School: .... Campus: ....

Academic Year: .... Subject Name: .... Subject Code: ....

Semester: .... Program: .... Branch: .... Specialization: ....

Date: ....

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiement : Build a Market – Basic NFT Marketplace Logic

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

ALGORITHM:

1.Initialize environment with Remix IDE and MetaMask on the Sepolia Testnet.
2.Write an ERC-721 compliant NFT smart contract using OpenZeppelin libraries.
3.Compile the smart contract in Remix using Solidity 0.8.x compiler.
4.Deploy the smart contract to the Sepolia Testnet through MetaMask.
5.Call the mint function to create an NFT with a specific metadata URI.
6.Verify the NFT minting transaction on Etherscan and confirm token ownership.

* Software used

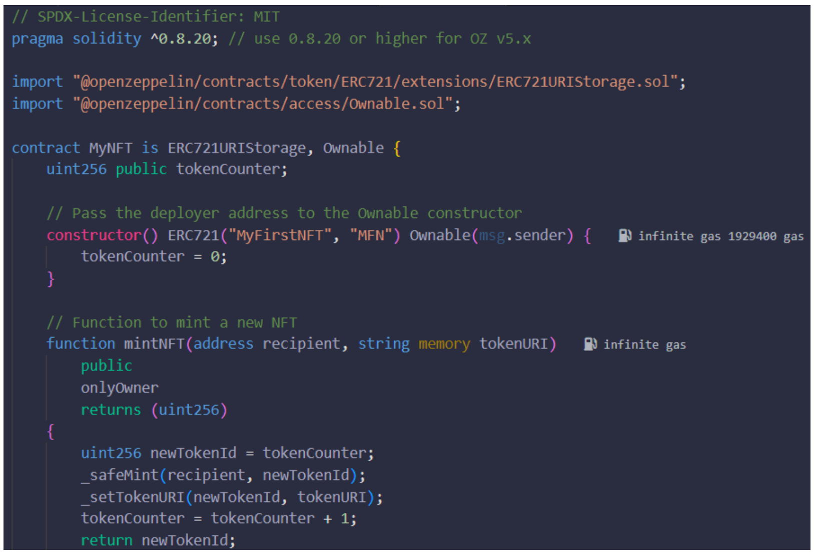
1.Remix IDE
2.MetaMask (Sepolia Testnet)
3.OpenZeppelin Contracts
4.IPFS / Pinata for storing NFT metadata
5.Etherscan Testnet for verification

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Testing Phase: Compilation of Code (error detection)

1.Compilation of Smart Contract

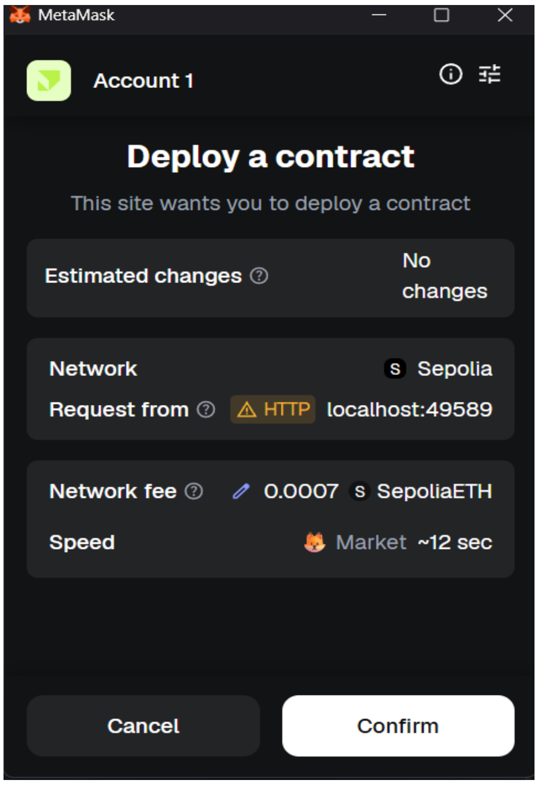
Open Remix IDE and paste the NFT smart contract code.
Select Solidity compiler version 0.8.20 or higher for compatibility with OpenZeppelin v5.x.
Click Compile MyNFT.sol and ensure no syntax errors occur.
Successful compilation indicates the contract is ready for deployment.



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20; // use 0.8.20 or higher for OZ v5.x

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

contract MyNFT is ERC721URIStorage, Ownable {
    uint256 public tokenCounter;

    // Pass the deployer address to the Ownable constructor
    constructor() ERC721("MyFirstNFT", "MFN") Ownable(msg.sender) {    infinite gas 1929400 gas
        tokenCounter = 0;
    }

    // Function to mint a new NFT
    function mintNFT(address recipient, string memory tokenURI)    infinite gas
        public
        onlyOwner
        returns (uint256)
    {
        uint256 newTokenId = tokenCounter;
        _safeMint(recipient, newTokenId);
        _setTokenURI(newTokenId, tokenURI);
        tokenCounter = tokenCounter + 1;
        return newTokenId;
```

2.Deployment on Sepolia Testnet

Connect MetaMask to the Sepolia Test Network.
In Remix, choose Injected Provider – MetaMask as the environment.
Click Deploy and approve the transaction in MetaMask.
Once confirmed, Remix displays the contract address under “Deployed Contracts.”
This confirms that the NFT contract is live on the blockchain.



MetaMask

Account 1

Deploy a contract

This site wants you to deploy a contract

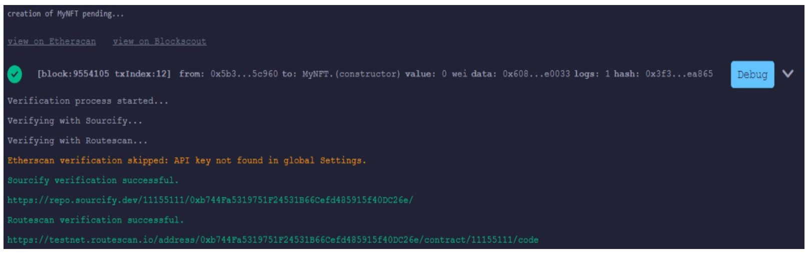
Estimated changes: No changes

Network: Sepolia

Request from: HTTP localhost:49589

Network fee: 0.0007 SepoliaETH

Speed: Market ~12 sec



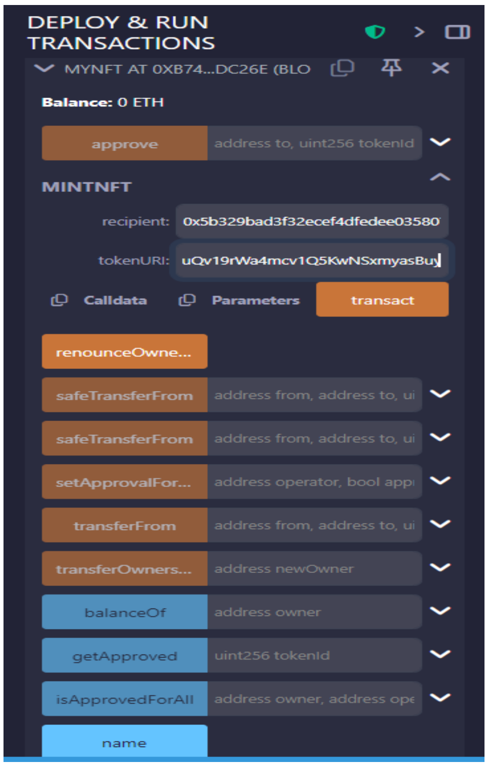
```
creation of MyNFT pending...

view on Etherscan   view on Blockscout

[block:9554105 txIndex:12] from: 0x5b3...5c960 to: MyNFT.(constructor) value: 0 wei data: 0x608...e0033 logs: 1 hash: 0x3f3...ea865
Verification process started...
Verifying with Sourcify...
Verifying with Routescan...
Etherscan verification skipped: API key not found in global Settings.
Sourcify verification successful.
https://repo.sourcify.dev/11155111/0xb744Fa5319751F24531B66Cefd485915f40DC26e/
Routescan verification successful.
https://testnet.routescan.io/address/0xb744Fa5319751F24531B66Cefd485915f40DC26e/contract/11155111/code
```

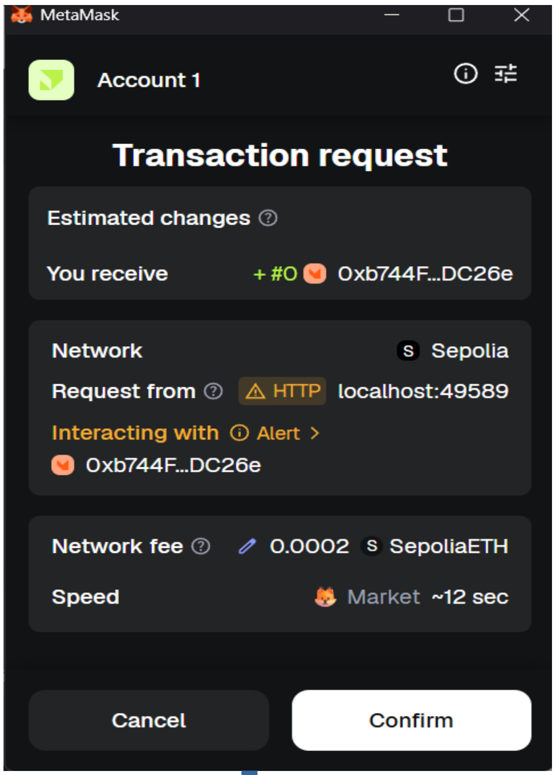
*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

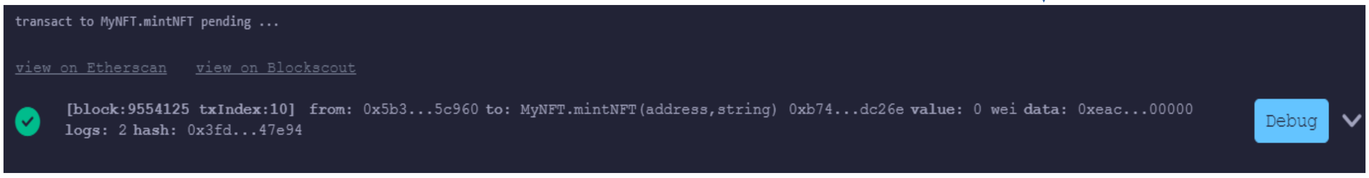
* Testing Phase: Compilation of Code (error detection)

3.Linking Metadata and Minting NFT

Expand the deployed contract in Remix under Deployed Contracts.
Locate and execute the mintNFT() function.
Input:
Recipient Address: your MetaMask wallet address.
Token URI: the IPFS link to the NFT's metadata JSON file.
Click Transact and confirm the mint transaction in MetaMask.
Once mined, the NFT is minted and assigned to your wallet.


DEPLOY & RUN TRANSACTIONS
MYNFT AT 0XB74...DC26E (BLO
Balance: 0 ETH
approve — address to, uint256 tokenId
MINTNFT
recipient: 0x5b329bad3f32ecef4dfedee03580
tokenURI: uQv19rWa4mcv1Q5KwNSxmyasBuy
Calldata | Parameters | transact
renounceOwne...
safeTransferFrom — address from, address to, ui
safeTransferFrom — address from, address to, ui
setApprovalFor... — address operator, bool appr
transferFrom — address from, address to, ui
transferOwners... — address newOwner
balanceOf — address owner
getApproved — uint256 tokenId
isApprovedForAll — address owner, address ope
name

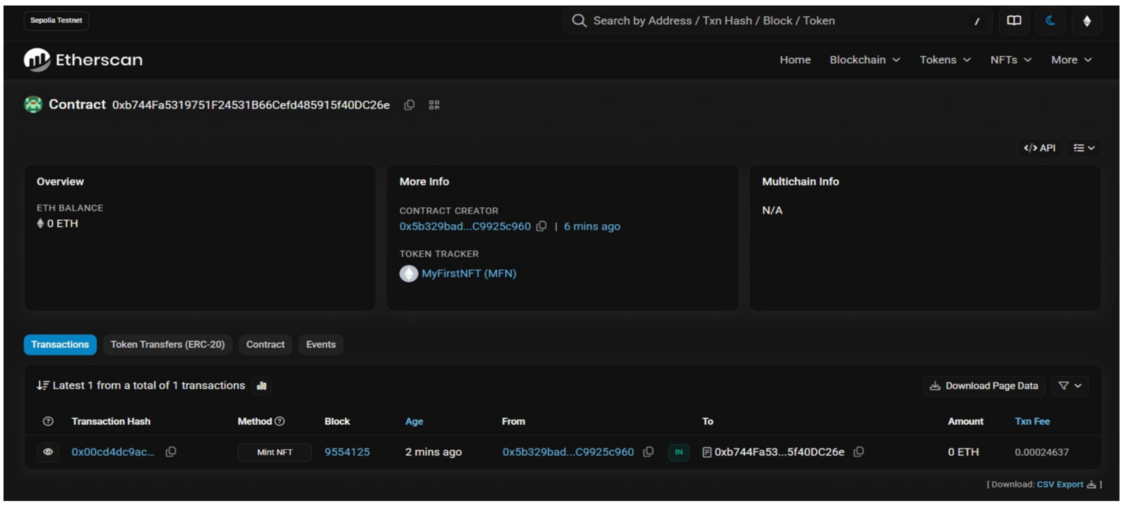

MetaMask
Account 1
Transaction request
Estimated changes
You receive + #0 0xb744F...DC26e
Network: Sepolia
Request from: HTTP localhost:49589
Interacting with: Alert
0xb744F...DC26e
Network fee: 0.0002 SepoliaETH
Speed: Market ~12 sec


```
transact to MyNFT.mintNFT pending ...

view on Etherscan   view on Blockscout

[block:9554125 txIndex:10] from: 0x5b3...5c960 to: MyNFT.mintNFT(address,string) 0xb74...dc26e value: 0 wei data: 0xeac...00000
logs: 2 hash: 0x3fd...47e94
```

4.Verification of Mint Transaction

Copy the transaction hash from Remix.
Visit https://sepolia.etherscan.io and paste the hash in the search bar.
Check transaction details, including contract interaction and confirmation.
This step verifies successful on-chain recording of the NFT minting.

* Testing Phase: Compilation of Code (error detection)



Sepolia Testnet

Contract 0xb744Fa5319751F24531B66Cefd485915f40DC26e

Overview
ETH BALANCE
0 ETH

More Info
CONTRACT CREATOR
0x5b329bad...C9925c960 | 6 mins ago
TOKEN TRACKER
MyFirstNFT (MFN)

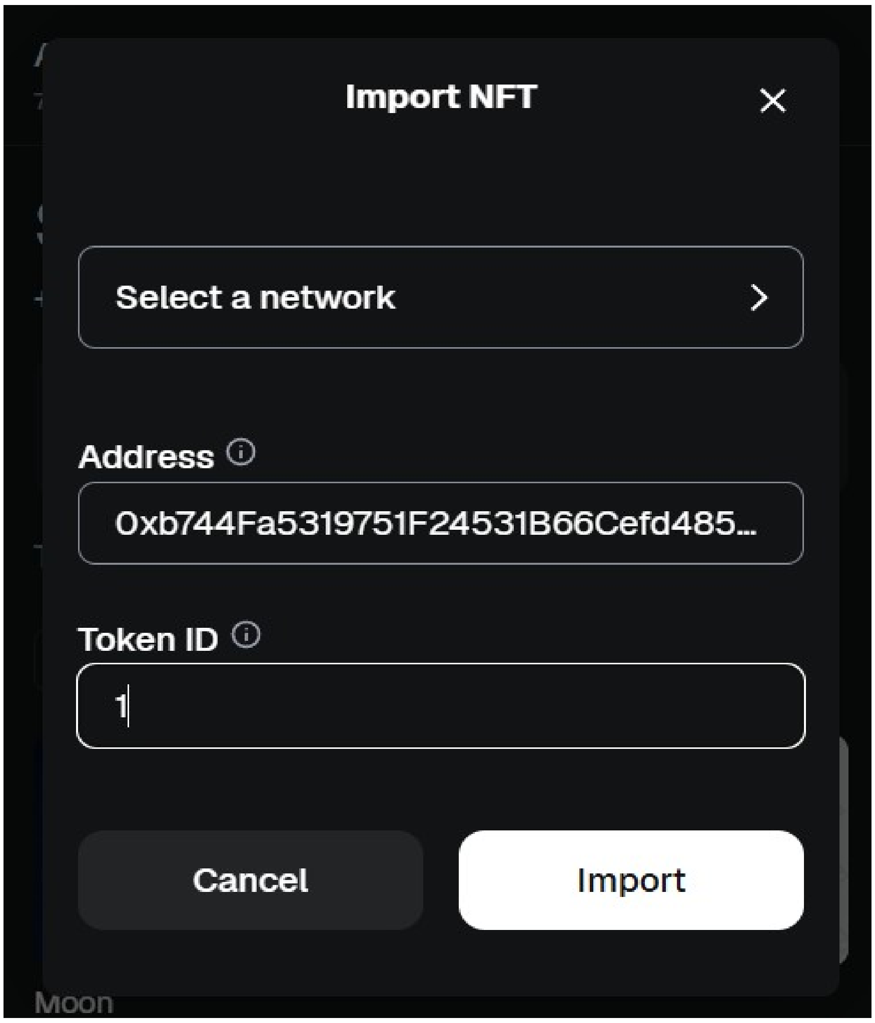
Multichain Info
N/A

Latest 1 from a total of 1 transactions

	Transaction Hash	Method	Block	Age	From		To	Amount	Txn Fee
	0x00cd4dc9ac...	Mint NFT	9554125	2 mins ago	0x5b329bad...C9925c960	IN	0xb744Fa53...5f40DC26e	0 ETH	0.00024637

5.Viewing the NFT in MetaMask

Open MetaMask → NFTs Tab → Import NFTs.
Enter the contract address and Token ID (e.g., 0).
Your NFT will appear in MetaMask, showing that it's owned by your wallet.
This confirms successful minting and ownership verification.



Import NFT

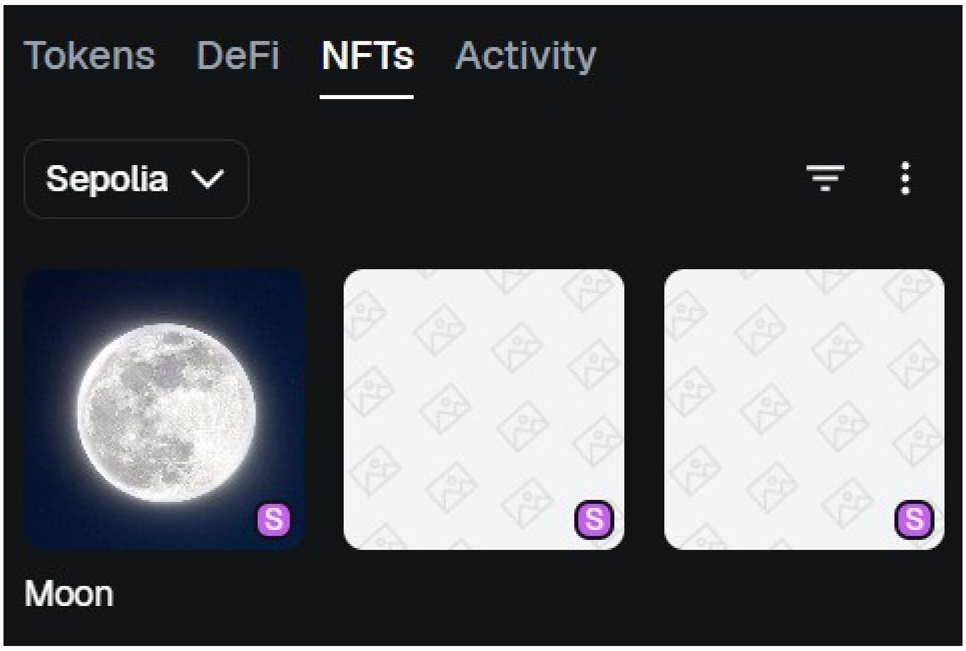
Address: 0xb744Fa5319751F24531B66Cefd485...

Token ID: 1

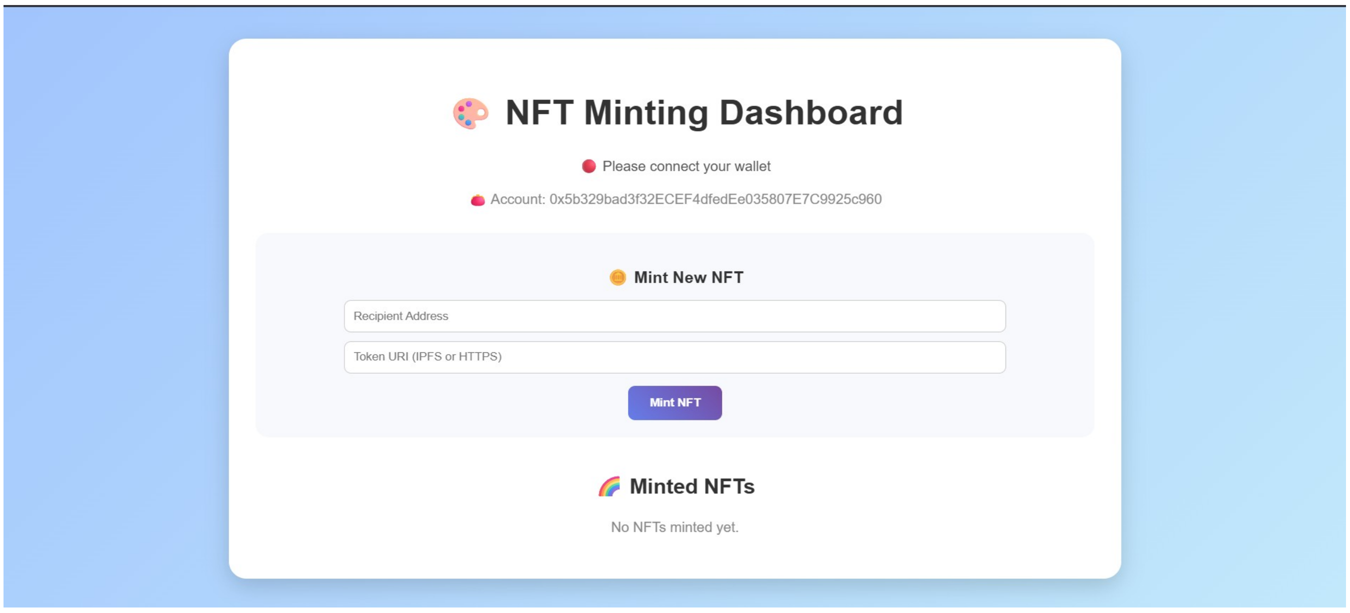
*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Testing Phase: Compilation of Code (error detection)

-NFT was successfully minted and deployed on Sepolia Testnet.
-Metadata (name, image, and description) fetched correctly from IPFS.
-Transaction verified through Etherscan, confirming blockchain interaction.
-Ownership recorded immutably on Ethereum blockchain.



* Implementation Phase: Final Output (no error)


🎨 NFT Minting Dashboard

Please connect your wallet

Account: 0x5b329bad3f32ECEF4dfedEe035807E7C9925c960

Mint New NFT

Recipient Address

Token URI (IPFS or HTTPS)

Mint NFT

🌈 Minted NFTs

No NFTs minted yet.

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Observations

1.Successfully connected the MetaMask wallet using Ethers.js (v6) and interacted with the deployed NFT smart contract.

2.Minting transactions were verified on the blockchain, and each NFT was assigned a unique token ID and metadata.

3.The frontend displayed minted NFTs dynamically with their image, name, and owner address.

4.The integration between smart contract and React frontend worked seamlessly, ensuring smooth NFT deployment and visualization.

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	**50**		

Signature of the Student:

Name :

Regn. No. :

Signature of the Faculty:

****As applicable according to the experiment. Two sheets per experiment (10-20) to be used.***