// SPDX-License-Identifier: MIT
// Compatible with OpenZeppelin Contracts ^5.4.0

pragma solidity ^0.8.27;
import {Ownable} from "@openzeppelin/contracts/access/Ownable.sol";

///@dev se importa el token para que sea valida la function hasMoreThan100Tokens , pero
/// aca se importa la interfaces
import {IERC20} from "@openzeppelin/contracts/token/ERC20/IERC20.sol";

import {IERC20Metadata} from "@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol";

import {ICO} from "./ICO.sol";

//El problema está en que IERC20 de OpenZeppelin no incluye la función decimals() en la interfaz básica.
// 🔗 Solución:
// Opción 1: Usar IERC20Metadata (Recomendado)

contract Permisos is Ownable {
mapping(address => bool) public blackList;

```
// IERC20 immutable public token;///aca igualmente hay que setear la direccion
del token en el constructor
// IERC20 public token;
   // ✅ Cambiar a IERC20Metadata
IERC20Metadata public token;

ICO public ico;

event BlackListSet(
    address indexed whoSet,
    address indexed whoWasSeteer,
    bool permission
);

event TokenSet(IERC20 indexed tokenAddress);

constructor() Ownable(msg.sender) {}

// ///@dev event cuando se hace el cambio del TOKEN.
// function SetToken(IERC20 _token) external onlyOwner {
// ✅ Actualizar el tipo del parámetro
function SetToken(IERC20Metadata _token) external onlyOwner {
    token = _token;
    emit TokenSet(_token);
}

    function SetICO(ICO _ico) external onlyOwner {
    ico = _ico;
```

```
    // emit TokenSet(_token);
}

///@dev le pone external por que va ser llamada de afuera , solo lo puede
llamar el onlyOwner , ya seteado en el constructor.
function setBlacklist(address _addr, bool _permission) external onlyOwner {
    blackList[_addr] = _permission;

    emit BlackListSet(msg.sender, _addr, _permission);
}

///@dev para que esto sea valido yo debo acceder al Token(contrato ) por eso se
importa Arriba
function hasMoreThan100Tokens(
    address _addr
) public view returns (bool _hasMoreThan100Tokens) {
    uint256 balances = token.balanceOf(_addr);

    // if(balances>100){
    //     return true;

    // } else return  false;
// ✅ Excluir al contrato ICO de esta verificación
if (_addr == address(ico)) {
    return false;
}



    return balances > 100 * 10 ** token.decimals()  ? true : false;
}
///@dev importantisismo FALTAN LOS EVENTO , Ya que cuando estas seteando una
Variable deberias porner los eventos!!
///@dev en este caso se van a poner los eventos.
function IsBlackList(address _add) public view returns (bool _blackList) {
    return blackList[_add];
}

function HavePermission(
    address _addr
) external view returns (bool _permission) {

        // ✅ Excluir al contrato ICO de esta verificación
if (_addr == address(ico)) {
    return true;
}



    if (IsBlackList(_addr) || hasMoreThan100Tokens(_addr)) {
        return false;
    }
```

```
        else return true;
    }
```

}