DevOps를 위한 CI/CD Pipeline 구축

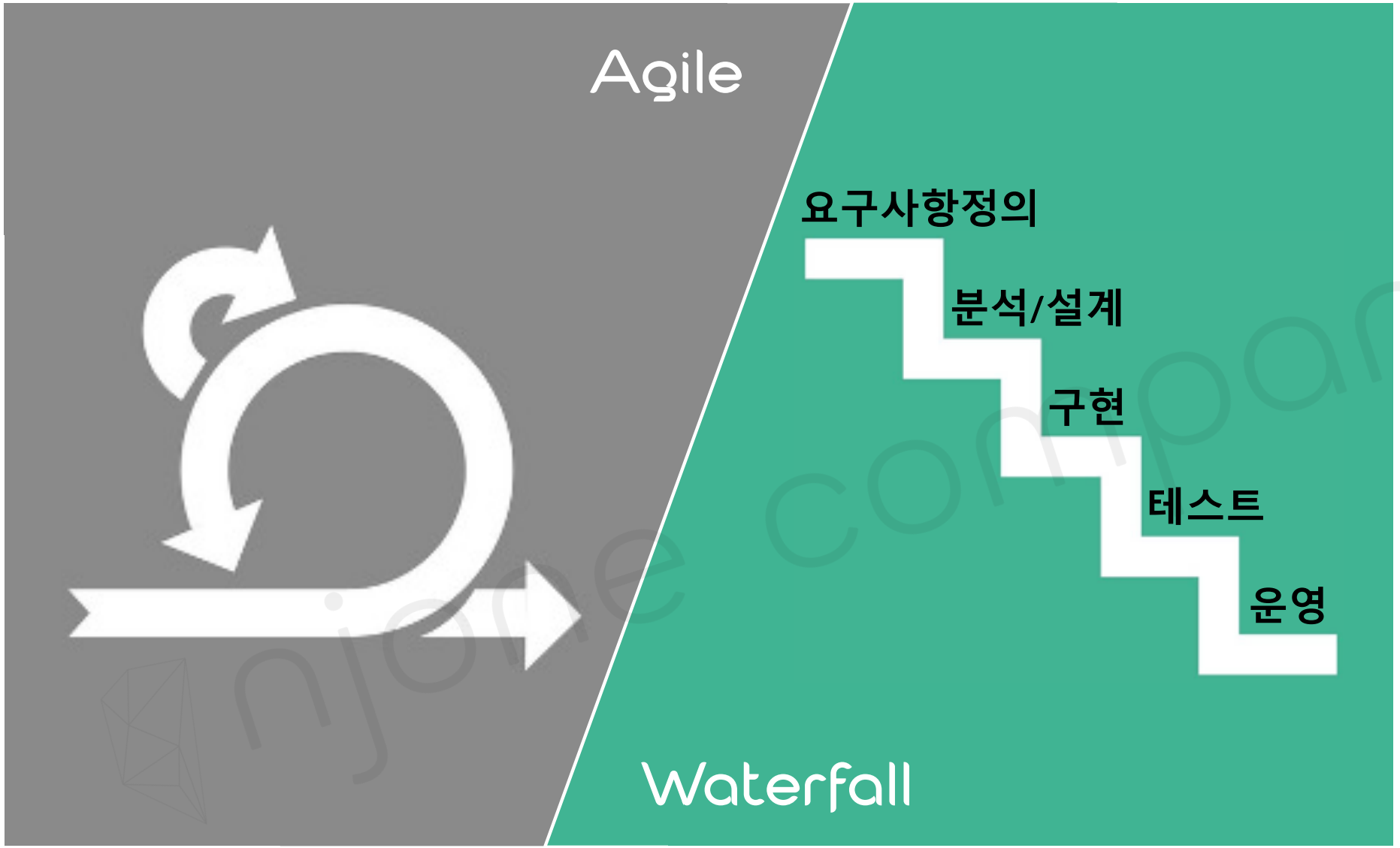
목차

Section 1.

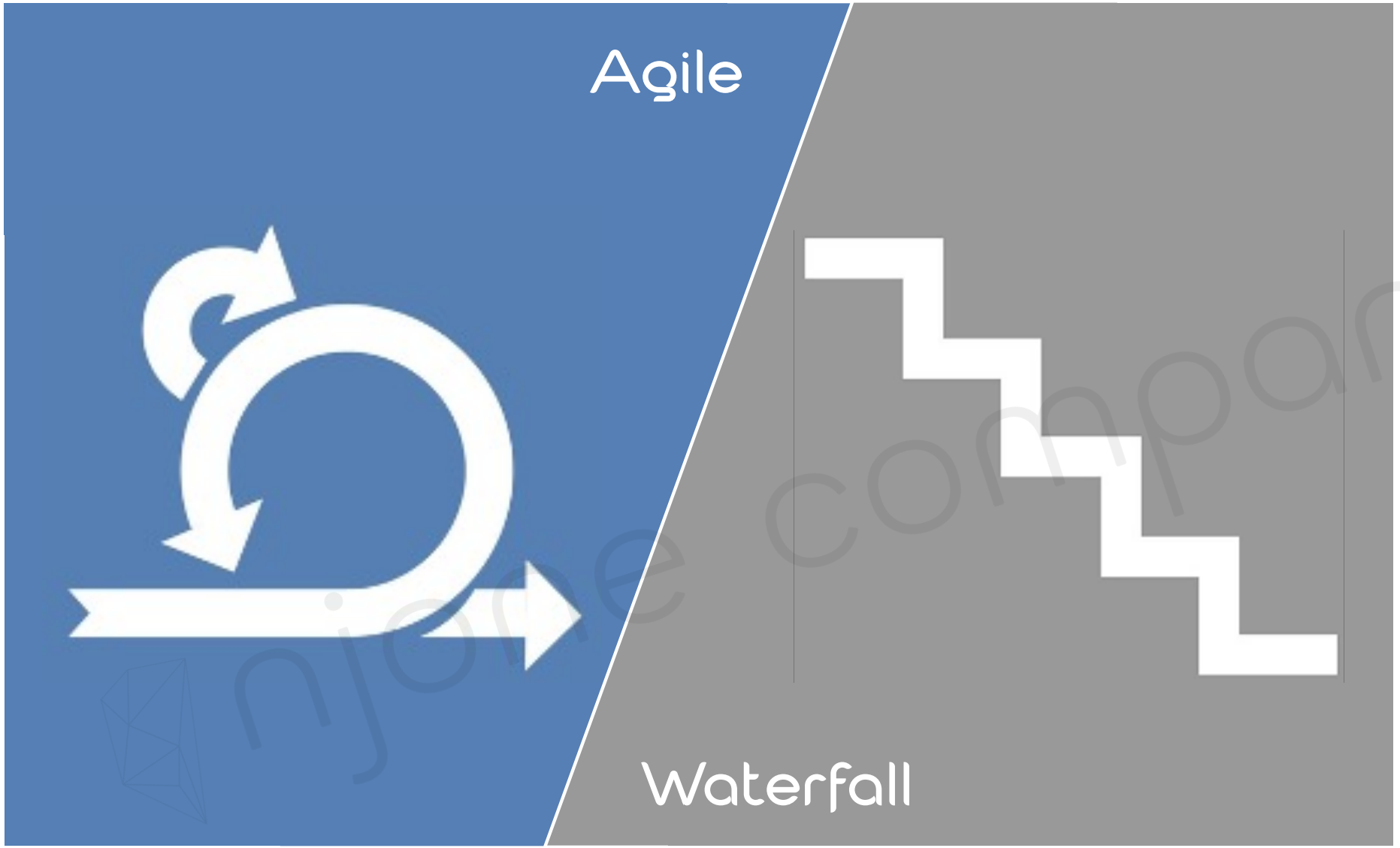
DevOps와 CI/CD

- DevOps와 CI/CD
- Cloud Native Application
- CI/CD 작업 흐름
- Jenkins

Waterfall vs Agile



Waterfall vs Agile



Waterfall vs Agile

- ***Manifesto for Agile Software Development***

> “We are uncovering better ways of ***developing software*** by doing it and ***helping others*** do it.
> Through this work we have come to value:
>
> ***Individuals and interactions*** over processes and tools
> ***Working software*** over comprehensive documentation
> ***Customer collaboration*** over contract negotiation
> ***Responding to change*** over following a plan
>
> That is, while there is value in the items on the right, we value the items on the left more.”

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

http://agilemanifesto.org/iso/en/manifesto.html

Waterfall vs Agile vs DevOps

	Development Process	Application Architecture	Deployment & Packaging	Application Infrastructure
~ 1980 ~ 1990	Waterfall	Monolithic	Physical Server	Datacenter
~ 2000	Agile	N-Tie	Virtual Servers	Hosted
~ 2010	DevOps	Microservices	Containers	Cloud

https://www.oracle.com/kr/cloud/cloud-native/what-is-cloud-native/

Cloud Native Architecture

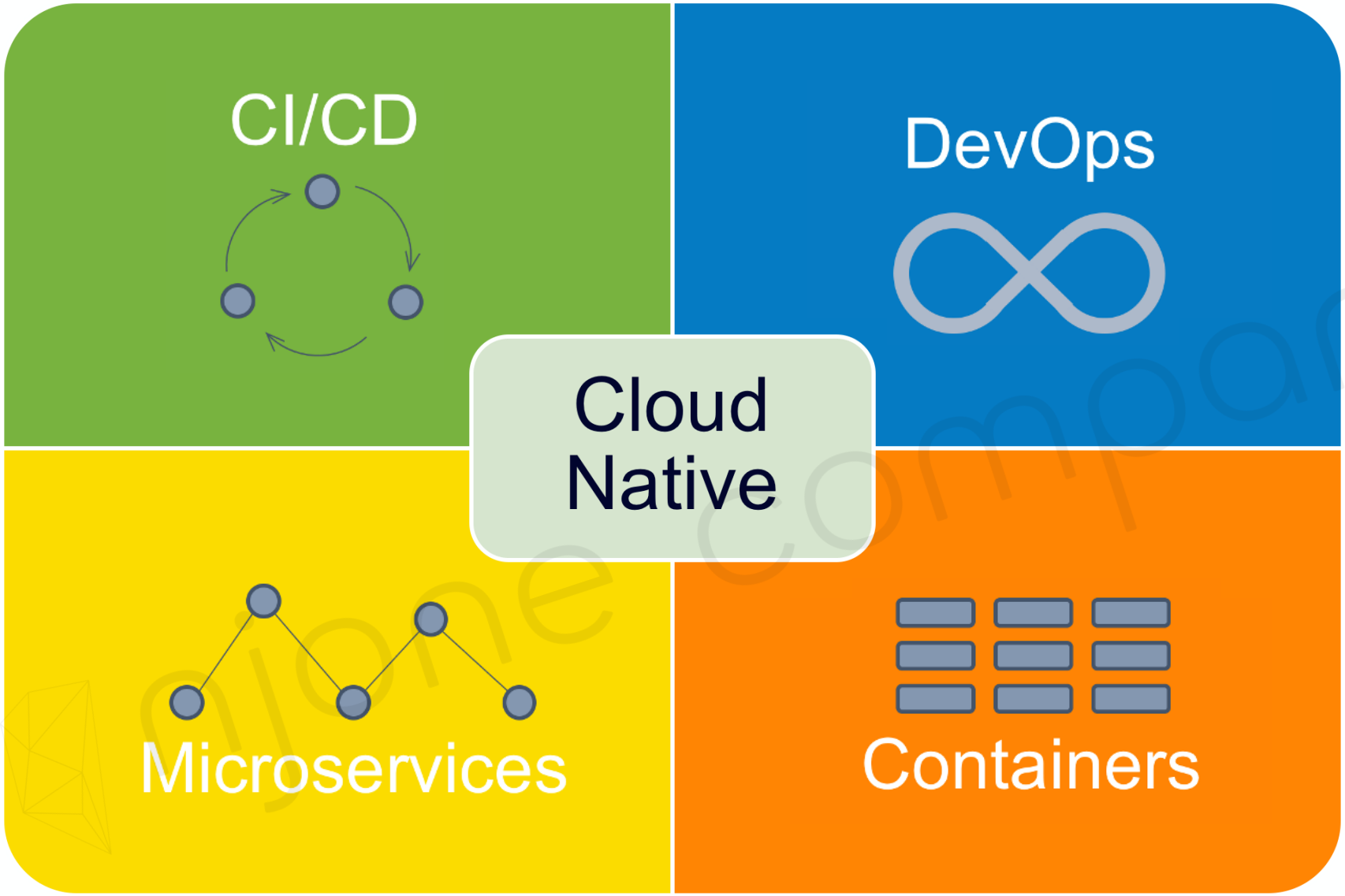
클라우드 네이티브 아키텍처 및 기술은 클라우드에서 빌드되고 클라우드 컴퓨팅 모델을 최대한 활용하는 워크로드를 디자인, 생성 및 운영하는 접근 방식입니다.

클라우드 네이티브 기술을 통해 조직은 퍼블릭, 프라이빗 및 하이브리드 클라우드와 같은 최신 동적 환경에서 확장 가능한 애플리케이션을 빌드하고 실행할 수 있습니다. 컨테이너, 서비스 메시, 마이크로 서비스, 변경할 수 없는 인프라 및 선언적 API는 이 접근 방식을 예로 들 수 있습니다.

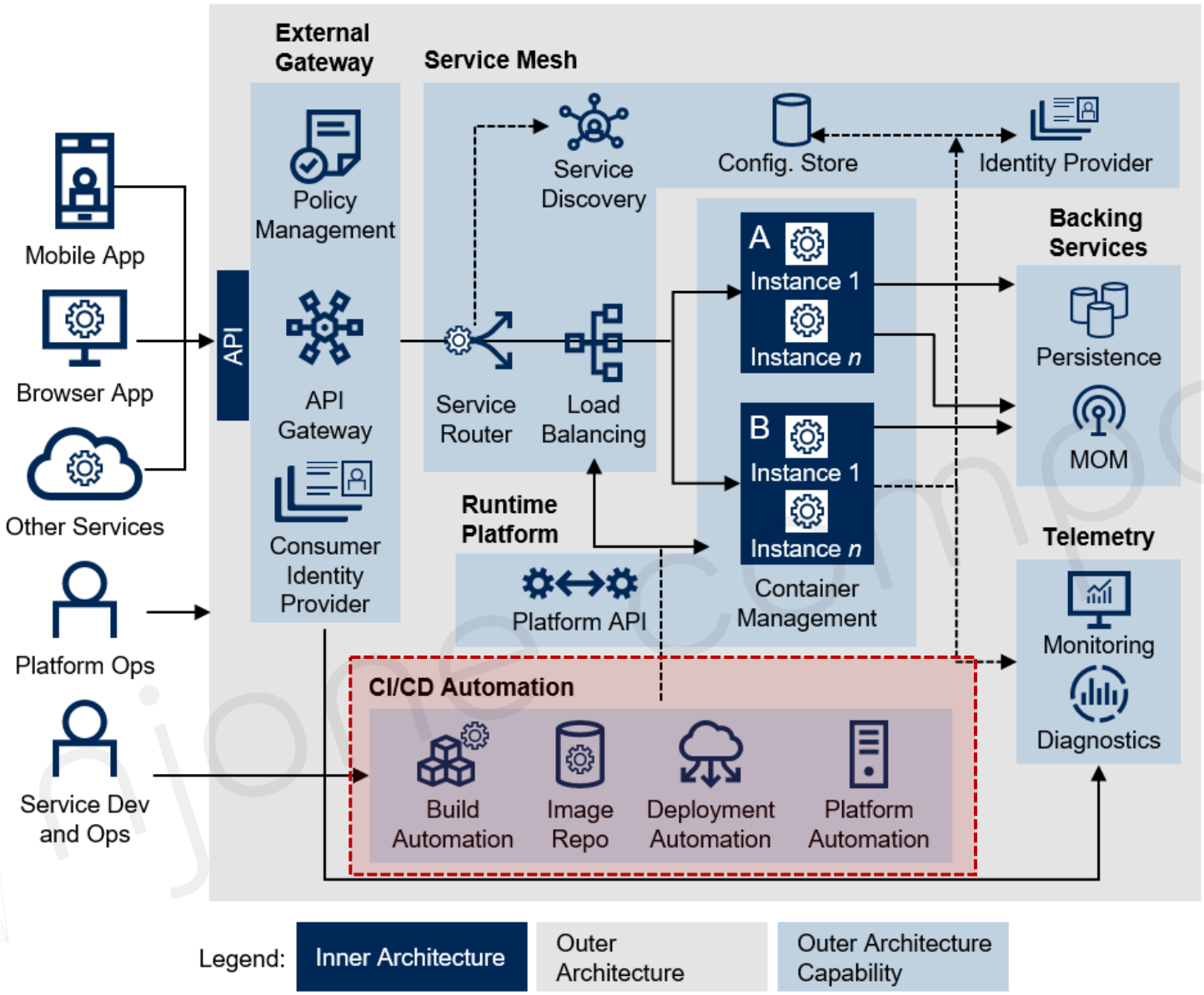
이러한 기술을 사용하면 복원력, 관리 가능 및 관찰 가능한 느슨하게 결합된 시스템을 사용할 수 있습니다. 강력한 자동화와 결합되어 엔지니어는 최소한의 수고로 자주 예측 가능하게 높은 영향을 미치는 변경을 할 수 있습니다.

https://docs.microsoft.com/ko-kr/dotnet/architecture/cloud-native/definition

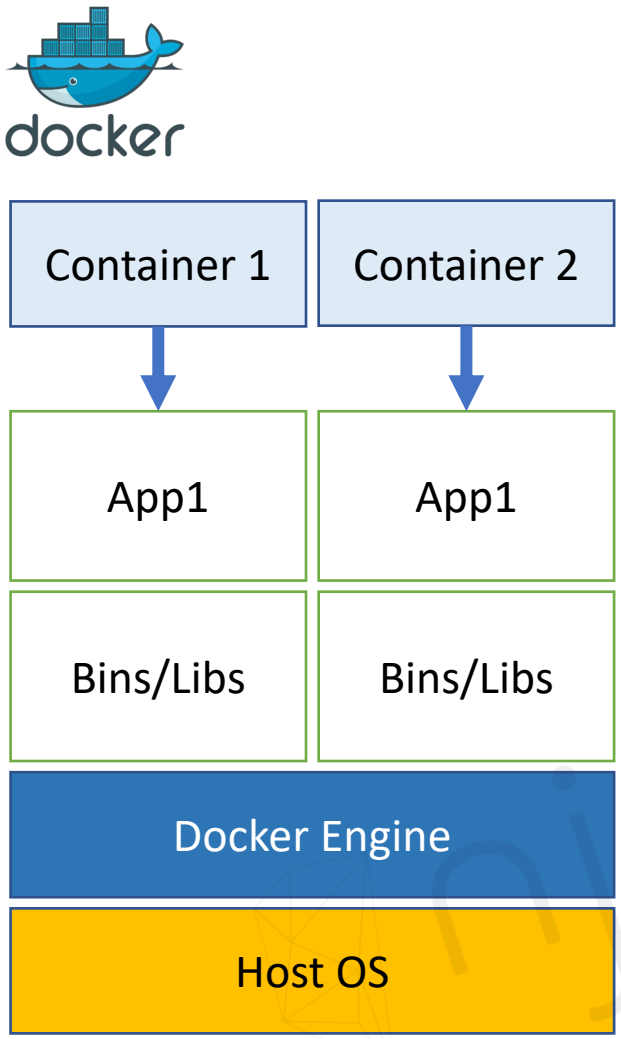
Cloud Native Application

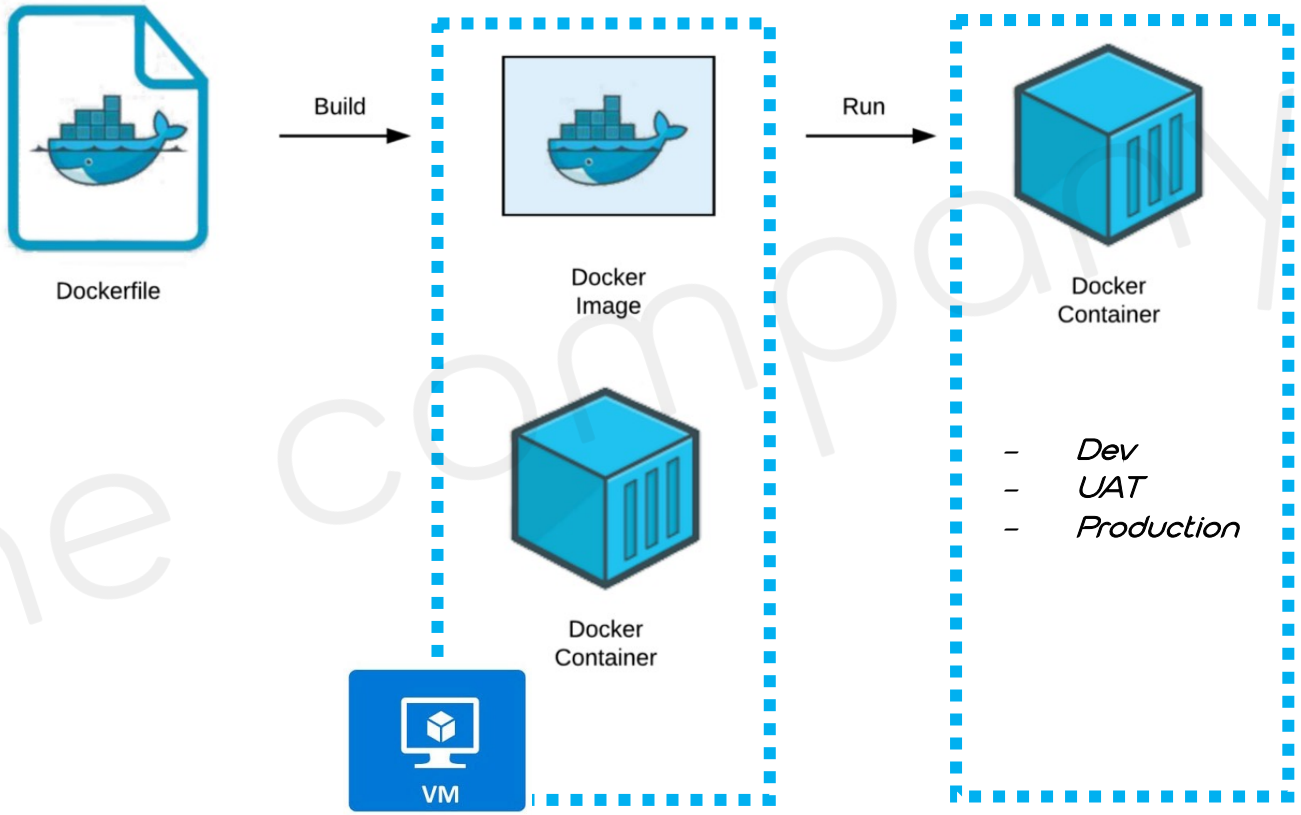


Cloud Native – MSA



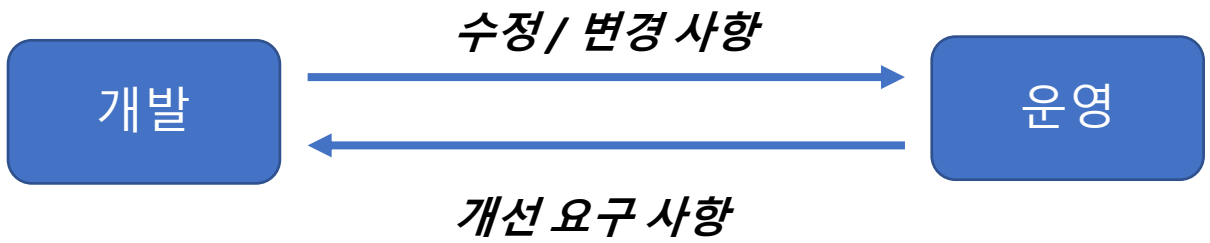
Cloud Native – Containerization





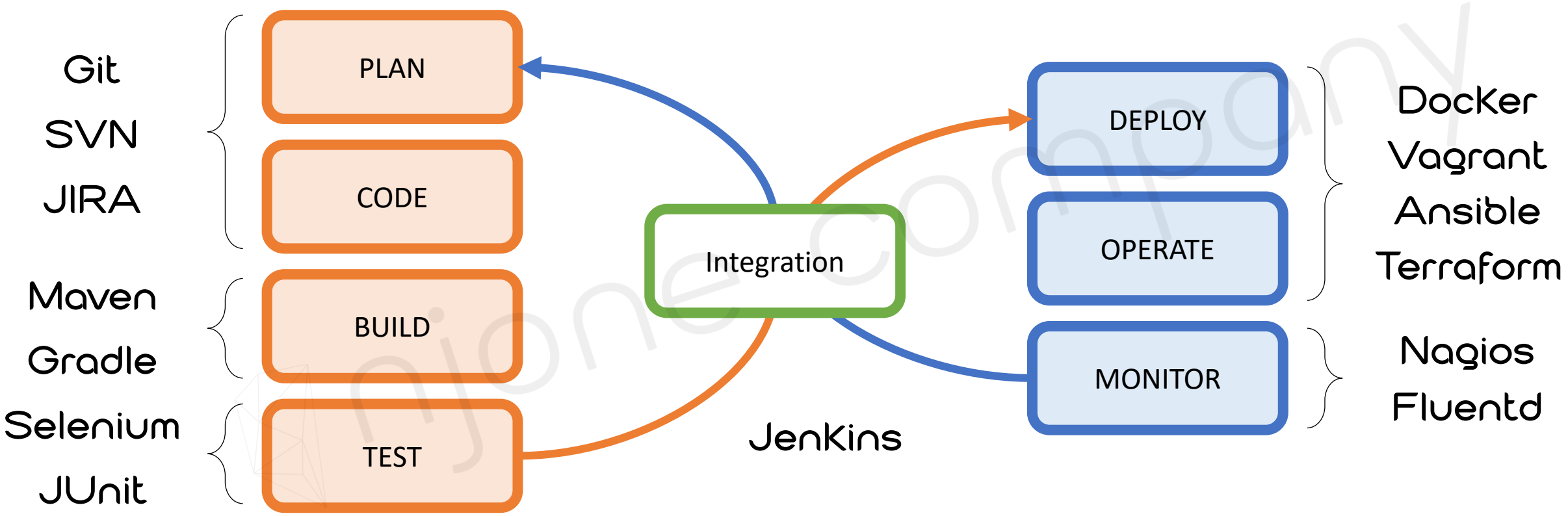
Cloud Native – DevOps

- ***2007~2008년 → IT 운영 및 SW 개발에 문제점 대두***
  - 개발과 배포가 다른 조직에서 관리 (다른 목표를 가짐)
- ***2009년 Belgium에서 첫 DevOps 컨퍼런스 →***
  - ***Development + Operations***
  - 인프라로 코드 관리 (Mark Burgess and Luke Kanies)
  - 애자일 인프라 스트럭쳐 (Andrew Shafer)
  - 애자일 시스템 관리 운동 (Patrick Debois)
  - Lean Startup (Eric Ries)
  - 지속적인 통합 및 배포 운동 → CI, CD

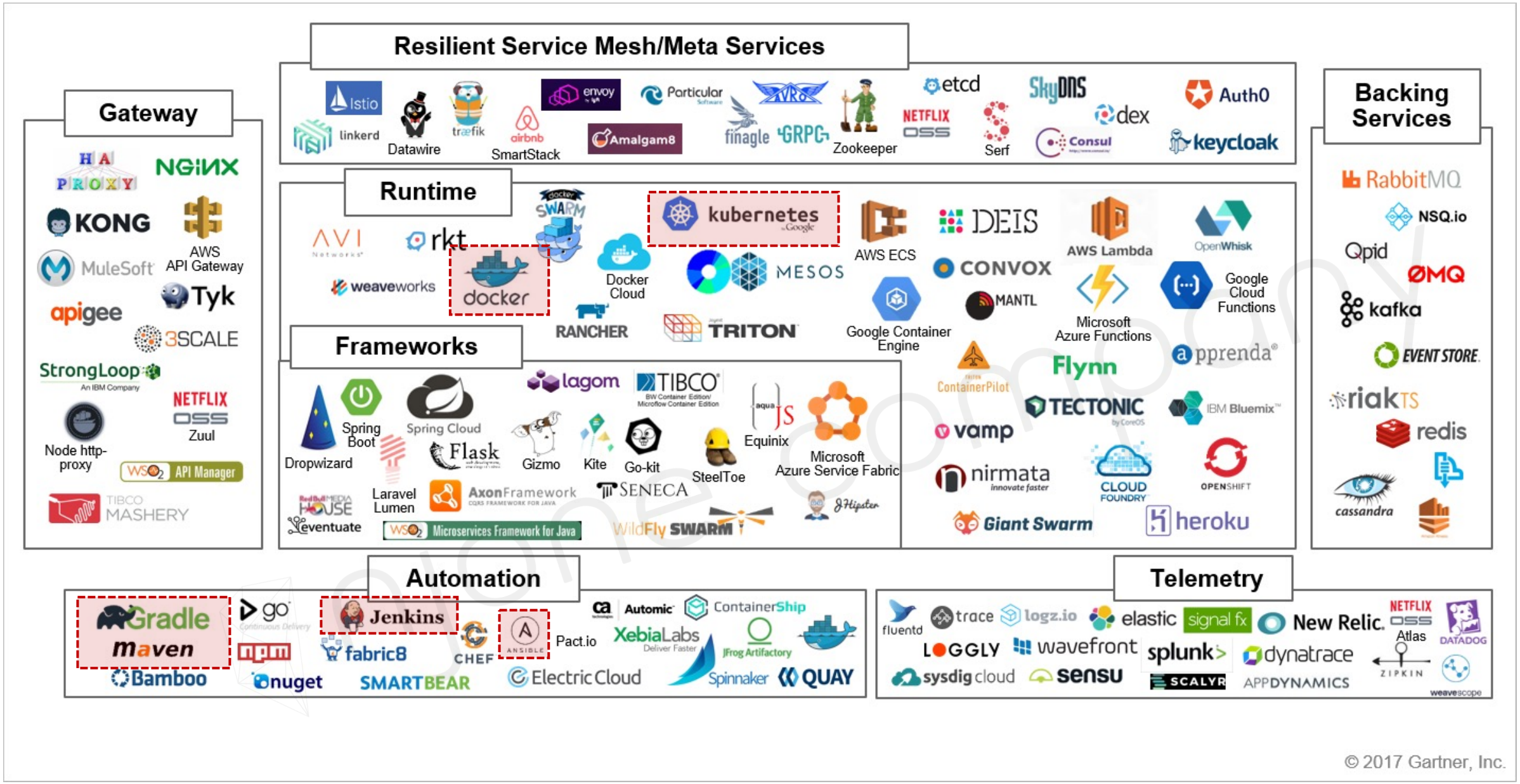


Cloud Native – DevOps

- *엔지니어가, 프로그래밍하고, 빌드하고, 직접 시스템에 배포 및 서비스를 RUN*
- *사용자와 끊임 없이 Interaction 하면서 서비스를 개선해 나가는 일련의 과정, 문화*

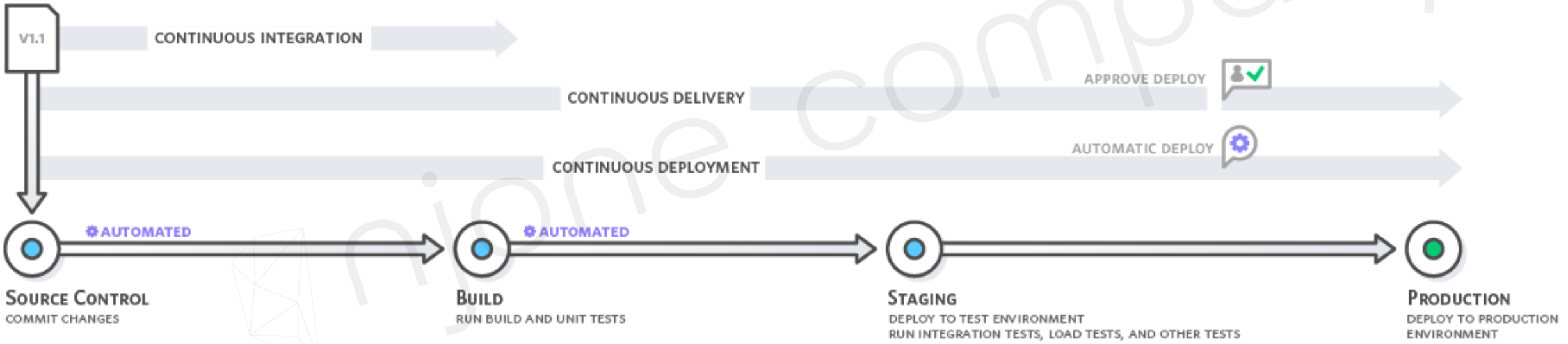


Cloud Native – DevOps



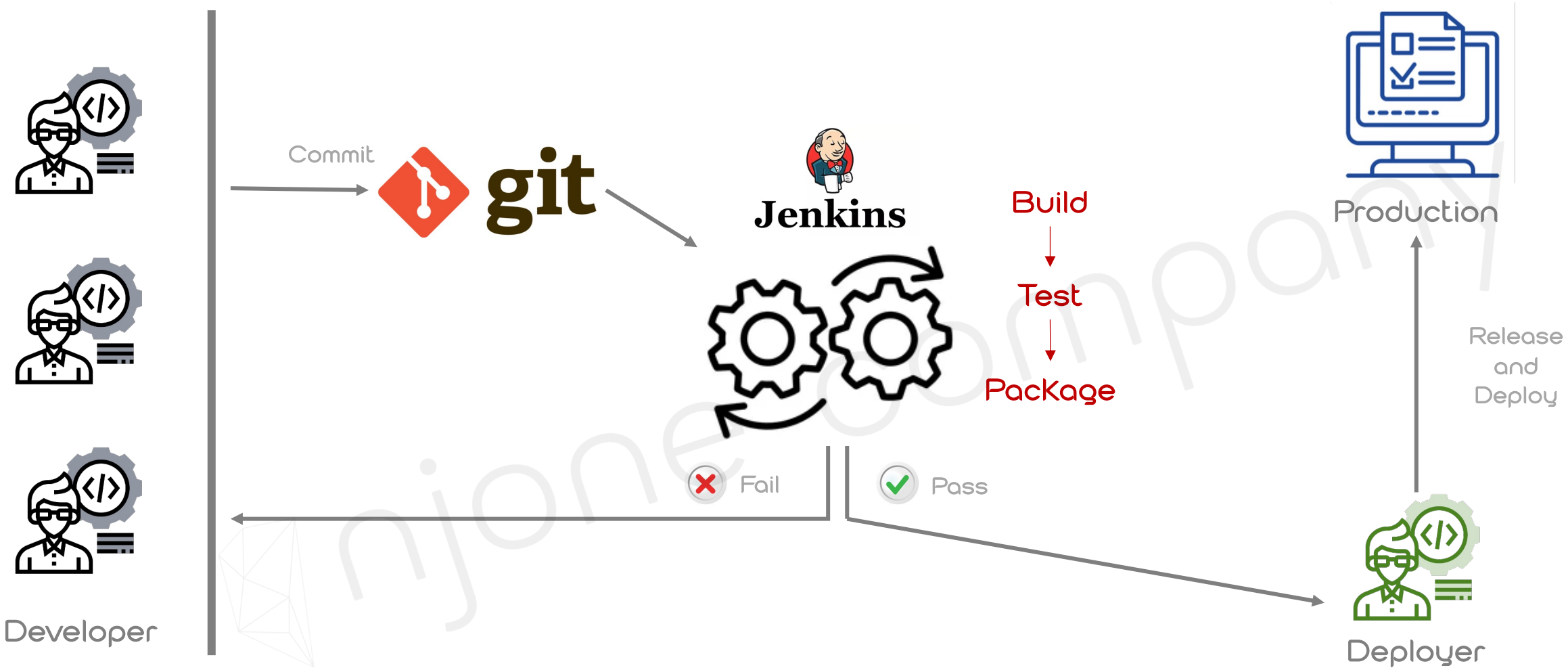
Cloud Native – CI/CD

- CI(Continuous Integration)?
- CD(Continuous Delivery)?
- CD(Continuous Deployment)?

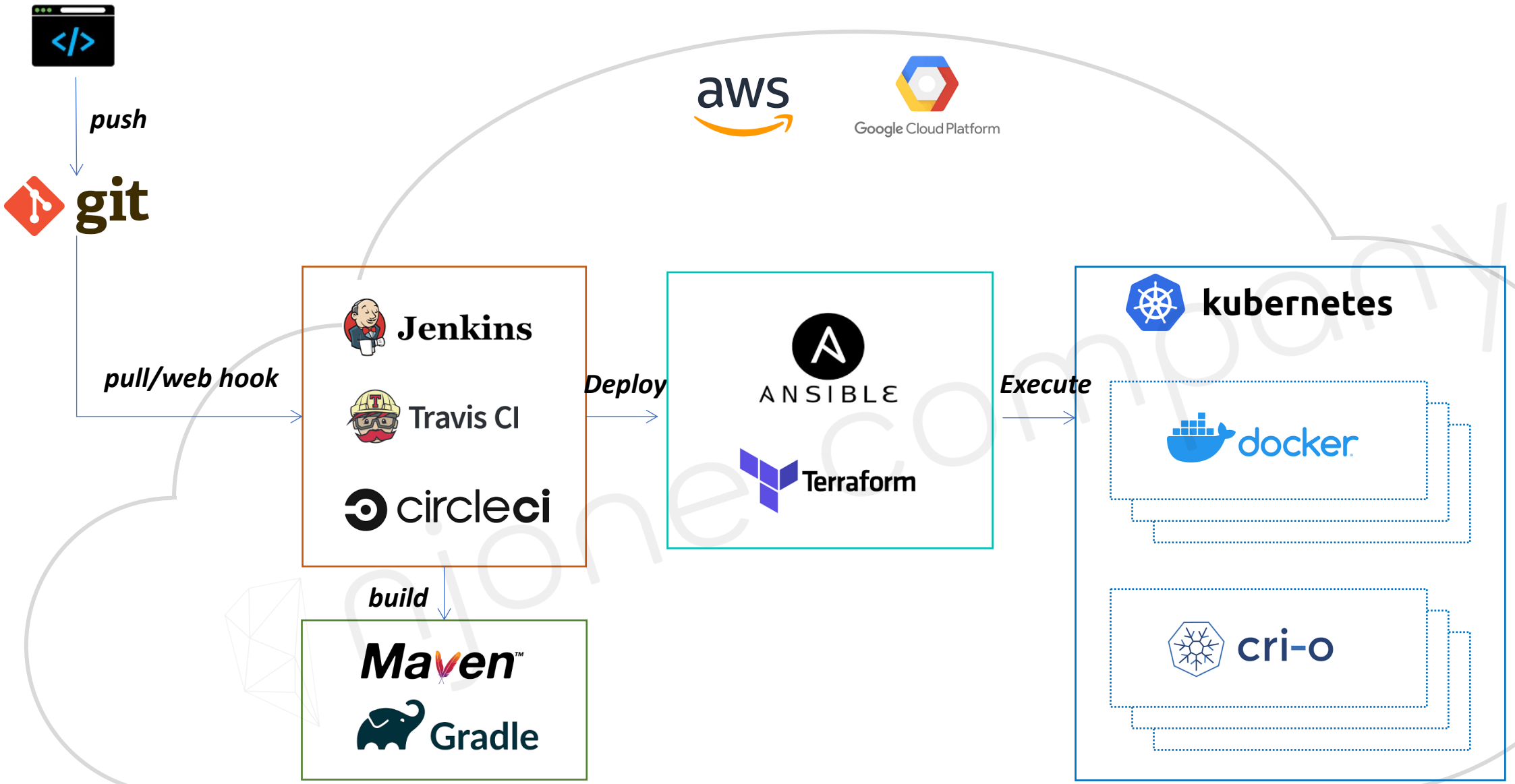


https://aws.amazon.com/ko/devops/continuous-integration

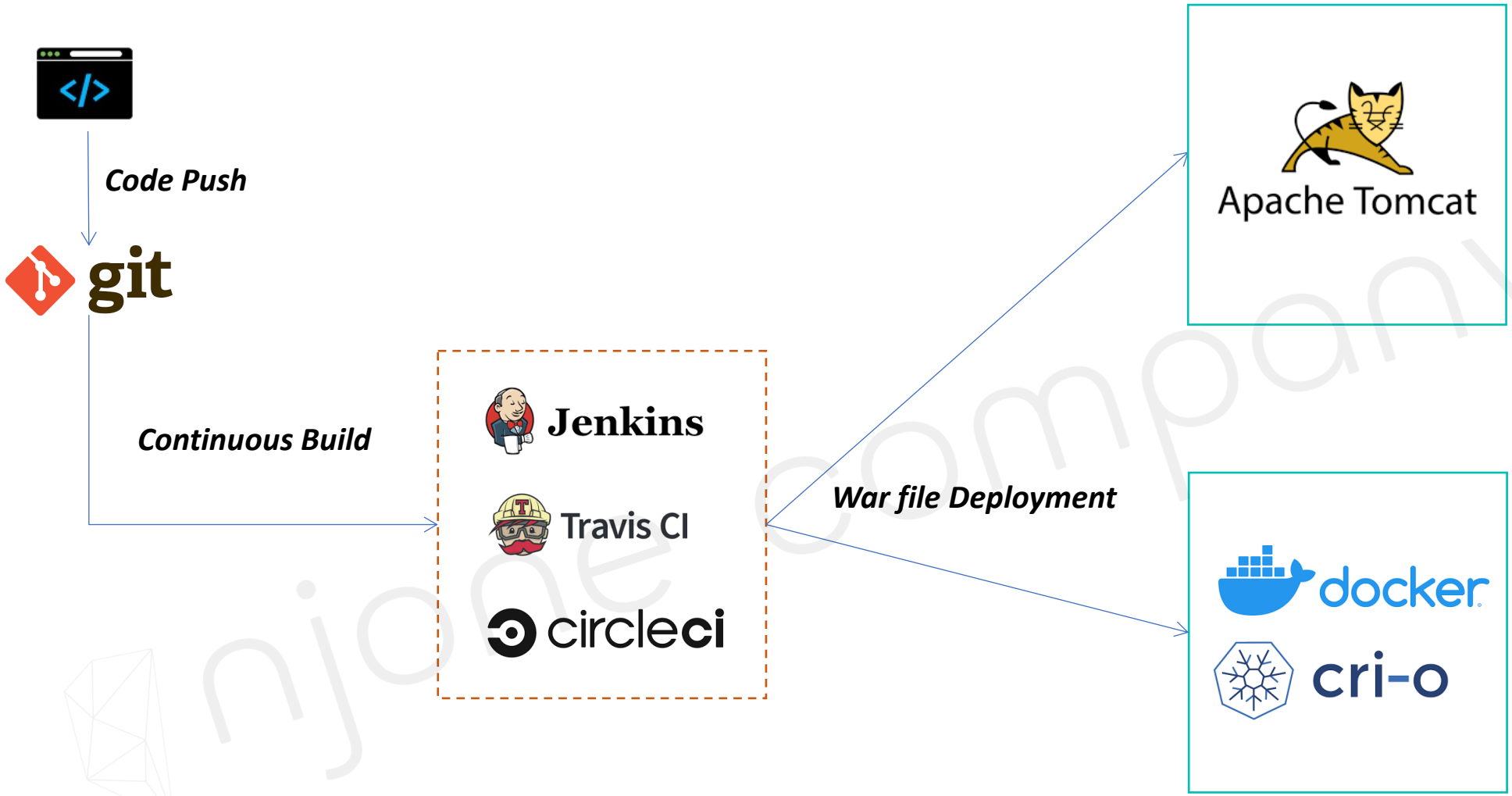
What is Continuous Integration?



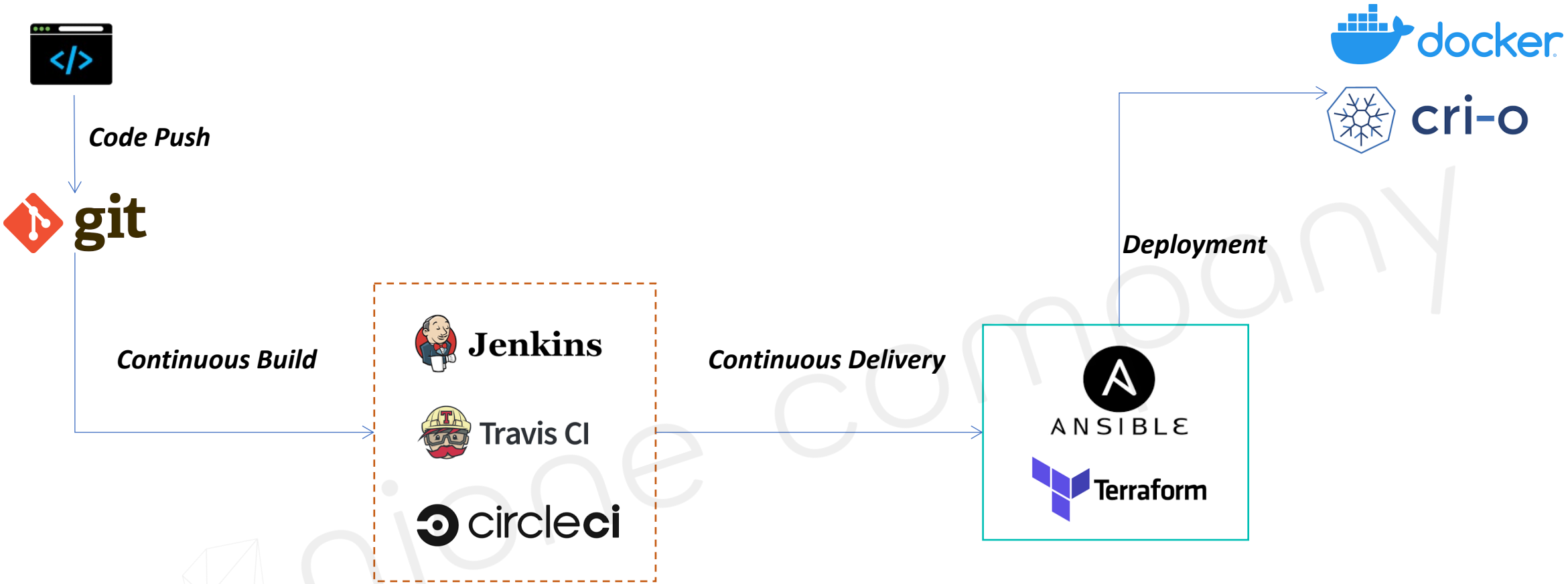
CI/CD Flow


push
git
pull/web hook
aws
Google Cloud Platform
Jenkins
Travis CI
circleci
build
Maven
Gradle
Deploy
ANSIBLE
Terraform
Execute
kubernetes
docker
cri-o

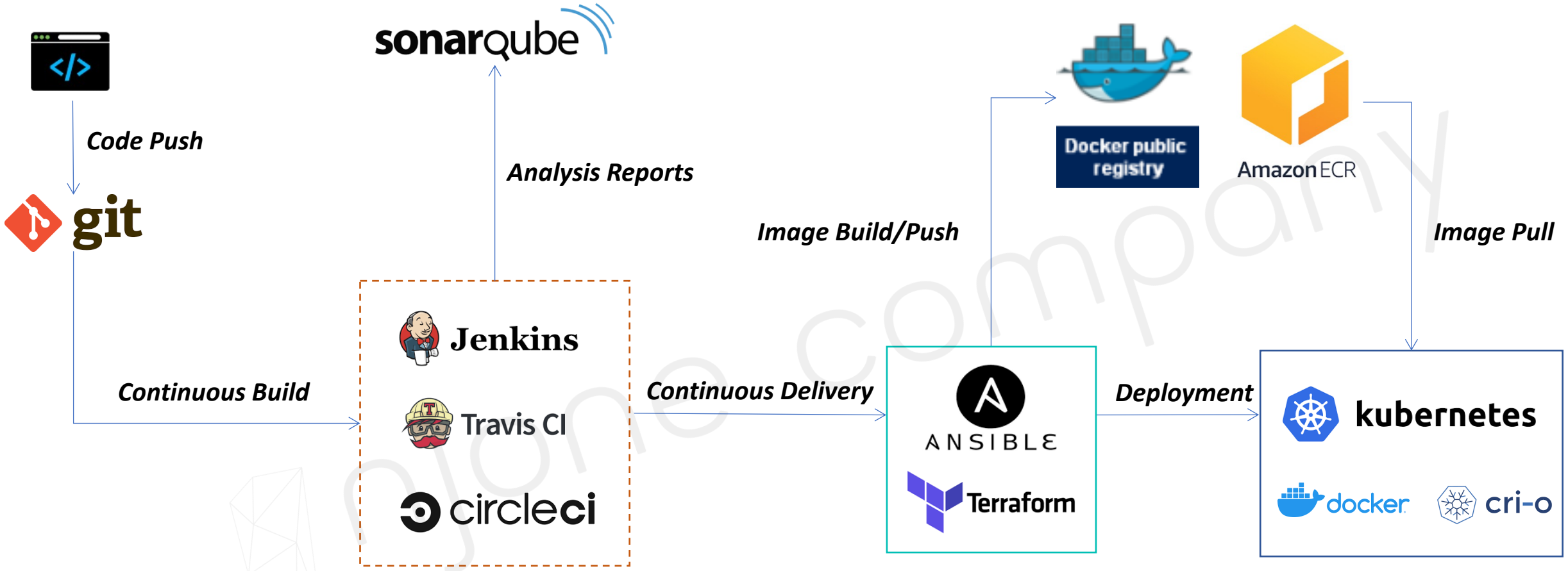
Deploy on Docker using Jenkins



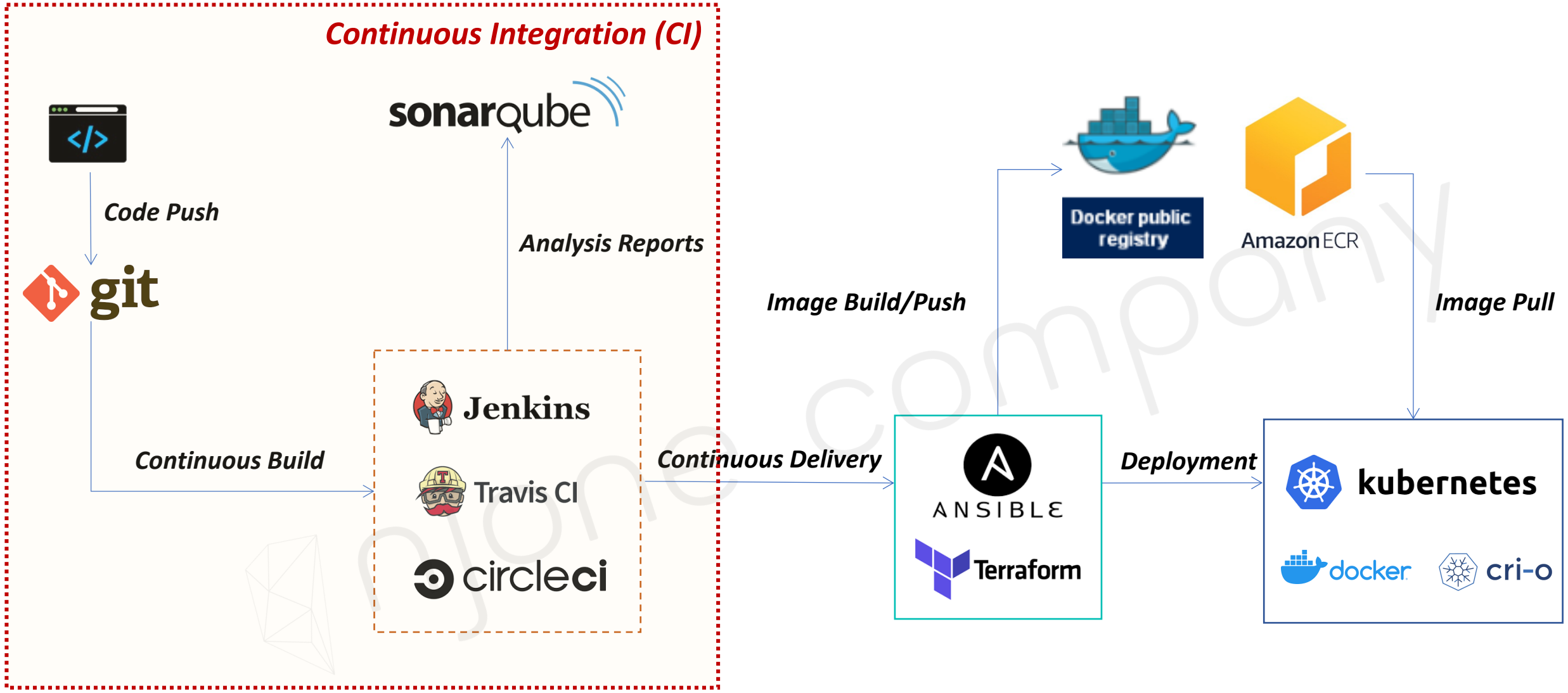
Deploy on Docker using Jenkins



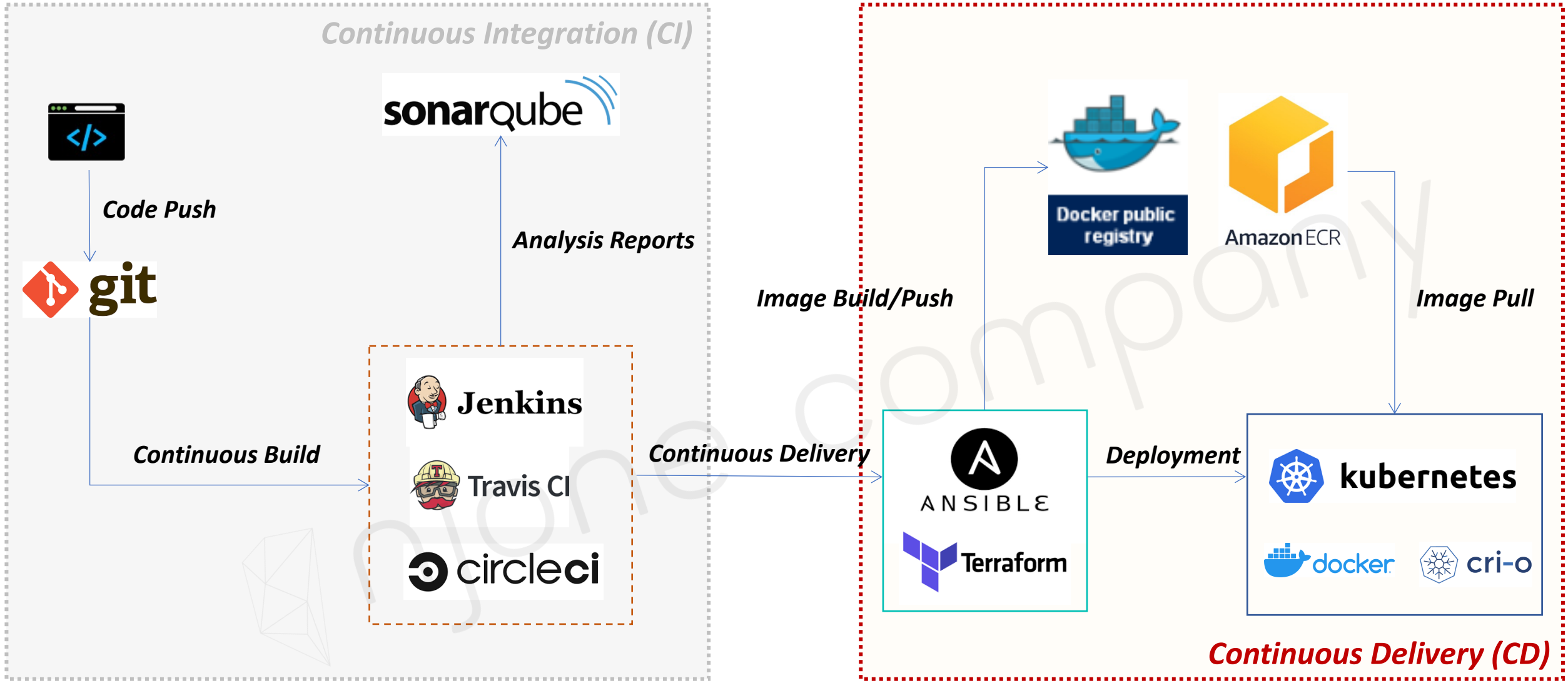
Deploy on Kubernetes



Continuous Integration(CI)



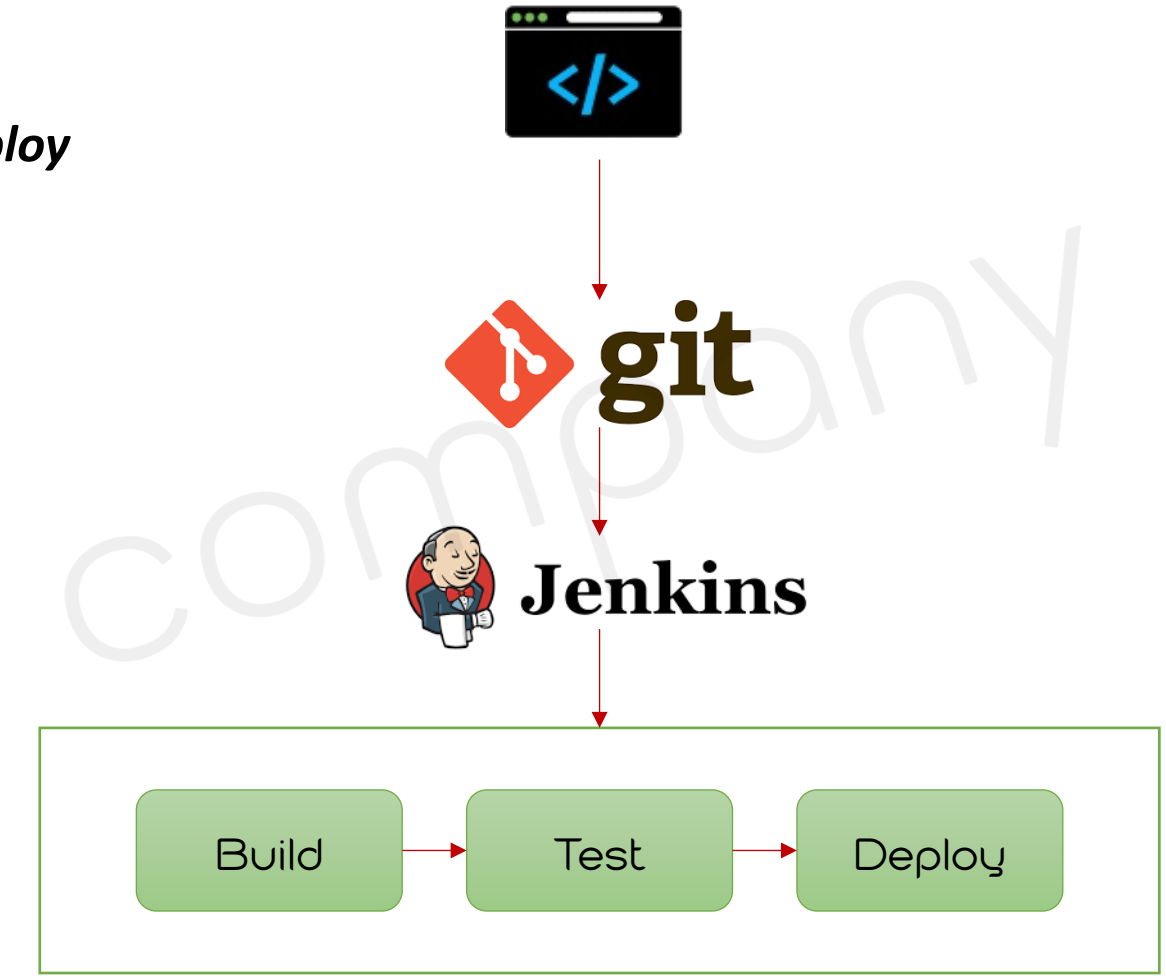
Continuous Delivery(CD)



Deploy on EC2/VM

Jenkins

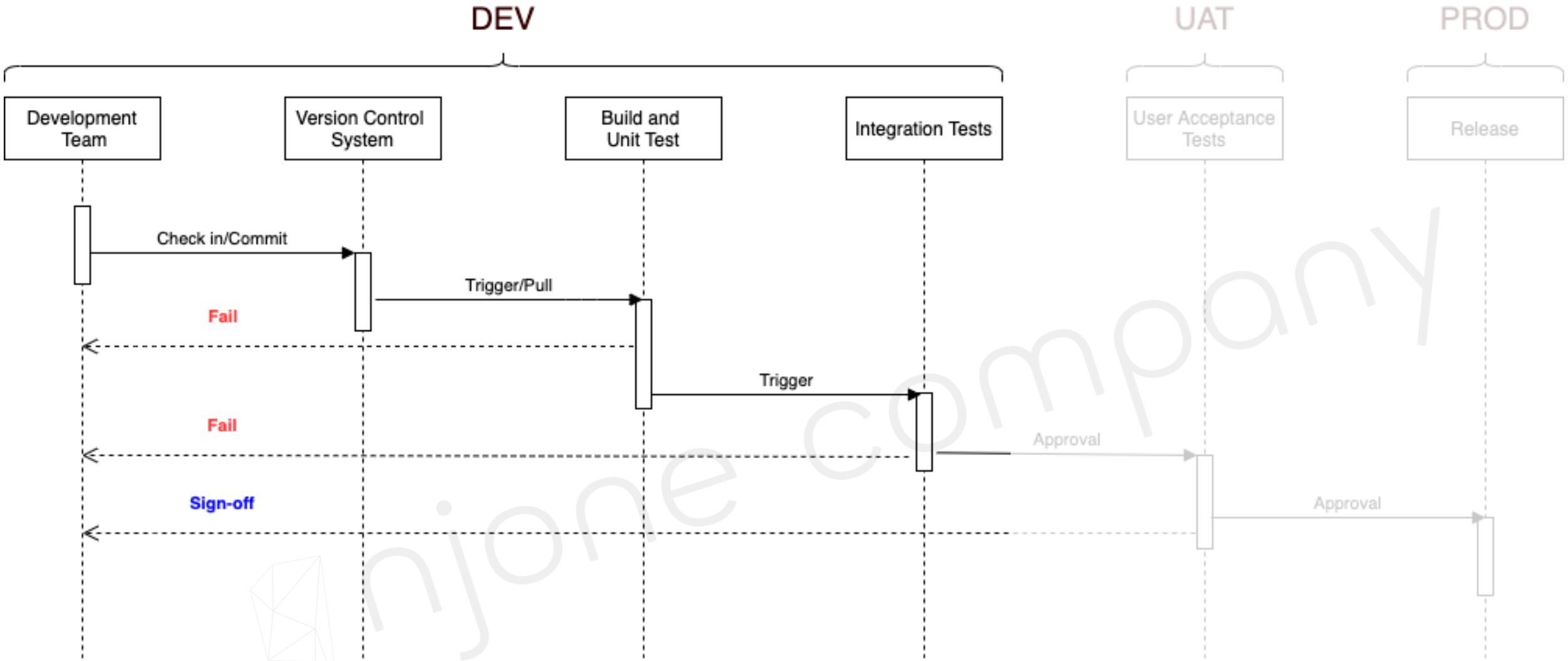
- 지속적인 통합과 배포 → ***Work flow*** *를 제어*
  - ***Continuous Integration Server***
    - ***Continuous Development, Build, Test, Deploy***
- 다양한 Plugins 연동
  - *Build Plugins:* ***Maven, Ant, Gradle …***
  - *VCS Plugins:* ***Git, SVN …***
  - *Languages Plugins:* ***Java, Python, Node.js …***



- Top 5 CI/CD tools

	Jenkins	circleci	TeamCity	Bamboo	GitLab
Open source	Yes	No	No	No	No
Ease of use & setup	Medium	Medium	Medium	Medium	Medium
Built-in features	3/5	4/5	4/5	4/5	4/5
Integration	★ ★ ★ ★ ★	★ ★ ★	★ ★ ★ ★	★ ★ ★	★ ★ ★ ★
Hosting	On premise & Cloud	On premise & Cloud	On premise	On premise & Bitbucket as Cloud	On premise & Cloud
Free version	Yes	Yes	Yes	Yes	Yes
Build Agent License Pricing	Free	From $39 per month	From $299 one-off payment	From $10 one-off payment	From $4 per month per user
Supported OSs	Windows, Linux, macOS, Unix-like OS	Linux or MacOS	Windows, Linux, macOS, Solaris, FreeBSD and more	Windows, Linux, macOS, Solaris	Linux distributions: Ubuntu, Debian, CentOS, Oracle Linux

Jenkins Pipeline



Install Jenkins

- https://www.jenkins.io/download/

Downloading Jenkins

Jenkins is distributed as WAR files, native packages, installers, and Docker images. Follow these installation steps:

1. Before downloading, please take a moment to review the **Hardware and Software requirements** section of the User Handbook.
2. Select one of the packages below and follow the download instructions.
3. Once a Jenkins package has been downloaded, proceed to the **Installing Jenkins** section of the User Handbook.
4. You may also want to verify the package you downloaded. Learn more about verifying Jenkins downloads.

Download Jenkins 2.303.1 LTS for:

- Generic Java package (.war)
  SHA-256: 4aae135cde63e398a1f59d37978d97604cb595314f7041d2d3bac3f0bb32c065
- Docker
- Ubuntu/Debian
- CentOS/Fedora/Red Hat
- Windows
- openSUSE
- FreeBSD
- Gentoo
- macOS
- OpenBSD

Download Jenkins 2.309 for:

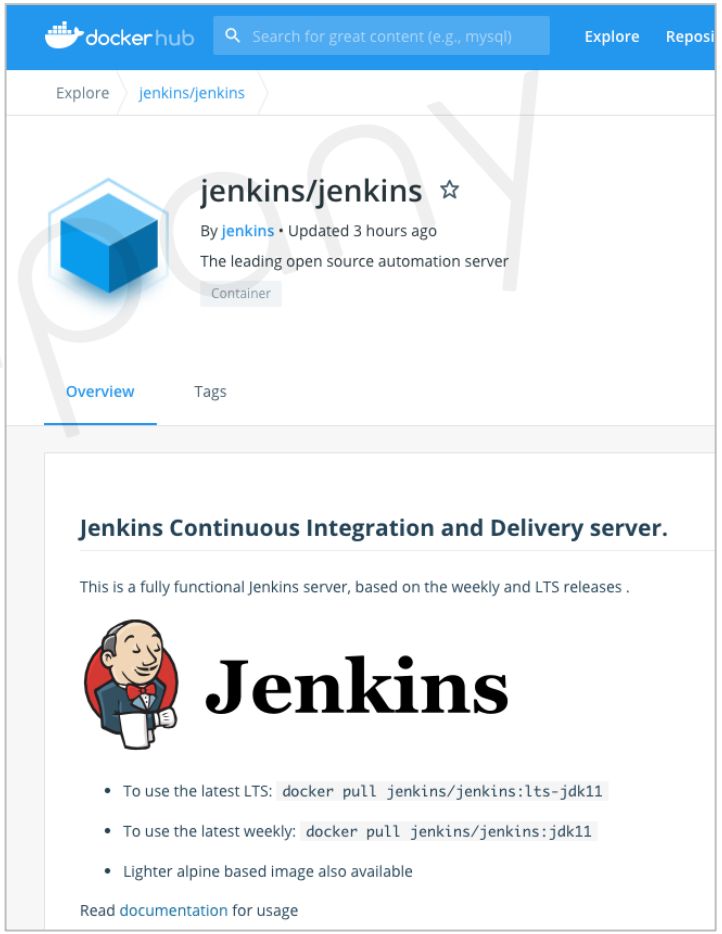
- Generic Java package (.war)
  SHA-256: 41c5de84c7afe8634212641e143ff02d75105c5697ab41141ff82b4c97916811
- Docker
- Ubuntu/Debian
- CentOS/Fedora/Red Hat
- Windows
- openSUSE
- Arch Linux
- FreeBSD
- Gentoo
- macOS

Install Jenkins

- https://hub.docker.com/r/jenkins/jenkins
- https://github.com/jenkinsci/docker/blob/master/README.md

```
$ docker run -d -p 8088:8080 -p 50000:50000 --name jenkins-server \
        --restart=on-failure \
        jenkins/jenkins:lts-jdk11
```

```
$ docker run -d -p 8088:8080 -p 50000:50000  --name jenkins-server \
        --restart=on-failure \
        -v jenkins_home:/var/jenkins_home \
        jenkins/jenkins:lts-jdk11
```



Install Jenkins

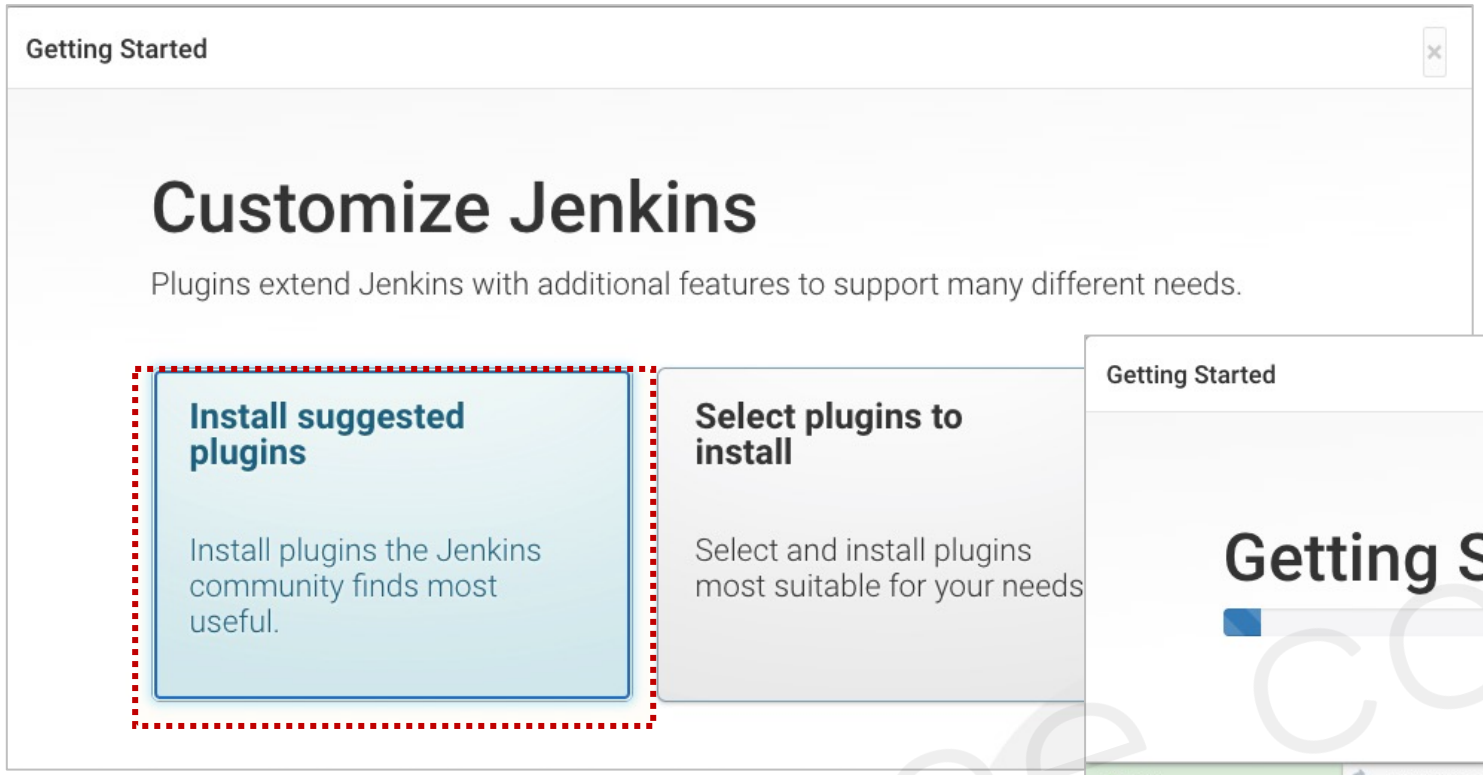
- Check the init password

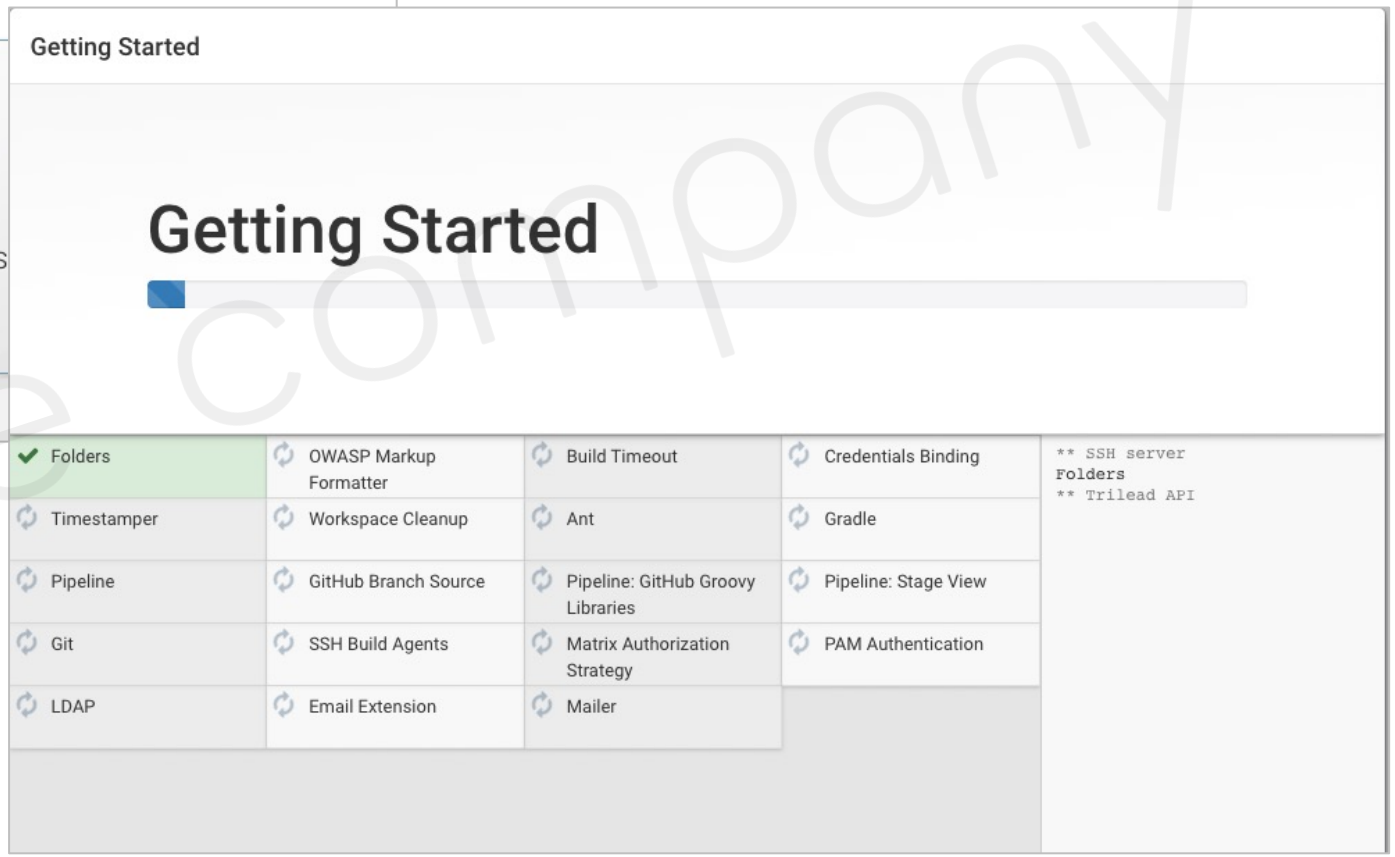
```
2021-09-02 06:35:47.922+0000 [id=29]    INFO    jenkins.install.SetupWizard#init:

*************************************************************
*************************************************************
*************************************************************

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

22263a23f9ed4e0ea8ed71f976f37c7b

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

*************************************************************
*************************************************************
*************************************************************

2021-09-02 06:35:58.710+0000 [id=29]    INFO    jenkins.InitReactorRunner$1#onAttained: Completed initialization
2021-09-02 06:35:58.726+0000 [id=23]    INFO    hudson.WebAppMain$3#run: Jenkins is fully up and running
2021-09-02 06:35:59.337+0000 [id=49]    INFO    h.m.DownloadService$Downloadable#load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
2021-09-02 06:35:59.338+0000 [id=49]    INFO    hudson.util.Retrier#start: Performed the action check updates server successfully at the attempt #1
2021-09-02 06:35:59.341+0000 [id=49]    INFO    hudson.model.AsyncPeriodicWork#lambda$doRun$0: Finished Download metadata. 11,777 ms
```

Setting Jenkins

- Access Jenkins
  - http://YOUR-SERVER-PUBLIC-IP:8088
  - Administrator Password: [Using the init password]

Setting Jenkins





Setting Jenkins

Getting Started

Create First Admin User

계정명:

암호:

암호 확인:

이름:

이메일 주소:

Instance Configuration

Jenkins URL: http://127.0.0.1:8080/

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.

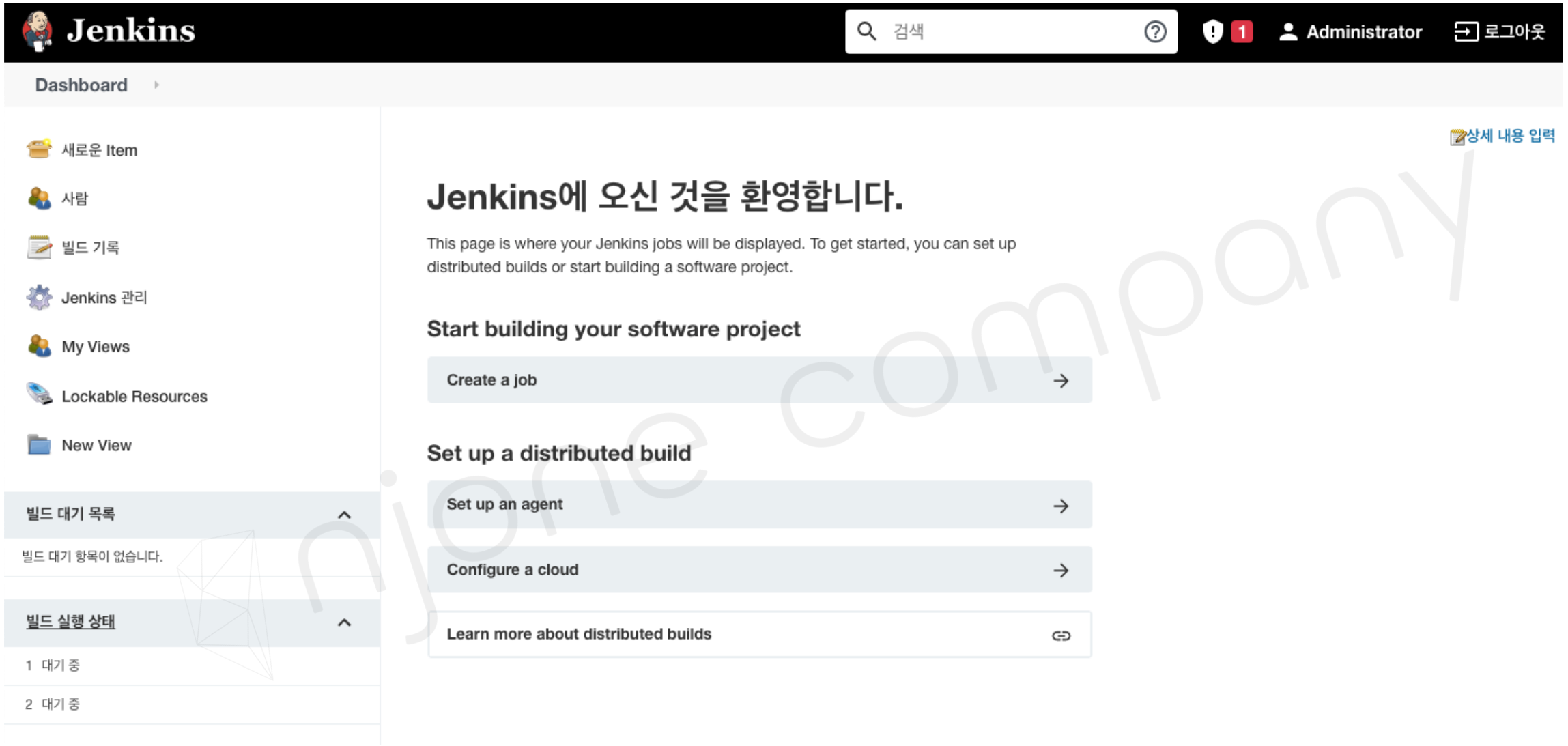
Jenkins is ready!

Your Jenkins setup is complete.

Start using Jenkins

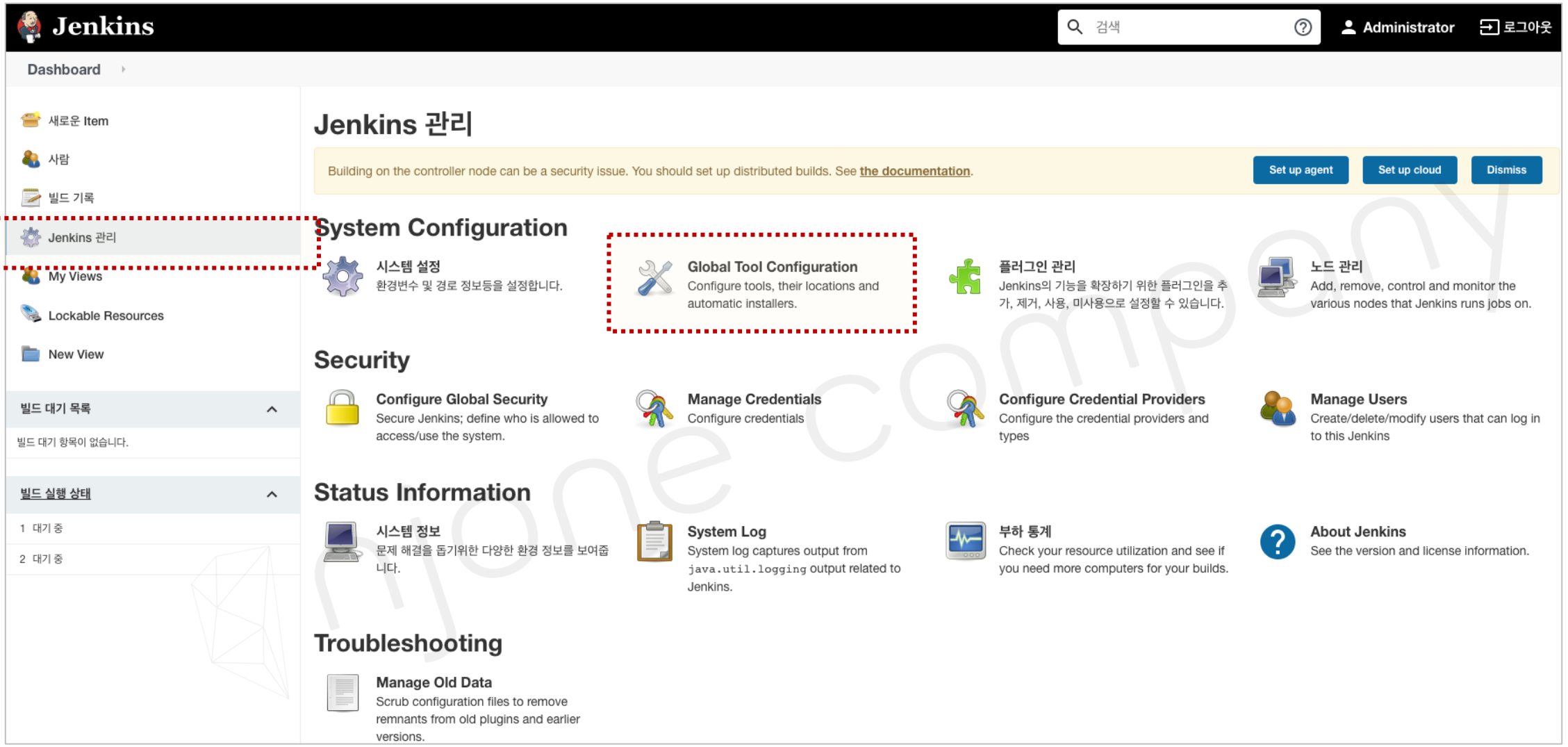
Setting Jenkins

- http://127.0.0.1:8088/



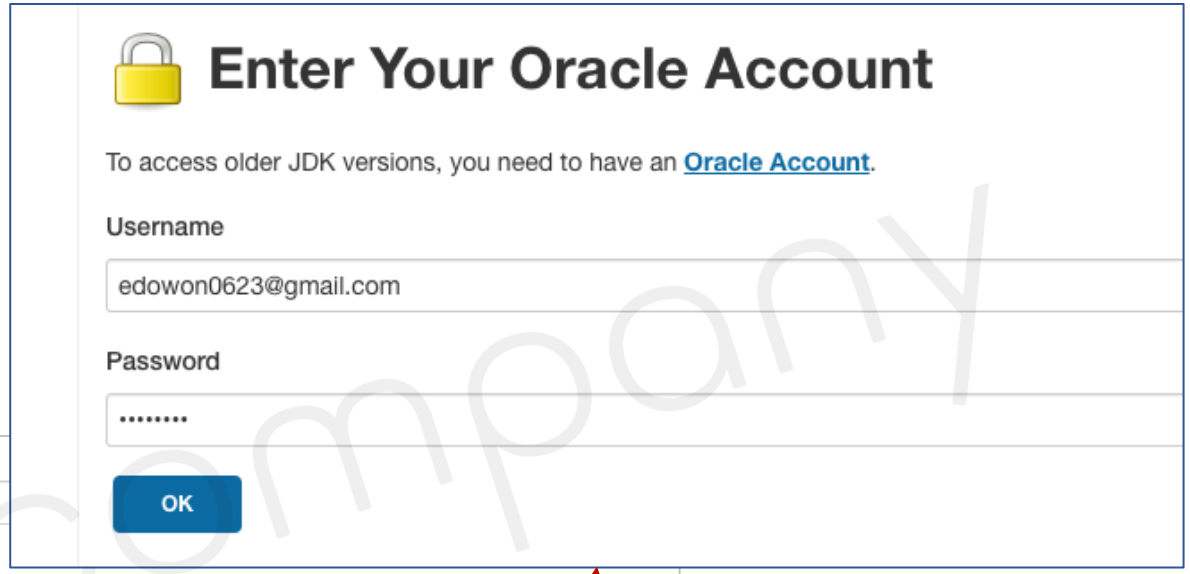
Setting Jenkins

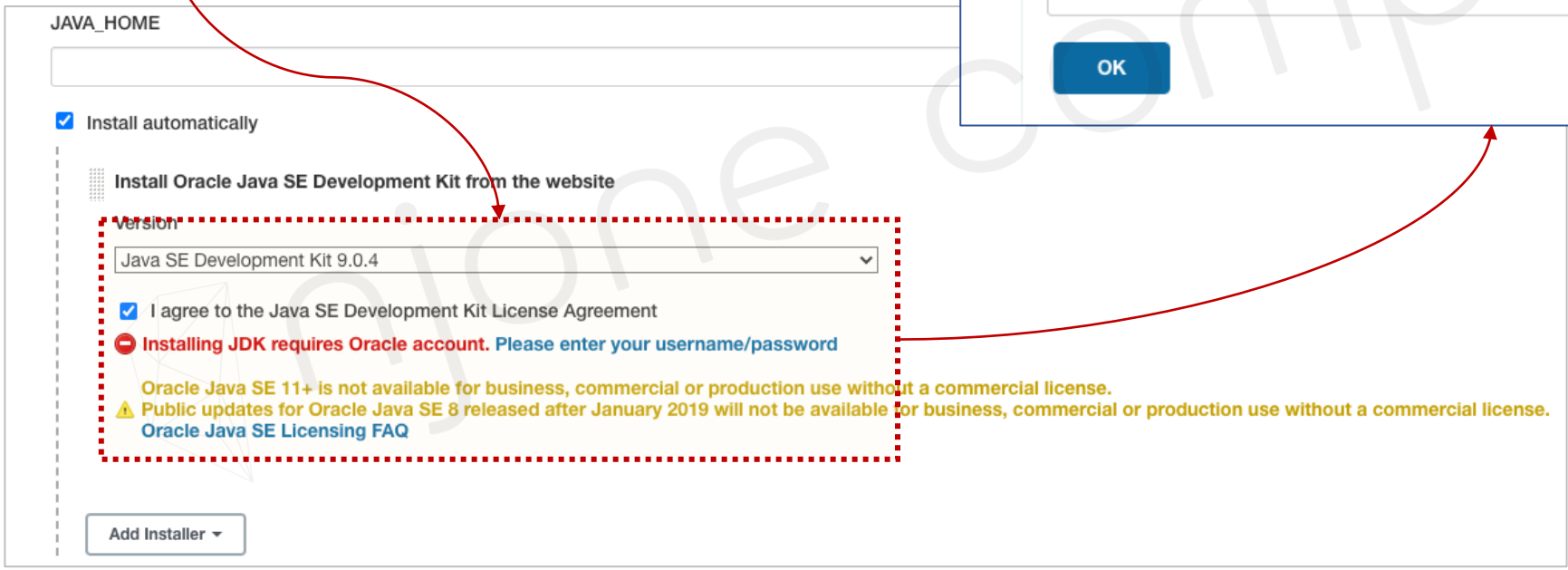
- Jenkins 설정



Setting Jenkins

- JDK



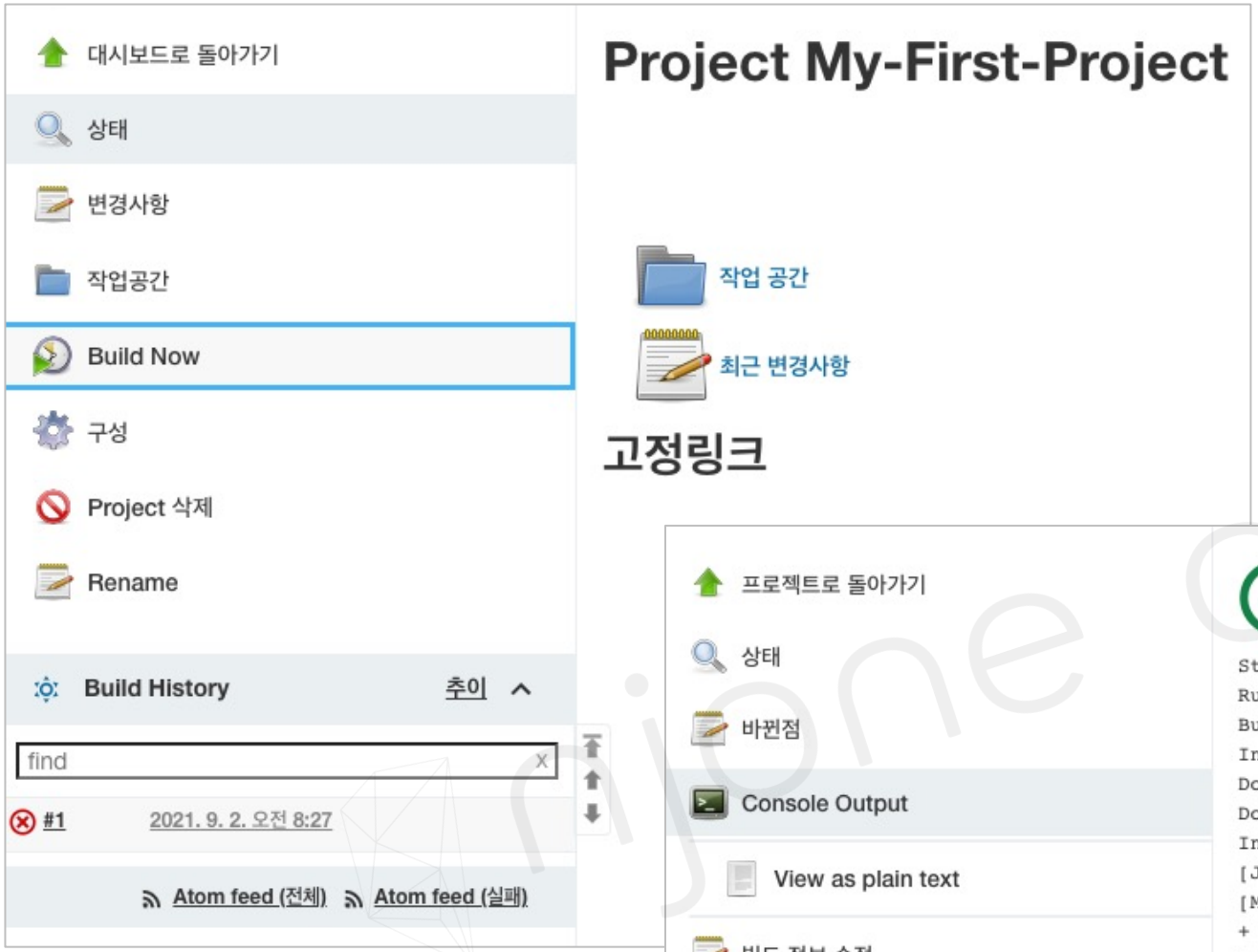


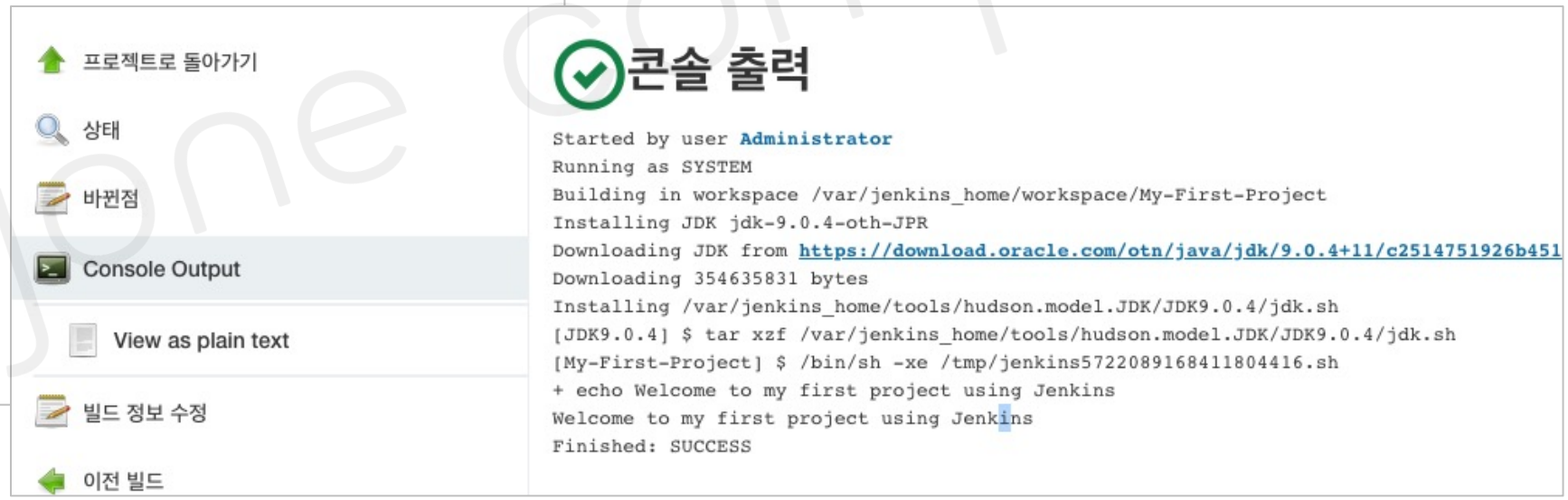
Exercise #1 Jenkins Job 1/2

- Item name → My-First-Project
  - ***Freestyle project***
  - ***Build > Execute shell***
  - ***Save***

Exercise #1 Jenkins Job 2/2

- Build Now





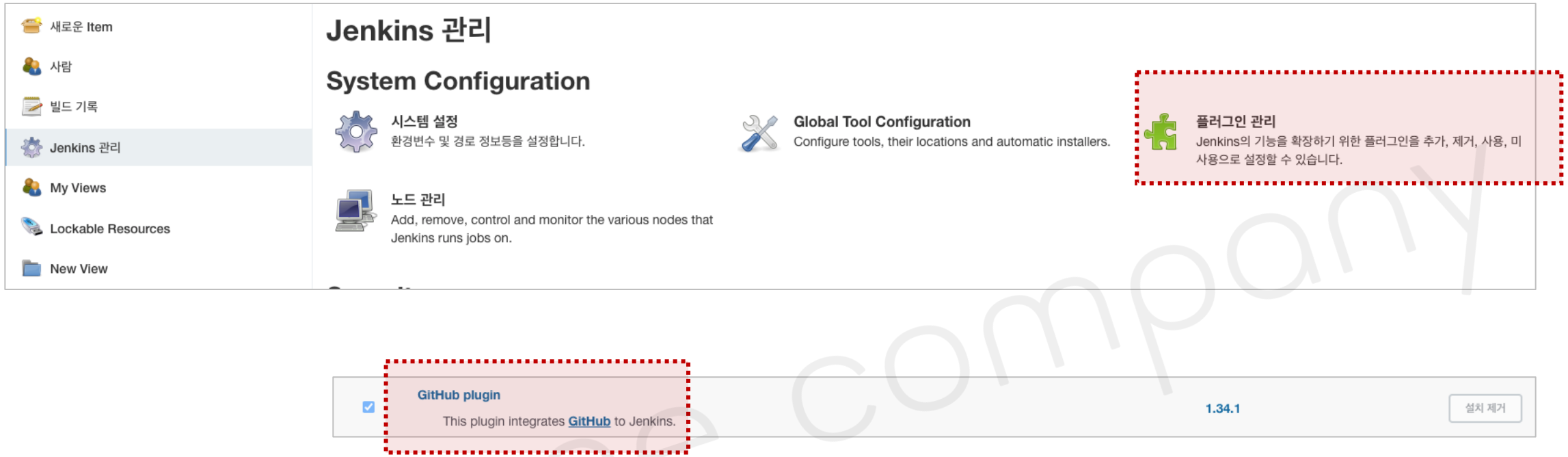
Section 2.

Jenkins를 이용한 CI/CD 사용

- CI/CD를 위한 Git & Maven 연동
- CI/CD를 위한 Tomcat Server 연동
- PollSCM 설정
- Docker를 이용한 실습 환경 구성

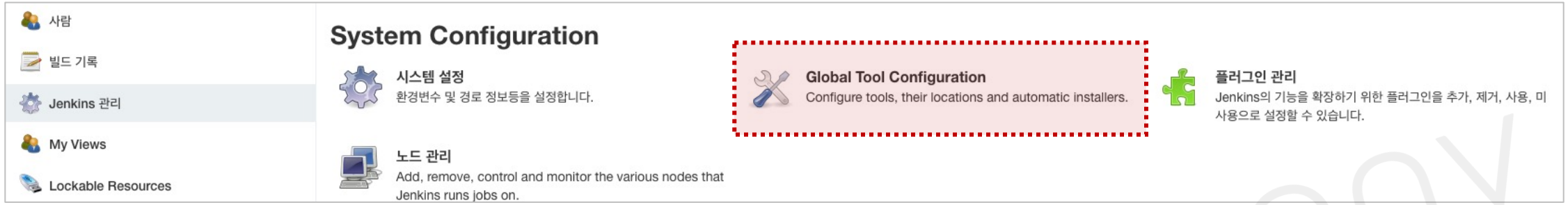
Setup Git plugin

- Manage Jenkins → Jenkins Plugins → available → github



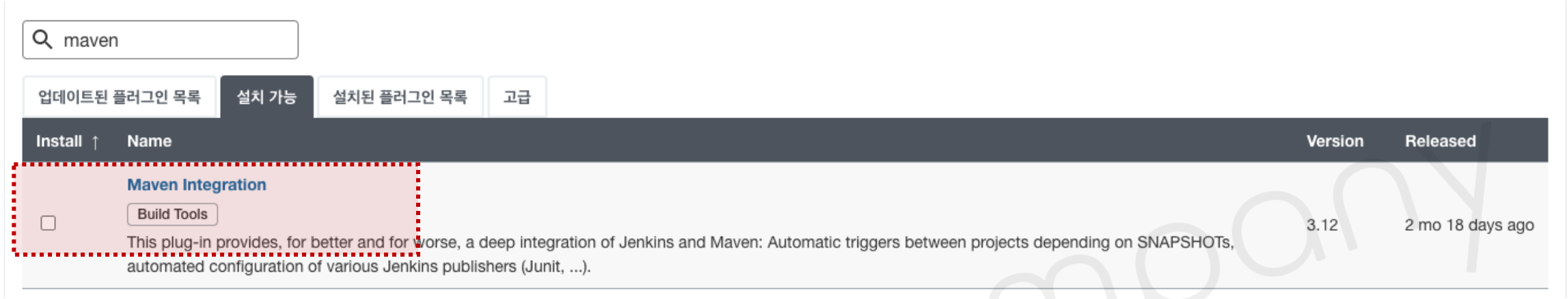
Setup Git plugin

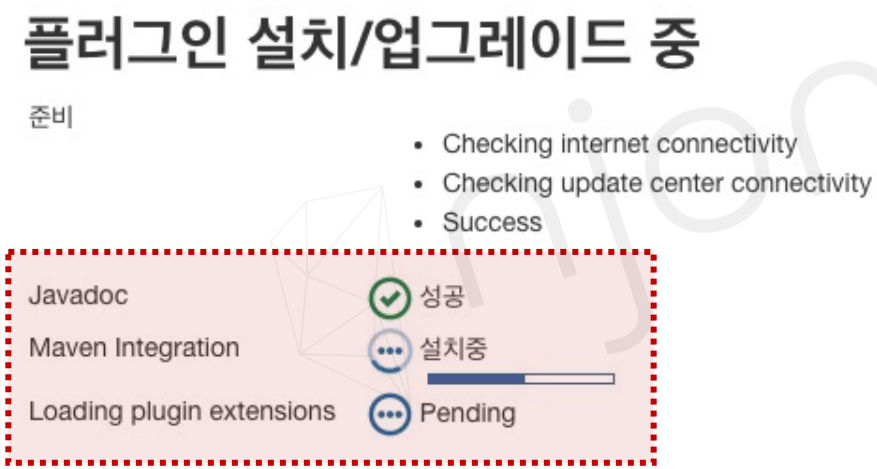
- Manage Jenkins → Global Tool Configuration → git



Setup Maven plugin

- Manage Jenkins → Jenkins Plugins → available → maven



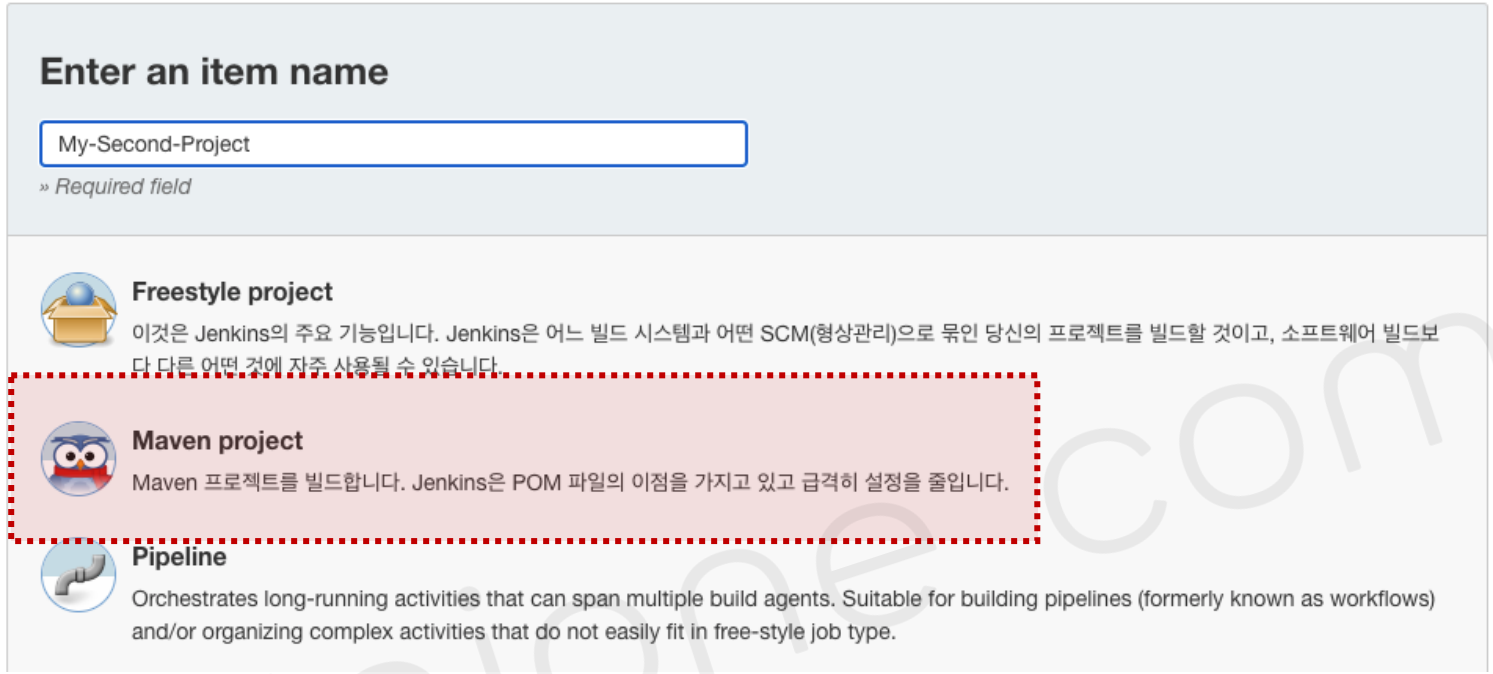


Setup Maven plugin

- Manage Jenkins → Global Tool Configuration → maven

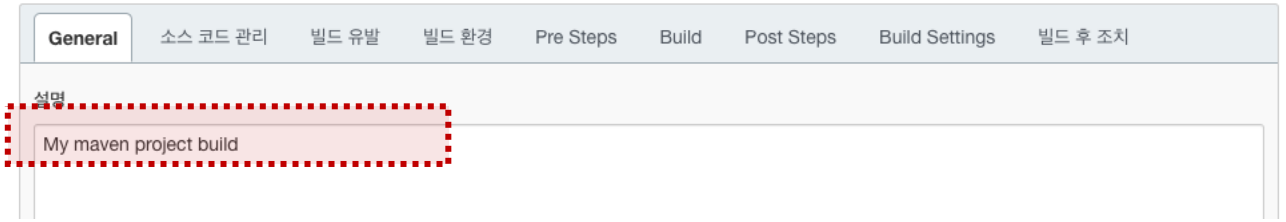
Exercise #2 Jenkins Job 1/4

- Item name → My-Second-Project
  - ***Maven project***



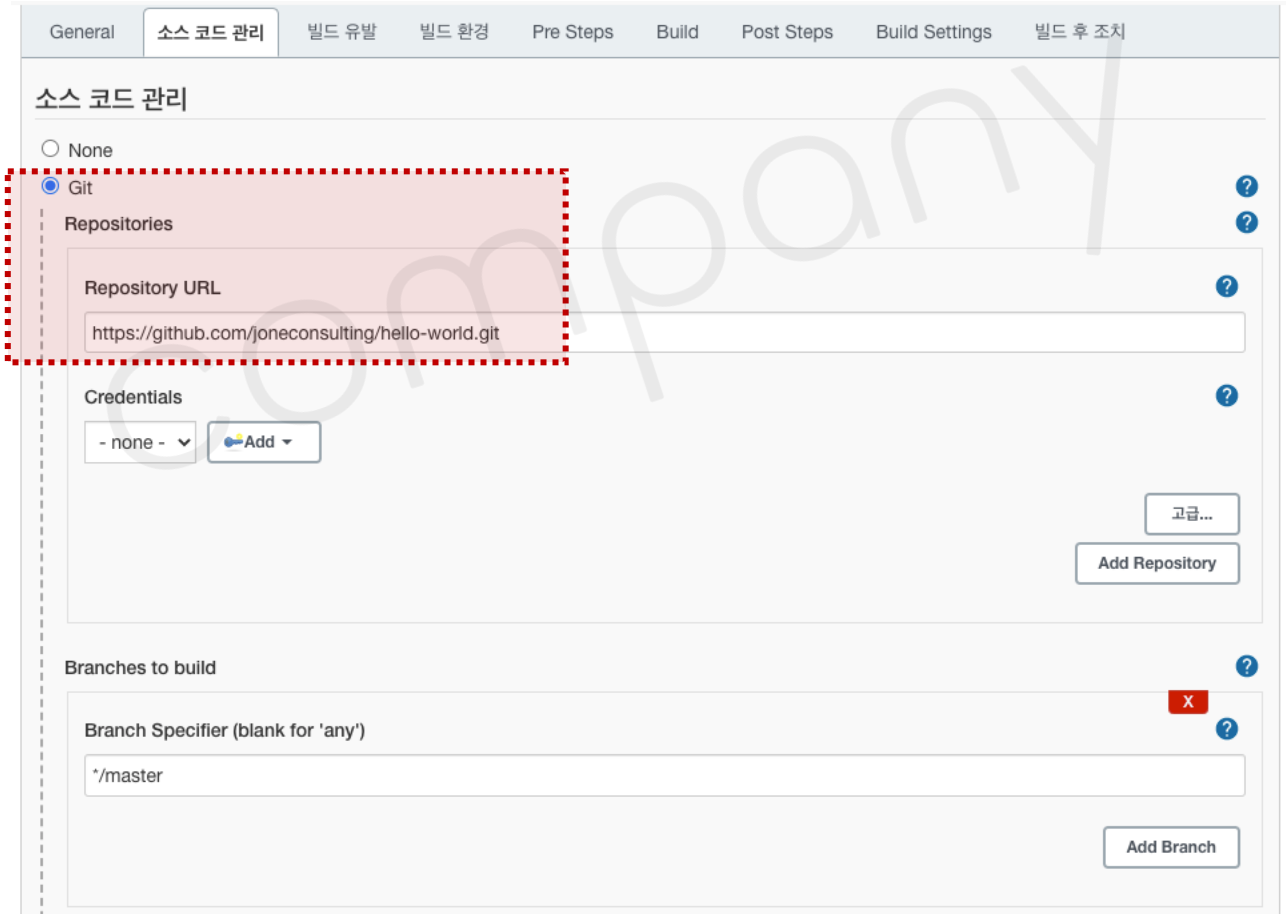
Exercise #2 Jenkins Job 2/4

- General
  - ***My maven project build***



- Source Code Management
  - Repository URL

    https://github.com/joneconsulting/cicd-web-project



Exercise #2 Jenkins Job 3/4

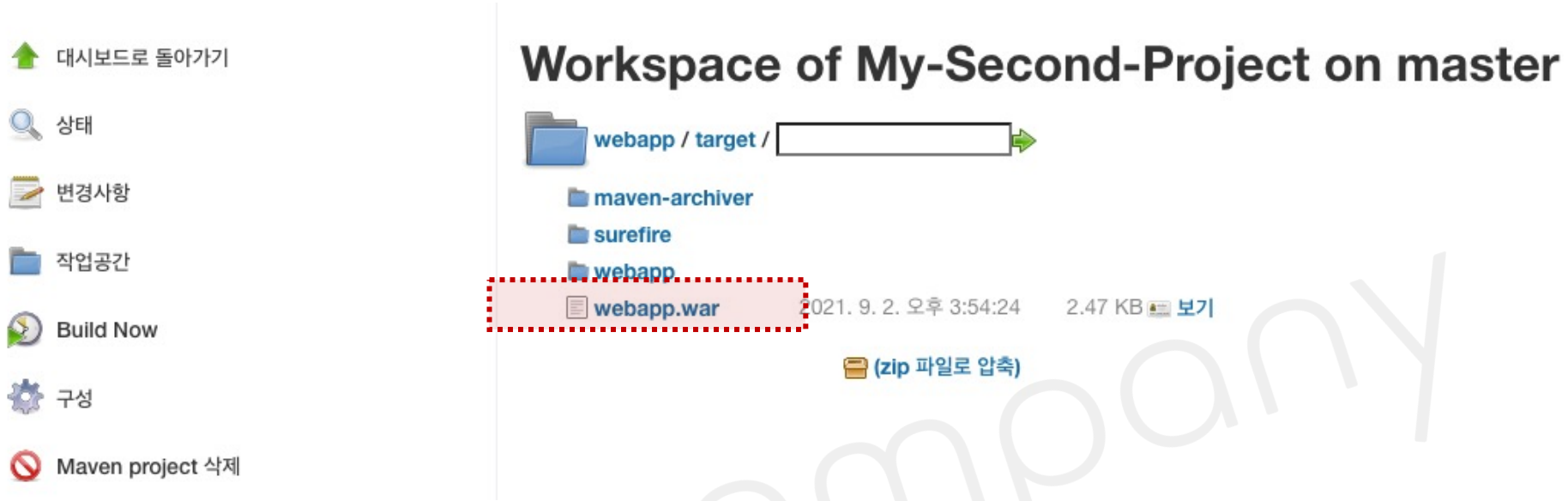
- Build
  - Root POM: ***pom.xml***
  - Golds: ***clean compile package***

- Save > Build Now

```
[INFO] Maven Project ..................................... SUCCESS [  7.118 s]
[INFO] Server ............................................ SUCCESS [ 50.442 s]
[INFO] Webapp ............................................ SUCCESS [  7.227 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  01:06 min
[INFO] Finished at: 2021-09-02T15:54:24Z
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/jenkins_home/workspace/My-Second-Project/webapp/pom.xml to com.example.maven-project/webapp/1.0-SNAPSHOT/webapp-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/jenkins_home/workspace/My-Second-Project/webapp/target/webapp.war to com.example.maven-project/webapp/1.0-SNAPSHOT/webapp-1.0-SNAPSHOT.war
[JENKINS] Archiving /var/jenkins_home/workspace/My-Second-Project/server/pom.xml to com.example.maven-project/server/1.0-SNAPSHOT/server-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/jenkins_home/workspace/My-Second-Project/server/target/server.jar to com.example.maven-project/server/1.0-SNAPSHOT/server-1.0-SNAPSHOT.jar
[JENKINS] Archiving /var/jenkins_home/workspace/My-Second-Project/pom.xml to com.example.maven-project/maven-project/1.0-SNAPSHOT/maven-project-1.0-SNAPSHOT.pom
channel stopped
Finished: SUCCESS
```

Exercise #2 Jenkins Job 4/4

- Workspace



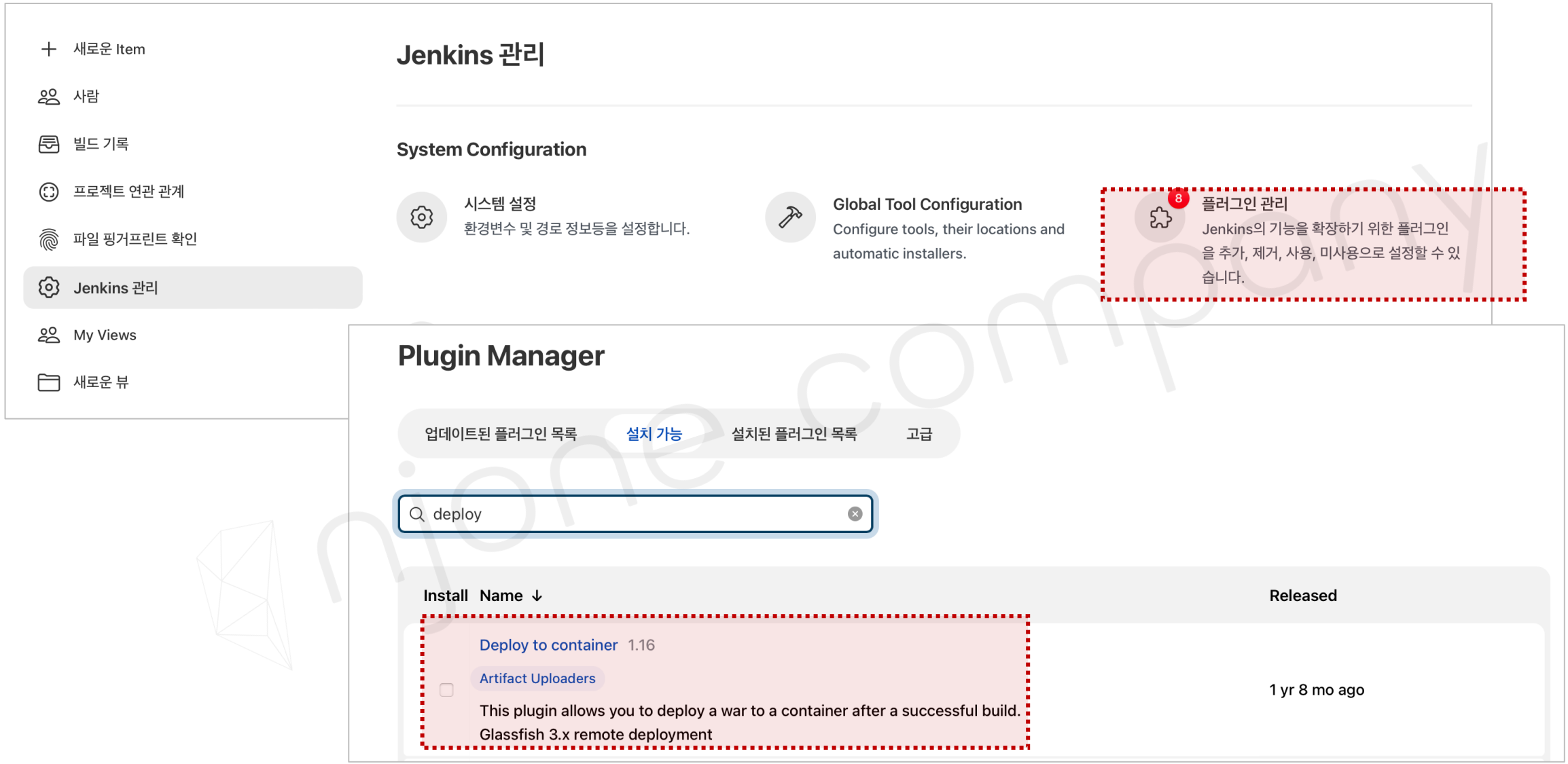
- Jenkins on Docker
  - HOST PC의 VOLUME MOUNT를 사용하지 않고 실행하였을 경우, jenkins_home의 내용을 확인하기 위해서는 jenkins docker container의 terminal로 접속하여 확인

    $ docker container exec -it [container_id or container_name] bash

    root@XXXXXX:~$ cd /var/jenkins_home/workspace

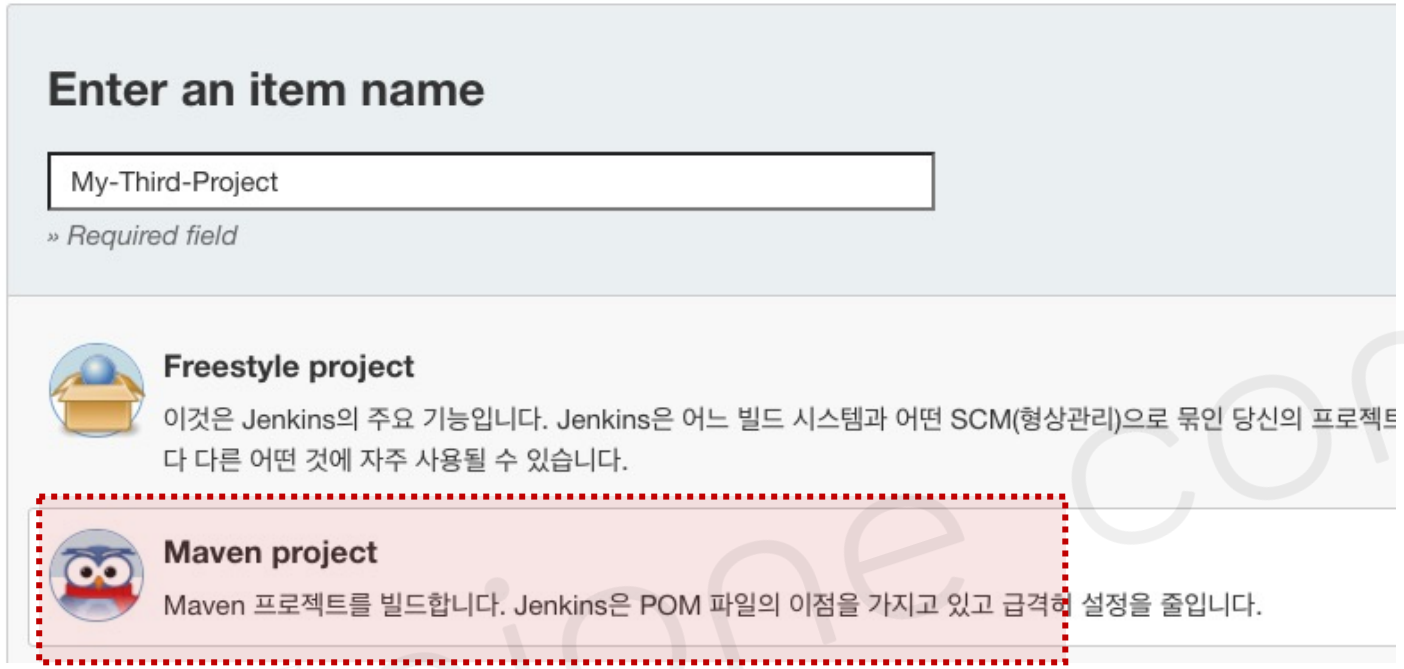
Setup Tomcat Plugin

- Manage Jenkins → Jenkins Plugins → available → deploy to container plugin



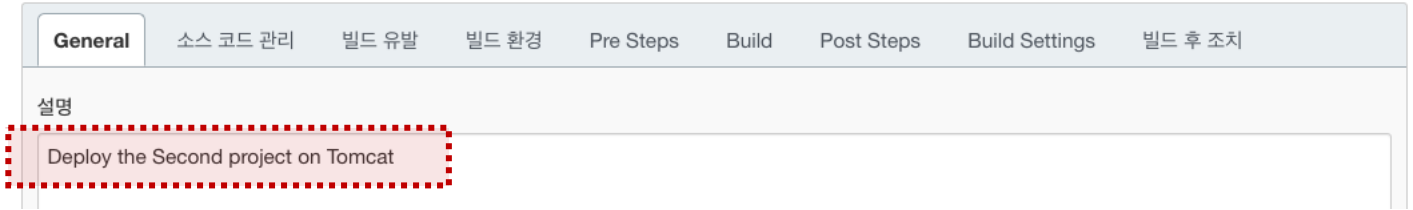
Exercise #3 Jenkins Job 1/7

- Item name → ***My-Third-Project***
  - ***Maven project***



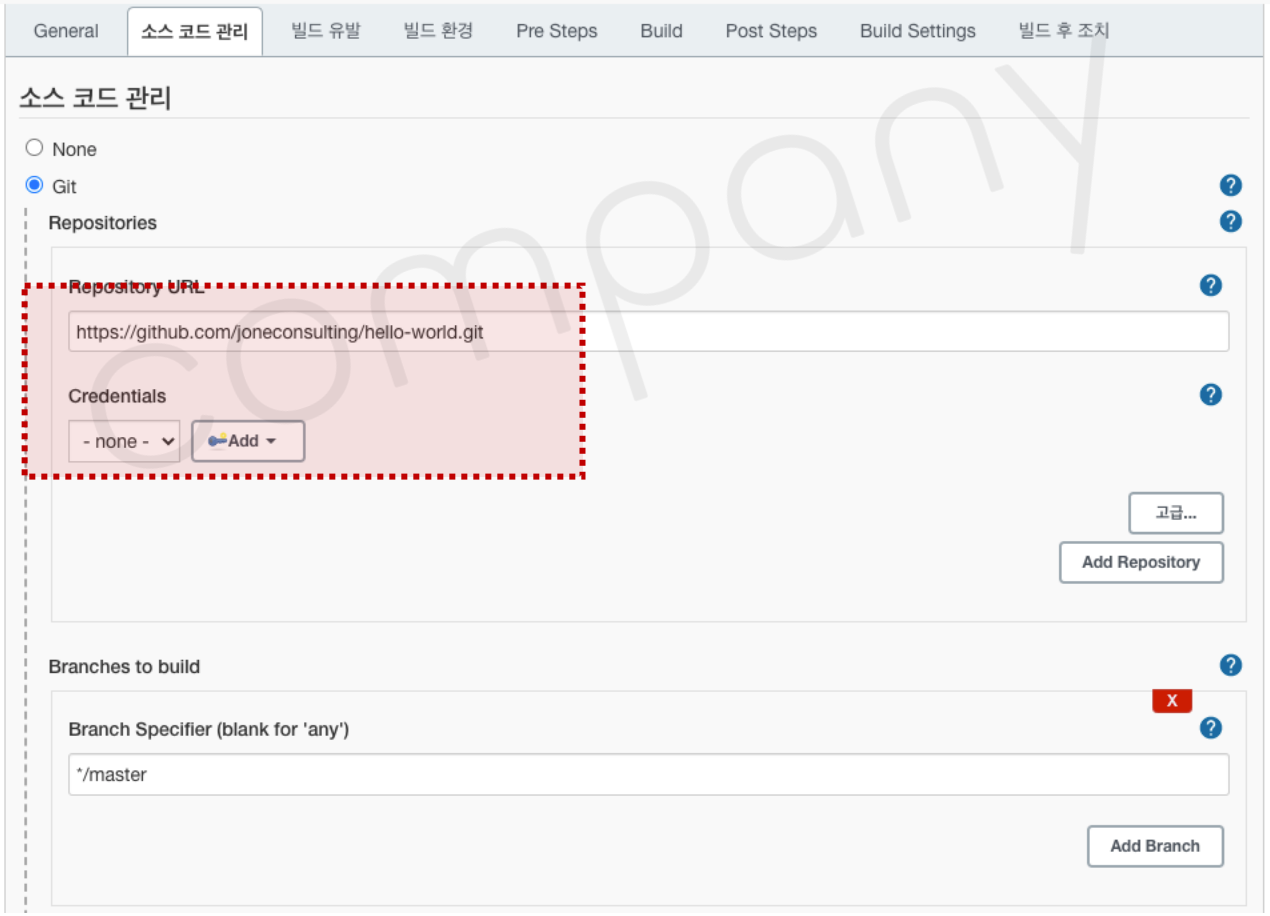
Exercise #3 Jenkins Job 2/7

- General
  - ***Deploy the Second project on Tomcat***



- Source Code Management
  - Repository URL

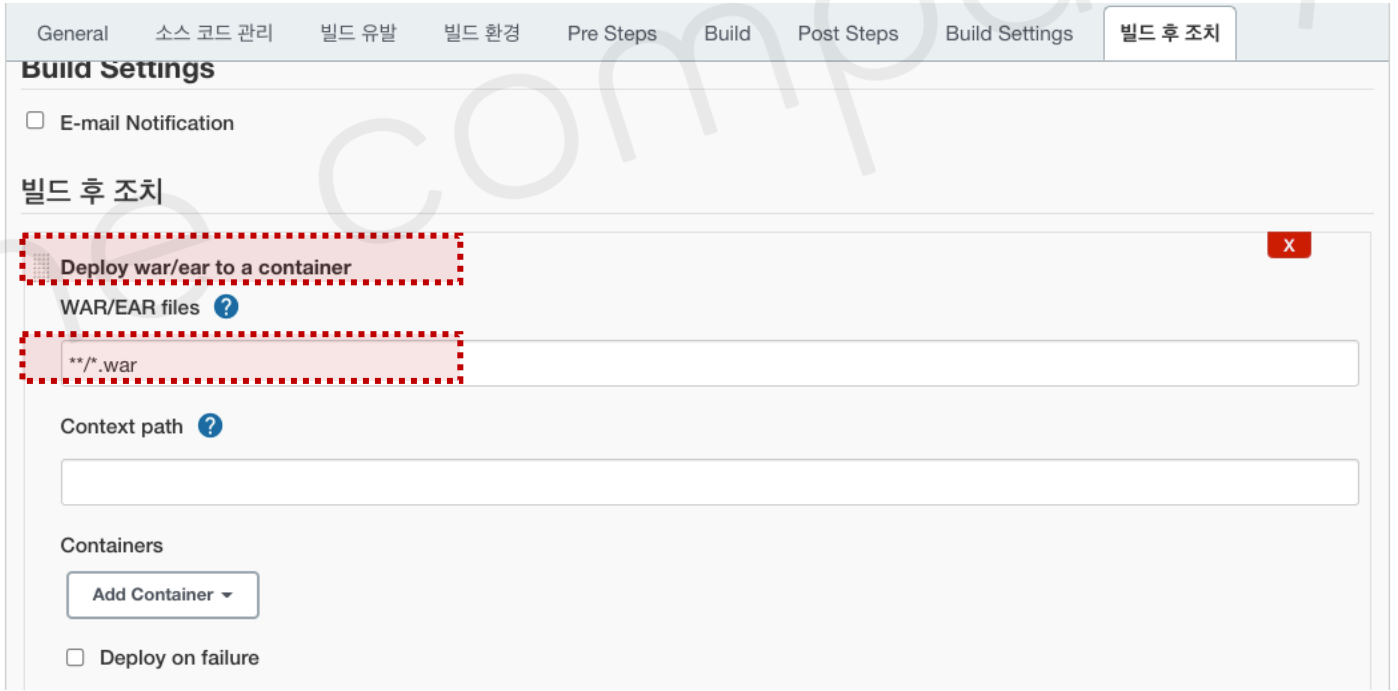
    https://github.com/joneconsulting/cicd-web-project



Exercise #3 Jenkins Job 3/7

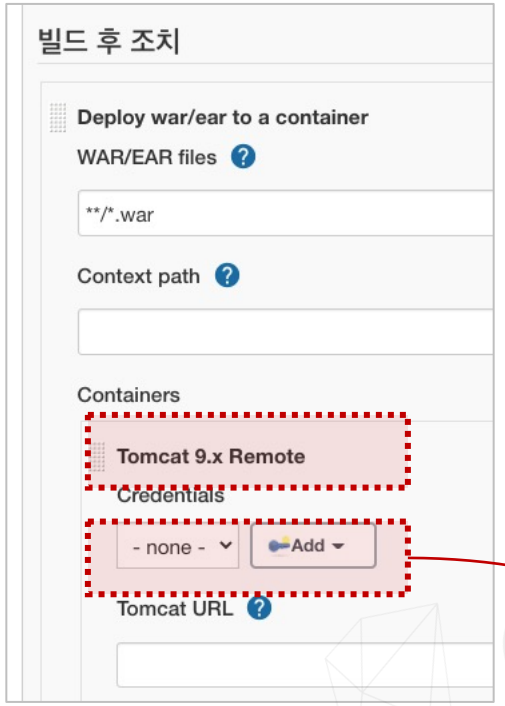
- Build
  - Root POM: ***pom.xml***
  - Golds: ***clean compile package***

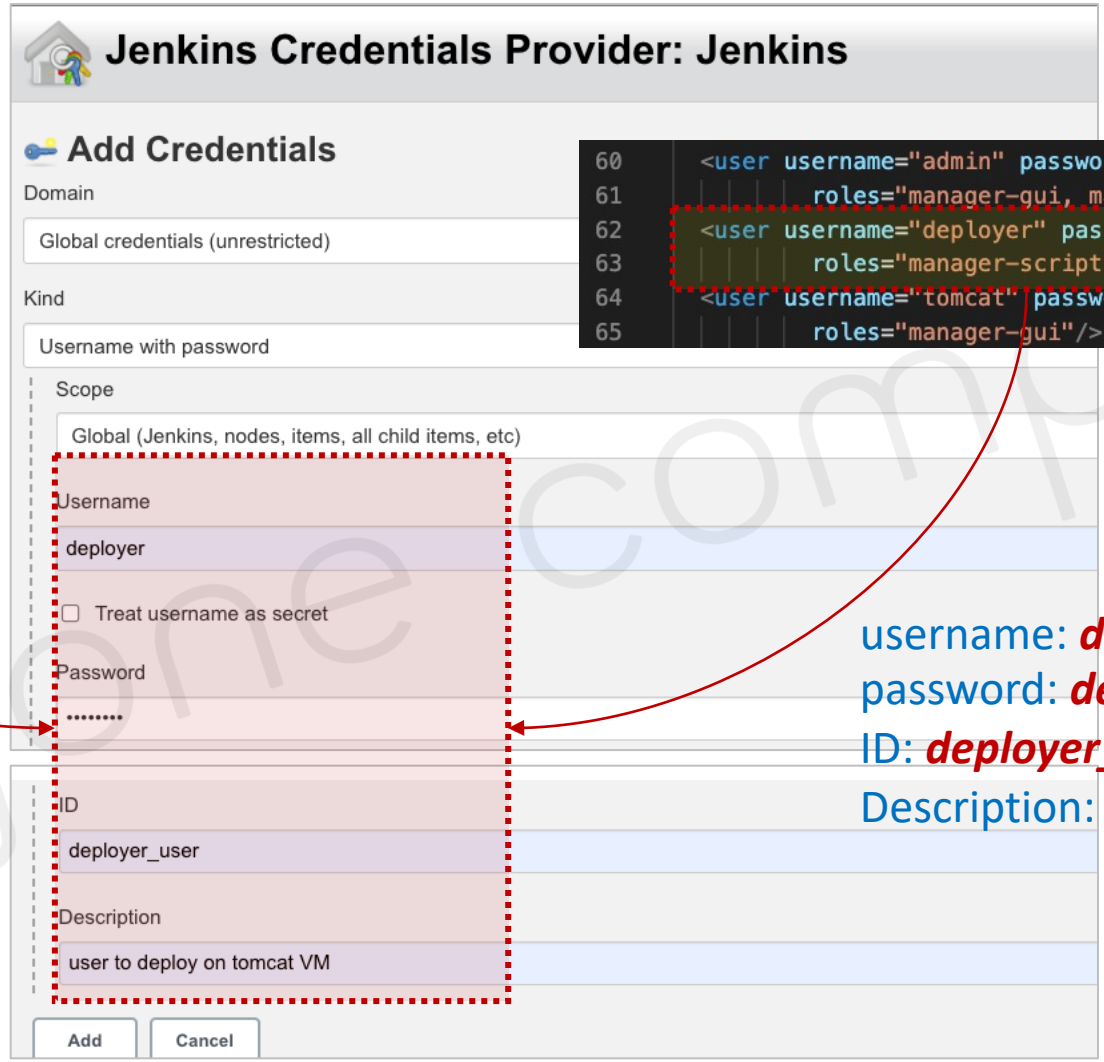
- Post-build Actions
  - ***Deploy war/ear to a container***
  - *****/*.war***



Exercise #3 Jenkins Job 4/7

- Post-build Actions
  - Container: ***Tomcat 9.x Remote***






```
60    <user username="admin" password="admin"
61          roles="manager-gui, manager-script, manager-jmx, manager-status"/>
62    <user username="deployer" password="deployer"
63          roles="manager-script"/>
64    <user username="tomcat" password="tomcat"
65          roles="manager-gui"/>
```

username: ***deployer***
password: ***deployer***
ID: ***deployer_user1***
Description: ***user to deploy on tomcat***

Exercise #3 Jenkins Job 5/7

- Post-build Actions
  - Credentials: ***deployer***
  - Tomcat URL: ***http://192.168.0.8:8080/***

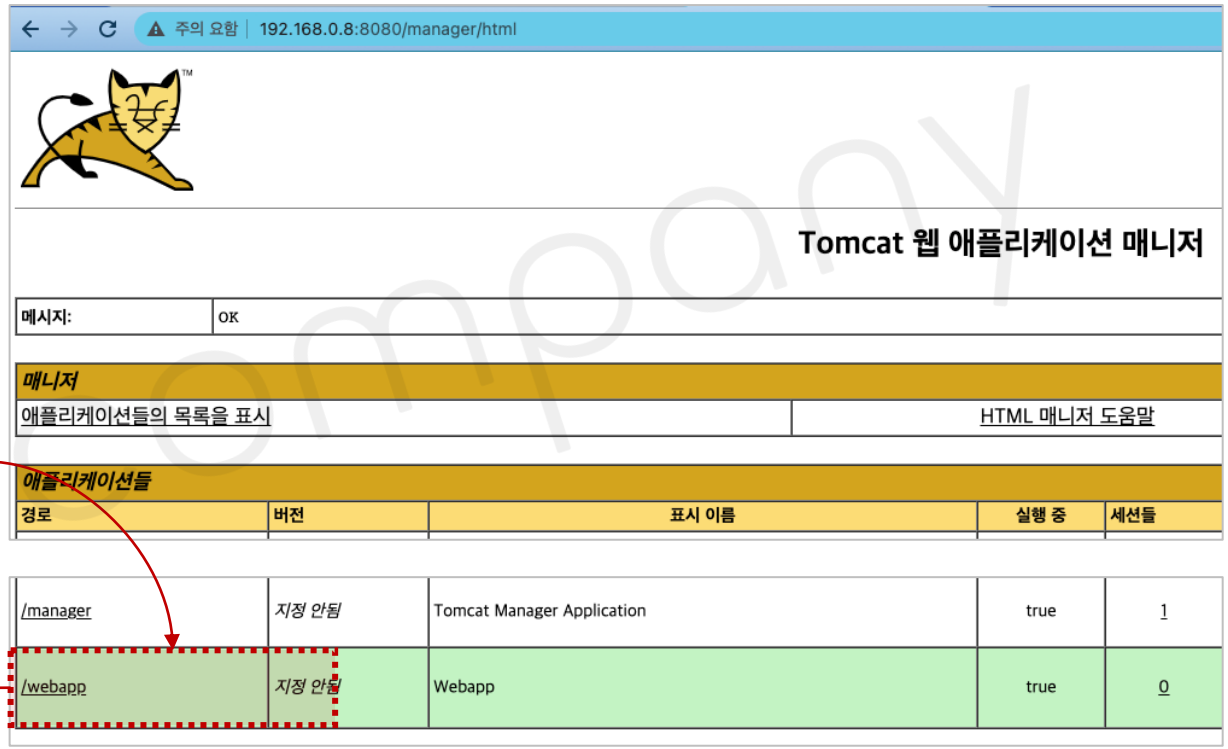
Exercise #3 Jenkins Job 6/7

- Save > Build Now

Exercise #3 Jenkins Job 7/7

- http://192.168.0.8:8080/

```
ls -l ./webapps
total 8
drwxr-xr-x@ 13 dowonlee  staff   416  7 31 06:12 ROOT
drwxr-xr-x@ 60 dowonlee  staff  1920  7 31 06:12 docs
drwxr-xr-x@  8 dowonlee  staff   256  7 31 06:12 examples
drwxr-xr-x@  7 dowonlee  staff   224  7 31 06:12 host-manager
drwxr-xr-x@  9 dowonlee  staff   288  7 31 06:12 manager
drwxr-x---   5 dowonlee  staff   160  9  3 14:10 webapp
-rw-r-----   1 dowonlee  staff  2529  9  3 14:10 webapp.war
```



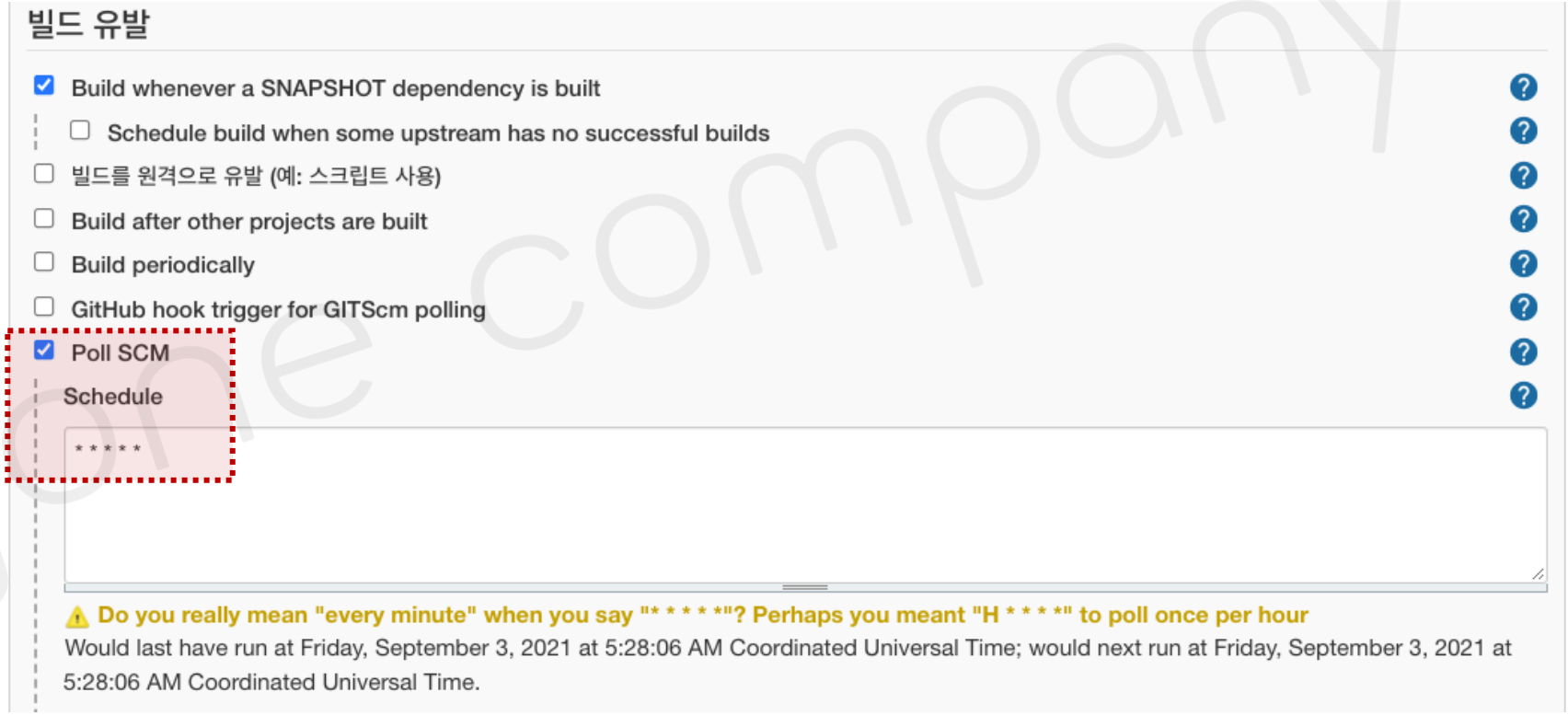
Setup PollSCM

- Project > Configure > Build Triggers
  - ***Build periodically*** → cron job
  - ***Poll SCM*** → cron job

cron

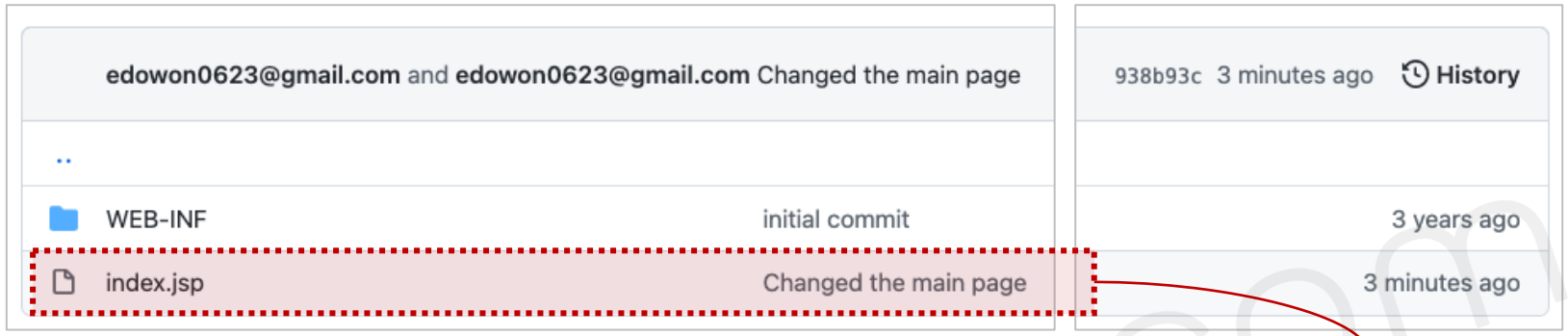
위키백과, 우리 모두의 백과사전.

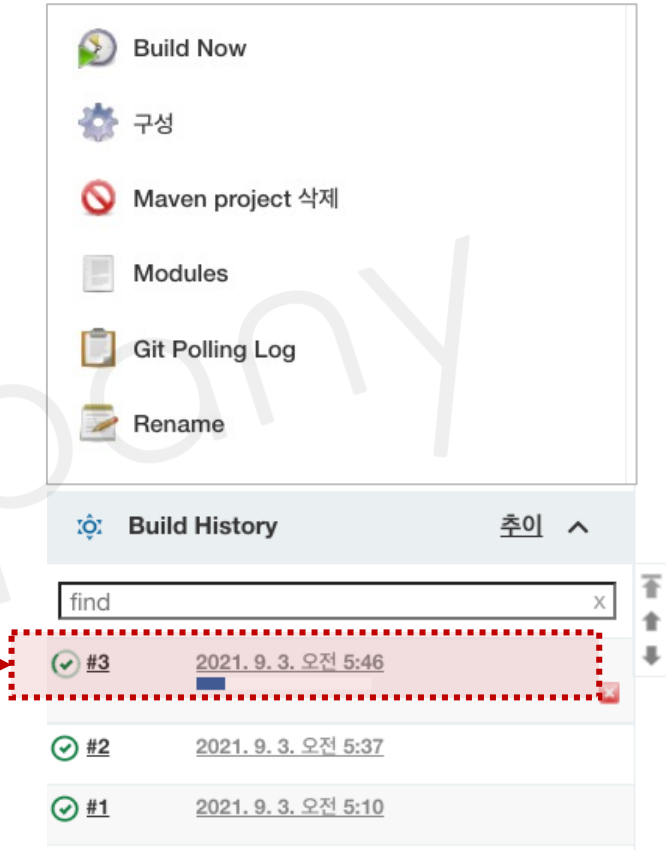
소프트웨어 유틸리티 **cron**은 유닉스 계열 컴퓨터 운영 체제의 시간 기반 잡 스케줄러이다. 소프트웨어 환경을 설정하고 관리하는 사람들은 작업을 고정된 시간, 날짜, 간격에 주기적으로 실행할 수 있도록 스케줄링하기 위해 cron을 사용한다.



Setup PollSCM

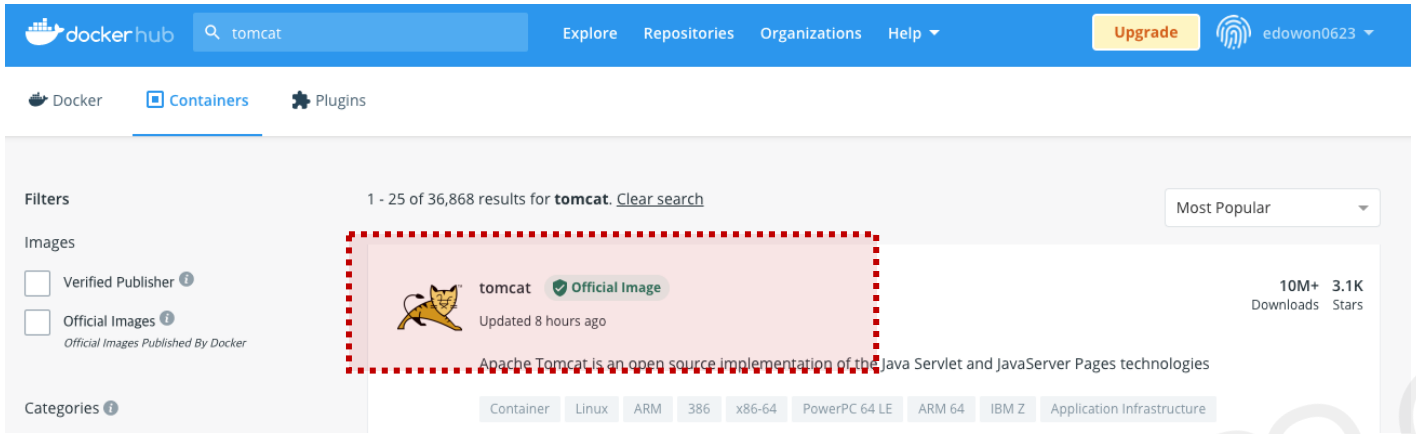
- New commit
  - 코드 수정
  - git add → git commit → git push



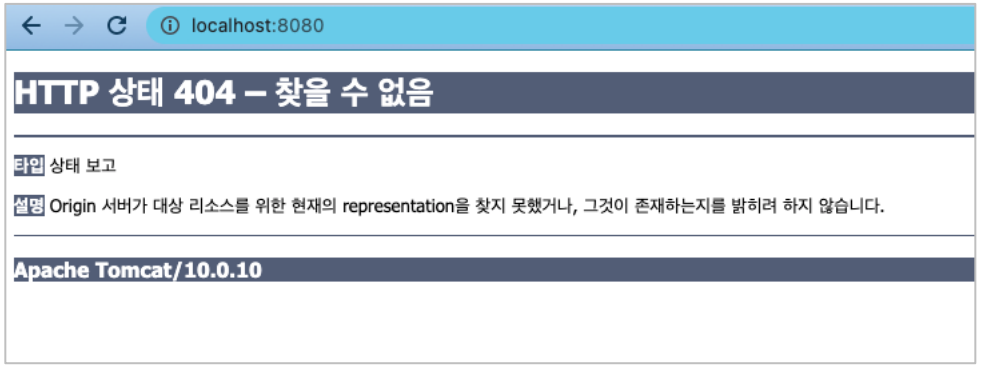


Install Tomcat Container on Docker

- https://hub.docker.com



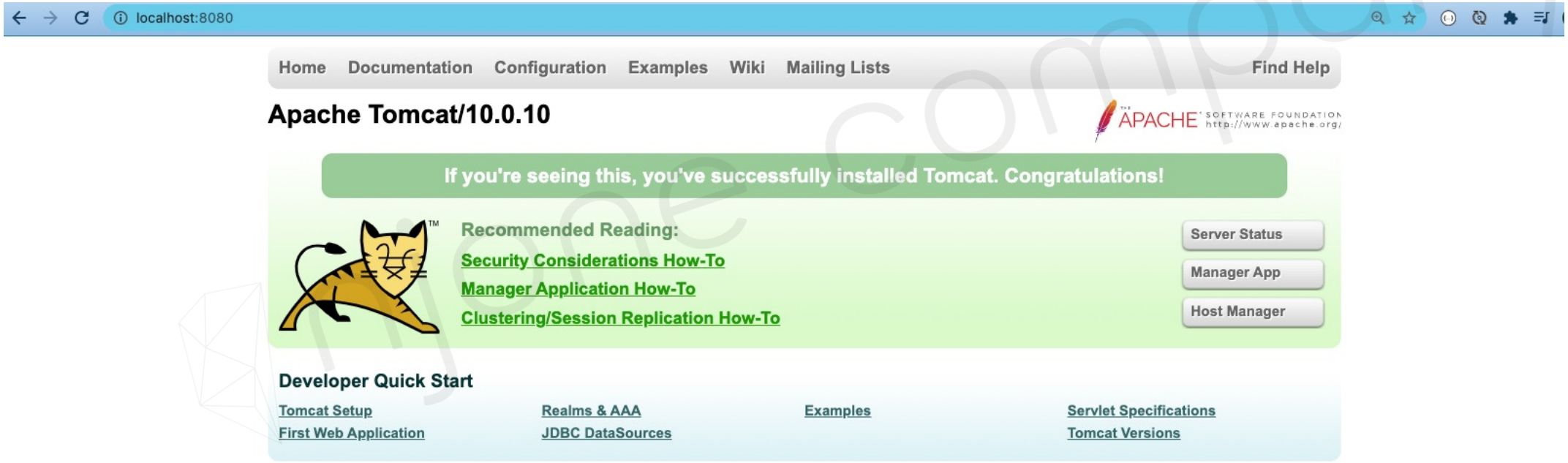
- $ ***docker pull tomcat:latest***
- $ ***docker run -d --name tomcat -p 8080:8080 tomcat:9.0***



Setup Tomcat Container on Docker

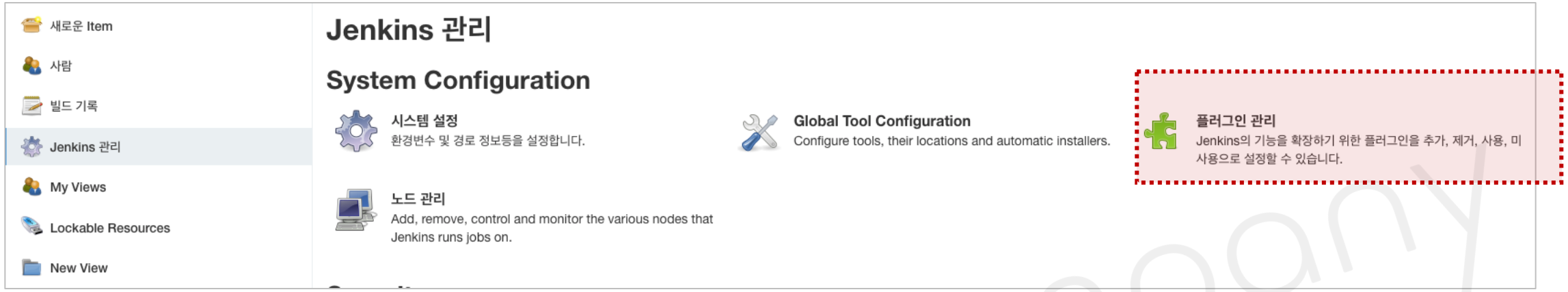
- $ ***docker stop [CONTAINER_ID] && docker rm [CONTAINER_ID]***
- $ ***docker run -d --name tomcat -p 8080:8080 tomcat:9.0***
- $ ***docker exec -it [CONTAINER_ID] bash***

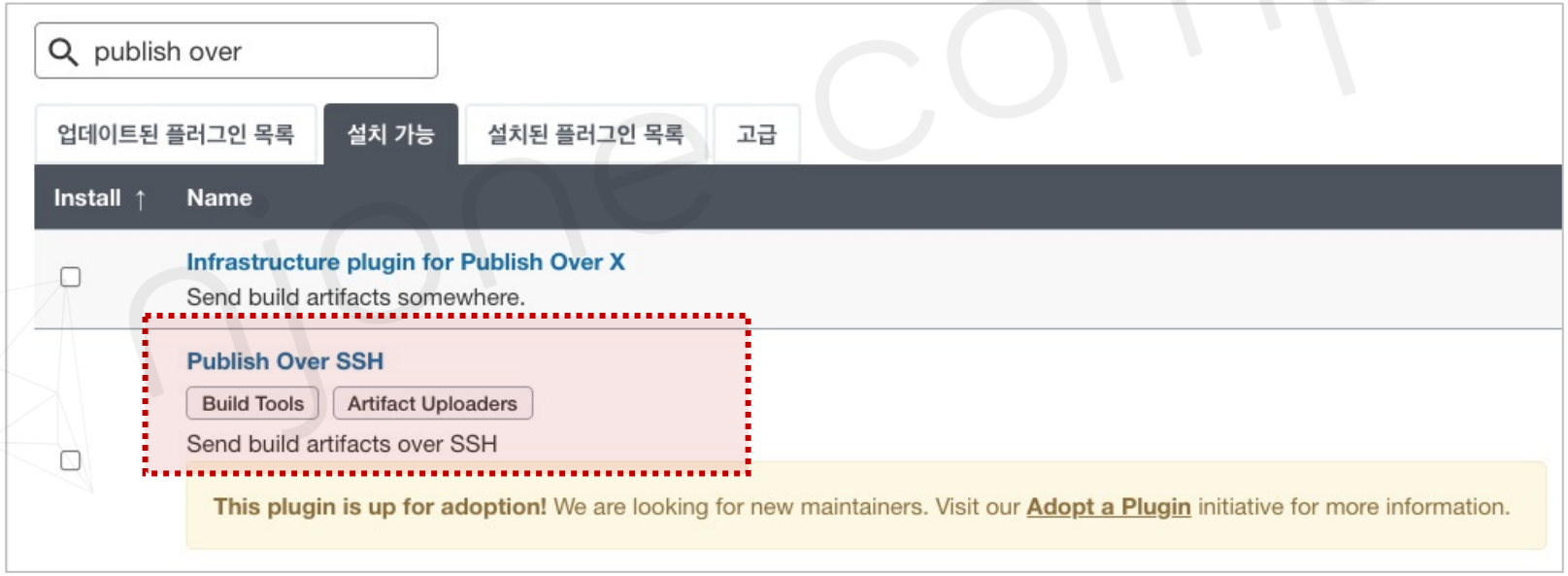
root@[CONTAINER_ID]:/usr/local/tomcat# ***cp -R ./webapps.dist/* ./webapps/***



Setup Publish Over Plugin

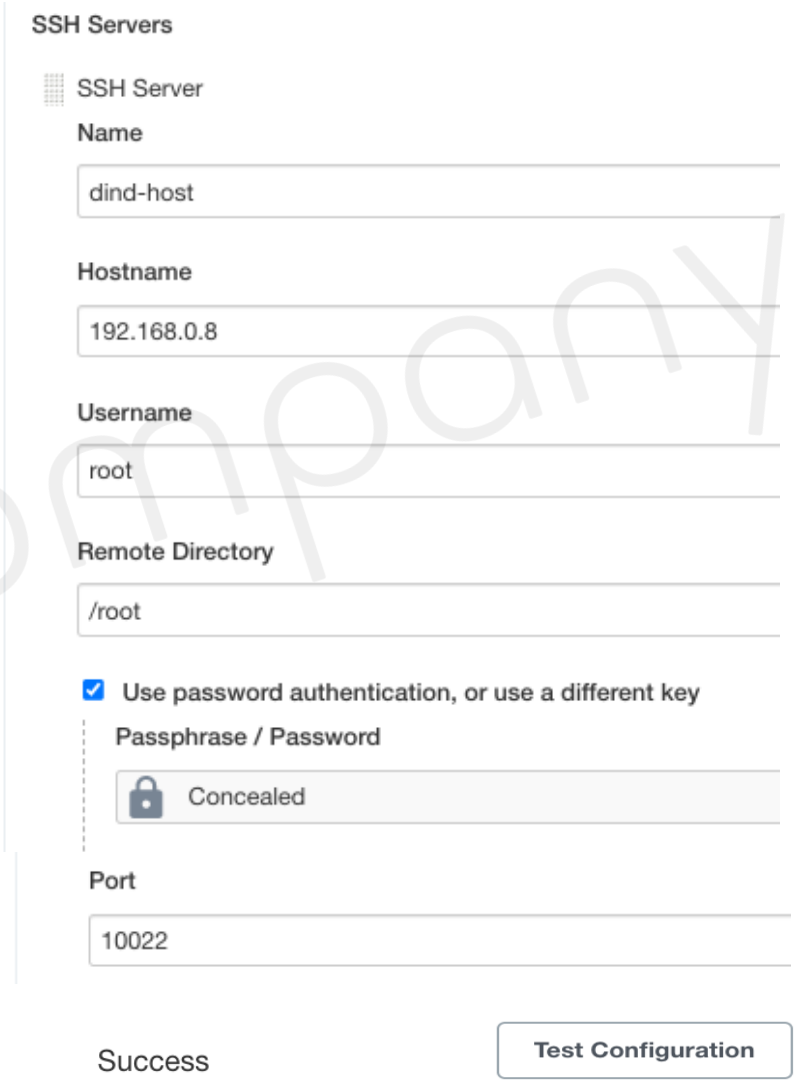
- Manage Jenkins → Jenkins Plugins → available → publish over ssh





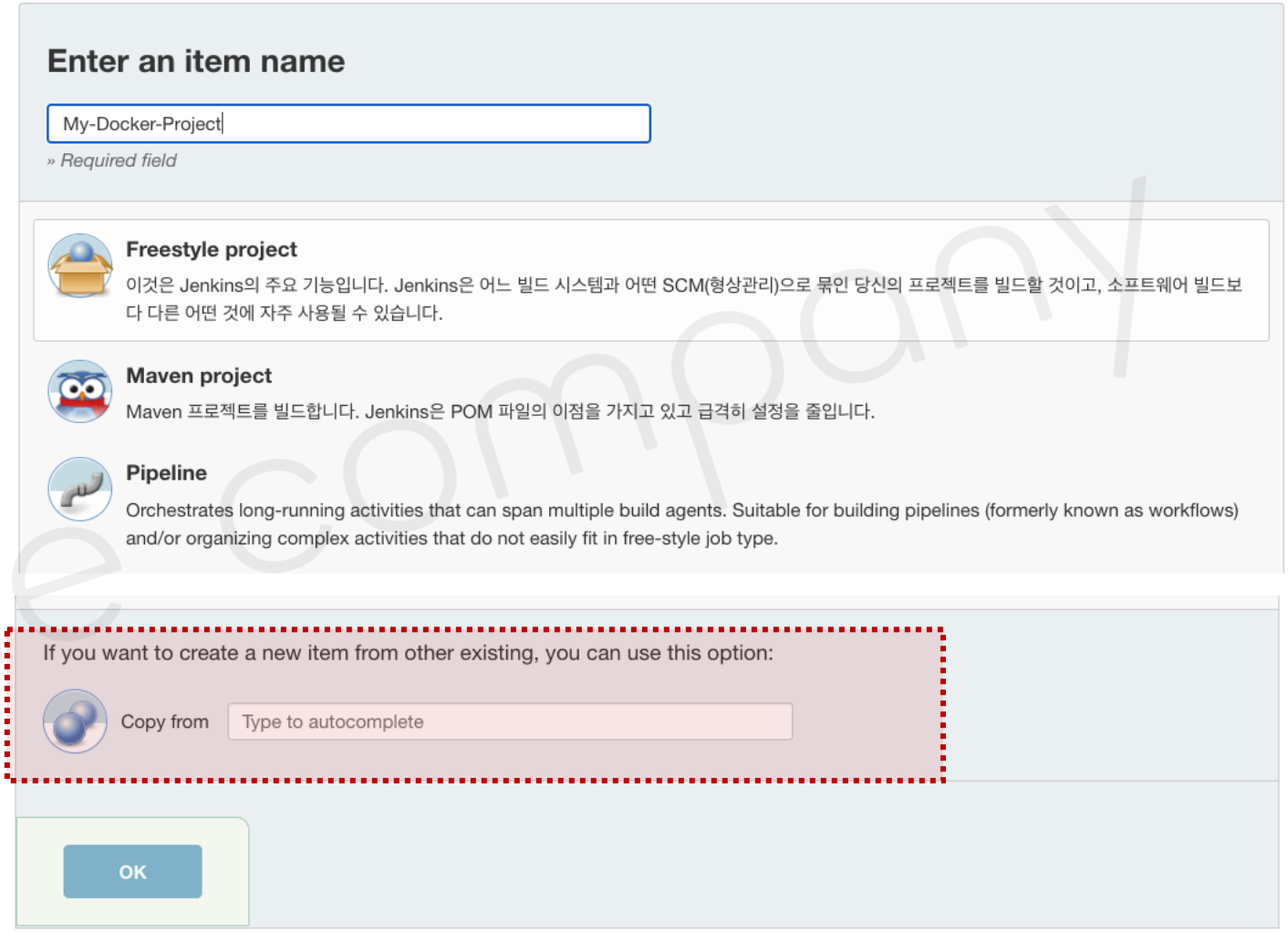
Setup Publish Over Plugin

- Manage Jenkins → Configure System → Publish over SSH
  - Add SSH Servers
    - Name: ***docker-host***
    - Hostname: ***[Remote IP] ex)192.168.0.8***
    - Username: ***root***
    - Passphrase/Password: ***P@ssw0rd***
    - Port: ***10022***
  - Test Configuration



Exercise 4# Jenkins Job 1/8

- Item name → ***My-Docker-Project***
  - Copy from: ***My-Third-Project***



Exercise 4# Jenkins Job 2/8

- Build Triggers
  - Poll SCM → ***Disable***
- Post-build Actions
  - Deploy war/ear to a container → ***Delete***
  - ***Send build artifacts over SSH***
    - ***SSH Server***
      - Name: ***[Publish over SSH에서 설정한 이름] ex) docker-host***
      - ***Transfer Set***
        - Source files: ***target/*.war***
        - Remove prefix: ***target***
        - Remote directory: ***.***

Exercise 4# Jenkins Job 3/8

- Save > Build Now

```
[INFO] Maven Project ..................................... SUCCESS [  0.617 s]
[INFO] Server ............................................ SUCCESS [  3.748 s]
[INFO] Webapp ............................................ SUCCESS [  1.167 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  7.019 s
[INFO] Finished at: 2021-09-03T13:26:06Z
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/webapp/pom.xml to com.example.maven-project/webapp/1.0-SNAPSHOT/webapp-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/webapp/target/webapp.war to com.example.maven-project/webapp/1.0-SNAPSHOT/webapp-1.0-
SNAPSHOT.war
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/server/pom.xml to com.example.maven-project/server/1.0-SNAPSHOT/server-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/server/target/server.jar to com.example.maven-project/server/1.0-SNAPSHOT/server-1.0-
SNAPSHOT.jar
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/pom.xml to com.example.maven-project/maven-project/1.0-SNAPSHOT/maven-project-1.0-SNAPSHOT.pom
channel stopped
SSH: Connecting from host [512076bd5890]
SSH: Connecting with configuration [docker-host] ...
SSH: Disconnecting configuration [docker-host] ...
SSH: Transferred 1 file(s)
Finished: SUCCESS
```

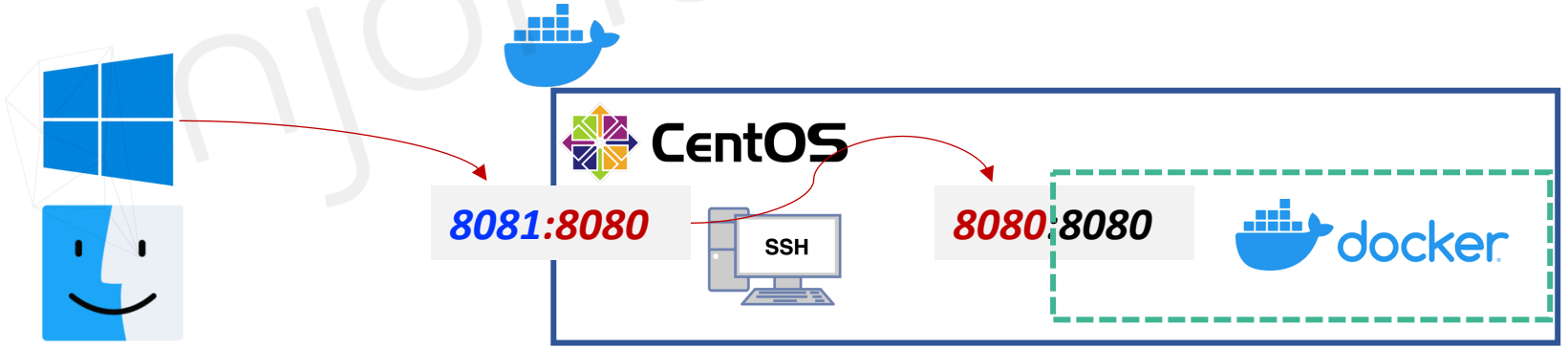
```
16b4daa4cf99:~# ls -l
total 8
-rw-r--r--    1 root     root           127 Sep  3 14:37 Dockerfile
-rw-r--r--    1 root     root          2443 Sep  5 14:04 webapp.war
```

Exercise 4# Jenkins Job 4/8

- Create a Dockerfile on Docker server

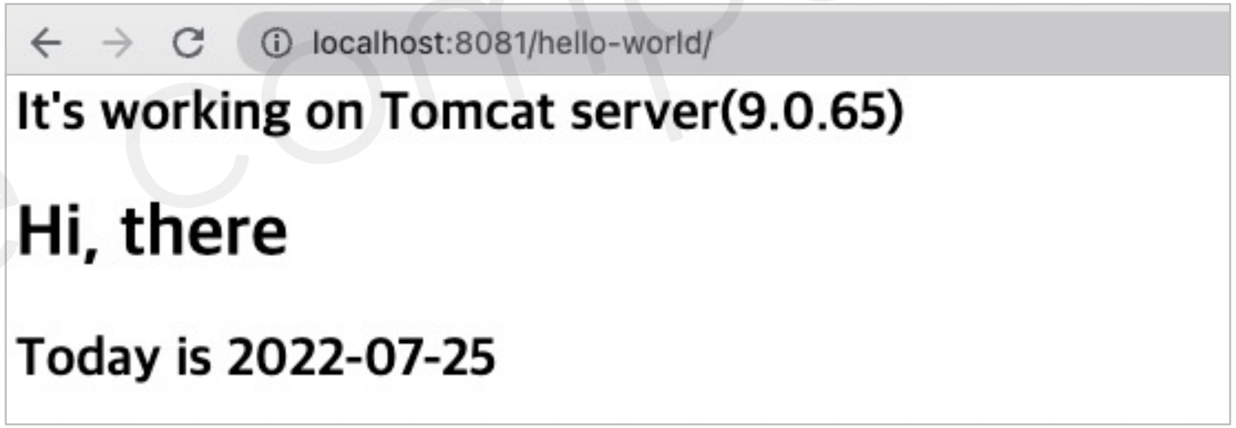
```
1   FROM tomcat:9.0
2
3   COPY ./hello-world.war /usr/local/tomcat/webapps
```

- $ ***docker build --tag=cicd-project -f Dockerfile .***
- $ ***docker images***
- $ ***docker image inspect cicd-project:latest***
- $ ***docker run -p 8080:8080 --name mytomcat cicd-project:latest***



Exercise 4# Jenkins Job 5/8


```
docker run -d -p 8081:8080 --name mytomcat cicd-project:latest
adecc32eb40825fff90de64a037792a1017b2fa157d1f3962bdb2551f826b597
(base)  dowonlee  ~/Desktop/Work/14.online/devops
docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED          STATUS          PORTS
                                                  NAMES
adecc32eb408   cicd-project:latest          "catalina.sh run"        3 seconds ago    Up 2 seconds    0.0.0.0:8081->8080/tcp, :::8081->8080/tcp
                                                  mytomcat
512076bd5890   jenkins/jenkins:lts-jdk11    "/sbin/tini -- /usr/…"   7 hours ago      Up 7 hours      0.0.0.0:50000->50000/tcp, :::50000->50000/
tcp, 0.0.0.0:8088->8080/tcp, :::8088->8080/tcp   nervous_wing
f48bfb0d1968   tomcat:latest                "catalina.sh run"        7 hours ago      Up 7 hours      0.0.0.0:8080->8080/tcp, :::8080->8080/tcp
                                                  tomcat
```



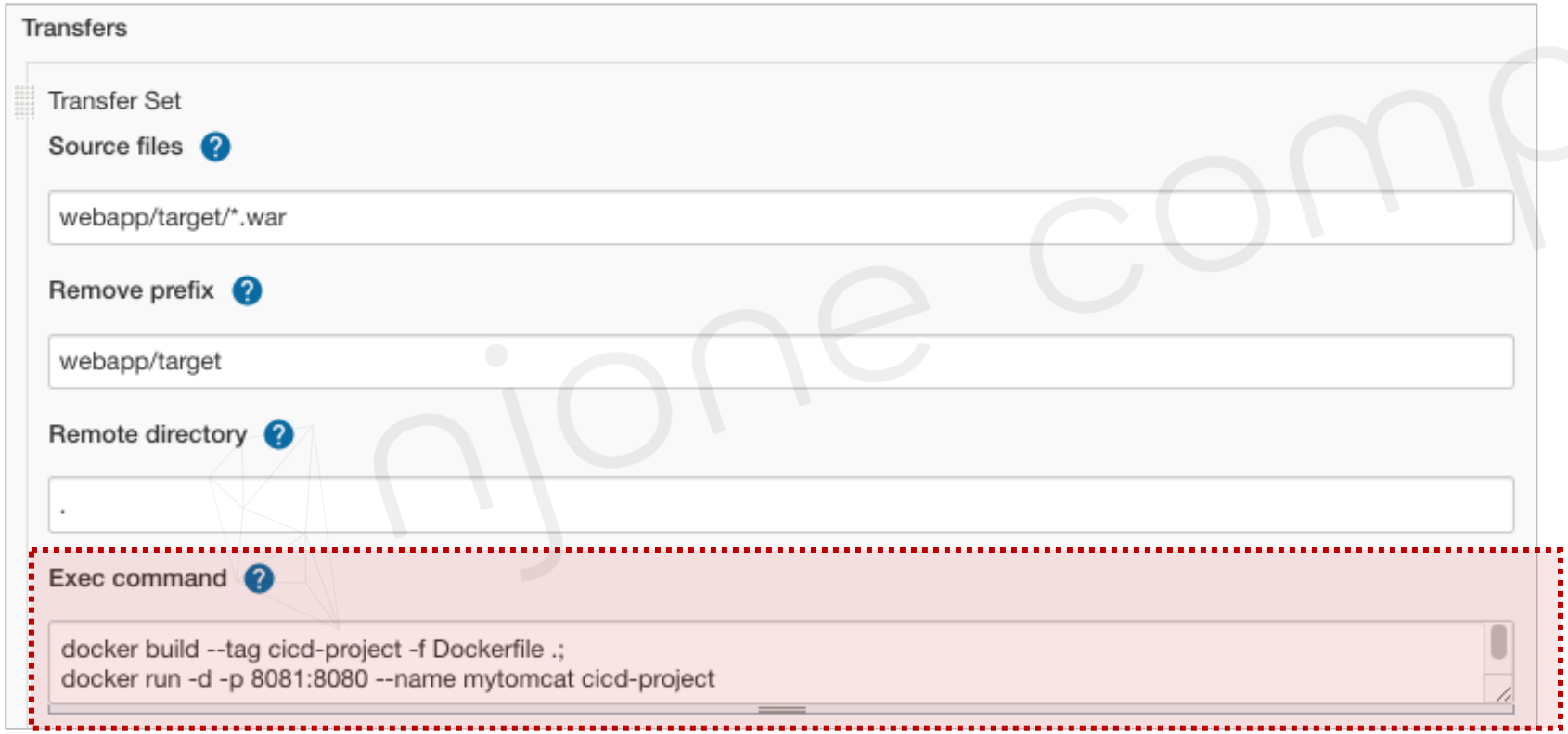
- $ ***docker stop mytomcat***
- $ ***docker rm mytomcat***
- $ ***docker rmi cicd-project:latest***
- $ ***rm -rf webapp.war***

Exercise 4# Jenkins Job 6/8

- Project > Configure > Post Build Actions
  - ***Exec command***

    docker build --tag=cicd-project -f Dockerfile .;

    docker run -d -p 8080:8080 --name mytomcat cicd-project:latest



Exercise 4# Jenkins Job 7/8

- Save > Build Now

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  6.874 s
[INFO] Finished at: 2021-09-03T14:54:14Z
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/webapp/pom.xml to com.example.ma
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/webapp/target/webapp.war to com.
SNAPSHOT.war
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/server/pom.xml to com.example.ma
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/server/target/server.jar to com.
SNAPSHOT.jar
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/pom.xml to com.example.maven-pro
SNAPSHOT.pom
channel stopped
SSH: Connecting from host [512076bd5890]
```

```
[root@ae2b6f08d2dc ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
cicd-project        latest              362a7c69e220        53 minutes ago      680 MB
docker.io/tomcat    latest              ab1f0e1bb1a1        41 hours ago        680 MB
[root@ae2b6f08d2dc ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                    NAMES
e6011dc8d047        cicd-project        "catalina.sh run"   53 minutes ago      Up 53 minutes       0.0.0.0:8081->8080/tcp   mytomcat
```

Exercise 4# Jenkins Job 8/8

- Build Now (after changed sources)

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  7.138 s
[INFO] Finished at: 2021-09-03T14:57:49Z
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/webapp/pom.xml to com.example.maven-project/webapp/1.0-SNAPS
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/webapp/target/webapp.war to com.example.maven-project/webapp
SNAPSHOT.war
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/server/pom.xml to com.example.maven-project/server/1.0-SNAPS
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/server/target/server.jar to com.example.maven-project/server
SNAPSHOT.jar
[JENKINS] Archiving /var/jenkins_home/workspace/My-Docker-Project/pom.xml to com.example.maven-project/maven-project/1.0-SNAPS
SNAPSHOT.pom
channel stopped
SSH: Connecting from host [512076bd5890]
SSH: Connecting with configuration [docker-host] ...
SSH: EXEC: completed after 1,005 ms
SSH: Disconnecting configuration [docker-host] ...
ERROR: Exception when publishing, exception message [Exec exit status not zero. Status [125]]
Build step 'Send build artifacts over SSH' changed build result to UNSTABLE
Finished: UNSTABLE
```

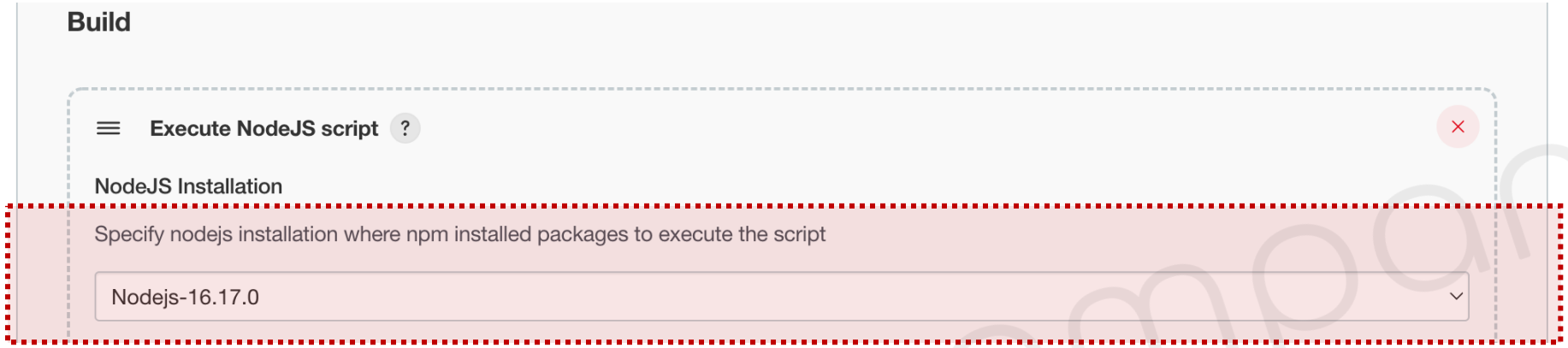
Exercise 4-2# Jenkins Job 1/5

- Jenkins Server) Nodejs 설치
  - Plugin Manager > Available > nodejs

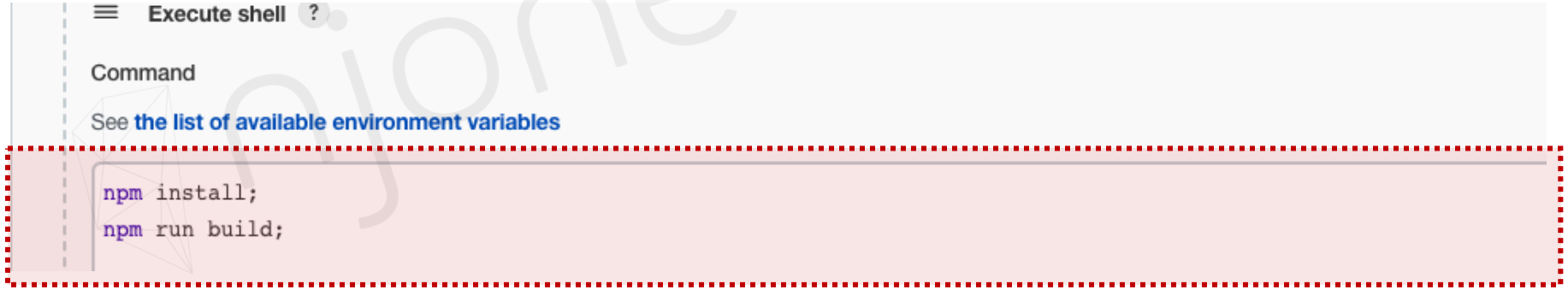
  - Global Tool Configuration > NodeJS 설치

Exercise 4-2# Jenkins Job 2/5

- Item name → ***My-Nodejs-Project***
  - Build > Execute NodeJS script > ***Nodejs-16.17.0 선택***



  - Build > Execute shell



Exercise 4-2# Jenkins Job 3/5

- Save > Build Now

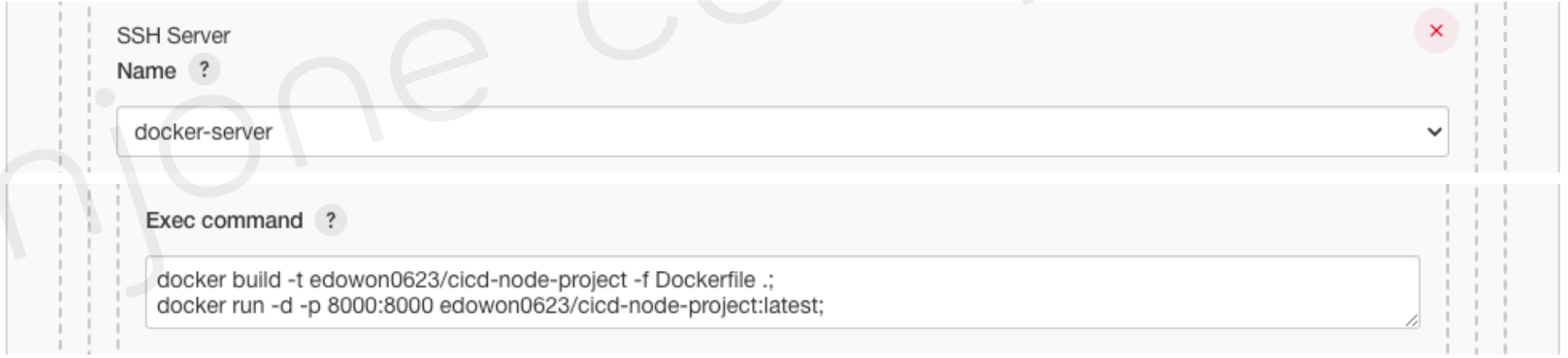
```
Commit message: "upload project files"
 > git rev-list --no-walk c9c23da81a9b98a5a53f5d8122086b035a0e7f3b # timeout=10
[My-Nodejs-Project] $ /var/jenkins_home/tools/jenkins.plugins.nodejs.tools.NodeJSInstallation/Nodejs-16.17.0/bin/node /tmp/jenkins14481587423122626179.js
[My-Nodejs-Project] $ /bin/sh -xe /tmp/jenkins16441738717920205999.sh
+ npm install

added 309 packages, and audited 310 packages in 2s

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

found 0 vulnerabilities
+ npm run build

> example@1.0.0 build
> babel app.js -d dist

Successfully compiled 1 file with Babel (148ms).
Finished: SUCCESS
```

```
jenkins@e1253d619fc1:~/workspace/My-Nodejs-Project$ ls -al
total 360
drwxr-xr-x   5 jenkins jenkins   4096 Sep 18 15:32 .
drwxr-xr-x  11 jenkins jenkins   4096 Sep 16 04:28 ..
drwxr-xr-x   8 jenkins jenkins   4096 Sep 18 15:32 .git
-rw-r--r--   1 jenkins jenkins    478 Sep 16 04:28 Dockerfile
-rw-r--r--   1 jenkins jenkins    252 Sep 16 04:28 app.js
-rw-r--r--   1 jenkins jenkins     42 Sep 16 04:28 babel.config.json
drwxr-xr-x   2 jenkins jenkins   4096 Sep 18 15:32 dist
-rw-r--r--   1 jenkins jenkins     92 Sep 16 04:28 ecosystem.config.js
drwxr-xr-x 190 jenkins jenkins  12288 Sep 18 15:32 node_modules
-rw-r--r--   1 jenkins jenkins 317553 Sep 18 15:32 package-lock.json
-rw-r--r--   1 jenkins jenkins    431 Sep 16 04:28 package.json
jenkins@e1253d619fc1:~/workspace/My-Nodejs-Project$
```

Exercise 4-2# Jenkins Job 4/5

- Delete → Build > Execute NodeJS script, Execute Shell
- Post Build > Send build artifacts over SSH → ***docker-server***
  - Source files: **
  - Remote directory: **.**
  - Exec command:

    docker build -t edowon0623/cicd-node-project -f Dockerfile . ;
    docker run -d -p 8000:8000 edowon0623/cicd-node-project:latest;



Exercise 4-2# Jenkins Job 5/5

- Save > Build Now

```
jenkins@e1253d619fc1:~/workspace/My-Nodejs-Project$ ls -al
total 32
drwxr-xr-x  3 jenkins jenkins 4096 Sep 19 02:07 .
drwxr-xr-x 11 jenkins jenkins 4096 Sep 16 04:28 ..
drwxr-xr-x  8 jenkins jenkins 4096 Sep 19 03:23 .git
-rw-r--r--  1 jenkins jenkins  271 Sep 19 02:07 Dockerfile
-rw-r--r--  1 jenkins jenkins  252 Sep 19 02:07 app.js
-rw-r--r--  1 jenkins jenkins   42 Sep 19 02:07 babel.config.json
-rw-r--r--  1 jenkins jenkins   92 Sep 19 02:07 ecosystem.config.js
-rw-r--r--  1 jenkins jenkins  431 Sep 19 02:07 package.json
jenkins@e1253d619fc1:~/workspace/My-Nodejs-Project$
```

```
[root@3bd341fe9bdd ~]# docker images
REPOSITORY                      TAG       IMAGE ID       CREATED          SIZE
edowon0623/cicd-node-project    latest    21109ecd9821   31 minutes ago   934MB
tomcat                          latest    d9c12d68a96f   2 weeks ago      469MB
node                            10        8951689bc94c   17 months ago    859MB
[root@3bd341fe9bdd ~]# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS
       PORTS                     NAMES
a31278417596   edowon0623/cicd-node-project:latest   "docker-entrypoint.s…"   32 minutes ago   Up 32 min
utes   0.0.0.0:8000->8000/tcp   zealous_tharp
```

Section 3.

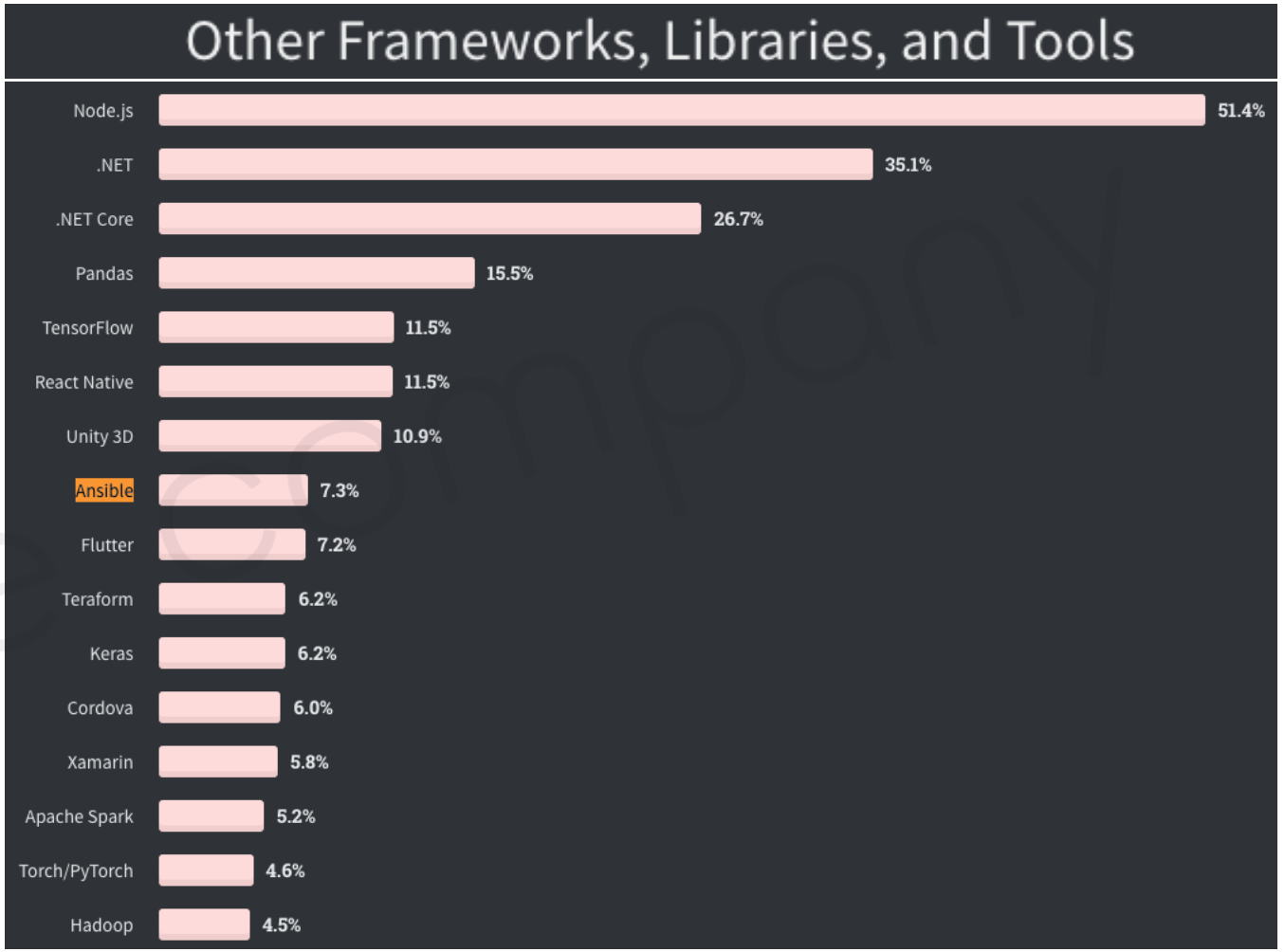
Jenkins + Infrastructure as Code

- IaC 개요
- Ansible 개요
- Ansible 설치
- Ansible Playbook
- Docker 이미지 배포

DevOps Tools

https://devops.com/11-open-source-devops-tools-we-love-for-2021

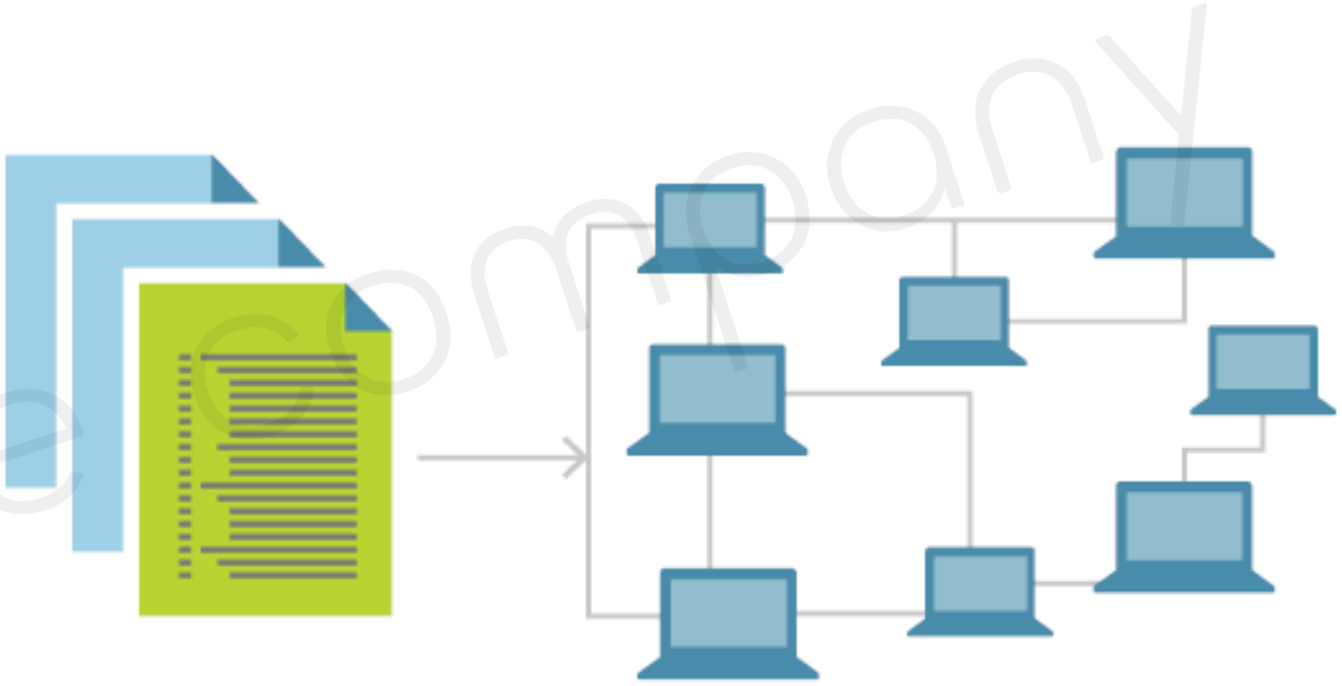
1. Kubernetes
2. Docker
3. Istio
4. GitHub Actions
5. Jenkins
6. Prometheus
7. Ansible
8. Chef
9. Terraform
10. JAMStack
11. ELK Stack



https://insights.stackoverflow.com/survey/2020#technology-other-frameworks-libraries-and-tools-all-respondents3

Infrastructure as Code

- 시스템, 하드웨어 또는 인터페이스의 구성정보를 파일(스크립트)을 통해 관리 및 프로비저닝
- IT 인프라스트럭처, 베어 메탈 서버 등의 물리 장비 및 가상 머신과 관련된 구성 리소스를 관리
- 버전 관리를 통한 리소스 관리



Terraform

ANSIBLE

AWS CloudFormation

CHEF

puppet

https://docs.microsoft.com/ko-kr/devops/deliver/what-is-infrastructure-as-code

Infrastructure as Code

terraform
검색어

ansible
검색어

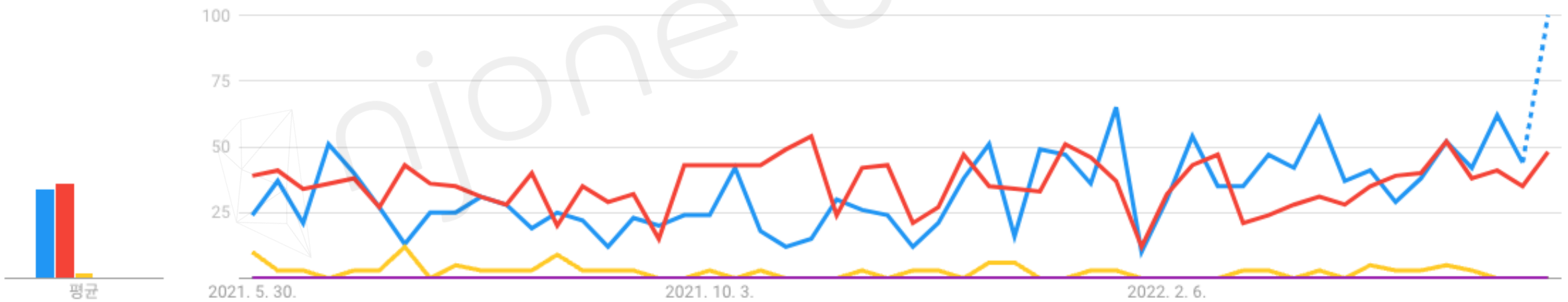
aws cloudformation
검색어

Chef infrastructure
검색어

Puppet infrastruct...
검색어

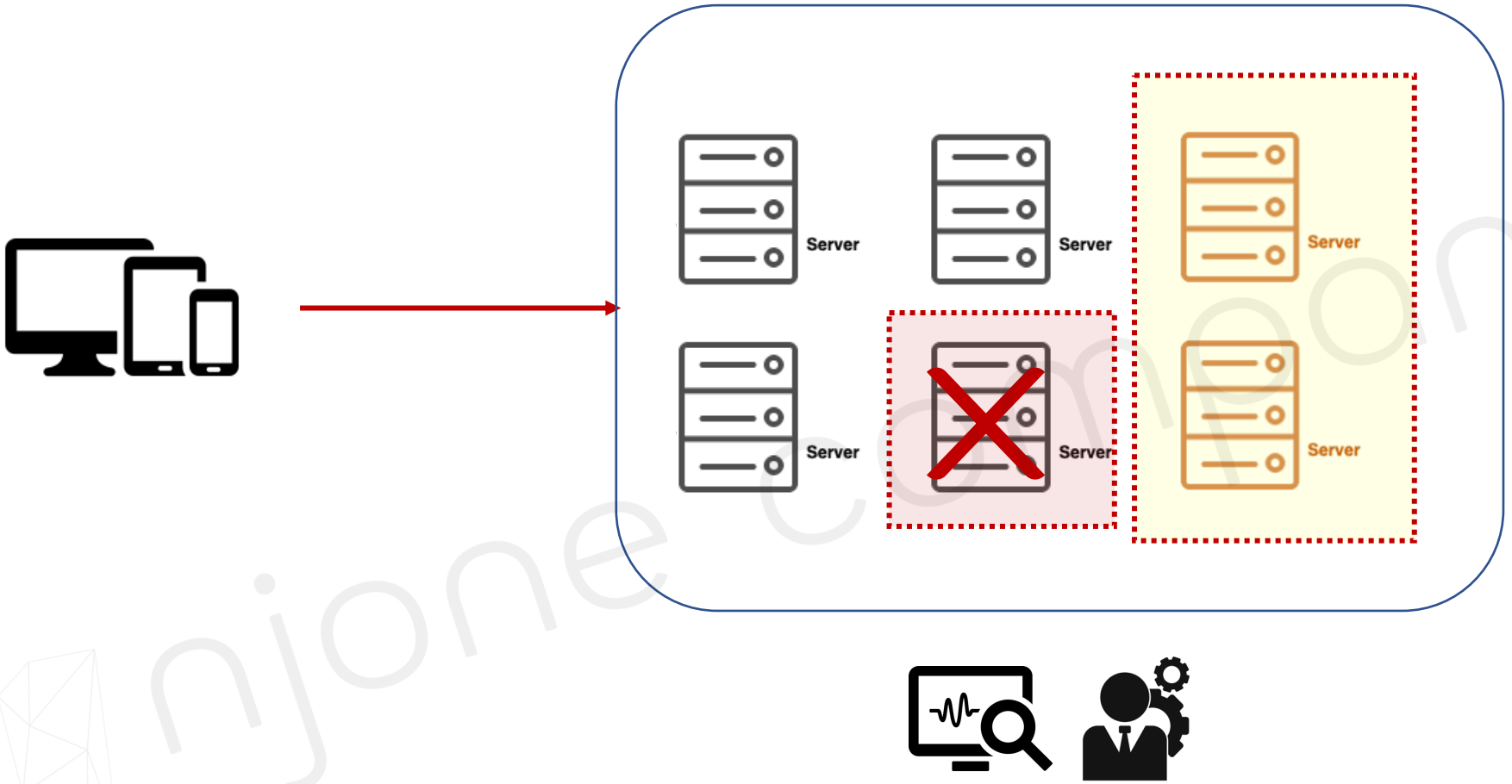
대한민국 ▾ 지난 12개월 ▾ 모든 카테고리 ▾ 웹 검색 ▾

시간 흐름에 따른 관심도 변화



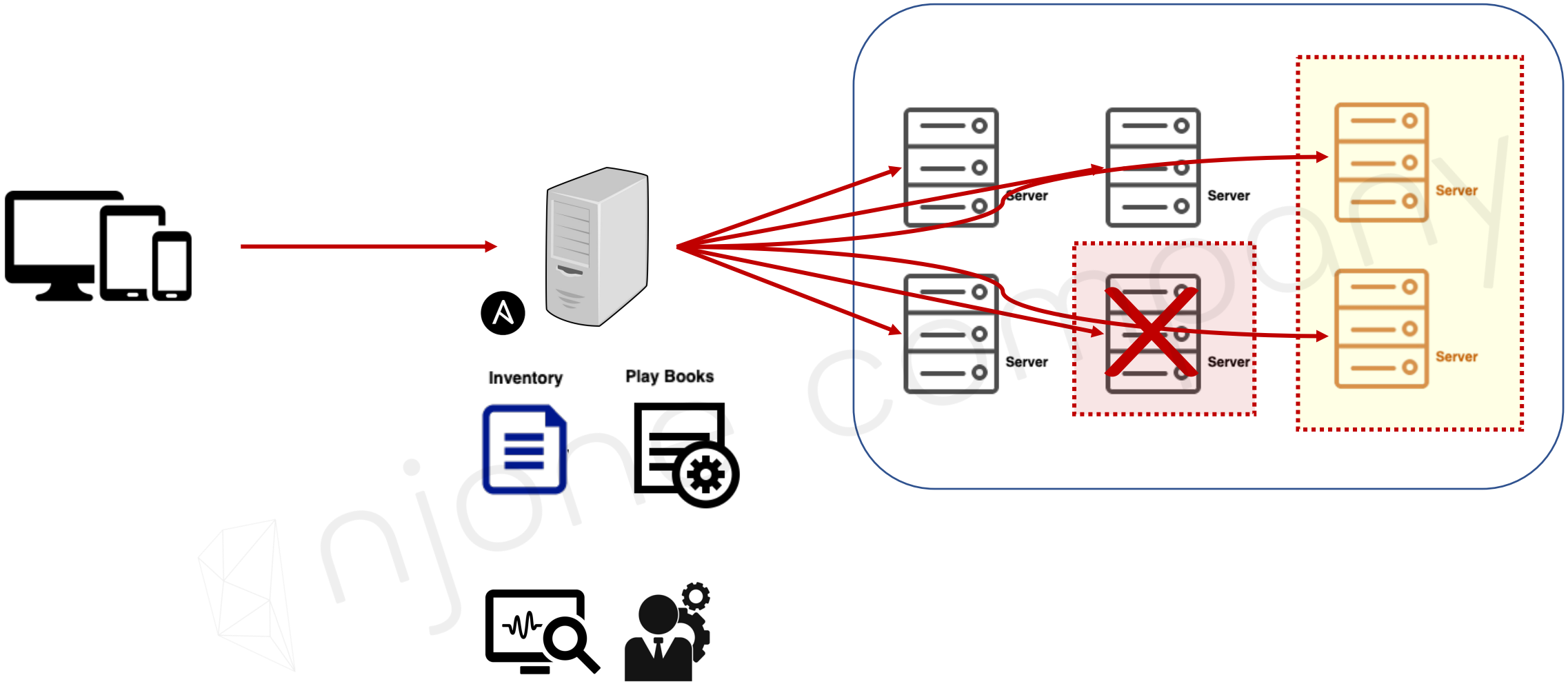
Before & After Configuration Management

- Before



Before & After Configuration Management

- After

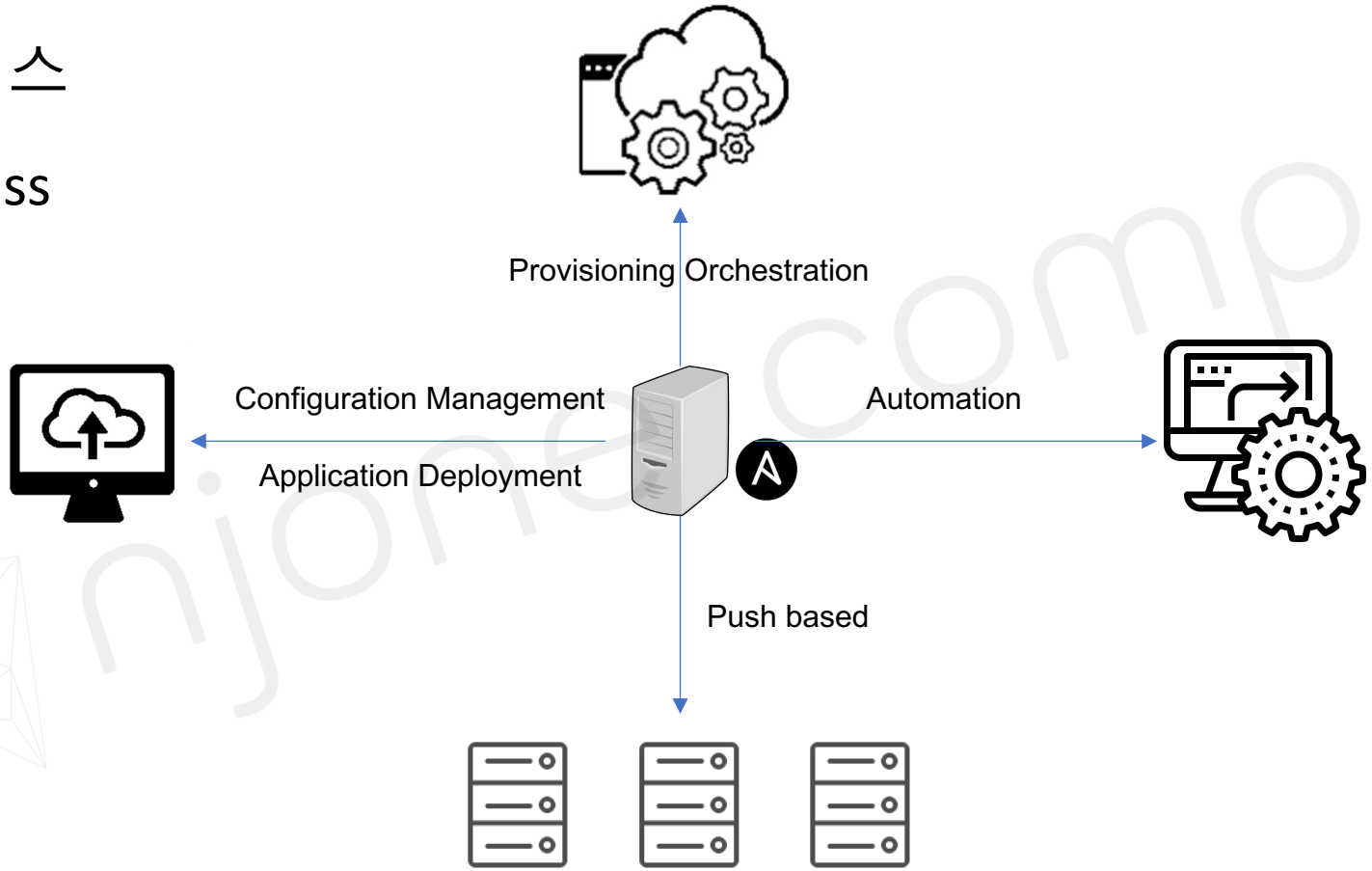


Configuration Management Tools

	Puppet	Chef	Salt	Ansible
개발사	Puppet Labs	Opscode	SaltStack	Redhat
등장	2005.08	2009.01	2011.03	2012.03
개발언어	Ruby	Ruby, Erlang	Python	Python
주요고객	Google, ebay, Disney …	Facebook, Ancestry.com …	Linkedin, HP …	Evernotes, Raskspace …
Base	Puppet Forge	Chef Supermarket	Slag-Formula	Ansible Galaxy
Web UI	Puppet Enterprise	Chef Manage	SaltStack Enterprise	Ansible Tower
Definition File	자체 DSL, 내장 Ruby	자체 DSL (Ruby 베이스)	YAML, 자체 DSL (Python 베이스)	YAML
Agent	필요	필요	필요 or 불필요	불필요
사용률	★★★	★★★	★	★★★
사용성	★	★	★	★★★

Ansible

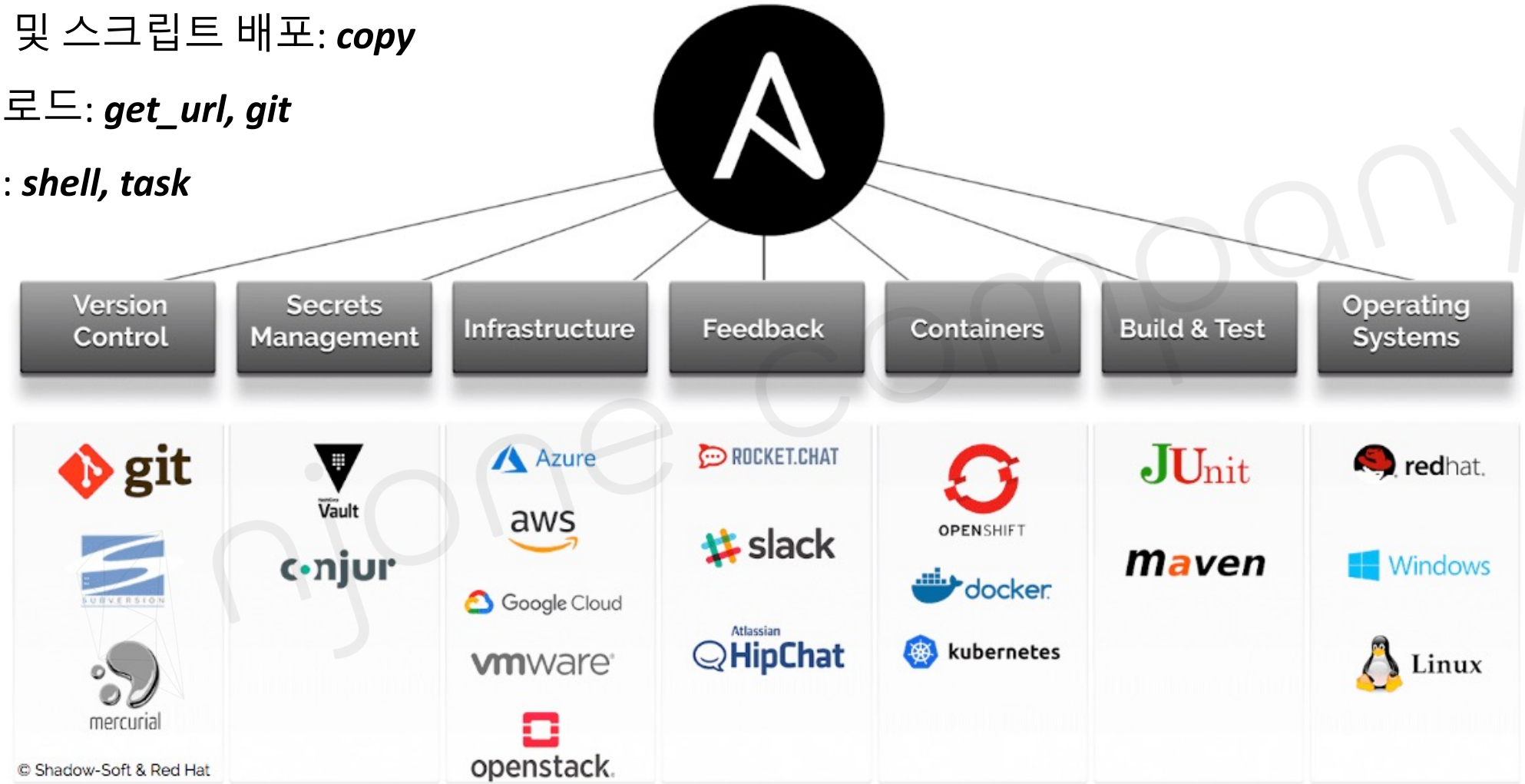
- 여러 개의 서버를 효율적으로 관리할 수 있게 해주는 환경 구성 자동화 도구
  - Configuration Management, Deployment & Orchestration tool
  - IT infrastructure 자동화
- Push 기반 서비스
- Simple, Agentless



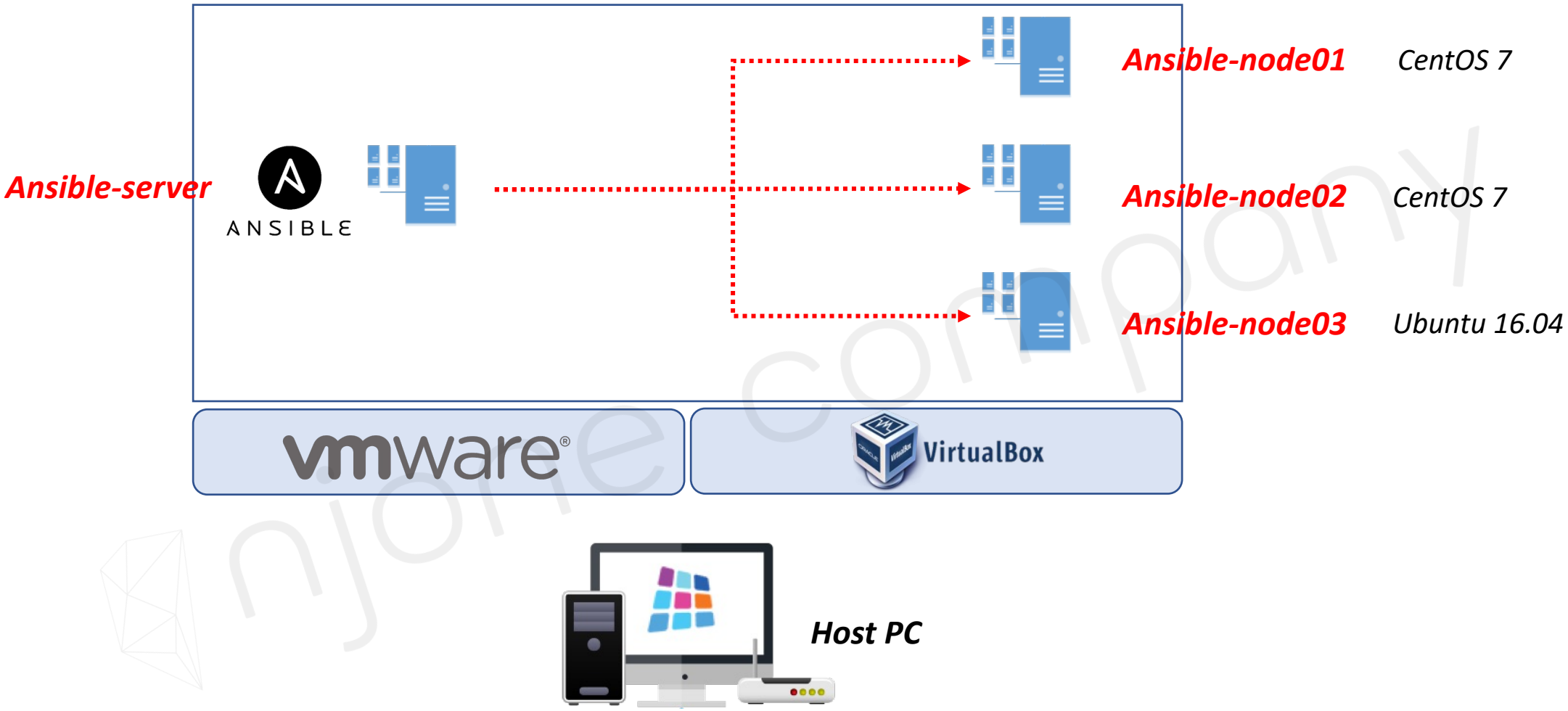
Ansible

- 할 수 있는 일
  - 설치: ***apt-get, yum, homebrew***
  - 파일 및 스크립트 배포: ***copy***
  - 다운로드: ***get_url, git***
  - 실행: ***shell, task***

- 결과
  - ***ok / failed / changed / unreachable***



- 구성도 예시



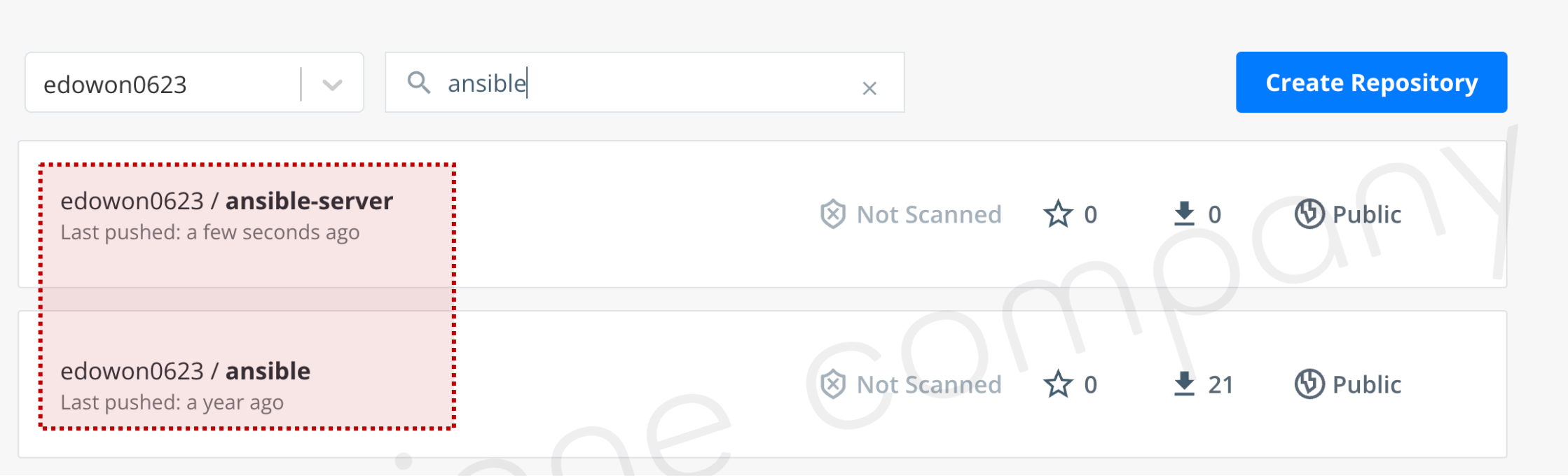
Install Ansible

- Ansible Server 설치 (Linux)
  - $ ***yum install ansible***
  - $ ***ansible --version***
- 환경 설정 파일 → ***/etc/ansible/ansible.cfg***
- Ansible에서 접속하는 호스트 목록 → ***/etc/ansible/hosts***

```
## db-[99:101]-node.example.com

[nginx]
172.20.10.11
172.20.10.12
172.20.10.13
```

Create a docker instance for Ansible

- https://hub.docker.com/r/edowon0623



- Windows, MacOS) $ ***docker pull edowon0623/ansible***
- Apple silicon, m1 chip) $ ***docker pull edowon0623/ansible-server:m1***

Run Ansible Server (Windows, MacOS)


```
$ docker run --privileged \
        --name ansible-server -itd \
        -p 20022:22 -p 8081:8080 \
        -e container=docker \
        -v /sys/fs/cgroup:/sys/fs/cgroup \
        edowon0623/ansible:latest /usr/sbin/init
```

$ **ssh root@localhost -p 20022** *(or docker exec -it ansible-server bash)*

```
ssh root@192.168.0.8 -p 20022
The authenticity of host '[192.168.0.8]:20022 ([192.168.0.8]:20022)' can't be established.
ECDSA key fingerprint is SHA256:Hpd/G+HMYEjzzkVucsl7CgliCp7M2rwQiYGBIGYZYz4.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[192.168.0.8]:20022' (ECDSA) to the list of known hosts.
root@192.168.0.8's password:
System is booting up. See pam_nologin(8)
[root@52a39c6a2ba9 ~]#
```

P@ssw0rd

Run Ansible Server (Windows, MacOS)

[root@52a39c6a2ba9 ~]# ***vi /etc/sysconfig/docker***

```
# /etc/sysconfig/docker

# Modify these options if you want to change the way the docker daemon Exercises
OPTIONS='--selinux-enabled=false --log-driver=journald --signature-verification=false'
if [ -z "${DOCKER_CERT_PATH}" ]; then
    DOCKER_CERT_PATH=/etc/docker
fi
```

[root@52a39c6a2ba9 ~]# ***sed -i -e 's/overlay2/vfs/g' /etc/sysconfig/docker-storage***

[root@52a39c6a2ba9 ~]# ***systemctl start docker***

```
[root@ae2b6f08d2dc ~]# sed -i -e 's/overlay2/vfs/g' /etc/sysconfig/docker-storage
[root@ae2b6f08d2dc ~]# systemctl start docker
[root@ae2b6f08d2dc ~]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; vendor preset: disabled)
   Active: active (running) since Mon 2021-09-06 05:26:06 UTC; 48min ago
     Docs: http://docs.docker.com
 Main PID: 283 (dockerd-current)
```

Run Ansible Server (Apple silicon chip, m1)


```
$ docker run --privileged \
        --name ansible-server -itd \
        -p 20022:22 -p 8081:8080 \
        -e container=docker \
        -v /sys/fs/cgroup:/sys/fs/cgroup --cgroupns=host \
        edowon0623/ansible-server:m1 /usr/sbin/init
```

$ **ssh root@localhost -p 20022** *(or docker exec -it ansible-server bash)*

```
ssh root@192.168.0.8 -p 20022
The authenticity of host '[192.168.0.8]:20022 ([192.168.0.8]:20022)' can't be established.
ECDSA key fingerprint is SHA256:Hpd/G+HMYEjzzkVucsl7CgliCp7M2rwQiYGBIGYZYz4.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[192.168.0.8]:20022' (ECDSA) to the list of known hosts.
root@192.168.0.8's password:
System is booting up. See pam_nologin(8)
[root@52a39c6a2ba9 ~]#
```

P@ssw0rd

Run Ansible Server

[root@8dd7ba5d82a8 ~]# ***ansible --version***

```
[root@8dd7ba5d82a8 ~]# ansible --version
ansible [core 2.13.2]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/local/lib/python3.8/site-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/local/bin/ansible
  python version = 3.8.8 (default, Aug 25 2021, 16:18:04) [GCC 8.5.0 20210514 (Red Hat 8.5.0-3)]
```

[root@ 8dd7ba5d82a8 ~]# ***vi /etc/ansible/hosts***

```
[devops]
172.17.0.3
172.17.0.4
```

docker network inspect bridge

```
"Containers": {
  "3228c6b59416fa46a11e73f2c345a98fdc56274c4508d16a41fce18a931bbd0c": {
  "Name": "docker-server",
  "IPv4Address": "172.17.0.3/16",
  ...
  "8dd7ba5d82a8432ba345161b6c3fdc84bb6e82e3efb60c9e9b3a21c2ba59a16e": {
  "Name": "ansible-server",
  "IPv4Address": "172.17.0.4/16",
  ...
  "e1253d619fc1da23a5bf30898e5a398bc5c08bc5a65513fd1394f35a50d7d709": {
  "Name": "jenkins-server",
  "IPv4Address": "172.17.0.2/16",
...
}
```

Run Ansible Server

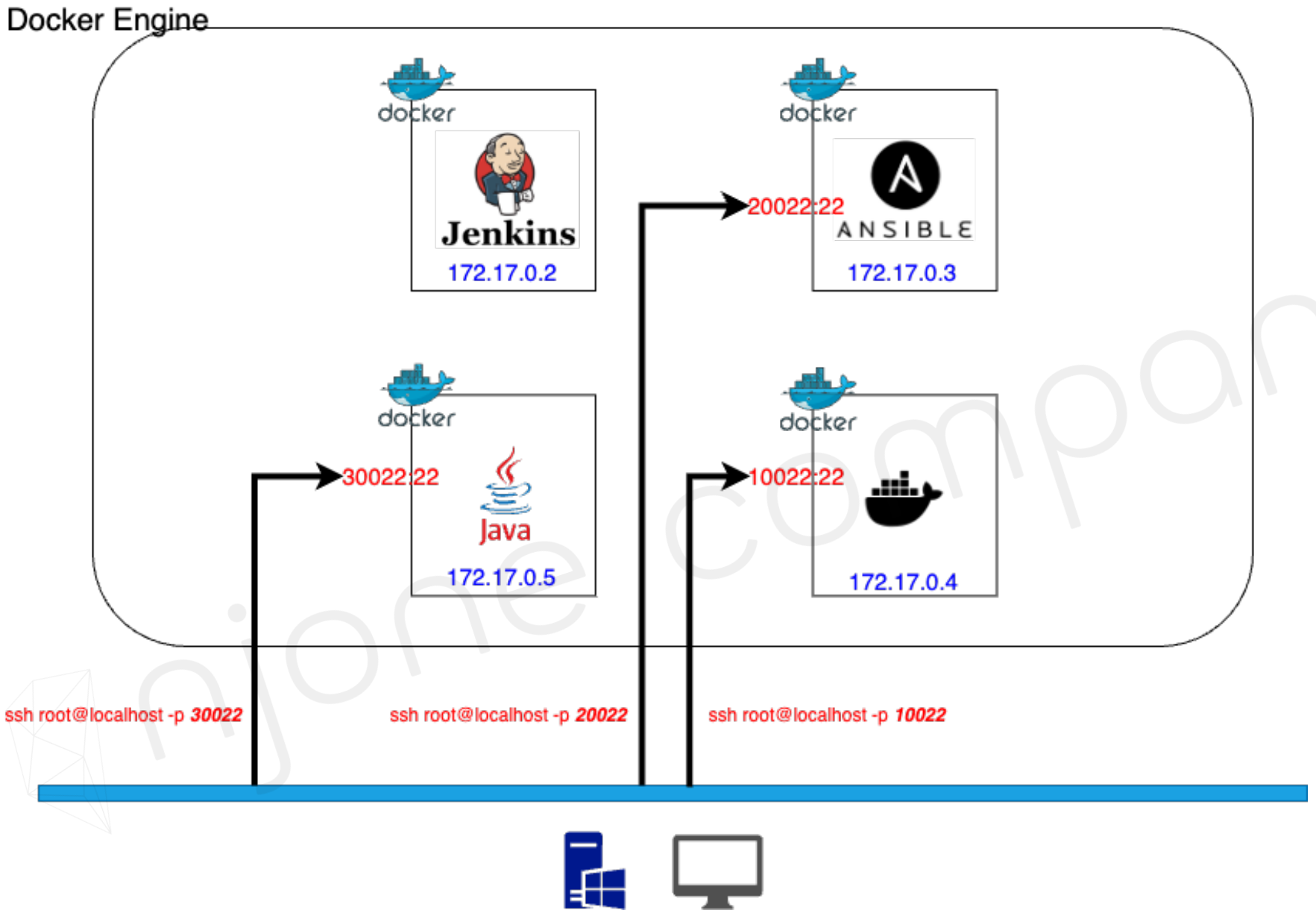
[root@52a39c6a2ba9 ~]# ***ssh-keygen***

```
[root@52a39c6a2ba9 ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
/root/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
```

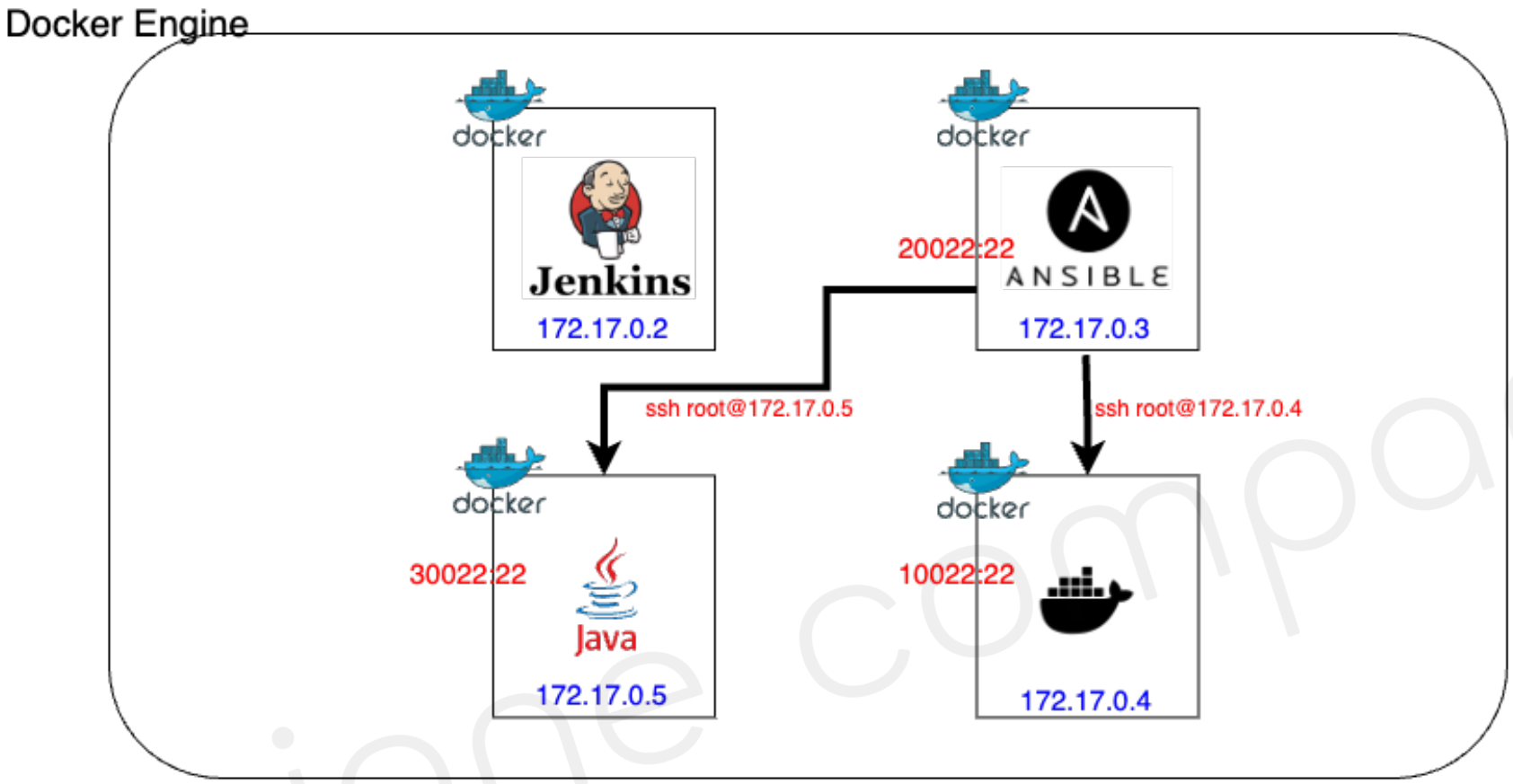
[root@52a39c6a2ba9 ~]# ***ssh-copy-id root@172.17.0.2***

```
[root@52a39c6a2ba9 ~]# ssh-copy-id root@172.17.0.2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@172.17.0.2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'root@172.17.0.2'"
and check to make sure that only the key(s) you wanted were added.
```

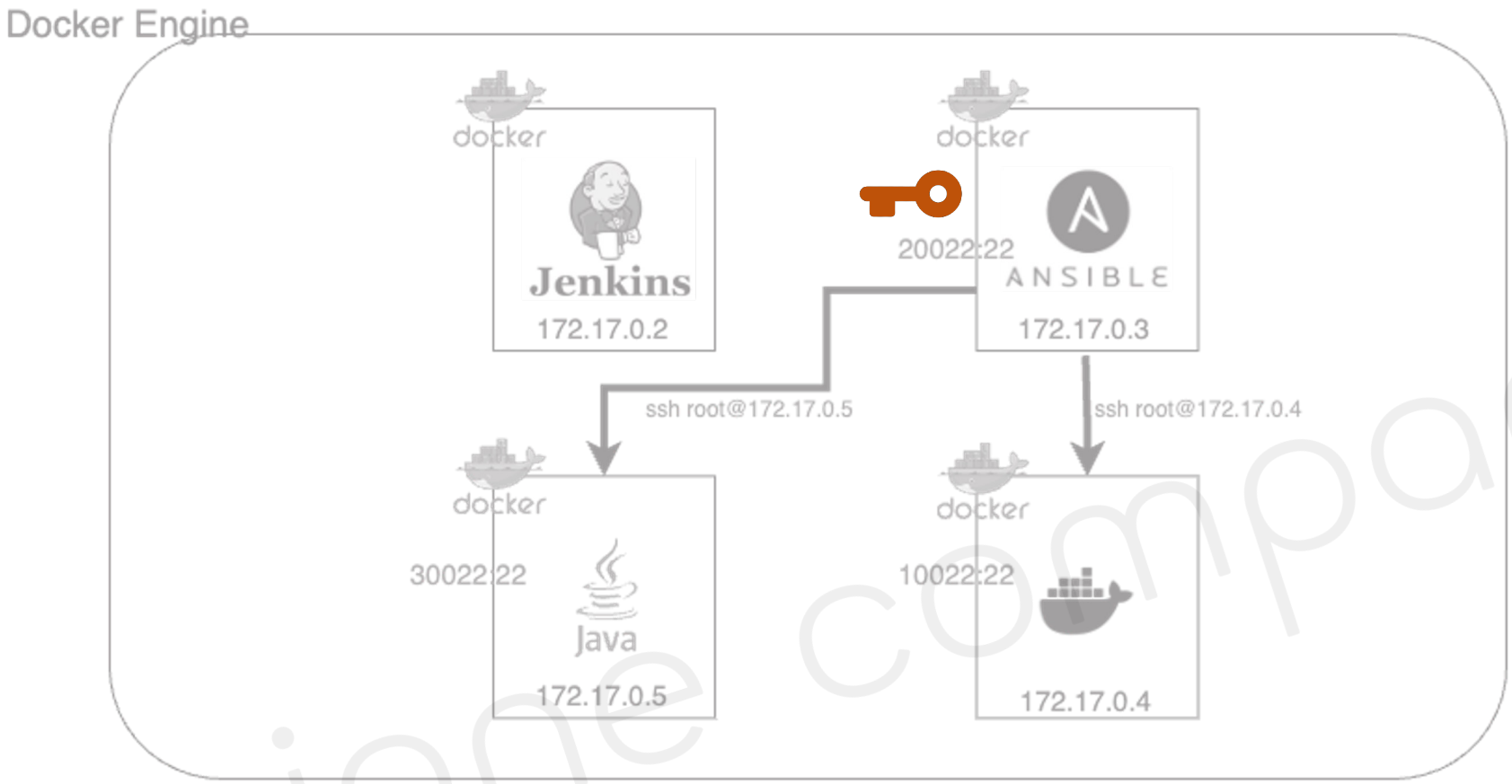
Run Ansible Server



Run Ansible Server



Run Ansible Server



Test Ansible module

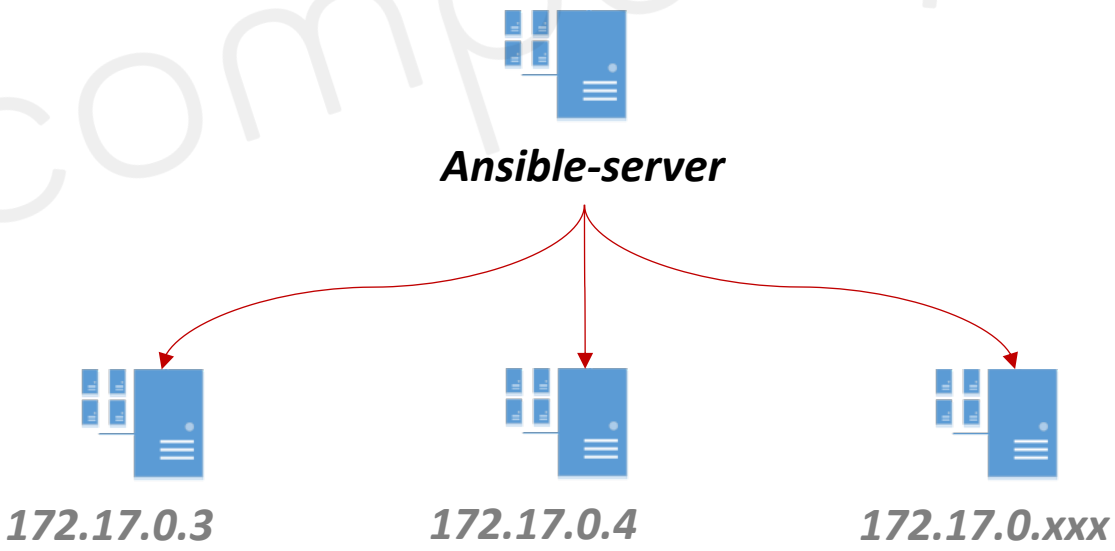
- 실행 옵션
  - *-i (--inventory-file) → 적용 될 호스트들에 대한 파일 정보*
  - *-m (--module-name) → 모듈 선택*
  - *-k (--ask-pass) → 관리자 암호 요청*
  - *-K (--ask-become-pass) → 관리자 권한 상승*
  - *--list-hosts → 적용되는 호스트 목록*
- ***멱등성***
  - 같은 설정을 여러 번 적용하더라도 결과가 달라지지 않는 성질

    ex) echo -e “[mygroup]\n172.20.10.11” >> /etc/ansible/hosts

Test Ansible module

- https://docs.ansible.com/ansible/2.9/modules/list_of_all_modules.html
- Ansible Test
  - $ ***ansible all -m ping***

```
[root@8dd7ba5d82a8 ~]# ansible all -m ping
172.17.0.4 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/
    },
    "changed": false,
    "ping": "pong"
}
172.17.0.3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/
    },
    "changed": false,
    "ping": "pong"
}
```



Test Ansible module

- *Exam1)* 디스크 용량 확인

[root@52a39c6a2ba9 ~]# ***ansible all -m shell -a "free -h"***

```
[root@52a39c6a2ba9 ~]# ansible all -m shell -a "free -h"
172.17.0.2 | CHANGED | rc=0 >>
              total        used        free      shared  buff/cache   available
Mem:           7.8G        3.3G        304M        338M        4.2G        3.9G
Swap:          1.0G        115M        908M
172.17.0.4 | CHANGED | rc=0 >>
              total        used        free      shared  buff/cache   available
Mem:           7.8G        3.3G        304M        338M        4.2G        3.9G
Swap:          1.0G        115M        908M
```

Test Ansible module

- *Exam2)* 파일 전송

[root@52a39c6a2ba9 ~]# ***ansible all -m copy -a "src=./test.txt dest=/tmp"***

```
[root@52a39c6a2ba9 ~]# ansible all -m copy -a "src=./test.txt dest=/tmp"
172.17.0.2 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "checksum": "ab6065a5305ad6f743c163dcd7124feaea41ac39",
    "dest": "/tmp/test.txt",
```

```
172.17.0.4 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "checksum": "ab6065a5305ad6f743c163dcd7124feaea41ac39",
    "dest": "/tmp/test.txt",
```

Test Ansible module

- *Exam3)* 서비스 설치

[root@52a39c6a2ba9 ~]# ***ansible all -m yum -a "name=httpd state=present"***

```
[root@52a39c6a2ba9 ~]# ansible all -m yum -a "name=httpd state=present"
172.17.0.4 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "changes": {
        "installed": [
            "httpd"
        ]
    },
```

```
[root@ae2b6f08d2dc tmp]# yum list installed | grep httpd
httpd.x86_64                            2.4.6-97.el7.centos             @updates
httpd-tools.x86_64                      2.4.6-97.el7.centos             @updates
[root@ae2b6f08d2dc tmp]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd(8)
           man:apachectl(8)
```

Ansible Playbook

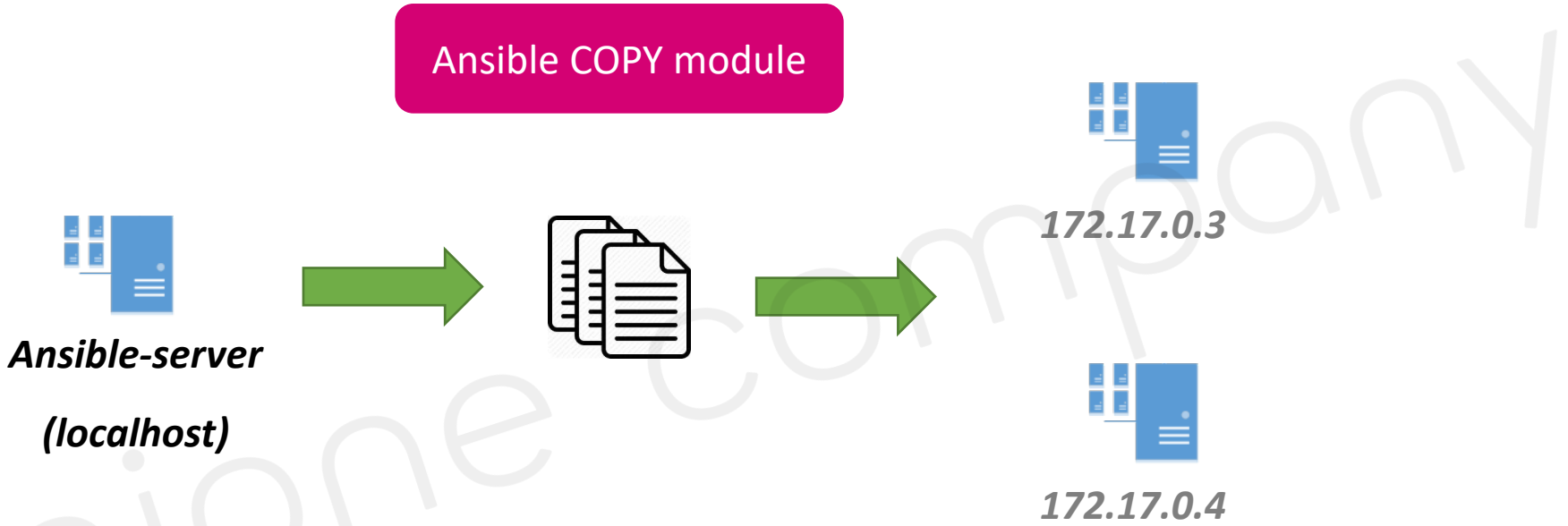
- 사용자가 원하는 내용을 미리 작성해 놓은 파일
  - ex) 설치, 파일 전송, 서비스 재시작 …
  - ex) 다수의 서버에 반복 작업을 처리하는 경우

- Playbook
  - $ vi first-playbook.yml 작성
  - $ ***ansible-playbook first-playbook.yml***
  - $ cat /etc/ansible/hosts

```
1   ---
2   - name: Ansible_vim
3     hosts: localhost
4     tasks:
5       - name: Add ansible hosts
6         blockinfile:
7           path: /etc/ansible/hosts
8           block: |
9             [mygroup]
10            172.20.10.11
```

```
[root@8dd7ba5d82a8 ~]# cat /etc/ansible/hosts
[devops]
172.17.0.3
172.17.0.4
# BEGIN ANSIBLE MANAGED BLOCK
[mygroup]
172.17.0.5
# END ANSIBLE MANAGED BLOCK
```

- Ansible Playbook 예제 – 파일 복사



Ansible Playbook

- Ansible Playbook 예제 – 파일 복사

```
[root@8dd7ba5d82a8 ~]# ansible-playbook playbook-sample1.yml
[WARNING]: An error occurred while calling ansible.utils.display.initialize_locale (unsupported locale setting). Thi
Display to print incorrect line lengths

PLAY [Ansible Copy Example Local to Remote] ************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [172.17.0.3]
ok: [172.17.0.4]

TASK [copying file with playbook] **********************************************************************************
changed: [172.17.0.4]
changed: [172.17.0.3]

PLAY RECAP *********************************************************************************************************
172.17.0.3                 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.17.0.4                 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

```
[root@8dd7ba5d82a8 ~]# ls -al /tmp/sample.txt
-rw-r--r-- 1 root root 11 Aug  2 15:41 /tmp/sample.txt
```

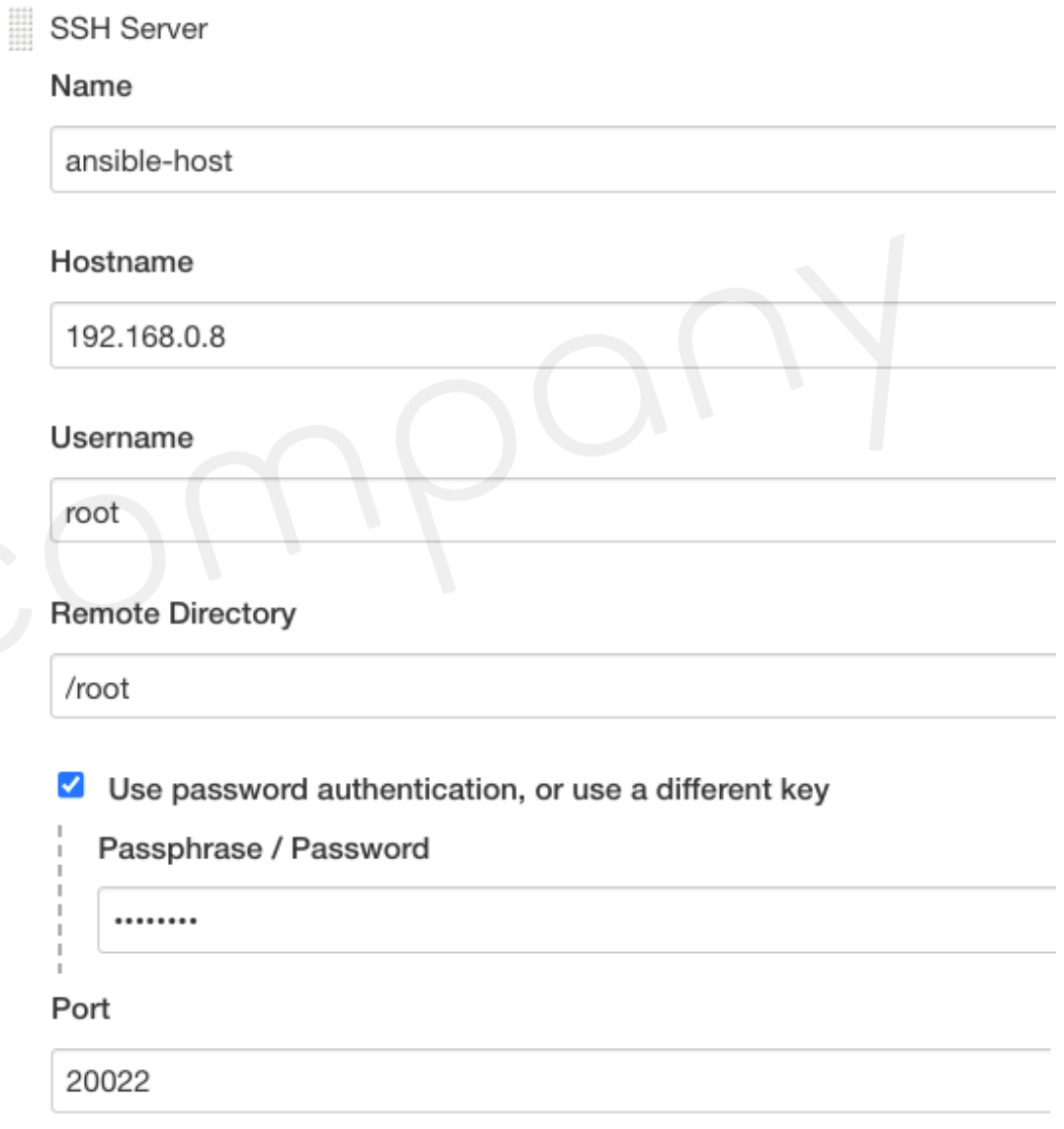
- Ansible Playbook 예제 – 다운로드

```yaml
---
- name: Download Tomcat9 from tomcat.apache.org
  hosts: devops
  tasks:
   - name: Create a Directory /opt/tomcat9
     file:
       path: /opt/tomcat9
       state: directory
       mode: 0755
   - name: Download Tomcat using get_url
     get_url:
       url: https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.65/bin/apache-tomcat-9.0.65.tar.gz
       dest: /opt/tomcat9
       mode: 0755
       checksum: sha512:https://downloads.apache.org/tomcat/tomcat-9/v9.0.65/bin/apache-tomcat-9.0.65.tar.gz.sha512
```

```
TASK [Download Tomcat using get_url] ********************************************************************
changed: [172.17.0.3]
changed: [172.17.0.4]

PLAY RECAP **********************************************************************************************
172.17.0.3                 : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.17.0.4                 : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

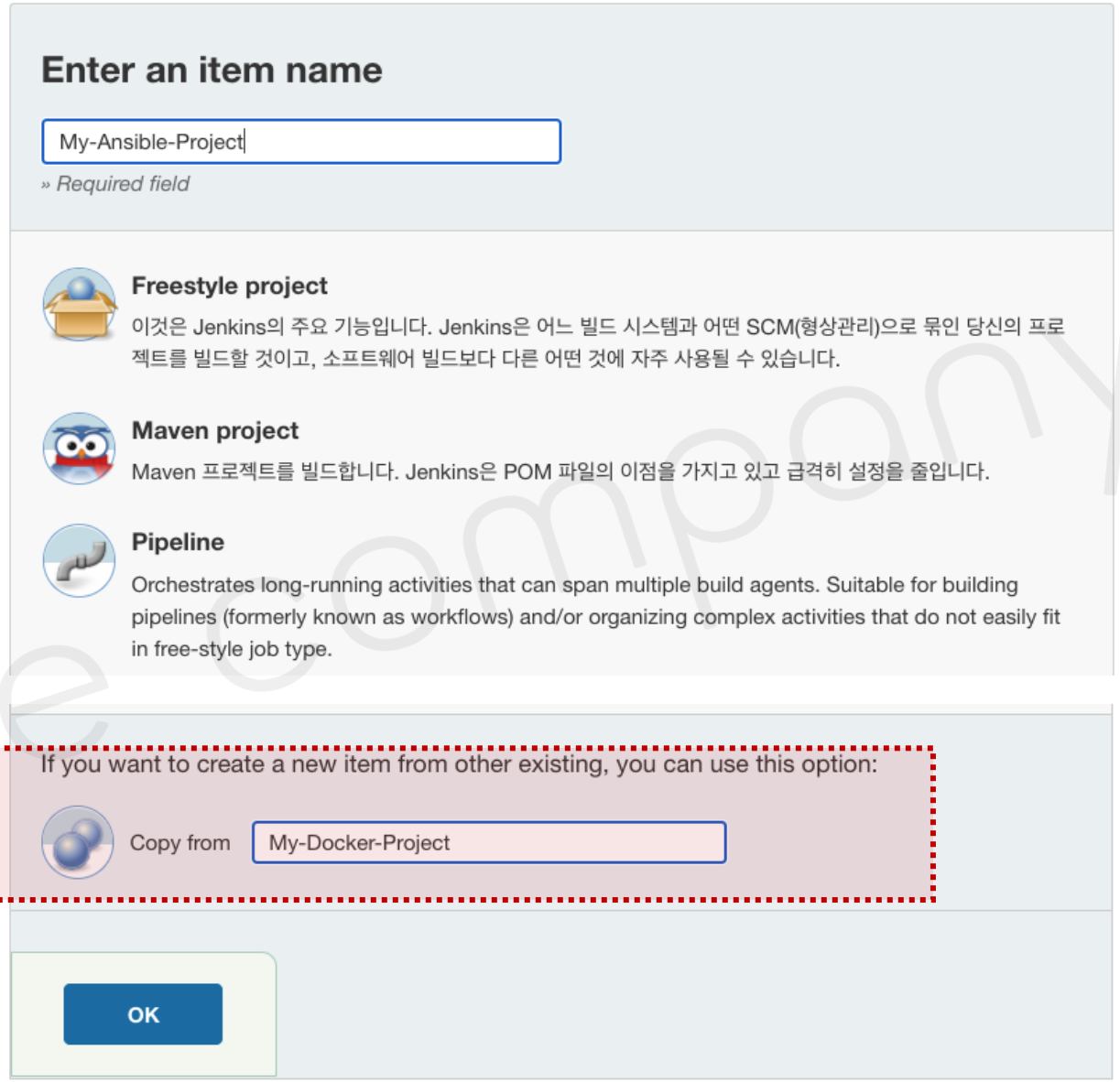
Setup Ansible on Jenkins

- Manage Jenkins → Configure System → Publish over SSH
  - Add SSH Servers
    - Name: ***ansible-host***
    - Hostname: ***[Remote IP] ex)192.168.0.8***
    - Username: ***root***
    - Passphrase/Password: ***P@ssw0rd***
    - Port: ***20022***
  - Test Configuration



Exercise 5# Jenkins Job 1/13

- Item name → ***My-Ansible-Project***
  - Copy from: ***My-Docker-Project***



Exercise 5# Jenkins Job 2/13

- Post-build Actions
  - Deploy war/ear to a container → ***Delete***
  - ***Send build artifacts over SSH***
    - ***SSH Server***
      - Name: ***[Publish over SSH에서 설정한 이름] ex) ansible-host***
      - ***Transfer Set***
        - Source files: ***target/*.war***
        - Remove prefix: ***target***
        - Remote directory: **.**

- Save > Build Now

```
[root@8dd7ba5d82a8 ~]# ls -l *.war
-rw-r--r-- 1 root root 8025885 Aug  2 07:59 hello-world.war
```

Exercise 5# Jenkins Job 3/13

- Create an ansible playbook file
  - $ ***vi first-devops-playbook.yml***

```
1  - hosts: all
2  #   become: true
3
4    tasks:
5    - name: build a docker image with deployed war file
6      command: docker build -t cicd-project-ansible .
7      args:
8          chdir: /root
```

- Create a hosts file
  - $ ***vi hosts***

```
[root@52a39c6a2ba9 ~]# cat hosts
172.17.0.4
```

Exercise 5# Jenkins Job 4/13

- $ ***ansible-playbook -i hosts first-devops-playbook.yml --check***

```
[root@52a39c6a2ba9 ~]# ansible-playbook -i hosts first-devops-playbook.yml --check

PLAY [all] *********************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [172.17.0.4]

TASK [build docker image using war file] ***************************************************************************
skipping: [172.17.0.4]

PLAY RECAP *********************************************************************************************************
172.17.0.4                 : ok=1    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

Exercise 5# Jenkins Job 5/13

- $ ***ansible-playbook -i hosts first-devops-playbook.yml***

```
[root@52a39c6a2ba9 ~]# ansible-playbook -i hosts first-devops-playbook.yml

PLAY [all] *********************************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [172.17.0.4]

TASK [build a docker image with deployed war file] *****************************************************
changed: [172.17.0.4]

PLAY RECAP *********************************************************************************************
172.17.0.4                 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@52a39c6a2ba9 ~]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
cicd-project-ansible    latest              429359784e79        8 seconds ago       680 MB
docker.io/tomcat        latest              ab1f0e1bb1a1        2 days ago          680 MB
```

Exercise 5# Jenkins Job 6/13

- Add a task for create a container

```
1  - hosts: all
2  #   become: true
3
4    tasks:
5    - name: build a docker image with deployed war file
6      command: docker build -t cicd-project-ansible .
7      args:
8          chdir: /root
9    - name: create a container using cicd-project-ansible image
10     command: docker run -d --name my_cicd_project -p 8081:8080 cicd-project-ansible
```

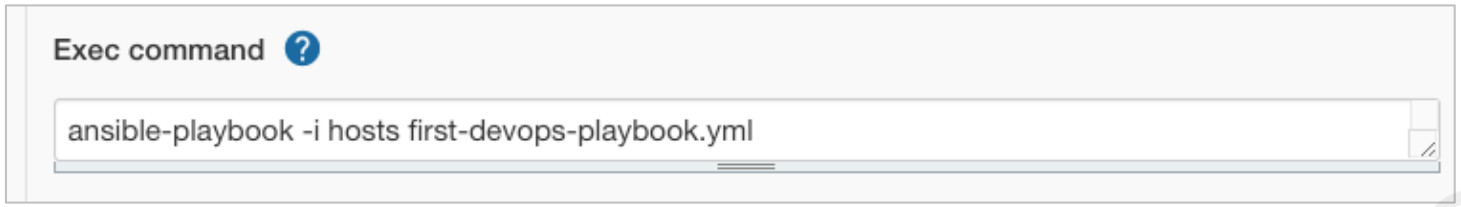
Exercise 5# Jenkins Job 7/13

- $ ***ansible-playbook -i hosts first-devops-playbook.yml***

```
[root@52a39c6a2ba9 ~]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
cicd-project-ansible    latest              429359784e79        4 minutes ago       680 MB
docker.io/tomcat        latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@52a39c6a2ba9 ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@52a39c6a2ba9 ~]# ansible-playbook -i hosts first-devops-playbook.yml

PLAY [all] *********************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************
ok: [172.17.0.4]

TASK [build a docker image with deployed war file] *****************************************************************************************
changed: [172.17.0.4]

TASK [create a container using cicd-project-ansible image] *********************************************************************************
changed: [172.17.0.4]

PLAY RECAP *********************************************************************************************************************************
172.17.0.4                 : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@52a39c6a2ba9 ~]# docker ps
CONTAINER ID        IMAGE                   COMMAND             CREATED             STATUS              PORTS                    NAMES
e3fddace3ddb        cicd-project-ansible    "catalina.sh run"   16 seconds ago      Up 8 seconds        0.0.0.0:8081->8080/tcp   my_cicd_project
```

Exercise 5# Jenkins Job 8/13

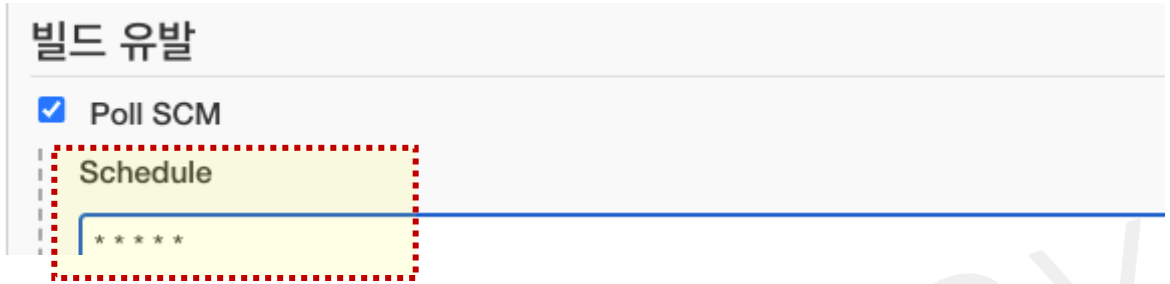
- ***Delete docker image and docker container***
- ***Execute ansible-playbook file on Jenkins***
  - *ansible-playbook -i hosts first-devops-playbook.yml*




```
[root@52a39c6a2ba9 ~]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
cicd-project-ansible    latest              225086aec8a0        22 seconds ago      680 MB
docker.io/tomcat        latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@52a39c6a2ba9 ~]# docker ps
CONTAINER ID        IMAGE                   COMMAND             CREATED             STATUS
f47416fc21f6        cicd-project-ansible    "catalina.sh run"   23 seconds ago      Up 13 seconds
```

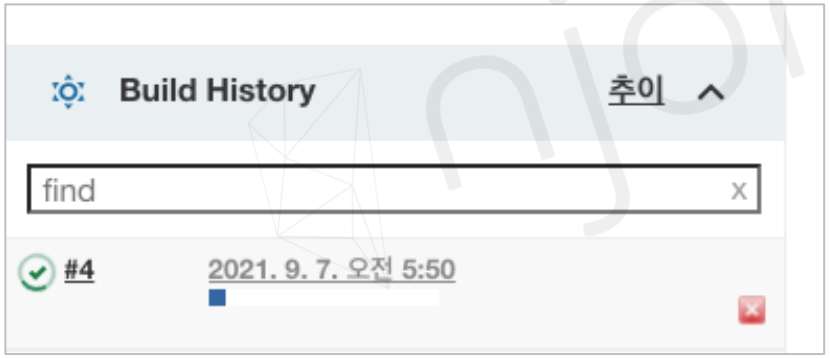
Exercise 5# Jenkins Job 9/13

- ***Modify the source code and Git commit (push)***
  - ***Build Triggers***



  - $ ***git add . && git commit -m "changed the index page" && git push***

```
1   <h1> Welcome to Simple Ecommerce ver 1.2</h1>
2   <h2> Deploy on Docker container with ansible</h2>
```



Exercise 5# Jenkins Job 10/13

- ***Build Now automatically → UNSTABLE***
  - ***Build a new images → OK***
  - ***Create a new container → Fail***

```
[root@52a39c6a2ba9 ~]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED                  SIZE
cicd-project-ansible    latest              4abd35f542cf        About a minute ago       680 MB
<none>                  <none>              225086aec8a0        34 minutes ago           680 MB
docker.io/tomcat        latest              ab1f0e1bb1a1        2 days ago               680 MB
[root@52a39c6a2ba9 ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
f47416fc21f6        225086aec8a0        "catalina.sh run"   34 minutes ago      Up 34 minutes
```

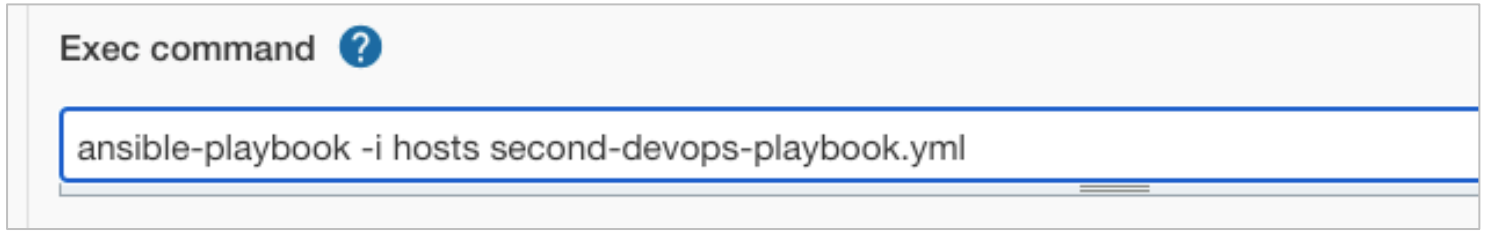
Exercise 5# Jenkins Job 11/13

- $ ***vi second-devops-playbook.yml***

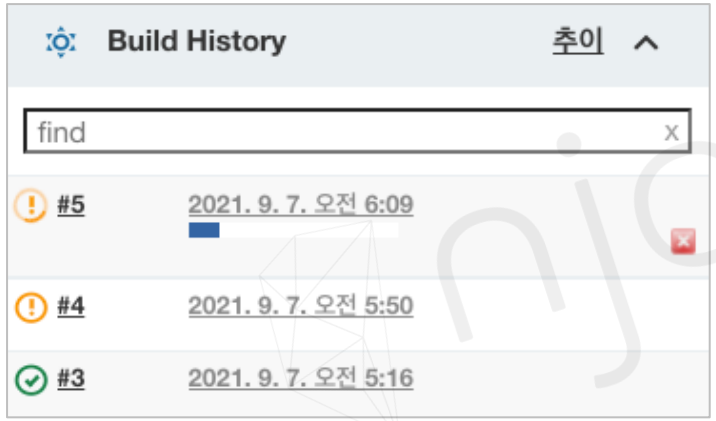
```
1  - hosts: all
2  #   become: true
3
4    tasks:
5    - name: stop current running container
6      command: docker stop my_cicd_project
7      ignore_errors: yes
8
9    - name: remove stopped cotainer
10     command: docker rm my_cicd_project
11     ignore_errors: yes
12
13   - name: remove current docker image
14     command: docker rmi cicd-project-ansible
15     ignore_errors: yes
16
17   - name: build a docker image with deployed war file
18     command: docker build -t cicd-project-ansible .
19     args:
20         chdir: /root
21
22   - name: create a container using cicd-project-ansible image
23     command: docker run -d --name my_cicd_project -p 8081:8080 cicd-project-ansible
```

Exercise 5# Jenkins Job 12/13

- ***Modify the execute command***
  - ***ansible-playbook -i hoosts second-devops-playbook.yml***



- ***Build Now again***




```
channel stopped
SSH: Connecting from host [512076bd5890]
SSH: Connecting with configuration [ansible-host] ...
SSH: EXEC: completed after 30,409 ms
SSH: Disconnecting configuration [ansible-host] ...
SSH: Transferred 1 file(s)
Finished: SUCCESS
```

Exercise 5# Jenkins Job 13/13

- ***Before***

```
[root@52a39c6a2ba9 ~]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
cicd-project-ansible    latest              4abd35f542cf        16 minutes ago      680 MB
<none>                  <none>              225086aec8a0        50 minutes ago      680 MB
docker.io/tomcat        latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@52a39c6a2ba9 ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
f47416fc21f6        225086aec8a0        "catalina.sh run"   50 minutes ago      Up 50 minutes
```

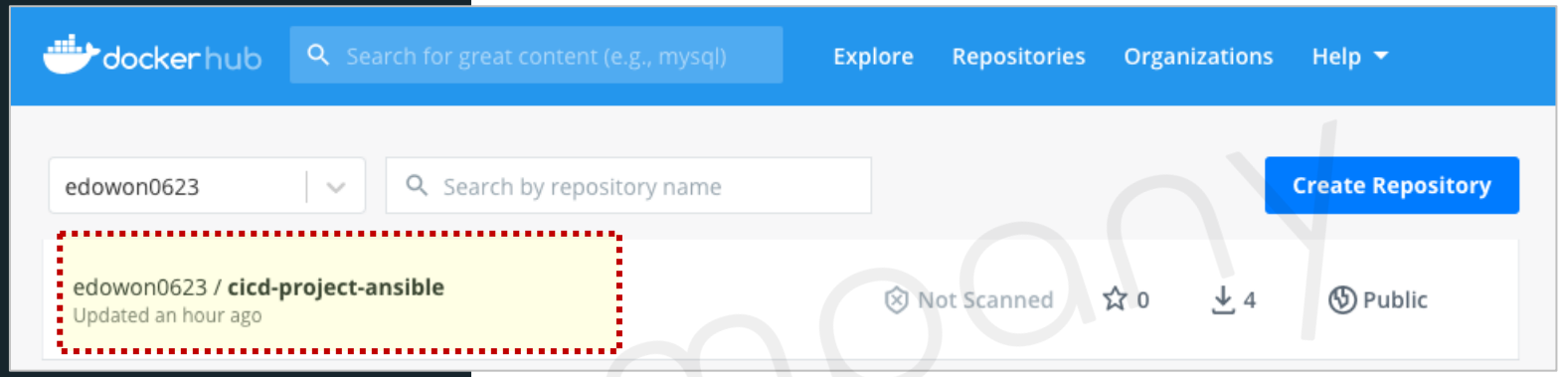
- ***After***

```
[root@52a39c6a2ba9 ~]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
cicd-project-ansible    latest              8694f2a9019f        About a minute ago  680 MB
<none>                  <none>              225086aec8a0        53 minutes ago      680 MB
docker.io/tomcat        latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@52a39c6a2ba9 ~]# docker ps
CONTAINER ID        IMAGE                   COMMAND             CREATED             STATUS
cc1f6bf62f99        cicd-project-ansible    "catalina.sh run"   About a minute ago  Up About a minute
```

Push the image on Docker hub

- $ ***docker push edowon0623/cicd-project-ansible***

```
[root@ansible-server ~]# docker push edowon0623/cicd-project-ansible
The push refers to a repository [docker.io/edowon0623/cicd-project-ansible]
1e5688f9bb9a: Preparing
d1cda6beea3c: Preparing
136a796cca9b: Preparing
4e4de253c94d: Preparing
3891808a925b: Preparing
d402f4f1b906: Waiting
00ef5416d927: Waiting
8555e663f65b: Waiting
d00da3cd7763: Waiting
4e61e63529c2: Waiting
799760671c38: Waiting
denied: requested access to the resource is denied
```




```
[root@ansible-server ~]# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head
over to https://hub.docker.com to create one.
Username: edowon0623
Password:
Login Succeeded
```

```
[root@ansible-server ~]# docker push edowon0623/cicd-project-ansible
The push refers to a repository [docker.io/edowon0623/cicd-project-ansible]
1e5688f9bb9a: Pushing 6.144 kB
d1cda6beea3c: Pushing 2.048 kB
136a796cca9b: Pushing  4.02 MB/20.08 MB
4e4de253c94d: Pushing 3.072 kB
```

Create a new playbook file

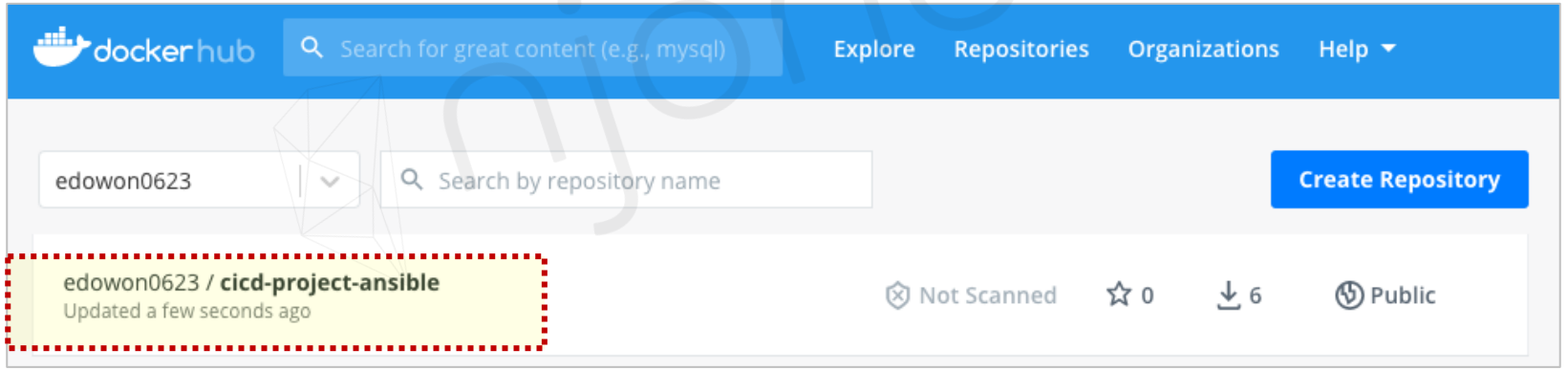
- $ ***vi create-cicd-devops-image.yml***

```
1  - hosts: all
2  #   become: true
3
4    tasks:
5    - name: create a docker image with deployed waf file
6      command: docker build -t edowon0623/cicd-project-ansible .
7      args:
8          chdir: /root
9
10   - name: push the image on Docker Hub
11     command: docker push edowon0623/cicd-project-ansible
12
13   - name: remove the docker image from the ansible server
14     command: docker rmi edowon0623/cicd-project-ansible
15     ignore_errors: yes
```

Create a new playbook file

- $ ***ansible-playbook -i hosts create-cicd-devops-image.yml***

```
PLAY [all] *********************************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [172.17.0.2]

TASK [create a docker image with deployed waf file] ****************************************************
changed: [172.17.0.2]

TASK [push the image on Docker Hub] ********************************************************************
changed: [172.17.0.2]

TASK [remove the docker image from the ansible server] *************************************************
changed: [172.17.0.2]

PLAY RECAP *********************************************************************************************
172.17.0.2                 : ok=4    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



Modify the previous playbook file

- $ ***mv second-devops-playbook.yml create-cicd-devops-container.yml***
- $ ***vi create-cicd-devops-container.yml***

```
4   tasks:
5   - name: stop current running container
6     command: docker stop my_cicd_project
7     ignore_errors: yes
8
9   - name: remove stopped cotainer
10    command: docker rm my_cicd_project
11    ignore_errors: yes
12
13  - name: remove current docker image
14    command: docker rmi edowon0623/cicd-project-ansible
15    ignore_errors: yes
16
17  - name: pull the newest docker image from Docker Hub
18    command: docker pull edowon0623/cicd-project-ansible
19
20  - name: create a container using cicd-project-ansible image
21    command: docker run -d --name my_cicd_project -p 8080:8080 edowon0623/cicd-project-ansible
```

Delete all of the images and containers

- ansible-server, docker-server 에 작업
  - $ ***docker rmi cicd-project-ansible***
  - $ ***docker stop my_cicd_project && docker rm my_cicd_proejct***

```
[root@ansible-server ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
docker.io/tomcat    latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@ansible-server ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@ansible-server ~]#
```

```
[root@docker-server ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
[root@docker-server ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@docker-server ~]#
```

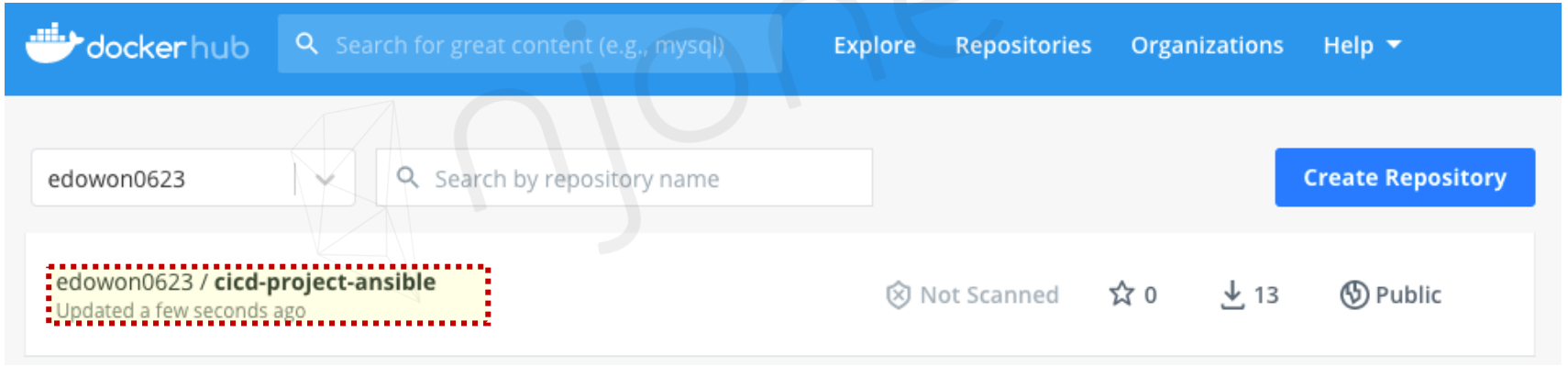
- change the hosts file (add the docker-server's ip address)

```
[root@ansible-server ~]# cat hosts
172.17.0.2
172.17.0.4
```

Execute the playbook file to create image

- $ ***ansible-playbook -i hosts create-cicd-devops-image.yml --limit 172.17.0.2***

```
PLAY [all] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.17.0.2]

TASK [create a docker image with deployed waf file] ****************************
changed: [172.17.0.2]

TASK [push the image on Docker Hub] ********************************************
changed: [172.17.0.2]

TASK [remove the docker image from the ansible server] *************************
changed: [172.17.0.2]

PLAY RECAP *********************************************************************
172.17.0.2                 : ok=4    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



Execute the playbook file to create container

njone company

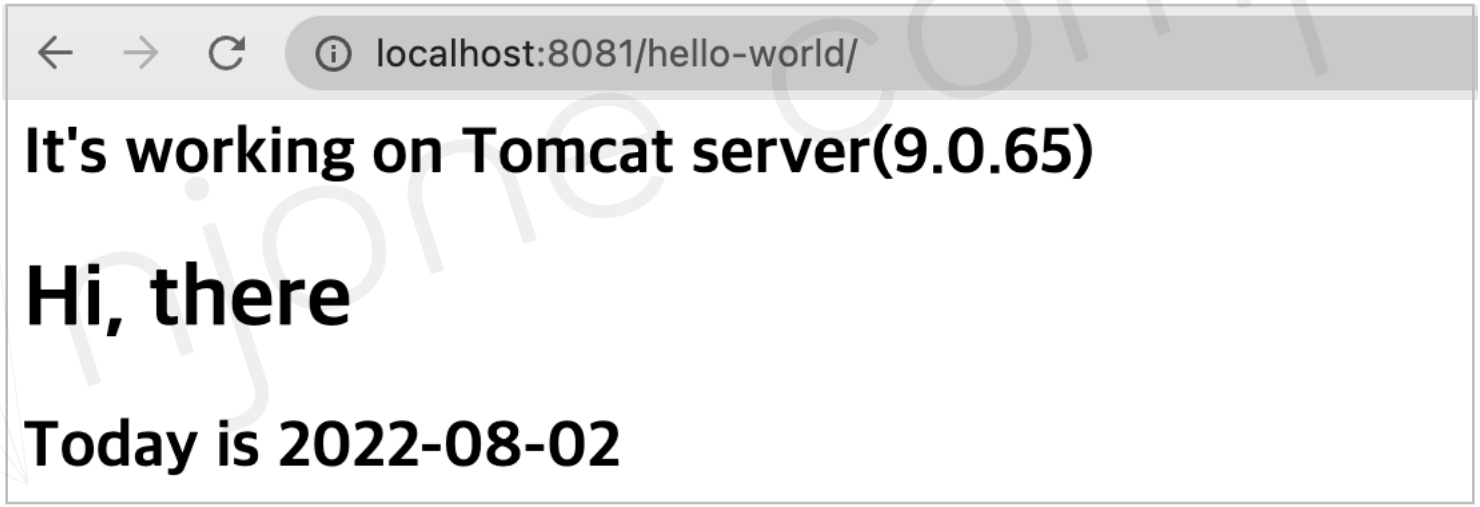
- $ ***ansible-playbook -i hosts create-cicd-devops-container.yml --limit 172.17.0.4***

```
[root@ansible-server ~]# ansible-playbook -i hosts create-cicd-devops-container.yml --limit 172.17.0.4

PLAY [all] *********************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************
ok: [172.17.0.4]

TASK [stop current running container] ******************************************************************************************
...ignoring

TASK [remove stopped cotainer] *************************************************************************************************
...ignoring

TASK [remove current docker image] *********************************************************************************************
...ignoring

TASK [pull the newest docker image from Docker Hub] ****************************************************************************
changed: [172.17.0.4]

TASK [create a container using cicd-project-ansible image] *********************************************************************
changed: [172.17.0.4]

PLAY RECAP *********************************************************************************************************************
172.17.0.4                 : ok=6    changed=5    unreachable=0    failed=0    skipped=0    rescued=0    ignored=3
```

Execute the playbook file to create container

- docker-server) $ ***docker ps -a***

```
[root@docker-server ~]# docker images
REPOSITORY                                  TAG                 IMAGE ID            CREATED             SIZE
docker.io/edowon0623/cicd-project-ansible   latest              c94b3843675d        13 minutes ago      680 MB
docker.io/tomcat                            latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@docker-server ~]# docker ps -a
CONTAINER ID        IMAGE                                       COMMAND             CREATED             STATUS              PORTS
  NAMES
25f5997e007a        edowon0623/cicd-project-ansible   "catalina.sh run"   32 seconds ago      Up 23 seconds       0.0.0.0:8080->8080/tcp
  my_cicd_project
```

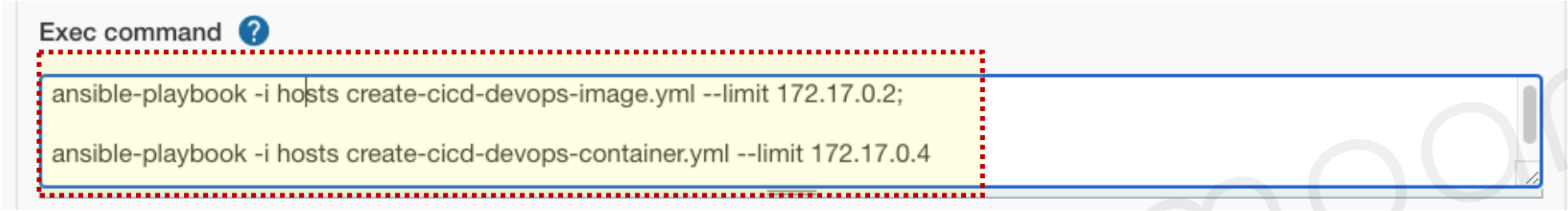


Exercise 6# Jenkins Job 1/4

- Item name → ***My-AnsibleBook-Project***
  - Copy from: ***My-Ansible-Project***

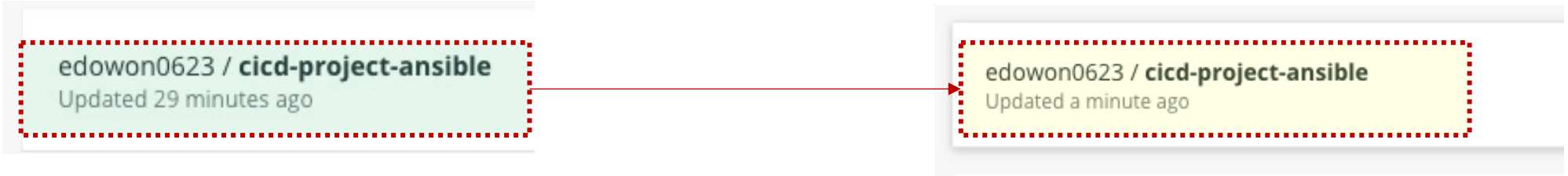
Exercise 6# Jenkins Job 2/4

- ***Change Exec command***
  - ***ansible-playbook -i hosts create-cicd-devops-image.yml --limit 172.17.0.2***
  - ***ansible-playbook -i hosts create-cicd-devops-container.yml --limit 172.17.0.4***



- ***Build Now***

Exercise 6# JenKins Job 3/4

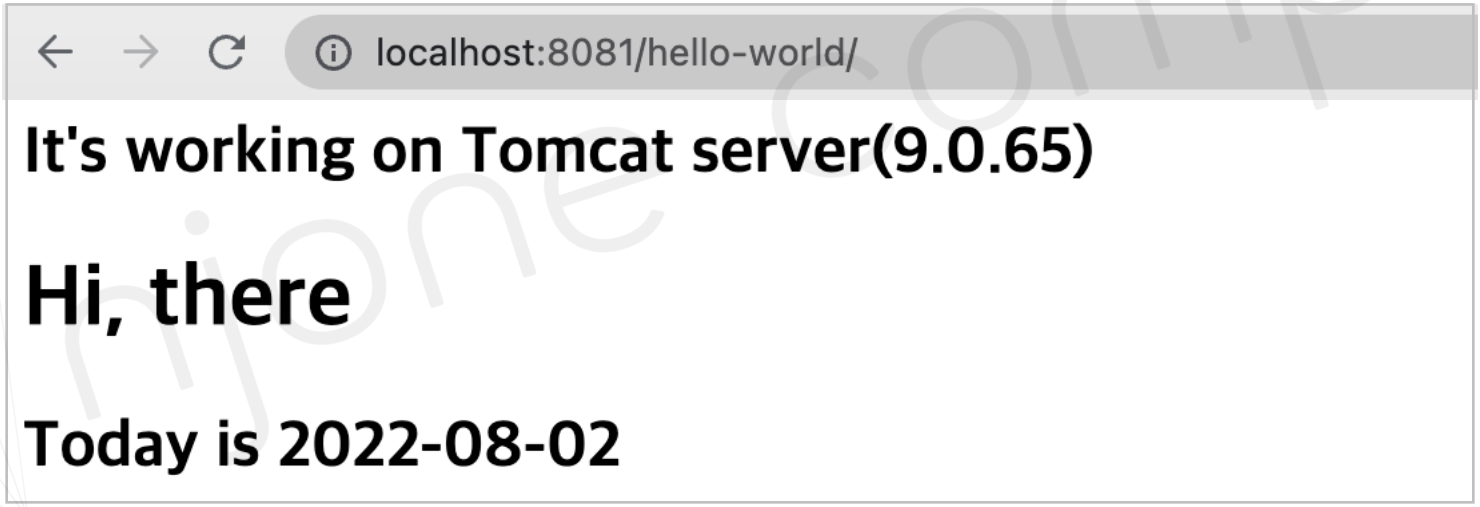



```
[root@docker-server ~]# docker images
REPOSITORY                                  TAG                 IMAGE ID            CREATED             SIZE
docker.io/edowon0623/cicd-project-ansible   latest              c94b3843675d        30 minutes ago      680 MB
docker.io/tomcat                            latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@docker-server ~]# docker ps -a
CONTAINER ID        IMAGE                                       COMMAND             CREATED             STATUS
  NAMES
25f5997e007a        edowon0623/cicd-project-ansible              "catalina.sh run"   16 minutes ago      Up 16 minutes
```

```
[root@docker-server ~]# docker images
REPOSITORY                                  TAG                 IMAGE ID            CREATED             SIZE
docker.io/edowon0623/cicd-project-ansible   latest              a932b39ee28a        About a minute ago   680 MB
docker.io/tomcat                            latest              ab1f0e1bb1a1        2 days ago          680 MB
[root@docker-server ~]# docker ps -a
CONTAINER ID        IMAGE                                       COMMAND             CREATED             STATUS
   NAMES
b7945bb60aae        edowon0623/cicd-project-ansible              "catalina.sh run"   About a minute ago   Up 57 seconds
   my_cicd_project
```

Exercise 6# Jenkins Job 4/4

- ***modify the sources***
  - $ ***git add .***
  - $ ***git commit -m “changed the index page”***
  - $ ***git push***



Section 4.

Advanced Jenkins

- Delivery Pipeline 사용
- Jenkins Pipeline 구성
- Private Registry 연동
- Jenkins Master + Slaves 구성

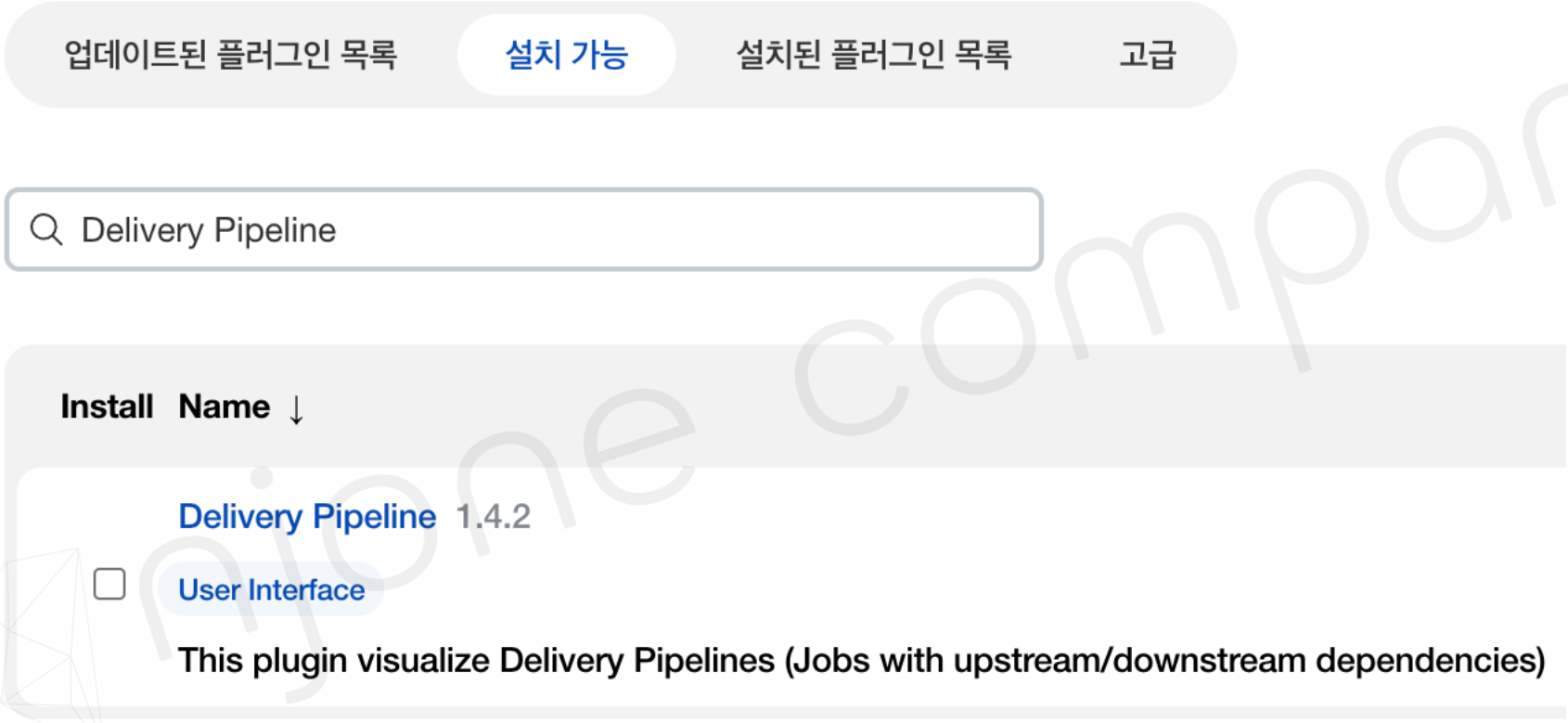
Create a Pipeline

My-First-Project → My-Second-Project → My-Third-Project

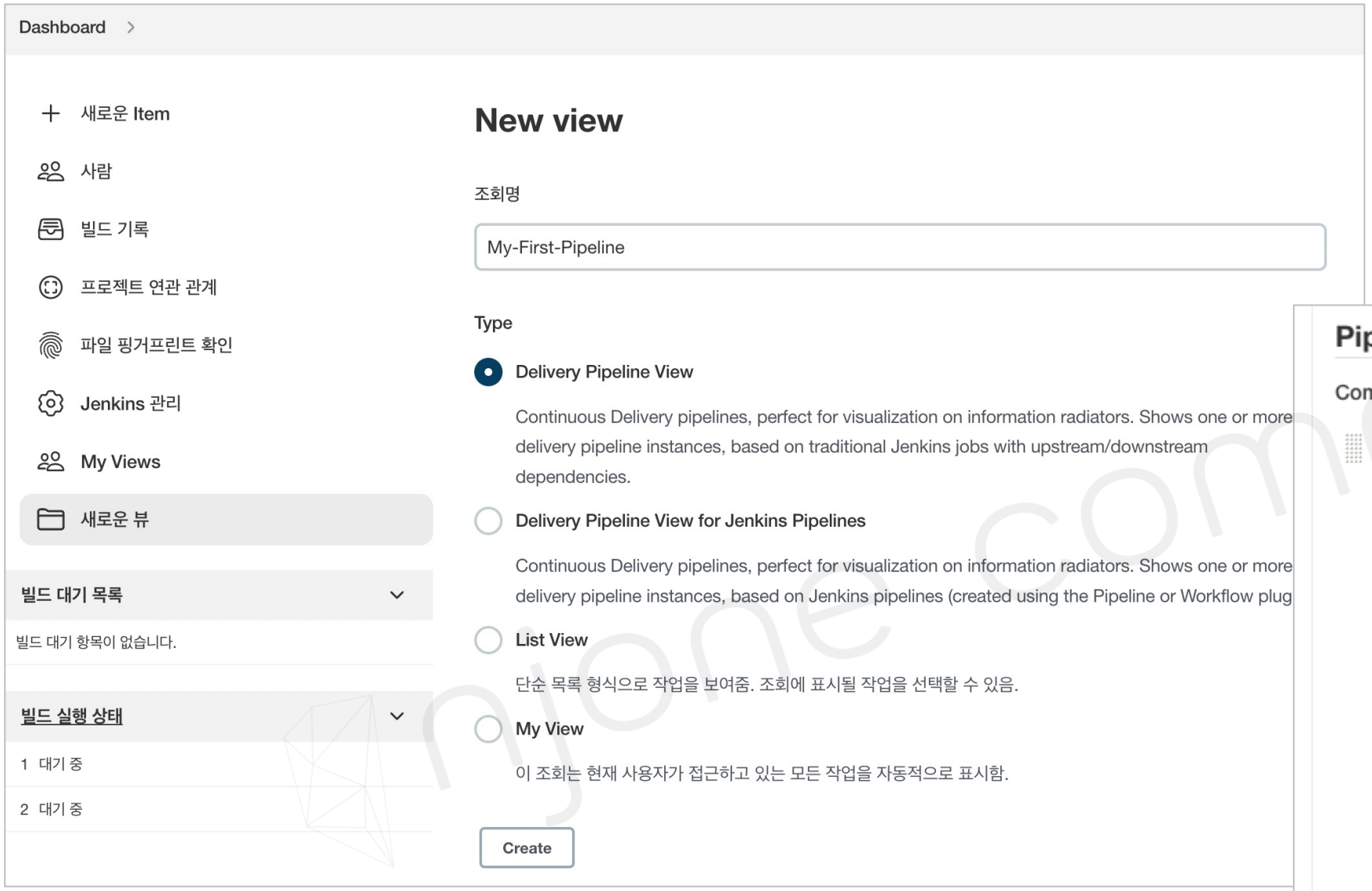
- My-First-Project
  - echo
- My-Second-Project
  - git
  - maven build
- My-Third-Project
  - git
  - maven build
  - deploy on Tomcat

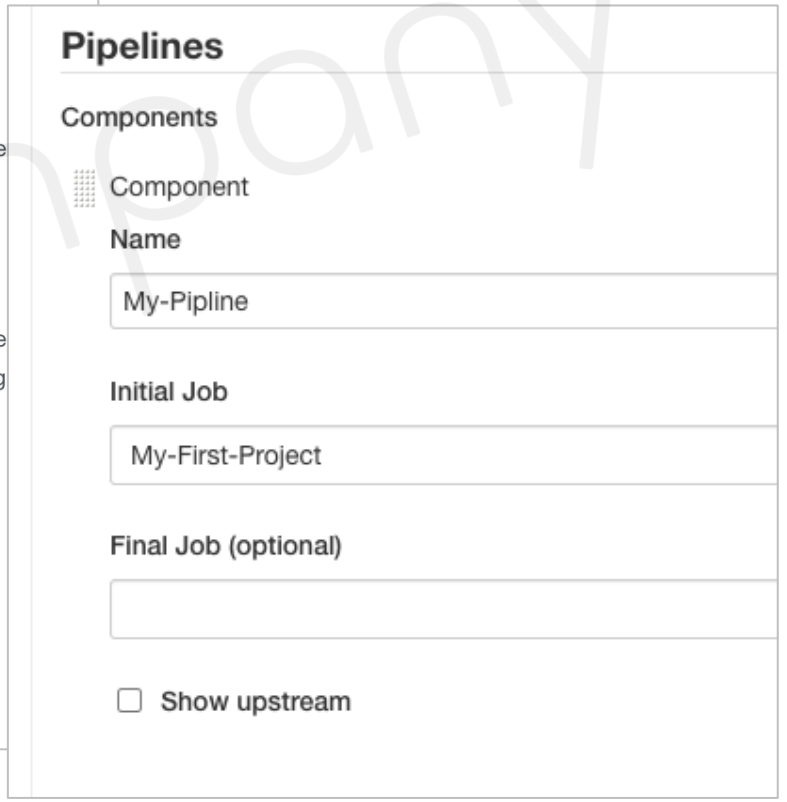
Create a Pipeline

- Manage Jenkins → Plugin Manager→ Available
  - Delivery Pipeline

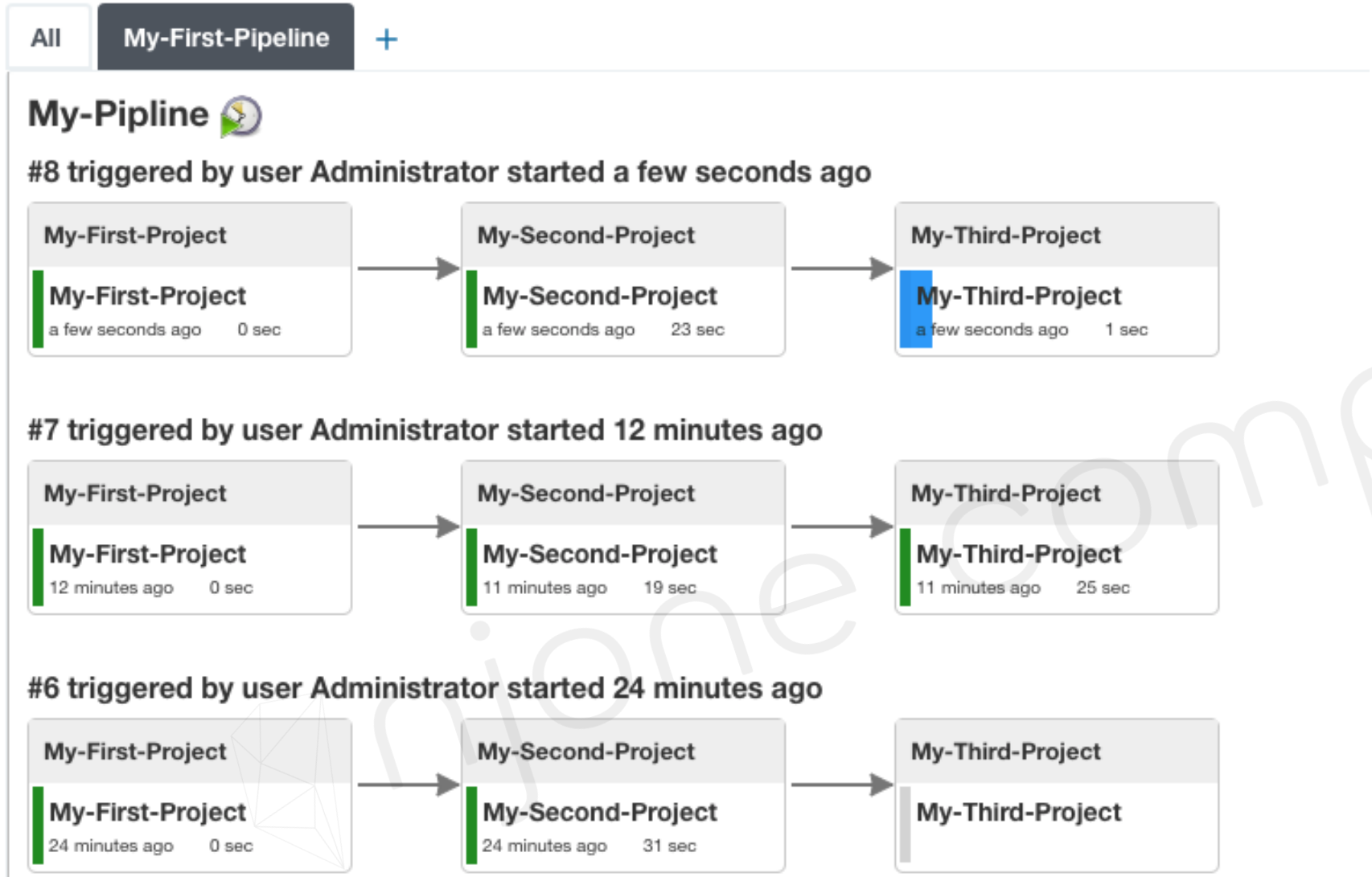


Create a Pipeline





Create a Pipeline



Jenkins Pipeline

- Declarative
- Scripted (Groovy + DSL)
- 차이점
  - 시작 시 유효성 검사 유무
  - 특정 Stage 실행 가능 여부
  - 제어문
  - Option

```
pipeline {
    agent any
    stages {
        steps('build') {
            //
        }

        steps('test') {
            //
        }

        steps('deploy') {
            //
        }
    }
}
```

```
node {
    stage('build') {
        //
    }

    stage('test') {
        //
    }

    stage('deploy') {
        //
    }
}
```

Jenkinsfile

Jenkins Pipeline

- Declarative
  - Groovy script 없이 간단하게 시작

```
pipeline {
    agent any  1)
    stages {
        steps('build') { 2)
            // 3)
        }

        steps('test') {  4)
            // 5)
        }

        steps('deploy') { 6)
            // 7)
        }
    }
}
```

1) → 실행가능한 Agent에서 Pipeline 실행

2) → build 스테이지 선언

3) → build 스테이지에 필요한 작업을 수행

4) → test 스테이지 선언

5) → test 스테이지에 필요한 작업을 수행

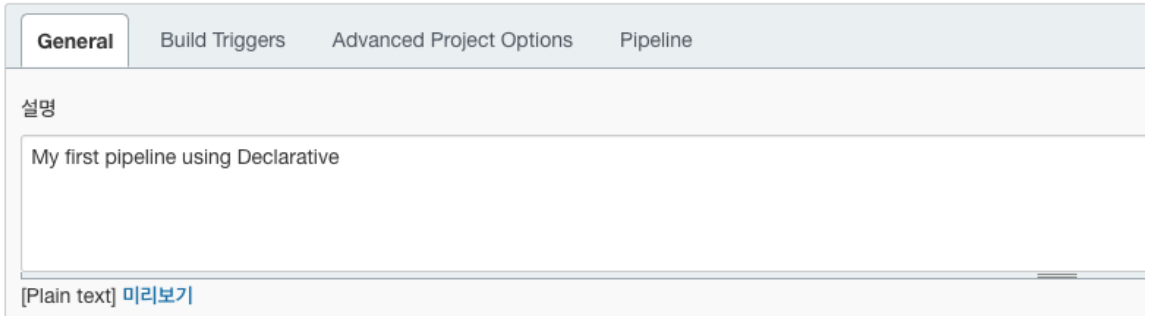
6) → deploy 스테이지 선언

7) → deploy 스테이지에 필요한 작업을 수행

Exercise 9# Jenkins Pipeline Job 1/4

njone company

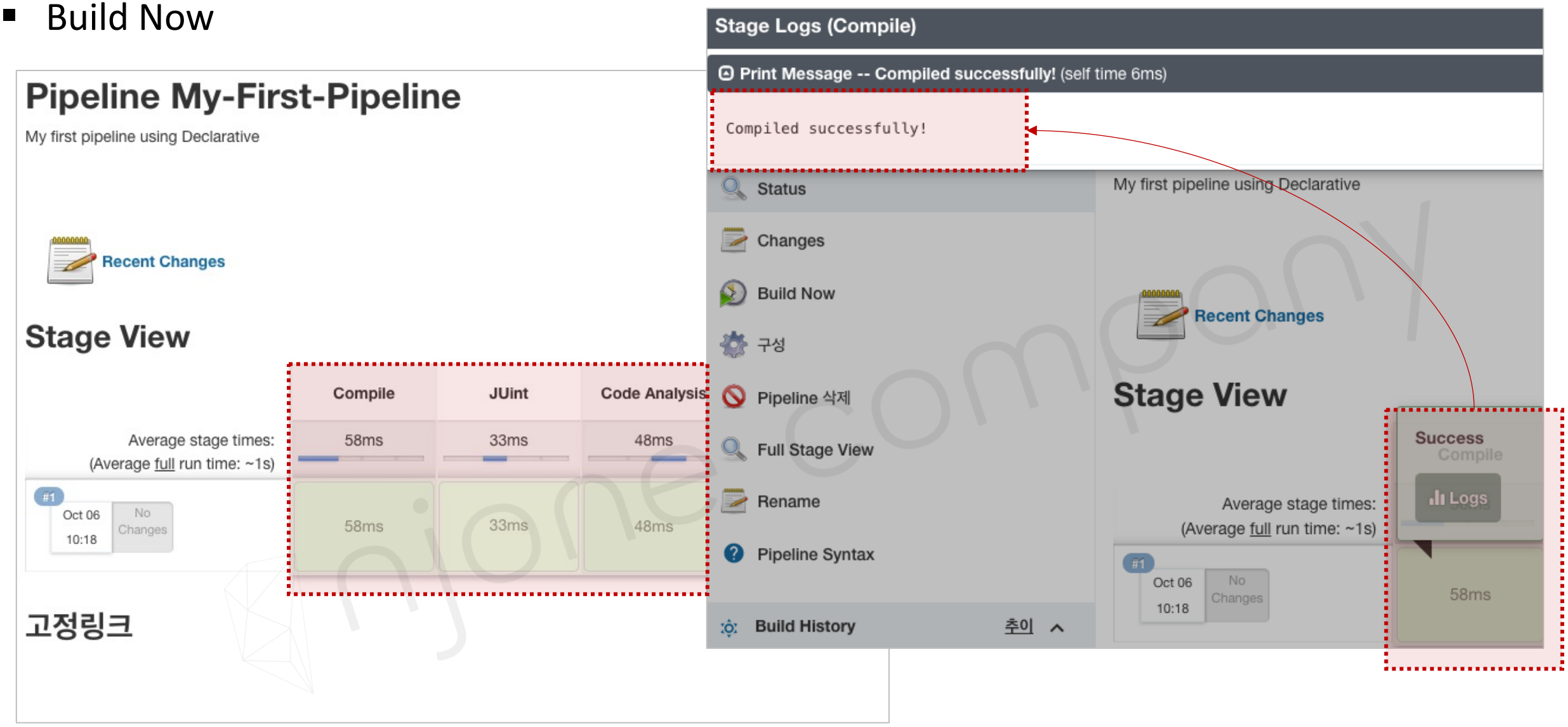
- Item name → ***My-First-Pipeline***



- Pipeline
  - Script

Exercise 9# Jenkins Pipeline Job 2/4

- Build Now



Exercise 9# Jenkins Pipeline Job 3/4

- Console Output

콘솔 출력

```
Started by user Administrator
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/My-First-Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Compile)
[Pipeline] echo
Compiled successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (JUint)
[Pipeline] echo
```

```
Deployed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Exercise 9# JenKins Pipeline Job 4/4

- Script 추가
  - ***post***

Pipeline

Definition

Pipeline script

Script

```
22      stage('Deploy') {
23          steps {
24              echo "Deployed successfully!";
25          }
26      }
27  }
28
29  post {
30      always {
31          echo 'This will always run'
32      }
33      success {
34          echo 'This will run when the run finished successfully'
35      }
36      failure {
37          echo 'This will run if failed'
38      }
39      unstable {
40          echo 'This will run when the run was marked as unstable'
41      }
42      changed {
43          echo 'This will run when the state of the pipeline has changed'
44      }
45  }
46  }
```

```
Deployed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
This will always run
[Pipeline] echo
This will run when the run finished successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Exercise 10# Jenkins Pipeline Job 1/3

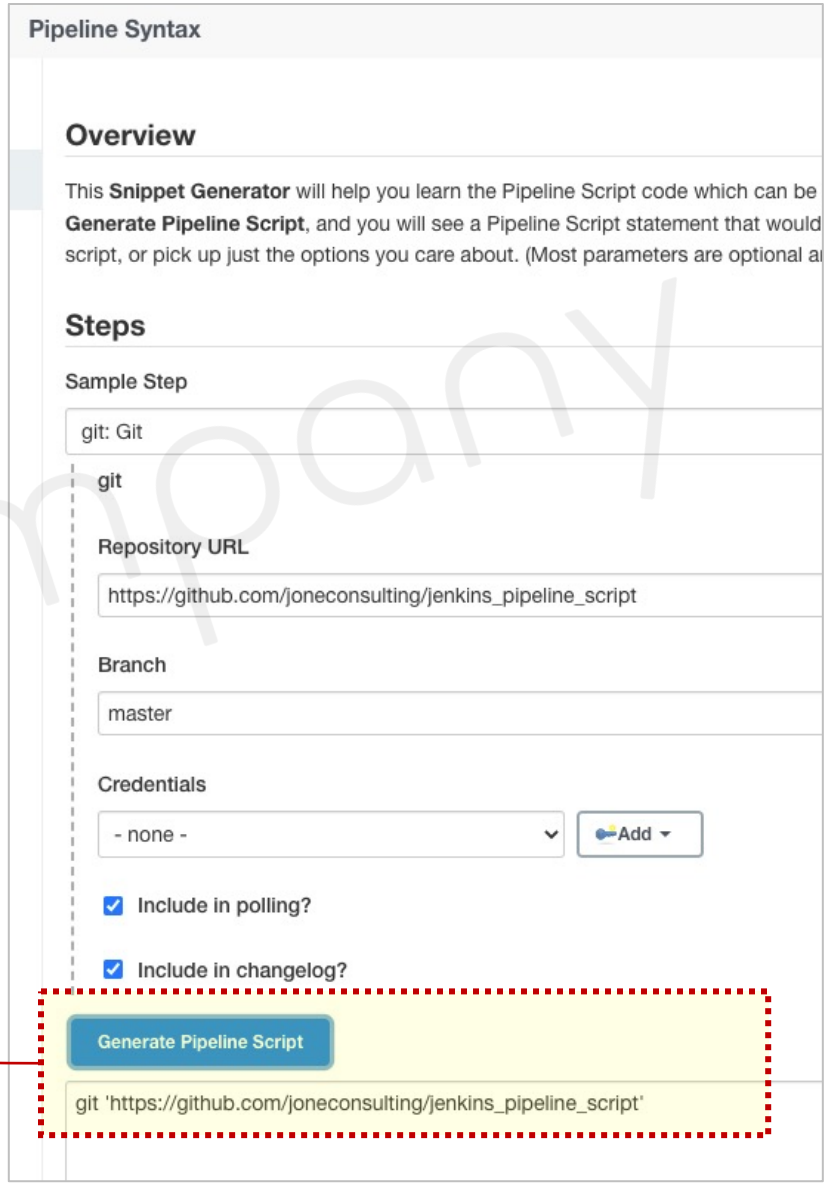
- Github에 저장된 Script 실행
  - https://github.com/joneconsulting/jenkins_pipeline_script

```
ls *.bat
build.bat   deploy.bat  quality.bat unit.bat
(base)  dowonlee  ~/Desktop/git/jenkins_pipeline_script  master
ls *.sh
build.sh   deploy.sh  quality.sh unit.sh
(base)  dowonlee  ~/Desktop/git/jenkins_pipeline_script  master
```

- Pipeline 추가
  - ***Pipeline Syntax*** 이용

Script

```
1 pipeline {
2     agent any
3     stages {
4         stage('Git clone') {
5             steps {
6                 git 'https://github.com/joneconsulting/jenkins_pipeline_script';
7             }
8         }
```



Exercise 10 # Jenkins Pipeline Job 2/3

- Build Now


```
Cloning the remote Git repository
Cloning repository https://github.com/joneconsulting/jenkins_pipeline_script
 > git init /var/jenkins_home/workspace/My-First-Pipeline # timeout=10
Fetching upstream changes from https://github.com/joneconsulting/jenkins_pipeline_script
 > git --version # timeout=10
 > git --version # 'git version 2.30.2'
 > git fetch --tags --force --progress -- https://github.com/joneconsulting/jenkins_pipeline_script
timeout=10
 > git config remote.origin.url https://github.com/joneconsulting/jenkins_pipeline_script # timeout=
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 5552317c4dbf21a896b27e707dcbe85168273377 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 5552317c4dbf21a896b27e707dcbe85168273377 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git checkout -b master 5552317c4dbf21a896b27e707dcbe85168273377 # timeout=10
Commit message: "upload batch files"
First time build. Skipping changelog.
```

Exercise 10 # Jenkins Pipeline Job 3/3

- Pipeline 수정 → Build Now

```
 9      stage('Compile') {
10          steps {
11              echo "Compiled successfully!";
12              sh './build.sh'
13          }
14      }
15
16      stage('JUint') {
17          steps {
18              echo "JUint passed successfully!";
19              sh './unit.sh'
20          }
21      }
22
23      stage('Code Analysis') {
24          steps {
25              echo "Code Analysis completed successfully!";
26              sh './quality.sh'
27          }
28      }
29
30      stage('Deploy') {
31          steps {
32              echo "Deployed successfully!";
33              sh './deploy.sh'
34          }
35      }
36  }
```

```
JUint passed successfully!
[Pipeline] sh
+ ./unit.sh
Running Unit Test Cases : Wed Oct  6 04:34:20 UTC 2021
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Code Analysis)
[Pipeline] echo
Code Analysis completed successfully!
[Pipeline] sh
+ ./quality.sh
Code Quality Check : Wed Oct  6 04:34:20 UTC 2021
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] echo
Deployed successfully!
[Pipeline] sh
+ ./deploy.sh
Deploying the Project : Wed Oct  6 04:34:21 UTC 2021
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
This will always run
[Pipeline] echo
This will run when the state of the pipeline has changed
[Pipeline] echo
This will run when the run finished successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

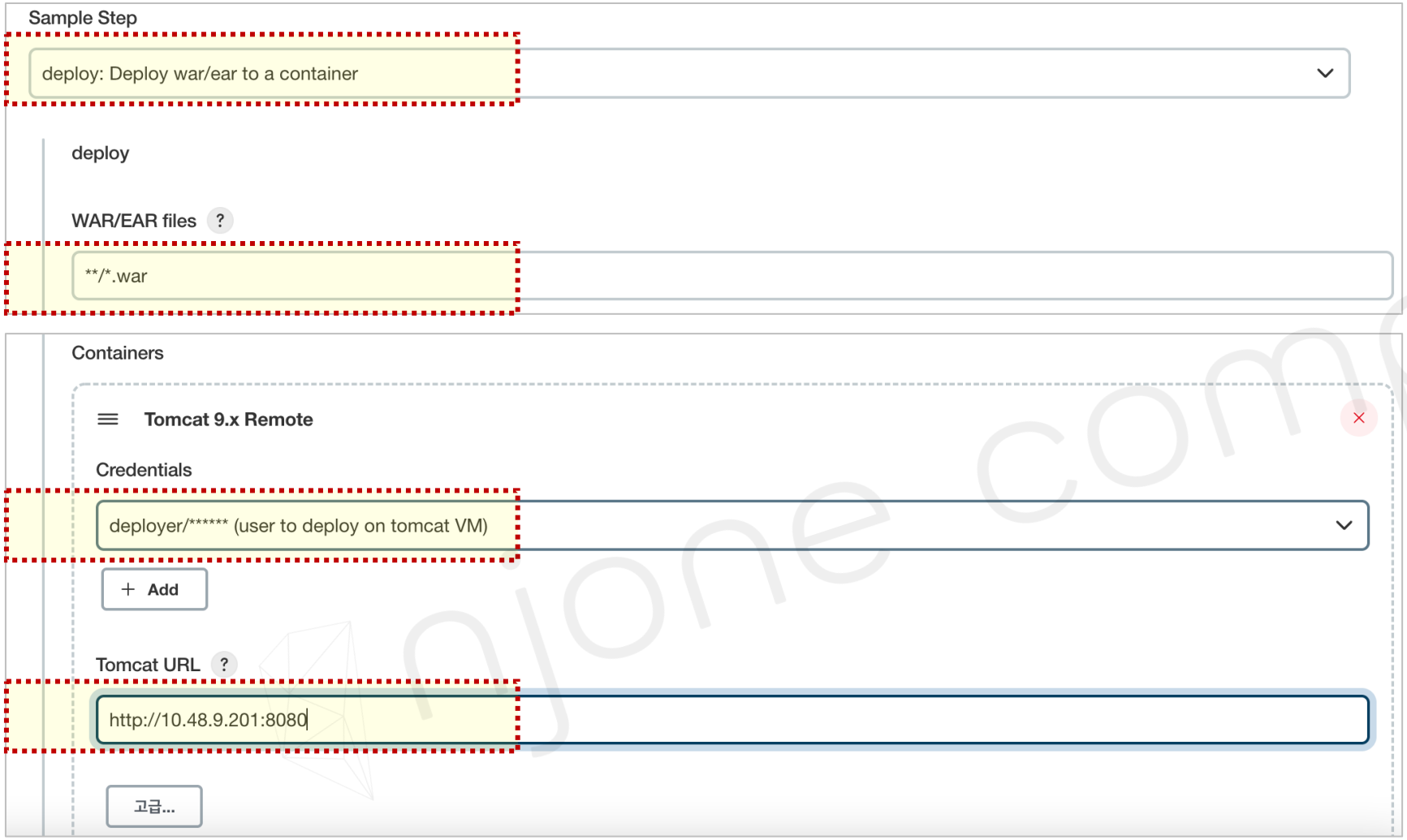
Exercise 11 # Maven build

- 이전에 작업했던 cicd-web-project 예제를 Maven 빌드 하기

```
pipeline {
    agent any
    tools {
      maven 'Maven3.8.5'
    }
    stages {
        stage('github clone') {
            steps {
                git branch: 'main', url: 'https://github.com/joneconsulting/cicd-web-project.git';
            }
        }

        stage('build') {
            steps {
                sh '''
                    echo build start
                    mvn clean compile package -DskipTests=true
                '''
            }
        }

```

Exercise 12 # Deploy on Tomcat 1/2

- Exercise 11번 작업의 결과물을 tomcat9 서버에 배포하기



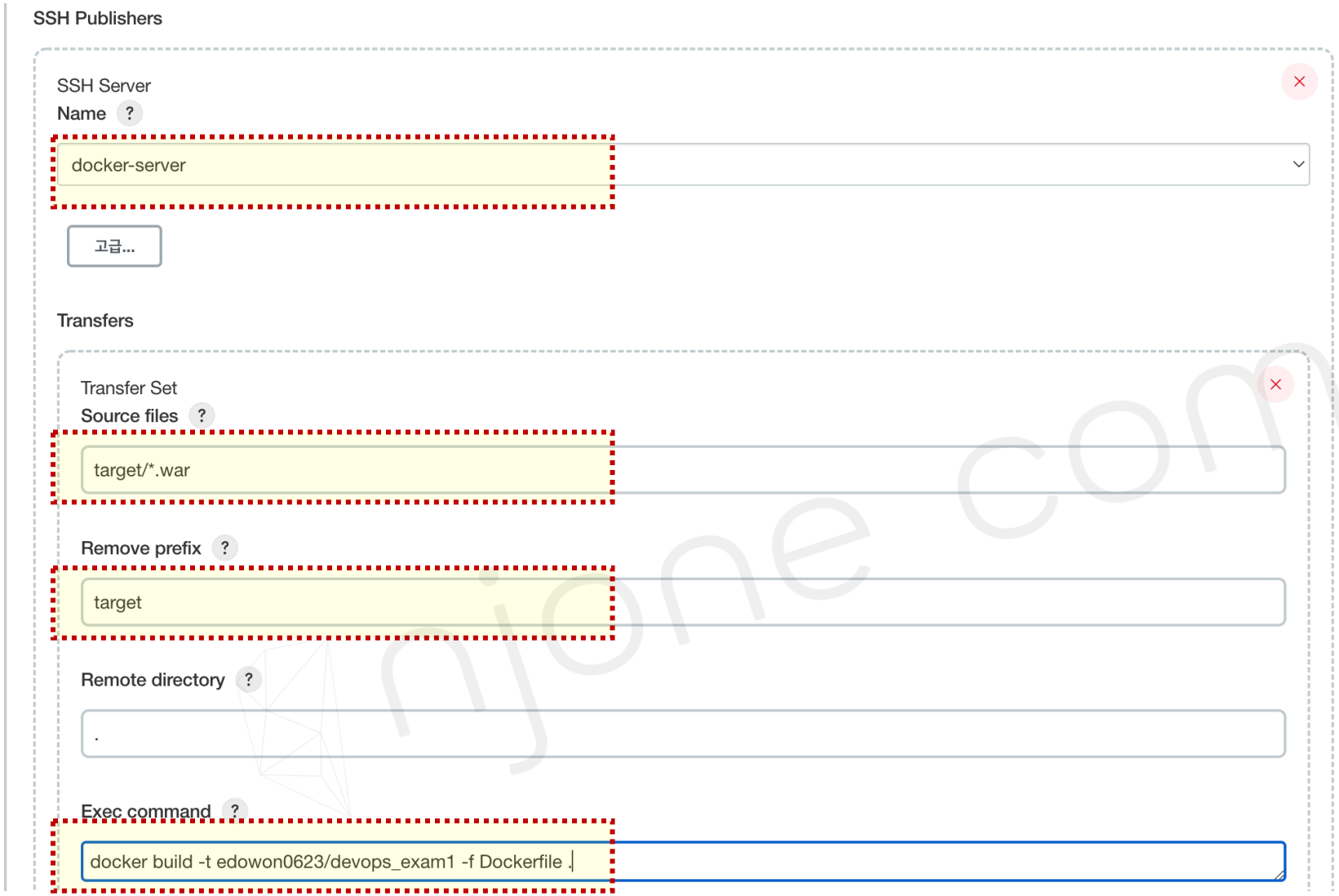
Exercise 12 # Deploy on Tomcat 2/2

- Exercise 11번 작업의 결과물을 tomcat9 서버에 배포하기

```
stage('deploy') {
        steps {
            deploy adapters: [tomcat9(credentialsId: 'deployer_user', path: '', url: 'http://10.48.9.201:8088/')], contextPath:
null, war: '**/*.war'
        }
    }
```

Exercise 13 # Deploy on Docker server 1/2

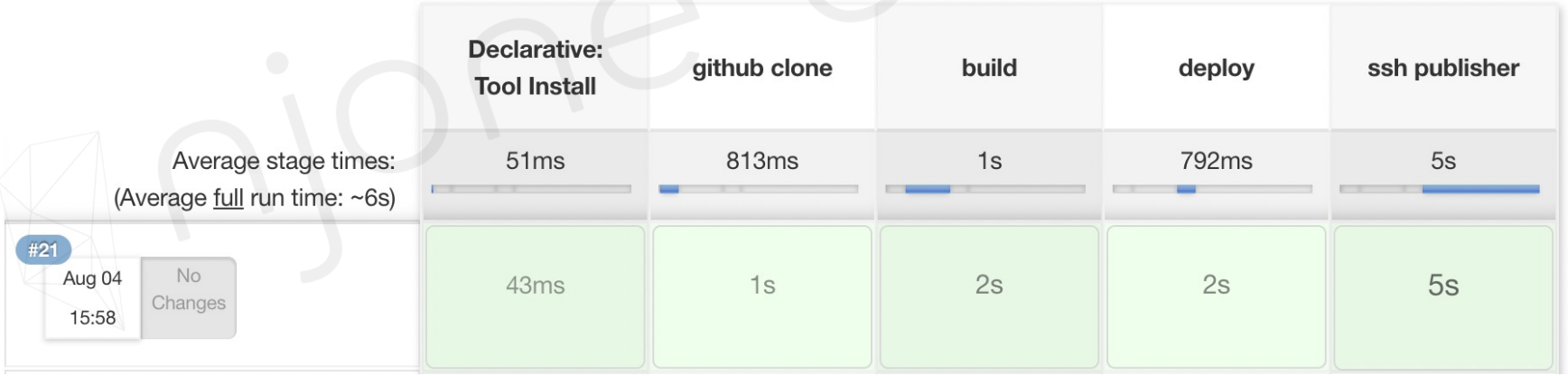
- Exercise 11번 작업의 결과물을 Docker server에 배포하기



Exercise 13 # Deploy on Docker server 2/2

- Exercise 11번 작업의 결과물을 Docker server에 배포하기

```
stage('ssh publisher') {
        steps {
            sshPublisher(publishers: [sshPublisherDesc(configName: 'docker-server', transfers: [sshTransfer(cleanRemote:
false, excludes: '', execCommand: 'docker build -t edowon0623/devops_exam1 -f Dockerfile .', execTimeout: 120000,
flatten: false, makeEmptyDirs: false, noDefaultExcludes: false, patternSeparator: '[, ]+', remoteDirectory: '.',
remoteDirectorySDF: false, removePrefix: 'target', sourceFiles: 'target/*.war')], usePromotionTimestamp: false,
useWorkspaceInPromotion: false, verbose: false)])
        }
    }
```



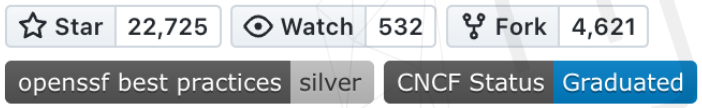
Docker Regitry를 대신하는 기술

Harbor

- https://goharbor.io/
- https://goharbor.io/docs/2.10.0/install-config/

What is Harbor?

Harbor is an open source registry that secures artifacts with policies and role-based access control, ensures images are scanned and free from vulnerabilities, and signs images as trusted. Harbor, a CNCF Graduated project, delivers compliance, performance, and interoperability to help you consistently and securely manage artifacts across cloud native compute platforms like Kubernetes and Docker.



Software	Version
Docker Engine	20.10.10-ce+ or higher
Docker Compose	1.18.0+
OpenSSL	Latest

Port	Protocol
443	HTTPS
4443	HTTPS
80	HTTP

인증서 설치

1) CA Certificates 생성
2) Server Certificates 생성
3) SAN(Subject Alternative Name) 등록
4) Docker Engine Certificate 업데이트

인증서 설치

1) CA Certificates 생성

- Root CA 비밀키 생성 → 공개키 생성

```
$ openssl genrsa –out ca.key 4096
```

```
$ openssl req -x509 -new -nodes -sha512 -days 365 -key ca.key -out ca.crt
```

2) Server Certificates 생성

- 서버의 인증서를 위한 비밀키 생성 → CSR(Certificate Signing Request) 파일 생성

```
$ openssl genrsa –out server.key 4096
```

```
$ openssl req -new -sha512 -key server.key -out server.csr
```

인증서 설치

3) SAN(Subject Alternative Name) 등록

- CSR을 이용하여 SAN에 도메인 정보 추가 → 서버의 인증키 생성

```
$ vi v3ext.cnf
```

```
[root@45ad8a699362 certs]# cat v3ext.cnf
subjectAltName = IP:172.30.11.76,IP:127.0.0.1
```

Host PC IP address

```
$ openssl x509 -req -sha512 -days 365 \
    -extfile v3ext.cnf \
    -CA ca.crt -CAkey ca.key -CAcreateserial \
    -in server.csr \
    -out server.crt
```

인증서 설치

4) Docker Engine Certificate 업데이트

- Docker Engine에서 사용 될 cert 파일 생성

```
$ openssl x509 -inform PEM -in server.crt -out server.cert
```

- Docker Engine에 인증서 파일 등록

```
$ mkdir -p /etc/docker/certs.d/server
$ cp server.cert /etc/docker/certs.d/server/
$ cp server.key /etc/docker/certs.d/server/
$ cp ca.crt /etc/docker/certs.d/server/
```

Harbor 설치

1) Harbor package 다운로드

- https://github.com/goharbor/harbor/releases

```
$ wget https://github.com/goharbor/harbor/releases/download/v2.2.2/harbor-offline-installer-v2.2.2.tgz
```

```
$ tag harbor-offline-installer-v2.2.2.tgz
```

2) harbor.yml 파일 생성 → 설정

```
$ cp harbor.yml.tmp harbor.yml
```

```
$ vi harbor.yml
```

Host PC IP address

```
hostname: 172.30.11.76

http:
  port: 80

https:
  port: 443
  certificate: /etc/docker/certs.d/server/server.cert
  private_key: /etc/docker/certs.d/server/server.key
```

Harbor 설치

3) Deploy

```
$ ./prepare
$ ./install.sh
```

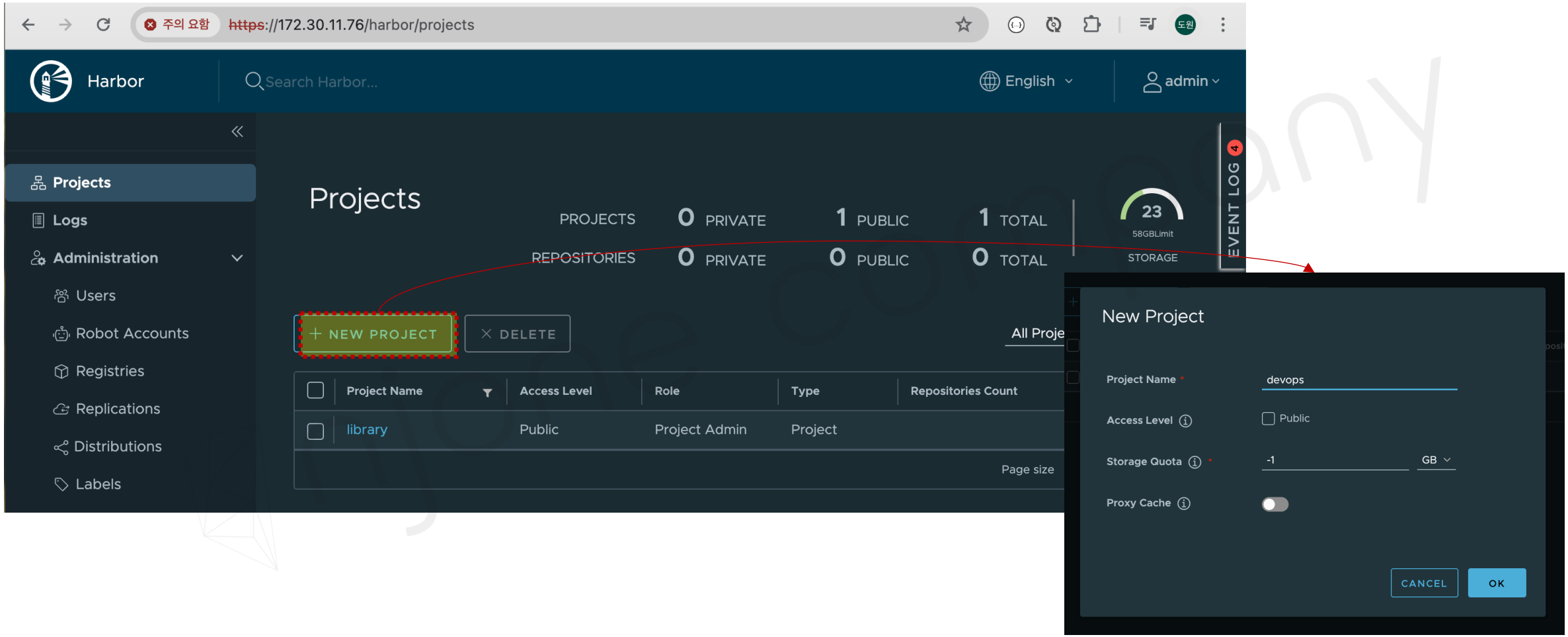
```
[root@45ad8a699362 harbor]# docker-compose ps
NAME                COMMAND                  SERVICE             STATUS              PORTS
harbor-core         "/harbor/entrypoint.…"   core                running (healthy)
harbor-db           "/docker-entrypoint.…"   postgresql          running (healthy)
harbor-jobservice   "/harbor/entrypoint.…"   jobservice          running (healthy)
harbor-log          "/bin/sh -c /usr/loc…"   log                 running (healthy)   127.0.0.1:1514->1051
harbor-portal       "nginx -g 'daemon of…"   portal              running (healthy)
nginx               "nginx -g 'daemon of…"   proxy               running (healthy)   0.0.0.0:80->8080/tcp
redis               "redis-server /etc/r…"   redis               running (healthy)
registry            "/home/harbor/entryp…"   registry            running (healthy)
registryctl         "/home/harbor/start.…"   registryctl         running (healthy)
```

Harbor 사용

1) Project 생성
2) *Container#1]* docker login –u admin <Harbor IP address>
3) *Container#1]* docker push <IMAGE>
4) *Container#2]* docker login –u user1 <Harbor IP address>
5) *Container#2]* docker pull <IMAGE>

Harbor 사용

1) Project 생성



Harbor 사용

2) *Container#1]* docker login –u admin <Harbor IP address>

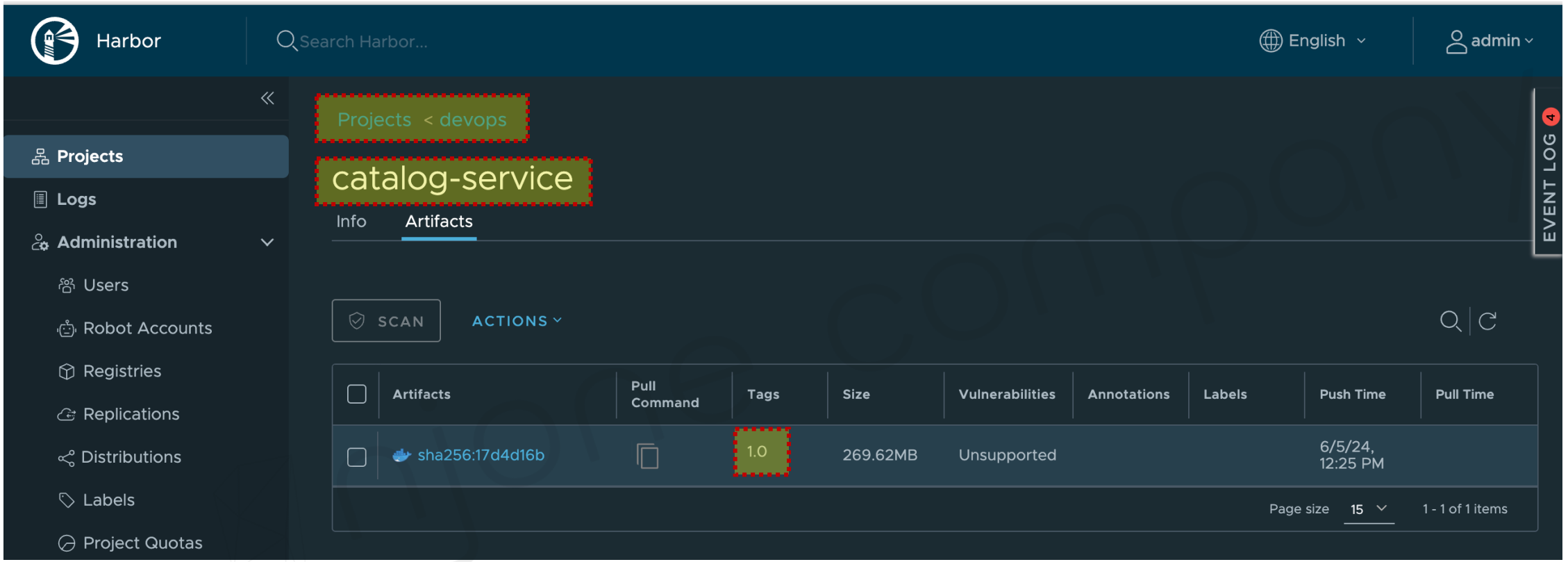
```
[root@45ad8a699362 harbor]# docker login -u admin 172.30.11.76
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@45ad8a699362 harbor]#
```

3) *Container#1]* docker push <IMAGE>

```
[root@45ad8a699362 harbor]# docker tag edowon0623/catalog-service:1.0 172.30.11.76/devops/catalog-service:1.0
[root@45ad8a699362 harbor]#
[root@45ad8a699362 harbor]# docker push 172.30.11.76/devops/catalog-service:1.0
The push refers to repository [172.30.11.76/devops/catalog-service]
b8029593adec: Layer already exists
3d3fdb9815af: Layer already exists
08664b16f94c: Layer already exists
9eb82f04c782: Layer already exists
1.0: digest: sha256:17d4d16bc218d5d686191edf72cc69a0d59b230d7a0c698dce915506d4ee30b5 size: 1165
```

Harbor 사용

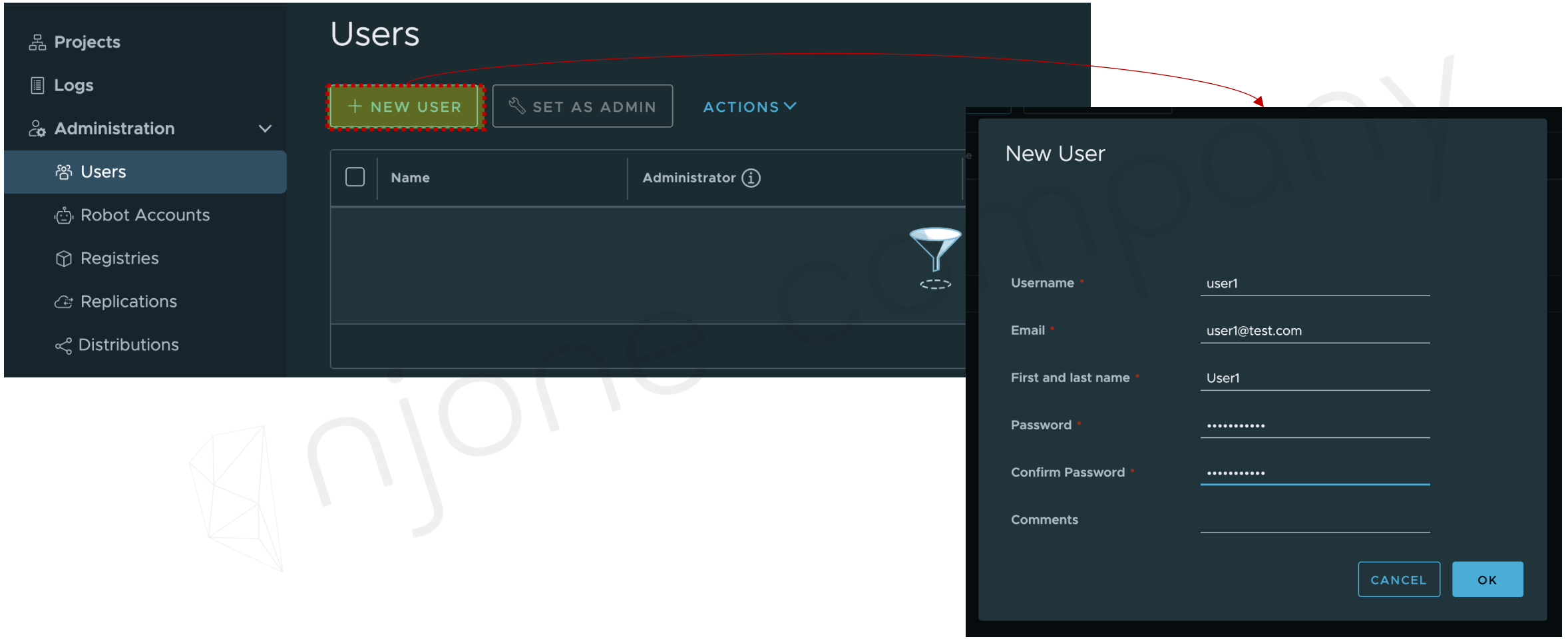
3) *Container#1]* docker push <IMAGE>



Docker Regitry를 대신하는 기술

Harbor 사용

4) *Container#2]* docker login –u user1 <Harbor IP address>



Harbor 사용

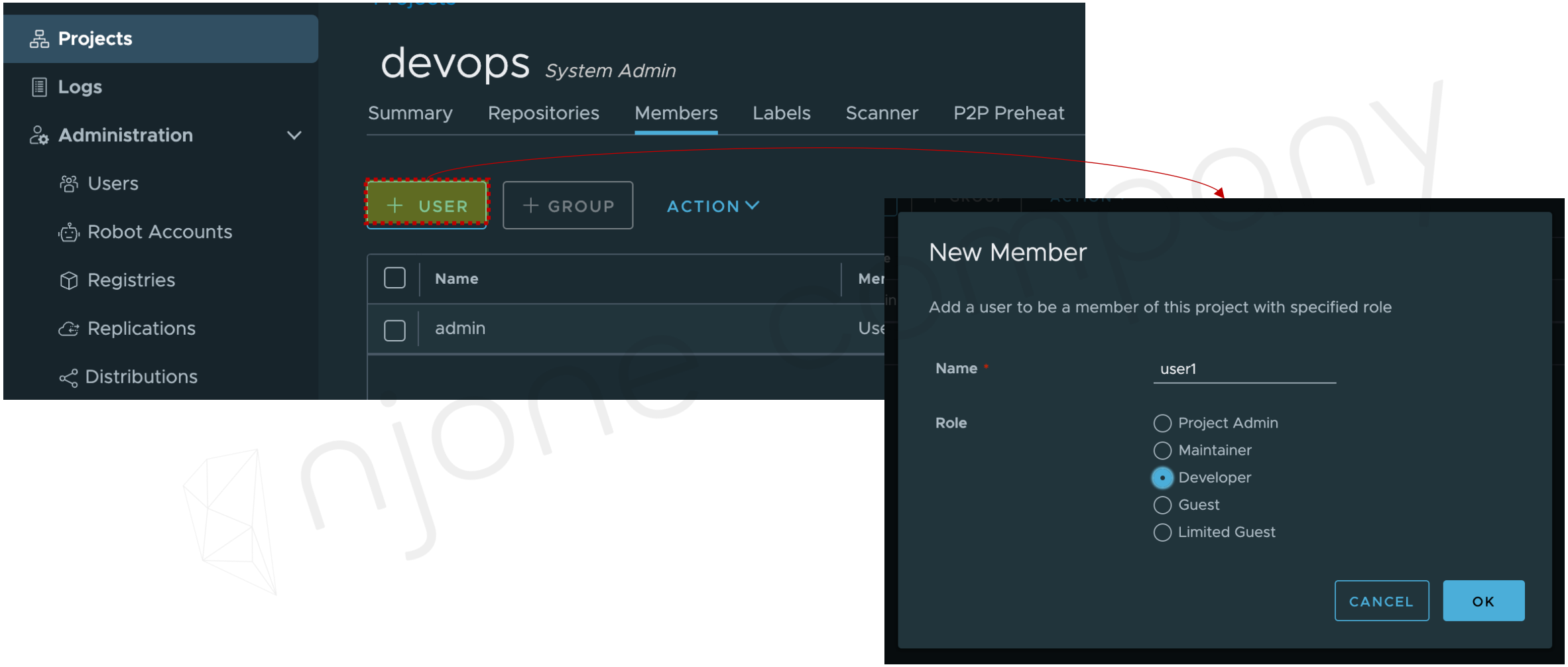
4) *Container#2]* docker login –u user1 <Harbor IP address>

```
[root@33c30115eced ~]# docker login -u user1 172.30.11.76
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
[root@33c30115eced ~]#
```

```
[root@33c30115eced ~]# docker pull 172.30.11.76/devops/catalog-service:1.0
Error response from daemon: unauthorized: unauthorized to access repository: devops/catalog-service,
action: pull: unauthorized to access repository: devops/catalog-service, action: pull
[root@33c30115eced ~]#
```

Harbor 사용

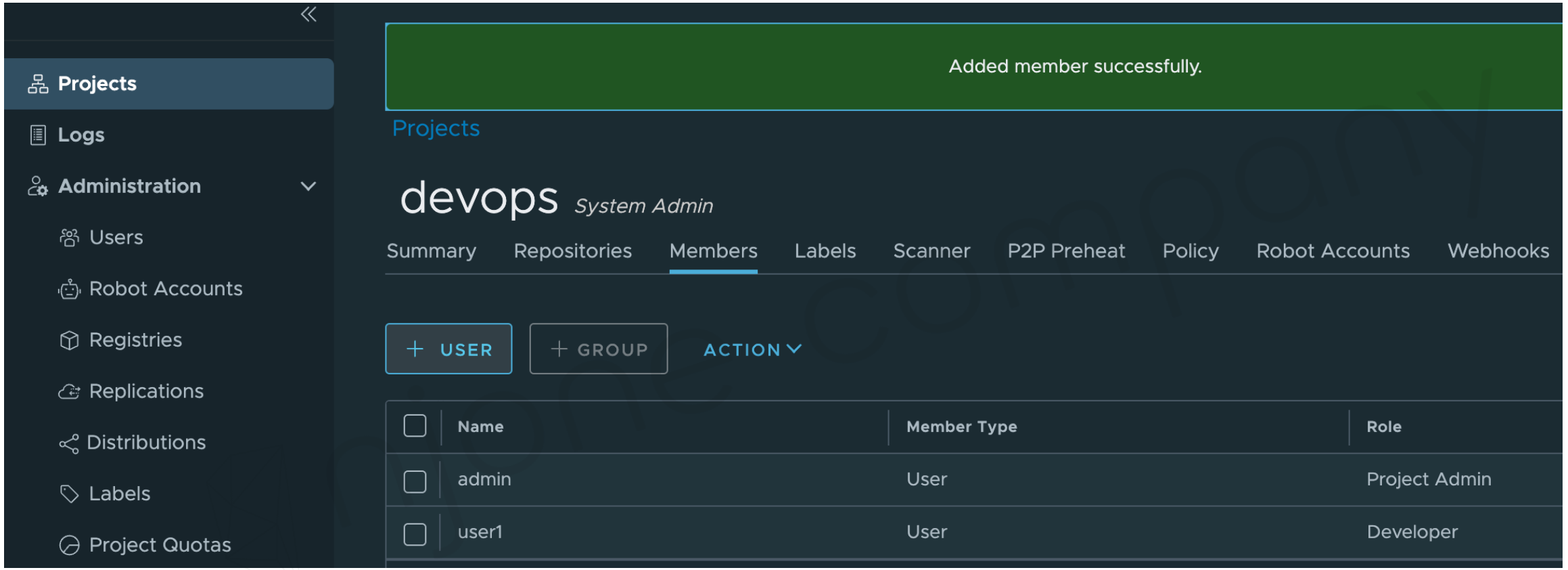
5) *Container#2]* docker pull <IMAGE>



Docker Regitry를 대신하는 기술

Harbor 사용

5) *Container#2]* docker pull <IMAGE>



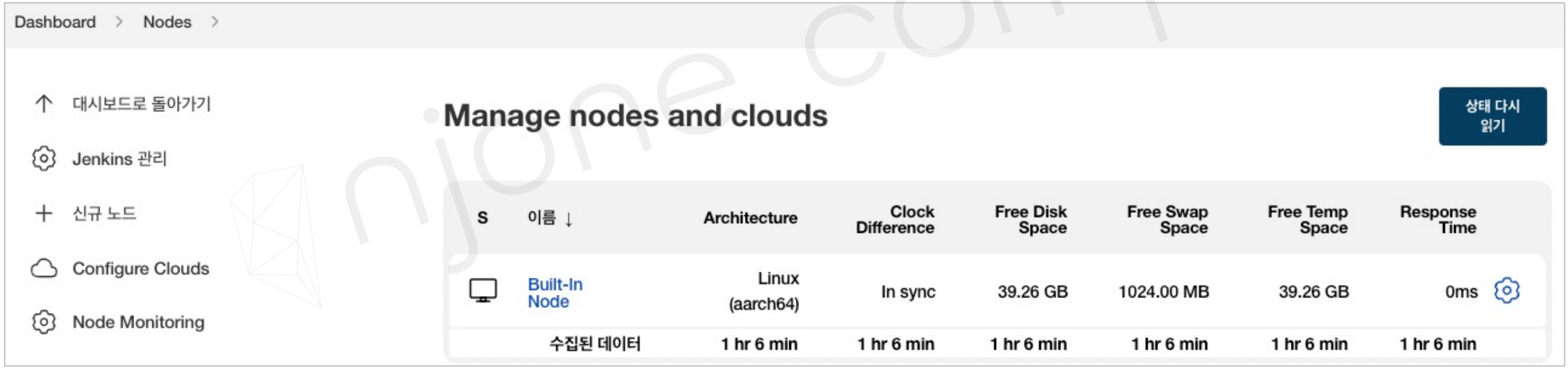
Harbor 사용

5) Container#2] docker pull <IMAGE>

```
[root@33c30115eced ~]# docker pull 172.30.11.76/devops/catalog-service:1.0
1.0: Pulling from devops/catalog-service
45b42c59be33: Pull complete
a91c0c19c848: Pull complete
0a37071eae8e: Pull complete
17bec8782113: Pull complete
Digest: sha256:17d4d16bc218d5d686191edf72cc69a0d59b230d7a0c698dce915506d4ee30b5
Status: Downloaded newer image for 172.30.11.76/devops/catalog-service:1.0
172.30.11.76/devops/catalog-service:1.0
[root@33c30115eced ~]#
```

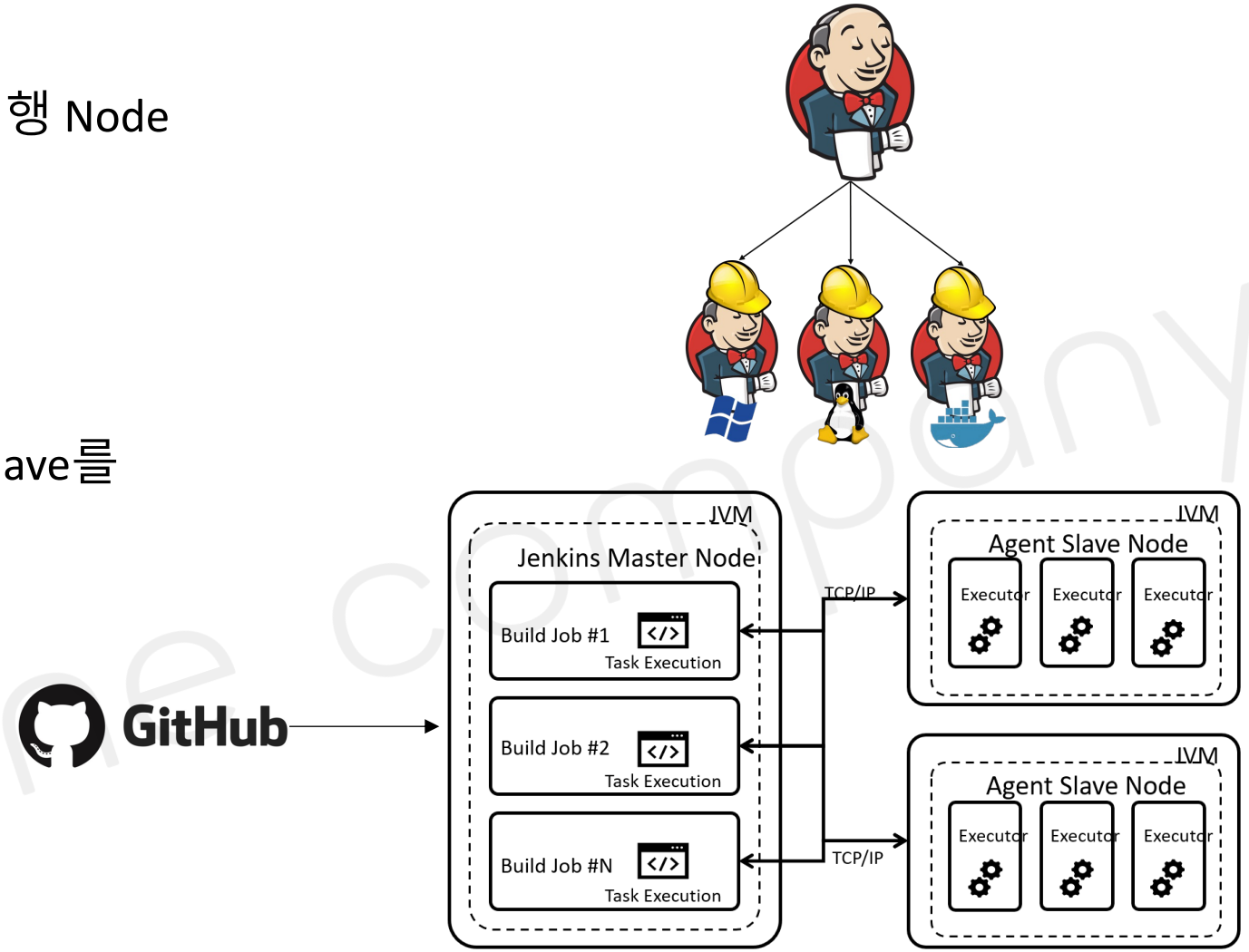
Jenkins Master + Slaves

- Jenkins Master
  - 기본 Jenkins 서버 → Master node
  - 빌드 작업 예약 및 빌드 실행
  - Slave에게 Build 할 Job 전달
  - Slave 모니터링
  - 빌드 결과 기록



Jenkins Master + Slaves

- Jenkins Slave
  - Remote에서 실행되는 Jenkins 실행 Node
  - Jenkins Master의 요청 처리
  - Master로부터 전달된 Job 실행
  - 다양한 운영체제에서 실행 가능
  - Jenkins 프로젝트 생성 시 특정 Slave를 선택하여 실행 가능



https://digitalvarys.com/how-to-configure-jenkins-master-slave-setup/

Exercise 17 # Add a slave node 1/3

- 새로운 Server 추가

Windows, MacOS)

```
$ docker run --privileged --name jenkins-node1 -itd -p 30022:22 \
             -e container=docker -v /sys/fs/cgroup:/sys/fs/cgroup --cgroupns=host \
             edowon0623/docker:latest /usr/sbin/init
```

Apple silicon, m1)

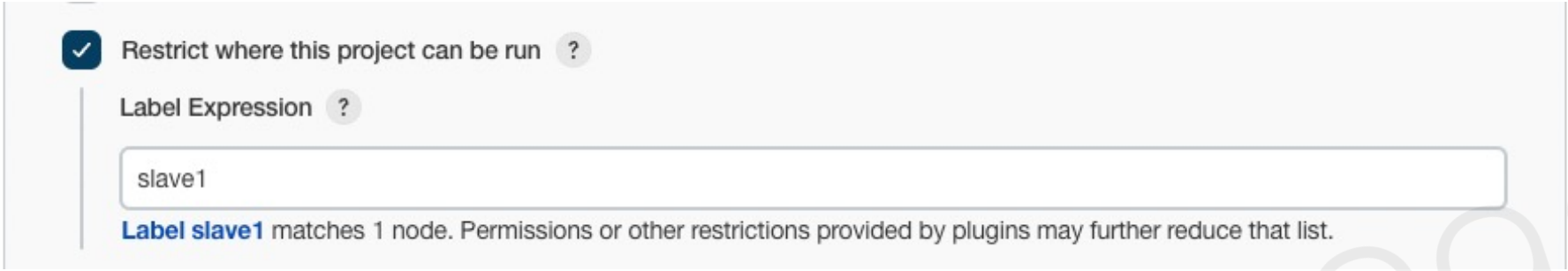
```
$ docker run --privileged --name jenkins-node1 -itd -p 30022:22 \
             -e container=docker -v /sys/fs/cgroup:/sys/fs/cgroup --cgroupns=host \
             edowon0623/docker-server:m1 /usr/sbin/init
```

Exercise 17 # Add a slave node 2/3

- Manage Jenkins → Manage Nodes → New Node
  - Node Name: ***slave1***
  - Description: ***Add a server as a slave machine***
  - Number of executors: ***5***
  - Remote root directory: ***/root/slave1***
  - Labels: ***slave1***
  - Usage: ***Use this node as much as possible***
  - Launch method: ***Launch agents via SSH***
    - Host: ***[Slave server IP address] es) 172.17.0.3***
    - Port: ***22***
    - Credentials: ***root/P@ssw0rd***

Exercise 17 # Add a slave node 3/3

- My-First-Project 수정
  - ***Restrict where this project can be run 선택***



콘솔 출력

```
Started by user Administrator
Running as SYSTEM
Building remotely on slave1 in workspace /root/slave1/workspace/My-First-Project
[My-First-Project] $ /bin/sh -xe /tmp/jenkins9356691432887153209.sh
+ echo 'Welcome to my first project using Jenkins'
Welcome to my first project using Jenkins
+ javac -version
javac 11.0.13
Triggering a new build of My-Second-Project
Finished: SUCCESS
```

```
[root@897f059a1ce0 workspace]# pwd
/root/slave1/workspace
[root@897f059a1ce0 workspace]# ls -al
total 16
drwxr-xr-x 4 root root 4096 Aug  5 05:05 .
drwxr-xr-x 5 root root 4096 Aug  5 05:06 ..
drwxr-xr-x 2 root root 4096 Aug  5 05:05 My-First-Project
drwxr-xr-x 5 root root 4096 Aug  5 05:06 My-Second-Project
```

Exercise 18 # Execute a pipeline on slaves 1/5

- New Item → Pipeline → My-Second-Pipeline
  - Copy from: ***My-First-Pipeline***

```
pipeline {
    agent {
        label 'slave1'
    }
    tools {
      maven 'Maven3.8.5'
    }
    stages {
        stage('github clone') {
            steps {
                git branch: 'main', url: 'https://github.com/joneconsulting/cicd-web-project.git';
            }
        }
    }
}
```

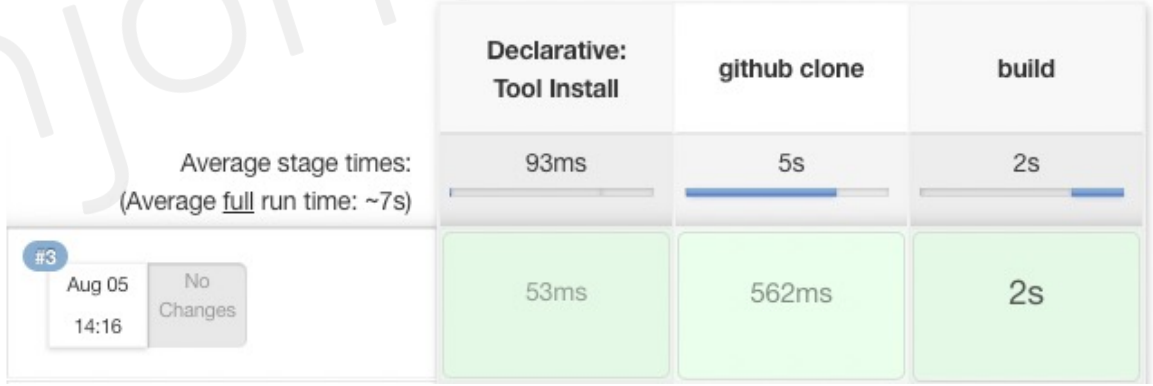
- Build Now

Exercise 18 # Execute a pipeline on slaves 2/5

- Pipeline 수정
  - ***Build Stage 추가***

```
...
        stage('build') {
            steps {
                sh '''
                    echo build start
                    mvn clean compile package -DskipTests=true
                '''
            }
        }
...
```

- Build Now



Exercise 18 # Execute a pipeline on slaves 3/5

- 새로운 Server 추가 (Slave2)

Windows, MacOS)

```
$ docker run --privileged --name jenkins-node2 -itd -p 40022:22 \
             -e container=docker -v /sys/fs/cgroup:/sys/fs/cgroup --cgroupns=host \
             edowon0623/docker:latest /usr/sbin/init
```

Apple silicon, m1)

```
$ docker run --privileged --name jenkins-node2 -itd -p 40022:22 \
             -e container=docker -v /sys/fs/cgroup:/sys/fs/cgroup --cgroupns=host \
             edowon0623/docker-server:m1 /usr/sbin/init
```

Exercise 18 # Execute a pipeline on slaves 4/5

- Manage Jenkins → Manage Nodes → New Node
  - Node Name: ***slave2***
  - Description: ***Add a server as the second slave machine***
  - Number of executors: ***2***
  - Remote root directory: ***/root/slave2***
  - Labels: ***slave2***
  - Usage: ***Use this node as much as possible***
  - Launch method: ***Launch agents via SSH***
    - Host: ***[Slave server IP address] es) 172.17.0.4***
    - Port: ***22***
    - Credentials: ***root/P@ssw0rd***

Exercise 18 # Execute a pipeline on slaves 4/5

- 정상 접속 시

Manage nodes and clouds

상태 다시 읽기

S	이름 ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (aarch64)	In sync	38.53 GB	1024.00 MB	38.53 GB	0ms
	slave1	Linux (aarch64)	In sync	38.53 GB	1024.00 MB	38.53 GB	30ms
	slave2	Linux (aarch64)	In sync	38.53 GB	1024.00 MB	38.53 GB	27ms
	수집된 데이터	47 sec	47 sec	47 sec	47 sec	47 sec	47 sec

Exercise 18 # Execute a pipeline on slaves 4/5

- 접속 오류 발생 시

```
Warning: no key algorithms provided; JENKINS-42959 disabled
SSHLauncher{host='172.17.0.4', port=22, credentialsId='root', jvmOptions='', javaPath='', prefixStartSlaveCmd='',
suffixStartSlaveCmd='', launchTimeoutSeconds=60, maxNumRetries=10, retryWaitTime=15,
sshHostKeyVerificationStrategy=hudson.plugins.sshslaves.verifiers.KnownHostsFileKeyVerificationStrategy,
tcpNoDelay=true, trackCredentials=true}
[08/05/22 05:22:17] [SSH] Opening SSH connection to 172.17.0.4:22.
Searching for 172.17.0.4 in /var/jenkins_home/.ssh/known_hosts
Searching for 172.17.0.4:22 in /var/jenkins_home/.ssh/known_hosts
[08/05/22 05:22:17] [SSH] WARNING: No entry currently exists in the Known Hosts file for this host. Connections
will be denied until this new host and its associated key is added to the Known Hosts file.
Key exchange was not finished, connection is closed.
SSH Connection failed with IOException: "Key exchange was not finished, connection is closed.", retrying in 15
seconds. There are 10 more retries left.
```

Exercise 18 # Execute a pipeline on slaves 5/5

njone company

- Pipeline 수정
  - ***Agent 수정***

```
pipeline {
    agent {
        label 'slave2'
    } …
```

- Build Now

콘솔 출력

```
Started by user Administrator
[Pipeline] Start of Pipeline
[Pipeline] node
Running on slave2 in /root/slave2/workspace/My-Second-Pipeline
[Pipeline] {
```

	Declarative: Tool Install	github clone	build
Average stage times: (Average full run time: ~40s)	2s	5s	33s
#6 Aug 05 14:47 No Changes	45ms	680ms	1s

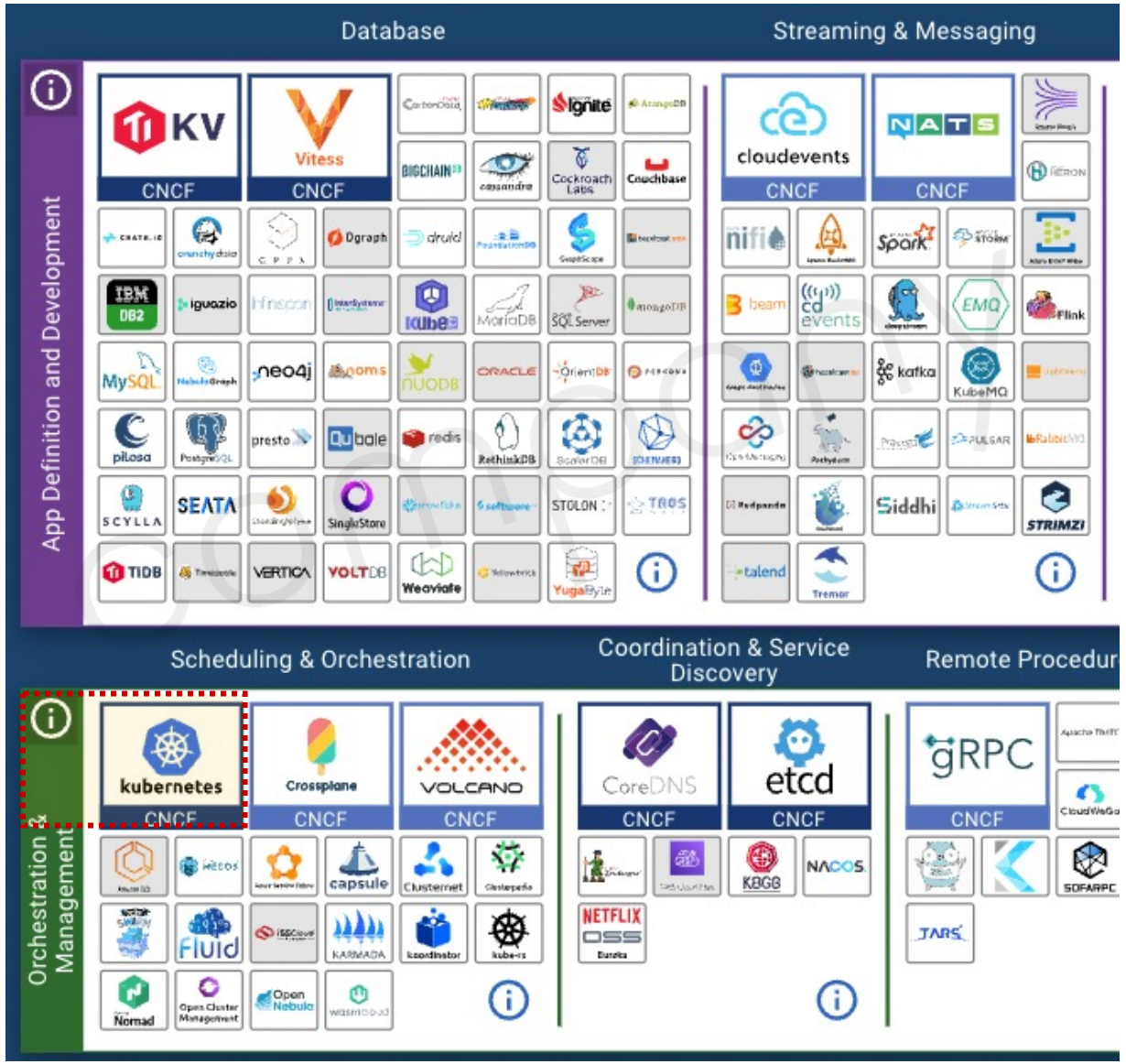
Section 5.

Kubernetes를 활용한 CD 운영환경 구축

- Kubernetes 환경에 배포
- ArgoCD

CNCF Cloud Native Interactive Landscape

- https://landscape.cncf.io/
- Kubernetes(K8s)
  - 오픈소스 기반의 컨테이너화 된 애플리케이션(워크로드와 서비스)의 자동 배포, 스케일링 등을 제공하는 관리 플랫폼

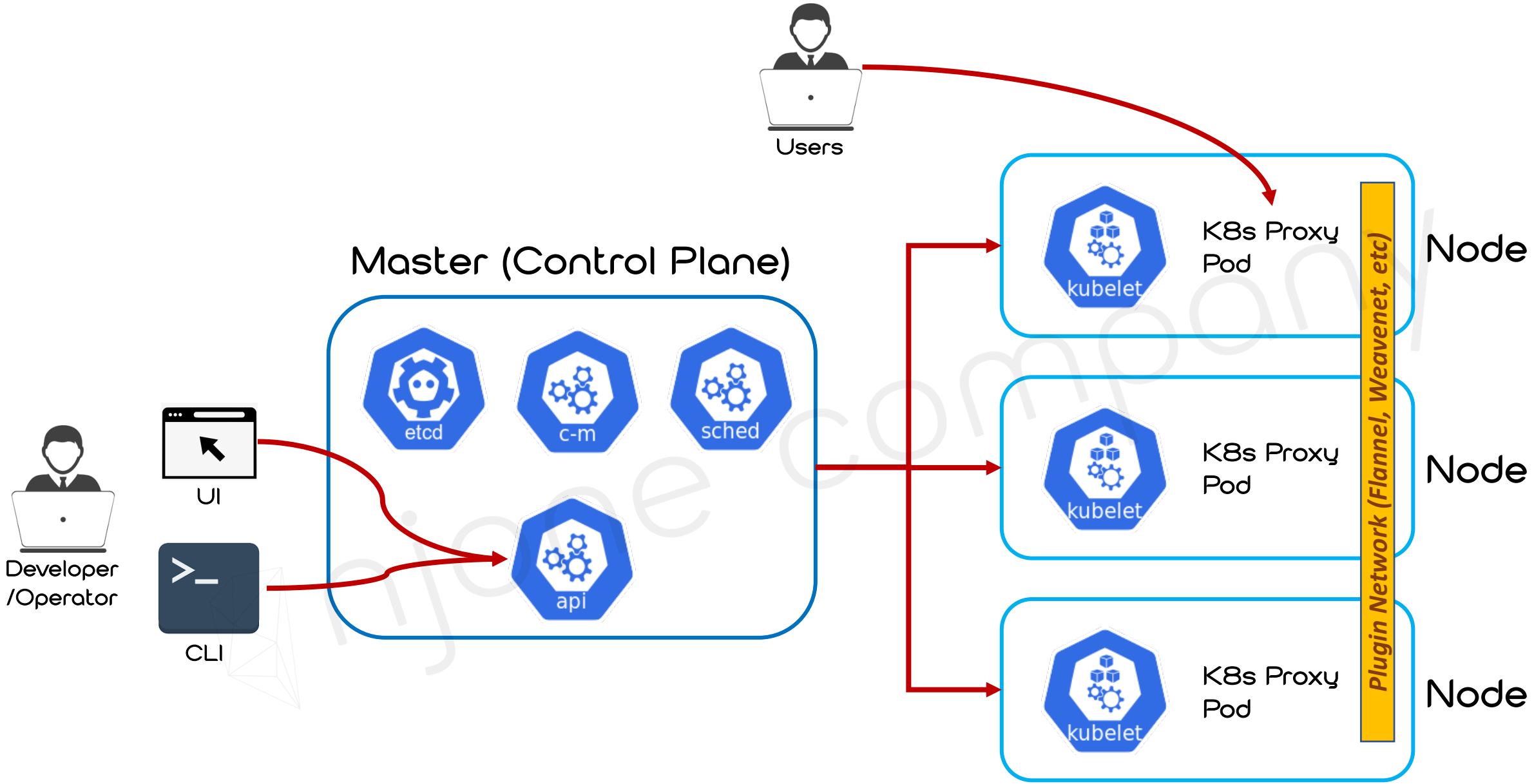


Kubernetes

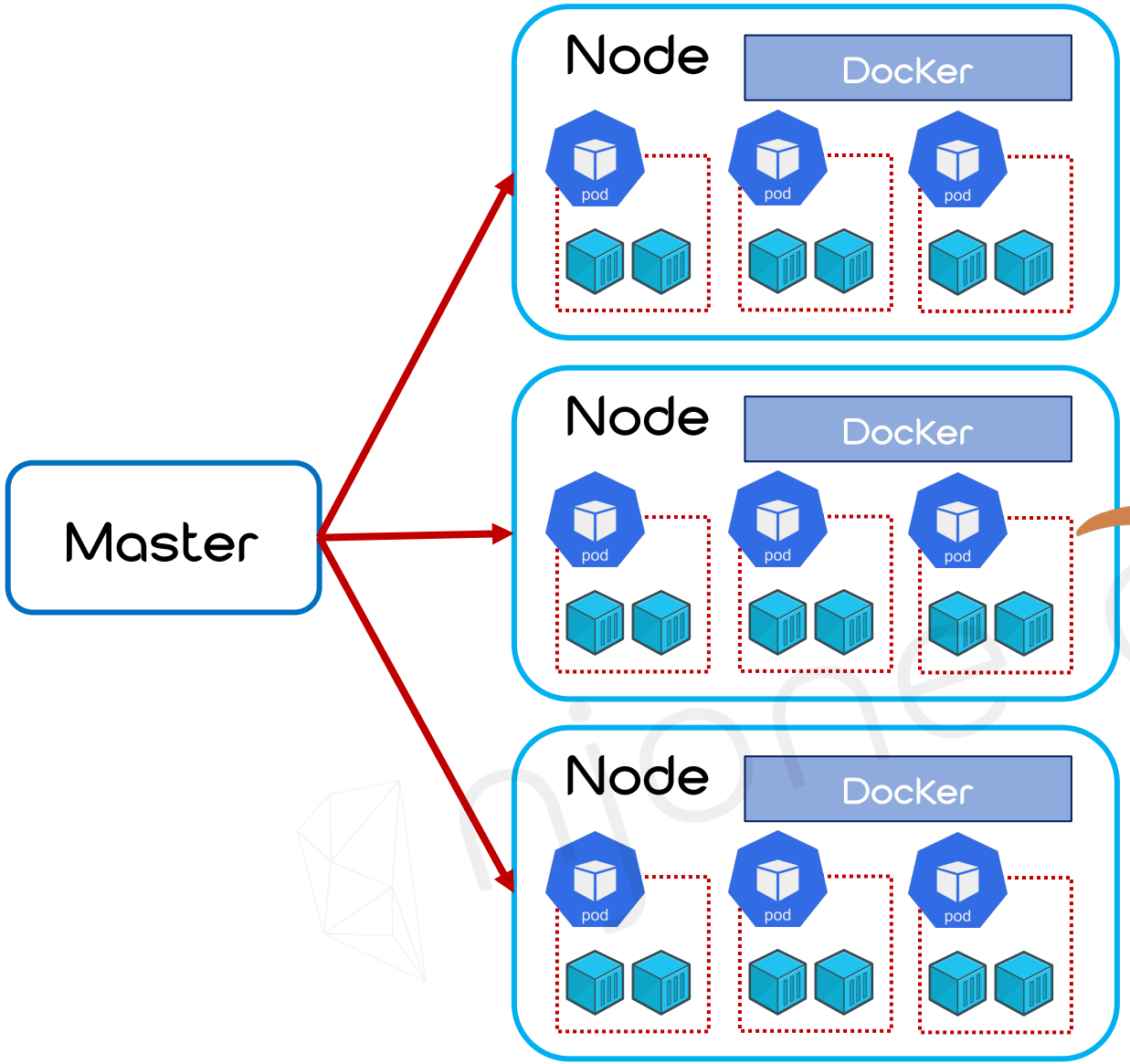
- 컨테이너화 된 애플리케이션 구동
- 서비스 디스커버리와 로드 밸런싱
- 스토리지 오케스트레이션
- 자동화된 롤아웃과 롤백
- 자동화된 빈 패킹(bin packing)
- 자동화된 복구(self-healing)
- 시크릿과 구성 관리

- 소스 코드 배포 X, 빌드 X
- 애플리케이션 레벨 서비스 X
- 로깅, 모니터링 솔루션 X
- 포괄적인 머신 설정, 유지보수, 관리, 자동 복구 시스템을 제공 X

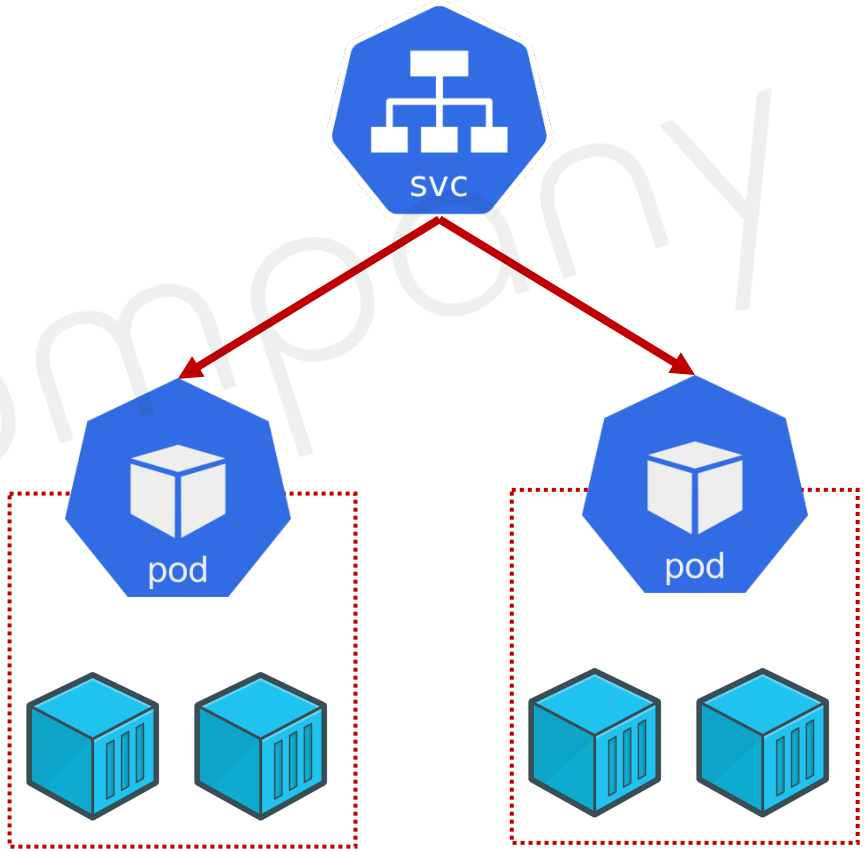
Kubernetes Cluster



Working of Kubernetes



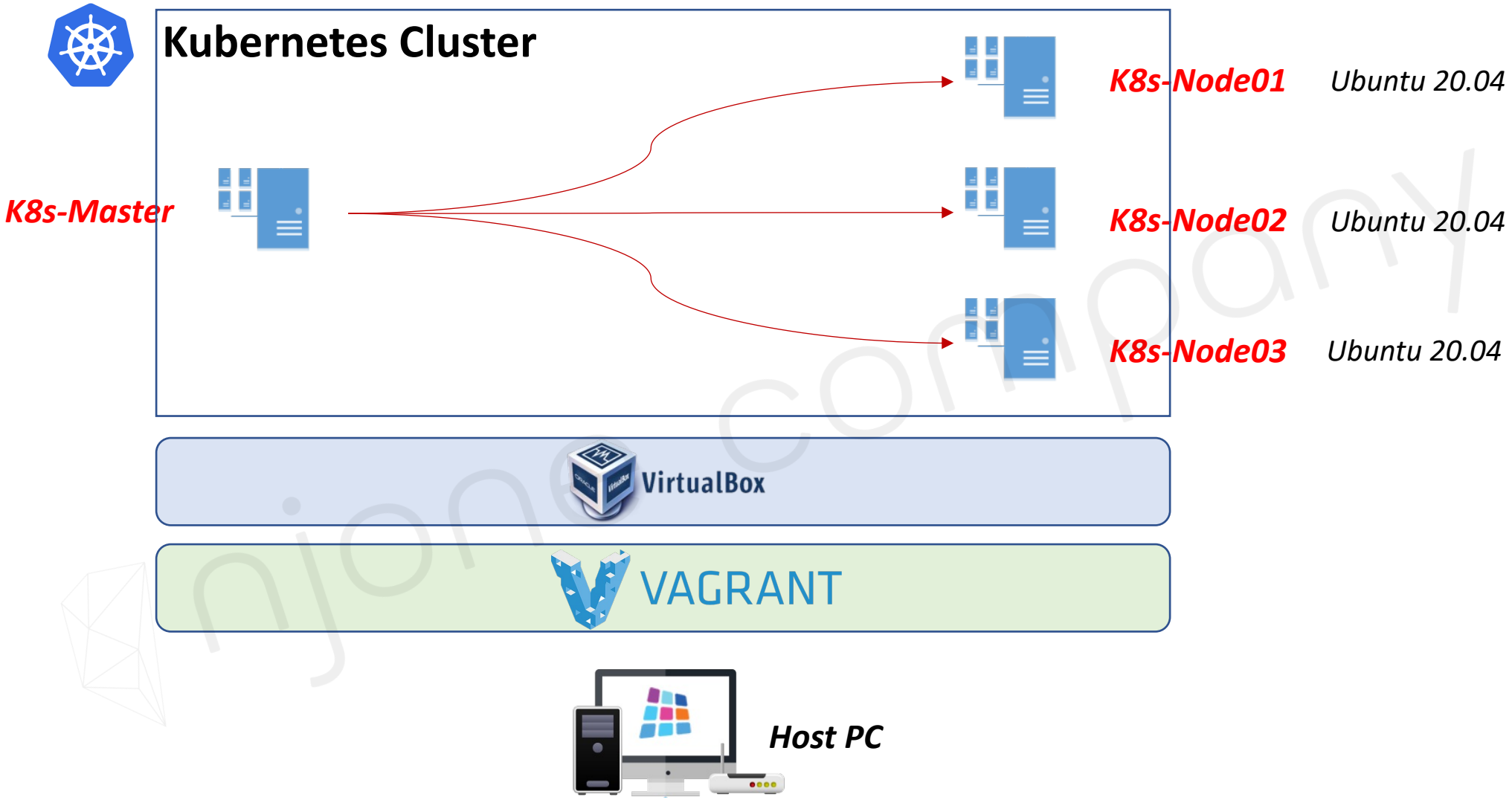
- Services: Replicated 된 PODS 그룹 간에 로드 밸런싱을 제공



- Pods: 애플리케이션을 위해 서로 상호 작용해야 하는 컨테이너들의 논리적인 집합

Install Kubernetes Cluster

- VM을 이용한 리눅스 설치



Install Kubernetes Cluster

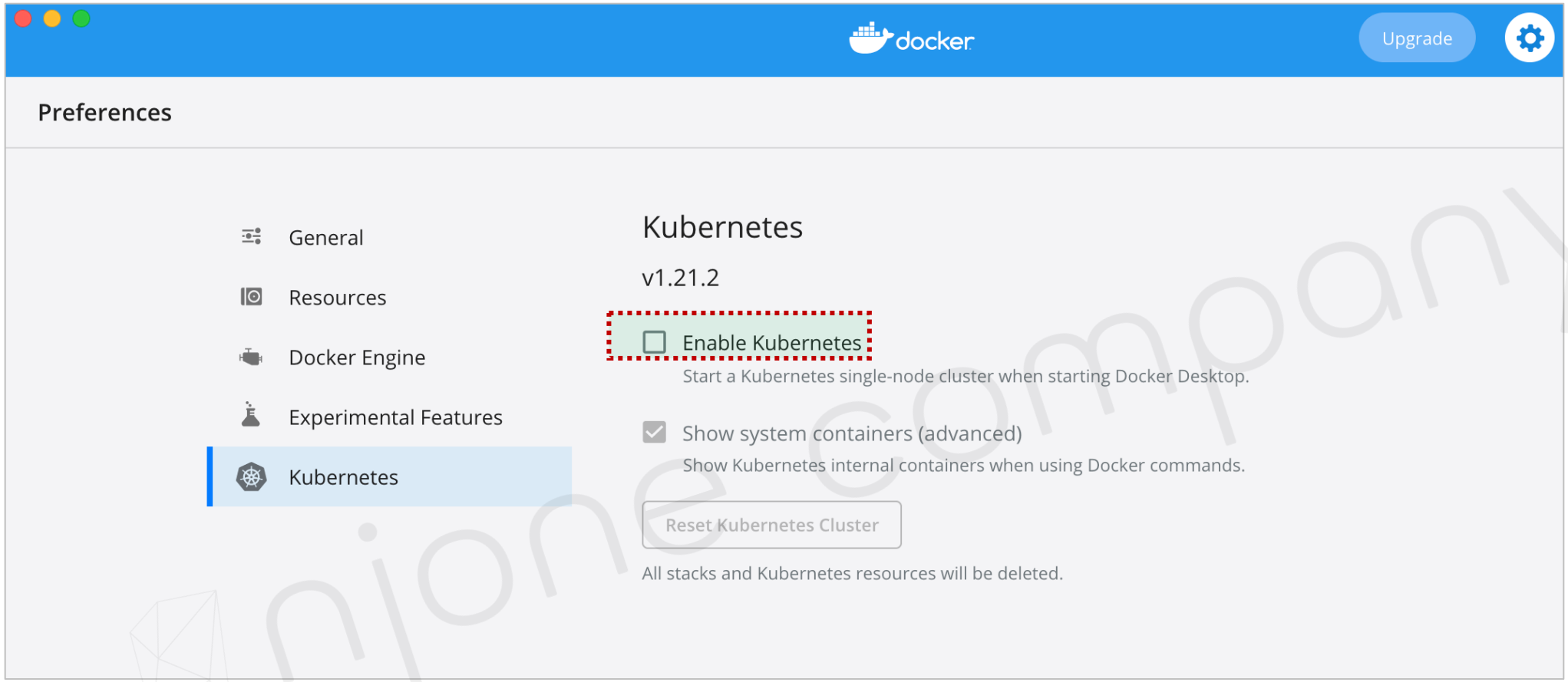
- https://github.com/joneconsulting/k8s/blob/master/install/kubernetes_install.md

```
vagrant status
Current machine states:

k8s-node01                running (virtualbox)
k8s-node02                running (virtualbox)
k8s-master                running (virtualbox)
```


```
[root@k8s-master ~]# kubectl get nodes
NAME         STATUS   ROLES                  AGE   VERSION
k8s-master   Ready    control-plane,master   24h   v1.20.5
k8s-node01   Ready    <none>                 24h   v1.20.5
k8s-node02   Ready    <none>                 24h   v1.20.5
```

Install MiniKube

- Exercise Kubernetes on ***Minikube***



Kubernetes – Test

- $ ***kubectl get nodes***

```
kubectl get nodes
NAME             STATUS   ROLES           AGE   VERSION
docker-desktop   Ready    control-plane   55d   v1.24.0
```

- $ ***kubectl get pods (or deployments, or services)***

```
kubectl get pods
NAME        READY   STATUS    RESTARTS      AGE
hello-pod   1/1     Running   1 (15m ago)   54d
(base)  edowon  ~
kubectl get deployments
No resources found in default namespace.
(base)  edowon  ~
kubectl get services
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
hello-svc    NodePort    10.100.207.206   <none>        8080:31145/TCP   54d
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP          55d
```

Kubernetes – Test

- 방법1)

  $ ***kubectl run sample-nginx --image=nginx --port=80***

- 방법2)

  $ ***kubectl create deployment sample-nginx --image=nginx***

  $ ***kubectl scale deployment sample-nginx --replicas=2***

- 방법3)

  $ ***kubectl apply -f sample1.yml***

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

Kubernetes – Test

- $ ***kubectl get pods***
- $ ***kubectl get pods -o wide***

```
❯ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           2m16s
(base)  dowonlee ❯ ~/Desktop/Work/14.online/devops/k8s
❯ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-66b6c48dd5-bfs4l   1/1     Running   0          2m19s
nginx-deployment-66b6c48dd5-qg8zx   1/1     Running   0          2m19s
(base)  dowonlee ❯ ~/Desktop/Work/14.online/devops/k8s
❯ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE    IP           NODE             NOMINATED NODE   READINESS GATES
nginx-deployment-66b6c48dd5-bfs4l   1/1     Running   0          3m56s  10.1.0.149   docker-desktop   <none>           <none>
nginx-deployment-66b6c48dd5-qg8zx   1/1     Running   0          3m56s  10.1.0.148   docker-desktop   <none>           <none>
```

- $ ***kubectl exec -it niginx-deployment-66b6c48dd5-qg8zx -- /bin/bash***
  - root@nginx-deployment-66b6c48dd5-qg8zx:/# ***apt-get update***
  - root@nginx-deployment-66b6c48dd5-qg8zx:/# ***apt-get install -y curl wget***
  - root@nginx-deployment-66b6c48dd5-qg8zx:/# ***hostname -i***
  - root@nginx-deployment-66b6c48dd5-qg8zx:/# ***curl -X GET http://10.10.0.148***

Kubernetes – Test

- Expose the deployment as service
  - $ ***kubectl expose deployment nginx-deployment --port=80 --type=NodePort***

```
kubectl expose deployment nginx-deployment --port=80 --type=NodePort
service/nginx-deployment exposed
(base)  dowonlee  ~/Desktop/Work/14.online/devops/k8s
kubectl get services
NAME               TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes         ClusterIP   10.96.0.1      <none>        443/TCP        43d
nginx-deployment   NodePort    10.111.25.43   <none>        80:31436/TCP   4s
```

Kubernetes – Test

- Display the pods
  - $ ***kubectl get pods***

```
kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-66b6c48dd5-bfs4l   1/1     Running   0          23m
nginx-deployment-66b6c48dd5-qg8zx   1/1     Running   0          23m
```

- Delete the pods
  - $ ***kubectl delete pod niginx-deployment-66b6c48dd5-qg8zx***

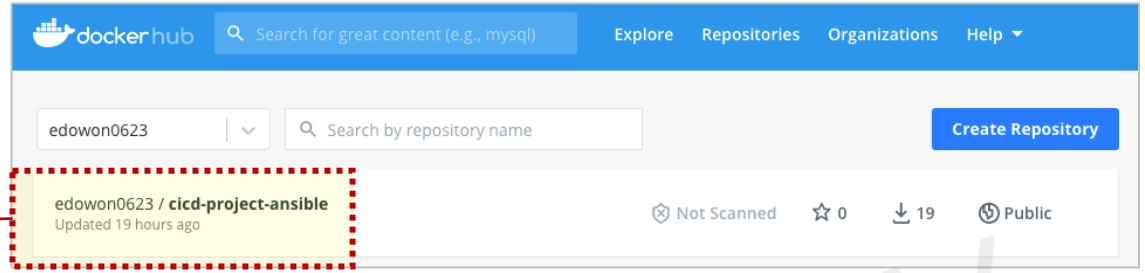
```
kubectl delete pod nginx-deployment-66b6c48dd5-qg8zx
pod "nginx-deployment-66b6c48dd5-qg8zx" deleted
(base)  dowonlee  ~/Desktop/Work/14.online/devops/k8s
kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-66b6c48dd5-bfs4l   1/1     Running   0          23m
nginx-deployment-66b6c48dd5-pn96x   1/1     Running   0          4s
```

- Delete the deployment
  - $ ***kubectl delete deployment niginx-deployment***

Kubernetes – Test

- Create deployment and service file

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: cicd-deployment
spec:
  selector:
    matchLabels:
      app: cicd-devops-project
  replicas: 2

  template:
    metadata:
      labels:
        app: cicd-devops-project
    spec:
      containers:
      - name: cicd-devops-project
        image: edowon0623/cicd-project-ansible
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
```




```yaml
apiVersion: v1
kind: Service
metadata:
  name: cicd-service
  labels:
    app: cicd-devops-project
spec:
  selector:
    app: cicd-devops-project
  type: NodePort
  ports:
    - port: 8080
      targetPort: 8080
      nodePort: 32000
```

Kubernetes – Test

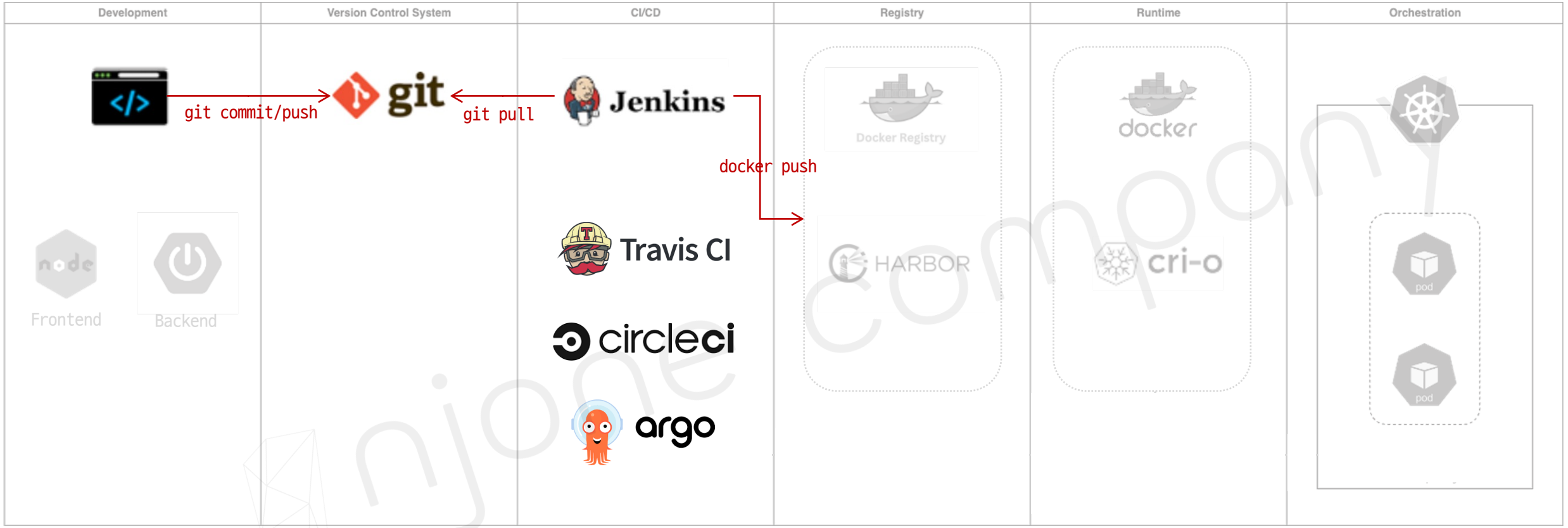
- Create deployment and services
  - $ ***kubectl apply –f cicd-devops-deployment.yml***
  - $ ***kubectl apply –f cicd-devops-service.yml***

```
kubectl apply -f cicd-devops-deployment.yml
deployment.apps/cicd-deployment created
(base)  dowonlee  ~/Desktop/Work/14.online/devops/k8s
kubectl get deployments
NAME              READY   UP-TO-DATE   AVAILABLE   AGE
cicd-deployment   1/2     2            1           5s
```

```
kubectl apply -f cicd-devops-service.yml
service/cicd-service created
(base)  dowonlee  ~/Desktop/Work/14.online/devops/k8s
kubectl get services
NAME            TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
cicd-service    NodePort    10.111.0.95    <none>        8080:32000/TCP   3s
kubernetes      ClusterIP   10.96.0.1      <none>        443/TCP          43d
```

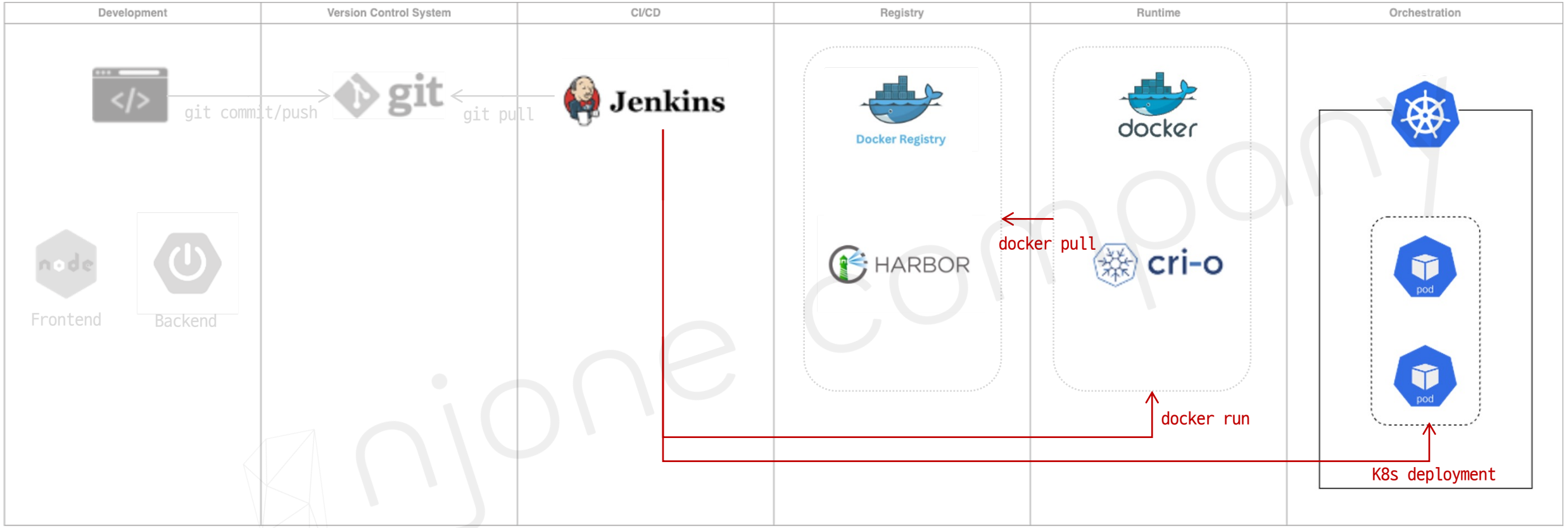
CI/CD 자동화 빌드 시스템 구축

CI/CD



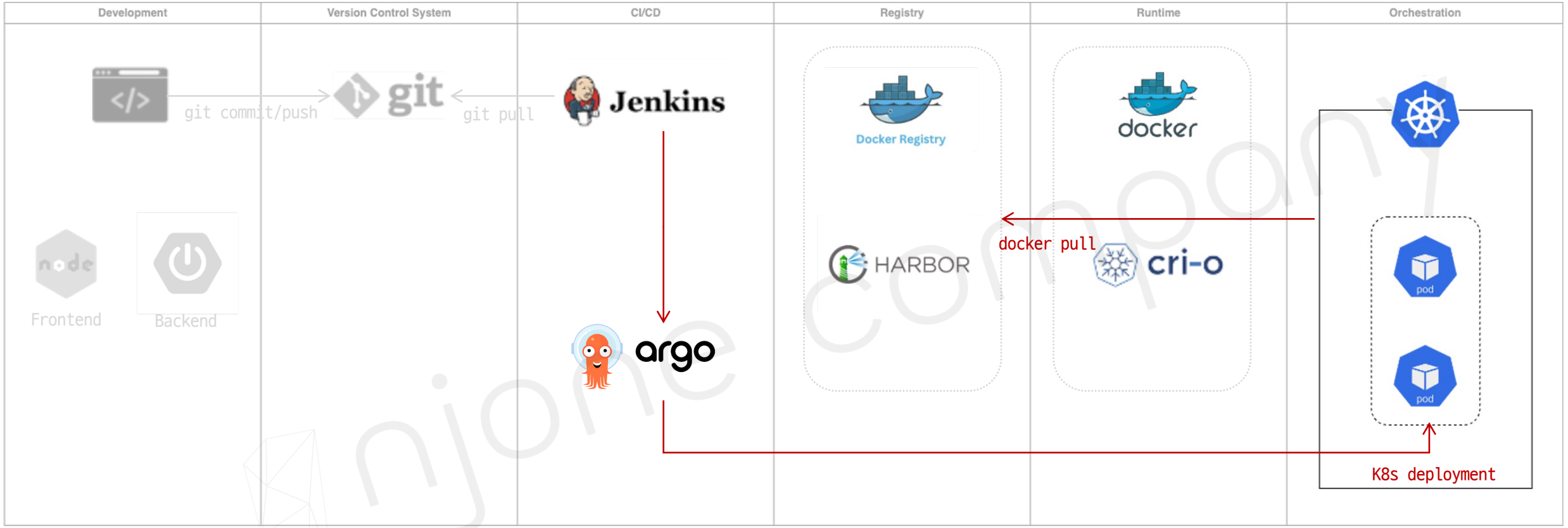
CI/CD 자동화 빌드 시스템 구축

CI/CD

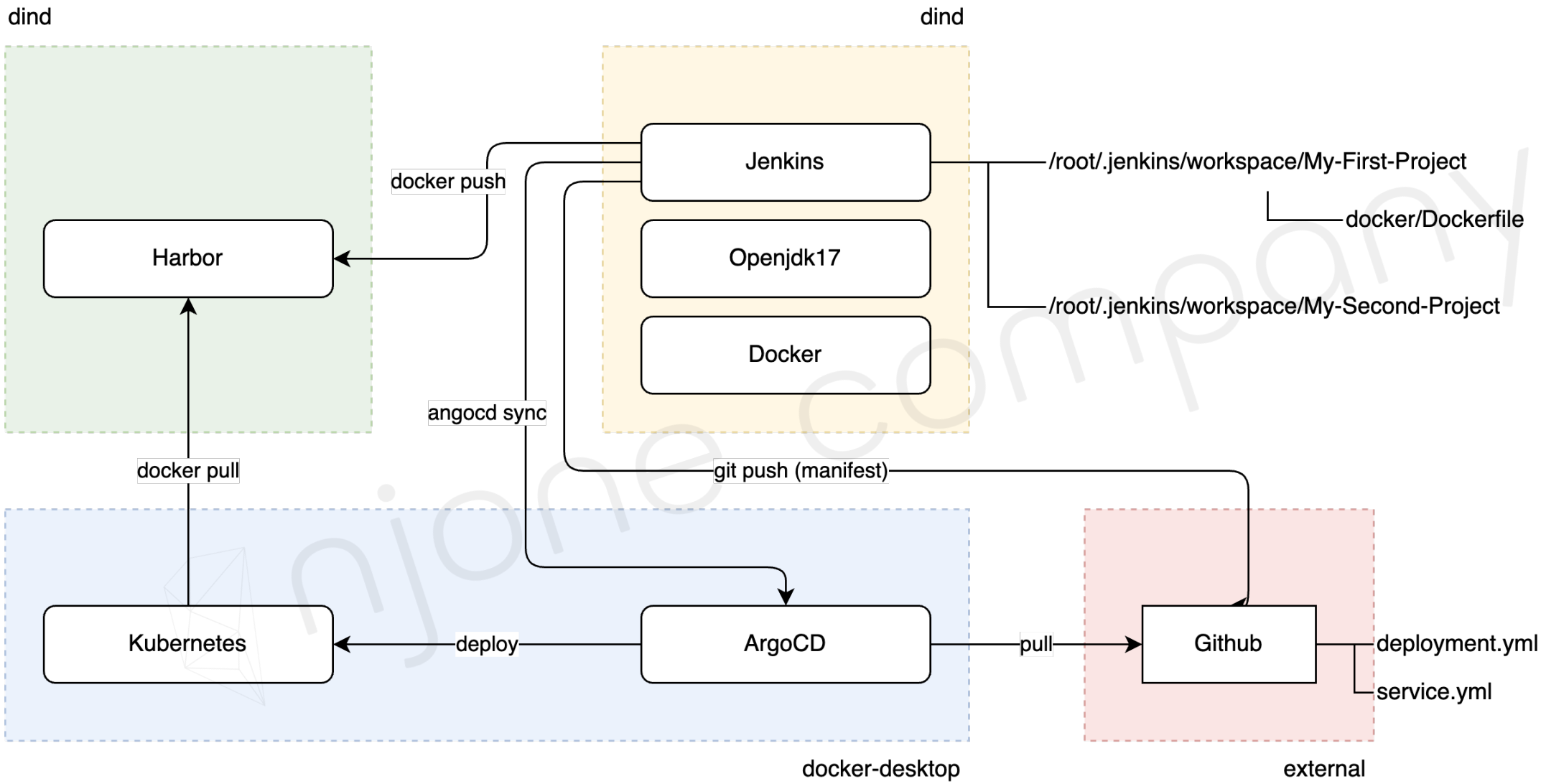


CI/CD 자동화 빌드 시스템 구축

CI/CD



CI/CD 동작의 흐름


dind
dind
Jenkins
Openjdk17
Docker
Harbor
docker push
/root/.jenkins/workspace/My-First-Project
docker/Dockerfile
/root/.jenkins/workspace/My-Second-Project
angocd sync
docker pull
git push (manifest)
Kubernetes
deploy
ArgoCD
pull
Github
deployment.yml
service.yml
docker-desktop
external

K8s + ArgoCD 연동

- K8s
  - Docker desktop에 포함된 K8s 사용
- ArgoCD 설치
  - kubectl create namaespace argocd
  - kubectl apply -y argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/v2.3.3/manifests/install.yaml
  - kubectl get pod -n argocd
  - kubectl get service -n argocd
  - kubectl edit svc argocd-server -n argocd → NodePort 변경

K8s + ArgoCD 연동

- ArgoCD 설치
  - argocd cli 설치 (Windows 기준, Powershell 사용)

```
$version = (Invoke-RestMethod https://api.github.com/repos/argoproj/argo-cd/releases/latest).tag_name
$url = "https://github.com/argoproj/argo-cd/releases/download/" + $version + "/argocd-windows-amd64.exe"
$output = "argocd.exe"
Invoke-WebRequest -Uri $url -OutFile $output
```

  - CMD에서 argocd 설치 확인
  - admin 로그인 암호 확인 (방법1)

```
$ argocd admin initial-password -n argocd
```

  - admin 로그인 암호 확인 (방법2)

```
$ kubectl get secret argocd-initial-admin-secret –n argocd –o jsonpath=“{.data.password}” → encode.b64로 저장
$ certutil –decode encode.b64 decode.txt
```

K8s + ArgoCD 연동

- ArgoCD Login
  - argocd login –insecure ***192.168.0.41:30764*** ← NodePort
- ArgoCD Token 생성
  - argocd account generate-token --account admin

Error) account 'admin' does not have apiKey capability 발생 시
- $ kubectl edit cm argocd-cm -n argocd → 아래 항목 추가

```
data:
    accounts.admin: apiKey
```

  - curl 명령어로 applications 확인

```
$ curl -k -L -H "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhcmdvY2QiLCJ..." \
        https://192.168.0.41:30674/api/v1/applications
```

K8s + ArgoCD 연동

- ArgoCD New Project
  - Project: default
  - Cluster: in-cluster (https://Kubernetes.default.svc)
  - Namespace: default
  - Repo URL: https://github.com/joneconsulting/argocd-demo
  - Target revision: main
  - Path: base
  - Retry options: Retry disabled

- Sample → https://github.com/joneconsulting/argocd-demo
  - base/deployment.yml
  - base/service.yml