

포팅 문서

현재 프로젝트 사용 스택

Frontend

Backend

Build & Distribute

프로그램 실행 방법

프론트엔드 & 백엔드 실행 방법

git clone 시행

프론트엔드 실행 방법

백엔드 실행 방법

도커 실행 방법

인프라 설정

1. 서버 시간 한국 시간으로 설정

2. 패키지 업데이트

3. 패키지 업그레이드

4. 스왑 영역 할당

5. jdk, docker, docker-compose 설치

6. jenkins 설치

6-1. jenkins 플러그인 설치 및 다른 툴과 연동

7. nginx 설치 및 HTTPS 적용

8. jenkins 권한 설정

8-1. gitlab 연동

8-2. 계정 연동

8-3. 파이프라인 생성

8-4. Webhooks과 빌드 트리거 설정

9. 서버 정보 입력

10. gradle 설정

11. Tool 설정

12. Plugin 설정

13. 전체 Credentials

14. 배포 스크립트 작성

15. Docker pipeline, SSH Agent 설치

16. 빌드 진행

17. Openvidu

18. NGINX 설정

도커 파일

* docker-compose.yml

* docker-compose-front.yml

* mysql.dockerfile

* nextjs.dockerfile

* rabbitmq.dockerfile

* spring.dockerfile

시연 시나리오

현재 프로젝트 사용 스택

Frontend

- Typescript: 5
- Next.js: 14.1
- styled-components: 6.1.8
- React Three Fiber: 8.16.2
- openvidu-browser: 2.29.1
- react-query: 3.39
- stomp.js: 7.0
- sockjs-client: 1.6.1
- zustand: 4.5.2
- cannon.js: 0.6.2

Backend

- Java : openjdk-21
- Spring boot : 3.2.4
- Spring security : 6.2.3
- MySQL : 8.3
- Flask : 3.0.2
- rabbitmq : 3.13.0
- redis: 7.2
- mongodb : 5.0.0
- queryDsl : 5.0.0
- flyway : 8.0
- stomp : 2.3.4
- openvidu : 2.0.0
- openvidu-client : 2.29.1

Build & Distribute

- Jenkins : 2.450
- nginx : 1.18
- docker
- openvidu

프로그램 실행 방법

프론트엔드 & 백엔드 실행 방법

git clone 시행

```
git clone https://lab.ssafy.com/s10-final/S10P31A303.git
```

프론트엔드 실행 방법

1. S10P31A303/develop-fe/ssafy-escape로 이동합니다.
2. 아래의 명령어를 입력합니다.

```
npm install  
npm run dev
```

백엔드 실행 방법

1. S10P31A303/develop-be로 이동합니다.
2. 아래의 명령어를 입력합니다.

```
./gradlew bootrun
```

도커 실행 방법

1. S10P31A303/docker로 이동합니다.
2. 아래의 명령어를 작동합니다.

```
docker compose -f docker-compose.yml up -d
```

인프라 설정

1.서버 시간 한국 시간으로 설정

```
sudo timedatectl set-timezone Asia/Seoul
```

- 제대로 적용되었는지 확인

```
date
```

2.패키지 업데이트

```
sudo apt update
```

error 발생 : 밑의 명령어 입력 후 다시 패키지 업데이트

- certbot이 없기 때문에 에러 발생, 추후에 certbot을 발급할 것 이기 때문에 지금 삭제해도 괜찮음

```
sudo apt-add-repository --remove ppa:certbot/certbot
```

3. 패키지 업그레이드

```
sudo apt -y upgrade
```

4. 스왑 영역 할당

```
free -h
sudo fallocate -l 4G /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile
sudo echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

1. `free -h` : 시스템의 메모리 사용량과 스왑 공간에 대한 정보를 보여주는 명령어입니다. `-h` 옵션은 사람이 읽기 쉬운 형식으로 출력합니다.
2. `sudo fallocate -l 4G /swapfile` : 4GB 크기의 스왑 파일을 생성하는 명령어입니다. `fallocate` 명령어는 파일을 지정된 크기로 할당합니다.
3. `sudo chmod 600 /swapfile` : 생성된 스왑 파일에 대한 권한을 설정하는 명령어입니다. 이 경우, 소유자에 대한 읽기 및 쓰기 권한만 허용하고 그 외 권한은 모두 거부합니다.
4. `sudo mkswap /swapfile` : 스왑 파일을 스왑 공간으로 사용할 수 있도록 준비하는 명령어입니다. 스왑 파일을 포맷하고 스왑 공간으로 사용할 수 있도록 메타데이터를 설정합니다.
5. `sudo swapon /swapfile` : 생성된 스왑 파일을 활성화하는 명령어입니다. 이 명령어를 사용하면 스왑 파일이 시스템의 스왑 공간으로 사용됩니다.
6. `sudo echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab` : 스왑 파일을 부팅시에도 자동으로 마운트되도록 `/etc/fstab` 파일에 해당 정보를 추가하는 명령어입니다. `tee` 명령어를 사용하여 표준 출력과 파일로 동시에 출력합니다. 이를 통해 스왑 파일에 대한 정보가 올바르게 유지됩니다.

5. jdk, docker, docker-compose 설치

```
// jdk21
sudo apt install -y openjdk-21-jdk

// 버전 확인
java --version
```

```
// docker
sudo apt install -y apt-transport-https ca-certificates curl software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
sudo apt update
sudo apt install -y docker-ce

// curl 명령을 이용하여 docker-compose 패키지를 /usr/local/bin/docker-compose 디렉토리에 설치
sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

// chmod를 이용하여 /usr/local/bin/docker-compose 디렉토리에 대해 모든 사용자에게 실행 가능하게 설정
sudo chmod +x /usr/local/bin/docker-compose
```

6. jenkins 설치

- jenkins (패키지 매니저 방식)

1. 외부에서 접속할 포트 오픈 후 상태 확인

```
sudo ufw allow 8080
sudo ufw reload
```

// ubuntu용 jenkins 패키지 파일 다운로드

```
// wget https://pkg.jenkins.io/debian-stable/direct/jenkins_2.414.3_all.deb
wget https://mirrors.jenkins-ci.org/debian/jenkins_2.448_all.deb
```

// 패키지 명령어로 설치 진행

```
// sudo dpkg -i jenkins_2.414.3_all.deb
sudo dpkg -i jenkins_2.448_all.deb
```

// 정상 실행 확인

```
sudo systemctl status jenkins
```

// 초기 패스워드 확인

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

// 환경설정 변경위해 서비스 중지

```
sudo systemctl stop jenkins
```

2. jenkins 환경 설정 변경

// jenkins 설치 경로 이동

```
cd /var/lib/jenkins
```

// update center에 필요한 CA파일 다운로드 후 권한 변경

```
sudo mkdir update-center-rootCAs
sudo wget https://cdn.jsdelivr.net/gh/lework/jenkins-update-center/rootCA/update-center-rootCA.pem -O update-center-rootCA.pem
```

```

sudo chown -R jenkins:jenkins update-center-rootCAs

// jenkins default 설정에서 특정 미러사이트 대체하도록 아래 명령어 수행
sudo sed -i 's#https://updates.jenkins.io/update-center.json#https://raw.githi

// 위의 명령어 실행 후 URL이 'https://raw.githubusercontent.com/lework/jenkins-up
sudo cat hudson.model.UpdateCenter.xml

// jenkins 재구동
sudo systemctl restart jenkins

```

6-1. jenkins 플러그인 설치 및 다른 툴과 연동

- `http://{domain_name}:8080` 입력 후 다음과 같은 창이 나오면 아래 명령어를 통해 패스워드 확인 후 입력

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

Continue

- Install suggested plugins 클릭 후 설치

Getting Started

Installation Failures

Some plugins failed to install properly, you may retry installing them or continue without the failed plugins

✗ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✗ Credentials Binding
✗ Timestampers	✗ Workspace Cleanup	✓ Ant	✗ Gradle
✗ Pipeline	✗ GitHub Branch Source	✗ Pipeline: GitHub Groovy Libraries	✗ Pipeline: Stage View
✗ Git	✗ SSH Build Agents	✓ Matrix Authorization Strategy	✓ PAM Authentication
✓ LDAP	✗ Email Extension	✓ Mailer	

Jenkins 2.414.3 [Continue](#) [Retry](#)

- 회원가입 진행

7. nginx 설치 및 HTTPS 적용

- nginx 설치 및 상태 확인

```
// nginx 설치
sudo apt install nginx -y

// nginx 상태 확인
sudo systemctl status nginx
```

- letsencrypt, certbot 설치

```
// letsencrypt 설치
sudo apt install letsencrypt

// certbot 설치
sudo apt install certbot python3-certbot-nginx
```

- certbot - nginx 연결

```
sudo apt update

sudo certbot --nginx
// 이후 도메인 입력(여기서는 j10a305.p.ssafy.io)
// -> 2번(redirect : http -> https) 입력
```

8. jenkins 권한 설정

8-1. gitlab 연동

- 팀 깃랩에서 Settings - Access Token 생성

Project Access Tokens

Generate project access tokens scoped to this project for your applications that need access to the GitLab API. You can also use project access tokens with Git to authenticate over HTTP(S). [Learn more.](#)

Active project access tokens ⓘ 1

Add new token

Token name	Scopes	Created	Last Used ⓘ	Expires	Role	Action
cyberescape	api, read_api, create_runner, k8s_proxy, read_repository, write_repository, ai_features	Apr 30, 2024	2 weeks ago	in 1 week	Maintainer	

- Jenkins 관리 - Credentials - Add credentials 후 깃랩 정보 입력

New credentials

Kind

GitLab API token

Scope ⓘ

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

API token

.....

ID ⓘ

gitlab-apiToken

Description ⓘ

Create

- Jenkins 관리 - System 이동 후 깃랩 정보 입력 후 Test Connection - Apply

GitLab

☒ Enable authentication for '/project' end-point ?

GitLab connections

Connection name ?

A name for the connection

cyberescape

GitLab host URL ?

The complete URL to the GitLab server (e.g. http://gitlab.mydomain.com)

https://lab.ssafy.com

Credentials ?

API Token for accessing GitLab

GitLab API token

+ Add

고급

Test Connection

8-2. 계정 연동

- 나의 깃랩 계정에서 Access Token 생성

Personal Access Tokens

You can generate a personal access token for each application you use that needs access to the GitLab API. You can also use personal access tokens to authenticate against Git over HTTP. They are the only accepted password when you have Two-Factor Authentication (2FA) enabled.

Active personal access tokens 2						Add new token
Token name	Scopes	Created	Last Used ?	Expires	Action	
bjwtoken	api, read_api, read_user, create_runner, k8s_proxy, read_repository, write_repository, ai_features	Apr 30, 2024	Never	in 1 week		
bjwtoken2	api, read_api, read_user, create_runner, k8s_proxy, read_repository, write_repository, ai_features	May 03, 2024	1 hour ago	in 1 week		

- Jenkins 관리 - Credentials - Add credentials 후 계정 정보 입력

Scope ?

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

Username ?

wjddn0308

☐ Treat username as secret ?

Password ?

 Concealed

Change Password

ID ?

baejeub


8-3. 파이프라인 생성

- Dashboard - +새로운 Item - 이름 입력 후 Pipeline 선택 후 OK(생성)


Enter an item name

cyberescape-


» Required field

**Freestyle project**


Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**Pipeline**

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**

다양한 환경에서의 테스트, 플래폼 특성 빌드, 기타 등등 처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.

**Folder**

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

8-4. Webhooks과 빌드 트리거 설정

- 파이프라인 이동 - 구성 - General - Build Triggers

Build Triggers

☐ Build after other projects are built ?

☐ Build periodically ?

☒ Build when a change is pushed to GitLab. GitLab webhook URL: <http://j10a305.p.ssafy.io:8080/project/travelmaker> ?

Enabled GitLab triggers

☐ Push Events ?

If enabled, then the build will be triggered when new commits are pushed to the GitLab repository. (from [GitLab Plugin](#))

☐ Push Events in case of branch delete ?

☐ Opened Merge Request Events ?

☐ Build only if new commits were pushed to Merge Request ?

☒ Accepted Merge Request Events ?

If enabled, then the build will be triggered when a merge request is accepted in the GitLab repository. (from [GitLab Plugin](#))

☐ Closed Merge Request Events ?

다음과 같이 Accepted Merge Request Events를 클릭 후 하단으로 내려서 고급 버튼 클릭 후 Secret Token 발급 받기

☐ Filter branches by regex ?

☐ Filter merge request by label

Secret token ?

54c6799c2955f50c97e49fba4f026938

Generate

Clear

- 팀 깃랩에서 Settings - Webhooks - Add new webhook

Webhooks

Webhooks enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

Project Hooks 2 Add new webhook

http://k10a303.p.ssafy.io:8080/project/cyberescapeback	Test ▾	Edit	Delete
Push events SSL Verification: enabled			
http://k10a303.p.ssafy.io:8080/project/cyberescapefront	Test ▾	Edit	Delete
Push events SSL Verification: enabled			

URL : jenkins 에 접근하는 url(프로젝트 주소/파이프 라인 이름)

Secret token : jenkins에서 발급해준 Build Trigger Secret Token

Regular expression : 실제로 배포할 branch 선택

9. 서버 정보 입력

- Jenkins 관리 - Credentials - Add credentials 후 서버 정보 입력

New credentials

Kind

SSH Username with private key

Scope ?

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

ID ?

ubuntu-a305

Description ?

Username

ubuntu

Private Key

☒ Enter directly

Key

```
zMH3EwKBgCm1+a/R+Y9p+RhgUJAuUcdY9YzrJwaST98/xwabXrMJppUkq8utNGs9
/JWw1YJ4g117bfhb4ByEJ3qN6IU6OMAL1rbX7qF9wnU+g1qJH1RX1M4CUT5XPrRF
BuLE1K+NogxaLtPEDeErK9JQru8d6gN0wuYskuU0mmvUo+51uP8Q
-----END RSA PRIVATE KEY-----
```

Passphrase

Create

AWS *.pem 키의 내용 주석까지 입력

10. gradle 설정

- jenkins 관리 - Tools - Gradle installations

Gradle installations

Add Gradle

≡ Gradle

name ?

gradle8.5

☒ Install automatically ?

≡ Install from Gradle.org

Version

Gradle 8.5

Add Installer ▼

Save

Apply

11. Tool 설정

- node.js

Name

nodejs

☒ Install automatically ?

≡ Install from nodejs.org

Version

NodeJS 20.11.1

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

☐ Force 32bit architecture

Global npm packages to install

Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax `packageName@version`

Global npm packages refresh hours

Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

72

- docker

≡ Docker

Name

docker

☒ Install automatically ?

≡ Download from docker.com

Docker version ?

latest

















Add Installer ▾

12. Plugin설정

<input type="checkbox"/>	이름 ↓	Released	설치됨
<input type="checkbox"/>	Branch API 2.1169.va_f810c56e895 Library plugins (for use by other plugins) This plugin provides an API for multiple branch based projects.	17 days ago	2.1163.va_f1064e4a_a_f3
<input type="checkbox"/>	Credentials Binding 677.vdc9d38cb_254d Build Wrappers credentials Allows credentials to be bound to environment variables for use from miscellaneous build steps.	19 days ago	657.v2b_19db_7d6e6d
<input type="checkbox"/>	Display URL API 2.204.vf6fddd8a_8b_e9 User Interface Library plugins (for use by other plugins) Provides the DisplayURLProvider extension point to provide alternate URLs for use in notifications.	19 days ago	2.200.vb_9327d658781
<input type="checkbox"/>	Docker 1.6.1 Cloud Providers Cluster Management docker This plugin integrates Jenkins with Docker	6 days 8 hr ago	1.6
<input type="checkbox"/>	Email Extension 1814.v404722f34263 Build Tools Build Notifiers email This plugin is a replacement for Jenkins's email publisher. It allows to configure every aspect of email notifications: when an email is sent, who should receive it and what the email says	1 day 12 hr ago	1806.v856a_01a_fa_39a_
<input type="checkbox"/>	Git 5.2.2 git Source Code Management This plugin integrates Git with Jenkins.	11 days ago	5.2.1
<input type="checkbox"/>	GitHub 1.39.0 External Site/Tool Integrations github This plugin integrates GitHub to Jenkins.	2 days 15 hr ago	1.38.0
<input type="checkbox"/>	GitHub Branch Source 1789.v5b_0c0cea_18c3 pipeline github Source Code Management Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.	16 days ago	1787.v8b_8cd49a_f8f1
<input type="checkbox"/>	GitLab 1.8.1 Build Triggers This plugin allows GitLab to trigger Jenkins builds and display their results in the GitLab UI.	7 days 5 hr ago	1.8.0
<input type="checkbox"/>	Gradle 2.12 Build Tools This plugin allows Jenkins to invoke Gradle build scripts directly.	4 days 1 hr ago	2.11
<input type="checkbox"/>	Icons API 74.v93d5eb_813d5f User Interface Provides Icons for Jenkins Plugins, internally known as "symbols". Check out the design-library how to use icons in your plugin.	6 days 13 hr ago	70.v2959a_b_74e3cf
<input type="checkbox"/>	Pipeline: API 1311.v4250456a_e552 Library plugins (for use by other plugins) Miscellaneous Plugin that defines Pipeline API.	2 days 10 hr ago	1291.v51fd2a_625da_7
<input type="checkbox"/>	Pipeline: Groovy 3894.3896.vca_2c931e7935 pipeline Miscellaneous Pipeline execution engine based on continuation passing style transformation of Groovy scripts.	1 mo 2 days ago	3894.vd0f0248b_a_fc4
<input type="checkbox"/>	Pipeline: Groovy Libraries 710.v4b_94b_077a_808 pipeline Library plugins (for use by other plugins) Libraries for Pipeline scripts allowing logic to be shared across jobs.	1 day 11 hr ago	704.vc58b_8890a_384
<input type="checkbox"/>	Pipeline: Nodes and Processes 1353.v1891a_b_01da_18 pipeline Miscellaneous Pipeline steps locking agents and workspaces, and running external processes that may survive a Jenkins restart or agent reconnection.	3 days 17 hr ago	1336.v768003e07199
<input type="checkbox"/>	Pipeline: Supporting APIs 907.v6713a_ed8a_573 Library plugins (for use by other plugins) Miscellaneous Common utility implementations to build Pipeline Plugin	17 days ago	896.v175a_a_9c5b_78f
<input type="checkbox"/>	Plain Credentials 182.v468b_97b_9dcb_8 Library plugins (for use by other plugins) Allows use of plain strings and files as credentials.	4 days 19 hr ago	179.vc5cb_98f6db_38
<input type="checkbox"/>	Script Security 1336.vf33a_a_9863911 Security Library plugins (for use by other plugins) Allows Jenkins administrators to control what in-process scripts can be run by less-privileged users. Applying this update will address security vulnerabilities in the currently installed version.	1 mo 2 days ago	1335.vf07d9ce377a_e
<input type="checkbox"/>	Timestamp 1.27 Build Wrappers Adds timestamps to the Console Output	5 days 14 hr ago	1.26

13. 전체 Credentials

Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	gitlab-apiToken	GitLab API token
		System	(global)	7SCCuQ13BMQgr73Pnzsm	wjddn0308@naver.com/*****
		System	(global)	docker-hub	baejeub/*****
		System	(global)	baejeub	wjddn0308/*****
		System	(global)	config	config-dev.yml
		System	(global)	docker-env	.env
		System	(global)	ubuntu-a303	ubuntu
		System	(global)	front-env	.env

14. 배포 스크립트 작성

- 백엔드 파이프라인

```
pipeline {
    agent any

    tools {
        gradle 'gradle8.5'
    }

    environment {
        backendImageName = "baejeub/cyberescape"
        registryCredential = 'docker-hub'
        releaseServerAccount = 'ubuntu'
        releaseServerUri = 'k10a303.p.ssafy.io'
        projectPath = '/var/lib/jenkins/workspace/cyberescape'
        releasePort = 3000
    }

    stages {
        stage('Git Clone') {
            steps {
                git branch: 'release',
                    credentialsId: 'baejeub',
                    url: 'https://lab.ssafy.com/s10-final/S10P31A303'
            }
        }
        stage('BE-SPRING-CONFIG'){
            steps{
```

```

        withCredentials([file(credentialsId: 'docker-env', variable:
            sshagent(credentials: ['ubuntu-a303']) {
                sh 'scp -o StrictHostKeyChecking=no -r ${dotenv} $releaseServerAccount@$releaseServerIp'
            }
        ])
    }
    withCredentials([file(credentialsId: 'config', variable: 'config') {
        sshagent(credentials: ['ubuntu-a303']) {
            sh 'scp -o StrictHostKeyChecking=no -r ${config} $releaseServerAccount@$releaseServerIp'
        }
    })
}

stage('FE-NEXTJS-CONFIG'){
    steps{
        withCredentials([file(credentialsId: 'front-env', variable:
            sshagent(credentials: ['ubuntu-a303']) {
                sh 'scp -o StrictHostKeyChecking=no -r ${fe} $releaseServerAccount@$releaseServerIp'
            }
        ])
    }
}

stage('Jar Build') {
    steps {
        dir ('escape-be') {
            sh 'gradle clean bootjar'
        }
    }
}

// stage('Backend Image Build & DockerHub Push') {
//     steps {
//         dir('escape-be') {
//             script {
//                 docker.withRegistry('', registryCredential) {
//                     sh "docker buildx build --platform linux/amd64"
//                     sh "docker buildx build --platform linux/amd64"
//                 }
//             }
//         }
//     }
// }

stage('Before Service Stop') {
    steps {
        script {
            sshagent(credentials: ['ubuntu-a303']) {
                sh '''
                ssh -o StrictHostKeyChecking=no $releaseServerAccount@$releaseServerIp
                '''
            }
        }
    }
}

```

```
}  
}  
}  
stage('Service Start') {  
    steps {  
        script {  
            sshagent(credentials: ['ubuntu-a303']) {  
                sh '''  
                    ssh -o StrictHostKeyChecking=no $releaseServerAcc  
                '''  
            }  
        }  
    }  
}  
}
```

- 프론트 파이프라인

```

pipeline {
    agent any

    tools {
        gradle 'gradle8.5'
    }

    environment {
        backendImageName = "baejeub/cyberescape"
        registryCredential = 'docker-hub'
        releaseServerAccount = 'ubuntu'
        releaseServerUri = 'k10a303.p.ssafy.io'
        projectPath = '/var/lib/jenkins/workspace/cyberescape-front'
        releasePort = 3000
    }

    stages {
        stage('Git Clone') {
            steps {
                git branch: 'release',
                    credentialsId: 'baejeub',
                    url: 'https://lab.ssafy.com/s10-final/S10P31A303'
            }
        }
        stage('FE-NEXTJS-CONFIG'){
            steps{
                withCredentials([file(credentialsId: 'front-env', variable:
                    sshagent(credentials: ['ubuntu-a303'])) {
                    sh 'scp -o StrictHostKeyChecking=no -r ${fe} $release

```

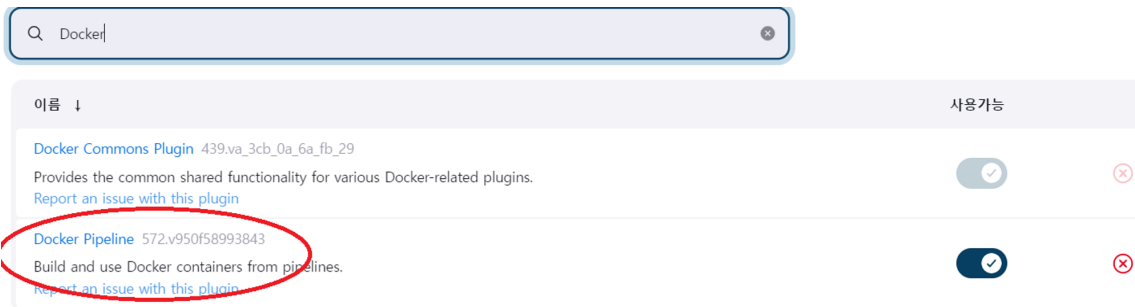
```

    }
  }
}
stage('Jar Build') {
  steps {
    dir ('escape-be') {
      sh 'gradle clean bootjar'
    }
  }
}
stage('Before Service Stop') {
  steps {
    script {
      sshagent(credentials: ['ubuntu-a303']) {
        sh '''
        ssh -o StrictHostKeyChecking=no $releaseServerAccount@$releaseServerIp
        '''
      }
    }
  }
}
stage('Service Start') {
  steps {
    script {
      sshagent(credentials: ['ubuntu-a303']) {
        sh '''
        ssh -o StrictHostKeyChecking=no $releaseServerAccount@$releaseServerIp
        '''
      }
    }
  }
}
stage('Copy Ingame Resource') {
  steps {
    script {
      sshagent(credentials: ['ubuntu-a303']) {
        sh '''
        ssh -o StrictHostKeyChecking=no $releaseServerAccount@$releaseServerIp
        '''
      }
    }
  }
}
}

```

15. Docker pipeline, SSH Agent 설치

- Jenkins 관리 - Plugins - Docker Pipeline, SSH Agent 설치



16. 빌드 진행

다음과 같은 에러가 뜨면 밑의 명령어를 터미널에 입력해서 해결

```
ERROR: failed to initialize builder mybuilder (mybuilder0): permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/containers/buildx_buildkit_mybuilder0/json ": dial unix /var/run/docker.sock: connect: permission denied
```

```
// /var/run/docker.sock 파일의 권한을 666으로 변경하여 그룹 내 다른 사용자도 접근 가능  
sudo chmod 666 /var/run/docker.sock
```

17. Openvidu

- OpenVidu를 배포하려면 루트 권한이 필요

```
sudo su
```

- OpenVidu 설치에 권장되는 폴더

```
cd /opt
```

- 이제 다음 명령을 실행하여 설치 스크립트를 다운로드하고 실행

```
curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_lat
```

```

=====
Openvidu Platform successfully installed.
=====

1. Go to openvidu folder:
$ cd openvidu

2. Configure DOMAIN_OR_PUBLIC_IP and OPENVIDU_SECRET in .env file:
$ nano .env

3. Start OpenVidu
$ ./openvidu start

For more information, check:
https://docs.openvidu.io/en/stable/deployment/ce/on-premises/

```

- openvidu .env

```

# OpenVidu configuration
# -----
# Documentation: https://docs.openvidu.io/en/stable/reference-docs/openvidu-config/

# NOTE: This file doesn't need to quote assignment values, like most shells do.
# All values are stored as-is, even if they contain spaces, so don't quote them.

# Domain name. If you do not have one, the public IP of the machine.
# For example: 198.51.100.1, or openvidu.example.com
DOMAIN_OR_PUBLIC_IP=k10a303.p.ssafy.io

# OpenVidu SECRET used for apps to connect to OpenVidu server and users to access to OpenVidu Dashboard
OPENVIDU_SECRET=my_secret_key

# Certificate type:
# - selfsigned: Self signed certificate. Not recommended for production use.
#               Users will see an ERROR when connected to web page.
# - owncert:    Valid certificate purchased in a Internet services company.
#               Please put the certificates files inside folder ./owncert
#               with names certificate.key and certificate.cert
# - letsencrypt: Generate a new certificate using letsencrypt. Please set the
#               required contact email for Let's Encrypt in LETSENCRYPT_EMAIL
#               variable.
CERTIFICATE_TYPE=letsencrypt

# If CERTIFICATE_TYPE=letsencrypt, you need to configure a valid email for notifications
LETSENCRYPT_EMAIL=wjddn0308@naver.com

# Proxy configuration
# If you want to change the ports on which openvidu listens, uncomment the following lines

# Allows any request to http://DOMAIN_OR_PUBLIC_IP:HTTP_PORT/ to be automatically
# redirected to https://DOMAIN_OR_PUBLIC_IP:HTTPS_PORT/.
# WARNING: the default port 80 cannot be changed during the first boot
# if you have chosen to deploy with the option CERTIFICATE_TYPE=letsencrypt
# HTTP_PORT=80

# Changes the port of all services exposed by OpenVidu.
# SDKs, REST clients and browsers will have to connect to this port
# HTTPS_PORT=443

```

- openvidu docker-compose.yml

- openvidu의 nginx와 내가 설치한 nginx의 충돌로 인해 openvidu nginx 제거

```
services:

  openvidu-server:
    image: openvidu/openvidu-server:2.29.0
    container_name: openvidu-server
    restart: on-failure
    network_mode: host
    entrypoint: ['/usr/local/bin/entrypoint.sh']
    volumes:
      - ./coturn:/run/secrets/coturn
      - /var/run/docker.sock:/var/run/docker.sock
      - ${OPENVIDU_RECORDING_PATH}:${OPENVIDU_RECORDING_PATH}
      - ${OPENVIDU_RECORDING_CUSTOM_LAYOUT}:${OPENVIDU_RECORDING_CUSTOM_LAYOUT}
      - ${OPENVIDU_CDR_PATH}:${OPENVIDU_CDR_PATH}
      - /opt/openvidu/.env:/opt/openvidu/.env
    environment:
      - SERVER_SSL_ENABLED=false
      - SERVER_PORT=5443
      - KMS_URI=[ "ws://localhost:8888/kurento" ]
      - COTURN_IP=${COTURN_IP:-auto-ipv4}
      - COTURN_PORT=${COTURN_PORT:-3478}
    ports:
      - "5443:5443"
    logging:
      options:
        max-size: "${DOCKER_LOGS_MAX_SIZE:-100M}"

  kms:
    image: ${KMS_IMAGE:-kurento/kurento-media-server:7.0.1}
    restart: always
    network_mode: host
    ulimits:
      core: -1
    volumes:
      - /opt/openvidu/kms-crashes:/opt/openvidu/kms-crashes
      - ${OPENVIDU_RECORDING_PATH}:${OPENVIDU_RECORDING_PATH}
      - /opt/openvidu/kurento-logs:/opt/openvidu/kurento-logs
    environment:
      - KMS_MIN_PORT=40000
      - KMS_MAX_PORT=57000
      - GST_DEBUG=${KMS_DOCKER_ENV_GST_DEBUG:-}
      - KURENTO_LOG_FILE_SIZE=${KMS_DOCKER_ENV_KURENTO_LOG_FILE_SIZE:-}
      - KURENTO_LOGS_PATH=/opt/openvidu/kurento-logs
    logging:
      options:
        max-size: "${DOCKER_LOGS_MAX_SIZE:-100M}"
```

```

coturn:
  image: openvidu/openvidu-coturn:2.29.0
  restart: on-failure
  ports:
    - "${COTURN_PORT:-3478}:${COTURN_PORT:-3478}/tcp"
    - "${COTURN_PORT:-3478}:${COTURN_PORT:-3478}/udp"
  env_file:
    - .env
  volumes:
    - ./coturn:/run/secrets/coturn
  command:
    - --log-file=stdout
    - --listening-port=${COTURN_PORT:-3478}
    - --fingerprint
    - --min-port=${COTURN_MIN_PORT:-57001}
    - --max-port=${COTURN_MAX_PORT:-65535}
    - --realm=openvidu
    - --verbose
    - --use-auth-secret
    - --static-auth-secret=${COTURN_SHARED_SECRET_KEY}
  logging:
    options:
      max-size: "${DOCKER_LOGS_MAX_SIZE:-100M}"

```

- openvidu 시작하기

```

cd /opt/openvidu
./openvidu restart

```

```

-----
OpenVidu is ready!
-----

```

```

* OpenVidu Server URL: https://k10a303.p.ssafy.io/

```

```

* OpenVidu Dashboard: https://k10a303.p.ssafy.io/dashboard
-----

```

18.NGINX 설정

- nginx 설정 변경


```
sudo vi /etc/nginx/sites-available/default
```

```
# include snippets/snakeoil.conf;

root /var/www/html;

# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;

server_name _;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    try_files $uri $uri/ =404;
}

# pass PHP scripts to FastCGI server
#
#location ~ \.php$ {
#    include snippets/fastcgi-php.conf;
#
#    # With php-fpm (or other unix sockets):
#    fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
#    # With php-cgi (or other tcp sockets):
#    fastcgi_pass 127.0.0.1:9000;
#}

# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
#location ~ /\.ht {
#    deny all;
#}

}

# Virtual Host configuration for example.com
#
# You can move that to a different file under sites-available/ and symlink it
# to sites-enabled/ to enable it.
#
#server {
#    listen 80;
#    listen [::]:80;
#}
```

```

#       server_name example.com;
#
#       root /var/www/example.com;
#       index index.html;
#
#       location / {
#           try_files $uri $uri/ =404;
#       }
#}

server {

    # SSL configuration
    #
    # listen 443 ssl default_server;
    # listen [::]:443 ssl default_server;
    #
    # Note: You should disable gzip for SSL traffic.
    # See: https://bugs.debian.org/773332
    #
    # Read up on ssl_ciphers to ensure a secure configuration.
    # See: https://bugs.debian.org/765782
    #
    # Self signed certs generated by the ssl-cert package
    # Don't use them in a production server!
    #
    # include snippets/snakeoil.conf;

    # root /var/www/html;

    # Add index.php to the list if you are using PHP
    # index index.html index.htm index.nginx-debian.html;
    server_name k10a303.p.ssafy.io; # managed by Certbot


    # pass PHP scripts to FastCGI server
    #
    #location ~ \.php$ {
    #    include snippets/fastcgi-php.conf;
    #
    #    # With php-fpm (or other unix sockets):
    #    fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
    #    # With php-cgi (or other tcp sockets):
    #    fastcgi_pass 127.0.0.1:9000;
    #}

    # deny access to .htaccess files, if Apache's document root

```

```

# concurs with nginx's one
#
#location ~ /\.ht {
#    deny all;
#}
#
location / {
    proxy_pass http://localhost:3000;

}

location /api {
    proxy_pass http://localhost:8081;
    rewrite ^/api(.*)$ $1?sargs break;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "Upgrade";
    proxy_set_header Host $host;
    proxy_read_timeout 216000;
    proxy_buffering off;
    chunked_transfer_encoding off;
}

location /images{
    proxy_ssl_server_name on;
    proxy_pass http://localhost:88;
    add_header 'Access-Control-Allow-Credentials' 'true';
    add_header 'Access-Control-Allow-Origin' 'http://localhost:3000';
    client_max_body_size 200M;
}

# WebSocket 경로 설정
# location /ws-stomp {
#     proxy_pass http://localhost:8081; # WebSocket을 처리할 백엔드 서버
#     proxy_http_version 1.1;
#     proxy_set_header Upgrade $http_upgrade;
#     proxy_set_header Connection "Upgrade";
#     proxy_set_header Host $host;

#     ssl_certificate /etc/letsencrypt/live/k10a303.p.ssafy.io/fullchain.pem;
#     ssl_certificate_key /etc/letsencrypt/live/k10a303.p.ssafy.io/private.key;
#     proxy_ssl_session_reuse on; # SSL 세션 재사용 설정

#     클라이언트의 실제 IP 주소를 백엔드 서버에 전달
#     proxy_set_header X-Real-IP $remote_addr;
#     proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
# }

```

```

        location /openvidu {
            proxy_pass http://k10a303.p.ssafy.io:5443;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection "upgrade";
        }

listen [::]:443 ssl ipv6only=on; # managed by Certbot
listen 443 ssl; # managed by Certbot
ssl_certificate /etc/letsencrypt/live/k10a303.p.ssafy.io/fullchain.pem; #
ssl_certificate_key /etc/letsencrypt/live/k10a303.p.ssafy.io/privkey.pem; #
include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

}
server {
    if ($host = k10a303.p.ssafy.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80 ;
    listen [::]:80 ;
    server_name k10a303.p.ssafy.io;
    return 404; # managed by Certbot

}

```

- nginx 다시 시작

```
sudo systemctl restart nginx
```

- nginx 에러 로그 확인

```
sudo tail -200f /var/log/nginx/error.log
```

- nginx 상태 확인

```
sudo systemctl status nginx
```

```
ubuntu@ip-172-26-7-170:/opt/openvidu$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2024-05-14 16:33:13 KST; 5 days ago
     Docs: man:nginx(8)
   Process: 3674001 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Process: 3674002 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 3674004 (nginx)
      Tasks: 5 (limit: 19165)
     Memory: 94.1M
    CGroup: /system.slice/nginx.service
            └─ 249886 nginx: worker process
               249887 nginx: worker process
               249888 nginx: worker process
               249889 nginx: worker process
               3674004 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;

May 14 16:33:13 ip-172-26-7-170 systemd[1]: Starting A high performance web server and a reverse proxy server...
May 14 16:33:13 ip-172-26-7-170 systemd[1]: Started A high performance web server and a reverse proxy server.
```

- 이미지 파일 보기

```
docker ps -a
```

- 실행되는 컨테이너 로그 보기

```
docker logs 컨테이너 이름
```

- 컨테이너 들어가기

```
docker exec - it 컨테이너 이름 /bin/bash
```

- mysql 한국어 지원 설정

```
SET NAMES utf8mb4 COLLATE utf8mb4_unicode_ci;
```

- rabbitmq plugin 추가 설정

```
rabbitmq-plugins enable rabbitmq_stomp && \
  rabbitmq-plugins enable rabbitmq_web_stomp && \
  rabbitmq-plugins enable rabbitmq_web_stomp_examples && \
  rabbitmq-plugins enable rabbitmq_mqtt && \
  rabbitmq-plugins enable rabbitmq_web_mqtt && \
  rabbitmq-plugins enable rabbitmq_web_mqtt_examples && \
  rabbitmq-plugins enable rabbitmq_management && \
  rabbitmq-plugins enable rabbitmq_federation
```

도커 파일

※ docker-compose.yml

```
services:
  spring:
    container_name: spring
    hostname: spring
    # 실행되는 컨텍스트 및 도커파일 지정
```

```

build:
  context: ../
  dockerfile: "../docker/dockerfile/spring.dockerfile"
ports:
  - "8081:8080"
restart: "always"
networks:
  - backend

mysql:
  container_name: mysql
  hostname: mysql
  # image : mysql
  build:
    context: ../
    dockerfile: "../docker/dockerfile/mysql.dockerfile"
  networks:
    - backend
  volumes:
    - /home/ubuntu/volume/mysql:/var/lib/mysql
  env_file: ".env"
  ports:
    - "3307:3306"

redis:
  container_name: redis
  image: redis
  networks:
    - backend
  ports:
    - "6379:6379"

rabbitmq:
  container_name: rabbitmq
  image: rabbitmq
  build:
    context: ../
    dockerfile: "../docker/dockerfile/rabbitmq.dockerfile"
  env_file: ".env"
  networks:
    - backend
  volumes:
    - /home/ubuntu/volume/mq:/var/lib/rabbitmq
  ports:
    - "5672:5672"
    - "61613:61613"
    - "15672:15672"

```

```

mongo:
  container_name: mongo
  image: mongo
  networks:
    - backend
  volumes:
    - /home/ubuntu/volume/data:/data
  env_file: ".env"
  ports:
    - "27017:27017"
mongo-express:
  image: mongo-express
  container_name: mongo-express
  restart: always
  networks:
    - backend
  ports:
    - "8082:8081"
  env_file: ".env"
  depends_on:
    - mongo

nginx:
  container_name: nginx-file
  image : nginx
  volumes:
    - ./conf/nginx.conf:/etc/nginx/nginx.conf
    - /home/ubuntu/volume/game/public:/home/images
  networks:
    - backend
  ports:
    - "88:80"

```

브리지 네트워크는 도커가 제공하는 네트워크 드라이버 중 하나로, 가상
인터페이스를 통해 컨테이너들을 연결한다.

```

networks:
  backend:
    ipam:
      driver: default
      config:
        - subnet: "172.16.20.0/24"
          gateway: "172.16.20.1"
  frontend:
    ipam:
      driver: default
      config:

```

```
- subnet: "172.16.21.0/24"
  gateway: "172.16.21.1"
```

※ docker-compose-front.yml

```
version: "3"

services:
  nextjs:
    container_name: nextjs
    build:
      context: ../
      dockerfile: "../docker/dockerfile/nextjs.dockerfile"
    ports:
      - "3000:3000"
```

※ mysql.dockerfile

```
FROM mysql:8.3

ADD ../docker/init_db /docker-entrypoint-initdb.d

RUN chmod 775 /docker-entrypoint-initdb.d/init.sql

ENV MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
ENV MYSQL_USER=${MYSQL_USER}
ENV MYSQL_PASSWORD=${MYSQL_PASSWORD}

EXPOSE 3307
```

※ nextjs.dockerfile

```
# Node.js 20 버전의 Alpine 이미지 사용
FROM node:20-alpine

# root에 /app 폴더 생성
RUN mkdir /app

# work dir 고정
WORKDIR /app

# Next.js 빌드 결과물과 정적 파일을 /app에 복사
COPY ../escape-fe/ssafy-escape/* .
```



```
# npm install 실행
RUN npm install

# Next.js 애플리케이션 빌드
RUN npm run build

RUN export PATH=$PATH:/home/root/app

# 애플리케이션 실행
CMD ["npm", "start"]
```

※ rabbitmq.dockerfile

```
FROM rabbitmq

ENV HOSTNAME=${RABBITMQ_HOST}

EXPOSE 5672 15672 61613

ENV RABBITMQ_DEFAULT_USER=${RABBITMQ_USERNAME}
ENV RABBITMQ_DEFAULT_PASS=${RABBITMQ_PASSWORD}
# 필요한 RabbitMQ 플러그인 활성화

RUN rabbitmq-plugins enable rabbitmq_stomp && \
    rabbitmq-plugins enable rabbitmq_web_stomp && \
    rabbitmq-plugins enable rabbitmq_web_stomp_examples && \
    rabbitmq-plugins enable rabbitmq_mqtt && \
    rabbitmq-plugins enable rabbitmq_web_mqtt && \
    rabbitmq-plugins enable rabbitmq_web_mqtt_examples && \
    rabbitmq-plugins enable rabbitmq_management && \
    rabbitmq-plugins enable rabbitmq_federation

CMD ["rabbitmq-server"]
```

※ spring.dockerfile

```
FROM openjdk:21
ENV TZ Asia/Seoul
ENV APP_HOME=/app
WORKDIR $APP_HOME

COPY escape-be/build/libs/escape.jar .

COPY docker/wait-for-it.sh /wait-for-it.sh

RUN chmod +x /wait-for-it.sh
```

```
# 컨테이너를 구동할 때 실행할 명령어 지정 (명령어를 스페이스로 나눈 것과 같다)
ENTRYPOINT ["/wait-for-it.sh", "mysql:3306", "--", "java", "-jar", "escape.jar"]
```

시연 시나리오

1. 사용자는 회원가입을 할 수 있다.
2. 사용자는 로그인을 할 수 있다.
3. 사용자는 랜덤 생성된 닉네임을 변경할 수 있다.
4. 사용자는 기본 프로필 이미지를 변경할 수 있다.
5. 친구 추가 알림을 받고 친구 추가 요청을 수락/거절 할 수 있다.
6. 친구 추가 요청을 보낼 수 있다.
7. 하단 게임 설명을 통해 게임 플레이에 대한 정보를 확인한다.
8. 싱글 모드 우주 테마를 플레이 한다.
9. 게임 종료 후 우주 테마 내 최고 기록을 확인한다.
10. 멀티 모드로 방을 생성하고 친구를 초대할 수 있다.
11. 방 찾기로 존재하는 방에 들어갈 수 있다.
12. 게임 초대 요청을 받아 친구가 있는 방에 들어갈 수 있다.
13. 랜덤 매칭을 통해 게임 방에 들어갈 수 있다.
14. 멀티 모드로 공포 테마를 진행한다.
15. 게임 종료 후 공포 테마 내 최고 기록을 확인한다.
16. 싱글 랭킹을 눌러 전체 랭킹을 확인한다.
17. 로그아웃을 하고 시연을 마친다.