



Міністерство освіти і науки України  
Національний технічний університет України  
“Київський політехнічний інститут імені Ігоря Сікорського”  
Факультет інформатики та обчислювальної техніки  
Кафедра інформаційні систем та технологій

**Лабораторна робота №2**  
із дисципліни «Розробка програмного забезпечення на платформі Node.JS»  
Тема: «Налаштування проекту та інструментів розробки»

Виконав:  
Студент групи ІА-24  
Боднар А.Д.

Київ-2024

## Завдання:

1. Створити пакети
2. Вибрати стиль коду
3. Налаштувати форматтер
4. Налаштувати лінтер або інший статичний аналізатор
5. Налаштувати Git-hook на комміт та пуш. Перевіряти форматування, лінтер, тести(для початку завжди успішна команда), збірку\компіляцію проекту (фронт\TS\...).

## Хід роботи

### Пакети

```
C:\Styding\Node\ocd>npm install express

added 68 packages, and audited 84 packages in 3s

17 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\Styding\Node\ocd>npm install --save-dev nodemon

added 29 packages, and audited 113 packages in 2s

21 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\Styding\Node\ocd>npm install dotenv

up to date, audited 113 packages in 1s

21 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

## Встановлюю пакети

```
"scripts": {  
  "start": "node index.js",  
  "dev": "nodemon index.js"  
}
```

У scripts додаю два рядки для зручного запуску додатку:

start — запускає додаток через node index.js.

dev — запускає додаток через nodemon index.js, що автоматично перезавантажує сервер при зміні файлів.

## Стилі

```
C:\Styding\Node\ocd>npm install --save-dev prettier  
  
added 1 package, and audited 16 packages in 2s  
  
3 packages are looking for funding  
run `npm fund` for details  
  
found 0 vulnerabilities  
  
C:\Styding\Node\ocd>npx prettier --write .  
package-lock.json 51ms (unchanged)  
package.json 3ms (unchanged)  
testDbConnection.js 36ms
```

```
PS C:\Styding\Node\ocd> npm install --save-dev eslint-config-airbnb eslint eslint-plugin-import eslint-plugin-jsx-a11y eslint-plugin-react eslint-plugin-react-hooks  
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests  
by a key value, which is much more comprehensive and powerful.  
npm warn deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported  
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported  
npm warn deprecated @humanwhocodes/object-schema@2.0.3: Use @eslint/object-schema instead  
npm warn deprecated @humanwhocodes/config-array@0.13.0: Use @eslint/config-array instead  
npm warn deprecated eslint@8.57.1: This version is no longer supported. Please see https://eslint.org/version-support for other options.  
  
added 223 packages, and audited 336 packages in 13s  
  
115 packages are looking for funding  
run `npm fund` for details  
  
found 0 vulnerabilities
```

```
PS C:\Styding\Node\ocd> npm outdated
```

<u>Package</u>	<u>Current</u>	<u>Wanted</u>	<u>Latest</u>	<u>Location</u>	<u>Depended by</u>
eslint	8.57.1	8.57.1	9.17.0	node_modules/eslint	ocd
eslint-plugin-react-hooks	4.6.2	4.6.2	5.1.0	node_modules/eslint-plugin-react-hooks	ocd

```
PS C:\Styding\Node\ocd> npm install eslint @latest
```

```
up to date, audited 336 packages in 1s
```

```
115 packages are looking for funding
```

```
run `npm fund` for details
```

```
found 0 vulnerabilities
```

```
PS C:\Styding\Node\ocd> npm install eslint-plugin-react-hooks @latest
```

```
up to date, audited 336 packages in 2s
```

```
115 packages are looking for funding
```

```
run `npm fund` for details
```

```
found 0 vulnerabilities
```

```
PS C:\Styding\Node\ocd> npx eslint --init
```

```
You can also run this command directly using 'npm init @eslint/config'.
```

```
Need to install the following packages:
```

```
@eslint/create-config@1.4.0
```

```
Ok to proceed? (y) y
```

```
> ocd@1.0.0 npx
```

```
> create-config
```

```
@eslint/create-config: v1.4.0
```

```
@eslint/create-config: v1.4.0
```

```
✓ How would you like to use ESLint? · problems
```

```
✓ What type of modules does your project use? · esm
```

```
✓ Which framework does your project use? · none
```

```
✓ Does your project use TypeScript? · javascript
```

```
✓ Where does your code run? · node
```

```
The config that you've selected requires the following dependencies:
```

```
eslint, globals, @eslint/js
```

```
✓ Would you like to install them now? · No / Yes
```

```
✓ Which package manager do you want to use? · npm
```

```
● Installing...
```

```
up to date, audited 339 packages in 1s
```

```
116 packages are looking for funding
```

```
run `npm fund` for details
```

```
found 0 vulnerabilities
```

```
Successfully created C:\Styding\Node\ocd\eslint.config.mjs file.
```

```
□
```

Так, тепер все правильно налаштовано для проекту. Ось що було зроблено:

Вибір налаштувань ESLint:

Стиль коду: Проблеми з кодом.

Модулі: ESM (ECMAScript Modules).

Фреймворк: Немає (None).

Тип коду: JavaScript (без TypeScript).

Місце виконання: Node.js.

Установка необхідних залежностей: ESLint встановив необхідні пакети:

eslint — основний пакет для перевірки коду.

globals — глобальні змінні для середовища Node.js.

@eslint/js — правила для стандартного JavaScript.

Створено конфігураційний файл ESLint: eslint.config.mjs, який містить налаштування для перевірки стилю коду в проекті.

```
PS C:\Styding\Node\ocd> npx eslint
PS C:\Styding\Node\ocd> 
```

Все працює успішно

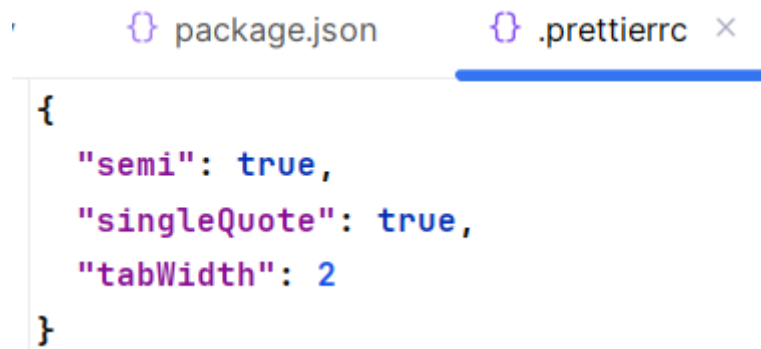
## Форматтер

```
PS C:\Styding\Node\ocd> npm install --save-dev prettier
```

```
up to date, audited 339 packages in 1s
```

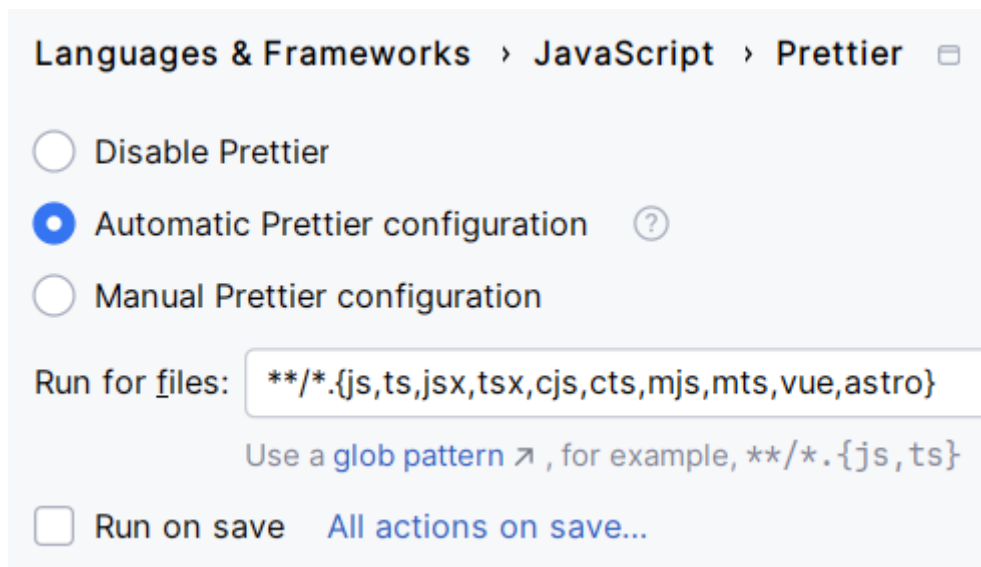
```
116 packages are looking for funding
  run `npm fund` for details
```

```
found 0 vulnerabilities
```



The image shows a code editor window with two tabs: 'package.json' and '.prettierrc'. The '.prettierrc' tab is active and displays the following JSON configuration:

```
{
  "semi": true,
  "singleQuote": true,
  "tabWidth": 2
}
```



Prettier успішно встановлений

# Лінтер

```
"scripts": {  
  "start": "node index.js",  
  "dev": "nodemon index.js",  
  "format": "prettier --write .",  
  "lint": "eslint ."  
},
```

Додаю "lint": "eslint ."

```
PS C:\Styding\Node\ocd> npm run lint
```

```
> ocd@1.0.0 lint  
> eslint .
```

```
PS C:\Styding\Node\ocd>
```

---

```
PS C:\Styding\Node\ocd> npm run lint
```

```
> ocd@1.0.0 lint  
> eslint .
```

```
C:\Styding\Node\ocd\testDbConnection.js
```

```
25:6 error Parsing error: Unexpected token to
```

```
✖ 1 problem (1 error, 0 warnings)
```

```
PS C:\Styding\Node\ocd> █
```

ESLint я був скачав ще коли стиль коду обирав,  
вище в звіті

Лінтер працює чудово

# Хуки

```
PS C:\Styding\Node\ocd> npm install --save-dev husky
```

```
added 1 package, and audited 340 packages in 2s
```

```
117 packages are looking for funding
  run `npm fund` for details
```

```
found 0 vulnerabilities
```

```
PS C:\Styding\Node\ocd> npx husky-init
```

```
Need to install the following packages:
```

```
husky-init@8.0.0
```

```
Ok to proceed? (y) y
```

```
husky-init updating package.json
```

```
"husky install" command already exists in prepare script, skipping.
```

```
husky - Git hooks installed
```

```
husky - created .husky/pre-commit
```

```
please review changes in package.json
```

Прописую команды

```
▼ .husky
  pre-commit
  pre-push
```

```
.husky\pre-commit x pre-push t
1 ▶ #!/usr/bin/env sh
2   . "$(dirname -- "$0")/_/husky.sh"
3
4   npm run format && npm run lint
```



```
.husky\pre-commit  pre-push x |
1  > #!/bin/sh
2  . "$(dirname "$0")/_/husky.sh"
3
4  npm test && npm run build
```

Створюю файли та прописую їх

```
PS C:\Styding\Node\ocd> git commit -m "Тест pre-commit хука"
```

```
> ocd@1.0.0 format
```

```
> prettier --write .
```

```
.prettierrc 64ms
```

```
eslint.config.mjs 13ms
```

```
package-lock.json 115ms (unchanged)
```

```
package.json 1ms (unchanged)
```

```
README.md 63ms
```

```
testDbConnection.js 16ms (unchanged)
```

```
> ocd@1.0.0 lint
```

```
> eslint .
```

```
C:\Styding\Node\ocd\testDbConnection.js
```

```
25:1  error  'авжооо' is not defined  no-undef
```

```
✖ 1 problem (1 error, 0 warnings)
```

```
husky - pre-commit hook exited with code 1 (error)
```

```
PS C:\Styding\Node\ocd> █
```

Пре-коміт хук видав помилку(яку я спеціально залишив) отже він працює правильно(З другої спроби звісно, все можна буде побачити по історії комітів)

```
PS C:\Styding\Node\ocd> git commit -m "L2 commit1maybe"
```

```
> ocd@1.0.0 format
```

```
> prettier --write .
```

```
.prettierrc 66ms (unchanged)
```

```
eslint.config.mjs 12ms (unchanged)
```

```
package-lock.json 107ms (unchanged)
```

```
package.json 2ms (unchanged)
```

```
README.md 62ms (unchanged)
```

```
testDbConnection.js 16ms (unchanged)
```

```
> ocd@1.0.0 lint
```

```
> eslint .
```

```
[main 7133ad8] L2 commit1maybe
```

```
22 files changed, 254 insertions(+), 123 deletions(-)
```

```
create mode 100644 .husky/pre-commit
```

```
create mode 100644 .husky/pre-push
```

```
delete mode 100644 node_modules/husky/bin.js
```

```
delete mode 100644 node_modules/husky/husky
```

```
create mode 100644 node_modules/husky/husky.sh
```

```
delete mode 100644 node_modules/husky/index.d.ts
```

```
delete mode 100644 node_modules/husky/index.js
```

```
create mode 100644 node_modules/husky/lib/bin.d.ts
```

```
create mode 100644 node_modules/husky/lib/bin.js
```

```
create mode 100644 node_modules/husky/lib/index.d.ts
```

```
create mode 100644 node_modules/husky/lib/index.js
```

```
create mode 100644 pre-commit
```

І тепер випадок коли пре-коміт спрацював як мав при коді без помилок

```
PS C:\Styding\Node\ocd> git push origin main

> ocd@1.0.0 test
> echo "No tests specified" && exit 0

"No tests specified"

> ocd@1.0.0 build
> echo "No build step specified" && exit 0

"No build step specified"
Enumerating objects: 38, done.
Counting objects: 100% (38/38), done.
Delta compression using up to 8 threads
Compressing objects: 100% (24/24), done.
Writing objects: 100% (25/25), 5.03 KiB | 859.00 KiB/s, done.
Total 25 (delta 10), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (10/10), completed with 9 local objects.
To https://github.com/BAntonD/Node-OCD
    b429e9b..7133ad8  main -> main
PS C:\Styding\Node\ocd>
```

Як і пре-пуш

## Висновки

Виконуючи цю лабораторну роботу, я ознайомився з процесом налаштування середовища розробки, зокрема:

### 1. Створення пакетів

Я розбив проєкт на окремі модулі, що дозволило краще структурувати код і зробити його більш підтримуваним та зрозумілим.

### 2. Вибір стилю коду

Був обраний стиль написання коду, який відповідає сучасним стандартам та забезпечує його читабельність і зрозумілість для інших розробників.

### 3. Налаштування форматтера

Було інтегровано Prettier для автоматичного форматування коду. Це дозволяє підтримувати єдиний стиль написання коду незалежно від уподобань розробника.

### 4. Налаштування лінтера

Я налаштував ESLint для проведення статичного аналізу коду, що дає змогу виявляти потенційні помилки та проблеми ще до виконання коду.

### 5. Налаштування Git-хуків

За допомогою Husky було створено Git-хуки для автоматизації процесів перевірки коду:

- **pre-commit**: перевіряє форматування та код за допомогою Prettier та ESLint.
- **pre-push**: виконує тестування та збірку/компіляцію проєкту, щоб запобігти пушу некоректного коду.

В результаті роботи я навчився налаштовувати інструменти для забезпечення високої якості коду, автоматизації рутинних перевірок і підвищення ефективності командної роботи. Налаштовані процеси дозволяють мінімізувати ризик потрапляння помилок у кодовій базі, що є важливим аспектом професійної розробки.