



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 Experiment : Token Launch – Deploying a Token Locally

### \* Coding Phase: Pseudo Code / Flow Chart / Algorithm

#### ALGORITHM:

- 1.Start
- 2.Open your remix IDE and create a new file and name it as per as your choice.
- 3.Write the solidity code to for creating a token
- 4.Now compile that file
- 5.After compiling go to deploy and transaction, select injected provider metamask, then write your token name and symbol in deploy and deploy the contract.
- 6.Confirm the transaction.
- 7.Now go to your wallet to import token.
- 8.End

### \* Software used

- 1.Remix IDE
- 2.Metamask
- 3.OpenZeppelin Contracts
- 4.Etherscan

## \* Testing Phase: Compilation of Code (error detection)

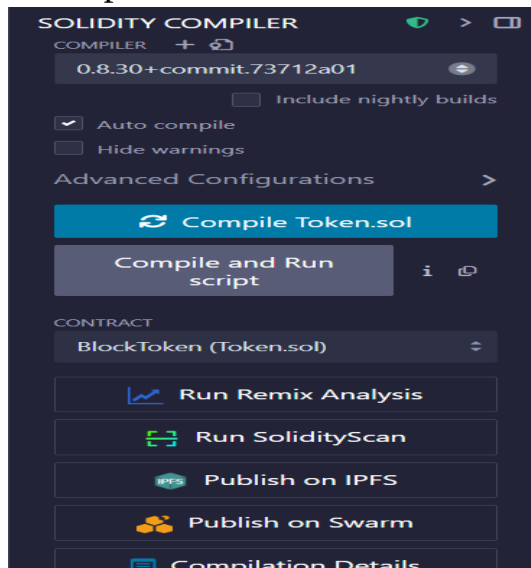
First open your remix IDE and create a new file in contracts named as Token.sol. Then write the contract for deploying the token

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

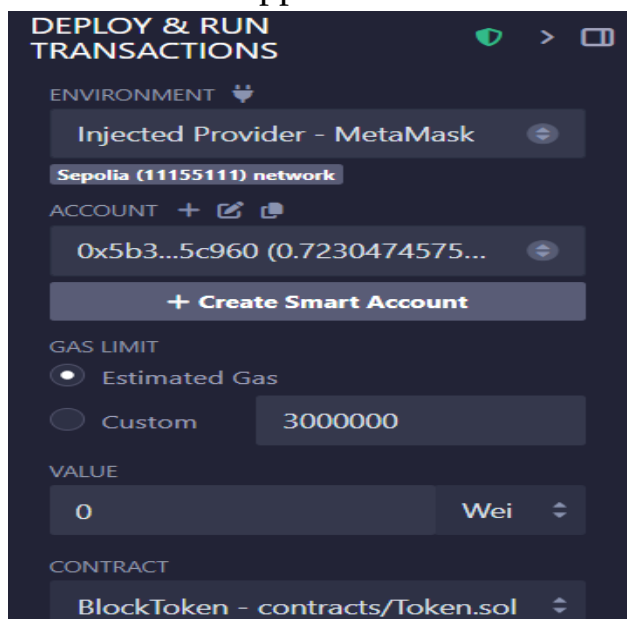
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
contract BlockToken is ERC20 {
    constructor(string memory name, string memory symbol) ERC20(name, symbol){
        _mint(msg.sender, 1000000 * 10 ** decimals());
    }
}
```

This code creates an ERC-20 token with customizable name and symbol. It also mints 1 million tokens to the deployer's wallet.

Now go to solidity compiler select compiler version 0.8.30 and then click on "Compile Token.sol".

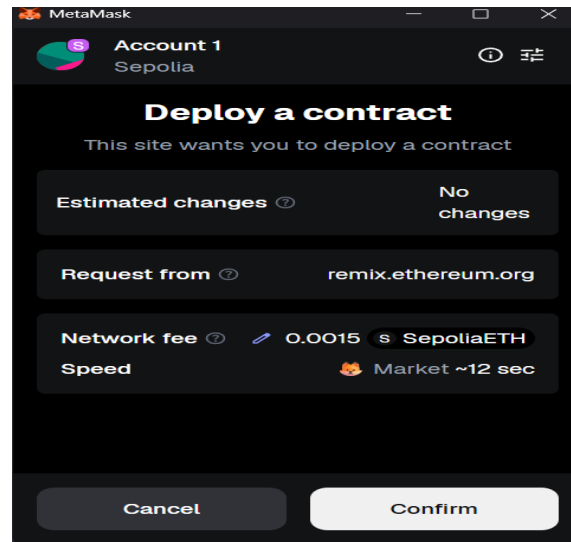
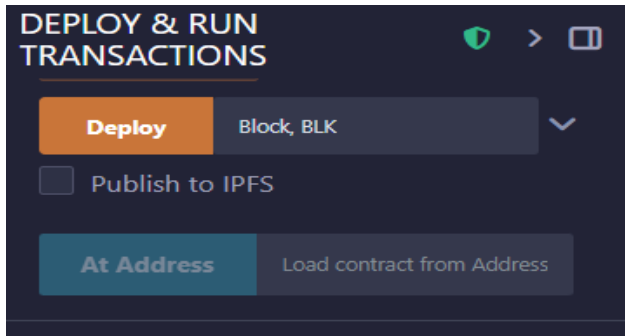


Now go to deploy and transaction select "injected provider metamask" as your environment. Approve the connection

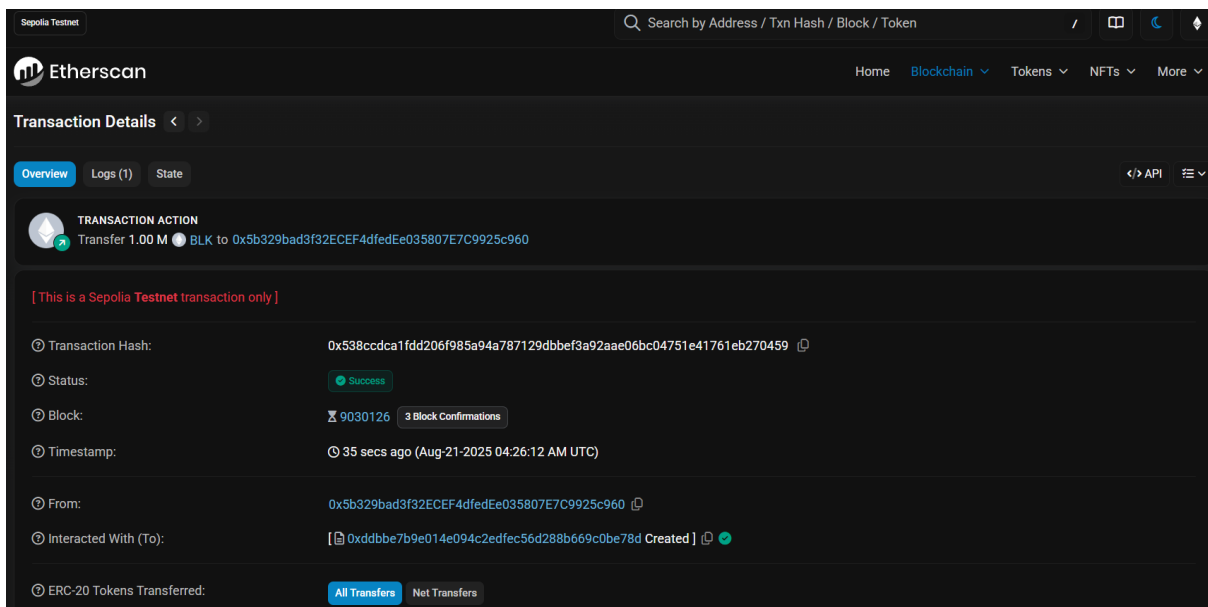


## \* Testing Phase: Compilation of Code (error detection)

Write your token name and symbol in the constructor parameter (e.g. BlockToken, BLK) and then deploy the contract. It will open a pop-up to deploy the contract click on confirm.



After deploying copy the address and paste it in etherscan to check transaction details of your token.



## \* Testing Phase: Compilation of Code (error detection)

Click on the token symbol written on transaction details page it will open to your created token where you will get your token contract address, copy that address.

The screenshot shows the Etherscan Sepolia Testnet interface. The top navigation bar includes 'Home', 'Blockchain', 'Tokens', 'NFTs', and 'More'. The main content area is titled 'Token Block (BLK)' and displays an 'Overview' section with the following data:

- MAX TOTAL SUPPLY: 1,000,000 BLK
- HOLDERS: 1
- TOTAL TRANSFERS: 1

The 'Market' section shows 'ONCHAIN MARKET CAP' and 'CIRCULATING SUPPLY MARKET CAP' as '-'. The 'Other Info' section displays the 'TOKEN CONTRACT (WITH 18 DECIMALS)' as '0xddbb7b9e014e094c2edfec56d288b669c0be78d'. Below this, there is an advertisement for 'Advertise your brand here!' with logos for BaseScan, Etherscan, BscScan, PolygonScan, and SolScan. The 'Transfers' tab is selected, showing a table with one transaction:

| Transaction Hash | Method     | Block   | Age       | From                   | To                     | Amount    |
|------------------|------------|---------|-----------|------------------------|------------------------|-----------|
| 0x538ccdca1fd... | 0x60806040 | 9030126 | 1 min ago | 0x00000000...000000000 | 0x5b329bad...C9925c960 | 1,000,000 |

Now import your token in your metamask wallet. Open your wallet go to tokens click on the three dots and then click on import tokens. Select the network sepolia and then paste your token contract address and click next.

The screenshot shows the Metamask 'Tokens' screen. The network is set to 'Sepolia'. There are two tokens listed: 'SepoliaETH' and 'BLK'. A context menu is open over the 'BLK' token, showing the options '+ Import tokens' and 'Refresh list'.

The screenshot shows the Metamask 'Import tokens' screen. The 'Custom token' section is active. The form is filled with the following information:

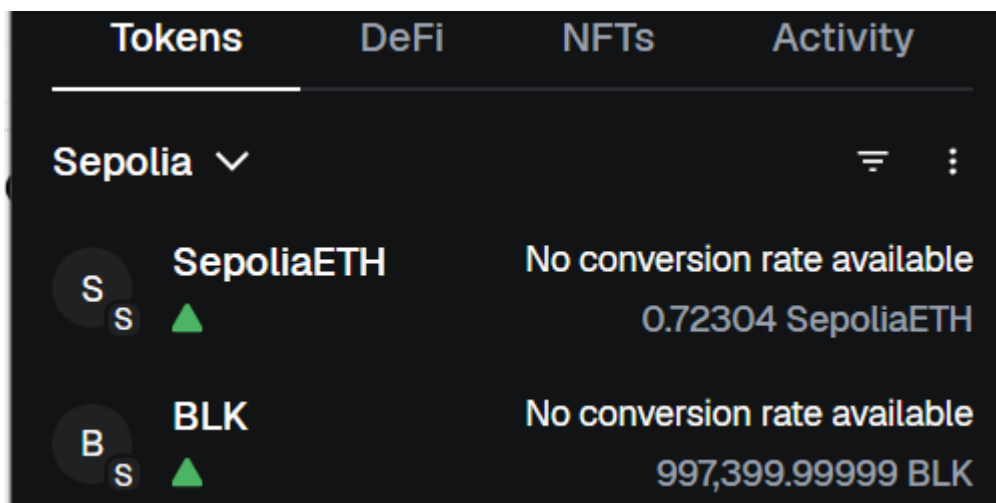
- Network: Sepolia
- Token contract address: 4E094C2eDfEc56d288B669c0bE78
- Token symbol: BLK
- Token decimal: 18

The 'Next' button is visible at the bottom of the form.

## \* Implementation Phase: Final Output (no error)

Applied and Action Learning

Your token is successfully launched in your wallet.



## \* Observations

1. Successfully deployed an ERC-20 compatible token smart contract on a local blockchain environment to simulate token creation.
2. Verified token functionalities like total supply, balance transfer, and allowance management in a controlled local setup before mainnet/testnet deployment.

## ASSESSMENT

| Rubrics  | Full Mark | Marks Obtained | Remarks |
|--|-----------|----------------|---------|
| Concept  | 10        |                |         |
| Planning and Execution/<br>Practical Simulation/ Programming | 10        |                |         |
| Result and Interpretation                                    | 10        |                |         |
| Record of Applied and Action Learning                        | 10        |                |         |
| Viva   | 10        |                |         |
| <b>Total</b>   | <b>50</b> |                |         |

**Signature of the Student:**

Name :

Regn. No. :

**Signature of the Faculty:**

Page No.....

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