Test Framework Documentation for Guru99 Demo

By: Minoli

1. Introduction

This document provides an overview of the Selenium-based test automation framework implemented for the Guru99 demo application. It outlines the setup, structure, test execution, and best practices for maintaining and using the framework.

2. Framework Overview

The framework is built using **Java**, **Selenium WebDriver**, and **TestNG**. It supports cross-browser testing and includes modular components for configuration, test execution, reporting, and utility functions.

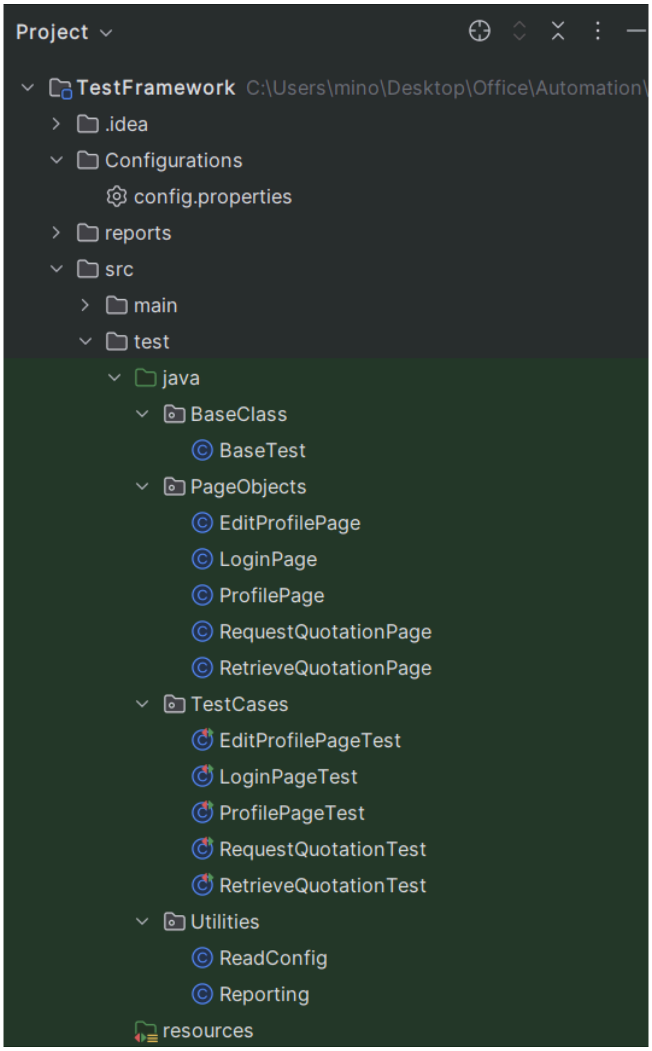
```
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java
-->

<dependency>

   <groupId>org.seleniumhq.selenium</groupId>

   <artifactId>selenium-java</artifactId>

   <version>4.25.0</version>

</dependency>
```

Key Features

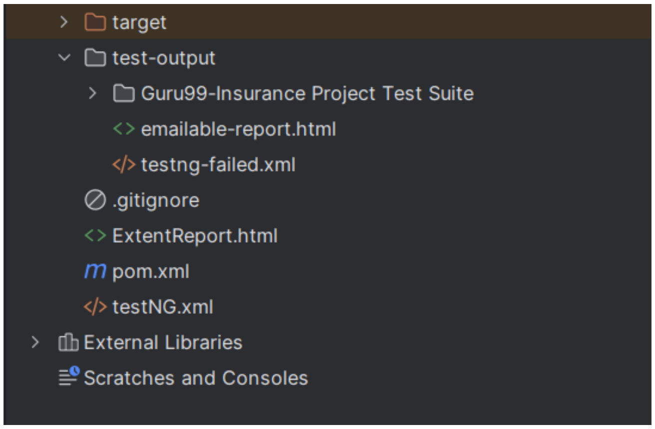
- Cross-browser support: Chrome and Firefox.
- Data-driven testing using configuration files.
- Modular design using Page Object Model (POM).
- Extensible for new test cases and features.

3. Project Structure

The project is organized as follows:



```
Project

TestFramework  C:\Users\mino\Desktop\Office\Automation\
  .idea
  Configurations
    config.properties
  reports
  src
    main
    test
      java
        BaseClass
          BaseTest
        PageObjects
          EditProfilePage
          LoginPage
          ProfilePage
          RequestQuotationPage
          RetrieveQuotationPage
        TestCases
          EditProfilePageTest
          LoginPageTest
          ProfilePageTest
          RequestQuotationTest
          RetrieveQuotationTest
        Utilities
          ReadConfig
          Reporting
      resources
```



```
  target
  test-output
    Guru99-Insurance Project Test Suite
    emailable-report.html
    testng-failed.xml
  .gitignore
  ExtentReport.html
  pom.xml
  testNG.xml
External Libraries
Scratches and Consoles
```

4. Framework Components

4.1 Configuration

The `config.properties` file stores global settings such as URLs and credentials:

```
url= https://demo.guru99.com/insurance/v1/index.php

email=minolirashmitha228@gmail.com

password=200055
```

4.2 Base Class

`BaseClass.java` handles setup and teardown methods with support for cross-browser testing:

```
@BeforeMethod
@Parameters("browser")
public void setup(String browser)
{
   if (browser.equals("chrome")) {

       driver = new ChromeDriver();
       driver.manage().window().maximize();
       driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

   } else if (browser.equals("firefox")) {

       driver = new FirefoxDriver();
       driver.manage().window().maximize();
       driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

   }

   driver.get(webUrl);
}

@AfterMethod
public void close() {
   // Close the browser
   driver.quit();
}
```

4.3 Page Object Model (POM)

Each page is represented as a separate class with locators and reusable methods:

```
public LoginPage(WebDriver driver) {
       this.driver = driver;
   }

    By emailField = By.id("email");
    By passwordField = By.id("password");
    By submitButton = By.cssSelector("input[value='Log in']");

   public void login(String email, String password) {
       driver.findElement(emailField).sendKeys(email);
       driver.findElement(passwordField).sendKeys(password);
       driver.findElement(submitButton).click();
   }

   // Methods
   public void enterEmail(String email) {
       driver.findElement(emailField).sendKeys(email);
   }

   public void enterPassword(String password) {
       driver.findElement(passwordField).sendKeys(password);
   }

   public void clickLogin() {
       driver.findElement(submitButton).click();
   }

}
```

4.4 Test Cases

Test classes extend the base test class and use TestNG annotations:

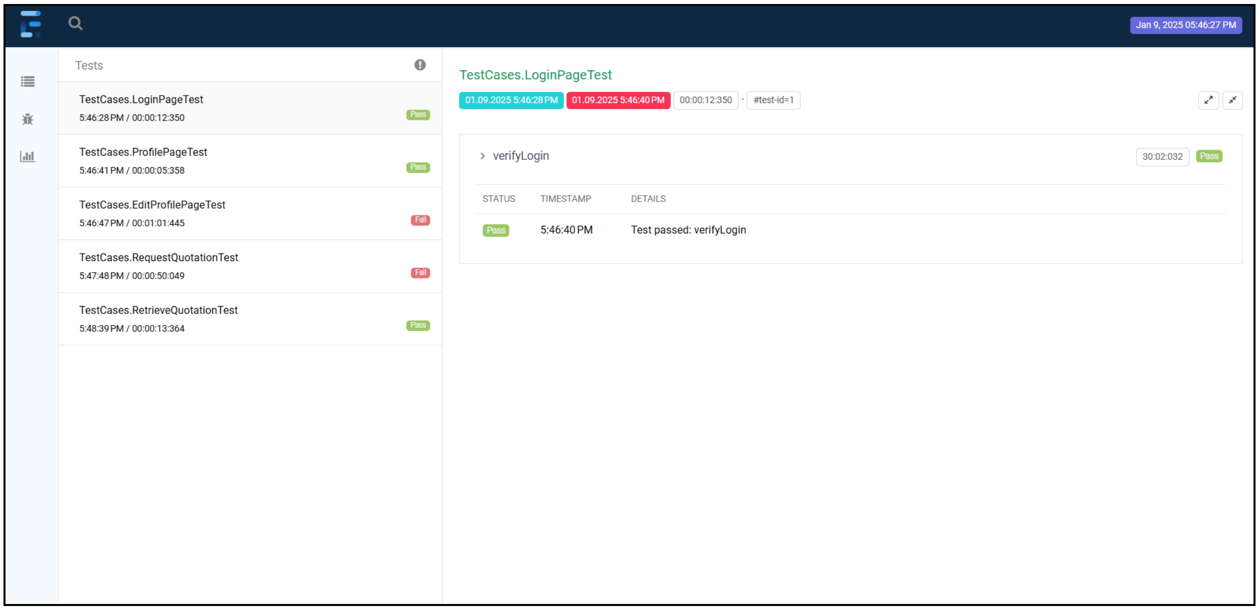
```
public class LoginPageTest extends BaseTest {

   @Test(priority = 0)
   public void verifyLogin() {
       LoginPage loginPage = new LoginPage(driver);
       loginPage.enterEmail(email);
       loginPage.enterPassword(password);
       loginPage.clickLogin();

       // Verify Login success
       boolean isProfileOptionVisible =
driver.findElement(By.cssSelector("#profile")).isDisplayed();
       Assert.assertTrue(isProfileOptionVisible, "Login failed: Profile option
is not visible");
   }
}
```

4.5 Reporting

The **`Reporting.java`** utility integrates ExtentReports for customized test execution reports.

```
String repName;

   public void onStart(ITestContext context) {

       String timeStamp = new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss").format(new Date());

       repName = "Test-Report-"+timeStamp+".html";

       sparkReporter = new ExtentSparkReporter( "./reports/"+repName);

       sparkReporter.config().setDocumentTitle("Automation Report of Guru99");

       sparkReporter.config().setReportName("Functional Testing of Insuarance Project");

       sparkReporter.config().setTheme(Theme.DARK);

       extent = new ExtentReports();

       extent.attachReporter(sparkReporter);

       extent.setSystemInfo("Application" , "Guru99");
```

```
        extent.setSystemInfo("Computer Name" , "localhost");
        extent.setSystemInfo("Environment" , "Test");
        extent.setSystemInfo("User Name" , System.getProperty("user.name"));
        extent.setSystemInfo("Tester Name" , "Minoli");
        extent.setSystemInfo("OS" , "Windows11");
        extent.setSystemInfo("Browser Name" , "Chrome");
    }
public void onTestSuccess(ITestResult result) {
        test = extent.createTest(result.getTestClass().getName());
        test.log(Status.PASS , "Test case Passed is : " + result.getName());
    }
    public void onTestFailure(ITestResult result) {
        test = extent.createTest(result.getTestClass().getName());
        test.log(Status.FAIL , "Test case Failed : " + result.getName());
        test.log(Status.INFO , result.getThrowable().getMessage());
```



Jan 9, 2025 05:46:27 PM

Tests

TestCases.LoginPageTest
5:46:28 PM / 00:00:12:350 Pass

TestCases.ProfilePageTest
5:46:41 PM / 00:00:05:358 Pass

TestCases.EditProfilePageTest
5:46:47 PM / 00:01:01:445 Fail

TestCases.RequestQuotationTest
5:47:48 PM / 00:00:50:049 Fail

TestCases.RetrieveQuotationTest
5:48:39 PM / 00:00:13:364 Pass

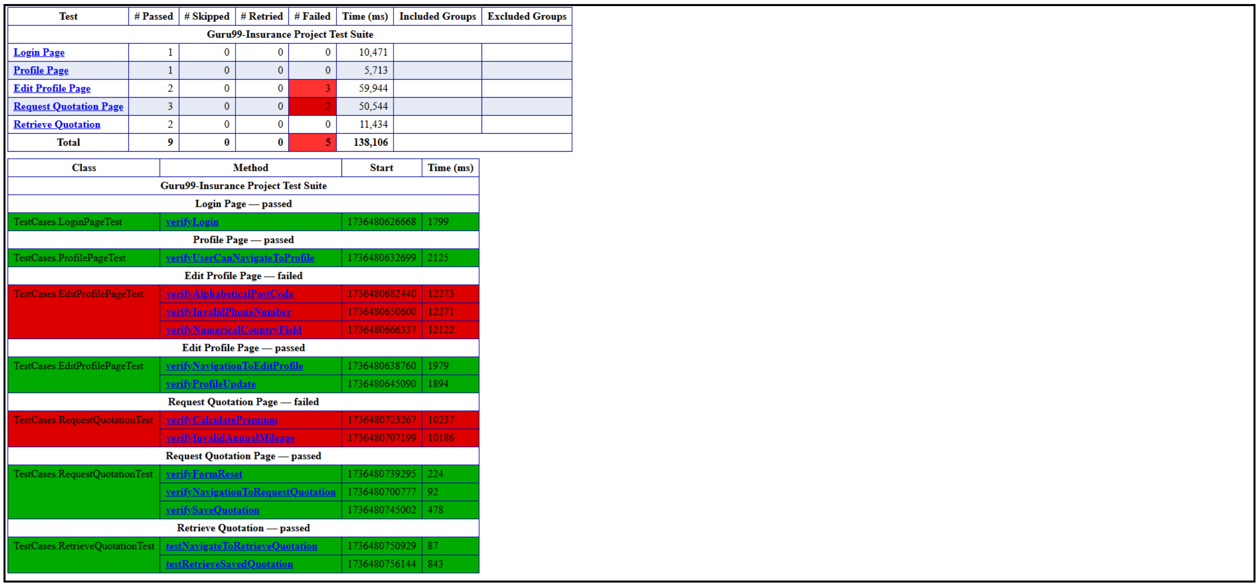
TestCases.LoginPageTest

01.09.2025 5:46:28 PM 01.09.2025 5:46:40 PM 00:00:12:350 · #test-id=1

› verifyLogin 30:02:032 Pass

STATUS	TIMESTAMP	DETAILS
Pass	5:46:40 PM	Test passed: verifyLogin

Extent Report



Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Guru99-Insurance Project Test Suite							
Login Page	1	0	0	0	10,471		
Profile Page	1	0	0	0	5,713		
Edit Profile Page	2	0	0	3	59,944		
Request Quotation Page	3	0	0	2	50,544		
Retrieve Quotation	2	0	0	0	11,434		
Total	**9**	**0**	**0**	**5**	**138,106**		

Class	Method	Start	Time (ms)
Guru99-Insurance Project Test Suite			
Login Page — passed			
TestCases.LoginPageTest	verifyLogin	1736480626668	1799
Profile Page — passed			
TestCases.ProfilePageTest	verifyUserCanNavigateToProfile	1736480632699	2125
Edit Profile Page — failed			
TestCases.EditProfilePageTest	verifyAlphabeticalPostCode	1736480682440	12273
	verifyInvalidPhoneNumber	1736480650600	12271
	verifyNumericalCountryField	1736480666337	12122
Edit Profile Page — passed			
TestCases.EditProfilePageTest	verifyNavigationToEditProfile	1736480638760	1979
	verifyProfileUpdate	1736480645090	1894
Request Quotation Page — failed			
TestCases.RequestQuotationTest	verifyCalculatePremium	1736480723267	10237
	verifyInvalidAnnualMileage	1736480707199	10186
Request Quotation Page — passed			
TestCases.RequestQuotationTest	verifyFormReset	1736480739295	224
	verifyNavigationToRequestQuotation	1736480700777	92
	verifySaveQuotation	1736480745002	478
Retrieve Quotation — passed			
TestCases.RetrieveQuotationTest	testNavigateToRetrieveQuotation	1736480750929	87
	testRetrieveSavedQuotation	1736480756144	843

TestNG Report

5. Test Execution

5.1 Running Tests

Update the `testng.xml` file to configure browser parameters and test classes:

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="Guru99-Insurance Project Test Suite">

   <listeners>
           <listener class-name="Utilities.Reporting" />
   </listeners>

   <test name="Login Page">
       <parameter name="browser" value="chrome"/>
       <classes>
           <class name="TestCases.LoginPageTest"></class>
       </classes>
   </test>
```

6. Maintenance

- **Adding New Test Cases**: Create a new class under the `TestCases` package, extend `BaseClass`, and write tests using TestNG annotations.
- **Adding New Pages**: Create a new class under the `Pages` package and define locators and methods for the new page.
- **Updating Dependencies**: Modify the `pom.xml` file to keep libraries up-to-date.

7. Conclusion

This framework provides a robust structure for testing the Guru99 demo application. It is modular, scalable, and integrates essential features like cross-browser testing and reporting.

With proper maintenance, the framework can be extended to cover additional scenarios and applications.