

배포가이드

[기술스택](#)

[백엔드](#)

[프론트엔드](#)

[데이터 정제](#)

[Infra](#)

[협업](#)

[버전 확인](#)

[timezone 설정](#)

[JAVA 설치](#)

[Docker/Docker Compose 설치](#)

[Mysql 설치 및 설정](#)

[redis 설치 및 설정](#)

[nginx 설치](#)

[https 설정](#)

[nginx proxy 설정](#)

[/etc/nginx/sites-available/server1](#)

[jenkins 설치 및 설정](#)

[docker-compose.yaml](#)

[Jenkins 접속 및 초기 설정](#)

[jenkins - Gitlab 자동배포](#)

[molaeng_back/Dockerfile](#)

[molaeng_front/Dockerfile](#)

[molaeng_front/default.conf](#)

[docker-compose.yaml](#)

기술스택

백엔드

- IDE : IntelliJ 2022.2
- JAVA : 8
- Framework : Spring boot 2.7.4
- Build : Gradle 7.5
- WAS : Tomcat
- DBMS : MySql 8.0
- DB API : JPA

프론트엔드

- IDE : VS Code 1.70.1
- Framework :
 - Vue2 5.0.8
 - Vuetify 2.6.0
 - Chart.js 3.9.1

데이터 정제

- JSoup
- Python
- Hadoop

Infra

- AWS
- Docker
- Nginx
- Jenkins

협업

- Git
- Jira

버전 확인

```
# Ubuntu 버전 확인
lsb_release -a
->Ubuntu 20.04 LTS

# 커널 버전 확인
uname -r
->5.4.0-1018-aws
```

timezone 설정

```
# 현재시간 확인
date

# timezone 확인
timedatectl

# 한국 timezone 확인
timedatectl list-timezones | grep Seoul
->한국 timezone : Asia/Seoul

# timezone 변경
sudo timedatectl set-timezone Asia/Seoul
```

JAVA 설치

```
# 패키지 업데이트 및 JAVA8 설치
sudo apt-get update
sudo apt-get install openjdk-8-jdk

# JAVA 버전 확인
java -version
->1.8.0_342
```

Docker/Docker Compose 설치

```
# 패키지 업데이트
# apt-transport-https:패키지 관리자가 https를 통해 데이터 및 패키지에 접근할 수 있도록 한다.
# ca-certificates:ca-certificate는 certificate authority에서 발행되는 디지털 서명. SSL 인증서
의 PEM 파일이 포함되어 있어 SSL 기반 앱이 SSL 연결이 되어있는지 확인할 수 있다.
# curl:HTTP, HTTPS, SCP, SFTP 및 FTP 등 지원되는 프로토콜 중 하나를 사용하여 데이터를 다운로드하거나
업로드할 수 있다.
# gnupg:GNU Privacy Guard. GPG 라고도 한다. GPG 키 사용을 위해 설치한다.
# lsb-release:리눅스 배포판 버전과 정보를 확인할 수 있다.
sudo apt-get update
sudo apt-get install apt-transport-https ca-certificates curl gnupg lsb-release

# GPG 키 추가
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/a
pt/keyrings/docker.gpg
echo \
    "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] http
s://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/nul
l

# Docker/Docker Compose 설치
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

```

# Docker 버전 확인
docker --version
->20.10.18, build b40c2f6

# Docker Compose 버전 확인
docker compose version
->v2.10.2

# Docker 실행
sudo service docker start

# Docker 설치 확인
sudo docker run hello-world
->정상 설치시 Hello from Docker! 출력됨

# Docker 컨테이너 목록 확인
sudo docker ps -a

# Docker 컨테이너 종료
sudo docker stop [컨테이너 이름]

# Docker 컨테이너 삭제
sudo docker rm [컨테이너 이름]

# Docker 이미지 목록 확인
sudo docker images

# Docker 이미지 삭제
sudo docker rmi [이미지 이름]

# docker.sock 권한 설정 (jenkins 내에서 docker 명령어 사용하기 위함)
sudo chmod 666 /run/docker.sock

# docker-compose 권한 설정 (jenkins 내에서 docker compose 명령어 사용하기 위함)
sudo chmod 777 /usr/libexec/docker/cli-plugins/docker-compose

```

Mysql 설치 및 설정

```

# mysql 이미지 다운로드
sudo docker pull mysql

# mysql 컨테이너 생성 및 실행
sudo docker run -d -p 3306:3306 -e MYSQL_ROOT_PASSWORD='molaengA604!' -v ~/mysql:/var/lib/mysql --name mysql-container mysql --character-set-server=utf8mb4 --collation-server=utf8mb4_general_ci

# mysql 컨테이너 접속
sudo docker exec -it mysql-container bash
->mysql -u root -p
->molaengA604!

# mysql root 이름 변경
use mysql
update user set user='a604' where user='root';

```

```

# schema 생성
CREATE DATABASE molaeng DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;

# 계정 생성
create user 'ssafy'@localhost identified by 'ssafyA604!';
create user 'ssafy'@'%' identified by 'ssafyA604!';

# 계정 권한 부여
grant all privileges on molaeng.* to 'ssafy'@localhost;
grant all privileges on molaeng.* to 'ssafy'@'%';

# mysql 사용자 조회
use mysql
select user, host from user;

# mysql 사용자 권한 확인
show grants for 'ssafy'@localhost;
show grants for 'ssafy'@'%';

# mysql timezone 확인
select @@global.time_zone, @@session.time_zone, @@system_time_zone;

# mysql timezone 변경
set time_zone='Asia/Seoul';
set global time_zone='Asia/Seoul';

# mysql 현재시간 확인
select now();

# mysql character set 확인
show variables like 'c%';

# 변경사항 적용
flush privileges;

```

redis 설치 및 설정

```

# redis 이미지 다운로드
sudo docker pull redis

# mysql 컨테이너 접속 후 버전 등 정보 확인
sudo docker exec -it redis-container bash
->redis-cli
->info

```

nginx 설치

```

# 패키지 업데이트 및 nginx 설치
sudo apt-get update
sudo apt-get install nginx

```

```
# nginx 버전 확인
sudo nginx -v
->nginx/1.18.0 (Ubuntu)

# nginx 실행
sudo service nginx start
```

https 설정

```
# core 설치 및 최신 버전 업데이트
sudo snap install core
sudo snap refresh core

# 기존 certbot이 있다면 삭제
sudo apt remove certbot

# certbot 설치
sudo snap install --classic certbot

# certbot 명령을 로컬에서 실행할 수 있도록 심볼릭링크 연결
sudo ln -s /snap/bin/certbot /usr/bin/certbot

# 인증서 발급 및 nginx 설정
sudo certbot --nginx
->이메일 입력 : adoramilky@gmail.com
->서비스약관 동의 : y
->개인정보 제공 동의 : n
->도메인 이름 확인 : j7a604.p.ssafy.io

공개키 경로 : /etc/letsencrypt/live/j7a604.p.ssafy.io/fullchain.pem
비밀키 경로 : /etc/letsencrypt/live/j7a604.p.ssafy.io/privkey.pem

# nginx 재시작
sudo service nginx restart
```

```

ubuntu@ip-172-26-12-79:~$ sudo certbot --nginx
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): adoramilky@gmail.com

- - - - -
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017-w-v1.3-notice.pdf.
You must agree in order to register with the ACME server. Do you agree?
- - - - -
(Y)es/(N)o: y

- - - - -
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
- - - - -
(Y)es/(N)o: n
Account registered.
Please enter the domain name(s) you would like on your certificate (comma and/or
space separated) (Enter 'c' to cancel): j7a604.p.ssafy.io
Requesting a certificate for j7a604.p.ssafy.io

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/j7a604.p.ssafy.io/fullchain.pem
Key is saved at: /etc/letsencrypt/live/j7a604.p.ssafy.io/privkey.pem
This certificate expires on 2022-12-14.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the
background.

Deploying certificate
Successfully deployed certificate for j7a604.p.ssafy.io to /etc/nginx/sites-enabled/default
Congratulations! You have successfully enabled HTTPS on https://j7a604.p.ssafy.io

- - - - -
If you like Certbot, please consider supporting our work by:
* Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
* Donating to EFF: https://eff.org/donate-le
- - - - -

```

nginx proxy 설정

```

# 패키지 업데이트 및 vim 설치
sudo apt-get update
sudo apt-get install vim

# 기본 설정파일 복사
cd /etc/nginx/sites-available/
sudo cp default server1

# vim 편집기 실행
sudo vim server1

# vim 편집기 전체 삭제 후 붙여넣기
vim의 입력모드가 아닌 ESC를 누른 일반 모드에서
dd: 첫번째 줄로 이동
Shift + v + g: 전체 선택
d: 전체 삭제

```

마우스 오른쪽 버튼 클릭(paste)

```
:wq!
```

```
# 기본 설정파일 삭제
```

```
cd /etc/nginx/sites-enabled/
```

```
sudo rm default
```

```
# 서버 설정파일 심볼릭링크 등록
```

```
sudo ln -s /etc/nginx/sites-available/server1 /etc/nginx/sites-enabled/server1
```

```
# nginx 설정 확인
```

```
sudo nginx -t
```

```
# nginx 재시작
```

```
sudo service nginx restart
```

/etc/nginx/sites-available/server1

```
# 80포트로 접근시 443포트로 리다이렉트
```

```
server {
```

```
    if ($host = j7a604.p.ssafy.io) {
```

```
        return 301 https://$host$request_uri;
```

```
    }
```

```
    listen 80 default_server;
```

```
    listen [::]:80 default_server;
```

```
    server_name j7a604.p.ssafy.io;
```

```
    return 404;
```

```
}
```

```
# 443포트로 접근시 ssl 적용
```

```
server {
```

```
    listen [::]:443 ssl ipv6only=on;
```

```
    listen 443 ssl;
```

```
    root /var/www/html;
```

```
# Add index.php to the list if you are using PHP
```

```
index index.html index.htm index.nginx-debian.html;
```

```
server_name j7a604.p.ssafy.io;
```

```
location / {
```

```
    proxy_pass http://localhost:8081;
```

```
    proxy_redirect off;
```

```
    charset utf-8;
```

```
# 넘겨 받을 때 프록시 헤더 정보를 지정
```

```
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;
}

location /molaeng {
    proxy_pass http://localhost:8080/molaeng;
    proxy_redirect off;
    charset utf-8;

    # 넘겨 받을 때 프록시 헤더 정보를 지정
    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;
}

include /etc/letsencrypt/options-ssl-nginx.conf;
ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;
ssl_certificate /etc/letsencrypt/live/j7a604.p.ssafy.io/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/j7a604.p.ssafy.io/privkey.pem;
}

```

jenkins 설치 및 설정

```

# 패키지 업데이트 및 jenkins 설치
sudo docker pull jenkins/jenkins

# jenkins 디렉토리 생성 및 docker-compose.yaml 파일 작성
# jenkins 안에서 프로젝트를 docker 컨테이너로 실행해야 한다.(docker in docker)
# 따라서 docker와 docker compose의 경로를 공유하여 jenkins 내에서 사용할 수 있도록 한다.
cd /home/ubuntu/
mkdir jenkins_home
vim docker-compose.yml

# docker compose 통해 컨테이너 생성 후 실행
sudo docker compose up -d

# 컨테이너 status 확인
sudo docker ps -a

# status가 exit인 경우 로그 확인
sudo docker compose logs

```

docker-compose.yaml

```

version: "3.9"

services:
  jenkins:
    container_name: jenkins-container
    image: jenkins/jenkins

```

```

ports:
  - "9090:8080"
volumes:
  - "/home/ubuntu/jenkins_home:/var/jenkins_home"
  - "/run/docker.sock:/var/run/docker.sock"
  - "/usr/bin/docker:/usr/bin/docker"
  - "/usr/libexec/docker/cli-plugins/docker-compose:/usr/libexec/docker/cli-plugins/docker-compose"
environment:
  TZ: "Asia/Seoul"

```

Jenkins 접속 및 초기 설정

```

# Jenkins 접속
j7a604.p.ssafy.io:9090

# jenkins 초기 관리자 비밀번호 확인
sudo docker compose logs

```

```

jenkins-container | 2022-09-17 11:20:56.406+0000 [id=32] INFO jenkins.install.SetupWizard#init:
jenkins-container | *****
jenkins-container | *****
jenkins-container | *****
jenkins-container | Jenkins initial setup is required. An admin user has been created and a password generated.
jenkins-container | Please use the following password to proceed to installation:
jenkins-container | 525aa3cbe502442c9977cee3ed3fa98c
jenkins-container | This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
jenkins-container | *****
jenkins-container | *****
jenkins-container | *****
jenkins-container | 2022-09-17 11:21:10.422+0000 [id=35] INFO jenkins.InitReactorRunner$1#onAttained:
jenkins-container | 2022-09-17 11:21:10.438+0000 [id=23] INFO hudson.lifecycle.Lifecycle#onReady: Jenk
jenkins-container | 2022-09-17 11:21:11.109+0000 [id=49] INFO h.m.DownloadService$Downloadable#load: 0
jenkins-container | 2022-09-17 11:21:11.110+0000 [id=49] INFO hudson.util.Retrier#start: Performed the
jenkins-container | 2022-09-17 11:21:11.112+0000 [id=49] INFO hudson.model.AsyncPeriodicWork#lambda$do
ubuntu@ip-172-26-12-79:~$

```

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

로그에서 확인한 초기 비밀번호 입력



왼쪽 Install suggested plugins 버튼 클릭

Getting Started

Getting Started

✓ Folders	✓ OWASP Markup Formatter	🔄 Build Timeout	🔄 Credentials Binding
🔄 Timestamper	🔄 Workspace Cleanup	🔄 Ant	🔄 Gradle
🔄 Pipeline	🔄 GitHub Branch Source	🔄 Pipeline: GitHub Groovy Libraries	🔄 Pipeline: Stage View
🔄 Git	🔄 SSH Build Agents	🔄 Matrix Authorization Strategy	🔄 PAM Authentication
🔄 LDAP	🔄 Email Extension	🔄 Mailer	

Folders

- ** JavaBeans Activation Framework (JAF) API
- ** JavaMail API
- ** bouncycastle API
- ** Instance Identity
- ** Mina SSHD API :: Common
- ** Mina SSHD API :: Core
- ** SSH server

OWASP Markup Formatter

- ** Struts
- ** Token Macro

** - required dependency

Jenkins 2.368

플러그인 설치중

Getting Started

계정명:

a604

암호:

.....

암호 확인:

.....

이름:

a604

이메일 주소:

adoramilky@gmail.com

Jenkins 2.368

Skip and continue as admin

Save and Continue

jenkins 계정 생성(id:a604, password:molaengA604!)

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.

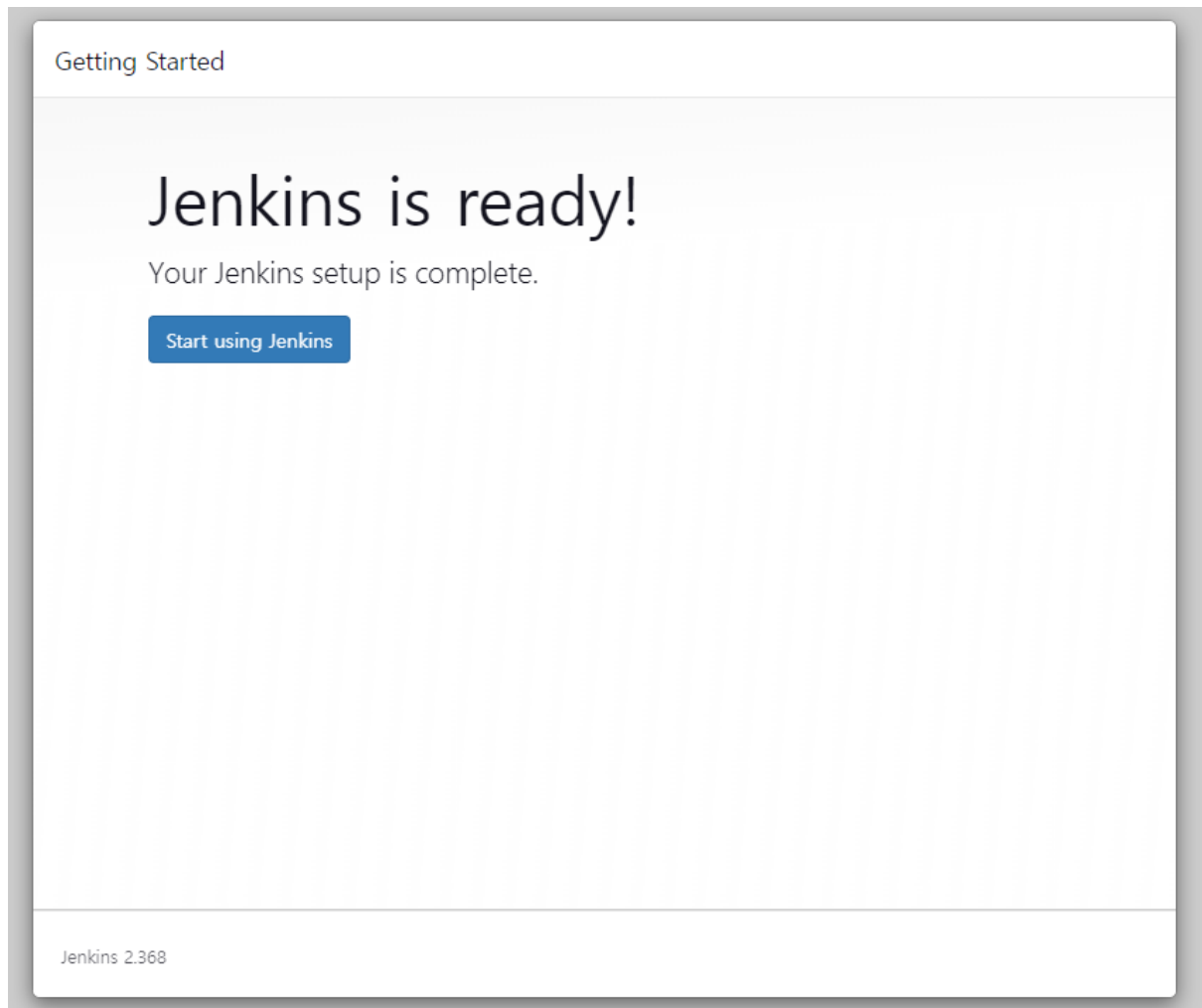
The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.319.2

Not now

Save and Finish

jenkins url 설정(기본값 그대로)



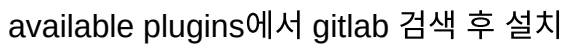
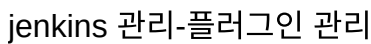
User Defined Time Zone

Time Zone ?

Current time on server in Korean Standard Time: Sep 17, 2022, 10:29:01 PM; in proposed display zone: Sep 17, 2022, 10:29:01 PM

사람-a604-설정-User Defined Time Zone을 Asia/Seoul로 설정 후 저장한다.

jenkins - Gitlab 자동배포



```
# 플러그인 설치 후 jenkins 재시작이 너무 오래 걸리면 오류임
# jenkins 컨테이너 status 확인 후 exit 상태면 수동으로 jenkins 시작해줘야 함
sudo docker ps -a
sudo docker compose up -d
```

+ 새로운 Item
사람
빌드 기록
Jenkins 관리
My Views

빌드 대기 목록
빌드 대기 항목이 없습니다.

빌드 실행 상태
1 대기 중
2 대기 중

Jenkins 관리

Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#).
Set up agent Set up cloud Dismiss

System Configuration

시스템 설정
환경변수 및 경로 정보등을 설정합니다.

Global Tool Configuration
Configure tools, their locations and automatic installers.

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

노드 관리
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

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.

jenkins 관리-manage credentials

+ 새로운 Item
사람
빌드 기록
Jenkins 관리
My Views

빌드 대기 목록
빌드 대기 항목이 없습니다.

빌드 실행 상태
1 대기 중
2 대기 중

Credentials

T	P	Store ↓	Domain	ID	Name
---	---	---------	--------	----	------

Stores scoped to Jenkins

P	Store ↓	Domains
	Jenkins	(global) ↓

아이콘: S M L

global 버튼 클릭

[↑ Back to credential domains](#)

[+ Add Credentials](#)

New credentials

Kind

Username with password

Scope ?

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

Username ?

leenak0

The username. (from [Credentials Plugin](#))

☐ Treat username as secret ?

Password ?

.....

ID ?

molaeng

Description ?

Create

add credentials 클릭

kind : Username with password 선택

Username : Gitlab Username

Password : Gitlab Password



ID : 입력하지 않으면 plugin이 자동 유니크 아이디 생성함(필요에 따라 직접 입력가능 단 유니크 하여야 함)

↑ Back to credential domains

+ Add Credentials

Global credentials (unrestricted)

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


ID	Name	Kind	Description
 molaeng	leenak0/*****	Username with password	


아이콘: S M L


credentials 생성됨


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, 플러그인 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.

OK

새로운 item-item 이름 작성-freestyle project 선택-ok

Freestyle project : 자신이 원하는 형태 및 스크립트로 빌드 진행

Pipeline : Jenkins Script를 직접 작성하거나 Jenkinsfile을 선택하여 빌드 진행

본인이 Jenkinsfile을 작성해두었거나 Pipeline script로 빌드를 진행할 거면 Pipeline을, 그렇지 않으면 Freestyle project를 선택하면 된다.

Configuration

소스 코드 관리

General

소스 코드 관리

빌드 유발

빌드 환경

Build Steps

빌드 후 조치

None

Git

Repositories

Repository URL

https://lab.ssafy.com/s07-bigdata-dist-sub2/S07P22A604.git

Credentials

leenak0/*****

+ Add

고급...

Add Repository

Branches to build

Branch Specifier (blank for 'any')

*/dev

Add Branch

소스 코드 관리-git 선택

Repository URL : Gitlab 프로젝트에서 Clone with HTTPS 복사 후 붙여넣기

Credentials : 위에서 생성한 Credentials 선택

Branch : jenkins build를 돌릴 gitlab의 브랜치를 지정해주면 된다. ex) */master, */dev

Configuration

빌드 유발

⚙️ General

🔑 소스 코드 관리

🕒 빌드 유발

🌐 빌드 환경

☰ Build Steps

📦 빌드 후 조치

☐ 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://f7a604.p.ssafy.io:9999/project/molaeng(dev) ?

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

Build when a change is pushed to Gitlab 선택

trigger : push events, accepted merge request events

- 🕒 빌드 유발
- 🌐 빌드 환경
- ☰ Build Steps
- 📦 빌드 후 조치

Allowed branches

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

Secret token ?

3936f6f66692be979b783bf9a049cc7c

Generate

Clear

- ☐ GitHub hook trigger for GITScm polling ?
- ☐ Poll SCM ?

빌드 환경

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s) ?
- ☐ Add timestamps to the Console Output

저장

Apply

고급 버튼 클릭-secret token generate-저장

Search page

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

 http://j7a604.p.ssafy.io:9090/project/molaeng

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

Secret token

 3936f6f66692be979b783bf9a049cc7c

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

Trigger

☒ Push events

 dev

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.

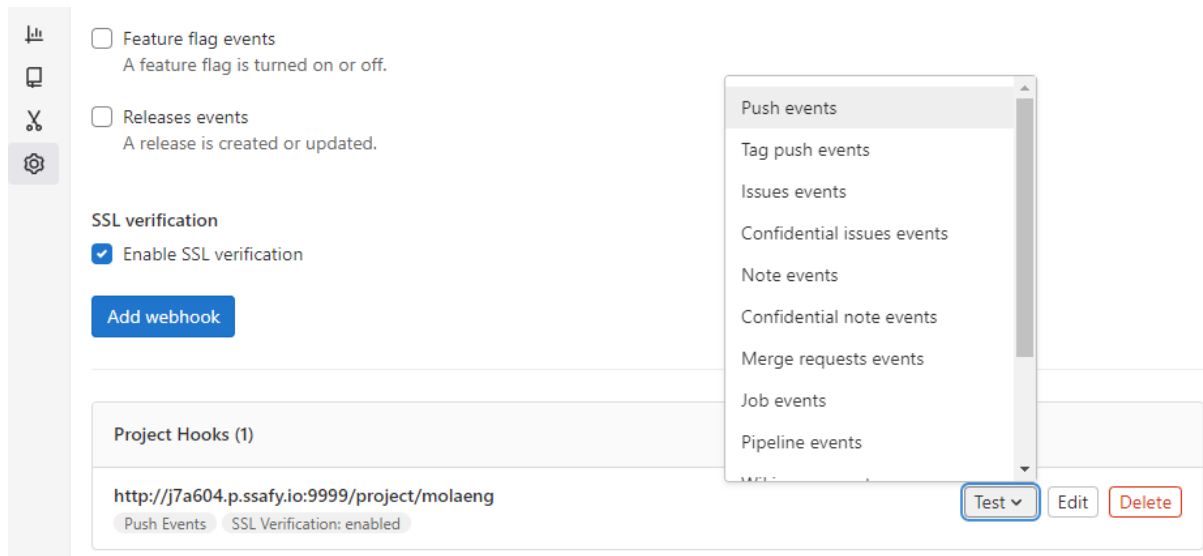
Gitlab 프로젝트-Settings-Webhooks

URL : Jenkins의 URL로, jenkins 설정 중 빌드 유발 부분에 나오는 jenkins url이 들어가면 된다.

Secret token : 빌드 유발 부분에서 Generate로 생성한 Secret token이 들어가면 된다.

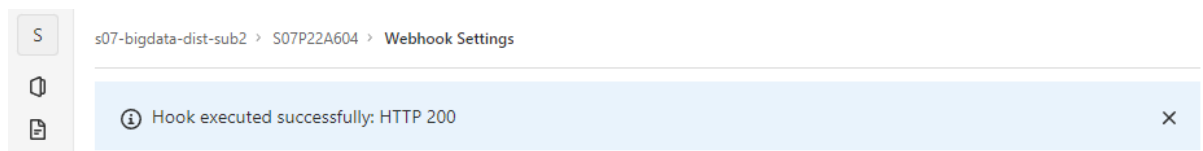
Trigger : Trigger를 발생시킬 시점과 이벤트를 발생시킬 branch를 지정한다.

Add webhook 버튼을 클릭하면 밑에 Project Hooks가 추가된다.



생성한 Project Hooks에서 test를 누르면 다음과 같은 목록이 나오게 된다.

Push events를 누르면 test trigger가 발생하게 된다.



트리거 발생에 성공하면 상단에 다음과 같은 알림이 뜨게 된다.

S	W	Name ↓	최근 성공	최근 실패	최근 소요 시간
✓	⚙️	molaeng	2 min 14 sec #1	—	0.84 sec ▶

jenkins에서 빌드 상태 확인

검색 (CTRL+K)

Dashboard > molaeng >

☰ 상태

</> 변경사항

📁 작업공간

🗑️ 작업공간 초기화

▶ 지금 빌드

⚙️ 구성

🗑️ Project 삭제

✎ Rename

Workspace of molaeng on Built-In Node

molaeng / →

📁 .git

📁 molaeng_back

📁 molaeng_front

📄 .gitignore 2022. 9. 17. 오후 10:22:44 6.77 KB 🌐 👁

📄 README.md 2022. 9. 17. 오후 10:22:44 0 B 🌐 👁

📄 (zip 파일로 압축)

🔍 Build History

추이 ▼

Filter builds...

🟢 #1

2022. 9. 17. 오후 10:22 KST

Started by GitLab push by 이나경

📡 Atom feed (전체)
📡 Atom feed (실패)

↕

↑

↓

프로젝트가 저장된 작업공간 : jenkins 디렉토리 밑 workspace 폴더에 item명으로 프로젝트가 생성됨

/home/ubuntu/jenkins_home/workspace/molaeng/

배포가이드

26

Build Steps

≡ Execute shell ?

Command

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

```
docker compose up -d
```

고급...

Add build step ▾

Build Steps-Execute shell 선택 후 명령어 입력

docker compose를 사용하여 프로젝트를 빌드하고 컨테이너를 생성 후 실행한다.

molaeng_back/Dockerfile

```
FROM openjdk:8-jdk-alpine AS builder

WORKDIR /app

COPY gradlew .
COPY gradle gradle
COPY build.gradle .
COPY settings.gradle .
COPY src src

RUN chmod +x ./gradlew
RUN ./gradlew clean build

FROM openjdk:8-jdk-alpine

COPY --from=builder /app/build/libs/*.jar app.jar

EXPOSE 8080

ENTRYPOINT ["java", "-jar", "app.jar"]
```

molaeng_front/Dockerfile

```
FROM node:lts-alpine as builder

WORKDIR /app

COPY package*.json ./
RUN npm install

COPY . .
RUN npm run build

FROM nginx:stable-alpine

COPY --from=builder /app/default.conf /etc/nginx/conf.d/default.conf
COPY --from=builder /app/dist /usr/share/nginx/html

EXPOSE 8081

CMD ["nginx", "-g", "daemon off;"]
```

molaeng_front/default.conf

```
server {
    listen 80;
    server_name localhost;

    location / {
        root    /usr/share/nginx/html;
        index   index.html index.htm;
        try_files $uri $uri/ /index.html;
    }
}
```

docker-compose.yml

```
version: "3.9"

services:
  molaeng_back:
    build:
      context: ./molaeng_back
    container_name: molaeng_back
    restart: always
    depends_on:
      - mysql
      - redis
    ports:
      - "8080:8080"
    environment:
```

```

    SPRING_DATASOURCE_URL: "jdbc:mysql://mysql-container:3306/molaeng"
    SPRING_DATASOURCE_USERNAME: "ssafy"
    SPRING_DATASOURCE_PASSWORD: "ssafyA604!"
    TZ: "Asia/Seoul"

molaeng_front:
  build:
    context: ./molaeng_front
  container_name: molaeng_front
  ports:
    - "8081:80"
  environment:
    TZ: "Asia/Seoul"

mysql:
  image: mysql
  container_name: mysql-container
  restart: always
  ports:
    - "3306:3306"
  volumes:
    - "/home/ubuntu/mysql:/var/lib/mysql"
  environment:
    MYSQL_ROOT_PASSWORD: "molaengA604!"
    MYSQL_DATABASE: "molaeng"
    MYSQL_USER: "ssafy"
    MYSQL_PASSWORD: "ssafyA604!"
    TZ: "Asia/Seoul"
  command:
    - "--character-set-server=utf8mb4"
    - "--collation-server=utf8mb4_general_ci"

redis:
  image: redis
  container_name: redis-container
  restart: always
  ports:
    - "6379:6379"
  volumes:
    - "/home/ubuntu/redis:/data"
  environment:
    TZ: "Asia/Seoul"

```

Build Steps

≡ Execute shell ?

Command

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

```
docker stop molaeng_front
docker stop molaeng_back
docker rm molaeng_front
docker rm molaeng_back
docker rmi molaeng-molaeng_front
docker rmi molaeng-molaeng_back
docker compose up -d
```

고급...

Add build step ▾

```
docker stop molaeng_front
docker stop molaeng_back
docker rm molaeng_front
docker rm molaeng_back
docker rmi molaeng-molaeng_front
docker rmi molaeng-molaeng_back
docker compose up -d
```

두번째 배포부터는 생성되어있는 이미지를 삭제 후 다시 생성해주어야 한다.

1.실행중인 컨테이너 중지 2.컨테이너 삭제 3.프로젝트 이미지 삭제 후 docker 실행