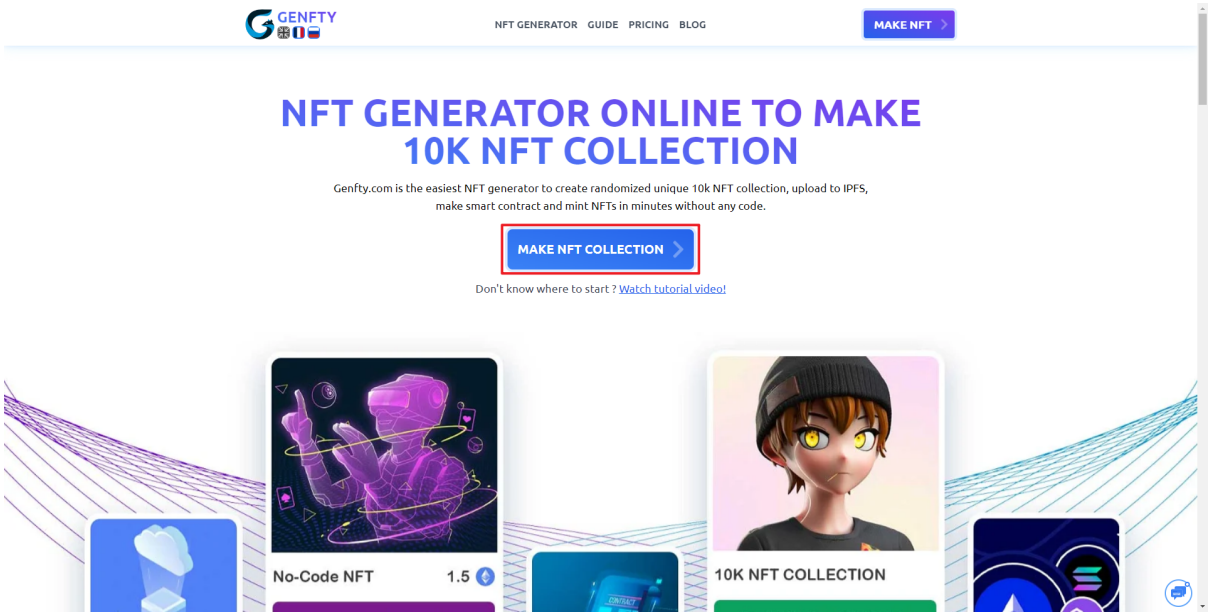
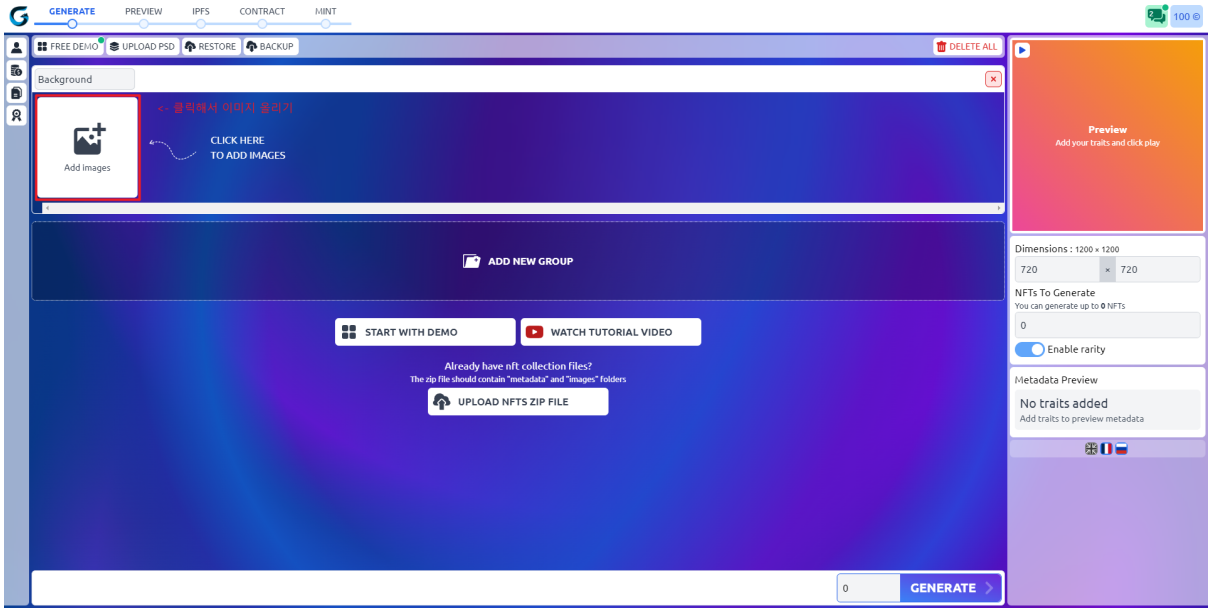
nft발행하고 리액트로 가져오기

태그	
날짜	@2022년 12월 7일

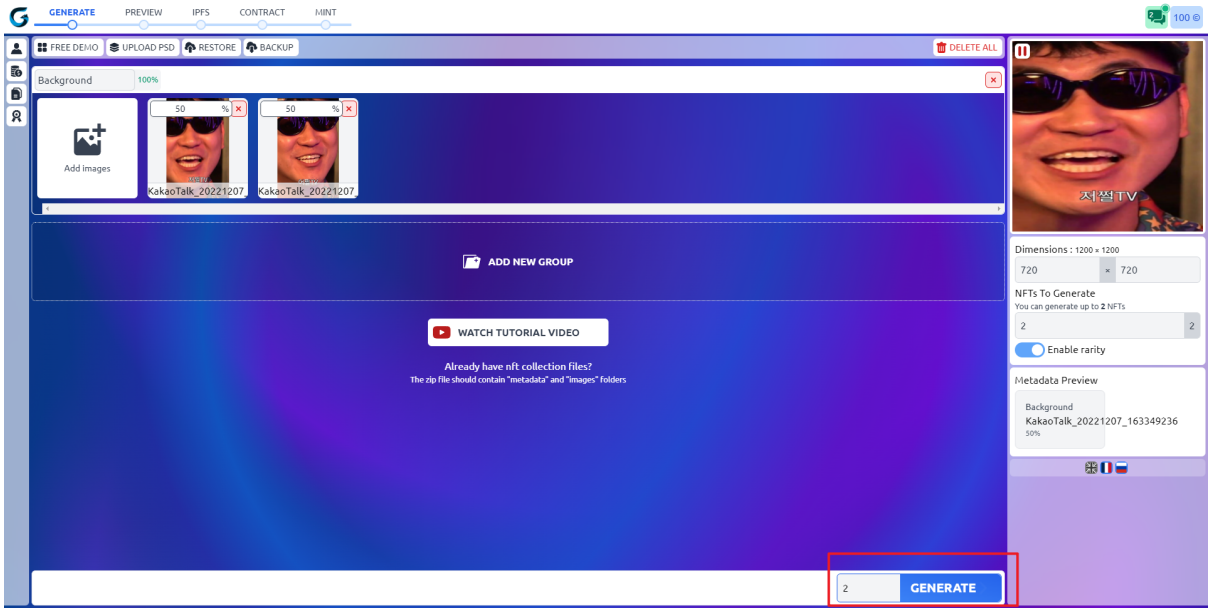
https://www.genfty.com/ nft사용할 수 있는 곳



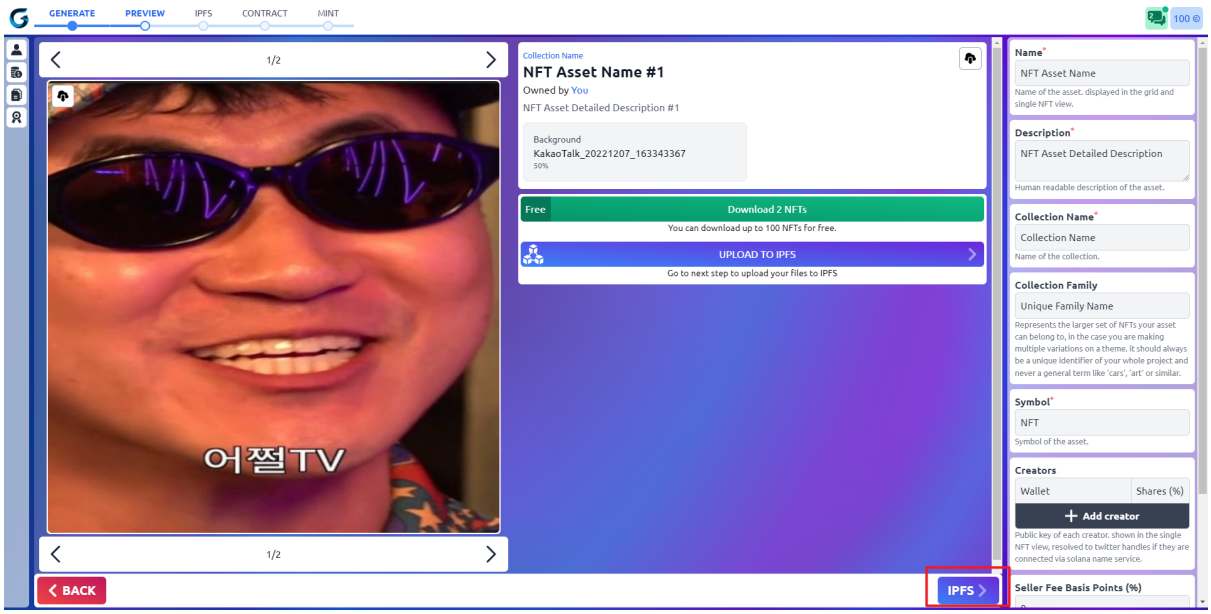
Make NFT COLLECTION을 눌러서 들어가자.



사진을 추가하는 곳이다.

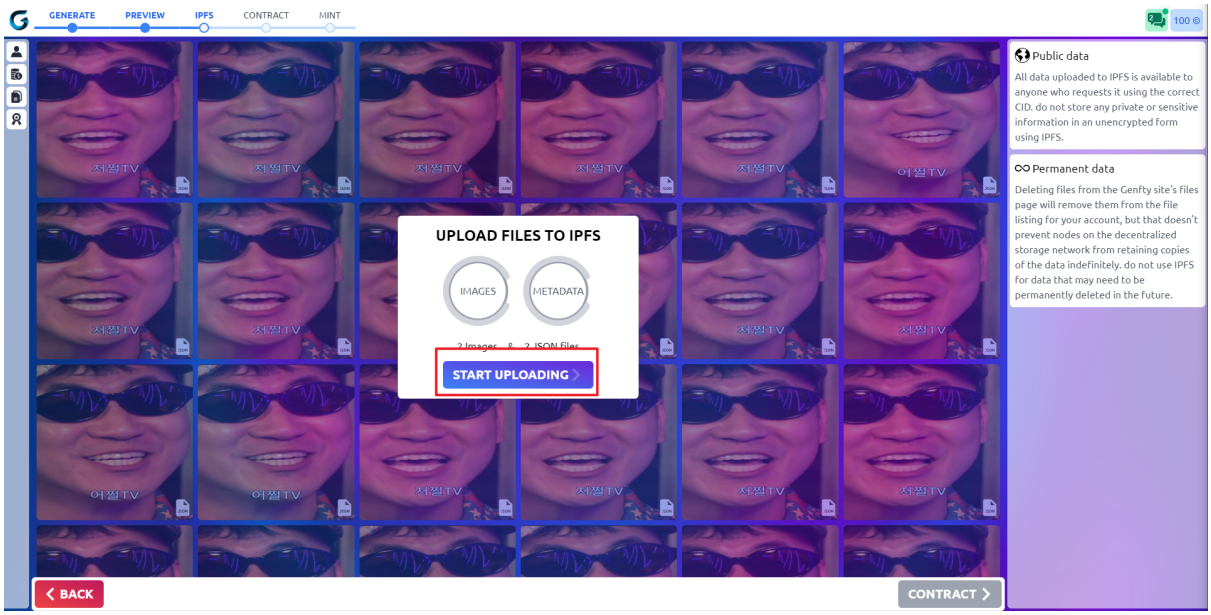


사진을 추가한 후에 아래 GENERATE을 눌러서 진행하도록 하자.

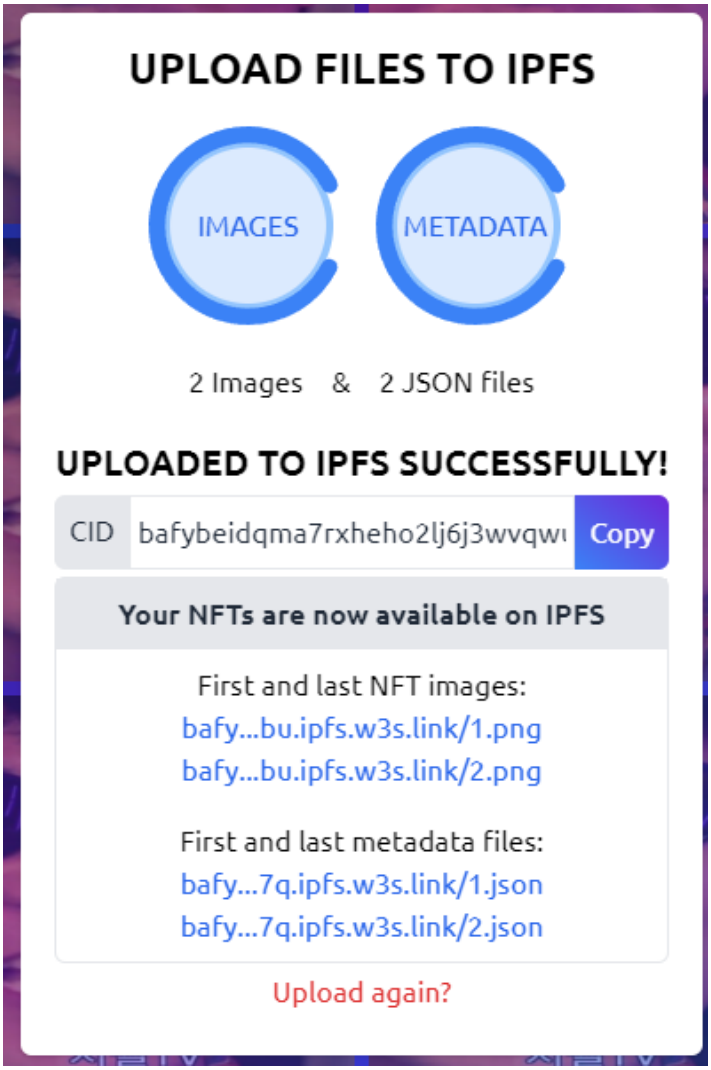


아래 IPFS를 눌러서 계속 진행한다.

옆에 Name이나 그외 항목들을 변경할 수도 있다.



스타트를 눌러서 진행하도록 하자.



발행이 되었다.

각 주소들을 눌러서 사진을 확인해보고 저장해두자

https://bafybeib4qivdtbv3xkkjrhmkfirjzudihozb54kjrqkku5m6jofygq6dbu.ipfs.w3s.link/1.png

https://bafybeib4qivdtbv3xkkjrhmkfirjzudihozb54kjrqkku5m6jofygq6dbu.ipfs.w3s.link/2.png

png에 경우 눌렀을때 그림이 나온다.

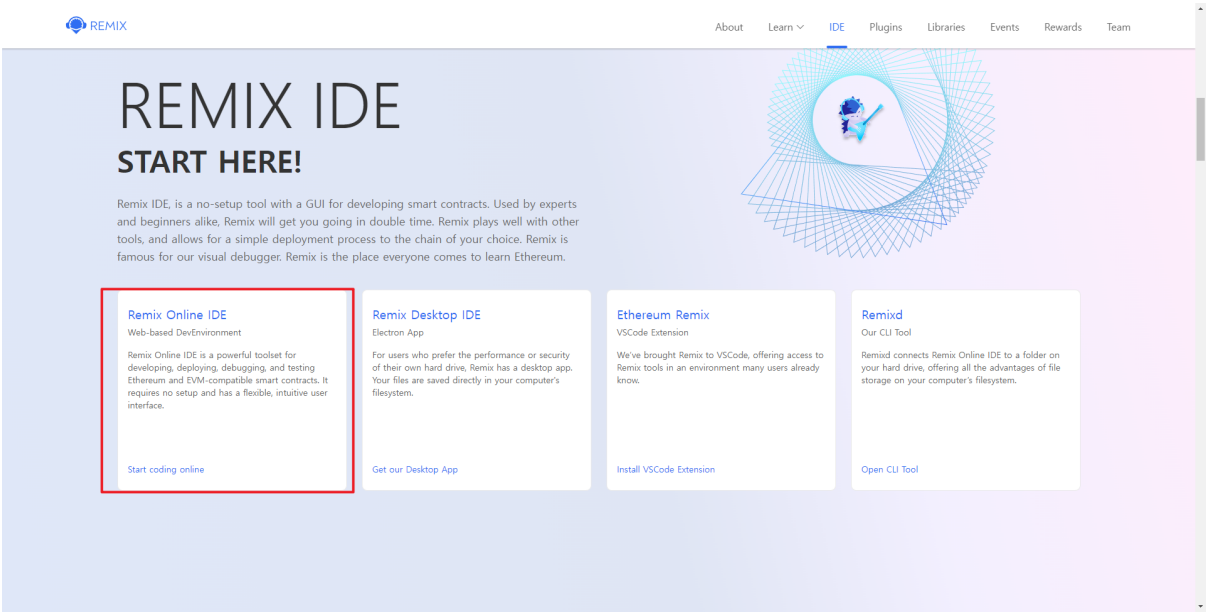
만약 2개 이상일 경우 주소가 나오지 않을 수도 있는데 뒤에 1. 2. 이 부분만 변경하면 확인할 수 있다.

https://bafybeidqma7rxheho2lj6j3wvqwugmi7mvdum6npr2njshuhbk5qsfby7q.ipfs.w3s.link/1.json

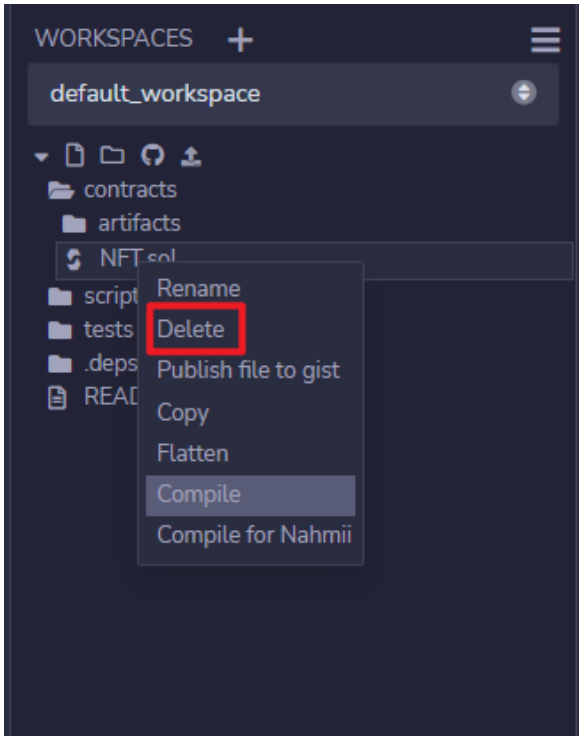
https://bafybeidqma7rxheho2lj6j3wvqwugmi7mvdum6npr2njshuhbk5qsfby7q.ipfs.w3s.link/2.json

JSON에 경우 위처럼 코드 값이 나온다.

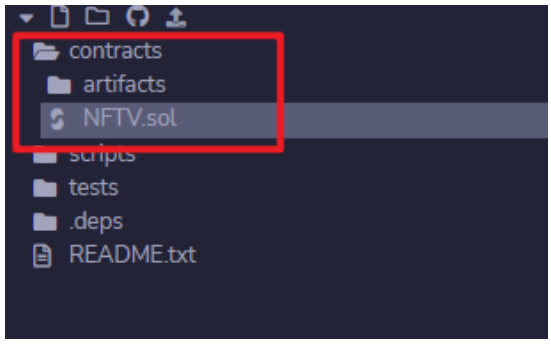
REMIX IDE



온라인으로 토큰을 발행할 수 있는 사이트이다. 빨간 네모 박스를 눌러 들어가자.



먼저 만들어것이 있다면 우클릭으로 삭제 한 후에 새로 파일을 만들어서 진행한다.



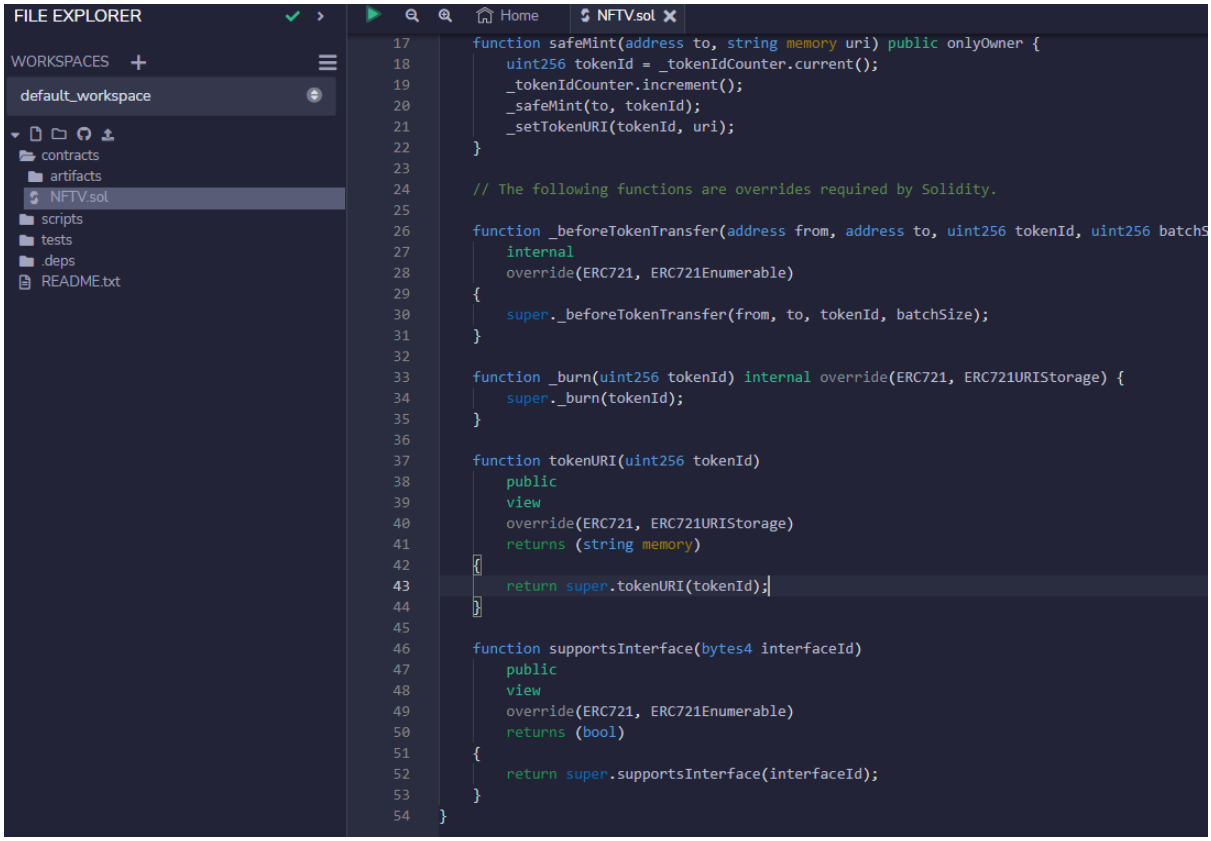
contracts 폴더 밑에 이름.sol 파일을 생성하면 된다.

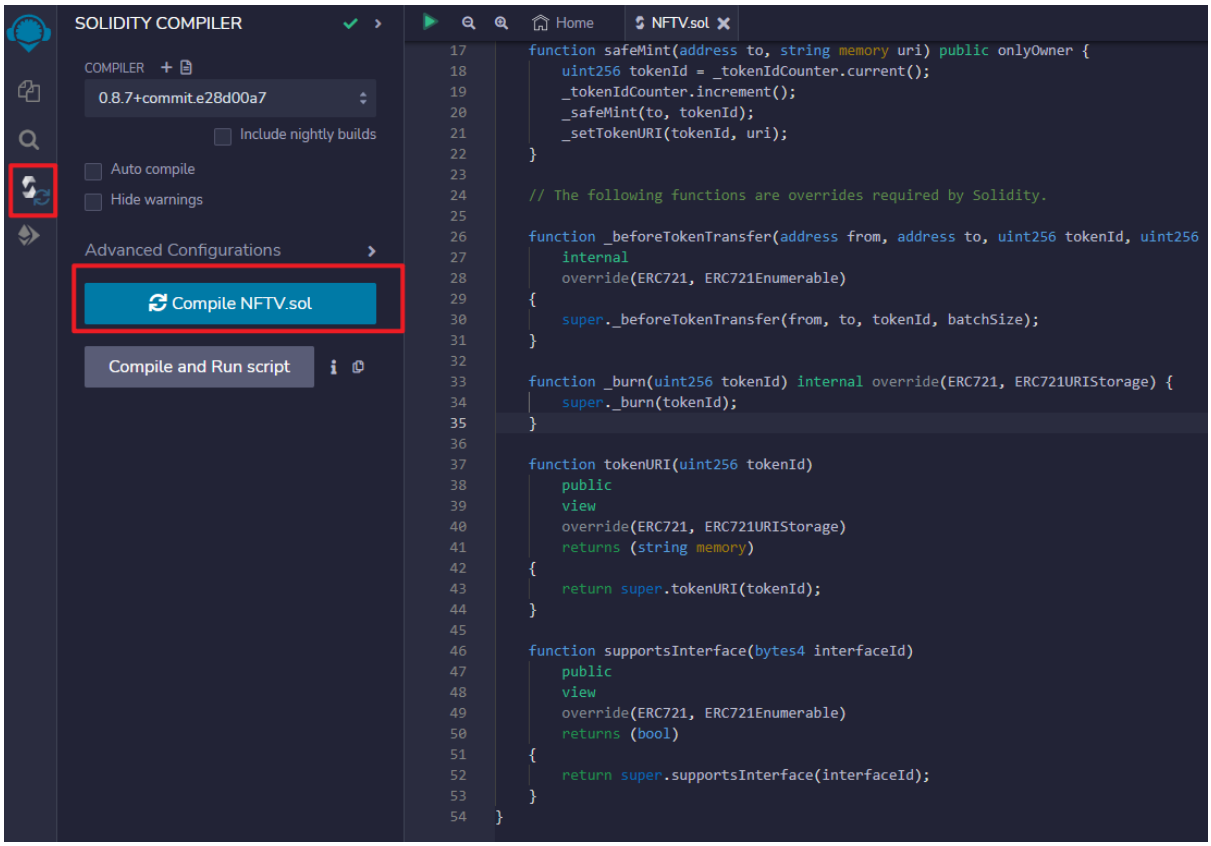
생성된 파일에 아래 코드를 붙혀넣기 하자.

붙혀넣게 되면 팝업창이 뜨는데 ok를 눌러 닫자.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.9;

import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol";
import "@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.sol";
import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/utils/Counters.sol";

contract MyToken is ERC721, ERC721Enumerable, ERC721URIStorage, Ownable {
    using Counters for Counters.Counter;
```

```
    Counters.Counter private _tokenIdCounter;

    constructor() ERC721("MyToken", "MTK") {}

    function safeMint(address to, string memory uri) public onlyOwner {
        uint256 tokenId = _tokenIdCounter.current();
        _tokenIdCounter.increment();
        _safeMint(to, tokenId);
        _setTokenURI(tokenId, uri);
    }

    // The following functions are overrides required by Solidity.

    function _beforeTokenTransfer(address from, address to, uint256 tokenId, uint256 batchSize)
        internal
        override(ERC721, ERC721Enumerable)
    {
        super._beforeTokenTransfer(from, to, tokenId, batchSize);
    }

    function _burn(uint256 tokenId) internal override(ERC721, ERC721URIStorage) {
        super._burn(tokenId);
    }

    function tokenURI(uint256 tokenId)
        public
        view
        override(ERC721, ERC721URIStorage)
        returns (string memory)
    {
        return super.tokenURI(tokenId);
    }

    function supportsInterface(bytes4 interfaceId)
        public
        view
        override(ERC721, ERC721Enumerable)
        returns (bool)
    {
        return super.supportsInterface(interfaceId);
    }
}
```

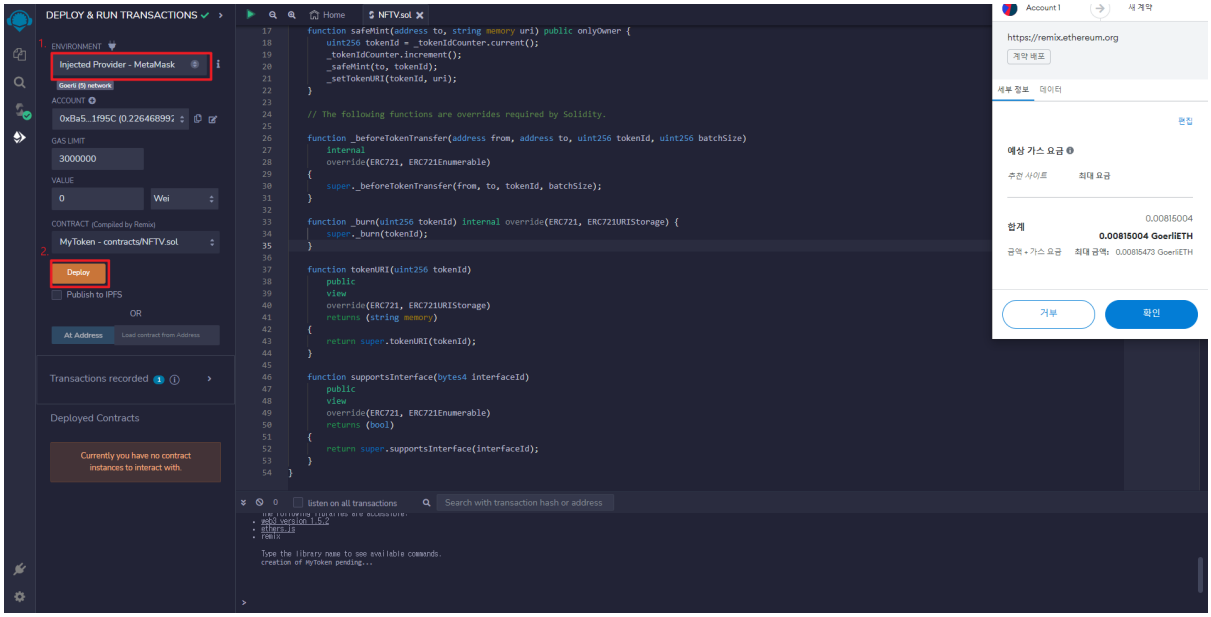

FILE EXPLORER
WORKSPACES
default_workspace
contracts
artifacts
NFTV.sol
scripts
tests
.deps
README.txt
Home
NFTV.sol
function safeMint(address to, string memory uri) public onlyOwner {
uint256 tokenId = _tokenIdCounter.current();
_tokenIdCounter.increment();
_safeMint(to, tokenId);
_setTokenURI(tokenId, uri);
}
// The following functions are overrides required by Solidity.
function _beforeTokenTransfer(address from, address to, uint256 tokenId, uint256 batchS
internal
override(ERC721, ERC721Enumerable)
{
super._beforeTokenTransfer(from, to, tokenId, batchSize);
}
function _burn(uint256 tokenId) internal override(ERC721, ERC721URIStorage) {
super._burn(tokenId);
}
function tokenURI(uint256 tokenId)
public
view
override(ERC721, ERC721URIStorage)
returns (string memory)
{
return super.tokenURI(tokenId);
}
function supportsInterface(bytes4 interfaceId)
public
view
override(ERC721, ERC721Enumerable)
returns (bool)
{
return super.supportsInterface(interfaceId);
}
}


SOLIDITY COMPILER
COMPILER
0.8.7+commit.e28d00a7
Include nightly builds
Auto compile
Hide warnings
Advanced Configurations
Compile NFTV.sol
Compile and Run script
Home
NFTV.sol
function safeMint(address to, string memory uri) public onlyOwner {
uint256 tokenId = _tokenIdCounter.current();
_tokenIdCounter.increment();
_safeMint(to, tokenId);
_setTokenURI(tokenId, uri);
}
// The following functions are overrides required by Solidity.
function _beforeTokenTransfer(address from, address to, uint256 tokenId, uint256
internal
override(ERC721, ERC721Enumerable)
{
super._beforeTokenTransfer(from, to, tokenId, batchSize);
}
function _burn(uint256 tokenId) internal override(ERC721, ERC721URIStorage) {
super._burn(tokenId);
}
function tokenURI(uint256 tokenId)
public
view
override(ERC721, ERC721URIStorage)
returns (string memory)
{
return super.tokenURI(tokenId);
}
function supportsInterface(bytes4 interfaceId)
public
view
override(ERC721, ERC721Enumerable)
returns (bool)
{
return super.supportsInterface(interfaceId);
}
}

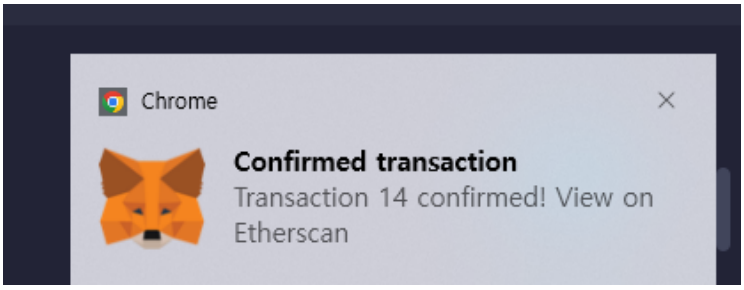
옆에 메뉴바를 통해서 넘어가고 Compile 을 진행하자

그림으로 넣어도 되고 JSON으로 넣어도 되는데 JSON에는 해쉬값이 적혀있다.
NFT민팅할 때 JSON으로 넣으면 opensea에도 뜬다.

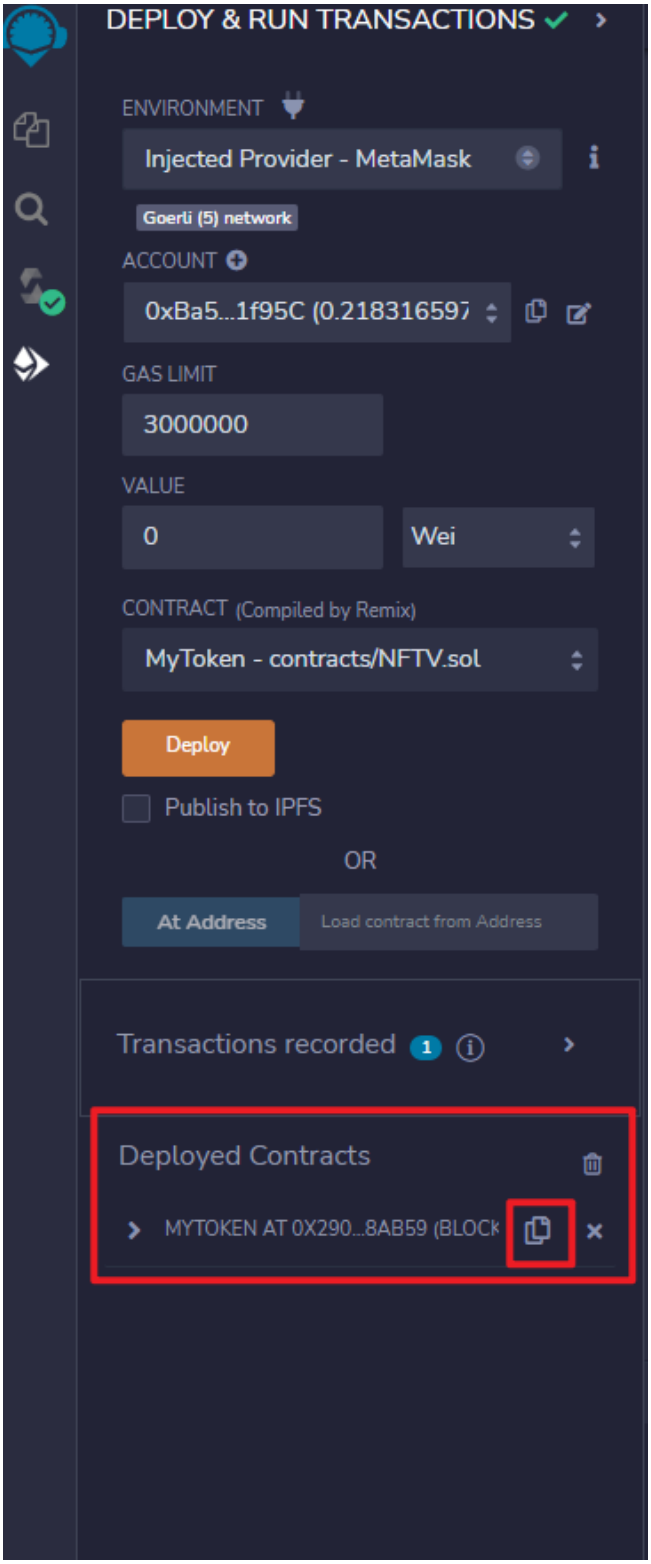
배포



맨 위 매뉴인 ENVIRONMENT 선택에서 3번째 Injected Provider - MetaMask로 변경하고
아래 Deploy를 누르면 된다 그럼 메타마스크에 연결 되면서 가스 지불화면이 나온다.
확인을 눌러 가스 요금을 지불하자.



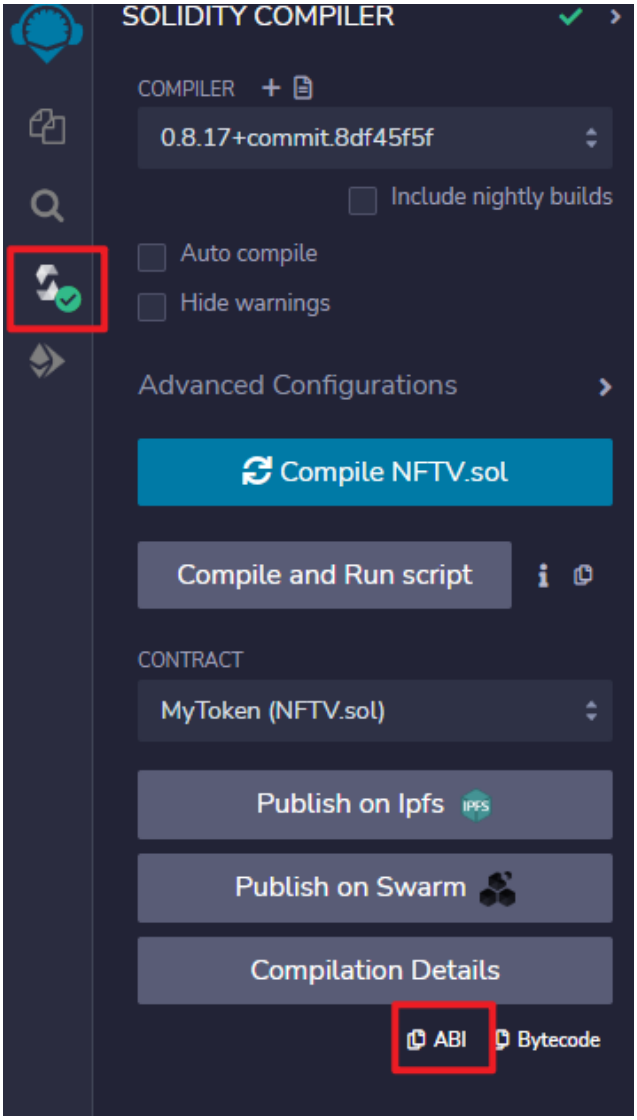
화면 오른쪽 아래에 이렇게 창이 뜨면서 메뉴바가 변경된다.



저기 적혀있는 코드 같은것은 컨트렉트 주소이다.

중요하게 쓰이니 꼭 메모해 두자

컨트렉트 주소 : 0x290BC3e15fED66647b686d6F841cBAAadb28aB59



메뉴바에서 이동하여 ABI 주소도 메모해두자

ABI코드

```
[
  {
    "inputs": [],
    "stateMutability": "nonpayable",
    "type": "constructor"
  },
  {
    "anonymous": false,
    "inputs": [
      {
        "indexed": true,
        "internalType": "address",
        "name": "owner",
        "type": "address"
      },
      {
        "indexed": true,
        "internalType": "address",
        "name": "approved",
```

```
        "type": "address"
      },
      {
        "indexed": true,
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
    "name": "Approval",
    "type": "event"
  },
  {
    "anonymous": false,
    "inputs": [
      {
        "indexed": true,
        "internalType": "address",
        "name": "owner",
        "type": "address"
      },
      {
        "indexed": true,
        "internalType": "address",
        "name": "operator",
        "type": "address"
      },
      {
        "indexed": false,
        "internalType": "bool",
        "name": "approved",
        "type": "bool"
      }
    ],
    "name": "ApprovalForAll",
    "type": "event"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "to",
        "type": "address"
      },
      {
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
    "name": "approve",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "anonymous": false,
    "inputs": [
      {
        "indexed": true,
        "internalType": "address",
        "name": "previousOwner",
        "type": "address"
      },
      {
```

```
        "indexed": true,
        "internalType": "address",
        "name": "newOwner",
        "type": "address"
      }
    ],
    "name": "OwnershipTransferred",
    "type": "event"
  },
  {
    "inputs": [],
    "name": "renounceOwnership",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "to",
        "type": "address"
      },
      {
        "internalType": "string",
        "name": "uri",
        "type": "string"
      }
    ],
    "name": "safeMint",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "from",
        "type": "address"
      },
      {
        "internalType": "address",
        "name": "to",
        "type": "address"
      },
      {
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
    "name": "safeTransferFrom",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "from",
        "type": "address"
      },
      {
```

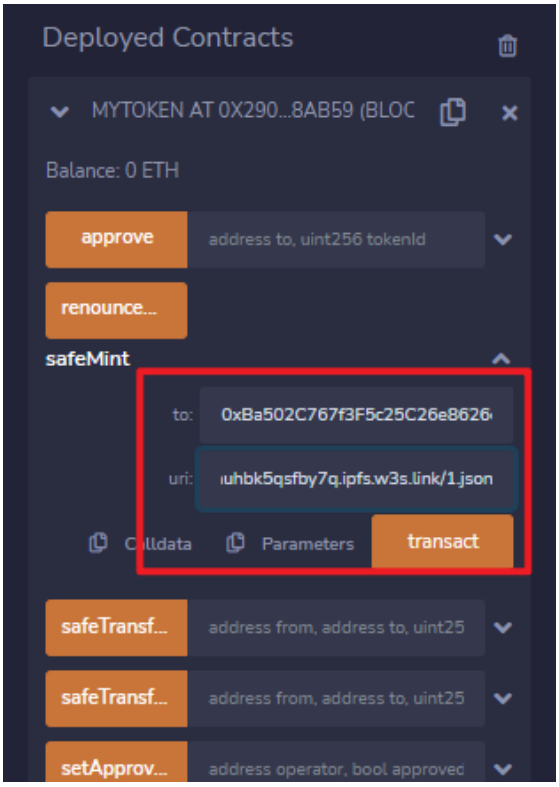
```
        "internalType": "address",
        "name": "to",
        "type": "address"
      },
      {
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      },
      {
        "internalType": "bytes",
        "name": "data",
        "type": "bytes"
      }
    ],
    "name": "safeTransferFrom",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "operator",
        "type": "address"
      },
      {
        "internalType": "bool",
        "name": "approved",
        "type": "bool"
      }
    ],
    "name": "setApprovalForAll",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "anonymous": false,
    "inputs": [
      {
        "indexed": true,
        "internalType": "address",
        "name": "from",
        "type": "address"
      },
      {
        "indexed": true,
        "internalType": "address",
        "name": "to",
        "type": "address"
      },
      {
        "indexed": true,
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
    "name": "Transfer",
    "type": "event"
  },
  {
    "inputs": [
```

```
      {
        "internalType": "address",
        "name": "from",
        "type": "address"
      },
      {
        "internalType": "address",
        "name": "to",
        "type": "address"
      },
      {
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
    "name": "transferFrom",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "newOwner",
        "type": "address"
      }
    ],
    "name": "transferOwnership",
    "outputs": [],
    "stateMutability": "nonpayable",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "owner",
        "type": "address"
      }
    ],
    "name": "balanceOf",
    "outputs": [
      {
        "internalType": "uint256",
        "name": "",
        "type": "uint256"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
    "name": "getApproved",
    "outputs": [
      {
        "internalType": "address",
```

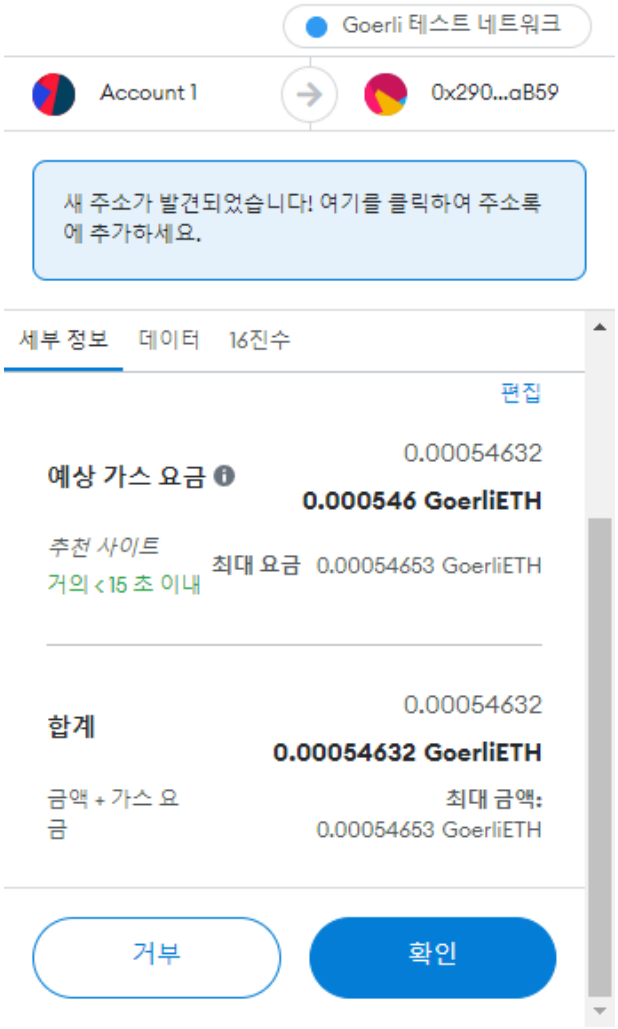
```
        "name": "",
        "type": "address"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "address",
        "name": "owner",
        "type": "address"
      },
      {
        "internalType": "address",
        "name": "operator",
        "type": "address"
      }
    ],
    "name": "isApprovedForAll",
    "outputs": [
      {
        "internalType": "bool",
        "name": "",
        "type": "bool"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [],
    "name": "name",
    "outputs": [
      {
        "internalType": "string",
        "name": "",
        "type": "string"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [],
    "name": "owner",
    "outputs": [
      {
        "internalType": "address",
        "name": "",
        "type": "address"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
```

```
    "name": "ownerOf",
    "outputs": [
      {
        "internalType": "address",
        "name": "",
        "type": "address"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "bytes4",
        "name": "interfaceId",
        "type": "bytes4"
      }
    ],
    "name": "supportsInterface",
    "outputs": [
      {
        "internalType": "bool",
        "name": "",
        "type": "bool"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [],
    "name": "symbol",
    "outputs": [
      {
        "internalType": "string",
        "name": "",
        "type": "string"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "uint256",
        "name": "index",
        "type": "uint256"
      }
    ],
    "name": "tokenByIndex",
    "outputs": [
      {
        "internalType": "uint256",
        "name": "",
        "type": "uint256"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
```

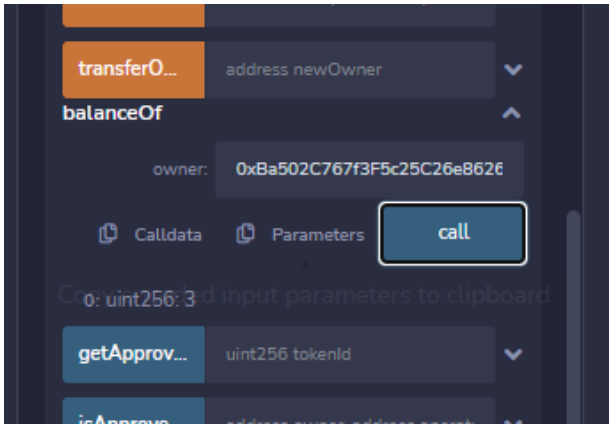
```
        "internalType": "address",
        "name": "owner",
        "type": "address"
      },
      {
        "internalType": "uint256",
        "name": "index",
        "type": "uint256"
      }
    ],
    "name": "tokenOfOwnerByIndex",
    "outputs": [
      {
        "internalType": "uint256",
        "name": "",
        "type": "uint256"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "uint256",
        "name": "tokenId",
        "type": "uint256"
      }
    ],
    "name": "tokenURI",
    "outputs": [
      {
        "internalType": "string",
        "name": "",
        "type": "string"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [],
    "name": "totalSupply",
    "outputs": [
      {
        "internalType": "uint256",
        "name": "",
        "type": "uint256"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  }
]
```



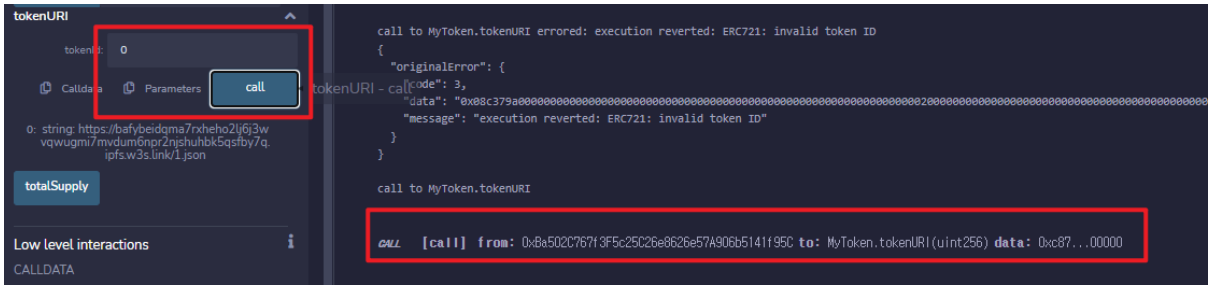
to에는 내 메타마스크 주소를 넣고, url은 아까의 json 주소를 넣어서 transact을 하자.

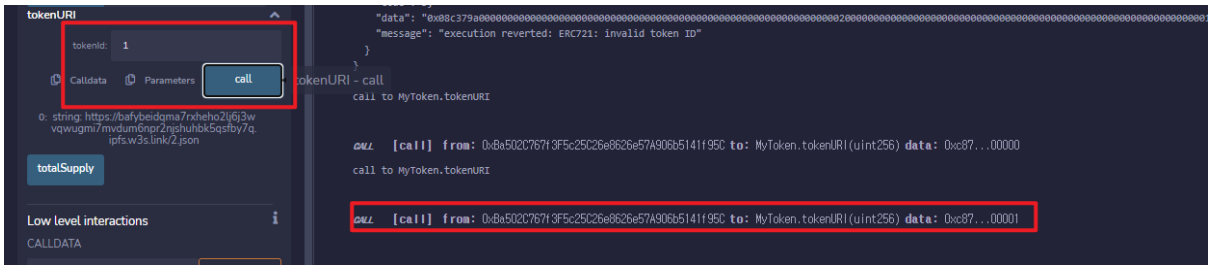


다시 가스를 지불하고 진행하자. 이후 2번째 json도 똑같은 방법으로 진행하면 된다.



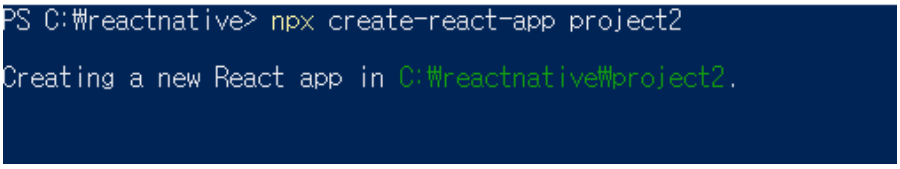
내 메타 마스크 주소를 넣고 진행 하자.





사진을 2개를 발행을 했기 때문에 0 ,1을 넣어서 CALL을 하면 아래 채워진 로그를 통해 발행이 됬는지 확인 할 수 있다.

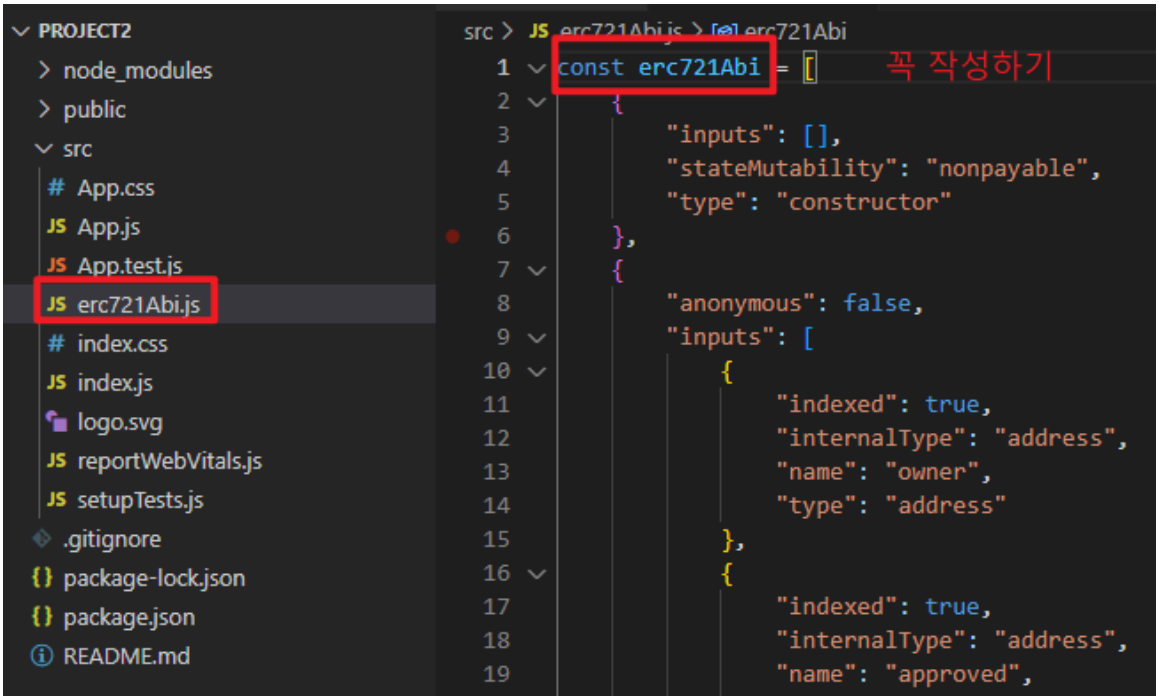
위 사진은 각 0을 넣었을 때 , 1을 넣었을 때 로그이다.



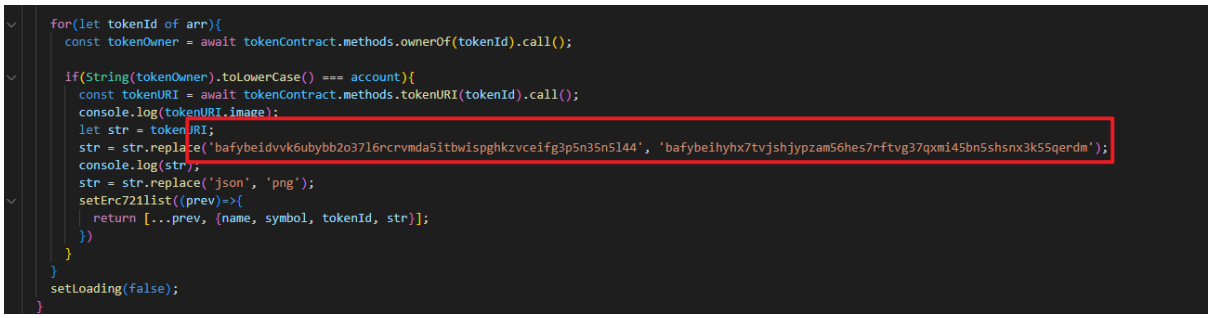
새 프로젝트를 만들자.

경로로 들어가서 web3도 설치한 후에 code .으로 vscode를 열자

App.js에서 쓰지 않을것들은 지우고 src폴더 밑에 erc721Abi.js파일을 생성해서 아까 복사해둔 abi코드를 넣고 저장한다.



App.js도 붙혀넣기 해주자.

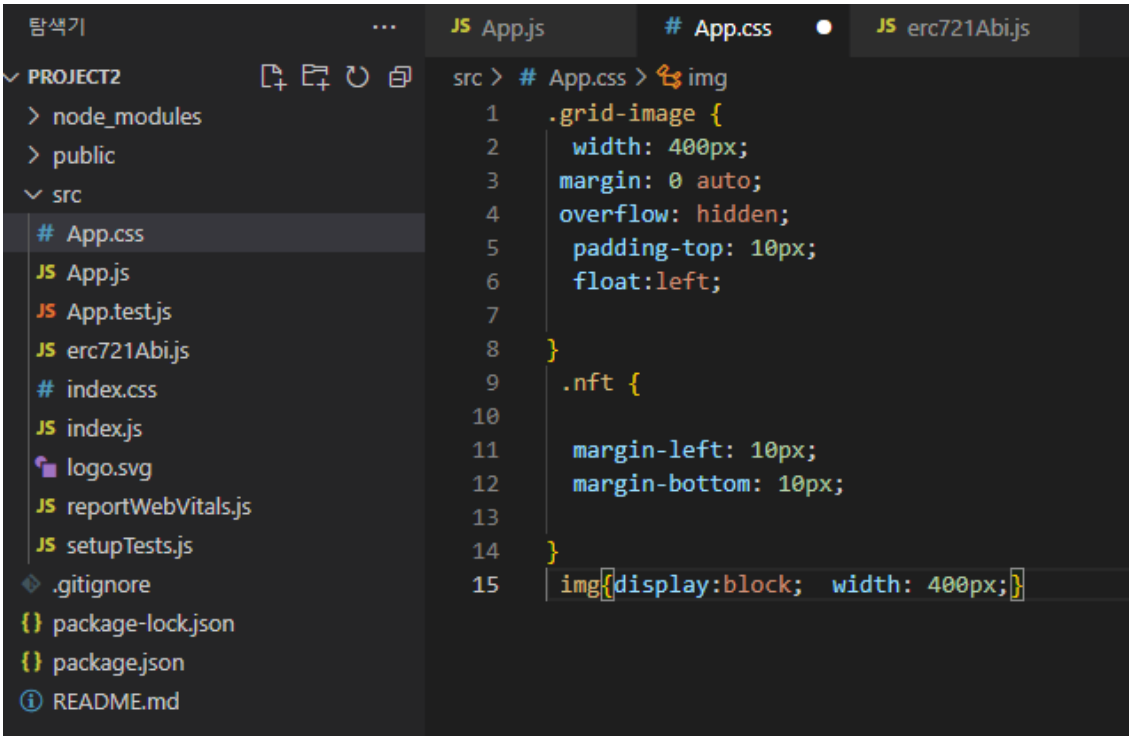


주의할 점은 저 빨간 부분을 json에서 png로 바꿔주는 부분이다.

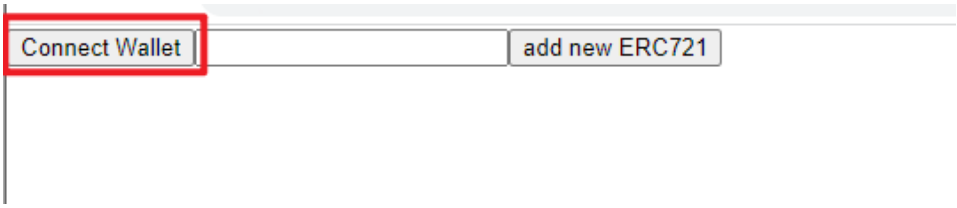
왼쪽에 json 오른쪽에 png주소를 필요한 부분만 넣고 실행하면 화면에서

그림이 나오게 작동한다.

App.css도 똑같이 해주자!



이제 npm start로 실행 시켜보자



첫 화면이다.

Connect Wallet을 누르면 내 메타마스크 계정이 연결된다.

그리고 빨칸 텍스트 칸에 아까 중요했던 컨트렉트 주소를 넣으면 된다.