Automate Test Documentation for Weather Information Application

This document provides a detailed overview of the automated tests conducted on the Weather Information application. The tests are designed to validate various functionalities of application, including fetching city summary, fetching current temperature, testing weather and Wikipedia APIs, and writing output to a file.

The tests are organized into four files, each focusing on a specific functionality:

1. **test_fetch_city_summary.py**: Contains tests for fetching current temperature.
2. **test_fetch_current_temperature.py**: Contains tests for fetching current temperature.
3. **test_weather_info_api.py**: Contains tests for testing weather and Wikipedia APIs.
4. **test_write_output_to_file.py**: Contains a test for writing output to a file.

The tests are automated using pytest, and are executed in an automated test environment. The test strategy is functional testing, where each test case is designed to validate a specific function of the application.

The risks associated with these tests include potential issues with the application's functionality, such as problems with fetching city summary, fetching current temperature, testing APIs, and writing output to a file. These could affect the user's ability to use the application.

By providing a detailed description of each test case, including the test steps, expected results, and risks, this document serves as a comprehensive guide to the automated testing process for the Weather Information application.

Test Fetch City Summary

Test case description

This test case verifies that the fetch_city_summary function returns a string when a valid city name is provided, and None when an invalid city name, an empty string, a string of spaces, or a string of special characters is provided.

Preconditions

The fetch_city_summary function is available.

Test steps

1. Call the fetch_city_summary function with a valid city name, an invalid city name, an empty string, a string of spaces, and a string of special characters.

Expected result

1. The function returns a string when a valid city name is provided, and None otherwise.

Test strategy

This is a functional test case that is automated using pytest.

Context

This test case is executed in an automated test environment.

Risks

If this test case fails, it could indicate a problem with the fetch_city_summary function. This could affect the user's ability to fetch the city summary.

Test Fetch Current Temperature

Test case description

This test case verifies that the fetch_current_temperature function returns a float when a valid city name is provided, and None when an invalid city name, an empty string, a string of spaces, or a string of special characters is provided.

Preconditions

The fetch_current_temperature function is available.

Test steps

1. Call the fetch_ current_temperature function with a valid city name, an invalid city name, an empty string, a string of spaces, and a string of special characters.

Expected result

1. The function returns a float when a valid city name is provided, and None otherwise.

Test strategy

This is a functional test case that is automated using pytest.

Context

This test case is executed in an automated test environment.

Risks

If this test case fails, it could indicate a problem with the fetch_current_temperature function. This could affect the user's ability to fetch the current temperature.

Test Weather and Wikipedia APIs

Test case description

This test case verifies that the weather and Wikipedia APIs return the expected responses when a valid city name is provided.

Preconditions

The weather and Wikipedia APIs are available.

Test steps

1. Make a request to the weather and Wikipedia APIs with a valid city name.

Expected result

1. The APIs return the expected responses.

Test strategy

This is a functional test case that is automated using pytest.

Context

This test case is executed in an automated test environment.

Risks

If this test case fails, it could indicate a problem with the weather or Wikipedia APIs. This could affect the user's ability to fetch weather information or city summary.

Test Write Output to File

Test case description

This test case verifies that the write_output_to_file function creates a file with the correct content when a valid city name, city summary, and current temperature are provided.

Preconditions

The write_output_to_file function is available.

Test steps

1. Call the write_output_to_file function with a valid city name, city summary, and current temperature.

Expected result

1. The function creates a file with the correct content.

Test strategy

This is a functional test case that is automated using pytest.

Context

This test case is executed in an automated test environment.

Risks

If this test case fails, it could indicate a problem with the write_output_to_file function. This could affect the user's ability to write output to a file.