|  |
| --- |
|  |



|  |
| --- |
| **NFTicket** |
| **포팅 메뉴얼** |
|  |
|  |
|  |
|  |
|  |



목차

[I. 개요 2](#_Toc96072459)

[1. 프로젝트 개요 2](#_Toc96072460)

[2. 프로젝트 사용 도구 2](#_Toc96072461)

[3. 개발환경 2](#_Toc96072462)

[4. 외부 서비스 2](#_Toc96072463)

[5. Gitgnore 처리한 핵심 키들 3](#_Toc96072464)

[II. 빌드 3](#_Toc96072465)

[1. 환경변수 형태 3](#_Toc96072466)

[2. 빌드하기 5](#_Toc96072467)

[3. 배포하기 5](#_Toc96072468)

# 개요

## 프로젝트 개요

공연이 끝난 뒤, 누군가와의 혹은 그 순간 나만의 소중한 추억이 담긴 티켓이 사라지는 것이 아쉽지 않으셨나요? 모아두고 싶어도 아름답지 않아 결국 사라지는 추억들.

수집가처럼 앨범을 준비하고 일상을 기록하기는 힘들지만 마음만큼은, 그 순간의 추억을 간직하고 싶은 당신을 위해 준비했습니다.

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Slack, Mattermost

디자인 : Figma

UCC : 모바비

## 개발환경

VS Code : 1.66.0

Solc : 0.8.11

Webstorm : 2021.3.1

Node.js : 14.18.3

SERVER : AWS EC2 Ubuntu 20.04.4 LTS

DB : MariaDB, IPFS

## 외부 서비스

외부 서비스는 사용하지 않았습니다.

## Gitgnore 처리한 핵심 키들

Back/app/app/.env : MariaDB 접속 정보

# 빌드

# NGINX

## 환경변수 형태

1. 프론트엔드
2. 백엔드 : 없음
3. 솔리디티 : 없음

## 빌드하기

1. 프론트엔드

npm i

npm run build

1. 솔리디티

npm i

truffle compile

## 배포하기

1. Nginx 설정

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name nfticket.plus;

return 301 https://$server\_name$request\_uri;

}

server {

listen 443 ssl;

listen [::]:443 ssl;

server\_name nfticket.plus;

ssl\_certificate /etc/letsencrypt/live/nfticket.plus/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/nfticket.plus/privkey.pem;

root /var/www/html/build;

index index.html;

location / {

try\_files $uri $uri/ /index.html;

}

location /api-docs {

#rewrite /api-docs/(.\*) /$1 break;

proxy\_pass http://localhost:3000/api-docs;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /api/v1 {

rewrite /api/v1/(.\*) /$1 break;

proxy\_pass http://localhost:3000/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /ipfs {

rewrite /ipfs/(.\*) /$1 break;

proxy\_pass http://localhost:5001/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /showipfs {

rewrite /showipfs/(.\*) /$1 break;

proxy\_pass http://localhost:8080;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

}

이후 sudo service nginx start

1. 백엔드 배포

Dockercomposefiles.zip을 열어 각각의 폴더에서 docker-compose 명령을 실행

Portainer, mysql, ipfs는 그냥 docker-compose up -d를 실행

Nodejs는 docker-compose.yml 파일을 열어volumes 호스트 부분 경로를 적절히 수정 후

호스트 부분에 git clone 실행

1. Truffle을 통한 Dapp 배포

truffle-config.js에서 HDWalletProvider 선언 후

networks: {

…

development: {

host: "127.0.0.1", // Localhost (default: none)

port: 7545, // Standard Ethereum port (default: none)

network\_id: "\*", // Any network (default: none)

},

ssafy: {

provider: () => new HDWalletProvider("[SSAFY Wallet 개인키]", "http://20.196.209.2:8545"),

host: "20.196.209.2",

port: 8545,

network\_id: "\*",

from: "[SSAFY Wallet 지갑 주소]"

}

…

}

로 설정하여 SSAFY Network Provider를 설정

compilers: {

solc: {

version: "0.8.11",

settings: {

optimizer: {

enabled: true,

runs: 200

},

}

}

}

로 설정하여 용량이 큰 Contract도 컴파일 할 수 있도록 허용

1. 프론트엔드 배포 방법

빌드된 React 프로젝트 파일들을 /var/www/html/build 폴더에 복사하여 nginx 시작

# MySQL

## 환경변수

- MYSQL\_ROOT\_PASSWORD=DB비밀번호

- TZ=Asia/Seoul

## 빌드하기

docker-compose up -d

## 배포하기

1. Nginx 설정

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name nfticket.plus;

return 301 https://$server\_name$request\_uri;

}

server {

listen 443 ssl;

listen [::]:443 ssl;

server\_name nfticket.plus;

ssl\_certificate /etc/letsencrypt/live/nfticket.plus/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/nfticket.plus/privkey.pem;

root /var/www/html/build;

index index.html;

location / {

try\_files $uri $uri/ /index.html;

}

location /api-docs {

#rewrite /api-docs/(.\*) /$1 break;

proxy\_pass http://localhost:3000/api-docs;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /api/v1 {

rewrite /api/v1/(.\*) /$1 break;

proxy\_pass http://localhost:3000/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /ipfs {

rewrite /ipfs/(.\*) /$1 break;

proxy\_pass http://localhost:5001/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /showipfs {

rewrite /showipfs/(.\*) /$1 break;

proxy\_pass http://localhost:8080;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

}

이후 sudo service nginx start

1. 백엔드 배포

Dockercomposefiles.zip을 열어 각각의 폴더에서 docker-compose 명령을 실행

Portainer, mysql, ipfs는 그냥 docker-compose up -d를 실행

Nodejs는 docker-compose.yml 파일을 열어volumes 호스트 부분 경로를 적절히 수정 후

호스트 부분에 git clone 실행

1. Truffle을 통한 Dapp 배포

truffle-config.js에서 HDWalletProvider 선언 후

networks: {

…

development: {

host: "127.0.0.1", // Localhost (default: none)

port: 7545, // Standard Ethereum port (default: none)

network\_id: "\*", // Any network (default: none)

},

ssafy: {

provider: () => new HDWalletProvider("[SSAFY Wallet 개인키]", "http://20.196.209.2:8545"),

host: "20.196.209.2",

port: 8545,

network\_id: "\*",

from: "[SSAFY Wallet 지갑 주소]"

}

…

}

로 설정하여 SSAFY Network Provider를 설정

compilers: {

solc: {

version: "0.8.11",

settings: {

optimizer: {

enabled: true,

runs: 200

},

}

}

}

로 설정하여 용량이 큰 Contract도 컴파일 할 수 있도록 허용

1. 프론트엔드 배포 방법

빌드된 React 프로젝트 파일들을 /var/www/html/build 폴더에 복사하여 nginx 시작

# IPFS

## 환경변수 형태

1. 프론트엔드
2. 백엔드 : 없음
3. 솔리디티 : 없음

## 빌드하기

1. 프론트엔드

npm i

npm run build

1. 솔리디티

npm i

truffle compile

## 배포하기

1. Nginx 설정

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name nfticket.plus;

return 301 https://$server\_name$request\_uri;

}

server {

listen 443 ssl;

listen [::]:443 ssl;

server\_name nfticket.plus;

ssl\_certificate /etc/letsencrypt/live/nfticket.plus/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/nfticket.plus/privkey.pem;

root /var/www/html/build;

index index.html;

location / {

try\_files $uri $uri/ /index.html;

}

location /api-docs {

#rewrite /api-docs/(.\*) /$1 break;

proxy\_pass http://localhost:3000/api-docs;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /api/v1 {

rewrite /api/v1/(.\*) /$1 break;

proxy\_pass http://localhost:3000/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /ipfs {

rewrite /ipfs/(.\*) /$1 break;

proxy\_pass http://localhost:5001/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /showipfs {

rewrite /showipfs/(.\*) /$1 break;

proxy\_pass http://localhost:8080;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

}

이후 sudo service nginx start

1. 백엔드 배포

Dockercomposefiles.zip을 열어 각각의 폴더에서 docker-compose 명령을 실행

Portainer, mysql, ipfs는 그냥 docker-compose up -d를 실행

Nodejs는 docker-compose.yml 파일을 열어volumes 호스트 부분 경로를 적절히 수정 후

호스트 부분에 git clone 실행

1. Truffle을 통한 Dapp 배포

truffle-config.js에서 HDWalletProvider 선언 후

networks: {

…

development: {

host: "127.0.0.1", // Localhost (default: none)

port: 7545, // Standard Ethereum port (default: none)

network\_id: "\*", // Any network (default: none)

},

ssafy: {

provider: () => new HDWalletProvider("[SSAFY Wallet 개인키]", "http://20.196.209.2:8545"),

host: "20.196.209.2",

port: 8545,

network\_id: "\*",

from: "[SSAFY Wallet 지갑 주소]"

}

…

}

로 설정하여 SSAFY Network Provider를 설정

compilers: {

solc: {

version: "0.8.11",

settings: {

optimizer: {

enabled: true,

runs: 200

},

}

}

}

로 설정하여 용량이 큰 Contract도 컴파일 할 수 있도록 허용

1. 프론트엔드 배포 방법

빌드된 React 프로젝트 파일들을 /var/www/html/build 폴더에 복사하여 nginx 시작

# Node.JS

## 환경변수 형태

1. 프론트엔드
2. 백엔드 : 없음
3. 솔리디티 : 없음

## 빌드하기

1. 프론트엔드

npm i

npm run build

1. 솔리디티

npm i

truffle compile

## 배포하기

1. Nginx 설정

server {

listen 80 default\_server;

listen [::]:80 default\_server;

server\_name nfticket.plus;

return 301 https://$server\_name$request\_uri;

}

server {

listen 443 ssl;

listen [::]:443 ssl;

server\_name nfticket.plus;

ssl\_certificate /etc/letsencrypt/live/nfticket.plus/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/nfticket.plus/privkey.pem;

root /var/www/html/build;

index index.html;

location / {

try\_files $uri $uri/ /index.html;

}

location /api-docs {

#rewrite /api-docs/(.\*) /$1 break;

proxy\_pass http://localhost:3000/api-docs;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /api/v1 {

rewrite /api/v1/(.\*) /$1 break;

proxy\_pass http://localhost:3000/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /ipfs {

rewrite /ipfs/(.\*) /$1 break;

proxy\_pass http://localhost:5001/;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

location /showipfs {

rewrite /showipfs/(.\*) /$1 break;

proxy\_pass http://localhost:8080;

proxy\_redirect off;

proxy\_set\_header Host $host;

}

}

이후 sudo service nginx start

1. 백엔드 배포

Dockercomposefiles.zip을 열어 각각의 폴더에서 docker-compose 명령을 실행

Portainer, mysql, ipfs는 그냥 docker-compose up -d를 실행

Nodejs는 docker-compose.yml 파일을 열어volumes 호스트 부분 경로를 적절히 수정 후

호스트 부분에 git clone 실행

1. Truffle을 통한 Dapp 배포

truffle-config.js에서 HDWalletProvider 선언 후

networks: {

…

development: {

host: "127.0.0.1", // Localhost (default: none)

port: 7545, // Standard Ethereum port (default: none)

network\_id: "\*", // Any network (default: none)

},

ssafy: {

provider: () => new HDWalletProvider("[SSAFY Wallet 개인키]", "http://20.196.209.2:8545"),

host: "20.196.209.2",

port: 8545,

network\_id: "\*",

from: "[SSAFY Wallet 지갑 주소]"

}

…

}

로 설정하여 SSAFY Network Provider를 설정

compilers: {

solc: {

version: "0.8.11",

settings: {

optimizer: {

enabled: true,

runs: 200

},

}

}

}

로 설정하여 용량이 큰 Contract도 컴파일 할 수 있도록 허용

1. 프론트엔드 배포 방법

빌드된 React 프로젝트 파일들을 /var/www/html/build 폴더에 복사하여 nginx 시작