



Universitat
de les Illes Balears

GRAU D'ENGINYERIA TELEMÀTICA
PRIVACITAT I GESTIÓ D'IDENTITATS DIGITALS
IMPLEMENTACIÓ DE CROMOS
DIGITALS A LA XARXA SEPOLIA
AMB ERC721 I ERC20
DOCUMENTACIÓ

Olga Domingo Muñoz

22/01/2025

Índex

Índex.....	2
Introducció.....	3
Disseny del Sistema.....	4
Estàndards ERC Utilitzats.....	4
Implementació de Pinata.....	4
Integració amb MetaMask.....	4
Integració amb OpenSea.....	5
Ús de la Xarxa de Proves Sepolia.....	5
Ús del IDE Remix.....	5
Desenvolupament i Implementació.....	6
Contracte Intel·ligent ERC20.....	6
Contracte Intel·ligent ERC721.....	7
Importació de contractes amb OpenZeppelin.....	10
Variables globals.....	10
Esdeveniments.....	10
Constructor.....	11
Funcions.....	11
Emmagatzematge de les metadades amb Pinata.....	12
Visualització dels NFTs a la plataforma OpenSea.....	13
Proves i Validació.....	15
Desplegament del contracte ERC20.....	15
Desplegament del contracte ERC721.....	16
Configuració d'aprovació de despeses (approve).....	18
Creació i assignació d'un NFT (safeMint).....	19
Compra i transferència de NFTs (buyNFT, safeTransferFrom).....	20
Retirada de fons (withdraw).....	25
Configuració de la Whitelist.....	28
Altres funcionalitats del ERC20.....	29
Altres funcionalitats del ERC721.....	34
Conclusions.....	37
Futures Millores.....	38
Referències.....	39
Annex.....	40

Introducció

En el marc de l'evolució tecnològica actual, la implementació de sistemes basats en *blockchain* està transformant la manera com interactuem amb el món digital. Aquest projecte es centra en el desenvolupament d'un **sistema de cromos digitals inspirat en sèries de televisió**, utilitzant les normes dels **estàndards ERC20 i ERC721 d'Ethereum**. L'objectiu principal és crear una plataforma que permeti als usuaris col·leccionar i gestionar cromos digitals d'una manera segura, transparent i descentralitzada.

El projecte combina la funcionalitat dels **tokens fungibles (ERC20)** per gestionar una moneda virtual, amb la singularitat dels **tokens no fungibles (ERC721)** per representar els cromos digitals. Cada cromo està associat a una sèrie de televisió i compta amb un identificador únic i metadades que defineixen les seves característiques.

El present document descriu el desenvolupament tècnic del projecte, incloent-hi els detalls sobre el disseny del sistema, el funcionament dels contractes intel·ligents implementats i les funcionalitats principals que s'ofereixen als usuaris. A més, es presenten les proves realitzades per garantir el correcte funcionament del sistema i es discuteixen les possibles millores futures per ampliar les seves funcionalitats.

Disseny del Sistema

En aquest apartat es descriu el funcionament dels estàndards i eines utilitzades en el projecte per gestionar els cromos digitals, centrant-nos en les seves funcionalitats principals i com contribueixen al sistema.

Estàndards ERC Utilitzats

- **ERC-20 (Tokens fungibles)**: Aquest estàndard permet crear una moneda digital anomenada *EpisodeCoin (EPIS)* que s'utilitza dins del sistema com a mitjà de pagament. Aquests tokens són intercanviables i uniformes, cosa que significa que qualsevol token del mateix tipus té el mateix valor. Aquests tokens són necessaris per adquirir els cromos digitals.
- **ERC-721 (Tokens no fungibles)**: Aquest estàndard s'utilitza per representar els cromos digitals anomenats *SeriesNFTs (SNFT)*, ja que garanteix que cada un d'ells sigui únic i no es pugui intercanviar per un altre de manera directa. Cada cromo està associat a un identificador únic i conté metadades que descriuen la sèrie que representen: el nom, una breu descripció i una foto.

Implementació de Pinata

Pinata s'utilitza per a l'**emmagatzematge de dades** dels cromos digitals. Aquesta eina permet guardar les metadades i les imatges de manera descentralitzada mitjançant el sistema **IPFS** (*InterPlanetary File System*). Això significa que la informació no depèn d'un servidor centralitzat i es garanteix la seva integritat i permanència al llarg del temps.

Les metadades inclouen informació com el **nom del cromo**, la seva **descripció** i l'**enllaç a la seva imatge**, també emmagatzemada de forma separada. Aquesta informació és accessible per qualsevol plataforma compatible amb els estàndards ERC721.

Integració amb MetaMask

S'integrarà **MetaMask** com a **cartera**, cosa que permet als usuaris emmagatzemar i gestionar els **tokens ERC20 i ERC721** com també altres criptomonedes, directament des del navegador. Aquest actua com una **extensió** que es connecta a diferents *blockchains*, permetent interactuar amb **contractes intel·ligents** i realitzar **transaccions**.

Integració amb OpenSea

OpenSea és una de les plataformes més grans i conegudes per al comerç i visualització d'actius digitals basats en *blockchain*, com són els **NFTs**. En aquest cas, s'ha integrat la **testnet** d'**OpenSea** perquè els usuaris puguin **visualitzar els cromos digitals** de manera fàcil i intuïtiva.

El perfil **SeriesNFTs** actua com una galeria virtual on els usuaris poden consultar tots els cromos digitals disponibles. D'aquesta manera, **OpenSea** no només és una plataforma per mostrar els cromos disponibles, sinó també un mitjà per augmentar la visibilitat i l'atractiu del projecte.

Ús de la Xarxa de Proves Sepolia

El desenvolupament i les proves del sistema s'han realitzat utilitzant la **xarxa de proves (testnet) Sepolia d'Ethereum**. Aquesta xarxa és ideal per simular transaccions reals sense utilitzar fons econòmics reals, ja que té avantatges com:

- La facilitat per obtenir **Sepolia ETH** a través de *faucets* per realitzar proves.
- L'entorn segur que verifica el correcte comportament dels contractes intel·ligents.
- La simulació de l'entorn *Ethereum* principal (*mainnet*) amb costos baixos.

Aquest enfocament assegura que el sistema està correctament implementat i funcionant abans de desplegar-lo a la xarxa principal.

Ús del IDE Remix

Es tracta d'una **plataforma de desenvolupament en línia** per crear, desplegar i provar contractes intel·ligents. Permet escriure codi en **Solidity**, compilar-lo, depurar-lo i interactuar amb els contractes desplegats.

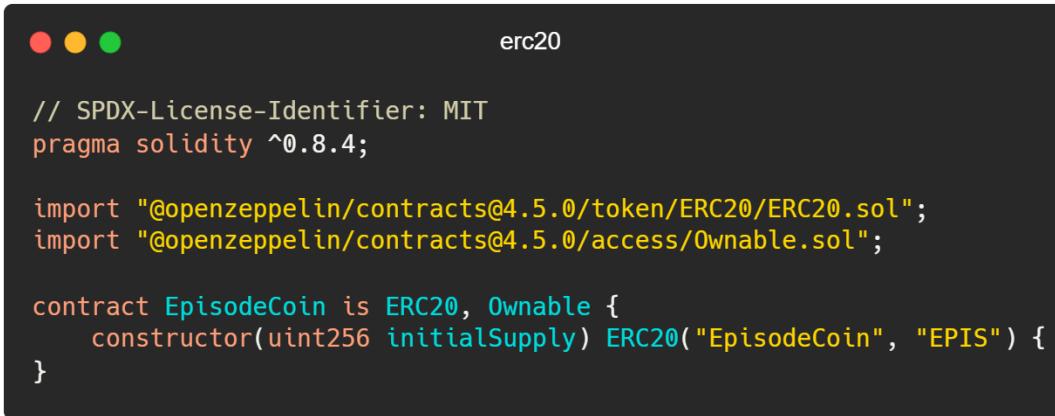
Amb aquest disseny, el sistema ofereix una experiència completa, segura i descentralitzada pels usuaris, combinant innovació *blockchain* amb eines modernes com **Pinata** i **OpenSea**.

Desenvolupament i Implementació

Per al desenvolupament del projecte, la primera passa és enllaçar **Remix** amb la nostra cartera de **MetaMask**, connectada a la xarxa de **Sepolia**, d'una forma molt senzilla.

Una vegada fet, podem programar els dos contractes intel·ligents que necessitem, els quals s'explicaran amb detall a continuació:

Contracte Intel·ligent ERC20



```
erc20

// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;

import "@openzeppelin/contracts@4.5.0/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts@4.5.0/access/Ownable.sol";

contract EpisodeCoin is ERC20, Ownable {
    constructor(uint256 initialSupply) ERC20("EpisodeCoin", "EPIS") {
}
```

Aquest crea la **moneda** que s'utilitzarà al projecte, un **token ERC20** anomenat **EpisodeCoin** (amb símbol **EPIS**). Ho fa utilitzant el codi de la biblioteca de **OpenZeppelin** per garantir que segueixi els estàndards de l'**ERC20**, un estàndard comú per a la **creació de tokens a Ethereum**.

Es tracta d'un contracte molt senzill, tot i que hi ha alguns aspectes a considerar:

1. **ERC20**: És un estàndard per a la creació de tokens que permet que els usuaris puguin **transferir, aprovar i gestionar les monedes** de manera senzilla. El contracte **hereta** la funcionalitat de l'ERC20 per a l'ús d'aquestes funcions bàsiques.
2. **Ownable**: Hereta el contracte **Ownable** de **OpenZeppelin**, el que fa que el contracte tingui un propietari (la persona que desplega el contracte) amb drets especials per realitzar accions com canviar paràmetres o transferir el control del contracte a una altra adreça. Això afegeix un sistema de control centralitzat.
3. **Constructor**: Quan el contracte es desplega, el constructor s'executa i emet (minteja) una **quantitat inicial de tokens EpisodeCoin** a l'adreça que ha desplegat el contracte. La quantitat d'**EPIS** es determina pel paràmetre **initialSupply**, que s'ha d'introduir en el moment de la creació del contracte.

Contracte Intel·ligent ERC721

```
● ● ● erc721

// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;

import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/utils/Counters.sol";
import "@openzeppelin/contracts/security/ReentrancyGuard.sol";


contract SeriesNFTs is ERC721, ReentrancyGuard {
    using Counters for Counters.Counter;
    address private owner; //Variable para almacenar la dirección del propietario
    IERC20 public tokenAddress; //Dirección del token ERC20
    uint256 public rate = 1 * 10 ** 18; //Valor de mintero - 1 EPI
    uint256 public maxWithdrawalAmount = 100 * 10 ** 18;
    string public baseTokenURI; //URL base para los metadatos
    Counters.Counter private _tokenIdCounter;
    event Mint(address indexed user, uint256 tokenId);
    event Whitelisted(address indexed user);
    event RemovedFromWhitelist(address indexed user);
    event NFTBought(address indexed buyer, uint256 indexed tokenId, uint256 price);

    mapping(address => bool) public whitelisted; //Lista blanca de direcciones

    constructor(address _tokenAddress, string memory _baseTokenURI)
        ERC721("SeriesNFTs", "SNFT") {
        owner = msg.sender; //Establecemos el propietario al desplegar el contrato
        tokenAddress = IERC20(_tokenAddress);
        baseTokenURI = _baseTokenURI;

        //Añadimos automáticamente al propietario a la lista blanca
        whitelisted[owner] = true;
    }

    //Función que devuelve la URL asociada a un token específico que contiene sus metadatos
    function tokenURI(uint256 tokenId)
        public
        view
        virtual
        override
        returns (string memory)
    {
        require(
            ownerOf(tokenId) != address(0),
            "ERC721Metadata: URI query for nonexistent token"
        );
        return baseTokenURI; // Devuelve la URL base proporcionada
    }

    //Modificador personalizado para verificar si el mensaje proviene de una dirección en la
    //lista blanca
    modifier onlyWhitelisted() {
        require(whitelisted[msg.sender], "You are not whitelisted to perform this action");
        _;
    }
}
```

```

//Modificador personalizado para verificar si el mensaje proviene del propietario
modifier onlyOwner() {
    require(msg.sender == owner, "Caller is not the owner");
    _;
}

//Función para agregar direcciones a la lista blanca, solo el propietario puede
ejecutarla
function addToWhitelist(address user) external onlyOwner {
    require(user != owner, "Owner cannot be added to the whitelist");
    require(!whitelisted[user], "Address is already whitelisted");
    whitelisted[user] = true;
    emit Whitelisted(user);
}

//Función para eliminar direcciones de la lista blanca, solo el propietario puede
ejecutarla
function removeFromWhitelist(address user) external onlyOwner {
    require(user != owner, "Owner cannot be removed from the whitelist");
    require(whitelisted[user], "Address is not whitelisted");
    whitelisted[user] = false;
    emit RemovedFromWhitelist(user);
}

//Función para acuñar un NFT solo si la dirección está en la lista blanca
function safeMint() public onlyWhitelisted {
    //Verificamos si el usuario tiene suficientes tokens
    require(tokenAddress.balanceOf(msg.sender) >= rate, "Not enough tokens to mint");

    //Intentamos hacer la transferencia
    require(tokenAddress.transferFrom(msg.sender, address(this), rate),
    "Payment failed");

    //Crear el nuevo token
    uint256 tokenId = _tokenIdCounter.current();
    _tokenIdCounter.increment();
    _safeMint(msg.sender, tokenId);
    emit Mint(msg.sender, tokenId);
}

//Función para retirar los tokens ERC20 del contrato solo si la dirección está en la
lista blanca
function withdraw(uint256 amount) external onlyWhitelisted nonReentrant {
    uint256 allowance = tokenAddress.allowance(msg.sender, address(this));
    //Verifica los tokens aprobados para el contrato ERC721
    require(allowance >= amount, "Not enough approved tokens to withdraw");

    //Limitar la cantidad de tokens que se pueden retirar
    require(amount <= maxWithdrawalAmount, "Withdrawal exceeds the limit");

    //Transfiere los tokens ERC20 desde el usuario al contrato ERC721
    require(tokenAddress.transferFrom(msg.sender, address(this), amount),
    "Transfer failed");

    //Los tokens ahora están almacenados dentro del contrato y pueden ser utilizados más
    tarde como para la conversión a otra moneda
}

```

```

//Función que permite a un usuario comprar un NFT mediante el pago en tokens ERC20,
//transfiriendo el token al propietario actual y el NFT al comprador
function buyNFT(uint256 tokenId) public nonReentrant {
    //Obtenemos el precio del NFT y verificamos usuario tenga suficientes tokens
    uint256 price = getMintPrice();
    require(tokenAddress.balanceOf(msg.sender) >= price, "Not enough EPIS to buy NFT");

    //Verificamos que el comprador no sea el propietario
    address seller = ownerOf(tokenId);
    require(msg.sender != seller, "You cannot buy your own NFT");

    //Calculamos la comisión (1% del precio) y se la transferimos al propietario del
    //contrato
    uint256 commission = price / 100; // 1% de comisión
    require(tokenAddress.transferFrom(msg.sender, owner, commission),
    "Commission transfer failed");

    //Transferimos el precio al propietario del NFT
    require(tokenAddress.transferFrom(msg.sender, seller, price - commission),
    "Transfer failed");

    //Transferimos el NFT al comprador
    _safeTransfer(seller, msg.sender, tokenId, "");

    //Emitimos un evento para la compra
    emit NFTBought(msg.sender, tokenId, price);
}

//Función para ver el balance de tokens del contrato
function getBalance() external view returns (uint256) {
    return tokenAddress.balanceOf(address(this));
}

//Función para cambiar el precio del NFT solo si la dirección está en la lista blanca
function setRate(uint256 newRate) public onlyWhitelisted {
    rate = newRate;
}

//Función para ver el precio de acuñar un NFT
function getMintPrice() public view returns (uint256) {
    return rate;
}

//Función para cambiar la cantidad máxima de tokens a retirar
function setMaxWithdrawalAmount(uint256 newLimit) external onlyWhitelisted {
    maxWithdrawalAmount = newLimit;
}

//Función para ver el propietario de un NFT
function getOwnerOf(uint256 tokenId) public view onlyWhitelisted returns (address) {
    return ownerOf(tokenId); // Devolvemos la dirección del propietario
}
}

```

Aquest contracte permet **crear i vendre NFTs** (tokens ERC721) **mitjançant pagaments en tokens ERC20**, els quals són anomenats **SeriesNFTs** (amb símbol **SNFT**).

Es poden distingir tres tipus d'usuaris en aquest contracte: **el propietari**, que té accés complet per gestionar el contracte i modificar paràmetres; **els que es troben la llista blanca**, que poden realitzar accions amb privilegis com mintejar NFTs o retirar tokens; i **els usuaris normals**, que només poden comprar NFTs i consultar informació del contracte, sense accés a les funcions restringides.

Són moltes les possibilitats que ens ofereix aquest contracte intel·ligent, sent més complexe que el anterior, combinant funcions d'un ERC721 (NFT) i un ERC20 (moneda). A continuació s'explicarà amb detall cada part del codi:

Importació de contractes amb OpenZeppelin

- **IERC20**: Interfície per interactuar amb contractes ERC20.
- **ERC721**: Contracte base per treballar amb NFTs.
- **Counters**: Eina per gestionar comptadors (els IDs dels tokens).
- **ReentrancyGuard**: S'utilitza per protegir contra atacs de reentrada, assegurant que les funcions crítiques no es poden cridar repetidament mentre estan en execució.

Variables globals

- **owner**: Desa l'adreça del propietari del contracte.
- **tokenAddress**: Adreça del contracte **ERC20** utilitzat per als pagaments, just explicat.
- **rate**: Defineix el preu inicial per crear un nou NFT en termes de tokens ERC20 (**1 EPI**).
- **maxWithdrawalAmount**: Defineix el límit màxim de tokens ERC20 que un usuari pot retirar.
- **baseTokenURI**: Adreça URL per les metades del NFT (on s'emmagatzemen les imatges i informació associada).
- **tokenIdCounter**: Comptador d'IDs per identificar els NFTs de forma única.
- **whitelisted**: Mapeig d'adreces permeses per interactuar amb certes funcions.

Esdeveniments

- **Mint**: S'emet quan es crea un nou NFT.
- **Whitelisted i RemovedFromWhitelist**: S'emet quan s'afegeix o elimina una adreça de la llista blanca.
- **NFTBought**: S'emet quan un NFT és comprat.

Constructor

Aquest estableix el **propietari** (qui desplega el contracte), l'**adreça del contracte ERC20** i l'**URL base de les metades** dels NFTs. També afegeix automàticament al propietari a la llista blanca.

Modificadors

- **onlyWhitelisted**: Permet executar funcions només a les adreces que estan a la llista blanca.
- **onlyOwner**: Permet executar funcions només al propietari del contracte.

Funcions

- **tokenURI(uint256 tokenId)**: Retorna l'URL base per a les metades d'un token, útil perquè les plataformes de NFTs (com **OpenSea**) puguin mostrar els detalls del NFT.
- **addToWhitelist(address user)**: Permet al propietari afegir adreces a la llista blanca.
- **removeFromWhitelist(address user)**: Permet al propietari eliminar adreces de la llista blanca.
- **safeMint()**: Aquesta funció permet crear (mintear) un nou NFT, però només si l'usuari està a la llista blanca. Abans de crear-lo, es verifica que l'usuari tingui suficients tokens ERC20 per pagar el mint. Si té saldo suficient, es transfereix el pagament i es crea el nou NFT. Una vegada mintejat es podrà visualitzar l'NFT tant a **MetaMask** com a la plataforma **OpenSea**.
- **withdraw(uint256 amount)**: Permet a un usuari retirar tokens ERC20 del contracte, només si està a la llista blanca. Es verifica que la quantitat aprovada per l'usuari sigui suficient i es limita la quantitat màxima que es pot retirar. Després, es transfereix la quantitat sol·licitada de tokens ERC20 al contracte ERC721. Les monedes seran retirades i emmagatzemades al contracte fins que l'usuari decideixi realitzar alguna altra acció (com una possible futura conversió a una altra moneda, amb l'objectiu de generar un mercat real).
- **buyNFT(uint256 tokenId)**: Aquesta funció permet comprar un NFT mitjançant un pagament en tokens ERC20. El comprador ha de tenir suficients tokens per cobrir el preu de l'NFT. Es verifica que el comprador no sigui el propietari de l'NFT i es calcula una **comissió del 1%** que es transfereix al propietari del contracte. Després, el comprador paga el preu restant al propietari de l'NFT i l'NFT es transfereix al comprador.
- **getBalance()**: Retorna la quantitat de tokens ERC721 emmagatzemats al contracte.
- **setRate(uint256 newRate)**: Permet canviar el preu de crear un NFT, però només si l'usuari està a la llista blanca.
- **getMintPrice()**: Retorna el preu actual per crear un NFT.
- **setMaxWithdrawalAmount(uint256 newLimit)**: Permet al propietari canviar el límit màxim de tokens ERC20 que es poden retirar del contracte.
- **getOwnerOf(uint256 tokenId)**: Retorna l'adreça del propietari d'un NFT específic només si l'usuari està a la llista blanca.

Juntament amb el codi, i amb l'objectiu d'ampliar les funcionalitats, hem integrat altres tecnologies com son **Pinata** i **OpenSea**:

Emmagatzematge de les metadades amb Pinata

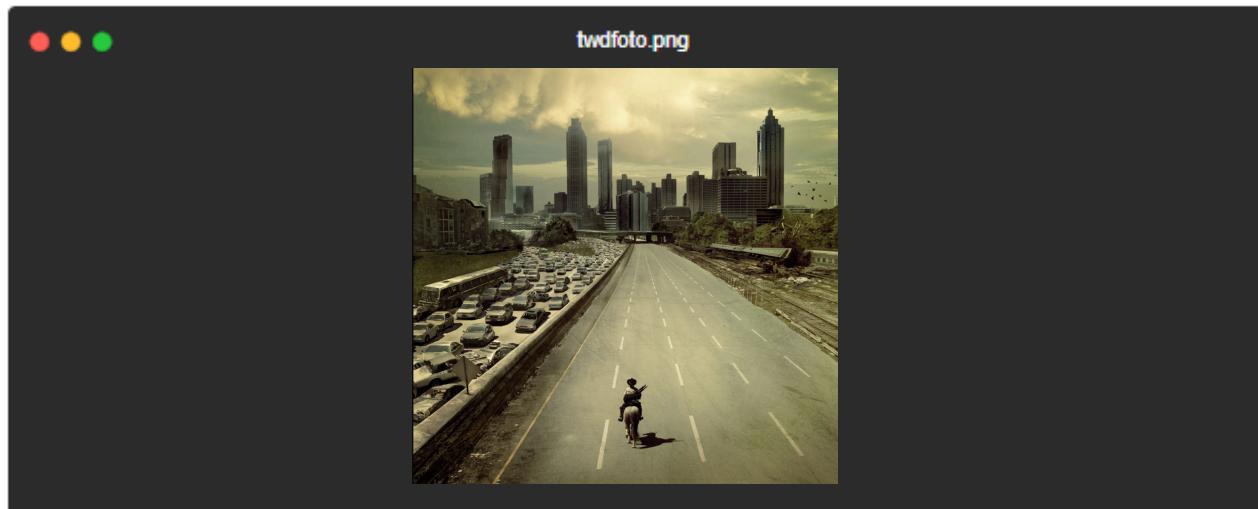
Per poder emmagatzemar les metadades i que siguin accesibles pel contracte, s'ha utilitzat **Pinata**, una plataforma de desenvolupament **Web3** que utilitz a **IPFS** (InterPlanetary File System) per proporcionar una solució completa per al desenvolupament i la gestió de projectes descentralitzats.

Allà s'han penjat els fitxers corresponents a cada NFT, en concret la seva imatge i un arxiu **JSON** amb les dades necessàries com el nom i la descripció, juntament amb la URL que apunta a la imatge, emmagatzemada de forma separada. Les següents imatges són un exemple:

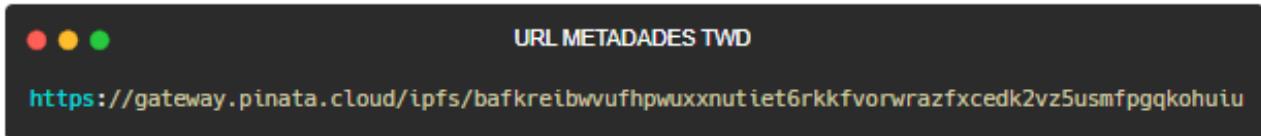
IPFS FILES					
	NAME	SIZE	CID	CREATED	
<input type="checkbox"/>	twdnft.json	328 B	bafkr...ohuiu		1/19/2025
<input type="checkbox"/>	twdfoto.png	1.70 MB	bafyb...rkbza		1/19/2025

```
twdnft.json

{
  "name": "The Walking Dead",
  "description": "Un grupo de supervivientes lucha por mantenerse con vida en un mundo invadido por caminantes, mientras descubren que el mayor desafío no son los muertos, sino los vivos.",
  "image": "https://ipfs.io/ipfs/bafybeicu43nrg2qs4c5biqyh6xobj5cca2p5baaus5a77k7lbneorkbza"
}
```



Això ens proporciona un enllaç únic i immutable per a cada NFT, al qual el pot accedir des de **Internet**, sense necessitat d'obtenir un servidor propi. Un enllaç com el següent és el paràmetre que s'ha d'introduir quan es crea el contracte ERC721 associat a un NFT específic, en aquest cas el associat a la sèrie “*The Walking Dead*”:



Aquests enllaços de totes les sèries seran compartits a l'apartat [Annex](#).

Finalment, per integrar **Pinata** amb els metadades dels NFTs, cal disposar d'una **secret key** i una **API key**. Aquestes claus són credencials proporcionades a cada usuari per autenticar-se i garantir que només persones autoritzades puguin penjar i gestionar fitxers a través de la seva API.

API KEYS

NAME	KEY	DATE ISSUED	MAX USES	TIMES USED	STATUS
newkey	[REDACTED]	1/16/2025	0	0	Active

Visualització dels NFTs a la plataforma OpenSea

Com s'ha explicat a l'apartat [Integració amb OpenSea](#), hem utilitzat aquesta plataforma en la seva versió de **testnet** per **visualitzar els NFTs**, sent una galeria virtual i oferint una interfície més extensa, intuïtiva i estètica que **MetaMask**. **OpenSea** permet explorar i veure els detalls dels tokens, com ara la imatge, les metadades i l'historial de propietat, cosa que facilita comprovar que els NFTs creats estan correctament registrats i funcionen de forma correcta.

No obstant això, els NFTs no es poden comprar ni vendre a través d'**OpenSea**, degut a l'utilització de la moneda pròpia **EpisodeCoin**, que no està integrada en aquesta plataforma. Això vol dir que la compra, venda o altres transaccions amb els NFTs es gestionen exclusivament a través del contracte intel·ligent que s'han desenvolupat, fora de l'entorn d'**OpenSea**.

A continuació es mostra el perfil “[SeriesNFTs](#)” a la plataforma, juntament amb un exemple de la visualització dels NFTs que ofereix:

OpenSea | Lanzamientos | Estadísticas | Crear | Buscar | I

SeriesNFTs

0x8eF0...3DEe Se ha unido December 2024
Explora una colección única inspirada en tus series favoritas, donde cada NFT captura momentos icónicos ... Ver más

Coleccionado 6 Ofertas realizadas Contratos Creado 31 Añadido a favoritos Actividad Más ▾

Estado Cadenas Buscar por nombre

6 artículos

The Walking Dead
SeriesNFTs

La Casa De Papel
SeriesNFTs

Breaking Bad
SeriesNFTs

Friends
SeriesNFTs

Stranger Things
SeriesNFTs

How I Met Your Mother
SeriesNFTs

SeriesNFTs

How I Met Your Mother

Propiedad de you

2 visualizaciones



Descripción

Un grupo de amigos pasa años haciendo lios y buscando el amor, mientras Ted les cuenta a sus hijos cómo conoció a su madre... pero nunca se decide a contarles realmente.

Acerca de SeriesNFTs

Detalles

Dirección del contrato

0x54c9...258f

ID del token

0

Estándar de token

ERC-721

Cadena

Sepolia

Última actualización

Hace 3 días

Ganancias del creador ⓘ

0%

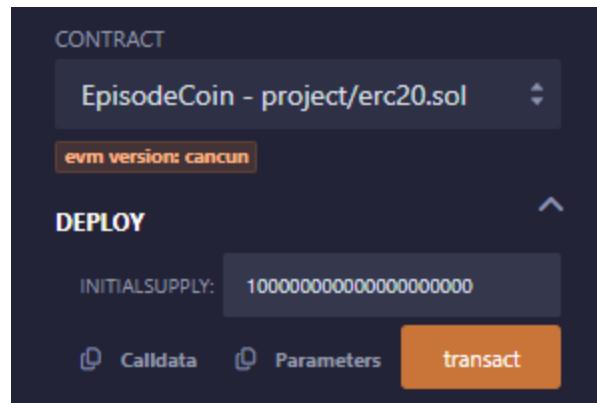
Proves i Validació

En aquest apartat es presenta el procés de proves i validació del projecte, amb l'objectiu de demostrar el correcte funcionament del flux principal relacionat amb la creació i compra de NFTs, així com l'ús de la moneda implementada.

A més, es validaràn funcionalitats extres desenvolupades en els contractes com la retirada de monedes, permisos d'aprovació i la personalització dels detalls associats als tokens. Aquestes proves no només asseguren la funcionalitat bàsica, sinó que també totes les característiques addicionals s'integren correctament.

Desplegament del contracte ERC20

Desplegam el contracte **ERC20** que defineix la moneda **EPI** juntament amb el paràmetre **initialSupply** (total de monedes que volem generar) amb un valor de **1000000000000000000000000**, l'equivalent a **100 EPIS**, degut als 18 decimals que té assignats.



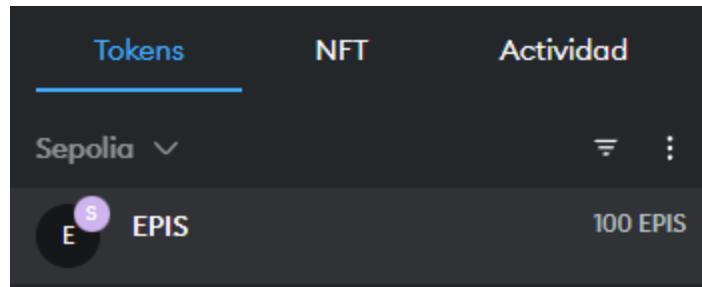
Amb això ja tenim el contracte desplegat, obtenint la seva direcció associada i totes les funcions que pot fer:



<code>approve</code>	address spender, uint256	▼
<code>decreaseAllow...</code>	address spender, uint256	▼
<code>increaseAllow...</code>	address spender, uint256	▼
<code>renounceOwn...</code>		
<code>transfer</code>	address to, uint256 amc	▼
<code>transferFrom</code>	address from, address to	▼
<code>transferOwner...</code>	address newOwner	▼
<code>allowance</code>		▼
<code>balanceOf</code>		▼
<code>decimals</code>		
<code>name</code>		
<code>owner</code>		
<code>symbol</code>		
<code>totalSupply</code>		

En taronja tenim les **funcions d'escriptura**, que modifiquen dades a la *blockchain*, consumint gas i generant una transacció. Per altra banda, en blau tenim les **funcions de lectura**, que només consulten dades del contracte i no consumeixen gas.

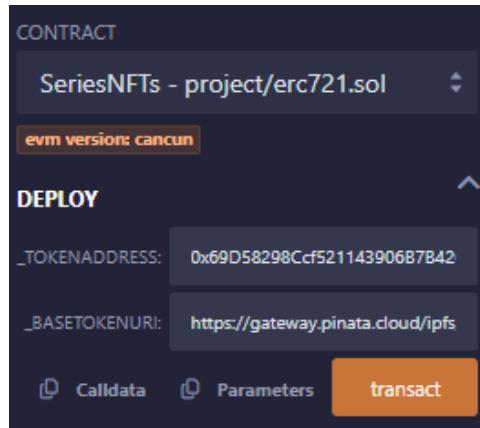
Per veure el saldo del contracte a **MetaMask**, hem de clicar Tokens → Importar Tokens i allà afegir la direcció del contracte **EpisodeCoin** que es troba a **Remix**.



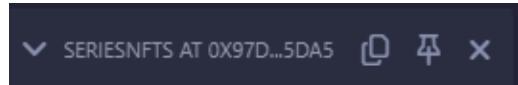
És important destacar que cal **desactivar l'opció "Publish to IPFS"** quan fem el **Deploy**.

Desplegament del contracte ERC721

A continuació desplegam el contracte **ERC721** que gestiona els tokens **SNFT**, afegint com a paràmetres al **_TokenAddress** la direcció del contracte **ERC20** ja desplegat per enllaçar-lo i al **_BaseTokenURI** l'URL de les metadades de l'NFT que volem, tal com s'ha explicat anteriorment.



Una vegada desplegat el contracte ***SeriesNFTs***, obtenim la seva direcció associada i totes les funcions que pot fer:



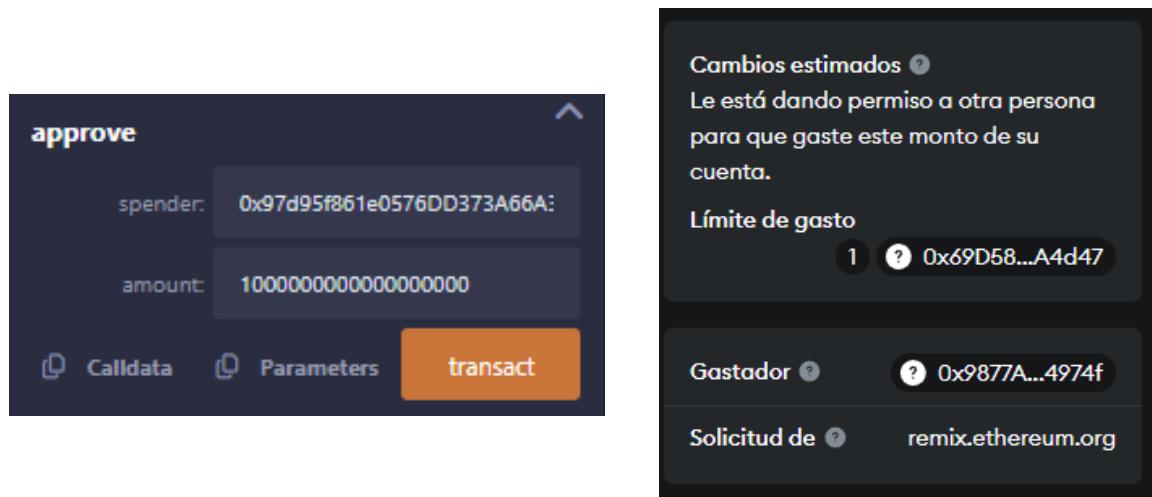
addToWhitelist	address user
approve	address to, uint256 tokenId
buyNFT	uint256 tokenId
removeFromWhitelist	address user
safeMint	
safeTransferFrom	address from, address to
safeTransferFromBatch	address from, address to
setApprovalForAll	address operator, bool isApproved
setMaxWithdrawableBalance	uint256 newLimit
setRate	uint256 newRate
transferFrom	address from, address to
withdraw	uint256 amount
balanceOf	address owner
baseTokenURI	
getApproved	uint256 tokenId
getBalance	
getMintPrice	
getOwnerOf	uint256 tokenId
isApprovedForAll	address owner, address operator
maxWithdrawableBalance	
name	
ownerOf	uint256 tokenId
rate	
supportsInterface	bytes4 interfaceId
symbol	
tokenAddress	
tokenURI	uint256 tokenId
whitelisted	address

És important destacar que cal **desactivar l'opció "Publish to IPFS"** quan fem el *Deploy*.

Configuració d'aprovació de despeses (*approve*)

En aquesta passa ja tenim els dos contractes enllaçats i podem començar amb la creació dels NFTs. No obstant això, perquè un contracte **ERC721** pugui gastar **EPIs** a l'hora de crear NFTs, s'ha d'aprovar la quantitat de monedes que permetem que utilitzi.

Per fer això utilitzarem la **funció *approve* del ERC20**, amb la qual deixam al contracte **ERC721** desplegat gastar una certa quantitat de monedes. En aquest cas aprovarrem la quantitat de **1 EPI**, el valor necessari per poder **mintear 1 NFT**.



Podem consultar la transacció generada a través d'**Etherscan**, una eina que permet veure i verificar les transaccions realitzades a la *blockchain*:

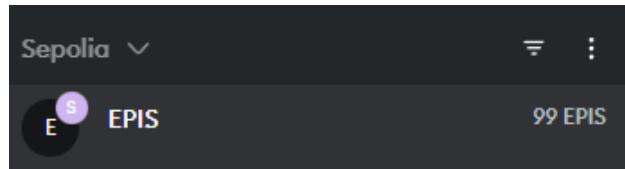
The image shows the Etherscan transaction details for the approve call. The transaction hash is 0x981f698acb85ba72fccb023739adfec07b3203a8da38bfc741954d48d949fd75. The status is Success, and it occurred in block 7503847 with 8 Block Confirmations. The transaction action was an Approve call to 0x69D582980cf521143906B7B42C45fc46846A4d47, spending 0 ETH. The gas price was 5.972425915 Gwei (0.000000005972425915 ETH).

Creació i assignació d'un NFT (*safeMint*)

En aquesta passa ja podrem crear i visualitzar l'NFT que hem escollit i ho farem amb la funció ***safeMint*** del **ERC721**, la qual el minteja i ho assigna a un compte específic (l'indicat a **Remix**), validant que només el propietari o adreces autoritzades puguin executar aquesta acció.



Aquesta funció consumirà **1 EPI** (l'assignat a la variable **rate**) i crearà l'NFT amb les metadades assignades i amb **ID = 0**, paràmetre que va augmentant a mesura que es van creant més NFTs des del mateix contracte. Aquests seran visibles tant des de **MetaMask** com des d'**OpenSea**:



How I Met Your Mother

Un grupo de amigos pasa años haciendo líos y buscando el amor, mientras Ted les cue... [Mostrar más](#)

Dirección de contrato 0x54C9B...8258F

ID de token 0

Estándar de tokenes ERC721

Visualització des de **MetaMask**

How I Met Your Mother SeriesNFTs

Visualització des d'**OpenSea**

Aquesta és la transacció generada visualitzada amb **Etherscan**:

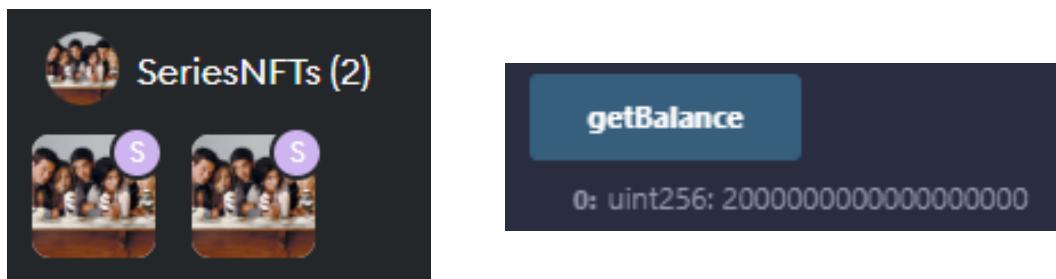
The screenshot shows a transaction details page from Etherscan. Key information includes:

- Transaction Hash: 0xbbba29effddc0a0b3a7cf93926354ed98e310844ed74e526b274a6732d35ef603
- Status: Success
- Block: 7504369 (33 Block Confirmations)
- Timestamp: 6 mins ago (Jan-16-2025 01:38:24 PM UTC)
- Transaction Action: Call **Safe Mint** Function by 0x8eF007c0...5F15E3DEe on 0x54C9B031...5D8f8258F
- From: 0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
- Interacted With (To): 0x54C9B031DCA4Ffe5080066EA81E63eA5D8f8258F
- ERC-20 Tokens Transferred: All Transfers (From 0x8eF007c0...5F15E3DEe To 0x54C9B031...5D8f8258F For 1 ERC-20: EpisodeCoin (EPIS))
- ERC-721 Tokens Transferred: ERC-721 Token ID [0] SeriesNFTs(SNFT) (From 0x00000000...0000000000 To 0x8eF007c0...5F15E3DEe)
- Value: 0 ETH
- Transaction Fee: 0.001132420760054894 ETH
- Gas Price: 8.212255501 Gwei (0.000000008212255501 ETH)

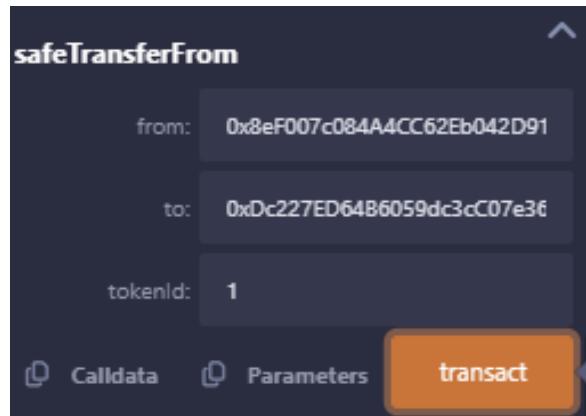
Compra i transferència de NFTs (*buyNFT, safeTransferFrom*)

Una vegada creat l'NFT, si el volguéssim transferir a un altre compte hauríem d'utilitzar la funció **safeTransferFrom** del **ERC721**, que permet transferir-lo de forma segura, assegurant-se que la direcció de destinació pugui gestionar-lo correctament.

Per mostrar un exemple, crearem dos NFTs (els quals tindran un **ID** igual a 0 i 1 corresponentment). Per validar aquesta quantitat creada, podem utilitzar la funció **getBalance**, la qual retorna el valor 2:



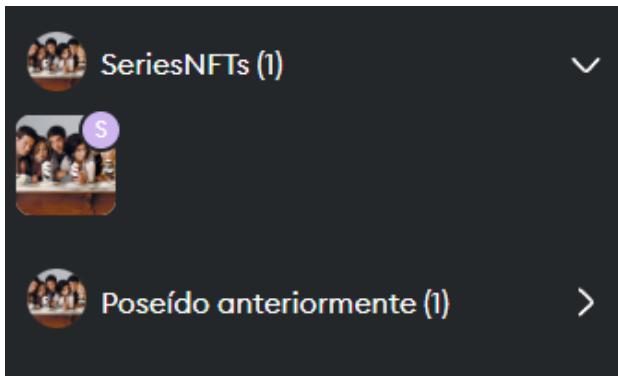
Ara ja podem usar ***safeTransferFrom*** per transferir el token amb ***ID = 1*** des del nostre compte principal a un secundari, com es mostra a la següent imatge:



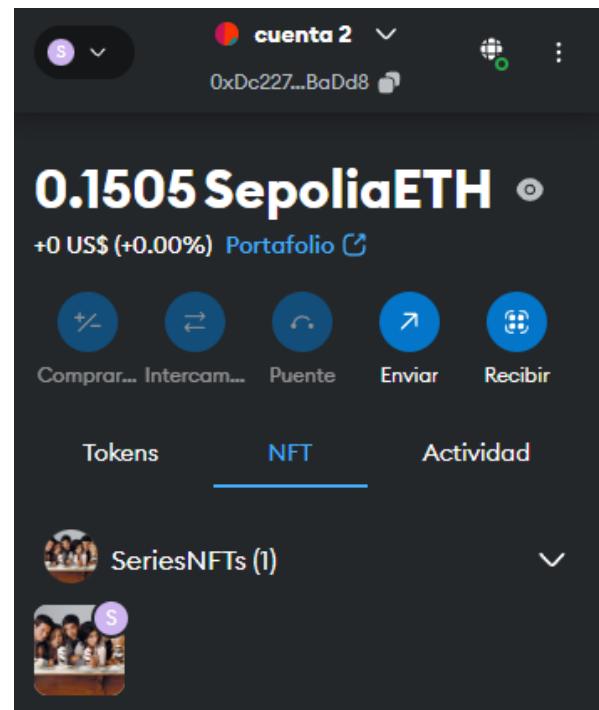
També podem veure la transacció generada:

① Transaction Hash:	0xe9b5e8b1b58c6b147c6ae46f05cddb92ec9a61958f3f190fdb0cedfd372ce8c2
② Status:	Success
③ Block:	7518416 3 Block Confirmations
④ Timestamp:	36 secs ago (Jan-18-2025 12:43:36 PM UTC)
⚡ Transaction Action:	Call Safe Transfer From Function by 0x8eF007c0...5F15E3DEe on 0x5c9101a4...06cf191D4
⑤ From:	0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
⑥ Interacted With (To):	0x5c9101a404AA91107a3a011fA2d96B06cf191D4 ✓
⑦ ERC-721 Tokens Transferred:	 ERC-721 Token ID [1] ● SeriesNFTs(SNFT) From 0x8eF007c0...5F15E3DEe To 0x5c9101a4...06cf191D4
⑧ Value:	0 ETH
⑨ Transaction Fee:	0.00019956715539232 ETH
⑩ Gas Price:	3.15551128 Gwei (0.00000000315551128 ETH)

Aleshores, podem comprovar com l'NFT amb ***ID = 1*** deixa de pertànyer al compte principal i passa a ser del compte secundari:



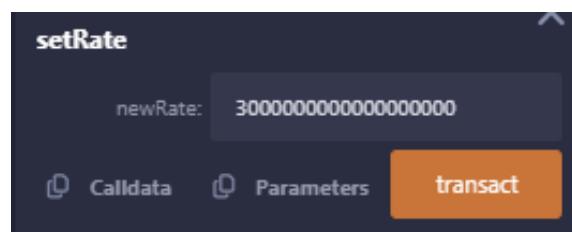
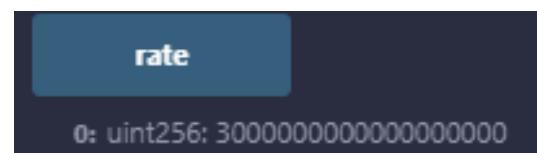
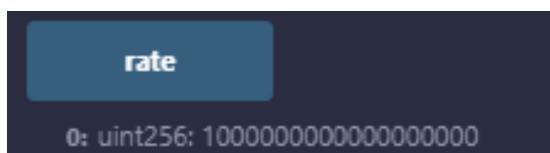
Compte principal



Compte secundari

Si ara ens proposam comprar l’NFT que acabam de transferir, ho podrem aconseguir amb la funció ***buyNFT*** del ERC721.

Per fer més complex l’exemple, es visualitzarà el preu actual de l’NFT (**1 EPI**) amb la funció **rate** i posteriorment s’augmentarà aquest valor (a **3 EPIS**) amb la funció **setRate** per així aconseguir un major benefici per ell.



La següent passa és utilitzar la funció ***buyNFT***, indicant com a paràmetre l'***ID*** del token que volem comprar, en aquest cas l'**1** (el **0** no es podria, ja que som el propietari):

The screenshot shows the Remix Ethereum IDE interface. On the left, there is a code editor with the following Solidity code:

```
function buyNFT(uint256 tokenId) public {  
    require(tokenId <= 10);  
    require(msg.value == 2.97 ether);  
    require(tokenId != 0);  
    require(tokenId != owner);  
    require(tokenId != address(this));  
  
    SeriesNFTs(tokenId).transferFrom(owner, msg.sender);  
    payable(owner).transfer(msg.value);  
}
```

Below the code editor, there are two buttons: "Calldata" and "Parameters". To the right of these buttons is an orange "transact" button. The "Parameters" field contains the value "1".

On the right side of the interface, there is a summary of the transaction details:

- Cambios estimados**: Shows a gas fee of - 2,97 and a recipient address of 0x649Ad...Fbf90.
- Usted recibe**: Shows a reward of + #1 SeriesNFTs.
- Solicitud de**: Shows the request is from remix.ethereum.org.
- Interactuando con**: Shows interaction with SeriesNFTs.

Podem veure la transacció amb més detall a [Etherscan](#):

The screenshot shows the Etherscan transaction details page for the transaction hash 0xa23bd1a17c0511930cda2df828b2fa1d15ae14903daadea8f0a3997e9267fb4a. The page displays the following information:

- Transaction Hash:** 0xa23bd1a17c0511930cda2df828b2fa1d15ae14903daadea8f0a3997e9267fb4a
- Status:** Success
- Block:** 7518433 (2 Block Confirmations)
- Timestamp:** 29 secs ago (Jan-18-2025 12:47:00 PM UTC)
- Transaction Action:** Call Buy NFT Function by 0x8eF007c0...5F15E3DEe on 0x5c9101a4...06cf191D4
- From:** 0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
- Interacted With (To):** 0x5c9101a404AAd91107a3a011fA2d96B06cf191D4
- ERC-20 Tokens Transferred:**
 - From 0x8eF007c0...5F15E3DEe To 0x8eF007c0...5F15E3DEe For 0.03 ERC-20: EpisodeCoin (EPIS)
 - From 0x8eF007c0...5F15E3DEe To 0xDc227ED6...3687BaDd8 For 2.97 ERC-20: EpisodeCoin (EPIS)
- ERC-721 Tokens Transferred:**
 - ERC-721 Token ID [1] SeriesNFTs(SNFT)
From 0xDc227ED6...3687BaDd8 To 0x8eF007c0...5F15E3DEe
- Value:** 0 ETH
- Transaction Fee:** 0.000337384528410852 ETH
- Gas Price:** 3.348264543 Gwei (0.000000003348264543 ETH)

Com podem observar, el cost és de **3 EPIS** però aquest es divideix en dues parts:

- l'**1% (0.03 EPIS)** representa la **comisió** que es queda el propietari del contracte **ERC721**, és a dir, l'adreça que l'ha desplegat.
- el **99% (2.73 EPIS)** representa el valor de la compra que va destinat al antic propietari de l'NFT, és a dir, l'adreça que el posseïa abans de la compra.

Com veiem a continuació, el compte principal (el qual tenia un total de **92 EPIS** abans de realitzar la **compra**) ara té **89,03 EPIS** degut a que se li ha restat els **3 EPIS** que costa l'NFT però també se li ha sumat els **0.03 EPIS** de la comisió que li pertany. El compte secundari (el qual tenia un total de **0 EPIS** abans de realitzar la **venta**) ara té **2.97 EPIS**, els **3 EPIS** del preu total menys els **0.03 EPIS** que pertanyen al propietari del contracte.

Resumint, el compte principal paga l'NFT i rep l'1% de la comisió que li pertany per desplegar el contracte, mentre que el secundari rep el 99% corresponent al pagament.

0.8853 SepoliaETH •
+0 US\$ (+0.00%) Portafolio [🔗](#)

Comprar... Intercambiar... Puente Enviar Recibir

Tokens NFT Actividad

Sepolia [🔗](#) 89,03 EPIS

Compte principal

0.1005 SepoliaETH •
+0 US\$ (+0.00%) Portafolio [🔗](#)

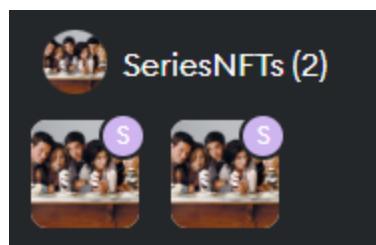
Comprar... Intercambiar... Puente Enviar Recibir

Tokens NFT Actividad

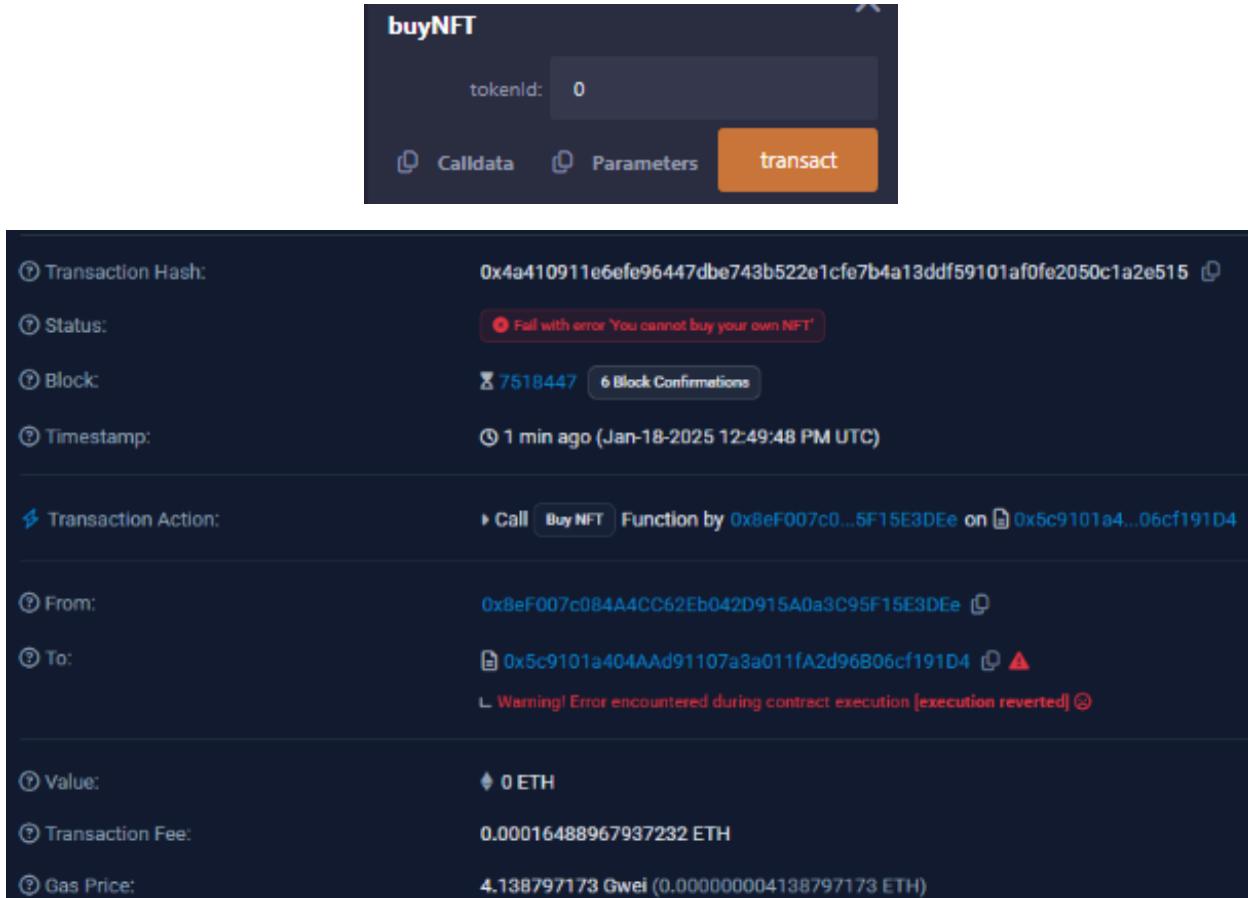
Sepolia [🔗](#) 2,97 EPIS

Compte secundari

Finalment, podrem veure que ambdós NTFs tornen a estar en possessió del compte principal, el qual primer ha transferit un d'ells i després l'ha comprat:



Si haguéssim intentat comprar l'NFT amb **ID = 0**, el qual pertany a l'adreça que està intentant fer la compra, és a dir, es vol comprar un NFT que ja és nostre, hauria aparegut un **error** com el que es mostra a la següent transferència fallida:

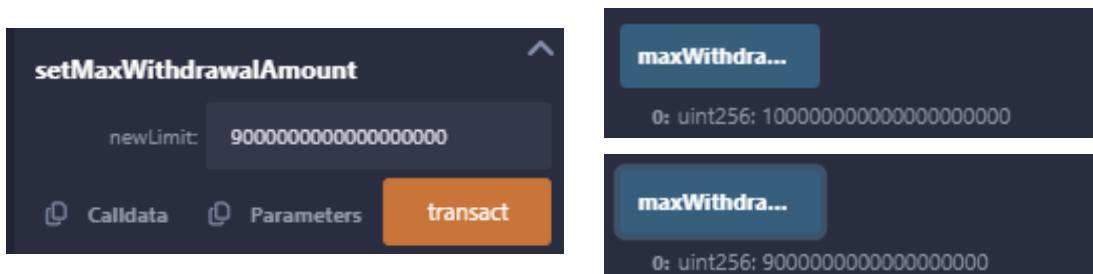


The screenshot shows a transaction details page for a failed NFT purchase. At the top, there's a small interface for a "buyNFT" function with a "tokenId: 0" input field and a "transact" button. Below this, the main transaction details are listed:

- Transaction Hash:** 0x4a410911e6efe96447dbe743b522e1cfe7b4a13ddf59101af0fe2050c1a2e515
- Status:** Failed with error "You cannot buy your own NFT"
- Block:** 7518447 (6 Block Confirmations)
- Timestamp:** 1 min ago (Jan-18-2025 12:49:48 PM UTC)
- Transaction Action:** Call Buy NFT Function by 0x8eF007c0...5F15E3DEe on 0x5c9101a4...06cf191D4
- From:** 0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
- To:** 0x5c9101a404AAD91107a3a011fA2d96B06cf191D4 (Warning! Error encountered during contract execution [execution reverted])
- Value:** 0 ETH
- Transaction Fee:** 0.00016488967937232 ETH
- Gas Price:** 4.138797173 Gwei (0.000000004138797173 ETH)

Retirada de fons (*withdraw*)

Si un usuari vol retirar monedes del contracte, hauria d'utilitzar la funció **withdraw** del ERC721. Amb la funció **setMaxWithdrawalAmount** podem determinar la quantitat màxima a retirar (per defecte **100 EPIS**). En aquest cas l'hem canviada a **9 EPIS**, amb la funció **maxWithdrawal** podem visualitzar aquesta quantitat:



The screenshot shows a blockchain interface with two main sections. On the left, there's a function call dialog for "setMaxWithdrawalAmount" with a "newLimit" parameter set to "90000000000000000000000000000000". Below it is a "transact" button. On the right, there are two results for the "maxWithdrawal" function, both showing a value of "0: uint256: 90000000000000000000000000000000".

Actualment tenim **99 EPIS** al contracte **ERC20** i només **50 EPIS** aprovats perquè el contracte **ERC721** els pugui gastar. A continuació intentarem retirar diverses quantitats de monedes per comprovar si ens deixa retirar-les del **ERC20** o si, per altra banda, la transferència falla:

1) Intentam retirar 60 EPIS (més de la quantitat aprovada)

The screenshot shows a transaction details page for a failed withdrawal attempt. At the top, there's a 'withdraw' interface with an amount input set to 60000000000000000000000000000000. Below it, a 'transact' button is highlighted in orange. The main section displays the following information:

- Transaction Hash: 0xe48e45c8c5dba66263776a55e63044ffdd74aef50466abb2e8c1b5543c5480c7
- Status: Failed with error 'Not enough approved tokens to withdraw'
- Block: 7525681 (2 Block Confirmations)
- Timestamp: 24 secs ago (Jan-19-2025 01:00:12 PM UTC)
- Transaction Action: Withdraw (Function by 0x8eF007c0...5F15E3DEe on 0x4325292E...d9b8cE7c2)
- From: 0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
- To: 0x4325292E0D245962e0BC3dBCE00A50ed9b8cE7c2 (✓)
- Value: 0 ETH
- Transaction Fee: 0.000402252337246 ETH
- Gas Price: 10.57167772 Gwei (0.00000001057167772 ETH)

En aquest cas la transferència **no** es podria realitzar pel fet que no hi ha prou **EPIS** aprovats (funció **approve**) perquè el contracte els pugui utilitzar (faltaria aprovar 10 més).

2) Intentam retirar 10 EPIS (menys que la quantitat aprovada però més del límit assignat)

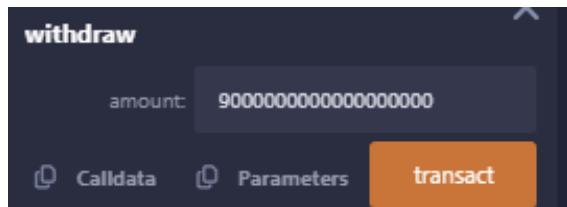
The screenshot shows a transaction details page for a successful withdrawal attempt. The withdrawal interface at the top has an amount input set to 10000000000000000000000000000000. Below it, a 'transact' button is highlighted in orange. The main section displays the following information:

- Transaction Hash: 0xe48e45c8c5dba66263776a55e63044ffdd74aef50466abb2e8c1b5543c5480c7
- Status: Success
- Block: 7525681 (2 Block Confirmations)
- Timestamp: 24 secs ago (Jan-19-2025 01:00:12 PM UTC)
- Transaction Action: Withdraw (Function by 0x8eF007c0...5F15E3DEe on 0x4325292E...d9b8cE7c2)
- From: 0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
- To: 0x4325292E0D245962e0BC3dBCE00A50ed9b8cE7c2 (✓)
- Value: 0 ETH
- Transaction Fee: 0.000402252337246 ETH
- Gas Price: 10.57167772 Gwei (0.00000001057167772 ETH)

① Transaction Hash:	0x7e22193f6d758969e2f4c45ac0f0f4c45b1384be0531e9bb6c23de3b0d7a0fd
① Status:	Fail with error 'Withdrawal exceeds the limit'
① Block:	7525675 3 Block Confirmations
① Timestamp:	32 secs ago (Jan-19-2025 12:59:00 PM UTC)
⚡ Transaction Action:	Call Withdraw Function by 0x8eF007c0...5F15E3DEe on 0x4325292E...d9b8cE7c
① From:	0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
① To:	0x4325292EeD245962e0BC3dB Ce00A50ed9b8cE7c2
① Value:	0 ETH
① Transaction Fee:	0.000426262921990044 ETH
① Gas Price:	10.617818014 Gwei (0.000000010617818014 ETH)

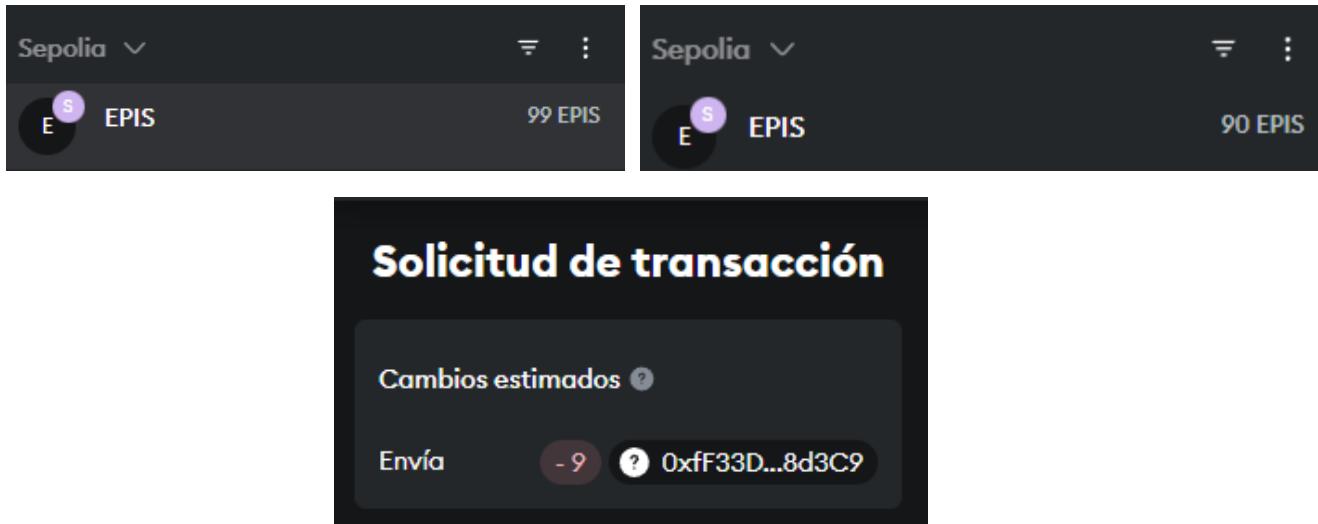
En aquest cas la quantitat és menor a l'aprovada, però la transferència **no** es podria realitzar pel fet que no hi ha suficients **EPIS** assignats com a límit per retirar (faltaria assignar 1 més).

3) Intentam retirar 9 **EPIS** (menys que la quantitat aprovada i que el límit assignat)



① Transaction Hash:	0x1a66797a85cf137ea152c8fd8a1765910f0b7a929855d5d599d3eb1abd38071d
① Status:	Success
① Block:	7525993 2 Block Confirmations
① Timestamp:	17 secs ago (Jan-19-2025 02:02:36 PM UTC)
⚡ Transaction Action:	Call Withdraw Function by 0x8eF007c0...5F15E3DEe on 0x43A09F52...b40656984
① From:	0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
① Interacted With (To):	0x43A09F52C100b83207070Cd9DAF9703b40656984
① ERC-20 Tokens Transferred:	All Transfers Net Transfers
	From 0x8eF007c0...5F15E3DEe To 0x43A09F52...b40656984 For 9 ERC-20: EpisodeCoin (EPIS)
① Value:	0 ETH
① Transaction Fee:	0.000580225084721681 ETH
① Gas Price:	7.761896977 Gwei (0.000000007761896977 ETH)

En aquest cas la transacció **sí** es podrà realitzar amb èxit pel fet que no se supera ni la quantitat aprovada ni el límit de retirada assignat.



Cal explicar que quan un usuari decideix fer una retirada, els **tokens ERC20 es transfereixen del contracte ERC20 i es guarden temporalment al contracte ERC721** per al seu futur ús. Aquesta funcionalitat podria ser útil en un possible escenari en el qual els usuaris volguessin convertir els seus tokens **ERC20** en alguna altra moneda, com per exemple, euros, o utilitzar-los en algun tipus de mercat d'intercanvi. Així que, tot i que la retirada de tokens es gestiona, el flux final de la conversió o utilització d'aquests tokens, com la seva possible integració amb un mercat real, no es duu a terme en aquesta versió del contracte.

Configuració de la Whitelist

La **whitelist** és una llista d'adreces autoritzades per realitzar accions restringides al contracte, com mintejar NFTs o retirar tokens. El propietari pot afegir o eliminar adreces segons sigui necessari.

Amb la funció **whitelisted** comprovam si una adreça es troba en ella o no:

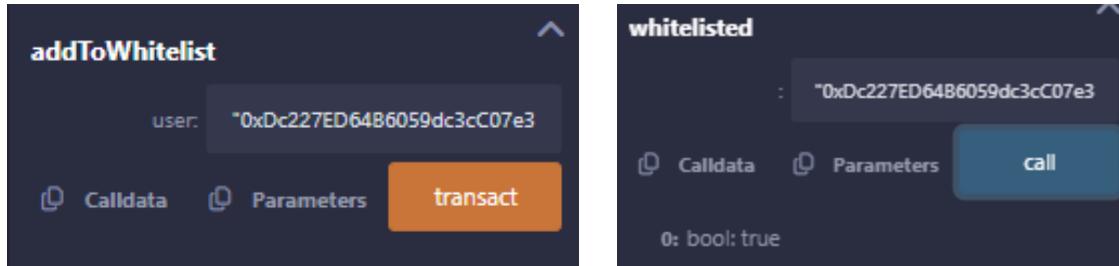
whitelisted
: 0x8eF007c084A4CC62Eb042D91
Calldata Parameters call
0: bool: true

Compte principal

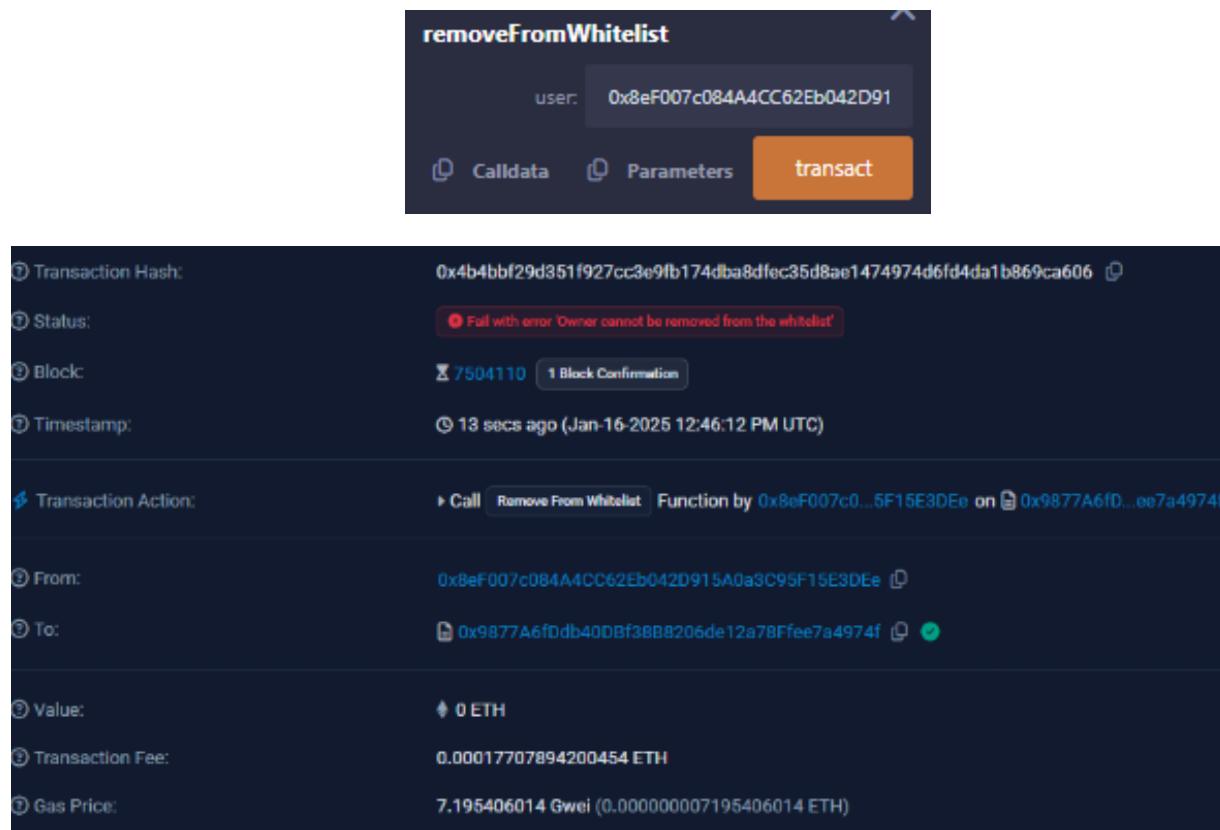
whitelisted
: "0xDc227ED64B6059dc3cC07e3
Calldata Parameters call
0: bool: false

Compte secundari

Amb la funció ***addToWhitelist*** el propietari del contracte pot afegir adreces per donar-lis més privilegis, afegirem el compte secundari:



Amb la funció ***removeFromWhitelist*** el propietari del contracte pot eliminar adreces per llevar-los els privilegis, si intentam eliminar el propietari del contracte (el compte principal) la transacció fallarà com es veu a la següent imatge:



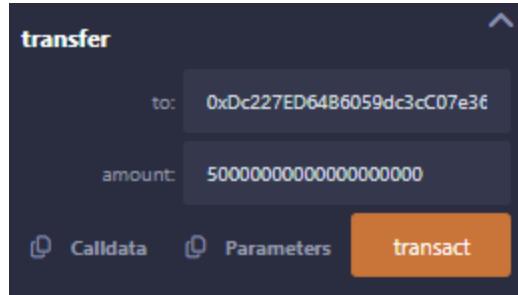
② Transaction Hash:	0x4b4bbf29d351f927cc3e9fb174dba8dfec35d8ae1474974d6fd4da1b869ca606
② Status:	Fail with error 'Owner cannot be removed from the whitelist'
② Block:	7504110 1 Block Confirmation
② Timestamp:	13 secs ago (Jan-16-2025 12:46:12 PM UTC)
↳ Transaction Action:	Call Remove From Whitelist Function by 0x8eF007c0...5F15E3DEe on 0x9877A6fD...ee7a4974f
② From:	0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
② To:	0x9877A6fDdb40DBf38B8206de12a78Ffee7a4974f
② Value:	0 ETH
② Transaction Fee:	0.00017707894200454 ETH
② Gas Price:	7.195406014 Gwei (0.000000007195406014 ETH)

Altres funcionalitats del ERC20

A continuació es mostraran altres funcions menys importants que les anteriors però amb diverses funcionalitats:

1) Funció *transfer*

La utilitzarem per transferir **50 EPIS** del compte principal (amb actualment **100 EPIS**) al compte secundari (amb **0 EPIS**):

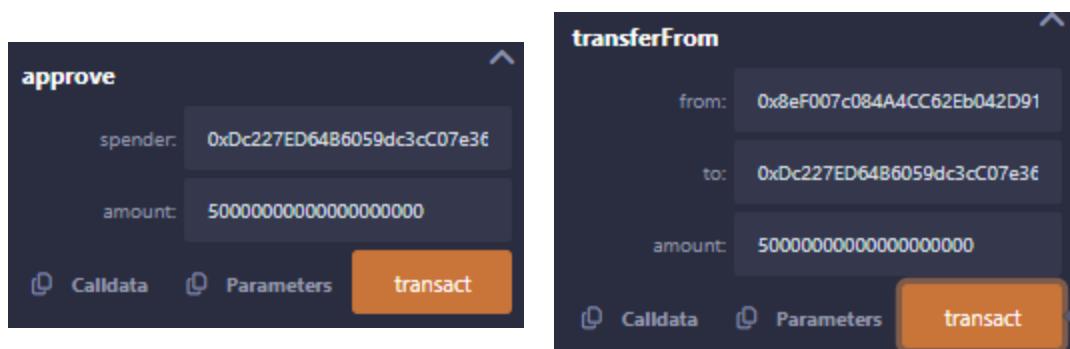


Aquesta és la transacció corresponent:

① Transaction Hash:	0xe091269af3851ed63e27df4c82be8b80e96297d9b71bb0ac1ee30707c397508
① Status:	Success
① Block:	7503788 4 Block Confirmations
① Timestamp:	1 min ago (Jan-16 2025 11:31:48 AM UTC)
⚡ Transaction Action:	Call Transfer Function by 0x8eF007c0...5F15E3DEe on 0x69D58298...6846A4d47
① From:	0x8eF007c084A4CC62Eb042D915A0a3C95F15E3DEe
① Interacted With (To):	0x69D58298Ccf521143906B7B42C45fc46846A4d47
① ERC-20 Tokens Transferred:	All Transfers Net Transfers
	From 0x8eF007c0...5F15E3DEe To 0xDc227ED6...3687BaDd8 For 50 ● ERC-20: EpisodeCoin (EPIS)
① Value:	0 ETH
① Transaction Fee:	0.00030320701184064 ETH
① Gas Price:	5.795241052 Gwei (0.00000005795241052 ETH)

2) Funció *safeTransfer*

Aquesta és molt pareguda a l'anterior, amb la diferència que el sender ha d'obtenir prèviament permís per fer la transacció mitjançant la funció *approve* (en la anterior no calia). Hem de donar prou *allowance* a l'adreça que envia les monedes perquè la transacció sigui exitosa.



3) Funció *balanceOf*

Amb aquesta podem veure la quantitat de **EPIS** que hi ha a cada compte, després de la transacció anterior hi haurà concretament **50 EPIS** a cada una:

balanceOf
account: 0x8eF007c084A4CC62Eb042D91
call
0: uint256: 50000000000000000000000000000000

Compte principal

balanceOf
account: 0xDc227ED6486059dc3cC07e36
call
0: uint256: 50000000000000000000000000000000

Compte secundari

Si utilitzam aquesta funció per a un compte diferent, el qual **no té cap EPI**, es retornarà que no té cap:

balanceOf
owner: 0x6d17434dAcEbBc617f579D2a
call
0: uint256: 0

Compte terciari

4) Funció *allowance*

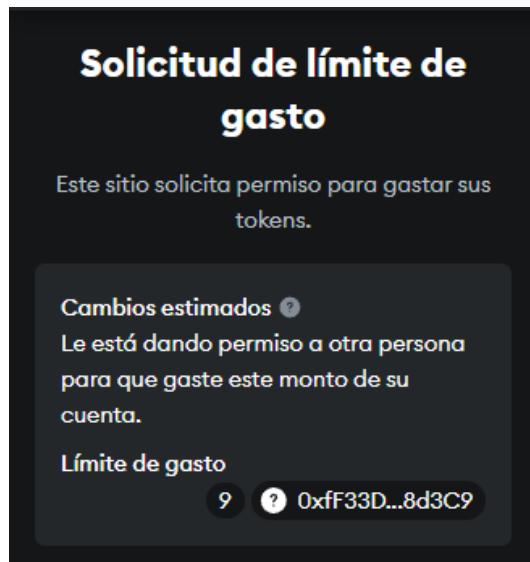
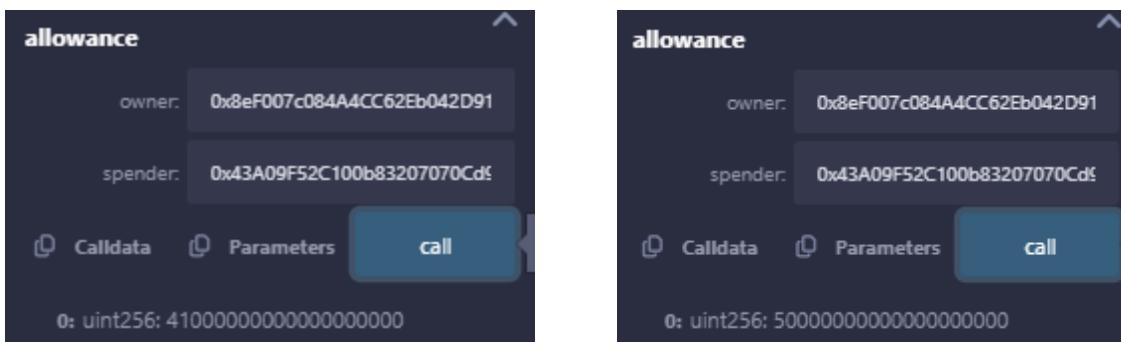
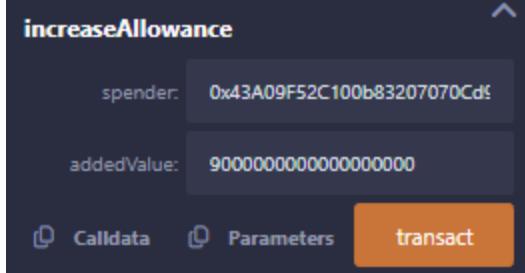
A diferència de la funció anterior que mostra el total de monedes que hi ha per compte, aquesta mostra les monedes que cada compte pot gastar, és a dir, les aprovades per la funció **approve**. En aquest cas, s'ha aprovat l'utilització de **50 EPIS** (tot i haver-hi **100 EPIS** en total com es veu al *balanceOf*)

allowance
owner: 0x8eF007c084A4CC62Eb042D91
spender: 0x4325292EeD245962e08C3dB4
call
0: uint256: 50000000000000000000000000000000

balanceOf
account: 0x8eF007c084A4CC62Eb042D91
call
0: uint256: 10000000000000000000000000000000

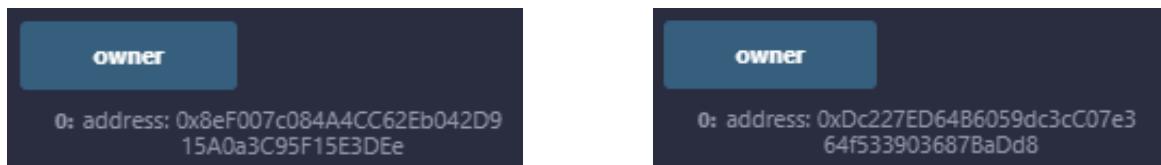
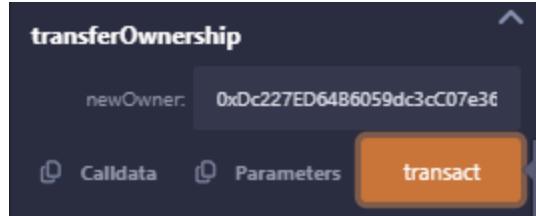
5) Funcions *increaseAllowance* i *decreaseAllowance*

Aquestes permeten pujar o baixar la quantitat de monedes aprovades (mitjançant la funció *approve*) a un compte. En aquest exemple tenim accessibles **41 EPIS** però volem incrementar el valor a **50**, afegint **9 EPIS**:



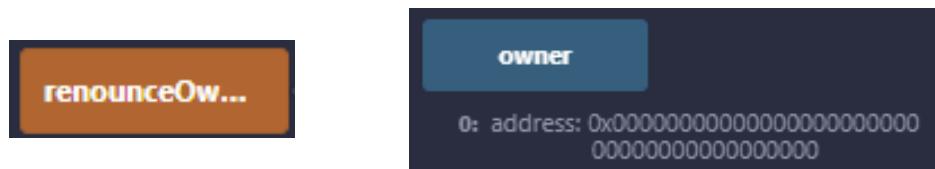
6) Funcions *owner* i *transfetOwnership*

Aquestes funcions es permeten saber qui és el propietari del contracte **ERC20** i transferir aquesta propietat a un altre compte. En l'exemple la transferim des del compte principal al secundari:



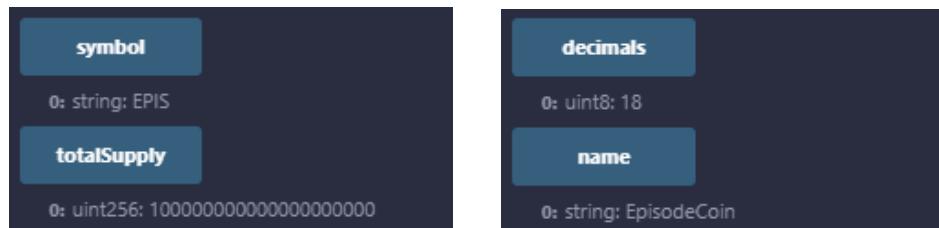
7) Funcions *renounceOwnership*

Aquesta ens permet renunciar a la propietat del contracte, el qual passa a no tenir propietari:



8) Funcions *decimals*, *name*, *symbol* i *totalSupply*

Aquestes funcions de lectura ens retornen valors útils com el total de monedes generades, el seu nom, símbol i número de decimals:

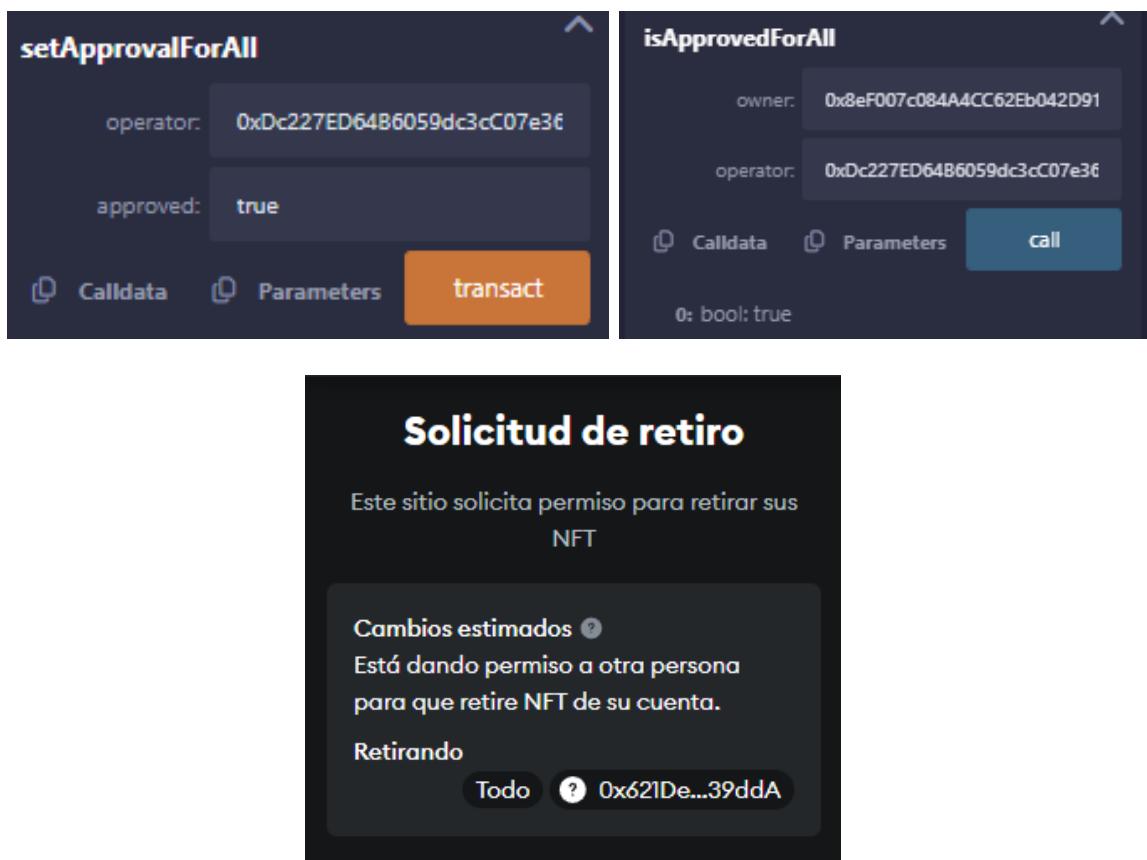


Altres funcionalitats del ERC721

Es mostraran altres funcions amb menys importància que les anteriors però amb diverses funcionalitats:

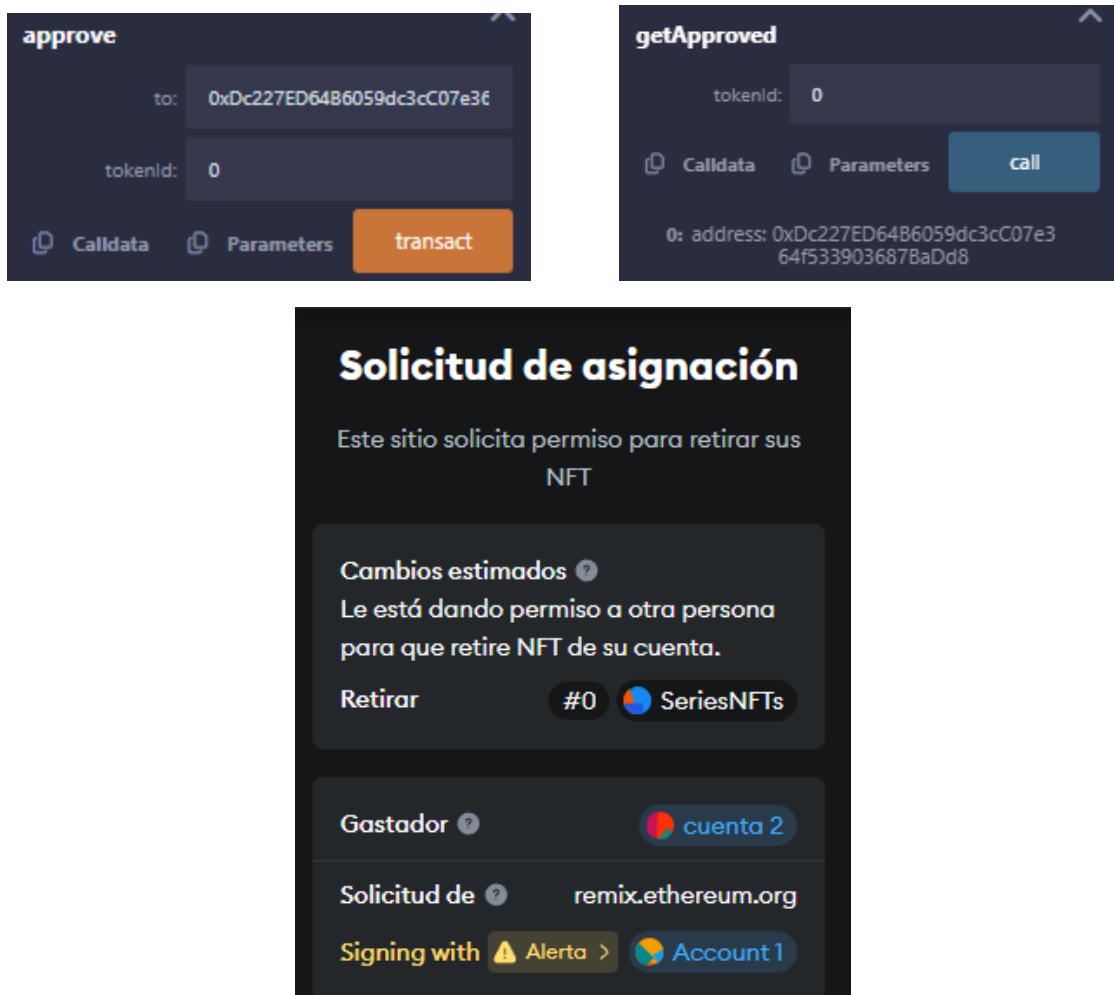
1) Funcions *setApprovalForAll* i *isApprovedForAll*

La funció *setApprovalForAll* permet autoritzar un operador perquè gestioni **tots els NFTs** d'un usuari, mentre que *isApprovedForAll* verifica si un operador té aquesta autorització. A l'exemple autoritzam al compte secundari i ho comprovem:



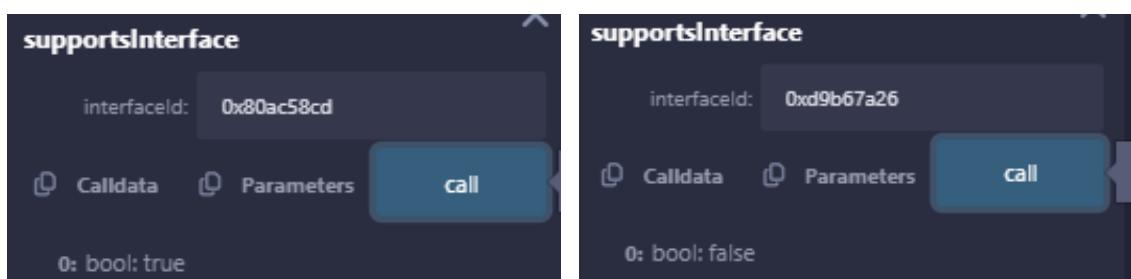
2) Funció *approve* i *getApproved*

La funció **approve** del ERC721 (no la utilitzada anteriorment que pertany al **ERC20**) dona permís a una adreça per gestionar un **NFT específic** (donat el seu **ID**), i **getApproved** comprova quina adreça té aquest permís. A l'exemple donam permís al compte secundari i després ho comprovem:



3) Funció `supportsInterface`

Aquesta funció permet verificar si un contracte implementa una interfície específica definida per un identificador (`interfaceId`). Aquest contracte **sí** soporta el **ERC721** (amb `interfaceId 0x80ac58cd`) però **no** el **ERC1155** (amb `interfaceId 0xd9b67a26`) com podem observar:



4) Funcions *balanceOf* i *ownerOf*

La funció ***balanceOf*** retorna la quantitat de NFTs que posseeix una adreça concreta. Per altra banda, ***ownerOf*** retorna l'adreça propietària d'un token específic identificat pel seu *ID*.

The image shows two side-by-side screenshots of a blockchain tool's interface. The left screenshot displays the **balanceOf** function. It has a parameter input field labeled "owner:" containing the address "0x8eF007c084A4CC62Eb042D9". Below the input fields are buttons for "Calldata" and "Parameters", and a prominent blue "call" button. Underneath the call button, the output is shown as "0: uint256: 1". The right screenshot shows the **ownerOf** function. It also has an "owner:" input field with the same address. Below it is a dropdown menu set to "0" which reveals the output: "0: address: 0x8eF007c084A4CC62Eb042D9 15A0a3C95F15E3DEe".

5) Funció *name*, *rate*, *symbol*, *tokenAddress* i *baseTokenURI*

Aquestes funcions de lectura es retornen valors com el nom del contracte, el seu símbol, el preu d'un NFT, l'adreça del contracte ERC20 enllaçat i l'URL de les metadades.

The image shows four function calls in a blockchain tool. The first call is **baseTokenURI**, which returns a string URL: "0: string: https://gateway.pinata.cloud/ipfs/bafkreiabmwaxrexdh3a3qwtp6zdom3qjtitmsqzoldw5c53zv5ledmzqai". The second call is **symbol**, returning "0: string: SNFT". The third call is **tokenAddress**, returning "0: address: 0x3D9543811900f220aeE23f66E5b164E6C5d174D2". The fourth call is **rate**, returning "0: uint256: 10000000000000000000000000000000". Each function call has its own input field and a "call" button.

Conclusions

En aquest projecte hem implementat un sistema complet basat en tecnologia *blockchain* que integra un contracte **ERC20** amb un **ERC721** per a la **creació, gestió i transferència d'NFTs**. S'ha desenvolupat una moneda pròpia (**EPIS**) per realitzar les transaccions, així com funcionalitats com la compra i venda d'NFTs, la configuració d'una whitelist i la retirada de fons.

Amb l'ús de **Metamask** i **Etherscan**, hem validat i monitoritzat les transaccions per assegurar la transparència i seguretat. Les funcions desenvolupades permeten una gestió senzilla i eficaç, assegurant una experiència d'usuari completa.

Aquest projecte exemplifica l'ús pràctic i potencial dels contractes intel·ligents. A més, estableix les bases per a futures millores i ampliacions que poden fer del sistema una plataforma encara més versàtil.

Futures Millores

Aquestes són algunes de les futures millores que es podrien aplicar al projecte:

1. Funcionalitat de Subhastes:

Es podria implementar un sistema de subhastes on els usuaris poguessin fer ofertes per un NFT en lloc de comprar-lo per un preu fix. Això aportaria més flexibilitat i dinamisme a la plataforma.

2. Comissions Dinàmiques:

Actualment, la comissió està fixada en un **1%** per cada compra d'un NFT. Una millora seria permetre que el propietari del contracte ajustés aquest percentatge de manera dinàmica, adaptant-lo segons la demanda o el tipus d'NFT.

3. Conversió a altres Monedes:

Permetre l'intercanvi de les monedes **ERC20** per una altra moneda, com euros, després de realitzar el ***withdraw***. Això faria que el sistema fos més atractiu per a usuaris que vulguin monetitzar els seus guanys fora de la *blockchain*.

4. Recompenses per Participació:

Implementar un sistema de recompenses que doni tokens **ERC20** als usuaris per la seva activitat dins la plataforma, com ara la creació i venda d'NFTs. Això augmentaria el compromís dels usuaris.

5. Integració amb altres Marketplaces:

Facilitar la interoperabilitat amb altres plataformes de compra i venda d'NFTs per augmentar la visibilitat i el valor dels tokens creats.

Aquestes millores ajudarien a fer el projecte més complet, atractiu i competitiu dins el mercat de la **blockchain** i els NFTs.

Referències

Aquestes són les referències usades pel desenvolupament del projecte:

1. [**Hola Mundo NFT - dev.to:**](#)

Aquesta guia introduceix els conceptes bàsics de la creació d'un contracte intel·ligent **ERC721**. Ha estat útil per entendre com desplegar el contracte, visualitzar-lo des de **OpenSea** i afegir metadades utilitzant la xarxa **IPFS**.

2. [**OpenZeppelin Documentation - ERC721:**](#)

La documentació oficial d'**OpenZeppelin** ha ajudat en la selecció de biblioteques necessàries pel projecte i a l'hora d'entendre les funcions bàsiques del contracte.

3. [**Remix IDE - Official Website:**](#)

És l'entorn utilitzat per escriure, compilar i desplegar els contractes intel·ligents. També ha estat fonamental per realitzar proves locals i simular el funcionament dels contractes.

4. [**Etherscan - Guia de Transaccions:**](#)

S'ha utilitzat per visualitzar i verificar les transaccions i per entendre com es gestiona la interacció amb la *blockchain*.

5. [**ERC20 Token Standard - Ethereum Improvement Proposals \(EIP\):**](#)

Aquesta proposta descriu el funcionament del token **ERC20**, ajudant a entendre les seves funcions bàsiques.

6. [**ChatGPT \(OpenAI\):**](#)

Ha estat utilitzada com a suport en la redacció i estructuració de la documentació del projecte. Ha ajudat a generar explicacions clares i organitzar la informació de manera coherent.

Annex

A continuació es comparteixen els enllaços a les metadades emmagatzemades en **Pinata**, són el paràmetre **baseTokenURI** que s'ha d'introduir a l'hora de desplegar el contracte **ERC721**:

Nom de la sèrie	baseTokenURI
Friends	https://gateway.pinata.cloud/ipfs/bafkreifodvq4icnmflhdqovj7nltyu57ahfs7fu3fegi3wuevs3kebkrmq
How I Met Your Mother	https://gateway.pinata.cloud/ipfs/bafkreiabmwaxrexdh3a3qwtp6zdom3qjtitmsqzoldw5c53zv5ledmzqai
Breaking Bad	https://gateway.pinata.cloud/ipfs/bafkreibljlx6bhtq6nvrr6s7fmbj5nateh2yekflf3vkmxrktu6tsohiqi
Stranger Things	https://gateway.pinata.cloud/ipfs/bafkreidurri6voa2y2hqoqtkzw7tyze2ivcvglbg2sxfzurkusugll422q
La Casa de Papel	https://gateway.pinata.cloud/ipfs/bafkreigydiesbiotlorz5u27mkk4upxbvycyyuqee3t576mecgqh4nyp2u
The Walking Dead	https://gateway.pinata.cloud/ipfs/bafkreibvvufhpwuxnutiet6rkkfvorwrazfxcedk2vz5usmfpgqkohuiu
Los Simpson	https://gateway.pinata.cloud/ipfs/bafkreie7urp7tjr2rnklqzsk53ebvx5re6gnxkglfrfo44d7og6vzv6nxu
Squid Games	https://gateway.pinata.cloud/ipfs/bafkreih5qlgatamak3a4gpf3fjiir7dflqnfdih3s7xz3npn32swfd6iky