GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

SISTEMA DESCENTRALIZADO PARA EL REGISTRO Y PROTECCIÓN DE OBRAS DIGITALES MEDIANTE BLOCKCHAIN

Autor: Santiago Manuel Sicilia Maroto

Director: Atilano Ramiro Fernández-Pacheco Sánchez-Migallón

Madrid-Marzo 2025

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Registro descentralizado de la propiedad intelectual.

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2024/25 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Santiago Manuel Sicilia Maroto Fecha: 13/Junio/2025

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

71216314B ATILANO RAMIRO FERNÁNDEZ-PACHECO

Digitally signed by 71216314B ATILANO RAMIRO FERNÁNDEZ-PACHECO
Date: 2025.06.13 18:50:18 +02'00'

Fdo.: Atilano Ramiro Fernández-Pacheco Sánchez-Migallón Fecha: 13/Junio/2025

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

SISTEMA DESCENTRALIZADO PARA EL REGISTRO Y PROTECCIÓN DE OBRAS DIGITALES MEDIANTE BLOCKCHAIN

Autor: Santiago Manuel Sicilia Maroto

Director: Atilano Ramiro Fernández-Pacheco Sánchez-Migallón

Madrid-2025

Agradecimientos

Quiero agradecer este Trabajo de Fin de Grado a mi familia, especialmente a mis padres y a mi hermano por apoyarme siempre en todo lo que hago.

También quiero agradecer a mi director de proyecto, Atilano, por darme la oportunidad de hacer este trabajo.

SISTEMA DESCENTRALIZADO PARA EL REGISTRO Y PROTECCIÓN DE OBRAS DIGITALES MEDIANTE BLOCKCHAIN

Autor: Sicilia Maroto, Santiago Manuel.
Director: Fernández-Pacheco Sánchez-Migallón, Atilano Ramiro.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Este Trabajo de Fin de Grado consiste en el desarrollo de una plataforma web 3.0[1] que permite registrar obras digitales como NFTs en la blockchain[2] de Ethereum de forma descentralizada. El sistema garantiza la autenticidad mediante el uso de contratos inteligentes y marcas de tiempo, otorgando la autoría a la primera obra registrada. Su despliegue en red de pruebas facilita una futura migración a la red principal.

Palabras clave: Blockchain, Contrato Inteligente, Token.

1. Introducción

En el año 2008 Satoshi Nakamoto publicó el conocido artículo "*Bitcoin: A Peer-To-Peer[3]Electronic Cash System"* [1], proponiendo un sistema de pagos entre pares que resolvía el problema de la doble transacción mediante una cadena de bloques basada en prueba de trabajo. Esta innovación dio origen a la tecnología blockchain, que registra transacciones de forma segura mediante funciones hash[4] y acompañadas de marcas de tiempo (timestamps).

Posteriormente, Vitalik Buterin publicó el artículo "*Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform"* [2], amplió el concepto con contratos inteligentes, introduciendo lógica programable en la red y posibilitando nuevas aplicaciones descentralizadas como las desarrolladas en este proyecto.

2. Definición del proyecto

El objetivo es crear una plataforma web 3.0 para registrar obras digitales en la blockchain, garantizando su autenticidad, trazabilidad y validez global sin depender de instituciones centralizadas.

El sistema utiliza contratos inteligentes para registrar obras como NFTs en Ethereum, asociando un hash y metadatos únicos. La plataforma incluye una interfaz gráfica conectada a MetaMask, un backend en Node.js con una API REST, y almacenamiento descentralizado en IPFS. El diseño se enfoca en accesibilidad, seguridad y descentralización.

3. Descripción del sistema

---

[1] Web descentralizada donde los usuarios controlan sus datos mediante blockchain.
[2] Cadena de bloques: base de datos descentralizada y segura que almacena información en bloques enlazados y verificados criptográficamente.
[3] P2P o comúnmente conocido como red de pares.
[4] Valor alfanumérico único generado por una función criptográfica.

El sistema se divide en tres capas: frontend, backend y almacenamiento distribuido (Figura 1). El frontend, accesible desde el navegador, permite al usuario interactuar con la plataforma y firmar transacciones mediante MetaMask.

El backend en Node.js gestiona la lógica de negocio, autenticación con JWT y conexión con los servicios de almacenamiento y blockchain.

Las obras se almacenan en IPFS junto a sus metadatos, y se registran como NFTs mediante un contrato ERC-721 desplegado en la red de pruebas de Sepolia. Además, una base de datos mantiene la información de usuario, garantizando una experiencia segura.

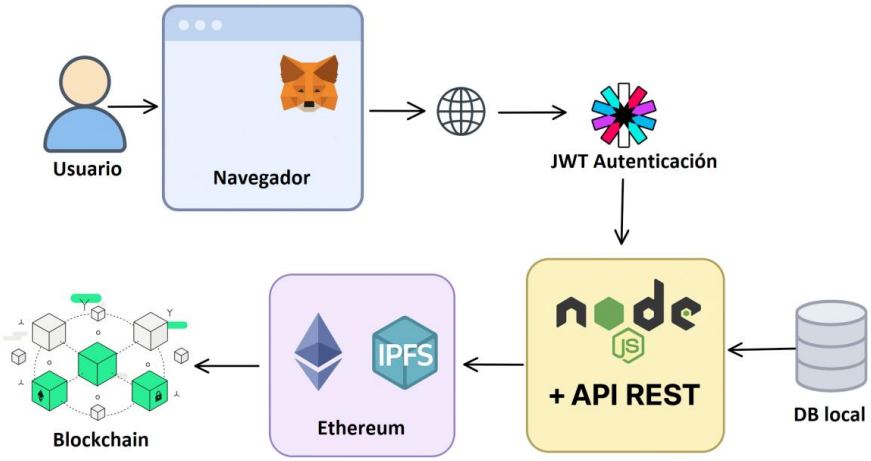

Figura 1. Arquitectura del sistema

4. Resultados

El sistema permite registrar y visualizar (Figura 2) NFTs desde la web conectando MetaMask, firmando la transacción y consultando su resultado en Etherscan.

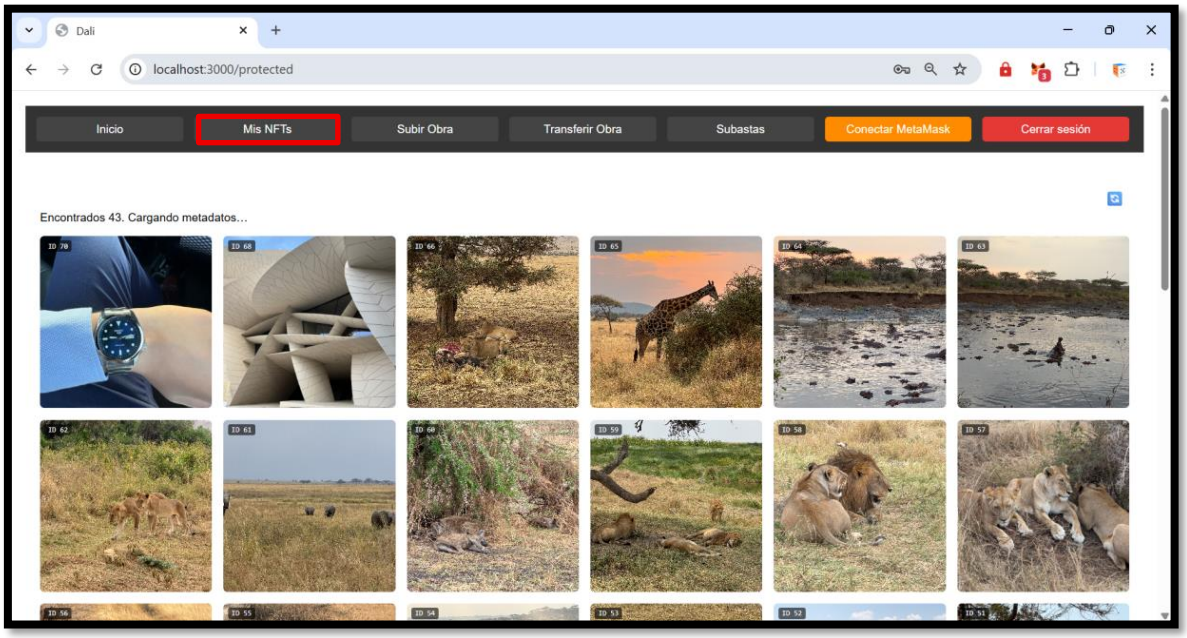

Figura 2. Visualización de mis NFTs

También se han implementado funcionalidades completas para la transferencia (Figura 3) de NFTs y la gestión de subastas. Los usuarios pueden transferir obras registradas entre diferentes cuentas de forma segura, verificando la identidad mediante MetaMask. Asimismo, la sección de subastas permite listar obras y realizar pujas directamente desde la plataforma, lo que amplía las posibilidades de interacción y comercialización de los activos digitales registrados.

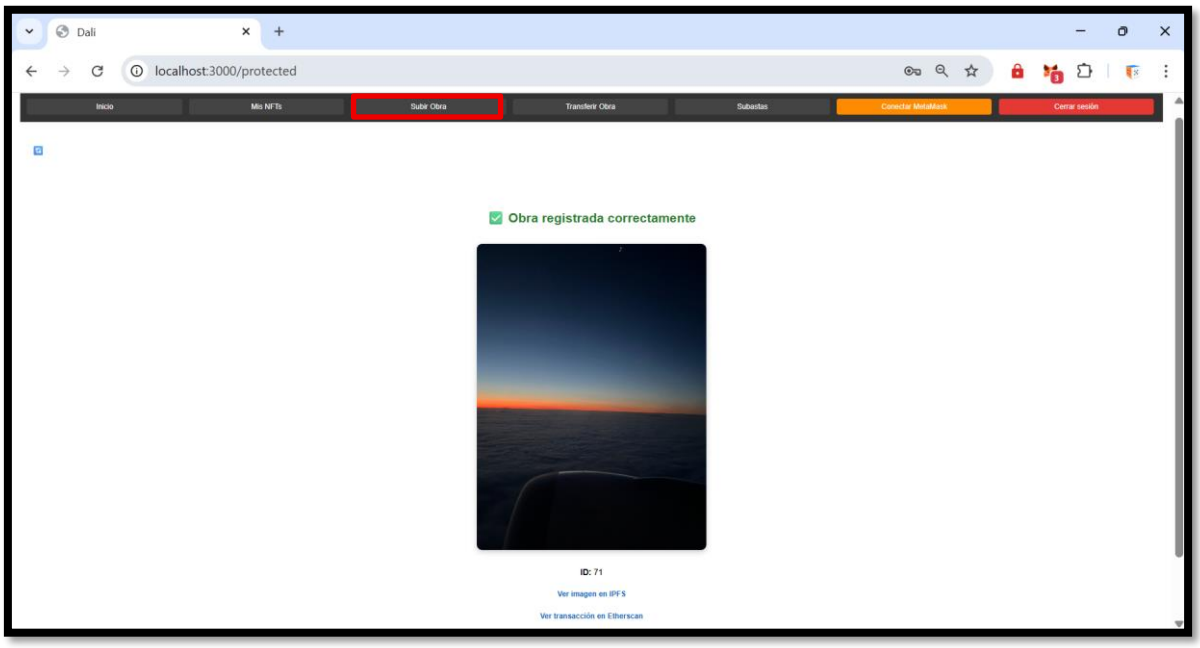

Figura 3. Obra registrada correctamente

5. Conclusiones

Este proyecto ha demostrado que es técnicamente viable registrar obras digitales en la blockchain de forma segura, accesible y descentralizada. Se ha desarrollado un sistema funcional que integra el registro de usuarios, la subida de archivos a IPFS, la creación de NFTs mediante contratos inteligentes ERC-721 y la validación de las transacciones en la red de pruebas de Ethereum.

Además de garantizar la autoría y trazabilidad de las obras registradas, el sistema ofrece una interfaz sencilla conectada a MetaMask y funcionalidades completas de transferencia y subastas. Como posibles mejoras futuras, se plantea el despliegue en la nube, la compatibilidad con otras redes blockchain más eficientes, la mejora del sistema de autenticación con soluciones descentralizadas y la incorporación de materiales educativos para facilitar el uso por parte de creadores sin conocimientos técnicos.

6. Referencias

[1] Nakamoto, Satoshi. “Bitcoin: A Peer-to-Peer Electronic Cash System”. 2008. https://bitcoin.org/bitcoin.pdf.

[2] Buterin, Vitalik. “Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform”. 2014. https://ethereum.org/en/whitepaper/.

DECENTRALIZED SYSTEM FOR THE REGISTRATION AND PROTECTION OF DIGITAL WORKS USING BLOCKCHAIN

Author: Sicilia Maroto, Santiago Manuel.
Supervisor: Fernández-Pacheco Sánchez-Migallón, Atilano Ramiro.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

This Bachelor´s Thesis consists of the development of a Web 3.0[5] platform that allows users to register digital works as NFTs on the Ethereum blockchain[6] in a decentralized manner. The system ensures authenticity using smart contracts and timestamps, granting authorship to the first registered version of the work. Deployment on the test network facilitates a future migration to the mainet.

Keywords: Blockchain, Smart Contract, Token.

1. Introduction

In 2008, Satoshi Nakamoto published the well-known article *"Bitcoin: A Peer-to-Peer[7] Electronic Cash System"* [1], introducing a peer-to-peer payment system that solved the double-spending problem using an immutable blockchain based on proof of work. This innovation led to the development of blockchain technology, where transactions are securely recorded using hash[8] functions and timestamps.

Later, Vitalik Buterin published the article *"Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform"* [2], enabling programmable logic within the blockchain and opening the door to decentralized applications like the one developed in this project.

2. Project definition

The goal is to create a web 3.0 platform to register digital works on the blockchain, ensuring authenticity, traceability, and global validity without relying on centralized institutions.

The system uses smart contracts to register works as NFTs on Ethereum, associating a hash and unique metadata. The platform includes a user-friendly interface connected to MetaMask, a backend developed with Node.js and a REST API, and decentralized file storage using IPFS. The design focuses on accessibility, security and decentralization.

3. System description

---

[5] Decentralized web where the users control their data through blockchain.
[6] A decentralized web and secure database that stores information in cryptographically linked and verified blocks.
[7] P2P, commonly known as peer-to-peer network.
[8] Unique alphanumeric value generated by a cryptographic function

The system is structured into three main layers: frontend, backend and distributed storage (Figure 4). The frontend, accessible via the browser, enables the user to interact with the platform and sign transactions using MetaMask.

The backend, built with Node.js, handles business logic, user authentication via JWT, and connections to blockchain and IPFS services.

Digital work is uploaded and stored in IPFS along with their metadata and registered as NFTs through an ERC-721 smart contract deployed on the Sepolia testnet. A local database stores user data, ensuring a secure and seamless experience.

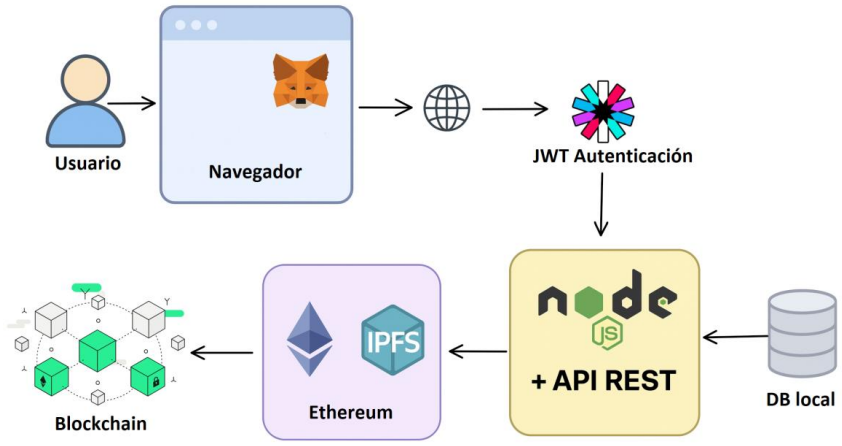

Figure 4. System architecture

4. Results

The system allows users to register and view their NFTs from the web interface (Figure 5), connecting MetaMask, signing transactions, and checking the results on Etherscan.

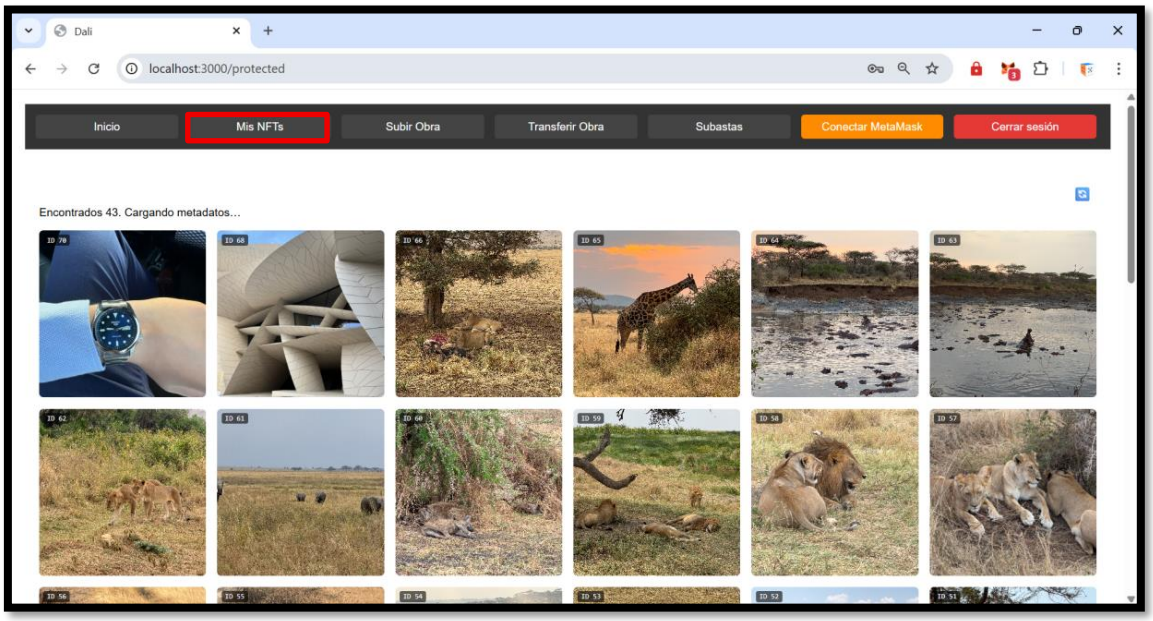

Figure 5. My NFTs viewer

In addition, full functionalities have been implemented for NFT transfers (Figure 6) and auctions. Users can transfer registered works between different accounts securely, verifying wallet ownership via MetaMask. The auction module enables users to list works and place bids directly from the platform, enhancing the possibilities of managing and commercializing digital assets.

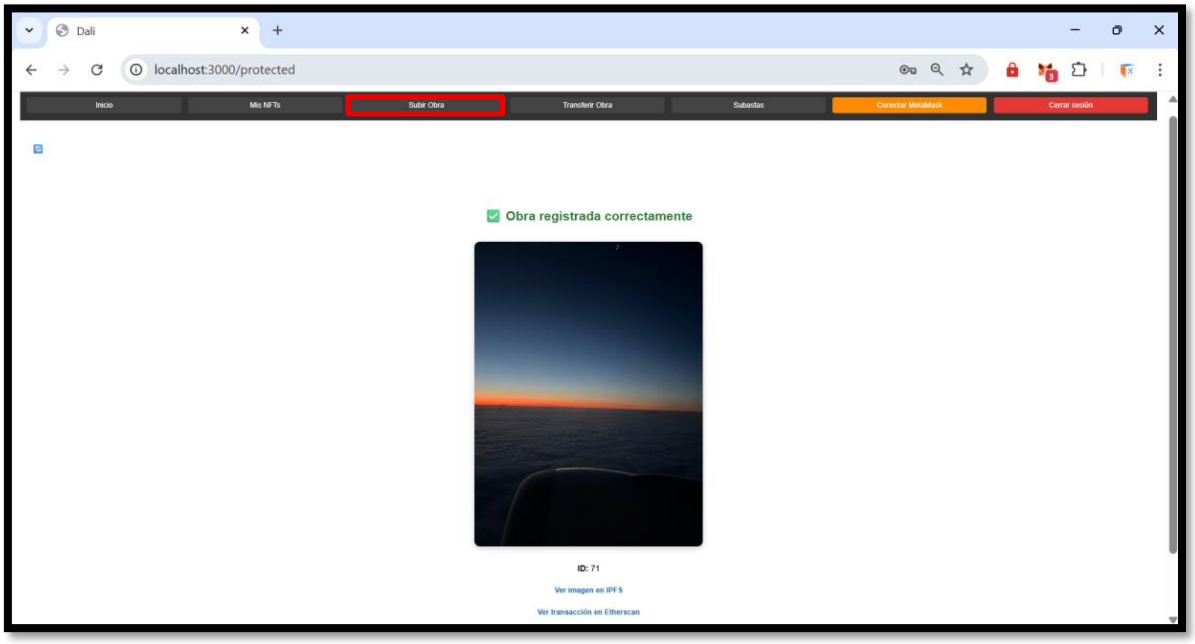

Figure 6. Successful NFT registration

5. Conclusions

This Project has demonstrated the technical feasibility of registering digital works on the blockchain in a secure, accessible, and decentralized way. A functional system has been developed, integrating user registration, file upload to IPFS, NFT creation through ERC-721 smart contracts, and transaction validation on the Ethereum test network.

In addition to ensuring authorship and traceability of the registered works, the platform offers a simple interface connected to MetaMask and fully implemented features for transfers and auctions. As future improvements, the project proposes deployment in the cloud, compatibility with more efficient blockchain networks, enhancement of the authentication system using decentralized solutions, and the inclusion of educational materials to support non-technical creators.

6. References

[1] Nakamoto, Satoshi. “Bitcoin: A Peer-to-Peer Electronic Cash System”. 2008. https://bitcoin.org/bitcoin.pdf.

[2] Buterin, Vitalik. “Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform”. 2014. https://ethereum.org/en/whitepaper/.

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. INTRODUCCIÓN

La tecnología ha estado presente en la humanidad desde el inicio de los tiempos. Esta permite al ser humano realizar tareas complejas de una forma mucho más sencilla. Con el paso de los años, ha habido más avances tecnológicos, pero estos crecían de una forma lineal, es decir, entre dos generaciones apenas había diferencias en términos tecnológicos, pero desde el siglo XX, especialmente el siglo XXI, vivimos en un cambio de paradigma, la tecnología está en constante desarrollo y para el ser humano supone un reto adaptarse a ellas.

En términos regulativos, este avance exponencial les repercute negativamente, ya que exige a estas instituciones a estar actualizándose tecnológicamente prácticamente a diario. En la gran mayoría de casos, estas actualizaciones llegan muy tarde o directamente no llegan, lo que crea vacíos legales, confrontación entre los usuarios o incongruencias entre ellos.

En el ámbito de la propiedad intelectual, el procedimiento tradicional de registro se realiza a través del Registro de la Propiedad Intelectual (RPI), gestionado por el Ministerio de Cultura y Deporte [1]. Es un sistema centralizado que ha funcionado durante décadas [2]. Este registro tiene validez únicamente en territorio español, aunque los derechos de autor están protegidos internacionalmente mediante tratados como el Convenio de Berna[9] [3]. La gestión del registro se realiza en colaboración con las comunidades autónomas [4].

El proceso puede realizarse de manera presencial o telemática, pero requiere contar con un certificado digital, acceso a la sede electrónica del Ministerio de Cultura, rellenar una serie de formularios, adjuntar la obra, realizar el pago de tasas, firmar digitalmente y, finalmente, enviar la solicitud. Todo esto hace que registrar una obra sea un proceso arduo y prolongado, además de ser burocrático y con falta de interoperabilidad internacional directa.

[9] Tratado internacional firmado en 1886 que establece la protección automática de los derechos de autor en todos los países firmantes, sin necesidad de registro previo.

En 2008, Satoshi Nakamoto publicó el artículo *"Bitcoin: A Peer-to-Peer*[10] *Electronic Cash System"* [5], en el que proponía un nuevo sistema electrónico para realizar transacciones monetarias directamente entre dos personas, sin necesidad de intermediarios financieros. Hasta entonces, el principal problema de este enfoque era la facilidad con la que podían falsificarse las transacciones en ausencia de una entidad central que las registrara. La idea revolucionaria del proyecto fue la introducción de la prueba de trabajo (proof of work), en la cual la cadena de bloques más larga se considera válida, ya que representa la mayor cantidad de esfuerzo computacional (CPU[11]). Esta cadena, conocida hoy como blockchain[12], registra todas las transacciones de forma segura, codificadas mediante funciones hash[13] y acompañadas de marcas de tiempo (*timestamps*[14]).

Años más tarde, Vitalik Buterin presentó el proyecto "*Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform"* [6], con el objetivo de ir más allá del simple intercambio de valor entre pares. Su propuesta introdujo la posibilidad de ejecutar contratos inteligentes, que son fragmentos de código programado capaces de automatizar y verificar condiciones sin necesidad de intermediarios. Ethereum incorpora dos tipos de cuentas: las externally owned accounts (EOAs), controladas por claves privadas, y las contract accounts, gestionadas por el propio código desplegado en la red. A diferencia de Bitcoin, Ethereum permite implementar lógica compleja dentro de su propia infraestructura.

Este Trabajo de Fin de Grado tiene como objetivo proponer una solución a las limitaciones del sistema tradicional de registro mediante el uso de tecnologías emergentes, concretamente la blockchain, para desarrollar un sistema de registro descentralizado de obras digitales. Esta

[10] P2P o comúnmente conocido como red de pares.
[11] Unidad central de procesamiento que ejecuta instrucciones y coordina las operaciones principales de un sistema informático.
[12] Cadena de bloques: base de datos descentralizada y segura que almacena información en bloques enlazados y verificados criptográficamente.
[13] Valor alfanumérico único generado por una función criptográfica que representa de forma compacta el contenido de una información.
[14] Marca de tiempo que indica el momento exacto en que ocurre un evento o se registra una información digital.

propuesta busca ofrecer una alternativa más segura, transparente y globalmente accesible para los autores.

En el contexto de Web 3.0, donde los usuarios controlan sus datos y activos digitales, surge la posibilidad de aplicar contratos inteligentes que automatizan el proceso de registro, garantizando la inmutabilidad de la información y eliminando la dependencia de intermediarios.

La finalidad del proyecto es adaptar el proceso de registro de obras digitales a las tecnologías actuales, haciéndolo mucho más ágil, accesible y eficiente que el modelo tradicional. Además de reducir la carga burocrática y los costes asociados, la propuesta busca facilitar un sistema con validez global, que permita a los autores proteger su obra desde cualquier lugar del mundo sin depender de instituciones centralizadas.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

2.1 BACKEND [15](API[16]/LÓGICA DEL SERVIDOR)

2.1.1 NODE.JS

Node.js [7] es un entorno de ejecución de JavaScript del lado del servidor que permite construir aplicaciones web rápidas, escalables y eficientes (Figura 7).

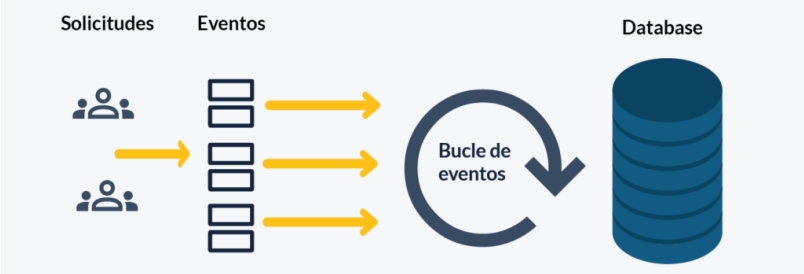

Figura 7. Funcionamiento de Node.js (Fuente Profile.es*)*

Una de las características más destacadas de Node.js es su arquitectura orientada a eventos y no bloqueante, lo que lo hace ideal para manejar múltiples conexiones simultáneas sin sacrificar rendimiento. Gracias a su ecosistema de paquetes a través de npm (Node Package Manager), es ampliamente utilizado para construir APIs REST[17], servidores web, herramientas CLI[18] y aplicaciones en tiempo real.

[15] Parte del desarrollo web que se encarga de la lógica interna de la aplicación, el procesamiento de datos, la conexión con bases de datos, servidores y, en este proyecto, también con la blockchain.
[16] Conjunto de reglas que permiten que dos programas se comuniquen entre sí.
[17] Tipo de API basada en el protocolo HTTP.
[18] Son programas que se usan desde la terminal o consola escribiendo comandos, sin interfaz gráfica.

2.1.2 EXPRESS

Express [8] es un framework web[19] minimalista y flexible para Node.js, que facilita la creación de servidores APIs de forma sencilla y estructurada.

Express permite definir endpoints[20] para manejar solicitudes GET, POST, PUT, DELETE, entre otra, y soporta middlewares[21], lo que lo convierte en una herramienta ideal para construir aplicaciones web y APIs RESTful. Su compatibilidad con librerías externas como CORS[22], Dotenv[23] o JWT (tecnologías que se verán más adelante) lo hace altamente extensible.

2.1.3 IPFS

IPFS [9] (InterPlanetary File System) es un protocolo de almacenamiento descentralizado diseñado para guardar y compartir archivos de manera distribuida, sin depender de servidores centrales. Funciona mediante un sistema de contenidos direccionados por hash (CID), lo que garantiza la integridad de los datos y permite la recuperación desde múltiples nodos de la red.

2.2 FRONTEND[24]

2.2.1 HTML

HTML (HyperText Markup Language) es el lenguaje estándar para estructurar y presentar el contenido de páginas web. Define la jerarquía y los elementos de una interfaz como textos,

[19] Conjunto de herramientas, librerías y reglas predefinidas que te ayudan a desarrollar aplicaciones de forma más rápida, estructurada y eficiente.
[20] URL especifica dentro de una API que representa un recurso o una acción.
[21] Es una función intermedia entre la solicitud del cliente y la respuesta del servidor que permite controlar el flujo, validar datos o realizar tareas antes de responder.
[22] Controla que orígenes pueden comunicarse con tu navegador.
[23] Librería de Node.js que facilita la gestión de datos sensibles como claves privadas.
[24] Parte visible de la aplicación web, es decir, la interfaz con la que interactúa el usuario.

formularios, imágenes y botones, sirviendo como base del frontend para cualquier aplicación web.

2.2.2 JAVASCRIPT

JavaScript es un lenguaje de programación interpretado, orientado a objetos y ampliamente utilizado para añadir interactividad, lógica y dinamismo a páginas web. En este proyecto, se emplea para gestionar eventos del usuario, interactuar con Metamask y comunicarse con el backend.

2.2.3 EJS

EJS (Embedded JavaScript) es un motor de plantillas que permite incrustar código JavaScript dentro de archivos HTML. Facilita la generación dinámica de contenido en el servidor antes de enviarlo al navegador, permitiendo renderizar vistas con datos personalizados, como mensajes de Login o confirmación de registros.

2.3 METAMASK

MetaMask [10] es una billtera de criptomonedas (software wallet[25]) que permite a los usuarios interactuar con la blockchain de Ethereum. Funciona como una extensión de navegador y una aplicación móvil, facilitando el acceso a aplicaciones descentralizadas (dApps) y la gestión de archivos digitales. Además de almacenar y gestionar claves de cuentas, MetaMask permite a los usuarios enviar y recibir criptomonedas y realizar transacciones.

[25] También denominada billetera caliente, es más cómoda para el uso frecuente y solo necesita conexión a internet y un buscador para funcionar. Pero está más expuesta a riesgos de seguridad.

2.4 AUTENTICACIÓN

2.4.1 BYCRIPT

Bcrypt [11] toma una contraseña en texto plano y la transforma en un hash seguro, que es una representación irreconocible e irreversible de la contraseña. Este hash se guarda en la base de datos en lugar de la contraseña real.

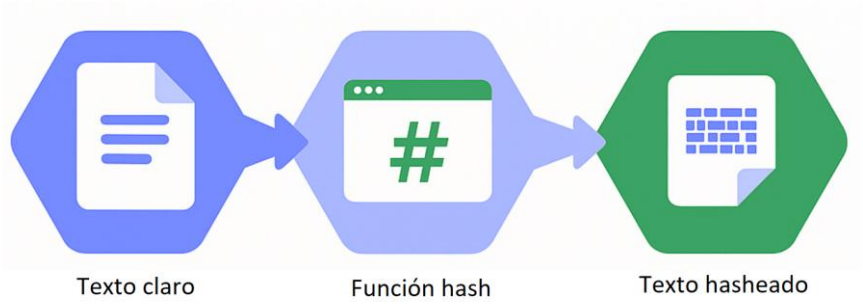

Figura 8. Funcionamiento interno de Bcrypt (Fuente: NordPass*)*

2.4.2 JWT

JWT (JSON Web Token) [12] es un estándar abierto (RFC 7519) que se usa para transmitir información de manera segura y compacta entre dos partes como un token firmado digitalmente. Se usa sobre todo para gestionar sesiones de usuario, es decir, saber quién está autenticado, sin tener que guardar sesiones en el servidor (Figura 9).

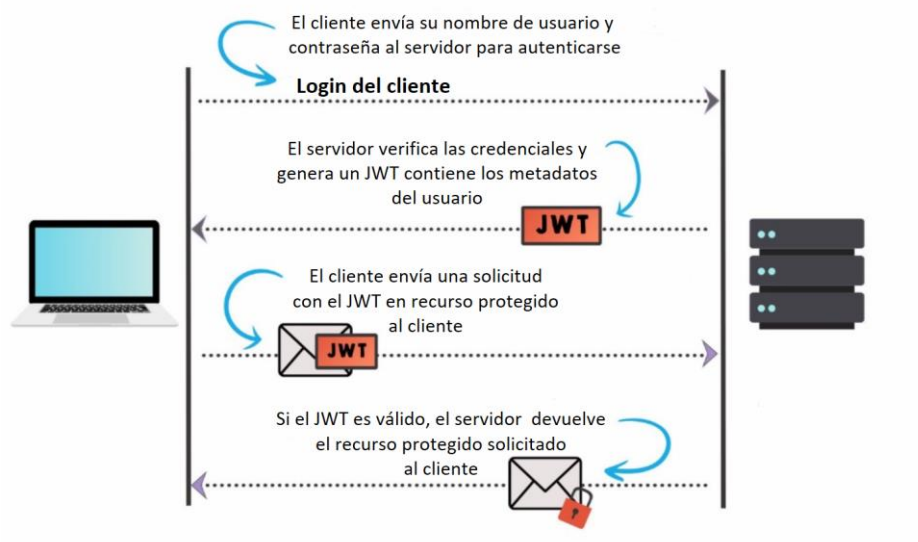

Figura 9. Funcionamiento de JWT (Fuente: NinaDurann*)*

2.4.3 COOKIE-PARSER

Cokie-parser es un middleware para Express.js que lee y analiza las cookies que el navegador envía en las peticiones HTTP. Una vez procesadas, las cookies quedan disponibles en req.cookies.

2.4.4 HTTPS

HTTPS (Hypertext Transfer Protocol Secure) es el protocolo que usan los navegadores y servidores para comunicarse a través de la web. Los datos que se envían entre el cliente y el servidor son cifrados.

2.5 DEVOPS Y CONTROL DE VERSIONES

2.5.1 GITHUB

GitHub [13] es una plataforma en la nube que permite a desarrolladores almacenar, gestionar y colaborar en proyectos de software usando Git, un sistema de control de versiones.

Una de las funcionalidades más potentes de GitHub son las ramas (branches). Las ramas (Figura 10) permiten a los desarrolladores trabajar en nuevas características o corregir errores de forma aislada, sin afectar a la versión principal del código (en este caso la rama *main*). Una vez que los cambios han sido probados y revisados, pueden integrarse a la rama principal mediante un pull request.

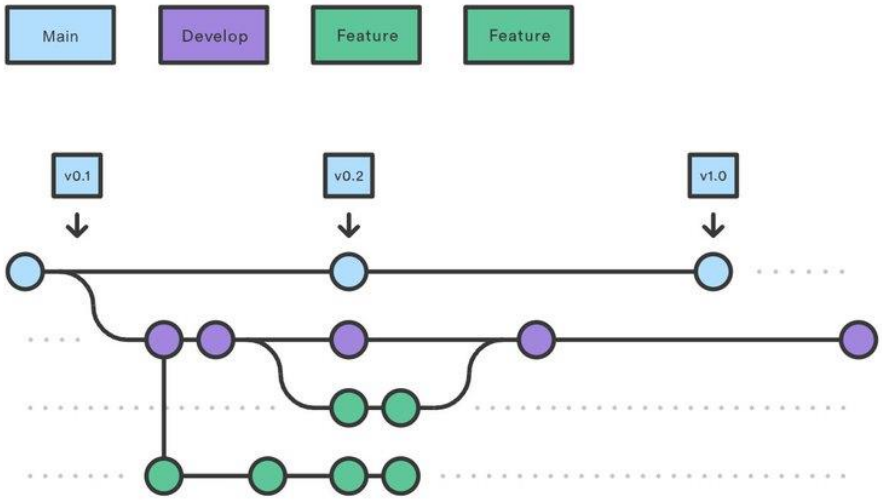

Figura 10. Estrategia de ramas en GitHub (Fuente: Midudev*)*

2.5.2 CI/CD

CI (Continuous Integration), la integración continua consiste en la ejecución automática de pruebas y la obtención de sus informes cada vez que cambia el código.

CD (Continuous Delivery/Deployment), el despliegue continuo permite instalar automáticamente los cambios una vez que estos han superado todas las pruebas.

Combinando ambos procesos es posible automatizar los despliegues de nuevas versiones con la certeza de que no tienen fallos y que, en caso de haberlos, se impida la integración de los cambios y su despliegue.

Capítulo 3. ESTADO DE LA CUESTIÓN

En esta sección se llevará a cabo un análisis de los trabajos previos y las soluciones existentes relacionadas con el ámbito del presente proyecto, con el objetivo de identificar avances, enfoques tecnológicos y posibles áreas de mejora que justifiquen la propuesta desarrollada en este Trabajo de Fin de Grado.

3.1 INTRODUCCIÓN

El avance de las tecnologías digitales ha transformado profundamente los modelos de creación, distribución y consumo de contenidos, haciendo más accesible tanto la producción como la difusión de obras creativas. Sin embargo, esta accesibilidad también ha traído consigo un aumento en los casos de plagio, copia no autorizada y distribución sin consentimiento de contenido original, lo cual ha puesto de manifiesto la necesidad de nuevas soluciones para la protección de la propiedad intelectual.

En este contexto, la tecnología blockchain ha emergido como una herramienta prometedora para abordar este problema, debido a su capacidad para registrar información de forma inmutable, verificable y descentralizada. Al combinar blockchain con otras tecnologías como IPFS (InterPlanetary File System) y los tokens no fungibles (NFTs), es posible establecer sistemas de registro de obras digitales que garanticen su integridad, autoría y trazabilidad.

Este capítulo presenta un análisis del estado actual del arte, identificando plataformas comerciales, investigaciones académicas y proyectos relacionados que abordan esta problemática. Se estudiarán sus aportaciones, limitaciones y enfoques tecnológicos, con el fin de establecer el contexto en el que se enmarca el presente trabajo y justificar su relevancia.

3.2 FUNDAMENTOS TÉCNICOS DE BLOCKCHAIN

3.2.1 BLOCKCHAIN

La blockchain [5] es una estructura de datos distribuida que permite registrar información de forma segura, transparente e inmutable. Está compuesta por una secuencia de bloques enlazados criptográficamente, cada uno de los cuales contiene un conjunto de transacciones y otros datos relevantes, como marcas de tiempo (timestamps), números de bloque, niveles de dificultad o raíces Merkle[26].

A diferencia de una base de datos tradicional, la blockchain no está alojada en un único servidor, sino que se replica entre miles de nodos distribuidos en todo el mundo. Cada nodo mantiene una copia del historial completo de la cadena y participa en el proceso de validación de nuevos bloques. Esta arquitectura descentralizada elimina la necesidad de intermediarios de confianza y proporciona resistencia a la censura y a la manipulación, ya que ninguna entidad individual puede modificar el contenido de los bloques por sí sola.

Para que un nuevo bloque se añada a la cadena, todos los nodos deben alcanzar un acuerdo sobre su validez mediante un algoritmo de consenso. Existen diferentes mecanismos para lograr este consenso:

- Proof of Work (PoW): es el método utilizado por Bitcoin y por Ethereum hasta 2022. Requiere que los nodos, conocidos como mineros, resuelvan un problema criptográfico complejo (generalmente encontrar un hash por debajo de un determinado umbral). El primero que lo consigue tiene derecho a añadir el bloque a la cadena y es recompensado. Este mecanismo es robusto, pero conlleva un elevado consumo energético.
- Proof of Stake (PoS): es el Sistema actualmente utilizado por Ethereum tras la actualización The Merge. En este modelo, los validadores son seleccionados para

[26] Es un hash único que representa todo el contenido de un conjunto de datos (como transacciones), generado mediante una estructura llamada árbol Merkle.

proponer y validar bloques en función de la cantidad de criptomonedas que tienen bloqueadas como garantía (stake). Este enfoque es mucho más eficiente energéticamente y mantiene altos niveles de seguridad.

En el caso de Ethereum [6] (la tecnología que se desarrollará en este proyecto), la blockchain funciona de forma similar a la de Bitcoin, pero con importantes diferencias estructurales. La más significativa es que cada bloque de Ethereum no solo contiene lista de transacciones, sino también una copia completa del "estado" más reciente del sistema, es decir, el conjunto de saldos, contratos y datos almacenados en la red en ese momento.

Tal y como se muestra en la Figura 11, una única transacción provoca un cambio en los balances y datos del estado, reflejado en el nuevo bloque.

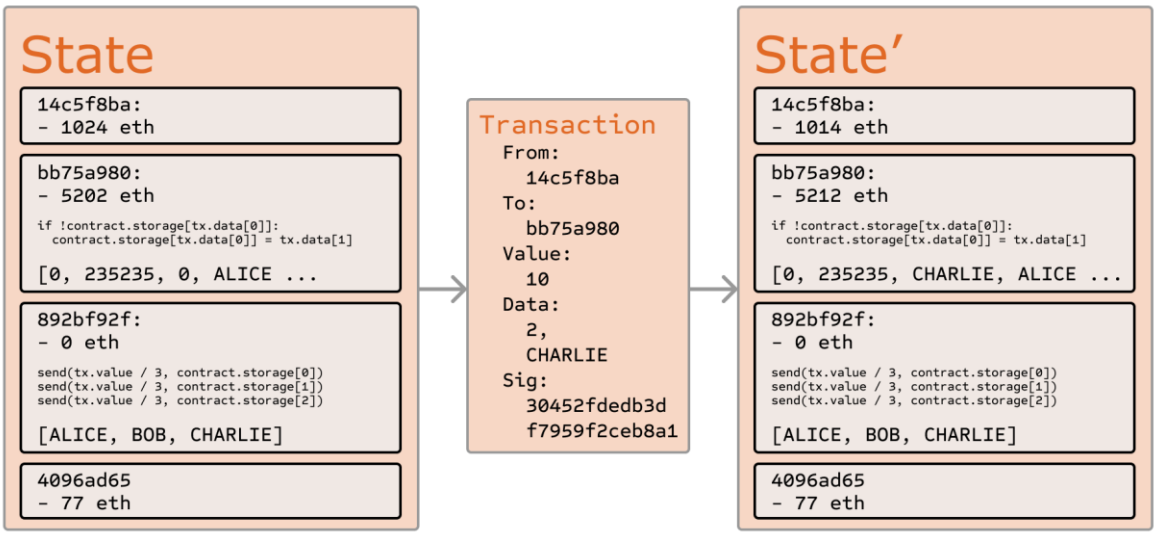

Figura 11. Ejemplo sistemático de como una transacción modifica el estado global de Ethereum (Fuente: Ethereum.org)

La validación de un bloque de Ethereum incluye varios pasos fundamentales: verificación de la existencia y validez del bloque anterior, comprobación de que el timestamp del nuevo bloque sea cronológicamente coherente, validación del número de bloque, dificultad, gas

limit[27] y raíces Merkle (Figura 12). En versiones previas a Ethereum 2.0, también se comprobaba la validez de la prueba de trabajo (PoW). Tras estas verificaciones, se procede a la ejecución secuencial de todas las transacciones del bloque, aplicando cambios al estado global de la red y controlando el consumo de gas. Finalmente, se valida que el estado resultante coincida con el state root[28] declarado en el encabezado del bloque.

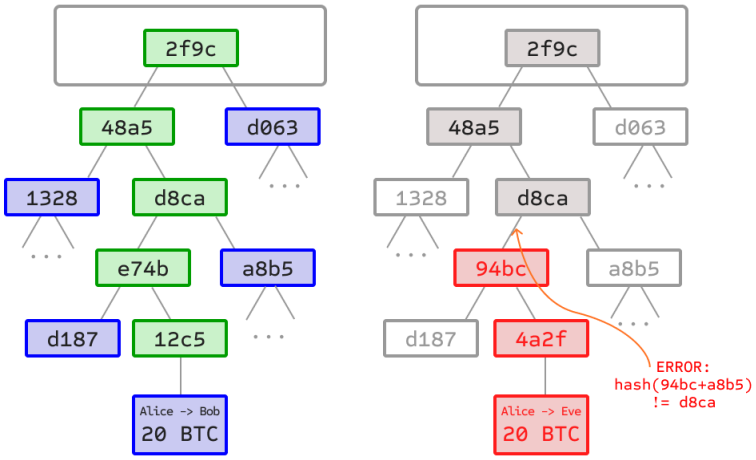

Figura 12. Árboles de Merkle

Izquierda: Basta con presentar solo un pequeño número de nodos en un árbol de Merkle para demostrar la validez de una bifurcación.

Derecha: Cualquier intento de cambiar cualquier parte del árbol de Merkle llevará eventualmente a una inconsistencia en algún lugar previo de la cadena.

Esta estructura garantiza que cualquier modificación en el contenido de un bloque alteraría su hash y, por tanto, invalidaría todos los bloques posteriores, rompiendo la cadena. Por esta razón, la blockchain es extremadamente resistente a las manipulaciones y constituye una herramienta muy eficaz para registrar información crítica como la propiedad intelectual de obras digitales.

[27] Cantidad máxima de unidades de gas que un usuario está dispuesto a consumir para ejecutar una transacción o contrato inteligente en Ethereum.
[28] Es el hash raíz del estado global de la red Ethereum en un bloque, que resume toda la información sobre saldos, contratos y almacenamiento en un solo valor criptográfico verificable

3.2.2 ETHEREUM

Ethereum [6] es una plataforma blockchain de código abierto diseñada para permitir la ejecución de contratos inteligentes (smart contracts). A diferencia de Bitcoin, cuya función principal es servir como moneda digital, Ethereum fue concebido como una red descentralizada en la que se pueden desplegar aplicaciones que funcionen sin intermediarios. Esto la convierte en una infraestructura ideal para desarrollar soluciones que requieran confianza, trazabilidad y automatización, como es el caso del registro de la propiedad intelectual.

Ethereum incorpora una máquina virtual llamada EVM (Ethereum Virtual Machine), encargada de ejecutar código de los contratos inteligentes de forma distribuida en todos los nodos de la red. Gracias a esto, cualquier interacción con el contrato inteligente (como registrar una obra digital) se convierte en una transacción inmutable, verificada por la propia red y accesible públicamente.

Para este proyecto se ha utilizado Sepolia, una de las redes de prueba oficiales de Ethereum. Las testnets como Sepolia replican el comportamiento de la red principal, pero utilizan tokens sin valor real y que permiten realizar pruebas sin coste económico. Esto facilita el desarrollo, despliegue y validación de contratos inteligentes en condiciones seguras antes de ser migrados a la red principal.

3.2.2.1 Etherscan

Con Etherscan [14], cualquier usuario puede consultar públicamente el historial de una obra registrada, verificar su autoría y rastrear todas las transferencias realizadas, lo que refuerza la confianza y la trazabilidad del sistema (Figura 13).

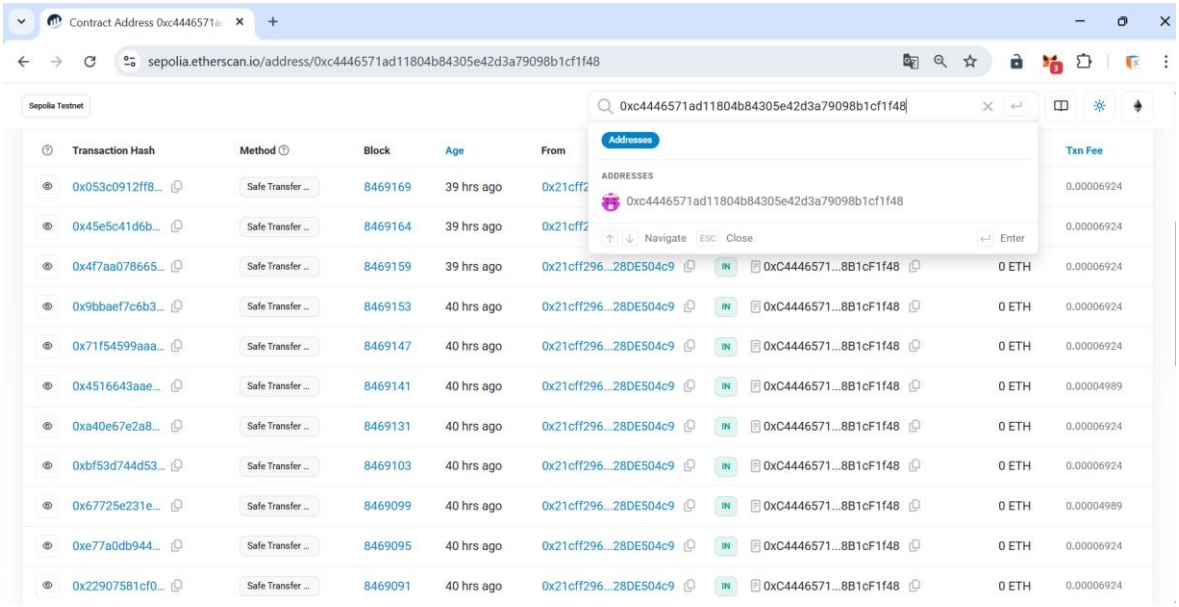

Figura 13. Ejemplo práctico de Sepolia Etherscan

3.2.3 CONTRATOS INTELIGENTES

Los contratos inteligentes son programas autoejecutables desplegados en la blockchain, cuya lógica se activa automáticamente cuando se cumplen ciertas condiciones predefinidas, sin necesidad de intermediarios. Tal como se expone en el whitepaper de Ethereum [6], estos contratos permiten mover activos digitales de acuerdo con las reglas arbitrarias programadas, lo que habilita aplicaciones como el registro de propiedad intelectual de obras digitales.

Ethereum proporciona un entorno ideal para su implementación al incluir un lenguaje de programación Turing-completo, lo que permite definir cualquier lógica de negocio o verificación. A través de los contratos inteligentes se registran obras digitales de forma inmutable, asignando hashes únicos vinculados a la obra, al autor y a la marca de tiempo del momento del registro.

3.2.3.1 Solidity

Solidity [15] es el lenguaje de programación principal utilizado para desarrollar contratos inteligentes en la red de Ethereum. Su sintaxis está inspirada en lenguajes como Javascript[29] y C++[30], y permite definir estructuras, funciones y eventos dentro del contrato. Gracias a su capacidad de definir reglas complejas y estados persistentes, Solidity se adapta perfectamente a la lógica requerida en el registro de obras digitales.

3.2.3.2 Remix IDE

Remix [16] es un entorno de desarrollo integrado (IDE) basado en navegador, diseñado específicamente para programar, compilar, depurar y desplegar contratos inteligentes escritos en Solidity. Ofrece una interfaz intuitiva, permite conectarse a redes de prueba como Sepolia, y facilita la interacción directa con los contratos mediante su consola integrada.

3.3 SOLUCIONES COMERCIALES.

3.3.1 ASCRIBE

Ascribe [17] fue una iniciativa pionera lanzada en 2014 que permitió a artistas y creadores registrar obras digitales en blockchain y transferir derechos digitales mediante contratos inteligentes. En sus inicios utilizaba la red de Bitcoin, y posteriormente, hacia 2016, se exploraron integraciones con Ethereum. A través de Ascribe, los autores podían generar certificados digitales de sus obras, establecer licencias y realizar seguimientos de transferencias de propiedad.

[29] Lenguaje de programación interpretado y orientado a objetos, ampliamente utilizado para desarrollar funcionalidades interactivas en páginas web.
[30] Lenguaje de programación compilado, de propósito general y orientado a objetos, conocido por su alto rendimiento y control preciso sobre los recursos del sistema.

Aunque la plataforma cesó sus operaciones activas alrededor de 2017, su enfoque influyó notablemente en el desarrollo de proyectos posteriores y en la normalización del uso de la tecnología blockchain en el ámbito del arte digital y los NFTs.

3.3.2 PO.ET

Po.et (2017) [18] fue una de las primeras plataformas en proponer un protocolo abierto y descentralizado basado en blockchain para registrar y gestionar contenido digital. Su objetivo principal era permitir a los creadores registrar metadatos de sus obras en la blockchain de Bitcoin, generando así una prueba de existencia y de autoría verificable públicamente. Además, proporcionaba herramientas para la monetización y distribución de contenidos.

A pesar de su visión innovadora, Po.et cesó sus operaciones en 2021, principalmente por dificultades en su modelo de negocio y en la adopción masiva de la plataforma. No obstante, sentó un precedente importante en la utilización de blockchain para certificar la propiedad intelectual de manera descentralizada.

3.3.3 OPENSEA Y PLATAFORMAS DE NFTS

OpenSea [19], Rarible [20], Mintable [21] y Foundation [22] son ejemplos actuales de plataformas que permiten a los usuarios crear, comprar y vender NFTs, tokens únicos que representan activos digitales. Estas plataformas están construidas principalmente sobre la blockchain de Ethereum y permiten a los creadores emitir NFTs para sus obras digitales, estableciendo cierta trazabilidad y reconocimiento de autoría.

El contenido vinculado a los NFTs suele almacenarse en servicios de almacenamiento distribuido como IPFS (InterPlanetary File System), lo que permite conservar una referencia del recurso digital asociado al token en la blockchain. Estas plataformas también proporcionan funcionalidades para la trazabilidad, la gestión de regalías y la visibilidad de las transacciones dentro del ecosistema NFT.

3.4 PROYECTOS ACADÉMICOS Y PROPUESTAS DE INVESTIGACIÓN.

En el ámbito académico, varios estudios han abordado la protección de la propiedad intelectual mediante blockchain desde diferentes perspectivas técnicas y legales.

3.4.1 PROTECCIÓN DE IMÁGENES MEDIANTE BLOCKCHAIN E IPFS

En el artículo titulado "A novel copyright protection scheme for digital images based on blockchain and IPFS" [23], los autores Cong, Lin y Li (2024) proponen una arquitectura innovadora para la protección de los derechos de autor de imágenes digitales mediante la combinación de tecnologías descentralizadas, concretamente IPFS y blockchain. En esta solución, los archivos de imagen se almacenan en IPFS, mientras que la información de propiedad intelectual, incluyendo el hash del archivo, se registra en una blockchain mediante contratos inteligentes. Esta estrategia permite verificar la integridad del contenido en cualquier momento, sin necesidad de acceder al archivo original, garantizando así la inmutabilidad y trazabilidad de los datos.

Además, el sistema introduce el uso de firmas digitales y marcas de agua invisibles (blind watermarking) para reforzar la autenticación de autoría. La firma digital asocia de forma segura al autor con el contenido registrado, y la marca de agua embebe de forma imperceptible la identidad del creador dentro de la imagen. El diseño también incluye una estructura de indexado DHT de doble capa, que optimiza significativamente la eficiencia de búsqueda en la red IPFS, reduciendo el tiempo de recuperación de archivos en un 33%.

El estudio concluye que esta combinación tecnológica proporciona un sistema eficiente, transparente y resistente a manipulaciones para certificar obras digitales. No obstante, se reconoce que la validación plena de la autoría original sigue representando un desafío, especialmente en escenarios donde no hay mecanismos externos que acrediten la identidad del creador.

3.4.2 PROTOCOLO DE PROTECCIÓN ENTRE COMPRADORES Y VENDEDORES

En el artículo "Blockchain and Smart Contracts for Digital Copyright Protection" [24], publicado por Zhang, Wei y Zhao (2024), se propone un protocolo innovador de marca de agua digital que permite la transmisión segura de obras digitales entre compradores y vendedores sin necesidad de una autoridad central confiable. Este enfoque, basado en el diseño "amigable para el comprador" (buyer-friendly), tiene como objetivo encontrar un equilibrio entre la usabilidad y la seguridad en la protección de los derechos de autor en entornos digitales.

El protocolo se apoya en la tecnología blockchain y en el uso de contratos inteligentes para definir reglas automáticas que se ejecutan únicamente bajo condiciones preestablecidas. Esto garantiza tanto la trazabilidad como la entrega segura del contenido protegido, sin requerir la intervención de terceros. Al prescindir de una autoridad central, se reducen significativamente los riesgos de colisión o manipulación, al tiempo que se incrementa la transparencia del sistema.

El estudio demuestra que este tipo de arquitectura es viable para construir un sistema de certificación y distribución de contenido eficiente, seguro y confiable. No obstante, al igual que en otras propuestas similares, los autores reconocen que la validación completa de la autoría original continúa siendo un reto, especialmente en ausencia de mecanismos externos de verificación de identidad.

3.4.3 GESTIÓN DE DERECHOS DIGITALES CON HYPERLEDGER.

En el artículo Digital Copyright Protection Based on Blockchain Technology [25], se presenta una arquitectura basada en Hyperledger Fabric para gestionar el ciclo de vida completo de los derechos digitales mediante contratos inteligentes. Esta propuesta se enfoca en la trazabilidad, la gobernanza y el control de acceso a los contenidos digitales, siendo aplicable a entornos corporativos y educativos.

3.5 TECNOLOGÍAS IMPLICADAS EN LOS ENFOQUES ESTUDIADOS.

Los trabajos y plataformas analizados comparten el uso de tecnologías clave que también forman parte del proyecto propuesto.

- Blockchain (Ethereum): Registro inmutable de transacciones, ideal para certificar la existencia de una obra en un momento dado.
- Contratos inteligentes: Automatización de procesos como el registro, la verificación y la transferencia de obras.
- IPFS: Sistema de almacenamiento descentralizado que permite preservar la integridad del contenido mediante identificadores únicos generados por hash.
- NFT (ERC-721 [26]): Representación digital única de cada obra, asociada a metadatos y derechos de autor.

Estos componentes permiten construir plataformas robustas para la protección de obras digitales, aunque su correcta integración y diseño de arquitectura son fundamentales para garantizar su usabilidad y efectividad.

3.6 ANÁLISIS

A partir del análisis realizado, se puede afirmar que las soluciones comerciales existentes han demostrado que la tecnología blockchain resulta eficaz para registrar la existencia de una obra digital de forma inmutable. No obstante, los estudios más recientes coinciden en señalar que aún existe un amplio margen de mejora, tanto en términos de validación de la autoría como en la integración de sistemas descentralizados de almacenamiento, usabilidad y adopción por parte del usuario final.

En este sentido, la elección del presente Trabajo de Fin de Grado responde no solo a la oportunidad de ofrecer una solución práctica a la problemática de la protección de la propiedad intelectual en entornos digitales, sino también al interés por profundizar en el conocimiento y aplicación de tecnologías emergentes como blockchain, IPFS y contratos inteligentes. Estas tecnologías no solo tienen implicaciones en el ámbito creativo, sino que

también están adquiriendo una relevancia creciente en sectores clave como la medicina, el ámbito jurídico, la trazabilidad industrial o la gestión de datos sensibles, lo que refuerza el valor formativo y transversal de este proyecto.

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

Tal y como se ha expuesto en los capítulos anteriores, el proceso tradicional de registro de la propiedad intelectual presenta una serie de limitaciones que afectan tanto a la usabilidad como a la efectividad del sistema. Actualmente, el registro se encuentra centralizado en entidades estatales, como el Registro de la Propiedad Intelectual gestionado por el Ministerio de Cultura en España, lo cual implica una fuerte dependencia institucional, procedimientos burocráticos complejos, costes asociados y una validez legal limitada territorialmente.

Además, aunque existen tratados internacionales como el Convenio de Berna [3] para la protección de derechos de autor, la interoperabilidad real entre sistemas de registro sigue siendo insuficiente, lo que dificulta la protección efectiva de los derechos de los creadores a escala global.

A esto se suma que los sistemas actuales no garantizan una verificación inmediata y descentralizada de la autoría. En muchos casos, demostrar quién fue el primero en registrar una obra implica trámites adicionales y pruebas documentales. Por tanto, se hace evidente la necesidad de una infraestructura tecnológica moderna, transparente y universal que permita a los autores registrar, verificar y proteger sus obras de forma segura, económica y accesible desde cualquier parte del mundo.

Al mismo tiempo, las soluciones comerciales actuales que integran blockchain, como OpenSea, Po.et o Ascribe, están centradas principalmente en la compra de NFTs o en el almacenamiento de metadatos, pero no ofrecen un sistema completo de certificación de autoría basado en la integridad del archivo y su inmutabilidad desde el momento del registro. Además, muchas de estas plataformas no están diseñadas para usuarios sin conocimientos técnicos avanzados.

Ante este contexto, el presente proyecto tiene como motivación desarrollar un sistema descentralizado de registro de obras digitales basado en tecnología blockchain, que permita registrar, consultar, verificar, transferir y subastar obras sin depender de entidades centrales. La propuesta contempla la implementación de contratos inteligentes en la red Ethereum, el uso de almacenamiento descentralizado mediante IPFS y el diseño de una web intuitiva y accesible para el usuario final, así como una API REST que posibilite integraciones con otros sistemas o aplicaciones.

La finalidad última es democratizar el acceso a la protección de la propiedad intelectual, facilitando un sistema robusto, transparente e internacional, donde la prueba de la autoría se base en la inmutabilidad del hash, la trazabilidad de las transacciones y el timestamp en la blockchain. Todo ello sin renunciar a la facilidad de uso, y ofreciendo una alternativa viable, escalable y alineada con el paradigma de la Web 3.0.

4.2 OBJETIVOS

El propósito general de este trabajo de Fin de Grado es diseñar e implementar una plataforma web 3.0 que permita a cualquier usuario registrar obras digitales en la blockchain, garantizando su integridad, autoría y trazabilidad de forma descentralizada, segura y accesible. Esta solución busca reducir la dependencia de instituciones centralizadas, simplificar el proceso de registro y permitir la verificación pública de los datos asociados a cada obra.

Para cumplir este propósito, se han definido los siguientes tres objetivos:

1. **Garantizar la autoría y la integridad de obras digitales** mediante el uso de tecnologías blockchain e IPFS, evitando manipulaciones y proporcionando una prueba pública e inmutable.
2. **Facilitar a cualquier usuario el registro y gestión de sus obras** a través de una interfaz intuitiva que no requiere conocimientos técnicos, promoviendo la democratización del acceso a tecnologías Web3.

3. **Demostrar la viabilidad técnica de un sistema descentralizado** para el registro de propiedad intelectual, mediante la integración funcional de contratos inteligentes, almacenamiento descentralizado y autenticación segura.

4.3 METODOLOGÍA

Para el desarrollo de este proyecto se ha seguido una metodología ágil basada en el marco Scrum, adaptada a las particularidades de un trabajo individual. Aunque Scrum está concebido para equipos, sus principios han sido aplicados de forma personal para organizar y priorizar tareas, con una estructura iterativa e incremental en el desarrollo.

El proyecto se ha dividido en módulos funcionales y fases temporales, que se han planificado como "sprints" informales. Cada sprint ha consistido en una serie de objetivos concretos que debían cumplirse en plazo relativamente cortos, y al final de cada uno se ha validado su funcionamiento para asegurar la calidad del sistema.

Fases de desarrollo:

- Noviembre a Diciembre – Backend y contratos inteligentes: en este primer sprint se configuró el servidor Express (API REST), se conectó con la red de pruebas Ethereum (Sepolia) mediante Infura y Metamask, y se desplegó el primer contrato inteligente (*RegistroObras.sol*[31]) en Solidity usando Remix IDE. Además, se definieron las estructuras de datos y lógica de registro de obras. También en diciembre se redactó el Anexo B[32] del proyecto, centrado en la motivación y los objetivos.
- Febrero a Abril – Desarrollo general del sistema y arquitecturas Web 3.0: durante estos meses se llevó a cabo el diseño general de la plataforma y el desarrollo iterativo del sistema. Se implementaron los contratos inteligentes compatibles con el estándar

[31] Nombre que recibió el contrato inteligente desplegado en Remix IDE.
[32] Documento en el que se especificaba el objetivo y las tareas a realizar en el Trabajo de Fin de Grado

ERC-721, permitiendo registrar obras como tokens no fungibles (NFT). También se integró el sistema de autenticación de usuarios, utilizando bcrypt para el cifrado de contraseñas, JWT para la generación de tokens de sesión y cookie-parser para su gestión en el cliente. A nivel frontend, se construyó una interfaz inicial con HTML, EJS y JavaScript, habilitando la conexión con MetaMask, la subida de archivos a IPFS mediante Pinata, y la interacción con la blockchain a través de ethers.js.

- Abril – Presentación técnica intermedia: se realizó una presentación explicando la arquitectura, los componentes utilizados (backend, contratos, IPFS, autenticación), así como las funcionalidades ya implementadas y los próximos pasos.
- Mayo – Finalización del frontend, sistema de pujas y mejoras de UX[33]: en esta última etapa se desplegó un contrato inteligente adicional dedicado a gestionar subastas de NFTs mediante lógica on-chain. Se mejoró la experiencia de usuario (UX) con una interfaz más clara, botones funcionales, feedback visual (spinners de carga[34], mensajes de confirmación) y navegación optimizada. Se terminó de integrar el diseño completo del frontend, garantizando una interacción fluida entre los distintos componentes del sistema.

4.4 *ESTIMACIÓN ECONÓMICA*

En este apartado se presenta un análisis económico detallado del desarrollo del presente Trabajo de Fin de Grado. Este análisis se divide en dos grandes bloques: por un lado, se estudian los costes materiales, es decir, aquellos asociados a los recursos físicos y tecnológicos utilizados; por otro, se analizan los costes humanos, que hacen referencia al tiempo de trabajo y la mano de obra necesaria para lleva a cabo el proyecto.

[33] Mejoras en la interfaz de usuario (User experience)
[34] Cuando un usuario realiza una acción que requiere tiempo de carga, sale en la pantalla cargando y anula la acción de cualquier otro botón por parte del usuario.

Además, se incluye una estimación del coste de mantenimiento en caso de que la solución desarrollada se desplegara en un entorno real, como por ejemplo un servidor de cloud[35] en Amazon Web Services (AWS). Finalmente, se exploran diferentes escenarios de comercialización con el objetivo de valorar la viabilidad económica del sistema y calcular los posibles beneficios o pérdidas que podría generar si se lanzara al mercado.

4.4.1 RECURSOS MATERIALES

En cuanto a los recursos materiales que han sido utilizados en el desarrollo de este Trabajo de Fin de Grado, han sido un ordenador portátil y un monitor externo. Las especificaciones técnicas de ambos dispositivos se encuentran recogidas en la Tabla 1 y la Tabla 2.

Ordenador Portátil Lenovo Ideapad 3 14ITL6	
Procesador	Intel® Core™ i5-1135G7 @ 2.40GHz
RAM[36]	8 GB DDR4
Sistema operativo	Windows 11 Home
Tipo de sistema	64 bits
Precio estimado	**579,00€**

Tabla 1. Especificaciones del ordenador portátil utilizado

Monitor ODYS i27-Q-180	
Tamaño de Pantalla	27 pulgadas
Resolución	WQHD (2560 x 1440)
Frecuencia de refresco[37]	180 Hz

[35] Modelo de computación que permite acceder a recursos como almacenamiento, servidores o aplicaciones a través de internet, sin necesidad de gestionarlos físicamente en un dispositivo local.
[36] Memoria volátil que almacena temporalmente los datos y programas que el ordenador está utilizando en ese momento para permitir un acceso rápido.
[37] Cantidad de imágenes que puede enseñar en un segundo.

Tipo de respuesta	1ms (MRPT)
Tipo de panel	Fast IPS
Precio estimado	**199,00 €**

Tabla 2. Especificaciones del monitor utilizado

Ambos dispositivos han sido herramientas esenciales durante todas las fases del proyecto, desde la implementación del backend hasta el diseño del frontent y la redacción de este documento. Aunque estos equipos no han sido adquiridos exclusivamente para este trabajo, se ha realizado una estimación proporcional del coste de uso basado en la vida útil de los dispositivos. Según las tablas de amortización publicadas por la Agencia Tributaria española [27] los ordenadores y periféricos informáticos suelen tener una vida útil estimada de 4 años (35.040 horas).

Sin embargo, dado que el monitor fue adquirido en Alemania, podrían existir diferencias normativas en su país de origen. No obstante, para mantener coherencia con los criterios utilizados por la Agencia Tributaria española, se ha aplicado la misma base de cálculo. Durante el desarrollo de este trabajo se ha estimado un uso aproximado de 700 horas, lo cual representa un porcentaje reducido sobre la vida útil total de ambos dispositivos. A partir de ello, se ha calculado el valor amortizado proporcional, resaltando en un coste de uso de aproximadamente 11,55 € para el portátil (Ecuación 2) y de 3,96 € para el monitor (Ecuación 4):

$$Portátil\ (coste\ por\ hora) = 579{,}00€/35.040\ horas\ \approx 0{,}0165\ €/hora$$

Ecuación 1. Coste del portátil por hora

$$Portátil\ (coste\ en\ el\ proyecto) = 700\ horas\ *\ 0{,}0165\ \approx 11{,}55€$$

Ecuación 2. Coste del portátil en el proyecto

$$Monitor\ (coste\ por\ hora) =\ 199{,}00€/35.040\ horas\ \approx 0{,}0056\ €/hora$$

Ecuación 3. Coste del monitor por hora

$$Monitor\ (coste\ por\ hora) = 700\ horas\ *0{,}00567\ horas\ \approx 3{,}96€$$

Ecuación 4. Coste del monitor en el proyecto

Esto supone un coste total de:

$$Coste\ total\ empleado\ en\ el\ proyecto = 11{,}55€ + 3{,}96€ = 15{,}51\ €$$

Ecuación 5. Coste material total empleado en el proyecto

Podemos concluir que con todo el material que se ha empleado en el proyecto el costo total es de 15,51€ (Ecuación 5).

Aunque durante el desarrollo del proyecto se ha trabajado de manera local con recursos personales, en un escenario de producción real sería necesario desplegar la aplicación en un entorno en la nube. Para este propósito, se ha estimado el coste mensual aproximado si se optará por utilizar los servicios de Amazon Web Services (AWS).[38]

Concretamente, se tendría que desplegar el backend sobre una instancia EC2[39] tipo t3.small (suficiente para un entorno de pruebas o baja carga), alojar la base de datos en un servicio RDS[40] básico (como PostgreSQL[41] db.t3.micro), almacenar archivos estáticos en S3[42] y servir la aplicación web desde un bucket estático. Además, habría que tener en cuenta los costes de transferencia de datos y el uso de servicios complementarios como Route 53 (DNS)[43] o certificado SSL[44].

La estimación mensual detallada se especifica en la siguiente Tabla 3:

Servicio	*Descripción técnica*	*Coste*
EC2 (instancia t3.small)	Backend de la API REST (24/7)	15,2 €

[38] plataforma de servicios en la nube que ofrece recursos informáticos, almacenamiento, bases de datos y otras funcionalidades bajo demanda.
[39] servicio de AWS que permite lanzar y gestionar servidores virtuales en la nube para ejecutar aplicaciones de forma flexible y escalable.
[40] Relational Database Service es un servicio de AWS que permite configurar, operar y escalar bases de datos relacionales en la nube de forma automatizada y gestionada.
[41] Sistema de gestión de bases de datos relacional y de código abierto, conocido por su robustez, escalabilidad y soporte avanzado para consultas SQL complejas.
[42] Simple Storage Service es un servicio de almacenamiento en la nube que permite guardar y recuperar cualquier cantidad de datos desde cualquier lugar de forma segura, escalable y duradera.
[43] Traduce dominios a direcciones IP.
[44] Cifra la conexión entre el navegador y el servicio web.

RDS (PostgreSQL)	Base de datos relacional	12,00 €
Amazon S3 (hosting web)	Almacenamiento y despliegue frontend	2,00 €
Transferencia de datos	Salida de datos estimada (5 GB/mes)	0,4 €
Route 53 + SSL	Dominio y certificado SSL	3,00€
Total mensual estimado:		**32,60 €**

Tabla 3. Coste mensual servidor AWS

En lo referente al software utilizado durante el desarrollo de este Trabajo de Fin de Grado, todos los programas empleados han sido gratuitos y de código abierto, lo que ha permitido evitar cualquier tipo de gasto en licencias comerciales. La única excepción es el sistema operativo preinstalado en el ordenador portátil, cuyo coste ya se encuentra integrado en el precio del propio equipo, por lo que no se considera un gasto adicional.

En consecuencia, el coste total asociado al software puede considerarse nulo en el contexto del desarrollo académico. No obstante, en un escenario de despliegue real en la red principal de Ethereum, sí existiría un coste asociado al registro de contratos inteligentes, estimado en aproximadamente 225 € por contrato, según las tarifas actuales de gas en 2024. Este sería el único gasto adicional significativo vinculado al software en un entorno de producción.

Una vez analizados los recursos materiales empleados durante el desarrollo del Trabajo de Fin de Grado, incluyendo la amortización proporcional del portátil y el monitor, así como la estimación de costes si se desplegara la solución en un entorno de producción basado en Amazon Web Services (AWS), se puede concluir que los costes asociados van en aumento como se muestra en la siguiente Tabla 4:

Tipo de gasto	*Concepto*	*Costo*
Desarrollo	Ordenador portátil	11,55 €
	Monitor externo	3,96 €
	Despliegue contato	225 €

Mantenimiento	AWS	391,2 € /año
Total (anual)		**631,71 €**

Tabla 4. Gasto total

4.4.2 RECURSOS HUMANOS

Para llevar a cabo el desarrollo de este Trabajo de Fin de Grado, ha sido necesaria la implicación de un único perfil profesional: un desarrollador full-stack, encargado de todas las fases del proyecto, desde la redacción de contratos inteligentes y backend, hasta la integración del frontend, así como de tareas de documentación y pruebas funcionales. Dado que se trata de un proyecto académico realizado de forma individual, no ha sido necesaria la participación de peritos, diseñadores u otros perfiles complementarios.

Según el portal de empleo de Glassdoor[45], el salario medio mensual de un desarrollador junior en España ronda los 1.700 € mensuales. Considerando que el tiempo estimado de trabajo ha sido de 5 meses, el coste estimado asociado a los recursos humanos sería el reflejado en la siguiente Tabla 5:

Concepto	Coste	Tiempo trabajado	Coste total
Desarrollador full-stack	1700 €/mes	5 meses	8.500 €

Tabla 5. Coste que supondría un desarrollador

Adicionalmente, para garantizar el correcto funcionamiento y evolución del sistema en un entorno de producción, sería necesario contemplar el coste de mantenimiento y soporte técnico. Una opción realista y conservadora consiste en la contratación de un desarrollador con dedicación parcial para asumir tareas como la gestión de incidencias, la actualización de dependencias, la implementación de mejoras funcionales y el soporte a usuarios.

[45] https://www.glassdoor.es/Sueldos/programador-junior-sueldo-SRCH_KO0%2C18.htm

En este caso, el coste anual estimado para dicho mantenimiento se estima en 9.000 € anuales[46], considerando una jornada reducida compatible con el volumen previsto de usuarios y transacciones.

Este planteamiento permitiría ofrecer un servicio con garantías de continuidad y calidad, sin depender exclusivamente de colaboradores voluntarios ni de la comunidad.

4.4.3 PRESUPUESTO TOTAL

Una vez analizados los costes derivados del desarrollo y mantenimiento de la plataforma, es posible realizar una estimación global del presupuesto, así como de evaluar su viabilidad económica en distintos escenarios del mercado. Para ello se parte de una base realista que contempla:

- Los costes materiales (uso amortizado de equipos).
- Los costes humanos (desarrollador durante el desarrollo y mantenimiento posterior).
- Los costes operativos (hosting web, base de datos y consumo de gas si finalmente se utiliza Ethereum real).

4.4.3.1 Costes iniciales de desarrollo

Los principales gastos iniciales están relacionados con el uso de recursos informáticos personales amortizados (portátil y monitor), junto con el coste estimado del trabajo del desarrollador a lo largo del proceso. El coste total del desarrollo queda reflejado en la siguiente Tabla 6:

Concepto	*Coste*
Coste material amortizado	15,51 €
Coste de desarrollador (5 meses)	8500 €

[46] https://gooapps.es/2025/03/14/cuanto-cuesta-mantener-una-app/

Despliegue del contrato	225 €
Total	**8740,51 €**

Tabla 6. Costes iniciales de desarrollo

4.4.3.2 Costes anuales de mantenimiento

Para mantener operativo el sistema, se ha estimado un coste anual basado en:

- Contratación de un desarrollador en régimen parcial (para soporte técnico y mejoras).
- Infraestructura en la nube (AWS).

El total se muestra en la siguiente Tabla 7:

Concepto	*Coste anual estimado*
Desarrollador de mantenimiento	9000 €
Despliegue en AWS	391,2 €
Total	**9391,2 €**

Tabla 7. Costes anuales de mantenimiento

4.4.3.3 Estimación de ingresos y retorno

Tras analizar los costes iniciales y los costes humanos asociados al desarrollo de este Trabajo de Fin de Grado, es posible realizar una estimación del retorno económico que podría obtenerse en caso de comercializar con la solución propuesta. Para ello se parte de las estadísticas más recientes del mercado internacional del arte y de las tarifas actuales de uso de la red de Ethereum.

Según el informe *The Art Market 2024* [28], se estima que durante el último año se realizaron aproximadamente 38 millones de transacciones de obras de arte a nivel mundial. A partir de

esta cifra se proponen dos escenarios realistas para evaluar la viabilidad financiera del sistema, basados en cuota de mercados moderadas.

A nivel técnico, cabe destacar que la tarifa media pagada por los usuarios al registrar una obra en la blockchain de Ethereum se ha reducido considerablemente tras la implementación de mejoras como la actualización Decun[47]. En la actualidad, el coste promedio de registrar una obra se sitúa en torno a 1,50 €, y se estima que la plataforma cobraría una comisión del 10 % sobre dicha cantidad, es decir, 0,15 € por obra registrada.

4.4.3.3.1 Escenario optimista: 0,5 % de cuota de mercado

En este primer escenario (Tabla 8) se asume una adopción moderada del sistema, captando el 0,5 % de las transacciones anuales del mercado global.

Concepto	*Valor estimado*
Cuota de mercado	0,5 %
Número de transacciones	190.000
Ingreso medio por transacción	0,15 €
Ingresos anuales	28.500€
Gastos anuales estimados	9391,2 €
Beneficio anual estimado	**19.108,8 €**

Tabla 8. Escenario optimista: 0,5% de cuota de mercado

4.4.3.3.2 Escenario conservador: 0,1% de cuota de mercado

[47] Introducen mejoras para reducir las emisiones y preparar la red para una mayor escalabilidad.

En este caso más prudente (Tabla 9), se proyecta una adopción mínima inicial del sistema, equivalente al 0,1 % de las transacciones globales de arte.

Concepto	*Valor estimado*
Cuota de mercado	0,1 %
Número de transacciones	38.000
Ingreso medio por transacción	0,15 €
Ingresos anuales	5.700 €
Gastos anuales estimados	9391,2 €
Beneficio anual estimado	**-3691,2**

Tabla 9. Escenario optimista: 0,1% de cuota de mercado

Estos escenarios permiten que, incluso con una adopción modesta, el sistema podría alcanzar una rentabilidad positiva si logra superar el umbral del 0.4% de cuota de mercado internacional. Asimismo, los costes iniciales de desarrollo (valorados en aproximadamente 8,500€) se podrían recuperar en menos de un año en el escenario optimista. La escalabilidad del modelo, sumada a la reducción de costes gracias a las nuevas mejoras de Ethereum, refuerzan la viabilidad de este proyecto como solución comercial en el ámbito de la propiedad intelectual descentralizada.

Capítulo 5. SISTEMA DESARROLLADO

En este capítulo se describe en detalle el sistema desarrollado en el marco de este Trabajo de Fin de Grado, cuyo objetivo es ofrecer una solución para el registro de obras digitales a través de la tecnología blockchain. El sistema permite a los usuarios subir una obra y registrar dicha obra como token no fungible (NFT) en la red de pruebas Sepolia de Ethereum. Para ello, se ha añadido una plataforma compuesta por tres elementos principales: un contrato inteligente desplegado en la blockchain, un servidor backend con una API REST desarrollada en Node.js, y una interfaz web desde la cual los usuarios pueden interactuar con todo el sistema de forma sencilla y accesible. En la siguiente Figura 14 se muestra un esquema:

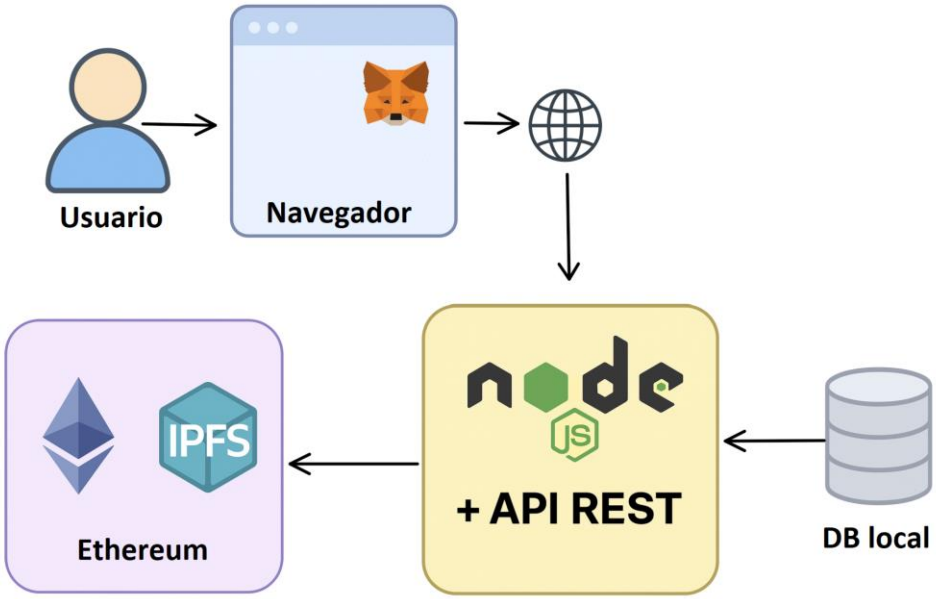

Figura 14. Arquitectura general del sistema

El contrato inteligente, programado en Solidity, se encarga de la gestión y resgistro de los NFTs siguiendo el estándar ERC-721, lo que garantiza la unicidad e interoperabilidad de los tokens generados. Por su parte, el servidor se ocupa de tareas críticas como la validación de los usuarios, la gestión de sesiones mediante JWT, la subida de archivos a IPFS a través de

la API de Pinata y la interacción con el contrato inteligente utilizando la biblioteca Ethers.js. Finalmente, la interfaz web, desarrollada utilizando HTML, CSS y JavaScript con soporte para plantillas EJS, proporciona al usuario las funcionalidades necesarias para registrarse, iniciar sesión, conectar su cartera MetaMask y registrar nuevas obras en la blockchain.

Una de las principales decisiones de diseño ha sido la separación entre el cliente web y el servidor, de modo que ambos puedan evolucionar de manera independiente. Este desacoplamiento permite mantener la arquitectura limpia, escalable y fácil de mantener, además de facilitar posibles integraciones futuras con otras plataformas o aplicaciones móviles. La aplicación web actúa como consumidor de la API REST, mientras que toda la lógica crítica, como la subida a IPFS, el manejo de la base de datos local y la interacción del contrato, se gestiona del lado del servidor.

Con el fin de presentar de forma clara el sistema desarrollado, este capítulo se divide en tres grandes bloques. En primer lugar, se lleva a cabo un análisis funcional, donde se exponen las historias de usuario y casos de uso que han guiado el desarrollo de la solución. A continuación, se describe el diseño general del sistema, detallando su arquitectura, la estructura de los datos tanto en la base de datos como en la blockchain, y los diagramas de secuencia que ilustran las principales interacciones. Finalmente, se analiza la implementación práctica de cada uno de los componentes, explicando cómo se han resuelto los retos técnicos y cómo se ha integrado cada tecnología en el conjunto de la plataforma.

5.1 ANÁLISIS DEL MODELO

El modelo desarrollado en este Trabajo de Fin de Grado surge como respuesta a la necesidad de ofrecer un método más accesible, ágil y fiable para registrar y proteger la autoría de obra digitales, sin depender de intermediarios o procedimientos legalmente complejos. En el contexto actual, los creadores digitales que desean acreditar la autoría de sus obras suelen enfrentarse a procesos burocráticos poco adaptados a los entornos digitales, en los mecanismos tradicionales de certificación, como el registro en oficinas de propiedad intelectual o la firma digital cualificada, resultan costosos, limitados geográfica y

técnicamente inaccesibles para muchos usuarios. Esto se agrava en el ámbito internacional, donde no existe un marco unificado para la validación de certificados ni una infraestructura común que garantice la interoperabilidad entre plataformas de distintos países.

El sistema propuesto en este proyecto planea un enfoque disruptivo al integrar tecnologías descentralizadas como la blockchain de Ethereum y el almacenamiento distribuido mediante IPFS. Gracias al uso de contratos inteligentes, cada obra digital registrada se convierte en un token no fungible único (NFT), vinculado a su correspondiente metadato, que contiene información sobre su autoría y una referencia inmutable a su archivo original. De esta forma, se garantiza la autenticidad, integridad y trazabilidad de la obra sin necesidad de confiar en terceros. Además, al estar basado en una red pública y transparente como Ethereum, cualquier usuario puede verificar de forma autónoma el origen y la propiedad actual de un NFT, independientemente del país en el que se encuentre o del marco legal aplicable.

El sistema desarrollado no requiere de intermediarios humanos ni certificados emitidos por entidades gubernamentales, sino que permite al propio autor, mediante una wallet personal como MetaMask, registrar sus obras y demostrar su autoría de forma descentralizada. Este enfoque contribuye a democratizar el acceso a sistemas de protección de la propiedad intelectual, y planea una vía para reducir el coste y la complejidad en procesos como la compraventa, la cesión o la exposición de obras digitales en mercados globales.

5.1.1 HISTORIAS DE USUARIO

Para comprender mejor las funcionalidades implementadas, es importante identificar los diferentes tipos de usuarios previstos en el sistema. En este caso, se han identificado dos perfiles, usuario registrado y usuario no registrado. A diferencia de otros sistemas tradicionales, aquí no existe un autentificador externo, ya que la propia creación y registro de la obra en blockchain por parte del usuario registrado se considera suficiente evidencia de autoría, gracias a la trazabilidad y la inmutabilidad que proporciona la tecnología utilizada.

A continuación, se describen las historias de usuario correspondientes a cada perfil:

- Como usuario no registrado, quiero poder crear una cuenta proporcionando un nombre de usuario, una dirección de wallet y una contraseña segura.
- Como usuario registrado, quiero acceder a una vista general de la plataforma que me informe sobre sus funcionalidades y posibilidades.
- Como usuario registrado, quiero disponer de un botón para conectar mi cuenta con MetaMask y así poder interactuar con mis obras y realizar transacciones.
- Como usuario registrado, quiero subir una obra seleccionando una imagen y pulsando un botón para que esta quede automáticamente registrada como NFT en la red de pruebas Sepolia y vinculada a mi perfil.
- Como usuario registrado, quiero consultar una galería con todas mis obras registradas, visualizar su token ID en la blockchain y ampliarlas individualmente.
- Como usuario registrado, quiero transferir una obra en mi poder indicando el nombre del destinatario y seleccionando cuál deseo transferir.
- Como usuario registrado, quiero subastar una obra especificando el precio inicial y la duración de la subasta, de modo que todos los usuarios puedan verla e interactuar con ella.
- Como usuario registrado, quiero realizar pujas de forma dinámica en las subastas de aquellas obras que me interesen
- Como usuario registrado, quiero cerrar sesión cuando haya terminado de utilizar la plataforma para proteger la seguridad de mi cuenta.

5.1.2 CASOS DE USO

A continuación, se detallan todos los casos de uso descritos anteriormente en la Figura 15:

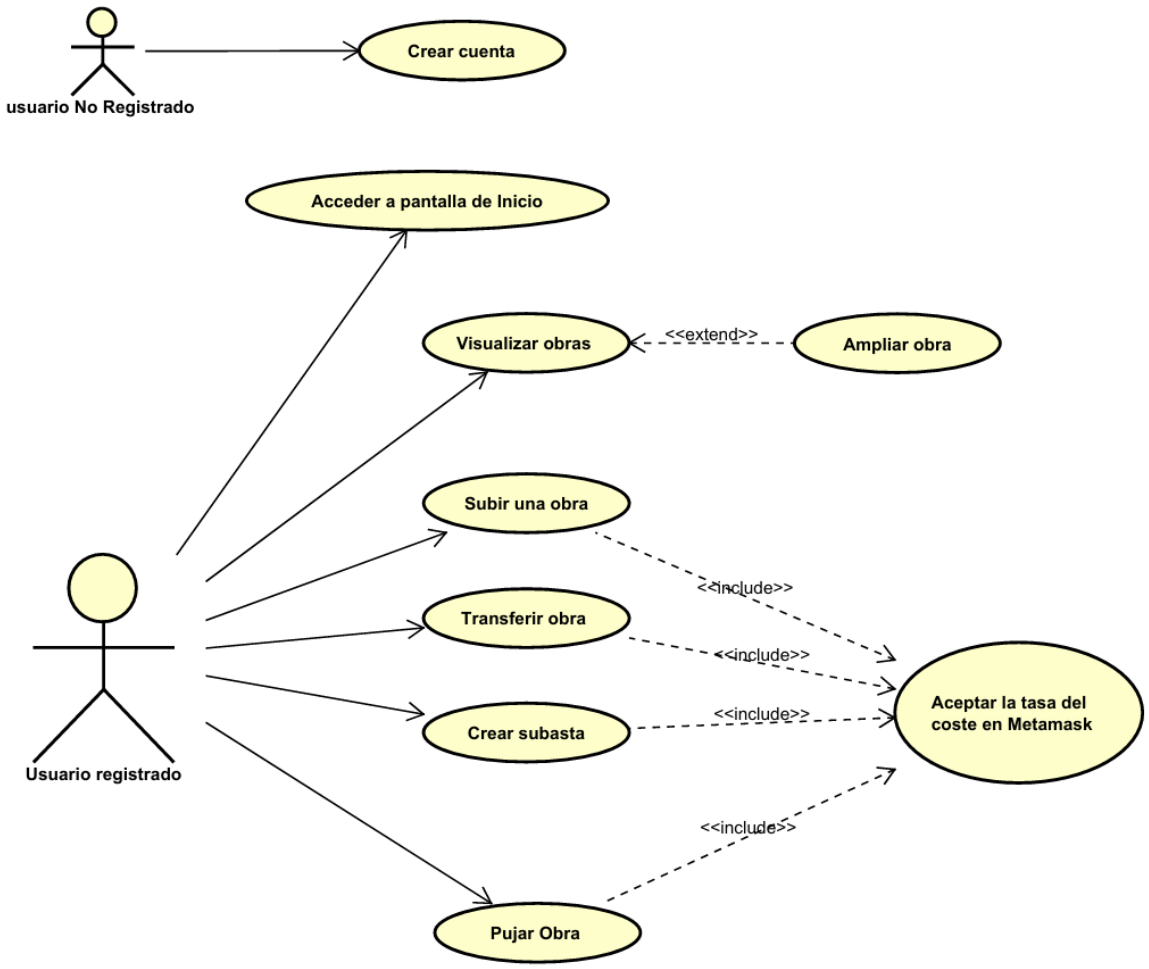

Figura 15. Casos de Uso

5.2 DISEÑO DEL SISTEMA

Una vez establecidos los requisitos funcionales del sistema y definidas las distintas interacciones del usuario, se procede al diseño de la solución. Este diseño tiene como objetivo estructurar los componentes clave que permitirán desarrollar la plataforma de registro y gestión de obras digitales sobre la tecnología blockchain. Para ello, se ha optado por una arquitectura modular basada entre la separación entre la interfaz de usuario, la lógica del servidor y el contrato inteligente desplegado en la red de pruebas de Ethereum Sepolia. En las siguientes secciones se describen el diseño de la arquitectura general, la estructura de los datos tanto en la base de datos como en la blockchain, y los principales flujos del sistema

representados mediante diagramas de secuencia. También se incluye un esquema de navegación que ilustra las diferentes vistas de la interfaz web.

5.2.1 ARQUITECTURA GENERAL

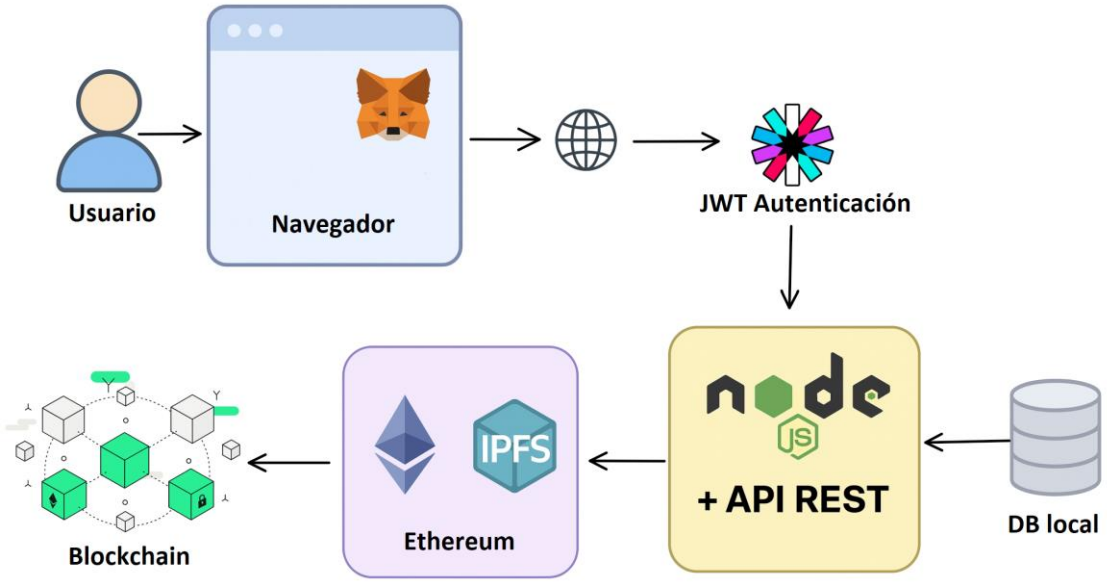

Figura 16. Diseño de la arquitectura del sistema

La arquitectura del sistema desarrollado sigue un enfoque modular basado en tres grandes bloques funcionales: la interfaz web del usuario, el servidor backend con su correspondiente API REST, y la capa de infraestructura descentralizada formada por IPFS y la red Ethereum. Esta separación de responsabilidades no solo facilita la comprensión del sistema, sino que también permite la evolución independiente de cada componente, facilitando el mantenimiento, la escalabilidad y futuras ampliaciones del proyecto.

El primer bloque corresponde al **cliente**, donde el usuario interactúa a través de un **navegador web** compatible con MetaMask, una extensión que actúa como wallet personal y puente hacia la red Ethereum. Desde esta interfaz, desarrollada en HTML, CSS y JavaScript (con plantillas EJS renderizadas por el backend), los usuarios pueden registrarse, iniciar sesión, conectar su wallet, subir obras, visualizar sus tokens y realizar transferencias. La conexión con MetaMask se realiza mediante la biblioteca ethers.js, que permite al

navegador firmar y enviar transacciones a contratos inteligentes desplegados en la red de pruebas Sepolia.

El segundo bloque es el **servidor backend**, implementado utilizando **Node.js** junto con el framework Express.js. Este servidor expone una **API REST** que gestiona las funcionalidades críticas del sistema. Entre sus responsabilidades se encuentran la autenticación de usuarios utilizando contraseñas cifradas con bcryprt y tokens **JWT (JSON Web Tokens)**, la conexión con la **base de datos local**, la gestión del flujo de subida de archivos a **IPFS** a través de Pinata, y la interacción con el contrato inteligente en **Ethereum** mediante la biblioteca ethers. El servidor está preparado para recibir peticiones desde el navegador, procesarlas y responder en función de la lógica de negocio implementada. Además, incluye middlewares como cookie-parser para el manejo de sesiones seguras, y dotenv para la gestión de variables de entorno sensibles como claves privadas o credenciales de API.

El almacenamiento de archivos digitales y sus metadatos se realiza en el sistema descentralizado IPFS (InterPlanetary File Systems), utilizando el servicio Pinata como intermediario para facilitar la subida de archivos y obtener su correspondiente CID (Content Identifier). Este CID se utiliza como tokenURI dentro del contrato inteligente, lo que garantiza que cada obra registrada pueda ser verificada públicamente mediante una dirección IPFS inmutable. Los metadatos de cada obra, generados en formato JSON, incluyen el nombre del autor, una descripción y la URL a la imagen subida, cumpliendo con los estándares del ecosistema NFT. En la siguiente Figura 17 se puede observar la conexión.

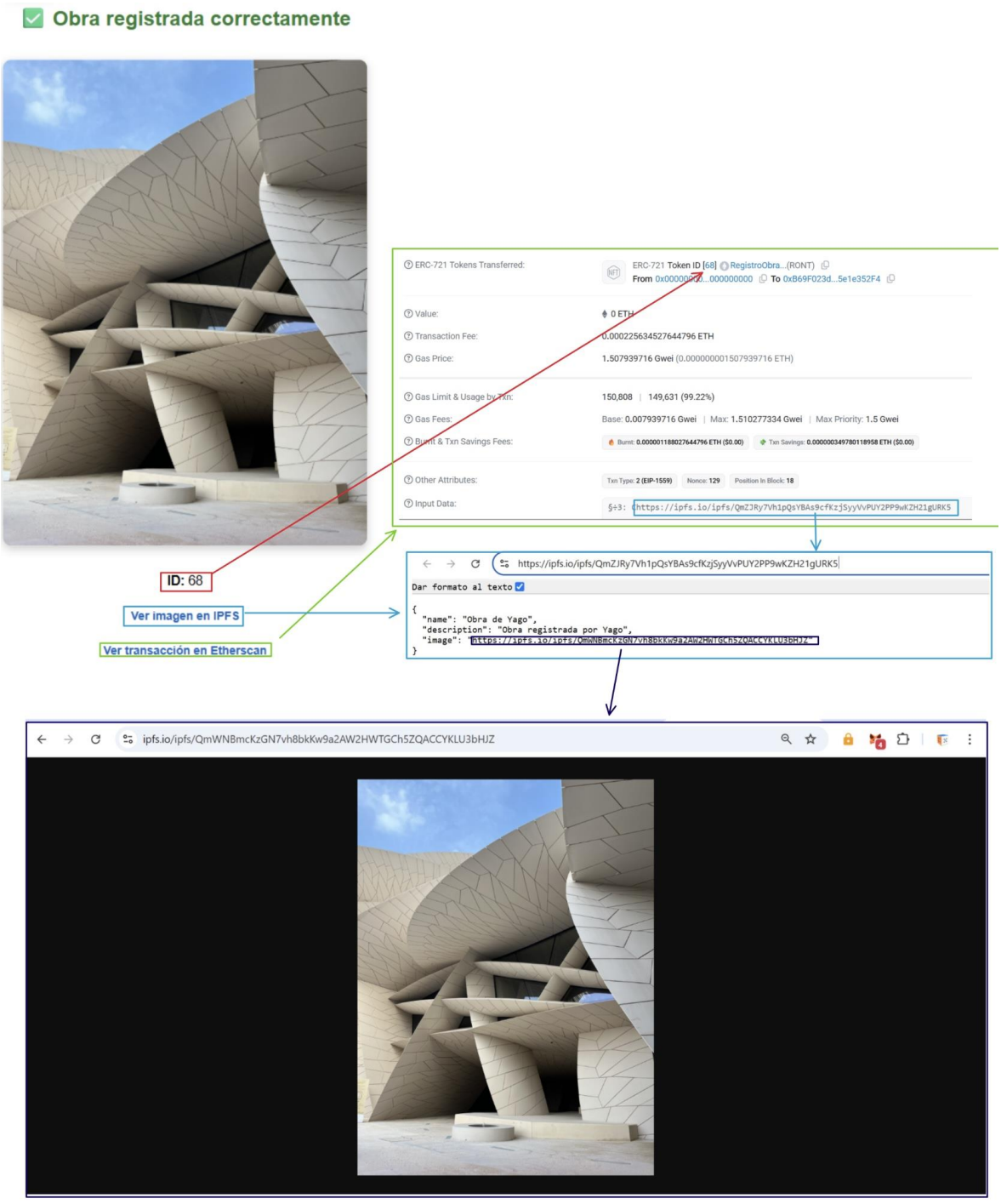

Figura 17. Conexionado de web, blockcahin e IPFS

En cuanto a la persistencia de datos no críticos, el sistema emplea una base de datos local basada en db-local, una solución ligera ordenada a objetos que guarda la información en archivos disco. Esta base de datos almacena los usuarios registrados (con campos como nombre de usuario, wallet y hash de contraseña), permitiendo un acceso rápido para funcionalidades como login, consulta de perfil o validación previa al registro de una obra. Aunque esta información no tiene el mismo nivel de inmutabilidad que la blockchain, permite mejorar el rendimiento y experiencia de usuario en operaciones internas de la plataforma.

Como se observa en la Figura 16, el usuario accede desde su navegador, el cual se comunica con el backend a través de la API REST. El servidor puede consultar la base de datos local, subir contenidos a IPFS y registrar referencias en la blockchain Ethereum mediante contratos inteligentes. Esta arquitectura híbrida permite aprovechar las ventajas de la descentralización sin renunciar a la eficiencia de un backend tradicional.

5.2.2 ESTRUCTURA DE DATOS

El sistema desarrollado emplea una arquitectura de datos híbrida que combina almacenamiento tradicional con persistencia descentralizada. Esta dualidad permite alcanzar un equilibrio entre eficiencia, velocidad de respuesta y garantía de integridad. Por un lado, se utiliza una base de datos local para almacenar información relacionada con usuarios y autenticación. Por otro lado, los archivos digitales y sus metadatos se almacenan en IPFS, mientras que el identificador único de cada obra se registra en la blockchain de Ethereum a través de un contrato inteligente. A continuación, se detalla la estructura de datos correspondiente a cada una de estas capas.

5.2.2.1 Base de datos

Para gestionar la información de los usuarios, se ha implementado una base de datos local mediante el paquete db-local, una herramienta orientada a objetos que permite almacenar y consultar datos de forma sencilla a través de archivos. json persistentes en disco. Esta solución resulta adecuada para entornos de desarrollo o proyectos académicos, donde no se requiere la complejidad de un sistema gestor de bases de datos relacional.

Cada usuario registrado en la plataforma queda representado por una entidad con los siguientes campos:

- _id: identificador único generado mediante UUID[48].
- username: nombre de usuario elegido por la persona durante el registro.
- walletAddress: dirección pública de Ethereum asociada al usuario.
- password: contraseña cifrada utilizando la función de bcrypt y un número definido de salt rounds.

A través de esta estructura se gestionan las operaciones de registro e inicio de sesión, así como la validación de que no existan duplicados en nombres de usuario ni direcciones wallet. Además, el uso de bcrypt garantiza la confidencialidad de las contraseñas almacenadas, incluso en caso de que se accediera directamente al archivo de la base de datos.

Este almacenamiento local no contiene información relacionada con las obras digitales ni con sus transacciones en la blockchain, lo cual asegura que los elementos críticos permanezcan en capas de mayor integridad e inmutabilidad.

5.2.2.2 Blockchain

El sistema utiliza la red de pruebas Ethereum Sepolia para registrar la autoría y existencia de cada obra como un token no fungible (NFT). Para ello, se ha desplegado un contrato inteligente compatible con el estándar ERC-721, que permite asociar un identificador único (tokenId) a cada obra registrada. Este contrato incorpora una función específica, *registrarObra(string memory _tokenURI),* que se encarga de crear nuevos NFTs vinculados a metadatos previamente almacenados en IPFS.

Los datos registrados en la blockchain incluyen:

- tokenId: número secuencial que identifica de forma única cada NFT.

[48] Significa Universally Unique Identifier. Es un estándar para generar identificadores únicos que no requieren de una base de datos central para garantizar su unicidad y que se generar a partir de reglas que combinan timestamps, direcciones MAC y número aleatorios.

- owner: dirección de Ethereum que posee actualmente el token.
- tokenURI: enlace IPFS que apunta a un archivo JSON con los metadatos de la obra.

Este archivo JSON contiene la información pública de la obra, con campos como:

- name: nombre de la obra o referencia al autor.
- description: breve descripción contextual.
- Image: URL en formato *https://ipfs.io/ipfs/{CID}* que permite visualizar la imagen registrada.

Gracias a esta estructura, cualquier usuario puede verificar la existencia, propiedad e integridad de una obra simplemente accediendo al contrato en Etherscan o consultando el tokenURI. Esta trazabilidad garantiza que la obra no ha sido modificada desde su registro y que su autoría está respaldada criptográficamente mediante la wallet con la que fue firmada la transacción.

En la Figura 18 se presenta el diagrama entidad-relación que define la estructura de datos utilizada para representar la relación entre los usuarios registrados en la plataforma y las obras digitales registradas como NFTs en la red de blockchain. Este modelo de base de datos permite almacenar, consultar y mantener un historial completo de las obras creadas, así como de los propietarios asociados a cada token.

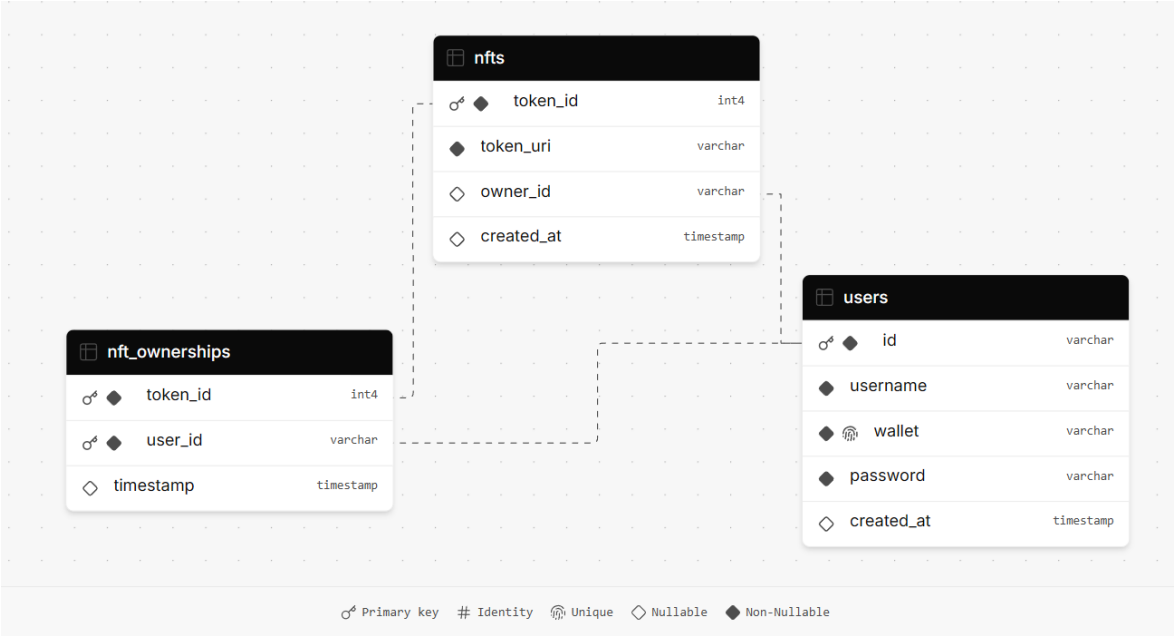

Imagen creada en Postgres Sandbox

Figura 18. Diagrama de clases

5.2.3 DIAGRAMAS DE SECUENCIA

En este apartado se presentan tres diagramas de secuencia que describen el flujo de interacción entre los distintos componentes del sistema para llevar a cabo las operaciones clave de la plataforma: el registro de una obra, la transferencia de una obra a otro usuario y la creación de una subasta. Estos diagramas permiten visualizar de forma estructurada cómo se comunican el usuario, el navegador, MetaMask, el backend, IPFS y los contratos inteligentes desplegados en la red Ethereum Sepolia.

5.2.3.1 Registro de una obra

El primer diagrama Figura 19 representan el flujo completo de registro de una obra digital como NFT en la blockchain. El proceso se inicia cuando el usuario selecciona un archivo desde la interfaz web y presiona el botón de subir obra. El navegador envía el archivo al backend, que lo sube a IPFS mediante Pinata y genera un archivo de metadatos en formato JSON. Este archivo también se sube a IPFS, y su *tokenURI* resultante es enviado de vuelta al navegador. A partir de ese momento, el usuario puede confirmar la operación, lo que

desencadena una transacción firmada a través de Metamask que llama al método *registrarObra(tokenURI)* del contrato inteligente ERC-721. Tras la confirmación de la transacción en la red, se obtiene el tokenId que identifica de forma única el NFT registrado.

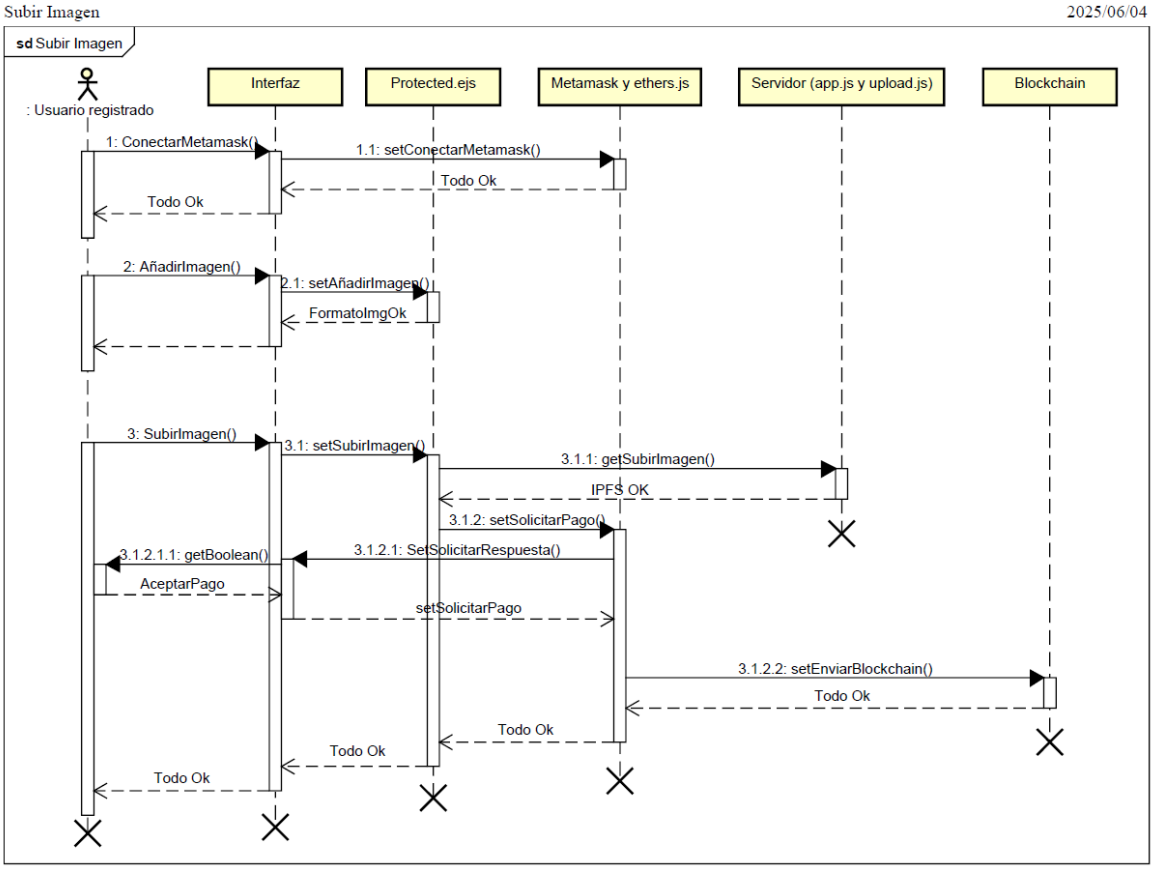

Figura 19. Diagrama de secuencia de subir Imagen

5.2.3.2 Transferencia de una obra

El segundo diagrama (Figura 20) muestra el flujo que se produce cuando un usuario desea transferir una obra en su poder a otro usuario. El proceso comienza cuando el usuario elige la obra que desea transferir y especifica el nombre o dirección wallet del destinatario. El navegador consulta al backend la dirección Ethereum asociada a ese usuario, y si es válida, se solicita a MetaMask que firme la transacción. La operación ejecuta el método *transferFrom* del contrato ERC-721, transfiriendo la propiedad del token. Tras confirmarse la transacción en la red Sepolia, el nuevo propietario queda registrado on-chain.

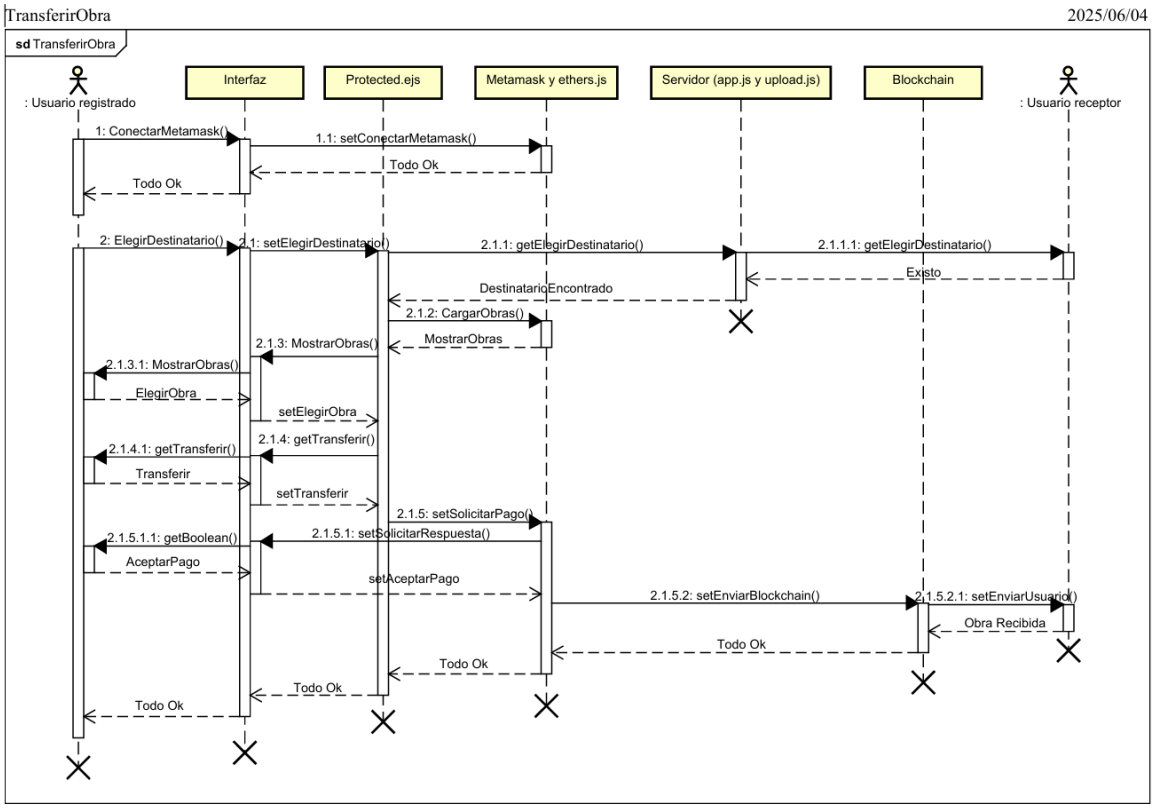

Figura 20. Diagrama de secuencia de realizar Transferencia

5.2.3.3 Creación y puja en una subasta

El tercer diagrama Figura 21 describe el flujo completo de creación de una subasta y participación en ella. En una primera fase, el usuario que posee un NFT accede a la plataforma, selecciona la obra y define los parámetros de la subasta: precio inicial, duración y demás condiciones. Esta información se envía desde el navegador al backend, que valida que el usuario es el propietario legítimo del token. Una vez verificado, se estructura la transacción y se solicita la firma a través de MetaMask. La transacción firmada ejecuta la función del contrato inteligente de subastas que registra dicha obra como subastada en la blockchain. A partir de este punto, la obra queda listada públicamente como disponible para ser subastada.

En la segunda fase, otros usuarios registrados en la plataforma pueden acceder al listado de obras en subasta y consultar sus condiciones. Cuando un usuario desea realizar una puja,

introduce el importe correspondiente y envía la acción desde el navegador. El sistema estructura los parámetros y lanza una transacción que será firmada nuevamente mediante MetaMask. Esta transacción llama a una función del contrato de subastas encargada de resgistrar las ofertas (*pujarObra*, por ejemplo). Si la puja es válida (mayor que la anterior y dentro del tiempo establecido), queda registrada en la blockchain.

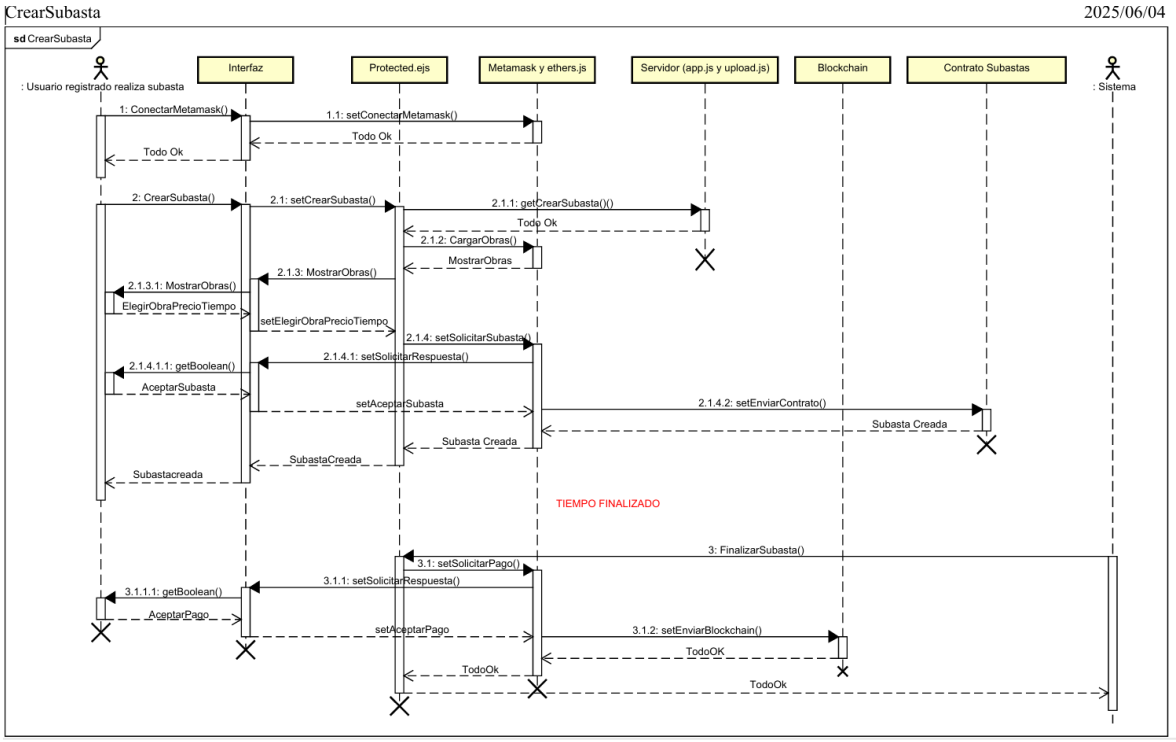

Figura 21. Diagrama de secuencia de Crear&FinalizarSubasta

Este diagrama refleja cómo diferentes usuarios pueden interactuar con una misma subasta, en una arquitectura completamente descentralizada. Gracias al uso de contratos inteligentes, tanto la publicación como la participación en la subasta se gestionan de forma segura, verificable y sin necesidad de intermediarios, garantizando la integridad del proceso de licitación.

Estos tres diagramas reflejan la arquitectura descentralizada del sistema, donde el navegador actúa como intermediario inteligente entre el usuario y los servicios ofrecidos por el backend, IPFS y la blockchain. El uso de MetaMask garantiza que todas las operaciones

sensibles sean firmadas por el propio usuario, mientras que los contratos inteligentes aseguran la ejecución automática, verificable y transparente de las reglas del sistema.

5.3 IMPLEMENTACIÓN TÉCNICA

Una vez descrito el diseño general del sistema, en este apartado se presenta la implementación práctica de sus tres componentes fundamentales: el contrato inteligente, la API desarrollada en Node.js y la interfaz web. Estos tres módulos interactúan de forma coherente para permitir a los usuarios registrar, transferir y subastar obras digitales utilizando la tecnología blockchain e IPFS.

Para la gestión descentralizada de las obras y su registro como NFTs, se ha desarrollado un contrato inteligente en Solidity compatible con el estándar ERC-721 desplegado en la red de pruebas Ethereum Sepolia.

5.3.1 CONTRATO INTELIGENTE (SMART CONTRACT)

5.3.1.1 Registro de Obras NFT

El contrato *RegistroObrasNFT*, desarrollado en lenguaje Solidity y desplegado sobre la red Ethereum Sepolia, representa el núcleo funcional del sistema de registro de obras digitales como tokens no fungibles (NFTs). Está basado en el estándar ERC-721, ampliamente adoptado en el ecosistema blockchain para representar archivos únicos e indivisibles, como obras de arte, documentos o archivos digitales.

Este contrato hereda de la librería *ERC721URIStorage* proporcionada por OpenZeppelin, lo que le permite extender la funcionalidad básica de ERC-721 e incorporar almacenamiento específico para las URIs asociadas a cada token. Esta elección no solo simplifica el desarrollo del contrato, sino que también garantiza compatibilidad con herramientas de terceros y plataformas que interactúan con NFTs.

A continuación, se muestra el código fuente:

```
//SPDX-License-Identifier: MIT
```

```
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.sol";
import "@openzeppelin/contracts/utils/Counters.sol";

contract RegistroObrasNFT is ERC721URIStorage {
    using Counters for Counters.Counter;
    Counters.Counter private _tokenIds;

    constructor() ERC721("RegistroObrasNFT", "RONT") {}

    function registrarObra(string memory _tokenURI) public returns (uint256)  {
        uint256 newTokenId = _tokenIds.current();
        _safeMint(msg.sender, newTokenId);
        _setTokenURI(newTokenId, _tokenURI);
        _tokenIds.increment();
        return newTokenId;
    }
}
```

La función principal del contrato es *registraObra(string memory _tokenURI)*, que permite a cualquier usuario registrado y autenticado mediante su wallet de Ethereum acuñar un nuevo NFT. Al ejecutar la función:

1. Se genera un nuevo identificador único para el token (*tokenId*) utilizando un contador interno (*_tokenIds*) que asegura la unicidad y secuencialidad de los NFTs dentro del contrato.
2. Se ejecuta la función *_safeMint*, que transfiere de forma segura la propiedad del nuevo NFT al usuario que invoca la función (*msg.sender)*. Esta función incluye mecanismos para verificar que el destinatario es capaz de recibir y gestionar NFTs, evitando errores en contrato incompatibles.
3. Se asigna al token generado una URI que apunta a un archivo JSON almacenado en IPFS. Este archivo contiene metadatos de la obra, como el nombre del archivo, el autor, una descripción contextual y una URL de la imagen también alojada en IPFS. Esta URI se registra en la blockchain mediante la función *_setTokenURI*, asociándola permanentemente al token.

Este proceso asegura que, desde el momento en que un usuario sube una obra, dicha información queda vinculada a un identificador inmutable registrado en la cadena de bloques, lo que garantiza la autenticidad, integridad y trazabilidad del contenido.

Cabe destacar que la función *registrarObra* es pública, lo que permite a cualquier usuario de la plataforma, autenticado a través de MetaMask, iniciar el proceso de registro sin necesidad de intermediarios ni validaciones externas. Además, al emplear estándares reconocidos como ERC-721 y almacenar los datos en IPFS, se asegura la interoperabilidad del sistema con marketplaces y exploradores de blockchain como Etherscan.

En conjunto, este contrato representa un mecanismo eficiente, transparente y descentralizado para certificar la existencia y autoría de archivos digitales, dotándolos de un identificador único y verificable en todo momento por cualquier agente de la red.

5.3.1.2 Contrato Inteligente de Subastas

Para extender la funcionalidad del sistema y permitir una interacción dinámica entre usuarios más allá del simple registro de obras, se ha implementado un contrato inteligente de subastas. Este contrato permite subastar obras registradas como NFTs siguiendo una modalidad de subasta inglesa tradicional, ampliamente reconocida por su transparencia y sencillez.

El contrato hereda de *ReentrancyGuard*, lo que impide llamadas anidadas que podrían comprometer los fondos del contrato.

El flujo de trabajo del contrato se articula en tres fases principales:

1. Creación de subasta (*createAuction*): el vendedor, que debe ser propietario de un token NFT (acuñado previamente mediante el contrato *RegistroObrasNFT),* transfiere temporalmente el NFT al contrato de subasta. Junto con esta transferencia, indica el precio mínimo de puja y la duración deseada de la subasta (en segundos). El NFT queda bloqueado dentro del contrato hasta que se finalice el proceso, lo que garantiza que no pueda ser transferido ni manipulado durante la subasta. Esta acción emite el evento *AuctionCreed.*

2. Pujas (*bid*): cualquier usuario puede participar en la subasta enviando una cantidad de ETH superior a la última puja registrada. El contrato verifica que la subasta este activa (es decir, que no haya expirado) y que la nueva puja supere a la anterior. En caso de existir una puja previa, el contrato devuelve automáticamente el importe al pujador anterior, lo que fomenta la confianza en el sistema y evita que los usuarios pierdan sus fondos. El nuevo pujador pasa a ser el actual "ganador provisional" de la subasta, y se emite el evento *NewBid*.
3. Finalización de la subasta (*finalize*): una vez transcurrido el tiempo definido para la subasta, cualquier usuario puede ejecutar la función *finalize*. Si la subasta ha recibido al menos una puja válida, el contrato transfiere el NFT al pujador ganador y entrega los fondos acumulados al vendedor original. En caso de que no se haya recibido ninguna puja, el contrato devuelve el NFT al vendedor. En ambos casos se emite el evento *AuctionEnded*, con la información final de la transacción.

El contrato utiliza una estructura de datos *Lot*, que almacena la información relevante de cada subasta activa:

```
struct Lot {
    address seller;
    uint40  endTime;
    address highBidder;
    uint256 highBid;
}
```

Esta estructura se mapea a través del identificador del token (*tokenId*) en el mapping:

```
mapping(uint256 => Lot) public lots;
```

Gracias a este esquema, el contrato puede gestionar múltiples subastas activas en paralelo, cada una asociada a un NFT distinto.

El contrato está diseñado con criterios de seguridad robustos. La herencia de *ReentrancyGuard* protege contra vulnerabilidades durante operaciones sensibles como la devolución de ETH o transferencias de tokens. Además, se validan condiciones clave antes

de cada operación, como la existencia de la subasta, la validación de los plazos y el importe mínimo requerido.

Asimismo, el contrato hace uso de *safeTransferForm* para garantizar que la transferencia de NFTs solo se realice hacia direcciones que implementen correctamente la interfaz ERC-721, evitando así errores de envío o perdida de tokens.

Este contrato actúa como complemento del contrato *RegistroObrasNFT*. El proceso típico consiste en que un usuario registre su obra como NFT, y posteriormente la seleccione para iniciar una subasta a través de la interfaz web. Desde el frontend, los usuarios interactúan con este contrato mediante MetaMask, firmando digitalmente cada operación, ya sea para crear una subasta, realizar una puja o finalizarla.

En resumen, el contrato proporciona al sistema una capa de dinamismo y descentralización en la compraventa de obras digitales, pemitiendo a los usuarios competir por adquirir NFTs mediante pujas públicas y auditables directamente en la blockchain.

5.3.2 BACKEND Y API REST

El backend del sistema ha sido desarrollado utilizando Node.js y el framework Express, implementando una arquitectura basada en una API REST que expone múltiples endpoints para facilita la interacción entre la interfaz web, la base de datos local y los contratos inteligentes desplegados en la red Ethereum Sepolia.

El objetivo principal del backend es actuar como orquestador de operaciones, permitiendo a los usuarios realizar acciones como registrar obras nuevas, consultar sus NFTs, transferir tokens o iniciar subastas, todo ello de forma segura y eficiente. Además, se encarga de la gestión de usuarios mediante autenticación basada en tokens JWT, almacenamiento cifrado de contraseñas y validación de roles.

5.3.2.1 Arquitectura general

El backend se organiza en los siguientes módulos principales:

- Gestión de usuarios (*user-repository.js*): permite registrar nuevos usuarios, encriptar sus contraseñas con *bcrypt*, verificar credenciales en el login, y gestionar la sesión mediante JWT. Cada usuario queda identificado por su nombre, dirección de wallet y clave cifrada.
- Carga de archivos (*upload.js*): gestiona la subida de obras digitales por parte de los usuarios. Una vez subido el archivo, este se envía a IPFS a través del servicio Pinata, generando un enlace único (*tokenURI*) que posteriormente se vincula como un NFT en la blockchain.
- Interacción con blockchain (*app.js* + *ethers.js*): las funciones del backend que implican interacción con los contratos inteligentes están implementadas utilizando la librería *ethers.js*. Por ejemplo, al registrar una obra, el backend construye la transacción para llamar a *registrarObra(tokenURI)*, que es posteriormente firmada por el usuario desde MetaMask.
- Control de rutas seguras (*protected.ejs*): aunque *protected.ejs* forma parte del frontend, muchas rutas protegidas se comunican directamente con endpoints del backend. Esto permite realizar operaciones como transferencia de tokens, creación de subastas o pujas, verificando siempre que el usuario está autenticado.

5.3.2.2 Seguridad

Para garantizar la seguridad del sistema, se han incorporado las siguientes medidas:

- Autenticación por tokens JWT: al iniciar sesión, el servidor genera un token firmado que se almacena como cookie en el navegador. Este token se incluye automáticamente en las peticiones del usuario a la API y permite validar su identidad sin necesidad de enviar repetidamente las credenciales.
- Hash de contraseñas con *bcrypt*: todas las contraseñas son almacenadas de forma cifrada mediante hashing con salt, lo que garantiza que incluso en caso de exposición de la base de datos, los datos sensibles permanezcan protegidos.

- CORS y control de cabeceras: la rutas del backend han sido configuradas para aceptar únicamente orígenes válidos, evitando así ataques de tipo CSRF[49] o solicitudes no autorizadas desde otros dominios.
- Validaciones del lado del servidor: todas las entradas de usuarios son validadas y sanitizadas en el backend antes de proceder a acciones críticas, como invocar contratos, guardar datos o consultar IPFS.

5.3.2.3 Funcionalidades clave expuestas por la API

Algunas de las rutas más representativas de la API son:

Método	***Ruta***	***Funcionalidad***	***Recibe (Body)***	***Respuesta***
POST	*/api/register*	Registra un nuevo usuario	*username*, *walletAddress*, *password*	*id* del nuevo usuario o mensaje de error
POST	*/api/login*	Autentica usuario y genera token JWT	*username*, *password*	*token* JWT + datos del usuario o mensaje de error
POST	*/api/upload*	Sube una obra a IPFS y devuelve su CID	FormData con *miArchivo* y *autor*	*tokenURI* con enlace a metadatos en IPFS
POST	*/api/registrar-obra*	Añade un nuevo NFT en la blockchain	*tokenURI* (enlace IPFS)	Confirmación del registro + enlace a Etherscan
POST	*/api/transferir-obra*	Transfiere un NFT a otro usuario	*tokenId*, *destinatario*	Confirmación de transferencia o error

[49] Es un ataque que engaña a un usuario autenticado para que ejecute acciones no deseadas en una aplicación web en la que no está autenticado.

GET	*/api/mis-obras*	Consulta las obras registradas por el usuario autenticado	JWT en cookies	Lista de NFTs con *tokenId*, *tokenURI*, etc.
POST	*/api/crear-subasta*	Inicia una subasta de una obra propia	*tokenId*, *precioInicial*, *duracion*	Confirmación + ID de subasta
POST	*/api/pujar*	Realiza una puja sobre una subasta activa	*subastaId*, *valorPuja*	Confirmación de puja o error (si es menor o fuera de tiempo)

Tabla 10. Endpoints principales de la API REST del sistema

5.3.2.4 Persistencia y datos locales

El backend utiliza un sistema de persistencia simple basado en archivos .json mediante la librería *db-local*. Esta solución, aunque no es óptima para un entorno productivo, resulta eficaz para entornos académicos, permitiendo simular la funcionalidad de una base de datos relacional con una curva de configuración mínima.

5.3.3 INTERFAZ WEB

La interfaz web del sistema ha sido desarrollada utilizando HTML, CSS y JavaScript, en combinación con EJS (Embedded JavaScript Templates) para la generación dinámica de vistas desde el servidor Express. Esta interfaz permite a los usuarios interactuar de forma sencilla e intuitiva con todas las funcionalidades del sistema, incluyendo el registro de usuarios, subida de obras, visualización de NFTs, transferencias y participación de subastas.

5.3.3.1 Objetivo de la interfaz

El objetivo de la interfaz web es facilitar al máximo la experiencia del usuario sin sacrificar la conexión con el backend ni la seguridad en la interacción con la blockchain. Para ello, se ha buscado una navegación clara, con botones de acción bien definidos y retroalimentación visual tras cada operación relevante (por ejemplo, después de acuñar un NFT o realizar una puja).

5.3.3.2 Navegación y secciones

La web se estructura en distintas secciones, accesibles a través de botones en la parte superior de la pantalla. Estas secciones están gestionadas dinámicamente mediante JavaScript y se alterna sin necesidad de recargar la página completa, simulando así el comportamiento de una SPA (Single Page Application):

- Inicio: explica brevemente la finalidad de la plataforma y permite a usuarios nuevos conectar su billetera o registrarse.
- Mis NFTs: muestra en forma de galería las obras registradas por el usuario. Cada obra se representa como una tarjeta que incluye la imagen, el *tokenId*, y un enlace a su metadata en IPFS.
- Subir Obra: permite seleccionar un archivo de imagen, subirlo automáticamente a IPFS mediate el backend, y registrar la obra como NFT en la blockchain con una única interacción de MetaMask.
- Transferir Obra: muestra las obras del usuario y permite seleccionar una para transferirla a otro usuario existente. El nombre del receptor se valida en tiempo real contra la base de datos.
- Subastas: esta sección se divide en dos:
    - Crear subasta: permite al usuario subastar una de sus obras registradas, indicando duración y precio mínimo.
    - Explorar subastas: lista todas las obras actualmente en subasta. Cualquier usuario puede hacer click sobre ellas y pujar directamente desde la web.

5.3.3.3 Comunicación con el backend

La interfaz web interactúa con el servidor Express a través de llamadas *fetch()* a la API REST. Los datos del usuario y sus NFTs se recuperan mediante peticiones protegidas con tokens JWT, y las acciones como registrar obras o transferir tokens se disparan desde eventos *onClick* asociados a botones en la interfaz.

Además, se ha integrado la librería Ethers.js en el frontend para realizar operaciones que requieren interacción con Metamask, como firmar transacciones para registrar NFTs o pujar en subastas. Gracias a esta integración, el frontend puede:

- Detectar si MetaMask está instalado y conectado.
- Solicitar permiso para acceder a la cuenta del usuario.
- Obtener la dirección de la wallet actual.
- Firmar y enviar transacciones al contrato inteligente desplegado.

5.3.3.4 Experiencia de usuario (UX)

Se ha prestado especial atención a la experiencia del usuario:

- Se muestran alertas visuales (éxito/error) tras cada operación.
- Los formularios están validados antes de su envío.
- Las transacciones que implican pago de gas solicitan confirmación vía MetaMask y muestran el coste estimado.
- El sistema informa al usuario si no tiene obras, si su sesión ha caducado o si intenta transferir a un usuario inexistente.

Esta interfaz completa la arquitectura del sistema permitiendo una experiencia end-to-end, donde un usuario puede registrarse, subir su obra, verificar su existencia en la blockchain, transferirla o subastarla, todo desde una única página web integrada.

5.4 VALIDACIÓN Y SEGURIDAD

Para garantizar la robustez y la fiabilidad del sistema, se han implementado múltiples medidas de validación y seguridad tanto del lado del cliente como en el servidor, así como en los contratos inteligentes desplegados en la red Ethereum Sepolia.

Desde el frontend, se han incorporado validaciones básicas que permiten verificar que los formularios estén correctamente completados antes de enviar una petición al servidor. Estas

validaciones contemplan la obligatoriedad de campos, la estructura de direcciones de wallet y la integridad de archivos subidos.

En el backend, desarrollado con Node.js y Express, se han aplicado validaciones más estrictas. Se comprueba que no existan duplicidades de nombres de usuario o direcciones de wallet, y se controla que los archivos tengan un formato válido antes de interactuar con IPFS o con la blockchain. Además, se protege el acceso a los endpoints críticos mediante autenticación basada en JWT (JSON Web Tokens), evitando así accesos no autorizados.

Las contraseñas de los usuarios son cifradas utilizando la función *bcrypt*, junto con un número configurable de salt rounds, lo cual proporciona resistencia ante ataques de diccionario y fuerza bruta incluso si la base de datos fuese comprometida.

En el contrato inteligente de subastas se ha incorporado la herencia *ReentrancyGuard* de OpenZeppelin, una medida fundamental para proteger el contrato ante ataques de reentrada, especialmente operaciones que implican el reembolso de fondos en ETH.

Por último, la interacción con la blockchain se realiza siempre mediante MetaMask, lo que garantiza que las transacciones sean firmadas de forma explícita por los usuarios. De esta manera, ninguna acción sensible se ejecuta sin el consentimiento expreso del propietario de la cuenta.

Estas medidas combinadas aseguran la confidencialidad de los datos, la integridad de los registros y la autenticidad de las operaciones realizadas en el sistema.

5.5 PRUEBAS Y RESULTADOS

Para verificar la correcta funcionalidad del sistema y validar su comportamiento esperado, se han realizado pruebas tanto mensuales como funcionalidades sobre los distintos componentes desarrollados.

A nivel de contratos inteligentes, se realizaron pruebas exhaustivas en el entorno de Remix IDE, donde se testearon las funciones principales del contrato *RegistroObrasNFT* y el

contrato *EnglishAuction* (el contrato de subastas). Se comprobó que el registro de obras solo era posible si el tokenURI era válido, que los NFTs se acuñaban correctamente y que las pujas en subastas respetaban las condiciones de precio mínimo y duración.

En cuanto al backend, se validaron todos los endpoints expuestos por la API REST. Se realizaron pruebas de registro, login, subida de archivos, creación y consulta de NFTs, transferencias y subastas. Para ello, se utilizaron herramientas como Postman, además de pruebas manuales realizadas desde el frontend con usuarios distintos.

En el apartado del frontend, se comprobó la integración completa con MetaMask. Se validó que las transacciones fueran correctamente firmadas y enviadas, y que los errores fueran gestionados adecuadamente (por ejemplo, cancelaciones del usuario, falta de fondos o errores de red). Además, se testearon las vistas dinámicas generadas por EJS para asegurar la correcta carga y visualización de datos, como las obras registradas, subastas activas o mensajes de sesión.

También se realizaron pruebas de interacción entre capas, como la subida de una imagen, su envío a IPFS vía Pinata, el registro del token en la blockchain y la posterior visualización de la obra en la galería del usuario.

Los resultados obtenidos en todas las pruebas han confirmado el correcto funcionamiento del sistema y su capacidad para ofrecer una solución integrada y fiable de registro, consulta, transferencia y subasta de obras digitales mediante tecnología blockchain.

Capítulo 6. ANÁLISIS DE RESULTADOS

Una vez finalizado el desarrollo del sistema descrito en el Capítulo 5. , se ha procedido a realizar una validación funcional del producto final desde la perspectiva del usuario. El sistema, accesible a través de una interfaz web conectada a una API REST, permite interactuar con la blockchain de Ethereum (red de pruebas de Sepolia) para registrar, visualizar, transferir y subastar obras digitales representadas como NFTs. A lo largo de esta sección se describen los resultados obtenidos al utilizar distintas funcionalidades implementadas, así como la experiencia general del usuario.

6.1 PÁGINA DE INICIO Y REGISTRO

Al acceder por primera vez a la plataforma, el usuario es redirigido a una pantalla de autenticación que le permite iniciar sesión si ya está registrado, o crear una nueva cuenta en caso contrario. Esta vista está diseñada de forma clara como se observa en la Figura 22.

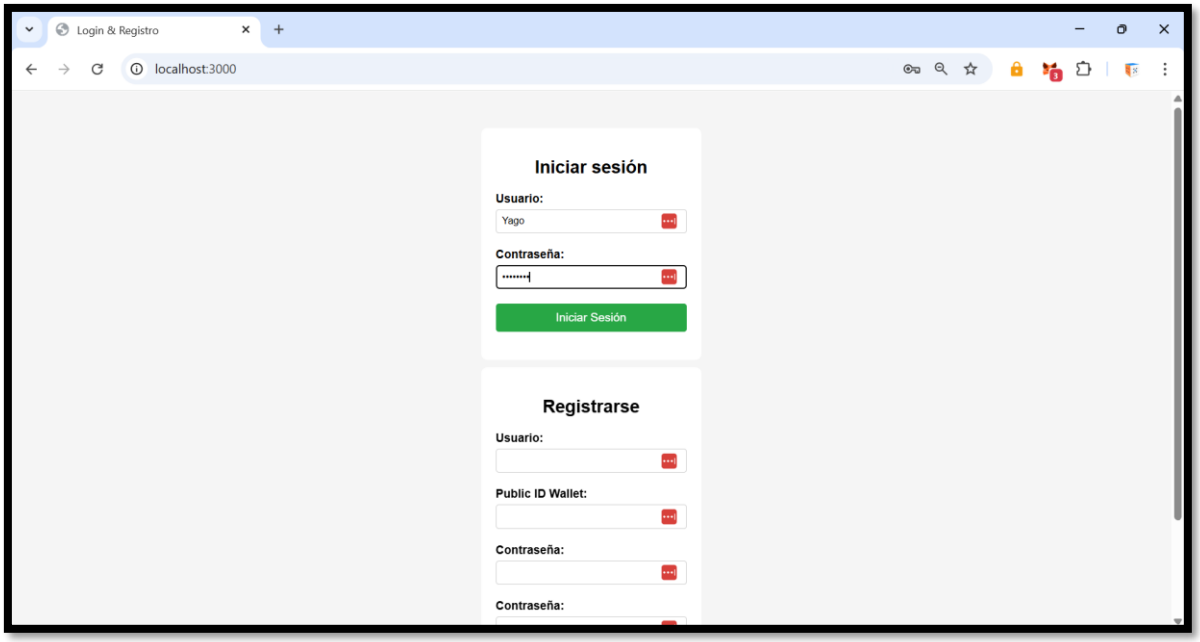

Figura 22. Pantalla de inicio de sesión y registro de usuario

Tras iniciar sesión exitosamente, el usuario es redirigido a la pantalla principal protegida de la plataforma. Como se muestra en la Figura 23, esta vista actúa como centro de operaciones del sistema. En la parte superior se despliega una barra de navegación con acceso a las secciones clave: Inicio, Mis NFTs, Subir Obra, Transferir Obra y Subastas.

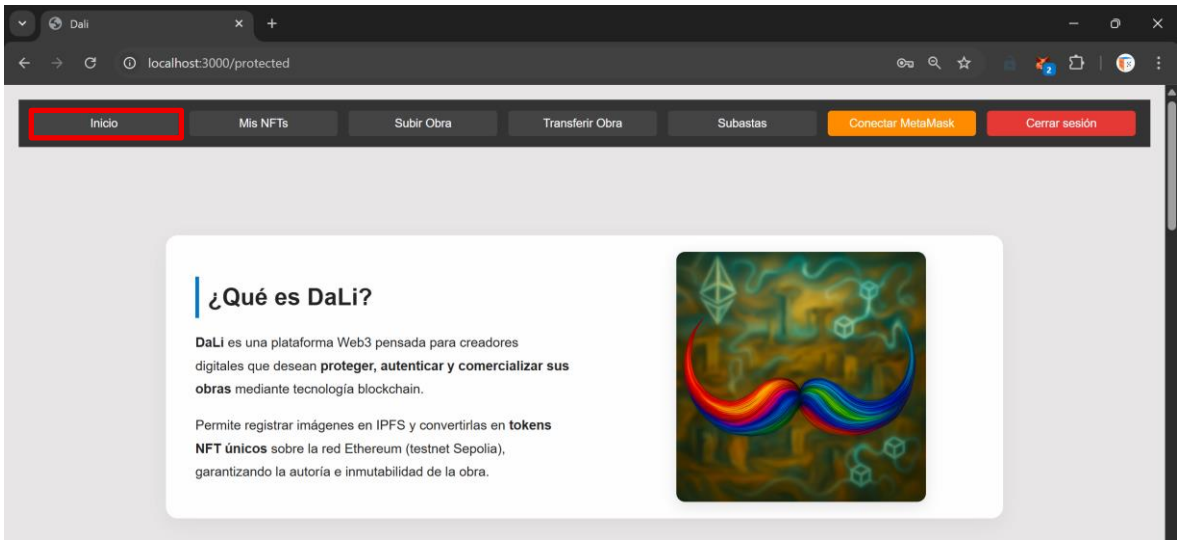

Figura 23. Pantalla de inicio

6.2 VISUALIZACIÓN DE NFTS PROPIOS

Una vez que el usuario ha iniciado sesión y conectado su wallet mediante MetaMask, puede acceder a la sección “Mis NFTs”, donde se visualizan todas las obras digitales que ha registrado como tokens no fungibles (NFTs) en la blockchain de Ethereum (red de pruebas Sepolia). Tal como se muestra en la Figura 24:

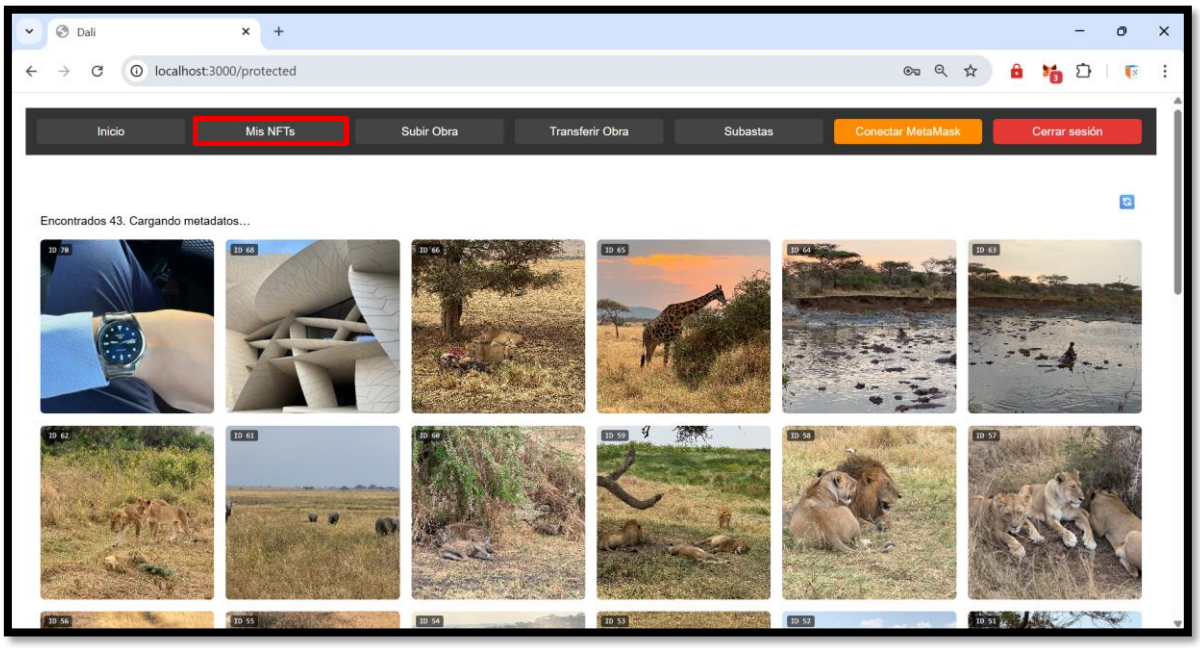

Figura 24. Visualización de NFTs registrados por el usuario.

6.3 SUBIDA DE UNA OBRA COMO NFT

Una de las funcionalidades principales del sistema es la posibilidad de registrar una obra como un NFT en la blockchain de Ethereum. Esta acción permite subir un archivo digital a IPFS, generar su metadata y acuñarlo como un token ERC-721 utilizando el contrato inteligente desplegado en la red de pruebas Sepolia. Una vez seleccionada la imagen, esta se previsualiza en pantalla antes de proceder con el registro, tal como se aprecia en la Figura 25:

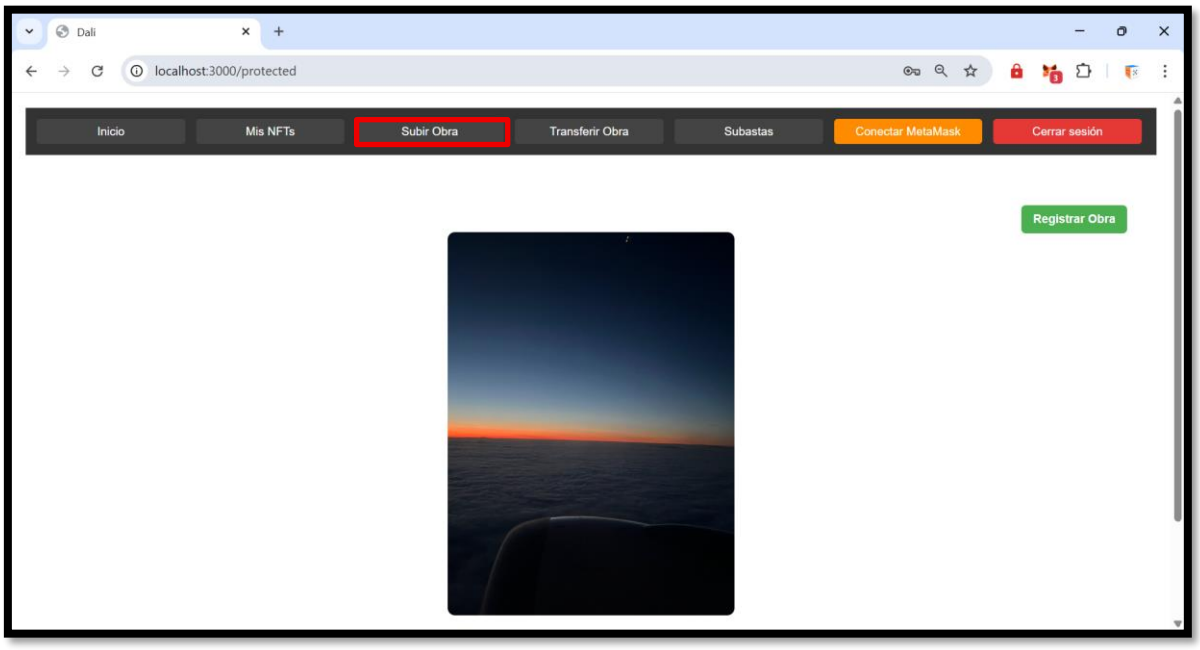

Figura 25. Sección de registrar Obra

Si la transacción se procesa correctamente, el sistema muestra una notificación de éxito y una vista resumen de la obra registrada, como se observa en la Figura 26. Esta vista incluye una previsualización de la imagen, el identificador del NFT (*tokenId*), un enlace al archivo JSON de metadatos en IPFS, y la dirección donde ha quedado la transacción firmada en la blockchain.

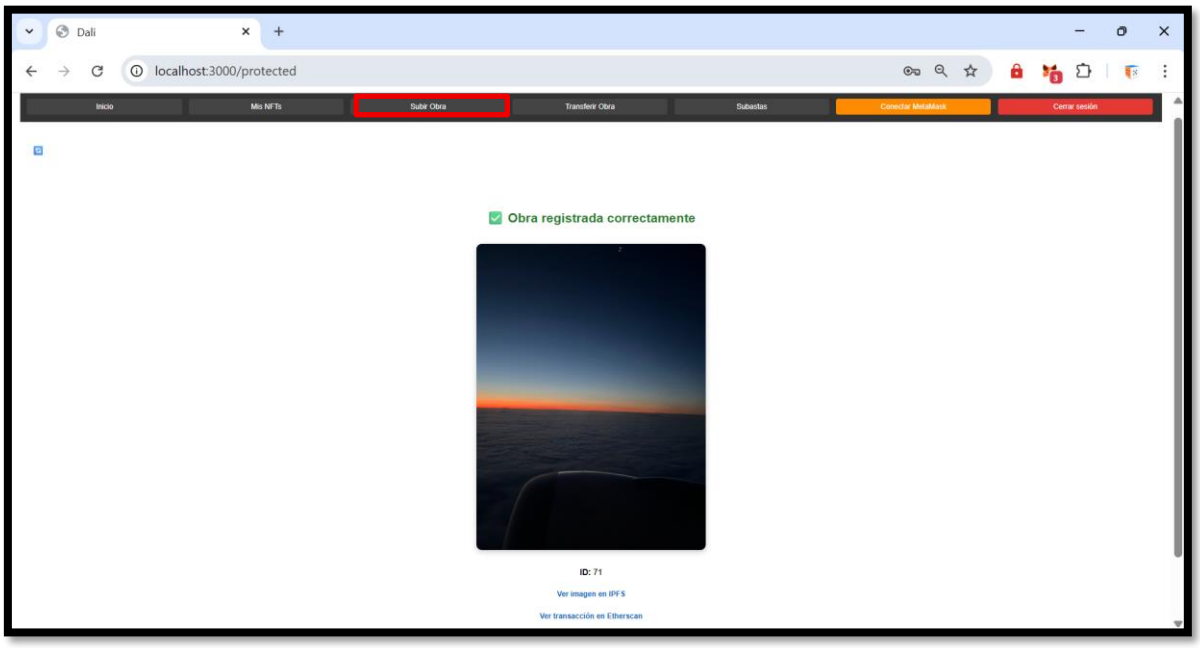

Figura 26. Respuesta: obra registrada correctamente

6.4 TRANSFERENCIA DE OBRAS

El sistema permite a los usuarios transferir obras registradas en la blockchain a otros usuarios de la plataforma. Esta funcionalidad garantiza la trazabilidad de la propiedad de los NFTs y simula un escenario real de compraventa o cesión de derechos digitales.

El proceso comienza en la sección Transferir Obra, donde el usuario —en este caso, Yago— introduce el nombre del destinatario al que desea transferir una obra (Rodri), como se muestra en la Figura 27. El sistema valida dicho nombre contra la base de datos local, comprobando si el usuario existe y recuperando su dirección de wallet Ethereum asociada.

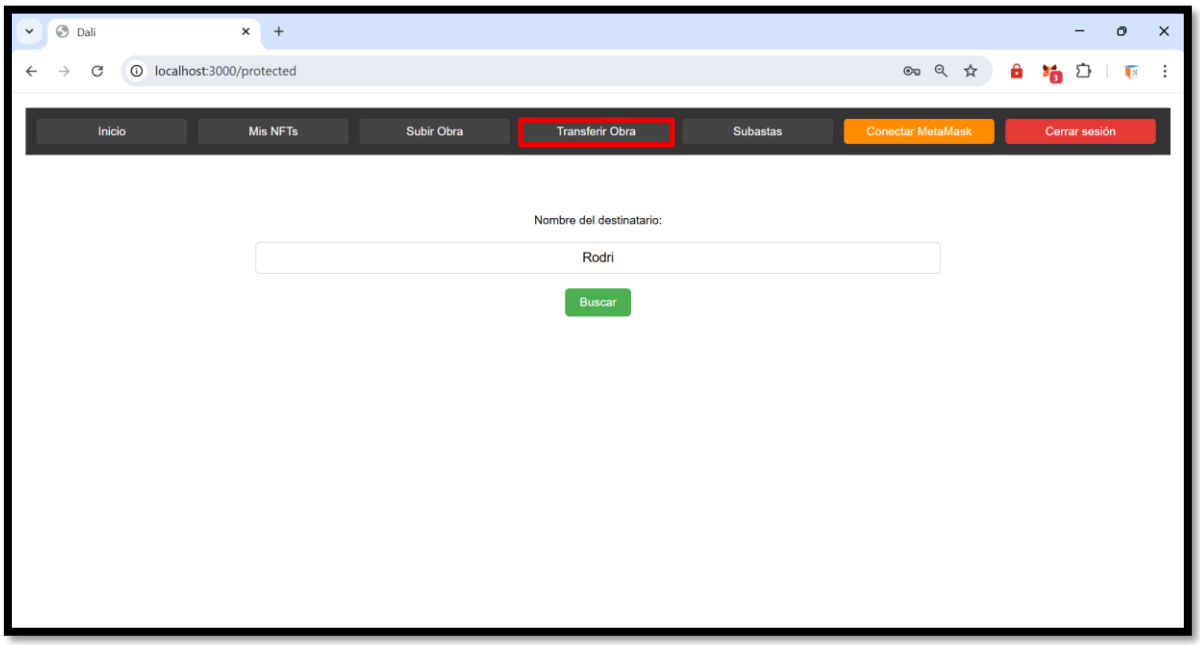

Figura 27. Escribir el nombre del destinatario

El usuario selecciona la obra que desea transferir (Figura 28):

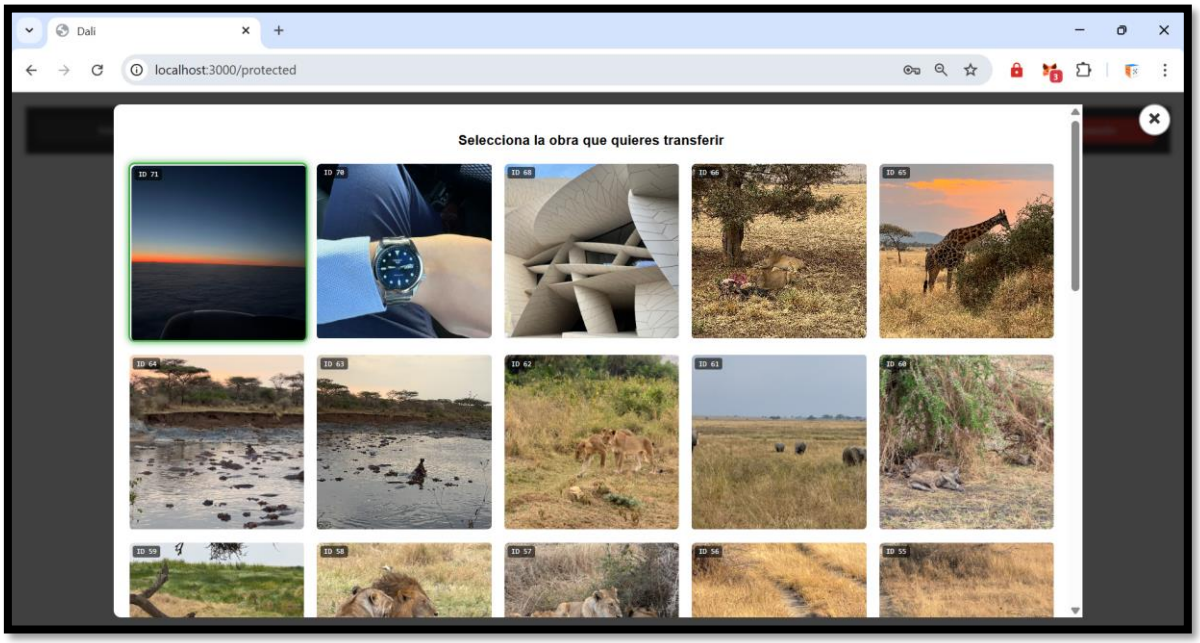

Figura 28. Seleccionar la obra a transferir

6.5 *SUBASTAS: CREACIÓN Y PUJAS*

El sistema desarrollado permite a los usuarios subastar sus obras registradas como NFTs mediante una implementación basada en subastas inglesas. Esta funcionalidad descentralizada permite fijar un precio mínimo, definir una duración determinada y recibir pujas por parte de otros usuarios en tiempo real, todo a través de la blockchain de Ethereum (red Sepolia).

El proceso comienza en la sección Subastas, donde el usuario puede visualizar las subastas activas y acceder al botón "Crear subasta", como se muestra en la Figura 29. En este ejemplo, el usuario Rodri decide subastar una de sus obras disponibles.

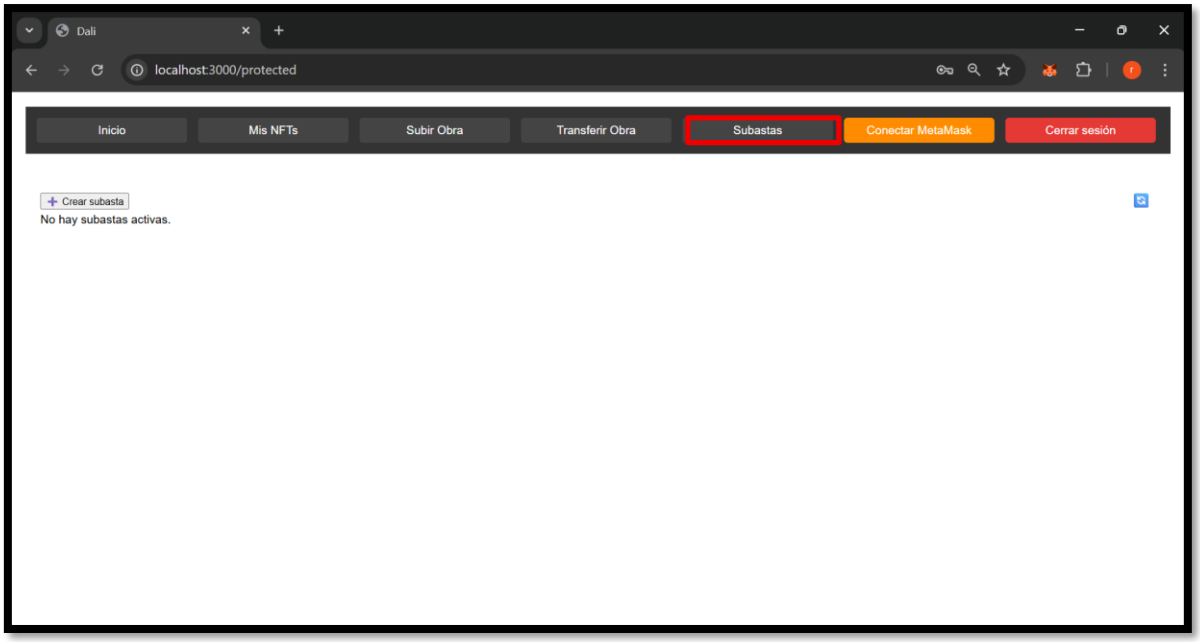

Figura 29. Sección de crear subasta

A continuación, el sistema despliega una galería con los NFTs que Rodri posee. En la Figura 30, se observa la selección de una de las obras registradas previamente. Una vez seleccionada la imagen, el usuario debe establecer el precio mínimo en ETH y la duración de la subasta en minutos y segundos, como se ve en la Figura 31.

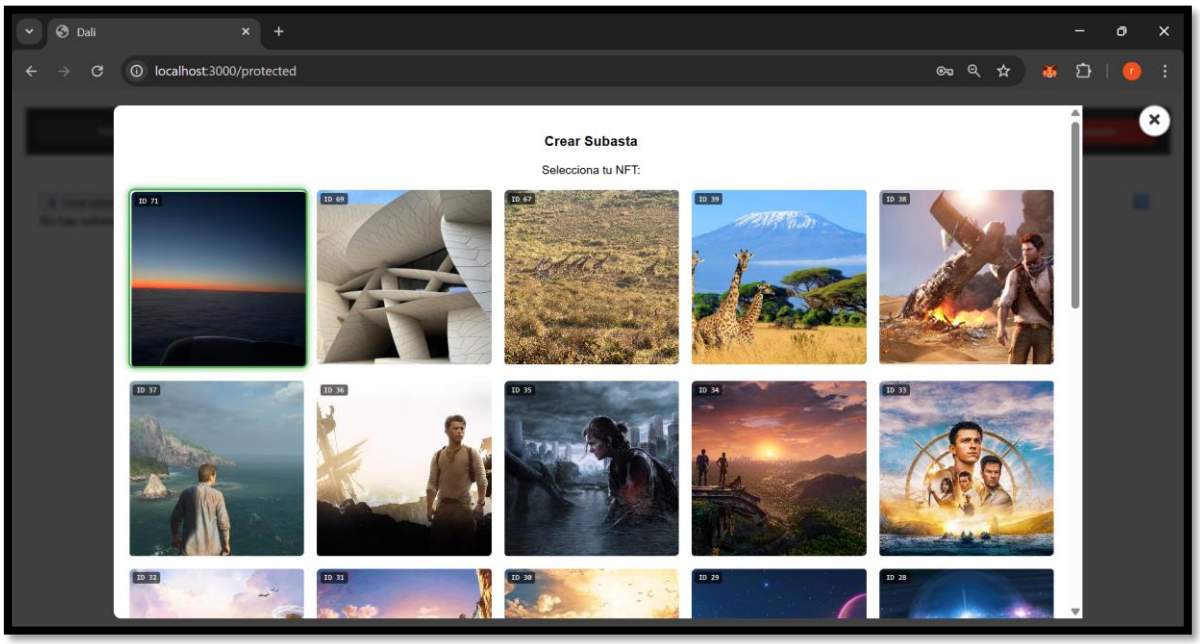

Figura 30. Seleccionar obra a subastar

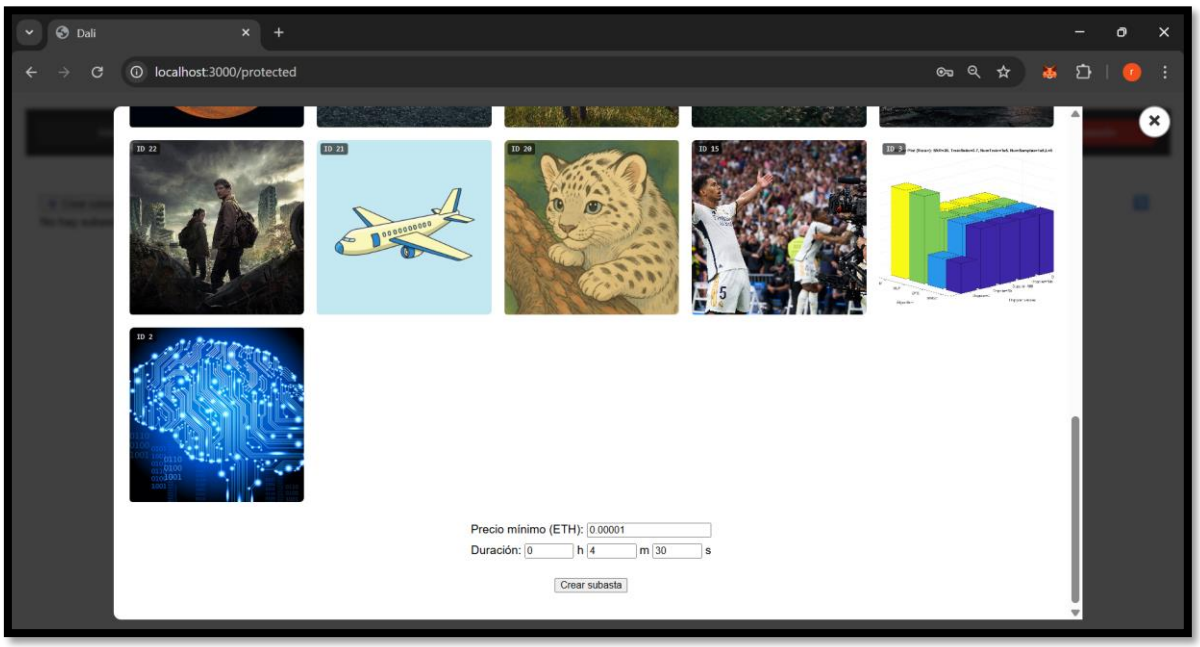

Figura 31. Introducir cantidad y duración

Una vez confirmada la transacción, la subasta queda publicada y visible en la plataforma para todos los usuarios, tal como se muestra en la Figura 32.

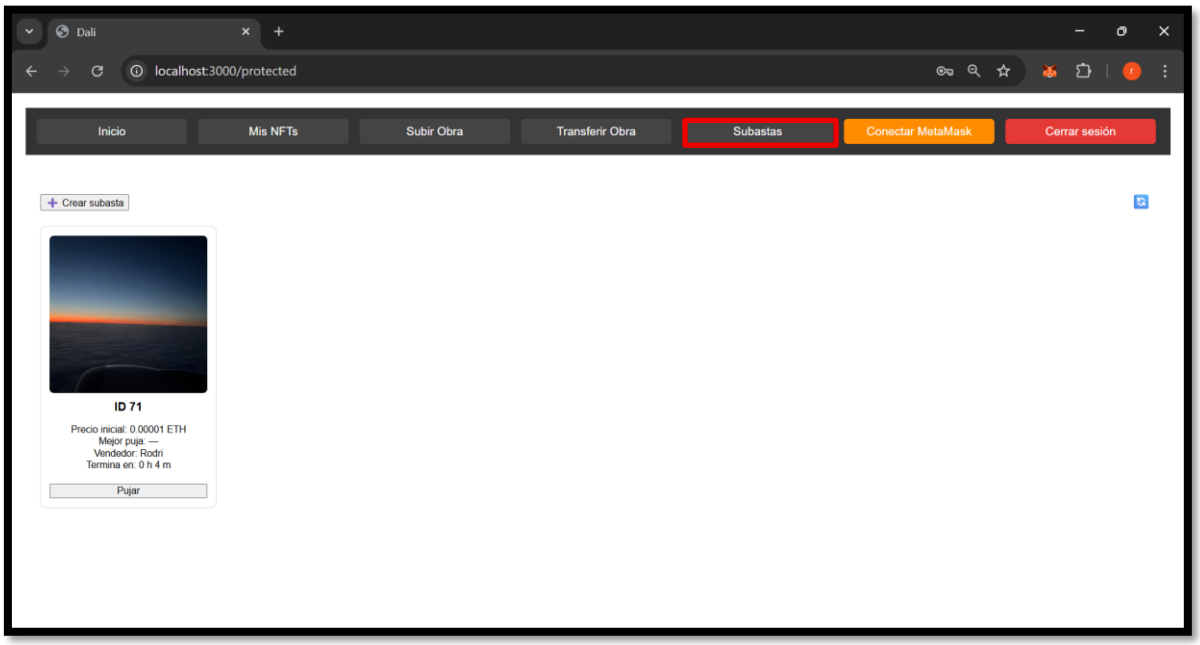

Figura 32. Subasta se puede observar desde el usuario: Rodri

En este caso, el usuario Yago decide participar como postor. En la Figura 33, introduce la cantidad de ETH que desea ofrecer y pulsa el botón de confirmación.

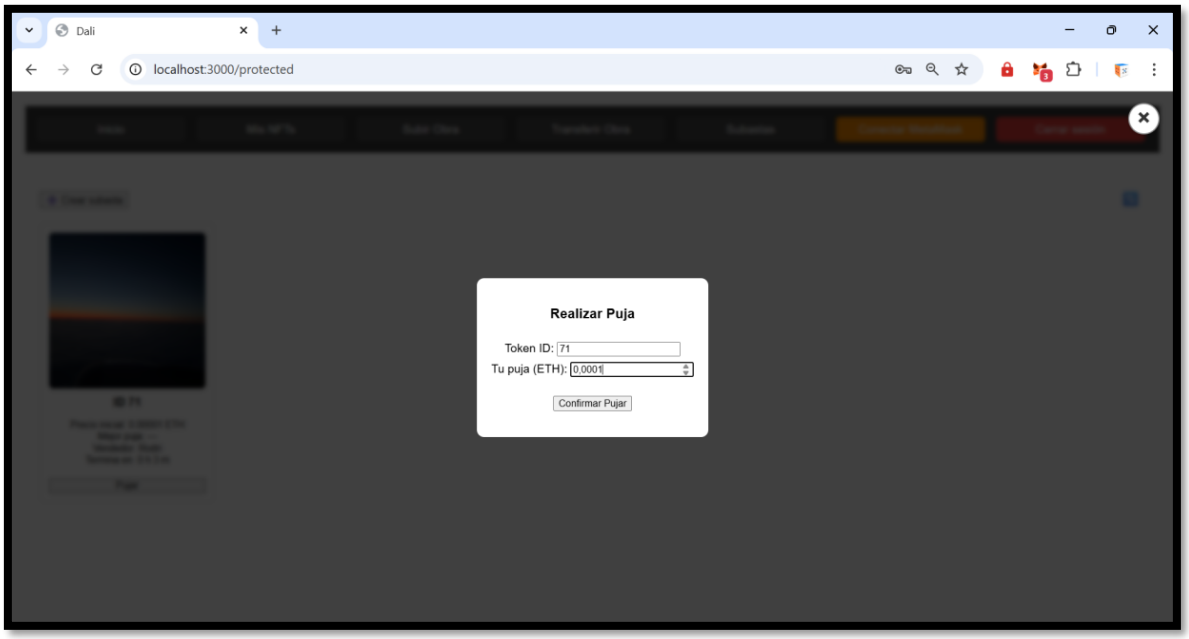

Figura 33. EL usuario: Yago, establece la cantidad de Sepolia a pujar

6.6 VERIFICACIÓN DE OBRAS

La plataforma también permite verificar si una obra ya ha sido registrada. Para ello, el usuario puede subir una imagen y el sistema genera su hash y lo consulta con la blockchain. Si existe un NFT registrado con ese hash, se muestra toda la información relevante, incluyendo propietarios actuales y anteriores, fechas de registro y enlace a IPFS. Como ya se mostró en la Figura 13 y Figura 17.

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

A lo largo del desarrollo de este Trabajo de Fin de Grado se han cumplido todos los objetivos propuestos en el apartado 4.2 Objetivos. El sistema diseñado e implementado permite registrar obras digitales en la blockchain de manera descentralizada, sencilla y segura, combinando tecnologías como Ethereum, IPFS, MetaMask y servidores backend Node.js. A continuación, se detallan las conclusiones alcanzadas y las posibles líneas de mejora.

El **primer objetivo** se centraba en el diseño y despliegue de un contrato inteligente que implementase la lógica necesaria para registrar obras como tokens no fungibles (NFTs), asegurar su autoría mediante hashes y facilitar su trazabilidad. Este objetivo se ha alcanzado con éxito, mediante el desarrollo de un contrato ERC-721 desplegado en la red de pruebas de Ethereum (Sepolia). Además del registro de obras, el contrato permite transferencias de propiedad y sienta las bases para extender el sistema hacia funcionalidades de subasta. Se han utilizado metadatos inmutables, incluyendo enlaces IPFS, para garantizar la autenticidad y permanencia de cada obra registrada.

El **segundo objetivo** consistía en construir una plataforma web que facilitase la interacción con la blockchain a usuarios sin conocimientos técnicos. Este objetivo también se ha cumplido. La arquitectura desarrollada incluye:

- Un backend con Node.js y Express que expone una API REST con endpoints para registrar obras, gestionar usuarios, y acceder a datos de la blockchain y de una base de datos local.
- Una interfaz web conectada a MetaMask, desde la cual los usuarios pueden subir archivos, firmar transacciones y consultar el estado de sus NFTs.
- Integración con IPFS a través de Pinata, permitiendo el almacenamiento descentralizado de los archivos y la generación automática de metadatos JSON asociados.

Todo esto se ha presentado en una interfaz accesible, desarrollada con HTML, CSS y EJS, centrada en la simplicidad de uso.

El **tercer objetivo** se centraba en validar el funcionamiento del sistema. Para ello, se realizaron pruebas continuas durante el desarrollo, incluyendo la subida de múltiples archivos, la firma de transacciones con distintas cuentas de MetaMask, y la visualización de los NFTs en Etherscan. Estas pruebas han demostrado que el sistema es funcional, estable y aplicable a un contexto real con mínimos ajustes.

En resumen, este TFG ha demostrado que es técnicamente posible y viable construir una solución descentralizada para el registro de propiedad intelectual utilizando tecnologías Web3. Además de haber cumplido los objetivos propuestos, el proyecto permite abrir nuevas vías de trabajo y mejora.

Con vistas a evolucionar el sistema hacia un entorno de producción, se proponen los siguientes trabajos futuros:

- Despliegue en la nube (AWS o equivalente): publicar la plataforma en un entorno cloud permitiría habilitar el acceso público, escalar el servicio y mejorar el rendimiento. Además, se podrían configurar sistemas de logs, backups[50] automáticos y monitorización del uso.
- Compatibilidad con otras redes blockchain: aunque Ethereum es la red más consolidada, sus tarifas elevadas limitan la escalabilidad. Sería conveniente estudiar la mitigación a redes como Polygon, Base o Arbiturm, que ofrecen comisiones más bajas y mayor rendimiento.
- Refactorización del sistema de autenticación: actualmente se utiliza JWT con cookies para gestionar las sesiones. Sería recomendable integrar estándares modernos como OAuth 2.0 o soluciones descentralizadas como Web3Auth para mejorar la seguridad y usabilidad.

[50] Copias de seguridad de datos que se crean para poder recuperarlos en caso de pérdida, fallo o corrupción del sistema original.

- Sistema de monetización por comisiones: incluir una lógica de comisión por operación (por ejemplo, un 2% sobre cada registro o transferencia) permitiría cubrir los costes del sistema y ofrecer soporte técnico a los usuarios.
- Educación y documentación para el usuario final: para fomentar el uso de la plataforma, especialmente entre artistas o creadores sin conocimientos técnicos, se podrían integrar tutoriales y materiales interactivos que expliquen el funcionamiento del sistema.

En definitiva, esta plataforma representa una contribución sólida al uso de blockchain para la protección de la propiedad intelectual, demostrando que es posible ofrecer un sistema descentralizado, accesible y seguro que democratice el acceso al registro de obras digitales. Con las mejoras mencionadas, el proyecto podría convertirse en una solución aplicable en entornos reales, contribuyendo al avance de los modelos Web3 en el ámbito cultural y creativo.

Capítulo 8. BIBLIOGRAFÍA

[1] Ministerio de Cultura y Deporte. "Solicitud telemática del registro de la propiedad intelectual". Gobierno de España. 2024. https://www.cultura.gob.es/cultura/areas/propiedadintelectual/mc/rpi/solicitud-telematica.html.

[2] Agencia Estatal Boletín Oficial del Estado. "Texto refundido de la Ley de Propiedad Intelectual (BOE-A-1996-8930)". Boletín Oficial del Estado. 1996. https://www.boe.es/buscar/act.php?id=BOE-A-1996-8930.

[3] Organización Mundial de la Propiedad Intelectual (OMPI). "Convenio de Berna para la protección de las obras literarias y artísticas". OMPI. https://www.wipo.int/treaties/es/ip/berne/.

[4] Comunidad de Madrid. "Información sobre el registro de la propiedad intelectual en la Comunidad de Madrid". Sede Electrónica. https://sede.comunidad.madrid/node/220100.

[5] Nakamoto, Satoshi. "Bitcoin: A Peer-to-Peer Electronic Cash System". 2008. https://bitcoin.org/bitcoin.pdf.

[6] Buterin, Vitalik. "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform". 2014. https://ethereum.org/en/whitepaper/.

[7] Profile. "¿Qué es Node.js y para qué sirve?". Profile Blog. 2024. https://profile.es/blog/que-es-nodejs/.

[8] Mozilla Developer Network. "Introducción a Express/Node". MDN Web Docs. 2024. https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction.

[9] Carceller, Víctor. "Introducción a IPFS — InterPlanetary File System". Víctor Carceller Blog. 2021. https://elpuig.xeill.net/Members/vcarceler/articulos/introduccion-a-ipfs.

[10] NFT Now. "How to Set Up a MetaMask Wallet". NFT Now, 2023. https://nftnow.com/guides/how-to-set-up-metamask-wallet/.

[11] VPN Unlimited. "Qué es Bcrypt". VPN Unlimited Blog. 2024. https://www.vpnunlimited.com/es/help/cybersecurity/bcrypt.

[12] OpenWebinars. "Qué es Json Web Token y cómo funciona". OpenWebinars Blog. 2024. https://openwebinars.net/blog/que-es-json-web-token-y-como-funciona/.

[13] EBAC. "Qué es GitHub y para qué sirve: una guía para principiantes". EBAC Blog. 2024. https://ebac.mx/blog/que-es-github.

[14] Coinbase. "¿Qué es Etherscan y cómo utilizarlo?". Coinbase Learn. 2024. https://www.coinbase.com/es-es/learn/crypto-glossary/what-is-etherscan-and-how-to-use-it.

[15] DataScientest. "Solidity: todo lo que necesitas saber". DataScientest, 2023. Available at: https://datascientest.com/es/solidity-todo-lo-que-necesitas-saber.

[16] Remix Project. "Welcome to Remix's documentation!". Remix - Ethereum IDE Documentation, 2025. Available at: https://remix-ide.readthedocs.io/en/latest/index.html.

[17] Trent, T., & Dapp, M. (2015). Towards an Ownership Layer for the Internet [Whitepaper]. Ascribe GmbH. Recuperado de https://bravenewcoin.com/assets/Whitepapers/ascribe-whitepaper-20150624.pdf

[18] Po.et Foundation. (2017). Po.et: A shared, open, universal ledger designed to record metadata and ownership information for digital creative assets [Whitepaper]. Recuperado de https://uploads-ssl.webflow.com/5a0c978e0d22aa0001464356/5a7796662b07370001ace7a1_whitepaper.pdf

[19] Ng, W. L. (2022). A new era of the visual art market? A platform analysis of the biggest NFT market OpenSea on diversity and equality [Tesis de maestría, Lund University]. DiVA Portal. Recuperado de https://www.diva-portal.org/smash/get/diva2:1692740/FULLTEXT02.pdf

[20] Gervais, A., Graef, I., & Husovec, M. (2023). Rights in NFTs and the Flourishing of NFT Marketplaces. International Journal of Law and Information Technology. Recuperado de https://academic.oup.com/ijlit/article/doi/10.1093/ijlit/eaae018/7746479

[21] Qin, Y., & Fu, Z. (2021). Non-Fungible Token (NFT): Overview, Evaluation, Opportunities and Challenges. Recuperado de https://www.researchgate.net/publication/351656444_Non-Fungible_Token_NFT_Overview_Evaluation_Opportunities_and_Challenges

[22] Fazli, M., Giannotti, F., & Pedreschi, D. (2021). Under the Skin of Foundation NFT Auctions. arXiv. Recuperado de https://arxiv.org/abs/2109.12321

[23] Cong, Y., Lin, H., & Li, Z. (2024). A novel copyright protection scheme for digital images based on blockchain and IPFS. Journal of Network and Computer Applications. Recuperado de https://www.sciencedirect.com/science/article/pii/S1546221824006878

[24] Zhang, W., Wei, Y., & Zhao, K. (2024). Blockchain and Smart Contracts for Digital Copyright Protection. Future Internet, 16(5), 169. Recuperado de https://www.mdpi.com/1999-5903/16/5/169

[25] Li, Y., Chen, T., & Xie, M. (2021). Digital copyright protection based on blockchain technology. Journal of Healthcare Engineering, 2021, Article ID 5575546. Recuperado de https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8459789/

[26] Coinbase. (s. f.). ¿Qué es ERC-721? Recuperado de https://www.coinbase.com/es-us/learn/crypto-glossary/what-is-erc-721?

[27] Ministerio de Economía y Hacienda. (2010). Orden EHA/733/2010, de 25 de marzo, por la que se aprueban los aspectos contables de las empresas públicas que operan en determinados sectores y se establecen criterios de amortización. Boletín Oficial del Estado, núm. 79, de 1 de abril de 2010. https://www.boe.es/diario_boe/txt.php?id=BOE-A-2010-5302

[28] Art Basel & UBS. The Art Market 2024. 2024. https://www.artbasel.com/about/initiatives/the-art-market

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

Este trabajo de Fin de Grado se alinea con varios Objetivos de Desarrollo Sostenible propuestos por la Agenda 2030 de las Naciones Unidas, al contribuir a una digitalización más segura, accesible y transparente de la propiedad intelectual. Algunos de estos objetivos se muestran en la siguiente Figura 34:

Figura 34. Objetivos de desarrollo sostenible de las Naciones Unidas (Fuente: UN)

En concreto, el proyecto impacta positivamente en los siguientes ODS:

- ODS 9 – Industria, Innovación e Infraestructura: al desarrollar una plataforma basada en tecnologías emergentes como blockchain, IPFS y WEB3, se impulsa la

innovación tecnológica y se promueve una infraestructura digital resiliente y descentralizada.

- ODS 16 – Paz, Justicia e Instituciones Sólidas: el sistema propuesto fomenta la transparencia y la trazabilidad en el registro de obras digitales, permitiendo verificar de forma pública y segura la autoría de los contenidos, lo que refuerza los mecanismos de justicia y protección de derechos.
- ODS 4 – Educación de Calidad (indirectamente): al democratizar el acceso a herramientas de registro digital sin necesidad de conocimientos técnicos avanzados, se promueve el acceso inclusivo al uso de tecnologías disruptivas y se contribuye a la alfabetización digital.

De este modo, el proyecto no solo ofrece una solución técnica funcional, sino que también respalda valores de equidad, acceso abierto y fortalecimiento de la confianza digital en entornos creativos y culturales.

ANEXO II: MANUAL DE INSTALACIÓN

En este apartado se describen los requisitos y preparativos necesarios para que cualquier usuario pueda utilizar la plataforma y registrar imágenes como NFTs en la blockchain de forma autónoma.

Para ello será necesario contar con los siguientes elementos:

- Disponer de una cuenta activa en MetaMask.
- Obtener el código fuente de la aplicación (en formato comprimido .zip o suministrado por el desarrollador).
- Descargar e instalar Node.js, junto con las dependencias necesarias para ejecutar el servidor.

Nota: Se recomienda utilizar el buscador de Google Chorme para garantizar una correcta visualización e integración con la extensión de MetaMask. Puede descargarse desde el siguiente enlace: (*pulsar aquí*)

A.II.1 MetaMask

Para comenzar a utilizar la plataforma, es necesario disponer de una wallet compatible con Ethereum. En este caso, se utilizará MetaMask.
A continuación, se detalla el procedimiento para crear una cuenta en MetaMask y configurar la red de pruebas de Sepolia:

1. Acceder al sitio oficial de MetaMask (*pulsar aquí*) y hacer click en el botón “GET MetaMask”.
2. Seleccionar la opción “Agregar al navegador” y, posteriormente, hacer click en “Añadir extensión”
3. Aceptar los términos y condiciones de uso.
4. Crear una contraseña para acceder a la cuenta.

5. (Opcional) Omitir la configuración de proteger monedero y la frase de recuperación si se desea continuar más tarde.
6. Después de esto, se accede al panel central de MetaMask. En la parte superior izquierda se mostrará la red activa. Si está configurada en "Ethereum Mainet", hacer click y seleccionar la opción "Mostrar redes de prueba" como se muestra en la Figura 35 y seleccionar Sepolia.
7. Para obtener fondos de prueba, se accede a un faucet o grifo de Sepolia (pulse aquí).
8. Iniciar sesión dentro de faucet con Google (A veces se inicia sesión automáticamente).
9. Introducir la dirección pública de la wallet (visible en el menú principal bajo el nombre de "account" Figura 36).
10. Solicitar la transferencia de 0.05 ETH Sepolia pulsando el botón "Receive 0.05 Sepolia ETH"
11. Una vez completado el proceso, se motrará el mensaje "Drip Complete" y podrá verificarse que el saldo, ya ha sido acreditado en la cuenta

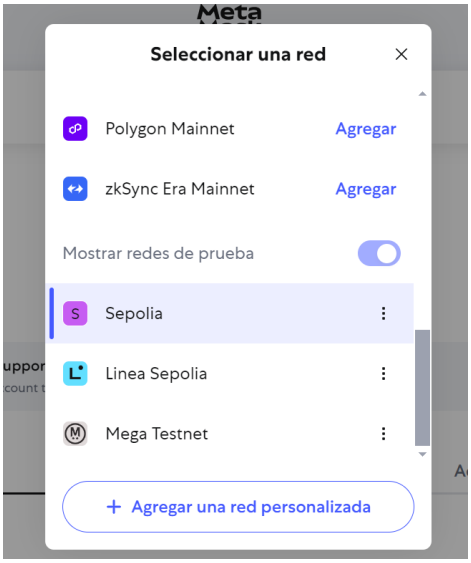

Figura 35. Seleccionar Red Sepolia

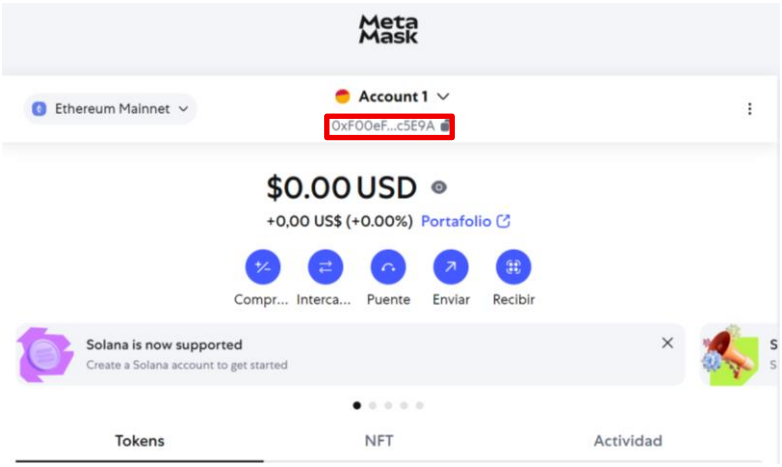

Figura 36. Id pública de la wallet

Con estos pasos, la cuenta de MetaMask estará lista para interactuar con la plataforma de registro de obras digitales.

A.II.2 Código

El código fuente de la aplicación se puede obtener en formato comprimido *.zip*, el cual contiene todos los archivos necesarios para su despliegue. Este archivo debe ser descomprimido localmente en el equipo del usuario.

1. Para descargarlo (pulsar aquí).

A.II.3 Despliegue del servidor

Para desplegar el servidor y ejecutar la aplicación de forma local, se deben de seguir los pasos que se detallan a continuación:

1. Descargar e instalar Node.js desde su página oficial (pulsar aquí)
2. Durante la instalación, en la ventana “Custom Setup”, asegurarse de marca la opción Add to PATH”, tal como se muestra en la Figura 37.
3. Una vez finalizada la instalación, abrir una ventana de comandos (cmd o PowerShell), se puede hacer directamente escribiendo cmd en el buscador de aplicaciones, después verificar la correcta instalación escribiendo el siguiente comando:

```
npm -v
```

El programa debería devolver la versión instalada de Node.js

4. A continuación, acceder a la carpeta donde se encuentra el código descomprimido. Por ejmplo:

```
cd C:\Users\...\Código
```

Dentro de esa carpeta, instalar las dependencias necesarias ejecutando:

```
npm install express ethers ipfs-http-client dotenv cors
```

5. Para lanzar el servidor, utilizar el siguiente comando:

```
npm start
```

Debería recibir como respuesta: *Servidor corriendo en http://localhost:3000*.

6. Finalmente, abrir el navegador e introducir la dirección *http://localhost:3000* (la respuesta que se ha mencionado antes). Debería aparecer la pantalla de inicio de sesión y registro de la plataforma.

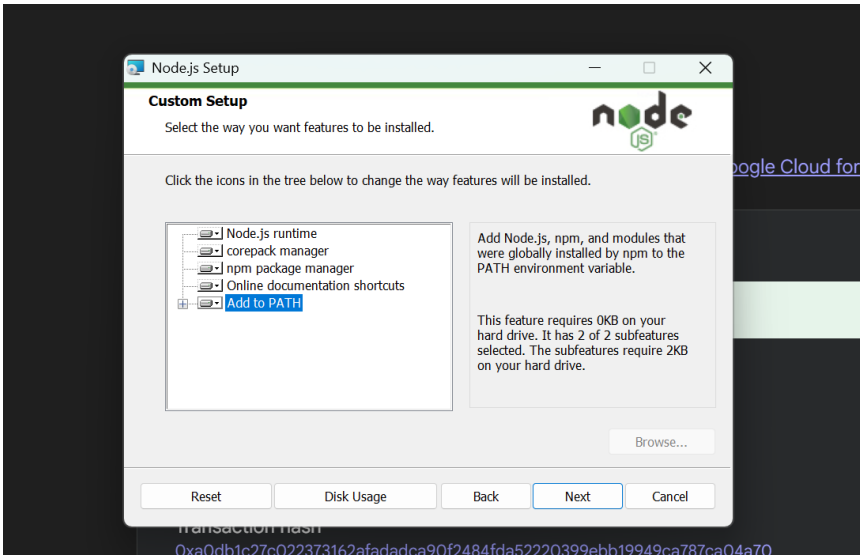

Figura 37. Seleccionar "Add to PATH"

ANEXO III: MANUAL DE USUARIO

A.III.1 Página de registro

Cuando el usuario accede por primera vez a la plataforma, se le presenta una pantalla de autenticación dividida en dos formularios principales: inicio de sesión y registro de cuenta nueva (Figura 38):

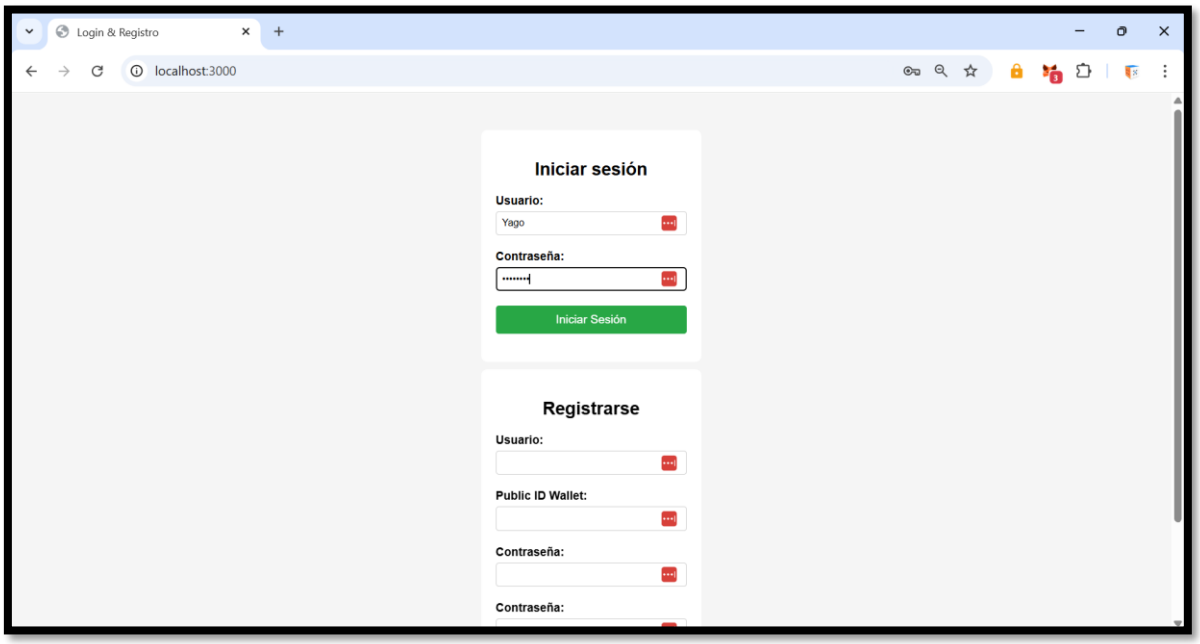

Figura 38. Pantalla de inicio de sesión y registro de usuario en el sistema

El proceso de registro consiste en:

1. Nombre de usuario (mínimo 4 caracteres).
2. Dirección pública de wallet Ethereum.
3. Contraseña (mínimo 6 caracteres).
4. Confirmación de la contraseña.

Para autenticarse, el usuario debe ingresar:

1. Su nombre de usuario.
2. Su contraseña.

A.III.2 Pantalla de inicio.

Una vez que el usuario ha iniciado sesión correctamente, es dirigido a la interfaz principal.

En la parte superior de la vista aparece una barra de navegación con varios botones que permiten cambiar entre las distintas secciones de la plataforma:

- Inicio.
- Subir Obra.
- Transferir Obra.
- Pujar.
- Cerrar Sesión.

Al acceder por defecto, se muestra activada la sección de inicio, la cual presenta un mensaje introductorio y permite al usuario familiarizarse con la estructura general del sistema (Figura 39). Además, aparece una ventana emergente solicitando la contraseña de la wallet de MetaMask con el fin de desbloquearla y permitir el acceso al contenido. En caso de que esta ventana sea cerrada accidentalmente, el usuario puede volver a acceder manualmente a través de la extensión de MetaMask en el navegador.

Para comprobar que la conexión con MetaMask se ha establecido correctamente, puede pulsarse el botón "Conectar a MetaMask", el cual verifica y notifica si la conexión es exitosa.

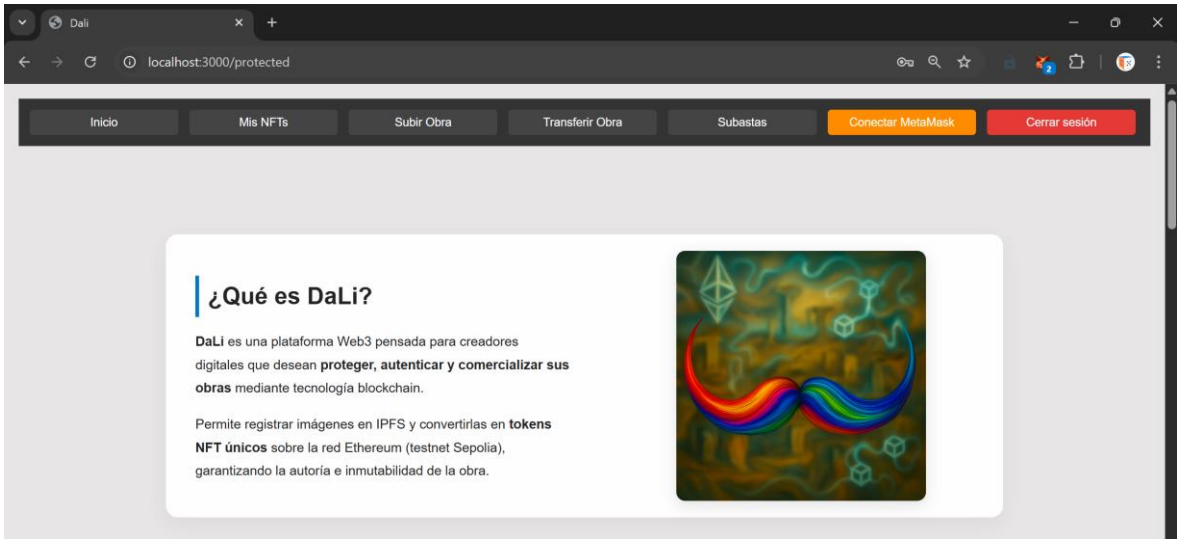

Figura 39. Pantalla de inicio

El usuario puede hacer scroll vertical para visualizar el contenido informativo de la plataforma.

Figura 40. ¿Cómo funciona?

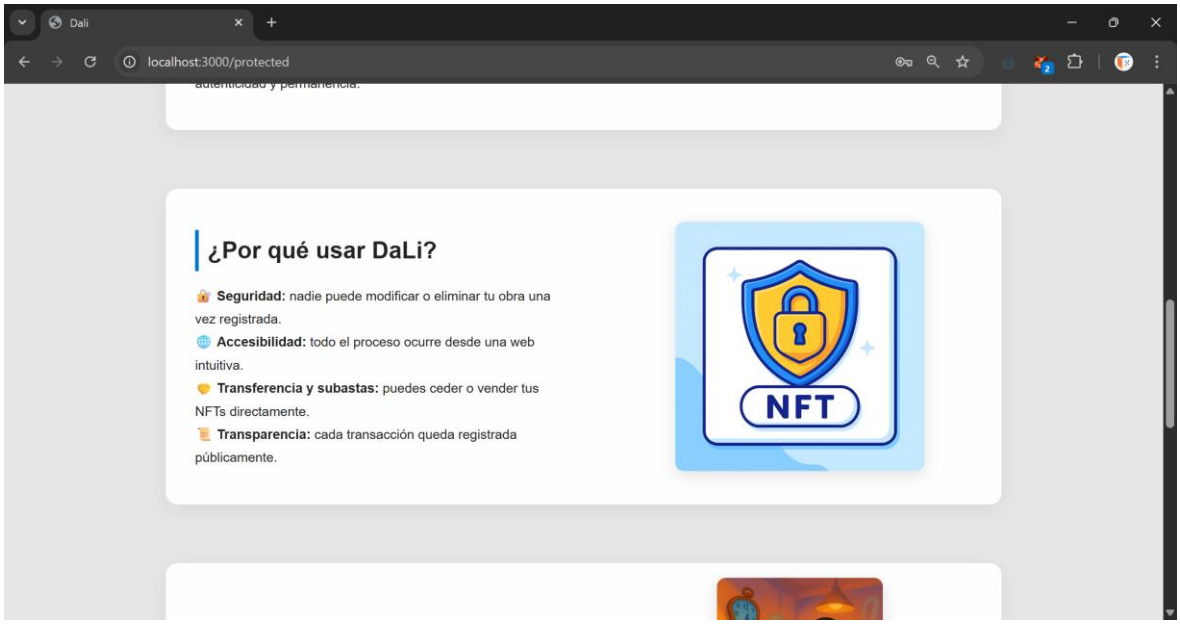

Figura 41. ¿Por qué usar DaLi?

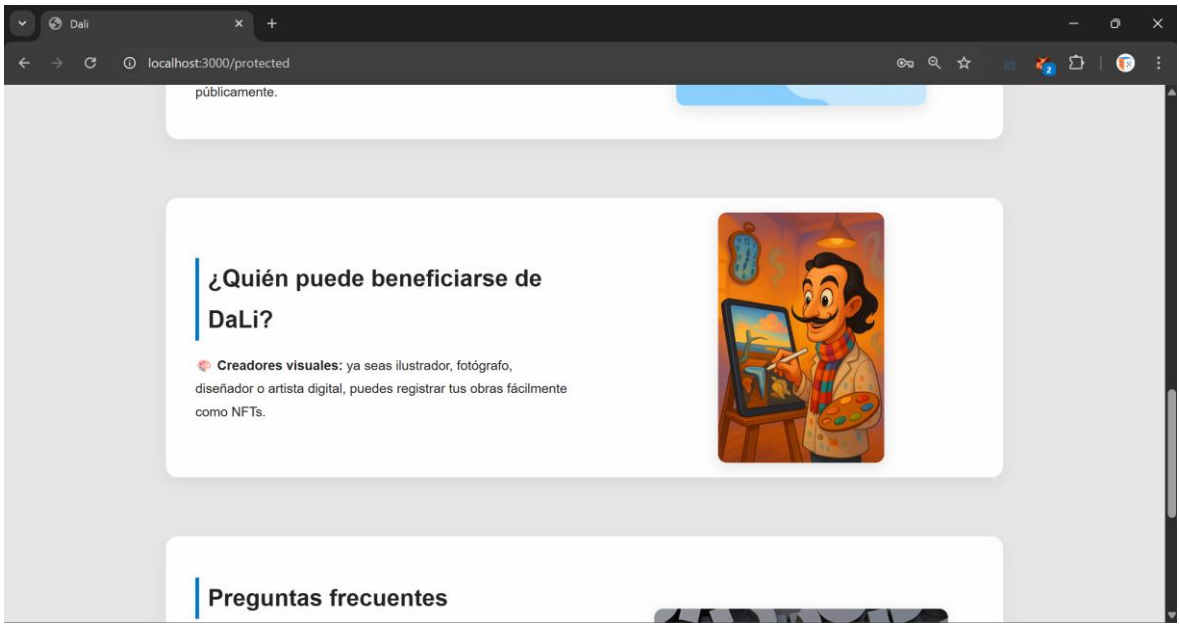

Figura 42. ¿Quién puede beneficiarse de DaLi?

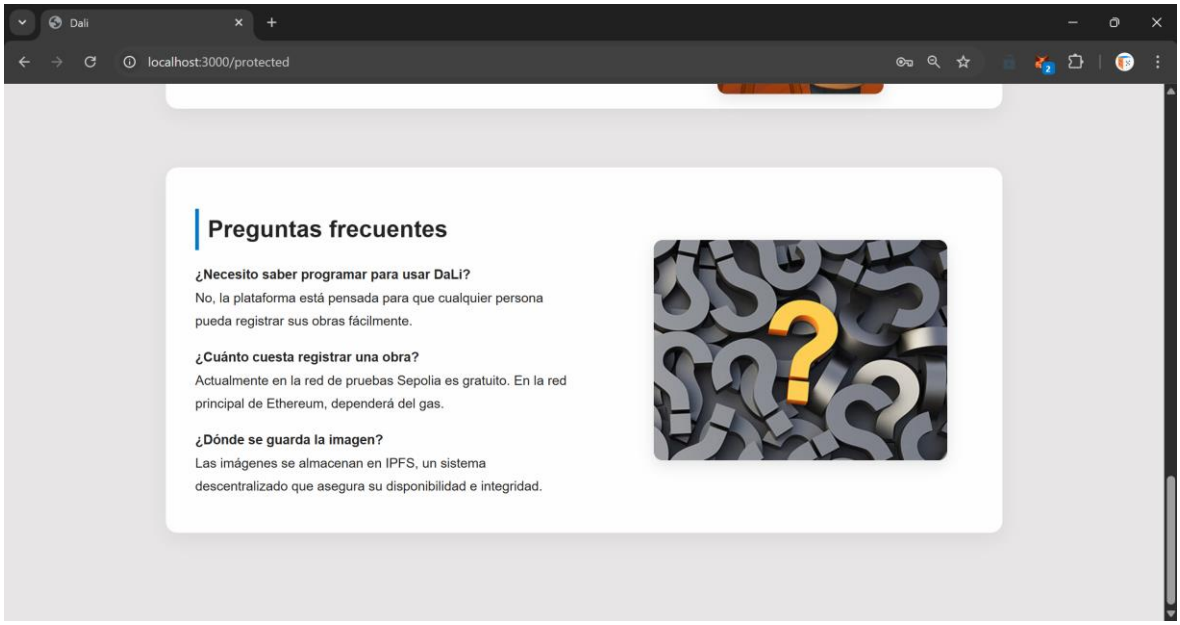

Figura 43. Preguntas frecuentes.

A.III.3 Visualización de NFTs propios.

Tras iniciar sesión y conectar su wallet, el usuario puede acceder a la sección “Mis NFTs”, donde se muestran las obras digitales que ha registrado como tokens en la blockchain. Esta vista permite verificar la autoría y consultar sus obras de forma visual e intuitiva:

1. Acceder a “Mis NFTs” desde el menú principal.
2. Visualizar la galería de obras registradas en formato tarjeta como se muestra en la Figura 44, si la cuenta acaba de ser creada no debe aparecer ninguna obra, para ello se debe subir una obra antes.
3. Recargar la galería con el botón ubicado arriba a la derecha.
4. Ampliar cualquier imagen haciendo click sobre ella (Figura 45)

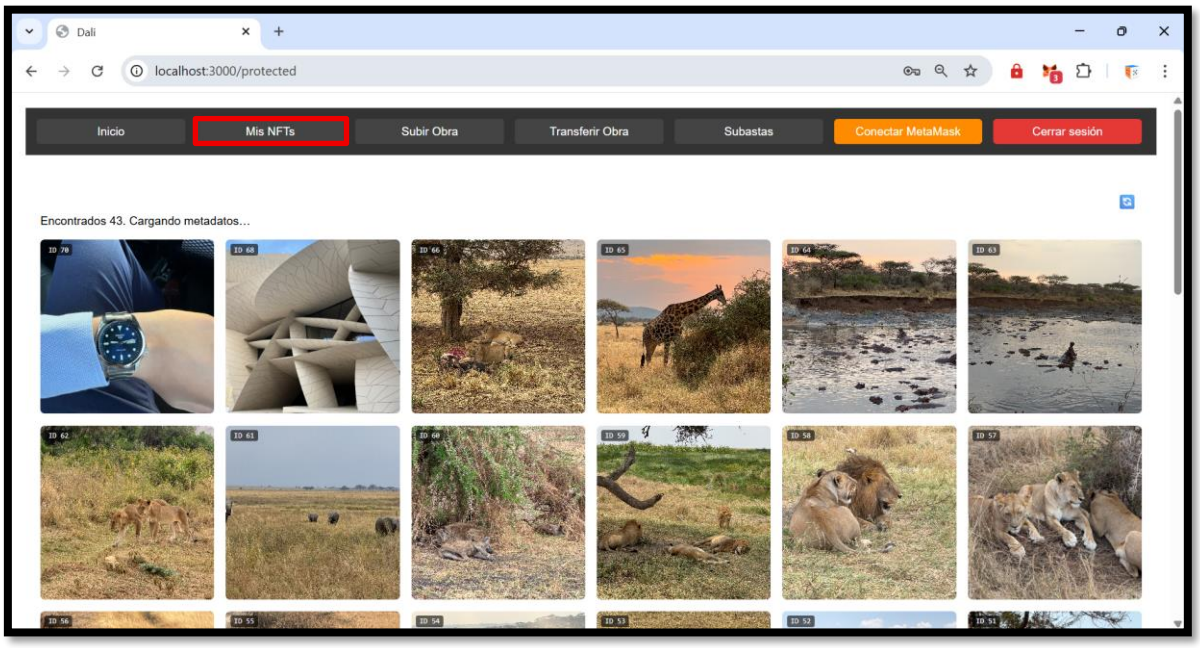

Figura 44. Visualización de NFTs registrados por el usuario.

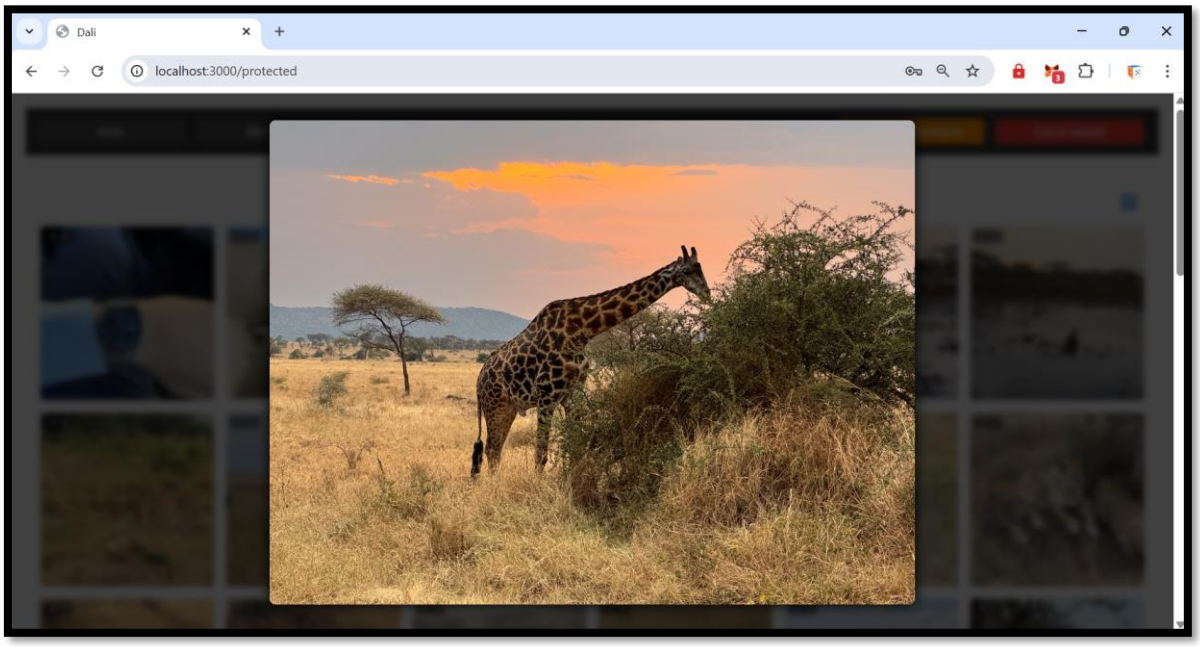

Figura 45. Seleccionar una imagen para verla ampliada

A.III.4 Subida de una obra como NFT.

A continuación, se detallan los pasos para subir una obra:

1. Acceder a la sección “Subir Obra” desde el menú principal tras iniciar sesión (Figura 46).
2. Seleccionar una imagen desde el dispositivo. La plataforma muestra una previsualización automática (Figura 47).
3. Pulsar el botón “Registrar Obra”, que lanza el proceso de registro.
4. El sistema realiza de forma automática:
    a. La subida del archivo a IPFS usando el servicio Pinata.
    b. La generación de metadatos en formato JSON.
    c. La preparación de la transacción en Ethereum, estimando el gas necesario.
5. Confirmar la operación en MetaMask, donde se muestra el contrato, el coste estimado y se solicita la firma (Figura 48).
6. Una vez aprobada, se ejecuta la función *registrarObra(tokenURI)* y se muestra un resumen con (Figura 49):
    a. Imagen registrada.
    b. ID del token.
    c. Enlace a IPFS.
    d. Enlace a la transacción de Etherscan.
7. Comprobar la nueva obra accediendo a “Mis NFTs”, donde aparecerá en la galería junto a las versiones anteriores (Figura 50).

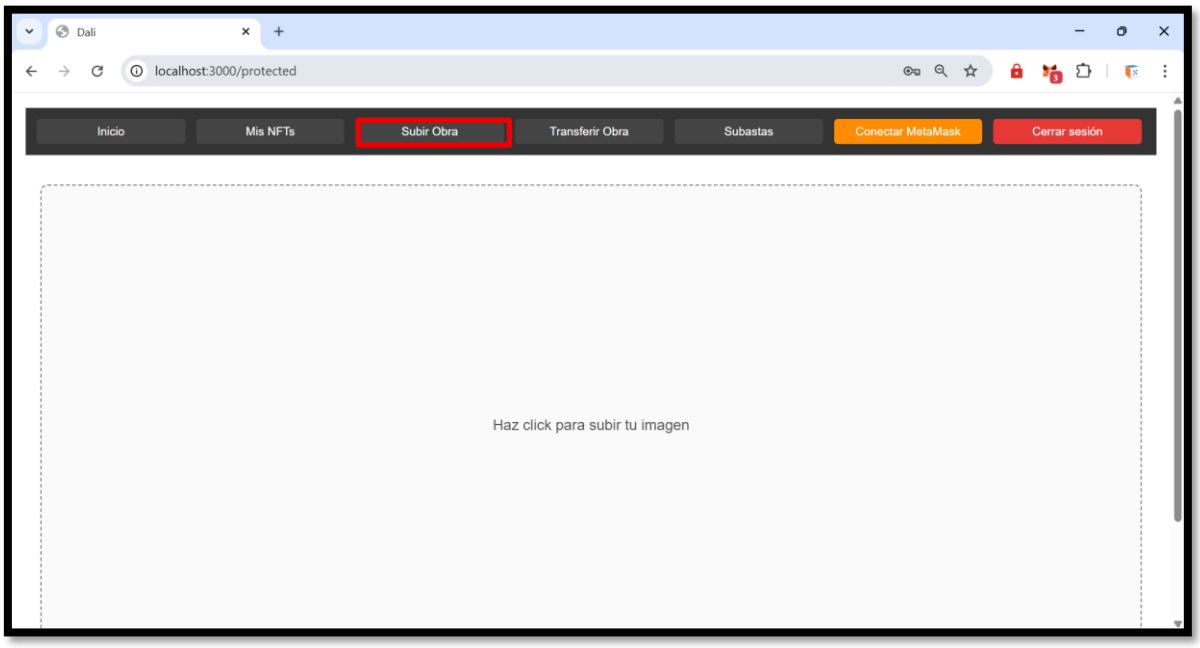

Figura 46. Pantalla de Subir Obra

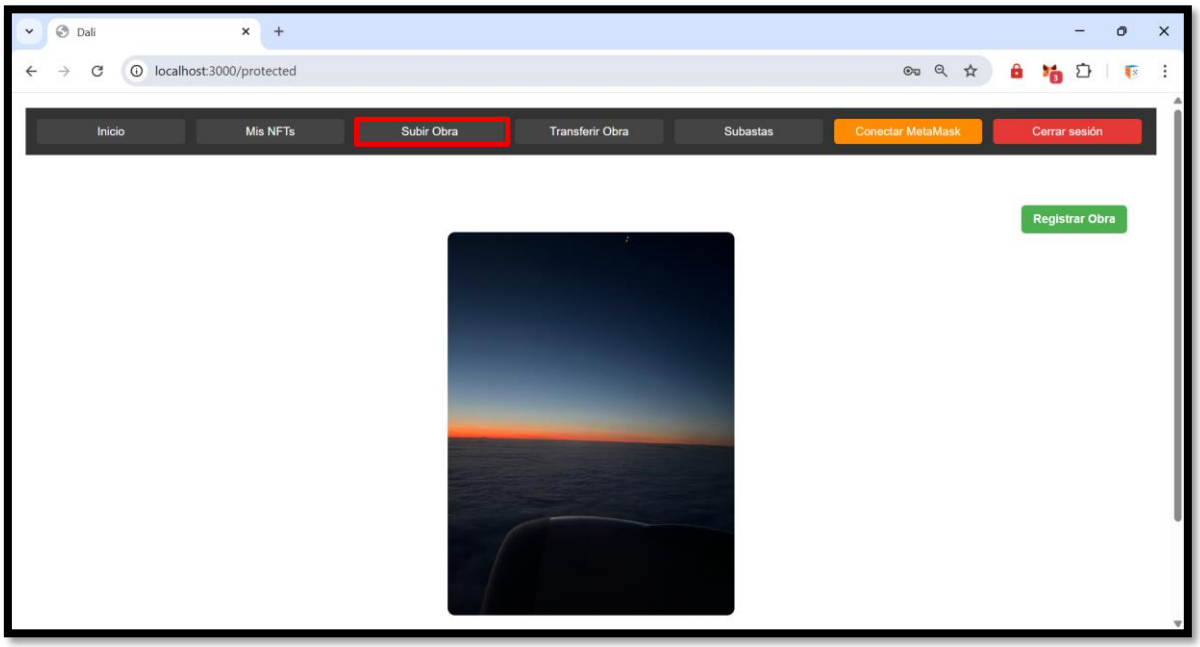

Figura 47. Obra Seleccionada

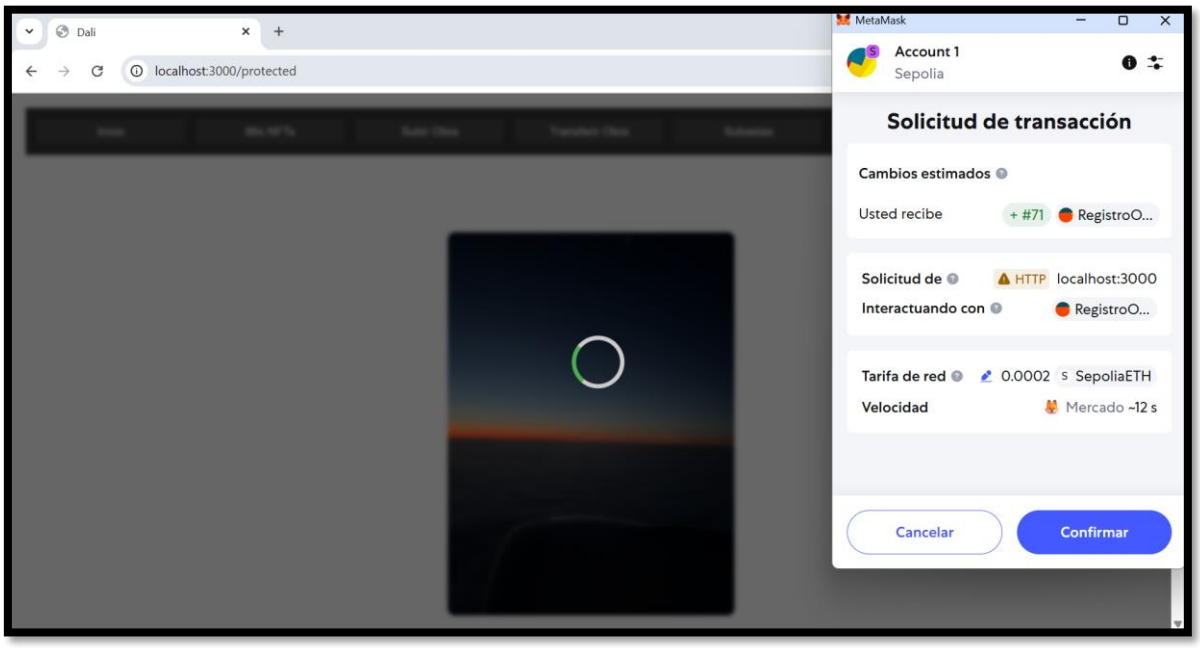

Figura 48. Confirmar mediante MetaMask la subida de la obra

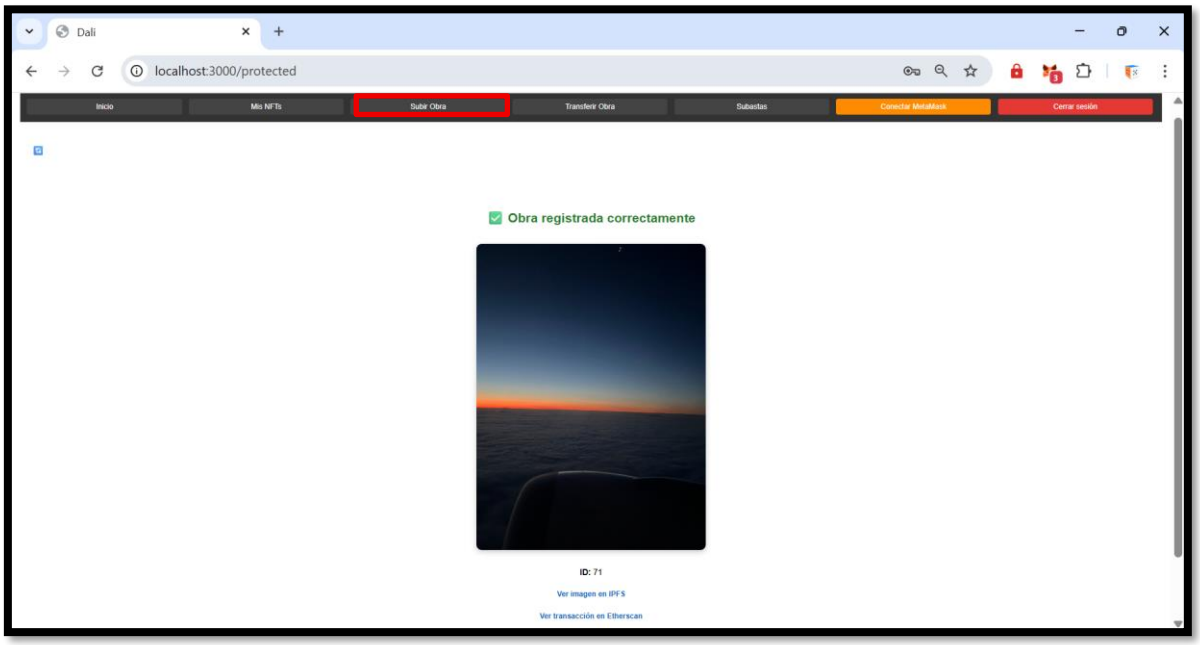

Figura 49. Respuesta: obra registrada correctamente

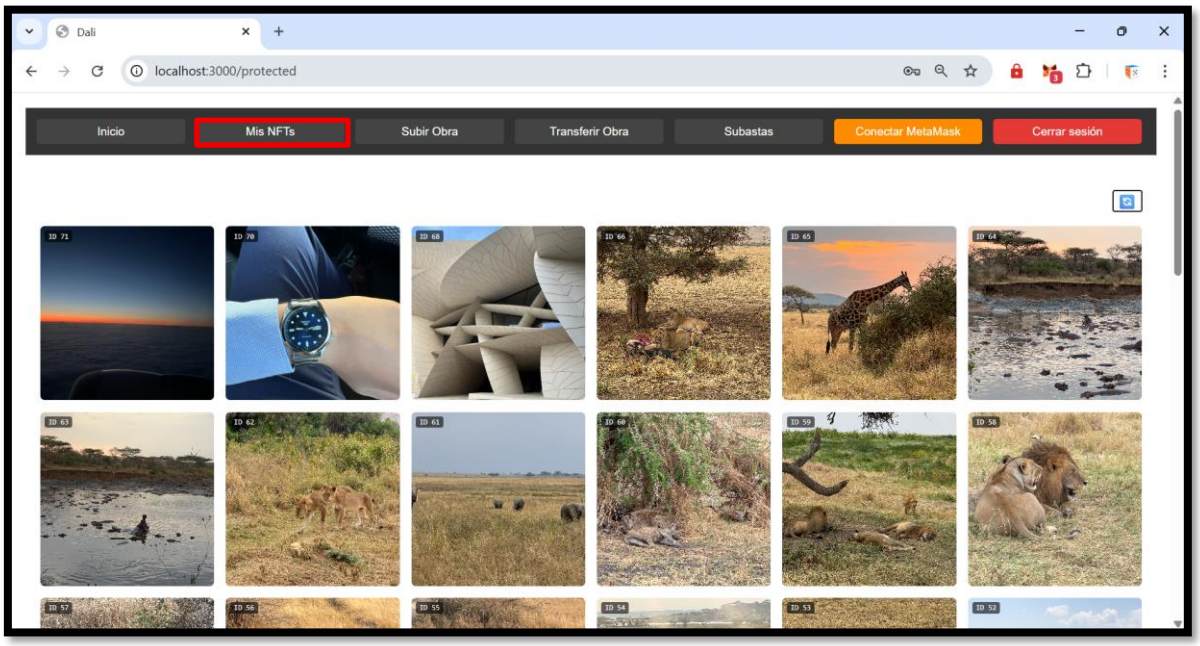

Figura 50. Nueva obra reflejada en la galería de NFTs del usuario: Yago

A.III.5 Transferencia de obras.

La plataforma permite a los usuarios transferir obras registradas a otros miembros y funciona de la siguiente manera:

1. Acceder a la sección “Transferir Obra” desde el menú principal tras haber iniciado sesión.
2. Introducir el nombre de usuario del destinatario (Figura 51).
3. El sistema valida el nombre y recupera automáticamente la dirección de wallet asociada si el usuario existe.
4. Una vez validado, se muestra una galería con todas las obras que el usuario actual tiene registradas (Figura 52).
5. Seleccionar la obra que se desea transferir.
6. Hacer scroll hacia abajo y pulsar el botón de “Transferir”, que inicia el proceso de firma (Figura 53).
7. Confirmar la transacción en MetaMask (Figura 54), donde se muestran:
    a. El *tokenId*.

b. La wallet del remitenete y la del destinatario.
c. La red (Sepolia)

8. Tras la firma, la plataforma espera la confirmación (*tx.wait()*) y muestra un mensaje de éxito al completar la operación (Figura 55).
9. Accediendo nuevamente a la sección "Mis NFTS" (usuario: Yago), el usuario puede comprobar que la obra ya no está en su galería (Figura 56).
10. Si se accede con la cuenta del destinatario, la obra aparece automáticamente en su galería personal, confirmando que el cambio de propiedad fue exitoso (Figura 57).

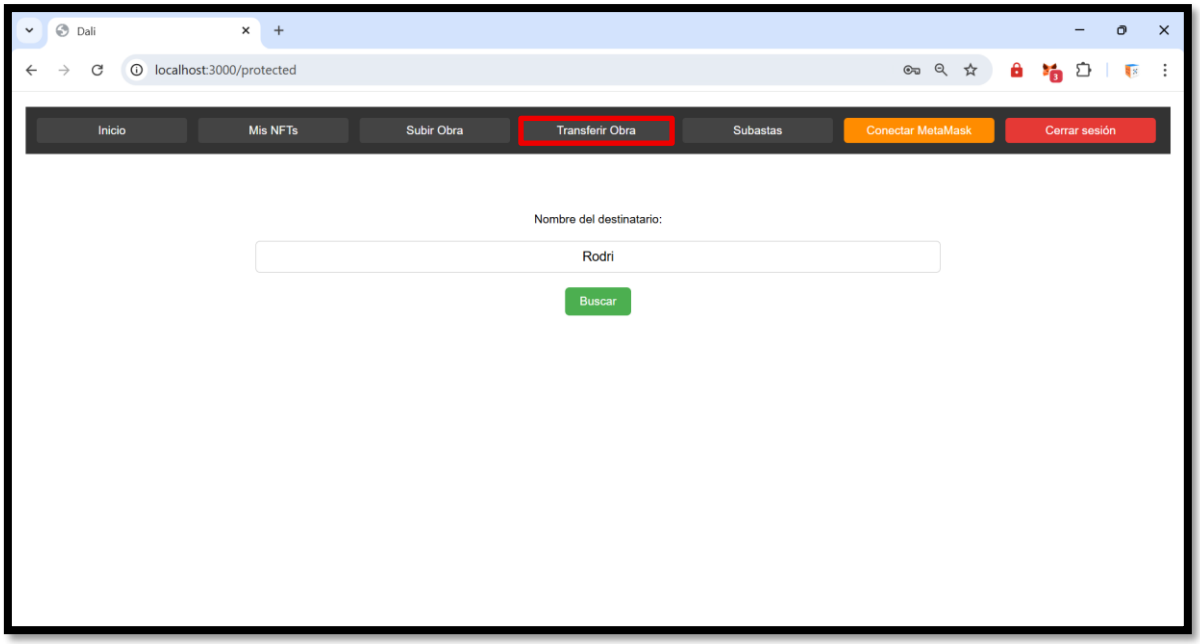

Figura 51. Escribir el nombre del destinatario

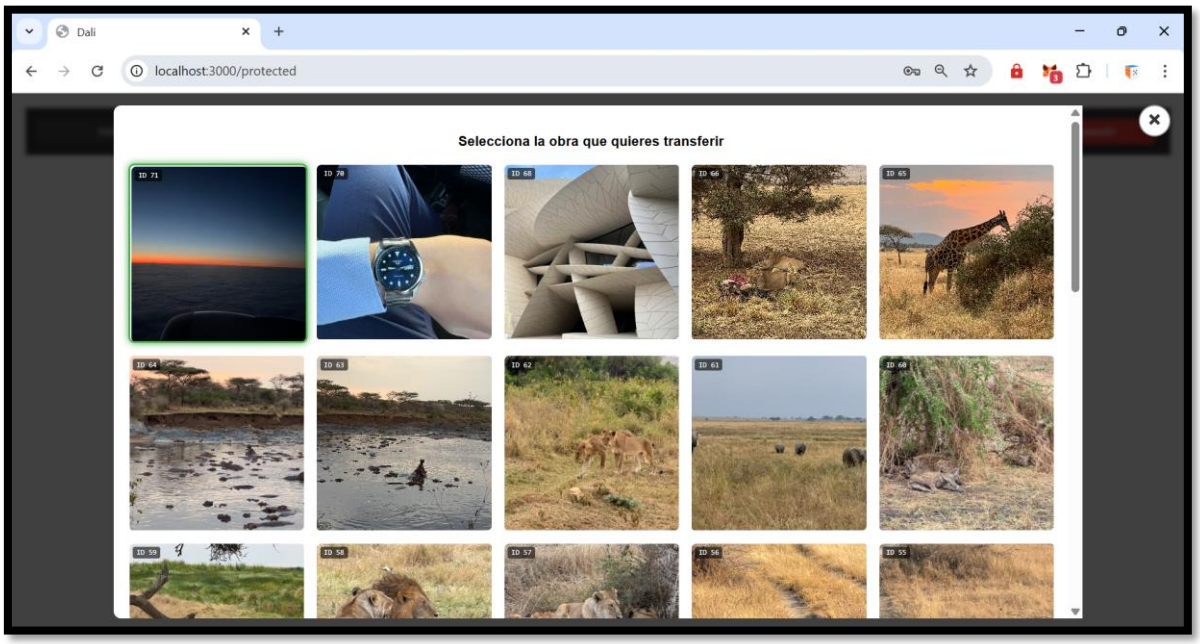

Figura 52. Seleccionar la obra a transferir

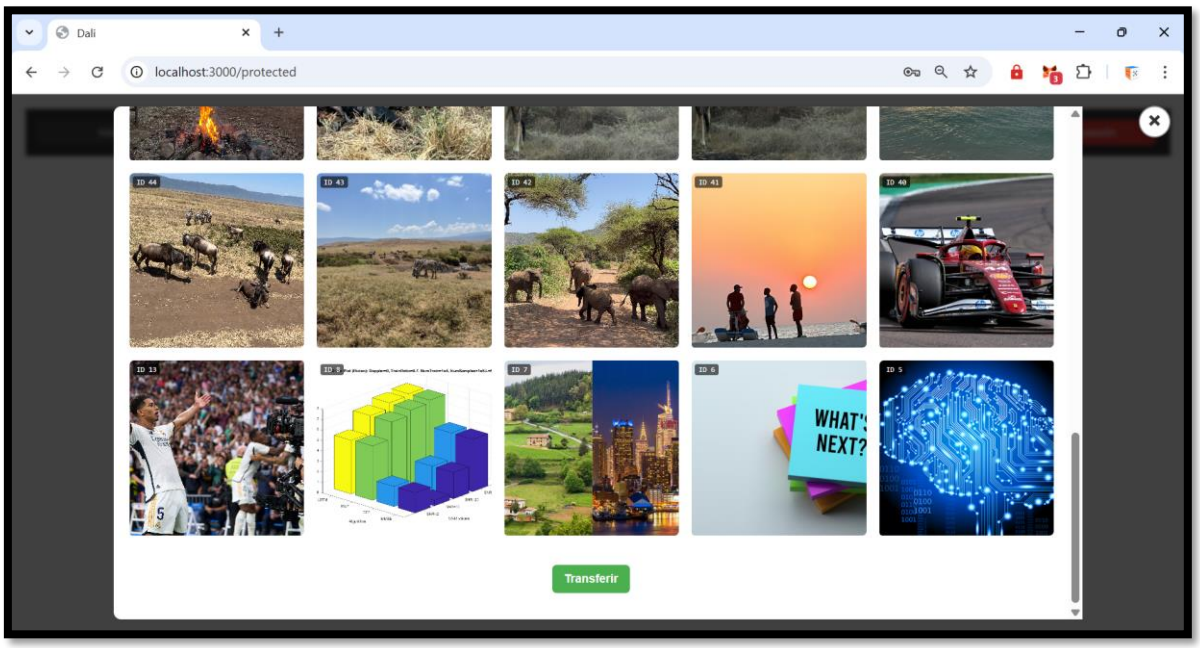

Figura 53. Botón de transferir

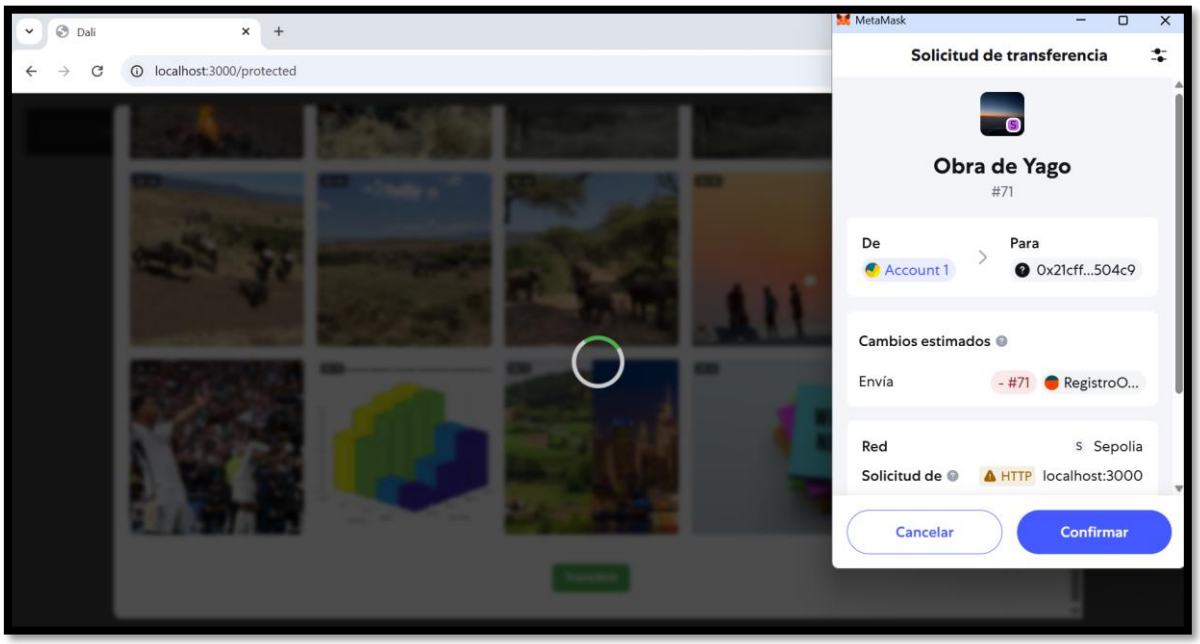

Figura 54. Confirmar la transferencia en MetaMask

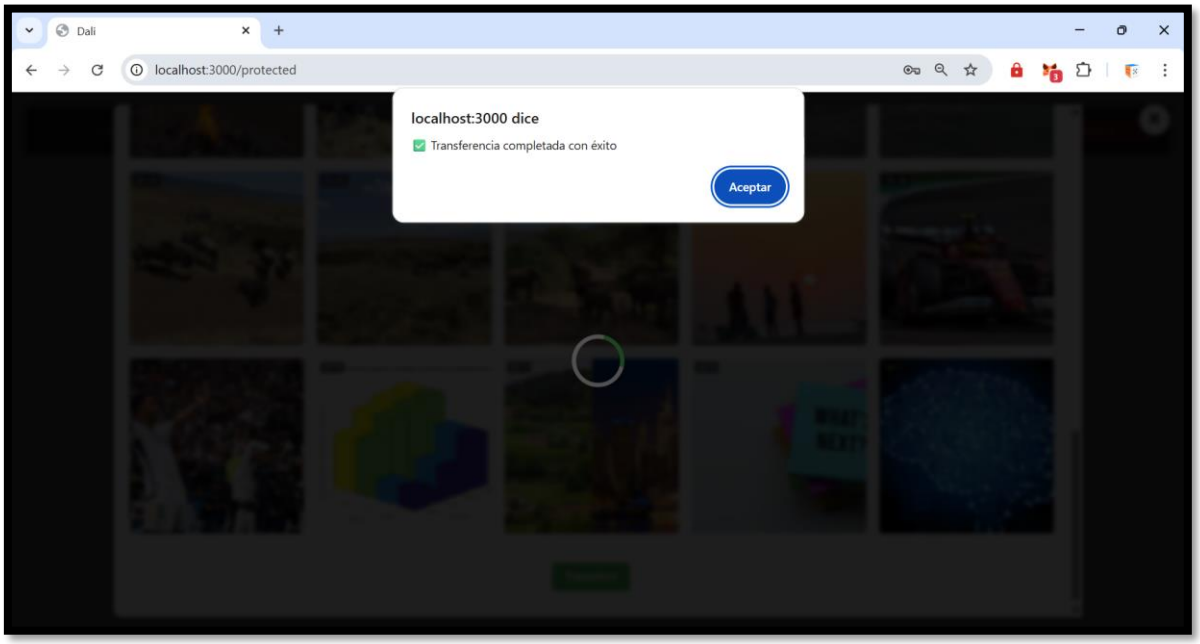

Figura 55. Respuesta: Obra transferida correctamente

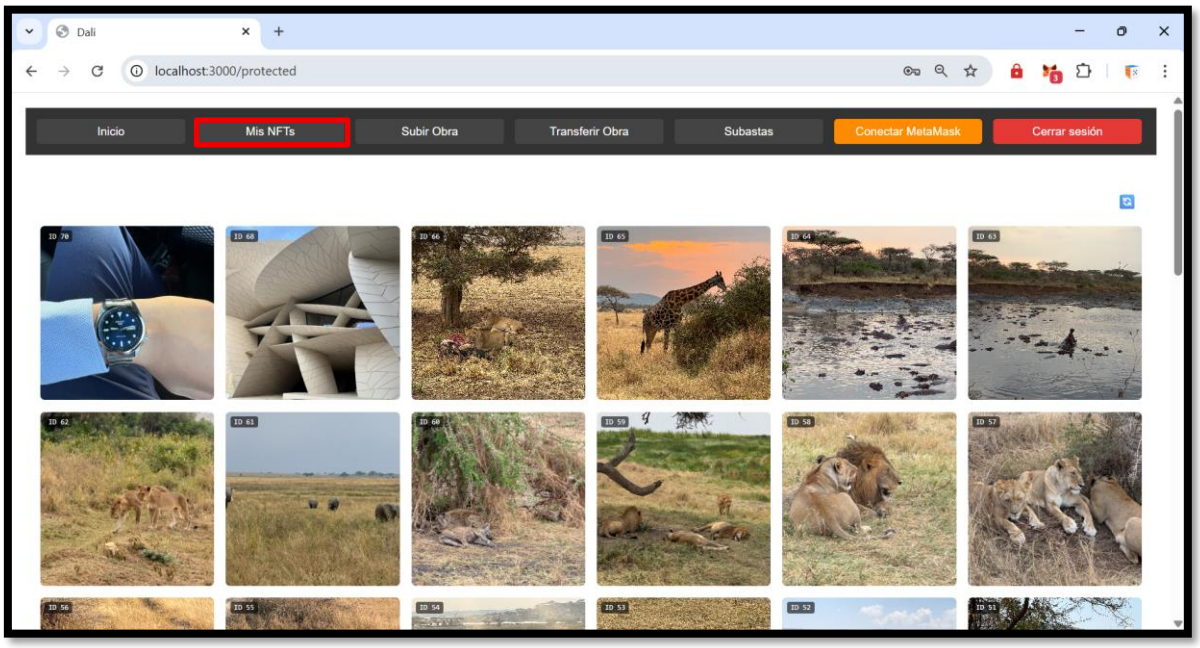

Figura 56. No está la obra en el perfil de Yago

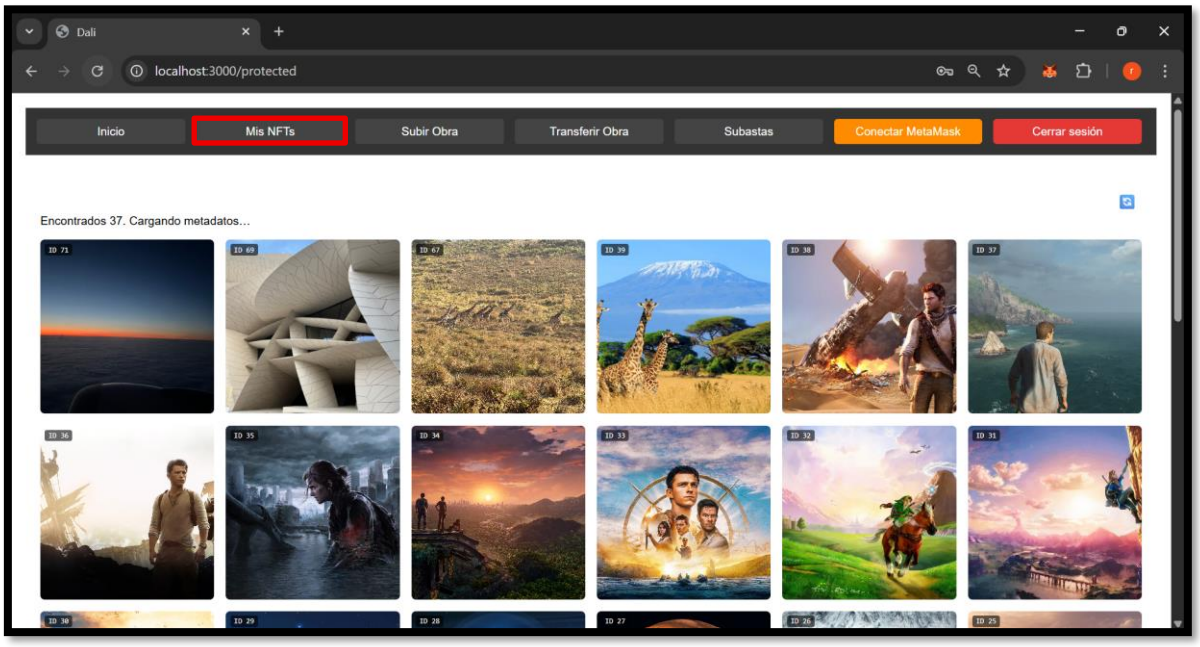

Figura 57. Obra en el perfil de Rodri

A.III.6 Subastas: Creación y Pujas.

La plataforma permite a los usuarios crear subastas descentralizadas para sus NTFs. Acciones para crear una subasta:

1. Acceder a la sección “Subastas” (usuario: Rodri) desde el menú principal tras iniciar sesión.
2. Pulsar el botón “Crear Subasta” para iniciar el proceso (Figura 58).
3. El sistema muestra una galería con los NFTs del usuario disponibles para subastar.
4. Seleccionar la obra que se desea subastar (Figura 59).
5. Hacer scroll hacia abajo e introducir (Figura 60):
    a. Precio mínimo en ETH.
    b. Duración de la subasta (en minutos y segundos).
6. Pulsar el botón “Crear subasta”.
7. Confirmar la transacción en MetaMask (Figura 61), donde se muestra:
    a. El token que será subastado.
    b. La dirección del contrato de subastas.
    c. El coste estimado de gas en SepoliaETH.
8. Una vez confirmada, la subasta queda publicada y visible para todos los usuarios de la plataforma (Figura 62).

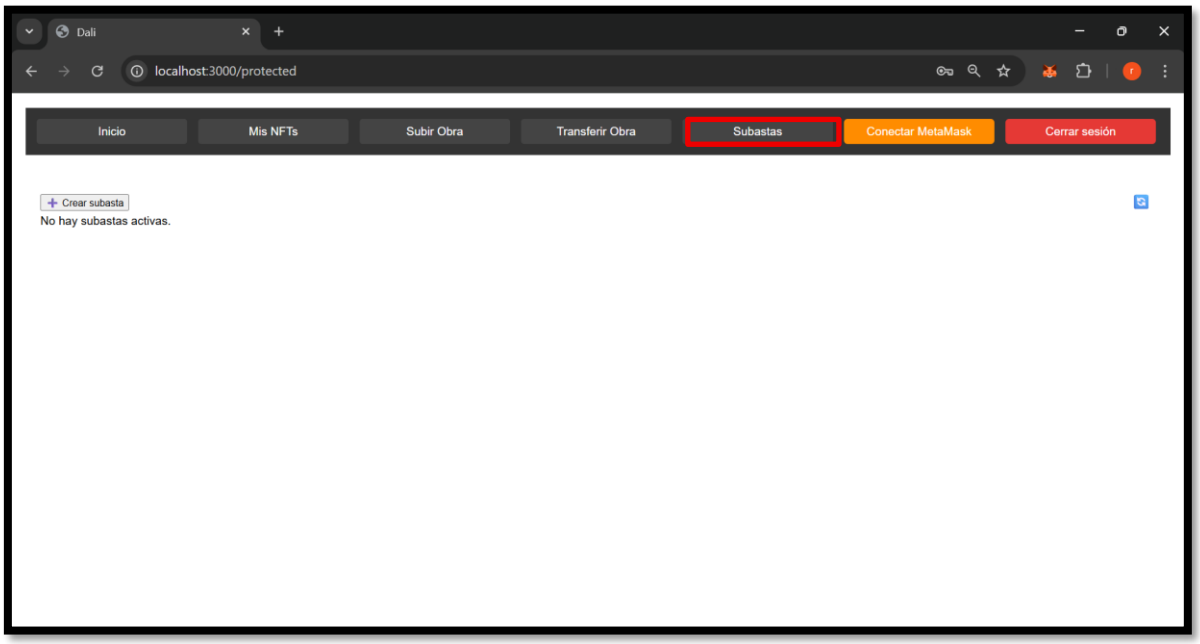

Figura 58. Sección de crear subasta

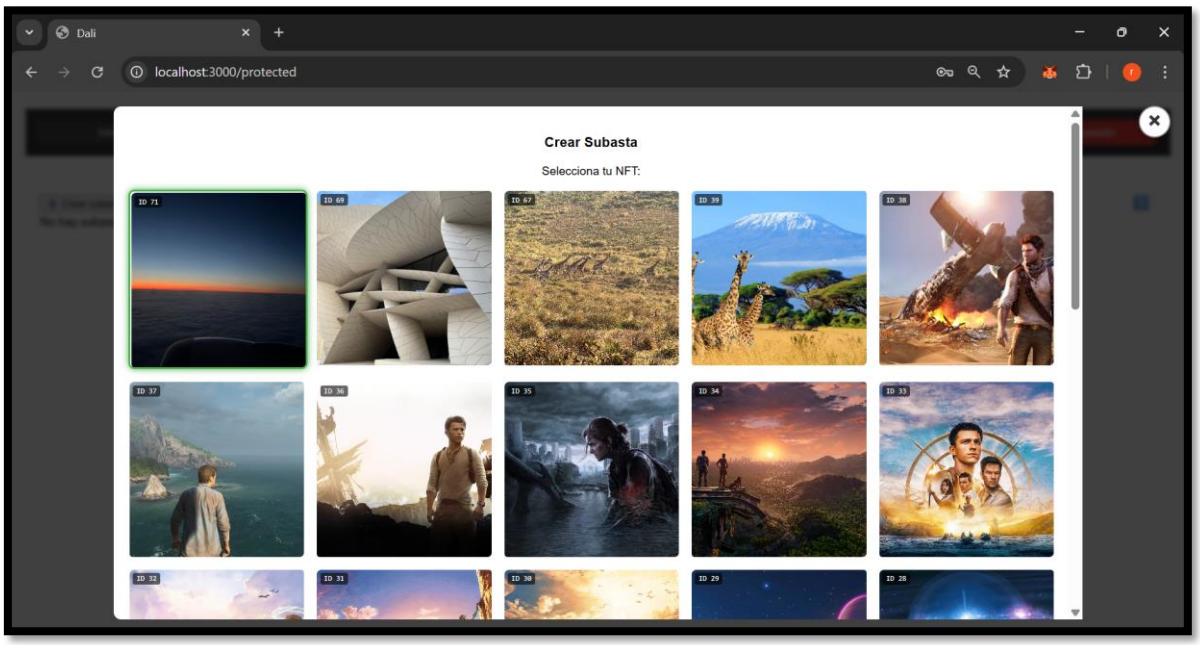

Figura 59. Seleccionar obra a subastar

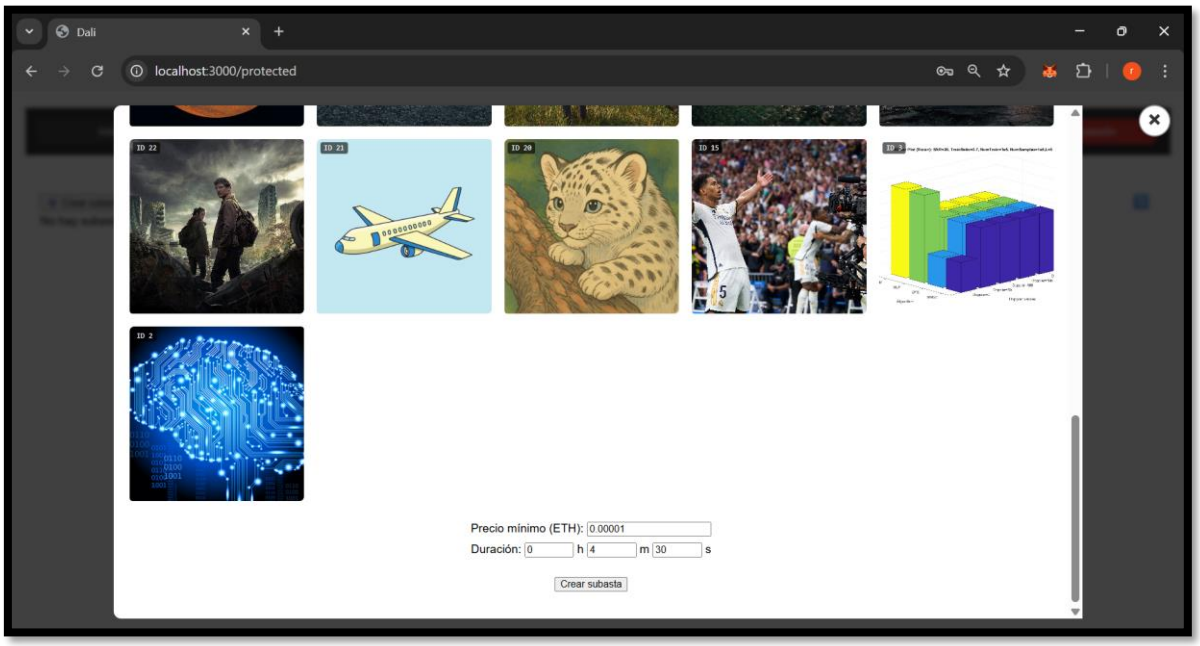

Figura 60. Introducir cantidad y duración

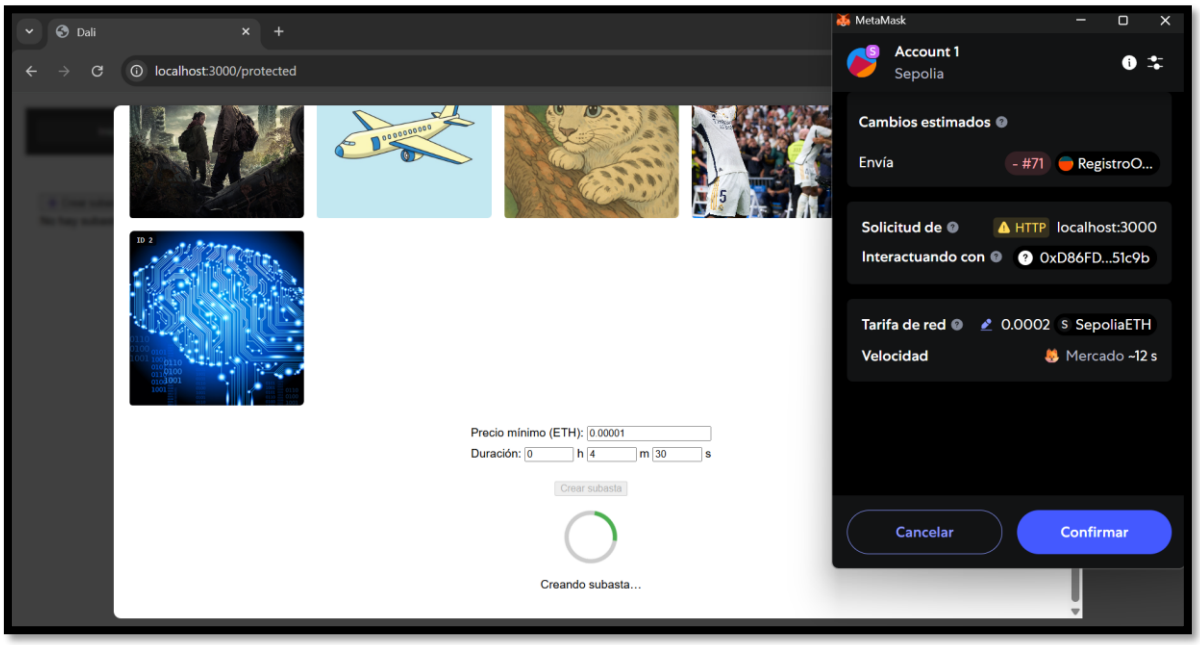

Figura 61. Confirmar la operación con MetaMask

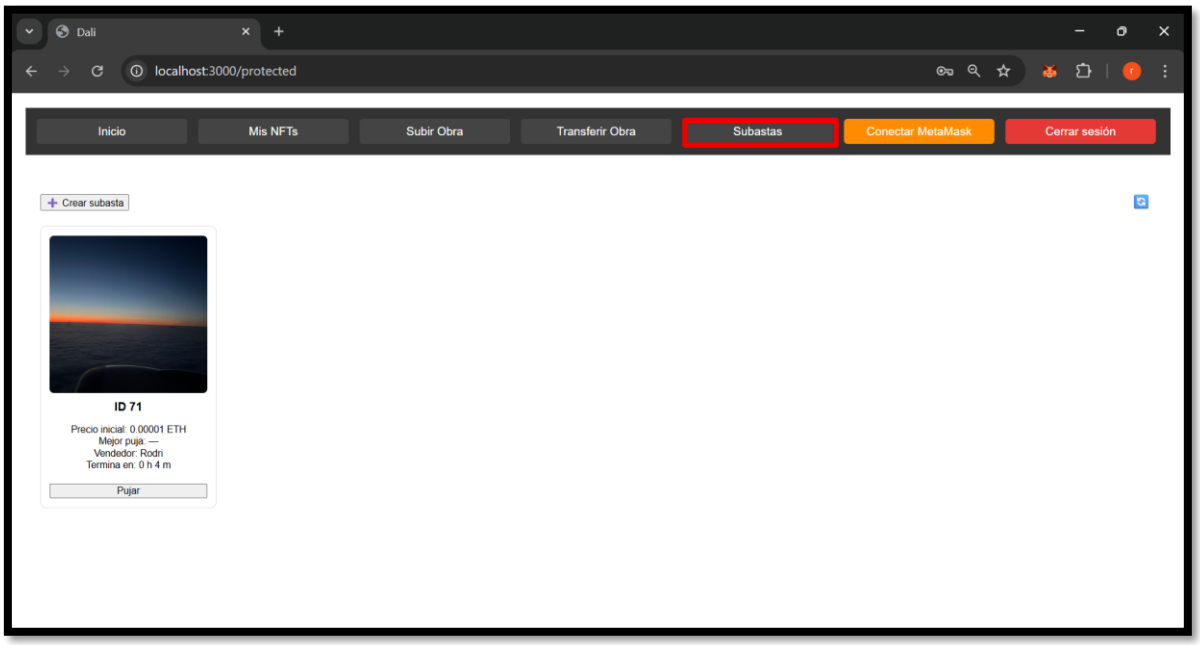

Figura 62. La subasta se puede observar desde Rodri

Cualquier usuario autenticado puede pujar por las obras disponibles en subasta. La puja más alta será la ganadora una vez se alcance el tiempo límite definido al crear la subasta. Acciones para participar en una subasta:

1. Acceder a la sección “Subastas” (usuario: Yago) y explorar las subastas activas (Figura 63).
2. Elegir una obra en subasta e introducir una oferta en ETH superior a las anteriores (Figura 64).
3. Pulsar el botón de confirmar puja.
4. Firmar la transacción en MetaMask, que valida la puja en la blockchain (Figura 65).
5. Esperar a que termine el tiempo de la subasta (Figura 66).
6. Al usuario con la puja más alta le aparece la función *finalize()* del contrato inteligente (Figura 67), que:
    a. Le transfiere el NFT a su cuenta.
    b. Envía los fondos al vendedor original.
7. El nuevo propietario puede comprobar que el NFT ha sido añadido a su galería personal de forma automática (Figura 68).

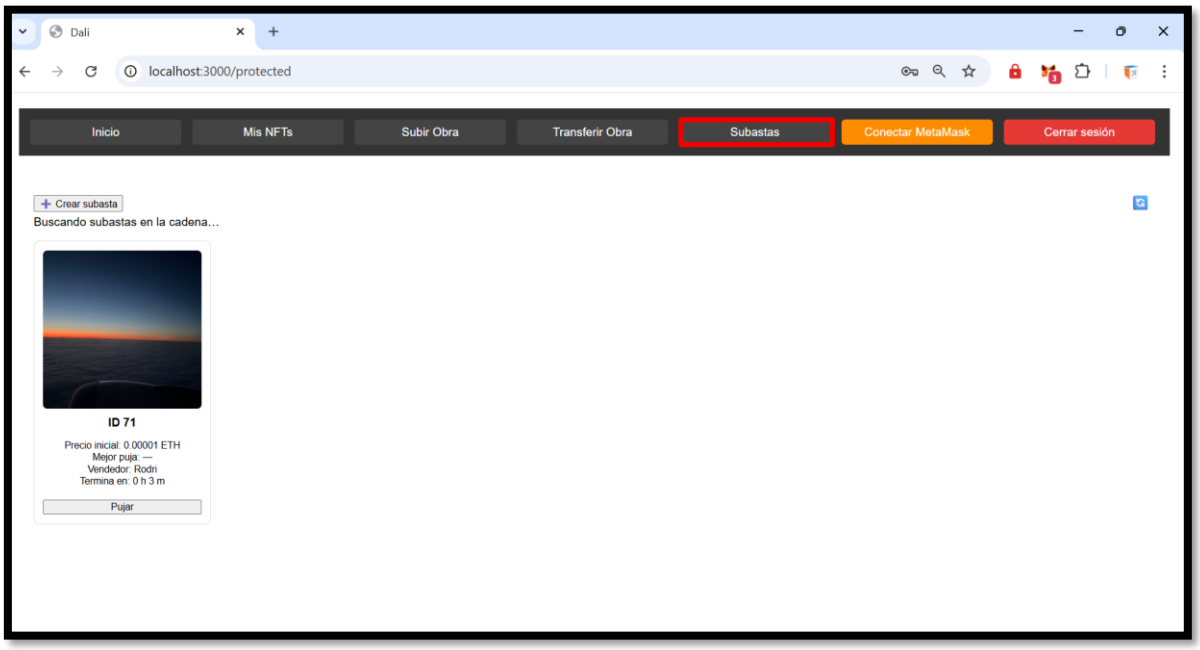

Figura 63. La subasta se puede observar desde Yago

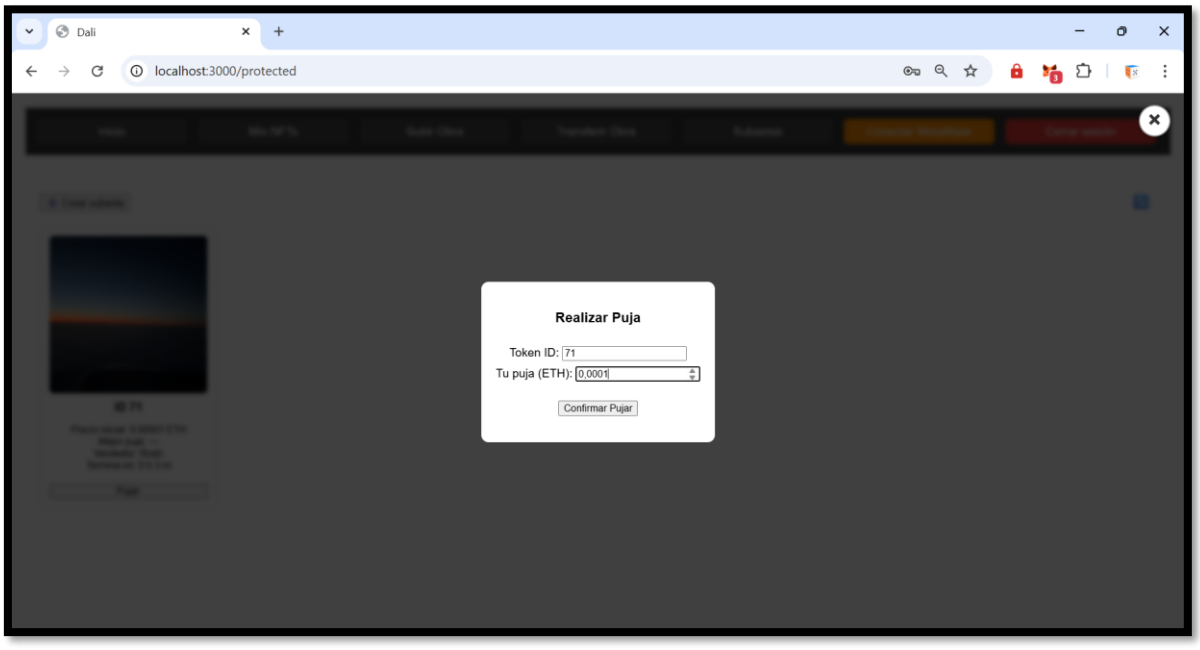

Figura 64. Usuario: Yago, establece la cantidad de Sepolia a pujar

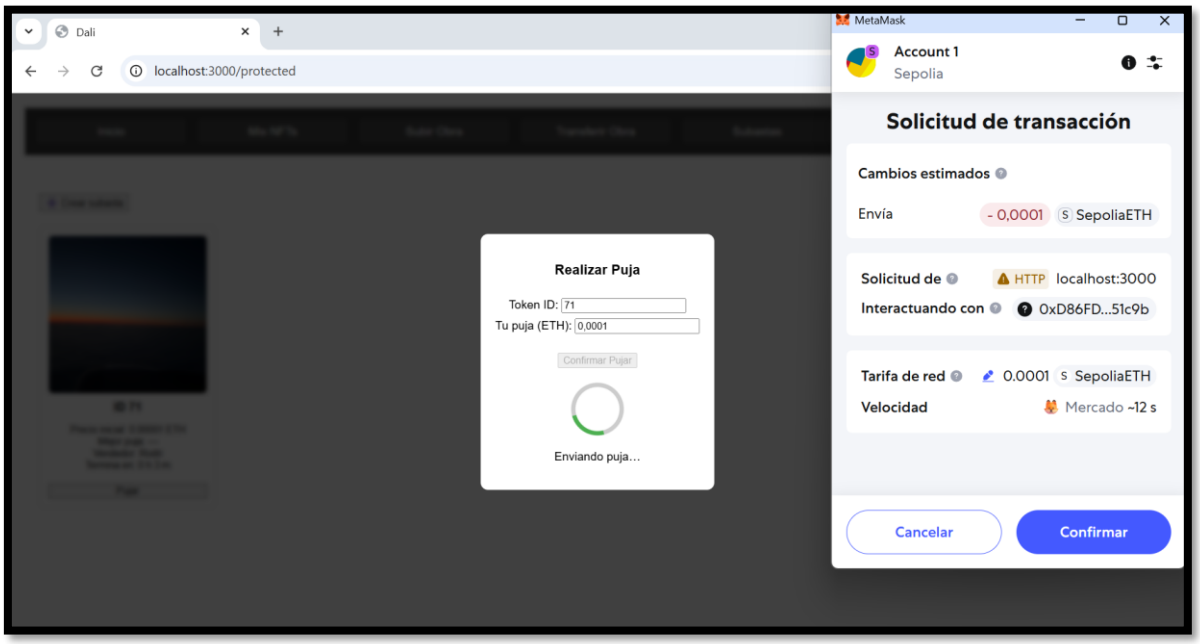

Figura 65. Se confirma la operación en MetaMask

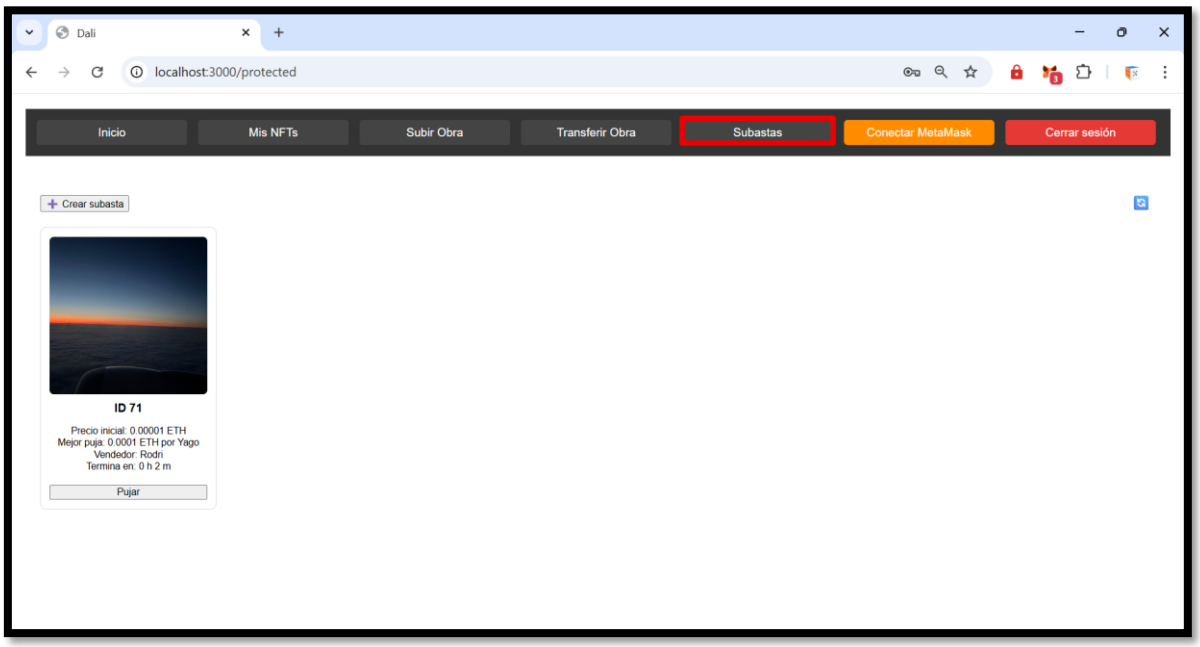

Figura 66. Se puede observar el usuario que ha pujado más alto

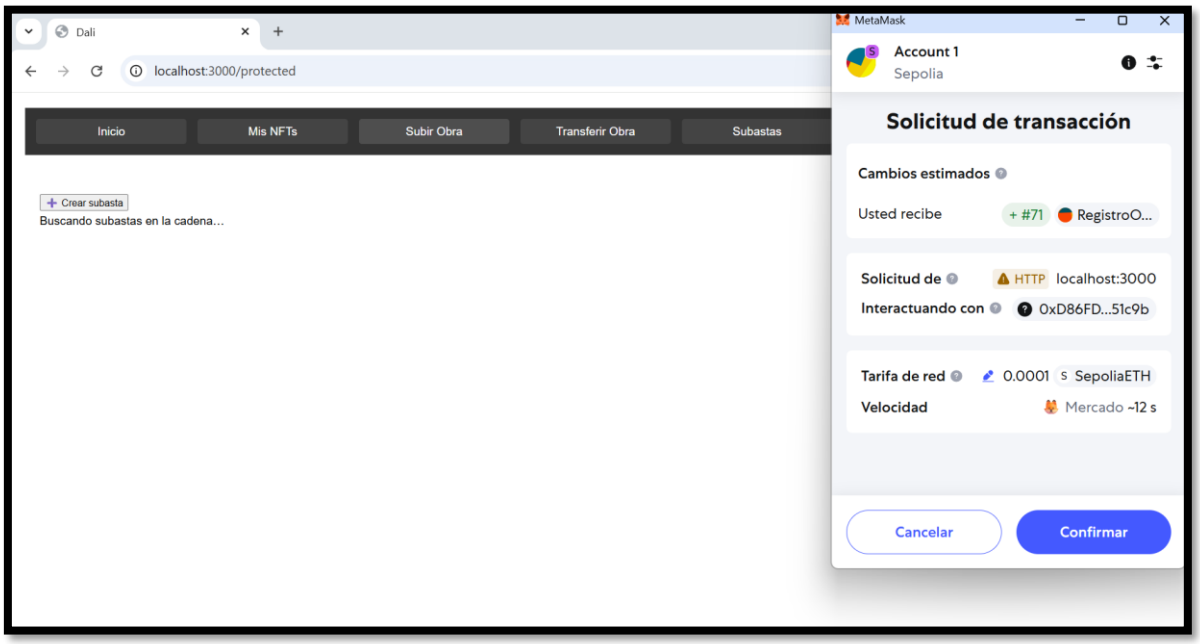

Figura 67. Finalmente se acepta el pago a la subasta

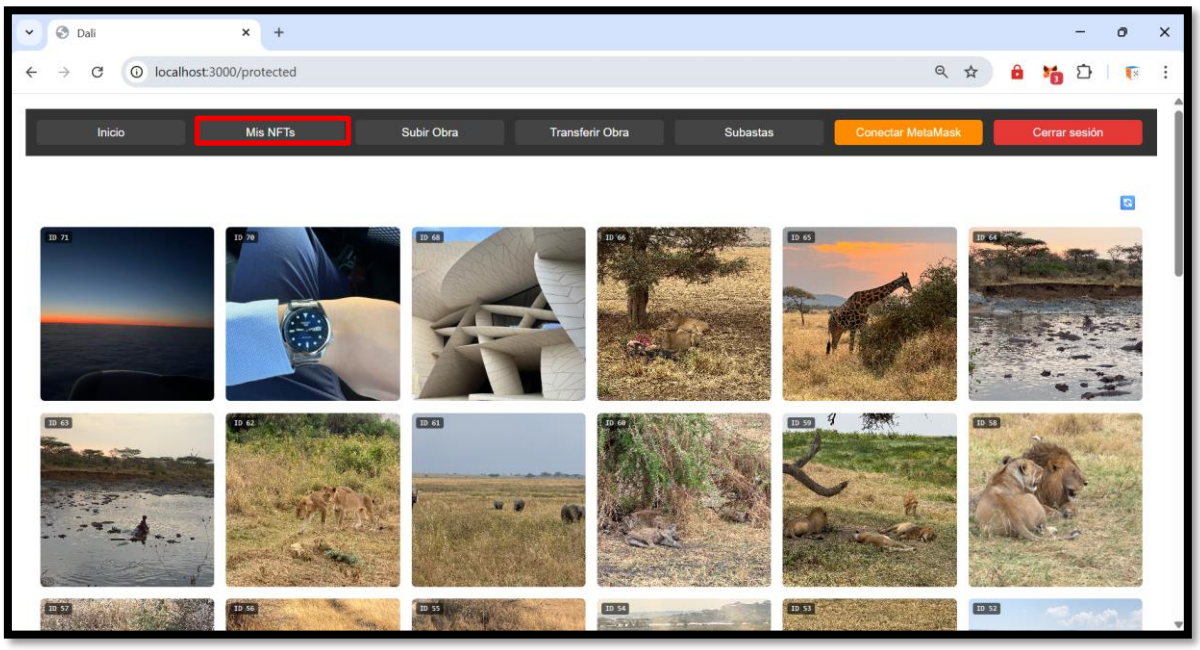

Figura 68. La obra vuelve a estar en el perfil de Yago