

포팅 매뉴얼

▼ 1) 프로젝트 기술스택

Backend

- IntelliJ
- spring boot 2.7.7
- spring-boot-jpa
- Spring Security
- Java 11
- AWS EC2
- mysql 8.0.31
- redis 5.0.7
- S3

UI/UX

- Figma

Frontend

- Visual Studio Code
- React.js 18.2.0
- redux 4.2.1
- redux-persist: 6.0.0

Web RTC

- openvidu-browser: 2.25.0

ImageGame

- Teachable Machine 0.8.5

CI/CD

- aws ec2
- docker 20.10.23
- nginx
- jenkins

협업 툴

- Gitlab
- Jira
- Notion
- Mattermost
- Webex

▼ 2) 백엔드 빌드방법

버전

```
Spring boot : 2.7.7
zulu-jdk : 11.0.17
Spring Data JPA
Spring Security
Spring Web
Lombok
MySQL Driver
Spring Boot Dev Tools
OAuth2 Client
```

build.gradle

```
// build.gradle

//querydsl
buildscript {
    dependencies {
        classpath("gradle.plugin.com.ewerk.gradle.plugins.querydsl-plugin:1.0.10")
    }
}

plugins {
    id 'java'
    id 'org.springframework.boot' version '2.7.7'
    id 'io.spring.dependency-management' version '1.0.15.RELEASE'
}

group = 'com.ssafy'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '11'

apply plugin: "com.ewerk.gradle.plugins.querydsl" // querydsl
```

```

configurations {
    compileOnly {
        extendsFrom annotationProcessor
    }
    all {
        exclude group: 'org.springframework.boot', module: 'spring-boot-starter-logging'
    }
}

repositories {
    mavenCentral()
}

dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'
    implementation 'org.springframework.boot:spring-boot-starter-security'
    implementation 'org.springframework.boot:spring-boot-starter-web'
    implementation 'io.springfox:springfox-swagger-ui:3.0.0'
    implementation 'io.springfox:springfox-swagger2:3.0.0'
    compileOnly 'org.projectlombok:lombok'
    developmentOnly 'org.springframework.boot:spring-boot-devtools'
    runtimeOnly 'com.mysql:mysql-connector-j'
    annotationProcessor 'org.projectlombok:lombok'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
    testImplementation 'org.springframework.security:spring-security-test'
    implementation group: 'org.json', name: 'json', version: '20220924'
    implementation 'org.springframework.boot:spring-boot-starter-log4j2'

    // querydsl
    implementation 'com.querydsl:querydsl-jpa'
    implementation 'com.querydsl:querydsl-apt'

    // Openvidu 의존성 2.18.0 으로 해도됨
    implementation group: 'io.openvidu', name: 'openvidu-java-client', version: '2.25.0'

    //s3 의존성
    implementation 'org.springframework.cloud:spring-cloud-starter-aws:2.2.6.RELEASE'

    // redis
    implementation 'org.springframework.boot:spring-boot-starter-data-redis'
    implementation 'org.springframework.session:spring-session-data-redis'
}

tasks.named('test') {
    useJUnitPlatform()
}

//querydsl
def querydslDir = "$buildDir/generated/querydsl"

//querydsl
querydsl {
    library = "com.querydsl:querydsl-apt"
    jpa = true
    querydslSourcesDir = querydslDir
}

//querydsl
sourceSets {
    main {
        java {
            srcDirs = ['src/main/java', querydslDir]
        }
    }
}

//querydsl
compileQuerydsl{
    options.annotationProcessorPath = configurations.querydsl
}

//querydsl
configurations {
    querydsl.extendsFrom compileClasspath
}

```

src > main > resources 에 application-API-KEY.properties 추가

```
kakao-admin-key = 카카오키
```

로컬로 실행할 시

src > main > resources 에 Application.properties 추가

```
spring.mvc.view.prefix=/WEB-INF/views/
spring.mvc.view.suffix=.jsp

server.port=8081

# MySQL 설정
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
# # DB Source URL
spring.datasource.url=jdbc:mysql://localhost:3306/raonzena?useSSL=false&useUnicode=true&serverTimezone=Asia/Seoul
# DB username
spring.datasource.username=유저네임
# DB password
spring.datasource.password=유저비밀번호

# true 설정시 JPA 쿼리문 확인 가능
spring.jpa.show-sql=true
# DDL(create, alter, drop) 정의시 DB의 고유 기능을 사용할 수 있다.
spring.jpa.hibernate.ddl-auto=update
# JPA의 구현체인 Hibernate가 동작하면서 발생한 SQL의 가독성을 높여준다.
spring.jpa.properties.hibernate.format_sql=true

#s3 ?? ??
cloud.aws.credentials.accessKey=AWS accessKey
cloud.aws.credentials.secretKey=AWS secretKey
cloud.aws.stack.auto=false

# AWS S3 Service bucket
cloud.aws.s3.bucket=s3 bucket
cloud.aws.region.static=aws region static

# Redis configuration
spring.redis.host=서버주소
spring.redis.port=port번호
spring.redis.password=redis 비밀번호

spring.profiles.include=API-KEY
```

src > main > java > com.ssafy.raonzena > service > userServiceImpl

```
// userServiceImpl.java

...

// HttpBody 오브젝트 생성
MultiValueMap<String, String> params = new LinkedMultiValueMap<>();
params.add("grant_type", "authorization_code");
params.add("client_id", kakao_admin_key);//kakao rest-api 키
params.add("redirect_uri", "http://localhost:3000/oauth/kakao/callback");
params.add("code", authorizedCode);

...
```

서버로 배포할 시

src > main > resources 에 Application.properties 추가

```
spring.mvc.view.prefix=/WEB-INF/views/
spring.mvc.view.suffix=.jsp

server.port=8081

# MySQL 설정
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
# DB Source URL
spring.datasource.url=jdbc:mysql://i8a507.p.ssafy.io:3306/raonzena?useSSL=false&allowPublicKeyRetrieval=true&useUnicode=true&serverTimezone=Asia/Seoul
# DB username
spring.datasource.username=유저네임
# DB password
spring.datasource.password=유저비밀번호
```

```
# true 설정시 JPA 쿼리문 확인 가능
spring.jpa.show-sql=true
# DDL(create, alter, drop) 정의시 DB의 고유 기능을 사용할 수 있다.
spring.jpa.hibernate.ddl-auto=update
# JPA의 구현체인 Hibernate가 동작하면서 발생한 SQL의 가독성을 높여준다.
spring.jpa.properties.hibernate.format_sql=true
```

```
#S3 ?? ??
cloud.aws.credentials.accessKey=AWS accessKey
cloud.aws.credentials.secretKey=AWS secretKey
cloud.aws.stack.auto=false
```

```
# AWS S3 Service bucket
cloud.aws.s3.bucket=s3 bucket
cloud.aws.region.static=aws region static
```

```
# Redis configuration
spring.redis.host=서버주소
spring.redis.port=port번호
spring.redis.password=redis 비밀번호
```

```
spring.profiles.include=API-KEY
```

src > main > java > com.ssafy.raonzena > service > userServiceImpl

```
// userServiceImpl.java

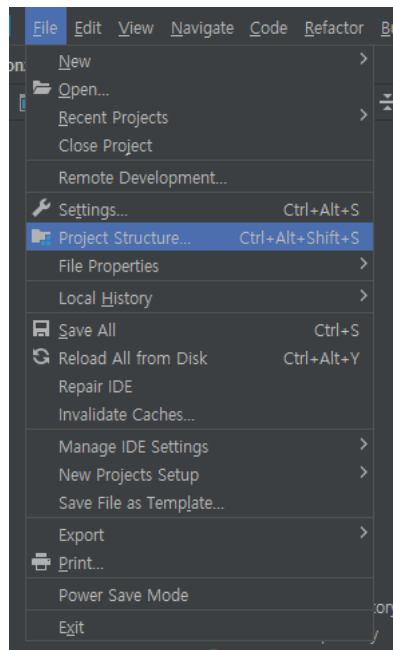
...


// HttpBody 오브젝트 생성
MultiValueMap<String, String> params = new LinkedMultiValueMap<>();
params.add("grant_type", "authorization_code");
params.add("client_id", kakao_admin_key); //kakao rest-api 키
params.add("redirect_uri", "https://i8a507.p.ssafy.io/oauth/kakao/callback");
params.add("code", authorizedCode);

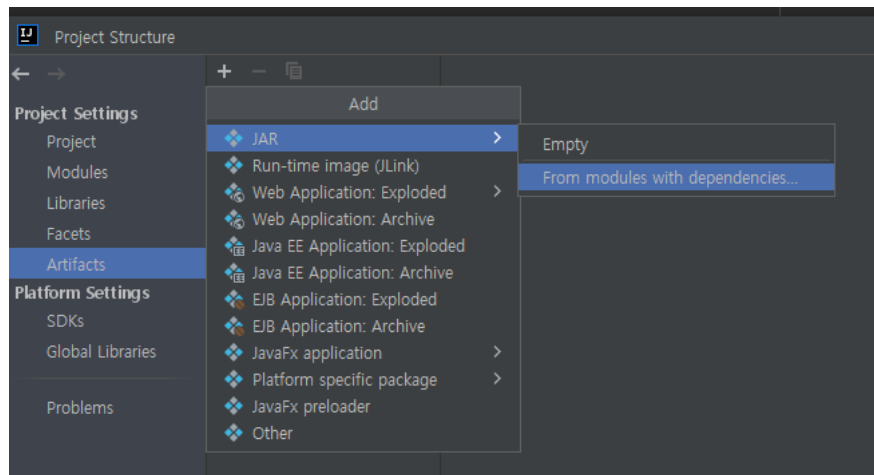
...
```

빌드 방법

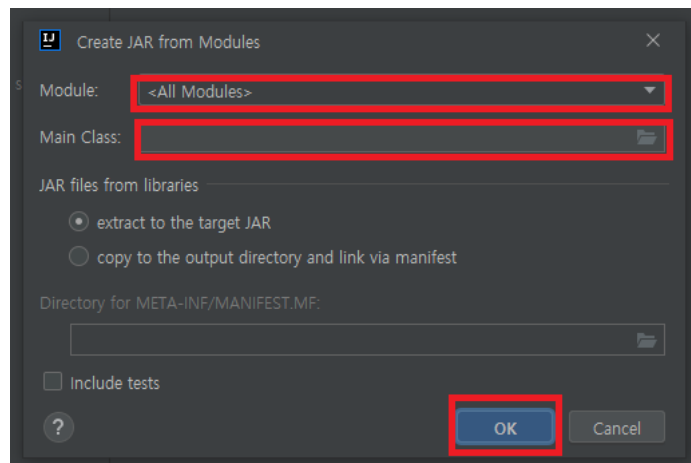
1. 상단 File > Project Structure 메뉴로 이동



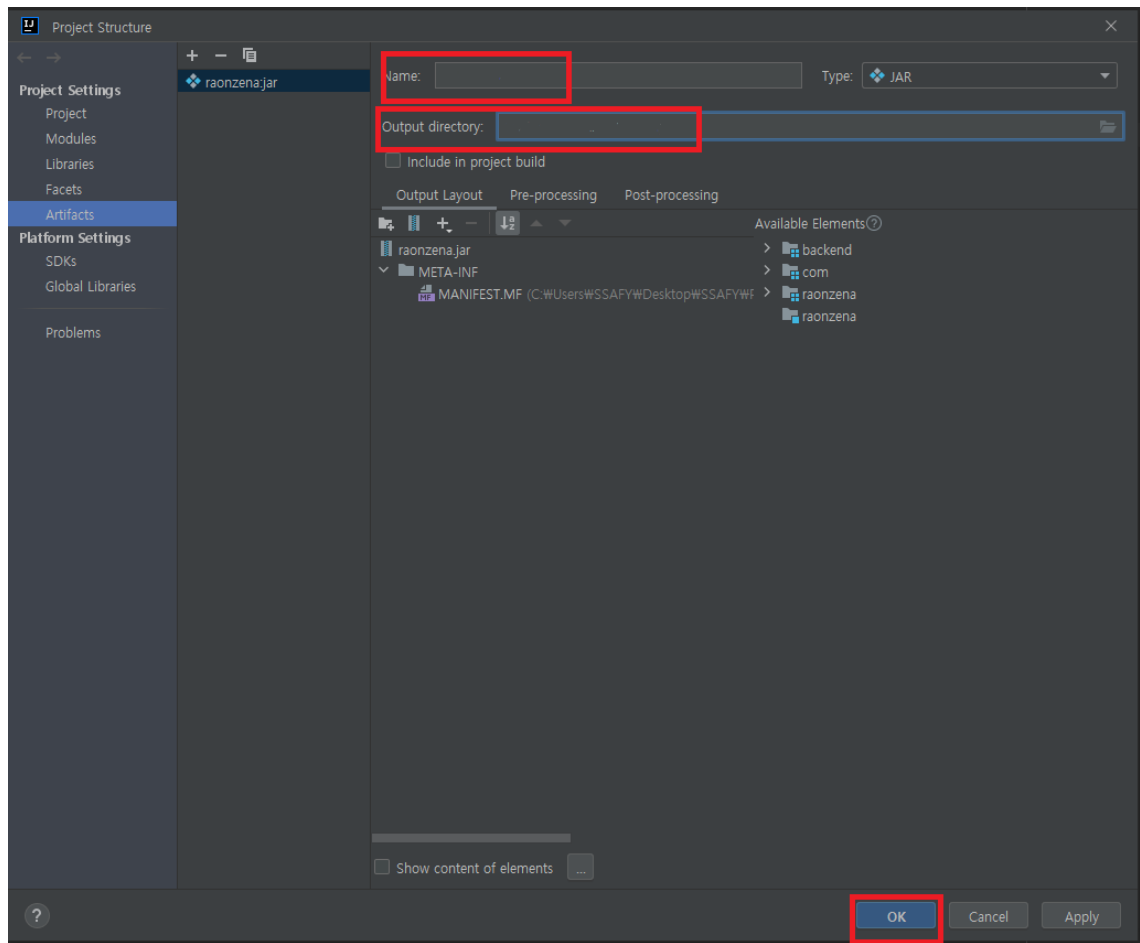
2. Artifacts 메뉴 >  버튼 > JAR > From modules with dependencies



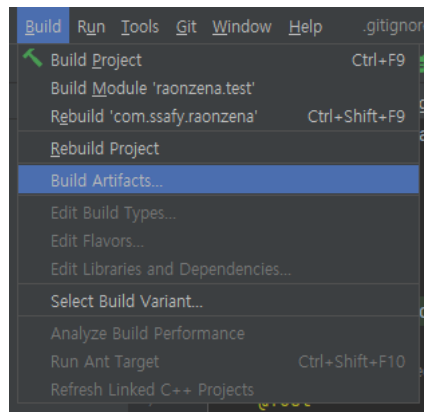
3. Main Class를 지정하고 **OK** 선택



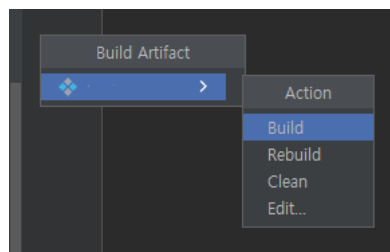
4. 파일명, Output directory를 확인 및 지정한 후, **OK** 선택



5. Build > Build Artifacts를 선택



6. Action에서 Build를 선택 하여 Jar를 생성



▼ 3) 프론트엔드 빌드방법

버전

```
node.js : 16.19.0
react: 18.2.0
axios: 1.3.3
react-redux: 8.0.5
react-router-dom: 6.8.1
```

패키지 설치

- 해당 경로 내 package.json 확인 후 설치

```
npm i
```

Root 폴더에 .env 파일 추가

```
// .env

// 카카오 API 키
REACT_APP_REST_API_KEY=카카오 api 키
```

로컬로 실행할 시

src/app/baseUrl.js

```
// src/app/baseUrl.js

...

const initialState = "http://localhost:8081/api/v1/"

...
```

src/app/redirectUrl.js

```
// src/aap/redirectUrl.js

...

const initialState = "http://localhost:3000"

...
```

서버로 배포할 시

src/app/baseUrl.js

```
// src/app/baseUrl.js

...

const initialState = "https://도메인주소/api/v1/"

...
```

src/app/redirectUrl.js

```
// src/aap/redirectUrl.js

...
```

```
const initialState = "https://도메인주소"
...
```

빌드방법

1. 터미널 창에서 아래 명령어 입력 → 빌드 폴더 생성

```
npm run build
```

▼ 4) EC2 세팅

EC2 인스턴스 접속

```
$ ssh -i [pem키 경로] [user]@[도메인] -p [포트]
```

패키지 관리자 업데이트

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

JDK 설치

```
$ sudo apt-get install openjdk-11-jdk
$ java -version
```

DB 설치

```
$ sudo apt-get install mysql-server
```

MySQL 유저생성 후 권한 설정

```
$ sudo mysql -uroot
$ create user '아이디'@'%'identified by '비밀번호'
$ grant all privileges on *.*to '아이디'@'%';
```

MySQL 외부 접속 허용

```
$ cd /etc/mysql/mysql.conf.d/
$ sudo vi mysqld.cnf
```

```
# /etc/mysql/mysql.conf.d/mysqld.cnf
...
// bind-address를 0.0.0.0d으로 변경
bind-address = 0.0.0.0
...
```

MySQL 재실행

```
$ sudo service mysql restart
```


Docker 설치

오래된 버전 삭제

```
$ sudo apt-get remove docker docker-engine docker.io containerd runc
```

repository 설정

```
$ sudo apt-get update
```

```
$ sudo apt-get install ca-certificates curl gnupg lsb-release
```

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo 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/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Docker Engine 설치

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

설치 완료

잘 설치되었는지 버전 확인

```
$ docker --version
```

Docker Compose 설치

최신버전 설정 후 다운로드

```
$ sudo apt install jq
```

```
$ VERSION=$(curl --silent https://api.github.com/repos/docker/compose/releases/latest | jq .name -r)
```

```
$ DESTINATION=/usr/local/bin/docker-compose
```

```
$ sudo curl -L https://github.com/docker/compose/releases/download/${VERSION}/docker-compose-$(uname -s)-$(uname -m) -o $DESTINATION
```

```
$ sudo chmod 755 $DESTINATION
```

버전 정보 확인

```
$ docker-compose -v
```

▼ 5) OpenVidu 설치 및 환경 세팅

OpenVidu 설치

```
$ cd /opt
```

```
$ sudo curl https://s3.eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_latest.sh | sudo bash
```

설정 파일 수정(.env)

openvidu 경로로 이동

```
$ cd /opt/openvidu
```

```
$ sudo vim .env
```

```
# /opt/openvidu/.env

DOMAIN_OR_PUBLIC_IP=<Linux 서버의 public ip 주소 또는 도메인>
OPENVIDU_SECRET=<사용할 비밀번호 입력>
CERTIFICATE_TYPE=letsencrypt
LETSencrypt_EMAIL=<이메일>
HTTP_PORT=8442
HTTPS_PORT=8443
# HTTP_PORT와 HTTPS_PORT는 letsencrypt 방식의 키를 발급 받기 전까진 기본 포트인 80, 443을 사용해야 함.
```

OpenVidu 서버 실행

```
$ sudo ./openvidu start
```

▼ 6) Nginx 설정과 SSL 발급

NginX 설치

```
$ sudo apt install nginx
```

certbot 이용해 SSL 인증서 발급

```
$ apt-get install python3-certbot-nginx
```

```
$ certbot certonly --nginx -d <도메인 주소>
```

NginX 설치된 경로로 이동하여 설정 파일 수정

- 새로운 conf파일 생성하여 진행

```
$ sudo vim /etc/nginx/sites-available/새로운 파일 이름.conf
```

```
# /etc/nginx/sites-available/themint.conf

server {
```

```

listen 80; #80포트로 받을 때
server_name <도메인 주소>;
return 301 https://<도메인 주소>$request_uri;

}
server {
    listen 443 ssl http2;
    server_name <도메인 주소>;

    # ssl 인증서 적용하기
    ssl_certificate /etc/letsencrypt/live/<도메인 주소>/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/<도메인 주소>/privkey.pem;

    location / {
        proxy_pass http://localhost:3000;
    }

    location /api { # location 이후 특정 url을 처리하는 방법을 정의
        proxy_pass http://localhost:8081;
        proxy_redirect off;
        charset utf-8;

        proxy_http_version 1.1;
        proxy_set_header Connection "upgrade";
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Host $http_host;
        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;
    }
}

```

저장하고 빠져나온 뒤 sites-enabled에 심볼릭 링크 생성

```
$ sudo ln -s /etc/nginx/sites-available/themint.conf /etc/nginx/sites-enabled
```

conf 파일 오류 없는지 확인하고 nginx 실행

```
$ sudo nginx -t
```

```
$ sudo service nginx start
```

▼ 7) Redis 설치 및 환경 세팅

Redis 서버 설치

```
$ sudo apt-get install redis-server
```

Redis 버전 확인

```
$ redis-server --version
```

Redis 설정파일 수정

```
sudo vi /etc/redis/redis.conf
```

```

##### NETWORK #####
...
bind 0.0.0.0

##### MEMORY MANAGEMENT #####

```

```
...
maxmemory 2g
maxmemory-policy allkeys-lru
```



Redis 재실행

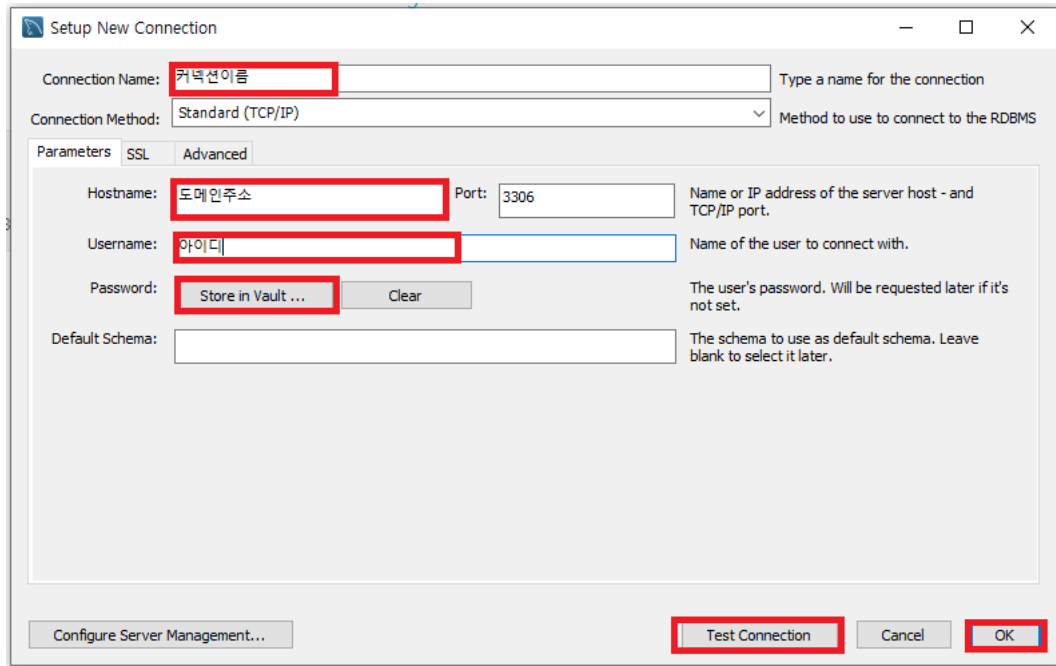
```
sudo systemctl restart redis-server
```

▼ 8) MySQL 워크벤치 사용방법

새로운 커넥션 만들기

- ec2 세팅에서 만들어놓은 유저아이디와 비밀번호 입력 후 커넥션 생성

MySQL Connections  



Setup New Connection

Connection Name: Type a name for the connection

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.

DB 덤프파일 실행시켜 스키마 생성

```

160
161 -----
162 -- Table 'raonzena`.`speak_and_draw`
163 -----
164 CREATE TABLE IF NOT EXISTS 'raonzena`.`speak_and_draw' (
165   'speak_no' BIGINT NOT NULL AUTO_INCREMENT,
166   'answer' VARCHAR(100) NOT NULL,
167   PRIMARY KEY ('speak_no'),
168   UNIQUE INDEX 'speak_no_UNIQUE' ('speak_no' ASC) VISIBLE)
169 ENGINE = InnoDB
170 AUTO_INCREMENT = 1
171 DEFAULT CHARACTER SET = utf8mb3;
172
173 INSERT INTO 'speak_and_draw' VALUES (1,'난지왕소'),(2,'명왕사'),(3,'공자왕정기'),(4,'음유집합'),(5,'산보'),(6,'익동'),(7,'영원사관'),(8,'월발'),(9,'노전당'),(10,'모은'),(11,'옥수수방'),(12,'다들로드'),(13,'사자
174
175 -----
176 -- Table 'raonzena`.`chance`
177 -----
178 CREATE TABLE IF NOT EXISTS 'raonzena`.`chance' (
179   'chance_no' BIGINT NOT NULL AUTO_INCREMENT,
180   'chance_id' VARCHAR(50) NULL,
181   'item' VARCHAR(200) NULL,
182   PRIMARY KEY ('chance_no'))
183 ENGINE = InnoDB;
184
185 INSERT INTO `` ('chance_no','chance_id','item') VALUES (1,'1','100점 당첨!'); INSERT INTO `` ('chance_no','chance_id','item') VALUES (2,'2','5점 당첨!'); INSERT INTO `` ('chance_no','chance_
186
187 SET SQL_MODE=@OLD_SQL_MODE;
188 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
189 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

▼ 9) Jenkins 관련 명령어 & GitLab 웹훅 설정

젠킨스 전용 디렉토리 생성

```
$ mkdir my-jenkins
```

Dockerfile 작성

- my-jenkins 디렉토리로 이동

```
$ cd /my-jenkins
```

- Dockerfile 작성

```
$ vim Dockerfile
```

```

# my-jenkins/Dockerfile
FROM jenkins/jenkins:lts

USER root

# install docker
RUN apt-get update && \
    apt-get -y install apt-transport-https \
        ca-certificates \
        curl \
        gnupg2 \
        zip \
        unzip \
        software-properties-common && \
    curl -fsSL https://download.docker.com/linux/$(. /etc/os-release; echo "$ID")/gpg > /tmp/dkey; apt-key add /tmp/dkey && \
    add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/$(. /etc/os-release; echo "$ID") \
    $(lsb_release -cs) \
    stable" && \
    apt-get update && \
    apt-get -y install docker-ce

```

docker compose 파일 작성

```
$ vim docker-compose.yml
```

```
version: '3.7'
services:
  jenkins:
    build:
      context: .
    container_name: jenkins
    user: root
    privileged: true
    ports:
      - 8080:8080
      - 50000:50000
    volumes:
      - ./jenkins_home:/var/jenkins_home
      - /var/run/docker.sock:/var/run/docker.sock
```

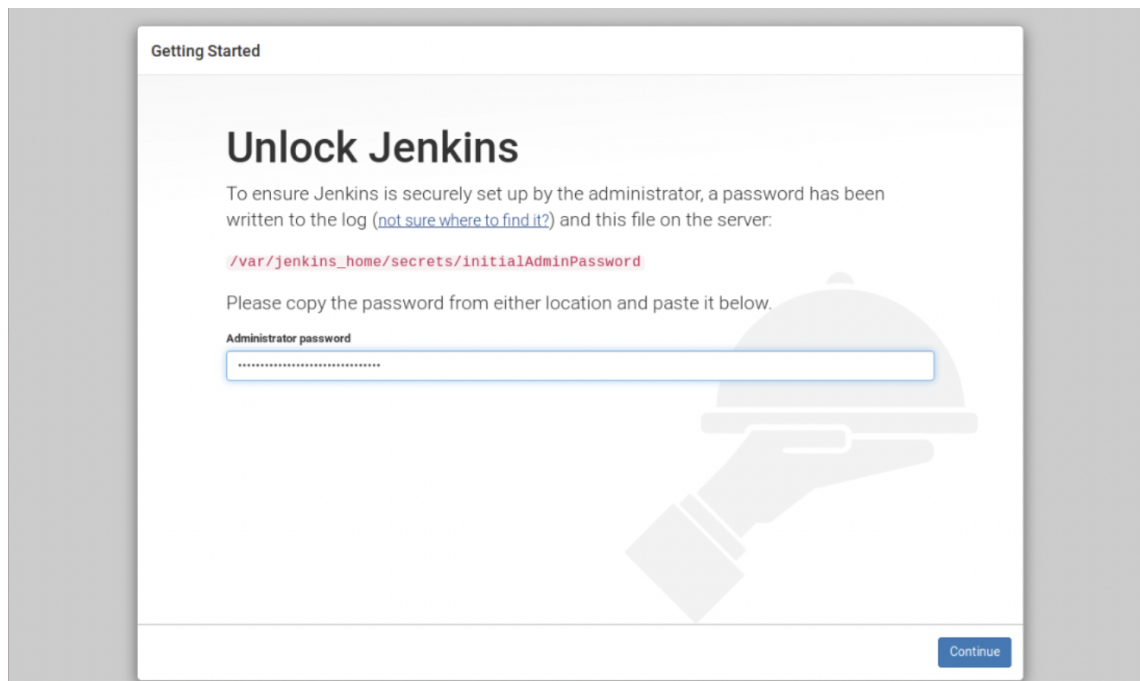
- 젠킨스 포트는 기본 포트인 8080으로

jenkins 실행

```
$ docker-compose up -d
```

jenkins 접속 후 플러그인 설치

- 브라우저에서 <도메인 주소 or IP주소>:<포트번호> 를 입력해 접속하면 시크릿키를 입력하라는 창이 뜬다.
- `docker logs <jenkins 컨테이너 id>` 명령어를 이용해 시크릿키를 찾아서 복사하여 입력한다.



- 그 다음 `Install suggested plugins` 를 클릭하여 플러그인 설치를 진행한다.

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.

관리자 계정 생성

Getting Started

Create First Admin User

계정명

암호

암호 확인

이름

이메일 주소

Jenkins 2.375.2

Skip and continue as admin

Save and Continue

- 젠킨스 이용시 사용할 url 등록 (ex. 도메인주소:포트번호)

Getting Started

Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.

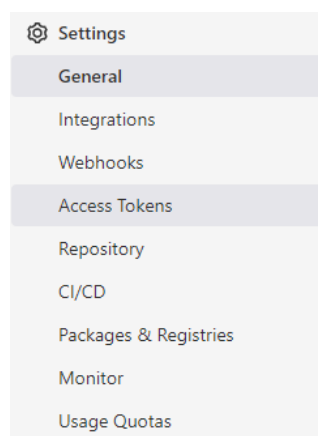
The proposed default value shown is not saved yet and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.375.2

Not nowSave and Finish

GitLab Access Token 발급

프로젝트 세팅에서 Access Tokens 탭 들어가기



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

Select a role

Select scopes

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

- ☐ **api**
Grants complete read/write access to the API, including all groups and projects, the container registry, and the package registry.
- ☐ **read_api**
Grants read access to the API, including all groups and projects, the container registry, and the package registry.
- ☐ **read_repository**
Grants read-only access to repositories on private projects using Git-over-HTTP or the Repository Files API.
- ☐ **write_repository**
Grants read-write access to repositories on private projects using Git-over-HTTP (not using the API).

Create project access token

- 토큰 이름을 원하는 것으로 설정
- 만료일자도 선택
- **write_repository** 빼고 다 체크
- 토큰 발급 완료되면 복사해서 어딘가에 저장

Jenkins 설정

Jenkins 관리 -> 시스템 설정 -> GitLab

Gitlab

- ☒ Enable authentication for '/project' end-point

GitLab connections

Connection name

A name for the connection

Gitlab host URL

The complete URL to the Gitlab server (e.g. http://gitlab.mydomain.com)

https://lab.ssafy.com

Credentials

API Token for accessing Gitlab

GitLab API token

+ Add

고급...

Test Connection

추가

- 이름 원하는 것으로 설정하고 **GitLab host URL**에는 깃랩 메인 주소를 입력
- Credentials**에서 **Add** 버튼을 누른다.

Add Credentials

Domain

Global credentials (unrestricted)

Kind

GitLab API token

Scope ?

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

API token

.....

ID ?

Description ?

Add

Cancel

- 깃랩에서 발급받은 Access Token 입력 후 **Add**
- 그 후, **Apply** 버튼

웹훅 설정

jenkins 프로젝트 생성

- 새로운 item**을 클릭해서 Pipeline 프로젝트를 생성

Build Trigger 설정

- **Build Triggers** 항목의 **Build when a change is pushed to GitLab. GitLab webhook URL:**를 체크하고 **고급** 버튼

Build Triggers

☐ Build after other projects are built ?
☐ Build periodically ?
☒ Build when a change is pushed to GitLab. GitLab webhook URL: <http://k7a207.p.ssafy.io:8080/project/internsanta> ?

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

고급...

- 아래쪽에 있는 **Secret token** 에서 **Generate** 버튼을 눌러 토큰을 생성 후 **Generate** 버튼

Secret token ?

Generate

Clear

GitLab 설정

- **webhook** 설정 탭으로 이동해서 아까의 gitlab webhook url 을 입력하고 발급받은 시크릿토큰도 입력
- **Trigger** 의 **Push events** 에서 트리거를 발생시킬 브랜치 이름을 입력
- **Add webhook** 버튼을 눌러서 완료

Webhooks

[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

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

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.

☐ Job events

A job's status changes.

☐ Pipeline events

A pipeline's status changes.

▼ 10) 배포 명령어 정리

Frontend 도커 이미지 생성 & 실행

React 프로젝트 루트 디렉토리에 Dockerfile 생성

```
# 이미지를 사용합니다. 뒤에 tag가 없으면 latest 를 사용.
FROM node:alpine

# root 에 app 폴더를 생성
RUN mkdir -p /app

# work dir 고정
WORKDIR /app

ENV PATH /app/node_modules/.bin:$PATH

COPY package*.json ./

RUN npm install --force
COPY . .

EXPOSE 3000
ENTRYPOINT ["npm", "run", "start"]FROM openjdk:11-jdk
```

Backend 도커 이미지 생성 & 실행

Springboot 프로젝트 루트 디렉토리에 Dockerfile 생성

```
FROM openjdk:11-jdk
```

```

COPY gradlew .
COPY gradle gradle
COPY build.gradle .
COPY settings.gradle .
COPY src src
RUN chmod +x ./gradlew
RUN ./gradlew bootJar

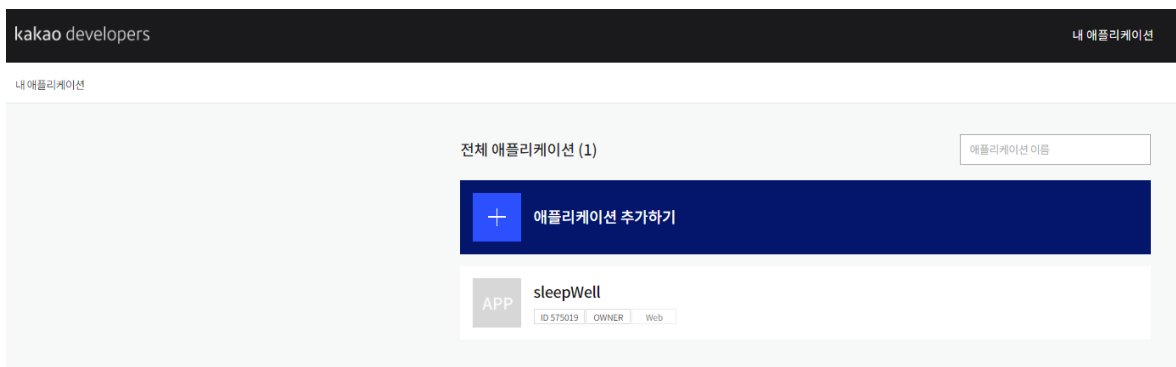
WORKDIR /build/libs

EXPOSE 8081
ENTRYPOINT ["java", "-jar", "raonzena-0.0.1-SNAPSHOT.jar"]

```

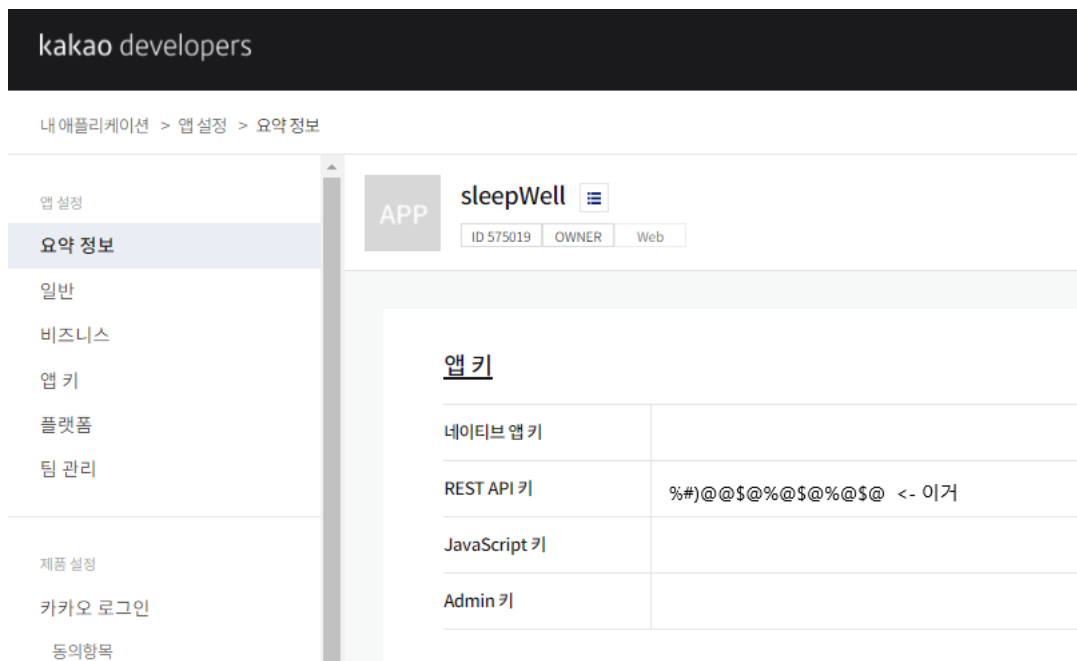
▼ 11) Kakao Dev 설정

애플리케이션 생성



- cliend_id와 redirect_uri를 받아와서 {REST_API_KEY}와 {REDIRECT_URI}에 채워주어야 한다.
- 두 가지를 얻으려면 우선 애플리케이션을 생성한다.

Redirect Url 추가



- cliend_id는 kakao developers에서 내 애플리케이션을 추가했을 때 생기는 REST_API 키를 넣어주면 되고, Redirect URI는 카카오 로그인 메뉴에 들어가서 추가를 해준다.

Redirect URI

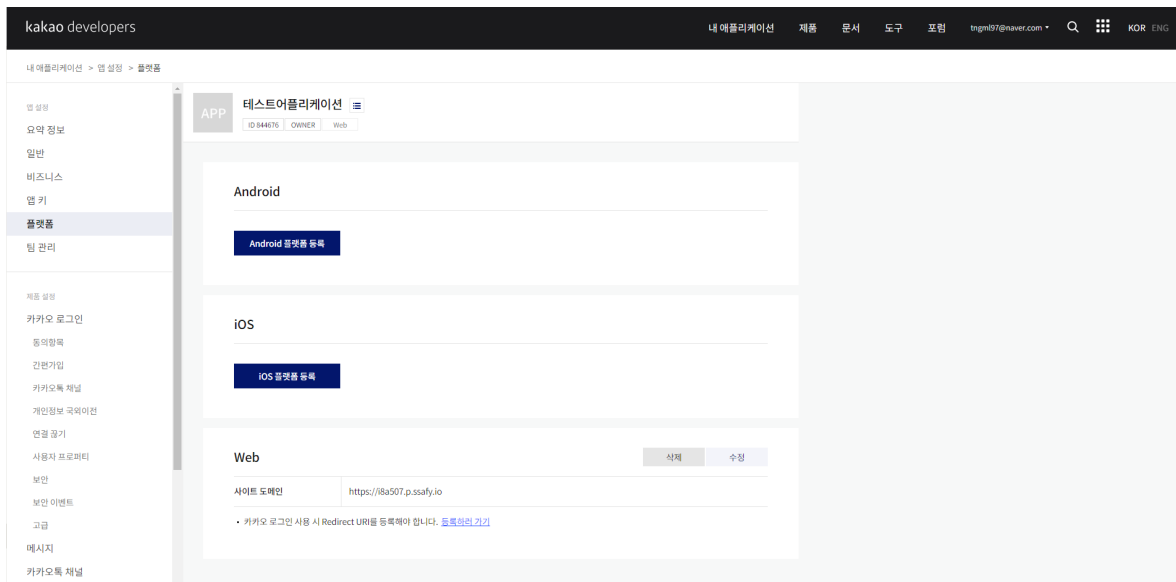
[삭제](#)[수정](#)

Redirect URI	https://i8a507.p.ssafy.io https://i8a507.p.ssafy.io/oauth/kakao/callback http://localhost:3000/oauth/kakao/callback
--------------	---

- 카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다. (최대 10개)
- REST API로 개발하는 경우 필수로 설정해야 합니다.

- Redirect URI는 반드시 **프론트에서 접근할 수 있는 Host로 지정해주어야 한다.**
- 왜냐하면 여기에서 인가 코드 받고 넘기고 등등 모든 작업이 이루어져야 하는데 프론트엔드가 접근할 수 없는 Host로 지정을 해버리면 말 그대로 접근을 못하니 아무것도 할 수 없다.
(localhost:8080 등... 대신 이걸 백엔드에서 자체 테스트할 때 사용할 수 있다)

플랫폼 추가



- Web에서 사용할 것이기 때문에 Web 플랫폼에 사이트 도메인을 추가한다.

▼ 12) AWS S3 Bucket

프리티어 계정 생성

aws 프리티어 계정을 생성

연락처 정보

AWS를 어떻게 사용할 계획이신가요?

- ☐ 비즈니스 - 업무, 학교 또는 조직의 경우
☒ 개인 - 자체 프로젝트의 경우

이 계정에 대해 누구에게 문의해야 하나요?

전체 이름

raonzena

전화 번호

+82

010-1234-5678

국가 또는 리전

대한민국

주소

Giheung-gu, Yongin-si, Gyeonggi-do

010, 1234-5678, 010-1234-5678

시

Yongin

시, 도 또는 리전

Giheung-gu

우편 번호

12345

☐ AWS 이용약관 및 개인정보의 수집 및 이
용에 대한 사항에 동의합니다.

계속(2/5단계)

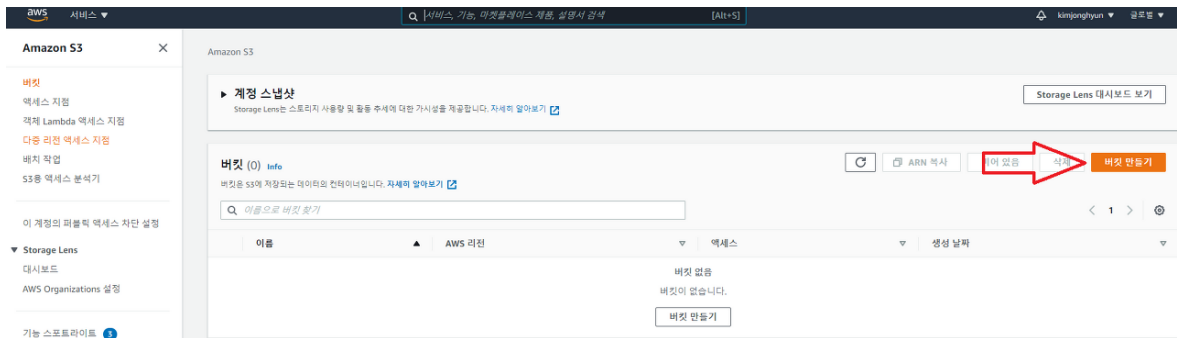
s3 버킷 및 객체 생성

AWS 콘솔 접속



- AWS에 로그인하여 콘솔화면 검색창에 s3 검색해서 들어간다.

AWS 버킷 생성하기



- 우측에 있는 버킷 만들기 버튼을 누르자



- 버킷이름을 입력하고 AWS 리전은 아시아 태평양 서울로 지정한다.

이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(엑세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 이 버킷 및 해당 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 모든 퍼블릭 액세스 차단을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지점에만 적용됩니다. AWS에서는 모든 퍼블릭 액세스 차단을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 이 버킷 또는 내부 객체에 대한 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 아래 개별 설정을 사용자 지정할 수 있습니다. [자세히 알아보기](#)

☐ 모든 퍼블릭 액세스 차단

이 설정을 활성화하면 아래 4개의 설정을 모두 활성화한 것과 같습니다. 다음 설정 각각은 서로 독립적입니다.

☐ 새 ACL(엑세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 새로 추가된 버킷 또는 객체에 적용되는 퍼블릭 액세스 권한을 차단하며, 기존 버킷 및 객체에 대한 새 퍼블릭 액세스 ACL 생성을 금지합니다. 이 설정은 ACL을 사용하여 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 권한을 변경하지 않습니다.

☐ 임의의 ACL(엑세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 모든 ACL을 무시합니다.

☐ 새 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 새 버킷 및 액세스 지점 정책을 차단합니다. 이 설정은 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 정책을 변경하지 않습니다.

☐ 임의의 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 및 교차 계정 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 정책을 사용하는 버킷 또는 액세스 지점에 대한 퍼블릭 및 교차 계정 액세스를 무시합니다.



모든 퍼블릭 액세스 차단을 비활성화하면 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있습니다.

정적 웹 사이트 호스팅과 같은 구체적으로 확인된 사용 사례에서 퍼블릭 액세스가 필요한 경우가 아니면 모든 퍼블릭 액세스 차단을 활성화하는 것이 좋습니다.



현재 설정으로 인해 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있음을 알고 있습니다.

- 일단 모든 사용자에게 액세스를 허용하기 위해 모든 퍼블릭 액세스 차단 체크박스를 해제하고 아래에있는 체크박스에 체크를 해두고 버킷만들기 버튼을 눌러 버킷을 생성한다.

버킷 버전 관리

버전 관리는 객체의 여러 버전을 동일한 버킷에서 관리하기 위한 수단입니다. 버전 관리를 사용하여 Amazon S3 버킷에 저장된 모든 객체의 각 버전을 보존, 검색 및 복원할 수 있습니다. 버전 관리를 통해 의도치 않은 사용자 작업과 애플리케이션 장애를 모두 복구할 수 있습니다. [자세히 알아 보기](#)

버킷 버전 관리

☒ 비활성화

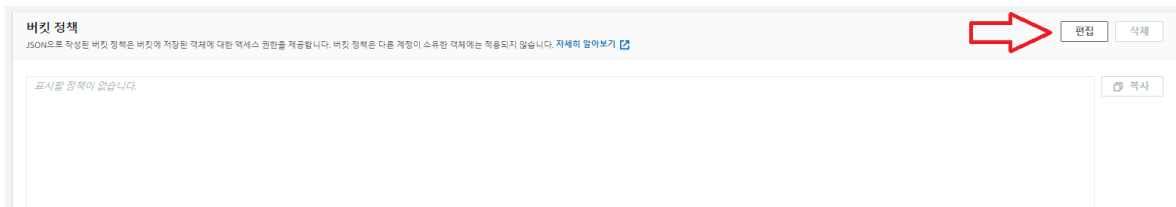
☐ 활성화

- 버킷내부에 있는 객체들의 버전을 관리하는건데 똑같은 객체가 생성될 경우 버전관리를 할지 말지 결정하는 것이다.
- 비활성화하면 기존객체를 덮어쓰게되고 활성화하면 덮어쓰이지않고 동일한 객체가 여러개 생성이 되며 각 객체를 식별할 수 있는 고유한 버전ID가 생긴다.
- 비활성화를 누르고 진행하자.

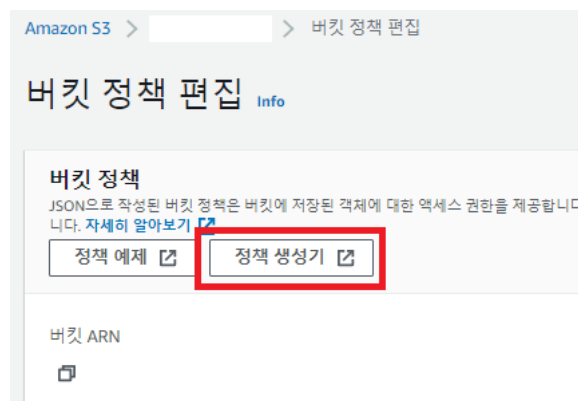
AWS 버킷 퍼블릭 설정



- 버킷을 생성하면 목록에 해당버킷이 나오는데 액세스에 **객체를 퍼블릭으로 설정할 수 있음** 이라고 나온다.
- 버킷의 정책을 수정해서 모든사람이 액세스할 수 있도록 수정해야한다. 버킷 상세페이지로 이동하자.



- 상세페이지 상단 '권한' 탭을 누르면 하단에 위와 같은 목록을 볼 수 있다. '편집' 버튼을 누르자



- 버킷 ARN값을 복사하고 정책생성기 버튼을 누르자

AWS Policy Generator

The AWS Policy Generator is a tool that enables you to create policies that control access to Amazon Web Services. For more information about AWS policies, see [key concepts in Using AWS Identity and Access Management](#). Here are [sample policies](#).

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an [IAM Policy](#), an [S3 Bucket Policy](#), or an [S3 Queue Policy](#).

Select Type of Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See [a description of elements](#) that you can use in

Effect ☒ Allow ☐ Deny

Principal
Use a comma to separate multiple values.

AWS Service
Use multiple statements to add permissions for more than one service.

Actions ☐ All Actions (*all*)

Amazon Resource Name (ARN)
ARN should follow the following format: arn:aws:s3:::{BucketName}/{KeyPrefix}.
Use a comma to separate multiple values.

[Add Conditions \(Optional\)](#)

- 사진과 같이 데이터를 설정한다. Action에는 GetObject, PutObject, DeleteObject 3개를 체크하고 ARN에는 복사해둔 ARN값을 입력한다. 버킷을 조회, 생성, 삭제하기 위함이다.
- ARN값을 입력하되 /* 값도 추가해줘야한다. ARN 값이 arn:aws:s3:::raonzena라 가정하면 arn:aws:s3:::raonzena/* 라고 입력해주면 된다.
- Add Statement버튼을 클릭하면 위사진처럼 나오게된다. Generate Policy버튼을 누르자
- Generate Policy버튼을 클릭하게되면 팝업으로 JSON구조의 버킷정책이 나온다.

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. [자세히 알아보기](#)

편집

삭제

{

"Version": "2012-10-17",

"Id": "Policy1675584050337",

"Statement": [

{

"Sid": "Stmt1675584048296",

"Effect": "Allow",

"Principal": "*",

"Action": [

"s3:DeleteObject",

"s3:GetObject",

"s3:PutObject"

],

"Resource": "arn:aws:s3:::raonzena/*"

}

]

}

복사

- 해당내용을 복사한 후 버킷정책에 붙여넣고 변경사항을 저장한다.

✓ 버킷 정책을 편집했습니다.

Amazon S3 >

Info

퍼블릭 액세스 가능

객체

속성

권한

지표

관리

액세스 지점

권한 개요

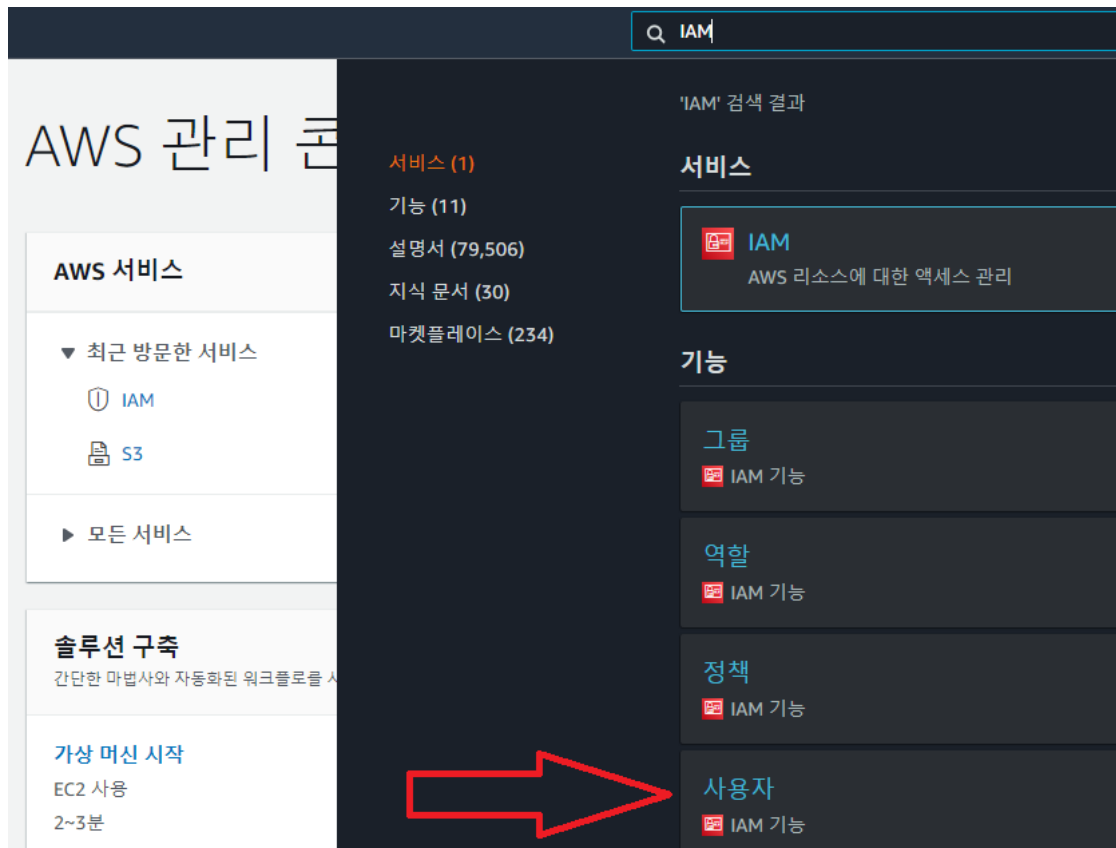
액세스

△ 퍼블릭

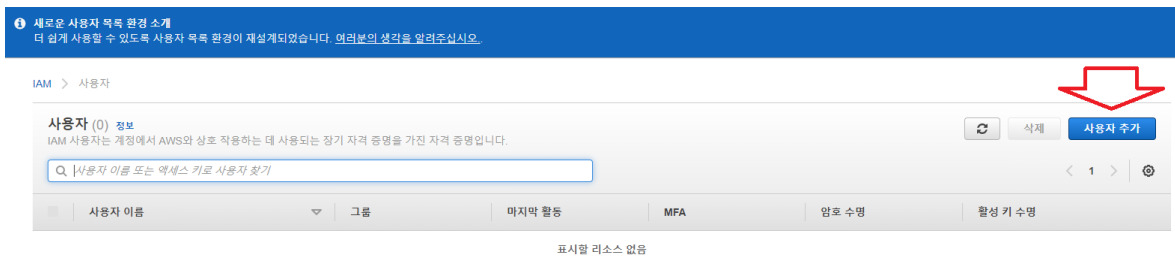
- 버킷정책이 퍼블릭으로 변경되었다.

IAM 사용자 및 사용자에게 대한 Role 추가 - access key/ secret key 발급

IAM 사용자 추가하기



- AWS Console 검색창에 'IAM' 이라고 입력한 뒤 '사용자'를 클릭하자.



- 사용자 추가버튼을 클릭하여 사용자를 추가한다.

사용자 세부 정보 지정

사용자 세부 정보

사용자 이름

raonzena

사용자 이름은 최대 64자까지 가능합니다. 유효한 문자: A-Z, a-z, 0-9 및 +, -, @, _ (하이픈)

☐ 콘솔 액세스 활성화 - 선택 사항

사용자가 AWS 관리 콘솔에 로그인할 수 있도록 허용하는 암호를 활성화합니다.

☒ 프로그래밍 방식의 액세스의 경우 사용자를 생성한 후 액세스 키를 생성할 수 있습니다. 자세히 알아보기

취소

다음

권한 설정

기존 그룹에 사용자를 추가하거나 새 그룹을 생성합니다. 직무별로 사용자의 권한을 관리하려면 그룹을 사용하는 것이 좋습니다. [자세히 알아보기](#)

권한 옵션

☐ 그룹에 사용자 추가
기존 그룹에 사용자를 추가하거나 새 그룹을 생성합니다. 그룹을 사용하여 직무별로 사용자 권한을 관리하는 것이 좋습니다.

☐ 권한 복사
기존 사용자의 모든 그룹 멤버십, 연결된 관리형 정책 및 인라인 정책을 복사합니다.

☒ 직접 정책 연결
관리형 정책을 사용자에게 직접 연결합니다. 사용자에게 연결하는 대신, 정책을 그룹에 연결한 후 사용자를 적용한 그룹에 추가하는 것이 좋습니다.

권한 정책 (1/1037)

새 사용자에게 연결할 정책을 하나 이상 선택합니다.

5 개 일치

amazonS3

필터 지우기

정책 이름	유형	연결된 엔터티
<input checked="" type="checkbox"/> AmazonS3FullAccess	AWS 관리형	0
<input type="checkbox"/> AmazonS3ObjectLambdaExecutionRolePolicy	AWS 관리형	0
<input type="checkbox"/> AmazonS3OutpostsFullAccess	AWS 관리형	0
<input type="checkbox"/> AmazonS3OutpostsReadOnlyAccess	AWS 관리형	0
<input type="checkbox"/> AmazonS3ReadOnlyAccess	AWS 관리형	0

▶ 권한 경계 - 선택 사항

권한 경계를 설정하여 이 사용자에게 대한 최대 권한을 제어합니다. 권한 관리를 다른 사용자에게 위임하는 데 사용되는 이 고급 기능을 사용합니다. [자세히 알아보기](#)

- 해당 사용자에게 AmazonS3FullAccess 권한을 부여하자 AWS SDK를 이용하여 애플리케이션에서 S3에 액세스 하기위함이다.

IAM > 사용자 > 사용자 생성

1단계
사용자 세부 정보 지정

2단계
권한 설정

3단계
검토 및 생성

검토 및 생성

선택 사항을 검토합니다. 사용자를 생성한 후 자동 생성된 암호를 보고 다운로드할 수 있습니다(활성화된 경우).

사용자 세부 정보

사용자 이름 raonizena	콘솔 암호 유형 None	암호 재설정 필요 아니요
---------------------	------------------	------------------

권한 요약

이름	유형	다음과 같이 사용
AmazonS3FullAccess	AWS 관리형	권한 정책

태그 - 선택 사항

태그는 리소스를 식별, 구성 또는 검색하는 데 도움이 되도록 AWS 리소스에 추가할 수 있는 키 값 쌍입니다. 이 사용자와 연결할 태그를 선택합니다.

리소스와 연결된 태그가 없습니다.

새 태그 추가

최대 50개의 태그를 더 추가할 수 있습니다.

취소

이전

사용자 생성

- 사용자 생성을 누른다.

새로운 사용자 목록 환경 소개

사용자가 성공적으로 생성됨

IAM > 사용자

사용자 (1) 정보

IAM 사용자는 계정에서 AWS와 상호 작용하는 데 사용되는 장기 자격 증명을 가진 자격 증명입니다.

<input type="checkbox"/>	사용자 이름	그룹	마지막 활동	MFA	암호 수명	활성 키 수명
<input type="checkbox"/>	raonizena	없음	안 함	없음	<	<

사용자 추가

- 다시 사용자 이름을 클릭해서 들어간다.

ARN am:aws:iam::174814983215:user/raonzena	콘솔 액세스 비활성화됨	엑세스 키 1 활성화되지 않음
생성됨 February 05, 2023, 18:43 (UTC+09:00)	마지막 콘솔 로그인 -	엑세스 키 2 활성화되지 않음

권한

그룹

태그

보안 자격 증명

엑세스 관리자

콘솔 로그인

콘솔 액세스 활성화

콘솔 로그인 링크
https://174814983215.signin.aws.amazon.com/console

콘솔 암호
활성화되지 않음

멀티 팩터 인증(MFA) (0)

제거

재등록하

MFA 디바이스 할당

MFA를 사용하여 AWS 환경의 보안을 강화합니다. MFA로 로그인하려면 MFA 디바이스의 인증 코드가 필요합니다. 각 사용자는 MFA 디바이스를 최대 8개까지 할당할 수 있습니다. [Learn more](#)

디바이스 유형	식별자	생성 날짜
MFA 디바이스가 없습니다. MFA 디바이스를 할당하여 AWS 환경 보안 개선하기		
<div>MFA 디바이스 할당</div>		

엑세스 키 (0)

엑세스 키 만들기

엑세스 키를 사용하여 AWS CLI, AWS Tools for PowerShell, AWS SDK 또는 직접 AWS API 호출을 통해 AWS에 프로그래밍 방식 호출을 전송합니다. 한 번에 최대 두 개의 엑세스 키(활성 또는 비활성)를 가질 수 있습니다. [Learn more](#)

엑세스 키 없음

- 보안 자격 증명을 누른 후 엑세스 키 만들기 버튼을 눌러 accessKey와 secretKey를 발급받는다.

IAM > 사용자 > raonzena > 엑세스 키 만들기

1단계

엑세스 키 모범 사례 및 대안

2단계 - 선택 사항

설명 태그 설정

3단계

엑세스 키 검색

엑세스 키 모범 사례 및 대안

보안 개선을 위해 엑세스 키와 같은 장기 자격 증명을 사용하지 마세요. 다음과 같은 사용 사례와 대안을 고려하세요.

☒ Command Line Interface(CLI)
 AWS CLI를 사용하여 AWS 계정에 액세스할 수 있도록 이 엑세스 키를 사용할 것입니다.

☐ 로컬 코드
 로컬 개발 환경의 애플리케이션 코드를 사용하여 AWS 계정에 액세스할 수 있도록 이 엑세스 키를 사용할 것입니다.

☐ AWS 컴퓨팅 서비스에서 실행되는 애플리케이션
 Amazon EC2, Amazon ECS 또는 AWS Lambda와 같은 AWS 컴퓨팅 서비스에서 실행되는 애플리케이션 코드를 사용하여 AWS 계정에 액세스할 수 있도록 이 엑세스 키를 사용할 것입니다.

☐ 서드 파티 서비스
 AWS 리소스를 모니터링 또는 관리하는 서드 파티 애플리케이션 또는 서비스에 액세스할 수 있도록 이 엑세스 키를 사용할 것입니다.

☐ AWS 외부에서 실행되는 애플리케이션
 애플리케이션을 온프레미스 호스트에서 실행하거나 로컬 AWS 클라이언트 또는 서드 파티 AWS 클라이언트를 사용할 수 있도록 이 엑세스 키를 사용할 것입니다.

☐ 기타
 귀하의 사용 사례가 여기에 나열되어 있지 않습니다.

권장되는 대안

- 브라우저 기반 CLI인 [AWS CloudShell](#)을 사용하여 명령을 실행합니다. [Learn more](#)
- [AWS CLI V2](#)를 사용하고 IAM 자격 증명 센터의 사용자를 통한 인증을 활성화합니다. [Learn more](#)

☐ 위의 권장 사항을 이해했으며 엑세스 키 생성을 계속하려고 합니다.

1단계

액세스 키 모범 사례 및 대안

2단계 - 선택 사항

설명 태그 설정

3단계

액세스 키 검색

액세스 키 검색

액세스 키

본실하거나 잊어버린 비밀 액세스 키는 검색할 수 없습니다. 대신 새 액세스 키를 생성하고 이전 키를 비활성화합니다.

액세스 키

AKIASRM6IGQXVNE4QEC

비밀 액세스 키

5UHDl9Ayt2CHcKWKLjTQzYpHJx7gdedpDebt+ak9 [숨기기](#)

액세스 키 모범 사례

- 액세스 키를 일반 텍스트, 코드 리포지토리 또는 코드로 저장해서는 안 됩니다.
- 더 이상 필요 없는 경우 액세스 키를 비활성화하거나 삭제합니다.
- 최소 권한을 활성화합니다.
- 액세스 키를 정기적으로 교체합니다.

액세스 키 관리에 대한 자세한 내용은 [AWS 액세스 키 관리 모범 사례](#)를 참조하세요.[.csv 파일 다운로드](#)[완료](#)[다운로드 zip - 편집](#)

- 완료 후 사용자의 accessKey와 secretKey가 나오는데 현재화면에서만 볼 수있기때문에 csv파일로 다운받아두자.
- accessKey와 secretKey가 있어야 애플리케이션에서 AWS Service인 S3에 액세스할 수 있다.