Vidyalankar Institute of Technology
Accredited A+ by NAAC

DEPARTMENT OF COMPUTER ENGINEERING

Experiment No. 04

Semester	B.E. Semester VII – Computer Engineering
Subject	Blockchain Lab (CSDL7022)
Subject Professor In-charge	Prof. Swapnil S. Sonawane
Academic Year	2024-25

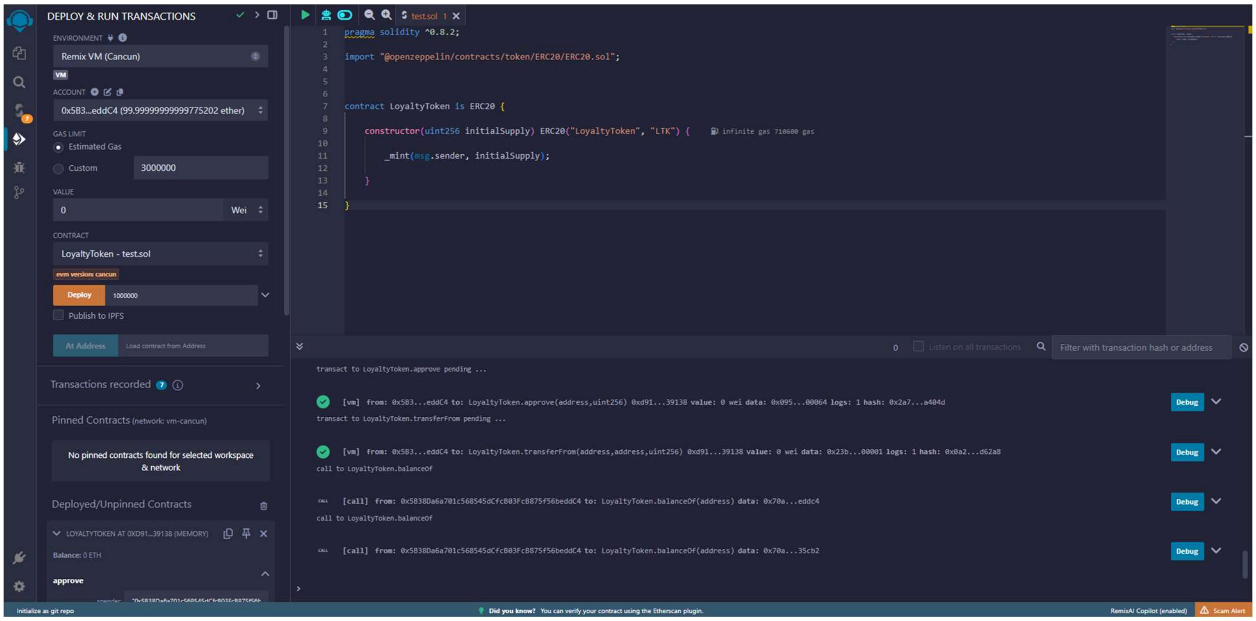
Student Name	Deep Salunkhe
Roll Number	21102A0014

Title: Create a simple token to represent points in a loyalty rewards program.

Program Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;

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



contract LoyaltyToken is ERC20 {

    constructor(uint256 initialSupply) ERC20("LoyaltyToken", "LTK") {

        _mint(msg.sender, initialSupply);

    }

}
```

Output:



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Remix VM (Cancun)

ACCOUNT: 0x5B3...eddC4 (99.99999999999775202 ether)

GAS LIMIT: ● Estimated Gas ○ Custom 3000000

VALUE: 0 Wei

CONTRACT: LoyaltyToken - test.sol

evm version: cancun

Deploy 1000000

□ Publish to IPFS

At Address — Load contract from Address

Transactions recorded 7

Pinned Contracts (network: vm-cancun)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

LOYALTYTOKEN AT 0XD91...39138 (MEMORY)

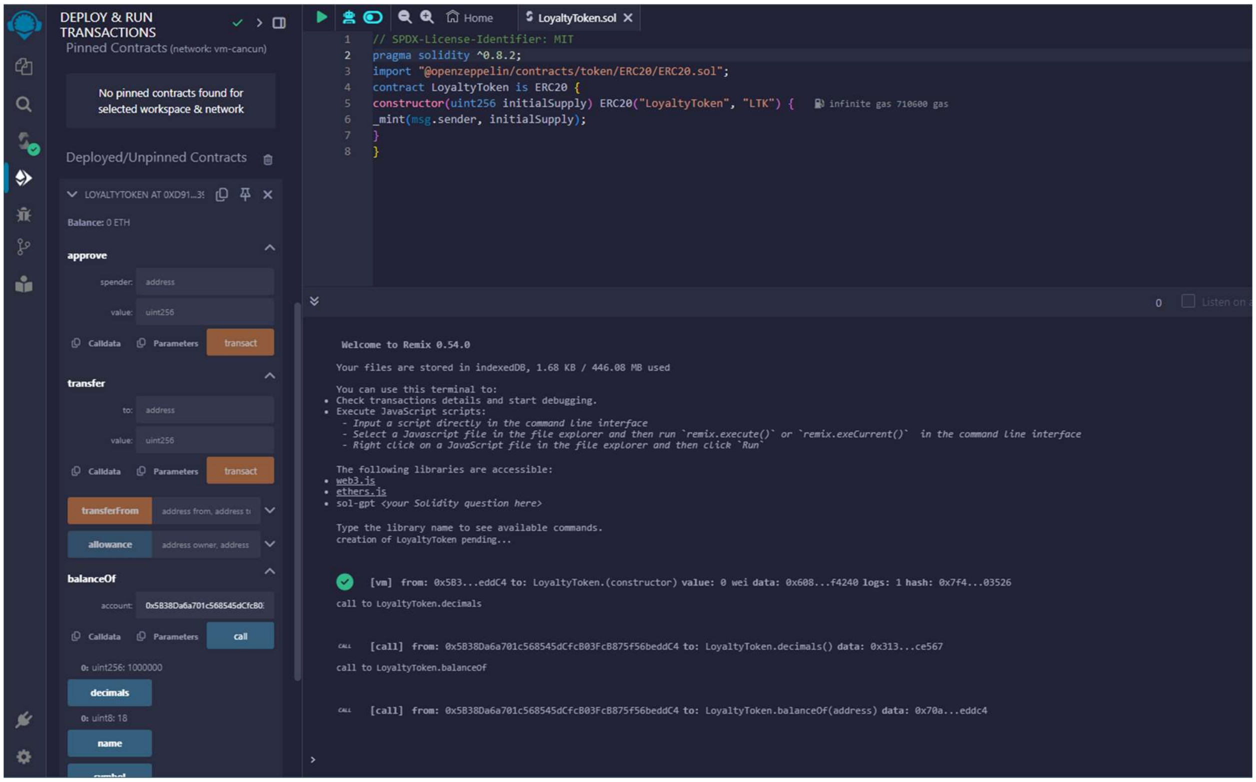
Balance: 0 ETH

approve

```
pragma solidity ^0.8.2;

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



contract LoyaltyToken is ERC20 {

    constructor(uint256 initialSupply) ERC20("LoyaltyToken", "LTK") {    infinite gas 710600 gas

        _mint(msg.sender, initialSupply);

    }

}
```

```
transact to LoyaltyToken.approve pending ...

[vm]  from: 0x5B3...eddC4 to: LoyaltyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...00064 logs: 1 hash: 0x2a7...a404d
transact to LoyaltyToken.transferFrom pending ...

[vm]  from: 0x5B3...eddC4 to: LoyaltyToken.transferFrom(address,address,uint256) 0xd91...39138 value: 0 wei data: 0x23b...00001 logs: 1 hash: 0x0a2...d62a8
call to LoyaltyToken.balanceOf

CALL  [call]  from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: LoyaltyToken.balanceOf(address) data: 0x70a...eddc4
call to LoyaltyToken.balanceOf

CALL  [call]  from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: LoyaltyToken.balanceOf(address) data: 0x70a...35cb2
```

Did you know? You can verify your contract using the Etherscan plugin.



DEPLOY & RUN TRANSACTIONS

Pinned Contracts (network: vm-cancun)

No pinned contracts found for selected workspace & network

Deployed/Unpinned Contracts

LOYALTYTOKEN AT 0XD91...3

Balance: 0 ETH

approve — spender: address, value: uint256 — transact

transfer — to: address, value: uint256 — transact

transferFrom — address from, address t

allowance — address owner, address

balanceOf — account: 0x5B38Da6a701c568545dCfcB0 — call

0: uint256: 1000000

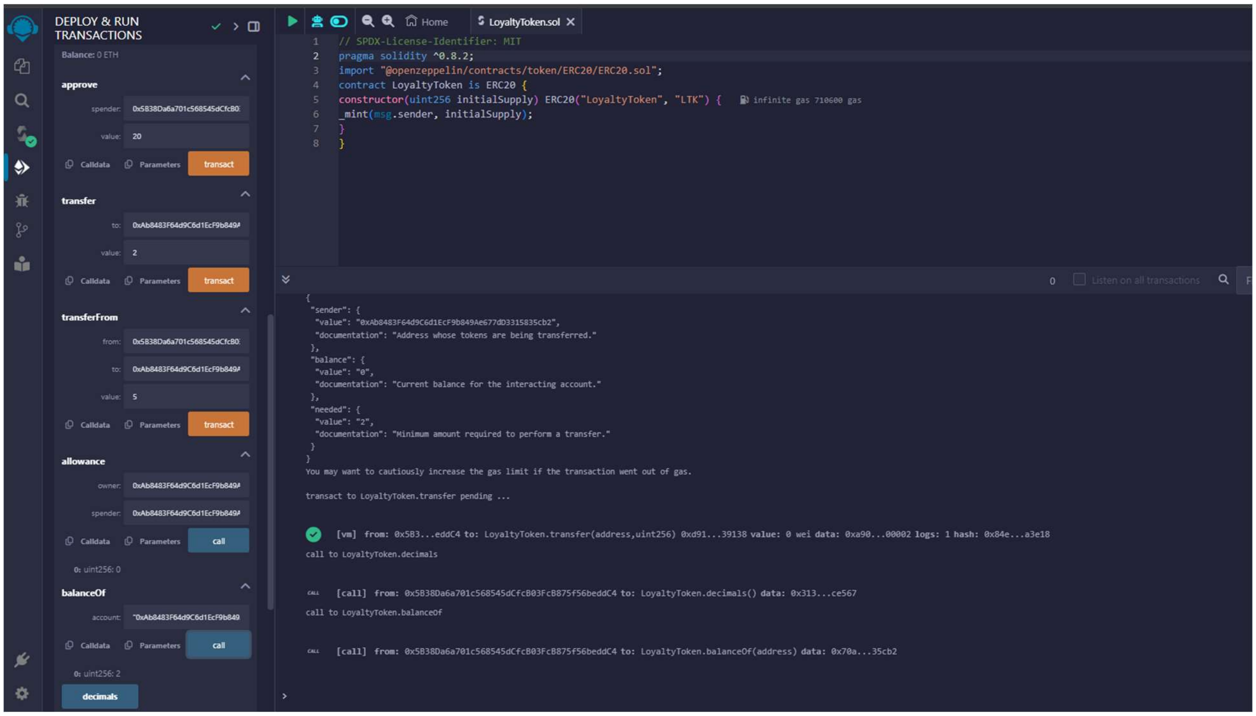
decimals

0: uint8: 18

name

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
contract LoyaltyToken is ERC20 {
constructor(uint256 initialSupply) ERC20("LoyaltyToken", "LTK") {    infinite gas 710600 gas
_mint(msg.sender, initialSupply);
}
}
```

```
 Welcome to Remix 0.54.0

Your files are stored in indexedDB, 1.68 KB / 446.08 MB used

You can use this terminal to:
• Check transactions details and start debugging.
• Execute JavaScript scripts:
  - Input a script directly in the command line interface
  - Select a Javascript file in the file explorer and then run `remix.execute()` or `remix.exeCurrent()`  in the command line interface
  - Right click on a JavaScript file in the file explorer and then click `Run`

The following libraries are accessible:
• web3.js
• ethers.js
• sol-gpt <your Solidity question here>

Type the library name to see available commands.
creation of LoyaltyToken pending...


[vm]  from: 0x5B3...eddC4 to: LoyaltyToken.(constructor) value: 0 wei data: 0x608...f4240 logs: 1 hash: 0x7f4...03526
call to LoyaltyToken.decimals


CALL  [call]  from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: LoyaltyToken.decimals() data: 0x313...ce567
call to LoyaltyToken.balanceOf


CALL  [call]  from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: LoyaltyToken.balanceOf(address) data: 0x70a...eddc4
```



Balance: 0 ETH

approve
spender: 0x5B38Da6a701c568545dCfcB0:
value: 20

transfer
to: 0xAb8483F64d9C6d1EcF9b849A
value: 2

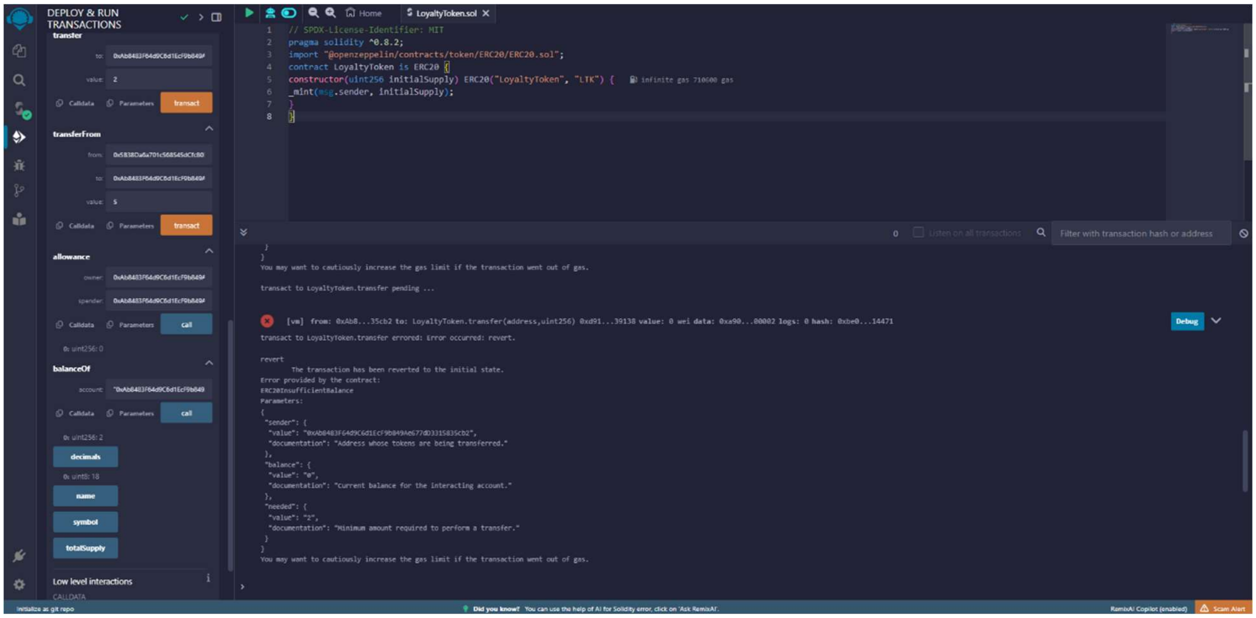
transferFrom
from: 0x5B38Da6a701c568545dCfcB0:
to: 0xAb8483F64d9C6d1EcF9b849A
value: 5

allowance
owner: 0xAb8483F64d9C6d1EcF9b849A
spender: 0xAb8483F64d9C6d1EcF9b849A
0: uint256: 0

balanceOf
account: "0xAb8483F64d9C6d1EcF9b849
0: uint256: 2

```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
contract LoyaltyToken is ERC20 {
constructor(uint256 initialSupply) ERC20("LoyaltyToken", "LTK") {     infinite gas 710600 gas
_mint(msg.sender, initialSupply);
}
}
```

```
{
 "sender": {
  "value": "0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2",
  "documentation": "Address whose tokens are being transferred."
 },
 "balance": {
  "value": "0",
  "documentation": "Current balance for the interacting account."
 },
 "needed": {
  "value": "2",
  "documentation": "Minimum amount required to perform a transfer."
 }
}
You may want to cautiously increase the gas limit if the transaction went out of gas.

transact to LoyaltyToken.transfer pending ...

[vm] from: 0x5B3...eddC4 to: LoyaltyToken.transfer(address,uint256) 0xd91...39138 value: 0 wei data: 0xa90...00002 logs: 1 hash: 0x84e...a3e18
call to LoyaltyToken.decimals

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: LoyaltyToken.decimals() data: 0x313...ce567
call to LoyaltyToken.balanceOf

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: LoyaltyToken.balanceOf(address) data: 0x70a...35cb2
```



transfer
to: 0xAb8483F64d9C6d1EcF9b849A
value: 2

transferFrom
from: 0x5B38Da6a701c568545dCfcB0:
to: 0xAb8483F64d9C6d1EcF9b849A
value: 5

allowance
owner: 0xAb8483F64d9C6d1EcF9b849A
spender: 0xAb8483F64d9C6d1EcF9b849A
0: uint256: 0

balanceOf
account: "0xAb8483F64d9C6d1EcF9b849
0: uint256: 2

decimals
0: uint8: 18

```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
contract LoyaltyToken is ERC20 {
constructor(uint256 initialSupply) ERC20("LoyaltyToken", "LTK") {     infinite gas 710600 gas
_mint(msg.sender, initialSupply);
}
}
```

```
 }
}
You may want to cautiously increase the gas limit if the transaction went out of gas.

transact to LoyaltyToken.transfer pending ...

[vm] from: 0xAb8...35cb2 to: LoyaltyToken.transfer(address,uint256) 0xd91...39138 value: 0 wei data: 0xa90...00002 logs: 0 hash: 0xbe0...14471
transact to LoyaltyToken.transfer errored: Error occurred: revert.

revert
        The transaction has been reverted to the initial state.
Error provided by the contract:
ERC20InsufficientBalance
Parameters:
{
 "sender": {
  "value": "0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2",
  "documentation": "Address whose tokens are being transferred."
 },
 "balance": {
  "value": "0",
  "documentation": "Current balance for the interacting account."
 },
 "needed": {
  "value": "2",
  "documentation": "Minimum amount required to perform a transfer."
 }
}
You may want to cautiously increase the gas limit if the transaction went out of gas.
```

Did you know? You can use the help of AI for Solidity error, click on 'Ask RemixAI'.