Commands used when I created envinronment

pip install pytest-playwright

pip install pytest pytest-bdd

venv\Scripts\activate

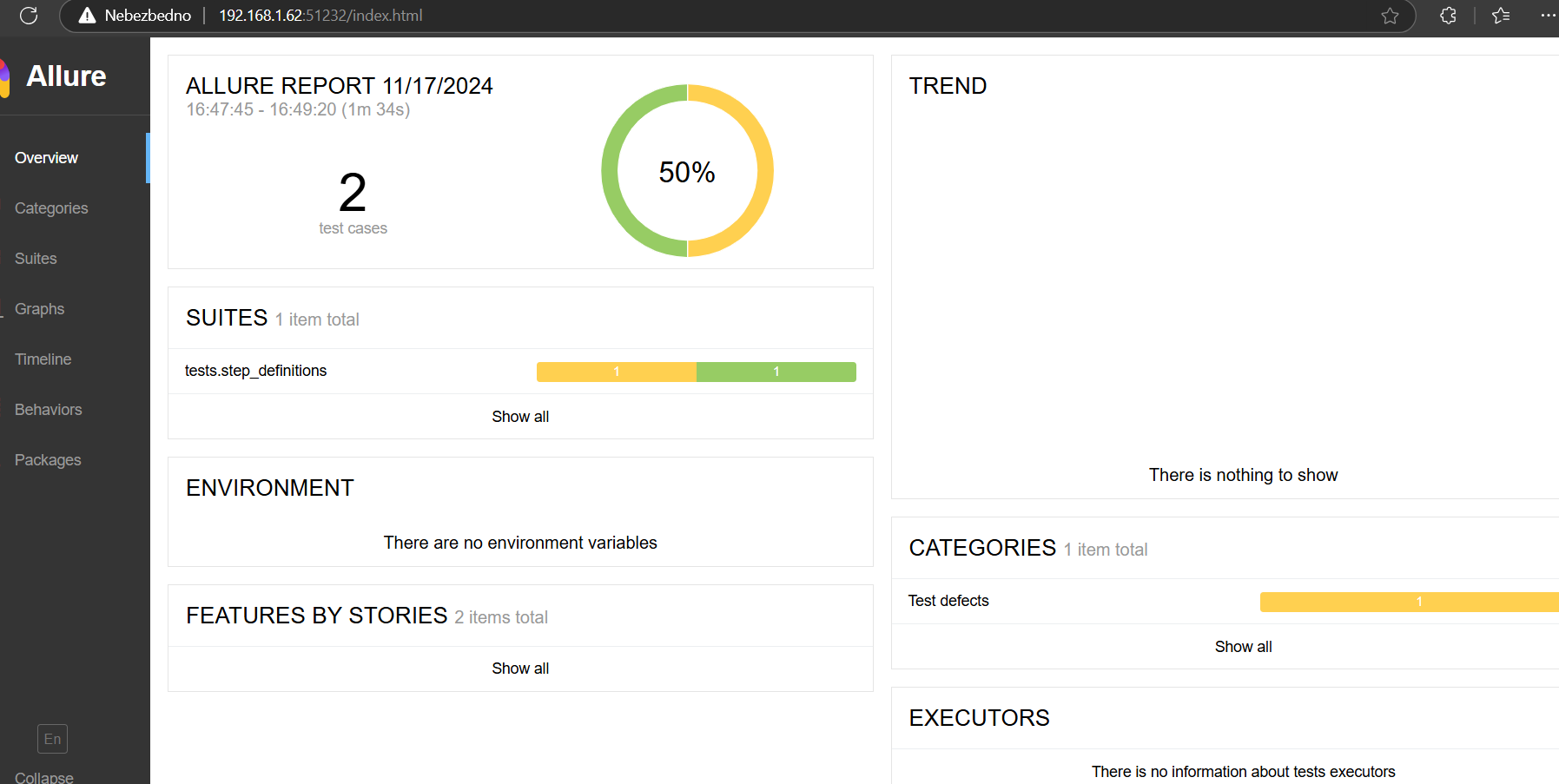
python -m pip install --upgrade pip

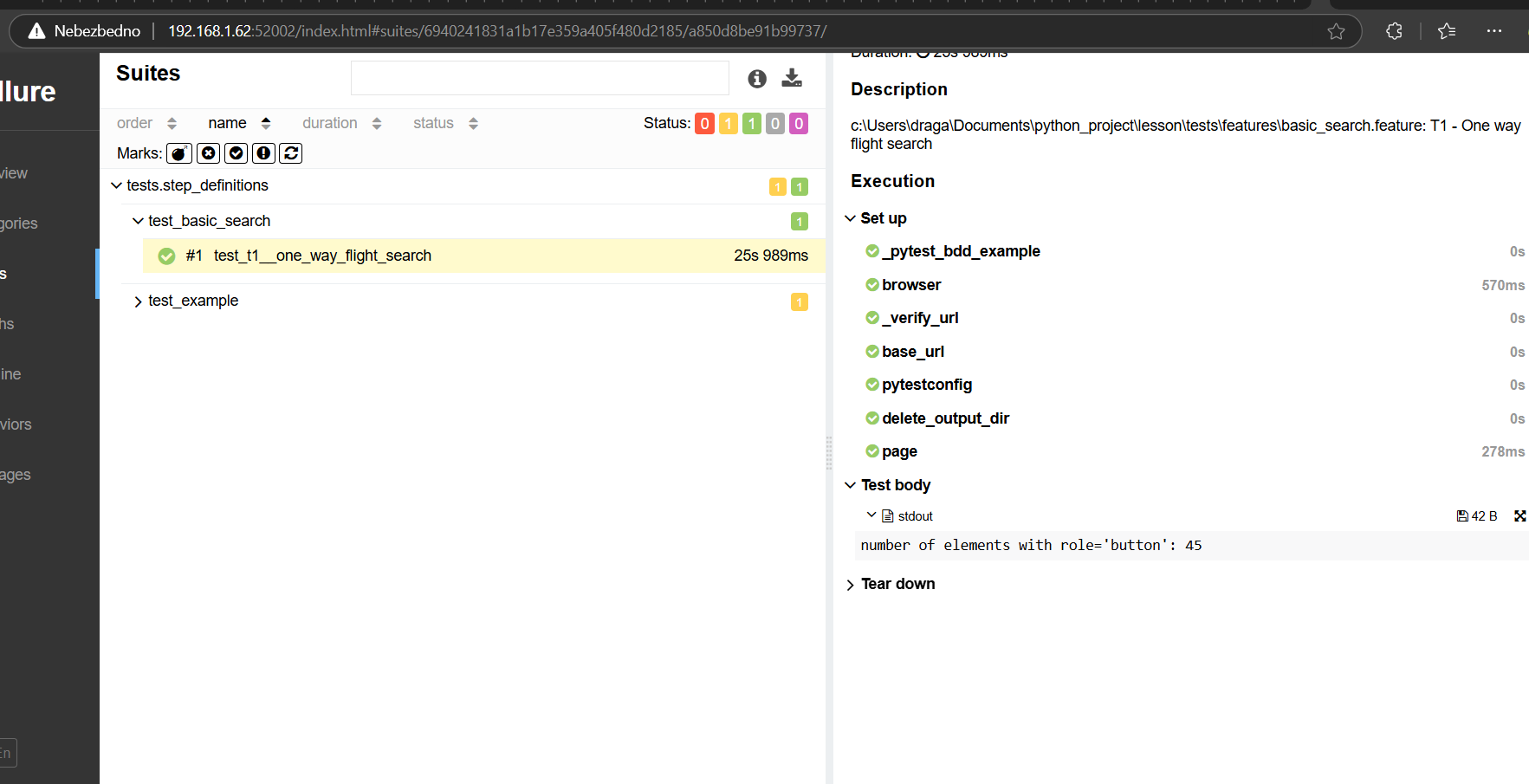
python -m venv venv

python -m ven v ven v

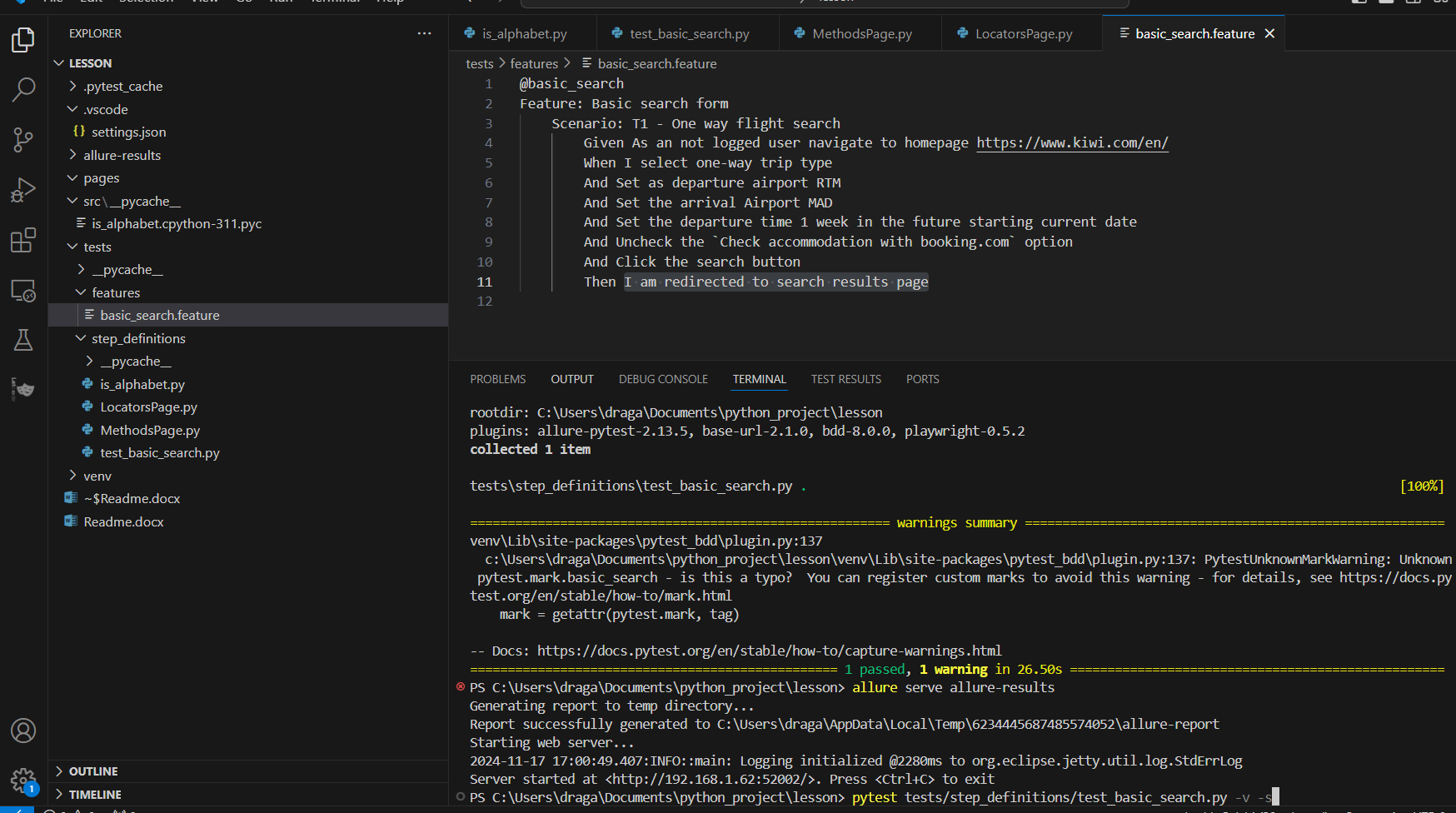
pytest --alluredir=allure-results

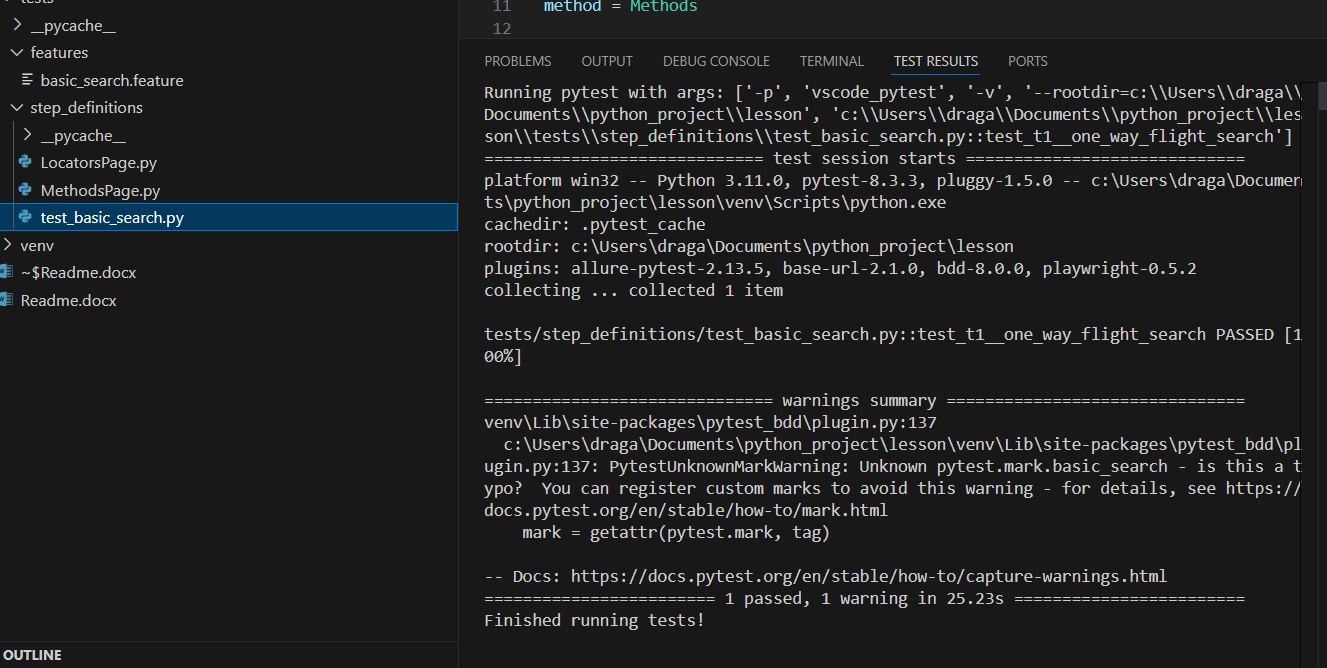
allure serve allure-results





For run test use command in terminal of Studio Code:  
pytest tests/step\_definitions/test\_basic\_search.py -v –s



When right click on test and run result is  
  
If we have more tests in suite, we can run that test from task, with name test\_basic\_search.py, using command  
pytest tests/step\_definitions/test\_basic\_search.py -v –s. Also we can use marking option in pytest and mark our test and easy run using mark.  
I needed to delete my town Belgrade on the beggining and after that import RTM and MAD. If you run test I have in code 2 sec wait reason is that you can see better results when browser will start.  
For departure time set 7 days after today I used

@when("Set the departure time 1 week in the future starting current date")

def departure\_time(page):

    xpath = "//input[@name='search-outboundDate']"

    page.click(xpath)

    time.sleep(2)

    # Calculate the date one week from today

    today ="(//\*[contains(text(), 'Today')])"

    div\_buttons = page.locator('div[role="button"][data-test ="CalendarDay"]').element\_handles()

    div\_buttons\_list = [div for div in div\_buttons]

    print("number of elements with role='button':", len(div\_buttons\_list))

    counter=0

    for div in div\_buttons\_list:

        if (counter ==7):

            div.click()

        counter =counter +1

    time.sleep(1)

I created list of buttons in calendar, before today, buttons are disabled, after that are data-test ="CalendarDay. Way how I went to +7 is with loop   
I clicked on +7 button from today

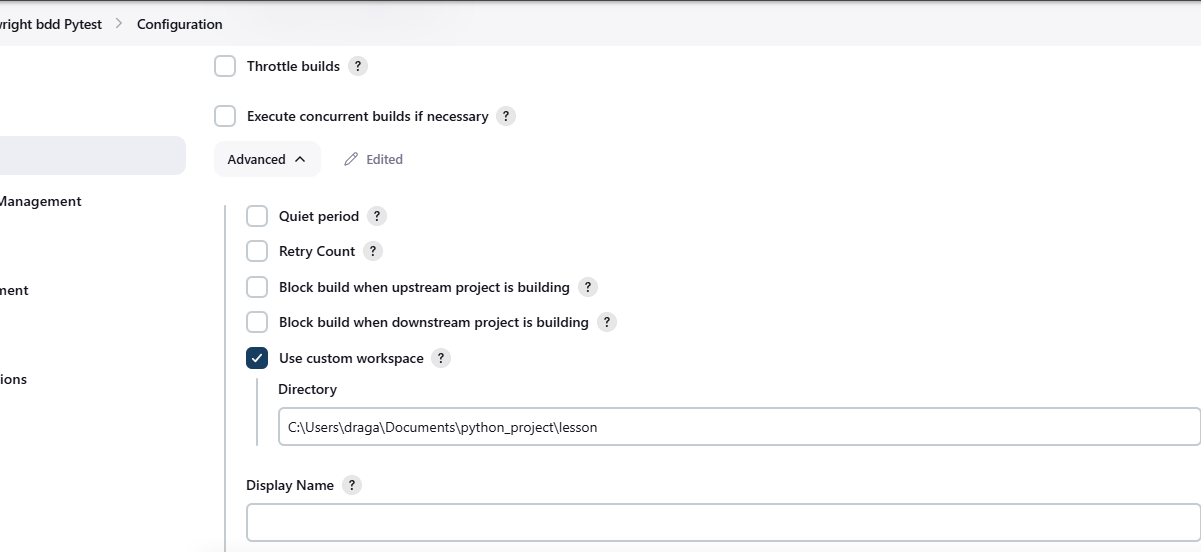
counter=0

    for div in div\_buttons\_list:

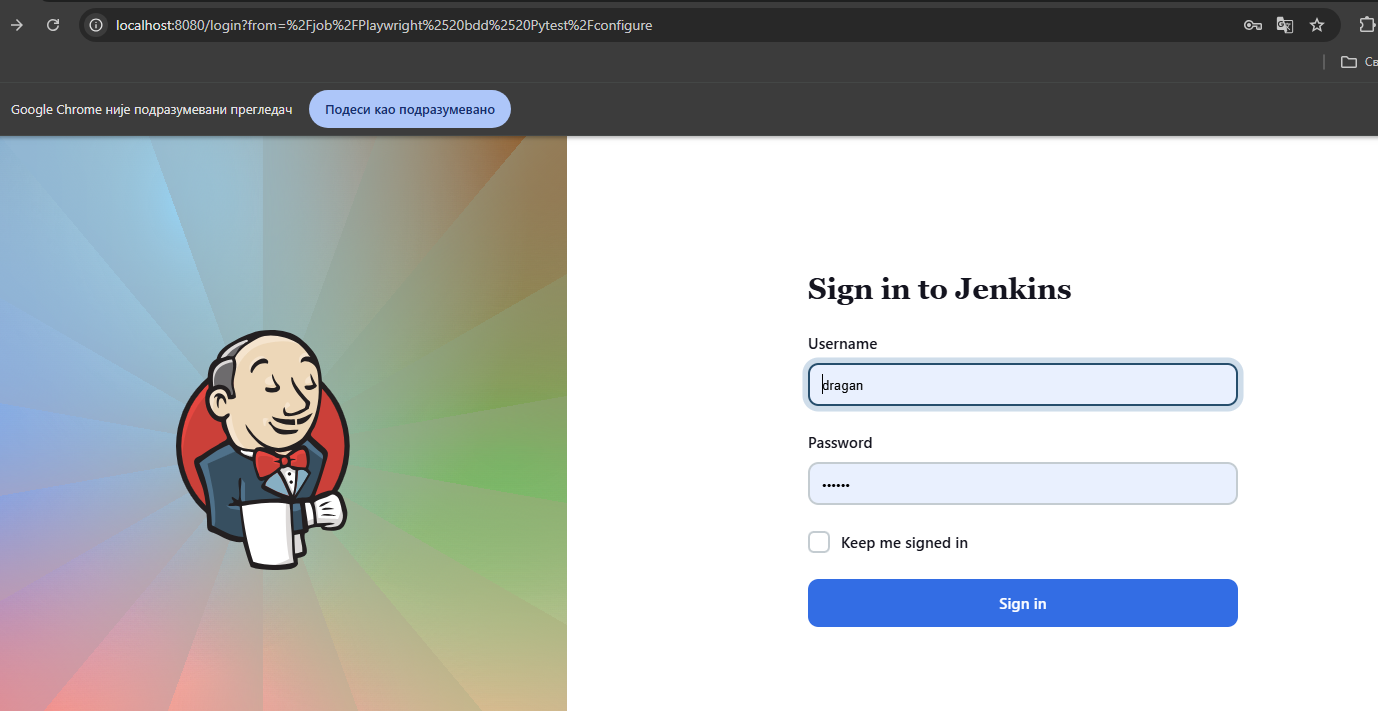
        if (counter ==7):

            div.click()

        counter =counter +1

In Jenkins need to install Playwright and create free style project import path  
  


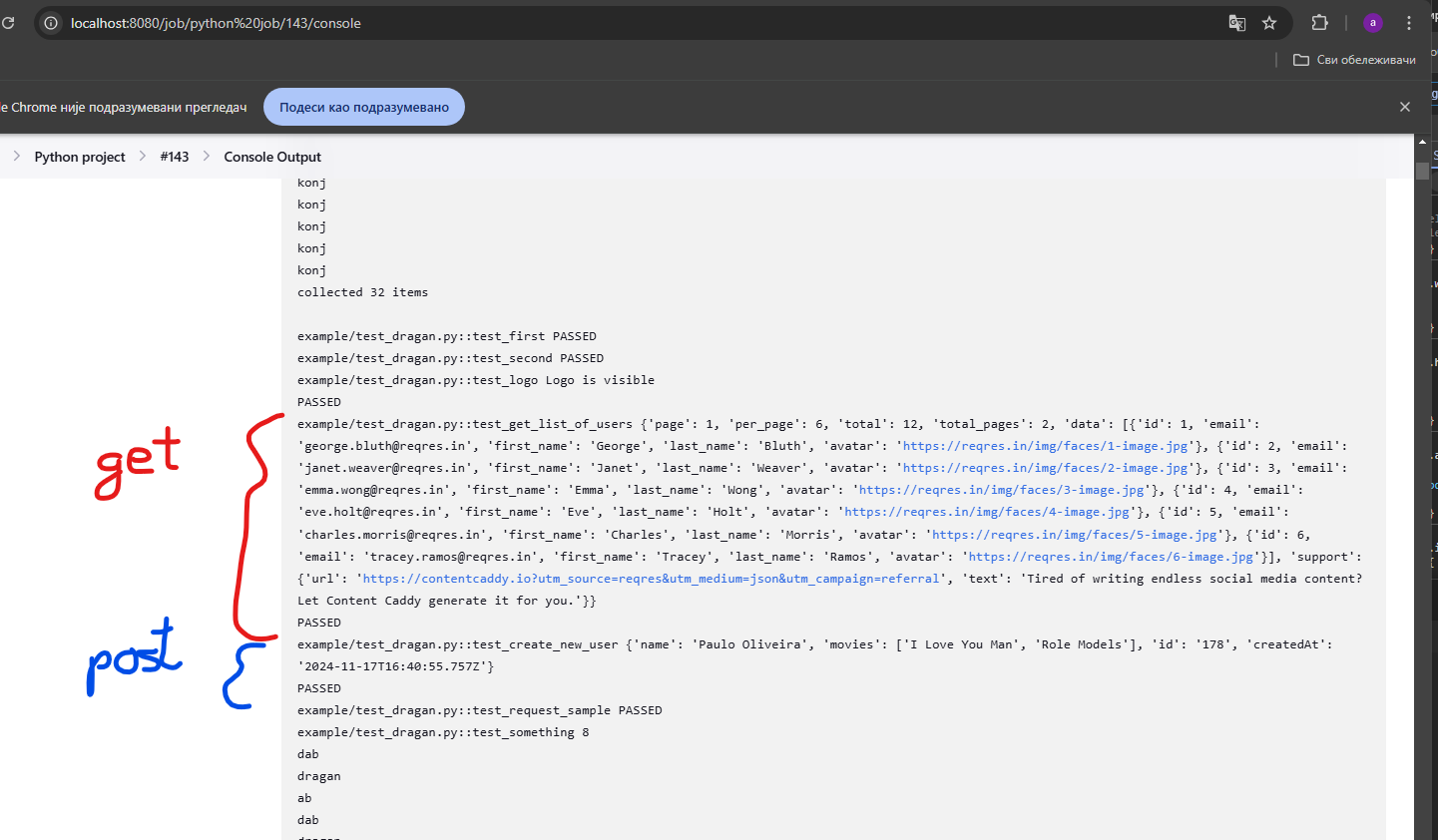
And run command   
pytest tests/step\_definitions/test\_basic\_search.py -v -s  
  

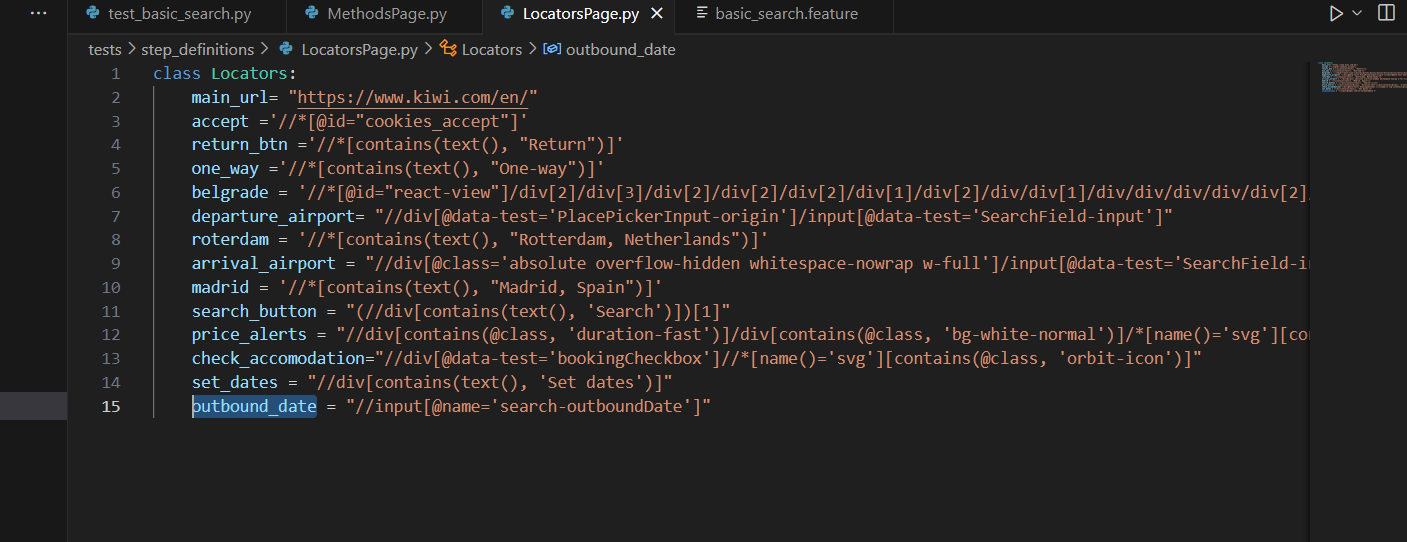
Here is visible pytest results from pytest  
some results of api test with pytest

def test\_get\_list\_of\_users():  
 url = "https://reqres.in/api/users"  
 response = requests.get(url)  
 assert response.status\_code == 200  
 print(response.json())  
  
  
def test\_create\_new\_user():  
 url = "https://reqres.in/api/users"  
 data = {  
 "name": "Paulo Oliveira",  
 "movies": ["I Love You Man", "Role Models"]  
 }  
 response = requests.post(url, data=data)  
 print(response.json())  
 assert response.status\_code == 201

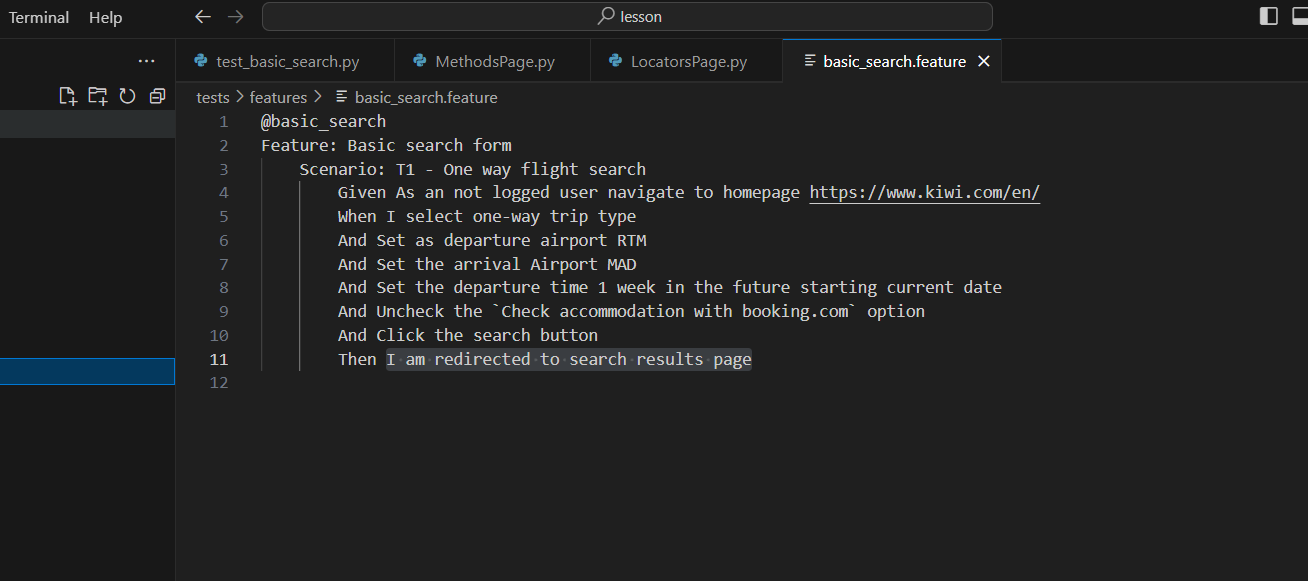
result in Jenkins is

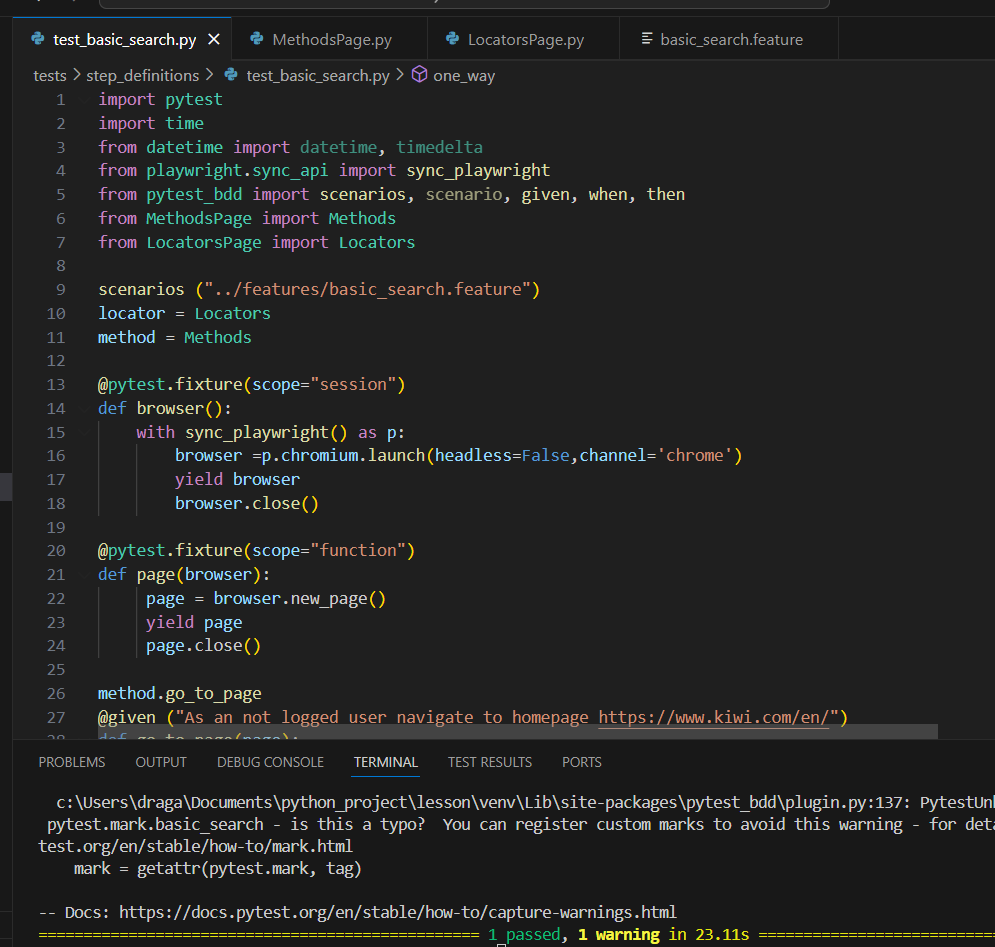


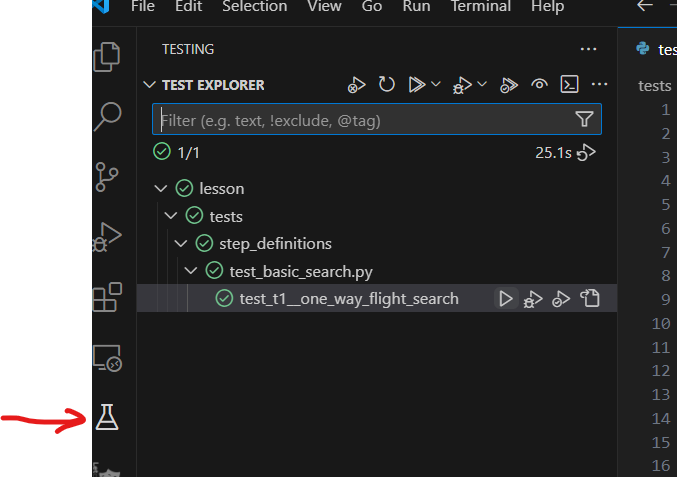
I separated locators in LocatorPage.py

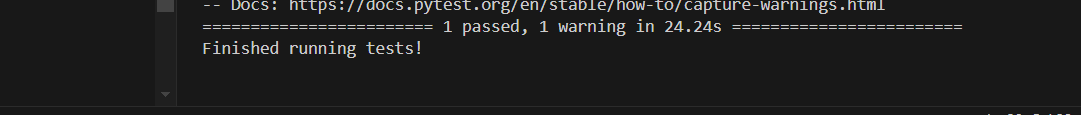


I have feature



Here is test  


Also here is possible to click Testing and run test  




Here is yaml file from pipeline for c# bdd Specflow project on dev Azure. Pipeline is created to start bdd tests on different repository when code is changed. Similar is possible to create pipeline to start pytest tests triggered by pushing changes to repo where is code for web application.

variables:

- name: BuildParameters.solution

  value: '\*\*\\*.sln'

- name: BuildParameters.ArtifactName

  value: drop

- name: Dragan.solution

  value: '\*\*/\*.sln'

- name: buildConfiguration

  value: 'Release'

- name: buildPlatform

  value: 'Any CPU'

resources:

  pipelines:

  - pipeline: DraganDosen.BDDTesting

    source: DraganDosen.BDDTesting

    trigger:

      branches:

        include:

        - main

trigger:

  branches:

    include:

    - refs/heads/master

name: $(date:yyyyMMdd)$(rev:.r)

jobs:

- job: Job\_1

  displayName: Agent job 1

  pool:

    vmImage: windows-latest

  steps:

  - checkout: self

    fetchDepth: 1

  - task: NuGetToolInstaller@0

    displayName: Use NuGet 6.6.1

    inputs:

      versionSpec: 6.6.1

  - task: NuGetCommand@2

    displayName: NuGet restore

    inputs:

      solution: $(BuildParameters.solution)

  - task: VSBuild@1

    displayName: Build solution

    inputs:

      solution: $(BuildParameters.solution)

      msbuildArgs: /p:DeployOnBuild=true /p:WebPublishMethod=Package /p:PackageAsSingleFile=true /p:SkipInvalidConfigurations=true /p:PackageLocation="$(build.artifactstagingdirectory)\\"

      platform: $(BuildPlatform)

      configuration: $(BuildConfiguration)

  - task: VSTest@2

    inputs:

     platform: '$(buildPlatform)'

     configuration: '$(buildConfiguration)'

Also we can create CI/CD with Jenkins and different tools.