

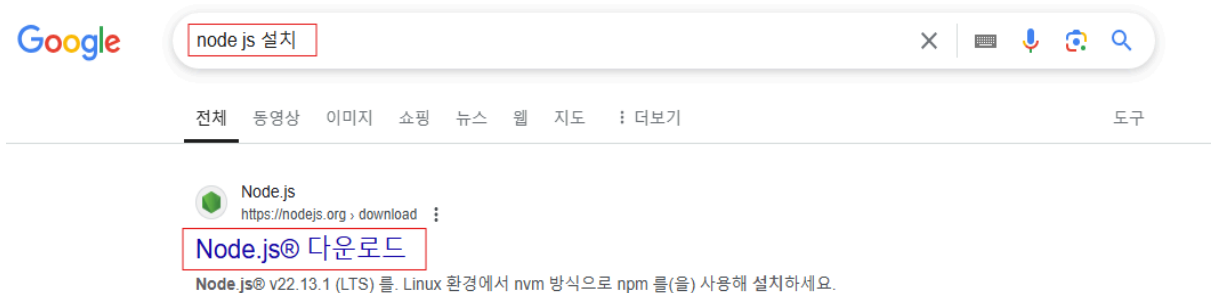


React 개발환경 구축 메뉴얼

Node.js란?

- Node.js는 JavaScript를 웹브라우저 밖에서도 실행할 수 있도록 만든 런타임 환경입니다.
- 즉, JavaScript로 서버를 구축하거나, 데스크톱 앱 개발, 그리고 리액트와 같은 프론트엔드 프레임워크의 개발 환경을 구축할 때 필수적입니다.

구글에서 "Node.js 설치" 검색 후 공식 홈페이지로 접속합니다.



LTS 버전의 **Windows 설치 프로그램(msi)** 파일을 클릭하여 다운로드합니다.

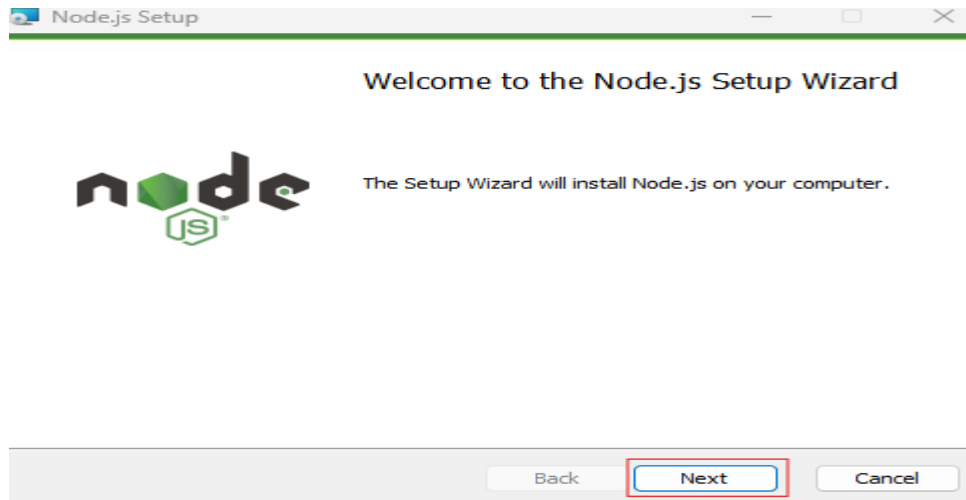


다운로드된 Node.js 설치 파일입니다. 더블 클릭하여 실행합니다.

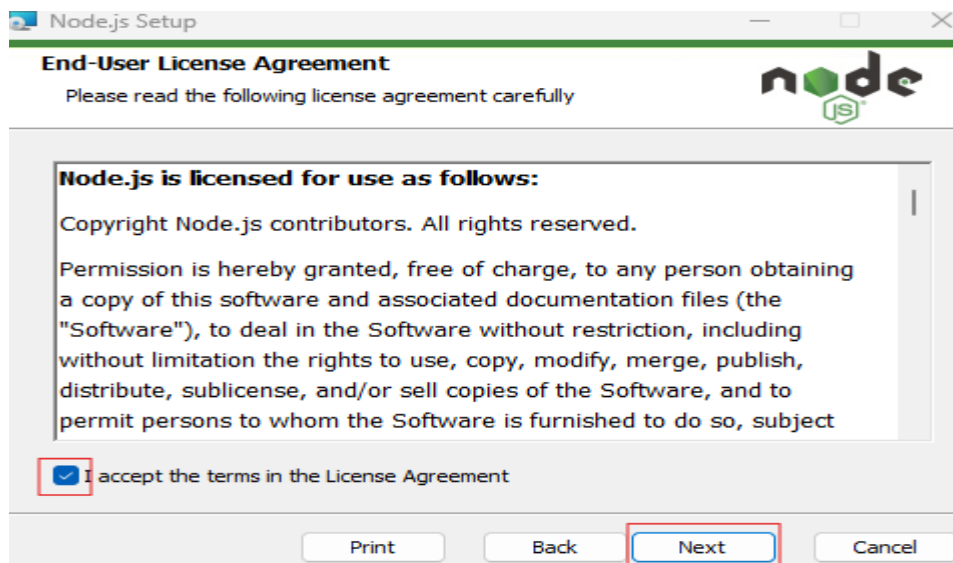


node-v22.1
3.1-x64

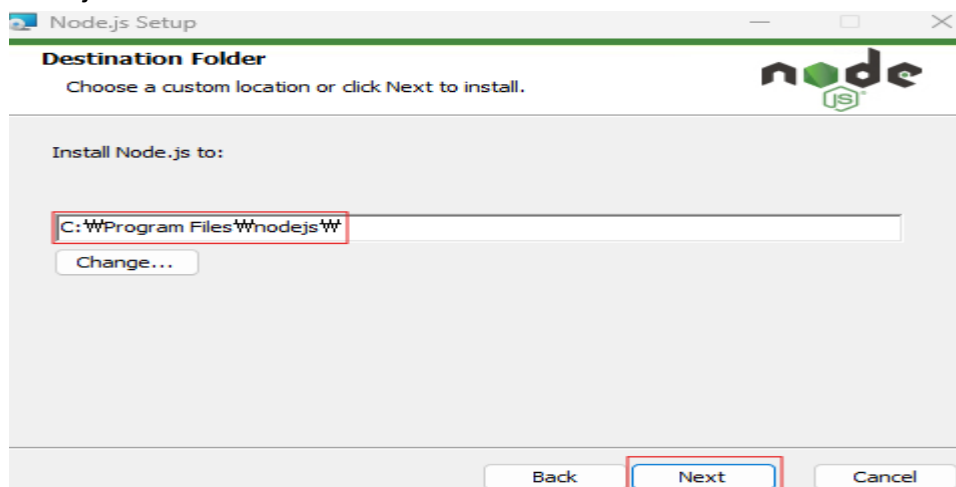
설치 마법사 시작 화면입니다. **Next** 클릭합니다.



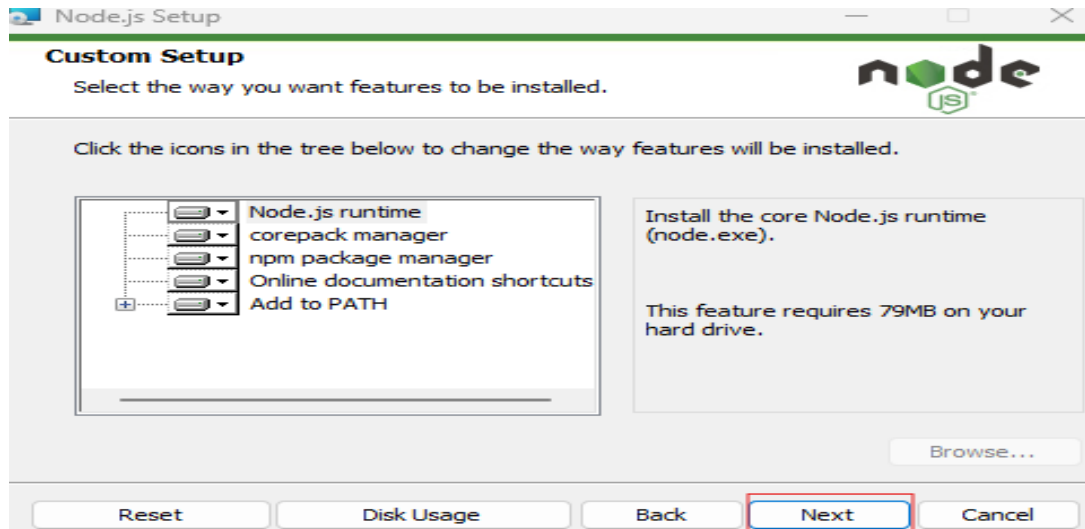
약관 동의 체크 후 **Next** 클릭합니다.



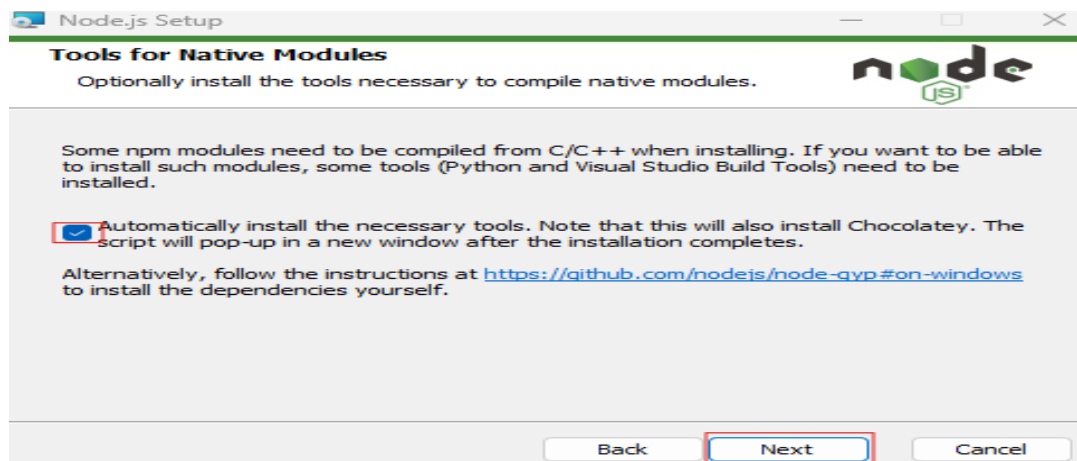
Node.js 설치 경로 지정 화면입니다. 기본 경로 유지 후 **Next** 클릭합니다.



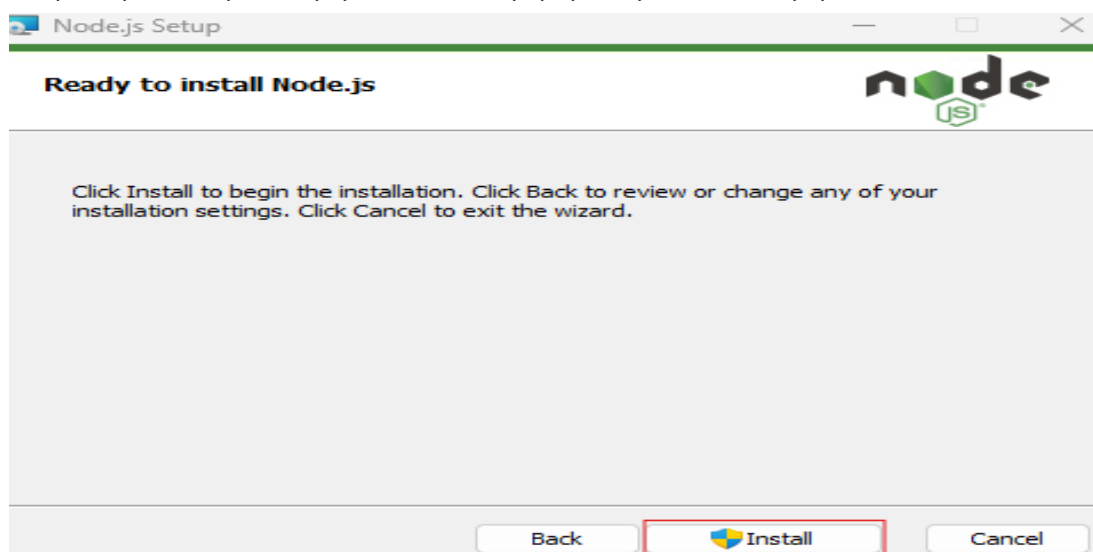
설치 구성 옵션 화면입니다. 기본 옵션 유지 후 **Next** 클릭합니다.



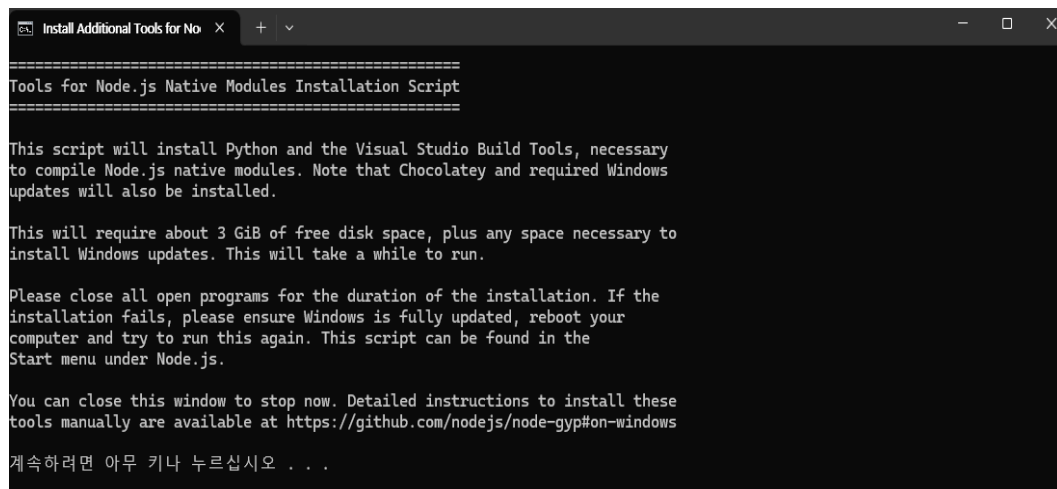
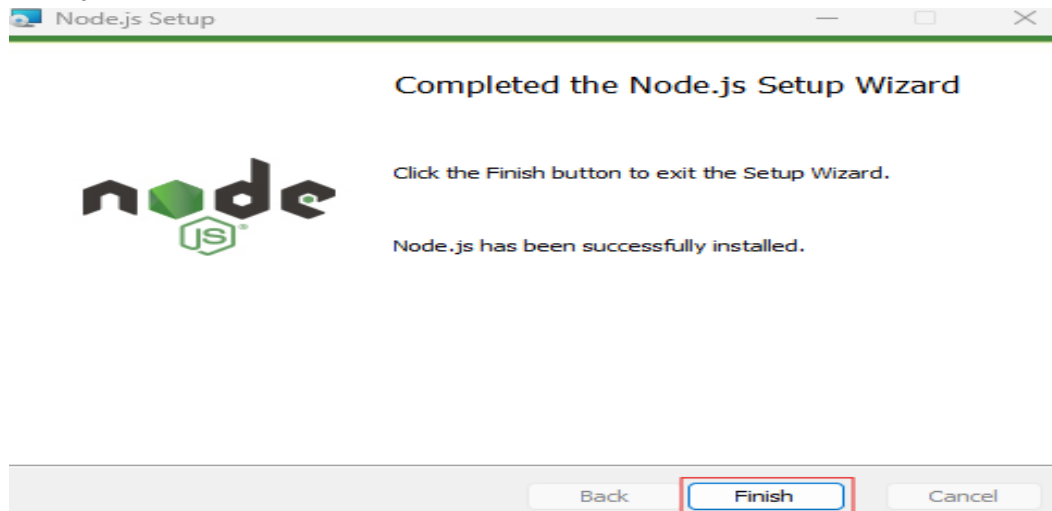
네이티브 모듈 추가 설치 옵션입니다. 체크한 상태로 두고 **Next** 클릭합니다.



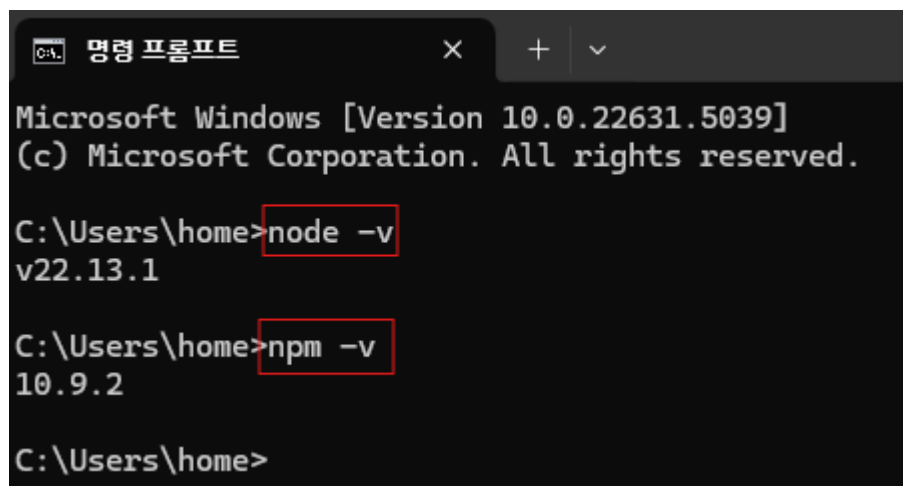
설치 준비 완료 화면입니다. **Install** 클릭하여 설치를 진행합니다.



Node.js 설치 완료 화면입니다. **Finish** 클릭하면 설치가 완료됩니다.



- 터미널에서 설치된 node.js 및 npm 버전 확인





React 프로젝트 생성 메뉴얼

npm이란?

- Node Package Manager의 약자로, 자바스크립트 관련 패키지(라이브러리)를 설치하고 관리해주는 도구입니다.
 - npm을 통해 리액트, 뷰, 앵귤러 등 수많은 자바스크립트 패키지를 편리하게 설치하고 관리할 수 있습니다.
-
- node js를 설치하면 npm도 설치가 되어있습니다.
 - 개발 중 발생하는 에러의 원인중 Node 버전 또는 npm 버전 문제일 수 있으므로, 버전을 확인하는 습관이 중요합니다.

✅ 요약

- 1) 터미널을 열고 프로젝트를 만들고 싶은 폴더에서 다음 명령어 입력합니다.

여기서 **my-react-app** 부분에 본인의 프로젝트 이름을 입력하면 됩니다.

```
npm create vite@latest my-react-app -- --template react
```

- 2) 이후 프로젝트 폴더로 이동 후 의존성을 설치합니다.

```
cd my-react-app
```

```
npm install
```

- 3) 설치가 끝나면 프로젝트를 실행합니다.

```
npm run dev
```

✓ 상세

Vite란 무엇인가?

- Vite(비트)는 현대적인 웹 프로젝트를 빠르고 쉽게 생성할 수 있게 도와주는 빌드 도구입니다.
- 예전에는 주로 Webpack이라는 빌드 도구를 사용했으나, 최근에는 Webpack보다 훨씬 빠른 Vite를 더 많이 사용합니다.

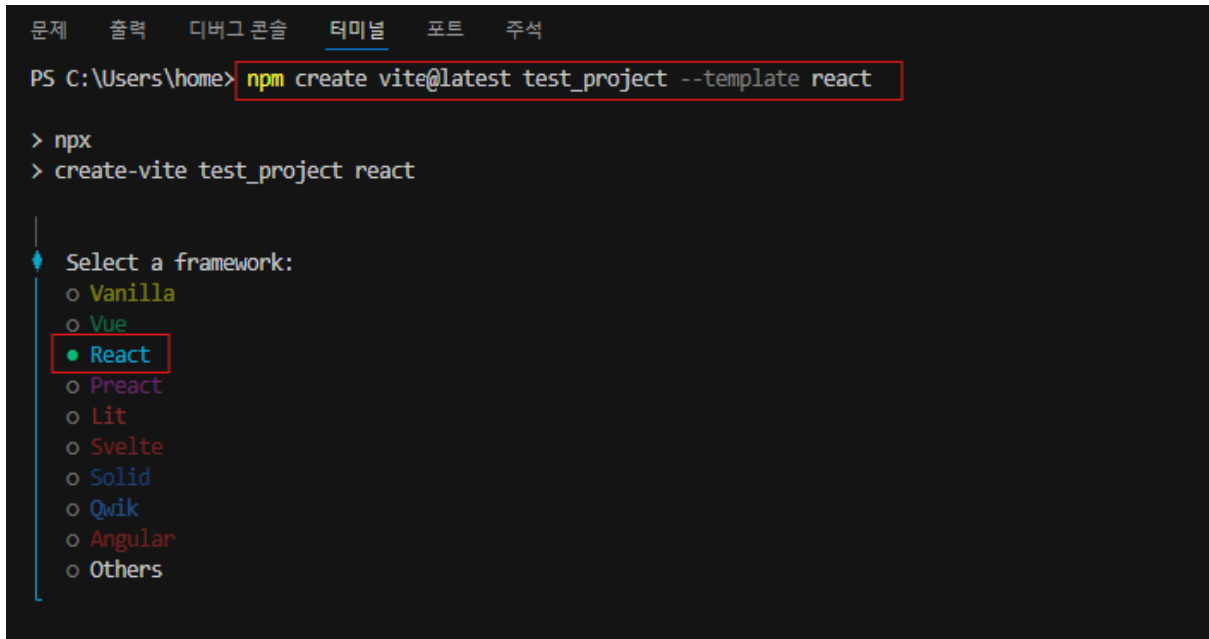
왜 Vite를 사용하는가? (Webpack 대신)

- 프로젝트 빌드 및 실행 속도가 Webpack보다 압도적으로 빠르기 때문입니다.
- 빠른 빌드 속도로 인해 개발자의 생산성을 높여줍니다.
- 기본 설정이 간단하고, 최신 JavaScript 기능을 별도 설정 없이 바로 사용할 수 있습니다.

1) 터미널을 열고 프로젝트를 만들고 싶은 폴더에서 다음 명령어 입력합니다.

```
npm create vite@latest my-react-app -- --template react
```

여기서 `my-react-app` 부분에 본인의 프로젝트 이름을 입력하면 됩니다.



```
문제 출력 디버그 콘솔 터미널 포트 주석
PS C:\Users\home> npm create vite@latest test_project --template react

> npx
> create-vite test_project react

Select a framework:
  o Vanilla
  o Vue
  o React
  o Preact
  o Lit
  o Svelte
  o Solid
  o Qwik
  o Angular
  o Others
```

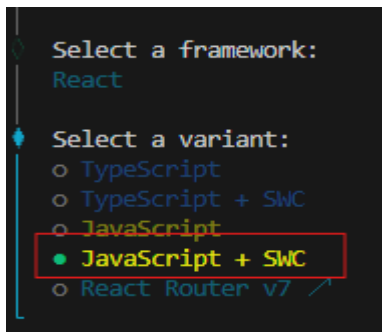
SWC란 무엇인가?

- SWC는 Rust라는 프로그래밍 언어로 만들어진 초고속 자바스크립트 컴파일러입니다.
- 자바스크립트 코드를 브라우저가 이해할 수 있게 빠르게 변환해주는 역할을 합니다. (기존에는 Babel을 주로 사용했지만, SWC가 훨씬 빠르게 동작합니다.)

왜 SWC를 사용하는가? (Babel 대신)

- SWC는 Babel보다 컴파일 속도가 몇 배 이상 빠릅니다.
- 컴파일 시간이 짧아지면 프로젝트 실행 속도와 개발 생산성이 높아집니다.
- 대규모 프로젝트일수록 Babel 대비 SWC의 성능 향상이 뚜렷하게 체감됩니다.
- 즉, **JavaScript + SWC**를 선택하는 이유는 일반적인 자바스크립트를 더 빠르게 처리하기 위함입니다.

- **JavaScript + SWC**를 선택합니다.

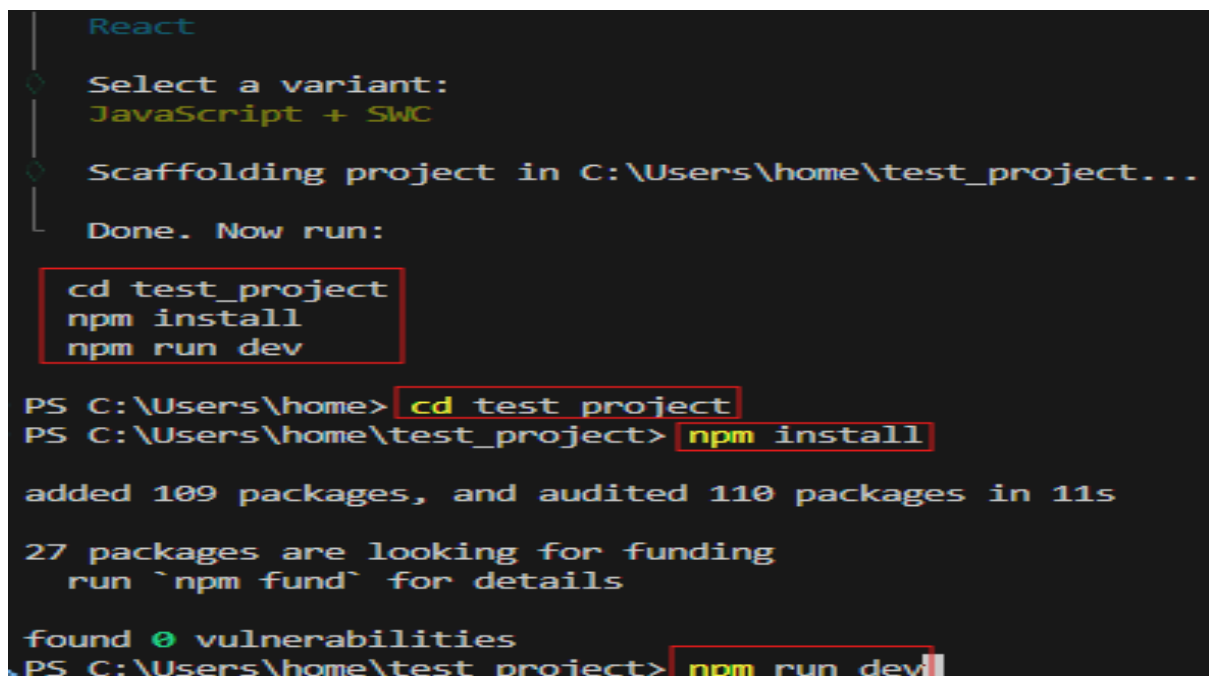


2) 이후 프로젝트 폴더로 이동 후 의존성을 설치한뒤 프로젝트를 실행합니다.

cd my-react-app

npm install

npm run dev



```
문제 출력 디버그 콘솔 터미널 포트 주석

VITE v6.2.2 ready in 335 ms

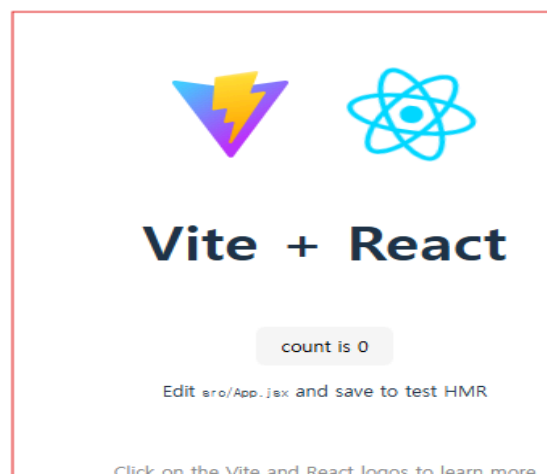
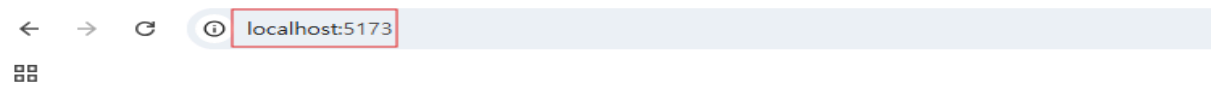
→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```

최종적으로 React + Vite가 정상적으로 설치되고 실행된 상태입니다.

- 리액트 프로젝트 환경 구축이 완료된 것이며, 이제부터 직접 앱을 개발하고 테스트할 준비가 끝났다는 뜻입니다.

화면 구성 요소의 의미

- **Vite 로고:** 프로젝트의 빌드 도구인 Vite를 사용하고 있다는 표시
- **React 로고:** 현재 사용하는 프레임워크가 React임을 의미
- **count is 0** 버튼: 리액트에서 상태(State)를 활용한 간단한 예제로, 클릭 시 숫자가 증가합니다. 리액트의 기본적인 상태관리를 이해하는 데 도움이 됩니다.



✓ npm 명령어 실행 Issue

💡 아래와 같은 이슈가 발생할 경우 다음과 같이 해결합니다.

npm : 이 시스템에서 스크립트를 실행할 수 없으므로 C:\Program Files\nodejs\npm.ps1 파일을 로드할 수 없습니다. 자세한 내용은 [about_Execution_Policies\(https://go.microsoft.com/fwlink/?LinkID=135170\)](https://go.microsoft.com/fwlink/?LinkID=135170)를 참조하십시오.

1. PowerShell을 관리자 권한으로 실행합니다.

- 시작 버튼 → 검색창에 PowerShell을 입력 → Windows PowerShell을 우클릭하여 관리자 권한으로 실행 선택

2. 다음 명령어를 입력하여 실행 정책을 변경합니다.

Set-ExecutionPolicy RemoteSigned

- 명령어 입력 후, Y(예)를 눌러서 실행 정책 변경을 확인합니다.

✓ ESLint (문법 검사기 & 코드 품질 향상 도구)

ESLint는 코드에 일관된 스타일과 규칙을 강제로 적용하는 도구입니다.

쉽게 비유하면 맞춤법 검사기나 문법 검사기와 비슷합니다.

즉, ESLint는 다음과 같은 역할을 합니다.

- 문법 에러 발견
 - 오타, 정의되지 않은 변수, 잘못된 구문 등 오류를 즉시 알려줍니다.
- 코딩 컨벤션 강제
 - 예를 들어, 변수를 정의했는데 사용하지 않은 경우 경고해줍니다.
 - 일관된 코드 스타일을 유지하도록 해줍니다.
 - 예를 들어, 작은따옴표 vs 큰따옴표, 세미콜론 사용 여부, 들여쓰기 등 규칙을 정할 수 있습니다.
- 버그 예방
 - 문제가 될 수 있는 코드 패턴을 감지하여 실수를 줄이도록 도와줍니다.

🔥 ESLint 설치

다음 명령어로 설치하세요.

```
npm install eslint --save-dev
npx eslint --init
```

- `eslint --init` 명령어를 실행하면 인터랙티브하게 질문에 답하면 설정 파일을 자동으로 생성해줍니다.

권장 설정 예시:

```
✓ How would you like to use ESLint? » To check syntax and find
problems
✓ What type of modules does your project use? » JavaScript modules
(import/export)
✓ Which framework does your project use? » React
✓ Does your project use TypeScript? » No
✓ Where does your code run? » Browser
✓ What format do you want your config file to be in? » JavaScript
```

의존성 설치를 물어보면 **Yes** 선택하면 자동으로 관련 패키지들이 설치됩니다.

✅ Prettier (자동 포매팅 도구)

Prettier는 코드를 자동으로 예쁘게 정리해주는 도구입니다.

쉽게 비유하면 **깔끔하게 정리해주는 자동 청소기** 같은 개념입니다.

즉, 다음과 같은 역할을 합니다.

- 자동 들여쓰기
- 코드 간격 조정
- 괄호의 위치 정렬
- 긴 코드 줄 자동 줄바꿈

코드를 저장할 때마다 설정된 규칙에 따라 자동으로 코드 형식을 정리해줍니다.

코드를 자동으로 일관된 규칙에 맞게 정리해주기 때문에, 개발자는 **형식을 맞추는 데 신경쓰지 않고 로직에만 집중**할 수 있습니다.

🔴 Prettier 설치

명령어 입력

```
npm install prettier eslint-config-prettier eslint-plugin-prettier
--save-dev
```

프로젝트 루트 폴더에 `.prettierrc` 파일을 만들고 아래와 같이 작성합니다.

```
{
  "semi": true,
  "singleQuote": true,
  "tabWidth": 2,
  "trailingComma": "es5"
}
```

이후 ESLint 설정파일 (`.eslintrc.cjs`)에 다음과 같이 설정을 추가합니다.

```
module.exports = {
  // ... 기존 설정

  extends: [
    'eslint:recommended',
    'plugin:react/recommended',
    'plugin:prettier/recommended' // ← 이 부분 추가
  ],
};
```

💡 ESLint와 Prettier 정리

구분	ESLint	Prettier
역할	문법 체크, 버그 예방, 코드 품질 향상	코드 스타일 자동 정리
주요 기능	규칙 위반 시 에러 또는 경고 알림	자동 포매팅 (저장할 때 자동 실행)
설정 방법	수동 규칙 설정이 많음 (커스터마이징)	설정은 간단, 대부분 자동화