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
- App.aave
- 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 **Goerli** or **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.
- **Avee.net** provides a **front-end interface** for interacting with deployed ERC20 tokens (view balances, make transfers, approve spenders, etc.).

Procedure:

Step 1: Write Token Contract (in Remix IDE)
Step 2: Open Remix IDE → Select compiler **0.8.x** → Compile MyToken.sol.
Step 3: **Deploy Token**

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

Step 4: **Verify on Avee.net**

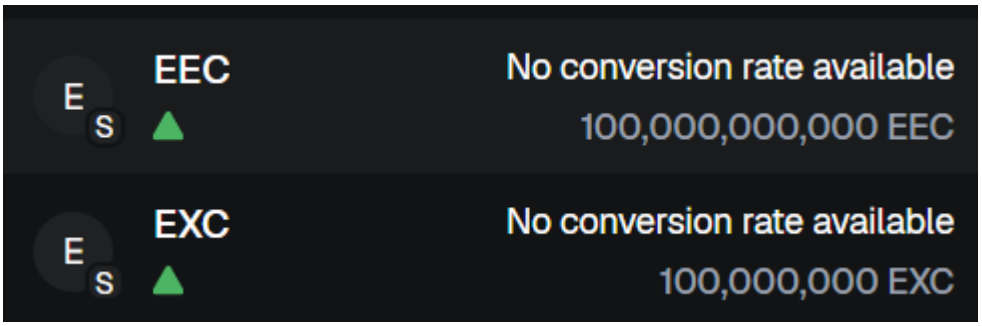
- Copy deployed contract address.
- Go to **Avee.net** → Add Token → Paste contract address.
- Token appears in the dashboard.

Step 5: **Interact with Token**

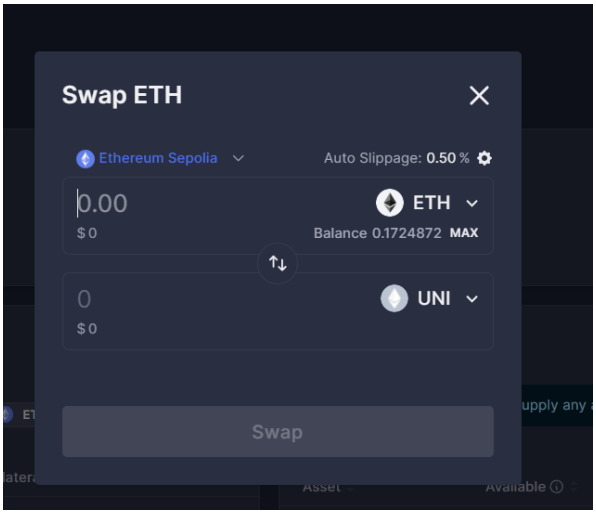
- Check total supply.
- Transfer tokens between MetaMask accounts.
- Use Avee.net to approve and transfer tokens on behalf of another account.

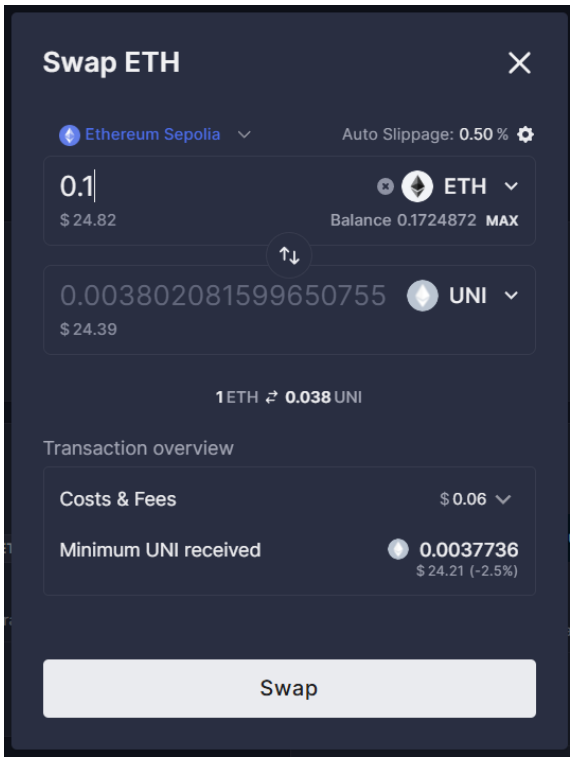
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.27;

import {ERC20} from "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import {ERC20Permit} from "@openzeppelin/contracts/token/ERC20/extensions/ERC20Permit.sol";

contract MyToken is ERC20, ERC20Permit {
    constructor(string memory name, string memory symbol)    infinite gas 1399400 gas
        ERC20(name, symbol)
        ERC20Permit(name) // ✅ Required for EIP-2612 (Permit)
    {
        _mint(msg.sender, 1000000 * 10 ** decimals());
    }
}

```

contract



Token





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 :

Regn. No. :

Signature of the Faculty: