Sistema de aseguramiento de archivos digitales

Juan Felipe Zora Zuluaga

Informe de práctica empresarial, como requisito para optar por el título de Ingeniero de Sistemas

Asesor

Deisy Alejandra Mazo Vélez, Magíster en Ciencias-Estadística

Universidad de Antioquia

Facultad de Ingeniería

Ingeniería de Sistemas

Medellín

2023

Cita	(Zora Zuluaga, 2023)
Referencia **Estilo APA 7 (2020)**	Zora Zuluaga, J. F. (2023). *Sistema de aseguramiento de archivos digitales* [Semestre de industria]. Universidad de Antioquia, Medellín.

Corrector de redacción y estilo: Deisy Alejandra Mazo Vélez
Proponente y financiador del proyecto: BCFort S.A.S
Asesor técnico: Edison Montoya

UNIVERSIDAD DE ANTIOQUIA
Vicerrectoría de Docencia

Sistema de Bibliotecas

Centro de Documentación Ingeniería (CENDOI)

Repositorio Institucional: http://bibliotecadigital.udea.edu.co

Universidad de Antioquia - www.udea.edu.co

Rector: John Jairo Arboleda Céspedes.

Decano/Director: Julio César Saldarriaga Molina.

Jefe departamento: Diego José Luis Botia Valderrama.

Dedicatoria

A mi familia y amigos, quienes han sido un pilar fundamental en mi crecimiento como persona y me han sostenido y apoyado incansablemente en el duro camino de la formación profesional. A todos los estudiantes de las universidades públicas de Colombia, quienes muchas veces carecen de recursos económicos pero no dejan que esto sea un impedimento para lograr sus metas.

Agradecimientos

En primer lugar, quiero agradecer a la Universidad de Antioquia, el Alma Mater que acoge cada año a miles de estudiantes provenientes de todos los rincones de Colombia y les ayuda a alcanzar sus sueños brindándoles un hogar en donde pueden crecer personal, profesional, social, espiritualmente y demás ámbitos de la vida. También agradezco al departamento de Ingeniería de Sistemas y a sus excelentes profesores, quienes siempre estuvieron prestos a atender mis inquietudes y opiniones como estudiante. Además, quiero dar las gracias a mi jefe Edison Montoya, a la empresa BCFort S.A.S y a sus asociados, quienes siempre han creído en mi talento y me han apoyado para seguir luchando por mis metas y aspiraciones académicas y laborales. Finalmente, agradezco a mi profesora Deisy Mazo, quien con su paciencia y servicialidad ha contribuido a que el proceso de prácticas académicas sea satisfactorio para todas las partes implicadas.

Tabla de contenido

Lista de figuras

Siglas, acrónimos y abreviaturas

API	Application Programming Interface
AWS	Amazon Web Services
IPFS	InterPlanetary File System
CQL	The Council on Quality and Leadership
DVD	Digital Versatile Disc
NSA	Agencia de Seguridad Nacional
CI/CD	Continuous Integration/Continuous Deployment
S3	Almacenamiento de datos seguro en la nube (S3)
EC2	Amazon Elastic Compute Cloud
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
SDK	Software Development Kit
UdeA	Universidad de Antioquia

Resumen

Para el desarrollo de este proyecto se construyó un programa de software que permite el almacenamiento de archivos en diferentes sistemas digitales centralizados y descentralizados, tales como S3, IPFS y Blockchain. Cabe destacar que almacenar un archivo completo en el *ledger* de una Blockchain es altamente costoso, por tanto, se opta por almacenar el resultado de aplicar la función hash con el algoritmo SHA-256 al contenido del archivo. Los objetivos planteados se logran por medio de una API que expone las rutas correspondientes a las operaciones de almacenar el archivo en cada uno de estos sistemas y un bot de Telegram que habilita la recepción de archivos por parte de un usuario final. Como resultado, se obtienen dos aplicaciones: la primera es un servidor que corre en una instancia EC2 de AWS y recibe mensajes HTTP de la segunda, la cual es un script que se encarga de correr el bot de Telegram y procesar los mensajes recibidos para ejecutar su debida operación.

Palabras clave: archivo, almacenamiento, hash, programa, bot, blockchain, S3, IPFS.

Abstract

For the development of this project, a software program that allows file storage on multiple digital systems such as S3, IPFS and Blockchain was built. It is important to remark that storing the whole file content is very expensive, so it has been chosen to store only the result of applying the SHA-256 hash operation to its content. The proposed goals are achieved by implementing an API that exposes the routes corresponding to the operations to store the file in each one of the systems and a Telegram bot which enables the file reception from the end-user. As result two applications are obtained: the first one is a server running on an EC2 machine and receiving HTTP messages from the second one, which is a script running the Telegram bot in charge of processing the messages and executing its corresponding operation.

Keywords: file, storage, hash, program, bot, blockchain, S3, IPFS.

Introducción

Los diferentes ataques de ciberseguridad que se han presentado en los últimos años y que tienen como objetivo información digital, han causado millones de dólares en pérdidas y han obligado a centrar la atención de las empresas y personas en la protección de sus recursos informáticos. En este informe, se presenta un estudio detallado de una solución de software que permite almacenar digitalmente distintos tipos de archivos en múltiples sistemas computacionales, con el propósito de que se complementen entre sí, para fortalecer las debilidades que podría tener cada uno y garantizar el cumplimiento de los pilares de la ciberseguridad, especialmente la integridad y la disponibilidad. Para alcanzar estos objetivos, se han escogido tecnologías de gran proyección y confiabilidad tales como Blockchain, IPFS y el servicio S3 de AWS. Como resultado, se obtiene una aplicación de software intuitiva y amigable con el usuario final, que garantiza la seguridad e inmutabilidad de sus archivos digitales, entre otros beneficios.

1 Objetivos

1.1 Objetivo general

Construir un programa de software que permita la recopilación y el almacenamiento de archivos y derivados de archivos en diferentes sistemas de información centralizados y descentralizados.

1.2 Objetivos específicos

- Producir una solución de software tipo bot en la red social Telegram que reciba archivos de los usuarios en el chat y transmita un enlace de descarga a otro programa de software tipo servidor.
- Crear un software tipo servidor capaz de recibir enlaces de descarga de archivos, bajarlos y guardarlos en el servicio de almacenamiento S3 de Amazon Web Services y una red descentralizada que siga el protocolo IPFS.
- Dotar al programa de software tipo servidor de la capacidad de crear un hash a partir del contenido de un archivo, usando el algoritmo SHA-256.
- Fabricar un software capaz de almacenar un hash en el *ledger* de una red tipo blockchain.
- Desplegar un nodo de una red IPFS que permita tener persistencia de los archivos almacenados y respuestas rápidas a los usuarios.

2 Marco teórico

El surgimiento de las computadoras y los sistemas de información han cambiado la manera en la que se perciben los datos, los cuales juegan un papel esencial en la sociedad actual. Los datos son importantes porque nos ayudan a mejorar la calidad de vida, determinar más fácilmente las causas de los problemas, tomar decisiones informadas, monitorear los sistemas, realizar análisis estratégico para conseguir los resultados que se desean, entre otros (The Council on Quality and Leadership [CQL], 2021).

Debido a la necesidad de almacenar los datos digitalmente en hardware como Unidades de Memoria Flash, DVD, Disco Rígido, Disco de Estado Sólido, etc; ha nacido toda una rama del conocimiento llamada ciberseguridad, dedicada a proteger sistemas críticos e información sensible de los ataques digitales (International Business Machine [IBM], s.f.). Con este objetivo se han planteado unos "pilares de la ciberseguridad" (DNV, s.f.), los cuales son:

1. **Confidencialidad:** indica que la información solo debe ser accesible para las entidades autorizadas.
2. **Integridad:** indica que la información no se puede modificar sin autorización.
3. **Disponibilidad:** indica que la información debe ser accesible para las entidades autorizadas en todo momento.

Esto conlleva al surgimiento de diferentes redes, protocolos y sistemas que permiten el almacenamiento digital de información y propician en mayor o menor medida cada una de las condiciones anteriormente mencionadas.

A continuación, se presenta un glosario de términos teóricos, necesarios para comprender el desarrollo del documento:

- **Archivo digital:** Es un contenedor de información representada en secuencias de bytes. Este viene asociado con un tipo de archivo o más comúnmente llamado, formato de archivo, el cual le permite diferenciar a los programas de software de qué manera deben interpretar la secuencia de bytes para mostrar correctamente la información que contiene el archivo (Oracle, s.f.). Físicamente hablando, los archivos son almacenados en unidades de almacenamiento tales como discos duros o discos de estado sólido.

- **Almacenamiento distribuido:** Se refiere a un sistema en el que los datos se almacenan en múltiples ubicaciones que están conectadas y coordinadas por una entidad centralizada (Hooda, 2023). En algunas implementaciones, esto se lleva a cabo dividiendo un archivo en varias cargas de bytes llamadas “chunks” y replicando un mismo chunk en varias computadoras a la vez.
- **Almacenamiento descentralizado:** Se refiere a un sistema en el que los datos se almacenan en múltiples ubicaciones (o nodos), pero cada ubicación es autónoma. Usualmente, este tipo de sistemas guarda una copia completa de la información en cada una de las diferentes locaciones (Hooda, 2023).
- **Almacenamiento en la nube:** Es un servicio ofrecido por empresas de terceros, las cuales prestan sus grandes capacidades de cómputo para almacenar datos y archivos que los clientes especifiquen, de manera que estos no gasten espacio de almacenamiento en sus propias computadoras. Amazon S3 es un servicio de almacenamiento en la nube provisto por la compañía Amazon Web Services (Amazon Web Services [AWS], s.f.).
- **Servidor:** Es un programa de software que se corre sobre una o varias computadoras interconectadas y que es capaz de recibir peticiones de un cliente y devolver una respuesta con base en su programación y su capacidad de cómputo. A este modelo de funcionamiento se le conoce como “arquitectura cliente-servidor”. Es importante mencionar que el cliente es también un programa de software que corre sobre una computadora siguiendo un protocolo (IONOS, 2023).
- **IPFS:** Es una red peer to peer para intercambiar información descentralizadamente. Los nodos en redes IPFS pueden almacenar en caché recursos que han descargado, para dejarlos disponibles indefinidamente para otros nodos (PROTOCOL LABS, s.f.).
- **Función Hash:** Es una función matemática que recibe un conjunto de entradas y las transforma en una salida, generalmente siendo tanto la entrada como la salida cadenas de caracteres (PROTOCOL LABS, s.f.). La función SHA-256 es un tipo de función hash diseñada por la Agencia de Seguridad Nacional (NSA), la cual cumple con unas propiedades, de las cuales destacan el “Determinismo”, es decir, que el mismo conjunto de entradas siempre devolverá la misma salida; y la “Resistencia a colisiones (CRHF)” que implica que encontrar dos conjuntos de entrada diferentes que devuelvan la misma salida es computacionalmente intratable (muy costoso en recursos y tiempo).

- **Telegram:** es una plataforma de mensajería enfocada en la comunicación en masa (TELEGRAM FZ-LLC, s.f.).
- **Bots de Telegram:** uno de los servicios que ofrece la plataforma de Telegram son los "bots"; los cuales son programas de software que cumplen con un set de reglas para que los usuarios de Telegram puedan interactuar con ellos por medio de la plataforma. Entre los casos de uso común de los bots de Telegram se encuentran la automatización de mensajes y respuestas, automatización de procesos como la realización de pedidos a un restaurante, hosteo de juegos, etc (TELEGRAM FZ-LLC, s.f.).
- **Blockchain:** es una red peer-to-peer de nodos interconectados que contienen una copia de un *ledger* único y sincronizado. Para cumplir el propósito de que el *ledger* sea único a través de la red, se implementan algunos conceptos importantes, como:
    - El conjunto de reglas que definen la manera correcta para realizar la transición de estados, es decir, modificaciones en la información contenida en el *ledger*. Se puede ver la transición de estados como la aplicación de unas operaciones controladas al estado actual del *ledger* haciendo que este cambie.
    - El método de consenso, define un protocolo para mantener una única copia sincronizada del *ledger* en todos los diferentes nodos de la red.
    - Los nodos, son capacidad computacional que ejecuta la implementación de la blockchain para poder hacer posible su funcionamiento y son remunerados monetariamente según los protocolos definidos.
    - *Smart contract*, es un script de código que se ejecuta cuando este recibe una transacción generando transiciones de estados, es decir, modificaciones en la información contenida en el *ledger*.
    - La cadena de bloques; dado lo mencionado en el primer ítem de esta lista, cada transición de estados nos deja un estado anterior (a las operaciones) y un estado posterior (a las operaciones). Cada una de estas versiones del estado recibe el nombre de "bloques" y son vinculadas a través de referenciación con otros bloques. La manera en la que se vinculan los bloques es que cada bloque tiene un identificador único, el cual se obtiene de pasar el contenido del mismo a través de una función hash; además, parte del contenido del bloque es a su vez el identificador único del bloque anterior. Esta estructura protege a la cadena de

bloques contra cambios en el sentido de que si un agente malicioso modifica el contenido de un bloque X, este cambiaría su identificador único (o el resultado de aplicar la función hash a su contenido ahora sería incorrecto) y generaría una discrepancia con el identificador único registrado en el siguiente bloque.

3 Metodología

Se procuró almacenar digitalmente un mismo archivo en diferentes sistemas. Los sistemas a usar fueron: el servicio de almacenamiento de objetos S3 provisto por AWS, la red principal de IPFS y una *testnet* de la blockchain de Algorand. Cabe destacar que para este último sistema se guardó sólo el resultado de aplicar al contenido del archivo la función hash SHA-256, que entrega como resultado una cadena de carácteres. Esta decisión viene motivada por dos razones principales: la primera es que el almacenamiento de un archivo completo puede llegar a ser particularmente costoso y la segunda es que almacenar la salida de la función hash permite aprovechar la propiedad de "determinismo" anteriormente mencionada garantizando que el archivo no pueda ser modificado sin dejar evidencia.

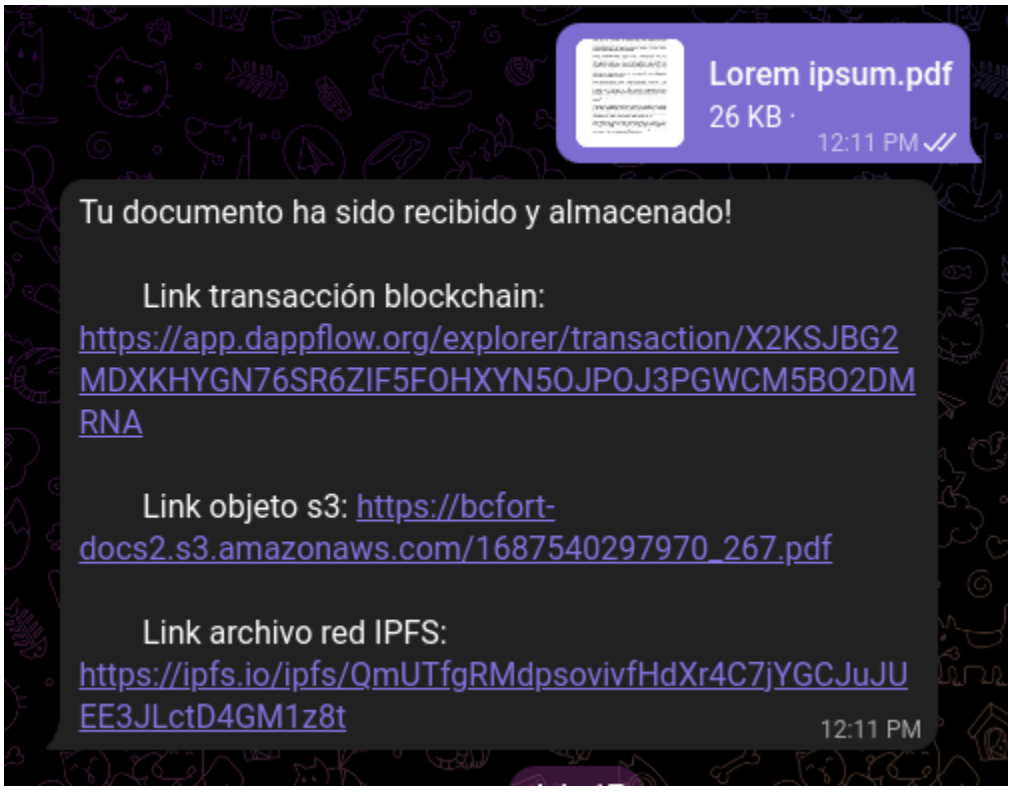
Para la recepción del archivo, se construyó un bot de Telegram que permite a cualquier usuario de esta red social enviar un archivo a través del chat (con el bot) y que este sea almacenado en uno, varios o todos los sistemas de almacenamiento mencionados anteriormente.

La gestión del proyecto se llevó a cabo siguiendo las técnicas definidas por la metodología Scrum. Se realizaron "sprints" mensuales con objetivos específicos apoyados por reuniones semanales con un asesor técnico para solventar dudas e implementar cambios. El desarrollo de los productos de software para cada sprint estará soportado por la plataforma GitLab, la cual permite control de versiones, seguimiento de bugs, revisión de código, CI/CD y más. Para el uso correcto del software de GitLab se seguirá una metodología interna definida por la empresa BCFort la cual define ciclos de desarrollo, testing e integración, auditorías, revisiones de calidad, etc. Cada producto de software que resulte de los sprints viene acompañado de su correspondiente documentación, abarcando funcionalidades, arquitectura, diseño, teoría, etc.

4 Resultados

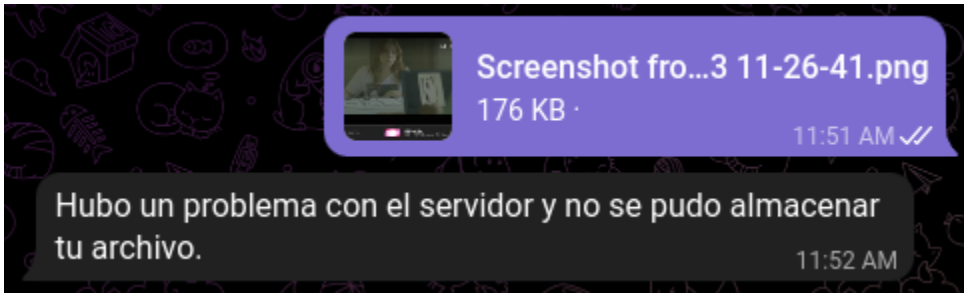
El resultado obtenido son dos soluciones de software independientes, que se conectan mediante una API para brindar una funcionalidad integral. La primera es un bot de Telegram pensado para la fácil interacción entre el usuario final y el programa, con el cual puede chatear cualquier usuario de esta plataforma, ya sea mediante la versión web o mediante la aplicación móvil. Su funcionamiento es simple: cuando el usuario escribe el comando "/start" en el chat recibe un mensaje de respuesta con instrucciones para usar la solución de almacenamiento de archivos; cuando el usuario escribe un mensaje aleatorio, el bot le pedirá que mande un archivo. Finalmente, cuando se envía un archivo, luego de unos segundos el bot responderá con los links correspondientes al archivo almacenado en los diferentes sistemas o con un mensaje de error de ser el caso (**Figura 1, Figura 2**). En la **Figura 3** se pueden observar desglosados los diferentes casos de uso del usuario en su interacción con el bot.

Figura 1 - *Respuesta bot de Telegram con links correspondientes al archivo almacenado*



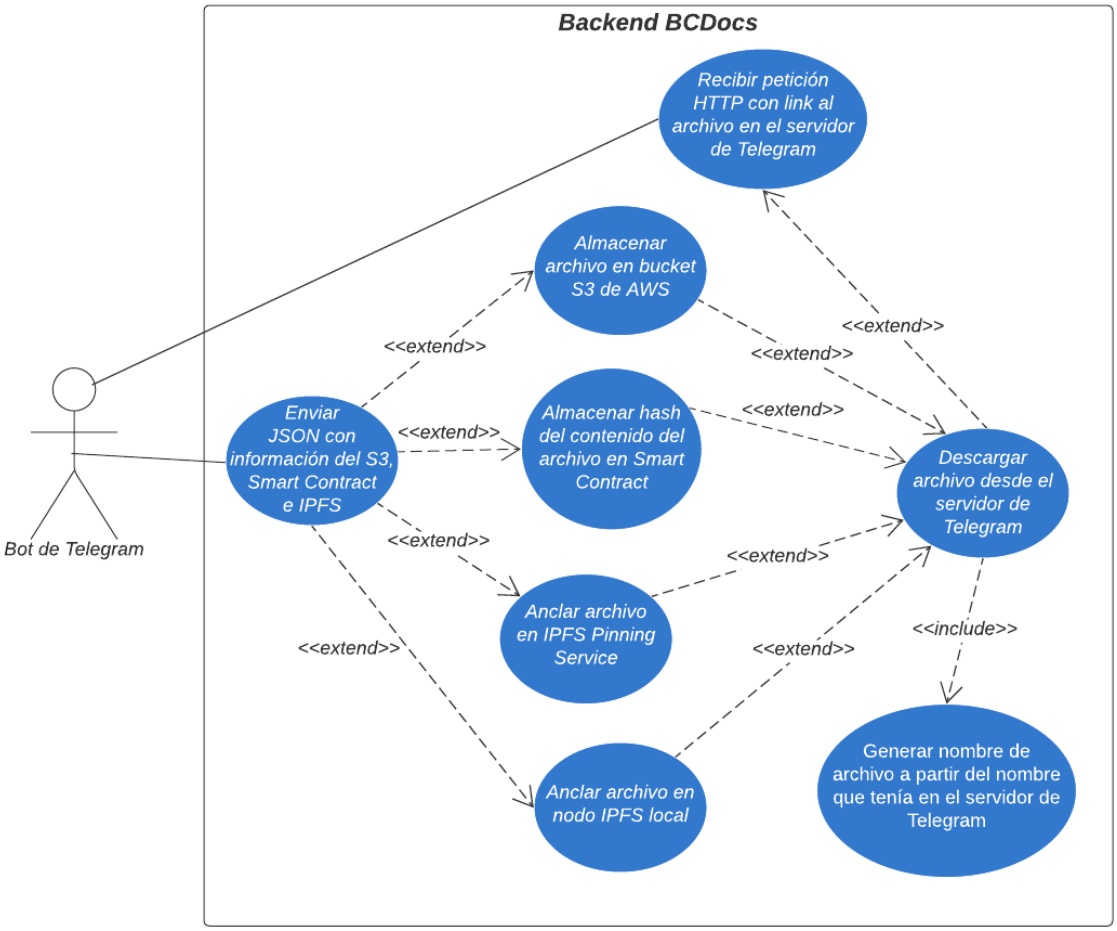
Nota. Fuente https://web.telegram.org

Figura 2 - *Respuesta bot de Telegram con mensaje de error*



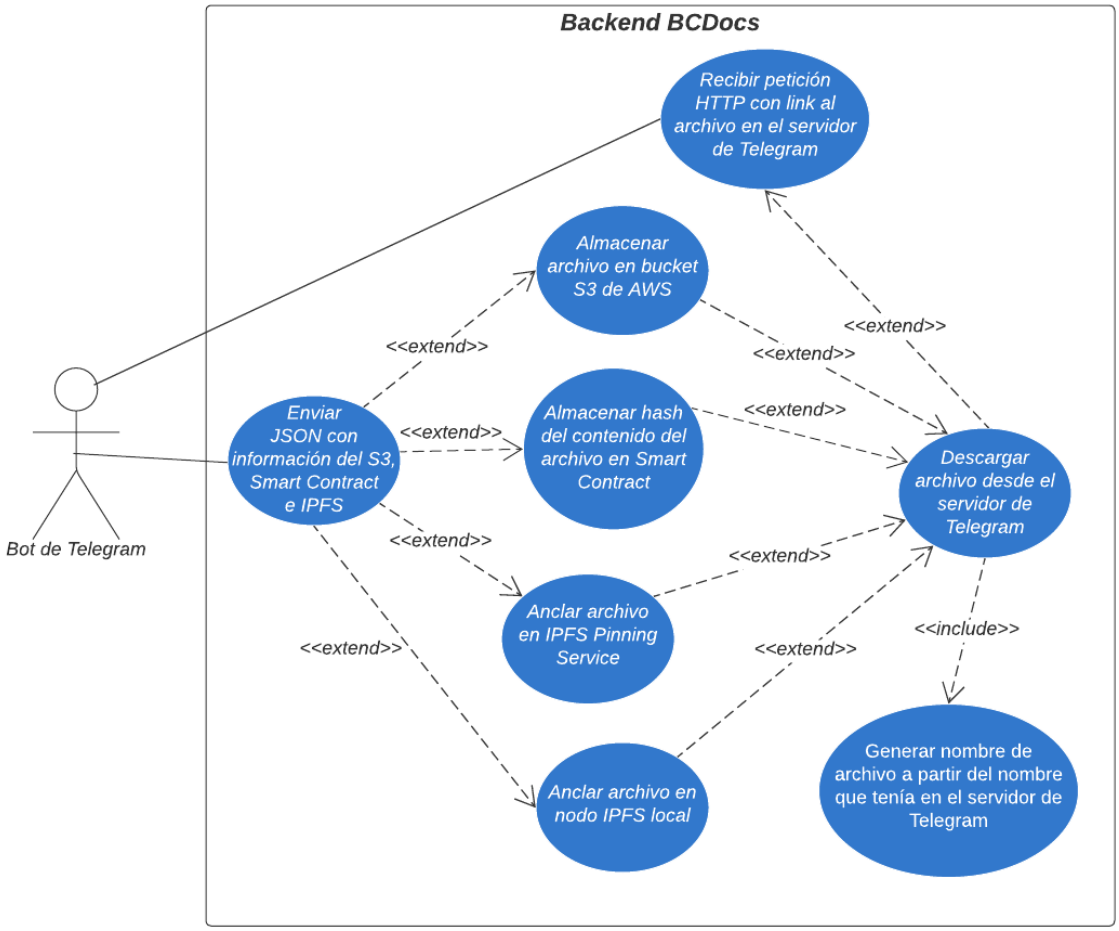
Nota. Fuente https://web.telegram.org

Figura 3 - *Diagrama de casos de uso del bot de Telegram*



El segundo componente, es un software tipo servidor que se encarga de exponer las funcionalidades de almacenamiento y dar respuesta al usuario final por medio del bot. La manera en la que se comunican el bot y el servidor se puede evidenciar en la **Figura 4**. Obsérvese que las interacciones se pueden resumir en dos: la primera es cuando el bot envía al servidor una petición HTTP tipo POST que contiene en la data un objeto JSON con el link de descarga al archivo que debe almacenar y la segunda, cuando el servidor responde a la primera petición con otro objeto JSON donde se condensan los links del archivo almacenado en diferentes sistemas.

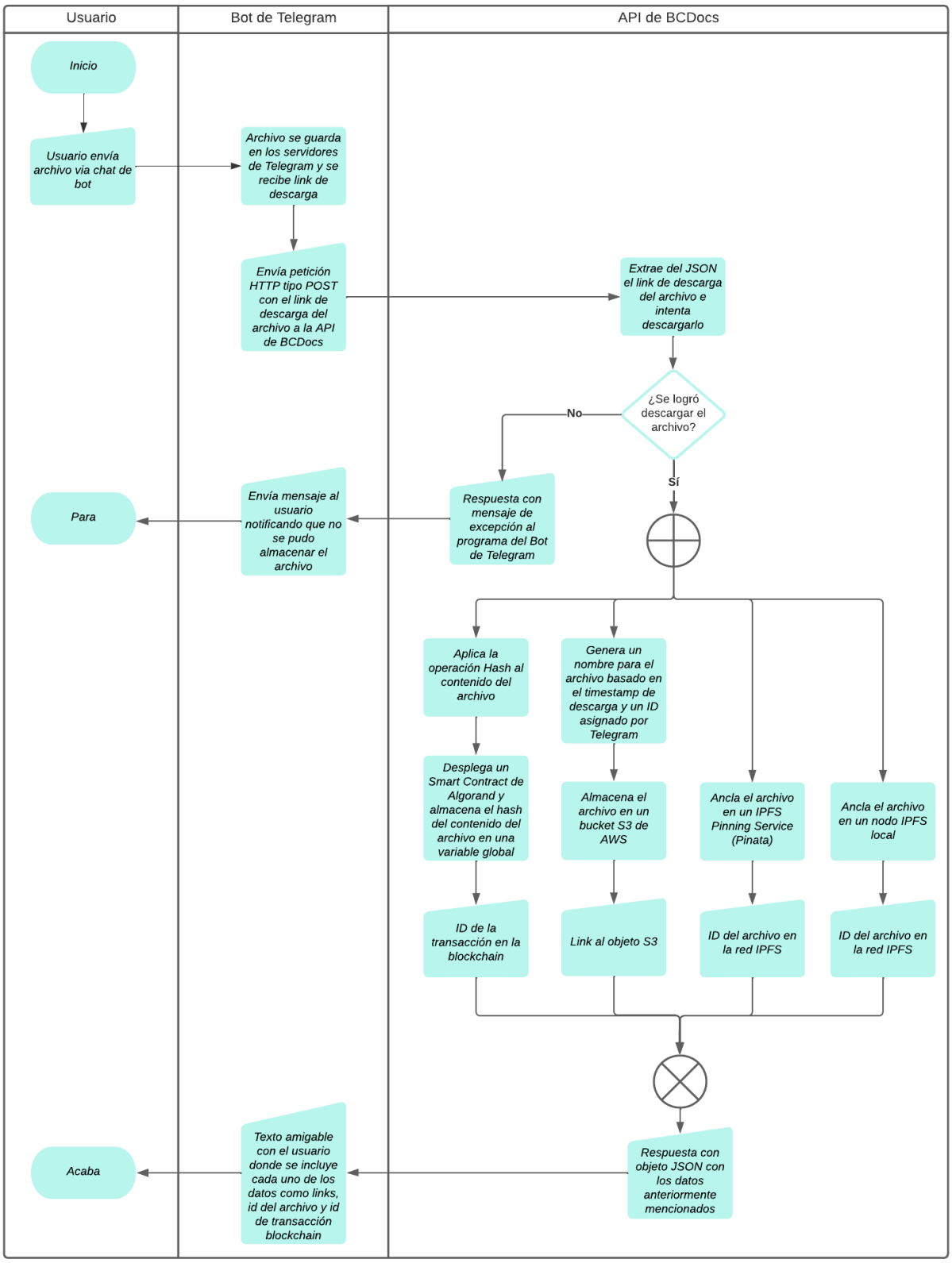
Figura 4 - *Diagrama de casos de uso del backend*



Adicionalmente, en las figuras (**Figura 4, Figura 5)** se detallan los procesos internos que realiza el software servidor para poder almacenar el archivo en los sistemas. Nótese que primero

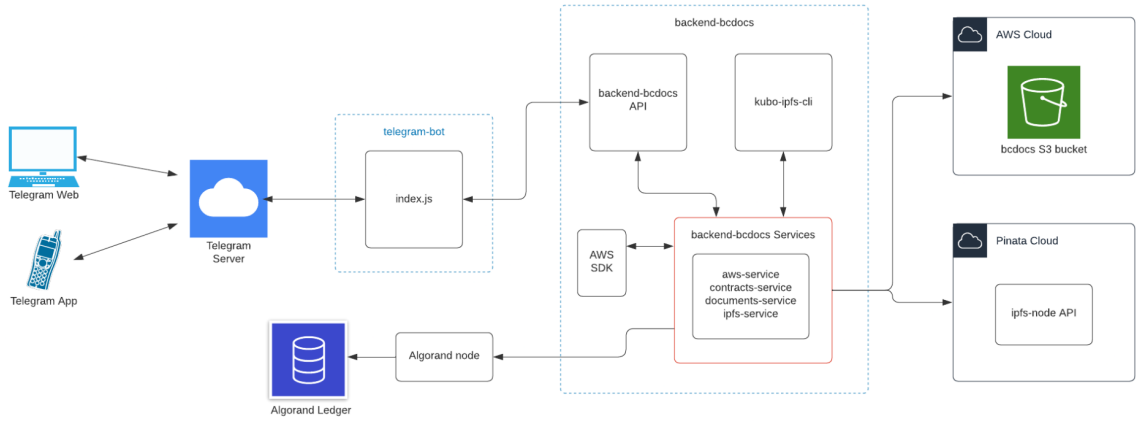
se debe descargar el archivo en el servidor para su posterior almacenamiento. Una vez descargado, se procede a almacenarlo en los sistemas IPFS. Luego, se asigna un nombre al archivo para poder guardarlo en el bucket S3 de AWS; este nombre se construye con base en un identificador asignado por Telegram y al *timestamp* correspondiente al momento de descarga del archivo por parte del servidor. Finalmente, se calcula el hash SHA-256 del contenido del archivo, se pone en una variable de estado global de un *Smart Contract* de Algorand y se despliega este último en la red. Una vez realizadas todas estas operaciones se condensa la información clave de referencia al archivo en los diferentes sistemas y se envía la respuesta al bot.

Figura 5 - *Diagrama de flujo BCDocs*



Finalmente, en la **Figura 6** se puede observar la arquitectura de software usada para el despliegue y funcionamiento del conjunto de programas que conforman la solución de BCDocs. Nótese que las funcionalidades que usa el bot son expuestas desde el backend por medio de servicios y que estos, a su vez, utilizan software como el CLI de kubo (el cual es la implementación de un nodo local de IPFS) y el SDK de Amazon para realizar sus tareas.

Figura 6 - *Diagrama de arquitectura de BCDocs*



5 Análisis

La realización de este análisis se hará con base en los objetivos específicos. El bot de usuario desempeña exitosamente su papel de recibir el archivo por parte de un usuario final; además, presenta una interfaz sumamente intuitiva y familiar para quien interactúa con este. En cuanto a la API y al servidor, dado que las funcionalidades a exponer eran relativamente simples y no ha sido necesario tener en consideración artefactos de programación extras como entidades o middlewares de avanzada complejidad, el framework empleado (Express.js) ha sido una elección ideal. Fue necesario el uso de algunos artefactos de software de terceros, como la conexión con la API de Pinata y la utilización del SDK de AWS, pero nuevamente la tarea a ejecutar era simple y estas herramientas incluían una clara y concisa documentación que evitó errores incomprendidos o complicaciones de código. Por el lado de blockchain, el esquema diseñado para almacenar el hash en el *ledger* puede llegar a ser notablemente costoso en términos monetarios si se llegara a tener un flujo alto de clientes, dado que cada uno de los documentos archivados requiere un nuevo Smart Contract desplegado en el *ledger* y esto puede llegar a ser un uso de espacio ineficiente en los nodos. Sin embargo, al tratarse de una prueba de concepto que interactúa con la *testnet* en lugar de la *mainnet*, toda interacción con el *ledger* es gratuita. Finalmente, en cuanto a lo que se refiere a levantar un nodo local de IPFS y a anclar los archivos recibidos a este, se instaló en la instancia EC2 la bien documentada implementación oficial hecha por el equipo de desarrollo de Protocol Labs llamada Kubo. Este programa viene con una CLI incorporada para ejecutar operaciones más fácilmente.

6 Conclusiones

El bot de Telegram ha sido una alternativa sorprendentemente simple y fácil de implementar para la interacción con el usuario, ya que solo se debían programar unas cuantas acciones como el estar atento a un mensaje y el reenviar un archivo recibido a una API; además, la interfaz gráfica ya estaba provista por la plataforma de Telegram a modo de un chat similar al de cualquier otra plataforma de mensajería con las cuales los usuarios ya están muy familiarizados.

Es importante destacar que para la ejecución de este proyecto no se ha considerado pertinente incorporar artefactos comunes a los desarrollos de software como lo son las bases de datos tradicionales y los sistemas de roles y credenciales. Este hecho ha permitido reducir la complejidad de implementación de la API y el servidor.

En la sección de Análisis fue mencionada la posibilidad de que el esquema usado para el almacenamiento en la blockchain de Algorand no fuera el más óptimo en cuanto a optimización de gas se trata. Un posible esquema alternativo que podría ahorrar gas sería tener un único Smart Contract con una única variable de estado que se modifique para almacenar un hash diferente cada que un nuevo archivo es enviado a la API. Nótese que esta solución, a pesar de ser más eficiente en términos de gas, podría presentar problemas de latencia (demora en las transacciones) ante un posible alto flujo de usuarios.

Referencias

AMAZON WEB SERVICES, INC. (s.f.). What is Amazon S3? AWS Docs. Consultado en Marzo 27, 2023, de https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html

AMAZON WEB SERVICES, INC. (s.f.). What is Cloud Storage? - Cloud Storage Explained. AWS Docs. Consultado el Marzo 31, 2023, de https://aws.amazon.com/what-is/cloud-storage/

AMBIT BST. (s.f.). ¿Conoces todos los sistemas de almacenamiento de datos? AMBIT - BST. Consultado el Marzo 27, 2023, de https://www.ambit-bst.com/blog/conoces-todos-los-sistemas-de-almacenamiento-de-datos

DNV. (s.f.). The three-pillar approach to cyber security: Data and information protection. Consultado el Marzo 27, 2023, de https://www.dnv.com/article/the-three-pillar-approach-to-cyber-security-data-and-information-protection-165683

Hayes, A. (2022, Septiembre 27). Blockchain Facts: What Is It, How It Works, and How It Can Be Used. Investopedia. Consultado el Marzo 31, 2023, de https://www.investopedia.com/terms/b/blockchain.asp

Hooda, P. (2023, Febrero 21). Comparison - Centralized, Decentralized and Distributed Systems. GeeksforGeeks. Consultado el Marzo 27, 2023, de https://www.geeksforgeeks.org/comparison-centralized-decentralized-and-distributed-systems/

INFURA INC. (2020, Agosto 17). An Introduction to IPFS. Infura Blog. Consultado el Marzo 27, 2023, de https://blog.infura.io/post/an-introduction-to-ipfs

INTERNATIONAL BUSINESS MACHINES CORPORATION. (s.f.). What is Cybersecurity? IBM. Consultado el Marzo 27, 2023, de https://www.ibm.com/topics/cybersecurity

IONOS. (2023, Marzo 1). What is a server? IONOS Digital Guide. Consultado el Marzo 27, 2023, de https://www.ionos.com/digitalguide/server/know-how/what-is-a-server-one-term-two-definitions/

ORACLE CORPORATION. (2023, Marzo 27). Conceptos básicos sobre el sistema de archivos (Solaris Common Desktop Environment: Guía del usuario). Oracle Docs. Consultado el Marzo 27, 2023, de https://docs.oracle.com/cd/E19683-01/816-3938/6ma6eh79q/index.html

PROTOCOL LABS. (s.f.). Hashing. IPFS Docs. Consultado el Marzo 27, 2023, de https://docs.ipfs.tech/concepts/hashing/

PROTOCOL LABS. (s.f.). What is IPFS? IPFS Docs. Consultado el Marzo 27, 2023, de https://docs.ipfs.tech/concepts/what-is-ipfs/

TELEGRAM FZ-LLC. (s.f.). Bots: An introduction for developers. Telegram APIs. Consultado el Marzo 27, 2023, de https://core.telegram.org/bots

TELEGRAM FZ-LLC. (s.f.). Telegram FAQ. Telegram. Consultado el Marzo 27, 2023, de https://telegram.org/faq

THE COUNCIL ON QUALITY AND LEADERSHIP. (s.f.). 12 Reasons Why Data Is Important. The Council on Quality and Leadership. Consultado el Marzo 27, 2023, de https://www.c-q-l.org/resources/guides/12-reasons-why-data-is-important/