Opgave: Lave Unit Tests med Jest

gode link

Jest(https://jestjs.io/)

Mål

Lær at skrive og køre unit tests med Jest ved at teste en funktion, der udfører grundlæggende matematiske operationer.

Opgave 1

Trin 1: Lav en funktion (calculator.js)

Opret en fil kaldet `calculator.js`. Her vil du definere en add-funktion og en subtract-funktion, som du senere vil teste.

```
function add(val, val) {}

function subtract(val, val) {}

module.exports = { add, subtract };
```

Trin 2: Lav testfilen (calculator.test.js)

Opret en testfil kaldet `calculator.test.js`. I denne fil vil du skrive tests til add- og subtract-funktionerne.

```
const { add, subtract } = require("./calculator");

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

    test("subtract should return the difference between two numbers", () => {
        expect(subtract(3, 2)).toBe(1);
        expect(subtract(-1, -2)).toBe(1);
        expect(subtract(1, -1)).toBe(2);
    });
});
```

Opret på samme måde funktionerne for divider

De nye funktioner skal skrives i filen `calculator.js` og test skal skives i filen `calculator.test.js`

Opgave 2

Opret endnu en file med nedenstående simple validering af e-mail

```
function validateEmail(email) {
    return typeof email === "string" && email.includes("@") &&
email.includes(".");
}
module.exports = { validateEmail };```
```

Opret en test fill til ovenstående validering `validateEmail.test.js` og skrive mindst 2 test til funktionen. Husk at impotere `const { isValidEmail } = require("./validateEmail");`

Opgave 3

Se om i kan finde kode fra tidligere som i kan skrive test for

Opgave 4

Valider at input til regne funtionerne i `calculator.js` er et tal og hvis ikke så opret en `new error`

Du kan fx. bruge funktionerne `parseFloat(value)` og `Number.isNaN(val)`

throw error