NFTicket

NFTicket

포팅 메뉴얼

목차

I. 개요

1. 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.

2. 프로젝트 사용 도구

이슈 관리 : JIRA
형상 관리 : Gitlab
커뮤니케이션 : Notion, Slack, Mattermost
디자인 : Figma
UCC : 모바비

3. 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

4. 외부 서비스

외부 서비스는 사용하지 않았습니다.

5. Gitgnore 처리한 핵심 키들

Back/app/app/.env : MariaDB 접속 정보

II. 빌드

A. NGINX

6. 환경변수 형태

1) 프론트엔드
2) 백엔드 : 없음
3) 솔리디티 : 없음

7. 빌드하기

1) 프론트엔드

   npm i

   npm run build

2) 솔리디티

   npm i

   truffle compile

8. 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

2) 백엔드 배포

Dockercomposefiles.zip 을 열어 각각의 폴더에서 docker-compose 명령을 실행
Portainer, mysql, ipfs 는 그냥 docker-compose up -d 를 실행
Nodejs 는 docker-compose.yml 파일을 열어 volumes 호스트 부분 경로를 적절히 수정 후 호스트 부분에 git clone 실행

3) Truffle 을 통한 Dapp 배포

truffle-config.js 에서 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 Provider 를 설정

```
compilers: {
    solc: {
      version: "0.8.11",
```

```
    settings: {
      optimizer: {
        enabled: true,
        runs: 200
      },
    }
  }
}
```

로 설정하여 용량이 큰 Contract 도 컴파일 할 수 있도록 허용

4) 프론트엔드 배포 방법

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

B. MySQL

1. 환경변수

- MYSQL_ROOT_PASSWORD=DB 비밀번호
- TZ=Asia/Seoul

2. 빌드하기

docker-compose up -d

3. 배포하기

5) 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

6) 백엔드 배포

Dockercomposefiles.zip 을 열어 각각의 폴더에서 docker-compose 명령을 실행
Portainer, mysql, ipfs 는 그냥 docker-compose up -d 를 실행
Nodejs 는 docker-compose.yml 파일을 열어 volumes 호스트 부분 경로를 적절히 수정 후 호스트 부분에 git clone 실행

7) Truffle 을 통한 Dapp 배포

truffle-config.js 에서 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 Provider 를 설정

```
compilers: {
    solc: {
      version: "0.8.11",
      settings: {
        optimizer: {
          enabled: true,
          runs: 200
        },
      }
    }
  }
```

로 설정하여 용량이 큰 Contract 도 컴파일 할 수 있도록 허용

8) 프론트엔드 배포 방법

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

C. IPFS

1. 환경변수 형태

4) 프론트엔드
5) 백엔드 : 없음
6) 솔리디티 : 없음

2. 빌드하기

3) 프론트엔드
   npm i
   npm run build
4) 솔리디티
   npm i
   truffle compile

3. 배포하기

9) 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

10) 백엔드 배포

Dockercomposefiles.zip 을 열어 각각의 폴더에서 docker-compose 명령을 실행
Portainer, mysql, ipfs 는 그냥 docker-compose up -d 를 실행
Nodejs 는 docker-compose.yml 파일을 열어 volumes 호스트 부분 경로를 적절히 수정 후 호스트 부분에 git clone 실행

11) Truffle 을 통한 Dapp 배포

truffle-config.js 에서 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 Provider 를 설정

```
compilers: {
    solc: {
      version: "0.8.11",
      settings: {
        optimizer: {
          enabled: true,
          runs: 200
        },
      }
    }
  }
```

로 설정하여 용량이 큰 Contract 도 컴파일 할 수 있도록 허용

12) 프론트엔드 배포 방법

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

D. Node.JS

1. 환경변수 형태

7) 프론트엔드
8) 백엔드 : 없음
9) 솔리디티 : 없음

2. 빌드하기

5) 프론트엔드
npm i
npm run build

6) 솔리디티
npm i
truffle compile

3. 배포하기

13) 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

14) 백엔드 배포

Dockercomposefiles.zip 을 열어 각각의 폴더에서 docker-compose 명령을 실행
Portainer, mysql, ipfs 는 그냥 docker-compose up -d 를 실행
Nodejs 는 docker-compose.yml 파일을 열어 volumes 호스트 부분 경로를 적절히 수정 후 호스트 부분에 git clone 실행

15) Truffle 을 통한 Dapp 배포

truffle-config.js 에서 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 Provider 를 설정

```
compilers: {
    solc: {
      version: "0.8.11",
      settings: {
        optimizer: {
          enabled: true,
          runs: 200
        },
      }
    }
  }
```

로 설정하여 용량이 큰 Contract 도 컴파일 할 수 있도록 허용

16) 프론트엔드 배포 방법

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