



C205: VeggieMeal (베지밀)

삼성SW청년아카데미 광주캠퍼스 7기

특화프로젝트 (22.08.22 - 22.10.07)

포팅 매뉴얼

담당 컨설턴트: 한기철

김윤주(팀장), 유이서, 김영서, 이기영, 정호진, 정지원

목차

| | |
|-------------------------------|----|
| 0. VeggieMeal (베지밀)이란? | 4 |
| 1. 프로젝트 기술스택 | 4 |
| 가. 이슈관리 | 4 |
| 나. 형상관리 | 4 |
| 다. 커뮤니케이션 | 4 |
| 라. 개발 환경 | 5 |
| 백엔드 기술스택 | 5 |
| 프론트엔드 기술스택 | 6 |
| 2. 프론트엔드 빌드 방법 | 6 |
| 가. Jenkins CI/CD | 6 |
| 나. Jenkinsfile | 7 |
| 다. Dockerfile | 8 |
| 3. 백엔드 세팅 방법 | 8 |
| 가. AWS EC2 DB 세팅 | 8 |
| 나. AWS EC2 Docker, Jenkins 세팅 | 10 |
| 다. AWS EC2 Nginx, HTTPS 세팅 | 11 |
| 라. AWS EC2 JDK 설치 | 11 |
| 마. 클러스터 Sqoop 설치 | 12 |
| 바. VM에 Hadoop 실습 환경 설정 | 15 |
| 사. VM에 Spark 실습 환경 설정 | 17 |
| 4. 백엔드 빌드 방법 | 18 |
| 가. Jenkins CI/CD | 18 |
| 나. Nginx + Https 설정 | 28 |
| 다. HDFS | 29 |
| 라. Spark | 31 |
| 마. Sqoop | 31 |
| 바. 로컬에서 백엔드 서버 빌드 | |

| | |
|------------------|----|
| 사. 파이썬 | 32 |
| 5. 외부 서비스 | 33 |
| 가. 카카오 | 33 |
| 나. 유튜브 | 34 |
| 다. 네이버 | 35 |

0. VeggieMeal(베지밀)이란?

VeggieMeal은 '채식주의자'란 'Veggie'와 '식사'라는 뜻의 'Meal'이 합쳐진 용어입니다. 본 서비스는 채식에 대한 접근성을 높이고 관련 재료의 물가 정보를 제공합니다. 사용자는 채식주의의 타입을 선택하거나 냉장고에 보유 중인 재료를 선택하여 맞춤 레시피를 찾을 수 있습니다. 레시피를 상세 클릭하여 구매하고자 하는 품목을 장바구니에 담을 수 있습니다. 장바구니에서 현재 마트에서 판매 중인 품목 관련 상품 정보를 확인한 뒤 비교하여 선택할 수 있습니다. **VeggieMeal**은 전국 도매시장 농수산품의 경매 데이터를 일별로 수집하여 평균가 및 최고가, 최저가를 분석합니다. 사용자는 분석 결과를 그래프와 표의 형태로 확인할 수 있습니다. 더 건강하고 더 경제적인 식사, 베지밀이 지향하는 가치입니다.

1. 프로젝트 기술스택

가. 이슈관리

- Jira

나. 형상관리

- Gitlab

다. 커뮤니케이션

- Mattermost
- Notion

라. 개발 환경

- OS: Ubuntu 22.04.1 LTS
- **사용 IDE:**
 - IntelliJ IDEA 2022.1.3
 - Visual Studio Code : 1.70.2v
 - UI/UX: Figma
 - Jupyter Notebook
- **백엔드 기술스택**
 - Springboot : 2.7.3
 - MariaDB : mariadb 10.3.34
 - AWS : ubuntu 20.04.4 LTS
 - Jenkins : 2.361.1
 - Docker : 20.10.18
 - Openjdk : 1.8.0
 - spring: gradle
 - nginx : nginx/1.18.0 (Ubuntu)
 - Hadoop : 3.2.1
 - Spark : 3.2.1
 - Sqoop : 1.4.7
 - Kafka : 3.2.3
 - Python : 3.9.12

● 프론트엔드 기술스택

- node.js : v16.15.1(LTS)
- npm : 8.11.0v
- next : 12.3.0v
- react : 18.2.0v
- recoil : 0.7.5v
- TypeScript : 4.8.3v
- Sass : 1.54.9v
- react-query : 3.39.2v

2. 프론트엔드 빌드 방법

가. Jenkins CI/CD

- 1) 백엔드와 방법 동일
- 2) 경로에 도커 파일과 젠킨스 파일 저장

나. Jenkinsfile

```
pipeline {
  agent any

  stages {
    stage("git"){ //git에 브랜치 url, credentialsId는 젠킨스에 등록된 인증으로
      steps{
        git branch: 'fe/feature', credentialsId: 'gitlabIDPW', url: 'https://lab.ssafy.com/s07-
        bigdata-dist-sub2/S07P22C205.git'
      }
    }
    stage('build') {
      steps{
        sh "ls -a"
        dir('Front') {
          dir('veggiemeal'){
            script{
              try{
                sh "docker build -t front ."
              } catch(e){
                echo "fail build"
              }
            }
            script{ //이미 실행 중인 컨테이너가 있으면 중지 후 삭제
              try{
                sh "docker stop front"
                sh "docker rm front"
              } catch(e){
                echo "container none"
              }
            }
          } //3000 포트에서 front이라는 이미지를 front이라는 컨테이너이름으로 설정해서 실행
          sh "docker run -d -p 3000:3000 --name front front "
        }
      }
    }
  }
}
```

다. Dockerfile

```
FROM node:14.19.2 as builder

WORKDIR /var/jenkins_home/workspace/frontend/Front/veggiemeal

COPY . ./
RUN npm i
RUN npm run build

CMD ["npm", "start"]
```

3. 백엔드 세팅 방법

가. AWS EC2 DB 세팅

1) MariaDB 설치

```
sudo apt update
sudo apt install mariadb-server -y
```

2) MariaDB 버전 확인, 실행 확인

```
mariadb -V #버전확인
sudo systemctl status mariadb
```


3) 권한 설정

```
sudo mysql -u root #DB 접속
create user '계정 아이디'@'%' identified by '비밀번호'; #계정생성
create database ssafy; #DB 생성

# ssafy에 대하여 ssafy로 접속하는 모든 위치(%)에서 모든 권한 부여(all privileges)
grant all privileges on ssafy.* to 'ssafy'@'%';
```

4) 원격 접속을 위한 설정 변경

```
# DB 외부 접속 허용하기
cd /etc/mysql/maria.conf.d sudo nano 50-server.cnf
#bind-address = 127.0.0.1 부분 -> #주석처리#
```

5) 데이터 베이스 타임존 설정

```
cd /etc/mysql/maria.conf.d
sudo nano 50-server.cnf #default-time-zone='+9:00' 추가
```

6) 설정 변경 적용을 위한 재시작

```
sudo service mysql restart
```

나. AWS EC2 Docker, Jenkins 세팅

1) 도커 설치

```
sudo curl https://get.docker.com/ > dockerinstall && chmod 777 dockerinstall && ./dockerinstall
```

2) 도커에 젠킨스 이미지 설치와 컨테이너 실행

```
sudo docker run -d -p 9090:8080 -v /var/jenkins:/var/jenkins_home -v /var/run/docker.sock:/var/run/docker.sock --name jenkins -u root jenkins/jenkins:lts-jdk11
```

3) 도커 젠킨스 컨테이너 내에 접속

```
sudo docker exec -it jenkins bash
```

4) 젠킨스 컨테이너에 도커 설치

```
curl https://get.docker.com/ > dockerinstall && chmod 777 dockerinstall && ./dockerinstall
```

다. AWS EC2 Nginx, HTTPS 세팅

1) nginx 설치

```
sudo apt-get update #운영체제 업데이트
sudo apt install nginx -y #nginx 설치
nginx -v #nginx 버전 확인
```

2) certbot 설치

```
sudo add-apt-repository ppa:certbot/certbot #certbot 설치
sudo apt-get update #업데이트
apt-get install python3-certbot-nginx #apt-get 이용해서 certbot 설치

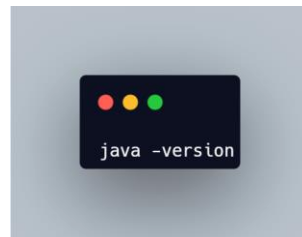
#도메인에 대한 인증서 발급(nginx가 실행 중이어 함)
certbot certonly --nginx -d 도메인 이름
sudo certbot renew --dry-run #인증서 자동 갱신 설정
```

라. AWS EC2 JDK 설치

1) JDK 설치

```
sudo apt-get update
sudo apt-get install openjdk-8-jdk
```

2) JDK 버전 확인

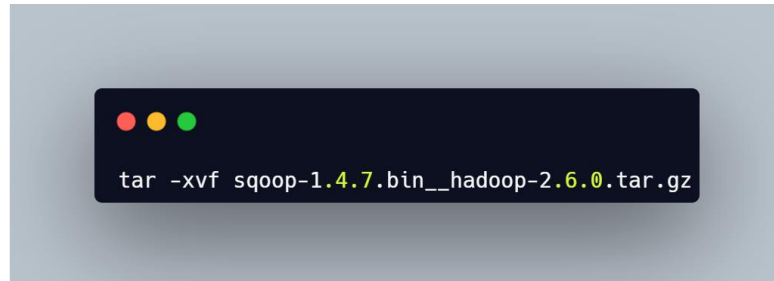


마. 클러스터 Sqoop 설치

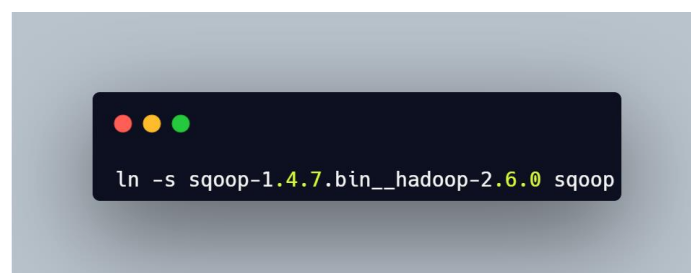
1) Sqoop 설치



2) Sqoop 압축 해제



3) 심볼릭 링크 설정



4) 환경 변수 설정

```
vi ~/.bashrc #밑에 추가 후 저장
export SQOOP_HOME=/home/사용자/sqoop
export SQOOP_CONF_DIR=$SQOOP_HOME/conf
export PATH=$PATH:$SQOOP_HOME/bin source ~/.bashrc #환경변수 적용
```

5) Sqoop 설정 예시 파일 복사

```
cp sqoop/conf/sqoop-env-template.sh sqoop/conf/sqoop-env.sh
```

6) 해당 디렉토리 이동

```
cd sqoop/conf/
```

7) 설정 파일 수정

```
vi sqoop-env.sh ##밑에 내용 추가 후 저장
export HADOOP_HOME=/home/사용자/hadoop
export HADOOP_COMMON_HOME=/home/사용자/hadoop
export HADOOP_MAPRED_HOME=/home/사용자/hadoop
```

8) Mysql 커넥터 설치

```
wget https://downloads.mysql.com/archives/get/p/3/file/mysql-connector-java-5.1.46.tar.gz
```

9) 커넥터 압축 해제

```
tar -xvf mysql-connector-java-5.1.46.tar.gz
```

10) sqoop의 lib 폴더에 jar 옮기기

```
cd mysql-connector-java-5.1.46/  
mv mysql-connector-java-5.1.46-bin.jar /home/hadoop01/sqoop/lib/
```

11) 의존 관계 commons lang 다운

```
## http://commons.apache.org/proper/commons-lang/download_lang.cgi에서 2.6 버전 다운을 찾아서 링크 복사  
wget 링크
```

12) commons lang 압축해제

```
tar -xvf commons-lang-2.6-bin.tar.gz
```

13) 폴더로 이동 후 파일 복사

```
cd commons-lang-2.6  
cp commons-lang-2.6.jar /home/사용자/sqoop/lib
```

14) 겹치는 기존 파일 이름 변경

```
cd /home/사용자/sqoop/lib/  
ll common*  
mv commons-lang3-3.4.jar commons-lang3-3.4.jar.bak
```

바. VM에 Hadoop 실행 환경 설정

1) 다운로드

```
wget http://kdd.snu.ac.kr/~kddlab/Project.tar.gz
```

2) 압축 해제와 하둡 환경 설정 설치

```
tar xzf Project.tar.gz #압축해제
sudo chown -R hadoop:hadoop Project #권한 설정
cd Project sudo mv hadoop-3.2.2 /usr/local/hadoop #폴더 이동
./set_hadoop_env.sh # 하둡 셋 실행
source ~/.bashrc #하둡 셋 적용
```

3) ssh 키 생성

```
ssh-keygen -t rsa -P "" #키 생성
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys #키복사
source ~/.bashrc #환경 설정 적용
hadoop namenode -format #네임노드 포맷
```

4) dfs 시작

```
start-mapred.sh
```

5) 수행 확인

```
jps
```


사. VM에 Spark 실행 환경 설정

1) 설치 및 압축해제

```
wget 다운로드 받을 스파크 링크  
sudo mkdir /opt/spark #폴더 생성  
tar -xvf 폴더 이름 # 압축해제
```

2) 환경 변수 권한 설정

```
sudo nano .bashrc #환경 변수 파일 접근  
# 아래 구문 작성 후 저장  
export SPARK_HOME=/opt/spark  
export PATH="$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin"  
  
#변경 사항 적용  
source ~/.bashrc  
  
#권한 설정  
sudo chmod -R 777 opt/spark
```

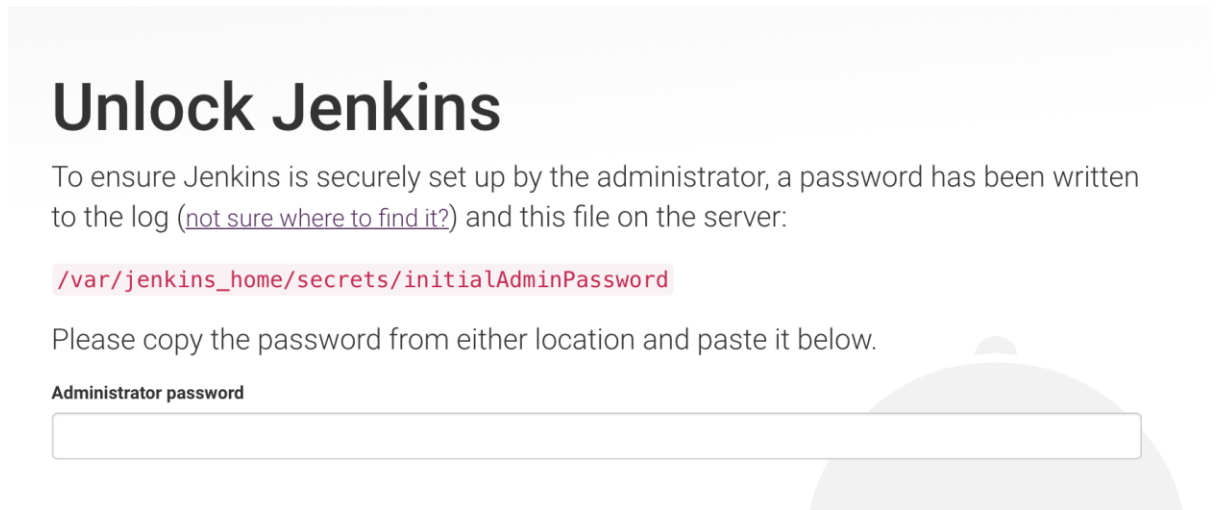
3) 실행

```
start-master.sh #마스터 실행  
start-worker.sh #워커 실행
```

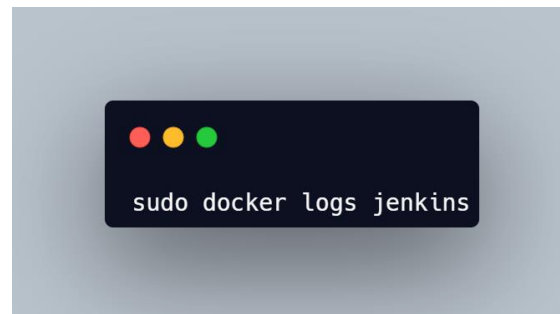
4. 백엔드 빌드 방법

가. Jenkins CI/CD

1) 링크접속



2) 도커 로그에서 비밀번호 확인 후 위 화면에 입력



```
2019-08-23 12:47:56.074+0000 [id=26] INFO jenkins.install.SetupWizard#init:

*****
*****
*****

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

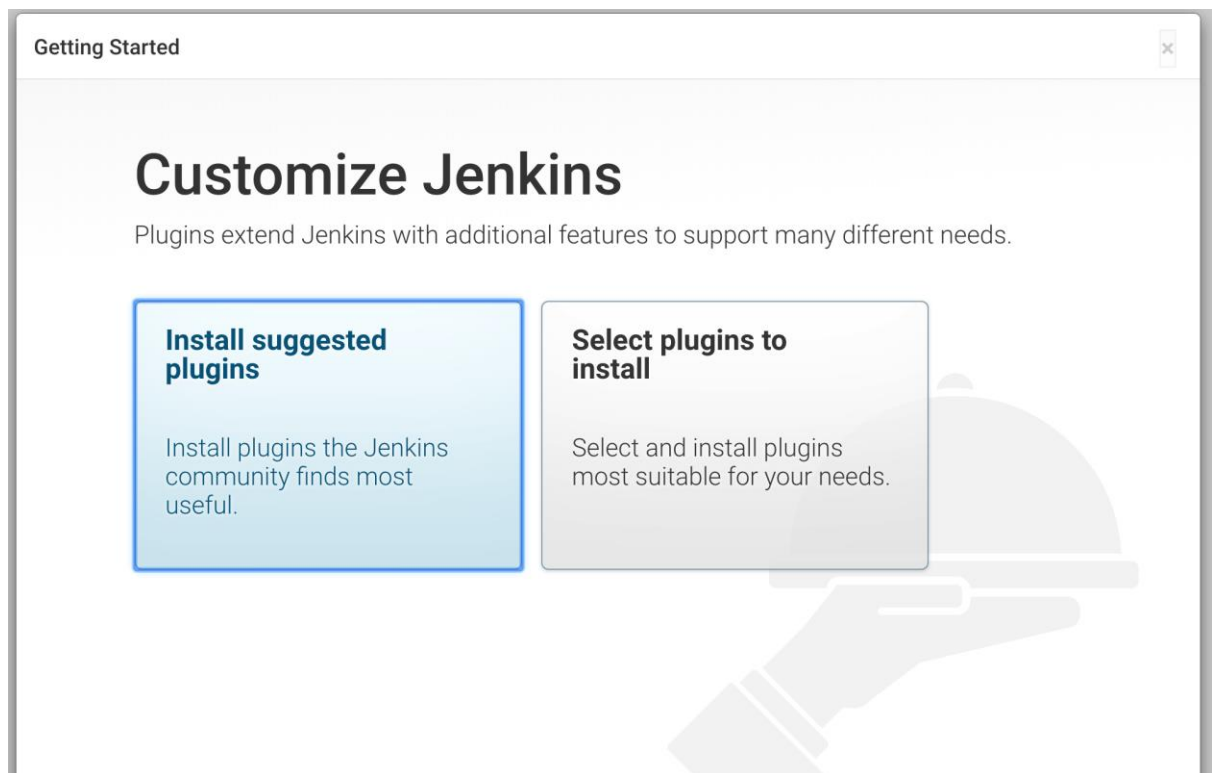
effbc3919412407085230355cc81bc4b

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*****
*****
*****

2019-08-23 12:47:56.082+0000 [id=40] INFO h.m.DownloadService$Downloadable#load: Obtained the updated data file fo
2019-08-23 12:47:56.083+0000 [id=40] INFO hudson.util.Retrier#start: Performed the action check updates server suc
2019-08-23 12:47:56.094+0000 [id=40] INFO hudson.model.AsyncPeriodicWork$1#run: Finished Download metadata. 18,487
2019-08-23 12:48:03.654+0000 [id=26] INFO hudson.model.UpdateSite#updateData: Obtained the latest update center da
2019-08-23 12:48:04.075+0000 [id=26] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
2019-08-23 12:48:04.161+0000 [id=19] INFO hudson.WebAppMain$3#run: Jenkins is fully up and running
```

3) 젠킨스 플러그인 설치, 설치 후 관리자 계정 생성



4) 젠킨스 관리에서 플러그인 관리 클릭



플러그인 관리

Jenkins의 기능을 확장하기 위한 플러그인을 추가, 제거, 사용, 미사용으로 설정할 수 있습니다.

5) 설치 가능에서 도커, git 관련 플러그인 설치

- git

이름 ↓

Generic Webhook Trigger Plugin 1.84

Can receive any HTTP request, extract any values from JSON or XML and trigger a job with those values available as variables. Works with GitHub, GitLab, Bitbucket, Jira and many more.

[Report an issue with this plugin](#)

Git client plugin 3.11.2

Utility plugin for Git support in Jenkins

[Report an issue with this plugin](#)

Git plugin 4.11.4

This plugin integrates Git with Jenkins.

[Report an issue with this plugin](#)

GitHub API Plugin 1.303-400.v35c2d8258028

This plugin provides GitHub API for other plugins.

[Report an issue with this plugin](#)

GitHub Branch Source Plugin 1677.v731f745ea_0cf

Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.

[Report an issue with this plugin](#)

GitHub plugin 1.34.5

This plugin integrates GitHub to Jenkins.

[Report an issue with this plugin](#)

Gitlab API Plugin 5.0.1-78.v47a_45b_9f78b_7

This plugin provides GitLab API for other plugins.

[Report an issue with this plugin](#)

GitLab Authentication plugin 1.16

This is the an authentication plugin using gitlab OAuth.

[Report an issue with this plugin](#)

This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.

GitLab Plugin 1.5.35

This plugin allows GitLab to trigger Jenkins builds and display their results in the GitLab UI.

[Report an issue with this plugin](#)

This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.

Pipeline: GitHub Groovy Libraries 38.v445716ea_edda_

Allows Pipeline Groovy libraries to be loaded on the fly from GitHub.

[Report an issue with this plugin](#)

- docker

이름 ↓

Docker API Plugin 3.2.13-37.vf3411c9828b9

This plugin provides **docker-java** API for other plugins.

[Report an issue with this plugin](#)

This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.

Docker Commons Plugin 1.19

Provides the common shared functionality for various Docker-related plugins.

[Report an issue with this plugin](#)

Docker Pipeline 521.v1a_a_dd2073b_2e

Build and use Docker containers from pipelines.

[Report an issue with this plugin](#)

Docker plugin 1.2.9

This plugin integrates Jenkins with **Docker**

[Report an issue with this plugin](#)

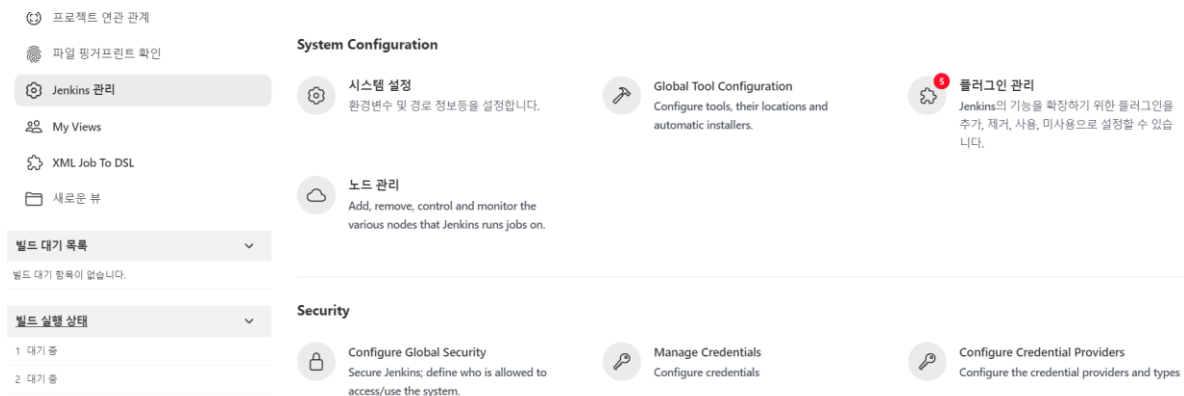
This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.

6) 설치 후 재시작



7) jenkins에 깃랩 계정 등록

- jenkins 관리에서 Manage Credentials 클릭



- Domains global에서 Add Credentials 클릭
- Credentials 생성 (Username : git 아이디, Password : git 패스워드, ID: credentials의 이름 설정, Description : credentials에 대한 설명)

New credentials

Kind

Username with password

Scope ?

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

Username ?

stopone2639

☐ Treat username as secret ?

Password ?

ID ?

gitlabID

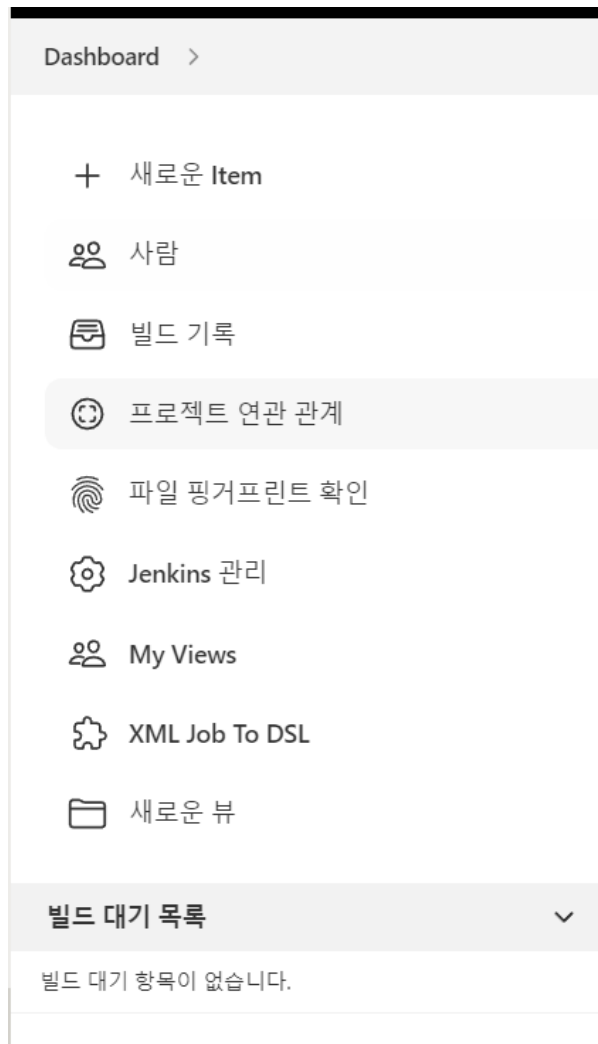
Description ?

깃랩 계정

Create

8) 파이프 라인 생성

- 메인 화면에서 새로운 Item 클릭



- Pipeline으로 생성

Enter an item name

» Required field



Freestyle project

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



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

다양한 환경에서의 테스트, 플랫폼 특성 빌드, 기타 등등 처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.



Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.



Organization Folder

Creates a set of multibranch project subfolders by scanning for repositories

- 생성한 파이프라인 클릭 후 구성 클릭

↑ Back to Dashboard

Pipeline backend

☰ Status

</> Changes

▶ 지금 빌드

⚙ 구성

🗑 Pipeline 삭제

🔍 Full Stage View

✎ Rename

❓ Pipeline Syntax



Recent Changes

Stage View

| | Declarative: Checkout SCM | git | build |
|---|------------------------------|-------|-------|
| Average stage times: (Average full run time: ~27s) | 708ms | 437ms | 25s |

- 빌드 트리거 설정

Build Triggers

☐ Build after other projects are built ?
☐ Build periodically ?
☒ Build when a change is pushed to GitLab. GitLab webhook URL: http://t7c208.p.ssafy.io:9090/project/backend ?

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...

- 파이프라인 설정 (Repository url에 깃 클론 URL 작성, Branch Specifier에 브랜치 설정, Script Path에 브랜치의 젠킨스 파일 위치를 설정)

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://lab.ssafy.com/s07-webmobile1-sub1/S07P11C208.git

Credentials ?

stopone2639/***** (깃랩계정)

+ Add

고급...

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/back

Add Branch

Repository browser ?

(자동)

Additional Behaviours

Add

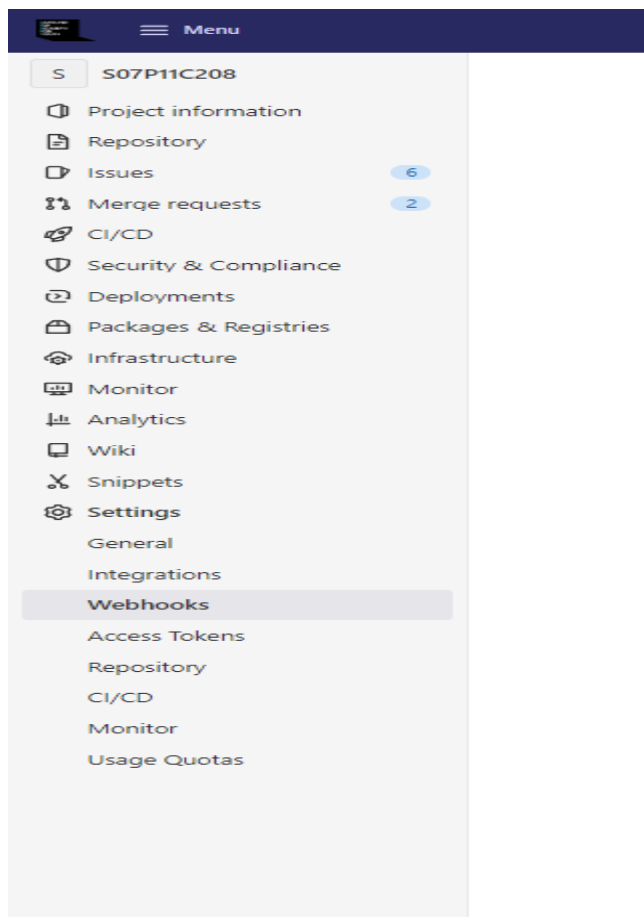
Script Path ?

./backend/jenkinsfile

☒ Lightweight checkout ?

[Pipeline Syntax](#)

- git 프로젝트에서 웹훅 설정(URL, 토큰은 젠킨스 파이프라인에 빌드 트리거에서 확인 가능, 이벤트 브랜치 설정 시 그 브랜치에서 이벤트 발생 시에만 자동 빌드 되도록 할 수 있음(백이라서 back 브랜치 푸쉬시 빌드 설정))



Q 검색 페이지

URL

<http://i7c208.p.ssafy.io:9090/project/backend>

URL에 하나 이상의 특수 문자가 포함된 경우 퍼센트로 인코딩되어야 합니다.

53e68943960f08a0630b38cd820a165d

수신된 페이로드를 확인하는 데 사용됩니다. X-Gitlab-Token HTTP 헤더 의 요청과 함께 전송됩니다.

☒ 푸시 이벤트

[back](#)

저장소로 푸시합니다.

- ☐ 푸시 이벤트 태그
새 태그가 저장소에 푸시됩니다.

- ☐ **코멘트**
문제 또는 병합 요청에 주석이 추가됩니다.

- ☐ 기밀 댓글
비밀 문제에 댓글이 추가되었습니다.

- ☐ 이슈 이벤트
문제가 생성, 업데이트, 종료 또는 재개되었습니다.

- ☐ 기밀 문제 이벤트
기밀 문제가 생성, 업데이트, 종료 또는 재개되었습니다.

- ☐ 병합 요청 이벤트
병합 요청이 생성, 업데이트 또는 병합됩니다.

- ☐ **작업 이벤트**
작업의 상태가 변경됩니다.

- ☐ **파이프라인 이벤트**
파이프라인의 상태가 변경됩니다.

- ☐ 위키 페이지 이벤트
Wiki 페이지가 생성되거나 업데이트됩니다.

- 깃 프로젝트에 jenkins 파일 작성(파이프 라인에서 설정한 경로에 파일 생성 해야함)

- 백 엔드 젠킨스 파일 내용

```
pipeline {
  agent any

  stages {
    stage("git"){ //git에 브랜치 url, credentialsId는 젠킨스에 등록된 인증으로
      steps{
        git branch: 'be/feature', credentialsId: 'gitlabIDPW', url: 'https://lab.ssafy.com/s07-bigdata-dist-sub2/S07P22C205.git'
      }
    }
    stage('build') {
      steps{
        sh "ls -a"
        dir('Back') { //gradle 권한 설정 후 gradle로 도커 이미지 빌드하는 명령어 실행
          script{
            try{
              sh "chmod +x gradlew"
              sh "./gradlew bootBuildImage --imageName=spring"
            } catch(e){
              echo "fail build"
            }
          }
        }
        script{ //이미 실행 중인 컨테이너가 있으면 중지 후 삭제
          try{
            sh "docker stop spring"
            sh "docker rm spring"
          } catch(e){
            echo "container none"
          }
        }
        //8083 포트에서 spring이라는 이미지를 spring이라는 컨테이너이름으로 설정해서 실행
        sh "docker run -d -p 8083:8083 --name spring spring "
      }
    }
  }
}
```

나. Nginx + Https 설정

1) nginx 설정 수정

```
sudo vim /etc/nginx/sites-available/default
```

2) 아래 내용으로 위파일 수정

```

server {
    listen 80; #80포트로 받을 때
    server_name j7c205.p.ssafy.io; #도메인주소, 없을경우 localhost
    return 301 https://j7c205.p.ssafy.io$request_uri; #리다이렉트 https url로
}
server {
    listen 443 ssl http2; # 443 포트로 받을 때
    server_name j7c205.p.ssafy.io;

    # ssl 인증서 적용하기
    ssl_certificate /etc/letsencrypt/live/j7c205.p.ssafy.io/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/j7c205.p.ssafy.io/privkey.pem;

    location / {
        proxy_pass http://j7c205.p.ssafy.io:3000; # /로 들어올 시에 프론트 포트에 연결 해주고 https 설정
        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;
    }
    location /api {
        proxy_pass http://j7c205.p.ssafy.io:8083; # /api로 들어올 시에 백 포트에 연결 해주고 https 설정
        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 $scheme;
    }
}
server {
    if ($host = j7c205.p.ssafy.io) { #호스트가 우리 호스트면
        return 301 https://$host$request_uri; #호스트에 request_uri를 https로 리다이렉트
    } # managed by Certbot

    listen 80;
    return 404; # managed by Certbot
}

```

3) nginx 재시작

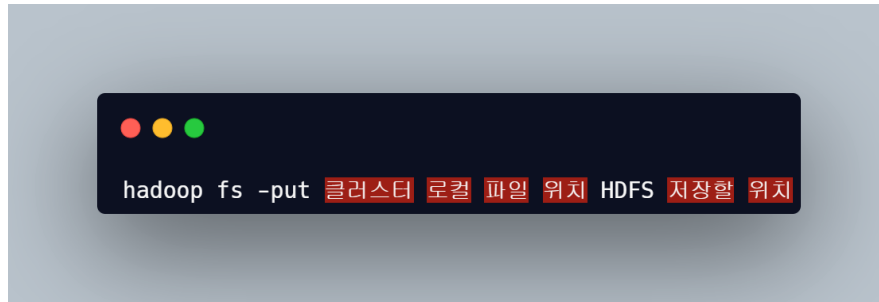
```

nginx 재시작
sudo nginx -s stop #nginx 실행 중지
sudo nginx #nginx 실행

```

다. HDFS

1) HDFS에 데이터 적재



2) HDFS에서 데이터 삭제



3) HDFS에서 데이터 조회



4) HDFS에서 폴더 생성



5) hadoop jar 파일 실행

```
hadoop jar jar파일이름 클래스위치 인풋데이터 아웃풋데이터
```

라. Spark

1) Spark jar 파일 실행

```
spark-submit --class 클래스경로와이름 jar파일 이름 HDFS 인풋 데이터 아웃풋 경로  
ex) spark-submit --class VeggieMeal VeggieMealScala.jar data/all output/Spark/all
```

마. Sqoop

1) Sqoop Export

```
sqoop export --connect jdbc:mysql://경로/DB이름 --table 테이블 --columns 컬럼들 이름 --export-dir 내보낼 경로  
--username db계정 아이디 --P
```



```
sqoop export --connect jdbc:mysql://j7c205.p.ssafy.io:3306/ssafy --table deal --columns  
deal_date,large,medium,small,origin,max,min,price --export-dir output/Spark/all --username ssafy --P
```

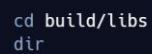
바. 로컬에서 백엔드 서버 빌드

1) gradle 빌드




```
gradlew clean build
```

2) 빌드 파일 확인



```
cd build/libs  
dir
```

3) 서버 실행



```
java -jar 파일 이름.jar
```


사. 파이썬

```
pip install --upgrade pip
pip install schedule, pymysql, selenium, requests, beautifulsoup4
nohup python mart-crawler.py &
```

5. 외부 서비스


가. 카카오

- 내 주변 마트 찾기 기능을 위해 카카오맵 API를 활용하였습니다. 카카오맵 API의 지도 생성 및 키워드 검색 기능 등을 통해 사용자의 GPS 정보를 기반으로 주변 마트 정보 및 지도, 길찾기 링크를 제공합니다.
- 카카오맵 API를 사용하기 위해서는 kakao developer에 카카오 계정으로 로그인해 애플리케이션의 기본 정보를 등록해주어야 합니다.



기본 정보

[수정](#)

| | |
|-------|---|
| 앱 아이콘 |  |
| 앱 이름 | 베지밀 |
| 사업자명 | 자란다 |

사용자가 카카오 로그인을 할 때 표시되는 정보입니다. 정보가 정확하지 않은 경우 서비스 이용이 제한될 수 있습니다.

- 플랫폼의 도메인을 등록해주면 카카오맵 API를 사용할 수 있습니다.

| Web | | 삭제 | 수정 |
|---------|--|----|----|
| 사이트 도메인 | http://localhost:3000 https://j7c205.p.ssafy.io | | |

• 카카오 로그인 사용 시 Redirect URI를 등록해야 합니다. [등록하러가기](#)

나. 유튜브

- 서비스의 관련 레시피 상세보기에서 관련 레시피 영상을 불러오기 위해 구글 API를 이용하였습니다.
- 프로젝트 등록을 한 뒤, API키를 등록합니다.
- 승인된 도메인을 사용하지 않을 경우 API는 테스트 버전만 사용가능합니다.
- 이를 위해 도메인을 https로 등록하였습니다.


애플리케이션 제한사항

애플리케이션 제한사항은 API 키를 사용할 수 있는 웹사이트, IP 주소 또는 애플리케이션을 제어합니다. 키별로 애플리케이션 제한사항 1개를 설정할 수 있습니다.

- ☐ 없음
- ☒ HTTP 리퍼러(웹사이트)
- ☐ IP 주소(웹 서버, 크론 작업 등)
- ☐ Android 앱
- ☐ iOS 앱

웹사이트 제한사항

키 사용량 요청을 지정된 웹사이트로 제한합니다.

 비워 두면 API 키가 모든 웹사이트의 요청을 수락합니다.

| | |
|-----------------------|---|
| j7c205.p.ssafy.io/* | ▼ |
| localhost:3000/* | ▼ |
| 항목 추가 | |

API 제한사항

API 제한사항은 이 키를 호출할 수 있는 사용 설정된 API를 지정합니다.

- ☐ 키 제한 안함
이 키는 모든 API를 호출할 수 있습니다.
- ☒ 키 제한

API 1개 ▼

선택한 API:

YouTube Data API v3

- 웹사이트 제한사항은 서버 주소 및 프론트 서버의 주소를 의미합니다.
- API 제한사항은 google cloud API 라이브러리 중 필수적으로 필요한 YouTube API만 허용하였습니다.

다. 네이버

- 물가 관련 뉴스 기능을 위해 네이버 검색 API를 활용하였습니다.
- 타 API와 동일하게 네이버 개발자 센터(NAVER Developers)에 애플리케이션을 등록해주어야 합니다.

- 이후 발급된 ClientID와 Client Secret를 Header에 넣어 요청을 보내야 합니다.

내 애플리케이션

비지밀

애플리케이션 등록

API 제휴 신청

계정 설정

비지밀

| | | | | | |
|----|--------|------|--------|--------|------------------|
| 개요 | API 설정 | 멤버관리 | 로그인 통계 | API 통계 | Playground(Beta) |
|----|--------|------|--------|--------|------------------|

애플리케이션 정보

| | |
|---------------|--------------------------------|
| Client ID | <div></div> |
| Client Secret | <div>.....</div> <div>보기</div> |