포팅 메뉴얼

🗃️ Index

---

1. Stacks

- Issue Management : **JIRA**
- SCM(Software Configuration Management) : **GITLAB**

- Communities : **Notion, MatterMost**
- Development Environment
    - **IntelliJ 2023.3**
    - **Android Studio Hedgehog | 2023.1.1**
    - **Java 17**
    - **Kotlin**
    - **Spring Boot 3.2.2**
    - **Ubuntu 20.04.6 LTS (AWS EC2)**
    - **MariaDB (azure)**

2. Env

- application.yml

```
server:
  port: 포트 번호
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: 싸피 DB
    username: 유저네임
    password: 비밀번호
  jpa:
    hibernate:
      ddl-auto: none

  data:
    redis:
      host: 레디스 호스트
      port: 포트 번호

  security:
    oauth2:
      client:
        registration:
          google:
            client-id:  발급 받은 client ID
            client-secret: 발급 받은 Client Secret
            authorization-grant-type: 인증 코드
            redirect-uri: /oauth2/code/registrationId
            scope:
              - email
  servlet:
    multipart:
      max-file-size: 100MB
      max-request-size: 100MB


logging:
  level:
    root: info

cloud:
  aws:
    credentials:
      access-key: 액세스 키
      secret-key: 시크릿 키
    s3:
      bucket: 버킷 이름
    region:
      static: 지역 이름
      auto: false
    stack:
      auto: false
decorator:
  datasource:
    p6spy:
      enable-logging: true

jwt:
  secret: jwt 시크릿키
```

- local.properties

```
naver_map_client_secret=네이버 api 인증키
```

3. Build & Distribute

- Git Clone

```
git clone https://lab.ssafy.com/s10-webmobile4-sub2/S10P12
```

- Spring Boot

```
# Build
cd togeduck
chmod +x ./gradlew
./gradlew clean build
docker build -t togeduck:latest .
```

- redis

```
docker pull redis:alpine
docker run -d -p ${port}:${port}--name=redis redis:alpine
```

- Android

```
build -> Generate Signed Bundle (apk로 추출)
```

4. Deployment Command

Local

- Backend Server

```
docker run -d -p ${port}:${port} --name togeduck togeduck:
```

EC2 - Jenkins Pipeline

- jenkins 프로젝트구성 - Pipeline - Pipeline Script 선택, 아래 스크립트 복사, 붙여넣기 및 채우기

```
pipeline {
    agent any
    environment {
            repository = {리포지토리}
            DOCKERHUB_CREDENTIALS = credentials('dockerhub')
    }
    stages {
        stage('gitlab clone') {
            steps {
                // Get some code from a GitHub repository
                git branch: 'master', credentialsId: {credent
            }
        }

        stage('build') {
            steps {
                dir('togeduck') {
                    // some block
                    sh'''
                        echo build start
                        chmod +x ./gradlew
                        ./gradlew clean build
                    '''
                }
            }
        }
        stage('Cleaning up') {
            steps {
                sh "docker rmi $repository:latest" // docker
            }
        }
        stage('Building our image') {
            steps {
                script {
                    dir('togeduck') {
                        // sh "docker build -t $repository:la
                        dockerimage = docker.build("$reposito
                    }
                }
            }
        }
        stage('Login'){
            steps{
                sh 'echo $DOCKERHUB_CREDENTIALS_PSW | docker
            }
        }
        stage('Deploy our image') {
            steps {
                script {
                    docker.withRegistry('https://registry.hub
                        //  Building new image
                        dockerimage.push();
                    }
                }
            }
        }
        stage('Pull image') {
            steps {
                sh "docker pull $repository:latest" // docker
            }
        }
        stage('Run Image') {

            steps {
                dir('togeduck') {
                    sh'''
                        docker stop togeduck
                        docker rm togeduck
                        docker run -d -p ${port}:${port}--nam
                    '''



                }
            }
        }
    }
}
```

5. Nginx default

```

server {
        listen 80 default_server;
        listen [::]:80 default_server;

        root /var/www/html;

        server_name _;

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


server {

        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 i10a301.p.ssafy.io; # managed by Certbot
        client_max_body_size 10M;

        location / {
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_
                proxy_set_header X-Forwarded-Proto $scheme;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection "upgrade";
                proxy_pass http://localhost:8082;
                proxy_redirect off;
        }

    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/i10a301.p.ssafy.io/
    ssl_certificate_key /etc/letsencrypt/live/i10a301.p.ssafy
    include /etc/letsencrypt/options-ssl-nginx.conf; # manage
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed
}
server {
    if ($host = {host_uri}) {
        return 308 https://$host$request_uri;
    } # managed by Certbot


        listen 80 ;
        listen [::]:80 ;
    server_name i10a301.p.ssafy.io;
    return 404; # managed by Certbot
}
```