BDD Automation Framework

The Behavioral Driven Development Automation Framework (BDD) allows automating testing on frontend E-Commerce Website and backend test of pet store using RestAssured.

Table of Contents

Description

BDD FW combines allows users to automate tests on the ECommerce website and API calls on the pet store with various scenarios mentioned in the feature file.

Installation

Prerequisites Maven should be installed and configured in environment variables by following this

Java should be installed and configured in environment variables by following this

Clone Github Repository to local The project folders and dependencies for BDD FW have been managed using maven pom and uploaded on Github. Download the latest version of the repository in your local machine using the below command.

```
git clone https://github.com/nikagrawal2000/WebAndRestAPI
```

Configuration

View Screenshot

Inside project directory, automation.properties file has already been created and pre-filled with below values:

```
WebURL=http://automationpractice.com/index.php?

browser=chrome

userName=nikagrawal774@gmail.com

password=feX.Nin3YtG2ehE

implicitTimeout=10

baseUrl=https://petstore.swagger.io/v2

appId=123456
```

Optionally, the `userName` and `password` can be set by creating an account on the eCommerce website and updating it in automation.properties: `http://automationpractice.com/index.php?`.

Project Structure

View Structure

Cucumber.options package has TestRunner features package has all feature files pages package has all classes related to ecommerce UI test commonActions has one class with the same name that contains all common actions of UI setpDefinations has all step definitions and Hooks utils have all helper classes pojo have all pojo classes for RestAssured REST API payload modification automation.properties have all configurable items.

There are two Feature Files created, PetStore.feature PlaceOrder.feature

PetStore.feature contains REST API testing feature file for PETSTORE API

PlaceOrder.feature contains UI automation testing feature file for E-Commerce website

Test Scenarios Covered

For E-Commerce Testing, a total of three test scenarios are covered using the below tags along with a detailed explanation:

View @newUserPlaceOrderBySearchingCategoryProduct This includes the creation of a new account on the UI, and then searches for products and adds to cart, and then checkout testing giving address and payment information

View @existingUserPlaceOrderBySearchingPopularProducts Similar to the first test scenario except an existing credential is used, details of which are fetched from the properties file. The products are fetched from the Popular category

View @existingUserChecksTotalOfProducts The existing user adds a total of three products, during checkout, checks and verifies the total of the products matches with the total value shown in the checkout, then deletes one product and checks that the total should reflect the change.

For PetStore, RestAssured is used. The scenarios along with tags that are covered are: View

@addNewPet Sends Post Request with payload having pet id and pet name. Verifies the response status code with pet id and name.

@getPetDetails Using GET Request, Fetches pet information and verifies response status code with pet id and name.

@updatePetInformation PUT request is sent to update Pet information with a new pet id and name. Verifies response status code with new pet id and name.

@deletePetDetails Using DELETE request, deleted pet information. Verification is done on status codes, pet name, pet id and response message.

Usage

Running ECommerce Web Test To run only UI automation, in the directory of project location, run using the below command

```
mvn test verify -Dcucumber.filter.tags="@UI"
```

Alternatively, you can also run by double clicking **runUI.bat** in the project directory

Running PetStore API Test To run PetStore API automation testing, in the directory of project location, run using the below command

```
mvn test verify -Dcucumber.filter.tags="@Backend"
```

Alternatively, you can also run by double clicking **runBackend.bat** in the project directory

Running Regression Test To run both UI and API automation testing, in the directory of project location, run using the below command

```
mvn test verify -Dcucumber.filter.tags="@Backend or @UI"
```

Alternatively, you can also run by double clicking **runAll.bat** in the project directory

Results

Reports Reports are generated in the following folders in the project directory in the following paths

/target/cucumber-html-reports/overview-features.html View

/target/cucumber/report.html View

/test output/PdfReport/ExtentPdf.pdf View

overview-features.html contains detailed reports showing different tabs wise reports for features, tags and tests

report.html contains reports with the screenshot attached to it

ExtentPdf.pdf contains a detailed pdf report is presented with total execution time along with speedometer, pie and bar chart representation of Features, Scenarios and steps representation. A detailed chapter is presented showing tags wise summary. It also contains test step details, hooks associated, total execution time, and error logs(if any).

Logs View Logs are generated in the below directory

/log/application.html

/log/application.log

Screenshots View Screenshots of pass and failure scenarios are attached in the report.html

Known Issues on ecomm UI and petStore API

Issue on Ecommerce site and workaround While running UI automation test execution, the E-Commerce website during random times throws a "Resource Limit Reached" error due to which test execution fails. On random testing, it is found the error is more likely to be encountered during 11 pm to 9 am IST time. It is advisable to run before or after the mentioned time. Also, as a workaround to avoid this error, during various scenarios testing on UI, driver life is created only once and ends when the last scenario is run

Issue on PetStore API and workaround While running POST request, twice, duplicate records are created in the API. When DELETE HTTP request is sent, it does not delete though the response code generated is 200 instead of 400. GET request gives pet not found message. As a workaround, users need to send post requests for each pet id only once. In case the user mistakenly sends a POST request twice, then change the pet id.

After running a POST request, it is observed that the record take few seconds to reflect using GET. As a workaround, 5 seconds wait is implemented while using GET request

Terms And Conditions

Code of Conduct This project owner is Nikhil Agrawal. By participating in this project you agree to abide by my terms.

Agreement All contributors are required to get pre-authorization in writing by sending an email to `nik_agrawal2000@yahoo.com`. Procedure for the same will be send as an email response thereafter.

Reporting Issues

Issues should be reported in the GitHub repo's issue tab.

Statement of Support

Thank you for taking the time in reading this document. I appreciate it

License