SAMALS

포팅 매뉴얼

Ⅰ. 개요

1. 프로젝트 개요
2. 개발 환경
3. 기술 스택

Ⅱ. 포팅 가이드

1. 환경 변수 설정
2. 빌드 및 배포
3. CI/CD 설정

Ⅰ. 개요

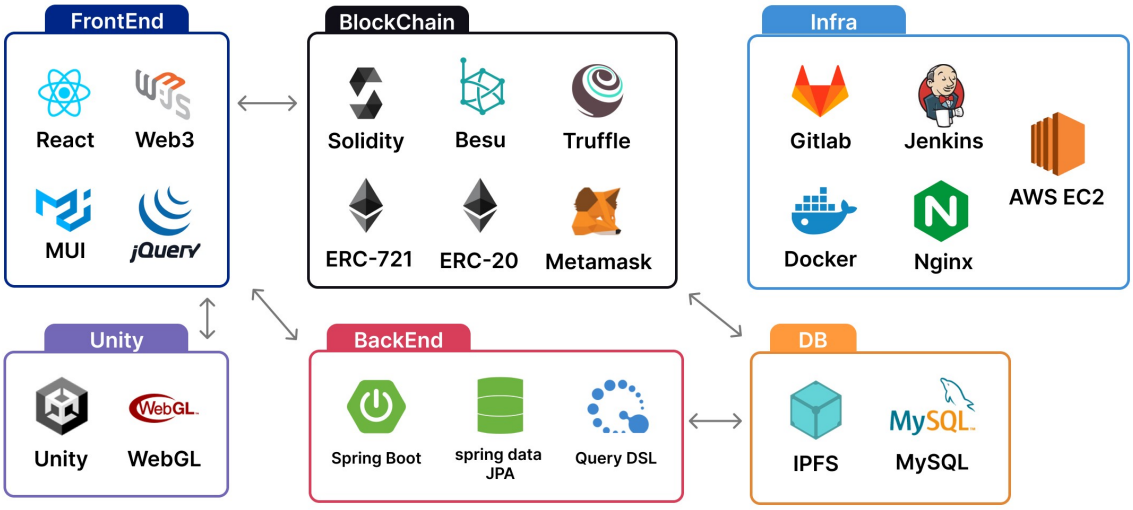
1. 프로젝트 개요

SAMALS는 멸종 위기 동물 보호를 위한 기부 및 NFT 거래 플랫폼으로, 제작한 동물 일러스트를 PFP화하여 NFT로 민팅 후 판매합니다. 판매 과정에서 발생한 수익은 WWF 등과 같은 동물 보호 단체에 기부하는 것을 목표로 하고 있습니다.

2. 개발 환경

- IntelliJ IDEA 2022.2.1
- Remix
- HeidiSQL
- VS Code : 1.66.0
- Unity 2021.3.8f1
- AWS EC2 Ubuntu 20.04 LTS

3. 기술 스택



Ⅱ. 포팅 가이드

1. 환경 변수 설정

- **Frontend (React)**

  없음

- **Backend (Spring boot)**

backend/Samals/src/main/resources/application.yml

```
spring:
# mysql DB
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
     url:jdbc:mysql://{도메인주소}/{테이블명}?serverTimezone=Asia/Seoul
    username: {Id}
    password: {Password}
```

- **DB (MySQL)**

Infra/db/docker-compose

```
version:'3'
        services:
            db:
                image: mysql:5.7
                container_name: Database
                ports:
                    - 3306:3306
                environment:
                    MYSQL_USER: {USER}
                    MYSQL_PASSWORD: {PASSWORD}
                    MYSQL_ROOT_PASSWORD: {PASSWORD}
                    TZ: Asia/Seoul
```

```
volumes:
        - ./db/mysql/data:/var/lib/mysql
        - ./db/mysql/sql:/sql
        - ./db/mysql/init:/docker-entrypoint-initdb.d
restart: always
```

- **Blockchain (Solidity)**

smart-contract/src/truffle-config.js

```
const PrivateKEY="{지갑 개인키}";
const WALLET = "{배포할 지갑 주소}";
```

2. 빌드 및 배포

A. Nginx + Certbot

B. Frontend

C. Backend API 서버

D. MySQL DB 서버

D. Besu Network

E. Smart Contract

A. Nginx + Certbot 설정

1. SSL 인증서 발급 + Nginx 배포 Code

Repository : Infra/nginx/

1. data/nginx/conf.d/app.conf
2. docker-compose.yml
3. init-letsencript.sh

2. Code 실행

```
$ ./init-letsencrypt.sh
```

3. HTTPS server 설정

```
$ docker ps
$ docker exec -it {Container Id} /bin/sh
```

Nginx container 내부

```
# vi /etc/nginx/nginx.conf
```

/etc/nginx/nginx.conf

```
http{
 ...
server {
    listen 80;
    listen [::]:80;
    server_name {도메인 주소};
    location /.well-known/acme-challenge/ {
            allow all;
            root /var/www/certbot;
    }
    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen 443 ssl;
    server_name {도메인 주소};
    server_tokens off;

    ssl_certificate /etc/letsencrypt/live/{도메인 주소}/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/{도메인 주소}/privkey.pem;
    include /etc/letsencrypt/options-ssl-nginx.conf;
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

    #Frontend
    location / {
        proxy_pass http://172.17.0.1:3001;
        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;
    }
    #Backend API 서버
    location /api {
        proxy_pass http://172.17.0.1:8080/api;
        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;
    }
    #Backend 이미지 서버
    location /image {
        proxy_pass http://172.17.0.1:9090/image;
        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;
    }
}}
```

B. Frontend 배포

Client/

빌드

```
$ npm install --force
$ npm run build
```

배포

```
$ docker-compose build
$ docker-compose up -d
```

C. Backend 배포

backend/samals/

빌드

```
$ chmod +x ./gradlew
$ ./gradlew clean build -Pprofile=dev
```

배포

```
$ docker build -t samals_be ./
$ docker run  -dp 8080:8080 --name samals_be samals_be
```

D. Mysql 서버 배포

Repository : Infra/db/

1. docker-compose.yml
2. samals.sql

```
$ docker-compose up –d
```

E. Besu Private Network 배포

```
$ docker run -d -p 8545:8545 hyperledger/besu:latest --
network=dev --miner-enabled --miner
coinbase=0x172aB7431BdBdE9E485b477bF0f434Ab7B219Bb6 --rpc-
http-enabled=true --rpc-http-host=0.0.0.0 --min-gas-price=0 --
rpc-http-api=ETH,NET,IBFT --host-allowlist="*" --rpc-http-
cors-origins="all”
```

F. Smart Contract 배포

smart-contract/src/

배포

```
$ truffle migrate --compile-all --network besu
```

3. CI/CD 설정

By **Jenkins**

A. Frontend 파이프라인

```
pipeline {
    agent any
    tools {
        nodejs "nodejs_16.16"
    }
    stages {
        stage ("********\n git pull \n********") {
            steps {
                git branch: 'FE', credentialsId: 'gitlab_ID_PW', url:
'{git clone 주소}'
            }
        }

        stage ("********\n npm install \n********") {
            steps {
                dir("./Client"){
                    sh "npm install --force"
                }
            }
        }

        stage ("********\n npm run build \n********") {
            steps {
                dir("Client"){
                    sh "npm run build"
                }
            }
        }

        stage ("********\n docker-compose \n********") {
```

```
            steps {

                dir("./Client"){
                    sh "docker-compose down"
                    sh "docker-compose build"
                    sh "docker-compose up -d"
                }
            }
        }
}
```

B. Backend 파이프라인

```
pipeline {
    agent any
    tools {
        gradle "gradle_7.5"
    }
    stages {
        stage ("******** '\n' git pull '\n' ********") {
            steps {
                git branch: 'BE', credentialsId: 'gitlab_ID_PW', url:
{git clone 주소}'
            }
        }

        stage ("******** \n Gradle Build \n ********") {
            steps {
                dir("backend/samals"){
                    sh "chmod +x ./gradlew"
                    sh "./gradlew clean build -Pprofile=dev"
                    // sh "ls -al"
                }
            }
```

```
        }
        stage ("******** \n docker build \n ********") {
            steps {
                dir("backend/samals"){
                    sh "docker build -t samals_be ./"
                }
            }
        }
        stage ("******** \n docker run \n ********") {
            steps {
                dir("backend/samals"){
                    sh "docker stop samals_be"
                    sh "docker rm samals_be"
                    sh "docker run  -dp 8080:8080 --name samals_be
samals_be"
                }
            }
        }
    }
}
```