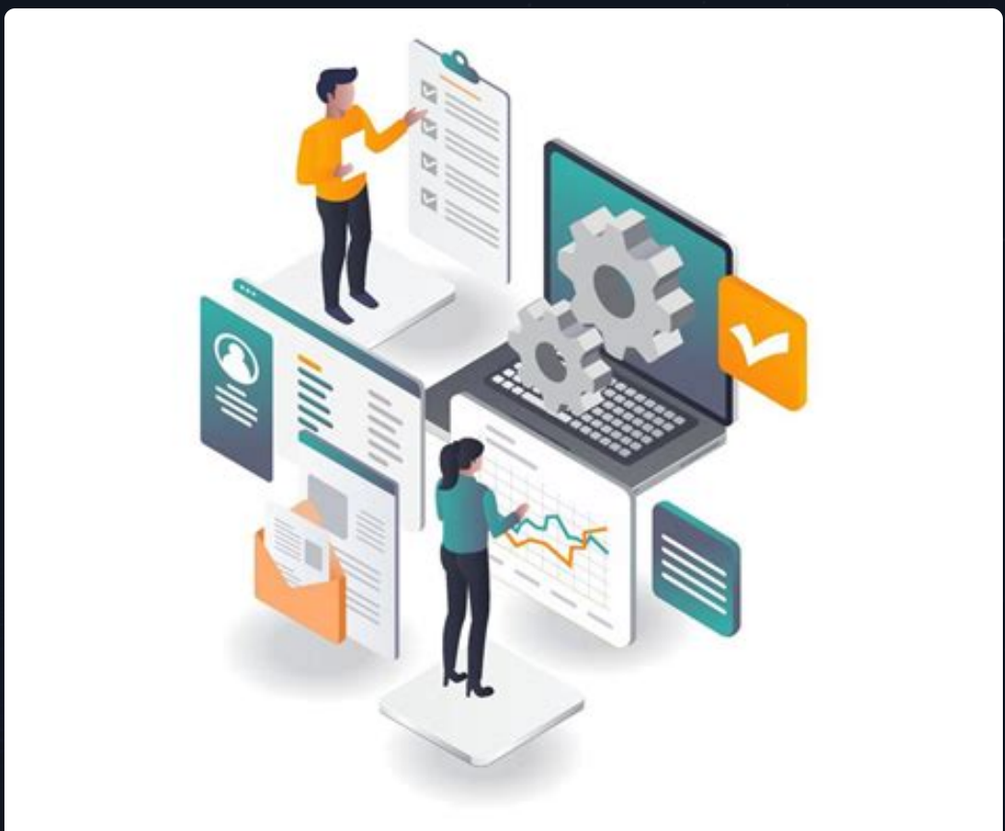
Pre-commit Hooks та Husky

Pre-commit hooks — це потужний механізм автоматизації, що дозволяє виконувати певні операції або скрипти перед збереженням змін у системі контролю версій, таких як Git. Вони забезпечують автоматичний запуск перевірок перед виконанням коміту, гарантуючи відповідність коду встановленим критеріям якості та стандартам команди.

Інструменти та можливості pre-commit hooks

Інструменти для налаштування

- **Husky**: Популярний пакет для Node.js, що спрощує налаштування хуків через package.json
- **git-hooks**: Вбудована можливість Git для створення власних сценаріїв на Unix



Ключові функції

- **Форматування коду**: Автоматичне застосування Prettier або ESLint
- **Виконання тестів**: Перевірка цілісності функціоналу перед комітом
- **Перевірка стилю**: Контроль відповідності правилам та уникнення синтаксичних помилок

Використання pre-commit hooks спрощує робочий процес команди, забезпечуючи консистентність та високу якість коду в репозиторії.

Встановлення Husky

01

Встановлення залежності

Перший крок — додати Husky як dev-залежність до вашого проєкту

02

Додавання скрипту prepare

Налаштуйте автоматичну ініціалізацію Husky через package.json

03

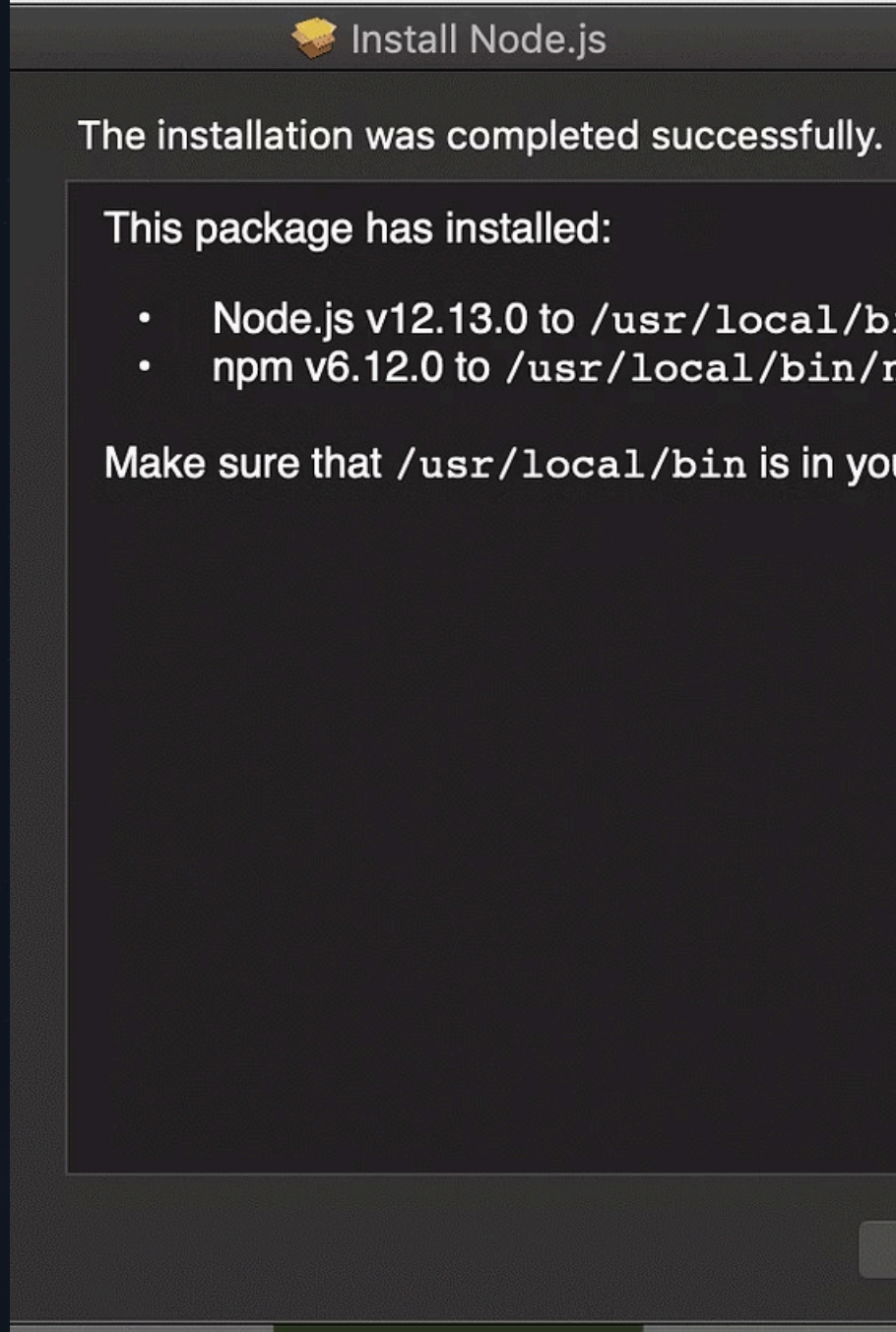
Запуск підготовки

Виконайте скрипт prepare для створення необхідної структури

04

Створення папки хуків

Husky створить папку .husky для зберігання всіх ваших хуків



Ініціалізація та налаштування

Крок 1: Ініціалізація

```
npx husky init
```

Команда автоматично налаштовує базову структуру Husky у вашому проєкті

Крок 2: Додавання скрипту

```
npm pkg set scripts.prepare="husky"
```

Це додає prepare скрипт до package.json для автоматичної ініціалізації

Крок 3: Виконання

```
npm run prepare
```

Запуск скрипту створює необхідні файли та папки для роботи хуків

Після виконання цих кроків, ви готові визначити конкретні хуки, які будуть виконуватися перед комітом. У нашому прикладі ми налаштуємо запуск ESLint для автоматичної перевірки якості коду.

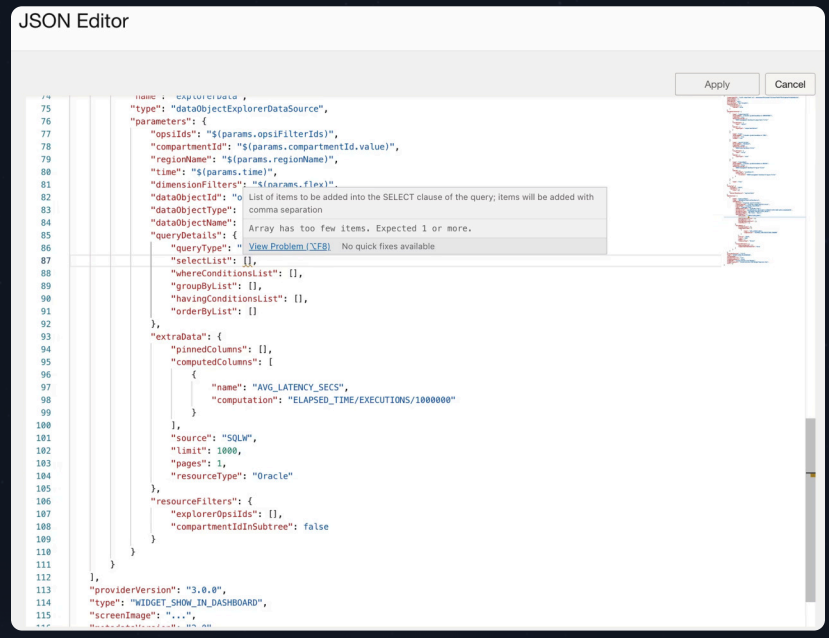
Конфігурація package.json

Структура файлу

```
{
  "devDependencies": {
    "eslint": "^8.54.0",
    "eslint-config-airbnb-base": "^15.0.0",
    "eslint-plugin-import": "^2.29.0",
    "husky": "^8.0.3",
    "prettier": "3.0.3"
  },
  "scripts": {
    "prepare": "husky install",
    "lint": "npx eslint --fix ."
  }
}
```

Ключові елементи

- **devDependencies**: Інструменти для розробки та перевірки коду
- **prepare script**: Автоматична ініціалізація Husky
- **lint script**: Команда для запуску ESLint з автовиправленням



Цей файл містить всі необхідні залежності та скрипти для роботи системи автоматизованої перевірки коду.

Створення pre-commit хука

1

Команда для створення хука

```
npx husky add .husky/pre-commit "npm run lint"
```

Ця команда створює файл pre-commit у папці .husky, який запускатиме lint скрипт

2

Автоматична перевірка

Тепер при кожному коміті автоматично запускається ESLint для перевірки коду

3

Блокування проблемного коду

Якщо виявлені помилки, коміт буде заблоковано до їх виправлення

Результат: Така конфігурація гарантує, що всі учасники команди дотримуються єдиних правил написання коду. Коміт можливий лише після успішного проходження всіх перевірок, що забезпечує високу якість кодової бази проєкту.