



[배포 가이드]

0. 환경 설정

1. 데이터베이스

1.1 MySQL

1.1.1 MySQL 설치 가이드

1.1.2 MySQL 접속 (Card Company DB)

1.1.3 MySQL에 DB 스키마 생성

1.1.4 MySQL에 dump 데이터 삽입

1.1.5 MySQL 접속 (cardian DB)

1.1.6 MySQL에 DB 스키마 생성

1.1.7 MySQL에 dump 데이터 삽입

2. AiMaker BE

2.1 Git Clone

2.2 프로젝트 실행

2.2.1 IntelliJ에서 프로젝트 열기

2.2.2 gradle 설정

2.2.3 Dockerfile

2.2.4 .gitignore 파일 추가

2.2.5 프로퍼티 파일

3. AiMaker FE

3.2 프로젝트 실행

3.2.1 VSCode에서 프로젝트 열기

3.2.2 dependency 설치

3.2.3 Dockerfile 파일 추가

3.2.4 .gitignore 파일 추가

3.2.5 .env 파일 추가

4. AWS 배포

4.1 사용하는 요금제

4.2 서버 접속 방법 (SSH)

4.2.1 MobaXterm 사용

4.2.2 우분투 서버 접속

4.3 데이터 베이스 Docker Container 설치

4.3.1 MySQL DB

4.3.3 Jenkins (CI/CD)



배포 순서

1. 데이터베이스(MySQL)
2. AiMaker BE
3. AiMaker FE
4. AWS 배포

0. 환경 설정

- Microsoft Windows 10
- CPU 인텔 i7
- RAM 16 GB
- x64

1. 데이터베이스


1.1 MySQL

1.1.1 MySQL 설치 가이드

▼ MySQL Installer 설치

MySQL :: Download MySQL Installer

Note: MySQL 8.0 is the final series with MySQL Installer. As of MySQL 8.1, use a MySQL product's MSI or Zip archive for installation. MySQL Server 8.1 and higher also bundle MySQL Configurator, a tool that helps configure MySQL Server.

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

- version 8.0.36
- OS Microsoft Windows
- community download

General Availability (GA) Releases
Archives

MySQL Installer 8.0.36

Note: MySQL 8.0 is the final series with MySQL Installer. As of MySQL 8.1, use a MySQL product's MSI or Zip archive for installation. MySQL Server 8.1 and higher also bundle MySQL Configurator, a tool that helps configure MySQL Server.

Select Version:
8.0.36

Select Operating System:
Microsoft Windows

Windows (x86, 32-bit), MSI Installer	8.0.36	2.1M	Download
(mysql-installer-web-community-8.0.36.0.msi)			
MD5: 81061532541f716cf6c6e2c4881a154c Signature			
Windows (x86, 32-bit), MSI Installer	8.0.36	285.3M	Download
(mysql-installer-community-8.0.36.0.msi)			
MD5: d63232c190d0c9c294a2f8d776ed1c20 Signature			

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

• 설치 가이드

MySQL 다운로드 및 설치하기(MySQL Community 8.0)

SQL을 본격적으로 사용하려면 DBMS를 설치해야 합니다. 여러 가지 DBMS 중에서 MySQL 설치 하는 방법을 알아보고, 정상적으로 설치가 되었는지 확인하는 방법을 알아보겠습니다. 2021년 10월

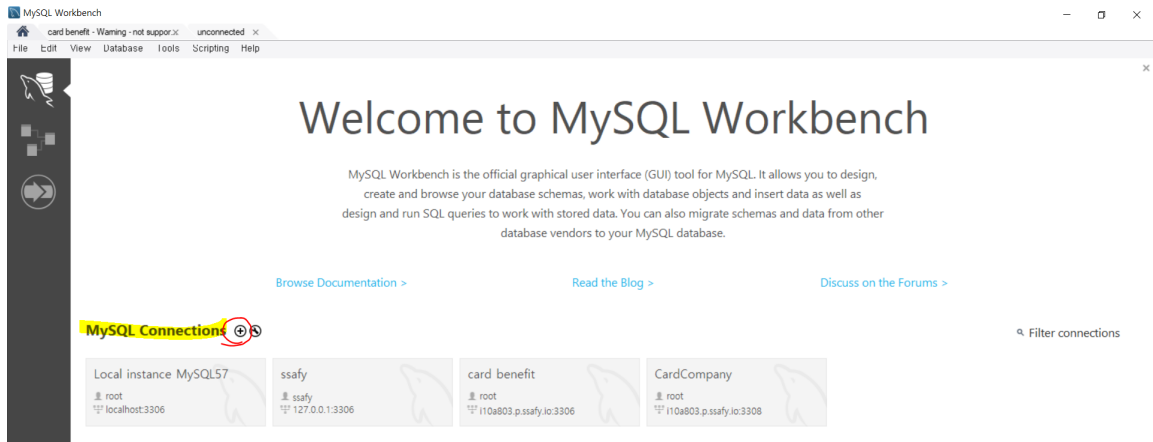
<https://hongong.hanbit.co.kr/mysql-다운로드-및-설치하기/mysql-community-8-0/>

▼ 15 中 계정 및 비밀번호

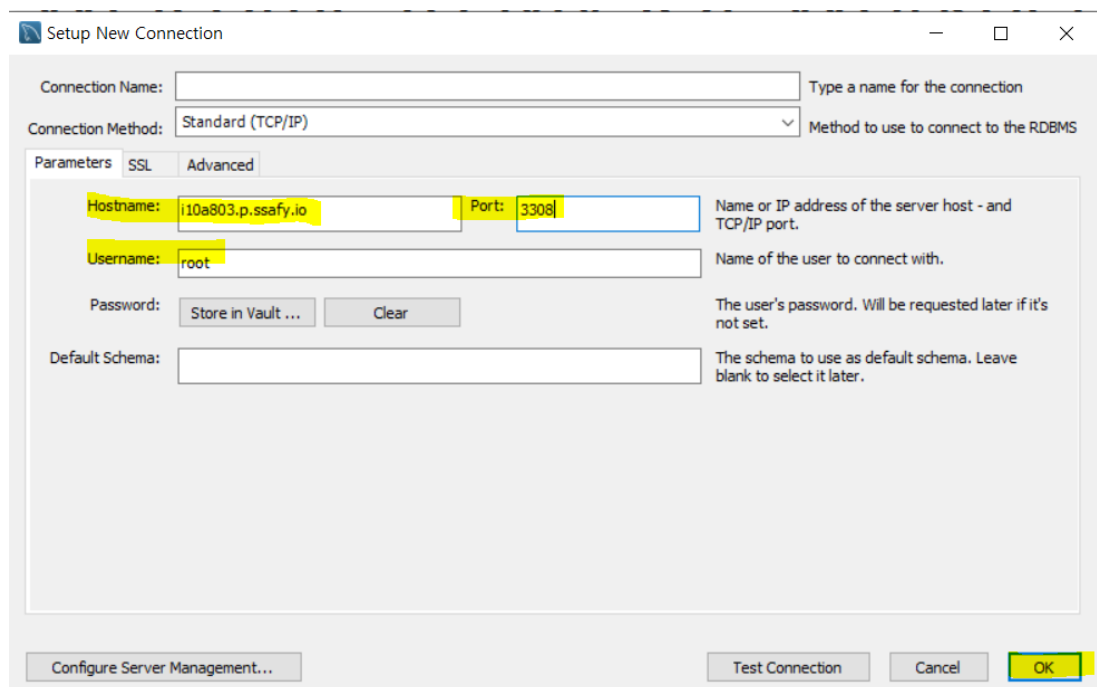
- 계정 : root
- 비밀번호 : a803

1.1.2 MySQL 접속 (Card Company DB)

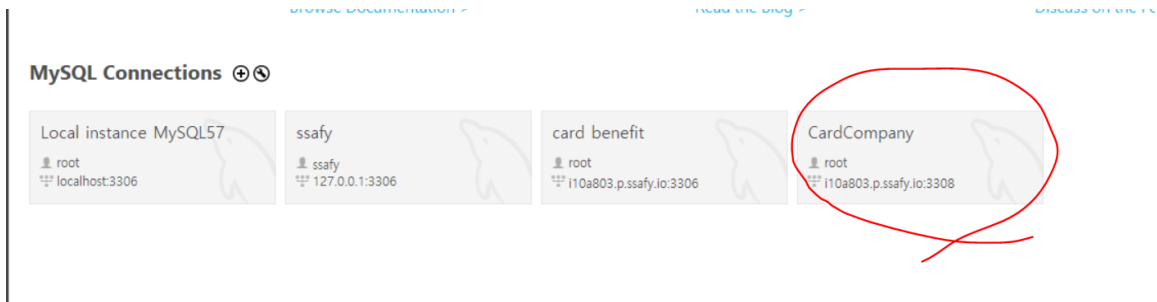
- MySQL Workbench 실행
- MySQL Connections 의 + 버튼 클릭



- Connection 설정
 - username : root
 - password : a803
 - port : 3306
 - localhost에서 접속한다면 Hostname을 localhost로 변경



- 생성된 connection 더블 클릭하여 DB에 접속

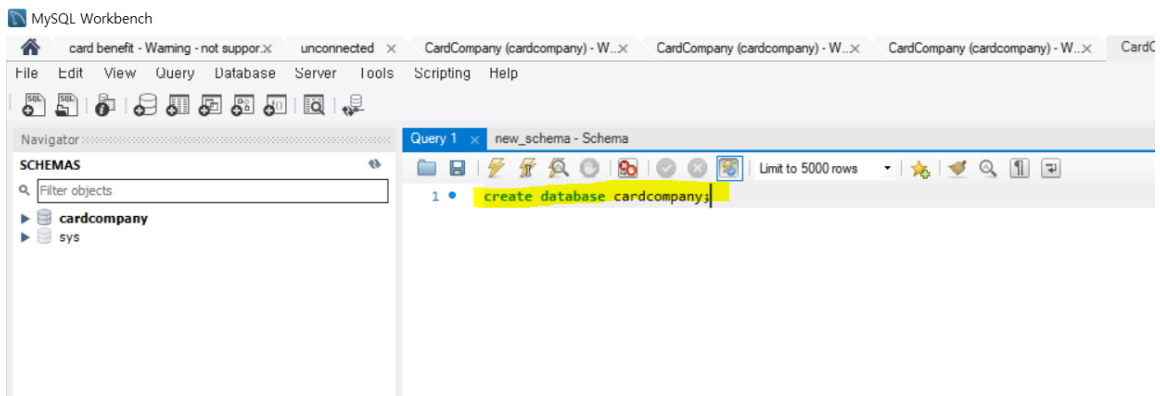


- 경고 문구가 나올 시 continue anyway 클릭

1.1.3 MySQL에 DB 스키마 생성

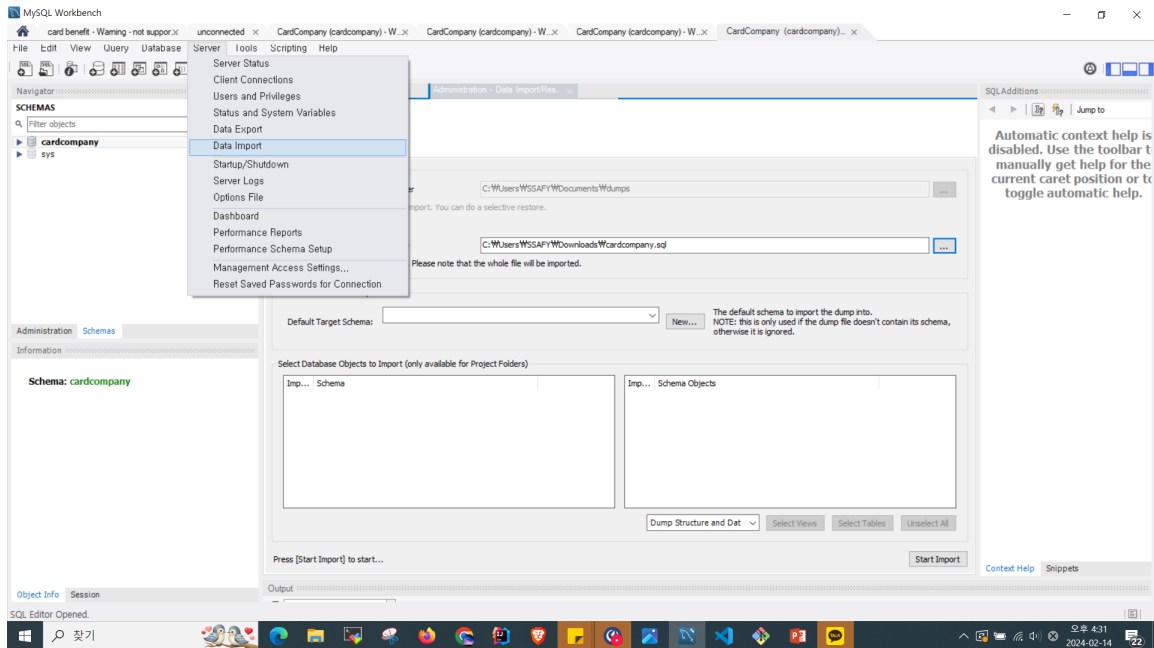
- Query에 다음과 같이 입력

```
create database cardcompany;
```

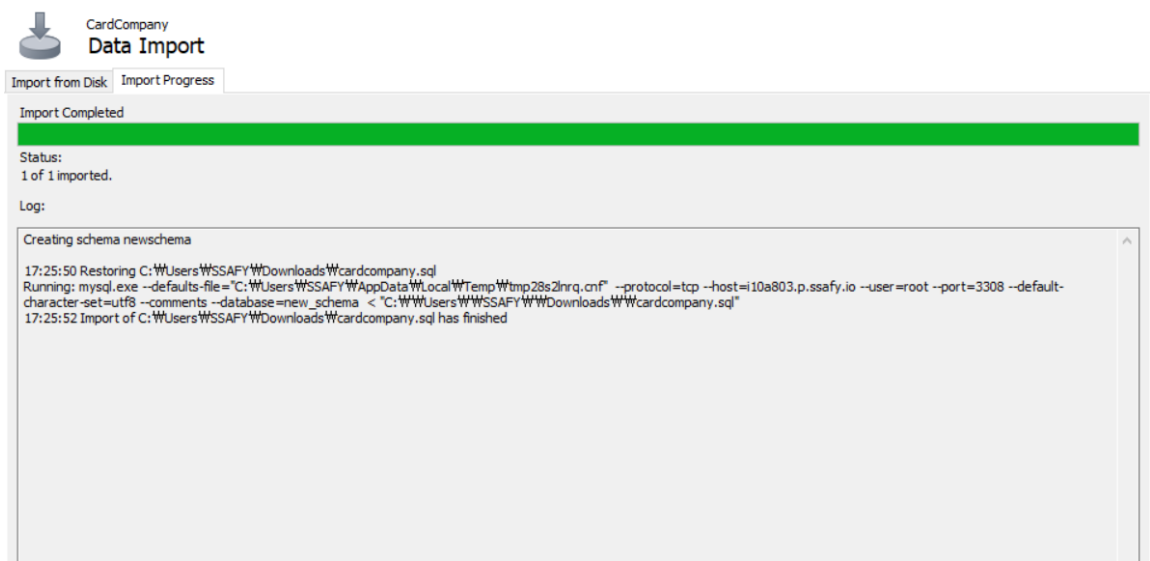


1.1.4 MySQL에 dump 데이터 삽입

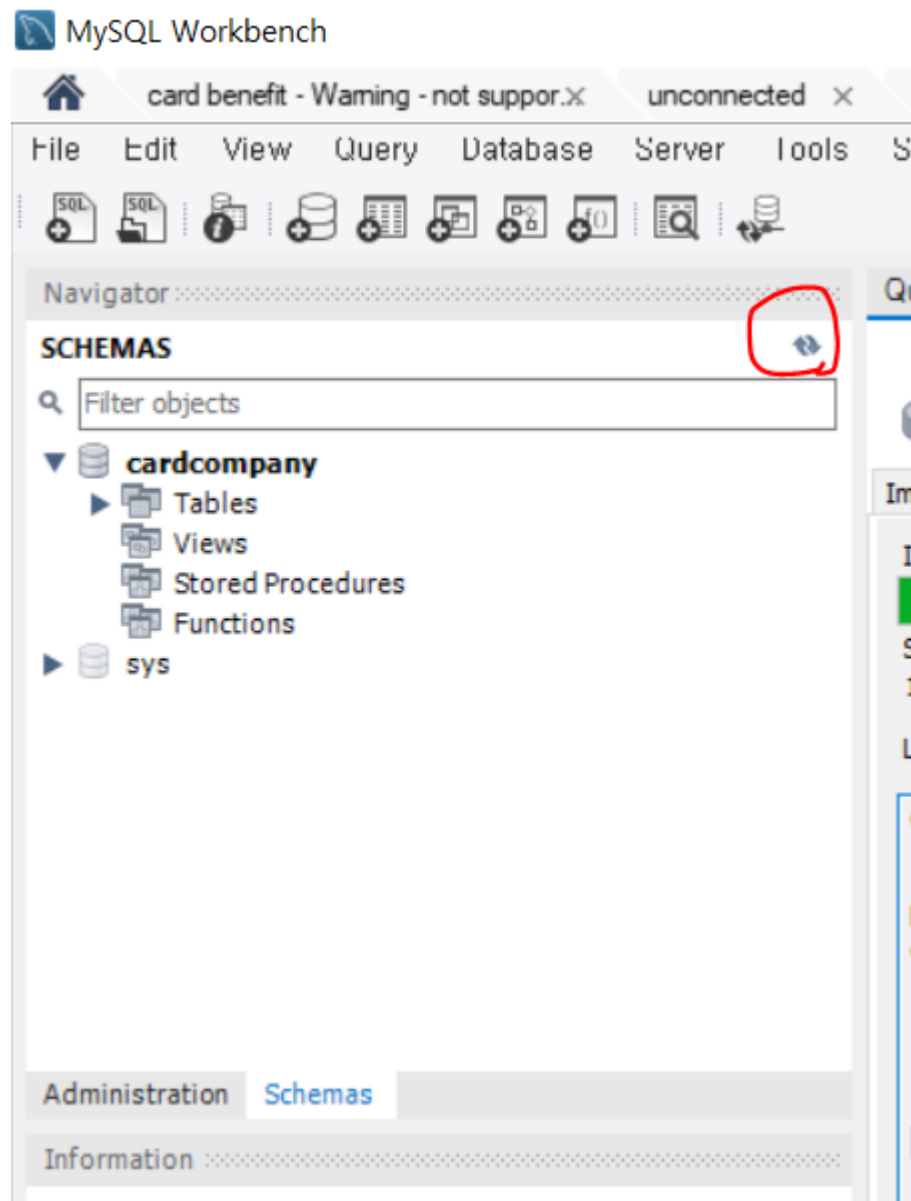
- server > Data Import > Import from Self-Contained File > Card Company.sql 선택



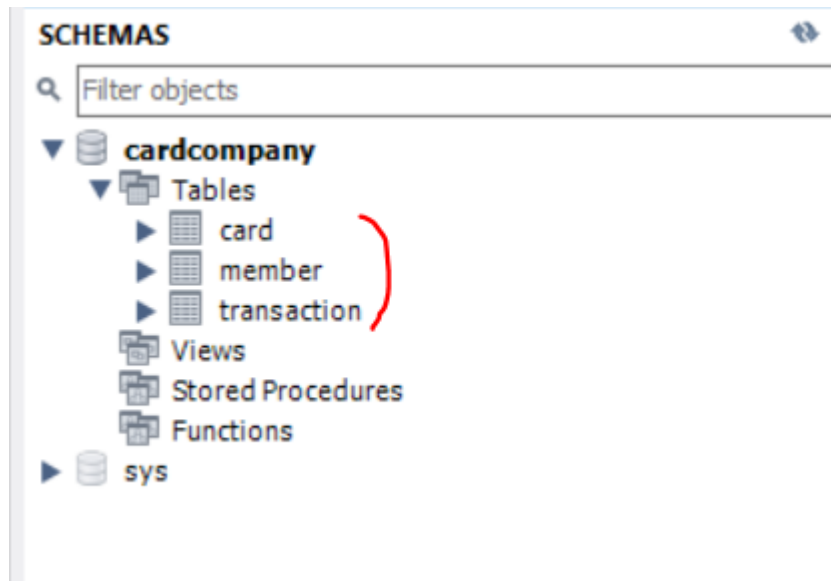
- 오른쪽 아래 start import 클릭
- import 성공시 아래와 같은 화면



- SCHEMAS 오른쪽의 새로고침 표시 클릭

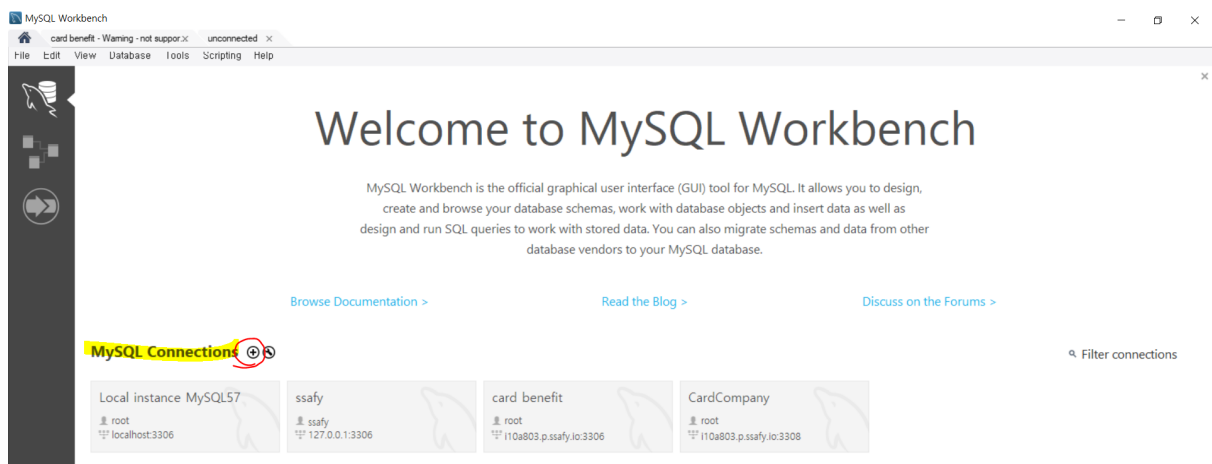


- 테이블 목록이 잘 들어갔는 지 확인



1.1.5 MySQL 접속 (cardian DB)

- MySQL Workbench 실행
- MySQL Connections 의 + 버튼 클릭



- Connection 설정
 - username : root
 - password : a302
 - port : 3306
 - localhost에서 접속한다면 Hostname을 localhost로 변경

Setup New Connection

Connection Name: Type a name for the connection

Connection Method: Method to use to connect to the RDBMS

Parameters SSL Advanced

Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

Password: Store in Vault ... Clear The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

Configure Server Management... Test Connection Cancel OK

- 생성된 connection 더블 클릭하여 DB에 접속
- 경고 문구가 나올 시 continue anyway 클릭

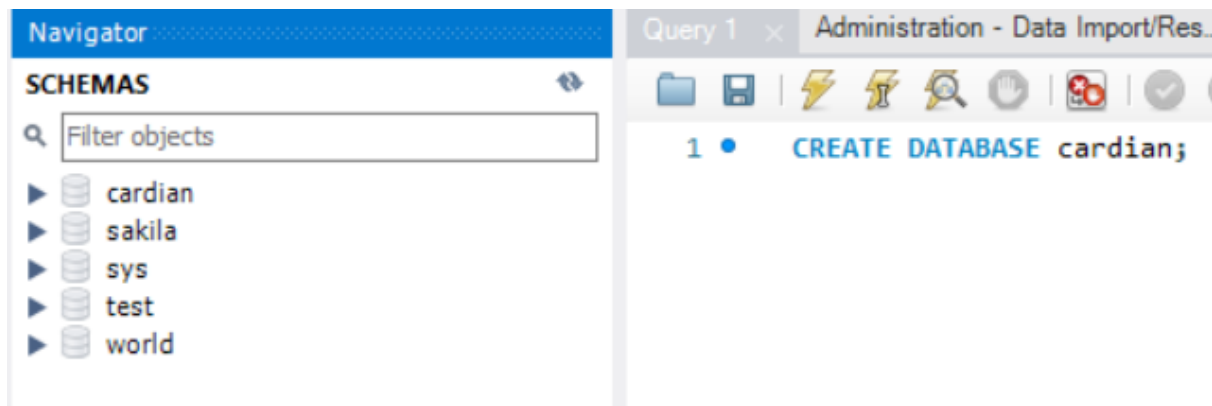
MySQL Connections + ⓘ

Connection Name	User	Host/Port
Local instance MySQL (auto...)	root	localhost:3306
cardiant	root	127.0.0.1:3306
cardian	root	i10a803.p.ssafy.io:3306

1.1.6 MySQL에 DB 스키마 생성

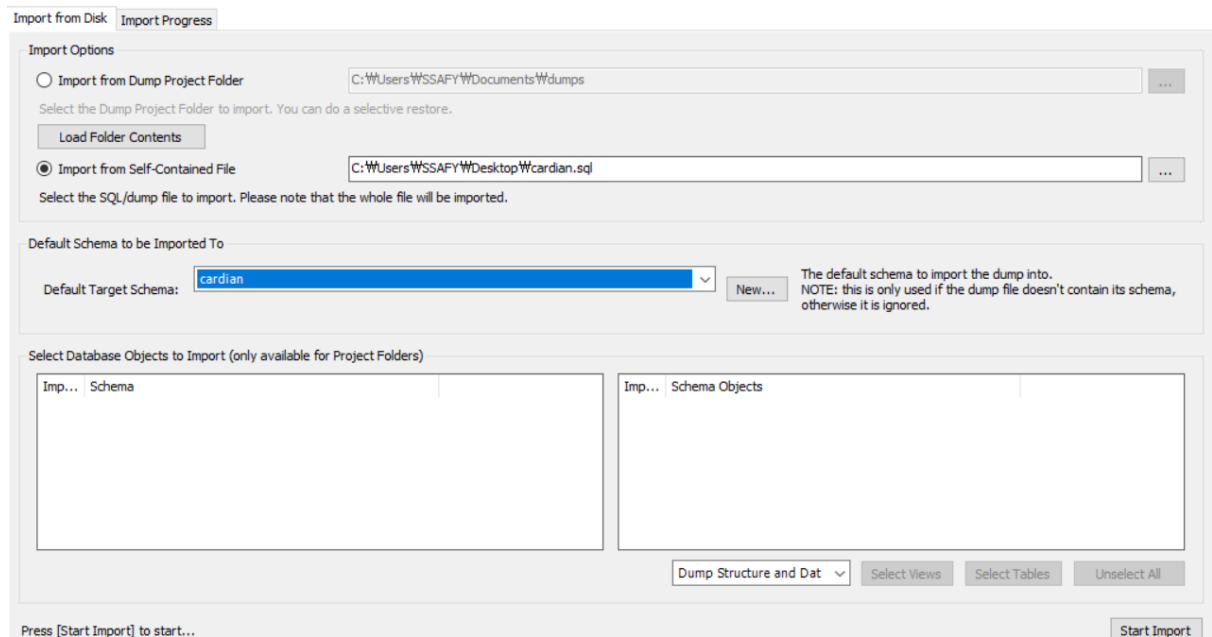
- Query에 다음과 같이 입력

```
CREATE DATABASE cardian;
```

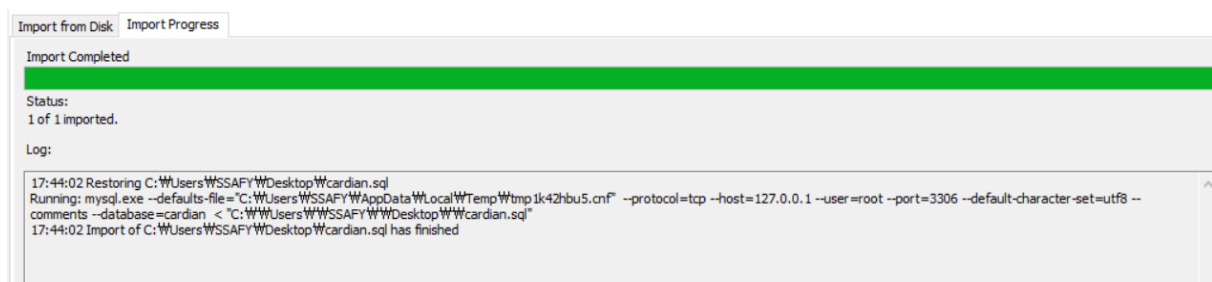


1.1.7 MySQL에 dump 데이터 삽입

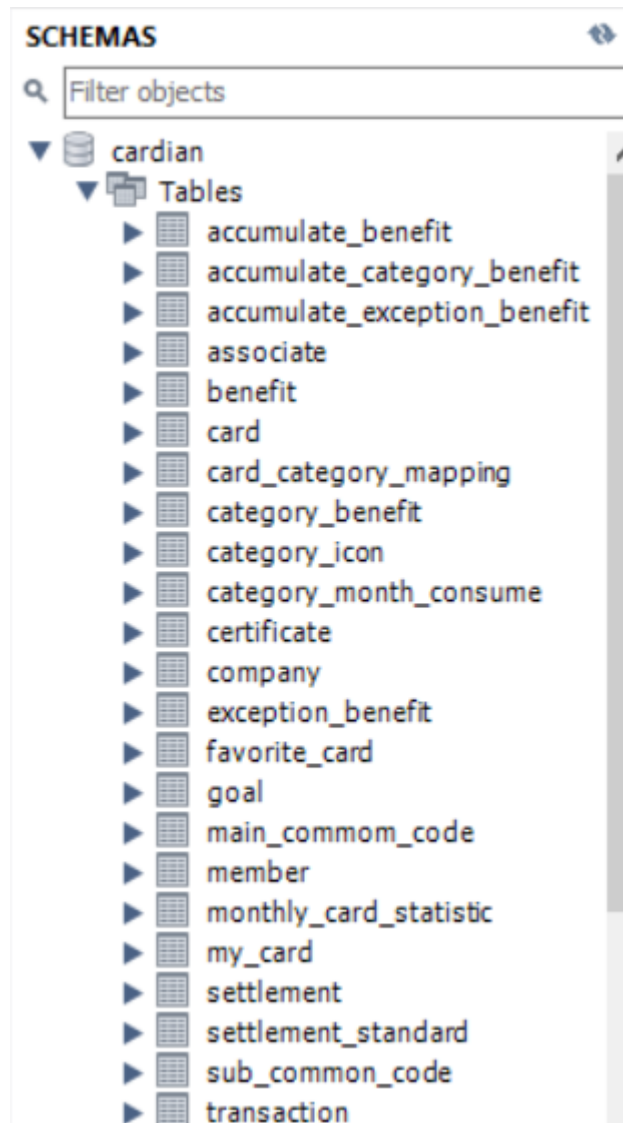
- server > Data Import > Import from Self-Contained File > cardian.sql 선택 후 우측 하단 Start Import 클릭



- import 성공시 아래와 같은 화면



- SCHEMAS 오른쪽의 새로고침 표시 클릭 후 테이블 목록이 잘 들어갔는 지 확인




2. AiMaker BE

2.1 Git Clone

- Git Lab Repository 주소 : <https://lab.ssafy.com/s10-ai-image-sub2/S10P22A302>

1. 레포지토리 주소 복사



Auto DevOps
It will automatically build, test, and deploy your application ba
Learn more in the [Auto DevOps documentation](#)
[Enable in settings](#)

S10P22A302

1 Commit 13 Branches 0 Tags 2.6 MiB Project Storage

Update BackEnd/ReadMe.md, FrontEnd/ReadMe.md
정여민 authored 4 weeks ago

master S10P22A302 / +

History Find file Edit **Code**

[Add README](#)
[Add LICENSE](#)
[Add CHANGELOG](#)
[Add CONTRIBUTING](#)
[Add Kubernetes cluster](#)

[Set up CI/CD](#)
[Add Wiki](#)
[Configure Integrations](#)

Name	Last commit	Last update
BackEnd	Update BackEnd/ReadMe.md, FrontEn...	4 weeks ago
FrontEnd	Update BackEnd/ReadMe.md, FrontEn...	4 weeks ago

Clone with HTTPS
`https://lab.ssafy.com/s10-ai-1m`

Open in your IDE
Visual Studio Code (HTTPS)
IntelliJ IDEA (HTTPS)

Download source code
zip
tar.gz
tar.bz2
tar

2. Git Bash

- clone 할 디렉토리에서 git bash 실행
- git clone

```
git clone https://lab.ssafy.com/s10-ai-image-sub2/S10P22A302
```

3. git bash 에서 clone 받은 프로젝트 디렉토리로 이동

```
cd S10P22A302/
```

4. AiMaker 자료 다운로드

- be branch로 switch

```
git switch -c be
```

- 원격 저장소의 데이터 pull

```
git pull origin be
```

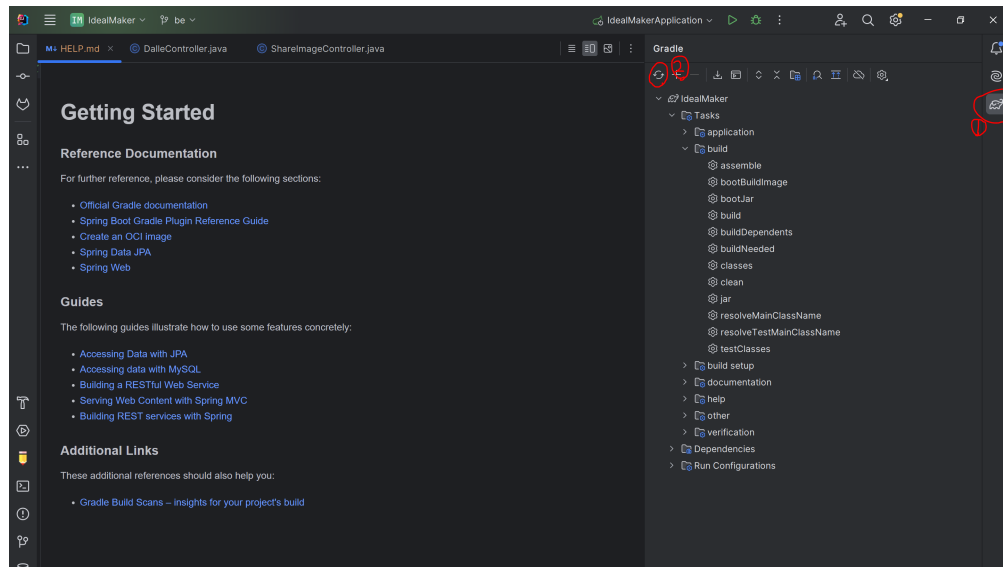
2.2 프로젝트 실행

2.2.1 IntelliJ에서 프로젝트 열기

- S10P22A302 > BackEnd > IdealMaker 디렉토리 열기

2.2.2 gradle 설정

- gradle 클릭 → Reload All Gradle Project 클릭



2.2.3 Dockerfile

- 파일 위치 : 프로젝트 최상단 폴더
 - Backend > IdealMaker
- 파일 내용

```
FROM docker
COPY --from=docker/buildx-bin:latest /buildx /usr/libexec/

FROM openjdk:17-jdk
ADD ./build/libs/IdealMaker-0.0.1-SNAPSHOT.jar app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

2.2.4 .gitignore 파일 추가

- 파일 위치 : 프로젝트 최상단 폴더
 - Backend > IdealMaker
- 파일 내용

```

HELP.md
.gradle
build/
!gradle/wrapper/gradle-wrapper.jar
!**/src/main/**/build/
!**/src/test/**/build/

### STS ###
.appt_generated
.classpath
.factorypath
.project
.settings
.springBeans
.sts4-cache
bin/
!**/src/main/**/bin/
!**/src/test/**/bin/

### IntelliJ IDEA ###
.idea
*.iws
*.iml
*.ipr
out/
!**/src/main/**/out/
!**/src/test/**/out/

### NetBeans ###
/nbproject/private/
/nbbuild/
/dist/
/nbdist/
/.nb-gradle/

```

```
### VS Code ###  
.vscode/  
  
### yaml ###  
**/application-*.yaml
```

2.2.5 프로퍼티 파일

- 파일 위치
 - Backend > IdealMaker > src > main > resources
- 파일명 : application.yaml
- 파일 내용

```
spring:  
  application:  
    name: idealmaker  
  profiles:  
    active:  
      - local  
      - common  
  group:  
    local:  
      - db-local  
    prod:  
      - db-prod  
  include:  
    - db  
    - cloud  
    - secret  
  
---  
spring:  
  config:  
    activate:  
      on-profile: local
```

```
logging:
  level:
    com: debug
    org.hibernate.SQL: debug
    org.hibernate.type: trace

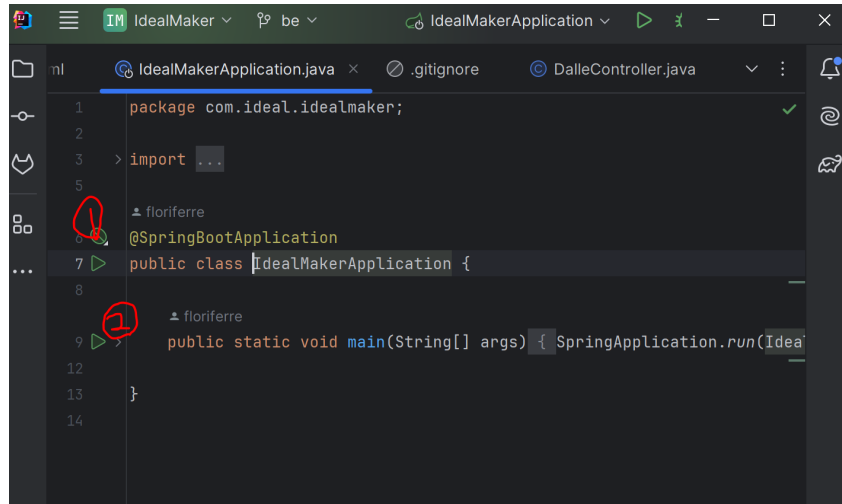
---
spring:
  config:
    activate:
      on-profile: prod

logging:
  level:
    com: info

---
```

2.2.6 프로젝트 실행

- 실행 파일 위치
 - Backend > IdealMaker> src > main > java > com > ideal > idealMaker > IdealMakerApplication.java
 - 해당 파일 실행
 - 1과 2 중 상관 없음



- 해당 메시지가 뜨면 실행 성공


2024-04-01T11:24:08.883+09:00 INFO 18768 --- [ideal

2024-02-14T15:42:37.518+09:00 INFO 20296 --- [main] c.o.cardcompany.CardcompanyApplication : Started CardcompanyApplication in 3.476 seconds

3.1 Git Clone

- Git Lab Repository 주소 : <https://lab.ssafy.com/s10-ai-image-sub2/S10P22A302>

1. 레포지토리 주소 복사




Auto DevOps


It will automatically build, test, and deploy your application ba

Learn more in the [Auto DevOps documentation](#)

[Enable in settings](#)

S10P22A302 

1 Commit 13 Branches 0 Tags 2.6 MiB Project Storage

 Update BackEnd/ReadMe.md, FrontEnd/ReadMe.md
정여민 authored 4 weeks ago

master S10P22A302 / +


History Find file Edit [Code](#)

[Add README](#)
[Add LICENSE](#)
[Add CHANGELOG](#)
[Add CONTRIBUTING](#)
[Add Kubernetes cluster](#)

[Set up CI/CD](#)
[Add Wiki](#)
[Configure Integrations](#)

Name	Last commit	Last update
BackEnd	Update BackEnd/ReadMe.md, FrontEn...	4 weeks ago
FrontEnd	Update BackEnd/ReadMe.md, FrontEn...	4 weeks ago

Clone with HTTPS

<https://lab.ssfy.com/s10-ai-im> 

Open in your IDE

Visual Studio Code (HTTPS)

IntelliJ IDEA (HTTPS)

Download source code

[zip](#)
[tar.gz](#)
[tar.bz2](#)
[tar](#)

2. Git Bash

- clone 할 디렉토리에서 git bash 실행
- git clone

```
git clone https://lab.ssfy.com/s10-ai-image-sub2/S1
```

3. git bash 에서 clone 받은 프로젝트 디렉토리로 이동

```
cd S10P22A302/
```

4. AiMaker 자료 다운로드

- fe branch로 switch

```
git switch -c fe
```

- 원격 저장소의 데이터 pull

```
git pull origin fe
```

3. AiMaker FE

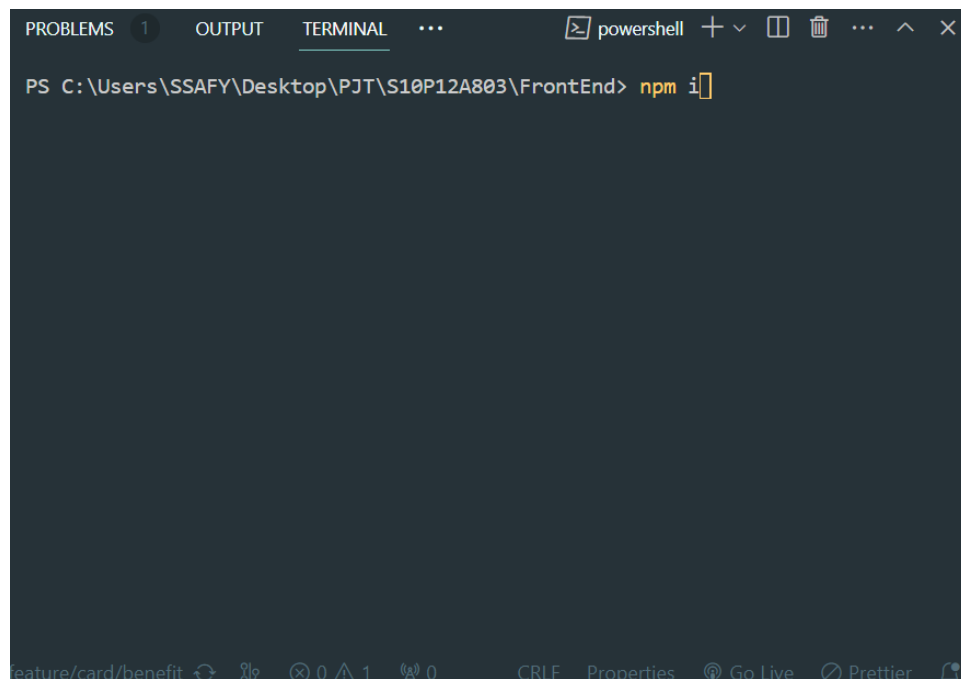
3.2 프로젝트 실행

3.2.1 VSCode에서 프로젝트 열기

- S10P22A802 > FrontEnd 디렉토리 열기

3.2.2 dependency 설치

- Ctrl+~ 누른 후, command 창 열기
npm i 엔터



The image shows a screenshot of a Visual Studio Code terminal window. The terminal is titled 'powershell' and shows the command prompt 'PS C:\Users\SSAFY\Desktop\PJT\S10P12A803\FrontEnd>'. The command 'npm i' has been entered, and the cursor is at the end of the command. The terminal window is dark-themed, and the command prompt is in a light color. The terminal is open in the VS Code editor, and the command prompt is in the PowerShell shell.

성공시

```
PROBLEMS 1 OUTPUT TERMINAL ... powershell + v [ ] [ ] ... ^ x

PS C:\Users\SSAFY\Desktop\PJT\S10P12A803\FrontEnd> npm i

up to date, audited 850 packages in 2s

227 packages are looking for funding
  run `npm fund` for details

2 vulnerabilities (1 moderate, 1 high)

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
PS C:\Users\SSAFY\Desktop\PJT\S10P12A803\FrontEnd> [ ]
```

3.2.3 Dockerfile 파일 추가

- 파일 위치 : 프로젝트 최상단 폴더
 - FrontEnd
- 파일 내용

```
# nginx 이미지 사용
FROM nginx:latest

# root에 /app 폴더 생성
RUN mkdir /app

# work dir 고정
WORKDIR /app

# work dir에 build 폴더 생성
RUN mkdir ./build

# host pc의 현재 경로의 build 폴더를 work dir의 build 폴더로 복사
ADD ./dist ./build

# nginx의 default.conf 삭제
RUN rm /etc/nginx/conf.d/default.conf
```

```
# host pc의 nginx.conf를 아래 경로에 복사
COPY ./nginx.conf /etc/nginx/conf.d

# 3000 포트 개방
EXPOSE 3000

# container 실행 시 자동으로 실행할 command. nginx 시작함
CMD ["nginx", "-g", "daemon off;"]
```

3.2.4 .gitignore 파일 추가

- 파일 위치 : 프로젝트 최상단 폴더
 - FrontEnd
- 파일 내용

```
# Logs
logs
*.log
npm-debug.log*
yarn-debug.log*
yarn-error.log*
pnpm-debug.log*
lerna-debug.log*

node_modules
#dist
dist-ssr
*.local

# Editor directories and files
.vscode/*
!.vscode/extensions.json
.idea
.DS_Store
*.suo
*.ntvs*
*.njsproj
```

```
*.sln
*.sw?
```

3.2.5 .env 파일 추가

- 파일 위치 : 프로젝트 최상단 폴더
 - FrontEnd
- 파일 내용

```
VITE_API_KEY = https://j10a302.p.ssafy.io/api
```

4. AWS 배포

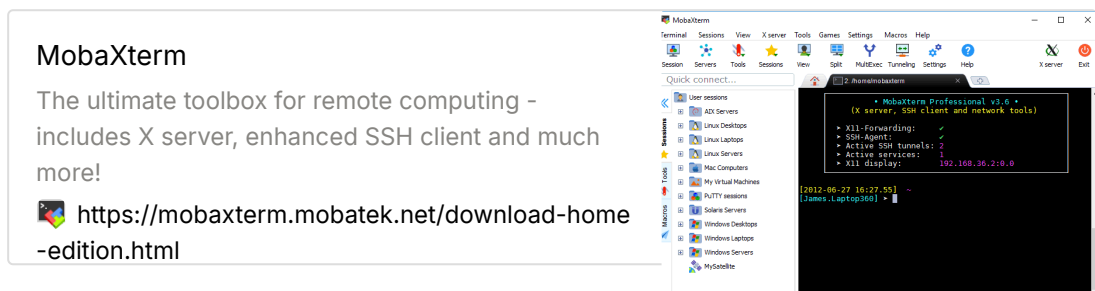
4.1 사용하는 요금제

- AWS EC2
- S3

4.2 서버 접속 방법 (SSH)

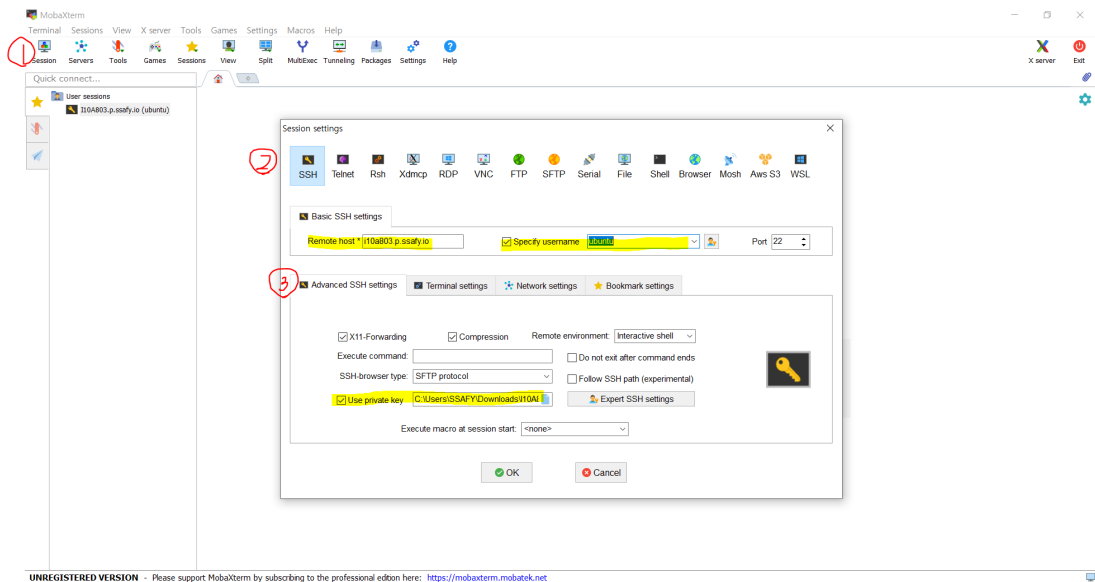
4.2.1 MobaXterm 사용

- 설치
 - 아래 링크에서 Home Edition (Installer edition) 다운 후 압축 해제



- 연결 설정

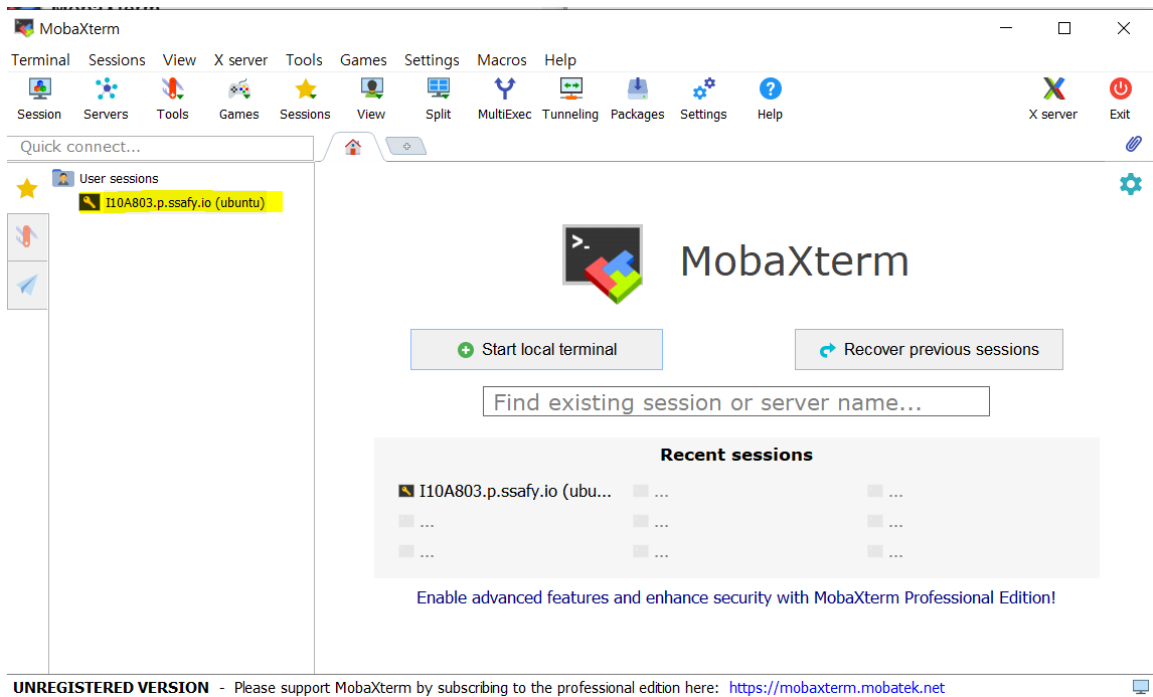
- Session > SSH > Advanced SSH settings 클릭
 - Remote host : j10a302.p.ssafy.io
 - Specify username 클릭 후 ubuntu 입력
 - port 22
 - use private key 클릭 후 pem 파일 등록



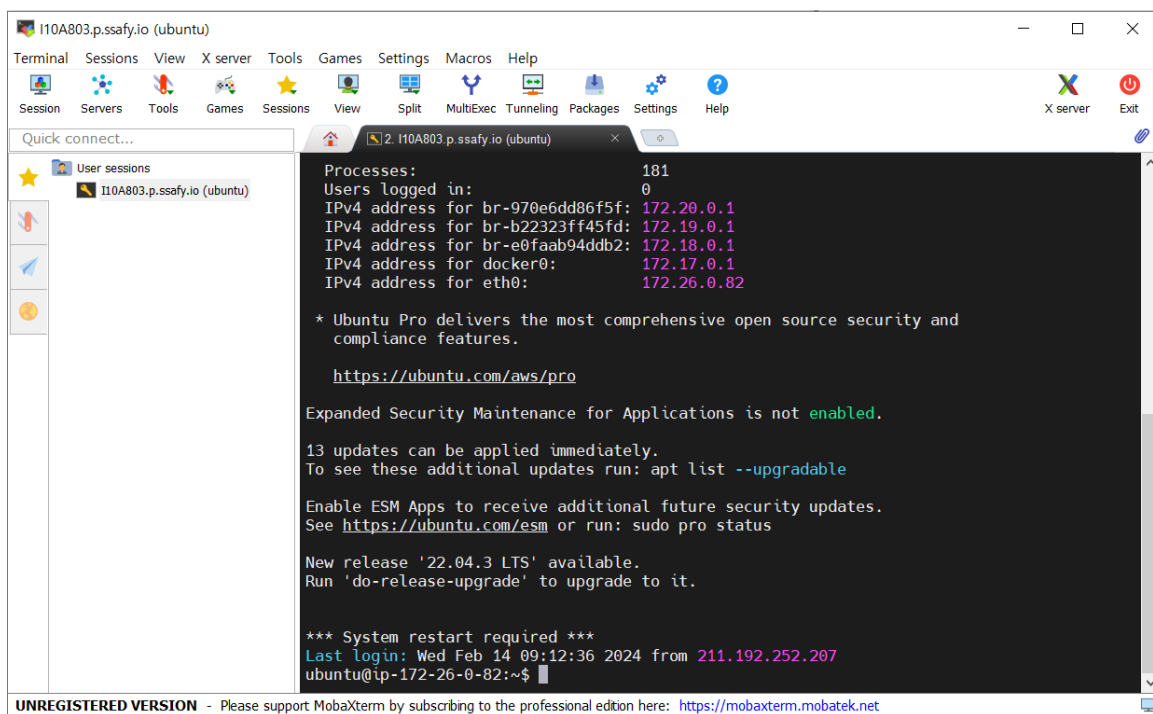
- 설정 완료 후 OK

4.2.2 우분투 서버 접속

- 왼쪽 별 > User sessions에서 j10a302.p.ssafy.io (ubuntu) 더블 클릭



- ubuntu 서버 접속 완료



4.3 데이터 베이스 Docker Container 설치

4.3.1 MySQL DB

- MySQL 이미지 받기

- docker pull 명령어를 통해 latest 버전의 mysql 이미지를 다운로드 한다.

```
sudo docker pull mysql:latest
```

- MySQL 컨테이너 실행

- **run** : 이미지를 가지고 컨테이너를 만들고 컨테이너 실행
- **d** : 컨테이너를 만들고 백그라운드에서 계속 구동하게 하는 옵션 (데몬)
- **p** : host:container 포트 연결 (3308)
- **e MYSQL_ROOT_PASSWORD=비밀번호** : 환경변수 설정 (root 계정에 대한 비밀번호 지정)
 - **반드시 '비밀번호' 대신 root에 사용할 임의의 비밀번호 지정 바람!**
- **v** : Host OS의 `/var/lib/mysql` 에 컨테이너에서 사용하는 `/var/lib/mysql` 의 볼륨 할당
 - 여기에서 -v /var/lib/mysql:/var/lib/mysql2로 해줄 것. (cardian 데이터베이스와 충돌을 막기 위함)
- **-name cardcompanydb** : 컨테이너 이름 지정 (미지정시 무작위로 이름 지정)
- **mysql** : dockerhub의 mysql 이미지 사용

```
docker run -d -p 3306:3306 -e MYSQL_ROOT_PASSWORD=비밀번호 -
```

- 컨테이너 실행 확인

- MobaXterm SSH 콘솔에 다음과 같이 입력

```
docker ps -a
```

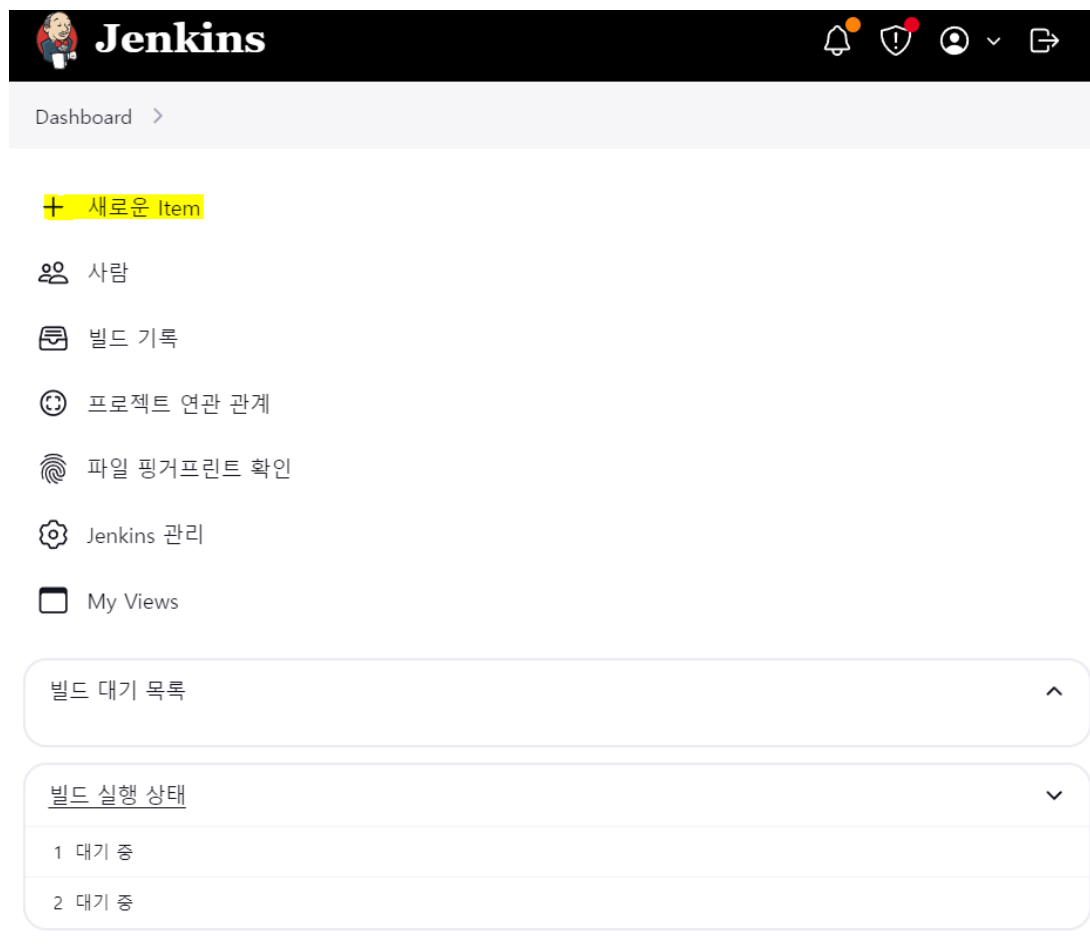
- 다음과 같이 뜨면 성공

```
223f28963c50 mysql "docker-entrypoint.s..." 2 weeks ago Up 7 days 3306/tcp, 0.0.0.0:3308->3306/tcp, :::3308->
```

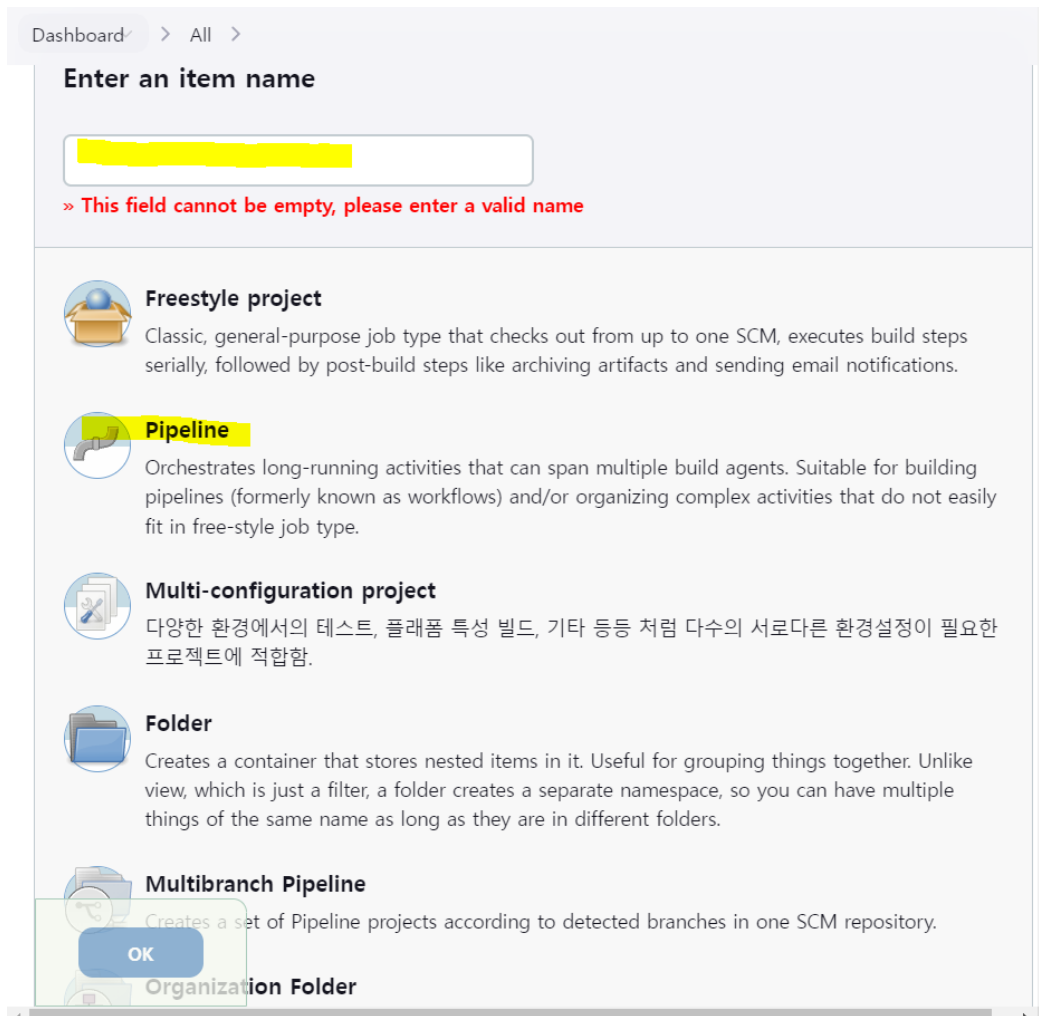
4.3.3 Jenkins (CI/CD)

- Jenkins 기본 설정은 다음을 참고

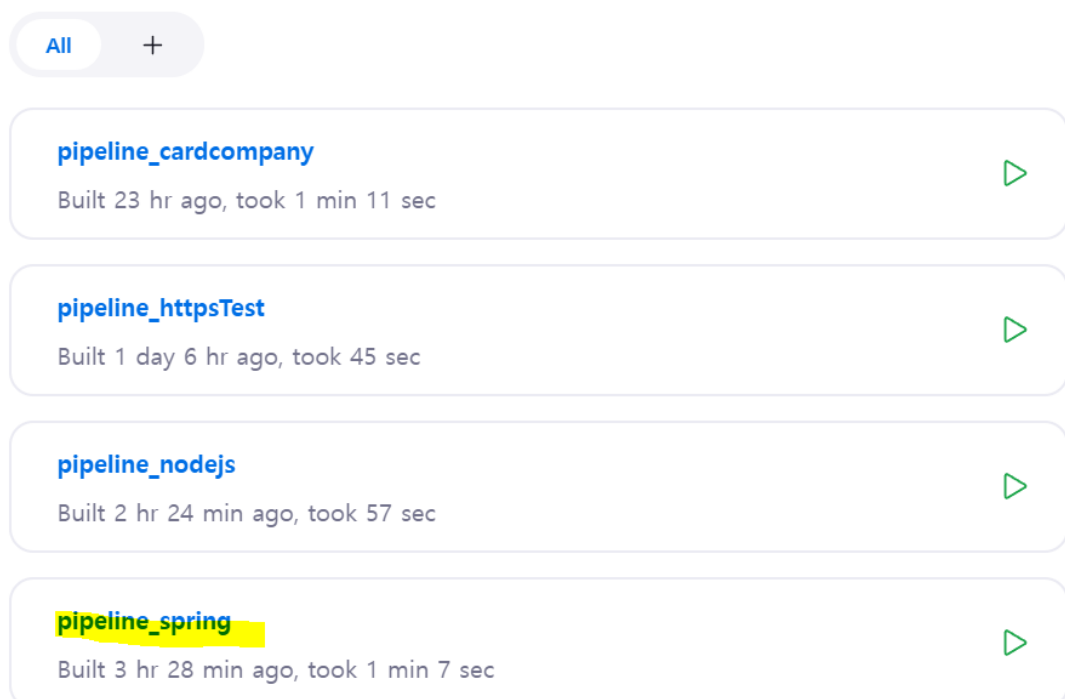
- Jenkins 접속 정보
 - 사용자 이름 : admin
 - 비밀번호 : a302
 - port : 8081
- 새로운 아이템 등록
 1. 새로운 item 클릭



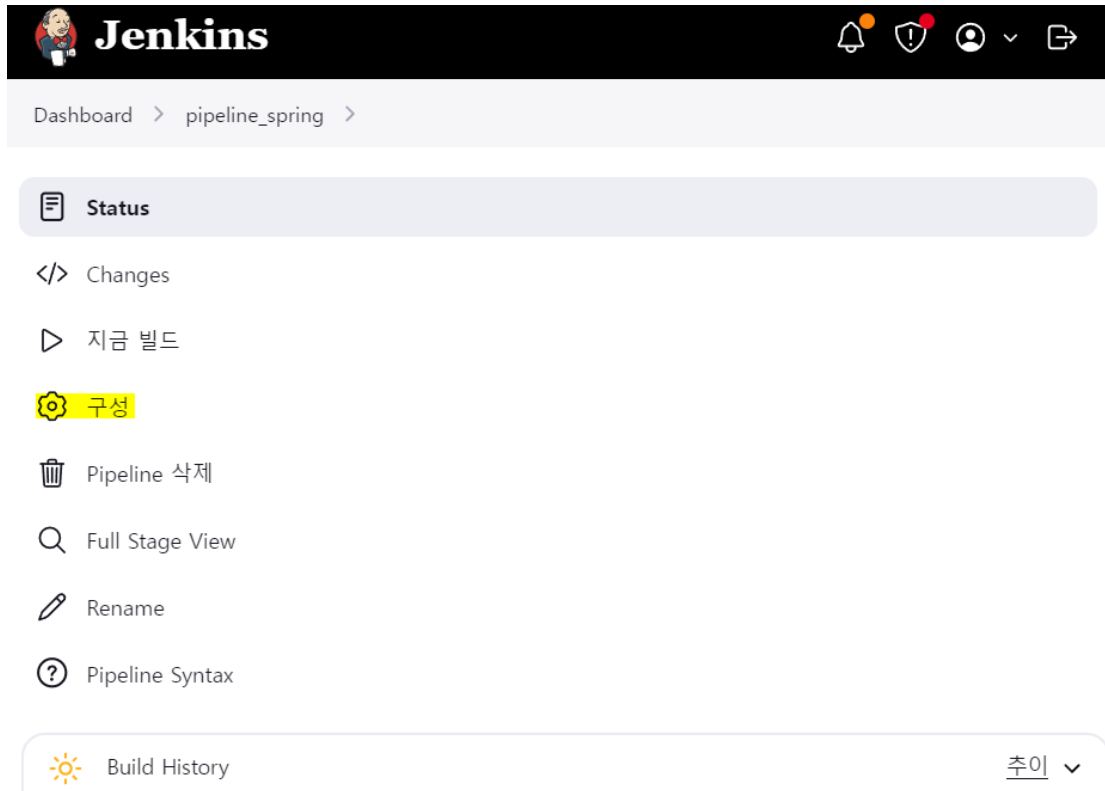
1. 아이템 이름을 입력하고, Pipeline 선택 후 OK



3. Dashboard > 생성한 item 이름 클릭



4. 구성 클릭



4-1. Build Triggers에서 **Build when a change is pushed to GitLab. GitLab webhook URL 선택**

- Push Events 활성화 : 연동한 gitlab에 push event가 발생할 때마다 젠킨스 파이프라인 실행
- 고급 토글 클릭 후 아래의 secret token의 generate 클릭
 - 해당 젠킨스 pipeline과 연동할 수 있는 토큰 생성



4-2. Pipeline 에서 script 작성

- IdealMaker BE Script

```
pipeline {
    agent any
    tools {nodejs "nodejs"}
```

```

environment {
    imageName = "sistina/a302-react"
    registryCredential = 'a302-docker'
    dockerImage = ''

    releaseServerAccount = 'ubuntu'
    releaseServerUri = 'j10a302.p.ssafy.io'
    releasePort = '3000'
}

stages {
    stage('Git Clone') {
        steps {
            git branch: 'fe',
                credentialsId: 'a302',
                url: 'https://lab.ssafy.com/s1
        }
    }
    stage('env add'){
        steps{
            dir("FrontEnd"){
                sh '''
                    touch .env
                    echo 'VITE_API_KEY = https://j
                '''
            }
        }
    }
    stage('Node Build') {
        steps {
            dir ('FrontEnd') {
                sh 'npm install'
                sh 'npm run build'
            }
        }
    }
}

```

```

stage('Image Build & DockerHub Push') {
    steps {
        dir('FrontEnd') {
            script {
                docker.withRegistry('', re
                sh "docker buildx crea
                sh "docker buildx buil
                sh "docker buildx buil
            }
        }
    }
}

stage('Before Service Stop') {
    steps {
        sshagent(credentials: ['a302-ubunt
        sh '''
        if test "`ssh -o StrictHostKey
        ssh -o StrictHostKeyChecking=n
        ssh -o StrictHostKeyChecking=n
        ssh -o StrictHostKeyChecking=n
        fi
        '''
    }
}

stage('DockerHub Pull') {
    steps {
        sshagent(credentials: ['a302-ubunt
        sh "ssh -o StrictHostKeyChecki
    }
}

stage('Service Start') {
    steps {
        sshagent(credentials: ['a302-ubunt
        sh '''
        ssh -o StrictHostKeyChecki

```

```

        '''
    }
}
}
stage('Service Check') {
    steps {
        sshagent(credentials: ['a302-ubuntu']) {
            sh '''
                #!/bin/bash

                for retry_count in \$(seq 1 20)
                do
                    if curl -s "http://j10a302-ubuntu"
                    then
                        curl -d '{"text":"[FE] OK"'
                        break
                    fi

                    if [ $retry_count -eq 20 ]
                    then
                        curl -d '{"text":"[FE] Error"'
                        exit 1
                    fi

                    echo "The server is not available"
                    sleep 5
                done
            '''
        }
    }
}
}
}
}

```

- IdealMaker FE Script

```

pipeline {
    agent any
}

```

```

tools {nodejs "nodejs"}

environment {
    imageName = "sistina/a803-nodejs"
    registryCredential = 'DockerHub-Yeomin'
    dockerImage = ''

    releaseServerAccount = 'ubuntu'
    releaseServerUri = 'i10a803.p.ssafy.io'
    releasePort = '80'
}

stages {
    stage('Git Clone') {
        steps {
            git branch: 'fe',
                credentialsId: 'YEOMIN',
                url: 'https://lab.ssafy.com/s1
        }
    }
    stage('env add'){
        steps{
            dir("FrontEnd"){
                sh '''
                    touch .env
                    echo 'VITE_BASE_URL = http://i
                '''
            }
        }
    }
    stage('Node Build') {
        steps {
            dir ('FrontEnd') {
                sh 'npm install'
                sh 'npm run build'
            }
        }
    }
}

```



```

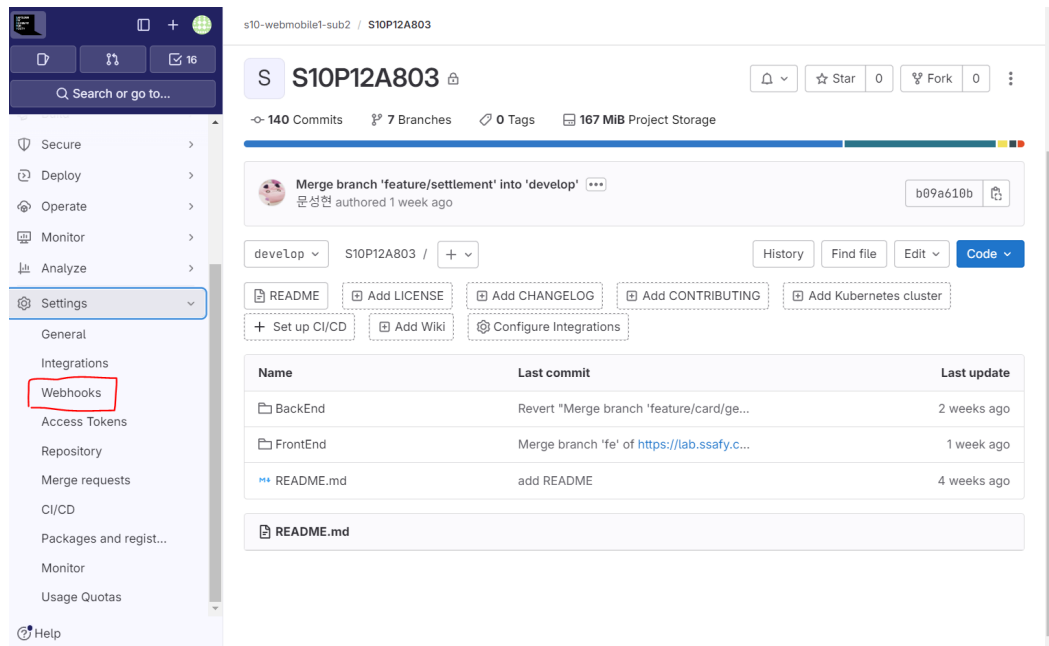
    }
    stage('Image Build & DockerHub Push') {
        steps {
            dir('FrontEnd') {
                script {
                    docker.withRegistry('', re
                        sh "docker buildx crea
                        sh "docker buildx buil
                        sh "docker buildx buil
                }
            }
        }
    }
    stage('Before Service Stop') {
        steps {
            sshagent(credentials: ['ubuntu-a80
                sh '''
                if test "`ssh -o StrictHostKey
                ssh -o StrictHostKeyChecking=n
                ssh -o StrictHostKeyChecking=n
                ssh -o StrictHostKeyChecking=n
                fi
                '''
        }
    }
    stage('DockerHub Pull') {
        steps {
            sshagent(credentials: ['ubuntu-a80
                sh "ssh -o StrictHostKeyChecki
            }
        }
    }
    stage('Service Start') {
        steps {
            sshagent(credentials: ['ubuntu-a80
                sh '''

```

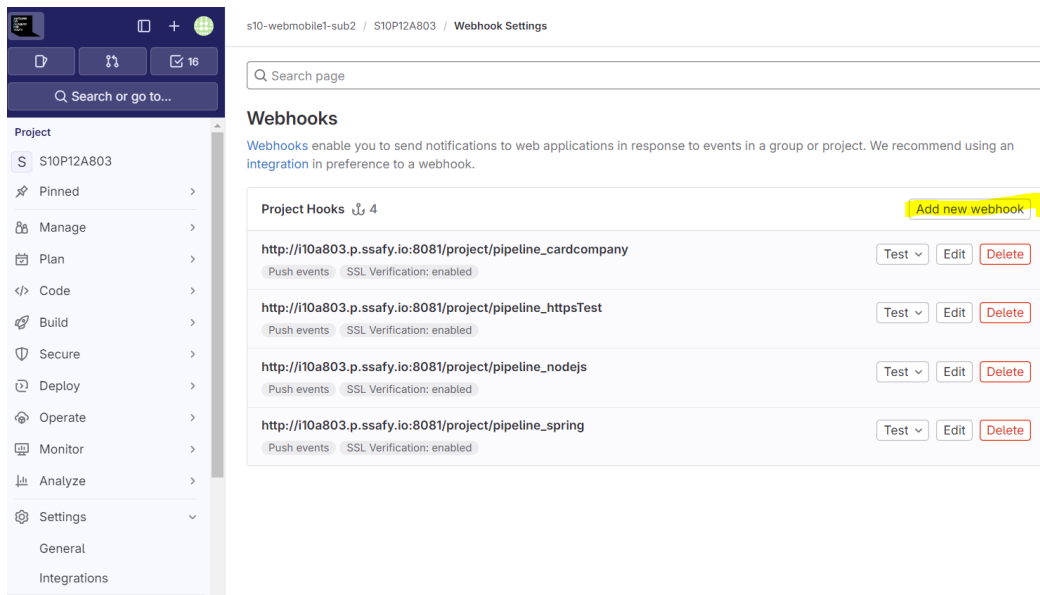

- 빌드한 도커 이미지
- registryCredential = 'a302-docker'
 - 젠킨스에 설정한 도커 허브 credential
- releaseServerAccount = 'ubuntu'
- releaseServerUri = 'j10a302.p.ssafy.io'
 - ubuntu 서버 주소
- releasePort = '3000'
 - 해당 서비스의 도커 컨테이너 포트 번호

5. gitlab Webhook 설정

- Repository 접속 후 Settings > Webhooks



- Add new Webhooks 클릭



- Webhook 설정

- url : 젠킨스 아이템 주소
- Secret token : 젠킨스 파이프라인 생성 시 발급 받은 토큰
- Trigger
 - push events
 - Regular Expression으로 company 브랜치의 push event만 반응하게

s10-webmobile1-sub2 / S10P12A803 / Webhook Settings / Webhook

Q 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

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

☒ Show full URL
☐ Mask portions of URL
 Do not show sensitive data such as tokens in the UI.

Secret token

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

Trigger

☒ Push events

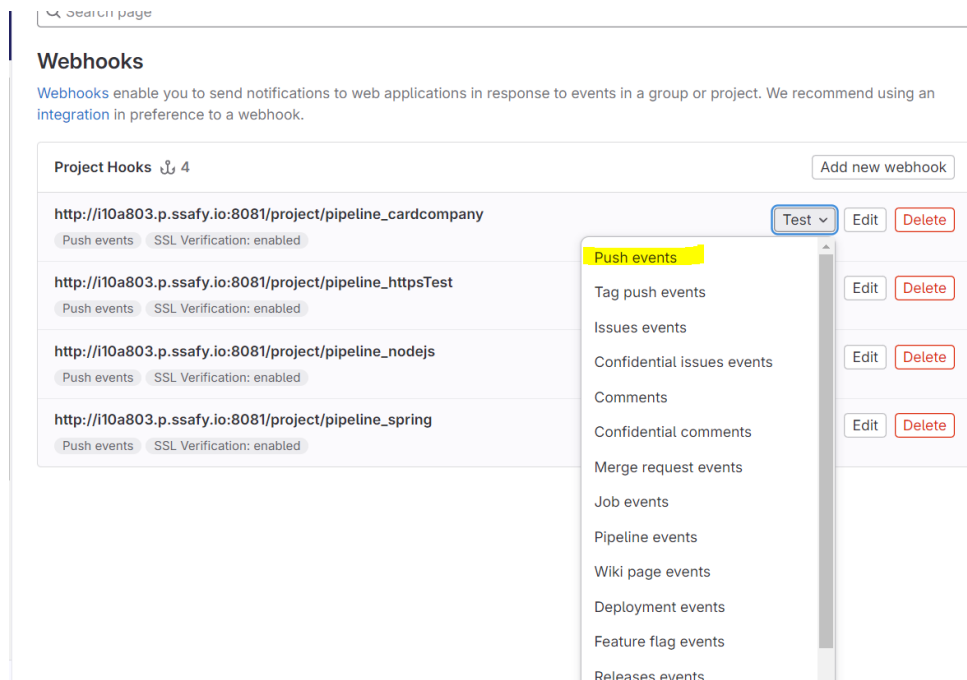
☐ All branches

☐ Wildcard pattern

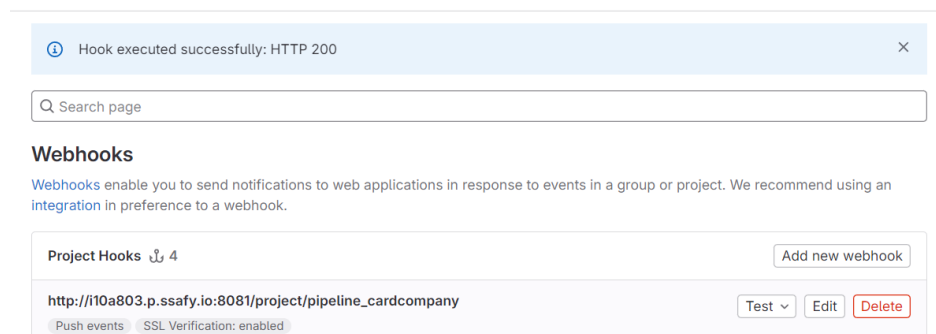
☒ Regular expression

Regular expressions such as `^(feature|hotfix)/` are supported.

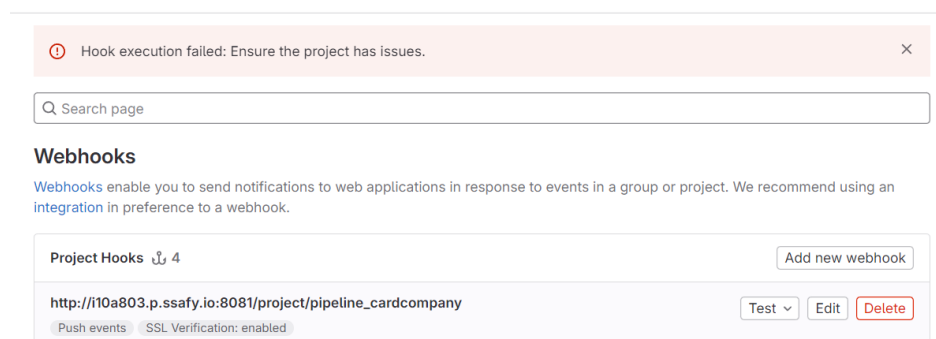
- Webhook 테스트
 - push event 클릭



■ 성공시 파란색 문구 표시



■ 실패시 빨간 문구 표시



6. Jenkins > 아이템 클릭 > 지금 빌드 클릭

- 빌드 성공 후 도커 컨테이너가 무사히 실행되면 아래와 같은 초록색 화면이 뜬다

Dashboard > pipeline_cardcompany >

Status **pipeline_cardcompany** CardCompany SpringBoot

Changes

지금 빌드

구성

Pipeline 식재

Full Stage View

Rename

Pipeline Syntax

내용 수정

프로젝트 중지하기

Stage View

Git Clone	Jar Build	Image Build & DockerHub Push	Before Service Stop	DockerHub Pull	Service Start	Service Check
2s	10s	21s	3s	5s	1s	3min 33s
945ms	4s	27s	6s	6s	1s	10s
1s	12s	25s	3s	5s	1s	27s

Average stage times: (Average full run time: ~1min 4s)

Build History

2월 04 일 23:08 1 commit

2024. 2. 14. 오후 5:51

Started by GitLab push by 정여민

- 배포 실패시 빨간 화면이 뜬다

#12

2월 04 일 23:08 1 commit

736ms

11s failed

58ms failed

55ms failed

57ms failed

61ms failed

58ms failed

- Build History를 선택하면 각 단계별 어디에서 에러가 났는지 콘솔 출력됨

#12

2024. 2. 4. 오후 11:08

Started by GitLab push by 정여민

Dashboard > pipeline_cardcompany > #12

Status **콘솔 출력**

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#12'

Polling Log

Git Build Data

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

Started by GitLab push by 정여민

[Pipeline] Start of Pipeline

[Pipeline] node

Running on Jenkins in /var/jenkins_home/workspace/pipeline_cardcompany

[Pipeline] {

[Pipeline] withEnv

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Git Clone) (show)

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (Jar Build)

[Pipeline] dir

Running in /var/jenkins_home/workspace/pipeline_cardcompany/BackEnd/cardcompany

[Pipeline] {

[Pipeline] sh

+ chmod +x ./gradlew

[Pipeline] sh

+ ./gradlew clean bootJar

Starting a Gradle Daemon (subsequent builds will be faster)

> Task :clean

> Task :compileJava

