



Blockchain
Trabajo Tutelado : Subastas

Adrián Robles Camporro, David Olivares Muñoz

December, 2023

Contents

1	Introducción	3
1.1	Motivación	3
1.2	Definición del escenario	3
1.3	Objetivos	3
2	Estudio del estado del arte	3
2.1	Análisis de soluciones parecidas	4
2.2	Comparativa con la solución propuesta	4
2.2.1	Funcionalidades	4
2.2.2	Ciberseguridad	5
3	Análisis	5
3.1	Actores	5
3.2	Requisitos	5
3.3	Justificación del uso de Ethereum	6
4	Diseño	6
4.1	Casos de uso	6
4.2	Diagrama de secuencia	10
4.3	Arquitectura de comunicaciones	10
4.4	Modelo de datos	11
4.5	Tecnologías utilizadas	11
5	Implementación	11
5.1	Implementación de los Smart Contracts	11
5.2	Implementación de la aplicación	16
6	Demostración del funcionamiento del sistema	16
6.1	Despliegue	16
7	Análisis de los riesgos de seguridad (e.g. SRM)	19
8	Evaluación y pruebas	20
8.1	Prueba 1: Iniciar subasta	20
8.2	Prueba 2: Pujar por la obra de arte	21
8.3	Prueba 3: Finalizar subasta y obtener información	22
8.4	Prueba 4: Obtener refunds y retirarlos	22
8.5	Prueba 5: Añadir más subastas e IPFS	23
9	Aspectos Legales	25
10	Manual de usuario	25
10.1	Estructura del proyecto	25
11	Conclusiones	27
12	Líneas futuras	27
13	Lecciones aprendidas	27
13.1	Incidencias	28
14	Referencias	29

1 Introducción

1.1 Motivación

La motivación detrás del desarrollo de una aplicación de subasta con smart contracts y OrbitDB reside en la creciente necesidad de contar con sistemas descentralizados y seguros para realizar transacciones económicas. Las subastas, como método de venta, han evolucionado en la era digital, y la incorporación de contratos inteligentes (smart contracts) y tecnologías de bases de datos descentralizadas como OrbitDB ofrece beneficios significativos en términos de transparencia, confianza y eficiencia.

1.2 Definición del escenario

El escenario para esta aplicación se sitúa en el ámbito de las subastas en línea, donde los participantes pueden realizar ofertas y pujas de manera descentralizada. Utilizando smart contracts en una plataforma blockchain, la aplicación garantiza la ejecución automática de las reglas de la subasta, eliminando la necesidad de intermediarios y proporcionando un entorno confiable para compradores y vendedores. Además, la implementación de OrbitDB como sistema de almacenamiento descentralizado permite mantener un historial inmutable de las transacciones y eventos de la subasta, garantizando la integridad de los datos y la accesibilidad descentralizada.

1.3 Objetivos

Los objetivos principales de este trabajo son:

1. Desarrollar una plataforma de subastas en línea utilizando contratos inteligentes en una blockchain, en este caso Ethereum.
2. Implementar la lógica de subastas para permitir la oferta y la puja descentralizadas, así como gestionar las transacciones monetarias.
3. Integrar OrbitDB para el almacenamiento descentralizado de datos relacionados con las subastas, garantizando la transparencia y la inmutabilidad (en planes futuros).
4. Desarrollar una aplicación descentralizada (dApp) con interfaz de usuario intuitiva y segura para los participantes en la subasta.
5. Garantizar la seguridad y la resistencia a ataques mediante prácticas de desarrollo seguras y pruebas exhaustivas.

Con la consecución de estos objetivos, la aplicación busca ofrecer una solución confiable y transparente para realizar subastas en línea, aprovechando la descentralización y la seguridad proporcionadas por la tecnología blockchain y OrbitDB.

2 Estudio del estado del arte

A día de hoy, el tema de la tecnología blockchain sigue creciendo y evolucionando, y continua siendo uno de los temas más candentes de la sociedad. Cada día hay actualizaciones sobre el tema y más noticias sobre los avances en ese mundo. En cuanto a las casas de subasta de obras de arte aún está en una etapa temprana y tiene un gran potencial aún por explotar, y es por eso que existe este movimiento social en torno a los criptoactivos. Se considera que la constante evolución de la tecnología es un parámetro fundamental antes del análisis del mercado y del mundo alrededor de blockchain.

Se ha tomado como el inicio del estado del arte, un acontecimiento que tuvo un importante impacto en la sociedad y que puede considerarse uno de los detonantes de su popularidad, como es la millonaria subasta del NFT de Beeple obra, titulada "Everydays: The First 5000 Days" (Todos los días: los primeros 5.000 días), por parte de [Christie's](#), valorada en 69 millones de dólares. Fue realizada en marzo del 2021. A medida que la tecnología avanza y los contratos inteligentes continúan su desarrollo, las subastas están experimentando una transformación significativa. La naturaleza descentralizada y transparente de la cadena de bloques ofrece un entorno ideal para las subastas.

En esta búsqueda encontramos, por ejemplo, proyectos de Casa de subasta que están aún en desarrollo,

en este caso específico, nos referimos a [AuctionHouse: An Ethereum Dapp For Auctioning On-Chain Goods](#), la cual se trata de una aplicación descentralizada construida en Ethereum para facilitar la subasta de bienes en cadena. El proyecto se enfoca en subastar bienes virtuales, nombres de dominio y tokens que representan activos del mundo real, como boletos. Es una version en alpha, por lo que aún se encuentra en desarrollo y su idea de subastar bienes virtuales aún no se ha conseguido. AuctionHouse fue construida por Doug Petkanics y Eric Tang, y ellos mismos indican que la crearon para adquirir experiencia en DApp, interfaz de usuario e implementación descentralizada, es por ello que sus [Smart Contracts](#) son públicos y pueden ser analizados por cualquiera que lo desee.

2.1 Análisis de soluciones parecidas

En otro ámbito, las subastas con contratos inteligente no solo se basan en el ámbito del arte y sus obras, también se ha expandido a otras áreas y servicios.

- Subastas de arte y coleccionables: los contratos inteligentes de subasta se pueden utilizar para realizar subastas de arte digital, objetos de colección y otros activos digitales. Los artistas y creadores pueden tokenizar sus obras de arte digitales o objetos de colección como tokens no fungibles (NFT) y utilizar contratos inteligentes de subasta para venderlos al mejor postor.
- Subastas de bienes raíces: Los contratos inteligentes de subasta pueden agilizar el proceso de subasta al permitir subastas directas entre pares en blockchain. Las propiedades inmobiliarias se pueden tokenizar como NFT y se pueden utilizar contratos inteligentes de subasta para facilitar el proceso de licitación y subasta, garantizando transparencia, seguridad y eficiencia.
- Subastas de juegos y activos virtuales: los contratos inteligentes de subasta se pueden utilizar en mercados de juegos y activos virtuales, donde los jugadores pueden comprar, vender o intercambiar activos virtuales, como artículos del juego o monedas digitales.

Además, la tecnología blockchain garantiza la integridad del historial de las transacciones. Cada puja y transacción se registra de forma inmutable en bloques enlazados, lo que brinda transparencia y confianza a los participantes al eliminar la posibilidad de alteraciones o fraudes.

2.2 Comparativa con la solución propuesta

La casa de subastas Christie's es la más conocida a nivel mundial ya que proporcionó una subasta cuya transacción se realiza a través de un contrato inteligente de Blockchain, lo cual marca dos hitos en la industria: es la primera obra de arte completamente digital que se subasta y por primera vez se aceptan criptomonedas para su pago. En base a esta casa de subasta de obras se explica mediante la siguiente tabla comparativa las actividades que realiza nuestro contrato inteligente.

Acciones	Casa Subasta Propia	Casa Subasta Christie's
Registro de Obras de Arte	Registrada previamente	Registro (NFT)
Subasta	Puja y recupera dinero	Pujan los interesados
Pago	Pago en ETH	Pago en ETH

Si bien, en la actualidad, la Casa de Subasta Christie's cuenta con una nueva plataforma llamada Christie's 3.0 la cual registra todas sus subastas a través de la red Blockchain de Ethereum y genera ventas de NFTs.

2.2.1 Funcionalidades

En función del trabajo que realiza nuestra aplicación Subastas de Obras de Artes y la Casa de Subasta Christie's, podemos asegurar que ambas presentan rasgos generales muy comunes. En ambos se debe cargar una Wallet como Metamask para generar la transacción de manera segura y sin intermediarios, además todas las transacciones se realizan a través de una blockchain de Ethereum. Sin embargo, en nuestro proyecto la identificación de usuarios se hace automática mediante su address mientras que en Christie's, es necesaria una autenticación para saber con qué permisos cuenta el usuario.

2.2.2 Ciberseguridad

Para realizar el estudio del estado del arte en base a la seguridad informática, se siguen los estándares globales tales como ISO/IEC 27001 y que deben ser aplicados en las casas de subastas:

- Encriptación de Datos para proteger la información confidencial y financiera.
- Seguridad de Redes para proteger las redes y los sistemas de información.
- Autenticación y Control de Acceso robustos para personal autorizado.
- Actualizaciones y Parches de Seguridad en sistemas y aplicaciones para mitigar vulnerabilidades.
- Monitoreo continuo a sistemas y redes en búsqueda de actividades inusuales.
- Capacitación del Personal en ciberseguridad y prácticas seguras.

Las prácticas de seguridad específicas de Christie's no se encuentran públicas por razones de seguridad y confidencialidad. Esto es así para resguardar la información de la empresa pero sí podemos concluir que se están cumpliendo todos los puntos anteriores y que resulta ser una aplicación extremadamente segura (de ahí su gran popularidad). En el caso de nuestro proyecto de la Casa de subasta de Obras de Artes contamos con:

- Seguridad de la información de los usuarios.
- Seguridad de las transacciones utilizando un wallet de MetaMask.
- En el apartado web se utiliza HTML básico, ya que es un proyecto ingresado en máquina virtual.

A futuro, se plantea llevar el proyecto a un ambiente real con el despliegue de un servicio OrbitDB y la mejora de los aspectos de seguridad en los que se falla que quedarán detallados más adelante.

3 Análisis

3.1 Actores

- Usuarios de la subasta: Usuarios registrados en la plataforma que participan activamente en las subastas, realizando ofertas de obras de arte y pujas, por tanto, pueden desempeñar los roles de "Propietario" o "Pujador".
- Administradores: Responsables de gestionar y supervisar el funcionamiento general de la plataforma, verificando la legitimidad de las subastas y abordando problemas de cumplimiento. En nuestro caso, sería la casa de subastas que serían los encargados de desplegar el contrato.
- Smart Contracts: Códigos automatizados que ejecutan las reglas de las subastas de manera autónoma y segura.

3.2 Requisitos

- Identificación de Usuarios: Los participantes deben poder ser identificados en la plataforma a través de la dirección de su cartera en MetaMask.
- Inicio de Subastas: Los usuarios propietarios de obras de arte deben tener la capacidad de iniciar nuevas subastas, definiendo el precio mínimo de subasta, y detalles del artículo. El tiempo de duración de la subasta viene predefinido por la casa de subastas al desplegar el contrato.
- Realizar Pujas: Los participantes deben poder realizar pujas en las subastas activas hasta que estas sean finalizadas por la casa de subastas y siempre que su puja supere la mayor registrada para esa obra.
- Gestión de Transacciones: Los participantes deben poder recuperar el importe de sus pujas si no han resultado ser los ganadores de la subasta y los propietarios deben poder obtener el importe de la mayor puja que le corresponde tras finalizar la subasta.

- **Ejecución Automática de Subastas:** Los smart contracts deben ejecutar automáticamente las reglas de la subasta, declarando al ganador al final del período de subasta.
- **Gestión de la Información:** Utilizar OrbitDB/IPFS para almacenar de manera descentralizada la información de las obras subastadas, asegurando un historial inmutable y accesible.
- **Interfaz de Usuario Intuitiva:** Desarrollar una interfaz de usuario intuitiva que permita a los participantes realizar acciones de manera fácil y segura.
- **Seguridad:** Implementar medidas de seguridad robustas para proteger la integridad de la plataforma y la información sensible de los usuarios.
- **Cumplimiento Normativo:** Garantizar que la plataforma cumple con las normativas y estándares legales relacionados con las subastas en línea.

3.3 Justificación del uso de Ethereum

A continuación, se detallan los principales motivos por los que está justificado el uso de Ethereum para las subastas en línea:

- **Smart Contracts:** Ethereum ofrece la capacidad de implementar contratos inteligentes, lo que facilita la automatización y ejecución confiable de las reglas de las subastas sin necesidad de intermediarios.
- **Seguridad:** La robustez y seguridad inherentes a la blockchain de Ethereum proporcionan un entorno confiable para la ejecución de transacciones financieras, esencial en un contexto de subastas en línea.
- **Ecosistema Desarrollado:** Ethereum cuenta con un ecosistema desarrollado de herramientas y recursos que simplifican el desarrollo y despliegue de aplicaciones descentralizadas, como la plataforma de subastas propuesta.
- **Descentralización:** La naturaleza descentralizada de Ethereum contribuye a la transparencia y confianza en el proceso de subastas, eliminando la dependencia de una autoridad centralizada.
- **Estándares de Tokens:** La posibilidad de utilizar estándares como ERC-20 o ERC-721 permite representar activos y bienes de manera estandarizada en la blockchain, facilitando la representación de los artículos subastados.

4 Diseño

En este apartado se realizará el diseño para la aplicación, indicando los casos de uso de la misma, así como las tecnologías utilizadas para desarrollar el proyecto y la forma en la que se estructura la arquitectura de comunicaciones.

4.1 Casos de uso

A continuación se detallan los casos de uso del Smart Contract:

Table 1: Caso de Uso 1: Iniciar Subasta (CU-01)

Campo	Descripción
ID del Caso de Uso	CU-01
Nombre del Caso de Uso	Iniciar Subasta
Descripción	Permite a los propietarios de obras de arte iniciar la subasta de una obra de arte concreta.
Actores Involucrados	Propietario de las obras
Escenarios Principales	<ol style="list-style-type: none"> 1. El propietario proporciona el título de la obra de arte, el autor de la obra, el precio inicial con el que sale a subasta y la dirección del contrato dónde está guardado el activo (obra). Esta información se guarda en un archivo en IPFS (se desearía guardar en OrbitDB) 2. Se crea un nuevo objeto ObraDeArte y se establece el momento de final de subasta. Además, se guarda la información del ganador actual (siendo nulo en un principio) 3. Se inicia el período de subasta.

Table 2: Caso de Uso 2: Realizar oferta (CU-02)

Campo	Descripción
ID del Caso de Uso	CU-02
Nombre del Caso de Uso	Realizar oferta
Descripción	Permite a los usuarios realizar ofertas de dinero (weis) por una obra de arte.
Actores Involucrados	Usuarios con Ethers ("Pujadores")
Escenarios Principales	<ol style="list-style-type: none"> 1. El usuario decidirá pujar una cantidad de Ethers por una obra de arte. 2. Si es la primera vez que puja se generará un registro para guardar posibles devoluciones que se le deben hacer. 3. Si la cantidad de dinero apostada por este usuario es la mayor hasta el momento, se convierte en el ganador provisional.
hline	

Table 3: Caso de Uso 3: Finalizar Subasta (CU-03)

Campo	Descripción
ID del Caso de Uso	CU-03
Nombre del Caso de Uso	Finalizar subasta
Descripción	Permite al administrador (casa de subastas) cerrar la subasta una vez haya finalizado el tiempo de subasta.
Actores Involucrados	Casa de Subastas
Escenarios Principales	<ol style="list-style-type: none"> 1. La casa de subastas decide cerrar la subasta de una obra de arte concreta (el tiempo de pujas debe haber finalizado). 2. Si hay un ganador se traspasa el activo subastado a este usuario y se almacenan los valores monetarios finales a recuperar por cada usuario que ha pujado y por el propietario de la obra de arte.

Table 4: Caso de Uso 4: Recuperar ETH (CU-04)

Campo	Descripción
ID del Caso de Uso	CU-04
Nombre del Caso de Uso	Recuperar ETH
Descripción	Permite al propietario de la obra de arte o a un usuario que puja obtener los ETH que le corresponden de la transacción.
Actores Involucrados	Propietario, Usuarios que pujan
Escenarios Principales	1. Se solicita saber qué cantidad de ethers le corresponden y se retiran de la bolsa.

Table 5: Caso de Uso 5: Obtener información de la subasta (CU-05)

Campo	Descripción
ID del Caso de Uso	CU-05
Nombre del Caso de Uso	Obtener información de la subasta
Descripción	Permite a cualquier usuario obtener información (ganador, precio de la mayor puja hasta el momento y estado de la subasta).
Actores Involucrados	Cualquier usuario
Escenarios Principales	1. Se solicita la información y una vez vista se retira (para poder ver otra)

Table 6: Caso de Uso 6: Registrar activo (CU-06)

Campo	Descripción
ID del Caso de Uso	CU-06
Nombre del Caso de Uso	Registrar activo
Descripción	Los propietarios de obras de arte deben registrar los activos que poseen
Actores Involucrados	Propietarios
Escenarios Principales	1. Se despliega un contrato que representa el activo del usuario.

A continuación, se puede ver un diagrama de flujo del funcionamiento del Smart Contract, así como el diagrama de los casos de uso, los cuales dan una visión más detallada del funcionamiento del contrato:

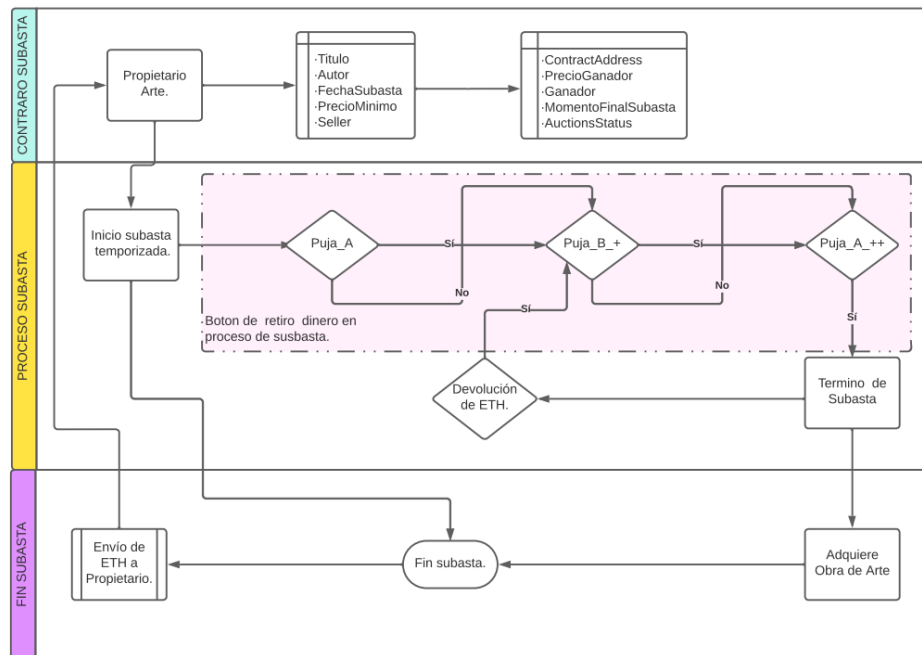


Figure 1: Diagrama de flujo del contrato

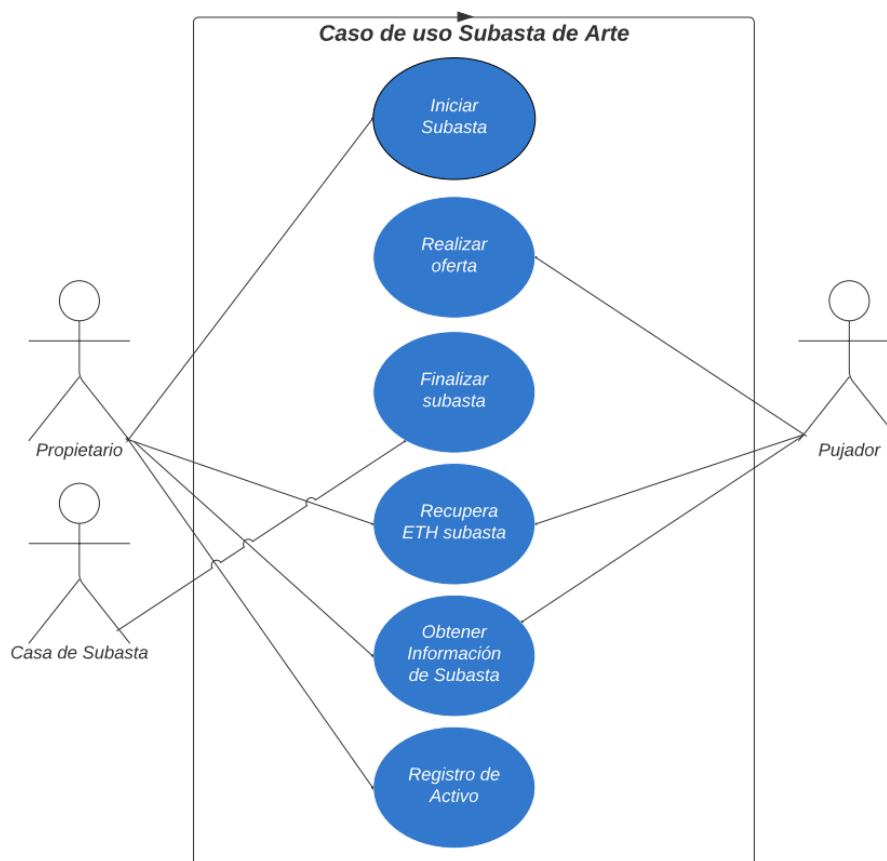


Figure 2: Diagrama de casos de uso del contrato

4.2 Diagrama de secuencia

A continuación, se muestra el diagrama de secuencia del proyecto, dónde se pueden ver las partes involucradas y cómo es la comunicación entre ellas en el desarrollo de una subasta.

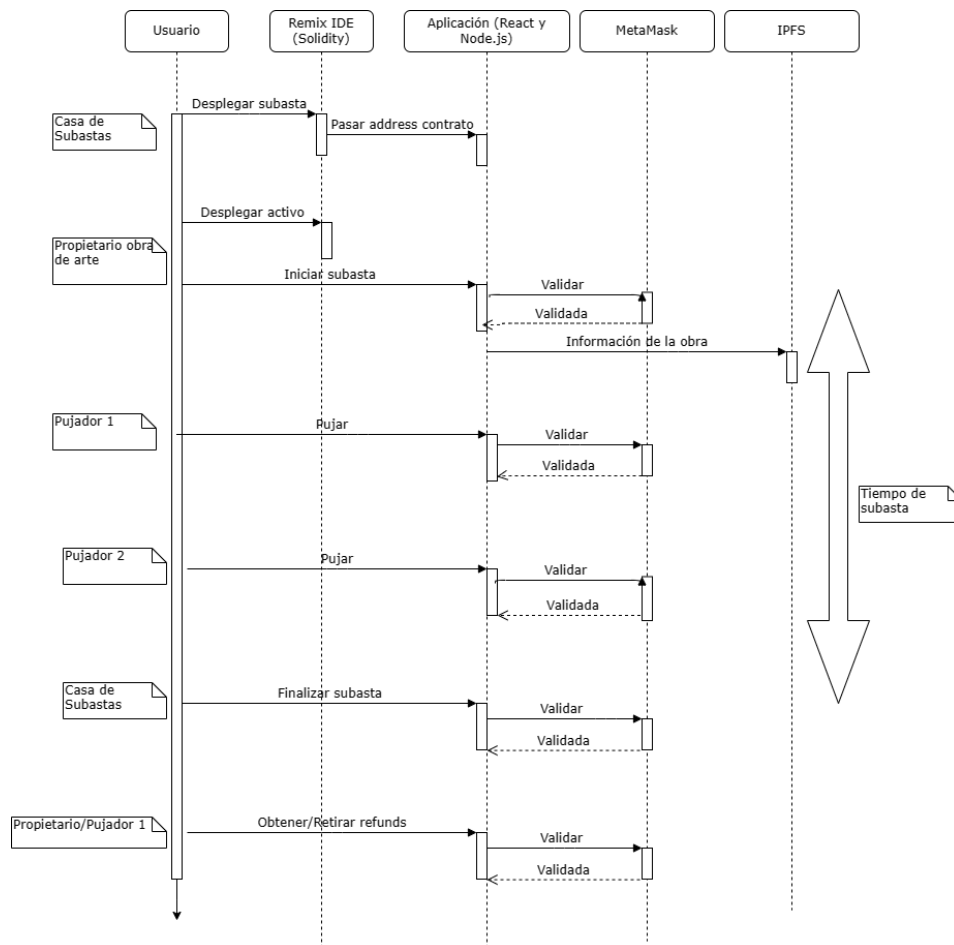


Figure 3: Diagrama de secuencia

4.3 Arquitectura de comunicaciones

En la figura 4 se muestra la arquitectura de comunicaciones del proyecto de subasta de obras de arte. Comienza con el despliegue del contrato de la subasta así como los contratos que representan cada obra de arte, seguido de la interacción del usuario en el sitio web. Esta interacción se realiza a través de un servidor remoto conectado a una máquina virtual Linux que aloja el IPFS. El Back-end recopila datos de las ofertas y gestiona la retirada de estas para avanzar en la subasta de las obras de arte. El servidor remoto establece el Front-end, permitiendo la interacción entre el IPFS y la página web de la casa de subastas, posibilitando que los usuarios inicien subastas de las obras de arte disponibles y también que puedan pujar por ellas.

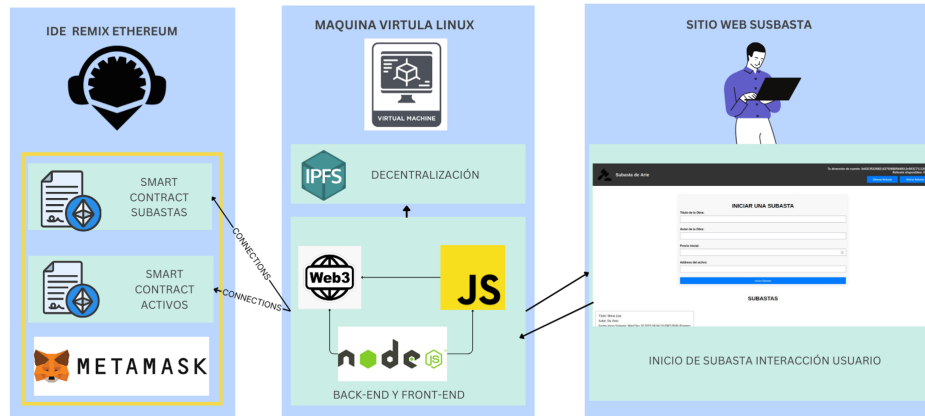


Figure 4: Modelo de Arquitectura Subastas Obras de Artes.

4.4 Modelo de datos

Para la casa de Subastas de Obras de Artes los datos que se utilizan son todos aquellos relacionados con la obra por un lado (información del autor de la obra de arte, título, precio inicial de la obra) y por otro, los relacionados con el desarrollo de la subasta (mayor puja, address de la persona que ha hecho la mayor puja y estado de la subasta). También hay que tener en cuenta los datos de MetaMask ya que son los que permiten identificar a los usuarios y representa sus carteras.

4.5 Tecnologías utilizadas

Las tecnologías que se han aplicado para este proyecto son las siguiente:

- IDE Remix : Plataforma para implementar y desplegar los Smart Contract.
- Metamask.io : Plataforma que actúa como una Wallet que permite a la aplicación del sitio web del proyecto comunicarse con la blockchain de Ethereum.
- IPFS : Plataforma para alojar los archivos con información sobre las obras de arte subastadas.
- Web3, React y Node.js : Aplicaciones utilizadas para implementar la aplicación descentralizada de la casa de subastas.
- HTML y CSS: Utilizados para definir la forma y el estilo de la página web principal de la aplicación.
- Kali Linux : Sistema Operativo que se utilizara como servidor virtual del proyecto. Desplegado en una máquina virtual para poder realizar pruebas de la aplicación.

5 Implementación

La implementación del proyecto presenta dos etapas, la primera consiste en implementar los Smart Contract necesarios para gestionar los eventos y transacciones de la casa de subastas, la segunda, en implementar la dApp que interactúa con estos contratos y que también se comunica con IPFS para almacenar la información deseada sobre las obras de arte. En ambas etapas, es necesario considerar MetaMask, ya que es el Wallet dónde los usuarios guardan sus ethers.

5.1 Implementación de los Smart Contracts

Para implemetar los Smart Contracts se ha empleado Remix IDE. En primer lugar, se debe desplegar un contrato que represente los activos (obras de arte). Este es un contrato sencillo que se muestra a continuación:

```

1  //SPDX-License-Identifier: GPL-3.0
2  pragma solidity ^0.8.10;
3
4  // Abstract contract for a not yet agreed upon standard for non-fungible
5  // on chain goods
6
7  contract Activo {
8      string private name;
9      address private ownerAddress;
10
11     constructor(string memory _name) {
12         ownerAddress = msg.sender;
13         name = _name;
14     }
15
16     function owner() public view returns (address _ownerAddress) {
17         return ownerAddress;
18     }
19
20     function setOwner(address _newOwner) public returns (bool success) {
21         ownerAddress = _newOwner;
22         return true;
23     }
24 }

```

En este contrato guardamos el nombre de la obra y la address del propietario. Además, se definen dos funciones: una para obtener el owner del activo y otra para modificar dicho owner.

A continuación, se pasa a detallar la implementación del contrato de la subasta. Se van a explicar las funciones principales ya que hay algunas funciones en el contrato usadas para obtener cierta información pero no utilizadas en la aplicación.

En primer lugar, se definen las estructuras, arrays y mappings del contrato, en los que se guardará toda la información.

```

1  //SPDX-License-Identifier: GPL-3.0
2  pragma solidity ^0.8.10;
3
4  import "./Activo.sol";
5
6  contract SubastaArte{
7      address public propietario;
8      uint public duracionSubasta;
9
10     struct Apuesta {
11         address bidder;
12         uint256 amount;
13         uint256 timestamp;
14     }
15
16     enum AuctionStatus {Active, Inactive}
17
18     struct ObraDeArte {
19         string titulo;
20         string autor;
21         uint256 fechaSubasta;
22         address seller;

```

```

23     address contractAddress; // Contract where the item exists
24     uint256 momentoFinalSubasta;
25     uint256 precioMinimo;
26
27 }
28
29 ObraDeArte[] public subastas;
30 mapping(uint => Apuesta[]) apuestas;
31 mapping(address => uint[]) public auctionsRunByUser;
32 // Pointer to auctions index for auctions run by this user
33 mapping(address => uint[]) public auctionsBidOnByUser;
34 // Pointer to auctions index for auctions this user has bid on
35 mapping(address => uint256) refunds;
36 mapping(uint => uint) preciosGanadores;
37 mapping(uint => address) Ganadores;
38 mapping(uint => AuctionStatus) status;

```

Podemos ver dos estructuras, “Apuesta”, en la que se guarda la address de quién realiza la apuesta, la cantidad pujada y el momento en el que se realizó, y “ObradeArte”, en la que se guarda toda la información importante de la obra (título, autor, propietario, dirección del contrato del activo, precio con el que sale a subasta...). Adicionalmente, se define un array para guardar las obras de arte (subastas) y varios mappings para guardar las apuestas para cada obra (apuestas), las cantidades a devolver a cada usuario (refunds) y los ganadores eventuales de la subasta o el estado de la subasta (preciosGanadores, Ganadores, status). Nótese que el estado de la subasta puede ser “Active” (0) o “Inactive” (1).

Los modifiers y el constructor del contrato se pueden ver a continuación, encontramos modifiers para que ciertas funciones sólo las pueda ejecutar la casa de subastas, el propietario del activo o sólo se puedan realizar si la subasta está activa. En el constructor, la casa de subastas debe indicar el tiempo de duración de la subasta.

```

1  constructor(uint _duracionSubasta) {
2      propietario = msg.sender;
3      duracionSubasta = _duracionSubasta;
4
5  }
6
7  modifier soloPropietario() {
8      require(msg.sender == propietario, "Solo el
9      propietario puede llamar a esta funcion.");
10     _;
11 }
12
13 modifier onlySeller(uint256 auctionId) {
14     require(subastas[auctionId].seller == msg.sender, "Not the seller");
15     _;
16 }
17
18 modifier subastaActiva(uint obraID) {
19     require(status[obraID] == AuctionStatus.Active, "Subasta no activa");
20     require(block.timestamp < subastas[obraID].momentoFinalSubasta,
21     "La subasta ha finalizado.");
22     _;
23 }

```

Por último, definimos las funciones principales:

- Función para iniciar una subasta: Se pasa la información de la obra de arte y se genera un nuevo objeto `ObradeArte` así como se inicializan todos los parámetros relacionados con la obra.

```

1  function iniciarNuevaSubasta(string memory _titulo, string memory
2  _autor, address _contractAddressOfAsset,
3      uint _precioInicial) external returns (uint256) {
4      require(_precioInicial >=0 , "El precio de salida no puede ser menor que 0");
5
6      subastas.push(ObradeArte(_titulo, _autor, block.timestamp,msg.sender,
7      _contractAddressOfAsset,block.timestamp + duracionSubasta, _precioInicial));
8      auctionsRunByUser[msg.sender].push(subastas.length - 1);
9      preciosGanadores[subastas.length-1] = 0;
10     Ganadores[subastas.length-1] = address(0);
11     status[subastas.length-1] = AuctionStatus.Active;
12     return subastas.length-1;
13
14 }

```

- Función para realizar pujas indicando el ID de la obra por la que se quiere pujar (sólo se puede llamar si la subasta está activa), en esta función se comprueba que la puja sea mayor que la mejor puja y mayor que el precio inicial. En ese caso, se modifica los parámetros de ganador y precio ganador de la obra y se almacena la cantidad de la puja anterior para devolverla al usuario. además hay dos funciones para que el usuario pueda saber cuanto dinero se le debe traspasar y otra para que se le traspase dicha cantidad de dinero.

```

1  function realizarOferta(uint _obraID) external payable subastaActiva(_obraID) {
2      require(msg.value > subastas[_obraID].precioMinimo, "La oferta debe
3      superar el precio minimo.");
4      require(msg.value > preciosGanadores[_obraID],
5      "La oferta debe ser mayor que la oferta actual");
6
7
8      uint256 bidIdx = apuestas[_obraID].length;
9      apuestas[_obraID].push(Apuesta(msg.sender,msg.value,block.timestamp));
10     preciosGanadores[_obraID] = msg.value;
11     Ganadores[_obraID] = msg.sender;
12     if (auctionsBidOnByUser[msg.sender].length == 0) {
13         refunds[msg.sender] = 0;
14     }
15
16
17     // Log refunds for the previous bidder
18     if (bidIdx > 0) {
19         Apuesta memory previousBid = (apuestas[_obraID])[bidIdx - 1];
20         refunds[previousBid.bidder] += previousBid.amount;
21     }
22     auctionsBidOnByUser[msg.sender].push(_obraID);
23     emit NuevaOferta(_obraID, msg.sender, msg.value);
24 }
25
26 function getRefundValue() external view returns (uint256) {
27     return refunds[msg.sender];
28 }
29
30 function withdrawRefund() external {

```

```

31     uint256 refund = refunds[msg.sender];
32     refunds[msg.sender] = 0;
33     require(payable(msg.sender).send(refund), "Reembolso fallido");
34 }

```

- Función para finalizar una subasta: Sólo puede ser invocada por la casa de subastas y en ella se almacena el valor de la puja ganadora para que el propietario de la obra de arte lo pueda reclamar y se cambia el estado de la subasta a “Inactive”.

```

1  constructor(uint _duracionSubasta) {
2      propietario = msg.sender;
3      duracionSubasta = _duracionSubasta;
4
5  }
6
7  modifier soloPropietario() {
8      require(msg.sender == propietario, "Solo el
9      propietario puede llamar a esta funcion.");
10     _;
11 }
12
13 modifier onlySeller(uint256 auctionId) {
14     require(subastas[auctionId].seller == msg.sender, "Not the seller");
15     _;
16 }
17
18 modifier subastaActiva(uint obraID) {
19     require(status[obraID] == AuctionStatus.Active, "Subasta no activa");
20     require(block.timestamp < subastas[obraID].momentoFinalSubasta,
21     "La subasta ha finalizado.");
22     _;
23 }

```

- Función para iniciar una subasta: Se pasa la información de la obra de arte y se genera un nuevo objeto `ObradeArte` así como se inicializan todos los parámetros relacionados con la obra.

```

1  constructor(uint _duracionSubasta) {
2      propietario = msg.sender;
3      duracionSubasta = _duracionSubasta;
4
5  }
6
7  modifier soloPropietario() {
8      require(msg.sender == propietario, "Solo el
9      propietario puede llamar a esta funcion.");
10     _;
11 }
12
13 modifier onlySeller(uint256 auctionId) {
14     require(subastas[auctionId].seller == msg.sender, "Not the seller");
15     _;
16 }
17
18 modifier subastaActiva(uint obraID) {
19     require(status[obraID] == AuctionStatus.Active, "Subasta no activa");

```

```

20     require(block.timestamp < subastas[obraID].momentoFinalSubasta,
21     "La subasta ha finalizado.");
22     _;
23 }

```

- Funciones para obtener información sobre una obra de arte como el estado, el ganador o la cantidad de la mayor puja. Se debe pasar el ID de la obra.

```

1  function getStatus(uint256 idx) external view returns (uint256) {
2      return uint256(status[idx]);
3  }
4
5  function getPrecioGanador(uint256 idx) external view returns (uint256) {
6      return preciosGanadores[idx];
7  }
8
9  function getGanador(uint256 idx) external view returns (address) {
10     return Ganadores[idx];
11 }

```

5.2 Implementación de la aplicación

Para desarrollar la aplicación descentralizada en la máquina de Kali Linux se ha utilizado React, Node.js y Web3. En primer lugar se debe crear la dApp con “`npx create-react-app subasta_dapp`”. Debemos indicarle el ABI de nuestro contrato de subasta y del contrato de los activos, así como la address del contrato de subasta desplegado desplegado. Se ha implementado una aplicación que interactúa con las funciones del contrato anteriormente descritas y representa una aplicación en la que los usuarios pueden iniciar subastas y pujar por las obras de arte que quieran. Algunos detalles a destacar de la implementación son:

- Las funciones deben ser definidas como asíncronas de manera que se espera a que las llamadas a funciones del contrato sean ejecutadas.
- Se deben definir varias constantes globales para poder acceder a ciertos elementos (contrato de la subasta, cuenta del usuario...) en las funciones de la aplicación.
- Se deben instalar una serie de librerías para que la aplicación funcione: “`npm install ethers@5.7.0 kubo-rpc-client buffer`”, “`npm install react-toastify`”, “`npm install web3`”.
- Para guardar la información en IPFS, se debe desplegar un nodo que será accesible a través de la URL “localhost:5001”.
- El frontend de la interfaz se define mediante HTML y CSS, tratando de diseñar una interfaz sencilla y fácil de utilizar para todos los usuarios.

6 Demostración del funcionamiento del sistema

En la demostración del funcionamiento del sistema de nuestro proyecto, se necesita validar la viabilidad práctica de la casa de subastas en blockchain. En este punto, se muestra cómo el sistema se implementa y ejecuta en un entorno simulado.

6.1 Despliegue

El despliegue del proyecto implica la implementación efectiva de la infraestructura tecnológica que respalda la plataforma de subastas. Esto incluye la configuración de nodos de blockchain, la integración de contratos inteligentes, el establecimiento de wallets criptográficas, y la inicialización de la aplicación descentralizada.

En primer lugar, debemos obtener varias cuentas de MetaMask para poder probar la aplicación así, debemos instalar esta extensión en el navegador que vamos a utilizar (Firefox): <https://metamask.io>, y crear las cuentas necesarias (a las cuales debemos proporcionarles SepoliaETH a través de sepoliafaucet).

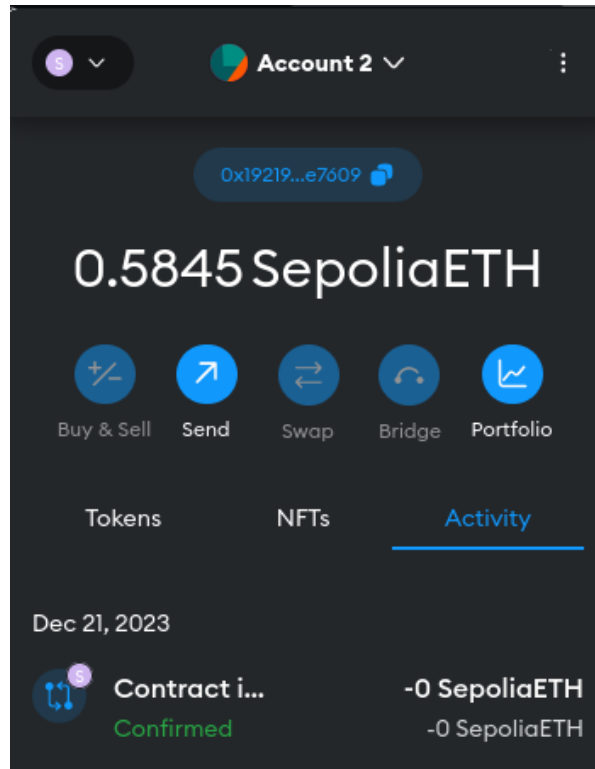


Figure 5: Extensión de MetaMask de Firefox

Tras esto, se deben desplegar los contratos en Remix IDE. Al contrato de los activos se le debe pasar el nombre de la obra de arte y al contrato de la subasta, la duración en segundos de la misma. Para poder desplegarlos se deben validar las transacciones en MetaMask y podremos ver lo siguiente:

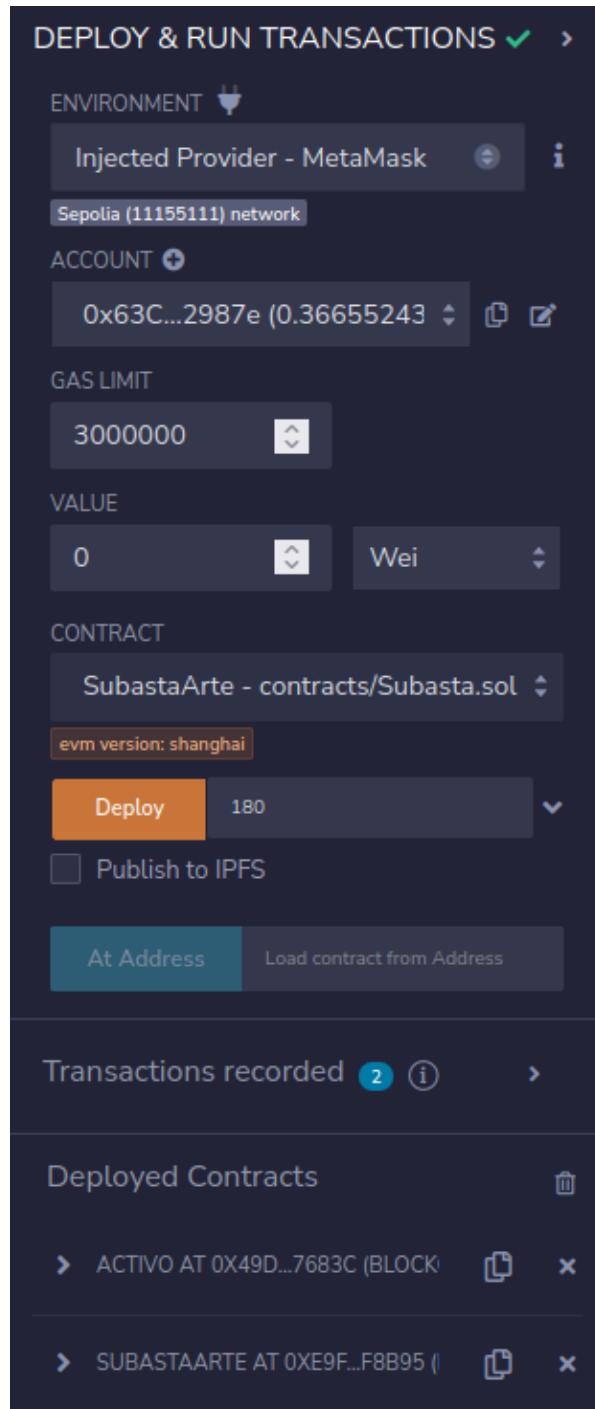


Figure 6: Extensión de MetaMask de Firefox

A continuación, debemos desplegar un nodo IPFS en la máquina virtual de Kali Linux para poder almacenar los archivos con información sobre las obras de arte. Para ello, debemos instalar Docker e iniciar el nodo con el comando “`docker run -d --name ipfs_host -v $PWD:/export -v $PWD:/data/ipfs -p 4001:4001 -p 4001:4001/udp -p 127.0.0.1:8080:8080 -p 127.0.0.1:5001:5001 ipfs/kubo`”. Tras esto, podremos acceder a la dirección “`http://127.0.0.1:5001`” y podremos ver el nodo desplegado.

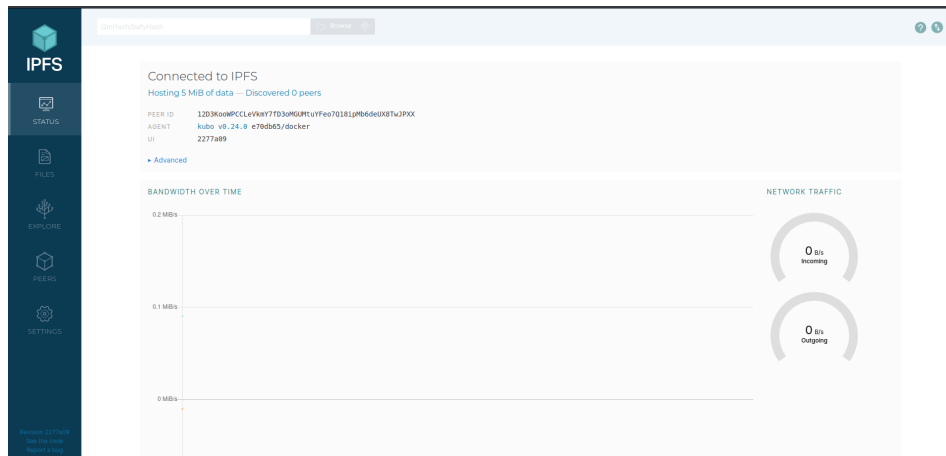


Figure 7: Nodo IPFS desplegado

Por último, debemos desplegar la aplicación, para ello simplemente debemos acceder por terminal a la carpeta de la misma, que es “subasta_dapp” e introducir el comando “npm start”. Tras esto, la aplicación estará lista para usarse y es accesible a través de la URL “localhost:3000”.

7 Análisis de los riesgos de seguridad (e.g. SRM)

En este apartado, se pretende identificar y abordar los riesgos potenciales que podrían afectar a la integridad del proyecto de la casa de subasta de obras de arte, algunos de los cuales se muestran a continuación:

- Una de las áreas críticas identificadas se centra en la necesidad de fortalecer la seguridad en el manejo del address del Smart Contract (tanto de la subasta como de los activos). Se deben implementar medidas de seguridad adicionales para salvaguardar y proteger el hash del contrato, evitando su manipulación o acceso no autorizado.
- Otra vulnerabilidad identificada se relaciona con la interacción del usuario en la página. La falta de un proceso de inicio de sesión para los usuarios puede representar un riesgo, ya que esto podría facilitar el acceso no autorizado a las cuentas de wallet de Metamask de los usuarios. Se debe desarrollar un sistema seguro de autenticación previo al acceso a la plataforma que pueda mitigar este riesgo.
- Se reconoce la carencia de un registro de autenticidad para las obras de arte. Esta falta de registro compromete la verificación de la autenticidad de las obras, lo que podría impactar negativamente en la confianza de los usuarios y en la integridad de las transacciones.
- Un usuario podría iniciar la subasta de una obra de arte que en realidad no exista o no sea de su propiedad. Esto quedaría solucionado si se implementase un método de validación de las obras de arte que comprobase que existen y que pertenecen al usuario que quiere iniciar la subasta y que proporcione garantías de autenticidad (implementando controles basados en blockchain para verificar y validar la autenticidad de las obras de arte).
- En el nodo IPFS en el que guardamos la información (o en la base de datos de OrbitDB en caso de poder implementarla) se pueden producir problemas de robos de datos (ya que consideramos una base de datos centralizada sin mecanismos de seguridad ni criptográficos) y también es un problema por resultar ser un “Single Point of Failure” y puede no poder atender un gran número de peticiones. Esto se podría solucionar utilizando una red P2P y utilizando comunicaciones cifradas.
- Por último, un atacante podría engañar a un usuario para obtener acceso a su wallet de Metamask y realizar subastas en su nombre, gastando ether de su monedero. Esto se trata de un caso de ingeniería social y se evita formando a los usuarios para que no puedan ser engañados o que resulte más complicado.

Estas medidas planteadas para solucionar las vulnerabilidades encontradas permitirán fortalecer la seguridad y la confianza en la plataforma de subasta de obras de artes, asegurando la integridad de las transacciones y la protección de los usuarios frente a posibles amenazas de seguridad.

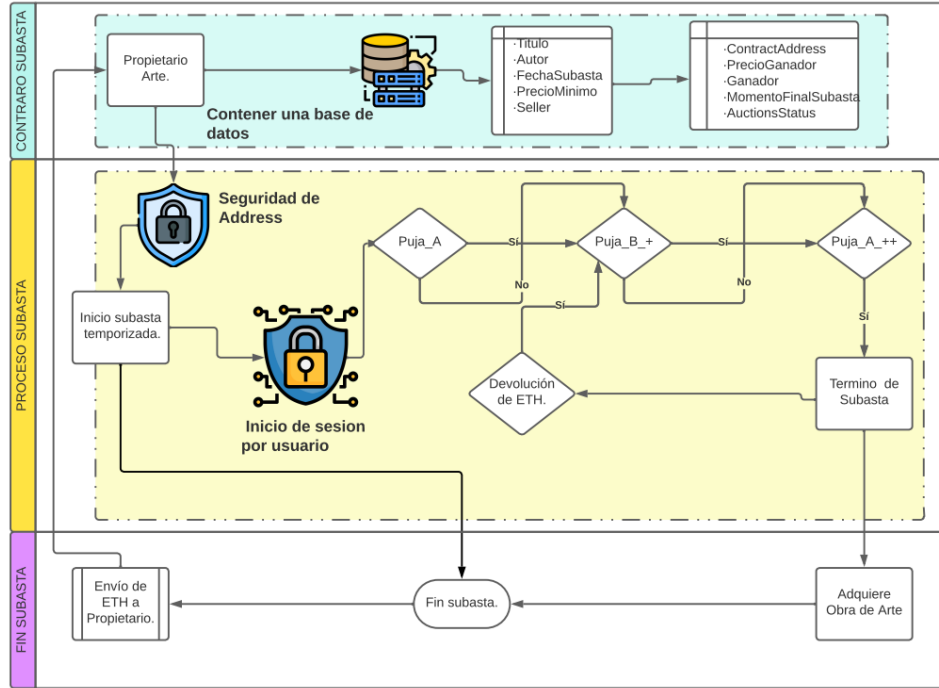


Figure 8: Análisis de mejoras a los riesgos de seguridad.

8 Evaluación y pruebas

A continuación, se van a mostrar los resultados de las pruebas realizadas para probar la aplicación. Estas pruebas tienen como objetivo mostrar que todos los casos de uso detallados en la sección 4.1 están correctamente implementados.

Algunos aspectos a tener en cuenta sobre las pruebas son:

- La simulación de distintos usuarios se ha realizado desde el mismo equipo con distintas cuentas de MetaMask.
- La casa de subastas está representada por la address “0x63Cf53258D1027f288B5940012cBf3C71112987e”.

8.1 Prueba 1: Iniciar subasta

La primera prueba consiste en que un usuario va a iniciar la subasta de un activo que posee (suponemos “Mona Lisa”). Para iniciar la subasta, debe rellenar el formulario de la página principal con todos los datos requeridos, pulsar el botón “Iniciar subasta” y esto provocará que esta subasta aparezca en la página principal tras la validación de la transacción en MetaMask (que supone un coste de gas).

The screenshot shows the top navigation bar of the 'Subasta de Arte' website. On the left is the logo and name. On the right, there is account information: 'Tu dirección de cuenta: 0x63C53258D1027288B5940012c8f3C71112987e', 'Refunda disponibles: 0 wei', 'Ganador:', 'Precio Ganador:', and 'Estado:'. Below this are two buttons: 'Obtener Refunda' and 'Retirar Refunda'.

Below the header is a form titled 'INICIAR UNA SUBASTA'. The form contains the following fields:

- Título de la Obra:** A text input field containing 'Mona Lisa'.
- Autor de la Obra:** A text input field containing 'Da Vinci'.
- Precio Inicial:** A text input field containing '20'.
- Address del activo:** A text input field containing a long hexadecimal string: '0x494489c6c38f142a3627e0d8548230d0da7683c'.

At the bottom of the form is a blue button labeled 'Iniciar Subasta'.

Figure 9: Formulario rellenado con todos los datos necesarios

The screenshot shows a modal window titled 'Subasta iniciada'. It displays the following information:

- Título:** Mona Lisa
- Autor:** Da Vinci
- Precio Mínimo:** 20
- Fecha Inicio Subasta:** Thu Dec 21 2023 06:10:50 GMT-0500 (Eastern Standard Time)

Below the information are two blue buttons: 'Obtener Información Subasta' and 'Resetear Información Subasta'.

Further down is a section for bidding:

- Cantidad Puja:** A text input field containing '0'.
- Pujar:** A blue button.

At the bottom is a blue button labeled 'Finalizar Subasta'.

Figure 10: Subasta iniciada

8.2 Prueba 2: Pujar por la obra de arte

En esta prueba se va a actuar con dos address distintas para simular dos usuarios que pujan por una obra de arte. Como el precio mínimo de la obra es 20 weis, el primero pujará 25 y el segundo 30. Para hacer esto, se debe indicar la cantidad que se quiere pujar y pulsar el botón “Pujar”.

This screenshot is similar to Figure 10, showing the 'Subasta iniciada' modal window. The information displayed is the same as in Figure 10:

- Título:** Mona Lisa
- Autor:** Da Vinci
- Precio Mínimo:** 20
- Fecha Inicio Subasta:** Thu Dec 21 2023 06:12:38 GMT-0500 (Eastern Standard Time)

The bidding section now shows:

- Cantidad Puja:** A text input field containing '30'.
- Pujar:** A blue button.

The 'Finalizar Subasta' button remains at the bottom.

Figure 11: Pujar por una obra de arte

En relación con esta prueba, se ha intentado realizar una puja inferior al precio mínimo de la obra y se

ha obtenido el siguiente mensaje de error:

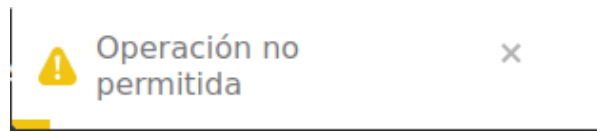


Figure 12: Error en la aplicación

Este mensaje de error aparecerá cada vez que haya un fallo en la ejecución de una función del smart contract porque algún “require” no se esté cumpliendo.

8.3 Prueba 3: Finalizar subasta y obtener información

Tras acabar el tiempo de subasta establecido por la casa de subastas, esta debe finalizar la subasta pulsando en el botón asignado para ello. Tras esto, cualquier usuario puede pulsar en el botón “Obtener Información Subasta” y podrá ver el ganador de la subasta, el precio pujado por el ganador y el estado de la subasta (esta información también se puede ver sin que la subasta haya finalizado).

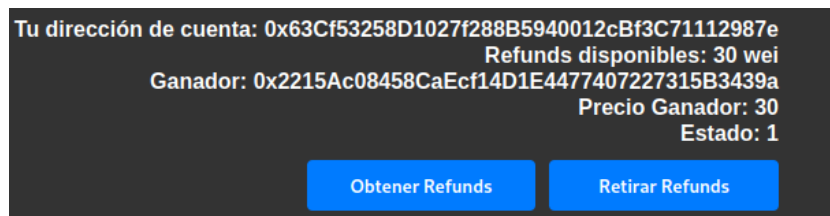


Figure 13: Información de una subasta finalizada

además, también se habrá modificado el owner del activo subastado. Si accedemos en Remix IDE al contrato de este activo y llamamos a la función “owner()”, podremos ver que el ganador de la subasta es el nuevo owner del activo.

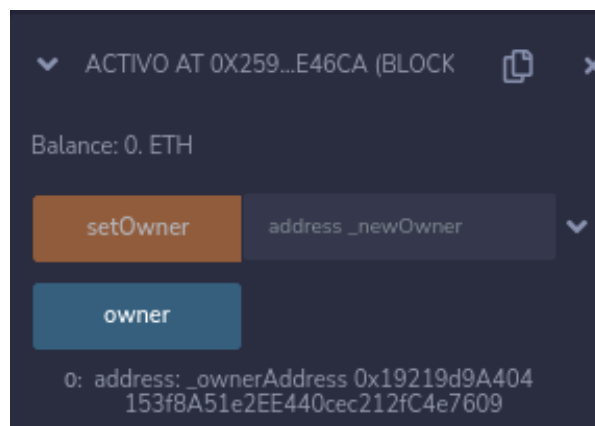
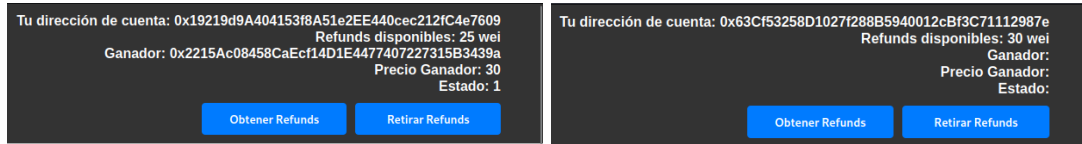


Figure 14: Nuevo owner del activo subastado

8.4 Prueba 4: Obtener refunds y retirarlos

Una vez finalizada la subasta, tanto el propietario original de la obra como el usuario que ha perdido pueden recuperar sus ingresos (30 weis) y sus apuestas (25 weis) respectivamente. En primer lugar, pulsan el botón “Obtener refunds” y pueden ver la cantidad almacenada para ellos en el contrato:

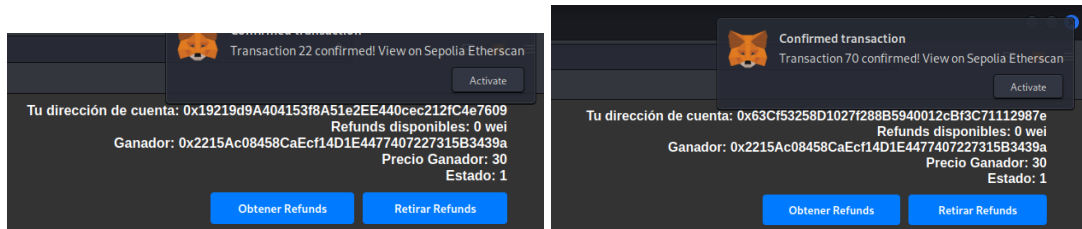


(a) Obtener refunds del perdedor

(b) Obtener refunds del propietario

Figure 15: Obtener refunds de los usuarios

Cuando lo deseen, estos usuarios pueden retirar esos refunds, los cuales se añadirán a sus Wallets de MetaMask. Así, podrán seguir pujando o retirarse si así lo desean.



(a) Retirar refunds del perdedor

(b) Retirar refunds del propietario

Figure 16: Retirar refunds de los usuarios

8.5 Prueba 5: Añadir más subastas e IPFS

Cualquier usuario que lo desee puede iniciar una subasta y todas serán visibles en la página principal de la aplicación tal y como se ve en la imagen siguiente:

Título: Mona Lisa
 Autor: Da Vinci
 Precio Mínimo: 20
 Fecha Inicio Subasta: Thu Dec 21 2023 06:39:30 GMT-0500 (Eastern Standard Time)

Obtener Información Subasta

Resetear Información Subasta

Cantidad Puja:

0

Pujar

Finalizar Subasta

Título: Las Meninas
 Autor: Velazquez
 Precio Mínimo: 30
 Fecha Inicio Subasta: Thu Dec 21 2023 06:39:30 GMT-0500 (Eastern Standard Time)

Obtener Información Subasta

Resetear Información Subasta

Cantidad Puja:

35

Pujar

Finalizar Subasta

Figure 17: Varias subastas desplegadas

Además, si accedemos a la sección “Files” del nodo IPFS podemos ver que se ha registrado la información de estas subastas activas (esta información se registra al iniciar la subasta).




Files		225B FILES	5MIB ALL BLOCKS	+ Import
Name ↑	Pin Status	Size		
 QmcknHmCvLwoY1USnRFQg6HVKA61cJwvYyYsUJoEjL6YS QmcknHmCvLwoY1USnRFQg6HVKA61cJwvYyYsUJoEjL6YS		106 B	...	
 QmRUyWUVX4AWMdMGubQ8ZcuX9yRQ5JStHezpAFDTCzn QmRUyWUVX4AWMdMGubQ8ZcuX9yRQ5JStHezpAFDTCzn		123 B	...	

Figure 18: Archivos guardados en IPFS

Files / Qmck...L6YS	225B FILES	5MIB ALL BLOCKS	... More
Nombre: Las Meninas Activo: 0x59e1eef71550de47f590e60c7872b83cc4dAcB72 Autor: Velazquez Precio inicial: 30			

Figure 19: Información guardada en IPFS

9 Aspectos Legales

Los aspectos legales sobre subastas de obras de arte no están concretamente definidos en España, nos encontramos así con diversas definiciones en cuanto a aspectos legales de contratos inteligentes y subastas.

En la plataforma Agencia Estatal de Boletín Oficial del Estado se describe que las subastas electrónicas están reguladas, por el comercio electrónico Ley 34/2002 y se establece la regulación en ciertos aspectos al momento de ejecutar estas subastas electrónicas, como en contratos del sector público, bienes inmuebles y muebles entre otros.

Si bien las subastas NTF han aumentado exponencialmente y sus sitios son seguros para la ejecución de transacciones inmediatas, aún existe un vacío legislativo en cuanto al consumidor, debido a que alguna filtración de datos o la autenticidad de dicho usuario pueden llevar al robo de información al consumidor que puede afectar sus activos. Por otra parte, encontramos aspectos legales a nivel mundial para los contratos inteligentes por parte de Estados Unidos y de el Código Comercial Uniforme (UCC). Esto significa que, en general, los contratos electrónicos serán legalmente exigibles. La siguiente legislación puede ayudar a determinar si esto se aplica a contratos inteligentes específicos.

El modelo regulatorio estadounidense de estar “abierto a los negocios” se traduce bien en el ámbito de los contratos inteligentes. En otras jurisdicciones, como Europa, los contratos inteligentes también están gozando de un amplio reconocimiento. Según la ley de firma electrónica de la UE (eIDAS), se reconocen tres niveles de firmas electrónicas, cada uno con diferentes niveles de reconocimiento.

Si bien la mayoría de los estados y naciones todavía están en el proceso de determinar cómo harán cumplir de buena manera los contratos inteligentes, para todas las transacciones las empresas deberían adoptar un enfoque seguro y tomar las medidas preventivas, al menos hasta que una legislación sea más clara y concreta.

10 Manual de usuario

10.1 Estructura del proyecto

El proyecto cuenta con el sitio web construido en Web3.0 y Java Script, el cual es la plataforma web donde el usuario interactúa con la información de las obras de artes, la puja, tiempos y la finalización de la subasta, como se muestra en la figura 20. Para desplegar la aplicación, se necesita desplegar un nodo IPFS en la máquina de Kali en la dirección “localhost:5001”, desplegar en Remix IDE los contratos de activos y de subasta y desplegar la aplicación, instalando las librerías “npm install ethers@5.7.0 kubo-rpc-client buffer”, “npm install react-toastify”, “npm install web3”.

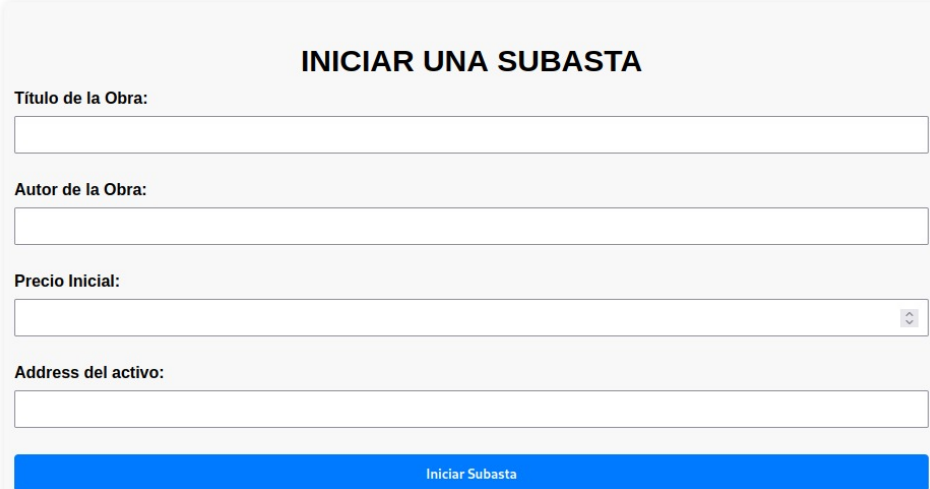
A continuación se muestra un manual de usuario para explorar cada botón y apartado del sitio web.



Figure 20: Plataforma web de Subasta de Obras de Artes.

Iniciar una Subasta: Está apartado central, es dónde se completa la información de la obra de arte para iniciar la subasta, como se indica en la figura 21. El usuario debe ingresar los datos en los campos: *Título de la Obra*, *Autor*, *Precio Inicial* y *Address*. Una vez rellenados los datos, el usuario debe pulsar

en el botón **iniciar subasta**, una vez ejecutada esa acción, se iniciará la subasta de esa obra de arte descrita anteriormente.



INICIAR UNA SUBASTA

Título de la Obra:

Autor de la Obra:

Precio Inicial:

Address del activo:

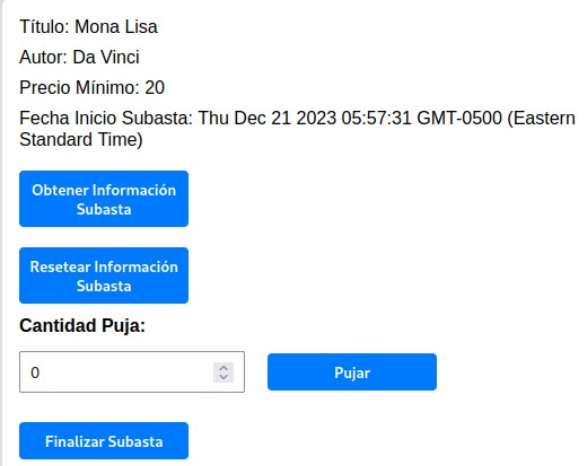
Iniciar Subasta

Figure 21: Apartado Iniciar una Subasta.

Subasta: En este apartado se despliega la información de *Iniciar una Subasta* y cuenta con unos botones interactivos, como lo muestra la figura 22. El botón **Obtener Información Subasta** entrega información sobre el estado de la subasta, el ganador y su puja en un momento determinado.

El botón **Resetear Información Subasta** reinicia la información obtenida anteriormente.

En el apartado *Cantidad de Puja* se ingresa la cantidad a pujar por la obra y se ejecuta la puja con el botón **Pujar**. Un usuario podrá pujar las veces que quiera siempre y cuando cuente con el saldo suficiente. El último botón es el **Finalizar Subasta** que finaliza la subasta en curso.



Título: Mona Lisa
Autor: Da Vinci
Precio Mínimo: 20
Fecha Inicio Subasta: Thu Dec 21 2023 05:57:31 GMT-0500 (Eastern Standard Time)

Obtener Información Subasta

Resetear Información Subasta

Cantidad Puja:

Pujar

Finalizar Subasta

Figure 22: Apartado Subasta.

Encabezado de información: En la parte superior derecha del sitio web del proyecto, se muestra información del estado de la subasta de la obra de arte deseada, como lo indica la figura 24. Además, existen dos botones, el primero, **Obtener Refunds**, que al ejecutarlo muestra los refunds disponibles para ese usuario y el segundo, **Retirar Refunds**, que al ejecutarlo permite al usuario retirar los refunds disponibles.

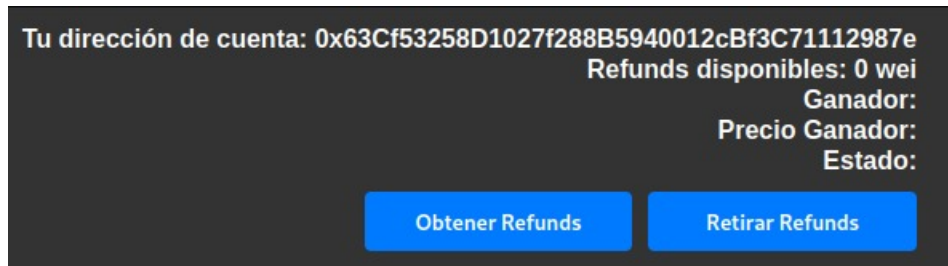


Figure 23: Apartado Información de Subasta.

Con esta información, el usuario podrá interactuar con nuestra aplicación.

11 Conclusiones

El desarrollo y análisis de nuestro proyecto que es referente a la casa de subasta de obras de arte basada en blockchain nos han llevado a identificar aspectos clave en la implementación de tecnologías como los Smart Contracts y el uso creciente de las criptomonedas en el ámbito de las subastas. A medida que se iba investigado sobre las casa de subastas en entornos blockchaing, hemos observado un notable crecimiento en la adopción de monedas criptográficas como método de pago en subastas, lo que subraya la tendencia hacia la descentralización y la innovación en este sector. Esta transición hacia lo digital impulsa el cambio en la dinámica tradicional de las casas de subastas.

La seguridad en estas transacciones se ha convertido en un aspecto crítico; verificar la autenticidad de las obras de arte y proteger los tokens criptográficos se vuelve primordial. La garantía de integridad en el intercambio de activos se convierte en un factor determinante en la adopción generalizada de esta tecnología.

En resumen, el proyecto de la casa de subasta en blockchain ha demostrado el potencial y los desafíos que rodean a la aplicación de tecnologías emergentes en un sector tradicional. El camino hacia la descentralización y la innovación en las subastas de obras de arte implica un equilibrio entre la seguridad, la autenticidad y la eficiencia en la implementación de estas soluciones.

12 Líneas futuras

En cuanto a las líneas futuras del proyecto se consideran las posibles mejoras y expansiones para el sistema de la casa de subasta en blockchain, se vislumbra un camino hacia el perfeccionamiento de la interfaz de usuario y la automatización de datos. Además, sería muy positivo adicionar bases de datos de galerías de obras de artes (OrbitDB) para que el usuario tenga una mejor experiencia al momento. Otro punto importante a mejorar es la seguridad de la aplicación, ya que se deben solucionar todos los puntos planteados ne la sección 7 implementando las contramedidas planteadas en la misma.

Ota línea a investigar sería mejorar la aplicación para que se puedan realizar distintos tipos de subastas, tanto ascendentes (las implementadas), como descendentes, en las que se parte de una cantidad y esta se va reduciendo progresivamente. Estas líneas futuras podrían impulsar aún más la evolución de las casas de subasta en blockchain, permitiendo adaptarse a las demandas cambiantes del mercado y mantener una posición competitiva en un entorno en constante transformación.

13 Lecciones aprendidas

Entre las lecciones aprendidas con este proyecto encontramos:

- La aplicabilidad de las blockchains a situaciones reales como es este de una subasta.
- La importancia de investigar sobre nuevas tecnologías para ampliar conocimientos y entender su funcionamiento.
- La necesidad de investigar en diferentes fuentes para solucionar problemas encontrados durante el desarrollo del proyecto.

- La dificultad de implementar ciertos aspectos en Solidity, como el hecho de acceder a información de structs para modificarlos o tener un array en el struct.
- La utilidad de IPFS, tanto para almacenar información en el propio nodo como para implementar OrbitDB sobre dicho nodo, lo cual proporciona mayor seguridad y eficiencia.
- La sencillez y utilidad de desarrollar aplicaciones descentralizadas utilizando React y Node.js, ya que resultan ser herramientas muy útiles que combinadas con Web3.js permiten relacionar aplicaciones con blockchains y contratos inteligentes.

13.1 Incidencias

A continuación se detallan las incidencias y los problemas encontrados durante el desarrollo del proyecto:

- En primer lugar, se ha intentado desplegar una base de datos de OrbitDb sobre el nodo IPFS con la intención de almacenar en ella la información sobre las obras de arte. El problema es que ha resultado imposible debido a la cantidad de problemas que daba. En el código se ha implementado para ello de la siguiente manera:

```

1  /// Creacion de base de datos
2  /// Iniciar conexión a IPFS
3  const ipfs = await create('/ip4/0.0.0.0/tcp/5001');
4  setIpfs(ipfs)
5  // Iniciar instancia de OrbitDB
6  const orbitdb = await createInstance(ipfs);
7  setOrbitDB(orbitdb);
8
9  /// Almacenar información en OrbitDB
10 const almacenarInformacionEnOrbitDB = async (obraID, titulo, autor, address,
11 precioInicial) => {
12   // Obtener una referencia a la base de datos
13   const db = await orbitDB.open('subasta-db');
14
15   // Almacenar la información en la base de datos
16   const entry = {
17     obraID: obraID.toString(),
18     titulo,
19     autor,
20     address,
21     precioInicial,
22   };
23
24   await db.put(entry);
25 };
26
27
28 /// Cargar información de OrbitDB
29 const getParticipatingArtworks = async (participant) => {
30   // Obtener información de la base de datos en OrbitDB
31   const db = await orbitDB.open('subasta-db');
32
33   // Obtener todas las entradas de la base de datos
34   const artworks = await db.all();
35   setSubastas(artworks);
36 };
37

```

Este código parece correcto pero al desplegarlo, en este caso saltaba un error de que “webpack” no estaba siendo detectado y al instalarlo con npm, el mensaje de error cambiaba a que “node:” no era detectado. Se probaron con distintas librerías para importar IPFS: “ipfs”, “ipfs-core”, “kubo-rpc-client”, y para OrbitDB: “orbit-db”, “@orbitdb/core”. Pero la conclusión es que con todas las pruebas posibles, se ha obtenido algún mensaje de error, bien por librerías o por versiones. Un ejemplo de error obtenido se muestra a continuación:

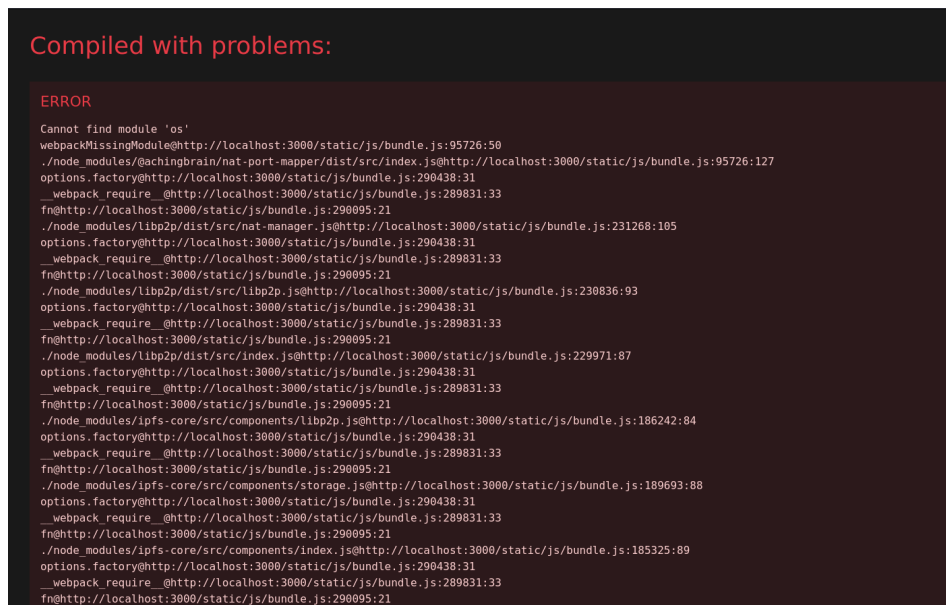


Figure 24: Error al intentar desplegar OrbitDB

- En la implementación del contrato también se han encontrado problemas, por ejemplo, la idea principal era guardar un array de Apuestas en el struct de las obras de arte, pero esto es algo que Solidity no contempla.
- Se pretendía mostrar la información de la subasta en la misma ventana que la información de la obra pero, como esta información se tuvo que guardar en mappings, se tuvo que mostrar en el header de la página y añadir un botón para resetear esta información con el fin de que no haya confusiones para el usuario y pueda tener su header “limpio”.

14 Referencias

Plataforma Wallet Ethereum:

- <https://metamask.io>.

Primera Subasta de por la casa de subasta Christie's:

- <https://www.christies.com/>
- <https://rialta.org/primera-obra-de-arte-digital-subastada-christies-record/>
- <https://www.christies.com/en/stories/monumental-collage-by-beeple-is-first-purely-digital-artwork-nft-to-come-to-auction-0463a2c0f3174b17997fba8a1fe4c865?lid=1>
- <https://www.christies.com/departments/Post-War-Contemporary-Art-74-1.aspx?pagesection=overview#overview>
- <https://onlineonly.christies.com/s/beeple-first-5000-days/lots/2020>
- <https://www.lavanguardia.com/vida/20210311/6359005/primera-obra-nft-alcanza-record-christies-arte-digital-jpg.html>

Auction Smart Contracts: A Deep Dive

- <https://medium.com/coinmonks/auction-smart-contracts-a-deep-dive-ccef76a2c802>
- <https://github.com/dob/auctionhouse>

Proyecto AuctionHouse:

- <https://petkanics.medium.com/introducing-auctionhouse-an-ethereum-dapp-for-auctioning-on-chain-goods-c91244bde469>
- <http://auctionhouse.dappbench.com/>

Casa de subastas de inmuebles:

- <https://spanish-marketing.medium.com/instituto-coinex-la-primera-subasta-nft-de-bienes-ra%C3%ADces-del-mundo-c%C3%B3mo-propy-logra-combinar-ab801f0c8848>
- <https://propy.com/browse/crypto-for-real-estate/>

Casa de subastas de Videojuegos:

- <https://academy.trubit.com/game-finance-gamefi>
- <https://about.gamefi.org/>

Nueva Casa de subasta Christie's Web3.0:

- <https://nft.christies.com/>

Aspectos Legales

- https://www.boe.es/biblioteca_juridica/codigos/abrir_pdf.php?fich=162_Codigo_de_Subastas_Electronicas.pdf
- <https://harris-sliwoski.com/blog/are-smart-contracts-legal-contracts/>