
# MASTER EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE MASTER

NFTs. Estudio tecnológico, de negocio y construcción de un *Marketplace* para su intercambio.

Autor: Iñigo Sagredo Ruiz

Director: David Contreras Bárcena

Madrid

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

**NFTs. Estudio tecnológico, de negocio y construcción de un *Marketplace* para su intercambio** en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2021/22 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.: Iñigo Sagredo Ruiz Fecha: 25/08/2022

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: David Contreras Bárcena Fecha: 25/08/2022

# Agradecimientos

En primer lugar, agradecer a mis padres la posibilidad de haber recibido una educación excelente, culminando en la realización de este trabajo. Su dirección a lo largo no solo del master sino también de toda mi educación ha hecho posible la realización de un TFM que simboliza el final de una educación de mucha exigencia.

A mi director, David Contreras Bárcena, por su comprensión en los momentos más complicados del último año y por su ayuda cuando más era necesario, además de por la idea que ha inspirado este trabajo final.

También me gustaría agradecer a Mauricio Muñoz López, amigo y alumno del master de Ingeniería de Telecomunicaciones, su ayuda gracias a su profunda experiencia con *Blockchain* . Sus explicaciones conceptuales sobre *Blockchain* han enriquecido la rigurosidad de este trabajo.

Una vez más, gracias a todos .

# NFTS. ESTUDIO TECNOLÓGICO, DE NEGOCIO Y CONSTRUCCIÓN DE UN *MARKETPLACE* PARA SU INTERCAMBIO.

**Autor:** Sagredo Ruiz, Iñigo
**Director**: Contreras Bárcena, David
**Entidad Colaboradora**: ICAI – Universidad Pontificia Comillas.

## RESUMEN DEL PROYECTO

El presente trabajo de fin de master se centra en la creacion de un *Marketplace* para el intercambio de NFTs. Para ello, se comienza dando un bagaje teórico sobre los NFTs, se hace uso del *Marketplace OpenSea*, se crea un *Marketplace* mediante *Smart Contracts* y angular y finalmente se realiza un estudio sobre las tendencias futuras de los criptoactivos.

**Palabras clave**: NFTs, *Marketplace*, PoC, criptoactivos

### 1. Introducción y definición del proyecto

El fenómeno de los NFTs (Non-Fungible *Token*s) ha sido, durante el 2021, uno de los fenómenos con mayor popularidad del año. Entre el año 2020 y el año 2021, el intercambio de estos cripto-activos ha aumentado en 21.000% llegando a un tamaño de mercado de 17.000 Millones de dólares [1].

Es por esto, que la comunidad inversora, en particular los *Hedge Funds*, están decidiendo invertir en este nuevo activo [2]. Pero ¿Qué hay detrás de los NFTs? ¿Cuál es la realidad tecnológica detrás de los mismos? ¿Y la componente financiera?

A estas preguntas tratará de dar respuesta esta trabajo. Para ello, se comenzará con un análisis de la cuestión y una agregación teórica de los elementos imprescindibles para comprender en profundidad la realidad de los NFTs. A continuación, se realizará un caso sencillo de *minting* de un NFT en la plataforma *OpenSea* con la intención de realizar un primer acercamiento practico a la cuestión. Finalmente, se construirá un *Marketplace* completo mediante solidity, lo que implicará la escritura de dos contratos: El contrato de los NFTs y el contrato del *Marketplace*. Además, se construirá una interfaz a través de Angular para poder hacer uso de este *Marketplace* mediante un navegador web.

Para terminar, se realizarán unas conclusiones sobre lo aprendido durante el trabajo y se dará una visión de futuro para estos cripto-activos, además de una perspectiva personal sobre la inversión en los mismos.

## 2. Objetivo del proyecto

Este proyecto tiene 4 objetivos diferenciados.

1. Comprender el contexto NFT: Se comenzará contando los pilares tecnológicos que deben ser comprendidos para poder entender el concepto de NFT. Para no caer en la repetición, los conceptos que hayan sido tratados durante el TFG serán resumidos o directamente omitidos siendo estos referenciados en caso de ser necesarios. Se comenzará entendiendo *Blockchain*, sus problemas de escalabilidad, la aparición de las finanzas descentralizadas, los *token*s y la aparición de los *token*s no fungibles (NFTs); además de realizar una explicación detallada de los mismos desde una perspectiva teórica.

2. Realizar una prueba practica: Una vez comprendido el contexto de los NFTs, se pasará a realizar una prueba práctica. Para ello, se cargará a *OpenSea* un NFT artístico y se estudiaran las posibilidades que ofrece esta plataforma para el intercambio de NFTs.

3. Crear NFTs y un *Marketplace*: La parte más relevante del proyecto se centra en este punto. Una vez hecho el análisis y la pequeña prueba de concepto, se procederá a programar mediante solidity un contrato NFT y un contrato *Marketplace*, además de una interfaz para poder visualizar los contratos creados e interactuar con ellos mediante Metamask. Queda pendiente de analizar los NFTs que serán venidos de esta manera, ya que debe haber una justificación para realizar la venta a través de este medio en vez de mediante OpenSea.

4. Presentar una teoría del valor de los criptoactivos: Una vez completado el trabajo, se procederá a realizar un análisis del futuro de este cripto-activo y cuáles son sus posibilidades y riesgos. Este apartado es de especial interés personal para el autor por lo que se desarrollará de manera académicamente rigurosa, sin perjuicio de que el autor se permita dar sus propias opiniones.

## 3. Resultados

Los resultados de este proyecto han sido, por un lado el aprendizaje más detallado de cuestiones financieras y tecnologías de los NFTs, además de la realización de un *Marketplace* mediante contratos. El esquema de carpetas finales obtenido tras el desarrollo del *Marketplace* es el siguiente:

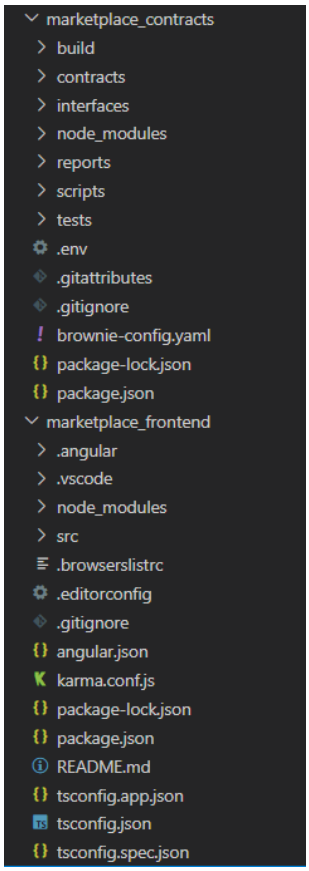


*Ilustración 1. Estructura final de los ficheros para la aplicación Marketplace*

## 4. Conclusiones

En resumen, con la realización de este proyecto, se ha comprendido en detalle las tecnologías NFT, se ha analizado las diferencias entre hacer uso de un *Marketplace* existente frente a uno propio, y se ha hecho un análisis riguroso financiero sobre los criptoactivos que se espera que en unos años demuestre la tesis planteada.

Mencionar también, que se espera que la temática elegida para el *Marketplace* pueda ser empleada para el desarrollo de un proyecto a fundado por un amigo del autor; HabitaXR

## 5. Referencias

[1] Vigna, P. (2022, May 3). NFT sales are flatlining. The Wall Street Journal. Retrieved May 21, 2022, from https://www.wsj.com/articles/nft-sales-are-flatlining-11651552616

[2] Kessler, S. (2022, March 2). Hedge fund giant Alan Howard backs $7.5M round for 'financial nfts' project. CoinDesk Latest Headlines RSS. Retrieved May 21, 2022, from https://www.coindesk.com/business/2022/03/02/hedge-fund-giant-alan-howard-backs-75m-round-for-financial-nfts-project/

# NFTS. TECHNOLOGICAL, BUSINESS STUDY AND CONSTRUCTION OF A *MARKETPLACE* FOR THEIR EXCHANGE.

**Author:** Sagredo Ruiz, Iñigo
Supervisor: Contreras Bárcena, David
Collaborating Entity: ICAI – Universidad Pontificia Comillas.

## ABSTRACT

This master's thesis focuses on the creation of a *Marketplace* for the exchange of NFTs. To do so, it is started by giving a theoretical background on NFTs, trying out OpenSea *Marketplace*, creating a *Marketplace* using *Smart Contract*s and Angular and finally carrying out a study on the future trends of crypto assets.

Keywords: NFTs, *Marketplace*, PoC, crypto assets

### 1. Introduction and project definition

The NFTs (Non-Fungible *Token*s) phenomenon has been, during 2021, one of the most popular phenomena of the year. Between 2020 and 2021, the exchange of these crypto assets has increased by 21,000% reaching a market size of $17 Billion [1].

This is why the investment community, in particular the Hedge Funds, are deciding to invest in this new asset [2]. However, what is behind the NFTs? What is the technological reality behind them? And what about the financial component?

This thesis will try to answer these questions. To do so, it will begin with an analysis of the issue and a theoretical aggregation of the essential elements to understand in depth the reality of NFTs. Then, a simple case of minting of an NFT in the OpenSea platform will be carried out with the intention of making a first practical approach to the task at hand. Finally, a complete *Marketplace* will be built using solidity, which will involve the writing of two *Contract*s: The NFTs *Contract* and the *Marketplace Contract*. In addition, an interface will be built through Angular to be able to use this *Marketplace* through a web browser.

Finally, conclusions will be drawn on what has been learned during the work and a vision for the future of these crypto assets, as well as a personal perspective on investing in them.

## 2. Project Objectives

This project has 4 distinct objectives.

1. Understanding the NFT context: Starting out by counting the technological pillars that must be understood in order to understand the concept of NFT. To avoid repeating the concepts that have been covered during the bachelor's thesis, basic concepts will be summarized or directly omitted being these referenced if necessary. The aim will be to understand *Blockchain*, its scalability problems, the emergence of *Decentralized Finance*, *token*s, and the emergence of non-fungible *token*s (NFTs); in addition to a detailed explanation of them from a theoretical perspective.

2. Carrying out a practical proof of concept: Once the context of NFTs is understood, a practical test will be carried out. For this purpose, an artistic NFT will be uploaded to OpenSea, and the possibilities offered by this platform for the exchange of NFTs will be studied.

3. Creating NFTs and a *Marketplace*: The most relevant part of the project is focused on this point. Once the analysis and the small proof of concept are done, an NFT *Contract* will be programmed as well as a *Marketplace Contract* through solidity, an interface to visualize the created *Contract*s and interact with them through Metamask. It remains to analyze the NFTs that will be sold in this way, since there must be a justification for selling through this medium instead of through OpenSea.

4. Present a theory of the value of crypto assets: Once the work is completed, an analysis of the future of crypto assets and what are its possibilities and risks will be done. This section is of special personal interest to the author, so it will be developed in an academically rigorous manner, without prejudice to the author being allowed to give his own opinions.

## 3. Results

The results of this project have been, on the one hand, a more detailed learning of financial issues and technologies of NFTs, and on the other hand, the realization of a *Marketplace* through *Smart Contract*s. The outline of the final portfolios obtained after the development of the *Marketplace* is as follows:

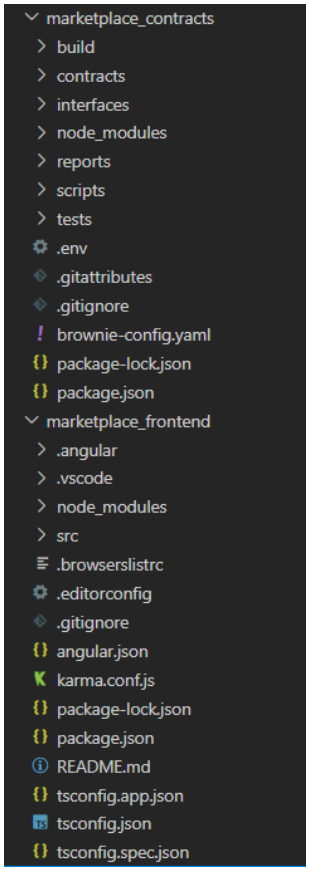


*Ilustración 2. Estructura final de los ficheros para la aplicación Marketplace*

## 4. Conclusions

In summary, with the completion of this project, we have understood in detail the NFT technologies, we have analyzed the differences between using an existing *Marketplace* versus our own, and we have made a rigorous financial analysis on crypto assets that is expected to prove the thesis in a few years.

It is also expected that the theme chosen for the *Marketplace* can be used for the development of a project founded by a friend of the author; HabitaXR.

## 5. References


[1] Vigna, P. (2022, May 3). NFT sales are flatlining. The Wall Street Journal. Retrieved May 21, 2022, from https://www.wsj.com/articles/nft-sales-are-flatlining-11651552616

[2] Kessler, S. (2022, March 2). Hedge fund giant Alan Howard backs $7.5M round for 'financial nfts' project. CoinDesk Latest Headlines RSS. Retrieved May 21, 2022, from https://www.coindesk.com/business/2022/03/02/hedge-fund-giant-alan-howard-backs-75m-round-for-financial-nfts-project/

# *Índice de la memoria*

# *Índice de Ilustraciones*

# Capítulo 1. INTRODUCCIÓN

El presente trabajo tiene como objetivo construir un *Marketplace* para la venta de NFTs. Para ello, se comenzará dando una explicación de los fundamentos de los NFTs para entender tanto los mismos como las posibilidades que estos permiten. Tras ello, se procederá a realizar una prueba de concepto con la subida de un NFT a un *Marketplace* ya existente. Para finalizar, se construirá un *Marketplace* mediante *Smart Contracts* en solidity.

A continuación, se expone por un lado la motivación del proyecto y por otro lado la estructura de este.

## 1.1 MOTIVACIÓN DEL PROYECTO

Este trabajo está motivado por dos ámbitos diferentes y complementarios: el ámbito académico y el personal.

**Académico:** En el ámbito académico, durante el final de 4º de Carrera, se realizó el TFG del grado en Ingeniería en Tecnologías de Telecomunicación. El título del mismo fue "Estudio sobre la cancelación de deuda circular mediante el uso de *Smart Contract*s en *Ethereum*". Este trabajo introdujo al autor a las tecnologías *Blockchain*, y en especial a la red *Ethereum*. Durante el presente trabajo, y en la medida de lo posible, se referenciarán aquellos aspectos que ya hubiesen sido tratados en ese trabajo. Sin embargo, la temática de este trabajo se distancia del anterior desde prácticamente el inicio, por lo que la investigación y aplicaciones prácticas serán independientes.

**Personal:** Desde la realización del TFG mencionado, el interés personal en la materia ha ido aumentando y se ha ido formando en la materia desde entonces. De hecho, el ecosistema ha crecido mucho, llegando a ofrecer un sistema de finanzas descentralizadas (DeFi) muy completo. Actualmente, muchos de los servicios comunes en la economía tradicional como la gestión de activos o la creacion de plataformas de préstamos han sido replicados con éxito

en el ecosistema de *Blockchain*. Finalmente, el fenómeno de las *StableCoins* y los NFTs ha estado muy presente en los medios y se considera que han sido de gran interés y comprender los mecanismos que permiten la existencia de estos activos permite dar una visión holista de la actualidad y noticias candentes hoy en día.

## 1.2 ESTRUCTURA DE PROYECTO

Este proyecto tiene por objetivo, por un lado estudiar los fundamentos alrededor de los NFTs y por otro construir un *Marketplace* mediante el uso de Solidity.

La estructura del proyecto será dividida en cuatro partes:

**Fundamentos de los NFTs:** En este apartado se comenzará dando una explicación sobre *Blockchain* a modo de establecer las bases que fundamentan esta tecnología. Tras eso, se hará un análisis de los cambios que se han producido desde la entrega del TFG hace 2 años. Se tratará en detalle el problema de la escalabilidad que ha hecho que aparezcan redes alternativas a *Ethereum* y como se ha conseguido establecer un sistema de finanzas descentralizado con las implicaciones que ello conlleva. Tras ello, se introducirá el concepto de los NFTs, pudiéndose así entender cuál es su funcionamiento tecnológico y la razón de ser de estos.

***OpenSea and minting*:** Una vez comprendido el concepto de los NFTs, se pasa a realizar una prueba de concepto. Para ello, se hará uso de un *Marketplace* ya existente; OpenSea. Se creará un perfil de autor y se subirán los datos de las obras de arte del autor a la plataforma (proceso denominado *Minting* en inglés y cuyo término se empleará a partir de ahora). Con esto se busca explorar este proceso antes de pasar a la construcción de un *Marketplace*.

***Marketplace*:** En este apartado, se procede a construir un *Marketplace* mediante el uso de solidity. Para ello, habrá que programar, por lo menos, dos *Smart Contracts*; por un lado uno de NFTs y por otro uno que permita el *Marketplace*. Tendrán que poder interactuar entre ellos y con el usuario final. El usuario final hará uso de un navegador web tradicional, por lo que será necesario que tenga un enlace con la red *Blockchain* (debe elegirse que red

*Blockchain* es preferible para esto). Este enlace se conseguirá mediante Metamask. Finalmente, el usuario final tendrá que poder visualizar la información a la que acceda en la red *Blockchain* donde reside la información de los contratos, por lo que una interfaz web deberá ser desarrollada.

Finalmente, hay que mencionar que a lo largo de todo el proyecto, se tendrá muy presente la perspectiva del valor. El valor de los criptoactivos, ya sean criptodivisas, *token*s o NFTs, es un tema fundamental en cualquier análisis sobre los NFTs, y entender en profundidad este aspecto es esencial a la hora de tomar decisiones inteligentes sobre los NFTs.

# Capítulo 2. FUNDAMENTOS DE LOS NFTS

## 2.1 ¿QUÉ ES *BLOCKCHAIN*?

Con la intención de hacer que la componente teórica requerida para entender los NFTs sea lo más holística posible, se comienza con una breve descripción de que es *Blockchain*. En caso de requerirse una comprensión más exhaustiva de la que se va a proporcionar a continuación, se anima a acceder al TFG del autor [1] titulado "Estudio sobre la cancelación de deuda circular mediante el uso de *Smart Contract*s en *Ethereum*.", donde se explica de manera mucho más detallada la tecnología detrás de *Blockchain*.

En esencia, *Blockchain* es una tecnología creada en el año 2008 por una persona o grupo de personas bajo el alias de Satoshi Nakamoto [2]. La tecnología especifica las reglas para la creacion de una cadena de bloques de manera descentralizada con información relevante que desea preservarse de manera inequívoca en el tiempo (por ejemplo, una lista de transacciones o los puntos por los que ha pasado un producto como forma de dotar de confiabilidad al proceso de *Supply Chain*) y verificable en cualquier momento. Además, cada bloque creado incorpora la información de todos los anteriores, de manera que la confiabilidad en la tecnología aumenta con cada bloque adicional.

Cuando Satoshi estableció las reglas que forman la tecnología *Blockchain*, las plasmo sobre el primer **protocolo** *Blockchain* establecido, el Bitcoin. A continuación, estos principios de *Blockchain* han dado lugar a muchos más protocolos como *Ethereum* o Monero, cada uno con sus particularidades y diferencias que tratan de fomentar la participación del mayor número de agentes en los mismos, promoviendo así la descentralización del protocolo.

Adicionalmente, estos protocolos pueden o no tener criptomonedas asociadas. Por ejemplo, el protocolo Bitcoin tiene la criptomoneda Bitcoin asociada al mismo y el protocolo *Ethereum* tiene el Ether como criptomoneda. Estas criptomonedas y su valor es un punto que

se investigara en este trabajo, ya que algunas tienen como utilidad la participación en el protocolo que las posee mientras que otras tienen un valor con una naturaleza más especulativa.

Esta tecnología ha sido tan significativa que ha dado lugar a lo que se ha denominado la tercera generación de internet, o Web 3.0. Esta tercera generación esta precedida por Web 1.0, donde en esencia el contenido ofrecido era paginas estáticas en modo solo lectura, sin mucha interacción con el usuario, abarcando el periodo entre 1985 a 2005 [3]. Tras ello, llego web 2.0, donde la interacción entre la web y el usuario era más interactiva, lo que permitió la personalización del contenido. Web 3.0, a pesar de no contar con una definición oficial, consistiría en la próxima evolución de la web en la que la tecnología *Blockchain* juega un papel esencial. Iniciativas como *Minds* [4], que es una red social descentralizada, están basadas en la proposición de web 3.0; la descentralización como forma de promover la democracia, la libertad y el dinamismo económico [5].

A lo largo del trabajo de fin de grado que se realizó en 2019, el principal foco a nivel tecnológico fue Bitcoin, dejando a *Ethereum* en un segundo plano a pesar de ser la base de la implementación práctica. En este trabajo, debido a la naturaleza del mismo, el foco será en *Ethereum*, ya que es el protocolo donde se realizaron los primeros *Smart Contracts* según lo que se entiende hoy en día por este terminó *(*más allá de los scripts programáticos que algunos protocolos como Bitcoin ya permitían).

## 2.2 NOVEDADES DEL ECOSISTEMA *BLOCKCHAIN*

### *2.2.1 ADOPCIÓN CRECIENTE*

#### *2.2.1.1 Wallets*

Las carteras son la manera en la que un usuario puede *guard*ar las claves publica privada para poder, en un futuro, interactuar con la *Blockchain* para mandar, recibir y gestionar criptoactivos. Una clave publica permite al usuario recibir transacciones mientras que una

clave privada le permite enviar de manera confiable criptoactivos (puede encontrarse más información sobre este proceso detallado en el caso de Bitcoin en el TFG del autor [1]).

En esencia existen 2 tipos de *wallets:*

1. **Wallets Online**: Son aquellas en las que la información de las claves pública-privada se encuentran en un servidor remoto. Ejemplos de este tipo de wallets son CoinBase o Binance que *guard*an en sus servidores estas claves y por tanto, teóricamente, podrían hacer uso de fondos de usuarios. Este tipo de wallets se oponen parcialmente a uno de los pilares de la filosofía *Blockchain*; la descentralización.

2. **Wallets Offline**: Este tipo de wallets *guard*an la clave pública y privada de manera local. Esto puede conseguirse de 3 maneras:
    a. *Guard*ar las claves en un trozo de papel, que es la manera menos informatizada posible. La principal desventaja de este método es lo tedioso que puede ser el operar, ya que cada vez que se desease interactuar con la red deben introducirse esos datos de manera manual.
    b. *Guard*ar las claves en un dispositivo Hardware sin conexión a internet. Estos dispositivos hardware suelen estar construidos de manera especializada para *guard*ar esta información, por lo que disponen de algoritmos de encriptación muy robustos, *Ledger Live* es un ejemplo de un proveedor de este tipo de carteras.
    c. *Guard*ar las claves en un navegador local. Efectivamente, puede *guard*arse el par clave pública-privada requerida para operar en el propio navegador de un ordenador (por ejemplo, en Google Chrome). El ejemplo más popular de este tipo de Wallet es Metamask, que será empleada en la parte práctica de este trabajo.

Finalmente, hay que mencionar que las *wallets* permiten hacer de intermediario entre un usuario y la complejidad que puedan tener los diferentes protocolos *Blockchain*, lo que ha fomentado muchísimo la adopción de esta tecnología que, de otra manera, podría ser

utilizada únicamente por aquellos con el conocimiento necesario para poder operar en su complejidad programática.

#### *2.2.1.2 Chainlink*

Los *Smart Contract*s son contratos programáticos que permiten ejecutar, mediante lógica, condiciones que se estipulen en los mismos. Por ello, si se desean establecer condiciones para que un *Contract*o se ejecute, estas condiciones deben de ser verificables por la red. Es por ello por lo que las condiciones de estos contratos deben ser observables en la propia *Blockchain*. Por ejemplo, una lógica que afirmase que Alice debe mandar un *token* determinado a Bob si el Hash del próximo bloque es un numero impar es una lógica verificable dentro de la red.

Sin embargo, si se desean hacer contratos más prácticos con dependencias de eventos fuera de la red *Blockchain*, hace falta una interacción con el mundo real. Por ejemplo, si se desease crear una casa de apuestas en *Blockchain*, y se desea poder apostar sobre eventos que se producen fuera de la *Blockchain* (por ejemplo, el resultado de un partido de tenis), es necesario poder recibir la información de fuera de la red. Esto crea un problema importante sobre cómo hacerlo, ya que no es posible verificar que lo que se introduce a la red desde fuera es correcto. Esta es la problemática que trata de atajar Chainlink, que es un protocolo *Blockchain* cuyo *token* es Link.

**¿Cómo lo hace?** Se ha explicado la complejidad de introducir datos del exterior en *Blockchain*, debido a los incentivos que pueden tener los agentes para no ser honestos (por ejemplo, si he apostado a un resultado estaré incentivado a decir que ese ha sido el resultado). Por ello, Chainlink se centra en crear los incentivos adecuados para garantizar la honestidad y confiabilidad de los datos.

Para ello, hace uso de una red de nodos que serán los encargados de actuar como puente entre el exterior y la red de Chainlink. Estos nodos reciben el nombre de oráculos. Pero ¿cómo se incentiva a estos oráculos a ser honestos? Existe una demanda real de poder interactuar con el exterior de la red *Blockchain*, y, si Chainlink permite esto, entonces habrá una disposición a pagar por este servicio. La forma en la que se paga por ello es mediante el

*token* de Chainlink, el Link. Los diferente usuarios que quieran hacer uso del servicio de ChainLink deberán comprar Link para pagar por ello. Esto en torno hace que el valor de Link sea un valor real, y pueda ser empleado por la red para pagar a sus oráculos.

El protocolo ChainLink hace uso de POS (*Proof of Stake*), concepto explicado en el TFG (1). En esencia, los distintos nodos u oráculos bloquean parte de sus Link *token*s hasta que una validación este completa, y si una vez esta se ha llevado a cabo no coincide con lo indicado por el oráculo, entonces este pierde los fondos bloqueados. Sin embargo, si coincide, entonces recibe una retorno por su contribución en forma de Link *token*s. Pero ¿cómo funciona a nivel tecnológico esta validación? Se basa en 2 elementos. Un *Requesting Contract* y un SLA (*Service Level Agreement*) *Contract*.

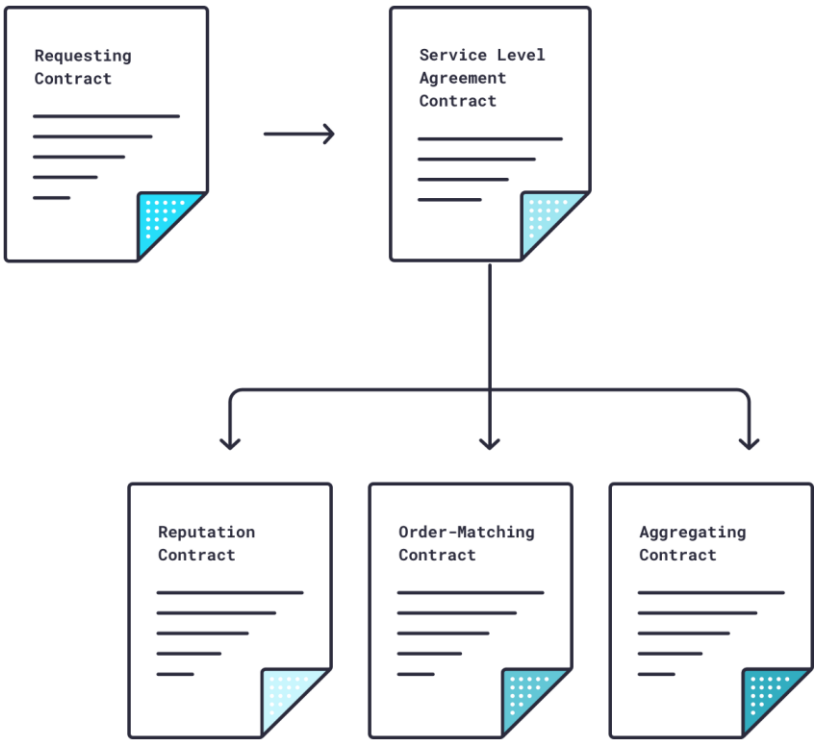


*Ilustración 3. Interaccion entre el Requesting Contract y el SLA contract*

El funcionamiento es el siguiente:

- ***Requesting Contract*:** Un usuario que desea obtener un dato del exterior validado por Chainlink (importante, no se suele buscar el dato en sí, sino que sea un dato validado por la red). Para ello, escribe un *Smart Contract* pidiendo este dato, y especificando si prefiere que el mismo se lo determinen una serie de nodos predefinidos o no. Este contrato es transparente y visible por cualquiera. Tras eso, se pasa a la fase de *Service Level Agreement*.
- **SLA:** Esta fase está compuesta por 3 contratos
  - ***Reputation Contract*:** Este contrato hace uso de varios elementos como la cantidad de LINK que un nodo está dispuesto a bloquear o como su reputación ha sido en el pasado.
  - ***Order-Matching Contract*:** Con las variables anteriores, se eligen un conjunto de nodos. Este contrato envía el *Requesting Contract* al conjunto de nodos que serán los validadores.
  - ***Agregating Contract*:** Los diferentes nodos u oráculos dan los resultados y ChainLink los agrega determinando cual es el resultado correcto y actuando en consecuencia (aquellos nodos que han sido honestos reciben un retorno y aquellos que no pierden la cantidad de LINK que bloquearon)

De esta forma, es posible interactuar con el exterior lo que incrementa muchísimo las aplicaciones y casos de uso de *Blockchain*, ya que las diferentes redes pueden integrar Chainlink para poder interactuar con el exterior, contribuyendo así a una mayor adopción de *Blockchain*.

## 2.3 FINANZAS DESCENTRALIZADAS (DEFI)

DeFi (*Decentralized Finance*) es un concepto que tiene por objetivo trasladar los servicios centralizados tradicionales de la economía Fiat a la economía descentralizada. A lo largo de la historia, una entidad centralizada como el estado o los bancos han sido los responsables

de controlar los sistemas financieros de manera centralizada. Por ejemplo, el banco central europeo (BCE) es el encargado de imprimir moneda.

Las criptomonedas como Bitcoin o *Ethereum* son consideradas dinero descentralizado, ya que su emisión y control no depende de una entidad central. Sin embargo, si se desea realizar un sistema financiero equiparable al sistema financiero centralizado, son necesario muchos mas servicios que un equivalente monetario descentralizado.

Existen 3 pilares fundamentales para construir un entorno de finanzas descentralizadas.

- Criptografía: Permite eliminar la necesidad de confiar en un agente central externo para poder realizar transacciones de manera segura y confiable.
- *Blockchain*: Es el ecosistema sobre el cual se basará la solución descentralizada.
- *Smart Contract*s: Dotan al sistema de confiabilidad mediante líneas de código inmutables, verificables, homogéneas y distribuidas.

Estos tres pilares fundamentales permiten la concepción y el uso de un ecosistema financiero tradicional pero descentralizado, que incluye la posibilidad de prestar dinero, pedir dinero prestado, intercambiar distintos criptoactivos, socializar el riesgo mediante seguros y cualquier tipo de condición programática que pueda ser concebida en un *Smart Contract* y pueda incluir la posibilidad de usar Oráculos (ChainLink visto en el apartado anterior).

A continuación se procede a realizar un análisis de los aspectos mas relevantes de DeFi y su funcionamiento. La idea es tener una idea a alto nivel de todos ellos, para entender que el *Marketplace* que se desarrollará en la parte practica de este trabajo no es mas que otro servicio DeFi.

### *2.3.1 CRIPTOACTIVOS*

Lo primero que requiere cualquier sistema financiero es una divisa. Una divisa sirve tanto como unidad de cuenta, como depósito de valor y como medio de intercambio. En esencia, posibilitan realizar transacciones y son la base sobre la que se construyen el resto de

propuestas del ecosistema financiero descentralizado. Por eso, es importante comprender la diferencia entre criptomonedas, *token*s, stablecoins y *wrapped token*s. Para ello se pasará a explicar por un lado sus definiciones, sus diferencias y como se consiguen de manera programática.

#### *2.3.1.1 Criptomonedas vs tokens*

Antes de entrar en la diferencia entre ambos, mencionar que dentro del apartado de *token*s, no se hará mención a los *token*s no fungibles (NFTs), ya que se reserva un apartado completo para los mismos dada su relevancia en este proyecto.

La principal diferencia entre una criptomoneda y un *token* es que una criptomoneda requiere el uso de una *Blockchain* propia, mientras que un *token* se monta, mediante *Smart Contracts*, sobre una red *Blockchain* existente. Se puede ver como que una criptomoneda otorga el derecho a participar en el ecosistema de la red *Blockchain* a la que dicha criptomoneda esta asociada. Por ejemplo, si se desea crear un *token* en la red *Ethereum*, para la creacion de dicho *token* será necesario hacer uso de *Smart Contract*s (según el estándar ERC20). Para poder interactuar con dicho *Smart Contract* (y para desplegarlo) será necesario hacer uso de Ether, la criptomoneda de *Ethereum*. Por tanto, como se observa, se requiere de la criptomoneda para interactuar con el *token*.

Existen situaciones en las que un *token* tiene mucho éxito y pasa a crearse una criptomoneda del mismo. Esta transferencia se consigue mediante un bridge, en la cual los poseedores de ese *token* pueden intercambiarlo por dicha criptomoneda que, al estar en otro red *Blockchain* diferente, requieren de ese Bridge.

Existen 5 tipos de *token*s a considerar [6]:

1. *Token*s de plataforma: Son *token*s que otorgan equidad. Por ejemplo, el *token UniSwap* otorga equidad sobre el Exchange *Uniswap*.
2. *Token*s de seguridad: Siguen el valor de otro activo como el oro. Se procede a explicar un caso concreto de estos *token*s de seguridad en el apartado sobre stablecoins.

3. *Token*s de transacción: Se emplean en transacciones sirviendo como unidad de cuenta.
4. *Token*s de utilidad: *Token*s que tienen un valor asociado con la propiedad de los mismos
5. *Token*s de gobierno: Otorgan derechos de voto a los acreedores de dichos *token*s.

En el siguiente apartado, se pasa a explicar un caso concreto de *token*s, las *stablecoins*.

#### *2.3.1.2 Stablecoins*

Este apartado ha sido muy mediático recientemente, por lo que se tratará de explicar en detalle que son las *stablecoins* y como mantienen su valor.

En esencia, una *stablecoin* es un *token* que tiene el objetivo de mantener la paridad con otro activo, comúnmente con divisas populares como el euro o el dólar.

En función del respaldo que tengan estas stablecoins, se dividen en dos tipos: Stablecoins colateralizadas y stablecoins algorítmicas.

**_Stablecoins_ Colateralizadas**

El punto mas importante a entender de este tipo de stablecoins es el hecho de que están respaldadas por el activo que pretenden seguir. Para comprender mejor que significa esto, se procede a ejemplificarlo con el caso mediático reciente de la *stablecoin* colateralizadas *Tether*USD.

*Tether* USD es un *token* cuyo objetivo es que se cumpla la siguiente ecuación:

$$1\ Tether\ \text{USD} = 1\ \text{USD}$$

La cantidad total de inversión en dicha *stablecoin* es de unos 80 billones de dólares. En esencia, este *token* no busca que se revalorice ni nada similar, sino que su propuesta de valor es entregar a su poseedor una manera de aparcar la liquidez sin necesidad de salir del sistema cripto.

Esto tiene un valor muy elevado para la comunidad cripto. Supóngase que un inversor de Bitcoin considera que el valor de Bitcoin es muy elevado y que por tanto es momento de vender y esperar a que baje para comprar de nuevo. Este inversor podría realizar la venta de Bitcoins a una entidad central, y a través de una transferencia bancaria recibir su dinero en dólares en su cuenta bancaria, deduciendo las correspondientes comisiones. Esto es un proceso largo y tediosos que podría robarle al inversor la oportunidad de realizar una inversión más ágil si las circunstancias lo propiciasen.

Ahora bien, ya se ha estipulado la utilidad de una stablecoin: permite al inversor cripto aparcar su valor sin tener que salir del ecosistema *Blockchain*. Sin embargo, ¿Cómo mantiene su valor?

Lo primero es indicar que recientemente, el valor de *Tether* USD ha pasado de ser $1 a $0.95. Esto es una bajada importantísima del 5%, que para un activo cuya propuesta de valor se basa en dicha paridad, es alarmante. La causa de esta bajada ha sido el FUD (*Fear Uncertainty and Doubt*) generados tras el colapso de la *stablecoin* algorítmica Terra (que se explicará en la siguiente sección). Pero gracias a los mecanismos colateralizados de *Tether* que se explicarán en esta sección, el valor se recupero rápidamente. Esta grafica muestra dicho pico.

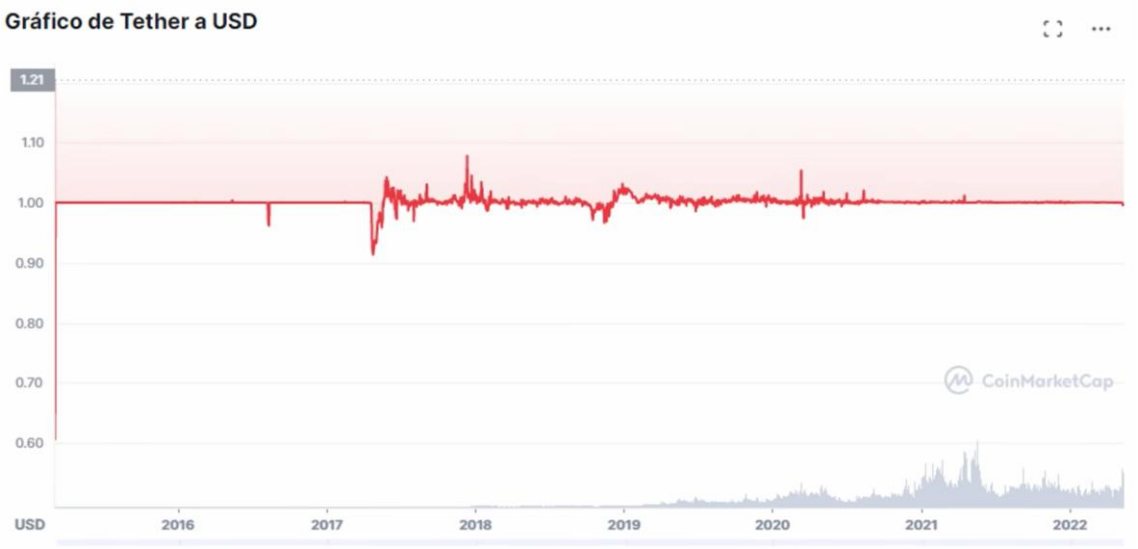


*Ilustración 4. Imagen que muestra la estabilidad de Tether a pesar del FUD*

El mecanismo para mantener el valor es el siguiente. Existen dos escenarios posibles de desviación:

- Desviación al alza: Si una unidad de *Tether* pasa a valer mas de un dólar, la solución para devolver el *token* a la paridad con el dólar es emitir mas unidades de *Tether*. A mayor oferta de *Tether*, con una demanda idéntica, el precio del activo *Tether* caerá, bajándolo de nuevo al valor de un dólar. De esta forma se corrige el precio al alza.
- Desviación a la baja: Este escenario es algo mas complejo. Si el precio baja, se ha de conseguir que el precio relativo del *Tether* respecto al dólar suba, por lo que se han de emplear dólares en comprar *Tether*. Es por ello, que *Tether* cuenta con suficientes activos en dólares para poder hacer dicho proceso cuando sea necesario.

Además, los tenedores de *Tether* con más de $100k que paguen una comisión de 150 dólares [7], podrán convertir cuando lo deseen el *Tether* que elijan por dólares, lo cual es un incentivo adicional para que *Tether* mantenga dicha paridad, y alinea los incentivos de la comunidad con los de *Tether*.

Ahora bien, si es el caso que *Tether* hace uso de un colateral, ¿en que consiste dicho colateral? ¿Qué activos hay para sostener los 80 billones de valor que tiene este *token*? En primer lugar, es importante mencionar que *Tether* requiere de activos pagaderos a muy corto plazo y muy seguros. Ejemplos de estos activos son dólares (el máximo nivel de liquidez), letras del tesoro americano a un máximo de un año, o papel comercial muy corto plazo (deuda corporativa) de compañías muy solventes como Amazon. De esta forma, *Tether* como compañía puede generar un retorno en base a los activos que gestiona, ya que está obteniendo por los mismos el retorno que ofrecen estos instrumentos de deuda a muy corto plazo. Además, en caso de que *Tether* no encontrase forma de colocar en el mercado estas deudas a muy corto plazo ya que su precio bajase repentinamente (es difícil debido a la franja temporal reducida) podría pedir un crédito colateralizado con esos mismos instrumentos de

deuda a un bajo tipo de interés, pudiendo así reflotar el valor de *Tether* si esto fuese necesario.

Al no ser *Tether* una empresa cotizada en bolsa, no se puede conocer de manera confiable la composición exacta de su colateral. Sin embargo, en 2021 indicaron que el 75% de su cartera estaba compuesta por efectivo o equivalente a efectivo, mientras que el 25% de su colateral estaba invertido en deuda a largo plaza (que en condiciones normales, tiene un mayor retorno que la deuda a corto plazo. La composición de su colateral es la siguiente:

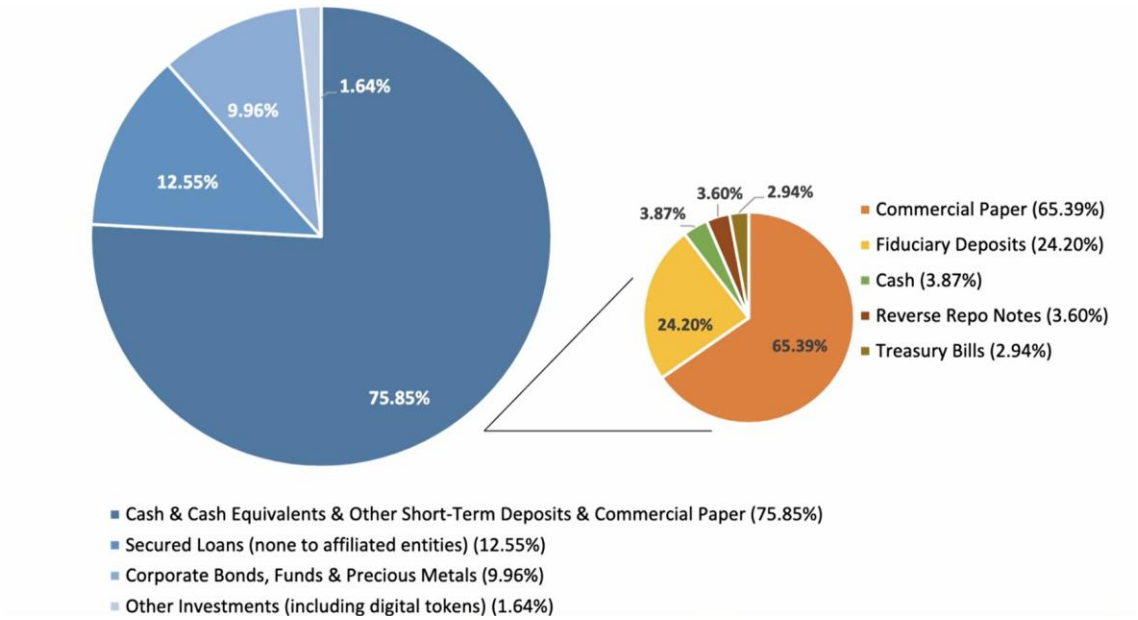


*Ilustración 5. Composición de los colaterales de Tether*

De esta forma, *Tether* garantiza que tiene dólares o su equivalente para corregir cualquier modificación brusca a la baja de *Tether*, mientras que mantiene una parte en instrumentos de deuda a largo plazo más rentables, pero que son mucho mas sensibles a las variaciones de los tipos de interés. Esto supone un *trade-off* muy interesante.

Actualmente, *Tether* indica que la mitad de sus activos se encuentran en deuda pública. De nuevo, al no ser una empresa cotizada no hay ninguna forma de comprobar si esta información es o no veraz, aunque todo apunta a que lo es, ya que han pasado la prueba del

mercado, sobreviviendo al enrome FUD que se produjo tras el colapso de la *stablecoin* Terra, que se mencionará a continuación.

**Stablecoins Algorítmicas**

Estas stablecoins tratan de mantener la paridad con su activo de referencia mediante algoritmos en vez de a través de un colateral. Este tipo de aproximación plantea una serie de problemas en la práctica. Se ejemplificarán los mismos con el caso de Terra-Luna, que tuvo una gran repercusión mediática durante este año 2022.

La idea es exactamente la misma en tanto que se busca que una unidad de *TerraUSD* valga lo mismo que un dólar. Ya se han explicado las ventajas que tiene para la comunidad inversora esta paridad. La forma en la que algorítmicamente Terra trata de mantener la paridad es mediante un sistema muy interesante de incentivos.

El algoritmo se basa en al existencia tanto de Terra USD como de Luna. Luna sirve como un ancla para Terra USD, que mantiene su valor en caso de que se desvié del valor de 1 dólar. Pero ¿Cómo consigue esto en la práctica?

Lo primero es que una unidad de *TerraUSD* puede ser intercambiado por una unidad de Luna. Es posible realizar esa conversión o realizarla a la inversa. Si un usuario desea *TerraUSD*, puede conseguirlo realizando una compra directa a cambio de dólares o realizarla a cambio de Luna. Para que este sistema de incentivos de intercambios funcione, el valor de Luna debe ser un valor positivo y suficiente. Luna es en esencia como las acciones de la red Terra. Terra emite muchas stablecoins, y los beneficios que genera la red se lo llevan los propietarios de Luna, es decir, los accionistas de la red Terra.

Existen dos tipos principales de beneficios como accionista de la red Terra: Por un lado, los beneficios por señoreaje (de la misma forma que hace el BCE, existe un beneficio a la hora de emitir moneda que se obtiene de el valor de la moneda menos los costes de emisión) y por otro lado los ingresos por validación de transacciones. Las validaciones en Terra se hacen

mediante el *staking* de Luna, por lo que por la validación, los inversores en luna reciben comisiones. Esto es como un impuesto Tobin. Por estas razones, Luna tiene un valor positivo y suficiente.

Por tanto, Terra como compañía emite stablecoins, las acciones de esta compañía son Luna y estas acciones tienen un valor positivo y suficiente por las fuentes de ingresos mencionadas. Los accionistas, en compensación, tienen el deber de estabilizar a las stablecoins, en el caso de este ejemplo a *TerraUSD*.

Se procede a explicar las mismas tesituras de desviaciones al alza y a la baja y como las resuelve cada uno.

- Desviación al alza: Si Terra USD sube por encima de un dólar, se emite más *TerraUSD* que se emplea en comprar Luna, aumentando la presión de la venta de *TerraUSD* y por tanto bajando su precio. Esto incrementa el valor de las acciones de Luna, por lo que la bajada de *TerraUSD* se compensa exactamente con la subida de Luna.
- Desviación a la baja: Si sucede lo contrario, se activan de inmediato las obligaciones contingentes de los accionistas de Luna, trasladando la diferencia a los accionistas, que deberán vender Luna para comprar *TerraUSD* y de esta forma reflotar su valor. Esto se consigue emitiendo nuevas acciones de Luna, diluyendo a los presentes accionistas (bajada del valor de Luna) y usando ello para reflotar *TerraUSD*.

Ahora bien, el problema principal viene cuando se entra en un escenario de realimentcion negativa provocado por un shock inicial fuerte. Supóngase que se levantan muchas dudas sobre la seguridad de *TerraUSD*, lo que hace que muchos individuos vendan sus unidades de *TerraUSD*. El algoritmo, para mantener la paridad, emitirá nuevas acciones de Luna para comprar *TerraUSD*. Sin embargo, esto baja el valor de la acción de Luna, que diluye a los accionistas. El problema viene cuando, la dilución es tan elevada, que los accionistas empiezan a no ver rentable poseer Luna, por lo que comienzan a venderlo y se entra en un bucle de realimentación negativa que culmina con la desaparición de la paridad *TerraUSD* al dólar y su colapso como se muestra en esta grafica.


*Ilustración 6. Desplome de Terra tras su realimentacion negativa*

### *2.3.1.3 Wrapped tokens*

Como ya se ha visto, una red *Blockchain* puede poseed una cripto moneda. Un *token* envuelto (o *wrapped token*) permite transferir una criptomoneda o un *token* de una red *Blockchain* a otra.

Por ejemplo, en la red Polygon, existe un *token* que es WETH (*Wrapped* Ether), es decir, que es un *token* que equivale a una unidad de Ether pero que se puede hacer uso del mismo en la red Polygon en vez de en la red *Ethereum*. Esto tiene grandes ventajas como que la red Polygon es mucho más rápida y tiene menos comisiones.

Es muy importante que este *token* este respaldado por el propio cripto activo al que representa. Para asegurarse de ello, y evitar el gasto duplicado, se hace uso de un Custodio, que no es mas que un intermediario que asocia por ejemplo un bitcoin a un WBTC, de tal forma que si se transfiere ese WBTC, entonces el propietario original del BTC no podrá gastarlo en la red Bitcoin, ya que el custodio sabrá que este activo ha sido transferido y no lo permitirá (para poder pasar de BTC a WBTC se le ha dado esta capacidad al custodio).

Es por tanto que se si bitcoin tiene una oferta total de 21 Millones, solo podrá haber 21 Millones de WBTC.

### *2.3.2 DEX*

DEX (*Decentralized EXchange*) simplemente sustituye la autoridad centralizada por un *Smart Contract*. Un Exchange permite el intercambio de *token*s dentro de una misma red *Blockchain* (ya que si se quisiesen intercambiar *token*s a través de redes *Blockchain*s se requeriría un *bridge*)

Los beneficios principales de estos *Exchanges* son que no se requiere KYC (*Know your Client)* para operar, no hay una autoridad centralizada y todo se opera con código abierto de manera muy rápida y bajas comisiones.

Las desventajas de los mismos son que no tiene soporte oficial al ser descentralizado y que la liquidez es en ocasiones escasa.

Por ello, es común el uso de *Liquidity Pools*. Una pool de liquidez se puede ver como un lugar donde existen reservas del *token* y de Ether (de nuevo, mencionar que podría ser una reserva *token*-*token* o *token*-criptomoneda, siendo en este ejemplo, *token*-criptomoneda). En función de la proporción de reserva que haya de cada uno de estos, se deriva el precio del *token* respecto al Ether como el cociente entre ambos. Por ejemplo, si se tuviese un total de 100 *token*s y 50 Ether, entonces el valor de conversión sería: 1 *token* = 0.5 Ether. Si alguien decidiese cambiar *token*s a Ether por este valor, podría introducir 20 *token*s, y retirar 10 Ether. De esta forma, la nueva distribución sería de 120 *token*s y 40 Ether, por lo que el nuevo valor del *token* sería de 1 *token* = 0.3 Ether Se puede ver cómo es un problema que el precio se vea tan afectado por la posible demanda de un único agente. Nótese que si la distribución hubiese sido de 1000 *token*s y 500 Ether, el cambio habría sido idéntico, pero al introducirse 20 *token*s y retirarse 10 Ether, esto no habría afectado apenas al precio. Este es el concepto de liquidez, y sin entrar muy en detalle, el Exchange descentralizado incentiva a que diferentes agentes introduzcan liquidez en el mercado.

### *2.3.3 DAO*

DAO (*Decentralized Autonomous Organization*) es una organización basada en código mediante *Smart Contract*s, resultando en una organización autónoma.

Todas las partes del proceso de una empresa se automatizan de manera descentralizada, aunque los propietarios/accionistas de un DAO pueden votar cambios que se reflejan en el cogido de los contratos

Un DAO tiene *token*s y estos *token*s permiten votar en materia de contratación, salarios, etc.

Los beneficios asociados a estos modelos es que no requieren confianza. No hay ningun CEO en el que la junta de accionistas deba confiar, sino que se hace todo automático con el código de los *Smart Contracts*. Además, no puede bloquearse por parte de una entidad central y al ser código abierto es totalmente confiable.

Por otro lado, las desventajas de este tipo de organizaciones descentralizadas aparecen principalmente justo por la naturaleza de código abierto que evita la existencia de secretos corporativos (por ejemplo, inversión en investigación y desarrollo no es viable) y dicho código abierto puede dar lugar a ataques. Uno muy conocido es el ataque a The DAO donde se robaron 50 millones dólares del fondo The DAO lo que dio lugar al *fork* entre *Ethereum classic* (donde permanece la cantidad robada) y Ethreum (donde se devolvió).

### *2.3.4 STAKING, LENDING AND INSURANCE*

***Staking***: Hoy en día, muchos mecanismos de verificación se basan en POS (*Proof of Stake*). Aquel que bloquee un mayor número de *token*s tendrá el derecho a minar el siguiente bloque y por tanto será recompensado por ello. Por lo tanto, se puede prestar los *token*s a otro usuario para que el haga la validación y en consecuencia comparta los beneficios obtenidos gracias a esa validación.

***Lending***: AAVE es una plataforma  Permite prestamos *peer-to-peer* y usa un algoritmo para ver los tipos de interés a cobrar. El objetivo es realizar un match automático . Los algoritmos

se basan en el colateral que se tiene. Es anónimo. Aunque se debe tener como colateral como mínimo la cantidad que se demanda.

**Seguros**: Se trata de una socialización del riesgo. Protege a los usuarios contra bugs de *Smart Contract*s, contra *rug pulls*, etc. No es más que otro intercambio a través de un *Smart Contract*s. Un ejemplo de una seguradora DeFi es Nexus.

## 2.4 NFTs

Las siglas NFT representan las palabras en ingles Non-Fungible *Token*. Por tanto, parece que no es más que un *token*, concepto que se ha visto anteriormente, y que tan solo requiere de redactar un *Smart Contract* con las condiciones de la creacion del mismo. Sin embargo, la particularidad de No fungible es relevante. Cuando algo es fungible, significa que es intercambiable y perfectamente remplazable. Un ejemplo típico es el dinero, es decir, a un usuario le es indiferente poseer una moneda de un euro que otra, o un billete de 10 euros que dos de 5. Por tanto se dice que el dinero es fungible. Lo mismo sucede con los *token*s mencionados en el apartado anterior, que en última instancia son equivalentes a la creacion de un dinero propio. Sobre una oferta de 1 millón de *token*s, me es igual poseer del 100 al 110 que del 500 al 510. Son por tanto fungibles.

Sin embargo, la capacidades programáticas de los *Smart Contract*s permiten crear *token*s que sí que sean diferenciables entre sí. Si se redacta un contrato en el que se indica que se crean 10 *token*s, cada uno con un nombre, entonces automáticamente se vuelven no fungibles ya que no son perfectamente intercambiables entre ellos, al tener cada uno un nombre diferente. La diferenciación de estos *token*s no fungibles es, por supuesto, más compleja, pero, en esencia, no deja de ser una serie de líneas de código escritas en un *Smart Contract*. Esta información en el *Smart Contract* es perfectamente auditable y su propiedad verificable gracias a las propiedades de la tecnología *Blockchain* mencionadas anteriormente. Por poner un ejemplo real que aclare este concepto, es posible programar un *token* que sea un gato y que tenga una serie de atributos que lo hagan diferente de otros *token*s de tipo gato. Esta iniciativa fueron los *CriptoKitties*, cuyo NFT son gatos con distintos aspectos, que existen

de manera distribuida en el *Smart Contract* en cuestión y cuya representación gráfica se muestra mediante una interfaz gráfica en una página web tradicional

# Capítulo 3. USO DE UN *MARKETPLACE*

El objetivo de esta sección es el de hacer uso de un *Marketplace* existente para realizar el proceso de *minting* y compraventa de un NFT.

## 3.1 WALLET, RED Y *MARKETPLACE*

Para poder realizar un caso práctico en el que se haga el *minting* (publicación) y la compraventa de un NFT, primero se ha de elegir la *wallet* a emplear, la red *Blockchain* en la que realizar la transacción y el *Marketplace* en el que se hará.

**Wallet:** Para ejemplificar este proceso, se hará uso de la cartera *Metamask*. La razón es que una cartera sencilla de utilizar que le es familiar al autor debido a que hizo uso de la misma en su trabajo de fin de grado de Ingeniería de las telecomunicaciones.

**Red**: Existen varias alternativas en materia de redes *Blockchain* que utilizar como ecosistema para los NFTs. La red que se va a emplear en este caso es Polygon, que como se ha explicado en apartados anteriores en este trabajo, es una solución de capa 2 (*layer 2*), por lo que es muy escalable, rápidas y eficientes lo que en torno implica menores costes en términos de comisiones por *Gas*. Al tratarse de una prueba de concepto, el incentivo es el de poder hacer uso de la funcionalidad completa para el caso práctico con los menores costes posibles, y esto lo permite la red Polygon.

***Marketplace***: Se hará uso de un *Marketplace* NFT para este caso de uso, es decir, un *Marketplace* creado por una empresa para facilitar el intercambio de NFTs a cambio de una comisión. Detrás de este *Marketplace*, a nivel tecnológico, existen una serie de contratos basados en estándares ERC. Estos estándares se analizarán más en detalle en el capítulo de Desarrollo de un *Marketplace*. El *Marketplace* que se empleará en este caso práctico es OpenSea. Es el *Marketplace* más popular [8] y tiene comisiones razonables por lo que representa un entorno idóneo para el caso práctico.

## 3.2 OPENSEA

OpenSea es una plataforma que pone en contacto a compradores y vendedores de NFTs. Para ello, ofrece dos opciones: La compraventa en el mercado primario y en el mercado secundario. El mercado primario, igual que en los mercados financieros, es aquel donde se produce la compraventa de un activo por primera vez, en este caso de un NFT. Tras esa primera compraventa, el nuevo propietario puede poner a la venta ese activo adquirido en el mercado secundario. Gracias a la naturaleza programática de los NFTs, esto presenta posibilidades para el autor original del NFT, pudiendo este incorporar condiciones en el NFT como la de recibir una comisión cada vez que se venda el producto en el mercado secundario.

A continuación se explica cual cómo funciona OpenSea a nivel tecnológico.

Cuando un usuario entra en OpenSea, observa el *frontend* de la página web, creado como interfaz entre el usuario y el *backend* (el *backend* incluye bases de datos de la web, servidores y las redes *Blockchain* en las que opere). Imágenes de la misma se mostrarán en el apartado "Proceso de Compraventa". OpenSea está conectado con varias redes *Blockchain* y un usuario puede hacer uso de la que desee para la compraventa de NFTs. Una vez elegida la red a utilizar, un usuario puede crear y poner a la venta un NFT sin ninguna comisión. La forma en la que esto es posible es debido a cómo funciona OpenSea; Cuando un usuario indica que quiere crear un NFT y ponerlo a la venta, esta orden se firma con la clave privada del usuario, pero no se manda aun a la red *Blockchain*. Esta orden firmada, la cual incluye el NFT que se pone a la venta, su precio y otros campos para poder proceder con la transacción, se *guard*a en un contrato centralizado bajo el control de Opensea. Por tanto, un usuario firma con su clave privada ciertas condiciones de transacción y esta se *guard*a de manera centralizada. Esto permite que potenciales compradores puedan observar estos NFTs en OpenSea sin que aun existan en la red *Blockchain*. Cuando se pone a la venta un NFT, el autor puede incluir condiciones para la transacción (e.g. que solo pueda comprarlo cierto grupo de usuarios, que se pueda realizar una subasta durante un periodo determinado de tiempo, etc.). Por tanto, todo tipo de operaciones previas a que se efectúe la transacción se

realizaran en este *Smart Contract* local de OpenSea lo que implicará que no tengan coste alguno para los usuarios.

Una vez que existe un emparejamiento entre una orden de compra y una de venta, la lógica del *Smart Contract* de OpenSea se ejecuta, mandando el par de transacciones (cada una firmada con la clave privada de sus propietarios) a la red *Blockchain* en la cual se encuentra un *Contract*o denominado Exchange.sol [9] que es parte del conjunto de Smart *Contract*s dentro del *Wayven Protocol* [10]. Este *Smart Contract* comprueba que las órdenes de compra y de venta indican lo encajan y procede a efectuar la transacción en la red, lo cual tiene sus correspondientes comisiones a ser cobradas por los agentes que minen la transacción. Como se muestra, OpenSea tiene un equilibrio entre centralización y descentralización. Por un lado, permite operar a los usuarios de manera centralizada lo cual facilita las interacciones sin necesidad de pagar gas hasta el punto en el que la transacción debe ser definitiva y se manda a la red *Blockchain* para su ejecución. OpenSea no puede manipular las transacciones ya que a pesar de quedar centralizadas hasta su envió final, estas deben ser firmadas por los usuarios, pudiendo por tanto como máximo no incluir las transacciones pero jamás modificarlas.

Finalmente, hay que mencionar que cada usuario que opera en OpenSea tiene asociado un *Proxy Contract,* lo cual a través de un Proxy *registry* consigue que los NFTs se *guard*en en dichos contratos, evitando así que se deban pagar comisiones cada vez que se crea y vende un NFT al no crearse un contrato (ERC 721) por cada NFT. En el apartado "Desarrollo de un *Marketplace*" se entrará en detalle sobre los diferentes campos, contratos y estándares para el contrato de NFT y *Marketplace*.

## 3.3 COLECCIONES

Una vez visto cómo funciona a nivel tecnológico OpenSea, y antes de ver el flujo y proceso seguido en esta prueba de concepto, cabe mencionar que los NFTs se agrupan dentro de

colecciones. Una colección es por tanto una agrupación de NFTs con cierta temática. Ejemplos conocidos son *CriptoKitties* o *CriptoPunks*. En este apartado se entrará en detalle sobre cuáles son las claves más importantes para maximizar las probabilidades de lanzar una colección de NFTs que tenga éxito, genere interés y por tanto curse numerosas operaciones de compraventa.

1. Atender a los elementos macro: Como es natural, la probabilidad de que una colección de NFTs tenga éxito es mayor si el mercado de NFTs está creciendo y de moda.
2. Comunidad: Crear una comunidad antes de lanzar la colección es imprescindible. Sin una comunidad que dote de cierta tracción al proyecto es difícil que este tenga éxito. Esta comunidad puede existir a priori (como es el caso de *Youtubers* o *influencers* que lanzan sus colecciones) o puede ser creada de cero previo al lanzamiento (requiere mayores incentivos a los compradores en términos de potenciales retornos). En este segundo caso, es importante encontrar inversores de calidad, conectar con entusiastas de los NFTs y con *influencers* que sean capaces de promocionar la colección, además de tener presencia en las principales redes sociales (*Twitter*, *TikTok*, *Telegram*, *Discord*, etc.) y crear campañas de marketing con incentivos como *whitelists* que otorguen privilegios a aquellos que apuesten pronto por el proyecto.
3. Precio: El precio correcto es un elemento importante para el éxito del proyecto. Un precio excesivo puede desmotivar a los inversores del proyecto mientras que un precio demasiado bajo puede de la misma forma desincentivar a inversores al ver poco retorno potencial además de que podrían estarse dejando beneficios para el autor sobre la mesa. Para encontrar el precio idóneo, se comienza por analizar proyectos similares para tener una noción inicial del rango y orden de magnitud de los precios. A continuación, una comunicación transparente con la comunidad que se ha creado es muy útil para conseguir llegar a un precio definitivo, teniéndose en mente que el autor puede beneficiarse de la compraventa de NFTs tanto en la venta original (mercado primario) como en las ventas posteriores en el mercado

secundario a través de una comisión. Esta comisión es parte de la decision del precio a tener en cuenta, al ser otra fuente de ingresos para el autor que captura parte del retorno de la comunidad inversora.

4. *Roadmap*: El *roadmap* consiste en la visión a largo plazo del proyecto. Si bien es cierto que un lanzamiento exitoso en el mercado primario es el primer paso, si se hace correctamente, la gran mayoría de los beneficios de la colección vendrán por royalties de las transacciones en el mercado secundario. Esto incentiva a tener una visión a largo plazo del proyecto y por tanto la comunidad puede estar abierta a royalties en los mercados secundarios para desincentivar visiones cortoplacistas que resulten en estafas como los *rug pulls.*
5. Comunicación: Una comunicación efectiva y frecuente con la comunidad es imprescindible, especialmente en los inicios del proyecto donde existe aún mucha FUD (*Fear Uncertainty and Doubt*). Los canales más habituales para mantener un contacto estrecho con la comunidad son Discord y Telegram. Tener moderadores en estos canales puede ser muy útil para gestionar la gran cantidad de mensajea que preceden el lanzamiento de un proyecto exitoso. Finalmente, si los autores de la colección se muestran a través de un video y realizan AMAs (*Ask Me Anything*) de manera frecuente, esto ayudara a reducir el FUD enormemente.
6. Calidad del código: El código de los NFTs creado debe ser de calidad. En el caso de OpenSea, las plantillas que emplea se encargan de este punto. Sin embargo, como se verá en el apartado de "Desarrollo de un *Marketplace*", cuando de hace uso de *Smart Contract*s propios, no es suficiente basarse en los estándares (ERC 771 y ERC 1155) sino que el código tiene que estar escrito considerando todos los posibles *exploits* de seguridad, al ser esto un elemento critico de cualquier colección.

## 3.4 PROCESO DE COMPRAVENTA

A continuación se documentan los pasos más relevantes del proceso de compraventa de NFTs mediante OpenSea. Para ello, se crearán 3 NFTs a partir de 3 imágenes del autor David Quintana, como prueba de concepto para el uso de OpenSea.

1. **Metamask:** En primer lugar, se creará un par clave publica privada a través de Metamask para la red *Blockchain Ethereum*, lo que resulta en una cuenta que nos permitirá interactuar con la *Blockchain*. Metamask, como se explicó en el apartado anterior sobre *Wallets*, mantiene la clave privada oculta a las páginas web en las que se navega y la empleará únicamente para firmar transacciones que se autoricen por el usuario. Una vez creada una cuenta en *Ethereum* mediante un par clave publica privada ya se puede comenzar a interactuar con la red.

2. **Polygon:** Polygon es una red de capa 2 (layer 2), es decir, que existe sobre la red *Ethereum* original pero crea una cadena de bloques encima de esta red que es capaz de realizar muchas más transacciones por segundo y con unas comisiones por transacción muy inferiores [11]. Esta red no solamente es muy popular en la comunidad *Ethereum* por los beneficios mencionados, sino que es muy útil para hacer una prueba de concepto con OpenSea con un comisiones bajas. A continuación se accede a la red Polygon y se conecta para poder operar en la misma a través de Metamask.

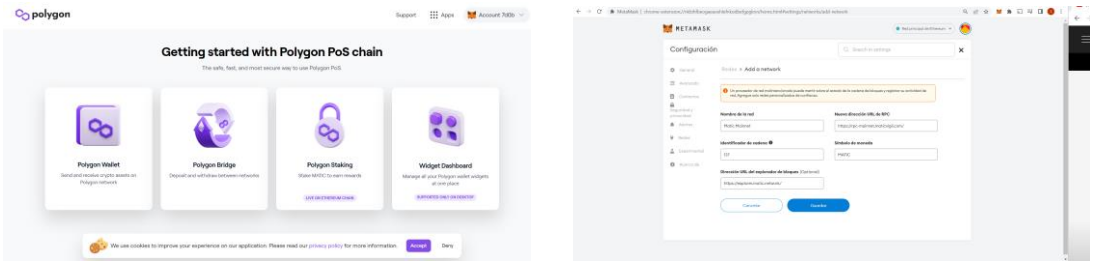


*Ilustración 7 Pagina de Polygon y conexión a través de Metamask*

3. **OpenSea. Creacion de cuenta:** Una vez se ha conectado a Polygon, se procede a crear una cuenta en OpenSea *Marketplace*. A diferencia de otras páginas web, donde una cuenta se crea mediante usuario y contraseña (normalmente requiriéndose un correo electrónico), en OpenSea se crea una cuenta asociando tu cuenta pública de *Ethereum* a OpenSea a través de Metamask. Para verificar que el usuario es propietario de la dirección *Ethereum*, OpenSea pide al usuario que firme mediante su clave privada un mensaje. Esto lo hace Metamask de manera transparente para el usuario. Tras ello, OpenSea recibe el mensaje y hace uso de la clave publica del usuario para descifrarlo y comprobar de esta forma que el usuario es propietario de esa dirección publica (y por tanto, del par clave pública-privada). Esto no solo sirve para que OpenSea comprueba la identidad del usuario sino que también sirve para que este acepte los términos y condiciones de la página. Por supuesto, Metamask pide al usuario permiso para realizar esta firma.

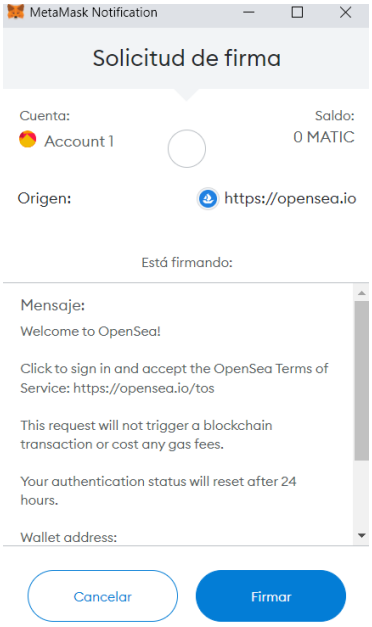


*Ilustración 8. Metamask firma la solicitud de OpenSea*

Con ello, se crea la cuenta con la que se procederá a realizar la prueba de concepto:

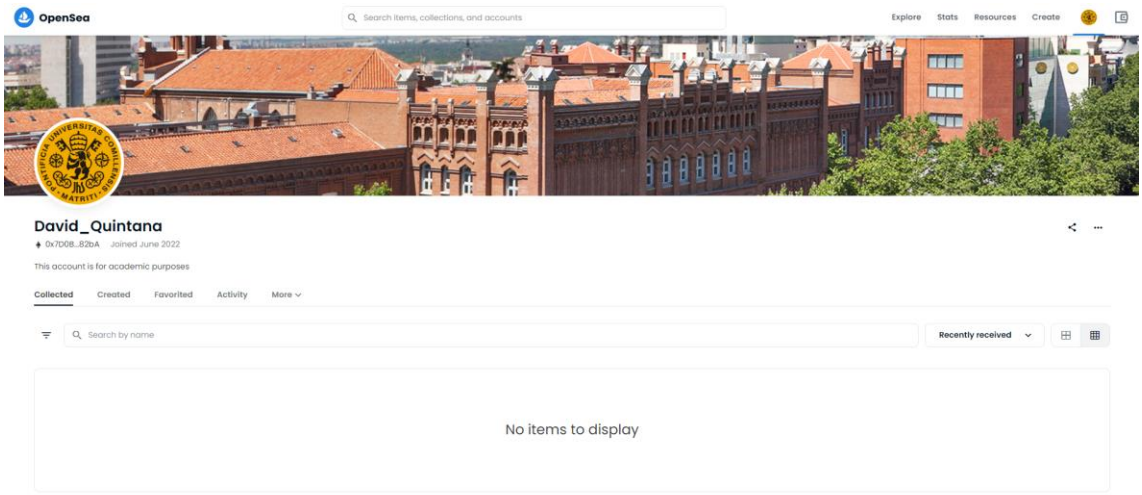


*Ilustración 9 Usuario de David Quintana creado para la prueba de concepto*

Nótese que no se han incluido ni foto ni descripción del autor ya que a pesar de contar con el conocimiento del autor, él no opera la cuenta directamente. De hecho, a lo largo del ejercicio de crear su colección y mintear sus NFTs, se añadirá la siguiente aclaración:

*IMPORTANT NOTE: This is just a test collection for academic purposes ONLY. This account, collection and NFTs are not managed by the author, David Quintana. Hence, the purchase of any element of this collection will NOT grant the buyer with any property rights or any other kind of rights that David Quintana could provide.*

Tras esto, se repite el proceso con una segunda cuenta que servirá para simular el proceso de compraventa.

4. **OpenSea. Explorando la página.** Tras crear las cuentas que permitirán la interacción con la página, se procede a realizar una exploración de la misma. Más allá de la posibilidad de crear y poner a la venta NFTs, la página permite explorar los NFTs que han sido puesta a la venta en el mercado primario/secundario. Ejemplos populares y mencionados a lo largo de este TFM son *CryptoKitties*, *CryptoPunks* o

*Bored Ape Yatch Club*. Se muestra como ejemplo *Cryptopunks*, además de otra vista adicional donde se puede ver la actividad en esta página.

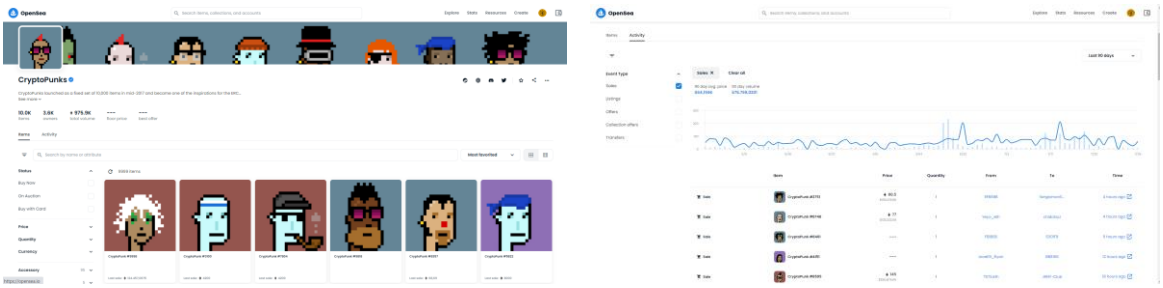

*Ilustración 10 Vista de la colección CryptoPunks en OpenSea y su actividad*

5. **Creacion de una colección:** A continuación se procede a crear una colección donde se *mintearan* los NFTs del autor, David Quintana. La colección se llamará DaquiCollage [12] y se creará para la *Blockchain* Polygon con una oferta por NFT de 1 (únicamente habrá un NFT único y no tendrá ninguna copia).

6. **Minteo de NFTs y puesta a la venta:** En este punto, se mintean los 3 NFTs y se *guarda* dentro de la colección DaquiCollage creada. Los nombres son "ChiCa Splash – Chongoso", "MapaChe" y "Manola 2". Se muestra la vista final de este paso:

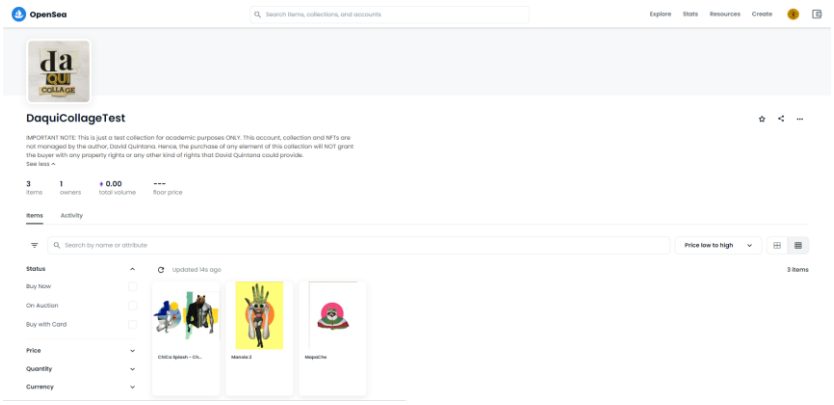


*Ilustración 11 Colección DaquiCollage con 3 NFTs minteados*

7. **Venta de NFT:** A continuación, se pone a la venta uno de los NFTs minteados.; "Manola 2". Para ello, OpenSea peticiona la firma de la transacción mediante Metamask, para comprobar que es el actual propietario quien está poniéndolo a la venta.

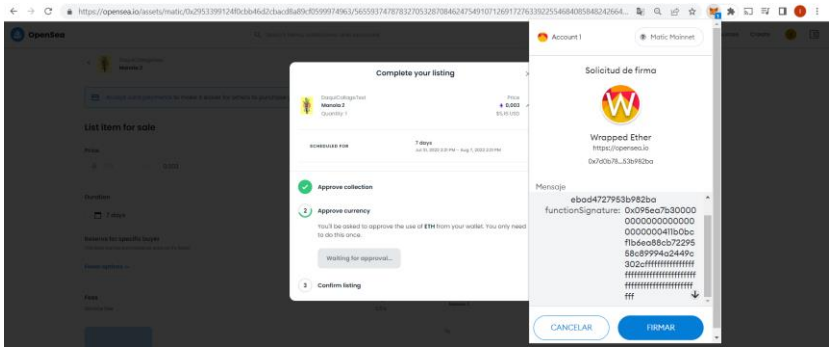


*Ilustración 12 Firma mediante metamask para autorizar la venta del NFT Manola 2*

Tras poner el NFT a la venta, este aparece en la página de OpenSea. El propietario y creador del NFT aun no habría realizado ningún pago por todo esto. Esto es debido a que, como se ha explicado, hasta que no haya un emparejamiento para la transacción, todo se realiza de manera centralizada en los servidores de OpenSea.

8. **Compra del NFT:** Se procede a comprar el NFT puesto a la venta para ver cómo se produce el cambio de propietario. El precio mínimo al que se pueden vender NFTs es de 0.003 ETH o unos $5. Por lo que es el precio que se empleará para este ejemplo. Se accede a la cuenta que se empleará para realizar la compra. Desde la misma, se aprecia como el propietario actual de "Manola 2" es David_Quintana.

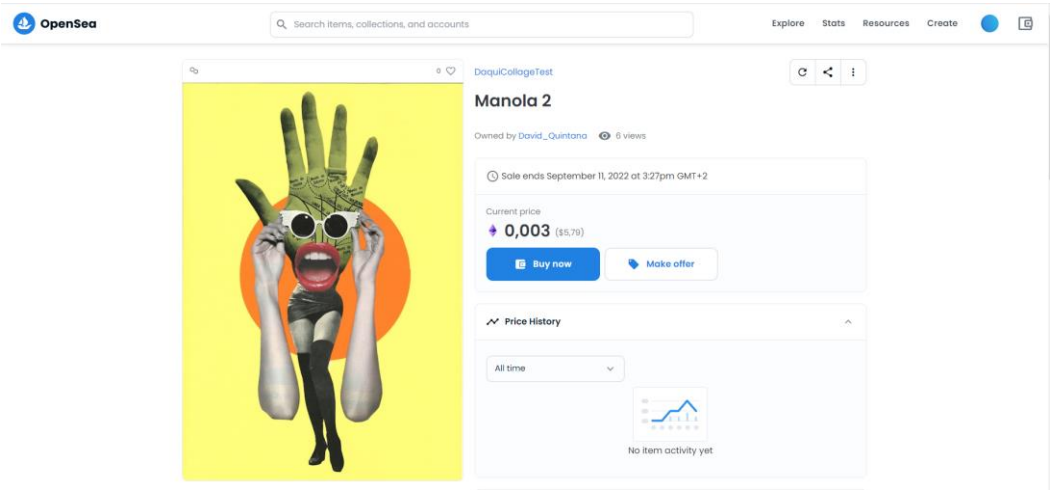


*Ilustración 13. NFT Manola 2 propiedad de David_Quintana*

Para poder realizar la compra, se requiere el *token* WETH (*Wrapped Ether*), que se trata de Ether pero en la red Polygon. Como el NFT ha sido puesto a la venta en la red Polygon, el pago se realiza en este *token*. Se omiten los pasos para conseguir este *token*, pero en esencia dentro de la Wallet Metamask existe la opción de conseguirlos. A continuación, se procede a realizar la compra del NFT.

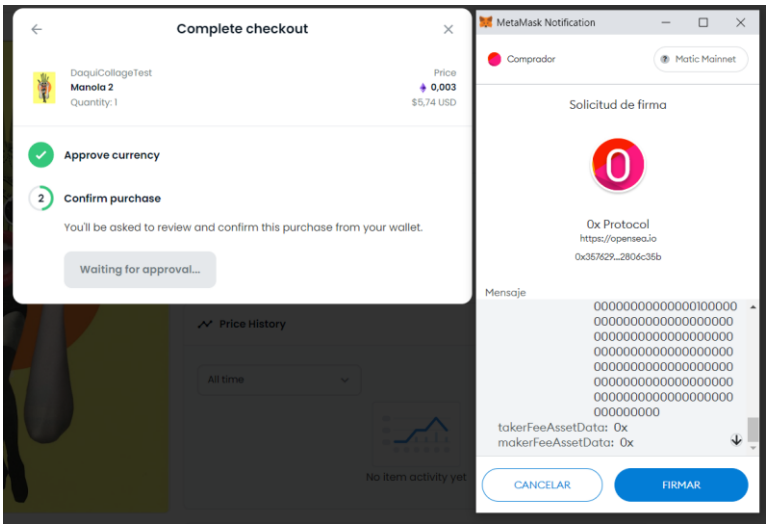


*Ilustración 14 Firma mediante metamask para confirmar la compra del NFT*

Se firma la compra y se observa el cambio de propietario.

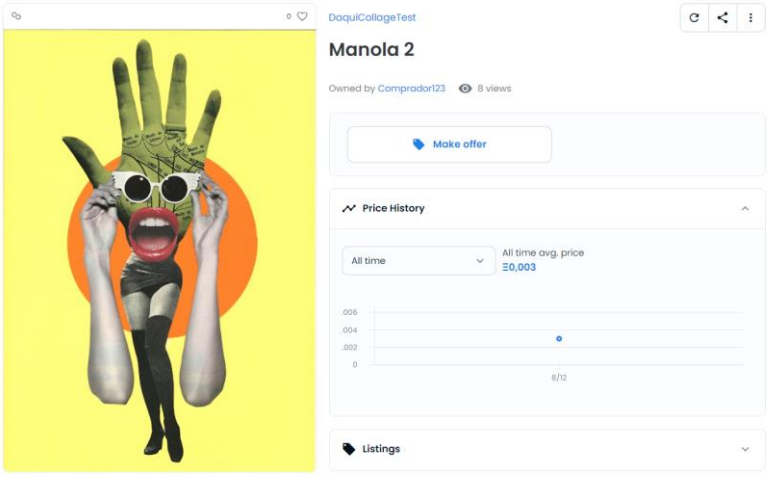


*Ilustración 15 Captura de pantalla mostrando el cambio de propietario*

Para recapitular lo sucedido, la cuenta de David_Quintana ha firmado una transacción que permite la venta de su NFT si se ofrece un determinado precio (0.003ETH). Esta transacción no se ha mandado a la *Blockchain* de Polygon, sino que se ha *guard*ado en un servidor centralizado de OpenSea, para evitar los costes que conllevaría interactuar con la red Polygon. Tras esto, el usuario Comprador123 ha firmado una transacción en la cual pide intercambiar unos *token*s por otro (*token*s WETH por el *token* no fungible Manola 2). Firma esta transacción y se la manda a OpenSea. OpenSea ahora tiene este par de transacciones y una vez que comprueba que son coincidentes, las manda a la red Polygon para hacer efectivo el intercambio, haciendo uso únicamente de la red Polygon y por tanto pagando comisiones cuando es estrictamente necesario.

# *Capítulo 4.* DESARROLLO DE UN *MARKETPLACE*

Este apartado tiene como intención el desarrollo de un *Marketplace* NFT mediante el uso de *Smart Contract*s. Para ello, se comienza con un apartado explicativo de las tecnologías que se empleara durante el mismo. Tras ello se procede a explicar los estándares de *Ethereum* que se emplearan para desarrollar los contratos NFT (ERC721) y *Marketplace* (ERC 1155). A continuación se describirá la temática para el *Marketplace*. Finalmente, se explicará un flujo conceptual explicando lo que sucederá al hacer uso de las tecnologías mencionadas y la red *Ethereum* y se documentará el uso del mismo. Nótese que al final del presente trabajo se incluirán unos anexos con el código empleado en los *Smart Contract*s (contrato *Marketplace* y contrato NFT) de esta prueba de concepto, así como cualquier otro código que sea pertinente.

## 4.1 TECNOLOGÍAS A EMPLEAR

A continuación, se procede a hacer una explicación de las tecnologías requeridas para el desarrollo del *Marketplace*. Se realizará una división en tecnologías frontend y tecnologías *Blockchain* (contratos). Téngase en cuenta que a veces hay cierta superposición entre ambas, pero es una buena forma de delimitarlos. Por supuesto, se omitirán explicaciones de lenguajes de programación como *JavaScript* o de lenguajes de *markup* o estilo como HTML o CSS ya que aportan escaso valor frente a tecnologías más relevantes para esta prueba de concepto.

### *4.1.1 TECNOLOGÍAS FRONTEND*

En esta sección, se explicarán las principales tecnologías empleadas en el cliente de la aplicación, para tener así una base sobre la cual la explicación del desarrollo se pueda basar.

#### *4.1.1.1 Anaconda (Python)*

Anaconda es una distribución de Python y R [19]. Una distribución es una forma conveniente de entregar paquetes de manera sencilla a un desarrollador. Paquetes como Jupyter son empleados por comunidades de científicos de datos. Para este proyecto, los paquetes que se van a requerir con los paquetes de Python y se obtendrán mediante el gestor de paquetes Conda para que otras tecnologías como Brownie (explicada a continuación) puedan hacer uso de estos paquetes.

Una vez instalado anaconda, se hace uso de pip para instalar las librerías de Python para brownie mediante el comando:

*pip install eth-brownie*

#### *4.1.1.2 Node JS & NPM*

Node.js es un entorno que permite ejecutar código *JavaScript* fuera del navegador. Esto lo consigue mediante el uso de un V8Engine. Mas detalles sobre esto pueden ser consultados en el TFG del autor [1].

Por otro lado, NPM (*Node Package Manager*) es un gestor de paquetes de node. Un paquete es cierta funcionalidad que se encapsula y mediante NPM se comparte con la comunidad de usuarios. NPM puede ser accedida mediante una interfaz visual (una página web) y mediante línea de comandos. La última opción suele ser la más frecuente. En consecuencia, si un usuario desea hacer uso de un paquete concreto desarrollado por un tercero, mediante NPM puede, de manera sencilla, instalar dicho paquete y todas las dependencias necesarias para que el mismo funcione. Si desea dejar de usarlo, y eliminarlo, NPN también permite retirar el paquete con todas las dependencias del mismo (que no estén siendo empleadas por otros paquetes).

Para este proyecto, el uso principal de node y npm es para instalar el paquete web3.js y poder así interactuar con los contratos.  Para instalarlo se hace uso de:

*npm install web3*

#### *4.1.1.3 Angular*

Angular es un *framework frontend* basado en *TypeScritp* (en esencia, *JavaScript* con sintaxis para tipos). En el TFG de Ingeniería de Telecomunicaciones se hizo construyo también una aplicación un tanto diferente. En la misma, se hizo uso de JQuery como *framework JavaScript*, lo cual fue bastante tedioso. Por recomendación de las fuentes empleadas y consultadas para la realización de este trabajo de fin de Master, se hace uso de Angular para construir la aplicación. Además, la interacción con web3 con el *frontend* se simplifica mucho con Angular. Añadir también, que al tratarse de un *framework* nuevo para el autor y con cierto nivel de complejidad, ha sido necesario cierto estudio del mismo. Por ello, a lo largo de la explicación de este capítulo, se hará hincapié en los aspectos de Angular que sean relevantes.

A alto nivel, Angular tiene el objetivo de crear *single page application*s, es decir, una página única sobre la cual se va cargando la información. Existen otros *frameworks* como React o Vue que también permiten hacer páginas de vista única.

Angular.js fue desarrollado por ingenieros de Google en 2010 como un *framework* basado en *JavaScript*, y desde 2015 varias versiones basadas en *TypeScript* se han ido actualizando. Angular se basa en componentes. Todo lo que se ve en una página son componentes, que tienen su propia funcionalidad y lógica. La suma de varias componentes da lugar a una vista en una página. Por tanto, principalmente cabe destacar los ficheros .html y .ts de cada componente. El .html incluir a la parte visual y el .ts la lógica. La idea está en que dichas componentes sean todo lo independientes posibles y se junten para crear una pagina, lo que simplifica mucho el desarrollo de aplicaciones.

Angular hace uso de enrutamiento de cliente en vez de enrutamiento de servidor. En aplicaciones tradicionales, cuando el usuario hace *click* en un link dentro de una página, el *frontend* se comunica con el servidor (*backend*) para pedirle esa información y poder así

mostrársela al usuario. Sin embargo, Angular, en general, no necesita pedirle contenido al servidor, ya que se encuentra todo en los componentes que están en el cliente.

Además, es un entorno excelente para el *testing* ya que permite hacer *testing* de módulos de manera independiente mediante *dependencies injections*, es decir, que es capaz de aislar un módulo de sus dependencias y simular la respuesta de las dependencias.

En esencia, Angular se basa en componentes, y cada componente tiene por un lado una vista y por otro lado una lógica. Las visitas son archivos .html mientras que la lógica se encuentra en archivos .ts.

Resumidamente, Angular plantea un paradigma diferente a las aplicaciones tradicionales, y para el autor constituye un conocimiento nuevo y para el que ha sido necesario estudio para aprender sobre el mismo. Por ello, se dedicará parte de esta memoria a explicar algunos de los puntos más relevantes para el desarrollo del proyecto.

Una vez instalado Angular, se obtiene la siguiente vista al ejecutar el comando:

*Ng serve*

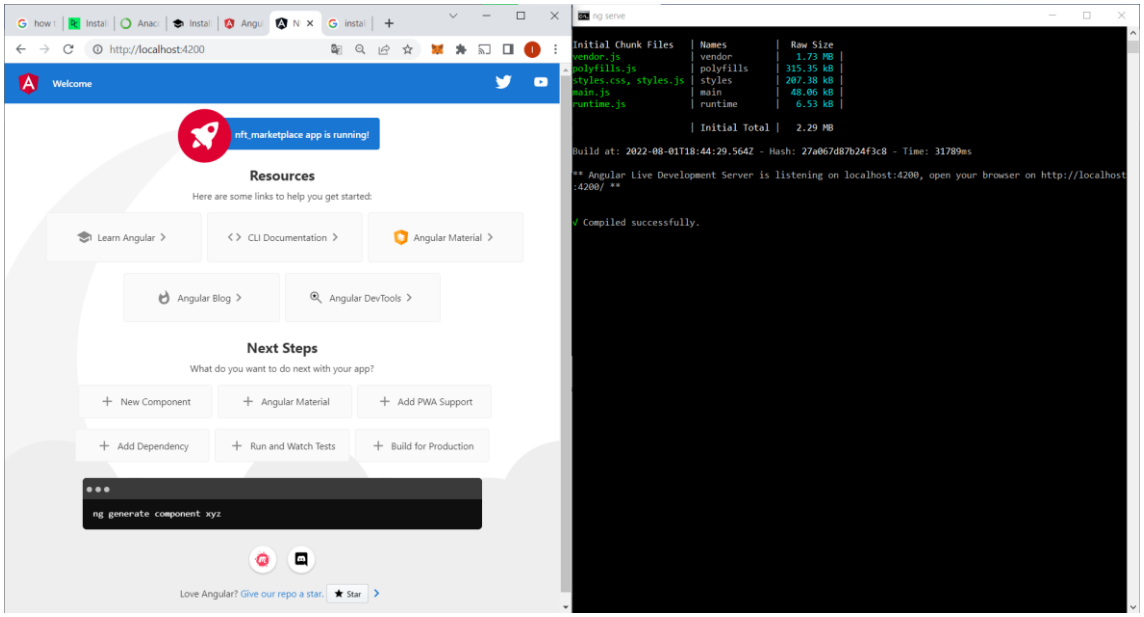


*Ilustración 16. Puesta en funcionamiento del servidor de Angular*

### *4.1.2 TECNOLOGÍAS BLOCKCHAIN*

A continuación, se procede a explicar las tecnologías relacionadas con el *backend* que en este caso es *Blockchain*. La tecnología *Blockchain*, como se ha explicado a lo largo de este trabajo, es una tecnología sobre la que existen varios protocolos, por lo que para poder hacer uso de este *Marketplace*, será necesario conectarse a uno de estos protocolos sobre los que se encontraran los contratos que se desplieguen

#### *4.1.2.1 Metamask*

Metamask actúa como un puente entre el navegador de un usuario y cualquier red *Blockchain* que haga uso de una EVM (*Ethereum* Virtual Machine). Esto lo hace permitiendo, de manera sencilla e intuitiva, crear y gestionar cuentas de usuario e interactuar con las redes *Blockchain* habilitadas.

Ya se ha explicado antes como Metamask actúa como una Wallet en el navegador y se ha mostrado la creacion de cuentas para operar en OpenSea. Para esta segunda parte del proyecto, se creará una nueva cuenta denominada “Programador”, que se usará para desplegar e interactuar con los contratos ERC 721 y ERC 1155. La dirección pública de esta cuenta es : 0x3BBD0Cf2E6DbcE8f39F05c5c185C7FED7EB405f4, mientras que se omite la clave privada al contener fondos que han sido necesarios para la prueba de concepto.

No se profundiza más en esta tecnología al haber sido también explicada en secciones anteriores.

#### *4.1.2.2 Web3*

Como se ha explicado en el apartado de Node JS & NPM, web3 es un paquete que incluye una serie de librerías que permiten la interacción entre el *frontend* de la aplicación y la red *Blockchain*. Aquí cabe realizar una distinción importante; la diferencia entre la interacción que realiza Metamask con la red *Blockchain* y la que realiza web3 con la red *Blockchain*.

Es posible que aparezcan dos alternativas pero en realidad son la misma, ya que Metamask usa web3 para realizar sus interacciones. Un usuario que haga uso de un navegador podrá firmar una transacción mediante Metamask (que a su vez usará web3). Sin embargo, si una web desea mostrar el contenido de varios NFTs sin que esto sea peticionado por el usuario, este tendrá que conectarse mediante al librería web3 a la red *Blockchain* para conseguir esta información (por supuesto, podría hacer las llamadas a mano haciendo uso del Código ABI de un *Smart Contract*, concepto que se explicará posteriormente, pero la forma tradicional y simple de hacer estas interacciones es mediante web3).

Para poder construir este *Marketplace*, el frontend de la aplicación debe poder interactuar con la red *Blockchain*, por lo que se descarga la librería con el siguiente comando:

*npm install web3*

#### *4.1.2.3 Brownie (Ganache)*

Brownie es un entorno de desarrollo y testing para redes que hagan uso de la *Ethereum Virtual Machine* (EVM). Este entorno está basado en Python.

En el TFG de Ingeniería de Telecomunicaciones, el autor hizo uso de Truffle en vez de Brownie, que es otro entorno de desarrollo y testing pero basado en *JavaScript*. Por la experiencia, el tratar con *JavaScript* en especial con funciones *async*/*await* hacía el código algo difícil de seguir y tedioso en algunos puntos. Por ello, se ha preferido optar por un entorno de desarrollo basado en Python.

Existen ventajas e inconvenientes notables para ambos lenguajes (*JavaScript* y Python) pero para esta prueba de concepto en concreto no supondrá una gran diferencia el decantarse por una frente a otra.

Tras haberse instalado Brownie, para crear una estructura de proyecto inicial, se hace uso del comando:

*Brownie init*

Esta estructura se mostrará más adelante en más detalle.

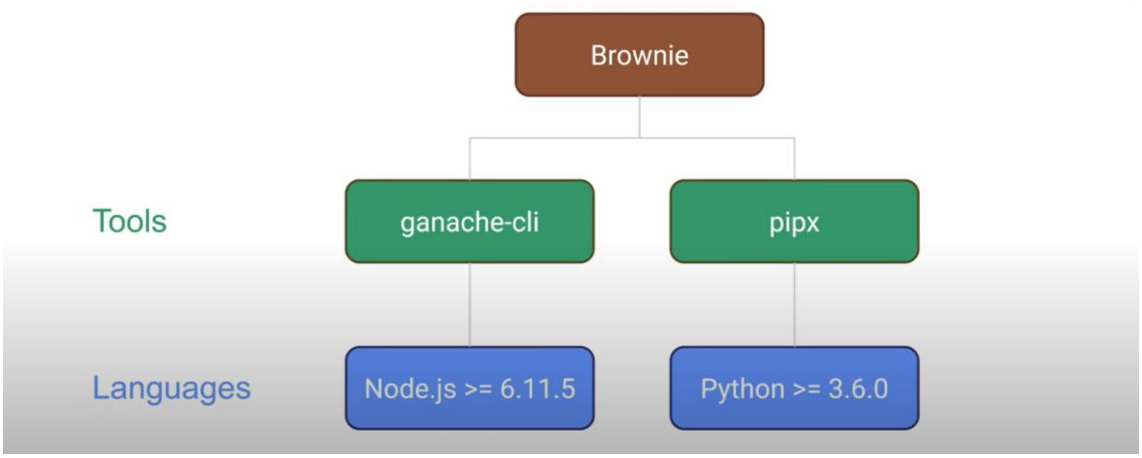


*Ilustración 17 Estructura jerárquica de Brownie*

Por otro lado, Brownie interactúa con Ganache. Ganache es una red de *Blockchain* de prueba utilizada para el *testing* de aplicaciones que interactúan con la red *Blockchain*. A pesar de que Brownie haga uso de Ganache, Ganache debe ser instalado aparte al no venir incluido con la instalación de Brownie. Es muy importante el uso de redes de *testing* como ganache, ya que de lo contrario, cualquier desarrollo de aplicaciones *Blockchain* debería de pasar por redes *Blockchain* públicas las cuales cobrarían gas por su uso y haría que el *testing* de estas aplicaciones fuese muy costoso.

#### *4.1.2.4 Binance Smart Chain (BSC)*

En apartados anteriores se ha explicado que la red Polygon es una red de capa 2 que corre sobre la red *Ethereum*, y que por ello consigue una mayor cantidad de transacciones a un precio más reducido. Es una red muy empleada para el desarrollo de aplicaciones *Blockchain*.

Por otro lado la *Binance Smart Chain* (BSC) fue creada por Binance, que es uno de los mayores exchanges de criptoactivos del mundo. Binance decidió montar su propia red *Blockchain* y crearon la *Binance Chain*, la cual no hacia uso de *Smart Contracts*, por lo que

tras un tiempo, decidieron montar la *Binance Smart Chain* (o *BNB Chain*). Esta red es un *fork* de *Ethereum*. El concepto de *fork* esta explicado en el TFG del autor [1] por lo que se omite el detalle de la misma, pero para simplificarlo, se trata de una copia de la red *Ethereum* a la que se le realizan ciertos ajustes. En el caso de BSC, los ajustes que se realizaron buscaban mayor velocidad de transacciones y un precio menor, y para conseguirlo, tuvieron que comprometer la descentralización de la red, al hacer uso de PoSA (*Proof of staked Authority*). Esto hace que las decisiones sobre aprobaciones de bloques estén delegadas en un número reducido de usuarios que tengan propiedad de la mayor parte del Stake (cantidad de moneda, en este caso BNB ya que la criptomoneda de BSC es BNB, que se pone en juego un validador para validar un bloque y obtener un retorno por ello). Esto ha hecho que muchos propietarios minoritarios de BNB hagan uso de una estrategia muy popular conocida como *staking*, que consiste en otorgarle a uno de los grandes poseedores de BNB una cantidad adicional de BNB para que pueda así participar en la creacion de bloques y obtener la recompensa por ello. En retorno, el gran propietario de BNB le otorga a quien le ha prestado el BNB una parte de esa recompensa obtenida.

La BSC, es una de las redes principales junto con Polygon para el desarrollo de entornos NFT. Ambas son ordenes de magnitud más baratas que la red *Ethereum*. La razón por la que se ha optado por BSC en vez de Polygon es para poder cubrir ambas en este trabajo, obteniéndose así una mayor exposición.

Igual que se hizo con Polygon, se procede a la conexión con la BSC mediante Metamask en la nueva cuenta creada para este apartado; Programador.

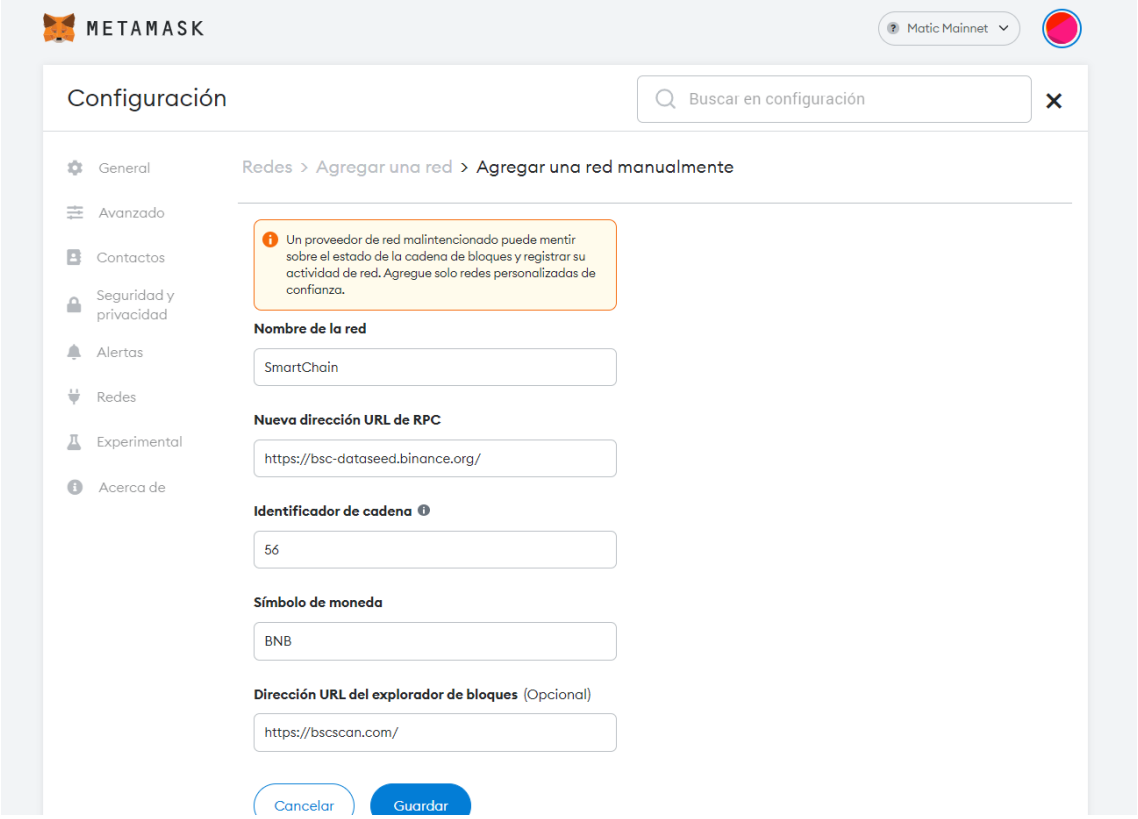


*Ilustración 18. Configuración de la BSC en Metamask*

Además, se obtienen 0.1 BNBs para interactuar con la red durante el despliegue del *Marketplace*.


*Ilustración 19 Saldo de 0.1 BNBs para la realización del apartado*

### *4.1.2.5 Pinata.cloud (IPFS)*

Antes de explicar que es Pinata.cloud, se comienza dando una breve explicación de IPFS.

IPFS (*InterPlanetary File Storage*) es un protocolo de almacenamiento descentralizado creado en 2015 por Juan Benet. La red IPFS está compuesta de nodos en los que se instancian clientes IPFS. Se pueden *guard*ar cualquier tipo de archivo (texto, música, video, imágenes, etc). A diferencia de http, los datos no están asociados a una localización física de un servidor (por ejemplo, cuando se accede a un contenido web se accede al mismo a través de un servidor físico, traduciendo la dirección IP de ese servidor físico a través de un servicio DNS) sino que están asociados a un hash. Si por ejemplo se sube una imagen a IPFS, para acceder a la misma se debe hacer uso del hash de esta imagen, es decir, un usuario pide que se le devuelva una imagen que tiene un hash determinado y la red IPFS lo devuelve.

Por otro lado, los ficheros IPFS se obtienen de varios nodos a la vez, para evitar que un solo node se sobrecargue enviando un fichero. Además, los nodos son incentivados a *guard*ar y compartir contenido mediante la criptomoneda FileCoin, que hace uso de Proof of Storage para recompensar a los nodos.

Pinata.Cloud proporciona a usuarios una interfaz sencilla para *guard*ar contenido en IPFS. Para ello, permite el uso de hasta 1GB de almacenamiento en IPFS gratuito, lo que será más que suficiente para la prueba de concepto que se hará en este apartado.

A través de pinata.cloud, se procederá a subir una serie de imágenes de los NFTs, lo cual proporciona un link IPFS. Tras eso, se completará el resto de los metadatos del NFT a través de un fichero JSON, done uno de los campos será el link IPFS a la imagen. Este fichero JSON se vuelve a subir a IPFS. Este fichero JSON por tanto tendrá su propio link IPFS que se utilizará en el proyecto.

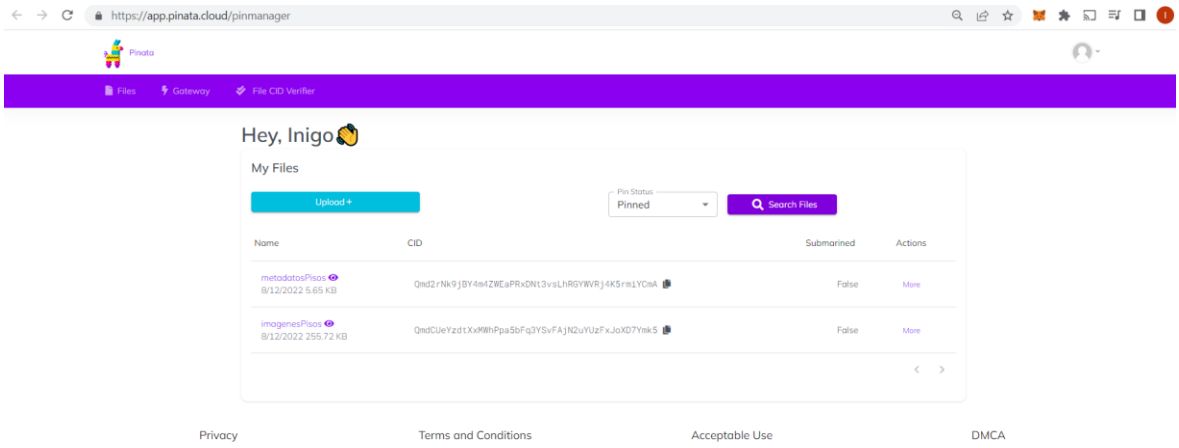


*Ilustración 20. Vista de la página Pinata.cloud*

## 4.2 ERC 721. EN DETALLE

A continuación se pasa a explicar los estándares para el contrato NFT (ERC 721) y para el contrato de *Marketplace* (ERC 1155). Se dedica un apartado a ambos ya que es la componente principal de este trabajo, y se omite el mismo nivel de detalle para las componentes más genéricas de *frontend* como puede ser Angular o Node.js. Independientemente de esto, en el siguiente apartado; Flujo Conceptual, se tratarán más en detalle estos apartados.

### *4.2.1 ERC 721*

En primer lugar, se explica lo que es un ERC. De la misma forma que se ha visto a la largo de la carrera con los RFC (*Request for Comments)*, un ERC (*Ethereum Request for Comments*) es una guía o estándar para la programación de *Smart Contract*s en Solidity. Existen varios ERCs. Por ejemplo, un desarrollador que desee programar un *Smart Contract* para la creacion de *token*s fungibles (suelen referirse como "*token*s"), puede seguir el estándar ERC 20 para programar dicho *Smart Contract*. De esta forma, podrá hacer uso de funciones, constructores, etc. Predefinidos, además de legitimar su proyecto y dotarlo de compatibilidad.

**¿Qué es?**: El ERC721 es un estándar para la programación de *Smart Contracts* de NFTs. En apartados anteriores se ha comentado ya la diferencia entre un NFT y un *token*, por lo que se omitirá dicha distinción. En enero de 2018, Dieter Shirley realizo un EIP (*Ethereum Improvement Proposal*) para estandarizar la producción de NFTs, convirtiéndose así en el primer estándar NFT de *Ethereum*. Por lo general, los NFTs tienen un solo dueño (*owner*), aunque están ganando en popularidad los NFTs con múltiples dueños [20]. Un contrato realizado con el estándar ERC721 puede mintear NFTs de la misma forma que un ERC20 puede mintear *token*s. Por tanto, ha de aclararse una concepción errónea común, ya que no se requiere un contrato por cada NFT, sino que con un solo contrato se pueden crear y *guard*ar varios NFTs.

Una vez empleado la funcionalidad de *minteo* de estos contratos, cabe preguntarse dónde se *guard*a la información de estos NFTs. Por supuesto, como mínimo un identificador del NFT debe de *guard*arse en el contrato. Sin embargo, más allá de ese ID, al cantidad de información que se desee *guard*ar en el contrato hará que los costes de operar con el mismo sean diferentes. Existen en esencia dos formas de *guard*ar los metadatos de los NFTs:

1. *On Chain*: *Guard*ar los datos de un NFT en el propio *Smart Contract* implica hacerlo en la cadena de bloques, y por tanto, escribir dicha información tiene un coste relativamente elevado (relativamente con la alternativa *off Chain*). Esta información es accesible de manera inmediate y sin coste a la hora de leer la *Blockchain* (ya que las funciones view no tienen coste alguno de gas). Sin embargo, si se deseasen crear un nuevo NFT o modificar uno existente (normalmente el proceso de modificación consiste en "quemar" el existente y crear uno nuevo) esto tiene un coste muy elevado de Gas por la información a escribir, que incluye normalmente una imagen con un coste de almacenamiento alto.
2. *Off Chain*: Dentro de la alternativa *Off Chain*, existen dos sub-alternativas a considerar:
    a. *Server*: La primera opción es *guard*ar la información en un servidor centralizado, ya sea un servidor local o remoto. Esto implica que cuando se desease cargar la información del NFT, el link apuntaría a dicho servidor y a

partir de ahí cargaría la información. Es una alternativa poco deseada, ya que centraliza dichos datos y permite a la entidad central que opera dicho servidor elimine, extravíe o modifique los mismos. En efecto, si lo que se *guard*a en un NFT es un link a un servidor, entonces el contenido del servidor puede ser modificado y el link no verse alterado.

b. *IPFS*: Ya se ha explicado en que consiste esta tecnología. Es una alternativa muy común empleada por parte de los creadores de NFTs, y la que se empleará en este proyecto Hace uso de la tecnología de almacenamiento descentralizada IPFS, lo que mantiene la filosofía de la descentralización en los NFTs. Además, en el *Smart Contract*, se *guard*a un puntero a los metadatos de un NFT concreto. Este puntero no es simplemente un link que apunta a una dirección de almacenamiento (ya que sino se podría modificar el contenido de aquello que se encuentre en dicha dirección), sino que el link incluye un CID, que hace la labor de un *hash* de los metadatos que se encuentran almacenados, haciendo así imposible modificar el contenido que se encuentra en el link, ya que si se cambiase el mismo, el hash variaría y por tanto el link cambiaría y ya no sería valido.

Por tanto, como se ha mencionado, se hará uso de IPFS para *guard*ar los metadatos de los NFTs que se creen.

A continuación, se explican cuáles son los principales elementos del contrato. En primer lugar, se ha de explicar que un contrato tiene una dirección para interactuar con el mismo. De la misma forma que si se desea transferir fondos de una cuenta a otra se hace mediante el uso de un *address*, con los contratos se opera de manera similar, mandando a la dirección pública del contrato una transferencia donde se indica que se desea hacer (por ejemplo, llamar a una función). Una vez comprendido como se interactúa con un contrato, se procede a explicar las principales funciones del ERC721 [21], focalizándose en aquellas empleadas durante el trabajo en el contrato realizado.

- **_safeMint(address to, uint256 *token*Id)**: Permite crear un *token* de manera seguro. Recibe como parámetros quien será el propietario de dicho NFT y cual será el identificador que tendrá este. Si dicho *token* ID existe, se revierte la operación, lo que conlleva un pequeño coste de gas.

- **_burn(address owner, uint256 *token*Id):** Esta función destruye un *token* existente, pasándole quien es su propietario y su identificador.

- ***token*URI(uint256 *token*Id):** Esta función devuelve la dirección donde se encuentran los metadatos de un NFT al recibir el identificador de dicho NFT. Normalmente, el identificador es un número que va incrementando de uno en uno cada vez que se crea un nuevo NFT. Por ello, se le puede indicar al contrato que se desea obtener la dirección donde se encuentran los metadatos para el NFT número 4. Una vez obtenida dicha dirección (que como se ha visto antes, puede estar en un servidor centralizado o en una estructura descentralizada), se pude acceder al contenido de la misma y operar con este (por ejemplo, si se desease mostrar)

- **_before*Token*Transfer(address from, address to, uint256 *token*Id):** Esta función debe ser llamada antes de la transferencia de cualquier NFT. Se considera como transferencia también la creacion de un NFT o la destrucción del mismo. Esta función verifica una serie de condiciones como que no se pasan valores nulos y cómo actuar en caso de que lo fuesen.

Existen además otras funciones que se han empleado en el contrato, pertenecientes a otros estándares como *ERC721Enumerable*. Dichas funciones y la interacción entre todas ellas se explicarán en el flujo conceptual.

## 4.3 JUSTIFICACIÓN DE LA TEMÁTICA. PISOS

Durante la prueba de concepto realizada en OpenSea, se ha hecho uso de obras de arte en las que se incluía una imagen. En el caso del *Marketplace*, se ha decidido modificar los NFTs, sustituyendo las obras de arte por inmuebles, en este caso pisos.

La razón por la que se ha decidido realizar esta modificación, es una razón principalmente personal. Un amigo ha comenzado un proyecto de emprendimiento en el que ofrece soluciones de realidad virtual para poder mostrar a un potencial comprador la pinta que tiene un piso o incluso la posibilidad de mostrar como quedaría un piso tras una reforma solicitada por el comprador. Estos son los servicios que se muestran en su página web [22].


*Ilustración 21. Servicios ofrecidos por HabitaXR*

Con la intención de buscar sinergias con el apasionante proyecto de los fundadores de HabitaXR, se ha decidido enfocar esta segunda parte al mismo sector. La idea es que, en un futuro, los HabitaXR también permita la posibilidad de comprar pisos para su arrendamiento a un conjunto de usuarios interesados. Para ello, estos pisos deben ser confiables y sus características deben ser persistentes en la red. El objetivo por tanto de esta sinergia seria hacer accesible la inversión en pisos a múltiples usuarios interesados sin tener que estar presente siquiera en el país donde se va a adquirir el piso.

En efecto, históricamente ha sido muy sencillo para un inversor poder exponerse a los retornos de inversión que proveen empresas alrededor de todo el mundo mediante la compra de acciones. Un inversor español podría perfectamente invertir en una empresa americana desde España. Sin embargo, es más difícil exponerse a los retornos que produce un inmueble en estados unidos, ya que haría falta realizar la compra en su totalidad (salvo si se usan ETFs de vivienda, pero estas son para un conjunto de viviendas y no una individual). Con la *token*izacion de los pisos a través de NFTs, se busca que inversores de todo el mundo puedan saber exactamente en que están invirtiendo (al tener acceso a todas las vistas de la vivienda mediante VR) y poder de esta forma comprar partes de la misma como si de una empresa se tratase. Por consiguiente, igual que una empresa, generaría flujos de caja (los pagos del alquiler) y también variaría su valor y por consiguiente el valor de las participaciones sobre esta vivienda. Además, mediante la *token*izacion en NFTs de estos pisos, se podrían realizar reglas programáticas sobre decisiones como que porcentaje de propietarios sería necesario para realizar una reforma, cuanto para realizar una venta de la totalidad de la vivienda, para aceptar o no a un nuevo inquilino, etc.

Todo lo anteriormente mencionado es de gran interés y potencial, pero escapa al desarrollo de este trabajo y se mantiene como un desarrollo futuro a partir de la prueba de concepto que se haga en la presente tesis.

Finalmente, hay que mencionar que al tratarse de una prueba de concepto, no se han incluido todos los campos que podrían haberse incluido con información mucho mas detallada de los pisos. Dicha información procedería incluirla en el futuro cuando se realice este proyecto de

manera completa. Los atributos que se han incluido en esta prueba de concepto (más allá de la descripción y la imagen) son: País en el que se encuentra el piso, ciudad en la que se encuentra el piso, dirección exacta del piso, planta en la que esta localizado, numero de metros cuadrados, numero de habitaciones y numero de aseos.

## 4.4 FLUJO CONCEPTUAL

En este apartado, se procede a explicar cual es el flujo de eventos que posibilitan el funcionamiento de este *Marketplace*. Independientemente del código o interfaz empleadas, se dará una visión a alto nivel de como interactúan las diferentes tecnologías mencionadas y en que orden. Para facilitar su comprensión, se complementará dicho flujo con capturas de pantalla del *Marketplace* construido. Hay que mencionar que la componente visual no se ha desarrollado en profundidad, ya que lo principal de esta prueba de concepto es la construcción del *Marketplace* y el uso de *Smart Contract*s.

Existen esencialmente dos elementos que interactúan en este *Marketplace*; el frontend de la aplicación (código que se ejecuta en el navegador del usuario) y la componente *Blockchain* (*Smart Contract*s en la red *Blockchain* que se emplee, en este caso la BSC).

Al entrar en la dirección http://localhost:4200, Angular procede a mostrar un *guard*a. Un *guard*a (*guard*) es sencillamente una vista variable en Angular, que mostrara páginas diferentes en base a unas condiciones. En este caso, la condición para permitir el acceso al *Marketplace* es que se haya iniciado sesión con Metamask en la página. Si no es el caso, no se debe permitir al usuario acceder al *Marketplace*, ya que no podría operar en el mismo. Esta es la vista que se muestra:

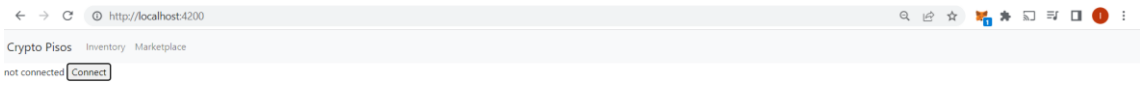


*Ilustración 22. Componente visual del Guard*

Tras hacer uso del botón *Connect*, esto inicializa la extensión de Metamask y pide que se conecte a la página. Para comprobar que el usuario es quien dice ser, se pide firmar mediante la clave privada un *challenge*, el cual es descifrado con la clave publica y puede de esta forma comprobarse que el usuario es quien dice ser. Se entrará en algo más de detalle sobre el proceso de esta validación y como se hace en Angular en el siguiente apartado. Tras comprobarse la validación, se le muestra al usuario sus pisos.

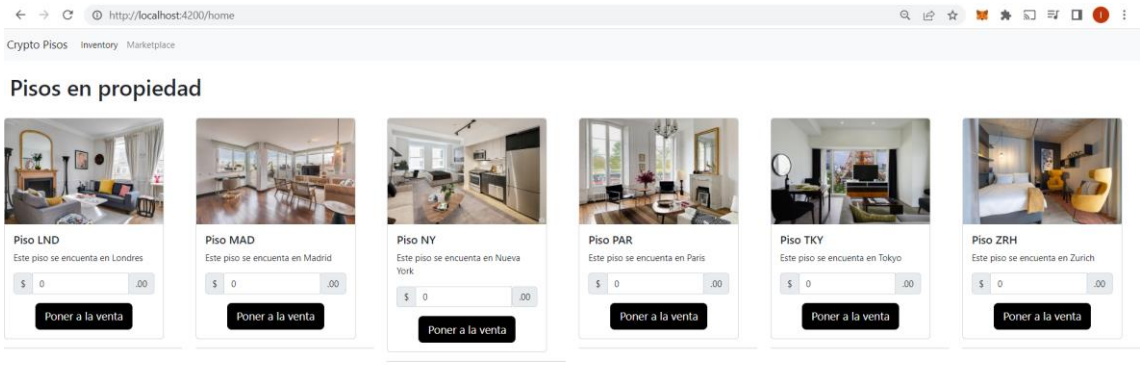


*Ilustración 23. Muestra de los pisos en propiedad para la cuenta Progrmador*

Como puede verse, el usuario tiene en propiedad pisos en las ciudades de Londres, Madrid, Nueva York, Paris, Tokyo (hubo un error al subir a la BSC el piso de Tokyo y se escribió el nombre incorrectamente, llamándose por error "Piso NY", pero con el fin de transmitir las localidades geográficas en la ilustración, se ha cambiado en el HTML) y Zurich.

Lo que está sucediendo por detrás es que el frontend está accediendo a la información del contrato NFT e indicándole, mediante Metamask, que dirección tiene el usuario para así poder mostrar los NFTs de este.

El usuario puede ver los pisos que tiene y a continuación puede poner a la venta uno de ellos. Se puso a la venta el piso de Londres en su momento por lo que se va a mostrar como se pone a la venta el piso de Madrid. Para ello, se indica que se desea poner el piso a la venta y el precio al que se desea hacer, en este caso 285k.

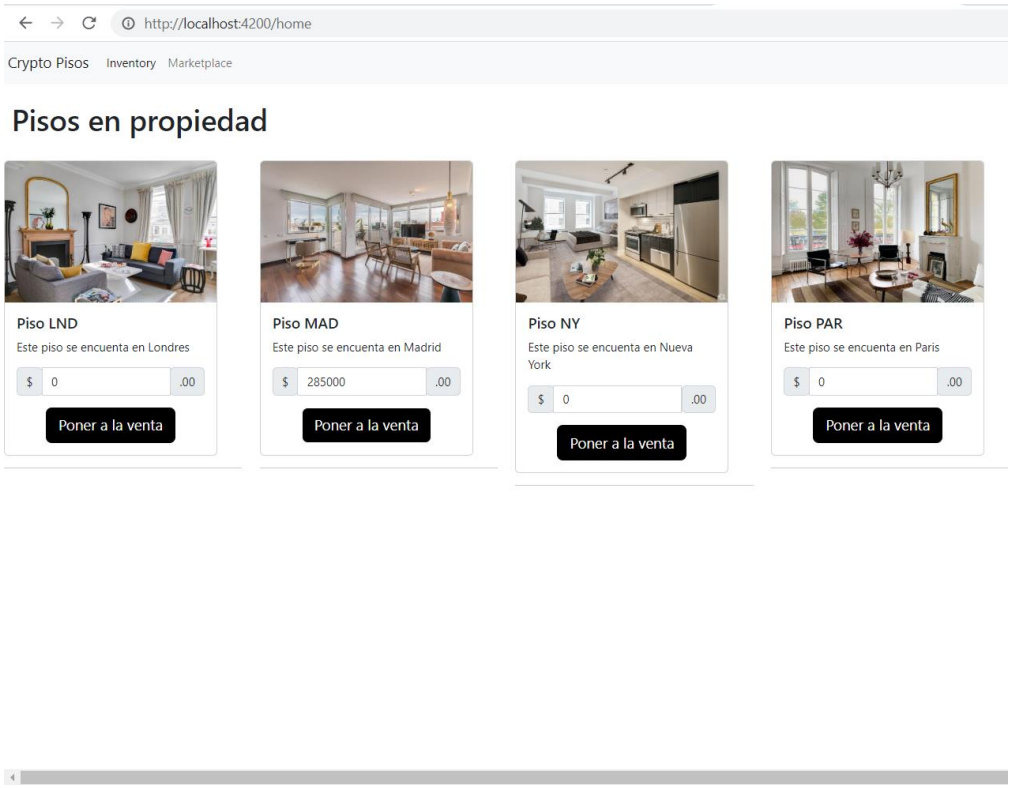


*Ilustración 24. Puesta a la venta del piso de Madrid*

A continuación, se pincha en "Poner a la venta" y esto hace que se ejecute la función toggleForSale() mediante la cual se pide autorización al usuario mediante Metamask para poner dicho NFT a la venta. Para ello, el frontend hace uso del ABI (Application Binary Interface) del contrato *Marketplace* y le indica a dicho *Smart Contract* que ponga a la venta este NFT.

El NFT en cuestión que se esta poniendo a la venta se encuentra en el Contrato NFT, con un link a un archivo IPFS. Dicho archivo es un JSON donde se incluyen atributos del Piso, además de un link a una imagen de los mismos. Esto permite que el frontend pueda, mediante una función view(), ver el contenido de este JSON a partir de el link y sacar tanto los atributos del piso como la imagen del mismo y así poder mostrarlos por pantalla. Para subir estos archivos a IPFS se ha hecho uso de *Piñatacloud*.

Por tanto, el contrato *Marketplace* recibe tanto el address como el precio al que se desea listar el piso en propiedad, firmado mediante la clave privada del usuario. Esto permite realizar una transacción en la BSC y que se vea reflejado en el *Marketplace*. La petición de firma se completa mediante Metamask, y tiene un coste en Gas.

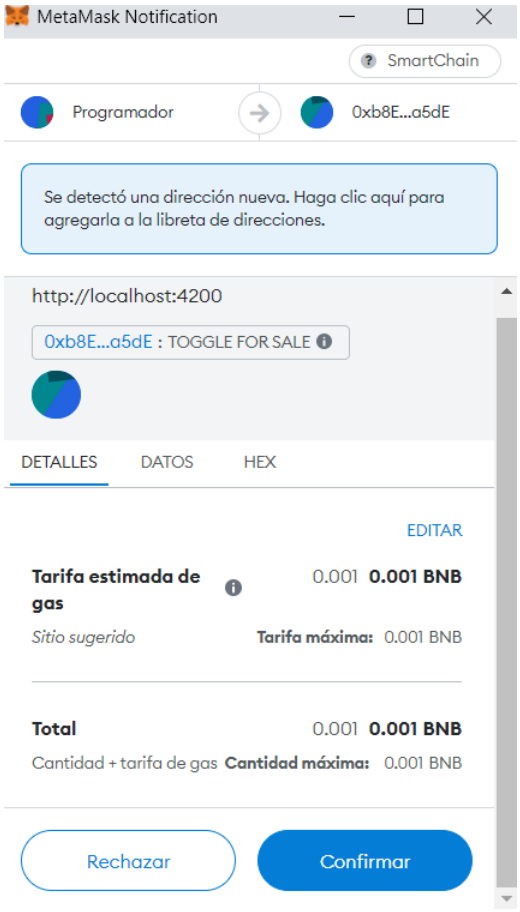


*Ilustración 25. Muestra de la transacción realizada con Metamask para la puesta a la venta del piso*

Se accede al contenido de dicho *Marketplace* mediante una función View, para ver el piso listado.

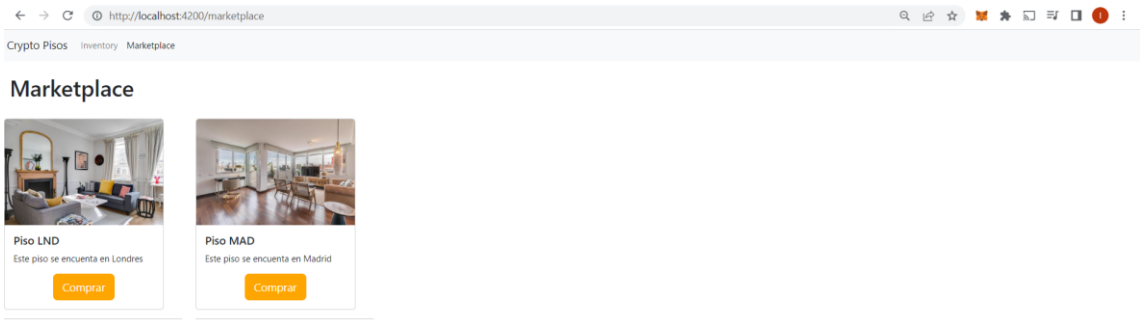


*Ilustración 26. Markeplace mostrando la adicion del piso de Madrid*

Cabe destacar que, a diferencia de cómo se opera en OpenSea, en este *Marketplace* que se ha creado si que se sube directamente a la BSC cualquier listado con cierto precio. Es por ello que se refleja en un coste y que, si se desease modificar este, habría que volver a hacer uso de Gas, mientras que en OpenSea la puesta a la venta no tenía ningún coste debido a que se subía a un servidor local y solamente se subía a la red *Blockchain* cuando había un *match* entre orden de venta y compra por cierto precio.

Finalmente, hay que mencionar también que este proceso de puesta en funcionamiento del *Marketplace* se hizo mediante la red de prueba Ganache, y que lo que se muestra ahora es habiéndolo hecho en la BSC. Esto se especificará mas en el siguiente apartado.

## 4.5 DOCUMENTACIÓN DE FUNCIONAMIENTO DEL *MARKETPLACE*

En este apartado se procede a realizar un análisis mas detallado de los aspectos programáticos que se han seguido.

En primer lugar, la estructura de carpetas seguidas es la siguiente:

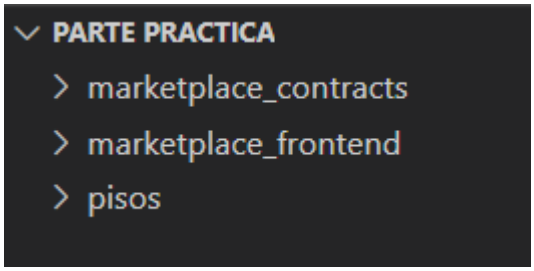


*Ilustración 27. Componente jerárquica de proyecto al más alto nivel*

En esencia, son 3 carpetas las cuales contienen cada una un aspecto del *Marketplace*. Por consiguiente, se procede a realizar la explicación de este apartado explicando en primer lugar el *frontend* realizado con Angular, en segundo lugar la carpeta Pisos y finalmente la carpeta que incluye a los contratos. Esta ultima tiene dos contratos principales; contrato NFT y contrato *Marketplace*. La forma en la que se ha ido construyendo la aplicación ha sido progresiva, por lo que se ha probado primero la funcionalidad únicamente entre Angular y el contrato NFT y después se ha incorporado el contrato *Marketplace*.

### *4.5.1 FRONTEND DEL MARKETPLACE*

Como se ha mencionado anteriormente, el frontend de esta aplicación ha sido realizado mediante Angular. Esta tecnología se ha explicado de manera aislada en apartados anteriores. Sin embargo, en este apartado se tratará de dar una visión mas cercana a la ejecución del proyecto. La composición de las carpetas es la siguiente:

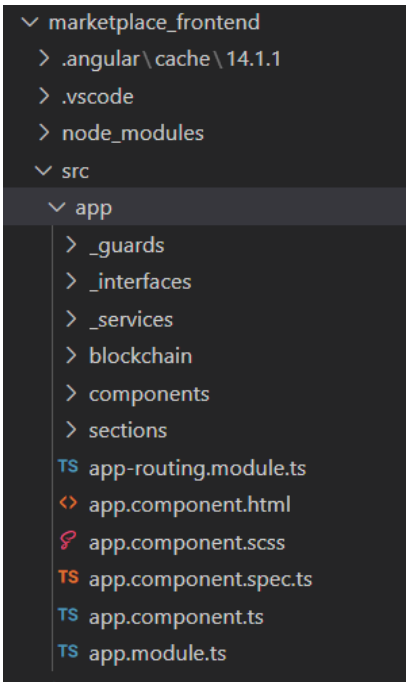


*Ilustración 28. Elementos principales del frontend*

Además de estas carpetas, hay otras que no se incluyen en la captura ya que no son el foco de esta explicación. Nótese que existen 3 tipos principales de archivos en Angular; .html, .scss y .ts.

De forma simplificada, los archivos .html y .scss se encargan del aspecto mas visual de la aplicación mientras que los archivos .ts contienen la lógica (son archivos TypeScript explicados en secciones anteriores). También existen archivos .spec para motivos de testing pero sobre los que no se profundizará.

Se procede a explicar las principales carpetas dentro de la carpeta seleccionada en la imagen "app".

1. Carpeta *_guard*s: Esta carpeta incluye los *guard*as que se han mencionado anteriormente. Contiene archivos visuales (.html y .scss) y .lógico ts. La lógica es muy sencilla, se crea en el html dos vistas, y se indica que en función de si se esta o no conectado mediante Metamask, se muestre una u otra vista. Esta validación se produce en el archivo .ts mediante una suscripción a

$is*Connect*ed, y en caso de estarlo se inicializan los servicios, los cuales se explican a continuación. Además, se cambia el campo this.isAllowed a true para poder mostrar la vista correcta en html.

2. Carpeta _interfaces: Esta carpeta simplemente contiene la definición del contrato piso en función de sus argumentos.
3. Carpeta _services: Un *service* otorga funcionalidad que puede ser luego empleada en otras componentes del proyecto. Funcionalidad clave reutilizable como puede ser coger el piso número 3 del contrato se puede programar en funciones para que se más accesible su uso. Existen 3 archivos en esta carpeta: *Marketplace*.service.ts, piso.service.ts y web3.service.ts. Estos archivos encapsulan por tanto funcionalidad. Cuando se verifica la conexión, se accede a pisoservice.
4. Carpeta *Blockchain*: Contiene los archivos ABI de los contratos realizados para este proyecto. Estos archivos se *guard*an en esta carpeta una vez se compila un contrato, y es la forma en la que interactuar con el mismo.
5. Carpeta Components: Esta carpeta contiene las componentes visuales que sirven como los ladrillos de la aplicación. Angular funciona como una jerarquía de componentes. La más elevada es el archivo que se muestra en la captura; app.components que cuenta con una parte lógica .ts y una visual .html.. Las componentes que se encuentran en esta carpeta son una componente de tipo piso y otra de tipo *Marketplace*. Esto lo que permitirá es que cuando se deseen mostrar varios pisos en una misma página, se pueda reutilizar esta componente piso para mostrar varios en una misma pagina cambiando los parámetros. Por tanto, cada una de estas pequeñas celdas que muestran un piso son una componente en Angular:

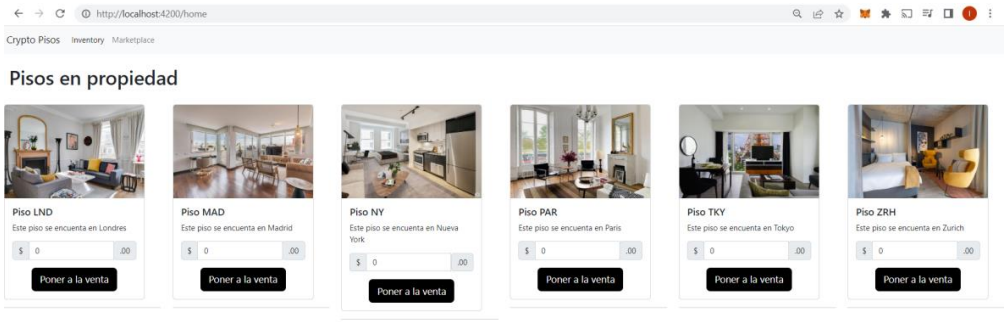


*Ilustración 29. Muestra de los pisos en propiedad*

6. Carpeta sections: Esta carpeta contiene las vistas de los paths /home y /*Marketplace*. Estas vistas, se construyen a partir de los bloques de vista anteriores según sea necesario. Por ejemplo, la vista de la imagen anterior, que es la vista /home, muestra por un lado el texto de "Pisos en propiedad" y por otro realiza un bucle dentro del html en función de cuantos índices haya, mostrando así un total de 6 pisos.

### *4.5.2 CARPETA PISOS*

Esta carpeta contiene tanto las imágenes como los metadatos en archivos JSON.

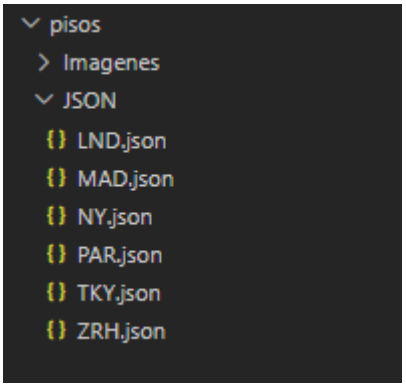


*Ilustración 30. Muestra de los archivos JSON de los pisos*

Estos archivos contienen la información necesaria para crear 6 NFTs de tipo piso. Esta información, se ha subido a IPFS. Para ello, se ha hecho uso de piñata.cloud, que es una forma sencilla de subir archivos a IPFS. En primer lugar, se han subido las imágenes, obteniéndose así un link IPFS a dichas imágenes. Tras eso, se ha construido un archivo .JSON para cada uno de los NFTS. Estos archivos tienen varios campos, entre los que se encuentra el campo imagen, la cual toma por valor el link obtenido anteriormente con la dirección a la imagen en IPFS. Todos estos datos JSON se suben de nuevo a IPFS, obteniéndose un link, el cual será empleado para mintear NFTs, siendo ese link lo único que se *guard*e en el contrato. Se muestra a continuación el formato que aparece en pinata.cloud.

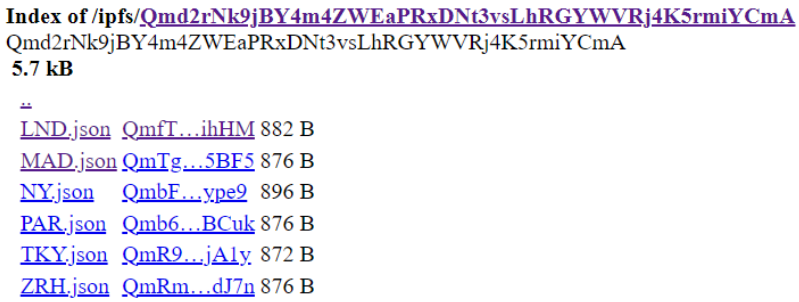


*Ilustración 31. Vista de pinata.cloud donde residen los JSON de los pisos*

### *4.5.3 CARPETA CONTRACTS*

En este apartado, se procede a centrarse en los archivos en la carpeta de los contratos, es decir, en la componente relacionada con *Blockchain*. Los principales elementos que amerita explicar se muestran en la siguiente captura:

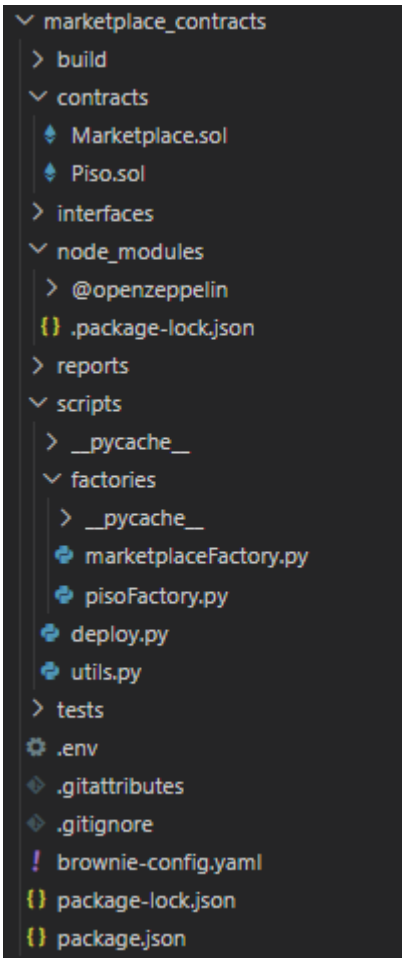


*Ilustración 32. Estructura de los elementos más importantes de marketplace_contracts*

Nótese que para crear esta estructura de carpetas, se ha hecho uso del comando del siguiente comando de brownie:

*Brownie init*

De la misma forma que en el apartado anterior, se pondrá el foco en los archivos más relevantes:

1. Carpeta build: Aquí se encuentran los archivos ABI
2. Carpeta *Contracts*: Aquí se encuentran los contratos. Los detalles sobre los mismos se comentan en el apartado "Incorporación del contrato ERC721"

3. Carpeta node_modules: Aquí destacar el modulo @openzeppellin, que se usará en los contratos y cuya explicación se realizará en el apartado "Incorporación del contrato ERC721".
4. Carpeta scripts: En esta carpeta hay que destacar la subcarpeta *factories*. El concepto de fabrica se utiliza tanto para pisos como markeplace. Una fabrica otorga funcionalidad mediante scripts en Python. Se pueden crear funciones en Python que interactúan con el *Smart Contract*. Esto permite encapsular la funcionalidad y hacerla mas accesible a través de Python para otras secciones del código. Por ejemplo, si se desea mintear un piso, se llamará a la función mintPiso de esta fábrica, la cual interactúa con la función safeMint del contrato Piso.sol. Otro caso clave de funcionalidad de la fabirca piso (*pisoFactory.py*), es el constructor (*_initi_*), el cual diferencia entre dos escenarios; por un lado, si la red en la que se esta operando es en la red de prueba Ganache, entonces despliega el contrato de cero, pero si en vez de estar en la red de testing se esta en la red *Blockchain* (en este caso la BSC), entonces carga el contrato desde su dirección. Nótese que según esto, el despliegue inicial del contrato en la BSC debe de hacerse "a mano". Este ha sido el caso como se documenta mas adelante. Finalmente, hay que comentar que en estas fabricas se diferencia entre funciones de tipo transacción y de tipo vista. Las de tipo transacción tienen un coste de Gas, al realizar modificaciones en la red *Blockchain*, mientras que las funciones de tipo vista (*getters*) no requieren Gas, al ser una simple lectura de la BSC. Finalmente, hay que mencionar que el archivo Python desde el cual se hacen todas las llamadas es el archivo deploy.py. Este archivo actúa como el "main" , y es el que llama a pisoFactory para la creacion de pisos.
5. Archivo .env: Aquí se *guard*an (en texto plano) las claves privadas de la cuenta que se ha creado para esta prueba de concepto así como la dirección del contrato Piso.sol en la BSC, que se emplea en la inicialización mencionada en el apartado anterior.
6. Archivo brownie-config.yaml: Este archivo contiene también lo mencionado en el archivo .env a modo de variable, ya que es el archivo consultado por el resto de elementos del proyecto que requieran estas variables. Además, contiene la

dependencia de OpenZepellin, la cual es un *framework* de código abierto que cuenta con iberia para contratos.

Una vez explicada la estructura, se pasa a documentar el proceso seguido para incorporar en primer lugar el contrato ERC721 y en segundo lugar el contrato *Marketplace*.

#### *4.5.3.1 Incorporación del contrato ERC721*

En este primer apartado se explica como se han integrado Angular y el contrato ERC721, es decir el contrato Piso.sol.

En primer lugar, se centrará la explicación en los campos incluidos en el contrato Piso.sol. Adicionalmente a los campos explicados en el apartado de "ERC 721. En detalle", se mencionarán los contratos que se heredan desde OpenZepellin.

OpenZepellin posee varios contratos a importar. Los contratos que se importarán para el contrato Piso.sol son:

- ERC721: Este contrato ya ha sido explicado en secciones anteriores
- ERC721Enumerable: Se trata de una extensión de ERC721 que permite enumerar todos los *token* Ids en el contrato, además de aquellos que pertenecen a una determinada cuenta. Si el contrato no fuese enumerable, entonces no se podría saber todos los NFTs que posee una cuenta a menos que se tuviese el *token*URI. Por tanto, para poder listar los NFTs de una cuenta, se requiere que el contrato sea ERC721Enumerable.
- ERC721URIStorage: Esta extensión indica que el almacenamiento de los NFTs se realizara con una URI y no introduciendo los datos directamente en la red *Blockchain*.
- Ownable: Permite que los contratos tengan un dueño. Esto permite que ciertas funciones del contrato sean únicamente accesible a dicho propietario, quien también podrá hacer transferencia de la propiedad de un contrato.

Una vez se han creado las estructuras de carpetas frontend y *Blockchain* (con Angular y Brownie), y una vez se han programado los contratos y el *frontend*, se procede a desplegar el contrato en la red de prueba Ganache. Nótese que no se entrará en detalle de el código realizado en esta memoria, sino que la estructura de archivos será incluida como anexo, manteniéndose los aspectos mas relevantes de dichos ficheros comentados en esta memoria.

El despliegue en ganache se realiza mediante el siguiente comando:

*Brownie run scripts/deploy*

Al no haberse indicado ninguna red como parámetro final, se despliega en la red de prueba Ganache.

Se realiza por tanto una llamada al deploy.py, que como se ha mencionado antes, al detectar que esta en una red de prueba, despliega el contrato de cero.

Se muestra el resultado de la ejecución de dicho comando:

```
C:\Users\Iñigo\Dropbox\Iñigo\ICAI\Master\TFM Teleco\Proyecto NFT\Parte practica\marketplace_contracts>brownie run scripts/deploy
INFORMACIÓN: no se pudo encontrar ningún archivo para los patrones dados.
Brownie v1.19.1 - Python development framework for Ethereum

MarketplaceContractsProject is the active project.

Launching 'ganache-cli.cmd --port 8545 --gasLimit 12000000 --accounts 10 --hardfork istanbul --mnemonic brownie'...

Running 'scripts\deploy.py::main'...
Transaction sent: 0xc4cf4294a3341912276fc2d05ab0b3cf7984c868c899b5e925e5cf9765982415
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  Piso.constructor confirmed   Block: 1   Gas used: 1671588 (13.93%)
  Piso deployed at: 0x3194cBDC3dbcd3E11a07892e7bA5c3394048Cc87

PisoContract deployed at:  0x3194cBDC3dbcd3E11a07892e7bA5c3394048Cc87
Transaction sent: 0xcc2cbf9d05e56b5e5d2d6cb6539ee428d0ca3af6da0893e2729c178306be8534
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  Piso.safeMint confirmed   Block: 2   Gas used: 200200 (1.67%)

Url from piso 0 is https://gateway.pinata.cloud/ipfs/Qmd2rNk9jBY4m4ZWEaPRxDNt3vsLhRGYWVRj4K5rmiYCmA/LND.json
Terminating local RPC client...
```

*Ilustración 33. Despliegue en la red de prueba Ganache*

La transacción en la red de prueba se ha realizado con éxito. Para comprobarlo, no solamente se ha desplegado el contrato, sino que también se ha minteado un piso (piso0). Se imprime por pantalla la url del piso 0, la cual apunta a IPFS a través del servicio de pinata.cloud visto anteriormente.

Una vez se ha desplegado exitosamente en la red de prueba, se procede a hacer el despliegue del contrato en la red *Blockchain* BSC. Esto tiene unos costes de Gas por lo que se adquiere anteriormente dicho criptoactivo para poder realizar la prueba de concepto. Se omite el proceso realizado para dicha adquisición.

El despliegue en la BSC va precedido por la comprobación de que Brownie soporta dicha red. Para ello, se ejecuta el siguiente comando:

*Brownie networks list*

Una vez que se comprueba que Brownie soporta la red BSC, se procede a desplegar el contrato mediante el comando:

*Brownie run scripts/deploy –network bsc-main*

Anteriormente, como ya se ha mencionado, se ha tenido que subir el contrato y se han *guard*ado en el archivo .env mencionado anteriormente la clave privada de la cuenta y la dirección del contrato, por lo que cuando se inicialice, al ver que se esta en la red BSC y no la de prueba, cargara dicha dirección del contrato.

```
1  // ============== PRIVATE KEYS ===============
2  export PRIVATE_KEY = 33d285d2359e7207d8ab84c08cae6738d99969[redacted]2f74
3
4  // ============== CONTRACT ADDRESSES ===============
5  export pisoContractAddress = 0x8Bf3d7c06162F53005951C89ce23c9D8EfCB36e5
```

*Ilustración 34. Archivo .env*

Nótese que se ha borrado parte de la clave privada por motivos de seguridad. Tras esto, se realiza el minteo de los 6 pisos.

```
C:\Users\Iñigo\Dropbox\Iñigo\ICAI\Master\TFM Teleco\Proyecto NFT\Parte practica\marketplace_contracts>brownie run scripts/deploy --newtork bsc-main
INFORMACIÓN: no se pudo encontrar ningún archivo para los patrones dados.
Brownie v1.19.1 - Python development framework for Ethereum

NB: Corrected --newtork to --network
Generating interface ABIs...
MarketplaceContractsProject is the active project.

Running 'scripts\deploy.py::main'...
Transaction sent: 0xe545cecd0f2fff8dd95de8b25c60e0e324ae75b1168691160e357657c6a85fbf
  Gas price: 5.0 gwei   Gas limit: 2000000   Nonce: 1
  Piso.safeMint confirmed   Block: 20390159   Gas used: 200200 (10.01%)

Transaction sent: 0x1f73313f03bcbf0017531a719a86bc036ba913ae637807e549fc46127461f3cc
  Gas price: 5.0 gwei   Gas limit: 2000000   Nonce: 2
  Piso.safeMint confirmed   Block: 20390161   Gas used: 232000 (11.60%)

Transaction sent: 0xd19657f980d0da0195ed94ec348f44a26823eee8dfa3a62814b70ec7f8b6b1e7
  Gas price: 5.0 gwei   Gas limit: 2000000   Nonce: 3
  Piso.safeMint confirmed   Block: 20390163   Gas used: 231988 (11.60%)

Transaction sent: 0xfc540c021ec329a88fddd140cacf04ea7370bae76904606e1725b3461df5fbd3
  Gas price: 5.0 gwei   Gas limit: 2000000   Nonce: 4
  Piso.safeMint confirmed   Block: 20390165   Gas used: 232000 (11.60%)

Transaction sent: 0x4d460a3a7907aa36a7552de14afe5ec11a939b5954133688c4518c3883f59560

Url from piso 0 is https://gateway.pinata.cloud/ipfs/Qmd2rNk9jBY4m4ZWEaPRxDNt3vsLhRGYWVRj4K5rmiYCmA/LND.json
Url from piso 1 is https://gateway.pinata.cloud/ipfs/Qmd2rNk9jBY4m4ZWEaPRxDNt3vsLhRGYWVRj4K5rmiYCmA/MAD.json
Url from piso 2 is https://gateway.pinata.cloud/ipfs/Qmd2rNk9jBY4m4ZWEaPRxDNt3vsLhRGYWVRj4K5rmiYCmA/NY.json
Url from piso 3 is https://gateway.pinata.cloud/ipfs/Qmd2rNk9jBY4m4ZWEaPRxDNt3vsLhRGYWVRj4K5rmiYCmA/PAR.json
Url from piso 4 is https://gateway.pinata.cloud/ipfs/Qmd2rNk9jBY4m4ZWEaPRxDNt3vsLhRGYWVRj4K5rmiYCmA/TKY.json
Url from piso 5 is https://gateway.pinata.cloud/ipfs/Qmd2rNk9jBY4m4ZWEaPRxDNt3vsLhRGYWVRj4K5rmiYCmA/ZRH.json

C:\Users\Iñigo\Dropbox\Iñigo\ICAI\Master\TFM Teleco\Proyecto NFT\Parte practica\marketplace_contracts>
```

*Ilustración 35. Despliegue en la BSC*

De la misma forma que se hizo con la red Ganache, se imprimen por pantalla las direcciones IPFS de dichos NFTs, comprobando de esta forma que ha funcionado correctamente.

El siguiente paso consiste en mostrar dichos NFTs en la dirección local. Para ello, Angular deberá interactuar con los *Smart Contracts*. Los NFTs que se han creado, se han creado todos con el propietario de la cuenta de "Programador" reservada para esta prueba de concepto. Por consiguiente, los 6 pisos tienen como propietario a dicha cuenta.

Cuando se accede a la pagina web construida con Angular, lo primero que se peticiona al usuario es que se conecte mediante Metamask. Una vez esto se ha hecho, se deben desplegar los NFTs (pisos) que tiene el usuario en su posesión. La lógica detrás de este proceso es la siguiente:

Un propietario puede tener varios NFTs, pero un NFT solo puede tener un propietario. Para poder mostrar todos los NFTs de un propietario, no se puede hacer uso de una función que haga esto, ya que en solidity no existen este tipo de bucles. Por tanto ¿Cómo se puede conseguir el objetivo de mostrar todos los NFTs de un usuario?

Para poder mostrar todos los NFTs de un usuario, se debe conocer su dirección y cuantos tiene, para poder de esta forma iterar llamadas al contrato y desplegar todos los NFTs que esa dirección posea.

Por consiguiente, se llama a la función balanceOf(owner), la cual recibe como parámetro la dirección de un determinado propietario y devuelve el número de *token*s no fungibles que este propietario posee. Por tanto, ya se tiene el número total de *token*s.

A continuación, se hace uso del hecho de que dicho contrato es Enumerable. Esto es muy importante, ya que implica que se le asocia un identificador numérico a cada NFT, comenzando desde 0 y siendo este incrementado cada vez que se mintea un nuevo NFT.

Esto es muy importante, ya que si no fuese enumerable, la única forma de peticionar un NFT seria peticionando su URI. Por tanto, solamente si se conociese esta URI podría accederse a los metadatos del NFT. Al ser Enumerable, es posible hacer una llamada a un endpoint que reciba la dirección y el índice, el cual devuelve el *token*URI.

Todo esto se hace desde la componente de Angular home.component.ts, la cual llama al piso Service, quien llama a la función getBalanceOf() averiguándose así el número de pisos. Ahora en la componente homecomponent.html, una vez conocido el número de NFTs, se procede a iterarlos. Este proceso es totalmente transparente para el usuario.

Tras iterar los índices se devuelven las *token*URIs de dichos NFTs. Únicamente existe un endopoint al que, mandándole la *token*URI, devuelve el contenido de dicho NFT. Pero como ya se ha realizado la iteración dado una dirección y un numero de índices conocidos, se pueden obtener de cada NFT los *token*URIs y a partir de esos *token*URIs, se puede obtener el contenido de los NFTs.

Una vez obtenidos todos los pisos y su contenido, se hace uso de la potencia de las componentes de Angular para, mediante estas llamadas nesteadas, desplegar todas las componentes en la página.

```
ngOnInit(): void {
  console.log("PISO INDEX:" + this.pisoIndex)
  this.pisoService.getPisoByIndex(this.pisoIndex).then((pisoId:number) =>
    this.pisoService.getPisoURL(pisoId).then((url:string) =>{
      this.pisoUrl = url
      this.http.get<Piso>(url).subscribe((piso) => {
        this.piso = piso;
        console.log(this.piso.image)
        this.loadIsApprove()
        this.loadIsInMarketplace()
      })
    }))
```

*Ilustración 36. Vista de la llamada nesteada ngOnInit()*

Primero se obtiene la URI del piso dado un índice y tras eso se obtiene el contenido conocida la URI. Una vez se tiene el contenido, al tratarse de un link a IPFS, se realiza una llamada mediante http para obtener dichos metadatos. El contenido devuelto se encapsula en un objeto de tipo piso. Una vez se ha encapsulado este piso, es cuestión de mostrar su contenido por pantalla.

### *4.5.3.2 Incorporación de Marketplace*

Una vez se ha conseguido listar los pisos de cierto propietario mediante Angular, se procede a construir el contrato *Marketplace*. De la misma forma que se procedió con el contrato anterior, se comienza por una primera prueba mediante Ganache de dicho contrato y a continuación se despliega en la BSC.

El contrato *Marketplace* únicamente es Ownable, y no se hace uso de ningún estándar adicional. Si bien es cierto que existe un estándar ERC 1155 para los *Marketplace*s, en este caso no se va a hacer uso del mismo. Nótese que, de la misma forma que el contrato piso podrá contener varios NFTs de tipo piso, el contrato *Marketplace* podrá construir también varios *Marketplace*s. Esto se ve reflejado en la estructura de *arrays* que se define al inicio del contrato.

La forma de construirlo de todas formas es muy similar. A continuación se procede a mencionar el flujo de este contrato, mencionando también las funciones que se emplean en el mismo.

En primer lugar, se crea una variable *pisoAddress*. Dicha variable *guard*ará la dirección del contrato *piso.sol* que se creó en el apartado anterior. Esto por supuesto es necesario, ya que si se va a tratar de un *Marketplace* de pisos, este contrato *Marketplace* debe poder tener acceso a dichos NFTs, lo que implica que debe tener acceso a la dirección del piso.

- La función status, dentro del contrato *Marketplace*.sol, sirve para saber si un NFT está o no a la venta. Esta función de si lo está o no devuelve *true* o *false*. Se puede modificar también el status mediante el setter *setStatus*.
- La función Price devuelve el precio del NFT listado. Nótese que si un NFT no esta listado (es decir, su status es false), se devuelve el valor 0. Se puede modificar el valor mediante el setter *setPrice*.
- La función *ToggleforSale* es una función para poner el NFT a la venta. Recibe el NFT que se desee listar y su precio. Lo primero que hace es comprobar si el NFT que se desea listar es propiedad de aquel agente que realice la transacción (esto se hace mediante un require). Tras eso, se comprueba si el contrato tiene permisos sobre los NFTs, ya que una vez que este a la venta el contrato debe poder transferir la propiedad de dicho NFT. Esto se hace llamando al contrato NFT con *isApproveForAll*, por lo que tiene que ver si ese *address* ha autorizado al contrato *Marketplace* a vender los NFTs.

Tras entender esto, se hace uso de *MarketplaceFactory* es un script como lo era *pisoFactory* para interactuar con el contrato. En deploy.py (el equivalente al fichero *main*) se incluye una función *testMarketplaceContract*() para realizar el testeo en la red Ganache.

La función *testMarketplaceContract*() primero despliega una *pisoFactory*, ya que necesitan NFTs para el *Marketplace*. Se realiza un *minteo* de 2 de ellos de prueba.

Tras esto, se despliega el contrato *Marketplace* y luego se llama a *setpisoAddress*. El contrato *Marketplace* necesita la dirección del contrato piso para interactuar con él. En vez de introducir este valor directamente, se construye este setter para poder pasárselo como una transacción al contrato *Marketplace*, por lo que se crea una variable para el mismo. La forma en la que el contrato *Marketplace* conocerá la dirección del contrato *piso.sol* es mediante una transacción a dicho setter.

El código resultante de dicha función es el siguiente:

```
def testMarketplaceContract():
    to = utils.get_account()
    pisoFactory = PisoFactory()
    pisoFactory.mintPiso(to,"sss")
    pisoFactory.mintPiso(to,"sss")
    marketplaceFactory = MarketplaceFactory()
    marketplaceFactory.contract.setPisoAddress(pisoFactory.contract.address)
    pisoFactory.approveAll(marketplaceFactory.contract.address)
    print(pisoFactory.contract.ownerOf(0), to)
    marketplaceFactory.contract.toggleForSale(0,10)
    marketplaceFactory.contract.buyPiso(0, {"from": to, "gas_limit": 2000000, "allow_revert": True, "value": 11})
```

*Ilustración 37. Vista de la funcion testMarketplaceContract()*

Se pone a la venta el NFT numero 0 por 10 BNB, y a continuación se desea comprarlo. Para ello, se requiere un valor superior a 10BnB, ya que hay que por un lado pagar el precio y por otro el gas para realizar la transacción.

En la versión desplegada en la BSC, se ha indicado el precio a 1 BNB ya que únicamente se puede operar con unidades, por lo que se omitirá el proceso de compra (el proceso de puesta a la venta ya ha sido explicado con anterioridad) al ser muy elevado el coste.

Tras haberlo testeado en la red de prueba, se pasa a hacerlo en la red BSC. Para ello se hace uso de la función en deploy.py de *deployMarketplace*(), donde sí que se ha indicado de manera explicita la dirección del contrato Piso.

Tras haber comprendido esto, se pasa a explicar la interacción con Angular de la misma forma que se hizo en el apartado anterior.

Se crea en la carpeta de frontend *Marketplace*. *Marketplacecomponents*.ts. Lo primero es conseguir el *totalmarketsupply*. Esta sección averiguar el número de NFTs que hay a la venta, por lo que se llama a totalmarketsuply para conocer este dato. Tras eso, se procede a iterarlos de la misma forma que iteraba el inventario de NFTs. Para ello, se requería la dirección y el ID para conseguir la URI. Tras saber cuántos elementos se tiene, se hace lo mismo que anteriormente se hizo con los pisos pero con la componente *Marketplace*.html donde se crean componentes en la página pero con los pisos que están a la venta. Esto resulta en un componente por cada índice, que se consigue cargar ya que conociendo el *token*ID y cargando así su información e manera idéntica que en el caso de los pisos.

Dentro de *Marketplaceindex*, lo primero que hace es averiguar el *tokenID*. Cada piso de inventario está en componentes en piso que es lo que se itera tendrá un botón para poner a la venta. *Toggleforsale* interactúa con el servicio *Marketplace service* y le dice que se ponga a la venta por valor de 1BNB. Del mismo modo que en inventario, en *Marketplace* habrá un botón para comprar, que interactúa con *Marketplaceservice*,.

En el componente piso, para poder poner a la venta un piso NFT, es necesario haber aprobado antes el contrato por lo que se incluye un botón para ello que se muestre solo si no está aprobado el contrato (piso.component.html).

Este ultimo punto es fundamental, ya que el contrato *Marketplace* debe tener la autorización de realizar una transferencia del NFT del usuario en caso de que otro usuario compre el mismo por el precio pactado.

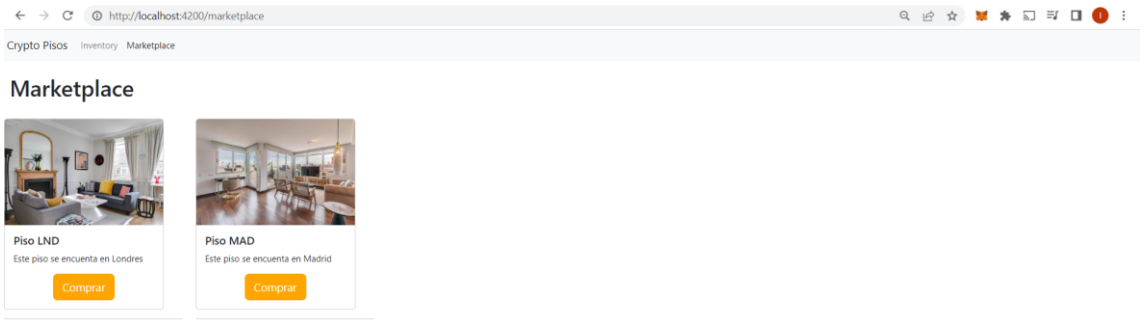


*Ilustración 38. Muestra del Marketplace final*

# Capítulo 5. TEORÍA DEL VALOR DE LOS CRIPTOACTIVOS

Antes de comenzar a analizar este apartado, se ha de aclarar el siguiente punto importante. Por el mero hecho de realizarse el trabajo sobre criptoactivos y en concreto NFTs, no se va a tomar una perspectiva sesgada sobre su valor. A continuación se tratará de hacer un análisis lo más imparcial posible, siendo las conclusiones obtenidas independientes de los intereses del autor.

Además, en ocasiones se hará uso del Bitcoin como proxy para el resto del mercado de criptoactivos. Esto no es del todo cierto pero permite un análisis que da un punto de vista comparable a un análisis de todos los criptoactivos haciendo un análisis exhaustivo del principal.

Habiendo aclarado estos puntos, se procede a analizar la teoría del valor de los criptoactivos.

## 5.1 BITCOIN DESDE UNA PERSPECTIVA FINANCIERA

En este primer apartado, se analizará que es Bitcoin desde un punto de vista financiero. Para poder identificar a Bitcoin como un activo, se deben definir las categorías a las que un activo puede pertenecer. Según el profesor de finanzas de Stern-NYU, Aswath Dasmodaran, los activos pueden pertenecer a una de cuatro categorías [16]:

1. **Activos que generan flujos de caja:** Son activos que en el tiempo generan flujos de caja positivos. Una acción de una empresa es un ejemplo, ya que otorga propiedad sobre una parte alícuota de la misma otorgando derecho a una parte de los flujos de caja. Los bonos del estado son otro ejemplo.

2. **Materias primas:** Las materias primas (*commodities*) son materiales empleados en los procesos productivos de otros productos. Ejemplos podrían ser el petróleo o el trigo.

3. **Divisas:** Una divisa es un activo financiero que se emplea como medio de intercambio, unidad de cuenta en una economía y depósito de valor.

4. **Elementos coleccionables:** En esta categoría entran las obras de arte u otros activos que como tal no generan flujos de caja por lo que su valor tiene una componente más subjetiva y se puede obtener por lo que se esté dispuesto a pagar.

Bitcoin en si no genera flujos de caja (como si lo hace la posesión de una acción ya que otorga propiedad sobre parte de una empresa que sí los genera) por lo que la primera categoría quedaría descartada. Bitcoin no se emplea en el proceso productivo de otros bienes por lo que tampoco es una materia prima. Quedarían dos alternativas: es una divisa o un elemento coleccionable.

Entre ambas opciones, la que mejor se ajusta es la de divisa (a pesar de que es debatible que podría ser un elemento coleccionable o podría terminar resultando eso en el futuro). Por otro lado, independientemente de a que categoría pertenezca, cabe analizar si es un buen elemento dentro de esta categoría, es decir, ¿es bitcoin una buena divisa?

Una buena divisa debe tener 3 características:

1. **Unidad de cuenta**: Bitcoin como unidad de cuenta es igual de valioso que cualquier otra unidad de cuenta que se desee emplear. En este aspecto no tiene diferencias significativas con otras divisas.
2. **Medio de intercambio**: Como medio de intercambio, es cierto que Bitcoin es cada vez más usado, sin embargo tiene comisiones muy elevadas relativas a los montos de transacciones que se hacen diariamente. Por una transacción hoy en día, el coste de la comisión es de en torno a 1 dólar [17]. Esto es independiente de si mandas 1 dólar

o 1 millón de dólares. Además, estas comisiones varían con el precio de bitcoin haciéndolo un medio de intercambio cuestionable.

3. **Depósito de valor**: Bitcoin como depósito de valor no es una gran divisa principalmente por la volatilidad de la misma. Se requiere cierta estabilidad y predictibilidad en el precio del activo para poderlo considerar un buen depósito de valor a largo plazo.

## 5.2 BITCOIN COMO PROTECCIÓN CONTRA LA INFLACIÓN

La conversación sobre el valor de los criptoactivos no estaría completa sin un análisis de su capacidad de proteger a un inversor contra la inflación.

Si un activo tiene la capacidad de proporcionar protección efectiva a un inversor contra la inflación, este tiene un valor intrínseco por el que los inversores estarán dispuestos a pagar por él, creando demanda por este bien. Prueba de ello es que los retornos nominales que proporcionan los bonos del tesoro americano son mayores que los retornos nominales proporcionados por los bonos del tesoro americanos protegidos contra la inflación (*TIPS: Treasurey Inflation Protected Securities)*, mostrando que los inversores están dispuestos a ceder cierta cantidad de retorno a cambio de tener protección contra la inflación.

Por ello, si se pudiese probar que Bitcoin proporciona una protección contra la inflación, cabría concluir que tendría un valor intrínseco.

Una de las ideas más comunes es que el Bitcoin, de la misma forma que oro, protege contra la inflación al tener una oferta limitada. Antes de proceder, hay que mencionar que el oro, contrario a la creencia popular, no es un buen protector contra la inflación. Esto queda demostrado en el *paper* titulado *The Golden Dilema* escrito por Claude B. Erb y Campbell R. Harvey [23] donde se contrasta que el oro es efectivo combatiendo la inflación pero en horizontes de inversión que no son prácticos (siglos) y que en el corto y medio plazo es volátil y no un buen protector contra la inflación. No se entrará en el detalle de los descubrimientos de este *paper* ya que escapa al alcance de este trabajo, sin embargo cabía

aclarar este punto debido a que la frecuente comparación del bitcoin con el oro para justificarlo como un buen protector contra la inflación.

Una vez desestimada la comparación del Bitcoin con el oro, se procede a analizar el Bitcoin de manera individual. La noción de que Bitcoin es un buen protector contra la inflación procede de la perspectiva monetarista que considera que lo único importante para protegerse contra la inflación es una oferta monetaria limitada, y al tener bitcoin una oferta monetaria de 21 millones [24], lo haría un activo ideal para protegerse contra la inflación. Sin embargo, esta perspectiva está muy desactualizada. Hay muchos más elementos que determinan la inflación más allá de la oferta monetaria (sin ir más lejos, la demanda monetaria). Independientemente de las discrepancias teóricas sobre este criptoactivos, ¿Qué dice la evidencia empírica al respecto?

Durante el periodo entre 2021-2022, en el que la inflación ha rondado el 10%, el Bitcoin ha bajado en términos reales un 20% [14]. Esto no es un protector efectivo contra la inflación, por lo menos en el corto plazo. Por si esto fuese poco, en el *paper* titulado *The Best strategies for inflationary times* [15], los autores parten de la hipótesis de que bitcoin debería tener una beta inflacionaria de 0 (es decir, cero correlación con la inflación) y una beta con el mercado de 0 (es decir, cero correlación con el mercado y teniendo retornos similares a los bonos estatales (*risk free rate*)). Sin embargo, las conclusiones de este estudio son que en realidad el bitcoin está fuertemente correlacionado con el mercado (por ejemplo, en marzo con la bajada fuerte de la bolsa debido al COVID-19 el Bitcoin también bajo fuertemente y a continuación, durante el periodo de recuperación de la bolsa, el bitcoin subió fuertemente. Esto muestra que bitcoin no es una reserva de valor (lo que lo haría un buen protector contra la inflación) sino una activo especulativo con una beta positivo frente al mercado.

Teniendo en cuenta que los mercados están correlacionados negativamente con la inflación y que bitcoin esta correlacionado con los mercados se concluye que Bitcoin no es un buen protector contra la inflación.

## 5.3 INVERSIÓN EN CRIPTOACTIVOS

Durante la realización del TFG hace 2 años, no se llegó a realizar un análisis desde el punto de vista financiero que permitiese considerarlo como un activo para la inversión. En este apartado, se agregarán las conclusiones vistas hasta ahora y, añadiendo algunas adicionales, se tratará de analizar qué lugar ocuparían los criptoactivos en una cartera de inversión (si es que tienen algún lugar). Hay que aclarar que este apartado es personal, en función de las valoraciones empíricas y cualitativas del autor, y no debe tomarse como ningún tipo de asesoramiento financiero. Se insta a que cada uno hagas sus propios análisis y llegue a sus propias conclusiones. Habiendo aclarado este punto, se procede a la cuestión; Inversión en Criptoactivos.

Se comienza analizando el valor de Bitcoin. Ya se ha establecido en apartados anteriores Bitcoin como una divisa. Por lo que cabe preguntarse, ¿Qué expectativa de retornos al largo plazo tienen las divisas? En un estudio realizado por Elroy Dimson, Paul Marsh, y Mike Staunton bajo el nombre de *Long-term asset returns* [18] los autores concluyen que en los últimos 150 años las divisas han variado mucho en valor relativo, pero que no hay un criterio claro para tener preferencia por unas frente a otras. Las divisas no tienen un retorno a largo plazo positivo. Sin embargo, esto podría deberse a que la oferta de divisas no está restringida mientras que la de bitcoin si lo está. Por ello una comparación con el oro sería más acertada. Sin embargo, el oro tampoco ha tenido retornos a largo plazo positivos.

Por consiguiente, que argumentos existirían para justificar el valor de Bitcoin. Algunos argumentos poco acertados que suelen emplearse para dotar de valor a Bitcoin son los siguientes:

1. **Véase cuanto ha subido el bitcoin en los últimos 12 años.** Efectivamente, en retrospectiva el Bitcoin es una inversión excelente. Entre 2011 y 2021, el retorno anualizado de bitcoin es de 230%, diez veces más que el NASDAQ 100. Sin embargo, retornos pasados no implican retornos futuros, por lo que este argumento

no puede emplearse para considerar Bitcoin como un activo sobre el que invertir en el presente

2. **Hay una oferta limitada por lo que el valor irá subiendo**. Esto es parcialmente cierto. Es verdad que una oferta limitada incrementa el valor de un activo frente a uno con una oferta ilimitada (*Ceteris paribus*). Sin embargo, este valor limitado es un dato conocido desde el inicio del protocolo, por lo que siendo un valor conocido está incluido en el precio del Bitcoin actualmente. Es decir, si ahora mismo se dijese que el número de bitcoin total va a ser 20 millones en vez de 21 el precio subiría instantáneamente mientras que si se dijese que pasara de 21 a 22 millones bajaría de la misma forma. Esto implica que esta información en el presente no sirve para probar retornos positivos futuros a largo plazo, al estar incluido este dato ya en e precio.
3. **Va a subir ya que cada vez lo usa más gente.** Este argumento es invalido por la misma razón. El precio de mercado de Bitcoin depende esencialmente de 4 elementos:
    - Oferta actual de Bitcoins.
    - Demanda actual de Bitcoins
    - Oferta futura de Bitcoins
    - Demanda futura de Bitcoins

    De estos elementos, solo hay uno desconocido en cada instante; La demanda futura de Bitcoins. Tanto la oferta actual, la demanda actual como la oferta futura se conocen, lo que no se sabe es el nivel de adopción que tendrá Bitcoin en un futuro. Si la adopción es masiva, cada Bitcoin tendrá mucho valor, mientras que si la adopción es nula cada Bitcoin no valdrá nada. Por ello, las variaciones de precio que se ven en Bitcoin son basadas en la adopción que el mercado espera en cada instante que tendrá bitcoin en un futuro. Las opiniones de todos los que participan en el mercado de Bitcoin se agregan y dan lugar a su precio. Sabiendo esto, si se tuviese un alto nivel de seguridad de que Bitcoin valdrá el doble dentro de un año, esto implicaría que el retorno de este activo sería del 100%, y teniendo en cuenta un retorno medio de acciones del 8%, el precio del Bitcoin subiría inmediatamente para captar esa oportunidad. Es por ello, que cualquier apuesta actual sobre si bitcoin va

a subir o bajar es en esencia eso, una apuesta contra el resto de las opiniones del mercado, basada en un análisis propio (o en pura conjetura) de que la adopción de bitcoin será mayor que la que ya está asumiendo actualmente el mercado.

Por lo que se ha analizado hasta ahora, parece que el valor de Bitcoin depende de manera muy significativa de la adopción futura que tendrá esta criptomoneda. Por ello, hay que analizar que incentivos existen para que se adopte Bitcoin cada vez más en un futuro:

1. Privacidad: Hasta cierto punto, Bitcoin proporciona privacidad en las transacciones (siempre y cuando no pueda asociarse la dirección publica de una cuenta con una persona física). Sin embargo, existen criptodivisas alternativas como Monero que cumplen con esta función de manera mucho más completa
2. Descentralización: La descentralización de una divisa evita el control de la misma por parte de ningún gobierno. Esto es una propuesta de valor atractiva para muchos individuos con preferencias por métodos de pago no controlados. Sin embargo, esta característica es compartida por la amplia mayoría de criptomonedas alternativas a Bitcoin
3. Participación en un ecosistema: De la misma forma que poseer dólares permite la participación en la economía estadounidense, Bitcoin permite la participación en el protocolo Bitcoin para realizar transacciones y con una cierta lógica básica no equiparable a la de los *Smart Contracts* en *Ethereum*. La participación en el protocolo Bitcoin parece de escasa utilidad comparado con otros ecosistemas que permitan la participación en entornos de aplicaciones descentralizadas para las que parece haber una demanda real. Pero ¿cuán notoria es esta demanda? Una aplicación descentralizada tiene como ventaja principal la descentralización que muchos individuos valoran pero esto es a costa de lentitud, precio elevado por uso, escalabilidad reducida y en general una experiencia de usuario peor. Debe ponerse sobre la balanza todo ello para determinar lo atractivo que es la propuesta de valor de las aplicaciones descentralizadas, pero independientemente de esto, Bitcoin no es un buen criptoactivos para participar en ecosistemas complejos de aplicaciones descentralizadas como si lo sería el Ether.

4. A modo de coleccionable: Si este es el caso, no cabría identificar al Bitcoin como una divisa sino que sería principalmente un coleccionable, es decir, por ser el primer criptoactivo en la historia tiene un valor por ello. Es dudoso que ese valor por sí solo justifique sin embargo su precio actual.

Para concluir, habiendo analizado todos los elementos quedar dar respuesta a dos preguntas. ¿Qué sucederá con el valor de Bitcoin? ¿Y qué sucederá con el valor del resto de criptoactivos?

A nivel personal se considera que el valor real de Bitcoin (es decir, sin tener en cuenta la inflación) en el futuro será muy muy bajo, si no nulo. Se tiene en cuenta que este tipo de posición es contraria a las grandes expectativas actuales. Sin embargo tras el análisis lógico realizado, el único valor que cabe otorgarle es el de coleccionable.

A pesar de esto, el resto de las criptodivisas sí que podrían tener un valor real debido a que permiten la participación en ecosistemas descentralizados. Lo que está por verse es cuan valioso son estos ecosistemas y si justifican el precio actual de estos criptoactivos. Haciendo referencia a lo visto durante este trabajo, la red BSC es mucho más centralizada que otras alternativas y sin embargo posee mucha aceptación por parte de la comunidad.

Si los usuarios valoran más lo que tiene que ofrecer el ecosistema *Blockchain* frente al valor coleccionable de Bitcoin, es cuestión de tiempo que el resto de monedas tengan una mayor capitalización que bitcoin.

Finalmente, es importante destacar que, basándose en el estudio reciente de Igor Makarov [25], se observa que muy pocos usuarios poseen las criptomonedas y que la principal razón por la que los usuarios invierten en Bitcoin es por la especulación. Además, otro paper [26] que recopila las características de los inversores de criptomonedas antes de invertir en cripto. Las conclusiones son que estos inversores invertían su dinero en acciones de tipo lotería, es decir, normalmente acciones de empresas con pequeño tamaño y alto ratio de precio a beneficios, que suelen ser acciones muy especulativas, con el potencial de grandes ganancias pero alta probabilidad de retornos muy bajos de media.

Por tanto, si los mismos inversores que invierten en criptoactivos son los que invierten en acciones de tipo lotería, no sería sorprendente que las criptomonedas se comportasen de la misma forma que estas acciones.

En resumen, no sería recomendable tener Bitcoin en un portfolio de inversión y se espera que su valor acabe siendo muy bajo.

# Capítulo 6. BIBLIOGRAFÍA

1. Sagredo Ruiz, I. (n.d.). Estudio sobre la cancelación de deuda circular mediante el uso de Smart Contracts en Ethereum. Retrieved from https://repositorio.comillas.edu/xmlui/bitstream/handle/11531/32763/TFG-%20Sagredo%20Ruiz%2C%20IAigo.pdf?sequence=1,
2. *Bitcoin: A peer-to-peer electronic cash system*. (n.d.). Retrieved August 25, 2022, from https://bitcoin.org/bitcoin.pdf
3. *World wide web and its journey from web 1.0 to web 4 - IJCSIT*. (n.d.). Retrieved August 25, 2022, from https://www.ijcsit.com/docs/Volume%205/vol5issue06/ijcsit20140506265.pdf
4. Takahashi, D. (2021, June 17). *Minds raises $10m for decentralized and encrypted social network and messaging app*. VentureBeat. Retrieved August 25, 2022, from https://venturebeat.com/2021/06/17/minds-raises-10m-for-decentralized-and-encrypted-social-network-and-messaging-app/
5. Ideas, W. I. R. E. D. (2022, May 12). *The Web3 decentralization debate is focused on the wrong question*. Wired. Retrieved August 25, 2022, from https://www.wired.com/story/web3-blockchain-decentralization-governance/
6. *The different types of cryptocurrency tokens explained*. MakerDAO Blog. (n.d.). Retrieved August 25, 2022, from https://blog.makerdao.com/the-different-types-of-cryptocurrency-tokens-explained
7. Wright, L. 'A. (2022, May 12). *You can redeem tether USDT 1:1 on tether.to but there's a catch*. CryptoSlate. Retrieved August 25, 2022, from https://cryptoslate.com/you-can-redeem-tether-usdt-11-on-tether-to-but-theres-a-catch/
8. *OpenSea the dominant NFT site with 60% market share*. (n.d.). Retrieved August 25, 2022, from https://nftnewstoday.com/2021/10/26/opensea-the-dominant-nft-site-with-60-market-share/
9. Wyvernprotocol. (2020, November 30). *Wyvern-V3/exchange.sol at master · WYVERNPROTOCOL/Wyvern-V3*. GitHub. Retrieved August 25, 2022, from https://github.com/wyvernprotocol/wyvern-v3/blob/master/contracts/exchange/Exchange.sol

10. Wyvernprotocol. (n.d.). *Wyvernprotocol/Wyvern-V3: Wyvern protocol v3.1, Ethereum Implementation*. GitHub. Retrieved August 25, 2022, from https://github.com/wyvernprotocol/wyvern-v3
11. Mkteam. (2022, May 29). *A quick guide to Polygon (Matic), the ETH layer-2 scaling powerhouse*. CoolWallet. Retrieved August 25, 2022, from https://www.coolwallet.io/a-quick-guide-to-polygon-matic-the-ethereum-layer-2-scaling-network
12. OpenSea. (n.d.). Https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/1033478955558883265544729911946991600960961960167014278915297281973703231406 09https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/1033478955558883265544729911946991600960961960167014278915297281973703231406 09https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/1033478955558883265544729911946991600960961960167014278915297281973703231406 09 - bored Ape Vans Metaverse. OpenSea. Retrieved August 25, 2022, from https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/1033478955558883265544729911946991600960961960167014278915297281984698347683 85
13. *Smart valor: Buy & sell bitcoin, Ethereum and other Digital assets*. SMART VALOR | Buy & Sell Bitcoin, Ethereum and other Digital Assets. (n.d.). Retrieved August 25, 2022, from https://smartvalor.com/en/news/bitcoin-performance
14. Robertson, H. (2021, November 22). *Bitcoin was looking good as an inflation hedge - then it plunged nearly 20%*. Business Insider. Retrieved August 25, 2022, from https://www.businessinsider.in/cryptocurrency/news/bitcoin-was-looking-good-as-an-inflation-hedge-then-it-plunged-nearly-20/articleshow/87817787.cms
15. Neville, H., Draaisma, T., Funnell, B., Harvey, C. R., & Van Hemert, O. (2021, March 29). *The best strategies for inflationary times*. SSRN. Retrieved August 25, 2022, from https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3813202
16. Damodaran, A. (1970, January 1). *The bitcoin boom: Asset, currency, commodity or collectible?* The Bitcoin Boom: Asset, Currency, Commodity or Collectible? Retrieved August 25, 2022, from https://aswathdamodaran.blogspot.com/2017/10/the-bitcoin-boom-asset-currency.html

17. Sarkar, A. (2022, April 18). *Bitcoin average transaction fees lowest in two years at $1.04*. Cointelegraph. Retrieved August 25, 2022, from https://cointelegraph.com/news/bitcoin-average-transaction-fees-lowest-in-two-years-at-1-04
18. Dimson, E., Marsh, P., & Staunton, M. (2016, December 1). *Long-term asset returns*. EconBiz. Retrieved August 25, 2022, from https://www.econbiz.de/Record/long-term-asset-returns-dimson-elroy/10011802694
19. *The world's most popular data science platform*. Anaconda. (n.d.). Retrieved August 25, 2022, from https://www.anaconda.com/
20. SpaceSeven. (2022, March 18). *Can nfts really have multiple owners? explained!* Medium. Retrieved August 25, 2022, from https://medium.com/space-seven/can-nfts-really-have-multiple-owners-explained-41a0a68a2bdf
21. *ERC 721*. OpenZeppelin Docs. (n.d.). Retrieved August 25, 2022, from https://docs.openzeppelin.com/contracts/2.x/api/token/erc721
22. *Vende Tu Vivienda A Reformar*. Habita Xr. (n.d.). Retrieved August 25, 2022, from https://www.habitaxr.com/
23. Erb, C. B., & Harvey, C. R. (2013, January 17). *The golden dilemma*. NBER. Retrieved August 25, 2022, from https://www.nber.org/papers/w18706
24. Hayes, A. (2022, July 13). *What happens to Bitcoin after all 21 million are mined?* Investopedia. Retrieved August 25, 2022, from https://www.investopedia.com/tech/what-happens-bitcoin-after-21-million-mined/
25. Makarov, I., & Schoar, A. (2021, October 25). *Blockchain analysis of the Bitcoin market*. NBER. Retrieved August 25, 2022, from https://www.nber.org/papers/w29396
26. *EconStor*. (n.d.). Retrieved August 25, 2022, from https://www.econstor.eu/bitstream/10419/218737/1/1698687354.pdf

# ANEXO I: ODS NACIONES UNIDAS

En este apartado se explica brevemente como se alinea este proyecto con los objetivos de desarrollo sostenible de naciones unidas. Se indican en primer lugar los 17 objetivos y a continuación se expresa el alineamiento.


*Ilustración 39. Objetivos ODS*

Sin lugar a duda, dicho trabajo se alinea principalmente con los siguientes 3 objetivos:

1. Objetivo número 8: Trabajo decente y crecimiento económico. Este objetivo se consigue con las propia tecnologia *Blockchain* y las oportunidades laborales que nacen de estos ecosistemas
2. Objetivo 9: Industria, innovación e infraestructura. La capacidad de realizar soluciones programáticas mediante el uso de *Smart Contract*s (en este caso, la creacion de un *Marketplace*), otorga oportunidades ilimitadas de innovación, además de una financiación adecuada de las mimsas
3. Objetivo 11: Ciudades y comunidades sostenibles: La creación, durante la prueba de concepto de este trabajo, de NFTs asociados a pisos, permite otorgar a los pisos que

se encuentren en estas ciudades la posibilidad de rendir cuentas a través de la red *Blockchain* confiable.