

CREACION DE SOFTWARE DE ALMACENAMIENTO DE DATOS ELECTORALES COLOMBIANOS CON BLOCKCHAIN

Sebastian Guevara Sanchez

2025 Julio 26

Colombia – Valle del cacuca - Santiago de cali

Contenido

GLOSARIO	4
INTRODUCCION	6
PLANTEAMIENTO DEL PROBLEMA:	7
OBJETIVO GENERAR Y ESPECIFICO	8
Objetivo general.....	8
Objetivos específicos.....	8
ESTADO DE ARTE	9
HISTORIA DE LAS VOTACIONES UNIVERSALES	9
HISTORIA DE LAS VOTACIONES ELECTRÓNICAS	9
Marco Legal en Colombia	11
Marco legal internacional:	11
METODOLOGIA.....	12
Diseño de interface de usuario	12
Diseño de base de datos	14
Ingreso de información al software	15
Encriptación de la información	16
Configuración de la base de datos mongodb.....	16
Diseño de pruebas del software	17
Desarrollo	18
Creación de interfaces de usuario:.....	18
Estructura de la base de datos	20
Adquisición de información de votantes	21
Creación de JSON con los datos los votos	21
Encriptación de información	22
Funcionamiento de la base de datos:	22
Prueba del software	24
ANALISIS ECONOMICO:	29
Análisis y cronograma	29
Historia del retorno sobre la inversión (ROI)	30
Metodología de análisis económico.....	30
Proyección de Costos Operativos Anuales.....	31
Cálculo del ROI	31

CONCLUSIONES	33
REFERENCIAS BIBLIOGRAFICAS	34

GLOSARIO

- **Blockchain:** Es la tecnología que guarda y enlaza bloques de información con encriptación robusta y en diferentes nodos de base de datos al tiempo, haciendo que la información sea poco probable de modificación, porque modificaría todo el hash criptográfico
- **Seguridad:** Reglas para proteger la información contra fuga de información, bloqueo de servicios o modificación de datos, asegurando que los datos sean confidenciales, íntegros y disponibles cuando se necesiten.
- **Ataques:** Acciones mal intencionadas de acceder, alterar, robar o destruir información y sistemas. En el contexto de Blockchain, estos pueden incluir ataques de red, ataques de doble gasto, y otros métodos que afecte la integridad del sistema.
- **Resultados electorales:** Datos relacionados a los candidatos de las elecciones, acompañado de los votos conseguidos.
- **Transparencia:** Visibilidad y verificación por las partes interesadas, asegurando así confianza y responsabilidad.
- **Descentralización:** Cuando la responsabilidad de el almacenamiento de datos es de diferentes servidores al tiempo
- **Escalabilidad:** Cuando un software puede aumentar en la carga de trabajo de manera eficiente, sin degradar el rendimiento.
- **Eficiencia:** La habilidad de un sistema para maximizar la productividad con el menor uso posible de recursos, tiempo y esfuerzo.
- **Almacenamiento:** Proceso y tecnología utilizada para guardar datos de manera segura y accesible para su posterior recuperación y uso.
- **Criptografía:** Ciencia que cifra y descifra información, asegurando que solo las partes autorizadas puedan acceder y modificar los datos.
- **Votaciones:** Proceso mediante el cual los electores expresan sus preferencias sobre candidatos o propuestas en una elección, que posteriormente se convierten en resultados electorales.
- **Tecnología:** Conjunto de conocimientos y herramientas utilizados para diseñar, crear y mejorar productos, servicios y procesos.
- **Desarrollo de software:** Proceso de levantar requisitos, diseñar, crear, probar un software.
- **Integridad de datos:** La característica que informa que los datos no han sido modificados.
- **Auditoría:** Actividad de probar y evaluar un software.
- **Procesos administrativos:** Decisiones relacionadas con la administración del software
- **NoSQL:** Forma de almacenamiento de información en colecciones JSON y no en tablas SQL
- **JSON:** Es una forma de tener información por medio de una lista de datos, sirve para agilizar las búsquedas

- Inmutabilidad: Característica de un sistema donde los datos, una vez escritos, no pueden ser alterados ni borrados, garantizando su permanencia y fiabilidad.
- Herramientas de programación: Grupo de sistemas que nos ayudan a crear código de un software.
- Metodologías ágiles: Enfoques de desarrollo de software que promueven la flexibilidad, la colaboración y la capacidad de respuesta rápida a los cambios mediante ciclos de trabajo iterativos e incrementales.
- ROI (Retorno de Inversión): Métrica financiera que evalúa la rentabilidad de una inversión comparando el beneficio neto obtenido con el costo total de la inversión.
- Pruebas de software: Proceso de evaluar y verificar que un software funcione como se espera, identificando y corrigiendo errores y defectos.
- Interfaces de usuario: La parte grafica del software donde el usuario hace uso del programa creado.
- MongoDB: Base de datos NoSQL que guarda los datos en JSON, ofreciendo alta escalabilidad y flexibilidad para gestionar grandes volúmenes de datos.

INTRODUCCION

Nombre : Sebastian Guevara Sanchez

Sector : Votaciones electorales

Departamento : Colombia

Municipio : Colombia

Desde el principio de la documentación de las elecciones, ha existido la desconfianza en los resultados, sin tener formas sencillas de validar la información. Esto ha llevado a crear diferentes formas de mejorar el almacenamiento y acceso a esos datos, llegando a la era informática de la 4 Revolución Industrial. Teniendo en cuenta lo anterior, desde el 2008 se empezó a profundizar el almacenamiento de datos descentralizada y enlazadas con un hash criptográfico. Esto ayuda a que la información recolectada de las elecciones sea inmodificable luego de ser almacenada (Swan, 2015). La descentralización que propone el Blockchain ayuda a distribuir los datos entre diferentes nodos de almacenamiento, dando seguridad al momento de guardar la información (Nakamoto, 2008). La creación e implementación de software de elecciones tienen varias leyes que los reglamentan, siendo este un gran desafío para la academia. En este documento, se presenta un análisis de los aspectos legales en España, Colombia y a nivel internacional, incluyendo referencias y citas de legislaciones y estudios relevantes. El objetivo principal de este proyecto es desarrollar e implementar un software para el almacenamiento de resultados electorales utilizando tecnología Blockchain. Este software tiene la importancia de actualizar los softwares electorales al Blockchain intentando lograr en las elecciones, precisos, inmutables y fácilmente verificables por cualquier parte interesada (Nofer et al., 2017).

PLANTEAMIENTO DEL PROBLEMA:

El problema detectado es la falta de seguridad en el almacenamiento de información electoral en diferentes partes del mundo, afectando totalmente la transparencia e integridad de los procesos electorales. Esta debilidad en la entidad que protege las elecciones, da desconfianza en la institucionalidad responsable. Las soluciones pueden ser varias, pero en la Revolución industria 4.0 nos da la oportunidad de ver Blockchain como un método de la era informática. Este sistema buscará garantizar la inmutabilidad y verificabilidad de los datos electorales, eliminando la posibilidad de manipulación y aumentando la transparencia del proceso. El fin de este software es proporcionar un sistema seguro y transparente que se pueda usar para fortalecer el almacenamiento de información electoral, intentando devolver la confianza en las instituciones relacionadas a las elecciones.

OBJETIVO GENERAR Y ESPECIFICO

Objetivo general

Crear e implementar un software de almacenamiento con tecnología Blockchain de resultados de votaciones con tecnología.

Objetivos específicos

Indagar y seleccionar la mejor forma de construcción del software de almacenamiento de resultados electorales basado en Blockchain.

Indicar protocolos de encriptación y seguridad para proteger la privacidad y confidencialidad de los datos electorales almacenados en la Blockchain.

Desarrollar interfaces graficas amigables con el usuario para su uso fácil.

Realizar pruebas a la encriptación y el almacenamiento de la información

ESTADO DE ARTE

En varias partes del mundo están empezando a usar el Blockchain como método de almacenar la información de forma segura. En Estonia y Suiza han mejorado la percepción de las elecciones por la implementación de Blockchain (Glaser, 2017). Entre las varias implementaciones se destaca una mejor seguridad y transparencias en las elecciones destacando su capacidad para ofrecer una solución más segura y transparente (Glaser, 2017).

HISTORIA DE LAS VOTACIONES UNIVERSALES

Las primeras elecciones de representantes se documenta desde las asambleas de la antigua Grecia y Roma 590ac , tenía muchas limitaciones para la más antidemocrática era que solo los hombres podían participar y solo las elites económicas se podían postular en estas elecciones, los avances democráticos en las elecciones de representantes se vieron pausados hasta la Revolución Francesa 1793 donde se amplió la posibilidad de elegir a los vasallos y posteriormente a los esclavos en el mundo. Los derechos del hombre alejaron el rechazo a las ideas de libertad económica y a los hombres nacidos en un territorio le daban la oportunidad de elegir líderes. (J. W. Lamare, 1981, p.256-257). Entre los años 1800 y 1900 se adelantó libertades para los mujeres, la fecha en que se decreta en cada país es 1869 en estados unidos, en 1893 nueva Zelanda, en 1895 Australia, entre 1906 y 1921 Finlandia, Noruega, Dinamarca y Suecia, habilitaron el voto femenino. Los países que decidieron darle la oportunidad de votar a las mujeres luego de la segunda guerra mundial 1943 fue Italia, Bélgica y Francia (J. W. Lamare, 1981, p.256-257). En Colombia se aprobó el voto femenino en 1954 durante la dictadura de Gustavo Rojas Pinilla, y algunos de los países donde no se avanzado en la democracia femenina son Afganistan, Arabia Saudita, Kuwait y Líbano (J. W. Lamare, 1981, p.256-257)

HISTORIA DE LAS VOTACIONES ELECTRÓNICAS

Las elecciones de líderes por medios electrónicos se empezó a documentar desde 1990 con el software de Brasil llamado DRE, luego en el 2000 en estados unidos en el estado de Arizona dio la posibilidad de votar por medio de internet (Business Wire, 1999). Del 2000 al presente ha habido muchos países que han avanzado en las votaciones electrónicas (ACE Project, 2024), en el año 2013 más de 31 países empezaron a usar formas electrónicas de votaciones. DRE en ingles direct-recording electronic voting machine (DRE) en español voto electrónico recogido directo, es una aplicación que ofrece el servicio recibir y almacenar información de una votación (United States Election Assistance Commission, 2005). Esta aplicación organiza en los votos y los muestra al final de cada tiempo de votación, imprime en pantalla los votos totales y los de cada individuo, esta información puede ser extraída en un disco duro y llevada a una central donde se hará el conteo 1996 (United States Election Assistance Commission, 2005). Este software también ayudaba a que las personas con discapacidad pudieran votar. En Brasil innovaron otra vez en el e-voting de las elecciones electrónica de ellos, este software se soporta en sistemas operativos Linux (Tribunal Superior Electoral, 2005). En el 2012 se creó la posibilidad de validación biométrica en varios países (Tribunal Superior Electoral, 2005). El DRE siguió fortaleciendo su almacenamiento de datos, sin llegar a implementar el Blockchain, pero se implementó el llamado VVPAT que sirve de comprobante de la votación, esta aplicación también tiene la particularidad de que el candidato

puede ver las personas que votaron por el (India Today. New counting method for Assembly polls, 2008).

En el país de Estonia también hubo avances tecnológicos, en la actualidad mas del 30% de sus elecciones son de manera electrónica y por internet, aun cuando hay posibilidad de hacerlo en papel (e-Estonia, 2024). La arquitectura que usan es de llave publica, pero cada votación mantiene la privacidad del votante, dando difícil ver la identidad de la persona que voto, se comprueba por medio de la SIM del celular del votante (Tribunal Superior Electoral, 2005). La aplicación Follow My Vote es un programa digital que ayuda a almacenar información en Blockchain y fue creada por empresas públicas de Estados Unidos, esto ayuda a identificar a los votantes y también es de código abierto, y fue escrita en el lenguaje de programación C++. En Noruega entre el 2011 y el 2013 se empezó la digitalización de las votaciones en el parlamento de ellos, el software que crearon fue Scytl y la empresa responsable de este software es Española, este software noruego recibió unos ataques informáticos que dañaron su imagen (J. Barrati-Esteve et al., 2012). En el territorio de China tienen el sistema de votación electrónica con tecnología Blockchain Liu y Wang, el cual cuenta con una estructura centrada en la privacidad o el anonimato por parte de los votantes, también cuentan con llave publica y firmas digitales de cada voto, cada grupo de personas relacionadas al software tienen roles específicos que sirven para ver los resultados en tiempo real, si se tiene alto privilegio (Liu, Yi et al., 2017).

Marco Legal en Colombia

Colombia, la normativa electoral y la legislación sobre protección de datos personales y tecnologías de la información son igualmente importantes para considerar la implementación de Blockchain en los procesos electorales. o Código Electoral Colombiano El Código Electoral establece las normas básicas para el desarrollo de elecciones en Colombia, garantizando la transparencia y la equidad del proceso electoral. La implementación de Blockchain en el sistema electoral colombiano requeriría ajustes en la normativa vigente para integrar esta tecnología de manera efectiva (Decreto 2241 de 1986). o Ley Estatutaria de Protección de Datos (LEPD) La LEPD, también conocida como Ley 1581 de 2012, regula el tratamiento de datos personales en Colombia. Cualquier sistema electoral basado en Blockchain debe cumplir con los principios de protección de datos establecidos en esta ley, asegurando que los datos de los votantes se traten de manera segura y confidencial (Ley 1581 de 2012).

o Decreto 1377 de 2013 Este decreto reglamenta la LEPD y establece las obligaciones de los responsables del tratamiento de datos personales. Para un sistema electoral basado en Blockchain, es crucial obtener el consentimiento de los ciudadanos y garantizar que los datos se utilicen únicamente para los fines autorizados (Decreto 1377 de 2013). o Ley 527 de 1999 La Ley de Comercio Electrónico regula el uso de tecnologías de la información y las comunicaciones en Colombia. Esta ley establece el marco legal para el uso de firmas digitales y la validez de los documentos electrónicos, lo cual es relevante para la implementación de Blockchain en el ámbito electoral (Ley 527 de 1999).

Marco legal internacional:

Internacionalmente hay diversas normativas con factores en común, algunas intentar regular la identidad del votantes y otras la seguridad en el almacenamiento, las cuales se pueden proteger con Blockchain y otras tecnologías de la actualidad, En la unión europea integral las directivas NIS, que establece reglas y protocolos de seguridad de las redes y sistemas, estos países ven viable la implementación de Blockchain para mejorar su seguridad en la información (Directiva (UE) 2016/1148 del Parlamento Europeo y del Consejo, 2016). También esta el convenio 108 del consejo de Europa con este instrumento se reglamentas las elecciones europeas. Cualquier software con Blockchain tiene que cumplir con las reglas del Convenio 108 (Consejo de Europa, 1981). o Ley Modelo de UNCITRAL sobre Comercio Electrónico La Ley Modelo de la Comisión de las Naciones Unidas para el Derecho Mercantil Internacional (UNCITRAL) proporciona un marco para la adopción de legislaciones sobre comercio

Blockchain en Suiza ha sido innovador con el Blockchain en las votaciones electrónica, dado más transparencia y credibilidad a sus elecciones (Ley Federal de la Innovación y Tecnología Blockchain, Suiza, 2020).

METODOLOGIA

La creación de este software de almacenamiento de datos electorales en Blockchain se hará con metodologías ágiles como SCRUM logrando un seguimiento inicial y en el desarrollo del proyecto con reuniones diarias, semanales y mensuales. Se hará una identificación de mejores herramientas y lenguajes usados con pruebas relacionadas a la seguridad de los datos (Beck et al., 2017). El desarrollo de software utilizando Blockchain requiere un enfoque especializado y el uso de metodologías ágiles como SCRUM para asegurar una implementación eficiente y segura (Beck et al., 2017).

Diseño de interface de usuario

Las interfaces web desarrolladas en este proyecto buscan ser muy amigables con el usuario y que lo puedan usar de manera fácil, se usara una barra de inicio para la navegación por las opciones del softwares, en la parte central de cada interfaces tiene un formulario de ingreso de datos, en las interfaces de muestra de datos, se muestra la información que tiene cada bloque creado con Blockchain, el frontend del software esta compuesto por las herramientas HTML, CSS, BOOTSTRAP y JAVASCRIPT, usando colores de la preferencia del cliente, recomendando colores claros y fondos oscuros para evitar el cansancio visual, un ejemplo se puede ver en la siguiente imagen.

Caracteristicas de la informacion	Datos que se van a ingresar
Nombre de la opcion 1:	
Cantidad de votos de la opcion 1:	
Nombre de la opcion 2:	
Cantidad de votos de la opcion 2:	

Subir

En la siguiente imagen se muestra cuando ya se creó el primer bloque.

Nombre de la opcion 2:	
Cantidad de votos de la opcion 2:	
Subir	

En la siguiente tabla, se muestra la informacion del bloque creado :

Partes del documento	Informacion dentro del documento
NumeroBloque	
nombre_1	
votacion_1	
nombre_2	
votacion_2	
HashAnterior	
HashActual	

Ver Bloques

La interfaz que muestra los datos guardados se llama “Ver Bloques” estando en la barra inicio de todo el software, donde se mostrara el listado de bloques creados con un ciclo for que recorre la colección en Blockchain también se puede devolver en las opciones de la barra de inicio, en estas interfaces se maneja anexándole el hash del bloque anterior y el hash del bloque actual, las herramientas usadas en esta interfaces son HTML, CSS, bootstrap y Javascrip, los colores que escogimos fue con el fin de economizar energía de la pantalla de los computadores con colores opacos y evitar el cansancio que se da con fondos muy brillantes. como se muestra en la figura siguiente.

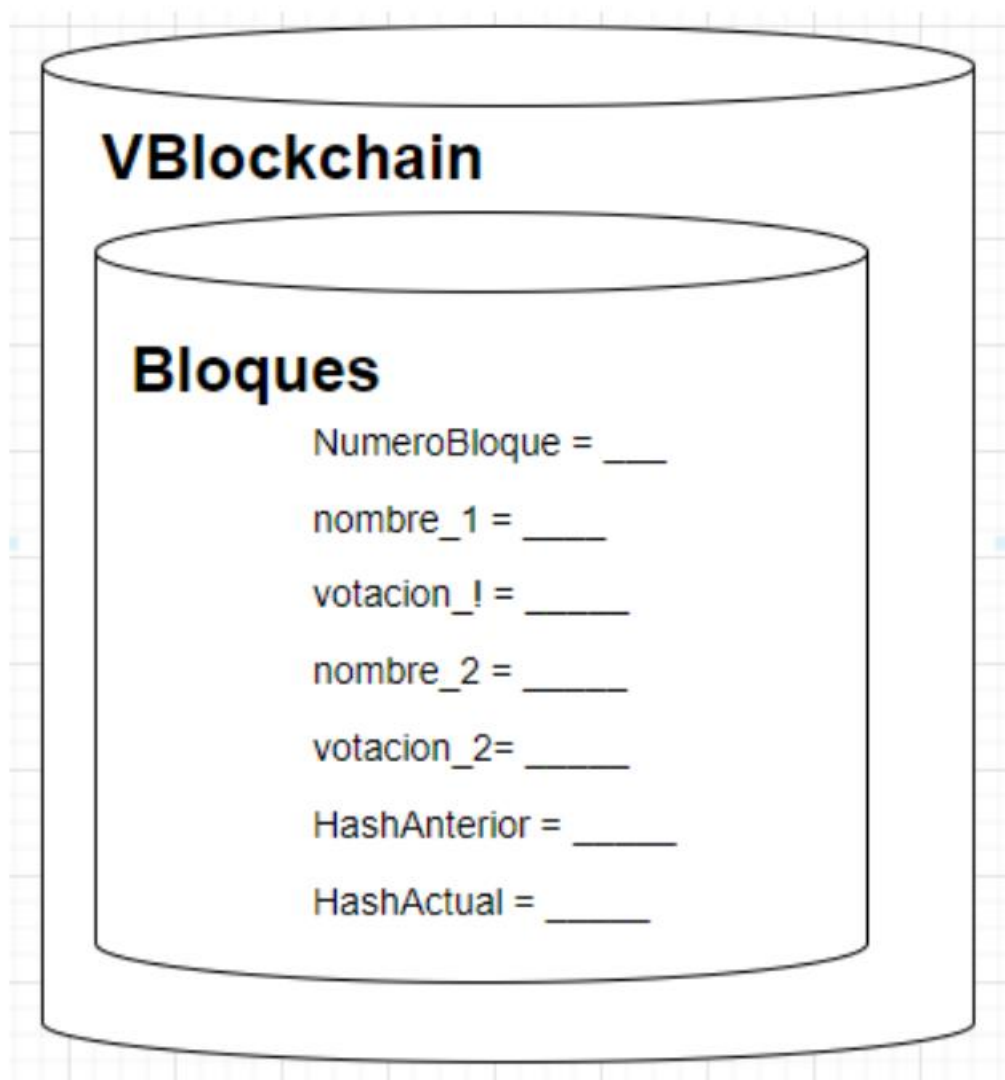
Las interfaces se comunican por medio de la barra de inicio como se muestra en la siguiente imagen.



Diseño de base de datos

El diseño de base de datos consistirá en datos JSON que ingresan a una base de datos MongoDB que esta distribuida entre varios servidor, en la implementación se usara la opción web de MongoDB, el nombre de la base de datos es “VBlockchain” y dentro de esta base de datos, la colección “Bloques” es donde se almacenara la información de los votos relacionados a cada candidato cada bloque tendrá un ID que es autoincrementar cada que se ingresa una planilla de votaciones o lista, cada bloque de datos creado, tendrá el hash del bloque anterior con el que se creara el hash actual para almacenarlo con el bloque, todos los datos se juntan en un JSON que ira a la base de datos mongoddb con Python, En la siguiente imagen se muestra un diagrama de base de datos donde nos muestra cómo se guardarán los datos.

Los datos que se le ingresaran al software serán por medio de un formulario HTML, con datos como el nombre del candidato en string y la cantidad de votos en numero entero, también e anexa el hash anterior del bloque y el hash actual de este, cuando ya se tengan todos los datos en JSON se enviara la información a la base de datos distribuida con PyMongo en la siguiente imagen se muestra detallada la información



Ingreso de información al software

Los datos se ingresarán por medio de un formulario y serán limitados a lo que quiera el cliente, en este ejemplo se agregaran 2 candidatos con su respectivos votos, mas el indicativo de la planilla y los hashes del Blockchain, la parte del frontend por medio de HTML, CSS y Bootstrap con un componente de PYTHON en la parte del backend, como se muestra en la siguiente imagen



Encriptación de la información

La encriptación de la información se realizará con el lenguaje de programación PYTHON y su librería hashlib, de esta librería usaremos la opción SHA256 la cual nos servirá para convertir toda la información del bloque de datos en 1 hash criptográfico de 32 bits llamado "hash actual" que posteriormente se guardará junto a la información, se escogió el SHA256 por la dificultad de descryptar esa variable encriptada.



Configuración de la base de datos mongodb

Luego de que los datos de la votación lleguen al lenguaje Python, usaremos el lenguaje Python y su librería PyMongo para crear el insert de los datos en la base de datos, antes de hacer la inserción de datos, se consulta el ultimo bloque creado para obtener su hash y ponerlo en el siguiente bloque como hash anterior, dado el caso de que no hay bloque creado, se le ingresa información cualquiera a la variable "Hash anterior".



Diseño de pruebas del software

En esta parte del proyecto se realizarán pruebas para confirmar la integridad de los datos almacenados, y validando que así se cambie 1 dígito de cualquier variable, el hash del bloque cambia totalmente, indicando que hubo un cambio en los datos del software.

Luego se probará creando varios bloques, con datos idénticos con la diferencia en 1 solo dato y así ver cómo se comporta el hash. Luego se cambiará una variable entera para validar que ni cambiando toda la palabra, el hash vuelve a salir igual a los datos iniciales.

La tercera prueba será modificar el bloque completo para validar el cambio en el hash actual de ese bloque, con uno anterior de diferentes datos las pruebas están enfocadas en demostrar la diferencia en los dígitos del hash criptográfico cuando hay un cambio en los datos.

Desarrollo

Esta parte del documento da los pasos para implementar la metodología anterior, en esta parte se dará un ejemplo de la creación de las interfaces de usuario, la estructura de la base de datos y su creación, de igual manera la información de los votos en el software, la creación del JSON con los datos del informe y luego almacenándolos para finalizar con pruebas al software.

El código fuente completo de la aplicación se encuentra en el siguiente github:

https://github.com/guevaraStian/Votaciones_Blockchain

Creación de interfaces de usuario:

El software web creado tiene 2 interfaces, una donde se ingresan los datos por un formulario y otra donde se ven los bloques informáticos creados. La interface primera tiene un formulario con un método POST de HTML y diferentes etiquetas input que sirven para insertarle la información al programa web, estos input tienen la opción de etiqueta class para el CSS y la opción name para enviar a la variable a la base de datos y herramienta type para indicar el tipo de dato, el formulario de la página uno, tendrá al final un botón submit para indicar el ingreso de los datos, la segunda página hace consulta a la base de datos y muestra los bloques separados por un for de Python, en estas 2 páginas web se usa CSS y BOOTSTRAP para darle color y dinámica a la interface. Con CSS y su framework BOOTSTRAP se crean los detalles del diseño de la página, escogiendo colores, tipos de letra, ubicación de las partes de la página y se configura para que sea responsive, que significa que funciona en cualquier dispositivo que abra la página web. En la siguiente imagen se muestra el código del formulario de ingreso de información que tiene la primera página.

```

<h1>Votacion electronica con Blockchain</h1>
<h2>Por favor ingrese la informacion de la votacion : </h2><br>
<form action="/upload" method="POST" enctype="multipart/form-data">
  <table class="table table-responsive table-bordered">
    <thead class="bg-danger">
      <tr>
        <th scope="col" > Caracteristicas de la informacion: </th>
        <td scope="col" > Datos que se van a ingresar: </td>
      </tr>
    </thead>
    <tbody>
      <tr>
        <th scope="row" class="bg-secondary">Nombre de la opcion 1:</th>
        <td scope="row" class="bg-secondary"><input type="text" name="nombre_1" class="btn btn-secondary" /></td>
      </tr>
      <tr>
        <th scope="row" class="bg-secondary">Cantidad de votos de la opcion 1:</th>
        <td scope="row" class="bg-secondary"><input type="text" name="votacion_1" class="btn btn-secondary" /></td>
      </tr>
      <tr>
        <th scope="row" class="bg-secondary">Nombre de la opcion 2:</th>
        <td scope="row" class="bg-secondary"><input type="text" name="nombre_2" class="btn btn-secondary" /></td>
      </tr>
      <tr>
        <th scope="row" class="bg-secondary">Cantidad de votos de la opcion 2:</th>
        <td scope="row" class="bg-secondary"><input type="text" name="votacion_2" class="btn btn-secondary" /></td>
      </tr>
      <tr>
        <th scope="row" class="bg-danger"><input type="submit" value="Subir" class="btn btn-secondary"/></th>
      </tr>
    </tbody>
  </table>

```

En la página web 2, su interface está compuesto un for que recorre un la respuesta de una consulta a la base de datos, siendo la consulta, todos los bloques creados y el for va mostrando cada dato de cada bloque, también tiene HTML para mostrar los datos teniendo la etiqueta card para cada bloque y con la opción tbody para mostrar los td y th para declarar filas y columnas de los datos consultados, se usa CSS para indicar los colores y las características de la interface, cada etiqueta HTML tiene su CSS en la característica class para mostrar la información con el diseño indicado se escogió colores de fondo opacos de luz con el fin de evitar el brillo en la pantalla, luego cerrada en esta pestaña de la página web. En la siguiente imagen se muestra el for que recorre los bloques para mostrarlos en pantalla.

```

46     {% block result %}
47
48     <div class="container" id="test">
49         <tbody id="body">
50             {% for i in listabloques%}
51                 <div class="card bg-secondary mb-3" >
52                     <div class="card-body">
53                         <tr>
54                             <td> <th> NumeroBloque = </th> {{ i["NumeroBloque"] }}</td>
55                             <br>
56                             <td> <th>nombre_1 = </th> {{ i["nombre_1"] }}</td>
57                             <br>
58                             <td> <th>votacion_1 = </th>{{ i["votacion_1"] }}</td>
59                             <br>
60                             <td> <th>nombre_2 = </th> {{ i["nombre_2"] }}</td>
61                             <br>
62                             <td> <th>votacion_2 = </th>{{ i["votacion_2"] }}</td>
63                             <br>
64                             <td> <th>HashAnterior = </th> {{ i["HashAnterior"] }}</td>
65                             <br>
66                             <td> <th>HashActual = </th> {{ i["HashActual"] }}</td>
67                         </tr>
68                     </div>
69                 </div>
70             {% endfor %}
71         </div>
72
73     {%endblock%}

```

Estructura de la base de datos

La aplicación web creada con este proyecto usará un motor de base de datos llamado MongoDB, fue escogido por ser innovador y tener estructura NoSql que con su forma de guardar datos en JSON se disminuye cantidad de almacenamiento por la misma cantidad de datos, y en MongoDB hay búsquedas más rápidas que en motores de bases de datos SQL, pero la razón fundamental es que la opción web de MongoDB nos da opción de almacenamiento distribuido en varios servidores al tiempo, con esto y el enlace de hash por bloque, se logra las características de Blockchain. La base de datos se controlará desde el Python del software, donde se crea la conexión, se consulta y se le ingresan datos. como se ve en la siguiente imagen

```

_id: ObjectId('664e68f79a315fc463002fcf')
NumeroBloque: "1"
nombre_1: "David"
votacion_1: "7"
nombre_2: "Darwin"
votacion_2: "9"
HashAnterior: "Genesis"
HashActual: "48f68a43f99713c5918cc0135f509b97228a3c96ee04893d310173e6b8805f00"

```

Queda almacenada en MongoDB como un JSON la información que se ingresa a el software, MongoDB queda en espera de la creación del siguiente bloque, ya que se le hará la consulta del hash anterior, para luego en Python crear el siguiente hash actual y con este hash actual crear el siguiente bloque.

Adquisición de información de votantes

En la inserción de datos se uso el lenguaje de etiquetas llamada HTML en cual nos da la opción de un FORM para dar opciones de ingresar datos a Python y luego MongoDB, el FORM se combina con la función POST que nos permite insertar o editar información que va en los espacios INPUTS estos datos ingresan al Python luego de dar click en un botón SUBMIT que le envía los datos a la función de Python llamada upload como se muestra en la siguiente imagen:

```

35 <h1>Votacion electronica con Blockchain</h1>
36 <h2>Por favor ingrese la informacion de la votacion : </h2><br>
37 <form action="/upload" method="POST" enctype="multipart/form-data">
46 <tr>
47 <th scope="row" class="bg-secondary">Nombre de la opcion 1:</th>
48 <td scope="row" class="bg-secondary"><input type="text" name="nombre_1" class="btn btn-secondary" /></td>
49 </tr>

```

Creación de JSON con los datos los votos

Luego de haber dado clic en el botón subir que es un SUBMIT, se procede a guardar cada INPUT por medio de su NAME de HTML y request.form de Python, cada dato se guarda en una variable, posteriormente se consulta la base de datos para ver el ultimo HASH creado y guardarlo en la variable hash anterior, con este ultimo dato se crea el has actual, y con esto se completa el JSON

con la información de las votaciones. La información de los inputs se almacena en las variables indicadas en la siguiente imagen.

```
23 #Este codigo se ejecuta cuando se activa guardar los datos
24 @app.route("/upload", methods=['POST'])
25 def uploader():
26     if request.method == 'POST':
27         #Se crean las variables y se guarda lo que viene del primer formulario
28         nombre_1 = request.form['nombre_1']
29         votacion_1 = request.form['votacion_1']
30         nombre_2 = request.form['nombre_2']
31         votacion_2 = request.form['votacion_2']
32         #Conexion a la base de datos "Blockchain" y seleccion de la coleccion "Bloques"
33         client = MongoClient('localhost', port=27017, username='', password='')
34         db = client['VBlockchain']
35         col = db['Bloques']
```

En la siguiente imagen se muestra como se almacena los datos en la base de datos

```
51 col.insert_one({'NumeroBloque': NumeroBloque, 'nombre_1': nombre_1, 'votacion_1': votacion_1,
52                'nombre_2': nombre_2, 'votacion_2': votacion_2, 'HashAnterior': HashAnterior, 'HashActual': HashActual
53                })
```

Encriptación de información

Este software contara con una encriptación robusta para la fecha de la propuesta, es el método de cifrado SHA256 el cual se usará por medio de Python y su librería hashlib primero se convierte en hash todo el JSON de información con el hash anterior y posteriormente ingresamos la opción hexdigest para convertir la variable en un hexadecimal de 32 caracteres que podemos guardar con facilidad, como se muestra en la siguiente imagen.

```
8 #hash para la encriptacion de la cadena de bloques
9 import hashlib

46 #Se encripta toda la informacion con sha256 anexando el hash
47 encr1 = hashlib.sha256(NumeroBloque.encode()+nombre_1.encode()+votacion_1.encode()
48                        +nombre_2.encode()+votacion_2.encode()+HashAnterior.encode())
49 HashActual = encr1.hexdigest()
```

Funcionamiento de la base de datos:

En la creación y establecimiento de la conexión con la base de datos MongoDB se usará Python y la librería PyMongo, como se muestra en la siguiente imagen

```
10 #Se guarda en una base de datos mongodb
11 from pymongo import MongoClient
```


En Python se usa la librería PyMongo el cual tiene la función MongoClient, donde se escribe en texto la conexión a la base de datos, con el usuario y contraseña luego de crear la base de datos se crea la colección Bloques como se muestra en la siguiente imagen.

```
32 #Conexion a la base de datos "Blockchain" y seleccion de la coleccion "Bloques"
33 client = MongoClient('localhost', port=27017, username='', password='')
34 db = client['VBlockchain']
35 col = db['Bloques']
```

Después de tener la conexión a la base de datos indicada, se usa PyMongo para realizar una consulta de 1 solo dato indicando que el numerobloque sea el 1, dado el caso de que no este creado el primer bloque se solicita la información del bloque anterior, de donde se sacara el hash para el siguiente bloque, si es el primer bloque que se crea se le pondrá una variable aleatoria para el hash anterior.

```
37 #Se guarda en la variable "colvacia" si la coleccion "Bloques" esta vacia
38 colvacia = col.find_one({"NumeroBloque": "1"})
39
40 #Si "colvacia" es "None" el HashAnterior es "Genesis" y el NumeroBloque es 1
41 if(colvacia == None):
42     HashAnterior = "Genesis"
43     NumeroBloque = 1
44     NumeroBloque = str(NumeroBloque)
```

Cuando hay un bloque anterior creado se solicita los datos con el siguiente comando `find().sort('NumeroBloque', -1).limit(1)`, este código trae toda la información del bloque anterior el cual recorreremos con un for para ver cada variable de la colección y poder obtener el hash actual del anterior para ponerlo en el JSON actual como hash anterior, este hash anterior se vuelve hexadecimal antes de guardarse o de crear el hash actual que se guardará al final como una parte importante de la cadena de bloques, como se muestra en la siguiente figura.

```
55 #Si "colvacia" es DIFERENTE de "None" el HashAnterior es varia y el NumeroBloque es depende de cuantos se hayan creado
56 if (colvacia != None):
57     #Se guarda en la variable "bloquecodificado" el ultimo bloque que se inserto
58     bloquecodificado = col.find().sort('NumeroBloque', -1).limit(1)
59     #se recorren los items dentro de ese ultimo bloque
60     #cada item se guarda en la variable "bloqueanterior"
61     for item in bloquecodificado :
62         bloqueanterior = item
63     #Dentro de la lista que se guardo en "bloqueanterior" se selecciona el campo "NumeroBloque"
64     numbloquant = bloqueanterior['NumeroBloque']
65     #Dentro de la lista que se guardo en "bloqueanterior" se selecciona el campo "HashActual"
66     hashbloquant = bloqueanterior['HashActual']
67     numbloquant = int(numbloquant)
68     NumeroBloque = numbloquant + 1
69     HashAnterior = hashbloquant
70     #Se convierten las variables en string para poderla guardar en la base de datos
71     NumeroBloque = str(NumeroBloque)
72     HashAnterior = str(HashAnterior)
```

Posteriormente se usa la opción `insert_one` de PyMongo ingresándole todo el JSON con la información de la votación, en el software se guarda también la variable `NumeroBloque` que es 1 número más que el anterior, también se guardan los nombres de los candidatos con su respectiva votación, luego el hash anterior antes de crear el hash actual creado con la librería `hashlib`, todos estos datos se ingresan en MongoDB como se ve en la siguiente imagen.

```
51 col.insert_one({'NumeroBloque': NumeroBloque, 'nombre_1': nombre_1, 'votacion_1': votacion_1,  
52               'nombre_2': nombre_2, 'votacion_2': votacion_2, 'HashAnterior': HashAnterior, 'HashActual': HashActual  
53               })
```

Prueba del software

Se realizarán 3 pruebas a la encriptación de cada bloque de información, primero se guardará una información específica con el hash criptográfico de esta información, se tomará captura de pantalla y luego se volverá a ingresar la misma información, con 1 solo dígito cambiado para evaluar cómo sale el hash de estas 2 validaciones, indicando el cambio completo del hash que indica un cambio en los bloques creados, de esta manera protegiendo la integridad de los datos como se muestra en la siguiente imagen

```
NumeroBloque = 1  
nombre_1 = David  
votacion_1 = 7  
nombre_2 = Darwin  
votacion_2 = 9  
HashAnterior = Genesis  
HashActual = 48f68a43f99713c5918cc0135f509b97228a3c96ee04893d310173e6b8805f00
```

Luego de tener la evidencia del primer insert, procedemos a eliminar el bloque y volver a ingresarla información, en esta oportunidad la votación 2, no será 9 sino 6, como se muestra en la siguiente imagen.

```
NumeroBloque = 1  
nombre_1 = David  
votacion_1 = 7  
nombre_2 = Darwin  
votacion_2 = 6  
HashAnterior = Genesis  
HashActual = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4
```


La 2da prueba que se realiza es parecida a la anterior, pero en este caso se cambia una variable completa no solo un dígito sino la variable entera, primero haremos la inserción de los datos indicados y creamos el bloque, analizando el hash recién creado. Como se muestra en la siguiente imagen.

```
NumeroBloque = 1
nombre_1 = David
votacion_1 = 7
nombre_2 = Darwin
votacion_2 = 6
HashAnterior = Genesis
HashActual = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4

NumeroBloque = 2
nombre_1 = David
votacion_1 = 15
nombre_2 = Darwin
votacion_2 = 10
HashAnterior = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4
HashActual = ebf122d105e851afed7562baf93b7fe319daf8bdc694ce1c8c259a887e4a3b96

NumeroBloque = 3
nombre_1 = David
votacion_1 = 20
nombre_2 = Darwin
votacion_2 = 25
HashAnterior = ebf122d105e851afed7562baf93b7fe319daf8bdc694ce1c8c259a887e4a3b96
HashActual = 0fa8e1978a00919a58088c6f01901e3787c8b7a0d8c3a7dfccde61ccdbab69b9
```

Luego en esta segunda prueba se borra el registro anterior y se vuelve a crear datos para ingresar cambiando un dato completo, el anterior era el dígito 10 y ahora será el dígito 9 siendo un cambio total de variable, de esta forma validamos que el cambio en el hash actual fue completamente, indicando un error en los datos ingresados.

```
NumeroBloque = 1
nombre_1 = David
votacion_1 = 7
nombre_2 = Darwin
votacion_2 = 6
HashAnterior = Genesis
HashActual = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4
```

```
NumeroBloque = 2
nombre_1 = David
votacion_1 = 15
nombre_2 = Darwin
votacion_2 = 9
HashAnterior = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4
HashActual = 004628b59cde276df17957d0a8e0156057fe91ba4b1602307417dcc7e370a072
```

```
NumeroBloque = 3
nombre_1 = David
votacion_1 = 20
nombre_2 = Darwin
votacion_2 = 25
HashAnterior = 004628b59cde276df17957d0a8e0156057fe91ba4b1602307417dcc7e370a072
HashActual = 120fa2e02f496a9b1709972d7d4ec98e8fa3319d56e6184bbaff4a238d3dcf6b
```

La 3er prueba que se realizara al software es la modificación total de todos los datos con el fin de ver como se comporta el has con el cambio general de los datos, primero se ingresan datos específicos y validamos la creación del bloque y del hashactual, evidenciando en una imagen todos los hexadecimales del hash actual como se muestra en la siguiente imagen.

```
NumeroBloque = 1
nombre_1 = David
votacion_1 = 7
nombre_2 = Darwin
votacion_2 = 6
HashAnterior = Genesis
HashActual = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4
```

```
NumeroBloque = 2
nombre_1 = David
votacion_1 = 15
nombre_2 = Darwin
votacion_2 = 10
HashAnterior = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4
HashActual = ebf122d105e851afed7562baf93b7fe319daf8bdc694ce1c8c259a887e4a3b96
```

```
NumeroBloque = 3
nombre_1 = David
votacion_1 = 20
nombre_2 = Darwin
votacion_2 = 25
HashAnterior = ebf122d105e851afed7562baf93b7fe319daf8bdc694ce1c8c259a887e4a3b96
HashActual = e47fc2f31d6f2281a5d9684f7102c9c84d7c339a1b69de6cdba85b2b65895377
```

Posteriormente a tener la evidencia anterior, procedemos a eliminar el segundo bloque creado con los datos electorales para volver a ingresar datos con todas las variables cambiadas y podemos evidenciar la siguiente imagen que el bloque cambia todos los hashes posteriores al anterior. Mostrando así que el hash de cada bloque es único y dependiente de los datos que se le vayan a ingresar.

NumeroBloque = 1
nombre_1 = David
votacion_1 = 7
nombre_2 = Darwin
votacion_2 = 6
HashAnterior = Genesis
HashActual = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4

NumeroBloque = 2
nombre_1 = David Espinosa
votacion_1 = 14
nombre_2 = Darwin Rosas
votacion_2 = 9
HashAnterior = b9776d1022db452a8f41a78b51a3499274547c821a41e0624fb1c6d35f7f4ac4
HashActual = bf078a365356b2acedfa4fb0da07b661eb510929b7655ddcd6581af8b12ea553

NumeroBloque = 3
nombre_1 = David
votacion_1 = 20
nombre_2 = Darwin
votacion_2 = 25
HashAnterior = bf078a365356b2acedfa4fb0da07b661eb510929b7655ddcd6581af8b12ea553
HashActual = 0b2078e2e456f55ed98a14a9b65415fdb8137fc08bce28ed26f11a7e2773085c

ANALISIS ECONOMICO:

Análisis y cronograma

El siguiente análisis económico nos ayuda a determinar la viabilidad y la rentabilidad de este proyecto relacionado a la Ciberseguridad y al Blockchain. En este análisis se pueden evidenciar los costos desde el principio del proyecto hasta el mantenimiento dándonos una visión clara sobre los beneficios relacionados con el desarrollo del software y su seguimiento, se anexan estimación de los costos iniciales, los costos operativos anuales, los costos y beneficios a 5 años, y los beneficios previstos y el cálculo del retorno de la inversión (ROI). Se tuvo en cuenta las tarifas horarias de los diferentes roles involucrados, el total de horas a trabajar, el valor por rol y el ROI que nos indica la ganancia, en un análisis anual y a 5 años teniendo un retorno de rentabilidad importante.

El cronograma indicado por cada rol dentro del proyecto es el siguiente, a 3 meses de el desarrollo del software.

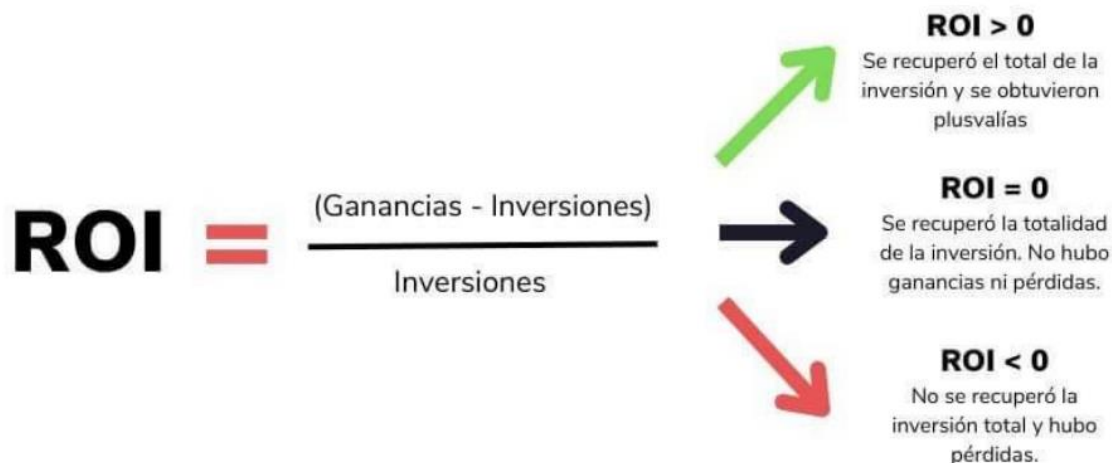
CRONOGRAMA	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
Arquitecto de sistemas												
Analista de requerimientos 1												
Analista de requerimientos 2												
Diseñador paginas web												
Desarrollador frontend 1												
Desarrollador frontend 2												
Desarrollador backend 1												
Desarrollador backend 2												
Tester de software web 1												
Tester de software web 2												
Ingeniero de ciberseguridad												

Todos los roles dentro del proyecto tendrán el mismo pago mensual, siendo 5 millones el valor estimado para cada cargo y 11 trabajadores que estarán disponibles para todas las etapas del proyecto, reuniéndolos con la metodología ágil SCRUM, en la etapa de creación del software se da un total invertido de 165 millones de pesos a 3 meses que dura el desarrollo del software, sin contar con el mantenimiento y seguimiento al software.

Rol en el proyecto	Tarifa hora	Tarifa dia	Tarifa Mes	3 Meses
Arquitecto de sistemas	20833	166666	5000000	15000000
Analista de requerimientos 1	20833	166666	5000000	15000000
Analista de requerimientos 2	20833	166666	5000000	15000000
Diseñador paginas web	20833	166666	5000000	15000000
Desarrollador frontend 1	20833	166666	5000000	15000000
Desarrollador frontend 2	20833	166666	5000000	15000000
Desarrollador backend 1	20833	166666	5000000	15000000
Desarrollador backend 2	20833	166666	5000000	15000000
Tester de software web 1	20833	166666	5000000	15000000
Tester de software web 2	20833	166666	5000000	15000000
Ingeniero de ciberseguridad	20833	166666	5000000	15000000
TOTAL	229163	1833326	55000000	165000000

Historia del retorno sobre la inversión (ROI)

El ROI retorno sobre inversión fue propuesto por primera vez por el señor Donaldson Brown un ingeniero eléctrico que se unió a una empresa y a principios de 1914 mientras trabajaba en DuPont como analista administrativo, propuso esta metodología para evaluar las ganancias probables de un proyecto, teniendo así la viabilidad de un proyecto, se ingresan los gastos estimados en un periodo de tiempo y luego se le resta a las ganancias obtenidas en ese tiempo, llamándolo retorno sobre la inversión, esta herramienta financiera se puede explicar de mejor manera en la siguiente imagen.



Metodología de análisis económico

Se usaran diferentes técnicas de evaluación financiera y de gestión de proyecto teniendo los datos de los costos iniciales y por un periodo de 5 años y también las ganancias anuales al evitar demandas y problemas jurídicos por fallos en el sistemas de almacenamiento de información de las votaciones, el mantenimiento y soporte del proyecto se divide en varias áreas teniendo todas las áreas la misma ganancia, esto sumado a la inversión inicial nos da una cantidad de dinero entendido como costo le el software, y por ultimo se calcula el ROI para ver el porcentaje de ganancias.

La siguiente imagen se muestran las horas laborales totales y el precio inicial del software.

Días	Horas	Horas mes	Trabajadores	Horas por mes todos los trabajadores	oras por 3 meses total trabajador	Total a pagar
30	8	240	11	2640	7920	164997360

Proyección de Costos Operativos Anuales

Luego de que el software se termine de desarrollar se consideraran los costos de mantenimiento que están explicados en la siguiente imagen con 5 roles del proyecto, de los cuales 2 de ellos necesitan horario diurno y nocturno para evitar errores no solucionados. Estos costos aseguran que el software permanezca actualizado, seguro y eficiente en su operación diaria.

Rol en el mantenimiento	Hora mensual	Horas anuales	Tarifa por hora	Tarifa anual
Mantenimiento y soporte	720	8640	20833	179997120
Actualizacion de software	240	2880	20833	59999040
Mejoras en seguridad	240	2880	20833	59999040
Auditorias de seguridad	240	2880	20833	59999040
Soporte tecnico	720	8640	20833	179997120
Total anual	2160	25920	104165	539991360

Cálculo del ROI

- Costos del software y su mantenimiento

Costo Inicial Total (sin IVA): \$165.000.000

Costo Operativo Anual: \$539.991.360

- Beneficios Anuales Estimados:

Reducción de costos de auditorías tradicionales: \$300,000,000

Disminución de fraudes y errores en el conteo de votos: \$100,000.000

Ahorro en tiempo y recursos de verificación de resultados: \$300,000.000

Beneficio anual estimado: $165.000.000 + 540.000.000 - (300.000.000 + 160.000.000 + 300.000.000)$

Beneficio anual estimado: 55.000.000

- Datos sobre 5 años

Inversión Total a 5 Años: $\$165.000.000 + (5 * \$540.000.000) = \$2.865.000.000$

Beneficios Totales a 5 Años: $760.000.000 * 5 = 3.800.000.000$

$3.800.000.000 - 2.865.000.000 = \$935.000.000$

- ROI

(Beneficios Totales – Inversión Total) Inversión Total ROI

$(3.800.000.000 - \$2.865.000.000) / \$2.865.000.000$

$935.000.000 / 2.865.000.000 = 0.32$

ROI = 32%

Inversion total 5 años	2.865.000.000
Beneficio total 5 años	3.800.000.000
ROI	32%

Concepto	Valor
Costo inicial	165.000.000
Costo operativo 5 años	2.865.000.000
Inversion total a 5 años	2.865.000.000
Beneficio anual estimado	935.000.000
Beneficio 5 años	3.800.000.000
Beneficio total 5 años	3.800.000.000
ROI en %	32,63525305

CONCLUSIONES

Se logro demostrar que el almacenamiento de la información con tecnología Blockchain se mejora la integridad de la información, la seguridad ante modificaciones externas y la descentralización de la responsabilidad en la veracidad de los datos. Blockchain es escalable y se puede usar en diferentes lenguajes de programación como C++, Solidity, Python y JavaScript

Este proyecto implemento el Blockchain con datos electorales usando encriptación SHA256, que en este momento es los mas seguros, se crearon vistas web para la iteración de ingreso y ver datos desde la pagina web, con esto se logra un software escalable y seguro frente a la modificación de los datos, se valido el software con diferentes pruebas en la seguridad de la creación de los bloques, se valida el hash anterior y el hash actual de cada bloque para percibir cambios entre ellos, asi se modifique un dato mínimo.

En el análisis económico del proyecto de almacenamiento de información electoral en Blockchain se calculo que al año de uso se presenta una ganancia de \$55.000.000 COP y a los 5 años del inicio del proyecto \$935.000.000, teniendo un ROI de 32%. Lo anterior demuestra la rentabilidad del proyecto y el fortalecimiento de la seguridad en procesos electorales, devolviendo la confianza en los sistemas de democracia.

REFERENCIAS BIBLIOGRAFICAS

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Nofer, M., Gomber, P., Hinz, O., & Schiereck, D. (2017). Blockchain. *Business & Information Systems Engineering*, 59(3), 183-187.

Swan, M. (2015). *Blockchain: Blueprint for a New Economy*. O'Reilly Media, Inc. Yli-Huumo, J., Ko, D., Choi, S., Park, S., & Smolander, K. (2016). Where is current research on Blockchain technology? —A systematic review. *PloS one*, 11(10), e0163477.

Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *Ethereum White Paper*. Dannen, C. (2017). *Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners*.

Apress. Hjalmarrsson, F., Hreiðarsson, G. K., Hamdaqa, M., & Hjalmtýsson, G. (2018). Blockchain-based e-voting system. In *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)* (pp. 983-986).

Katz, J., & Lindell, Y. (2007). *Introduction to Modern Cryptography*. CRC press. Glaser, F. (2017). Pervasive Decentralisation of Digital Infrastructures: A Framework for Blockchain-enabled System and Use Case Analysis. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.

Beck, R., Avital, M., Rossi, M., & Thatcher, J. B. (2017). Blockchain Technology in Business and Information Systems Research. *Business & Information Systems Engineering*, 59(6), 381- 384.

Liu, Yi; Wang Qi. (2017). *An E-Voting Protocol Based on Blockchain*. Southern University of Science and Technology, Shenzhen, China.

J. W. Lamare. (1981). "Eva Etzioni-Halevy. Political Manipulation and Administrative Power: A Comparative Study", *The ANNALS of the American Academy of Political and Social Science* 453.1

Business Wire. (1999). Arizona Democratic Party Selects Votation.com to Hold World's First Legally-Binding Public Election Over the Internet. ACE Project. (2024). Página web del proyecto ACE. Disponible en: <http://aceproject.org/>

United States Election Assistance Commission. (2005). *Voluntary Voting System Guidelines. Volume I*, version 1.0

Tribunal Superior Electoral. (2005). ¿O que faz a urna funcionar? Página web del tribunal superior de Brasil. Disponible en: <http://www.tse.jus.br/o-tse/escola-judiciariaeleitoral/publicacoes/revistas-da-eje/artigos/revista-eletronica-eje-n.-5-ano-5/digressoessobre-as-doacoes-de-campanha-oriundas-de-pessoas-juridicas>.

India Today. (2008). New counting method for Assembly polls. 2008. Disponible en: <https://www.indiatoday.in/elections-2008/story/new-counting-method-for-assemblypolls-34541-2008-12-04>.

e-Estonia. (2024). Página web de e-Estonia. Disponible en: <https://e-estonia.com/>

J. Barrati-Esteve, B. Goldsmith y J. Turner. (2012). "International experience with e-voting", Norwegian E-Vote Project. International Foundation for Electoral Systems. Document disponible

online la adresa <http://www.ifes.org/Content/Publications/News-inBrief/2012/June/%7E/media/B7FB434187E943C18F4D4992A4EF75DA>

Catalini, C., & Gans, J. S. (2016). Some Simple Economics of the Blockchain. NBER Working Paper No. 22952. NBER Horváth, A., & Zimányi, K. (2022). Exploratory Analysis of Blockchain Platforms in Supply Chain Management. *Economies*, 10(9), 206. DOI: 10.3390/economies10090206 PwC. (2020). Blockchain technologies could boost the global economy US\$1.76 trillion by 2030.

PwC VATrader. All about Return On Investment (ROI) - Formula, History, Pros and Cons! VATrader Consejo de Europa. (1981). Convenio para la Protección de las Personas con respecto al Tratamiento Automatizado de Datos de Carácter Personal (Convenio 108). <https://www.coe.int/en/web/conventions/full-list/-/conventions/treaty/108> Decreto 2241 de 1986, Código Electoral Colombiano. <https://www.suinjuriscol.gov.co/viewDocument.asp?ruta=Decretos/30018231>

Decreto 1377 de 2013, por el cual se reglamenta parcialmente la Ley 1581 de 2012. <https://www.suin-juriscol.gov.co/viewDocument.asp?id=1136422>

Directiva (UE) 2016/1148 del Parlamento Europeo y del Consejo, sobre medidas para garantizar un alto nivel común de seguridad de las redes y sistemas de información en la Unión. <https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A32016L1148>

Ley 12/2018, de 24 de mayo, de seguridad de las redes y sistemas de información, España. <https://www.boe.es/buscar/doc.php?id=BOE-A-2018-6612> Ley 1581 de 2012,

Ley de Protección de Datos Personales, Colombia. <https://www.suinjuriscol.gov.co/viewDocument.asp?ruta=Leyes/1606220> Ley 527 de 1999, Ley de Comercio Electrónico, Colombia. <https://www.suinjuriscol.gov.co/viewDocument.asp?id=1566549>

Ley Federal de la Innovación y Tecnología Blockchain, Suiza, 2020. <https://www.admin.ch/opc/es/classified-compilation/20201144/index.html>

Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales, España. <https://www.boe.es/buscar/doc.php?id=BOE-A2018-16673>

Ley Orgánica 5/1985, de 19 de junio, del Régimen Electoral General, España. <https://www.boe.es/buscar/doc.php?id=BOE-A-1985-11672>

Real Decreto 3/2010, de 8 de enero, por el que se regula el Esquema Nacional de Seguridad en el ámbito de la Administración Electrónica, España. <https://www.boe.es/buscar/doc.php?id=BOE-A-2010-1330>

Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos (Reglamento General de Protección de Datos). <https://eurlex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A32016R0679> UNCITRAL. (1996).

Ley Modelo de la UNCITRAL sobre Comercio Electrónico. BBC News Mundo. (2024). Venezuela: desafíos electorales y la lucha contra el fraude. Razón Pública. Recuperado de <https://razonpublica.com/venezuela-desafios-electorales-lalucha-fraude/>

DW. (2024). DW Verifica: ¿Qué tan grande es el riesgo de fraude electoral en Estados Unidos? Deutsche Welle. <https://www.dw.com/es/dw-verifica-qu%C3%A9-tan-grande-es-el-riesgo-de-fraude-electoral-en-estados-unidos/a-69598694>

France24. (2024). Oposición en Sudáfrica denunció fraude electoral y dijo que emprenderá acciones legales. France 24. <https://www.france24.com/es/video/20240602-oposici%C3%B3n-en-sud%C3%A1frica-denunci%C3%B3-fraude-electoral-y-dijo-queemprender%C3%A1-acciones-legales>

CNN Español. (2024). Difícil que se cometa alguna forma de fraude, dice Insulza previo a participar como observador electoral en México. CNN. Recuperado de <https://cnnespanol.cnn.com/video/difcil-que-se-cometa-alguna-forma-de-fraude-diceinsulza-previo-a-participar-como-observador-electoral-en-mexico/>

EFE Verifica. (2024). Los votos atribuidos a Junts fuera de Cataluña no son un fraude. EFE. <https://verifica.efe.com/junts-resultados-madrid-elecciones-europeas-error/>