// SPDX-License-Identifier: MIT
// Compatible with OpenZeppelin Contracts ^5.4.0
pragma solidity ^0.8.27;

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

///@dev aca se importa la interfaz de permisos para que pueda usar la funcion HavePermission
interface IPermisos {
function HavePermission(
address _addr
) external view returns (bool _permission);
}

contract token is ERC20 {
IPermisos public immutable permisos;
error NoPermission(address _whois);
constructor(
/*address recipient , */ IPermisos _permisos
) ERC20("EthKipu", "EKT") {
_mint(msg.sender, 100 * 10 ** decimals());
permisos = _permisos;
}
///@dev dado que el _tranfer no tiene virtual origina no se TOCA , solo se pone los 2 transfer y transferFrom
/// se cambia virtual por override.
///@dev aca se modifica por el punto del que tranfiere tenga los permisos!!(osea se importa ?? se hace una interfas??

```
/// TODO ESTO POR QUE DICE PERMITE TRANSFERIR SI TIENE LOS PERMISOS
HavePermission(address _addr)
function transfer(
    address to,
    uint256 value
) public override returns (bool) {
    // address owner = _msgSender();
    // _transfer(owner, to, value);
    // // return true;
    if (!permisos.HavePermission(to)) revert NoPermission(to);
    ///@dev aca ves si el receptor del token tiene permisos para recibir
    return super.transfer(to, value);
}
function transferFrom(
    address from,
    address to,
    uint256 value
) public override returns (bool) {
    // address spender = _msgSender();
```

```
        // _spendAllowance(from, spender, value);
        // _transfer(from, to, value);
        // return true;
        ///@dev aca ves si el receptor del token tiene permisos para recibir
        ///@dev como se reviete aca se agrega el CUSTOM ERROR!!
        if (!permisos.HavePermission(to)) revert NoPermission(to);
        return super.transferFrom(from, to, value);
    }
```

}