

# Chapter 2: ESLint for Next.js Projects

What is ESLint?

ESLint is a static code analysis tool for identifying problematic patterns found in JavaScript/TypeScript code. It helps maintain consistent code style and catches errors early in the development process.

Why Use ESLint in Next.js Projects?

- **Code Quality:** Ensures your codebase is clean and follows best practices.
- **Consistency:** Enforces coding standards and style guides across the project.
- **Error Detection:** Detects errors and potential issues before they cause problems.
- [Setting Up ESLint in a Next.js Project](#)

# Setting Up ESLint in a Next.js Project

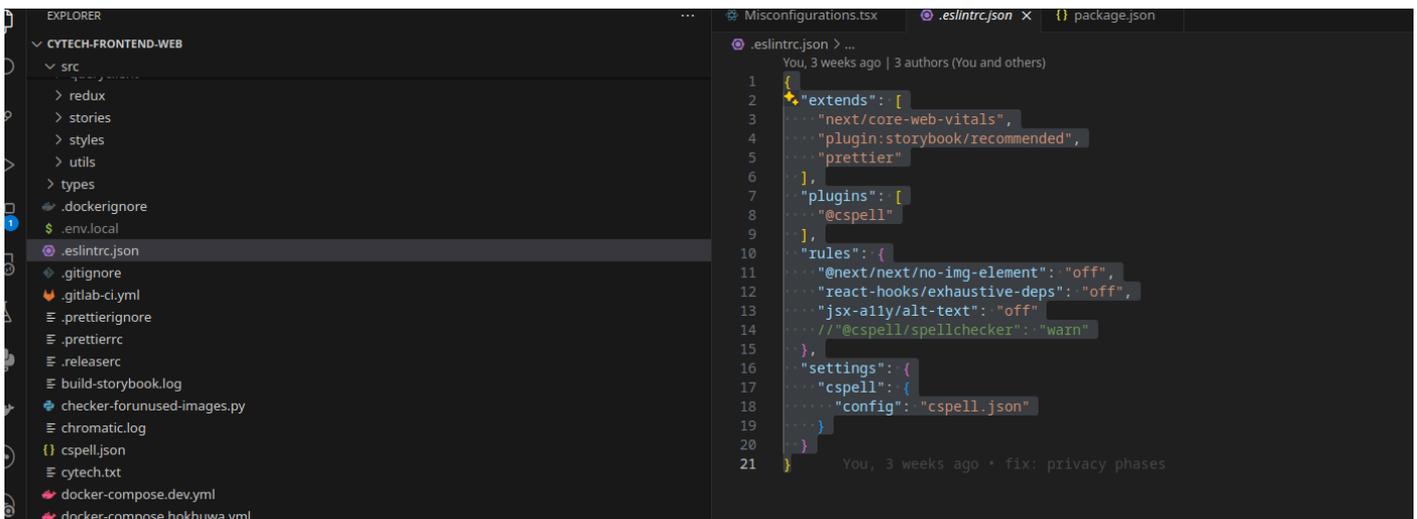
## Installation

Install ESLint along with the necessary plugins for Next.js:

```
npm install eslint eslint-config-next --save-dev
```

## Configuration

Create a configuration file `.eslintrc.json` in the root of your project:

A screenshot of a code editor showing the configuration of an ESLint project. On the left, the Explorer sidebar shows a file tree for a project named 'CYTECH-FRONTEND-WEB'. The file '.eslintrc.json' is selected and highlighted. On the right, the editor displays the content of the '.eslintrc.json' file. The configuration includes extending 'next/core-web-vitals', using 'prettier' as a plugin, and configuring the '@cspell' plugin with a custom rule for the spellchecker. The 'settings' section is also configured to use 'cspell.json' for the cspell plugin. The code is as follows:

```
1 {
2   "extends": [
3     "next/core-web-vitals",
4     "plugin:storybook/recommended",
5     "prettier"
6   ],
7   "plugins": [
8     "@cspell"
9   ],
10  "rules": {
11    "@next/next/no-img-element": "off",
12    "react-hooks/exhaustive-deps": "off",
13    "jsx-ally/alt-text": "off",
14    "@cspell/spellchecker": "warn"
15  },
16  "settings": {
17    "cspell": {
18      "config": "cspell.json"
19    }
20  }
21 }
```

## Running ESLint

```
npm run lint
```

## Recommendations for Improvement

- **Use Prettier with ESLint:** Integrate Prettier for automatic code formatting alongside ESLint.
- **Integrate with CI/CD:** Include ESLint in your CI/CD pipeline to ensure every commit meets the coding standards.
- **Custom Rules:** Define and enforce custom rules that cater to the specific needs of your project.
- **Developer Training:** Ensure all developers are familiar with the ESLint rules and configuration to maintain consistency.