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

Coding Phase : Pseudo Code/Flow Chart/Algorithm

- Open Remix IDE.
- Import the ERC20 contract from OpenZeppelin.
- Create a constructor that sets the token name, symbol, and mints initial supply.
- Compile the contract with Solidity ^0.8.20.
- Deploy to local blockchain using account 1.
- Use functions `name()`, `symbol()`, and `totalSupply()` to verify token.

Apparatus/Software Used:

- OS: Windows or others.
- Remix IDE.
- Wallet: MetaMask.
- Library: OpenZeppelin ERC20

Testing Phase:

- Call name() → returns **"HatiSwornaTokens"**
- Call symbol() → returns **"HST"**
- Call totalSupply() → returns **100000000**
- Call balanceOf(owner) → shows total supply in deployer's account.

Implementation Phase: Final Output (no error)

Step 1: Open Remix IDE.

- Open Browser/Brave.
- Search Remix IDE.

Step 2: Write Smart Contract.

- Create a new file in inside of contract using .sol.
- Write code:

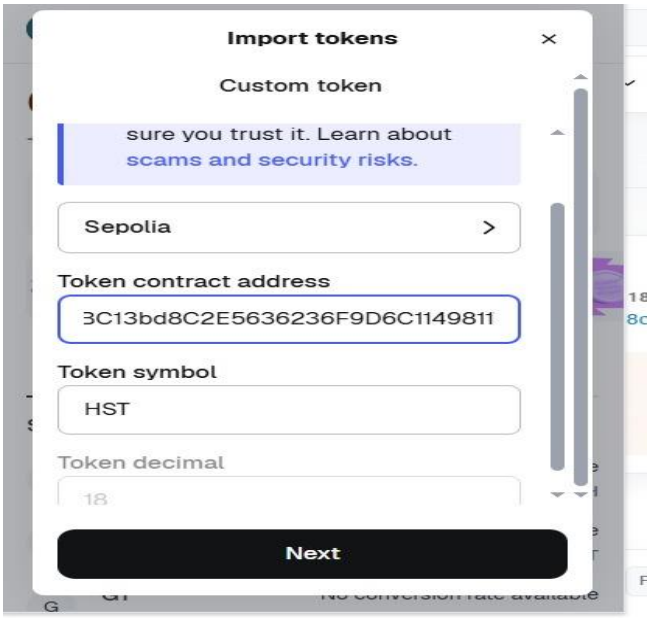
```
// SPDX-License-Identifier: MIT


pragma solidity ^0.8.0;

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

contract MyToken is ERC20 {

constructor(string memory name,string memory symbol)    infinite gas 710800 gas
ERC20(name,symbol){
_mint(msg.sender,1000000 * 10 ** decimals());
}
}
```

Step 3: Code Compile and Deploy .

- Click the Solidity compiler and compile this file.
- Click Deploy & run transaction Option .
- Deploy the write of some string name and symbol.

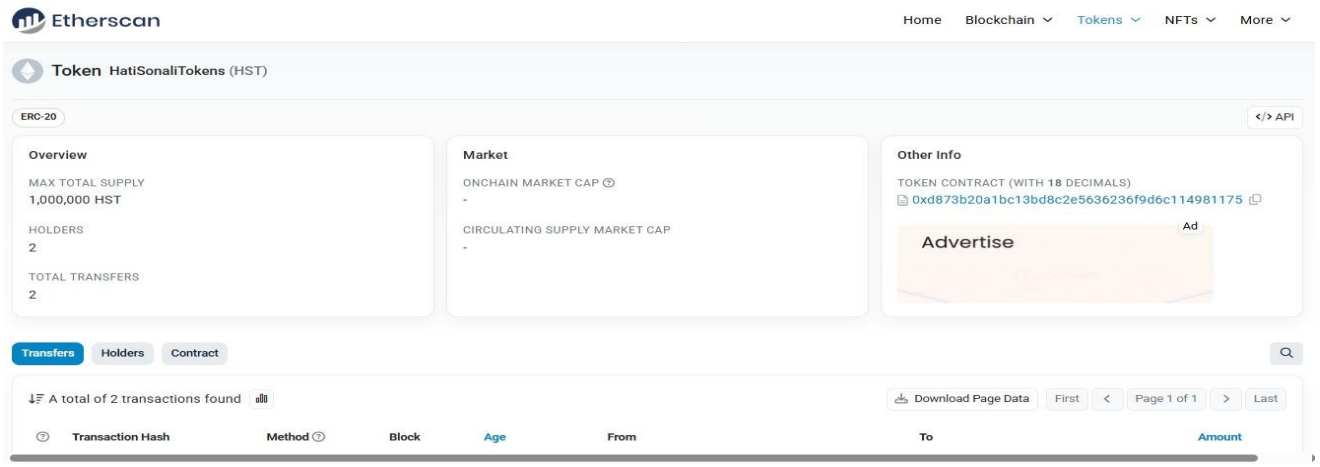
Step 4: Add Token to MetaMask.

- In MetaMask, click **"Import Tokens"**.
- Enter your deployed contract address.
- Your token should appear in the wallet.



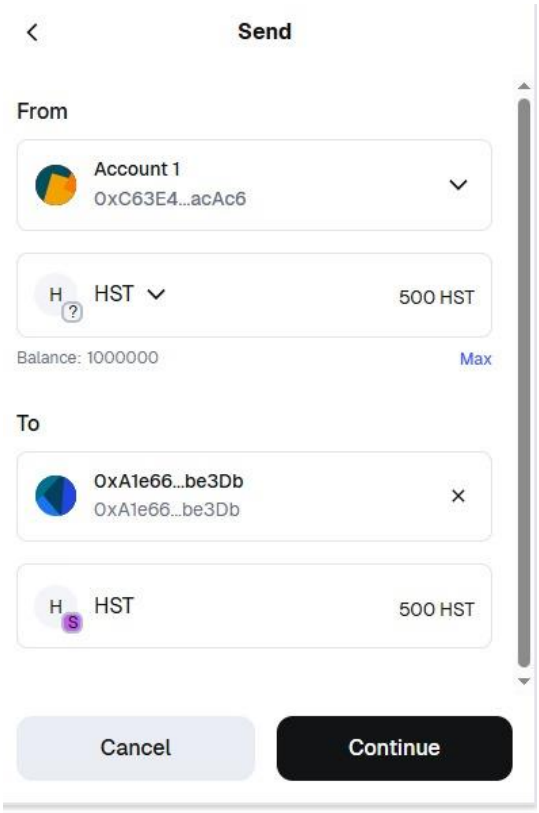
Step 5:Check Token Details.

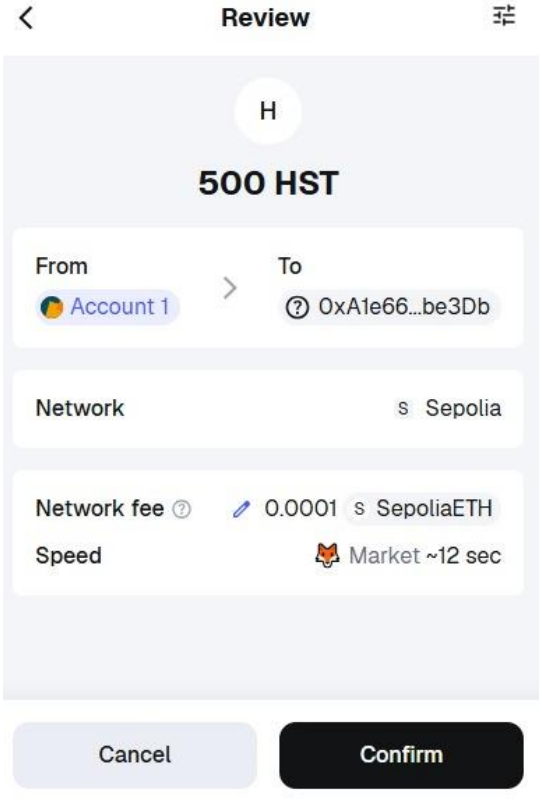
- name() → should return "HatiSwornaTokens"
- symbol() → should return "HST"
- totalSupply() → should return 10000000.



Step 6: Transfer Token:

- Use transfer(receiver_address, amount) to send tokens to another account.
- Then check balance Of (receiver_address) to confirm transfer.





Observations

- The OpenZeppelin library makes ERC20 token creation quick and secure.
- Local deployment requires no real ETH.
- Token functions behave exactly the same as on testnet/mainnet.

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:

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