

# 포팅 매뉴얼

## 메이븐으로 backend 서버 빌드 배포하기

부여받은 EC2 주소로 접속을 위해 CLI 도구를 받습니다.

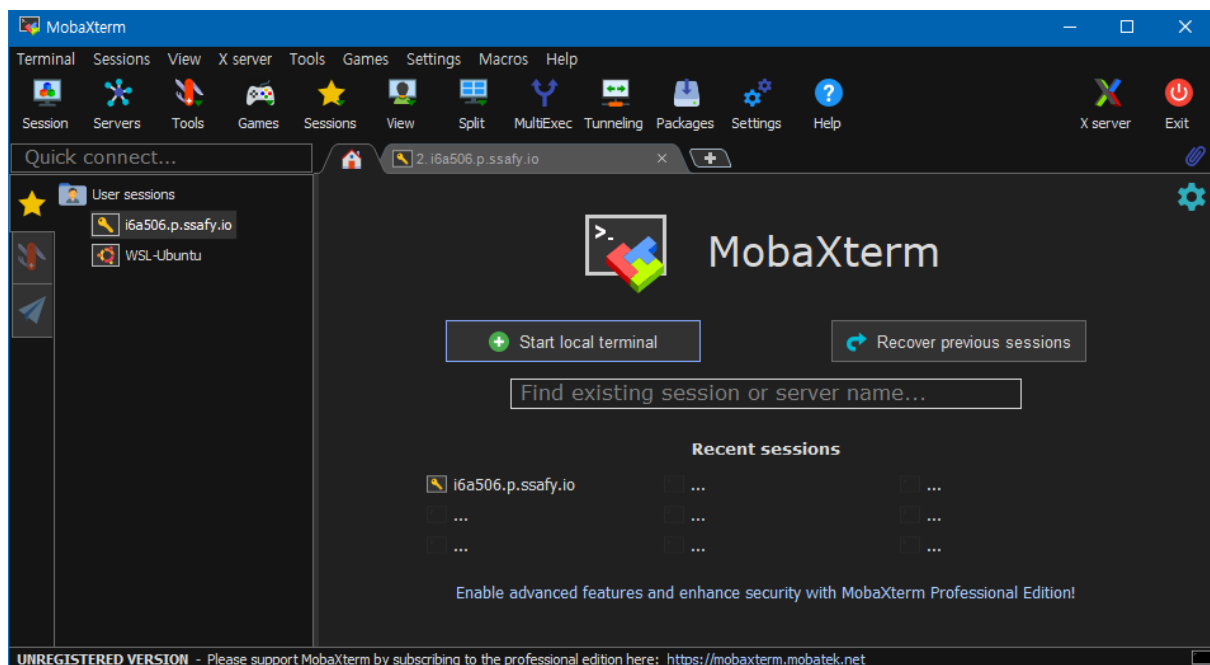
저희의 경우 mobaXterm을 사용하였고,

배포 과정에서도 이를 사용한다는 가정하에 진행하겠습니다.

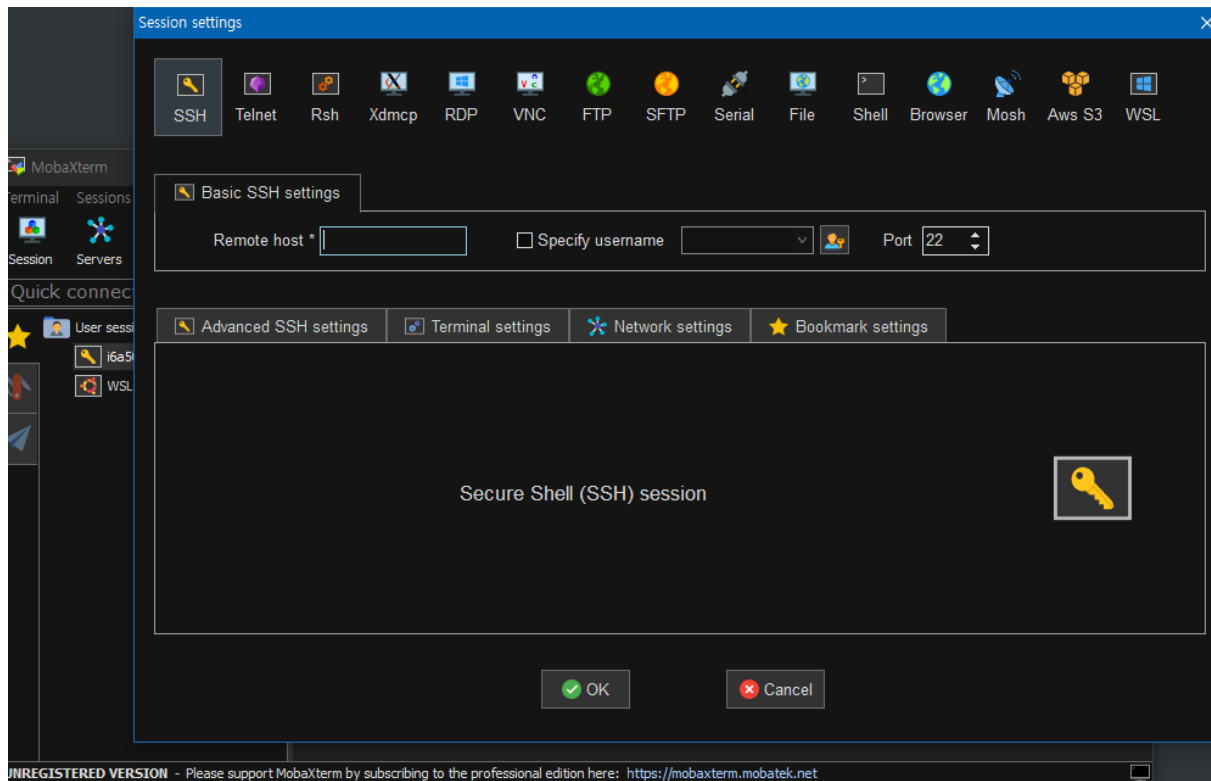
<https://mobaxterm.mobatek.net/download.html>

접속 후 home edition으로 다운로드

압축을 풀어주면 mobaxterm 실행파일 생성



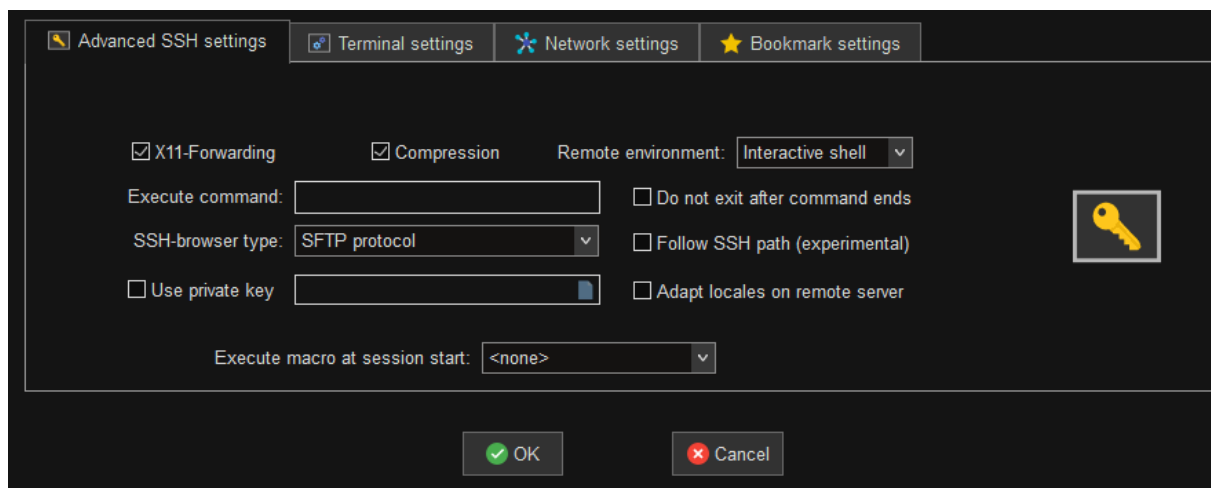
접속화면



세션 클릭 후 SSH 클릭

host에 부여받은 주소 (<http://i6a506.p.ssafy.io/>) 입력

username에는 ubuntu 입력



Advanced SSH settings 에서 sshkey.pem 파일을 매칭 시켜서

OK 클릭하면 SSH 접속이 가능합니다.

EC2 접속 후

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

## 자바 설치

```
~$ sudo apt-get install openjdk-8-jdk
```

## 메이븐 설치

```
~$ sudo apt install maven
```

## 버전 확인

```
~$ java -version
openjdk version "1.8.0_312"
OpenJDK Runtime Environment (build 1.8.0_312-8u312-b07-0ubuntu1~20.04-b07)
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)

~$ mvn -v
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 1.8.0_312, vendor: Private Build, runtime: /usr/lib/jvm/java-8-openjdk-amd64/jre
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "5.4.0-1018-aws", arch: "amd64", family: "unix"
```

## 깃 클론

```
git clone https://lab.ssafy.com/s06-webmobile2-sub2/S06P12A506
```

## 명령어 입력후 계정과 비밀번호 입력

## 프로젝트가 있는 디렉토리에서

```
~/S06P12A506/backend$ mvn package
```

## target 디렉터리 생성 뒤 [persona-0.0.1-SNAPSHOT.jar] 파일 생성

## 이후 실행 명령어를 입력해주면

```
~/S06P12A506/backend/target$ java -jar persona-0.0.1-SNAPSHOT.jar
```

```
generated sources/.../person-0.0.1-SNAPSHOT.jar
ubuntu@ip-172-26-7-250:~/S06P12A506/backend/target$ java -jar persona-0.0.1-SNAPSHOT.jar

:: Spring Boot ::
(v2.6.3)

2022-02-18 11:08:48.932 INFO 224800 --- [main] com.ssafy.persona.PersonaApplication : Starting PersonaApplication v0.0.1-SNAPSHOT using Java 1.8.0_312 on ip-172-26-7-250 with PID
224800 (/home/ubuntu/S06P12A506/backend/target/persona-0.0.1-SNAPSHOT.jar started by ubuntu in /home/ubuntu/S06P12A506/backend/target)
2022-02-18 11:08:48.935 DEBUG 224800 --- [main] com.ssafy.persona.PersonaApplication : Running with Spring Boot v2.6.3, Spring v5.3.15
2022-02-18 11:08:48.935 INFO 224800 --- [main] com.ssafy.persona.PersonaApplication : No active profile set, falling back to default profiles: default
2022-02-18 11:08:50.059 INFO 224800 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JDBC repositories in DEFAULT mode.
2022-02-18 11:08:50.079 INFO 224800 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 14 ms. Found 0 JDBC repository interfaces.
2022-02-18 11:08:50.772 INFO 224800 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8082 (http)
2022-02-18 11:08:50.785 INFO 224800 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2022-02-18 11:08:50.785 INFO 224800 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.56]
2022-02-18 11:08:50.847 INFO 224800 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2022-02-18 11:08:50.847 INFO 224800 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1852 ms
2022-02-18 11:08:52.260 INFO 224800 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2022-02-18 11:08:52.710 INFO 224800 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2022-02-18 11:08:52.879 INFO 224800 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8082 (http) with context path ''
2022-02-18 11:08:53.074 INFO 224800 --- [main] com.ssafy.persona.PersonaApplication : Started PersonaApplication in 5.229 seconds (JVM running for 5.782)
```

해당 화면과 함께 실행이 되고 접속이 가능해집니다.

## AWS에 도커 설치

```
~$ sudo apt install apt-transport-https
~$ sudo apt install ca-certificates
~$ sudo apt install curl
```

컬은 특정한 웹사이트에서 데이터를 다운받을때 사용.

### Docker 공식 GPG 키 등록

```
~$ sudo apt install software-properties-common
~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

### 저장소 설정

```
~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"
~$ sudo apt update
```

### 도커 설치

```
~$ apt-cache policy docker-ce
~$ sudo apt install docker-ce
```

## 도커 명령어

### 도커 실행 로그 실시간 확인

→ docker logs -t -f [docker name]

### 도커 리스트 확인

→ docker ps (-a)

-a 옵션을 추가할 경우 실행중인 컨테이너만 확인 가능

도커 중지/시작

`docker stop/start [docker name]`

도커 컨테이너 삭제

→ `docker rm [docker name]`

도커 이미지 파일 확인

→ `docker images`

도커 이미지 삭제

→ `docker rmi [image name]`

도커 볼륨 리스트 확인

→ `docker volume ls`

도커 사용하지 않는 볼륨 삭제

→ `docker volume prune (y/n)`

## docker에 mysql 실행

mysql의 경우 docker 에 이미지가 존재하기 때문에 실행 가능

`docker run -d -p 3306:3306 - --name my sql -e MYSQL_ROOT_PASSWORD=ssafytest1234 -d mysql:8.0`

mysql docker 접속 하여 확인

```
~$docker exec -it mysql /bin/bash
root@a1286eac2ea0:/# mysql -u root -p
```

```
ubuntu@ip-172-26-7-250:~/S06P12A506/backend/target$ docker exec -it mysql /bin/bash
root@a1286eac2ea0:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2095
Server version: 8.0.28 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

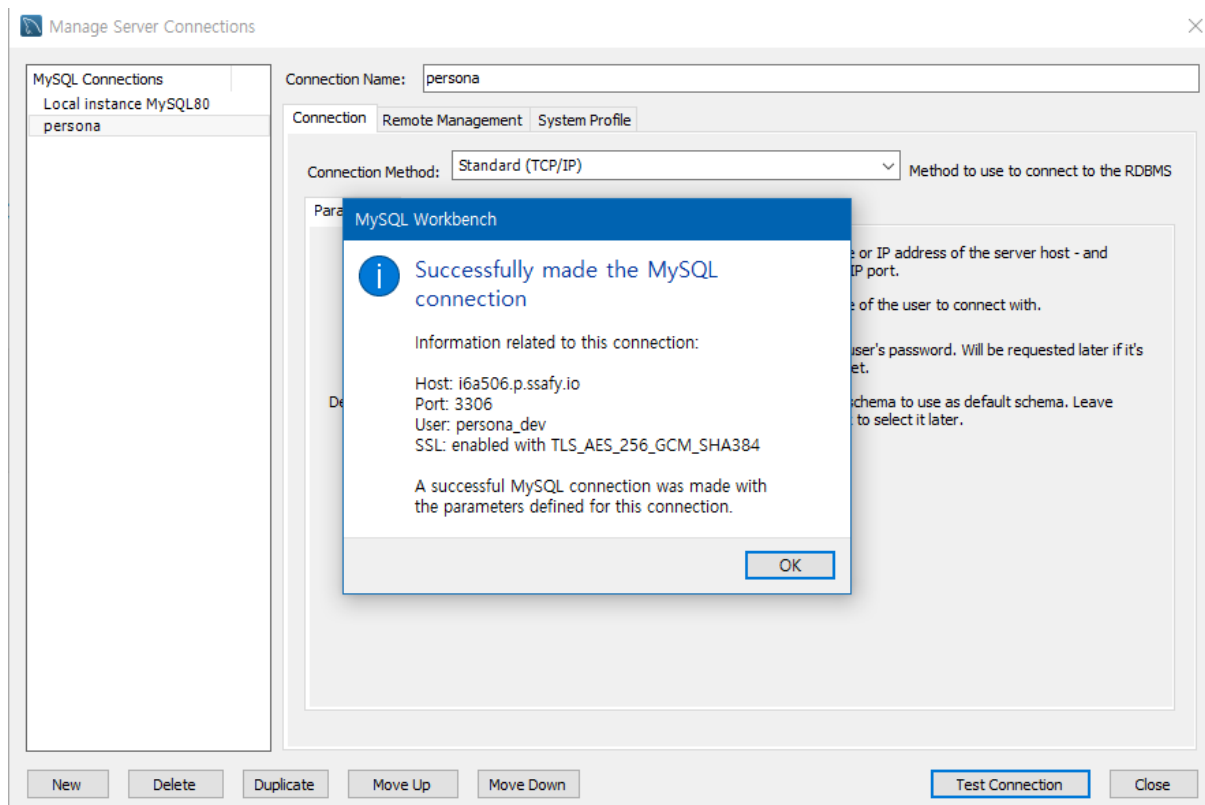
mysql>
```

mysql접속 성공!

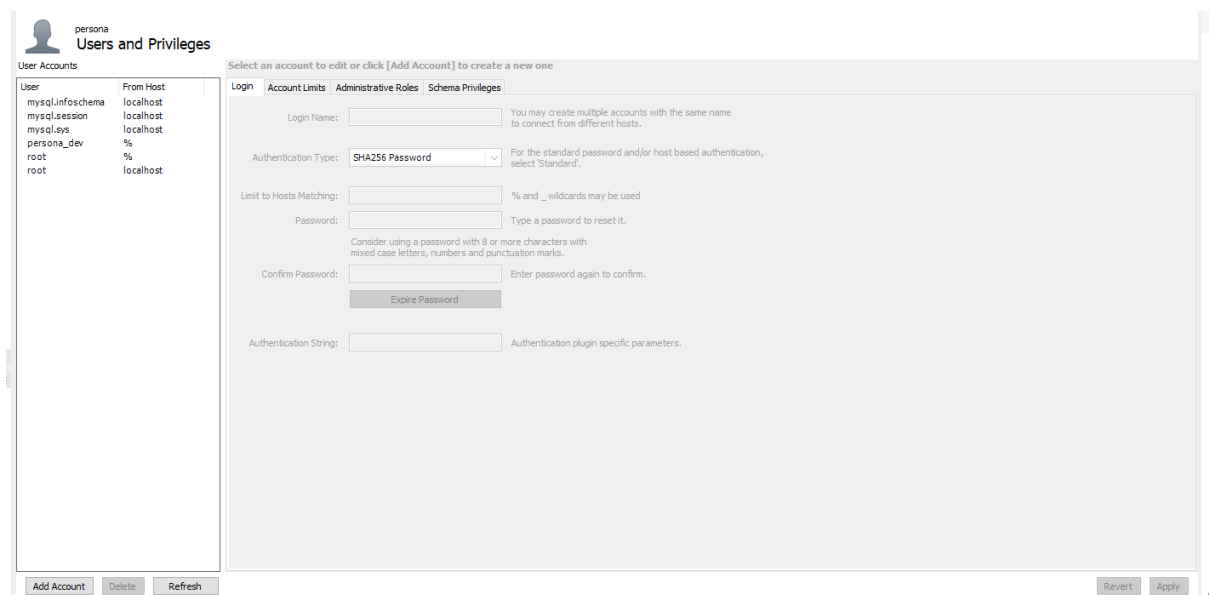
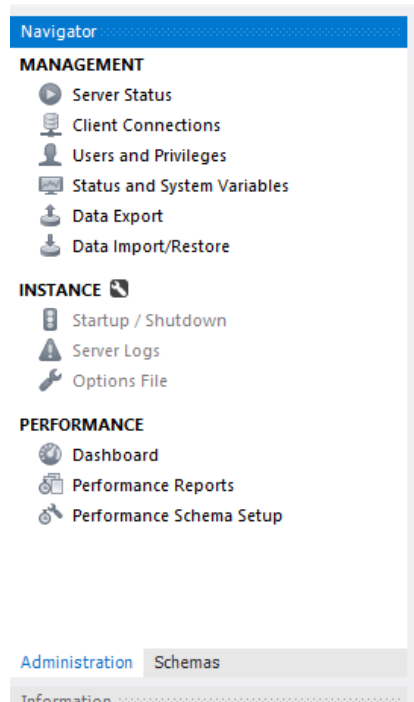
이후는 workbench에서 연결하여 user 생성 및 권한 부여  
스키마, table 설정이 가능합니다.

<https://downloads.mysql.com/archives/installer/>

접속 하여 window에 mysql 다운로드



접속하여 좌측 Administration 클릭 후 Users and Privileges 클릭



적절한 권한을 지닌 계정을 생성하여 불필요한 root 계정 사용을 막을 수 있습니다.

## 젠킨스 설치

작업의 편리를 위한 도커의 권한부여

```
sudo chmod 006 /var/run/docker.sock
```

이미지 다운로드

```
docker pull jenkins/jenkins:jdk11
```

## 젠킨스 컨테이너 실행

```
docker run -d --restart always -p 8080:8080 -v /jenkins:/var/jenkins_home --name jenkins -u root jenkins/jenkins:jdk11
```

- **d** 는 데몬, 백그라운드에서 계속 실행
- **-restart** 는 호스트 재부팅 시, 자동으로 컨테이너 재시작
- **p** 는 포트바인딩, EC2의 8080포트를 도커 컨테이너의 8080포트와 연결한다는 의미. 8080포트는 젠킨스의 디폴트 port
- **v** 는 볼륨 마운팅,
  - 볼륨은, 호스트(내 PC)의 디스크공간(볼륨)을 도커 컨테이너와 함께 사용, 그래서 /jenkins라는 디렉토리를 컨테이너 내의 /var/jenkins\_home과 연동하게 되는데, 컨테이너에서도 해당 디렉토리 내부에 read+write가 가능하고, 호스트에서도 가능. 볼륨을 연결하므로써 컨테이너가 삭제되더라도 지워져선 안 되는 데이터를 보존 또는 쉽게 호스트와 컨테이너와의 파일을 공유가능
- **-name** 은 컨테이너의 이름. 여기서는 jenkins로 설정
- **u** 는 유저 이름. 여기서는 root로 설정

## 웹 콘솔로 접속

<http://i6a506.p.ssafy.io:8080/>

로 접속하면 젠킨스 기본 설정 페이지로 이동



## 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

초기 비밀번호 입력 요구가 뜨는데 해당 부분에서 확인 가능  
(또는 젠킨스를 실행할 때 로그로도 확인 가능합니다.)

비밀번호를 입력하면 해당창이 뜹니다.



## 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.

추천 플러그인을 설치한 후

## Getting Started

---

# Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

---

Jenkins 2.295

[Skip and continue as admin](#)

[Save and Continue](#)

젠킨스 계정을 생성합니다. (해당 계정으로 젠킨스에 로그인할 수 있습니다.)

# 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 2.295

Not now

Save and Finish

마지막 URL 입력창에는 EC2의 퍼블릭 IP를 적으면 됩니다.

The screenshot shows the Jenkins Dashboard. On the left is a sidebar with navigation links: 새로운 Item, 사람, 빌드 기록, 프로젝트 연관 관계, 파일 링거프린트 확인, Jenkins 관리, My Views, Lockable Resources, and New View. Below these are sections for '빌드 대기 목록' (Build Queue) and '빌드 실행 상태' (Build Execution Status). The main area displays a table of jobs:

S	W	Name	최근 성공	최근 실패	최근 소요 시간
✓	⚙	front_git	56 min #153	2 days 0 hr #111	1 min 13 sec
✓	⚙	gitlab_connect	2 hr 30 min #51	4 days 19 hr #35	18 sec

Below the table, there are tabs for '아이콘: S M L' and a section for 'Icon legend' with links to 'Atom feed 모두', 'Atom feed 실패', and 'Atom feed 최근 빌드'.

접속 성공 시 화면

## 도커파일로 백엔드 컨테이너 제작

```
~$ cd ~/S06P12A506/backend/target/
```

아까 git으로 받은 프로젝트로 이동해서 아래의 명령어로 Dockerfile 파일 작성

```
~/S06P12A506/backend/target$ vim Dockerfile
```

```
FROM openjdk:8-jdk-alpine
ADD persona-0.0.1-SNAPSHOT.jar persona-0.0.1-SNAPSHOT.jar
ENTRYPOINT ["java", "-jar", "persona-0.0.1-SNAPSHOT.jar"]
```

vi는 리눅스의 텍스트 문서 작성 프로그램으로 vi 또는 vim으로 생성/수정할 수 있습니다.

작성 시에는 insert에 해당하는 i 로 들어가서 작성하고 작성 후에는 esc 키를 눌러 저장 모드로 나올 수 있습니다.

저장

→ :wq

강제 종료

→ :q!

docker 컨테이너 생성

```
docker build -t back .
```

back이라는 도커 이미지가 생성된것을 확인

```
docker images
```

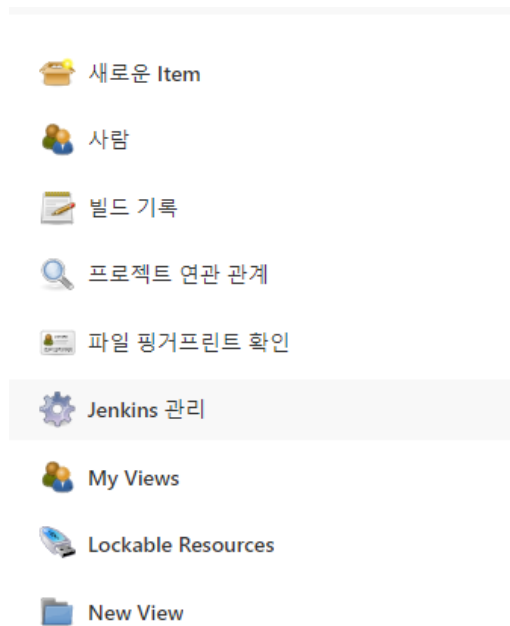
```
ubuntu@ip-172-26-7-250:~/S06P12A506/backend/target$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
front         latest    92aabe09861c   About an hour ago   494MB
back          latest    f8908d19f260   3 hours ago       141MB
```

docker 실행

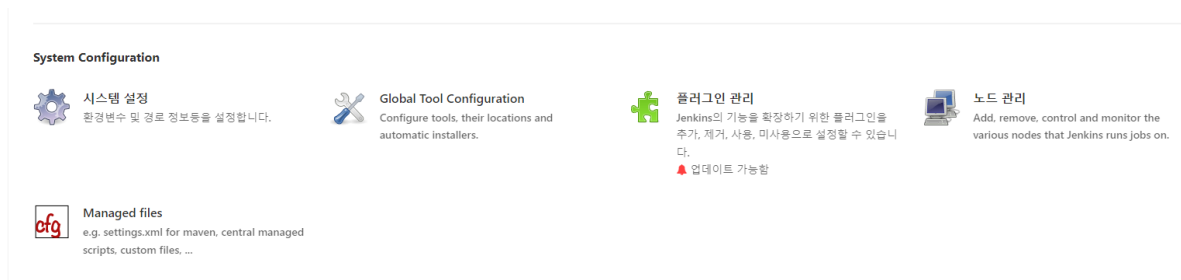
```
sudo docker run -d --name back -v files:/files -p 8082:8082 back
```

파일 정보를 저장하기 위해 files 디렉터리를 마운트 하여 실행

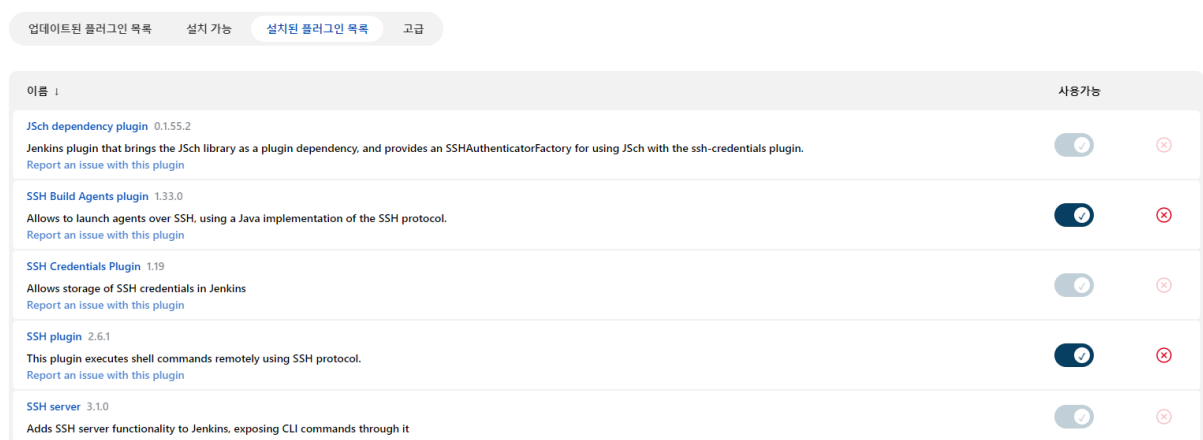
## 젠킨스 설정



## 젠킨스 관리에 들어가서




## 플러그인 관리 클릭





SSH 라고 검색하여 위의 해당하는 플러그인을 설치 후 재시작 해줍니다.


## Security 항목에서 Manage Credentials 클릭


**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

**In-process Script Approval**  
Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions.

Kind

Username with password

Username with password

GitHub App

GitLab API token

GitLab Personal Access Token

SSH Username with private key

Secret file

Secret text

Certificate

☐ Treat username as secret ?

Password ?

ID ?

Description ?

OK

해당 항목으로 인증키를 생성 해 줍니다.

목록에 생성된 것을 확인 가능 합니다.

다시 시스템 설정으로 돌아가서

## SSH remote hosts

### SSH sites

Hostname ?

172.26.7.250

Port ?

22

Credentials

ubuntu (AWS EC2) ▾

➕Add ▾

Pty ?

serverAliveInterval ?

0

timeout ?

0

Check connection

삭제


SSH sites 로 연결후 Credentials 에서 방금 추가한 키를 선택

Successfull connection


Check connection


Check connection을 누르면 성공하는 것을 확인할 수 있습니다.


이후 새로운 아이템을 클릭


 새로운 Item

 사람


 빌드 기록


 프로젝트 연관 관계

 파일 핑거프린트 확인

 Jenkins 관리

 My Views


 Lockable Resources


 New View


**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.



If you want to create a new item from other existing, you can use this option:

 Copy from


**OK**

이름을 지정하고 Freestyle project 로 생성해줍니다.

**Dashboard** ▶ **gitlab\_connect** ▶

-  대시보드로 돌아가기
-  상태
-  변경사항
-  작업공간
-  Build Now
-  구성
-  Project 삭제
-  Rename

## Project gitlab\_connect

 **작업 공간**
 **최근 변경사항**

### 고정링크

- Last build, (#51), 2 hr 46 min 전
- Last stable build, (#51), 2 hr 46 min 전

해당 프로젝트 내부로 들어와서 구성을 선택하고



**빌드 환경**

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s) ?
- ☐ Provide Configuration files ?
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☒ Execute shell script on remote host using ssh ?

**SSH site**

ubuntu@172.26.7.250:22

**Pre build script** ?

```
rm /home/ubuntu/S06P12A506/backend/target/persona-0.0.1-SNAPSHOT.jar
rm /home/ubuntu/S06P12A506/backend/target/persona-0.0.1-SNAPSHOT.jar.original
```

**Post build script** ?

```
cd /home/ubuntu/S06P12A506/backend/
git pull origin backend
mvn package
cd target/
sudo docker rmi old_back
sudo docker commit back old_back
sudo docker stop back
sudo docker rm back
sudo docker build -t back .
sudo docker run -d --name back -v files:/files -p 8082:8082 back
```

☐ Hide command from console output

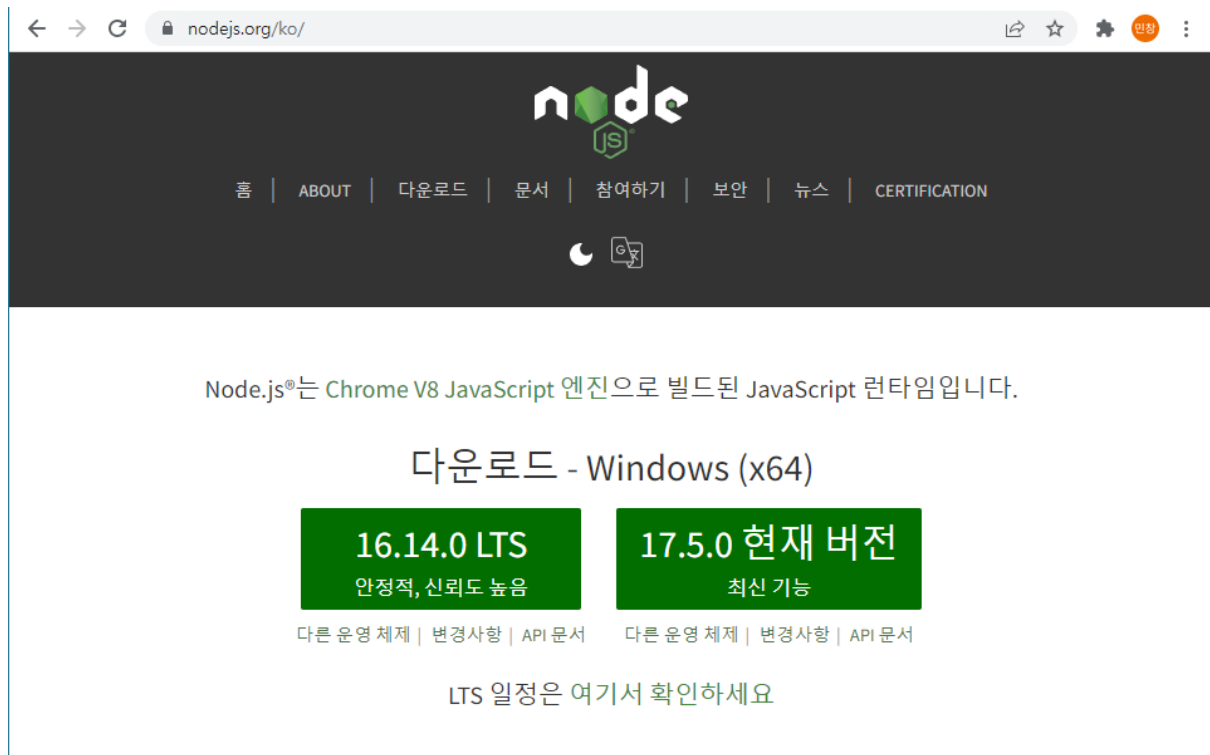
빌드 환경에서 ssh와 연결지어서 빌드를 눌렀을 때 ec2에서 실행될 명령어들을 작성해줍니다.

Build History		추이 ^
Filter builds...		
✓ #51	2022. 2. 18. 오전 9:05	
✓ #50	2022. 2. 17. 오후 7:59	
✓ #49	2022. 2. 17. 오전 11:14	
✓ #48	2022. 2. 17. 오전 10:13	
✓ #47	2022. 2. 16. 오후 3:13	
✓ #46	2022. 2. 16. 오후 3:08	
✓ #45	2022. 2. 16. 오전 11:28	
✓ #44	2022. 2. 16. 오전 2:28	
✓ #43	2022. 2. 16. 오전 1:23	
✓ #42	2022. 2. 15. 오후 6:44	
✓ #41	2022. 2. 15. 오후 4:49	
✓ #40	2022. 2. 14. 오후 5:01	
✓ #39	2022. 2. 14. 오후 4:41	
✓ #38	2022. 2. 14. 오후 4:06	

이후 빌드가 원활히 되는 것을 확인할 수 있습니다.

# FrontEnd

- [Node.org](https://nodejs.org) 접속



- **16.13.2** 버전 설치
- 명령 프롬프트에서 `node -v` 입력 후, 버전이 나오는지 확인

```
C:\Users\pang>node -v
v16.13.2
```

- git 파일 다음과 같은 명령어로 pull

```
# git 클론
git clone (경로)

# frontend 브랜치 생성 및 이동
git branch -b frontend

# 파일 pull
git pull origin frontend
```

- 패키지 설치 및 실행

```
# frontend 폴더로 이동
cd frontend
```

```
# frontend 패키지 설치  
npm i  
  
# 실행  
npm run start
```