

초격차 패키지 Online.

안녕하세요 백엔드 개발자를 위한 DevOps 입문 강의를 진행할 이성미, 양재현입니다.

- Chap1** | 백엔드 아키텍처 맛보기 (멀티모듈, MSA, DDD)
- Chap2** | 백엔드 개발자를 위한 클라우드 인프라 운영 - Amazon
- Chap3** | 백엔드 개발자를 위한 컨테이너 서비스 - Docker
- Chap4** | 클라우드 기반 컨테이너 서비스 운영 - Amazon ECR, ECS, Fargate
- Chap5** | 백엔드 개발자를 위한 빌드 자동화 - Jenkins
- Chap6** | MVP 프로젝트: 채용사이트 구축하기

백엔드 개발자를 위한 빌드 자동화- Jenkins

- 1 Git을 이용한 소스코드 버전관리
- 2 Git Branch를 통한 협업 개발하기
- 3 Jenkins 프로그램 소개 및 설치하기
- 4 Git 연동을 위한 Jenkins 환경설정
- 5 Jenkins 파이프라인을 이용한 컨테이너 빌드
- 6 Jenkins Slack Webhook 연동하기
- 7 실전 빌드하기

source push → container build → ecr에 업로드

8 LAB: CI 구축하기

git 코드 업로드 부터 컨테이너 빌드까지

백엔드 개발자를 위한 빌드 자동화- Jenkins

1 Git을 이용한 소스코드 버전관리

Git을 이용한 소스코드 버전관리

1.

Git을 이용한
소스코드 버전관리

학습내용

Git과 GitHub

Git 설치

Git 기본 명령어

Git 브랜치로 버전관리

Git과 GitHub

1.

Git을 이용한
소스코드 버전관리



• Git?

- 2005년 리누스 토발즈가 리눅스 커널 프로젝트 개발을 위해 만든 **분산 버전관리** 툴
- 코드의 원본이나 변경 내역을 저장하는 **형상관리(Configuration Management)** 툴
- 버전관리

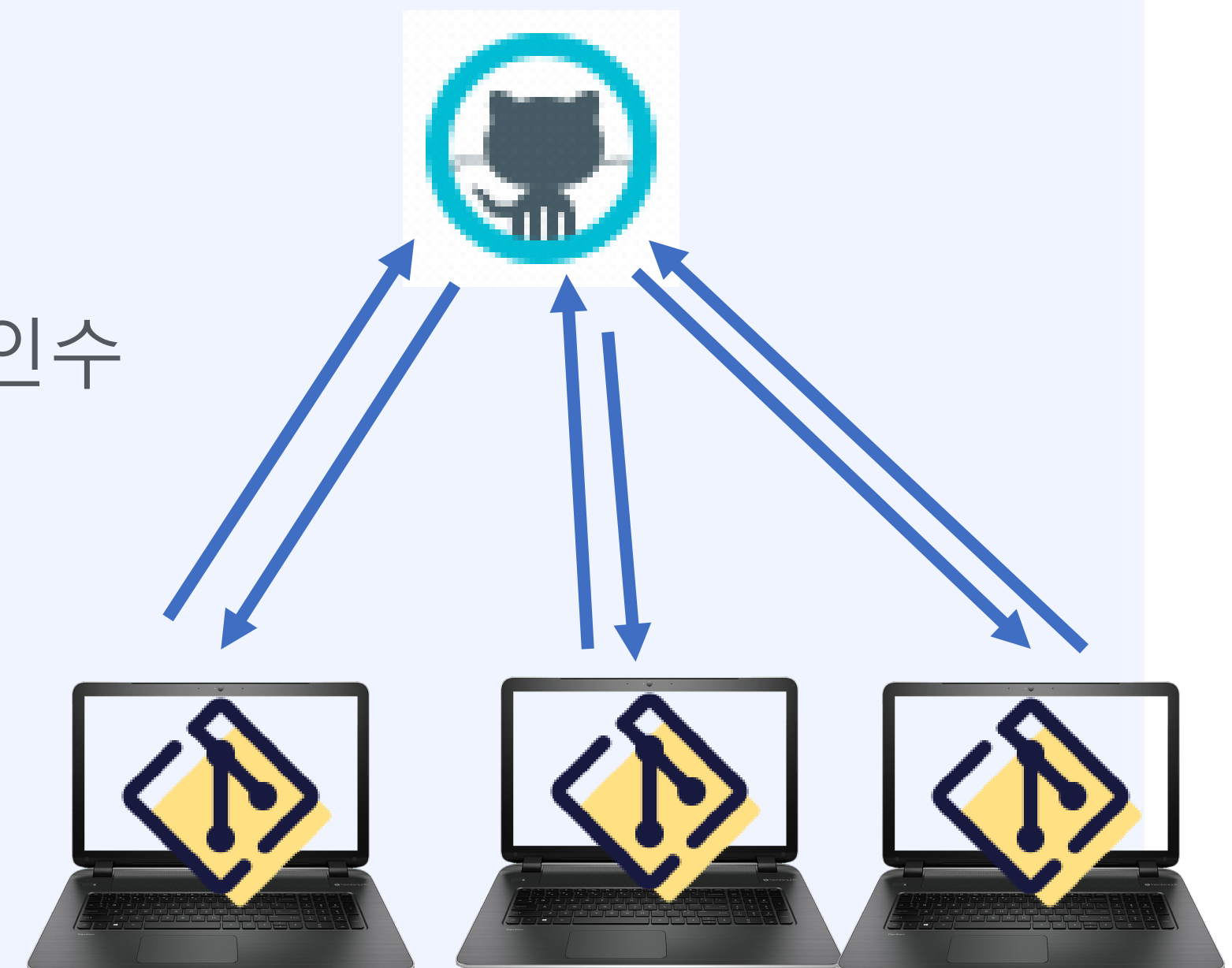
소프트웨어 개발과정에서 코드의 추가 변경과정을 모두 기록하여
특정 시점으로 돌아가거나 문제가 생긴 파일을 복원하는 과정

• GitHub

- **Git** 을 관리해주는 **웹 호스팅 서비스**로 2008년 설립. 2018년도에 MS 인수
- 분산 버전 제어, 액세스 제어, 소스 코드 관리,
버그 추적, 기능 요청 및 작업 관리를 제공
- 개발자들 사이의 거대한 커뮤니티를 형성

• 참고 도서

- [Pro Git\(https://git-scm.com/book/ko/v2\)](https://git-scm.com/book/ko/v2)
- [알잘딱깔센GitHub](#)



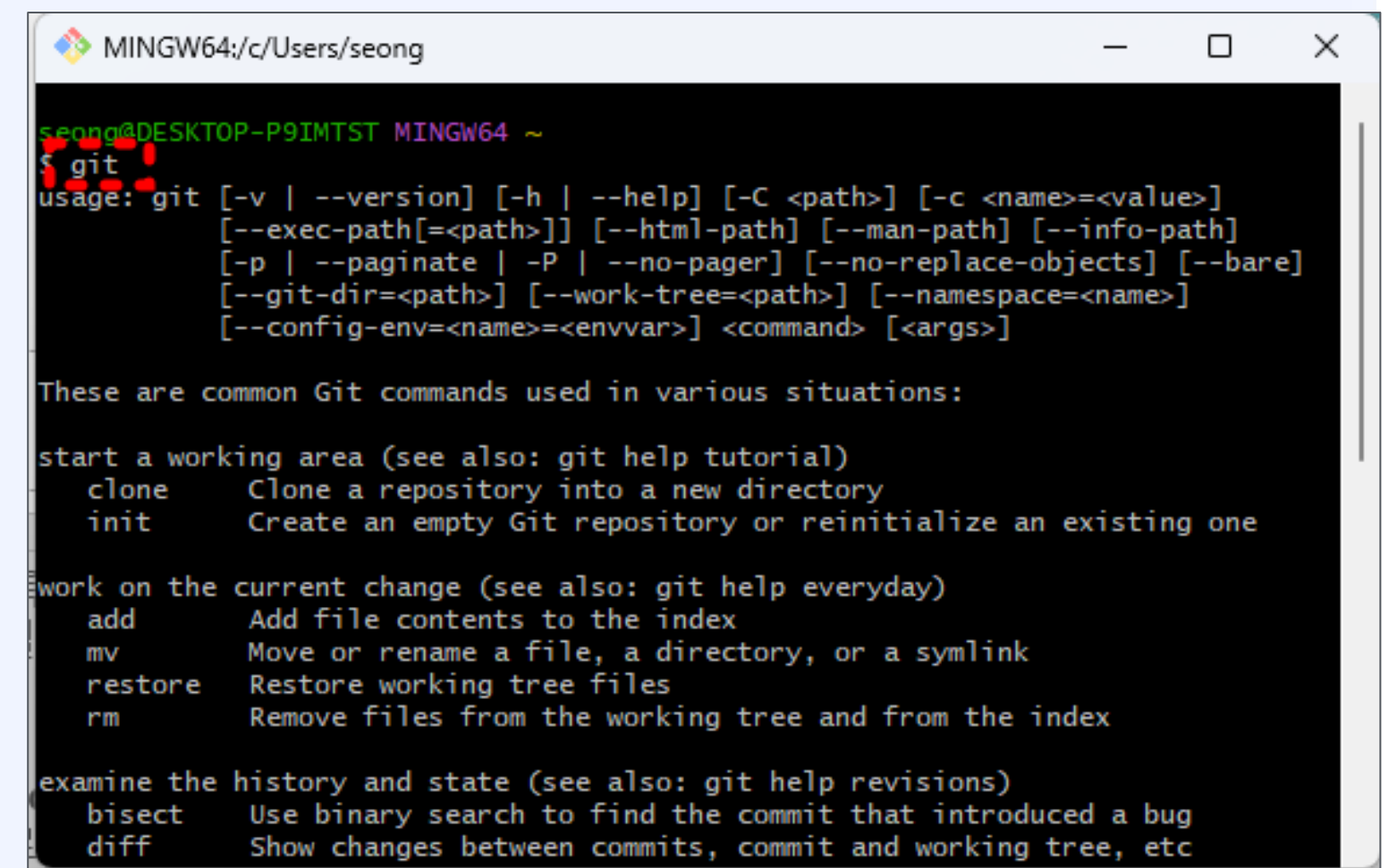
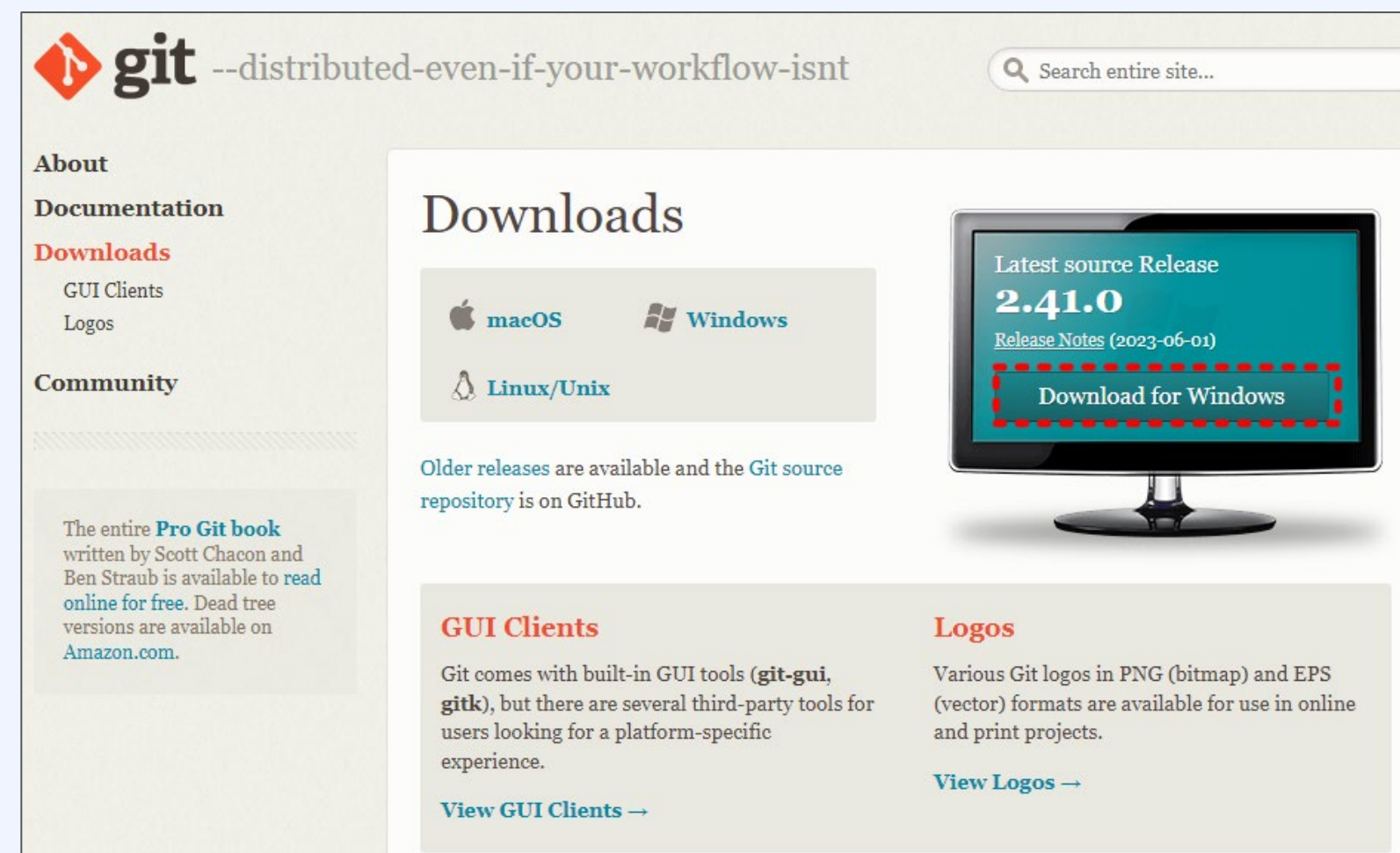
Git 설치

1.

Git을 이용한
소스코드 버전관리

Git 설치

- Windows
 - <https://git-scm.com/downloads> -> Download for Windows -> latest 64bit
- Mac
 - 기본으로 설치되어 있음
- Linux
 - Ubuntu
 - `apt update`
 - `apt install -y git`
 - Redhat
 - `yum install -y git`



Git 기본 구성

1.

Git을 이용한
소스코드 버전관리

Git 사용자 정보 설정

- git 은 누가 커밋했는지 확인하기 위해 사용자정보(email 주소와 이름)을 설정
- 설정파일의 위치
 - git 기본 환경 구성(/etc/gitconfig) ③
 - 유저별 설정 : HOME_DIR/.gitconfig ②
 - 작업 디렉토리별 구성 : /소스코드디렉토리/.git/config ①

- 유저별 설정

```
git config --global user.name "237summit"
```

```
git config --global user.email "seongmi.lee@gmail.com"
```

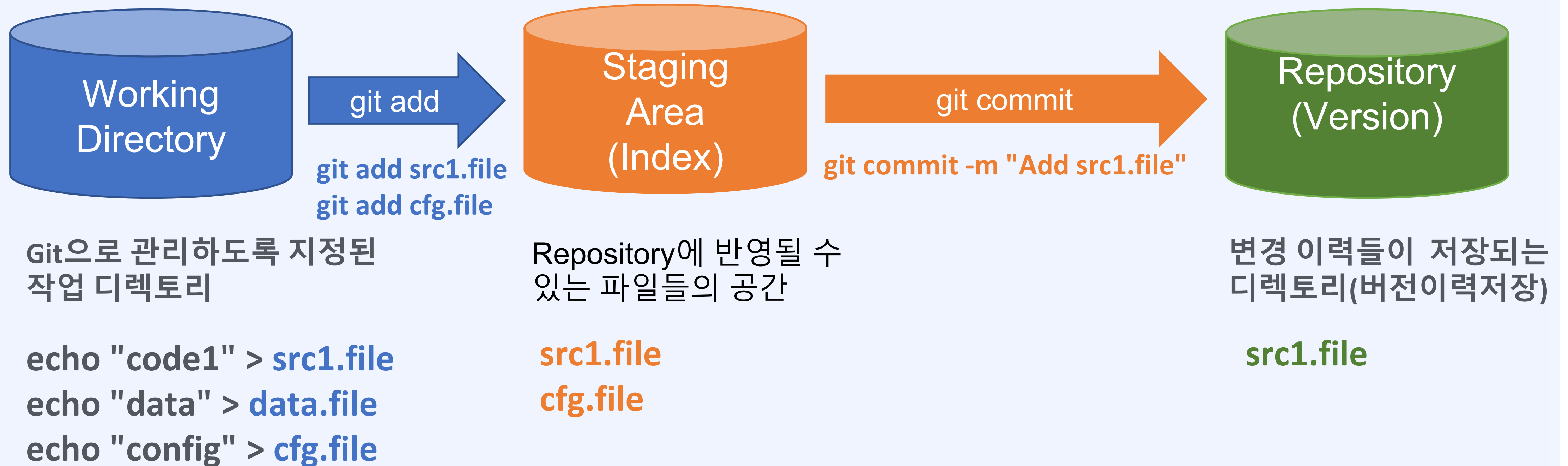
```
cat ~/.gitconfig
```


Git Workflow

1.

Git을 이용한
소스코드 버전관리

Git의 3가지 작업공간



Git 사용하기(1)

1.

Git을 이용한
소스코드 버전관리

Git 기본 명령어

- 레포지토리 초기화: **git init**
- Git Repository나 Staging Area에 추가되지 않아야 하는 파일 정의: **.gitignore**
- Working directory에 있는 파일을 Staging Area로 옮기는 명령 : **git add**
- Git repository의 브랜치, 추적중인 파일과 변경된 파일 등의 현재 상태 확인: **git status**
- Staging Area에 있는 파일을 Repository 에 저장(변경 내용 기록) : **git commit**
- Repository의 commit된 로그 확인: **git log**
- 워킹 디렉토리, 스테이지 영역, 리포지토리 사이의 변경사항 확인 : **git diff**



Git 사용하기(2)

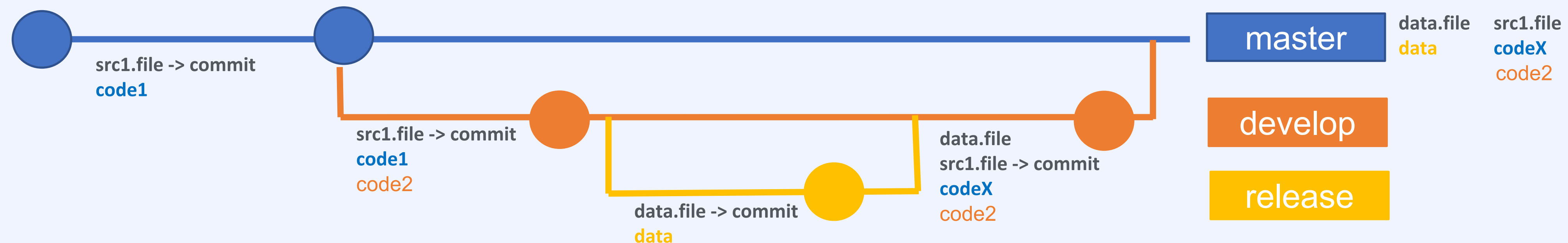
1.

Git을 이용한
소스코드 버전관리

Git 브랜치 관리 명령어

브랜치(Branch)란 나뉘어가지(branch)의 의미로, 소프트웨어 개발시 원래 코드와 상관없이 독립적으로 개발해야할 경우 발생하는데 이때 **브랜치를 사용하여 가지를 만들어 독립적으로 개발하거나 병합**

- 브랜치 리스트 확인(현재 사용중인 브랜치 : ***master**) : **git branch**
- 브랜치 생성 : **git branch branch_name**
- 브랜치 삭제 : **git branch -d branch_name**
- 브랜치 이름 변경: **git branch -m branch_name**
- 브랜치 바꾸기: **git checkout 브랜치이름**
- 브랜치들의 커밋 상태 보기: **git log --oneline --decorate --graph --all**
- 브랜치 병합: **git merge other_branch**



우리가 배운 내용

- git 설치후 git-bash 실행
- git 기본 명령어 사용
- git 브랜치 관리 명령어 사용
- 개발 환경에서의 브랜치

백엔드 개발자를 위한 빌드 자동화- Jenkins

2 Git Branch를 통한 협업 개발하기

Git을 이용한 소스코드 버전관리

2.

Git Branch를 통한
협업 개발하기

학습내용

Git 브랜치 Model

GitHub 가입

GitHub Repository 관리

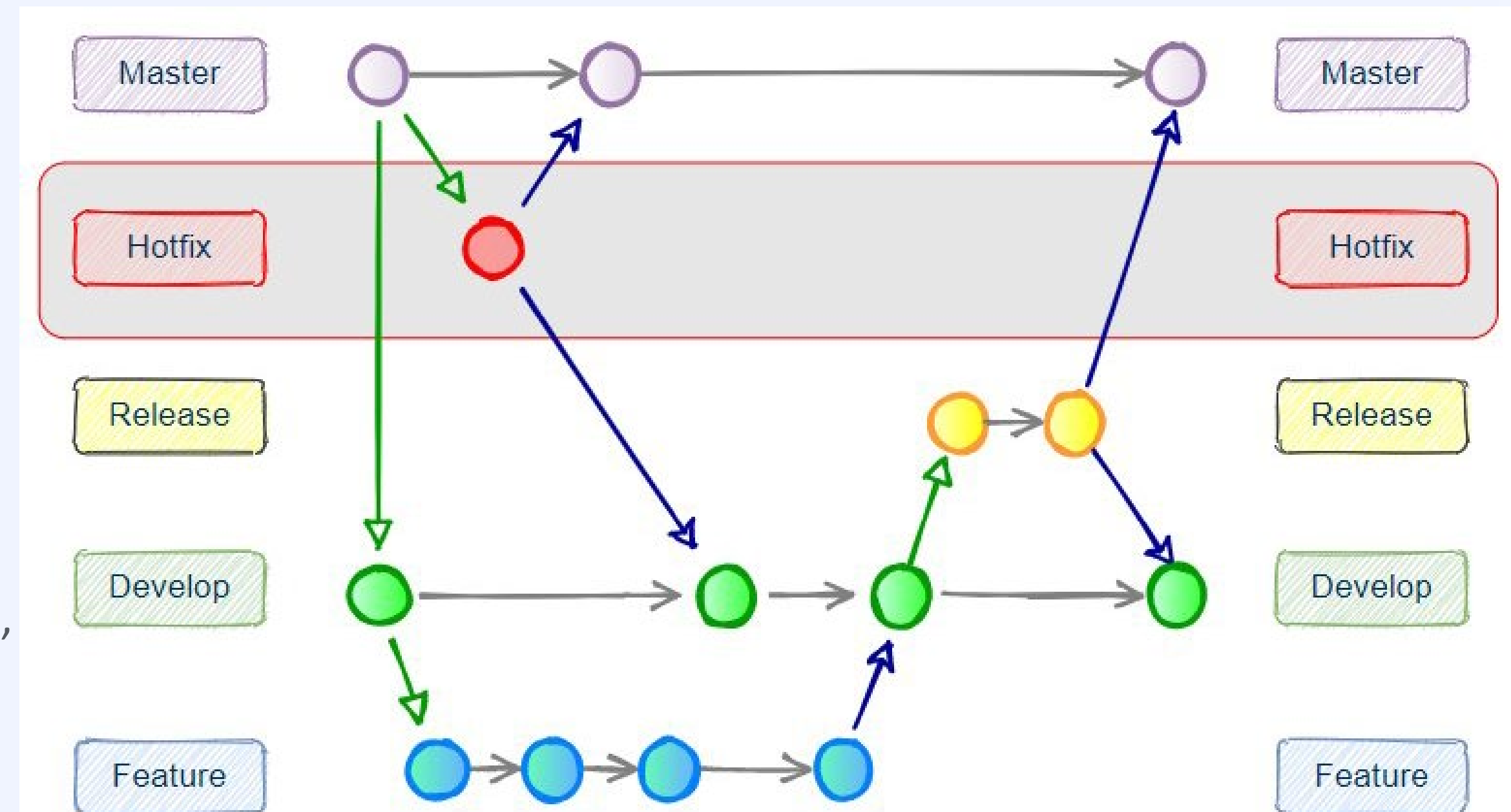
Git 브랜치 모델

2.

Git Branch를 통한
협업 개발하기

Git 브랜치 Model

- 운용 모델에서는 크게 나눠 4가지 종류의 브랜치를 이용하여 개발을 진행
 - **메인 브랜치(Main branch)**
 - **master** : 배포 가능한 상태만을 관리.
 - **develop** : 통합 브랜치의 역할을 하며, 평소에는 이 브랜치에서 개발을 진행
 - **피쳐 브랜치(Feature branch)**
 - 새로운 기능 개발 및 버그 수정이 필요할 때에 'develop' 브랜치로부터 분기
 - **릴리스 브랜치(Release branch)**
 - 버그를 수정하거나 새로운 기능을 포함한 상태로 모든 기능이 정상적으로 동작하는지 확인
 - **핫픽스 브랜치(Hotfix branch)**
 - 배포한 버전에 긴급하게 수정을 해야 할 필요가 있을 경우, 'master' 브랜치에서 분기하는 브랜치



<https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/GitFlow-Hotfix-Branch-Example-Start-Finish>

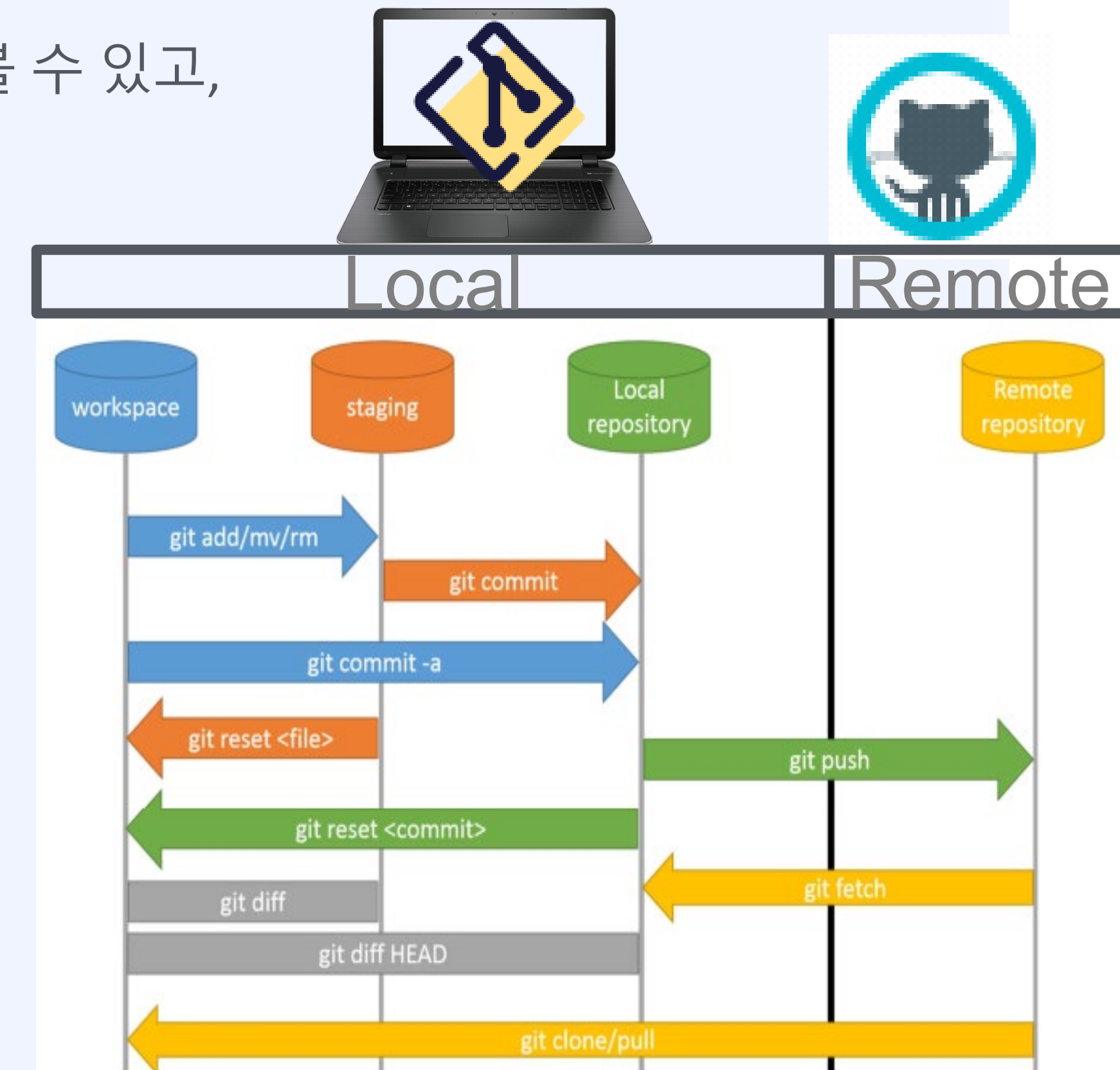
Git과 GitHub

2.

Git Branch를 통한
협업 개발하기

GitHub

- Git 저장소를 관리해주는 웹 호스팅 서비스
- 한 프로젝트에 여러 명의 사람이 참여하여 버전 제어 및 공동 작업이 가능
- 자신의 프로젝트는 물론 다른 개발자들의 인기 있는 프로젝트의 코드를 볼 수 있고, 이슈(issue)를 제기할 수 있으며 원한다면 다른 이의 프로젝트를 수정하고 발전시키는 데에 참여
- 개발자들 사이의 거대한 커뮤니티를 형성
- 여러 명의 사람이 참여하여 버전 제어 및 공동 작업이 가능



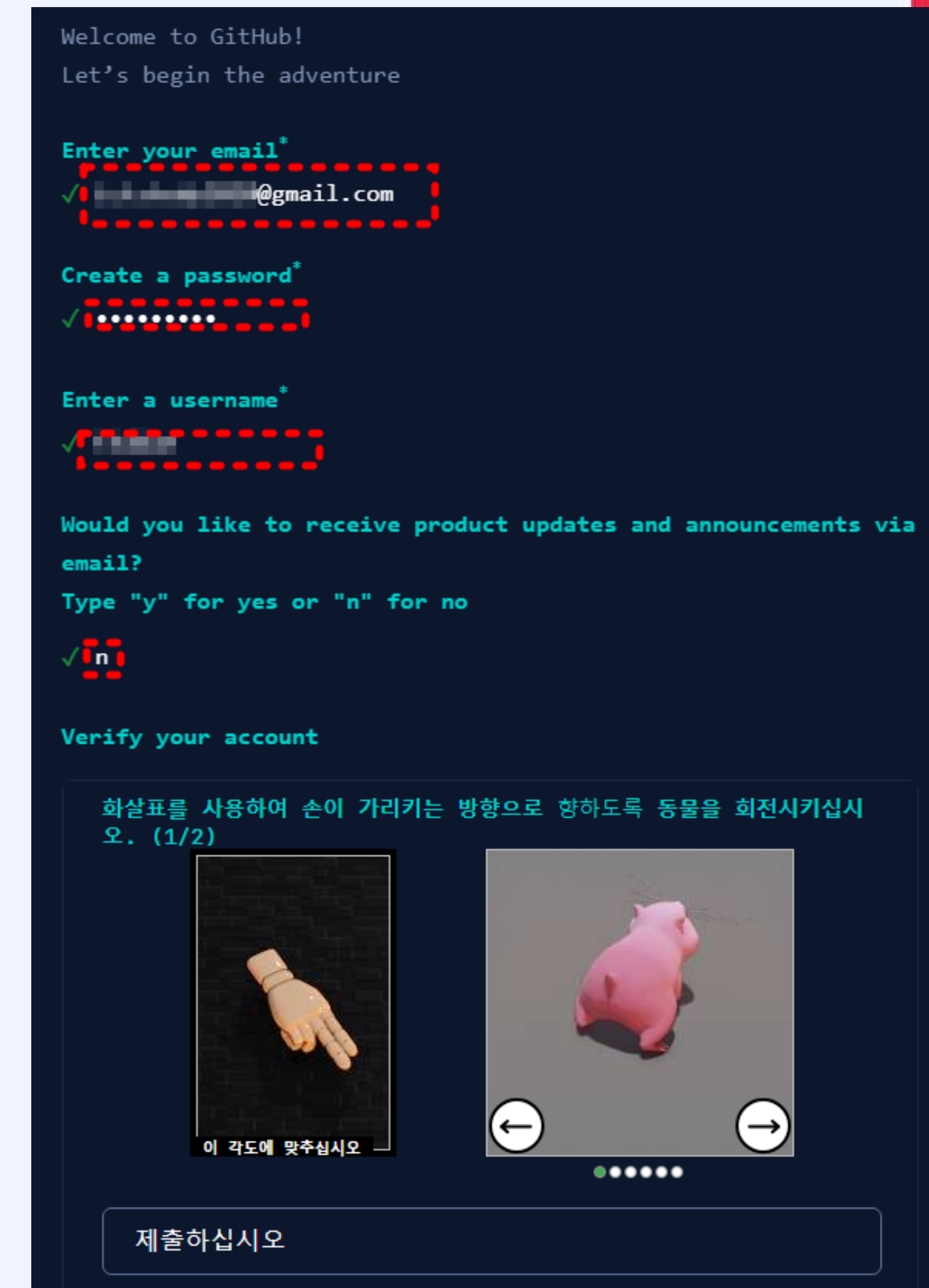
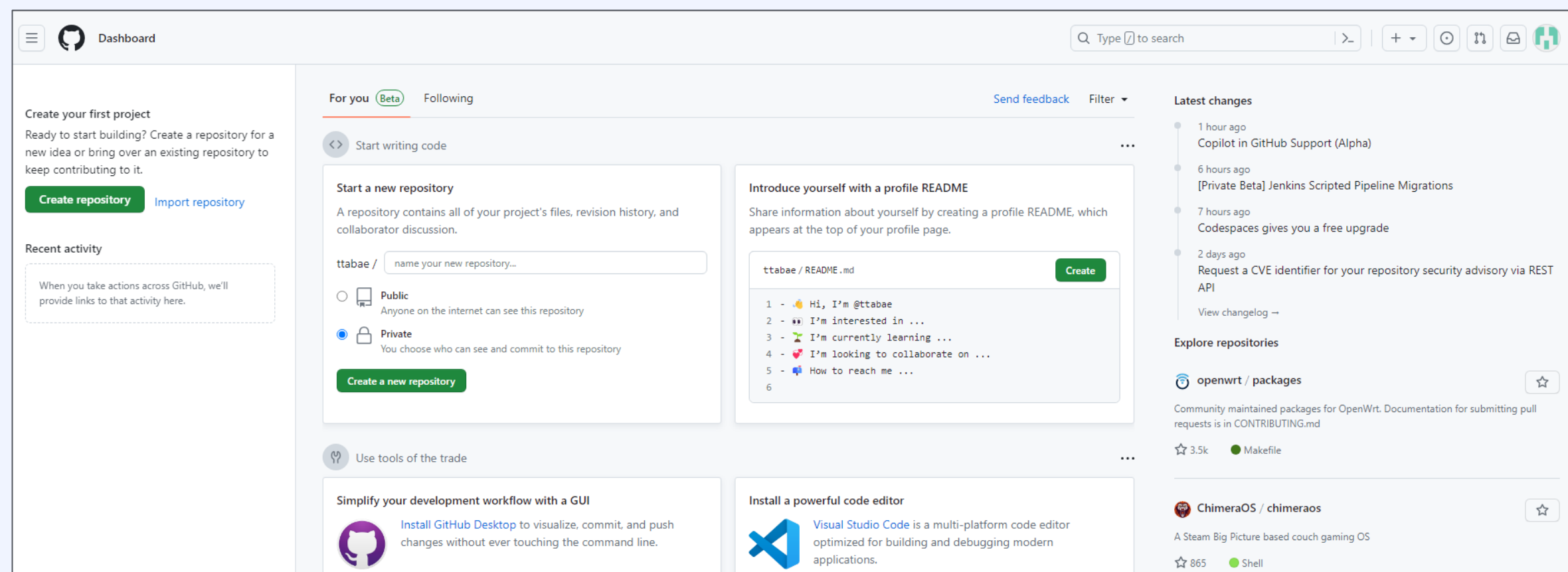
GitHub 계정만들기

2.

Git Branch를 통한
협업 개발하기

GitHub 계정 만들기(github.com)

- github.com -> Sign up ->
- email 주소, 패스워드 입력 후 [Continue]
- 계정(username)이름(fastcampusdeveloper) 입력 후 [Continue]
- 제품 업데이트 정보 메일통해 받을지? <n>
- 계정 검증 후[제출하십시오] 클릭(2회)
- [Create Account] 버튼 클릭 -> Email 로 전달된 코드 입력
- 설문조사 - 프로젝트 도구 추천해줌
- 이용금액 안내 [Continue for free]
- Git Hub Dashboard 화면이 표시되고 Repository 생성가능



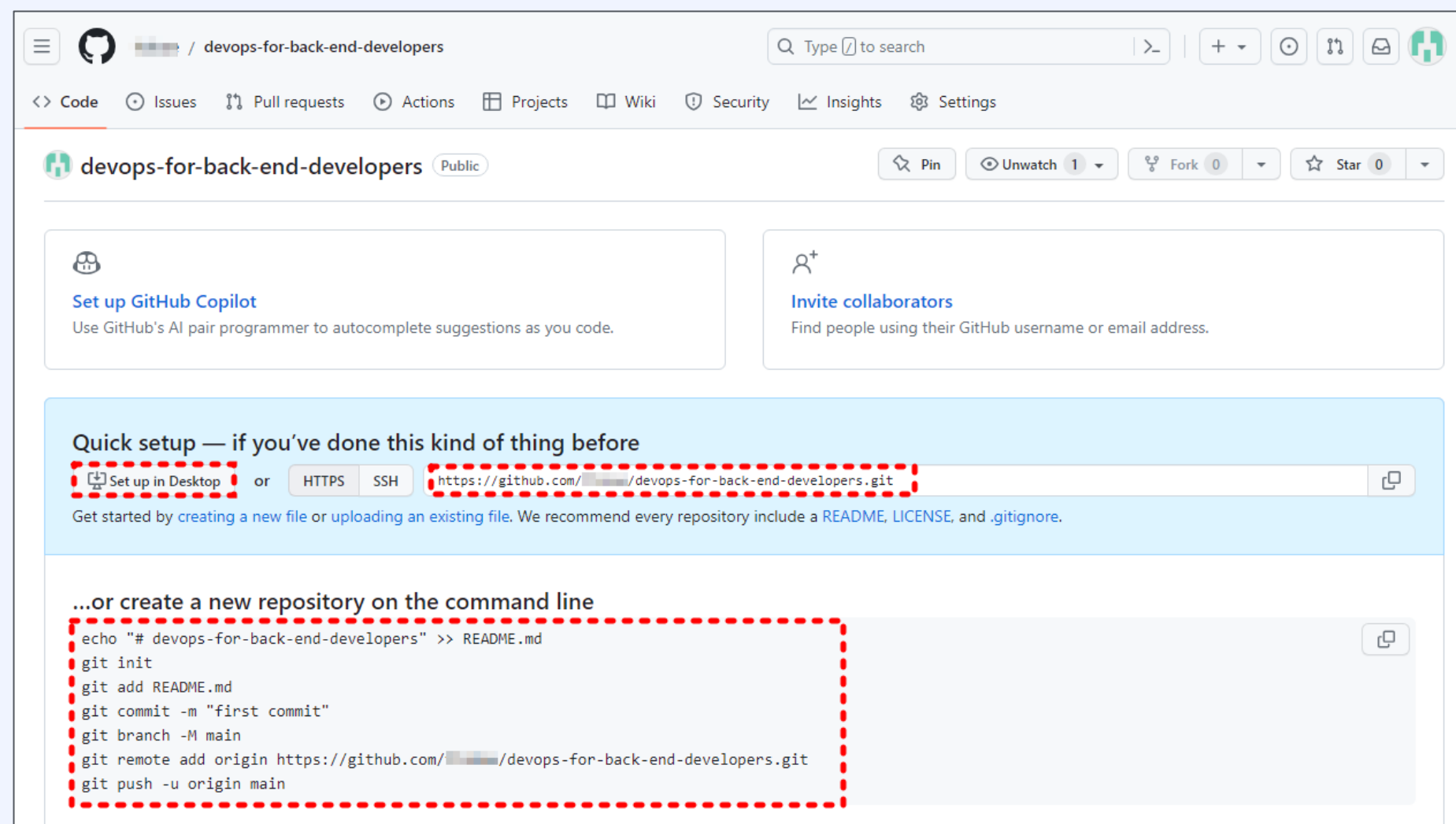
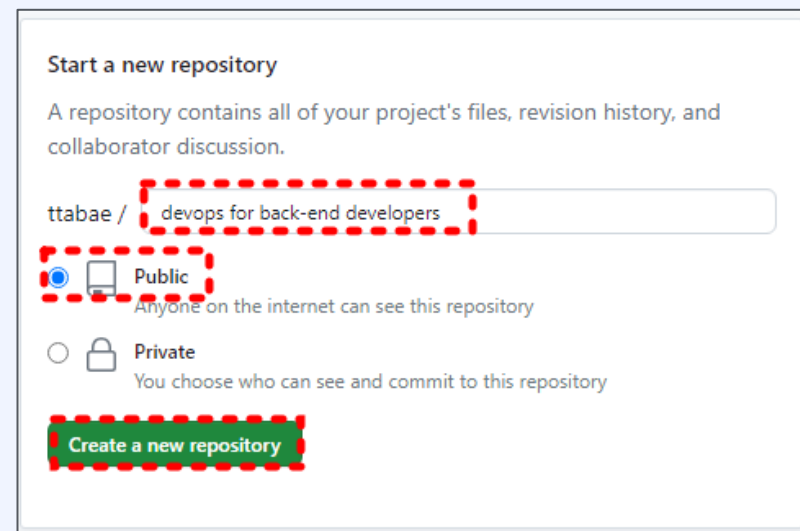
Github 공개 Repository 만들기

2.

Git Branch를 통한
협업 개발하기

GitHub 신규 Repository 만들기

- Start a new repository



- GitHub Desktop 다운로드 후 설치
- 로그인 토큰 만들기
 - 오른쪽 계정 > Settings
 - [Developer settings] - [Personal access tokens] Tokens(classic) > [Generate new token]
 - NOTE : 토큰 설명
 - Expiration : 토큰 만료일
 - Select scopes : 토큰 적용 범위 선택
 - > repo - command line에서 repository access
 - > user - command line 에서 계정 정보 수정
 - [Generate token]
 - 토큰 생성되면 복사해서 저장.

Git Branch를 통한 협업 개발하기

2.

Git Branch를 통한
협업 개발하기

README.md

- Git Repository의 최상위 페이지로 Markdown 문법으로 작성
- 치트시트 : <https://www.markdownguide.org/cheat-sheet/>

GitHub 관리 명령어

- remote repository를 로컬 저장소와 연결
`git remote add [alias-name] [github_url]`
- 마스터 브랜치 업로드
`git push -u [alias-name] branch`
- GitHub Repository master 브랜치에서 다운로드
`git clone [github_url]`

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *

Repository name *

devops-lab

devops-lab is available.

Great repository names are short and memorable. Need inspiration? How about [automatic-meme](#) ?

Description (optional)

example

Public

Anyone on the internet can see this repository. You choose who can commit.

Private

You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

ⓘ You are creating a public repository in your personal account.

Create repository

우리가 배운 내용

- 개발 환경에서의 브랜치 모델
- GitHub 계정 등록
- GitHub 레파지토리 연결 및 사용

백엔드 개발자를 위한 빌드 자동화- Jenkins

3 Jenkins 프로그램 소개 및 설치하기

Jenkins 프로그램 소개 및 설치하기

3.

Jenkins 프로그램
소개 및 설치하기

학습 내용

CICD

Jenkins 설치

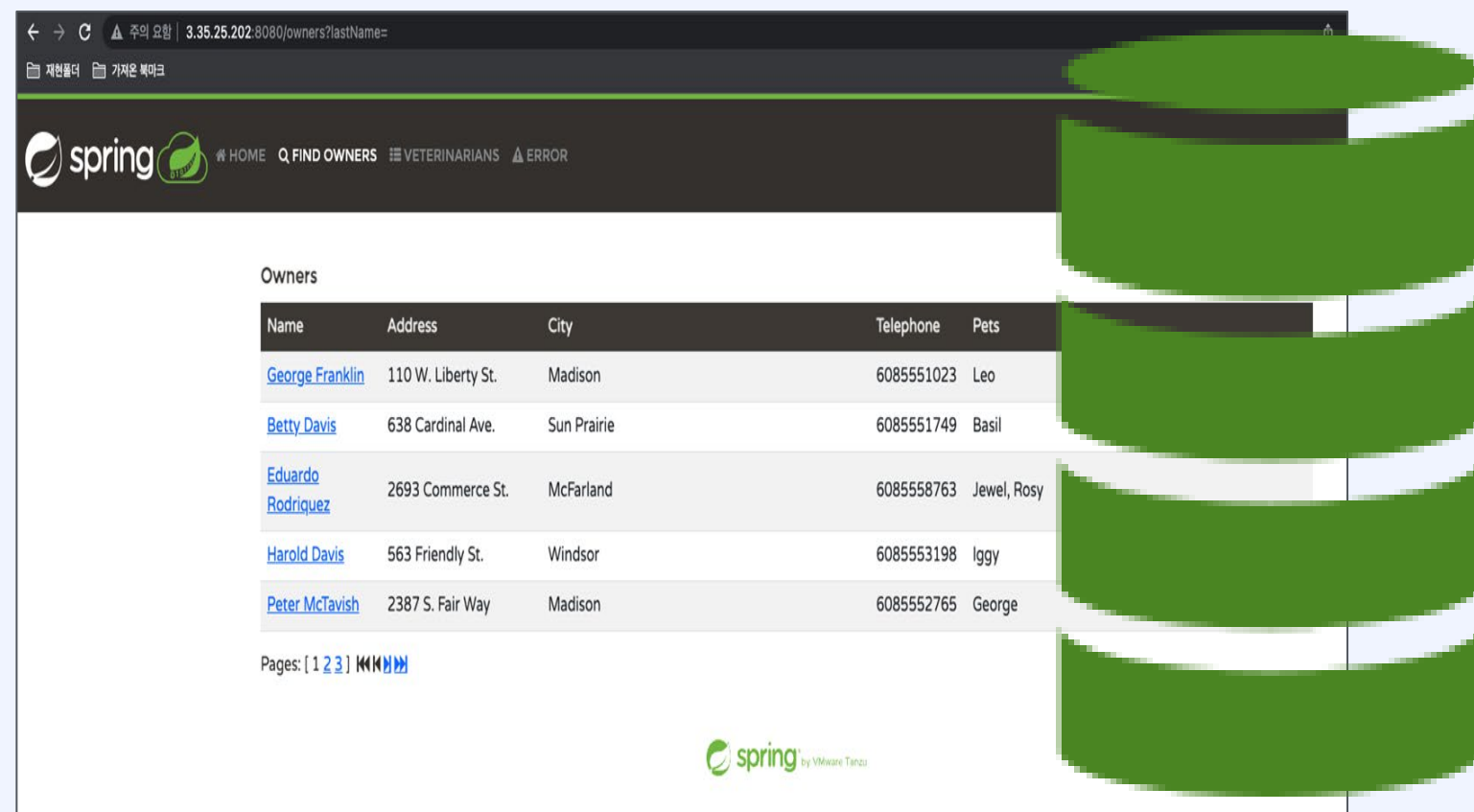
클라우드 인스턴스를 이용한 애플리케이션 운영

3.

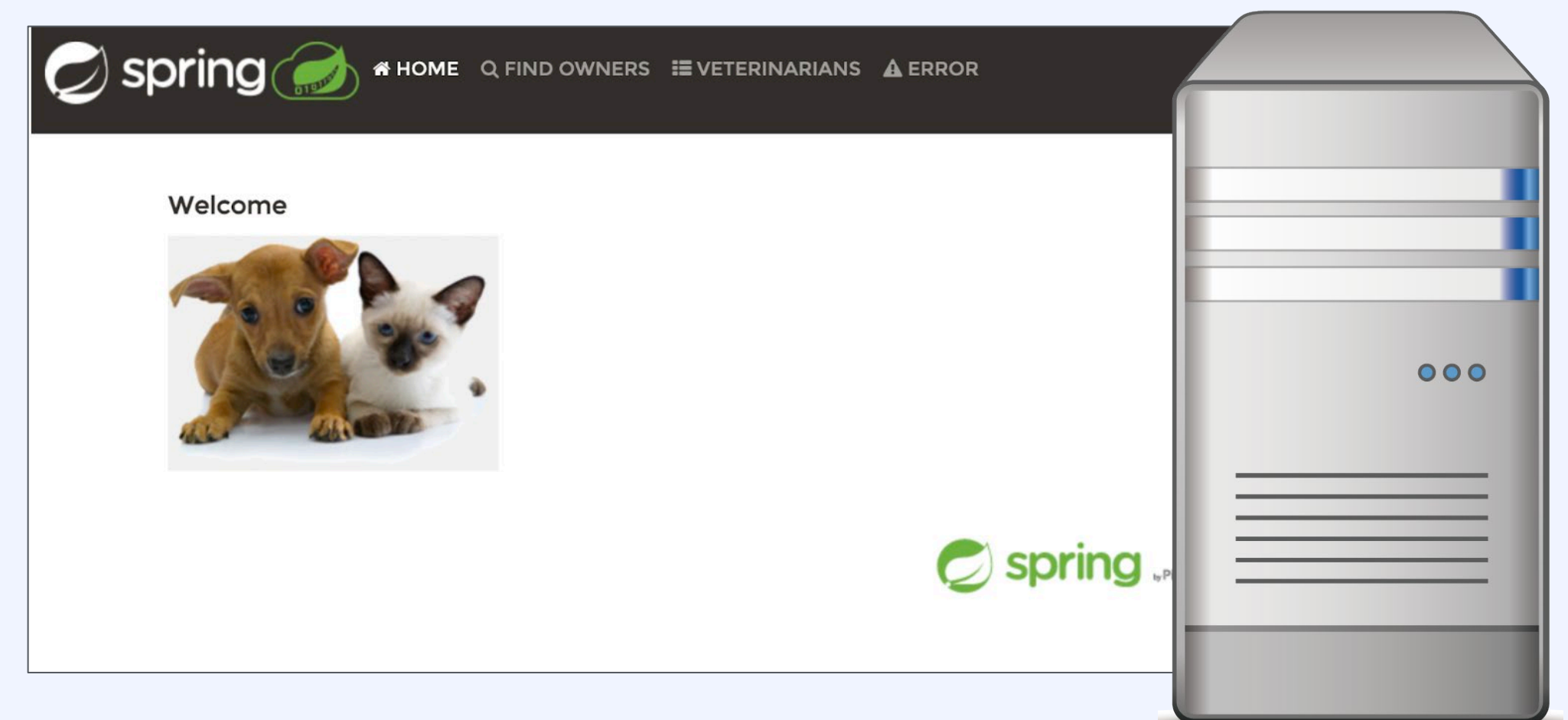
Jenkins 프로그램
소개 및 설치하기

클라우드 인스턴스를 이용한 애플리케이션 운영

Amazon RDS(MySQL 8.x) 구성



PetClinic 애플리케이션 서버 구성 및 실행



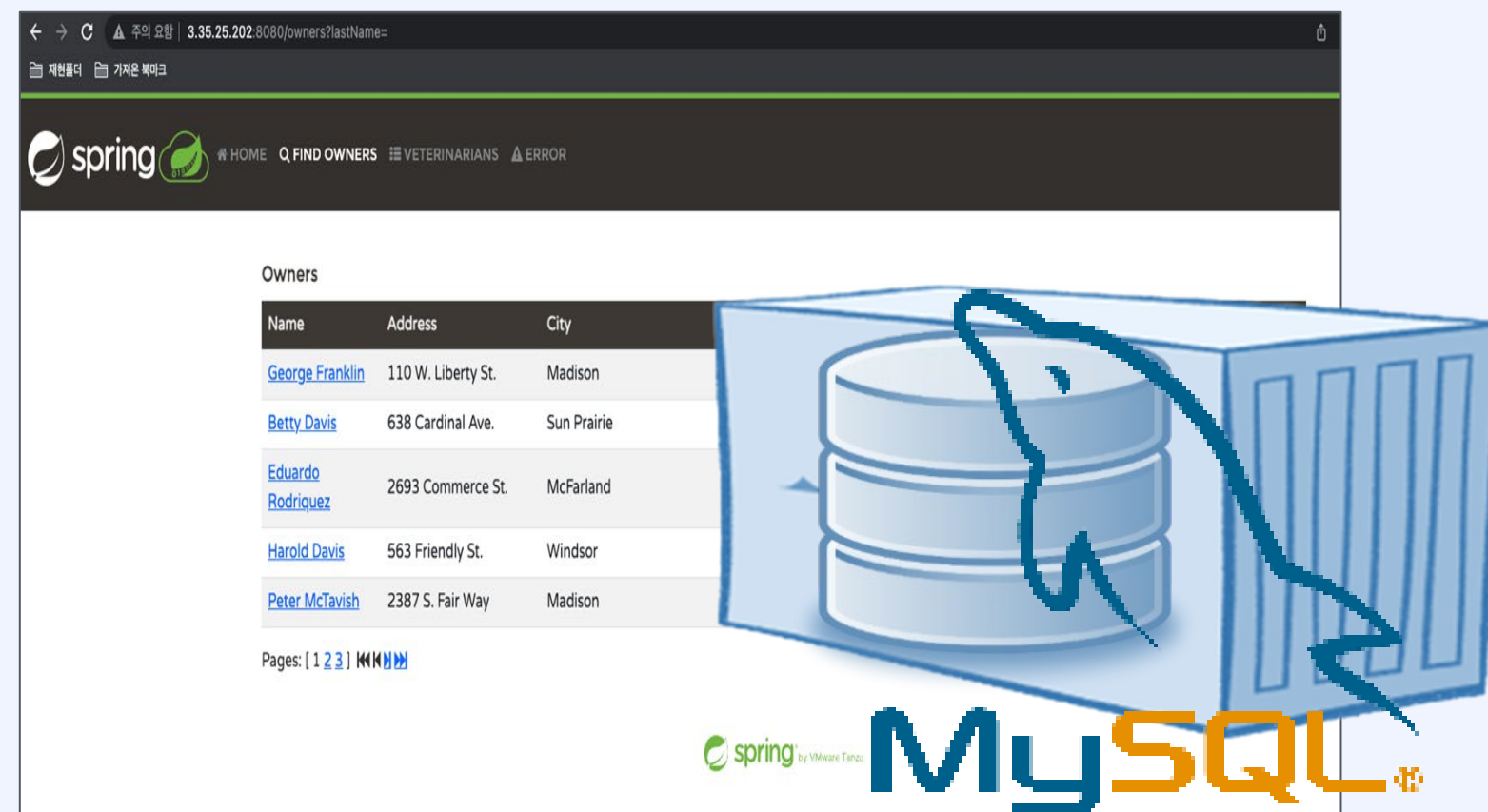
컨테이너 애플리케이션 운영

3.

Jenkins 프로그램
소개 및 설치하기

컨테이너 기반의 애플리케이션을 빌드하고 운영하기

MySQL 8.x 데이터베이스 실행



애플리케이션 컨테이너 빌드 및 운영



CICD 기반의 자동화 배포 및 운영

3.

Jenkins 프로그램
소개 및 설치하기

자동 빌드와 자동 배포 서비스 운영

MySQL 8.x 데이터베이스 실행



애플리케이션 컨테이너 빌드 및 운영



Jenkins 개요

3.

Jenkins 프로그램
소개 및 설치하기

Jenkins 란?

- 소프트웨어 구축, 테스트, 전달 및 배포와 관련된 모든 종류의 작업을 자동화하는데 사용할 수 있는 오픈 소스 자동화 서버로 시스템 패키지, Docker 또는 JRE(Java Runtime Environment)가 설치된 모든 환경에서 실행할 수 있습니다. 젠킨스는 다양한 플러그인들을 조합하여 일을 처리하는 Pipeline을 통해 CI/CD Pipeline을 구축합니다.

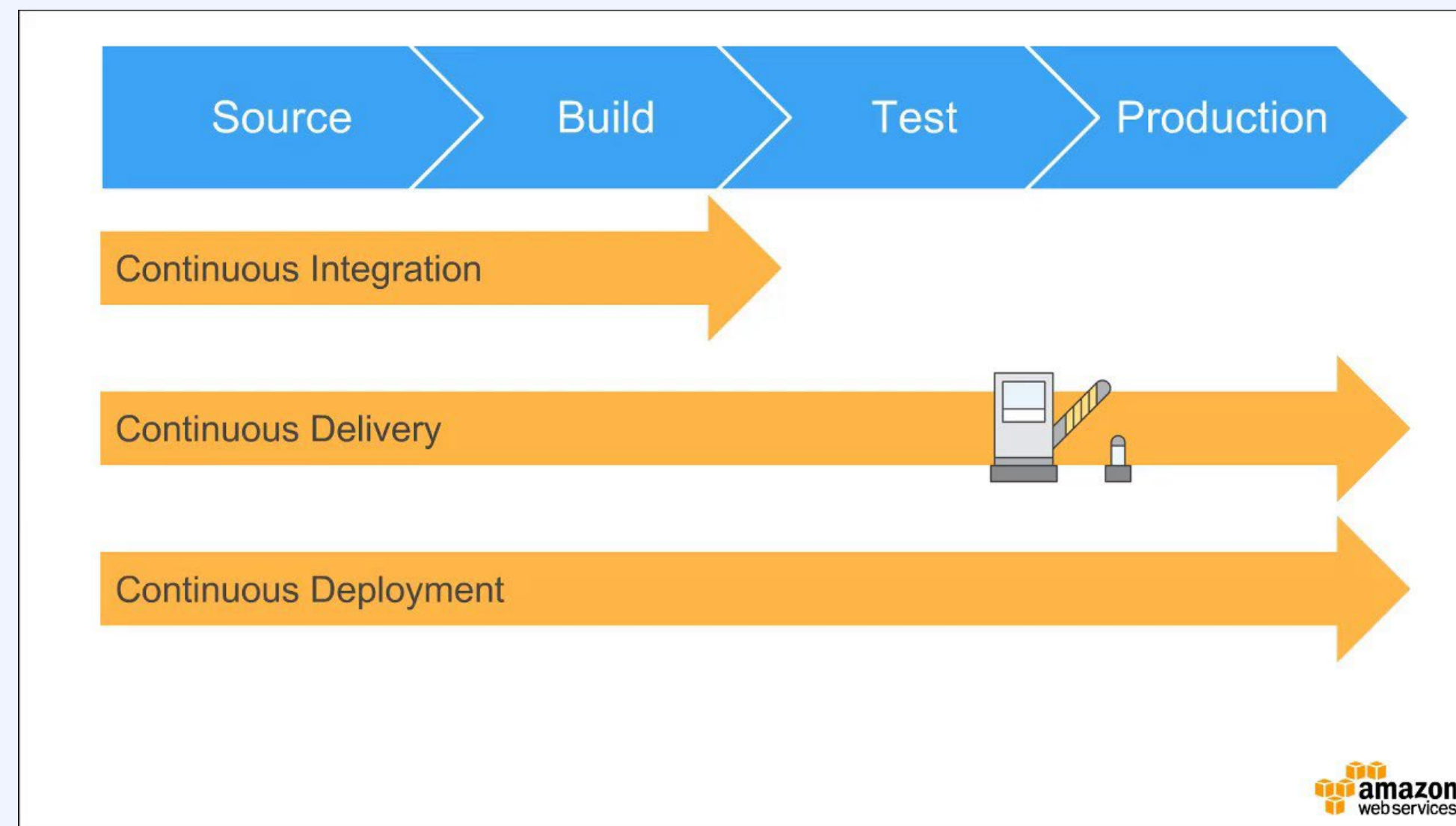
CICD 기반의 자동화 배포 및 운영

3.

Jenkins 프로그램
소개 및 설치하기

CICD

- CI(Continuous Integration, 지속적인 통합)
 - 어플리케이션의 새로운 코드 변경 사항이 정기적으로 빌드 및 테스트 되어 공유 레포지토리에 통합하는 것
- CD(Continuous Delivery/Continuous Deployment, 지속적인 서비스 제공 혹은 지속적인 배포)
 - 개발자의 변경 사항이 레포지토리를 넘어, 고객의 프로덕션(Production) 환경까지 릴리즈 되는 것

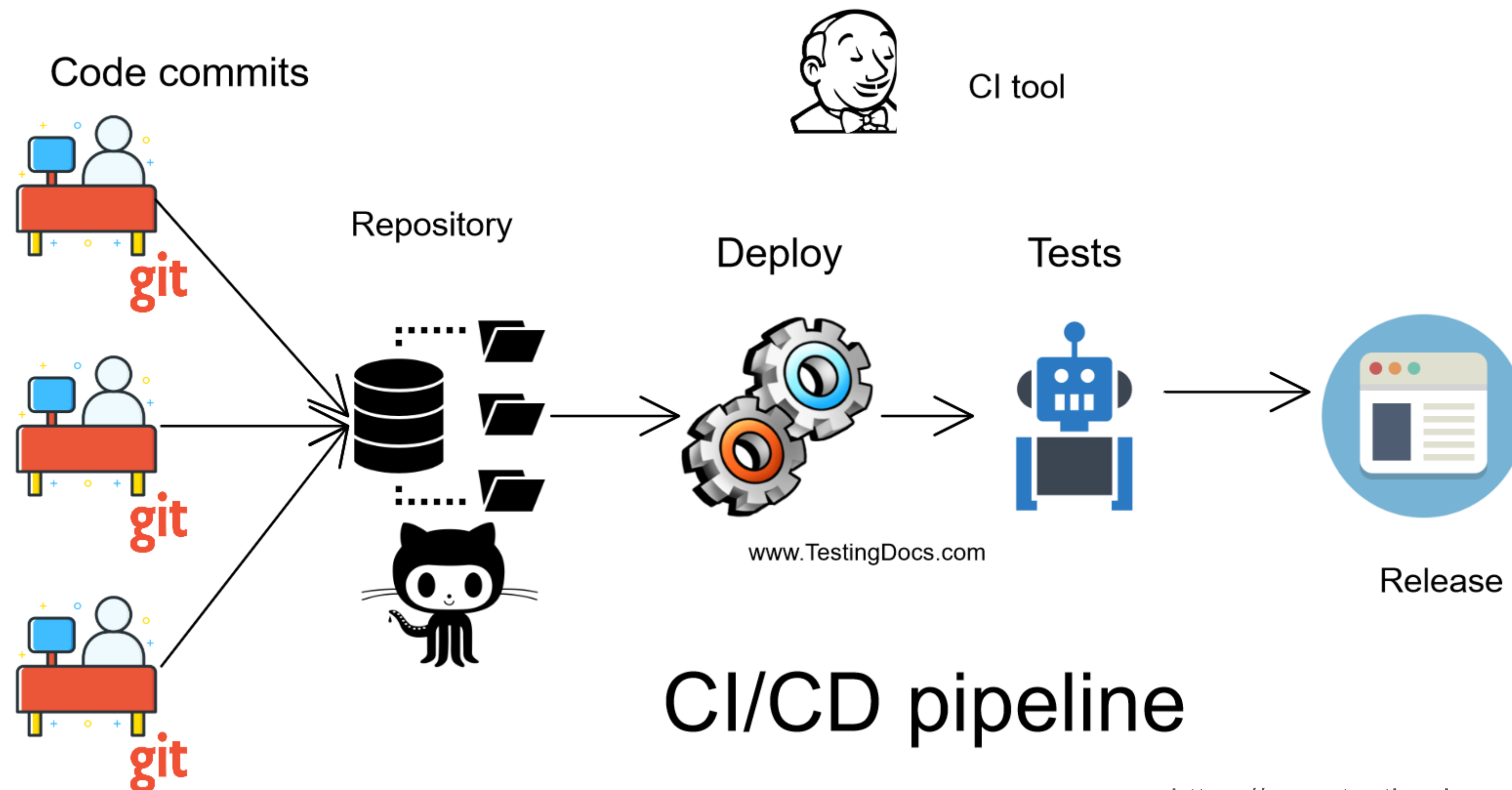


CICD 기반의 자동화 배포 및 운영

3.

Jenkins 프로그램
소개 및 설치하기

CICD



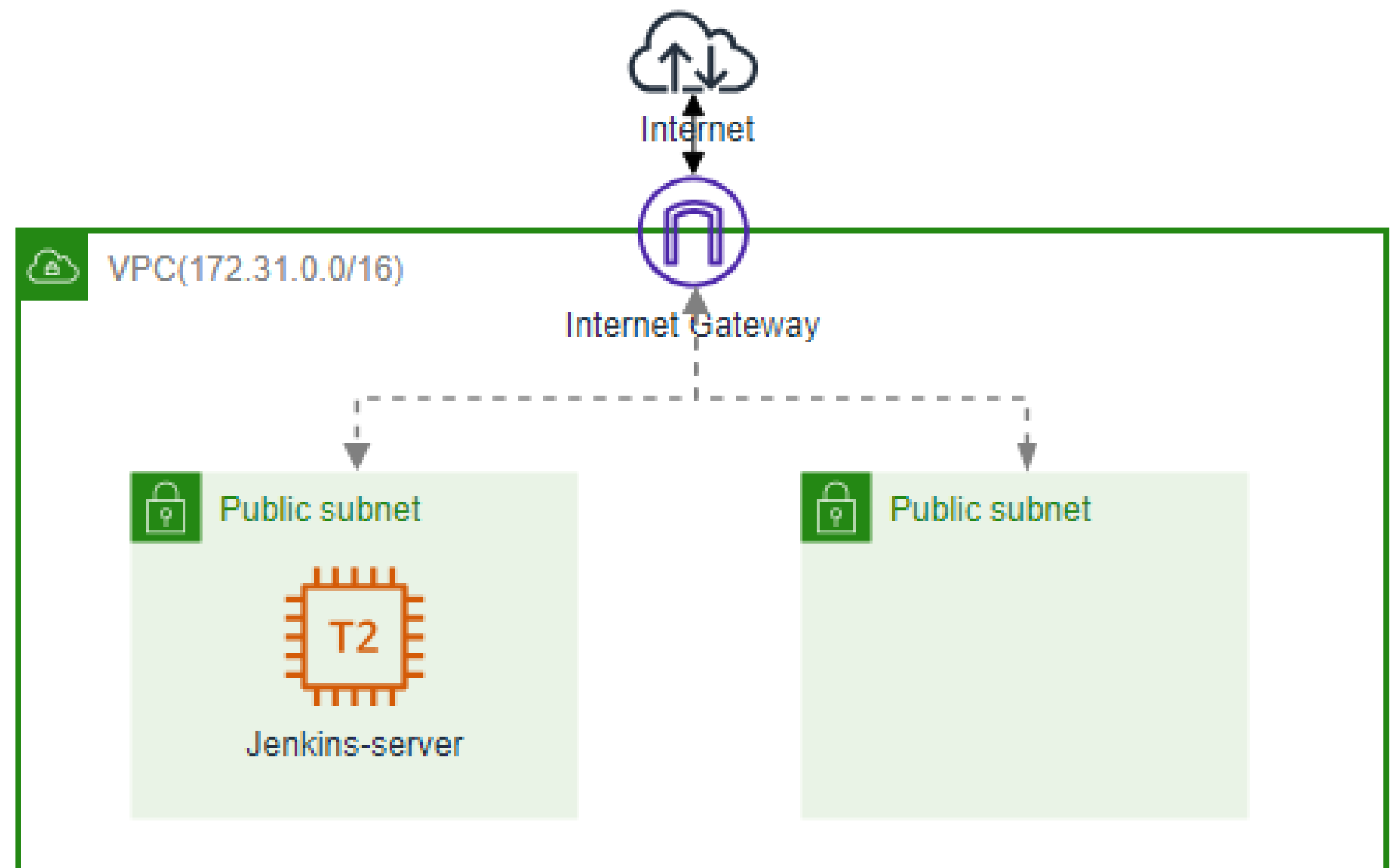
Amazon EC2 에 Jenkins 설치하기(1)

3.

Jenkins 프로그램
소개 및 설치하기

Jenkins server 구성 - 인프라

- EC2
 - name: jenkins-server
 - AMI: Amazon Linux(Kernel 5.10)
 - Instance : **t2.medium**
 - ssh Login Access Key
- VPC
 - default 2a, 퍼블릭 IP 할당
 - 보안그룹: jenkins-all - 모든트래픽허용
- 스토리지
 - 범용 SSD(gp3) : 50GiB



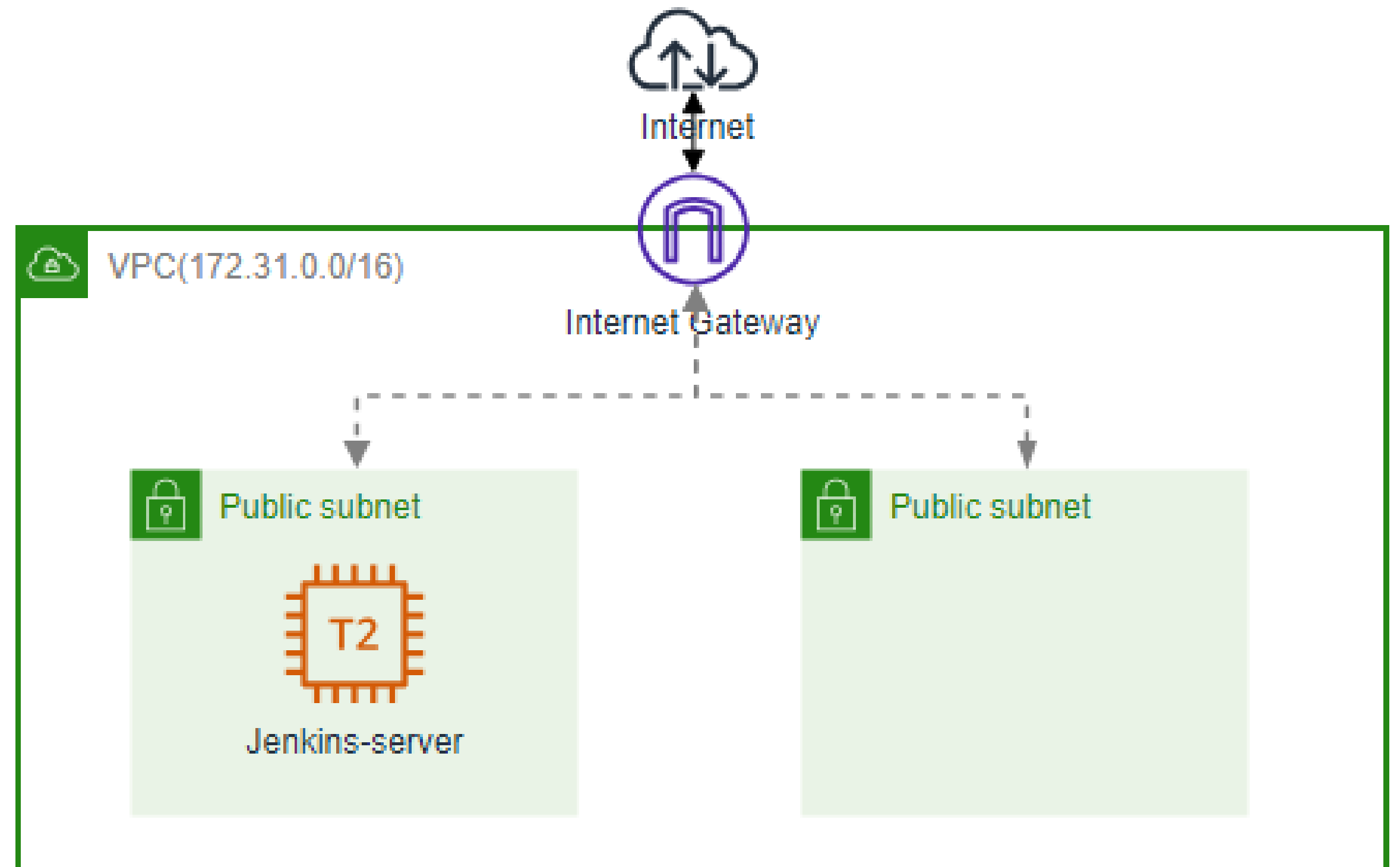
Amazon EC2 에 Jenkins 설치하기(2)

3.

Jenkins 프로그램
소개 및 설치하기

Jenkins server 구성 - 서비스 환경

- docker 설치
 - ec2-user 계정 로그인 후 docker install
- Jenkins
 - 컨테이너 기반 Jenkins 설치
- 스토리지
 - 범용 SSD(gp3) : 50GiB



우리가 배운 내용

- 클라우드 환경에서 Jenkins 인프라 구성
- Jenkins 컨테이너 빌드
- Jenkins 운영

백엔드 개발자를 위한 빌드 자동화- Jenkins

4 Git 연동을 위한 Jenkins 환경설정

Git 연동을 위한 Jenkins 환경설정

4.

Git 연동을 위한
Jenkins 환경설정

학습 내용

Jenkins pipeline 이란?

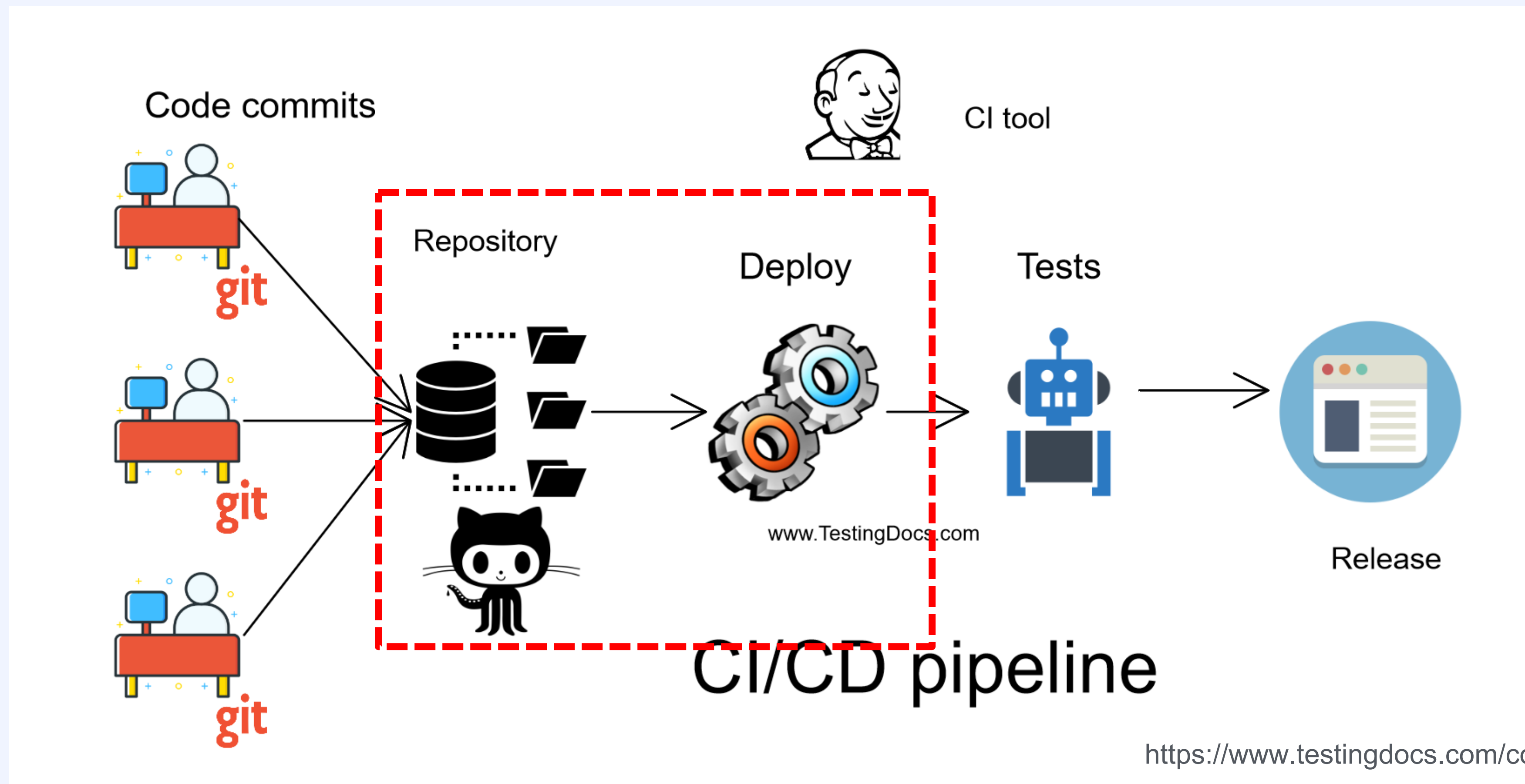
Git 연동을 위한 Jenkins 환경설정

Jenkins Pipeline

4.

Git 연동을 위한
Jenkins 환경설정

Jenkins Pipeline 첫단계 : 깃 연동



Jenkins Pipeline Script란

Jenkins pipeline script

- Pipeline DSL(Domain Specific Language) 코드로 워크로드 정의
- Job을 수동으로 설치할때 반복되는 생성, 버전관리 등의 유지보수의 어려움 대응 - 코드로 운영

pipeline { ... } : 파이프라인 구문의 시작/종료

section

agent : 블록의 최상단에 기록

jenkins 실행자를 전체 파이프라인에 사용(any), 특정 stage에 사용 (none)할지 설정.

stage : pipeline에는 하나 이상의 stage를 포함.

steps 하위 항목이 Jenkins 플러그인(sh: Shell Script)으로 실행되어 명령어 실행

```
stage('stage명') {
    steps{
        할 일
    }
}
```

Directives : 파이프라인의 config 설정 값

environment : 파이프라인 혹은 스테이지 내부에서 쓰일 변수값을 정의

tool : 자동 설치나 Path에 추가할 도구를 정의

parameter : 파라미터 정의

```
1 pipeline {
2   agent any
3   stages {
4     stage('sh 플러그인 실행') {
5       steps {
6         sh '''
7           id
8           echo $$
9           sleep 30
10          pwd
11          cat /proc/$(echo $$)/cmdline
12          env | sort
13          ...
14        '''
15      }
16    }
17  }
```

git 연동을 위한 Jenkins 파이프라인 구성: 수동빌드

4.

Git 연동을 위한
Jenkins 환경설정

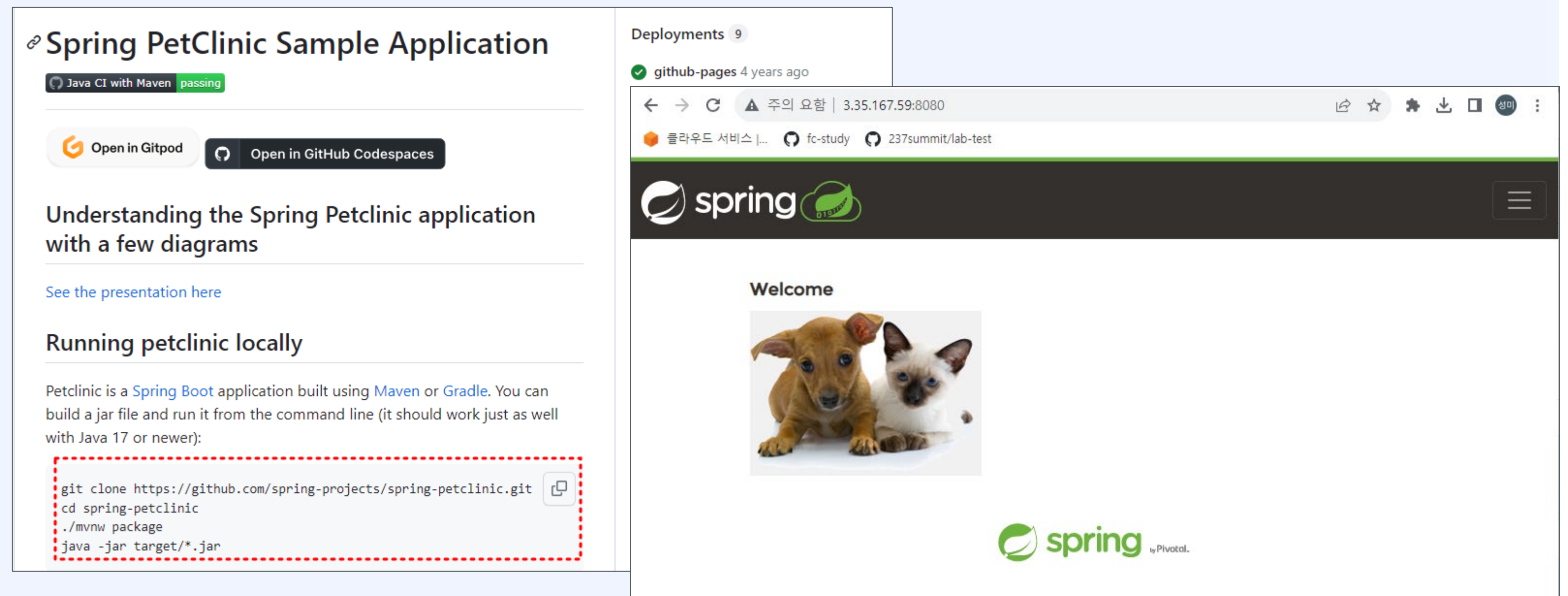
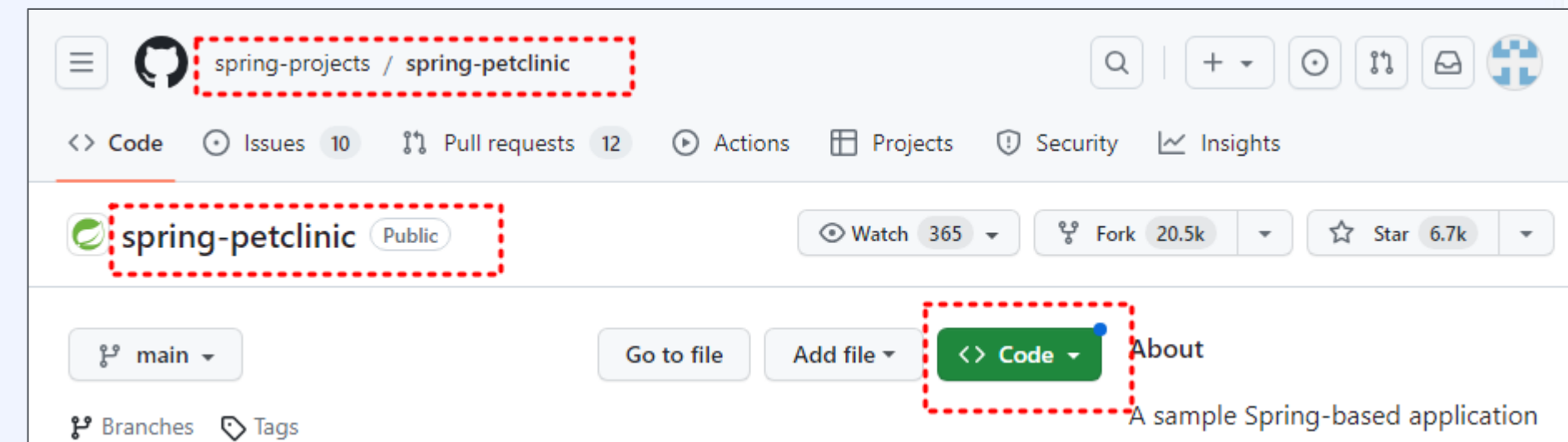
수동빌드 예:
소스 코드 다운로드 후 github에 binary 업로드 - 구현

```
git clone https://github.com/spring-projects/spring-petclinic.git
cd spring-petclinic
./mvnw package
```

target/petclinic-2.7.0-SNAPSHOT.jar

동작 TEST
java -jar target/*.jar

웹브라우저 접속 TEST
http://jenkins-server's_EIP:8080/



git 연동을 위한 Jenkins 파이프라인 구성: 자동 빌드

4.

Git 연동을 위한
Jenkins 환경설정

Jenkins Project 만들기

- 새로운 Item 만들기
 - item name: petclinic
 - Pipeline
- Configure
 - 설명 : 1. GIT: petclinic build
 - Pipeline : pipeline script
 - 스크립트 넣고 [저장]
- 빌드 진행
- 콘솔 로그보기
- 아카이브 링크 확인

The screenshot shows the Jenkins 'New Item' configuration page for a Pipeline. The 'petclinic' item name is entered. The 'Pipeline' option is selected. The 'Pipeline script' section is expanded, showing a Groovy script for a pipeline with two stages: 'Checkout' and 'Build'. The 'Checkout' stage uses the 'git' step to fetch code from a GitHub repository. The 'Build' stage uses the 'sh' step to run 'mvnw clean package'. The 'post' section shows build success messages. The 'OK' button is highlighted. The 'Pipeline Syntax' section shows the script being saved. The 'Last Successful Artifacts' section shows the built artifact 'spring-petclinic-2.7.0-SNAPSHOT.jar' with a size of 50.71 MB. The 'Stage View' section shows the average stage times for 'Checkout' (1s) and 'Build' (6min 13s).

Enter an item name

petclinic
» Required field

Freestyle project
이것은 Jenkins의 주요 기능
소프트웨어 빌드보다 다른 어

Pipeline
Orchestrates long-running
workflows) and/or organiz

Multi-configuration p
다양한 환경에서의 테스트,

Folder
Creates a container that st
creates a separate namesp

Multibranch Pipeline
Creates a set of Pipeline p

Organization Folder
Creates a set of multibranc

If you want to create a new item

OK

Pipeline

Definition

Pipeline script

Script ?

```

1 pipeline {
2   agent any
3   stages {
4     stage('Checkout') {
5       steps {
6         git branch: 'main', url: 'https://github.com/237summit/petclinic.git'
7       }
8     }
9   }
10
11   stage('Build') {
12     steps {
13       sh './mvnw clean package'
14     }
15   }
16
17   post {
18     success {
19       Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-compress/1.20/commons-compress-1.20.jar (632 kB at 7.3 MB/s)
20       Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.3.1/plexus-utils-3.3.1.jar (262 kB at 2.9 MB/s)
21       [INFO] Building jar: /var/jenkins_home/workspace/petclinic/target/spring-petclinic-2.7.0-SNAPSHOT.jar
22       [INFO]
23       [INFO] --- spring-boot-maven-plugin:2.7.1:repackage (repackage) @ spring-petclinic ---
24       [INFO] Replacing main artifact with repackaged archive
25       [INFO] -----
26       [INFO] BUILD SUCCESS
27       [INFO] -----
28       [INFO] Total time: 06:11 min
29       [INFO] Finished at: 2023-08-07T03:05:06Z
30       [INFO] -----
31       Post stage
32       [Pipeline] archiveArtifacts
33       Archiving artifacts
34       [Pipeline] }
35       [Pipeline] // stage
36       [Pipeline] }
37       [Pipeline] // node
38       [Pipeline] End of Pipeline
39       Finished: SUCCESS

```

소스코드 다운로드

컴파일 및 결과확인

저장 **Apply**

Use Groovy Sand

Pipeline Syntax

Last Successful Artifacts
spring-petclinic-2.7.0-SNAPSHOT.jar 50.71 MB view

Stage View

	Checkout	Build
Average stage times: (Average full run time: ~6min 15s)	1s	6min 13s
#1 8월 07 일 11:58	1s	6min 13s

우리가 배운 내용

- DSL코드를 이용한 Jenkins Pipeline 구성
- Git 연동으로 소스코드 다운로드 및 컴파일

백엔드 개발자를 위한 빌드 자동화- Jenkins

5 Jenkins 파이프라인을 이용한 컨테이너 빌드

Jenkins 파이프라인을 이용한 컨테이너 빌드

5.

Jenkins
파이프라인을
이용한 컨테이너
빌드

학습 내용

도커 플러그인 설치

Pipeline을 이용한 컨테이너 빌드

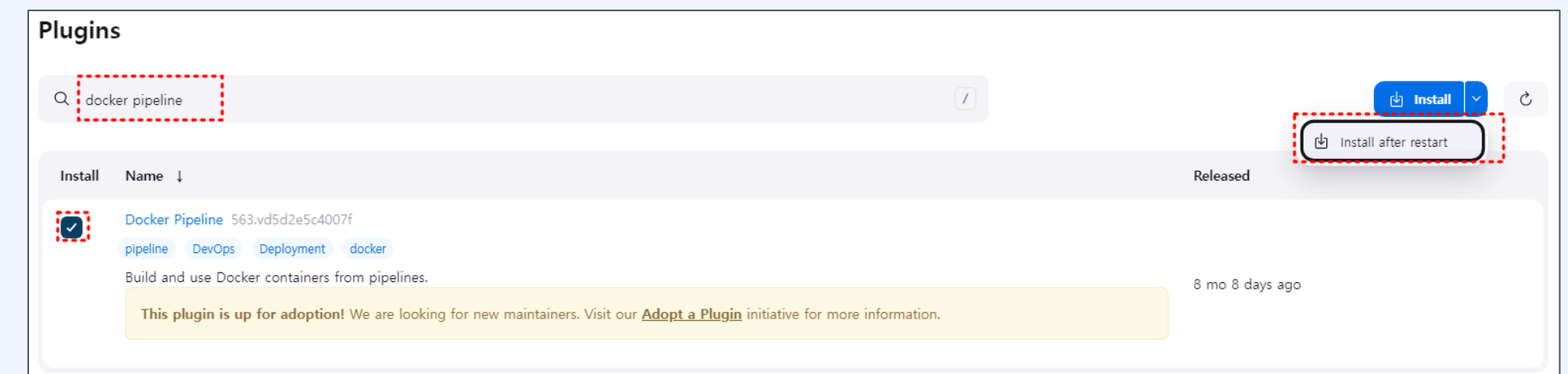
Jenkins Pipeline을 이용한 컨테이너 빌드

5.

Jenkins
파이프라인을
이용한 컨테이너
빌드

Jenkins Pipeline을 이용한 Docker 연동

- 플러그인 설치
 - Jenkins 관리 -> [System Configuration] - Plugins -> [Available package] - 'docker pipeline' 검색후 선택 > Install
- Pipeline script 구성후 컨테이너 빌드



```
pipeline {
  agent any
  stages {
    stage('Checkout') {
      steps {
        git branch: 'main', url: 'https://github.com/237summit/petclinic.git'
      }
    }
    stage('Build') {
      steps {
        sh './mvnw clean package'
      }
    }
    stage('Docker Image Build') {
      steps {
        script {
          docker.build("petclinic:v${BUILD_ID}")
        }
      }
    }
  }
}
```

우리가 배운 내용

- Jenkins Plugin 설치
- Jenkins pipeline을 이용한 컨테이너 빌드

백엔드 개발자를 위한 빌드 자동화- Jenkins

6 Jenkins Slack Webhook 연동하기

Jenkins Slack Webhook 연동하기

6.

Jenkins Slack
Webhook 연동하기

학습 내용

Slack의 워크스페이스와 채널 생성

젠킨스에 슬랙 플러그인 설치 및 구성

Jenkins Pipeline 이용해 Slack 연동

Jenkins Slack Webhook 연동하기

6.

Jenkins Slack
Webhook 연동하기

Slack 이란?

- Slack은 채널 기반 메시징 플랫폼입니다. Slack 워크스페이스는 사람들이 협업하고 모든 소프트웨어 도구와 서비스를 연결하며 최고의 작업을 수행하는 데 필요한 정보를 찾을 수 있는 장소입니다. - slack.com



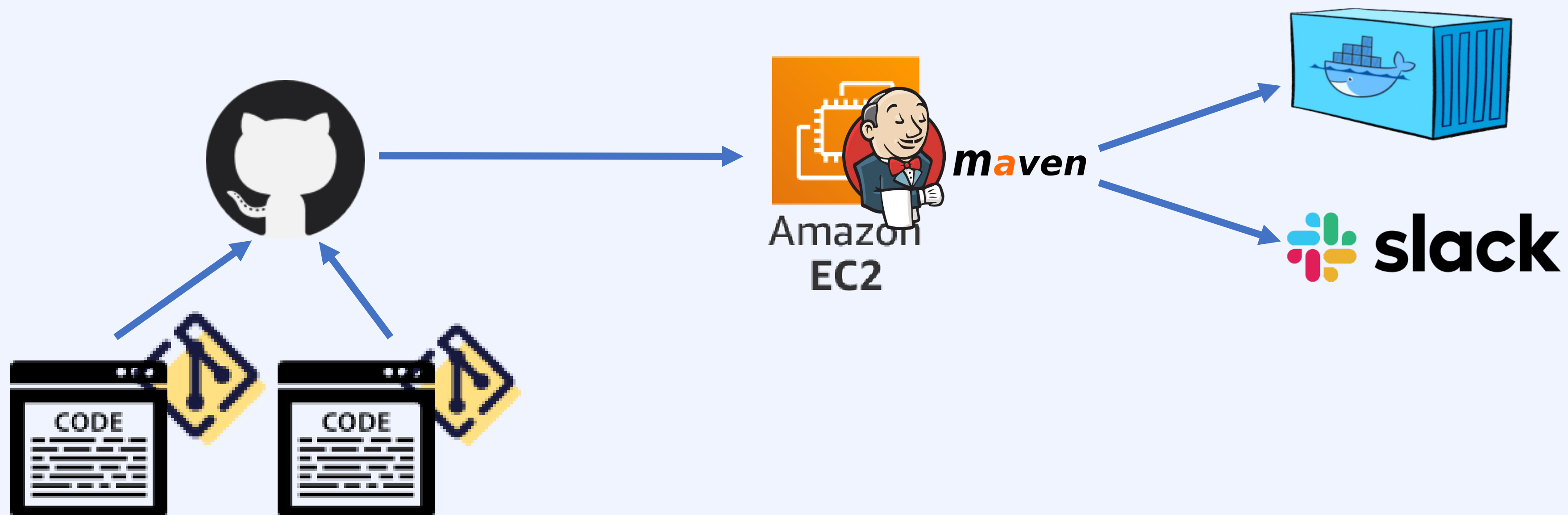
Jenkins Slack Webhook 연동하기

6.

Jenkins Slack
Webhook 연동하기

Jenkins Slack 연동

- Jenkins 에서 Pipeline script를 동작할때 성공 및 실패 여부를 슬랙을 통해 noti 할수 있다.



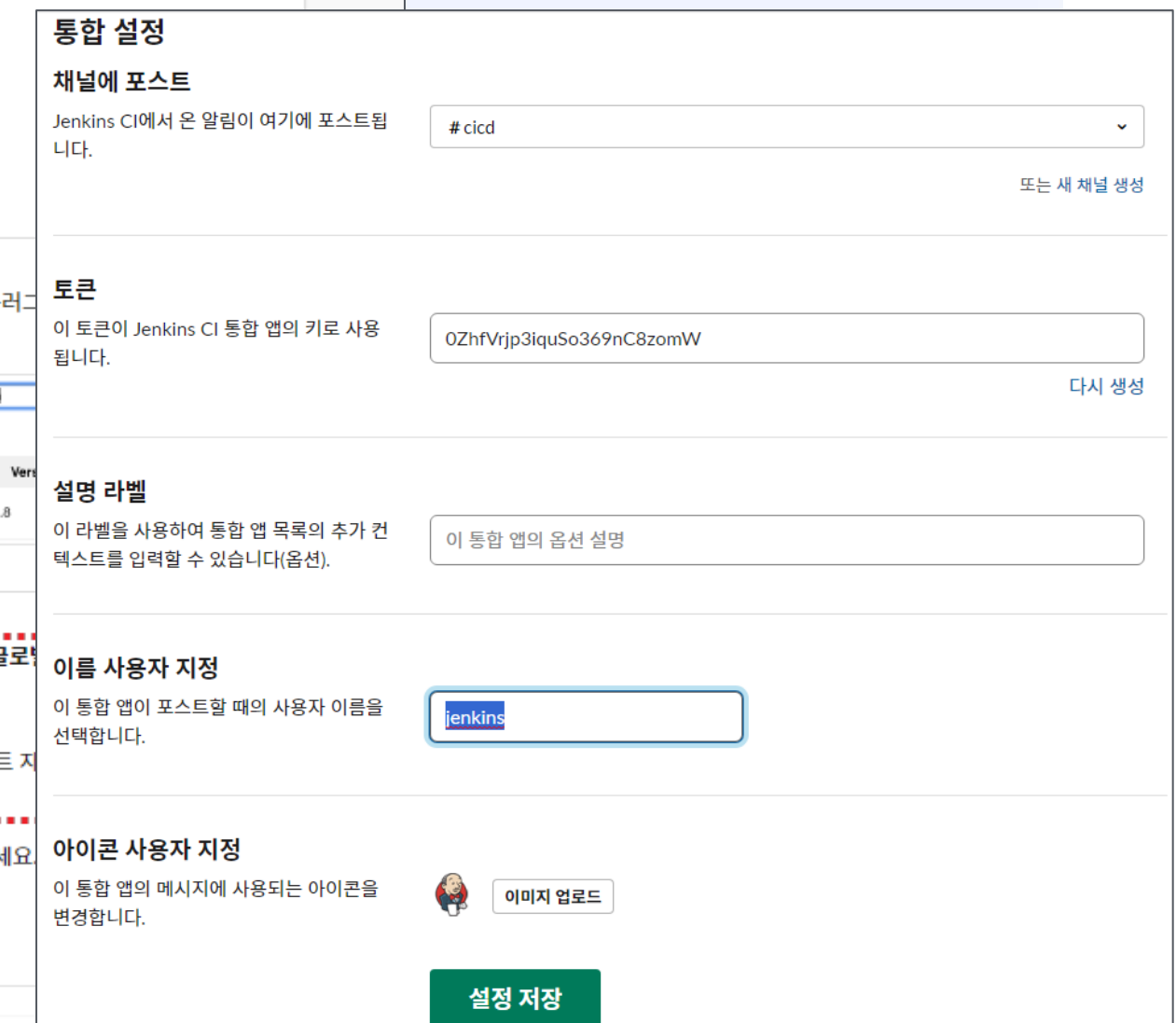
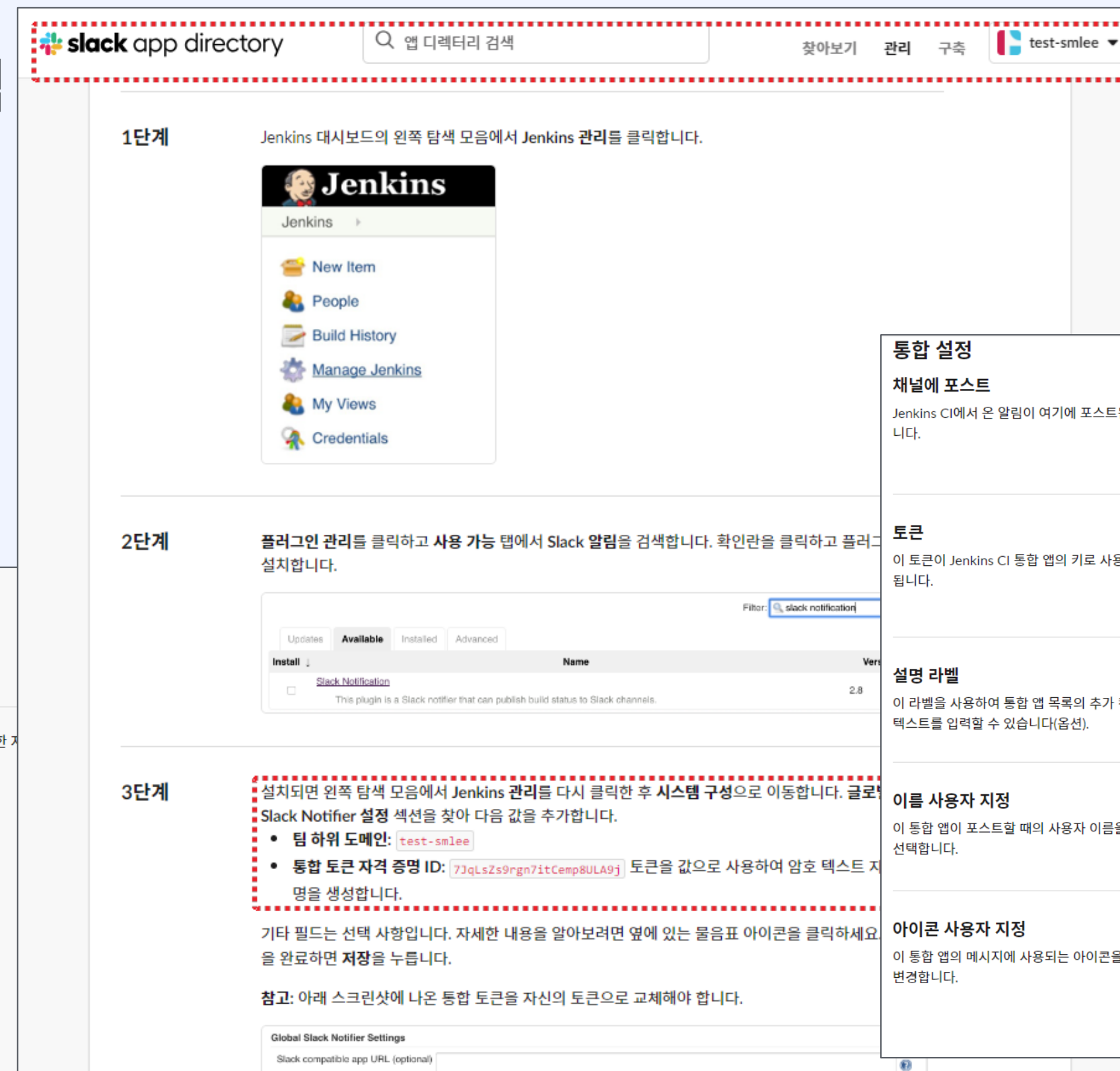
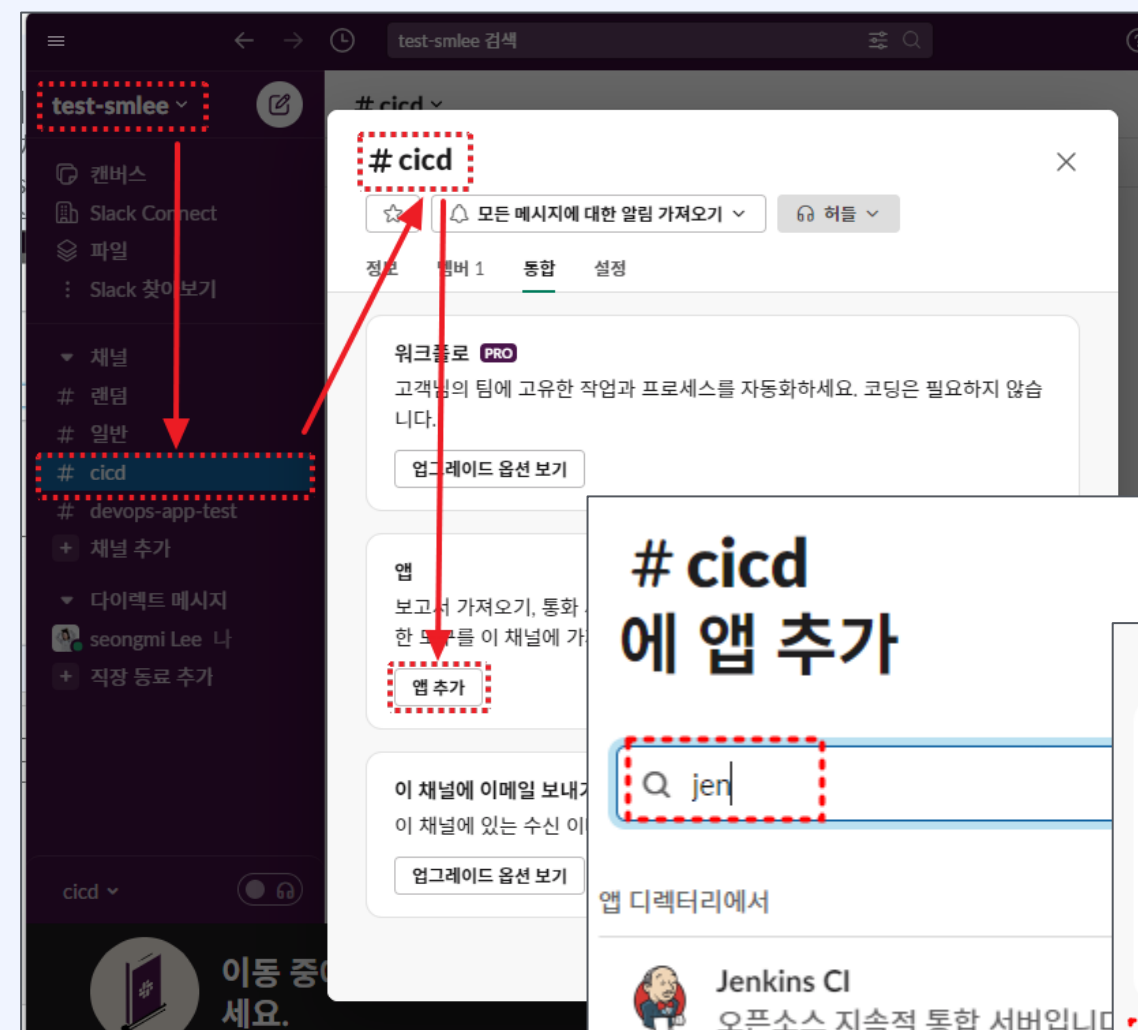
Jenkins Slack Webhook 연동하기

6.

Jenkins Slack Webhook 연동하기

Slack과 Jenkins 연동하기

- 슬랙 워크스페이스 생성 -> 채널 생성
- 채널에 Jenkins 연동 : Jenkins에서 slack 채널에 연결할 수 있도록 권한 할당
 - 통합 > 앱 추가 > jenkins 검색 후 설치
- 웹 브라우저 실행되면 - 슬랙에 추가 > 채널 선택



Jenkins Slack Webhook 연동하기

6.

Jenkins Slack Webhook 연동하기

Slack과 Jenkins 연동하기

- 슬랙 워크스페이스 생성 -> 채널 생성
- 채널에 Jenkins 연동 : Jenkins에서 slack 채널에 연결할 수 있도록 권한 할당
 - 통합 > 앱 추가 > jenkins 검색 후 설치 > 웹 브라우저 실행되면 - 슬랙에 추가 > 채널 선택
- Jenkins 관리
 - 플러그인 설치: 'slack notification' 플러그인 설치
 - 시스템 구성 - slack (Credential - [Add] 클릭)
 - petclinic 프로젝트에 pipeline script 로 slack noti 구성

3단계

설치되면 왼쪽 탐색 모음에서 Jenkins 관리를 다시 클릭한 후 시스템 구성으로 이동합니다. **글로벌 Slack Notifier 설정** 섹션을 찾아 다음 값을 추가합니다.

- 팀 하위 도메인: test-smlee
- 통합 토큰 자격 증명 ID: 7JqLsZs9rgn7itCemp8ULA9j 토큰을 값으로 사용하여 암호 텍스트 필드를 생성합니다.

기타 필드는 선택 사항입니다. 자세한 내용을 알아보려면 옆에 있는 물음표 아이콘을 클릭하세요. 작업을 완료하면 **저장**을 누릅니다.

참고: 아래 스크린샷에 나온 통합 토큰을 자신의 토큰으로 교체해야 합니다.

Global Slack Notifier Settings

Slack compatible app URL (optional)

Dashboard > Jenkins 관리 > System >

E-mail로 알려줌

SMTP 서버

Default user e-mail suffix

Test configuration by sending

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain

Global credentials (unrestricted)

Kind

Secret text

Scope

Global (Jenkins, nodes, items, all child items, etc)

Secret

.....

ID

slack-token

Description

slack-token

Add Cancel

Slack

Workspace ?

test-smlee

Credential ?

slack-token

Add

Default channel / member id ?

#cicd

Custom slack app bot user ?

고급

저장 Apply

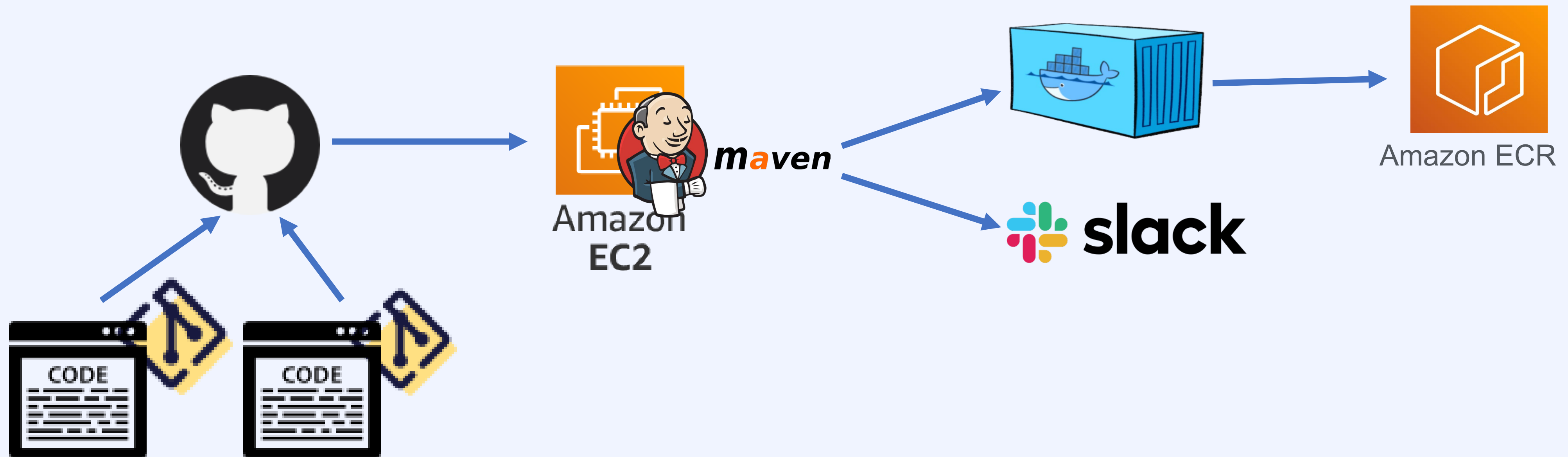
백엔드 개발자를 위한 빌드 자동화- Jenkins

6 Jenkins Slack Webhook 연동하기(2)

Amazon ECR 구성(1)

6.

Jenkins Slack
Webhook
연동하기(2)



Jenkins Slack Webhook 연동하기(2)

6.

Jenkins Slack
Webhook
연동하기(2)

학습 내용

AWS :

ECR 정책 생성
사용자 콘솔 액세스키 생성
Amazon ECR 생성

Jenkins:

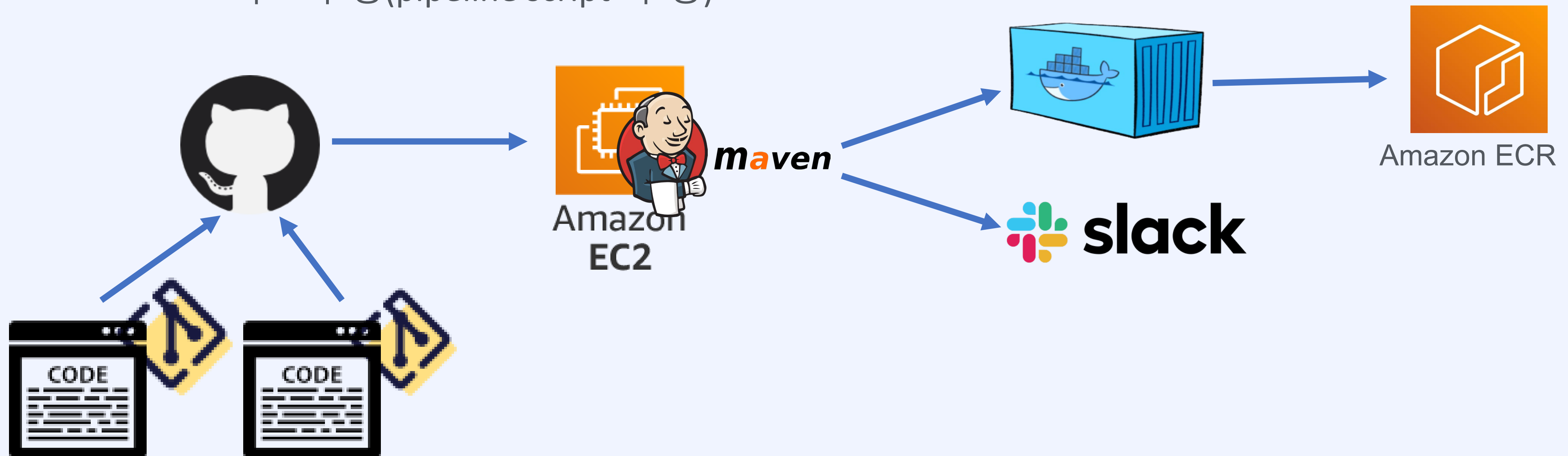
Plugin : AWS Credentials, Pipeline: AWS Steps
Credentials : 사용자 콘솔 액세스키 등록
프로젝트 구성 : pipeline script 구성

Amazon ECR 구성(1)

6.

Jenkins Slack
Webhook
연동하기(2)

- **빌드 된 컨테이너 이미지가 로컬이 아닌 Amazon ECR에 저장**
- **AWS** : ECR 정책(role) 생성 -> 사용자 콘솔 액세스 키 생성 -> Amazon ECR 생성
- **Jenkins**: Plugin(AWS Credentials, Pipeline: AWS Steps)설치 -> Jenkins-Server: aws cli 설치
Jenkins 관리: AWS Credentials 등록
프로젝트 구성(pipeline script 구성)



Amazon ECR 구성(1)

6.

Jenkins Slack
Webhook
연동하기(2)

AWS 구성

- ECR 역할(role) 생성 : **ecr-registry-full-access(AmazonEC2ContainerRegistryPowerUser)**
- 사용자 콘솔 액세스키 생성
- Amazon ECR 생성

IAM > 역할 > 역할 생성

1단계
신뢰할 수 있는 엔터티 선택

신뢰할 수 있는 엔터티 선택 정보

신뢰할 수 있는 엔터티 유형

☐ AWS 서비스
EC2, Lambda 등의
AWS 서비스가 이 계
정에서 작업을 수행하
도록 허용합니다.

☒ AWS 계정
사용자 또는 서드 파티
에 속한 다른 AWS 계
정의 엔터티가 이 계
정에서 작업을 수행하
도록 허용합니다.

☐ SAML 2.0 연동
기업 디렉터리에서
SAML 2.0과 연동된
사용자가 이 계정에서
작업을 수행할 수 있
도록 허용합니다.

☐ 사용자 지정 신뢰
정책
다른 사용자가 이 계
정에서 작업을 수행할 수
있도록 사용자 지정 신
뢰 정책을 생성합니다.

AWS 계정

사용자 또는 서드 파티에 속한 다른 AWS 계정의 엔터티가 이 계정에서 작업을 수행하도록 허용합니다.

☒ 이 계정(047675330097)

☐ 다른 AWS 계정

옵션

☐ 외부 ID 필요(서드 파티가 이 역할을 수임하는 경우 모범 사례)

☐ MFA 필요
수임 엔터티가 멀티 팩터 인증을 사용해야 합니다.

취소

다음

IAM > 역할 > 역할 생성

1단계
신뢰할 수 있는 엔터티 선택

권한 추가 정보

권한 정책 (선택됨 1/876) 정보
새 역할에 연결할 정책을 하나 이상 선택합니다.

필터 지우기

정책 이름

☐ AmazonEC2ContainerServiceforEC2Role

☐ AmazonEC2ContainerServiceRole

☐ AmazonEC2ContainerRegistryReadOnly

☒ AmazonEC2ContainerRegistryPowerUser

취소

다음

IAM > 역할 > 역할 생성

1단계
신뢰할 수 있는 엔터티 선택

2단계
권한 추가

3단계
이름 지정, 검토 및 생성

이름 지정, 검토 및 생성

역할 세부 정보

역할 이름
이 역할을 식별하는 의미 있는 이름을 입력합니다.
최대 64자입니다. 영숫자 및 '+', '@', '_' 문자를 사용하세요.

설명
이 역할에 대하여 간단한 설명을 추가합니다.
최대 1,000자입니다. 영숫자 및 '+', '@', '_' 문자를 사용하세요.

취소

다음

Amazon ECR 구성(2)

6.

Jenkins Slack
Webhook
연동하기(2)

AWS 구성

- ECR 권한 생성
- 사용자 콘솔 액세스키 생성 : user > 계정선택 > 액세스키, 비밀 액세스키 복사
- Amazon ECR 생성

IAM > 사용자 > developer > 액세스 키 만들기

1단계
액세스 키 모범 사례 및 대안

2단계 - 선택 사항
설명 태그 설정

3단계
액세스 키 검색

액세스 키 모범 사례 및 대안 정보

보안 개선을 위해 액세스 키와 같은 장기 자격 증명을 사용하지 마세요. 다음과 같은 사용 사례와 대안을 고려하세요.

사용 사례

- ☒ Command Line Interface(CLI)
AWS CLI를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.

확인

☒ 위의 권장 사항을 이해했으며 액세스 키 생성을 계속하려고 합니다.

취소 다음

IAM > 사용자 > developer > 액세스 키 만들기

1단계
액세스 키 모범 사례 및 대안

2단계 - 선택 사항
설명 태그 설정

3단계
액세스 키 검색

설명 태그 설정 - 선택 사항 정보

이 액세스 키에 대한 설명은 이 사용자에게 태그로 연결되고, 액세스 키와 함께 표시됩니다.

설명 태그 값

이 액세스 키의 용도와 사용 위치를 설명합니다. 좋은 설명은 나중에 이 액세스 키를 자신있게 교체하는데 유용합니다.

최대 256자까지 가능합니다. 허용되는 문자는 문자, 숫자, UTF-8로 표현할 수 있는 공백 및 _ . : / = + - @입니다.

취소 이전 액세스 키 만들기

IAM > 사용자 > developer > 액세스 키 만들기



1단계
액세스 키 모범 사례 및 대안

2단계 - 선택 사항
설명 태그 설정

3단계
액세스 키 검색

액세스 키 검색 정보

분실하거나 잊어버린 비밀 액세스 키는 검색할 수 없습니다. 대신 새 액세스 키를 생성하고 이전 키를 비활성화합니다.

액세스 키	비밀 액세스 키
 AKIAQWGNLZYY2JYGLIEO	 ***** 표시

액세스 키 모범 사례

- 액세스 키를 일반 텍스트, 코드 리포지토리 또는 코드로 저장해서는 안 됩니다.
- 더 이상 필요 없는 경우 액세스 키를 비활성화하거나 삭제합니다.
- 최소 권한을 활성화합니다.
- 액세스 키를 정기적으로 교체합니다.

액세스 키 관리에 대한 자세한 내용은 [AWS 액세스 키 관리 모범 사례](#)를 참조하세요.

.csv 파일 다운로드 완료

Amazon ECR 구성(3)

6.

Jenkins Slack
Webhook
연동하기(2)

AWS 구성

- ECR 권한 생성
- 사용자 콘솔 액세스키 생성 : user > 계정선택 > 액세스키, 비밀 액세스키 복사
- Amazon ECR 생성

Amazon ECR > 리포지토리 > 리포지토리 생성

리포지토리 생성

일반 설정

표시 여부 설정 | 정보
리포지토리에 대한 가시성 설정을 선택합니다.

☒ 프라이빗
액세스는 IAM 및 리포지토리 정책 권한에 의해 관리됩니다.

☐ 퍼블릭
이미지 풀에 대해 공개적으로 표시되고 액세스할 수 있습니다.

리포지토리 이름
간결한 이름을 제공합니다. 개발자는 이름으로 리포지토리 콘텐츠를 식별할 수 있어야 합니다.

047675330097.dkr.ecr.ap-northeast-2.amazonaws.com/ **petclinic**

최대 256자 중 9자(최소 2자 이상) The name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, periods and forward slashes.

Amazon ECR > 리포지토리

Private Public

프라이빗 리포지토리 (1) 리포지토리 생성

리포지토리 찾기

	리포지토리 이름	URI	생성 날짜	태그 변경 불가능	스캔 빈도	암호화 유형	플스루캐시
<input type="checkbox"/>	petclinic	047675330097.dkr.ecr.ap-northeast-2.amazonaws.com/petclinic	2023년 8월 21일 20:48:57 (UTC+09)	비활성화됨	수동	AES-256	비활성

Jenkins 구성(1)

6.

Jenkins Slack
Webhook
연동하기(2)

Jenkins 구성

- Plugin 설치 : AWS Credentials, Pipeline: AWS Steps
Jenkins-Server: aws cli 설치
- Credentials 등록 : AWS 사용자 콘솔 액세스키 등록
- Jenkins 프로젝트 구성 : pipeline script 구성

Jenkins Plugins

Search: aws cre

Install **Name** **Released**

Install	Name	Released
<input checked="" type="checkbox"/>	CloudBees AWS Credentials 191.vcb_f183ce58b_9	1 yr 5 mo ago

Allows storing Amazon IAM credentials within the Jenkins Credentials API. Store Amazon IAM access keys (AWSAccessKeyId and AWSSecretKey) within the Jenkins Credentials API. Also support IAM Roles and IAM MFA Token.

Download progress

- Checking internet connectivity
- Checking update center connectivity
- Success

Oracle Java SE Development Kit Installer	성공
Command Agent Launcher	성공
Amazon Web Services SDK :: SQS	성공
Amazon Web Services SDK :: SNS	성공
Amazon Web Services SDK :: CloudFormation	성공
Amazon Web Services SDK :: Elastic Beanstalk	성공
Amazon Web Services SDK :: ECS	성공
Amazon Web Services SDK :: IAM	성공
Amazon Web Services SDK :: ECR	성공
Amazon Web Services SDK :: EFS	성공
Amazon Web Services SDK :: SSM	성공
Amazon Web Services SDK :: kinesis	성공
Amazon Web Services SDK :: Logs	성공
Amazon Web Services SDK :: CodeBuild	성공
Amazon Web Services SDK :: Secrets Manager	성공
Amazon Web Services SDK :: All	성공
Pipeline: AWS Steps	성공
Amazon Web Services SDK :: All	성공
Loading plugin extensions	성공

Please wait while Jenkins is restarting ...

Your browser will reload automatically when Jenkins is ready.

Safe Restart
Builds on agents can usually continue.

```
[ec2-user@ip-172-31-3-191 ~]$ docker start jenkins_prod
jenkins_prod
```

Jenkins 구성(2)

6.

Jenkins Slack
Webhook
연동하기(2)

Jenkins 구성

- Plugin 설치 : AWS Credentials
- Credentials 등록 : AWS 사용자 콘솔 액세스키 등록(aws-login)
- Jenkins 프로젝트 구성 : pipeline script 구성

The screenshots show the Jenkins interface for configuring credentials. The first screenshot shows the 'Credentials' page with 'System' selected. The second screenshot shows the 'System' page with 'Global credentials (unrestricted)' selected. The third screenshot shows the 'Global credentials (unrestricted)' page with the '+ Add Credentials' button highlighted. Arrows indicate the flow from the 'Credentials' page to the 'System' page, then to the 'Global credentials (unrestricted)' page, and finally to the '+ Add Credentials' button.

The screenshot shows the AWS IAM console 'Access Key' creation page. The 'Access Key' section is highlighted, showing the 'Access Key ID' (AKIAQWGNLZY2JYGLIEO) and the 'Secret Access Key' (AKIAQWGNLZY2JYGLIEO). The 'Access Key ID' and 'Secret Access Key' are highlighted with red boxes, and dashed arrows point from these boxes to the corresponding fields in the Jenkins 'New credentials' form.

The screenshot shows the Jenkins 'New credentials' form. The 'Kind' field is set to 'AWS Credentials'. The 'Scope' is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'ID' is set to 'aws-login'. The 'Description' is set to 'aws-login'. The 'Access Key ID' is set to 'AKIAQWGNLZY2JYGLIEO'. The 'Secret Access Key' is set to 'AKIAQWGNLZY2JYGLIEO'. The 'IAM Role Support' section is visible at the bottom. The 'Create' button is at the bottom right.

Jenkins 구성(3)

6.

Jenkins Slack
Webhook
연동하기(2)

Jenkins 구성

- Plugin 설치 : AWS Credentials
- Credentials 등록 : AWS 사용자 콘솔 액세스키 등록
- Jenkins 프로젝트 구성 : pipeline script 구성

우리가 배운 내용

- AWS 관리콘솔에서 ECR 접근권한/엑세스키 생성, 컨테이너 저장소(Amazon ECR)만들기
- Jenkins Plugin(AWS Credentials) 등록을 통해 Jenkins 에서 컨테이너 빌드 후 ECR에 업로드

백엔드 개발자를 위한 빌드 자동화- Jenkins

7 실전 빌드하기

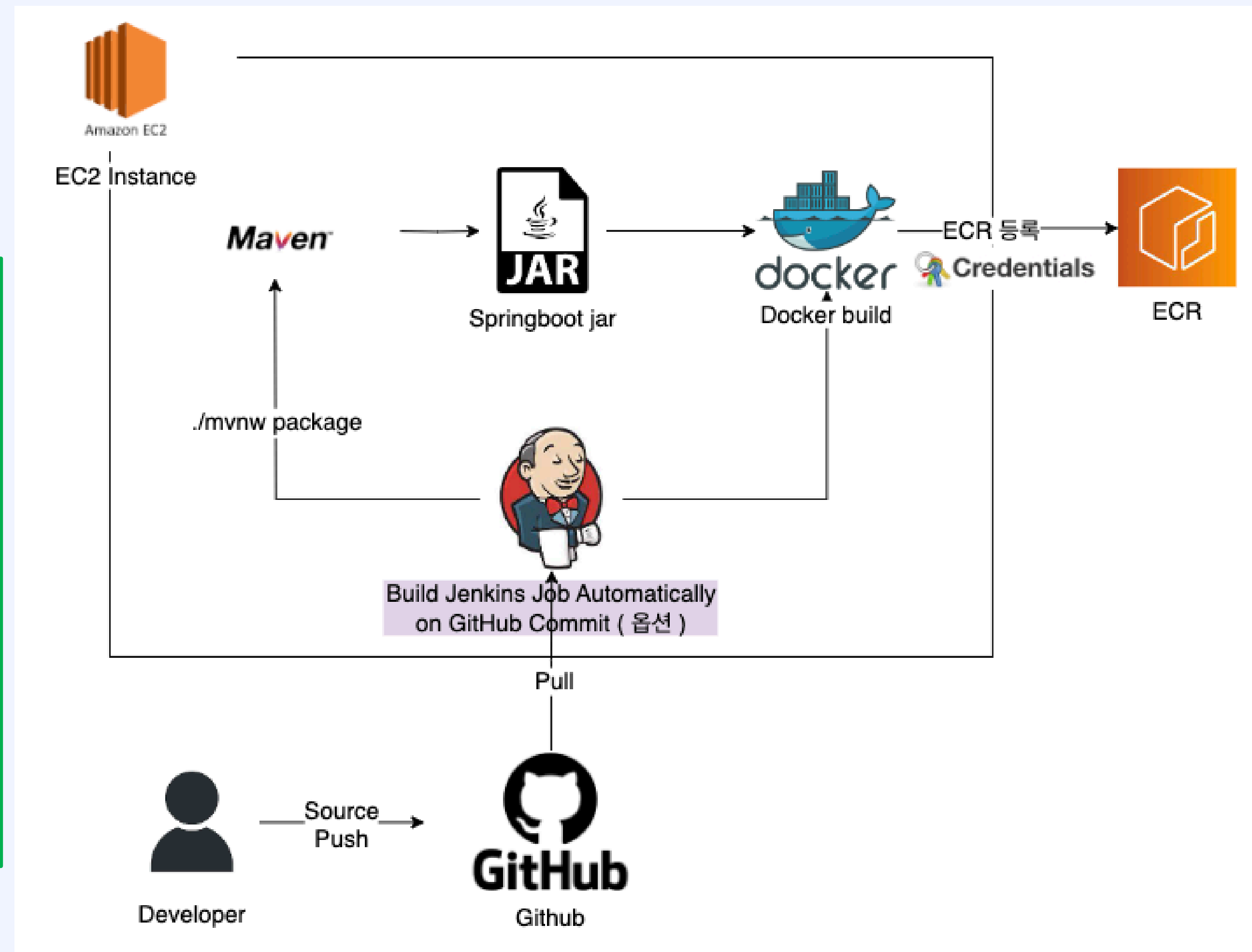
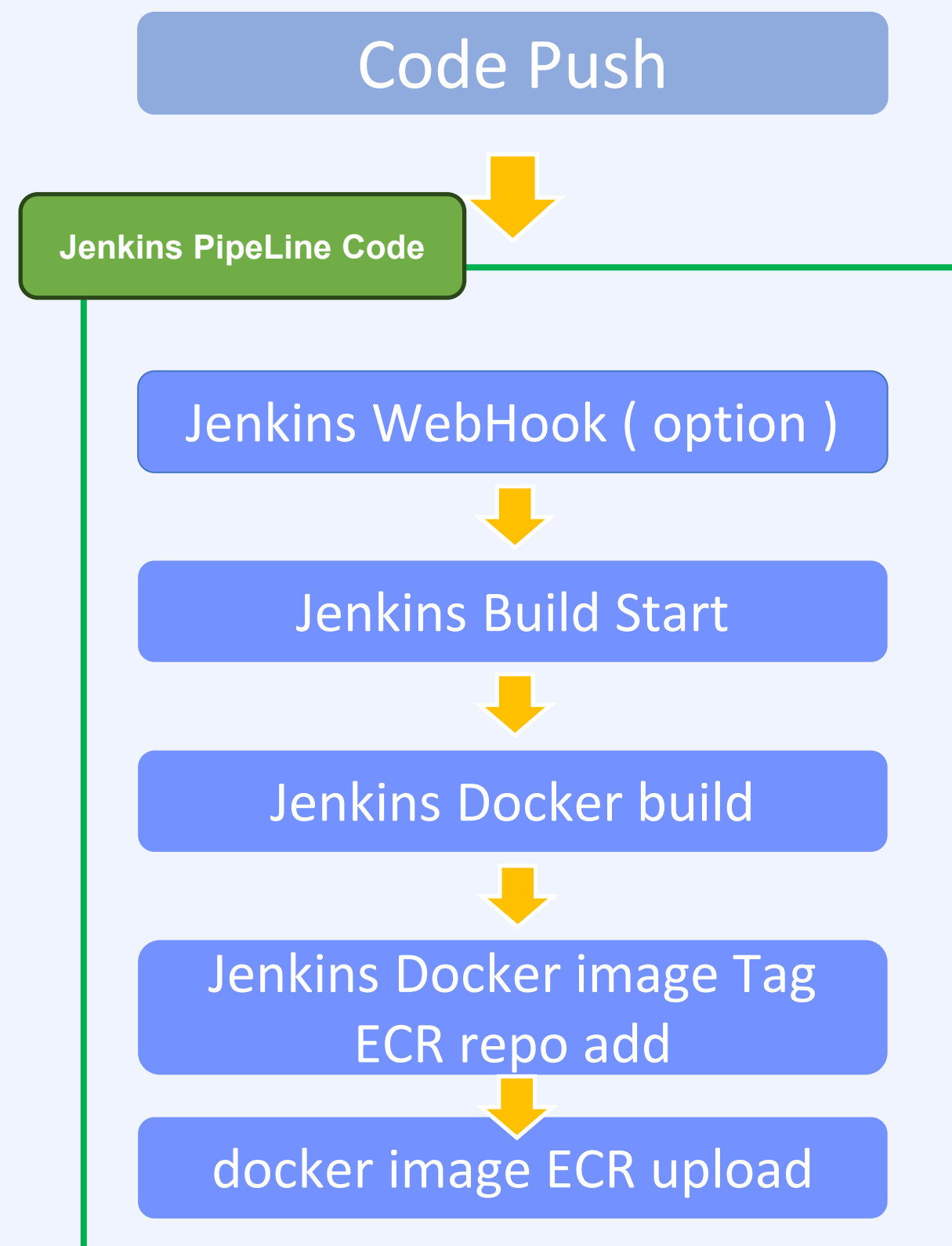
source push → container build → ecr에 업로드

실전 빌드하기: source push → container build → ecr에 업로드

7.

Jenkins Slack
Webhook 연동하기

• Jenkins Pipeline Flow



실전 빌드하기: source push → container build → ecr에 업로드

7.

Jenkins Slack
Webhook 연동하기

Jenkins Pipeline 작업순서

- iam – [Policy, Rule 생성](#) (ECR 업로드 하기 위해서 사용합니다)
 - Policy 생성
 - Rule 생성
- ECR repo 생성 - Name : spring-petclinic
- Aws Credential 등록
 - 보안자격증명 > 액세스 키 생성 > Accesskey , SecretKey 생성
 - jenkins credential 등록
- [pipeline 코드 리뷰](#) - Jenkins Pipeline [공식 가이드 문서](#)
- pipeline job 등록
- Jenkins Webhook 등록 - Auto Job Execute (선택)
 - LB 생성 – github webhook url 등록시 필요 (**주의** - public ip는 사용불가)

우리가 배운 내용

- IAM 생성 - ECR 업로드시 필요한 Rule
- aws credential 등록
- jenkins pipeline 코드 리뷰 - 키워드 withAWS , dir , slackSend

백엔드 개발자를 위한 빌드 자동화- Jenkins

8 LAB: CI 구축하기
git 코드 업로드 부터 컨테이너 빌드까지

LAB: git 코드 업로드 부터 컨테이너 빌드까지 - CI 구축하기

8.

LAB: git 코드
업로드 부터
컨테이너 빌드까지
CI 구축하기

문제 : 빌드한 젠킨스 컨테이너로 git 코드 업로드부터 컨테이너 빌드 구성

1. **Jenkins 컨테이너 이미지 빌드** : Jenkins 플러그인을 포함한 컨테이너 이미지 빌드 후 jenkins 실행
chapter-5/build 디렉토리에 있는 Dockerfile을 이용해 jenkins 컨테이너 빌드
docker build -t jenkins-plugin:2.419 .

2. **Petclinic 소스코드 clone 후 자신의 Github에 소스코드 push.** 이때 Github 로그인 토큰 생성
git clone <https://github.com/237summit/petclinic.git>

mv petclinic spring-petclinic

git push -u origin master

3. **Jenkins 통해 pet-clinic 컨테이너 이미지 빌드 후 Amazon ECR에 Push**

AWS 구성: 액세스키 생성, role생성 (ecr-full-access: AmazonEC2ContainerRegistryPowerUser),

Amazon ECR private repository : spring-petclinic

젠킨스 구성 – slack-token, aws credendtiels 등록(aws-login-token), 새로운 아이템 생성: spring-petclinic

LAB: git 코드 업로드 부터 컨테이너 빌드까지 - CI 구축하기

8.

LAB: git 코드
업로드 부터
컨테이너 빌드까지
CI 구축하기

1:00:00

우리가 배운 내용

- Jenkins를 플러그인을 포함한 컨테이너 이미지 빌드
- Jenkins 파이프라인 구성을 통해 애플리케이션 이미지 빌드 자동화

미래가 오늘이다.