Test Report

Run ID: N/A • Generated: 2026-01-14 04:36:34 • Duration: 51ms
Plugin: v0.1.0 (b7a157f6cb9189cc50a17c846484c8454deeac61) [dirty]
Repo: v0.1.0 (b7a157f6cb9189cc50a17c846484c8454deeac61) [dirty]
LLM: ollama / llama3.2 (complete context, 4 annotated)

100.0%
Total Coverage

4	4	0	0
TOTAL TESTS	PASSED	FAILED	SKIPPED

0	0	0
XFAILED	XPASSED	ERRORS

Source Coverage Per Test Details Failures Only

Source Coverage

FILE	STMTS	MISS	COVER	%	COVERED LINES	MISSED LINES
example_pkg/calculator.py	6	0	6	100.0%	1, 3, 6, 8-10	-

📄 **tests/test_calculator.py** 4 tests

PASSED tests/test_calculator.py::test_add_basic 0ms 1

AI ASSESSMENT

Scenario: This test verifies that the `add` function returns the correct result for basic addition operations.

Why Needed: This test prevents bugs related to incorrect implementation or missing edge cases in the `add` function, ensuring it behaves as expected for simple arithmetic operations.

Key Assertions:

- The test checks if the `add` function returns the correct result for positive numbers (e.g., `add(1, 2)`).
- It verifies that the `add` function handles single-digit inputs correctly (e.g., `add(5, 3)`).
- This test also ensures the `add` function behaves as expected with zero input (e.g., `add(0, 4)`).

COVERAGE

example_pkg/calculator.py	1 lines (ranges: 3)

PASSED tests/test_calculator.py::test_add_negative 0ms 1

AI ASSESSMENT

Scenario: This test verifies that the addition function correctly handles negative numbers.

Why Needed: This test prevents bugs where incorrect results are produced when adding two negative numbers, potentially leading to incorrect calculations or errors in downstream code.

Key Assertions:

- The test checks if the result of add(-1, -1) is equal to -2
- It ensures that the function handles both negative inputs correctly
- The test also implicitly verifies that the function does not produce a division by zero error when given two negative numbers

COVERAGE

example_pkg/calculator.py	1 lines (ranges: 3)

PASSED tests/test_calculator.py::test_divide_success 0ms 1

AI ASSESSMENT

Scenario: This test verifies that the 'divide' function returns the correct result when dividing two non-zero numbers.

Why Needed: This test prevents division by zero errors, ensuring the calculator module handles this edge case correctly and provides informative error messages.

Key Assertions:

- The 'divide' function raises a ValueError with a descriptive message when attempting to divide by zero.
- The 'divide' function returns the correct result (5) for dividing 10 by 2.
- The 'divide' function performs integer division (not floating-point division) when both inputs are integers.
- The 'divide' function handles negative numbers correctly, returning a positive result as expected.

COVERAGE

example_pkg/calculator.py	2 lines (ranges: 8, 10)

PASSED tests/test_calculator.py::test_divide_zero 0ms 1

AI ASSESSMENT

Scenario: This test verifies that the calculator's divide function correctly raises a ValueError when attempting to divide by zero.

Why Needed: This test prevents potential bugs caused by silent division or incorrect results due to division by zero, which can lead to unexpected behavior in calculations.

Key Assertions:

- The function `divide` checks if the divisor is zero before performing the division.
- If the divisor is zero, it raises a ValueError with a descriptive error message.
- The test uses pytest's `raises` context manager to verify that the correct exception is raised.
- It ensures that the calculator's divide function behaves as expected in this edge case.
- By testing for division by zero, this test helps catch bugs early on and improves overall code reliability.

COVERAGE

example_pkg/calculator.py	2 lines (ranges: 8-9)