Centurion
UNIVERSITY
Shaping Lives...
Empowering Communities...

School:..Campus:..

AcademicYear:......................SubjectName:...SubjectCode:.........................

Semester:...............Program:.......................................Branch:........................Specialization:..........................

Date: ....

Applied and Action Learning

(LearningbyDoingandDiscovery)

Name of the Experiement :

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

Algorithm Steps:

1. Create a new ERC-20 smart contract in Remix IDE.
2. Define key variables such as token name, symbol, total supply, and decimals.
3. Implement standard ERC-20 functions (e.g., balanceOf, transfer, approve, transferFrom).
4. Compile and deploy the contract on a testnet using MetaMask.
5. Verify and interact with the token through a blockchain explorer or wallet.

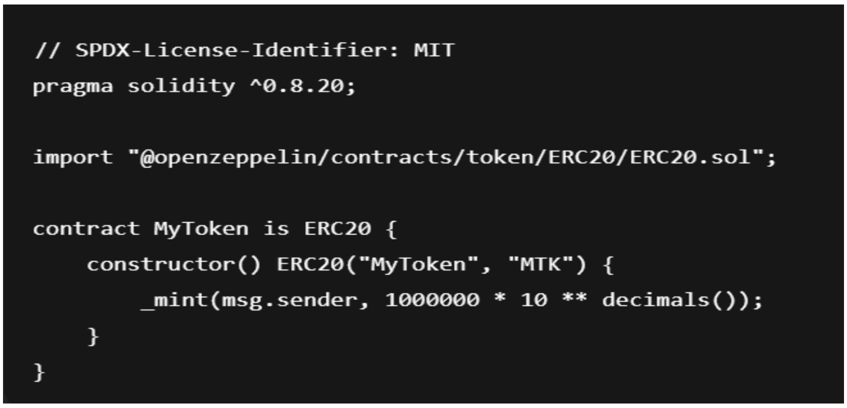
* Softwares used

1.RemixIDE
2.MetaMask
3.Etherscan
4.OpenZeppelinContracts
5.BraveWebBrowser

*_**Asapplicableaccordingtotheexperiment.
Twosheetsperexperiment(10-20)tobeused.**_

*TestingPhase:CompilationofCode(errordetection)

1. Open https://remix.ethereum.org in a web browser.
2. In the “contracts” folder, click the “+” icon and create a new file named MyToken.sol.
3. Paste the ERC-20 token code using the OpenZeppelin library.

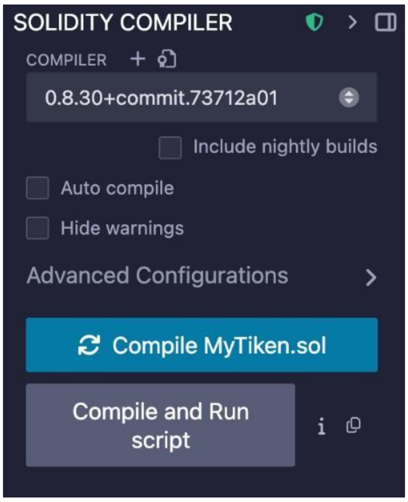


```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

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

contract MyToken is ERC20 {
    constructor() ERC20("MyToken", "MTK") {
        _mint(msg.sender, 1000000 * 10 ** decimals());
    }
}
```

4.Go to the Solidity Compiler tab, select version 0.8.20, and click Compile MyToken.sol.



SOLIDITY COMPILER

COMPILER

0.8.30+commit.73712a01

□ Include nightly builds

□ Auto compile

□ Hide warnings

5. Ensure there are no compilation errors before proceeding.

*Asapplicableaccordingtotheexperiment.
Twosheetsperexperiment(10-20)tobeused.

*TestingPhase:CompilationofCode(errordetection)

Deployment Phase

In Remix, go to **“Deploy & Run Transactions.”**
Select **Injected Provider – MetaMask** as the environment.
Approve the connection between MetaMask and Remix.
Click **Deploy**, confirm the transaction in MetaMask, and wait for confirmation.
Copy the **contract address** from Remix or MetaMask.
Open https://sepolia.etherscan.io and search your contract address to view token details.

Add Token to MetaMask

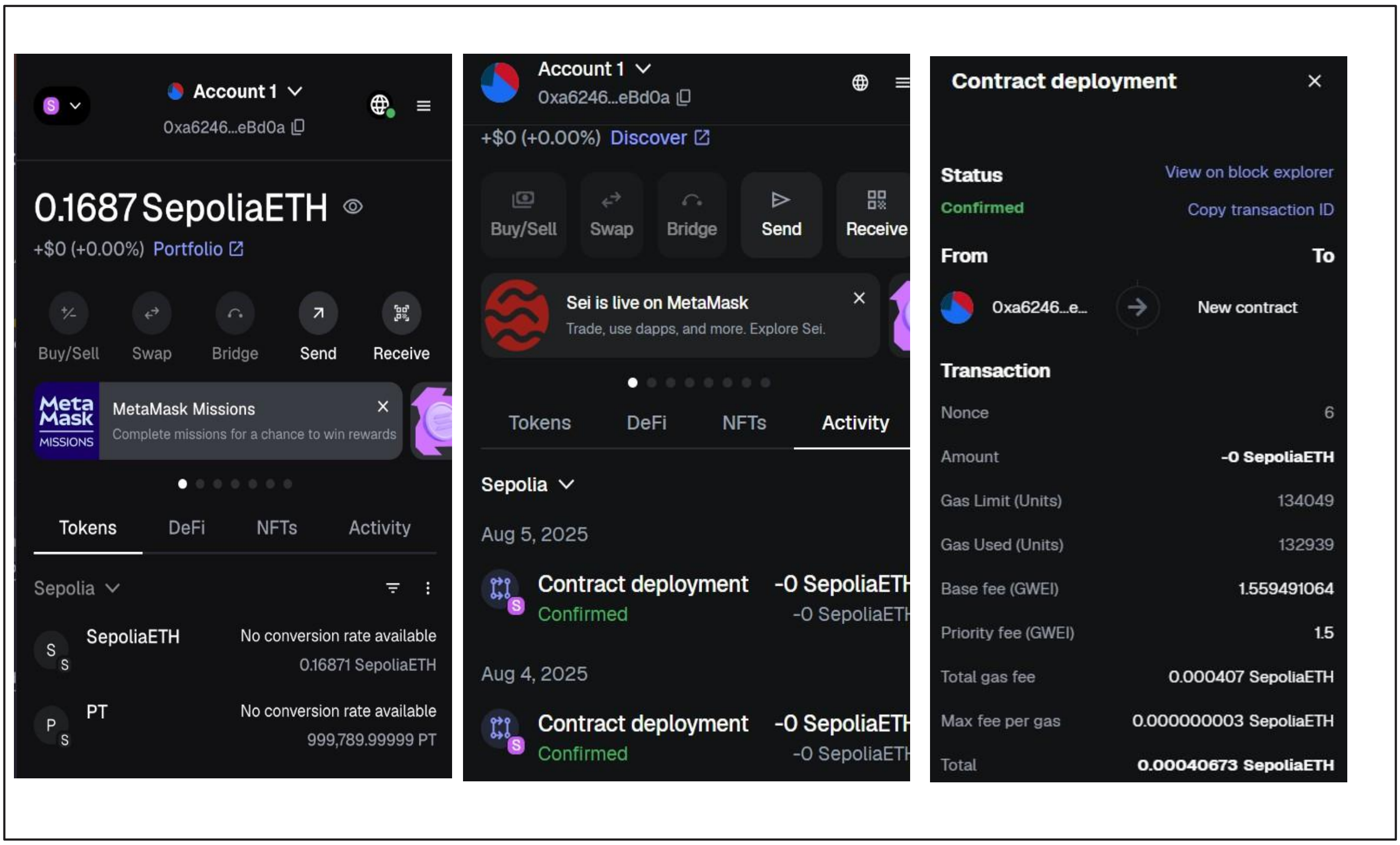
In MetaMask, click **Import Tokens**.
Paste your **contract address**.
MetaMask automatically fills in the **token symbol** and **decimals**.
Click **Add Custom Token → Import Tokens**.
Your token balance (e.g., 1,000,000 MTK) will appear in the wallet

Transfer Tokens to Another Wallet

In Remix, open your **Deployed Contracts** section.
Expand the contract and locate the **transfer(address, uint256)** function.
Enter the recipient’s wallet address and the number of tokens to send.
Approve the MetaMask transaction.
Verify the transfer on **Etherscan Testnet** using the transaction hash.

*Asapplicableaccordingtotheexperiment.
Twosheetsperexperiment(10-20)tobeused.

* Implementation Phase: Final Output (no error)



Account 1
0xa6246...eBd0a

0.1687 SepoliaETH
+$0 (+0.00%) Portfolio

MetaMask Missions
Complete missions for a chance to win rewards

SepoliaETH	No conversion rate available 0.16871 SepoliaETH
PT	No conversion rate available 999,789.99999 PT

Account 1
0xa6246...eBd0a
+$0 (+0.00%) Discover

Sei is live on MetaMask
Trade, use dapps, and more. Explore Sei.

Aug 5, 2025
Contract deployment — Confirmed — -0 SepoliaETH — -0 SepoliaETH

Aug 4, 2025
Contract deployment — Confirmed — -0 SepoliaETH — -0 SepoliaETH

Contract deployment

Status	View on block explorer
Confirmed	Copy transaction ID
From	To
0xa6246...e...	New contract

Transaction

Nonce	6
Amount	-0 SepoliaETH
Gas Limit (Units)	134049
Gas Used (Units)	132939
Base fee (GWEI)	1.559491064
Priority fee (GWEI)	1.5
Total gas fee	0.000407 SepoliaETH
Max fee per gas	0.000000003 SepoliaETH
Total	0.00040673 SepoliaETH

* Observations

-
- The ERC-20 token contract compiled without any syntax or logic errors.
- Deployment was completed using MetaMask with smooth testnet integration.
- Token appeared automatically in MetaMask after import.
- Transaction verified on Etherscan showed accurate transfer data.
- The OpenZeppelin library simplified token creation and ensured ERC-20 compliance.

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:

****Asapplicableaccordingtotheexperiment. Twosheetsperexperiment(10-20)tobeused.***