



campinity

포팅 메뉴얼

구미1반 D101조

목차

1. 개발환경
2. 배포서버 환경 구축(CI/CD)
3. DB 환경 설정
4. 빌드 및 로컬환경 테스트 방법
5. Kakao 로그인 설정



1. 개발 환경

형상 관리

- Gitlab

이슈 관리

- Jira

CI/CD

- jenkins(2.375.2)
- Docker

IDE

- IntelliJ 2022 1.3
- Android Studio

Server

- AWS EC2
 - Ubuntu 20.04LTS
 - Docker 20.10.18

Communication

- Notion
- Discord
- Google Meet

UI/UX

- Figma

OS

- Window 10
- MacOS Monterey

DataBase

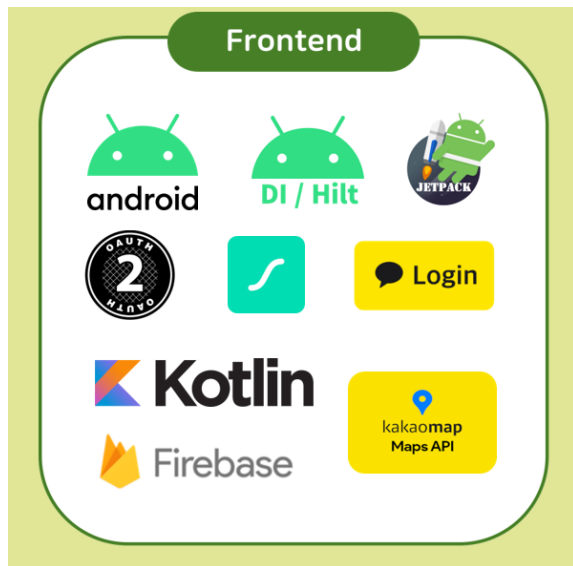
- Ver 15.1 Distrib 10.10.2-MariaDB
- MySQL WorkBench 6.X
- MongoDB
- Redis

기타 편의 툴

- PostMan 9.28.1
- WireShark

Front-End

- LiveData
- ViewModel



- DataBinding
- ViewPager2
- OkHttp3
- Retrofit2
- Coroutine
- Navigation
- Glide
- Hilt
- Version catalog

Back-End



- Java Open-JDK 11.0.15
- Gradle 8.0
- Spring Boot 2.7.7
 - Spring Data JPA
 - Lombok
 - Swagger 3.0.0
- Spring Security 5.7.6
- JUnit5 5.8.1
- Firebase 9.1.1
- Websocket 2.6.2
- Spring Batch



2. 배포서버 환경 구성(CI/CD)

목차

[Docker 및 docker-compose 설치](#)
[도커 실행](#)
[젠킨스 설치 및 실행](#)
[젠킨스 초기 설정](#)
[Jenkins 플러그인 추가 설치](#)
[젠킨스 Credential등록](#)
[젠킨스 세부 설정](#)
[Gitlab WebHook설정](#)
[Jenkins 설정 후 서버 빌드 및 배포 방법](#)

▼ Docker 및 docker-compose 설치

1. 패키지 업데이트 진행

```
sudo apt update & apt upgrade
```

2. Docker 설치에 필요한 필수 패키지 설치

```
sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common
```

3. Docker의 GPG key 인증

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

4. Docker Repository등록

```
sudo add-apt-repository \
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \
stable"
```

5. Docker 설치

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

6. docker-compose 설치

```
sudo curl -L \
"https://github.com/docker/compose/releases/download/1.28.5/dockercompose-$(uname -s)-$(uname -m)" \
-o /usr/local/bin/docker-compose
# 이 명령어는 외부에서 그대로 파일을 가져와서 현재의 시스템에 올려 놓는 것이다.
# 결과적으로 "/usr/local/bin/" 경로에 "docker-compose" 라는 이름의 파일로 다운로드.
# 참고) https://github.com/docker/compose/releases 에서 최신 버전 확인이 가능하다.
# 최신 버전을 설치하고 싶다면 위 명령어에 보이는 1.28.5 라는 버전 숫자를 바꿔주면 된다!

sudo chmod +x /usr/local/bin/docker-compose # chmod 를 통해서 실행이 가능하게 세팅

docker-compose -v # docker-compose 명령이 제대로 먹히는 지 확인한다.
```

▼ 도커 실행

```
sudo systemctl enable docker
```

```
sudo service docker start
```

▼ 젠킨스 설치 및 실행

1. Dockerfile을 만들어서 jenkins 이미지를 활용해 docker.sock을 활용해 통신가능하도록 컨테이너를 띄워준다.

```
FROM jenkins/jenkins:lts

USER root

RUN apt-get update \
  && apt-get -y install lsb-release \
  && curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg \
  && echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/debian $" \
  && apt-get update \
  && apt-get -y install docker-ce docker-ce-cli containerd.io
RUN usermod -u 1000 jenkins && \
  groupmod -g 998 docker && \
  usermod -sg docker jenkins
```

- 이때 주의할점은 아래의 옵션들이다. usermod -u {호스트의사용자아이디} groupmod -g {호스트의 도커 그룹 아이디}
- ubuntu host에서 호스트의 사용자 아이디를 가져오려면 `id -u` 명령어를 활용하고, 도커 그룹 아이디를 가져오고 싶다면 `cat /etc/group | grep docker` 를 사용하면 된다.

2. 이미지를 만든다

```
docker build -t my-jenkins:0.1 .
```

2. 이미지 확인

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
my-jenkins	0.1	d37227db069f	32 minutes ago	985MB

2. volume 만들기

```
docker volume create jenkins
```

3. volume 확인 (docker volume ls)

DRIVER	VOLUME NAME
local	jenkins

6. jenkins 컨테이너 실행

```
docker run -d --name jenkins \
  -v /var/run/docker.sock:/var/run/docker.sock \
  -v jenkins:/var/jenkins_home \
  -p 8080:8080 my-jenkins:0.1
```

▼ 젠킨스 초기 설정

1. <http://서버아이피:8080>으로 접속 한다.

2. 젠킨스에 처음 접속하면 초기 관리자 계정의 비밀번호를 입력하라고 나온다.

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (**not sure where to find it?**) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Continue

3. 우선 jenkins 컨테이너에 접속한다

```
docker exec -it jenkins /bin/bash
```

4. 초기 관리자 비밀번호를 확인한다

```
cat /var/jenkins_home/secrets/initialAdminPassword
```

5. 비밀번호를 입력하면 아래와 같은 화면이 나오는데, Install suggested plugins 클릭

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.289

6. 그대로 쪽 설치진행 하면 아래와 같이 진행이 된다, 설치가 모두 완료되면 관리자의 아이디 패스워드 설정하는창이 나오고 본인이 설정하고싶은 패스워드로 설정하면된다.

Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding	Workspace Cleanup Ant ** JavaScript GUI Lib: ACE Editor bundle ** JavaScript GUI Lib: jQuery bundles (jQuery and jQuery UI) ** Pipeline: SCM Step ** Pipeline: Groovy ** Pipeline: Job ** Apache HttpComponents Client 4.x API ** Display URL API Mailer ** Pipeline: Basic Steps Gradle ** Pipeline: Milestone Step ** Jackson 2 API ** Pipeline: Input Step ** Pipeline: Stage Step ** Pipeline: Graph Analysis ** Pipeline: REST API ** JavaScript GUI Lib: Handlebars bundle ** JavaScript GUI Lib: Moment.js bundle Pipeline: Stage View ** Pipeline: Build Step ** Pipeline: Model API ** - required dependency
✓ Timestampers	✓ Workspace Cleanup	✓ Ant	✓ Gradle	
🔄 Pipeline	🔄 GitHub Branch Source	🔄 Pipeline: GitHub Groovy Libraries	✓ Pipeline: Stage View	
🔄 Git	🔄 Subversion	🔄 SSH Slaves	🔄 Matrix Authorization Strategy	
🔄 PAM Authentication	🔄 LDAP	🔄 Email Extension	✓ Mailer	

▼ Jenkins 플러그인 추가 설치

Dashboard > Jenkins 관리 > Plugin Manager

Updates
 Available plugins
 Installed plugins
 Advanced settings

Plugins

Install	Name ↓	Released
---------	--------	----------

설치할 플러그인 목록

- Gitlab
- Gradle(이미 설치되어 있다면 설치 안해줘도 됨)

플러그인 검색창에서 Gitlab, Gradle 검색 후 설치해준 뒤 Jenkins를 재시작 시켜준다.

설치완료가 되면 Jenkins admin페이지 하단에 restart Jenkins라는 버튼이 생기고, 눌러주면 자동으로 재시작이 완료된다.

그러나 가끔 재시작이 정상적으로 되지 않는 경우가 생기는데 그때는 EC2에 원격접속 후 Jenkins 컨테이너를 아래와 같은 명령어를 사용해 재실행해준다.

```
docker restart {container_id 또는 container name}
```

▼ Jenkins Credential등록

1. Jenkins관리 → Manage Credentials에 들어간다

Security



Configure Global Security

Secure Jenkins; define who is allowed to access/use the system.



Manage Credentials

Configure credentials



Configure Credential Providers

Configure the credential providers and types



Manage Users

Create/delete/modify users that can log in to this Jenkins.

2. System 선택

Credentials

T	P	Store ↓	Domain
		System	(global)

3. 우측 상단에 Add Credentials 클릭

Global credentials (unrestricted)

[+ Add Credentials](#)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
ubermen5ch1308	ubermen5ch1308/***** (gitlab)	Username with password	gitlab

4. 우선 Gitlab 프로젝트 Repository에 들어가서 accesstoken을 발급받는다

Campinity

Project information

Repository

Issues 0

Merge requests 0

CI/CD

Security & Compliance

Deployments

Packages and registries

Infrastructure

Monitor

Analytics

Wiki

Snippets

Settings

General

Integrations

Webhooks

Access Tokens

s08-webmobile4-sub2 > Campinity > Access Tokens

Search page

Project Access Tokens

Generate project access tokens scoped to this project for your applications that need access to the GitLab API.
You can also use project access tokens with Git to authenticate over HTTP(S). [Learn more.](#)

Add a project access token

Enter the name of your application, and we'll return a unique project access token.

Token name

For example, the application using the token or the purpose of the token. Do not give sensitive information for the name of the token, as it will be visible to all project members.

Expiration date

2023-03-18

Select a role

Guest

Select scopes

Scopes set the permission levels granted to the token. [Learn more.](#)

☐ api
Grants complete read and write access to the scoped project API, including the Package Registry.

☐ read_api
Grants read access to the scoped project API, including the Package Registry.

☐ read_repository

5. 발급 받은 accessToken을 Credential을 만들때 password로 입력해주고 나머지 부가 정보들도 입력해주고 Create해준다

New credentials

Kind
Username with password

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

Username ?
ubermer5ch1308 **Gitlab username**

☐ Treat username as secret ?

Password ?
Project accesstoken입력

ID ?
Gitlab ID입력

Description ?

Create


▼ 젠킨스 세부 설정


1. 젠킨스 로그인을 완료하고 새로운 Item추가를 클릭한다.
2. item이름을 입력하고 Freestyle project를 선택한다.

Enter an item name

campinity-develop2

» Required field

 **Freestyle project**
이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프 트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.

 **Pipeline**

3. 소스 코드 관리 목록중 Git을 선택하고 Repository URL에 Gitlab프로젝트 URL을 입력하고 이전에 설정해둔 Credential로 선택 해준다.

소스 코드 관리

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://lab.ssafy.com/s08-webmobile4-sub2/S08P12D101.git (깃랩 repository URL)

! Please enter Git repository.

Credentials ?

- none - GitLab repository token을 기반으로 만든 Credential등록

+ Add

고급...

4. 바로 아래에 어떤 브랜치에서 commit들고와서 Integration 시킬지 branch를 명시해준다. (deploy-be로 설정)

Branches to build ?

Branch Specifier (blank for 'any') ?

*/deploy-be

Add Branch

5. 빌드 유발 설정 부분을 아래와 같이 설정해준다 .

빌드 유발

☐ 빌드를 원격으로 유발 (예: 스크립트 사용) ?

☐ Build after other projects are built ?

☐ Build periodically ?

☒ Build when a change is pushed to GitLab. GitLab webhook URL: http://f8d101.p.ssafy.io:8080/project/campinity-develop2 ?

Enabled GitLab triggers

☐ Push Events

☐ Push Events in case of branch delete

☐ Opened Merge Request Events

☐ Build only if new commits were pushed to Merge Request ?

☒ Accepted Merge Request Events

☐ Closed Merge Request Events

Rebuild open Merge Requests

Never

☐ Approved Merge Requests (EE-only)

☐ Comments

Comment (regex) for triggering a build ?

Jenkins please retry a build

6. Build Steps 설정은 다음과 같다

Build Steps

≡ Invoke Gradle script ?

☒ Invoke Gradle ?

Gradle Version

gradle 8.0

☐ Use Gradle Wrapper ?

Tasks ?

build -p /var/jenkins_home/workspace/campinity-develop/Server -x test

고급...

≡ Execute shell ?

Command

See [the list of available environment variables](#)

```
cd Server
docker-compose up --build -d
```

저장

Apply

▼ Gitlab WebHook설정

1. Gitlab WebHook 탭에 들어가서 설정값들을 입력해준다.

Webhook

[Webhooks](#) enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

URL `jenkins-server-url/jenkins-item-name`

`http://i8d101.p.ssafy.io:8080/project/campinity-develop`

URL must be percent-encoded if it contains one or more special characters.

Secret token

.....

Used to validate received payloads. Sent with the request in the `X-Gitlab-Token` HTTP header.

Trigger

☐ Push events

`deploy-be`

Push to the repository.

☐ Tag push events

A new tag is pushed to the repository.

☐ Comments

A comment is added to an issue or merge request.

☐ Confidential comments

A comment is added to a confidential issue.

☐ Issues events

An issue is created, updated, closed, or reopened.

☐ Confidential issues events

A confidential issue is created, updated, closed, or reopened.

☒ Merge request events

A merge request is created, updated, or merged.

2. 설정값중 Secret token은 jenkins서버에 접속해서 item(campinity-develop) → 구성 → 빌드유발 → 고급 → Secret token → generate 클릭해서 토큰을 발급받고 입력해준다.

☒ Build when a change is pushed to GitLab. GitLab webhook URL: http://[redacted]/project/[redacted]

Enabled GitLab triggers

Push Events	<input checked="" type="checkbox"/>
Opened Merge Request Events	<input checked="" type="checkbox"/>
Accepted Merge Request Events	<input type="checkbox"/>
Closed Merge Request Events	<input type="checkbox"/>
Rebuild open Merge Requests	Never ▾
Approved Merge Requests (EE-only)	<input type="checkbox"/>
Comments	<input checked="" type="checkbox"/>
Comment (regex) for triggering a build	Jenkins ?

Enable [ci-skip] ☒

Ignore WIP Merge Requests ☒

Set build description to build cause (eg. Merge request or Git Push) ☒

Build on successful pipeline events ☐

Pending build name for pipeline

Cancel pending merge request builds on update ☐

Allowed branches

- ☒ Allow all branches to trigger this job ?
- ☐ Filter branches by name ?
- ☐ Filter branches by regex ?
- ☐ Filter merge request by label

Secret token [redacted] Generate

3. Gitlab WebHook으로 다시 돌아가서 WebHook test

SSL verification

☒ Enable SSL verification

Save changes

Test ▾

Delete

Status	Trigger	Elapsed time	Request time	
200	Merge Request Hook	0.02 sec	just now	View details

4.

▼ Jenkins 설정 후 서버 빌드 및 배포 방법

자동으로 빌드 및 배포

jenkins에 등록된 project URL에 Merge Request가 accept되면 자동으로 빌드 및 배포가 된다.

수동으로 빌드 및 배포

Dashboard > campinity-develop >

상태

</> 변경사항

작업공간

▶ 지금 빌드

구성

Project 삭제

Rename

Build History 추이 ▼

Project campinity-develop

고정링크

- [Last build, \(#225\), 25 min 전](#)
- [Last stable build, \(#225\), 25 min 전](#)
- [Last successful build, \(#225\), 25 min 전](#)
- [Last failed build, \(#131\), 5 days 15 hr 전](#)
- [Last unsuccessful build, \(#155\), 3 days 3 hr 전](#)
- [Last completed build, \(#225\), 25 min 전](#)

지금 빌드 버튼을 누르면 item 설정된 URL 및 branch를 jenkins가 인식하고 commit들을 fetch후 build 및 deploy를 진행한다.

참고 사항

위의 과정을 그대로 따라서 거쳐온다면, 서버가 제대로 작동하지 않을것이다. 그 이유는 MariaDB 컨테이너에 Database가 생성되지 않은 상태이기 때문이다.

따라서 EC2에 원격접속 후 DB관련 환경설정을 해주어야한다.

3. MariaDB 환경설정



3. MariaDB 환경설정



MariaDB Container에 접속 후 접속할 유저 생성

1. EC2 접속

```
ssh -i I8D101.pem ubuntu@i8d101.p.ssafy.io
```

2. MariaDB container 접속

```
docker exec -it database /bin/bash
```

3. MariaDB 서버에 root계정으로 접속

```
mysql -u root -p
```

4. User 등록

```
create user userid@접속할 host 외부아이피 identified by '비밀번호';
```

5. 권한 부여

```
GRANT ALL PRIVILEGES ON DB명.테이블 TO 계정아이디@host IDENTIFIED BY '비밀번호';
```

6. 권한 반영

```
flush privileges;
```




MariaDB 원격접속 후 Database 생성 및 데이터 삽입

1. MySQL Workbench로 접속

- MariaDB서버에서 root권한을 생성한 user의 username과 password를 설정해주고, hostname은 host의 아이피(Domain Name)를 입력해준다. 포트는 3306

Connection Name:

Connection Remote Management System Profile

Connection Method: Method to use to connect to the RDBMS

Parameters SSL Advanced

Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

Password: The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

2. 접속 후 project repository/exec 폴더에있는 backup.sql을 다운받아서 sql을 execute 해준다.



4. 빌드 및 로컬환경 테스트방법



환경 구성

- 운영체제 : Window 10
- 준비물 : IntelliJ, Docker Desktop, Git



빌드 및 테스트 방법

1. Repository clone

```
git clone --single-branch --branch develop-be https://lab.ssafy.com/s08-webmobile4-sub2/S08P12D101.git
```

2. batch application.yml 수정

```
~/S08P12D101/Server/demo-batch-server/src/main/resources/application.yml
```

위 경로에 있는 파일을 열어서 spring:profiles:active: local로 수정한다

```
server:
  port: 8002

spring:
  profiles:
    active: local # 기본 환경을 prod로 셋팅 local 개발할때는 local로 바꾸세요.

  batch-db:
    datasource:
      driver-class-name: org.mariadb.jdbc.Driver
  campinity-db:
    datasource:
      driver-class-name: org.mariadb.jdbc.Driver
```

3. core application.yml 수정

~/S08P12D101/Server/demo-core/src/main/resources/application.yml

위 경로에 있는 파일을 열어 spring:profiles:active: local로 수정한다

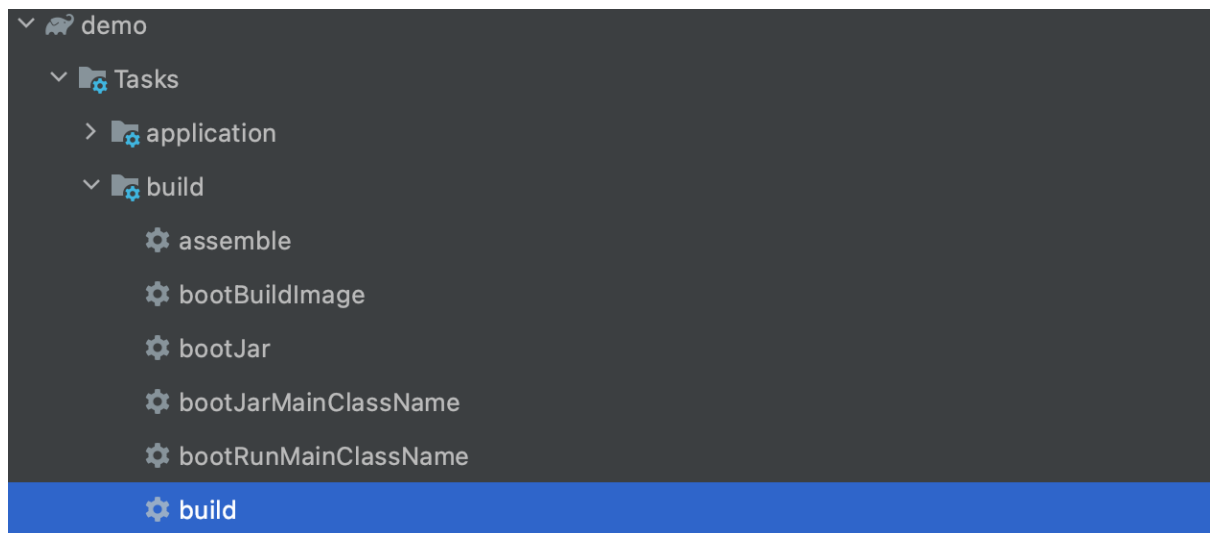
```
server:
  port: 8003

spring:
  profiles:
    active: local # 기본 환경을 prod로 셋팅 local 개발할때는 local로 바꾸세요.

hikari:
  connectionTimeout : 30000
  maximumPoolSize : 30
  maxLifeTime : 97
  poolName : HikariCP
  readOnly : false
```

4. Gradle을 이용해 빌드(jar파일 생성)

IntelliJ의 Gradle을 컨후 빌드



5. docker-compose로 container 띄우기

```

* ~/SSAFY/common_project/clone/S08P12D101/Server/ [develop-be*] docker-compose up -d
Creating network "server_default_bridge" with the default driver
Creating redis_boot ... done
Creating mongo ... done
Creating database ... done
Creating server_application_1 ... done
Creating server_batch-server_1 ... done
* ~/SSAFY/common_project/clone/S08P12D101/Server/ [develop-be*] docker ps

```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
1151c3c57e3a	server_application	"java '-Duser.timezone..."	3 seconds ago	Up 2 seconds	0.0.0.0:8003->8003/tcp, :::8003->8003/tcp	server_application_1
3f8530afd788	server_batch-server	"java -Duser.timezone..."	3 seconds ago	Up 2 seconds	0.0.0.0:8002->8002/tcp, :::8002->8002/tcp	server_batch-server_1
6febbefd20ec	redis:alpine	"docker-entrypoint.s..."	5 seconds ago	Up 3 seconds	0.0.0.0:6379->6379/tcp, :::6379->6379/tcp	redis_boot
9ad3d14bc70e	mariadb	"docker-entrypoint.s..."	5 seconds ago	Up 3 seconds	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp	database
3a62c67df135	mongo:4.4.3	"docker-entrypoint.s..."	5 seconds ago	Up 3 seconds	0.0.0.0:27017->27017/tcp, :::27017->27017/tcp	mongo

6. docker-compose로 container 띄우기

접속이 잘되는지 테스트해보기



5. Kakao 로그인 설정

Kakao Developers

카카오 API를 활용하여 다양한 어플리케이션을 개발해보세요. 카카오 로그인, 메시지 보내기, 친구 API, 인공지능 API 등을 제공합니다.

 <https://developers.kakao.com/>

kakao developers

1. 카카오 Developers에서 어플리케이션 추가

어플리케이션 추가하기

앱 아이콘

이미지
업로드

파일 선택

JPG, GIF, PNG
권장 사이즈 128px, 최대 250KB

앱 이름

내 어플리케이션 이름

사업자명

사업자 정보와 동일한 이름


- 입력된 정보는 사용자가 카카오 로그인을 할 때 표시됩니다.
- 정보가 정확하지 않은 경우 서비스 이용이 제한될 수 있습니다.

☒ [서비스 이용이 제한되는 카테고리](#), [금지된 내용](#), [금지된 행동](#) 관련 운영정책을 위반하지 않는 앱입니다.

취소

저장

2. 카카오 로그인 활성화



Campinity

ID 851334

EDITOR

Biz

Android

카카오 로그인 ON

동의 화면 미리보기

활성화 설정

상태

ON

3. Redirect URI등록

Redirect URI

Redirect URI

카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다.

여러개의 URI를 줄바꿈으로 추가해주세요. (최대 10개)

REST API로 개발하는 경우 필수로 설정해야 합니다.

예시: (O) <https://example.com/oauth> (X) <https://www.example.com/oauth>

```
http://localhost:8003/api/v4/members/login-kakao
http://i8d101.p.ssafy.io:8003/api/v4/members/login-kakao
```

취소

저장