[SERVER] 포팅 메뉴얼

로컬에서 EC2 서버에 접속하기

💡 **ssh -i {pem키이름},pem ubuntu@{도메인}**

1. 로컬에서 EC2에서 받은 pem 키가 있는 위치에서 powershell을 열어줍니다.
2. 터미널 창에서 위의 명령어를 입력하면 접속할 수 있습니다.

서버 환경 구축하기

서버에서 데이터를 활용할 DB환경을 구축합니다.

MySQL 설치하기

1. ec2서버에 mysq l 을 설치 및 실행합니다.
2. 사용자 계정 및 비밀번호를 설정하고 로그인합니다
3. 필여한 DB를 생성합니다.
    a. 스프링 JPA 로 필요한 테이블은 서비스가 실행되며 자동 생성됩니다.
4. mysql서버를 실행하면 서비스에서 해당 DB에 접근할 수 있습니다.

```
apt-get update
apt-get upgrade
apt-get install mysql-server
```

```
mysql -u root -p
mysql> alter user "root"@"localhost" identified with mysql_na
mysql> create database 'dbname'
```

MongoDB 설치하기

1. ec2서버에 MongoDB 을 설치 및 실행합니다.
2. 사용자 계정 및 비밀번호를 설정하고 로그인합니다
3. 필요한 컬렉션을 생성합니다.
4. MongoDB 서버를 실행하면 서비스에서 해당 DB에 접근할 수 있습니다.

```
sudo apt-get install gnupg
curl -fsSL https://pgp.mongodb.com/server-6.0.asc | \
   sudo gpg -o /usr/share/keyrings/mongodb-server-6.0.gpg \
   --dearmor
echo "deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/mo
sudo apt-get update
sudo apt-get install -y mongodb-org

// mongodb start
sudo service mongod start
sudo service mongod status

// 접속 및 계정설정
mongosh
db.createUser({user: "input", pwd: "input", roles:["root"]});
db.createCollection(name, options)
```

https 접근을 위한 환경을 세팅합니다.

SSL 인증서 발급받기

certbot을 활용하여 ec2 도메인에 대한 letsencrypt 인증서를 발급합니다.

```
sudo apt update
sudo apt-get install letsencrypt -y
cd /root
sudo service nginx stop
certbot certonly --standalone -d owl-dev.me -d www.owl-dev.me
```

nginx 를 통해 url 주소에 따라 백엔드 / 프론트엔드 요청을 처리합니다

nginx 를 설치합니다

```
sudo apt-get install nginx
```

/etcc/nginx/sites-enabls/ 아래에 있는 *.conf 파일을 생성 및 수정해서 사용자 설정을 구성합니다.

아래 코드는 예시입니다.

```
server{
        listen 8080;

        server_name i10b102.p.ssafy.io;

        return 301 https://localhost$request-uri;
}

server{
        listen 443 ssl;


        server_name i10b102.p.ssafy.io;

        ssl_certificate /etc/letsencrypt/live/i10b102.p.ssafy
        ssl_certificate_key /etc/letsencrypt/live/i10b102.p.s

        location /{
                root /home/ubuntu/saferoutine/S10P12B102/FE_S
                index index.html;
```

```
                try_files $uri /index.html;
        }
        location /api {

                proxy_pass https://localhost:8443;
                charset utf-8;
                proxy_redirect off;

                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x
                proxy_set_header X-NginX-Proxy true;
                proxy_set_header X-Forwarded-Proto $scheme;

        }

}
~
```

프로젝트 파일을 EC2 내부에 저장 및 실행합니다

GIT 에 있는 프로젝트를 EC2에 복제하기

```
git clone { git 주소 }
```

SpringBoot 실행하기

1. https 설정을 위해 SSL 인증서를 통해 받은 pem 키로 인증서 키 생성
2. 해당 키를 spring 프로젝트의 resource 폴더로 복사
3. 프로젝트에서 해당 키를 읽을 수 있도록 하기 위해 파일에 x 권한 부여
4. gradlew 를 통해 프로젝트 빌드
5. java -jar 명령어를 통해 build 폴더 내의 jar파일 실행

```
sudo cp /etc/keystore.p12 ./src/main/resources/
sudo chmod +r ./src/main/resources/keystore.p12
```

```
chmod +x ./gradlew
./gradlew clean build
java -jar ./build/libs/{@@}.jar
```

React 실행하기

1. React 프로젝트의 src 폴더가 있는 위치로 이동합니다.
2. 필요한 의존성을 설치 후 빌드합니다.

```
npm i
rm -rf /* dist 과거에 실행한적 있다면 실행 ( 현재 위치에 dist 파일이 있
npm run build


sudo service nginx restart
```

Nginx 실행하기

```
service nginx start
```