우리의 추억 포레스트

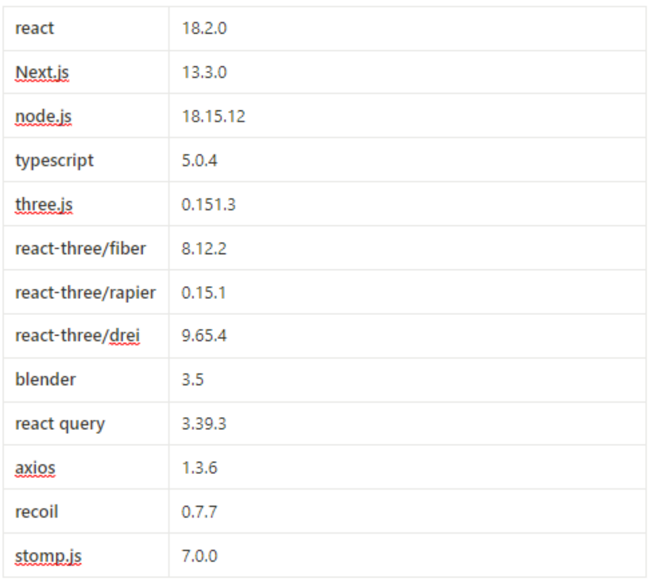
츄레스트

포팅매뉴얼

목차

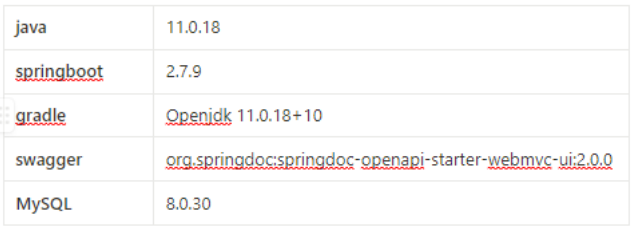
1. 프로젝트 개발 환경

1) FRONT END



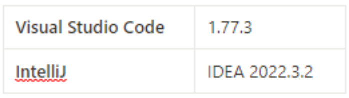
react	18.2.0
Next.js	13.3.0
node.js	18.15.12
typescript	5.0.4
three.js	0.151.3
react-three/fiber	8.12.2
react-three/rapier	0.15.1
react-three/drei	9.65.4
blender	3.5
react query	3.39.3
axios	1.3.6
recoil	0.7.7
stomp.js	7.0.0

2) BACK END



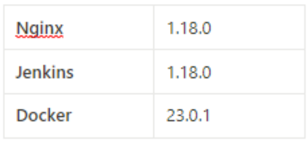
java	11.0.18
springboot	2.7.9
gradle	Openjdk 11.0.18+10
swagger	org.springdoc:springdoc-openapi-starter-webmvc-ui:2.0.0
MySQL	8.0.30

3) IDE



Visual Studio Code	1.77.3
IntelliJ	IDEA 2022.3.2

4) Server



Nginx	1.18.0
Jenkins	1.18.0
Docker	23.0.1

2. 설정 파일 목록

1) Front End

- **.env**
    - 카카오톡 REDIRECT URI 설정

```
NEXT_PUBLIC_IMAGE_ROOT="https://storage.cloud.google.com/churest-bucket"
NEXT_PUBLIC_API_KAKAO_KEY = bc4a08b635ae352a453b10a7dc3d78ca
# local 용
NEXT_PUBLIC_API_REDIRECT_URL = http://localhost:3000/redirect
# 서버용
NEXT_PUBLIC_API_REDIRECT_URL = https://k8a505.p.ssafy.io/redirect
```

- **package.json**

```
"scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start -p 3000",
    "lint": "next lint"
  },
```

- **배포 설정 파일**
    - Docker File

```
FROM node:18.15.0-alpine

WORKDIR /var/jenkins_home/workspace/Development/Development/FE/churest
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build

EXPOSE 3000

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

- 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 {
        ##
        # Basic Settings
        ##

        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;
        keepalive_timeout 65;
        types_hash_max_size 2048;
        client_max_body_size 2048;
        # server_tokens off;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

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

        ##
        # SSL Settings
        ##

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

        ##
        # Logging Settings
        ##

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

        ##
        # Gzip Settings
        ##

        gzip on;

        # gzip_vary on;
        # gzip_proxied any;
        # gzip_comp_level 6;
        # gzip_buffers 16 8k;
        # gzip_http_version 1.1;
        # gzip_types text/plain text/css application/json application/javascript text/xml application/xml application/xml+rss text

        ##
        # Virtual Host Configs
        ##

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

2) Back End

- **Spring Boot 설정 파일**
  - WebConfig

```
@Override
    public void addCorsMappings(CorsRegistry registry) {
        registry.addMapping("/**")
                .allowedHeaders("Origin", "Content-Type", "Accept", "X-AUTH-TOKEN")
                .allowedOrigins("*","http://localhost:8080", "http://localhost:3000"
                        , "https://k8a505.p.ssafy.io", "https://k8a505.p.ssafy.io:80", "https://k8a505.p.ssafy.io:8080", "https://
                .allowedMethods("OPTIONS", "GET", "POST", "PUT", "DELETE", "PATCH");
    }
}
```

- **application.properties**

```
#port
server.port=8080

#base url
# server.servlet.contextPath=/api
# Charset of HTTP requests and responses. Added to the "Content-Type" header if not set explicitly.
server.servlet.encoding.charset=UTF-8
# Enable http encoding support.
server.servlet.encoding.enabled=true
# Force the encoding to the configured charset on HTTP requests and responses.
server.servlet.encoding.force=true

# MySQL Driver
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# DB URL serverTimezone=Asia/Seoul
spring.datasource.url=jdbc:mysql://k8a505.p.ssafy.io:3306/churest?serverTimezone=UTC&useUniCode=yes&characterEncoding=UTF-8&allow

# DB username
spring.datasource.username=churest

# DB password
spring.datasource.password=churest7581

spring.mvc.pathmatch.matching-strategy=ant_path_matcher

spring.jpa.database=mysql
```

- **배포 설정 파일**
    - Docker File

```
FROM adoptopenjdk/openjdk11 AS builder
COPY gradlew .
COPY gradle gradle
COPY build.gradle .
COPY settings.gradle .
COPY src src
RUN chmod +x ./gradlew
RUN ./gradlew bootJAR

FROM adoptopenjdk/openjdk11
COPY --from=builder build/libs/*.jar app.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "app.jar"]
```

    - build.gradle

```
plugins {
  id 'java'
  id 'org.springframework.boot' version '2.7.9'
  id 'io.spring.dependency-management' version '1.1.0'
}

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

configurations {
  compileOnly {
    extendsFrom annotationProcessor
  }
}

repositories {
  mavenCentral()
}

dependencies {
//  implementation 'org.springframework.boot:spring-boot-starter-data-jdbc'
  implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
  implementation 'org.springframework.boot:spring-boot-starter-security'
  implementation 'org.springframework.boot:spring-boot-starter-webflux'
  implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'
  implementation 'org.springframework.boot:spring-boot-starter-web'
  compileOnly 'org.projectlombok:lombok'
  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'

  //  swagger
  implementation 'io.springfox:springfox-swagger2:2.9.2'
  implementation 'io.springfox:springfox-swagger-ui:2.9.2'

  // google cloud
  implementation group: 'com.google.cloud', name: 'spring-cloud-gcp-starter', version: '3.4.3'
  implementation group: 'com.google.cloud', name: 'spring-cloud-gcp-storage', version: '3.4.3'

  // validation 체크
  implementation 'org.springframework.boot:spring-boot-starter-validation'

  // jwt
  implementation 'io.jsonwebtoken:jjwt:0.9.1'

  //websocket
  implementation 'org.springframework.boot:spring-boot-starter-websocket'
  implementation 'org.webjars:sockjs-client:1.1.2'
  implementation 'org.webjars:stomp-websocket:2.3.3-1'
//
////   new
//  implementation "com.h2database:h2"
//
//  implementation "io.jsonwebtoken:jjwt-api:0.11.2"
//  implementation "io.jsonwebtoken:jjwt-impl:0.11.2"
//  implementation "io.jsonwebtoken:jjwt-jackson:0.11.2"

  implementation 'com.google.firebase:firebase-admin:9.1.1'

  // Multipart file
  implementation 'commons-io:commons-io:2.11.0'    /* Apache commons-io */
  implementation group: 'commons-fileupload', name: 'commons-fileupload', version: '1.4' /* Apache Commons FileUpload */

}

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

bootJar{
  bootJar.enabled=true
}

jar {
    enabled = false
}
```

3. EC2 설정 시나리오

0) 사용 포트

구분	포트 번호
Front-end	3000
Back-end	8080
Sub-Back-end	9090
MySQL	3306

1) EC2 인증키를 이용한 접근

cmd 또는 windows powershell 을 이용해 다운받은 Pem 파일이 있는 폴더에서 명령어를 입력

```
$ ssh -i K8A505T.pem ubuntu@k8a505.p.ssafy.io
```

2) 방화벽 설정

현재 방화벽 status 확인 및 설정

```
$ sudo ufw status
$ sudo ufw allow 22
$ sudo ufw enable
$ sudo reboot
```

3) Docker 설치

Ubuntu에 도커 설치

```
$ sudo apt update

# 필수 패키지 설치
$ sudo apt-get install -y ca-certificates \
    curl \
    software-properties-common \
    apt-transport-https \
    gnupg \
    lsb-release

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

# Docker 설치 (docker-compose 도 추가로 설치해주기)
$ sudo apt update
$ sudo apt install docker-ce docker-ce-cli containerd.io docker-compose

# 도커 확인
$ sudo service docker status
```

4) Jenkins 설치 및 설정

1. docker-compose 를 이용해 젠킨스 컨테이너 생성

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

docker-compose.yml 파일

- Esc 이후 :wq 를 입력하여 파일 저장

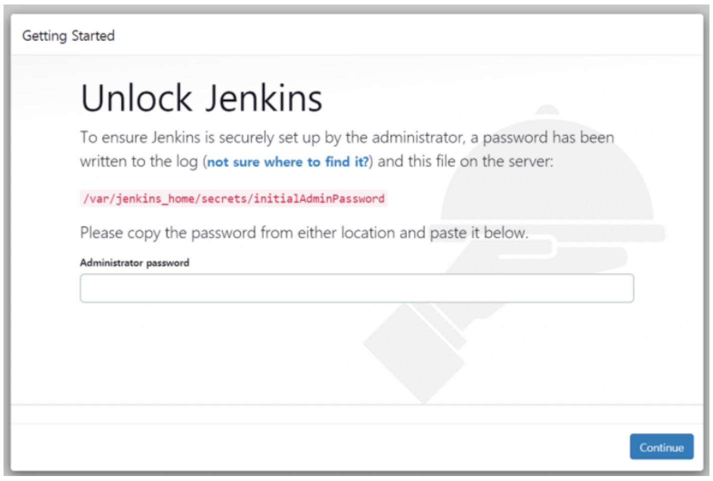
```
version: '3'

services:
    jenkins:
        image: jenkins/jenkins:lts
        container_name: jenkins
        volumes:
      - /usr/bin/docker:/usr/bin/docker
            - /var/run/docker.sock:/var/run/docker.sock
            - /jenkins:/var/jenkins_home
        ports:
            - "9090:8080"
        privileged: true
        user: root
```

컨테이너 생성

```
$ sudo docker-compose up -d
컨테이너 확인
$ sudo docker ps
```

2. jenkins 접속 (서버 주소)

서버공인 IP:9090 으로 접속하면 젠킨스 시작 화면이 나오게 됨


Getting Started

Unlock Jenkins

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

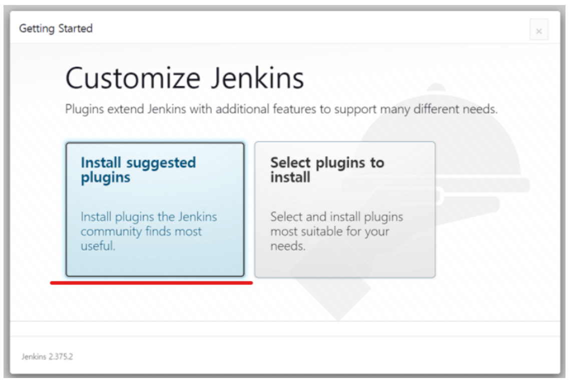
/var/jenkins_home/secrets/initialAdminPassword

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

Administrator password

```
# 비밀번호 확인 방법
$ sudo docker logs jenkins
```

3. 기본 설치


Getting Started

Customize Jenkins

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

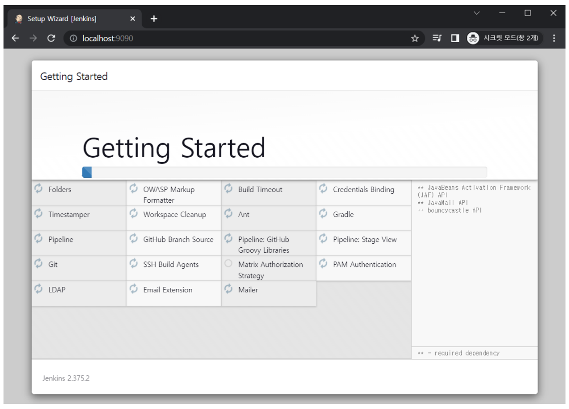
Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.375.2



Getting Started

Getting Started

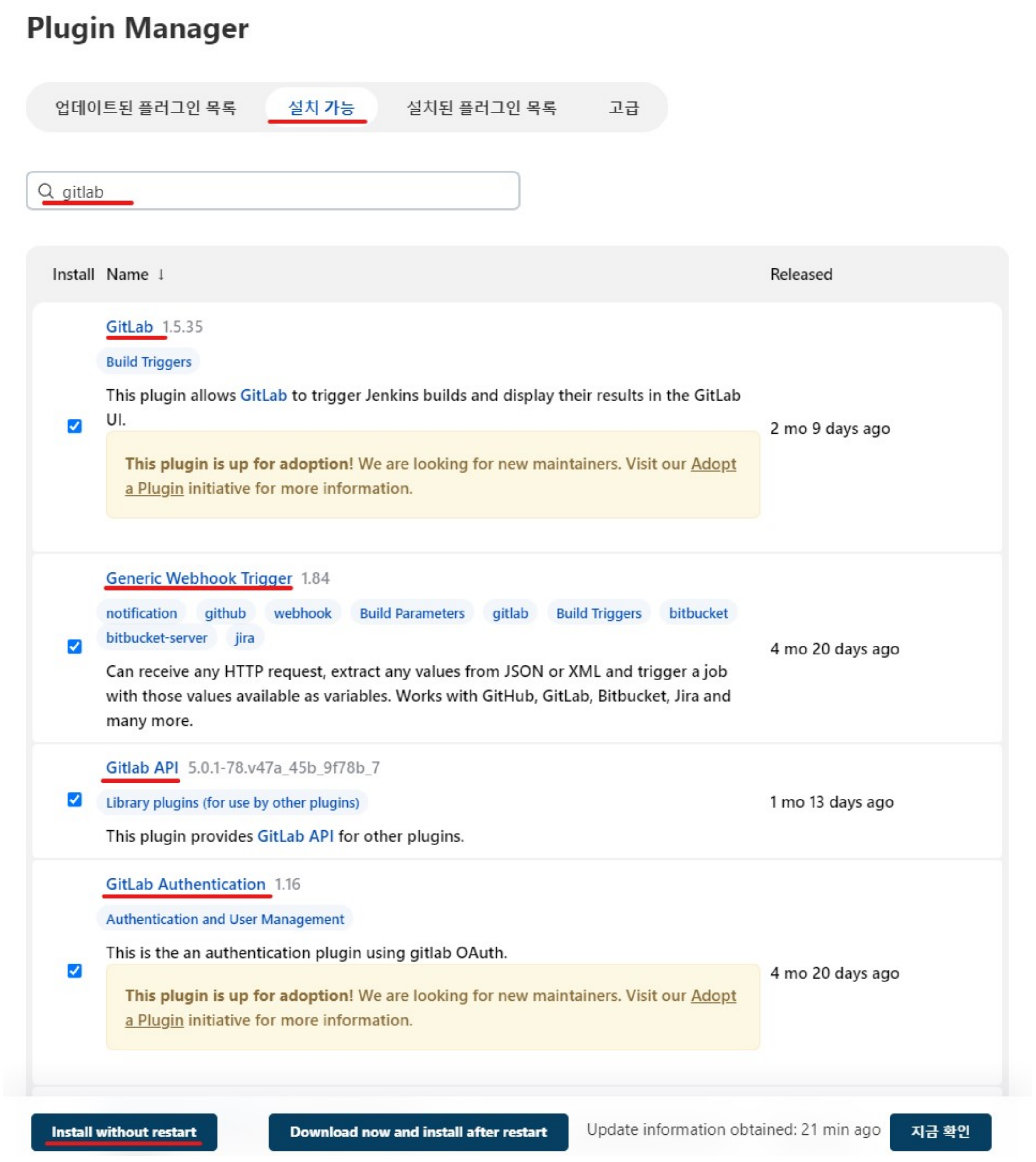
Folders	OWASP Markup Formatter	Build Timeout	Credentials Binding
Timestamper	Workspace Cleanup	Ant	Gradle
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View
Git	SSH Build Agents	Matrix Authorization Strategy	PAM Authentication
LDAP	Email Extension	Mailer	

```
** JavaBeans Activation Framework
(JAF) API
** JavaMail API
** bouncycastle API

** - required dependency
```

Jenkins 2.375.2

4. jenkins 플러그인 설치

- **Dashboard → Jenkins 관리 → 플러그인 관리 → Available plugins**
- **Gitlab 관련 항목 설치**
  - Gitlab, Generic Webhook Trigger, Gitlab API, Gitlab Authentication 설치


Plugin Manager

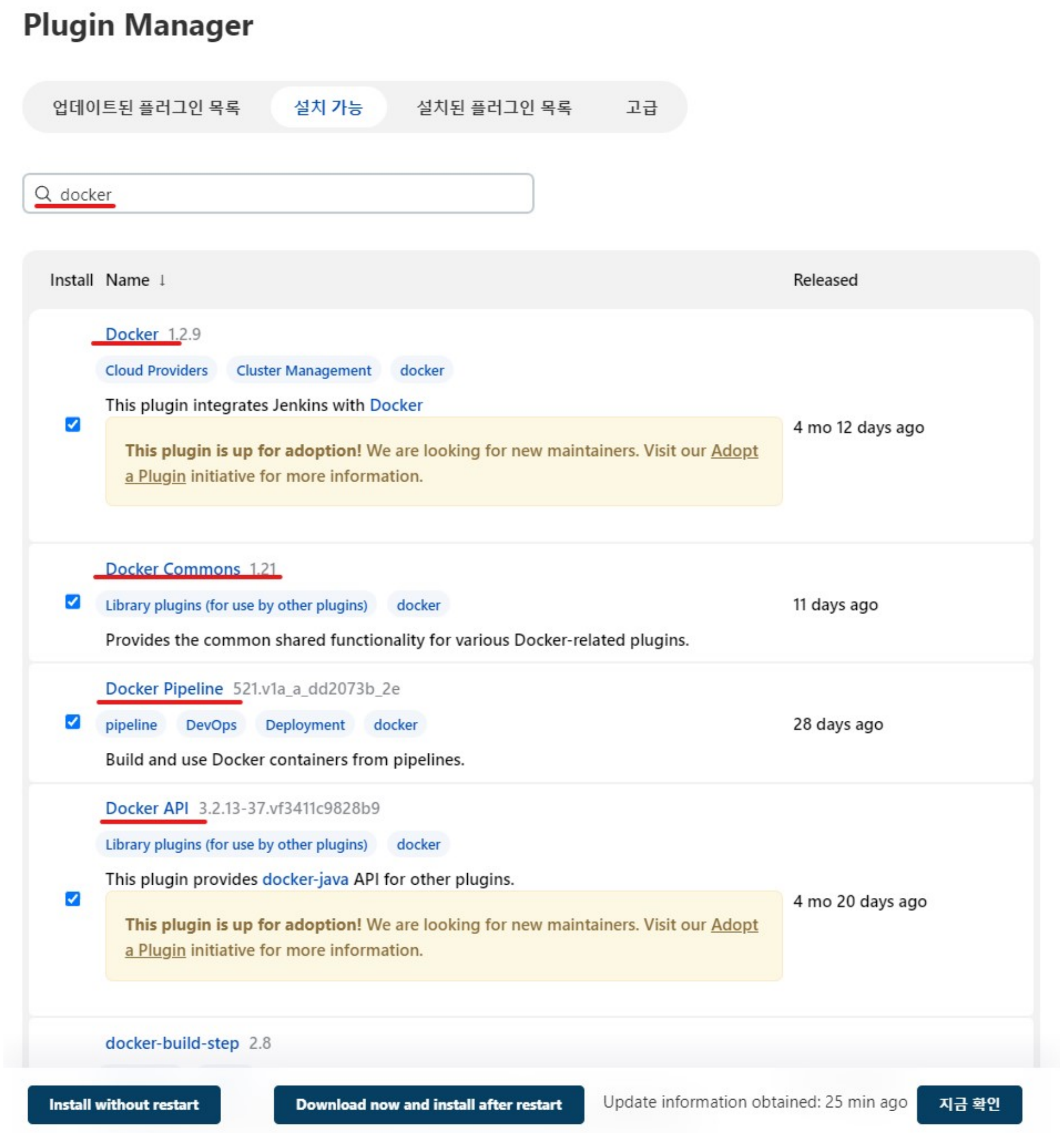
업데이트된 플러그인 목록 | 설치 가능 | 설치된 플러그인 목록 | 고급

gitlab

Install	Name ↓	Released
✓	GitLab 1.5.35 Build Triggers This plugin allows GitLab to trigger Jenkins builds and display their results in the GitLab UI. **This plugin is up for adoption!** We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	2 mo 9 days ago
✓	Generic Webhook Trigger 1.84 notification github webhook Build Parameters gitlab Build Triggers bitbucket bitbucket-server jira 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.	4 mo 20 days ago
✓	Gitlab API 5.0.1-78.v47a_45b_9f78b_7 Library plugins (for use by other plugins) This plugin provides GitLab API for other plugins.	1 mo 13 days ago
✓	GitLab Authentication 1.16 Authentication and User Management This is the an authentication plugin using gitlab OAuth. **This plugin is up for adoption!** We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	4 mo 20 days ago

Install without restart | Download now and install after restart | Update information obtained: 21 min ago | 지금 확인

- **Docker 관련 항목 설치**
  - Docker, Docker Commons, Docker Pipeline, Docker API 설치



Plugin Manager

Docker 1.2.9
Cloud Providers | Cluster Management | docker
This plugin integrates Jenkins with Docker
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
4 mo 12 days ago

Docker Commons 1.21
Library plugins (for use by other plugins) | docker
Provides the common shared functionality for various Docker-related plugins.
11 days ago

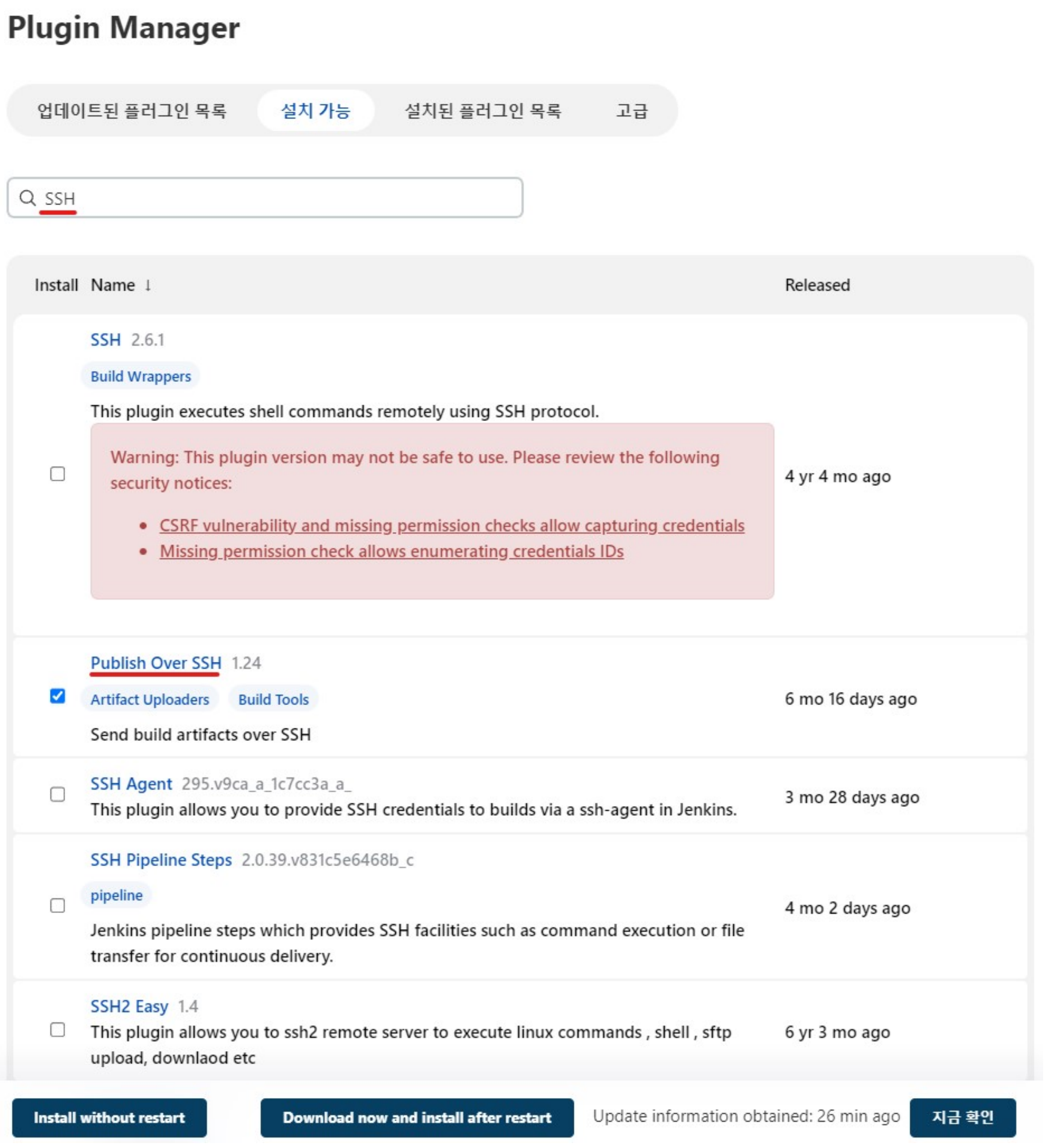
Docker Pipeline 521.v1a_a_dd2073b_2e
pipeline | DevOps | Deployment | docker
Build and use Docker containers from pipelines.
28 days ago

Docker API 3.2.13-37.vf3411c9828b9
Library plugins (for use by other plugins) | docker
This plugin provides docker-java API for other plugins.
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
4 mo 20 days ago

docker-build-step 2.8

Install without restart
Download now and install after restart
Update information obtained: 25 min ago
지금 확인

- **SSH 연결 관련 항목 설치**
  - Publish Over SSH 설치


Plugin Manager

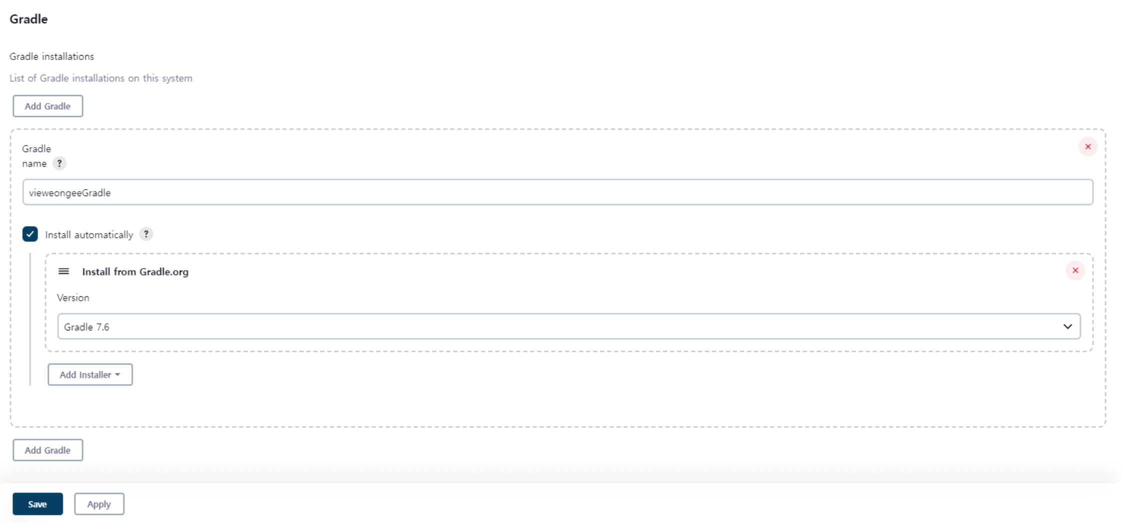
업데이트된 플러그인 목록 | 설치 가능 | 설치된 플러그인 목록 | 고급

SSH

Install	Name ↓	Released
☐	SSH 2.6.1 Build Wrappers This plugin executes shell commands remotely using SSH protocol. Warning: This plugin version may not be safe to use. Please review the following security notices: • CSRF vulnerability and missing permission checks allow capturing credentials • Missing permission check allows enumerating credentials IDs	4 yr 4 mo ago
☑	Publish Over SSH 1.24 Artifact Uploaders Build Tools Send build artifacts over SSH	6 mo 16 days ago
☐	SSH Agent 295.v9ca_a_1c7cc3a_a_ This plugin allows you to provide SSH credentials to builds via a ssh-agent in Jenkins.	3 mo 28 days ago
☐	SSH Pipeline Steps 2.0.39.v831c5e6468b_c pipeline Jenkins pipeline steps which provides SSH facilities such as command execution or file transfer for continuous delivery.	4 mo 2 days ago
☐	SSH2 Easy 1.4 This plugin allows you to ssh2 remote server to execute linux commands , shell , sftp upload, downlaod etc	6 yr 3 mo ago

Install without restart
Download now and install after restart
Update information obtained: 26 min ago
지금 확인

5. Gradle 사용하는 경우
    - Jenkins 관리 → Global Tool Configuration → Gradle 버전 설정 후 추가


Gradle

Gradle installations

List of Gradle installations on this system

Gradle name: vieweongeeGradle

Install automatically ✓

Install from Gradle.org

Version: Gradle 7.6

5) Nginx 설치

```
# 설치
$ sudo apt-get install nginx

# 설치 확인 및 버전 확인
$ nginx -v

# Nginx 설정은 SSL 인증서 발급 후에
```

6) SSL 인증서 발급

```
# certbot 설치
$ sudo apt-get install python3-certbot-nginx
$ sudo certbot certonly --nginx -d k8a505.p.ssafy.io
# 인증서 발급
$ sudo certbot certonly --nginx -d k8a505.p.ssafy.io
# 인증서 내역 확인
$ sudo certbot certificates

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Found the following certs:
  Certificate Name: k8a505.p.ssafy.io
    Domains: k8a505.p.ssafy.io
    Expiry Date: 2023-07-27 03:01:38+00:00 (VALID: 89 days)
    Certificate Path: /etc/letsencrypt/live/k8a505.p.ssafy.io/fullchain.pem
    Private Key Path: /etc/letsencrypt/live/k8a505.p.ssafy.io/privkey.pem
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

7) Nginx 설정

```
$ sudo vi /etc/nginx/sites-available/churest.conf
$ sudo vi /etc/nginx/conf.d/default.conf
```

```
# 잘 돌아가는 지 테스트
$ sudo nginx -t

# nginx 실행
$ sudo systemctl start nginx

# 실행 확인
$ sudo systemctl status nginx
```

```
server {

         location / {
                proxy_hide_header Access-Control-Allow-Origin;
                add_header 'Access-Control-Allow-Origin' '*';
                add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';
                add_header 'Access-Control-Allow-Headers' 'DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Typ
                add_header 'Access-Control-Expose-Headers' 'Content-Length,Content-Range';
                proxy_connect_timeout 90;
                proxy_send_timeout 90;
                proxy_read_timeout 90;

                proxy_pass http://localhost:3000;

                proxy_buffer_size 128k;
                proxy_buffers   4 256k;
                proxy_busy_buffers_size 256k;
                proxy_http_version 1.1;
    #         proxy_set_header Upgrade $http_upgrade;
     #          proxy_set_header Connection "upgrade";
      #         proxy_set_header Host $host;
       #        proxy_set_header Origin "";


#proxy_set_header    X-Real-IP           $remote_addr;
#    proxy_set_header    X-Forwarded-For     $proxy_add_x_forwarded_for;
location /api {
                proxy_hide_header Access-Control-Allow-Origin;
                add_header 'Access-Control-Allow-Origin' '*';
                add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';
                add_header 'Access-Control-Allow-Headers' 'DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Typ
                add_header 'Access-Control-Expose-Headers' 'Content-Length,Content-Range';
                proxy_connect_timeout 90;
                proxy_send_timeout 90;
                proxy_read_timeout 90;
                proxy_pass http://localhost:8080/api;

#               proxy_http_version 1.1;
 #              proxy_set_header Upgrade $http_upgrade;
  #             proxy_set_header Connection "upgrade";
   #            proxy_set_header Host $host;
    #            proxy_set_header Origin "";


     }

        location /chat {
#               proxy_hide_header Access-Control-Allow-Origin;
   #             add_header 'Access-Control-Allow-Origin' '*';
  #              add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';
 #               add_header 'Access-Control-Allow-Headers' 'DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Ty
#                add_header 'Access-Control-Expose-Headers' 'Content-Length,Content-Range';
 #               proxy_connect_timeout 90;
 #               proxy_send_timeout 90;
#                proxy_read_timeout 90;
                proxy_pass http://localhost:8080/chat; # WebSocket Server

               proxy_http_version 1.1;
               proxy_set_header Upgrade $http_upgrade;
               proxy_set_header Connection "upgrade";
               proxy_set_header Host $host;
                proxy_set_header Origin "";

#proxy_set_header    X-Real-IP           $remote_addr;
 #   proxy_set_header    X-Forwarded-For     $proxy_add_x_forwarded_for;

        }

        listen 443 ssl;

        ssl_certificate /etc/letsencrypt/live/k8a505.p.ssafy.io/fullchain.pem;
        ssl_certificate_key /etc/letsencrypt/live/k8a505.p.ssafy.io/privkey.pem;
 }
 server {
        server_name k8a505.p.ssafy.io;
        listen 80;
        return 301 https://$server_name$request_uri;
 }
```

```
$ sudo vi /etc/nginx/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 {
        ##
        # Basic Settings
        ##

        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;
        keepalive_timeout 65;
        types_hash_max_size 2048;
        client_max_body_size 2048;
        # server_tokens off;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

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

        ##
        # SSL Settings
        ##

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

        ##
        # Logging Settings
        ##

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

        ##
        # Gzip Settings
        ##

        gzip on;

        # gzip_vary on;
        # gzip_proxied any;
        # gzip_comp_level 6;
        # gzip_buffers 16 8k;
        # gzip_http_version 1.1;
        # gzip_types text/plain text/css application/json application/javascript text/xml application/xml application/xml+rss text/jav

        ##
        # Virtual Host Configs
        ##

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

8) MySQL 설치

(1) Ubuntu에 MySQL 설치

```
# MySQL을 설치
sudo apt-get update
sudo apt-get install mysql-server

# MySQL 구동
sudo systemctl start mysql.service

# MySQL 접속s
$ sudo mysql
```

```
# 계정 생성
mysql> CREATE USER '계정이름'@'%' IDENTIFIED BY '비밀번호';

# GRANT로 권한 부여 - 어떠한 ip에서든 해당 계정에 모든 권한 부여
mysql> GRANT ALL PRIVILEGES ON . TO '계정이름'@'%' WITH GRANT OPTION;
mysql> FLUSH PRIVILEGES;

# 현재 mysql에서 기본으로 세팅 되어있는 유저들과 추가된 유저를 확인
mysql > SELECT user,authentication_string,plugin,host FROM mysql.user;

# database 생성- utf8 확장 버전
mysql > CREATE DATABASE '데이터베이스명' CHARACTER SET utf8mb4 collate utf8mb4_general_ci;

# 해당 계정이 database의 모든 테이블에 모든 권한 행사
mysql > GRANT ALL PRIVILEGES ON '데이터베이스명'.* TO '계정이름'@'%';
```

(2) MySQL Workbench 사용법

1. MySQL Workbench 설치
2. Connection 설정
    - MySQL Connection → '+' 버튼
    - Server에 있는 MySQL과 연결
        - Connection Name: 원하는 이름
        - Hostname: 접속할 서버 주소
        - Username: 생성한 MySQL 계정의 username

3. 배포

1) Jenkins & GitLab 연동

(1) jenkins에서 프로젝트 생성

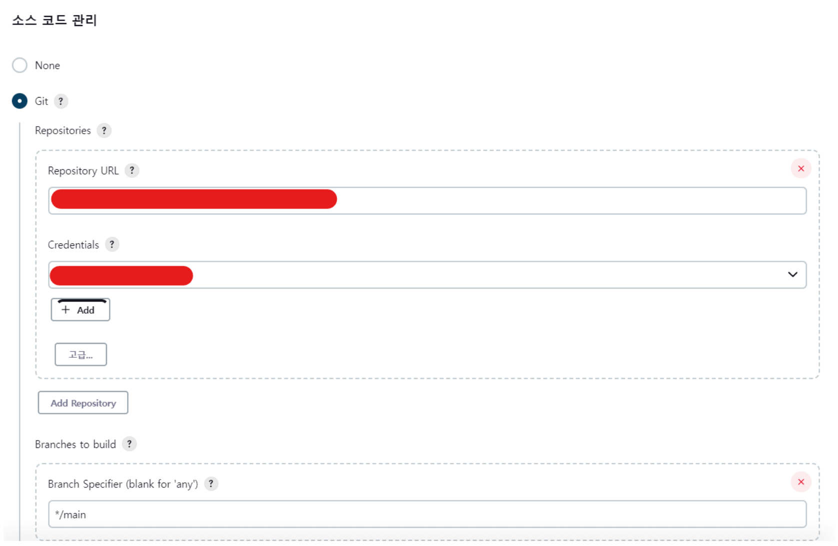
Dashboard → 새로운 Item → 프로젝트 이름 입력 → Freestyle project

(2) 소스 코드 관리

소스 코드 관리 > Git 선택 > git clone 주소 입력

→ Credentials 아래의 Add > Jenkins 선택 > Username: 싸피깃 아이디 / Password: 싸피깃 비밀번호 / ID: Credential 구별할 아무 텍스트 입력하기 > Add 버튼

→ 저장한 credential을 클릭했을 때 에러메세지가 뜨지 않으면 정상 접근 연동 성공


소스 코드 관리

None

● Git ?

Repositories ?

Repository URL ?

[redacted]

Credentials ?

[redacted]

+ Add

고급...

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

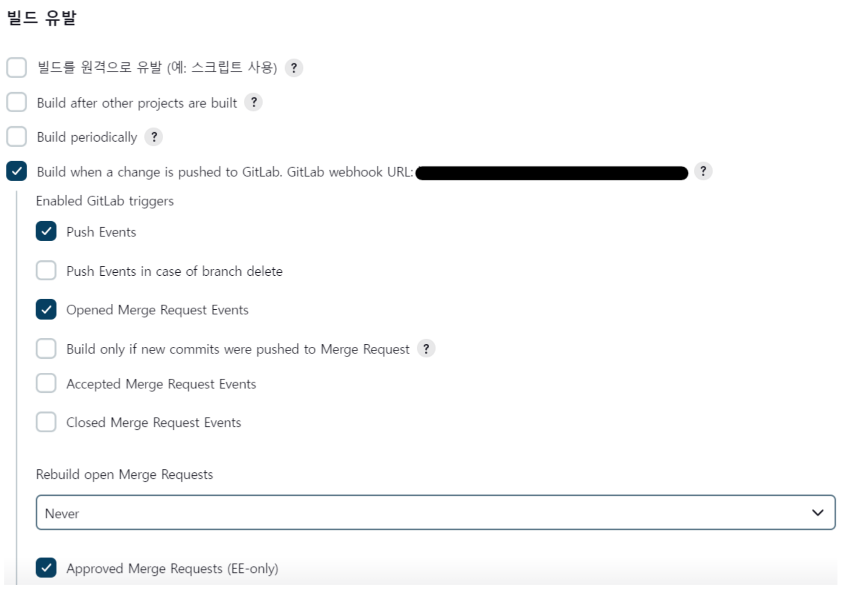
*/main

(3) 빌드 유발

- `Build when a change is pushed to GitLab` 체크

(4) Webhook 설정

1. Jenkins project 선택 > 구성 > 빌드 유발 > Secret token Generate(따로 기록해두기) > 저장


빌드 유발

- [ ] 빌드를 원격으로 유발 (예: 스크립트 사용) ?
- [ ] Build after other projects are built ?
- [ ] Build periodically ?
- [✓] Build when a change is pushed to GitLab. GitLab webhook URL: [redacted] ?

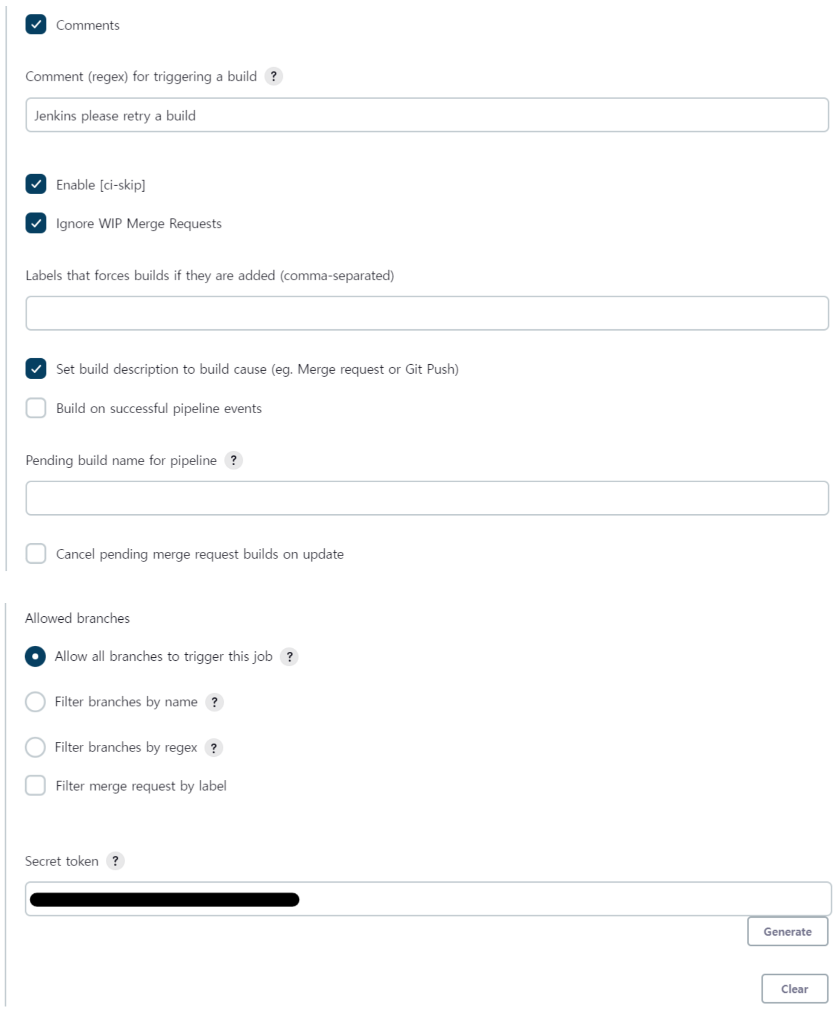
Enabled GitLab triggers

- [✓] 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

[✓] Enable [ci-skip]

[✓] Ignore WIP Merge Requests

Labels that forces builds if they are added (comma-separated)

[✓] Set build description to build cause (eg. Merge request or Git Push)

[ ] Build on successful pipeline events

Pending build name for pipeline ?

[ ] Cancel pending merge request builds on update

Allowed branches

● Allow all branches to trigger this job ?

○ Filter branches by name ?

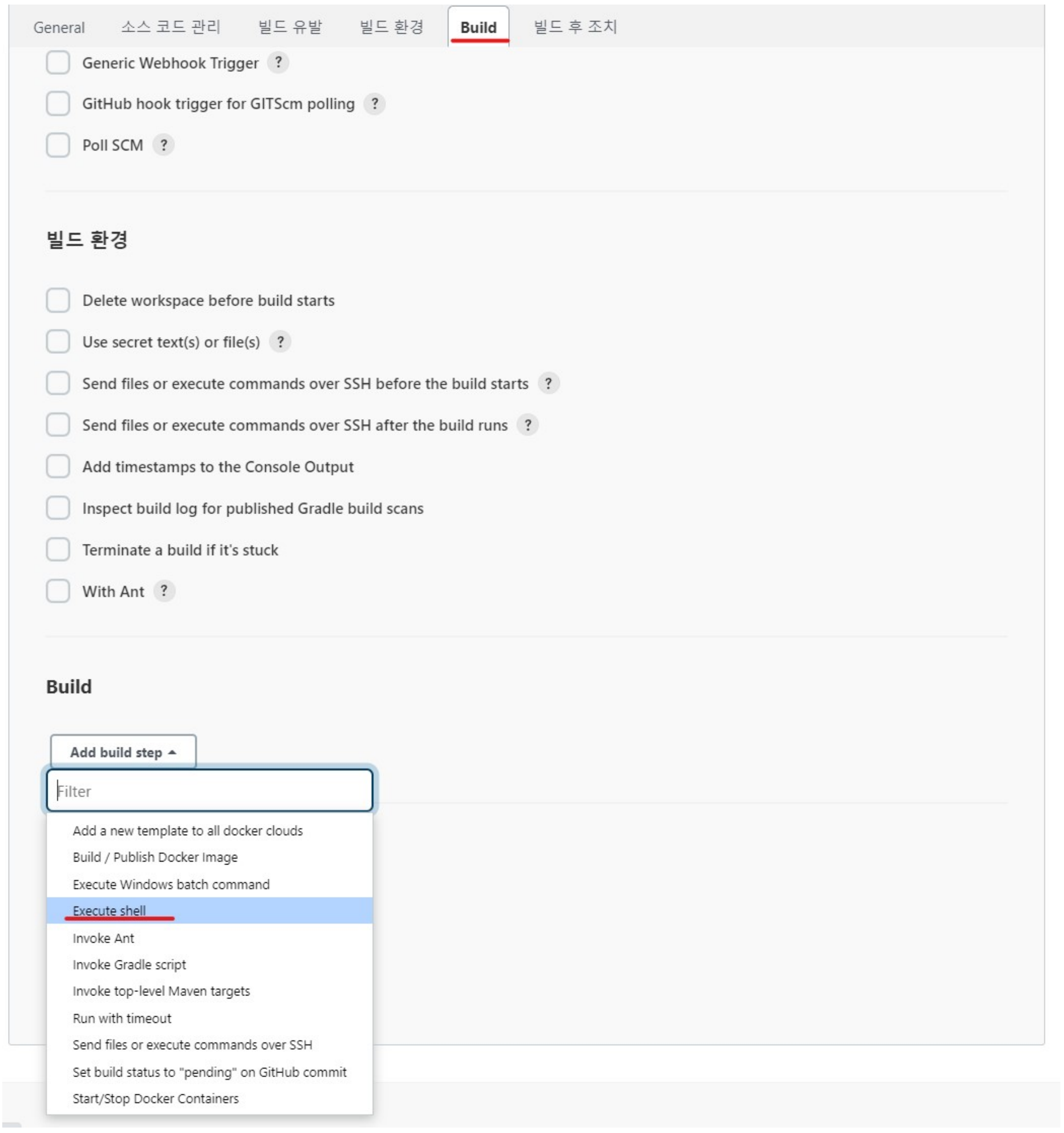
○ Filter branches by regex ?

[ ] Filter merge request by label

Secret token ?

[redacted]

- Build 탭 > Add build Step > Execute Shell 선택


General 소스 코드 관리 빌드 유발 빌드 환경 Build 빌드 후 조치

- [ ] Generic Webhook Trigger
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

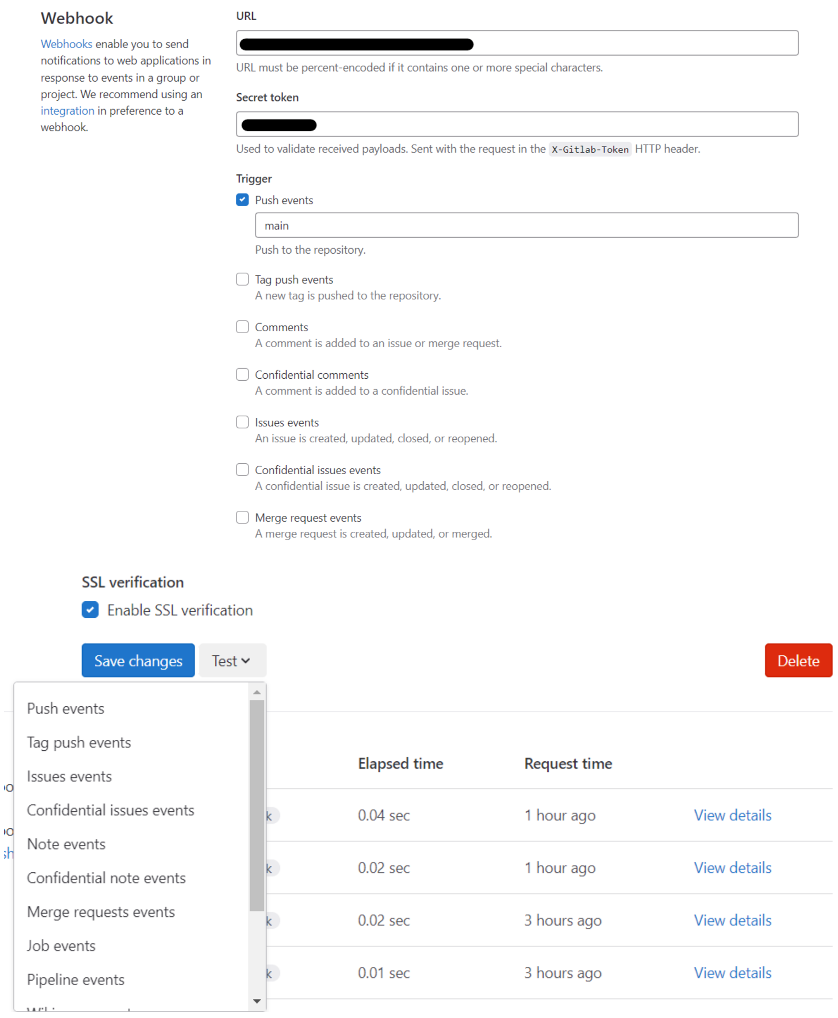
빌드 환경

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Send files or execute commands over SSH before the build starts
- [ ] Send files or execute commands over SSH after the build runs
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published Gradle build scans
- [ ] Terminate a build if it's stuck
- [ ] With Ant

Build

Add build step
Filter
Add a new template to all docker clouds
Build / Publish Docker Image
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Send files or execute commands over SSH
Set build status to "pending" on GitHub commit
Start/Stop Docker Containers

2. GitLab Settings > Webhooks



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

[redacted]

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

Secret token

[redacted]

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

Trigger

[✓] Push events

main

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.

SSL verification

[✓] Enable SSL verification

Elapsed time	Request time	
0.04 sec	1 hour ago	View details
0.02 sec	1 hour ago	View details
0.02 sec	3 hours ago	View details
0.01 sec	3 hours ago	View details

3. GitLab Webhooks의 URL과 Secret token에 Jenkins의 URL과 Secret token 작성
4. 원하는 브랜치에 push할 때마다 자동 빌드 되도록 설정 (develop)

(5) Execute Shell

```
echo "Run BE"
if (docker ps | grep "BackEnd"); then docker stop BackEnd; fi
if (docker images | grep "backimg"); then docker rmi backimg; fi
docker build -t backimg ./Development/BE/churest
docker run -it -d --rm -p 8080:8080 --name BackEnd backimg

echo "Run FE"
if (docker ps | grep "FrontEnd"); then docker stop FrontEnd; fi
if (docker images | grep "frontimg"); then docker rmi frontimg; fi
docker build -t frontimg ./Development/FE/churest
docker run -it -d --rm -p 3000:3000 --name FrontEnd frontimg
```

4. 외부 서비스

1) Kakao

- **기본 정보**
- **redirect**

```
https://k8a505.p.ssafy.io/redirect
http://localhost:3000/redirect
```

- **수집 정보**
    - 닉네임, 카카오계정 (이메일)

2) Google Cloud Storage

- **키 발급**
    - google cloud platform console 접속 후 project 생성
    - Storage > browser에서 버킷 생성
- **SpringBoot 설정**
    - build.gradle

```
implementation group: 'com.google.cloud', name: 'spring-cloud-gcp-starter', version: '3.4.3'
implementation group: 'com.google.cloud', name: 'spring-cloud-gcp-storage', version: '3.4.3'
```

- application.properties

```
# Google cloud key
spring.cloud.gcp.storage.credentials.location=classpath:churest-project-a89b305ab00e.json
```