B109 : 공간 109

삼성 청년 SW 아카데미 대전캠퍼스 7 기
특화 프로젝트 (2022.08.29 ~ 2022.10.07)

포팅 매뉴얼

박재현(팀장), 이원우, 이윤경, 임웅균, 주혜령, 황상윤

목차

공간 109 는 다가오는 Web3.0 시대의 새로운 자산으로 인식되는 디지털 자산 NFT 를 생성/거래하기 위한 플랫폼입니다. 공간 109 에서 내 작품을 NFT 로 만들고 갤러리에 전시, 판매해 보세요!

1. 프로젝트 기술 스택

1. 이슈 관리 : Jira
2. 형상 관리 : GitLab
3. 커뮤니케이션 : Mattermost, Notion, Discord
4. 개발 환경
    1) OS : Window 10
    2) IDE
        A. Visual Studio Code 1.70.0
        B. UI/UX : Figma
        C. 3D : Autodesk AutoCAD 21.0.52, Autodesk 3ds Max 19.0.1072
    3) Database : MySQL Workbench 8.0 CE
    4) Server : AWS EC2 (MobaXterm)
        A. Ubuntu 20.04.4 LTS
        B. Docker 20.10.18
        C. Jenkins 2.361.1
        D. Nginx 1.22.0
5. 상세 내용
    1) Backend
        A. Node 16.15.0
        B. Node Express 4.16.1
        C. Solidity 0.8.10 (solc-js)
        D. Truffle 5.5.28
        E. Ganache 7.4.0
    2) Frontend
        A. React 18.2.0
        B. React-three/fiber 8.7.3
        C. React-three/drei 9.29.1
        D. TypeScript 4.8.3
        E. Three.js 0.144.0
        F. Web3.js 1.7.5
        G. Ipfs 0.64.0

2. 빌드 시 사용되는 환경 변수 등 주요 내용

1. backend/.env 파일 추가

```
DB_HOST=172.17.0.2
DB_PORT=3306
DB_USER=root
DB_PASSWORD=ssafy
DB_SCHEMA=space109
BASE_THUMBNAIL_PATH=/tmp/space109/thumbnail
```

2. frontend/.env 파일 추가

```
REACT_APP_BACKEND_HOST=https://j7b109.p.ssafy.io/api
REACT_APP_PROJECT_ID=2F6WFaN05FMtbO93ODOLhwvE6EY
REACT_APP_PROJECT_SECRET_KEY=1998e51a7c7b5c7a15c51d493138c943
REACT_APP_SSAFY_NFT=0xd9530bdC6e3660F900D9eB12926ca6EEB262D4c4
REACT_APP_SALE_FACTORY=0x7eCE314eaDbEE830be70961E38814E9F73e066Bb
REACT_APP_SSAFY_TOKEN=0x0c54E456CE9E4501D2c43C38796ce3F06846C966
REACT_APP_SPACE_TOKEN=0x447a6898C21F0521AAc7975bA27210DF82D3AF6e
```

3. smart-contracts/.env 파일 추가

```
PRIVATEKEY=0xfdc731bd8b69fc427fe95812407392b9654f0b1f08b6881788be5857d5015f35
SSAFY_HTTP1=http://20.196.209.2:8545
SSAFY_HTTP2=http://52.141.42.92:8545
SSAFY_HTTP3=http://20.41.85.203:8545
```

4. 관리자 계정 활성화

```
su passwd root
```

5. 서버에 docker, docker-compose 설치

```
sudo apt update
sudo apt-get upgrade -y

sudo apt-get install -y ca-certificates \
    curl \
    software-properties-common \
    apt-transport-https \
    gnupg \
    lsb-release

sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg |
sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

echo \
    "deb [arch=$(dpkg --print-architecture)
signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt install -y docker-ce docker-ce-cli containerd.io docker-compose
```

6. 관리자가 아닌 일반 계정도 docker 실행 할 수 있도록 설정

```
# docker permission
$ sudo groupadd docker
$ sudo usermod -aG docker $USER

# docker-compose permission
$ sudo chmod 666 /var/run/docker.sock
```

7. 젠킨스 컨테이너 실행을 위한 docker-compose 작성 (~/docker-compose.yml)

```
version: '3'

services:
    jenkins:
        image: jenkins/jenkins:lts
        container_name: jenkins
        volumes:
            - /var/run/docker.sock:/var/run/docker.sock
            - /jenkins:/var/jenkins_home
            - /home/ubuntu:/home/ubuntu
        ports:
            - "9090:8080"
        privileged: true
        user: root
```

8. 젠킨스에서 설치해야하는 플러그인 목록

    A. GitLab

    B. Generic Webhook Trigger

    C. Gitlab API

    D. GitLab Authentication

    E. Docker Commons

    F. Docker API

    G. Docker

    H. Docker Commons

    I. Publish Over SSH

    J. nodeJS

    K. Mattermost Notification

9. 젠킨스 컨테이너 내부 접속 후 5 번 따라 docker 설치

```
docker exec -it jenkins bash
```

10. 프론트 컨테이너 내부에서 사용할 letsencrypt 파일 설정

```
sudo docker run -it --rm --name certbot -p 80:80 -v
"/home/ubuntu/certbot/conf:/etc/letsencrypt" -v
"/home/ubuntu/certbot/log:/var/log/letsencrypt" -v
"/home/ubuntu/certbot/www:/var/www/certbot" certbot/certbot certonly
```

3. 배포 과정

1. 사전에 작성한 docker-compose 파일 실행해 젠킨스 띄우기

```
sudo docker-compose up -d
```

2. http://{서버주소}:9090/로 접속해 젠킨스 설치 및 환경 세팅

    1) 젠킨스 비밀번호는 docker logs jenkins 로 확인

    2) 젠킨스 필수 플러그인 들 설치 후 freestyle 프로젝트 생성 (SPACE109)

    3) 해당 프로젝트에 깃랩 연결 후 빌드

    4) 서버 /jenkins/workspace/SPACE109 생성 확인

3. DB 로 사용할 MySQL 컨테이너 생성 및 세팅

    1) MySQL 컨테이너 생성

```
docker run --name space109_DB -e MYSQL_ROOT_PASSWORD=ssafy -d mysql
```

    2) Dump 넣기

```
docker cp {덤프파일 위치} space109_DB:/home
```

3) space109_DB 내부 접속

```
docker exec -it space109_DB bash
```

4) Dump import

```
mysql -hlocalhost -uroot -pssafy < /home/{덤프파일명}
```

4. /jenkins/workspace/SPACE109 에 docker 파일, .env 파일 작성

1) /jenkins/workspace/SPACE109/backend

A. Dockerfile (.env 파일은 2 번 참고)

```
FROM node:16

# 앱 디렉터리 생성
WORKDIR /var/jenkins_home/workspace/SPACE109/backend

# 앱 의존성 설치
# 가능한 경우(npm@5+) package.json과 package-lock.json을 모두 복사하기 위해
# 와일드카드를 사용
COPY package*.json ./
# 타임존 설정
ENV TZ=Asia/Seoul

RUN npm install --save --legacy-peer-deps
# 프로덕션을 위한 코드를 빌드하는 경우
# RUN npm ci --only=production

# 앱 소스 추가
COPY . .

EXPOSE 3000
CMD [ "node", "./bin/www" ]
```

2) /jenkins/workspace/SPACE109/frontend

A. Dockerfile (.env 파일은 2 번 참고)

```
FROM node:16 as build-stage

# 앱 디렉터리 생성
WORKDIR /var/jenkins_home/workspace/SPACE109/frontend

# 앱 의존성 설치
# 가능한 경우(npm@5+) package.json과 package-lock.json을 모두 복사하기 위해
# 와일드카드를 사용
COPY package*.json ./

RUN npm install --save --legacy-peer-deps
# 프로덕션을 위한 코드를 빌드하는 경우
# RUN npm ci --only=production

# 앱 소스 추가
COPY . .

RUN npm run build

FROM nginx:stable-alpine as production-stage
COPY --from=build-stage /var/jenkins_home/workspace/SPACE109/frontend/build /usr/share/n
ginx/html
COPY --from=build-stage /var/jenkins_home/workspace/SPACE109/frontend/deploy_conf/nginx.
conf /etc/nginx/conf.d/default.conf

EXPOSE 3001
CMD ["nginx", "-g", "daemon off;"]
```

B. deploy_conf/nginx.conf

```
upstream backend{
        ip_hash;
        server 172.17.0.4:8080;
}

map $http_upgrade $connection_upgrade {
   default upgrade;
   ''      close;
}

server {
        listen  80;
        server_name     j7b109.p.ssafy.io;
        location / {
                return 301 https://$host$request_uri;
        }
}
```

```
server {
    listen       443 ssl;
#    listen  [::]:443;
    server_name  j7b109.p.ssafy.io;

    access_log  /var/log/nginx/host.access.log  main;

   ssl_certificate /etc/letsencrypt/live/j7b109.p.ssafy.io/fullchain.pem;
   ssl_certificate_key /etc/letsencrypt/live/j7b109.p.ssafy.io/privkey.pem;
  # ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3 SSLv3;
  # ssl_ciphers ALL;

    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
        proxy_redirect off;
        try_files $uri.html $uri $uri/ /index.html;
        charset utf-8;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        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_set_header X-Nginx-Proxy true;
    }
```

```
        location /api {
        proxy_pass http://backend/;
        proxy_redirect     off;
        charset utf-8;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        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_set_header X-Nginx-Proxy true;
    }

    #error_page  404              /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }
```

3) /jenkins/workspace/SPACE109/smart-contracts - 2 번 참고해 .env 파일 작성

5. 젠킨스에 접속해 빌드 설정 마무리
    1) Build Steps - Excute shell 추가

```
docker image prune -a --force
mkdir -p /var/jenkins_home/images
cd /var/jenkins_home/workspace/SPACE109/frontend/
docker build -t react .
docker save react > /var/jenkins_home/images/react.tar

cd /var/jenkins_home/workspace/SPACE109/backend/
docker build -t node .
docker save node > /var/jenkins_home/images/node.tar

ls /var/jenkins_home/images
docker load < /var/jenkins_home/images/react.tar
docker load < /var/jenkins_home/images/node.tar

if (docker ps -a | grep "react"); then docker stop react; fi
if (docker ps -a | grep "node"); then docker stop node; docker rm node; fi

docker run -it -d --rm -p 80:80 -p 443:443 -v /home/ubuntu/certbot/conf:/etc/letsencryp
t/ -v /home/ubuntu/certbot/www:/var/www/certbot --name react react
echo "Run frontend"
docker run -it -d -p 8080:3000 -v /tmp/space109:/tmp/space109 --name node node
echo "Run backend"
```

6. 재빌드 후 **https://{서버주소}**로 접속 확인

4. 배포 특이사항

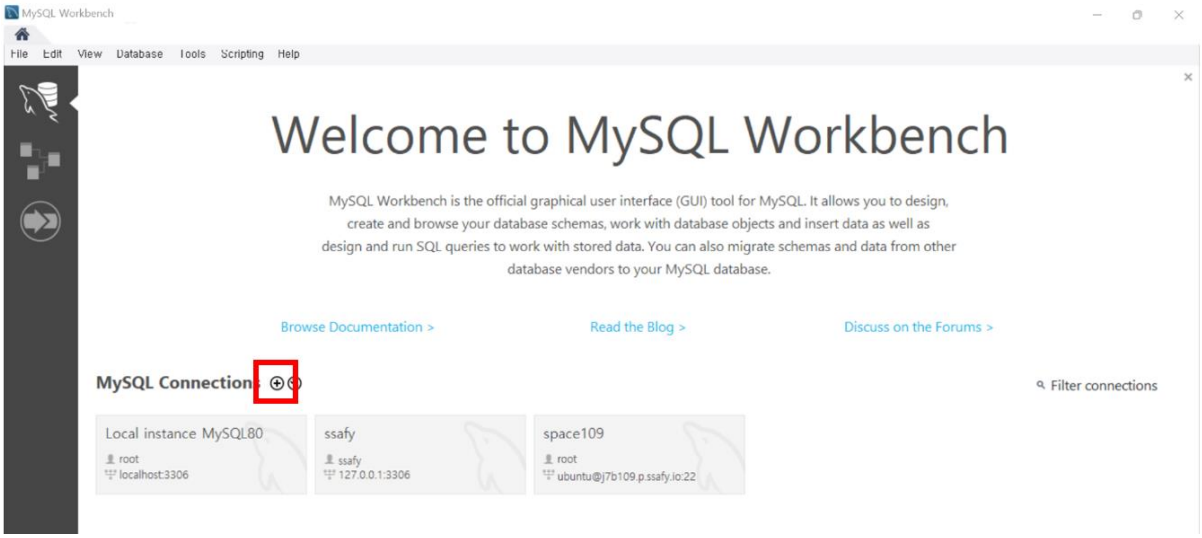
1. frontend
   react 컨테이너 내부에 있는 nginx 의 https 설정을 위해 사전에 certbot 을 이용해 생성한 암호 파일들을 매번 넣어 주어야했으나 컨테이너를 실행 하는 과정에서
   -v /home/ubuntu/certbot/conf:/etc/letsencrypt/
   -v /home/ubuntu/certbot/www:/var/www/certbot 옵션을 넣어 작업량 감소

2. backend
   클라이언트가 보낸 사진을 서버에 저장하는 로직이 있어 docker 실행시 –v /tmp/space109:/tmp/space109 옵션으로 컨테이너 밖에 저장

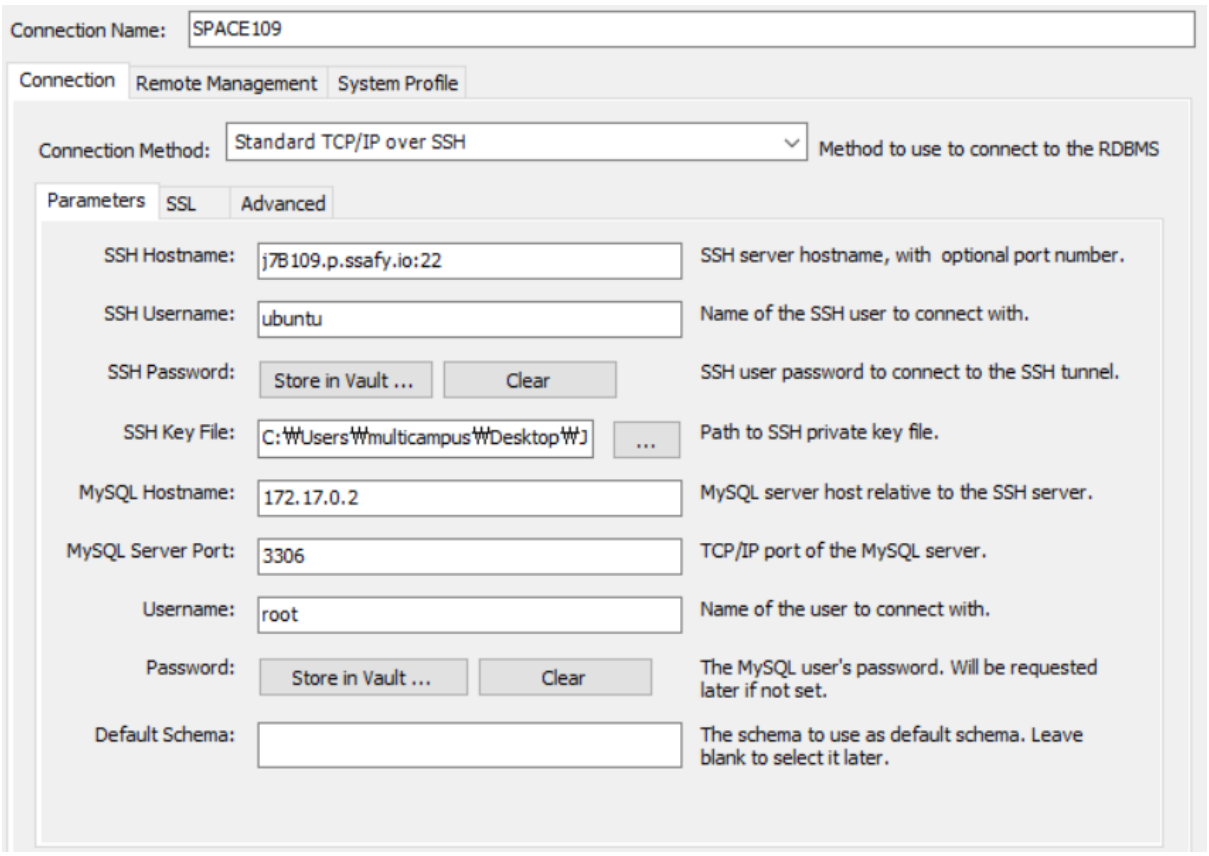
3. DB
   해당 포팅 매뉴얼은 DB 컨테이너가 react 컨테이너와 node 컨테이너보다 먼저 실행되어 계속 돌고있다는 가정하에 작성되었으므로 반드시 먼저 실행해야 함

5. DB 접속 정보 및 프로퍼티

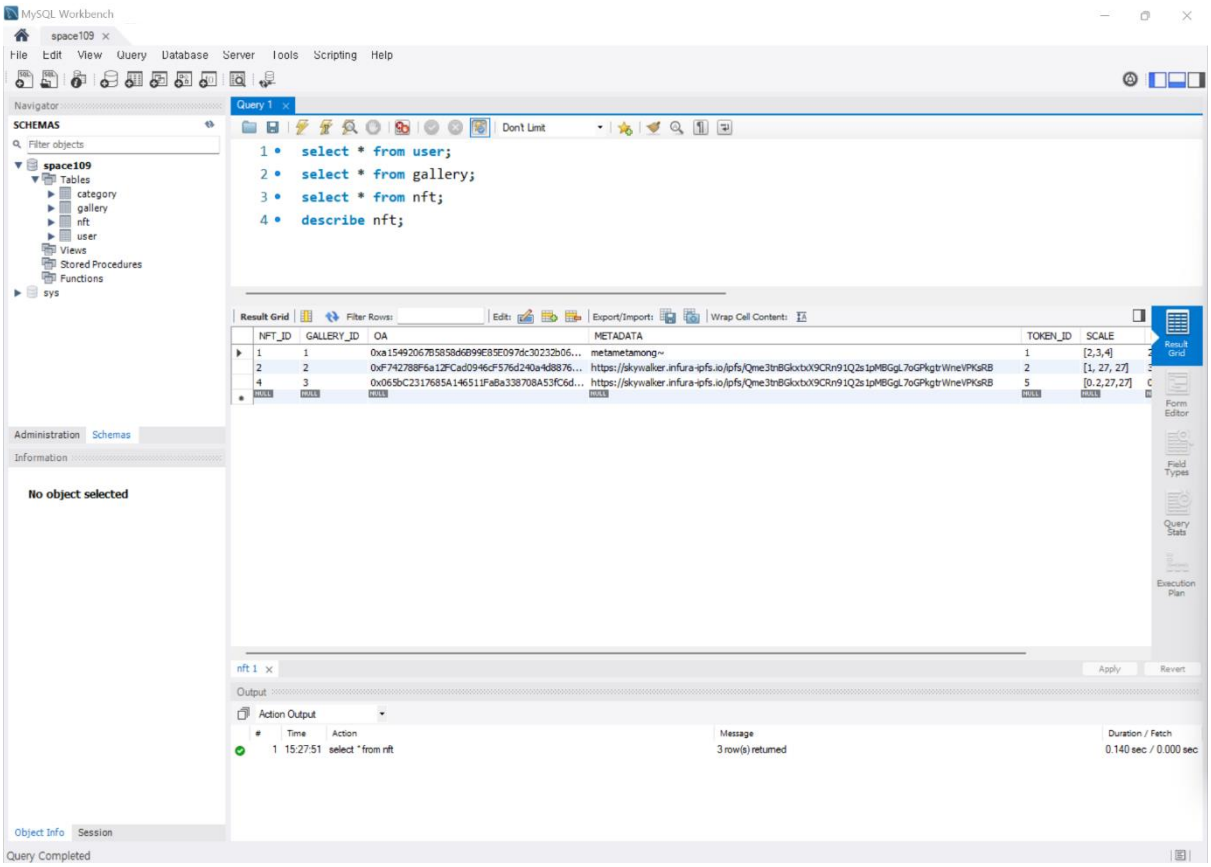
1. 새로운 커넥션 생성



2. 아래와 같이 설정



3. 접속 확인


MySQL Workbench
space109
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
space109
Tables
category
gallery
nft
user
Views
Stored Procedures
Functions
sys
Administration Schemas
Information
No object selected
Object Info Session
Query 1
Don't Limit
1 • select * from user;
2 • select * from gallery;
3 • select * from nft;
4 • describe nft;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:
NFT_ID GALLERY_ID OA METADATA TOKEN_ID SCALE
1 1 0xa15492067B5858d6B99E85E097dc30232b06... metametamong~ 1 [2,3,4]
2 2 0xF742788F6a12FCad0946cF576d240a4d8876... https://skywalker.infura-ipfs.io/ipfs/Qme3tnBGkxtxX9CRn91Q2s1pMBGgL7oGPkgtrWneVPKsRB 2 [1, 27, 27]
4 3 0x065bC2317685A146511FaBa338708A53fC6d... https://skywalker.infura-ipfs.io/ipfs/Qme3tnBGkxtxX9CRn91Q2s1pMBGgL7oGPkgtrWneVPKsRB 5 [0.2,27,27]
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan
nft 1
Apply Revert
Output
Action Output
Time Action Message Duration / Fetch
1 15:27:51 select * from nft 3 row(s) returned 0.140 sec / 0.000 sec
Query Completed