Logotipo

El contenido generado por IA puede ser incorrecto.

Universidad Central del Ecuador

Facultad de Ingeniería y Ciencias Aplicadas

Carrera de Ingeniería Informática

**Prototipo de aplicación para la certificación de documentos digitales en instituciones de educación superior basado en la tecnología Blockchain.**

Trabajo de titulación – Opción: Proyecto de investigación presentado para obtener el grado académico de Ingeniero Informático.

Autores: Henry Paolo Felicita Vasco

Tutor: Ing. Santiago Morales Cardoso, PhD

Quito, 2024

**Cesión de derechos de autor.**

1. **Cesión de derechos de autor.**

Mediante el envío del presente correo electrónico ACEPTO EN FORMA EXPRESA en calidad de autor, titular de los derechos morales y patrimoniales del trabajo de titulación: ***Prototipo de aplicación para la certificación de documentos digitales en instituciones de educación superior basado en la tecnología blockchain***. De conformidad con el Artículo 114 del **Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación**:

“De los titulares de derechos de obras creadas en las instituciones de educación superior y centros educativos.- En el caso de las obras creadas en centros educativos, universidades, escuelas politécnicas, institutos superiores técnicos, tecnológicos, pedagógicos, de artes y los conservatorios superiores, e institutos públicos de investigación como resultado de su actividad académica o de investigación tales como trabajos de titulación, proyectos de investigación o innovación, artículos académicos, u otros análogos, sin perjuicio de que pueda existir relación de dependencia, la titularidad de los derechos patrimoniales corresponderá a los autores. Sin embargo, el establecimiento tendrá una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra con fines académicos.”

Asimismo, autorizo a la Universidad Central del Ecuador para que realice la publicación de este trabajo de titulación en el repositorio institucional en formato digital tal como lo establece el documento RHCU.SO.22 CIRCULAR No. 027-2021 del 11 de junio del año 2021 con nombre “Creación del archivo digital de la Universidad Central del Ecuador”, y de conformidad a lo dispuesto en el Artículo 144 de la **Ley Orgánica de Educación Superior**:

“Trabajos de Titulación en formato digital. - Todas las instituciones de educación superior estarán obligadas a entregar los trabajos de titulación que se elaboren para la obtención de títulos académicos de grado y posgrado en formato digital para ser integradas al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.”

El autor declara que la obra objeto de la presente autorización es original en su forma de expresión y no infringe el derecho de autor de terceros, asumiendo la responsabilidad por cualquier reclamación que pudiera presentarse por esta causa y liberando a la Universidad de toda responsabilidad.

El autor reconoce que este mensaje electrónico de aceptación reemplaza a mi firma física, y tiene la validez legal de conformidad a lo dispuesto en la **Ley de Comercio Electrónico, Firmas y Mensajes de Datos**, en sus artículos 3, 6 y 8,

**“Art. 3.-** Incorporación por remisión. - Se reconoce validez jurídica a la información no contenida directamente en un mensaje de datos, siempre que figure en el mismo, en forma de remisión o de anexo accesible mediante un enlace electrónico directo y su contenido sea conocido y aceptado expresamente por las partes.”

**“Art. 6.-** Información escrita. - Cuando la Ley requiera u obligue que la información conste por escrito, este requisito quedará cumplido con un mensaje de datos, siempre que la información que éste contenga sea accesible para su posterior consulta.”

**“Art. 8.-** Conservación de los mensajes de datos. - Toda información sometida a esta Ley, podrá ser conservada; este requisito quedará cumplido mediante el archivo del mensaje de datos, siempre que se reúnan las siguientes condiciones:

1. Que la información que contenga sea accesible para su posterior consulta;
2. Que sea conservado con el formato en el que se haya generado, enviado o recibido, o con algún formato que sea demostrable que reproduce con exactitud la información generada, enviada o recibida;
3. Que se conserve todo dato que permita determinar el origen, el destino del mensaje, la fecha y hora en que fue creado, generado, procesado, enviado, recibido y archivado; y,
4. Que se garantice su integridad por el tiempo que se establezca en el reglamento a esta ley.

Toda persona podrá cumplir con la conservación de mensajes de datos, usando los servicios de terceros, siempre que se cumplan las condiciones mencionadas en este artículo.”

(Este documento esta validado por el autor a través del formato en el archivo de Mensaje de datos con descripción: “DERECHOS DE AUTOR”)

**Henry Paolo Felicita Vasco**

**1723166532**

**Dedicatoria.**

Dedico este trabajo con todo mi corazón a mi madre, María Soledad Vasco, por ser mi pilar más firme, por su amor incondicional, su ejemplo de vida y el constante apoyo que me ha brindado en cada paso de este camino. Su fortaleza y entrega han sido mi mayor fuente de inspiración.

Extiendo esta dedicatoria a mi esposa, Yahira Michell Andrade, por su amor, comprensión y compañía inquebrantable en los momentos más difíciles. Su presencia y aliento me han dado la fuerza necesaria para seguir adelante y superar cada desafío.

Con profundo amor y gratitud, también dedico este logro a la memoria de mi abuelita, María Mercedes Chasipanta (1932–2009), quien me inculcó principios de ética y humildad, me enseñó a caminar siempre por el sendero correcto, y me mostró que el sacrificio y la constancia son las claves para alcanzar nuestros sueños.

**Agradecimiento.**

En primer lugar, deseo expresar mi más profundo agradecimiento a mis padres, Pedro y Soledad, por ser un pilar fundamental en cada etapa de mi formación, desde el inicio de mi camino como estudiante hasta la culminación de mi carrera universitaria, siempre estuvieron presentes, brindándome todo su apoyo, guiándome con sabiduría, escuchándome con paciencia y rodeándome con su amor incondicional para alcanzar mis objetivos, espero que todo el tiempo hayan sentido cuanto los valoro, respeto y admiro.

A mi esposa, Yahira, gracias por caminar conmigo en este largo trayecto, tu presencia constante, tu preocupación diaria y tu apoyo incondicional han sido fundamentales en todo momento, especialmente durante la elaboración de esta tesis, gracias por ser parte de cada paso y por acompañarme con tanto amor.

Extiendo también mi agradecimiento a mis hermanos, sobrinos y tíos, cuyas palabras de aliento y muestras de ánimo me han dado fuerza. Gracias a ustedes, estos últimos meses han sido más llevaderos y llenos de motivación. Su compañía ha sido fundamental en el cierre de una etapa más en mi vida.

Finalmente, quiero expresar mi sincero agradecimiento al Ing. Santiago Morales Cardoso, PhD, por su valiosa guía, dedicación y compromiso, no solo en el desarrollo de este trabajo, sino también por su labor como docente, siempre transmitiendo un conocimiento claro, sólido y enriquecedor, sus consejos han sido de gran ayuda tanto en el ámbito académico como en lo personal.

**Resumen.**

Las instituciones de educación superior enfrentan desafíos significativos en la certificación y validación de documentos académicos, especialmente en lo referente a la integridad y verificación confiable de la información. Este trabajo documenta el desarrollo de un prototipo de aplicación descentralizada para la gestión de certificaciones académica, utilizando la tecnología blockchain como solución innovadora que permite aprovechar sus propiedades para enfrentar estos retos. El estudio inicia con un análisis de la red blockchain y sus aplicaciones especialmente en el ámbito educativo, posteriormente se realiza una entrevista estructurada para comprender el proceso actual de generación de certificados académicos. Para el desarrollo de la aplicación, se adoptó la metodología de investigación aplicada, la cual proporciona un marco flexible adecuado para desarrollar un producto, se comienza con la propuesta del prototipo, en la que se recopilan las necesidades de los usuarios finales y se diseña el modelo de la aplicación, se describe la arquitectura del sistema y se emplean herramientas de código abierto y marcos de desarrollo que facilitan la integración de los distintos componentes del sistema con la red blockchain, una vez definido lo anterior se codifica y despliega el contrato inteligente donde se definen las reglas de negocio, finalmente se implementa una interfaz gráfica de usuario que permite la interacción directa con el contrato inteligente, brindando a los usuarios un acceso seguro y verificable a los certificados emitidos por el Instituto Académico de Idiomas.

Palabras clave: Blockchain, Smart contract, Certificación académica, Aplicaciones descentralizadas, Prototipo.

**Abstract.**

Higher education institutions face significant challenges in the certification and validation of academic documents, particularly concerning the integrity and reliable verification of information. This work presents the development of a decentralized application prototype for managing academic certifications, using blockchain technology as an innovative solution that leverages its properties to address these challenges. The study begins with an analysis of blockchain networks and their applications, particularly in the educational sector. A structured interview was conducted to understand the current process of generating academic certificates.  
To develop the application, an applied research methodology was adopted, providing a flexible framework suitable for creating a functional product. The process started with the design of the prototype, including the collection of end-user requirements and the modeling of the application. The system architecture is described, and open-source tools and development frameworks are used to facilitate the integration of the system components with the blockchain network. Once the previous steps were completed, the smart contract was coded and deployed, defining the business logic rules. Finally, a graphical user interface was implemented to allow users to interact directly with the smart contract, providing secure and verifiable access to the certificates issued by the Academic Language Institute.

# Índice general de contenidos

[Índice general de contenidos i](#_Toc200706752)

[Lista de tablas ii](#_Toc200706753)

[Lista de figuras ii](#_Toc200706754)

[Lista de anexos iv](#_Toc200706755)

[Capítulo 1 5](#_Toc200706756)

[1. Generalidades 5](#_Toc200706757)

[1.1. Introducción 5](#_Toc200706758)

[1.2. Antecedentes 6](#_Toc200706759)

[1.3. Planteamiento del problema 8](#_Toc200706760)

[1.4. Hipótesis y Justificación 9](#_Toc200706761)

[1.4.1. Hipótesis 9](#_Toc200706762)

[1.4.2. Justificación 9](#_Toc200706763)

[1.5. Objetivos 10](#_Toc200706764)

[1.5.1. General 10](#_Toc200706765)

[1.5.2. Específicos 11](#_Toc200706766)

[Capítulo 2 12](#_Toc200706767)

[2. Marco Teórico 12](#_Toc200706768)

[2.1. Conceptos 12](#_Toc200706769)

[2.1.1. Certificación 12](#_Toc200706770)

[2.1.2. Certificación Académica 12](#_Toc200706771)

[2.2. Blockchain 12](#_Toc200706772)

[2.2.1. Características de Blockchain 13](#_Toc200706773)

[2.2.2. Arquitectura de blockchain 14](#_Toc200706774)

[2.2.3. Bloque 14](#_Toc200706775)

[2.2.4. Mecanismo de Consenso 16](#_Toc200706776)

[2.3. Smart contracts 17](#_Toc200706777)

[2.3.1. Estructura de un smart contract 17](#_Toc200706778)

[2.4. Herramientas tecnológicas 18](#_Toc200706779)

[Capítulo 3 23](#_Toc200706780)

[3. Metodología experimental 23](#_Toc200706781)

[Capítulo 4 25](#_Toc200706782)

[4. Cálculos y resultados 25](#_Toc200706783)

[4.1. Propuesta del prototipo 25](#_Toc200706784)

[4.1.1. Propósito 25](#_Toc200706785)

[4.1.2. Alcance 25](#_Toc200706786)

[4.1.3. Identificación de procesos 26](#_Toc200706787)

[4.1.4. Identificar a los stakeholders 27](#_Toc200706788)

[4.1.5. Modelo propuesto 27](#_Toc200706789)

[4.1.6. Funciones del prototipo 28](#_Toc200706790)

[4.1.7. Requisitos funcionales 35](#_Toc200706791)

[4.1.8. Requisitos No funcionales 36](#_Toc200706792)

[4.2. Descripción de la arquitectura 37](#_Toc200706793)

[4.3. Selección de herramientas 38](#_Toc200706794)

[4.3.1. Node.js 39](#_Toc200706795)

[4.3.2. React.js 39](#_Toc200706796)

[4.3.3. Tailwind CSS 39](#_Toc200706797)

[4.3.4. Vite JS 39](#_Toc200706798)

[4.3.5. Metamask 40](#_Toc200706799)

[4.3.6. Express JS 40](#_Toc200706800)

[4.3.7. MongoDB 40](#_Toc200706801)

[4.3.8. Hardhat 40](#_Toc200706802)

[4.3.9. Ethers JS 41](#_Toc200706803)

[4.4. Desarrollo del prototipo propuesto 41](#_Toc200706804)

[4.4.1. Certificación académica LangCertIAI 41](#_Toc200706805)

[4.4.2. Estructura del proyecto 41](#_Toc200706806)

[4.4.3. Desarrollo del contrato inteligente 42](#_Toc200706807)

[4.4.4. Codificación del contrato inteligente 46](#_Toc200706808)

[4.4.5. Copilar y desplegar el contrato 63](#_Toc200706809)

[4.4.6. Incorporar dentro de la red blockchain de prueba 65](#_Toc200706810)

[4.5. Interacción con el contrato inteligente 66](#_Toc200706811)

[Capítulo 5 79](#_Toc200706812)

[5. Conclusiones y recomendaciones 79](#_Toc200706813)

[5.1. Discusión 79](#_Toc200706814)

[5.2. Conclusiones 81](#_Toc200706815)

[5.3. Recomendaciones 82](#_Toc200706816)

[6. Referencias 83](#_Toc200706817)

[7. Anexos 89](#_Toc200706818)

# Lista de tablas

[Tab. 1. Ventajas e inconvenientes de una red blockchain. 19](#_Toc200705205)

[Tab. 2. Ventajas e inconvenientes del entorno de desarrollo integrado 21](#_Toc200705206)

[Tab. 3. Ventajas e inconvenientes de los diferentes lenguajes de programación 21](#_Toc200705207)

[Tab. 4. Hallazgos 27](#_Toc200705208)

[Tab. 5. Descripción de los actores 27](#_Toc200705209)

[Tab. 6. Autenticación y Autorización 31](#_Toc200705210)

[Tab. 7. Gestionar certificados 33](#_Toc200705211)

[Tab. 8. Validar Certificado 35](#_Toc200705212)

[Tab. 9. Requisitos funcionales 36](#_Toc200705213)

[Tab. 10. Requisitos no funcionales 36](#_Toc200705214)

[Tab. 11. Funciones del contrato LanguageCertificate 50](#_Toc200705215)

[Tab. 12. Dependencias del archivo backend. 51](#_Toc200705216)

[Tab. 13. Tabla de restricción de acceso a determinadas rutas 58](#_Toc200705217)

[Tab. 14. Funciones del archivo certificatecontrolle.js 59](#_Toc200705218)

# Lista de figuras

[Fig. 1. Estructura blockchain 14](#_Toc200706370)

[Fig. 2. Estructura de un bloque 15](#_Toc200706371)

[Fig. 3. Estructura de un contrato inteligente 18](#_Toc200706372)

[Fig. 4. Modelo propuesto: Certificación descentralizada de documentos académicos. 28](#_Toc200706373)

[Fig. 5. Funciones del prototipo de software, función nivel cero. 29](#_Toc200706374)

[Fig. 6. Diagrama de casos de uso para la autenticación y autorización, función en nivel uno. 30](#_Toc200706375)

[Fig. 7. Diagramas de caso para gestionar los certificados, función en nivel uno. 32](#_Toc200706376)

[Fig. 8. Diagramas de caso para validar los certificados, función en nivel uno. 34](#_Toc200706377)

[Fig. 9. Arquitectura propuesta de la aplicación. 38](#_Toc200706378)

[Fig. 10. Estructura de directorios del proyecto 41](#_Toc200706379)

[Fig. 11. Esquema de fases para el desarrollo del contrato inteligente 42](#_Toc200706380)

[Fig. 12. Estructura de directorios de la carpeta smart-contract 44](#_Toc200706381)

[Fig. 13. Código fuente del archivo deploy.js 45](#_Toc200706382)

[Fig. 14. Código fuente del archivo hardhat.config.js 46](#_Toc200706383)

[Fig. 15. Código fuente del archivo hardhat.config.js 46](#_Toc200706384)

[Fig. 16. Contrato inteligente llamado LanguageCertificate.sol 47](#_Toc200706385)

[Fig. 17. Especificación de la versión de Solidity 47](#_Toc200706386)

[Fig. 18. Estructura de datos del certificado 48](#_Toc200706387)

[Fig. 19. Mapeo del certificado 48](#_Toc200706388)

[Fig. 20. Función emitir certificado. 49](#_Toc200706389)

[Fig. 21. Implementación de las funciones en el contrato LanguageCertificate 50](#_Toc200706390)

[Fig. 22. Estructura de directorios de la carpeta backend 51](#_Toc200706391)

[Fig. 23 Archivo de configuración .env 52](#_Toc200706392)

[Fig. 24. Contenido del archivo .env 52](#_Toc200706393)

[Fig. 25. Conexión con la base de datos MongoDB 53](#_Toc200706394)

[Fig. 26. Estructura para almacenar los certificados en la base de datos 54](#_Toc200706395)

[Fig. 27. Estructura para almacenar los certificados en la base de datos 55](#_Toc200706396)

[Fig. 28. Autenticación de los roles de los usuarios. 56](#_Toc200706397)

[Fig. 29. Rutas para la gestión de certificados 57](#_Toc200706398)

[Fig. 30. Contenido del archivo authMiddleware.js autentificación 58](#_Toc200706399)

[Fig. 31. Contenido del archivo certificateController.js 59](#_Toc200706400)

[Fig. 32. Emitir certificado 60](#_Toc200706401)

[Fig. 33. Verificar validez 60](#_Toc200706402)

[Fig. 34. Obtener un certificado 60](#_Toc200706403)

[Fig. 35. Obtener todos los certificados 61](#_Toc200706404)

[Fig. 36. Generar PDF 61](#_Toc200706405)

[Fig. 37. Invalidar certificado 62](#_Toc200706406)

[Fig. 38. Contenido del archivo auth.controler.js 63](#_Toc200706407)

[Fig. 39. Inicialización del nodo local de Ethereum 64](#_Toc200706408)

[Fig. 40. Archivos generados de archivo dentro de la carpeta artifacts 65](#_Toc200706409)

[Fig. 41. Transacción correspondiente al despliegue del contrato. 65](#_Toc200706410)

[Fig. 42. Estructura de directorios de la carpeta fronend 67](#_Toc200706411)

[Fig. 43. Contenido del archivo tailwind.config.js 68](#_Toc200706412)

[Fig. 44. Contenido del archivo index.html 69](#_Toc200706413)

[Fig. 45. Contenido del archivo main.jsx 69](#_Toc200706414)

[Fig. 46. Contenido del archivo App.jsx 70](#_Toc200706415)

[Fig. 47. Contenido del archivo Navbar.jsx 71](#_Toc200706416)

[Fig. 48. Estructura de la carpeta pages 72](#_Toc200706417)

[Fig. 49. Página principal del componente Home.jsx 73](#_Toc200706418)

[Fig. 50. Página de inicio de sesión del componente Login.jsx 74](#_Toc200706419)

[Fig. 51. Página de registro de nuevos usuarios al sistema del componente Register.jsx 75](#_Toc200706420)

[Fig. 52. Página de emisión de certificados del componente IssueCertificate.jsx 76](#_Toc200706421)

[Fig. 53. Página para verificar la validez del certificado, del componente VerifyCertificate.jsx 77](#_Toc200706422)

[Fig. 54. Página para consultar y visualizar el certificado, del componente GetCertificate.jsx 78](#_Toc200706423)

[Fig. 55. Página de vista global y control sobre los certificados existentes, del componente AllCertificates.jsx 78](#_Toc200706424)

# Lista de anexos

# Capítulo 1

## Generalidades

### Introducción

En la actualidad la mayoría de las instituciones educativas continúan dependiendo en gran medida de procesos manuales y centralizados para la gestión de certificados académicos, cada institución maneja su propio sistema para generar, almacenar y verificar estos documentos. Las instituciones emiten certificados académicos y estos son entregados al estudiante de forma física o digital, para posteriormente ser validados por otras dependencias de la universidad, instituciones externas o empleadores que solicitan su autenticidad, estos procedimientos tradicionales están fragmentadas entre diversas áreas e involucra múltiples actores, esta situación presentan numerosos desafíos como la posibilidad de imitar documentos mediante programas de edición, posible pérdida de información sensible en caso de fallos del sistema, demoras en la verificación de la autenticidad, y costos administrativos elevados, lo cual compromete la eficiencia, seguridad y confiabilidad del sistema de gestión documental académica.

La Universidad Central del Ecuador cuenta con aproximadamente 50.000 estudiantes en todos sus niveles educativos incluidos los de preparación, a partir del 2023 es obligatorio que los estudiantes de tercer nivel aprueben un nivel B1 de idioma como requisito para obtener su título. El Instituto Académico de Idiomas registra actualmente 5.271 estudiantes matriculados en los distintos idiomas y niveles, lo que genera un alto flujo de solicitudes de certificados académicos durante todo el año. La dirección de admisiones y registro emite alrededor de 60 certificados académicos al mes y por solicitud directa de los estudiantes un promedio de 1120 por periodo, para atender esta demanda, es necesario contar con una infraestructura tecnológica adecuada y una coordinación eficiente entre los actores involucrados.

Ante este contexto, se propone el desarrollo de un prototipo de aplicación descentralizada basado en tecnología blockchain y contratos inteligentes, esta solución permite emitir, verificar y almacenar certificados académicos de forma segura, confiable y sin necesidad de intermediarios, garantizando la integridad de la información. Cuando el Instituto Académico de Idiomas emite un certificado correspondiente al nivel de idioma aprobado, la transacción se registra automáticamente en la cadena de bloques mediante un contrato inteligente que contiene datos esenciales como nombre, apellido, cédula del estudiante, así como los detalles del idioma, nivel aprobado y la institución emisora.

A través de la aplicación web, los estudiantes como las instituciones externas pueden acceder fácilmente y compartir de forma segura sus certificados digitales, como resultado, se obtiene una herramienta tecnológica capaz de generar certificados digitales inmutables registrados en la blockchain, resolviendo así el problema identificado y optimizando los procesos de certificación y validación de títulos académicos dentro de la institución.

### Antecedentes

El desarrollo de nuevas tecnologías abre posibilidades para la creación de negocios innovadores y la optimización de los ya existentes, la cuarta revolución industrial, impulsada por internet, ha dado lugar a importantes avances tecnológicos que han transformado numerosos aspectos de la vida cotidiana, generando un notable impacto en la sociedad. La tecnología blockchain surge como una solución innovadora para abordar diversos problemas en distintas áreas como la salud, educación el gobierno entre otros. En el ámbito educativo, esta tecnología facilita la eliminación de cuellos de botella en diversos procesos y proporciona una mayor protección para la información dentro de las instituciones. [1]

Generalmente las instituciones educativas suelen gestionar grandes volúmenes de expedientes, los cuales son frecuentemente solicitados por los estudiantes para diversos fines, como la transferencia de créditos, la obtención de becas, seguros, cambio de carrera o el cumplimiento de requisitos para ingresar a nuevos programas académicos. Aunque la transferencia de expedientes académicos es una tarea común en el día a día de las instituciones, a menudo implica costos significativos debido a que los procesos de transferencia y verificación son altamente manuales. [2]

Además, de los problemas asociados con la emisión de certificados académicos que amenazan la integridad de las instituciones educativas, incluso en las más prestigiosas, como el Instituto Tecnológico de Massachusetts. [3]

En ese sentido la tecnología blockchain podrían abordar tales retos a través del despliegue de una plataforma abierta, descentralizada, y directa, en donde se pueda des intermediar el flujo de información, logrando que se pueda mejorar la transparencia y la confianza en los registros educativos. [4]

**Aplicación de blockchain en el sector de la salud**

La tecnología blockchain se presenta como una herramienta muy prometedora para abordar el problema de la dispersión de la información médica de los pacientes cuando reciben atención en distintos centros de salud. En la actualidad, los datos médicos suelen estar fragmentados y carecen de un expediente único, lo que dificulta su acceso. Gracias a la blockchain, es posible recuperar de manera rápida y segura los registros médicos, garantizando la integridad y permanencia de la información relacionada con el diagnóstico y tratamiento de cada paciente. Todo lo que se refiere a exámenes, biopsias y procedimientos quedan almacenados en una red descentralizada, creando un historial médico inmutable que solo puede ser consultado con la autorización del paciente, lo cual es fundamental para el desarrollo de tratamientos personalizados. [5]

**Aplicación de blockchain en la cadena de suministros**

La gestión y distribución de medicamentos o productos farmacéuticos representa un desafío crucial tanto a nivel nacional como internacional, existiendo el riesgo de que estos productos sean adulterados, etiquetados de manera fraudulenta o desviados desde su punto de origen con la intención de engañar a los consumidores, haciéndoles creer que están adquiriendo un producto auténtico. Asimismo, la cadena de suministro en el sector de la salud es un complejo sistema de negocios que abarca procesos como adquisición, producción, almacenamiento y distribución de productos y servicios médicos. Esta red incluye normas y procedimientos esenciales para garantizar el acceso a los medicamentos y satisfacer las necesidades de los pacientes, profesionales de la salud e instituciones médicas. En términos generales, la circulación de medicamentos falsificados o de baja calidad representa un riesgo significativo para los pacientes y sus familias. En este contexto, la tecnología blockchain surge como una solución innovadora para optimizar la transparencia, eficiencia y seguridad en la cadena de suministro. Al descentralizar el control de los datos y garantizar un registro inalterable de la información, esta tecnología permite eliminar intermediarios y establecer sistemas de seguimiento en tiempo real, ofreciendo mayor confianza a todos los actores involucrados en el proceso. [6]

### Planteamiento del problema

Los certificados académicos son documentos oficiales que respaldan el historial académico de una persona y que son emitido por el Instituto académico de Idiomas de la Universidad Central del Ecuador estos constituye un componente esencial en el perfil profesional de los estudiantes, sin embargo, los métodos actuales utilizados para emitir este tipo de certificados están sujetos a diversas situaciones que pueden generar inconvenientes, por ejemplo, la certificación y verificación de documentos académicos como títulos y certificados, sigue dependiendo en gran medida de procesos manuales y centralizados, esto conlleva a posibles problemas como la falsificación de documentos, perdida de información sensible, demoras en la verificación de la autenticidad y costos administrativos elevados, lo cual compromete la eficiencia, seguridad y confiabilidad del sistema de gestión documental académica.

Las soluciones basadas en el envío de documentos por correo electrónico, la transferencia de archivos en formato PDF y el uso de portales seguros para la entrega de certificados digitales son cada vez más comunes, aunque estas alternativas representan un avance hacia la digitalización aún presentan desafíos en términos de escalabilidad, integridad y verificación confiable [1].

En este contexto, la tecnología blockchain ha emergido como una alternativa sólida y prometedora para garantizar la autenticidad, inmutabilidad y trazabilidad de documentos digitales. Sin embargo, a pesar de su potencial, muchas universidades aún no han adoptado esta tecnología debido a la ausencia de soluciones prácticas y accesibles que faciliten su implementación.

Por ello, surge la necesidad de desarrollar un prototipo de aplicación basado en la tecnología blockchain que permita a las instituciones de educación superior certificar y verificar documentos digitales de manera segura, descentralizada y eficiente, por esta razón este estudio busca analizar la viabilidad de dicha solución y evaluar su impacto en la reducción del fraude académico y la optimización de procesos administrativos en el ámbito educativo.

### Hipótesis y Justificación

#### Hipótesis

¿Es posible implementar un prototipo basado en Blockchain para la certificación de documentos digitales en instituciones de educación superior?

#### Justificación

En Ecuador, las instituciones educativas operan de manera independiente en cuanto a la emisión y verificación de certificados académicos, estos sistemas generalmente centralizados, incrementan la carga de trabajo administrativo especialmente a la hora de verificar la autenticidad de los documentos, los cuales pueden ser vulnerables a perdidas o falsificaciones, como resultado los estudiantes deben encargarse de conservar sus certificados para posteriormente integrarlos en su portafolio profesional.

Este proyecto propone el desarrollo de un prototipo para la certificación académica utilizando una blockchain, con el fin de aprovechar las ventajas que ofrece este tipo de tecnología, tales como, mantener los registros descentralizados, seguros, confiables y de acceso ágil, esto permitirá optimizar los procesos de validación de los certificados de culminación de un idioma emitidos por el Instituto Académico de Idiomas de la Universidad Central del Ecuador.

La adopción de esta tecnología en el ámbito educativo ofrece múltiples beneficios, como la eliminación de intermediarios en la validación de documentos reduciendo errores y tiempos de espera, también facilita el acceso inmediato a los registros académicos, eliminando la necesidad de solicitar documentos físicos. Además, de garantizar la autenticidad e inmutabilidad de la información reduciendo los riesgos de pérdidas o falsificación, y fortaleciendo la transparencia y confianza de los sistemas de gestión académica.

El impacto de esta propuesta no se limita a la Universidad Central del Ecuador, sino que también puede inspirar a otras instituciones de educación superior interesadas en modernizar sus procesos, transformando el sistema hacia entornos interconectados, abiertos y confiables, capaces de registrar con precisión el desempeño estudiantil y cualquier otra información relevante relacionada con la trayectoria universitaria de los estudiantes.

### Objetivos

#### General

Crear un prototipo de aplicación web utilizando tecnología blockchain para la certificación de la culminación de estudios del idioma inglés en el Instituto Académico de Idiomas de la Universidad Central del Ecuador, garantizando la autenticidad, seguridad y transparencia en la validación de los certificados académicos.

#### Específicos

* Proponer un modelo de solución basado en blockchain para la validación de certificados de estudios del idioma inglés en el Instituto Académico de Idiomas de la Universidad Central del Ecuador.
* Identificar y seleccionar la herramienta tecnológica más adecuada para el desarrollo del prototipo.
* Aplicar una metodología adecuada que permita mejorar la flexibilidad y eficiencia en el desarrollo del proyecto.

# 

# Capítulo 2

## Marco Teórico

### Conceptos

#### Certificación

El término certificación se define como la acción y efecto de certificar o el documento en que se asegura la verdad de un hecho. Es decir, es un documento emitido por una entidad competente que confirma la autenticidad de algo. La certificación se materializa en un certificado, que puede ser de distintos tipos, como certificación de empresas, productos y personas. [7]

#### Certificación Académica

Es un documento oficial que acredita el historial académico de una persona. Generalmente, es emitido por una institución educativa (como una universidad) y contiene información detallada sobre los estudios realizados por el estudiante, incluyendo las asignaturas cursadas, las calificaciones obtenidas, las fechas de asistencia y la titulación alcanzada.

Este tipo de certificación es útil cuando un estudiante o egresado necesita demostrar su formación académica para trámites laborales, admisiones a otros programas educativos, convalidaciones de estudios en el extranjero o solicitudes de becas, entre otros. [1]

### Blockchain

El término *Blockchain*, también conocido como *cadena de bloques*, es una tecnología considerada disruptiva que evoluciona constantemente y es utilizada por diversas empresas para aprovechar sus características. Se considera una de las principales tecnologías de la cuarta revolución industrial. [8]

Blockchain es una red compuesta por varios nodos que contienen registros almacenados en cadenas de bloques. Funciona como una base de datos distribuida en la que la información se organiza en bloques, y cada bloque está vinculado al anterior mediante criptografía, formando una cadena. Esta cadena se replica en cada nodo de la red, garantizando disponibilidad y seguridad. [9]

Esta tecnología elimina la necesidad de un tercero como verificador, ya que la red misma verifica y valida las transacciones, haciendo que sea extremadamente difícil alterar o modificar la información sin que el resto de la red lo detecte. [10]

#### Características de Blockchain

A continuación, se presentan algunas propiedades que distinguen a esta tecnología.

1. **Descentralización:** La información dentro de una blockchain se distribuye entre múltiples nodos en la red, evitando que una sola entidad tenga control absoluto sobre los registros. Esto minimiza el riesgo de manipulación y fallos del sistema. [11]
2. **Inmutabilidad:** Una de las características esenciales de blockchain es que la información se registra en bloques enlazados criptográficamente. Una vez añadido un bloque a la cadena, no puede ser alterado ni eliminado, garantizando así la seguridad e integridad de los datos almacenados. [11]
3. **Disponibilidad:** Dado que la cadena de bloques está replicada en múltiples nodos, la información es accesible en todo momento. Esto garantiza una alta disponibilidad, ya que las transacciones no dependen de un único servidor. [12]
4. **Transparencia:** En una blockchain pública, todos los participantes pueden visualizar las transacciones registradas, lo que permite su verificación y auditoría. Esta cualidad fortalece la confianza en la red, ya que cualquiera puede revisar el historial de operaciones. [1]
5. **Anonimato y privacidad:** A pesar de la transparencia de las blockchain públicas, los usuarios pueden preservar su identidad mediante direcciones alfanuméricas en lugar de nombres reales. En el caso de blockchain privadas, es posible ajustar los niveles de anonimato y control de acceso para una mayor protección en las transacciones. [9]

#### Arquitectura de blockchain

Una cadena de bloques es una secuencia de bloques que almacena un historial completo de todas las transacciones en un libro contable descentralizado y público.

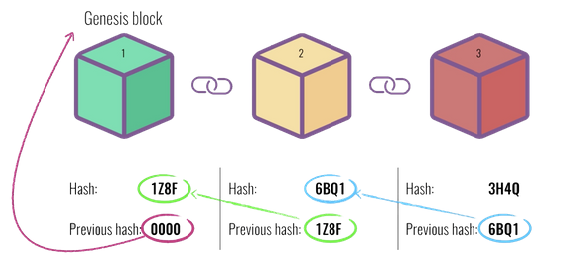


Fig. 1. Estructura blockchain

Nota: Álvaro Santos García [13]

La figura 1 ilustra un ejemplo de una cadena de bloques. Cada bloque está vinculado al anterior mediante un hash criptográfico, en un bloque solo puede existe un bloque padre, el único bloque que no contiene un bloque padre es el primer bloque de la cadena también conocido como bloque génesis. [14]

#### Bloque

Un bloque es un registro público que agrupa un conjunto de transacciones verificadas y distribuidas en todos los nodos que conforman una cadena de bloques. [8]

Diagrama, Texto

El contenido generado por IA puede ser incorrecto.

Fig. 2. Estructura de un bloque

Nota: Carlos Reyes Sánchez [8]

Como muestra la figura 2, la estructura general de un bloque dentro de una blockchain, el cual se compone de dos partes principales: el encabezado y el cuerpo.

El encabezado del bloque contiene metadatos esenciales, como:

1. La *versión del bloque*, que define las reglas de validación que deben aplicarse.
2. Una *marca de tiempo (timestamp),* que registra el momento exacto en el que se creó el bloque.
3. El *tamaño del bloque*, medido en bits, que establece el límite para considerar válido un hash.
4. El *nonce*, un número entero que comienza en cero y aumenta progresivamente con cada intento de resolución del problema criptográfico asociado a la creación del bloque.
5. El *hash del bloque padre*, un valor de 256 bits que enlaza con el bloque anterior, excepto en el caso del bloque génesis.
6. La *raíz del árbol de Merkle*, una estructura de datos que almacena hashes criptográficos organizados de manera jerárquica, facilitando la verificación eficiente de la integridad de grandes volúmenes de información. [8]

Por otro lado, el cuerpo del bloque almacena las transacciones validadas, asegurando la integridad y la inmutabilidad de los datos dentro de la cadena. [12]

Gracias a la estructura del árbol de Merkle, la seguridad de las transacciones dentro de un bloque está garantizado, ya que cualquier alteración en los datos provocaría un cambio en la raíz de Merkle, afectando la cadena de hashes y, en consecuencia, invalidando la autenticidad del bloque. [14]

#### Mecanismo de Consenso

En términos generales, los mecanismos de consenso, también conocidos como algoritmos o protocolos de consenso, se refieren a un conjunto de reglas que regulan el flujo de información dentro de una blockchain. Constituyen en el fundamento de cualquier blockchain, ya que determinan quién y cómo se genera un nuevo bloque en la cadena. Existen diversos tipos de mecanismos de consenso, algunos de los más relevantes se describen a continuación. [15]

1. **Prueba de trabajo (PoW - Proof of Work):** Este mecanismo se basa en la resolución de un problema matemático complejo que requiere un alto poder computacional para validar y agregar un nuevo bloque a la blockchain. [11]
2. **Prueba de participación (PoS - Proof of Stake):** En este modelo, la probabilidad de que un usuario genere el próximo bloque está determinada por la cantidad de monedas que posee. A mayor participación en la red, menor es la posibilidad de que un usuario actúe de manera maliciosa. [12]
3. **Prueba de participación delegada (DPoS - Delegated Proof of Stake):** En este enfoque, los participantes de la red seleccionan delegados responsables de validar y añadir nuevos bloques a la cadena. [11]

Cualquier mecanismo de consenso debe cumplir con tres principios fundamentales:

1. *Validez*, garantizando que solo las transacciones legítimas sean aprobadas por nodos autorizados.
2. *Conformidad,* asegurando que todos los nodos honestos lleguen a un acuerdo sobre el estado de la red.
3. *Terminación*, permitiendo que todos los nodos legítimos alcancen una decisión definitiva.

Además, estos algoritmos deben incluir mecanismos de seguridad para proteger la blockchain contra ataques y eventos adversos. [11]

### Smart contracts

La aparición de la tecnología blockchain en la segunda generación demostró que podía usarse más allá de realizar simples pagos, transacciones y transferencias que involucran a sistemas financieros, por tanto, surgen los contratos inteligentes. Esta tecnología se lo puede definir como un programa que contiene un conjunto de instrucciones que se ejecutan de forma automática en cada nodo presente en la red en función del cumplimiento de las condiciones específicas del código. Esto crea confianza entre dos partes que no se conocen entre sí, ya que el código se ejecuta públicamente en la red y es verificable. [4]

#### Estructura de un smart contract

Un contrato inteligente se compone de los siguientes elementos:

1. *Dirección*: Es un identificador único dentro de la blockchain.
2. *Valor*: Representa la cantidad de criptomoneda asociada al contrato.
3. *Funciones*: Son instrucciones programadas que se pueden ejecutar dentro del contrato.
4. *Estado*: Son los datos almacenados, los cuales pueden modificarse dependiendo de la ejecución de las funciones.

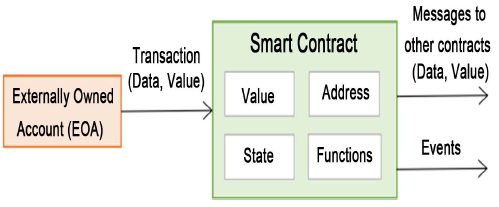


Fig. 3. Estructura de un contrato inteligente

Nota: Bhabendu Kumar [16]

Como se indica en la figura 3 se presenta un ejemplo simple de un contrato inteligente. Este contrato recibe la transacción como entrada, ejecuta el código pertinente y activa los eventos de salida. Según la lógica implementada en la función, los estados del contrato cambian durante su ejecución [16]

### Herramientas tecnológicas

El desarrollo acelerado de la tecnología blockchain ha despertado un creciente interés en la implementación de redes descentralizadas que prioricen la seguridad y la transparencia, este avance ha dado origen a múltiples plataformas diseñadas para la creación y administración de redes blockchain, cada una con características, beneficios y retos particulares. La evolución constante de estas tecnologías responde, en gran parte, a la necesidad de soluciones innovadoras que afronten desafíos relacionados con la confianza, la protección de datos y la optimización de procesos en diversas industrias [17].

Para el desarrollo de la propuesta planteada en este trabajo, se analizaron distintas opciones de redes blockchain, entre las cuales destacan Ethereum (ETH), Hyperledger Fabric (FABRIC), Solana (SOL) y Polkadot (DOT). Cada una de estas plataformas presenta fortalezas y limitaciones que fueron consideradas en el análisis comparativo que se muestra en la Tabla 1.

|  |  |  |
| --- | --- | --- |
| **Red** | **Ventajas** | **Inconvenientes** |
| Ethereum (ETH) | Alta versatilidad para la ejecución de contratos inteligentes.  Plataforma de código abierto, accesible para cualquier desarrollador.  Amplia comunidad y ecosistema de desarrollo.  Seguridad robusta basada en mecanismos de consenso PoW. | Escalabilidad limitada y congestión en la red.  Privacidad estándar, menos adecuada para aplicaciones que requieren confidencialidad. |
| Hyperledger Fabric (FABRIC) | Orientación empresarial con funciones avanzadas.  Control de acceso detallado y privacidad granular.  Buena escalabilidad y eficiencia en entornos corporativos. | Configuración técnica más compleja.  Comunidad y soporte más reducidos en comparación con Ethereum. |
| Solana (SOL) | Alta velocidad de procesamiento (hasta 65,000 TPS).  Bajas tarifas de transacción y alto rendimiento.  Soporte completo para Turing, ideal para aplicaciones complejas. | Tecnología menos madura y con riesgo potencial.  Comunidad y ecosistema en desarrollo, pero aún limitados. |
| Polkadot (DOT) | Interoperabilidad entre múltiples blockchains.  Escalabilidad mejorada mediante el uso de cadenas paralelas. | Menor adopción y madures en comparación con Ethereum.  Mayor complejidad técnica para desarrolladores. |

Tab. 1. Ventajas e inconvenientes de una red blockchain.

Se eligió la red Ethereum debido a la amplia gama de herramientas que proporciona, su flexibilidad para la implementación de contratos inteligentes y el respaldo de una comunidad activa y consolidada. Su alto nivel de adopción y el desarrollo de un ecosistema sólido la posicionan como una de las plataformas más maduras del sector. Adicionalmente, Ethereum permite la creación de contratos inteligentes que operan con su criptomoneda nativa, lo cual facilita la implementación de incentivos económicos para los usuarios que interactúan con el sistema.

Para el desarrollo de la aplicación se consideraron tres entornos de desarrollo integrados (IDE): Remix, Truffle y Visual Studio (VS Code) cada uno de estos entornos ofrece características particulares orientadas al desarrollo, prueba y despliegue de contratos inteligentes [18]. La Tabla 2 detalla un análisis comparativo de sus principales ventajas y limitaciones desde una perspectiva técnica.

|  |  |  |
| --- | --- | --- |
| **IDE** | **Ventajas** | **Inconvenientes** |
| Remix | Plataforma accesible y fácil de usar, disponible de manera online.  Soporte completo para el desarrollo y despliegue de contratos inteligentes en Ethereum.  Amplia documentación y comunidad activa. | Funcionalidades limitadas en comparación con IDEs locales más avanzados.  Dependencia de una conexión a Internet para su uso. |
| Truffle | Potente suite de herramientas para el desarrollo de aplicaciones en Ethereum.  Permite el desarrollo local y pruebas integradas.  Proporciona herramientas avanzadas para testing, migración y despliegue de contratos. | Curva de aprendizaje empinada para desarrolladores novatos.  Requiere configuración local y dependencias adicionales. |
| VS Code | Integración con el popular editor de código Visual Studio Code.  Gran capacidad de personalización y soporte de una amplia comunidad. | Requiere configuración adicional y la instalación de extensiones.  Mayor soporte nativo para Ethereum en comparación con Remix. |

Tab. 2. Ventajas e inconvenientes del entorno de desarrollo integrado

Se seleccionó VS Code debido a su interfaz amigable y su integración directa con la red Ethereum, lo que simplifica significativamente el proceso de desarrollo y depuración de contratos inteligentes, al ser un editor de código fuente multiplataforma permite la escritura, prueba y despliegue de contratos inteligentes en Ethereum de forma eficiente y accesible.

Existen varios lenguajes de programación empleados en el desarrollo de contratos inteligentes. En el marco de esta investigación, se han considerado como opciones principales Solidity y Vyper, cuyas ventajas y desventajas se detallan en la Tabla 3.

|  |  |  |
| --- | --- | --- |
| **Lenguaje** | **Ventajas** | **Inconvenientes** |
| Solidity | Diseñado específicamente para contratos inteligentes en la plataforma Ethereum.  Utiliza un enfoque orientado a objetos, permitiendo la creación de código reutilizable y fácil de mantener.  Amplia adopción y una comunidad activa que respalda su desarrollo.  Integración fluida con herramientas de desarrollo como Remix y Truffle. | Posibles vulnerabilidades y errores de seguridad debido a su complejidad.  Limitaciones al compararlo con lenguajes más maduros. |
| Vyper | Lenguaje más seguro y fácil de comprender en comparación con Solidity.  Diseño enfocado en la simplicidad y la seguridad.  Menos propenso a errores y vulnerabilidades de seguridad. | Menor adopción y una comunidad más reducida en comparación con Solidity.  Limitación en funcionalidades avanzadas, con menos soporte y herramientas de desarrollo disponibles. |

Tab. 3. Ventajas e inconvenientes de los diferentes lenguajes de programación

La opción de Vyper fue descartada debido a su menor adopción y a las restricciones que presenta frente a Solidity. Por otro lado, Solidity es ampliamente utilizado en el ecosistema Ethereum y se integra de forma nativa con herramientas como Remix. Está diseñado específicamente para la creación de contratos inteligentes, lo que lo convierte en la opción más adecuada para este caso de uso.

# Capítulo 3

## Metodología experimental

La metodología es un componente fundamental dentro del proceso de desarrollo de una investigación, según la Real Academia Española la define como la ciencia del método, es decir, el conjunto de procedimientos que se siguen en una investigación científica o en una exposición doctrinal [19]. En este sentido, el término “método” hace referencia a la ruta a seguir a través de un conjunto de pasos, normas y procedimientos previamente establecidos, aplicados de forma consciente y deliberada, con el propósito de alcanzar un determinado fin que puede ser tangible o conceptual [20].

Según Fidias [21], la metodología de investigación aplicada tiene como propósito la producción de nuevos conocimientos con aplicación inmediata a la solución de problemas prácticos. Esta se fundamenta principalmente en los hallazgos tecnológicos de la investigación básica, desempeñando un papel clave en la unión entre la teoría y el producto. Por su parte Lozada [22], señala que este tipo de investigación puede arrojar dos clases de resultados: el primero corresponde al núcleo tecnológico, que engloba el conocimiento general requerido para desarrollar un producto; mientras que el segundo se relaciona con la tecnología específica, donde se materializa un producto o una de sus partes con un valor añadido.

Tomando en cuenta estas consideraciones, se ha decidido emplear esta metodología, ya que se pretende desarrollar un prototipo para la certificación de la culminación de estudios del idioma inglés en el Instituto académico de Idiomas de la Universidad Central del Ecuador. Esta elección permitirá establecer una ruta clara y detallada para el proyecto, facilitando la creación de una solución efectiva y asegurando el cumplimiento de los objetivos planteados.

Por lo tanto, se propone una serie de pasos a seguir para el desarrollo del presente trabajo.

**Propuesta del prototipo:** El primer paso es definir con precisión la funcionalidad que tendrá el prototipo de aplicación. Para comprender el alcance del proyecto, se decidió llevar a cabo una serie de entrevistas con el propósito de conocer el proceso de certificación académica dentro del Instituto Académico de Idiomas. A través de estas, se identificará qué áreas y personas intervienen en dicho proceso, lo que permitirá determinar los requisitos, funcionalidades y características que deberá cumplir el contrato inteligente, así como elaborar el diagrama del modelo con base en la información proporcionada por las partes interesadas.

**Descripción de la arquitectura:** El siguiente paso consiste en llevar a cabo un análisis exhaustivo de las funcionalidades del contrato inteligente para definir los requerimientos de la arquitectura y el entorno de desarrollo. Esto permitirá tener una visión clara de la correcta preparación del entorno de trabajo.

**Selección de herramientas:** Una vez definidos los requerimientos para el desarrollo del proyecto, el siguiente paso es seleccionar un conjunto de herramientas, preferiblemente de software libre, que respalden la arquitectura y el entorno de desarrollo establecidos en la fase inicial.

**Desarrollo del prototipo propuesto:** Con la arquitectura y el entorno de trabajo establecidos, lo siguiente es la creación de los contratos inteligentes, por tanto, esta fase se centrará en la codificación, detallando minuciosamente cada uno de sus componentes. Al final, se describirá el proceso de configuración, así como los pasos necesarios para compilar y desplegar el contrato inteligente en la red de pruebas.

**Interacción con el contrato inteligente:** En la fase final, se llevará a cabo la interacción con los contratos inteligentes mediante una interfaz gráfica, permitiendo su implementación en la red de pruebas. En este apartado, se explicará el desarrollo de la interfaz, sus componentes y la forma en que los usuarios deberán utilizarla para garantizar la validez del certificado generado.

# Capítulo 4

## Cálculos y resultados

### Propuesta del prototipo

Antes de desarrollar el prototipo basado en tecnología blockchain, orientado a la emisión de certificados de culminación de estudios del idioma inglés en el Instituto Académico de Idiomas (IAI) de la Universidad Central del Ecuador, es fundamental identificar a los actores que intervienen directamente, ya que esto permitirá definir con claridad el modelo y diseño del producto propuesto. Una vez establecidos los stakeholders, se procederá a especificar los requerimientos funcionales de cada uno, definir la arquitectura del sistema, seleccionar la plataforma más adecuada para su implementación y, finalmente, se llevará a cabo el desarrollo del prototipo, seguido de pruebas para evaluar su funcionamiento y efectividad.

#### Propósito

Esta sección describe los aspectos clave relacionados con el diseño y desarrollo del prototipo, en función de la "Especificación de Requisitos de Software" (ERS). La aplicación web resultante servirá como una herramienta para gestionar la verificación y validación de la autenticidad de los certificados de suficiencia emitidos por el Instituto Académico de Idiomas, haciendo uso de la tecnología blockchain y contratos inteligentes. Además, la documentación generada ofrece a todos los actores involucrados una visión detallada de la perspectiva del proyecto y una comprensión general del prototipo de software.

#### Alcance

La aplicación web que se desarrollará actuará como un prototipo de demostración de una plataforma descentralizada para la certificación de documentos académicos, la misma que permitirá gestionar el perfil del administrador autorizado encargado de emitir los certificados a los usuarios finales. El Instituto Académico de Idiomas tendrá la capacidad de generar los certificados para su validación y certificación, mientras que los usuarios finales, ya sean estudiantes o instituciones externas podrán acceder a la información y obtener el certificado digital. El propósito principal de este prototipo es eliminar la intermediación de terceros en los procesos de certificación académica del idioma inglés, estableciendo un canal directo entre la institución y los estudiantes. De esta manera, se garantiza la disponibilidad y autenticidad de la información, reforzando la confianza en el sistema de certificación académica.

#### Identificación de procesos

Para comprender el alcance de la aplicación e identificar a los actores clave dentro del sistema, se llevaron a cabo entrevistas en el Instituto Académico de Idiomas de la Universidad Central del Ecuador. Este proceso permitió analizar los principales desafíos asociados a la verificación y certificación académica. En la tabla 4 se presentan los hallazgos obtenidos a partir de estas entrevistas, los cuales sirvieron de base para definir los requisitos funcionales del sistema.

|  |  |
| --- | --- |
| **Dominio de la información** | **Hallazgo** |
| Almacenamiento de la información de los estudiantes | La información de los estudiantes se almacena en la PC del instituto académico. En esta se almacena el historial de solicitudes atendidas y no atendidas. |
| Áreas encargadas de la emisión de certificados académicos. | Inicialmente, el área de TI se encarga de gestionar y administrar la información mediante la plataforma SIIU. Por otro lado, el área de Secretaría genera el certificado, y posteriormente, el área de Dirección valida y autentica el documento. Finalmente, el certificado es enviado al estudiante. |
| Áreas encargadas de la verificación los certificados académicos | En primer lugar, se encuentra el área de admisiones, encargada de hacer el seguimiento de los registros académicos de los estudiantes. Por otro lado, una entidad externa recibe la información necesaria para proceder con la validación notarial del certificado. |
| Problema identificado | Actualmente, en el IAI no existe ningún mecanismo ágil ni seguro para verificar la autenticidad de los certificados digitales. |

Tab. 4. Hallazgos

#### Identificar a los stakeholders

Para identificar a los stakeholders, es fundamental analizar qué actores van a interactuar con el prototipo de la aplicación, esto permitirá determinar el rol de cada participante dentro del sistema y la relación entre ellos. Basándonos en la sección anterior de este documento, se pueden identificar los siguientes actores que se muestra en la tabla 5:

|  |  |
| --- | --- |
| **Rol** | **Descripción** |
| Estudiante | Él es el actor que recibe el certificado emitido por el Instituto Académico de Idiomas. |
| Instituto Académico de Idiomas (Administrador) | Es el actor encargado de emitir y validar los certificados. Tiene la capacidad de generar, almacenar y enviar los certificados tanto al estudiante como a otras dependencias de la universidad. |
| Otras dependencias de la Universidad | Este actor tiene la capacidad de consultar los certificados del estudiante, los cuales han sido previamente validados por el Instituto Académico de Idiomas. |

Tab. 5. Descripción de los actores

#### Modelo propuesto

La gestión de certificados académicos en el Instituto Académico de Idiomas involucra procedimientos administrativos complejos, que frecuentemente requieren la participación de otras dependencias dentro de la Universidad. A través de la adopción de tecnologías emergentes como blockchain y los contratos inteligentes, es posible implementar un modelo descentralizado que proporcione mecanismos de confianza e integridad entre las áreas involucradas en la administración de certificados académicos, como se ilustra en la figura 4. Bajo este enfoque, se propone desarrollar una solución tecnológica que permita la generación digital de certificados académicos para los estudiantes, cuyo registro se realice como una transacción inmutable en una red blockchain.

En este contexto, los usuarios finales podrán interactuar con la plataforma mediante contratos inteligentes, lo que les permitirá obtener, validar y consultar sus certificados de forma autónoma y segura. Este mecanismo garantiza la validación y verificación inmediata de los documentos, permitiendo a los interesados consultar y comprobar su autenticidad con rapidez y fiabilidad.

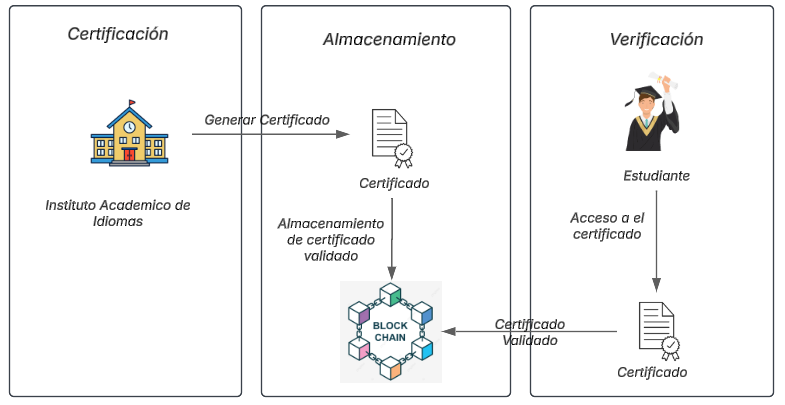


Fig. 4. Modelo propuesto: Certificación descentralizada de documentos académicos.

#### Funciones del prototipo

Este segmento describe las funciones específicas del sistema orientadas en la emisión y validación de certificados académicos, considerando el perfil del usuario que interactúa con él. El enfoque principal es analizar las operaciones que debe realizar el sistema, resaltando la autentificación de los certificados de culminación de un idioma como un caso de uso esencial. El primer caso de uso de nivel cero se presenta en la Figura 5, en la cual se detallan las principales funcionalidades del actor responsable de validar el certificado y del usuario final, en el proceso de generación segura de los certificados académicos.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fig. 5. Funciones del prototipo de software, función nivel cero.

Como se puede visualizar, el prototipo de software está conformado por tres funciones principales.

En la primera funcionalidad se considera la autentificación y autorización para poder acceder a la plataforma, esta funcionalidad permite asegurar el ingreso restringido de usuarios autorizados. Segundo gestionar certificado, es donde el administrador autorizado genera el certificado en la blockchain. Tercero, permite a los usuarios finales verificar la autenticidad del certificado digital, asegurando la disponibilidad y la inmutabilidad de estos.

Estas funcionalidades se presentan en su forma más elemental, cada caso de uso ilustra, cómo el sistema interactúa con los distintos actores proporcionando un escenario representativo con el fin de facilitar una mejor comprensión, dichos casos serán desarrollados a mayor detalle, denominado nivel uno, para ello, se dividirán en funciones más específicas. El primer caso de uso se encuentra representado en la Figura 6.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fig. 6. Diagrama de casos de uso para la autenticación y autorización, función en nivel uno.

El ingreso al sistema por parte del usuario administrador se realiza mediante un mecanismo de autenticación, el cual actúa como punto de control para restringir el acceso a las funcionalidades del sistema. Tras la validación de credenciales, el sistema identifica el perfil del usuario y habilita el acceso a las secciones correspondientes. En el caso del perfil de administrador, se le otorgan permisos completos para gestionar los certificados, mientras que los usuarios con perfil del estudiante acceden únicamente a las funcionalidades relacionadas con la verificación de certificados previamente validados.

|  |  |  |  |
| --- | --- | --- | --- |
| **CU – 01** | **Autenticación y Autorización** | | |
| **Versión** | 1.0 | | |
| **Precondición** | El usuario con perfil de administrador debe acceder a la plataforma para iniciar sesión o completar el proceso de registro. | | |
| **Descripción** | Este caso de uso describe el comportamiento del sistema durante el proceso de autenticación y autorización, permitiendo el acceso controlado. | | |
| **Secuencia normal** | **Caso de uso (Nivel Uno)** | **Acción** | |
| **1** | Iniciar sesión: El usuario inicia el proceso de autenticación ingresando sus credenciales en el formulario correspondiente. | |
| **2** | Registrarse: Si el usuario no posee una cuenta, puede acceder a la opción de registro para crear un perfil dentro de la plataforma. | |
| **3** | Llenar Formulario: El sistema solicita al usuario completar un formulario con los datos requeridos para crear su perfil. | |
| **4** | Recuperar clave: Si el usuario olvido o necesita actualizar su clave, el sistema ofrece un mecanismo para restablecer la contraseña. | |
| **Postcondición** | Una vez autenticado, el usuario podrá gestionar los certificados de los estudiantes. | | |
| **Excepciones:** | **Referencia al caso de uso** | **Acción** | |
| **3** | Si el usuario no completa todos los campos obligatorios del formulario de registro. | |
| **E1** | El sistema muestra un mensaje indicando que todos los campos requeridos deben completarse. |
| **Comentarios** | El administrador podrá crear un perfil desde el cual gestionará la emisión y validación de certificados digitales dentro de la plataforma. | | |

Tab. 6. Autenticación y Autorización

En la figura 7 se presenta el diagrama de caso de uso correspondiente a la funcionalidad de gestión de certificados, asignada al administrador autorizado por el Instituto Académico de Idiomas. Para acceder a estas operaciones, el administrador debe autenticarse previamente en la plataforma. Esta funcionalidad contempla las acciones de generación, almacenamiento, búsqueda y visualización de certificados de culminación de un idioma.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fig. 7. Diagramas de caso para gestionar los certificados, función en nivel uno.

La Tabla 7 describe la funcionalidad del sistema relacionada con la gestión de certificados académicos digitales. En este contexto, el administrador autorizado posee la facultad y la responsabilidad de generar nuevos certificados y consultar el certificado de un estudiante en específico garantizando la integridad, disponibilidad y autenticidad de estos documentos.

|  |  |  |  |
| --- | --- | --- | --- |
| **CU – 02** | **Gestionar Certificado** | | |
| **Versión** | 1.0 | | |
| **Dependencias** | CU-01 | | |
| **Precondición** | El administrador debe estar autenticado en la plataforma. | | |
| **Descripción** | Este caso de uso describe la funcionalidad de la plataforma que permite al administrador generar, almacenar, buscar y visualizar certificados académicos. | | |
| **Secuencia normal** | **Caso de uso (Nivel Uno)** | **Acción** | |
| **1** | **Generar certificado:** El administrador ingresa los datos del estudiante para generar el certificado académico correspondiente. | |
| **2** | **Almacenar certificado:** El sistema almacena el hash del certificado en la blockchain y los datos relevantes en la base de datos del sistema. | |
| **3** | **Buscar certificado:** La plataforma permite al administrador realizar búsquedas de certificados almacenados en la red blockchain. | |
| **4** | **Visualizar certificado:** El administrador puede visualizar todos los certificados generados dentro de la plataforma. | |
| **Postcondición** | El administrador carga los certificados y genera su certificación en la blockchain, asegurando su integridad y disponibilidad. | | |
| **Excepciones:** | **Referencia al caso de uso** | **Acción** | |
| **3** | Si el administrador intenta generar un certificado duplicado. | |
| **E1** | El sistema informa que el certificado no ha sido emitido y evita la duplicación. |
| **Comentarios** | Este caso de uso permite al administrador gestionar totalmente los certificados garantizando la integridad, disponibilidad y autenticidad de la información. | | |

Tab. 7. Gestionar certificados

En la Figura 8 se presenta el diagrama de caso de uso asociado al usuario final, correspondiente a la función de validación de certificados emitidos por el Instituto Académico de Idiomas. Esta funcionalidad permite al usuario acceder a la plataforma, consultar su certificado, visualizarlo, descargarlo y compartirlo. Las acciones vinculadas a esta operación, según el perfil del usuario final, se representan en dicho esquema.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fig. 8. Diagramas de caso para validar los certificados, función en nivel uno.

La Tabla 8 describe el caso de uso asociado a la validación de certificados digitales por parte del usuario final, quien, mediante la introducción de su número de cédula, puede verificar la autenticidad de su certificado, visualizarlo, descargarlo en formato PDF o compartirlo a través de un enlace. Este proceso se realiza de manera eficiente y segura, garantizando la disponibilidad, integridad y autenticidad del certificado dentro de la plataforma.

|  |  |  |  |
| --- | --- | --- | --- |
| **CU – 03** | **Validar Certificado** | | |
| **Versión** | 1.0 | | |
| **Precondición** | El usuario final debe tener su número de cedula. | | |
| **Descripción** | Este caso de uso describe la funcionalidad destinada al usuario final para validar certificados académicos emitidos por el Instituto Académico de Idiomas. La operación contempla la consulta, visualización, descarga y opción de compartir el certificado digital. | | |
| **Secuencia normal** | **Caso de uso (Nivel Uno)** | **Acción** | |
| **1** | **Consultar certificado:** El usuario accede a la plataforma e introduce los datos requeridos para localizar su certificado. | |
| **2** | **Descargar certificado:** El usuario puede descargar el documento digital certificado desde la plataforma en formato PDF. | |
| **3** | **Compartir certificado:** La plataforma ofrece opciones para compartir el certificado a través de medios digitales. | |
| **Postcondición** | El usuario accede exitosamente a su certificado, pudiendo visualizarlo, descargarlo o compartirlo, validando su autenticidad. | | |
| **Excepciones:** | **Referencia al caso de uso** | **Acción** | |
| **3** | Si el usuario ingresa datos incorrectos o el certificado no está disponible en el sistema. | |
| **E1** | El sistema informa al usuario que no se ha encontrado un certificado asociado a los datos ingresados. |
| **Comentarios** | Este caso de uso permite al usuario final verificar directamente sus documentos emitidos por el Instituto Académico de Idiomas, asegurando su disponibilidad y autenticidad. | | |

Tab. 8. Validar Certificado

#### Requisitos funcionales

Los requisitos funcionales definen las acciones específicas que el sistema debe ejecutar para cumplir con los objetivos del proyecto. A partir del análisis de los casos de uso, estos requisitos han sido identificados, clasificados e indexados mediante etiquetas que facilitan su referencia y comprensión. La Tabla 9 resume las funcionalidades clave previstas para la plataforma, estableciendo la relación directa con los casos de uso correspondientes.

|  |  |  |
| --- | --- | --- |
| Requerimiento | Descripción | Caso de uso |
| RF-1 | El sistema debe permitir que un usuario autorizado inicie sesión en la plataforma, siempre que se encuentre previamente registrado. | CU-01-1 |
| RF-2 | El sistema debe ofrecer al administrador la opción de registrarse en la plataforma. | CU-01-2 |
| RF-3 | El sistema debe presentar al usuario un formulario de registro con los campos requeridos para crear su perfil. | CU-01-3 |
| RF-4 | El sistema debe permitir al administrador recuperar y restablecer su contraseña en caso de olvido o actualización. | CU-01-4 |
| RF-5 | El sistema debe permitir al administrador ingresar los datos necesarios del estudiante para la generación del certificado. | CU-01-5 |
| RF-6 | El sistema debe almacenar el hash del certificado generado en la blockchain y los datos en la base de datos. | CU-02-1 |
| RF-7 | El sistema debe permitir al administrador buscar certificados académicos dentro de la plataforma. | CU-02-2 |
| RF-8 | El sistema debe permitir al administrador visualizar el listado de certificados generados. | CU-02-3 |
| RF-9 | El sistema debe permitir al usuario final consultar la existencia y validez de su certificado digital. | CU-02-4 |
| RF-10 | El sistema debe permitir al usuario final visualizar el contenido de su certificado digital. | CU-03-1 |
| RF-11 | El sistema debe permitir al usuario final descargar su certificado en formato PDF. | CU-03-2 |
| RF-12 | El sistema debe permitir al usuario final compartir un enlace de acceso a su certificado validado. | CU-03-3 |

Tab. 9. Requisitos funcionales

#### Requisitos No funcionales

|  |  |
| --- | --- |
| Requisito | Descripción |
| RNF-1 | Para fines de desarrollo y pruebas, la plataforma será implementada y desplegada en un entorno local controlado. |
| RNF-2 | La interfaz de usuario debe ofrecer una experiencia intuitiva, clara y fácil de usar, priorizando la simplicidad en la navegación. |
| RNF-3 | La plataforma debe estar optimizada para minimizar los tiempos de procesamiento y la carga del servidor al momento de visualizar un certificado. |
| RNF-4 | El sistema debe proporcionar una experiencia fluida y sin latencias significativas durante la descarga de certificados digitales. |
| RNF-5 | La arquitectura del sistema debe permitir un mantenimiento sencillo y actualizaciones periódicas, garantizando la continuidad operativa y la corrección eficiente de errores. |
| RNF-6 | Toda información sensible, incluyendo datos personales y certificados, debe ser almacenada y transmitida mediante mecanismos de cifrados seguros (como HTTPS y algoritmos de encriptación estándar). |

Tab. 10. Requisitos no funcionales

El cumplimiento de los requisitos no funcionales de la tabla 10 es fundamental para garantizar la eficiencia operativa, la seguridad de los datos y la escalabilidad del sistema. Estos criterios complementan los requisitos funcionales al asegurar que la plataforma no solo cumpla con sus objetivos técnicos, sino que también lo haga de manera segura, confiable y con una experiencia de usuario optimizada.

### Descripción de la arquitectura

Para el desarrollo de una aplicación descentralizada (DApp) orientada al ámbito educativo, se han considerado elementos clave como la tecnología blockchain, los contratos inteligentes, la descentralización y la certificación digital. La solución propuesta se estructura en una arquitectura de múltiples capas, lo que permite una integración eficiente de estas tecnologías.

Capa de presentación: Esta capa corresponde a la interfaz gráfica del sistema, permitiendo la interacción del usuario con la DApp. Para su desarrollo se emplean tecnologías como React.js que permite crear interfaces dinámicas, Tailwind CSS que facilita un diseño responsivo y Vite.js, encargado de la configuración del entorno del cliente web. Además, Metamask actúa como puente entre la DApp y los navegadores convencionales, permitiendo al usuario interactuar con la blockchain de forma sencilla y segura.

Capa lógica o de negocio: En esta capa se gestiona la lógica principal de la aplicación y su comunicación con la base de datos, por lo tanto, se implementa Node.js junto con el framework Express.js permitiendo un desarrollo robusto del backend. La persistencia de datos se maneja mediante MongoDB, una base de datos NoSQL utilizada para guardar información relevante de los usuarios y sus certificaciones académicas.

Capa blockchain: Esta capa es responsable de la gestión descentralizada de las operaciones críticas de la aplicación. Se implementan contratos inteligentes utilizando el lenguaje Solidity, desplegados sobre la red Ethereum, que garantizan la inmutabilidad, trazabilidad y seguridad en la emisión y validación de certificados. Para fines de desarrollo y pruebas, se hace uso de Hardhat, una blockchain local que permite la simulación de un entorno Ethereum controlado, lo cual facilita el despliegue y validación de los contratos.

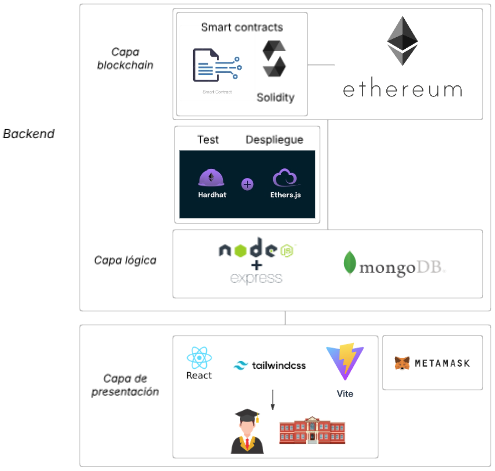


Fig. 9. Arquitectura propuesta de la aplicación.

### Selección de herramientas

A partir del prototipo propuesto, así como de la arquitectura de aplicación y las plataformas expuestas en los capítulos anteriores, se ha elegido un conjunto específico de herramientas para el desarrollo de la plataforma descentralizada orientada al sector educativo. A continuación, se describirán detalladamente las herramientas seleccionadas.

#### Node.js

Es un entorno de ejecución para JavaScript que se construye sobre un motor V8 de Google Chrome, que se caracteriza por su arquitectura no bloqueante y se basa en eventos, lo que le permite gestionar un gran número de conexiones simultáneas de manera eficiente, además cuenta con un repositorio de paquetes conocidos como npm (Node Package Manager), que brinda acceso a miles de bibliotecas y herramientas desarrolladas por la comunidad [23].

#### React.js

Es una biblioteca de JavaScript de código abierto que ofrece una gran flexibilidad en el desarrollo de interfaces de usuario. Su arquitectura se basa en el uso de componentes reutilizables, cada uno con una función específica, lo que permite construir aplicaciones de forma modular y escalable. Estos componentes pueden gestionar sus propios estados, facilitando la creación de interfaces más organizadas, dinámicas y fáciles de mantener [24].

#### Tailwind CSS

Es una herramienta basada en CSS que facilita la creación ágil de interfaces de usuario personalizadas. Su alto nivel de configuración brinda al desarrollador los recursos necesarios para diseñar de forma modular y sencilla. Tailwind ha sido utilizado para implementar todo el CSS en la aplicación [25].

#### Vite JS

Es una herramienta diseñada para acelerar el desarrollo de proyectos web modernos. Su principal ventaja es la capacidad de configurar de forma ágil y sencilla el entorno necesario para una aplicación web, eliminando muchos de los pasos y requisitos habituales de otros métodos tradicionales. En este caso, se ha utilizado para establecer la configuración del cliente web de la aplicación [26].

#### Metamask

Extensión para navegadores web diseñada para facilitar la interacción entre las aplicaciones descentralizadas (DApp) y el entorno del navegador. Su propósito principal es actuar como un intermediario entre el usuario y la red blockchain, permitiendo la gestión segura de cuentas y la visualización de los criptoactivos asociados a una entidad [27].

#### Express JS

Es uno de los frameworks backend más utilizados en el entorno de desarrollo con Node.js, diseñado con el propósito de facilitar la construcción de aplicaciones web. Se destaca por su simplicidad y flexibilidad, al ofrecer una amplia variedad de herramientas preconfiguradas que optimizan el rendimiento y agilizan el proceso de desarrollo [28].

#### MongoDB

Sistema de gestión de bases de datos NoSQL que almacena la información en colecciones de documentos, utilizando específicamente el formato BSON (Binary JSON). Gracias a su arquitectura altamente flexible, MongoDB se posiciona como una solución robusta para la gestión de datos, ofreciendo capacidades avanzadas de escalabilidad, consistencia, tolerancia a fallos, agilidad y flexibilidad. Estas características lo convierten en una herramienta ideal para entornos que requieren un desarrollo ágil y operaciones con mínima interrupción del servicio [28].

#### Hardhat

Es un entorno de desarrollo diseñado específicamente para Ethereum, que simplifica la compilación y despliegue de contratos inteligentes escritos en el lenguaje Solidity, además, ofrece una red de prueba local que permite ejecutar transacciones simuladas de manera segura utilizando tokens ficticios de Ethereum, lo que facilita el desarrollo y la validación sin riesgos [29].

#### Ethers JS

Es una biblioteca escrita en JavaScript que facilita la interacción con la blockchain de Ethereum. Proporciona todas las herramientas necesarias para comunicarse tanto con proveedores de billeteras como con servicios de nodos de Ethereum, simplificando el desarrollo de aplicaciones descentralizadas [26].

### Desarrollo del prototipo propuesto

#### Certificación académica LangCertIAI

Antes de iniciar la fase de desarrollo, se ha decidido nombrar al prototipo de aplicación descentralizada como LangCertIAI, el nombre surge de la combinación de las siguientes siglas: Lang: abreviatura de language (idioma), Cert: derivado de certification (certificación), IAI: siglas del Instituto Académico de Idiomas. LangCertIAI representa una nueva forma de gestionar certificados digitales de competencias lingüísticas, validada en cadena de bloques para garantizar autenticidad, integridad y confianza académica.

#### Estructura del proyecto

El proyecto contiene la siguiente estructura de carpetas:

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 10. Estructura de directorios del proyecto

La organización del proyecto se divide en tres carpetas principales. La primera, denominada fronend, la cual contiene la interfaz gráfica con la que interactúa el usuario. Esta interfaz fue desarrollada utilizando la herramienta Vite, Tailwind CSS junto con el framework React, y está programada en JavaScript. La segunda carpeta, llamada backend establece la conexión con el contrato inteligente desplegado en la blockchain local de la red Hardhat. La tercera carpeta llamada smart-contract, alberga el contrato inteligente y los archivos generados a partir de su desarrollo. Este contrato fue creado con la herramienta Hardhat y está escrito en el lenguaje Solidity.

#### Desarrollo del contrato inteligente

La Figura 11 muestra el proceso seguido para la construcción del contrato inteligente, este proceso inicia con la fase de codificación, donde se define la estructura de datos, variables, eventos y funciones que compondrán la lógica del contrato, una vez finalizada esta etapa, el siguiente paso es compilar y desplegar el contrato utilizando la herramienta Hardhat. Es fundamental que la compilación esté libre de errores para poder avanzar a la siguiente fase, que es el despliegue del contrato en la red de pruebas local proporcionada por Hardhat. Una vez que el contrato ha sido correctamente implementado en dicha red, se desarrolla una interfaz web para facilitar la interacción con el contrato, permitiendo así la ejecución y validación de sus funcionalidades dentro de un entorno simulado.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fig. 11. Esquema de fases para el desarrollo del contrato inteligente

**Configuración del contrato**

Inicialmente, el directorio denominado *smart-contract* se encuentra vacío, por lo que se utilizará *Hardhat* para generar la estructura básica necesaria para el desarrollo del contrato inteligente. Para ello, desde una terminal de comandos, se accede al directorio */smart-contract* y se ejecuta el siguiente conjunto de instrucciones:

*npm init -y*: Inicializa un nuevo proyecto Node.js generando automáticamente el archivo *package.json* con valores predeterminados.

*npm install*: Instala las dependencias listadas en el archivo *package.json*.

npm install dotenv @nomicfoundation/hardhat-toolbox: permite instalar dos dependencias, *dotenv*: Carga las variables de entorno desde el archivo *.env* para proteger información sensible como claves privadas o las URLs de los nodos y *@nomicfoundation/hardhat-toolbox*: Proporciona un conjunto de herramientas esenciales para el entorno de desarrollo Hardhat que incluye plugins como ethers.js, waffle, chai, entre otros, facilitando tareas como pruebas, despliegues y depuración.

*npm install --save-dev Hardhat*: Instala Hardhat como dependencia de desarrollo para compilar, desplegar, probar e interactuar con contratos inteligentes de Ethereum.

*npm install ethers*: Este paquete permite interactuar con contratos inteligentes desde JavaScript, es ampliamente utilizada junto con Hardhat para el manejo de cuentas, transacciones y lectura o escritura de datos en la blockchain.

**Inicialización y ejecución de Hardhat**

Una vez instaladas las dependencias, se procede a ejecutar el siguiente comando:

*npx hardhat*: Muestra un asistente interactivo que guía al usuario en la configuración inicial, en el cual se debe seleccionar la opción “Crear un proyecto básico”, para genera una estructura predeterminada de carpetas y archivos como *contracts/, scripts/, test/, y hardhat.config.js* que sirven de base para el desarrollo del contrato inteligente, esta estructura se representa en la Figura 12.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 12. Estructura de directorios de la carpeta smart-contract

El contrato que se desarrollará se ubicará dentro del directorio *contracts* y el directorio scripts contiene el archivo *deploy.js*, el cual será utilizado para compilar y desplegar el contrato en la red blockchain local proporcionada por *Hardhat*, el contenido de dicho archivo se presenta en la Figura 13, este archivo de *JavaScript* despliega el contrato *LanguageCertificate.sol* en la red blockchain de prueba, obtiene su dirección y su ABI (Application Binary Interface) y los guarda en archivos JSON para que puedan ser utilizados en capa de backend.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 13. Código fuente del archivo deploy.js

El archivo *hardhat.config.js*, mostrado en la figura 14, es fundamental para el correcto funcionamiento de *deploy.js*, ya que define la configuración del entorno de trabajo de Hardhat, en este archivo se especifica el uso del paquete *@nomicfoundation/hardhat-toolbox*, el cual proporciona plugins como *hardhat-ethers*, que permite la interacción con contratos mediante *Ethers.js*; *hardhat-waffle*, para la realizar pruebas; y *hardhat-etherscan*, que facilita la verificación de contratos en Etherscan. Además, se define que los contratos serán compilados utilizando la versión 0.8.28 del compilador de Solidity.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 14. Código fuente del archivo hardhat.config.js

El archivo package.json, mostrado en la figura 15 contiene las dependencias principales como las de desarrollo, en la sección *dependencies* se incluye paquetes esenciales como *@nomicfoundation/hardhat-toolbox*, que agrupa herramientas útiles para el entorno de Hardhat; *ethers*, una biblioteca JavaScript para interactuar con la blockchain de Ethereum; y *dotenv*, utilizada para gestionar variables de entorno de manera segura, y en la sección *devDependencies*, se incluye *hardhat*, la herramienta principal del proyecto para compilar, desplegar y probar contratos inteligentes.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 15. Código fuente del archivo hardhat.config.js

#### Codificación del contrato inteligente

Como primer paso, se crea un contrato inteligente dentro de la carpeta *contracts*, este contrato se guarda en un archivo con extensión .*sol*, como se muestra en la Figura 16, esta extensión corresponde a **Solidity**, el lenguaje de programación utilizado para desarrollar contratos inteligentes en la red Ethereum.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fig. 16. Contrato inteligente llamado LanguageCertificate.sol

A continuación, es necesario especificar la versión del compilador de Solidity, actualmente la versión 0.8 de Solidity es considerada estable y ampliamente utilizada, esto garantiza que las futuras subversiones dentro de la misma serie mantendrán compatibilidad con el contrato, incluso si en algún momento se requiere actualizar o modificar su código, como se muestra en la Figura 17.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Fig. 17. Especificación de la versión de Solidity

**Estructura de datos**

Solidity permite definir estructuras de datos personalizadas mediante la palabra reservada *struct*. En este caso, se crea una estructura llamada *Certificate* que contiene los campos necesarios para representar un certificado de idiomas. Tal como se observa en la Figura 18, los campos definidos son los siguientes: *instituteName* para almacenar el nombre del instituto, *facultyName* para el nombre de la facultad, *studentName* para el nombre del estudiante, *cedula* para el número de identidad (clave única), *language* para idioma que curso, *level* nivel aprobado (A1, B2, etc.), *issueDate* fecha de emisión del certificado y *valid* si el certificado es válido o no (booleano).

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 18. Estructura de datos del certificado

**Mapeo del certificado**

Para almacenar los certificados, se utiliza un *mapping*, que actúa como una tabla de asociación entre la cédula del estudiante y el certificado correspondiente. Esto permite acceder fácilmente a los datos desde fuera del contrato, como se ilustra en la Figura 19.



Fig. 19. Mapeo del certificado

**Funciones del contrato**

Las funciones representan bloques de código ejecutable dentro del contrato inteligente y pueden invocarse tantas veces como sea necesario. Se definen utilizando la palabra reservada *function*, seguida del nombre de la función y una lista de parámetros de entrada. También es necesario declarar la visibilidad de la función (public, private, internal o external), y si la función debe retornar un valor, se debe especificar su tipo y utilizar la palabra clave *return*.

La Figura 20 muestra la función *issueCertificate*, la cual crea y registra un nuevo certificado, esta función recibe los siguientes parámetros \_instituteName: Nombre del instituto que emite el certificado, \_facultyName: Facultad, \_studentName: Nombre del estudiante, \_cedula: Documento de identidad del estudiante, \_language: Idioma que certifica, \_level: Nivel del idioma (A1, B2, C1, etc.).

Una vez recibidos los datos, la función crea un nuevo objeto de tipo *Certificate*, le asigna los valores proporcionados, registra la fecha de emisión utilizando *block.timestamp*, establece el estado como válido (true), y finalmente almacena este certificado en el mapping, usando la cédula como clave de acceso.

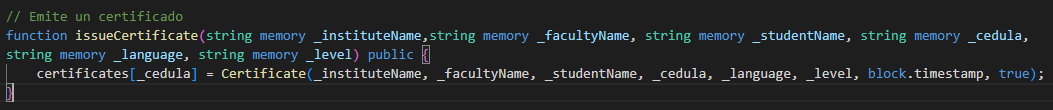


Fig. 20. Función emitir certificado.

El contrato principal *LanguageCertificate* cuenta con varias funciones que deben ser implementadas correctamente para garantizar su funcionamiento. La utilidad de cada una de estas funciones se detalla en la Tabla 11.

|  |  |  |
| --- | --- | --- |
| **Nombre de la función** | **Tipo de dato que retorna** | **Descripción** |
| issueCertificate | *void* (no retorna datos) | Emite un nuevo certificado de idioma para un estudiante, almacenando sus datos en un registro *(mapping)* llamado *certificates*, donde la clave es la cédula del estudiante |
| verifyCertificate | bool | Permite verificar la validez de un certificado consultando su cédula. Retorna el valor del campo valid: true si el certificado está activo, false si fue invalidado. |
| invalidateCertificate | *void* (no retorna datos) | Invalida un certificado previamente emitido, cambiando el valor del estado de *valid* a false. |
| updateLevel | *void* (no retorna datos) | Actualiza el nivel de idioma de un certificado ya emitido, utilizando la cédula del estudiante como clave. |
| getCertificate | Certificate *memory* | Retorna una estructura completa de tipo *Certificate* con todos los datos del certificado asociado a una cédula específica. |
| getIssueDate | uint256 | Permite conocer cuándo fue emitido un certificado para verificar su vigencia, antigüedad o caducidad. |

Tab. 11. Funciones del contrato LanguageCertificate

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 21. Implementación de las funciones en el contrato LanguageCertificate

**Configuración del Backend**

Para comenzar el desarrollo del servidor nos ubicamos dentro del archivo backend, se inicializa un proyecto *Node.js* utilizando el comando *npm init -y*, este comando genera automáticamente un archivo *package.json* para gestionar las dependencias necesarias para el desarrollo del backend, esto se muestra en la figura 22.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fig. 22. Estructura de directorios de la carpeta backend

**Instalación de dependencias**

A continuación, se instalan las principales bibliotecas que permitirán construir la API y conectar con la base de datos: *npm install express mongoose dotenv cors jsonwebtoken pdfkit bcryptjs*, el uso de estas se detalla en la tabla 12.

|  |  |
| --- | --- |
| **Librería** | **Descripción** |
| express | Permite crear servidores y gestionar rutas HTTP. |
| mongoose | para interactuar con bases de datos MongoDB. |
| dotenv | Permite manejar variables de entorno definidas en un archivo .env. |
| cors | Permite habilitar solicitudes entre diferentes dominios (Cross-Origin Resource Sharing). |
| jsonwebtoken | Genera y verifica tokens JWT para autenticación segura. |
| pdfkit | Permite generar archivos PDF de manera dinámica desde el backend. |
| bcryptjs | Utilizado para cifrar contraseñas antes de almacenarlas en la base de datos. |

Tab. 12. Dependencias del archivo backend.

**Configuración de la base de datos MongoDB**

Primero, se crea un archivo de configuración llamado .env ubicado fuera del código fuente principal, tal como se muestra en la Figura 23, este archivo almacena variables de entorno que son leídas por la aplicación en tiempo de ejecución mediante la biblioteca dotenv.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fig. 23 Archivo de configuración .env

En la Figura 24 se muestra el contenido del archivo .env utilizado en este proyecto, donde PORT: Define el puerto en el cual se ejecutará el servidor Express, MONGO\_URI: Contiene la cadena de conexión a la base de datos *Certificados\_idiomas,* JWT\_SECRET: Clave secreta para la autenticación segura de usuarios, PRIVATE\_KEY: Representa la clave privada usada para interactuar con el contrato inteligente desplegado en la red blockchain.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 24. Contenido del archivo .env

A continuación, se crea el archivo *db.js* dentro de la carpeta *config*, para conectar el servidor con la base de datos, por lo cual, se utiliza la biblioteca *mongoose*, la cual facilita la interacción con MongoDB mediante un modelo de datos orientado a objetos (ODM). En la Figura 25 se muestra un fragmento de código que define una función asíncrona llamada *connectDB*, cuya función principal es gestionar y establecer dicha conexión de forma eficiente y segura.

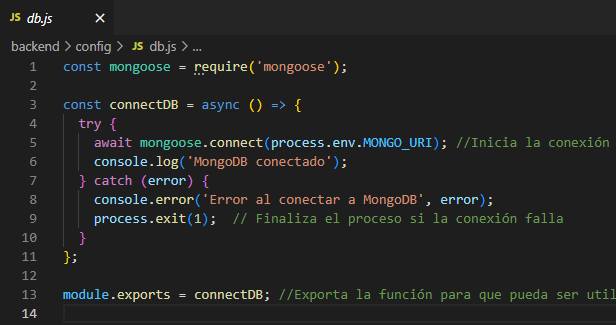


Fig. 25. Conexión con la base de datos MongoDB

**Estructura para almacenar los certificados**

Para almacenar la información de los certificados de idioma en la base de datos MongoDB, se crea un modelo llamado *Certificate.js* ubicado dentro de la carpeta *models*, en este archivo se define un esquema denominado *certificateSchema*, que representa la estructura lógica que seguirán todos los certificados almacenados. Este esquema está diseñado para incluir todos los campos necesarios que describen un certificado de idioma, tales como el nombre del instituto, la facultad correspondiente, los datos del estudiante, el idioma certificado, el nivel alcanzado, la fecha de emisión y el estado de validez como se presenta en la figura 26.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 26. Estructura para almacenar los certificados en la base de datos

**Estructura para almacenar los usuarios**

Para gestionar los usuarios dentro del sistema, se define un modelo denominado *User.js*, ubicado en la carpeta *models*, este modelo utiliza la biblioteca Mongoose para estructurar los datos y facilitar su interacción con la base de datos MongoDB. Además, incorpora la biblioteca *bcryptjs* para asegurar la confidencialidad de las contraseñas mediante técnicas de encriptación, esto se puede ver en la figura 27. Por último, el modelo incluye un método denominado *comparePassword*, que permite comparar una contraseña proporcionada por el usuario durante el proceso de autenticación con la contraseña encriptada almacenada, garantizando así una validación segura y eficiente.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 27. Estructura para almacenar los certificados en la base de datos

**Definición de rutas de autenticación de usuarios**

El archivo *authRoutes.js*, ubicado en la carpeta *routes*, define las rutas relacionadas con la autenticación y el registro de usuarios dentro del sistema. Utiliza el framework Express para crear un enrutador que gestiona las peticiones HTTP correspondientes a estas funcionalidades como muestra la figura 28.

La ruta *POST /register* permite registrar nuevos usuarios en el sistema. Al recibir la solicitud, se extraen del cuerpo (req.body) los campos email, password, cedula y role. Luego, se crea una nueva instancia del modelo User.js y se guarda en la base de datos.

La ruta *POST /login* permite autenticar a los usuarios. Se verifica si existe un usuario con el correo proporcionado y si la contraseña ingresada coincide con la registrada. Si la autenticación es exitosa, se genera un token JWT (JSON Web Token) firmado con una clave secreta definida en el archivo *.env*. Este token contiene el ID y el rol del usuario, y será utilizado posteriormente para autorizar el acceso a rutas protegidas.

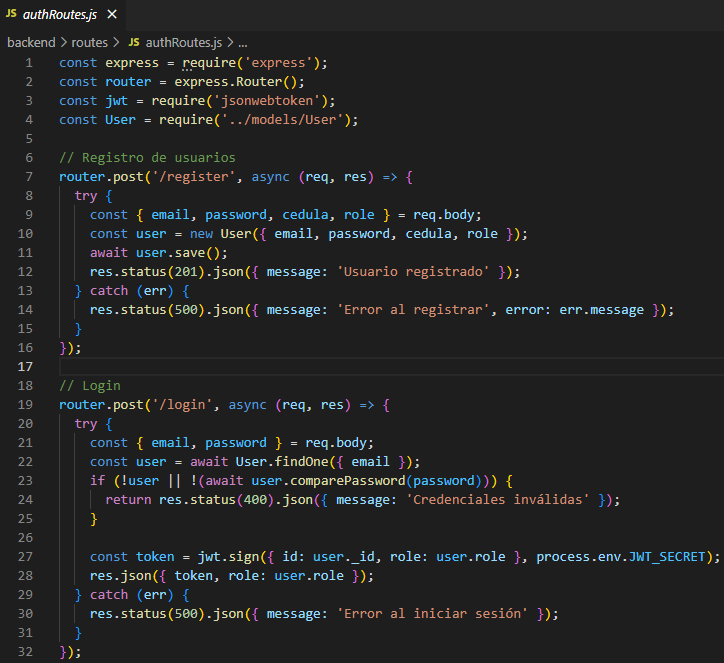


Fig. 28. Autenticación de los roles de los usuarios.

**Definición de rutas para la gestión de certificados**

El archivo *certificateRoutes.js*, ubicado dentro de la carpeta *routes*, contiene la definición de las rutas que permiten gestionar los certificados de idioma. Estas rutas utilizan el framework Express y están vinculadas a funciones específicas definidas en el archivo *certificateController.js*, el contenido del archivo lo podemos visualizar en la figura 29.

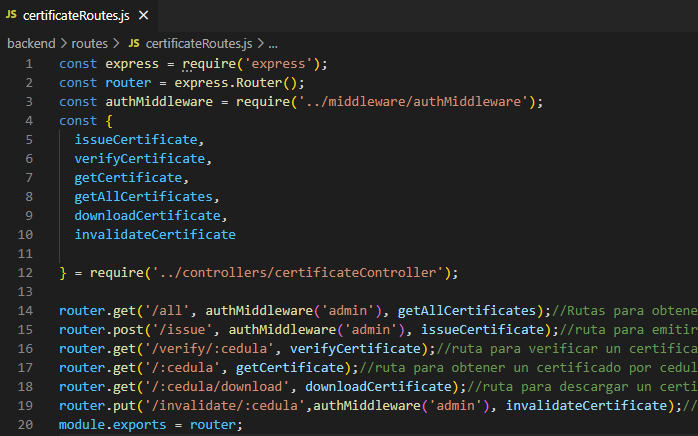


Fig. 29. Rutas para la gestión de certificados

Además, se incorpora un *middleware* de autenticación *authMiddleware.js* que restringe el acceso a determinadas rutas, asegurando que solo usuarios con el rol de administrador puedan ejecutar ciertas operaciones como se muestra en la tabla 13.

|  |  |  |
| --- | --- | --- |
| **Ruta** | **Rol** | **Descripción** |
| GET /all | Administradores | Permite obtener la lista completa de certificados almacenados. |
| POST /issue | Administradores | Permite emitir un nuevo certificado al registrar la información del estudiante. |
| GET /verify/:cedula | Administrador/usuario | Permite verificar si un certificado es válido o ha sido invalidado, consultando mediante el número de cédula. |
| GET /:cedula | Administrador/usuario | Permite obtener todos los datos de un certificado específico, buscando por la cédula del estudiante. |
| GET /:cedula/download | Administrador/usuario | Ofrece la posibilidad de descargar el certificado en formato PDF. |
| PUT /invalidate/:cedula | Administrador | Permite invalidar un certificado previamente emitido, cambiando su estado a no válido. |

Tab. 13. Tabla de restricción de acceso a determinadas rutas

La figura 30 muestra el contenido del archivo *authMiddleware.js* utilizado para proteger rutas específicas dentro de la aplicación. Su propósito es verificar que el usuario esté autenticado mediante un token JWT (JSON Web Token) y que posea el rol adecuado para acceder a ciertos recursos.



Fig. 30. Contenido del archivo authMiddleware.js autentificación

**Lógica de negocio**

La figura 31 muestra el contenido del archivo certificateController.js, que contiene toda la lógica del servidor relacionada con los certificados de idioma. Actúa como el controlador principal que maneja las solicitudes (requests) que llegan al backend sobre certificados, conectando la base de datos (MongoDB), la generación de PDF y la blockchain (Ethereum).

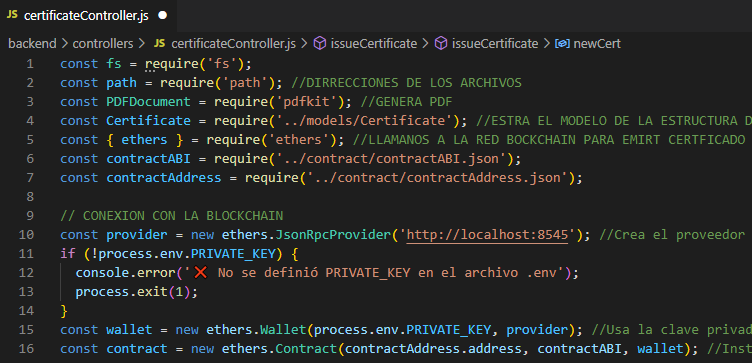


Fig. 31. Contenido del archivo certificateController.js

La Tabla 14 muestra un resumen de todas las funciones implementadas para gestionar la lógica relacionada con los certificados de idioma. Dichas funciones se encuentran definidas dentro del archivo certificateController.js, el cual centraliza el control de las operaciones como la emisión, verificación, descarga, consulta e invalidación de certificados.

|  |  |
| --- | --- |
| **Función** | **Descripción** |
| issueCertificate | Emite y guarda un certificado en blockchain y MongoDB |
| verifyCertificate | Verifica si el certificado es válido o no es valido |
| getCertificate | Busca y devuelve todos los datos de un certificado por cédula. |
| getAllCertificates | Muestra todos los certificados (solo admin) |
| downloadCertificate | Genera un PDF con los datos del certificado |
| invalidateCertificate | Marca el certificado como no válido (revoca) |

Tab. 14. Funciones del archivo certificatecontrolle.js

A continuación, se presenta la implementación de estas funciones en las siguientes imágenes.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 32. Emitir certificado

Pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Fig. 33. Verificar validez

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Fig. 34. Obtener un certificado

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 35. Obtener todos los certificados

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 36. Generar PDF

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 37. Invalidar certificado

Por último, la figura 38 muestra el archivo *auth.controller.js*, ubicado en la carpeta de *controllers*, implementa la lógica necesaria para el registro de nuevos usuarios en la plataforma. Su función principal es validar que no exista previamente un correo electrónico registrado y, en caso contrario, crear y almacenar un nuevo usuario en la base de datos MongoDB utilizando el modelo *User.js*. Esta funcionalidad garantiza un control adecuado sobre la autenticación y el acceso al sistema.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 38. Contenido del archivo auth.controler.js

#### Copilar y desplegar el contrato

En esta etapa se procede a compilar el contrato inteligente para asegurarse de que su sintaxis y lógica son correctas, este proceso permite que las funciones definidas en el contrato estén disponibles para ser invocadas desde la aplicación, estableciendo la base sobre la cual se registrarán y validarán los certificados emitidos.

Antes de desplegar el contrato inteligente, es necesario contar con una red blockchain activa. Para ello, *Hardhat* permite ejecutar un nodo local que simula una red de Ethereum ejecutando el comando *npx hardhat node* que permite levantar un nodo *localhost.* Al iniciarse, genera varias cuentas de prueba preconfiguradas con una gran cantidad de Ether ficticio, estas cuentas se pueden utilizar para desplegar contratos sin necesidad de una red pública, esto podemos observar en la figura 39.

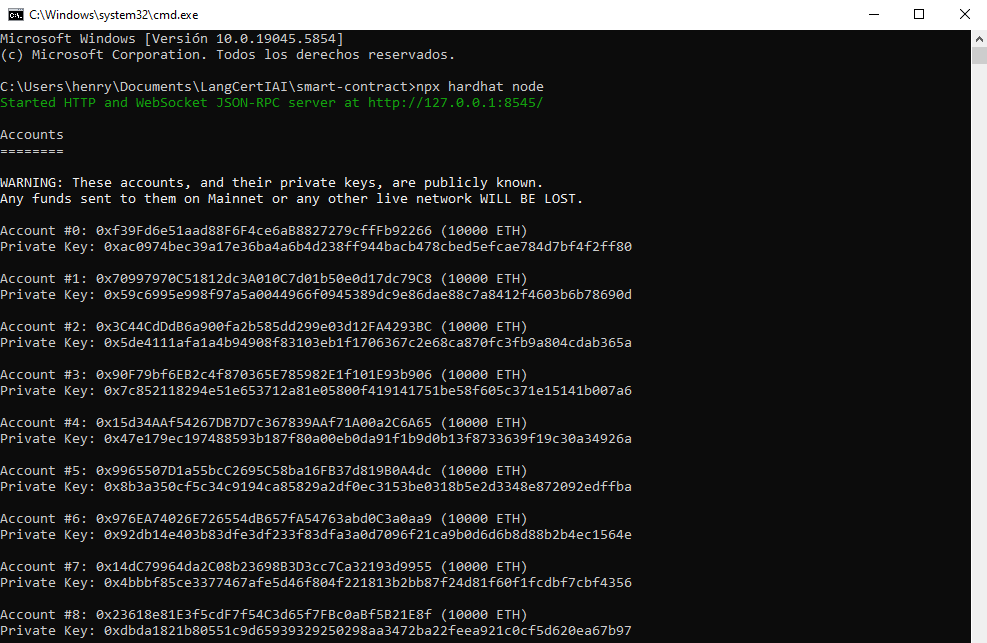


Fig. 39. Inicialización del nodo local de Ethereum

Una vez que el entorno está configurado y el contrato ha sido desarrollado, ejecutamos el comando *npx hardhat compile* dentro de la carpeta *smart-contract* utilizando *cmd* del sistema, este comando verifica la sintaxis del contrato inteligente y lo convierte en *bytecode* que puede ser desplegado en la red Ethereum. Si no se detectan errores, se genera automáticamente el archivo *artifacts/,* el cual contiene el ABI y otros metadatos necesarios para interactuar con el contrato desde la aplicación, como se ve en la figura 40.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 40. Archivos generados de archivo dentro de la carpeta artifacts

Una vez que el nodo está en ejecución, el contrato inteligente puede ser desplegado sobre esta red mediante el comando *npx hardhat run scripts/deploy.js --network localhost*. Este comando ejecuta el script de despliegue *deploy.js*, obteniendo la dirección del contrato en la red, lo que permite integrarlo con la aplicación backend para emitir, verificar e invalidar certificados mediante llamadas al contrato inteligente.

#### Incorporar dentro de la red blockchain de prueba

Después de su despliegue, el contrato inteligente se incorpora dentro de una red blockchain de prueba local, lo que permite simular un entorno real de ejecución. Esta red actúa como un entorno seguro para verificar que la interacción entre la aplicación y el contrato inteligente funcione correctamente antes de considerar su implementación con una interfaz web.

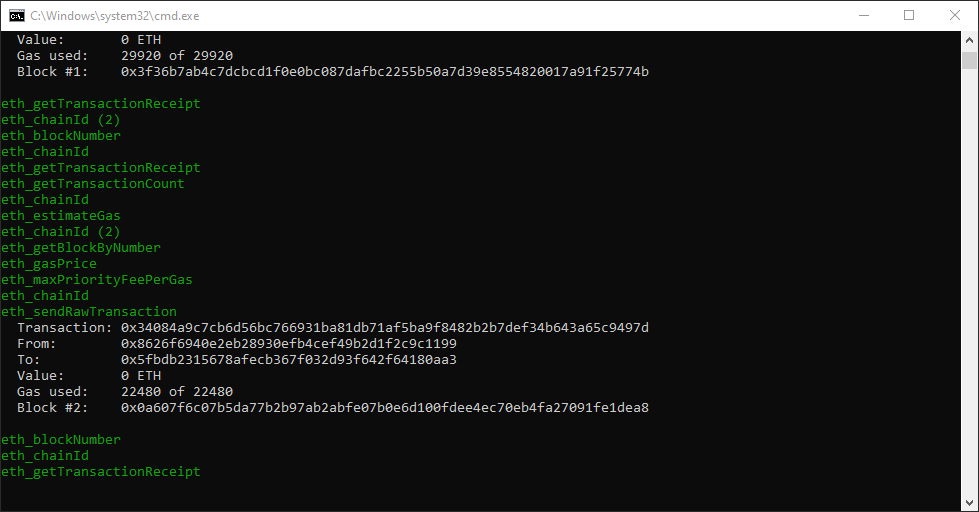


Fig. 41. Transacción correspondiente al despliegue del contrato.

La figura 41 muestra un mensaje en la red local de Ethereum, que el contrato fue correctamente desplegado a la dirección 0x5fbdb2315678afecb367f032d93f642f64180aa3 en el bloque número 1 de la red local.

### Interacción con el contrato inteligente

Una vez que el contrato inteligente ha sido correctamente programado e implementado en la blockchain local de Hardhat, el siguiente paso consiste en desarrollar una aplicación web que permita interactuar con dicho contrato. Para ello, se utilizará la herramienta *Vite*, que facilita la creación de un proyecto basado en la biblioteca React.

**Configuración del entorno Fronend**

Este entorno se configura desde el directorio del proyecto, específicamente dentro de la carpeta *backend*, desde la línea de comandos ejecutamos el comando *npm create vite@latest fronend*, permite crear un nuevo proyecto fronend empleando la herramienta *Vite*, que ofrece una estructura optimizada para aplicaciones modernas, durante su ejecución se selecciona *React* como biblioteca principal, lo cual permite construir una interfaz dinámica y modular. Una vez generado el proyecto se instala todas las dependencias necesarias definidas en el archivo *package.json*, ejecutando el comando *npm install* para que todas las librerías funcionen correctamente como muestra la figura 42, para iniciar la interfaz de desarrollo proporcionada por *Vite*, se ejecuta *npm run dev* para levanta un servidor de desarrollo local con la dirección http://localhost:5173 permitiendo cargar una página de prueba generada automáticamente.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fig. 42. Estructura de directorios de la carpeta fronend

Antes de comenzar a programar la interfaz de usuario, es necesario integrar un sistema de estilos flexible. Para ello, desde la terminal se ejecuta npm install -D tailwindcss@3 postcss autoprefixer, se instala Tailwind CSS en su versión 3, junto con postcss y autoprefixer. Tailwind facilita la creación de interfaces visuales utilizando clases utilitarias, mientras que Postcss y Autoprefixer permiten procesar y mejorar la compatibilidad de los estilos con los navegadores. Una vez finalizada esta instalación, se genera los archivos de configuración *tailwind.config.cjs* y *postcss.config.cjs* ejecutando el comando *npx tailwindcss init -p,* para personalizar los estilos como se muestra en la figura 43.

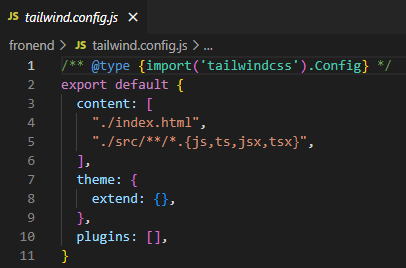


Fig. 43. Contenido del archivo tailwind.config.js

Para mejorar la experiencia de navegación entre diferentes vistas como página de inicio, formulario de emisión, lista de certificados sin necesidad de recargar la página, instalamos la biblioteca de gestión de rutas de React ejecutando el comando *npm install react-router-dom*, finalmente instalamos la biblioteca *Axios* para realizar peticiones HTTP desde el cliente ejecutando el comando *npm install axios,* para la comunicación entre el fronend y el backend, ya que permite enviar solicitudes, recibir respuestas y manejar datos provenientes del servidor o de la red blockchain.

A continuación, se define la estructura fundamental de la aplicación, la cual se ilustra en la figura 44, este archivo HTML es la plantilla base que carga la aplicación web. Define la configuración inicial y el contenedor donde el código JavaScript se encargará de construir y mostrar la interfaz de usuario.



Fig. 44. Contenido del archivo index.html

El archivo *main.jsx* constituye el punto de entrada principal de la aplicación desarrollada con React como se observa en la figura 45, su función es importar los estilos necesarios y el componente raíz de la aplicación, denominado *App*.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 45. Contenido del archivo main.jsx

El archivo *App.jsx* constituye el componente principal de la aplicación, encargado de gestionar la navegación y el control de acceso mediante la biblioteca React Router tal como se muestra en la figura 46, en este archivo se importan los componentes visuales y funcionales necesarios, así como los estilos mediante Tailwind CSS.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 46. Contenido del archivo App.jsx

El archivo *Navbar.jsx* facilita el desplazamiento entre las diferentes vistas del sistema, este componente se adapta dinámicamente según el estado de autenticación y el rol del usuario, como se observa en la figura 47, el componente consulta el almacenamiento local del navegador (localStorage) para determinar si existe un token de autenticación y qué rol tiene el usuario (ya sea administrador o estudiante). En función de esta información, se renderizan de manera condicional los enlaces disponibles en la interfaz.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 47. Contenido del archivo Navbar.jsx

Dentro de la carpeta *pages* se desarrollan las distintas vistas de la aplicación, cada una con extensión *.jsx* las cuales conforman la interfaz gráfica del sistema, estas páginas corresponden a las opciones de navegación disponibles, tales como: Inicio, Verificar, Emitir, Certificados, Registrar, Consultar y Cerrar sesión, las cuales se muestran según el rol del usuario autenticado. Adicionalmente, se incluye la página principal *Home.jsx*, que sirve como punto de bienvenida al sistema. La estructura de esta carpeta se ilustra en la Figura 48.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Fig. 48. Estructura de la carpeta pages

La página *Home.jsx* corresponde a la vista principal del sistema como se muestra en la figura 49, la cual se presenta al usuario al acceder a la ruta raíz de la aplicación, este componente permite gestionar el estado del rol del usuario para personalizar la interfaz según el perfil correspondiente.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Fig. 49. Página principal del componente Home.jsx

El archivo *Login.jsx* define el componente responsable de gestionar el proceso de inicio de sesión en el sistema, como se ilustra en la Figura 50. Esta vista permite a los usuarios autenticarse mediante sus credenciales institucionales, utilizando su correo institucional y contraseña para acceder a las funcionalidades correspondientes a su perfil.

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Fig. 50. Página de inicio de sesión del componente Login.jsx

El archivo *Register.jsx* define el componente encargado del registro de nuevos usuarios dentro del sistema, esta página permite capturar datos mediante un formulario que solicita el correo institucional, la contraseña, la cédula de identidad y el rol del usuario (estudiante o administrador), como se muestra en la Figura 51.

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Fig. 51. Página de registro de nuevos usuarios al sistema del componente Register.jsx

El archivo *IssueCertificate.jsx* define el componente responsable de la emisión de certificados digitales dentro del sistema, una funcionalidad exclusiva para usuarios con rol de administrador, esta vista presenta un formulario que recopila información clave como el nombre del instituto, la facultad, los datos del estudiante (nombre y cédula), el idioma cursado y el nivel alcanzado, tal como se muestra en la Figura 52.

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Fig. 52. Página de emisión de certificados del componente IssueCertificate.jsx

El archivo *VerifyCertificate.jsx* define el componente encargado de verificar la validez de un certificado digital emitido por el sistema, como se observa en la figura 53, esta funcionalidad está orientada a cualquier usuario que desee comprobar la autenticidad de un certificado mediante el ingreso del número de cédula del titular.

Interfaz de usuario gráfica, Aplicación, Word, Teams

El contenido generado por IA puede ser incorrecto.

Fig. 53. Página para verificar la validez del certificado, del componente VerifyCertificate.jsx

El archivo *GetCertificate.jsx* define el componente que permite a los usuarios consultar y visualizar los certificados emitidos en el sistema mediante el número de cédula del estudiante, como se muestra en la figura 54, esta funcionalidad está orientada principalmente a los estudiantes, con el objetivo de acceder fácilmente a la información de un certificado específico.

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Fig. 54. Página para consultar y visualizar el certificado, del componente GetCertificate.jsx

El archivo *AllCertificates.jsx* implementa un componente encargado de mostrar, buscar, descargar e invalidar certificados emitidos dentro del sistema, esta interfaz está diseñada principalmente para usuarios administradores que requieren una vista global y control sobre los certificados existentes, como se muestra en la figura 55.

Tabla

El contenido generado por IA puede ser incorrecto.

Fig. 55. Página de vista global y control sobre los certificados existentes, del componente AllCertificates.jsx

# Capítulo 5

## Conclusiones y recomendaciones

### Discusión

Es evidente que las tecnologías emergentes como Blockchain y los Smart Contracts se destacan por su versatilidad y potencial en diversos sectores, en el ámbito de la educación, se consolidan como elementos clave en la transformación digital, al ofrecer soluciones eficaces para la gestión y certificación de documentos académicos. A medida que las instituciones adoptan procesos más digitalizados, blockchain se presenta como una herramienta confiable que garantiza seguridad, transparencia e integridad en el manejo de la información.

Numerosos estudios respaldan su implementación, destacando su capacidad para fortalecer la seguridad de la información y garantizar la autenticidad de los documentos, lo que representa un beneficio tanto para los estudiantes como para las instituciones educativas. Entre ellos, la investigación de David Hernández [18] plantea un sistema de gestión y verificación de credenciales académicas basándose en un análisis de las capacidades de la tecnología Blockchain, demostrando que la acreditación digital de una asignatura puede verificarse de forma confiable mediante dicha tecnología, y que la integración de contratos inteligentes impide su manipulación, reforzando la seguridad, transparencia e integridad de los registros académicos, permitiendo a los estudiantes demostrar sus logros de manera confiable.

A partir del análisis del problema y los objetivos definidos en esta investigación, se exploraron distintas alternativas con el propósito de diseñar un modelo que aproveche las ventajas de la tecnología blockchain. Para ello, se desarrolló un prototipo de aplicación descentralizada, partiendo de una entrevista estructurada a los actores involucrados, con el fin de identificar sus necesidades. Sobre esta base, se diseñó un modelo de solución descentralizado apropiado para el Instituto Académico de Idiomas de la Universidad Central del Ecuador. Posteriormente, se definió una arquitectura multicapa para la aplicación web, y se evaluaron diferentes plataformas blockchain como Ethereum, Hyperledger Fabric, Solana y Polkadot, seleccionando finalmente Ethereum por su idoneidad en el caso de estudio.

Se desarrolló un contrato inteligente con las funciones necesarias para gestionar los certificados académicos, este fue compilado utilizando Hardhat, un entorno de desarrollo para Ethereum que permite gestionar contratos y simular redes blockchain localmente, la compilación genera un archivo JSON ABI, que permite la interacción entre la interfaz web y el contrato inteligente, una vez compilado sin errores el contrato fue desplegado en una red local de Hardhat, que simula una blockchain de Ethereum, este entorno genera cuentas preconfiguradas con Ether de prueba, lo que facilita el desarrollo sin necesidad de acceder a redes públicas.

Tras el despliegue exitoso del contrato, se desarrolló una interfaz gráfica utilizando *Vite*, que facilita la creación de un proyecto basado en la biblioteca React, enfocada en la funcionalidad y priorizando la facilidad de uso, evitando elementos innecesarios que interfieran con la experiencia del usuario, por seguridad se implementó un componente de inicio de sesión mediante el correo institucional y la contraseña, permitiendo el acceso a las funcionalidades específicas según el rol del usuario.

Finalmente, se construyó un prototipo funcional que permite demostrar la viabilidad de la solución propuesta, este prototipo valida la hipótesis de que los certificados académicos pueden ser gestionados de forma eficiente y segura mediante el uso de blockchain y contratos inteligentes. En términos prácticos, el proyecto integra múltiples frameworks y herramientas modernas en toda su arquitectura, aprovechando componentes y recursos de código abierto que permiten desarrollar aplicaciones de última generación con mayor eficiencia y robustez.

### Conclusiones

La fase de propuesta del prototipo, centrada en la planificación y recopilación de información clave, fue esencial para el desarrollo del proyecto, gracias a esta etapa, fue posible comprender a fondo la problemática planteada y establecer los requisitos necesarios para diseñar un modelo descentralizado de gestión de certificados académicos utilizando blockchain y contratos inteligentes. Las entrevistas realizadas como fuente primaria permitieron identificar con precisión las debilidades del sistema actual y orientar el desarrollo de una solución que garantice la integridad, seguridad y confiabilidad de los certificados emitidos.

Durante la construcción de la arquitectura del sistema, se integraron diversas herramientas y frameworks modernos, desde la interfaz de usuario que se diseñó con Tailwind CSS, hasta el backend que fue desarrollado en Node.js aprovechando su eficiencia y escalabilidad, esta combinación de tecnologías libres y de código abierto facilitó la creación de una solución robusta y moderna.

La aplicación de una metodología experimental permitió seguir una ruta clara y estructurada, desde la identificación de requerimientos hasta el despliegue y la interacción con contratos inteligentes a través de una interfaz gráfica, esta metodología garantiza no solo el cumplimiento de los objetivos planteados, sino también la viabilidad técnica y funcional del prototipo propuesto, sentando las bases para una solución innovadora y aplicable en el contexto educativo.

Por otro lado, la integración y pruebas de los diferentes componentes permitieron validar el correcto funcionamiento de la aplicación, se comprobó que el sistema no solo cumple con los requisitos técnicos establecidos, sino que también resuelve de manera efectiva el problema planteado al inicio del proyecto.

En resumen, la implementación de este prototipo, basado en tecnologías como blockchain y contratos inteligentes, ha demostrado ser una solución eficaz para garantizar la autenticidad y validez de los certificados digitales. La naturaleza inmutable de la blockchain asegura que los certificados no puedan ser modificados una vez registrados, lo que brinda altos niveles de confianza tanto a las instituciones emisoras como a los beneficiarios. Además, la automatización del proceso mediante contratos inteligentes mejora la eficiencia y elimina la necesidad de intermediarios, consolidando una propuesta tecnológica innovadora y segura para el sector educativo.

### Recomendaciones

En primer lugar, es fundamental identificar con precisión los elementos clave del sistema a desarrollar, con el objetivo de definir claramente su alcance y los recursos necesarios para su implementación exitosa, en este sentido, se recomienda recopilar información tanto directa como indirectamente del proceso que se busca mejorar, esto permitirá diseñar el sistema desde un enfoque sistémico, considerando todos los componentes e interacciones requeridas para satisfacer las expectativas de los usuarios finales.

Asimismo, se sugiere ampliar la funcionalidad del sistema para incorporar otros tipos de certificados digitales, como títulos académicos, certificados de capacitación o certificados médicos, dado que la infraestructura desarrollada es adaptable, esta ampliación permitiría gestionar una variedad más amplia de documentos digitales, aumentando la versatilidad y el valor práctico de la plataforma.

Si bien el uso de contratos inteligentes proporciona un alto nivel de seguridad, es recomendable reforzar la protección de la interfaz de usuario y del servidor backend, implementando mecanismos que permitan la recuperación de acceso en caso de pérdida de credenciales o claves privadas, estos mecanismos podrían incluir preguntas de seguridad o procesos de verificación de identidad, con el fin de garantizar que solo los usuarios autorizados puedan tener acceso total al sistema.

## Referencias

|  |  |
| --- | --- |
| [1] | M. P. Jaramillo y N. Piedra, «IEEE Xplore,» Use of blockchain technology for Academic Certification in Higher Education Institutions, 19 03 2021. [En línea]. Available: https://ieeexplore.ieee.org/abstract/document/9381181. [Último acceso: 13 9 2024]. |
| [2] | A. Badr, L. Rafferty, Q. H. Mahmoud y K. Elgazzar, «IEEE Xplore,» A Permissioned Blockchain-Based System for Verification of Academic Records, 24 06 2019. [En línea]. Available: https://ieeexplore.ieee.org/document/8763831. [Último acceso: 13 9 2024]. |
| [3] | I. Alnafrah, «ScienceDirect,» Revitalizing blockchain technology potentials for smooth academic records management and verification in low-income countries, 9 09 2021. [En línea]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0738059321001139?via%3Dihub. [Último acceso: 14 9 2024]. |
| [4] | M. Morales Morales, L. Rosero Correa y S. Morales Cardoso, «REPOSITORIO INSTITUCIONAL UNIVERSIDAD CENTRAL DEL ECUADOR,» Registro de títulos académicos mediante una aplicación basada en Blockchain y Smart Contracts, 29 05 2020. [En línea]. Available: https://revistadigital.uce.edu.ec/index.php/CATEDRA/article/view/2200. [Último acceso: 14 9 2024]. |
| [5] | V. Gómez Bocanegra, G. S. García-delaTorre, C. A. PantojaMeléndeza y B. Loose Rojo, «ScienceDirect,» Blockchain aplicado en afecciones mamarias: desafíos y consideraciones éticas, 05 11 2024. [En línea]. Available: https://www.sciencedirect.com/science/article/pii/S021415822400080X. [Último acceso: 21 9 2024]. |
| [6] | R. I. Lara Moran, «Repositorio Institucional de la Universidad Politécnica Salesiana,» Arquitectura de información para asegurar la trazabilidad en la distribución de medicina en el Ministerio de Salud Pública basado en Blockchain, 23 07 2024. [En línea]. Available: http://dspace.ups.edu.ec/handle/123456789/29313. [Último acceso: 21 9 2024]. |
| [7] | F. J. Miranda González., A. Chamorro Mera y S. Rubio Lacoba, «Revistas ICE,» Clarificando el concepto de certificación: El caso español, 28 11 2004. [En línea]. Available: https://www.revistasice.com/index.php/BICE/article/view/3690. [Último acceso: 8 10 2024]. |
| [8] | C. Reyes Sáncheza, «Interconectando Saberes,» Blockchain: Funcionamiento y pertinencia en sectores públicos y privados, 17 08 2022. [En línea]. Available: https://is.uv.mx/index.php/IS/article/view/2734. [Último acceso: 8 10 2024]. |
| [9] | C. Kuchkovsky, G. Gómez Lardies, D. Díez García y Í. Molero, «Google academico,» BLOCKCHAIN: LA REVOLUCIÓN INDUSTRIAL DE INTERNET, 2017. [En línea]. Available: https://www.planetadelibros.cl/libros\_contenido\_extra/36/35615\_Blockchain.pdf. [Último acceso: 8 10 2024]. |
| [10] | C. Zozaya, J. Incera y L. A. Franzoni, «Google Academico,» BLOCKCHAIN: UN TUTORIAL, 2019. [En línea]. Available: https://scholar.google.es/scholar?hl=es&as\_sdt=0%2C5&q=BLOCKCHAIN%3A+UN+TUTORIAL&btnG=. [Último acceso: 8 10 2024]. |
| [11] | B. Murthy, Lawanya Shri, K. Seifedine y S. Lim, «IEEE Xplore,» Blockchain Based Cloud Computing: Architecture and Research Challenges, 11 09 2020. [En línea]. Available: https://ieeexplore.ieee.org/abstract/document/9252909. [Último acceso: 8 10 2024]. |
| [12] | J. G. Orosco Pantoja y J. J. Sánchez Lucas, «REPOSITORIO INSTITUCIONAL UNIVERSIDAD CENTRAL DEL ECUADOR,» Desarrollo de un prototipo de aplicación para la gestión de derechos de autor de obras musicales basado en tecnología blockchain y smart contract, 2023. [En línea]. Available: https://www.dspace.uce.edu.ec/entities/publication/463ef5e6-5ed3-4976-b8a3-0453372b9ca1. [Último acceso: 8 10 2024]. |
| [13] | Á. Santos García, «REPOSITORIO UNIVERSIDAD CARLOS III DE MADRID,» Caracterización de Smart Contracts en Ethereum, 18 10 2019. [En línea]. Available: https://e-archivo.uc3m.es/entities/publication/080c2dc1-e2b3-46a5-aa60-6478bbf607ab. [Último acceso: 31 10 2024]. |
| [14] | Z. Zibin, X. Shaoan , D. Hongning , C. Xiangping y W. Huaimin , «IEEE Xplore,» 25 9 2017. [En línea]. Available: https://ieeexplore.ieee.org/document/8029379. |
| [15] | Satoshi Nakamoto, «bitcoin.org,» Bitcoin: un sistema de dinero en efectivo electrónico peer-to-peer, 2008. [En línea]. Available: https://bitcoin.org/files/bitcoin-paper/bitcoin\_es.pdf. [Último acceso: 31 10 2024]. |
| [16] | K. M. Bhabendu, P. Soumyashree y J. Debasish , «IEEE Xplore,» An Overview of Smart Contract and Use Cases in Blockchain Technology, 18 10 2018. [En línea]. Available: https://ieeexplore.ieee.org/abstract/document/8494045. [Último acceso: 15 11 2024]. |
| [17] | B. F. Ocaña Valdez, «Repositorio Institucional de la Universidad Politécnica Salesiana,» Desarrollo de un prototipo de una criptomoneda con herramientas de código abierto basado en Ethereum de la Universidad Politécnica Salesiana, 02 2024. [En línea]. Available: https://dspace.ups.edu.ec/handle/123456789/26831. [Último acceso: 15 11 2024]. |
| [18] | . D. F. Hernández Formento, «Repositorio Universidad de Sevilla,» Análisis y capacidades de la tecnología Blockchain en el ámbito de la acreditación Académica., 26 06 2024. [En línea]. Available: https://idus.us.es/items/25eaf1c7-dc3e-4926-9e5c-33758fcc3ea3. [Último acceso: 15 11 2024]. |
| [19] | J. Ramirez, B. Castillo, J. C. Benavides y Y. Peralta, «E Nicaragua - Revista de La Universidad Autónoma,» Metodología de la Investigación e Investigación Aplicada para Ciencias Económicas y Administrativas, 2018. [En línea]. Available: https://jalfaroman.wordpress.com/wp-content/uploads/2019/03/dosier-metodologia-e-investigacion-aplicada-2018.pdf. [Último acceso: 5 3 2025]. |
| [20] | M. A. Lopera Vélez, «Google Scholar,» METODOLOGÍA DE LA INVESTIGACIÓN, 2012. [En línea]. Available: https://d1wqtxts1xzle7.cloudfront.net/31872466/2.\_METODOLOGIA\_DE\_LA\_INVESTIGACION-libre.pdf?1392460870=&response-content-disposition=inline%3B+filename%3DMETODOLOGIA\_DE\_LA\_INVESTIGACION\_Metodolo.pdf&Expires=1740691871&Signature=ZzuqUQ6ELNci3gCZL0FSfnDOS~s. [Último acceso: 5 03 2025]. |
| [21] | F. Arias, «Introducción a la metodología científica,» de *El proyecto de investigación*, 6ta ed., Caracas-República Bolivariana de Venezuela, Episteme, 2006, p. 143. |
| [22] | J. Lozada, «Dialnet,» Investigación Aplicada: Definición, Propiedad Intelectual e Industria, 12 2014. [En línea]. Available: https://dialnet.unirioja.es/servlet/articulo?codigo=6163749. [Último acceso: 5 3 2015]. |
| [23] | P. Esquivel, L. Quisaguano, A. Caluña y S. Llambo, «Dialnet - Universidad Técnica de Cotopaxi,» Frameworks del lado del Servidor Caso de Estudio Node JS, Django y Laravel, 19 11 2024. [En línea]. Available: https://dialnet.unirioja.es/servlet/articulo?codigo=9966614. [Último acceso: 22 05 2025]. |
| [24] | A. Eriksson, «Digitala Vetenskapliga Arkivet,» Utveckling av Satellitsidor och Presentationsappli kation med React.js och Express.js, 22 03 2024. [En línea]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1876346&dswid=4147. [Último acceso: 30 04 2025]. |
| [25] | P. Llebrés Ferriol, «Repositorio Universitat Oberta de Catalunya,» Diseño e implementación de una web de catalogación de libros de segunda mano., 5 01 2025. [En línea]. Available: https://openaccess.uoc.edu/handle/10609/151846. [Último acceso: 23 5 2025]. |
| [26] | O. Rodríguez Setó , «Google academico - Universitat Politècnica de Catalunya,» A Decentralized, Secure and Censorship Resistant Governance Protocol on Ethereum, 06 2022. [En línea]. Available: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://upcommons.upc.edu/bitstream/handle/2117/380645/final\_degree\_thesis\_oriolrodriguezseto.pdf?sequence=2. [Último acceso: 23 5 2025]. |
| [27] | K. S. Narváez y C. D. Alava , «ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL,» Implementación de un Sistema de un VideoVigilancia con un método de Autenticación basado en cadena de bloques, 2023. [En línea]. Available: https://www.dspace.espol.edu.ec/handle/123456789/60637. [Último acceso: 1 5 2025]. |
| [28] | N. Gupta, R. Singh y S. S. Das, «Google academico-psvpec.in,» MERN Stack in Web Development: An Interactive Approach, 2 08 2023. [En línea]. Available: https://www.psvpec.in/jcres/2023\_2/57.pdf. [Último acceso: 01 05 2025]. |
| [29] | R. S. Bastos Russi, «Repositorio UNIVERSIDAD DE LOS ANDES COLOMBIA,» Desarrollo de soluciones blockchain: Plataforma de perfiles financieros para el servicio de Open Banking, a través de pruebas de conocimiento cero, 14 12 2024. [En línea]. Available: https://repositorio.uniandes.edu.co/entities/publication/5572ed06-dcce-4501-9105-88545be6dcc8. [Último acceso: 23 5 2025]. |
| [30] | M. Vaccargiu , S. Aufiero, C. Ba y S. Bartolucci, «arXiv-Cornell University,» Mining a Decade of Event Impacts on Contributor Dynamics in Ethereum: A Longitudinal Study, 7 02 2025. [En línea]. Available: https://arxiv.org/abs/2502.05054. [Último acceso: 1 5 2025]. |
| [31] | J. Marugán Rivilla, «REPOSITORIO UNIVERSIDAD POLITÉCNICA DE MADRID,» Desarrollo de una Red Blockchain para un caso de uso especifico, 07 2019. [En línea]. Available: https://oa.upm.es/62849/. [Último acceso: 23 05 2025]. |

## Anexos