# 2024.2.22. Cloud Project: AWS EC2 인스턴스 2개로 Jenkins + Spring

• Tags: #02-2024 #CI\_CD #CI\_CD/Jenkins #Backend #Backend/Spring #Cloud

### 이번 주 수업 내용 정리

- AWS EC2, IAM, key pair 개념
- AWS EC2 인스턴스에 Docker, Jenkins 설치 및 github-webhook 를 통한 CI 구성

### 오늘 실습 목표

EC2 인스턴스를 t2.2xl 및 t2.mic로 구성하여, 2xl에는 Docker+Jenkins, mic에는 미리 빌드된
 .jar 파일을 이용하여 Spring Web Server 구동

### Jenkins Plugin: Publish over SSH

• 개요

Jenkins에서 빌드 후 특정 서버로 배포 시 사용하는 플러그인 2022. 1. 12.부로 Jenkins의 보안 정책으로 인해 배포 중단됨 즉, 명시적인 설치 필요.

TODO: 보안 문제를 해결한 WORKAROUND 찾아볼 것!

### **AWS EC2 Jenkins Server Setup**

### ■ Docker + Jenkins를 위한 EC2 Instance 생성:

• Instance 타입: t2.2xlarge

• 05: Ubuntu 22.04 LTS

Tasks:

- 1. 인스턴스 생성 후 초기 셋업
- APT 업데이트/업그레이드, 스왑 메모리 구성, 재부팅
- 2. Docker 설치
- 3. Docker를 통해 Jenkins 이미지 끌어오기
- 4. Jenkins 컨테이너 실행
- 5. Jenkins 컨테이너 초기 셋업
- 6. Jenkins의 Publish over SSH 플러그인 설치

### **Spring Web Server Setup**

₩ 스프링 웹서버를 위한 EC2 Instance 생성:

• Instance 타입: t2.micro

• **0S**: Ubuntu 22.04 LTS

#### Tasks:

- 1. 인스턴스 생성 후 초기 셋업
  - APT 업데이트/업그레이드, 스왑 메모리 구성, 재부팅
- 2. 인스턴스에 openjdk-17-jdk 설치
- 3. .jar 파일을 통해 스프링 서버 실행

### 🚀 Jenkins Deployment Pipeline:

• 아래와 같은 pipeline을 구성:

```
pipeline {
    agent any
    stages {
        stage('git clone') {
            steps {
                echo 'clone start'
                git branch: 'main',
                credentialsId: '<EXAMPLE_CREDENTIAL_ID>',
                url: 'https://github.com/Mugen-
Houyou/fisa_testrep_240920.git'
                echo 'clone end'
        }
        stage('list view') {
            steps {
                echo '리스트 보기'
                sh ''' ls '''
            }
        }
        stage('build') {
            steps {
                echo 'Building Gradle project'
                sh ''' ./gradlew build '''
        }
        stage('deploy') {
            steps {
                sshPublisher(
                    publishers: [
                         sshPublisherDesc(
                             configName: 'ec2-t2mic-ubuntu',
                             transfers: [
                                 sshTransfer(
                                     cleanRemote: false,
```

```
excludes: '',
                                     execCommand: 'nohup java -jar
step06_citest-0.0.1-SNAPSHOT.jar > output.log 2>&1 &',
                                     execTimeout: 4400,
                                    flatten: false,
                                     makeEmptyDirs: false,
                                     noDefaultExcludes: false,
                                     patternSeparator: '[, ]+',
                                     remoteDirectory: '',
                                     remoteDirectorySDF: false,
                                     removePrefix: '',
                                     sourceFiles:
'build/libs/*SNAPSHOT.jar'
                                )],
                            usePromotionTimestamp: false,
                            useWorkspaceInPromotion: false,
                            verbose: true
                    )])
            }
            post {
                success {
                    echo "DONE"
            }
        }
   }
```

Jenkins에서 최종 결과 확인

# 

AWS EC2간 연동 및 배포 실습

### **Stage View**

	git clone	list view	build	deploy
Average stage times: (Average <u>full</u> run time: ~9s)	673ms	353ms	7s	2s
#27 Feb 22 No Changes	659ms	346ms	7s	741ms
#26 Feb 22 No Changes	693ms	355ms	7s	1s
#25 Feb 22 No Changes	699ms	367ms	7s	5s
#24 Feb 22 No Changes	669ms	353ms	7s	5s
#23 Feb 22 No Changes	672ms	355ms	7s	2s

## 🏋 구현 시 노트

• .jar 파일을 백그라운드로 구동하기 위한 nohup

## 🝃 참고자료

• 쉽게 설명한 nohup 과 & (백그라운드) 명령어 사용법,

https://joonyon.tistory.com/entry/%EC%89%BD%EA%B2%8C%EC%84%A4%EB%AA%85%ED%95%9C-nohup-%EA%B3%BC%EB%B0%B1%EA%B7%B8%EB%9D%BC%EC%9A%B4%EB%93%9C-

<u>%EB%AA%85%EB%A0%B9%EC%96%B4-%EC%82%AC%EC%9A%A9%EB%B2%95</u>