Capstone Project - Source Code

src/test/java

BaseTest.java

```
package finalProject.TestScripts;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.edge.EdgeDriver;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeTest;

public class BaseTest {
	WebDriver driver;

	@BeforeTest
	public void LaunchApplication() {
		// 1. Open the browser
		driver = new EdgeDriver();

		// 2. Maximize it
		driver.manage().window().maximize();

		// 3. Navigate to http://localhost:9010
		driver.get("http://localhost:9010");
	}

	// Close the browser
	@AfterTest
	public void CloseBrowser() {
		driver.quit();

	}
}
```

PlaceOrderPageTest.java

```
package finalProject.TestScripts;

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

import finalProject.Pages.AddCartTextVerifyPage;
import finalProject.Pages.CartPage;
import finalProject.Pages.HomePage;
import finalProject.Pages.LandingPage;
import finalProject.Pages.LogoutPage;
import finalProject.Pages.PlaceOrderTextVerifyPage;
import finalProject.Pages.UserRegisterPage;
```

```
public class PlaceOrderPageTest extends BaseTest {

	@Test
	public void BuyShoes() throws InterruptedException {
		// Click on 'New User'
		LandingPage landingPage = new LandingPage(driver);
		landingPage.clickNewUser();

		// Register 'New User'
		UserRegisterPage userRegisterPage = new
UserRegisterPage(driver);
		userRegisterPage.enterName();
		userRegisterPage.enterEmail();
		userRegisterPage.enterPassword();
		userRegisterPage.clickRegisterBtn();

		// Logout
		LogoutPage logoutPage = new LogoutPage(driver);
		logoutPage.clickLogoutBtn();

		// Login
		landingPage.enterLoginEmail();
		landingPage.enterLoginPassword();
		landingPage.clickLoginBtn();

		// Add Product in cart
		HomePage homePage = new HomePage(driver);
		homePage.clickAddToCartBtn();

		// Verify Add cart Text
		AddCartTextVerifyPage addCartTextVerifyPage = new
AddCartTextVerifyPage(driver);
		String expectedText = "Message:Shoe BlueWave Running Shoes
Added Successfully to Cart";
		String actualText = addCartTextVerifyPage.getMessageText();
		Assert.assertEquals(actualText, expectedText);
		addCartTextVerifyPage.clickHome();

		homePage.clickCart();

		CartPage cartPage = new CartPage(driver);
		cartPage.clickPlaceOrderBtn();

		// Verify Place order Text
		PlaceOrderTextVerifyPage placeOrderTextVerifyPage = new
PlaceOrderTextVerifyPage(driver);
		String finalExpectedText = "Message:Order Placed Successfully
with ID: 3";
		String finalActualText =
placeOrderTextVerifyPage.getMessageText();
		Assert.assertEquals(finalActualText, finalExpectedText);
		addCartTextVerifyPage.clickHome();

	}

}
```

All_Request.java

```
package restAssured;

import java.util.HashMap;
import org.testng.annotations.Test;
import io.restassured.RestAssured;


public class All_Request extends TestBase {
	HashMap<String, String> map = new HashMap<String, String>();

	@Test(priority = 1)

	public void getProducts() {
		RestAssured.when().get("http://localhost:9010/get-
shoes").then().assertThat().statusCode(200);
		logger.info("Reterived Products");
	}

	@Test(priority = 2)

	public void getUsers() {
		RestAssured.when().get("http://localhost:9010/get-
users").then().assertThat().statusCode(200);
		logger.info("Reterived Users");
	}

	@Test(priority = 3)
	public void addProduct() {

	RestAssured.given().contentType("application/json").body(map).when()
.post(
				"http://localhost:9010/add-
shoe?id=101&image=image_url&name=SampleShoe&category=Running&sizes=9&price=
1000")
				.then().statusCode(200).log().all();
		logger.info("Added Product");
	}

}
```

TestBase.java

```
package restAssured;

import org.apache.log4j.Level;
import org.apache.log4j.Logger;
import org.apache.log4j.PropertyConfigurator;
import org.testng.annotations.BeforeClass;

import io.restassured.response.Response;
import io.restassured.specification.RequestSpecification;

public class TestBase {
	public static RequestSpecification httpRequest;
```

```
	public static Response response;

	public Logger logger;

	@BeforeClass
	public void setUp() {
		logger = Logger.getLogger("LogDemo");
		PropertyConfigurator.configure("log4j.properties");
		logger.setLevel(Level.DEBUG);
	}

}
```

src/main/java

LandingPage.java

```
package finalProject.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 LandingPage {
	@FindBy(linkText = "New User? Register Here")
	private WebElement newUser;

	@FindBy(id = "email")
	private WebElement emailTextBox;

	@FindBy(id = "password")
	private WebElement passwordTextBox;

	@FindBy(xpath = "//button[contains(@class, btn)]")
	private WebElement loginBtn;

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

	public void clickNewUser() {
		newUser.click();
	}

	public void enterLoginEmail() {
		emailTextBox.sendKeys("yokesh@gmail.com");
	}

	public void enterLoginPassword() {
		passwordTextBox.sendKeys("pass@123");
	}

	public void clickLoginBtn() {
		loginBtn.click();
	}
```

```
}
```

LogoutPage.java

```
package finalProject.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 LogoutPage {
	@FindBy(linkText = "Logout")
	private WebElement logoutBtn;

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

	public void clickLogoutBtn() {
		logoutBtn.click();
	}
}
```

UserRegisterPage.java

```
package finalProject.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 UserRegisterPage {
	@FindBy(id = "name")
	private WebElement nameTextBox;

	@FindBy(id = "email")
	private WebElement emailTextBox;

	@FindBy(id = "password")
	private WebElement passwordTextBox;

	@FindBy(xpath = "//button[contains(@class, 'btn')]")
	private WebElement registerBtn;

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

	public void enterName() {
```

```
		nameTextBox.sendKeys("Yokesh");
	}

	public void enterEmail() {
		emailTextBox.sendKeys("yokesh@gmail.com");
	}

	public void enterPassword() {
		passwordTextBox.sendKeys("pass@123");
	}

	public void clickRegisterBtn() {
		registerBtn.click();
	}
}
```

PlaceOrderTextVerifyPage.java

```
package finalProject.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 PlaceOrderTextVerifyPage {
	@FindBy(xpath = "/html/body/div[3]/div/p")
	private WebElement getPlaceOrderSuccessMessageText;

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

	public String getMessageText() {
		String text = getPlaceOrderSuccessMessageText.getText();
		return text;
	}
}
```

HomePage.java

```
package finalProject.Pages;

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 HomePage {
	private JavascriptExecutor js;

	@FindBy(xpath = "(//a[contains(@class, 'btn')])[5]")
	private WebElement addToCartBtn;

	@FindBy(linkText = "Cart")
	private WebElement navigateToCartPage;
```

```
	public HomePage(WebDriver driver) {
		PageFactory.initElements(driver, this);
		js = (JavascriptExecutor) driver;
	}

	public void clickAddToCartBtn() throws InterruptedException {
		Thread.sleep(2000);
		js.executeScript("arguments[0].scrollIntoView();",
addToCartBtn);
		js.executeScript("arguments[0].click();", addToCartBtn);

	}

	public void clickCart() {
		navigateToCartPage.click();
	}
}
```

CartPage.java

```
package finalProject.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 CartPage {
	@FindBy(linkText = "Place Order")
	private WebElement placeOrderBtn;

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

	public void clickPlaceOrderBtn() {
		placeOrderBtn.click();
	}
}
```

AddCartVerifyTextPage.java

```
package finalProject.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 AddCartTextVerifyPage {

	@FindBy(xpath = "/html/body/div[3]/div/p")
```

```
	private WebElement getAddToCartSuccessMessageText;

	@FindBy(linkText = "Home")
	private WebElement navigateToHomePage;

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

	public String getMessageText() {
		String text = getAddToCartSuccessMessageText.getText();
		return text;
	}

	public void clickHome() {
		navigateToHomePage.click();
	}
}
```

Log4j.properties

```
log4j.rootLogger=ERROR,stdout
log4j.logger.com.endeca=INFO
log4j.logger.LogDemo=DEBUG, dest1
# Logger for crawl metrics
log4j.logger.com.endeca.itl.web.metrics=INFO

log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%p\t%d{ISO8601}\t%r\t%c\t[%t
]\t%m%n
```

JMeter

FinalProject.jmx


```
<?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="FinalProject"
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"
guiclass="ArgumentsPanel" testclass="Arguments" enabled="true">

        <collectionProp name="Arguments.arguments"/>

      </elementProp>

      <boolProp name="TestPlan.tearDown_on_shutdown">false</boolProp>

    </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">&amp;quot;Chromium&amp;quot;;v=&amp;quot;116&amp;quot;,
&amp;quot;Not)A;Brand&amp;quot;;v=&amp;quot;24&amp;quot;, &amp;quot;Microsoft
Edge&amp;quot;;v=&amp;quot;116&amp;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/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">"Windows"</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 Edg/116.0.1938.54</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">pass@123</stringProp>

<stringProp name="Argument.metadata">=</stringProp>
<boolProp name="HTTPArgument.use_equals">true</boolProp>
</elementProp>
<elementProp name="email" elementType="HTTPArgument">
<boolProp name="HTTPArgument.always_encode">true</boolProp>
<stringProp name="Argument.name">email</stringProp>
<stringProp name="Argument.value">yokesh@gmail.com</stringProp>
<stringProp name="Argument.metadata">=</stringProp>
<boolProp name="HTTPArgument.use_equals">true</boolProp>
</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">7129</stringProp>

</ConstantTimer>

<hashTree/>

</hashTree>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="http://localhost:9010/add-to-cart;jsessionid=C36A7061323CD1F336AE37F8D8EC57C3?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>

<boolProp name="HTTPArgument.use_equals">true</boolProp>

</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;jsessionid=C36A7061323CD1F336AE37F8D8EC57C3</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">7611</stringProp>

</ConstantTimer>

<hashTree/>

</hashTree>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy" testname="http://localhost:9010/home" 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.path">home</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">13895</stringProp>

</ConstantTimer>

<hashTree/>

</hashTree>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy"
testname="http://localhost:9010/cart" 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.path">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">1931</stringProp>

</ConstantTimer>

<hashTree/>

</hashTree>

<HTTPSamplerProxy guiclass="HttpTestSampleGui" testclass="HTTPSamplerProxy"
testname="http://localhost:9010/place-order" 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.path">place-order</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">3765</stringProp>
          </ConstantTimer>
          <hashTree/>
        </hashTree>
      </hashTree>
      <ResultCollector guiclass="ViewResultsFullVisualizer" testclass="ResultCollector"
testname="View Results Tree" enabled="true">
        <boolProp name="ResultCollector.error_logging">false</boolProp>
        <objProp>
          <name>saveConfig</name>
          <value class="SampleSaveConfiguration">
            <time>true</time>
            <latency>true</latency>
            <timestamp>true</timestamp>
            <success>true</success>
            <label>true</label>
            <code>true</code>
```

<message>true</message>

<threadName>true</threadName>

<dataType>true</dataType>

<encoding>false</encoding>

<assertions>true</assertions>

<subresults>true</subresults>

<responseData>false</responseData>

<samplerData>false</samplerData>

<xml>false</xml>

<fieldNames>true</fieldNames>

<responseHeaders>false</responseHeaders>

<requestHeaders>false</requestHeaders>

<responseDataOnError>false</responseDataOnError>

<saveAssertionResultsFailureMessage>true</saveAssertionResultsFailureMessage>

<assertionsResultsToSave>0</assertionsResultsToSave>

<bytes>true</bytes>

<sentBytes>true</sentBytes>

<url>true</url>

<threadCounts>true</threadCounts>

<idleTime>true</idleTime>

<connectTime>true</connectTime>

</value>

</objProp>

<stringProp name="filename"></stringProp>

</ResultCollector>

<hashTree/>

<ResultCollector guiclass="GraphVisualizer" testclass="ResultCollector" testname="Graph Results" enabled="true">

<boolProp name="ResultCollector.error_logging">false</boolProp>

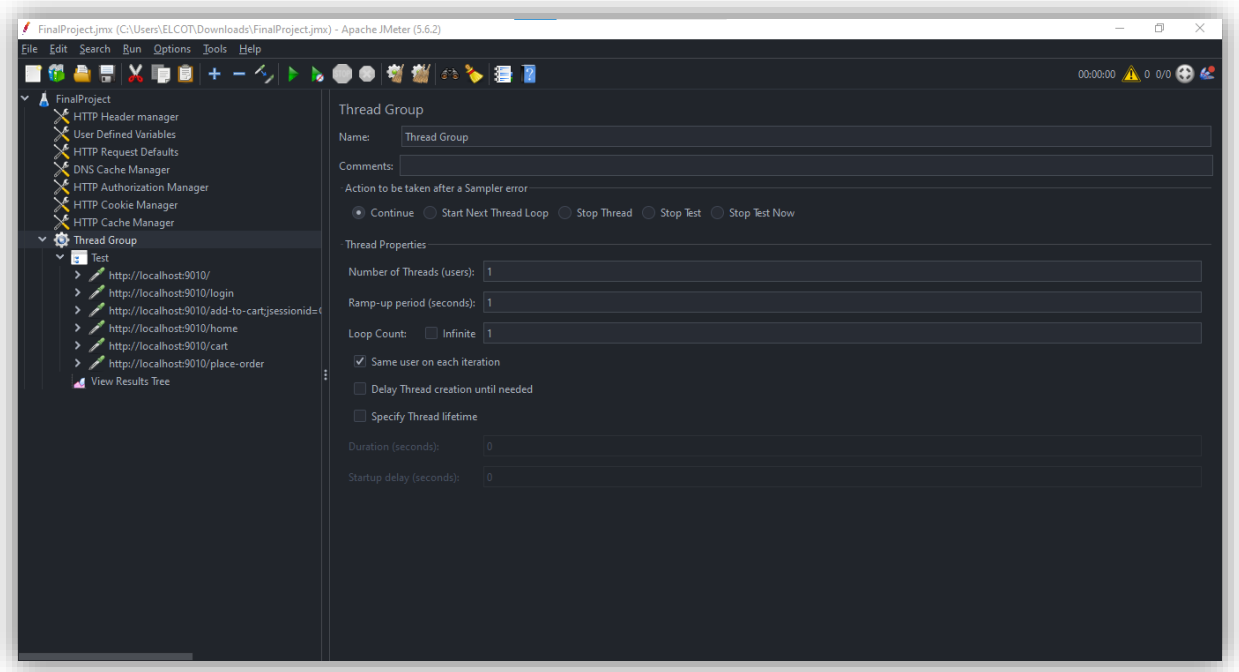
<objProp>

<name>saveConfig</name>

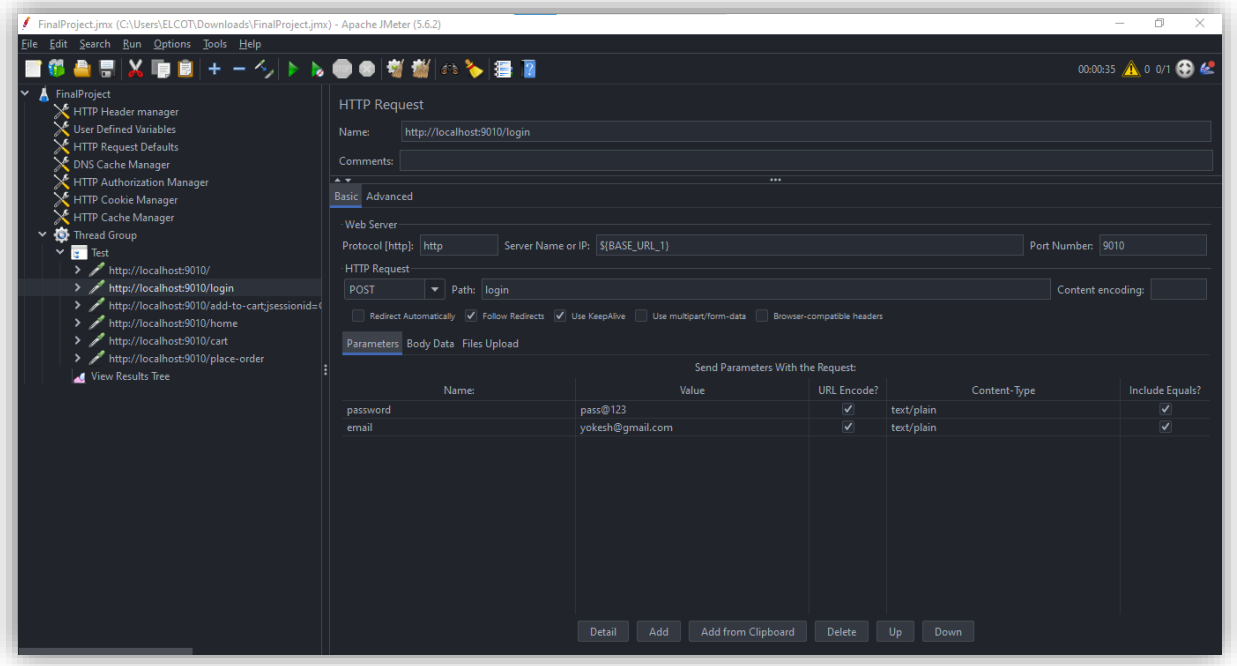
```
 <value class="SampleSaveConfiguration">
  <time>true</time>
  <latency>true</latency>
  <timestamp>true</timestamp>
  <success>true</success>
  <label>true</label>
  <code>true</code>
  <message>true</message>
  <threadName>true</threadName>
  <dataType>true</dataType>
  <encoding>false</encoding>
  <assertions>true</assertions>
  <subresults>true</subresults>
  <responseData>false</responseData>
  <samplerData>false</samplerData>
  <xml>false</xml>
  <fieldNames>true</fieldNames>
  <responseHeaders>false</responseHeaders>
  <requestHeaders>false</requestHeaders>
  <responseDataOnError>false</responseDataOnError>
  <saveAssertionResultsFailureMessage>true</saveAssertionResultsFailureMessage>
  <assertionsResultsToSave>0</assertionsResultsToSave>
  <bytes>true</bytes>
  <sentBytes>true</sentBytes>
  <url>true</url>
  <threadCounts>true</threadCounts>
  <idleTime>true</idleTime>
  <connectTime>true</connectTime>
 </value>
</objProp>
<stringProp name="filename"></stringProp>
```

```
        </ResultCollector>
        <hashTree/>
      </hashTree>
    </hashTree>
  </hashTree>
</jmeterTestPlan>
```

1. Thread Group

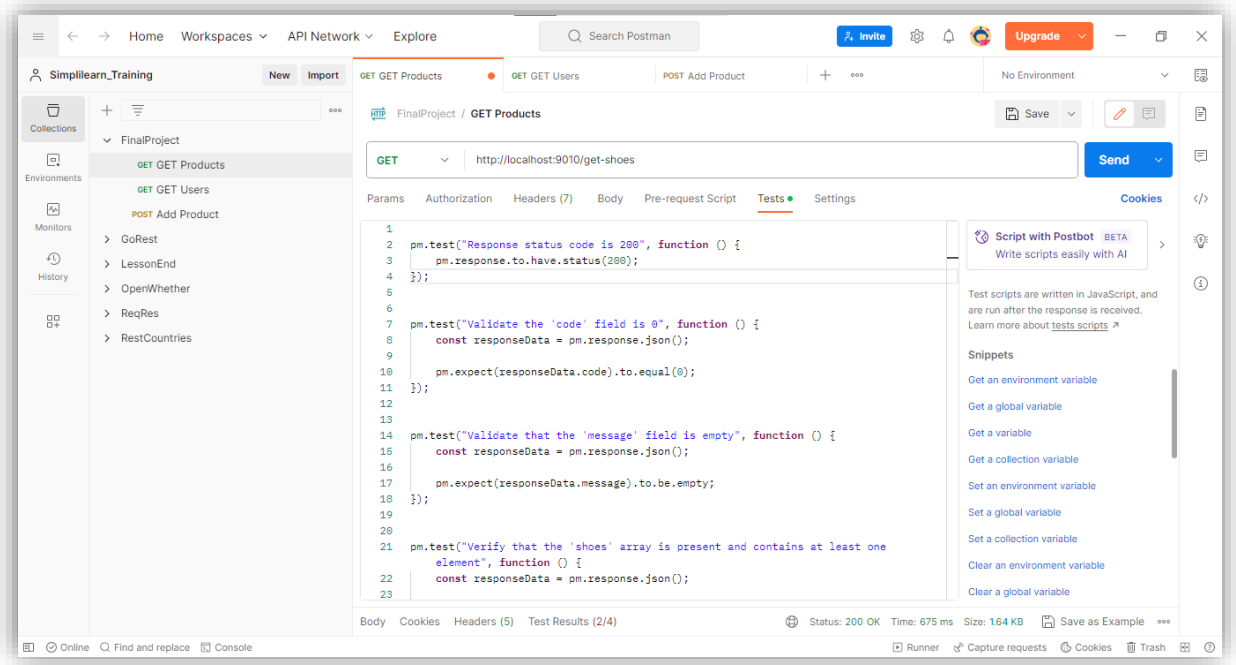


2. Login Credential

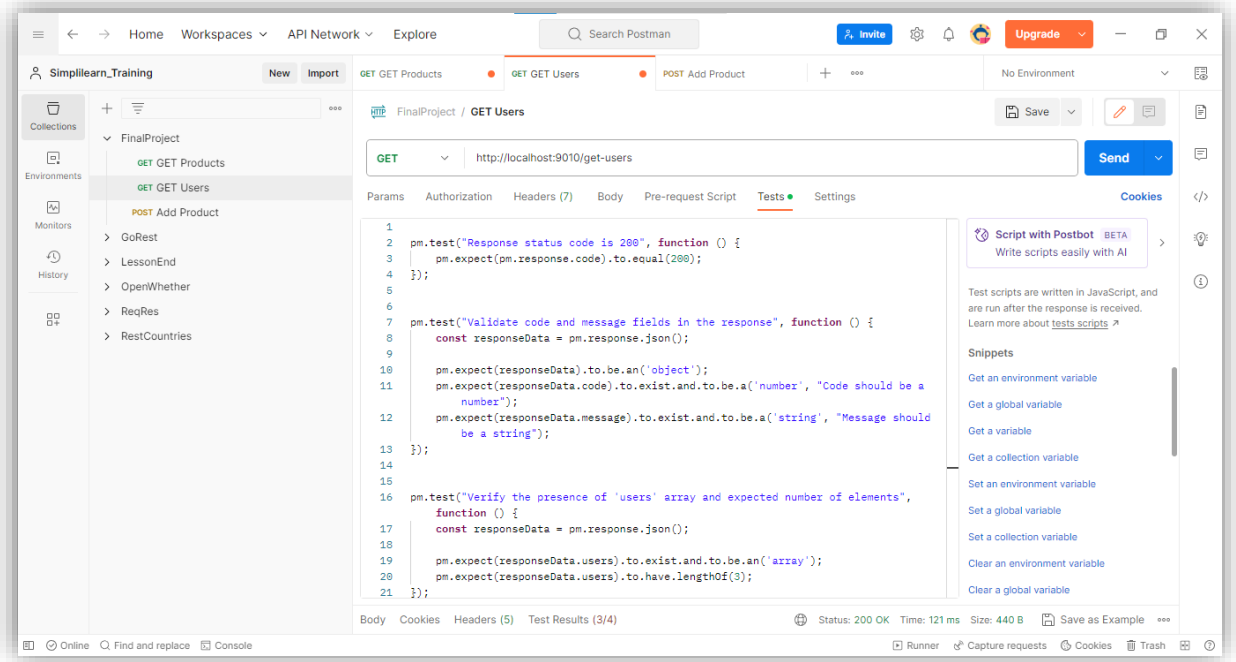


Postman

1. GET Products



2. Get Users



3. Add Product