Approbation Token Gating dApp T.P

Julián Vázquez SIB-2025s1 UNQ

1. Objetivo General

Desarrollar una aplicación descentralizada (dApp) que implemente un sistema de token gating educativo basado en el estándar ERC-1155, permitiendo dos flujos diferenciados: uno para estudiantes y otro para docentes. Se validan NFTs previos para permitir la emisión y recepción de un segundo NFT que acredita la aprobación de la cursada.

2. Actores del Sistema

Alumno	Se conecta con su wallet, presenta NFTs previos y emite un **Proof of Work NFT**. Recibe, si se dan las condiciones, un **Approval NFT** que certifica la aprobación del curso.
Docente	Recibe NFTs de los alumnos. Puede emitir un **Approval NFT**.

3. Flujos Funcionales

Flujo 1 – Validación de cumplimiento – Alumno

1. Se conecta a la dApp vía Metamask y se verifica que no pertenezca a un docente, ya que es de uso exclusivamente para alumnos.
2. Se listan sus NFTs bajo contrato 0x1fee62...f91d.
3. Se validan 3 condiciones:
   - Posee al menos 10 NFTs del contrato.
   - Fueron recibidos antes del 28/04/2025.
   - No fueron transferidos:
     * Fue recibido desde `address(0)`, lo que indica que fue minteado directamente.

* No debe existir ningún evento posterior donde ese NFT haya salido de la wallet.

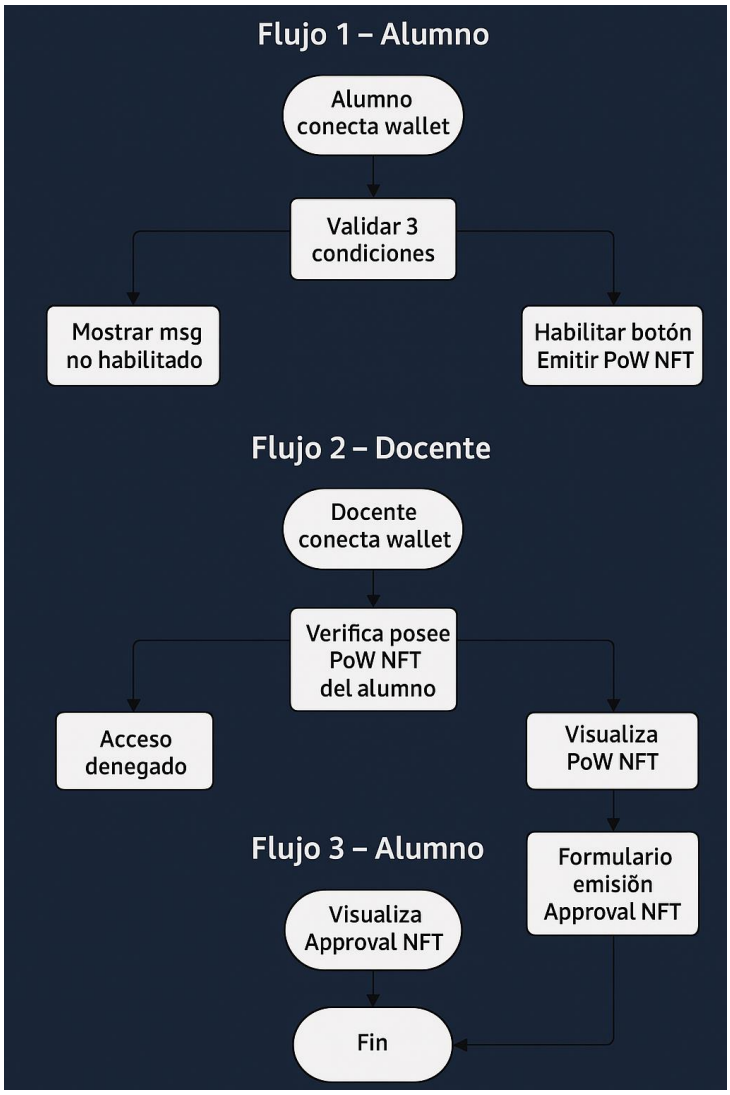
4. Si se cumplen las 3 condiciones, se habilita un botón para emitir y transferir un **Proof of Work NFT** automáticamente a ambos docentes en una sola transacción (mint + transfer).

Flujo 2 – Docente

1. Se conecta con wallet vía Metamask y se valida que pertenece a los docentes, caso contrario no se accede a esta parte.
2. Se verifica que el docente posea al menos un **Proof of Work NFT** que haya sido emitido por el alumno cuya wallet haya sido ingresada como input.
3. Si la validación es correcta visualiza el **Proof of Work NFT** recibido y sus datos on-chain.
4. Se habilita el formulario para emitir un **Approval NFT** que incluye como campos on-chain: alumno, nota, comentario, y se transfiere a la wallet del alumno mediante un botón que hace el mint+transfer del mismo.

Flujo 3 – Estado del curso

1. Se conecta con wallet vía Metamask y se valida que pertenece a los docentes, caso contrario no se accede a esta parte.'
2. El alumno podrá visualizar el/los **Approval NFT** recibidos de cada profesor para verificar si efectivamente aprobó o no el curso. En caso de no encontrarlos aún se redirige a la pestaña de "Validación de cumplimiento – Alumno"



4. Infraestructura Técnica

FRONTEND

- React + Vite + Tailwind.
- Ethers.js para conexión Web3.
- ABI del contrato ERC-1155.
- Componente <NFTCard> reutilizable para visualización de ambos tipos de NFT

BACKEND

No se requiere backend persistente. Toda la lógica de emisión, validación y consulta de datos está contenida on-chain en los contratos inteligentes.

5. Smart Contracts

Dos contratos ERC-1155

Proof of Work NFT	NFT emitido por alumno a profesores
Approval NFT	NFT emitido por profesor al alumno

Estructuras y funciones requeridas (Proof of Work NFT):

Metadata fija

- **`name`**: `Proof of Work NFT.`
- **`description`**: `Token de prueba de trabajo y cumplimiento de los requisitos para la aprobación del Seminario de Introducción a Blockchain, dictado en la Universidad Nacional de Quilmes.`
- **`image`**: `https://i.imgur.com/cjnFZ9m.png`

Datos variables on-chain:

- **`fecha`**: `Fecha de emisión del NFT.`
- **`alumno`**: `Nombre del alumno.`
- **`emisor`**: `Wallet que emitió el NFT.`
- ***`PoF`***: `La lista` **`PoFEntry`** `con los datos de los 10 NFT de cursada.`

```
struct PoFEntry {
    uint256 id;
    address contractAddress;
}

struct TPData {
    string fecha;
    string alumno;
    address emisor;
    PoFEntry[] PoF;
}

mapping(uint256 => TPData) private datos;
```

Funciones del estándar ERC-1155:

- `balanceOf(address, uint256)`
  - Devuelve cuántos tokens del tipo `id` tiene la cuenta `account`.

- `uri(uint256 tokenId)`
  - Devuelve la URI del archivo JSON que contiene la metadata del token

Funciones propias:

```
function mintAndTransfer(address receptor, string memory fecha,
string memory alumno, address emisor, PoFEntry[] memory PoF)
```

- Emite (mint) un nuevo **Proof of Work NFT** asociado a un token-id único directamente en la wallet del receptor indicada.
  Guarda en el contrato los datos variables on-chain correspondientes a ese tokenId.

```
function getProofOfWork(uint256 tokenId)
```

- Devuelve los datos representativos de un **Proof of Work NFT** según si Id.

Restricciones lógicas del contrato y condiciones de emisión:

Cada alumno puede emitir un único **Proof of Work NFT** a cada profesor. Esto significa que:

- Solo el alumno puede emitir su propio NFT.
- Cada uno de los PoFEntry reportados deben provenir del contrato asociado a los NFT de Asistencia a Clase.
- Valida que la wallet que emitirá el **Proof of Work NFT** efectivamente tenga cada uno de los 10 NFT de Asistencia a Clase que informó tener.
- Valida que el receptor (a quien se transfiere) el **Proof of Work NFT** no tenga yá uno de estos NFT en su wallet.

Estructuras y funciones requeridas (Approval NFT):

Metadata fija:

- **name**: Approval NFT
- **description**: Certificado de aprobación del Seminario de Introducción a Blockchain de la Universidad Nacional de Quilmes emitido por el equipo docente.
- **image**: https://i.imgur.com/wFFE43t.png

Datos variables on-chain:

- **comentario**: Comentario que el docente desee trasmitir al alumno.
- **nota**: nota de cursada.
- **emisor**: Wallet adress del profesor que emitió el NFT.

```
struct Evaluacion {
    string comentario;
    string nota;
    address emisor;
}

mapping(uint256 => Evaluacion) private evaluaciones;
```

Funciones del estándar ERC-1155:

- `balanceOf(address, uint256)`
  - Se usa desde el frontend para validar posesión de NFTs (profesores y alumnos).

- `uri(uint256 tokenId)`
  - Devuelve la URI del archivo JSON que contiene la metadata del token

Funciones propias:

```
function mintEvaluacion(address alumno, string memory comentario,
string memory nota)
```

- Emite el **Approval NFT** a la adress del alumno indicada.

```
function _poseeProofOfWork(address emisor)
```

- Funcion validadora que indica si la wallet posee ya un **Proof of Work NFT.**

```
function getEvaluacion(uint256 tokenId)
```

- Devuelve los datos representativos de un **Proof of Work NFT** según si Id.

Restricciones lógicas del contrato y condiciones de emisión:

- Valida que el emisor sea un profesor autorizado.
- Verifica que la wallet posea un **Proof Of Work NFT** para poder emitir el **Approval NFT**.

6. Componentes

Componente	Descripción breve
Header	Encabezado fijo con título "Approbation Token Gating dApp" y botones para conectar o desconectar Metamask. Muestra la wallet conectada si está activa.
NFTListPanel	Cards que muestran los NFTs poseídos por la wallet conectada. Renderiza imagen, ID, tema, fecha y estado de transferencia para cada NFT.
CheckPanel	Panel de validación de condiciones para aprobar: (A) tener 10 NFTs de clase, (B) todos antes del 28/05/2025, y (C) que ninguno haya sido transferido. Permite revalidar
MintPanel	Formulario de emisión de un ProofOfWork NFT. Solicita nombre, apellido y fecha; y al confirmar ejecuta la función de minteo. Incluye mensaje preventivo de emisión única por docente.
ProofStatusPanel	Muestra detalles de un NFT de aprobación encontrado, su metadata y permite emitir un NFT de aprobación con nota y comentario. Incluye buscador de contrato y opción de desconexión.
ApprovalStatusPanel	Panel que consulta y lista los NFTs de aprobación del alumno conectado. Si no tiene ninguno, ofrece botón para iniciar validación de los 10 NFTs base.
ValidationFlow	Flujo que integra CheckPanel, MintPanel, NFTListPanel y Toast. Gestiona la validación y emisión de ProofOfWork NFTs con feedback completo del proceso.
Toast	Componente de notificación flotante que muestra mensajes de confirmación de transacciones. Incluye hash y link a Etherscan Sepolia. Se autocierra tras un tiempo configurado.