Sourcecode

1. **Browser-based end-user testing using Selenium WebDriver with TestNG Framework.**

- SetupCheck

```
package com.sportyshoe.SeleniumCucumberScripts;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;


public class SetupCheck {


	@Test

	public void test_setup_selenium()
	{
		WebDriver driver = new ChromeDriver();

		driver.manage().window().maximize();

		driver.get("http://localhost:9010/");

		String text =
driver.findElement(By.xpath("//div[@class='container mt-
3']/descendant::p[1]")).getText();

		System.out.println(text);

		System.out.println(driver.getTitle());
	}

}
```

- Homepage

```
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 TestBase {

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

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

	private WebDriver driver;

	public HomePage(WebDriver driver) {
		PageFactory.initElements(driver, this);

	}

	public String getURL_page()
	{
		driver = null;
		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();
	}



}
```

- HomepageTest

```
package com.sportyshoe.Tests;

import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.Test;

import com.sportyshoe.SeleniumCucumberScripts.HomePage;
```

```java
import com.sportyshoe.SeleniumCucumberScripts.TestBase;

import static org.testng.Assert.assertEquals;

import org.testng.Assert;
import org.testng.Assert.*;


public class HomePageTest extends TestBase {

	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();
	}

}
```

- Loginpage

```
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("sonal@gmail.com");
		loginpassword.sendKeys("sonal@123");
		loginbtn.click();
	}


	public void click_cart()
	{
		clickCart.click();
	}




}
```

- LoginpageTest

```
package com.sportyshoe.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.TestBase;


public class LoginPageTest extends TestBase {

	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 sonal !";
```

```
            String actualText = rp.validate_registration_Text();
            Assert.assertEquals(actualText, expected);
      }


@Test(priority='4')


public void test_click_cart()
{
      lp.click_cart();
}


}
```

- RegisterPage

```
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 TestBase{

      @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;

      private WebDriver driver;


      public RegisterPage(WebDriver driver) {
            PageFactory.initElements(driver, this);
```

```
	}

	public void register_user()
	{
		registername.sendKeys("sonal");
		registeremail.sendKeys("sonal@gmail.com");
		registerpassword.sendKeys("sonal@123");
		registerBtn.click();
	}

	public String validate_registration_URL()
	{
		driver = null;
		String register_url = driver.getCurrentUrl();
		return register_url;
	}

	public String validate_registration_Text()
	{
		String user_name = userText.getText();
		return user_name;
	}

}
```

- RegisterPageTest

```
package com.sportyshoe.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.TestBase;


public class RegisterPageTest extends TestBase {

	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 sonal !";
		String actualText = rp.validate_registration_Text();
		Assert.assertEquals(actualText, expected);
	}
```

```
}
```

- AddtoCart

```
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 AddtoCartPage {


	@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 AddtoCartPage(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 successstext = successText.getText();
		return successstext;
```

```
	}


}
```

- AddtoCartPageTest

```
package com.sportyshoe.Tests;

import org.testng.Assert;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.Test;

import com.sportyshoe.SeleniumCucumberScripts.AddtoCartPage;
import com.sportyshoe.SeleniumCucumberScripts.HomePage;
import com.sportyshoe.SeleniumCucumberScripts.LoginPage;
import com.sportyshoe.SeleniumCucumberScripts.RegisterPage;
import com.sportyshoe.SeleniumCucumberScripts.TestBase;


public class AddtoCartPageTest extends TestBase {

	HomePage hp;
	RegisterPage rp;
	LoginPage lp;
	AddtoCartPage ac;

	@BeforeTest
	public void start_browser()
	{
		OpenBrowser("Chrome");
		hp = new HomePage(driver);
		rp = new RegisterPage(driver);
		lp = new LoginPage(driver);
		ac = new AddtoCartPage(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);
	}
}
```

- OrderPage

```
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();

	}

}
```

- OrderPageTest

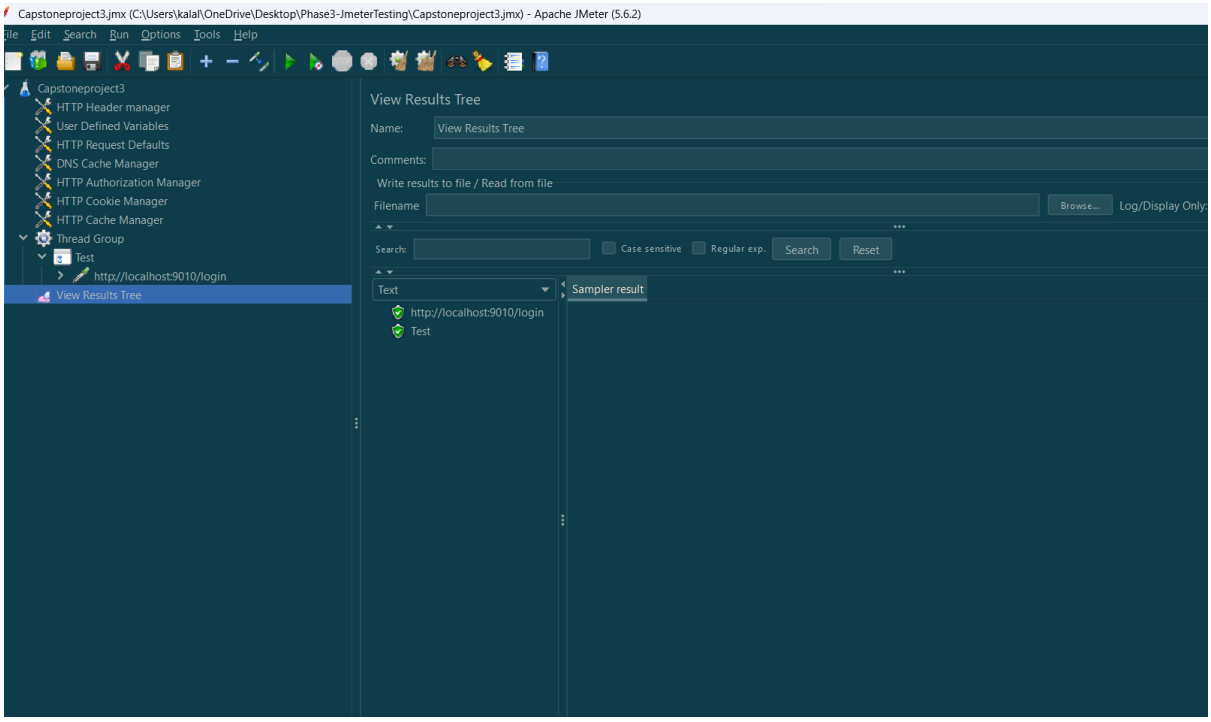
```java
package com.sportyshoe.Tests;

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.TestBase;


public class OrderpageTest  extends TestBase {

	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();
```

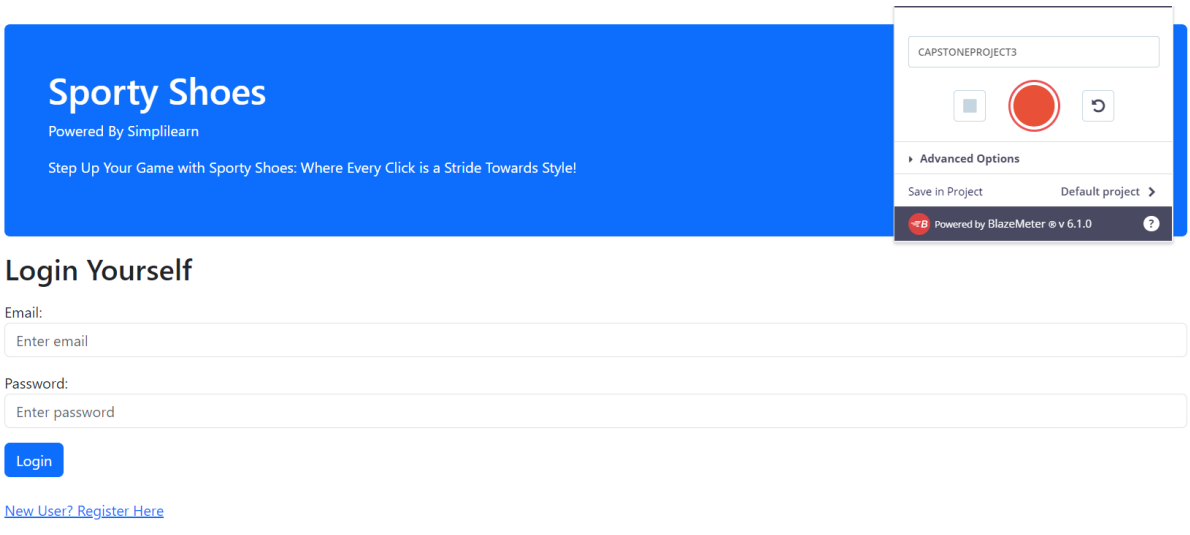
```
            Assert.assertEquals(Actual, expected);

      }

}
```

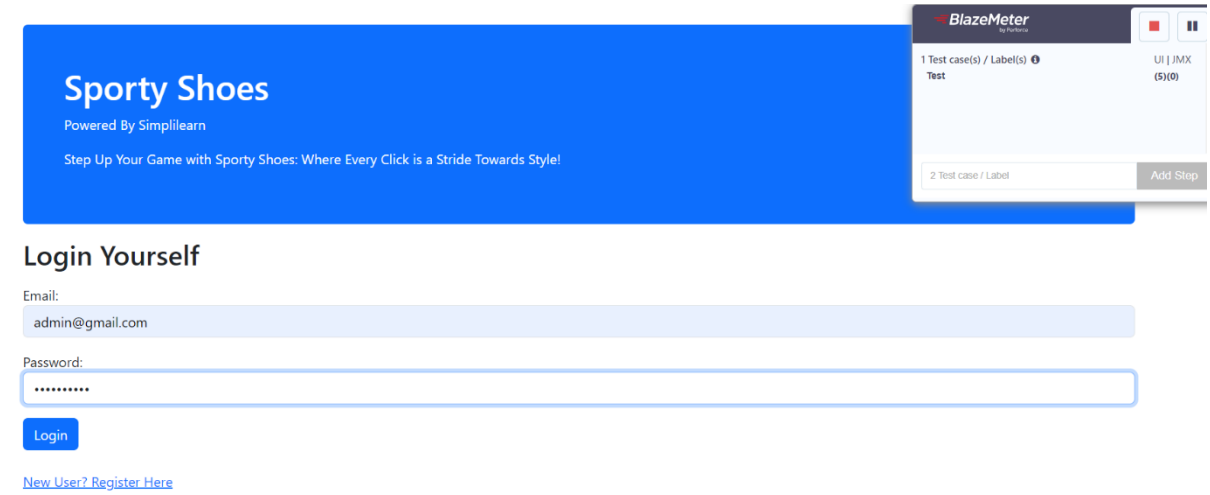
- TestBase

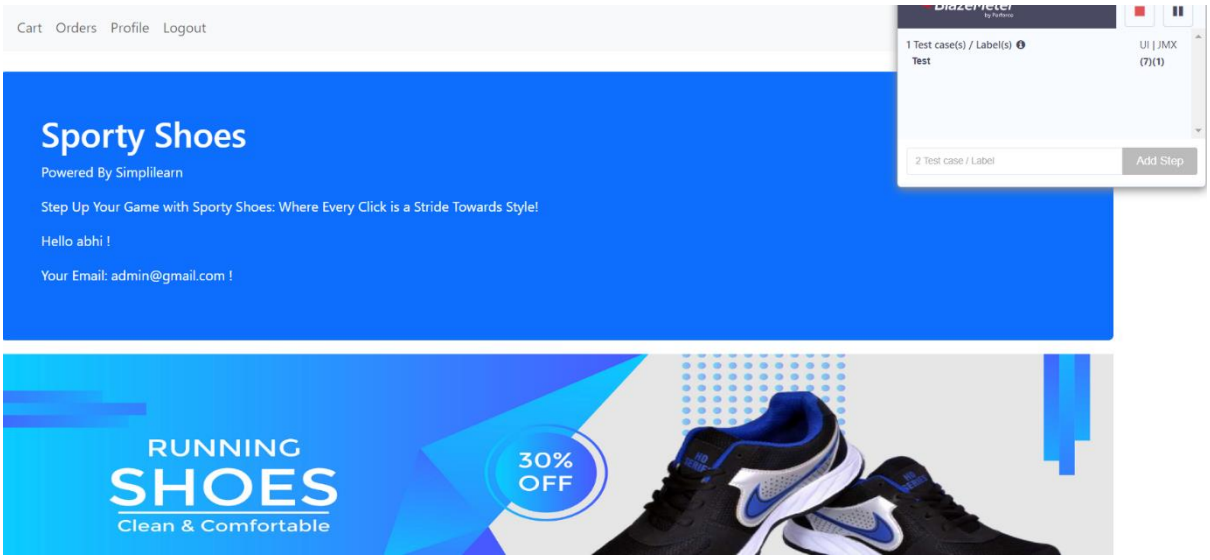
```
package com.sportyshoe.SeleniumCucumberScripts;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;



public class TestBase {

      public static WebDriver driver;

      public static void  OpenBrowser(String browser)
      {
            if(browser == "Chrome")
            {
                  driver = new ChromeDriver();
            }

            if(browser == "FireFox")
            {
                  driver = new FirefoxDriver();
            }

            driver.manage().window().maximize();
            driver.manage().deleteAllCookies();
            driver.get("http://localhost:9010/");

      }

      public static void closebrowser()
      {
            driver.close();
      }


}
```

2.Load Testing using JMeter.









3.API Testing with Cucumber.

```
Feature: Product API Tests

Scenario: Retrieve the list of all products
    Given I send a GET request to "/get-shoes"
    When I receive a response from the server
    Then the response status code should be 200
    And the response should contain a list of products

Scenario: Retrieve the list of all registered users
    Given I send a GET request to "/get-users"
    When I receive a response from the server
    Then the response status code should be 200
    And the response should contain a list of registered users
```

```
Scenario: Add a product
    Given I send a POST request to "/add-shoe" with the following
JSON body:json
    {
        "id": 102,
        "image": "image_url",
        "name": "NewShoe",
        "category": "Running",
        "sizes": "9,10,11",
        "price": 1200
    }

    When I receive a response from the server
    Then the response status code should be 201
    And the response should contain the newly added product

Scenario: Delete a product
    Given I send a DELETE request to "/delete-shoe?id=102"
    When I receive a response from the server
    Then the response status code should be 204

Scenario: Update a product
    Given I send a PUT request to "/update-shoe" with the following
JSON body:
    json
    {
        "id": 102,
        "name": "UpdatedShoe",
        "category": "Basketball",
        "sizes": "8,9,10",
        "price": 1500
        "image": "updated_image_url"
    }

    When I receive a response from the server
    Then the response status code should be 200
    And the response should contain the updated product
```

PROGRAM:

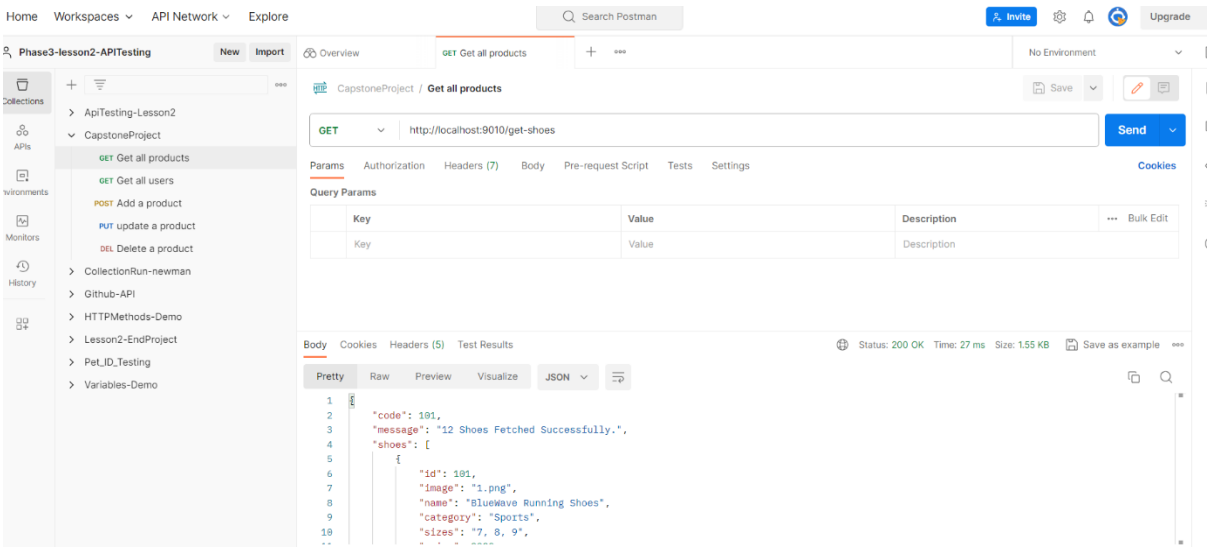
```
package steps;

import io.cucumber.java.Given;
import io.cucumber.java.When;
import io.cucumber.java.Then;
import io.restassured.RestAssured;
import io.restassured.response.Response;

import static org.junit.Assert.assertEquals;

public class ProductAPIStepDefinitions {
```

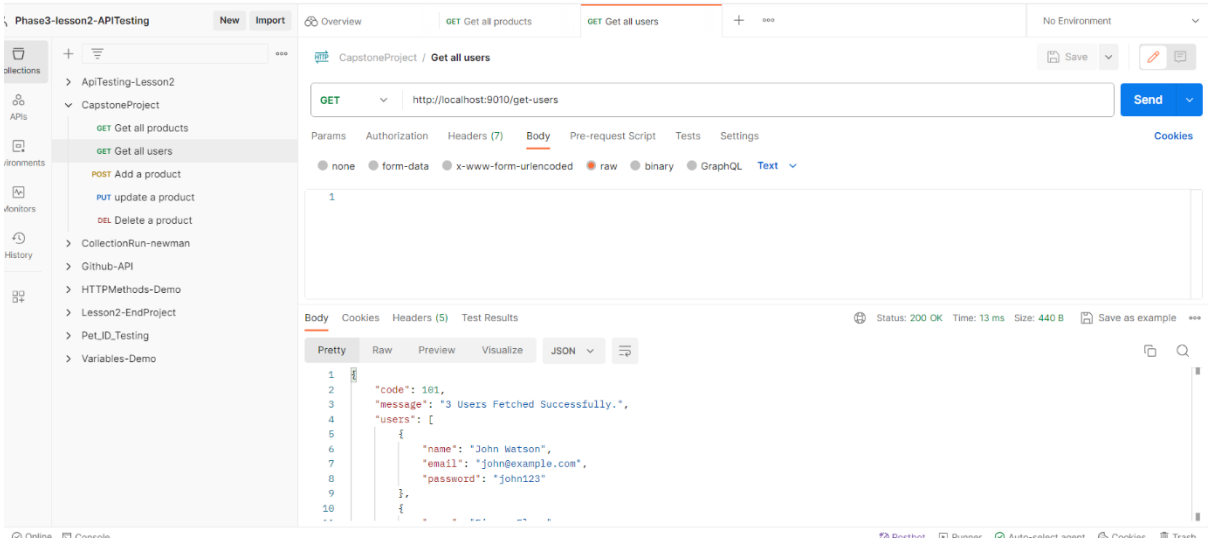
```
    private Response response;

    @Given("I send a GET request to \"/get-shoes\"")
    public void sendGETRequestToGetProducts() {
        response = RestAssured.get("http://localhost:9010/get-
shoes");
    }

    @When("I receive a response from the server")
    public void receiveResponseFromServer() {
        // Check if the response was successful
        assertEquals(200, response.getStatusCode());
    }

    @Then("the response status code should be 200")
    public void responseStatusCodeShouldBe200() {
        assertEquals(200, response.getStatusCode());
    }

    @And("the response should contain a list of products")
    public void responseShouldContainListOfProducts() {
        // Verify that the response body contains a list of products
        // Parse the JSON response and validate the structure
    }

    @Given("I send a GET request to \"/get-users\"")
    public void sendGETRequestToGetUsers() {
        response = RestAssured.get("http://localhost:9010/get-
users");
    }

    @Then("the response should contain a list of registered users")
    public void responseShouldContainListOfRegisteredUsers() {
        // Verify that the response body contains a list of
registered users
        // Parse the JSON response and validate the structure
    }

    @Given("I send a POST request to \"/add-shoe\" with the
following JSON body:")
    public void sendPOSTRequestToAddShoeWithJSONBody(String
jsonBody) {
        response = RestAssured.given()
                .contentType("application/json")
                .body(jsonBody)
                .post("http://localhost:9010/add-shoe");
    }

    @Then("the response status code should be 201")
    public void responseStatusCodeShouldBe201() {
        assertEquals(201, response.getStatusCode());
    }

    @And("the response should contain the newly added product")
    public void responseShouldContainNewlyAddedProduct() {
```

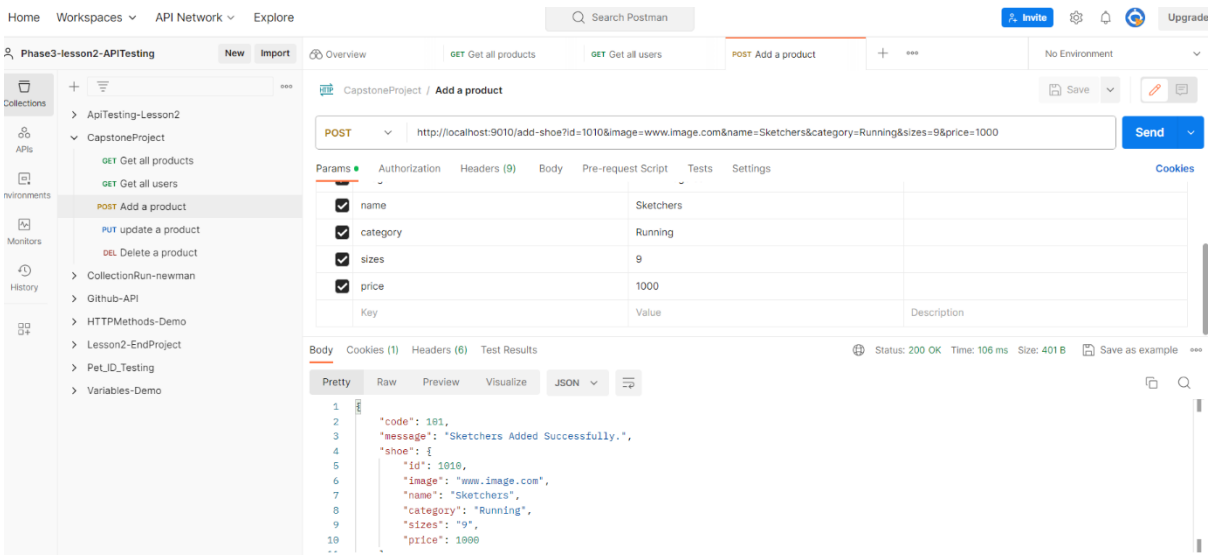
```
        // Verify that the response body contains the newly added
product
        // Parse the JSON response and validate the structure
    }

    @Given("I send a DELETE request to \"/delete-shoe?id=102\"")
    public void sendDELETERequestToDeleteShoe() {
        response = RestAssured.delete("http://localhost:9010/delete-
shoe?id=102");
    }

    @Then("the response status code should be 204")
    public void responseStatusCodeShouldBe204() {
        assertEquals(204, response.getStatusCode());
    }

    @Given("I send a PUT request to \"/update-shoe\" with the
following JSON body:")
    public void sendPUTRequestToUpdateShoeWithJSONBody(String
jsonBody) {
        response = RestAssured.given()
                .contentType("application/json")
                .body(jsonBody)
                .put("http://localhost:9010/update-shoe");
    }
}
```

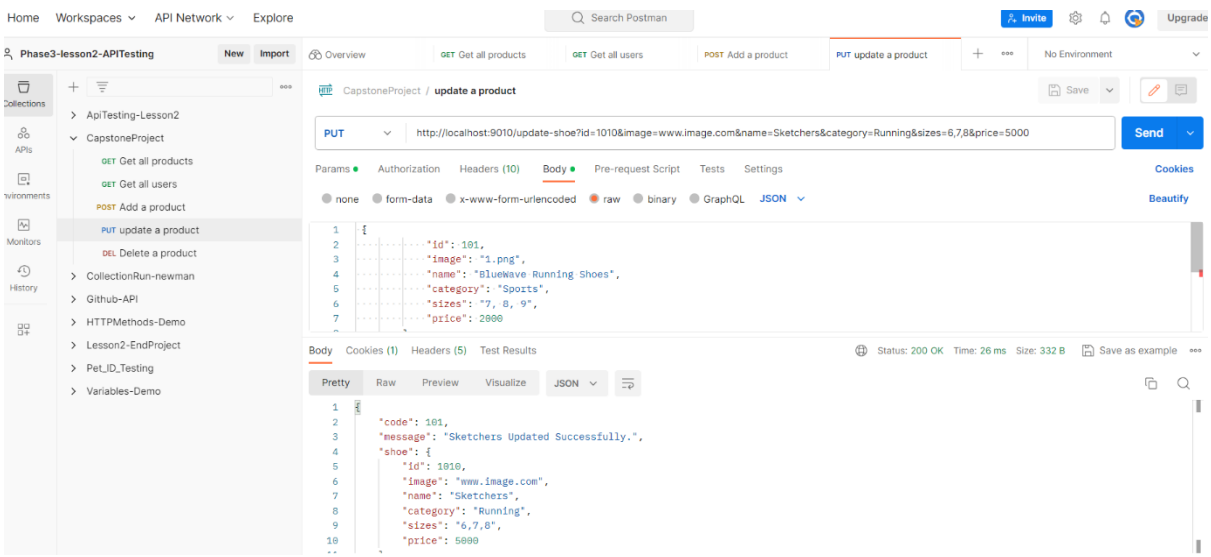
4.API Testing with Postman and Rest Assured.

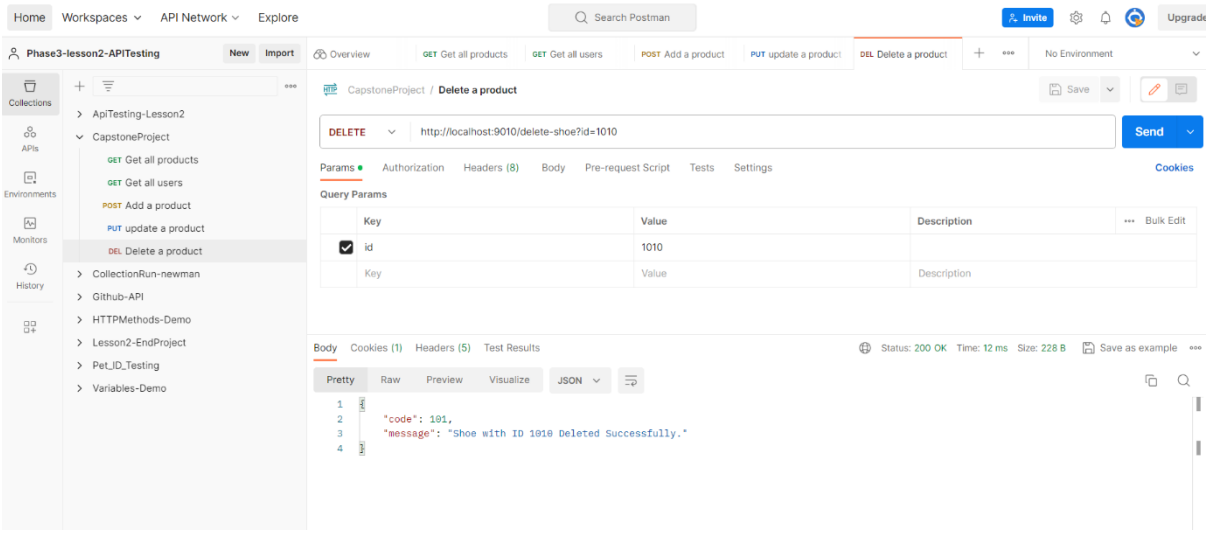
POSTMAN:




Phase3-lesson2-APITesting
New
Import
Overview
GET Get all products
GET Get all users
No Environment
CapstoneProject / Get all users
Save
GET
http://localhost:9010/get-users
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
Text
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 13 ms
Size: 440 B
Save as example
Pretty
Raw
Preview
Visualize
JSON
ApiTesting-Lesson2
CapstoneProject
GET Get all products
GET Get all users
POST Add a product
PUT update a product
DEL Delete a product
CollectionRun-newman
Github-API
HTTPMethods-Demo
Lesson2-EndProject
Pet_ID_Testing
Variables-Demo
{
"code": 101,
"message": "3 Users Fetched Successfully.",
"users": [
{
"name": "John Watson",
"email": "john@example.com",
"password": "john123"
},
{


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
Phase3-lesson2-APITesting
New
Import
Overview
GET Get all products
GET Get all users
POST Add a product
No Environment
CapstoneProject / Add a product
Save
POST
http://localhost:9010/add-shoe?id=1010&image=www.image.com&name=Sketchers&category=Running&sizes=9&price=1000
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
name
Sketchers
category
Running
sizes
9
price
1000
Key
Value
Description
Body
Cookies (1)
Headers (6)
Test Results
Status: 200 OK
Time: 106 ms
Size: 401 B
Save as example
Pretty
Raw
Preview
Visualize
JSON
ApiTesting-Lesson2
CapstoneProject
GET Get all products
GET Get all users
POST Add a product
PUT update a product
DEL Delete a product
CollectionRun-newman
Github-API
HTTPMethods-Demo
Lesson2-EndProject
Pet_ID_Testing
Variables-Demo
{
"code": 101,
"message": "Sketchers Added Successfully.",
"shoe": {
"id": 1010,
"image": "www.image.com",
"name": "Sketchers",
"category": "Running",
"sizes": "9",
"price": 1000


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
Phase3-lesson2-APITesting
New
Import
Overview
GET Get all products
GET Get all users
POST Add a product
PUT update a product
No Environment
CapstoneProject / update a product
Save
PUT
http://localhost:9010/update-shoe?id=1010&image=www.image.com&name=Sketchers&category=Running&sizes=6,7,8&price=5000
Send
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id": 101,
"image": "1.png",
"name": "BlueWave Running Shoes",
"category": "Sports",
"sizes": "7, 8, 9",
"price": 2000
Body
Cookies (1)
Headers (5)
Test Results
Status: 200 OK
Time: 26 ms
Size: 332 B
Save as example
Pretty
Raw
Preview
Visualize
JSON
ApiTesting-Lesson2
CapstoneProject
GET Get all products
GET Get all users
POST Add a product
PUT update a product
DEL Delete a product
CollectionRun-newman
Github-API
HTTPMethods-Demo
Lesson2-EndProject
Pet_ID_Testing
Variables-Demo
{
"code": 101,
"message": "Sketchers Updated Successfully.",
"shoe": {
"id": 1010,
"image": "www.image.com",
"name": "Sketchers",
"category": "Running",
"sizes": "6,7,8",
"price": 5000



REST ASSURED:

Getallshoes:

```
package com.sportyshoe.restAssuredScripts;

import org.testng.annotations.Test;

import io.restassured.RestAssured;

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='2')
public void get_all_users()
{

      RestAssured.given()
```

```
      .baseUri("http://localhost:9010")
      .basePath("/get-users")
      .when()
      .get()
      .then()
      .statusCode(200)
      .log()
      .all();

}

}
```

PostandPutnewshoes:

```
package com.sportyshoe.restAssuredScripts;

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();

}
}
```