포팅 매뉴얼

1. 개발환경

1.1. Frontend

react

1.2. Backend

SpringBoot

1.3. Database

PostgreSQL, MariaDB, MongoDB, Redis

1.4. IDE

intellij

visualstudio

1.5. 형상/이슈관리

gitlab

jira

1.6. 기타 툴

postman

2. 환경변수

2.1. Frontend

env

```
VITE_BASE_URL =
VITE_APP_FCM_API_KEY=
VITE_APP_FCM_AUTH_DOMAIN=
VITE_APP_FCM_PROJECT_ID=
VITE_APP_FCM_STORAGE_BUCKET=
VITE_APP_FCM_MESSAGING_SENDER_ID=
VITE_APP_FCM_APP_ID=
VITE_APP_FCM_MEASUREMENT_ID=
VITE_APP_VAPID_KEY=
```

2.2. Backend

env

```
DB_USERNAME=
DB_PASSWORD=
```

```
# news
MongoDBUrl=
```

main.yml

```
spring:
  application:
    name: c203
  datasource:
    url: jdbc:postgresql://j11c203.p.ssafy.io:5432/postgres
    #    url: jdbc:postgresql://localhost:5432/postgres
    username: # PostgreSQL ID
    password: # PostgreSQL PW
  data:
    mongodb:
      uri: mongodb+srv://S11P23C203:jGhlp5h55N@ssafy.ngivl.mongodb.net/S11P23C203?authSource=
admin
    redis:
      #      host: localhost
      host: pigin-redis
      port: 6379
  jpa:
    hibernate:
      dialect: org.hibernate.dialect.PostgreSQL10Dialect
      ddl-auto: update
    properties:
      hibernate:
        format_sql: true
        default_batch_fetch_size: 100
    show-sql: true
  batch:
    job:
      enabled: false # 배치 작업 활성화
    jdbc:
      initialize-schema: never # 배치 메타데이터 테이블 자동 생성
  jwt:
    secret: # JWT 토큰 생성 Key

message:
  key:
    secret: # coolSMS Key
    apiKey: # coolSMS Key

ssafy:
  ssafydata:
    url: https://j11c203.p.ssafy.io/wallet
  api:
    key: c3e4356318eb4343bcf8a509ca1020f6
  securities:
    url: https://j11c203.p.ssafy.io/securities

server:
  servlet:
    context-path: /api

#Swagger
```

```
springdoc:
  api-docs:
    path: /api-docs
  swagger-ui:
    path: /swagger-ui.html

logging.level:
  org.hibernate.SQL: # Hibernate SQL 로그 레벨
  org.springframework.web: # Spring Web 로그 레벨
```

securities.yml

```
spring:
  application:
    name: SSAFYSecurities
  data:
    mongodb:
      uri: mongodb+srv://S11P23C203:jGhlp5h55N@ssafy.ngivl.mongodb.net/S11P23C203?authSource=
admin

koreainvestment:
  PROD: https://openapi.koreainvestment.com:9443
  APPKEY: # 한국 주자증권 App Key
  APPSECRET: # 한국 투자증권 AppSecret Key

stock:
  websocket:
    url: ws://ops.koreainvestment.com:21000
  codes: 005930,086520,000660,035420,373220,352820,035720,005380,001040,105560,000810,010130,
000100,068270,006400,051910,000670,247540,096770,196170

upbit:
  PROD: https://api.upbit.com/v1
  WEBSOCKET:
    PROD: wss://api.upbit.com/websocket/v1
    ACCESSKEY: # 업비트 API AppKey
    SECRETKEY: # 업비트 API SecretKey
  codes: KRW-BTC,KRW-ETH,KRW-USDT,KRW-XLM,KRW-XRP,KRW-LINK,KRW-ADA,KRW-DOGE,KRW-SOL,KRW-DOT

gold:
  APIKEY: # 한국 투자증권 API Key

server:
  port: 8089
  servlet:
    context-path: /securities
```

wallet

```
spring:
  application:
    name: MySSAFYData
  datasource:
    url: jdbc:mysql://stg-yswa-kr-practice-db-master.mariadb.database.azure.com:3306/S11P22C2
03?serverTimezone=UTC&useUnicode=true&characterEncoding=utf8
```

```
    username: S11P22C203
    password: CeMgWdnuyQ
    driver-class-name: org.mariadb.jdbc.Driver
  jpa:
    hibernate:
      ddl-auto: update
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MariaDBDialect

server:
  port: 8088
  servlet:
    context-path: /wallet

SSAFY:
  api:
    key: c3e4356318eb4343bcf8a509ca1020f6
```

news

```
MongoDBUrl=mongodb+srv://S11P23C203:jGhlp5h55N@ssafy.ngivl.mongodb.net/S11P23C203?authSource=
admin
```

2.3. 설정파일

2.3.1 Nginx: Nginx.conf

```
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
    worker_connections 768;
}

http {
    resolver 127.0.0.11 valid=30s;

    sendfile on;
    tcp_nopush on;
    tcp_nodelay on;
    keepalive_timeout 65;
    types_hash_max_size 2048;

    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
    ssl_prefer_server_ciphers on;

    access_log /var/log/nginx/access.log;
    error_log /var/log/nginx/error.log;

    gzip on;
```

```
    include /etc/nginx/conf.d/*.conf;
    include /etc/nginx/sites-enabled/*;
}
```

2.3.2 Nginx: default

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    index index.html index.htm index.nginx-debian.html;

    server_name _;

    location / {
        try_files $uri $uri/ =404;
    }

}

server {
    root /var/www/html;

    index index.html index.htm index.nginx-debian.html;
        server_name j11c203.p.ssafy.io;

    location / {
            proxy_pass http://localhost:5173;
            proxy_http_version 1.1;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;

            proxy_buffer_size 128k;
            proxy_buffers 4 256k;
            proxy_busy_buffers_size 256k;
            }

    location /api/ {
            proxy_pass http://localhost:8080;
            proxy_http_version 1.1;
            proxy_set_header Host $http_host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;

            proxy_buffer_size 128k;
            proxy_buffers 4 256k;
            proxy_busy_buffers_size 256k;
    }

    location /wallet/ {
            proxy_pass http://localhost:8088;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
```

```
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
        }

        location /securities/ {
                proxy_pass http://localhost:8089;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }

    # 백엔드 news 애플리케이션
        location /news/ {
                proxy_pass http://localhost:3000;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;
        }

    location /sw.js {
            proxy_pass http://localhost:5173/sw.js;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;

        # Ensure the correct MIME type is sent
            default_type application/javascript;
            add_header Cache-Control "no-cache";
    }

    listen [::]:443 ssl ipv6only=on;
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/j11c203.p.ssafy.io/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/j11c203.p.ssafy.io/privkey.pem;
    include /etc/letsencrypt/options-ssl-nginx.conf;
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

}
server {
    if ($host = j11c203.p.ssafy.io) {
        return 301 https://$host$request_uri;
    }


    listen 80 ;
    listen [::]:80 ;
    server_name j11c203.p.ssafy.io;
    return 404;
}
```

2.3.2 Jenkins: Jenkinsfile

main

```
@Library('shared-library@back') _

pipeline {
    agent any
    environment {
        SSH_CREDENTIALS = 'ec2-ssh-key'
    }

    stages {
        stage('Check Environment Variables') {
            steps {
                sh 'echo "EC2_IP is ${EC2_IP}"'
            }
        }
        stage('Git Clone') {
            steps {
                git branch: 'back',
                    url: 'https://lab.ssafy.com/s11-fintech-finance-sub1/S11P21C203.git',
                    credentialsId: 'c203-gitlab-jenkins-at'
            }
        }
        stage('Check for Changes') {
            steps {
                script {
                    env.CHANGES = sh(script: "git diff --name-only HEAD~1 HEAD", returnStdou
t: true).trim()
                    echo "Changes detected: ${env.CHANGES}"

                    if (!env.CHANGES.contains("back/c203/")) {
                        echo "No changes detected in './back/c203/'. Skipping the build."
                        currentBuild.result = 'ABORTED'
                        error("No changes in the relevant folder, stopping the build.")
                    } else {
                        echo "Changes detected in './back/c203/'. Proceeding with the build."
                    }
                }
            }
        }
        stage('Setup') {
            when {
                expression { env.CHANGES.contains("back/c203/") }
            }
            steps {
                withCredentials([file(credentialsId: 'back-yml-file-id', variable: 'MY_SECRET
_FILE')]) {
                    sh 'mkdir -p ./back/c203/src/main/resources'
                    sh 'chmod -R 755 ./back/c203/src/main/resources'
                    sh 'cp $MY_SECRET_FILE ./back/c203/src/main/resources/application.yml'
                }
            }
        }
        stage('Set Permission') {
            when {
                expression { env.CHANGES.contains("back/c203/") }
            }
            steps {
                dir('./back/c203') {
```

```
                sh 'chmod +x ./gradlew'
            }
        }
    }
    stage('Build Jar') {
        when {
            expression { env.CHANGES.contains("back/c203/") }
        }
        steps {
            dir('./back/c203') {
                sh './gradlew clean build -x test'
            }
        }
    }
    stage('Build Docker Image') {
        when {
            expression { env.CHANGES.contains("back/c203/") }
        }
        steps {
            sh 'docker build -t pigin-be-main:main-latest ./back/c203'
        }
    }
    stage('Push and Deploy') {
        when {
            expression { env.CHANGES.contains("back/c203/") }
        }
        steps {
            script {
                withCredentials([usernamePassword(credentialsId: 'dockerhub-credentials',
usernameVariable: 'DOCKERHUB_USER', passwordVariable: 'DOCKERHUB_PASS')]) {
                    sh 'echo $DOCKERHUB_PASS | docker login -u $DOCKERHUB_USER --password
-stdin'
                    sh 'docker tag pigin-be-main:main-latest $DOCKERHUB_USER/pigin-be-mai
n:main-latest'
                    sh 'docker push $DOCKERHUB_USER/pigin-be-main:main-latest'

                    sshagent([SSH_CREDENTIALS]) {
                        sh """
                        scp -o StrictHostKeyChecking=no ./docker-compose.be-main.yml ubun
tu@${EC2_IP}:/home/ubuntu/myapp/docker-compose.be-main.yml
                        ssh -o StrictHostKeyChecking=no ubuntu@${EC2_IP} '
                            cd /home/ubuntu/myapp
                            docker-compose -f docker-compose.be-main.yml down
                            docker-compose -f docker-compose.be-main.yml pull
                            docker-compose -f docker-compose.be-main.yml up -d --build
                            '
                        """
                    }
                }
            }
        }
    }
    stage('Clean Up Docker Images') {
        when {
            expression { env.CHANGES.contains("back/c203/") }
        }
        steps {
            sh 'docker image prune -a -f'
```

```
            }
        }
    }
    post {
        success {
            echo '백엔드 메인 배포 완료'
            sendNotification('success')
        }
        aborted {
            echo '빌드가 변경 사항이 없어 중단되었습니다.'
        }
        failure {
            echo '백엔드 메인 배포 실패'
            sendNotification('failure')
        }
    }
}
```

securities

```
@Library('shared-library@back') _

pipeline {
    agent any
    environment {
        SSH_CREDENTIALS = 'ec2-ssh-key'
    }

    stages {
        stage('Check Environment Variables') {
            steps {
                sh 'echo "EC2_IP is ${EC2_IP}"'
            }
        }
        stage('Git Clone') {
            steps {
                git branch: 'back',
                    url: 'https://lab.ssafy.com/s11-fintech-finance-sub1/S11P21C203.git',
                    credentialsId: 'c203-gitlab-jenkins-at'
            }
        }
        stage('Check for Changes') {
            steps {
                script {
                    env.CHANGES = sh(script: "git diff --name-only HEAD~1 HEAD", returnStdou
t: true).trim()
                    echo "Changes detected: ${env.CHANGES}"

                    if (!env.CHANGES.contains("back/SSAFYSecurities/")) {
                        echo "No changes detected in './back/SSAFYSecurities/'. Skipping the
build."
                        currentBuild.result = 'ABORTED'
                        error("No changes in the relevant folder, stopping the build.")
                    } else {
                        echo "Changes detected in './back/MySSAFYData/'. Proceeding with the
build."
```

```
                    }
                }
            }
        }
        stage('Setup') {
            when {
                expression { env.CHANGES.contains("back/SSAFYSecurities/") }
            }
            steps {
                withCredentials([file(credentialsId: 'securities-yml-file-id', variable: 'MY_
SECRET_FILE')]) {
                    sh 'chmod -R 755 ./back/SSAFYSecurities/src/main/resources'
                    sh 'cp $MY_SECRET_FILE ./back/SSAFYSecurities/src/main/resources/applicat
ion.yml'
                }
            }
        }
        stage('Set Permission') {
            when {
                expression { env.CHANGES.contains("back/SSAFYSecurities/") }
            }
            steps {
                dir('./back/SSAFYSecurities') {
                    sh 'chmod +x ./gradlew'
                }
            }
        }
        stage('Build Jar') {
            when {
                expression { env.CHANGES.contains("back/SSAFYSecurities/") }
            }
            steps {
                dir('./back/SSAFYSecurities') {
                    sh './gradlew clean build -x test'
                }
            }
        }
        stage('Build Docker Image') {
            when {
                expression { env.CHANGES.contains("back/SSAFYSecurities/") }
            }
            steps {
                sh 'docker build -t pigin-be-securities:securities-latest ./back/SSAFYSecurit
ies'
            }
        }
        stage('Push and Deploy') {
            when {
                expression { env.CHANGES.contains("back/SSAFYSecurities/") }
            }
            steps {
                script {
                    withCredentials([usernamePassword(credentialsId: 'dockerhub-credentials',
usernameVariable: 'DOCKERHUB_USER', passwordVariable: 'DOCKERHUB_PASS')]) {
                        sh 'echo $DOCKERHUB_PASS | docker login -u $DOCKERHUB_USER --password
-stdin'
                        sh 'docker tag pigin-be-securities:securities-latest $DOCKERHUB_USER/
pigin-be-securities:securities-latest'
```

```
                        sh 'docker push $DOCKERHUB_USER/pigin-be-securities:securities-lates
t'

                        sshagent([SSH_CREDENTIALS]) {
                            sh """
                            scp -o StrictHostKeyChecking=no ./docker-compose.be-securities.ym
l ubuntu@${EC2_IP}:/home/ubuntu/myapp/docker-compose.be-securities.yml
                            ssh -o StrictHostKeyChecking=no ubuntu@${EC2_IP} '
                                cd /home/ubuntu/myapp
                                docker-compose -f docker-compose.be-securities.yml down
                                docker-compose -f docker-compose.be-securities.yml pull
                                docker-compose -f docker-compose.be-securities.yml up -d --bu
ild
                                '
                            """
                        }
                    }
                }
            }
        }
        stage('Clean Up Docker Images') {
            when {
                expression { env.CHANGES.contains("back/SSAFYSecurities/") }
            }
            steps {
                sh 'docker image prune -a -f'
            }
        }
    }
    post {
        success {
            echo '백엔드 증권 배포 완료'
            sendNotification('success')
        }
        aborted {
            echo '빌드가 변경 사항이 없어 중단되었습니다.'
        }
        failure {
            echo '백엔드 증권 배포 실패'
            sendNotification('failure')
        }
    }
}
```

wallet

```
@Library('shared-library@back') _

pipeline {
    agent any
    environment {
        SSH_CREDENTIALS = 'ec2-ssh-key'
    }

    stages {
        stage('Check Environment Variables') {
```

```
            steps {
                sh 'echo "EC2_IP is ${EC2_IP}"'
            }
        }
        stage('Git Clone') {
            steps {
                git branch: 'back',
                    url: 'https://lab.ssafy.com/s11-fintech-finance-sub1/S11P21C203.git',
                    credentialsId: 'c203-gitlab-jenkins-at'
            }
        }

        stage('Setup') {
            steps {
                withCredentials([file(credentialsId: 'wallet-yml-file-id', variable: 'MY_SECR
ET_FILE')]) {
                    sh 'chmod -R 755 ./back/MySSAFYData/src/main/resources'
                    sh 'cp $MY_SECRET_FILE ./back/MySSAFYData/src/main/resources/application.
yml'
                }
            }
        }

        stage('Check for Changes') {
            steps {
                script {
                    env.CHANGES = sh(script: "git diff --name-only HEAD~1 HEAD", returnStdou
t: true).trim()
                    echo "Changes detected: ${env.CHANGES}"

                    if (!env.CHANGES.contains("back/MySSAFYData/")) {
                        echo "No changes detected in './back/MySSAFYData/'. Skipping the buil
d."
                        currentBuild.result = 'ABORTED'
                        error("No changes in the relevant folder, stopping the build.")
                    } else {
                        echo "Changes detected in './back/MySSAFYData/'. Proceeding with the
build."
                    }
                }
            }
        }
        stage('Set Permission') {
            when {
                expression { env.CHANGES.contains("back/MySSAFYData/") }
            }
            steps {
                dir('./back/MySSAFYData') {
                    sh 'chmod +x ./gradlew'
                }
            }
        }
        stage('Build Jar') {
            when {
                expression { env.CHANGES.contains("back/MySSAFYData/") }
            }
            steps {
                dir('./back/MySSAFYData') {
```

```
                sh './gradlew clean build -x test'
            }
        }
    }
    stage('Build Docker Image') {
        when {
            expression { env.CHANGES.contains("back/MySSAFYData/") }
        }
        steps {
            sh 'docker build -t pigin-be-wallet:wallet-latest ./back/MySSAFYData'
        }
    }
    stage('Push and Deploy') {
        when {
            expression { env.CHANGES.contains("back/MySSAFYData/") }
        }
        steps {
            script {
                withCredentials([usernamePassword(credentialsId: 'dockerhub-credentials',
usernameVariable: 'DOCKERHUB_USER', passwordVariable: 'DOCKERHUB_PASS')]) {
                    sh 'echo $DOCKERHUB_PASS | docker login -u $DOCKERHUB_USER --password
-stdin'
                    sh 'docker tag pigin-be-wallet:wallet-latest $DOCKERHUB_USER/pigin-be
-wallet:wallet-latest'
                    sh 'docker push $DOCKERHUB_USER/pigin-be-wallet:wallet-latest'

                    sshagent([SSH_CREDENTIALS]) {
                        sh """
                        scp -o StrictHostKeyChecking=no ./docker-compose.be-wallet.yml ub
untu@${EC2_IP}:/home/ubuntu/myapp/docker-compose.be-wallet.yml
                        ssh -o StrictHostKeyChecking=no ubuntu@${EC2_IP} '
                            cd /home/ubuntu/myapp
                            docker-compose -f docker-compose.be-wallet.yml down
                            docker-compose -f docker-compose.be-wallet.yml pull
                            docker-compose -f docker-compose.be-wallet.yml up -d --build
                            '
                        """
                    }
                }
            }
        }
    }
    stage('Clean Up Docker Images') {
        when {
            expression { env.CHANGES.contains("back/MySSAFYData/") }
        }
        steps {
            sh 'docker image prune -a -f'
        }
    }
}
post {
    success {
        echo '백엔드 지갑 배포 완료'
        sendNotification('success')
    }
    aborted {
        echo '빌드가 변경 사항이 없어 중단되었습니다.'
```

```
        }
        failure {
            echo '백엔드 지갑 배포 실패'
            sendNotification('failure')
        }
    }
}
```

news

```
@Library('shared-library@back') _

pipeline {
    agent any
    environment {
        SSH_CREDENTIALS = 'ec2-ssh-key'
    }

    stages {
        stage('Check Python Version') {
            steps {
                sh 'python3 --version'
            }
        }
        stage('Check Environment Variables') {
            steps {
                sh 'echo "EC2_IP is ${EC2_IP}"'
            }
        }
        stage('Git Clone') {
            steps {
                git branch: 'back',
                    url: 'https://lab.ssafy.com/s11-fintech-finance-sub1/S11P21C203.git',
                    credentialsId: 'c203-gitlab-jenkins-at'
            }
        }
        stage('Check for Changes') {
            steps {
                script {
                    env.CHANGES = sh(script: "git diff --name-only HEAD~1 HEAD", returnStdou
t: true).trim()
                    echo "Changes detected: ${env.CHANGES}"

                    if (!env.CHANGES.contains("back/NewsCrawling/")) {
                        echo "No changes detected in './back/NewsCrawling/'. Skipping the bui
ld."
                        currentBuild.result = 'ABORTED'
                        error("No changes in the relevant folder, stopping the build.")
                    } else {
                        echo "Changes detected in './back/NewsCrawling/'. Proceeding with the
build."
                    }
                }
            }
        }
        stage('Setup Environment') {
```

```
            when {
                expression { env.CHANGES.contains("back/NewsCrawling/") }
            }
            steps {
                withCredentials([file(credentialsId: 'news-env-file-id', variable: 'MY_ENV_FI
LE')]) {
                    sh 'cp $MY_ENV_FILE ./back/NewsCrawling/.env'
                }
            }
        }
        stage('Install Dependencies') {
            when {
                expression { env.CHANGES.contains("back/NewsCrawling/") }
            }
            steps {
                dir('./back/NewsCrawling') {
                    sh 'python3 -m pip install -r requirements.txt'
                }
            }
        }
        stage('Build Docker Image') {
            when {
                expression { env.CHANGES.contains("back/NewsCrawling/") }
            }
            steps {
                sh 'docker build -t pigin-be-news:news-latest ./back/NewsCrawling'
            }
        }
        stage('Push and Deploy') {
            when {
                expression { env.CHANGES.contains("back/NewsCrawling/") }
            }
            steps {
                script {
                    withCredentials([usernamePassword(credentialsId: 'dockerhub-credentials',
usernameVariable: 'DOCKERHUB_USER', passwordVariable: 'DOCKERHUB_PASS')]) {
                        sh 'echo $DOCKERHUB_PASS | docker login -u $DOCKERHUB_USER --password
-stdin'
                        sh 'docker tag pigin-be-news:news-latest $DOCKERHUB_USER/pigin-be-new
s:news-latest'
                        sh 'docker push $DOCKERHUB_USER/pigin-be-news:news-latest'

                        sshagent([SSH_CREDENTIALS]) {
                            sh """
                            scp -o StrictHostKeyChecking=no ./docker-compose-be-news.yml ubun
tu@${EC2_IP}:/home/ubuntu/myapp/docker-compose-be-news.yml
                            ssh -o StrictHostKeyChecking=no ubuntu@${EC2_IP} '
                                cd /home/ubuntu/myapp
                                docker-compose -f docker-compose-be-news.yml down
                                docker-compose -f docker-compose-be-news.yml pull
                                docker-compose -f docker-compose-be-news.yml up -d --build
                                '
                            """
                        }
                    }
                }
            }
        }
```

```
        stage('Clean Up Docker Images') {
            when {
                expression { env.CHANGES.contains("back/NewsCrawling/") }
            }
            steps {
                sh 'docker image prune -a -f'
            }
        }
    }
    post {
        success {
            echo '백엔드 뉴스 배포 완료'
            sendNotification('success')
        }
        aborted {
            echo '빌드가 변경 사항이 없어 중단되었습니다.'
        }
        failure {
            echo '백엔드 뉴스 배포 실패'
            sendNotification('failure')
        }
    }
}
```

front

```
@Library('shared-library@front') _

pipeline {
    agent any
    environment {
        SSH_CREDENTIALS = 'ec2-ssh-key'
    }

    stages {
        stage('Check Environment Variables') {
            steps {
                sh 'echo "EC2_IP is ${EC2_IP}"'
            }
        }
        stage('Git Clone') {
            steps {
                git branch: 'front',
                    url: 'https://lab.ssafy.com/s11-fintech-finance-sub1/S11P21C203.git',
                    credentialsId: 'c203-gitlab-jenkins-at'
            }
        }
        stage('Install Dependencies') {
            steps {
                dir('./front') {
                    sh 'npm install'
                }
            }
        }
        stage('Build Frontend') {
            steps {
```

```
                dir('./front') {
                    sh 'npm run build'
                }
            }
        }
        stage('Front Docker Image') {
            steps {
                sh 'docker build -t pigin-front:latest ./front'
            }
        }
        stage('Push and Deploy') {
            steps {
                script {
                    withCredentials([usernamePassword(credentialsId: 'dockerhub-credentials',
usernameVariable: 'DOCKERHUB_USER', passwordVariable: 'DOCKERHUB_PASS')]) {
                        sh 'echo $DOCKERHUB_PASS | docker login -u $DOCKERHUB_USER --password
-stdin'
                        sh 'docker tag pigin-front:latest $DOCKERHUB_USER/pigin-front:latest'
                        sh 'docker push $DOCKERHUB_USER/pigin-front:latest'

                        sshagent([SSH_CREDENTIALS]) {
                            sh """
                            scp -o StrictHostKeyChecking=no ./docker-compose.frontend.yml ubu
ntu@${EC2_IP}:/home/ubuntu/myapp/docker-compose.frontend.yml
                            ssh -o StrictHostKeyChecking=no ubuntu@${EC2_IP} '
                                cd /home/ubuntu/myapp
                                docker-compose -f docker-compose.frontend.yml down
                                docker-compose -f docker-compose.frontend.yml pull
                                docker-compose -f docker-compose.frontend.yml up -d
                            '
                            """
                        }
                    }
                }
            }
        }
        stage('Clean Up Docker Images') {
            steps {
                sh 'docker image prune -a -f'
            }
        }
    }
    post {
        success {
            echo '프론트 배포 완료'
            sendNotification('success')
        }
        failure {
            echo '프론트 배포 실패'
            sendNotification('failure')
        }
    }
}
```

2.3.3 Dockerfile: Backend

main

```
FROM openjdk:17
WORKDIR /app
COPY build/libs/*.jar app.jar

ENV TZ=Asia/Seoul

ADD https://raw.githubusercontent.com/vishnubob/wait-for-it/master/wait-for-it.sh /wait-for-i
t.sh
RUN chmod +x /wait-for-it.sh

CMD ["/wait-for-it.sh", "postgres:5432", "--timeout=30", "--strict", "--", "java", "-jar", "a
pp.jar"]
```

securities

```
FROM openjdk:17
ARG JAR_FILE=build/libs/*.jar
COPY ${JAR_FILE} app.jar

ENV TZ=Asia/Seoul

ENTRYPOINT ["java", "-jar", "app.jar"]
```

wallet

```
FROM openjdk:17

ARG JAR_FILE=build/libs/*.jar
COPY ${JAR_FILE} app.jar

ENV TZ=Asia/Seoul

ENTRYPOINT ["java", "-jar", "app.jar"]
```

news

```
FROM python:3.12.5-slim

WORKDIR /app

COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

COPY . .

CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "3000", "--reload"]
```

news

```
beautifulsoup4
requests
pandas
python-dotenv
pymongo
fastapi[all]
```

```
uvicorn
selenium
```

2.3.4 Dockerfile: Front

```
# Node.js v20.16.0 이미지 사용
FROM node:20.16.0

# 작업 디렉토리 설정
WORKDIR /app

# 시간대 설정
ENV TZ=Asia/Seoul

# package.json과 package-lock.json 복사
COPY package.json package-lock.json* ./

# 의존성 설치
RUN npm install

# 소스 코드 복사
COPY . .

# 애플리케이션 빌드
RUN npm run build

# serve 패키지 글로벌 설치
RUN npm install -g serve

# 애플리케이션을 serve로 서빙
CMD ["serve", "-s", "dist", "-l", "5173"]

# 컨테이너의 포트 노출
EXPOSE 5173
```

2.3.5 DockerCompose: Backend

docker-compose.be-main.yml

```
services:
  backend:
    image: zzun73/pigin-be-main:main-latest
    container_name: pigin-be-main
    ports:
      - "8080:8080"
    environment:
      SPRING_DATASOURCE_URL: jdbc:postgresql://postgres:5432/postgres
      SPRING_DATASOURCE_USERNAME: ${DB_USERNAME}
      SPRING_DATASOURCE_PASSWORD: ${DB_PASSWORD}
    depends_on:
      - postgres
    networks:
      - pigin-network
    restart: always
    healthcheck:
      test: ["CMD-SHELL", "curl -f http://localhost:8080 || exit 1"]
```

```
      interval: 30s
      timeout: 10s
      retries: 3
      start_period: 40s

  postgres:
    image: postgres:16.4
    container_name: pigin-postgres
    environment:
      POSTGRES_USER: ${DB_USERNAME}
      POSTGRES_PASSWORD: ${DB_PASSWORD}
    volumes:
      - postgres-data:/var/lib/postgresql/data
    ports:
      - "5432:5432"
    networks:
      - pigin-network
    restart: always
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U ${DB_USERNAME}"]
      interval: 30s
      timeout: 10s
      retries: 5
      start_period: 10s
  redis:
    image: redis:latest
    container_name: pigin-redis
    ports:
      - "6379:6379"
    networks:
      - pigin-network
    volumes:
      - redis-data:/data
    restart: always

networks:
  pigin-network:
    driver: bridge

volumes:
  postgres-data:
  redis-data:
```

docker-compose-be-news.yml

```
services:
  backend-news:
    image: zzun73/pigin-be-news:news-latest
    container_name: pigin-be-news
    ports:
      - "3000:3000"
    environment:
      - MongoDBUrl=${MongoDBUrl}
    networks:
      - myapp_pigin-network
    restart: unless-stopped

networks:
```

```
  myapp_pigin-network:
    external: true
```

docker-compose.be-securities.yml

```
services:
  backend-securities:
    image: zzun73/pigin-be-securities:securities-latest
    container_name: pigin-be-securities
    ports:
      - "8089:8089"
    environment:
      SPRING_DATASOURCE_URL: jdbc:postgresql://pigin-postgres:5432/postgres
      SPRING_DATASOURCE_USERNAME: ${DB_USERNAME}
      SPRING_DATASOURCE_PASSWORD: ${DB_PASSWORD}
    networks:
      - myapp_pigin-network
    restart: always
    healthcheck:
      test: ["CMD-SHELL", "curl -f http://localhost:8089 || exit 1"]
      interval: 30s
      timeout: 10s
      retries: 3
      start_period: 40s

networks:
  myapp_pigin-network:
    external: true
```

docker-compose.be-wallet.yml

```
services:
  backend-wallet:
    image: zzun73/pigin-be-wallet:wallet-latest
    container_name: pigin-be-wallet
    ports:
      - "8088:8088"
    networks:
      - pigin-network
    restart: always
    healthcheck:
      test: ["CMD-SHELL", "curl -f http://localhost:8088 || exit 1"]
      interval: 30s
      timeout: 10s
      retries: 3
      start_period: 40s

networks:
  pigin-network:
    driver: bridge
```

3. EC2 세팅

3.1. Docker 설치

```
# 패키지 업데이트
sudo apt-get update

# 패키지 설치
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common

# Docker의 공식 GPG 키 추가
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

# Docker 저장소 추가
sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"

# 패키지 재 업데이트
sudo apt-get update

# Docker CE 설치
sudo apt-get install docker-ce docker-ce-cli containerd.io

# Docker 서비스 시작
sudo systemctl start docker

# Docker 설치 확인
docker --version
```

3.2. Docker 실행 권한 설정

```
# 도커그룹생성
sudo groupadd docker

# 도커그룹에 유저추가 (Docker 데몬에 대한 접근 권한을 부여)
sudo usermod -aG docker ${USER} or ${whoami}
# or
sudo gpasswd -a $USER docker

# 도커 재시작
sudo service docker restart
# or
newgrp docker # 현재 세션의 사용자 그룹을 변경


# 현재 사용자 로그아웃 및 재로그인 필수
# exit 후 su username

# 테스트
docker ps
```

3.3. Docker Compose 설치

```
# docker-compose 설치
sudo curl -L "https://github.com/docker/compose/releases/download/v2.29.3/docker-compose-$(un
ame -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose

# 테스트
docker compose --version
```

3.4. Nginx 설치

```
# Ubuntu에서 Certbot 설치하기
sudo apt update
sudo apt install certbot
sudo apt install python3-certbot-nginx

# Certbot을 이용한 SSL 인증서 발급
sudo certbot --nginx

# certbot 버전 확인
certbot --version
certbot 0.40.0
```

3.5. EC2 Port

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:9090            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:3000            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:5173            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:5432            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8089            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8088            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6010          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:6379            0.0.0.0:*               LISTEN
tcp6       0      0 :::8989                 :::*                    LISTEN
tcp6       0      0 :::8988                 :::*                    LISTEN
tcp6       0      0 :::9090                 :::*                    LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::443                  :::*                    LISTEN
tcp6       0      0 :::3000                 :::*                    LISTEN
tcp6       0      0 :::5173                 :::*                    LISTEN
tcp6       0      0 :::5432                 :::*                    LISTEN
tcp6       0      0 :::29418                :::*                    LISTEN
tcp6       0      0 ::1:6010                :::*                    LISTEN
tcp6       0      0 :::8089                 :::*                    LISTEN
tcp6       0      0 :::8088                 :::*                    LISTEN
tcp6       0      0 :::8080                 :::*                    LISTEN
tcp6       0      0 :::6379                 :::*                    LISTEN
udp        0      0 127.0.0.53:53           0.0.0.0:*
udp        0      0 172.26.5.131:68         0.0.0.0:*
```

3.6. 방화벽(UFW) 설정

```
To                         Action      From
--                         ------      ----
22                         ALLOW       Anywhere
8989                       ALLOW       Anywhere
443                        ALLOW       Anywhere
80                         ALLOW       Anywhere
29418/tcp                  ALLOW       Anywhere
9090/tcp                   ALLOW       Anywhere
5173                       ALLOW       Anywhere
6379/tcp                   ALLOW       Anywhere
22 (v6)                    ALLOW       Anywhere (v6)
8989 (v6)                  ALLOW       Anywhere (v6)
443 (v6)                   ALLOW       Anywhere (v6)
80 (v6)                    ALLOW       Anywhere (v6)
29418/tcp (v6)             ALLOW       Anywhere (v6)
9090/tcp (v6)              ALLOW       Anywhere (v6)
5173 (v6)                  ALLOW       Anywhere (v6)
6379/tcp (v6)              ALLOW       Anywhere (v6)
```