// SPDX-License-Identifier: MIT

pragma solidity ^0.8.9;

import "[@openzeppelin/contracts@4.8.1/token/ERC20/ERC20.sol](mailto:@openzeppelin/contracts@4.8.1/token/ERC20/ERC20.sol)";

import "[@openzeppelin/contracts@4.8.1/token/ERC20/extensions/ERC20Burnable.sol](mailto:@openzeppelin/contracts@4.8.1/token/ERC20/extensions/ERC20Burnable.sol)";

import "[@openzeppelin/contracts@4.8.1/access/Ownable.sol](mailto:@openzeppelin/contracts@4.8.1/access/Ownable.sol)";

import "[@openzeppelin/contracts@4.8.1/token/ERC20/extensions/draft-ERC20Permit.sol](mailto:@openzeppelin/contracts@4.8.1/token/ERC20/extensions/draft-ERC20Permit.sol)";

import "[@openzeppelin/contracts@4.8.1/token/ERC20/extensions/ERC20Votes.sol](mailto:@openzeppelin/contracts@4.8.1/token/ERC20/extensions/ERC20Votes.sol)";

contract SimpleToken is ERC20, ERC20Burnable, Ownable, ERC20Permit, ERC20Votes {

constructor() ERC20("DKCoin.sol", "DKC") ERC20Permit("DKCoin.sol") {

\_mint(msg.sender, 1000000 \* 10 \*\* decimals());

}

function mint(address to, uint256 amount) public onlyOwner {

\_mint(to, amount);

}

// The following functions are overrides required by Solidity.

function \_afterTokenTransfer(address from, address to, uint256 amount)

internal

override(ERC20, ERC20Votes)

{

super.\_afterTokenTransfer(from, to, amount);

}

function \_mint(address to, uint256 amount)

internal

override(ERC20, ERC20Votes)

{

super.\_mint(to, amount);

}

function \_burn(address account, uint256 amount)

internal

override(ERC20, ERC20Votes)

{

super.\_burn(account, amount);

}

}

Etherscan: https://goerli.etherscan.io/tx/0xef4e0f3b7263aa8eb56c6023c569a6dd3ba9d636e641d6b16f6559caefb78e21