

# 87

## 리마인드

### 가상화 / 컨테이너화 (virtualization / containerization)

- 서버 가상화  
서비스를 제공 front end 서비스 db 서비스
- 데스크톱 가상화  
워드 프로세스 파워포인트 업무용 서비스
- 컨테이너  
커널 x 가장 두드러진 특징  
커널을 안 올리기에 부팅이 빠르다  
정확히는 부팅이라는 것 자체가 없는 편 다운타임이 최소화

## 실습

### 1도커

- install\_docker.sh

```
#!/bin/bash

sudo systemctl stop firewalld
sudo systemctl disable firewalld
sudo curl -o /etc/yum.repos.d/docker-ce.repo https://download.docker.com/linux/cen
sudo yum install -y docker-ce

sudo systemctl start docker
sudo systemctl enable docker

# docker run --rm hello-world
```

- set\_dns.sh

```
#!/bin/bash

cp hosts /etc/hosts
yum install -y dnsmasq
systemctl start dnsmasq
systemctl enable dnsmasq
```

## 2 것

- create\_sert.sh

```
#!/bin/bash

mkdir /auth

openssl \
req -newkey rsa:4096 -nodes -sha256 -x509 \
-days 365 -keyout /auth/myregistry.com.key -out /auth/myregistry.com.crt \
-subj '/CN=myregistry.com' \
-addext "subjectAltName = DNS:myregistry.com"

mkdir -p /etc/docker/certs.d/myregistry.com
cp /auth/myregistry.com.crt /etc/docker/certs.d/myregistry.com/ca.crt
```

- docker-compose.yaml

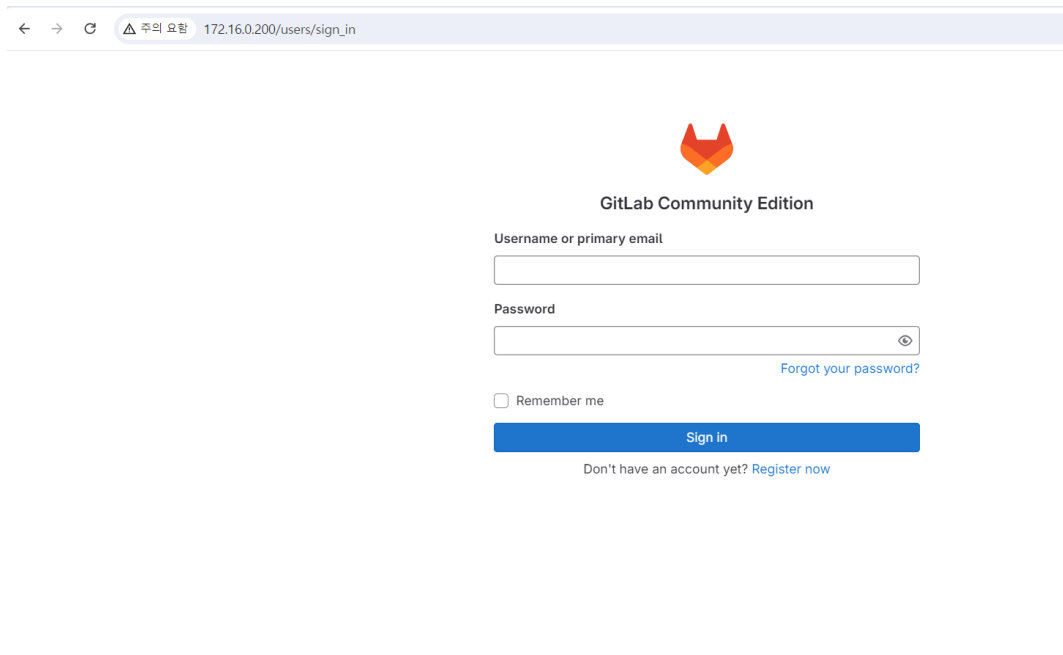
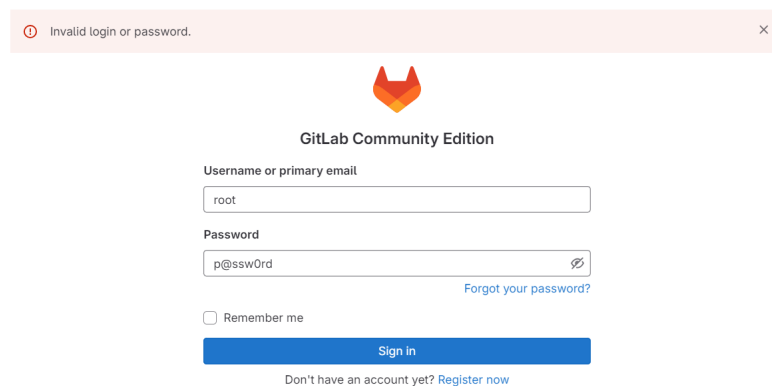
```
services:
  gitlab:
    image: 'quay.io/uvelyster/gitlab-ce:latest'
    restart: always
    hostname: 'mygitlab.com'
    container_name: gitlab
    dns: 172.16.0.200
    environment:
      GITLAB_ROOT_PASSWORD: P@ssw0rd
      GITLAB_OMNIBUS_CONFIG: |
        external_url 'http://mygitlab.com'
        registry_external_url 'https://myregistry.com'
    ports:
      - '80:80'
      - '443:443'
    volumes:
```

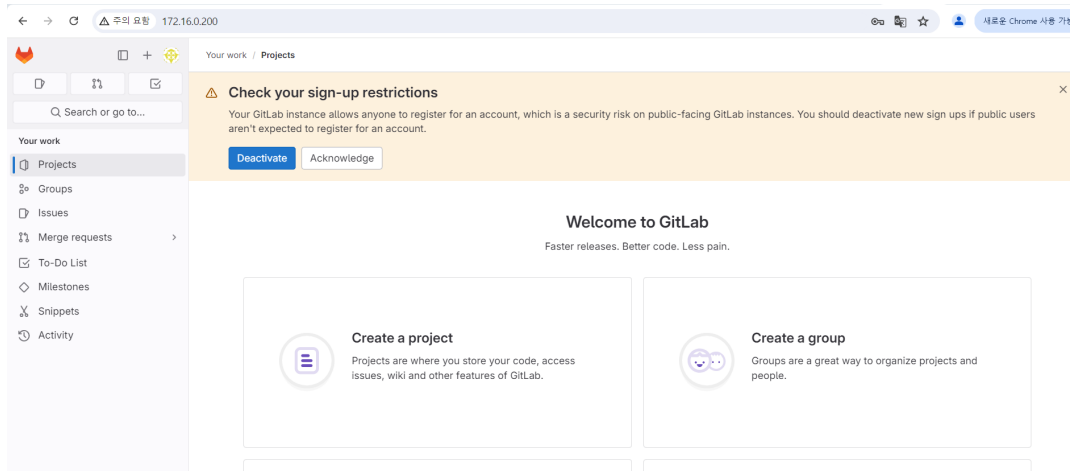
```

- '/root/gitlab/config:/etc/gitlab'
- '/auth:/etc/gitlab/ssl'
- '/root/gitlab/logs:/var/log/gitlab'
- '/root/gitlab/data:/var/opt/gitlab'
- '/root/gitlab/backup:/var/opt/gitlab/backups'
- '/root/gitlab/registry:/var/opt/gitlab/gitlab-rails/shared/registry'

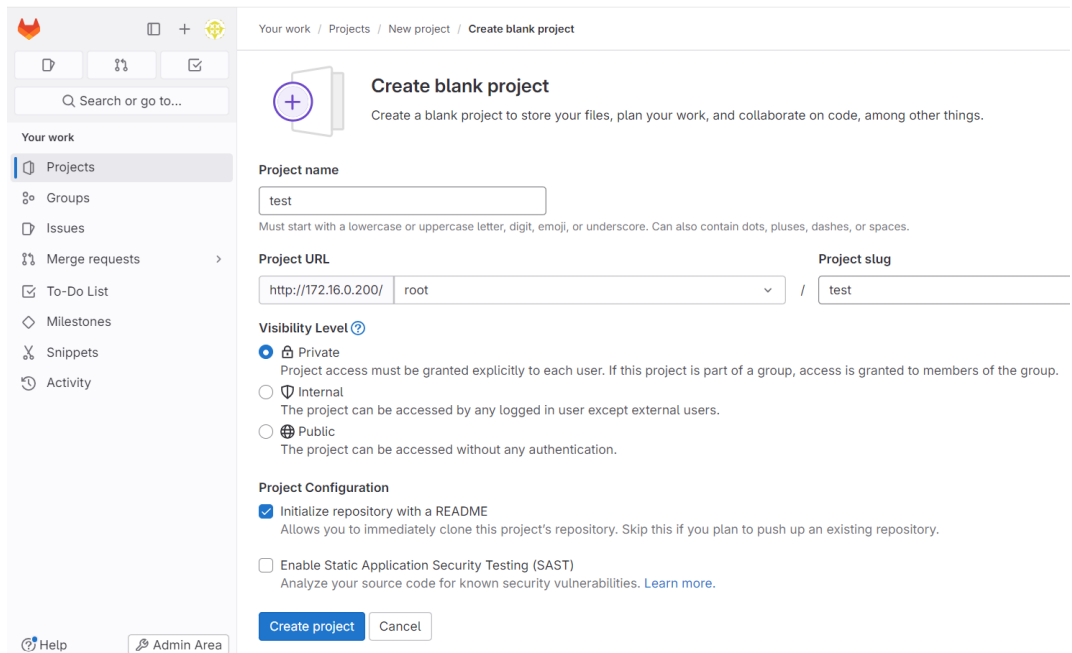
```

docker compose up -d로 컨테이너 올린다.



- 프로젝트 생성



- build 후 올리기

```
docker build -t myregistry.com/riit/test/myimage
```

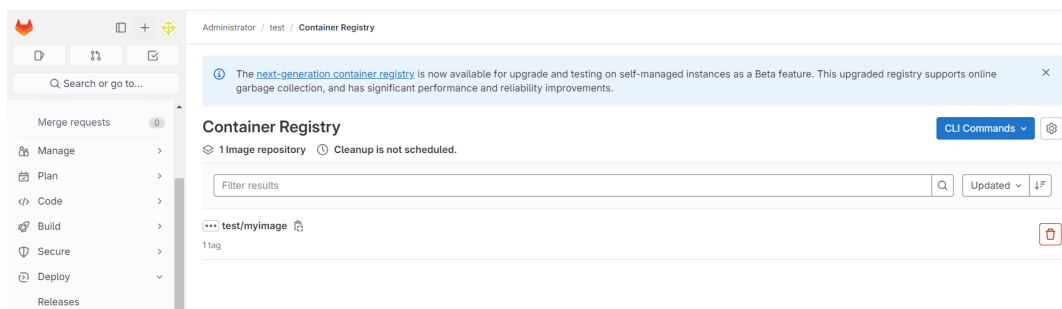
```
docker login myregistry.com
```

```
[root@controller nodejs]# docker login myregistry.com
Username: root
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

레지스트리에 올라가기까지의 과정이 ci이다.

```
quay.io/velystar/jenkins:latest 08d31204f1e3 2 months ago 47.0MB
[root@controller nodejs]# docker push myregistry.com/root/test/myimage
Using default tag: latest
The push refers to repository [myregistry.com/root/test/myimage]
5f70bf18a086: Pushed
292c82b01064: Pushed
7b15117ecec: Pushed
e4cfd543eb2d: Pushed
latest: digest: sha256:996caffc0bcc5f7dabe1e47cddb554e8bdf20f31ff584a89bf252a1956d9e1d1 size: 1157
[root@controller nodejs]#
```



## 젠킨스

- run\_jenkins.sh

```
#!/bin/bash

docker rm -f jenkins

docker run -d --name jenkins \
--restart always \
--dns 172.16.0.200 \
-p 8080:8080 \
-p 50000:50000 \
-v jenkins_home:/var/jenkins_home \
-v /var/run/docker.sock:/var/run/docker.sock \
quay.io/velystar/jenkins
#jenkins/jenkins:1ts
```

```
# Install Plugin
# docker cp plugins.txt jenkins:/var/jenkins_home/
# docker exec jenkins jenkins-plugin-cli -f /var/jenkins_home/plugins.txt

# Add Group and docker binary
# docker exec -u 0 jenkins groupadd -g 991 docker
# docker exec -u 0 jenkins usermod -aG 991 jenkins
# docker cp /usr/bin/docker jenkins:/usr/bin/docker

# docker restart jenkins
```

- docker logs jenkins

비밀번호 확인 후 로그인

```
*****
*****
*****

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

ddc59203285b4a2fb9d9a12cecc5b875

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

*****
*****
*****
```

- plugins.txt

```
git:latest
gitlab-branch-source:latest
github-branch-source:latest
workflow-multibranch:latest
build-timeout:latest
credentials-binding:latest
antisamy-markup-formatter:latest
cloudbees-folder:latest
timestamper:latest
ws-cleanup:latest
gradle:latest
junit:latest
workflow-aggregator:latest
pipeline-graph-view:latest
email-ext:latest
kubernetes-cli:latest
docker-plugin:latest
```

```
docker-workflow:latest
nodejs:latest
sonar:latest
```

- 플러그인 다운로드

```
docker cp plugins.txt jenkins:/var/jenkins_home/
docker exec jenkins jenkins-plugin-cli -f /var/jenkins_home/plugins.txt
```

**Jenkins** Search (CTRL+K) admin 로그아웃

Dashboard > Jenkins 관리 > Plugins

**Plugins** Search installed plugins

Updates 1 Available plugins Installed plugins Advanced settings

이름 ↓	사용가능
<b>Apache HttpComponents Client 4.x API Plugin</b> 4.5.14-208.v438351942757 Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins. <a href="#">Report an issue with this plugin</a>	<input checked="" type="checkbox"/> <input type="checkbox"/>
<b>Apache HttpComponents Client 5.x API Plugin</b> 5.3.1-110.v77252fb_d4da_5 Bundles <a href="#">Apache HttpComponents Client</a> and allows it to be used by Jenkins plugins. <a href="#">Report an issue with this plugin</a>	<input checked="" type="checkbox"/> <input type="checkbox"/>
<b>ASM API Plugin</b> 9.7-33.v4d23ef79fcc8 This plugin provides the <a href="#">ASM APIs</a> (v9.7) for other plugins. <a href="#">Report an issue with this plugin</a>	<input checked="" type="checkbox"/> <input type="checkbox"/>
<b>Authentication Tokens API Plugin</b> 1.119.v50285141b_7e1 This plugin provides an API for converting credentials into authentication tokens in Jenkins. <a href="#">Report an issue with this plugin</a>	<input checked="" type="checkbox"/> <input type="checkbox"/>
<b>Bootstrap 5 API Plugin</b> 5.3.3-1 Provides <a href="#">Bootstrap 5</a> for Jenkins Plugins. Bootstrap is (according to their self-perception) the world's most popular front-end component library to build responsive, mobile-first projects on the web. <a href="#">Report an issue with this plugin</a>	<input checked="" type="checkbox"/> <input type="checkbox"/>

- jenkins credential

**Jenkins** Search (CTRL+K) admin 로그아웃

Dashboard > Jenkins 관리 > Credentials > System > Global credentials (unrestricted) >

**Global credentials (unrestricted)** + Add Credentials

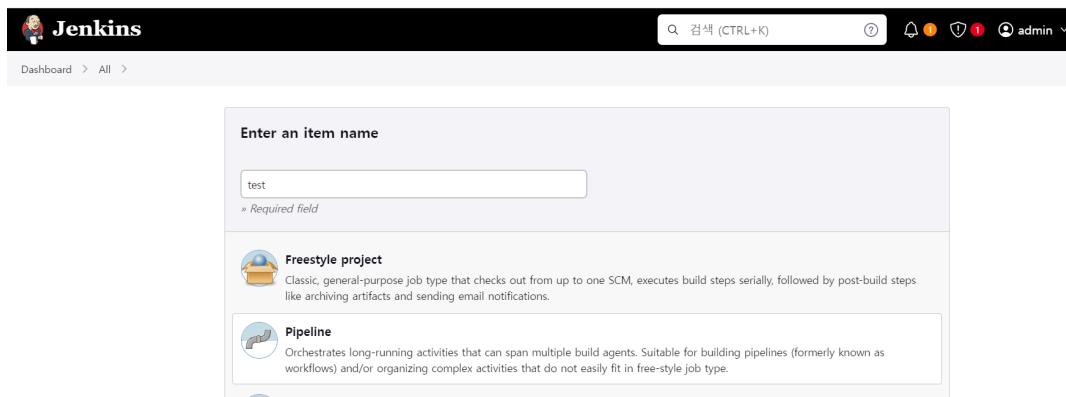
Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
This credential domain is empty. How about <a href="#">adding some credentials</a> ?			

아이콘: S M L

크리덴셜을 통해 깃 접근 가능

- jenkins 파이프라인 생성



jenkins script는 dsl(domain specific language) 중 에서 groovy를 사용한다.

- 스크립트 추가

```
pipeline {
  agent any

  tools {
    // Install the Maven version configured as "M3" and add it to the path.
    maven "M3"
  }

  stages {
    stage('Build') {
      steps {
        // Get some code from a GitHub repository
        git 'https://github.com/jglick/simple-maven-project-with-tests.git'

        // Run Maven on a Unix agent.
        sh "mvn -Dmaven.test.failure.ignore=true clean package"

        // To run Maven on a Windows agent, use
        // bat "mvn -Dmaven.test.failure.ignore=true clean package"
      }
    }

    post {
      // If Maven was able to run the tests, even if some of the test
      // failed, record the test results and archive the jar file.
      success {
        junit '**/target/surefire-reports/TEST-*.xml'
        archiveArtifacts 'target/*.jar'
      }
    }
  }
}
```



```
}  
}  
}
```

## • 도구 추가

Dashboard > Jenkins 관리 > Tools

### Maven installations

Maven installations ^ Edited

Add Maven

Maven

Name

M3

☒ Install automatically ?

Install from Apache

Version

3.9.8

Add Installer ^

이름을 script와 동일하게 매핑

## • 빌드 진행

### ! test

상세 내용 입력

프로젝트 중지하기

Last Successful Artifacts

simple-maven-project-with-tests-1.0-SNAPSHOT.jar

2.03 KiB

view

가장 최근 테스트 결과 (1 failure / -2)

고정링크

- Last build, (#3), 2 min 8 sec 전
- Last successful build, (#3), 2 min 8 sec 전
- Last failed build, (#1), 7 min 28 sec 전
- Last unstable build, (#3), 2 min 8 sec 전
- Last unsuccessful build, (#3), 2 min 8 sec 전
- Last completed build, (#3), 2 min 8 sec 전

테스트 결과 현황

Passed

Skipped

Failed

7

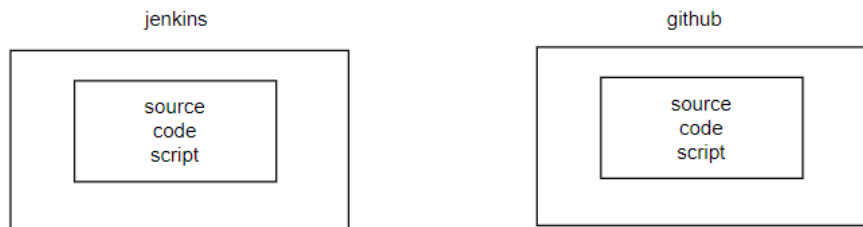
6

5

4

#2

#3



```

943 cp -r nodejs/ /root/
944 cp 04_harbor/Jenkinsfile /root/nodejs/
945 cd /root/nodejs/
946 git status
947 git init
948 git remote add origin http://mygitlab.com/root/hello.git
949 git remote -v
950 git status
951 git add .
952 git commit -m "init"
953 git push -u origin master

```

The screenshot shows the Jenkins web interface for a project named 'hello'. The left sidebar contains navigation links: Project, Pinned, Issues, Merge requests, Manage, Plan, Code, Build, Secure, Deploy, Operate, Monitor, Analyze, and Settings. The main content area displays the 'hello' project details, including a commit history table and project information.

Name	Last commit	Last update
Dockerfile	init	2 minutes ago
Jenkinsfile	init	2 minutes ago
package.json	init	2 minutes ago
server.js	init	2 minutes ago

Project information:

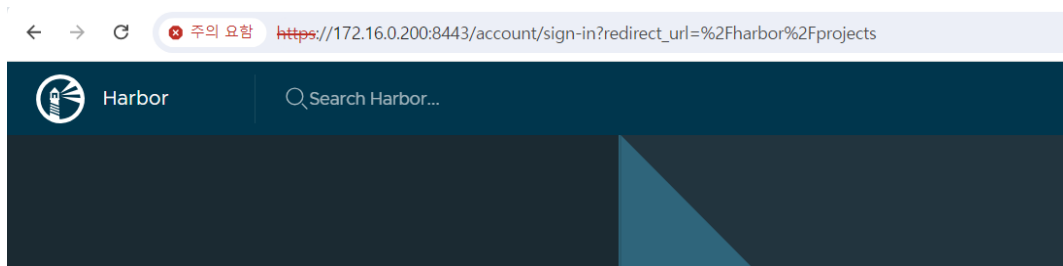
- 1 Commit
- 1 Branch
- 0 Tags
- 1 KIB Project Storage
- Auto DevOps enabled
- + Add README
- + Add LICENSE
- + Add CHANGELOG
- + Add CONTRIBUTING
- + Add Kubernetes cluster
- + Add Wiki
- + Configure Integrations



깃허브와 달리 깃랩은 레포지토리 안만들고 push 날려도 만들어진다.

- docker-pipeline
- docker tools 구성 도커 바이너리 복사
- script - Jenkinsfile

## harbor



- 로그인

admin / Harbor12345

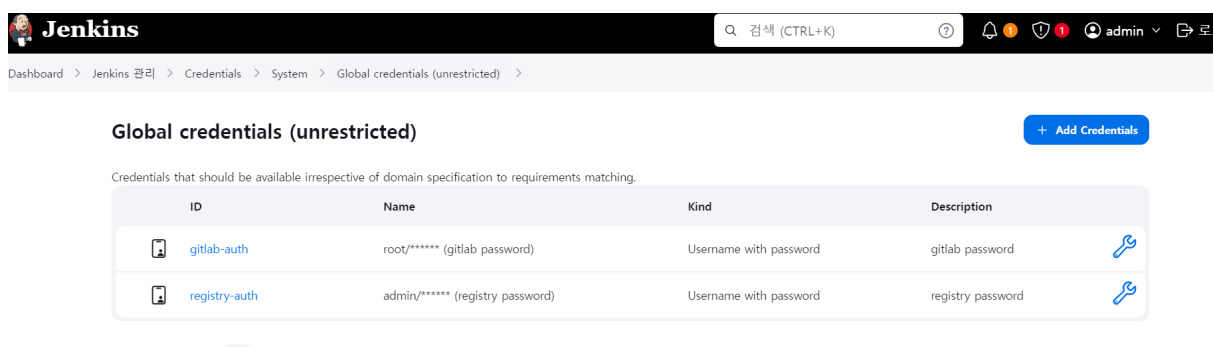
- jenkins credential

registry-auth

admin / Harbor 12345

gitlab

admin / P@ssw0rd



- 최종구조

