산출물 정리

대충 목차 및 링크

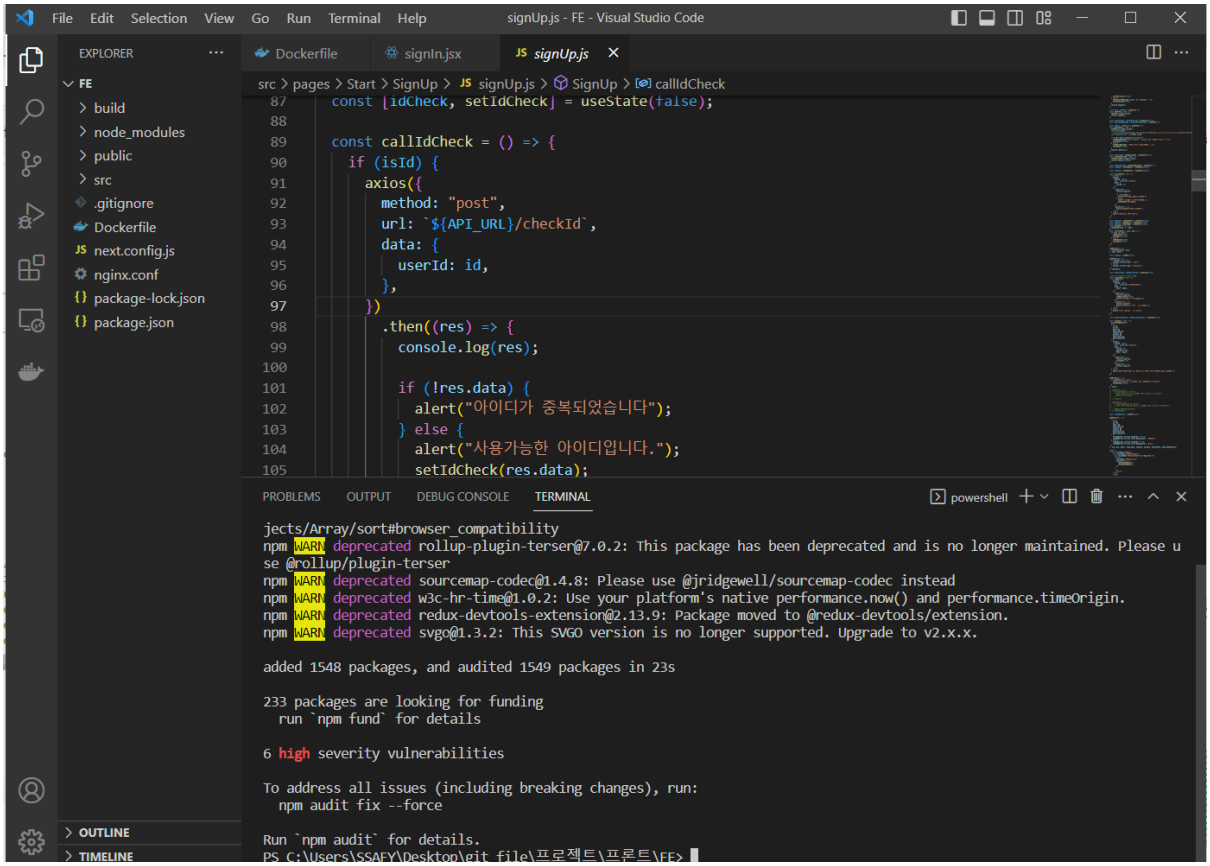
기술스택

1. 작업형상관리 : Jira
2. 형상관리 : GItlab
3. 메신져 : Mattermost
4. 개발환경
    a. OS : Window 10
    b. IDE
        i. Visual Studio Code
        ii. Spring Tool Suite(STS): 3.914
        iii. MySQL Workbench: 8.0.31
    c. 데이터베이스
        i. Mysql 8.0.31
        ii. Redis 3.0.504
    d. 프론트엔드
    e. 백엔드
        i. Java : Jdk-11
        ii. Gradel : 7.6
        iii. Spring : 2.7.8
        iv. SpringSecurity : 5.7.6
        v. Spring-Cloud-Netfilx-Eureka : 3.1.4
        vi. SpringSecurityOauth 2.0
    f. 서버
        i. Amazon EC2
        ii. Ubuntu 20.04 LTS
        iii. Docker 20.10.17
        iv. Jenkins 2.346.1 LTS
        v. Amazon S3
        vi. nginx 1.18.0

빌드 상세

프론트엔드

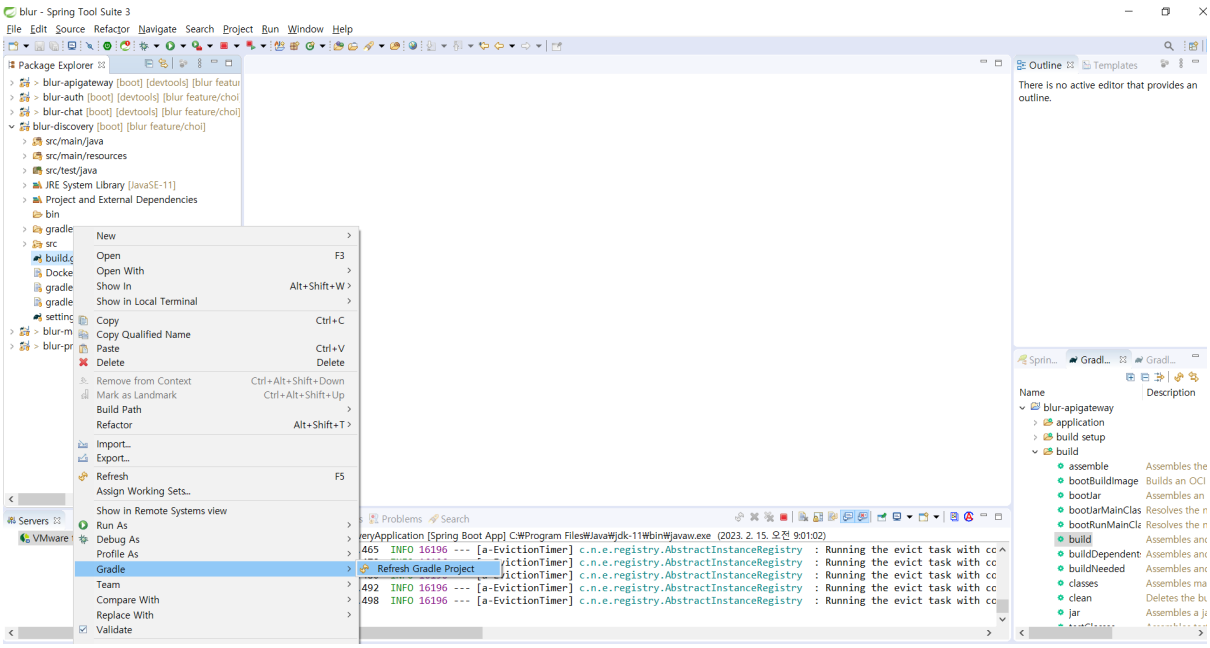
1. npm i 실행 후 node_moudles 생성



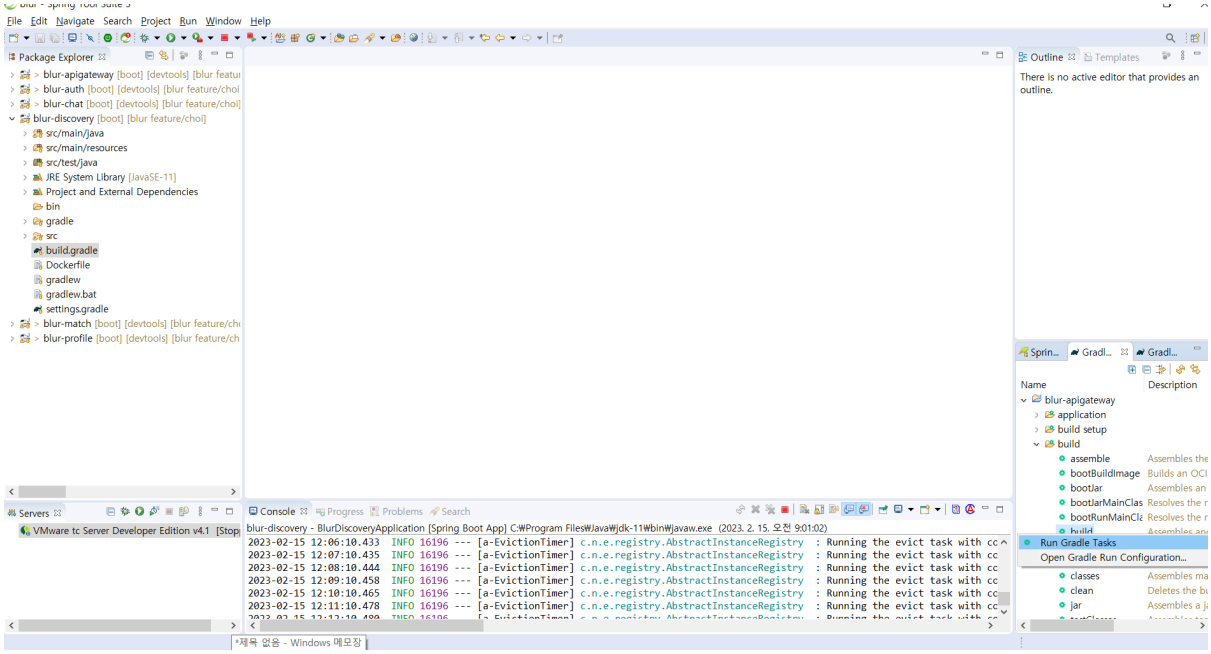
2. 로컬에서 실행할 경우 터미널에서 ‘npm start’ 입력
   배포환경에서 빌드할 경우 ‘npm run build’ 입력

백엔드

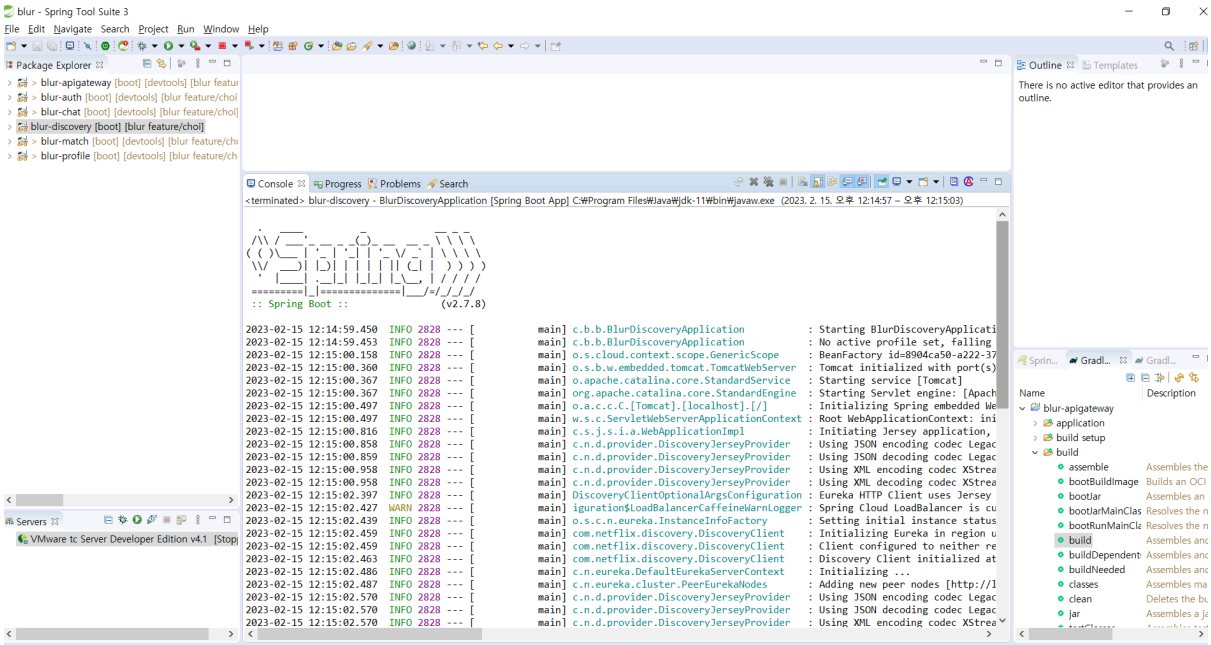
1. bulid.gradle 우클릭 및 Refresh Gradle Project 우클릭



2. GradleTask에서 빌드 실행



3. 프로젝트 실행



EC2에 도커 설치

```
#설치 가능한 리스트 업데이트
sudo apt-get update

sudo apt-get install -y \
apt-transport-https \
curl \
ca-certificates \
software-properties-common
```

```
#docker의 공식 GPG(GNU Privacy Guard) key를 추가
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

#추가된 키 + 도커에서 배포한 키확인
sudo apt-key fingerprint
sudo apt-key fingerprint 0EBFCD88

#debian 계열의 docker repository 추가 후 apt update 수행
sudo add-apt-repository \
"deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

sudo apt-get update
#docker CE 버전 설치
sudo apt-get install docker-ce

#sudo 없이 도커 사용
sudo usermod -aG docker zeff
sudo systemctl enable docker
sudo systemctl restart docker
sudo reboot
```

nginx 설치 및 ssl 설정

```
#nginx 설치
sudo apt-get update
sudo apt-get install nginx
sudo systemctl start nginx
sudo systemctl status nginx

# 무료 인증서인 letsencypt 설치 및 인증서 발급

sudo apt-get install letsencrypt

sudo systemctl stop nginx

sudo letsencrypt certonly --standalone -d www제외한 도메인 이름

# nginx 설정파일
cd /etc/nginx/site-available
sudo vi default
# sudo vi default 명령어를 통해서 접속후 i키를 눌러 편집 가능/ 저장할 때는 :wq하면 저장됨
server {
    # https 통신을 위한 ssl 인증서 적용
    listen 443 ssl;
    server_name i8b307.p.ssafy.io;
    ssl_certificate /etc/letsencrypt/live/i8b307.p.ssafy.io/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/i8b307.p.ssafy.io/privkey.pem;

    location / {
        proxy_pass http://localhost:3000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        error_page 405 = $uri;
    }

    location /blur-match {
        proxy_pass http://localhost:8000/blur-match;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        error_page 405 = $uri;
    }

    location /blur-auth {
        proxy_pass http://localhost:8000/blur-auth;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        error_page 405 = $uri;
    }

    location /blur-chat {
        proxy_pass http://localhost:8000/blur-chat;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        error_page 405 = $uri;
    }

    location /blur-profile {
        proxy_pass http://localhost:8000/blur-profile;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        error_page 405 = $uri;
```

```
    }
    location /oauth2 {
        proxy_pass http://localhost:9999/oauth2;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        error_page 405 = $uri;
    }

    location /login {
            proxy_pass http://localhost:9999/login;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            error_page 405 = $외부비스
```

MYSQL 도커 컨테이너 생성 및 실행

```
# mysql 도커 컨테이너 샐상
docker run –name mysql-auth -p 3307:3306 -e MYSQL_ROOT_PASSWORD=본인이 원하는 비밀번호 \
-d mysql:8.0.3 –character-set-server=utf8 –collation-server=utf8_unicode_ci

#mysql 컨테이너 접속
docker exec -it mysql-auth /bin/sh
mysql -u root -p
Enter password: MYSQL_ROOT_PASSWORD로 설정한 비밀번호


# userid를 가진 유저와 '비밀번호'를 패스워드 설정
create user 'userid'@'%' identified by '비밀번호';
# userid에게 비밀번호 설정과 모든 권한 부여
grant all privileges on *.* to userid@'%' identified by '비밀번호' with grant option;


# 도커 컨테이너를 생성하는 명령어들 / 각각 생성 후에 위에서 했던  과정을 반복하면 됩니다.
# 도커 컨테이너 접속하고 접근권한 부여 등
docker run –name mysql-chat -p 3308:3306 -e MYSQL_ROOT_PASSWORD=본인이 원하는 비밀번호 \
-d mysql:8.0.3 –character-set-server=utf8 –collation-server=utf8_unicode_ci

docker run –name mysql-match -p 3309:3306 -e MYSQL_ROOT_PASSWORD=본인이 원하는 비밀번호 \
-d mysql:8.0.3 –character-set-server=utf8 –collation-server=utf8_unicode_ci

docker run –name mysql-profile -p 3310:3306 -e MYSQL_ROOT_PASSWORD=본인이 원하는 비밀번호 \
-d mysql:8.0.3 –character-set-server=utf8 –collation-server=utf8_unicode_ci
```

EC2서버 에서 배포 시작

```
# EC2 서버에 git 설치
sudo apt-get install git
git clone (배포 레포지토리 url)

# 이후 각각 도커파일이 있는 디렉토리에 cd를 이용해서 진입하고 도커 여는 명령어 실행

#Frontend
# FE폴더에까지 들어온 후
docker build -t nginx-vue .
docker stop nginx_vue
docker run --name nginx_vue -d -p 3000:80 nginx-vue

#Backend

# 마찬가지로 각각 서비스의 도커파일이 있는 위치에서 서비스에 맞는 명령어를 입력한다.
docker build --tag blur-apigateway .
docker build --tag blur-discover .
docker build --tag blur-auth .
docker build --tag blur-chat .
docker build --tag blur-match .
docker build --tag blur-profile .

# docker 컨테이너 세우기 docker run 명령어는 현재 위치와 관계없이 작동한다.

docker run -d -p 8761:8761 --network ecommerce-network --name blur-discover -e "eureka.client.serviceUrl.defaultZone=http://blur-disco
docker run -d -p 8000:8000 --network ecommerce-network --name blur-apigateway -e "eureka.client.serviceUrl.defaultZone=http://blur-dis
docker run -d -p 9999:9999 --network ecommerce-network --name blur-auth -e "eureka.client.serviceUrl.defaultZone=http://blur-discover:
docker run -d --network ecommerce-network --name blur-chat -e "eureka.client.serviceUrl.defaultZone=http://blur-discover:8761/eureka/"
docker run -d --network ecommerce-network --name blur-match -e "eureka.client.serviceUrl.defaultZone=http://blur-discover:8761/eureka/
docker run -d --network ecommerce-network --name blur-profile -e "eureka.client.serviceUrl.defaultZone=http://blur-discover:8761/eurek
```

```
# 여기까지 완료하면 본인이 설정한 도메인 주소로 접속하면 정상 작동이 될 것이다.
```

외부서비스

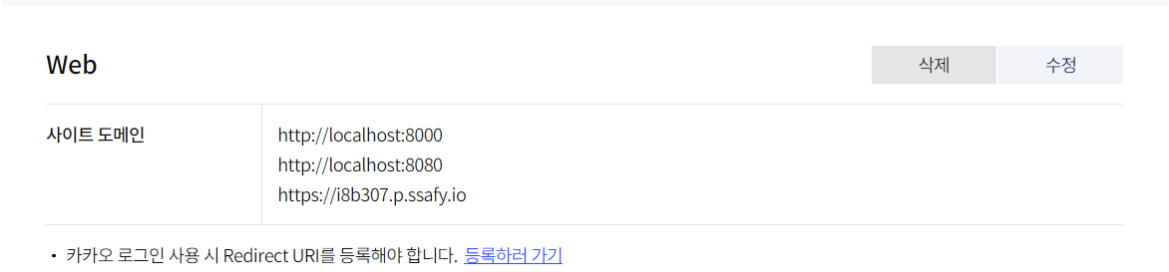
소셜로그인

- 서비스의 로그인을 카카오를 통해서도 가능합니다.
- 소셜 로그인에서 사용자의 이메일을 확인하고, 해당 이메일로 가입한 사용자가 있는지 확인합니다. 만약 사용자가 있으면 해당 계정으로 로그인하고 회원가입이 완료됩니다.

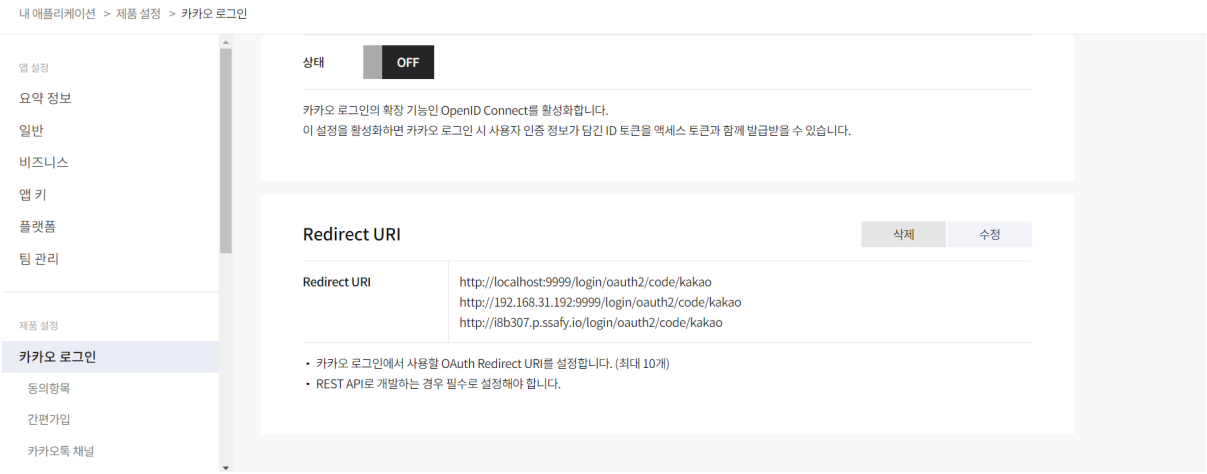
카카오 로그인

1. 애플리케이션 추가

2. 도메인 등록

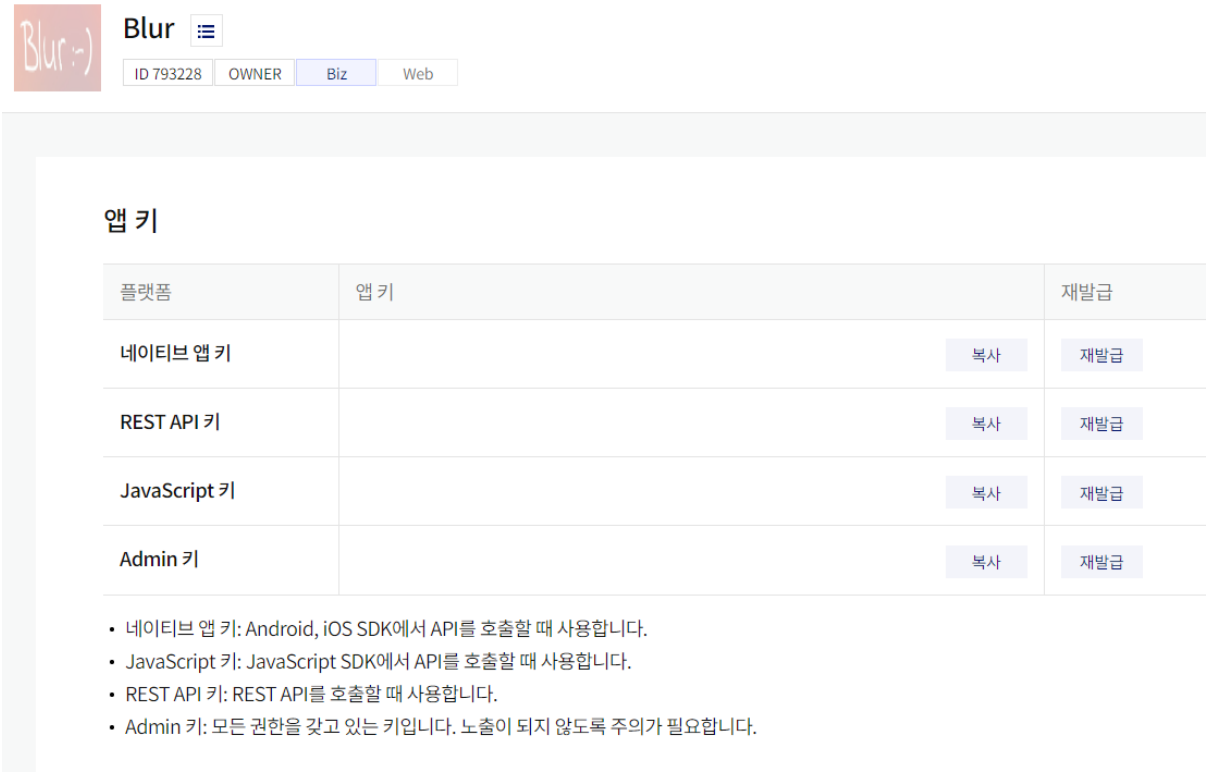


3. Redirct URL 설정



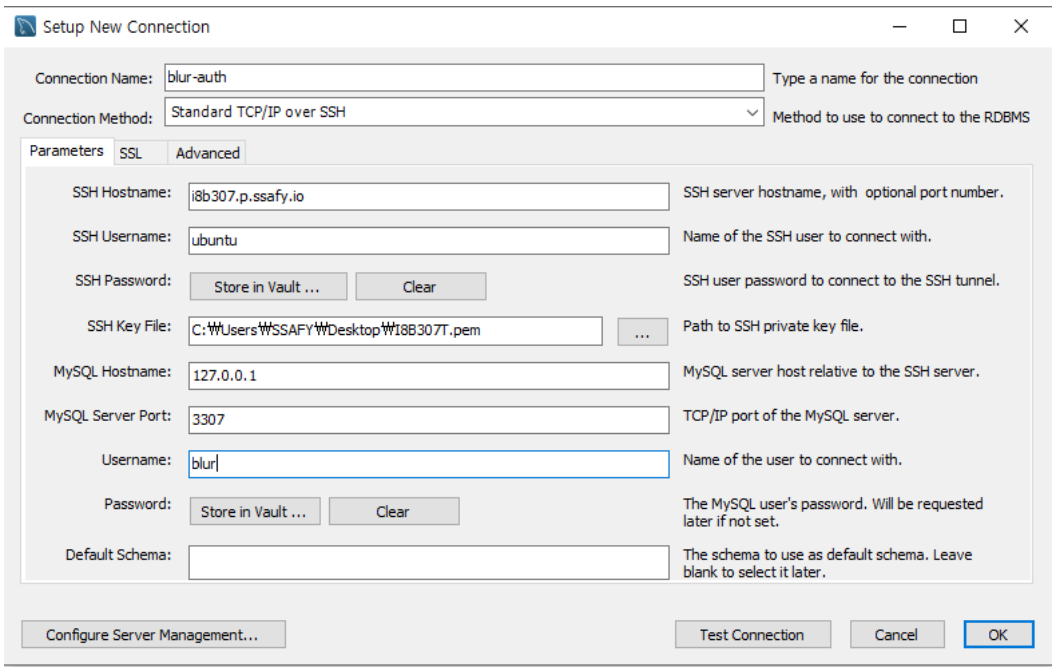
4. 로그인 활성화

5. 카카오 앱 키 확인



DB

Mysql 워크벤치 사용방법



1. 배포서버 도메인 입력
2. SSH key 등록
3. 서버 포트 설정
4. Mysql 설정에서 변경한 username 입력

5. Mysql에서 설정한 password 입력