Jack Gallaher

Automate ryanair booking page test

**INTRODUCTION**

For this test I was given a task of automating the Ryanair booking website. Here the task includes booking a flight but, it must fail on the payment page by entering incorrect details on the bank card.

To complete this task, I used certain tools in order to make it work. These tools include **Java** as the programming language and **Eclipse** as the IDE. I had to download Chrome Driver in order to access Google chrome when running the test. I also used **Cucumber BDD** software. This is an automation testing tool that follows behaviour driven development. This is based on analysing the behaviour of an application. It follows a structure known as the **Gherkin** structure. This is a type of language that is English based. The benefits of the Gherkin structure is to allow technical and non-technical people to understand the functionality of an application. It uses certain key words such as GIVEN, AND, THEN, WHEN. An example as follows:

**Given** I make a booking from “DUB” to “SXF” on 12/03/2017 for 2 adults and 1 child

**When** I pay for booking with card details “5555 5555 5555 5557”, “10/18” and “265”

**Then** I should get payment declined message.

I also used a page object pattern in the test. This is a design pattern that is used to create object repository for Web UI elements. Under this model for each web page in the application, there should be a corresponding page class. This page class will find the Web Elements of that web page and also contains page methods which perform operations on those elements.

**MY EXAMPLE**

Feature: Declined payment for flight

Scenario: Checking to see if booking flight fails

Given browser is open

And user is on booking flight page

When user enters in their flight information (date, one way trip, one adult, desintation, departure location)

And selects search button

Then user is navigated to select flight page

And user selects flight

When user enters in passenger details

Then user is navigated to select seat page

Given user selects seat

And clicks continue

Then user will be directed to luggage page

Given user selects one small bag

And clicks continue

Then user will be directed to insurance page

Given User selects continue

Then user will be directed to overview page

When user selects basket

And selects check out

Then user should be directed to Payment page

When user selects in payment information

And selects pay now

Then Payment should fail

**IMPORTANT NOTE: The automation test does not run due to an error which could not be resolved.**

To run the scripts you must run the script via the Eclipse IDE (or your own IDE). You must run the **BookingFlight.feature file.** This should open the Google Chrome browserand direct the user to the Ryanair page where the test shall begin. How this works is, is that the keywords in the Gherkin script such as **GIVEN** are connected to the test case under the name **@Given** in the step definitions file named BookingFlightSteps.java. (Please see images below) When the script is ran and has passed through the test with the word Given, it would look for it in the step definitions file and run the Java code.

Feature File



Step Definitions file

Graphical user interface, text, application

Description automatically generated

The code we see above is creating the Driver Object as we need it, in order to access Chrome. We also have the path set for the Driver application as seen in the image above with the C:\\Users path

**Explanation**

Text, letter

Description automatically generated

By examining the script above we will see how this runs. When you are directed to the page to enter in your information for booking your flight, there text fields and boxed to fill out. Things like Date, One or return, Destination, etc. The script picks up this elements on the web page by their HTLM Tags. For example ID, each element on the web page has an ID. The word **Driver** is referring to the web driver (Chrome Driver) as we are using Google Chrome. This uses the **findElement** method to search for web page elements. As you can see from above we use ID. The ID of the Destination field on the webpage where you select your destination is named input-button\_\_destination. So therefore, it locates the element on the web page and uses the **sendKeys** Methods to enter in London as the destination. By following these methods above **driver.findElement(by.ID)** (Not just ID but also by other elements such as classname or name), this helps us locate what we are looking for on the web page. By using methods like **SendKeys** and **.Click()** we can interact with those elements by performing functions like entering in Data or clicking buttons in order to progress through the webpage. This section just explained is what the Page object model is used for. By using the methods like findElements, we look for the elements on the web page and perform an action on them.

Here we see the Test in how it should perform on the front end when it begins.

Enter Flight information

Graphical user interface, application

Description automatically generated

Select Flight

A picture containing graphical user interface

Description automatically generated

Continue

Graphical user interface, text, application, chat or text message

Description automatically generated

Enter Passenger information

Graphical user interface, website

Description automatically generated

Select Seat

Graphical user interface, application

Description automatically generated

Select Bag

Graphical user interface, application

Description automatically generated

Checkout

Graphical user interface, application

Description automatically generated

Enter in payment details followed by a failed transaction

Graphical user interface, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

As we reach the end of the test, we are presented with a failure on the payment page as we entered incorrect card details.