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

Coding Phase : Pseudo Code/Flow Chart/Algorithm

Begin process.

- Upload image file to Pinata and obtain CID.
- Create a JSON metadata file linking to that CID.
- Upload JSON to Pinata and obtain metadata CID.
- Write NFT contract using OpenZeppelin ERC721URIStorage.
- Compile the contract in Remix IDE.
- Deploy contract on Sepolia Testnet using MetaMask.
- Confirm transaction → NFT created.
- Verify NFT on MetaMask.
- End.

Apparatus/Software Used:

1 . MetaMask Wallet

2 . Remix IDE (Solidity Compiler 0.8.20)

3 . Pinata (for IPFS hosting)

4 . Sepolia Ethereum Testnet + Faucet

5 . OpenZeppelin ERC721 Library

6 . OpenSea Testnet (NFT Viewer)

Testing Phase :

1. Compilation: No errors in Remix with Solidity `0.8.24`.

2. Deployment: Successfully deployed with constructor args → (name, symbol, owner)

3. Minting: `mintTo(myWallet, "ipfs://CID/metadata.json")` worked.

4. Verification: NFT appeared in MetaMask + OpenSea testnet with correct metadata.

Implementation Phase: Final Output (No Error) :

1. Contract deployed on Sepolia testnet.

```
// 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 gas 1930600 gas
        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 (uint256) {    infinite gas
        _nextId += 1;
        uint256 tokenId = _nextId;
        _safeMint(to, tokenId);
        _setTokenURI(tokenId, metadataURI);
        return tokenId;
    }

    function totalMinted() external view returns (uint256) {    2454 gas
        return _nextId;
    }
}
```

2.NFT minted with `ipfs://` metadata URI.

3.`totalMinted()` increased correctly after minting.

4,NFT visible on **OpenSea Testnet** with image + description.

Observation:

1. Contract follows **OpenZeppelin best practices** (secure + upgradeable).
2. Only **owner can mint**, making it suitable for curated collections.
3. `metadataURI flexibility → Any IPFS-hosted JSON can be attached.`
4. OpenSea successfully fetches and displays metadata from Pinata.
5. Works perfectly as a **collection-style NFT project**.

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.