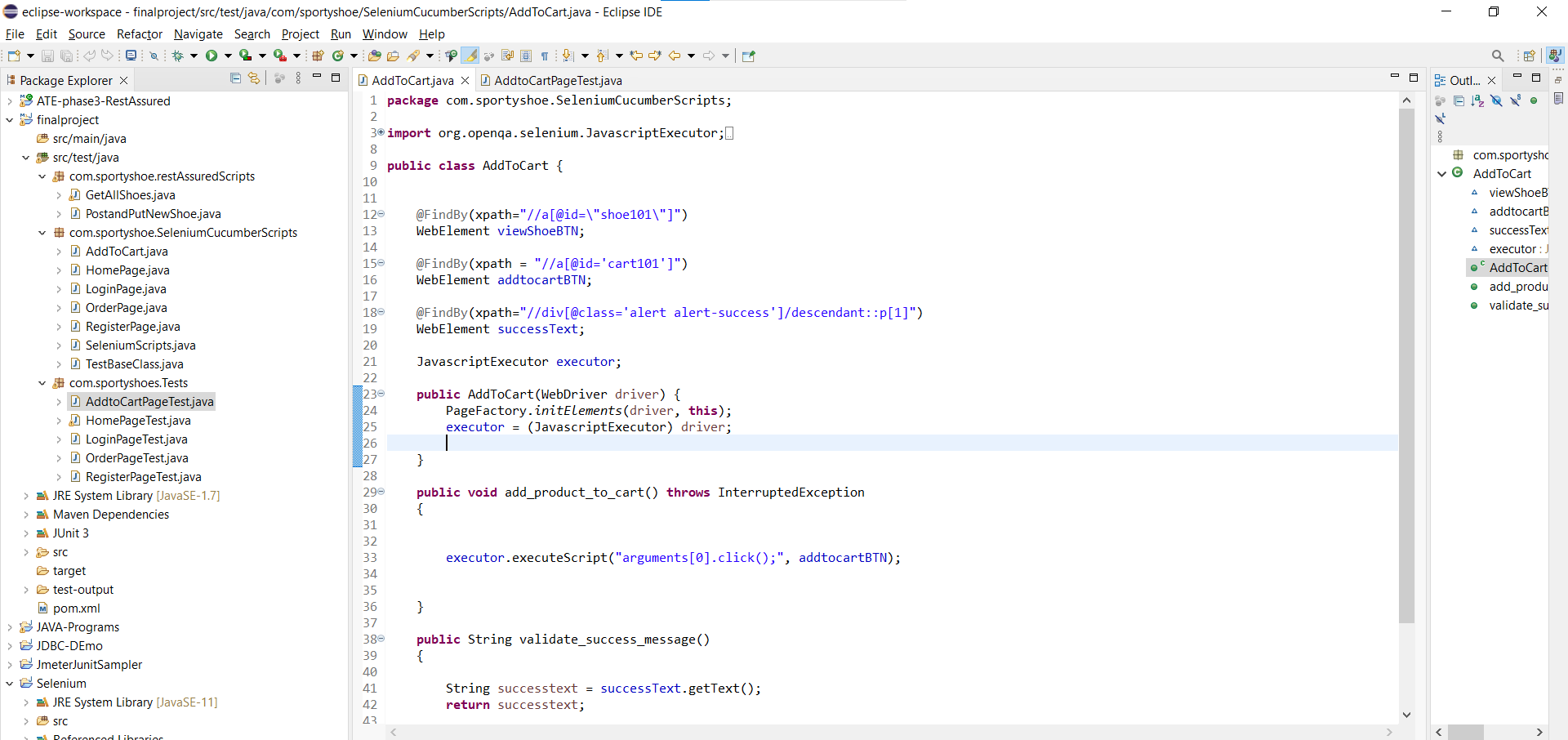
String actualText= ac.validate\_success\_message();

Assert.assertEquals(actualText, expected);

}

}

**OUTPUTS:**



1. **ORDER PAGE:**

Code for order

package com.sportyshoe.SeleniumCucumberScripts;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class OrderPage {

@FindBy(linkText="Orders")

WebElement orderlink;

public OrderPage(WebDriver driver) {

PageFactory.initElements(driver, this);

}

public void click\_orderPage()

{

orderlink.click();

}

}

Code for order test

package com.sportyshoes.Tests;

import org.testng.Assert;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

import com.sportyshoe.SeleniumCucumberScripts.HomePage;

import com.sportyshoe.SeleniumCucumberScripts.LoginPage;

import com.sportyshoe.SeleniumCucumberScripts.OrderPage;

import com.sportyshoe.SeleniumCucumberScripts.RegisterPage;

import com.sportyshoe.SeleniumCucumberScripts.TestBaseClass;

public class OrderPageTest extends TestBaseClass {

HomePage hp;

RegisterPage rp;

LoginPage lp;

OrderPage op;

@BeforeTest

public void start\_browser()

{

OpenBrowser("Chrome");

hp = new HomePage(driver);

rp = new RegisterPage(driver);

lp = new LoginPage(driver);

op = new OrderPage(driver);

}

@Test(priority='1')

public void test\_login()

{

lp.user\_login();

}

@Test(priority='2')

public void test\_click\_orders()

{

op.click\_orderPage();

}

@Test(priority='3')

public void test\_getTitle\_page()

{

String expected = "http://localhost:9010/orders";

String Actual = hp.getURL\_page();

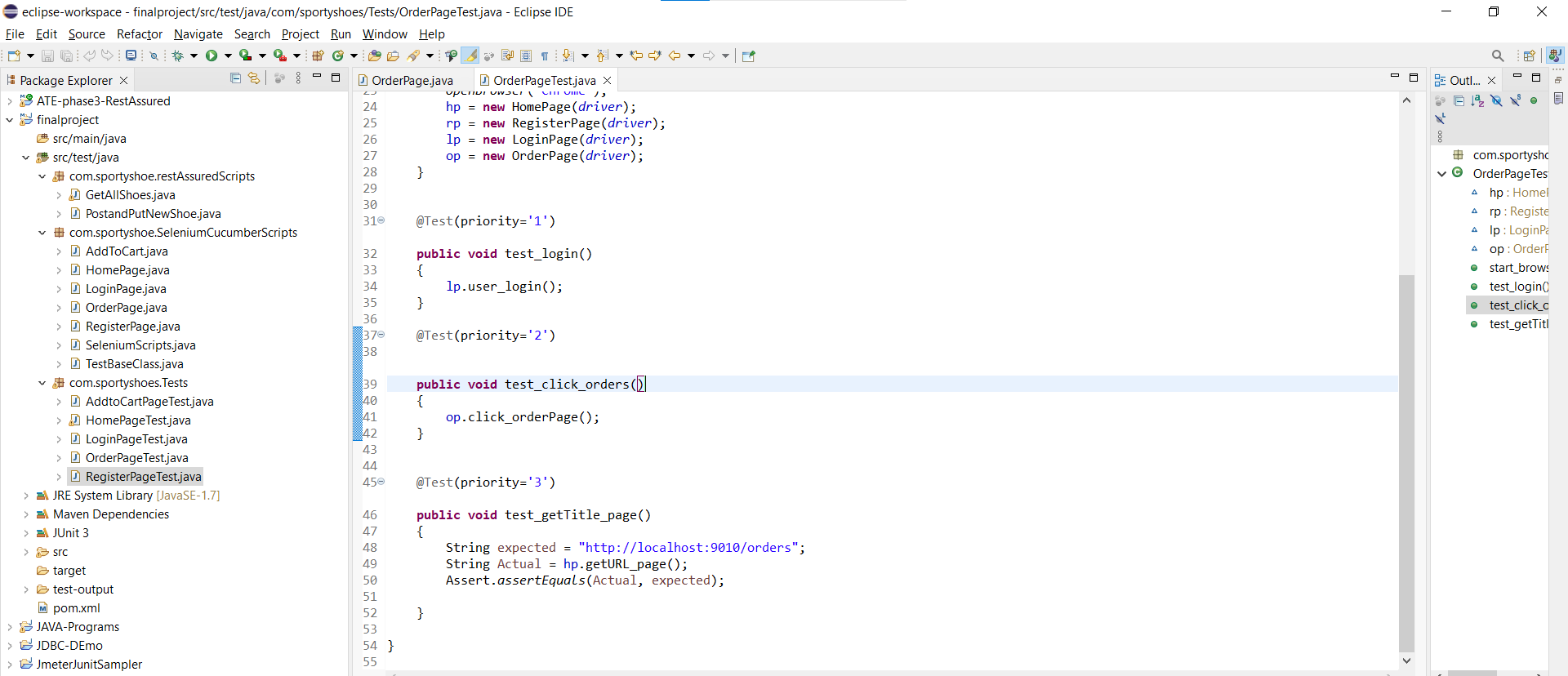
Assert.assertEquals(Actual, expected);

}

}

**OUTPUTS:**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**Automate the below API endpoints using Rest-Assured**

Now we retrieve all products and all registered users in rest assured.

**Code**

**package** com.sportyshoe.restAssuredScripts;

**import** **static** org.hamcrest.Matchers.*equalTo*;

**import** **static** org.hamcrest.Matchers.*hasItems*;

**import** org.json.JSONObject;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** io.restassured.http.ContentType;

**public** **class** GetAllShoes {

@Test (priority='1')

**public** **void** get\_all\_shoes()

{

RestAssured.*given*()

.baseUri("http://localhost:9010")

.basePath("/get-shoes")

.when()

.get()

.then()

.statusCode(200)

.log()

.all();

}

@Test(priority='3')

**public** **void** get\_all\_users()

{

RestAssured.*given*()

.baseUri("http://localhost:9010")

.basePath("/get-users")

.when()

.get()

.then()

.statusCode(200)

.log()

.all();

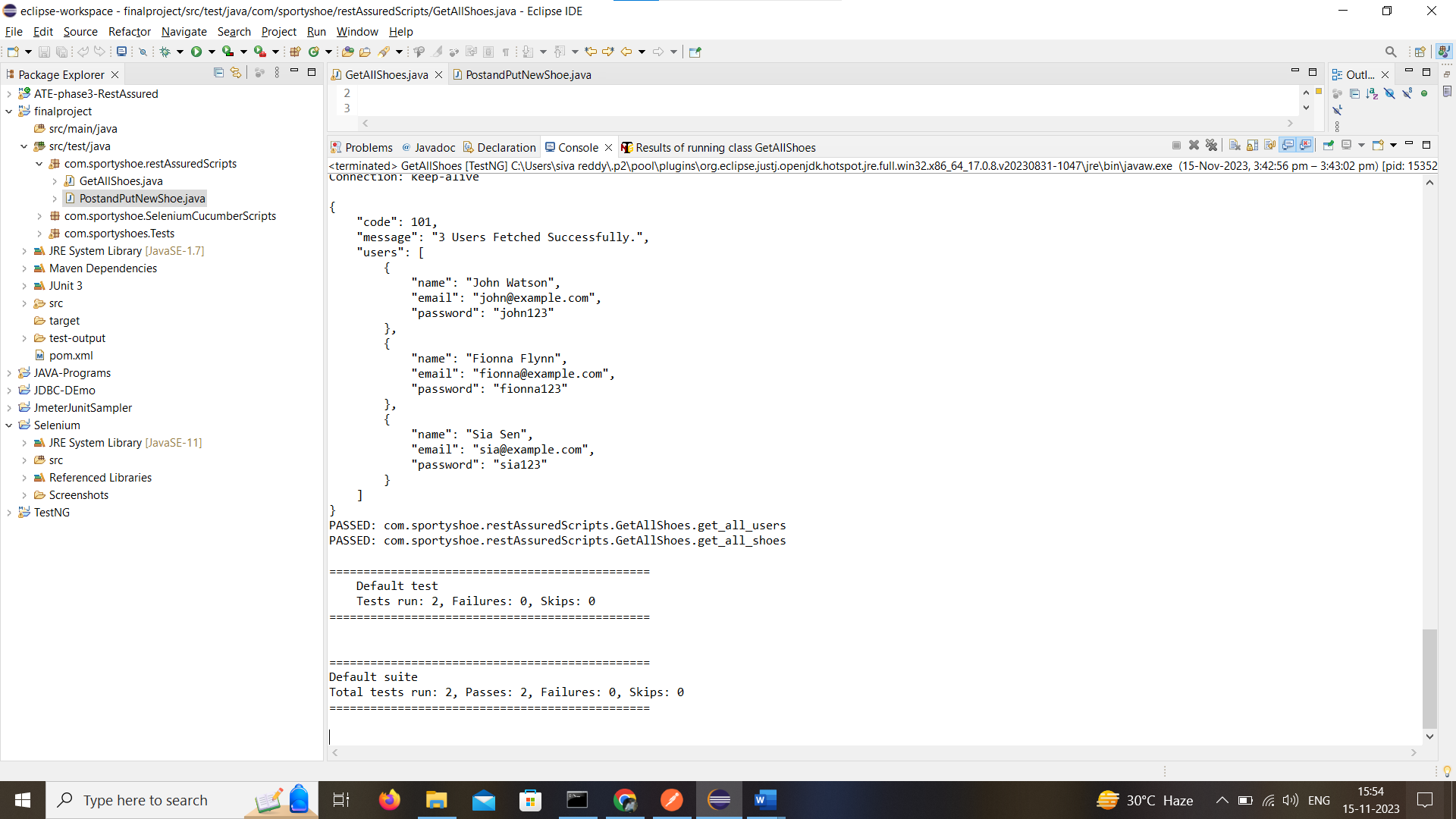
}

}

**OUTPUTS:**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

Code for add delete product.

package com.sportyshoe.restAssuredScripts;

import static org.hamcrest.Matchers.equalTo;

import org.testng.annotations.Test;

import io.restassured.RestAssured;

public class PostandPutNewShoe {

@Test(priority='1')

public void add\_new\_product()

{

RestAssured.given()

.baseUri("http://localhost:9010")

.basePath("/add-shoe")

.queryParam("id","1020")

.queryParam("image", "www.imge.com")

.queryParam("name","Nike")

.queryParam("category", "Running")

.queryParam("sizes","5,6,7")

.queryParam("price", "2000")

.when()

.post()

.then()

.log().all();

}

@Test(priority='2')

public void update\_a\_product()

{

RestAssured.given()

.baseUri("http://localhost:9010")

.basePath("/update-shoe")

.queryParam("id","1020")

.queryParam("image", "www.imge123.com")

.queryParam("name","Reebok")

.queryParam("category", "Running")

.queryParam("sizes","5,6,7")

.queryParam("price", "2500")

.when()

.put()

.then()

.log().all();

}

@Test(priority='3')

public void delete\_product()

{

RestAssured.given()

.baseUri("http://localhost:9010")

.basePath("/delete-shoe")

.queryParam("id", "1020")

.when().delete()

.then().log().all();

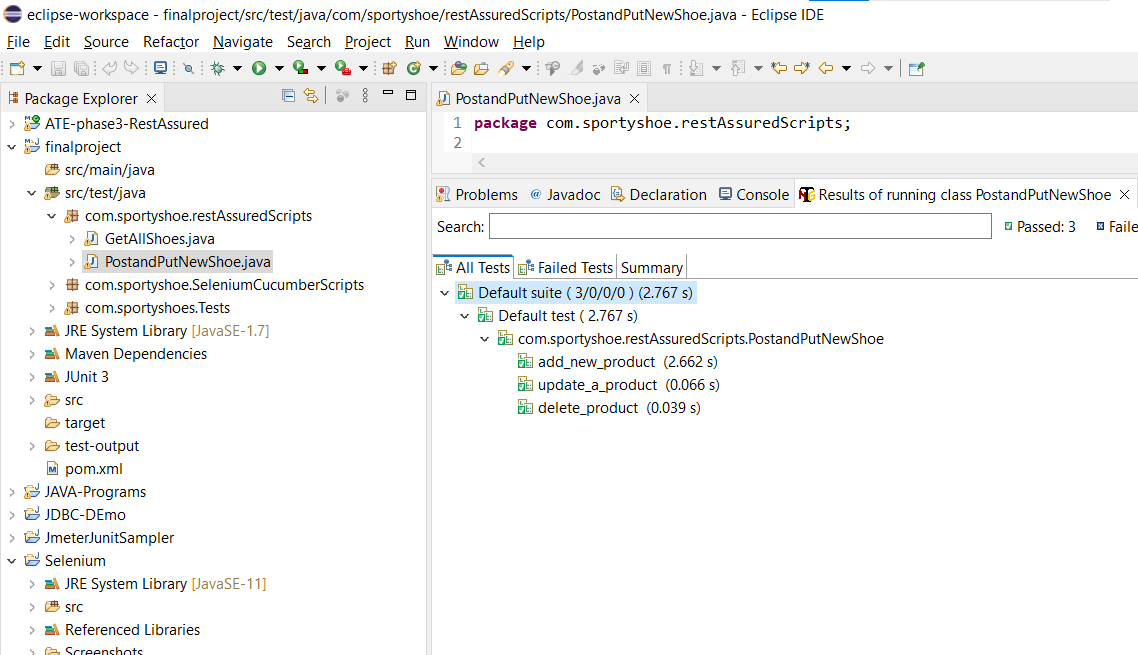
}

}

**OUTPUT:**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

Now we can run the feature file and step file in cucumber.

Code for feature file

Feature: Implement The Lesson End Project1.

Scenario: Rest API testing on reqres.in Automate the below API endpoints using Rest-Assured

Given User get Retrieve the list of all products in the store.

Given User Retrieve the list of all registered users.

Then User Add the product.

When User Delete the product.

And User Update the product.

Code for Step file

**package** steps;

**import** org.json.JSONObject;

**import** io.cucumber.java.en.And;

**import** io.cucumber.java.en.Given;

**import** io.cucumber.java.en.Then;

**import** io.cucumber.java.en.When;

**import** io.restassured.RestAssured;

**import** io.restassured.http.ContentType;

**public** **class** Final {

@Given("User get Retrieve the list of all products in the store.")

**public** **void** user\_get\_retrieve\_the\_list\_of\_all\_products\_in\_the\_store() {

JSONObject body = **new** JSONObject();

body.put("name", "Siva");

body.put("job", "Tester");

RestAssured.*given*()

.baseUri("http://localhost:9010")

.basePath("/get-shoes")

.when().get()

.then().statusCode(200).log().all();

}

@Given("User Retrieve the list of all registered users.")

**public** **void** user\_retrieve\_the\_list\_of\_all\_registered\_users() {

RestAssured.*given*()

.baseUri("http://localhost:9010")

.basePath("/get-users")

.when().get()

.then().statusCode(200).log().all();

}

@Then("User Add the product.")

**public** **void** user\_add\_the\_product() {

RestAssured.*given*()

.baseUri("http://localhost:9010/add-shoe?id=10001&image=www.image.com&name=SampleShoe&category=Running&sizes=12&price=1500")

.when().get()

.then().statusCode(405);

}

@When("User Delete the product.")

**public** **void** user\_delete\_the\_product() {

RestAssured.*given*()

.baseUri("http://localhost:9010")

.basePath("/delete-shoe?id=10001")

.when().get()

.then().statusCode(405);

}

@And("User Update the product.")

**public** **void** user\_update\_the\_product() {

RestAssured.*given*()

.baseUri("http://localhost:9010/add-shoe?id=10001&image=www.image.com&name=SampleShoe&category=Running&sizes=12&price=1500")

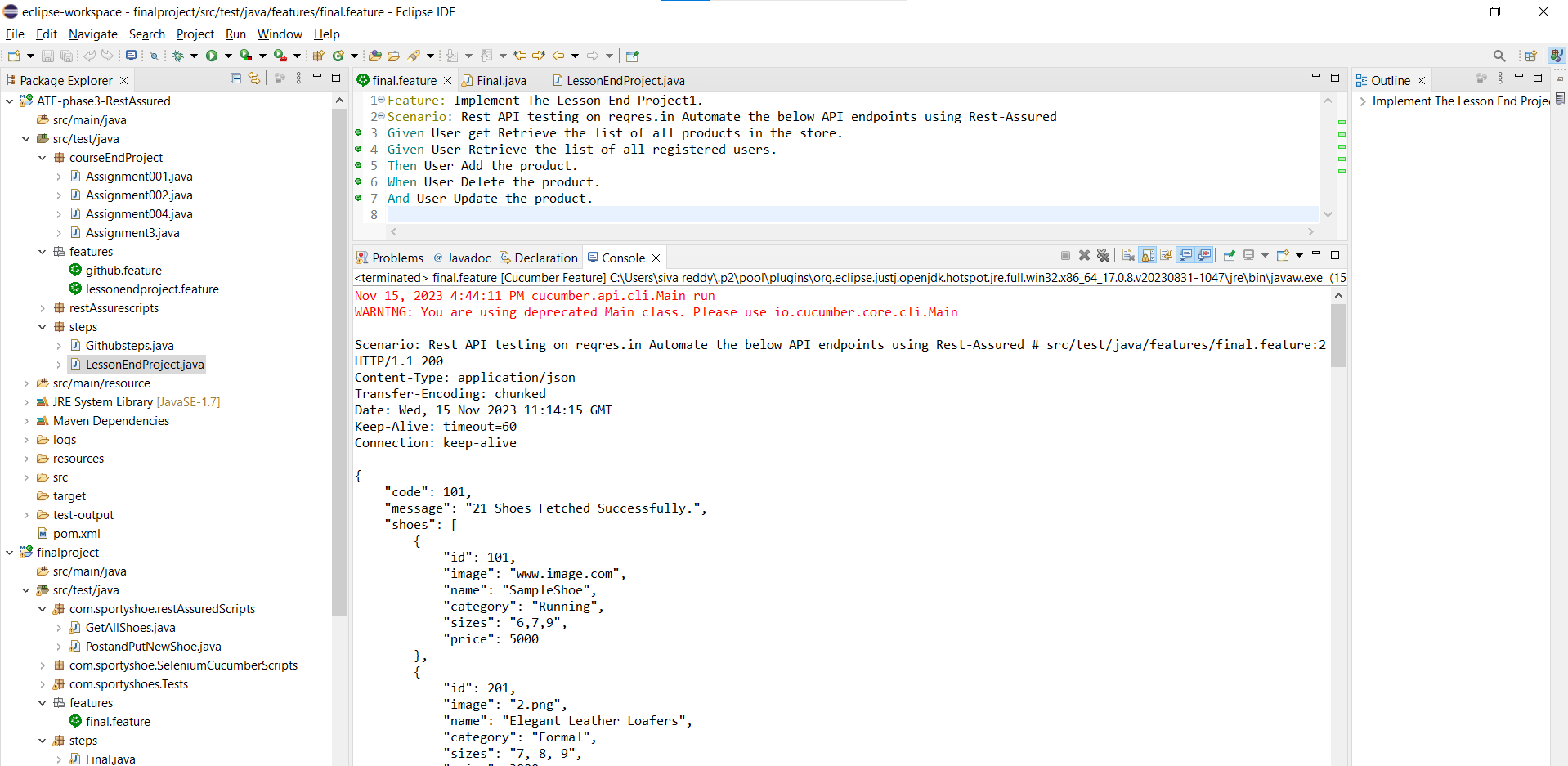
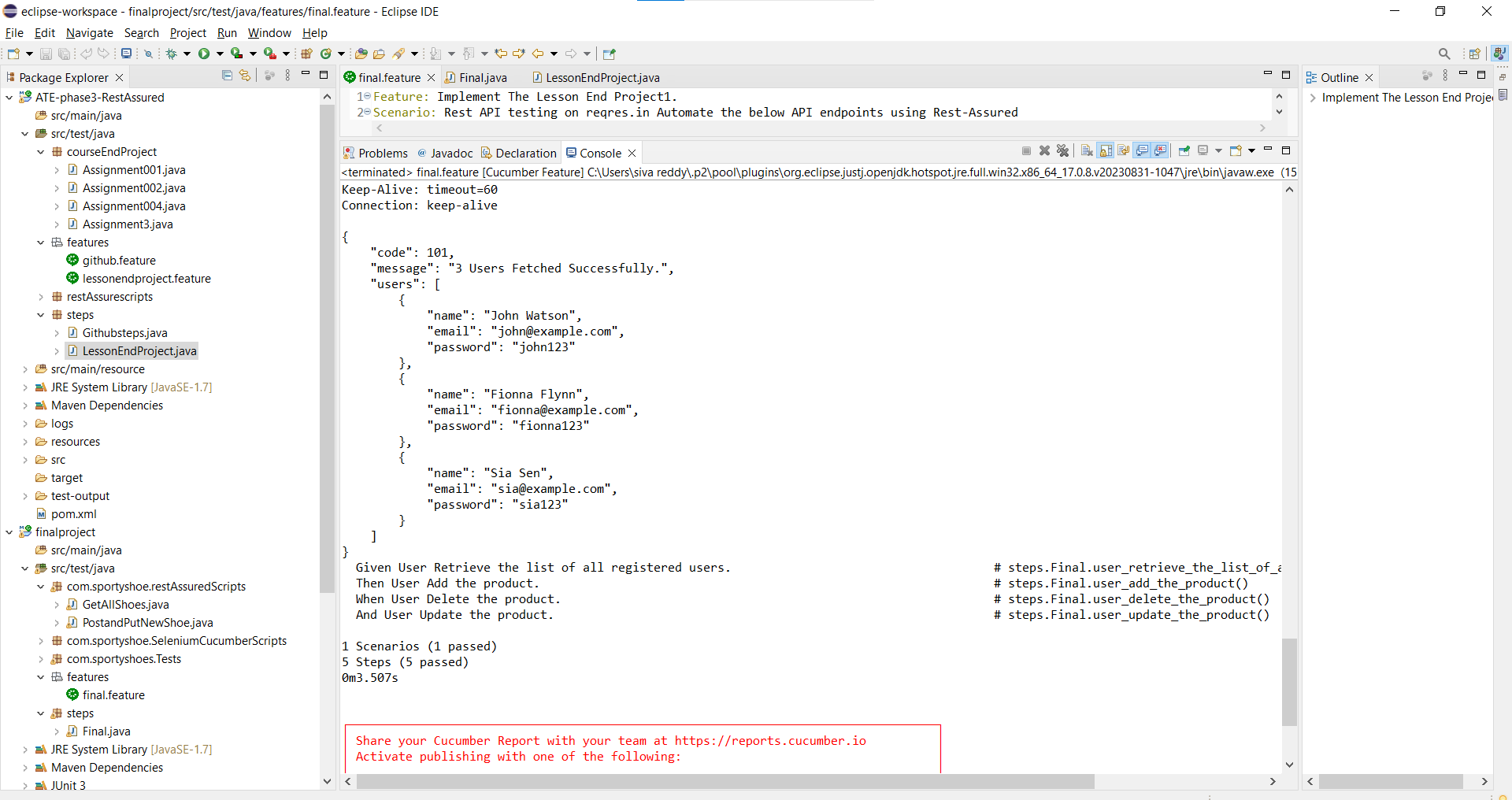
.when().get()

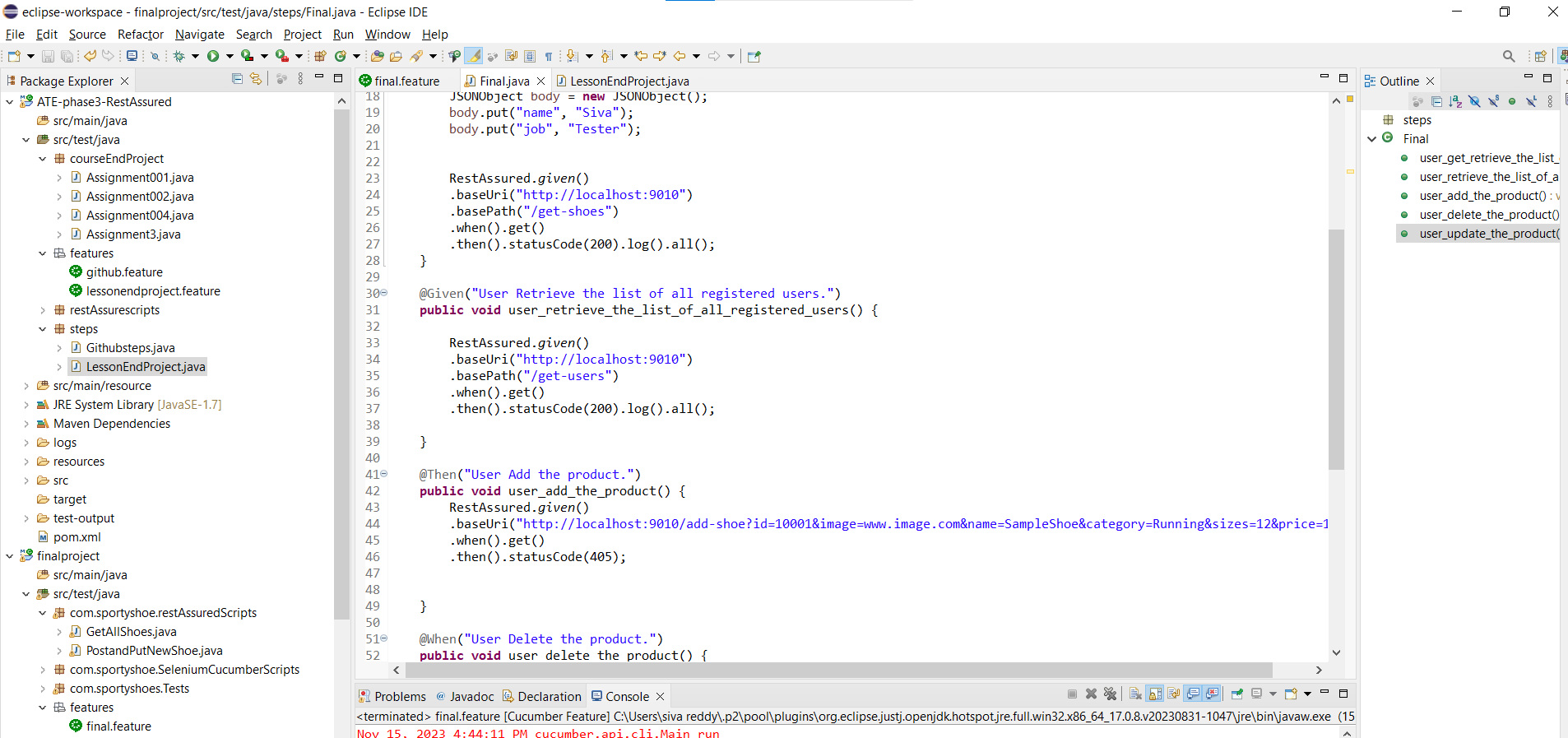
.then().statusCode(405);

}

}

**OUTPUTS:**

****

****