**Appium eBay App Automation Framework :**

Consideration :

Since Appium comes with various flavours based on OS and can be installed as in commandline or can have as Desktop app in order to enhance additional features, Considering Python 3.\* and Appium are placed in tools path or installed.

Framework :

The framework is designed as 2-tier architecture serving generic purpose. For instance the app or it's core functionality can be replaced with its equivalent one (similar e-commerce app), which are handled cautiously with minimal impact in the system.

Main.py :

The main.py placed in root folder instantiates the reuired functionalities based on getting two arguments one is the port number which is mandatory to establish the Appium server connection (by default 4723 is considered).

TestSuite (test\_demo.py) :

This other argument is the test suite which is designed to serve the full purpose of eBay App specific functionality.

Hand-shaking with framework, it allows to set up Appium and launch its Server in Host and then allows to perform user defined functionalities by launch Android app, which are namely,

Setting up and checking server connection is established, (Used Unittest for auto tear down here)

Fetching and Logging into the app which was shared,

Authenticating the user and serve the Appium specific purposes like,

click / select operation

swipe (handled scroll here considering all 4 directions),

orientation,

searching a product,

selecting the product,

adding it to cart

then selected product checkout , payment

Config :

desired\_caps.json deals with handling necessary params for Appium connection like,

platformName

platformVersion

deviceName

appPackage

appActivity

logging.yaml :

Serves it's own functionality of logging module with setting console and debug info for log capture detailing

Core :

Files here in core handles the setting up paths & environments and variables too.

lib:

csv\_parser.py & json parser.py :

Assists in parsing login credential to user and parsing desired cap data respectively

adb\_util.py & ebay.py :

support the core functionality of app usage right from login

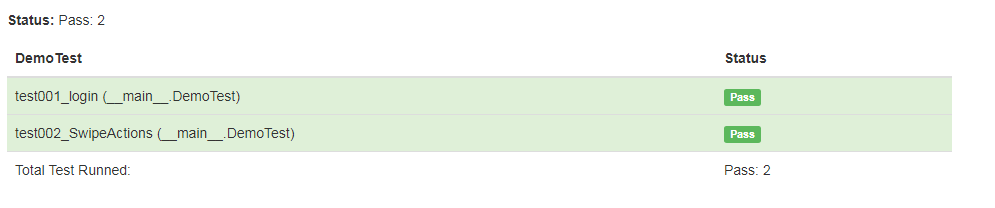
globals:

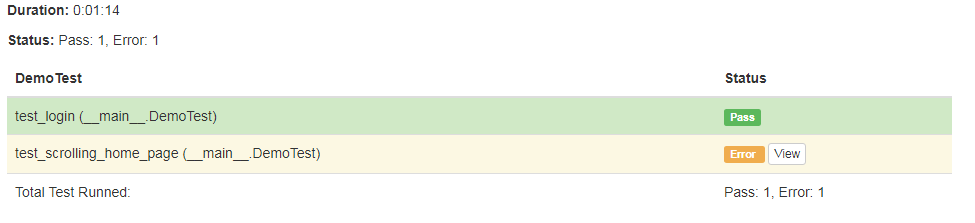
Instead of hard-coding any input keys or values that are required to pass to the fuctions, those are handled here as variables

Logs :

contains commandline suite\_excution.log and debug.log for better understanding and also included html files for graphical representation.

Collected both pass and negative scenario logs.





Test data :

contains the input credential details in form of login.csv

Executable :

python main.py -t 4723 -s test\_demo.py