



Infra

[기술 스택](#)
[아키텍처](#)
[Porting Manual](#)
[외부 서비스 정보](#)
[DB 데이터](#)
[계정 정보](#)
[시연 시나리오](#)

▼ 기술 스택

Cooperation

Git GitLab Notion Jira Mattermost

Tools

IntelliJ 2023.1.3 AndroidStudio 2022.3.1.21

Infra

AWS EC2 20.04.6 Nginx 1.18.0 Jenkins 2.430 Docker 24.0.7

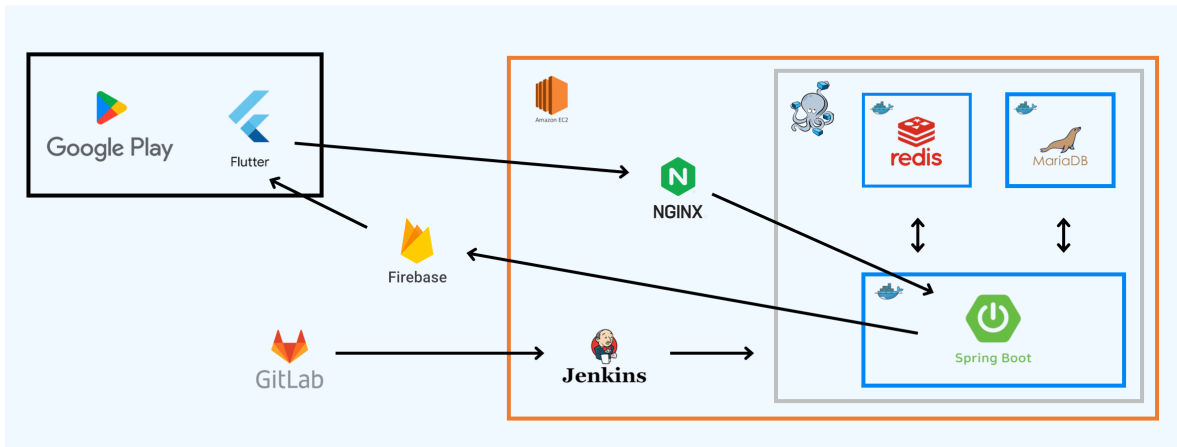
BackEnd

Openjdk 11 Gradle 8.3 SpringBoot 2.7.17 JPA 2.7.17 MariaDB 11.1.2 JWT 4.2.1 Redis 7.2.3 querydsl 5.0.0

FrontEnd

flutter 3.13.9 flame 1.10.1

▼ 아키텍처



▼ Porting Manual

EC2 접속

■ 접속 방법

```
//PEM키가 있는 위치에서 터미널 열고
ssh -i K9A406T.pem ubuntu@k9a406.p.ssafy.io

or SSH platform 'termius'에 PEM키 등록 → 접속 편함!
```

EC2 내부 설치

■ EC2 버전 관리

```
//인스턴스에 설치된 패키지 목록을 최신화
sudo apt-get update

//인스턴스에 설치된 패키지들을 최신 버전으로 업그레이드
sudo apt-get upgrade

-> 서버에 접속할 때마다 입력하여 인스턴스를 최신 상태로 유지하자!
```

■ EC2 Time zone 설정

```
//한국 표준시로 맞추기
sudo timedatectl set-timezone Asia/Seoul
```

■ 방화벽 설정

```
//ubuntu에서 기본으로 제공하는 방화벽 설정 도구 ufw 설치
sudo apt install ufw

//모든 인바운드(수신) 연결 차단
sudo ufw default deny incoming
```

```
//모든 아웃바운드(송신) 연결 허용
sudo ufw default allow outgoing

//ssh(22) 허용
sudo ufw allow ssh

//http(80) 허용
sudo ufw allow http

//https(443) 허용
sudo ufw allow https

-> 이렇게 하면 다른 사용자들은 22, 80, 443번 포트만 서버에 접근 가능!

//방화벽 활성화
// 주의) ssh 포트가 허용되지 않은 상태로 활성화하면 서버에 접속이 불가능 -> 서버 초기화를 해야 함!
sudo ufw enable

//허용된 포트 확인
sudo ufw status

(
+ 규칙 삭제할 경우


sudo ufw delete (allow/deny) ${port}
)
```

■ openjdk-11 설치

```
sudo apt install -y openjdk-11-jdk
```

■ Jenkins 설치

Linux
Jenkins – an open source automation server which enables developers around the world to reliably build, test, and deploy their software

 <https://www.jenkins.io/doc/book/installing/linux/>



Jenkins

→ 최신 버전 확인!

```
//키 추가
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian/jenkins.io-2023.key

//저장소 설정
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null

//latest
sudo apt-get update

//Jenkins 설치
sudo apt-get install jenkins
```

■ Jenkins port 변경 시(Jenkins 기본 포트 8080)

```
//jenkins.service 파일 수정
sudo vi /lib/systemd/system/jenkins.service

-> 원하는 포트로 변경하기
Ex) Environment="JENKINS_PORT=8081"
```

```
//해당 포트 ufw 추가
sudo ufw allow 8081
```

• Jenkins 접속

```
//Jenkins 활성화
sudo systemctl enable jenkins

//Jenkins 실행
sudo systemctl start jenkins

//Jenkins 상태 확인
sudo systemctl status jenkins

(
+ jenkins 실행 이후 포트 변경하는 경우

//해당 포트 ufw 추가
sudo ufw allow 8081

//Jenkins 재시작
sudo systemctl stop jenkins
sudo systemctl start jenkins

or

sudo systemctl restart jenkins
)

-> 주소창에 '서버도메인:Jenkins port'로 접속
Ex) k9a406.p.ssafy.io:8081

//Jenkins Unlock
sudo cat /var/lib/jenkins/secrets/initialAdminPassword

-> secret key를 접속한 Jenkins의 Administrator Password에 입력

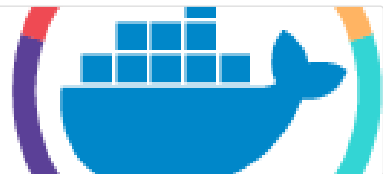
-> suggested plugins 설치
```

• Docker 설치

Install Docker Engine on Ubuntu

Jumpstart your client-side server applications with Docker Engine on Ubuntu. This guide details prerequisites and multiple methods to install Docker Engine on Ubuntu.

 <https://docs.docker.com/engine/install/ubuntu/>



→ 최신 버전 확인!

```
//충돌 방지를 위해 구버전 Docker 패키지 삭제
for pkg in docker.io docker-doc docker-compose podman-docker containerd runc; do sudo apt-get remove $pkg; done

sudo apt-get update

//패키지 설치
sudo apt-get install ca-certificates curl gnupg

//공식 키 추가
sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg

//안정 버전 저장소 설정
echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
"$(. /etc/os-release && echo "$VERSION_CODENAME)" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
//latest
sudo apt-get update

//Docker 설치
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

//설치 확인
docker --version (or -v)

//접속계정 확인
whoami

//Docker group 생성
sudo groupadd docker


//Docker 실행권한 부여
sudo usermod -aG docker ${접속계정명}
Ex) sudo usermod -aG docker ubuntu

//재로그인
newgrp docker

//재시작
sudo service docker restart
```

- Docker-Compose 설치

Releases · docker/compose
Define and run multi-container applications with Docker - docker/compose



<https://github.com/docker/compose/releases/>

150 Contributors · 13k Used by · 14 Discussions · 31k Stars · 5k Forks

→ 최신 버전 확인!

- Nginx 설치

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

- Certbot(Let's encrypt) 설치

```
sudo snap install core
sudo snap refresh core
sudo apt remove certbot
sudo snap install --classic certbot
ln -s /snap/bin/certbot /usr/bin/certbot
sudo ln -s /snap/bin/certbot /usr/bin/certbot
sudo certbot --nginx
sudo apt-get -y install apt-transport-https ca-certificates curl gnupg-agent software-properties-common
```

- `/etc/nginx/conf.d/default.conf` 파일 생성 후 변경

- ▼ default.conf

```
server {
    //80 적용
    listen 80;
    server_name ${domain};

    //ssl 적용
    listen 443 ssl;
```

```

ssl_certificate /etc/letsencrypt/live/${domain}/fullchain.pem; # managed by Certbot
ssl_certificate_key /etc/letsencrypt/live/${domain}/privkey.pem; # managed by Certbot
include /etc/letsen

location / {
    proxy_pass http://${domain}:${port};
    # proxy_redirect off;
    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-Host $server_name
}
}

```

- or `nginx.conf` 파일 변경

▼ nginx.conf

```

user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
    worker_connections 768;
    # multi_accept on;
}

http {
    sendfile on;
    tcp_nopush on;
    tcp_nodelay on;
    keepalive_timeout 65;
    types_hash_max_size 2048;

    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    server {
        listen 80;
        server_name k9a406.p.ssafy.io;

        listen 443 ssl;
        server_name k9a406.p.ssafy.io;
        ssl_certificate /etc/letsencrypt/live/k9a406.p.ssafy.io/fullchain.pem; # managed by Certbot
        ssl_certificate_key /etc/letsencrypt/live/k9a406.p.ssafy.io/privkey.pem; # managed by Certbot
        include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot

        location / {
            proxy_pass http://k9a406.p.ssafy.io:8080;
            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;
        }
    }

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: P00DLE
    ssl_prefer_server_ciphers on;

    access_log /var/log/nginx/access.log;
    error_log /var/log/nginx/error.log;

    gzip on;

    include /etc/nginx/conf.d/*.conf;
    include /etc/nginx/sites-enabled/*;
}

```

배포 설정

- Jenkins 관리 - Jenkins Credential 설정 - Username with password
 - Password에 Gitlab 계정 Password 넣으니 권한 에러 발생

- Gitlab에서 Access Token - add new token - Select scopes 모든 권한 선택 후 생성해서 넣기

- Jenkins - 새로운 Item - \${projectName} - pipeline 생성

- Build Triggers - Build when a change is pushed to GitLab. GitLab webhook URL:
http://\${domain}:\${port}/project/\${projectName}

- 고급 - Secret token - Generate - 생성된 토큰 복사해놓기

- Pipeline

- ▼ Script

```

pipeline {
    agent any
    environment {
        CREDENTIALS_ID = 'kingmaker_backend'
        GIT_BRANCH = 'backend'
        GIT_URL = 'https://lab.ssafy.com/s09-final/S09P31A406.git'
    }
    stages {
        stage("Clone Repository") {
            steps {
                echo "Clone Repository"

                git branch: "${GIT_BRANCH}",
                    credentialsId: "${CREDENTIALS_ID}",
                    url: "${GIT_URL}"
            }
        }
        stage("Set Environment") {
            steps {
                echo "Set Environment"

                // dir("./backend/src/main/resources") {
                //     sh "mkdir firebase"
                // }
                sh "cp /app/config/kingmaker/application.yml /var/lib/jenkins/workspace/kingmaker_backend/backend/src/main/resources/"
                sh "cp /app/config/kingmaker/env.yml /var/lib/jenkins/workspace/kingmaker_backend/backend/src/main/resources/"
                // sh "cp /app/config/kingmaker/application-oauth2.yml /var/lib/jenkins/workspace/kingmaker_backend/backend/src/main/resources/"
                sh "cp /app/config/kingmaker/kingmaker-811e9-firebase.json /var/lib/jenkins/workspace/kingmaker_backend/backend/src/main/resources/"

                dir("./backend") {
                    sh "cp /app/config/kingmaker/DockerFile ./DockerFile"
                }
                sh "chmod +x /var/lib/jenkins/workspace/kingmaker_backend/backend/gradlew"
                sh "/var/lib/jenkins/workspace/kingmaker_backend/backend/gradlew init"

                dir("./backend") {
                    // sh "./gradlew clean"
                    sh "./gradlew build -x test"
                }
            }
        }
        stage("Down Docker") {
            steps {
                echo "Down Docker"

                sh "docker-compose -f /app/config/kingmaker/docker-compose.yml down --rm all"
            }
        }
        stage("Build Docker") {
            steps {
                echo "Build Docker"

                sh "docker-compose -f /app/config/kingmaker/docker-compose.yml build --no-cache"
            }
        }
        stage("Docker Up") {
            steps {
                echo "Docker Up"

                sh "docker-compose -f /app/config/kingmaker/docker-compose.yml up -d"
            }
        }
    }
}

```

- Gitlab과 Webhook설정
 - Settings - Webhooks - Add new webhook
 - URL: `http://${domain}:${port}/${projectName}`
 - Secret token: 복사한 jenkins Secret token 넣기
 - Trigger - push events - Wildcard pattern - 사용할 `${branchName}` 작성
 - SSL verification - Enable SSL verification
- server에서만 사용할 파일은 `/app/config/kingmaker` 폴더에서 cp해오는 걸로 설정
→ 서버에 폴더 생성

```
sudo mkdir /app/config/kingmaker
```

- BE 추가해야 하는 파일(/app/config/kingmaker 내부에 생성)

▼ DockerFile

```
FROM gradle:8.3-jdk11
WORKDIR /server
COPY ./build/libs/kingmaker-0.0.1-SNAPSHOT.jar kingmaker.jar
ENTRYPOINT ["java", "-jar", "kingmaker.jar"]
```

▼ docker-compose.yml

```
version: "3"
services:
  database:
    image: mariadb:latest
    container_name: kingmaker-db
    environment:
      MYSQL_ROOT_PASSWORD: kingmaker@406
      MYSQL_USER: kingmaker
      MYSQL_PASSWORD: kingmaker@406
      MYSQL_DATABASE: kingmaker
      MYSQL_CHARACTER_SET_SERVER: utf8mb4
      MYSQL_COLLATION_SERVER: utf8mb4_unicode_ci
      TZ: Asia/Seoul
    ports:
      - 3306:3306
    volumes:
      - kingmaker-db:/var/lib/mysql
    networks:
      - kingmaker

  redis:
    image: redis:latest
    container_name: kingmaker-redis
    environment:
      - REDIS_PASSWORD=kingmaker@406
    hostname: kingmaker-redis
    command: redis-server --requirepass kingmaker@406 --port 6379
    volumes:
      - kingmaker-redis-volume:/data
    ports:
      - 6379:6379
    restart: always
    networks:
      - kingmaker

  backend:
    build:
      context: /var/lib/jenkins/workspace/kingmaker_backend/backend
      dockerfile: DockerFile
    restart: always
```



```

    depends_on:
      - database
    ports:
      - 8080:8080
    container_name: kingmaker-backend
    networks:
      - kingmaker

networks:
  kingmaker:
    external: true

volumes:
  kingmaker-db:
  kingmaker-redis-volume:

```

▼ env.yml

```

MARIADB_DATABASE_URL: jdbc:mariadb://k9a406.p.ssafy.io:3306/kingmaker?useSSL=false&serverTimezone=Asia/Seoul&allowPublicKe
DATABASE_USERNAME: kingmaker
DATABASE_PASSWORD: kingmaker@406
REDIS_URL: kingmaker-redis
REDIS_PASSWORD: kingmaker@406
REDIS_PORT: 6379
SECRET_KEY: FD09BUD09FGR09FJ40WJMFC90243JKR04329GFJH23490JGF3409GH3904GJH3094DF923FG2983298F9823982
GOOGLE_CLIENT_ID: 1050976651761-eif415ke1hjp1hocu7gpp751j0tatgv9.apps.googleusercontent.com
GOOGLE_SECRET_KEY: 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
GOOGLE_REDIRECT_URI: https://k9a406.p.ssafy.io/auth/google
KAKAO_CLIENT_ID: 6185c87a40b7613b3fdc4b680badfce3
KAKAO_SECRET_KEY: 333mFCM6Pe24EraMsb7DNXqqzxQZpUu4
KAKAO_REDIRECT_URI: https://k9a406.p.ssafy.io/auth/kakao

```

▼ application.yml → `validate` 로 변경

```

server:
  port: 8080

servlet:
  encoding:
    charset: UTF-8
    force: true

spring:
  config:
    import:
      - env.yml
      - application-redis.yml
      - application-oauth2.yml
      - application-jwt.yml

  datasource:
    driver-class-name: org.mariadb.jdbc.Driver
    url: ${MARIADB_DATABASE_URL}
    username: ${DATABASE_USERNAME}
    password: ${DATABASE_PASSWORD}
    hikari:
      maximum-pool-size: 2
      max-lifetime: 30000

  jpa:
    show-sql: true
    hibernate:
      ddl-auto: validate
    properties:
      hibernate:
        format-sql: true
        dialect: org.hibernate.dialect.MariaDBDialect
    defer-datasource-initialization: true

  sql:
    init:
      mode: always

```

▼ kingmaker-811e9-firebase.json

```
{
  "type": "service_account",
  "project_id": "kingmaker-811e9",
  "private_key_id": "6d36bc72e69d0fb714e6d114b517a38a02249006",
  "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvgIBADANBgkqhkiG9w0BAQEFAASCBAgEAAoIBAQUdUpPIFju1Z0FAD\nDnDxE1Qs0wy\n"client_email": "firebase-adminsdk-9im1k@kingmaker-811e9.iam.gserviceaccount.com",
  "client_id": "101554037956780932011",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/firebase-adminsdk-9im1k%40kingmaker-811e9.iam\n"universe_domain": "googleapis.com"
}
```

■ FE 추가해야 하는 파일(개발 시)

▼ .env

```
URL="https://k9a406.p.ssafy.io"
#URL="http://10.0.2.2:8080"
NATIVE_APP_KEY=35d48fe78a4ae625ef1d0f37dd18c93f
```

▼ android/gradle.properties

```
org.gradle.jvmargs=-Xmx1536M
android.useAndroidX=true
android.enableJetifier=true
KAKAO_API_KEY=kakao35d48fe78a4ae625ef1d0f37dd18c93f
```

▼ android/app/google-services.json

```
{
  "project_info": {
    "project_number": "790469978744",
    "project_id": "kingmaker-811e9",
    "storage_bucket": "kingmaker-811e9.appspot.com"
  },
  "client": [
    {
      "client_info": {
        "mobilesdk_app_id": "1:790469978744:android:5e743e723ff19a74e39b8d",
        "android_client_info": {
          "package_name": "a406.kingmaker.kingmaker"
        }
      },
      "oauth_client": [],
      "api_key": [
        {
          "current_key": "AIzaSyAoUTE40XMkEqqx8CfYdR_wmX_krHqn10Y"
        }
      ],
      "services": {
        "appinvite_service": {
          "other_platform_oauth_client": []
        }
      }
    },
    {
      "client_info": {
        "mobilesdk_app_id": "1:790469978744:android:83ecd779892e904fe39b8d",
        "android_client_info": {
          "package_name": "com.dollyanddot.kingmaker"
        }
      },
      "oauth_client": [],
    }
  ]
}
```

```

    "api_key": [
      {
        "current_key": "AIzaSyAoUTE40XMkEqqx8CfYdR_wmX_krHqn10Y"
      }
    ],
    "services": {
      "appinvite_service": {
        "other_platform_oauth_client": []
      }
    }
  }
],
"configuration_version": "1",
"installed":{"client_id":"245352649719-chfkk0at1o3gbha2uagqr9cymb568jek.apps.googleusercontent.com","project_id":"kingma

```

▼ android/key.properties

```

storePassword=123456
keyPassword=123456
keyAlias=key
storeFile=C:/Users/SSAFY/Desktop/key.jks

```

배포 확인

- git clone 해온 프로젝트 확인하는 장소
 - Jenkins관리 - System - 홈 디렉터리: `/var/lib/jenkins/workspace`

- 현재 실행 중인 docker container 확인

```
docker ps -a
```

- docker container log 확인

```
docker logs -f ${containerName}
```

- MariaDB 확인

```

//DB 접속
docker exec -it ${containerName} /bin/bash

//계정 변경
mariadb -u ${databaseName} -p

//Password 입력
${databasePassword}

//나가기
exit

```

- Redis 확인

```

//DB 접속
docker exec -it ${containerName} redis-cli

//Password 입력

```

```
auth ${databasePassword}

//나가기
exit
```

▼ 트러블슈팅

- `sudo apt-get upgrade` 했는데 젠킨스 에러 발생

```
Job for jenkins.service failed because the control process exited with error code.
See "systemctl status jenkins.service" and "journalctl -xe" for details.
invoke-rc.d: initscript jenkins, action "restart" failed.
• jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
   Active: activating (auto-restart) (Result: exit-code) since Thu 2023-11-02 10:45:48 KST; 6ms ago
   Process: 292823 ExecStart=/usr/bin/jenkins (code=exited, status=1/FAILURE)
   Main PID: 292823 (code=exited, status=1/FAILURE)
dpkg: error processing package jenkins (--configure):
 installed jenkins package post-installation script subprocess returned error exit status 1
Setting up sosreport (4.5.6-0ubuntu1~20.04.2) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.22) ...
Errors were encountered while processing:
 jenkins
E: Sub-process /usr/bin/dpkg returned an error code (1)
```

- `sudo systemctl status jenkins`

```
• jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: > Active: failed (Result: exit-code)
   Process: 294351 ExecStart=/usr/bin/jenkins (code=exited, status=1/FAILURE)
   Main PID: 294351 (code=exited, status=1/FAILURE)

Nov 02 10:59:36 ip-172-26-13-220 systemd[1]: jenkins.service: Scheduled restart jo>Nov 02 10:59:36 ip-172-26-13-220 system
```

- `sudo systemctl start jenkins`

```
sudo systemctl start jenkins
Job for jenkins.service failed because the control process exited with error code.
See "systemctl status jenkins.service" and "journalctl -xe" for details.
```

- `journalctl -xe`

```
-- Defined-By: systemd
-- Support: http://www.ubuntu.com/support
--
-- A start job for unit jenkins.service has finished with a failure.
--
-- The job identifier is 26963 and the job result is failed.
Nov 02 11:09:49 ip-172-26-13-220 systemd[1]: jenkins.service: Scheduled restart jo>-- Subject: Automatic restarting of a u
-- Defined-By: systemd
-- Support: http://www.ubuntu.com/support
--
-- Automatic restarting of the unit jenkins.service has been scheduled, as the res>-- the configured Restart= setting for
Nov 02 11:09:49 ip-172-26-13-220 systemd[1]: Stopped Jenkins Continuous Integratio>-- Subject: A stop job for unit jenkins
-- Defined-By: systemd
-- Support: http://www.ubuntu.com/support
--
-- A stop job for unit jenkins.service has finished.
--
-- The job identifier is 27036 and the job result is done.
Nov 02 11:09:49 ip-172-26-13-220 systemd[1]: jenkins.service: Start request repeat>Nov 02 11:09:49 ip-172-26-13-220 system
-- Defined-By: systemd
-- Support: http://www.ubuntu.com/support
--
```

```
-- The unit jenkins.service has entered the 'failed' state with result 'exit-code'.Nov 02 11:09:49 ip-172-26-13-220 systemd
-- Defined-By: systemd
-- Support: http://www.ubuntu.com/support
--
-- A start job for unit jenkins.service has finished with a failure.
--
-- The job identifier is 27036 and the job result is failed.
Nov 02 11:10:03 ip-172-26-13-220 kernel: [UFW BLOCK] IN=eth0 OUT= MAC=02:a2:8b:f2:>
```

- `sudo systemctl stop jenkins`

```
Warning: The unit file, source configuration file or drop-ins of jenkins.service changed on disk. Run 'systemctl daemon-reload'
```

- `sudo vi /lib/systemd/system/jenkins.service` 에서 다시 포트 변경
- `sudo systemctl daemon-reload`
- `sudo systemctl restart jenkins.service`
- `sudo systemctl status jenkins` 로 재실행 확인

▼ 외부 서비스 정보

- 소셜 인증(로그인)
 - Google OAuth
- Kakao OAuth

▼ DB 데이터

```
INSERT INTO category (category_nm)
VALUES
    ("집안일"),
    ("일상"),
    ("학습"),
    ("건강"),
    ("업무"),
    ("기타");

INSERT INTO notification_type (type)
VALUES
    ('MORNING'),
    ('TODO'),
    ('ROUTINE'),
    ('EVENING');

INSERT INTO reward(reward_nm, reward_cond, reward_msg, hidden)
VALUES
    ('밀림의 왕', '미달성한 일정 50개 도달', '몬스터로 인해 백성 250명이 떠났습니다', 1),
    ('나무늘보도 너보다는 부지런하겠다', '미달성한 일정 100개 도달', '여신님이 이리라고 주신 기회가 아닐 텐데요?', 1),
    ('백수의 왕', '아무 것도 등록하지 않고 30일을 보냄', '여신님이 일정 좀 등록하라고 하십니다.', 1),
    ('죽은 자의 소생', '마지막 수행으로부터 30일 이상 지난 경우', '도망이 아닌 도약이었다.', 1),
    ('나를 인정받는 국왕', '백성 수 3000명 도달', '왕국의 기반을 제대로 쌓아가고 있으시네요!', 0),
    ('여기가...맞나요?', '성으로 변경되는 단계에 도달', '왕국의 전성기가 시작되고 있습니다!', 0),
    ('오늘부터 황제', '만렙에 도달', '이제 여신님의 축복이 왕국에 깃들니다!', 0),
    ('알록달록한 세상', '카테고리별 달성 1개씩', '이제 너도 알잖아. 알록달록한 세상', 0),
    ('오잉!? 할 일의 상태가...!', '중요도 상인 일정 1개 최초 수행', '중요한 일도 하는 군주로 진화했습니다!', 0),
    ('내가 이제게 국왕?', '최초 가입', '건국을 축하합니다!
    몬스터를 처치하며 왕국을 번영시키세요!', 0),
    ('이게...내가 지닌 힘?', '첫 몬스터 처치', '처음으로 몬스터를 해치웠습니다!', 0),
    ('루틴 부자', '루틴 20개 생성', '원래 국왕이란 바쁜 법이죠!', 0),
```

```
('할 일 만수르', '할 일 100개 생성', '잠은 죽어서 자는 것이 아니라,  
잠을 못 자서 죽는 것이다.', 0),  
( '몬스터 파크 개장', '오늘 일정이 30개에 도달', '언제나 즐거움이 가득한 몬스터파크에 오신 것을 환영합니다!', 0),  
( '2310 국가 브랜드 평판 S급', '전 달 달성률이 90% 이상', '포브스 선정 국가 브랜드 평판 S급!', 0);
```

▼ 계정 정보



[Jenkins]

Port: 8081

Id: kingmaker

Password: kingmaker@406

Name: kingssafy



[MariaDB]

Port: 3306

Id: kingmaker

Password: kingmaker@406



[Redis]

Port: 6379

Password: kingmaker@406

▼ 시연 시나리오

1. 구글 회원 가입
2. 스토리 보여주기
3. 로그아웃 후 카카오 로그인 접속
4. 전 날 미달성 업무 알림 및 저녁 보고 알림 및 오늘 아침 알림 온 거 삭제하지 말고 보관하기
→ 일정 알림과 구분되도록
5. 미리 일정 등록 및 수행해서 캘린더 및 성과 달력 알록달록하게
→ 대신 중요도 없고 업무 카테고리 아닌 일정만! 시연 때 달성 보여줄 것!
6. 오늘의 일정 5개 정도 등록해 놓기
→ 처음 메인 페이지 들어갔을 때 몬스터 움직이는 거 보여주기
→ 캐릭터 말풍선 바뀌는 거 보여주기
7. 할 일 등록(업무)
→ 1시간 1-2분 후로 등록해서 일정 알림 오는 거 보여주기
8. 할 일 수행 버튼 클릭
→ 업적 달성: 한 번에 2개 이상 달성 되도록! 6단계 달성 + 카테고리별 수행
→ 등록했던 일정 중 2개만 남기고 다 수행
9. 메인 페이지 변경 사항 보여주며 설명
→ 몬스터/백성 수/레벨 변화

10. 마이페이지 변경 사항 보여주며 설명
 - 각 부분별 자세히 설명
11. 업적 리스트 변경사항 보여주며 설명
 - 들어가서 돌리면 업적 메시지 뜨는 거도 보여주기
12. 루틴 등록(중요도 상)
13. 이번엔 메인 페이지에서 캐릭터 클릭 후 수행하는 거 보여주기
 - 중요도별 글자색 차이 보여주기
 - 업적 달성: 중요도 상 수행
14. 메인 페이지 변경 사항
15. 마이페이지 변경 사항
16. 업적 리스트 변경사항
 - 최신순 정렬되는 거 보여주기