Centurion UNIVERSITY

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 :

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

Algorithm:

1. Players own NFT assets like in-game characters (spaceships).
2. Each NFT is created using a smart contract (ERC-721 standard).
3. Players can upgrade their NFT character (e.g., level up).
4. Smart contract ensures ownership and level attribute management.
5. Interactions are made via blockchain wallet (e.g., MetaMask).

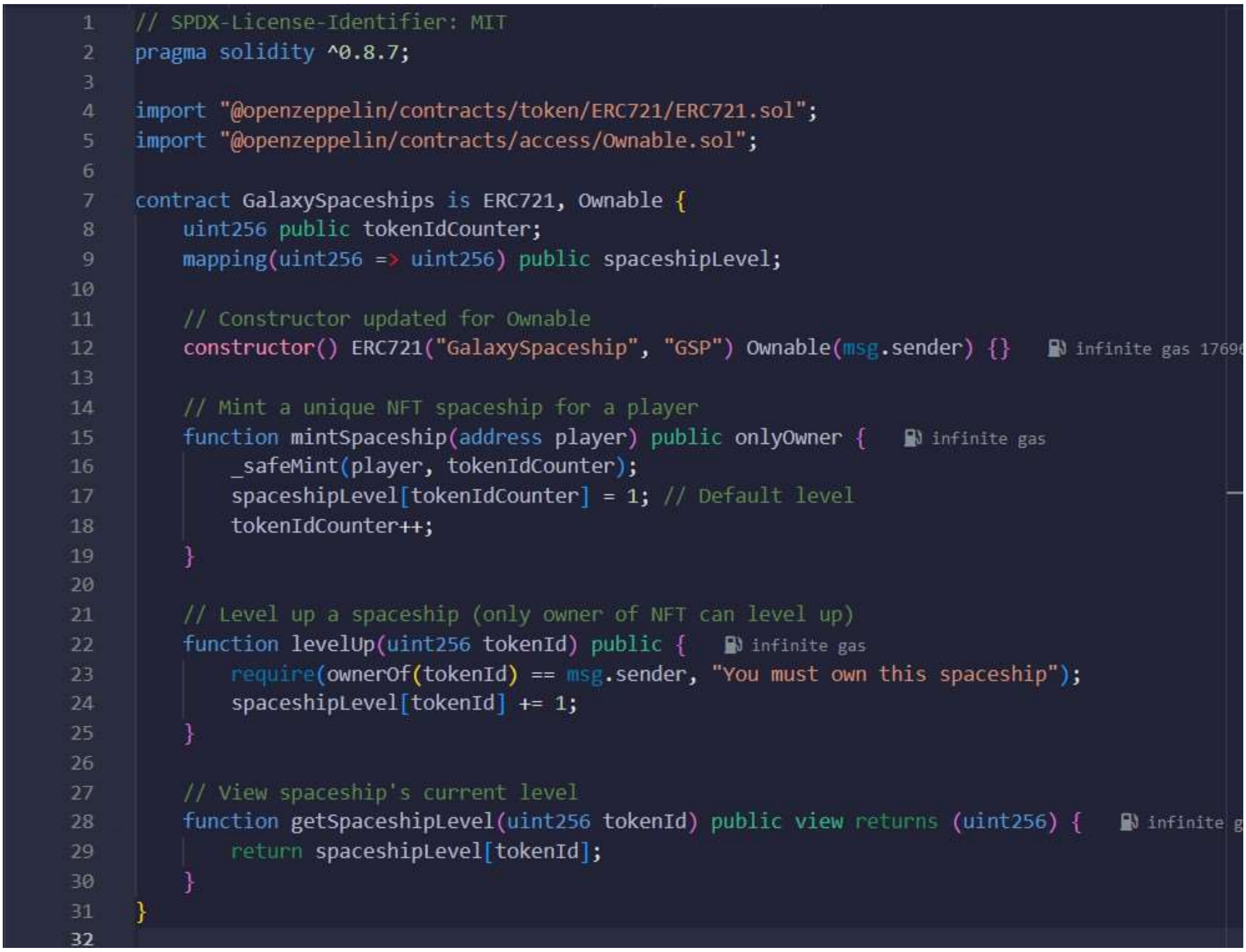
* Softwares used

1. Remix IDE
2. Solidity ^0.8.7
3. MetaMask Wallet
4. OpenZeppelin Contracts.

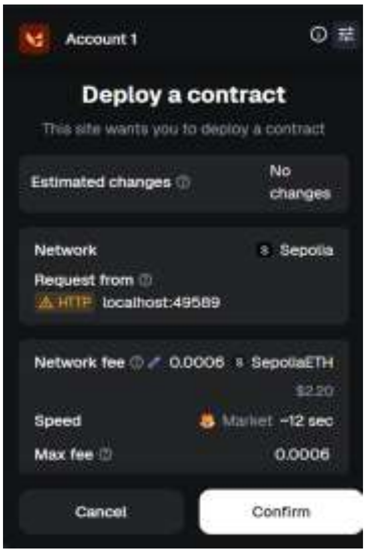
*****_As applicable according to the experiment._
Two sheets per experiment (10-20) to be used.

* Testing Phase: Compilation of Code (error detection)

1. Smart contract compiled and deployed using Remix on Sepolia testnet.
2. Minted spaceship NFT using mintSpaceship function.
3. Verified NFT appears in MetaMask under "NFTs" tab for owner address.
4. Used levelUp function to increase spaceship level from 1 to higher values.
5. getSpaceshipLevel returned correct values confirming logic works properly.



```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.7;

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

contract GalaxySpaceships is ERC721, Ownable {
    uint256 public tokenIdCounter;
    mapping(uint256 => uint256) public spaceshipLevel;

    // Constructor updated for Ownable
    constructor() ERC721("GalaxySpaceship", "GSP") Ownable(msg.sender) {}

    // Mint a unique NFT spaceship for a player
    function mintSpaceship(address player) public onlyOwner {
        _safeMint(player, tokenIdCounter);
        spaceshipLevel[tokenIdCounter] = 1; // Default level
        tokenIdCounter++;
    }

    // Level up a spaceship (only owner of NFT can level up)
    function levelUp(uint256 tokenId) public {
        require(ownerOf(tokenId) == msg.sender, "You must own this spaceship");
        spaceshipLevel[tokenId] += 1;
    }

    // View spaceship's current level
    function getSpaceshipLevel(uint256 tokenId) public view returns (uint256) {
        return spaceshipLevel[tokenId];
    }
}

```



Account 1

Deploy a contract

This site wants you to deploy a contract

Estimated changes: No changes

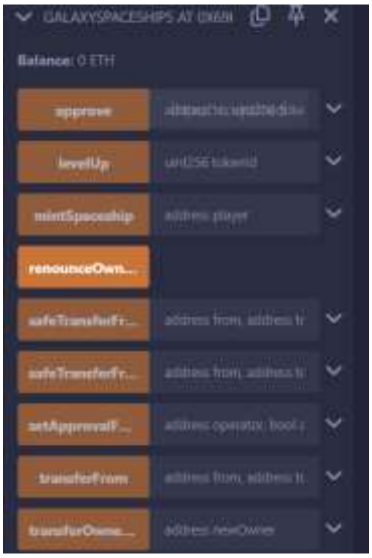
Network: Sepolia

Request from: ⚠ HTTP localhost:49589

Network fee: 0.0006 SepoliaETH $2.20

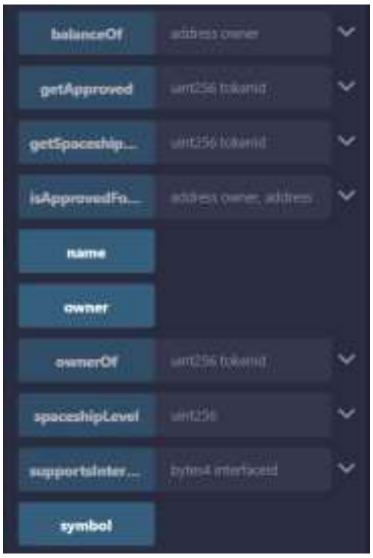
Speed: Market ~12 sec

Max fee: 0.0006



GALAXYSPACESHIPS AT 0X6B

Balance: 0 ETH



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

* Implementation Phase: Final Output (no error)

1. NFT successfully minted with a default level = 1.
2. NFT ownership verified via blockchain explorer (e.g., Sepolia Etherscan).
3. Level upgraded by contract interaction using levelUp().
4. Smart and secure ownership logic enforced by contract checks.

* Observations

1. NFT smart contracts allow for in-game ownership and user-controlled upgrades.
2. Using blockchain, users can own and trade assets independently of game servers.
3. Leveling logic adds value to the NFT, making it usable in a GameFi environment.
4. This basic implementation can be expanded into a full P2E blockchain game.

ASSESSMENT

Rubrics	Full Mark	Marks Obtalned	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.