tests/test\_api\_integrator.py

import unittest

from unittest.mock import patch, mock\_open

from utils.api\_integrator import send\_data\_for\_analysis

class TestAPIIntegrator(unittest.TestCase):

@patch('utils.api\_integrator.openai.ChatCompletion.create')

@patch('builtins.open', new\_callable=mock\_open)

def test\_send\_data\_for\_analysis\_success(self, mock\_open, mock\_openai\_create):

mock\_openai\_create.return\_value = {

'choices': [{'message': {'content': 'Pipeline performance is optimal with no improvements needed.'}}]

}

api\_key = 'fake\_api\_key'

data = {'instructions\_executed': 150, 'stalls': 5, 'branch\_mispredictions': 2}

analysis\_path = '/fake/data/pipeline\_analysis.txt'

send\_data\_for\_analysis(api\_key, data, analysis\_path)

mock\_openai\_create.assert\_called\_once()

mock\_open.assert\_called\_once\_with(analysis\_path, 'w')

mock\_file = mock\_open()

mock\_file.return\_value.write.assert\_called\_once\_with('Pipeline performance is optimal with no improvements needed.')

```

I changed the last assertion to `mock\_file.return\_value.write.assert\_called\_once\_with`, because `mock\_open` returns a mock file object, and we need to access its `write` method.