



KoolCart

TAB, ORDER, STAY FRESH



포팅 메뉴얼

A101 - PROJECT101

삼성 청년 SW 아카데미 서울 캠퍼스 10기 공통 프로젝트

24.01.02 ~ 24.02.16

담당 컨설턴트 : 오형남 컨설턴트

박건률(팀장), 이수민, 임병은, 이규형, 노현경, 조한준





01

02

03

EC2 서버 세팅

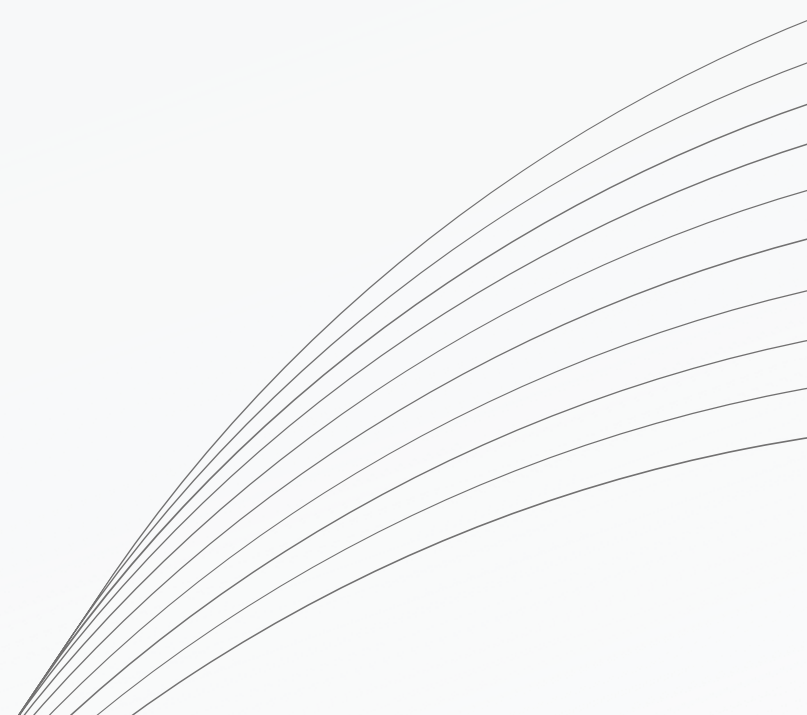
Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Duis
vulputate nulla at ante
rhoncus, vel efficitur
felis condimentum.
Proin odio odio.

빌드 상세 내용

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Duis
vulputate nulla at ante
rhoncus, vel efficitur
felis condimentum.
Proin odio odio.

외부 서비스

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Duis
vulputate nulla at ante
rhoncus, vel efficitur
felis condimentum.
Proin odio odio.



EC2 서버 설정

MySQL 설치

```
sudo apt install mysql-server -y
```

```
sudo mysql
```

```
use mysql;  
create user '아이디'@'%' identified with mysql_native_password by '비밀번호';
```

```
create database [DB스키마이름];
```

```
grant all privileges on [DB스키마이름].* to '아이디'@'%';
```

```
exit
```

A decorative graphic consisting of multiple thin, black, wavy lines that originate from the left edge and curve downwards and to the right, creating a sense of motion or a stylized wave.

EC2 서버 설정

Docker 설치

EC2 서버에 접속하여 아래의명령어를순차적으로실행합니다.

```
apt-get update
```

```
apt-get install sudo
```

```
apt-get install vim
```

```
sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common curl -fsSL  
https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add sudo add-apt-repository \
```

```
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \
```

```
$(lsb_release -cs) stable"
```

```
sudo apt-get update && sudo apt-get install docker-ce docker-ce-cli containerd.io
```



EC2 서버 설정

docker compose 설치

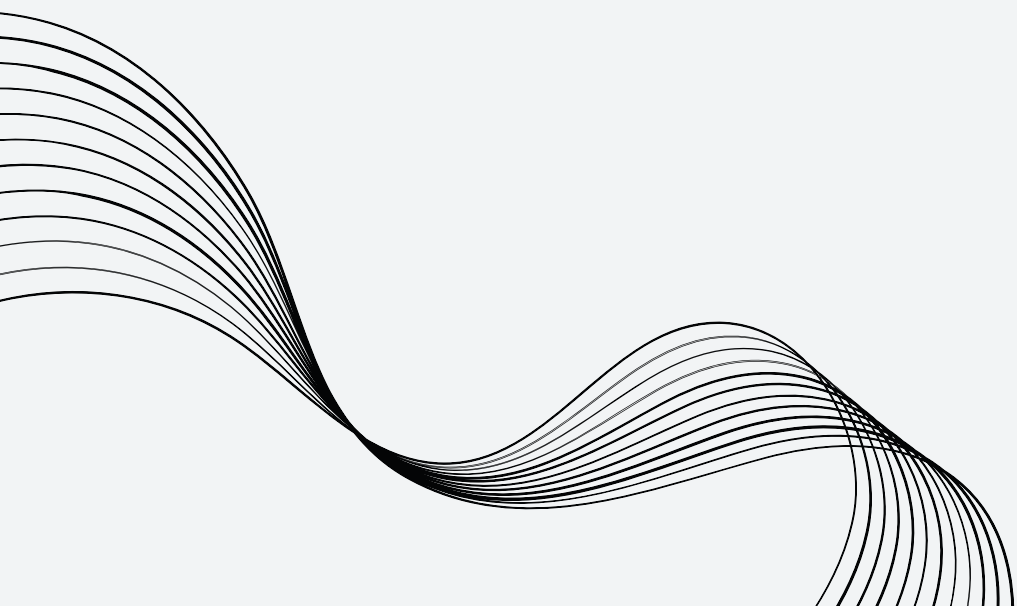
```
$sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

```
$sudo chmod +x /usr/local/bin/docker-compose
```

도커 이미 설치되어 있으므로 바로 레포지토리 클론

```
$ git clone [https://lab.ssafy.com/s10-webmobile3-sub2/KoolCart.git](https://lab.ssafy.com/s10-webmobile3-sub2/KoolCart.git)
```

token으로 인증 -> 클론 완료



EC2 서버 설정

Nginx 설치

리버스 프록시로 사용할 nginx를 EC2서버에 설치합니다.

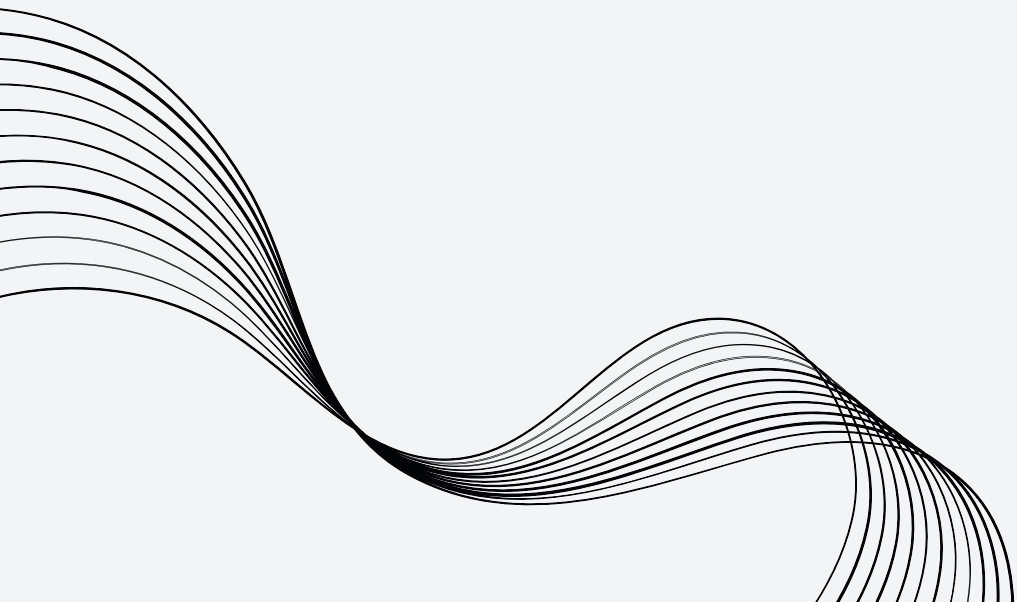
```
sudo apt install nginx
```

그이후아래의폴더에서리버스프록시관련설정을진행합니다.

```
cd /etc/nginx/conf.d
```

해당 프로젝트에서는 총 세 개의 nginx 컨픽을 사용중입니다.

default.conf : 운영할 서비스에 관련된 리버스프록시 설정 파일



빌드 상세 내용

KoolCart Dockerfile

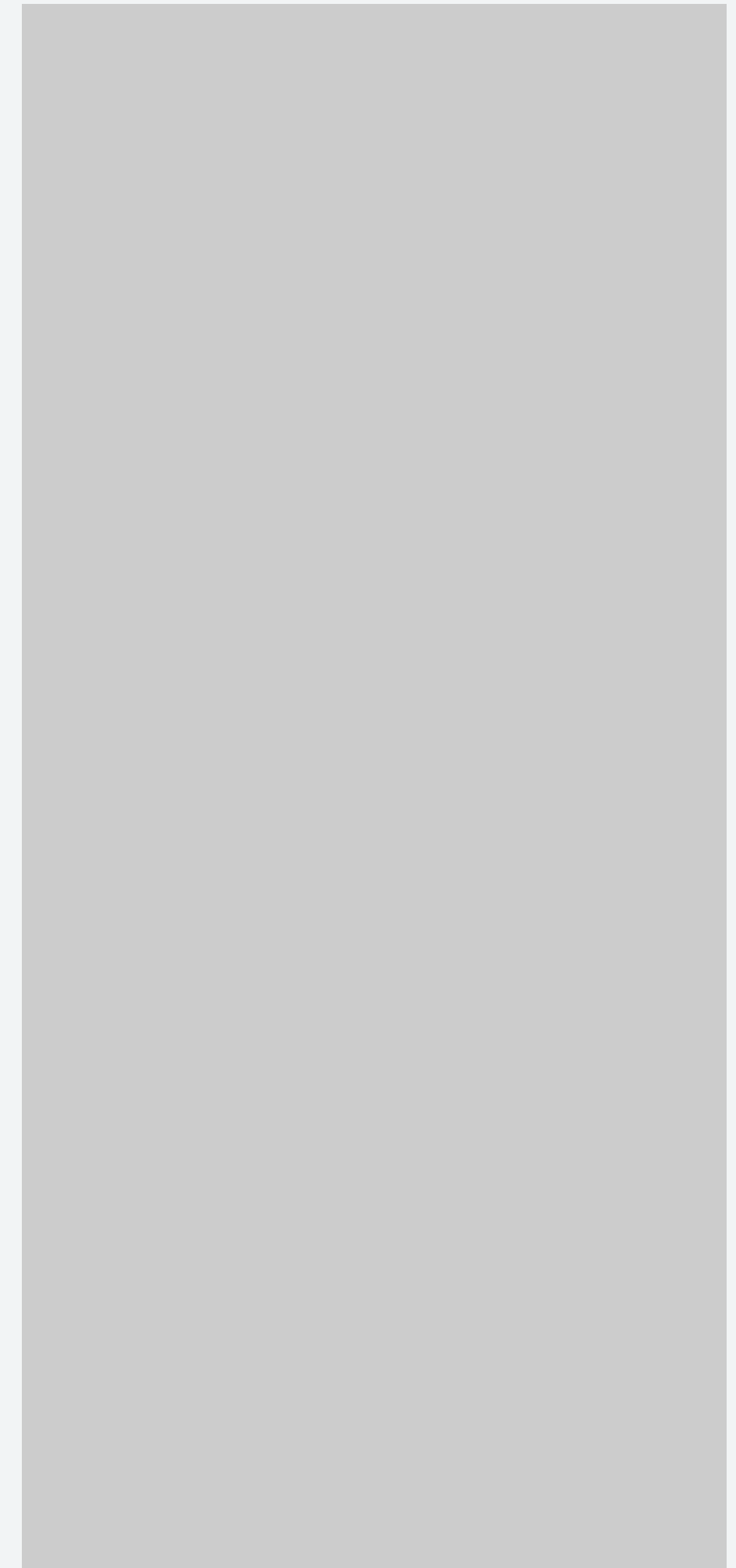
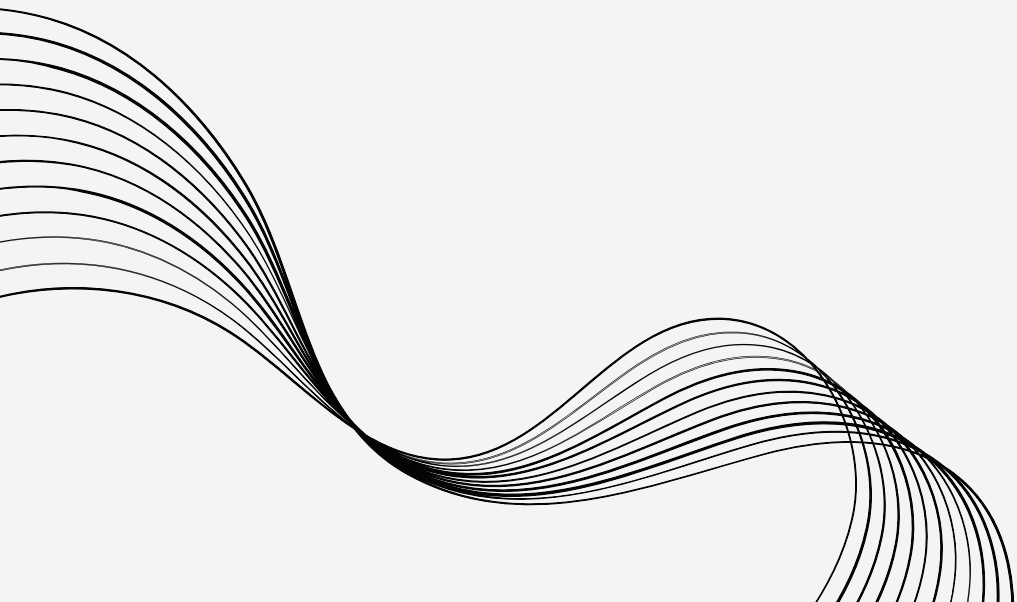
<Dockerfile code>

```
FROM openjdk:17
```

```
ARG JAR_FILE=build/libs/KoolCart-0.0.1-SNAPSHOT.jar
```

```
COPY ${JAR_FILE} koolcart.jar
```

```
ENTRYPOINT [ "java", "-jar", "koolcart.jar" ]
```



빌드 상세 내용

Shopping_mall_server Dockerfile

<Dockerfile code>

FROM python:3.9.13

ENV PYTHONUNBUFFERED 1

WORKDIR /code

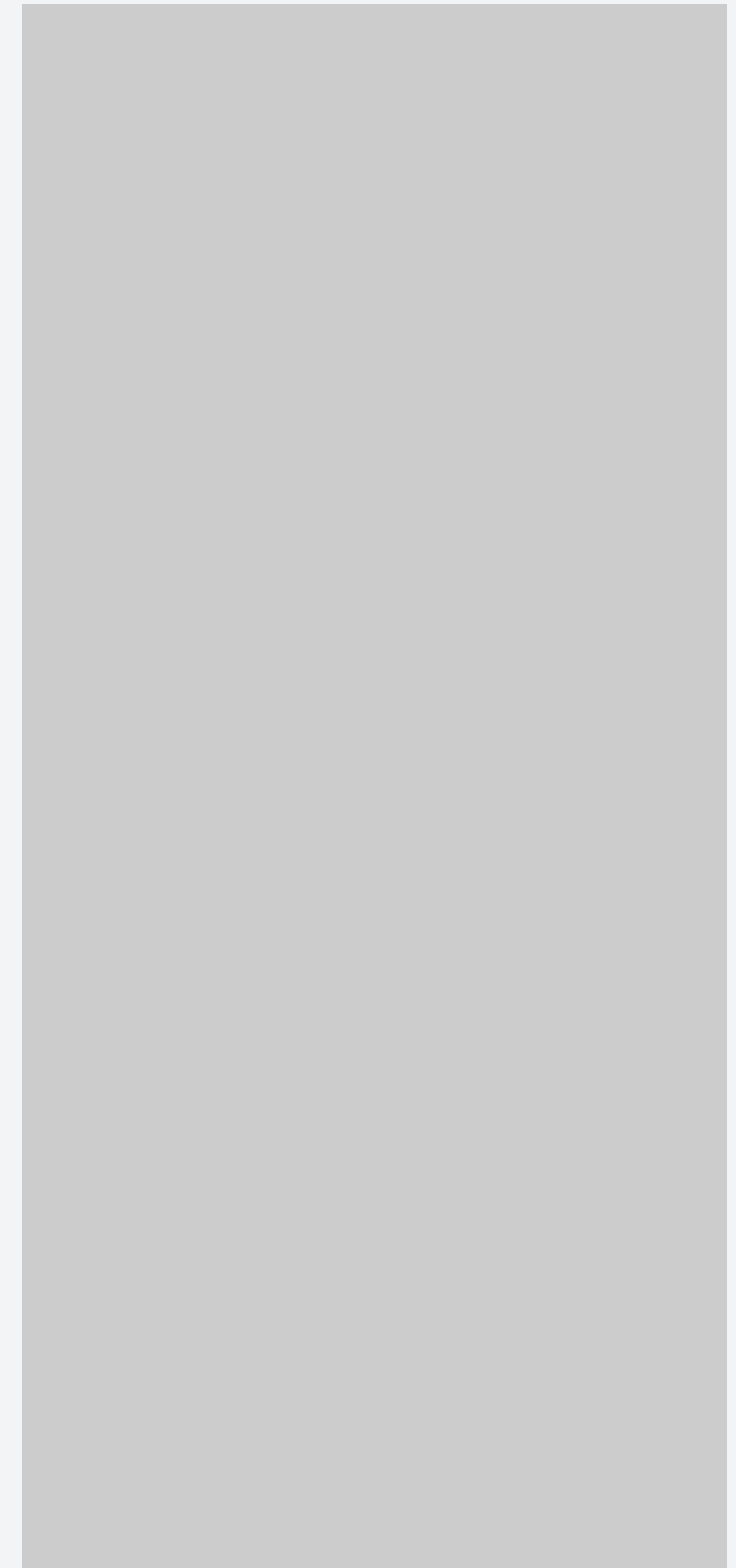
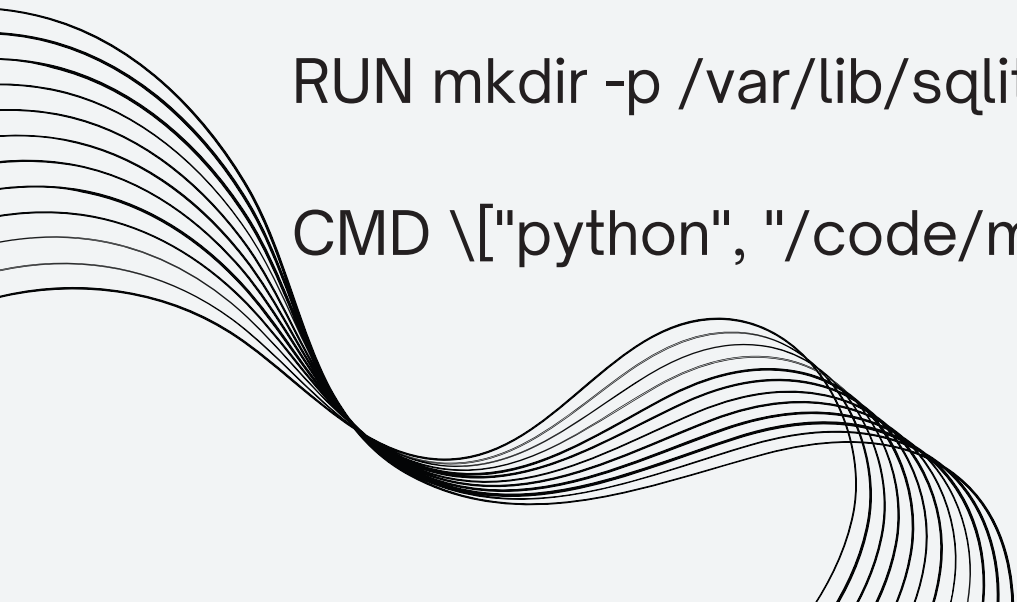
COPY requirements.txt /code/

COPY . /code/

RUN pip install -r requirements.txt

RUN mkdir -p /var/lib/sqlite && mv /code/db.sqlite3 /var/lib/sqlite/

CMD \["python", "/code/manage.py", "runserver", "0.0.0.0:8000"\]



빌드 상세 내용

Build 과정(Nginx - React vite)

기본적으로 필요한 파일들은 전부 Gitlab 코드에 명시되어 있기 때문에 Git Clone한뒤 진행합니다.

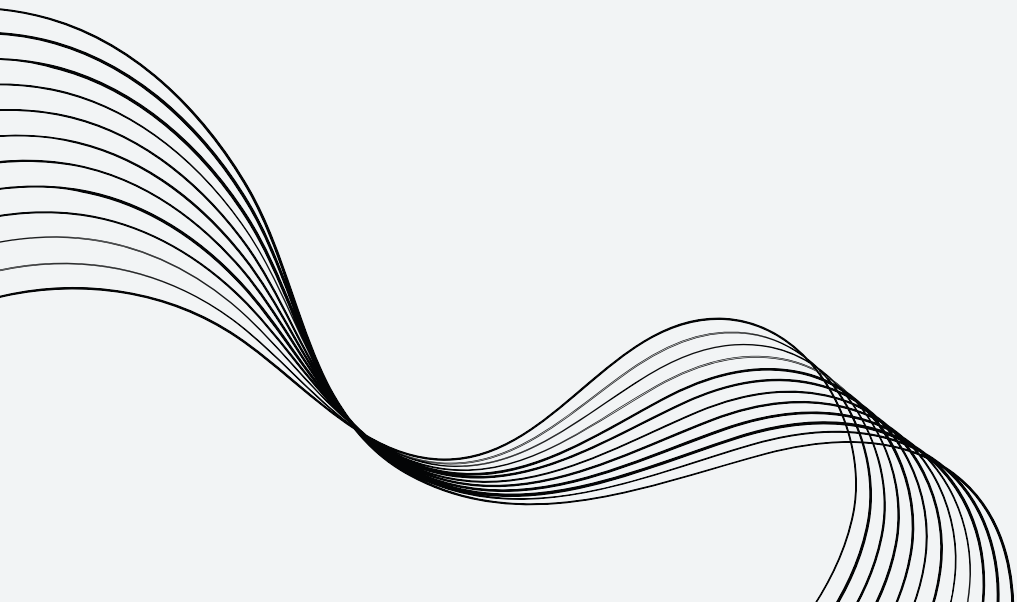
```
git clone https://lab.ssafy.com/s10-webmobile3-sub2/KoolCart.git
```

프로젝트 권한을 가진 Token으로 Clone

```
cd ~/KoolCart/src/main/frontend
```

```
npm install  
npm run build
```

```
sudo service nginx restart
```



빌드 상세 내용

Build 과정(Docker - Spring)

기본적으로 필요한 파일들은 전부 Gitlab 코드에 명시되어 있기 때문에 Git Clone한뒤 진행합니다.

```
git clone https://lab.ssafy.com/s10-webmobile3-sub2/KoolCart.git
```

프로젝트 권한을 가진 Token으로 Clone

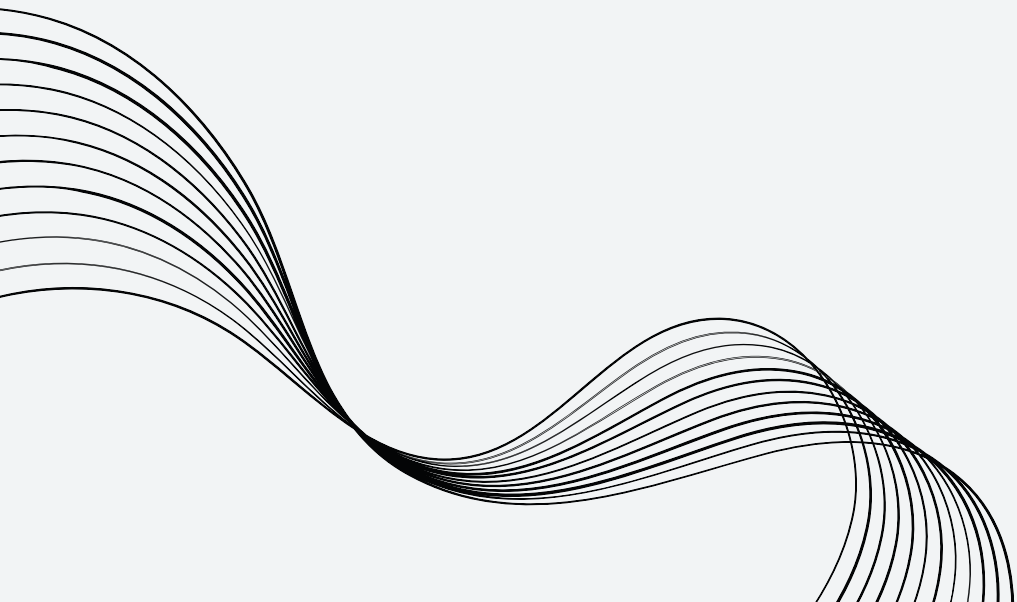
```
cd ~/KoolCart
```

```
chmod +x gradlew
```

```
./gradlew build
```

```
docker build -t koolcart .
```

```
docker run -dp 8080:8080 koolcart
```



빌드 상세 내용

Build 과정(Docker - Shoppingmall)

기본적으로 필요한 파일들은 전부 Gitlab 코드에 명시되어 있기 때문에 Git Clone한뒤 진행합니다.

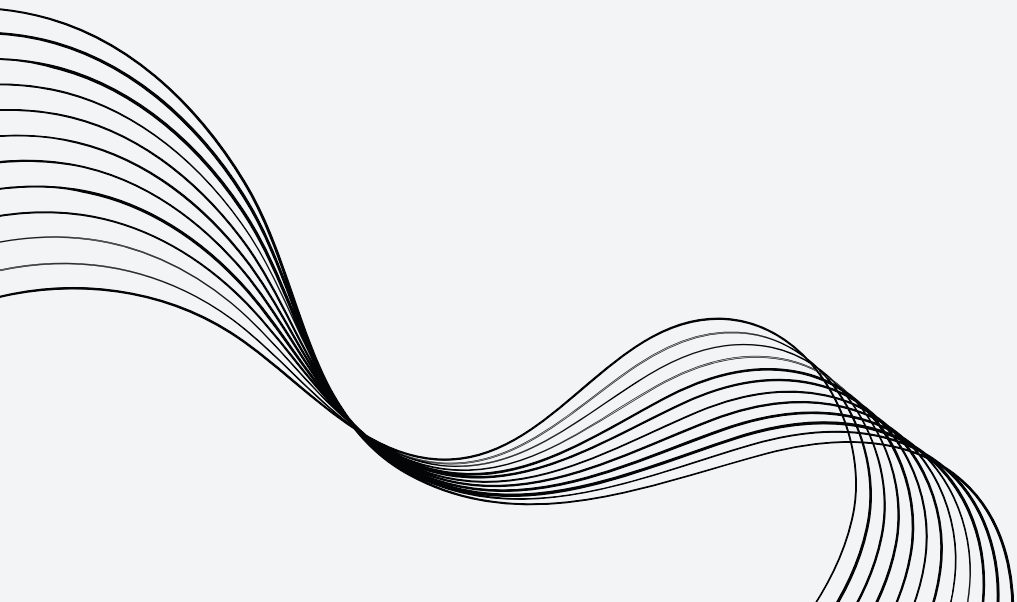
```
git clone https://lab.ssafy.com/s10-webmobile3-sub2/KoolCart.git
```

프로젝트 권한을 가진 Token으로 Clone

```
cd ~/KoolCart/shoppingmall_server
```

```
$ cd shoppingmall\_server
```

```
$ docker compose-up —build -d
```



외부 서비스

Weather API

해당 프로젝트에서는 openweathermap 을 사용합니다.
city와 apiKey를 입력하여 사용하면 됩니다.

```
const coordinates = await axios.get(
  `http://api.openweathermap.org/geo/1.0/direct?q=${city}&limit=5&appid=${apiKey}`
);
const desiredCity = coordinates.data.find((c) => c.name === city);
if (desiredCity) {
  const { lat, lon } = desiredCity;
  setCityName(desiredCity.local_names.ko)
  const weatherResponse = await axios.get(
    `https://api.openweathermap.org/data/2.5/weather?lat=${lat}&lon=${lon}&appid=${apiKey}&units=metric`
  );
  setWeatherData(weatherResponse.data);
} else {
  console.error(`City not found: ${city}`);
}
```

