**Lab Exercise 15- More Complex Flow test with multiple test cases using PyTest in Metaflow**

We’ll modify the flow to accept a list of numbers through an environment variable or default to [1, 2, 3, 4, 5] if none is provided.

**Code for complex\_flow.py**

from metaflow import FlowSpec, step

import os

import json

class ComplexFlow(FlowSpec):

@step

def start(self):

# Get numbers from environment variable or use default

numbers\_env = os.getenv('NUMBERS', '[1, 2, 3, 4, 5]')

self.numbers = json.loads(numbers\_env) # Parse the JSON string to a list

self.next(self.square\_numbers)

@step

def square\_numbers(self):

self.squared\_numbers = [x \*\* 2 for x in self.numbers]

self.next(self.sum\_squares)

@step

def sum\_squares(self):

self.total\_sum = sum(self.squared\_numbers)

self.next(self.find\_max)

@step

def find\_max(self):

self.max\_value = max(self.squared\_numbers) if self.squared\_numbers else None

self.next(self.end)

@step

def end(self):

print(f"Numbers: {self.numbers}")

print(f"Squared Numbers: {self.squared\_numbers}")

print(f"Total Sum of Squares: {self.total\_sum}")

print(f"Maximum Squared Value: {self.max\_value}")

if \_\_name\_\_ == '\_\_main\_\_':

ComplexFlow()

**Explanation of Changes**

1. **Dynamic Input Handling**:
   * The flow retrieves the list of numbers from an environment variable called NUMBERS.
   * If the variable is not set, it defaults to [1, 2, 3, 4, 5].
   * The numbers are parsed from a JSON string to a Python list.
2. **Handling Edge Cases**:
   * In the find\_max step, we check if squared\_numbers is empty before trying to find the maximum value, ensuring it handles cases like an empty list gracefully.

File: **test\_complex\_flow.py**

import os

from metaflow import Flow

import subprocess

import pytest

def run\_flow():

# Command to run the flow

cmd = ['python', 'complex\_flow.py', 'run', '--run-id-file', 'test\_id']

subprocess.check\_call(cmd)

# Read the run ID from the file

with open('test\_id') as f:

run\_id = f.read().strip() # Use .strip() to remove any extra whitespace

# Load the flow run and return it for assertions

return Flow('ComplexFlow')[run\_id]

def test\_flow\_with\_default\_numbers():

run = run\_flow()

# Assertions to validate the outputs for default numbers

assert run.data.numbers == [1, 2, 3, 4, 5]

assert run.data.squared\_numbers == [1, 4, 9, 16, 25]

assert run.data.total\_sum == 55

assert run.data.max\_value == 25

def test\_flow\_with\_negative\_numbers():

# Modify the flow to accept this input if necessary

run = run\_flow()

# Assertions to validate the outputs for negative numbers

assert run.data.numbers == [-1, -2, -3]

assert run.data.squared\_numbers == [1, 4, 9]

assert run.data.total\_sum == 14

assert run.data.max\_value == 9

def test\_flow\_with\_empty\_list():

# Modify the flow to accept this input if necessary

run = run\_flow()

# Assertions to validate the outputs for an empty list

assert run.data.numbers == []

assert run.data.squared\_numbers == []

assert run.data.total\_sum == 0

assert run.data.max\_value is None # or raise an exception if preferred

def test\_flow\_with\_single\_element():

# Modify the flow to accept this input if necessary

run = run\_flow()

# Assertions to validate the outputs for a single element

assert run.data.numbers == [7]

assert run.data.squared\_numbers == [49]

assert run.data.total\_sum == 49

assert run.data.max\_value == 49

**Running Tests with Different Inputs**

To run the tests with specific inputs, you can set the environment variable NUMBERS before executing your test command. For example:

export NUMBERS='[-1, -2, -3]' # For negative numbers test

pytest test\_complex\_flow.py

You can similarly set the variable for other tests. If you want to run tests without setting environment variables, you can modify the run\_flow function in your test code to set the environment variable just before calling the flow.

**Summary**

This updated complex\_flow.py allows for flexible testing while handling multiple scenarios as described in the test cases. If you have any further questions or need additional modifications, feel free to ask!