포팅 매뉴얼 : MEET:Z

> **SSAFY 11기 공통 프로젝트 광주 1반 8팀**

개발 환경

프로젝트 기술 스택

Frontend

- Vite
- React
- TypeScript
- Zustand (상태 관리)
- Tailwind CSS
- React Router Dom
- Stomp JS (웹소켓 통신)

Backend

- Spring Boot
- Spring Security

데이터베이스

- MySQL (관계형 데이터베이스)
- MariaDB
- Redis

개발 도구 및 환경

- Jira (일정 관리)
- GitLab (코드 저장소)
- IntelliJ

- VS Code

클라우드 및 인프라

- Naver Cloud (클라우드 플랫폼)
- Nginx (웹 서버 및 리버스 프록시)
- Jenkins (CI/CD)
- Docker (컨테이너화)

환경변수 설정

Frontend : .env

```
VITE_NOW_BASEURL=deploy
VITE_MODE=local
VITE_API_LOCAL_URL = http://localhost:8088
VITE_API_DEPLOYED_URL = https://i11c108.p.ssafy.io
VITE_API_OPENVIDU_URL = https://i11c108.p.ssafy.io:8443
WDS_SOCKET_PORT=0
VITE_PUBLIC_URL=https://i11c108.p.ssafy.io/
VITE_CRYPTO_KEY=${crypto_key}
VITE_STT_API_KEY={api_key}
```

Backend : application.properties 파일

```
spring.application.name=meetz
server.port=8088
#server.ssl.enabled=true
#server.ssl.certificate=classpath:fullchain.pem
#server.ssl.certificate-private-key=classpath:privkey.pem
#server.ssl.trust-certificate=classpath:fullchain.pem
#server.ssl.protocol=TLSv1.2

#MariaDB Config
spring.datasource.driver-class-name=org.mariadb.jdbc.Driver
spring.datasource.url=${mariadb_url}
spring.datasource.username=${user_name}
```

```
spring.datasource.password=${uesr_password}

#JSP Config
spring.jpa.hibernate.ddl-auto=none
spring.jpa.hibernate.naming.physical-strategy=org.hibernate.b

#mariadb ??
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect

#Google SMTP
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=${email}
spring.mail.password=${password}
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.timeout=5000
spring.mail.properties.mail.smtp.starttls.enable=true

#Redis Configuration
spring.data.redis.host=${redis_url}
#spring.data.redis.host=${redis_host}
spring.data.redis.port=7546

#logging level Config
logging.level.root=INFO

#OpenVidu Settings(HTTPS PORT? ????)
OPENVIDU_URL: https://i11c108.p.ssafy.io:8443/
OPENVIDU_SECRET: ${secret}
#server.ssl.enabled: false

spring.jwt.secret=${jwt_secret}

spring.servlet.multipart.enabled=true
spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=10MB
server.tomcat.max-http-form-post-size=10MB
```

```
#Naver Clova speech
naver.bucket.name=meetz
clova.speech.invoke.url=${clova_url}
clova.speech.secret=${clova_secret}
naver.storage.path=${clova_path}

# Naver Object Storage
cloud.aws.credentials.accessKey=${cloud_accesskey}
cloud.aws.credentials.secretKey=${cloud_secretKey}
cloud.aws.region.static=kr-standard
cloud.aws.s3.endpoint=${endpoint}

logging.level.org.springframework.security=DEBUG
logging.level.org.springframework.web=DEBUG
```

Docker 파일

Frontend

```
#베이스 이미지 설정
FROM node:latest AS build
#작업 디렉토리 설정
WORKDIR /meetz-front/
#의존성 설치
COPY package*.json ./
RUN npm install
#앱 소스 복사
COPY . .
# .env 파일 생성
RUN echo "VITE_NOW_BASEURL=deployed\n\
    VITE_MODE=deployed\n\
    VITE_API_LOCAL_URL=http://localhost:8088\n\
    VITE_API_DEPLOYED_URL=https://i11c108.p.ssafy.io\n\
    VITE_PUBLIC_URL=https://i11c108.p.ssafy.io\n\
    WDS_SOCKET_PORT=0\n\
    WDS_SOCKET_PORT=0" > .env
```

```
#포트번호
EXPOSE 3000
#빌드
RUN npm run build
#서버 실행
CMD ["npm","run","dev"]
```

Backend

```
FROM openjdk:17
ARG JAR_FILE=build/libs/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
```

배포 방법

1. EC2 내부 방화벽 설정
2. 도커 설치
    a. 우분투 시스템 패키지 업데이트

    ```
    sudo apt-get update
    ```

    b. 필요한 패키지 설치

    ```
    sudo apt-get install apt-transport-https ca-certificate
    ```

    c. Docker의 공식 GPG키 추가

    ```
    curl -fsSL https://download.docker.com/linux/ubuntu/gpg
    ```

    d. Docker의 공식 apt 저장소 추가

    ```
    sudo add-apt-repository "deb [arch=amd64] https://downl
    ```

    e. 시스템 패키지 업데이트

```
sudo apt-get update
```

f. Docker 설치

```
sudo apt-get install docker-ce docker-ce-cli containerd
```

g. Docker 실행 상태 확인

```
sudo systemctl status docker
```

3. 젠킨스 설치 및 파이프라인 작성

a. 설치

```
cd /home/ubuntu && mkdir jenkins-data

ufw allow *8080*/tcp
ufw reload
ufw status

docker run -d -p 8080:8080 -v /home/ubuntu/jenkins-dat

docker logs jenkins

docker stop jenkins
docker ps -a
```

b. 파이프라인

i. frontend pipeline

```
pipeline {
    agent any

    tools {
        nodejs "nodejs"
    }

    environment {
```

```
        timestamp = "${System.currentTimeMillis() / 
    }

    stages {
        stage('Prepare') {
            steps {
                script {
                    // Get the ID of the meetzfront:
                    def oldImageId = sh(script: "doc
                    env.oldImageId = oldImageId
                }

                git branch: 'develop', credentialsId
                    url: 'https://lab.ssafy.com/s11-
            }

            post {
                success {
                    sh 'echo "Successfully Cloned Re
                }
                failure {
                    sh 'echo "Fail Cloned Repository
                }
            }
        }

        stage('Build Docker Image') {
            steps {
                dir('meetz-front') {
                    sh "docker build --tag meetzfron
                }
            }
        }

        stage('Run Docker Container') {
            steps {
                script {
                    // Check if the container is alr
```

```
                    def containerId = sh(script: "do

                    if (containerId) {
                        sh "docker rm -f meetzfront_
                    }

                    // Run the new container
                    def runStatus = sh(script: """
                        docker run \
                          --name=meetzfront_1 \
                          -p 3000:3000 \
                          -v /docker_projects/meetzf
                          --restart unless-stopped \
                          -e TZ=Asia/Seoul \
                          -d \
                          meetzfront_1:${timestamp}
                        """, returnStatus: true)

                    if (runStatus != 0) {
                        // If the container failed t
                        containerId = sh(script: "do

                        if (containerId) {
                            sh "docker rm -f meetzfr
                        }

                        def imageExists = sh(script:

                        if (imageExists) {
                            sh "docker rmi meetzfron
                        }

                        error("Failed to run the Doc
                    }

                    // If there's an existing 'lates
                    def latestExists = sh(script: "d
```

```
                    if (latestExists) {
                        sh "docker rmi meetzfront_1:
                    }

                    // If there was a previous image
                    if (env.oldImageId) {
                        sh "docker rmi ${env.oldImag
                    }

                    // Tag the new image as 'latest'
                    sh "docker tag meetzfront_1:${en
                }
            }
        }
    }
}
```

ii. backend pipeline

```
pipeline {
    agent any

    tools {
        nodejs "nodejs"
    }

    environment {
        timestamp = "${System.currentTimeMillis() / 
    }

    stages {
        stage('Prepare') {
            steps {
                script {
                    // Get the ID of the meetzfront:
                    def oldImageId = sh(script: "doc
                    env.oldImageId = oldImageId
                }
```

```
                git branch: 'develop', credentialsId
                    url: 'https://lab.ssafy.com/s11-
            }

            post {
                success {
                    sh 'echo "Successfully Cloned Re
                }
                failure {
                    sh 'echo "Fail Cloned Repository
                }
            }
        }

        stage('Build Docker Image') {
            steps {
                dir('meetz-front') {
                    sh "docker build --tag meetzfron
                }
            }
        }

        stage('Run Docker Container') {
            steps {
                script {
                    // Check if the container is alr
                    def containerId = sh(script: "do

                    if (containerId) {
                        sh "docker rm -f meetzfront_
                    }

                    // Run the new container
                    def runStatus = sh(script: """
                        docker run \
                          --name=meetzfront_1 \
                          -p 3000:3000 \
```

```
                -v /docker_projects/meetzf
                --restart unless-stopped \
                -e TZ=Asia/Seoul \
                -d \
                meetzfront_1:${timestamp}
            """, returnStatus: true)

        if (runStatus != 0) {
            // If the container failed t
            containerId = sh(script: "do

            if (containerId) {
                sh "docker rm -f meetzfr
            }

            def imageExists = sh(script:

            if (imageExists) {
                sh "docker rmi meetzfron
            }

            error("Failed to run the Doc
        }

        // If there's an existing 'lates
        def latestExists = sh(script: "d

        if (latestExists) {
            sh "docker rmi meetzfront_1:
        }

        // If there was a previous image
        if (env.oldImageId) {
            sh "docker rmi ${env.oldImag
        }

        // Tag the new image as 'latest'
        sh "docker tag meetzfront_1:${en
```

```
                    }
                }
            }
        }
    }
```

4. HTTPS 적용
    a. 80, 443포트 방화벽 해제
    b. 기본 라이브러리 설치
    c. Nginx 설치 및 conf 파일 작성