**Name: Kruthi Venkatesh 11/11/2016**

**SJSU ID: 010120422 CMPE 287**

**Individual Test Automation Project**

Table of Contents

[Test Plan Identifier. 1](#_Toc466650669)

[Introduction/Test Objective 1](#_Toc466650670)

[Test Items. 2](#_Toc466650671)

[Test Strategy 2](#_Toc466650672)

[Tools 2](#_Toc466650673)

[Test Data 3](#_Toc466650674)

[Environment 3](#_Toc466650675)

[Risk and Assumptions 4](#_Toc466650676)

[Report and Results 4](#_Toc466650677)

[Integration/Scheduling 5](#_Toc466650678)

[Automation Script 5](#_Toc466650679)

**Yelp Login Page** ( <https://www.yelp.com/login> )

This page is for the registered users with valid credentials to log in to Yelp. I choose this Yelp login page for automation as it has all the fields required to implement the functional validation, validation of web elements, programmatic solution, object oriented design, string manipulation and data driven techniques.

**Test Plan**

# Test Plan Identifier.

YLP-ATP01.00

# Introduction/Test Objective

This automation plan is created as a part of assignment for the course CMPE 287. This automation plan concentrates on the login page of the Yelp website. After one-time registration to Yelp, user can login to Yelp using this page by providing details such as email and password or using Facebook login.

This document concentrates on the automation of login form for various inputs and illustrates the good web automation techniques such as object oriented techniques, Data driven techniques, programmatic approaches such as use of loops and conditional statements, String manipulation such as string comparison. Since login need to be tested for different inputs both valid and invalid with various email and password combinations, this is the right candidate for automation.

This document includes an automation script written in Java using Selenium Web Driver. All the web automation techniques are explained in detail with reference to the script.

# Test Items.

* Login form

# Test Strategy

**Type of Automation**

Selenium web driver is used for automation testing as this is GUI testing.

**Naming Convention**

* test[Feature being tested]: This naming convention is used to name the tests for the features being tested. Example: testLoginForm
* Other naming conventions are same as Java naming conventions. Example:
  + CamelCase for class names e.g. YelpLogin
  + Mixed case for variables e.g. driver, baseUrl etc.

**Automation Framework**

Framework used is hybrid, which is combination of

* Data driven technique - uses properties file to supply the test data for email and password for testing the login form.
* Object oriented – Checks the button click after entering the data to text box. Hence testing the two objects button and textbox.
* Programmatic – uses conditional statement such as if else to check the successful login success/fail and uses while loop to iterate through the test data.

uses try catch block to capture the exceptions such as FileNotFoundException, ElementNotVisibleException etc.

**Preconditions**

* To test the automation script need to have following downloaded
  + jar file for selenium web driver (http://www.seleniumhq.org/download/)
  + executable geckodriver.exe to work with selenium 3.0 with firefox (https://github.com/mozilla/geckodriver/releases)
* If logging is enabled, then download jar log4j (https://logging.apache.org/log4j/1.2/download.html)

# Tools

* Selenium web driver is the automation tool used to automate the Yelp login. Since selenium supports different platforms we can extend this automation in future to support all the platforms.
* Junit is used to create unit tests as we are testing only login form. In future if we want to extent the automation for entire Yelp application, we can just go on adding the tests for different features as unit tests.

**Note**: I wanted to add random email and password generator to test different possible combination of email and password strings. Due to the small script constraint and to show the data driven technique not involved the random email and password generator.

# Test Data

To make this automation data driven the login credentials are stored in file with properties as extension and using while loop data is fed to the login form to check for valid and invalid login

**Data source**

loginData.properties – Properties are configuration values managed as key/value pairs. In each pair, the key and value are both [String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) values. The key identifies, and is used to retrieve value i.e. email5 is key and [kruthi.vijay31@gmail.com](mailto:kruthi.vijay31@gmail.com) is the value.

**Data stored in loginData.properties**

totalTests=5

email1=jhdu

pwd1=1234

email2=gmail.com

pwd2=12df

email3=

pwd3=

email4=l.gmail@.com

pwd4=12\_f@12

email5=kruthi.vijay31@gmail.com

pwd5=\*\*\*\*\*\*\* Note: Not disclosing the valid credential

**File Location:**

("/Users/Kruthi/Documents/CMPE 287/Kruthi/Assignments/Test\_Automation/Yelp\_Automation/src/yelp\_testdata/loginData.properties

**Data Type:**

* String for both email and password
* Int for total number of tests

# Environment

General Environment for Selenium automation with Firefox web browser and Java as coding language with logs.

* Any computer with Firefox browser support. (Since we are using Firefox web driver)
* Eclipse IDE
* Java (JDK)
* Selenium Java Client driver
* Logger for logging data

Specific Environment used for automation of yelp login page are

* **Machine**
  + MackBook Pro, OS X EI Capitan Version 10.11.3
  + Processor 2.6GHz Intel Core i5
  + Memory 2.6GB 1600 MHz DDR3
  + Graphics Intel Iris 1536MB
* **Browser**
  + Firefox 49.0.2
* **Development Environment**
  + Eclipse Java EE IDE for Web Developers.
  + Version: Mars.2 Release (4.5.2)
  + Build id: 20160218-0600
* **Selenium**
  + selenium-server-standalone-3.0.1.jar
* **Logger and Property configuration** 
  + log4j-1.2.17.jar (Note: Not included in script to keep it short)
* **Java**
  + JavaSE-1.8

# Risk and Assumptions

* If firefox without geckodriver is used with selenium3.0, then throws error.
* If there is version miss match between the firefox and selenium, then automation script will not run.
* If the firefox is down, then testing cannot be done hence its assumed firefox is always up and running.
* To test the login button xpath is used. If in future Yelp releases a new version with UI modification, then this path need to be changed for this script to produce the expected results without any error.

Note: using xpath as there is no ID for the login button

<button class="ybtn ybtn--primary submit ybtn-full" type="submit" value="submit">

<span>Log In</span>

</button>

# Report and Results

**Test Case**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Feature** | **Test Description** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 1 | Login Email Textbox | Test if user can enter data into email textbox i.e. its not readonly | User should be able to enter data into email textbox | Was able to write to email textbox | Pass |
| 2 | Login Password Textbox | Test if user can enter data into password textbox i.e. its not readonly | User should be able to enter data into Password textbox | Was able to write to password textbox | Pass |
| 3 | Login Button | Test if user is able to click on login button i.e. check button is not disabled | User should be able to click on the button | Was able to click the login button | Pass |
| 4 | Textbox Error for empty fields | Test for error message such as Please fill out email/password field if user click on login button without entering data | User should see the error message if text box is empty | Was able to see the error message when empty email and password was passed as input | Pass |
| 5 | Valid Login | Test if user is able to login with correct credentials | User should be directed to home page | Was able to login with valid credentials | Pass |
| 6 | Invalid Login | Test if user is not able to login with incorrect credentials | User should get error message and should not be redirected to home page | Got error message and was not redirected to home page | Pass |

**Results**

Below are the test results for the login form printed on the eclipse console using the data from the file which has five login credentials with four being invalid and one being valid. (refer test data section in this document for the data)

“Summary of test

Total number of tests 5

Total number of successful login from the file input is 1

Total number of login failed from the file input is 4”

**Report**

From the results, its seen that the automation script is working as expected. The test data file was feeding four invalid credentials and one valid credentials to the email and password textboxes hence one successful login and four unsuccessful logins are printed in the console.

**Note:** Even loggers can be used to log the results, to keep the script short, logger is not added in the script.

# Integration/Scheduling

No Integration plans involved as this automation plan tests only one test item and hence uses unit test case approach.

# Automation Script

**package** com.yelp.automation;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.IOException;

**import** java.util.Properties;

**import** java.util.concurrent.TimeUnit;

**import** org.junit.After;

**import** org.junit.Before;

**import** org.junit.Test;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.ElementNotVisibleException;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** YelpLogin {

//create private variables

**private** WebDriver driver;

**private** String baseUrl;

**private** Properties dataForTest;

//set up driver, file path and define url

@Before

**public** **void** congfigure() **throws** IOException{

System.*setProperty*("webdriver.gecko.driver","/Users/Kruthi/Documents/CMPE 287/Kruthi/Assignments/Test\_Automation/geckodriver.exe");

File file = **new** File("/Users/Kruthi/Documents/CMPE 287/Kruthi/Assignments/Test\_Automation/Yelp\_Automation/src/yelp\_testdata/loginData.properties");

FileInputStream fileInput = **null**;

//illustrates programmatic approach by using try catch

**try** {

fileInput = **new** FileInputStream(file);

} **catch** (FileNotFoundException e) {

e.printStackTrace();

}

dataForTest = **new** Properties();

//load properties file

**try** {

dataForTest.load(fileInput);

} **catch** (IOException e) {

e.printStackTrace();

}

driver = **new** FirefoxDriver();

driver.manage().timeouts().implicitlyWait(5, TimeUnit.***SECONDS***);

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

baseUrl = "https://www.yelp.com/login";

}

//test login form for different email and password

@Test

**public** **void** testLoginForm(){

// getting the number of test cases stored in file for testing login form

**int** numberOfTests = Integer.*parseInt*(dataForTest.getProperty("totalTests"));

//initialize variables

**int** i = 1, successLogin = 0, failLogin = 0, totalTests = numberOfTests;

//test for all the classes - **illustrates looping**

**while** (numberOfTests > 0) {

driver.get(baseUrl);

//**illustrates the object oriented concept** using button and textbox, data driven using file to read data

**try**{

driver.findElement(By.*id*("email")).sendKeys(dataForTest.getProperty("email" + i));

driver.findElement(By.*id*("password")).sendKeys(dataForTest.getProperty("pwd" + i));

//driver.findElement(By.xpath("//\*[@id='ajax-login']/button[@type='submit' and span='Log In']")).click();

driver.findElement(By.*xpath*("//form[contains(@action,'/login')]//button[@type='submit' and span='Log In']")).click();

//illustrates string manipulation- gets the title and compare with the string

String pageTitle = driver.getTitle();

}

**catch**(ElementNotVisibleException e){

System.***out***.println(e);

}

// checking for login page title

//illustrates the conditional statement

**if**(pageTitle.equals("San Jose Restaurants, Dentists, Pubs, Beauty Salons, Doctors")){

System.***out***.println("Login Success for the inputs " + "email" + i + " , " + "pwd" + i);

successLogin++;

driver.navigate().back();

}

**else**{

System.***out***.println("Login Fail for the inputs " + "email" + i + " , " + "pwd" + i);

failLogin++;

}

numberOfTests--;

i++;

}

System.***out***.println("Summary of test");

System.***out***.println("Total number of tests " + totalTests);

System.***out***.println("Total number of successful login from the file input is " + successLogin);

System.***out***.println("Total number of login failed from the file input is " + failLogin);

}

@After

**public** **void** close() {

driver.quit();

}

}