

# MYTOKEN – ERC-20 Token Project Report

**Submitted by:** Ashrita Gogula

**Contract Address:** 0xd9145CCE52D386f254917e481eB44e9943F39138

## 1. Project Overview

This project demonstrates the creation of a fully functional **ERC-20 token** on the Ethereum blockchain using **Solidity** and **Remix IDE**.

The project includes token metadata, total supply creation, transfer functions, allowance mechanism, and event logging.

This token works like any real cryptocurrency built on ERC-20 standards.

## 2. Token Details

Property	Value
<b>Token Name</b>	MyToken
<b>Symbol</b>	MTK
<b>Decimals</b>	18
<b>Total Supply</b>	1,000,000 MTK (with 18 decimals)
<b>Contract Address</b>	0xd9145CCE52D386f254917e481eB44e9943F39138

## 3. Development Environment

- **IDE:** Remix Ethereum IDE
- **Language:** Solidity 0.8.x
- **Network:** Remix VM (Prague)
- **Contract File:** MyToken.sol

Steps followed:

1. Opened Remix IDE
2. Created a new Solidity contract file
3. Wrote the ERC-20 token code
4. Compiled contract → No errors
5. Deployed contract on Remix VM
6. Interacted with contract functions using Remix UI

## 4. Features Implemented

### ✓ ERC-20 Token Metadata

- Token name
- Symbol
- Decimals
- Total supply

### ✓ Balance Tracking

- Balance of any address

### ✓ Transfer Function

- Transfer tokens from one account to another

### ✓ Allowance Mechanism

- Approve a spender
- Check allowance
- Spend via transferFrom

### ✓ Events

- **Transfer** event
- **Approval** event

These events confirm on-chain activity and allow tracking in logs.

---

## 5. Screenshots Included

### 5.1 Compilation Success

The screenshot shows the REMIX IDE interface with the following details:

- FILE EXPLORER**: Shows the project structure with files like states, vm-prague, artifacts, build-info, MyToken\_metadata.json, MyToken.json, contracts (1\_Storage.sol, 2\_Owner.sol, 3\_Ballot.sol), and MyToken.sol.
- CODE EDITOR**: Displays the Solidity code for MyToken.sol. The code defines a token contract with name "MyToken", symbol "MTK", 18 decimals, and a total supply of 1 million units. It includes functions for balanceOf, allowance, transfer, and approval, along with their respective event emissions.
- CONTRACTS**: Shows the compiled contract details: contracts/MyToken.sol at line 15:16.
- TRANSACTIONS**: A transaction log entry: [vm] from: 0x5B3...eddC4 to: MyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...c0000 logs: 1 hash: 0xd60...dab52 call to MyToken.allowance.

The screenshot shows the REMIX IDE interface with the following details:

- SOLIDITY COMPILER**: Set to version 0.8.0+commit.73712a01. Options include "Include nightly builds", "Auto compile", and "Hide warnings".
- CODE EDITOR**: Displays the same Solidity code for MyToken.sol as in the previous screenshot.
- TRANSACTIONS**: A transaction log entry: [vm] from: 0x5B3...eddC4 to: MyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...c0000 logs: 1 hash: 0xd60...dab52 call to MyToken.allowance.

## 5.2 Deployment of Token Contract

### MyToken Contract Details

- Token Name:** MyToken
- Symbol:** MTK
- Decimals:** 18
- Total Supply:** 1,000,000 MTK (1 million)
- Contract Address:**  
0xd9145CCE52D386f254917e481eB44e9943F39138

**REMX** 1.3.0

### DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Remix VM (Prague) | Reset State

VM

ACCOUNT: 0x5B3...eddC4 (99.999999999975904 ETH)

+ Authorize Delegation

GAS LIMIT: Estimated Gas (3000000)

CONTRACT: MyToken - contracts/MyToken.sol

evm version: prague

DEPLOY: \_initialSupply: uint256

Calldata | Parameters | **transact**

At Address | Load contract from Address

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract MyToken {
    string public name = "MyToken";
    string public symbol = "MTK";
    uint8 public decimals = 18;
    uint256 public totalSupply;

    mapping(address => uint256) public balanceOf;
    mapping(address => mapping(address => uint256)) public allowance;

    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);

    constructor(uint256 _initialSupply) {
        totalSupply = _initialSupply;
        balanceOf[msg.sender] = _initialSupply;
        emit Transfer(address(0), msg.sender, _initialSupply);
    }

    function approve(address spender, uint256 value) external returns (bool) {
        allowance[msg.sender][spender] = value;
        emit Approval(msg.sender, spender, value);
        return true;
    }

    function transfer(address to, uint256 value) external returns (bool) {
        require(balanceOf[msg.sender] >= value, "Insufficient balance");
        balanceOf[msg.sender] -= value;
        balanceOf[to] += value;
        emit Transfer(msg.sender, to, value);
        return true;
    }

    function transferFrom(address from, address to, uint256 value) external returns (bool) {
        require(allowance[from][msg.sender] >= value, "Insufficient allowance");
        allowance[from][msg.sender] -= value;
        balanceOf[from] -= value;
        balanceOf[to] += value;
        emit Transfer(from, to, value);
        return true;
    }
}
```

Explain contract

[vm] from: 0x5B3...eddC4 to: MyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...c0000 logs: 1 hash: 0xd60...dab52 call to MyToken.allowance

[call] from: 0x5B38D... to: MyToken.allowance(address,address) data: 0xdd6...35cb2 transact to MyToken.transferFrom pending ...

DEPLOY & RUN TRANSACTIONS

Transactions recorded: 6

Deployed Contracts: MYTOKEN at 0xD91...39138 (Balance: 0 ETH)

APPROVE: \_spender: 0xAb8483F64d9C6d1EcF9b849Ae6; \_value: 30000000000000000000000000000000

TRANSFER: \_to: 0xAb8483F64d9C6d1EcF9b849Ae6; \_value: 10000000000000000000000000000000

TRANSFERFROM: \_from: 0x5B38D...; \_to: 0x4B209938...; \_value: 10000000000000000000000000000000

Compile | Home | MyToken.sol | Explain contract

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract MyToken {
    string public name = "MyToken";
    string public symbol = "MTK";
    uint8 public decimals = 18;
    uint256 public totalSupply;

    mapping(address => uint256) public balanceOf;
    mapping(address => mapping(address => uint256)) public allowance;

    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);

    constructor(uint256 _initialSupply) {
        totalSupply = _initialSupply;
        balanceOf[msg.sender] = _initialSupply;
        emit Transfer(address(0), msg.sender, _initialSupply);
    }

    function approve(address spender, uint256 value) external returns (bool) {
        allowance[msg.sender][spender] = value;
        emit Approval(msg.sender, spender, value);
        return true;
    }

    function transfer(address to, uint256 value) external returns (bool) {
        require(balanceOf[msg.sender] >= value, "Insufficient balance");
        balanceOf[msg.sender] -= value;
        balanceOf[to] += value;
        emit Transfer(msg.sender, to, value);
        return true;
    }

    function transferFrom(address from, address to, uint256 value) external returns (bool) {
        require(allowance[from][msg.sender] >= value, "Insufficient allowance");
        allowance[from][msg.sender] -= value;
        balanceOf[from] -= value;
        balanceOf[to] += value;
        emit Transfer(from, to, value);
        return true;
    }
}
```

0 Listen on all transactions | Filter with transaction hash

[vm] from: 0x5B3...eddC4 to: MyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...c0000 logs: 1 hash: 0xd60...dab52 call to MyToken.allowance

[call] from: 0x5B38D... to: MyToken.allowance(address,address) data: 0xdd6...35cb2 transact to MyToken.transferFrom pending ...

REMX 1.3.0

default\_workspace

DEPLOY & RUN TRANSACTIONS

BALANCEOF : 0x5838Da6a701c568545dCfcB03Fd

Calldata Parameters call

0: uint256: 999998000000000000000000  
000

decimals

0: uint8: 18

name

0: string: MyToken

symbol

0: string: MTK

totalSupply

0: uint256: 1000000000000000000000000  
0000

Low level interactions

CALldata

Transact

Compile ▾ Home MyToken.sol X

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract MyToken {
    string public name = "MyToken";
    string public symbol = "MTK";
    uint8 public decimals = 18;
    uint256 public totalSupply;

    mapping(address => uint256) public balanceOf;
    mapping(address => mapping(address => uint256)) public allowance;

    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);

    constructor(uint256 _initialSupply) {
        totalSupply = _initialSupply;
        balanceOf[msg.sender] = _initialSupply;
        emit Transfer(0, msg.sender, _initialSupply);
    }
}
```

Explain contract

0 Listen on all transactions

[vm] from: 0x583...eddC4 to: MyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...c  
call to MyToken.allowance

CALL [call] from: 0x5838Da6a701c568545dCfcB03FcB875f56beddC4 to: MyToken.allowance(address,address) data: 0xc  
transact to MyToken.transferFrom pending ...

## 💡 5.3 Token Information (name, symbol, decimals, totalSupply)

## decimals → 18

**name → MyToken**

## **symbol → MTK**

**totalSupply** → 1000000000000000000000000000..

REMX 1.3.0

default\_workspace

DEPLOY & RUN TRANSACTIONS

BALANCEOF : 0x5B38Da6a701c568545dCfcB03Fc

Calldata Parameters call

0: uint256: 999998000000000000000000  
000

decimals

0: uint8: 18

name

0: string: MyToken

symbol

0: string: MTK

totalSupply

0: uint256: 1000000000000000000000000  
0000

Low level interactions i

CALldata Transact

Compile ▾ Home MyToken.sol X

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract MyToken {
5     string public name = "MyToken";
6     string public symbol = "MTK";
7     uint8 public decimals = 18;
8     uint256 public totalSupply;
9
10    mapping(address => uint256) public balanceOf;
11    mapping(address => mapping(address => uint256)) public allowance;
12
13    event Transfer(address indexed from, address indexed to, uint256 value);
14    event Approval(address indexed owner, address indexed spender, uint256 value);
15
16    constructor(uint256 _initialSupply) {
17        totalSupply = _initialSupply;
18        balanceOf[msg.sender] = _initialSupply;
19        emit Transfer(address(0), msg.sender, _initialSupply);
    }
}
```

Explain contract

[vm] from: 0x5B3...eddC4 to: MyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...  
call to MyToken.allowance

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: MyToken.allowance(address,address) data: 0x...  
transact to MyToken.transferFrom pending ...

## 🌟 5.4 Transfer Test – Successful Token Transfer

## DEPLOY & RUN TRANSACTIONS

**Deployed Contracts** 1

MYTOKEN AT 0xD91...39138 (MEMORY)

**BALANCE:** 0 ETH

**APPROVE**

\_spender: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2  
 \_value: 30000000000000000000

**TRANSFER**

\_to: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2  
 \_value: 10000000000000000000

**TRANSFERFROM**

\_from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4  
 \_to: 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db  
 \_value: 10000000000000000000

**ALLOWANCE**

REMX 1.3.0

DEPLOY & RUN TRANSACTIONS

MyToken.sol

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract MyToken {
5     string public name = "MyToken";
6 }
```

**ALLOWANCE**

0: uint256: 30000000000000000000

**BALANCEOF**

0: uint256: 99999800000000000000000000000000

**DECIMALS**

0: uint8: 18

**NAME**

0: string: MyToken

**SYMBOL**

0: string: MTK

**TOTALSUPPLY**

0: uint256: 100000000000000000000000000000000

**EXPLAIN CONTRACT**

CALL [vm] from: 0xB3...eddC4 to: MyToken.(constructor) value: 0 wei data: 0x608...0000 logs: 1 hash: 0xb3...b2bb call to MyToken.decimals

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

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: MyToken.symbol() data: 0x95d...89b41 call to MyToken.totalSupply

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: MyToken.totalSupply() data: 0x181...60ddd call to MyToken.name

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: MyToken.name() data: 0x86f...dde03 call to MyToken.balanceOf errored: Error encoding arguments: TypeError: Invalid address (argument="address", value="", code=INVALID\_ARGUMENT, version=6.14.0) (argument="") call to MyToken.balanceOf

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

Scan Alert Initialize as git repo

ENG IN 22:53 03-12-2025

```
[vm] from: 0x5B3...edC4 to: MyToken.(constructor) value: 0 wei data: 0x608...00000 logs: 1 hash: 0x0b3...b2bbb
call to MyToken.decimals

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB8875f56beddC4 to: MyToken.decimals() data: 0x313...ce567
call to MyToken.symbol

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB8875f56beddC4 to: MyToken.symbol() data: 0x95d...89b41
call to MyToken.totalSupply

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB8875f56beddC4 to: MyToken.totalSupply() data: 0x181...60ddd
call to MyToken.name

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB8875f56beddC4 to: MyToken.name() data: 0x06f...dde03
call to MyToken.balanceOf errored: Error encoding arguments: TypeError: invalid address (argument="address", value="", code=INVALID_ARGUMENT, version=6.14.0) (argument=""

call to MyToken.balanceOf
```

## ⌚ 5.5 Approve & Allowance Test

### ◆ Step 1: Approve Spender

I used the **approve()** function to allow another address to spend my tokens.

- **Owner Address:** 0x5B38...edC4
- **Spender Address:** 0xAb84...35cb2
- **Approved Amount:** 30000000000000000000000000

After clicking **transact**, the transaction succeeded and generated an **Approval event**.

### ◆ Step 2: Check Allowance

Next, I used the **allowance()** function to confirm the approved amount.

- **Owner:** 0x5B38...edC4
- **Spender:** 0xAb84...35cb2
- **Returned Allowance:** 3000000000000000000000000000

This confirms the spender is allowed to transfer the approved number of tokens.

### ◆ Step 3: transferFrom Test

Finally, I performed **transferFrom()**, where the spender transfers tokens from the owner to another account.

- **From:** 0x5B38...edC4
- **To:** 0x4B20...02db
- **Value:** 1000000000000000000000000000

The transaction was successful and emitted a **Transfer event**.

▼ MYTOKEN AT 0xD91...39138 (MEMORY)

Balance: 0 ETH

**APPROVE**

\_spender: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2

\_value: 30000000000000000000000000000000

Calldata Parameters **transact**

**TRANSFER**

\_to: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2

\_value: 10000000000000000000000000000000

Calldata Parameters **transact**

**TRANSFERFROM**

\_from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

\_to: 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db

\_value: 10000000000000000000000000000000

Calldata Parameters **transact**

**ALLOWANCE**

: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2

Calldata Parameters **call**

0: uint256: 30000000000000000000000000000000

**BALANCEOF**

: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

Calldata Parameters **call**

0: uint256: 99999800000000000000000000000000

[vm] from: 0x5B3...eddC4 to: MyToken.approve(address,uint256) 0xd91...39138 value: 0 wei data: 0x095...c0000 logs: 1  
hash: 0xdc2...6315f

Debug ▾

## 5.6 transferFrom

The screenshot shows a blockchain transaction interface with two separate sections for the `transferFrom` function.

**TRANSFER** section:

- `_to:` 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
- `_value:` 10000000000000000000

**Buttons:**

- Calldata
- Parameters
- 

**TRANSFERFROM** section:

- `_from:` 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
- `_to:` 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db
- `_value:` 10000000000000000000

**Buttons:**

- Calldata
- Parameters
-