UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Desarrollo de un Smart Contract en Blockchain para la Trazabilidad de Productos: Aplicación a Metric Salad

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: Rodriguez de Limia de Miguel, Jorge

Tutor/a: Ramos Peinado, Germán

Cotutor/a externo: Zaragoza Rubio, Salvador

CURSO ACADÉMICO: 2024/2025

Agradecimientos

En primer lugar, deseo agradecer a mi familia, por la cual ha sido posible estudiar esta carrera aquí en Valencia.

Quiero expresar mi profundo agradecimiento a todas las personas que han contribuido a la realización de este trabajo durante mi período en la empresa Metric Salad, especialmente a mi tutor en la empresa, Salvador Zaragoza Rubio, por su orientación, apoyo y conocimientos compartidos durante mi estancia en Metric Salad y a mi jefa de proyectos Yolanda Trujillo por su ayuda y orientación en el proyecto. Sus consejos fueron fundamental para el desarrollo exitoso del mismo.

Asimismo, quiero agradecer a mi tutor en la Universitat Politècnica de València (UPV), Germán Ramos Peinado, por su dedicación y asesoramiento a lo largo de este proceso. Sus comentarios y sugerencias fueron de gran valor para mejorar este trabajo.

Por último, deseo expresar mi reconocimiento a todos los profesores de la UPV que me han acompañado y guiado a lo largo de mi carrera universitaria. Su enseñanza y orientación han sido fundamentales para mi crecimiento académico y profesional.

Gracias a todos por su apoyo y contribución a este proyecto.

Resumen

Uno de los mayores desafíos que ha enfrentado el mundo en los últimos años es la falta de transparencia y trazabilidad en numerosas industrias, incluida la agrícola. Esta opacidad no solo dificulta la gestión eficiente de la cadena de suministro, sino que también plantea serias preocupaciones sobre la seguridad alimentaria, la sostenibilidad y la equidad en el comercio. Sin embargo, gracias a la tecnología Blockchain, estamos ante la oportunidad de abordar este problema de manera innovadora y efectiva.

El presente trabajo tiene como objetivo explorar el potencial de la Blockchain en el ámbito agrícola, enfocándose en sus capacidades para proporcionar transparencia, trazabilidad y seguridad en toda la cadena de suministro. Al establecer un registro inmutable y descentralizado de transacciones, la Blockchain ofrece la oportunidad de rastrear el origen y el destino de los productos agrícolas de manera eficiente y confiable, desde su producción hasta su consumo final.

Además, se analizarán los beneficios tangibles que la Blockchain puede aportar a los agricultores, las empresas agrícolas, los consumidores y la sociedad en general, incluyendo la reducción del fraude, la optimización de los procesos logísticos y la mejora de la calidad y autenticidad de los productos agrícolas.

Abstract

One of the biggest challenges the world has faced in recent years is the lack of transparency and traceability in numerous industries, including agriculture. This opacity not only hinders efficient supply chain management but also raises serious concerns about food safety, sustainability, and fairness in trade. However, thanks to Blockchain technology, we have the opportunity to address this problem in an innovative and effective way.

The aim of this work is to explore the potential of Blockchain in the agricultural sector, focusing on its capabilities to provide transparency, traceability, and security throughout the supply chain. By establishing an immutable and decentralized record of transactions, Blockchain offers the opportunity to track the origin and destination of agricultural products efficiently and reliably, from production to final consumption.

Furthermore, the tangible benefits that Blockchain can bring to farmers, agricultural companies, consumers, and society as a whole will be analyzed, including fraud reduction, optimization of logistical processes, and improvement of the quality and authenticity of agricultural products.

Resum

Un dels majors desafiaments que ha enfrontat el món en els últims anys és la falta de transparència i traçabilitat en nombroses indústries, inclosa l'agrícola. Aquesta opacitat no només dificulta la gestió eficient de la cadena de subministrament, sinó que també planteja serioses preocupacions sobre la seguretat alimentària, la sostenibilitat i l'equitat en el comerç. No obstant això, gràcies a la tecnologia Blockchain, estem davant l'oportunitat d'abordar aquest problema de manera innovadora i efectiva.

El present treball té com a objectiu explorar el potencial dla Blockchain en l'àmbit agrícola, enfocant-se en les seues capacitats per proporcionar transparència, traçabilitat i seguretat en tota la cadena de subministrament. En establir un registre immutable i descentralitzat de transaccions, la Blockchain ofereix l'oportunitat de rastrejar l'origen i el destí dels productes agrícoles de manera eficient i fiable, des de la seua producció fins al seu consum final.

A més, s'analitzaran els beneficis tangibles que la Blockchain pot aportar als agricultors, les empreses agrícoles, els consumidors i la societat en general, incloent la reducció del frau, l'optimització dels processos logístics i la millora de la qualitat i autenticitat dels productes agrícoles.

RESUMEN EJECUTIVO

La memoria del TFG del Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación debe desarrollar en el texto los siguientes conceptos, debidamente justificados y discutidos, centrados en el ámbito de la ingeniería de telecomunicación

CONCEPT (ABET)	**CONCEPTO (traducción)**	**¿Cumple? (S/N)**	**¿Dónde? (páginas)**
1. IDENTIFY:	1. IDENTIFICAR:		
1.1. Problem statement and opportunity	1.1. Planteamiento del problema y oportunidad	S	5
1.2. Constraints (standards, codes, needs, requirements & specifications)	1.2. Toma en consideración de los condicionantes (normas técnicas y regulación, necesidades, requisitos y especificaciones)	S	5 y 16
1.3. Setting of goals	1.3. Establecimiento de objetivos	S	6
2. FORMULATE:	2. FORMULAR:		
2.1. Creative solution generation (analysis)	2.1. Generación de soluciones creativas (análisis)	S	16
2.2. Evaluation of multiple solutions and decision-making (synthesis)	2.2. Evaluación de múltiples soluciones y toma de decisiones (síntesis)	S	21-34
3. SOLVE:	3. RESOLVER:		
3.1. Fulfilment of goals	3.1. Evaluación del cumplimiento de objetivos	S	34
3.2. Overall impact and significance (contributions and practical recommendations)	3.2. Evaluación del impacto global y alcance (contribuciones y recomendaciones prácticas)	S	36

INDICE

1.- INTRODUCCIÓN 5
1.1.- OBJETIVOS 6
2.- HISTORIA Y EVOLUCION DE LA BLOCKCHAIN 7
2.1- BLOCKCHAIN 8
2.2- TIPOS DE RED BLOCKCHAIN 9
2.3- APLICACIONES Y CASOS DE USO 10
3.-SMART CONTRACT 12
3.1- ¿QUE ES UN SMART CONTRACT? 12
3.2- VENTAJAS Y CARACTERISTICAS 12
3.3.- APLICACIÓN DE LOS SMART CONTRACTS 13
4.- METRICSALAD 14
4.1.- EMPRESAS 14
5.- IMPLEMENTACIÓN Y DESARROLLO 16
5.1.- ¿PARA QUE SE USARIA AWS? 18
5.2.- EJEMPLOS DE TRAZABILIDAD DE PRODUCTOS CON BLOCKCHAIN 19
6.- DISEÑO DE LA SOLUCIÓN Y RESULTADOS 21
7.- ANALISIS DEL RESULTADO 36
8.- RELACIÓN DEL TRABAJO CON LOS ODS 38
9.-CONCLUSIONES 40
10.-TRABAJO FUTURO 41
11.-REFERENCIAS 42
12.-ANEXOS 44

LISTA DE FIGURAS

Figura 1: Evolución de la Blockchain [5] 7

Figura 2 : Tipos de redes Blockchain [7] 10

Figura 3 : Ventajas Smart contracts [15] 13

Figura 4 : Logo de MetricSalad 14

Figura 5 : Producto Carrefour Bio [24] 19

Figura 6: Esquema para la realización de una red blockchain 21

Figura 7: Componentes de la red 29

Figura 8: Funciones del Smart contract 31

Figura 9: Aprobación del Smart contract 33

Figura 10: Resultado de la ejecución de la app 35

LISTA DE TABLAS

Tabla 1: Comparativa aws vs gcp vs azure 17

Tabla 2 : Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible 38

LISTA DE ABREVIATURAS

Amazon Web Service --- AWS

Google Cloud Platform --- GCP

Certificado de autorización --- CA

1.- INTRODUCCIÓN

Hoy en día, la agricultura se enfrenta a diversas dificultades, como puede ser entre otras el desconocimiento de la procedencia de los alimentos, por lo que provoca la falta de confianza de los clientes y la reducción del consumo. Aquí es donde entra en juego la tecnología Blockchain.

Esta tecnología nos permite crear un registro claro y verificable de cada paso que da un producto, desde que se planta en el campo hasta que llega a nuestra mesa. Así, los consumidores podemos conocer toda la historia detrás del producto.

El objetivo principal, es estudiar en detalle cómo puede ayudar Blockchain en el sector agrícola, viendo ejemplos prácticos y descubriendo sus ventajas y desafíos. Con esta investigación, esperamos entender mejor cómo Blockchain puede cambiar cualquier ámbito, en nuestro caso el agrícola y como poder sacarles el mejor rendimiento posible a los dos campos juntos.

Este proyecto es una aplicación, desarrollada para la empresa MetricSalad, en la cual estuve realizando mis prácticas de empresa, para la realización de un canal que contenga un Smart contract, programas diseñados para ejecutarse automáticamente, cuando se cumplen ciertas condiciones preestablecidas que guardarán toda la trazabilidad de los productos de una serie de empresas permitiendo que el consumidor realice un seguimiento del producto en todas las fases que este ha tenido desde su origen hasta llegar al cliente final .

La Blockchain es una tecnología de registro distribuido que permite almacenar información de manera segura, transparente y descentralizada. Su seguridad se basa en técnicas criptográficas que hacen que los datos no puedan ser manipulados y al eliminar los intermediarios y no depender de un ordenador central disminuye el riesgo de fallos o manipulaciones aumentando asi la confianza.

En resumen, este proyecto se encuentra en un punto medio entre la innovación tecnológica y las necesidades urgentes de la agricultura, por lo que es perfecto tanto para conocer esta nueva tecnología Blockchain, como para encontrar una solución que beneficie al sector agrícola.

1.1.- OBJETIVOS

Los objetivos que se buscan completar para la finalización de este trabajo son los siguientes:

- Crear un sistema de trazabilidad enfocado a productos agrícolas. El sistema deberá permitir tanto guardar como recuperar los datos del ciclo de vida que ha seguido un producto, a través de contratos inteligentes que se ejecutarán en una red Blockchain.
- Desarrollar la configuración necesaria para la creación e implementación de los contratos inteligentes en Blockchain.
- Extraer conclusiones sobre la viabilidad y aplicación de la tecnología Blockchain para este propósito.

2.- HISTORIA Y EVOLUCION DE LA BLOCKCHAIN

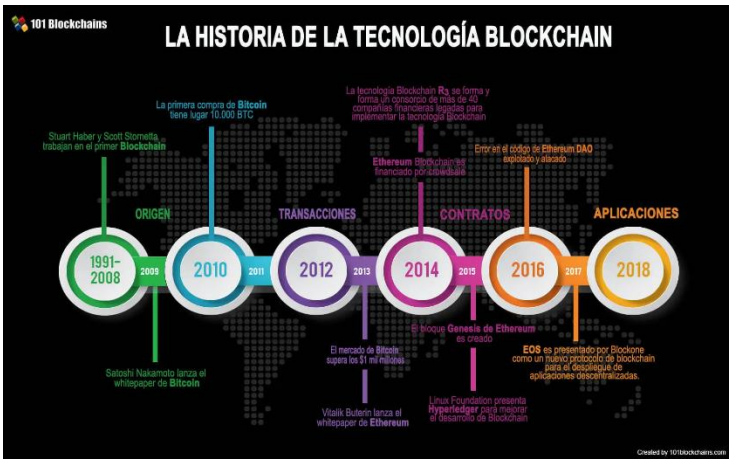

FIGURA 1: EVOLUCIÓN DE LA BLOCKCHAIN [5]

Como podemos ver en la figura1, en 1991, Stuart Haber y W. Scott Stornetta, [1], introdujeron una solución computacional para que los documentos digitales no pudieran ser modificados o manipulados. Este sistema usó una cadena de bloques, es decir una cadena compuesta por estructuras de datos para registrar los datos de las transacciones [2], donde estos bloques están enlazados entre sí y protegidos criptográficamente. Un año después actualizaron el sistema incorporando árboles de Merkle, una estructura de datos que sirve como resumen de todas las transacciones permitiendo verificar el contenido de una gran cantidad de datos [3], lo que la hizo más eficiente permitiendo almacenar varios documentos en un solo bloque.

A finales de 2008, Satoshi Nakamoto [4]**,** conceptualizó el primer Blockchain basado en el algoritmo de Prueba de Trabajo de Hashcash, [1], protocolo utilizado para combatir el correo no deseado (spam) y garantizar la seguridad en las redes digitales, pero no fue hasta el año siguiente cuando Nakamoto minó el primer bloque de Blockchain, es decir verificó y registró la transacción en la cadena de bloques.

La evolución de la tecnología Blockchain ha pasado por tres fases distintas [4].

1. En la primera fase, surgió Bitcoin como la primera aplicación significativa de Blockchain. Esto introdujo la idea de una red descentralizada de pagos peer-to peer, es decir la transferencia de dinero directamente entre dos personas sin necesidad de intermediarios de forma inmediata, estableciendo asi los fundamentos de lo que hoy en día conocemos como Blockchain.
2. La segunda fase se desarrolló entre 2013-2015 y se centró en la creación de Ethereum, una plataforma de Blockchain diseñada para desarrollar aplicaciones descentralizadas y ejecutar contratos inteligentes (Smart contract), por parte de Vitalik Buterin [4].
3. Finalmente, la tercera fase comenzó alrededor de 2018 y continua en la actualidad. Durante esta fase, han surgido numerosas plataformas y proyectos que buscan abordar las limitaciones y los desafíos de las fases anteriores, como **NEO** e **IOTA**

[4], que quiere resolver problemas de escalabilidad y seguridad. Además, empezaron a surgir las Blockchain privadas e híbridas, como las desarrolladas por Microsoft, que buscaban mejorar la eficiencia operativa.

Anteriormente en 2015, la Fundación Linux, presentó un proyecto de Blockchain de código abierto, **Hyperledger** [4], que quería promover el uso de Blockchain en aplicaciones empresariales en vez de orientadas a las criptomonedas. El objetivo principal era fomentar el uso de esta tecnología para mejorar el rendimiento y confiabilidad de los sistemas actuales para respaldar las transacciones [4].

2.1- BLOCKCHAIN

La Blockchain es un libro de contabilidad compartido e inmodificable que facilita el registro de transacciones y el seguimiento de activos en una red empresarial. Permite rastrear cualquier cosa de valor en una red Blockchain, lo que reduce riesgos y costos. Proporciona información inmediata, compartida y transparente, almacenada en un ledger (libro mayor) inmodificable al que solo pueden acceder los miembros de la red con permisos. Además, agiliza procesos, abarata transacciones y elimina intermediarios [6].

La Blockchain es una tecnología basada en una cadena de bloques descentralizada y pública. Cada transacción generada se almacena en un bloque que se añade a la cadena de bloques existente.

La tecnología Blockchain funciona mediante bloques, que son estructuras de datos que contienen un conjunto de transacciones [2] , donde cada bloque esta enlazado al anterior formando una cadena de bloques (de ahí el nombre de Blockchain). Estos bloques dependen de la validación de las transacciones por parte de los nodos y este proceso es realizado por los mineros, unos participantes de la red que verifican que las transacciones dentro de un bloque sean válidas. Una vez que un minero valida un bloque completo de transacciones, lo añade a la cadena de bloques. Los nodos almacenan y distribuyen una copia completa de la cadena de bloques, asegurando que esa información sea accesible para todos los participantes de la red [7].

En el sector agrícola, la Blockchain está revolucionando la gestión de la cadena de suministro y la seguridad alimentaria al proporcionar una trazabilidad completa desde el campo hasta el consumidor final. Al utilizar la Blockchain, los agricultores pueden registrar cada etapa del proceso de producción, incluida la siembra, el cultivo, la cosecha

y el embalaje, junto con información detallada sobre el uso de pesticidas, fertilizantes y otras prácticas agrícolas. [8]

Hoy en día, la tecnología Blockchain está proporcionando transparencia en la cadena de suministro de alimentos, asegurando los datos de atención médica, innovando en el mundo de los juegos y cambiando la forma en que manejamos datos y propiedad a gran escala.

En resumen, la Blockchain es una tecnología revolucionaria que ofrece transparencia, seguridad y eficiencia en la gestión de activos y transacciones.

2.2- TIPOS DE RED BLOCKCHAIN

Las redes Blockchain pueden ser de 4 tipos: públicas, privadas, de consorcio o hibridas.

Una Blockchain **pública** es una red descentralizada y abierta a la que cualquier persona puede acceder libremente desde Internet. En estas redes, los datos son públicos y accesibles para cualquier usuario, lo que permite su visualización por parte de todos. Además, su funcionamiento se basa en un sistema de incentivos económicos, como la minería de criptomonedas, en la cual los mineros reciben una compensación en forma de criptomonedas. Un ejemplo de red pública podría ser Ethereum [7].

Por otro lado, una Blockchain **privada** cuenta con una entidad central que controla todas las acciones dentro de la red. A diferencia de las Blockchains públicas, el acceso a estas redes está restringido y el libro de transacciones es privado. La entidad central de estas redes administra los permisos de los usuarios y controla las funciones dentro de la Blockchain. Uno de los ejemplos más comunes de este tipo de Blockchain seria Hyperledger [9].

También tenemos otros tipos de red, como las de **consorcio**, que son redes semiprivadas donde un grupo de nodos o participantes controlan la validación de las transacciones y la administración de la red. Estas redes se usan en entornos empresariales donde se requiere confianza y colaboración entre los participantes, pero se desea mantener cierto grado de control sobre la red y los datos, donde la administración y los permisos son compartidos entre los nodos participantes. Uno de los ejemplos más comunes seria Quorum de JPMorgan y Hyperledger Fabric.

Finalmente, la Blockchain híbrida combina elementos de las Blockchains públicas y privadas. En este modelo, la participación en la red es privada, pero el libro contable con el registro de todas las transacciones es de acceso público. Esta combinación busca

aprovechar lo mejor de ambos mundos, ofreciendo control y privacidad junto con transparencia y accesibilidad de datos. Este enfoque es especialmente útil en sectores donde se requiere un equilibrio entre la transparencia y el control, como en el ámbito de la salud.

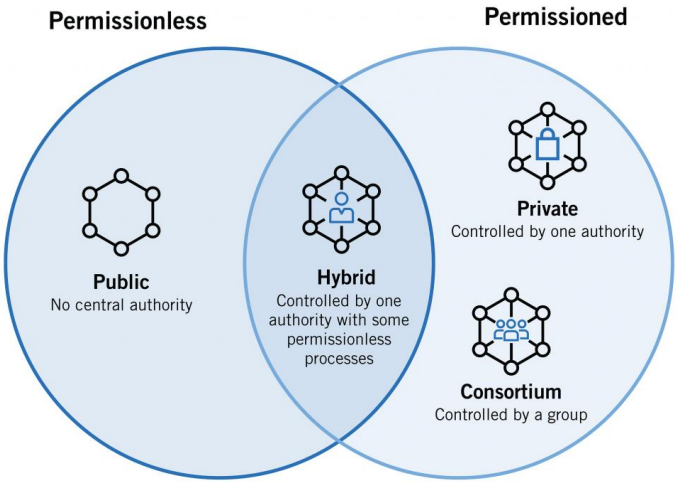

FIGURA 2 : TIPOS DE REDES BLOCKCHAIN [7]

2.3- APLICACIONES Y CASOS DE USO

Las redes Blockchain han ido cogiendo mucha importancia en la tecnología en muchos ámbitos diferentes, por lo que la consolidan como una de las tecnologías más prometedoras.

Una de sus aplicaciones más importantes es en la sanidad, donde gracias a Blockchain se pueden almacenar historiales médicos, donde los usuarios pueden acceder a él en cualquier momento e incluso compartirlo el acceso con una transacción para que los médicos con los permisos necesarios puedan actualizar el historial [10].

Otra de las aplicaciones más comunes de la Blockchain es la seguridad alimenticia, considerado uno de los problemas mundiales más importantes. La falta de transparencia en la trazabilidad de los productos es una de las mayores preocupaciones a nivel mundial. Con la incorporación de Blockchain, cada alimento colocado en la cadena de suministro se puede etiquetar con información vital, como la temperatura de almacenamiento, los productos utilizados, el tipo de riego o el tratamiento utilizado para su crecimiento. También se establece que nadie puede modificar los datos a lo largo de la cadena del

suministro, por lo que verificamos la importancia que puede tener Blockchain en este sector [10].

Por último, vale la pena destacar una de las aplicaciones más relevantes de la Blockchain, que es la criptografía. La criptografía es un método de protección de información de modo que solo los usuarios a los que está destinada la información podrán leerla y procesarla. Las cadenas de bloques utilizan dos tipos de algoritmos criptográficos: algoritmos de clave asimétrica y las funciones de Hash, que sirven para mantener la integridad de los datos almacenados dentro de un bloque y la vinculación de los bloques entre sí. Estas cadenas de bloque utilizan el algoritmo hash SHA-256 (algoritmo que se utiliza para convertir datos de cualquier longitud en una cadena de caracteres alfanuméricos de longitud fija). Además de su aplicación en la seguridad de datos y comunicaciones, la criptografía también juega un papel crucial en el funcionamiento de las criptomonedas, que utilizan pares de claves públicas (usada para cifrar datos y garantizar la seguridad de la comunicación) y privadas (valor secreto que se utiliza para acceder a los datos y autorizar las transacciones) para mantener las direcciones de los usuarios en la cadena de bloques [11].

3.-SMART CONTRACT

3.1- ¿QUE ES UN SMART CONTRACT?

Los Smart contracts, también llamados chaincode o contratos inteligentes, son programadas diseñados para ejecutarse automáticamente, cuando se cumplen ciertas condiciones preestablecidas. Funciona sin necesidad de una verificación de una entidad supervisora ya que la tecnología Blockchain garantiza la seguridad de las transacciones. Es un acuerdo que se establece entre 2 o más partes, escrito en lenguajes de programación, y visible para todos [12]. La cadena de bloques se actualiza de forma automática cada vez que se completa una transacción. Este proceso ocurre cuando los mineros validan un bloque y lo añaden a la cadena. Una vez este bloque se añade a la cadena es inmutable, lo que significa que no se puede cambiar o alterar una transacción ya que, si alguien intentara modificar un bloque el hash del bloque cambiaria, lo que rompería la conexión entre los bloques de la cadena.

3.2- VENTAJAS Y CARACTERISTICAS

En cuanto a las características de los Smart contracts destacamos una de las más importantes, la seguridad gracias a la criptografía y la transparencia de las transacciones registradas en la cadena de bloques. También eliminan la necesidad de realizar procesos manuales y reducir costos sin necesidad de intermediarios para la ejecución, dando una mayor libertad a las partes para realizar acuerdos [13].

Los contratos inteligentes proporcionan también varias ventajas respecto a los tradicionales. El proceso se agiliza y se asegura una ejecución de forma segura, garantizando la integridad de los acuerdos y protegiendo la información sensible. Al eliminar el intermediario, aparte de reducir los costos, las relaciones comerciales son más directas y transparentes garantizando que las transacciones sean registradas de forma inmutable y transparente [14].

Podemos ver en la figura 3 como los Smart contracts permiten un proceso más eficiente, seguro y transparente, al eliminar la dependencia de terceros y automatizar la ejecución de acuerdos según condiciones preestablecidas.

FIGURA 3 : VENTAJAS SMART CONTRACTS [15]

3.3.- APLICACIÓN DE LOS SMART CONTRACTS

Los Smart contracts permiten automatizar procesos de manera segura y eficiente. Por ejemplo, en el ámbito financiero, se utilizan para agilizar el proceso de préstamos, mientras que en el campo de la salud simplifican la administración de historiales médicos y posibilitan el intercambio seguro de información entre los proveedores de atención médica [16].

La gestión de identidad es una de las aplicaciones más importantes de los Smart contracts, creando sistemas seguros para facilitar el intercambio de información entre diferentes entidades. Esto permite agilizar los procesos de verificación y reducir el riesgo de errores o fraudes.

Por último, los Smart contracts facilitan mucho el proceso de las elecciones para poder hacer unas votaciones de manera segura, eliminando la posibilidad de fraude y manipulación de los resultados.

4.- METRICSALAD

MetricSalad [17] es una empresa española, con más de 18 años de experiencia que opera principalmente en España, pero también en multitud de países: Arabia Saudí, Canadá, Suiza, Holanda y Australia. En 2010 abrieron otra oficina en Nueva York, desde donde operan para todo el continente, especialmente USA, Canadá, Chile y Perú, bajo la marca thelinebBTWN.

Integran en los procesos de negocio de las organizaciones las técnicas más avanzadas en Inteligencia Artificial, que van desde el Aprendizaje Automático (Machine Learning) hasta los Servicios Cognitivos (IBM Watson, AWS AI, Google AI, etc.), aportando ventajas competitivas y disruptivas de las empresas. Así mismo afrontan retos como mitigar sesgos que deriven en discriminación (Fairness Learning) e interpretar los resultados aportando transparencia en las decisiones. Entre los servicios que desarrollan se encuentran: Machine Learning, Reinforce Learning, Process Mining, Motores de recomendación, Análisis de grafos y Servicios cognitivos.

Hace unos meses empezaron con un proyecto de Blockchain sobre la trazabilidad de los productos de una gran cantidad de empresas, en el cual yo he participado. Este proyecto viene recogido en un PERTE que nos explican cómo está organizado, la dedicación de cada participante o incluso los entregables que hay que realizar, así que mi primera tarea fue leérmelo y comenzar con una investigación sobre todas las empresas que participan en él, para así poder conocerlas mejor, incluso que se puedan intuir posibles datos que recogen en la trazabilidad de sus productos.

FIGURA 4 : LOGO DE METRICSALAD

4.1.- EMPRESAS

Una vez iniciado el proyecto, el siguiente paso fue conocer todas las empresas del sector agrícola y alimentario con las que íbamos a trabajar, para poder estimar cuales son los datos que recogen para la trazabilidad de sus productos. Esta investigación se realizó a 54 empresas, cuyos nombres no se mencionan debido a que no tengo el permiso necesario para citarlas.

La investigación de las empresas cubrió los siguientes aspectos:

- Localización
- Resumen de la actividad: Descripción de cada empresa para entender el sector en el que operan
- Estimación de los datos de trazabilidad: Una estimación sobre los datos que podrían recoger estas empresas para la trazabilidad de sus productos:

    1. Identificación del producto (ID): Cada producto o lote se identifica de manera única en la red. Este ID es fundamental para rastrear el producto a lo largo de su trazabilidad.
    2. Fechas importantes: Como podrían ser fechas de cosecha y recolección, para tener constancia sobre la frescura del producto.
    3. Lugar de Producción: El lugar donde se cultivo el producto para poder conocer el origen geográfico del producto.
    4. Características del producto: Como variedad de la fruta, el peso, los fertilizantes o plaguicidas utilizados en el producto.
    5. Información sobre los participantes de la trazabilidad: El cliente destino o el agricultor para asegurar que el producto llegue al consumidor final y poder identificar a los responsables en caso de problemas con el producto.
    6. Control y monitoreo de cultivo: Sesiones de riego y abonado para proporcionar una información adicional al producto.

5.- IMPLEMENTACIÓN Y DESARROLLO

Una vez hecho el estudio de las empresas y sabiendo cuales son los posibles datos que saldrán en la trazabilidad del producto, el siguiente paso era elegir la plataforma donde íbamos a desarrollar nuestra red de Blockchain y aunque hayamos contemplado otras opciones como son **Stellar, Ripple** o **Cardano**, nos hemos centrado en las dos plataformas principales como son **Ethereum** y **Hyperledger**. [18]

La primera de estas opciones**, Ethereum** es conocida por su flexibilidad y capacidad para ejecutar contratos inteligentes, lo que la convierte en una excelente opción para facilitar el intercambio directo entre empresas y clientes, con transacciones transparentes y visibles. Sin embargo, tras una investigación más detallada, hemos optado por descartar Ethereum debido a un inconveniente económico significativo: Cada transacción en la red requiere que se pague una tarifa de gas, una tarifa que tienes que pagar en Ether (la moneda de Ethereum) para que se realicen las transacciones o se ejecuten los contratos inteligentes por lo que podría traducirse en costos adicionales que afectarían negativamente a nuestra rentabilidad. [19]

Por otro lado, hemos explorado la plataforma **Hyperledger**, por su gran estabilidad y su capacidad. **Hyperledger** ofrece dos modelos principales para el desarrollo de redes Blockchain como son **Fabric** y **Sawtooth** [4]. Aunque inicialmente consideramos **Sawtooth** como la mejor opción, debido a su desvinculación de **Hyperledger** y estar ahora en otra plataforma como es Splinter [20], así como la dificultad para encontrar un proveedor de servicios compatible, hemos concluido que **Hyperledger Fabric** es la mejor opción para nuestro proyecto.

Una vez seleccionada la plataforma Blockchain, surge la necesidad de un proveedor de servicios que mejore la escalabilidad de nuestra red. Dado que almacenar grandes volúmenes de información en la cadena puede resultar poco práctico debido a la necesidad de replicar los datos en todos los nodos, es crucial encontrar la solución más adecuada. Hemos identificado tres posibles proveedores**: Microsoft Azure, Google Cloud Platform (GCP)** y **Amazon Web Services (AWS).**

La primera de ellas, **Microsoft Azure** se descartó por diversas razones, destacando su modelo de facturación por minuto en vez de por minutos, lo que podría resultar más costoso comparado con **AWS** y **GCP**. Otro de los motivos principales por el cual fue descartada es que puede tener una menor estabilidad en ciertos aspectos de la plataforma y una limitación en cuanto al uso de tecnologías que no sean de Microsoft. [21]

Por lo que la decisión se reduce a dos, **AWS** o **GCP**. Ambos son muy parecidos, pero podemos observar algunas diferencias clave, **AWS** es tanto el pionero como el líder en el mercado de servicios en la nube, con una amplia gama de opciones de configuración que

la hacen muy flexible, sólida y confiable y aunque **GCP** está creciendo rápidamente, aún no puede igualar a **AWS** en términos de funcionalidad y alcance geográfico, lo que podría limitar su uso para ciertos usuarios.

En términos económicos, ambas opciones podrían ser comparables, aunque **GCP** tiende a ser ligeramente más costoso.

	AWS	GCP	AZURE
Mercado	50%	30%	20%
Fiabilidad y Estabilidad	35%	35%	30%
Alcance geográfico	40%	30%	30%
Tecnologías y compatibilidad	40%	35%	25%
Costos	40%	35%	25%

TABLA 1: COMPARATIVA AWS VS GCP VS AZURE

En resumen, tras analizar las opciones disponibles he elegido **AWS** como proveedor de servicios debido a su liderazgo en el mercado, su gran alcance geográfico y su modelo de facturación por segundos.

Por último, tuve que decidir en qué lenguaje de programación iba a desarrollar el Smart contract de mi red, teniendo como opciones**: JavaScript (JS), Go** y **TypeScript**. Elegí **TypeScript** debido a dos factores claves:

El primero de ellos fue la compatibilidad con **JavaScrip**t, ya que, al ser un superconjunto de JavaScript, cualquier código escrito en este lenguaje es válido en **Typescript**.

El segundo factor clave fue la posibilidad de añadir variables estáticas, es decir indica al ordenador que tipo de datos debe esperar en cada variable. Esto es muy importante para el desarrollo de los Smart contract, ya que podemos detectar errores antes de que el código se ejecute.

5.1.- ¿PARA QUE SE USARIA AWS?

La red Blockchain se va a desplegar en un entorno local, por lo que la red depende directamente del ordenador donde este configurada, ya que si este equipo se apaga o sufre un fallo técnico la red Blockchain dejaría de estar operativa. Para solucionar este problema hay que trasladar la red a un proveedor de servicios en la nube como AWS.

Trasladar la red Blockchain a AWS tiene varias ventajas claves para este proyecto:

- Alta disponibilidad: AWS garantiza que la red esté operativa siempre independientemente de posibles fallos en el servidor.
- Escalabilidad: Permite aumentar la capacidad de la red si fuera necesario, permitiendo gestionar un mayor número de transacciones o usuarios que un ordenador local.
- IoT: AWS ofrece varios servicios como IoT Core [23] , permitiendo conectar dispositivos de IoT, de forma segura para poder recopilar datos en tiempo real mejorando la trazabilidad del producto.
- Optimización: Al trasladar la red eliminaríamos la gestión manual del hardware y software ya que de eso se encargaría AWS, permitiendo asi centrarme en optimizar el sistema y añadir mejoras.

5.2.- EJEMPLOS DE TRAZABILIDAD DE PRODUCTOS CON BLOCKCHAIN

La tecnología Blockchain se ha implementado en muchos sectores, destacando especialmente en la trazabilidad de los productos. A continuación, voy a mostrar algunos ejemplos reales donde las empresas han realizado la trazabilidad del producto en Blockchain:

- **Carrefour Bio:** A partir de 2022 Carrefour Bio implementó la tecnología Blockchain para sus productos ecológicos.

FIGURA 5 : PRODUCTO CARREFOUR BIO [24]

En la figura 4, podemos ver un producto de Carrefour Bio con un código QR en su etiqueta. Al escanear este código QR el consumidor puede conocer todo sobre el ciclo de vida del producto [24]:

- Origen y el camino que ha seguido: Nombre del productor, ubicación del campo, ubicación del empaque, medio de transporte.
- Calidad: Fecha de cosecha, resultado de análisis, variedad.
- Certificación orgánica: Fecha de conversión, certificado oficial o iniciativas adicionales implementadas por el productor.

Este sistema de trazabilidad no solo ofrece una mayor transparencia, sino que también garantiza la seguridad de los datos al ser inmutables, por lo que el consumidor puede tener la certeza de que la información es verídica.

- **IBM:** La empresa IBM desarrollo una plataforma basada en Blockchain, IBM Food Trust, para conseguir una trazabilidad transparente y sostenible. Su objetivo era aumentar la seguridad a cada miembro de la cadena de suministro y conseguir un proceso más eficiente gracias a la digitalización de las transacciones permitiendo a los participantes acceder a la información sobre la trazabilidad del

producto desde su origen hasta el consumidor final. Este sistema trajo muchos beneficios para todos los participantes de la cadena de suministro [25]:

- Trazabilidad: Permite rastrear cada etapa del producto de la cadena de suministro.
- Certificaciones: Puedes añadir certificados de productos, como su calidad o condiciones de conservación y hacerlos disponibles de inmediato a los participantes de la cadena.
- Entrada y acceso de datos: La plataforma facilita el intercambio de datos entre las organizaciones participantes de forma segura.
- Seguridad: Proporciona una gran seguridad de los datos, protegiéndolos contra amenazas.

  Además, IBM Food Trust está construido sobre Hyperledger Fabric, permitiendo que la red sea privada y controlada.

6.- DISEÑO DE LA SOLUCIÓN Y RESULTADOS

Al ser un proyecto largo que depende de muchas empresas externas, durante mi estancia en la empresa los datos reales no estaban disponibles, por lo que hice una red Blockchain con datos ficticios sobre la trazabilidad de los productos.

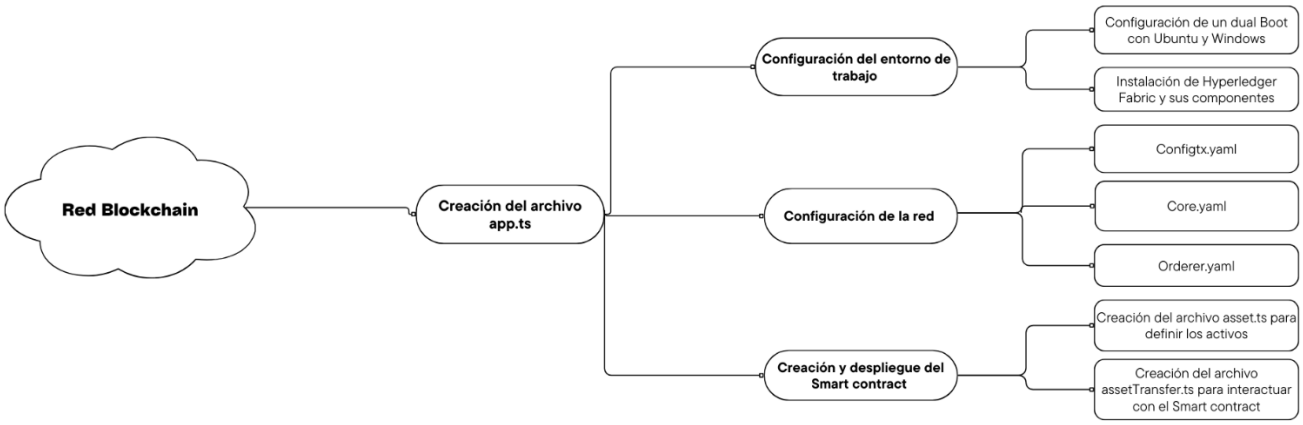

FIGURA 6: ESQUEMA PARA LA REALIZACIÓN DE UNA RED BLOCKCHAIN

Lo primero que hice fue un Dual Boot, es decir tener dos sistemas operativos diferentes en el mismo ordenador, en mi caso fue Linux y Windows. En mi caso utilice Ubuntu para Linux. Este proceso implicó dividir el disco duro en particiones, una para Windows y otra para Linux, de forma que cada sistema operativo tuviera su propio espacio, dedicando asi el entorno de Linux para desarrollar y ejecutar la red Blockchain.

Una vez hecho eso, tuve que instalar el Hyperledger Fabric. Para ello, seguí los pasos detallados en la documentación oficial de Hyperledger [26], que incluía la instalación de varios componentes esenciales como el **Docker**, para poder crear un entorno aislado para la red Hyperledger Fabric , el **Docker Compose,** para gestionar aplicaciones que utilizan varios Docker en un archivo YAML y poder iniciar, detener o gestionar todos los contenedores, mediante el comando: **docker-compose up/down/restart** y las imágenes de Hyperledger Fabric , que son plantillas que contienen todos los componentes esenciales de la Blockchain, como los nodos o las funciones de la red.

También se nos crea una carpeta llamada test-network que es un entorno de prueba que contiene scripts y configuraciones para realizar la red Blockchain, por lo que decidí realizar aquí mi proyecto modificando los scripts en función de lo que fuera necesario.

Una vez instalado todos los componentes del Hyperledger Fabric, nos falta por instalar el Hyperledger Fabric CA, para poder gestionar la autenticación de los certificados digitales para los participantes de la red.

Para instalarlo vamos al siguiente enlace: https://github.com/hyperledger/fabric-ca/releases y nos descargamos el binario más reciente para mi sistema operativo [27]. Una vez descargado cree una subcarpeta llamada fabric-ca-client, la cual pegue en la carpeta test-network.

Una vez hecho esto, inicie el servidor CA, desde con el siguiente comando: */fabric-ca-server init -b <ADMIN_USER>:<ADMIN_PWD>* asignando jorge como ADMIN_USER y metricsalad como contraseña, asignándome como administrador de la red y creando todos los certificados digitales necesarios.

Este proyecto esta desarrollado sobre fabric-samples, que es el directorio principal del proyecto. Dentro de esta carpeta podemos encontrar un gran número de carpetas que definen toda la configuración de la red, por ejemplo, encontramos la carpeta de config donde encontramos los archivos importantes para la configuración del canal:

- **Configtx.yaml:** En este archivo se definen como se configuran los canales, orderer y otros parámetros de la red, como, por ejemplo, como se comunican los nodos. De este código quiero destacar dos partes:

  - La primera de ellas sería el **OrdererEndpoints**, donde se definen los hosts de la red:

    OrdererEndpoints:

    - "Host1: 7050"

    - "Host2: 7050"

    - "Host3: 7050"

    Es decir, los nodos orderer están escuchando del puerto 7050 para aceptar las conexiones de otros nodos.

    -La segunda parte que quiero destacar son los **AnchorPeers** que se utilizan para la comunicación entre los nodos dentro de la red, asegurando que las transacciones se envíen correctamente.

    AnchorPeers:

    Host: peer0.org1.example.com

Port: 7050

Host: peer1.org1.example.com

Port: 7050

Host: peer2.org1.example.com

Port: 7050

En este archivo, definí 3 nodos (peer0, peer1 y peer2) aunque posteriormente utilicé únicamente 2 de ellos para el despliegue y ejecución del Smart contract, para ahorrar recursos durante la fase inicial de pruebas. A pesar de usar solo dos nodos, la configuración permite una expansión de la red al poder añadir el tercero cuando sea necesario.

- **Core.yaml:** En este archivo se especifican las configuraciones que afectan al comportamiento de los nodos. Dentro de este archivo en la parte de peer podemos encontrar 3 partes muy interesantes:

  - **listenAddress**: 0.0.0.0:7051: Especifica la dirección IP y puerto el peer escucha las conexiones de otros nodos o clientes, en este caso en el puerto 7051.

  - **chaincodeListenAddress:** 0.0.0.0:7052: Especifica la dirección de escucha para las conexiones del Smart contract, también llamado chaincode.

  - **gateway:** Intermediario que facilita la comunicación entre los clientes y los nodos dentro de la red. Es el punto de entrada para que los clientes puedan enviar y recibir transacciones.

    enabled: true

    endorsementTimeout: 30s: Tiempo máximo que el Gateway espera para recibir una respuesta de los nodos que validan las transacciones.

    broadcastTimeout: 30s: Tiempo máximo que el Gateway espera para recibir una respuesta de los nodos que ordenan las transacciones

    dialTimeout: 2m: Tiempo máximo que el Gateway espera para establecer una conexión con otro nodo en la red. Sino se logra establecer se genera un error de conexión.

- **Orderer.yaml:** En este archivo se configuran una serie de parámetros para definir el comportamiento de los orderers (nodos que ordenan las transacciones):

    - **ListenAddress:** 127.0.0.1: Configura la dirección IP en la que el orderer escucha, en este caso solo se admitirán conexiones locales (localhost).

    - **ListenPort:** 7050: Establece el puerto de escucha para las conexiones de otros nodos en 7050.

Una vez ya modificados y conocidos los archivos de configuración, podemos levantar la red, gracias a un archivo que nos da Hyperledger Fabric, el archivo network.sh, que se encuentra en la carpeta de test-network. Este archivo interactúa con los archivos de configuración explicados anteriormente (configtx, core y orderer) y tiene varias funciones como las que permiten iniciar la red Blockchain o crear el canal.

- **Iniciar la red:** function networkUp: Antes de iniciar cualquier componente de la red, esta función verifica que este todo configurado, como los dockers o certificados, si no esta configurado los crea. Una vez ese ha asegurado de que todo este correcto utiliza el Docker para levantar los componentes de la red. Estos dockers ejecutan los nodos y demás elementos de la red.

- **Crear el canal:** function createChannel: Esta función se encarga de crear un canal y gestionar la participación de los nodos. Un canal es la parte de la red donde se realizan las transacciones entre participantes. Esta función antes de crear el canal verifica que la red este en funcionamiento, gracias a esta línea de código: **if ! $CONTAINER_CLI info > /dev/null 2>&1; then** , que comprueba que el Docker esté en funcionamiento.

Aparte de estas funciones el network.sh también están las variables definidas como COMPOSE_FILE_BASE, COMPOSE_FILE_COUCH, COMPOSE_FILE_CA, que indican los archivos YAML de Docker Compose que se utilizarán para levantar los dockers necesarios para la red de Hyperledger Fabric. Estos archivos contienen las definiciones de los contenedores que incluyen los nodos (peer) y las autoridades de certificación (CA) necesarias:

COMPOSE_FILE_BASE=**compose-test-net.yaml**

COMPOSE_FILE_CA=compose-ca.yaml

El archivo network.sh interactúa con el Docker Compose para definir la configuración de los nodos, en este caso en el archivo **docker-compose-test-net.yaml ,** se encuentra dentro de la carpeta compose/Docker y está configurado para dos nodos solamente , por lo que si más tarde para añadir otro nodo solo tendríamos que ampliar este código.

Si esta en funcionamiento ejecuta el script createChannel.sh, que esta dentro de la carpeta de scripts para crear el canal: scripts/createChannel.sh $CHANNEL_NAME $CLI_DELAY $MAX_RETRY $VERBOSE $bft_true.

Vemos entonces que el script de network.sh llama al createChannel.sh para poder crear el canal una vez se ha comprobado que el Docker este activo, por lo que voy a explicar las dos partes mas importantes de este código:

- **Genesis Block:** El genesis block o bloque génesis es el primer bloque de una cadena de bloques, el inicial. Este contiene la configuración de la red y es el punto de partida de todas las transacciones que ocurren en el canal:

  createChannelGenesisBlock() {

  setGlobals 1

  which configtxgen

  if ["$?" -ne 0]; then

  fatalln "configtxgen tool not found."

  fi

  local bft_true=$1

  set -x

  if [$bft_true -eq 1]; then

  configtxgen -profile ChannelUsingBFT -outputBlock ./channel-artifacts/${CHANNEL_NAME}.block -channelID $CHANNEL_NAME

  else

  configtxgen -profile ChannelUsingRaft -outputBlock ./channel-artifacts/${CHANNEL_NAME}.block -channelID $CHANNEL_NAME

  fi

  res=$?

  { set +x; } 2>/dev/null

  ***verifyResult $res** "Failed to generate channel configuration transaction..."*

}

Paso a explicar las partes más importantes de este fragmento del código:

- setGlobals 1: Configura las variables necesarias para que la red sepa que nodo se utiliza en este script, en este caso indica que es el peer 0 de la organización 1, es decir el peer0.org1, que como podemos ver en el configtx.yaml es el primer nodo que hemos definido en la red.
- configtxgen -profile ...: En esta línea se utiliza el configtxgen para crear el bloque genesis (. block)
- verifyResult $res: Comprueba si se creo correctamente el bloque génesis del canal.

- **CreateChannel:** Esta función crea el canal, añadiendo los orderers:

```
createChannel() {
        local rc=1
        local COUNTER=1
        local bft_true=$1
        infoln "Adding orderers"
        while [ $rc -ne 0 -a $COUNTER -lt $MAX_RETRY ] ; do
                sleep $DELAY
                set -x
    . scripts/orderer.sh ${CHANNEL_NAME}> /dev/null 2>&1
    if [ $bft_true -eq 1 ]; then
      . scripts/orderer2.sh ${CHANNEL_NAME}> /dev/null 2>&1
      . scripts/orderer3.sh ${CHANNEL_NAME}> /dev/null 2>&1
      . scripts/orderer4.sh ${CHANNEL_NAME}> /dev/null 2>&1
    fi
                res=$?
```

```
            { set +x; } 2>/dev/null

            let rc=$res

            COUNTER=$(expr $COUNTER + 1)

        done

        cat log.txt

        verifyResult $res "Channel creation failed"

    }
```

A continuación voy a explicar las partes más importantes de este fragmento del código:

- Local rc=1: Define una variable llamada rc para controlar si se creó el canal.
- while [$rc -ne 0 -a $COUNTER -lt $MAX_RETRY]: Intenta agregar los orderes a la red hasta que se logre o se llegue al número máximo de intentos (MAX_RETRY), que por defecto está configurado en 5.
- verifyResult $res "Channel creation failed": Verifica que se pudo crear el canal. Si no lo fue, muestra un mensaje de error.

Una vez visto los scripts neccesarios para levantar la red, podemos ejecutar el comando: **./network.sh up createChannel -c mychannel -ca**

- ./network.sh: Ejecuta el script network.sh y con el up se especifica que se quere levantar la red, es decir crear y ejecutar los Dockers.
- createChannel: Ejecuta el script de createChannel para crear el canal donde se ejecutarán las transacciones.
- -c mychannel: Indica el nombre del canal que se va a crear
- -ca: Indica que se van a utilizar los certificados de autoridad

Una vez ejecutado este comando se nos habrá creado la red, con los siguientes componentes:

- Contenedores Docker: Se han creado los Docker necesarios para ejecutar los diferentes nodos. Estos dockers o contenedores incluyen:

    1. Peer nodes: Son los nodos de los participantes de la red que mantienen una copia del ledger (todas las transacciones que se han realizado en la red), como comenté anteriormente la red esta configurada para crear tres nodos, pero solamente he creado dos (llamados peer0.org1.example.com y peer0.org2.example.com) como viene definido en el archivo de docker-compose-test-net.yaml.

    2. Orderers: Los dockers como orderer.example.com son los responsables de gestionar las transacciones en la red.

    3. Certificados de autoridad: Los dockers fabric-ca orderer y fabric-ca-org1 generan los certificados de autentificación para los participantes de la red

- Canal: Se ha creado un canal llamado mychannel donde los participantes realizan las transacciones de forma privada.

- Certificados de Autoridad (CA): Al haber especificado el parámetro -ca se han generado y utilizado los certificados para gestionar las identidades y permisos de la red.

También podemos ver que se nos ha creado el bloque génesis, por lo que la red se ha levantado y funciona correctamente:

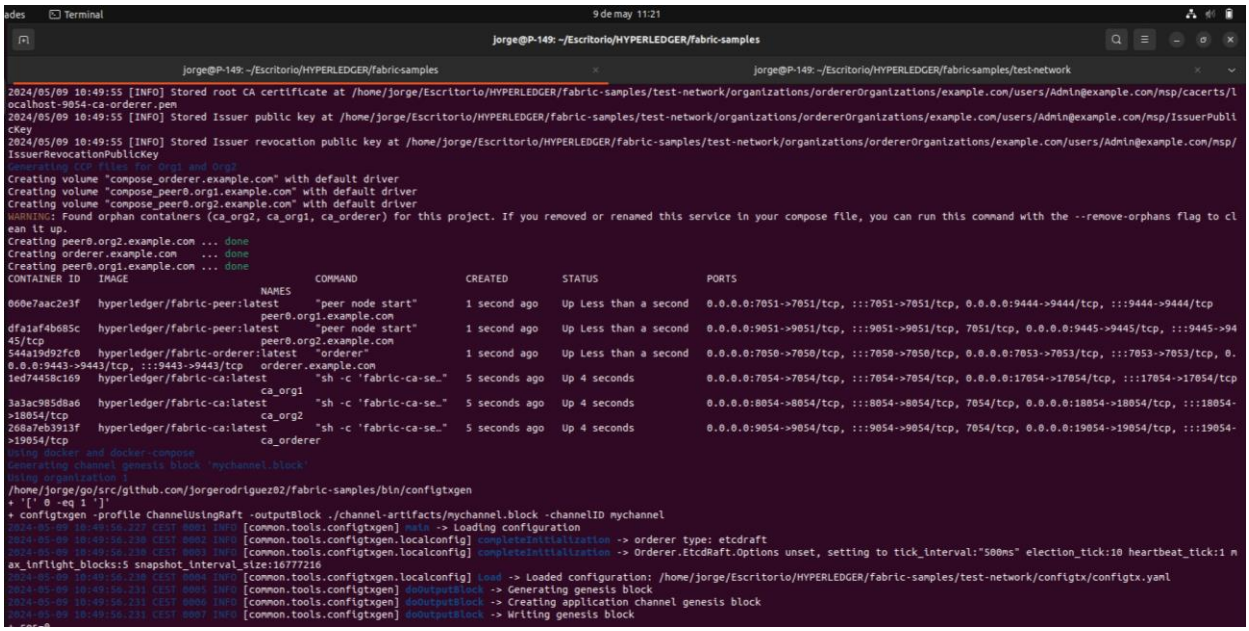

FIGURA 7: COMPONENTES DE LA RED

Una vez creada la red, el siguiente paso es la creación del Smart contract (contrato inteligente) mediante el lenguaje typescript con el uso de unos datos ficticios.

Dentro de la carpeta de asset-transfer-basic/Smart contract-typescript/src encontramos los archivos necesarios del Smart contract.

Por un lado, tenemos el archivo **asset.ts,** donde se definen los datos que se gestionaran dentro del ledger:

```
@Property()
public ID: string;
@Property()
public FechaCosecha: string;
@Property()
public FechaRecoleccion: string;
@Property()
public LugarCosecha: string;
@Property()
```

public VariedadFruta: string;

@Property()

public SesionesRiego: number;

@Property()

public SesionesAbonado: number;

@Property()

public Fertilizantes: string;

@Property()

public Plaguicidas: string;

@Property()

public Peso: number;

@Property()

public ClienteDestino: string;

@Property()

public Agricultor: string;

En este archivo podemos encontrar la estructura de datos que se almacenará en el ledger. Cada dato viene precedido por **@Property()** que significa que quiero que se guarde el dato en el ledger de la Blockchain.

Por otro lado, tenemos el archivo **assetTrasnfer.ts** donde tenemos varias funciones para poder interactuar con el Smart contract. Este archivo permite realizar diferentes acciones sobre los activos definidos anteriormente en el archivo **asset.ts:**

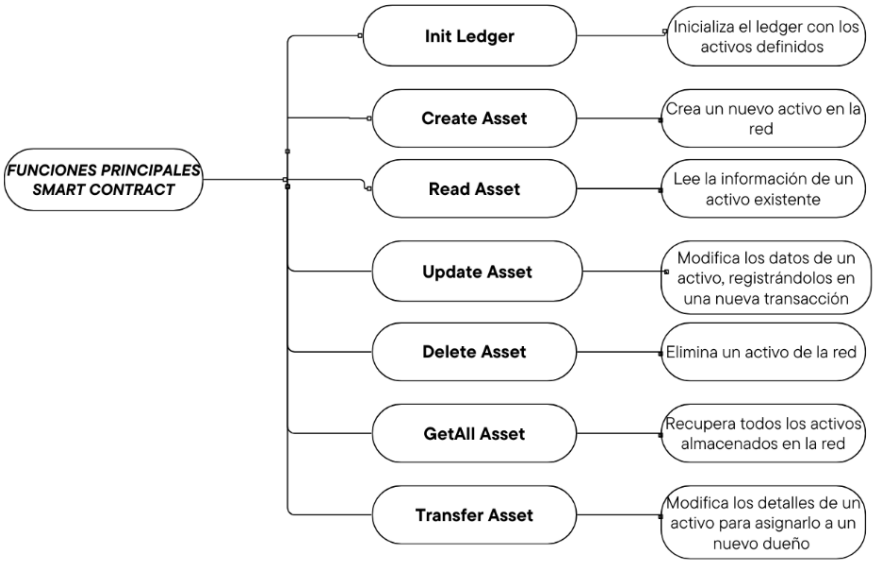

FIGURA 8: FUNCIONES DEL SMART CONTRACT

- **Función InitLedger:** Inicializa el ledger asignando valores a cada activo y se almacenan en la Blockchain con un ID único:
  1.- Dentro del array assets se definen los activos que se van a guardar en el ledger, como podría ser la variedadFruta, Peso o varios más.

```
public async InitLedger(ctx: Context): Promise<void> {
  const assets: Asset[] = [
    {
      ID: 'asset1',
      FechaCosecha: '2024-05-10',
      FechaRecoleccion: '2024-05-15',
      LugarCosecha: 'Finca A',
      VariedadFruta: 'Manzana',
      SesionesRiego: 5,
      SesionesAbonado: 2,
      Fertilizantes: 'FertA',
      Plaguicidas: 'PlagA',
      Peso: 100,
      ClienteDestino: 'Distribuidor XYZ',
      Agricultor: 'Agricultor1'
    },
    {
```

```
            ID: 'asset2',
            FechaCosecha: '2024-05-11',
            FechaRecoleccion: '2024-05-16',
            LugarCosecha: 'Finca B',
            VariedadFruta: 'Naranja',
            SesionesRiego: 3,
            SesionesAbonado: 1,
            Fertilizantes: 'FertC',
            Plaguicidas: 'PlagC',
            Peso: 80,
            ClienteDestino: 'Distribuidor ABC',
            Agricultor: 'Agricultor2'
        }
    ];
```

2.- Se hace un bucle for para recorrer el array de assets y almacenar cada activo en el ledger utilizando: **await ctx.stub.putState**

```
    for (const asset of assets) {
        asset.docType = 'asset';
        await ctx.stub.putState(asset.ID, Buffer.from(stringify(sortKeysRecursive(asset))));
        console.info(`Asset ${asset.ID} initialized`);
    }
```

Una vez creado el Smart contract y todas las funciones para interactuar con él, hay que instalarlo en los nodos de la red, para ello ejecutaremos el script de deployCC, que se encuentra dentro de la carpeta de scripts de test-network.

Este script es el que se utiliza para instalar y aprobar el Smart contract en la red:

1. Se empaqueta el chaincode para que sea instalado en los nodos:
   ./scripts/packageCC.sh $CC_NAME $CC_SRC_PATH $CC_SRC_LANGUAGE $CC_VERSION
2. Se instala en los nodos:
   installChaincode 1
   installChaincode 2
3. Se tiene que aprobar el Smart contract:
   approveForMyOrg 1
   approveForMyOrg 2
4. Una vez aprobado, se confirma el Smart contract en la red, registrando el contrato para su ejecución:
   commitChaincodeDefinition 1 2

5. Inicializa el ledger:
   chaincodeInvokeInit 1 2

Por tanto, si añadimos un tercer nodo, también tendremos que modificar esta parte del código, haciendo que también se instale el Smart contract en este nuevo nodo.

Para ejecutar este script utilizamos el siguiente comando:

./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-typescript/ -ccl typescript

Donde basic es el nombre del Smart contract, -ccp ../asset-transfer-basic/chaincode-typescript/ especifica la ruta donde se encuentra el Smart contract y -ccl typescript indica el lenguaje de programación utilizado para escribir el Smart contract.

En la figura (6) podemos ver como los nodos han aprobado el Smart contract

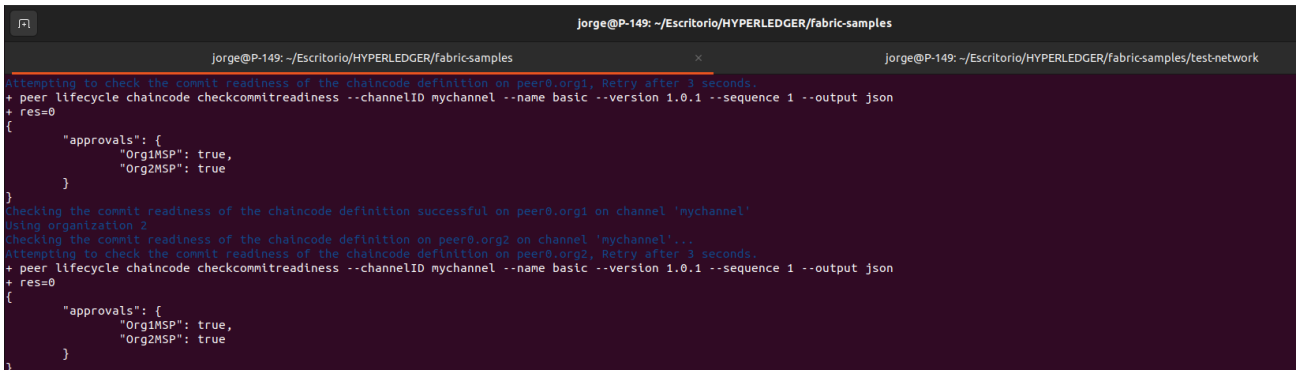

FIGURA 9: APROBACIÓN DEL SMART CONTRACT

Ahora ya tenemos el Smart contract instalado y aprobado por los nodos, por lo que el siguiente paso seria configurar la aplicación para interactuar con él.

En el directorio asset-transfer-basic/application-gateway-typescript/, encontramos el archivo package.json, el cual es muy importante ya que nos define todas las librerías necesarias para que la aplicación funcione correctamente, asi como algunos comandos predefinidos para ejecutar desde el terminal:

- Dependencias:
  "dependencies": {
  "@grpc/grpc-js": "^1.9.7", Librería para la comunicación entre cliente y nodo de Hyperledger Fabric.
  "@hyperledger/fabric-gateway": "~1.4.0", Librería principal para interactuar con la red de Hyperledger Fabric.
  }

- Dependencias de desarrollo:
  "devDependencies": {
  "@tsconfig/node18": "^18.2.2",
  "@types/node": "^18.18.6",
  "@typescript-eslint/eslint-plugin": "^6.9.0",
  "@typescript-eslint/parser": "^6.9.0",
  "eslint": "^8.52.0",
  "typescript": "~5.2.2"
  }
- Comandos o scripts:
  "scripts": {
  ***"build": "tsc**",* Genera los archivos JavaScript
  "build:watch": "tsc -w",
  "lint": "eslint . --ext .ts",
  "prepare": "npm run build",
  "pretest": "npm run lint",
  ***"start": "node dist/app.js**",* Ejecuta la aplicación de Node.js desde el archivo app.js
  },

El comando build transforma el código Typescript en JavaScript, ya que los nodos solo entienden JavaScript.

El comando npm start ejecuta el archivo dist/app.js (después de que el código ha sido compilado) iniciando la aplicación que se conecta a la red y empieza a interactuar con el Smart contract.

En el caso de que no estén instaladas todas las librerías, podemos ejecutar el comando *npm install* para que sean descargadas.

Una vez tenemos todas las librerías instaladas podemos ejecutar la aplicación con el comando *npm start.*

Al ejecutar este comando se compilará el código Typescript y se generará un archivo en la carpeta dist llamado *app.js.* Este archivo es el que interactúa con la red de Hyperledger Fabric y ejecuta las funciones definidas en el Smart contract.

En la figura 7 podemos ver el resultado de ejecutar el comando *npm start*

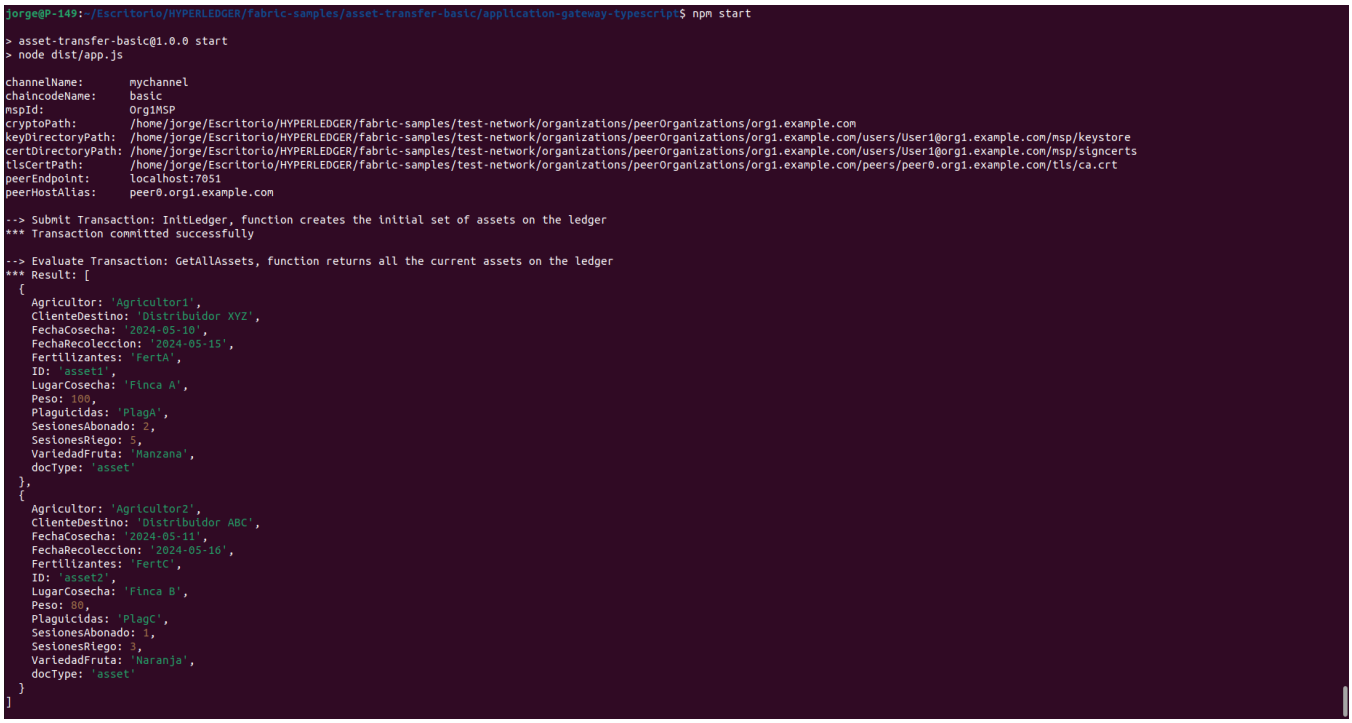

FIGURA 10: RESULTADO DE LA EJECUCIÓN DE LA APP

7.- ANALISIS DEL RESULTADO

Como he expuesto anteriormente, este trabajo se centra en la trazabilidad de los productos agrícolas utilizando una red de Blockchain para aumentar su seguridad y transparencia de la trazabilidad de los productos. La creación de esta red Blockchain permite gestionar de manera mas eficiente el ciclo de vida de un producto, asegurando que la información relacionada con este sea inmutable y solo accesible para los usuarios autorizados.

Para la implementación de la red se ha diseñado una arquitectura compuesta por:

- Canales y nodos: La red esta organizada en un canal que conecta dos nodos principales (peer). Estos nodos son los encargados de almacenar el ledger (libro mayor) y ejecutar las transacciones. El uso de canales garantiza que las transacciones sean privadas, ofreciendo un nivel adicional de seguridad.

- Nodo orderer: Que garantiza que las transacciones estén ordenadas correctamente antes de enviarlas a los nodos.

- Certificados de autoridad: Estos certificados permiten gestionar las identidades de los usuarios y controlan el acceso a la red permitiendo el acceso solo a personas autorizadas.

El Smart contract desarrollado en este proyecto está escrito en Typescript y define los activos (assets) que se registrarán en la red Blockchain. Estos activos se definen en función de la investigación realizada a las empresas.

Este Smart contract tiene varias funciones importantes con las que poder interactuar como:

- Crear activos: Permite crear activos en la red asegurando que se registren con un ID único y que se validen antes de ser almacenados en el ledger.
- Leer activos: Permite leer los datos almacenados en la red.
- Actualizar activos: Permite modificar los datos asociados a un activo, creando una nueva transacción con la información actualizada, manteniendo los registros anteriores.
- Eliminar activos: Permite eliminar activos de la red.

Al implementar estas funciones en una red Blockchain nos aseguramos de que los datos sean inmutables una vez son registrados. Esto significa que si un activo es creado no se puede modificar.

Los resultados obtenidos indican que la red Blockchain funciona según lo esperado, pudiendo gestionar las transacciones y registrar todos los datos en el ledger.

El uso de certificados de autoridad (CA) garantiza que solo los usuarios autorizados puedan interactuar con la red, por tanto, cree una identidad llamada jorge y la asigne como usuario administrador con la contraseña metricsalad. Esta implementación de los certificados de autoridad es importante para garantizar que solo los usuarios autorizados puedan interactuar con los activos y las transacciones dentro de la red, mejorando la seguridad de la red.

La implementación del sistema permite verificar que la red Blockchain funciona según lo esperado, mostrando las siguientes características:

1.- La red es capaz de gestionar las transacciones de forma eficiente, asegurando que todas sean registradas en el ledger

2.- El uso de canales y certificados de autoridad garantizan que las transacciones sean privadas y solo accesibles para los usuarios autorizados.

3.- Aunque el sistema actual cuente con 2 nodos, su diseño permite añadir nuevos nodos y canales conforme crezca la red.

4.- La red permite registrar todo el ciclo de vida de los productos agrícolas desde su cosecha hasta el destino final, por lo que ofrece una transparencia clave para aumentar la confianza del consumidor.

A pesar de los objetivos logrados, el sistema presenta algunas limitaciones que podrían ser solucionadas en trabajos futuros:

1.- Aunque la red funciona según lo esperado, los datos utilizados son ficticios, por lo que añadir datos reales permitirá validar el correcto funcionamiento de esta.

2.- La incorporación de los sensores IoT permitiría monitorear activos en tiempo real, lo que mejoraría la trazabilidad de los productos.

En resumen, este trabajo ha demostrado la viabilidad del uso de Blockchain para la trazabilidad de los productos agrícolas. Esta red asegura la transparencia e inmutabilidad de los datos, aumentando la confianza del consumidor.

8.- RELACIÓN DEL TRABAJO CON LOS ODS

Objetivos de Desarrollo Sostenibles		Alto	Medio	Bajo	No Procede
ODS 1.	Fin de la pobreza.				x
ODS 2.	Hambre cero.				x
ODS 3.	Salud y bienestar.				x
ODS 4.	Educación de calidad.			x	
ODS 5.	Igualdad de género.				x
ODS 6.	Agua limpia y saneamiento.				x
ODS 7.	Energía asequible y no contaminante.				x
ODS 8.	Trabajo decente y crecimiento económico.				x
ODS 9.	Industria, innovación e infraestructuras.	x			
ODS 10.	Reducción de las desigualdades.				x
ODS 11.	Ciudades y comunidades sostenibles.				x
ODS 12.	Producción y consumo responsables.	x			
ODS 13.	Acción por el clima.			x	
ODS 14.	Vida submarina.				x
ODS 15.	Vida de ecosistemas terrestres.				x
ODS 16.	Paz, justicia e instituciones sólidas.				x
ODS 17.	Alianzas para lograr objetivos.				x

TABLA 2 : GRADO DE RELACIÓN DEL TRABAJO CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE

Este trabajo tiene relación con varios objetivos de desarrollo sostenible.

En primer lugar, el proyecto tiene una gran relación con el **ODS 9, industria, innovación e infraestructura**, ya que Blockchain puede mejorar el sector agrícola gracias a la digitalización y automatización de la trazabilidad de sus productos, reduciendo los intermediarios, provocando una mayor fiabilidad y transparencia.

También tiene una alta implicación con el **ODS 12, producción y consumo responsable**, ya que permite asegurar la trazabilidad de los productos, garantizando que se cultiven y procesen de forma correcta. La capacidad de rastrear los productos agrícolas desde su origen hasta el consumidor permite garantizar que las empresas mantengan prácticas sostenibles en la producción. Esto mejora la confianza del consumidor. Aunque nuestro proyecto no está estrictamente relacionado con el **ODS 13, Acción por el clima,** puede jugar un papel importante en la reducción de emisiones de gases de efecto

invernadero, gracias a que se puede hacer una monitorización de las emisiones de gases de efecto invernadero los productores pueden tomar medidas más sostenibles para reducir los gases.

Por último, podemos relacionar el proyecto con el **ODS 4, educación de calidad**, ya que, al usar una tecnología innovadora, los agricultores pueden recibir formación y certificados verificables para mejorar el sector.

9.-CONCLUSIONES

En este trabajo se ha conseguido cumplir con los objetivos planteados, creando un sistema de trazabilidad enfocado en productos agrícolas gracias a la tecnología Blockchain.

La red diseñada permite almacenar y recuperar datos de forma segura y transparente, gracias a la creación e implementación de un Smart contract en la red Blockchain donde se recogen los datos asociados al ciclo de vida de los productos, es decir la trazabilidad de un producto, asegurando transparencia y seguridad de la información. La red garantiza que los datos de cada producto, como la fecha de cosecha, el lugar de origen, los procesos aplicados durante su ciclo de vida y su destino sean accesibles y transparentes. Además, la tecnología Blockchain garantiza que la información no se pueda modificar ni eliminar una vez es registrada.

El uso de esta tecnología facilita la gestión de los contratos entre las partes involucradas, sin necesidad de intermediarios, lo que reduce costos y mejora la productividad.

Con este trabajo se puede comprobar la viabilidad del uso de Blockchain para la trazabilidad de los productos, gracias a las ventajas que ofrece a nivel de transparencia y seguridad del producto.

En resumen, se demuestra el potencial que tiene la tecnología Blockchain para transformar el sector agrícola, aportando una plataforma segura para gestionar la trazabilidad de los productos y garantizando la calidad de los productos ofreciendo al consumidor una mayor confianza en los alimentos que consumen.

10.-TRABAJO FUTURO

Como trabajo a futuro, uno de los objetivos sería completar la implementación de la red Blockchain utilizando datos reales sobre la trazabilidad de los productos agrícolas. El uso de datos reales permitiría garantizar que la red es capaz de manejar situaciones del mundo real, lo que no solo mejoraría la confianza del consumidor, sino que también ofrecería un sistema más seguro para la trazabilidad del producto.

La integración de sensores IoT sería una de las principales mejoras a realizar en el proyecto, ya que permitiría monitorear los activos en tiempo real, registrando datos como la temperatura o humedad.

Otra mejora importante sería trasladar la red Blockchain a un proveedor de servicios en la nube como AWS, lo que haría que la red sea más fácil de usar, pero sobre todo para que la red esté disponible siempre para su uso, ya que no dependería de un ordenador local.

11.-REFERENCIAS

[1] «academy.binance,» [En línea]. Available: https://academy.binance.com/es/articles/history-of-blockchain.

[2] «ITDO,» [En línea]. Available: https://www.itdo.com/blog/que-son-los-bloques-en-la-tecnologia-blockchain/.

[3] «ITDO,» [En línea]. Available: https://www.itdo.com/blog/que-es-el-arbol-merkle-en-blockchain/.

[4] «101blockchain,» [En línea]. Available: https://101blockchains.com/hyperledger-sawtooth-vs-fabric/.

[5] N. Rodríguez, «101blockchain,» [En línea]. Available: https://101blockchains.com/es/historia-de-la-blockchain/.

[6] «IBM,» [En línea]. Available: https://www.ibm.com/es-es/topics/blockchain.

[7] J. S. Hurtado, «.iebschool,» 2024. [En línea]. Available: https://www.iebschool.com/blog/blockchain-cadena-bloques-revoluciona-sector-financiero-finanzas/. [Último acceso: 2024].

[8] B. Becher, «builtin,» [En línea]. Available: https://builtin.com/blockchain .

[9] «observatorioblockchain,» [En línea]. Available: https://observatorioblockchain.com/hypernifty/redes-blockchain-tipos/.

[10] N. Rodriguez, «101blockchain,» [En línea]. Available: https://101blockchains.com/es/uso-de-blockchain/.

[11] C. Simões, «ITDO,» [En línea]. Available: https://www.itdo.com/blog/que-es-la-criptografia-cual-es-su-papel-en-blockchain/ .

[12] «bit2me,» [En línea]. Available: https://academy.bit2me.com/que-son-los-smart-contracts/.

[13] «plain concepts,» 05 01 2023. [En línea]. Available: https://www.plainconcepts.com/es/smart-contracts/.

[14] «Santander,» 27 05 2022. [En línea]. Available: https://www.santander.com/es/stories/smart-contracts .

[15] [En línea]. Available: https://www.joakimvivas.com/crypto/smart-contracts-blockchain/.

[16] «unknowngravity,» [En línea]. Available: https://www.unknowngravity.com/articulos/ejemplos-y-aplicaciones-de-los-smart-contracts .

[17] «MetricSalad,» [En línea]. Available: https://metricsalad.com.

[18] «iebschool,» [En línea]. Available: https://www.iebschool.com/blog/las-mejores-plataformas-blockchain-para-el-desarrollo-de-aplicaciones-financieras-tecnologia/.

[19] N. Rodriguez, «101blockchain,» [En línea]. Available: https://101blockchains.com/es/hyperledger-or-ethereum/.

[20] [En línea]. Available: https://www.lfdecentralizedtrust.org/hyperledger-sawtooth-1-0.

[21] R. Díez, «linkedin,» [En línea]. Available: https://www.linkedin.com/pulse/aws-azure-o-google-cloud-c%C3%B3mo-decidir-ricardo-d%C3%ADez-m-/.

[22] [En línea]. Available: https://www.ceste.es/noticias/aws-vs-azure-vs-google-cloud-comparativa-de-las-3-plataformas-cloud-lideres/.

[23] «ausum.cloud,» [En línea]. Available: https://ausum.cloud/aws-iot-core/.

[24] [En línea]. Available: https://financialfood.es/carrefour-implanta-la-tecnologia-blockchain-con-sus-productos-bio/.

[25] «smcloud,» [En línea]. Available: https://www.smcloud.es/trazabilidad-alimentaria-y-transparencia-con-ibm-food-trust/.

[26] «Hyperledger fabric,» [En línea]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5/ .

12.-ANEXOS

Acceso a todos los códigos de la red Blockchain:
https://github.com/hyperledger/fabric-samples/tree/e79e96394dc9d06b66f2714e981c0b0ffd3f0cf9