Calc.js Script

I am using Jest as my testing framework for my JavaScript calculator script. Jest is a popular and easy-to-use JavaScript testing library that provides various features such as mocking, snapshot testing, code coverage, and asynchronous testing

My testing strategy was to write unit tests for each of the calculator functions, such as add, subtract, multiply, divide etc.

Bit about Unit Tests

Unit tests are small and isolated tests that verify the functionality and behaviour of a single unit of code, such as a function or a module34.

To write unit tests in Jest, I use the test function, which takes two arguments: a descriptive string of what is being tested, and a callback function that contains the test logic and assertions.

For example, to test the add function, I can write something like this:

```
describe('Calculator functions', () => {
    test('add function should return the sum of two numbers', () => {
        expect(add(2, 3)).toBe(5);
        expect(add(-2, 3)).toBe(1);
        expect(add(0, 0)).toBe(0);
```

```
        expect(add(100, 200)).toBe(300);
        expect(add(-100, -200)).toBe(-300);
        expect(add(123456789, 987654321)).toBe(1111111110);
    });
```

To run the tests, I use the `npm test` command in the terminal, which invokes the Jest command defined in the package.json file. Jest will automatically find and execute all the test files that match the *.test.js pattern in the current directory.

By writing unit tests for each of the calculator functions, I can ensure that they work correctly and reliably, and that they handle several types of inputs and edge cases. I can also refactor or modify the code with confidence, knowing that the tests will catch any errors or bugs that might arise.