**Source Code**

**CREATED SELENIUM SCRIPT USING TESTNG**

**MAIN CODE:**

package sportyshoes.testscripts;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.Assert;

import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

import sportyshoes.pages.LoginPage;

import sportyshoes.pages.AddToCart;

import sportyshoes.pages.Verification;

public class ShoeSelectionTest {

WebDriver driver;

@BeforeTest

public void launchApplication()

{

driver = new ChromeDriver();

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

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

}

@Test

public void shoeSelection() throws InterruptedException

{

LoginPage loginPage = new LoginPage(driver);

loginPage.enterEmail("knight3@gmail.com");

loginPage.enterPassword("abc123");

loginPage.clickLoginButton();

AddToCart addToCart = new AddToCart(driver);

addToCart.buyShoe();

Verification verification = new Verification(driver);

String expectedMsg = "Success!\r\n"

+ "Message:Shoe BlueWave Running Shoes Added Successfully to Cart";

String actualMsg = verification.verify();

Assert.assertEquals(actualMsg, expectedMsg);

}

@AfterTest

public void closeBrowser()

{

driver.quit();

}

}

**PAGE OBJECT MODEL CODE:**

package sportyshoes.pages;

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(id = "email")

private WebElement emailId;

@FindBy(id = "password")

private WebElement pass;

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

private WebElement login;

public LoginPage(WebDriver driver)

{

PageFactory.initElements(driver, this);

}

public void enterEmail(String email)

{

emailId.sendKeys(email);

}

public void enterPassword(String password)

{

pass.sendKeys(password);

}

public void clickLoginButton()

{

login.click();

}

}

package sportyshoes.pages;

import java.time.Duration;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

import org.openqa.selenium.support.ui.WebDriverWait;

public class AddToCart {

@FindBy(xpath = "(//a[contains(@class,'btn btn-primary)])[1]")

private WebElement cart;

JavascriptExecutor js;

WebDriverWait wait;

Actions actions;

public AddToCart(WebDriver driver)

{

js = (JavascriptExecutor) driver;

PageFactory.initElements(driver, this);

actions = new Actions(driver);

wait = new WebDriverWait(driver, Duration.ofSeconds(60));

}

public void buyShoe() throws InterruptedException

{

Thread.sleep(2000);

js.executeScript("arguments[0].scrollIntoView();", cart);

js.executeScript("arguments[0].click()", cart);

}

}

package sportyshoes.pages;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class Verification {

@FindBy(xpath = "//div[contains(@class,'alert alert-success')]")

private WebElement success;

public Verification(WebDriver driver)

{

PageFactory.initElements(driver, this);

}

public String verify()

{

String text = success.getText();

return text;

}

}

**USING POSTMAN**

GET Request generated using the link: <http://localhost:9010/get-shoes>

GET Request generated using the link: <http://localhost:9010/get-users>

POST Request generated using the link: <http://localhost:9010/add-shoe>

**API TESTING USING RESTASSURED**

**package** RestAssured.RestAssured\_Demo;

**import** java.util.HashMap;

**import** org.testng.annotations.BeforeMethod;

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

**import** io.restassured.RestAssured;

**public** **class** SportyShoes {

@Test(priority = 0)

**public** **void** getProducts()

{

RestAssured

.*when*()

.get("http://localhost:9010/get-shoes")

.then()

.assertThat()

.statusCode(200)

.log().all();

}

@Test(priority = 1)

**public** **void** getRegisteredUsers()

{

RestAssured

.*when*()

.get("http://localhost:9010/get-users")

.then()

.assertThat()

.statusCode(200)

.log().all();

}

HashMap<String, String> map = **new** HashMap<String, String>();

@BeforeMethod

**public** **void** addProduct()

{

map.put("Name", "SampleShoe");

map.put("Category", "Running");

map.put("Size", "9");

map.put("Price", "1000");

RestAssured.*baseURI* = "http://localhost:9010/";

RestAssured.*basePath* = "/add-shoe";

}

@Test(priority = 2)

**public** **void** showProductDetails()

{

RestAssured

.*given*()

.contentType("application/json")

.body(map)

.when()

.post()

.then()

.statusCode(400)

.log().all();

}

}

**JMX FILE GENERATED WHILE RECORDING USING BLAZEMETER**

<?xml version="1.0" encoding="UTF-8"?>

<jmeterTestPlan version="1.2" properties="5.0" jmeter="5.6.2">

<hashTree>

<TestPlan guiclass="TestPlanGui" testclass="TestPlan" testname="Capstone project" enabled="true">

<stringProp name="TestPlan.comments">This test plan was created by the BlazeMeter converter v.3.1.23. Please contact support@blazemeter.com for further support.</stringProp>

<boolProp name="TestPlan.functional\_mode">false</boolProp>

<boolProp name="TestPlan.serialize\_threadgroups">false</boolProp>

<elementProp name="TestPlan.user\_defined\_variables" elementType="Arguments">

<collectionProp name="Arguments.arguments"/>

</elementProp>

<stringProp name="TestPlan.user\_define\_classpath"></stringProp>

</TestPlan>

<hashTree>

<HeaderManager guiclass="HeaderPanel" testclass="HeaderManager" testname="HTTP Header manager" enabled="true">

<collectionProp name="HeaderManager.headers">

<elementProp name="sec-ch-ua" elementType="Header">

<stringProp name="Header.name">sec-ch-ua</stringProp>

<stringProp name="Header.value">&quot;Chromium&quot;;v=&quot;116&quot;, &quot;Not)A;Brand&quot;;v=&quot;24&quot;, &quot;Google Chrome&quot;;v=&quot;116&quot;</stringProp>

</elementProp>

<elementProp name="sec-ch-ua-mobile" elementType="Header">

<stringProp name="Header.name">sec-ch-ua-mobile</stringProp>

<stringProp name="Header.value">?0</stringProp>

</elementProp>

<elementProp name="Accept" elementType="Header">

<stringProp name="Header.name">Accept</stringProp>

<stringProp name="Header.value">text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.7</stringProp>

</elementProp>

<elementProp name="Upgrade-Insecure-Requests" elementType="Header">

<stringProp name="Header.name">Upgrade-Insecure-Requests</stringProp>

<stringProp name="Header.value">1</stringProp>

</elementProp>

<elementProp name="sec-ch-ua-platform" elementType="Header">

<stringProp name="Header.name">sec-ch-ua-platform</stringProp>

<stringProp name="Header.value">&quot;Windows&quot;</stringProp>

</elementProp>

<elementProp name="User-Agent" elementType="Header">

<stringProp name="Header.name">User-Agent</stringProp>

<stringProp name="Header.value">Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36</stringProp>

</elementProp>

</collectionProp>

</HeaderManager>

<hashTree/>

<Arguments guiclass="ArgumentsPanel" testclass="Arguments" testname="User Defined Variables" enabled="true">

<collectionProp name="Arguments.arguments">

<elementProp name="BASE\_URL\_1" elementType="Argument">

<stringProp name="Argument.name">BASE\_URL\_1</stringProp>

<stringProp name="Argument.value">localhost</stringProp>

<stringProp name="Argument.metadata">=</stringProp>

</elementProp>

</collectionProp>

</Arguments>

<hashTree/>

<ConfigTestElement guiclass="HttpDefaultsGui" testclass="ConfigTestElement" testname="HTTP Request Defaults" enabled="true">

<elementProp name="HTTPsampler.Arguments" elementType="Arguments" guiclass="HTTPArgumentsPanel" testclass="Arguments" enabled="true">

<collectionProp name="Arguments.arguments"/>

</elementProp>

</ConfigTestElement>

<hashTree/>

<DNSCacheManager guiclass="DNSCachePanel" testclass="DNSCacheManager" testname="DNS Cache Manager" enabled="true">

<collectionProp name="DNSCacheManager.servers"/>

<boolProp name="DNSCacheManager.clearEachIteration">true</boolProp>

<boolProp name="DNSCacheManager.isCustomResolver">false</boolProp>

</DNSCacheManager>

<hashTree/>

<AuthManager guiclass="AuthPanel" testclass="AuthManager" testname="HTTP Authorization Manager" enabled="true">

<collectionProp name="AuthManager.auth\_list"/>

<boolProp name="AuthManager.controlledByThreadGroup">false</boolProp>

</AuthManager>

<hashTree/>

<CookieManager guiclass="CookiePanel" testclass="CookieManager" testname="HTTP Cookie Manager" enabled="true">

<collectionProp name="CookieManager.cookies"/>

<boolProp name="CookieManager.clearEachIteration">true</boolProp>

<boolProp name="CookieManager.controlledByThreadGroup">false</boolProp>

</CookieManager>

<hashTree/>

<CacheManager guiclass="CacheManagerGui" testclass="CacheManager" testname="HTTP Cache Manager" enabled="true">

<boolProp name="clearEachIteration">true</boolProp>

<boolProp name="useExpires">false</boolProp>

<boolProp name="CacheManager.controlledByThread">false</boolProp>

</CacheManager>

<hashTree/>

<ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup" testname="Thread Group" enabled="true">

<stringProp name="ThreadGroup.on\_sample\_error">continue</stringProp>

<elementProp name="ThreadGroup.main\_controller" elementType="LoopController" guiclass="LoopControlPanel" testclass="LoopController" enabled="true">

<stringProp name="LoopController.loops">1</stringProp>

<boolProp name="LoopController.continue\_forever">false</boolProp>

</elementProp>

<stringProp name="ThreadGroup.num\_threads">1</stringProp>

<stringProp name="ThreadGroup.ramp\_time">1</stringProp>

<boolProp name="ThreadGroup.scheduler">false</boolProp>

<stringProp name="ThreadGroup.duration">0</stringProp>

<stringProp name="ThreadGroup.delay">0</stringProp>

<boolProp name="ThreadGroup.delayedStart">false</boolProp>

<boolProp name="ThreadGroup.same\_user\_on\_next\_iteration">true</boolProp>

</ThreadGroup>

<hashTree>

<TransactionController guiclass="TransactionControllerGui" testclass="TransactionController" testname="Test" enabled="true">

<boolProp name="TransactionController.includeTimers">false</boolProp>

<boolProp name="TransactionController.parent">false</boolProp>

</TransactionController>

<hashTree>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="http://localhost:9010/" enabled="true">

<boolProp name="HTTPSampler.postBodyRaw">false</boolProp>

<elementProp name="HTTPsampler.Arguments" elementType="Arguments" guiclass="HTTPArgumentsPanel" testclass="Arguments" enabled="true">

<collectionProp name="Arguments.arguments"/>

</elementProp>

<stringProp name="HTTPSampler.domain">${BASE\_URL\_1}</stringProp>

<stringProp name="HTTPSampler.port">9010</stringProp>

<stringProp name="HTTPSampler.protocol">http</stringProp>

<stringProp name="HTTPSampler.method">GET</stringProp>

<boolProp name="HTTPSampler.follow\_redirects">true</boolProp>

<boolProp name="HTTPSampler.auto\_redirects">false</boolProp>

<boolProp name="HTTPSampler.use\_keepalive">true</boolProp>

<boolProp name="HTTPSampler.DO\_MULTIPART\_POST">false</boolProp>

<boolProp name="HTTPSampler.BROWSER\_COMPATIBLE\_MULTIPART">false</boolProp>

<boolProp name="HTTPSampler.image\_parser">false</boolProp>

<boolProp name="HTTPSampler.concurrentDwn">false</boolProp>

<stringProp name="HTTPSampler.concurrentPool">6</stringProp>

<boolProp name="HTTPSampler.md5">false</boolProp>

<intProp name="HTTPSampler.ipSourceType">0</intProp>

</HTTPSamplerProxy>

<hashTree>

<ConstantTimer guiclass="ConstantTimerGui" testclass="ConstantTimer" testname="Constant Timer" enabled="true">

<stringProp name="ConstantTimer.delay">0</stringProp>

</ConstantTimer>

<hashTree/>

</hashTree>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="http://localhost:9010/login" enabled="true">

<boolProp name="HTTPSampler.postBodyRaw">false</boolProp>

<elementProp name="HTTPsampler.Arguments" elementType="Arguments" guiclass="HTTPArgumentsPanel" testclass="Arguments" enabled="true">

<collectionProp name="Arguments.arguments">

<elementProp name="password" elementType="HTTPArgument">

<boolProp name="HTTPArgument.always\_encode">true</boolProp>

<stringProp name="Argument.name">password</stringProp>

<stringProp name="Argument.value">KZK2kzk\*</stringProp>

<stringProp name="Argument.metadata">=</stringProp>

</elementProp>

<elementProp name="email" elementType="HTTPArgument">

<boolProp name="HTTPArgument.always\_encode">true</boolProp>

<stringProp name="Argument.name">email</stringProp>

<stringProp name="Argument.value">knight3@gmail.com</stringProp>

<stringProp name="Argument.metadata">=</stringProp>

</elementProp>

</collectionProp>

</elementProp>

<stringProp name="HTTPSampler.domain">${BASE\_URL\_1}</stringProp>

<stringProp name="HTTPSampler.port">9010</stringProp>

<stringProp name="HTTPSampler.protocol">http</stringProp>

<stringProp name="HTTPSampler.path">login</stringProp>

<stringProp name="HTTPSampler.method">POST</stringProp>

<boolProp name="HTTPSampler.follow\_redirects">true</boolProp>

<boolProp name="HTTPSampler.auto\_redirects">false</boolProp>

<boolProp name="HTTPSampler.use\_keepalive">true</boolProp>

<boolProp name="HTTPSampler.DO\_MULTIPART\_POST">false</boolProp>

<boolProp name="HTTPSampler.BROWSER\_COMPATIBLE\_MULTIPART">false</boolProp>

<boolProp name="HTTPSampler.image\_parser">false</boolProp>

<boolProp name="HTTPSampler.concurrentDwn">false</boolProp>

<stringProp name="HTTPSampler.concurrentPool">6</stringProp>

<boolProp name="HTTPSampler.md5">false</boolProp>

<intProp name="HTTPSampler.ipSourceType">0</intProp>

</HTTPSamplerProxy>

<hashTree>

<HeaderManager guiclass="HeaderPanel" testclass="HeaderManager" testname="HTTP Header manager" enabled="true">

<collectionProp name="HeaderManager.headers">

<elementProp name="Content-Type" elementType="Header">

<stringProp name="Header.name">Content-Type</stringProp>

<stringProp name="Header.value">application/x-www-form-urlencoded</stringProp>

</elementProp>

</collectionProp>

</HeaderManager>

<hashTree/>

<ConstantTimer guiclass="ConstantTimerGui" testclass="ConstantTimer" testname="Constant Timer" enabled="true">

<stringProp name="ConstantTimer.delay">15994</stringProp>

</ConstantTimer>

<hashTree/>

</hashTree>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="http://localhost:9010/add-to-cart?id=101" enabled="true">

<boolProp name="HTTPSampler.postBodyRaw">false</boolProp>

<elementProp name="HTTPsampler.Arguments" elementType="Arguments" guiclass="HTTPArgumentsPanel" testclass="Arguments" enabled="true">

<collectionProp name="Arguments.arguments">

<elementProp name="id" elementType="HTTPArgument">

<boolProp name="HTTPArgument.always\_encode">false</boolProp>

<stringProp name="Argument.name">id</stringProp>

<stringProp name="Argument.value">101</stringProp>

<stringProp name="Argument.metadata">=</stringProp>

</elementProp>

</collectionProp>

</elementProp>

<stringProp name="HTTPSampler.domain">${BASE\_URL\_1}</stringProp>

<stringProp name="HTTPSampler.port">9010</stringProp>

<stringProp name="HTTPSampler.protocol">http</stringProp>

<stringProp name="HTTPSampler.path">add-to-cart</stringProp>

<stringProp name="HTTPSampler.method">GET</stringProp>

<boolProp name="HTTPSampler.follow\_redirects">true</boolProp>

<boolProp name="HTTPSampler.auto\_redirects">false</boolProp>

<boolProp name="HTTPSampler.use\_keepalive">true</boolProp>

<boolProp name="HTTPSampler.DO\_MULTIPART\_POST">false</boolProp>

<boolProp name="HTTPSampler.BROWSER\_COMPATIBLE\_MULTIPART">false</boolProp>

<boolProp name="HTTPSampler.image\_parser">false</boolProp>

<boolProp name="HTTPSampler.concurrentDwn">false</boolProp>

<stringProp name="HTTPSampler.concurrentPool">6</stringProp>

<boolProp name="HTTPSampler.md5">false</boolProp>

<intProp name="HTTPSampler.ipSourceType">0</intProp>

</HTTPSamplerProxy>

<hashTree>

<ConstantTimer guiclass="ConstantTimerGui" testclass="ConstantTimer" testname="Constant Timer" enabled="true">

<stringProp name="ConstantTimer.delay">8490</stringProp>

</ConstantTimer>

<hashTree/>

</hashTree>

</hashTree>

</hashTree>

</hashTree>

</hashTree>

</jmeterTestPlan>