A PROJECT REPORT
ON

NFTCART

BY

Mandalia Jaynish (CE076) 19CEUBS100
Matroja Seturaj (CE079) 19CEUOS104
Miyatra Yash (CE083) 19CEUBS097

Bachelors of Technology
Semester VI

Subject: System Design Practice

Guided By:

Prof. Pandav K. Patel Assistant Professor

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

CERTIFICATE

This is to certify that the practical/term work carried out in the subject of Software Engineering and recorded in the journal is the bona fide work of

Mandalia Jaynish (CE076) 19CEUBS100
Matroja Seturaj (CE079) 19CEUOS104
Miyatra Yash (CE083) 19CEUBS097
Of B.Tech Semester VI in branch of
Computer Engineering
during the academic year 2021-2022

Prof. Pandav K. Patel
Assistant Professor,
Dept. Of Computer Engg.
Faculty of technology
Dharmsinh Desai University,
Nadiad

Dr. C. K. Bhensdadia
Head,
Dept. of Computer Engg.
Faculty of technology
Dharmsinh Desai University,
Nadiad

Table of Contents

1. Abstract and Introduction

Abstract

Nonfungible tokens (NFTs) have gained popularity in the recent times, generating billions of dollars in transaction volume. The Web3 technology is being widely used in the recent times. The NFTCART is a NFT market place built using Web3 technology and on Ethereum framework. The online platform serves both the buyers' and sellers' community by establishing effective communication. A generic platform where the sellers can put their NFT's on Sale and the buyers can buy them and add to their collections. The metamask wallet is used for the effective payment between both the buyers and sellers.

Introduction

NFT Marketplace is platform designed to facilitate the trade of non-fungible tokens. NFT's are like an asset like property, shares in a company etc. is non-fungible if it's unique.

The NFTCART provides platform to buyers and sellers community where they can buy and sell NFT's respectively. The payment is done securely using the metamask wallet.

Technologies/Tools Used:

- Platform used: Visual Studio code 2021 and powershell for Admin side, Local development server and hardhat testing
- Technology: Next.js, Ethereum, Metamask
- Languages: HTML, CSS, JS, Solidity

2. Software Requirement Specifications

Online Home Services

Manage User:

Sign-in(Web3):

I/p: Enter metamask wallet private key
O/p: You are redirected to home page

NFT:

Sell-NFT:

I/p: You have to choose NFT you will sell

O/p: Service will be put on sale

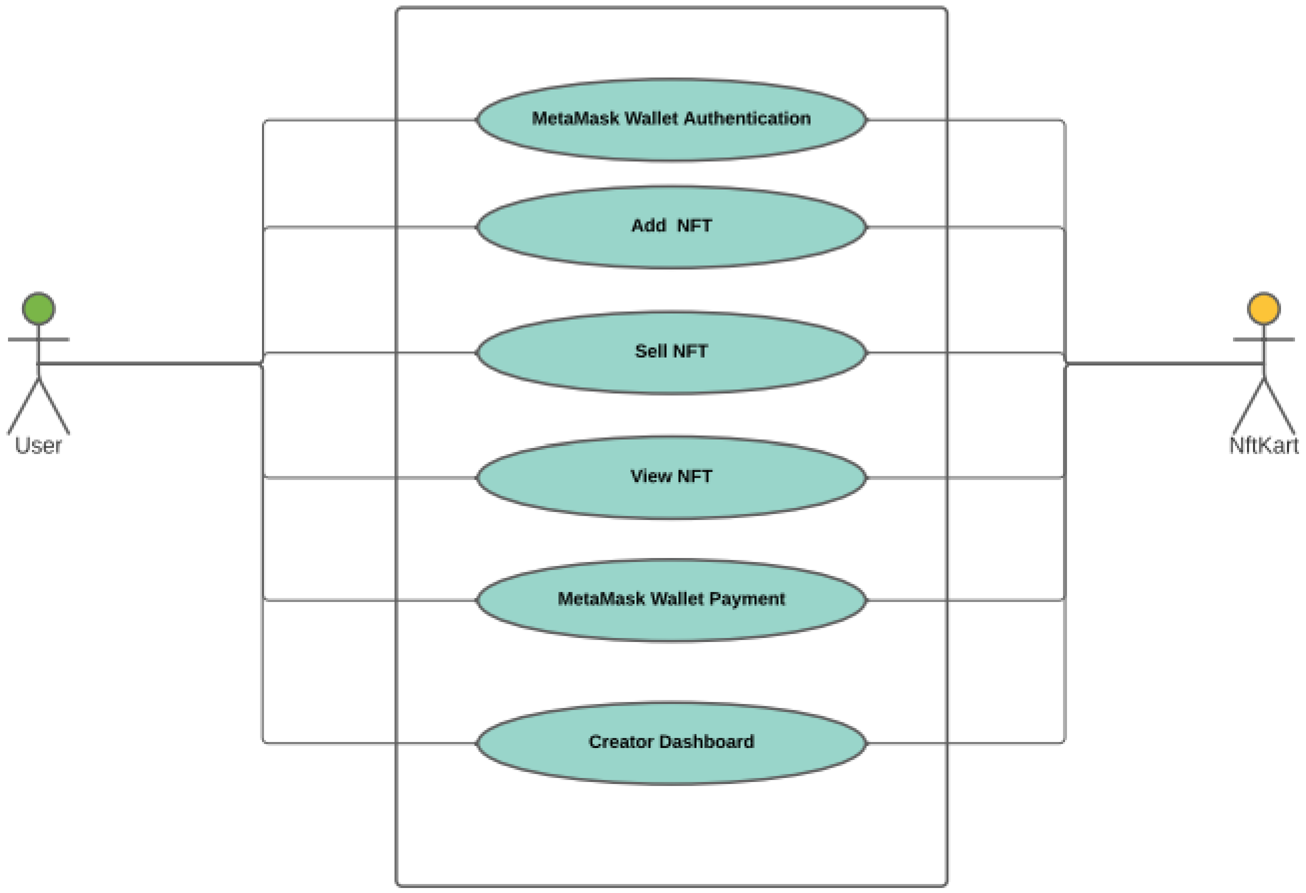
Buy NFT:

I/p: Select the required NFT you want and make payment by metamask wallet

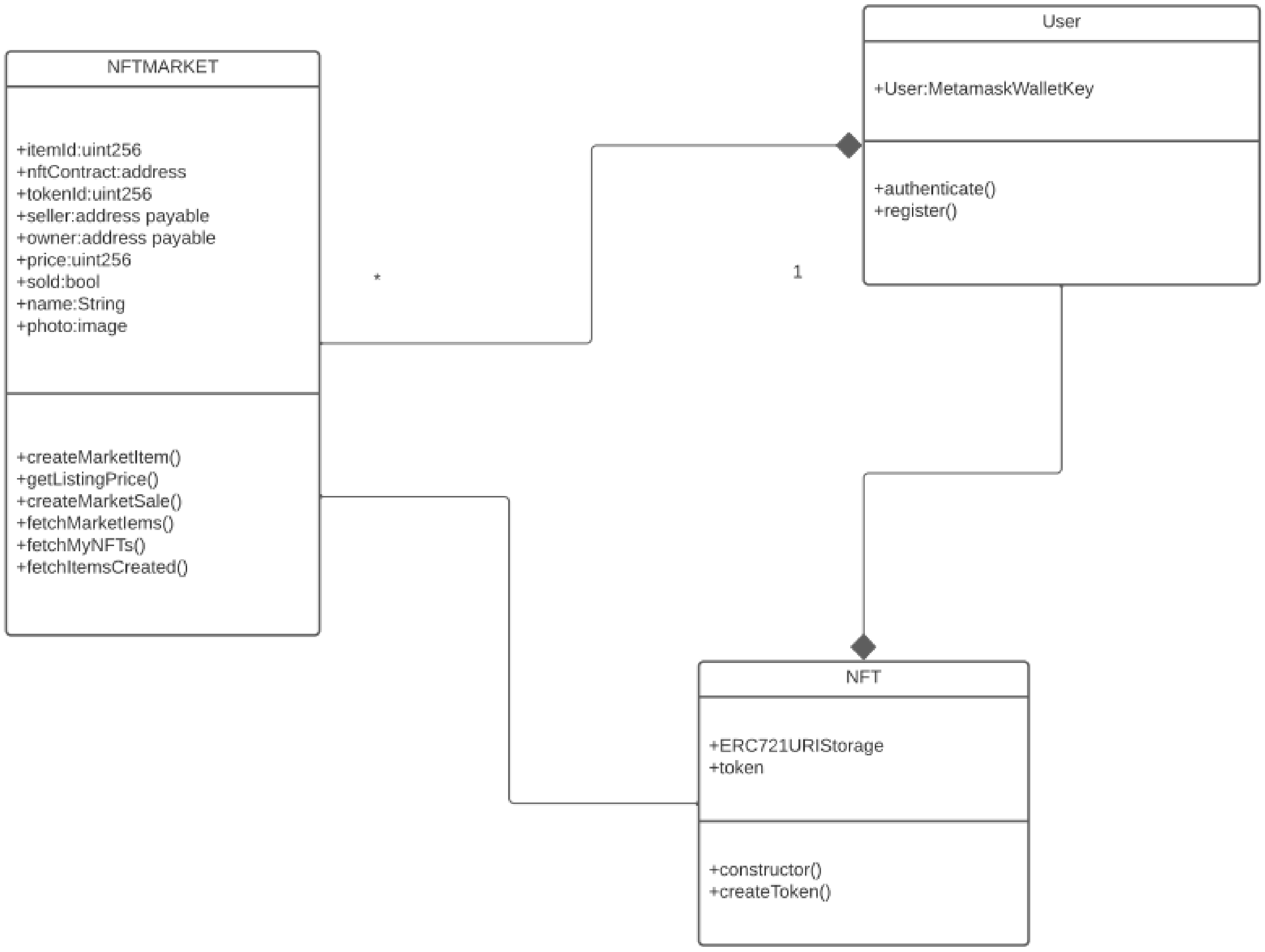
O/p: NFT will be added to the owned items and Ethereum will be deducted from your wallet

3. Design Documents:

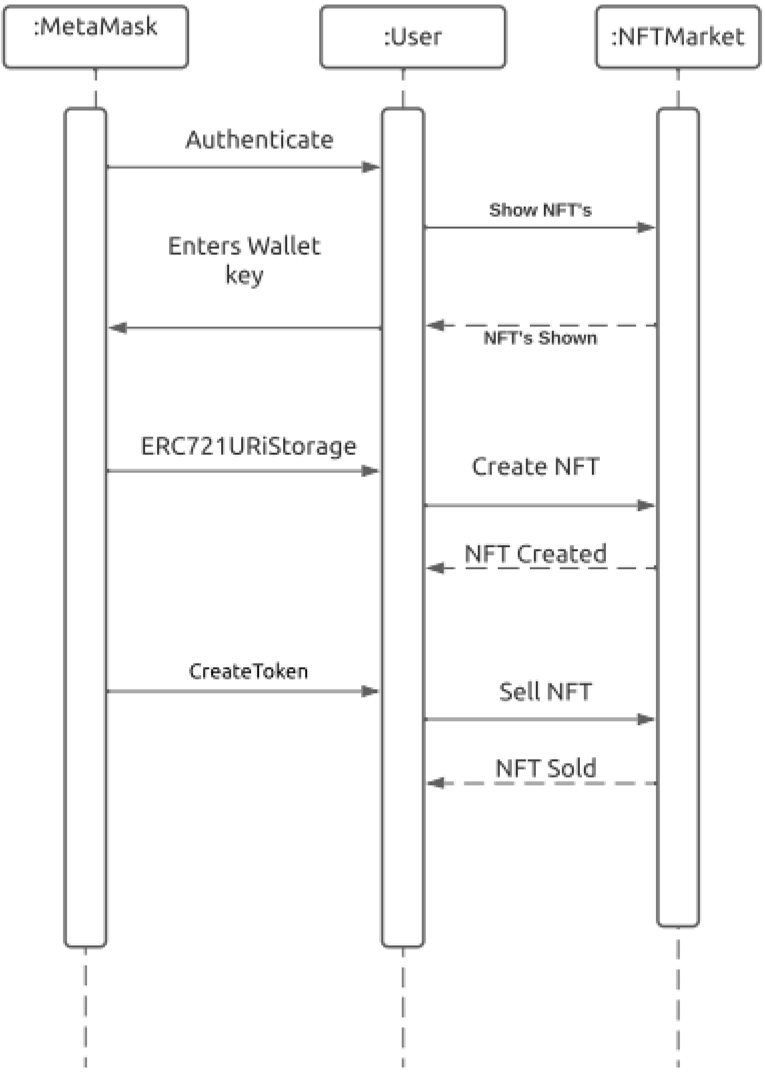
Use Case Diagram:



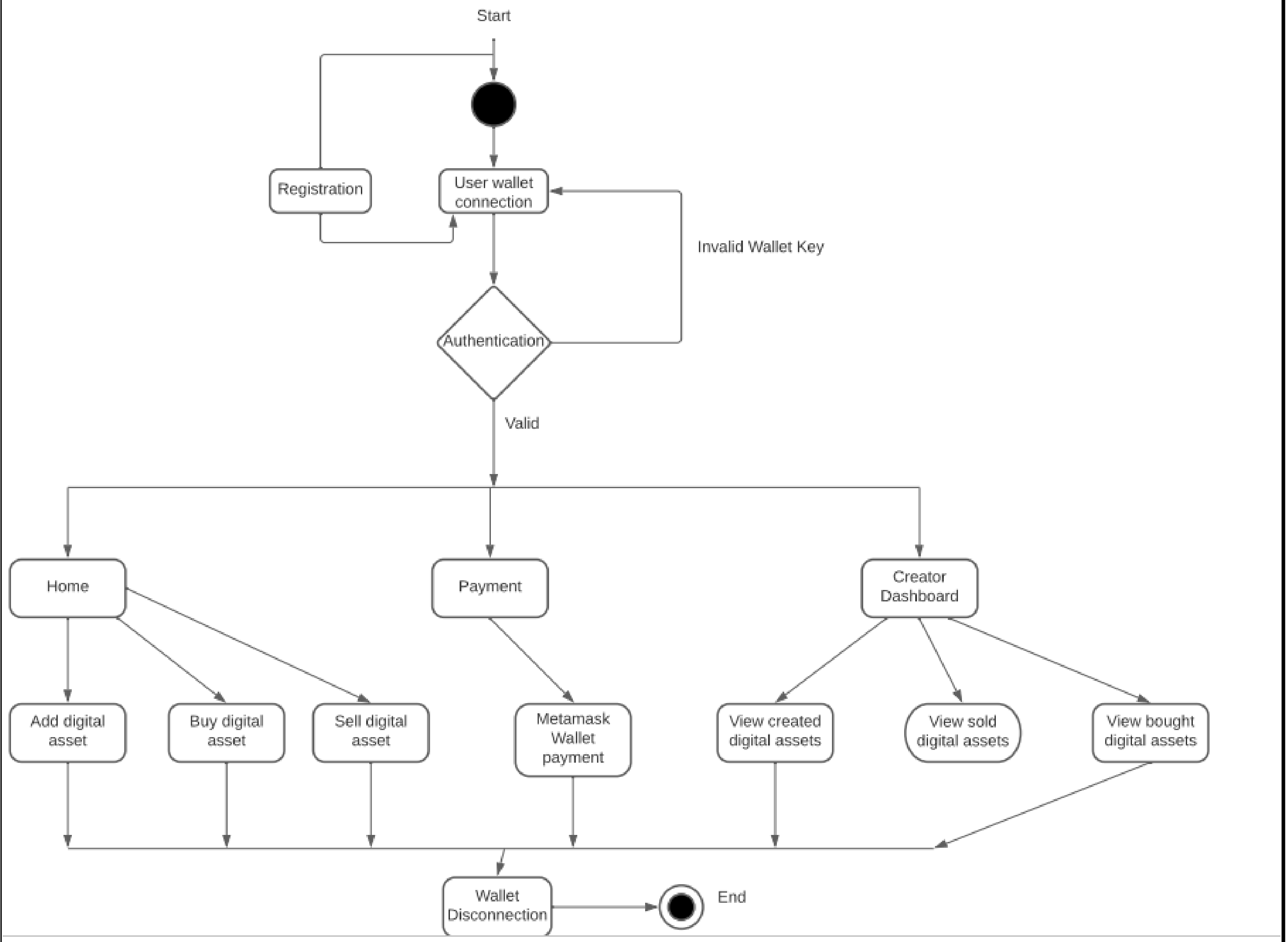
Class Diagram:



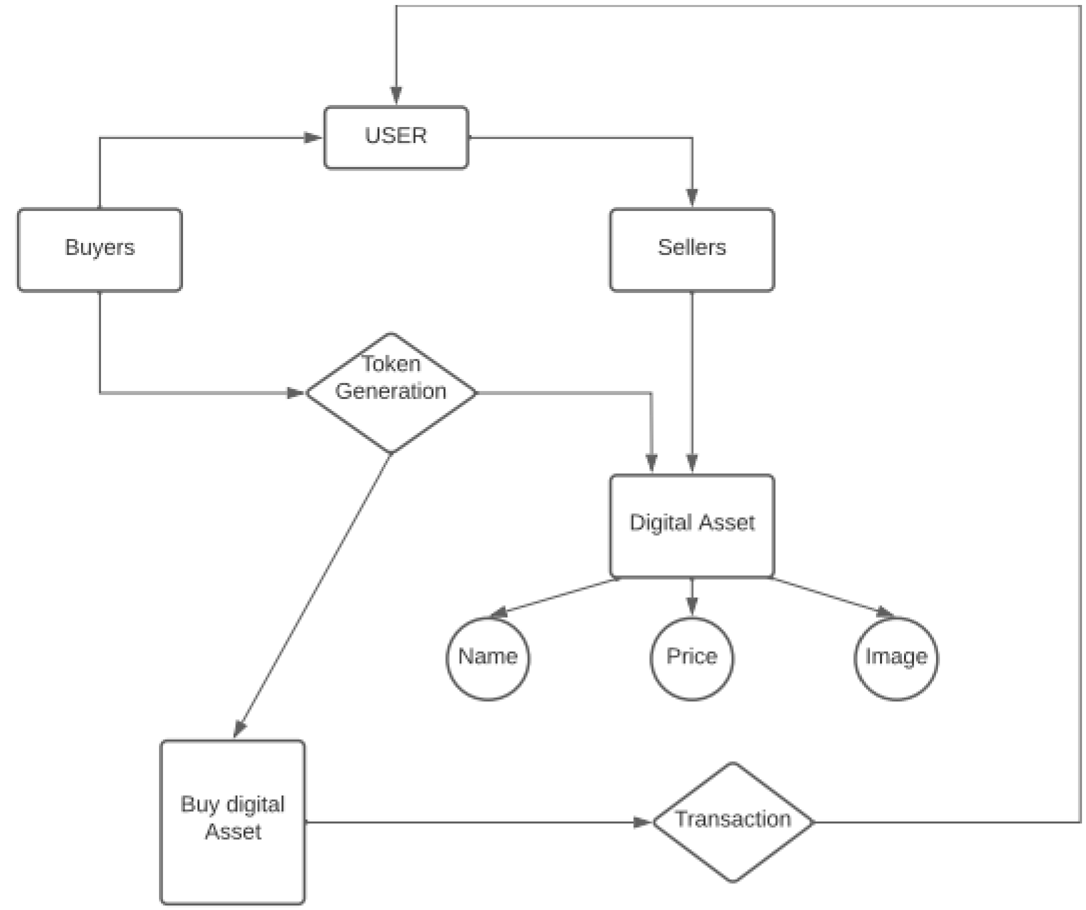
Sequence Diagram:



Activity Diagram:



E-R Diagram:


USER
Buyers
Sellers
Token Generation
Digital Asset
Name
Price
Image
Buy digital Asset
Transaction

Data Dictionary:

NFT								
Sr no.	Field Name	Data Type	width	Unique	Default	Primary Key/ Foreign Key	Referred Table	Description
1	itemId	Uint256	-	Yes	None	-	-	Auto_Increment
2	tokenId	Uint256	-	Yes	None	-	-	-
3	seller	Address payable	-	No	-	-	-	-
4	owner	Address payable	-	No	-	-	-	-
5	price	Uint256	-	No	-	-	-	-
6	sold	bool	-	No	false	-	-	-
7	name	String	-	No	-	-	-	-
8	photo	image	-	-	-	-	-	-

4. Implementation Details

The system consists of 3 major modules:

1. User Module
2. NFT Module
3. NFT market Module

Each module consists of major methods to implement the required functionality. The implementation is done using Mern Stack

User Module

This module is for the sign in purpose.

NFT Module

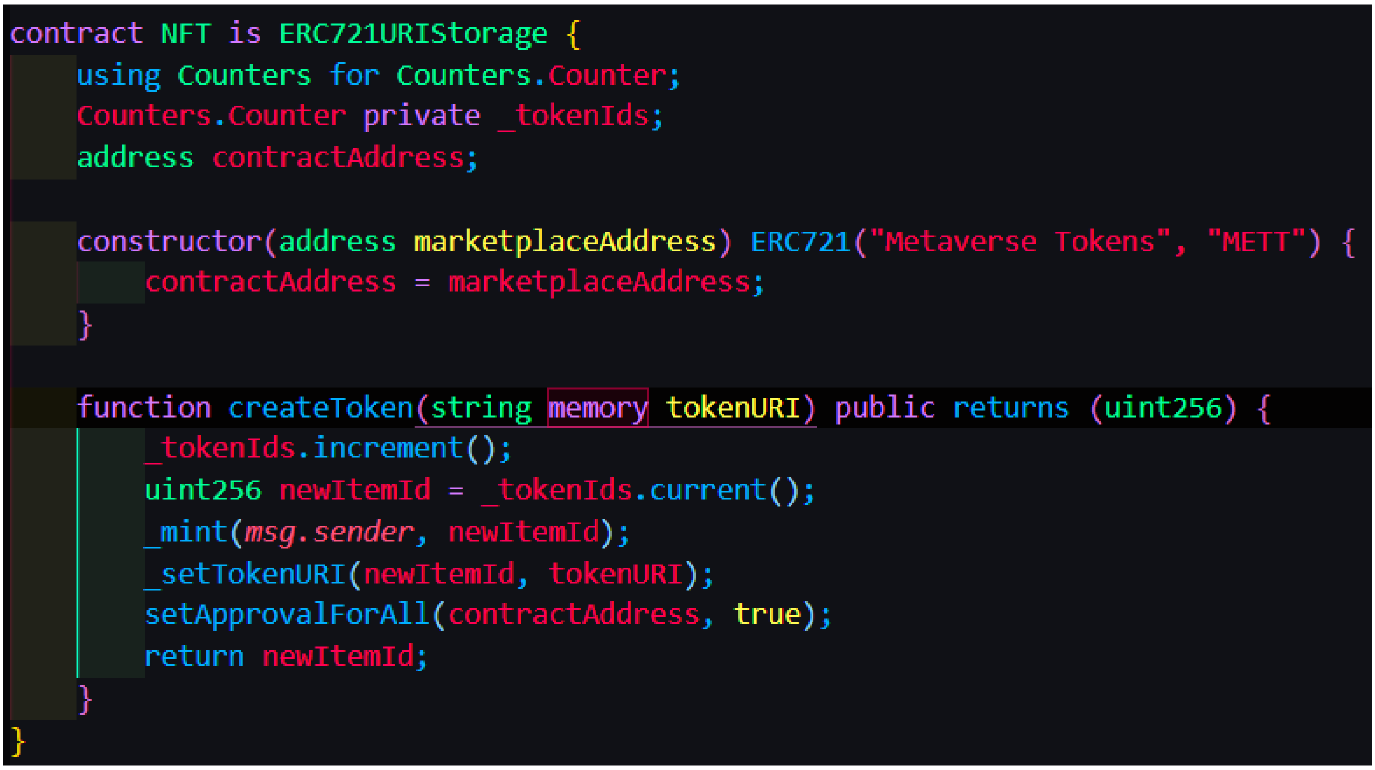
This module handles the token part where the tokens are created and keep track of token ids.

NFT Market Module

This module handles the part where the buyer can buy NFTs and sellers can sell NFTss.

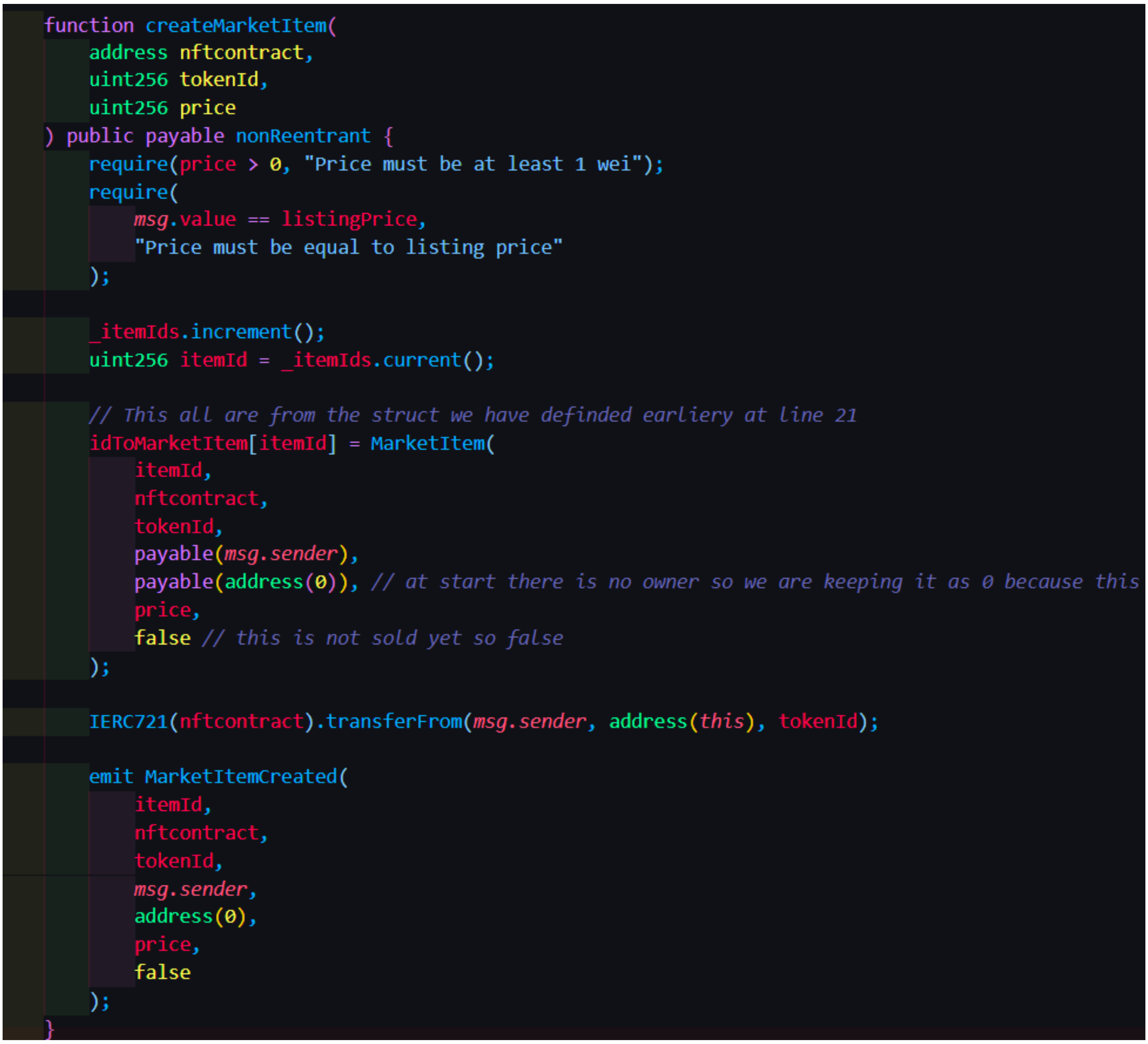
After the NFT is sold, it will be added to the buyer owned items .

Function Prototypes



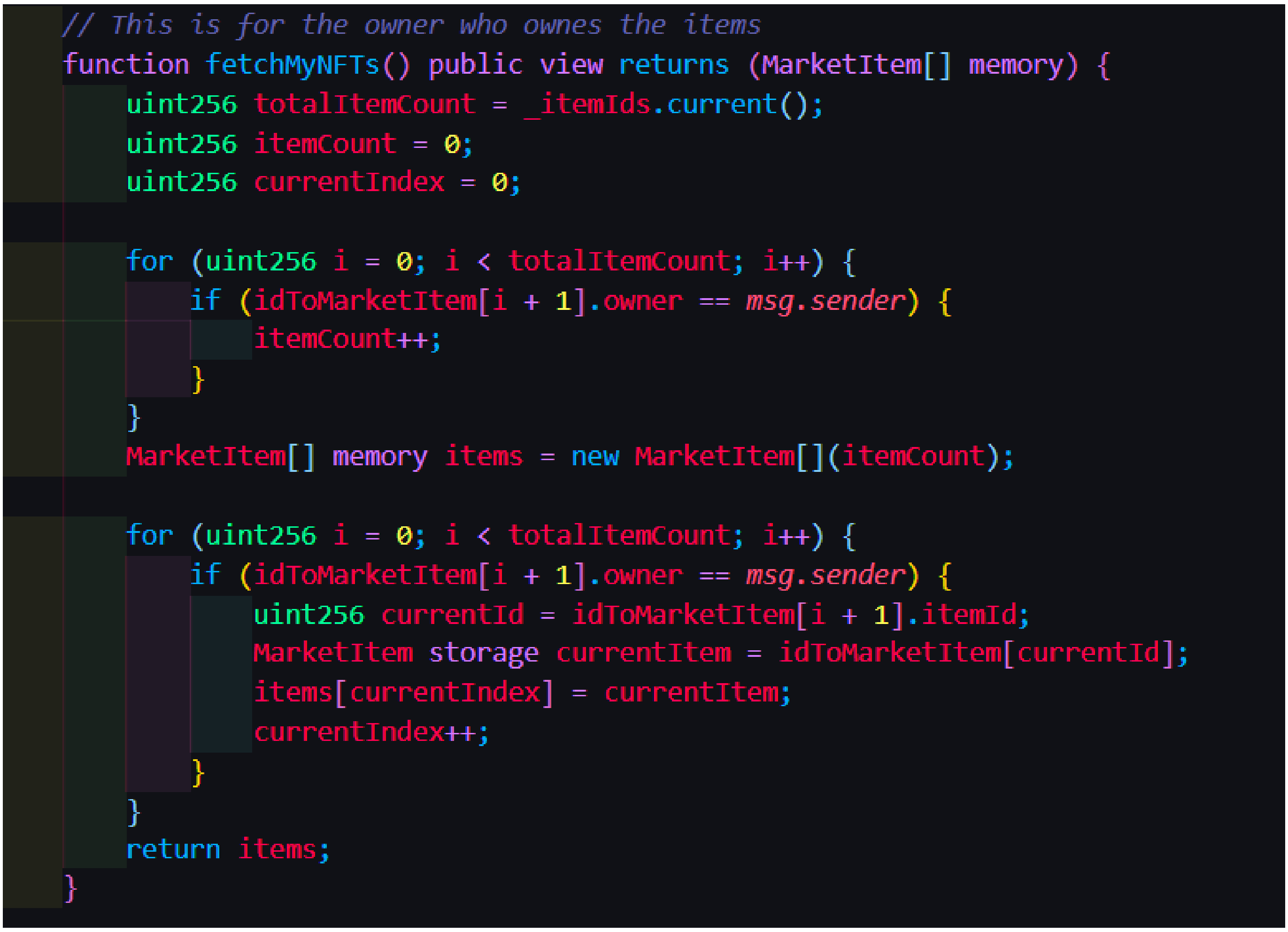
```
contract NFT is ERC721URIStorage {
    using Counters for Counters.Counter;
    Counters.Counter private _tokenIds;
    address contractAddress;

    constructor(address marketplaceAddress) ERC721("Metaverse Tokens", "METT") {
        contractAddress = marketplaceAddress;
    }

    function createToken(string memory tokenURI) public returns (uint256) {
        _tokenIds.increment();
        uint256 newItemId = _tokenIds.current();
        _mint(msg.sender, newItemId);
        _setTokenURI(newItemId, tokenURI);
        setApprovalForAll(contractAddress, true);
        return newItemId;
    }
}
```

Create Token



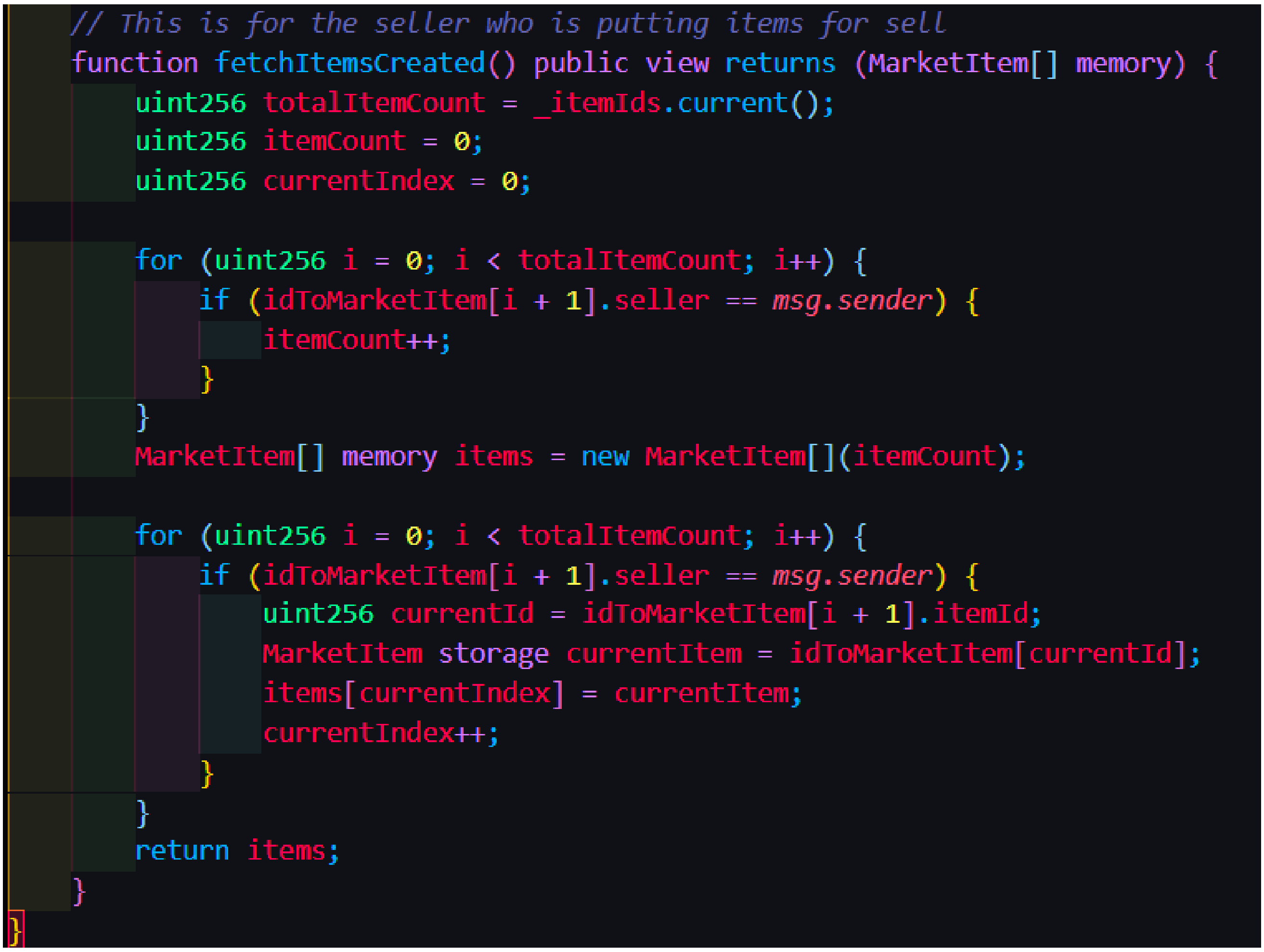
```
function createMarketItem(
    address nftcontract,
    uint256 tokenId,
    uint256 price
) public payable nonReentrant {
    require(price > 0, "Price must be at least 1 wei");
    require(
        msg.value == listingPrice,
        "Price must be equal to listing price"
    );

    _itemIds.increment();
    uint256 itemId = _itemIds.current();

    // This all are from the struct we have definded earliery at line 21
    idToMarketItem[itemId] = MarketItem(
        itemId,
        nftcontract,
        tokenId,
        payable(msg.sender),
        payable(address(0)), // at start there is no owner so we are keeping it as 0 because this
        price,
        false // this is not sold yet so false
    );

    IERC721(nftcontract).transferFrom(msg.sender, address(this), tokenId);

    emit MarketItemCreated(
        itemId,
        nftcontract,
        tokenId,
        msg.sender,
        address(0),
        price,
        false
    );
}
```

Create Market Item



```
// This is for the owner who ownes the items
function fetchMyNFTs() public view returns (MarketItem[] memory) {
    uint256 totalItemCount = _itemIds.current();
    uint256 itemCount = 0;
    uint256 currentIndex = 0;

    for (uint256 i = 0; i < totalItemCount; i++) {
        if (idToMarketItem[i + 1].owner == msg.sender) {
            itemCount++;
        }
    }
    MarketItem[] memory items = new MarketItem[](itemCount);

    for (uint256 i = 0; i < totalItemCount; i++) {
        if (idToMarketItem[i + 1].owner == msg.sender) {
            uint256 currentId = idToMarketItem[i + 1].itemId;
            MarketItem storage currentItem = idToMarketItem[currentId];
            items[currentIndex] = currentItem;
            currentIndex++;
        }
    }
    return items;
}
```

Owned NFTs



```
    // This is for the seller who is putting items for sell
    function fetchItemsCreated() public view returns (MarketItem[] memory) {
        uint256 totalItemCount = _itemIds.current();
        uint256 itemCount = 0;
        uint256 currentIndex = 0;

        for (uint256 i = 0; i < totalItemCount; i++) {
            if (idToMarketItem[i + 1].seller == msg.sender) {
                itemCount++;
            }
        }
        MarketItem[] memory items = new MarketItem[](itemCount);

        for (uint256 i = 0; i < totalItemCount; i++) {
            if (idToMarketItem[i + 1].seller == msg.sender) {
                uint256 currentId = idToMarketItem[i + 1].itemId;
                MarketItem storage currentItem = idToMarketItem[currentId];
                items[currentIndex] = currentItem;
                currentIndex++;
            }
        }
        return items;
    }
}
```

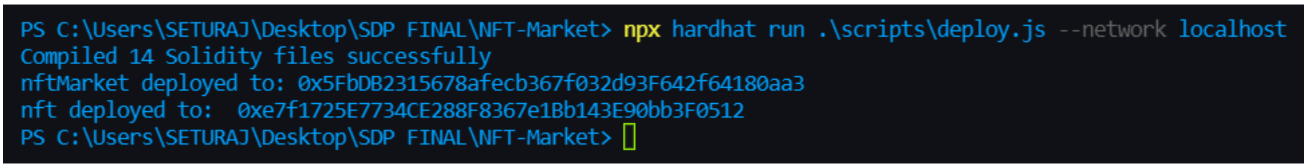
Items on sell

5. Testing

Hardhat testing was performed in order to find and fix the bugs in development process.

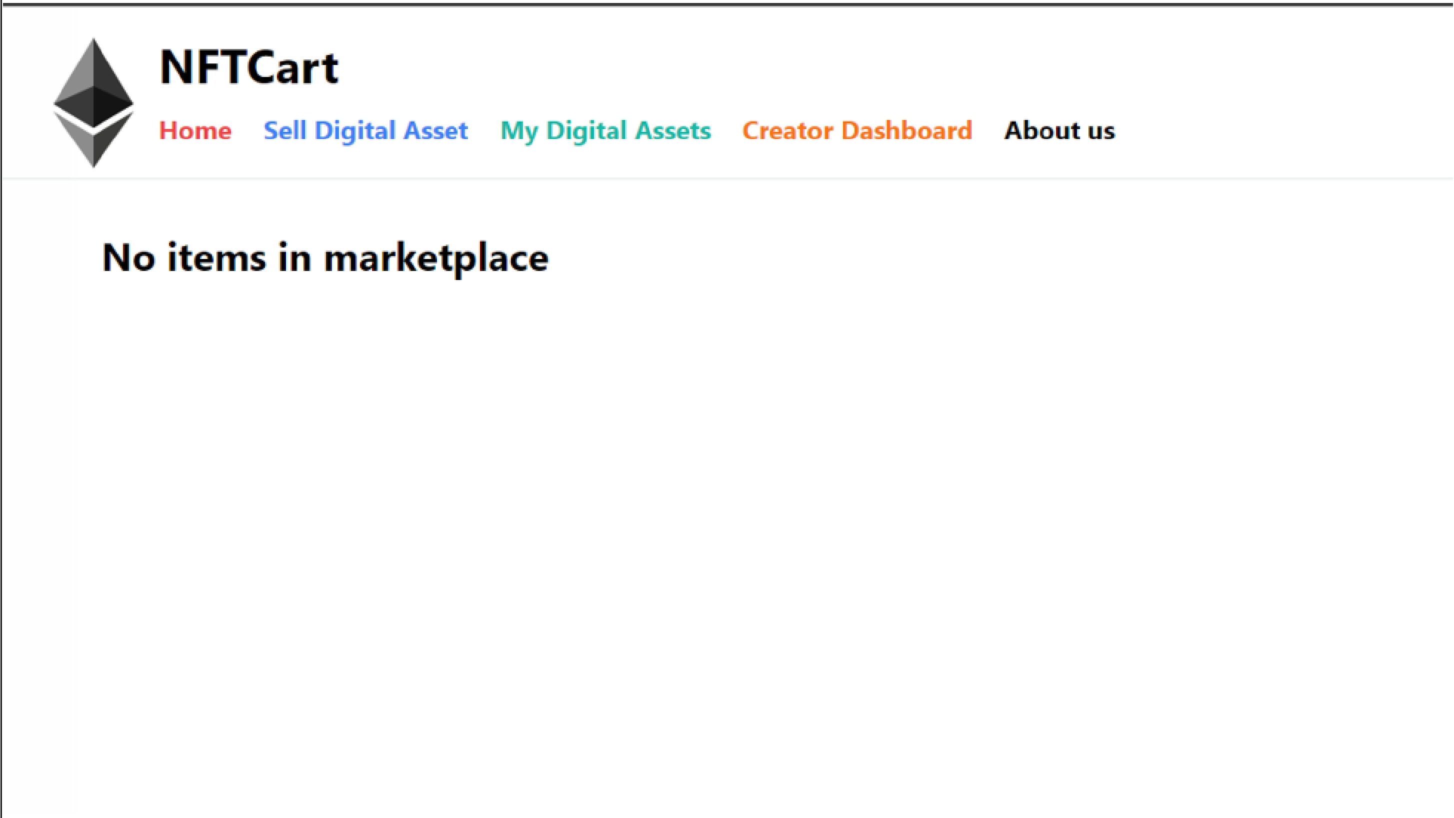
Testing Method: Hardhat Testing

- The hardhat testing was done by writing a simple contract, a test for that contract , a sample script that deploys the contract, and an example of a task implementation.
- Run npx hardhat node for listing the available accounts
- Then test the deploy js script using hardhat
    - ➢ The nft and nftmarket will be deployed.
    - ➢ The item is item which have a price, tokenId, seller, owner and tokenUri.



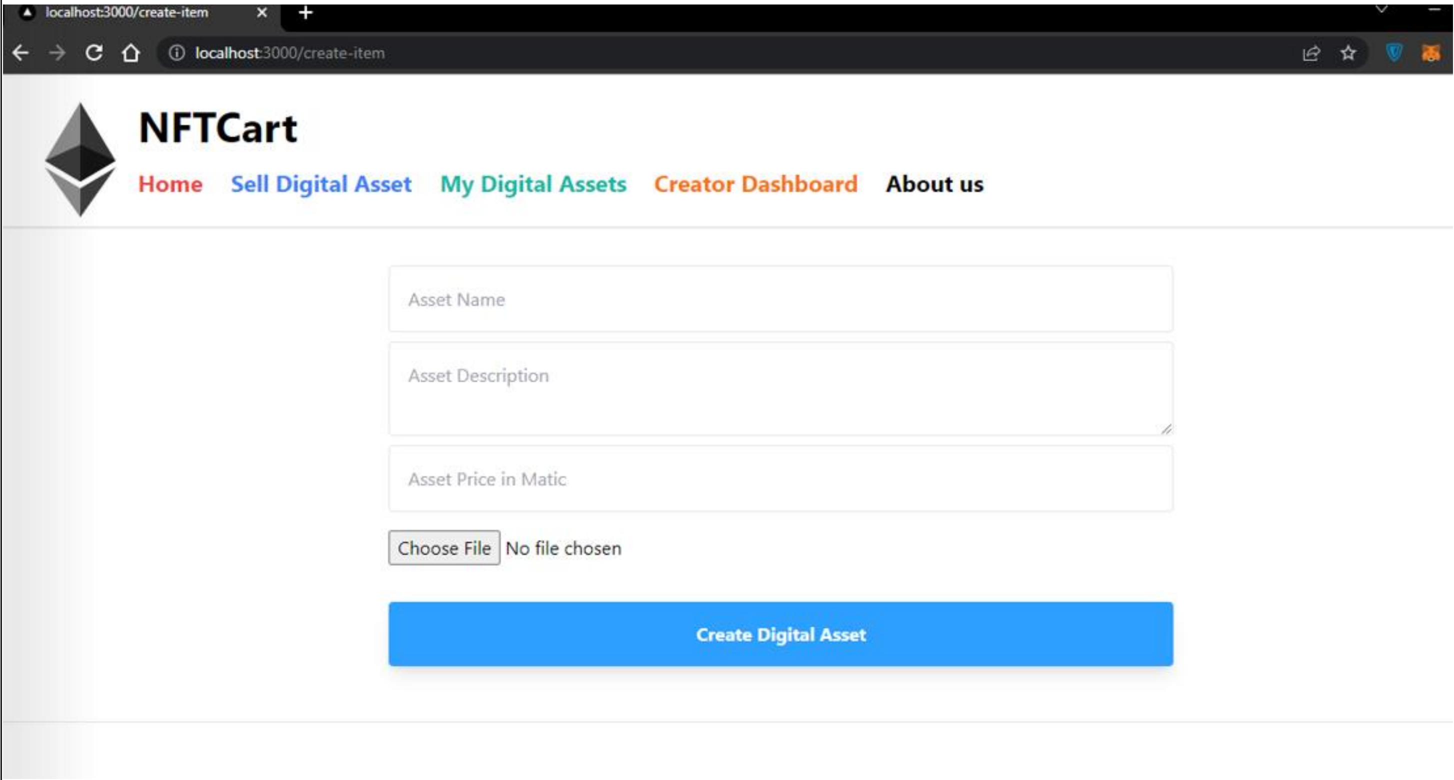
```
PS C:\Users\SETURAJ\Desktop\SDP FINAL\NFT-Market> npx hardhat run .\scripts\deploy.js --network localhost
Compiled 14 Solidity files successfully
nftMarket deployed to: 0x5FbDB2315678afecb367f032d93F642f64180aa3
nft deployed to:  0xe7f1725E7734CE288F8367e1Bb143E90bb3F0512
PS C:\Users\SETURAJ\Desktop\SDP FINAL\NFT-Market> 
```

6. Screenshots

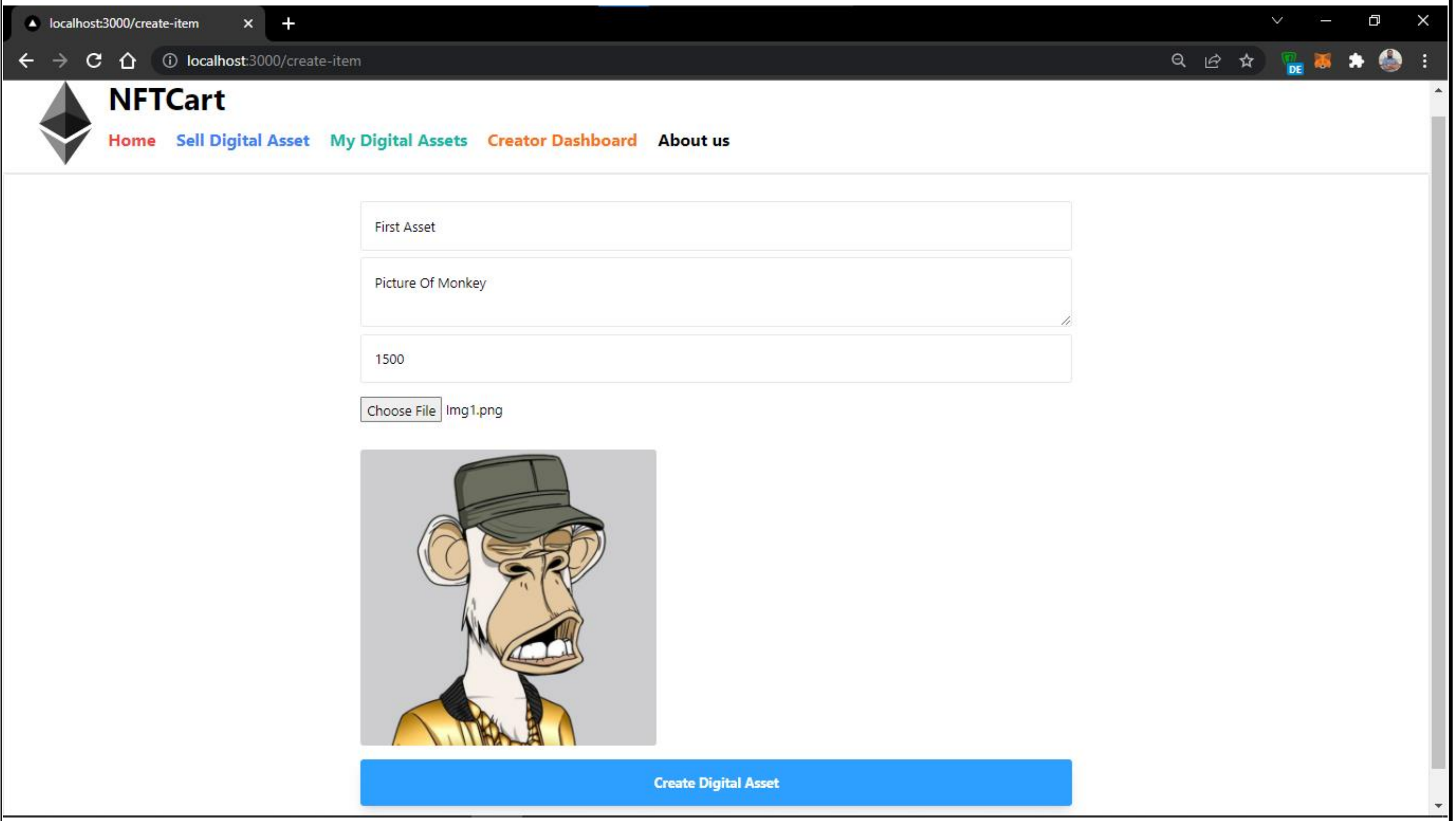


No items in marketplace

Initial Welcome Page



Create Asset page



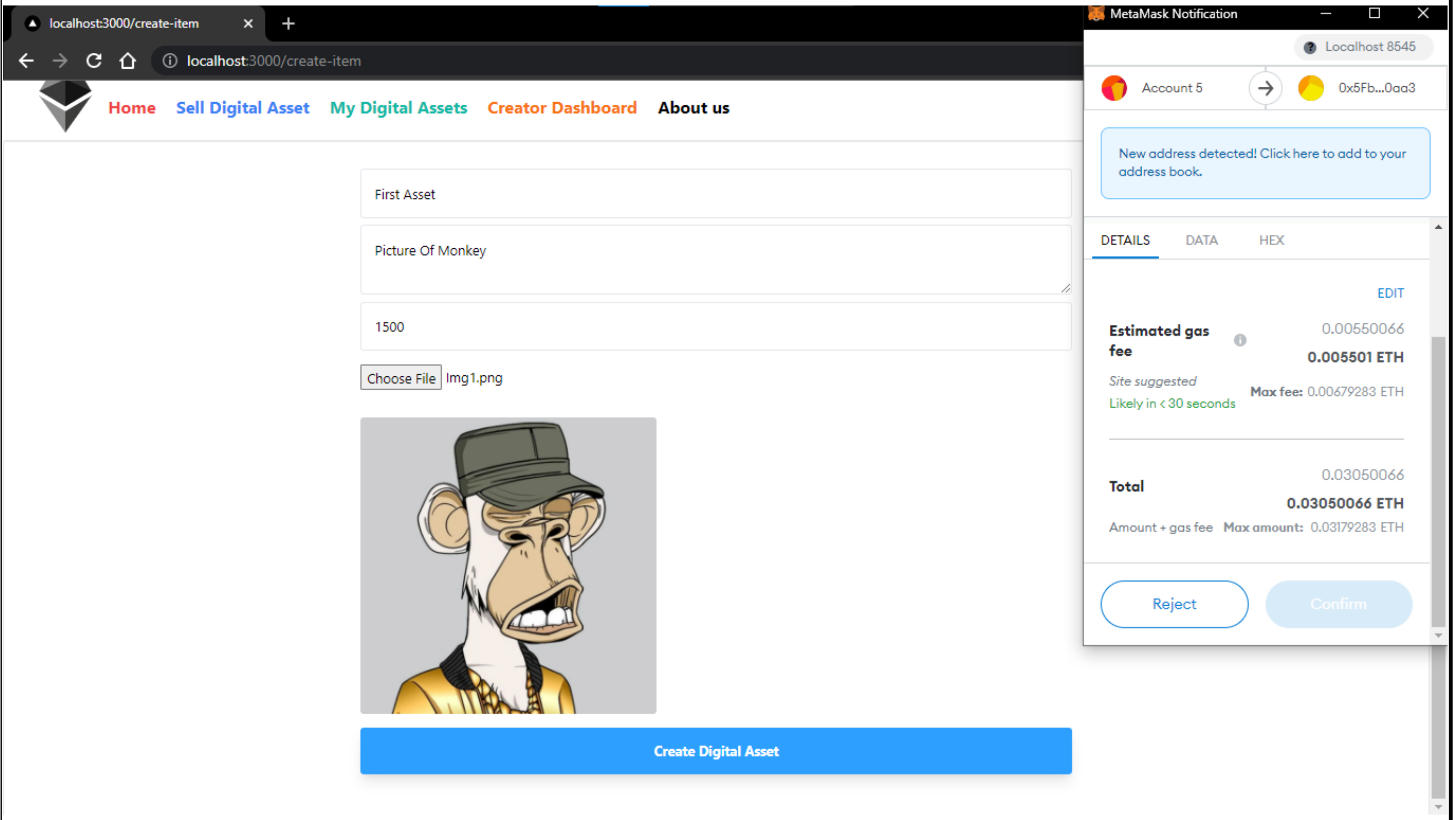
First Asset

Picture Of Monkey

1500

Img1.png

Digital Asset upload



First Asset

Picture Of Monkey

1500

Img1.png

Account 5 → 0x5Fb...0aa3

New address detected! Click here to add to your address book.

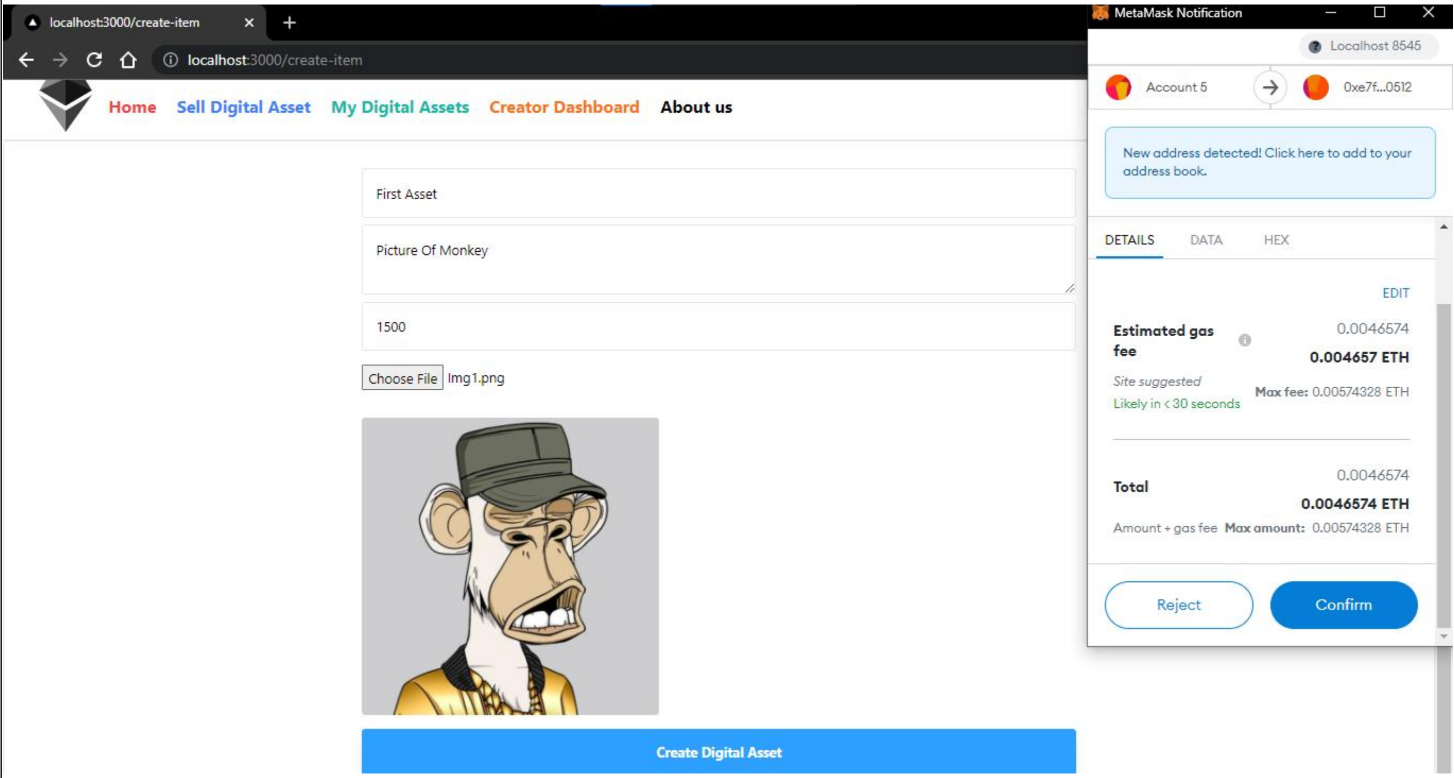
Estimated gas fee 0.00550066 0.005501 ETH

Site suggested
Likely in < 30 seconds
Max fee: 0.00679283 ETH

Total 0.03050066 0.03050066 ETH

Amount + gas fee Max amount: 0.03179283 ETH

Transaction page



First Asset

Picture Of Monkey

1500

Img1.png

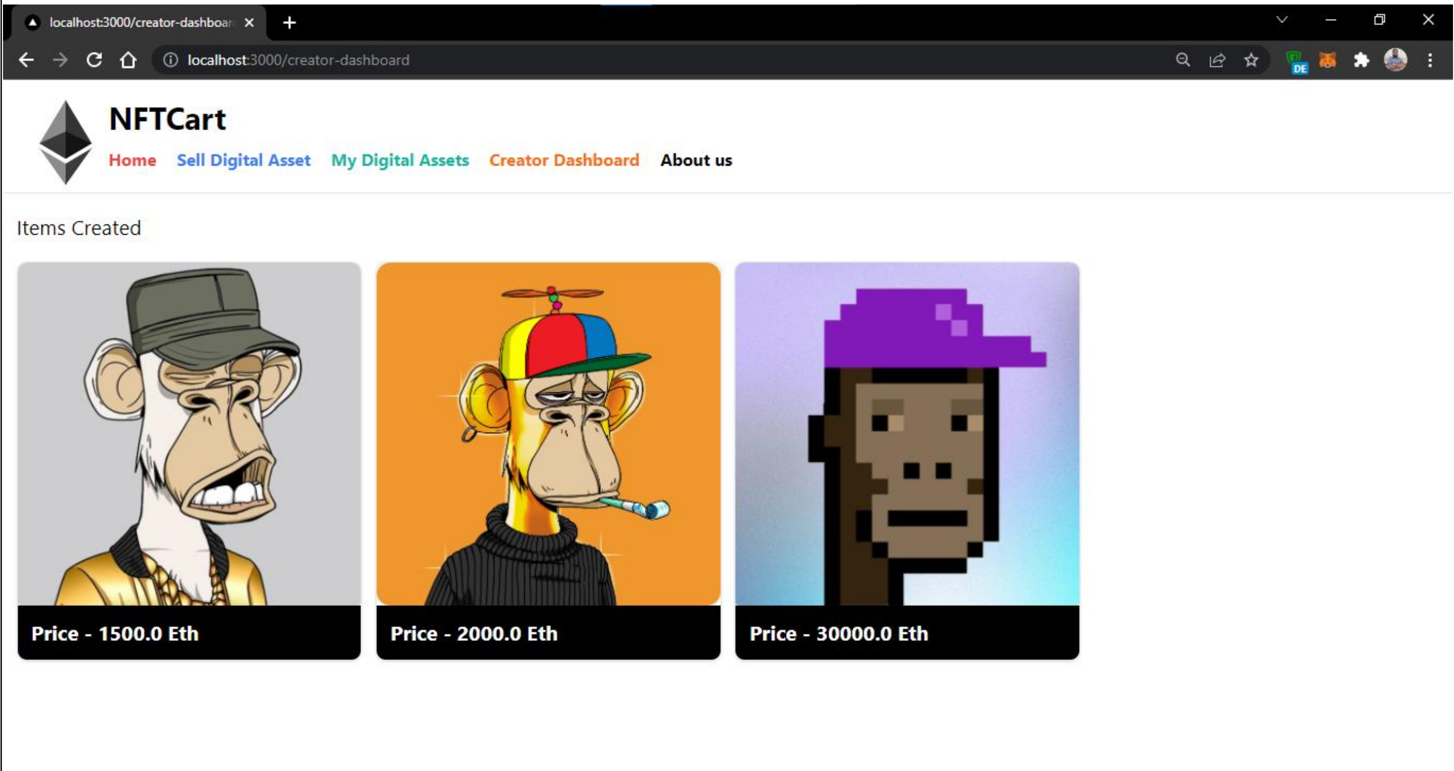
Account 5 → 0xe7f...0512

New address detected! Click here to add to your address book.

Estimated gas fee 0.0046574
0.004657 ETH
Site suggested
Likely in < 30 seconds
Max fee: 0.00574328 ETH

Total 0.0046574
0.0046574 ETH
Amount + gas fee Max amount: 0.00574328 ETH

Transaction Done



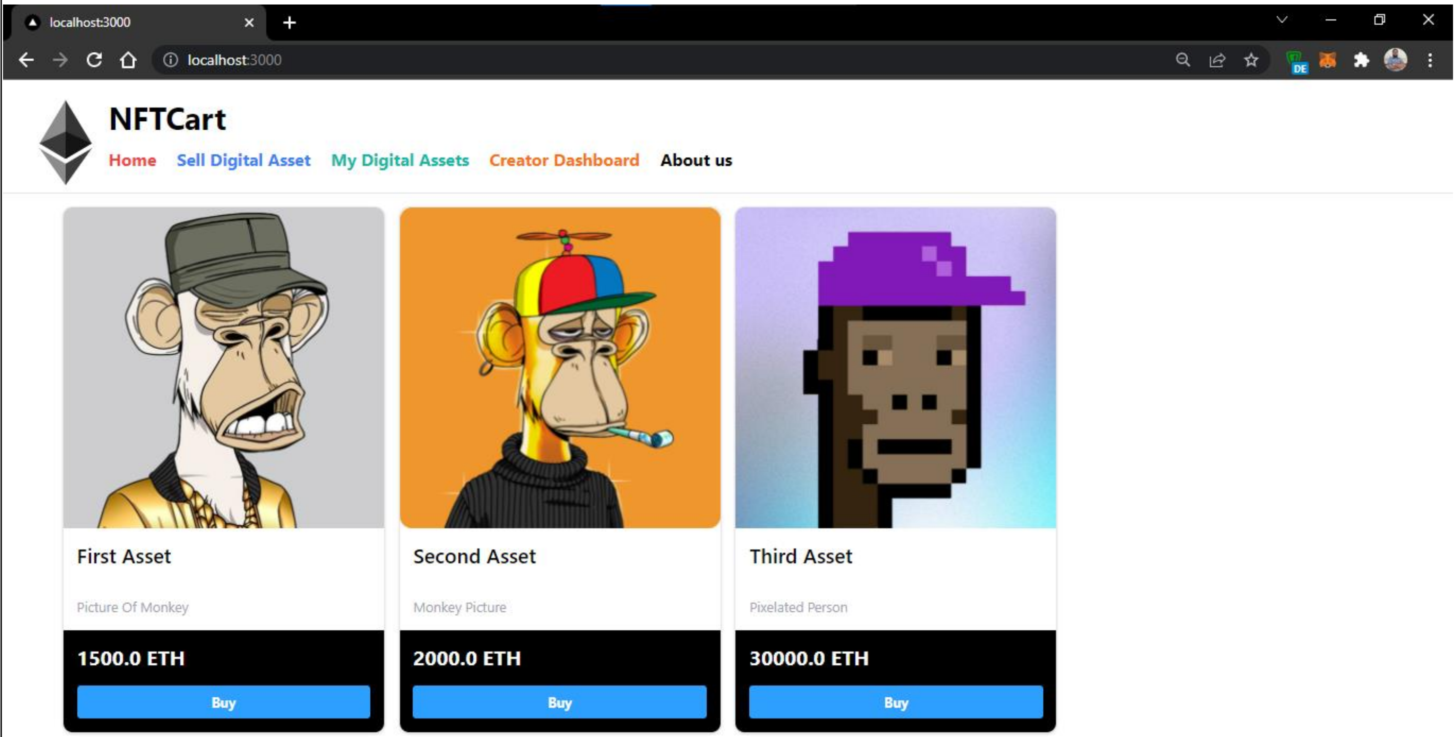
Items Created

Price - 1500.0 Eth

Price - 2000.0 Eth

Price - 30000.0 Eth

Created Items



First Asset

Picture Of Monkey

1500.0 ETH

Second Asset

Monkey Picture

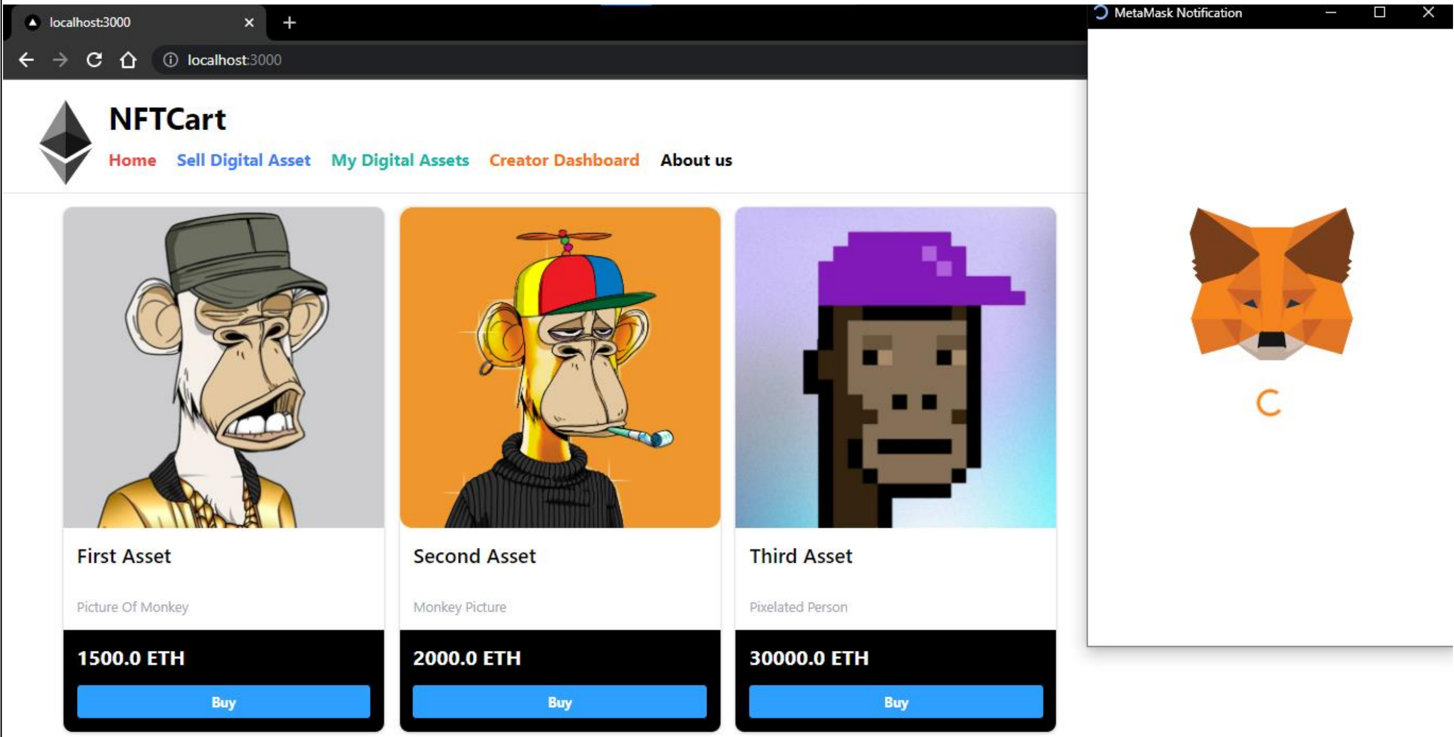
2000.0 ETH

Third Asset

Pixelated Person

30000.0 ETH

Home page after items added



First Asset

Picture Of Monkey

1500.0 ETH

Second Asset

Monkey Picture

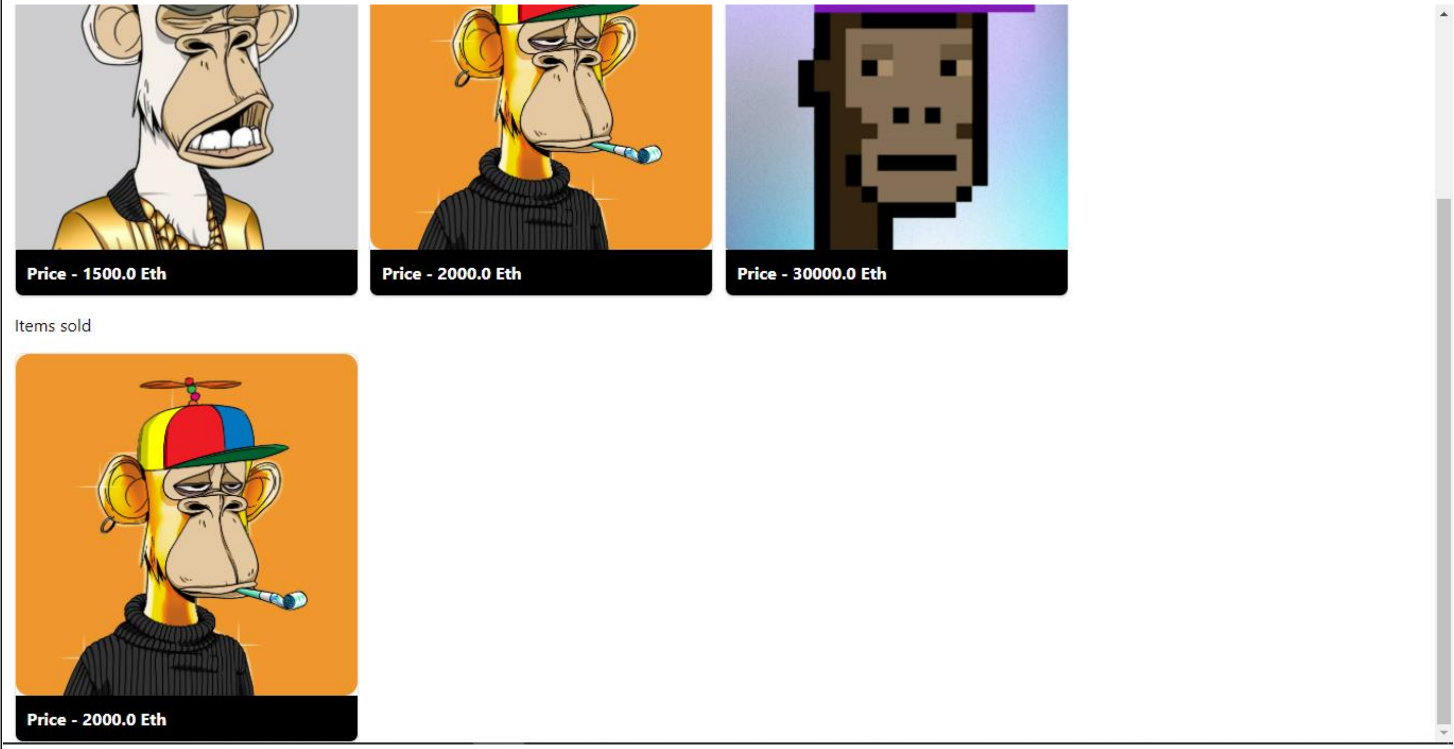
2000.0 ETH

Third Asset

Pixelated Person

30000.0 ETH

Selling Asset



Price - 1500.0 Eth

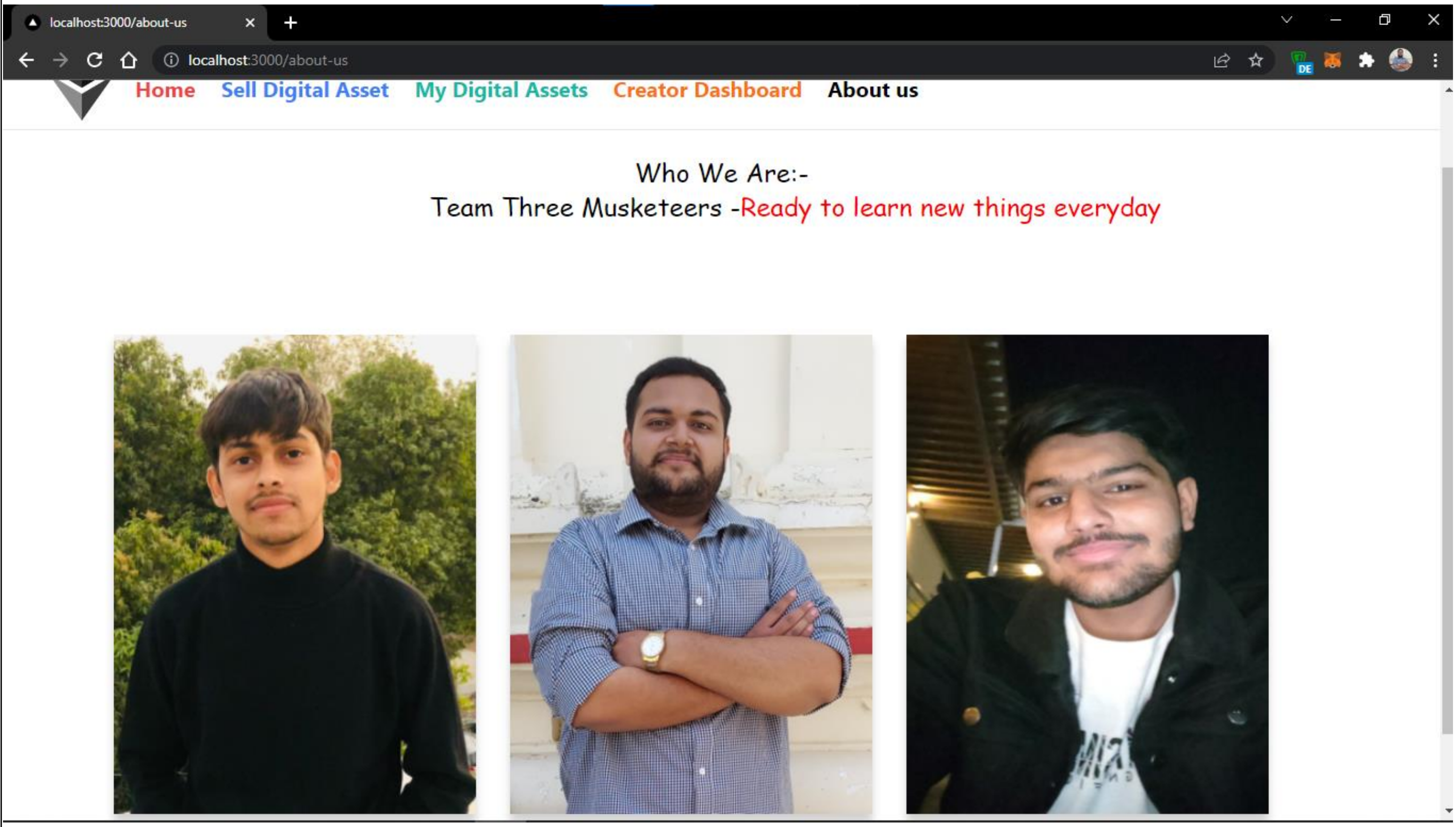
Price - 2000.0 Eth

Price - 30000.0 Eth

Items sold

Price - 2000.0 Eth

Dashboard after item sold



Who We Are:-
Team Three Musketeers -Ready to learn new things everyday

About Us

7. Conclusion

The functionalities are implemented in system after understanding all the system modules according to the requirements. Functionalities that are successfully implemented in the system are:

- Wallet Login/Sign-in
- Create NFT
- Add NFT
- Buy NFT
- Sell NFT
- Wallet Payment
- User Dashboard

After the implementation and coding of system, comprehensive testing was performed on the system to determine the errors and possible flaws in the system.

8. Limitations and Future Enhancements

We are able to implement some of the functionality of all modules. We aim to complete all the functionality of all modules and make this product ready to be used practically in all scenarios. Currently, the project runs completely fine if all the inputs / selections are given within proper criteria but it doesn’t cover all the corner cases.

We have not created bidding system in our current website. The user can also login using MetaMask wallet and not other wallets like coinbase wallet, walletconnect, fortmatic.

We will extend this further more to create those functionality and try to make the project which runs anywhere (Not just in localhost).

9. Reference / Bibliography

Following links and websites were referred during the development of this project:

- getbootstrap.com
- stackoverflow.com
- https://nextjs.org/
- https://ethereum.org/en/developers/github.com
- https://opensea.io/
- https://docs.soliditylang.org/en/v0.8.12/

Project Git Repository Link:

https://github.com/SETURAJ/NFT-MARKET-PLACE