WORKSHOP

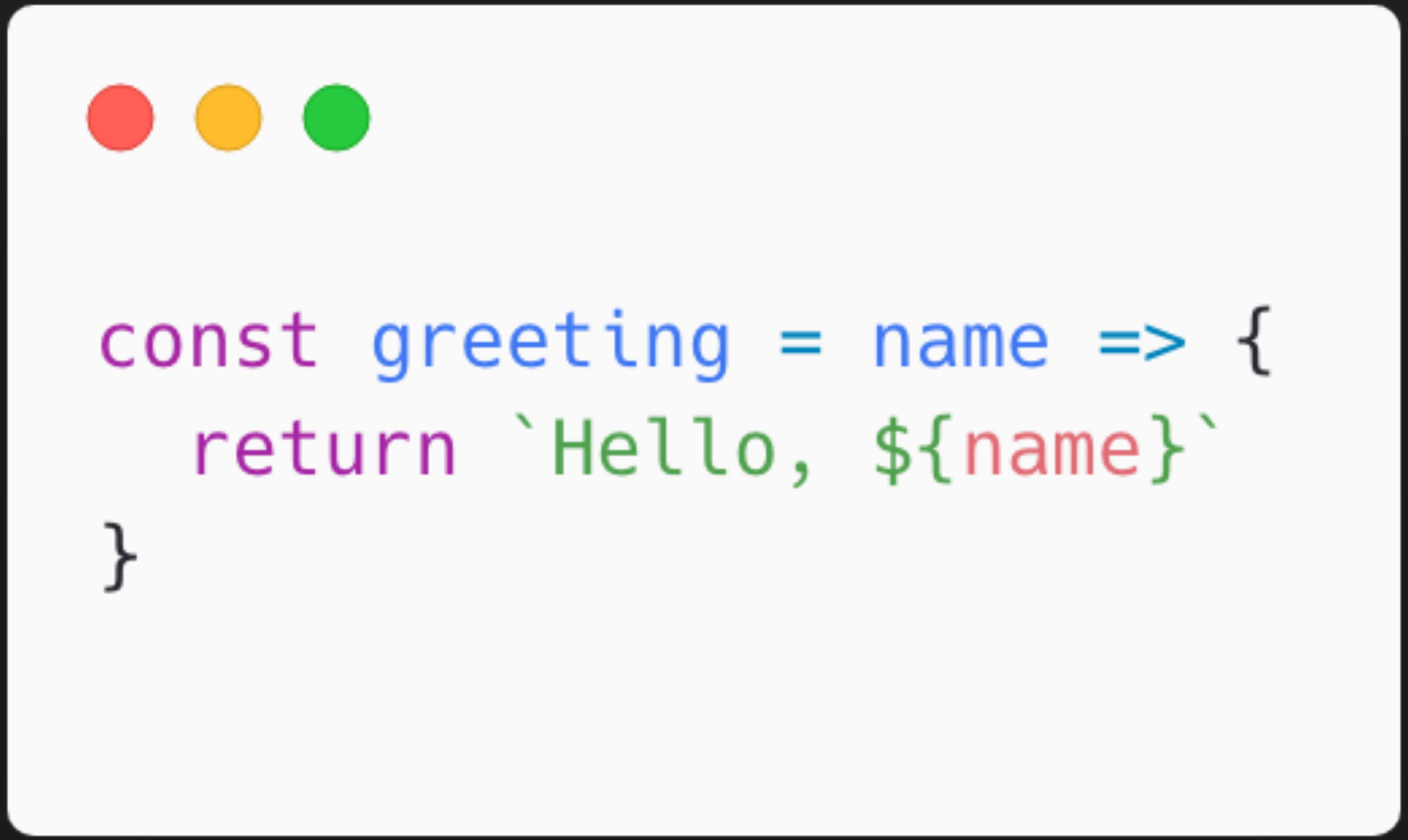
TESTING IN JAVASCRIPT

OUTLINE

- JavaScript testing.
- Frontend testing (React).
- Backend testing (Express).

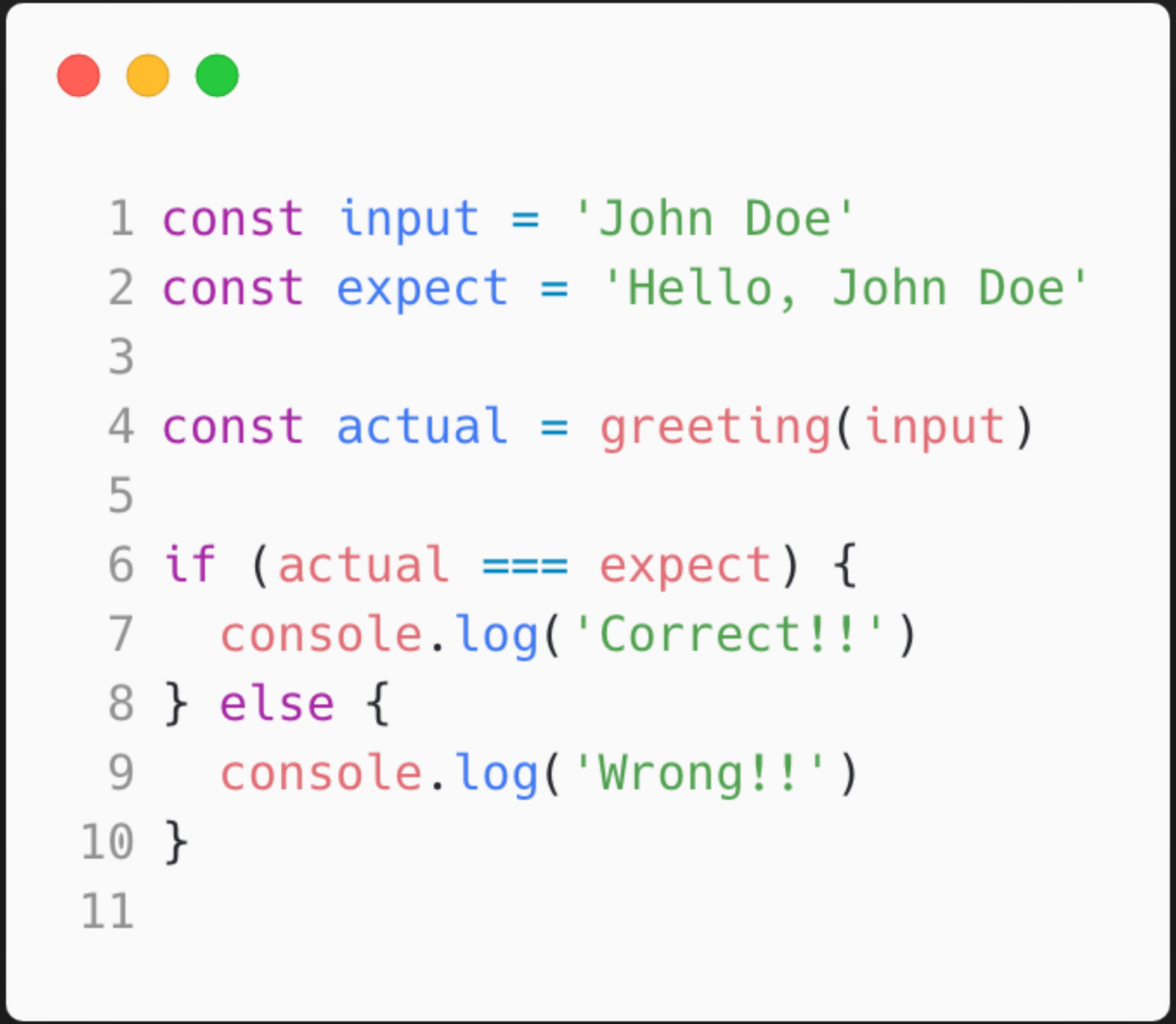
WHY TESTING

GREETING FUNCTION



```
const greeting = name => {
  return `Hello, ${name}`
}
```

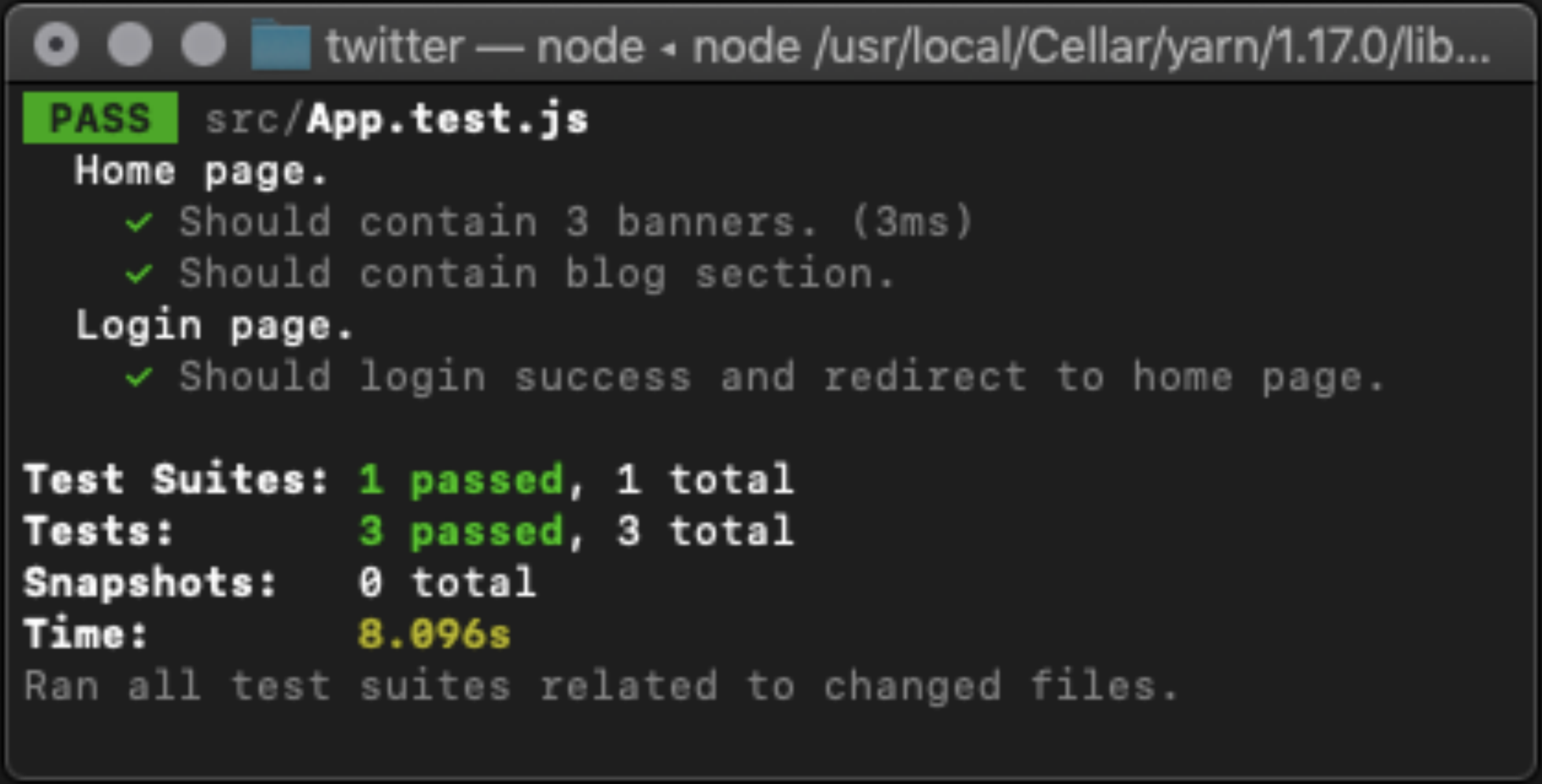
SIMPLE WAY TO WRITE TEST



```javascript
const input = 'John Doe'
const expect = 'Hello, John Doe'

const actual = greeting(input)

if (actual === expect) {
  console.log('Correct!!')
} else {
  console.log('Wrong!!')
}

```

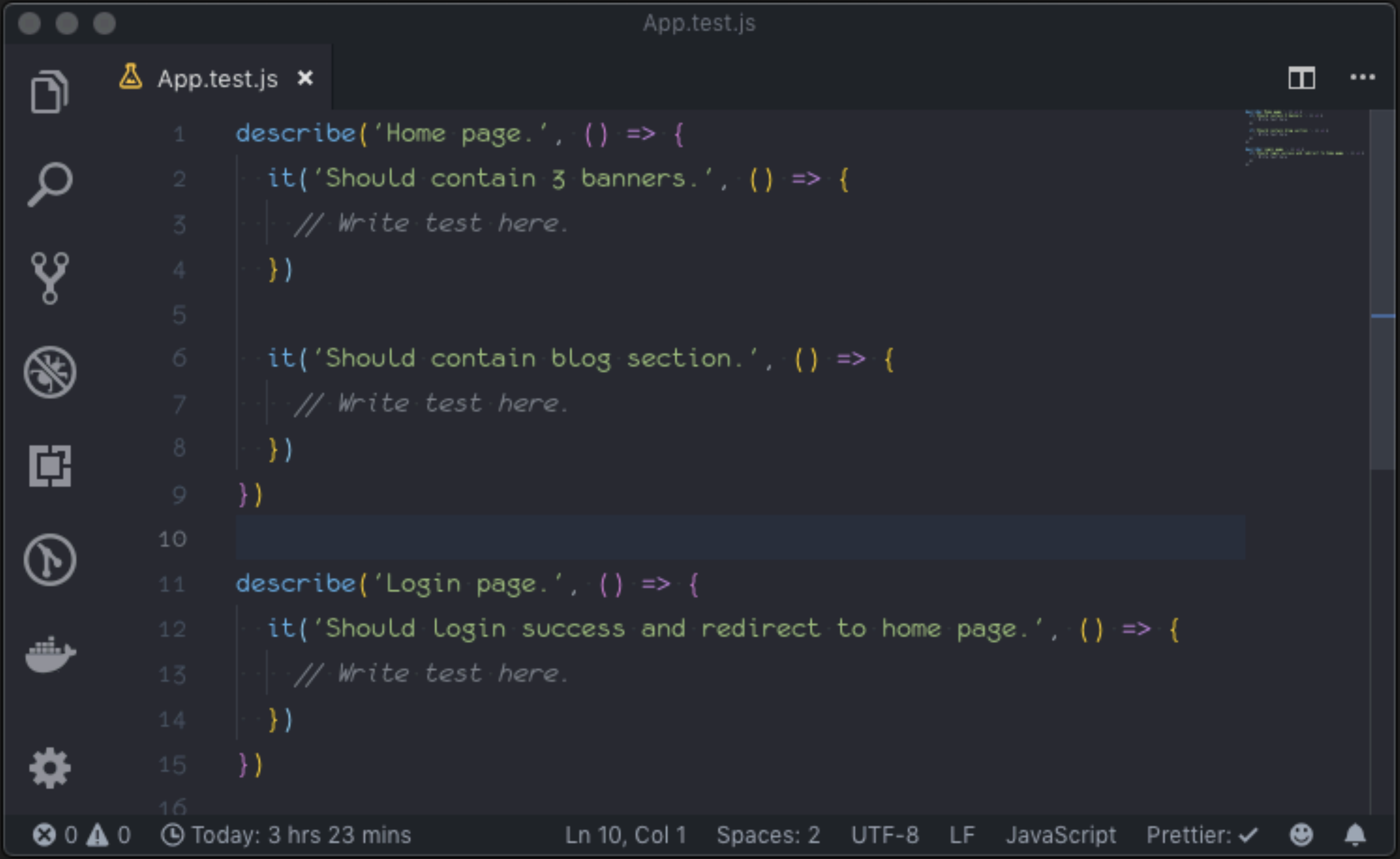
TEST RUNNER

NICE REPORT



```
PASS  src/App.test.js
  Home page.
    ✓ Should contain 3 banners. (3ms)
    ✓ Should contain blog section.
  Login page.
    ✓ Should login success and redirect to home page.

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        8.096s
Ran all test suites related to changed files.
```

IT, DESCRIBE



```
describe('Home page.', () => {
  it('Should contain 3 banners.', () => {
    // Write test here.
  })

  it('Should contain blog section.', () => {
    // Write test here.
  })
})

describe('Login page.', () => {
  it('Should login success and redirect to home page.', () => {
    // Write test here.
  })
})
```

0 0 Today: 3 hrs 23 mins Ln 10, Col 1 Spaces: 2 UTF-8 LF JavaScript Prettier: ✓

USEFUL MATCHERS

- expect(1+1).toEqual(2)
- expect([‘apple’, ‘banana’, ‘orange’]).toHaveLength(3)
- expect(onClick).toHaveBeenCalledTimes(1)

More matcher - https://jestjs.io/docs/en/expect

TEST RUNNER

- Jest - https://jestjs.io/
- Mocha - https://mochajs.org/

CODING TIME

WRITE TEST FOR CALCULATOR MODULE

TIME TO CODING

- Workshop repository - https://github.com/aofleejay/testing-in-javascript
- Write test for function sum inside /calculator directory.

WATCH MODE

- Run yarn test --watch or edit script in package.json to yarn --watch

TDD

TEST DRIVEN DEVELOPMENT

ABSOLUTE CALCULATOR

- New requirement “absolute input before summation”.
- Use absolute with command Math.abs(-1).
- Use TDD approach!!!.

TODO TESTCASE

- it.todo('2 - 1 should equal 1.')

TEST COVERAGE

- Run yarn test --coverage to generate coverage report.
- Open /coverage/lcov-report/index.html

DATA DRIVEN TESTING

- Use Jest.each to remove repetitive code.

Blog - มาลอง Data Driven Testing ใน Jest กันหน่อย

TESTING IN REACT

ENZYME

- Testing utilities for React
- Documentation - https://airbnb.io/enzyme/

TEST RENDERER

- Use shallow to render component
- const wrapper = shallow(<MyComponent />)
- Wrapper contains a lot of useful functions to test React component.

ENZYME WRAPPER FUNCTIONS

- find('#id'), find('.class-name'), find('element')
- find('button').simulate('click')
- find('p').text()

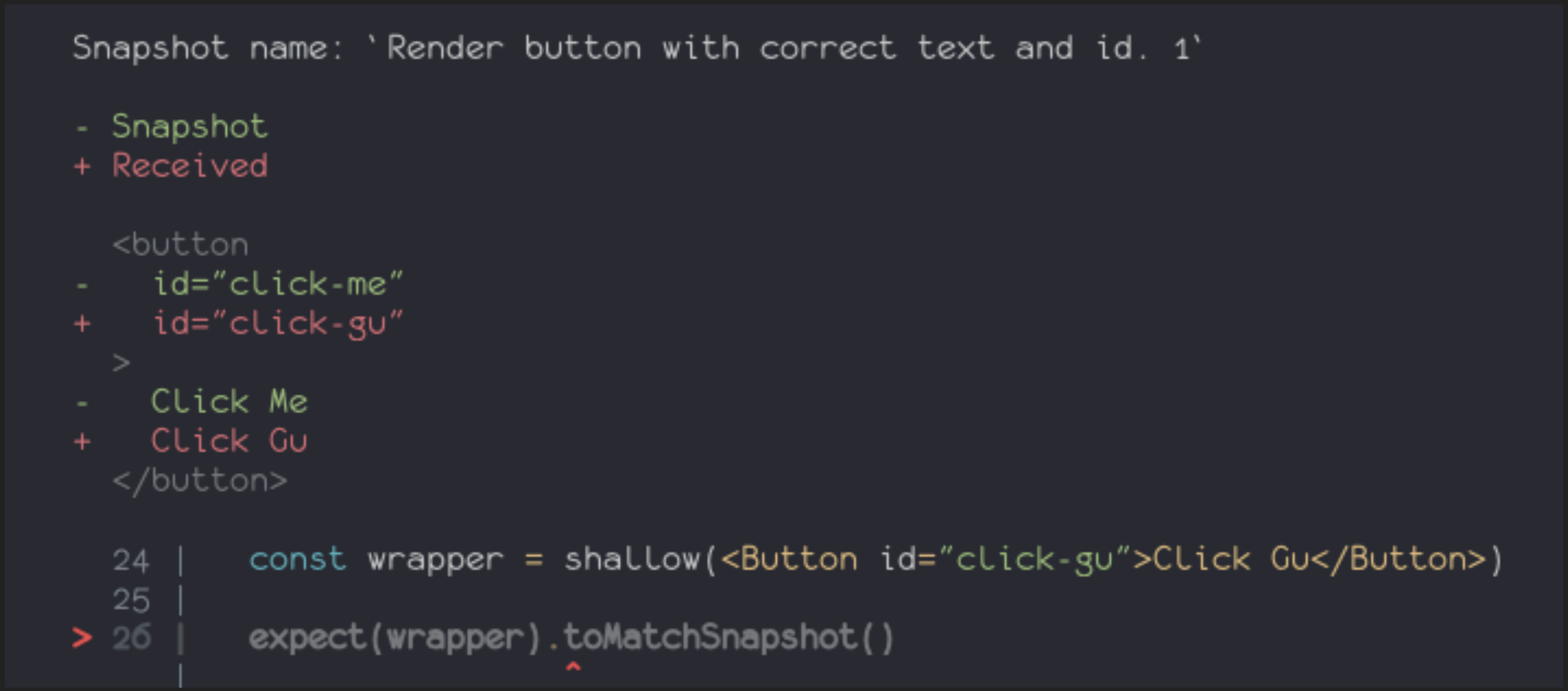
SPY FUNCTION

- Use jest.fn() to record function behavior.
- So we can use assertion like this expect(spyOnClick).toHaveBeedCalled()

TIME TO CODING AGAIN

- Write test inside /counter and /twitter directory.

SNAPSHOT TESTING



```
Snapshot name: `Render button with correct text and id. 1`

- Snapshot
+ Received

  <button
-   id="click-me"
+   id="click-gu"
  >
-   Click Me
+   Click Gu
  </button>

  24 |   const wrapper = shallow(<Button id="click-gu">Click Gu</Button>)
  25 |
> 26 |   expect(wrapper).toMatchSnapshot()
     |                  ^
```

USAGE

- To make enzyme wrapper compatible with Jest snapshot use enzyme-to-json
- Contain two API .toMatchSnapshot() and .toMatchInlineSnapshot()

CAUTION

- Treat snapshot as code!!
  - Keep snapshot small using no-large-snapshots rules in eslint-plugin-jest
- Avoid to use on fragile component.
- Prefer .toMatchInlineSnapshot() (Require prettier.)
- It’s not a panacea. Find element, normal assertion .etc maybe better.

EXPRESS

BACKEND TESTING

MEDIUM API

- Get all post using - /posts
- Get post by id - /post/001

TRY DIFFERENT TOOLS

- Use Mocha instead of Jest.
- Use Sinon instead of Jest mock.
- Use Chai instead of Jest matchers.
- Use Supertest to simulate HTTP request.
- (Optional) Use Istanbul instead of Jest coverage.