Centurion UNIVERSITY
Shaping Lives... Empowering Communities...

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 : Mint it Yourself – NFT Creation and Deployment

Objective/Aim:

To create and deploy a Non-Fungible Token (NFT) smart contract using Ethereum testnet and OpenZeppelin libraries, and mint your own NFT by linking it with metadata stored on IPFS.

Apparatus/Software Used:

- MetaMask Wallet
- Brave Web Browser
- Remix IDE – https://remix.ethereum.org
- IPFS Pinning Service (nft.storage / Pinata)
- Ethereum Sepolia Testnet

Theory/Concept:

NFT (Non-Fungible Token)

NFTs are unique blockchain-based tokens that represent ownership of digital assets. They follow the **ERC-721 standard**, which assigns a unique tokenId to each collectible or asset.

IPFS (InterPlanetary File System)

A decentralized storage system used to store NFT images and metadata. NFTs typically store only the **CID (Content Identifier)**, not the actual image.

ERC-721 TokenURI

Every NFT includes a tokenURI pointing to metadata stored on IPFS. The metadata file contains:

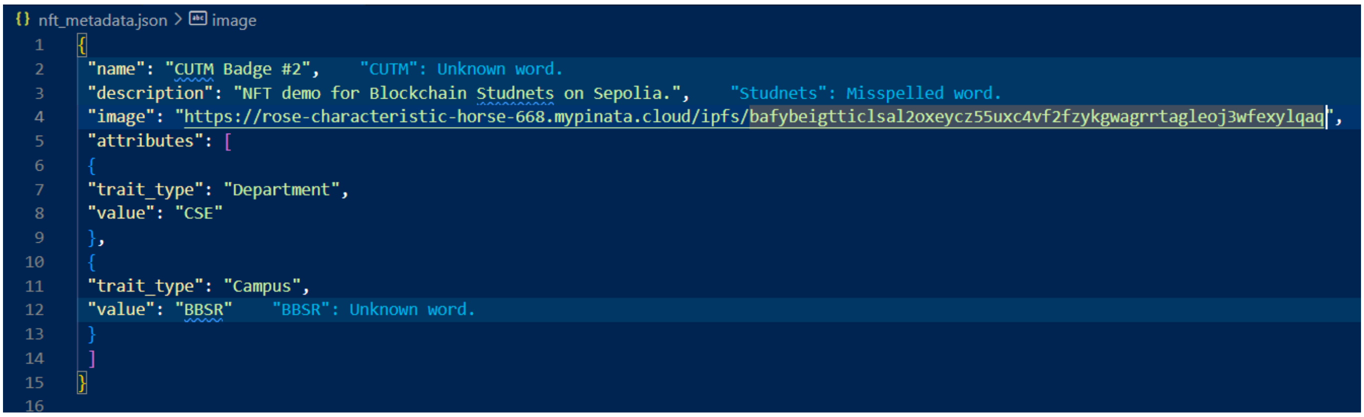
- name
- description
- image (IPFS link)

Deployment Flow Summary

1. Upload asset → IPFS
2. Create metadata.json → IPFS
3. Deploy ERC-721 contract on testnet
4. Mint NFT → provide tokenURI

Procedure:

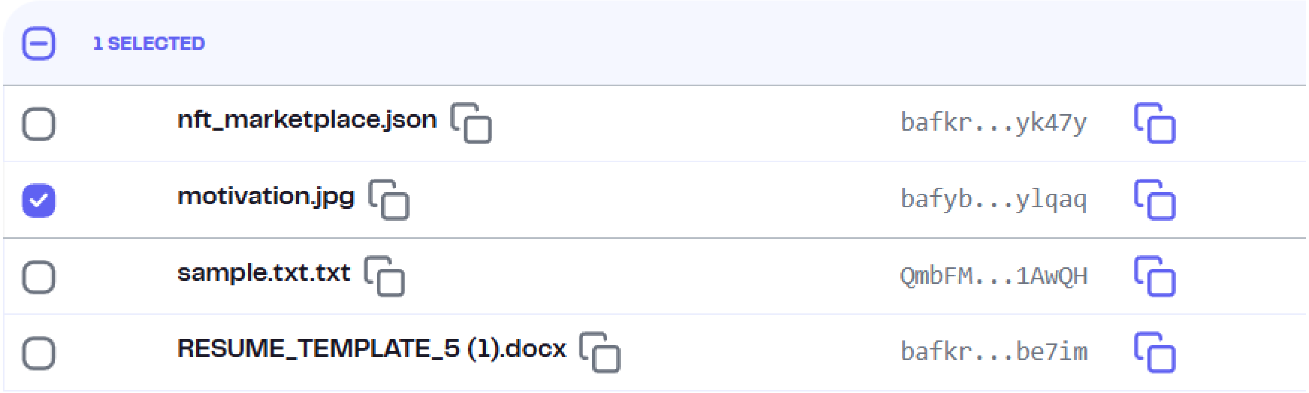
1. Open MetaMask and switch to the Ethereum Sepolia Testnet.
2. Visit any IPFS pinning service (nft.storage / Pinata) in your browser.
3. Upload an image or file that will be used as the NFT’s media asset.
4. Copy the CID generated after uploading (this will be used in metadata).
5. Create a metadata.json file containing name, description, and the IPFS link to the image.
6. Upload metadata.json to IPFS and copy the metadata CID.
7. Open Remix IDE in your browser.
8. Create a new Solidity file and paste an ERC-721 NFT contract using OpenZeppelin.
9. Compile the contract using the Solidity compiler.
10. Click Deploy, select Injected Provider – MetaMask as the environment.
11. Approve the MetaMask connection and deploy the contract to Sepolia.
12. After deployment, go to the deployed contract section.
13. Locate the mint function (e.g., mintNFT(address, tokenURI)).
14. Paste your wallet address as the recipient.
15. In the tokenURI field, enter:
    ipfs://<metadata_CID>
16. Click Transact and confirm the minting transaction in MetaMask.
17. Wait for the transaction confirmation and record the transaction hash.
18. View the minted NFT on an NFT explorer (OpenSea Testnet or NFTScan).



{} nft_metadata.json > image

```
{
"name": "CUTM Badge #2",     "CUTM": Unknown word.
"description": "NFT demo for Blockchain Studnets on Sepolia.",     "Studnets": Misspelled word.
"image": "https://rose-characteristic-horse-668.mypinata.cloud/ipfs/bafybeigtticlsal2oxeycz55uxc4vf2fzykgwagrrtagleoj3wfexylqaq",
"attributes": [
{
"trait_type": "Department",
"value": "CSE"
},
{
"trait_type": "Campus",
"value": "BBSR"     "BBSR": Unknown word.
}
]
}
```

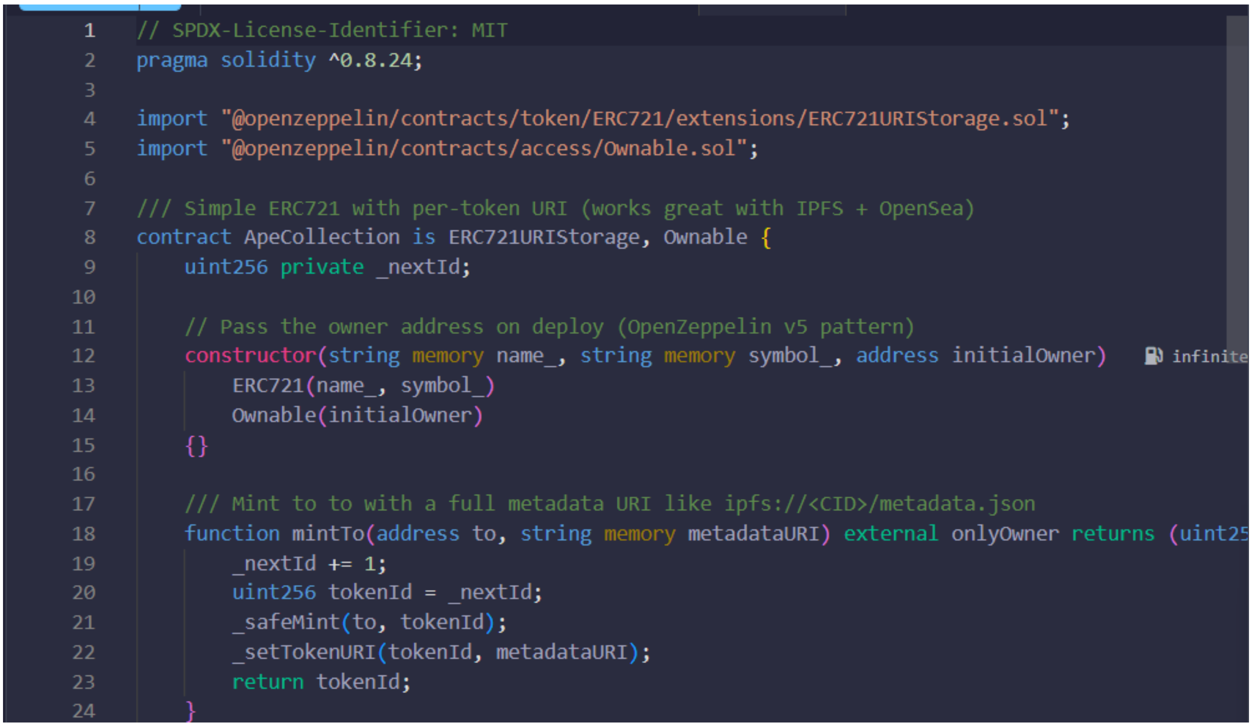
.json file



⊖ 1 SELECTED

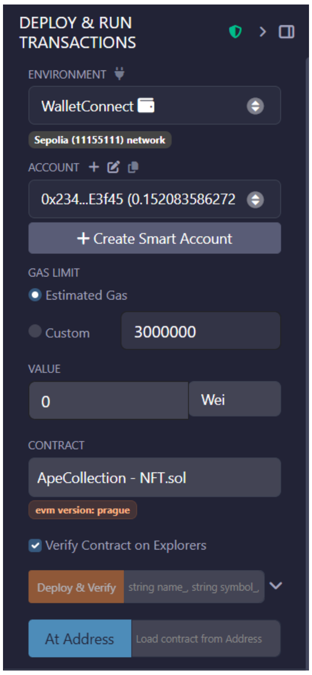
☐	nft_marketplace.json	bafkr...yk47y
☑	motivation.jpg	bafyb...ylqaq
☐	sample.txt.txt	QmbFM...1AwQH
☐	RESUME_TEMPLATE_5 (1).docx	bafkr...be7im

Pinata file



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.24;

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

/// Simple ERC721 with per-token URI (works great with IPFS + OpenSea)
contract ApeCollection is ERC721URIStorage, Ownable {
    uint256 private _nextId;

    // Pass the owner address on deploy (OpenZeppelin v5 pattern)
    constructor(string memory name_, string memory symbol_, address initialOwner)    infinite
        ERC721(name_, symbol_)
        Ownable(initialOwner)
    {}

    /// Mint to to with a full metadata URI like ipfs://<CID>/metadata.json
    function mintTo(address to, string memory metadataURI) external onlyOwner returns (uint25
        _nextId += 1;
        uint256 tokenId = _nextId;
        _safeMint(to, tokenId);
        _setTokenURI(tokenId, metadataURI);
        return tokenId;
    }
```

Solidity code



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: WalletConnect

Sepolia (11155111) network

ACCOUNT: 0x234...E3f45 (0.152083586272

+ Create Smart Account

GAS LIMIT

● Estimated Gas

○ Custom 3000000

VALUE: 0 Wei

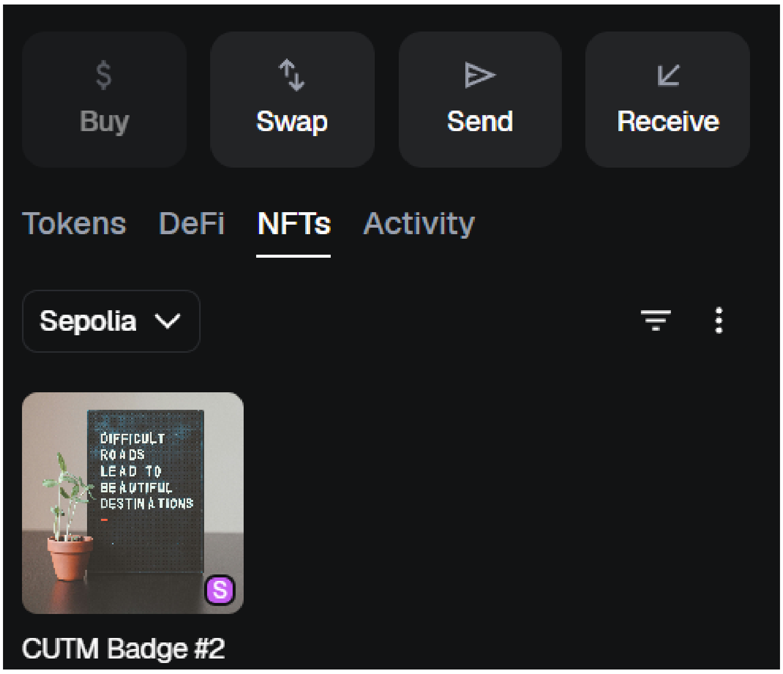
CONTRACT: ApeCollection - NFT.sol

evm version: prague

☑ Verify Contract on Explorers

Deploy & Verify | string name_, string symbol_

At Address | Load contract from Address



CUTM Badge #2

Deploy of contract **NFT**

Observation

S.No	Step Description	Observation / Result
1	MetaMask connected to Sepolia Testnet	Successful
2	NFT image uploaded to IPFS	CID generated
3	Metadata JSON created and uploaded	CID generated
4	Smart Contract deployed on testnet	Deployed Address noted
5	mintNFT() function executed	Transaction successful
6	NFT viewed on Etherscan / OpenSea testnet	NFT visible and verified

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: