**Exploratory Testing**

In order to test the booking website, an exploratory test was carried out. This is because there wasn't any given requirement to test against. This approach was chosen

to determine how the application currently works and also to identify bugs currently in the application if any.

**Test Cases**

***Happy Path (written in gherkin BDD style)***

**(Create booking)**

Given a user navigates to *http://hotel-test.equalexperts.io/*

And the user enters First and last name as string inputs.

And the user enters the price as positive numbers not more than 12 digits.

And the user selects a value of true or false for deposit.

And the user enters the check in/Checkout date not in the past.

When the user clicks on save.

Then the user should be able to see the created booking displayed.

**(Delete booking)**

Given a user navigates to *http://hotel-test.equalexperts.io/*

When the user clicks delete on an existing booking

Then the deleted bookings should not be displayed

***Negative Testing (not written in gherkin)***

Verify that the user is not able to submit request with numbers saved as names

Verify that the user is not able to save request with a negative price amount

Verify that the user is not able to save request when check-in date is in the past

Verify that the user is not able to save request when checkout date is in the past

Verify that the user is not able to save request when there are special characters in the names field

Verify the input fields does not overlap each other but adjusts based on input size

Verify that dates entered are validated against required format

**Bugs on the UI**

The user can save a negative price value

The user can enter digits (numbers) in the first and last name field

There are no validation on dates, the user can check in and checkout in the past

The user can check-in in the future and checkout in the past

Random numbers such as (10, 0, 1, 66, 4) entered into the calendar defaults to 1970-01-01

The user can enter random symbols as name

The price field defaults to null after a given amount of input

The price fields overlaps the deposit field when the price digits exceeds 12

**Bugs on the API**

create booking requests yields a 200 ok instead of 201 created

Delete Boking requests yields a 201 created instead of a 204 no content

Post request with an invalid payload yields a 500 server error instead of a 400 bad request

Author: Michael Osarumwense

Senior SDET, QA Automation Engineer, Test Analyst