ECSE 428 – Assignment B
Cucumber and Selenium Test Automation

Julien Courbebaisse – 260614548

March 11th 2019

Summary of Deliverables:

Story Statement

Story:

As a Gmail user

I would like to be able to send emails with attachments

So I can share pictures with my friends

Normal Flow: Sending an email with recipient, subject, message, and image attachment

Given I am on my inbox page

When I click on compose
And I enter recipient
And I enter title
And I enter message
And I select an image
And I click send
Then the email should be sent

Alternate Flow: Sending an email with recipient, subject, message, but no attachment

Given I am on my inbox page

When I click on compose
And I enter recipient
And I enter title
And I enter message
And I click send
Then the email should be sent

Alternate Flow: Sending an email with recipient, attachment, message, but no title

Given I am on my inbox page

When I click on compose
And I enter recipient
And I enter message
And I select an image
And I click send
Then the email should be sent

Error Flow: Sending an email with title, attachment, message, but no recipient address

Given I am on my inbox page

When I click on compose
And I enter title
And I enter message
And I select an image
And I click send
Then the email should not be sent

Environment Description

General Setup:

The project uses jdk8u201 for java. The IDE used to develop the program was Eclipse Neon 2018-2019. It features good Maven integration and is recommended for this project. Maven was also used in the project. Find the Maven dependencies below. The tests were built on pc and are Windows specific. The browser used here was Google Chrome version 72.0.3626.21 and Selenium-Java version 3.141.0. The automation was guided with Cucumber, find more information in the appropriate section. Finally, AutoIt was used to develop an executable file for selecting an image attachment.

Maven:

Install the latest version of Maven available through the described Eclipse distribution. Navigate the the pom.xml file and make sure the following depencies are listed:

```
<dependencies>
    <dependency>
   <groupId>junit</groupId>
   <artifactId>junit</artifactId>
   <version>4.10</version>
   <scope>test</scope>
</dependency>
    <dependency>
   <groupId>org.seleniumhq.selenium</groupId>
   <artifactId>selenium-java</artifactId>
   <version>3.141.0</version>
</dependency>
<dependency>
   <groupId>info.cukes</groupId>
   <artifactId>cucumber-java</artifactId>
   <version>1.0.2</version>
   <scope>test</scope>
</dependency>
<dependency>
   <groupId>info.cukes</groupId>
   <artifactId>cucumber-junit</artifactId>
   <version>1.0.2</version>
   <scope>test</scope>
</dependency>
  </dependencies>
```

Observe the version numbers on Junit , Cucumber-java, and Cucumber-Junit.

Cucumber:

The logical instructions (written in Gherkin Syntax) from Cucumber are stored in the “sendemail.feature” file in “src/test/java”. The Cucumber annotation are then used in the sendemail.java file to descrive the flow of the test execution.

Our project has a way of seting up before each test and returning to its original state after each test. The way it sets up (opens browser and logs in to gmail every time) is through the “Background” keywod in our feature file. It associates a mini scenario to be executed before running each test. The corresponding step definitions are implemented in sendemail.java. The method to shut down browser is not present in the feature file. It uses the annotation @after in the sendemail.java file and cucumber automatically knows to executes this method after each test.

If one wanted to add scenarios, one would need to first create a new scenario (or scenario outiline) in sendemail.feature and the possibly complete the sendemail.java file with any new uncovered steps. To add examples to a scenario outline one would simply need to locate the right scenario outline and add the example data in the appropriate Examples section. Also note that to run the tests one would need to update the image paths in the .feature file.

AutoIt:

It should be noted that AutoIt is not necessary in itself to run the project. However if the current code became deprecated, it could lead to errors and so its workings should be explained. AutoIt enables the user to write a script which can then be compile to an executable file. In this case it used to navigate the folder explorer when selecting an attachment for the email. Find below the AutoIt script (before compilation) used for this project. Also note that one would need to update the path to the executable to match one’s own system before running tests.

```
WinWaitActive("Open")

Send($CmdLine[1])

Send("{ENTER}")
```

Cucumber Gherkin Scripts

```
Feature: sendemail

Background: User is on his inbox homepage
       Given User navigates to gmail signin page
       When User signs in to gmail as "assignmentB260614548" and pwd "ecse428ab"
       Then User should be on inbox homepage

#Scenario send self email without image (alternate)
Scenario Outline: Send email without image attachement
   When I click on compose
   And I enter recipient as "<to>"
   And I enter title as "<title>"
   And I enter message as "<message>"
   And I click send
   Then the email should be sent

Examples:
       |to|title|message|
       |assignmentb260614548@gmail.com|Hello|Hi from selenium/cucumber without image|
       |allockicmoi@gmail.com|Hello| test for assignment|



#Scenario send email with image
Scenario Outline: Send email with image attachement
   When I click on compose
   And I enter recipient as "<to>"
   And I enter title as "<title>"
   And I enter message as "<message>"
   And I select an image from "<imagesource>"
   And I click send
   Then the email should be sent

 Examples:
       |to|title|message|imagesource|
       |assignmentb260614548@gmail.com|Hello|Hi from selenium/cucumber with an
image!|C:\\Users\\alloc\\Documents\\image|
       |assignmentb260614548@gmail.com|Another Email!|I've attached a photo,check it
out!|C:\\Users\\alloc\\Documents\\rengar|
       |assignmentB@gmail.com| | test with empty
subject|C:\\Users\\alloc\\Documents\\rengar|

#Scenario error flow attempt to send email without recipient
Scenario:
       When I click on compose
       And I enter recipient as ""
       And I enter title as "Error flow message"
       And I enter message as "will this send?"
       And I select an image from "C:\Users\alloc\Documents\rengar"
       And I click send
       Then the email should not be sent
```

Source Code

The source code for the project can be found at :
https://github.com/allockicmoi/SendGmailTestAutomation

Step definitions can be found under:

- src/test/java/sendemail/sendemail.java

Feature file can be found at:

- src/test/java/sendemail/sendemail.feature

Structure and file contents

```
SendGmailTestAutomation
        |
        --- src/main/java
                |
                --- com.test.maven.assbtry2
                        |
                        *  App.java              // mock app file generated by Maven template
        |
        --- src/test/java
                |
                --- com.test.maven.assbtry2
                        |
                        *  AppTest.java     //mock test app
                --- sendemail
                        |
                        *  runTest.java      // test run (run as junit)
                        * sendemail.java  // contains our java step defintitions
                        * sendemail.feature //cucumber scenarios and background
```

* uploadfile3.exe // the provided executable used in a step definition

|

---JRE System Library //self explanatory

---Maven Dependencies   //self explanatory

|

* pom.xml   //xml file containing our maven dependencies

Instructions for Environment Installation and Running the Tests

1- First thing to do is install Java 8 JDK from https://www.oracle.com/technetwork/java/javase/downloads/index.html
2- Download and install Eclipse from https://www.eclipse.org/downloads/ (Eclipse Java)
3- Assuming you have Git installed, clone the repository from https://github.com/allockicmoi/SendGmailTestAutomation into desired workspace
4- Open cloned project in Eclipse
5- Navigate to pom.xml file
6- Make sure dependencies match with dependencies listed in environment description
7- In Eclipse, Go to Project → Clean (wait a couple minutes to complete)
8- In sendemail.feature, modify the image paths in Examples to match the location of your own files
9- Download ChromeDriver from http://chromedriver.chromium.org/downloads
10- In sendemail.java, line 33, update the path to the correct chromedriver path
11- In sendemail.java, line 93, update the path to the correct .exe path for uploadfile
12- You should be ready to go! Now right click on runTest.java → Run As Junit test
13- Watch the test execute in the browser!

Video Demonstration

Although the video is too big to be stored on the GithHub repo, please find it attached in mycourses submission. It shows the successful execution of all the tests in the brower.