UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Universidad de las Ciencias Informáticas

Facultad 1

Sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor:

Alejandro Lázaro Medero Lacosta

Tutor:

MS. c. Hubert Viltres Sala

La Habana, diciembre de 2022

"Año 64 de la Revolución"

Agradecimientos:

DECLARACIÓN DE AUTORÍA

Declaro por este medio que yo **Alejandro Lázaro Medero Lacosta**, con carné de identidad 97121007623 soy el autor principal del trabajo titulado “Sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los _____ días del mes de _____

de _____

_____	_____
Alejandro Lázaro Medero Lacosta	**MS. C. Hubert Viltres Sala**
Autor	**Tutor**

Resumen

La comercialización de productos informáticos constituye una fuente de ingreso para instituciones y empresas que se dedican a la producción de software. Con los avances tecnológicos, la utilización de pagos electrónicos mediante el uso de criptomonedas permite realizar una transacción comercial de forma rápida y segura para clientes y proveedores. La Universidad de las Ciencias Informáticas como parte de su objeto social ofrece software y servicios a diferentes clientes; en este proceso de comercialización han existidos problemas relacionados a con la dificultad para realizar los pagos al no disponer de cuentas en bancos extranjeros y por las mismas limitaciones impuestas por el bloqueo económico y financiero de los EEUU. Con el objetivo de mejorar el proceso de pagos de los productos y servicios que brinda la Universidad se realizó un estudio sobre las principales plataformas de pagos mediante criptomonedas. El análisis de la bibliografía permitió identificar que la universidad no dispone de un sistema que posibilite el pago de sus productos mediante criptomonedas. Para solventar las deficiencias identificadas se diseñó e implemento un portal web para la comercialización de los productos y servicios de la Universidad mediante el uso de un token. La propuesta desarrollada fue validada mediante métodos teóricos y empíricos que permitieron comprobar que la solución propuesta es funcional, tiene un rendimiento adecuado, es segura y mejora el proceso de comercialización de los productos de la Universidad.

Palabras clave: criptomonedas, comercialización, pagos, portal web, token.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Introducción

El aumento del uso de las tecnologías de la información, hace cada vez aparezcan nuevas ramas de la materia informática. Comunidades del mundo se encuentran en investigación y desarrollo de diferentes áreas de la informática con el fin de encontrar soluciones innovadoras a nuevos problemas aparecen como parte de la evolución. *Blockchain* es una de estas ramas que busca la transparencia, seguridad, y descentralización de las bases de datos y sistemas informáticos.

En (Nakamoto, 2008) se introduce un concepto que solucionaría el problema de tener una entidad centralizada para poder realizar transacciones entre varios nodos. Propone un sistema *peer-to-peer* (persona a persona) de transferencia de valor que se conoce como Bitcoin. A este concepto de almacenamiento de información distribuida se le llamó *blockchain* y es la tecnología que hace posible la implementación de criptomonedas. *Blockchain* o cadena de bloques es, en simples palabras, una base de datos distribuida en donde se agrupan y guarda registro de todas las transacciones de forma ordenada dentro de bloques, al mantener cierta estructura que hace imposible la modificación de un registro. Como especifica el nombre, estos bloques están vinculados entre sí para generar esta estructura.

Desde su creación, las criptomonedas han despertado interés y altas expectativas entre los inversionistas y expertos en tecnología, por su alta rentabilidad y riesgo, pero también por las alternativas que representan en términos de las operaciones que se pueden realizar a través del protocolo de cadena de bloques, incluyendo recientemente nuevas alternativas para la financiación empresarial. En términos económicos y financieros, también se han generado dudas y críticas entre la comunidad académica y los organismos financieros supervisores y reguladores, esencialmente, por su carácter descentralizado, es decir, que no dependen ni son garantizadas por ninguna institución y por el anonimato que se genera dentro de la red, el cual abre las puertas a un sinfín de usos de toda índole(Nakamoto, 2008).

Según (Cabrera Soto, 2021) las criptomonedas o criptoactivo son un medio digital de intercambio que utiliza criptografía (encriptación) fuerte y diferentes protocolos para asegurar las transacciones, controlar la creación de unidades adicionales y verificar la transferencia de activos. Las características comunes de la mayoría de los criptoactivos son: No tienen representación física, su emisión es descentralizada, no son controlados por ningún Estado o entidad financiera, con frecuencia operan en circuitos trasnacionales y no necesitan intermediación.

Los *token* en criptomonedas se utilizan para facilitar las transacciones. No son autosostenibles, es decir que requieren de una cadena de bloques para poder operar, y se generan en formato de *Smart contracts* (contratos inteligentes). Estos son los que determinan el funcionamiento y limitaciones de este *token* particular dentro de la tecnología *blockchain*(Nakamoto, 2008).

Según (Ramírez, 2021) un *token* es la representación digital en el mundo *Blockchain* de algo que tiene valor dentro de un contexto. Es emitido por una entidad privada y solo es válido bajo este universo concreto.

En el universo cripto son representadas a través de piezas de código que contienen información intercambiable y que, por ende, aportan la connotación de dinero o de activo digital (Ramírez, 2021).

Un *token* entonces, puede usarse para diferentes finalidades más allá que el de forma de pago: puede servir para pagar un objeto, para participar en un evento, puede ser una acción dentro de una empresa, un elemento coleccionable, una entrada a un juego, o cualquier cosa pueda ser representada en el mundo real (Ramírez, 2021).

Todos los *token* criptográficos se dividen en dos categorías: los *token* fungibles y no fungibles. En muchos lugares se encontrará que las criptomonedas suelen confundirse con los *token* o las *altcoins*. Al tener clara la categorización de *token* y su significado se presenta los diferentes tipos de tokens de criptomonedas: *Utility token* (*Token* de utilidad), *Security* (*Equity token*), *Governance token*(*Token* de gobernanza) y *Asset token*(*Token* de activos) (Ramírez, 2021).

El uso de las criptomonedas se ha transformado en un vehículo que da seguridad a muchas transacciones financieras llevadas a cabo a través de Internet. La importancia de las monedas digitales es que se apoyan en un lenguaje críptico que les proporciona un rango de seguridad bastante elevado. Es en el empleo de este lenguaje que las criptodivisas (o criptoactivos) se pueden encontrar en una feliz reunión con la Inteligencia artificial(González, 2020).

Las monedas digitales poco a poco han ocupado un mayor espacio en los negocios porque se muestran capaces de garantizar el éxito de toda transacción. Una de sus ventajas es que posibilitan el monitoreo del proceso de transferencia del dinero de unas manos a otras, por medio de su poderosa tecnología. En unión de la IA incrementan los niveles de acierto en relación con todos los momentos de definición de cursos de acciónінформática que contribuya al proceso de pagos digitales se potenciará el desarrollo de software en la UCI(González, 2020).

La Oficina de Control de Activos es un organismo de control financiero dependiente del Departamento del Tesoro de los Estados Unidos. Se ocupa de la aplicación de las sanciones internacionales estadounidenses en el ámbito financiero, sobre todo en el marco de la

protección de la seguridad nacional y en el apoyo de la política exterior de los Estados Unidos. Como un componente del Departamento del Tesoro de los Estados Unidos, la OFAC opera bajo mando de la Oficina de Terrorismo e Inteligencia Financiera y está compuesta principalmente por asesores de inteligencia y abogados.

Si bien la Casa Blanca establece en términos generales muchos de los objetivos de la OFAC, la mayoría de los casos individuales se desarrollan como resultado de las investigaciones realizadas por la Oficina de Globalización de Objetivos de la OFAC (OGT) (Granma, 2022).

El bloqueo económico y financiero estadounidense a Cuba es un extenso entramado jurídico estadounidense que incluye leyes y regulaciones que prohíben y regulan las relaciones económicas con este país. el entramado jurídico estadounidense que sostiene el embargo contra Cuba está compuesto por los siguientes normas y leyes(KP, 2020):

-Ley de Comercio con el Enemigo, de 1917, sección 5.b
-Ley de Cooperación Internacional, de 1961, sección 620.a
-Regulaciones al Control de los Activos Cubanos, de 1963
-Ley para la Democracia en Cuba, de 1992, también conocida como Ley Torricelli
-Ley para la Libertad y la Solidaridad Democrática Cubanas, de 1996, también conocida como - Ley Helms-Burton
-Ley de Sanciones Comerciales e Incremento del Comercio, de 2000.

Los obstáculos a las operaciones bancarias internacionales de Cuba se han incrementado en los últimos años, debido al impacto del embargo estadounidense a la Isla, de acuerdo con fuentes oficiales. En los últimos tres años se han registrado «más de 850 acciones para impedir las operaciones con bancos extranjeros», según explicó Juan Jorge Galego, directivo del Banco Central de Cuba (BCC).

Tales medidas se aplicaron en un escenario muy complejo, caracterizado por el impacto de la pandemia de la COVID-19, en medio de la cual una parte importante de esas acciones estuvieron asociadas a la adquisición de medicamentos, materias primas y otros recursos para el sector de la salud, así como alimentos destinados a la población.

El funcionario también recordó que incluso desde antes de la llegada de Donald Trump a la Casa Blanca, desde el gobierno de Barack Obama, se han venido en conclusiones «significativas multas» a bancos extranjeros por sus operaciones con Cuba. Entre los multados, Galego mencionó a *Credit Suisse* de Suiza, HSBC *Bank* de Inglaterra, ING *Bank* de Holanda,

los franceses BNP *Paribas* y *Societé Generale Banque* y el *Commenrzbank* AG de Alemania, los cuales, dijo, «pagaron altísimos montos y cerraron sus operaciones con la Isla». Los obstáculos a las operaciones bancarias y transacciones financieras desde y hacia el exterior son reiteradamente referidos por autoridades e instituciones cubanas como uno de los principales costos del embargo estadounidense.

Según refieren, los mismos dificultan la adquisición de productos y materias primas, el cobro de exportaciones y servicios e, incluso, el pago de obligaciones contraídas, entre otras dificultades. Aumentan los obstáculos a las operaciones bancarias de Cuba por sanciones de EEUU (Arias Rivera, 2021).

En los años 2017-2022 han sido complicados para la economía cubana. Además de los problemas estructurales del modelo económico, se sumó la escalada de las sanciones de la Administración Trump, que continúan con Biden, el impacto de la crisis infinita de la economía venezolana y la pandemia. Las exportaciones han venido cayendo de manera sistemática en todo el período. El PIB desde 2019 entró en recesión, al tener una de las caídas más pronunciadas de América Latina en 2020. Se cerraron los aeropuertos y eso detuvo el turismo, que es uno de los sectores más importantes de la economía y en el cual descansan las esperanzas de recuperación postpandemia (González, 2020).

Finalmente, la reforma monetaria se puso en marcha en 2021, con impactos mixtos en la economía. Se ganó transparencia en los balances financieros de las empresas estatales y en los precios relativos, y propició que se abrieran nuevos espacios al sector privado. Pero, por otro, se aceleró la inflación y han quedado pendientes temas clave, como la convertibilidad del peso cubano, la dolarización y el diferencial con la tasa de cambio del mercado informal(Alejandro, 2022). En cadena de las crueles leyes Torricelli y Helms-Burton, respaldadas por el embargo económico impuesto injustamente a Cuba hace más de 60 años y las difíciles medidas que ha adoptado nuestro gobierno para salir adelante en esta situación.

Muchas instituciones en Cuba se ven afectadas directamente por la compleja situación del embargo económico y las relaciones políticas con EEUU, entre ellas se encuentra la Universidad de Ciencias Informáticas. La UCI funciona como centro de desarrollo de software, los cuales se comercializan y distribuyen a numerosos clientes alrededor del mundo, en muchas ocasiones se dejan proyectos a mitad de desarrollo por problemas en los pagos desde bancos exteriores. Los *token* funcionan como métodos de pagos digitales seguros y eficaces,

sin necesidad de intermediarios, lo cual hace una brecha en las sanciones impuestas a nuestro país. Es de vital importancia crear un *token* auténtico para la comercialización de servicios en el ecosistema universitario y se utilizará la descentralización de la tecnología *blockchain* para ser libre financieramente a la inversión.

Según informes del Departamento de pagos digitales de la UCI, muchos clientes y empresas extranjeras se interesan en todas las ramas informáticas de la universidad, pero, sin embargo, a lo largo de los años han existido serios problemas de financiamiento y se deja de ingresar casi un 90% de la posible ganancia bruta. Muchos bancos de América, y sobre todo los del Caribe, tienen relaciones directas con bancos norteamericanos, esto hace imposible que, dentro de la propia área del caribe, la universidad poder distribuir nuestros servicios. Con previo acuerdo al Banco Central de Cuba, bajo la orden de la RESOLUCIÓN 215/2021 publicada en la Gaceta Oficial de la República de Cuba, se atribuye a el *token* UCI la usabilidad en diferentes servicios, los cuales serán:

Cursos cortos académicos: Este servicio se realiza con personas naturales, sin embargo, las comisiones e impuestos de bancos latinoamericanos, sobre todo, hacen que se pierdan clientes por presentar tantas dificultades en las transferencias.

Aplicaciones desarrolladas: Se desea agregar micro pagos a aplicaciones desarrolladas por los informáticos profesionales, como son apklis, Picta, Todus, y en el caso de Cosmox (centro de desarrollo de videojuegos) crear un ecosistema de billeteras personales para que los clientes cajeen sus *token* por recompensas.

Servicios profesionales informáticos: Depende del tipo de metodología usada para la elaboración de proyectos informáticos, se incluirán pagos por cuotas, o directamente al principio y finalización del trabajo.

Trabajadores: Se puede incluir algunas recompensas motivacionales a los profesionales; se deposita en sus *Wallet* o carteras electrónicas, *token* generados por la deflación (característica del *token* más adelante explicada).

Al crear un *token* como nuevo sistema de pago para la Universidad de las Ciencias Informáticas, sin contar con las monedas estables USDT o BUSD, se contribuye a la autenticación del trabajo expansionista, y se excluye inversiones externas en dólares o divisas extranjeras que pasan actualmente por una crisis financiera global, y se le atribuye un uso primordial para el desarrollo interno del ecosistema universitario, y por supuesto, el apoyo económico a otras entidades. En el proceso de desarrollo de software a través de estos pagos electrónicos se formaliza una gestión de proyectos más influyente y mejor elaborada.

La gestión de riesgo y gestión de costos quedará mejor estructurada. Se contará con la comercialización de herramientas informáticas y demás materiales que se necesite al momento, de forma prácticamente inmediata. Debido a las características que se le asocian incrementará la demanda y a lo largo del tiempo, será mayor provecho económico, con respaldo de los bienes y servicios de la universidad.

Por lo planteado con anterioridad se define como **problema de investigación:** ¿Cómo facilitar el proceso de gestión de pagos en los servicios informáticos?

Como **objeto de estudio** de esta investigación se plantea: Sistema para la gestión de pagos electrónicos. Enfocado al **campo de acción**: Sistema para la gestión de pagos electrónicos que apliquen tecnología *blockchain* y *token*.

Para darle cumplimiento al problema planteado se define como **objetivo general**: Desarrollar un sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas.

Objetivos específicos:

1. Elaborar el marco teórico-referencial de la tecnología *blockchain*, criptomonedas, *token* y pagos electrónicos.
2. Seleccionar las herramientas relacionadas con los pagos electrónicos con las tecnologías *blockchain*, criptomonedas, *token.*
3. Diseño e implementación de un sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas.
4. Validación del sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas mediante las pruebas funcionales, de rendimiento, de seguridad e integración.

Hipótesis

Con el desarrollo del sistema para la Tokenización del pago de los bienes y servicios permitirá potenciar el desarrollo de software en la Universidad de las Ciencias Informáticas.

Métodos Científicos utilizados son

Métodos Empíricos

- **Encuestas:** Para obtener información sobre las características del *token*. Identificar los elementos que los clientes, desarrolladores o personas que interactúen con el *token* deseen agregar para una mejor funcionalidad. El instrumento a utilizar es el cuestionario.

- **Entrevista:** Para evaluar la importancia de la creación de un *token* como método de pagos digitales para la UCI. El instrumento a utilizar es la guía de entrevista.

Métodos Teóricos

- **Analítico-Sintético:** Para el análisis de teorías, materiales y documentos relacionados con la tecnología *Blockchain*, criptomonedas y *token*, con el objetivo de establecer las bases teóricas de la investigación.
- **Hipotético-Deductivo:** Se utiliza para la creación de la hipótesis planteada y a partir de ella derivar conclusiones en el transcurso de la investigación.
- **Histórico-Lógico:** Se realiza un estudio del desarrollo del token, para tener una visión global de la problemática planteada.

Estructura del documento

El presente trabajo de diploma tendrá la siguiente estructura: introducción, tres capítulos, conclusiones y recomendaciones. Además, se agregan los anexos que permiten comprender mejor la investigación realizada.

En el **Capítulo 1. Fundamentación teórica para el proceso de gestión de pagos de servicios a través de un token,** se realiza un análisis de los conceptos *blockchain*, criptomonedas y *token*. Se realiza un estudio de las herramientas para el desarrollo web de un sistema gestor capaz de procesar pagos a través de criptoactivos.

En el **Capítulo 2. Análisis y diseño del sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas** describe el desarrollo de las actividades correspondientes al análisis y diseño de la página web. Se explican las características principales de los productos de trabajo generados como resultado de esas actividades.

En el **Capítulo 3: Implementación y Pruebas del sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas**, se muestra una descripción de la validación propuesta. Se describe la estrategia de pruebas utilizadas y los resultados obtenidos.

Posteriormente se presentan las conclusiones generales, recomendaciones, referencias bibliográficas y los anexos de la presente investigación.

CAPÍTULO 1: Fundamentación teórica para el proceso de gestión de pagos de servicios a través de un token

En el presente capítulo se fundamenta toda la base teórica de la presente investigación. Se analizan las herramientas que resuelven problemáticas similares con el objetivo de evaluar su adaptabilidad a este contexto. Se plantea la metodología utilizada y se realiza una fundamentación de las tecnologías, herramientas y lenguajes a utilizar en todo el proceso de desarrollo del *token* UCI, así como su posible usabilidad.

1.1 Descripción de los conceptos asociados al dominio del problema

Para el desarrollo de la presente investigación se exponen los siguientes conceptos del negocio para un mejor entendimiento de los procesos:

Blockchain: Una *blockchain* es una base de datos que lleva registro de todas las transacciones que se llevaron a cabo en una red. Esta base de datos es replicada y distribuida entre todos los participantes. La principal característica es que permite que participantes que no se conocen y sin necesidad de confiar entre sí, puedan realizar transacciones de forma segura sin la necesidad de una tercera parte. Estas transacciones se procesan en bloques y cada bloque es identificado por un hash criptográfico(Jiménez y Nieves, 2019).

Cada bloque hace referencia al anterior, es por esto que recibe el nombre de cadena de bloques y una vez que el bloque es creado y agregado a la cadena no puede ser modificado o revertido, lo cual hace que se mantenga la integridad de las transacciones. Para procesar y verificar los bloques, nodos especiales llamados mineros, resuelven un algoritmo matemático y reciben una recompensa por invertir poder computacional y propagar los nuevos bloques a la red. El resto de los nodos verifican que los nuevos bloques sean verídicos y continúan construyendo sobre el de ser así. Estas *blockchain*, pilares de la estructura de cualquier criptomoneda, tienen la característica de poder ejecutar líneas de código conocidas como contratos inteligentes(Rossi y Houlin, 2021).

Contratos inteligentes: Los contratos inteligentes (*Smart contracts*), son un código ejecutable que funciona sobre una *blockchain* y, entre otras, tienen la capacidad de recibir y enviar criptomonedas entre los usuarios participantes del contrato automáticamente. También pueden leer información de fuentes externas y utilizarla en el proceso de cálculo, a través de un mecanismo conocido como *oracles*. Esta característica podría resultar útil para crear derivados financieros. por ejemplo, pueden tomar la cotización de mercado del activo para luego calcular y distribuir los *cash flows* entre los participantes(Rossi y Houlin, 2021).

Criptomonedas: Una criptomoneda es un activo digital que emplea un cifrado criptográfico para garantizar su titularidad y asegurar la integridad de las transacciones, y controlar la creación de unidades adicionales, es decir, evitar que alguien pueda hacer copias como haríamos, por ejemplo, con una foto. Estas monedas no existen de forma física: se almacenan en una cartera digital. Las criptomonedas cuentan con diversas características diferenciadoras respecto a los sistemas tradicionales: no están reguladas ni controladas por ninguna institución y no requieren de intermediaros en las transacciones. Se usa una base de datos descentralizada, *blockchain* o registro contable compartido, para el control de estas transacciones (Jiménez y Gallardo, 2022).

Características:

1. No tienen representación física.
2. Su emisión es descentralizada.
3. No son controlados por ningún Estado o entidad financiera.
4. Con frecuencia operan en circuitos trasnacionales.
5. No necesitan intermediación.

Token: El término “token” realmente describe a todos los productos de estas cadenas de bloques que representan el valor de la cadena y dan acceso a sus funcionalidades. No obstante, se suele llamar “criptodivisa” a representaciones de valor “auto hospedadas (generadas por la propia cadena de bloques) como Bitcoin, mientras que "token" se usa para describir aquellos alojados en cadenas de bloques de terceros.” (traducción del autor) (Penny). Así, los *token* provenientes de las ICO están construidos sobre una cadena de bloques ajena y dependen de su capacidad computacional para funcionar. A cambio, la plataforma “padre” ve aumentado el tráfico en su red al ser necesaria su criptomoneda para comprar los *token* (Zorrilla Moreno, 2019).

Exchange: Los *exchanges* de criptomonedas, mercado de criptomonedas o intercambio de criptomonedas son plataformas o mercados digitales que permiten intercambiar monedas digitales por dinero *fiat* y/u otras criptomonedas o mercancías. Generalmente se trata de plataformas centralizadas, las cuales pueden tener una dinámica automatizada muy parecida a la de otros tipos de plataformas de inversión (como las denominadas "cripto-bolsas"), basarse en mecanismos que faciliten la compra y venta de persona a persona mediante sistemas de anuncios, puntuación y depósito de garantía o incluso acudir a un esquema de compra y venta de empresa a persona(Zorrilla Moreno, 2019).

Wallet: Un monedero digital o *wallet* es, en realidad, un software o aplicación donde es posible almacenar, enviar y recibir criptomonedas. Lo cierto es que, a diferencia de un monedero de dinero físico, lo que realmente se almacena en los *wallets* o monederos digitales son las claves que nos dan la propiedad y derecho sobre las criptomonedas, y nos permiten operar con ellas. Expresado de otra forma, basta con conocer las claves para poder transferir las criptomonedas, y la pérdida o robo de las claves puede suponer la pérdida de las criptomonedas, sin posibilidad de recuperarlas(Jiménez y Gallardo, 2022).

Trading: El *trading* es un negocio en el cual las personas estudian los mercados financieros para invertir en diferentes productos financieros con el objetivo de sacar una rentabilidad; el trading se ha convertido en una actividad cada vez más usada como camino para alcanzar la anhelada libertad financiera.
El trading ofrece a las personas una gran cantidad de oportunidades para trabajar desde cualquier lugar del mundo, a cualquier hora y con la facultad de producir dinero en cantidades inimaginables; después de todo, también posee la propiedad de la liquidez, y en tan solo unas horas puedes cobrar el dinero realizado y gastarlo sin tener que esperar hasta los últimos días del mes (Ordoñez, 2018).

Minería: La minería de criptomonedas es el proceso en el que los mineros utilizan la potencia informática (hash), para procesar transacciones y obtener recompensas, en este caso criptomonedas**.** Expresado de otra forma, es el proceso de agregar nuevos registros de transacciones como bloques a la cadena de bloques. O también, que es el proceso de registrar transacciones de *blockchain*, a cambio de una recompensa, entregada en el mismo tipo de criptomoneda que se mineen. Por ejemplo, si se mina Bitcoin la recompensa es en BTC. Una de las principales características de la minería es la concentración de recursos. Es decir, que requiere de una gran cantidad de potencia informática que pueda cumplir con las exigencias de minería. Además de permitir que todos los participantes de la red minera se pongan de acuerdo sobre la eficacia y precisión de la cadena de bloques(Santaella, 2021).

Descentralización: Si se habla de descentralización en *blockchain*, generalmente se relaciona con la transferencia de tareas de supervisión y toma de decisiones de un grupo centralizado, que puede ser un individuo, una corporación o un grupo de personas hacia una red dispersa (Díaz, 2022).

Economía de un token o Tokenomics: representación de manera abstracta de un valor en correspondencia con el activo real. Los empresarios e innovadores han comenzado a darse cuenta del poder disruptivo de la tecnología *blockchain* y de los *token*. De hecho, en este

escenario han cobrado importancia las ICO o *Initial Coin Offerings* (ofertas iniciales de moneda) como forma de financiación empresarial. Estas venden una serie de *token* a los primeros usuarios a cambio de criptomonedas en una suerte de ronda de financiación alternativa (en tanto en cuanto no es el circuito habitual de los bancos o inversores en capital riesgo), lo que posibilita la captación de fondos para muchas empresas emergentes. Sin embargo, los *token* no son solo una nueva forma de recaudar fondos, sino que suponen una nueva vía de construir ecosistemas, en remodelar el espíritu empresarial (Chen, 2018).

1.2 Descripción del negocio

El *token* propuesto debe tener ciertas características que lo hagan perdurable y eficiente en el tiempo. La usabilidad que puede adquirir con el tiempo es muy prometedora para el desarrollo de aplicaciones. El *token* debe presentar los siguientes parámetros:

Durabilidad: Se propone tener un *token* que perdure en el tiempo junto a los equipos de desarrolladores, por lo cual se debe implementar de carácter infinito, para que siempre tenga disponibilidad para el cliente solicitar algún servicio.

Escalabilidad: Tiene que presentar un sistema que a lo largo del tiempo mejore la calidad de servicios que ofrece, a su vez que nuestros desarrolladores, profesores o estudiantes encuentren en el token un refugio de valor para sus cuentas digitales.

Minteo: Se tendrá un *token* con un suministro total no fijo; se adaptará a la demanda de los clientes. Se propone sacar a disposición cantidades cada cierto tiempo, ya depende de la planificación puede ser mensual, trimestral, etc.

Burn: Se le denomina *burn* a la acción de quemar o hacer desaparecer *token* en circulación, para limitar la cantidad existente en circulación y proponer por cada transacción un % mínimo destinado a recompensa de *holders* del *token*, así aumentaría el interés de los clientes para adquirirlo y recibirían recompensas pasivas por solamente tenerlo en *wallet*.

Gobernanza: A pesar de ser montado en una *blockchain* descentralizada, el *token* tendrá su propio proyecto de respaldo (Universidad de las Ciencias Informáticas) y se regirá por las normas corporativas que le sean asociadas, para un mayor control y beneficio propio.

Respaldo monetario: Con previo acuerdo al Banco Central de Cuba y el Banco de Comercio Exterior, inicialmente se puede respaldar en monedas *fiat* fuertes (Euro, Dólar, Yen, Libras Esterlinas) o monedas de curso legal.

Programación: El *token* debe ser reprogramable para en caso de cambios estructurales, se les pueda realizar directamente y no acudir a llevar a cabo el desarrollo de otro *token*.

A continuación, un ejemplo práctico del objetivo de esta propuesta de investigación:

La Universidad de Las Ciencias Informáticas tiene centros de desarrollo y gestión de software, famosos por su excelencia y eficiencia en los servicios que ofrece. Se acerca un cliente tanto nacional o internacional, y reunido con el representante del tipo de software deseado, se lleva a cabo un contrato para ambas partes y se elige la metodología a emplear, las condiciones, etc. La UCI, con previo acuerdo con el Banco Central de Cuba y el Banco de Comercio Exterior, y en uso del sitio web propuesto, le solicita al cliente dirigirse a alguna de estas entidades para realizar el intercambio o depósito correspondiente por el servicio solicitado.

En el banco se procede a intercambiar los *token* con un precio de utilidad a 1 USDT =1 UCI (nombre del *token*) para de esta forma contar con el dinero *fiat* en una piscina de liquidez. Luego estos *token* a través de la tecnología *blockchain* se transfieren a la *wallet* de la UCI para comenzar el proceso de desarrollo, hasta concluirlo. Este proceso pretende resolver y fomentar el desarrollo económico tanto de la Universidad, como del país.

Al cliente depositar la moneda fuerte, se contará con más capital para invertir en las necesidades del país, y a su vez, con los *token* adquiridos por la UCI, se podrá intercambiar por USDT en las *exchanges* que se logre listar el *token*, en consecuencia, la ganancia sería doble por parte de las instituciones involucradas, y el cliente solo invertirá una única vez. Otra solución con vistas al futuro es la siguiente: Se sabe que para la creación de un *token* se necesita de una liquidez inicial para poder listarla, hasta ahora Cuba cuenta con las monedas de curso legal, o respaldo de bienes y servicios de la Universidad.

Mediante los pagos de servicios de la forma anteriormente descrita se puede crear una piscina de liquidez para en algún momento el *token* tenga su propio respaldo y no necesite de inversión para salir al mercado, puesto que la misma fue generada a través de su propia usabilidad, y hará que tenga un valor mayor a lo largo del tiempo.

1.3 Estudio de sistemas homólogos

La tecnología *Blockchain* es famosa por su transparencia y descentralización. Esto se requiere principalmente en el sector financiero. Varias empresas consideran el desarrollo de aplicaciones financieras modernas en sistemas *blockchain*. Esto es el enfoque de varias empresas. La tecnología ha facilitado el desarrollo de nuevas aplicaciones a través de contratos inteligentes(Parmar, 2021).

1.3.1 Ámbito internacional

Overstock es conocido predominantemente como un negocio de comercio electrónico que proporciona una gama de productos como muebles, decoración del hogar y artículos para el hogar. Sin embargo, a partir de 2014, *Overstock* implementó un enfoque multifacético para

aumentar su exposición a la tecnología *blockchain* y las criptodivisas. En primer lugar, la empresa se asoció con *Coinbase* para permitir a sus clientes de comercio electrónico pagar con bitcoin, y *Overstock* añadió la flexibilidad de mantener partes de este bitcoin ganado en su balance. En segundo lugar, *Overstock* comenzó una iniciativa más holística para desarrollar y avanzar en la tecnología *blockchain* a través de sus filiales que se denominaron colectivamente el negocio Medici. El objetivo de Medici era aprovechar las propiedades de transparencia y seguridad de la tecnología *blockchain* para resolver problemas en seis áreas principales: gestión de la identidad, derechos de propiedad y gestión, banca central y divisas, mercados de capitales, cadenas de suministro y comercio, y sistemas de votación.

El negocio de Medici estaba compuesto principalmente por Medici Ventures, un negocio de capital riesgo que invierte en empresas de *blockchain*, y tZERO un negocio que aplica la tecnología de *blockchain* a los mercados de capitales, predominantemente en el área de los valores de activos digitales.(Society, 2022)

Microsoft: El gigante de la tecnología computacional empezó a introducir el Bitcoin como medio de adquisición de productos y servicios a partir del 2014. Desde entonces, la empresa dispone de la plataforma *BitPay* para transferir dinero a cuentas Microsoft. Desde allí es posible gestionar las compras de videojuegos, aplicaciones y otros artículos. Por supuesto, este tipo de transacciones pueden ser gestionadas a través de *Changelly*. Esta plataforma de *exchange* te proporciona excelentes tasas por transacción para comprar, vender o intercambiar criptodivisas. Aunque Microsoft no acepta el bitcoin para el pago de otros productos, la compañía ha asegurado que esta novedad coloca a la tecnológica en primera línea. El uso de divisas digitales, cuando aún no son masivas, crece más allá de los primeros entusiastas.(Financial, 2014)

Bitrefill: Se trata de la multinacional más grande del mundo dedicada a ser una pasarela de pago para adquirir tarjetas de regalo. Bitrefill comenzó su andadura en 2016 y ha tenido un crecimiento descomunal, con más de 1.650 negocios asociados en 170 países, entre ellos Cuba (se puede obtener recargas internacionales y tarjetas prepagos de ETECSA). Su grado de notoriedad en la actualidad es realmente relevante, mencionada en medios tan importantes como Forbes, *FXStreet* y *Yahoo Finance*. El motivo de su éxito radica en que ofrece un servicio novedoso cada vez más demandado por miles de personas alrededor del mundo: el uso práctico de las criptomonedas**.** Posee como característica contar con un *token* propio creado en la red BNB, pero se pueden realizar las compras en cientos de criptomonedas, además de divisas. (Financial, 2014)

Telegram: Telegram es un servicio de mensajería instantánea basado en la nube con código abierto del lado del cliente, desarrollado por Telegram Messenger LLP, fundado por Pavel Durov. Las aplicaciones de Telegram brindan a los usuarios la capacidad de enviar mensajes, fotos, videos, audio, *stickers* y archivos de cualquier tipo, y están disponibles para la mayoría de los sistemas operativos. Telegram es famoso por sus funciones de privacidad, como los mensajes encriptados de cliente-servidor, el cifrado de extremo a extremo opcional para llamadas de voz y los chats "secretos" cifrados de extremo a extremo que eliminan información después de un período de tiempo. Una de las novedades de Telegram, es la de poder integrar ahora pagos a través de criptomonedas de TON con el *Bot* @donate. Hace cuatro años, parte del equipo de desarrolladores de Telegram, iniciaron un proyecto que trabaja sobre *blockchain* de una nueva y mejor manera. Esta nueva generación se llamó “TON”, y su misión es la de crear un *token* que revolucionaría la manera en que se almacenaban fondos de este tipo. De esta manera, Nikolai y Pavel Durov, ambos fundadores de Telegram, decidieron cambiar la manera en que Bitcoin y Ethereum trabajan.(Pallares, 2022)

1.3.2 Ámbito Nacional

Hasta noviembre del 2022 en la bibliografía consultada, no se identificó la utilización de *token* para el comercio de bienes y servicios en Cuba. plantea:

En Cuba ya se han dado pasos de avance en tal sentido y, actualmente, se desarrollan varios proyectos que, como base, emplean las lógicas del *blockchain*. Uno de ellos, es el que se implementa desde 2019 con la empresa Tecnomática para establecer la trazabilidad del combustible de aviación. Esto le va a permitir a Cupet ser pionera en el proceso de trazabilidad y, gracias a esta herramienta digital, sustituir la documentación física, escrita, que es engorroso y ralentiza el trabajo.

Otro proyecto viene de la mano de la Empresa de Tecnologías de la Información, perteneciente a BioCubaFarma, y que tiene como propósito crear un sistema descentralizado, basado en tecnología de cadenas de bloques, para la identificación de propietarios de medicamentos y la gestión de su información y transferencia.

Lo que se propone (Antón y González, 2022), es tokenizar todos los fármacos que se comercializan y producen en el país, con lo cual habrá una unidad de criptomoneda por cada blíster disponible en el sistema de farmacias y hospitales.

Como resultado, habrá una mejor logística respecto a las tenencias de los medicamentos de cada producto y de las transferencias. De los movimientos que se hacen entre las droguerías,

se dota al proceso de mayor control y evita algunos hechos indeseables relacionados, casi siempre, con actividades ilícitas en la adquisición y compra de medicamento.

En fases posteriores, se quiere llegar incluso a que las prescripciones del sistema de salud se realicen por la misma vía, un método que en 2022 solo utilizan las farmacéuticas más avanzadas y modernas del mundo.

También se tiene el objetivo de concretar un monedero virtual de conjunto con la Empresa de Tecnologías de la Información para la Defensa (Xetid) y la pasarela para pagos digitales Enzona.

Este último proyecto, resulta ser el de uso común de la tokenización, ya que no es necesario contar con una tarjeta magnética para efectuar la compra y se garantiza la seguridad de los datos de los usuarios y facilita las transacciones.

Soluciones que ya están sobre la marcha y que aportarán a esa transformación digital a la que busca llegar el país, así deberán desarrollarse sin dilaciones innecesarias ni escollos burocráticos.

También está el reto de explotar mejor el potencial de las criptomonedas, y que, desde el 15 de septiembre del 2021, entró en vigor la Resolución 215/2021, del Banco Central de Cuba, vinculada a la regulación del uso de estos activos en la Isla.

1.3.3 Resultados obtenidos del análisis de sistemas Blockchain con características similares

Después de haber realizado un análisis de las *blockchain* y características anteriores, se puede constatar que a pesar de ser sistemas innovadores que gestionan las transacciones y que brindan un punto de encuentro para los desarrolladores, estos han sido adaptados para darle solución a problemas específicos de las instituciones, lo que dificulta su adopción por cualquier otra. Sus aplicaciones no cumplen en su totalidad con las propiedades deseables para el desarrollo de la propuesta de solución. De manera general no se adaptan a las necesidades específicas de la situación, sin embargo, aportan a la investigación elementos ideales a implementarse en la propuesta de solución.

El *token* propuesto contará con características singulares en el mundo de las *blockchain* y que serán del agrado de los inversores futuros y desarrolladores. Tiene potencial para ser pionera en el mundo *blockchain* como *token* de gobernanza, que se muestran a continuación:

Características		BTC	ETH	ADA	XRP	XLM	EOS	TRX	UCI
Durabilidad	finito	x	x						
	infinito			x	x	x	x	x	x
Escalabilidad	si		x	x		x			x
	no	x			x		x	x	
Minteo	si			x				x	x
	no	x	x		x	x	x		
Burn	si			x		x	x	x	x
	no	x	x		x				
Gobernanza	si								x
	no	x	x	x	x	x	x	x	
Programación	fija	x	x					x	
	reprogramable			x		x	x		x
Recompensa Hold	si							x	x
	no	x	x	x	x	x	x		
Stable coin	si		x					x	
	no	x		x	x	x	x		x

Figura 1 Criterio de comparación para el estudio de homólogos en criptomonedas.

Después de profundizar en algunos de los sistemas de facilidad de pagos similares en el mundo y posibles soluciones en Cuba, se percibe un abundante enramado de opciones para justificar la importancia del uso de los *token* y criptomonedas en pagos digitales. En justificación de algunas características particulares de los métodos que se usan para facilitar esta opción, se realiza una comparación en la tabla siguiente que evidencia la diversidad de todas. Atributos como el tipo de dominio (privado o público) permite conocer el acceso a la web; la tecnología (web o móvil) desde que dispositivos podemos acceder a la plataforma; el ámbito (libre o empresarial) brinda información de que tipo de red *blockchain* es empleada en su funcionamiento; y la monetización (si o no) relaciona los posibles fee a pagar a la hora de usarse.

Tabla 1 Criterio de comparación para el estudio de homólogos en sitios de pagos por criptomonedas.

Características	Overstock	Microsoft	Bitrefill	Telegram
Dominio	Privado	Privado	Público	Público
Tecnología	Web/Móvil	Web	Web/Móvil	Web/Móvil
Ámbito	Empresarial	Empresarial	Libre	Libre
Monetización	Si	Si	Si	Si

Al finalizar la comparativa de algunos sitios que poseen la facilidad para pagar con sus propios *token* o criptomonedas, podemos llegar a la conclusión que cada uno brinda una visión distinta del mismo objetivo a lograr. No sería correcto mezclarlos, porque cada uno cumple su objetivo específico. Para el sistema de gestión que se desarrolla en esta investigación es fundamental

que el dominio sea público para que llegue a la mayor cantidad de clientes interesados en los servicios de la UCI, y sobre todo que la monetización no sea un obstáculo a la hora de cerrar negocios de interés. Por lo cual es necesario crear un nuevo sitio web, donde la comercialización sea el centro predominante, y las empresas e instituciones elegidas para el análisis no ocupan del todo las soluciones que se requieren.

1.5 Metodología, lenguajes de programación, tecnologías y herramientas

1.5.1 Metodología de desarrollo de software

El Proceso Unificado Ágil de Scott Ambler o *Agile Unified Process* (AUP) en inglés es una versión simplificada del *Rational Unified Process* (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio si se usan técnicas ágiles y conceptos que aún se mantienen válidos en RUP. El AUP aplica técnicas ágiles incluyendo:

1. Desarrollo Dirigido por Pruebas. (*Test Driven Development* - TDD en inglés)
2. Modelado ágil.
3. Gestión de Cambios Ágil.
4. Refactorización de Base de Datos para mejorar la productividad.

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva. De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero al modificar el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, a la que llamaremos Ejecución y se agrega la fase de Cierre (Rodríguez, 2014).

Con la adaptación de AUP que se propone para la actividad productiva de la UCI se logra estandarizar el proceso de desarrollo de software, se otorga cumplimiento además a las buenas prácticas que define CMMI-DEV v1.3. Se logra hablar un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajos. Se redujo a 1 la cantidad de metodologías que se usaban y de más de 20 roles en total que se definían se redujeron a 11 (Rodríguez, 2014).
A partir de que el Modelado de negocio propone tres variantes a utilizar en los proyectos (CUN, DPN o MC) y existen tres formas de encapsular los requisitos (CUS, HU, DRP), surgen cuatro escenarios para modelar el sistema en los proyectos, al mantener en dos de ellos el MC, y queda de la siguiente forma:

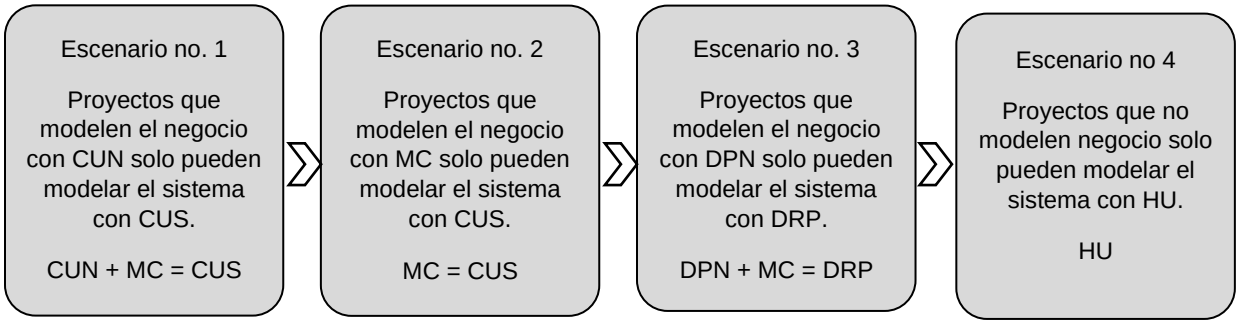

Figura 2 Representación gráfica de los escenarios de la metodología AUP-UCI.

Para el desarrollo del sitio web se decide hacer uso de el escenario 2 que aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

1.5.2 Herramientas y tecnologías

JavaScript 1.8.5: JavaScript es un lenguaje de programación de scripts (secuencia de comandos) orientado a objetos. Es utilizado para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Con JavaScript podemos crear efectos especiales en las páginas y definir las interacciones con el usuario. JavaScript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Toda esta potencia de JavaScript se pone a disposición del programador, que se convierte en el verdadero dueño y controlador de cada cosa que ocurre en la página (Álvarez, 2012).

En la investigación, permite aportarle el movimiento y dinamismo necesario para la ejecución y alcance de los objetivos previstos para esta, tanto del lado del desarrollador del sistema gestor, como en la parte del cliente.

Ventajas:

1. Nuevas posibilidades interactivas con el nuevo DOM y sus nuevas funciones.
2. Geolocalización del usuario.
3. Posibilidad de arrastrar y soltar cualquier elemento HTML.
4. Animaciones interactivas con *canvas*.
5. Almacenamiento de datos con *Session Storage* y *Local Storage*.

6. Trabajo offline con Cache de Aplicaciones, *File System API* y base de datos del lado del cliente.
7. Comunicación con el servidor Web Socket.

HTML5: HTML5 (HyperText Markup Language, versión 5) es la quinta revisión del lenguaje HTML. Esta nueva versión y en conjunto con CSS3, define los nuevos estándares de desarrollo web, al rediseñar el código para resolver problemas y actualizándolo así a nuevas necesidades. No se limita solo a crear nuevas etiquetas o atributos, sino que incorpora muchas características nuevas y proporciona una plataforma de desarrollo de complejas aplicaciones web(Gauchat, 2012).

Ventajas:

1. Nuevas etiquetas semánticas (los buscadores deben poder distinguir que es importante y que no dentro del código de una página web).
2. Multimedia sin *plugins* (audio, video y animaciones de acceso universal).
3. Formularios más usables y que validan automáticamente lo ingresado por el usuario, al reducir el uso de JavaScript.

CSS3: CSS es la sigla de *Cascading Style Sheets* (hojas de estilo en cascada). Es el lenguaje utilizado para definir la apariencia de los elementos definidos en un documento HTML. El CSS no es tanto un lenguaje de programación como lo es de reglas, una expresión básica en CSS se compone de dos partes, un selector y un conjunto de reglas. Cada regla a su vez se compone de dos fragmentos, una propiedad y un valor. Las reglas van al interior de corchetes, se separa la propiedad del valor con dos puntos, y las reglas se separan entre si con un punto y coma. Existen un gran número de propiedades que abarcan desde el tamaño y tipo de fuente hasta transformaciones geométricas y animaciones simples (Buenaventura, 2019).

Ventajas:

- Uso de cualquier tipografía con @font-face de CSS3 (aunque el usuario no la tenga instalada en su dispositivo).
- Nuevos modelos de color que incluyen translucidez y degradado sin imágenes.
- Efectos visuales como sombras en textos y cajas, bordes redondeados y con imágenes, múltiples fondos para un mismo elemento.
- Técnicas para manipular la presentación y el movimiento de elementos con CSS, como transformaciones, transiciones y animaciones.

Visual Studio Code 1.72.0: Visual Studio Code es un editor de código fuente ligero, pero de gran potencia que se ejecuta en las computadoras con sistemas operativos Windows, MacOS o Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js, y con un buen ecosistema de extensiones para otros lenguajes (Álvarez, 2012).

Es seleccionado para el desarrollo de la presente investigación porque permite configurar la interfaz a nuestro gusto para poder tener un código más visible, propicia el acceso rápido a las carpetas del proyecto y a una terminal con los detalles del problema o error. Su curva de aprendizaje es suave y existe una gran documentación al respecto.

Lenguaje para el modelado (UML) 2.1: El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual de propósito general que se utiliza para especificar, visualizar, construir y documentar los artefactos de un sistema software. Captura decisiones y conocimiento sobre sistemas que deben ser construidos. Se usa para comprender, diseñar, ojear, configurar, mantener y controlar la información sobre tales sistemas. Está pensado pasa ser utilizado con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre las técnicas de modelado e incorporar las mejores prácticas de software actuales en una aproximación estándar. UML incluye conceptos semánticos, notación y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está pensado para ser apoyado por herramientas de modelado visuales e interactivas que dispongan de generadores, tanto de código, como de informes. La especificación de UML no define un proceso estándar, pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos existentes.

UML capta la información sobre la estructura estática y el comportamiento dinámico del sistema. Un sistema es modelado como una colección de objetos discretos que interactúan para realizar un trabajo que en última instancia beneficia a un usuario externo. La estructura estática define tipos de objetos importantes para un sistema y para su implementación, así como las relaciones entre los objetos.

El comportamiento dinámico define la historia de los objetos a lo largo del tiempo y la comunicación entre objetos para cumplir los objetivos. El modelado de un sistema desde varios puntos de vista separados pero relacionados, permite entenderlo para diferentes propósitos(Mendoza Peña y Hernández González, 2021).

Ventajas:

1. Mejores tiempos totales de desarrollo (de 50 % o más).
2. Modelar sistemas (y no sólo de software) al utilizar conceptos orientados a objetos.
3. Establecer conceptos y artefactos ejecutables.
4. Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
5. Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
6. Mejor soporte a la planeación y al control de proyectos.
7. Alta reutilización y minimización de costos

Visual Paradigm 15.1: Visual Paradigm para UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción y despliegue. Es multiplataforma, utiliza UML como lenguaje de modelado, y tiene con una versión libre para la comunidad (Comunity Edition), al ayudar de una manera rápida a la construcción de aplicaciones de mayor calidad y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Esta herramienta también proporciona una mejor interfaz gráfica de usuario y una mayor base de datos de esquema de apoyo, permite generar la documentación del sistema en los formatos PDF, HTML y el formato de documentos de Microsoft Word y permite importar proyectos de otras herramientas de modelado como Rational Rose, Erwin y Microsoft Visio. Soporta la revisión ortográfica, al brindar sugerencias para los idiomas: inglés, español, francés, alemán y portugués(Mendoza Peña y Hernández González, 2021).

PostgreSQL 13.3: PostgreSQL es un potente sistema de base de datos relacional de objetos de código abierto con más de 30 años de desarrollo activo que le ha ganado una sólida reputación por su fiabilidad, solidez de funciones y rendimiento. Se puede encontrar una gran cantidad de información que describe cómo instalar y usar PostgreSQL a través de la documentación oficial. La comunidad de PostgreSQL proporciona muchos lugares útiles para familiarizarse con la tecnología, descubrir cómo funciona y encontrar oportunidades profesionales.

Reactjs 16.9: (también llamada React.js o ReactJS) es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre. En el proyecto hay más de mil desarrolladores libres.

React intenta ayudar a los desarrolladores a construir aplicaciones que usan datos que cambian todo el tiempo. Su objetivo es ser sencillo, declarativo y fácil de combinar. React sólo

maneja la interfaz de usuario en una aplicación; React es la Vista en un contexto en el que se use el patrón MVC (Modelo-Vista-Controlador) o MVVM (Modelo-vista-modelo de vista). También puede ser utilizado con las extensiones de React-based que se encargan de las partes no-UI (que no forman parte de la interfaz de usuario) de una aplicación web.

Node.js 18.12.1: es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como, por ejemplo, servidores web. Fue creado por Ryan Dahl en 2009 y su evolución está apadrinada por la empresa Joyent, que además tiene contratado a Dahl en plantilla. Node.js es similar en su propósito a Twisted o Tornado de Python, Perl Object Environment de Perl, libevent o libev de C, EventMachine de Ruby, vibe.d de D y JavaEE de Java existe Apache MINA, Netty, Akka, Vert.x, Grizzly o Xsocket. Al contrario que la mayoría del código JavaScript, no se ejecuta en un navegador, sino en el servidor. Node.js implementa algunas especificaciones de CommonJS. Node.js incluye un entorno REPL para depuración interactiva.

Cointools 1.04: Herramienta de terceros usada para la creación y desarrollo de *token*. Posee un amplio catálogo de servicios dedicado a los desarrolladores, donde se puede incluir características distintivas para cada proyecto desarrollado. También tiene una sala de auditoria profesional y es pionera en ofrecer *blockchain* con contratos inteligentes.

1.5.3 Arquitectura del sistema

Frontend: Frontend se encarga de estilizar la página de tal manera que la misma pueda presentar la información de forma agradable para el usuario. La persona responsable del Frontend, debe de conocer las técnicas de experiencia de usuario para brindar una mejor interacción entre la persona y la página que visita, así mismo debe tener conocimientos de diseño de Interacción para colocar los elementos de tal manera que el usuario las pueda ubicar de forma rápida y cómoda. Existen muchas tecnologías relacionadas a estos tres lenguajes que deben ser conocidas por el desarrollador Frontend. Por ejemplo, para JavaScript existen Angular y BackboneJS, los cuales se pueden apoyar en librerías como animate.css, JQuery y otras. También existen lenguajes de transferencia de información como XML, JSON y Ajax para hacer solicitudes al servidor sin necesidad de refrescar la página completa.

Backend: Se denomina Backend a la capa de acceso a los datos de un software que no es accesible para el usuario final. Además, esta capa contiene toda la lógica de la aplicación que maneja los datos. Cabe destacar que los datos de una aplicación se encuentran almacenados en una base de datos dentro de un servidor.

La persona encargada del Backend debe tener conocimientos, según el lugar donde trabaje, de los lenguajes del lado del servidor, como ser: Java, C#, PHP. Node.JS, entre otros. Además, de aquellos que interactúan con la base de datos, como ser: MySQL, PostgreSQL, SQLServer, MongoDB, entre otras.

API: Una API de REST, o API de RESTfull, es una interfaz de programación de aplicaciones (API o API web) que se ajusta a los límites de la arquitectura REST y permite la interacción con los servicios web de RESTfull. El informático Roy Fielding es el creador de la transferencia de estado representacional (REST).

Las API son conjuntos de definiciones y protocolos que se utilizan para diseñar e integrar el software de las aplicaciones. Suele considerarse como el contrato entre el proveedor de información y el usuario, donde se establece el contenido que se necesita por parte del consumidor (la llamada) y el que requiere el productor (la respuesta). Por ejemplo, el diseño de una API de servicio meteorológico podría requerir que el usuario escribiera un código postal y que el productor diera una respuesta en dos partes: la primera sería la temperatura máxima y la segunda, la mínima.

Conclusiones del capítulo

Durante este capítulo se desarrolló un análisis de los elementos teóricos-metodológicos que sirven como base al presente trabajo, al permitir de esta forma llegar a las siguientes conclusiones:

1. La definición de los principales conceptos asociados al dominio de la presente investigación y las relaciones entre estos, permite alcanzar una mayor comprensión de la propuesta de solución.
2. El estudio de las principales *blockchain* y ecosistemas criptos, donde el *token* propuesto en la presente investigación se logrará y contará con las características también descritas. Esto logra la autonomía y libertad financiera en la mayor parte de los centros de desarrollo de la Universidad.
3. Los avances en el sistema *blockchain* que recién se empiezan a aplicar en Cuba en algunas esferas que necesitan ser informatizadas, y que posibilidades nuevas brindan estos servicios.

4. Las herramientas y tecnologías antes seleccionadas son las más idóneas dentro del marco tecnológico establecido para facilitar la integración con otras aplicaciones que se están al desarrollar. Se define como metodología de desarrollo de software AUP en su formato UCI en el escenario 2, los lenguajes de programación JavaScript y HTML, la herramienta Visual Paradigm para un mejor diseño de diagramas e informes y Cointools para configurar las características del *token*.

Capítulo 2. Análisis y diseño del sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas

En todo desarrollo de sistemas de software es necesario seguir algunas especificaciones, que permitan a los desarrolladores tener una disciplina que haga que todas las etapas del desarrollo del sistema, sean coherentes y formales. El desarrollo del sistema que se propone, al ser una solución que pretende tener aplicación dentro del contexto de un problema real, tiene que seguir un proceso de análisis y diseño que proporcione las bases bajo las cuales se va a desarrollar el *token*. El objetivo de este capítulo es presentar los resultados que se obtuvieron una vez cumplidas las fases de Análisis y Diseño que propone la metodología AUP-UCI. Se detallan las características del sistema, y se especifican los requisitos funcionales y no funcionales. Se realiza la descripción de la propuesta de solución, para proporcionar un mayor entendimiento.

2.1 Características de la propuesta de solución

Con las herramientas y tecnologías seleccionadas en el capítulo anterior se propone desarrollar un Portal Web para la el sistema de pago digital del *token* UCI con el objetivo de beneficiar económicamente a la universidad y brindar la totalidad de servicios que se deseen comerciar. El portal cumplirá con la relación de requisitos funcionales y no funcionales que se definirán a continuación en el capítulo. Dándole la posibilidad a los clientes de tener disponibles los servicios donde se detallen las características que cada uno posee, los detalles particulares de las aplicaciones disponibles, así como la descripción del estado de disponibilidad de las aplicaciones. El administrador a cargo de recibir las notificaciones de pago, tiene la posibilidad de reelaborar algún servicio disponible, y comprobar las facturas.

2.2 Modelo conceptual

Un modelo de conceptual es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, presentado como uno o más diagramas de clases. Se pueden utilizar para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de software o de otro tipo. Los objetos del dominio o clases pueden obtenerse a partir de una especificación de requisitos (Mendoza Peña y Hernández González, 2021). Componen el modelo conceptual definido para la presente investigación:

Cliente: Encargado de solicitar y seleccionar el contrato que desee adquirir, además de realizar el pago correspondiente depende de la selección mediante una factura.

Factura: Almacena el pago que paga el cliente.

Contrato: Registro que realiza el cliente junto la factura para comercializar los servicios UCI, y que recibe el departamento comercial UCI.

Servicios UCI: Registro que muestra todos los servicios disponibles para comercializar a los clientes.

Aplicaciones: Se almacena todo lo referente a las aplicaciones disponibles para realizar los micro pagos integrados.

Informáticos: Se almacena todo lo referente a los servicios informáticos disponibles para realizar los pagos integrados.

Departamento comercial UCI: Departamento encargado de recibe los contratos elaborados por los clientes.

UCI: Entidad que administra y controla todo el proceso llevado a cabo por el departamento comercial de la UCI.

Centro de desarrollo: Entidad que recibe las solicitudes de la UCI para desarrollar lo solicitado y organiza el equipo de trabajo.

Equipo: Grupo de personas que pertenecen al centro de desarrollo y están disponibles para trabajar en los servicios solicitados.

Especialistas: Se almacena todo lo referente a los especialistas del equipo de desarrollo.

Estudiantes: Se almacena todo lo referente a los estudiantes del equipo de desarrollo.

Proyectos: Se almacenan los proyectos que se crean para ofertar en los servicios UCI.

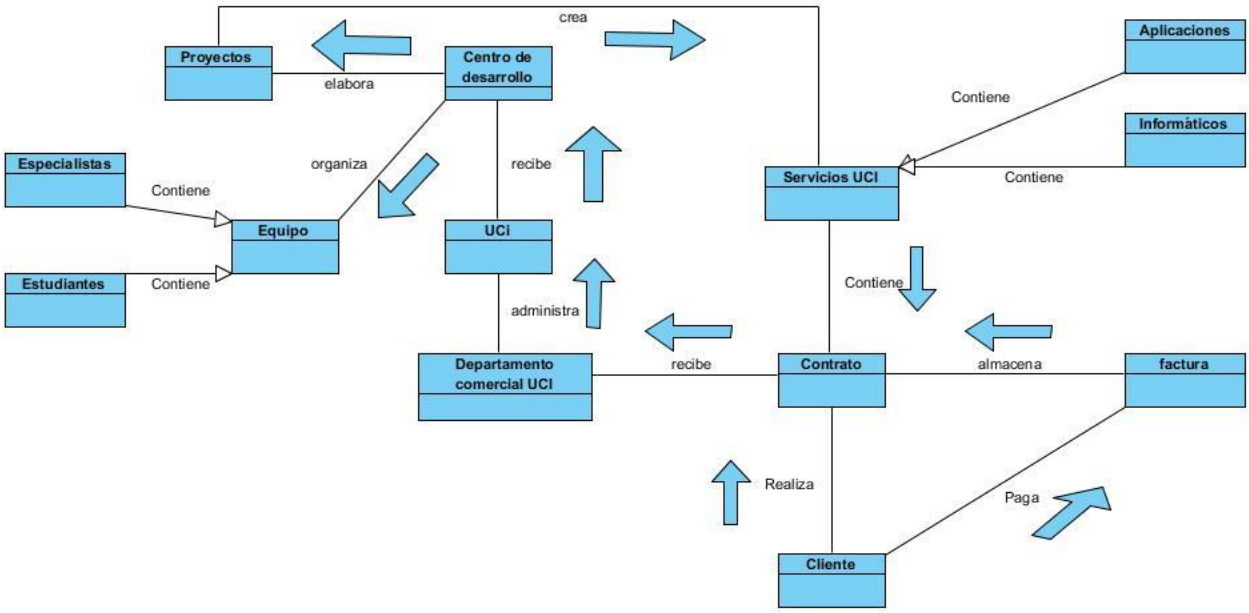

Figura 3 Modelo conceptual del funcionamiento actual del sistema de pagos.

2.3 Especificación de requisitos

Los requisitos para un sistema de software determinan lo que hará el sistema y definen las restricciones de su operación e implementación. Se pueden clasificar en: funcionales y no funcionales. A partir de la descripción de clases representadas en el Modelo de Dominio y las necesidades del cliente, fueron determinados los requisitos funcionales y no funcionales del sistema(García-Nieto, 2017).

2.3.1 Requisitos funcionales

Los Requisitos Funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionara entradas particulares y de cómo se debe comportar en situaciones particulares(García-Nieto, 2017).

Tabla 2 Descripción de Requisitos Funcionales.

No.	Nombre	Descripción	Prioridad	Complejidad
RF1.	Registrar producto	Permitir al administrador registrar algún producto.	Alta	Media
RF2.	Modificar productos	Permitir al administrador modificar los productos disponibles.	Media	Baja
RF3.	Eliminar productos	Permite al administrador eliminar los productos disponibles.	Media	Baja
RF3.	Mostrar productos	Permite al administrador mostrar los productos disponibles	Baja	Media
RF4.	Filtrar productos	Permitir al administrador filtrar los productos disponibles.	Media	Media
RF5.	Listar productos	Permitir al administrador listar los productos disponibles.	Alta	Media
RF6.	Modificar usuario	Permitir al usuario con los permisos suficientes actualizar la información de otro usuario o los datos del propio usuario.	Media	Media
RF7.	Eliminar usuario	Permitir al usuario con los permisos suficientes eliminar a otro usuario registrado en el sistema.	Media	Baja
RF8.	Mostrar usuario	Permitir al usuario con los permisos	Media	Media

		suficientes mostrar los usuarios registrados en el sistema		
RF9.	Listar usuario	Permitir al usuario con los permisos suficientes ver un listado de los usuarios registrados en el sistema.	Media	Media
RF10.	Registrar usuario	Permitir al usuario con los permisos suficientes registrar manualmente otro usuario.	Media	Media
RF11.	Autenticar usuario	Permite al usuario iniciar sesión en el sistema.	Alta	Baja
RF12.	Cerrar sesión de usuario	Permite al usuario cerrar sesión del sistema.	Alta	Baja
RF13.	Buscar usuario	Permitir al usuario con los permisos suficientes buscar otro usuario.	Baja	Baja
RF14.	Crear pago	Permitir al cliente crear un pago para algún producto	Alta	Media
RF15.	Modificar pago	Permitir al administrador actualizar el estado de algún pago.	Media	Media
RF16.	Eliminar pago	Permitir al administrador eliminar algún pago	Media	Media
RF17.	Mostrar pago	Permitir al administrador mostrar el pago efectuado.	Alta	Baja
RF18.	Listar pago	Permitir al administrador listar el pago.	Alta	Media
RF19.	Realizar pago	Permitir al cliente realizar el pago por el servicio correspondiente.	Alta	Media
RF20.	Cancelar pago	Permitir al cliente cancelar un pago antes de realizarlo	Media	Media
RF21.	Mostrar *wallet*	Permitir al cliente poder ver la *wallet* disponible para los depósitos.	Alta	Baja
RF22.	Modificar *wallet*	Permitir al administrador modificar la *wallet* de depósitos.	Baja	Alta
RF23.	Listar *wallet*	Permitir al administrador listar la *wallet* de depósitos.	Media	Media

RF24.	Crear Rol	Permitir al usuario con los permisos suficientes crear un rol en el sistema.	Alta	Baja
RF25.	Modificar Rol	Permitir al usuario con los permisos suficientes modificar un rol registrado antes en el sistema.	Media	Media
RF26.	Eliminar Rol	Permitir al usuario con los permisos suficientes eliminar un rol del sistema.	Media	Media
RF27.	Mostrar Rol	Permitir al usuario con los permisos suficientes ver la información perteneciente a un rol registrado en el sistema.	Alta	Baja
RF28.	Listar Rol	Permitir al usuario con los permisos suficientes ver el listado de roles registrados en el sistema.	Media	Media
RF29.	Registrar permiso	Permitir al usuario con los permisos suficientes registrar un permiso en el sistema.	Alta	Media
RF30.	Modificar permiso	Permitir al usuario con los permisos suficientes modificar los datos de un permiso registrado en el sistema.	Media	Media
RF31.	Eliminar permiso	Permitir al usuario con los permisos suficientes eliminar un permiso del sistema.	Media	Media
RF32.	Mostrar permiso	Permitir al usuario con los permisos suficientes ver los datos de un permiso en especificó registrado en el sistema.	Baja	Media
RF33.	Listar permiso	Permitir al usuario con los permisos suficientes ver el listado de los permisos registrados en el sistema.	Media	Media
RF34.	Asignar permiso	Permitir al usuario con los permisos suficientes asignar un permiso a un rol registrado en el sistema.	Alta	Media

RF35.	Cambiar contraseña	Permitir al usuario con los permisos suficientes cambiar la contraseña de otro usuario o el propio usuario cambiar su propia contraseña.	Alta	Media
RF36.	Depositar saldo	Permite al administrador o al cliente solicitar un depósito de *token*.	Alta	Media
RF37.	Acreditar saldo	Permite al administrador acreditar un saldo solicitado por algún cliente.	Alta	Media
RF38.	Generar reporte	Permite al administrador generar un reporte luego de verificar un pago.	Alta	Alta
RF39.	Mostrar producto para cliente.	Permite al cliente mostrarle los productos ya adquiridos.	Alta	Baja

2.3.2 Requisitos no funcionales

Los Requisitos No Funcionales son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Se caracterizan por no estar vinculados de forma directa con las funciones del sistema, sino a las propiedades de éste(García-Nieto, 2017).

Tabla 3 Descripción de Requisitos no Funcionales.

No.	Descripción.
RNF1	Seguridad
RnF1.1	Se garantizará la integridad de la información mediante mecanismos de control de acceso al usar usuarios, contraseñas y niveles de accesos para cada usuario, de manera que cada uno pueda tener disponible solamente las opciones que se encuentran en correspondencia con su actividad.
RnF1.2	Se asignarán los permisos de acceso, escritura, lectura en dependencia del rol que desempeñe cada usuario del sistema.
RnF1.3	Existirá un solo administrador del sistema.
RnF1.4	El sistema debe permitir recuperar la contraseña a los usuarios.
RnF1.5	La contraseña del usuario debe tener un mínimo de 8 caracteres en combinación de letras, números y caracteres especiales.
RnF1.6	La *wallet* brindada por el administrador va a estar certificada y no se podrá

	modificar.
RnF2	Portabilidad
RnF2.1	Se podrá acceder al portal web desde cualquier dispositivo al emplear cualquier navegador.
RnF2.2	Su interfaz debe ser adaptable a cualquier dispositivo, ya sea una computadora, tableta o teléfono celular.
RnF3	Confiabilidad
RnF3.1	El acceso al sistema será controlado a través de la autenticación por nombres de usuario y contraseñas internos del sistema.
RnF3.2	En caso de que ocurran fallas, los errores deben mostrarse sin detalles de información que puedan comprometer la integridad y seguridad del mismo. Sólo se mostrarán detalles ampliados del error a usuarios con privilegios de administración.
RnF3.3	Tendrá implementado mecanismos de tratamientos de errores.
RnF4	Soporte
RnF4.1	El sistema estará bien documentado de forma tal que el tiempo de mantenimiento sea mínimo en caso de necesitarse.
RnF5	Usabilidad
RnF5.1	El sistema debe presentar una interfaz intuitiva para el usuario.
RnF5.2	El sistema tiene un diseño *responsive* que permitirá que se adapte a cualquier dispositivo.
RnF5.3	Debe tener una opción de ayuda sobre las principales funcionalidades que brinda el sistema y sus iconos respectivos, para un mejor entendimiento.
RnF6	Rendimiento
RnF6.1	El proceso de ejecución de la prueba tiene prioridad en cuanto a la asignación de recursos.
RnF6.2	Cada transición debe demorar, como promedio, cinco segundos.
RnF6.3	El sistema debe permitir que hasta 1000 usuarios interactúen con él de manera concurrente.
RnF7	Software
RnF7.1	El servidor debe tener como requerimientos mínimos de software sistemas operativos Linux o Windows 7 o superior.

RnF7.2	El servidor debe tener como requerimientos mínimos de software PostgreSQL8.3.9 como Sistema Gestor de Base de Datos.
RnF7.3	El sistema podrá ser visualizado en dispositivos desde las resoluciones 320x480, 768x1024, 1024x980 y 1325x980.
RnF8	Hardware
RnF8.1	El servidor de base de datos debe poseer una capacidad mínima de 20 GB.
RnF8.2	El servidor de aplicaciones web debe poseer una capacidad mínima de 80 GB.
RnF8.3	El servidor de aplicaciones web y de base de datos deben poseer como mínimo un CPU Dual Core a 1.30 GHz.
RnF8.4	Los servidores web y de base de datos deben poseer como mínimo 2 GB de memoria RAM.

2.4 Diagrama de casos de uso del sistema

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Los casos de uso permiten mostrar el entorno (actores) y el alcance (requisitos funcionales expresados como casos de uso) de un sistema. Un caso de uso describe la secuencia de interacciones que se producen entre el sistema y los actores del mismo para realizar una determinada función. Los actores son elementos externos (personas, otros sistemas, etc.) que interactúan con el sistema como si de una caja negra se tratase. Un actor puede participar en varios casos de uso y un caso de uso puede interactuar con varios actores (Pressman, 2010).

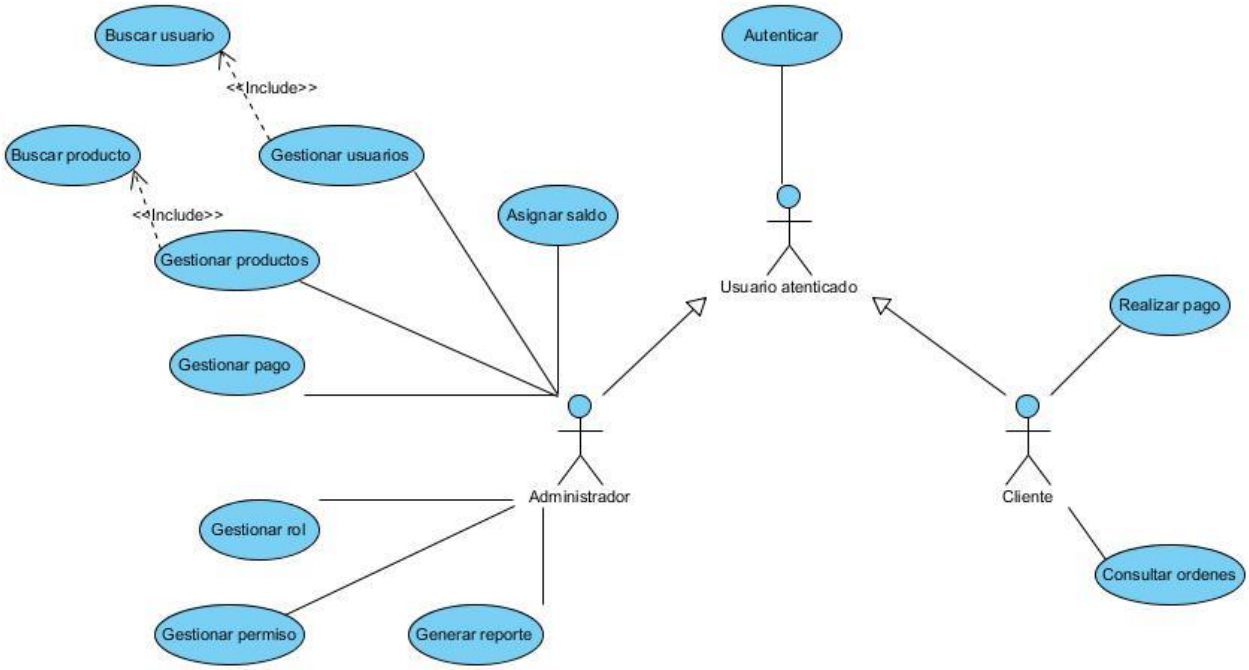

Figura 4 Diagrama de casos de uso del sistema de gestión de pagos.

2.5 Arquitectura

Para seleccionar la arquitectura a utilizar por el sistema se tuvo en consideración que la misma sea la apropiada para las aplicaciones web, así como que sea adaptable a las tecnologías proporcionadas por nuestro lenguaje de programación. Se consideró que el estilo arquitectónico que cumple con las condiciones planteadas es el cliente servidor hace uso del patrón modelo-vista-controlador. También se trató de lograr que todo el desarrollo esté soportado sobre software libre, para que la herramienta no herede ningún tipo de limitación en lo relativo a la soberanía tecnológica.

2.5.1 Patrón Modelo-Vista-Controlador

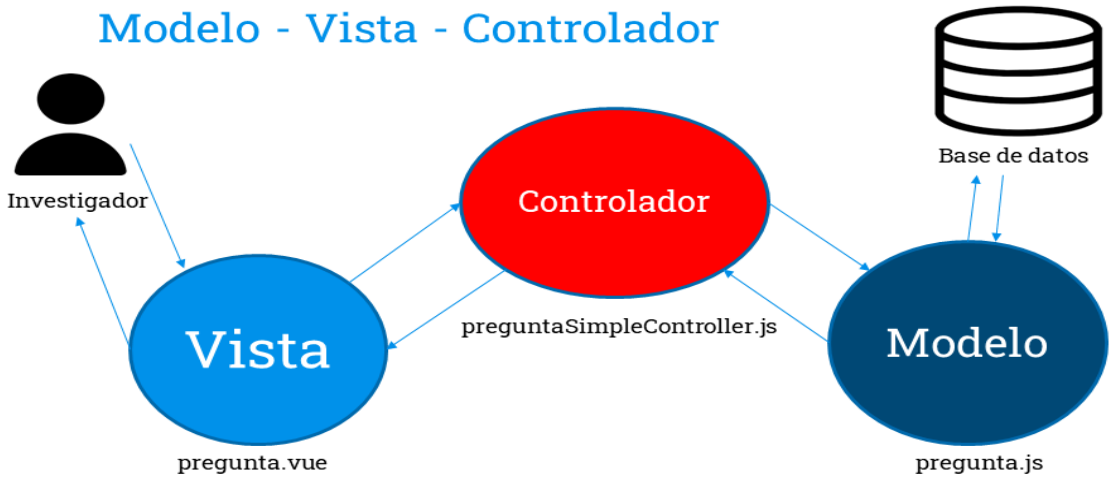

Figura 5 Modelo Vista Controlador.

El patrón de arquitectura de software MVC separa los datos de la aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos, al permitir mayor independencia, mantenimiento y reutilización (Coba y Fernando, 2016).

Este patrón arquitectónico está compuesto por tres componentes:

1. **Modelo:** Es la clase que representa las tablas de la base de datos para manejar los datos y controlar todas sus transformaciones. Además, se especifican las relaciones entre clases y las migraciones que expresan cambios en la base de datos.
2. **Vista:** Convierte el modelo en una página web que facilita al usuario interactuar con ella. Contiene los elementos asociados a la presentación de los datos y la información visual de los elementos que conforman el reporte, encontrándose el código asociado a las interfaces, que se encargan de visualizar la información que el controlador devuelve.
3. **Controlador:** Es el encargado de procesar las interacciones del usuario y ejecuta los cambios adecuados en el modelo o en la vista. La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista), lo que permite un mantenimiento más sencillo de las aplicaciones. El controlador es el encargado de aislar al modelo y a la vista de los detalles del protocolo usado para las peticiones (HTTP, consola de comandos, email).

Con el empleo de la arquitectura MVC se separa la lógica del negocio y la presentación de la información, con ello se logra crear un software más ordenado y permite potenciar su mantenimiento y reutilización de código con mayor facilidad. Con este patrón se pueden realizar proyectos pequeños o de gran escala. Brinda la posibilidad de cambiar la vista de acuerdo al dispositivo que se utilice sin afectar el modelo creado originalmente, al utilizar el controlador como intermediario.

2.6 Patrones de diseño

Un patrón de diseño provee un esquema para refinar los subsistemas o componentes de un sistema de software, o las relaciones entre ellos. Describe la estructura comúnmente recurrente de los componentes en comunicación, que resuelve un problema general de diseño en un contexto particular. Tienden a ser independientes de los lenguajes y paradigmas de programación y su aplicación no afecta necesariamente al sistema completo, pero si a un subsistema o parte del mismo (Ramírez, 2020). A continuación, se muestran los patrones de diseño empleados en la implementación de la propuesta de solución.

2.6.1 Patrón Grasp

Experto: es un patrón que se usa para asignar responsabilidades; es un principio básico que se utiliza en el diseño orientado a objetos. Se conserva el encapsulamiento, porque los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento (Visconti, M., & Astudillo, H., 2012). El patrón se evidencia en el código por el impacto que tiene el usuario en la generación de tickets cuando se crea un depósito.

```
const Tickets = db.define('tickets',{
    id_user:{
        type: DataTypes.NUMBER
    },
    id_product:{
        type: DataTypes.STRING
    },
},{
```

```
const Deposits = db.define('deposits',{
    total:{
        type: DataTypes.STRING
    },
    img:{
        type: DataTypes.STRING
    },
    state:{
        type: DataTypes.STRING
    },
    id_user:{
        type: DataTypes.NUMBER
    },
},{
```

Figura 6 Patrón experto usado en el código.

Creador: guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento (Visconti, M., & Astudillo, H., 2012). Se evidencia en el código en el controlador donde se crea un depósito.

```
            await Deposits.create({
                total: parseFloat(total),
                img: imgsrc,
                id_user: decode.userId,
                state: 'pending',
            });
            const user = await Users.findByPk(decode.userId);
            await user.update({
                pending: true,
            });
            res.json({message: "Depósito Agregado Correctamente", success: true});
```

Figura 7 Patrón creador usado en el código.

Controlador: Un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación. El mismo asigna la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase que

represente el “sistema” global (Visconti, M., & Astudillo, H., 2012). En el sistema se evidencia en todos los controladores, al tomar de muestra el controlador products.js como se muestra en la figura.

```
export const setProduct = async(req, res) => {
    try {
        const authHeader = req.headers['authorization'];
        const token = authHeader && authHeader.split(' ')[1];
        if(token == null) return res.sendStatus(401);
        const rol = jwt.decode(token).rol;
        if(rol === 'admin'){
            console.log(req.body.id);
            const product = await Products.findByPk(req.body.id);
            console.log(product);
            await product.update({
                name: req.body.name,
                description: req.body.description,
                price: req.body.price,
                id_type: req.body.id_type,
            });
            res.json({success: true, message: 'Producto cambiado satisfactoriamente'})
        }
        else{
            res.json({success: false, message: 'Invalid action'});
        }
    } catch (error) {
        console.log(error);
    }
}
```

Figura 8 Patrón controlador usado en el código.

Bajo acoplamiento: Es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras clases (Visconti, M., & Astudillo, H., 2012). Se evidencia en la relación existente entre orden y servicio.

Alta Cohesión: la información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con la clase y al objetivo para lo cual fue creada. Con la Cohesión Lógica el módulo realiza múltiples tareas relacionadas, pero, en tiempo de ejecución, sólo una de ellas será llevada a cabo (Visconti, M., & Astudillo, H., 2012). Se evidencia en la relación existente entre los productos y el tipo de producto, los servicios y los tickets.

2.6.2 Patrones GOF

Estos patrones están definidos por *The Gang of Four* (GOF) en procesos de desarrollo de software orientados a la web. El grupo GOF se dedica al análisis de los problemas recurrentes en el desarrollo de software y realizan una clasificación y agrupación a partir de dos criterios, su propósito y alcance (Guerrero et al., 2012).

Singleton: El patrón Singleton garantiza que una clase sólo tenga una instancia y proporciona un punto de acceso global a esta instancia (Pérez, 2017). Se evidencia en la creación de la conexión para la base de datos.

```
1  import {Sequelize} from "sequelize";
2
3  const db = new Sequelize('auth_db','root','',{
4      host: "localhost",
5      dialect: "mysql"
6  });
7
8  export default db;
```

Figura 9 Patrón singleton usado en el código.

Mapeo objeto-relacional: El patrón Mapeo objeto-relacional (ORM) es una técnica de programación para convertir datos del sistema de tipos utilizado en un lenguaje de programación orientado a objetos utilizado en una base de datos relacional. En la práctica esto crea una base de datos virtual orientada a objetos sobre la base de datos relacional. Esto posibilita el uso de las características propias de la orientación a objetos (esencialmente la herencia y el polimorfismo) (Pérez, 2017). En el código se evidencia en la utilización del ORM sequelize para el mapeo de los objetos de la base de datos.

```
1  import { Sequelize } from "sequelize";
2  import db from "../config/Database.js";
```

Figura 10 Patrón ORM usado en el código.

Observer: El patrón Observer puede ser utilizado cuando hay objetos que dependen de otro, al necesitar ser notificados en caso de que se produzca algún cambio en él (Pérez, 2017). Se evidencia cuando se produce un depósito por la compra de un producto por parte del cliente, que se debe generar un *token* en base al cambio que se produce en la tabla depósito.

```
45  export const createTickets = async(req, res) => {
46      const { id_product } = req.body;
47      const authHeader = req.headers['authorization'];
48      const token = authHeader && authHeader.split(' ')[1];
49      if(token == null) return res.sendStatus(401);
50      const decode = jwt.decode(token);
51      const rol = decode.rol
52      try {
53          if(rol === 'admin' || rol === 'client'){
54              await Tickets.create({
55                  id_product: id_product,
56                  id_user: decode.userId,
57              });
58              const user = await Users.findByPk(decode.userId);
59              const product = await Products.findByPk(id_product);
60              console.log(user);
61              console.log(product);
62              await user.update({
63                  saldo: user.saldo - product.price,
64              });
65              res.json({message: "Compra realizada correctamente", success: true});
66          }
```

Figura 11 Patrón observer utilizado en el código.

2.7 Diagrama de clases

Un diagrama o modelo entidad-relación es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información, así como sus interrelaciones y propiedades.

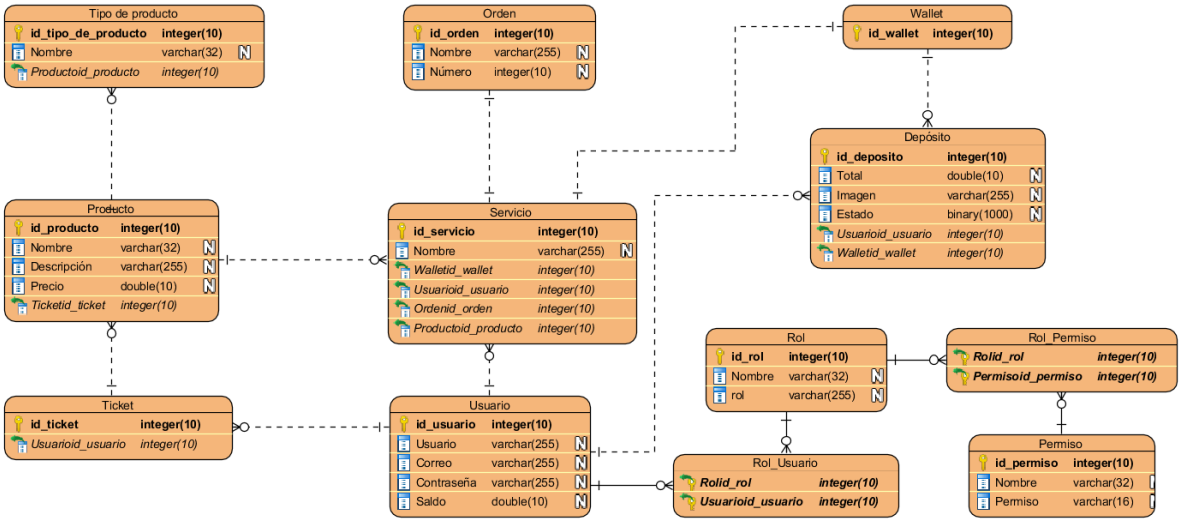

Figura 12 Modelo de Base de datos del sistema de gestión de pagos.

El modelo de datos de la propuesta de solución tiene 7 tablas y 9 relaciones entre ellas. Los usuarios podrán tener 2 roles, acceso a la autenticación y cambiar la contraseña de acceso. Los servicios que solicite el cliente en el portal podrán tener uno o varios productos, además de poder tener diferentes tipos de productos disponibles, todo se realiza a través de la compra, la cual llamamos Ticket. Los servicios luego de ser solicitados pasan a ser ordenes que constan del pago realizado a la *wallet* brindada, para esto se requiere una foto del comprobante de pago. Luego de este proceso pasan a ser una solicitud que puede ser aprobada o no por el administrador que se encarga de verificar que todo este correctamente.

2.8 Diagrama de despliegue

Un Diagrama de Despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esto muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos.

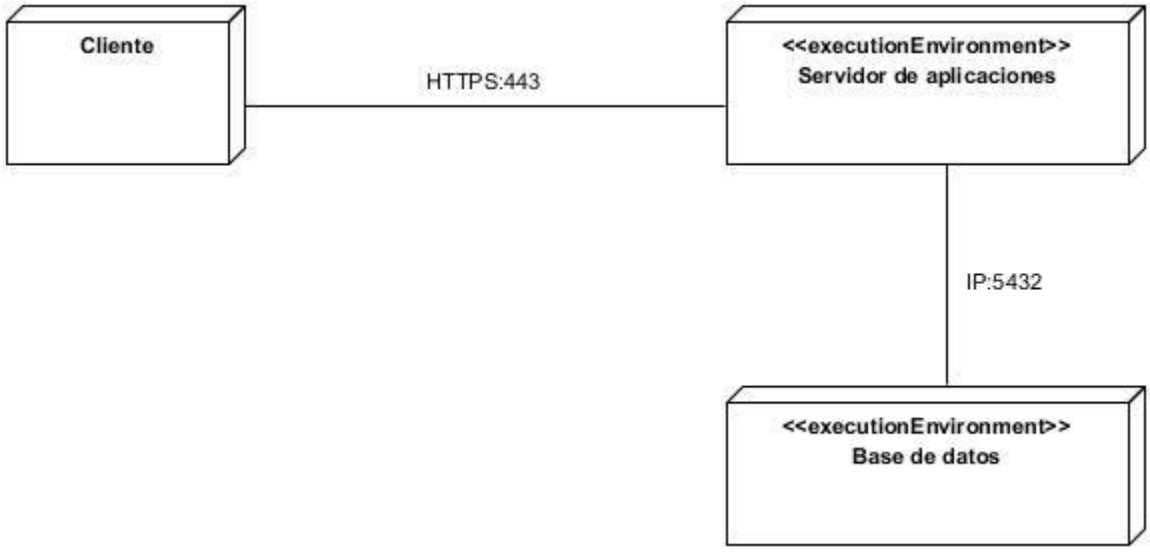

Figura 13 Diagrama de despliegue del sistema de gestión de pagos.

A continuación, se describen los elementos que componen el diagrama de despliegue para el módulo que se muestra anteriormente.

✓ **Servidor de aplicación:** Es la estación de trabajo que hospeda el código fuente de la aplicación, y que les brinda a los usuarios las interfaces de la misma para realizar los procesos definidos por cada uno de los roles del sistema. Esta estación se comunica con el servidor de base de datos donde se almacenan los datos de la aplicación al realizar la comunicación mediante el protocolo TCP/IP.

✓ **Servidor de base de datos:** Este servidor es el encargado del almacenamiento de los datos del sistema, y se comunica con el servidor de aplicaciones del sistema.

✓ **Dispositivo cliente:** Dispositivo con el que el cliente se comunica al servidor de aplicaciones.

Especificación de despliegue:

1. **Sistema Operativo:** Windows 10 Home 21H1 19043.1237
   - **Microprocesador:** Intel(R) Core (TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz
   - **Memoria RAM:** 8 GB
   - **Almacenamiento:** 250 GB SSD M.2
2. **Servidor Web:** Apache 2.14
3. **Servidor de Base de Datos:** PostgreSQL 13
4. **Motor:** Node >= 14.18.1
5. **Gestor de paquetes:** Yarn >= 1.22.17, NPM >= 8.1.3

Interfaz de Comando: @nestjs/cli >= 8.1.2, @Reactjs 18.1

2.9 Diagrama de Componentes

El diagrama de componentes muestra los componentes de un sistema de software conectados por las relaciones de dependencias lógicas entre cada uno de ellos. Provee una vista arquitectónica de alto nivel del sistema, al ayudar a los desarrolladores a visualizar el camino de la implementación. Cada componente representa una unidad del código (fuente, binario o ejecutable), que permite mostrar las dependencias en tiempo de compilación y ejecución. La realización del diagrama posibilita tomar decisiones respecto a las tareas de implementación y los requisitos (Larman, 2003).

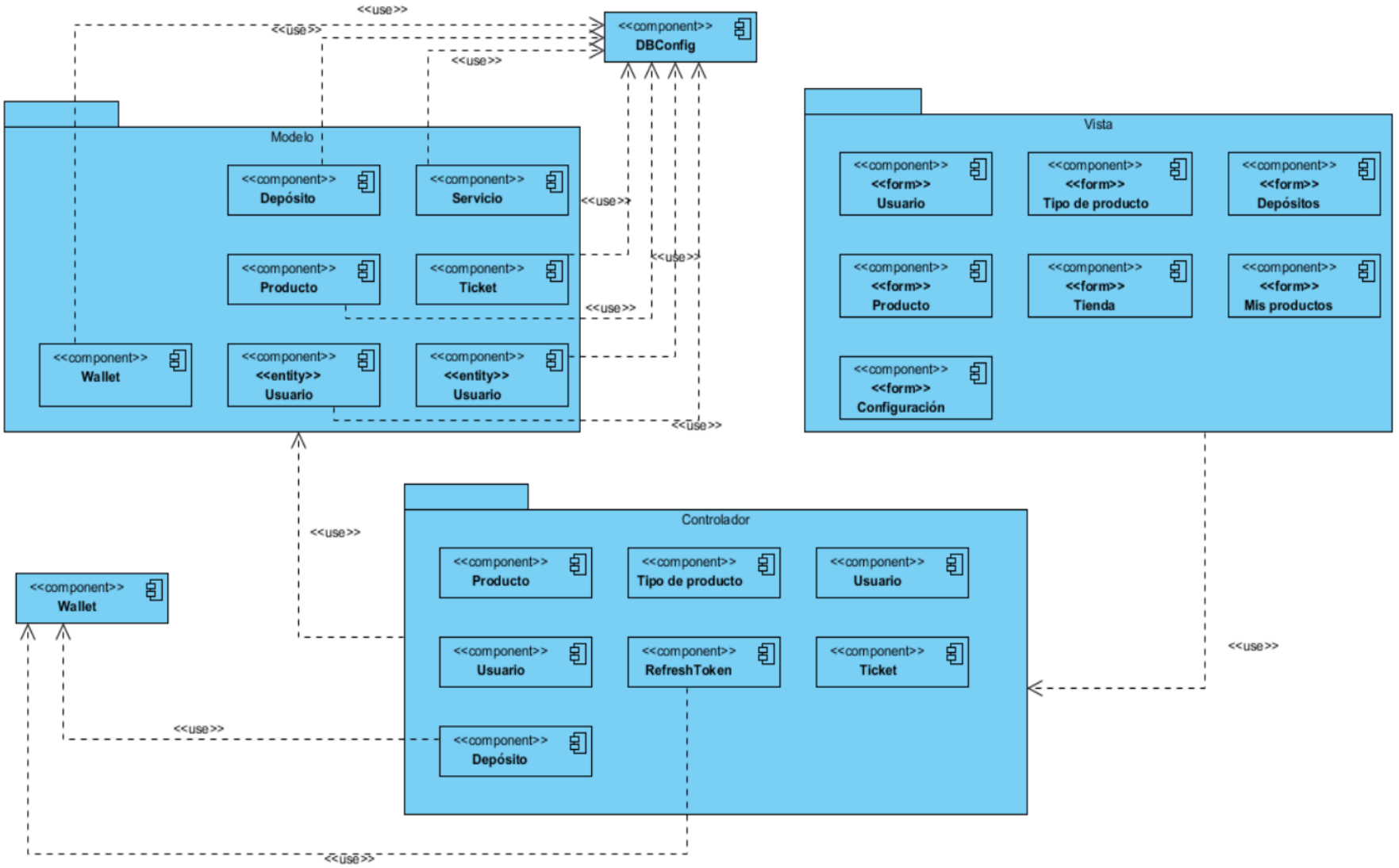

Figura 14 Diagrama de componentes del Sistema para la Tokenización del pago de los bienes y servicios.

El diagrama anterior refleja la relación existente entre los diferentes componentes que integra la página web desarrollada, donde las diferentes vistas hacen solicitudes a los controladores específicos a cada caso; el controlador depósito conecta directamente con el servicio de *wallet*, y *RefreshToken* genera un ID único para cada depósito. Los controladores requieren de todos los módulos correspondientes para las llamadas a la base de datos, logrando un correcto funcionamiento y una integración total del sistema.

Conclusiones del capítulo

Tras el desarrollo del presente capítulo se arriba a las siguientes conclusiones:

1. La obtención de los requisitos funcionales y no funcionales permitió identificar las funcionalidades que el sistema presentará, así como las restricciones en las que operará el sistema.
2. La definición del modelo de datos provee de un resumen de la estructura de datos general de la aplicación, así como la interrelación de las clases que lo componen.
3. La aplicación de los patrones GRASP y GOF permitió al autor definir la estructura interna del software obtenido, garantizando en todo momento el cumplimiento de buenas prácticas de desarrollo de software, la reutilización de código y la simplificación de fututas tareas de mantenimiento.

4. El diagrama de componentes mostró cómo interactúan los paquetes definidos con sus respectivos componentes.

Capítulo 3 Implementación y Pruebas del sistema para la Tokenización del pago de los bienes y servicios de la Universidad de las Ciencias Informáticas

En el desarrollo de este capítulo se cuantifica el nivel de fiabilidad de los resultados obtenidos y la calidad en el desarrollo de la propuesta de solución y desarrollo del *token* propuesto. Para lograr esta valoración se definen estrategias de pruebas de software en consonancia con los estándares actuales, donde se decide aplicar distintas disciplinas de pruebas internas y de funcionalidad que se definen en la metodología que se empleó como guía durante todo el desarrollo de la propuesta de solución, con la única finalidad de verificar la calidad del producto final antes de la entrega al cliente.

3.1 Propuesta de implementación del token UCI

Cointool es una herramienta de terceros que ha evolucionado de manera exponencial, ofrece actualmente más de 60 servicios destinados al desarrollo *blockchain* y creación de *token* de utilidad. Funciona como red de código abierto a nuevos desarrolladores y basado fundamentalmente en la red BNB *Smart Chain*, aunque tiene una amplia gama de redes para implementación. Para la creación del *token* UCI se usa las características descritas anteriormente en el capítulo 1, y los pasos a seguir son:

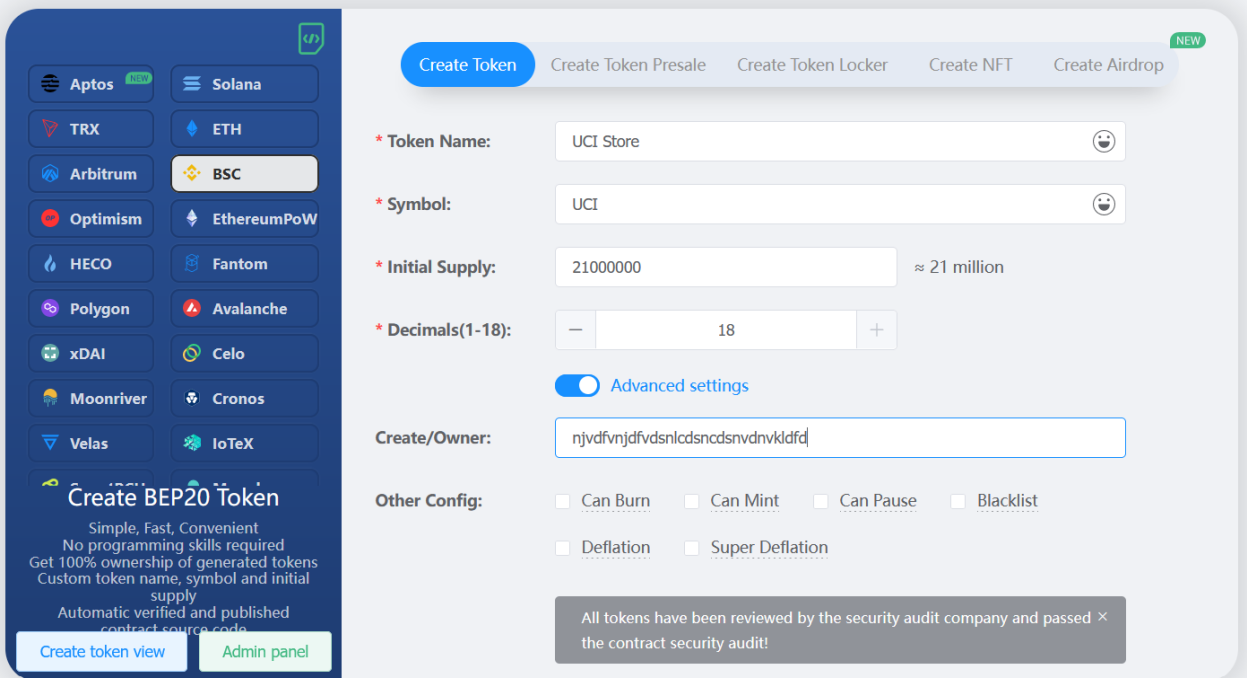

Figura 15 Interfaz de la configuración del token.

1-Seleccionar la red de contrato inteligente a implementar (BSC).

2-Rellenamos el campo Nombre, Símbolo (abreviatura del *token*), cantidad inicial (números de *token* puestos en preventa o disponibles), decimales (código encriptado para generar nuestra *wallet* en *exchange*) y las configuraciones adicionales dispuestas para darle un mejor rendimiento.

3- En el campo *owner* colocar la *wallet* donde se tendrá la piscina de liquidez, para atribuir un valor inicial y pagar posibles fee de transferencias.

3.2 Implementación de la página web

En este proceso se procede a la codificación de cada módulo, con el objetivo principal de desarrollar la arquitectura y el sistema como un todo, planificar qué subsistemas deben ser implementados y en qué orden deben ser integrados, notificar si se encuentran errores de diseño e integrar el sistema al seguir el plan (Fernández y Cadelli, 2014). En la figura no. 15 se muestra un fragmento de código de la implementación del sitio web del sistema de gestión de pagos elaborado en esta investigación:

```
export const addDeposit = async(req, res) => {
    const { total } = req.body;
    const authHeader = req.headers['authorization'];
    const token = authHeader && authHeader.split(' ')[1];
    if(token == null) return res.sendStatus(401);
    const decode = jwt.decode(token);
    const rol = decode.rol;
    console.log(decode);
    try {
        if(rol === 'admin' || rol === 'client'){
            const imgsrc = 'http://127.0.0.1:5000/uploads/images/' + req.file.filename
            await Deposits.create({
                total: parseFloat(total),
                img: imgsrc,
                id_user: decode.userId,
                state: 'pending',
            });
            const user = await Users.findByPk(decode.userId);
            await user.update({
                pending: true,
            });
            res.json({message: "Depósito Agregado Correctamente", success: true});
        }
        else{
            res.json({success: false, message: 'Invalid action'});
        }
    } catch (error) {
        res.json({message: "Error al depositar", success: false});
    }
```

Figura 16 Implementación del sistema de pago.

3.3 Estándares de codificación

Son un punto de referencia para que todos los programadores dentro de un equipo sigan una misma línea de implementación, y de esta forma mejora la comunicación entre ellos. Esto es viable dado que estas buenas prácticas posibilitan que el código sea claro y de esta manera

fácil de entender y mantener. Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. Cuando se desarrollan aplicaciones web es importante que el código sea fácil de leer y modificar, para esto se deben seguir una serie de normas comunes para todos los desarrolladores.

Indentación: Consiste en insertar cuatro espacios en determinadas líneas de código y no se debe dejar espacios en blanco al final de cada línea para facilitar su comprensión

Operadores: Los operadores binarios, utilizados entre dos valores, deben estar separados a ambos lados del operador por un espacio. Esto se aplica a operadores como +, -, *, /, =, ==, !=, >, <. (Concatenación de cadenas), .=, +=, -=, etc. Los operadores unarios como ++, -- no deben tener separación. Está presente en la figura no. 16 en la línea de código 7.

Uso de comillas: Pueden ser usadas tanto las comillas simples (‘cadena') como las comillas dobles ("cadena") para delimitar las cadenas de caracteres, si es necesario el uso de variables dentro de la cadena, se deben usar las comillas dobles Está presente en la figura no. 16 en la línea de código 1.

Funciones: Los nombres de las funciones se escribirán al usar el estilo camello (la primera palabra comenzará con minúscula y la letra inicial de las próximas palabras estará en mayúscula), con el objetivo de evitar duplicidad de funciones. Está presente en la figura no. 16 en la línea de código 8

Comentar el código: El uso de los comentarios pueden hacerse al utilizar las etiquetas /* */ para comentarios en varias líneas y // para comentarios de una única línea. Está presente en la figura no. 16 en la línea de código 5,

Estructuras de control: Las estructuras de control deben cumplir un conjunto de normas para su correcto funcionamiento; las estructuras (if, while, for, etc.) y el primer paréntesis deben tener un espacio intermedio, para no confundirlas con la nomenclatura de las funciones; la llave de apertura ({) estará en la primera línea separada por un espacio y es recomendado usar las dos llaves ({ }) aunque el código lo permita; y las estructuras else y else if serán escritas en la línea siguiente de la llave de cierre anterior (}). Está presente en la figura no. 16 en la línea de código 13.

En la figura no. 16 se presenta un fragmento de código que reúne todos los estándares de codificación antes mencionados.

```
import Deposits from "../models/DepositsModel.js";
import jwt from "jsonwebtoken";
import db from "../config/Database.js";
import Users from "../models/UserModel.js";
//Para obtener un depósito//
export const getDeposits = async(req, res) => {
    try {
        const authHeader = req.headers['authorization'];
        const token = authHeader && authHeader.split(' ')[1];
        if(token == null) return res.sendStatus(401);
        const rol = jwt.decode(token).rol;
        console.log(req.body);
        if(rol === 'admin') {
            const temp = await db.query(
            'SELECT t1.id, t1.total, t1.img, t1.state, t2.email, t1.id_user FROM deposits t1, users t2  WHERE t1.id_user = t2.id');
            res.json({success: true, deposits: temp[0]});
        }
        else{
            res.json({success: false, message: 'Invalid action'});
        }
    } catch (error) {
        console.log(error);
    }
}
```

Figura 17 Estándares de codificación utilizados.

3.4 Validación del sistema de gestión de pagos mediante el token UCI

Durante la etapa de implementación, se cometen errores y pueden pasarse por alto algunos elementos que son importantes para el correcto funcionamiento del sistema. Por tal motivo, es fundamental llevar a cabo la fase de validación, en la cual, a través de varios tipos y métodos de pruebas de software (estrategia de pruebas), se pretende comprobar el cumplimiento de las especificaciones del diseño y de la codificación, identificar los posibles errores cometidos y validar la solución propuesta en los capítulos anteriores. En este epígrafe se muestran los resultados de la estrategia de prueba diseñada para el Sistema de Gestión de pagos mediante el *token* UCI mediante una web (Pressman, 2013).

3.4.1 Estrategia de pruebas a utilizar

Una estrategia de pruebas, no es una serie de instrucciones que detallan el paso a paso de cómo se debe hacer las pruebas. La estrategia define los objetivos a alcanzar, a menudo con recursos limitados y en condiciones de incertidumbre, donde se construye un plan de alto nivel para lograr los objetivos específicos de la prueba. La estrategia que se adopte dependerá de principalmente del proceso de desarrollo que se utiliza al interior de una organización TI y de las capacidades existentes del equipo de prueba. (Lee, 2021)

Pruebas de software a implementar:

- **Pruebas funcionales:** es una prueba de tipo caja negra basada en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software.

- **Pruebas de rendimiento:** Esta prueba es llevada a cabo por un grupo de pruebas independientes para asegurar que el rendimiento del sistema está dentro de los parámetros definidos. Se puede requerir herramientas para simular clientes y cargas pesadas, y se realizan las mediciones precisas de desempeño.
- **Pruebas de seguridad:** es un conjunto de pruebas objetivas con el fin de detectar las vulnerabilidades de un sistema, al tener muy claro que ningún sistema es 100% seguro o inviolable.
- **Prueba de integración:** El objetivo de las pruebas de integración es verificar el correcto ensamblaje entre los distintos componentes una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces, tanto internas como externas, cubren la funcionalidad establecida y se ajustan a los requisitos no funcionales especificados en las verificaciones correspondientes.

3.4.2 Pruebas funcionales

Este tipo de pruebas se realiza para validar que las funcionalidades implementadas funcionen de acuerdo a las descripciones de los requisitos especificados previamente. Para la realización de estas pruebas, se emplean dos métodos: el método de Caja Blanca y el método de Caja Negra. El primero está encaminado a pruebas al código de las aplicaciones; mientras que el segundo, a través del estudio de los datos de entrada y de salida, permite encauzar la atención en el funcionamiento de la interfaz del sistema. Se denomina caja negra a aquel elemento que es estudiado desde el punto de vista de las entradas que recibe y las salidas o respuestas que produce. Las pruebas de caja negra se llevan a cabo sobre la interfaz del software, para obviar el comportamiento interno y la estructura del programa (García, 2013). La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca. Más bien se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores (Pressman, 2010).

Los casos de prueba de la caja negra pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma correcta.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

A continuación, en la tabla no.5, se muestra el caso de prueba resultante en las iteraciones realizadas al siguiente caso de uso.

. Tabla 4: Caso de prueba para crear producto.

Caso de prueba: Crear producto	
Código de caso de prueba	**Nombre de requisito:** Crear producto
Nombre de la persona que realiza la prueba: Alejandro Lázaro Medero Lacosta	
Descripción de la prueba: Prueba a la funcionalidad crear producto	
Entrada/Pasos de ejecución: Se seleccionan los siguientes datos para crear un producto	
Nombre del producto Tipo de producto Descripción Precio El administrador selecciona los datos para gestionar un producto, y si los datos están correctos se almacena en el sistema; no deben existir datos incorrectos porque los campos tienen los atributos y excepciones bien definidas.	
Evaluación de la prueba: Satisfactoria	

3.4.2.1 Resultado de las pruebas funcionales

Para validar que el sistema cumpla con las funciones específicas para las cuales ha sido creado en la primera iteración se encontraron 20 no conformidades y se resolvieron 7. En la segunda iteración se quedan pendiente 13 no conformidades, se resuelven las 13. En la tercera iteración no se encontró inconformidades restantes En la imagen 17 se muestran los resultados obtenidos en las iteraciones de pruebas realizadas al Sistema de Gestión de pagos de la UCI a través de un *token*.

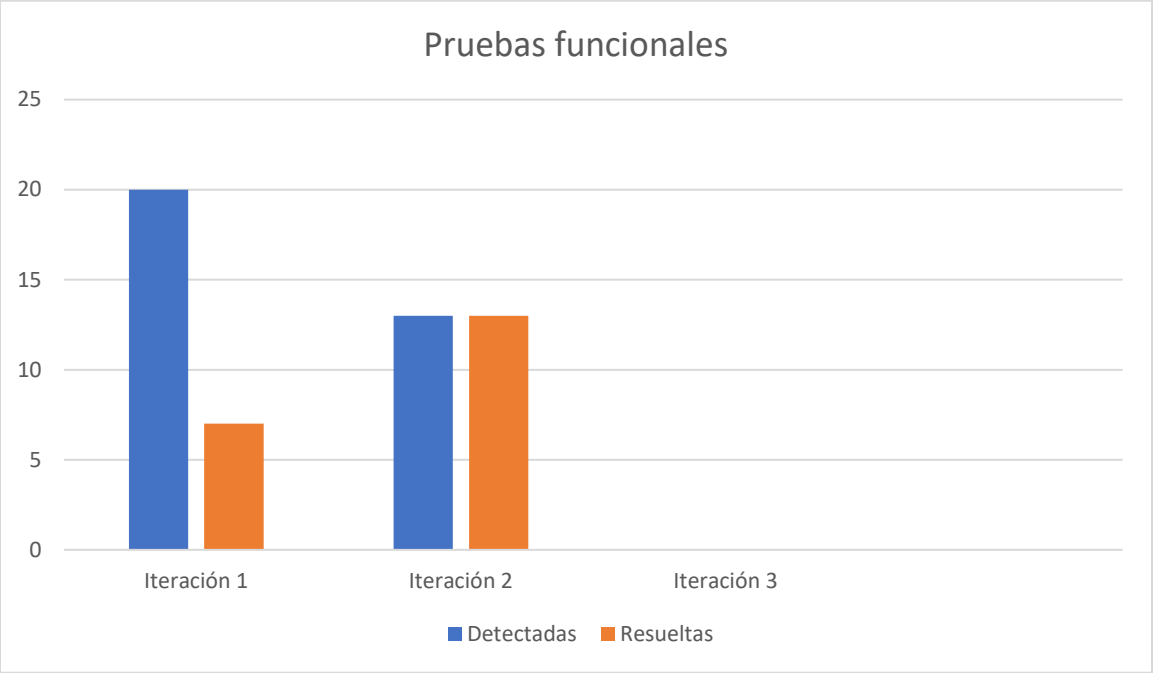

Figura 18 Resultados de las pruebas funcionales del sistema para la Tokenización del pago de los bienes y servicios.

Entre las no conformidades detectadas en el proceso de pruebas funcionales se encuentran:

- Errores de estructuración de los contenidos mostrados en las vistas.
- Los mensajes presentan problemas de idioma.
- Opciones que no funcionan.
- Validaciones que no admite el sistema.
- Campos en blanco.

3.4.3 Pruebas de rendimiento

La prueba de rendimiento se diseña para poner a prueba el rendimiento del software en tiempo de ejecución, dentro del contexto de un sistema integrado. Esta prueba ocurre a lo largo de todos los pasos del proceso de prueba. Incluso en el nivel de unidad, puede accederse al rendimiento de un módulo individual conforme se realizan las pruebas. Sin embargo, no es sino hasta que todos los elementos del sistema están plenamente integrados cuando puede determinarse el verdadero rendimiento de un sistema (Pressman, 2010). Los tipos de pruebas de rendimiento que se le realizaron al portal web fueron las pruebas de carga y estrés, y se utilizó como herramienta Apache JMeter, diseñado para pruebas de carga de comportamientos funcionales y la medición del rendimiento.

3.4.3.1 Pruebas de carga

Las pruebas de carga permiten la simulación del acceso de muchos usuarios a un servidor al mismo tiempo, al posibilitar observar el comportamiento de una aplicación bajo una cantidad de peticiones esperadas. La carga puede ser el número de usuarios concurrentes que se espera que utilicen la aplicación, y que realizan durante el tiempo en que dura la carga un número específico de transacciones. Este tipo de prueba facilita la monitorización del servidor y la base de datos para obtener el cuello de botella en la aplicación, y puede mostrar los tiempos de respuesta de todas las transacciones importantes.

3.4.3.2 Pruebas de estrés

Las pruebas de estrés posibilitan la obtención de datos sobre la carga del sistema. Tiene como objetivo generar cargas en el sistema hasta hacerlo inutilizable, para centrarse en verificar la calidad de los mensajes de error o establecer alertas para anticipar los fallos. Las pruebas de estrés son uno de los últimos tipos de pruebas que se deben efectuar, debido a que tienen un carácter poco realista porque podría darse el caso que nunca se diera en la vida real la situación de carga simulada.

3.4.3.3 Análisis de los resultados de las pruebas de rendimiento

Para realizar las pruebas de carga y estrés se utilizaron 2 PC, una como cliente y la otra como servidor, al crear así un entorno de trabajo en el cual se efectúo un análisis del tráfico de usuarios en el sistema con ayuda de la herramienta Apache JMeter. Debido a los datos obtenidos, se realizaron las pruebas de carga y estrés en el caso crítico en que se conecten 100 usuarios concurrentes. Las pruebas de carga y estrés se desarrollaron, en un ambiente al utilizar dos ordenadores con las siguientes características:

PC servidor

- Sistema operativo Windows 11.
- Microprocesador Intel Core i5 a 2.40 GHz.
- Memoria RAM 12GB.

PC cliente

- Sistema operativo Windows 10.
- Microprocesador Intel Core i7 a 2.60 GHz.
- Memoria RAM 16GB.

Ambas PC

• Características de la red: 100.0 Mbps

• Para poder acceder a esta pantalla debe correr el servidor web de la aplicación y el servidor de base de datos.

Los resultados obtenidos en las pruebas de carga se consideran satisfactorios. La propuesta de solución generó una buena transferencia de datos para diferente cantidad de usuarios concurrentes esperados, lo que incurrió en un rendimiento medio de 9,0 segundos. Se demuestra que la propuesta de solución es estable, porque se mantuvo en función los servicios todo el tiempo sin incurrir en fallos. Los resultados se muestran detallados en la tabla no.6.

Tabla 5 Pruebas de rendimiento sistema para el sistema de pago de los bienes y servicios.

Etiquetas	No. muestras	Media	Mediana	Línea 90%	Min	Máx.	%Error	Rendimiento (peticiones/ segundos)	Kb/s Recibidos
Conjunto1	120	350	365	560	12	1021	0.00%	28.0/min	2.1
Conjunto 2	300	400	380	700	14	1422	0.00%	26.0/min	2.0
Conjunto 3	420	410	375	620	43	1385	0.00%	29.0/min	1.8
Conjunto 4	670	326	300	635	50	1754	0.00%	21.0/min	2.1
Conjunto 5	700	620	630	1015	27	1796	0.00%	30.0/min	1.5
Conjunto 6	1000	515	500	814	33	2014	0.00%	25.0/min	1.4

A continuación, se describen las variables que miden el resultado de las pruebas de carga y estrés realizadas al sistema:

Muestra: Cantidad de peticiones realizadas para cada URL.

Media: Tiempo promedio en milisegundos en el que se obtienen los resultados.

Mediana: Tiempo en milisegundos en el que se obtuvo el resultado que ocupa la posición central.

Línea 90 %: Máximo tiempo utilizado por el 90 % de la muestra, al resto de la misma le llevo más tiempo.

Min: Tiempo mínimo que demora un hilo en acceder a una página.

Max: Tiempo máximo que demora un hilo en acceder a una página.

% Error: Por ciento de error de las páginas que no se llegaron a cargar de manera satisfactoria.

Rendimiento (peticiones/segundos): El rendimiento se mide en cantidad de solicitudes por segundo.

Kb/s recibidos: Velocidad de carga de las páginas.

Como se muestra en la siguiente tabla, se simularon las peticiones realizadas al sistema por un total de 100 usuarios simultáneamente en cada caso, para 200 muestras los cuales realizan hasta 5 peticiones por segundo. Se obtuvieron los siguientes resultados:

Tabla 6 Resultados de las pruebas de rendimiento.

Usuarios	No. Muestra	Media	Mediana	Linea 90%	Min	Máx	% Error	Rendimiento (peticiones/ segundos)	Kb/s recibidos
100	600	700	620	1240	10	2101	0.00%	3.1	15.5

Las pruebas realizadas muestran que el sistema es capaz de responder a 600 peticiones de 100 usuarios conectados simultáneamente en un tiempo promedio de 3.1 segundos con 0.00 % de error, de acuerdo a los resultados arrojados y a las prestaciones del hardware donde se realizaron las pruebas se considera que constituye un resultado satisfactorio.

3.4.4 Pruebas de seguridad

Según (Pressman, 2010) las pruebas de seguridad intentan verificar que los mecanismos de protección incorporados en el sistema lo protegerán de accesos inapropiados. Durante las pruebas de seguridad, el responsable de la prueba desempeña el papel de un individuo que desea entrar en el sistema. Debe intentar conseguir las claves de acceso por cualquier medio, debe bloquear el sistema, se niega así el servicio a otras personas.

3.4.4.1 Resultados de las pruebas de seguridad

Con el objetivo de evaluar la seguridad de la solución propuesta se emplea la herramienta Acunetix WVS la cual arrojó los siguientes resultados luego de realizada una primera iteración.

Tabla 7 Resultados de las pruebas de seguridad.

Categorías de vulnerabilidades	Cantidad de errores
Formularios HTML sin protección CSRF	3
Credenciales de usuarios enviadas en texto plano	2
Campos de contraseña con auto completamiento activado	2
Campos de usuario y contraseña mostrados	1
Vínculos rotos	0
Total	8

Después de analizar los resultados obtenidos en las pruebas se procedió a corregir las deficiencias encontradas. Para ello se llevaron a cabo un conjunto de acciones que permitió reforzar la seguridad del portal web. Los formularios HTML sin protección CSRF (falsificación de petición en sitios cruzados, en español) es una clase de ataque que afecta a las aplicaciones basadas en web. El ataque funciona mediante la inclusión de un enlace o secuencia de comandos en una página que tiene acceso a un sitio al que se conoce el usuario (o se supone) que se han autenticado.

Esta vulnerabilidad también es conocida por otros nombres como sección de manejo y ataque de un click. Para darle solución a esta alerta se utilizaron unos módulos llamados securelogin y seckit que se encargan de proveerle seguridad al portal y a su vez a los formularios. Campos de contraseña con auto completamiento activado, se introduce un nuevo nombre y contraseña en un formulario y se envía el formulario, el navegador le pregunta si la contraseña debe ser guardada. Se muestra el formulario, el nombre y la contraseña se rellenan de forma automática o se completan como se introduce el nombre. Un atacante con acceso local podría obtener la contraseña de texto plano de la caché del navegador. Para darle solución, la función de la contraseña de autocompletar debe ser desactivada.

3.4.5 Pruebas de integración

Las pruebas de integración constituyen el mecanismo para comprobar el correcto ensamblaje del sistema completo. Estas pruebas se realizan al terminar las pruebas unitarias con el objetivo de verificar que los módulos o componentes que conforman un sistema funcionan correctamente una vez que han sido integrados. Para la realización de las mismas debe tenerse en consideración un conjunto de aspectos tales como, el grado de complejidad de los módulos o componentes, su importancia funcional con respecto a las especificaciones del

sistema, la interrelación entre los mismos y las estadísticas de error en las pruebas unitarias de forma tal que se prueben primeramente los más críticos. Para ejecutar las mismas se seleccionan los casos de pruebas que serán diseñados en tener en consideración el grado de dependencia entre los componentes para realizar una funcionalidad específica (Pressman, 2010).

Tabla 8 Resultados de las pruebas de integración.

Caso de prueba: RF37. Acreditar saldo.			
Condiciones de ejecución: El usuario debe estar registrado en el sistema.			
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC Comprobar que se acredita saldo de forma correcta.	El administrador comprueba el depósito realizado por algún cliente para asegurar la sostenibilidad y correcto funcionamiento del sistema gestor.	Muestra al cliente en el menú superior izquierdo la actualización de su saldo	1. El administrador accede a la ventana de depósitos y comprueba que haya llegado satisfactoriamente el pago a la *wallet* suministrada. 2. El administrador le cambia el estado al cliente de ¨pendiente¨ a ¨ok¨ si todo está correctamente estipulado. 5. El cliente recibirá una notificación de actualización de su cuenta en el menú superior derecho. Ahora puede comprar algún producto de los que se ofrece.

3.5 Validación de la hipótesis

Para la validación de la hipótesis científica se utiliza el método de consulta a expertos en su variante Delphi (Sánchez, 2015) siguiendo los puntos siguientes:

- Identificación de los posibles expertos.
- Selección de los expertos.
- Realización de consultas a expertos, procesamiento y valoración de la información obtenida.

Para la identificación de los posibles expertos, se tiene en consideración: experiencia laboral, disposición de participar en la encuesta, competencia, relación con este tipo de software y conocimiento sobre tecnologías blockchain. Estas características permitirán que el experto se involucre más con la propuesta de solución y facilite su despliegue. En la Tabla 9 se muestran la selección de expertos:

Tabla 9 Expertos utilizados en la validación de la propuesta de solución

No.	Experto	Entidad	Años de Experiencia
1	Yohan González Almaguer	CIDI	2
2	Marylaura Martell León	CIDI	1
3	Christian Dubier López Nuñez	CESOL	1
4	Dayanis Herranz Rodríguez	XETID	1

Luego de la selección, se sometió a consideración un instrumento para validar el sistema para la Tokenización del pago de los bienes y servicios. El instrumento se compone de 5 sentencias relacionadas con el funcionamiento del sistema gestor. Los expertos para expresar su opinión o valoración pueden utilizar las siguientes categorías:

- Muy adecuado (MA).
- Bastante adecuado (BA).
- Adecuado (A).
- Poco adecuado (PA).
- Inadecuado (I).

Tabla 10 Sentencias a evaluar por los expertos para validar la hipótesis científica

No.	Sentencias plasmadas en la consulta realizada a expertos.
1	¿Se agrega correctamente el producto con la descripción y el precio?
2	¿Se autentica bien el usuario con su contraseña?
3	¿Llega correctamente la foto del comprobante enviada por el cliente?
4	¿Se agrega correctamente la *wallet* de depósito?
5	Considera que el sistema de gestión mejora el proceso de pagos en la Universidad.

Se calcula el coeficiente de Kendall que permite el análisis de la concordancia en las valoraciones realizadas por los expertos (Sampieri, 2014). El coeficiente de concordancia (W) será un índice de la divergencia del acuerdo efectivo por parte de los expertos y se calcula con la siguiente ecuación:

$$W = 12S/K2\ (N3-N)$$

Donde S representa el cuadrado de las desviaciones medias, K el número de expertos y N el número total de aspectos a evaluar, el valor de W oscila en el rango de 0 a 1. El valor 1 significa una concordancia de acuerdos total y el 0, un desacuerdo total. Se aplica además la Prueba de Significación de Hipótesis para comprobar el grado de consideración de Kendall, planteándose la hipótesis nula y la alternativa de la siguiente forma:

- H0: no existe concordancia entre los expertos.
- H1: existe concordancia entre los expertos.

$$X2 = K\ (N-1)\ W$$

X2=0.352

El resultado de X2 es comparado con el valor tabulado en la tabla de la distribución X2. Para tener un 95% de confianza se utilizará α = 0.05. Si se cumple que el resultado de X2 es menor que X2(α, N-1) se obtiene que 0.352 < 9.4877 entonces se valida la hipótesis alternativa H1, de que existe concordancia entre los expertos. Los criterios aportados por los expertos son sometidos a una prueba estadística no paramétrica, que permite concluir la valoración final que tiene cada uno de estos aspectos a evaluar. Para los datos anteriores se debe confeccionar una distribución de frecuencia a partir de los datos primarios para cada uno de los aspectos sometidos a consulta (Castro, 2014).

Tabla 11 Distribución de frecuencia para los datos primarios obtenidos

Categorías evaluativas	Frecuencia absoluta	Frecuencia relativa
Muy adecuado	28	0.875
Bastante adecuado	2	0.0625
Adecuado	2	0.0625
Poco adecuado	0	0
Inadecuado	0	0

Los resultados obtenidos en la validación se muestran en la Figura 19:

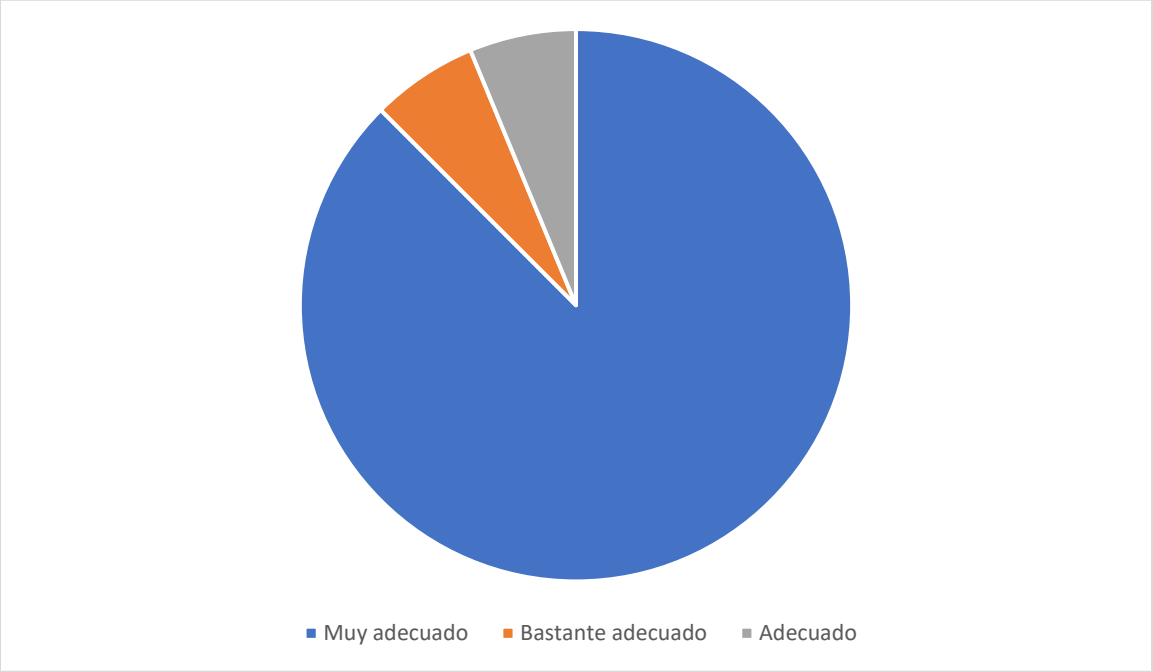

Figura 19 Comportamiento de la valoración de expertos por categorías evaluadas

Los datos de la Figura 19 muestran que un 87.5% de los aspectos fueron evaluados de muy adecuado, el 6.25% de bastante adecuado y un 6.25% de adecuado. El análisis de los resultados permitió identificar la existencia de una coherencia en las valoraciones realizadas. Los indicadores fueron evaluados satisfactoriamente y de esta forma se demuestra la validez de la hipótesis científica de la investigación, evidenciándose la mejora del procesamiento de la información.

Conclusiones del capítulo

Tras el desarrollo del presente capítulo se llega a las siguientes conclusiones:

1- La creación del *token* y su configuración permitió sentar las bases del correcto funcionamiento del sistema propuesto.

2- La definición de los estándares de codificación en la implementación del sistema, permitió que el código generado posea alta calidad, menos errores y que pueda ser mantenido fácilmente.

3- La realización de las pruebas de software al sistema facilitó la identificación de errores, su rápida solución, para determinar y asegurar la calidad del sistema de gestión de pagos a través de *tokens*.

Conclusiones

Con la culminación de la presente investigación se obtuvo como resultado, la implementación del Sistema para la gestión de pagos digitales a través de un *token*. A continuación, se exponen las conclusiones generales:

- La definición de un marco teórico referencial de la investigación asociada a *blockchain*, criptomonedas y *tokens*, dio lugar a la comprensión total del presente trabajo de diploma, así como sentar las bases teórico-metodológicas y conceptuales del mismo.
- La definición de la metodología de desarrollo de software, definió las pautas para el proceso ordenado y detallado de la propuesta de solución.
- La definición de las tecnología y herramientas que formaron parte del entorno de desarrollo, están acorde con los requerimientos necesarios para el desarrollo del sistema gestor.
- Las pruebas ejecutadas al software fueron un punto clave para detectar y corregir errores en el software, los resultados obtenidos al aplicar las técnicas cualitativas y cuantitativas de validación reafirmaron el correcto funcionamiento del sistema de gestión de pagos.

Recomendaciones

Se recomienda desarrollar otras funcionalidades e incorporarlas al sistema, como es el factor de doble autenticación para incrementar la seguridad de las transacciones.

Referencias bibliográficas

ALEJANDRO, P.V., [2020]. El impacto económico de las sanciones estadounidense a Cuba, 1994-2020. , pp. 31.

ALVAREZ, M.A., 2012. Manual de JavaScript. *DesarrolloWeb.com* [en línea]. [Consulta: 10 junio 2022]. Disponible en: https://desarrolloweb.com/manuales/manual-javascript.html.

ANTÓN, S. y GONZÁLEZ, Y. del S., 2022. Criptonegocios y blockchain en Cuba: ¿utopía o realidad? (+ Video). *Granma.cu* [en línea]. [Consulta: 9 junio 2022]. Disponible en: https://www.granma.cu/cuba/2022-04-12/criptonegocios-y-blockchain-en-cuba-utopia-o-realidad-12-04-2022-20-04-26.

ARIAS RIVERA, L., 2021. El bloqueo estadounidense contra Cuba. Una actualización. *CLACSO* [en línea]. [Consulta: 20 junio 2022]. Disponible en: https://www.clacso.org/el-bloqueo-estadounidense-contra-cuba-una-actualizacion/.

Aumentan los obstáculos a las operaciones bancarias de Cuba por sanciones de EEUU, según funcionario | OnCubaNews. [en línea], [2021]. [Consulta: 12 julio 2022]. Disponible en: https://oncubanews.com/cuba/aumentan-los-obstaculos-a-las-operaciones-bancarias-de-cuba-por-sanciones-de-eeuu-segun-funcionario/.

CABRERA SOTO, M., 2021. Criptomonedas: ¿qué son y qué pretenden ser?*. [en línea]. [Consulta: 2 junio 2022]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S0252-85842022000100008.

Exchange (criptomonedas). En: Page Version ID: 143715879, *Wikipedia, la enciclopedia libre* [en línea], 2022. [Consulta: 9 junio 2022]. Disponible en: https://es.wikipedia.org/w/index.php?title=Exchange_(criptomonedas)&oldid=143715879.

FINANCIAL, G.S.A., 2014. Microsoft ya acepta bitcoin para la compra de aplicaciones y juegos | Blog Bankinter. [en línea]. [Consulta: 19 octubre 2022]. Disponible en: https://www.bankinter.com/blog/lo-ultimo/microsoft-ya-acepta-bitcoin-para-la-compra-de-aplicaciones-y-juegos.

GARCÍA-NIETO, M., 2017. Aplicación del proceso unificado en el desarrollo de un software que estima el inventario y el crecimiento-rendimiento maderable en plantaciones de eucalipto. [en línea]. [Consulta: 13 septiembre 2022]. Disponible en: https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-04712017000100163.

GAUCHAT, J.D., 2012. *El gran libro de HTML5, CSS3 y Javascript*. S.l.: Marcombo. ISBN 978-84-267-1782-5.

GONZÁLEZ, R., 2020. Cómo la Inteligencia Artificial está transformando la industria de las criptomonedas. *Big Data Magazine* [en línea]. [Consulta: 2 junio 2022]. Disponible en: https://bigdatamagazine.es/como-la-inteligencia-artificial-esta-transformando-la-industria-de-las-criptomonedas.

GUERRERO, C.A., SUÁREZ, J.M. y GUTIÉRREZ, L.E., 2012. Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. [en línea]. [Consulta: 29 octubre 2022]. Disponible en: https://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-07642013000300012.

JIMÉNEZ, J. y GALLARDO, M., 2022. ¿Qué son las criptomonedas y cómo funcionan? [en línea]. [Consulta: 9 junio 2022]. Disponible en: https://www.santander.com/es/stories/guia-para-saber-que-son-las-criptomonedas.

JIMÉNEZ, P. y NIEVES, M., 2019. De la tecnología blockchain a la economía del token. *Derecho PUCP*, no. 83, pp. 61-87. ISSN 0251-3420. DOI 10.18800/derechopucp.201902.003.

KP, teleSUR-ds-, [2022]. ¿En qué consiste el bloqueo de EE.UU. contra Cuba? [en línea]. [Consulta: 20 junio 2022]. Disponible en: https://www.telesurtv.net/telesuragenda/Bloqueo-de-Estados-Unidos-contra-Cuba-20170202-0051.html.

MENDOZA PEÑA, D. y HERNÁNDEZ GONZÁLEZ, B., 2021. Extensión de la herramienta Visual Paradigm for UML para la evaluación y corrección de diagramas de actividades. En: Accepted: 2021-11-09T19:37:33Z, *UCIENCIA 2021* [en línea], [Consulta: 13 septiembre 2022]. Disponible en: https://repositorio.uci.cu/jspui/handle/123456789/9641.

NAKAMOTO, S., 2008. *Bitcoin: A Peer-to-Peer Electronic Cash System.* [en línea]. 2008. S.l.: s.n. Disponible en: https://bitcoin.org/bitcoin.pdf.

ORDOÑEZ, R.D.R., 2018. El trading como alternativa de trabajo. *Contexto*, vol. 7, pp. 1-4. ISSN 2346-0784. DOI 10.18634/ctxj.7v.0i.888.

PALLARES, unocero-J., 2022. ¿Qué es TON y cómo llevará a Telegram a los pagos con criptomonedas? *unocero* [en línea]. [Consulta: 19 octubre 2022]. Disponible en: https://www.unocero.com/noticias/ton-como-llevara-telegram-pagos-con-criptomonedas/.

PARMAR, D., 2021. Las 8 mejores plataformas de blockchain para crear aplicaciones financieras modernas. *Geekflare* [en línea]. [Consulta: 10 junio 2022]. Disponible en: https://geekflare.com/es/finance/blockchain-platforms-for-finance-applications/.

RAMÍREZ, L., 2021. ¿Qué es un token y qué tipos existen? [en línea]. [Consulta: 2 junio 2022]. Disponible en: https://www.iebschool.com/blog/tipos-de-token-que-es-finanzas/.

ROSSI, J. y HOULIN, J., 2021. Smart contracts como clearing house : un modelo de exchange descentralizado. En: Accepted: 2022-04-26T21:03:56Z [en línea], [Consulta: 9 junio 2022]. Disponible en: http://repositorio.udesa.edu.ar/jspui/handle/10908/19120.

SANTAELLA, J., 2021. ¿En qué consiste la minería de criptomonedas y qué usos tiene? *Economia3* [en línea]. [Consulta: 9 junio 2022]. Disponible en: https://economia3.com/que-es-mineria-criptomonedas-y-que-usos-tiene/.

SOCIETY, F., [2022]. Cuatro empresas que lideran el auge de la tecnología Blockchain. *Funds Society* [en línea]. [Consulta: 19 octubre 2022]. Disponible en: https://www.fundssociety.com/es/news/alternatives/glx22-cuatro-empresas-que-lideran-el-auge-de-la-tecnologia-blockchain/.

ZORRILLA MORENO, R., 2019. Nuevos métodos de financiación del emprendimiento : financiación por tokens block-chain. En: Accepted: 2018-06-12T08:21:23Z [en línea], [Consulta: 2 junio 2022]. Disponible en: https://repositorio.comillas.edu/xmlui/handle/11531/27591.

Anexos

Pequeña entrevista realizada a Luis Raciel Rodríguez Silva, director del centro de servicios digitales de la Universidad de las Ciencias Informáticas.

1- ¿Dónde cree que podamos utilizar este tipo de servicios de pagos digitales?

En aquellos servicios cortos, que no dependen de mucho tiempo para su implementación, incluso hay servicios que se consumen en la red, el cual lo veo muy útil; uno son los cursos, que generalmente los abonan personas naturales, que deben encontrar algún banco que permita transacciones con Cuba, y son realmente pocos en Latinoamérica; también tenemos algunas plataformas que tratamos de abrirlas al exterior, caso de APKlis, Picta y Cosmos, y tratamos de encontrar la forma de monetizarlas con personas extranjeras, y creo que las criptomonedas pueden funcionar como micropagos para esto; y el último caso viene por parte de los servicios profesionales informáticos.

2- ¿Qué opina de la nueva Resolución emitida por el Banco Central de Cuba con respecto a las criptomonedas y su legalidad en Cuba?

Yo pienso que no debemos quedarnos atrás con este tema, es algo que está revolucionando el mundo de las finanzas y los mercados. Es una posibilidad real de implantar un método de pago no solo para la Uci, sino para Cuba, y me alegra que el banco tenga los ojos puestos en esta oportunidad de negocios.

3- ¿Cómo ha sido el proceso de pagos para los servicios informáticos hasta ahora?

Han sido años tediosos para nuestra Universidad, no llevamos la cuenta como tal, pero tenemos un largo historial de procesos que se abandonan a mitad de camino debido a la dificultad de los pagos en los clientes, pues a veces, dependiendo del proyecto, tomamos pagos a plazos, y en cuanto se dificultan tenemos como institución reembolsarlos. Muchas veces apuramos proyectos por esta razón, pero no quiero imaginarme el problema que nos ahorrara los pagos digitales por criptomonedas.