UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Aplicación de un sistema experimental de blockchain orientado a la industria farmacéutica

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: García Such, Samuel

Tutor/a: Roca Martinez, Alicia

CURSO ACADÉMICO: 2023/2024

Resumen

Este Trabajo de Fin de Grado (TFG) presenta el diseño, desarrollo e implementación de un sistema de Blockchain en Python, enfocado en proporcionar una solución segura y eficiente para transacciones y operaciones de datos. El sistema presenta una arquitectura modular que integra diversos componentes como gestión de bloques, interacción con bases de datos SQL, criptografía avanzada y control de accesos y permisos de usuario. El núcleo del sistema reside en una cadena de bloques implementada en Python, que asegura la integridad y transparencia de las transacciones mediante el uso de algoritmos criptográficos y un mecanismo de consenso. Además, el sistema incorpora una interfaz de usuario intuitiva y un back-end robusto para manejar operaciones y transacciones de manera eficiente.

La evaluación del sistema se llevó a cabo a través de una serie de pruebas que incluyen funcionalidad, rendimiento, seguridad, usabilidad, mantenibilidad y conformidad con los requisitos. Los resultados demostraron la eficacia del sistema en términos de seguridad de los datos, rendimiento bajo carga y facilidad de uso, destacando su potencial para aplicaciones reales. Este trabajo contribuye al campo de la tecnología Blockchain, ofreciendo conocimiento sobre la implementación de sistemas de Blockchain en Python y su evaluación en un contexto práctico.

Abstract

This Final Degree Project (TFG) presents the design, development, and implementation of a Blockchain system in Python, aimed at providing a secure and efficient solution for data transactions and operations. The system is based on a modular architecture that integrates various components such as block management, interaction with SQL databases, advanced cryptography, and user access and permission control. The core of the system lies in a Blockchain implemented in Python, which ensures the integrity and transparency of transactions through cryptographic algorithms and a consensus mechanism. In addition, the system incorporates an intuitive user interface and a robust backend to efficiently handle operations and transactions.

The system evaluation was conducted through a series of tests including functionality, performance, security, usability, maintainability, and compliance with requirements. The results demonstrated the system's effectiveness in terms of data security, performance under load, and ease of use, highlighting its potential for real-world applications. This work contributes to the field of Blockchain technology, offering insights into the implementation of Blockchain systems in Python and their evaluation in a practical context.

Resum

Aquest Treball de Fi de Grau (TFG) presenta el disseny, desenvolupament i implementació d'un sistema de Blockchain en Python, enfocat a proporcionar una solució segura i eficient per a transaccions i operacions de dades. El sistema es basa en una arquitectura modular que integra diversos components com la gestió de blocs, la interacció amb bases de dades SQL, la criptografia avançada i el control d'accés i permisos d'usuari. El nucli del sistema resideix en una cadena de blocs implementada en Python, que assegura la integritat i transparència de les transaccions mitjançant algoritmes criptogràfics i un mecanisme de consens. A més, el sistema incorpora una interfície d'usuari intuïtiva i un backend robust per a manejar operacions i transaccions de manera eficient.

L'avaluació del sistema es va dur a terme a través d'una sèrie de proves que inclouen funcionalitat, rendiment, seguretat, usabilitat, mantenibilitat i conformitat amb els requisits. Els resultats van demostrar l'eficàcia del sistema en termes de seguretat de les dades, rendiment sota càrrega i facilitat d'ús, destacant el seu potencial per a aplicacions reals. Aquest treball contribueix al camp de la tecnologia Blockchain, oferint visió sobre la implementació de sistemes de Blockchain en Python i la seua avaluació en un context pràctic.

Tabla de contenidos

1.- Introducción

En una era caracterizada por avances tecnológicos exponenciales y una digitalización cada vez más profunda de todos los sectores industriales, la tecnología Blockchain emerge como una revolución en la forma en que gestionamos y protegemos la información. Originalmente concebida para criptomonedas como Bitcoin, su aplicación ha trascendido estas fronteras, revelándose como una herramienta poderosa con un amplio espectro de aplicaciones en diversas industrias.

Este Trabajo Fin de Grado se adentra en el mundo de la tecnología Blockchain y su aplicación específica en un sector de gran importancia: el sector farmacéutico. Dicha industria, con sus intrincadas cadenas de suministro, estrictas regulaciones y la necesidad imperiosa de asegurar la seguridad del paciente, representa un área ideal para explorar las capacidades innovadoras de la tecnología Blockchain. El trabajo se centra en el diseño, desarrollo y evaluación de un sistema experimental basado en Blockchain, adaptado específicamente para mejorar la seguridad, trazabilidad y eficiencia operativa en el sector farmacéutico.

La seguridad es un pilar fundamental en este contexto, donde la presencia de productos falsificados puede tener consecuencias fatales. La tecnología Blockchain ofrece un mecanismo robusto para asegurar la autenticidad de las transacciones realizadas con los medicamentos. Mediante un registro inmutable y permanente de las transacciones, cada producto puede ser rastreado hasta su origen, asegurando su autenticidad y seguridad. Esta característica es de vital importancia en un sector donde la confianza en la calidad y origen de los productos es esencial.

Otro aspecto a tener en cuenta es la trazabilidad en la cadena de suministro de medicamentos, conocida por su complejidad. La implementación de la tecnología Blockchain promete una trazabilidad total y en tiempo real de cada producto, desde su fabricación hasta que llega al consumidor final. Esta transparencia no solo optimiza la logística, sino que también facilita un cumplimiento normativo más eficaz y eficiente, algo crítico en un sector altamente regulado.

Finalmente, la eficiencia operativa es un objetivo de gran importancia. La automatización de procesos a través de contratos inteligentes y la eliminación de intermediarios pueden resultar en una reducción significativa de los costos y tiempos de operación. La tecnología Blockchain, con su capacidad para simplificar y acelerar procesos, tiene el potencial de transformar radicalmente la manera en que se manejan las operaciones diarias en este sector.

1.1- Descripción del proyecto

Este proyecto se centra en la creación e implementación de una infraestructura Blockchain diseñada específicamente para abordar y resolver posibles desafíos presentes en el sector farmacéutico. Como pueden ser duplicación de medicamentos, falsas transacciones o el no correcto registro de aquellas transacciones verídicas.

La infraestructura ha sido desarrollada utilizando Python, un lenguaje de programación elegido por su robustez, versatilidad y la rica biblioteca de recursos disponibles, lo que lo hace ideal para construir aplicaciones complejas y seguras. Python se destaca en este contexto por su capacidad para manejar eficientemente tanto los aspectos de back-end como los de procesamiento de datos.

El sistema emplea tecnologías de SQL para el almacenamiento y gestión de datos. Esta elección se debe a la eficiencia, escalabilidad y madurez de SQL en el manejo de grandes volúmenes de datos, una característica importante en el sector farmacéutico, donde se generan y procesan grandes cantidades de información crítica. La integración de SQL en el sistema asegura que la gestión de datos sea eficiente y segura, proporcionando un acceso rápido a la información y manteniendo al mismo tiempo la integridad y la trazabilidad de los datos.

El núcleo de la arquitectura del proyecto es la tecnología Blockchain, la cual ha sido implementada para crear un entorno de descentralizado y altamente seguro. La descentralización es un factor clave en la robustez del sistema, ya que elimina los puntos únicos de fallo y distribuye la información a través de una red, aumentando así la resistencia a manipulaciones y ataques cibernéticos.

La implementación de la tecnología Blockchain en este proyecto no solo asegura la transparencia y la trazabilidad de los productos, sino que también fortalece la confianza en toda la cadena de suministro. Al ofrecer una visibilidad completa del recorrido de los productos, desde la fabricación hasta el punto de venta, se minimizan las posibilidades de falsificación y se mejora la eficiencia operativa. Este enfoque innovador no solo representa un avance tecnológico, sino que también se alinea con la creciente demanda de soluciones más seguras y transparentes en la industria farmacéutica.

1.2- Alcances y objetivos del proyecto

El objetivo principal es demostrar cómo esta innovadora tecnología puede aplicarse eficazmente para resolver desafíos específicos del sector, como son la falsificación de medicamentos y las ineficiencias en la cadena de suministro. La meta es no solo probar la funcionalidad técnica del sistema Blockchain, sino también su habilidad para ofrecer soluciones concretas y efectivas a problemas que pueden estar presentes en el día a día de las transacciones en este sector como pueden ser las mencionadas previamente de falsificación de transacciones o duplicidad de medicamentos.

Un importante aspecto del proyecto es el desarrollo de un sistema robusto, seguro, amigable y accesible para los usuarios. Esto incluye la creación de una interfaz de usuario intuitiva, desarrollada con Tkinter, para facilitar la interacción entre los usuarios y el sistema Blockchain.

El alcance del trabajo abarca desde la conceptualización teórica de la solución hasta su implementación práctica. Este proceso incluye el diseño de la construcción de la cadena Blockchain, el desarrollo del código, las pruebas y la evaluación final, asegurando así que cada fase del sistema sea cuidadosamente planificada y ejecutada. La fase de implementación práctica del sistema es especialmente importante, ya que proporciona una validación real de la teoría y establece una base sólida para futuras mejoras y adaptaciones.

Aunque el foco principal del proyecto está en el sector farmacéutico, los principios y metodologías utilizados son potencialmente adaptables para ser aplicados en otros sectores que requieran altos estándares de seguridad y trazabilidad en sus procesos.

2.- Fundamentos teóricos

En este apartado haremos hincapié en los fundamentos teóricos sobre los que se apoya este trabajo. Vamos a indicar qué es un sistema de Blockchain, cuáles son los aspectos más importantes de esta tecnología, sus aplicaciones, y consideraciones futuras con respecto a esta.

2.1- Introducción a la tecnología Blockchain

La tecnología Blockchain, a menudo asociada con criptomonedas como Bitcoin, es en realidad una innovación tecnológica más amplia y fundamental. Su aplicación puede ser útil además de las finanzas digitales puede extenderse a otras industrias, transformando la manera en que almacenamos, gestionamos y validamos la información.

2.1.1- Orígenes y Evolución

Aunque el concepto de Blockchain como lo conocemos hoy tiene sus raíces en el trabajo de varios investigadores y desarrolladores, su aplicación práctica más conocida y su verdadero debut en el escenario mundial ocurrió en 2008 con la publicación del "white paper" de Bitcoin [1].

El documento, publicado por una persona o un grupo anónimo bajo el seudónimo de Satoshi Nakamoto introdujo Bitcoin como la primera criptomoneda descentralizada y además presentó la tecnología subyacente que la hacía posible: la Blockchain. La tecnología Blockchain de Bitcoin fue concebida como un registro digital público y descentralizado, capaz de registrar transacciones de manera segura y transparente sin la necesidad de una autoridad central.

La innovación clave de la tecnología Blockchain es su estructura de datos en la que los registros, agrupados en bloques, están enlazados y cifrados para proteger la seguridad e integridad de la información de transacciones. Cada bloque contiene un hash criptográfico del bloque anterior, el cual al mínimo cambio por pequeño que sea cambia completamente, creando una cadena ininterrumpida que es prácticamente a prueba de manipulaciones. Esta característica hace que la historia de cada transacción sea permanente y visible para todos los participantes de la red, garantizando la transparencia y la confianza entre las partes.

La evolución de la tecnología Blockchain ha dado lugar a diversas variantes y generaciones. La primera generación, representada por Bitcoin, se centró en las transacciones de criptomonedas.

La segunda generación, liderada por plataformas como Ethereum, introdujo el concepto de contratos inteligentes, que amplían el uso de la tecnología Blockchain más allá de las transacciones financieras simples para facilitar, verificar o negociar contratos de manera confiable y automática. Las generaciones posteriores de la tecnología Blockchain han buscado abordar desafíos como la escalabilidad, la velocidad de transacción y la interoperabilidad entre diferentes cadenas de bloques.

2.1.2- ¿Qué es la tecnología Blockchain?

La tecnología Blockchain, en su esencia, es una base de datos distribuida que funciona como un libro mayor digital. Esta innovación tecnológica se caracteriza por su capacidad para registrar y almacenar transacciones o cualquier tipo de dato de manera secuencial en "bloques". Cada bloque en la cadena contiene los datos de la transacción, y está enlazado de forma segura a los bloques anteriores mediante el uso de criptografía. Utilizando una función hash SHA-256 generamos un resumen de 256 bits de los datos de un bloque [2]. Este resumen constituye uno de los campos del siguiente bloque, con lo cual, si intentáramos falsificar cualquiera de los bloques de la cadena saltaría la alarma ya que habría una incoherencia en los datos. Se forma así una cadena continua e inquebrantable de registros. Esta estructura de bloques enlazados asegura que cada entrada en la cadena sea permanente y prácticamente inalterable, lo que resulta en un registro de transacciones extremadamente seguro y fiable.

2.1.3- Descentralización y Seguridad

La descentralización es una de las características más distintivas y poderosas de la tecnología Blockchain, diferenciándose fundamentalmente de las estructuras de bases de datos tradicionales. En los sistemas convencionales, los datos son almacenados y controlados por una entidad centralizada, como podría ser un banco, una empresa o un organismo gubernamental. Esta centralización, aunque eficiente en ciertos contextos, crea puntos únicos de vulnerabilidad. Un único punto de fallo, como un servidor central, puede ser un blanco para ataques cibernéticos, manipulación de datos, o incluso fallas internas, poniendo en riesgo toda la integridad del sistema.

En contraste, la tecnología Blockchain opera en una red distribuida de nodos, que generalmente son computadoras individuales repartidas geográficamente. Cada uno de estos nodos posee una copia completa de la cadena de bloques, funcionando de manera colectiva para validar y registrar nuevas transacciones y bloques. Esta distribución de la información significa que no existe un único punto de fallo. Para alterar cualquier registro en la Blockchain, un atacante tendría que modificar la copia de la cadena en la mayoría de los nodos simultáneamente, una tarea que resulta extremadamente difícil, si no imposible, dada la naturaleza del consenso distribuido y la criptografía utilizada.

En cuanto la seguridad para un sistema de Blockchain, tenemos que verificar que el sistema cumpla tres propiedades principales:

Verificar la integridad de los datos: Gracias a los hashes utilizados y a sus propiedades de resistencia preimagen, resistencia de colisiones y determinismo se puede verificar que los registros introducidos a la cadena de bloques no cambiarán la información que contienen y, en caso de hacerlo será fácilmente comprobable.

Autenticidad de operación: Otro aspecto importante es verificar la identidad del usuario que ha realizado una operación y comprobar que esta ha sido realmente realizada por él. Esto se consigue gracias al uso de una firma digital ECDSA basada en encriptación de curva elíptica (ECC) haciendo uso de criptografía de clave pública [3]. Gracias a esto podemos verificar que se cumpla la característica de no repudio, por la que si un usuario realiza una operación firmándola así con sus claves quede un registro inmutable de que este ha realizado la operación.

Disponibilidad: Se tiene que poder realizar y validar transacciones en el sistema en todo momento pese a la carga a la que se sujete el sistema, manteniéndose a prueba de ataques como los de denegación de servicio (DDoS), con los cuales un atacante puede aprovechar el colapso del sistema para modificar la cadena de datos o introducir operaciones maliciosas a la cadena.

2.1.3- Hash criptográfico

Un hash criptográfico es una función que convierte una entrada de longitud arbitraria (como un bloque de transacciones) en una salida de longitud fija y única. Las funciones de hash criptográfico se diseñan para tener ciertas características [4]:

Preimagen resistente: difícil de revertir el hash para encontrar la entrada original.

Resistencia a colisiones: computacionalmente difícil encontrar dos entradas diferentes que produzcan el mismo hash. La resistencia a colisiones se subdivide en dos categorías:

- **Colisión Débil (o Resistencia a la Segunda Preimagen):** es prácticamente imposible encontrar una segunda entrada que produzca el mismo hash que una entrada especificada.
- **Colisión Fuerte (o Resistencia a la Colisión):** es extremadamente difícil encontrar dos entradas diferentes que resulten en el mismo hash.

En la teoría, debido a la naturaleza finita de los hashes, siempre habrá colisiones. Sin embargo, un buen algoritmo de hash criptográfico asegura que estas colisiones sean tan improbables que no sean una preocupación práctica.

Eficiencia: El algoritmo tiene que ser computacionalmente rápido para procesar la entrada y producir el hash.

Determinismo: La misma entrada siempre produce la misma salida.

El algoritmo SHA-256 es ampliamente utilizado en la tecnología Blockchain debido a su capacidad para mantener la integridad y la estructura de cadenas de datos de manera eficiente por el tipo de operaciones que usa: xor, and, or, suma módulo 2^{32} y desplazamientos de bits. A continuación mostramos una iteración del algoritmo SHA-256, para completar la creación de un hash tendríamos realizar 80 iteraciones.

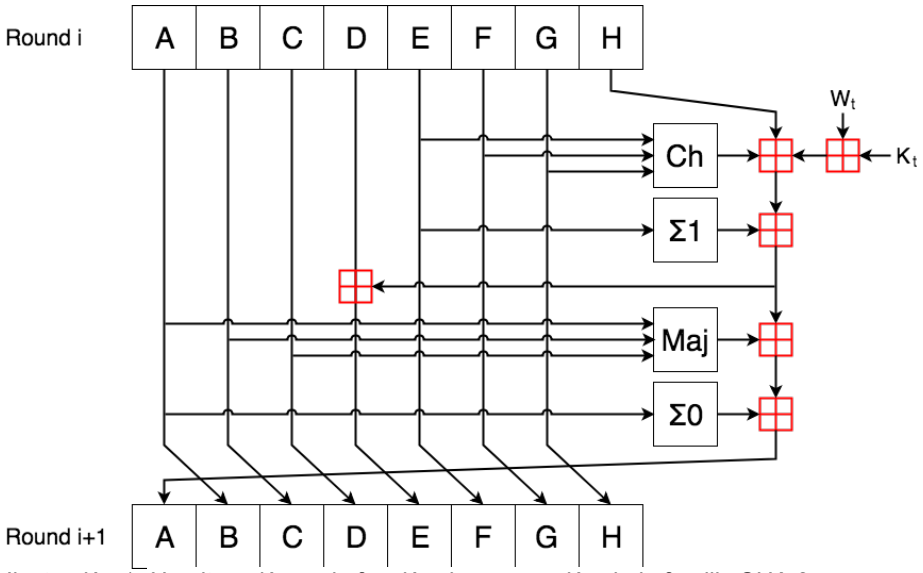


Ilustración 1- Una iteración en la función de compresión de la familia SHA-2

Como parte de la familia de algoritmos Secure Hash Algorithms, SHA-256 transforma cualquier entrada en un valor de hash único de 256 bits. La naturaleza determinista del algoritmo garantiza que una entrada específica produzca siempre el mismo hash. Además, su diseño como función unidireccional hace que sea computacionalmente inviable descubrir la entrada original a partir del hash, proporcionando así una capa sólida de seguridad. La sensibilidad del algoritmo a los cambios en la entrada significa que cualquier modificación, por mínima que sea, resulta en un hash completamente diferente, lo que contribuye a la detección temprana de cualquier alteración de datos. Por último, la resistencia a colisiones de SHA-256, tanto débiles como fuertes, asegura la rareza de que dos entradas distintas puedan producir el mismo hash, manteniendo así la unicidad y la autenticidad de cada bloque de datos dentro de la cadena.

2.1.4- Consensos y Validación

Un importante aspecto de la tecnología Blockchain es el concepto de consenso y validación, procesos esenciales para agregar nuevos bloques a la cadena. El consenso en una red Blockchain se refiere al proceso mediante el cual los nodos de la red acuerdan colectivamente la veracidad y la validez de las transacciones antes de agregarlas a la cadena. Este acuerdo se logra a través de diversos algoritmos de consenso, siendo el **Proof of Work (PoW)** y el **Proof of Stake (PoS)** los más reconocidos.

En el modelo **PoW**, utilizado por Bitcoin y otras criptomonedas, los nodos, conocidos como mineros, compiten entre sí para resolver complejos problemas matemáticos. El primero en resolver el problema y validar el bloque es recompensado con una cantidad de la criptomoneda. Este proceso valida y agrega el bloque a la cadena, asegura la red contra ataques maliciosos, ya que alterar la tecnología Blockchain requeriría una cantidad inmensa de poder computacional. En este trabajo hemos utilizado este consenso, aplicando una versión simplificada de la resolución de estos puzles.

Por otro lado, el **Proof of Stake (PoS)** representa un enfoque diferente. En lugar de depender de la capacidad computacional, el **PoS** selecciona validadores en proporción a sus tenencias de la moneda o token respectivo en la red. En este sistema, cuanto mayor sea la participación de un nodo, mayores serán sus posibilidades de ser elegido para validar un nuevo bloque.

Esta metodología reduce significativamente el consumo de energía en comparación con **PoW**, ya que no requiere de una gran cantidad de poder de procesamiento. Además, el **PoS** ofrece un mayor nivel de escalabilidad y eficiencia, haciéndolo atractivo para muchas nuevas criptomonedas y aplicaciones de Blockchain.

Ambos mecanismos, **PoW** y **PoS**, juegan un papel crucial en garantizar que todos los nodos de la red mantengan un consenso sobre el estado y la validez de la Blockchain, lo cual es fundamental para la integridad y seguridad del sistema.

2.1.5- Transparencia y Trazabilidad

La transparencia y la trazabilidad son aspectos fundamentales que diferencian a la tecnología Blockchain de otras tecnologías de gestión de datos y que la convierten en una herramienta valiosa en múltiples sectores.

En la tecnología Blockchain, cada transacción es registrada de forma transparente y está disponible para ser verificada por cualquier usuario que tenga acceso a la red. Esta transparencia es posible gracias a la naturaleza distribuida de la Blockchain, donde cada nodo de la red almacena una copia idéntica de la cadena.

Cuando se realiza una transacción, esta se transmite a toda la red, y tras ser validada, se añade a un bloque que se enlaza a la cadena. Este registro es público y permanente, lo que significa que cualquier usuario puede revisar las transacciones pasadas. Esta característica aporta una transparencia sin precedentes, y fomenta la responsabilidad y la confianza entre los usuarios, ya que todas las acciones quedan registradas de manera indeleble.

La inmutabilidad de los registros en la Blockchain proporciona trazabilidad. Una vez que un bloque es añadido a la cadena, la información contenida en él se vuelve prácticamente imposible de alterar, debido a la naturaleza criptográfica de cómo se enlazan los bloques. Cada bloque contiene un hash criptográfico único que incluye la información del bloque anterior, creando así un enlace seguro. Alterar cualquier información en un bloque anterior requeriría recalcular todos los hashes de los bloques subsiguientes, una tarea computacionalmente inviable en una red distribuida grande [2].

Esta trazabilidad es particularmente valiosa en aplicaciones como la cadena de suministro y la industria farmacéutica. En la cadena de suministro, por ejemplo, la Blockchain permite rastrear la historia y el movimiento de un producto desde su origen hasta el consumidor final. Esto es crucial para verificar la autenticidad de los productos, gestionar eficientemente las cadenas de suministro y aumentar la confianza del consumidor. En el sector farmacéutico, garantiza el origen de los medicamentos, permitiendo rastrear su producción, distribución y administración, lo cual es fundamental para la seguridad del paciente y la eficacia del medicamento.

2.1.6- Aplicaciones además de las Criptomonedas

Como hemos mencionado previamente, la aplicación de la tecnología Blockchain se ha expandido a numerosos campos. Por ejemplo, en la cadena de suministro, puede proporcionar un registro transparente y a prueba de manipulaciones desde el fabricante hasta el consumidor final. En el sector de la salud, se utiliza para mantener registros médicos seguros y portables. También está encontrando uso en los sectores de bienes raíces, votaciones electrónicas, derechos de autor y propiedad intelectual, entre otros.

2.1.7- Contratos Inteligentes

Un avance notable en la tecnología Blockchain son los contratos inteligentes. Estos son programas autoejecutables que residen en el código de la Blockchain y se activan cuando se cumplen condiciones predefinidas. Los contratos inteligentes eliminan la necesidad de intermediarios, lo que reduce los costos y aumenta la eficiencia de las transacciones.

Un contrato inteligente es un tipo de código autoejecutable que opera dentro de la infraestructura de una plataforma de Blockchain. Estos contratos se desencadenan automáticamente al cumplirse condiciones específicas que han sido acordadas y codificadas previamente en su programación. Residen en una dirección específica en la Blockchain y son inmutables una vez desplegados, lo que significa que no se pueden cambiar.

Cada vez que se inicia un contrato inteligente, todos los nodos de la red de Blockchain lo ejecutan, utilizando los datos de la transacción como entrada. Esto garantiza que se realice la acción programada, como transferir fondos o registrar la propiedad, solo si la transacción cumple con las reglas del contrato. La ejecución y los resultados de los contratos inteligentes son verificados por la red, proporcionando seguridad y confianza sin la necesidad de terceros.

2.1.8- Desafíos y Consideraciones Futuras

A pesar de sus numerosas ventajas, la tecnología Blockchain enfrenta desafíos, incluyendo escalabilidad, consumo de energía (especialmente con PoW), y cuestiones regulatorias y legales. Además, la adaptación de la tecnología en sistemas existentes y su aceptación por parte del público en general son obstáculos que aún se están superando.

En resumen, la tecnología Blockchain representa una cambio de visión en la forma en que manejamos y aseguramos datos digitales. Su potencial para transformar industrias y crear sistemas más transparentes, seguros y eficientes es inmenso, aunque aún está en una fase temprana de su desarrollo y adopción. La continua innovación y el desarrollo en este campo prometen abrir nuevas fronteras en la era digital.

2.2- Criptografía de Curva Elíptica (ECC)

La Criptografía de Curva Elíptica (ECC) es un enfoque avanzado en el campo de la criptografía que ha sido implementado en este proyecto para reforzar la seguridad y la eficiencia del sistema de Blockchain diseñado para la industria farmacéutica. La elección de ECC se basa en sus características únicas que la hacen especialmente adecuada para entornos donde la seguridad y la eficiencia son de suma importancia.

2.2.1- Fundamentos y Ventajas de ECC

La Criptografía de Curva Elíptica (ECC) ofrece un enfoque más eficiente y seguro para la criptografía de clave pública. En comparación con sistemas de criptografía más antiguos como RSA, ECC logra el mismo nivel de seguridad con claves de tamaño mucho menor. Esto se debe a la complejidad matemática de resolver el problema del logaritmo discreto en el contexto de las curvas elípticas.

La curva SECP256k1, que es una especificación de la ECC utilizada por el Algoritmo de Firma Digital de Curva Elíptica (ECDSA), se ha elegido en este proyecto por su robustez y eficiencia. La curva define el conjunto de puntos que satisfacen la ecuación $\mathbf{y^2 = x^3 + ax + b}$, junto con un grupo de operaciones definidas sobre estos puntos. En el caso de SECP256k1, la ecuación se simplifica a $\mathbf{y^2 = x^3 + 7}$ debido a su forma particular, lo cual optimiza las operaciones de cálculo. Los fundamentos teóricos de ECC se centran en la dificultad de calcular el logaritmo discreto de un punto en la curva, lo que significa que es fácil multiplicar un punto por un número, pero

extremadamente difícil de hacer la operación inversa sin conocer ese número. Esta asimetría es la que permite la seguridad en las claves públicas y privadas dentro de ECC. En el siguiente apartado explicaremos los procesos de generación y verificación de firmas con el algoritmo ECDSA.

En la práctica, la clave privada en ECC es un número secreto, mientras que la clave pública es un punto en la curva que resulta de multiplicar el punto base de la curva (un punto conocido y fijo para todos los que usan la misma curva) por la clave privada. La seguridad de la clave privada se basa en la dificultad de la tarea inversa, es decir, determinar la clave privada a partir de la clave pública, conocida como el Problema del Logaritmo Discreto de Curva Elíptica (ECDLP) [7].

El Problema del Logaritmo Discreto de Curva Elíptica (ECDLP) es un problema matemático fundamental en el campo de la criptografía de curva elíptica (ECC). Se basa en la dificultad de calcular, dada una curva elíptica sobre un campo finito, el multiplicador **k** que, aplicado a un punto dado **G** de la curva (el punto base), produce otro punto **R** (la clave pública).

Matemáticamente, si conocemos **G** y **R**, encontrar k tal que **k * G = R** es computacionalmente difícil. Esta dificultad es la que sustenta la seguridad de los sistemas de criptografía basados en ECC, ya que garantiza que, aunque se conozca la clave pública y el punto base, descubrir la clave privada correspondiente (el multiplicador **k**) no es factible con los medios computacionales actuales. Esto contrasta con la relativa facilidad de realizar la operación directa: dado un número **k** y un punto **G**, calcular **k * G** es computacionalmente sencillo y eficiente en ECC.

El uso de SECP256k1 en Blockchain mejora la eficiencia al reducir el tamaño de las claves y, por ende, el espacio necesario para almacenarlas y el tiempo requerido para procesar las firmas digitales [8]. La adopción de SECP256k1 por Bitcoin ha contribuido a su popularidad y a la confianza en su seguridad, a pesar de la creciente capacidad de cómputo disponible para potenciales atacantes. La elección de esta curva en el proyecto brinda una excelente seguridad sin sacrificar la velocidad, permitiendo que el sistema de Blockchain funcione de manera eficiente incluso cuando gestiona datos sensibles y valiosos en la industria farmacéutica.

2.2.1- Generación y verificación de firmas electrónicas usando ECDSA

En este apartado explicaremos los procesos de generación y verificación de firmas con el algoritmo ECDSA. Primero, hablaremos de la generación de estas claves.

Para la generación tenemos como entradas al algoritmo un mensaje M de bits, una clave privada **d** en el intervalo [1, n−1] y con dominio de parámetros **D** y una función hash aprobada con longitud de salida de bits y fuerza de diseño de seguridad mayor o igual que la del par de claves. Como salida obtendremos un par de enteros (r, s) cada uno en el intervalo [1, n-1].

El proceso de generación de la firma es de la siguiente manera:

1. Calcular H = Hash(M) utilizando la función hash establecida o la función de salida extensible (XOF), donde la cadena de bits H tiene hashlen bits.

2. Derivar el entero e a partir de H de la siguiente manera:

a. Si len(n) ≥ hashlen, entonces se establece E = H. De lo contrario, se establece E igual a los bits más a la izquierda de H, tomando los primeros ⌈log2(n)⌉ bits.
b. Convertir la cadena de bits E al entero e.

3. Generar un número secreto por mensaje k, tal que $0 < k < n$, para los parámetros de dominio D siguiendo uno de los procedimientos en la Sección 6.3.

4. Calcular k^{-1} mod n utilizando la rutina en el Apéndice B.1.

5. Calcular el punto de la curva elíptica R = [k]G.

6. Establecer x_R como la coordenada x de la representación afín del punto $R = (x^R, y^R)$.

7. Convertir el elemento de campo x_R al entero r, utilizando la rutina de conversión en NIST SP 800-186, Apéndice F.1.

8. Establecer $r = r_1 \bmod n$.

9. Calcular $s = k^{-1} \cdot (e + r \cdot d) \bmod n$.

10. Destruir de forma segura k y k^{-1}.

11. Si r = 0 o si s = 0, y k fue generado de manera determinista, entonces indicar fallo. De lo contrario, si r = 0 o si s = 0, volver al Paso 3.

Por otra parte, para la verificación necesitamos como entradas un mensaje M, un par de enteros (r,s) y clave de verificación de firma supuesta Q y parámetros de dominio D [9].

El proceso es de la siguiente manera:

1. Verificar que tanto r como s sean enteros en el intervalo [1, n − 1]. Emitir "rechazar" si la verificación falla.
2. Calcular H = Hash(M) utilizando la función hash establecida o XOF, donde la cadena de bits H tiene hashlen bits.
3. Derivar el entero e a partir de H de la siguiente manera:
   a. Si $\lceil \log_2(n) \rceil \geq$ hashlen, establecer E = H. De lo contrario, establecer E igual a los bits más a la izquierda $\lceil \log_2(n) \rceil$ de H.
   b. Convertir la cadena de bits E al entero e como se especifica en el Apéndice B.2.1.
4. Calcular s^{-1} mod n utilizando la rutina en el Apéndice B.1.
5. Calcular $u = e \cdot s^{-1} \bmod n$ y $v = r \cdot s^{-1} \bmod n$.
6. Calcular $R_1 = [u]G + [v]Q$. Emitir "rechazar" si R_1 es el elemento de identidad (el punto en el infinito).
7. Establecer x_R como la coordenada x de la representación afín de R1 = (xR, yR).
8. Convertir el elemento de campo x_R al entero r_1, utilizando la rutina de conversión en SP 800-186, Apéndice F.1.
9. Verificar que $r = r_1 \bmod n$. Emitir "rechazar" si la verificación falla; emitir "aceptar" de lo contrario.

2.3- La Importancia de la Tecnología Blockchain en el sector farmacéutico

La tecnología Blockchain se está estableciendo como una solución revolucionaria en la industria farmacéutica, un sector de vital importancia para la salud global. Esta industria, que enfrenta desafíos significativos en seguridad, trazabilidad y eficiencia en su cadena de suministro, encuentra en la Blockchain un aliado potencial para abordar estos retos de manera efectiva.

2.3.1- Trazabilidad y Autenticidad en la Lucha Contra la Falsificación de Medicamentos

La falsificación de medicamentos representa un desafío crítico en la industria farmacéutica. La Organización Mundial de la Salud ha señalado que una proporción considerable de medicamentos en países en desarrollo son falsificados, lo que conlleva graves riesgos para la salud de los pacientes y pérdidas económicas significativas. La tecnología Blockchain ofrece una solución innovadora a este problema. Al registrar cada medicamento en la cadena de bloques desde su producción hasta su entrega, se establece un sistema de trazabilidad que permite rastrear el origen y la autenticidad de cada producto. Esta capacidad de rastreo no solo reduce el riesgo de falsificación, sino que también mejora la confianza de los consumidores en la calidad y seguridad de los medicamentos.

2.3.2- Optimización de la Cadena de Suministro Farmacéutica

La cadena de suministro en la industria farmacéutica es notoriamente compleja, involucrando a múltiples partes interesadas, como fabricantes, distribuidores, farmacias y pacientes. La tecnología Blockchain puede introducir una transparencia y eficiencia sin precedentes en este sistema. Al utilizar un registro único y consensuado en la cadena, las discrepancias y malentendidos entre las partes interesadas se minimizan significativamente. Esto no solo agiliza el proceso de suministro, sino que también mejora la gestión del inventario y reduce los costos operativos. La capacidad de seguir un medicamento a lo largo de toda la cadena de suministro facilita la identificación de ineficiencias y puntos críticos, permitiendo así una respuesta más rápida y precisa a los problemas de suministro.

2.3.3- Cumplimiento Normativo y Seguridad de Datos Mejorada

La industria farmacéutica está sujeta a estrictas regulaciones en cuanto a la gestión y el reporte de datos. La tecnología Blockchain, al proporcionar un registro detallado e inmutable de todas las transacciones, facilita enormemente el cumplimiento de estas normativas. La seguridad de los datos es otro aspecto crítico en el que la Blockchain brilla, gracias a su estructura criptográfica y su naturaleza descentralizada. Los datos en la tecnología Blockchain están protegidos contra modificaciones no autorizadas y violaciones, lo que es crucial en un sector donde la confidencialidad y la integridad de los datos son primordiales.

3- Diseño del sistema

En este apartado hablaremos sobre cómo se ha diseñado el sistema experimental propuesto en este trabajo centrándonos en una visión más general del sistema, donde hablaremos de la infraestructura de la cadena, la gestión de datos y almacenamiento, la capa de aplicación y negocio, seguridad, gestión de permisos y finalmente la interfaz. Por otra parte, en el siguiente subapartado nos centraremos en mostrar las herramientas utilizadas para esto y sus ventajas correspondientes.

3.1- Arquitectura general del sistema

La arquitectura de este sistema se ha diseñado cuidadosamente para abordar los retos específicos del sector, garantizando al mismo tiempo la seguridad, eficiencia y transparencia. La descripción detallada de cada componente clave de la arquitectura proporciona una visión clara de cómo cada elemento contribuye al funcionamiento y la eficacia del sistema en su conjunto.

3.1.1- Capa de Infraestructura Blockchain

El bloque de la cadena de bloques está diseñado como un registro digital que almacena información para la integridad y continuidad de la cadena. Incluye un identificador único "**id"** que lo distingue dentro de la cadena, y detalla una operación en el campo **"operación"**. Además, contiene un campo **"permiso"** en el que definimos los permisos que representan la autoridad del usuario utilizando la aplicación, definiendo así que acciones puede realizar o no.

Un sello de tiempo **"timestamp"** marca el momento exacto en que el bloque fue añadido, y un hash del bloque anterior **"hash_anterior"** asegura su conexión secuencial en la cadena. Un "**nonce"** se usa durante la minería para encontrar un hash válido bajo el esquema de Prueba de Trabajo. Una firma digital es utilizada para verificar la autenticidad de la transacción. Finalmente, el hash del bloque se calcula con toda esta información, creando una huella digital única que sella el bloque y verifica su contenido.

Conformación de la Cadena: La cadena de bloques, como su nombre indica, es una serie de bloques enlazados secuencialmente. Como hemos comentado previamente, cada bloque contiene un hash criptográfico del bloque anterior, formando una cadena literal que se extiende hacia atrás hasta el bloque inicial, conocido como el bloque génesis. Esta disposición asegura que una vez que un bloque se ha añadido a la cadena, cualquier intento de alterar su contenido se hace evidente, ya que cambiaría el hash no solo del bloque en cuestión sino de todos los bloques subsiguientes, creando una discordancia detectable en toda la red.

Mecanismo de consenso: En nuestro sistema, hemos adoptado el mecanismo de Prueba de Trabajo, que requiere que los nodos realicen un trabajo computacionalmente intensivo para validar las transacciones y crear bloques nuevos. Esta labor disuade a los actores malintencionados, ya que cualquier intento de alterar la cadena requeriría una cantidad prohibitiva de recursos computacionales, manteniendo así la integridad y fiabilidad de nuestro libro contable distribuido.

3.1.2- Gestión de datos y almacenamiento

Base de Datos SQL: Además de la cadena de bloques, el sistema emplea dos bases de datos donde se almacenan los usuarios, con información sobre su papel en el sistema como son el saldo, los permisos y la firma electrónica y otras que representan lo que sería el stock de una farmacia real.

Interacción entre Blockchain y Base de Datos SQL: Se ha diseñado una interacción fluida entre la cadena de bloques y la base de datos SQL gracias a python. Esto ha sido conseguido gracias la fácil comunicación con las bases de datos gracias a la ejecución de "queries", los cuales son comandos básicos y eficaces de SQL. Esto permite aprovechar las ventajas de ambos sistemas: la seguridad y la inmutabilidad de la Blockchain y la eficiencia y la flexibilidad de una base de datos SQL tradicional.

3.1.3- Capa de Aplicación y Lógica de Negocio

Los módulos de operaciones están compuestos por clases y métodos en Python que gestionan la lógica de negocio para una cadena de bloques en el contexto de la industria farmacéutica. Incluyen funciones para recargar dinero a usuarios, realizar compras de medicamentos, consultar y restablecer el inventario de productos farmacéuticos, y otras operaciones relacionadas con la gestión y el mantenimiento de la cadena de suministro.

Para realizar una operación, primero se verifica el permiso del usuario para llevar a cabo la acción deseada. Luego, se accede a una base de datos para actualizar la información relevante, como el saldo de un usuario o el stock de un medicamento. Por ejemplo, en una compra, el sistema verificaría si el usuario tiene fondos suficientes y los permisos necesarios, y si es así, procedería a reducir el saldo del usuario y actualizar el inventario de la farmacia. En caso de fallo o permisos insuficientes, se notificaría al usuario y la operación no se llevaría a cabo. Estas operaciones también pueden generar un código de operación que resume la acción realizada para el registro o auditoría.

En cuantos, a los contratos inteligentes, en caso de estar presentes estos elementos programables automatizan las operaciones, ejecutándose cuando se cumplen condiciones predefinidas. Estos contratos inteligentes son fundamentales para mejorar la eficiencia, reducir errores humanos y automatizar procesos clave dentro del sistema. Hablaremos más tarde de cómo los hemos implementado en nuestro trabajo.

3.1.4- Seguridad y Encriptación

La seguridad de los datos es un aspecto importante en el ámbito de este trabajo, donde la confidencialidad y la integridad de la información son primordiales. En este sistema, se ha implementado la Criptografía de Curva Elíptica (ECC) para la encriptación de los datos, tanto en la cadena como en la base de datos SQL. La elección de ECC se debe a sus múltiples ventajas en términos de seguridad y eficiencia.

3.1.5- Interfaz de Usuario

Interfaz Gráfica de Usuario (GUI): Desarrollada con las herramientas Tkinter en Python, la GUI es intuitiva y fácil de usar. Permite a los usuarios interactuar eficientemente con el sistema, realizando operaciones, consultando datos y gestionando actividades relacionadas con la cadena de suministro farmacéutica.

3.1.6- Gestión de Permisos y Autenticación

Control de Acceso: El sistema incluye un mecanismo de control de acceso para asegurar que solo los usuarios autorizados puedan acceder a información sensible y realizar operaciones. Esto se lleva a cabo en varios módulos del sistema, de manera que si en el caso de que la verificación de estos permisos haya fallado por ejemplo en los módulos de operación se verifiquen en el módulo principal y sigamos teniendo la certeza de que un usuario que no tenga de esos permisos no pueda operar.

3.2- Elección de Tecnologías y Herramientas para el desarrollo del Sistema de Blockchain en el sector Farmacéutico

La construcción de un sistema eficiente y seguro para la industria farmacéutica implica una cuidadosa selección de tecnologías y herramientas, cada una elegida por sus características únicas y su capacidad para integrarse armoniosamente en el diseño general del sistema.

3.2.1- Python para el Desarrollo del Sistema

Se eligió Python como candidato principal debido a su flexibilidad y facilidad de uso, que facilita tanto el desarrollo como el mantenimiento del código. La amplia disponibilidad de bibliotecas en Python permite integrar fácilmente diversas funcionalidades, desde la interacción con bases de datos hasta la implementación de algoritmos de cifrado.

3.2.2- SQL para la Gestión de Datos

Se utiliza SQL para el manejo eficiente de grandes volúmenes de datos, proporcionando un acceso rápido y eficiente, fundamental en el dinámico entorno de la industria farmacéutica. Además, la compatibilidad y escalabilidad de las bases de datos SQL facilitan la expansión y adaptación futura del sistema.

Criptografía de Curva Elíptica (ECC) para la Encriptación

Para la encriptación de datos, se ha optado por la Criptografía de Curva Elíptica debido a su seguridad avanzada con menor carga computacional. ECC ofrece una seguridad comparable o superior a otros algoritmos con claves de menor tamaño, resultando en una eficiencia operativa mejorada, lo que es especialmente importante en un sistema donde la seguridad de los datos es primordial.

3.2.3- Tkinter para la Interfaz de Usuario

La interfaz de usuario, creada con la biblioteca Tkinter, es una elección ventajosa debido a su simplicidad y eficiencia en el desarrollo de GUIs. Como un componente estándar de Python, facilita la integración con el sistema, permitiendo una experiencia de usuario simple y coherente. Tkinter permite la creación rápida de interfaces visuales que son nativas a la plataforma, lo que reduce significativamente el tiempo y los recursos necesarios para el desarrollo y mantenimiento del software, al tiempo que proporciona las herramientas para construir aplicaciones interactivas y funcionales.

4- Desarrollo del sistema Blockchain

En esta sección, ofrecemos una mirada al interior del desarrollo modular de nuestro sistema. Comenzamos describiendo la creación de bloques individuales, siguiendo con su ensamblaje en una cadena continua, la ejecución de las operaciones, y culminando con los módulos de seguridad e interfaz. Este proceso ilustra cómo cada módulo contribuye de manera esencial al funcionamiento armónico y seguro de la cadena de bloques, desde el nivel de bloque individual hasta su integración en un sistema distribuido mayor.

4.1- Diseño e Implementación de Bloques (Bloque.py)

En este apartado hablaremos la estructura de los bloques que componen la cadena y su implementación en el módulo **Bloque.py**.

En este apartado hablaremos de que función empeñan los diferentes métodos del módulo **CadenaBlockchain.py** y cuál es el funcionamiento de estos.

Cada bloque está diseñado para almacenar datos importantes: Id, operación, hash_anterior, Timestamp, permiso, nonce, hash y firma.

La variable **id** proporciona una identificación única y que aumenta en una unidad para cada bloque generado para el seguimiento y la diferenciación de los bloques dentro de la cadena.

La variable **operacion** almacena los detalles específicos de la transacción farmacéutica, como son los identificadores de medicamentos, la farmacia donde se da la operación y la operación exacta dentro de la cadena de suministro. Esta variable está diseñada para adoptar tres formatos distintos, cada uno representando diferentes tipos de operaciones: consultas de stock y precio, compras y recargas de stock, y recargas de saldo a los usuarios. Los primeros tres caracteres indican el tipo de operación (por ejemplo, CMP para compras, RST para recargas de stock), seguidos de tres caracteres que representan el medicamento involucrado en la operación, utilizando los primeros tres caracteres de su nombre. En el caso de compras y recargas, los dígitos siguientes representan la cantidad de unidades o la cantidad de dinero involucrado en la operación. Por último, para las consultas, compras y recargas, los últimos caracteres indican la farmacia donde se realiza la operación.

La variable **permiso** nos ayuda a llevar un control de qué tipo de usuario ha realizado una operación específica, viendo así si por ejemplo el usuario ha realizado una operación que requiera de un permiso más alto al que tiene y detectar la vulnerabilidad.

El **hash_anterior** es una referencia al hash del bloque previo, creando un enlace entre bloques y garantizando la inmutabilidad ya que cualquier intento de alteración se detectaría fácilmente debido a la naturaleza concatenada de la cadena, cambiando así todos los hashes de los bloques de la cadena.

El **timestamp** actúa como un registro inmutable del momento exacto en que se añadió el bloque, lo que es útil para mantener un registro cronológico y validar la secuencia de los bloques.

La variable **nonce** se utiliza en el proceso de minería para encontrar un hash válido bajo el protocolo de Prueba de Trabajo, mientras que la **firma** valida la autenticidad de la transacción.

Finalmente, el **hash** del bloque, calculado a partir de todas estas variables, actúa como un sello digital, asegurando que cualquier cambio en los datos se refleje inmediatamente, preservando la integridad y la seguridad de toda la cadena.

Estas variables trabajan en conjunto para formar cada bloque, proporcionando una estructura detallada y segura que es fundamental para la trazabilidad y el cumplimiento en el entorno regulado de la industria farmacéutica. La implementación técnica de estos bloques en la cadena Blockchain asegura la fiabilidad y la transparencia, facilitando la interacción entre los diferentes actores del sector.

4.2- Creación y Manejo de la Cadena de Bloques (CadenaBlockchain.py)

En este apartado hablaremos de que función empeñan los diferentes métodos del módulo **CadenaBlockchain.py** y cuál es el funcionamiento de estos.

El proceso de adición de un bloque en el sistema Blockchain, comienza con la creación del bloque génesis a través del método **bloqueGenesis**, que establece el fundamento de la cadena. Este bloque, se inicializa con unos datos irrelevantes por defecto y un **hash_anterior** vacío, formando la base de la cadena.

Cuando se va a añadir un nuevo bloque, el método **agregar_bloque** toma el centro del escenario. Este método comienza verificando que el hash del último bloque registrado coincida con el hash anterior almacenado en el nuevo bloque, proporcionando la conexión entre ellos, y una vez realizada esta comprobación pasa a verificar que el hash cumple con la prueba de trabajo impuesta. Además, el método **agregar_bloque_sql** permite la inserción del bloque en una base de datos SQL, lo que sugiere una persistencia de datos más allá de la memoria volátil.

Tras esta verificación, se realiza la Prueba de Trabajo (PoW) a través del método **PoW**, donde se calcula y verifica el hash adecuado que satisface la dificultad establecida de la cadena de bloques. Un hash válido se obtiene variando el valor **nonce** del bloque hasta que el hash cumpla con el criterio de dificultad, que es tener un número específico de ceros al principio del hash.

Una vez que el bloque ha pasado la PoW, se utiliza el método **hash_PoW** para comprobar la validez de este trabajo y confirmar que el hash cumple con los requisitos. Con la PoW completada y verificada, el bloque es considerado válido y el método **agregar_bloque** lo anexa a la cadena.

Además, también encontramos el método **ultimo_bloque**, con el cual recuperamos el último bloque anexado a la cadena, este método es de gran utilidad a la hora de cargar la cadena de bloques en memoria desde la base de datos SQLite cada vez que iniciamos el sistema. Este procedimiento asegura que cada bloque no solo se conecte adecuadamente con su predecesor, manteniendo la integridad de la cadena, sino que también se haya sometido a una validación intensiva, fortaleciendo la seguridad de la cadena en su conjunto.

4.3- Gestión de Datos y Operaciones(Operaciones.py, Operaciones_datos.py)

En este apartado describiremos los módulos encargados de la implementación de operaciones en el sistema como se comunican entre sí.

En el módulo **Operaciones.py**, se implementan métodos que gestionan la lógica de negocio y revalidan los permisos de los usuarios para realizar operaciones específicas. Por ejemplo, antes de efectuar una compra, se verifica que el usuario tenga los permisos necesarios y luego se hace llamada a los métodos de **Operaciones_datos.py** que son los que se encargaran de gestionar las bases de datos ejecutando los **"queries"** previamente mencionados.

Posteriormente, **Operaciones_Datos.py** se encarga de interactuar con la base de datos para reflejar los cambios resultantes de las operaciones. Utiliza métodos como **compra_medicamento** para ajustar el inventario tras una compra y **resta_saldo** para deducir el costo de la compra del saldo del usuario. Este módulo asegura que las transacciones se reflejen correctamente en la base de datos y que la información se mantenga actualizada y precisa.

Ambos módulos trabajan de manera coordinada: **Operaciones.py** maneja la lógica de alto nivel y las reglas de negocio, mientras que **Operaciones_Datos.py** lleva a cabo las operaciones en la base de datos.

4.4- Implementación de la Base de Datos SQL

En este apartado hablaremos de la implementación de las bases de datos utilizadas para el sistema así como el uso de triggers para la cadena de bloques.

La base de datos de Blockchain almacena la información para la integridad de la cadena. Cada registro, identificado por un **id**, captura una **operacion** específica que se ha llevado a cabo, junto con el nivel de **permiso** requerido para esa operación. El **timestamp** proporciona un registro cronológico del momento en que se añadió el bloque a la cadena, mientras que el **hash_anterior** asegura la conexión con el bloque previo, manteniendo así la secuencia inalterada. El **hash** del bloque actual es una suma criptográfica derivada de los datos del bloque, que junto con el **nonce** y la **firma**, proporcionan seguridad contra alteraciones, verificando la autenticidad y la validez del bloque dentro de la cadena.

Para cumplir con la integridad requerida en una cadena de bloques, para intentar replicar como se mantiene esta en una cadena de bloques real hemos hecho uso de “triggers”, unos códigos de SQL con los cuales dotamos a la base de datos con la integridad requerida al evitar que se puedan modificar o eliminar bloques ya añadidos a la cadena.

La base de datos del inventario de la farmacia enumera los medicamentos disponibles. Cada entrada lista el **Medicamento** con su **Stock** correspondiente, indicando la cantidad que tiene la farmacia, y el **Precio** a pagar por unidad. Este sistema ayuda al seguimiento de la disponibilidad de productos y la gestión eficiente de los recursos de la farmacia.

Finalmente, la base de datos de usuarios maneja las credenciales y los permisos dentro del sistema. Contiene el **Usuario** y su **Password** para el acceso, el nivel de **Permiso** que define la autorización de cada usuario dentro del sistema, el **Sueldo** asociado al perfil del usuario, y la **clave_publica** que se utiliza para realizar transacciones seguras y verificar la identidad en transacciones de Blockchain, asegurando así una operación segura y conforme a las regulaciones.

4.5 Seguridad y Encriptación de Datos (Encrypt.py)

En este apartado describiremos el proceso por el cual se crean y añaden las firmas electrónicas correspondientes de cada bloque y operación.

En el módulo **encrypt.py** podemos encontrar funciones las criptográficas para asegurar la seguridad de la cadena de bloques y las transacciones de los usuarios. Utilizando el algoritmo de curva elíptica ECDSA, específicamente la curva SECP256k1, el método **generar_claves** produce un par de claves seguras que son la base para la encriptación y la firma digital, asegurando que todas las transacciones sean verificables y protegidas contra alteraciones.

Una vez generadas, estas claves se almacenan de forma segura utilizando el método **guardar_claves**, que las serializa y las guarda en archivos **.dat**, lo que permite su recuperación y uso en sesiones futuras del sistema a través del método **cargar_claves**. Esta estrategia de persistencia de claves es esencial para mantener la confidencialidad y la accesibilidad de las credenciales de seguridad de los usuarios.

Finalmente, el método **firmar_datos** se utiliza para firmar digitalmente los datos de las transacciones, creando una huella que verifica la integridad y la procedencia de los datos. Esta firma es verificada más adelante en el proceso a través del método **verificar_firma**, que utiliza la clave pública correspondiente para asegurar que los datos no han sido alterados desde su firma, manteniendo así la integridad del sistema Blockchain y la confianza en las transacciones registradas.

4.6- Gestión de Permisos y Roles de Usuario (Permisos.py)

En este apartado desarrollaremos sobre el módulo encargado de la gestión y atribución de permisos a los usuarios del sistema.

El módulo `**Permisos.py**` es responsable de la gestión de acceso en el sistema Blockchain. Mediante el método `**consulta_usuario**`**,** el módulo accede a la base de datos SQLite para verificar las credenciales del usuario. Si el usuario y la contraseña proporcionados son correctos, el sistema procede a recuperar el nivel de permiso y el sueldo asociados al usuario de la tabla `**Usuarios**`. Estos atributos, `**permiso**` y `**sueldo**`, son esenciales para determinar qué operaciones puede realizar el usuario dentro del sistema y establecer su compensación por el uso del sistema. Con estos datos, el sistema puede autenticar al usuario y proporcionar el nivel de acceso adecuado de acuerdo con su rol y las políticas de seguridad.

5- Implementación del Sistema

En este apartado hablaremos del proceso de desarrollo del sistema propuesto, exponiendo así qué opciones hemos barajado durante el desarrollo de este y el porqué de la elección final para cada parte del proyecto además de incluir una breve guía de uso para nuevos usuarios.

5.1- Descripción del proceso de desarrollo

En esta sección hablaremos de como fue el proceso de desarrollo de este sistema. Comenzaremos hablando sobre cómo se desarrolló la estructura de los bloques, tras esto seguiremos con el desarrollo de la cadena y finalmente hablaremos de la implementación de permisos, operaciones, interfaz y seguridad.

5.1.1- Definición del bloque

En este apartado hablaremos del proceso de creación de los bloques y como han ido variando sus campos a medida que ha ido avanzando el proyecto.

El desarrollo del bloque comenzó con la construcción de su estructura en el archivo **Bloque.py** como base para esto. Esta estructura de bloque está diseñada para encapsular de manera eficiente los elementos esenciales de la información de la Blockchain. Esta estructura ha ido cambiando conforme el proyecto ha ido avanzando para cumplir los requisitos de este.

En primer lugar, el bloque se componía tan solo de 4 elementos básicos : **Id**, **Timestamp**, **Hash_anterior** y **Hash** los cuales hemos descrito en la sección anterior y son los encargados del funcionamiento correcto de la agregación de bloques a la cadena.

Gracias a **Id** y **Timestamp** podemos mantener un control sobre el orden lineal en el que se añaden los bloques en la cadena. Esto, nos permite evitar ataques como el de doble gasto, que ocurre cuando alguien logra adquirir un producto varias veces de forma simultánea. Esto se hace mediante la manipulación de la red para aceptar dos transacciones diferentes que usan los mismos fondos.

Por otra parte, los campos **Hash_anterior** y **Hash** nos ayudan a verificar la integridad tanto de la cadena como de los bloques. Para la creación de estos “hashes” se ha decidido hacer uso de SHA-256, que independientemente de la longitud o contenido de los datos con los que se forma siempre devolverá una cadena de texto de longitud fija de 256 bits única para cada bloque.

Más tarde, se añadió el campo **Nonce** debido a la implementación del consenso **Proof Of Work** a la cadena.

Una vez creada la estructura base de los bloques se añadieron los campos de **permiso** y **operación**, gracias a los cuales se pueden integrar las funciones de operaciones y gestión en la base de datos con las de la cadena.

Finalmente, se añadió el último campo de cada bloque, **firma**, una firma electrónica integrada en cada bloque que nos ayuda a asegurar la integridad de estos en la cadena, pudiendo verificar en cualquier momento si los datos un bloque han sido modificados tras la creación de este. Para la creación de dicha firma se ha implementado una firma digital basada en Criptografía de Curva Elíptica (ECC).

5.1.2- Definición de la cadena de bloques.

En este apartado hablaremos del proceso por el cual se implementaron las diferentes funciones encargadas de la gestión y agregación de bloques a la cadena.

El diseño de la cadena de bloques en este Trabajo de Fin de Grado comenzó con la implementación de las funciones esenciales que constituyen el núcleo de un sistema de cadena de bloques en el módulo **CadenaBlockchain.py** una función de agregación de los bloques a la cadena (**agregar_bloque)**, una función que devuelva el último bloque de la cadena (**ultimo_bloque**), una función para la creación del bloque génesis (**bloqueGenesis**) y las funciones correspondientes al mecanismo de consenso (**Hash_PoW** y **PoW**).

La primera de las funciones agregadas fue la de **bloqueGenesis**, esta función nos sirve como primer paso para crear la cadena, con la que creamos un bloque inicial con datos por defecto que nos sirve como base para esta.

La siguiente función añadida fue **ultimo_bloque**, función sencilla, pero de gran importancia. Gracias a esta podemos devolver el último bloque de la cadena en cualquier momento, paso esencial para desarrollar correctamente las funciones de la cadena.

Una vez añadida esta, se agregó la función **agregar_bloque**, función con la que se construye la cadena, siendo utilizada para cada operación realizada en el sistema, a la cual más tarde le añadimos la comprobación de cumplimiento de la prueba de trabajo.

En cuanto al mecanismo de consenso, aunque inicialmente se consideró un modelo de **Proof of Authority** debido a su eficiencia y control, se decidió finalmente implementar un consenso basado en **Proof of Work**. Las funciones **PoW** y **hash_PoW** son las encargadas de esto. En ellas, se implementa un puzle computacional donde el hash de cada nuevo bloque debe comenzar con un número predefinido de ceros. Esta tarea garantiza que añadir un bloque a la cadena requiera un esfuerzo computacional significativo, lo que a su vez protege la red contra manipulaciones maliciosas y proporciona una forma de validar la autenticidad y el esfuerzo invertido en la creación de cada bloque.

5.1.3- Definición de operaciones

En este apartado hablaremos del proceso por el cual se implementaron los diferentes módulos encargados de la gestión de operaciones en el sistema.

El tratamiento de datos en las bases SQL y la lógica de las operaciones en el sistema fueron desarrollados en los módulos **Operaciones.py** y **Operaciones_Datos.py**. Inicialmente, el módulo **Operaciones.py** se dedicó a las operaciones básicas de consulta de stock, la compra y reposición, esenciales para el control de inventario en el contexto farmacéutico. Estas operaciones básicas permiten ajustar el stock de los medicamentos en función de las compras y ventas, asegurando así una gestión eficiente de los recursos.

A medida que el proyecto creció se creó el módulo **Operaciones_Datos.py**, dedicado exclusivamente a la definición de las consultas SQL necesarias para interactuar con la base de datos. La separación de la lógica de operaciones de la ejecución de consultas SQL permite una mayor claridad y mantenimiento del código. Mientras **Operaciones.py** controla la lógica detrás de las operaciones (como verificar la disponibilidad de stock o la adecuación del precio), **Operaciones_Datos.py** se encarga de la implementación directa de estas operaciones en la base de datos a través de consultas SQL.

La funcionalidad del sistema se enriquece con la inclusión de operaciones para la consulta de **stock** y **precio**, así como la definición de **permisos** de usuario para cada operación. Esta expansión de capacidades mejora significativamente la utilidad y la seguridad del sistema. Las consultas de **stock** y **precio** permiten a los usuarios obtener información en tiempo real sobre la disponibilidad y el costo de los medicamentos, mientras que la implementación de **permisos** garantiza que solo los usuarios autorizados puedan realizar operaciones críticas.

Posteriormente, se añadieron funciones adicionales como la **recarga de saldo** y la **deducción de saldo** al realizar compras. Estas funciones complementan el conjunto existente de operaciones, proporcionando un sistema más completo y coherente para la gestión de transacciones financieras. Por ejemplo, la función de recarga permite a los usuarios aumentar su saldo, facilitando así futuras compras, mientras que la deducción automática de saldo al realizar una compra asegura una gestión eficiente y precisa de los fondos.

La culminación de este desarrollo fue la creación de la variable **var_op**, un elemento para registrar las operaciones realizadas en la cadena de bloques. Esta variable está diseñada para adoptar tres formatos distintos, cada uno representando diferentes tipos de operaciones: consultas de stock y precio, compras y recargas de stock, y recargas de saldo a los usuarios.

5.1.4- Implementación de los permisos

En este apartado hablaremos del proceso de desarrollo de la implementación de los permisos, viendo el módulo donde esta está implementada y con qué base de datos.

El módulo **Permisos.py** es el encargado de la gestión de accesos y la asignación de roles en la aplicación, esto gira en torno a la función **consulta_usuario**. Esta función es esencial para el proceso de autenticación y autorización dentro del sistema. Cuando un usuario intenta acceder a la aplicación, **consulta_usuario** se encarga de verificar sus credenciales (nombre de usuario y contraseña) con los registros almacenados en la base de datos **users.db**.

La implementación de esta lógica de permisos y saldos en **Permisos.py** es de gran importancia para el funcionamiento eficiente y seguro de la aplicación. Asegura que la aplicación mantenga un control riguroso sobre quién puede acceder a qué información y qué acciones pueden realizar. Esta segregación de acceso no solo mejora la seguridad del sistema, sino que también contribuye a una mejor experiencia de usuario, ya que cada usuario recibe un entorno personalizado y adecuado a sus necesidades y autorizaciones.

5.1.5- Encriptación y firma electrónica

En este apartado hablaremos de cómo hemos implementado la encriptación y creación de la firma electrónica en cada uno de los bloques añadidos a la cadena, haciendo así un resumen de las funciones utilizadas para ello.

La primera función del módulo no fue otra que la correspondiente de generar las claves públicas y privadas únicas de cada usuario, **generar_claves**. La clave privada es utilizada para firmar datos, mientras que la clave pública se utiliza para verificar la firma. Estas claves se generan utilizando la biblioteca ECDSA, específicamente con la curva SECP256k1, ampliamente reconocida por su seguridad y eficiencia [referencia].

Una vez generadas se añaden de la siguiente manera en el ecosistema de la aplicación:

- **Almacenamiento de la Clave Pública:** La clave pública se almacena en la base de datos SQLite. Esta base de datos actúa como un registro centralizado de todos los usuarios y sus claves públicas, facilitando la verificación de firmas por parte de otros usuarios o del sistema.

- **Almacenamiento Seguro de la Clave Privada:** La clave privada se guarda localmente en el ordenador del usuario en un archivo ".dat" encriptado. La función **guardar_claves** se encarga de esta operación, asegurando que solo el usuario correspondiente tenga acceso a su clave privada. Esta función va de la mano de **cargar_claves** , con la que podemos utilizar estas claves guardadas en cualquier momento.

Finalmente, las funciones **firmar_datos** y **verificar_firma** las encargadas proceso de encriptación y verificación de datos. La firma de datos se realiza utilizando la clave privada del usuario, lo que garantiza que la información proviene de una fuente auténtica y no ha sido alterada. La verificación de la firma, por otro lado, se realiza utilizando la clave pública correspondiente, permitiendo a cualquier parte con acceso a esta clave pública verificar la autenticidad de los datos firmados.

5.1.6- Interfaz

El desarrollo final implicó la creación de una interfaz de usuario, una etapa para hacer que el sistema Blockchain y la base de datos SQL fueran accesibles y manejables para los usuarios. Aunque inicialmente se consideró el uso del framework Flask para desarrollar una interfaz web que funcionara localmente, esta idea fue finalmente descartada debido a que gracias a que Tkinter es una interfaz instalada por defecto en Python, para el usuario sería mucho más accesible al no precisar de la instalación de Flask para el uso de la aplicación.

La interfaz de Tkinter proporcionó un medio eficiente para recoger los datos introducidos por el usuario, como nombres de usuario, contraseñas y detalles de operaciones específicas como son la cantidad de un medicamento con el que vamos a realizar una operación, el nombre de este y la farmacia en la que operaremos. Estos datos se utilizaban luego para acceder y activar la lógica definida en los módulos de operaciones y otros componentes del sistema. Por ejemplo, una vez que un usuario se autenticaba correctamente a través de la interfaz, podía realizar operaciones que se reflejaban tanto en la base de datos SQL como en la Blockchain, asegurando una gestión coherente y segura de la información

5.2- Funcionamiento de la aplicación

En este apartado nos centraremos en mostrar una breve guía sobre el despliegue y funcionamiento de la aplicación creada. A continuación, mostraremos una captura de la interfaz de la misma.

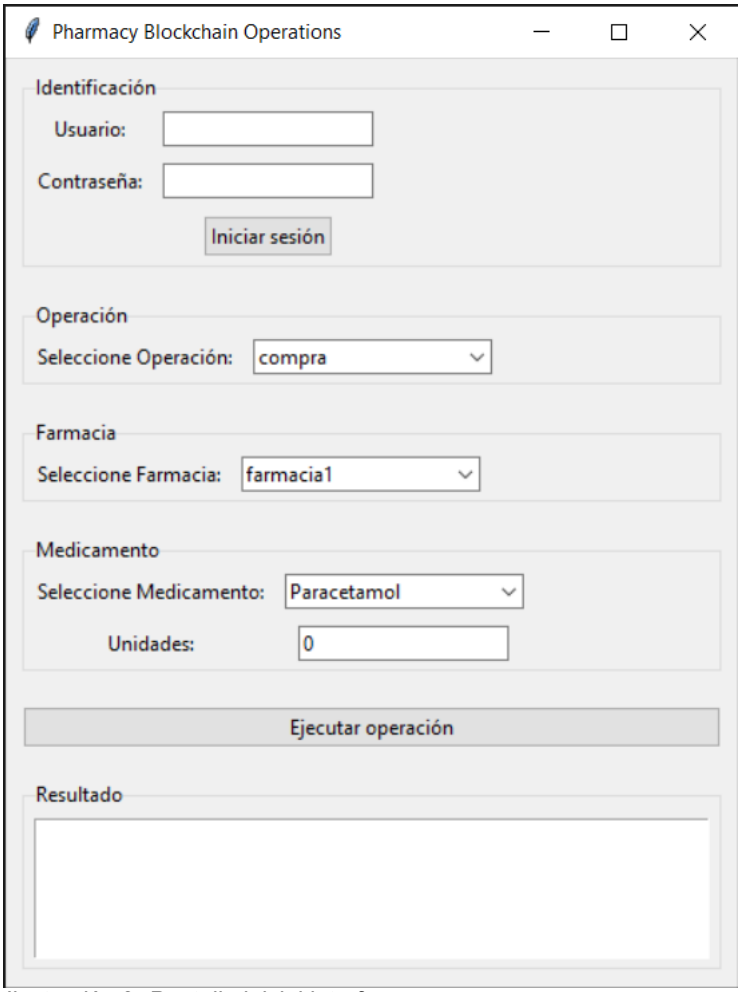


Ilustración 2- Pantalla inicial interfaz

Como podemos ver podemos identificar 5 secciones principalmente. La sección de identificación, donde introduciremos nuestros credenciales e iniciaremos sesión para que el sistema nos atribuya con los permisos que correspondan. La sección operación, con la que elegiremos qué operación realizar, la sección farmacia donde elegiremos la farmacia donde realizaremos la operación, una sección medicamento para elegir este y la cantidad con la que operar. Y finalmente, un botón de ejecutar operación junto al recuadro de texto resultado que nos mostrará información sobre las operaciones que hagamos.

En primer lugar, como hemos mencionado ya, tendremos que identificarnos con nuestros credenciales para obtener unos permisos y un sueldo acorde con nuestra identidad mostrándose por pantalla, en caso de no ser un usuario que pertenezca al sistema automáticamente se nos asignará un permiso “bajo” y un saldo de 0 euros.

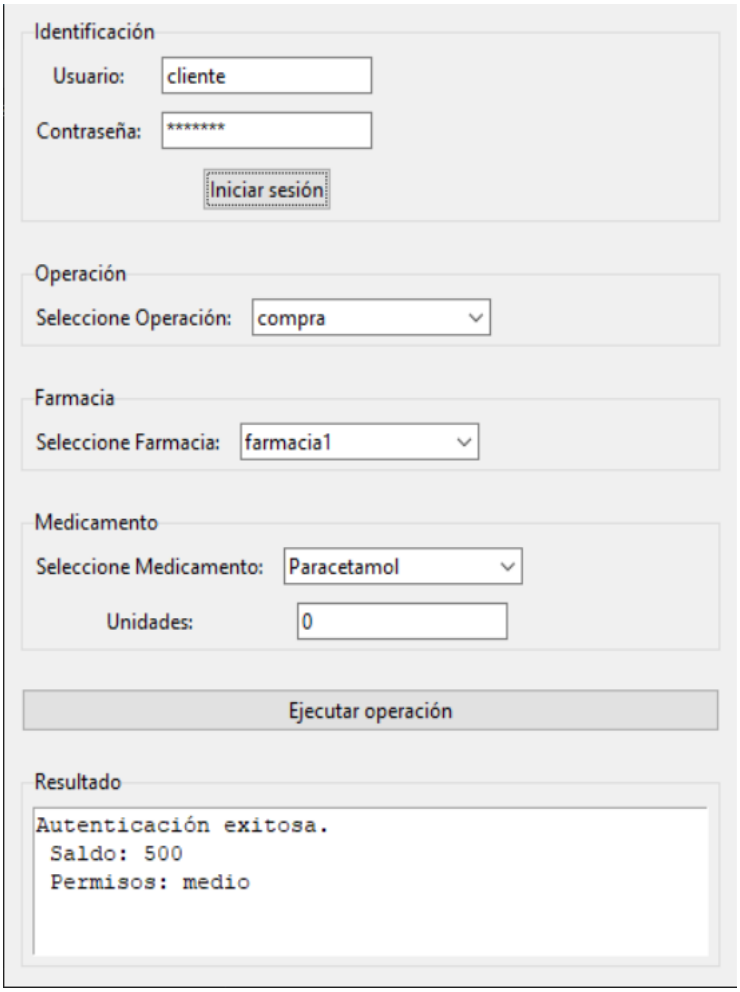


Ilustración 3- Pantalla de la interfaz una vez identificado el usuario

Tras esto, deberemos seleccionar en el menú desplegable de operación la operación que queremos realizar, la farmacia sobre la que queremos operar, el medicamento y finalmente las unidades. En caso de ser exitosa la operación, se mostrará un mensaje por pantalla acorde con la operación que hayamos llevado a cabo, en caso de ser fallida se mostrarán distintos errores por pantalla los cuales abordaremos en el siguiente apartado.

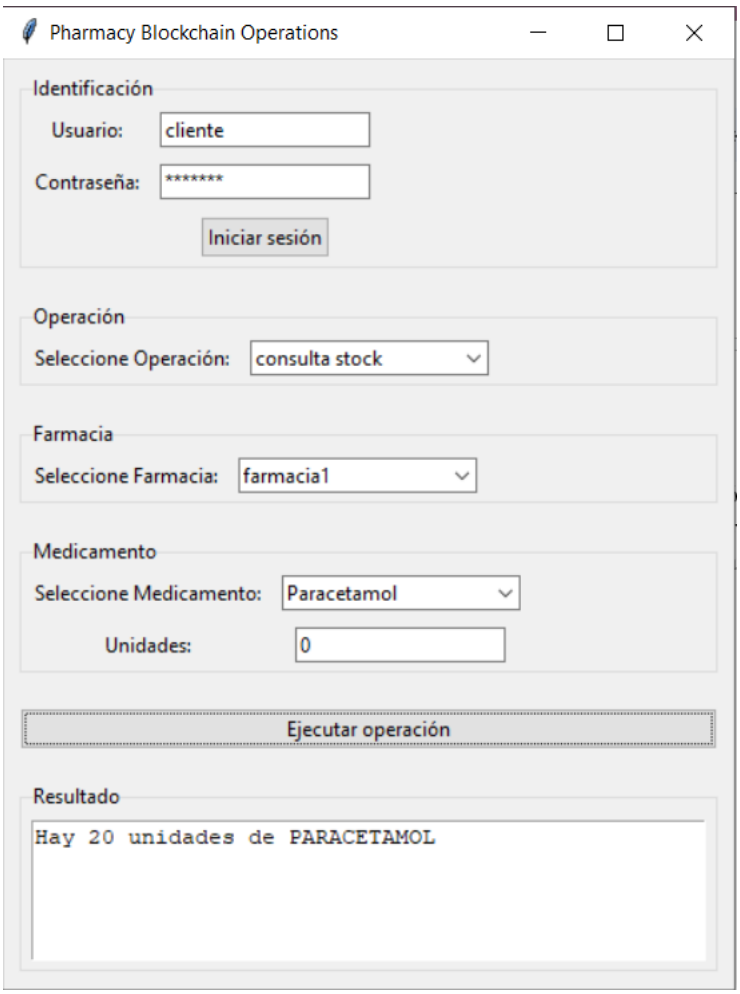


Ilustración 4- Pantalla de la interfaz una vez realizada una operación correcta

6- Evaluación del sistema

6.1- Métodos y criterios de evaluación

En esta sección, delineamos el conjunto de métodos y criterios adoptados para evaluar exhaustivamente la seguridad, rendimiento y rendimiento del sistema de Blockchain desarrollado. La evaluación meticulosa es fundamental para asegurar la integridad y eficiencia del sistema, particularmente en el ámbito de aplicación crítica de la industria farmacéutica.

Para dicha evaluación del sistema llevaremos a cabo una serie de pruebas estructuradas, cada una diseñada para explorar y validar diferentes aspectos.

Comenzaremos creando la cadena desde cero, teniendo tan solo el bloque génesis de esta:

	id	operacion	permiso	timestamp	hash_anterior	hash	nonce	firma
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
1	0	CTS	bajo	1697539779.80962	*NULL*	002fbf8e304d658611342951afcd2f2fe...	220	

Ilustración 5- Imagen del bloque génesis de la cadena de bloques

Pruebas funcionales

En estas pruebas comprobaremos la correcta funcionalidad de los elementos de nuestro sistema, comprobando que tanto las operaciones como la agregación de bloques al sistema se llevan a cabo correctamente al ejecutar el programa.

Consulta de Stock

Comenzaremos mostrando la funcionalidad la consulta de stock de un medicamento, en este caso del paracetamol, esta solicitud la haremos a “farmacia1” estando identificados como el usuario “cliente”, de permisos medios, aptos para la tarea que vamos a realizar.

El programa así nos devolverá el siguiente resultado:

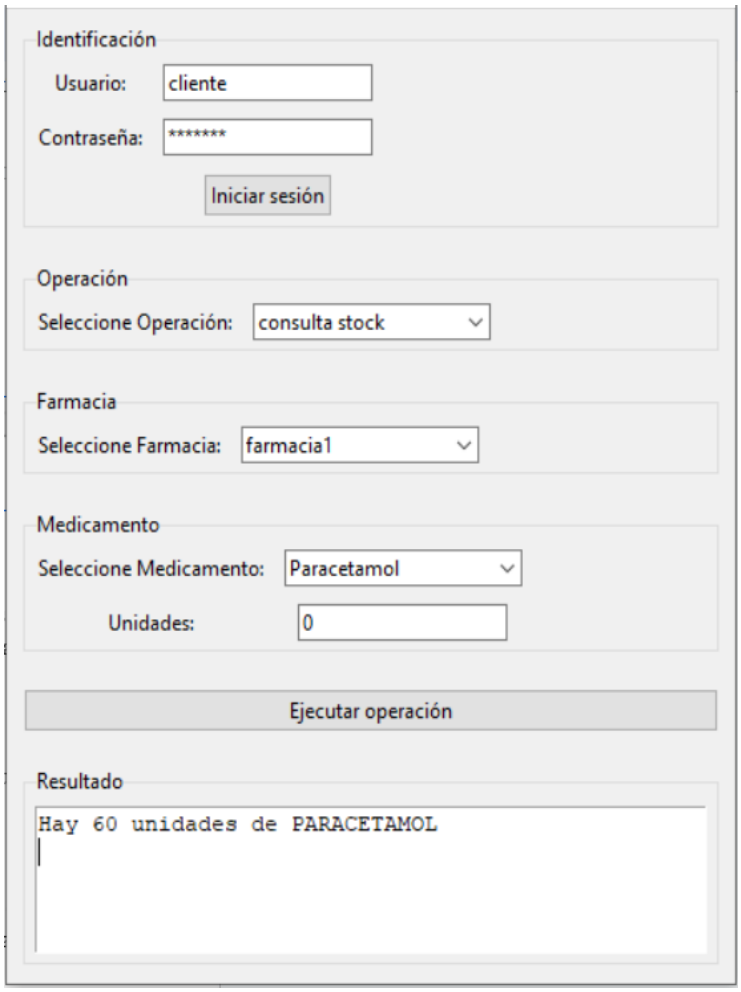


Ilustración 6- Pantalla interfaz tras realizar consulta de stock de paracetamol

Ahora, si vamos a la tabla donde se encuentra la información sobre los productos de “farmacia1”, podremos comprobar que, efectivamente, hay 60 unidades de paracetamol.

Finalmente, si miramos ahora la cadena de bloques veremos que se ha generado un nuevo bloque con código de operación correspondiente a esta consulta de medicamento en la farmacia indicada.

Consulta precio

Pasaremos ahora con la consulta de precio de este mismo medicamento, realizando la consulta a la misma farmacia e identificado igualmente como “cliente”. La salida devuelta por el programa será:

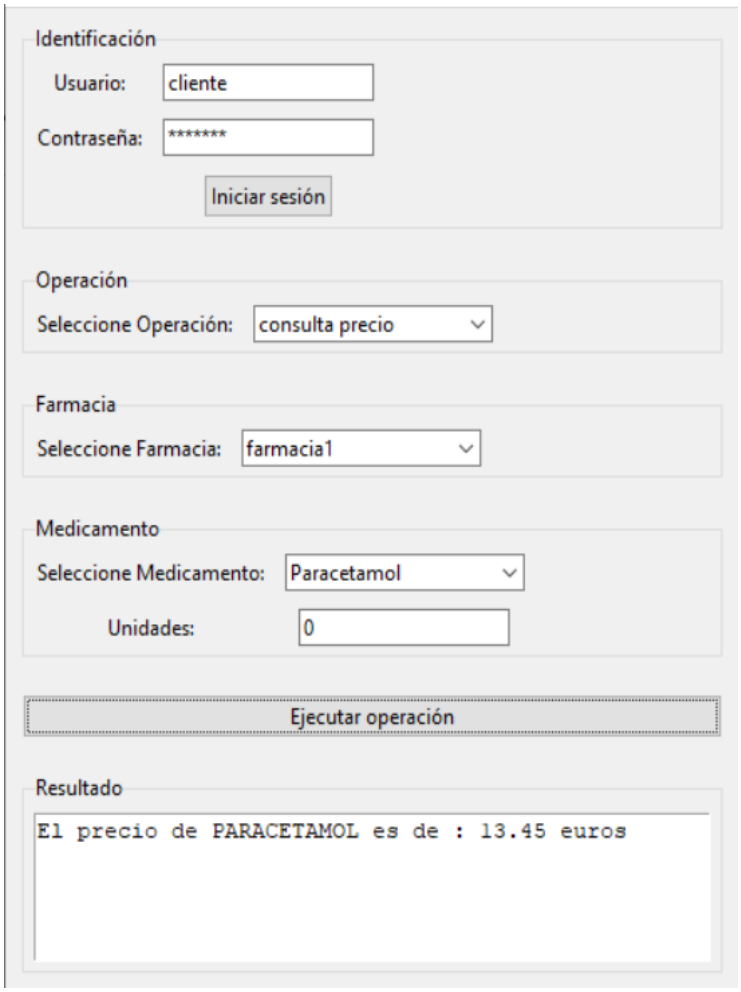


Ilustración 7- Pantalla interfaz tras realizar consulta de precio de un medicamento

Indicándonos que el precio del paracetamol es de 13.45 euros, lo cual coincide con la tabla y además veremos que se habrá añadido un nuevo bloque a nuestra cadena de bloques:

Compra

Ahora comprobaremos la compra de 4 unidades de diazepam. En este caso llevaremos a cabo la operación en la “farmacia2” y seguiremos con autentificados como “cliente”. Vemos aquí el estado actual del registro del diazepam.

Ejecutamos la operación y se nos devolverá como resultado lo siguiente:

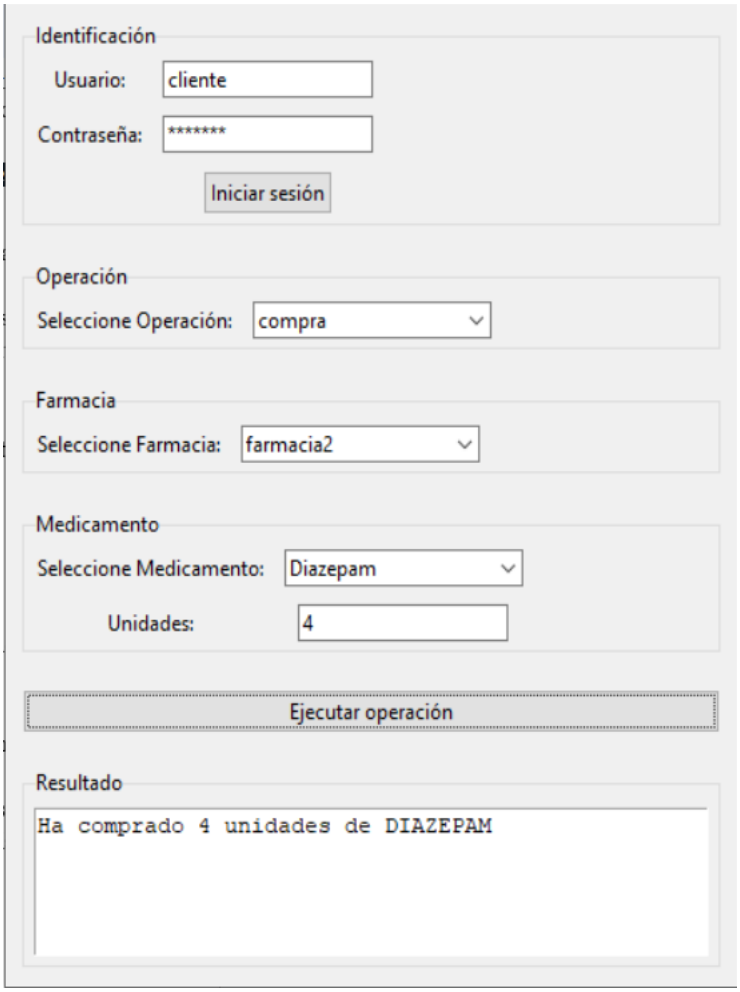


Ilustración 8- Pantalla interfaz tras realizar compra de un medicamento

A continuación, si refrescamos la base de datos de “farmacia2”, veremos que efectivamente, se ha llevado a cabo correctamente la operación.

Finalmente, veremos que se ha añadido el bloque a la cadena correctamente.

Una vez demostrado el correcto funcionamiento de la compra veremos cómo, si intentamos realizar esta sin cumplir las condiciones correctas de permiso, saldo disponible o stock, obtendremos un error.

Primero, si el saldo de nuestro usuario es inferior al requerido para realizar la operación. Por ejemplo, intentando comprar 35 unidades de Amoxicilina obtendremos el siguiente mensaje de error y por consecuente ni se crearía un bloque ni se realizaría la operación.

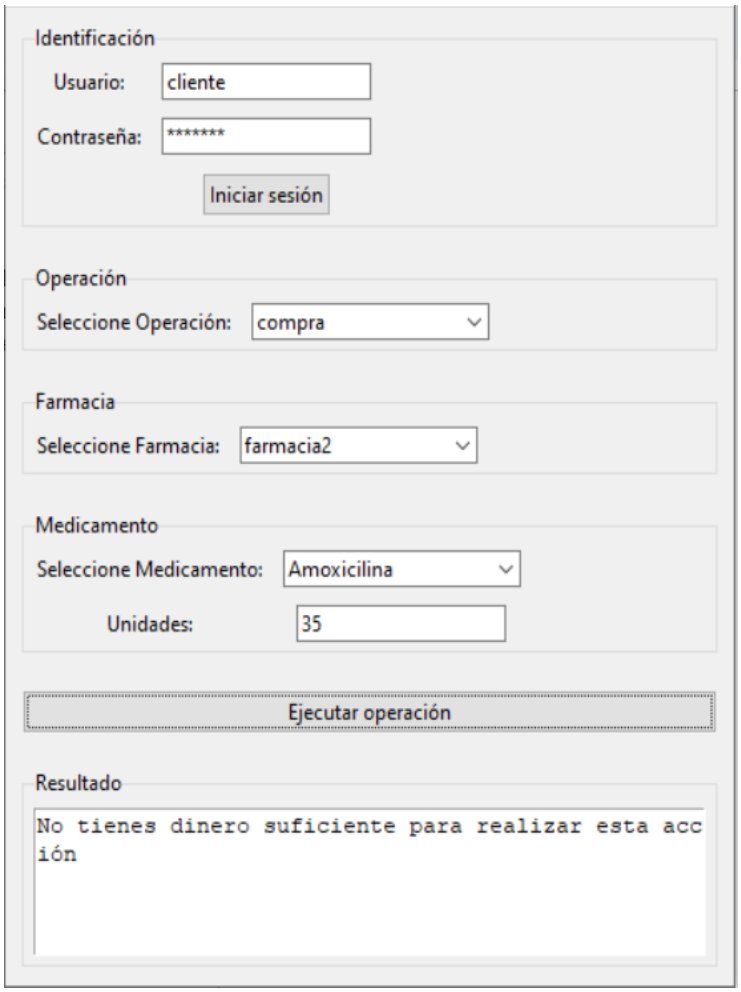


Ilustración 9- Pantalla error saldo insuficiente en operación de compra

Ahora, intentaremos comprar una cantidad de un medicamento de la cual sí tengamos los fondos necesarios, pero no haya el stock suficiente, por ejemplo intentaremos comprar 25 cajas de diazepam, cantidad que sobrepasa el stock actual de 22. Obtendremos el siguiente error:

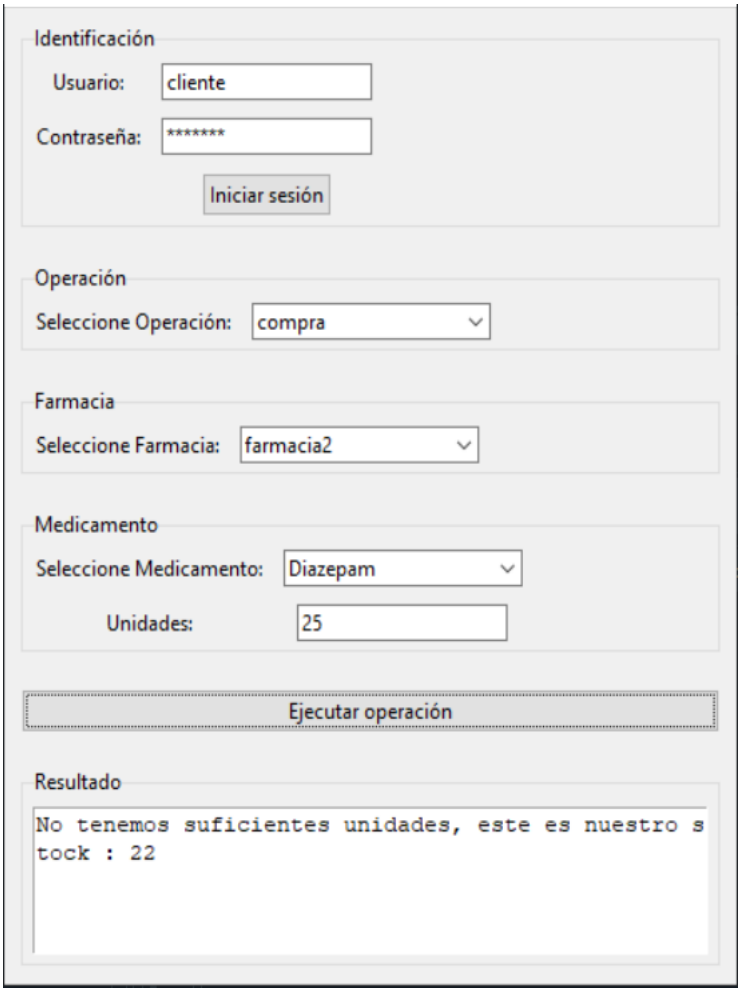


Ilustración 10- Pantalla error falta stock en operación de compra

Restock

Comprobaremos la reposición de 10 unidades de ibuprofeno. En este caso llevaremos a cabo la operación en la "farmacia3", ahora tendremos que cambiar al usuario administrador para poder realizar correctamente la reposición, sino no podremos llevar a cabo la operación correctamente como veremos más adelante. Al realizarlo se mostrará lo siguiente por pantalla:

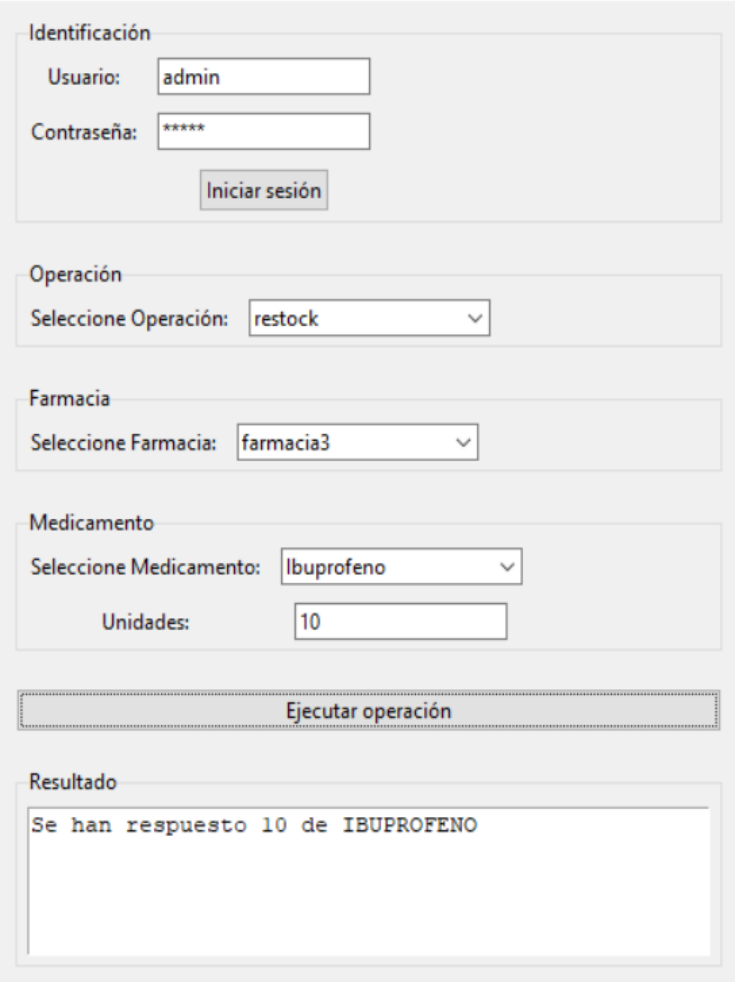


Ilustración 11- Pantalla interfaz tras realizar stock de medicamento

Y podremos ver reflejado otra vez, que se ha creado un bloque nuevo en la cadena. Además, habremos visto incrementada

Recarga

Por último, nos encargaremos de realizar las pruebas de la recarga de saldo a un usuario, en este caso volveremos con el usuario cliente, que tiene 500 euros de saldo y le haremos una recarga de 10 euros. Se mostrará lo siguiente por pantalla:

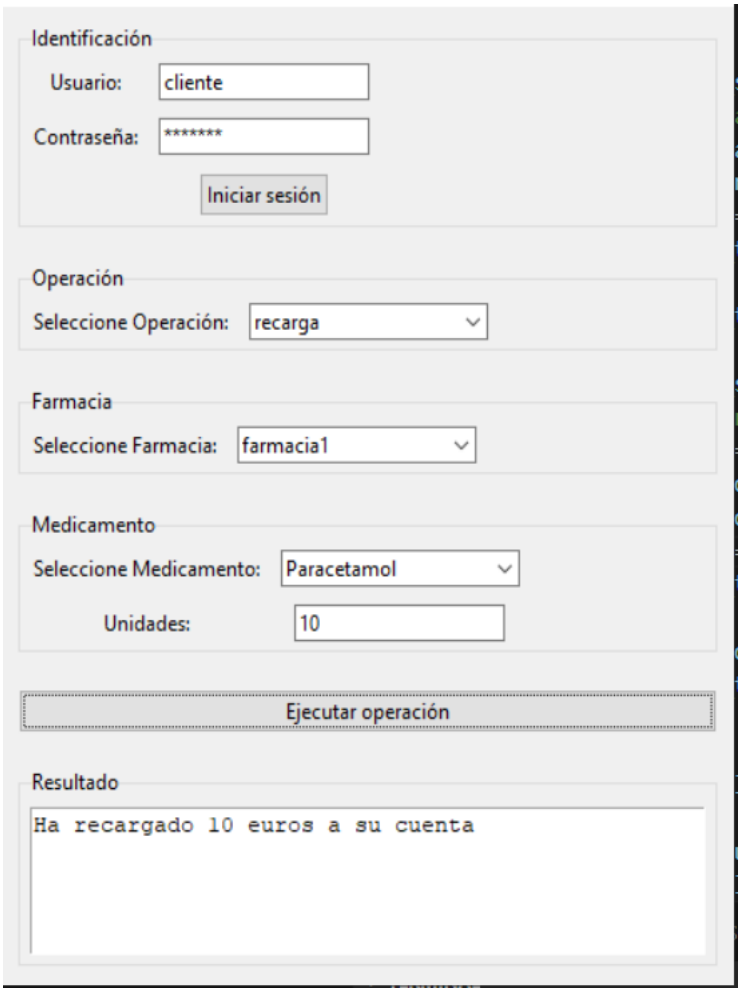


Ilustración 12- Pantalla interfaz tras operación de recarga de saldo de el usuario cliente

Y, si comprobamos el saldo del usuario ahora veremos que ha incrementado en 10 euros:

Ilustración 13- Pantalla de información de usuario tras operación de saldo para usuario cliente

	id	operacion	permiso	timestamp	hash_anterior	hash	nonce	firma
	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro	Filtro
1	0	CTS	bajo	1697539779.80962	*NULL*	002fbf8e304d658611342951afcd2f2fe...	220	
2	1	CTSPARfar1	medio	1705938999.03149	002fbf8e304d6...	000006e1f94da751706919c5b96a811...	129508	*BLOB*
3	2	CTPPARfar1	medio	1705939673.41526	000006e1f94da...	000003f70bc29d1c9c7e245457519f84...	2106624	*BLOB*
4	3	CMP4DIAfar2	medio	1705940882.31334	000003f70bc29...	00000b153969c135dde99fbd4721689...	2178865	*BLOB*
5	4	RST10IBUfar3	alto	1705945953.16992	00000b153969c...	00000cbab9371684e256713753b3416...	350234	*BLOB*
6	5	RCG10cliente	medio	1705946610.53528	00000cbab9371...	000001ed1e0e5557a7b4b31de3b0c52...	1454363	*BLOB*

Tabla 1- Cadena de bloques utilizada para la evaluación del sistema (Blockchain.db)

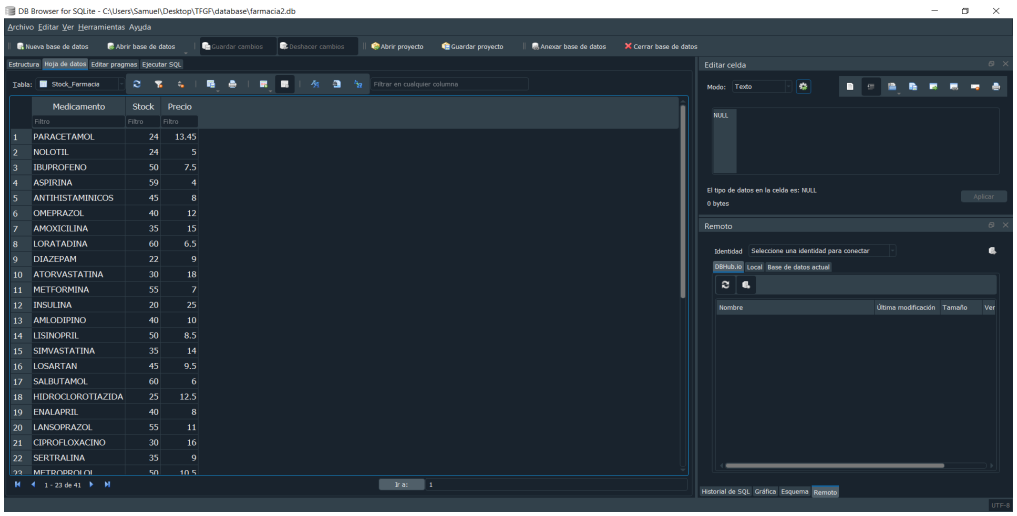


Tabla 2- Farmacia 2 en estado inicial

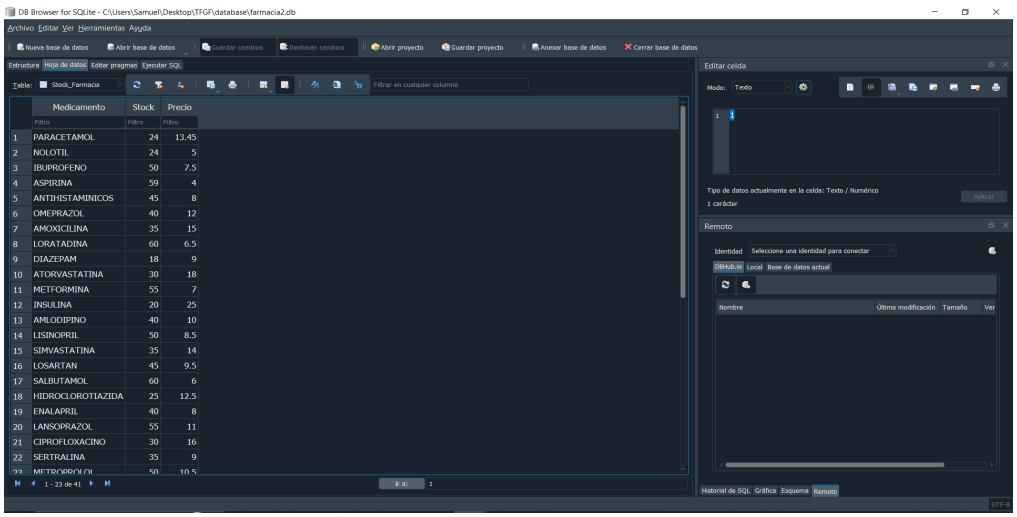


Tabla 3- Farmacia 2 tras realizar las operaciones

Pruebas de seguridad

En este apartado nos enfocaremos en probar las funciones y conceptos básicos de la seguridad de nuestro sistema. Abordaremos dos aspectos principales,el control de acceso y la validación e integridad de la Blockchain.

Control de acceso

En esta sección abordaremos varios aspectos principales del control de acceso, la atribución correcta de permisos y saldo a cada usuario tras una autenticación correcta, la verificación de permisos para cada operación, la actualización de estos si dentro de la misma sesión cambiamos de usuario y finalmente, el reestablecimiento de estos tras cerrar la sesión.

Comencemos primero viendo el control de los permisos. En el primero de los casos nos conectaremos a la aplicación como el usuario “cliente”, previamente utilizado, e intentaremos realizar una operación de permiso alto, el restock. Como resultado podremos comprobar que nos sale el siguiente mensaje de error por pantalla:

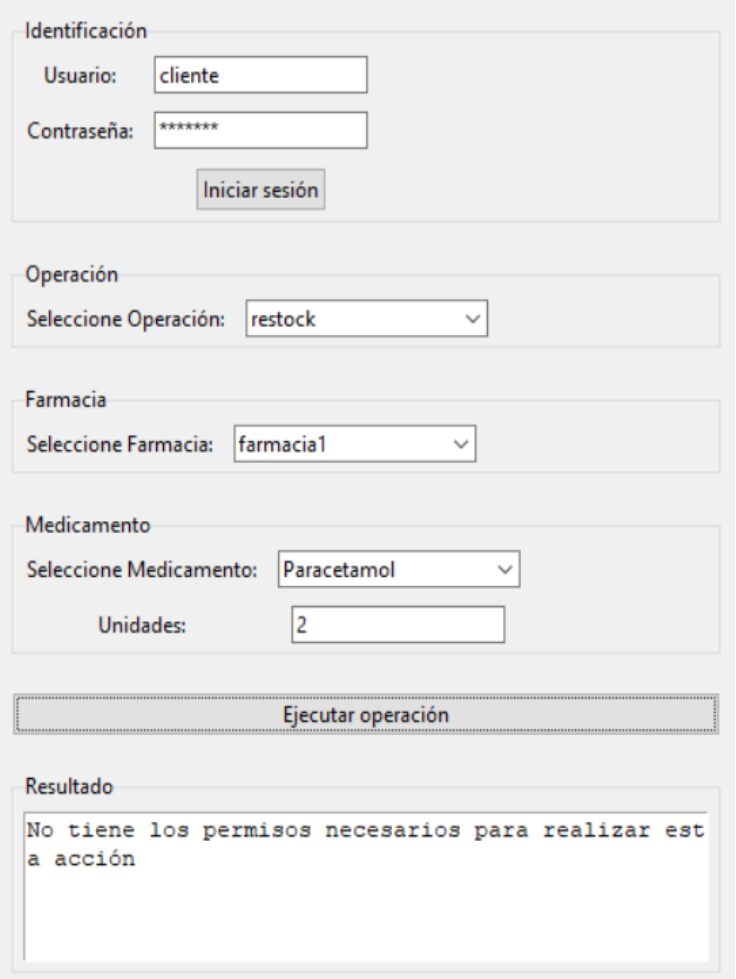


Ilustración 14- Pantalla interfaz de error al intentar realizar restock sin permisos necesarios

Esto mismo ocurrirá si, tras no habernos podido identificar correctamente como ningún usuario registrado en el sistema intentamos realizar una compra:

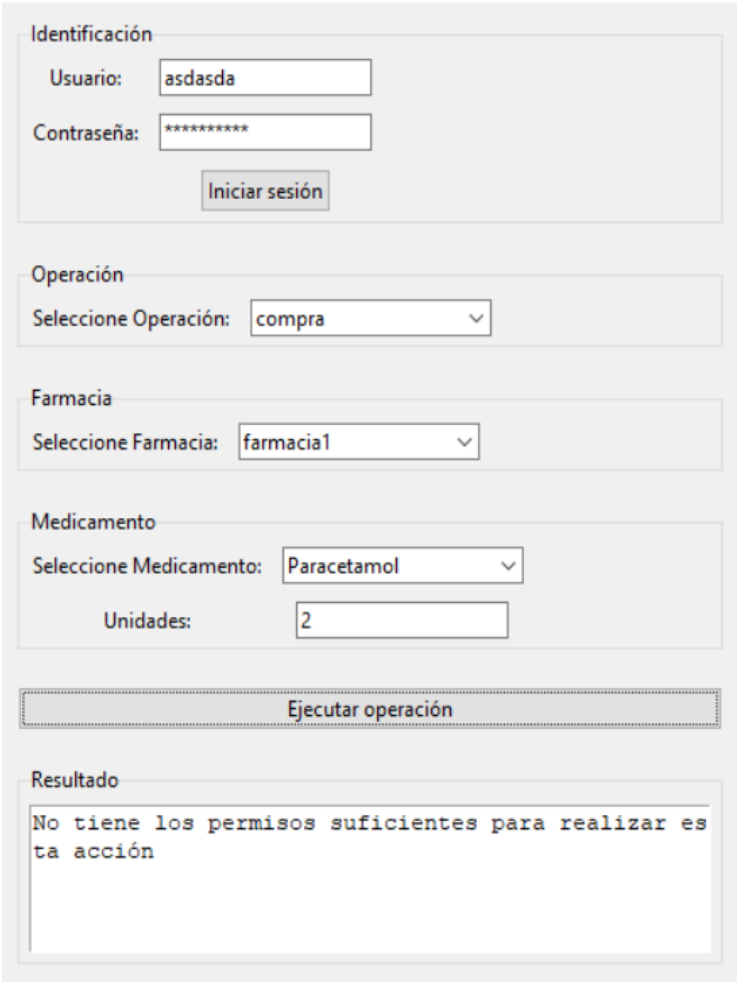


Ilustración 15- Pantalla interfaz error al intentar realizar compra sin permisos necesarios

Ahora toca cubrir un importante punto en la seguridad que podría ser fruto de un ataque malicioso si no se lleva el control correctamente, el cambio de permisos. En este caso comprobaremos que tras habernos identificado como el usuario administrador y haber realizado una operación que requiera permisos altos como es el de restock. Si posteriormente nos identificamos como otro usuario ya sea por ejemplo “cliente”, los permisos se ajustarán a esta nueva sesión y no podremos volver a realizar dicha operación.

Primero nos identificamos como administrador y realizamos el restock:

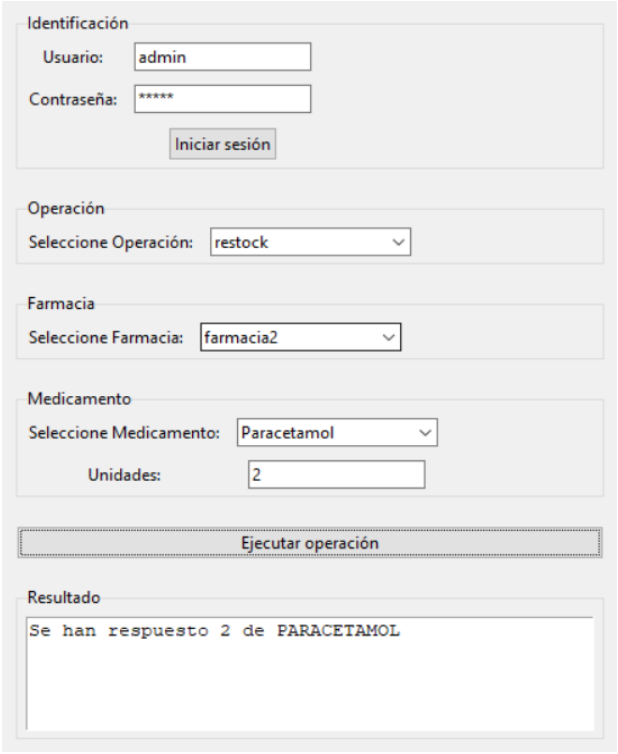
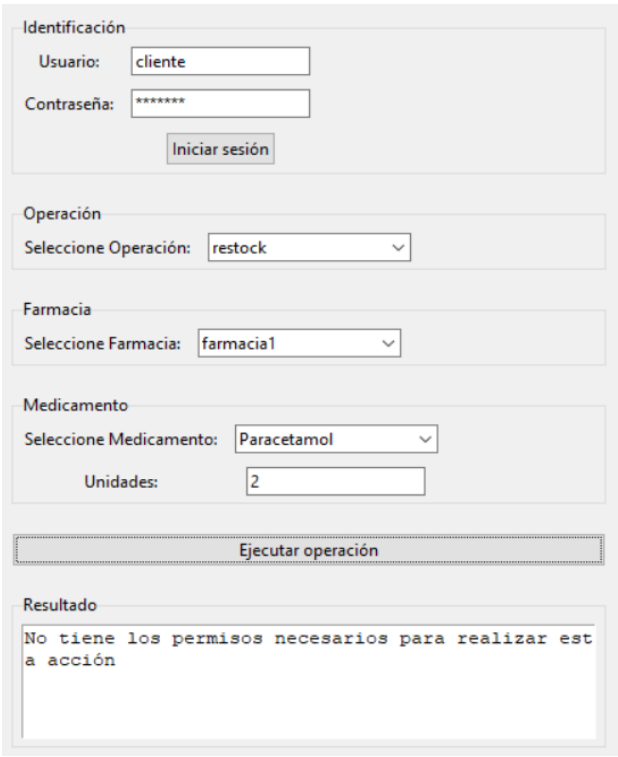


Ilustración 16- Pantalla interfaz de operación restock

Una vez hecho esto pasaremos a identificarnos como “cliente” y veremos que obtendremos el mismo error que en la ilustración 13, indicándonos que no tenemos los permisos necesarios para realizar dicha acción:



Finalmente comprobaremos que tras cerrar la aplicación efectivamente, estos permisos no se guardan y tendremos que identificarnos de nuevo para hacer cualquier acción que requiera un permiso más allá del bajo.

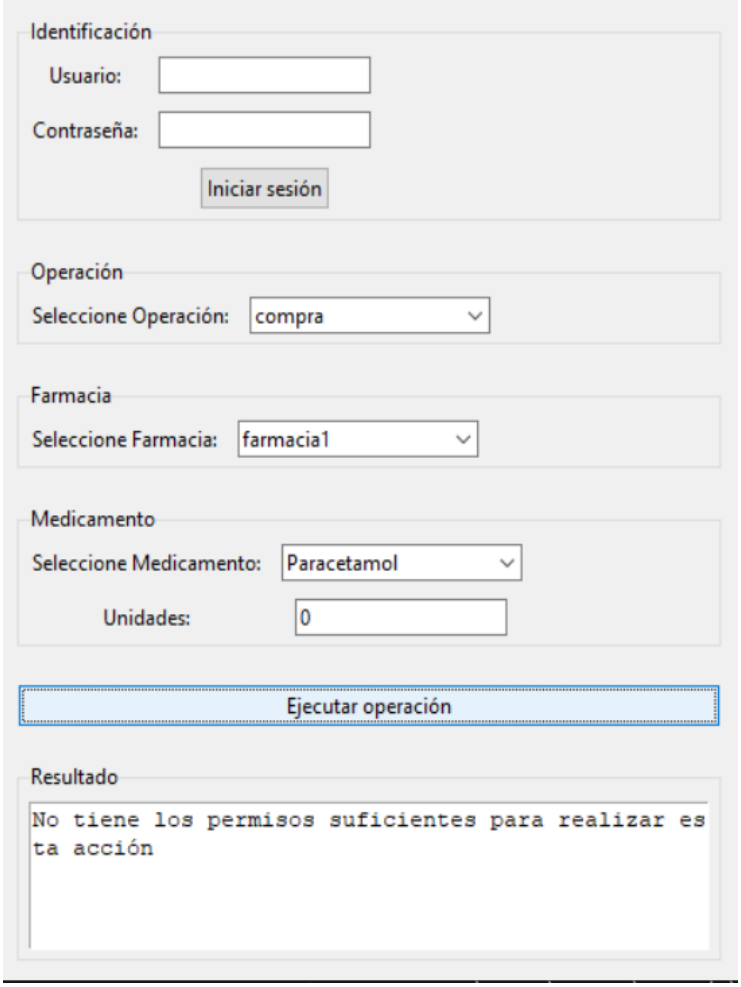


Ilustración 17-Intento de compra tras reiniciar la aplicación

Una vez comprobados los permisos y su correcta atribución pasaremos al punto que probablemente sea más importante de toda esta sección. La integridad de la Blockchain. Para cumplir esta propiedad hemos tenido que tomar una aproximación diferente a la que se suele utilizar en la mayoría de sistemas Blockchain debido a la naturaleza de la nuestra. Al guardar localmente en una base de datos esta cadena no podemos verificar su integridad comparándola con copias de esta misma que otros usuarios tienen en la nube, por lo que hemos decidido hacer uso de un “trigger” en nuestra base de datos SQLite, de manera que si intentamos borrar o actualizar cualquiera de los bloques ya integrados a nuestra Blockchain nos será imposible y recibiremos un error notificandonos de esto.

Por ejemplo en la siguiente imagen intentaremos cambiar el número de unidades de la última operación realizada.

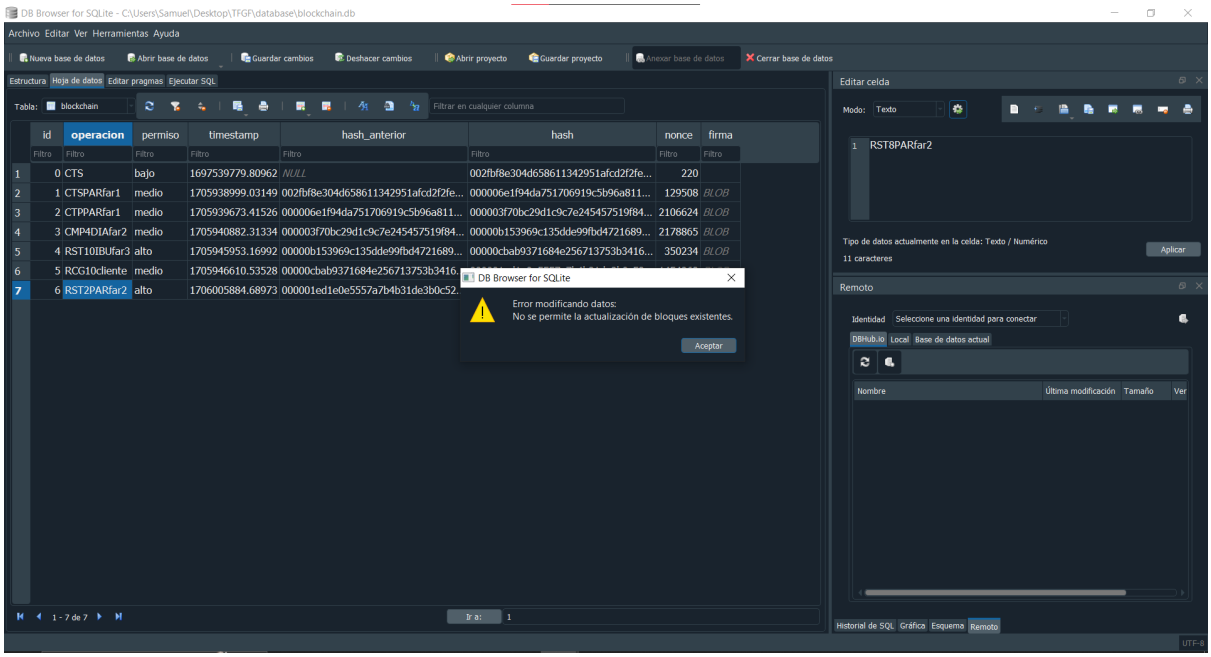


Ilustración 18-Demostración del trigger implementado en la base de datos donde está almacenada la cadena de bloques

Pruebas de rendimiento

En este apartado pondremos a prueba el sistema realizando el mayor número de operaciones en un periodo corto de tiempo y ver cómo este responde. La operación elegida será la compra de un ibuprofeno a “farmacia1” desde el usuario “cliente”.

Hemos puesto 6 como número de ceros con los que tiene que empezar el hash del nuevo bloque creado en lo que respecta al **Proof of Work**, con lo que aumentaremos significativamente la dificultad computacional para resolver el puzle para ver qué tal responde el sistema a este cambio.

Dicho esto, procedemos a realizar la mayor cantidad de operaciones posibles de una sentada, vemos que el programa tarda en responder, pero finalmente en 3:10 minutos consigue realizar en este caso las 32 operaciones solicitadas, creando y adhiriendo un bloque nuevo para cada una de estas.

10	9	CMP1PARfar1	medio	1706015117.36089	000003f6d03163cf1379529d0ecf46be...	0000021a5bd57dee646e15f01bfac18...	4317964	BLOB
11	10	CMP1PARfar1	medio	1706015133.77991	0000021a5bd57dee646e15f01bfac18...	000002d1bee38a9dd21c7ad5111a81f...	578568	BLOB
12	11	CMP1PARfar1	medio	1706015136.51511	000002d1bee38a9dd21c7ad5111a81f...	000006fcc3afe68933ad15ef05338d77...	1588787	BLOB
13	12	CMP1PARfar1	medio	1706015144.08261	000006fcc3afe68933ad15ef05338d77...	00000a28a38ef66ab73fc0d79939083...	1790709	BLOB
14	13	CMP1PARfar1	medio	1706015150.87817	00000a28a38ef66ab73fc0d79939083...	00000064639a2a6b174a3441b0f3002...	175482	BLOB
15	14	CMP1PARfar1	medio	1706015151.56399	00000064639a2a6b174a3441b0f3002...	00000ceb5b19f0cb54608b3263e575b...	355597	BLOB
16	15	CMP1PARfar1	medio	1706015152.91789	00000ceb5b19f0cb54608b3263e575b...	000006220a109284d27341001d1be9...	141258	BLOB
17	16	CMP1PARfar1	medio	1706015153.46252	000006220a109284d27341001d1be9...	00000b589fdcbda40afe48e33c63be04...	906263	BLOB
18	17	CMP1PARfar1	medio	1706015156.90098	00000b589fdcbda40afe48e33c63be04...	000007a0132da0da97d616a03f5e439...	929990	BLOB
19	18	CMP1PARfar1	medio	1706015160.3792	000007a0132da0da97d616a03f5e439...	00000b2425dc9d28ad5e4a8e4bc1f5c...	5100140	BLOB
20	19	CMP1PARfar1	medio	1706015184.53625	00000b2425dc9d28ad5e4a8e4bc1f5c...	00000f5fa7f1f2a5fc4a0b6e844a50587...	1284353	BLOB
21	20	CMP1PARfar1	medio	1706015190.62955	00000f5fa7f1f2a5fc4a0b6e844a50587...	0000031db733613cd0b5e6cde07b8a3...	2117300	BLOB
22	21	CMP1PARfar1	medio	1706015198.67808	0000031db733613cd0b5e6cde07b8a3...	000004350d99447d8ab127f5c025e13...	2099692	BLOB
23	22	CMP1PARfar1	medio	1706015206.63411	000004350d99447d8ab127f5c025e13...	000001bcdb09a2839d84b48ae384d46...	1062424	BLOB
24	23	CMP1PARfar1	medio	1706015210.63292	000001bcdb09a2839d84b48ae384d46...	00000d25d56d7b9a9c354f66d609164...	1284420	BLOB
25	24	CMP1PARfar1	medio	1706015215.49655	00000d25d56d7b9a9c354f66d609164...	00000bc673be4afcdf2859b942e547d0...	51000	BLOB
26	25	CMP1PARfar1	medio	1706015215.70487	00000bc673be4afcdf2859b942e547d0...	000005041d27e1397e81622597963cd...	2109405	BLOB
27	26	CMP1PARfar1	medio	1706015223.74462	000005041d27e1397e81622597963cd...	0000091d900f79d18dfd8b983267646...	1368270	BLOB
28	27	CMP1PARfar1	medio	1706015228.91627	0000091d900f79d18dfd8b983267646...	00000f0ed8da109eb419342a0ee965f...	3430587	BLOB
29	28	CMP1PARfar1	medio	1706015241.90875	00000f0ed8da109eb419342a0ee965f...	00000fb278207190d6aff3b379be1b32...	4808579	BLOB

Tabla 4- Cadena de bloques utilizada para la evaluación del sistema tras la agregación masiva de bloques

7- Conclusiones y futuro trabajo

7.1- Conclusiones

La evaluación de la presente propuesta sistema Blockchain ha demostrado un estar a la altura en cuanto a la eficacia, seguridad y adaptabilidad. La evaluación se ha centrado en validar la funcionalidad, seguridad y rendimiento del sistema, aspectos críticos para su aplicación exitosa en un ámbito tan sensible como el farmacéutico.

El sistema ha mostrado una gran funcionalidad, manejando de forma eficiente y precisa las operaciones de consultas de stock, consultas de precios, compras y recargas de saldo. La implementación de pruebas funcionales confirma que cada componente operativo del sistema responde de manera adecuada a las solicitudes, garantizando la integridad y fiabilidad de las transacciones. Además, la capacidad del sistema para manejar múltiples operaciones en periodos cortos de tiempo sin comprometer su estabilidad o rendimiento subraya su robustez y su preparación para enfrentar escenarios de alta demanda.

En términos de seguridad, el sistema ha probado su solidez. La implementación de controles rigurosos de acceso, verificación de permisos y gestión de roles asegura que solo los usuarios autorizados puedan realizar operaciones, minimizando así el riesgo de acciones no autorizadas o malintencionadas. La integridad de la cadena de bloques se mantiene sólida gracias a mecanismos como el uso de “triggers” en la base de datos SQLite, que previenen alteraciones en los bloques ya integrados a la cadena.

En conclusión, la evaluación del sistema ha confirmado su capacidad para funcionar de manera eficiente, segura y fiable.

7.2- Recomendaciones para futuras investigaciones

El sistema Blockchain desarrollado ha demostrado ser una solución robusta y confiable, proporcionando un nivel significativo de seguridad, trazabilidad y cumplimiento. Sin embargo, como en toda solución tecnológica, siempre hay margen para el perfeccionamiento y la innovación. A continuación, se presentan algunas recomendaciones para investigaciones futuras que podrían ampliar aún más las capacidades del sistema y explorar nuevos ámbitos en la aplicación de Blockchain en sectores críticos.

Implementación de un Sistema Peer-to-Peer (P2P)

Con una mayor disponibilidad de recursos, sería beneficioso desarrollar un sistema P2P para fortalecer la descentralización del sistema. Un enfoque descentralizado no solo reforzaría la seguridad al distribuir los datos a través de múltiples nodos, sino que también aumentaría la resistencia del sistema frente a puntos de fallo únicos. Además, un sistema P2P permitiría una verificación más robusta y una distribución más equitativa del poder computacional, alineándose con los principios fundamentales de la tecnología Blockchain.

La implementación de un sistema P2P requiere una serie de pasos sistemáticos y bien estructurados. Inicialmente, se llevaría a cabo una evaluación de la infraestructura actual, con el objetivo de identificar los componentes susceptibles de ser descentralizados. A continuación, definiríamos los requisitos técnicos del sistema, abarcando aspectos como el rendimiento, la seguridad y la funcionalidad esperada.

Posteriormente, habría que investigar la selección de tecnologías apropiadas, inclinando hacia soluciones de código abierto que faciliten la integración y la colaboración futura. La adaptación y el rediseño de los componentes del proyecto para operar eficientemente en un contexto descentralizado es el siguiente paso.

La fase de prototipado emerge como un elemento para la validación de la viabilidad técnica, permitiendo ajustes iterativos antes de la implementación a gran escala. En los casos en que se incorpore tecnología Blockchain, se deberá integrar adecuadamente la funcionalidad de los nodos y los contratos inteligentes.

Las pruebas de seguridad y rendimiento deben proporcionar una base sólida para un despliegue gradual y controlado. El monitoreo y la optimización continuos basados en la retroalimentación y el análisis del rendimiento son indispensables para el refinamiento del sistema.

La documentación completa y el desarrollo de una comunidad en torno al proyecto contribuirán significativamente al soporte y la evolución del sistema P2P. Por último, la escalabilidad del sistema debe asegurar que la red pueda manejar un incremento en la carga de trabajo y en el número de nodos participantes de manera efectiva.

Integración con Tecnologías de Contratos Inteligentes

La integración de contratos inteligentes podría automatizar y asegurar aún más las transacciones dentro del sistema. Estos contratos podrían usarse para ejecutar acuerdos automáticamente cuando se cumplen ciertas condiciones, reduciendo la necesidad de intermediarios y mejorando la eficiencia operativa. En el contexto farmacéutico, los contratos inteligentes podrían gestionar acuerdos de suministro, asegurar la trazabilidad de los medicamentos y automatizar los procesos de pago y facturación.

Para la implementación y automatización de los procesos de pago y facturación hay que comenzar con la selección de un procesador de pagos y este debe basarse en la compatibilidad con las necesidades del proyecto, asegurando que ofrezca una integración técnica sin problemas, tarifas competitivas y altos estándares de seguridad. Una vez seleccionado, es fundamental asegurarse de que el sistema cumpla con todas las regulaciones financieras pertinentes, incluidas las normas KYC (Know Your Client), AML (Anti-Money Laundering) y PCI DSS (Payment Card Industry Data Security Standard), para garantizar la legalidad y la seguridad de las transacciones. La fase de integración técnica implica trabajar con las API del procesador de pagos para incorporar la funcionalidad de pago en la plataforma del proyecto, lo que incluye configurar correctamente los carritos de compra, los botones de pago y las gestiones de suscripciones. Finalmente, se implementan medidas de seguridad robustas, como el cifrado de datos y la autenticación de dos factores, y asegurarse de que todo el sitio web opere bajo un protocolo HTTPS con certificados SSL válidos, para proteger las transacciones y la información personal de los usuarios contra accesos no autorizados y otros riesgos de seguridad cibernética.

Mejoras en la Escalabilidad y Eficiencia del Sistema:

Aunque el sistema ha demostrado un rendimiento sobresaliente, siempre hay espacio para mejorar la escalabilidad y eficiencia. Investigaciones futuras podrían explorar nuevos algoritmos de consenso que consuman menos recursos y ofrezcan tiempos de respuesta más rápidos, especialmente importantes en operaciones que involucran un gran volumen de transacciones. Además, se podrían investigar técnicas de sharding o particionamiento de la cadena para mejorar la capacidad de manejo de datos y la velocidad de las transacciones.

Estas recomendaciones tienen como objetivo no solo mejorar la eficiencia y la seguridad del sistema actual sino también explorar nuevas posibilidades y aplicaciones de la tecnología Blockchain. Al seguir estas recomendaciones, las investigaciones futuras podrían llevar a innovaciones significativas y a una adopción más amplia de esta tecnología en la industria farmacéutica y otros sectores críticos.

8- Referencias

[1] Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System.

[2] Bashir, I. (2024). *Mastering Blockchain*.

[3] Nakov, S. *Practical Cryptography for Developers*.

[4] Menezes, A. J., Van Oorschot, P. C., & Vanstone, S. A. (2018). Handbook of applied cryptography.

[5] Kurki, J. (2016). Benefits and guidelines for utilizing Blockchain technology in pharmaceutical supply chains.

[6] Cryptographic primitives in Blockchain technology. Andreas Bolfing, Oxford university Press, 2020.

[7]Washington, L. C. (2008). *Elliptic Curves: Number Theory and Cryptography*. Chapman and Hall/CRC

[8] Bitcoin key mechanism and elliptic curves over finite fields. John D. Cook. https://www.johndcook.com/blog/2018/08/14/bitcoin-elliptic-curves/

[9] National Institute of Standards and Technology. (2013). *Digital Signature Standard (DSS) (NIST FIPS 186-4)*. U.S. Department of Commerce. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf