Code Understanding Report

Generated: 2025-05-06 16:03:35

This report presents automated insights based on large language models and code analysis tools.

File: pasted_code.py

Summary

- The code above is a decorator for logging function calls. It works by wrapping the function it's given and printing out a statement every time it's called. The print statements are formatted to include the name of the function being called, the arguments passed to it, and the result of the function.

It's common to use decorators for things like error handling, logging, or tracking function calls. This kind of decorator lets us add the same code to many functions in our codebase, without having to manually duplicate it for each function. - The provided code snippet is written in Python, a high-level programming language designed for simplicity. It includes a function definition which is called add. The function takes two parameters and returns the sum of these two parameters.

Explanation:

Python's syntax uses indentation to define the blocks of code, unlike some other programming languages. This feature makes the code readable and self-explanatory. In this case, the function add was defined with two parameters and a return statement.

Docstring

- ### Code: def log_calls(func): def wrapper(*args,* *kwargs): print(f"Calling {func.**name**} with arguments: {args}, {kwargs}") result = func(*args,* *kwargs) print(f"{func.**name**} returned: {result}") return result return wrapper

Docstring:

This function is a decorator that logs the calls to a function.

Parameters: - `func`: The function to be decorated.

Returns: - `wrapper`: A new function that logs the calls to - ### Code: def add(a, b): return a + b

Docstring:

''' Adds two numbers.

:param a: The first number. :param b: The second number. :return: The sum of a and b. '''

Test Cases:

Code Quality

Tool: `pylint`
Issues: 2`

```text [AST Parse Error] expected an indented block after function definition on line 1 (line 2)

---

[AST Parse Error] expected an indented block after function definition on line 1 (line 2) ```

## Conclusion

This Python codebase is a wrapper for logging function calls. By decorating our add function, every call to the add function will be logged, along with the arguments passed and the result. This makes it easy to track the usage of our code. The log_calls function serves as a convenient decorator, providing a simple way to log function calls in our codebase. The code snippet used in the explanation is a classic example of how to create a decorator in Python.
```