

# 1월 18일 (화)

## 1. 엘리스 SW 엔지니어 트랙 - 화요일 실습

### 1교시

git 사용법 복습

json-server

lowdb

이고잉님이 생각하는 RESTful API

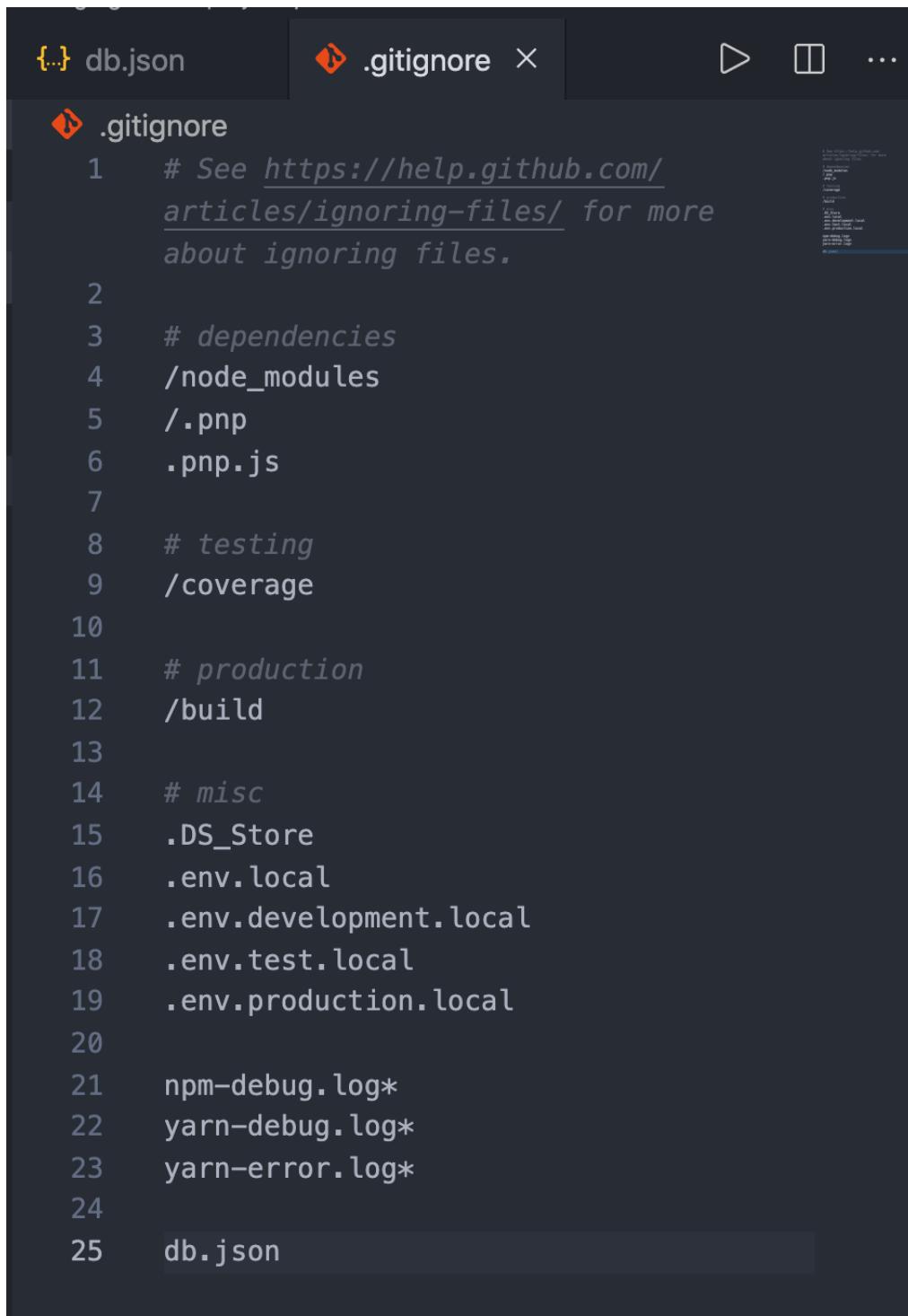
서버와 클라가 통신할 때 사용하는 베스트 프랙티스 http가 가지고 있는 기능 headers, method, 상태코드, 등 속성을 잘 활용하자 라는 것이 restful 하다라는 것

PUT vs PATCH : put은 전체를 바꾸고 patch는 부분을 바꿀 수 있다.

response의 프로퍼티 하나만 변경하고 싶다? → patch 사용

하나의 객체 데이터 모두 바꾸고 싶다 → put 사용

db.json 같은 파일은 버전관리하는게 아니다. gitignore 안에 넣어준다.



The screenshot shows a GitHub repository interface. On the left, there's a sidebar with a tree view of the repository's structure. In the main area, there are two tabs: ".gitignore" and "db.json". The ".gitignore" tab is active, displaying the following content:

```
1 # See https://help.github.com/articles/ignoring-files/ for more
2   about ignoring files.
3
4   # dependencies
5   /node_modules
6   /.pnp
7   .pnp.js
8
9   # testing
10  /coverage
11
12  # production
13  /build
14
15  # misc
16  .DS_Store
17  .env.local
18  .env.development.local
19  .env.test.local
20  .env.production.local
21
22  npm-debug.log*
23  yarn-debug.log*
24  yarn-error.log*
25
26  db.json
```

## 2교시

redux 복습

vscode 리팩터링 기능 유용함 !!

redux devtool 을 활성화 하려면 아래 처럼 설정해줘야 한다.

```
const store = createStore(reducer, window.__REDUX_DEVTOOLS_EXTENSION__ && window.__REDUX_DEVTOOLS_EXTENSION__());
```

## 3교시

**dispatch , useSelector를 사용해서 서버에서 가져온 데이터를 List로 띄우기**

### react-router-dom 사용하기

1. index.js 에서 BrowserRouter 를 import 하고 App 컴포넌트를 browserRouter 태그로 감싸준다.
2. 이동하는 a태그를 모두 Link로 바꾼다. link는 react-router-dom 에서 임포트함. ⇒ Link 태그는 페이지 이동을 안 시키고 url만 변경한다.
3. Routes, Route를 사용해서 링크 이동시 렌더링할 페이지 설정

```
App.js > App
  );
  [], []);

return (
  // (Main Effect)
  <div className="App">
    <Header />
    <Nav />
    <Article />
    <Routes>
      <Route
        path="/"
        element={
          <article>
            <h1>Welcome</h1>Hello, World
          </article>
        }
      />
      <Route
        path="/read/:id"
        element={
          <article>
            <h1>Read</h1>
          </article>
        }
      />
    </Routes>
  </div>
);
```

index에 넣어도 되고 app안에 이렇게 변경할 부분에 넣어도 된다.

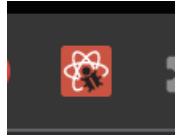
## 2. 실습

### 1교시

CI/CD : 커밋 한 번으로 테스트, 알림, 배포까지 원큐에 자동으로 하는 기능

## Production VS Development

(Development 환경)npm start로 로컬서버 페이지를 띄우면 페이지의 리액트 로고가 이렇게 바뀐다. 즉, 배포해서 사용하기에 안좋다는 뜻. 이 로고로 production인지 development인지 확인 가능하다.



/elices/ 엘리스 SW 엔지니어 트랙 13주차 실습

## Development VS Production

### Development

디버깅 및 개발 환경에 초점

### Production

실제 배포 환경에 초점

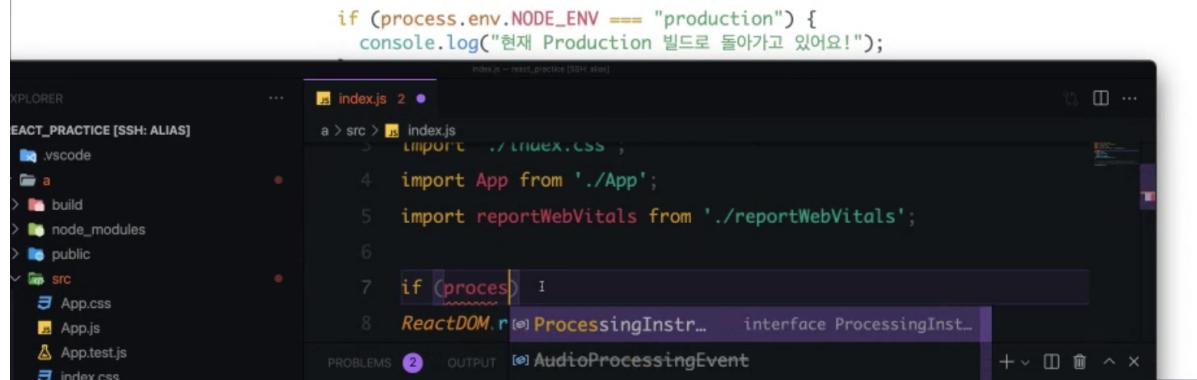
/relices 웰리스 SW 엔지니어 트랙 13주차 실습

## 왜 Development로 배포하면 안 되는데요?

느려!

/relices 웰리스 SW 엔지니어 트랙 13주차 실습

## 진짜 Production 맞아요?



A screenshot of the Visual Studio Code interface. The left sidebar shows a file tree for a project named 'REACT\_PRACTICE [SSH: ALIAS]'. The 'src' folder contains files like App.css, App.js, App.test.js, and index.css. The main editor tab shows a portion of 'index.js' with the following code:

```
if (process.env.NODE_ENV === "production") {
  console.log("현재 Production 빌드로 돌아가고 있어요!");
```

The code editor has syntax highlighting and intellisense, showing suggestions for 'process' and 'ReactDOM'.

웹팩

## Webpack?



웹팩은 여러가지를 합쳐주는 번들러이다.

왜 쓰나 이걸?

## 브라우저에 이런게 있었나?

```
import React from "react";
import ReactDOM from "react-dom";
import "./index.css";
import App from "./App";
import reportWebVitals from "./reportWebVitals";
```

하위호환에서는 저러한 코드 사용이 불가능

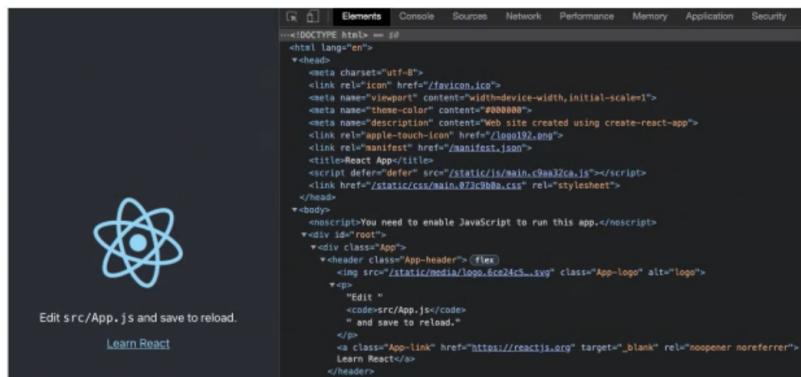
/elice 웰리스 SW 엔지니어 트랙 13주차 실습

## 아니요, 이렇게 했었죠!

```
<!DOCTYPE html>
<html>
  <head> </head>
  <body>
    <script src="./src/a.js"></script>
    <script src="./src/b.js"></script>
  </body>
</html>
```

/elice 웰리스 SW 엔지니어 트랙 13주차 실습

## 파일을 다시 보자.



js가 bundle.js로 변경되어 있다.

heroku 는 0원이지만, 서버가 있는게 아니다. 소스코드를 띄워주는것이다.

클라우드에서 서버를 빌려서 작업하는 것 (VM ⇒ Virtual Machine) Azure , AWS(점유율 높고, 매우 안정적이며, 다양한 기능 제공 대신 비싸다.)

aws는 가입하면 100달러를 준다. 이걸로 버팀 학생이면 매년 100달러 준다.

Web Hosting ⇒ 서버의 ‘일부자원’을 빌려서 작업 (PaaS) HeroKu

우리는 ci/cd 연습할 땐 heroku 사용할 것 → 공짜이니까. 혼자 프로젝트할 땐 AWS 써보는 것을 추천

relices 웰리스 SW 엔지니어 트랙 13주차 실습

## 유료 클라우드를 쓰면, 우리에게 맞는 사이즈는 뭘까?

프론트만 쌩으로 돌리면 : Github Page, Netlify, Vercel 같은 걸로 때워도 충-분

Node.js도 돌릴 건데요! : 시간 제한을 감수할 수 있다면 Heroku로도 충분…

Nginx 같은 웹 서버도 돌리고, 다양한 작업을 하고 싶어요. : 어지간하면 램 2G, 1 Core 정도로도 충분…

전 Docker도 빵빵하게 돌리고 싶은데요! : 아무리 커도 8G, 4 Core 이상으로 올릴 필요 없음!

/\*elice\*/ 웨리스 SW 엔지니어 트랙 13주차 실습

## 실물 서버 쓰면 안 되나요?

### 장점

엄청 싸다! (일시불로 지불해야 하긴 하지만…)

'내' 서버라는 안도감(?)

but 단점: DDos 같은 공격에 굉장히 취약하다.

/\*elice\*/ 웨리스 SW 엔지니어 트랙 13주차 실습

## 실물 서버 쓰면 안 되나요?

안녕하세요

펌프잇업 관리자입니다.

갑작스러운 서버 점검 진행으로 게임 이용에 불편을 드린 점 대단히 사과드립니다.

12월 21일 화 서버의 저장 장치 손상으로 인해 서비스 장애가 발생하였습니다.  
장치 교체 및 백업 데이터로 복구를 시도하였으나 백업 데이터까지 연쇄 손상되어  
인터넷계도 펌프잇업의 모든 서버 데이터가 유실되었습니다.

<데이터 손상 범위>

\* 펌프잇업의 모든 서버 데이터

<데이터 복구 일정>

UCS	~2018-09-06
홈페이지 계정	~2018-09-06
카드 정보	~2021-12-20
플레이 데이터	복구 불가
진호 횟수 데이터	~2021-08-11
스팀 해금 데이터	~2021-08-11
보유 아바타	~2021-08-11

\* 한국 시간 기준으로 (GMT+9) 입니다.

<게임 이용>

1. 2021-08-11 이전 계정은 정상 로그인 가능합니다

2. 2021-08-11 이후 생성된 계정은 데이터가 유실되었습니다.

<홈페이지 이용>

1. 일부 유저는 정상 로그인 가능합니다 (2018-09-06 이전 생성)

2. 로그인이 되지 않는 계정은 자주에 복구할 수 있는 기능을 준비 중이니 기다려주시기 바랍니다.

Nginx 는 뭐인가?

: 웹서버이다.

/\*elice\*/ 엘리스 SW 엔지니어 트랙 13주차 실습

## Apache Web Server, Nginx

**웹 서버(Web Server)**는 HTTP를 통해 웹 브라우저에서 요청하는 HTML 문서나 오브젝트(이미지 파일 등)을 전송해주는 서비스 프로그램을 말한다.

웹 서버 소프트웨어를 구동하는 하드웨어도 웹 서버라고 해서 혼동하는 경우가 간혹 있다.

출처: 위키백과 "웹 서버" 문서

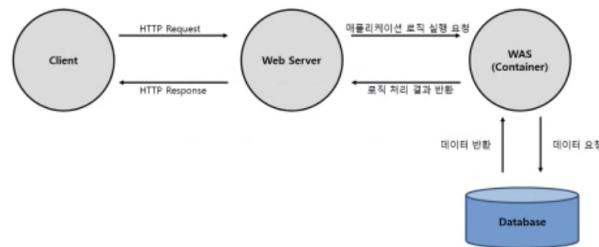
/\*elice\*/ 엘리스 SW 엔지니어 트랙 13주차 실습

## 최고의 의문

정의만 보면 Express도 웹 서버 아닌가요?

근데 왜 Nginx를 '굳이' 쓰는거죠?

## Web Server/Web Application Server



### 웹 어플리케이션 서버(Web Application Server)

웹 어플리케이션과 서버 환경을 만들어 동작시키는 기능을 제공하는 소프트웨어 프레임워크.

express는 정확히 웹 어플리케이션 서버(WAS)이다.

WAS는 동적인 처리가 가능 (상황에 따라 추가적인 행동이 가능)

웹서버는 정적인 파일만 가능 / WAS는 동적인(복잡한) 요청 가능

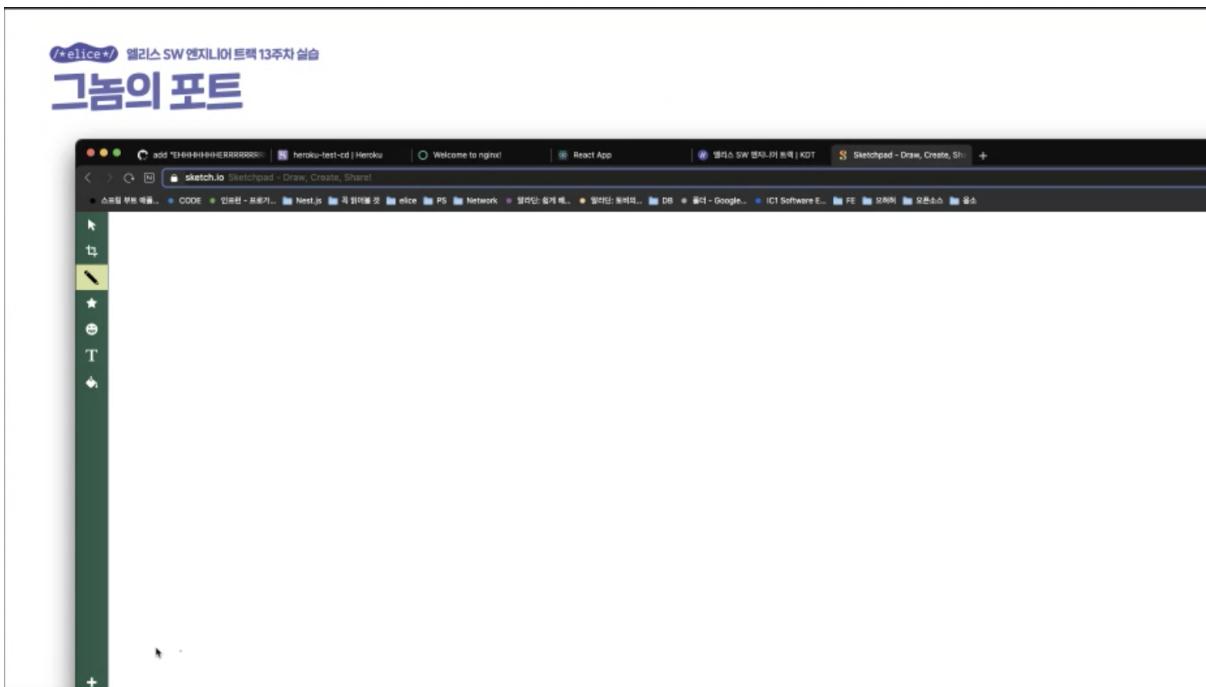
404가 뜰 경우 문구를 따로 제작한 페이지는 WAS가 맛이 갔는데 어떻게 404 페이지를 띄우지?

이럴 때 web server가 정적인 파일로 404 페이지를 보내줌

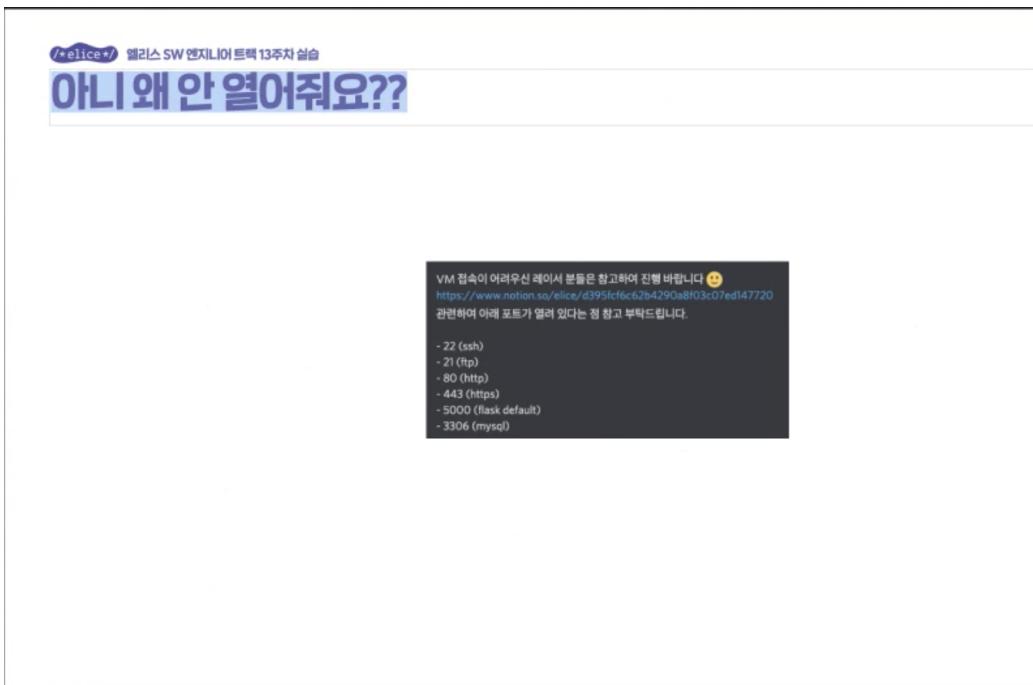
Nginx가 이러한 역할을 함 웹서버 → WAS , WAS → 웹서버로 넘기기 가능

## 왜 Web Server와 WAS를 분리할까?



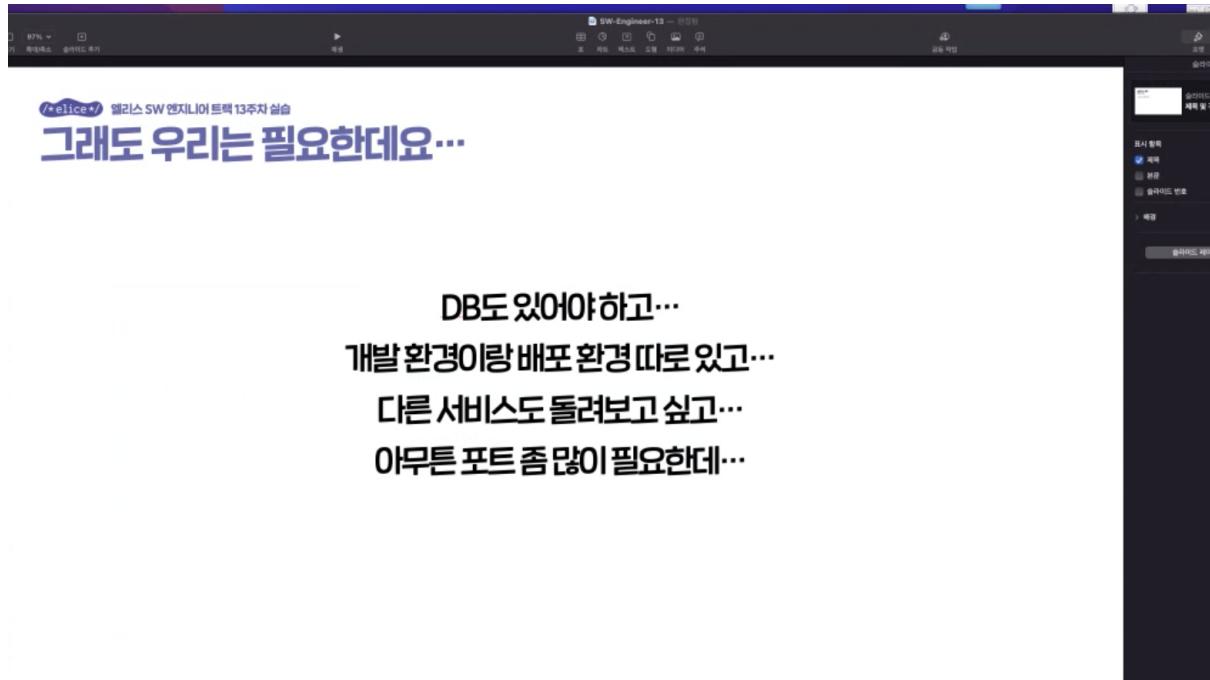


포트는 무엇일까?



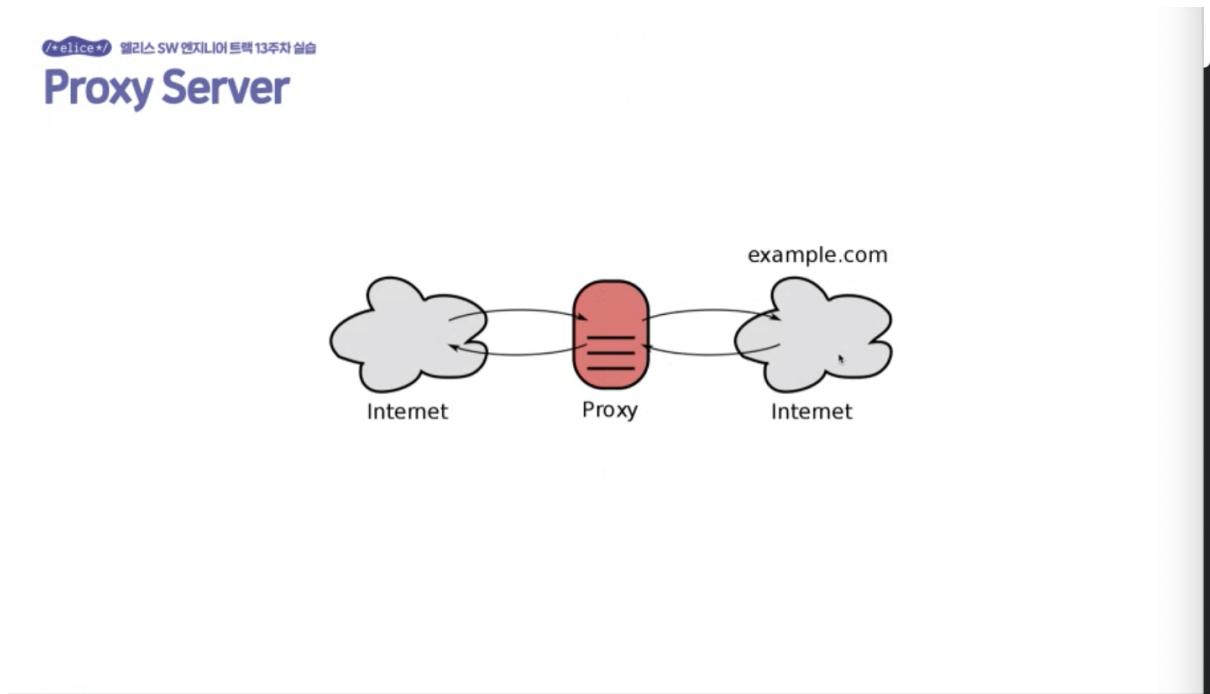
포트를 더 못열어주는 이유?

포트를 막 열어주면 보안에 취약

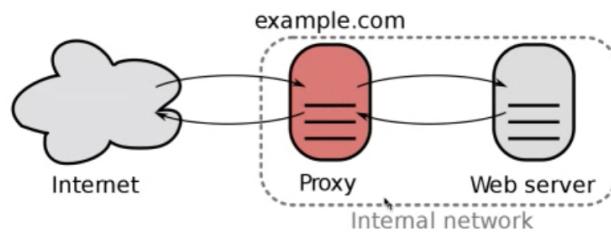


### Proxy server

“스토커”와 “나”와 “스토커랑 나와 친구사이인 친구” 사이의 관계



## Nginx Reverse Proxy



포트가 막힌 상태에서 포트에 들어갈 수 있도록 한다.

## Q. 한 서버에서 서로 다른 두 개의 서비스를 만들려면?

그냥 포트 번호 다르게 하면 되겠지?

## Q. 한 서버에서 서로 '같은' 두 개의 서비스를 만들려면?

엇…? 이것도 포트 다르게 하면 안 되나요?

## Q. 한 서버에서 서로 '같은' 두 개의 서비스를 만드는데, 개발 환경과 배포 환경을 엄격하게 구분하려면?



## 가상화



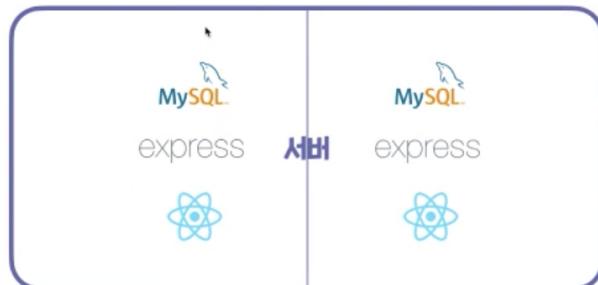
개발서버 와 배포서버를 하나의 서버로 구축?

## 가상화



이를 가상화라고 부른다

## 가상화



서버를 완전 격리된 컨테이너로 놓수 있다면...

개발 환경 구축하다가 충돌이 발생하거나, 의존성 고칠 이유도 없고  
각각의 환경이 서로 독립된 구조라는게 보장됨.

이러한 기술을 사용하기 위해 도커를 사용한다.

도커쓰려면 포트가 많이 필요 —> Nginx 리버스 프록시 사용해야함

## CI/CD

### Continuous Integration 지속적인 통합

커밋을 하기만 하면 테스트 ⇒ 빌드를 자동으로 해주는 것

CI를 지원하는 방법이 많다.

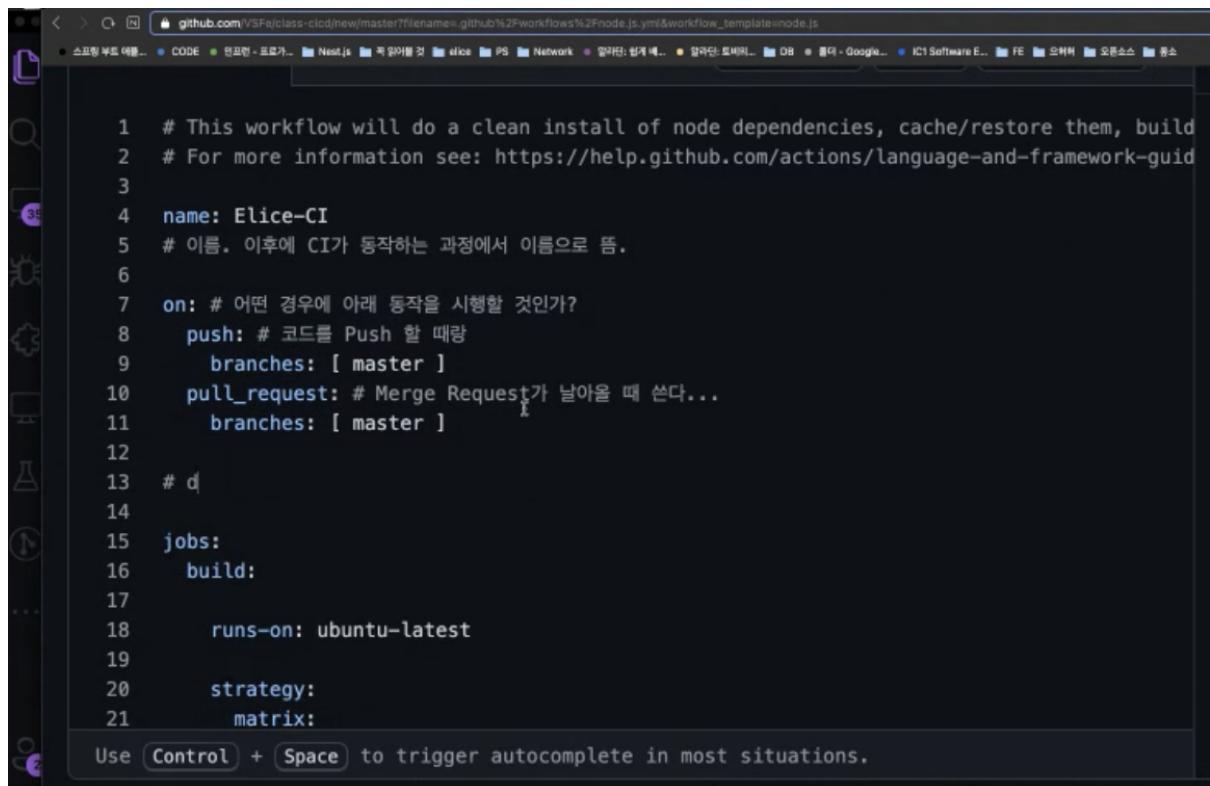
- Github - Action (not bad)
- Gitlab - not bad
- Travis CI - 별도 서버 X지만 상황에 따라 유료
- Jenkins (현업에서 많이 쓴 대신 별도의 빌드 서버가 있어야 됨 개인 플젝시 서버있어야돼서 좀 어렵..)

### Continuous Deployment 지속적인 배포

## 디스코드 Web Hook



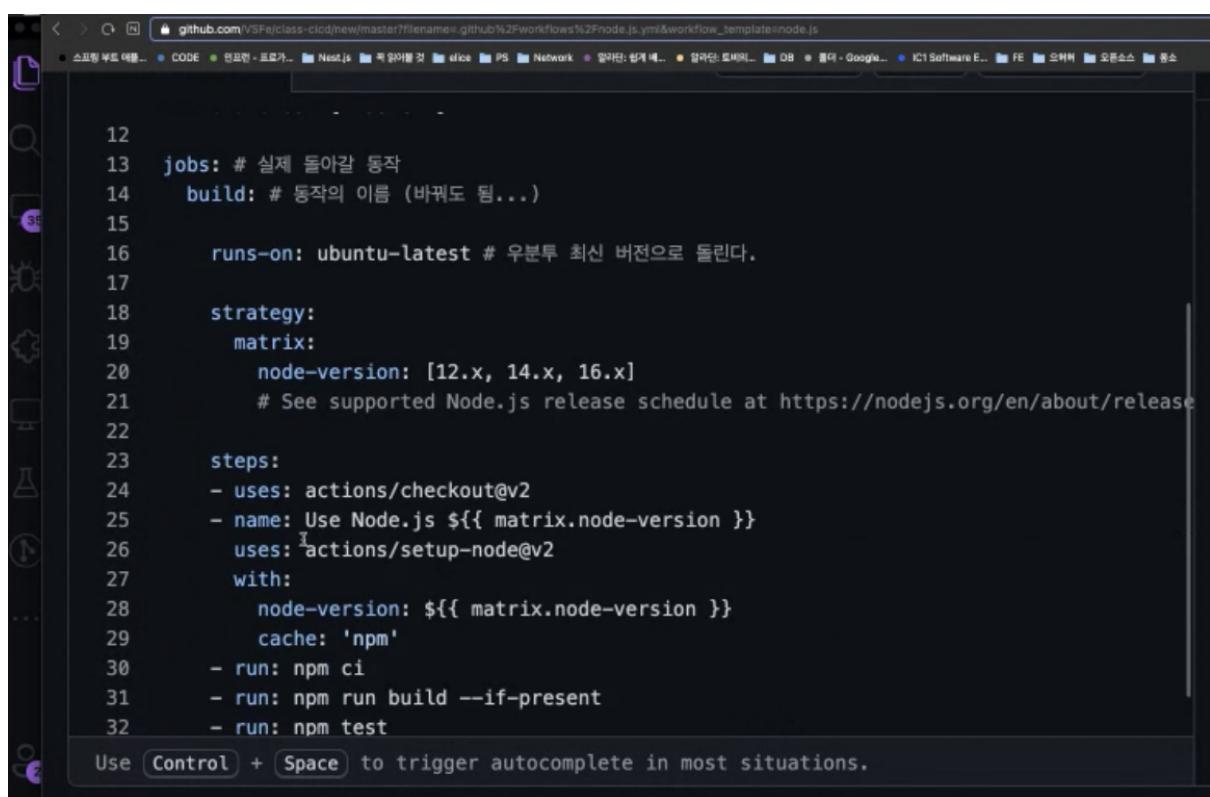
CI 는 yml 로 작성



A screenshot of the Visual Studio Code interface showing a GitHub workflow template for Node.js. The code is displayed in a dark-themed editor. The URL in the address bar is [github.com/VSF-Fe/class-cicd/new/master?filename=.github%2Fworkflows%2Fnode.js.yml&workflow\\_template=node.js](https://github.com/VSF-Fe/class-cicd/new/master?filename=.github%2Fworkflows%2Fnode.js.yml&workflow_template=node.js). The code itself is as follows:

```
1 # This workflow will do a clean install of node dependencies, cache/restore them, build
2 # For more information see: https://help.github.com/actions/language-and-framework-guides
3
4 name: Elice-CI
5 # 이름. 이후에 CI가 동작하는 과정에서 이름으로 뜸.
6
7 on: # 어떤 경우에 아래 동작을 시행할 것인가?
8   push: # 코드를 Push 할 때랑
9     branches: [ master ]
10  pull_request: # Merge Request가 날아올 때 쓴다...
11    branches: [ master ]
12
13 # 디렉토리
14
15 jobs:
16   build:
17
18     runs-on: ubuntu-latest
19
20     strategy:
21       matrix:
```

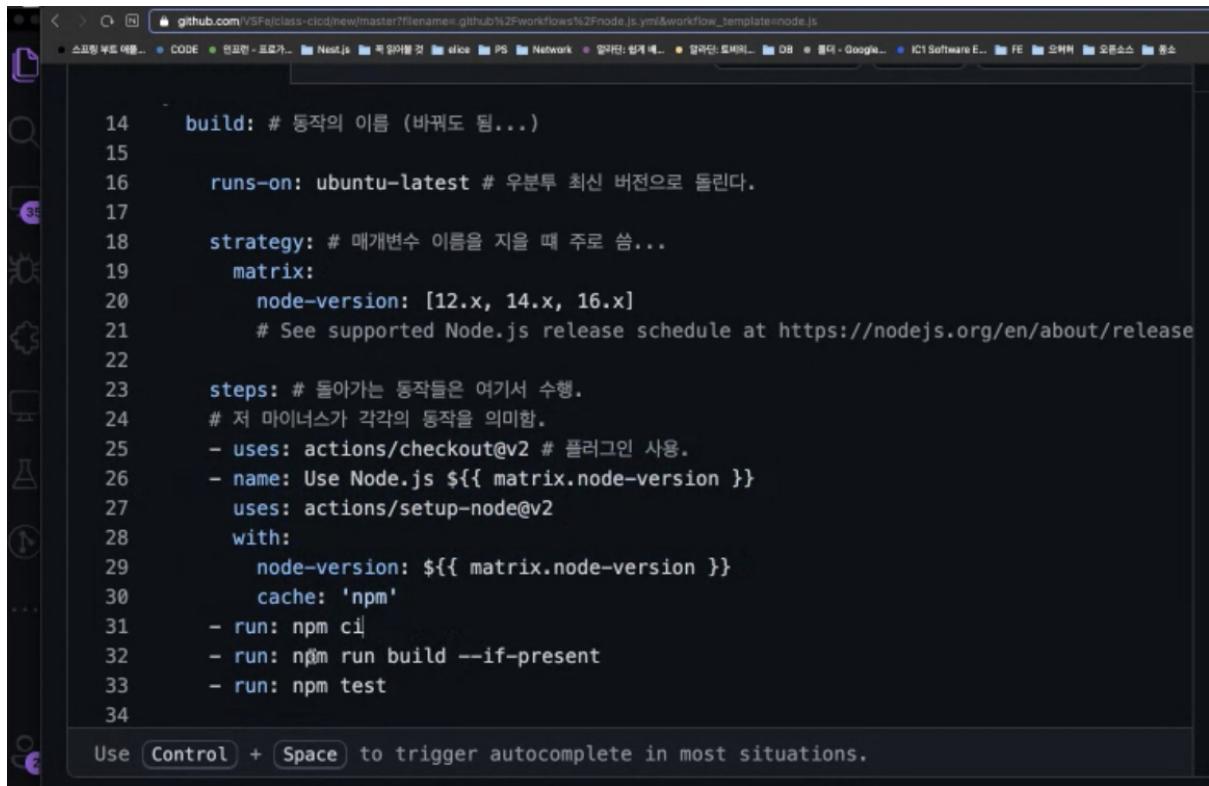
At the bottom of the code editor, there is a message: "Use **Control** + **Space** to trigger autocomplete in most situations."



A continuation of the screenshot from the previous section, showing the rest of the GitHub workflow template for Node.js. The code continues from line 12:

```
12
13 jobs: # 실제 돌아갈 동작
14   build: # 동작의 이름 (바꿔도 됨...)
15
16     runs-on: ubuntu-latest # 우분투 최신 버전으로 돌아간다.
17
18     strategy:
19       matrix:
20         node-version: [12.x, 14.x, 16.x]
21         # See supported Node.js release schedule at https://nodejs.org/en/about/release-schedule
22
23     steps:
24       - uses: actions/checkout@v2
25       - name: Use Node.js ${{ matrix.node-version }}
26         uses: actions/setup-node@v2
27         with:
28           node-version: ${{ matrix.node-version }}
29           cache: 'npm'
30         - run: npm ci
31         - run: npm run build --if-present
32         - run: npm test
```

At the bottom of the code editor, there is a message: "Use **Control** + **Space** to trigger autocomplete in most situations."



A screenshot of a GitHub Actions workflow configuration file (node.js.yml) in a code editor. The file is titled 'node.js.yml' and is located in a 'nodejs' directory. The code is written in YAML and defines a workflow named 'nodejs'. The workflow includes a build step, runs on 'ubuntu-latest', and uses a strategy matrix for Node.js versions 12.x, 14.x, and 16.x. It also includes steps for checkout, setup node, and running npm ci, build, and test commands.

```
build: # 동작의 이름 (바꿔도 됨...)
runs-on: ubuntu-latest # 우분투 최신 버전으로 돌린다.

strategy:
  matrix:
    node-version: [12.x, 14.x, 16.x]
    # See supported Node.js release schedule at https://nodejs.org/en/about/release

steps: # 돌아가는 동작들은 여기서 수행.
# 저 마이너스가 각각의 동작을 의미함.
- uses: actions/checkout@v2 # 플러그인 사용.
- name: Use Node.js ${{ matrix.node-version }}
  uses: actions/setup-node@v2
  with:
    node-version: ${{ matrix.node-version }}
    cache: 'npm'
- run: npm ci
- run: npm run build --if-present
- run: npm test
```