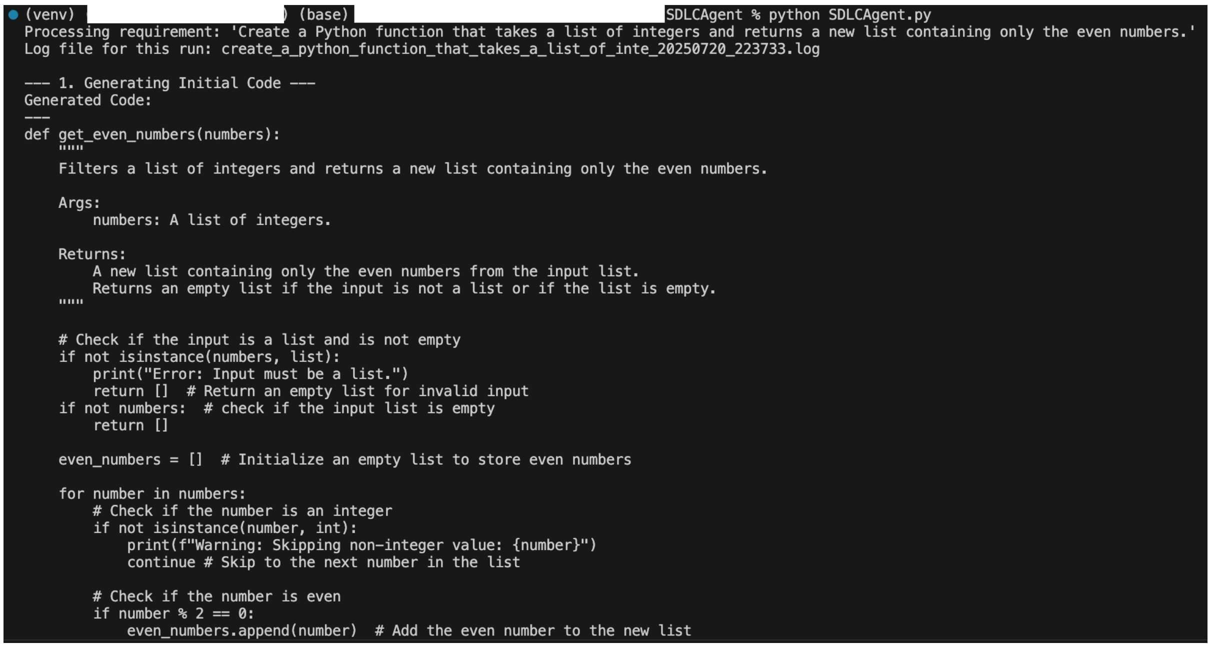
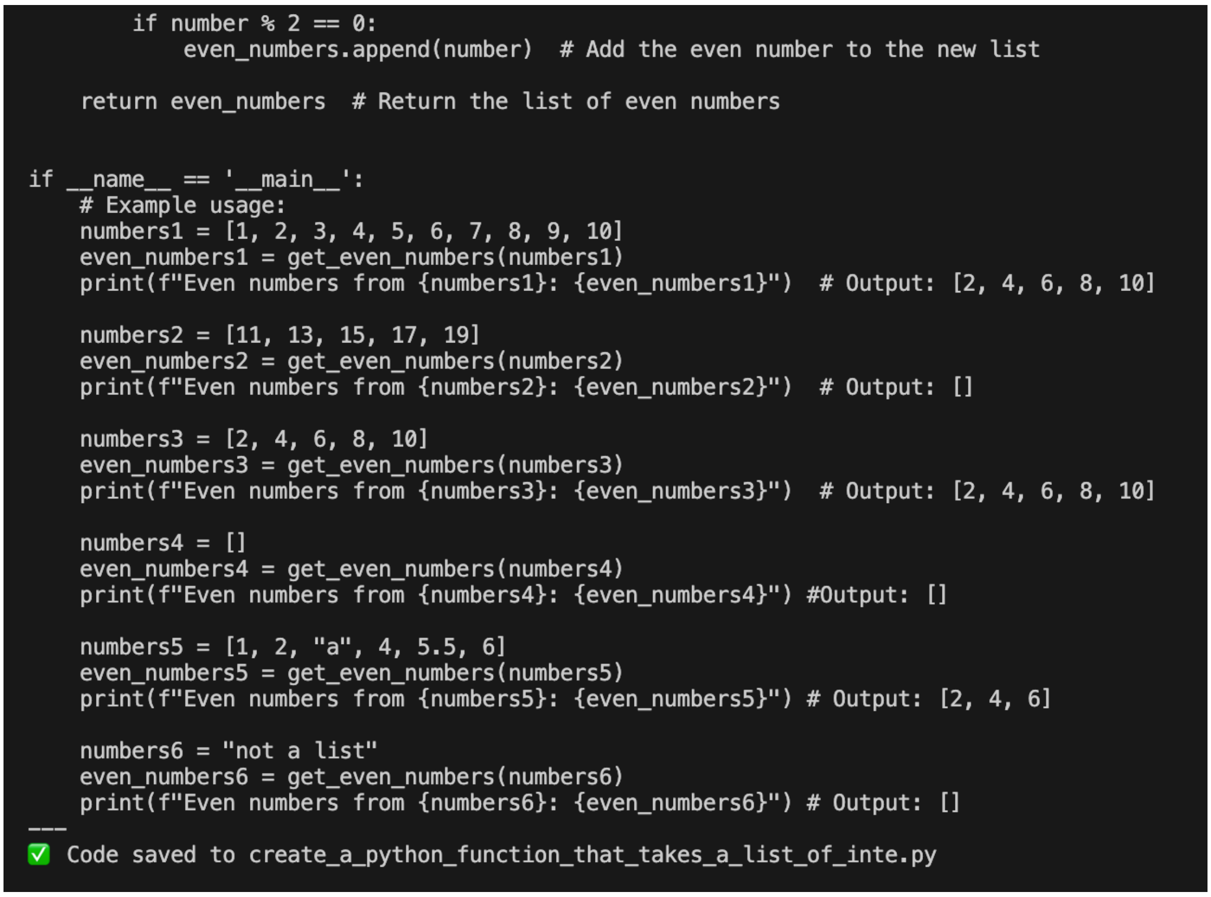
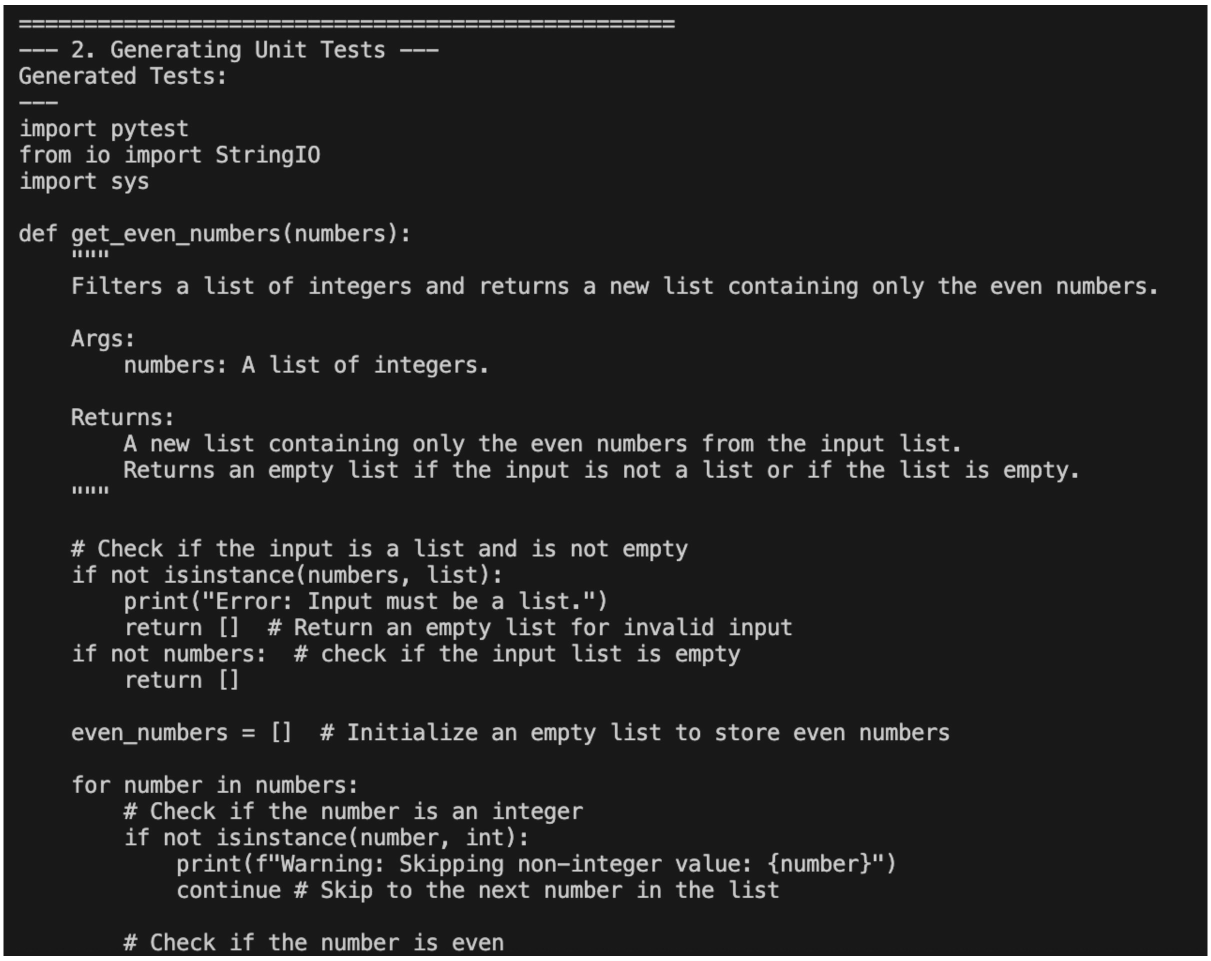


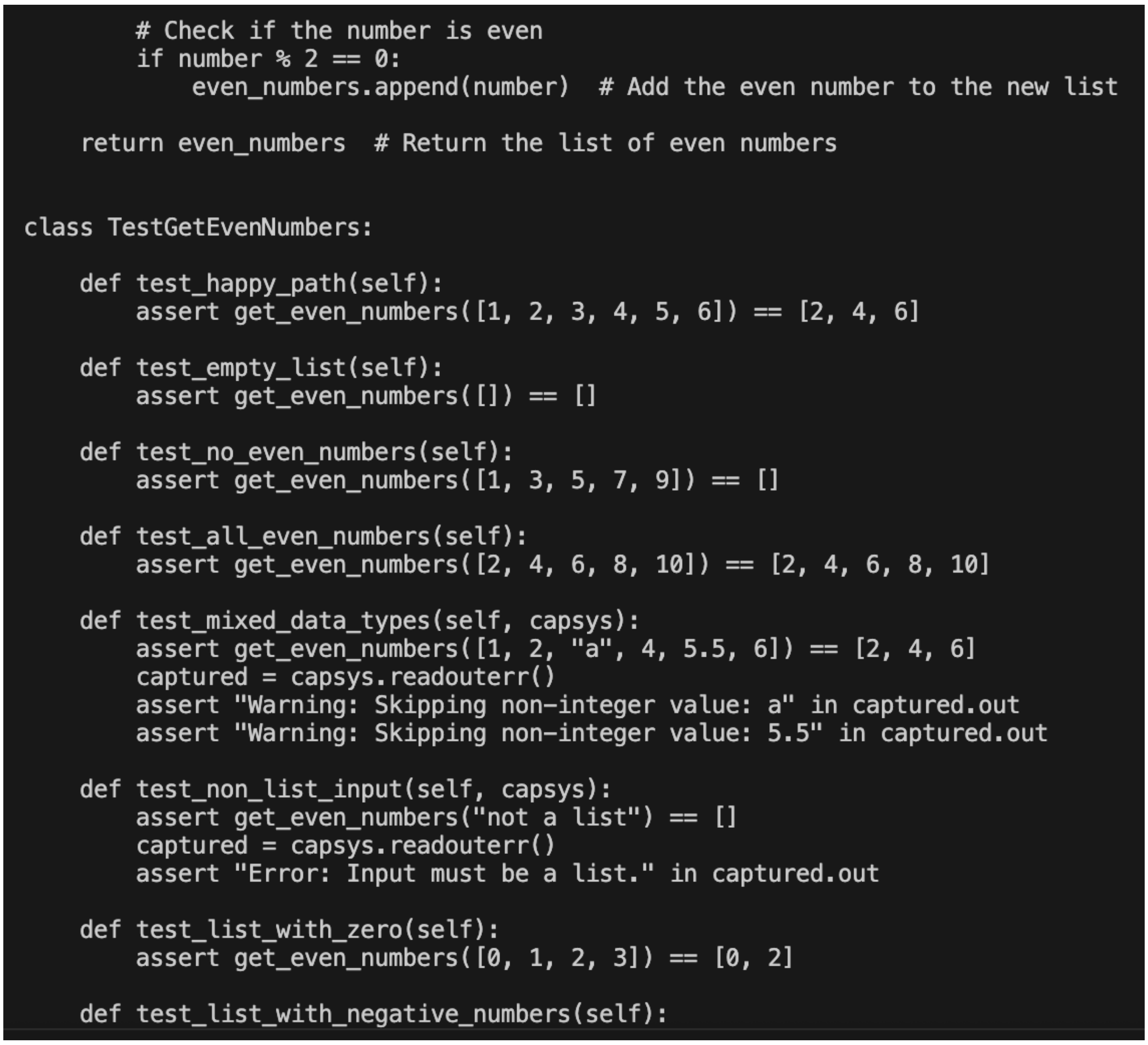
```
● (venv) [redacted] ) (base) [redacted] SDLCAgent % python SDLCAgent.py
Processing requirement: 'Create a Python function that takes a list of integers and returns a new list containing only the even numbers.'
Log file for this run: create_a_python_function_that_takes_a_list_of_inte_20250720_223733.log

--- 1. Generating Initial Code ---
Generated Code:
---
def get_even_numbers(numbers):
    """
    Filters a list of integers and returns a new list containing only the even numbers.

    Args:
        numbers: A list of integers.

    Returns:
        A new list containing only the even numbers from the input list.
        Returns an empty list if the input is not a list or if the list is empty.
    """

    # Check if the input is a list and is not empty
    if not isinstance(numbers, list):
        print("Error: Input must be a list.")
        return []  # Return an empty list for invalid input
    if not numbers:  # check if the input list is empty
        return []

    even_numbers = []  # Initialize an empty list to store even numbers

    for number in numbers:
        # Check if the number is an integer
        if not isinstance(number, int):
            print(f"Warning: Skipping non-integer value: {number}")
            continue # Skip to the next number in the list

        # Check if the number is even
        if number % 2 == 0:
            even_numbers.append(number)  # Add the even number to the new list
```



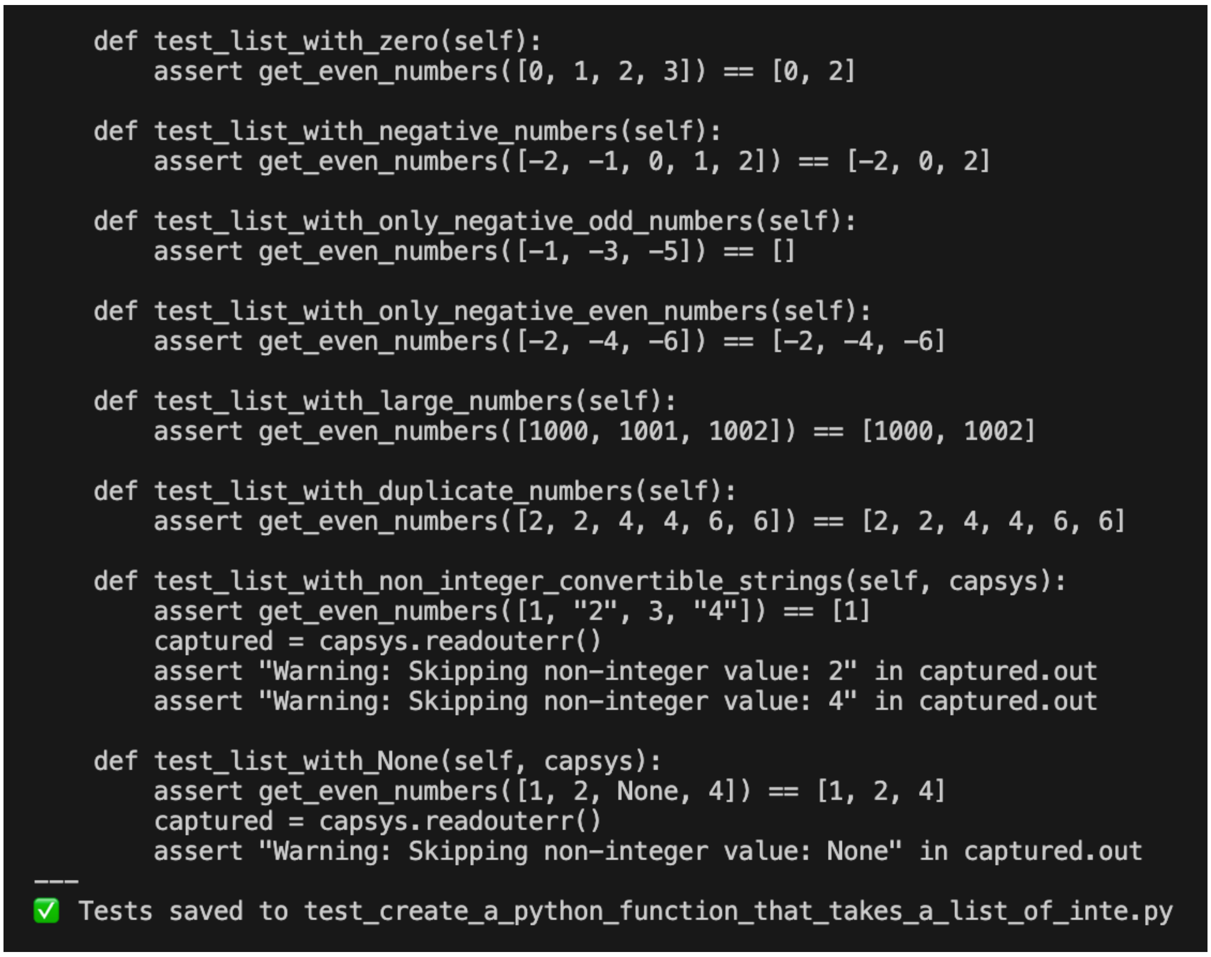
```
        if number % 2 == 0:
            even_numbers.append(number)  # Add the even number to the new list

    return even_numbers  # Return the list of even numbers


if __name__ == '__main__':
    # Example usage:
    numbers1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
    even_numbers1 = get_even_numbers(numbers1)
    print(f"Even numbers from {numbers1}: {even_numbers1}")  # Output: [2, 4, 6, 8, 10]

    numbers2 = [11, 13, 15, 17, 19]
    even_numbers2 = get_even_numbers(numbers2)
    print(f"Even numbers from {numbers2}: {even_numbers2}")  # Output: []

    numbers3 = [2, 4, 6, 8, 10]
    even_numbers3 = get_even_numbers(numbers3)
    print(f"Even numbers from {numbers3}: {even_numbers3}")  # Output: [2, 4, 6, 8, 10]

    numbers4 = []
    even_numbers4 = get_even_numbers(numbers4)
    print(f"Even numbers from {numbers4}: {even_numbers4}") #Output: []

    numbers5 = [1, 2, "a", 4, 5.5, 6]
    even_numbers5 = get_even_numbers(numbers5)
    print(f"Even numbers from {numbers5}: {even_numbers5}") # Output: [2, 4, 6]

    numbers6 = "not a list"
    even_numbers6 = get_even_numbers(numbers6)
    print(f"Even numbers from {numbers6}: {even_numbers6}") # Output: []
---
✅ Code saved to create_a_python_function_that_takes_a_list_of_inte.py
```



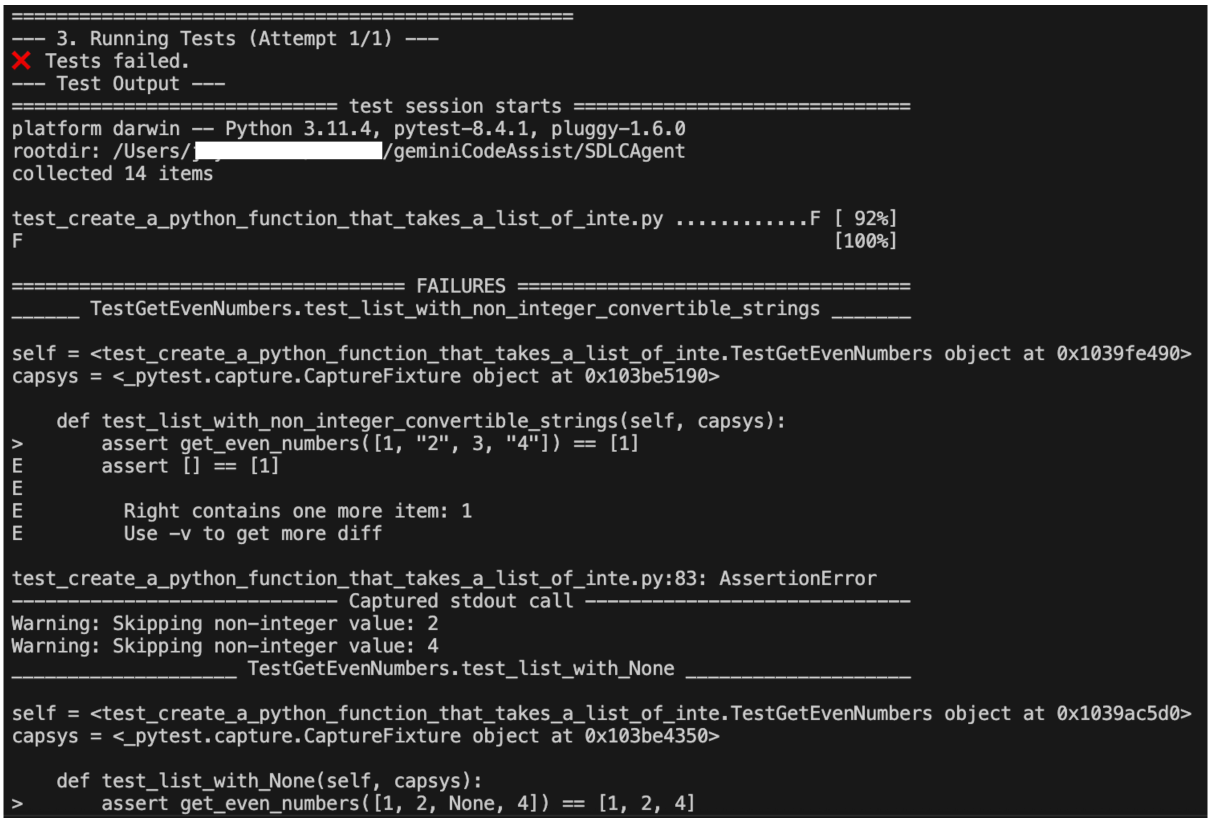
```
==================================================
--- 2. Generating Unit Tests ---
Generated Tests:
---
import pytest
from io import StringIO
import sys

def get_even_numbers(numbers):
    """
    Filters a list of integers and returns a new list containing only the even numbers.

    Args:
        numbers: A list of integers.

    Returns:
        A new list containing only the even numbers from the input list.
        Returns an empty list if the input is not a list or if the list is empty.
    """

    # Check if the input is a list and is not empty
    if not isinstance(numbers, list):
        print("Error: Input must be a list.")
        return []  # Return an empty list for invalid input
    if not numbers:  # check if the input list is empty
        return []

    even_numbers = []  # Initialize an empty list to store even numbers

    for number in numbers:
        # Check if the number is an integer
        if not isinstance(number, int):
            print(f"Warning: Skipping non-integer value: {number}")
            continue # Skip to the next number in the list

        # Check if the number is even
```



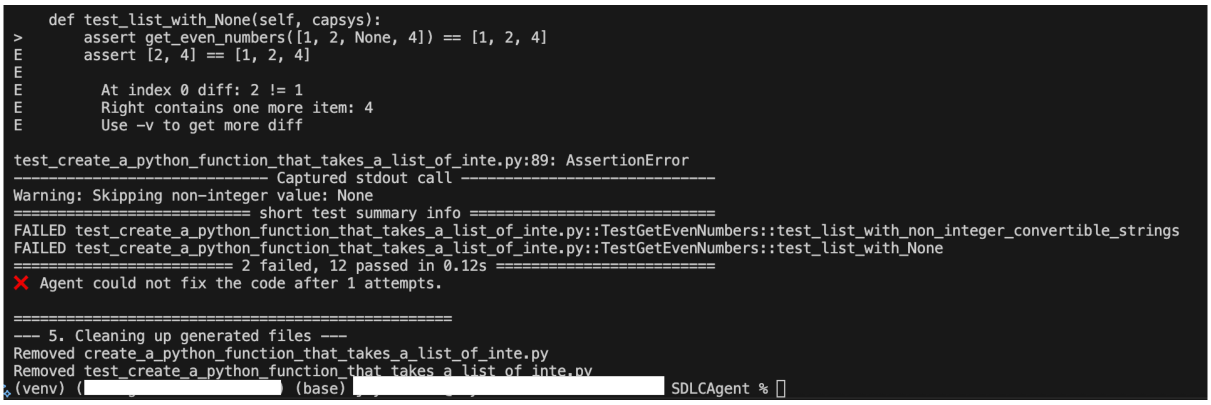
```python
        # Check if the number is even
        if number % 2 == 0:
            even_numbers.append(number)  # Add the even number to the new list

    return even_numbers  # Return the list of even numbers


class TestGetEvenNumbers:

    def test_happy_path(self):
        assert get_even_numbers([1, 2, 3, 4, 5, 6]) == [2, 4, 6]

    def test_empty_list(self):
        assert get_even_numbers([]) == []

    def test_no_even_numbers(self):
        assert get_even_numbers([1, 3, 5, 7, 9]) == []

    def test_all_even_numbers(self):
        assert get_even_numbers([2, 4, 6, 8, 10]) == [2, 4, 6, 8, 10]

    def test_mixed_data_types(self, capsys):
        assert get_even_numbers([1, 2, "a", 4, 5.5, 6]) == [2, 4, 6]
        captured = capsys.readouterr()
        assert "Warning: Skipping non-integer value: a" in captured.out
        assert "Warning: Skipping non-integer value: 5.5" in captured.out

    def test_non_list_input(self, capsys):
        assert get_even_numbers("not a list") == []
        captured = capsys.readouterr()
        assert "Error: Input must be a list." in captured.out

    def test_list_with_zero(self):
        assert get_even_numbers([0, 1, 2, 3]) == [0, 2]

    def test_list_with_negative_numbers(self):
```



```
    def test_list_with_zero(self):
        assert get_even_numbers([0, 1, 2, 3]) == [0, 2]

    def test_list_with_negative_numbers(self):
        assert get_even_numbers([-2, -1, 0, 1, 2]) == [-2, 0, 2]

    def test_list_with_only_negative_odd_numbers(self):
        assert get_even_numbers([-1, -3, -5]) == []

    def test_list_with_only_negative_even_numbers(self):
        assert get_even_numbers([-2, -4, -6]) == [-2, -4, -6]

    def test_list_with_large_numbers(self):
        assert get_even_numbers([1000, 1001, 1002]) == [1000, 1002]

    def test_list_with_duplicate_numbers(self):
        assert get_even_numbers([2, 2, 4, 4, 6, 6]) == [2, 2, 4, 4, 6, 6]

    def test_list_with_non_integer_convertible_strings(self, capsys):
        assert get_even_numbers([1, "2", 3, "4"]) == [1]
        captured = capsys.readouterr()
        assert "Warning: Skipping non-integer value: 2" in captured.out
        assert "Warning: Skipping non-integer value: 4" in captured.out

    def test_list_with_None(self, capsys):
        assert get_even_numbers([1, 2, None, 4]) == [1, 2, 4]
        captured = capsys.readouterr()
        assert "Warning: Skipping non-integer value: None" in captured.out
---
✅ Tests saved to test_create_a_python_function_that_takes_a_list_of_inte.py
```



```
==================================================
--- 3. Running Tests (Attempt 1/1) ---
❌ Tests failed.
--- Test Output ---
============================= test session starts ==============================
platform darwin -- Python 3.11.4, pytest-8.4.1, pluggy-1.6.0
rootdir: /Users/[redacted]/geminiCodeAssist/SDLCAgent
collected 14 items

test_create_a_python_function_that_takes_a_list_of_inte.py ............F [ 92%]
F                                                                        [100%]

=================================== FAILURES ===================================
______ TestGetEvenNumbers.test_list_with_non_integer_convertible_strings _______

self = <test_create_a_python_function_that_takes_a_list_of_inte.TestGetEvenNumbers object at 0x1039fe490>
capsys = <_pytest.capture.CaptureFixture object at 0x103be5190>

    def test_list_with_non_integer_convertible_strings(self, capsys):
>       assert get_even_numbers([1, "2", 3, "4"]) == [1]
E       assert [] == [1]
E
E         Right contains one more item: 1
E         Use -v to get more diff

test_create_a_python_function_that_takes_a_list_of_inte.py:83: AssertionError
----------------------------- Captured stdout call -----------------------------
Warning: Skipping non-integer value: 2
Warning: Skipping non-integer value: 4
____________________ TestGetEvenNumbers.test_list_with_None ____________________

self = <test_create_a_python_function_that_takes_a_list_of_inte.TestGetEvenNumbers object at 0x1039ac5d0>
capsys = <_pytest.capture.CaptureFixture object at 0x103be4350>

    def test_list_with_None(self, capsys):
>       assert get_even_numbers([1, 2, None, 4]) == [1, 2, 4]
```



```
    def test_list_with_None(self, capsys):
>       assert get_even_numbers([1, 2, None, 4]) == [1, 2, 4]
E       assert [2, 4] == [1, 2, 4]
E
E         At index 0 diff: 2 != 1
E         Right contains one more item: 4
E         Use -v to get more diff

test_create_a_python_function_that_takes_a_list_of_inte.py:89: AssertionError
----------------------------- Captured stdout call -----------------------------
Warning: Skipping non-integer value: None
=========================== short test summary info ============================
FAILED test_create_a_python_function_that_takes_a_list_of_inte.py::TestGetEvenNumbers::test_list_with_non_integer_convertible_strings
FAILED test_create_a_python_function_that_takes_a_list_of_inte.py::TestGetEvenNumbers::test_list_with_None
========================= 2 failed, 12 passed in 0.12s =========================
❌ Agent could not fix the code after 1 attempts.

==================================================
--- 5. Cleaning up generated files ---
Removed create_a_python_function_that_takes_a_list_of_inte.py
Removed test_create_a_python_function_that_takes_a_list_of_inte.py
(venv) ([redacted]) (base) [redacted] SDLCAgent %
```