**Python Playwright Automation Framework Usage**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author/Editor |
| 0.1 | 08/09/2024 | First Draft | Priya Tayade |
|  |  |  |  |

Table of Contents

1. **What is Python Playwright Automation Framework**

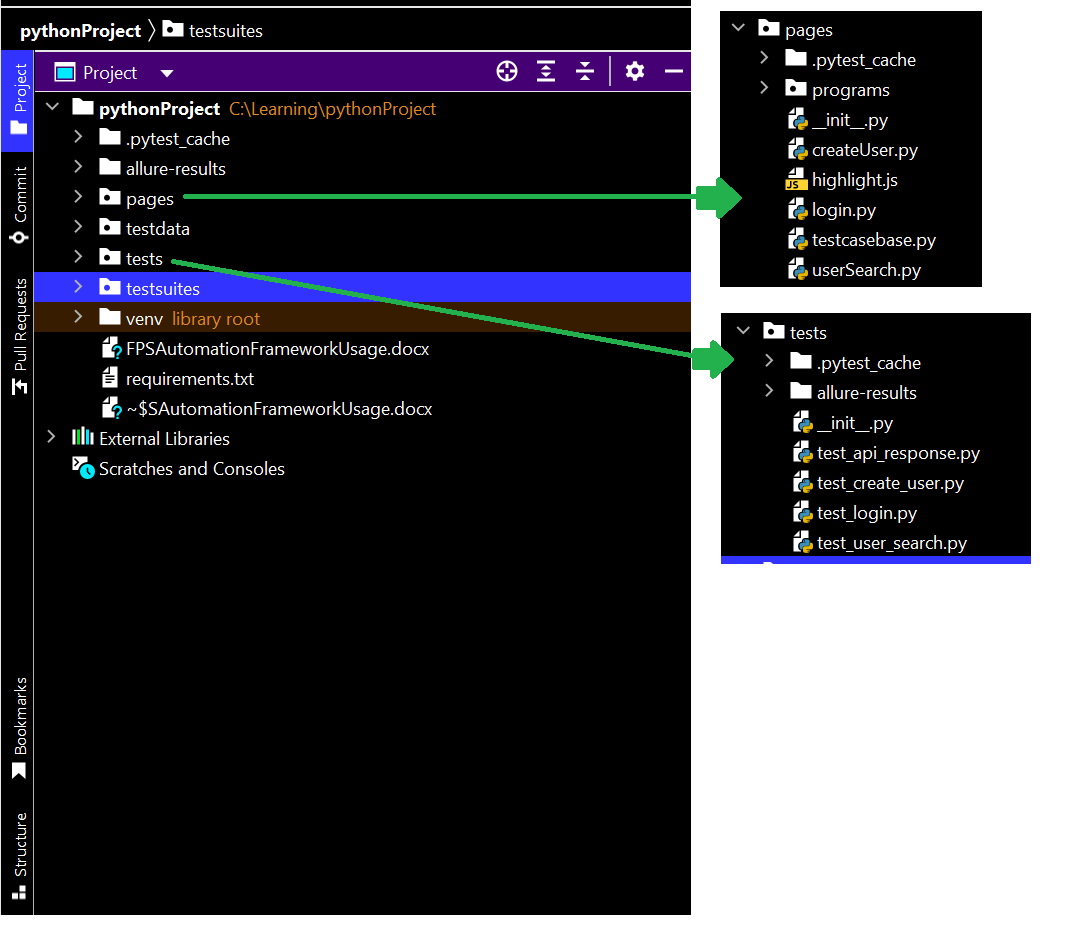
[**2.** **The Playwright Automation Framework folder structure**](#_Toc108429699) **and working**

[**3.** **Project Repository**](#_Toc108429700)

[**4.** **Diagrammatical representation of the Automation Flow**](#_Toc108429701)

1. **What is Python Playwright Automation Framework?**
2. This Python Playwright automation framework basically deals with Functional UI testing and API testing.
3. This framework is specified in Python language.
4. It is a python project which makes use of Pytest and Playwright framework.
5. It includes Playwright scripts to automate site activities.
6. We can use Pycharm IDE to create, debug and execute test scripts in easy way.

# **The Python Playwright Automation Framework folder structure**



1. **Requirements.txt:**

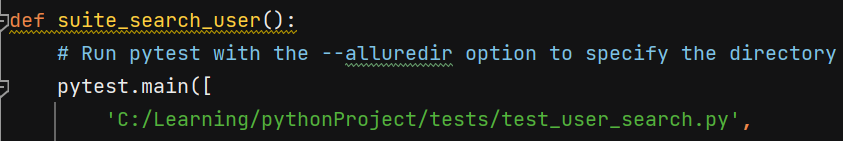
The requirements.txt file contains information about all the required libraries/packages to build the project. User need to run below command to install packages from this file. We don’t need to add externally.

**pip install -r /path/to/requirements.txt**

1. **testsuites**:

This package handles python test runner files. It contains different python suites for different test scripts. Without going into technical implementation one can run test classes without running any python/pytest commands. Please make sure to add correct test file path. We can add more methods/files as per project requirements. As of now it has, 4 different tests for 4 scenarios:

* Login Functionality
* User Search Functionality
* Create User Functionality
* Test API Functionality



*Future Scope: We can create multiple test suites as per different applications available in single project, to run multiple test files.*

1. **testdata**:

This package consists of python file to store data. [Ex: Locators, URLs, Credentials, Listners]

*Future scope: We can add multiple test files as per project requirements*.

1. **allureResults**:

This is autogenerated folder to store allure test reports with pass/fail comments.

*Future Scope: We can customize allure reports as per your need.*

1. **tests**:

This folder contains the actual test files as per test scenarios. We can add more methods/files as per project requirements. As of now it has, 4 different tests for 4 scenarios:

* Login Functionality
* User Search Functionality
* Create User Functionality
* Test API Functionality

1. **pages:**

This folder is for creating test definitions, which we are further calling in tests scripts. As of now it has 4 page object classes for above mentioned 4 test scripts.

1. **TestcaseBase**:

This file available in pages package. It works like utils class. It has all core methods available written in python, playwright. This methods are called by page object classes. It contains methods to automate web, giving input, naviagating to url, click any button, waits, assertions etc.

# **Project Repository:**

**https://github.com/priya145/PlaywrightPython**

# **Diagrammatical representation of the Automation Flow**

Flow

using

Where:

**Generates AllureReport**

**Test Suite**

**Test Class**

**Properties file in TestData folder**

**Pages Class**

**TestCaseBase Class**