**Design Document**

**Mercury Tours**

(Automation using Java-Selenium-TestNG)

Owner: Mohsin Mulla

Version: 1.0

**Table of Contents:**

1. Introduction to Design document
2. Software and hardware requirements
3. Architecture and Design diagram
4. Control, Execution flow
5. Reports
6. Test cases
7. **Introduction to Design document**

This design document provides the design details of the automation framework for the Mercury Tours website (<http://newtours.demoaut.com/mercurywelcome.php>).

This document gives a detailed description of the automation framework architecture of the flight booking system. It specifies the structure and design of some of the modules. It also displays flow diagram.

Framework is design by using Java-Selenium-TestNG features which covers functional test case scenario execution for the above website.

The design document contains below,

* Software requirements.
* Folder structure
* Folder contents and use of each.
* POM (Page Object Model)
* Test Cases
* Test Data
* Utilities
* Logs and Reports
* Control and execution flow.

1. **Software Requirements**

The software used are with the below configuration.

1. Eclipse Kepler Service Release 2
2. Java JDK 1.8
3. Selenium 3.5.3 and Chromedriver 2.33
4. TestNG 6.11
5. POI 3.14
6. Extent Reports 2.41.2
7. MS Excel (for Test data)
8. **Architecture and Design diagram**

**.xlsx Data**

**.png Screenshots**

**Logs/Reports**

**.html**

**Utility**

Test Data, Screenshots, TestNG Listener etc.

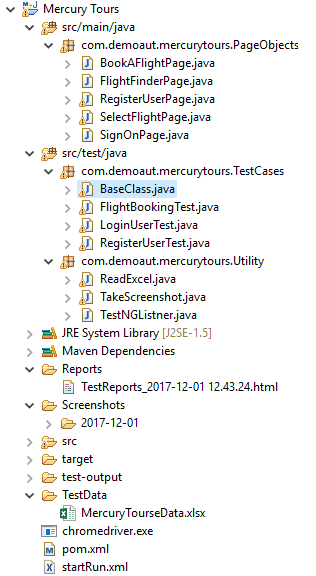
**POM** (Pages.java)

**Base Class**

**Test Cases**

**startRun.xml**

Run as TestNG



Java classes which will read test data, take screenshot and listen to the test events.

Maven is used to save all the dependencies.

Test reports will be generated here using extent reports.

Test data stored in .xlsx

Screenshots will be taken only if test case failed.

Test cases are written here

POM concept is used to store page objects

This is TestNG.xml which will trigger and control the automation flow.

Above is the folder structure of the automation framework.

**Page Object Model (POM):** As described in above structure POM with page factory design pattern is used for the framework design which will help in the modularity of the code.

***com.demoaut.mercurytours.PageObjects:*** package defines all the page objects and page specific methods as,

*BookAFlightPage.java* > contains web elements on ‘Book A Flight’ page and methods for entering data from .xlsx and a method for final purchase button which will just a click on ‘Secure Purchase’ button.

*FlightFinderPage.java* > contains web elements on ‘Flight Finder’ page and a method for entering flight details to search flight.

*RegisterUserPage.java* > contains web elements on ‘Register’ page and a method for entering user details to register user.

*SelectFlightPage.java* > contains web elements on ‘Select Flight’ page which allows user to select departing and returning flight and a method to click on continue button.

*SignOnPage.java* > contains web elements on ‘Select Flight’ page which allows user to enter username and password and a method to click on submit button.

**com.demoaut.mercurytours.TestCases:** package for all the test case execution script including validations of test output.

*BaseClass.java* > It’s a parent class. This class is extended by all the other test case classes.

*FlightBookingTest.java* > this test case will perform end to end scenario for flight booking.

*LoginUserTest.java* > there are two scenario included in this class file. Login with valid username and password and Login with invalid username and password.

*RegisterUserTest.java* > this test will perform register new user scenario.

*VerifyItinerary.Java* > this test will check booking history and verify itinerary

**com.demoaut.mercurytours.Utility:** package contains utilities like read data from .xlsx, take screenshot and TestNGListener.

**Reports:** .html Extent reports along with step wise logs are generated here. The folder is created under root folder of the project. Name of the report .html is currently configured as TestReport\_<system date><system time>

**Screenshots:** Screenshots will be captured here only when the test cases failed. Name of the screenshot is the method name <system time>.

**TestData**: Test data will be taken from .xlsx stored in this folder.

*pom.xml*: It is an XML file that contains information about the project and configuration details used by Maven to build the project.

**Note:** Maven is used for folder structure and to maintain common dependencies only.

*startRun.xml*: This is a testNG.xml file which defines the execution, sequence of classes and test cases.

1. **Control and execution flow**

startRun.xml will trigger the execution of test cases as per the class files mentioned in the xml. It will start running the classes sequentially.

Test case classes are extended to Base Class which has @BeforeTest, @AfterTest, @BeforeMethod and @AfterMethod methods.

2D array object is created to retrieve the data from .xlsx and pass the data to POM methods.

Extent report started and ended after every test case to record the results and logs are added step level.

After response from the POM the validations are performed by using Assertion.

If the test case fails, onTestFailure method gets called from TestNGListener class which will trigger the getsScreenhot method from takeScreenshot.java class and screenshot will be captured and stored with method name\_<system time>.

After completion of the test execution extend reports will be generated with the name TestReport\_<system date><system time>

1. **Test Cases**

There are 4 test cases scenarios identified and 4 test classes are created,

1. **LoginUserTest >** Two test cases are created for Login scenario.
2. **Login with valid username and password**

Steps:

* 1. Navigate to <http://newtours.demoaut.com/mercurywelcome.php>
  2. Click on ‘SIGN ON’ link
  3. Provide valid username and password
  4. Click on ‘SUBMIT’ button.

Expected: User should be logged in successfully.

1. **Login with In-valid username and password**

Steps:

* 1. Navigate to <http://newtours.demoaut.com/mercurywelcome.php>
  2. Click on ‘SIGN ON’ link
  3. Provide In-valid username and password
  4. Click on ‘SUBMIT’ button.

Expected: User should NOT be logged in successfully.

1. **RegisterUserTest >** To verify new user registered successfully.
2. Navigate to <http://newtours.demoaut.com/mercurywelcome.php>
3. Click on ‘REGISTER’ link
4. Provide registration details
5. Click on ‘SUBMIT’ button.

Expected: User should be registered successfully.

1. FlightBookingTest > To verify user can complete flight booking successfully
   1. Navigate to <http://newtours.demoaut.com/mercurywelcome.php>
   2. Click on ‘SIGN ON’ link
   3. Provide valid username and password
   4. Click on ‘SUBMIT’ button.
   5. Provide Flight Finder details to search the flight
   6. Select depart and return flight options by checking radio buttons
   7. Click on ‘Continue’ button
   8. Provide passengers and transaction details on ‘Book a flight’ page
   9. Click on ‘Secure Purchase’ button
   10. Check if user navigates to ‘Flight Confirmation’ page.

Expected: User can book a flight successfully.

1. VerifyItinerarytest > To check and verify itinerary
   1. Navigate to <http://newtours.demoaut.com/mercurywelcome.php>
   2. Click on ‘SIGN ON’ link
   3. Provide valid username and password
   4. Click on ‘SUBMIT’ button.
   5. Click on ‘ITINERARY’ link
   6. Verify the correct navigation for itinerary page
   7. Also verify the itinerary data is available on the page.

Expected: Itinerary data is displayed on the page

**Note:** Please refer user guide ‘*README.txt’* from project folder to execute automation.

For any queries please write to: Mohsin.Mulla@mindtree.com