인수인계 문서

코드별 설명

타입스크립트 사용이유.

클립뿐만 아니라 카이카스나 키파일 등 다양한 클레이튼 관련 계정 툴들이 존재함. 따라서Strategy패턴을 이용하여 NftTransaction 클래스(혹은 인터페이스)를 Strategy로, KlipNftTransaction나 KaikasNftTransaction를 ConcreteStrategy로 구현하여 유연하게 구성하고자 하였음.

NftWalletInfo.ts

```
export interface NftWalletInfo{
    getWallet():Promise<string>;
    setWallet(wallet:string):void;

    getTokenInfoURL(tokenId: number):Promise<string>;

    getAccountTokenBalance():Promise<number>;
    getTokenIdOfOwnerByIndex(index:number):Promise<number>;

    getAllTokenOfOwner(): Promise<number[]>;
}
```

로그인, 조회 관련 인터페이스. getWallet, setWallet을 제외하면 KIP17 interface를 이용한 메서드들이다.

getWallet():Promise<string>;

유저의 지갑 주소를 가져오는 함수.

setWallet(wallet:string):void;
유저의 지갑 주소를 직접 입력 할 수 있는 함수. 따로 클레이튼 주소로 로그인 하지 않고 db에 저장된 유저 지갑 주소를 이용할 수 있게 하기 위해 생성

getTokenInfoURL(tokenId: number):Promise<string>;
토큰의 속성, 이미지 주소 등의 정보를 가져올 수 있는 함수

getAccountTokenBalance():Promise<number>;

유저가 보유하고 있는 토큰 갯수를 불러오는 함수.

getTokenIdOfOwnerByIndex(index:number):Promise<number>;
유저가 보유하고 있는 토큰 id를 인덱스로 불러오는 함수.

getAllTokenOfOwner(): Promise<number[]>;

유저가 보유하고 있는 모든 토큰 id를 불러오는 함수.

getAccountTokenBalance로 갯수를 가져오고 getTokenIdOfOwnerByIndex로 갯수만큼 반복문을 돌려 구현할수 있다.

NftMart.ts

```
export interface NftMart{

    setApprove(): Promise<any>;
    cancelApprove(): Promise<any>;

    setSaleOnCont(tokenId:number,price:number):Promise<any>;
```

```
    removeSaleOnCont(tokenId:number):Promise<any>;

    getTokenPrice(tokenId:number):Promise<number>;

    buyToken(tokenId:number,price:number):Promise<any>;

}
```

마켓, 거래관련 인터페이스. setApprove, cancelApprove는 KIP17 인터페이스이고 다른 함수들은 SoundgramAlbumSales.sol의 함수들을 이용한다.

setApprove(): Promise<any>;

토큰 거래 권한을 SoundgramAlbumSales컨트랙트에 부여하는 함수

cancelApprove(): Promise<any>;
SoundgramAlbumSales컨트랙트에 부여된 토큰 거래 권한을 취소하는 함수

setSaleOnCont(tokenId:number,price:number):Promise<any>;

토큰을 판매등록하는 함수

removeSaleOnCont(tokenId:number):Promise<any>;
토큰을 판매 취소하는 함수

getTokenPrice(tokenId:number):Promise<number>;
판매등록된 토큰의 가격을 조회하는 함수

buyToken(tokenId:number,price:number):Promise<any>;

판매등록된 토큰을 구매하는 함수

KlipNftTransaction .ts

```
import * as WalletInfo from './NftWalletInfo';
import * as Mart from './NftMart';
import { prepare, request, getResult, getCardList, web2app } from 'klip-sdk';
import * as Caver from "caver-js-ext-kas";

class KlipNftTransaction implements WalletInfo.NftWalletInfo, Mart.NftMart {

    bappName = '(주)사운드그램';
    userWallet: string;

    //TODO 하드코딩 하지말고 파일이나 기타 설정파일에서 관리하며 읽어오기-자주 변화가 있을수 있으니
    klipTokenAddress = '0x8817B4883fB534E005613b5f36ad3960906C604f';
    salesContractAddress = '0xd6646EA5152b9d8e244a4d24027D1D4f5Abc0Ee5';
    config = {
        accessKeyId: "KASKE371R6CEN79Y0P1BVHIW",
        secretAccessKey: "-olZWq3qoviWnTNjf7X4TmdSKkdaZnrXAEuEnFFZ"
    };

    DEPLOYED_ABI=[{"constant":true,"inputs":[{"name":"interfaceId","type":"bytes4"}],"name":"supportsInterface","outputs":[{"name":"",
    DEPLOYED_ABI_ALBUMSALES= [{"constant":true,"inputs":[],"name":"nftAddress","outputs":[{"name":"","type":"address"}],"payable":fals
    cav = new Caver(8217, this.config.accessKeyId, this.config.secretAccessKey);
    tokenContract = new this.cav.klay.Contract(this.DEPLOYED_ABI, this.klipTokenAddress);
    salesContract = new this.cav.klay.Contract(this.DEPLOYED_ABI_ALBUMSALES, this.salesContractAddress);

    constructor(){

    }
이하생략
```

klipTokenAddress - 클립파트너스에서 제공하는 클립 토큰의 컨트랙트 주소

salesContractAddress - 클립 거래시 사용되는 SoundgramAlbumSales.sol 을 배포한 컨트랙트 주소.

config - https://www.klaytnapi.com/ko/landing/main에서 제공하는 프로바이더를 사용하기위한 엑세스 키
DEPLOYED_ABI- KIP17 인터페이스의 ABI

getWallet 함수 호출시 객체내에 저장된 userWallet이 없을시 자동으로 클립 로그인을 호출하는 방식으로 구현하였다.

클립의 api 호출 구조

prepare-request-result 3단계

prepare-실행시킬 내용 준비(abi 및 파라메터 설정)후 리퀘스트 키 얻어오기.

request-리퀘스트 키를 실행요청

result - 클립에 인증 요청이 가고 실행결과 반환

KaikasNftTransaction .ts

```
import * as WalletInfo from './NftWalletInfo';
import * as Mart from './NftMart';
import * as Caver from "caver-js-ext-kas";

class KaikasNftTransaction implements WalletInfo.NftWalletInfo, Mart.NftMart {

    userWallet: string;

    klipTokenAddress = '0x8817B4883fB534E005613b5f36ad3960906C604f';
    salesContractAddress = '0xd6646EA5152b9d8e244a4d24027D1D4f5Abc0Ee5';
    config = {
        accessKeyId: "KASKE371R6CEN79Y0P1BVHIW",
        secretAccessKey: "-olZWq3qoviWnTNjf7X4TmdSKkdaZnrXAEuEnFFZ"
    };

    DEPLOYED_ABI=[{"constant":true,"inputs":[{"name":"interfaceId","type":"bytes4"}],"name":"supportsInterface","outputs":[{"name":"",
    DEPLOYED_ABI_ALBUMSALES= [{"constant":true,"inputs":[],"name":"nftAddress","outputs":[{"name":"","type":"address"}],"payable":fals
    cav = new Caver(8217, this.config.accessKeyId, this.config.secretAccessKey);
    tokenContract = new this.cav.klay.Contract(this.DEPLOYED_ABI, this.klipTokenAddress);
    salesContract = new this.cav.klay.Contract(this.DEPLOYED_ABI_ALBUMSALES, this.salesContractAddress);

    constructor(window:any){
        this.__init__(window);


    }
    async __init__(window){
      typeof window.klaytn !== 'undefined';
      const accounts = await window.klaytn.enable();
      this.userWallet = accounts[0];
      console.log("get account",this.userWallet);

      //카이카스 계정 수정 추적
      window.klaytn.on('accountsChanged', function(accounts) {
          this.userWallet=accounts[0];
        console.log("accountsChanged",accounts);
      });

      if (typeof window.klaytn !== 'undefined') {
        this.cav = new Caver(window.klaytn);
        const provider = window['klaytn']
        this.cav.setProvider(provider);
        console.log("TestKaikas setProvider",provider);
      }
  }

이하생략
```

※ KaikasNftTransaction는 전체적으로 테스트 되지 않음. 사용 전 테스트 필요

카이카스를 이용한 케이버 메서드를 실행시킬때 카이카스 프로바이더를 제공하지 않으면 오류가 났었음.

카이카스 provider제공을 위해선 카이카스 확장 설치시 브라우저 window에 들어있는 객체를 이용해 프로바이더를 가져와야해서 klip과 달리 객체 생성시 window를 전달해주어야한다.

tsconfig.json

```
{
  "compilerOptions": {
    /* Visit https://aka.ms/tsconfig to read more about this file */

    /* Projects */
    // "incremental": true,                              /* Save .tsbuildinfo files to allow for incremental compilation of projects.
    // "composite": true,                                /* Enable constraints that allow a TypeScript project to be used with project
    // "tsBuildInfoFile": "./.tsbuildinfo",              /* Specify the path to .tsbuildinfo incremental compilation file. */
    // "disableSourceOfProjectReferenceRedirect": true,  /* Disable preferring source files instead of declaration files when referenc
    // "disableSolutionSearching": true,                 /* Opt a project out of multi-project reference checking when editing. */
    // "disableReferencedProjectLoad": true,             /* Reduce the number of projects loaded automatically by TypeScript. */

    /* Language and Environment */
    "target": "es2016",                                  /* Set the JavaScript language version for emitted JavaScript and include com
    // "lib": [],                                        /* Specify a set of bundled library declaration files that describe the targe
    // "jsx": "preserve",                                /* Specify what JSX code is generated. */
    // "experimentalDecorators": true,                   /* Enable experimental support for TC39 stage 2 draft decorators. */
    // "emitDecoratorMetadata": true,                    /* Emit design-type metadata for decorated declarations in source files. */
    // "jsxFactory": "",                                 /* Specify the JSX factory function used when targeting React JSX emit, e.g.
    // "jsxFragmentFactory": "",                         /* Specify the JSX Fragment reference used for fragments when targeting React
    // "jsxImportSource": "",                            /* Specify module specifier used to import the JSX factory functions when usi
    // "reactNamespace": "",                             /* Specify the object invoked for 'createElement'. This only applies when tar
    // "noLib": true,                                    /* Disable including any library files, including the default lib.d.ts. */
    // "useDefineForClassFields": true,                  /* Emit ECMAScript-standard-compliant class fields. */
    // "moduleDetection": "auto",                        /* Control what method is used to detect module-format JS files. */

    /* Modules */
    "module": "commonjs",                                /* Specify what module code is generated. */
    // "rootDir": "./",                                  /* Specify the root folder within your source files. */
    // "moduleResolution": "node",                       /* Specify how TypeScript looks up a file from a given module specifier. */
    // "baseUrl": "./",                                  /* Specify the base directory to resolve non-relative module names. */
    // "paths": {},                                      /* Specify a set of entries that re-map imports to additional lookup location
    // "rootDirs": [],                                   /* Allow multiple folders to be treated as one when resolving modules. */
    // "typeRoots": [],                                  /* Specify multiple folders that act like './node_modules/@types'. */
    // "types": [],                                      /* Specify type package names to be included without being referenced in a so
    // "allowUmdGlobalAccess": true,                     /* Allow accessing UMD globals from modules. */
    // "moduleSuffixes": [],                             /* List of file name suffixes to search when resolving a module. */
    // "resolveJsonModule": true,                        /* Enable importing .json files. */
    // "noResolve": true,                                /* Disallow 'import's, 'require's or '<reference>'s from expanding the number

    /* JavaScript Support */
    // "allowJs": true,                                  /* Allow JavaScript files to be a part of your program. Use the 'checkJS' opt
    // "checkJs": true,                                  /* Enable error reporting in type-checked JavaScript files. */
    // "maxNodeModuleJsDepth": 1,                        /* Specify the maximum folder depth used for checking JavaScript files from '

    /* Emit */
    // "declaration": true,                              /* Generate .d.ts files from TypeScript and JavaScript files in your project.
    // "declarationMap": true,                           /* Create sourcemaps for d.ts files. */
    // "emitDeclarationOnly": true,                      /* Only output d.ts files and not JavaScript files. */
    // "sourceMap": true,                                /* Create source map files for emitted JavaScript files. */
    // "outFile": "./",                                  /* Specify a file that bundles all outputs into one JavaScript file. If 'decl
    // "outDir": "./",                                   /* Specify an output folder for all emitted files. */
    // "removeComments": true,                           /* Disable emitting comments. */
    // "noEmit": true,                                   /* Disable emitting files from a compilation. */
    // "importHelpers": true,                            /* Allow importing helper functions from tslib once per project, instead of i
    // "importsNotUsedAsValues": "remove",               /* Specify emit/checking behavior for imports that are only used for types. *
    // "downlevelIteration": true,                       /* Emit more compliant, but verbose and less performant JavaScript for iterat
    // "sourceRoot": "",                                 /* Specify the root path for debuggers to find the reference source code. */
    // "mapRoot": "",                                    /* Specify the location where debugger should locate map files instead of gen
    // "inlineSourceMap": true,                          /* Include sourcemap files inside the emitted JavaScript. */
    // "inlineSources": true,                            /* Include source code in the sourcemaps inside the emitted JavaScript. */
    // "emitBOM": true,                                  /* Emit a UTF-8 Byte Order Mark (BOM) in the beginning of output files. */
    // "newLine": "crlf",                                /* Set the newline character for emitting files. */
    // "stripInternal": true,                            /* Disable emitting declarations that have '@internal' in their JSDoc comment
    // "noEmitHelpers": true,                            /* Disable generating custom helper functions like '__extends' in compiled ou
    // "noEmitOnError": true,                            /* Disable emitting files if any type checking errors are reported. */
    // "preserveConstEnums": true,                       /* Disable erasing 'const enum' declarations in generated code. */
    // "declarationDir": "./",                           /* Specify the output directory for generated declaration files. */
    // "preserveValueImports": true,                     /* Preserve unused imported values in the JavaScript output that would otherw

    /* Interop Constraints */
    // "isolatedModules": true,                          /* Ensure that each file can be safely transpiled without relying on other im
    // "allowSyntheticDefaultImports": true,             /* Allow 'import x from y' when a module doesn't have a default export. */
    "esModuleInterop": true,                             /* Emit additional JavaScript to ease support for importing CommonJS modules.
    // "preserveSymlinks": true,                         /* Disable resolving symlinks to their realpath. This correlates to the same
    "forceConsistentCasingInFileNames": true,            /* Ensure that casing is correct in imports. */

    /* Type Checking */
    //"strict": true,                                     /* Enable all strict type-checking options. */
    // "noImplicitAny": true,                            /* Enable error reporting for expressions and declarations with an implied 'a
    // "strictNullChecks": true,                         /* When type checking, take into account 'null' and 'undefined'. */
    // "strictFunctionTypes": true,                      /* When assigning functions, check to ensure parameters and the return values
    // "strictBindCallApply": true,                      /* Check that the arguments for 'bind', 'call', and 'apply' methods match the
    // "strictPropertyInitialization": true,             /* Check for class properties that are declared but not set in the constructo
    // "noImplicitThis": true,                           /* Enable error reporting when 'this' is given the type 'any'. */
```

```
    // "useUnknownInCatchVariables": true,               /* Default catch clause variables as 'unknown' instead of 'any'. */
    // "alwaysStrict": true,                             /* Ensure 'use strict' is always emitted. */
    // "noUnusedLocals": true,                           /* Enable error reporting when local variables aren't read. */
    // "noUnusedParameters": true,                       /* Raise an error when a function parameter isn't read. */
    // "exactOptionalPropertyTypes": true,               /* Interpret optional property types as written, rather than adding 'undefine
    // "noImplicitReturns": true,                        /* Enable error reporting for codepaths that do not explicitly return in a fu
    // "noFallthroughCasesInSwitch": true,               /* Enable error reporting for fallthrough cases in switch statements. */
    // "noUncheckedIndexedAccess": true,                 /* Add 'undefined' to a type when accessed using an index. */
    // "noImplicitOverride": true,                       /* Ensure overriding members in derived classes are marked with an override m
    // "noPropertyAccessFromIndexSignature": true,       /* Enforces using indexed accessors for keys declared using an indexed type.
    // "allowUnusedLabels": true,                        /* Disable error reporting for unused labels. */
    // "allowUnreachableCode": true,                     /* Disable error reporting for unreachable code. */

    /* Completeness */
    // "skipDefaultLibCheck": true,                      /* Skip type checking .d.ts files that are included with TypeScript. */
    "skipLibCheck": true                                 /* Skip type checking all .d.ts files. */
  }
}
```

타입스크립트를 자바스크립트로 변환하기 위한 설정파일

작업환경이 다양해서 require와 import가 섞여있어 "target": "es2016", "module": "commonjs"등의 설정이 중요하다

CardMinting.js

※ 참고용 용어

클립카드=토큰=nft

민팅=토큰 발행

```
saveAccessToken()
getAllCard()
CardImageUpload()
KlipCardMinting()
```

saveAccessToken()은 카드 민팅시 필요한 엑세스키를 발급받기 위해 사용되는 함수.

24시간마다 엑세스 키가 만료되므로 24시간마다 갱신이 필요하다.

현재 자동화가 되어있지 않아 24시간마다 수동으로 실행하여 직접 security.authorization에 넣어 갱신하여 사용중

파일안에 넣고 있으므로 Minting.js에서 파일을 읽어 사용하던가 자동으로 security.js를 수정하도록 수정한후 24시간마다 자동 실행되도록 해야한다.

CardImageUpload() 카드민팅시 사용할 이미지를 업로드하고 해당 이미지에 접근가능한 경로를 받환받을수 있는 함수. 반드시 이 함수를 통해 얻은 이미지 url이 아니더라도 누구나 접근 가능한 url을 클립민팅시 넘겨주면 민팅이 가능하므로 필수는 아니다.

KlipCardMinting()은 민팅을 위한 함수인데 이 클래스에서의 함수는 테스트용으로만 사용되고 실제사용은 후술할 Minting.js에서 사용한다.

여기부턴 D3.0_dev내부의 파일(disk페이지와 연동하기 위해서 추가된 market1.0에는 없거나 다른 코드이다.)

/var/www/html/disk/D0.3_dev/api/mint.php

```
<?php
  header('Content-type: application/json');
    require_once("db_connection.php");

    $user_id = $_REQUEST["user_id"];
    $imageurl = $_REQUEST["imageurl"];
    $userAddress = $_REQUEST["userAddress"];
    $album_id = $_REQUEST["album_id"];
    $album_name = $_REQUEST["album_name"];
    $chipdisc_model = $_REQUEST["chipdisc_model"];
    $product_code = $_REQUEST["product_code"];
    $artist = $_REQUEST["artist"];
```

```
    $edition_no = $_REQUEST["edition_no"];
    $chip_code = $_REQUEST["chip_code"];

    //db check
    $query_check_mintable = "SELECT mintable FROM APIServer_nfckey WHERE id=$chip_code";
    $mintable = sql_query($query_check_mintable)->fetch_assoc()["mintable"];

    $query_token_id = "SELECT MAX(token_id) as token_id FROM APIServer_nft";
    $token_id = sql_query($query_token_id)->fetch_assoc()["token_id"] + 1;

    $query_wallet_id = "SELECT id FROM APIServer_user_wallet WHERE wallet_address = '$userAddress'";
    $wallet_id = sql_query($query_wallet_id)->fetch_assoc()["id"];


    if($mintable != "1") $return_code="NFT 발행 불가";
    else {
      $return_code="발행 성공";

      chdir("../js/");
      exec("node Minting.js --imageurl=$imageurl --userAddress=$userAddress --album_name=\"$album_name\" --chipdisc_model=$chipdisc_mo

      $http_status = $output[0];
      $hash = $output[1];

      $query_insert_nft = "INSERT INTO APIServer_nft (nfckey_id, token_id, owner_wallet_id, mint_date, update_date, mint_hash) values(
      sql_query($query_insert_nft);
      //쿼리 에러 어케잡어

      $query_update_nfckey = "UPDATE APIServer_nfckey SET mintable=2, nft_token_id = $token_id WHERE id=$chip_code";
      sql_query($query_update_nfckey);
      //쿼리 에러 어케잡어

      $query_insert_history = "INSERT INTO APIServer_nft_history (album_id, nfckey_id, nft_token_id, action_user_id, dst_user_id, dst_
                               values($album_id, $chip_code, $token_id, $user_id, $user_id, '$userAddress', 1, NOW())";
      sql_query($query_insert_history);
    }

    $res = array(
        "return_code" => $return_code,
        "mintable"=> $wallet_id,
        "outputs"=> $output,
        "query_mintable" => $query_wallet_id,
        "query_insert" => $query_insert_nft,
        "query_update" => $query_update_nfckey,
        "query_history" => $query_insert_history
    );

    echo json_encode($res);
?>
```

```
exec("node Minting.js --imageurl=$imageurl --userAddress=$userAddress --album_name=\"$album_name\" --chipdisc_model=$chipdisc_model --
```

exec를 사용. php에서 노드를 이용하여 인자를 전달하며 minting.js를 실행시킨다.

/var/www/html/disk/D0.3_dev/js/Minting.js

```
const security = require("./security");
const axios = require('axios');
const argv = require('minimist')(process.argv.slice(2),{string:['album_name', 'userAddress', 'edition_no', 'chip_code', 'artist']});

function KlipCardMinting(argv) {
    //띄어쓰기 있는 앨범 이름은 _로 치환하여 받은 후 다시 띄어쓰기로 바꿔줌
    var album_name = argv.album_name;
    album_name = album_name.replace(/_/g, " ");
    var artist_name = argv.artist;
    artist_name = artist_name.replace(/_/g, " ");

//*
    axios({
        url: 'https://api.klipwallet.com/v2/wallet/mint',
        method: 'post',
        headers: {
                authorization: security.authorization ,
                'Content-Type': 'application/json'
                },
        data: {
            "pin": security.pin,
            "to_address": [argv.userAddress],
            "contract_address": security.contract_address,
            //"name": security.service_name,
            "name": album_name ,
```

```
            "description" : "klip minting test 1",
            "image": argv.imageurl,
            "sendable" : false,
            "send_friendly_only" : false,
            "attributes" : [
                {
                    "trait_type": "Chipdisc Model",
                    "value": argv.chipdisc_model
                },
                {
                    "trait_type": "Product Code",
                    "value": argv.product_code
                },
                {
                    "trait_type": "Artist",
                    "value":artist_name
                },
                {
                    "trait_type": "Edition no",
                    "value": argv.edition_no
                },
                {
                    "trait_type": "Chip code",
                    "value": argv.chip_code
                }
            ]
            }
        })
        .then(function a(response) {
            //console.log(response["status"]);
            //console.log(response["data"]["hash"]);
            console.log(response);
        })
        .catch(function (error) {
            console.log(error);
        });
//*/
}
KlipCardMinting(argv);
```

위 php에서 요청을하면 실행되는 민팅함수. 인자에 띄어쓰기가 있을시 minimist라이브러리가 두개의 단어로 파싱을 하여 전처리과정이 들어갔다.

/var/www/html/disk/D0.3_dev/js/webpack.config.js

```
var path = require('path');

module.exports = {
  mode: 'none',
  entry: './klipWebpack.js',
  output: {
    path: path.resolve(__dirname, ''),
    filename: 'KlipTransactionBundle.js',
  },
  node: {
      fs:'empty',
  },
};
```

/var/www/html/disk/D0.3_dev/js/klipWebpack.js

```
window.KlipNftTransaction = require('./KlipNftTransaction.js');
```

node로 서버를 실행시키는 market1.0과 달리 D3.0_dev은 아파치 서버를 사용하고 /var/www/html/disk/D0.3_dev/nfc_tot.php에서 스크립트를 직접 주입하므로 노드모듈안에 있는 라이브러리를 사용하지 못하고 번들링하여 전역변수인 window안에 정보를 넣고 사용해야 한다.

위코드에선 klipWebpack.js를 번들링하여 사용하므로 window에 바로 KlipNftTransaction스크립트가 넣어진다.

KlipNftTransaction.ts에 변경사항이 있다면 번들링 사용전 tsc KlipNftTransaction.ts로 컴파일하여KlipNftTransaction.js파일을 생성해야한다

/var/www/html/disk/D0.3_dev/js/nft_common.js

KlipNftTransaction.js를 이용한 동작구현이 이루어지는 코드 위치이다.

해야할일

1. 엑세스 키 자동 갱신
2. SoundgramAlbumSales.sol 보안 체크
3. NftMart 관련 함수 구현에서 반환값으로 트랜젝션 해쉬값 및 기타 필요한 정보를 반환하도록 형식 정하기. 지금은 클립의 반환값을 그대로 돌려주고 있음.
4. (D0.3_dev에 존재하는 코드는 Strategy를 사용하지 않고 KlipNftTransaction 하나로만 구현된 코드)기존 Strategy로 이용할려던 NftTransaction 인터페이스를 역할 별 인터페이스인 NftWalletInfo, NftMart 두개로 나누어서 베이스가 될 Strategy가 없는 상태. 이후 카이카스 등을 수월하게 이용하기위해선 NftTransaction 추상클래스를 만들어 상속시켜 사용하면 된다.
5. 카이카스 연동 및 기타 플랫폼 연동
6. 프라이빗 체인 구현

원격 접속환경이 안좋을때

터미널에 pkill node 사용