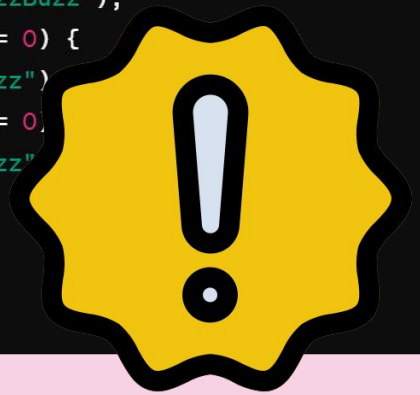
CODE LOOKING LIKE THIS?

```
for (let i = 1; i <= 100; i++)
{
  if (i % 3 === 0 && i % 5 === 0)
  {
      console.log(
      "FizzBuzz"
      )
  }
  else
  {
      if (i % 3 === 0)
      {
      console.log("Fizz")
      }
      else if (i % 5 === 0)
      {
      console.log("Buzz")}
      else
      {
          console.log(i)}
      }
}
```

10 'Format Document's LATER

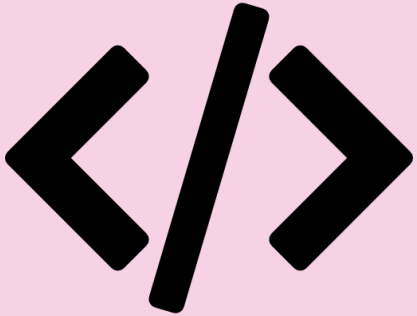
MERGE CONFLICT!

```
        console.log("FizzBuzz");
    } else if (i % 3 === 0) {
        console.log("Fizz")
    } else if (i % 5 === 0)
        console.log("Buzz"
    } else {
        console.log(i);
    }
}
```



HUSKY

WHAT IS HUSKY



Git hook manager

Simple to use

WHY HUSKY

01 CLEAN UP YOUR CODE

02 EARLY ERROR DETECTION

03 FACILITATE COLLABORATION

HOW DOES HUSKY DO ALL THAT?

CODE CLEANLINESS

ESLINT

Linter to ensure code adheres to your project's guidelines

PRETTIER

Code formatter to ensure code is formatted in a standardised manner

LINT-STAGED

Customize tasks based on type of file staged

package.json

```
"scripts": {
    "pre-commit": "lint-staged",
```

```
"lint-staged": {
    "*.{ts,svelte}": [
        "eslint",
        "prettier --write .",
        "git add ."
    ]
},
```

.husky/pre-commit

```
npm run pre-commit
```

ERROR DETECTION

JEST

JavaScript testing framework

package.json

```
"scripts": {
    "test": "jest",
```

ENSURES THAT REMOTE CODE PASSES ALL TESTS!

.husky/pre-push

```
npm run test
```

FACILITATE COLLABORATION

STANDARDISED HOOKS

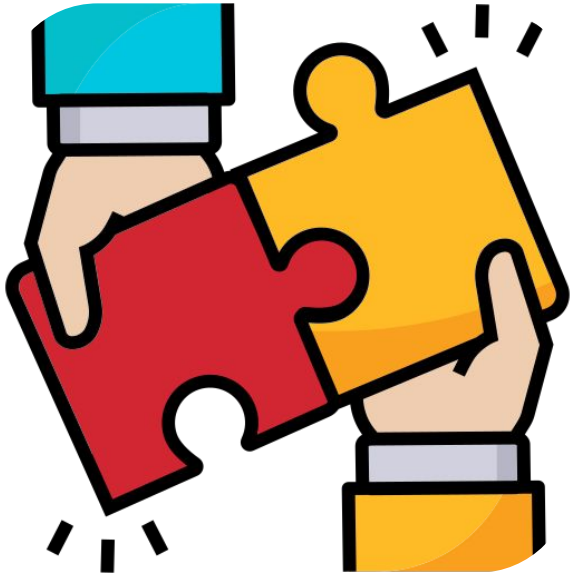
Ensures that all commits have standardised coding standards

STREAMLINE PROCESSES

Saves time, Reduces risk of human error

EASILY MANAGED

As project scales, hooks can be managed and added easily



```
npm i husky --save-dev
npx husky init
```

SAVE DEV TIME AND INSTALL HUSKY TODAY!