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 : Token Launch – Deploying a Token Locally

Objective/Aim:

To understand the steps involved in creating and launching a custom ERC20 token smart contract on a local blockchain network using Hardhat, and to observe how smart contracts function in a development environment.

Apparatus/Software Used:

- Laptop / PC
- Remix IDE
- Metamask
- Etherscan

Theory/Concept:

- **ERC20 Token Standard**: A fungible token standard defining basic functions like transfer, approve, balanceOf, etc.
- **Local Deployment / Testnet Deployment**:
  - Local deployment uses simulated blockchain (Ganache/Hardhat).
  - Testnet deployment uses real test networks like Goerli or Sepolia with test ETH.
- Testnets (like **Sepolia**) are public Ethereum networks that function like mainnet but use **test ETH** instead of real ETH.
- Developers deploy contracts on testnets to simulate real-world conditions (gas fees, confirmations, multiple users) while avoiding financial risk.
- Testnets allow interaction with wallets (like MetaMask), block explorers (like Etherscan testnet), and dApps.

Procedure:

Step 1: **Write Token Contract (in Remix IDE).**

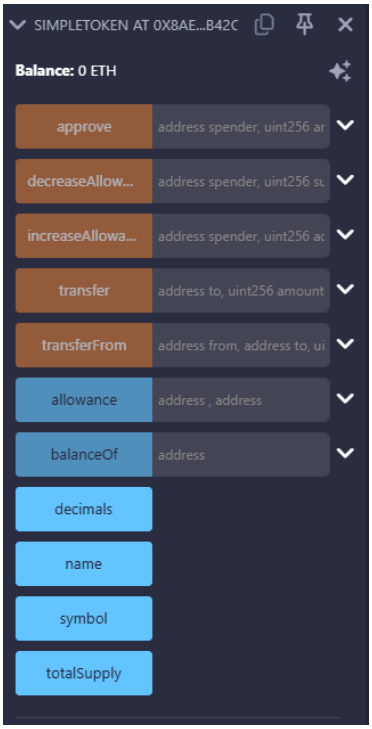
Step 2: **Compile and Deploy Token**

- In Remix → "Compile the smart contract "then "Deploy & Run Transactions" tab.
- Select **Injected Web3** (MetaMask connected to local testnet).

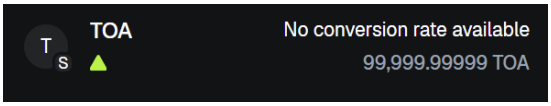
Step 3: Import your token to wallet **click 3 dots→import token copy and paste the token address there and select the network like sepolia.**

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

contract SimpleToken {
    string public name;
    string public symbol;
    uint8 public decimals = 18;
    uint256 public totalSupply;

    mapping(address => uint256) public balanceOf;
    mapping(address => mapping(address => uint256)) public allowance;

    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);

    constructor(string memory _name, string memory _symbol, uint256 _initialSupply) {
        name = _name;
        symbol = _symbol;
        totalSupply = _initialSupply * (10**uint256(decimals));
        balanceOf[msg.sender] = totalSupply;
        emit Transfer(address(0), msg.sender, totalSupply);
    }

    function transfer(address to, uint256 amount) external returns (bool) {
        _transfer(msg.sender, to, amount);
```

Contract



contract features



Token showing in the wallet

Observation

- Token contract deployed successfully.
- Contract address was visible and verified through Avee.net.
- Initial supply was credited to the deployer's wallet (MetaMask).
- Able to transfer tokens and check balances on Avee.net interface.

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 :

Signature of the Faculty:

Regn. No. :