웹 프로젝트 배포

SSAFY

SAMSUNG SOFTWARE ACADEMY FOR YOUTH

DAILY CONTENT

AWS EC2

Amazon Web Service

Elastic Compute Cloud

01

AWS EC2

AMAZON WEB SERVICE ELASTIC COMPUT CLOUD

Amazon에서 서비스하는 가상 서버

하드웨어에 대한 투자없이

가상 개발 환경을 통해

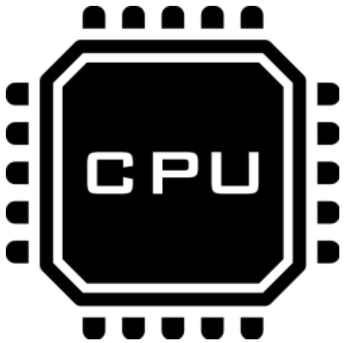
신속하게 개발하고 배포

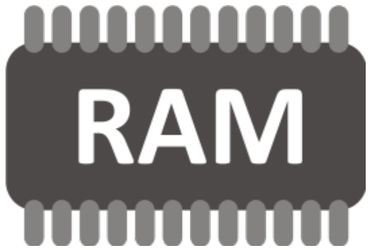
AWS EC2

AMAZON WEB SERVICE ELASTIC COMPUT CLOUD

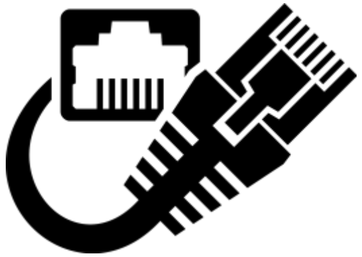
ELASTIC

확장이 용이한 컴퓨팅 용량 제공





STORAGE

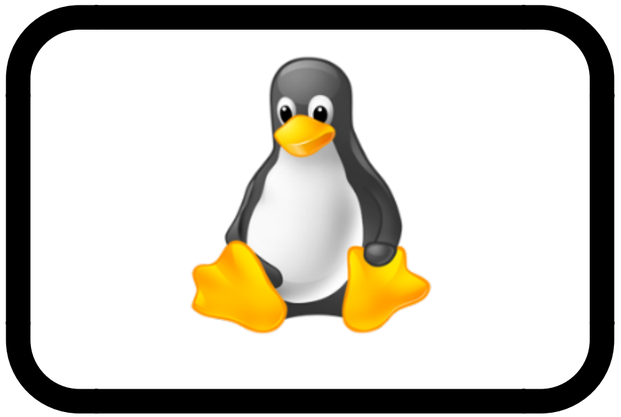


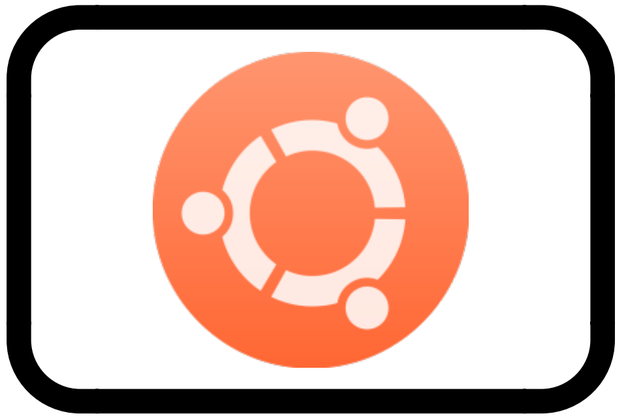
NETWORK

AWS EC2

AMAZON WEB SERVICE ELASTIC COMPUT CLOUD

AMI(Amazone Machine Image)를 기반으로 다양한 인스턴스 제공





AWS EC2

AMAZON WEB SERVICE ELASTIC COMPUT CLOUD

Amazon EC2

클라우드 웹 서비스 솔루션

어플리케이션 호스팅

컴퓨터

Amazon S3

SIMPLE STORAGE SERVICE

데이터 스토리지 솔루션

바이너리 파일 등의 데이터 저장

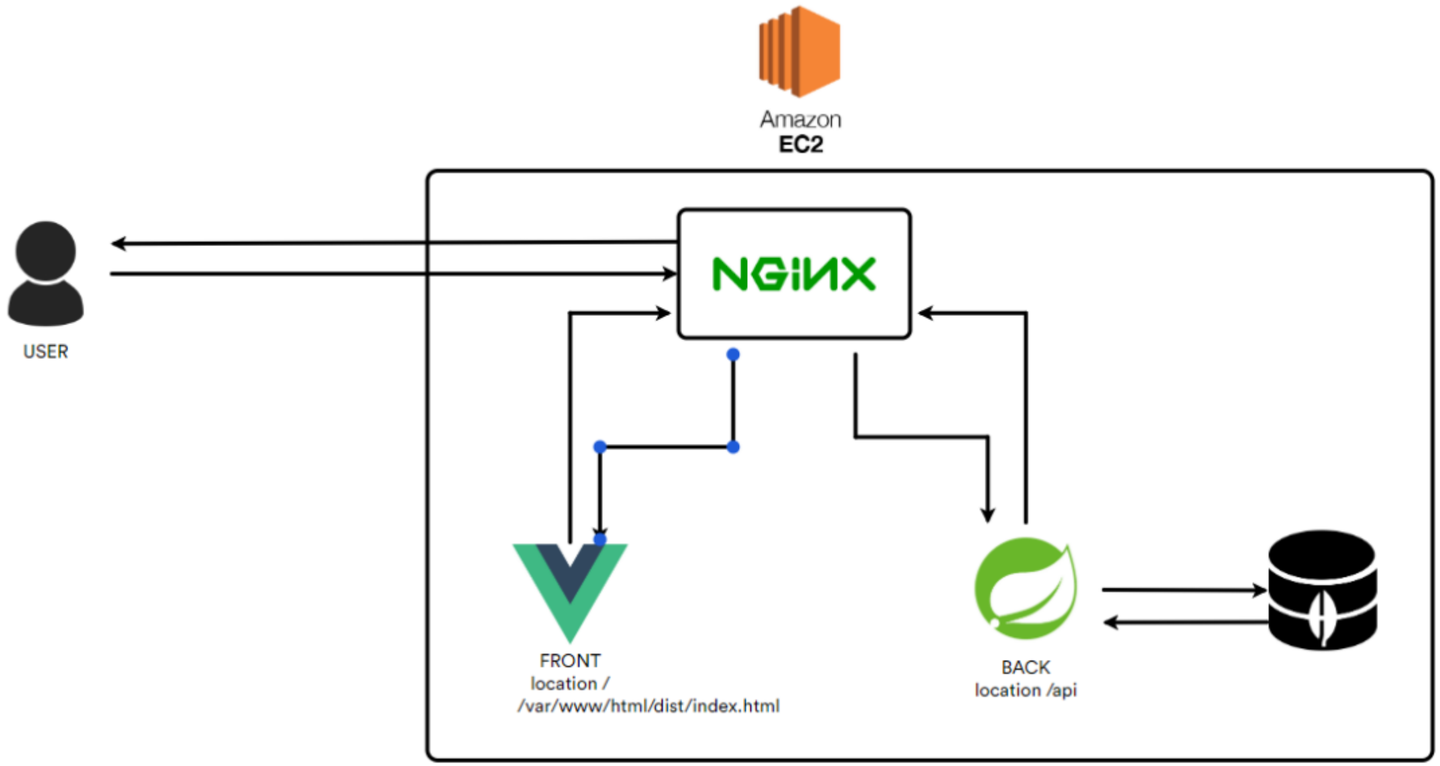
하드디스크

AWS DEPLOYMENT

Vue.js 와 Spring 프레임워크로 빌드 된 어플리케이션을 Amazon EC2 인스턴스에서 배포하기 위하여 전체적인 구조를 설계합니다. 또한, 어플리케이션 배포를 수행하기 위해 필요한 사항과 절차들을 제시하고 가이드합니다.

02

NGINX & VUE.JS & SPRING ARCHITECTURE



Server : Ubuntu

Front : Vue.js

Back : SpringBoot

WebServer SW : NginX

Database : MongoDB

NGINX?

트래픽이 많은 웹 사이트을 위해 확장성을 고려하여
설계한 비동기 이벤트 기반 구조의 웹서버 소프트웨어
Event-Driven

Proxy 서버 역할 및 정적 데이타에 대한 캐싱 수행
+ 로드 밸런싱, 보안

NGINX

Event-Driven 방식
적은 수의 쓰레드로 여러 요청 처리

대용량 요청 처리 유리

모듈 수가 적음

동적 컨텐츠 처리 X

성능

APACHE TOMCAT

1개의 프로세스 또는 쓰레드가
하나의 작업을 수행

대용량 요청 처리 한계

다양한 모듈 지원

동적 컨텐츠 처리 가능

안정성, 호환성, 확장성

NGINX 설치

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install nginx
```

apt-get update ： 설치된 패키지들의 새로운 버전이 있는지 확인

apt-get upgrage : apt-get update 를 통해 최신 버전이 확인된 패키지들의 버전을 업그레이드

NGINX 환경설정

conf 파일 설정

```
$ cd /etc/nginx/sites-available
$ vi default
```

환경 설정 후 Nginx 시작

```
$ sudo service nginx start
// or
$ sudo systemctl start nginx
```

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /var/www/html/dist;          # Front 빌드 파일 위치
    index index.html index.htm ;      # index 파일명
    server_name _;                    # 서버 도메인

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

    location /api {
        proxy_pass http://localhost:8399/api/;
        proxy_redirect off;
        charset utf-8;

        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-NginX-Proxy true;
    }
}
```

Frontend 설정

Backend Proxy 설정

FRONTEND 배포 - Vue.js

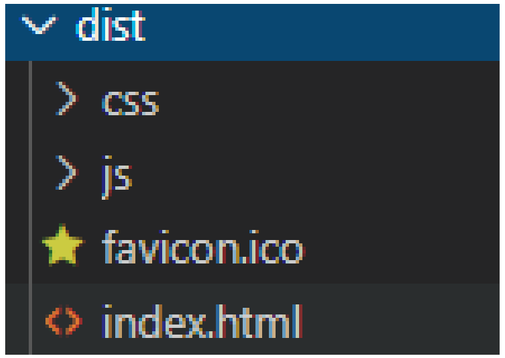
1. pakage.json

```
"scripts": {
  "serve": "vue-cli-service serve",
  "build": "vue-cli-service build",
  "lint": "vue-cli-service lint"
},
```

2. build 명령어 입력

```
$ npm run build
```

/dist 폴더



Local EC2 Instance

3. /dist → /var/www/html/dist

Nginx restart

변경사항이 적용되지 않을 경우 Nginx 서비스를 재시작

```
$ sudo service nginx restart
// or
$ sudo systemctl restart nginx
```

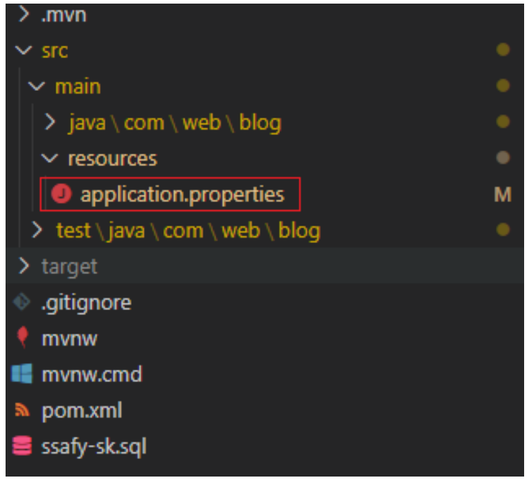
상태 확인

```
$ sudo service nginx status
// or
$ sudo systemctl status nginx
```

BACKEND 배포 - Spring

포트설정

application.properties 파일에서
server.port=<포트번호>



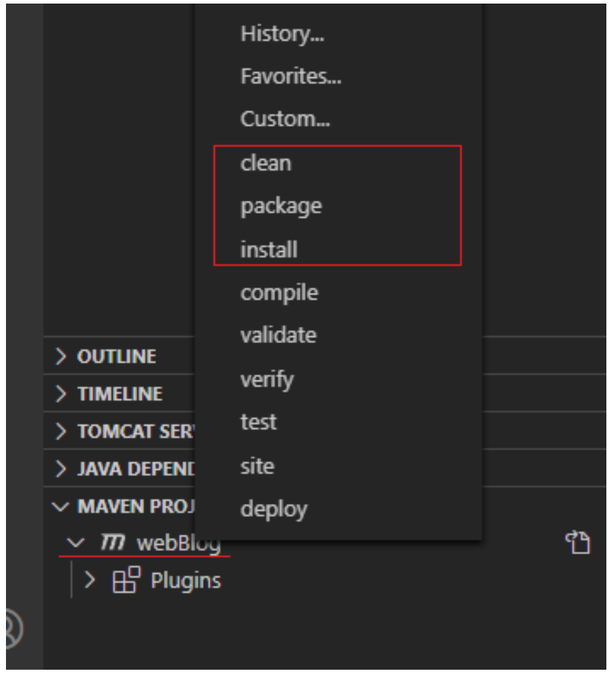

```
server.port=8399

spring.datasource.driverClassName=
spring.datasource.url=jdbc:mysql:/
```

빌드

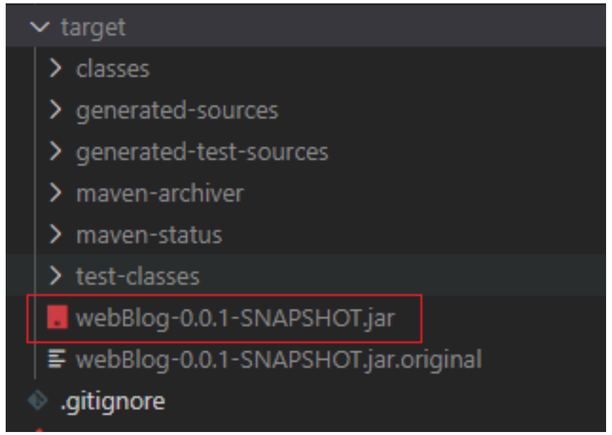
VS Code 좌측하단의 Maven Projects에서
프로젝트 오른쪽 클릭

clean > install > package 순으로 실행



빌드 확인

target 폴더 내에
.jar 파일이 생성되었는지 확인



BACKEND 배포 - Spring

.jar 파일 이동

EC2 인스턴스로 jar파일 복사

Nginx conf 포트 확인

Nginx 설정 파일의 포트가
spring에서 설정한 포트와
일치하는지 확인

```
location /api {
    proxy_pass http://localhost:8399/api/;
    proxy_redirect off;
    charset utf-8;

    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-NginX-Proxy true;
}
```

.jar 실행

해당 폴더로 이동 후
java -jar 명령어로 jar파일 실행

```
$ sudo java -jar <파일명>.jar
```

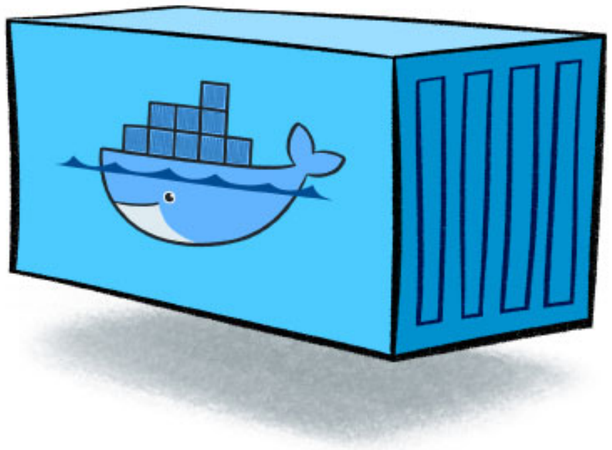
java 명령어가 실행되지 않을 경우
해당 패키지 설치 후 실행

jar 파일을 실행하면 내부에 있던
Tomcat 서버가 구동됨

배포 확인

http://<도메인>:<포트번호>/swagger-ui.html
입력 후 swagger 화면이 나오는지 확인

DOCKER CONTAINER



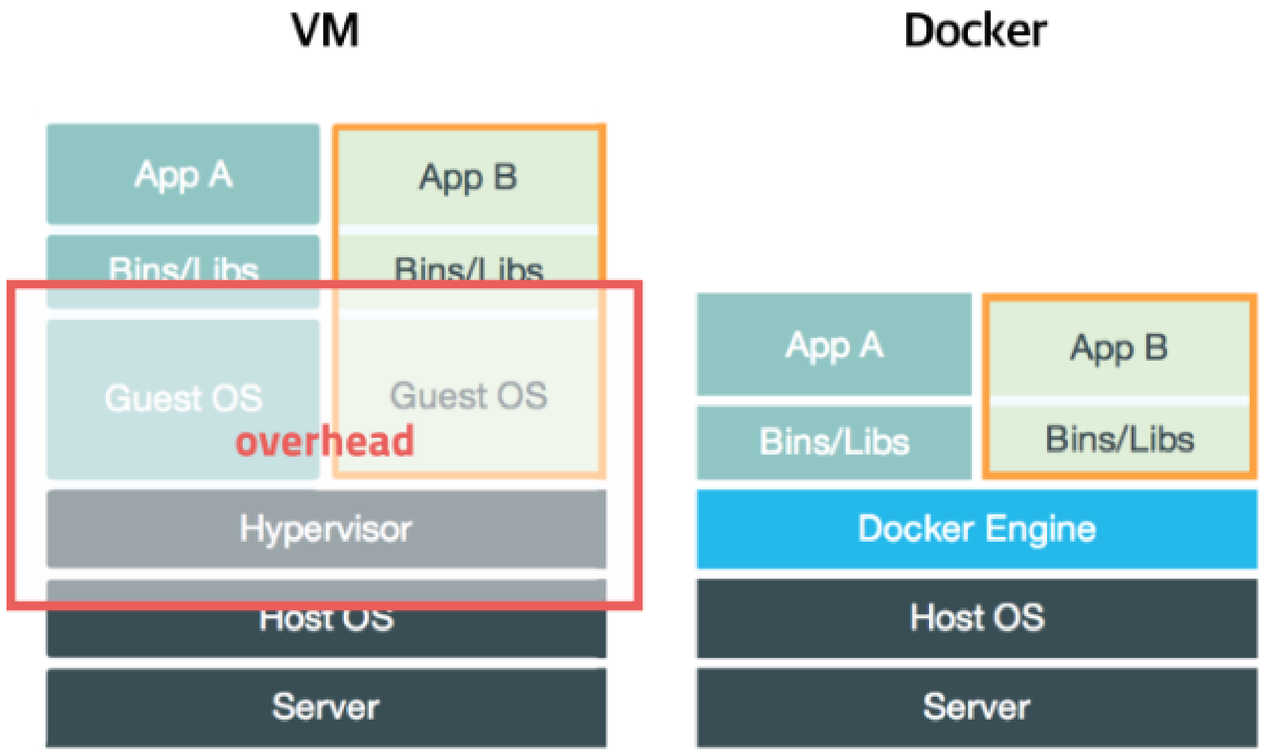
docker container

컨테이너 기반의 오픈소스 가상화 플랫폼

가상화 기술 중의 하나로 OS에서
프로세스를 격리하여 가볍고 빠르게 동작

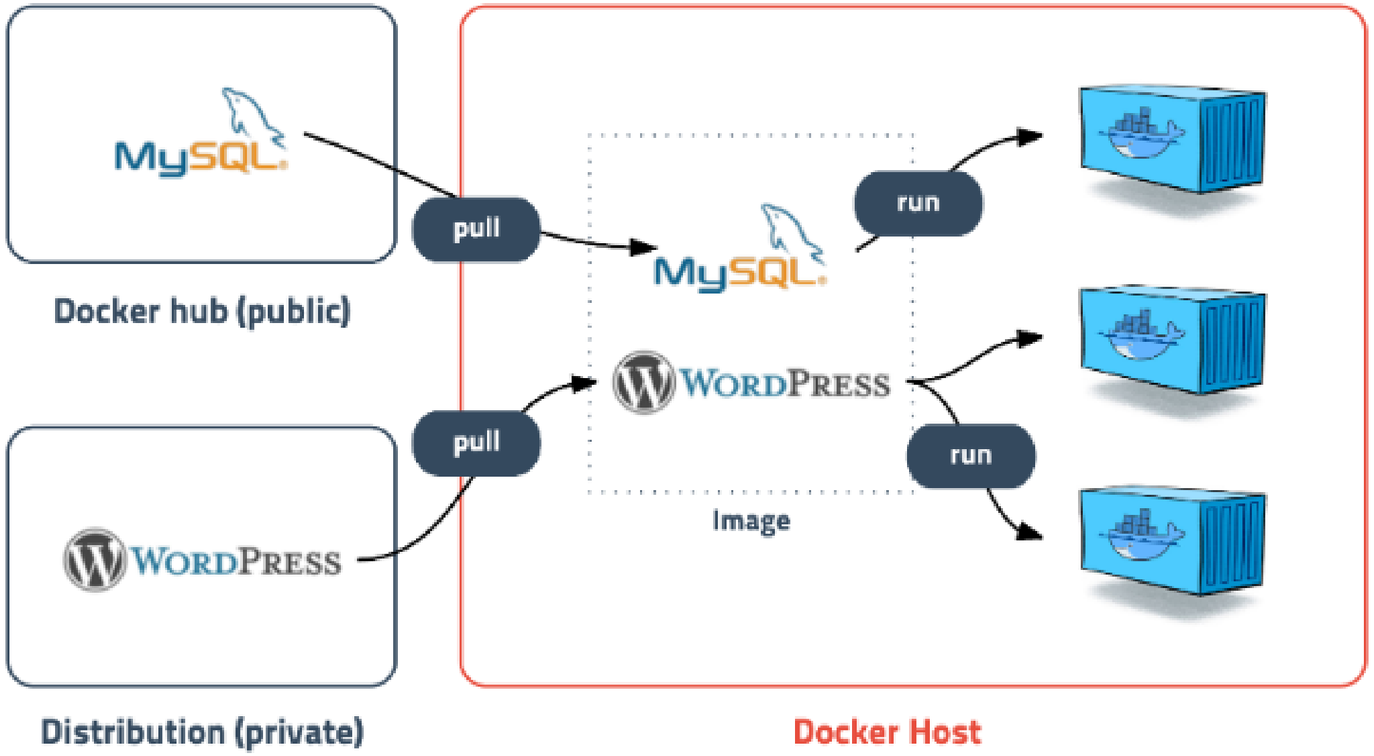
기존의 가상화 방식은 주로 OS 자체를 가상화
(VMWare, VMBox)

DOCKER CONTAINER



가상머신과 도커

DOCKER IMAGE



Docker image

Docker Image

컨테이너 실행에 필요한 파일과 설정값 등을
포함하고 있는 것

상태값을 가지지 않고 변하지 않음
(컨테이너의 상태가 변하여도
이미지는 그대로 존재)

도커 이미지는 Docker hub에 등록하거나
Docker Registry 저장소를 직접 만들어 관리

DOCKER - 도커는 왜 핫한가?

이미 존재하던

오버레이 네트워크, 유니온 파일 시스템 등의 기술들을

잘 조합하고 사용하기 쉽게 하여

사용자의 요구사항들을 충족시킴

DB 연결 - Docker 설치

Docker 공식 GPG 키와
저장소 추가

```
$ 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"
```

Docker CE 설치

```
$ sudo apt-get update && sudo apt-get install docker-ce
```

DB 연결 - MariaDB 컨테이너 실행

컨테이너 실행

```
$ docker run --name maria-db -p 3306:3306 -e MYSQL_ROOT_PASSWORD={패스워드} -d mariadb
```

컨테이너 내부에서 mariaDB 접속

```
$ docker exec -it maria-db mysql -u root -p
```

-it 옵션을 이용하여 실행하는 컨테이너 내부의 표준 입출력을 사용

mysql 진입 후 명령어를 통해 스키마 구성

DB 연결 - Spring 설정

pom.xml

```xml
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>

<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <scope>runtime</scope>
</dependency>
```

resources/application.properties

```
spring.datasource.driverClassName=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/ssafy?autoReconnect=true&useUnicode=true&characterEncoding=utf8
spring.datasource.username=root
spring.datasource.password=<패스워드>
```

THANK YOU

PRESENTATION FOR YOUR PROJECT

SSAFY

SAMSUNG SOFTWARE ACADEMY FOR YOUTH