Twitter API Automation
By Behave & Requests Python

Sadesh S

Project Details

Language : **Python**

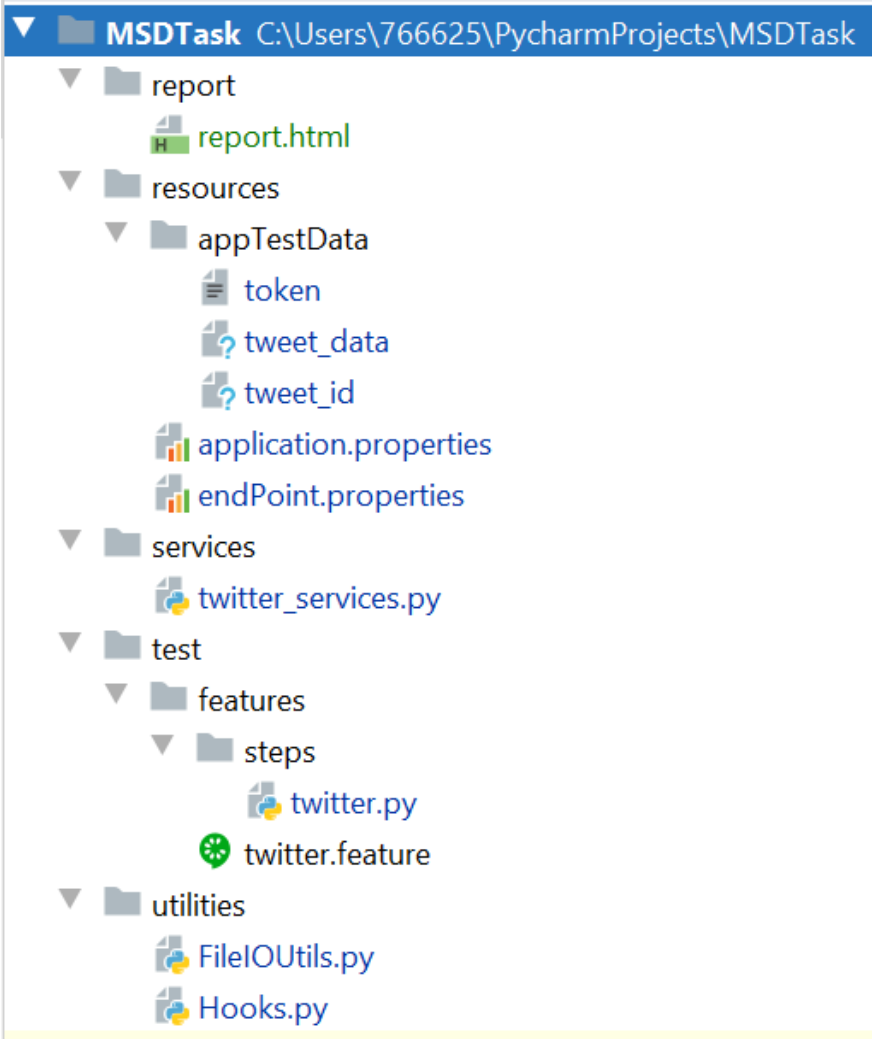
Framework : **BDD with Behave & pyTest**

BDD Plugin : **Gherkin**

Report : Implemented **Simple HTML** report

Assertion : **Generalized Custom Assertions**

Project Structure



Packages

1. **resources**
2. **services**
3. **test**
4. **report**
5. **utilities**

Overview of Packages

resources: Contains test data files, application related URLs and endpoints.

services: Contains all the service implementations of the project.

report: Will contain the HTML report which is generated after each run

tests: Will contain behave feature files and their step definitions

utilities: Will contain utilities related to text and property file handling and add on facilities to our project.

Design decisions

- The project was designed with the **code reusability** in mind. We will see in detail about what every files from all the packages does in detail.
- **fileIOUtils.py** contains all the utils related to text and property files for read and write operations
- **Hooks.py** can place the utils where token can be generated before run and the token deletion is done after each run
- **twitter.feature** contains the BDD scenarios of the twitter API
- The **steps** contains the step definitions of the above feature file

- The **services** package will have the actual implementations of the step definitions implemented from the scenarios
- All the **endpoints** of the twitter will be placed in **endpoint.properties**. **Reason:** If an endpoint is changed in a particular service we will have the convenience of accessing it in the property file directly.
- Same is applicable for **application.properties** file which maintains the URL and keys to access the APIs