

# **COMP3334 Computer Systems Security**

## **Group Project**

**(2021 – 2022)**

## **Security Analysis Report**

### **Group 13**

<b>Name</b>	<b>Student ID</b>	<b>Email</b>
Cheung Sui Wing	21027547D	21027547D@connect.polyu.hk
Lau Man Chun	21027257D	21027257D@connect.polyu.hk
Kwong Chun Him	21028468D	21028468D@connect.polyu.hk
Cheng Chi Kit	21028079D	21028079D@connect.polyu.hk

## **Abstract**

We are going to build a digital art platform, a NFT trading platform, for representing, storing and exchanging digital artwork. Computer security is the priority in building a system which involves trading. Below, we will perform a through security analysis for the system, and we are going to discuss the integrity, confidentiality, and availability of the system. Moreover, the system may be attacked by other attacker, and we are going to discuss the potential method those attackers may be using, such as phishing attack and reentrancy attack.

## Table of Contents

Security requirement analysis with justifications .....	1
A literature review of similar systems in the market .....	3
System design specification .....	5
System Diagram .....	5
Use case Diagram .....	6
Functional requirements .....	11
Nonfunctional requirements .....	12
Data .....	12
Technologies .....	13
Security Mechanisms .....	13
Security Policy .....	15
Testing .....	16
Installation Guide .....	17
User Guide .....	19
References .....	25

# Security requirement analysis with justifications

## Confidentiality

- User identities and information should not be revealed in order to protect user confidentiality.
- Each digital artwork should only be listed as sold when the owner of the artwork decides to sell it and set the price greater than 0.
- When the owner of the artwork does not put the artwork on the market, the digital artwork should not be displayed on the market and be purchased. The sale is based on the owner's idea.

## Integrity

The first most important security measure when creating a digital artwork platform is the integrity of the artwork. If the artwork can easily be changed by people who do not have permission, the digital artwork creator may not like using our system.

### Data Integrity:

- The digital artwork should be well protected. The system should make all the attributes of the digital artwork uneditable by users besides the owner, including the price, description, and the NFT itself.
- The digital artwork should be first bought first owned, and each digital artwork can only be owned by a single person so that the digital artwork ownership can be protected, and the uniqueness and the integrity of the digital artwork are protected.
- The transactions record of NFT should be transparent and can be seen, ensuring the validation and integrity of each transaction.

### Origin Integrity(Authentication):

- The system should ensure the source of the digital artwork is created from owner issued/held. Ensure the source of the NFT is believable, authentic, and complete.

## Availability

- In its operational state, the system should be capable of supporting heavy loads. As a result, several users can manufacture digital artworks, acquire digital artworks, and sell digital artworks at the same time without the system becoming unworkable or negatively impacting users.
- All functions in the system should be available for available users. All digital artworks should be available for authorized users, such as digital artworks that are for sale should be listed for everyone. Users can also create their digital artworks using the system.
- All performance interacts with smart contracts and blockchain. Unless the chain getting 51% attack or no one being the verification node of the chain, we can

still access the data or call smart contract on chain. Although the user experience will be poor if our frontend is shutdown by getting DDoS.

- The artwork stored in the NFT contract should be accessed and seen even if the system is not available. Therefore, the image should store in a decentralized storage system such as IPFS. The artwork in IPFS can be preserved and accessed forever theoretically. Also, the image should not be encrypted. If our system is unavailable, even if the NFT owner has owned the artwork, the owner can only access the encrypted artwork on the IPFS. The owner can't decrypt the artwork, which means the artwork's not available. The NFT becomes meaningless. Also, the owner of the NFT may not sell the NFT on other marketplaces since other marketplaces do not have the private key to decrypt the artwork after getting the metadata from the NFT contract. Therefore, to ensure the availability and the value of the NFT, the artwork should store in IPFS without encrypting.

### **Security Policy**

The system should define a set of statements defining what is and is not permitted. The NFT marketplace's objectives, rules, and procedures for restricting access to the system and its data should be precise, exhaustive, and well-defined. A sound security policy protects more than just data and systems. Additionally, it acts as a visible statement of the NFT marketplace's commitment to external security.

### **Security Mechanisms**

#### **MetaMask**

We may use MetaMask to connect the user to the system for all the authentication, and there is no need to create an account on our system, and there have not been any security incidents regarding MetaMask. Also, the wallet is encrypted in the user's browser with a password. The concerns for security are people losing their seed phrase or private key by phishing attacks or the computer being hacked by others and uploaded the key to the cloud, and so on.

#### **File storage**

If the tokenURI is pointed to the centralized server. The server owner can respond to different files according to the request address. Even if it is the same NFT, it can become different documents in the hands of different holders.

#### **Blockchain**

51% attack, When a group of people owned 51% of the total hashing power of that chain. The attacker can have the power to modify the transaction. Ethereum Classic is an example affected by this attack.

#### **Prevent phishing attack**

A phishing attack is an attack where the attacker sends a fake website or program to the user in order to gain access to their account or other information, most cases are, stealing approval to steal crypto coins and asking the user to input their seed phrase/private key of their wallet. The system should remind the user not to click suspicious links or websites.

## **Smart contract security**

### **Prevent reentrancy attack**

A reentrancy attack is an attack in which someone uses a callback to recursively call a contract while it is still executing and before the balance is updated. The attacker may steal all the funds in a wallet when doing this. The solution is to import an external function called ReentrancyGuard to help prevent reentrant calls to a function. In our case, only the contract owner can withdraw funds.

### **Logic error**

Due to the non-tamperable nature of blockchain, once a smart contract is deployed, it cannot be modified again, unlike previous centralized software. Every error in logic can result in permanent loss of NFT or money.

## **A literature review of similar systems in the market**

### **Introduction**

NFTs have been around for many years, there are different companies or people have taken advantage of them and tried to create a business by opening an NFT trading platform. Opensea and Looksrare will be used for literature review, and below will describe what security issues happened in NFT Marketplaces and the security measures they are using, how different their websites work compared to each other, and so on.

### **Main Body**

#### **NFT Marketplace Structures**

The NFT marketplace, such as Opensea and Looksrare, is fundamentally analogous to traditional art marketplaces. They are primarily composed of two parties: creators/owners/sellers and purchasers. NFTs are the things for sale in Opensea and Looksrare. They are collections of artwork, music, games, and so on, whose trade is enabled by the underlying distributed ledger, which maintains consistent records of the transaction history. NFT tokens are one-of-a-kind and vary in value according to their rarity, quality, and creators' reputation. Users can bid or buy NFT collectibles on Opensea and Looksrare NFT marketplaces. When an NFT is sold or resold, its metadata and ownership information are added to a new block on the blockchain, therefore recording and preserving the ownership history [1].

The topic of counterfeiting in the traditional art markets has long been a source of contention. The most significant aspects of such NFT marketplaces are that the ownership chain is well-structured, as opposed to traditional art markets, and that fraudulent events are difficult to perpetrate. NFT tokens are created and traded on top of a decentralized blockchain, ensuring that their ownership cannot be readily tampered with. As the NFT tokens cannot be readily tampered with, the information on provenance, such as facts about an artwork's genealogy, is difficult to tamper with, this may boost public trust in the artwork's authenticity and assist in identifying forgeries.

NFTs provide this functionality backed by the underlying blockchain to handle the counterfeiting of artworks, closing the traceability gap and potentially increasing the system's trustworthiness [1]. As a result, the NFT marketplaces can provide much more security than traditional art markets.

Security Issues happened in NFT marketplaces, and the strategy the marketplace has taken to face these issues

As for the security concern, NFT Marketplaces' integrity (data integrity and owner integrity) is very secure, implying that NFT Marketplaces has few integrity-related security vulnerabilities. The security problem with NFT marketplaces is usually phishing.

One of the incidents that have happened was a phishing attack, in which at least 32 users had lost their NFTs worth \$1.7 million [2], and OpenSea stated that the phishing attack originated outside of OpenSea's website. According to Cybavo's report [2], OpenSea uses off-chain signatures that make the trade fast and easy. During that time, OpenSea is updating its smart contract from version 1 to version 2.3 on the Wyvern Protocol. The purpose of the OpenSea migration is to resolve inactive listings of old NFTs, and they intend to do so by upgrading to a new contract. They announced on social media and user alerts that all user's Ethereum listings require to be "migrated" to the new smart contract. At the time of the hack, Hackers took advantage of the update procedure and opted to swindle NFT users by resending the identical email format used by OpenSea to its victims. The hacker uses phishing attacks to steal these signatures. When the NFT users click the link of the phishing email, it opens a phishing website and asks them to sign a transaction. The NFT users that signed the transaction will send the NFTs to the attacker. This caused this loss.

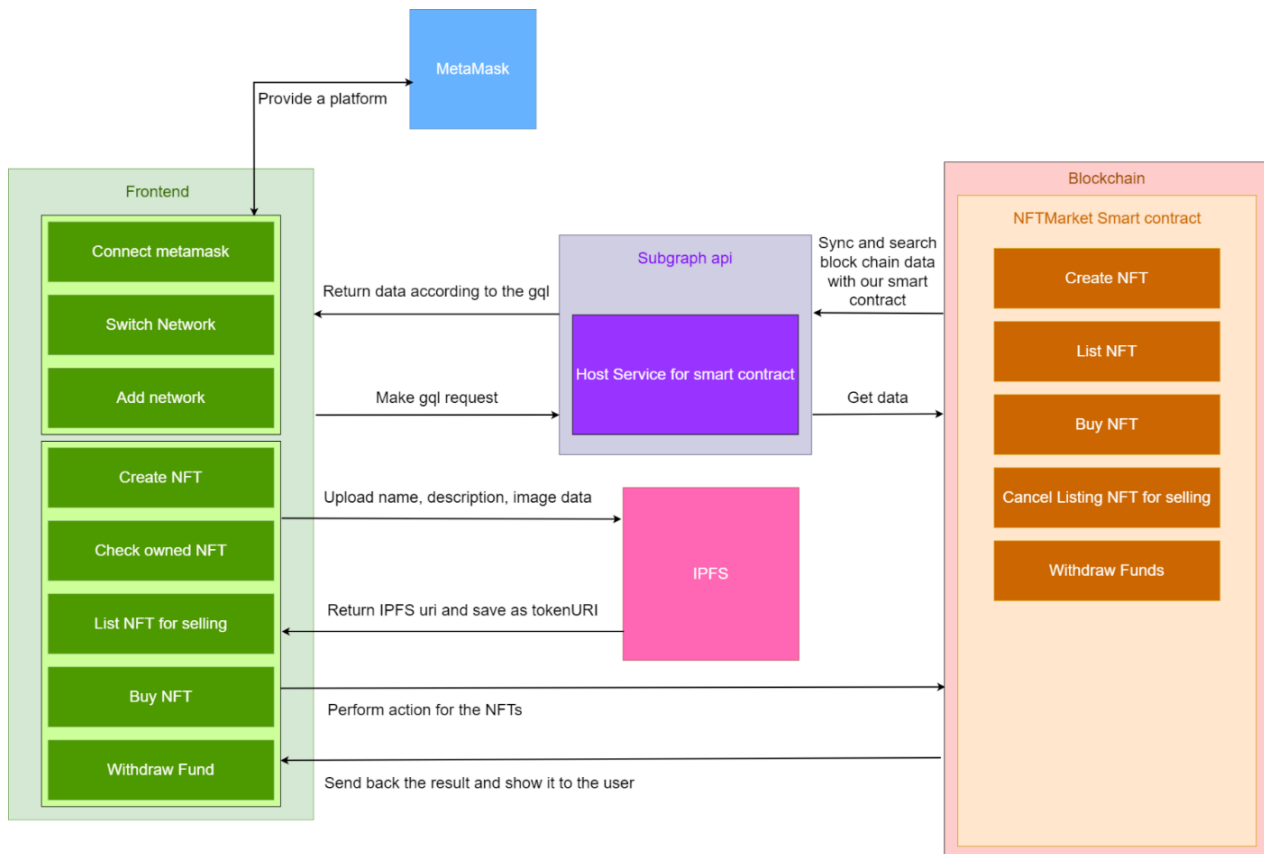
## Conclusion

The most common security flaw that can happen seems to be a phishing attack, where the user clicks a suspicious link or purposely goes to a suspicious website, which is hardly preventable by the company that owns the website. Some of the websites repeatedly remind the user not to click suspicious websites in places that are easily viewable such as the chatbox with other users, the Q&A pages, and so on. The NFT trading platform should inform any new users with proper security information when they first enter the NFT world.

# System design specification

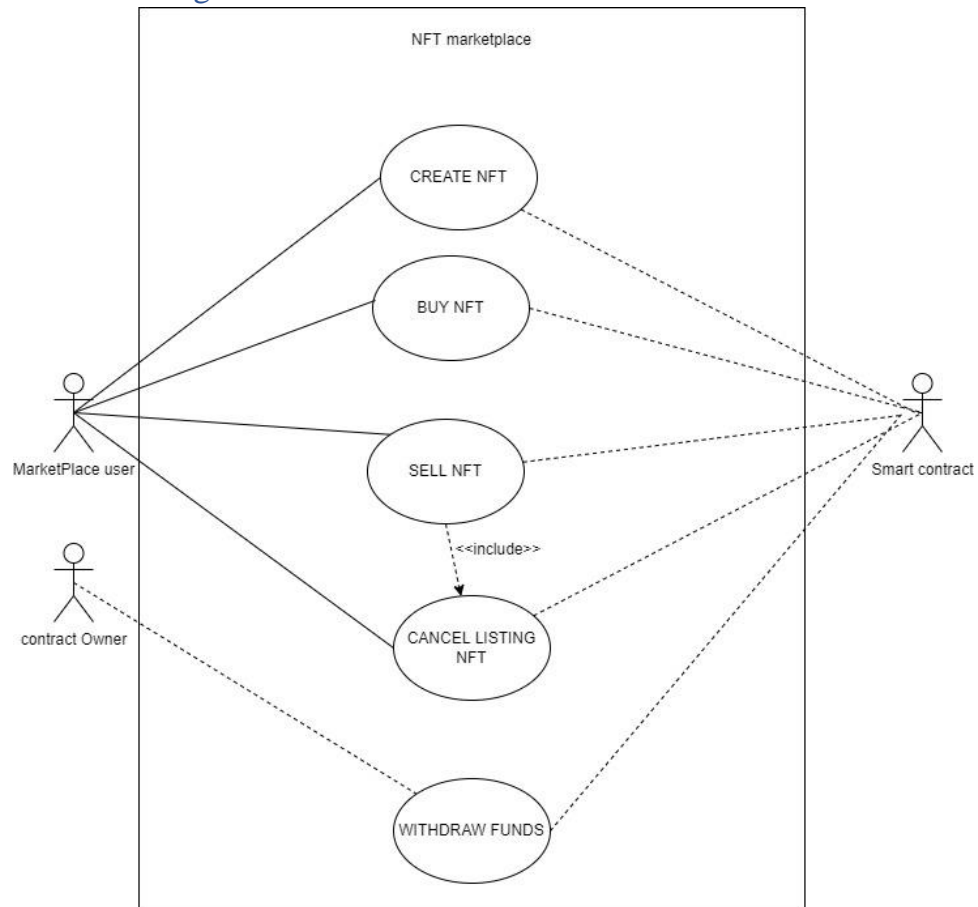
This system only works on Polygon Mumbai Testnet.

## System Diagram

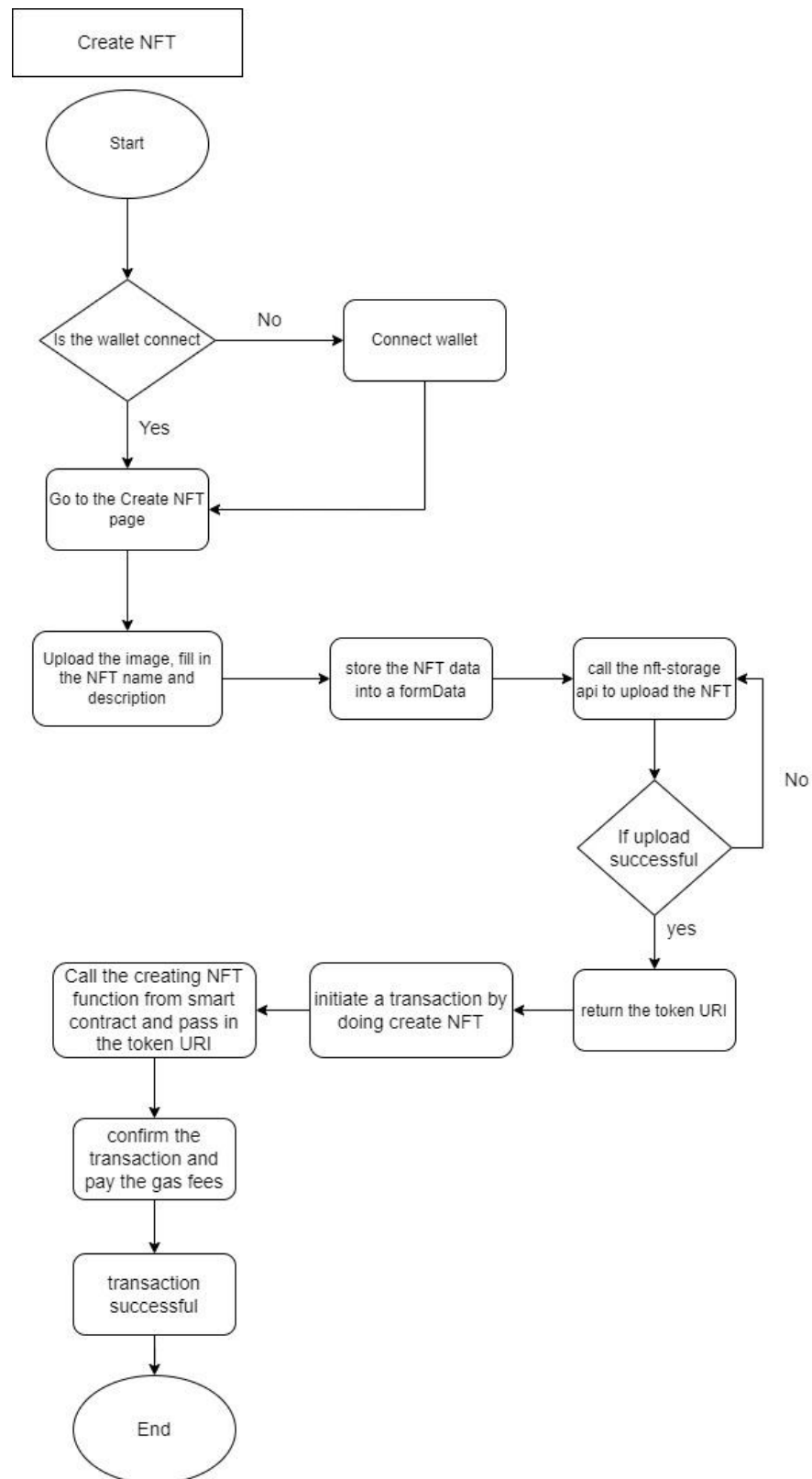




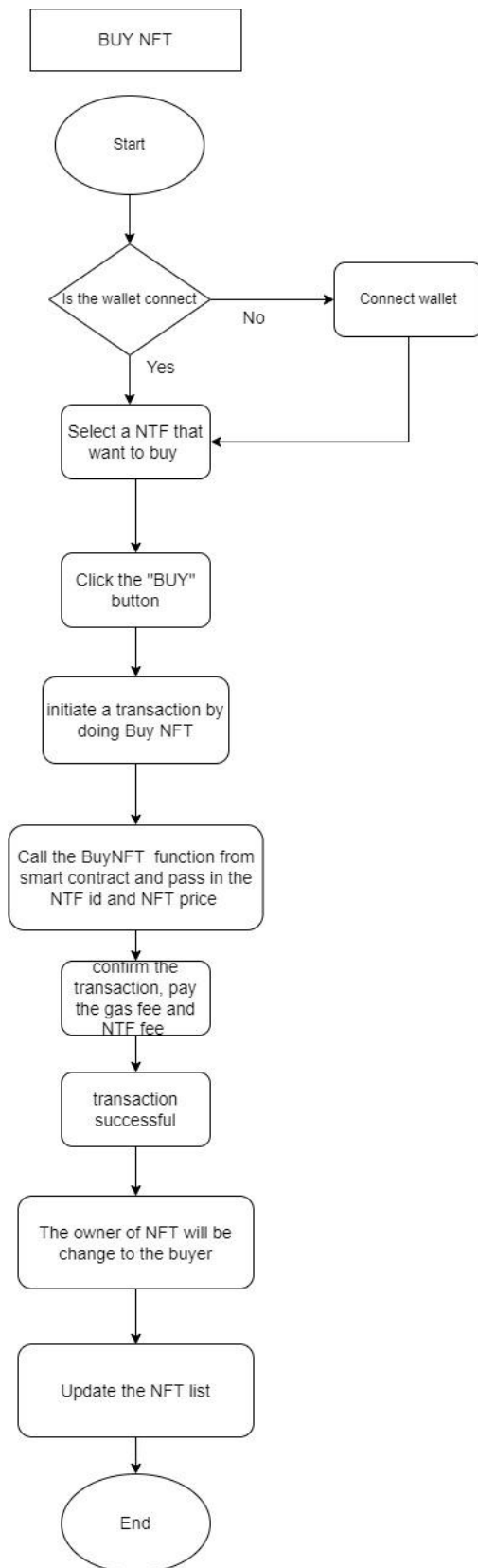
## Use case Diagram



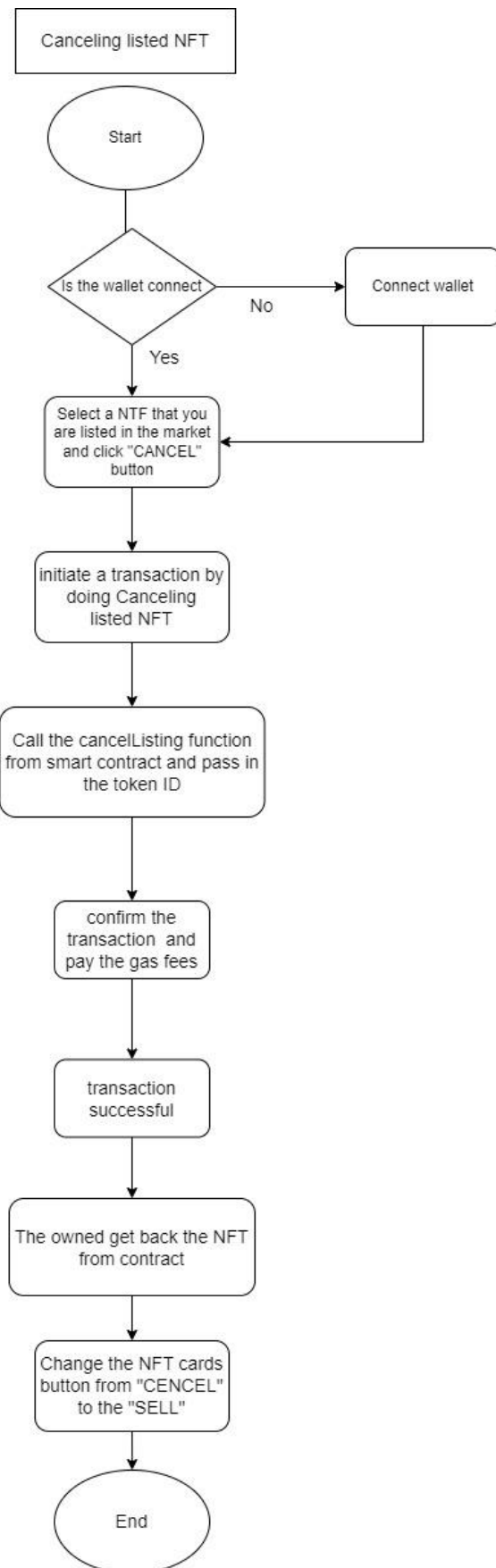
## Create NFT



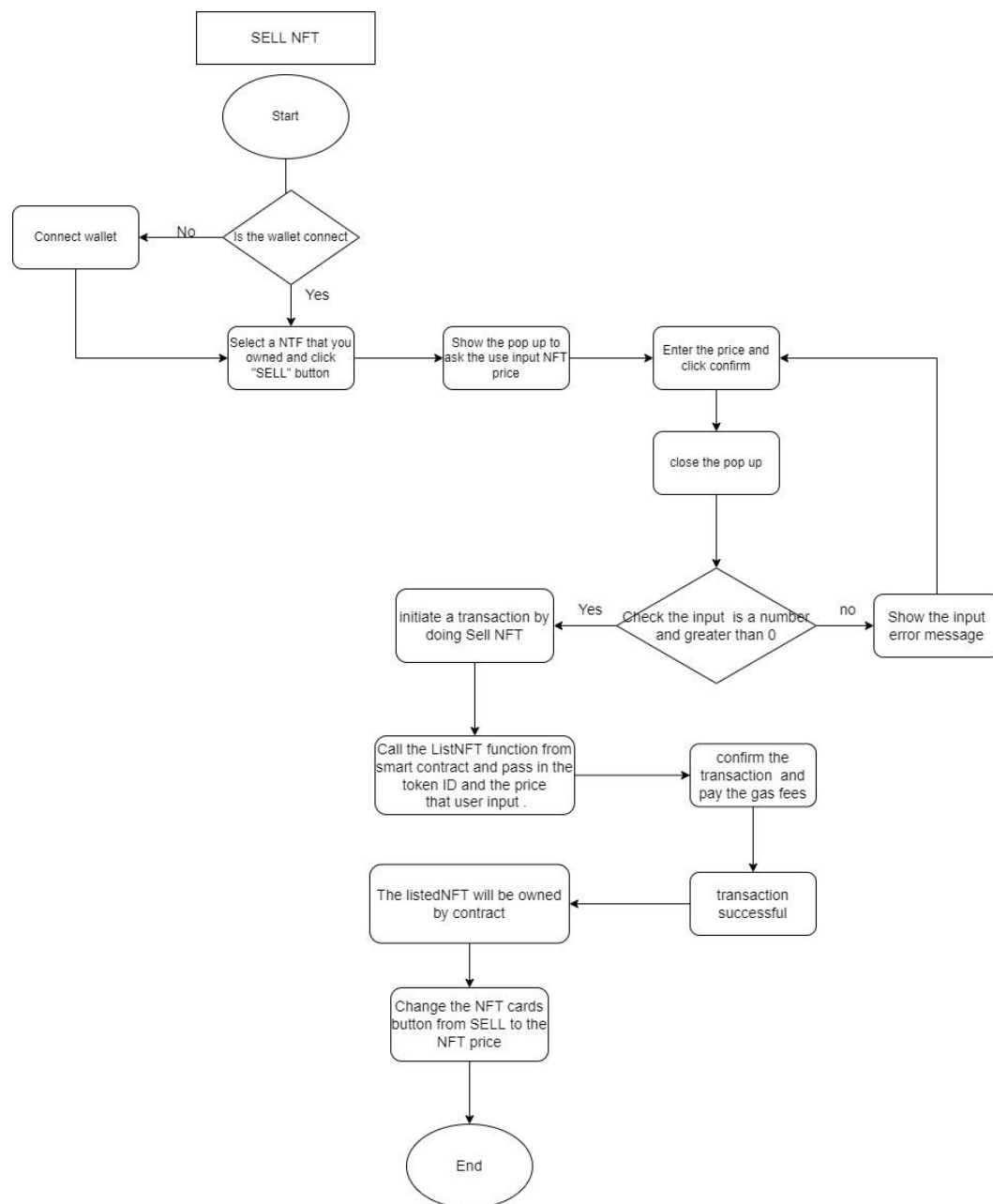
## Buy NFT



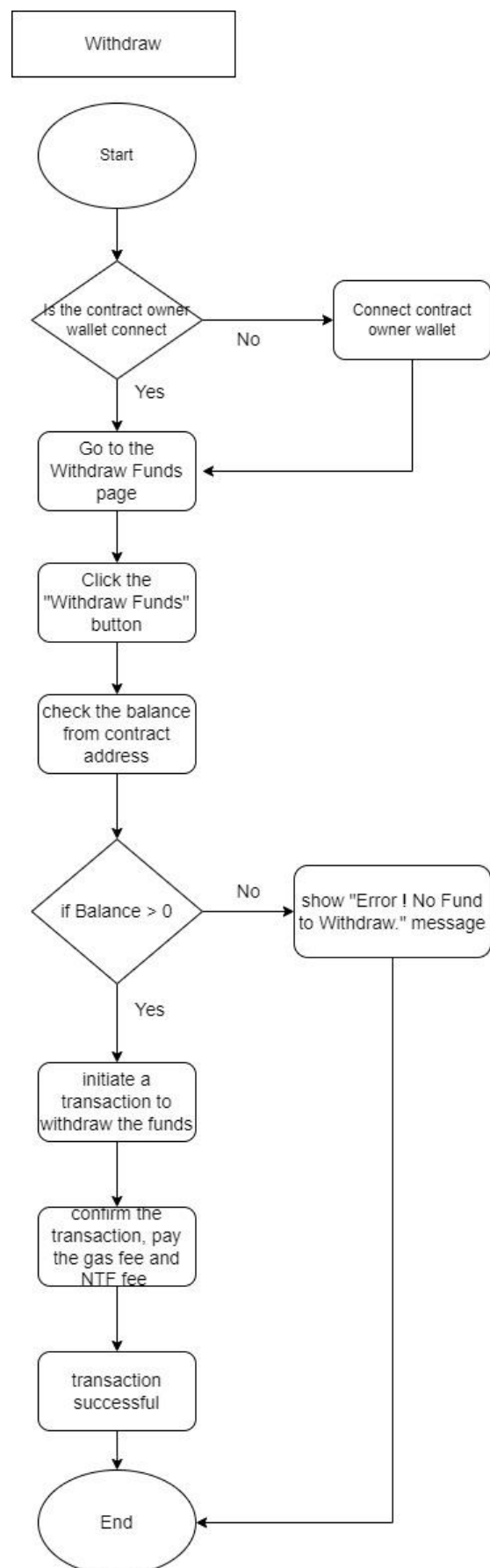
## Canceling listed NFT



## Sell NFT



## Withdraw Funds



## Functional requirements

Frontend:

Functions	Description
Connect MetaMask	Allow user to connect their crypto wallet
Switch Network	Check if the current wallet network is Polygon Mumbai. If no, show the button for a user to switch the network.
Add network	Add the polygon Mumbai network to the MetaMask automatically if the user has not set the network
Create NFT	Allow user to upload gif/jpg with name and description, then create an ERC721 token.
Check owned NFT	List all NFT tokens that the user owned
List NFT for sell	User can sell their NFT by setting the price >0
Unlisting NFT	User can cancel their own listing NFT
Buy NFT	User can buy other listing NFT
Withdraw Fund	Contract creator wallet can withdraw the transaction tax earned in the NFTMarket contract (5% tax for each trade)

NFTMarket Smart contract:

Functions	Description
createNFT	Mint a new NFT token with new ID and token URI, then send to message sender
listNFT	Transfer the NFT from owner to this contract with price
buyNFT	Paying with the price and buy the NFT, the owner transfer from this contract to the message sender(buyer) <b>95%</b> of the price will transfer to the seller <b>5%</b> of the price keep in the contract as the tax
cancelListing	Set the NFT price to 0 and transfer the NFT owner from this contract to the original seller
withdrawFunds	Transfer the contract balance(tax earned) to contract creator

## Nonfunctional requirements

Functions	Description
Portability and compatibility	<ul style="list-style-type: none"><li>All browsers that support MetaMask can use the system.</li></ul>
Performance	<ul style="list-style-type: none"><li>Transaction speed should be fast</li></ul>
Usability.	<ul style="list-style-type: none"><li>Easy for the user to use.</li></ul>
Cost.	<ul style="list-style-type: none"><li>Low gas fee</li></ul>
Localization	<ul style="list-style-type: none"><li>The NFT marketplace uses English, which is one of the most used languages in the world. Therefore, more people will be able to understand.</li></ul>
Accessibility.	<ul style="list-style-type: none"><li>The user can still go to the blockchain to find their NFT(s) if our system is down</li></ul>

## Data

ERC721 token data

Name	Description
id	Token id
owner	Token owner address
price	Selling price. (0 mean not selling)
tokenURI	Ipfs URI for getting the NFT metadata

Example:

```
/:
  id: "10"
  owner: "0x91102eb08495cb472345c8a0124bee8d431597c0"
  price: "20.0"
  tokenURI: "ipfs://bafyreiczqha6u2u6g4csjxamnz5mqj7gejq74er2i7t27dodwvgw7mamye/metadata.json"
```

NFT metadata store in IPFS

Name	Description
Name	NFT name
Description	NFT description
image	NFT image uri in IPFS

Example:

<https://ipfs.io/ipfs/bafyreiczqha6u2u6g4csjxamnz5mqj7gejq74er2i7t27dodwvgw7mamye/metadata.json>

```
{"name": "poly_jump", "description": "poly_jump", "image": "ipfs://bafybeig5k4mqhpdzhmtm42jv76q7jf625z6vxy537x7xrvngbqqz116xu/POLYU_JUHP.png"}
```

## Technologies

Name	Description
React/Next.js/ES7	Framework/Programming language used to develop the frontend website.
MetaMask	Crypto wallet
Polygon Mumbai Testnet	Polygon testnet (Use it because it's cheap and easy to get test tokens)
Polygon Faucet	Website to get the test tokens (MATIC)
Hardhat	It is an environment for testing the smart contract locally, and it can deploy the smart contract to the target network.
Solidity v0.8.13	Programming language used to develop the smart contract
Openzeppelin	Library to build secure Smart Contracts
IPFS service (nft storage)	Service for storing the NFT data
Subgraphs	It is a decentralized protocol for indexing and querying data from blockchains. It allows our react app to query data directly
Apollo	The platform helps us to build schema and query that use to interact with the subgraphs API.

Our Subgraphs hosted service link: <https://thegraph.com/hosted-service/subgraph/csw0126/comp3334>

## Security Mechanisms

In order to fulfill the requirements mentioned above, these mechanisms should be adopted in the implementation.

### Using blockchain technology:

- The data in the chain cannot be altered. There is no need to worry about the database being hacked or altered.
- Anonymity. Since no account is required, there is no need to save the username or password, as is the case with other trading platforms. Each user maintains his or her own wallet (seed phrase/private key). The user's privacy is protected and maintained, and there is no need to worry about storing account numbers, passwords, or other user information securely.
- Transparency, all transactions can be seen on the blockchain. High transparency, no deception.
- Traceability, anyone can trace back to the original creator of the NFT to check if it is genuine or not.

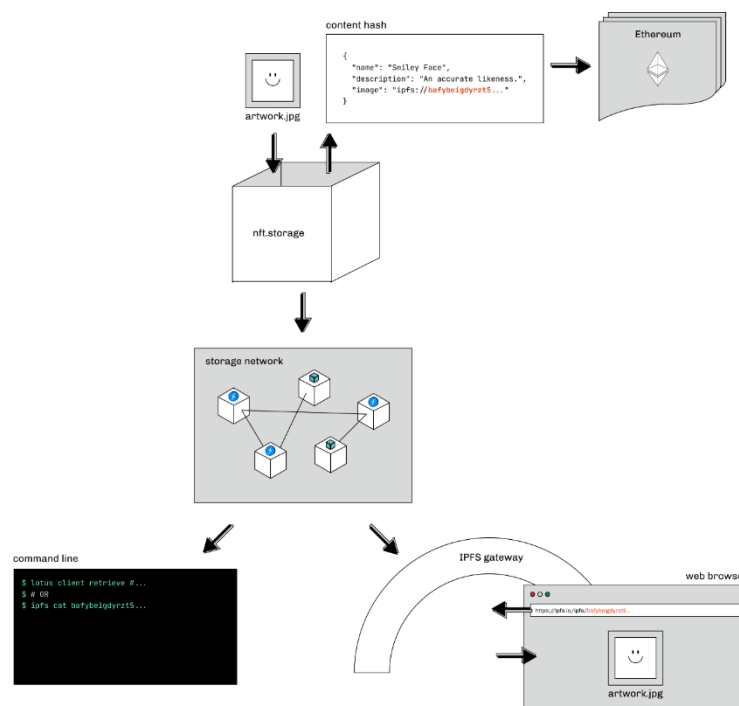


### Using Smart contract:

- Without the intervention of a third-party organization, it can ensure openness and transparency, the encrypted records of transactions are shared among participants, and the data cannot be changed
- To prevent human operation, it will only follow the code that has been set.

### Using InterPlanetary File System (IPFS)

- High security, as no cloud storage or local storage, can guarantee that its server will not be attacked by hackers and ensure data security. Under the IPFS protocol, the file will keep its records at each node when it is uploaded, and the system will automatically restore the file when it detects that the file is lost. And due to its distributed storage characteristics, hackers cannot attack all nodes at the same time.
- Untamperability and Immutability. Since the domain name used to address the stored file contains the hash of the file, it is as tamper-evident as the blockchain. This can ensure that all data stored in IPFS is integrity. As a result, more in line with the spirit of cryptographic primitives.
- Hash value  
Every file uploaded to IPFS will use a hash value for validation and searching to ensure that the file being downloaded is the correct one.



Source: <https://nft.storage/images/diagram-store-and-retrieve@2x.png>

- As all the data is not stored in a database, like a typical website, they are stored in the IPFS, no data can be changed easily, the integrity can be highly secure, and there is nothing to hack on the server-side.

## Openzeppelin

Their purpose is "Build Secure Smart Contracts in Solidity". And we use this library to build our NFT Market contract.

- ERC721URIStorage.sol  
Allow us to inherit the ERC721 standard implemented by OpenZeppelin  
Some functions that we use:
  - \_safeMint: Prevents NFT minting into contracts that cannot handle ERC721 tokens
  - safeTransferFrom: check the "to" address can or cannot receive ERC721 token, prevent token lost
- Counters.sol  
Provides counters that can only be incremented or decremented by one. Prevent using +/- to calculate the value which would cause overflow vulnerability.  

```
_tokenId.increment();
```
- SafeMath.sol  
Provides counters that can only be incremented or decremented by one. Prevent using +/- to calculate the value which would cause overflow vulnerability.
- Ownable.sol  
onlyOwner, only the contract owner can call this function to withdraw the tax earned. Prevent unauthorized use.

```
// withdraw funds
function withdrawFunds() public onlyOwner {
    uint256 balance = address(this).balance;
    require(balance > 0, "balance is zero");
    payable(msg.sender).transfer(balance);
}
```

## Security Policy

- The user requires a MetaMask plugin with their own crypto wallet to perform basic actions besides viewing all the NFTs that are listed in the system, those actions create NFT, buy NFT, or sell NFT.
- MetaMask must be under the correct network (Polygon Mumbai in our project). The webpage will display normal content, such as what NFTs are on the market. Otherwise, a button will be displayed to let the user add and switch to the correct network. This prevents bad requests such as wrong network transfers from being sent.
- Only the owner of an NFT token can sell the NFT at their desired price. When an NFT is bought by another person, the right to sell the NFT is transferred to

the person who bought it. The one that can collect the platform fee is the owner and creator of the system.

- Only the contract owner can withdraw the tax earned.
- The buyer sent value should equal to the NFT price in order to buy the NFT
- Cookies and sessions should be used for a better user browsing experience.

## Testing

Use hardhat to test the smart contract locally.

Test file: **/nft-marketplace/test/index.ts**

Testing result:

```
D:\git_repo\COMP3334_GroupProject\nft-marketplace>npx hardhat test

NFTMarket
  CreateNFT
    ✓ should create NFT with correct owner and uri (56ms)
  listNFT
    ✓ should revert if price is 0 (45ms)
    ✓ should revert if not call by owner (50ms)
    ✓ should list the nft for sell (64ms)
  buyNFT
    ✓ should revert if NFT is not listed for sale
    ✓ should revert if NFT if buy with different price (84ms)
    ✓ should transfer the owner to the buyer (113ms)
  cancellisting
    ✓ should revert if NFT is not listed for sale
    ✓ should revert if the caller is not the seller of the listing (68ms)
    ✓ should transfer the ownership back to the seller if all requirements are met (90ms)
  withdrawFunds
    ✓ should revert if called by a signer other than the owner
    ✓ should transfer all funds from the contract balance to the owner's (149ms)
    ✓ should revert if contract balance is zero

13 passing (2s)
```

# Installation Guide

## Folder structure

```
-
├─ Comp3334          # Subgraph abi setting folder, not important
├─ nft-marketplace   # Main folder contain all source code
│   └─ test          # folder contain the hardhat test case code
│   └─ src            # folder contain React source code
│   └─ contracts      # folder contain the NFTMarket.sol code
│   └─ .env           # environment variable, including the private key of the contract owner
│   └─ ...
└─ ...
```

Install MetaMask in Chrome online store:



[https://chrome.google.com/webstore/detail/MetaMask/nkbihfbeogaeaoehlefnkodbefgpgknn?utm\\_source=chrome-ntp-icon](https://chrome.google.com/webstore/detail/MetaMask/nkbihfbeogaeaoehlefnkodbefgpgknn?utm_source=chrome-ntp-icon)

then follow the instruction to create your own wallet.

Windows10:

1. Install node.js v16.14.2 from <https://nodejs.org/en/>
2. Install Yarn corepack using command :

**corepack enable**

3. Go to nft-marketplace folder:

**cd nft-marketplace**

4. Install all dependences:

**yarn**

5. Test the NFTMarket.sol smart contract:

**npx hardhat test**

6. Run the react application

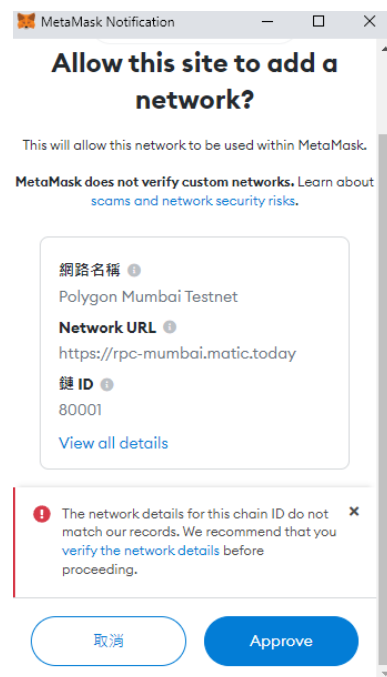
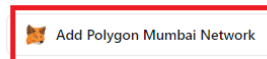
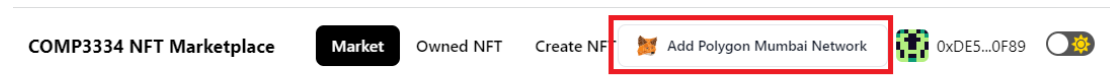
**yarn dev**

7. Then it will show the localhost url: <http://localhost:3000> in this case

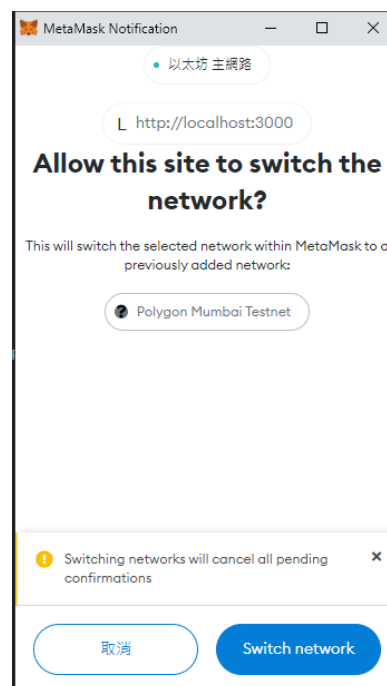
```
ready - started server on 0.0.0.0:3000, url: http://localhost:3000
```

## Add polygon Mumbai network to MetaMask:

If the user is on the wrong network, the user can click the "Add polygon Mumbai Network" button to add or switch networks.

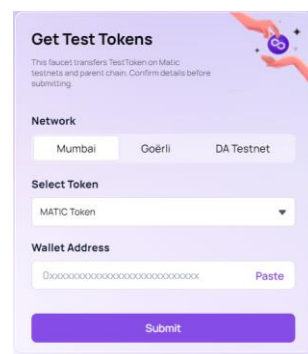


Add network to MetaMask



switch network

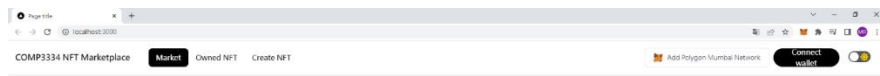
If you do not have a test token, you can go <https://faucet.polygon.technology/>



Input your wallet address and get free testing token.

## User Guide

### Connect Wallet



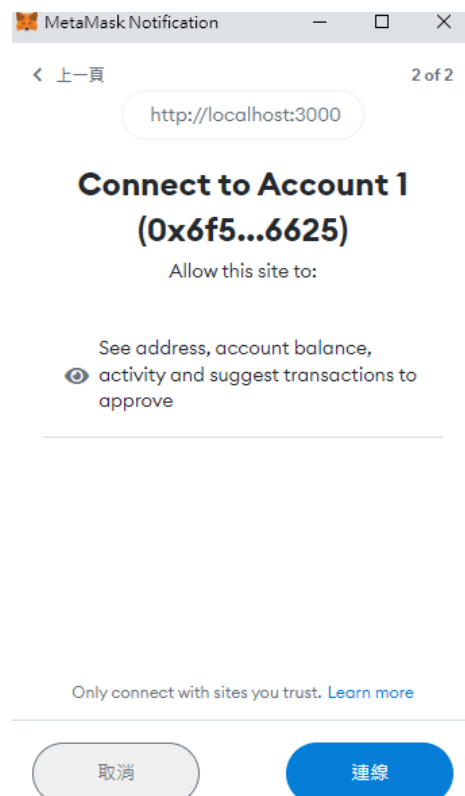
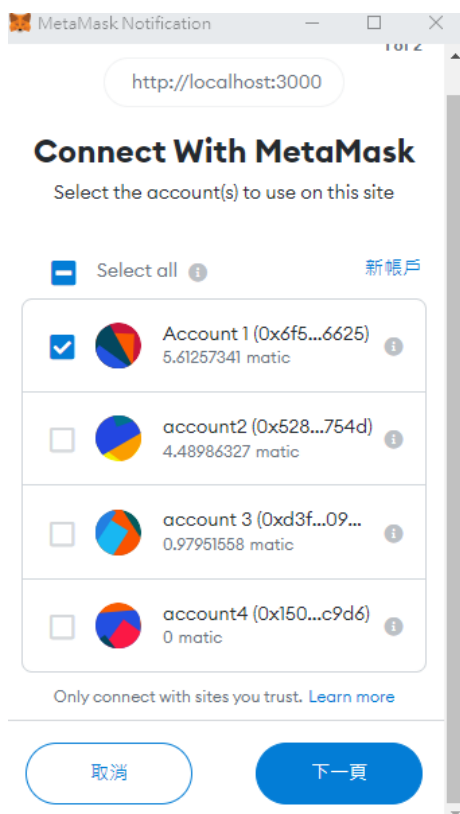
1. Click "connect wallet"

Connect  
wallet

, then click MetaMask icon



to



2. Select the account you want to connect to and click "next page(下一頁)," then click "connect(連線)" to connect your wallet.

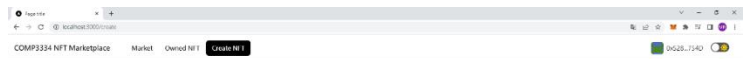
# Create NFT

1. Click "Create NFT" go to the Create NFT page

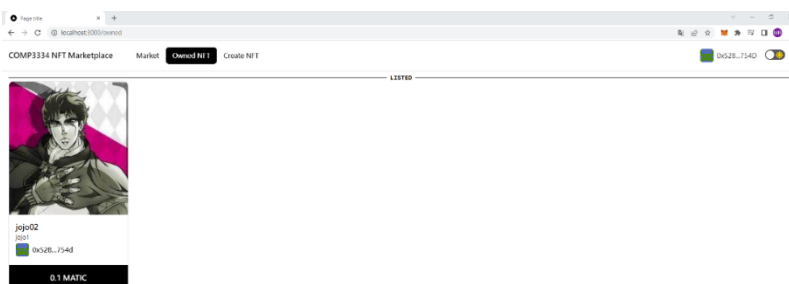
Create NFT

A screenshot of the 'Create NFT' form. It features a large 'Upload' button on the left. To the right are two input fields: 'name' and 'description...'. A 'Create' button is located at the bottom right of the form.

2. Upload an NFT image, fill-in the NFT name and description, then click "Create".

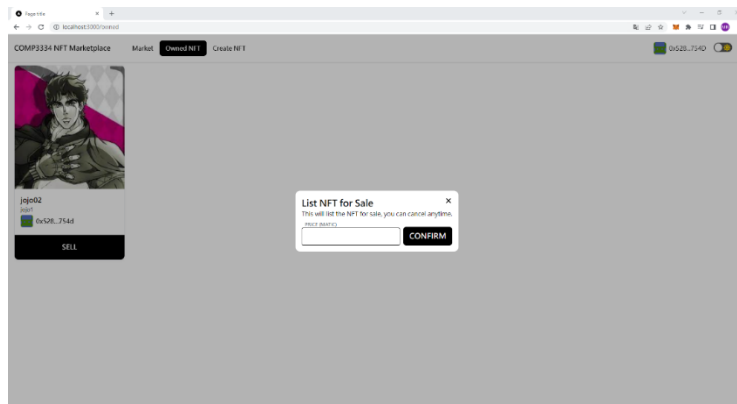
A screenshot of the 'Create NFT' form after an image upload. The image is a digital artwork of a character. The 'name' field contains 'jaja02' and the 'description' field contains 'jaja01'. The 'Create' button is at the bottom.

3. The NFT will be showed on your owned NFT page



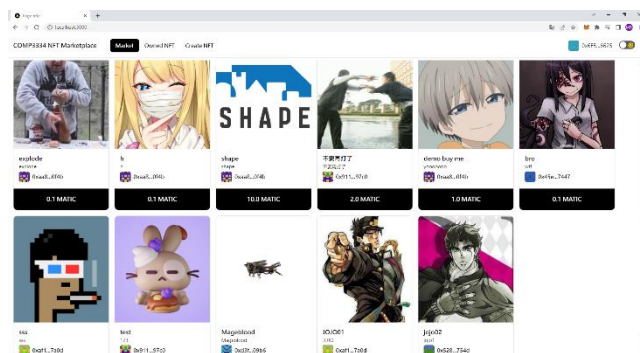
## Sell NFT

1. Click the "SELL" button on a NFT and fill in a price for this NFT, Then click **CONFIRM** button




2. The MetaMask notification will be shown. If not, you can click the "1" icon to open it, then click "confirm(確認)", after the transactions successful

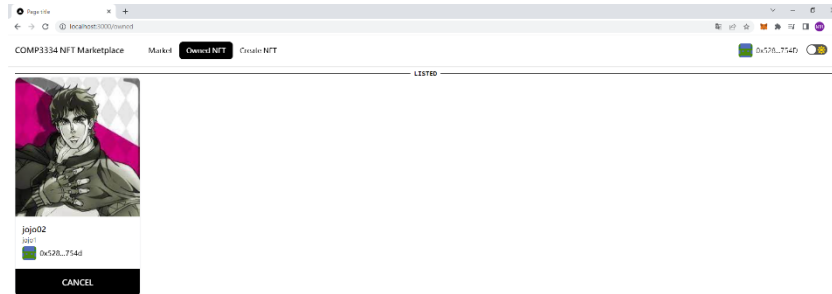
3. Press "F5", and NFT will be shown on the market page.




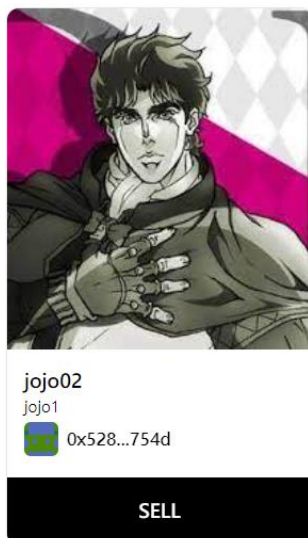


## Cancel Sell NFT

1. Go to Owned NFT page and click the  button to cancel the list of the NFT



2. The MetaMask notification will be shown. If not, you can click the " " icon to open it, then click "confirm(確認)" after the transaction successful




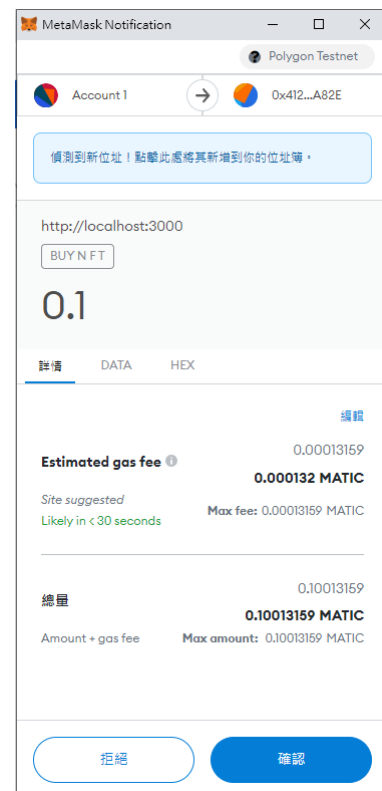
3. Press "F5". The NFT will not show on the market, and you can sell it again

## Buy NFT



1. Choose one NFT that you have enough crypto to buy, move your mouse to the NFT you want to buy, and click "BUY"

2. The MetaMask notification will be shown. If not, you can click the " " icon to open it, then click "confirm (確認)".

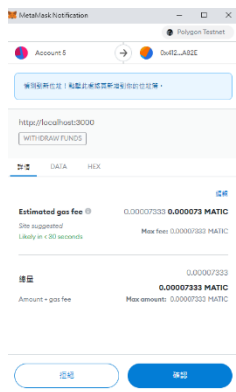
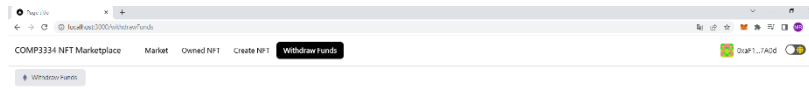


3. After the transaction is successful, then press "F5" the NFT will be shown on your Owned NFT page.

## Withdraw Funds

Withdraw Funds

1. Click button to go to the withdraw funds page

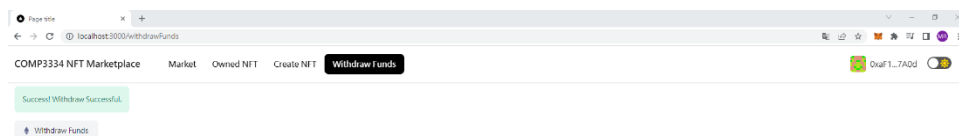


2. Then click the button, and the MetaMask notification will be shown. If not, you can click the



" icon to open it, then click " confirm (確認)."

3. After the transactions are successful, the success message will be shown.



## References

- [1] T. Sharma, Z. Zhou, Y. Huang, and Y. Wang, "'It's A Blessing and A Curse': Unpacking Creators' Practices with Non-Fungible Tokens (NFTs) and Their Communities," 2022. <https://arxiv.org/abs/2201.13233>
- [2] Crybvo, OpenSea Phishing "Hack": What Really Happened and How To Protect Your NFTs, Mar 2, 2022 <https://www.cybavo.com/zh-tw/blog/open-sea-phishing-hack-how-to-protect-your-nfts/>