PYTEST-BDD

Muhammad Sibtain Haider

Agenda

What is Pytest-BDD ?

Tools and Technology Used

Pytest-BDD Framework

Advantages

Introduction

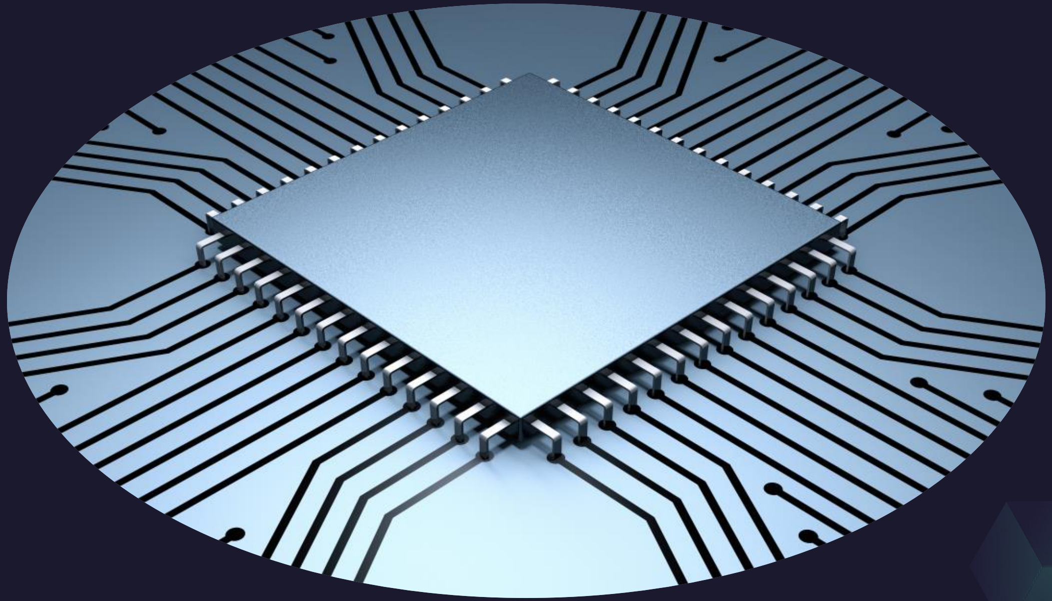
Purpose of this presentation is to guide about the Pytest-BDD framework, what all tools and technologies are used in its installation with an example web UI test suite and discuss its advantages.

What Is Pytest-BDD ?

- Pytest-BDD is a Python BDD framework used for agile software development technique that promotes collaboration between non-technical or business stakeholders with the developers and testers.

- Pytest-BDD implements a subset of the Gherkin language to enable automating project requirements testing and to facilitate behavioral driven development.

Tools And Technologies Used

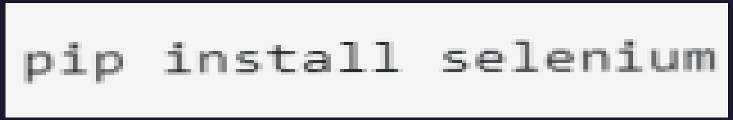


Prerequisites

1. Python

2. Selenium

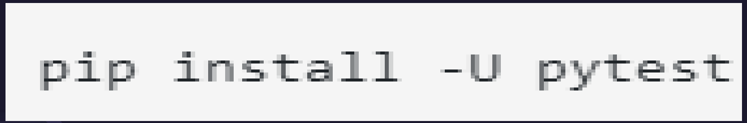
   For Selenium installation, you can use the pip command



```
pip install selenium
```

3. Pytest

   Using the pip command, install Pytest



```
pip install -U pytest
```

4. Intellij IDEA

Install Intellij IDEA from Software Center and login in with a valid ID

5. JAVA

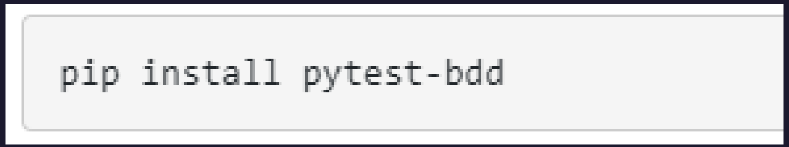
Install JAVA set JAVA Jdk path in User Variables and the JAVA bin path in the System Variable -> Path both in Environment Variable

6. Allure

Install Allure and set Allure bin path in System Variable -> Path in Environment Variables

Installation

1. Pytest-
   BDD
   Pytest-BDD using the pip command



```
pip install pytest-bdd
```

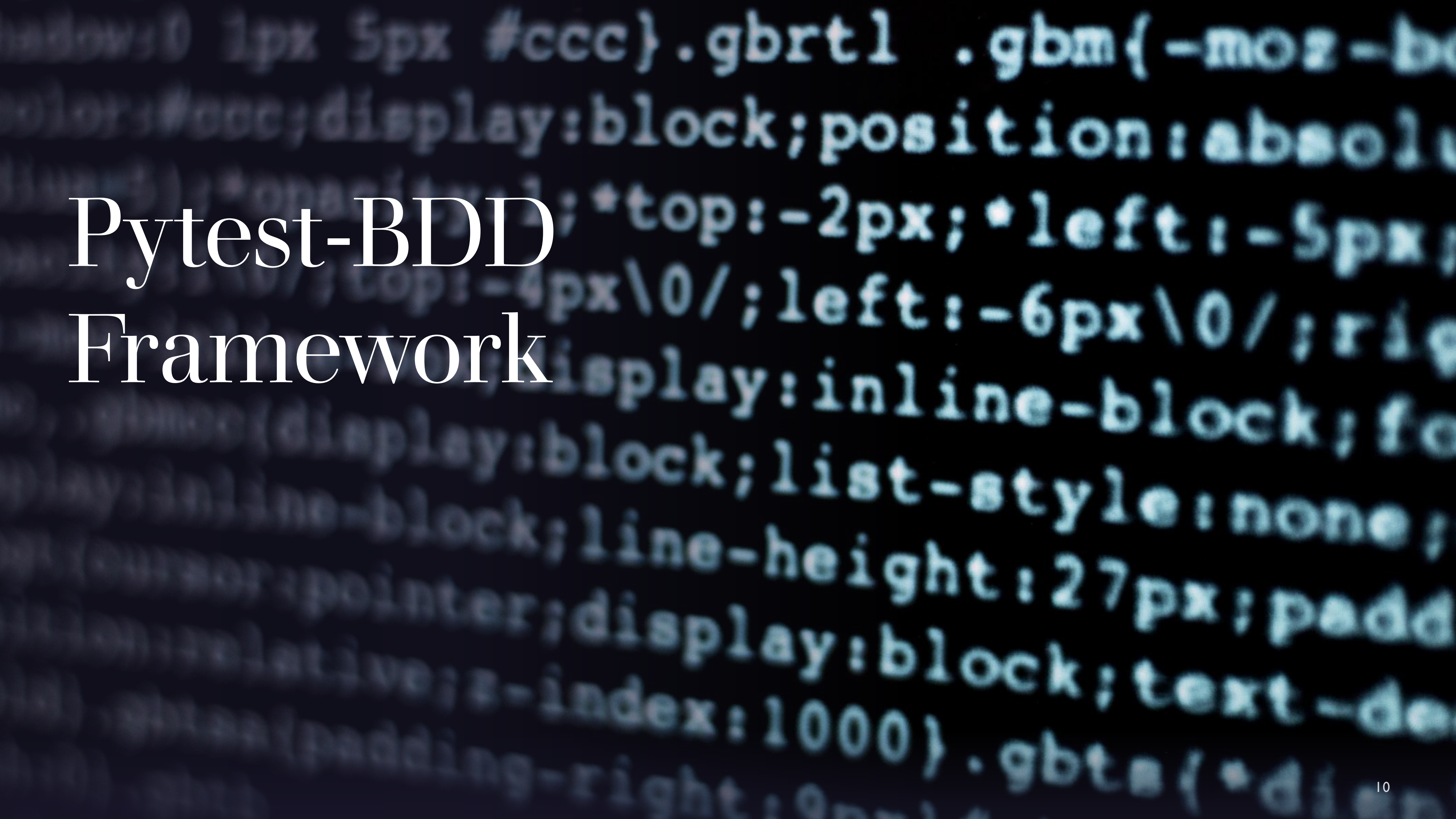
2. PLUGINS

    1. CUCUMBER +

    2. Gherkin

   Located in IntelliJ -> File -> Settings ->Plugins

Install



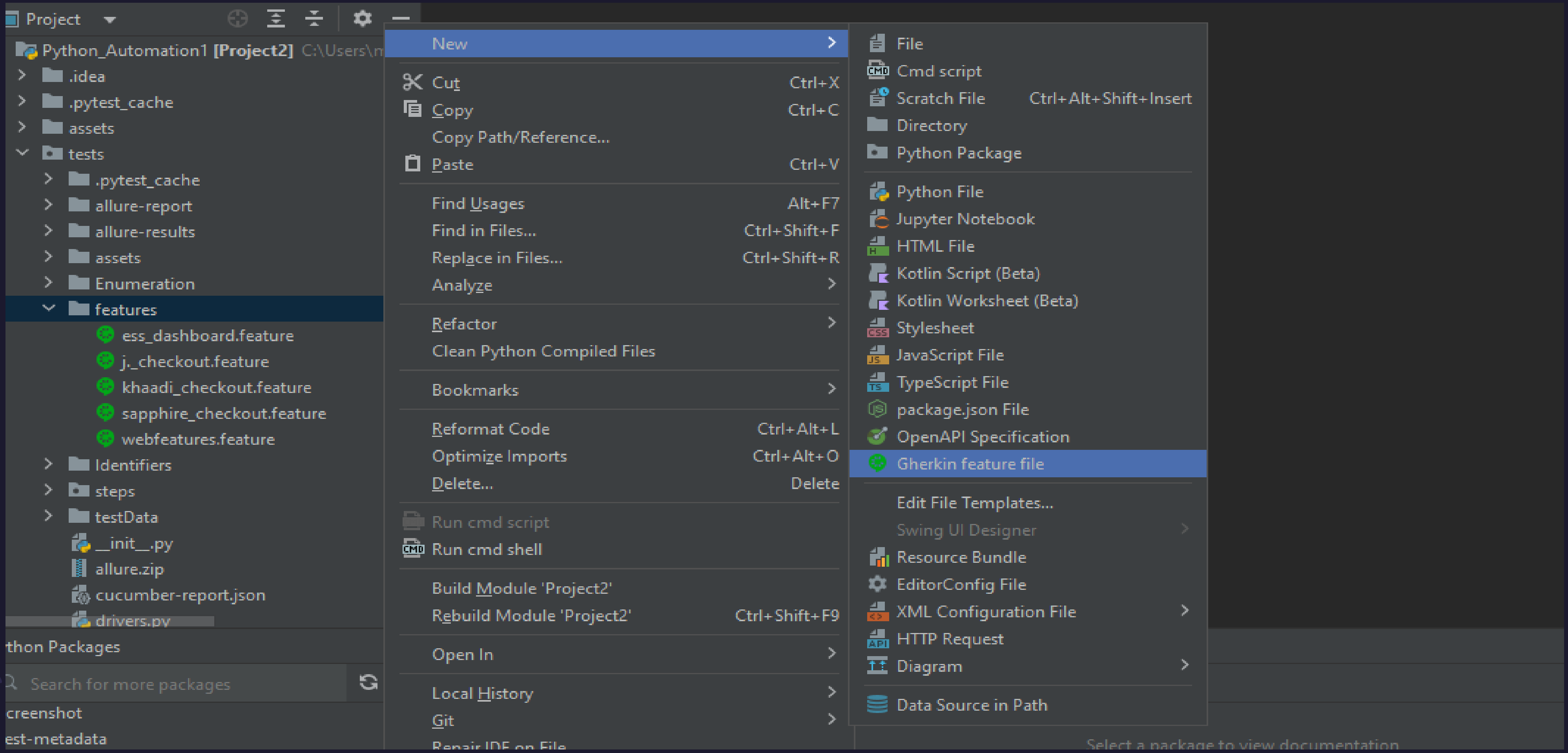
Pytest-BDD Framework

Structure

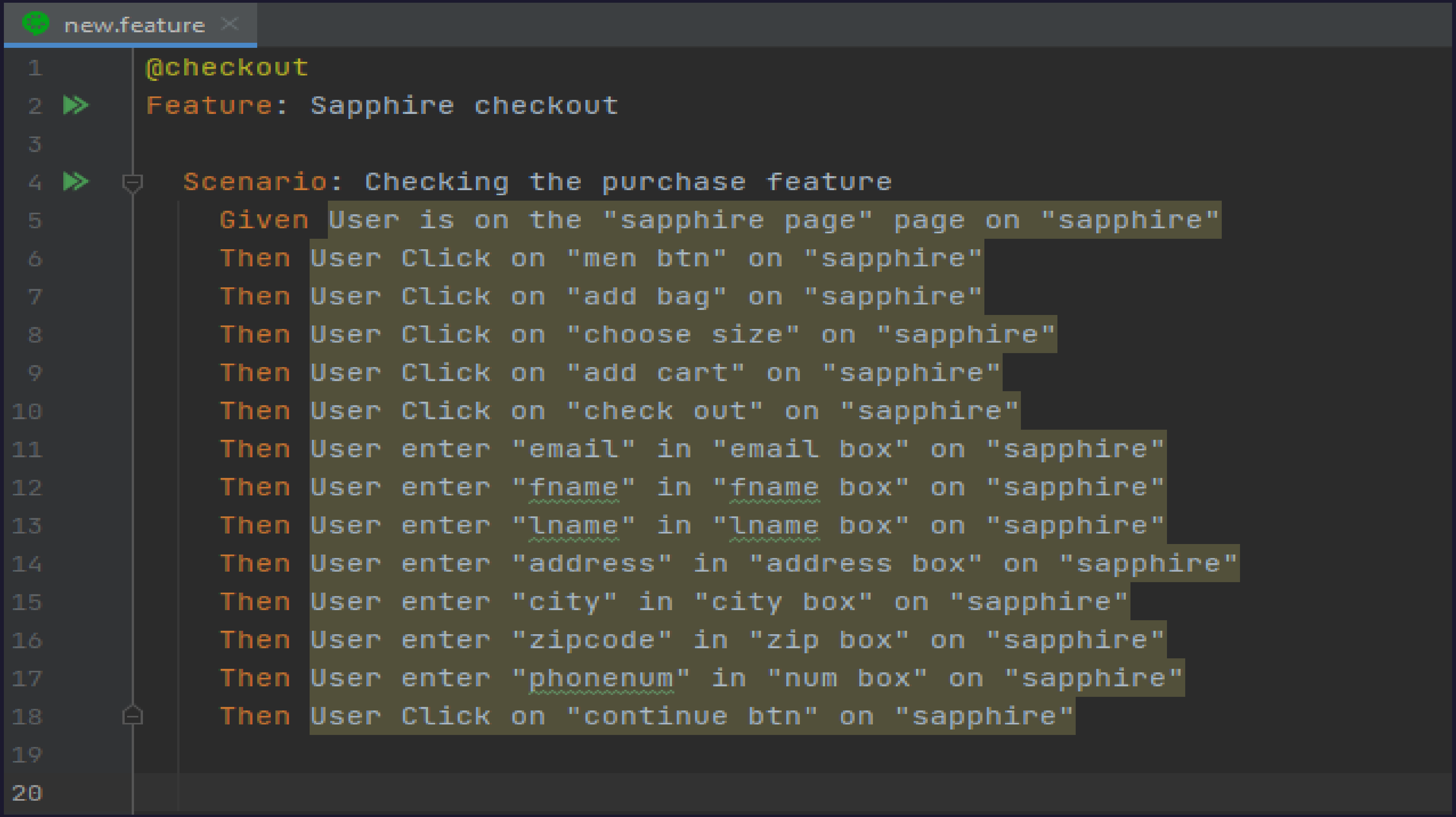
```
Project
        |_ Test
               |_ features
                     *.feature
               |_ steps
                      test_step_defs.py
```

Creating a Feature File

1. Right-click on the desired directory and create a new Gherkin feature file.



2. Write your Gherkin scenario(s) in the Feature file



new.feature

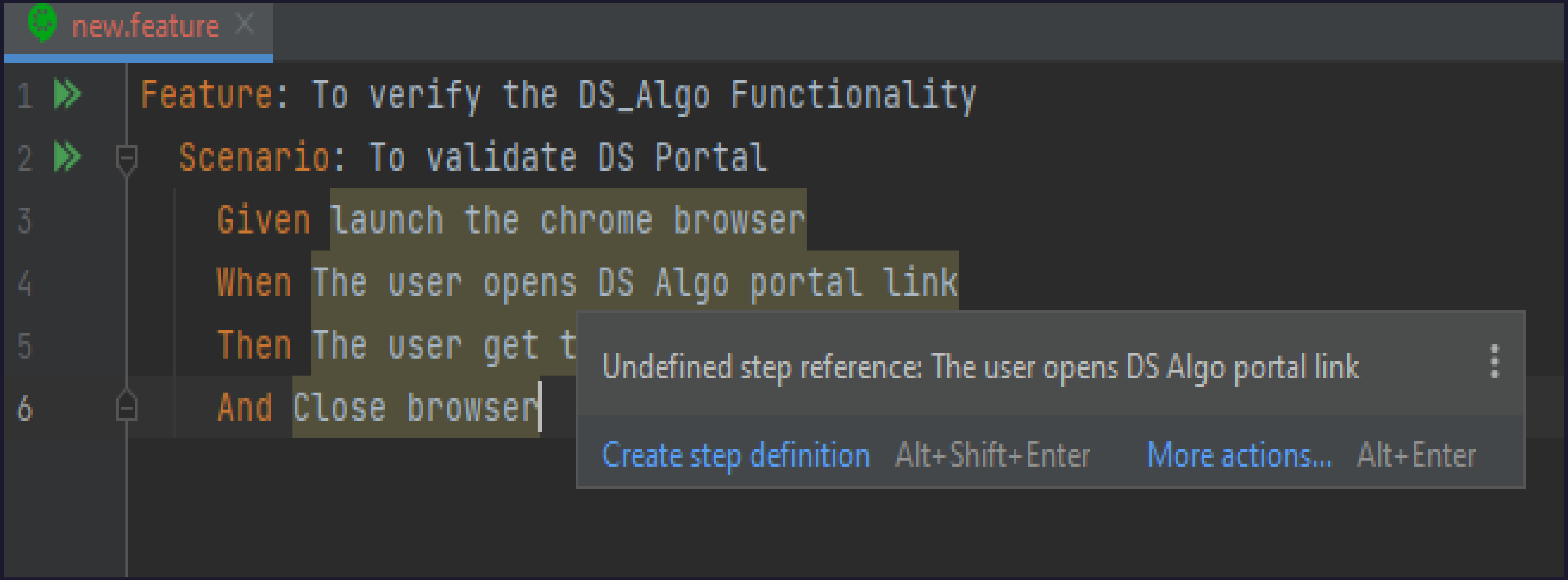
```
@checkout
Feature: Sapphire checkout

  Scenario: Checking the purchase feature
    Given User is on the "sapphire page" page on "sapphire"
    Then User Click on "men btn" on "sapphire"
    Then User Click on "add bag" on "sapphire"
    Then User Click on "choose size" on "sapphire"
    Then User Click on "add cart" on "sapphire"
    Then User Click on "check out" on "sapphire"
    Then User enter "email" in "email box" on "sapphire"
    Then User enter "fname" in "fname box" on "sapphire"
    Then User enter "lname" in "lname box" on "sapphire"
    Then User enter "address" in "address box" on "sapphire"
    Then User enter "city" in "city box" on "sapphire"
    Then User enter "zipcode" in "zip box" on "sapphire"
    Then User enter "phonenum" in "num box" on "sapphire"
    Then User Click on "continue btn" on "sapphire"

```

Creating Step Definition

1. Hover over the step statement Click -> Create Step Definition

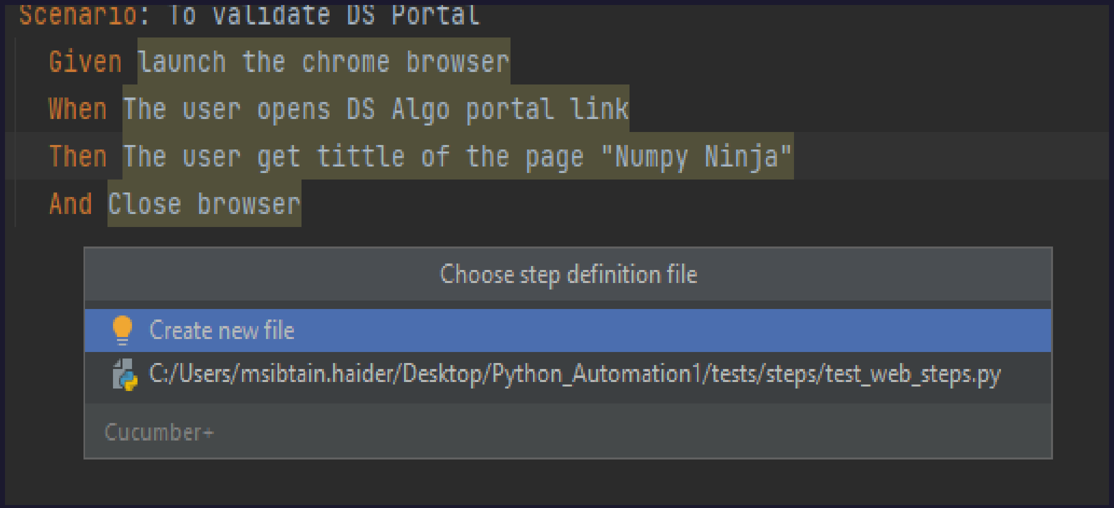


```
Feature: To verify the DS_Algo Functionality
  Scenario: To validate DS Portal
    Given launch the chrome browser
    When The user opens DS Algo portal link
    Then The user get t
    And Close browser
```

Undefined step reference: The user opens DS Algo portal link

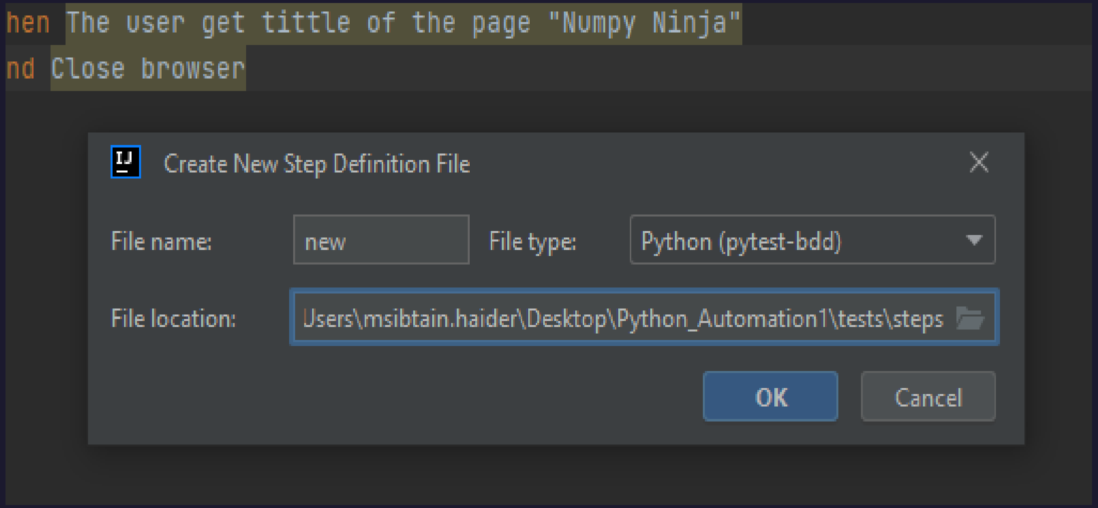
Create step definition Alt+Shift+Enter
More actions... Alt+Enter

2. Click on Create a new file



```
Scenario: To validate DS Portal
  Given launch the chrome browser
  When The user opens DS Algo portal link
  Then The user get tittle of the page "Numpy Ninja"
  And Close browser
```

Choose step definition file
Create new file
C:/Users/msibtain.haider/Desktop/Python_Automation1/tests/steps/test_web_steps.py
Cucumber+



```
hen The user get tittle of the page "Numpy Ninja"
nd Close browser
```

Create New Step Definition File

File name: new

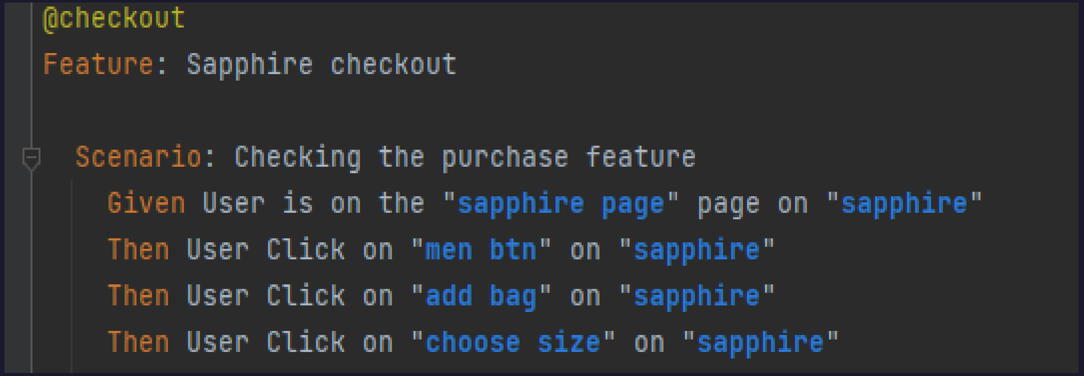
File type: Python (pytest-bdd)

File location: Users\msibtain.haider\Desktop\Python_Automation1\tests\steps

3. Implement logic for each step

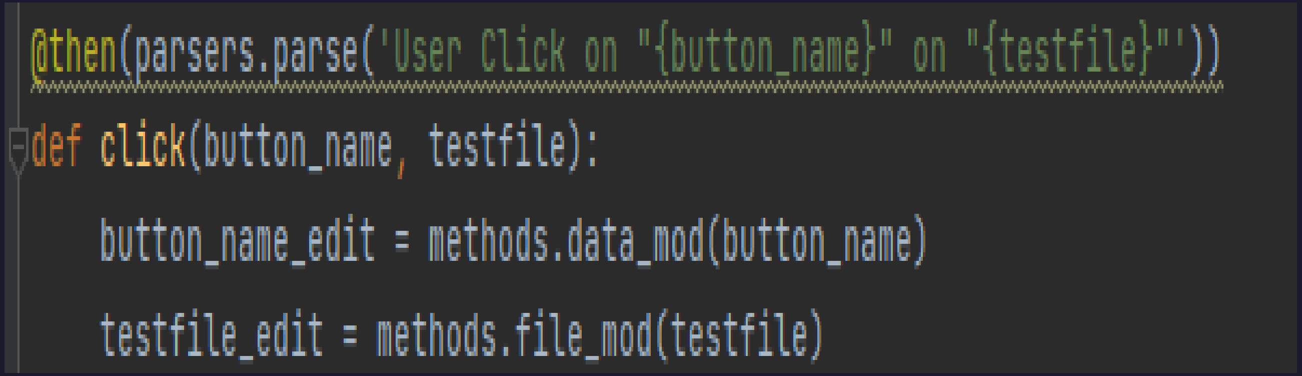
4. Incase you want to make the steps reusable you will have to make changes in the feature file and the step definition

Write the keyword in the step in the feature file within inverted commas ("keyword") and in the step definition file write parsers.parse and the generic keyword in inverted commas and curly brackets ("{data}") and pass it as the function parameters and do import parsers from pytest-bdd



```
@checkout
Feature: Sapphire checkout

  Scenario: Checking the purchase feature
    Given User is on the "sapphire page" page on "sapphire"
    Then User Click on "men btn" on "sapphire"
    Then User Click on "add bag" on "sapphire"
    Then User Click on "choose size" on "sapphire"
```

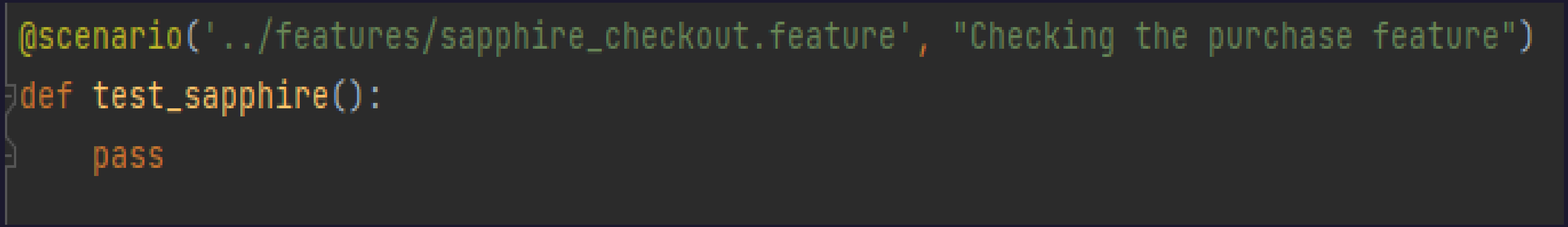


```
@then(parsers.parse('User Click on "{button_name}" on "{testfile}"'))
def click(button_name, testfile):
    button_name_edit = methods.data_mod(button_name)
    testfile_edit = methods.file_mod(testfile)
```

5. Do remember that do not define the steps name with "test_" as it is case sensitive and will produce errors as it will consider it as test too. Write simple names for the step definitions.

Writing Test

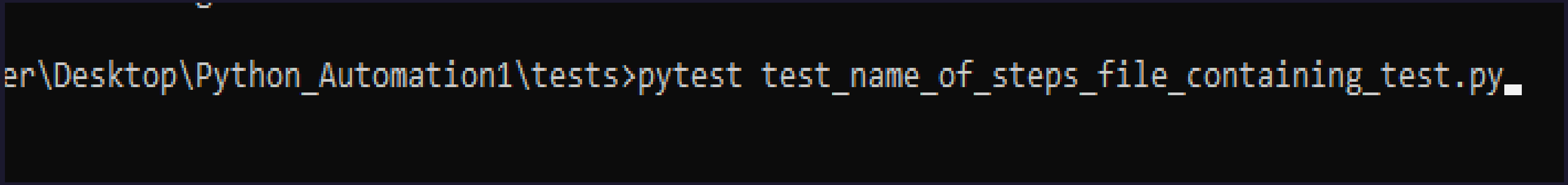
1. Write a test method as suggested by pytest in the step definition file to execute the test. Specify the feature file path we need to refer to and the scenario we want to execute



```
@scenario('../features/sapphire_checkout.feature', "Checking the purchase feature")
def test_sapphire():
    pass
```

Where the first parameter is the location of the feature file and the second one is the name of the scenario.

2. To envoke the test write the command or just run the steps file containing the test function.



```
er\Desktop\Python_Automation1\tests>pytest test_name_of_steps_file_containing_test.py
```

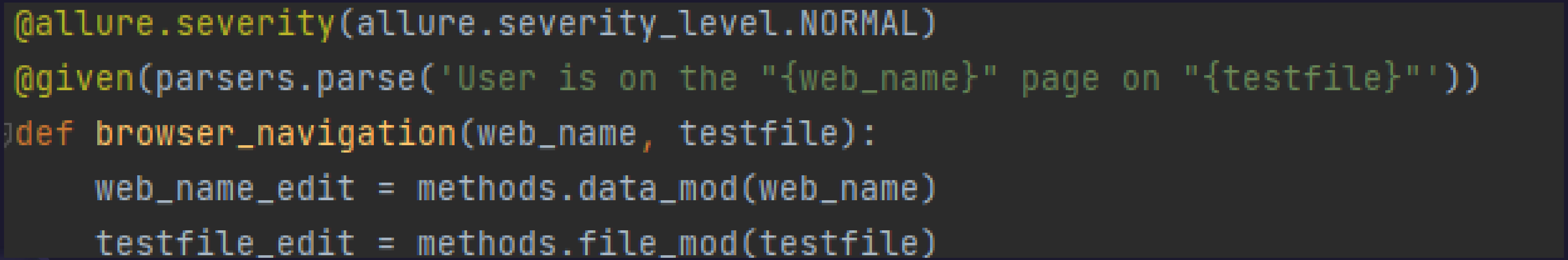
Allure Report

1. To generate Allure report import allure in the step definition and then install python packages names

   1. Allure-pytest-bdd

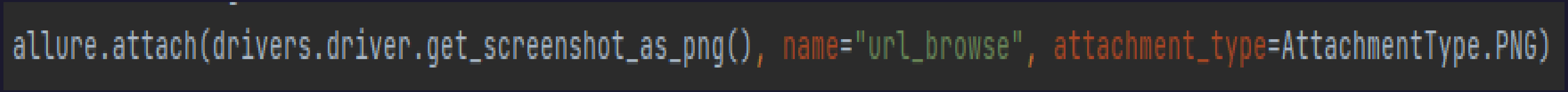
   2. Allure-python-commons

2. Write the command as follows before every step definition and change the severity level according to your demand



```
@allure.severity(allure.severity_level.NORMAL)
@given(parsers.parse('User is on the "{web_name}" page on "{testfile}"'))
def browser_navigation(web_name, testfile):
    web_name_edit = methods.data_mod(web_name)
    testfile_edit = methods.file_mod(testfile)
```

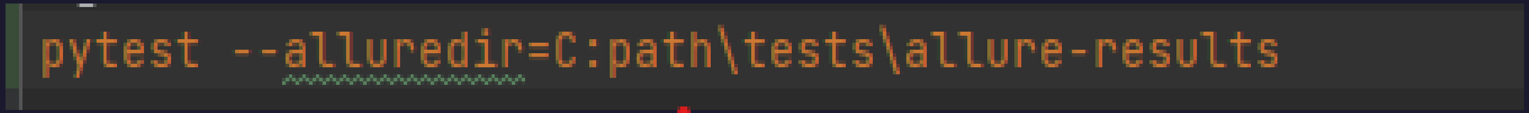
3. To attach screenshots to your allure report write the command inside the step definition as follows



```
allure.attach(drivers.driver.get_screenshot_as_png(), name="url_browse", attachment_type=AttachmentType.PNG)
```

with the name of your choice. Do import AttachmentType from allure-commons.types for this line of code to run.

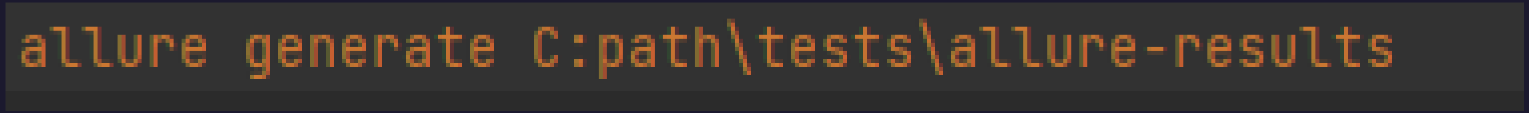
4. To generate allure report run this command



```
pytest --alluredir=C:path\tests\allure-results
```

This command with generate the allure-results in the form of json and png files.

To convert these files into the html and css files run command



```
allure generate C:path\tests\allure-results
```

Open the allure-report folder and open the index.html file on the browser of your choice to view the report.

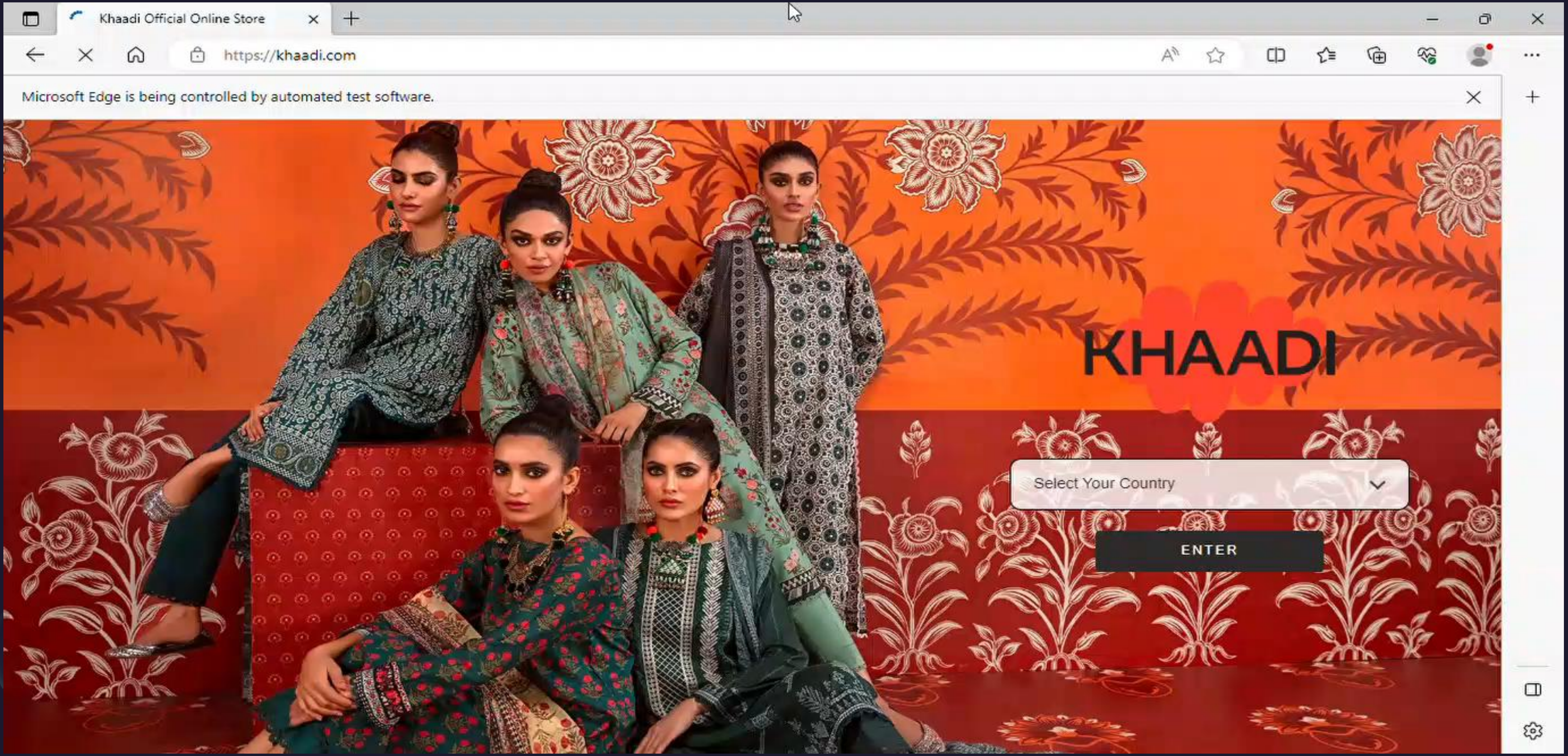
Advantages

IT FULLY SUPPORTS THE GHERKIN LANGUAGE.

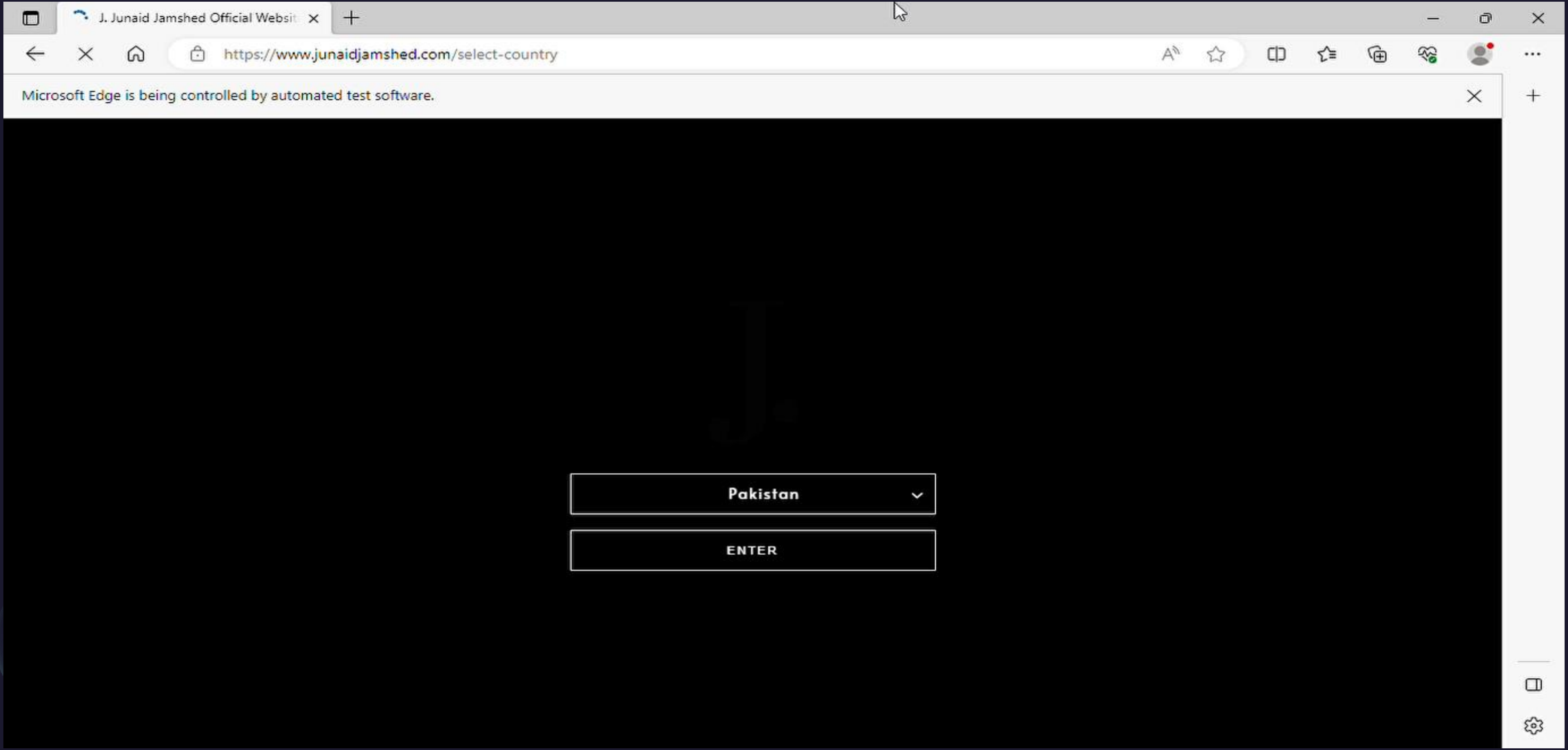
ENVIRONMENTAL FUNCTIONS AND FIXTURES MAKE SETUP AND CLEANUP EASY.

IT HAS DJANGO AND FLASK INTEGRATIONS.

Khaadi Checkout Feature Flow

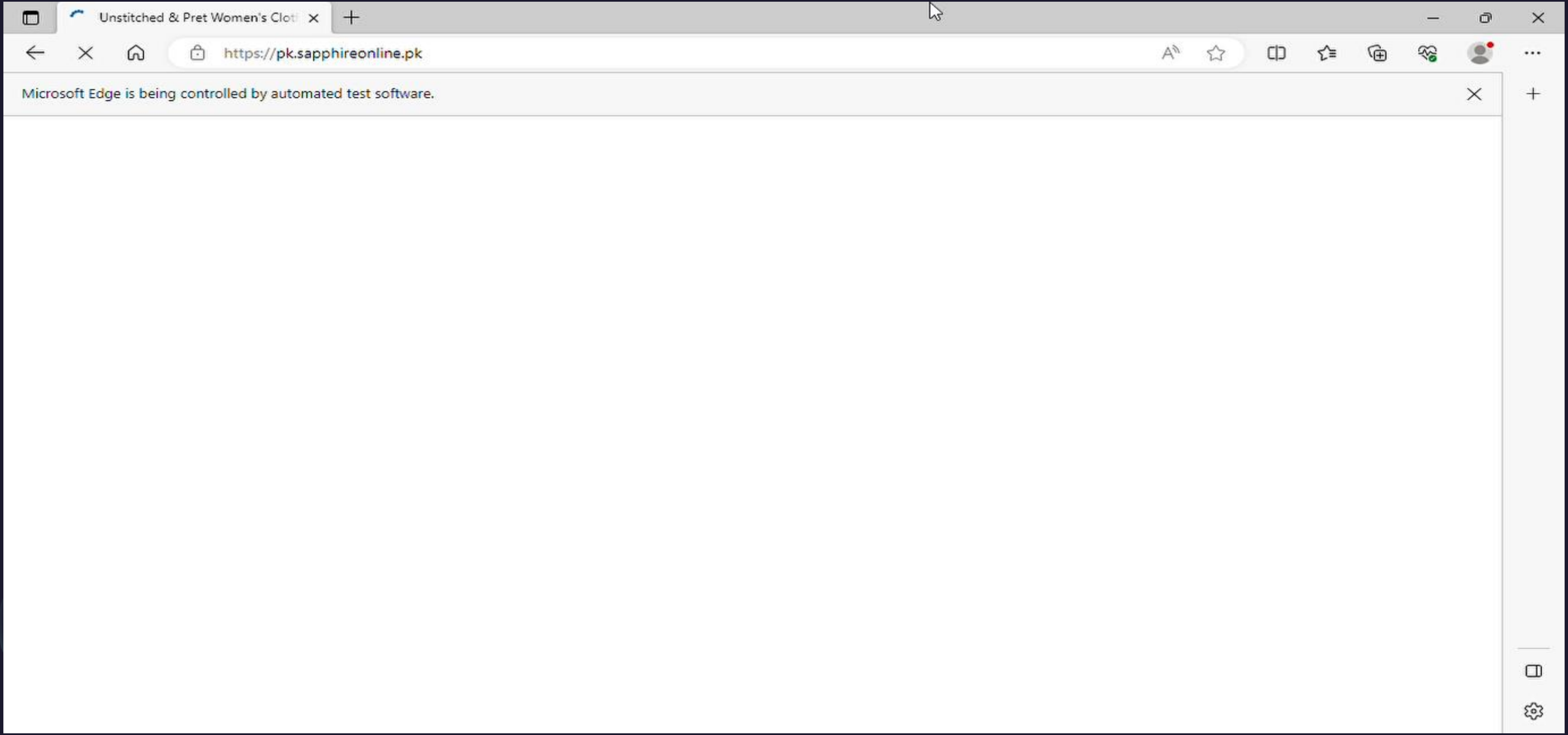

Khaadi Official Online Store
https://khaadi.com
Microsoft Edge is being controlled by automated test software.
KHAADI
Select Your Country
ENTER

J. Checkout Feature Flow


J. Junaid Jamshed Official Websit
https://www.junaidjamshed.com/select-country
Pakistan
ENTER

Microsoft Edge is being controlled by automated test software.

Sapphire Checkout Feature Flow



Microsoft Edge is being controlled by automated test software.

Thank You

Muhammad Sibtain Haider

msibtain.haider@systemsltd.com