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 : Build DeFi – AMM or Lending Prototype

Objective/Aim:

Perform basic operations such as swapping tokens (AMM) or depositing/borrowing assets (Lending), and record transactions and observations.

Apparatus/Software Used:

1. MetaMask Wallet
2. Brave Web Browser
3. Uniswap Testnet or Aave Testnet Website
4. Ethereum Sepolia Testnet Network

Theory/Concept:

Automated Market Maker (AMM)

AMM platforms like Uniswap allow users to **swap tokens** using **liquidity pools** instead of traditional order books.

- Liquidity providers deposit token pairs
- Users swap tokens directly against the pool
- Pools adjust prices using mathematical formulas such as **x × y = k**

Lending Protocol (Aave Model)

A lending protocol enables:

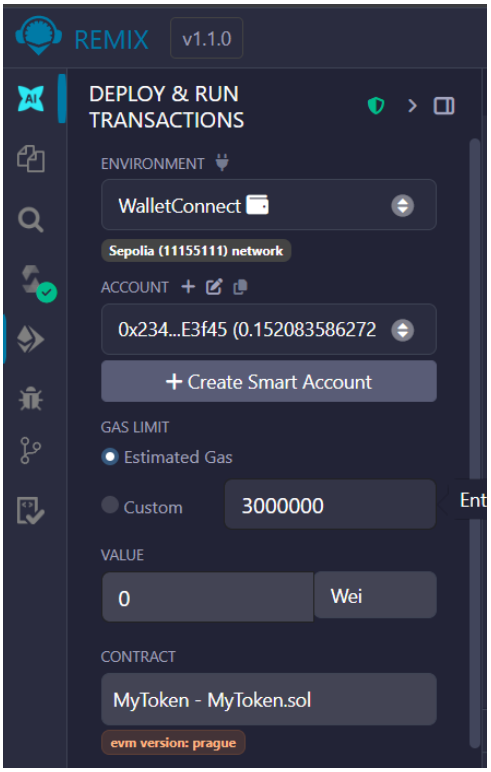
- Depositing assets → earn interest
- Borrowing assets → using collateral
- Interest rates determined by supply–demand
  On testnets, all functions work the same but with test tokens.

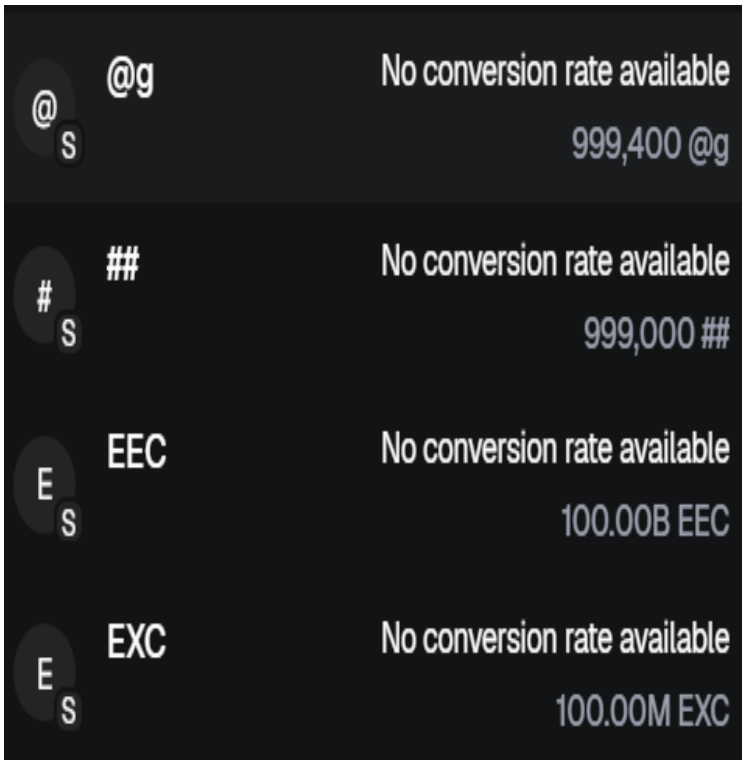
Procedure:

1. Open MetaMask → switch to **Sepolia Testnet**.
2. Visit **aave.com**, click **Enter App**.
3. Connect Wallet → MetaMask.
4. Enable **Testnet Mode** and choose **Ethereum Sepolia**.
5. Select **Deposit** and choose an asset (e.g., ETH).
6. Enter deposit amount → confirm.
7. Next, select **Borrow** to borrow a test asset (e.g., DAI).
8. Review gas fees and loan details → confirm.
9. Observe dashboard for health factor, interest rate, and collateral value.

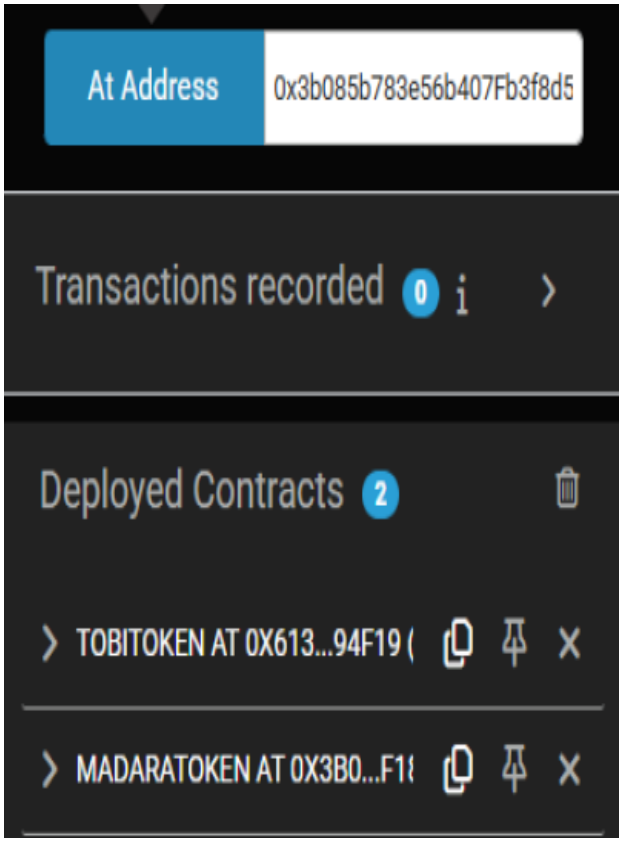
Solidity contract

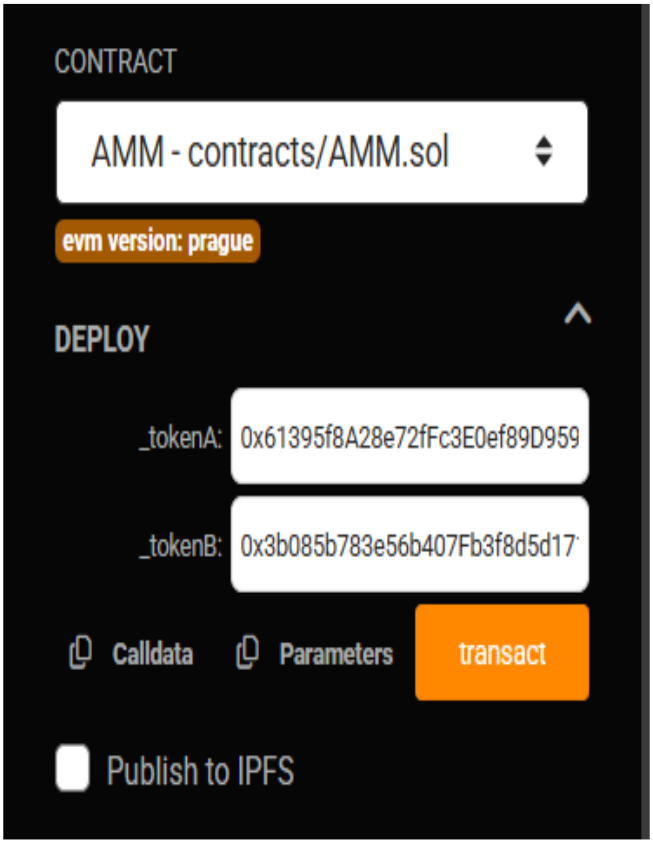
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract tobiToken is ERC20 {
    constructor(string memory name, string memory symbol) ERC20(name, symbol) {
        _mint(msg.sender, 1000000 * 10 ** decimals());
    }
}
```

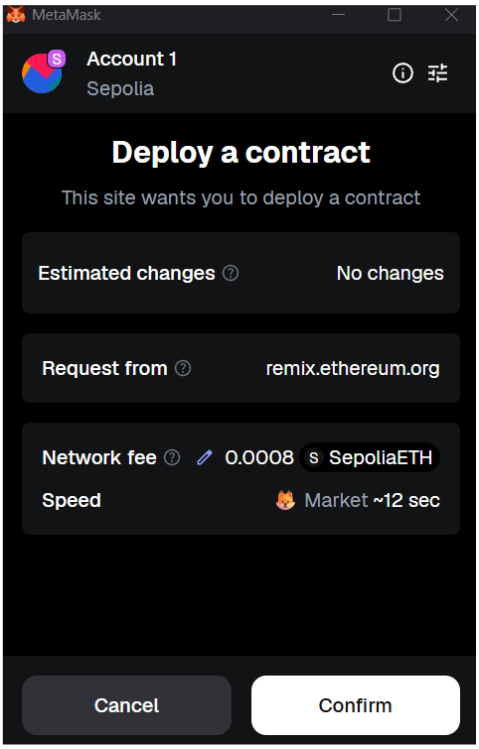


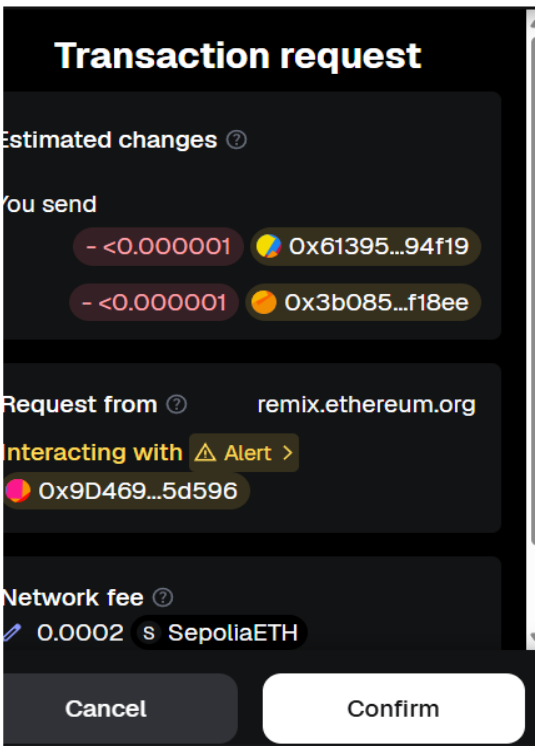


```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
contract AMM {
    IERC20 public tokenA;
    IERC20 public tokenB;
    uint public reserveA;
    uint public reserveB;
    constructor(IERC20 _tokenA, IERC20 _tokenB) {
        tokenA = _tokenA;
        tokenB = _tokenB;
    }
    function provideLiquidity(uint amountA, uint amountB) external {
        require(tokenA.transferFrom(msg.sender, address(this), amountA));
        require(tokenB.transferFrom(msg.sender, address(this), amountB));
        reserveA += amountA;
        reserveB += amountB;
    }
    function swapAforB(uint amountA) external {
        uint amountB = (amountA * reserveB) / (reserveA + amountA);
        require(tokenB.transfer(msg.sender, amountB));
        require(tokenA.transferFrom(msg.sender, address(this), amountA));
        reserveA += amountA;
        reserveB -= amountB;
    }
}

```









Observation

Action Performed	Result	Notes
Wallet Connected	Successful	MetaMask connected to Sepolia
AMM Swap / Lending Action	Completed	Transaction confirmed on testnet
Gas Fee Visibility	Yes	Gas fees and slippage clearly shown
Token Transfer/Lending	Successful	Test ETH swapped/deposited/borrowed

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: