QA Technical Test

Task Description

The task is to create an automated test suite to verify the functionality of a hotel booking web service. The web service is quite simple and will ultimately allow users to:

- check the availability of rooms at the hotel
- make a booking at the hotel

This functionality is provided by 2 basic operations which take the form of service calls - checkAvailability and bookRoom for a given date. There is more detail on these calls below.

Your goal is to write a test suite which would cover the basic functionality through a couple of automated test cases and find any bugs as a result. The automated tests should make requests to the test server using the specified URLs in the *Server API info* section below.

To provide you with an environment to test this we have attached a zip file containing a test server which you can extract and run. The test server covers the specified functionality, but has some logical bugs, left on purpose for you to find.

The service source code is provided as Groovy scripts. Feel free to have a look inside.

The choice of the language to write the tests is entirely up to you. The main goal is that you produce an automated test suite that executes against the test server and exposes the bugs.

We are also looking some end to end web tests and are eager to see you approach to take to UI based automation. again test cases, reusability, complexity and code used will all be reviewed for overall result

QA Technical Test

Notes

- Only functional testing is required i.e. no load, performance, stress testing, etc.
- Date functionality in the BookingService Groovy scripts works as expected.
- There are some bugs in the system and you are looking to uncover those. Treat this as you would any set of requirements - don't assume anything works fully as expected.

Feel free to copy and paste your assumptions into your feedback as your comments are one area we will look at, to see how you based your results against your analysis. Any edge cases / negative cases / trying different things a user might try are a bonus.

We will be sharing out the zip file required as a Google doc as we cannot attach it in email. If you do not receive this email within minutes of getting this email, please let me know.

You can use any tools or methods you wish. We like a mix of skills in the team so feel free to tackle it any way you wish.

What's in it for you?

We are not looking for you to spend more than a couple of hours (at most) on this but that's your call. Typically, people spend 2 to 3 hours on the test.

In return for completing this, we will send you results on a variety of areas such as defects found, your approach taken, comments, etc. which people find of great benefit. In that way, you gain expert feedback on your approach which you can then use to your own benefit.

So, we feel it is worth candidates making the effort with this test. It also greatly strengthens a candidate in the interview process for a position on the TDL QA Team if we see good technical approach and effort applied.

QA Technical Test

Running the test server

Unzip the provided zip file, then run the server using provided commands:

* Unix and MacOS:

```
./run.sh
```

* Windows:

```
run.bat
```

Note: The startup script requires at least 1.6 version of JAVA to run.

The server starts on port *9090*.

Server API info

Getting room availability

The service returns room availability for a given date. Included in the response are details of the date the request was made for, the number of rooms available and the price for that specific date. Note the number of rooms should not go below zero.

Service URL: http://localhost:9090/checkAvailability/<yyy-mm-dd>

Parameters:

- date - in format *yyyy-mm-dd*, e.g. 2014-01-01

Example request:
http://localhost:9090/checkAvailability/2013-04-11

The returned response is in *json* format, for example:

```
{
  "date" : "2013-04-11",
  "rooms_available" : 10,
  "price" : 150
}
```

QA Technical Test

Booking room

This service books a room for a specified date and number of days. Included in the response are details of the Check-In date requested, the Checkout date and the Total Price. The Check Out date is based on the number of days requested, e.g. if the number of days is only 1 the Check Out date would be the next day after the Check-In Date.

The total price is based on the price of the room for each night during the booking, e.g if the first night was 150 and the second night was 160 the Total Price should be 310.

The request has to be made as POST request, in json format.

Service url: http://localhost:9090/bookRoom

Request has no URL parameters.

Example request body:

```
{"numOfDays":1, "checkInDate":"2013-04-10"}
```

and response:

```
{
   "checkInDate": "2013-04-10",
   "checkOutDate": "2013-04-11",
   "totalPrice": 130
}
```