QA Tools Training Program:

Module-3 Session-1 Assessment

==============================

1. Consider the following output of "show ap summary" of Controller 'ewlc'. Write parser for this output using RegEx and obtain the output in dictionary format.

ewlc#show ap summary

Number of APs: 3

CC = Country Code

RD = Regulatory Domain

AP Name Slots AP Model Ethernet MAC Radio MAC CC RD IP Address State Location

-------------------------------------------------------------------------------------------------------------------------------------------------------------

APBC26.C7A3.1970 2 AIR-AP3802E-B-K9 bc26.c7a3.1970 00b7.7166.bea0 US -B 192.165.7.199 Registered default location

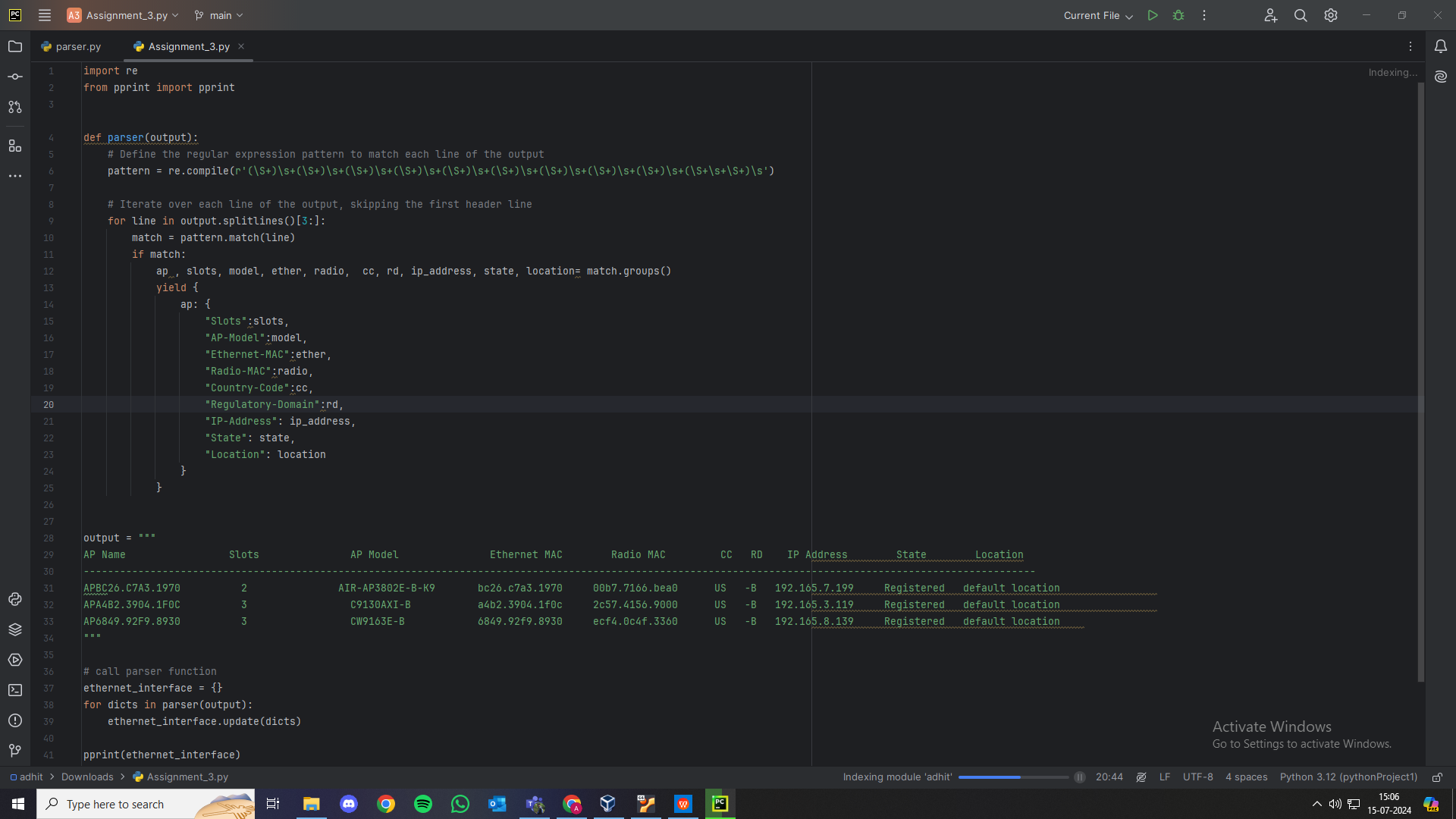
APA4B2.3904.1F0C 3 C9130AXI-B a4b2.3904.1f0c 2c57.4156.9000 US -B 192.165.3.119 Registered default location

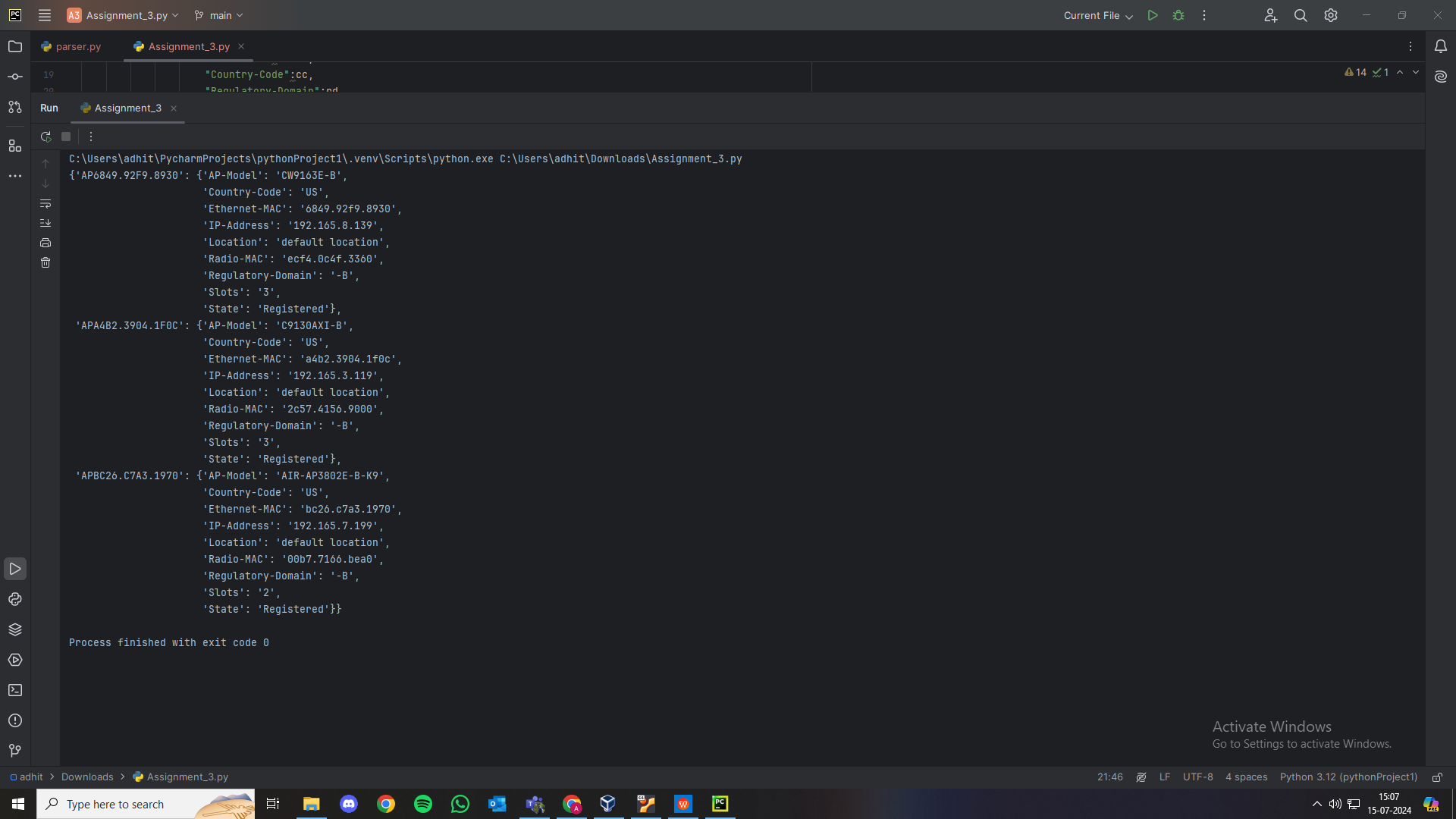
AP6849.92F9.8930 3 CW9163E-B 6849.92f9.8930 ecf4.0c4f.3360 US -B 192.165.8.139 Registered default location

ANSWER

**CODE**

import re  
from pprint import pprint  
  
def parser(output):  
 *# Define the regular expression pattern to match each line of the output* pattern = re.compile(r'(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+\s+\S+)\s')  
  
 *# Iterate over each line of the output, skipping the first header line* for line in output.splitlines()[3:]:  
 match = pattern.match(line)  
 if match:  
 ap , slots, model, ether, radio, cc, rd, ip\_address, state, location= match.groups()  
 yield {  
 ap: {  
 "Slots":slots,  
 "AP-Model":model,  
 "Ethernet-MAC":ether,  
 "Radio-MAC":radio,  
 "Country-Code":cc,  
 "Regulatory-Domain":rd,  
 "IP-Address": ip\_address,  
 "State": state,  
 "Location": location  
 }  
 }  
  
  
output = """  
AP Name Slots AP Model Ethernet MAC Radio MAC CC RD IP Address State Location  
-------------------------------------------------------------------------------------------------------------------------------------------------------------  
APBC26.C7A3.1970 2 AIR-AP3802E-B-K9 bc26.c7a3.1970 00b7.7166.bea0 US -B 192.165.7.199 Registered default location   
APA4B2.3904.1F0C 3 C9130AXI-B a4b2.3904.1f0c 2c57.4156.9000 US -B 192.165.3.119 Registered default location   
AP6849.92F9.8930 3 CW9163E-B 6849.92f9.8930 ecf4.0c4f.3360 US -B 192.165.8.139 Registered default location   
"""  
  
*# call parser function*ethernet\_interface = {}  
for dicts in parser(output):  
 ethernet\_interface.update(dicts)  
  
pprint(ethernet\_interface)





2. Using the parsed output of 1st question, write a script to perform the following:

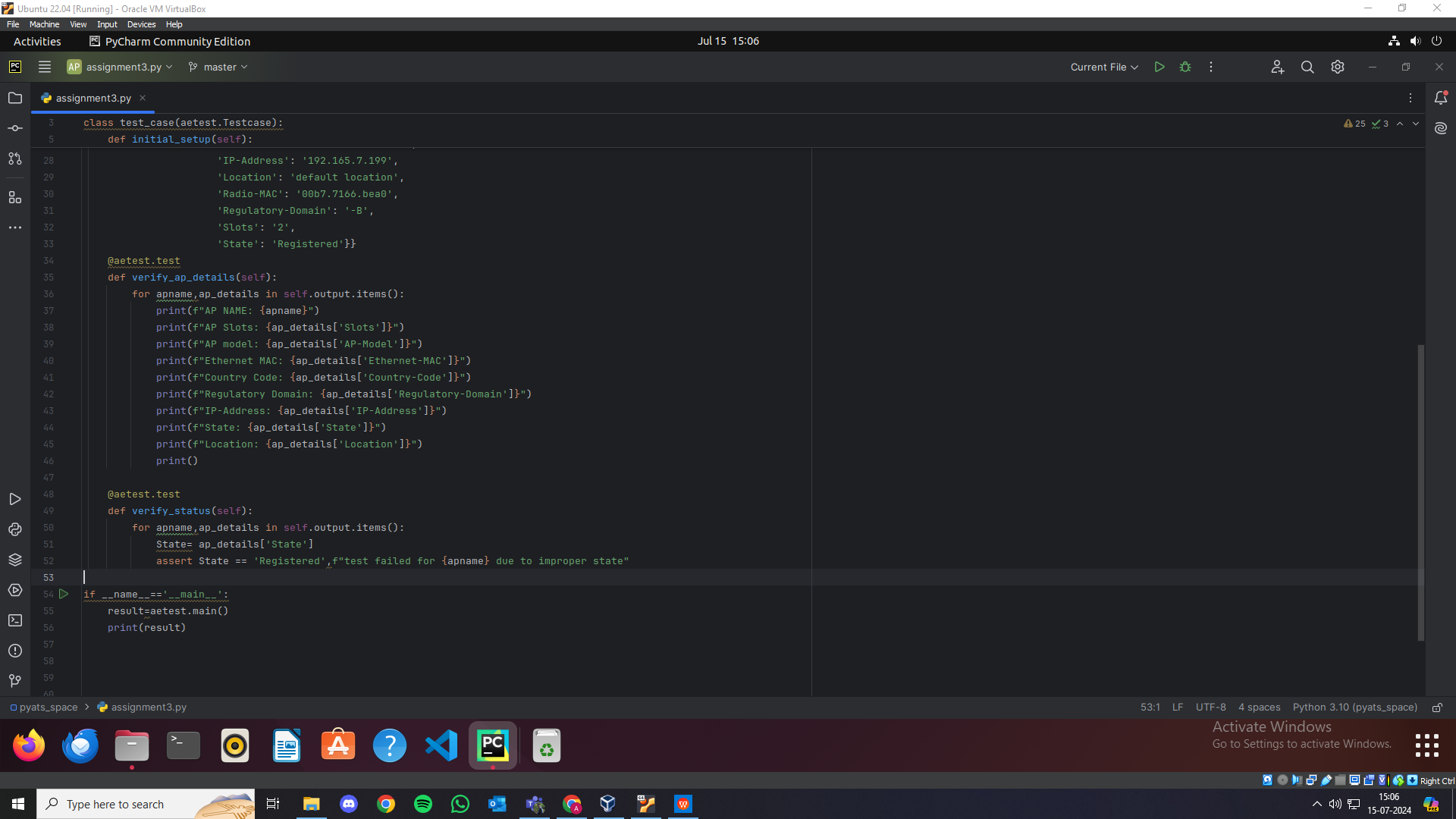
- Fetch No of Slots, AP Model, Ethernet MAC, Radio MAC, CC, RD, IP Address of all the three AP's (APBC26.C7A3.1970, APA4B2.3904.1F0C and AP6849.92F9.8930) and print them.

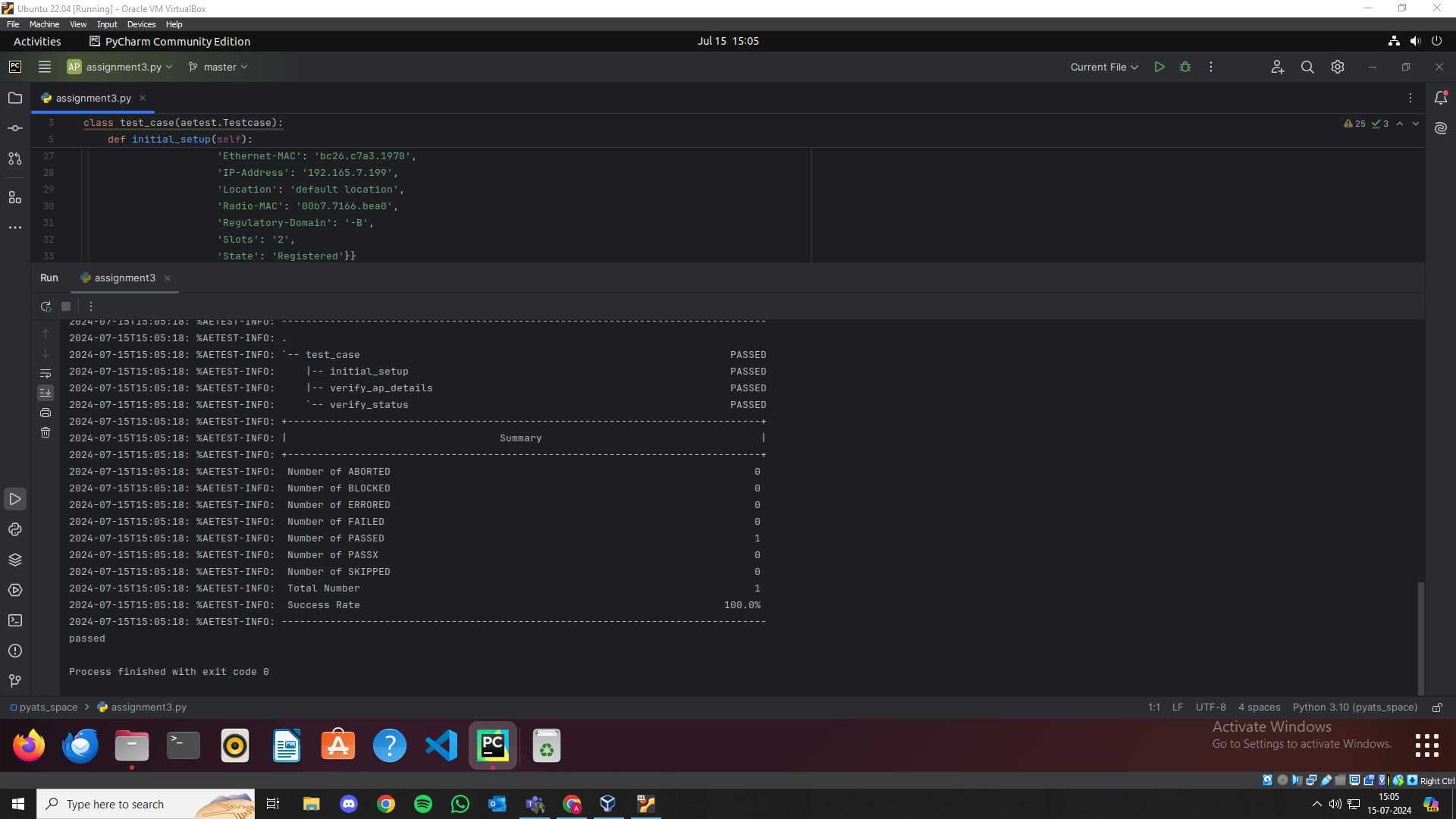
- Verify 'State' for all the three AP's. Pass/Fail Criteria: Test should pass if state is 'Registered', otherwise it should fail.

ANSWER:

**CODE**

from pyats import aetest  
  
class test\_case(aetest.Testcase):  
 @aetest.setup  
 def initial\_setup(self):  
 self.output={}  
 self.output={'AP6849.92F9.8930': {'AP-Model': 'CW9163E-B',  
 'Country-Code': 'US',  
 'Ethernet-MAC': '6849.92f9.8930',  
 'IP-Address': '192.165.8.139',  
 'Location': 'default location',  
 'Radio-MAC': 'ecf4.0c4f.3360',  
 'Regulatory-Domain': '-B',  
 'Slots': '3',  
 'State': 'Registered'},  
 'APA4B2.3904.1F0C': {'AP-Model': 'C9130AXI-B',  
 'Country-Code': 'US',  
 'Ethernet-MAC': 'a4b2.3904.1f0c',  
 'IP-Address': '192.165.3.119',  
 'Location': 'default location',  
 'Radio-MAC': '2c57.4156.9000',  
 'Regulatory-Domain': '-B',  
 'Slots': '3',  
 'State': 'Registered'},  
 'APBC26.C7A3.1970': {'AP-Model': 'AIR-AP3802E-B-K9',  
 'Country-Code': 'US',  
 'Ethernet-MAC': 'bc26.c7a3.1970',  
 'IP-Address': '192.165.7.199',  
 'Location': 'default location',  
 'Radio-MAC': '00b7.7166.bea0',  
 'Regulatory-Domain': '-B',  
 'Slots': '2',  
 'State': 'Registered'}}  
 @aetest.test  
 def verify\_ap\_details(self):  
 for apname,ap\_details in self.output.items():  
 print(f"AP NAME: {apname}")  
 print(f"AP Slots: {ap\_details['Slots']}")  
 print(f"AP model: {ap\_details['AP-Model']}")  
 print(f"Ethernet MAC: {ap\_details['Ethernet-MAC']}")  
 print(f"Country Code: {ap\_details['Country-Code']}")  
 print(f"Regulatory Domain: {ap\_details['Regulatory-Domain']}")  
 print(f"IP-Address: {ap\_details['IP-Address']}")  
 print(f"State: {ap\_details['State']}")  
 print(f"Location: {ap\_details['Location']}")  
 print()  
  
 @aetest.test  
 def verify\_status(self):  
 for apname,ap\_details in self.output.items():  
 State= ap\_details['State']  
 assert State == 'Registered',f"test failed for {apname} due to improper state"  
  
if \_\_name\_\_=='\_\_main\_\_':  
 result=aetest.main()  
 print(result)





Note:

Write the code for both questions in pyATS format:

- Create a separate function for parser

- Perform the operations of Question 2 in script file using the parsed output

- Write a separate function for verifying AP state and use that in script

- Have the main function with run API in job file

Run it in any compiler to obtain the output.