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 assets like property, shares in a company etc.

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. The creator dashboard is provided for the buyers and sellers to get information about the digital assets they own or traded.

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

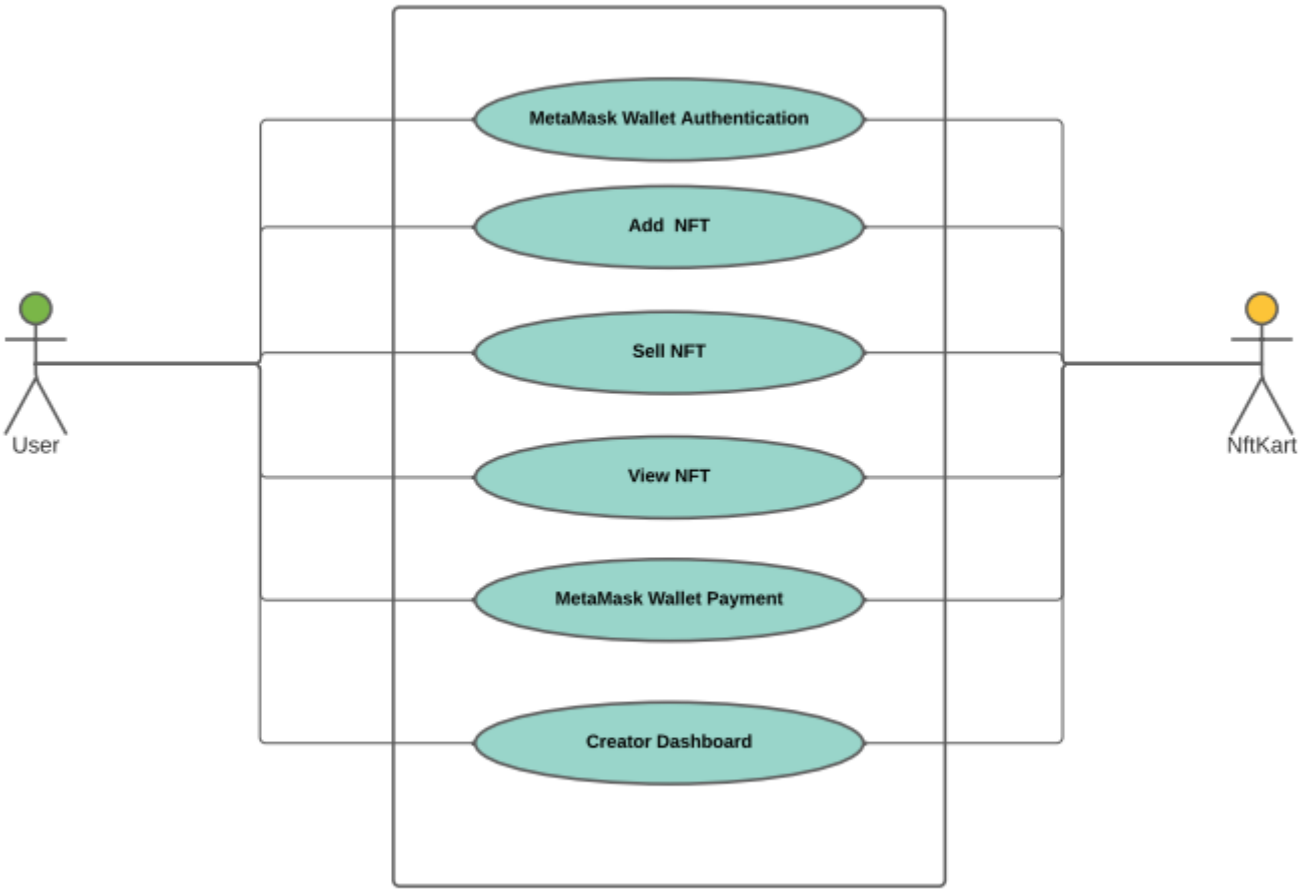
Buy NFT:

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

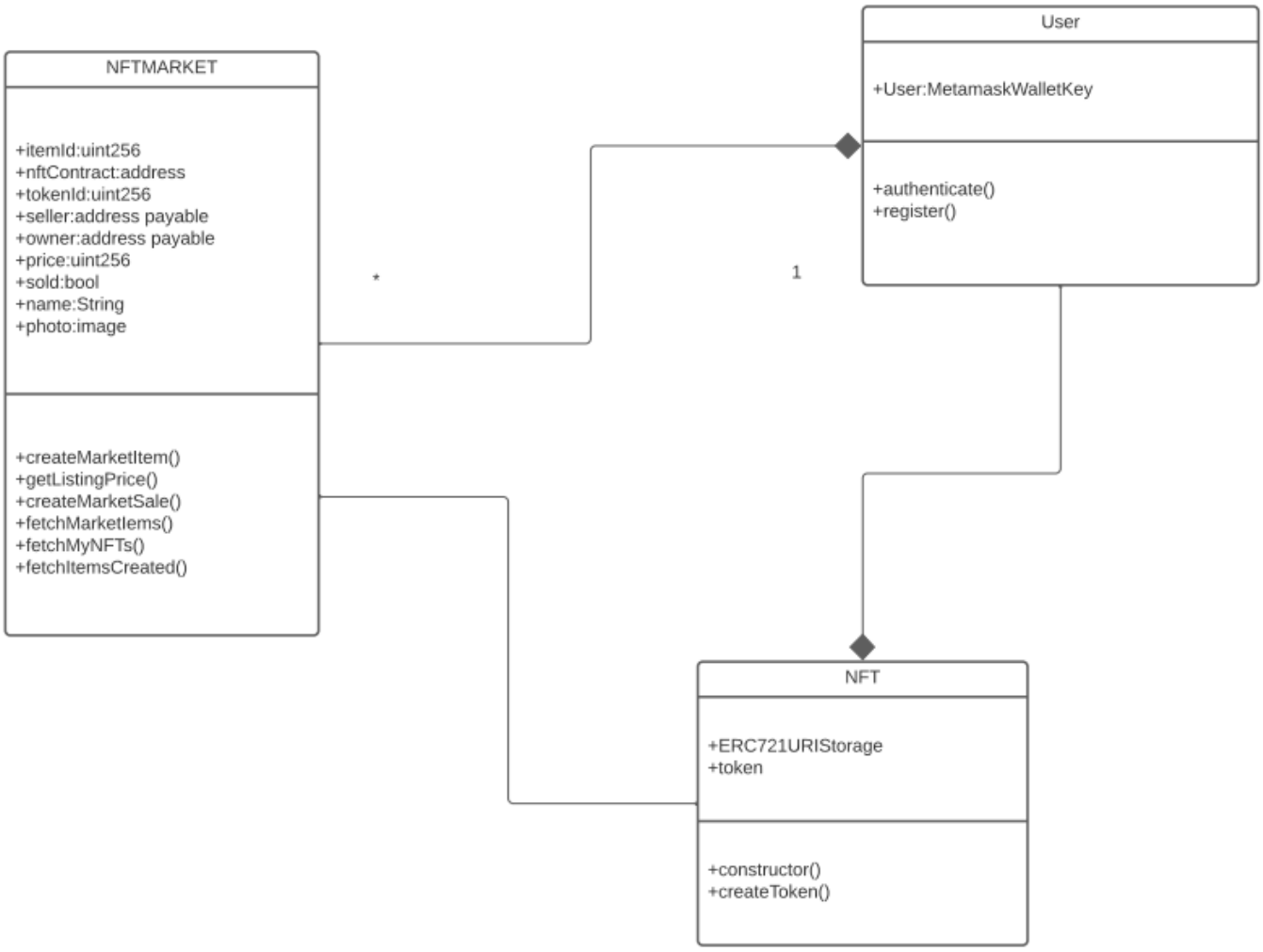
O/p: NFT will be added to the owned items and Ethereum will be deducted from your wallet

3. Design Documents:

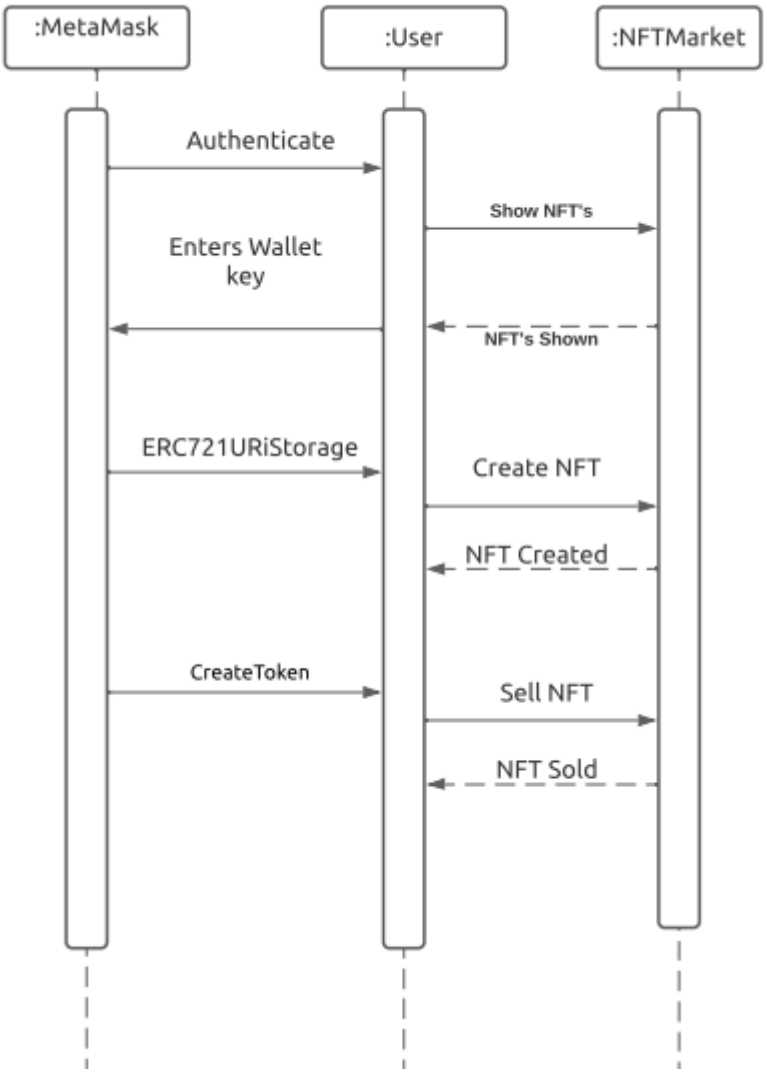
Use Case Diagram:



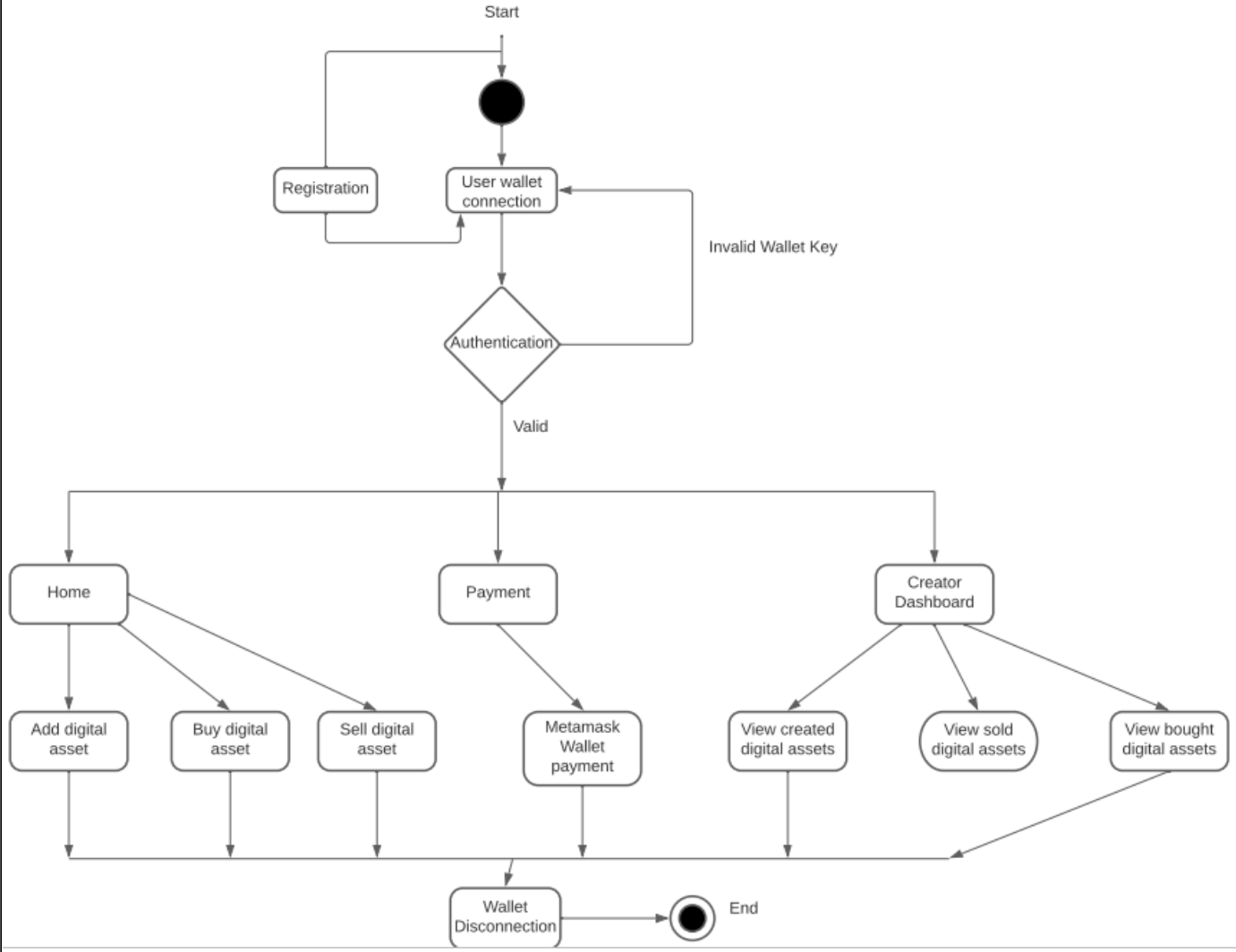
Class Diagram:



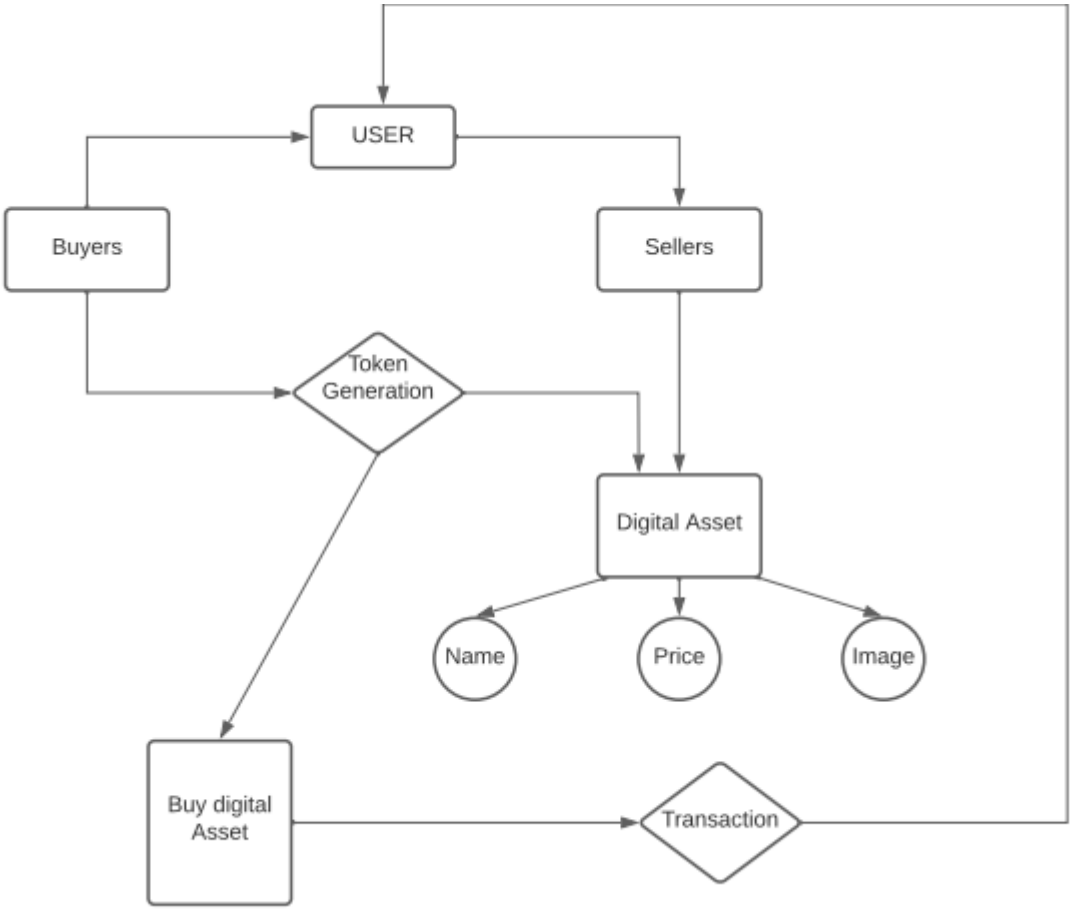
Sequence Diagram:



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 NFTs.

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 local accounts

```
PS D:\Sem 6\SDP\NFT MarketPlace\nft-marketplace> npx hardhat node
Started HTTP and WebSocket JSON-RPC server at http://127.0.0.1:8545/

Accounts
========

WARNING: These accounts, and their private keys, are publicly known.
Any funds sent to them on Mainnet or any other live network WILL BE LOST
.

Account #0: 0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266 (10000 ETH)
Private Key: 0xac0974bec39a17e36ba4a6b4d238ff944bacb478cbed5efcae784d7bf
4f2ff80

Account #1: 0x70997970c51812dc3a010c7d01b50e0d17dc79c8 (10000 ETH)
Private Key: 0x59c6995e998f97a5a0044966f0945389dc9e86dae88c7a8412f4603b6
b78690d

Account #2: 0x3c44cdddb6a900fa2b585dd299e03d12fa4293bc (10000 ETH)
Private Key: 0x5de4111afa1a4b94908f83103eb1f1706367c2e68ca870fc3fb9a804c
dab365a

Account #3: 0x90f79bf6eb2c4f870365e785982e1f101e93b906 (10000 ETH)
Private Key: 0x7c852118294e51e653712a81e05800f419141751be58f605c371e1514
1b007a6

Account #4: 0x15d34aaf54267db7d7c367839aaf71a00a2c6a65 (10000 ETH)
Private Key: 0x47e179ec197488593b187f80a00eb0da91f1b9d0b13f8733639f19c30
a34926a

Account #5: 0x9965507d1a55bcc2695c58ba16fb37d819b0a4dc (10000 ETH)
Private Key: 0x8b3a350cf5c34c9194ca85829a2df0ec3153be0318b5e2d3348e87209
```

- Then test the deploy.js script using hardhat

```
PS D:\Sem 6\SDP\NFT MarketPlace\nft-marketplace> npx hardhat run .\scrip
ts\deploy.js --network localhost
nftMarket deployed to: 0x5FbDB2315678afecb367f032d93F642f64180aa3
nft deployed to:  0xe7f1725E7734CE288F8367e1Bb143E90bb3F0512
```

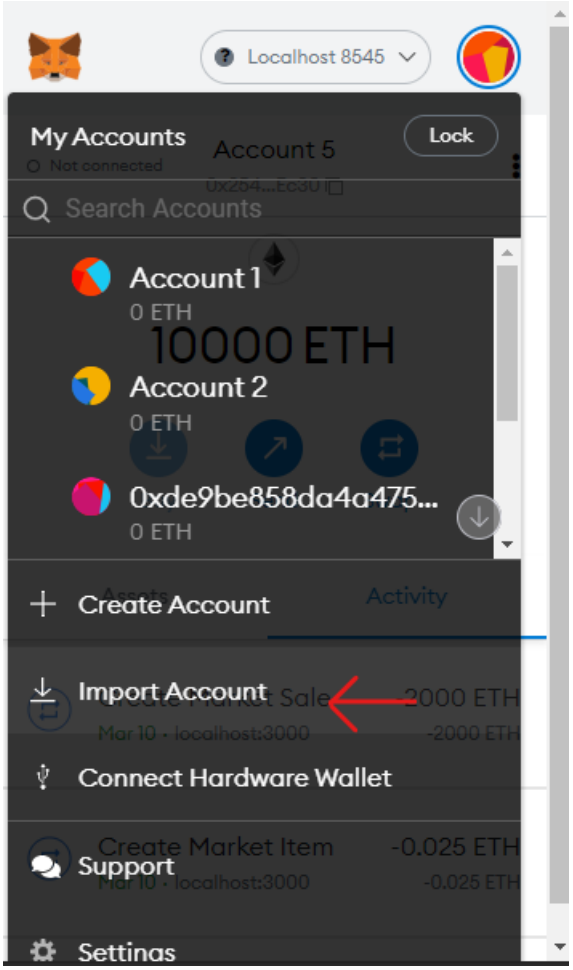
Information about these two addresses…

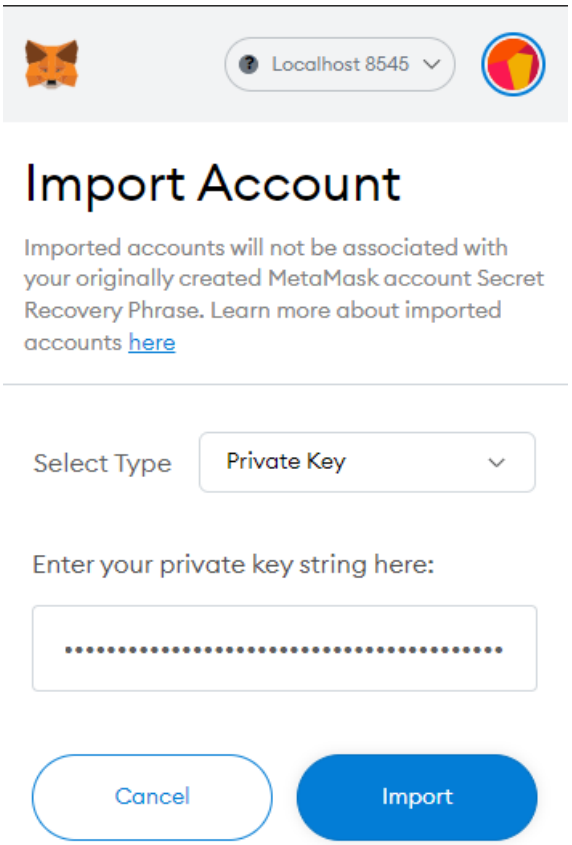
```
eth_chainId
eth_accounts
eth_blockNumber
eth_chainId (2)
eth_estimateGas
eth_getBlockByNumber
eth_feeHistory
eth_sendTransaction
  Contract deployment: NFTMarket
  Contract address:    0x5fbdb2315678afecb367f032d93f642f64180aa3
  Transaction:         0x46b2542c51324800c15fb9b96bcf738975fb636a23ef13b
710b71c01c1af2236
  From:                0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266
  Value:               0 ETH
  Gas used:            1522086 of 1522086
  Block #1:            0x48a3bca14d111e6292db2bb95ed0b470726db45faba5fe1
056b7791d1bb75e9f

eth_chainId
eth_getTransactionByHash
eth_chainId
eth_getTransactionReceipt
eth_accounts
eth_chainId
eth_estimateGas
eth_feeHistory
eth_sendTransaction
  Contract deployment: NFT
  Contract address:    0xe7f1725e7734ce288f8367e1bb143e90bb3f0512
  Transaction:         0xc73ed12f0962df610c4e1985fd4269055a395590eb438d2
c70561c3ebfaa1e2e
  From:                0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266
  Value:               0 ETH
  Gas used:            2523687 of 2523687
  Block #2:            0x128259d397aeadbf482fa28cef70bc11658758a60dc0962
```

- The nft and nftmarket will be deployed.
- The item is item which have a price, tokenId, seller, owner and tokenUri.

- At last, we will run the project using npm run dev

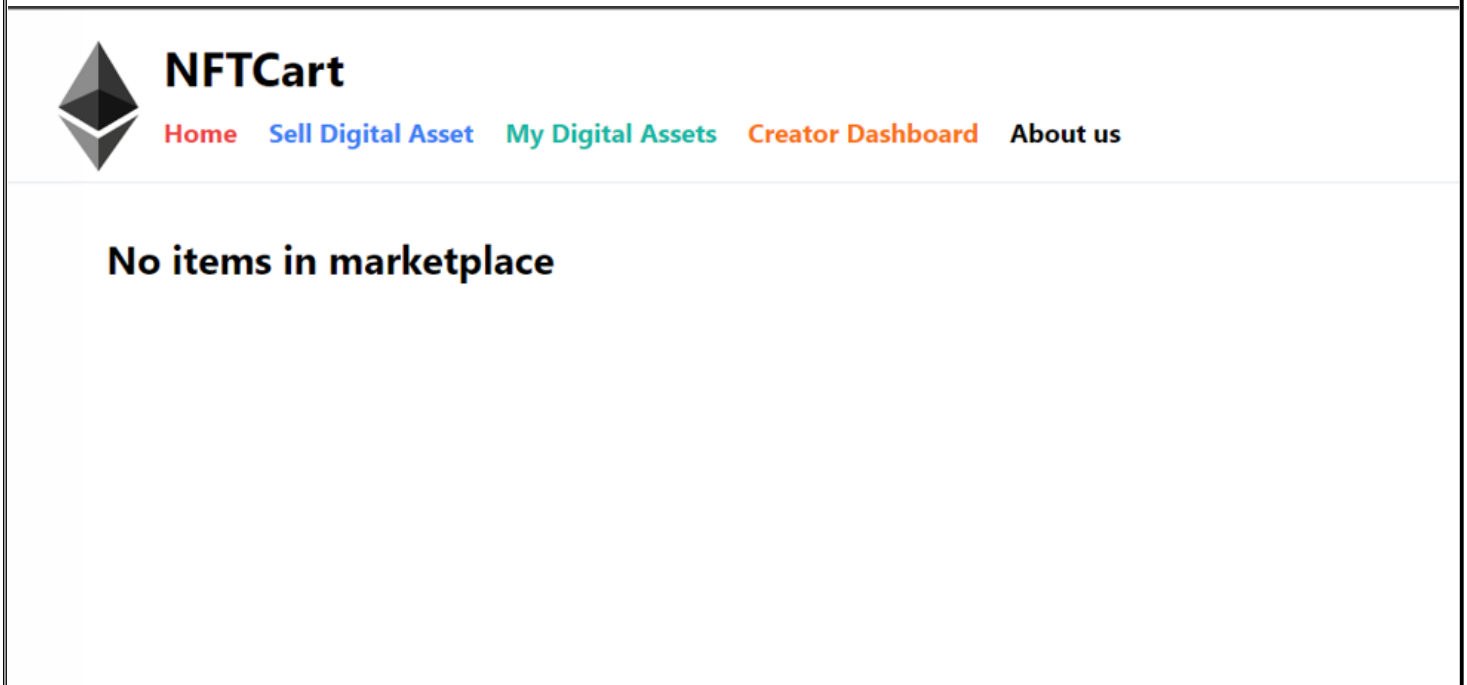
command. After project start running, we have to open MetaMask in browser and we have to import testing account from all above given accounts.



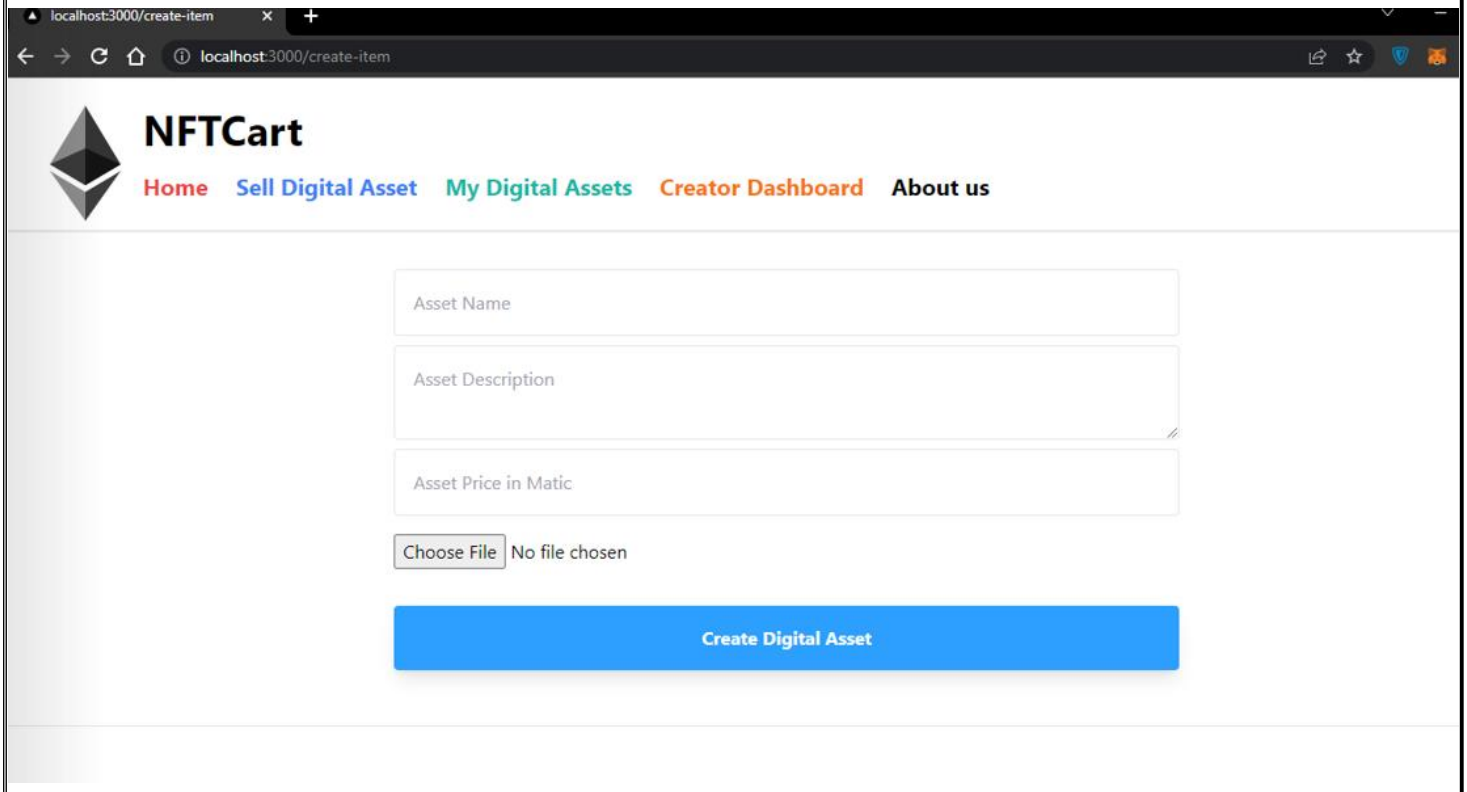


- Take any private key from above accounts and paste it here and you are good to go for testing Eths.

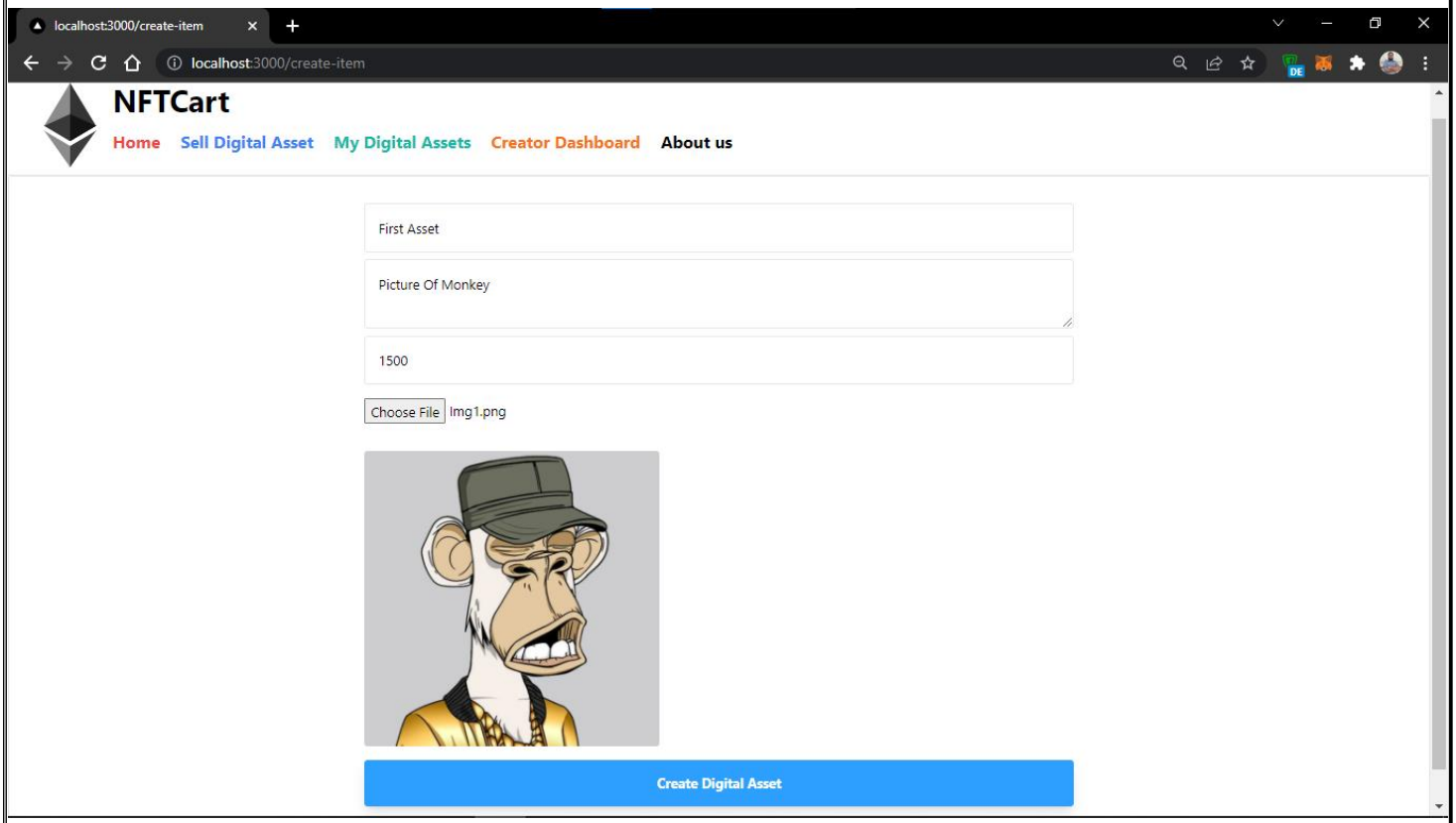
6. Screenshots



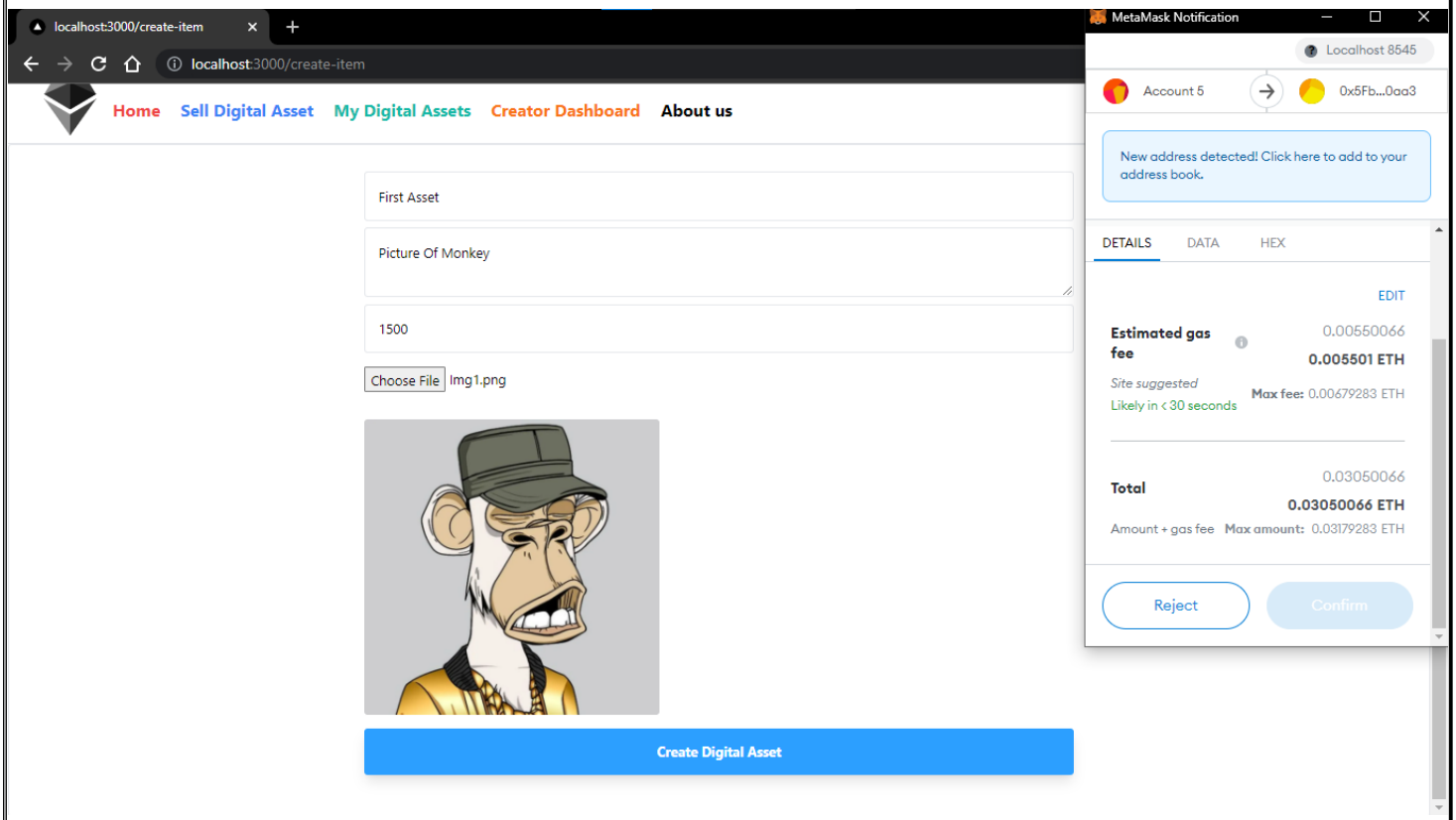
Initial Welcome Page



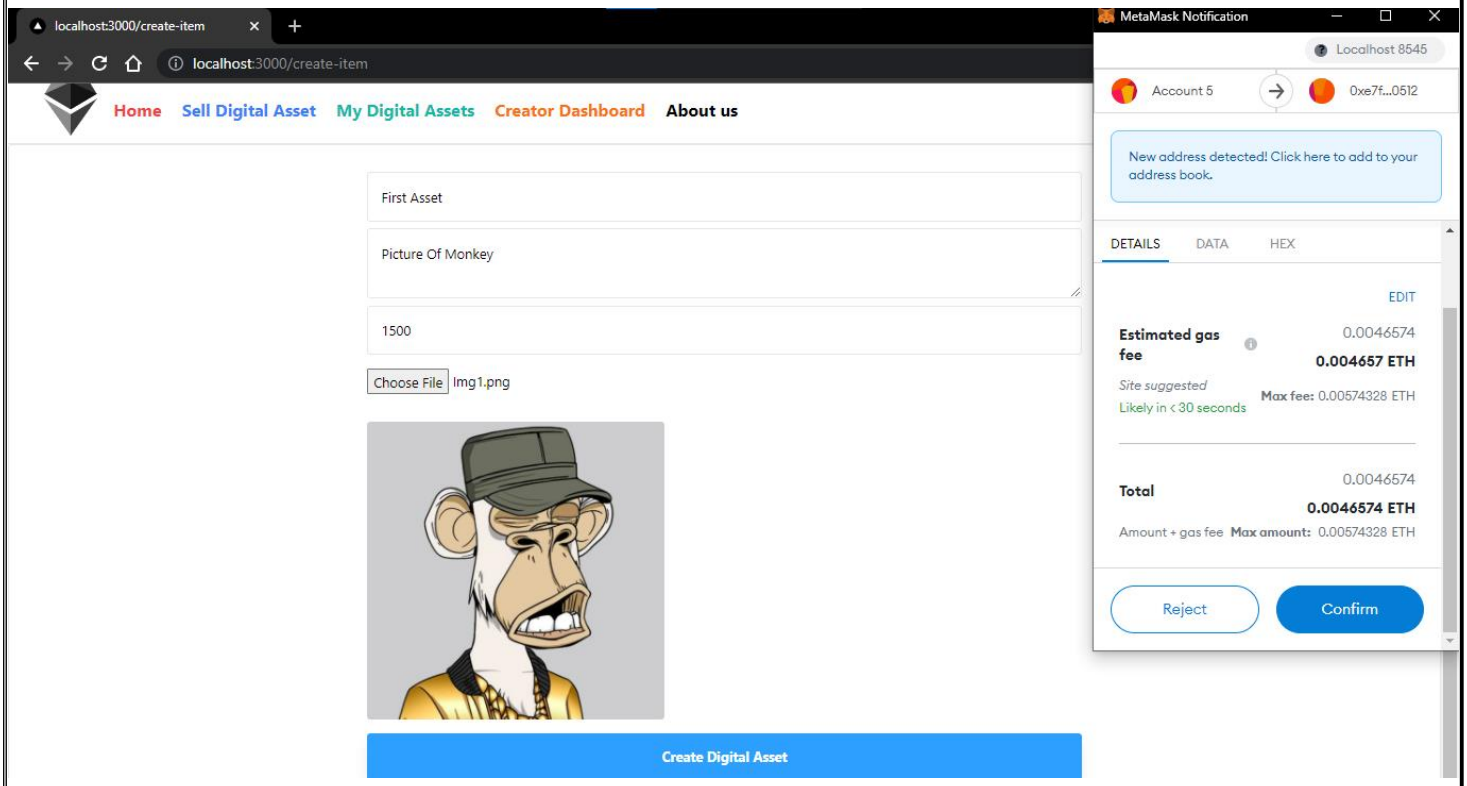
Create Asset page



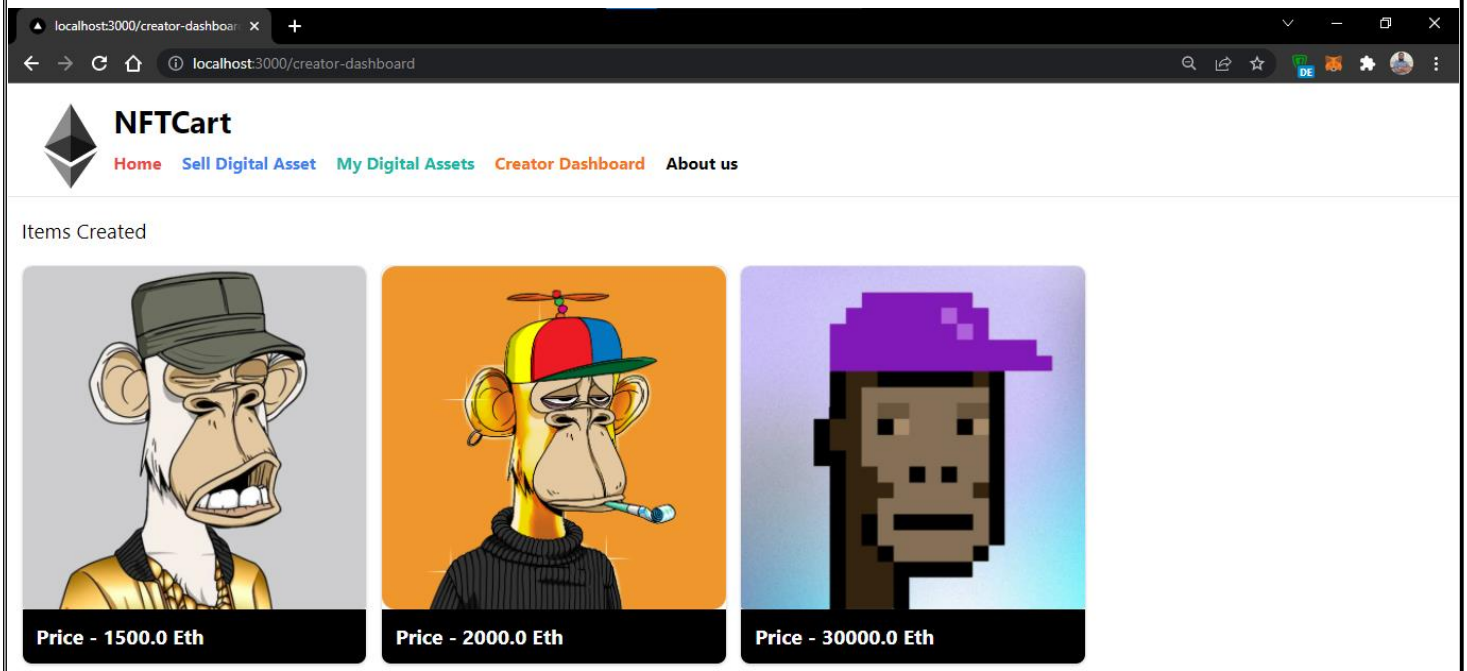
Digital Asset upload



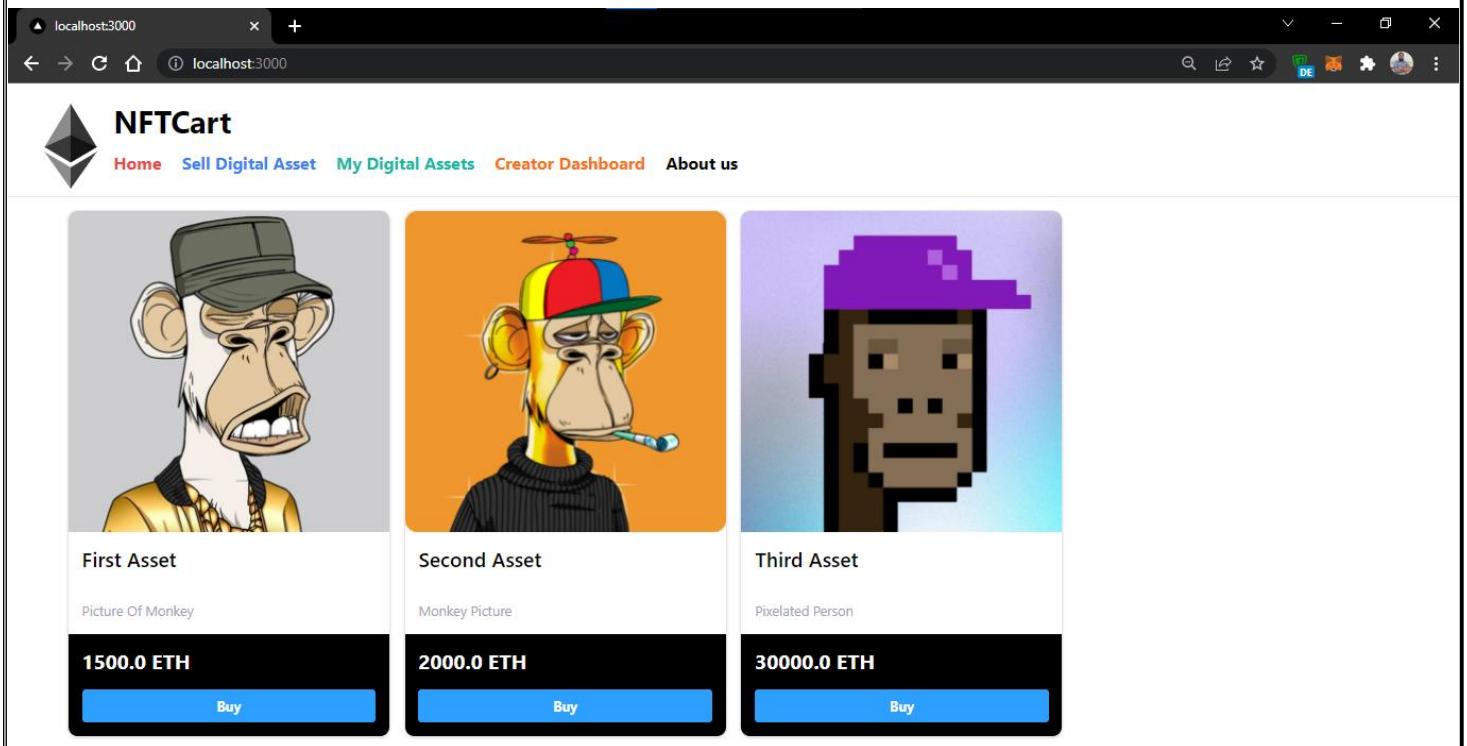
Transaction page



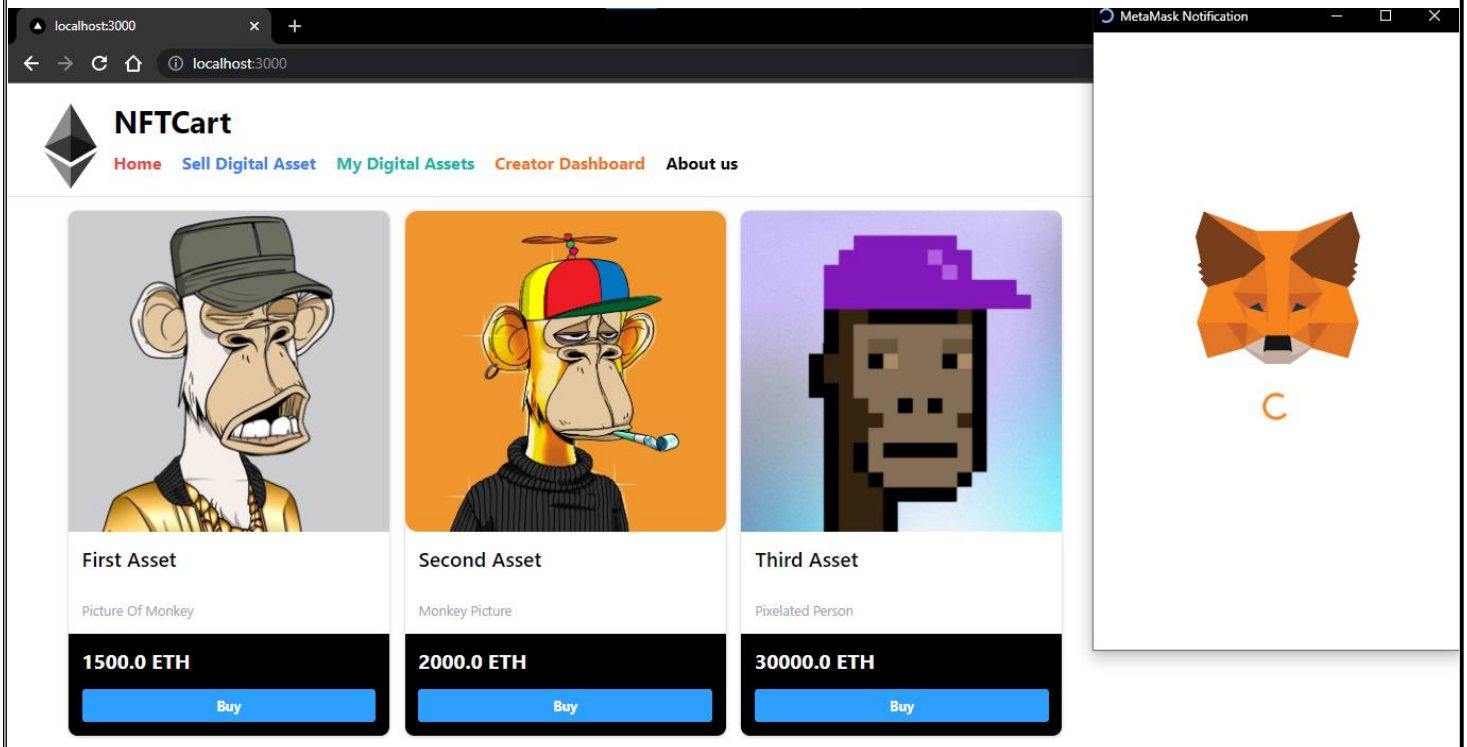
Transaction Done



Created Items

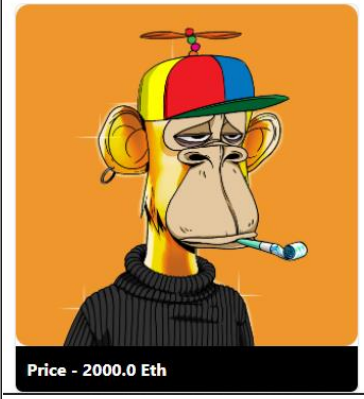


Home page after items added

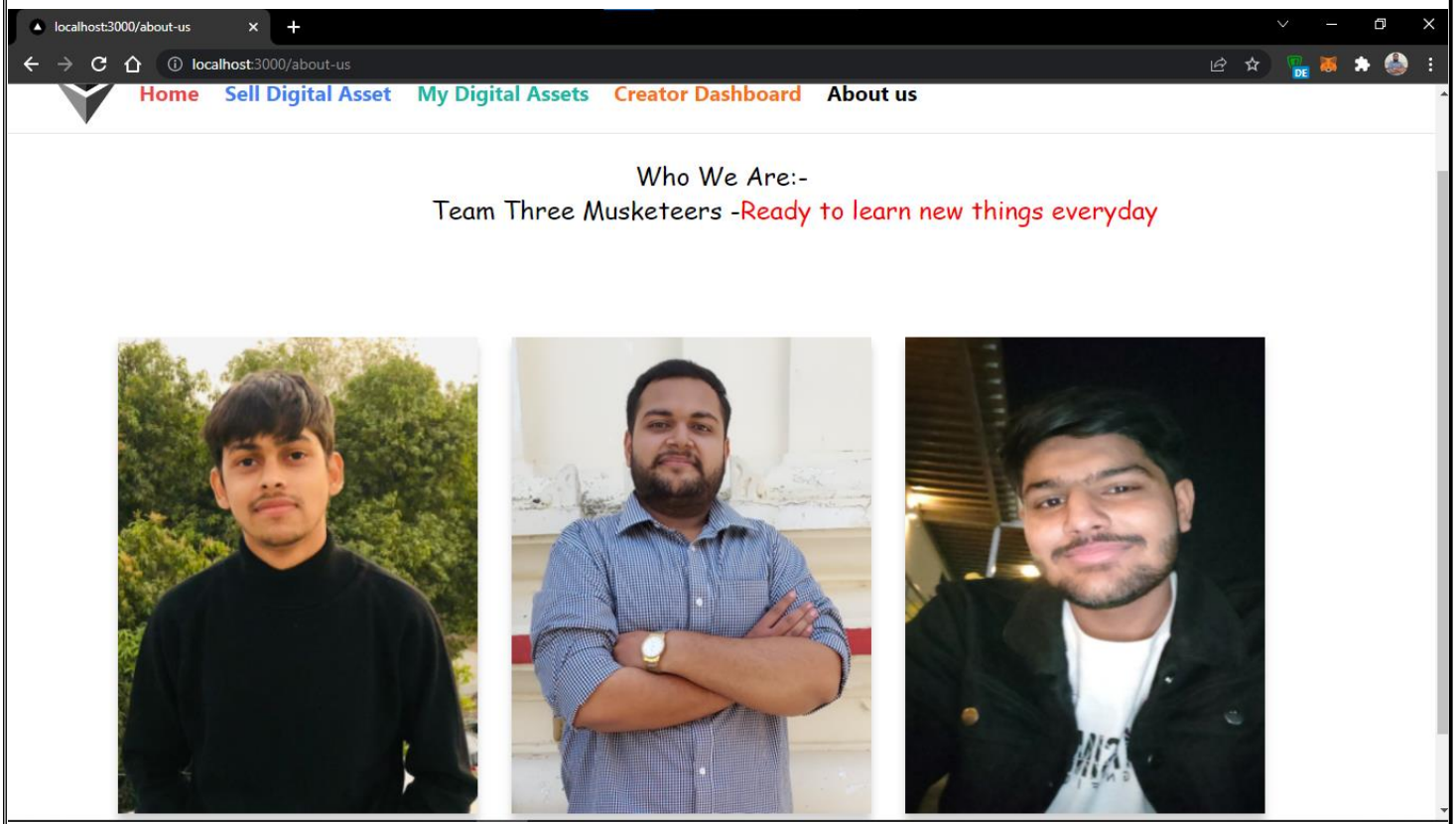


Selling Asset

Items sold



Dashboard after item sold



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 no other wallets like Coinbase wallet, walletconnect, fortmatic.

We will extend this further more to create those functionalities 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