

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.

Autor: Henry Paolo Felicita Vasco

Tutor: Ing. Santiago Morales Cardoso, PhD

Quito, 2025

**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.”

Declaro 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.

Reconozco 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**

**C.I. 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 comienza 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 que lleva a cabo el Instituto Académico de Idiomas en la emisión los 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, el proceso inicia con la propuesta del prototipo, donde se identifican las necesidades de los usuarios finales y se diseña el modelo de la aplicación, se define la arquitectura del sistema para seleccionar herramientas de código abierto y marcos de desarrollo que facilitan la integración de los distintos componentes con la red blockchain. Una vez completada esta fase, se codifica, copila y despliega el contrato inteligente que establece las reglas de negocio, por último, 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, Aplicación web, Certificación académica, Aplicaciones descentralizadas, Prototipo.

**Abstract.**

# Índice general de contenidos

[Índice general de contenidos i](#_Toc202350248)

[Lista de tablas iii](#_Toc202350249)

[Lista de figuras iii](#_Toc202350250)

[Capítulo 1 1](#_Toc202350251)

[1. Generalidades 1](#_Toc202350252)

[1.1. Introducción 1](#_Toc202350253)

[1.2. Antecedentes 2](#_Toc202350254)

[1.3. Planteamiento del problema 4](#_Toc202350255)

[1.4. Hipótesis y Justificación 5](#_Toc202350256)

[1.4.1. Hipótesis 5](#_Toc202350257)

[1.4.2. Justificación 6](#_Toc202350258)

[1.5. Objetivos 6](#_Toc202350259)

[1.5.1. General 7](#_Toc202350260)

[1.5.2. Específicos 7](#_Toc202350261)

[Capítulo 2 8](#_Toc202350262)

[2. Marco Teórico 8](#_Toc202350263)

[2.1. Conceptos 8](#_Toc202350264)

[2.1.1. Certificación 8](#_Toc202350265)

[2.1.2. Certificación Académica 8](#_Toc202350266)

[2.2. Blockchain 8](#_Toc202350267)

[2.2.1. Características de Blockchain 9](#_Toc202350268)

[2.2.2. Arquitectura de blockchain 10](#_Toc202350269)

[2.2.3. Bloque 10](#_Toc202350270)

[2.2.4. Mecanismo de Consenso 12](#_Toc202350271)

[2.3. Smart contracts 13](#_Toc202350272)

[2.3.1. Estructura de un smart contract 13](#_Toc202350273)

[2.4. Herramientas tecnológicas 14](#_Toc202350274)

[Capítulo 3 19](#_Toc202350275)

[3. Metodología experimental 19](#_Toc202350276)

[Capítulo 4 21](#_Toc202350277)

[4. Cálculos y resultados 21](#_Toc202350278)

[4.1. Propuesta del prototipo 21](#_Toc202350279)

[4.1.1. Propósito 21](#_Toc202350280)

[4.1.2. Alcance 21](#_Toc202350281)

[4.1.3. Identificación de procesos 22](#_Toc202350282)

[4.1.4. Identificar a los stakeholders 23](#_Toc202350283)

[4.1.5. Modelo propuesto 23](#_Toc202350284)

[4.1.6. Funciones del prototipo 24](#_Toc202350285)

[4.1.7. Requisitos funcionales 32](#_Toc202350286)

[4.1.8. Requisitos No funcionales 33](#_Toc202350287)

[4.2. Descripción de la arquitectura 34](#_Toc202350288)

[4.3. Selección de herramientas 36](#_Toc202350289)

[4.3.1. Node.js 36](#_Toc202350290)

[4.3.2. React.js 37](#_Toc202350291)

[4.3.3. Tailwind CSS 37](#_Toc202350292)

[4.3.4. Vite JS 37](#_Toc202350293)

[4.3.5. Metamask 37](#_Toc202350294)

[4.3.6. Express JS 38](#_Toc202350295)

[4.3.7. MongoDB 38](#_Toc202350296)

[4.3.8. Hardhat 38](#_Toc202350297)

[4.3.9. Ethers JS 38](#_Toc202350298)

[4.4. Desarrollo del prototipo propuesto 38](#_Toc202350299)

[4.4.1. Certificación académica LangCertIAI 39](#_Toc202350300)

[4.4.2. Estructura del proyecto 39](#_Toc202350301)

[4.4.3. Desarrollo del contrato inteligente 39](#_Toc202350302)

[4.4.3.1. Codificación del contrato inteligente 44](#_Toc202350303)

[4.4.3.2. Copilar y desplegar el contrato 63](#_Toc202350304)

[4.4.3.3. Incorporar dentro de la red blockchain de prueba 65](#_Toc202350305)

[4.4.3.4. Interacción con el contrato inteligente 66](#_Toc202350306)

[Capítulo 5 79](#_Toc202350307)

[5. Conclusiones y recomendaciones 79](#_Toc202350308)

[5.1. Discusión 79](#_Toc202350309)

[5.2. Conclusiones 80](#_Toc202350310)

[5.3. Recomendaciones 82](#_Toc202350311)

[6. Referencias 82](#_Toc202350312)

[7. Anexos 89](#_Toc202350313)

# Lista de tablas

[Tab. 1. Ventajas e inconvenientes de una red blockchain 15](#_Toc202347386)

[Tab. 2. Ventajas e inconvenientes del entorno de desarrollo integrado 17](#_Toc202347387)

[Tab. 3. Ventajas de los lenguajes de programación 17](#_Toc202347388)

[Tab. 4. Hallazgos 23](#_Toc202347389)

[Tab. 5. Descripción de los actores 23](#_Toc202347390)

[Tab. 6. Autenticación y Autorización 28](#_Toc202347391)

[Tab. 7. Gestionar certificados 30](#_Toc202347392)

[Tab. 8. Validar Certificado 32](#_Toc202347393)

[Tab. 9. Requisitos funcionales 33](#_Toc202347394)

[Tab. 10. Requisitos no funcionales 34](#_Toc202347395)

[Tab. 11. Funciones del contrato LanguageCertificate 48](#_Toc202347396)

[Tab. 12. Dependencias del archivo backend. 50](#_Toc202347397)

[Tab. 13. Tabla de restricción de acceso a determinadas rutas 57](#_Toc202347398)

[Tab. 14. Funciones del archivo certificatecontrolle.js 60](#_Toc202347399)

# Lista de figuras

[Fig. 1. Estructura de una red blockchain. 10](#_Toc202347529)

[Fig. 2. Contenido de un bloque 11](#_Toc202347530)

[Fig. 3. Estructura de un contrato inteligente 14](#_Toc202347531)

[Fig. 4. Modelo propuesto: Certificación descentralizada de documentos académicos 24](#_Toc202347532)

[Fig. 5. Diagrama de casos de uso del prototipo web, función nivel cero 25](#_Toc202347533)

[Fig. 6. Diagrama de casos de uso para la autenticación y autorización, función nivel uno 26](#_Toc202347534)

[Fig. 7. Diagramas de casos de uso para gestionar los certificados, función en nivel uno 28](#_Toc202347535)

[Fig. 8. Diagramas de casos de uso para validar los certificados, función en nivel uno 30](#_Toc202347536)

[Fig. 9. Arquitectura propuesta de la aplicación 36](#_Toc202347537)

[Fig. 10. Estructura de directorios del proyecto 39](#_Toc202347538)

[Fig. 11. Esquema de fases para el desarrollo del contrato inteligente 40](#_Toc202347539)

[Fig. 12. Estructura de directorios de la carpeta smart-contract 41](#_Toc202347540)

[Fig. 13. Código fuente del archivo deploy.js 42](#_Toc202347541)

[Fig. 14. Código fuente del archivo hardhat.config.js 43](#_Toc202347542)

[Fig. 15. Código fuente del archivo hardhat.config.js 44](#_Toc202347543)

[Fig. 16. Contrato inteligente llamado LanguageCertificate.sol 45](#_Toc202347544)

[Fig. 17. Especificación de la versión de Solidity 45](#_Toc202347545)

[Fig. 18. Estructura de datos del certificado 46](#_Toc202347546)

[Fig. 19. Mapeo del certificado 46](#_Toc202347547)

[Fig. 20. Función emitir certificado. 47](#_Toc202347548)

[Fig. 21. Implementación de las funciones en el contrato LanguageCertificate 48](#_Toc202347549)

[Fig. 22. Estructura de directorios de la carpeta backend 49](#_Toc202347550)

[Fig. 23 Archivo de configuración .env 51](#_Toc202347551)

[Fig. 24. Contenido del archivo .env 51](#_Toc202347552)

[Fig. 25. Conexión con la base de datos MongoDB 52](#_Toc202347553)

[Fig. 26. Estructura para almacenar los certificados en la base de datos 53](#_Toc202347554)

[Fig. 27. Estructura para almacenar los certificados en la base de datos 54](#_Toc202347555)

[Fig. 28. Autenticación de los roles de los usuarios. 55](#_Toc202347556)

[Fig. 29. Rutas para la gestión de certificados 56](#_Toc202347557)

[Fig. 30. Contenido del archivo authMiddleware.js autentificación 58](#_Toc202347558)

[Fig. 31. Contenido del archivo certificateController.js 59](#_Toc202347559)

[Fig. 32. Implementación de la función para emitir un certificado 60](#_Toc202347560)

[Fig. 33. Implementación de la función para verificar su validez 60](#_Toc202347561)

[Fig. 34. Implementación de la función para obtener un certificado 61](#_Toc202347562)

[Fig. 35. Implementación de la función para obtener todos los certificados 61](#_Toc202347563)

[Fig. 36. Implementación de la función para generar un archivo PDF 61](#_Toc202347564)

[Fig. 37. Implementación de la funció invalidar un certificado 62](#_Toc202347565)

[Fig. 38. Contenido del archivo auth.controler.js 63](#_Toc202347566)

[Fig. 39. Inicialización del nodo local de Ethereum 64](#_Toc202347567)

[Fig. 40. Archivos generados de archivo dentro de la carpeta artifacts 65](#_Toc202347568)

[Fig. 41. Transacción correspondiente al despliegue del contrato 66](#_Toc202347569)

[Fig. 42. Estructura de directorios de la carpeta fronend 67](#_Toc202347570)

[Fig. 43. Contenido del archivo tailwind.config.js 68](#_Toc202347571)

[Fig. 44. Contenido del archivo index.html 69](#_Toc202347572)

[Fig. 45. Contenido del archivo main.jsx 69](#_Toc202347573)

[Fig. 46. Contenido del archivo App.jsx 70](#_Toc202347574)

[Fig. 47. Contenido del archivo Navbar.jsx 71](#_Toc202347575)

[Fig. 48. Estructura de la carpeta pages 72](#_Toc202347576)

[Fig. 49. Página principal del componente Home.jsx 73](#_Toc202347577)

[Fig. 50. Página de inicio de sesión del componente Login.jsx 74](#_Toc202347578)

[Fig. 51. Página de registro de nuevos usuarios al sistema del componente Register.jsx 75](#_Toc202347579)

[Fig. 52. Página de emisión de certificados del componente IssueCertificate.jsx 76](#_Toc202347580)

[Fig. 53. Página para verificar la validez del certificado, del componente VerifyCertificate.jsx 77](#_Toc202347581)

[Fig. 54. Página para consultar y visualizar el certificado, del componente GetCertificate.jsx 78](#_Toc202347582)

[Fig. 55. Página de vista global y control sobre los certificados existentes, del componente AllCertificates.jsx 78](#_Toc202347583)

# 

# 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 esta situación, 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

Los avances tecnológicos han contribuido al surgimiento de nuevos modelos de negocio innovadores así como para mejorar los ya existentes, en este panorama, la denominada cuarta revolución industrial, impulsada principalmente por el internet, ha generado cambios significativos en múltiples aspectos de la vida diaria, transformando la manera en que la sociedad interactúa en distintos sectores productivos, dentro de este entorno de innovación, la tecnología blockchain surge como una solución disruptiva capaz de aborda diversos problemas en distintas áreas como la salud, educación, el gobierno entre otros. En el ámbito educativo, esta tecnología facilita la eliminación de intermediarios en la gestión de procesos institucionales 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].

**a. 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].

**b. 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, en el caso del Instituto Académico de Idiomas (IAI) de la Universidad Central del Ecuador son 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].

Ante esta situación, la tecnología blockchain se presenta como una alternativa con gran potencial al ofrecer características clave como la autenticidad, inmutabilidad y descentralización de los documentos académicos. Sin embargo, a pesar de su creciente relevancia, 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 viable desarrollar e implementar un prototipo que utilice tecnología blockchain como base para la certificación de documentos digitales en instituciones de educación superior, garantizando mayor seguridad, transparencia y eficiencia en los procesos de emisión y validación?

#### 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 que emite el IAI 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 basado en tecnología blockchain que permita certificar la finalización de estudios de una lengua extranjera 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

1. Proponer un modelo de solución que utilice tecnología blockchain para la validación de certificados de estudios de un idioma en el IAI de la Universidad Central del Ecuador.
2. Identificar y seleccionar la herramienta tecnológica más adecuada para el desarrollo del prototipo.
3. 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, es reconocida como una de las tecnologías clave de la llamada cuarta revolución industrial [8].

Blockchain es una red compuesta por varios nodos que contienen registros almacenados en cadenas de bloques, estos bloques están conectados entre sí de forma secuencial mediante algoritmos criptográficos, lo que conforma una cadena inmutable de datos, cada nodo mantiene una copia exacta de esta cadena, lo que permite asegurar tanto la integridad como la disponibilidad de la información [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 blockchains de carácter público, 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.

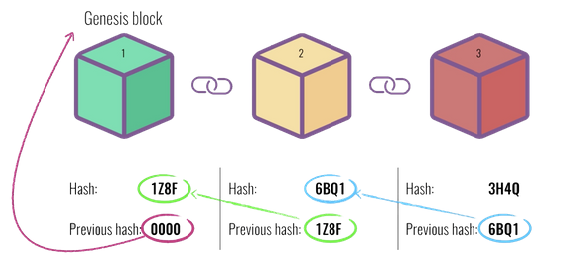


Fig. 1. Estructura de una red blockchain.

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

Dentro de una red blockchain, un bloque representa una unidad estructural fundamental que almacena un conjunto de transacciones previamente validadas, esta información es registrada de forma pública y distribuida entre todos los nodos participantes en la red, lo cual garantiza su disponibilidad e integridad [8].

Diagrama, Texto

El contenido generado por IA puede ser incorrecto.

Fig. 2. Contenido de un bloque

Nota: Carlos Reyes Sánchez [8]

Como muestra la figura 2, la estructura general de un bloque dentro de una blockchain se divide principalmente en dos secciones: 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.

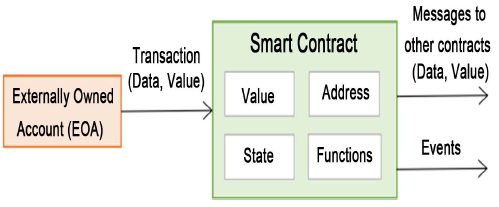


Fig. 3. Estructura de un contrato inteligente

Nota: Bhabendu Kumar [16]

Tal como se ilustra en la figura 3, se muestra un ejemplo básico del funcionamiento 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, Polkadot, Solana y Hyperledger Fabric. 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 optó por utilizar Visual Studio Code debido a su entorno intuitivo y su compatibilidad directa con herramientas para el desarrollo en Ethereum, este editor de código, al ser multiplataforma, facilita de manera eficiente la escritura, prueba y despliegue de contratos inteligentes, convirtiéndose en una opción práctica para trabajar con tecnologías basadas en blockchain.

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 de los 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 el lenguaje predominante para el desarrollo de contratos inteligentes en esta red, ya que ha sido diseñado específicamente con ese propósito y cuenta con una integración nativa con entornos de desarrollo como Remix. Estas características lo convierten en la alternativa más adecuada para el presente proyecto.

# 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 sistemático de técnicas aplicadas en una investigación científica o 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 una solución tecnológica que permita certificar la finalización de estudios de un idioma en el IAI 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 plantean una serie de etapas que guiarán el proceso de desarrollo del presente proyecto.

**- 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 desarrollo correctamente definidas, esta etapa se enfoca en la implementación de los contratos inteligentes que conforman el núcleo funcional del sistema, 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 emisión y validación de la autenticidad de los certificados de suficiencia emitidos en el IAI, 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 IAI de la Universidad Central del Ecuador. Este proceso permitió analizar los principales desafíos asociados a la emisión y validación de los certificados académicos. 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.

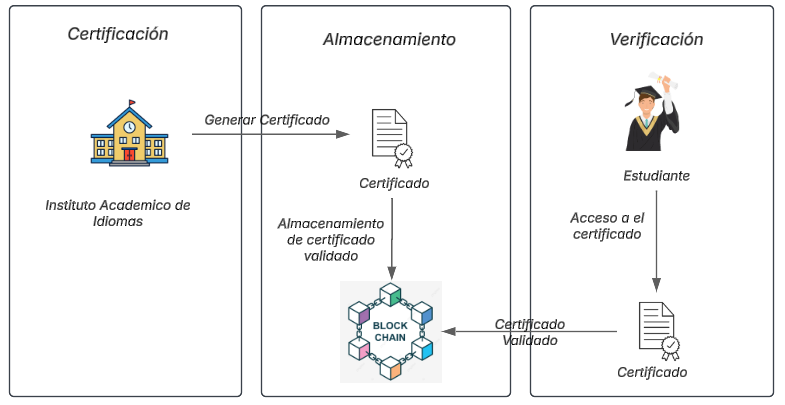


Fig. 4. Modelo propuesto: Certificación descentralizada de documentos académicos

#### Funciones del prototipo

Este segmento describe las funciones específicas del sistema orientadas a 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. Diagrama de casos de uso del prototipo web, 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 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 casos de uso para gestionar los certificados, función en nivel uno

La Tabla 7 presenta las funciones del sistema orientadas al manejo de certificados académicos digitales, en este escenario, el administrador con los permisos adecuados tiene la capacidad de emitir nuevos certificados y acceder a la información de un estudiante específico, asegurando que dichos documentos sean confiables, accesibles y protegidos contra alteraciones.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 casos de uso 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 detalla la funcionalidad dirigida al usuario final, permitiéndole verificar la validez de los certificados académicos emitidos por el IAI, entre las acciones disponibles se incluye la posibilidad de consultar, revisar, descargar y compartir el certificado en formato 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 ofrecer al administrador una opción segura para recuperar su cuenta mediante un proceso de restablecimiento de contraseña, ya sea por olvido o por solicitud de 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 brindar al administrador la capacidad de localizar certificados académicos mediante una funcionalidad de búsqueda integrada. | CU-02-2 |
| RF-8 | El sistema debe proporcionar al administrador una vista general del conjunto de certificados académicos que han sido generados dentro de la plataforma. | 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 garantizar un proceso ágil y eficiente al momento de descargar certificados digitales, minimizando cualquier retraso perceptible para el usuario. |
| 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 implementar una aplicación descentralizada (DApp) enfocada en el sector educativo, se ha diseñado una solución que integra tecnologías fundamentales como blockchain, contratos inteligentes, mecanismos de descentralización y validación digital de documentos. Esta propuesta se organiza bajo un modelo de arquitectura en capas, lo cual facilita la integración modular y eficiente de cada uno de estos componentes.

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.

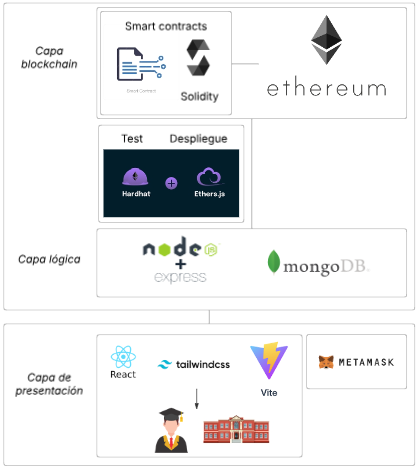


Fig. 9. Arquitectura propuesta de la aplicación

### Selección de herramientas

A partir del prototipo propuesto, así como de la arquitectura de aplicación y las plataformas expuestas en los capítulos anteriores, se definió un conjunto de herramientas específicas para llevar a cabo 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: abreviación de certification (certificación), IAI: iniciales 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

La organización del proyecto se distribuye en tres directorios principales, como se muestra en la Figura 10:

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 10. Estructura de directorios del proyecto

La primera, denominada fronend, la cual contiene la interfaz gráfica con la que interactúa el usuario, esta fue desarrollada utilizando Vite como entorno de construcción, React como framework principal, y Tailwind CSS para el diseño visual. 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

**a. 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*: Añade Hardhat como dependencia de desarrollo, permitiendo compilar, implementar, testear y trabajar con contratos inteligentes en redes compatibles con Ethereum.

*npm install ethers*: Este paquete facilita la interacción con contratos inteligentes desde aplicaciones desarrolladas en JavaScript. Es una herramienta ampliamente adoptada en conjunto con Hardhat, ya que permite gestionar cuentas, realizar transacciones, y consultar o modificar información almacenada en la blockchain.

**b. Inicialización y ejecución de Hardhat**

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

*npx hardhat*: Este comando despliega un asistente interactivo que guía al usuario durante la configuración inicial del proyecto, en este proceso, se debe elegir la opción “Crear un proyecto básico”, lo cual genera automáticamente una estructura estándar de directorios y archivos, incluyendo carpetas como *contracts/, scripts/, test/* y el archivo *hardhat.config.js*, esta organización inicial está ilustrada 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 inteligente que se desarrollará se alojará en el directorio *contracts*, mientras que el directorio *scripts* contiene el archivo *deploy.js*, encargado de compilar y desplegar el contrato en la red blockchain local proporcionada por *Hardhat*, el contenido de este archivo, mostrado en la figura 13, corresponde a un script en *JavaScript* que despliega el contrato *LanguageCertificate.sol* en la red de pruebas, recupera su dirección y su ABI (Interfaz Binaria de Aplicación), y almacena esta información en archivos JSON, estos archivos son utilizados posteriormente por la capa backend para interactuar con el contrato desplegado.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 13. Código fuente del archivo deploy.js

El archivo *hardhat.config.js*, ilustrado en la figura 14, es esencial para el correcto funcionamiento del script *deploy.js*, ya que configura el entorno de desarrollo de *Hardhat*, en este archivo se incluye el paquete *@nomicfoundation/hardhat-toolbox*, que integra varios plugins importantes, tales como *hardhat-ethers* para facilitar la interacción con contratos mediante *Ethers.js*; *hardhat-waffle* para la ejecución de pruebas automatizadas; y *hardhat-etherscan*, que simplifica la verificación de contratos en la plataforma *Etherscan*. Asimismo, se establece que la compilación de los contratos se realizará con la versión 0.8.28 del compilador Solidity.

Texto

El contenido generado por IA puede ser incorrecto.

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

El archivo *package.json*, representado en la figura 15, contiene tanto las dependencias principales del proyecto como las específicas para el entorno de desarrollo, en la sección *dependencies* se incluyen paquetes fundamentales como *@nomicfoundation/hardhat-toolbox*, que reúne utilidades clave para trabajar con Hardhat; *ethers*, una biblioteca en JavaScript que permite la interacción con la red Ethereum; y *dotenv*, usada para manejar variables de entorno de forma segura, por otro lado, en *devDependencies* se encuentra *hardhat*, herramienta central del proyecto encargada de compilar, desplegar y realizar pruebas sobre 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 empleado para construir contratos inteligentes que se ejecutan sobre 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 declarar la versión del compilador de Solidity que se utilizara, actualmente la versión 0.8 de Solidity es reconocida por su estabilidad y uso extendido, lo que asegura que las actualizaciones menores dentro de esta misma línea mantendrán compatibilidad con el contrato, esto resulta especialmente útil si en el futuro se requiere realizar ajustes o mejoras en el código. Este proceso se ejemplifica 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

**a. Estructura de datos**

El lenguaje Solidity permite la creación de estructuras de datos personalizadas utilizando la palabra clave *struct*, en este proyecto, se define una estructura denominada *Certificate*, que agrupa los atributos necesarios para representar un certificado de idiomas como se muestra en la figura 18, esta estructura incluye los siguientes campos: *instituteName*, que almacena el nombre de la institución; *facultyName*, correspondiente a la facultad; *studentName*, para el nombre del estudiante; *cedula*, que actúa como identificador único; *language*, que representa el idioma cursado; *level*, indicando el nivel aprobado (por ejemplo, A1, B2, etc.); *issueDate*, que registra la fecha de emisión; y *valid*, un valor booleano que señala si el certificado es válido.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 18. Estructura de datos del certificado

**b. Mapeo del certificado**

Para gestionar el almacenamiento de los certificados, se implementa un *mapping*, que funciona como una estructura de asociación entre la cédula del estudiante y su respectivo certificado, esta configuración permite realizar consultas eficientes y directas a los datos desde otros componentes del sistema, como se muestra en la figura 19.



Fig. 19. Mapeo del certificado

**c. 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*.

Como se observa en la figura 20, la función *issueCertificate* se encarga de generar y almacenar un nuevo certificado, para ello, recibe como argumentos el nombre de la institución emisora (*\_instituteName*), el nombre de la facultad (*\_facultyName*), el nombre del estudiante (*\_studentName*), su número de cédula (*\_cedula*), el idioma aprobado (*\_language*) y el nivel correspondiente (*\_level*).

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.

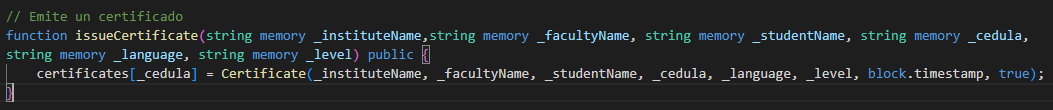


Fig. 20. Función emitir certificado.

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 comprobar si un certificado continúa siendo válido mediante la verificación del número de cédula del estudiante. Retorna el valor del campo valid: true si el certificado está activo, false si fue invalidado. |
| invalidateCertificate | *void* | Invalida un certificado que ya ha sido emitido, cambiando el valor del estado de *valid* a false. |
| updateLevel | *void* | Modifica el nivel del idioma correspondiente a un certificado que ya ha sido emitido, utilizando la cédula del estudiante como identificador para encontrar el registro. |
| getCertificate | Certificate *memory* | Devuelve la estructura completa del certificado correspondiente a una cédula determinada. |
| getIssueDate | uint256 | Retorna la fecha en la que fue emitido un certificado específico, lo que permite verificar la antigüedad, vigencia o posible caducidad del documento. |

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

**i. Configuración del Backend**

El proceso de configuración del servidor inicia dentro del directorio correspondiente al backend del proyecto, para establecer el entorno de trabajo, se procede a inicializar un proyecto basado en *Node.js* mediante la ejecución del comando *npm init -y*, este comando crea automáticamente el archivo *package.json*, el cual servirá para administrar las dependencias necesarias durante el desarrollo del backend. La figura 22 ilustra este procedimiento inicial.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fig. 22. Estructura de directorios de la carpeta backend

**a. Instalación de dependencias**

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

|  |  |
| --- | --- |
| **Librería** | **Descripción** |
| express | Framework que facilita la creación de servidores web y la gestión de rutas HTTP. |
| mongoose | Biblioteca que permite la interacción con bases de datos MongoDB. |
| dotenv | Herramienta para la gestión de variables de entorno definidas en un archivo .env. |
| cors | Middleware que habilita el intercambio de recursos entre diferentes dominios (Cross-Origin Resource Sharing). |
| jsonwebtoken | Módulo utilizado para la creación y validación de tokens JWT, asegurando la autenticación. |
| pdfkit | Librería para la generación dinámica de archivos PDF desde el servidor. |
| bcryptjs | Utilizado para el cifrado seguro de contraseñas antes de almacenarlas en la base de datos. |

Tab. 12. Dependencias del archivo backend.

**b. Configuración de la base de datos MongoDB**

Como primer paso, se crea un archivo de configuración denominado *.env*, el cual se ubica fuera del directorio principal del código fuente, este archivo tiene la función de almacenar variables de entorno sensibles que la aplicación leerá durante su ejecución, utilizando para ello la biblioteca *dotenv*. La Figura 23 ilustra la ubicación de este archivo.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fig. 23 Archivo de configuración .env

La figura 24 presenta el contenido del archivo *.env* que se emplea en este proyecto, dentro del archivo, la variable *PORT* especifica el puerto donde se ejecutará el servidor *Express*, la variable *MONGO\_URI* contiene la cadena de conexión necesaria para acceder a la base de datos denominada *Certificados\_idiomas*, por otro lado, *JWT\_SECRET* almacena la clave secreta utilizada para garantizar la autenticación segura de los usuarios, mientras que *PRIVATE\_KEY* representa la clave privada que permite la interacción 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 un archivo denominado *db.js* dentro del directorio *config*, con el propósito de establecer la conexión entre el servidor y la base de datos, por lo cual, se utiliza la biblioteca *mongoose*, la cual permite interactuar con MongoDB a través de un modelo de datos orientado a objetos (ODM), la figura 25 muestra un fragmento de código en el que se define una función asíncrona llamada connectDB, cuya responsabilidad es gestionar esta conexión de forma segura y eficiente.

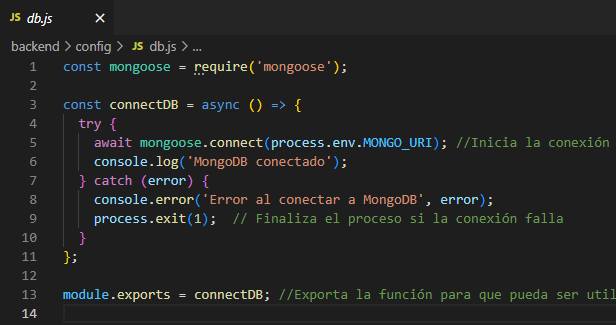


Fig. 25. Conexión con la base de datos MongoDB

**c. Estructura para almacenar los certificados**

Para gestionar el almacenamiento de los certificados de idioma en la base de datos MongoDB, se implementa un modelo denominado *Certificate.js*, ubicado en el directorio *models*, en este archivo se establece el esquema *certificateSchema*, el cual define la estructura lógica que deben seguir todos los registros de certificados, este esquema contempla los campos esenciales para describir un certificado del idioma, entre ellos: el nombre de la institución emisora, la facultad correspondiente, los datos personales del estudiante, el idioma evaluado, el nivel alcanzado, la fecha de emisión y el estado de validez. La figura 26 ilustra esta estructura.

Texto

El contenido generado por IA puede ser incorrecto.

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

**d. Estructura para almacenar los usuarios**

Con el objetivo de administrar los usuarios del sistema, se implementa un modelo llamado *User.js*, ubicado en el directorio *models*, este modelo emplea la biblioteca *Mongoose* para definir la estructura de los datos y facilitar su interacción con la base de datos MongoDB. Asimismo, se incorpora la librería *bcryptjs* para proteger las contraseñas mediante procesos de encriptación, como se muestra en la figura 27, además, el modelo incluye un método llamado *comparePassword*, que permite verificar de forma segura si la contraseña ingresada por el usuario durante la autenticación coincide con la versión encriptada almacenada en la base de datos.

Texto

El contenido generado por IA puede ser incorrecto.

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

**e.** **Definición de rutas de autenticación de usuarios**

El archivo *authRoutes.js*, localizado en el directorio *routes*, contiene la definición de las rutas asociadas al proceso de autenticación y registro de usuarios en el sistema, por tanto, se emplea el framework *Express*, que permite crear un enrutador encargado de gestionar las solicitudes HTTP relacionadas con estas funciones, como se ilustra en la figura 28.

La ruta *POST /register* está destinada al registro de nuevos usuarios, al recibir una solicitud se extraen del cuerpo (req.body) los datos correspondientes al correo electrónico, contraseña, cédula y rol, posteriormente, se crea una nueva instancia del modelo *User.js*, la cual se almacena en la base de datos.

Por otro lado, la ruta *POST /login* permite validar las credenciales de acceso, primero se comprueba si existe un usuario con el correo ingresado y si la contraseña proporcionada coincide con la registrada, en caso de una autenticación exitosa, se genera un token JWT (JSON Web Token), firmado mediante una clave secreta definida en el archivo *.env*. este token incluye el identificador y el rol del usuario, y será utilizado para autorizar el acceso a rutas que requieren autenticación.

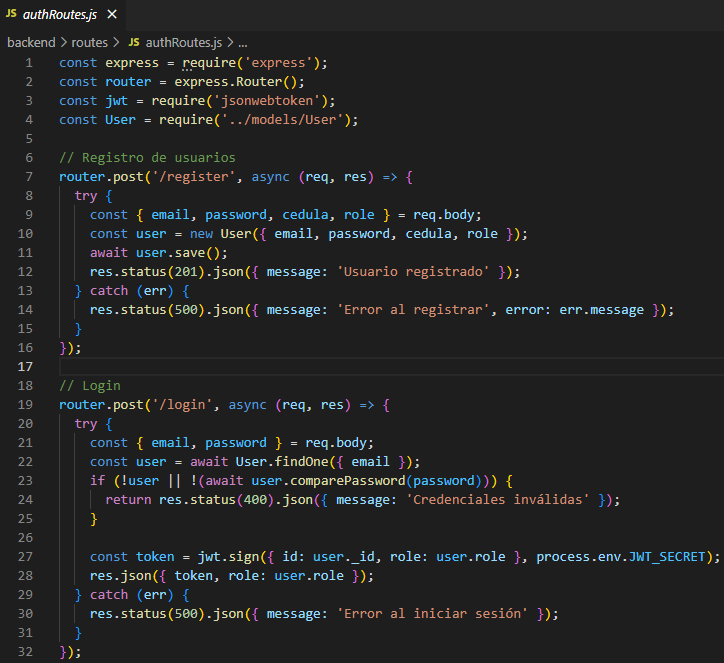


Fig. 28. Autenticación de los roles de los usuarios.

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

El archivo *certificateRoutes.js*, situado en el directorio *routes*, define las rutas encargadas de la gestión de los certificados de idioma dentro del sistema, estas rutas están implementadas utilizando el framework Express y se enlazan directamente con funciones específicas que se encuentran en el archivo *certificateController.js*. La figura 29 muestra el contenido de dicho archivo y su estructura funcional.

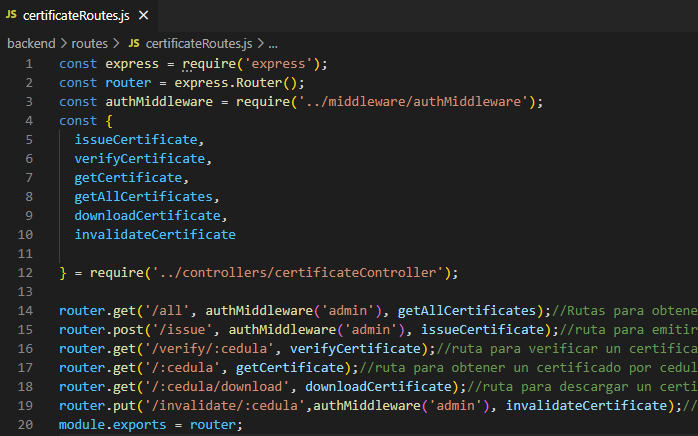


Fig. 29. Rutas para la gestión de certificados

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 | Administrador | Esta ruta permite recuperar la lista completa de certificados registrados en el sistema. |
| POST /issue | Administrador | Permite la emisión de un nuevo certificado mediante el registro de los datos correspondientes 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

**g. Desarrollo de la lógica de negocio**

La figura 31 presenta el contenido del archivo *certificateController.js*, el cual concentra la lógica principal del servidor relacionada con la gestión de certificados del idioma, este archivo funciona como el controlador central encargado de procesar las solicitudes dirigidas al backend en lo referente a certificados, estableciendo la conexión con la base de datos MongoDB, gestionando la generación de archivos PDF y realizando la interacción correspondiente con la red blockchain basada en Ethereum.

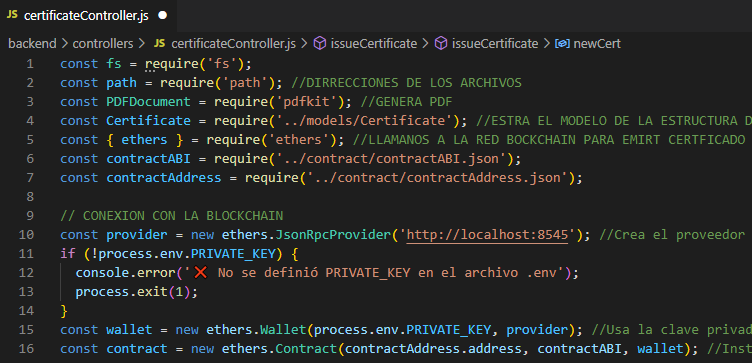


Fig. 31. Contenido del archivo certificateController.js

La tabla 14 resume las funciones implementadas para manejar la lógica asociada a los certificados del idioma, todas estas funciones están definidas en el archivo *certificateController.js*, el cual actúa como componente central en la gestión de operaciones tales como la emisión, verificación, descarga, consulta e invalidación de certificados.

|  |  |
| --- | --- |
| **Función** | **Descripción** |
| issueCertificate | Emite y almacena un certificado tanto en la blockchain como en MongoDB. |
| verifyCertificate | Comprueba la validez del certificado, determinando si es legítimo o inválido. |
| getCertificate | Recupera y retorna toda la información de un certificado utilizando la cédula como referencia. |
| getAllCertificates | Permite al administrador visualizar la lista completa de certificados registrados. |
| downloadCertificate | Genera un archivo PDF que contiene los datos del certificado correspondiente. |
| invalidateCertificate | Marca un certificado como inválido, procediendo a su revocación formal dentro del sistema. |

Tab. 14. Funciones del archivo certificatecontrolle.js

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

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 32. Implementación de la función para emitir un certificado

Pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Fig. 33. Implementación de la función para verificar su validez

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Fig. 34. Implementación de la función para obtener un certificado

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 35. Implementación de la función para obtener todos los certificados

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 36. Implementación de la función para generar un archivo PDF

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 37. Implementación de la función invalidar un certificado

Por último, la figura 38 presenta el contenido del archivo *auth.controller.js*, ubicado en la carpeta *controllers*, este archivo contiene la lógica responsable del registro de nuevos usuarios en la plataforma, su función principal es verificar que el correo electrónico proporcionado no se encuentre registrado previamente y, en caso de que no exista, proceder a la creación y almacenamiento de un nuevo usuario en la base de datos MongoDB, utilizando el modelo *User.js*, esta funcionalidad asegura un control eficiente sobre los procesos de autenticación y 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.

Previo al desplegar, 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.

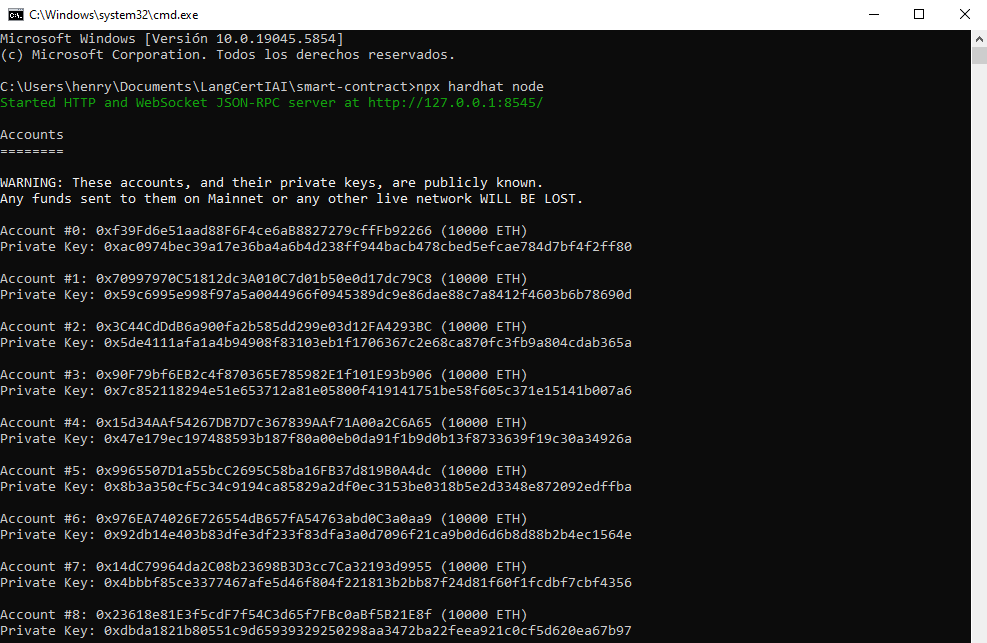


Fig. 39. Inicialización del nodo local de Ethereum

Una vez configurado el entorno y desarrollado el contrato inteligente, se procede a ejecutar el comando *npx hardhat compile* dentro de la carpeta *smart-contract*, utilizando la consola de comandos del sistema, esta instrucción tiene como propósito verificar la sintaxis del contrato y compilarlo, generando su correspondiente *bytecode*, el cual es apto para ser desplegado en la red Ethereum. En caso de que no se detecten errores durante la compilación, se crea automáticamente la carpeta *artifacts/,* la cual contiene el ABI y otros metadatos esenciales para permitir la interacción de la aplicación con el contrato, este proceso se ilustra en la figura 40.

Texto

El contenido generado por IA puede ser incorrecto.

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

Con el nodo local en ejecución, es posible desplegar el contrato inteligente sobre dicha red utilizando el comando *npx hardhat run scripts/deploy.js --network localhost,* esta instrucción ejecuta el script de despliegue *deploy.js*, lo que permite obtener la dirección del contrato en la red simulada, esta dirección es fundamental para establecer la integración con el backend de la aplicación, ya que permite emitir, verificar e invalidar certificados a través de llamadas directas al contrato inteligente.

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

Una vez completado el despliegue, el contrato inteligente queda incorporado dentro de una red blockchain de prueba local, este entorno simulado permite replicar las condiciones de ejecución de una red real, ofreciendo un espacio seguro para validar el correcto funcionamiento de la interacción entre la aplicación y el contrato inteligente, de este modo, se puede asegurar la estabilidad del sistema antes de avanzar hacia su integración con una interfaz web.

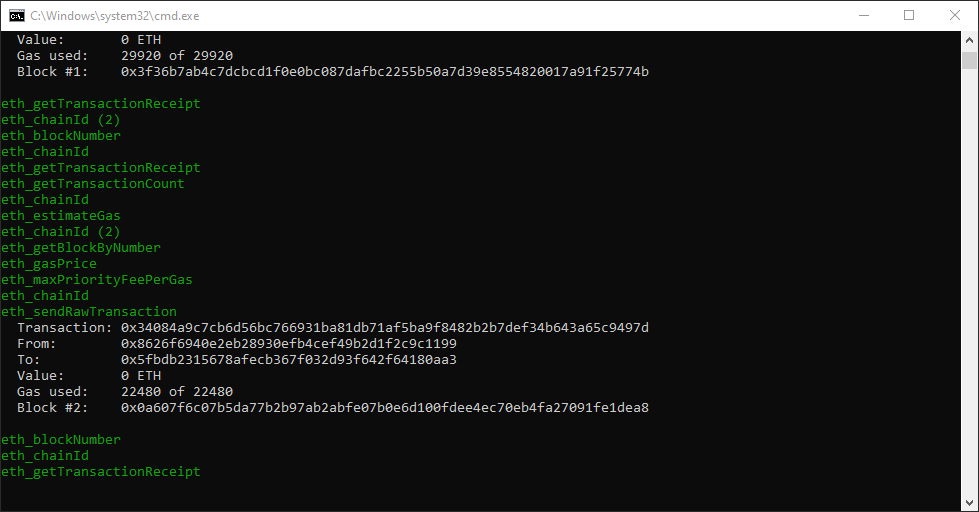


Fig. 41. Transacción correspondiente al despliegue del contrato

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

Tras la correcta compilación y despliegue del contrato inteligente en la red blockchain local proporcionada por Hardhat, el siguiente paso consiste en desarrollar una aplicación web que permita la interacción directa con dicho contrato, por tanto, se emplea la herramienta Vite, la cual simplifica la creación de proyectos utilizando la biblioteca React, ofreciendo un entorno ágil y moderno para el desarrollo del fronend.

**a. Configuración del entorno Fronend**

La configuración del entorno fronend se realiza desde el directorio del proyecto, específicamente dentro de la carpeta *backend*, desde la línea de comandos, se ejecuta el comando *npm create vite@latest fronend*, el cual permite generar un nuevo proyecto fronend utilizando la herramienta Vite, esta herramienta proporciona una estructura moderna y optimizada para el desarrollo de aplicaciones web, durante el proceso de creación, se selecciona *React* como biblioteca principal, lo que facilita la construcción de una interfaz modular y dinámica. Una vez creado el proyecto, se instalan todas las dependencias necesarias definidas en el archivo *package.json* mediante el comando *npm install*, asegurando así el correcto funcionamiento de las librerías requeridas, como se muestra en la figura 42. Finalmente, para iniciar la interfaz de desarrollo provista por Vite, se ejecuta el comando *npm run dev*, lo que levanta un servidor local accesible desde la dirección *http://localhost:5173*, mostrando una página de inicio 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 iniciar con la programación de la interfaz de usuario, es fundamental incorporar un sistema de estilos que sea flexible y eficiente, para ello, se procede a instalar Tailwind CSS versión 3, junto con postcss y autoprefixer, ejecutando el comando *npm install -D tailwindcss@3 postcss autoprefixer* desde la terminal. Tailwind CSS permite diseñar interfaces visuales mediante el uso de clases utilitarias, lo que facilita un desarrollo más ágil y controlado, mientras que, Postcss y Autoprefixer se encargan de procesar los estilos y garantizar su compatibilidad con distintos navegadores.

Tras completar la instalación, se generan los archivos de configuración *tailwind.config.cjs* y *postcss.config.cjs* mediante el comando *npx tailwindcss init -p*, lo que permite personalizar el sistema de estilos según las necesidades del proyecto, como se muestra en la figura 43.

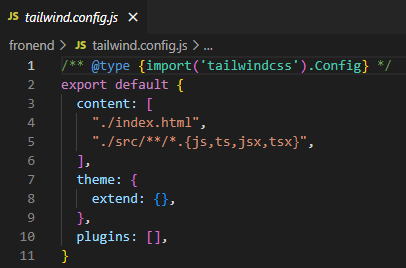


Fig. 43. Contenido del archivo tailwind.config.js

Para mejorar la experiencia de navegación entre distintas vistas como la página de inicio, el formulario de emisión o la lista de certificados sin necesidad de recargar completamente la página, se procede a instalar la biblioteca de enrutamiento para React mediante el comando *npm install react-router-dom,* esta herramienta permite gestionar de manera eficiente las rutas dentro de la aplicación, adicionalmente, se instala la biblioteca Axios utilizando el comando *npm install axios*, la cual facilita la realización de peticiones HTTP desde el cliente, esta biblioteca es clave para establecer la comunicación entre el fronend y el backend, ya que permite enviar solicitudes, recibir respuestas y manejar datos provenientes tanto del servidor como de la red blockchain.

A continuación, se define la estructura general de la aplicación, la cual se muestra en la figura 44, este archivo HTML funciona como plantilla principal de carga para la aplicación web, dentro de esta estructura se establece la configuración inicial y el contenedor donde el código JavaScript será renderizado, permitiendo construir y desplegar la interfaz de usuario de forma dinámica.



Fig. 44. Contenido del archivo index.html

El archivo *main.jsx* representa el punto de entrada principal de la aplicación desarrollada con React, tal como se muestra en la figura 45, su función principal es importar los estilos requeridos y renderizar el componente raíz de la aplicación, denominado App, permitiendo así iniciar la carga y visualización de la interfaz de usuario.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 45. Contenido del archivo main.jsx

El archivo *App.jsx* cumple el rol de componente central dentro de la aplicación, y se encarga de administrar la navegación entre vistas y el control de acceso a través de la biblioteca *React Router*, como se ilustra en la figura 46, en este archivo se incorporan tanto los componentes visuales como funcionales requeridos para el funcionamiento del sistema, además de los estilos definidos con Tailwind CSS.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 46. Contenido del archivo App.jsx

El archivo *Navbar.jsx* cumple la función de facilitar la navegación entre las distintas secciones del sistema, este componente adapta su comportamiento de acuerdo con el estado de autenticación del usuario y el rol que tenga asignado, tal como se muestra en la figura 47, para ello, accede al almacenamiento local del navegador (localStorage) con el propósito de verificar si existe un token de autenticación y determinar si el usuario es un administrador o un estudiante.

Texto

El contenido generado por IA puede ser incorrecto.

Fig. 47. Contenido del archivo Navbar.jsx

Dentro de la carpeta *pages* se encuentran las diferentes vistas que conforman la interfaz gráfica de la aplicación, cada una desarrollada con extensión .jsx, estas páginas representan las distintas opciones de navegación disponibles en el sistema, tales como: Inicio, Verificar, Emitir, Certificados, Registrar, Consultar y Cerrar sesión. La visibilidad de estas vistas varía en función del rol del usuario que haya iniciado sesión. Además, se incluye la página principal Home.jsx, que actúa como pantalla de bienvenida al sistema. La organización de esta carpeta se representa 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* representa la vista principal del sistema, tal como se observa en la figura 49, y se muestra al usuario al acceder a la ruta raíz de la aplicación, este componente permite gestionar el estado asociado al rol del usuario, lo que posibilita personalizar la interfaz de acuerdo con el perfil correspondiente, ya sea administrador o estudiante.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Fig. 49. Página principal del componente Home.jsx

El archivo *Login.jsx* contiene el componente encargado de gestionar el proceso de autenticación de usuarios dentro del sistema, como se muestra en la figura 50, esta vista permite a los usuarios iniciar sesión utilizando sus credenciales institucionales, ingresando su correo electrónico y contraseña, con el fin de acceder a las funcionalidades específicas asociadas a su rol dentro de la plataforma.

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* contiene el componente responsable del registro de nuevos usuarios en el sistema, esta página presenta un formulario diseñado para recopilar información clave, incluyendo el correo institucional, la contraseña, la cédula de identidad y el rol del usuario tal como se ilustra 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* implementa el componente encargado de gestionar la emisión de certificados digitales dentro del sistema, una funcionalidad reservada exclusivamente para usuarios con rol de administrador, esta página presenta un formulario a través del cual se recopilan datos esenciales, como el nombre del instituto, la facultad, la información del estudiante como el nombre y la cédula, el idioma cursado y el nivel alcanzado, según se observa 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* implementa el componente responsable de verificar la autenticidad de los certificados digitales emitidos por el sistema, tal como se muestra en la figura 53, esta funcionalidad está disponible para cualquier usuario y permite comprobar la validez de un certificado ingresando el número de cédula del titular correspondiente.

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* contiene el componente encargado de permitir a los usuarios consultar y visualizar certificados emitidos por el sistema, utilizando como criterio de búsqueda el número de cédula del estudiante, tal como se muestra en la figura 54, esta funcionalidad está dirigida principalmente a los estudiantes, con el propósito de facilitar el acceso a la información detallada 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 destinado a la visualización, búsqueda, descarga e invalidación de los certificados emitidos en el sistema, esta interfaz ha sido diseñada específicamente para usuarios con rol de administrador, ya que proporciona una vista general y permite ejercer control sobre la gestión de los certificados existentes, tal como se observa 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, fortaleciendo la transparencia e integridad de los registros académicos, facilitando que los estudiantes puedan demostrar sus logros de forma segura y 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 IAI de la Universidad Central del Ecuador. Adicionalmente, se estableció una arquitectura web de tipo multicapa y se evaluaron distintas plataformas blockchain entre ellas Ethereum, Hyperledger Fabric, Solana y Polkadot, seleccionando Ethereum por su pertinencia y adecuación a las características del 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

En la fase de la propuesta del prototipo, se enfoca en la planificación y recopilación de información clave, que 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 garantizo 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, demostró ser una solución eficaz para garantizar la autenticidad y validez de los certificados académicos, la característica inalterable de la cadena de bloques 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 propósito de asegurar que únicamente 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

**Anexo A. Entrevista**

Entrevista estructurada dirigida al personal encargado del proceso de emisión y validación de certificados de culminación de un idioma en el Instituto Académico de Idiomas, para el trabajo de grado de titulación “*Prototipo de aplicación para la certificación de documentos digitales en instituciones de educación superior basado en la tecnología Blockchain.”*

**Objetivo**

Obtener información detallada sobre el proceso actual de emisión y verificación de certificados académicos, con el fin de identificar oportunidades de mejora mediante el uso de tecnología blockchain.

**Preguntas:**

1. ¿Cuál es su nombre completo?

2. ¿Qué cargo desempeña actualmente en el Instituto Académico de Idiomas?

3. ¿Qué requisitos debe cumplir un estudiante para obtener el certificado de suficiencia de un idioma?

4. ¿Quién es la persona encargada de autorizar la emisión del certificado una vez verificados los requisitos del estudiante?

5. ¿Qué tipo de información del estudiante se registra o almacena al momento de emitir el certificado?

6. De la información almacenada, ¿Cuál considera que es de carácter confidencial y no debe divulgarse públicamente?

7. ¿Qué datos pueden considerarse públicos y ser visibles en un sistema de verificación sin comprometer la privacidad del estudiante?

8. ¿Dónde se almacena actualmente la información de los certificados emitidos? (sistema, base de datos, archivo físico, etc.)

10. ¿Quiénes pueden solicitar o consultar la validez de un certificado académico emitido por el instituto?

11. ¿Cuál es el procedimiento actual para verificar la validez de un certificado?

12. ¿Qué dificultades o inconvenientes se han presentado en el proceso de verificación de certificados?

13. ¿Podría describir el procedimiento que se sigue actualmente para la emitir los certificados académicos?