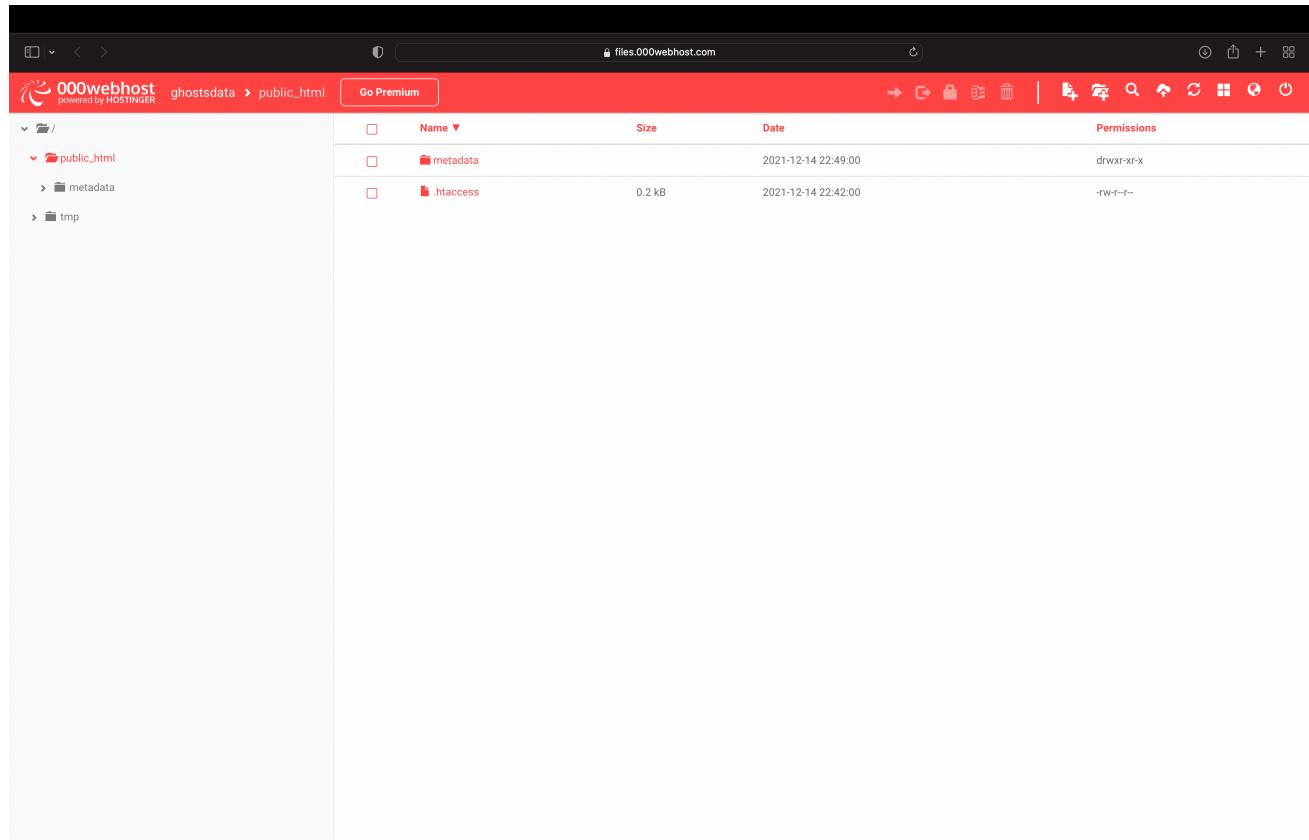


DOCUMENTATION FOR

APE CLUB NFT DROP

PART ONE

1.) Create a 'metadata' folder wherever you plan on hosting your data. It can either be a normal server with secure socket layer or IPFS. A normal server in this case



2.) Upload your collection images into this 'metadata' folder. I have provided ten example images that I will be using here. You can find them in 'Sample Metadata'

	Name	Size	Date	Permissions
	1.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	2.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	3.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	4.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	5.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	6.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	7.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	8.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	9.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--
	10.gif	1.6 MB	2021-12-15 00:12:00	-rw-r--r--

3.) Now pop open the 'Sample Metadata' folder in your favourite editor. In this case I'm using VS code.

```

1.json — Sample Metadata
EXPLORER
SAMPLE METADATA
  1.gif
  1.json
  2.gif
  2.json
  3.gif
  3.json
  4.gif
  4.json
  5.gif
  5.json
  6.gif
  6.json
  7.gif
  7.json
  8.gif
  8.json
  9.gif
  9.json
  10.gif
  10.json
  Archive.zip

1.json > image
1 {
  "dna": "1011000100",
  "name": "#1",
  "description": "Sample NFT Provided For The Project",
  "image": "https://ghostsdata.000webhostapp.com/metadata/1.gif",
  "edition": 1,
  "date": 1639523716347,
  "attributes": [
    {
      "trait_type": "Background",
      "value": "None"
    },
    {
      "trait_type": "Attire",
      "value": "None"
    },
    {
      "trait_type": "Hair",
      "value": "None"
    },
    {
      "trait_type": "Accessories",
      "value": "None"
    }
  ]
}

OUTLINE
  dna 1011000100
  name #1
  description Sample NFT Provided For The Project
  image https://ghostsdata.000webhostapp.com/metadata/1.gif
  edition 1
  date 1639523716347
  2 selections (383 characters selected) Spaces: 4 UTF-8 LF JSON Go Live

```

4.) Open up every .json and replace the image url on line 5 with your url as can be seen above.

You can then modify the traits from line 8 to 24.

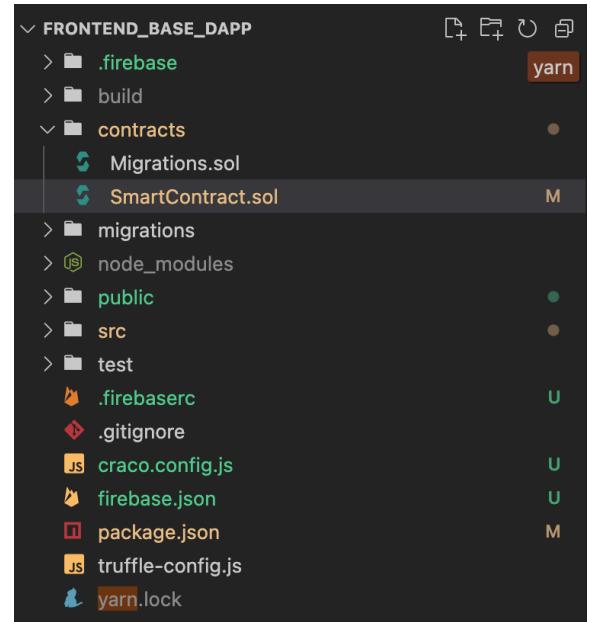
5.) Now upload the new '.json' files to the same metadata folder where your images are stored. Your metadata folder should look something like this :

	Name	Size	Date	Permissions
	1.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	1.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	2.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	2.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	3.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	3.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	4.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	4.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	5.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	5.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	6.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	6.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	7.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	7.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	8.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	8.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	9.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	9.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--
	10.gif	1.6 MB	2021-12-15 00:12:00	-rW-r--r--
	10.json	0.6 kB	2021-12-15 00:21:00	-rW-r--r--

PART TWO

1.) Time to move onto the smart contract!

So if you unzip the project and pop into the 'frontend_base_dapp' directory , you will find a smart contracts folder. In this folder is the smart contract for your drop , its called 'SmartContract.sol'

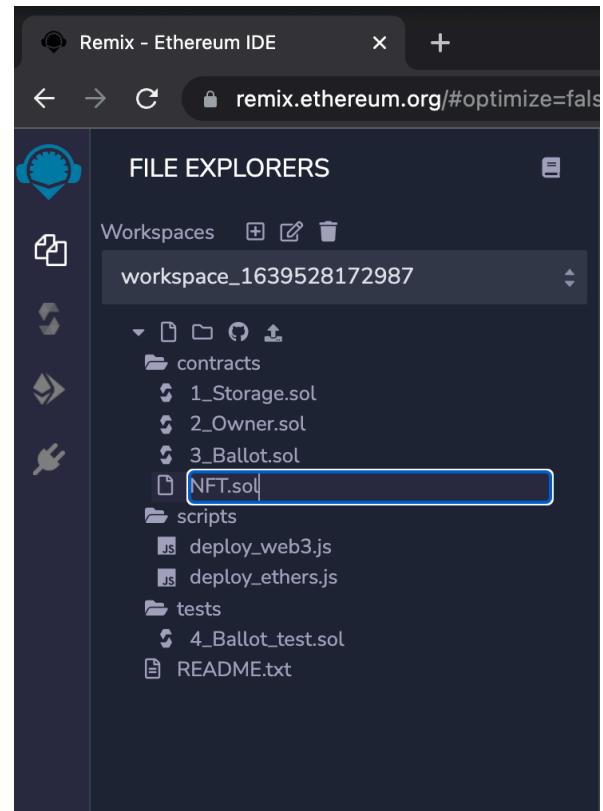


2.) If you open up the 'SmartContract.sol' file you will see the cost (amount charged per item for the mint) on line 16 , the max supply (total amount of NFTs to be minted) on line 17 and the max mint amount (the max amount someone can mint at for their wallet) on line 18.

```
SmartContract.sol M ×
contracts > SmartContract.sol
1 // SPDX-License-Identifier: GPL-3.0
2
3
4
5
6 pragma solidity ^0.8.0;
7
8 import "@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol";
9 import "@openzeppelin/contracts/access/Ownable.sol";
10
11 contract ApeClub is ERC721Enumerable, Ownable {
12     using Strings for uint256;
13
14     string public baseURI;
15     string public baseExtension = ".json";
16     uint256 public cost = 0.05 ether;
17     uint256 public maxSupply = 10000;
18     uint256 public maxMintAmount = 20;
19     bool public paused = false;
20     mapping(address => bool) public whitelisted;
21
22     constructor(
23         string memory _name,
24         string memory _symbol,
25         string memory _initBaseURI
26     ) ERC721(_name, _symbol) {
27         setBaseURI(_initBaseURI);
28         mint(msg.sender, 1);
29     }
30 }
```

3.) Make the changes you want to and then copy the entire file and head over to <https://remix.ethereum.org/>

4.) Right click on the 'contracts' folder and create a new file and name it 'NFT.sol'



5.) Paste the contents of 'SmartContract.sol' into the 'NFT.sol' file in Remix like so :

A screenshot of the Remix Ethereum IDE showing the code editor. The file "NFT.sol" is open. The code is as follows:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.0;

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

contract ApeClub is ERC721Enumerable, Ownable {
    using Strings for uint256;

    string public baseURI;
    string public baseExtension = ".json";
    uint256 public cost = 0.05 ether;
    uint256 public maxSupply = 10000;
    uint256 public maxMintAmount = 20;
    bool public paused = false;
    mapping(address => bool) public whitelisted;

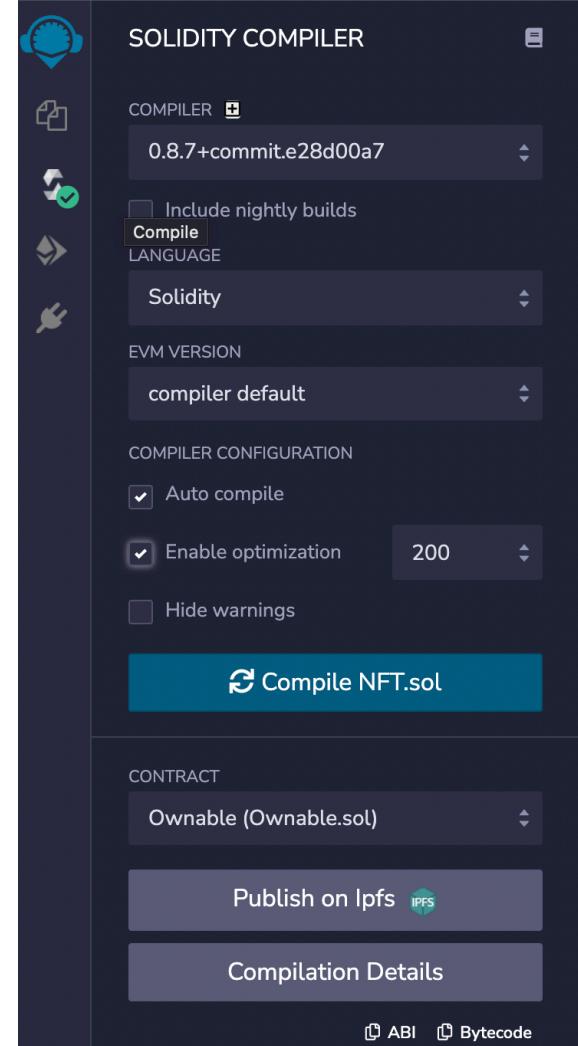
    constructor(
        string memory _name,
        string memory _symbol,
        string memory _initBaseURI
    ) ERC721(_name, _symbol) {
        setBaseURI(_initBaseURI);
        mint(msg.sender, 1);
    }

    // internal
    function _baseURI() internal view virtual override returns (string memory) {
        return baseURI;
    }

    // public
    function mint(address _to, uint256 _mintAmount) public payable {
        uint256 supply = totalSupply();
        require(!_paused);
        require(_mintAmount > 0);
        require(_mintAmount <= maxMintAmount);
        require(supply + _mintAmount <= maxSupply);
    }
}
```

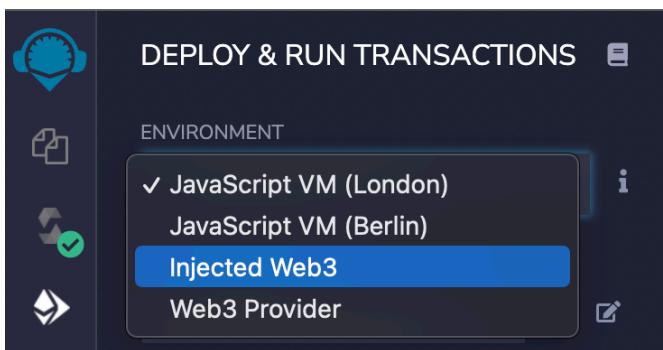
The code is a Solidity smart contract for an NFT collection called "ApeClub". It inherits from ERC721 and ERC721Enumerable, and includes Ownable and a baseURI mapping. It defines a constructor to initialize the baseURI, and a mint function to allow users to purchase tokens.

6.) Click on the second tab to the left of the Remix IDE. The Compiler tab. Check auto compile and enable optimisation. Then press the large ‘Compile NFT.sol’ button

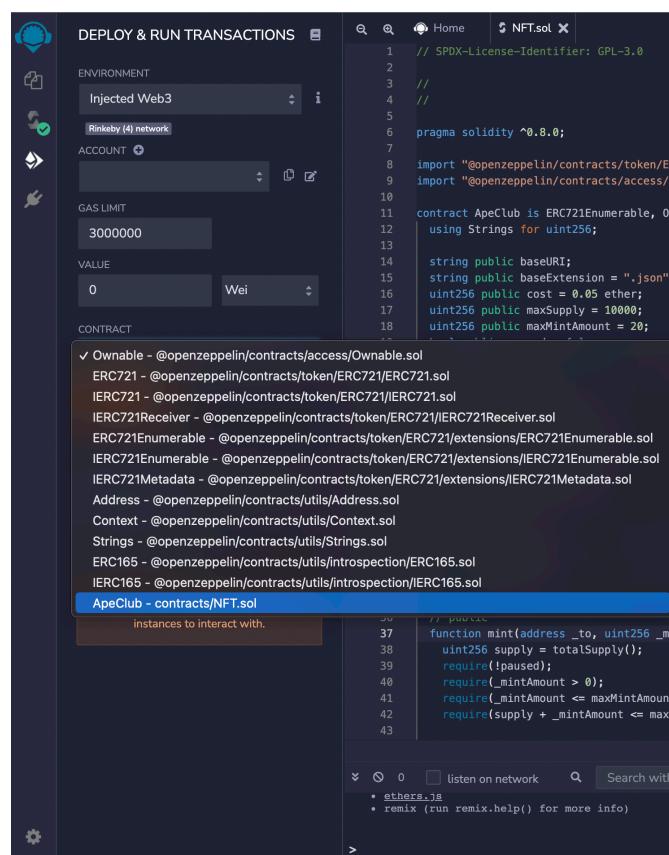


7.) Now hop over to the third tab to the left of the IDE. The Deploy & Run transactions tab.

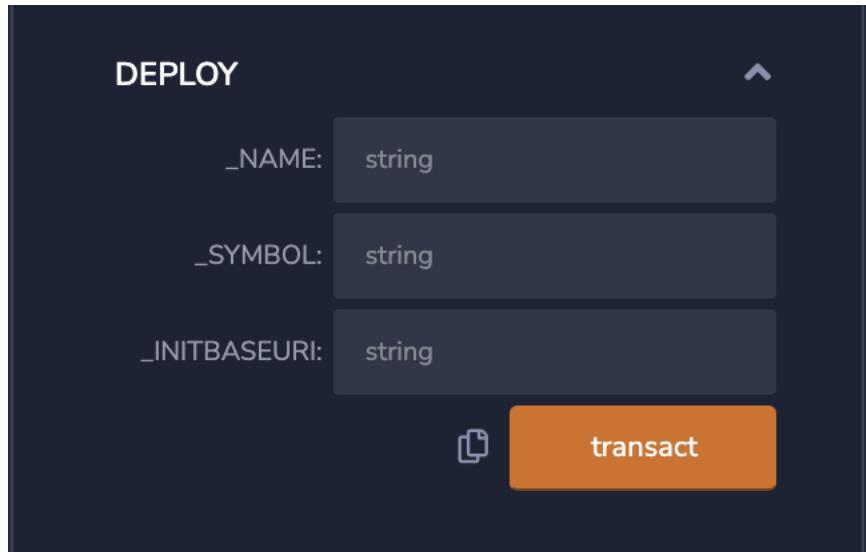
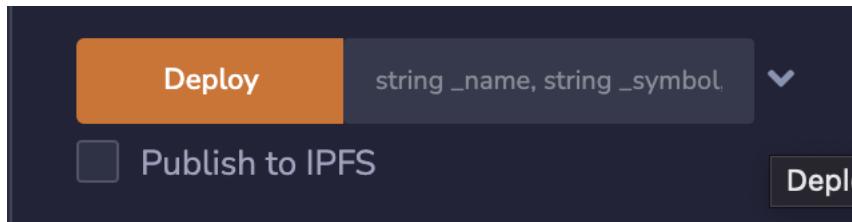
7.1) Under Environment , select ‘injected web3’ and connect MetaMask



7.2) Then under Contract , select ‘NFT.sol’

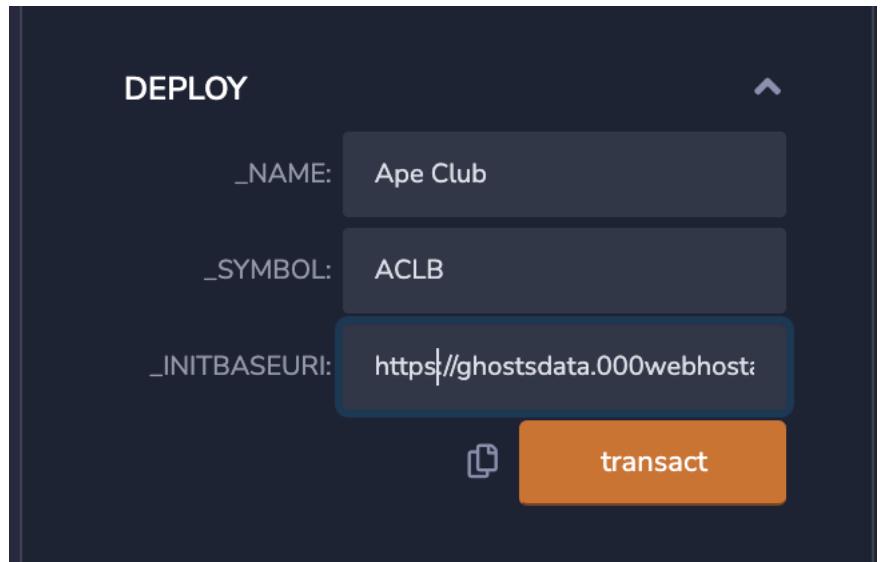


7.3) Press the arrow to the right to expose the options.



7.4 Here you will find the name , symbol and baseURI for the collection. The baseURI refers to the URL of the metadata. So for this collection it is <https://ghostsdata.000webhostapp.com/metadata/>

Here is the example :

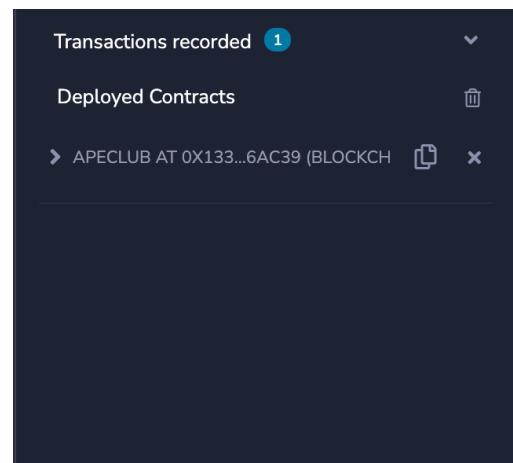


7.5) Press the transact button to deploy the contract. You will then be asked to sign the transaction in MetaMask and pay the gas fees.

In this case I am deploying on the Rinkeby test net.

7.6) Once the transaction is confirmed , your contract will pop up in the bottom left corner of the deploy page.

Copy the contract address



7.7) Head over to <https://opensea.io/get-listed> , select the correct network and paste in the contract address that was received after deploying in Remix. Then press submit.

Enter your contract address

What is the address of your ERC721 or ERC1155 contract on the testnet Network?

Rinkeby

0x13386d10b2a4d98dF010e333470Fc12cbDc6aC39

Submit

7.8 So now the collection is working because we can see it on OpenSea and all our metadata and images are correct. Great , time to move on!

testnets.opensea.io/collection/ape-club-v2

OpenSea Testnets

Search items, collections, and accounts

Explore Stats Resources Create

1 item 1 owner --- ₿ 0.00 volume traded

Welcome to the home of Ape Club V2 on OpenSea. Discover the best items in this collection.

Items Activity

Filter Status Price Chains On Sale In

Buy Now On Auction New Has Offers

United States Dollar (USD) Min to Max Apply

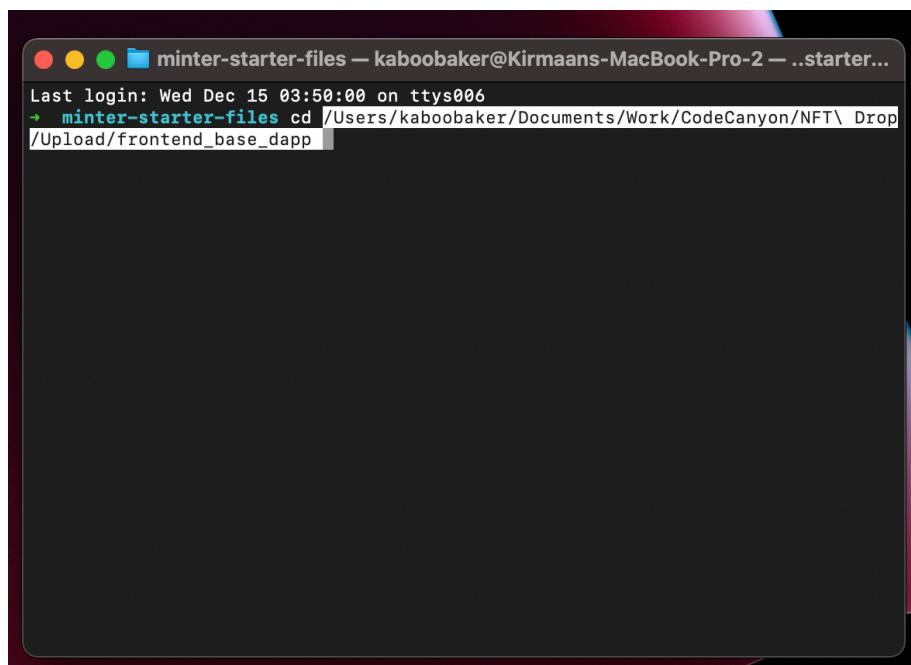
1 result

Ape Club V2 #1

Heart 0

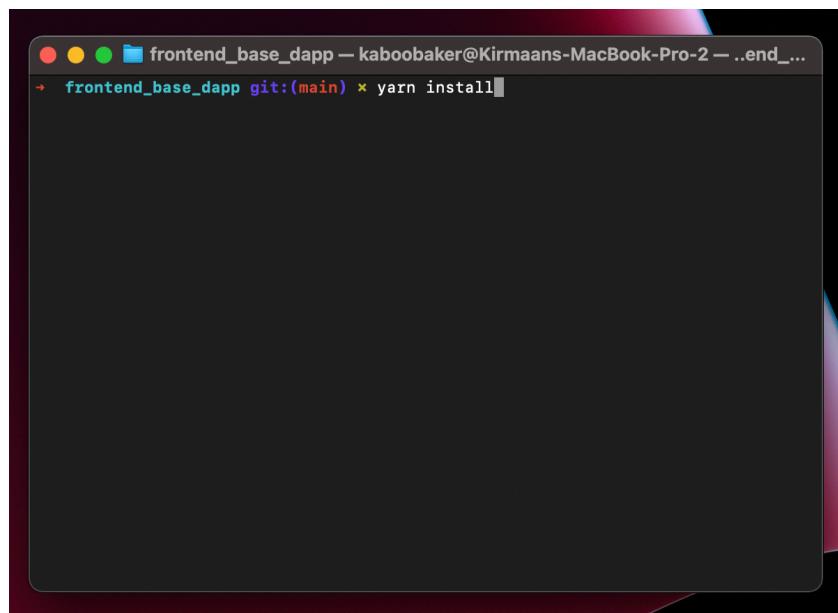
PART THREE

1.) Unzip the project file and navigate into the 'frontend_base_dapp' folder using terminal



```
minter-starter-files — kabooebaker@Kirmaans-MacBook-Pro-2 — ..starter...
Last login: Wed Dec 15 03:50:00 on ttys006
→ minter-starter-files cd /Users/kabooebaker/Documents/Work/CodeCanyon/NFT\ Drop
/UUpload/frontend_base_dapp
```

2.) Initialise the node modules with the following command 'yarn install'

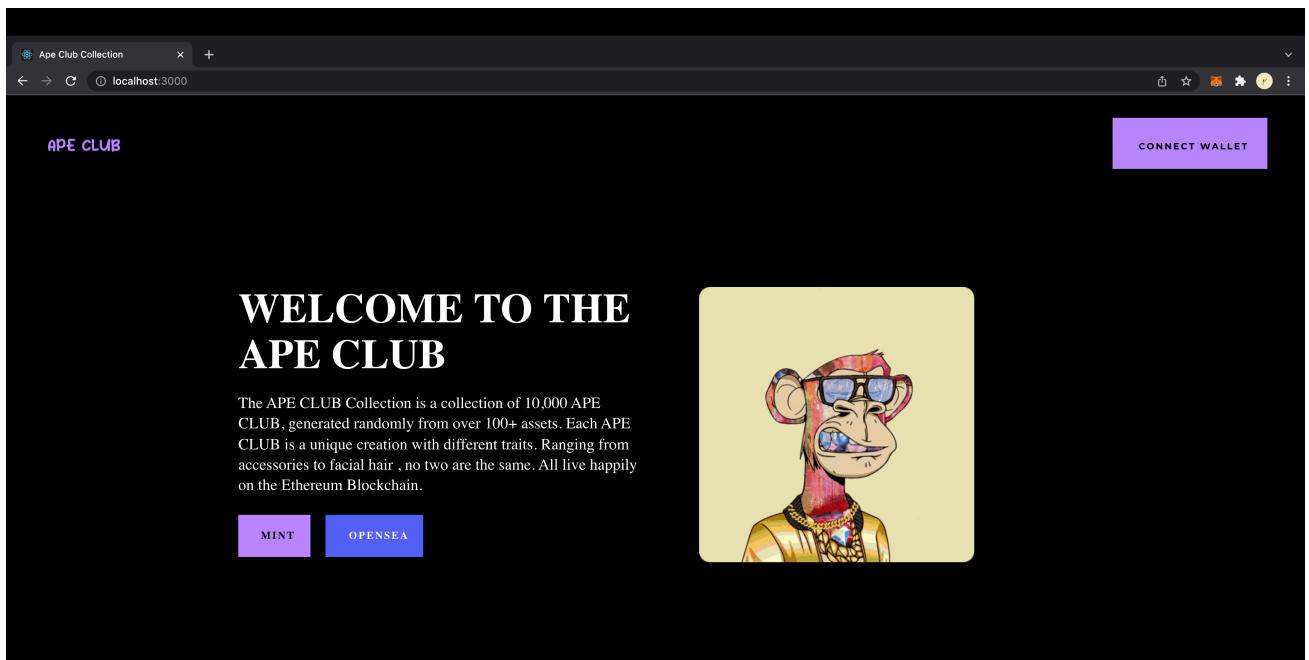


```
frontend_base_dapp — kabooebaker@Kirmaans-MacBook-Pro-2 — ..end_...
→ frontend_base_dapp git:(main) ✘ yarn install
```

3.) Once completed , run the command 'npm start' so we can make sure the project is working correctly.

```
frontend_base_dapp — kaboobaker@Kirmaans-MacBook-Pro-2 — ..end_...
yarn install v1.22.17
[1/4] ⚡ Resolving packages...
[2/4] 🛡 Fetching packages...
[3/4] ⚡ Linking dependencies...
warning " > @testing-library/user-event@12.8.3" has unmet peer dependency "@testing-library/dom@>=7.21.4".
warning " > react-canvas-draw@1.1.1" has incorrect peer dependency "react@16.x".
warning "react-scripts > @typescript-eslint/eslint-plugin > tsutils@3.21.0" has unmet peer dependency "typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta".
warning " > react-signature-canvas@1.0.3" has unmet peer dependency "prop-types@^15.5.8".
warning " > react-signature-canvas@1.0.3" has incorrect peer dependency "react@0.14 - 16".
warning " > react-signature-canvas@1.0.3" has incorrect peer dependency "react-dom@0.14 - 16".
warning " > styled-components@5.3.1" has unmet peer dependency "react-is@>= 16.8.0".
warning Workspaces can only be enabled in private projects.
[4/4] ✨ Building fresh packages...
✨ Done in 7.88s.
→ frontend_base_dapp git:(main) ✘ npm start
```

3.1) The project will launch in your default browser and you should have the working Dapp running!



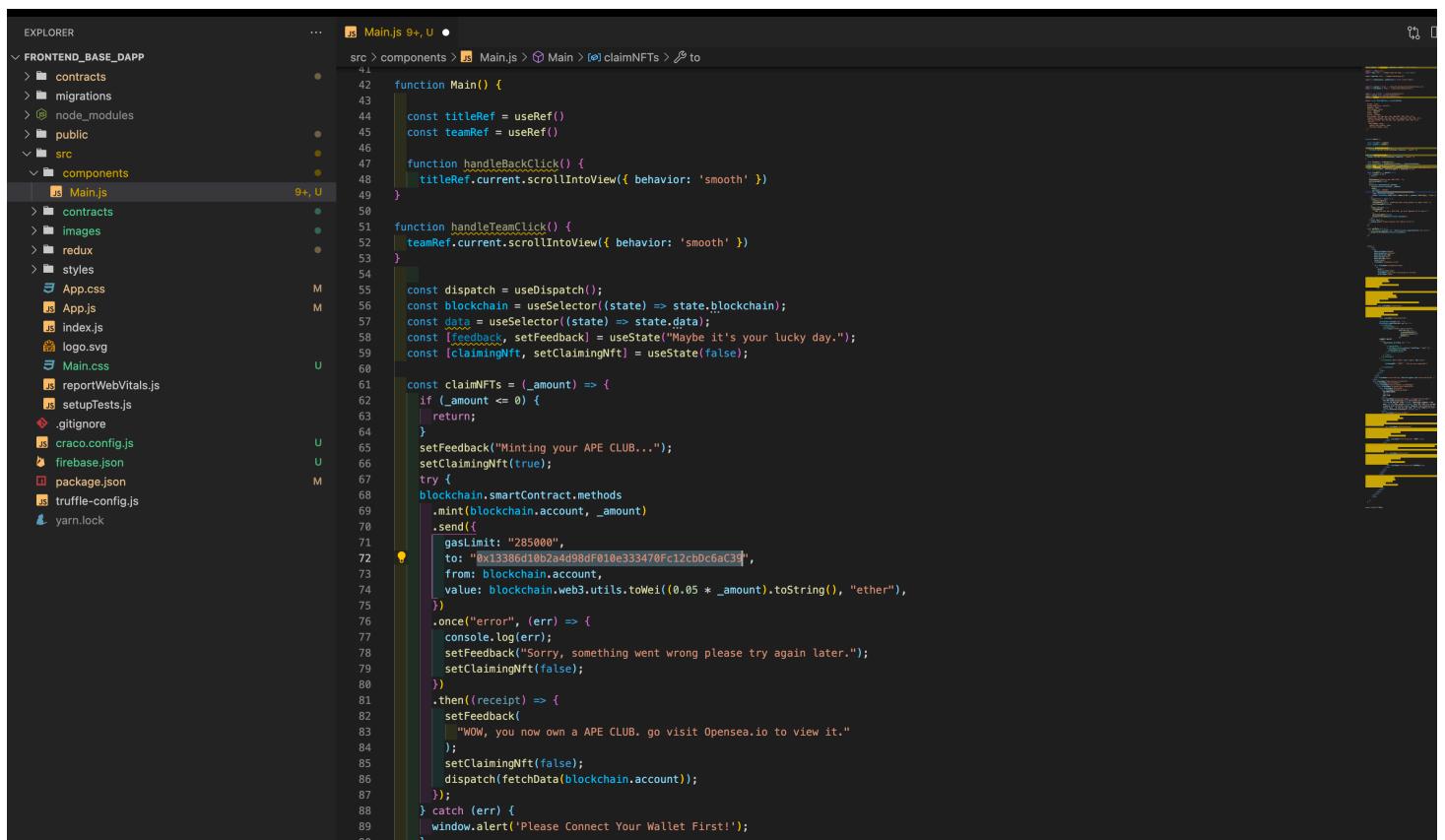
PART FOUR

1.) So now its time to swap in your contract address so that it mints an NFT from your collection.

Open the ‘frontend_base_dapp’ directory in your favourite text editor

1.1) And navigate to ‘Main.js’ in the components folder located in the src folder in the root directory.

Replace the contract address on line 72 with the one we received from Remix.



```
EXPLORER
FRONTEND_BASE_DAPP
  contracts
  migrations
  node_modules
  public
  src
    components
      Main.js
  contracts
  images
  redux
  styles
    App.css
    App.js
    index.js
    logo.svg
    Main.css
    reportWebVitals.js
    setupTests.js
    .gitignore
    craco.config.js
    firebase.json
    package.json
    truffle-config.js
    yarn.lock

Main.js 9+, U
src > components > Main.js > Main > claimNFTs > to
function Main() {
  const titleRef = useRef()
  const teamRef = useRef()

  function handleBackClick() {
    titleRef.current.scrollIntoView({ behavior: 'smooth' })
  }

  function handleTeamClick() {
    teamRef.current.scrollIntoView({ behavior: 'smooth' })
  }

  const dispatch = useDispatch()
  const blockchain = useSelector((state) => state.blockchain)
  const data = useSelector((state) => state.data)
  const [feedback, setFeedback] = useState("Maybe it's your lucky day.")
  const [claimingNft, setClaimingNft] = useState(false)

  const claimNFTs = (_amount) => {
    if (_amount <= 0) {
      return;
    }
    setFeedback("Minting your APE CLUB...");
    setClaimingNft(true);
    try {
      blockchain.smartContract.methods
        .mint(blockchain.account, _amount)
        .send({
          gasLimit: "285000",
          to: "0x13386d10b52a4d98dF010e333470Fc12cbDc6aC39",
          from: blockchain.account,
          value: blockchain.web3.utils.toWei((0.05 * _amount).toString(), "ether"),
        })
        .once("error", (err) => {
          console.log(err);
          setFeedback("Sorry, something went wrong please try again later.");
          setClaimingNft(false);
        })
        .then((receipt) => {
          setFeedback(
            "Wow, you now own a APE CLUB. go visit Opensea.io to view it."
          );
          setClaimingNft(false);
          dispatch(fetchData(blockchain.account));
        });
    } catch (err) {
      window.alert('Please Connect Your Wallet First!');
    }
  }
}
```

1.2) Now head over to the blockchainActions.js file located in the blockchain folder , which is located in the redux folder located in the src folder in the root of the project.

Replace the contract on line 54 with the one we received from Remix.

FRONTEND_BASE_DAPP

```

> contracts
> migrations
> node_modules
> public
< src
  < components
    Main.js
  > contracts
  > images
  > redux
  > blockchain
    blockchainActions.js
    blockchainReducer.js
  > data
    store.js
  > styles
    App.css
    App.js
    index.js
    logo.svg
    Main.css
    reportWebVitals.js
    setupTests.js
  .gitignore
  craco.config.js
  firebase.json
  package.json
  truffle-config.js
  yarn.lock

```

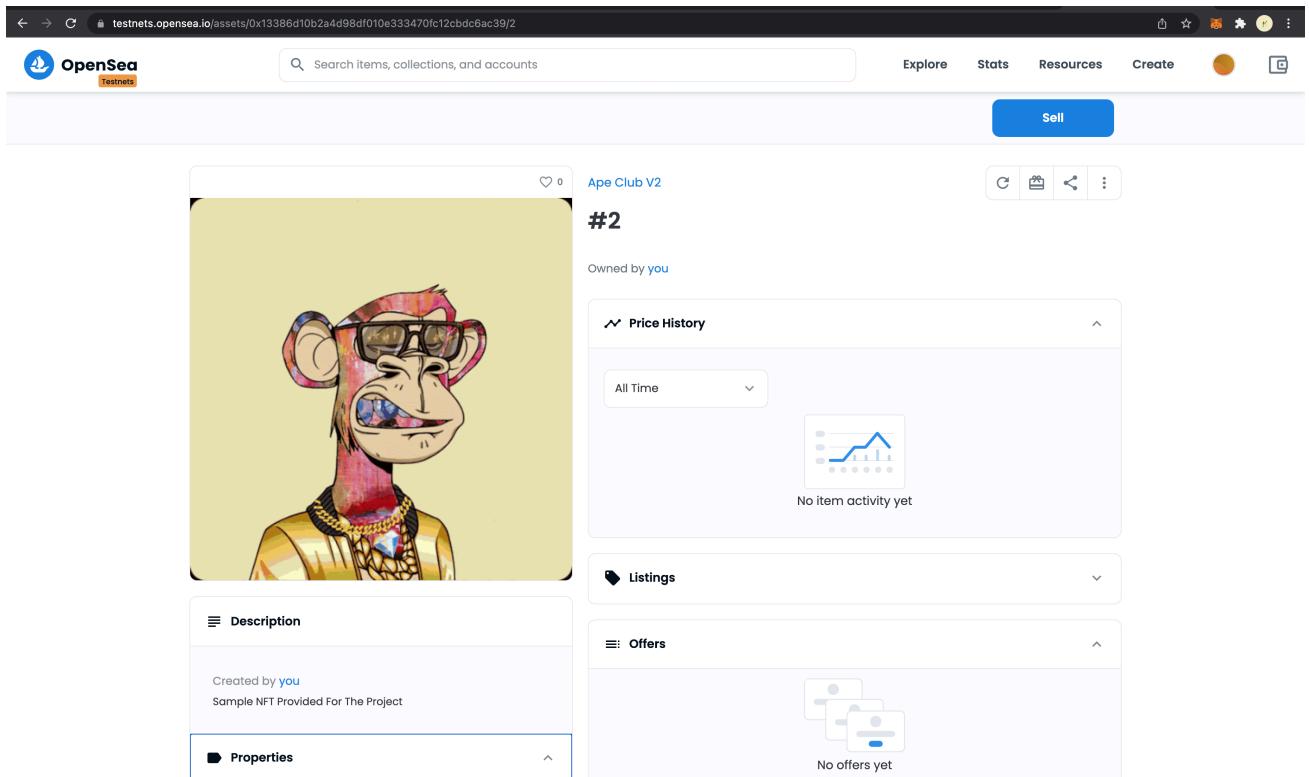
```

src > redux > blockchain > blockchainActions.js > connect > SmartContractObj
29
  return {
29
  type: "UPDATE_ACCOUNT",
30  payload: payload,
31}
32
33
34
35 export const connect = () => {
36  return async (dispatch) => {
37    dispatch(connectRequest());
38    const { ethereum } = window;
39    const metamaskInstalled = ethereum && ethereum.isMetaMask;
40    if (metamaskInstalled) {
41      Web3EthContract.setProvider(ethereum);
42      let web3 = new Web3(ethereum);
43      try {
44        const accounts = await ethereum.request({
45          method: "eth_requestAccounts",
46        });
47        const networkId = await ethereum.request({
48          method: "net_version",
49        });
50        // const NetworkData = await SmartContract.networks[networkId];
51        if (networkId === 4) {
52          const SmartContractObj = new Web3EthContract(
53            SmartContract,
54            "0x13386d10b2a4d98d010e333470fc12cbdc6ac39"
55            //"/0xcd850B1968317557AbB78f7Dfde2fc79219775F"
56            //"/0x827acb09a2dc20e39c9aad7f7190d9bc53534192"
57          );
58          dispatch(
59            connectSuccess({
60              account: accounts[0],
61              smartContract: SmartContractObj,
62              web3: web3,
63            })
64          );

```

2.) Run the project again with 'npm start'.

Connect your wallet and mint an item , it should work absolutely perfectly!



**WE ARE NOW
COMPLETE!**