Facultad de Ciencias

APLICACIÓN DE BLOCKCHAIN PARA LA PREVENCIÓN DE VIOLENCIA DE GÉNERO: UNA APROXIMACIÓN SOCIAL Y TECNOLÓGICA

(BLOCKCHAIN APPLICATION FOR GENDER-BASED VIOLENCE PREVENTION: A SOCIAL AND TECHNOLOGICAL APPROACH)

Trabajo de Fin de Máster
para acceder al

MÁSTER EN INGENIERÍA INFORMÁTICA

Autor: David Marcos Saiz

Director\es: Carlos Blanco Bueno

Septiembre - 2023

RESUMEN

La violencia de género es un problema social y una violación fundamental de los derechos humanos que afecta a personas de todas las edades, culturas y contextos socioeconómicos. Se manifiesta en diversas formas, como la violencia física, psicológica, sexual y económica, y tiene consecuencias devastadoras para las víctimas y la sociedad en su conjunto.

A pesar de los esfuerzos y avances en la lucha contra la violencia de género, sigue siendo un desafío persistente en todo el mundo. Las víctimas a menudo enfrentan barreras para buscar ayuda y protección, y es fundamental desarrollar nuevas estrategias y soluciones innovadoras para prevenir y abordar esta problemática.

En este proyecto se presenta una innovadora aplicación web que utiliza la tecnología Blockchain para abordar y prevenir la violencia de género de manera efectiva y significativa. La aplicación permite identificar aquellas zonas y momentos en los que se ha quebrantado la orden de alejamiento, facilitando la gestión de órdenes de alejamiento e incidencias, capacitando a las autoridades competentes una valiosa herramienta para el seguimiento y la verificación de eventos ocurridos como prueba del hecho a los usuarios realizar reportes geolocalizados en el mapa, lo que les brinda la oportunidad de identificar zonas conflictivas o peligrosas a escala nacional e incluso mundial

La utilidad de esta aplicación radica en su capacidad para generar conciencia y proporcionar datos valiosos para mejorar las políticas y programas de prevención de la violencia de género. Su implementación en la sociedad podría marcar un punto de inflexión en la lucha contra este grave problema social, alentando un enfoque colaborativo y tecnológico para un cambio positivo y duradero.

PALABRAS CLAVE:

Aplicación descentralizada, violencia de género, prevención, Blockchain, Ethereum, concienciación social, web3.

ABSTRACT

Gender-based violence is a social problem and a fundamental violation of human rights that affects individuals of all ages, cultures, and socioeconomic contexts. It manifests in various forms, such as physical, psychological, sexual, and economic violence, with devastating consequences for victims and society as a whole.

Despite efforts and advancements in combating gender-based violence, it remains a persistent challenge worldwide. Victims often face barriers in seeking help and protection, emphasizing the need to develop new strategies and innovative solutions to prevent and address this issue.

This project presents an innovative web application that leverages Blockchain technology to effectively and significantly address gender-based violence. The application enables users to submit geolocated reports on a map, empowering them to identify conflict-prone or dangerous areas both nationally and globally. Additionally, the platform offers an additional feature facilitating the management of restraining orders and incidents, providing relevant authorities with a valuable tool to monitor and verify events as immutable evidence.

The utility of this application lies in its ability to raise awareness, promote collective action, and provide valuable data to enhance gender-based violence prevention policies and programs. Implementation of this solution in society has the potential to be a turning point in the fight against this pressing social issue, fostering a collaborative and technology-driven approach for positive and lasting change.

KEY WORDS

Decentralized application, gender-based violence, prevention, Blockchain, Ethereum, social awareness, web3.

ÍNDICE

ÍNDICE DE ILUSTRACIONES

1. INTRODUCCIÓN

La violencia de género es una problemática social de gran magnitud que afecta a millones de personas en todo el mundo. Las víctimas, en su mayoría mujeres, enfrentan situaciones de riesgo y conflicto que requieren medidas de prevención y seguimiento efectivas para garantizar su seguridad y protección. Sin embargo, a pesar de los esfuerzos realizados en la lucha contra la violencia de género, existe una escasez de herramientas tecnológicas y digitales diseñadas específicamente para abordar este grave problema. La falta de soluciones tecnológicas adecuadas dificulta la prevención temprana de incidentes y la gestión adecuada de órdenes de alejamiento, lo que deja a las víctimas en situaciones de vulnerabilidad. En este contexto, la necesidad de desarrollar una herramienta innovadora y eficiente que utilice la tecnología para la prevención y seguimiento de la violencia de género se convierte en una prioridad en la búsqueda de una sociedad más segura y empoderada.

Las órdenes de alejamiento son una medida legal crucial para proteger a las víctimas de violencia doméstica y acoso, al establecer una distancia obligatoria entre la víctima y el presunto agresor. Sin embargo, su efectividad puede verse comprometida debido a los desafíos que surgen al tratar de determinar si alguna de las partes ha incumplido la orden. En casos donde existen versiones contradictorias sobre quién se acercó a quién, la falta de evidencia clara puede dificultar la aplicación adecuada de la orden de alejamiento.

Por lo tanto, la situación es más complicada cuando las víctimas y los agresores presentan diferentes relatos de los eventos, lo que dificulta saber con certeza quién violó la orden o si, de hecho, se produjo algún incumplimiento. Este tipo de conflictos pueden generar disputas legales prolongadas y angustiantes, afectando tanto a las víctimas que buscan protección como a los presuntos agresores que pueden enfrentar consecuencias legales injustas.

Se pretende proporcionar un sistema de prevención en la que se permita al usuario utilizar una función de seguimiento y reporte de posibles casos de incidentes en un mapa interactivo. Esta característica, facilitará la recopilación de información sobre posibles situaciones de violencia de género en ubicaciones específicas, lo que puede ayudar a identificar patrones y áreas de mayor riesgo para la prevención de nuevos incidentes

Unido a ello, la tecnología Blockchain [1] ha surgido como una poderosa herramienta con un potencial transformador en diversos sectores. La Blockchain, como una base de datos descentralizada y transparente, ofrece la oportunidad de crear sistemas seguros, confiables e

inmutables que pueden tener un impacto significativo en muchos dominios de aplicación, como la prevención y la protección de las víctimas de violencia de género.

Esta tecnología, al permitir la creación de aplicaciones descentralizadas (dApps) [2] y contratos inteligentes, puede proporcionar un marco seguro y transparente para desarrollar soluciones de prevención y rastreo de agresores. Mediante el uso de la Blockchain de Ethereum, es posible crear un ecosistema donde las víctimas tengan la capacidad de registrar y compartir de manera segura información relevante sobre incidentes y agresores, así como acceder a recursos y apoyo de manera más efectiva.

Además, una dApp es un tipo de aplicación que opera en una red descentralizada, como una cadena de bloques (Blockchain). A diferencia de las aplicaciones tradicionales, que se ejecutan en servidores centralizados controlados por una entidad o empresa, las dApps utilizan la tecnología de cadena de bloques para funcionar en una red distribuida y sin un punto único de control.

Entonces, se puede definir que un contrato inteligente [3] es un programa informático autónomo, ejecutado en una cadena de bloques como Ethereum, que se utiliza para automatizar, validar y ejecutar acuerdos y transacciones sin necesidad de intermediarios. Estos contratos son escritos en lenguajes de programación específicos, en el caso de este proyecto es Solidity, para la cadena de bloques y se almacenan en la Blockchain, lo que garantiza su inmutabilidad y seguridad en la red de Ethereum. Ethereum, una de las plataformas más populares para DApps y contratos inteligentes, introdujo este concepto en 2015. Es una plataforma descentralizada que permite a los desarrolladores crear y desplegar DApps y contratos inteligentes en su red.

Además, el ecosistema de Ethereum [4] se basa en su criptomoneda nativa, llamada Ether (ETH). Ether es la unidad de valor que se utiliza para realizar transacciones y operaciones dentro de la red Ethereum. Cuando los usuarios envían Ether de una dirección a otra, están efectuando transacciones en la cadena de bloques de Ethereum. Estas transacciones pueden implicar, por ejemplo, el intercambio de Ether por bienes y servicios o la transferencia de fondos entre diferentes cuentas.

Por consiguiente, para realizar cualquier acción o ejecutar un contrato inteligente en Ethereum, es necesario pagar una tarifa en Ether, conocida como "GAS" [4]. El GAS se utiliza para medir el costo computacional necesario para realizar una determinada operación en la red. Cada operación que se ejecuta en Ethereum, como una transacción o una llamada a un contrato inteligente, requiere una cantidad específica de GAS para completarse. Cuanto más compleja o

demandante sea una operación, mayor será la cantidad de GAS necesaria para realizarla. Esta medida tiene como objetivo incentivar a los participantes de la red a ser eficientes en el uso de recursos y evitar que se realicen operaciones excesivamente complejas o maliciosas que podrían ralentizar la red.

Por lo tanto, cuando un usuario envía una transacción o una llamada a un contrato inteligente, debe especificar la cantidad de GAS que está dispuesto a pagar por esa operación. Si la cantidad de GAS proporcionada es insuficiente para completar la operación, esta se detendrá y se reembolsará al usuario, pero la operación no se ejecutará por completo. Por otro lado, si el usuario proporciona suficiente GAS y la operación se realiza con éxito, el GAS utilizado se consume y se paga a los validadores y mineros que aseguran y mantienen la red Ethereum.

La motivación detrás de este trabajo de fin de máster radica en la necesidad urgente de ayudar a las víctimas de violencia de género y en el reconocimiento del potencial de la tecnología Blockchain para abordar esta problemática. Es fundamental desarrollar soluciones innovadoras y efectivas que no solo ofrezcan protección, sino que también empoderen a las víctimas y promuevan un cambio cultural hacia la igualdad y el respeto.

1.1. OBJETIVOS

Los objetivos de este trabajo de fin de máster son claros y se enfocan en abordar la violencia de género desde una perspectiva tecnológica. Se busca desarrollar una dApp descentralizada en la Blockchain de Ethereum que permita rastrear a los agresores y proporcionar a las víctimas una herramienta para protegerse y documentar incidentes de manera segura y confiable.

Además, se busca explorar y analizar en profundidad los principios de la tecnología Blockchain en relación con la violencia de género, evaluando su potencial para mejorar la prevención, la seguridad y la visibilidad de estos casos. Esto implica investigar la aplicabilidad de la tecnología Blockchain en el ámbito social y comprender cómo puede integrarse de manera efectiva en las estrategias existentes de prevención y apoyo a las víctimas.

1. Investigar y aplicar los principios de la tecnología Blockchain en el contexto de la violencia de género: Se realizará una investigación sobre cómo la tecnología Blockchain puede contribuir a la prevención y abordaje de la violencia de género. Se explorarán los aspectos técnicos y conceptuales de la tecnología Blockchain, así como su aplicabilidad en el ámbito social y particularmente en el contexto de la violencia de género. Esto incluirá la comprensión de conceptos como contratos inteligentes, descentralización, transparencia y seguridad en el contexto de la violencia de género.

2. Desarrollar una dApp una dApp descentralizada basada en la tecnología Blockchain de Ethereum para prevenir la violencia de género. La aplicación permitirá a las víctimas de violencia documentar y protegerse de manera segura mediante el rastreo de incidencias y el uso de órdenes de alejamiento registradas en la Blockchain. Se recolectarán y almacenarán de forma segura coordenadas geográficas relevantes para generar evidencia confiable que pueda ser utilizada en casos legales o para buscar apoyo y recursos. Además, habrá un sistema para reportar supuestos incidentes en un mapa, siendo visible para cualquier usuario con acceso a la dApp.
3. Contribuir conocimiento y la conciencia sobre el potencial de la tecnología Blockchain para resolver problemas sociales, especialmente en el ámbito de la violencia de género. Se busca compartir hallazgos y lecciones aprendidas a través de publicaciones y presentaciones con el fin de fomentar el diálogo y la colaboración entre profesionales tecnológicos y expertos en violencia de género. La meta es promover el uso responsable y ético de la tecnología para abordar este importante problema social.

Con este trabajo, se pretende dar un paso hacia adelante en la protección de las víctimas de violencia de género, aprovechando las capacidades de la tecnología Blockchain para brindar soluciones innovadoras y promover un cambio positivo en nuestra sociedad.

La integración de esta tecnología en la prevención de la violencia de género no solo busca mejorar la seguridad y protección de las víctimas, sino también concienciar y promover un cambio de actitud hacia este problema, fomentando una sociedad más justa e igualitaria.

2. MATERIAL Y MÉTODOS UTILIZADOS

2.1. TECNOLOGÍAS Y HERRAMIENTAS

Al desarrollar este TFM, se han seleccionado cuidadosamente una serie de tecnologías que considero ideales para lograr los objetivos del proyecto y abordar los desafíos específicos que implica la prevención de la violencia de género. Se explicarán el porqué de la elección de estas tecnologías en lugar de otras similares:

- **Blockchain**: La tecnología Blockchain, en particular la Blockchain de Ethereum, se ha establecido como una solución confiable y transparente para el almacenamiento seguro de datos descentralizados, brindando un enfoque innovador en determinados asuntos sociales.

- **Visual Studio Code**: es un editor de código ampliamente utilizado, ofrece una interfaz intuitiva y herramientas adicionales para mejorar la productividad en el desarrollo de aplicaciones descentralizadas, entre otras.
- **NPM (Node Package Manager):** para gestionar paquetes de Node.js [5], proporciona una solución eficiente para administrar dependencias y módulos en el desarrollo de aplicaciones.
- **Node.js**: un entorno [5] de ejecución de JavaScript permite desarrollar aplicaciones de servidor escalables y de alto rendimiento, siendo una opción ideal para interactuar con la Blockchain de Ethereum.
- **Vue.js**: un framework [6] de JavaScript progresivo, ofrece una forma elegante y sencilla de construir interfaces de usuario interactivas y receptivas.
- **Vue router**: es una librería [7] oficial de Vue.js que facilita la gestión del enrutamiento en aplicaciones de una sola página. Permite definir rutas de manera declarativa, asociando cada ruta con un componente específico de Vue.js.
- **Ganache**: es una herramienta [8] de desarrollo que brinda una red de prueba local para el desarrollo y la prueba de aplicaciones descentralizadas, permitiendo un entorno controlado y seguro en la Blockchain de Ethereum.
- **Solidity**: un lenguaje de programación [9] específico de Ethereum ofrece un enfoque poderoso para escribir contratos inteligentes, brindando la flexibilidad necesaria para implementar la lógica compleja requerida.
- **Truffle**: es un conjunto de herramientas [10] diseñado para desarrollar aplicaciones dentro del ecosistema de Ethereum, como entornos de trabajo, tests, etc.
- **Metamask**: es una billetera digital [11] y extensión para navegadores que permite a los usuarios interactuar con aplicaciones descentralizadas en la cadena de bloques Ethereum y administrar sus criptomonedas de forma segura. Facilita la firma de transacciones y la ejecución de contratos inteligentes directamente desde el navegador. Es una herramienta que permite interconectar de forma sencilla las dApps y un frontal web.
- **Hardhat.js**: es un entorno de desarrollo [12] de código abierto especialmente diseñado para la creación y despliegue de aplicaciones descentralizadas (dApps) en la red Ethereum. Ofrece una infraestructura robusta para compilar, probar y desplegar contratos inteligentes de forma eficiente y segura.

- **Leaflet**: es una biblioteca [13] de código abierto y altamente popular para la creación de mapas interactivos en aplicaciones web. Su enfoque se centra en ser ligero, eficiente y fácil de usar.
- **Web3.js**: es una biblioteca [14] de JavaScript que proporciona una interfaz de programación de aplicaciones (API) para interactuar con la Blockchain de Ethereum, permitiendo el envío de transacciones y la lectura de datos de contratos inteligentes.
- **GitHub**: es una plataforma [15] líder para el control de versiones del código fuente que facilita la colaboración entre desarrolladores, el seguimiento de cambios y la revisión del código en el desarrollo.
- **ESlint**: es una herramienta [16] de linting de código abierto para JavaScript. Su enfoque principal es detectar y corregir errores en el código, asegurando así una alta calidad de este. Mediante reglas configurables, ESLint identifica problemas de sintaxis, errores comunes y también promueve prácticas de codificación consistentes y de estilo. Su integración con flujos de trabajo de desarrollo y sistemas de control de versiones lo hace una herramienta valiosa para mejorar la calidad y la mantenibilidad del código en proyectos de cualquier tamaño, siendo especialmente útil en el contexto de trabajos de fin de máster (TFM) donde la calidad del código es esencial.
- **TypeScript**: es un lenguaje [17] de programación de código abierto desarrollado por Microsoft. Es una extensión de JavaScript que agrega tipos estáticos opcionales al lenguaje.

2.2. METODOLOGÍA

Se ha seguido una metodología iterativa e incremental [18] para llevar a cabo el proyecto. Este enfoque implica dividir el proyecto en diferentes etapas o ciclos. En cada etapa, el producto se va actualizando y desarrollando gradualmente hasta alcanzar el producto final que cumpla con los requisitos y objetivos establecidos. Esta elección metodológica se basa en la ventaja de desarrollar las funcionalidades una a una, permitiendo pulir cada aspecto del producto en la iteración correspondiente y utilizar esas implementaciones como base para el desarrollo de las etapas siguientes.

ILUSTRACIÓN 1 - METODOLOGÍA

En este proyecto cada iteración se ha dividido en 4 partes:

- Requisitos.
- Diseño.
- Implementación.
- Pruebas.

2.3. PLANIFICACIÓN DEL TRABAJO

La planificación del desarrollo de la dApp se estructura en distintas etapas que abarcan desde la fase de investigación inicial hasta la elaboración de la documentación y presentación final. A continuación, se ofrece una visión general de las principales etapas y las respectivas tareas asociadas, junto con su estimación de duración en semanas:

1. **Investigación de tecnologías y herramientas (2,5 semanas)** Esta fase se centrará en una investigación detallada de las tecnologías y herramientas esenciales, véase el apartado 2.1, para la construcción de la dApp.
2. **Análisis del problema (1 semana)** Se identificarán las principales limitaciones y obstáculos existentes en el proceso actual y se buscarán soluciones efectivas para abordarlos. Esta fase de análisis será esencial para asegurar que el desarrollo de la dApp se realice con una visión completa y bien fundamentada del problema que se pretende abordar, y así proporcionar una solución informática efectiva para prevenir y enfrentar la violencia de género de manera proactiva y oportuna.
3. **Diseño de la arquitectura (1,5 semanas):** En esta fase, se diseñará la arquitectura de la dApp siguiendo el patrón Modelo-Vista-Controlador (MVC). Los Smart Contracts escritos en Solidity actuarán como modelo en el Backend, gestionando los reportes y órdenes de alejamiento en la Blockchain de Ethereum de forma segura. El Frontend desarrollado con Vue.js proporcionará una interfaz para que las víctimas emitan reportes y los funcionarios

gestionen las órdenes sin necesidad de una API adicional, ya que los Smart Contracts proveerán directamente la funcionalidad requerida para la interacción con la Blockchain como modelo.

4. **Desarrollo frontend (2 semanas):** Durante esta etapa, se procederá al desarrollo del frontend de la dApp mediante el uso de Vue.js. Se implementará una página específica para que las víctimas emitan reportes de estado, la cual incluirá la integración de mapas Leaflet para visualizar los reportes geográficamente. A su vez, los funcionarios con acceso a esta aplicación podrán crear órdenes de alejamiento y ver las incidencias asociadas a esta. Mediante una representación de la víctima y el agresor en el mapa y de la interacción posible entre ellos.
5. **Desarrollo del backend y smart contracts (4 semanas):** Simultáneamente al desarrollo del frontend, se llevará a cabo la implementación de los smart contracts en Solidity, los cuales tendrán la finalidad de gestionar los reportes y crear órdenes de alejamiento e incidencias. Se realizarán pruebas para garantizar su correcto funcionamiento. También se abarca la implementación del backend utilizando Ganache
6. **Integración (1 Semana):** se establecerá la comunicación efectiva entre Frontend, Backend y Smart Contracts, permitiendo a los usuarios interactuar con la Blockchain de forma segura. Se realizará compilación y despliegue de estos contratos en la Blockchain y se realizarán las llamadas a sus funciones utilizando TypeScript con la librería de “ethers” [20] para este fin.
7. **Programa de carga (1 semana):** Esta etapa abarca el desarrollo de un programa para alimentar la Blockchain de datos de calidad. Su objetivo es realizar una simulación de la obtención de posiciones de víctima y agresor.
8. **Pruebas y ajustes:** En esta etapa se llevarán a cabo pruebas de la aplicación con el propósito de identificar y corregir posibles errores. Estas pruebas se realizarán a lo largo de todo el proyecto con el fin de cumplir el desarrollo iterativo e incremental
9. **Documentación y presentación (2 semanas):** Finalmente, se procederá a la elaboración de la memoria del TFM, la cual detallará minuciosamente el desarrollo del proyecto. Se documentarán los smart contracts y otros componentes relevantes. Además, se preparará una presentación adecuada para el tribunal.

El tiempo total estimado para llevar a cabo el desarrollo completo de la dApp será de 12 semanas, tomando en cuenta todas las etapas mencionadas anteriormente. A continuación, se presenta un diagrama Gantt de la planificación descrita.

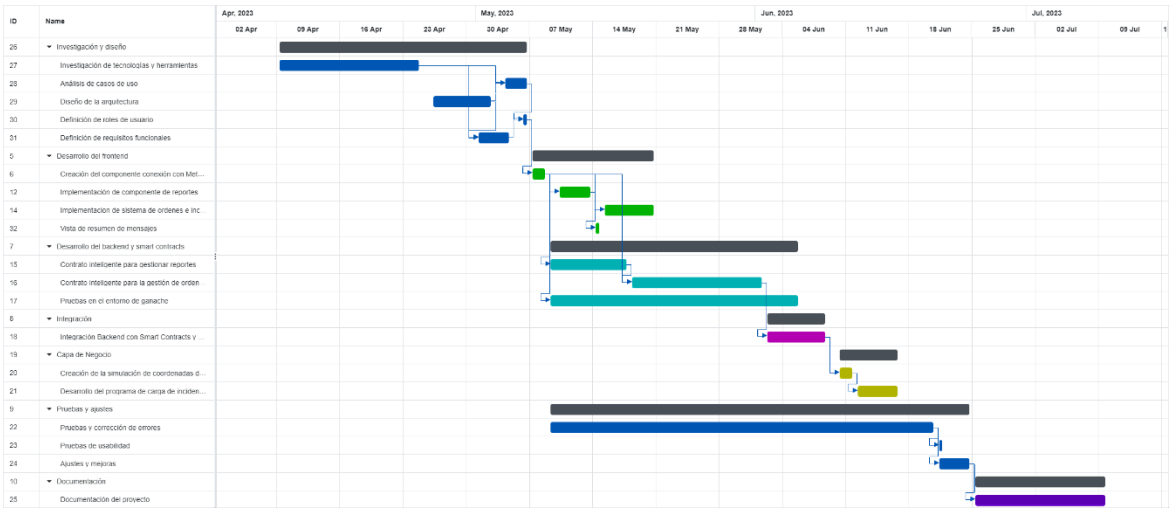

ILUSTRACIÓN 2 - DIAGRAMA GANTT

3. ANÁLISIS DEL PROBLEMA

Este capítulo presenta los requisitos de la aplicación a desarrollar, la cual es una dApp que utiliza la tecnología Blockchain para gestionar reportes de estado y órdenes de alejamiento, con el objetivo de prevenir y hacer seguimiento de situaciones de riesgo o conflictos. La dApp se basa en Ethereum y utiliza la billetera digital MetaMask para asegurar la autenticidad de las transacciones y garantizar el tratamiento anónimo de los datos de los usuarios. Se ha realizado una división de los permisos que estos usuarios van a tener, por una parte, está el usuario genérico, que solo tendrá acceso a cierta funcionalidad de la aplicación como los reportes de estado y, por otro lado, los funcionarios, que podrán ver e interactuar con todo. Este enfoque seguro y descentralizado permite a los usuarios generar reportes de manera anónima y a los funcionarios gestionar estas situaciones a través de una interfaz interactiva. El documento describe detalladamente las funcionalidades clave de la dApp y los requisitos considerados durante su desarrollo, enfatizando en la protección de datos y la prevención de incidencias para contribuir a una sociedad más segura y consciente en un mundo interconectado y digital.

Las funcionalidades clave y los requisitos funcionales de cada una de ellas se describen a continuación.

3.1. REQUISITOS FUNCIONALES

1. Conexión con la plataforma Blockchain.
   [1] La dApp permite a los usuarios registrarse mediante su billetera digital MetaMask.
   [2] El acceso estará restringido por roles y un listado determinado, comprobando así que billetera puede conectarse y con qué permisos.
   [3] Habrá dos roles existentes, el de usuario genérico y el de funcionario.

[4] El rol de usuario genérico solo podrá ver e interactuar con los reportes de estado.

[5] El usuario con rol de funcionario podrá visualizarlo todo.

2. Generación de reportes de estado.

[6] Los usuarios podrán generar reportes de estado que incluyen coordenadas geográficas y mensajes descriptivos sobre la situación reportada.

[7] Estos reportes serán almacenados en la cadena de bloques de Ethereum mediante una transacción que debe ser confirmada manualmente a través de MetaMask, garantizando la inmutabilidad de la información.

[8] Se pedirá al usuario confirmación de permisos para acceder a la ubicación de su dispositivo para realizar un reporte más rápido, extrayendo las coordenadas de donde se encuentre.

[9] Se podrá pulsar directamente sobre el mapa para obtener las coordenadas de forma automática.

[10] Se podrá navegar por el mapa para ver las zonas donde hay más reportes de estado. El mapa es a nivel mundial.

[11] Se podrá consultar cada reporte en la sección de resumen, en la parte inferior de la página, donde se ve uno a uno los estados reportados por los diferentes usuarios de la dApp.

[12] En el resumen de los reportes, se mostrarán las coordenadas y descripción del reporte, además de la fecha y hora.

3. Gestión de órdenes de alejamiento.

[13] Los usuarios con roles de funcionario tendrán acceso a una sección especializada para la gestión de órdenes de alejamiento.

[14] Los funcionarios podrán consultar información de la víctima y el agresor, además de ver las incidencias para esa orden de alejamiento, si las hubiera.

[15] Los funcionarios podrán crear nuevas órdenes de alejamiento, ingresando información detallada sobre las víctimas y los agresores involucrados en la situación,

4. Simulación de incidencias.

[16] La dApp permitirá la carga que muestra la trayectoria de pasos entre víctima y agresor relacionadas con las órdenes de alejamiento almacenadas en la cadena de bloques.

[17] Este programa permitirá visualizar movimientos y acciones de víctimas y agresores en el mapa, lo que facilita el seguimiento y comprensión de las situaciones simuladas.

[18] Si para la orden dada existen más de una incidencia, esto se verá reflejado en un selector de fecha en la parte superior del mapa, fuera de este.

[19] Se permitirá seleccionar la fecha deseada de las incidencias para verla tantas veces como se quiera.

[20] Una vez comenzada la reproducción, no se podrá seleccionar otra fecha.

[21] El seguimiento de incidencias se verá representado en intervalos de un segundo para poder realizar un seguimiento punto a punto de lo ocurrido.

5. Visualización de datos.

[22] El rol de funcionario podrá visualizar todo el contenido de la aplicación.

[23] El rol de usuario solo podrá ver la parte de reportes de estados, donde podrá visualizar tanto sus reportes como los de otros usuarios.

[24]Los reportes de estado y las órdenes de alejamiento se presentarán de manera interactiva y visual en un mapa.

[25] La dApp ofrece una representación gráfica de los datos, mostrando marcadores en el mapa para ubicar los reportes e incidencias en las órdenes de alejamiento.

[26] En las incidencias se distinguirá con un icono rojo al agresor y con un icono verde a la víctima.

6. Búsquedas y filtrado.

[27] La dApp contará con una función de búsqueda y filtrado basada en el DNI de las partes involucradas en las órdenes de alejamiento.

3.2. REQUISITOS NO FUNCIONALES

1. Seguridad y Privacidad:

[1] Garantizar la confidencialidad de los datos almacenados en la cadena de bloques mediante el uso de la tecnología Blockchain.

[2] Implementar contratos inteligentes para asegurar el acceso controlado a la información relevante y prevenir vulnerabilidades de seguridad.

[3] Anonimizar los reportes de estado utilizando el ID de la billetera MetaMask para proteger la privacidad de los usuarios.

2. Escalabilidad y rendimiento

[4] Diseñar la dApp con una arquitectura que permita manejar un aumento significativo de usuarios y transacciones sin afectar el rendimiento y la velocidad de respuesta.

[5] Optimizar la eficiencia en el almacenamiento y acceso a los datos para garantizar una experiencia rápida y sin demoras a través de los contratos inteligentes.

3. Usabilidad y experiencia de usuario

[6] Proporcionar una experiencia sin complicaciones para los usuarios, desde la conexión con MetaMask hasta la generación de reportes y visualización de órdenes de alejamiento.

4. Interoperabilidad

[7] Garantizar la compatibilidad de la dApp con diferentes navegadores web para ofrecer una experiencia fluida y accesible a una amplia audiencia de usuarios.

5. Eficiencia y optimización de recursos

[8] Implementar mecanismos para mejorar la eficiencia en el consumo de gas en las transacciones de la dApp.

4. DISEÑO

Al tratarse de una aplicación que se basa en Blockchain, su diseño es algo especial. El patrón arquitectónico utilizado es Modelo-Vista-Controlador (MVC) [19] en el que las diferentes partes se componen por:

- Modelo: está implementado por los contratos inteligentes de Reportes de estado y de Órdenes de alejamiento. Estos son los encargados de enviar los datos a la Blockchain. Su funcionamiento es parecido al de un API, en la que se pueden llamar a determinadas funciones para que maneje los datos de la forma deseada.
- Vista: se define la interfaz de usuario encargada de mostrar toda la información requerida para el correcto funcionamiento de la dApp.
- Controlador: implementado dentro de los componentes VUE, escrito en TypeScrip, donde se maneja la lógica de negocio y se realzan las llamadas a los contratos inteligentes desde la interfaz de usuario.

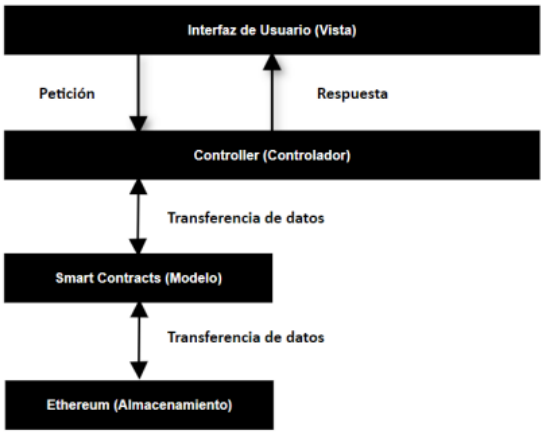

ILUSTRACIÓN 3 - DISEÑO

4.1. DISEÑO DE LA INTERFAZ DE USUARIO (FRONTEND)

El desarrollo del frontend requiere una etapa inicial de diseño donde se establezcan los componentes visuales que conformarán la interfaz. Esta fase de diseño es crucial para asegurar que la interfaz sea atractiva, intuitiva y fácil de utilizar.

Para obtener una representación de la interfaz, se emplearán capturas de pantalla que servirán como esqueleto de las páginas. Estas capturas permitirán visualizar la estructura y disposición de los elementos de manera eficaz, brindando una representación para el entendimiento y contexto de la dApp.

La interfaz se compone de tres vistas, en todas ellas habrá un componente de navegación en la parte superior con botones para navegar entre las diferentes vistas y conectar la billetera de metamask:

- **Inicio**: Esta vista es meramente informativa, donde se presenta el título de su funcionalidad principal, pequeñas advertencias y consideraciones sobre la violencia de género y una pequeña descripción de lo que se puede hacer dado el rol del usuario.
- **Reportes de estado:** dependiendo de si la billetera está conectada, o no. Habrá componentes que no se muestren hasta que se realiza el enlace. Una vez enlaza la billetera, se muestran todos los reportes existentes almacenados en la Blockchain tanto en el mapa como en un resumen de tarjetas con información del reporte.
- **Gestión de órdenes:** La vista presenta un título correspondiente a la ubicación del usuario, aunque algunos componentes no serán visibles si la billetera no está conectada. Una vez conectada, se muestra una tabla con órdenes disponibles, un buscador para filtrar por DNI y un botón para crear nuevas órdenes. En la tabla, una columna de "Incidencias" contiene un icono de bandera roja que, al hacer clic, abre un mapa con un selector de fechas para interactuar con las incidencias disponibles. También se incluyen botones de "reproducir" y "cerrar" para controlar la visualización y navegación en la tabla si hay demasiadas órdenes por página.

4.2. DISEÑO DEL MODELO (BACKEND)

La Blockchain utilizada para la realización de esta dApp es la Blockchain de Ethereum. Para poder utilizarla se utiliza Ganache. Esta herramienta permite levantar una cadena de bloques en un entorno local, teniendo así un entorno ideal para realizar desarrollos a nivel de usuario sin coste económico y con un control total del entorno. También se pueden visualizar los bloques y logs.

5. IMPLEMENTACIÓN

En este apartado se realiza la explicación de la implementación del diseño explicado en el apartado anterior, haciendo hincapié en todos aquellos aspectos complejos, más significativos o de mayor relevancia.

5.1. MODELO

5.1.1. CONTRATOS INTELIGENTES

Las transacciones en un contrato inteligente, cada operación realizada en la cadena de bloques requiere una tarifa de gas, que es una unidad de medida para medir la cantidad de recursos computacionales utilizados. El gas es pagado en la criptomoneda nativa de la cadena de bloques, como Ether en el caso de Ethereum. Por lo tanto, cada vez que se ejecuta una función o se realiza una transacción en un contrato inteligente, se debe pagar una cierta cantidad de gas.

Las consultas de datos en un contrato inteligente no alteran el estado de la cadena de bloques y, por lo tanto, no tienen costo asociado. Estas operaciones de solo lectura permiten a la dApp obtener información almacenada en el contrato sin modificar la cadena de bloques. Al declarar "public view" en la firma del método, se indica que la función solo será de lectura y, por lo tanto, será gratuita.

Desplegar un contrato inteligente puede ser una tarea relativamente sencilla gracias a las herramientas y plataformas actuales. Por ejemplo, en el contexto de este proyecto, se utiliza Hardhat que, a través de un fichero de configuración y despliegue en JavaScript, con solo un comando se puede compilar y desplegar el contrato en la Blockchain.

El sistema comprende dos contratos inteligentes: uno para el registro de órdenes de alejamiento y seguimiento de incidencias, y otro para la gestión de reportes en un mapa interactivo. Con el primer contrato, las víctimas pueden reportar incidentes de manera anónima en ubicaciones específicas, almacenando estos reportes en la Blockchain para que todos los usuarios puedan visualizar áreas potencialmente peligrosas. El segundo contrato permite el registro de órdenes de alejamiento y proporciona un seguimiento transparente y seguro de las interacciones entre las partes involucradas. Cada incidencia se registra en la cadena de bloques Ethereum en Ganache, asegurando la inmutabilidad y veracidad de los datos.

5.1.1.1. REPORTES DE ESTADO

El contrato inteligente "ReportesEstado", escrito en Solidity, tiene como objetivo principal permitir a los usuarios interactuar mediante el envío y recepción de mensajes geolocalizados a través de una plataforma interactiva.

Para enviar un mensaje, los usuarios utilizan la función "enviarMensaje" del contrato inteligente. Esta función requiere que los usuarios proporcionen el contenido del mensaje junto con las coordenadas de latitud y longitud que corresponden a la ubicación en coordenadas del mensaje.

Una vez que el mensaje es enviado, el contrato registra la información en una estructura de datos denominada "Mensaje", la cual almacena detalles relevantes como el texto del mensaje, las coordenadas, la dirección del remitente y el momento exacto del envío.

Además de facilitar el envío de mensajes, el contrato realiza un seguimiento del número total de mensajes enviados por cada usuario. Este seguimiento se lleva a cabo mediante el uso de un mapeo llamado "mensajesPorUsuario", que asocia cada dirección de Ethereum con un contador que representa la cantidad de mensajes enviados por dicho usuario.

Además de las funcionalidades de envío y recepción de mensajes, el contrato ofrece funciones de lectura, tales como "getMensajes" y "getNumMensajes". La función " getMensajes " devuelve la lista completa de mensajes almacenados en el contrato, lo que permite a los usuarios explorar y visualizar todos los mensajes enviados. Por su parte, " getNumMensajes " proporciona el recuento total de mensajes enviados, brindando a los usuarios una visión general de la actividad y participación en la plataforma.

5.1.1.2. ÓRDENES DE ALEJAMIENTO

Este contrato inteligente es el núcleo de la gestión de órdenes de alejamiento. Su propósito es facilitar la gestión de órdenes de alejamiento, permitiendo a los usuarios crear, almacenar y acceder a la información relacionada con dichas órdenes.

El contrato incluye distintas estructuras de datos para organizar la información de manera eficiente. Entre ellas, destacan las estructuras "Persona", que almacenan datos de las personas involucradas en la orden, como su DNI, nombre y apellidos. Asimismo, se encuentra la estructura "Incidencia", la cual registra detalles relevantes sobre eventos específicos asociados a una orden, tales como la marca de tiempo y las coordenadas geográficas de la víctima y el agresor.

```
struct Persona {
    string dni;
    string nombre;
    string apellidos;
}
 struct Incidencia {
    uint256 timestamp;
    string latitudVictima;
    string longitudVictima;
    string latitudAgresor;
    string longitudAgresor;
}

struct Orden {
    Persona victima;
    Persona agresor;
    uint256 distanciaMaxima;
}
```

ILUSTRACIÓN 4 - ESTRUCTURA DE DATOS EN BACKEND

Es importante destacar que le lenguaje de programación Solidity para la versión utilizada (^0.8.4) no permite la utilización de arrays dentro de una estructura de datos, por lo que la implementación de un array dentro de las órdenes de alejamiento no es posible. Para ello, se crea un mapeo de incidencias, que tiene forma de matriz, ya que se da un valor por cada array de incidencias. Este valor será el índice de la orden de alejamiento, pudiendo acceder así a todas las incidencias creadas para una orden de alejamiento dada.

```
mapping(uint256 => Incidencia[]) public incidenciasPorOrden;
Orden[] allOrdenes;
```

ILUSTRACIÓN 5 - DECLARACIÓN DE INCIDENCIAS Y ORDENES EN SOLIDITY

La dApp utiliza la función "crearOrden" para generar una nueva orden de alejamiento, proporcionando datos sobre la víctima, el agresor y la distancia mínima de alejamiento permitida. La orden creada se almacena en la estructura "Orden", que se agrega a la matriz "allOrdenes" y se indexa en el mapeo "ordenesPorUsuario". Esto permite realizar un seguimiento del número de órdenes creadas por cada usuario en la plataforma..

```
function crearOrden(
    string memory dniVictima,
    string memory nombreVictima,
    string memory apellidosVictima,
    string memory dniAgresor,
    string memory nombreAgresor,
    string memory apellidosAgresor,
    uint256 distanciaMinima
) public {
    Persona memory nuevaVictima = Persona(dniVictima, nombreVictima, apellidosVictima);
    Persona memory nuevoAgresor = Persona(dniAgresor, nombreAgresor, apellidosAgresor);

    allOrdenes.push(Orden(nuevaVictima, nuevoAgresor, distanciaMinima));
    ordenesPorUsuario[msg.sender] += 1;

     console.log("numero de ordenes de",msg.sender," %s >> ", ordenesPorUsuario[msg.sender]);

    //se utiliza el hash del DNI ya que con string no se puede utilziar como
    //clave de mapeo. De esta forma podemos seguir utilziando el dni como clave indirecta.
    uint256 dniVictimaFormat = stringToUint(dniVictima);
    uint256 dniAgresorFormat = stringToUint(dniAgresor);

    dniPorOrdenes[dniVictimaFormat].push(numOrdenes); // Indexar el DNI de la víctima
    dniPorOrdenes[dniAgresorFormat].push(numOrdenes); // Indexar el DNI del agresor

    numOrdenes++;
}
```

ILUSTRACIÓN 6 - CREAR UNA NUEVA ORDEN

Además, el contrato lleva a cabo un mapeo de los DNI de las personas involucradas en las órdenes, indexándolos a los identificadores correspondientes de las órdenes en la matriz "dniPorOrdenes". Esta función facilita la búsqueda y recuperación de órdenes relacionadas con una persona específica, ya sea víctima o agresor. Este método se utilizará para relacionar una pareja de agresor/víctima con sus órdenes asociadas, de esta manera se gestionan las incidencias a esa orden relaciona en el momento de la carga de datos de los incidentes ocasionados.

```
mapping(address => uint256) public ordenesPorUsuario;
mapping(uint256  => uint[]) public dniPorOrdenes; // Mapeo
```

ILUSTRACIÓN 7 - MAPEOS DE ÓRDENES E INDEXACIÓN DE DNI

Para el registro de nuevas incidencias relacionadas con una orden en particular, los usuarios pueden utilizar la función "agregarIncidencia". Esta función permite incluir detalles como la fecha y las coordenadas geográficas de la víctima y el agresor en el momento de la incidencia. Las incidencias se almacenan en una matriz de matrices denominada "incidenciasPorOrden", donde cada fila representa las incidencias asociadas a una orden específica, como se ha explicado anteriormente.

```
// Sera necesaria la posicion en el array para informar las incidencias de una orden
function agregarIncidencia(uint256 ordenIndex, uint256 fecha,
                           string memory _latV, string memory _lngV,
                           string memory _latA, string memory _lngA) public {
    require(ordenIndex < allOrdenes.length, "Indice de orden invalido");
    Incidencia memory incidencia = Incidencia(fecha, _latV, _lngV, _latA, _lngA);
    incidenciasPorOrden[ordenIndex].push(incidencia);
}
```

ILUSTRACIÓN 8 - MÉTODO PARA AGREGAR INCIDENCIAS

En cuanto a la consulta de información, el contrato proporciona funciones de lectura. La función "getAllOrdenes" devuelve todas las órdenes almacenadas en el contrato, mientras que "getTotalOrdenes" devuelve el recuento total de órdenes creadas desde la inicialización del contrato.

```
function getAllOrdenes() public view returns (Orden[] memory) {
    return allOrdenes;
}

function getTotalOrdenes() public view returns (uint256) {
    return numOrdenes;
}
```

ILUSTRACIÓN 9 - MÉTODOS PARA LA OBTENCIÓN DE ÓRDENES

5.1.2. DESPLIEGUE

Para realizar el despliegue es necesario tratar que se utiliza “npm hardhat” para la creación del setUp del proyecto. Este comando despliega la estructura necesaria para comenzar a desplegar contratos inteligentes en la Blockchain.

Primero, es necesario completar el archivo configuración “hardhat.config.js” creado:

```
module.exports = {
  solidity: '0.8.4',
  paths: {
    sources: './solidity/contracts',
    artifacts: './src/artifacts',
  },
  networks: {
    hardhat: {
      chainId: 1337,
    },
    ganache: {
      chainId: 1337,
      url: 'http://127.0.0.1:7545',
    }
```

ILUSTRACIÓN 10 - CONFIGURACIÓN DE DESPLIEGUE DE CONTRATOS INTELIGENTES

En este se indican varios parámetros como “sources” donde se indica el directorio de origen donde se van a almacenar los contratos inteligentes. Y “artifacts” que almacena los archivos compilados de los contratos. Se indica también, las redes que se utilizan, el caso que requiere el proyecto es Ganache, que servirá para posteriormente el comando de compilación.

El archivo de compilación se utiliza para compilar y desplegar uno o varios contratos inteligentes en la cadena de bloques, se muestra la porción de código encarga de esta funcionalidad:

```
const ordenAlejamientoFactory = await hre.ethers.getContractFactory('OrdenAlejamiento')
const ordenAlejamientocontract = await ordenAlejamientoFactory.deploy({})

await ordenAlejamientocontract.deployed()

console.log('OrdenAlejamiento address: ', ordenAlejamientocontract.address)
```

ILUSTRACIÓN 11 - EJEMPLO DE CÓDIGO PARA DESPLEGAR EL CONTRATO DE "ORDENESALEJAMIENTO"

Una vez realizada la configuración inicial y el contrato inteligente, se utiliza el siguiente comando para compilar y desplegar el contrato en la Blockchain ‘*npx hardhat run ./solidity/scripts/deploy.js --network ganache*’,

5.2. VISTA

Para organizar la vista de la DApp, se han creado diversos componentes Vue.js que abarcan diferentes aspectos funcionales de la aplicación. A continuación, se describen algunos de los componentes principales:

5.2.1. BILLETERA METAMASK

Se utiliza la biblioteca “pinia” para definir y crear un almacén de estado para la gestión de datos relacionados con la billetera que se va a conectar. Esta biblioteca se utiliza para almacenar el estado de la billetera a lo largo de las interacciones con la dApp.

```
import { defineStore } from 'pinia'

export const useWalletStore = defineStore('wallet', {
  state: () => {
    return { walletData: null }
  },

  actions: {
    //@ts-ignore
    saveWalletData(payload: any) {
      this.walletData = payload
    },
```

ILUSTRACIÓN 12 - MÉTODO PARA EL MANEJO DE LA BILLETERA METAMASK EN TODA LA APLICACIÓN

La propiedad "walletData" en el almacén de estado sirve como el lugar central donde se pueden almacenar los detalles de la billetera, como el saldo, las transacciones recientes o cualquier otra información relevante. Al tener esta propiedad en un almacén de estado separado, evitamos la necesidad de pasar los datos entre componentes de manera manual, lo que reduce la complejidad del código y mejora la claridad.

Además, la acción "saveWalletData" nos permite actualizar fácilmente los datos de la billetera. Cuando se necesita actualizar la información de la billetera, simplemente llamamos a esta acción y le proporcionamos los nuevos datos a través del argumento "payload". La acción se encarga de modificar el estado del almacén, lo que a su vez notificará automáticamente a todos los componentes que dependen de "walletData" para que muestren los datos actualizados.

En cada componente donde se requiera su utilización se importa este almacén conjunto y se define una constante en el “setup()” del componente VUE, como se ve en as siguiente imágenes.

```
import { useWalletStore } from '@/stores/wallet'

setup() {
  const walletStore = useWalletStore()
```

ILUSTRACIÓN 13 - INSTANCIAR LA BILLETERA

5.2.2. NAVBAR

Representa la barra de navegación superior de la DApp. Contiene enlaces y el botón para conectar con la billetera de metamask.

ILUSTRACIÓN 14 - BARRA DE NAVEGACIÓN

A su vez, este diseño también se ha implementado por si se accede desde un teléfono móvil para que sea responsive.

ILUSTRACIÓN 15 - FORMATO TELÉFONO

5.2.3. REPORTES DE ESTADO

Se muestra la vista de los reportes que hacen los diferentes usuarios de la aplicación. Si no está conectada la billetera se mostrará de la siguiente manera.

ILUSTRACIÓN 16 - PÁGINA DE REPORTES DE ESTADO

En cambio, si se conecta la billetera. Se podrán ver más datos acerca de los reportes de estado.

ILUSTRACIÓN 17 - PÁGINA DE REPORTES DE ESTADO CON LA BILLETERA CONECTADA

Y la versión en el teléfono móvil.

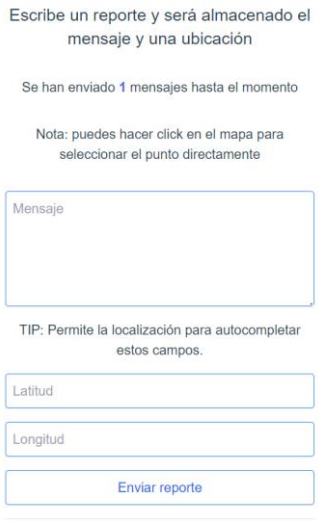

ILUSTRACIÓN 18 - EJEMPLO EN FORMATO TELÉFONO

Cada marcador tiene un mensaje asociado. Este mensaje es el que ha dejado el usuario. Si se pasa el cursor por encima de esté, se podrá obtener más información del reporte.

ILUSTRACIÓN 19 - DETALLE DEL MAPA DE REPORTES DE ESTADO

Para la introducción de datos se podrá seleccionar en el mapa o conceder permisos al navegador para obtener la ubicación del dispositivo con el que se está accediendo. Debajo, se muestra el resumen de reportes en forma de tarjetas con distintos datos de estos.

ILUSTRACIÓN 20 - DETALLE DEL REPORTE A INTRODUCIR

5.2.4. ÓRDENES DE ALEJAMIENTO

En este componente se realiza el seguimiento de las órdenes de alejamiento y sus incidencias relacionadas. Una vez se haya conectado la cartera y esta tenga los roles correspondientes, se mostrará la vista que se muestra en la figura 21. Nótese que se ha añadido un nuevo botón a la barra de navegación.

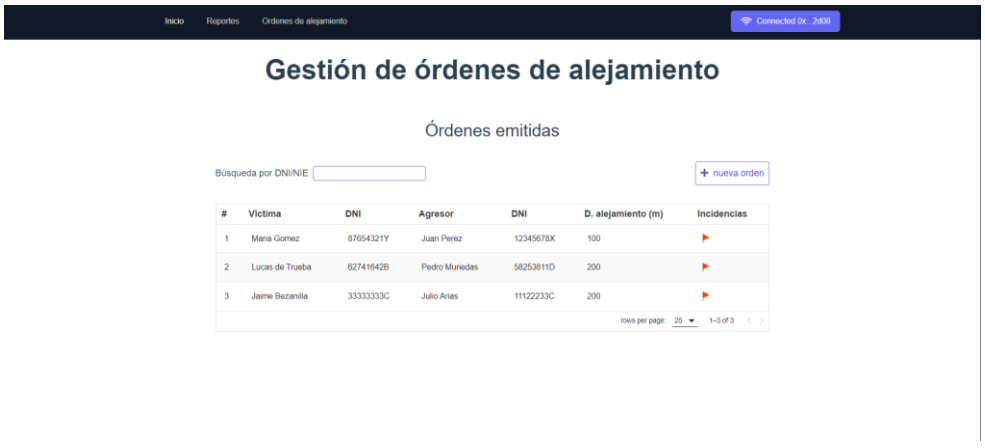

ILUSTRACIÓN 21 - PAGINA DE GESTIÓN DE ÓRDENES DE ALEJAMIENTO

Como se observó en el diseño, esta vista comprende diferentes partes. La primera es el buscador de la tabla. Un campo de texto donde se puede introducir cualquier valor, como, por ejemplo, la letra de DNI.

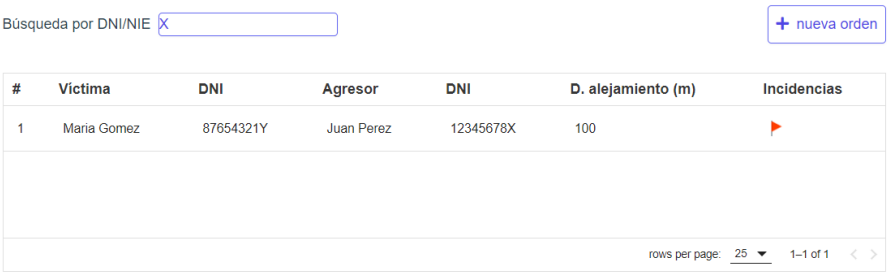

ILUSTRACIÓN 22 - EJEMPLO DE BÚSQUEDA POR DNI

Se tiene la columna de incidencias donde se ve representado por el icono de una bandera roja. Si este se pulsa, se abrirá un dialogo con toda la lógica para ver las incidencias de la orden de alejamiento seleccionada como se muestra en la siguiente imagen.

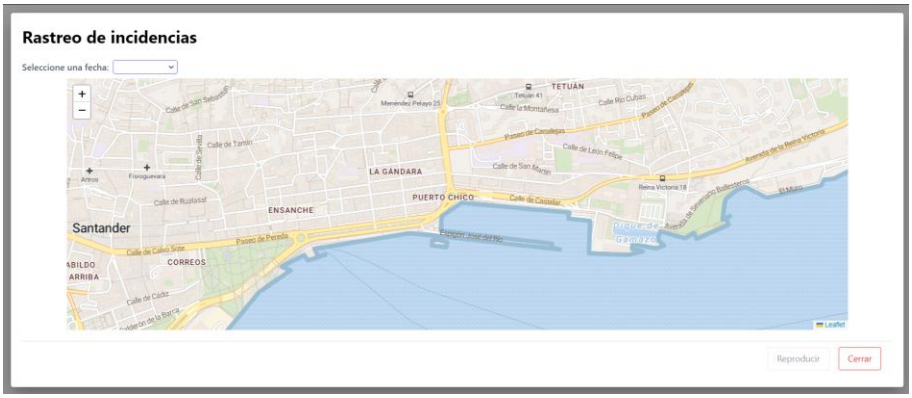

ILUSTRACIÓN 23 - VISUALIZACIÓN DEL MAPA DE SEGUIMIENTO DE INCIDENCIAS

En esta nueva ventana se ve a primera vista un mapa, un selector de fechas y dos botones; el botón para dar comienzo a la reproducción de las incidencias, y el botón que cierra la ventana emergente.

Si se selecciona una fecha, el botón se desbloqueará.

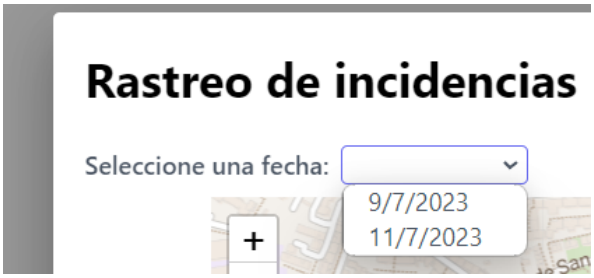

ILUSTRACIÓN 24 - SELECTOR DE FECHA

Y se podrá dar comienzo a la visualización de la incidencia. En esta simulación, se ven dos iconos. El icono rojo representa el agresor, y el icono verde a la víctima. A continuación, se muestra el resultado de la simulación.

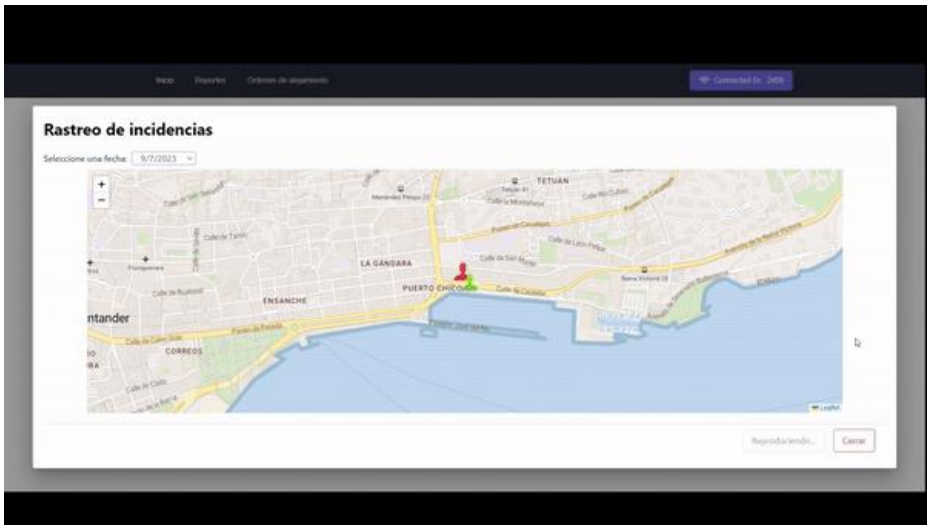

ILUSTRACIÓN 25 - VISUALIZACIÓN DE UN PUNTO DE LA INCIDENCIA

Para la creación de una nueva orden, se ha de volver a la pantalla principal de gestiones de órdenes de alejamiento. Encima de la tabla se verá, a la derecha, un botón con el texto “nueva orden”. Pulsándolo se abrirá una ventana emergente donde introducir los datos de la víctima y agresor, además de la distancia mínima. No es posible realizar el envío de la orden hasta que todos los campos que se ven en la figura 26.

ILUSTRACIÓN 26 - FORMULARIO DE EMISIÓN DE INCIDENCIAS

5.3. CONTROLADOR

Su principal función radica en procesar y gestionar los datos que provienen de la interfaz de usuario, garantizando que sean adecuadamente preparados y enviados a los contratos inteligentes para su registro en la cadena de bloques. Además, el controlador es el encargado de recibir y procesar las respuestas provenientes de la Blockchain, presentando los resultados y actualizaciones en la interfaz de usuario para ofrecer una experiencia fluida a los usuarios de la dApp.

En esta sección se tratarán los aspectos clave de la dApp, como el seguimiento de reportes de estado, en control de órdenes de alejamiento e incidencias. Se explicará en detalle cómo se

interactúa con un contrato inteligentes y aquellas partes que son necesarias para tal fin. Además, como se estructuran los métodos y un ejemplo de código relevante para ilustrar el flujo del proceso en el que un usuario interactúa con el frontal web hasta que los datos son recogidos en la Blockchain y vuelven a ser representados en el frontal.

A partir de este punto, se supone que se tiene una conexión establecida con el servidor Ganache a través de una billetera Metamask, por lo tanto, los componentes web se verán con su funcionalidad completa. La cuenta tendrá los roles de funcionario.

5.3.1. COMPONENTE: REPORTES DE ESTADO

Como bien se explicó con anterioridad, en la vista de reportes de estado se encuentran un mapa y 3 campos para insertar datos. Estos campos serán el mensaje y las coordenadas que se almacenarán en la Blockchain con el reporte de estado.

Este flujo comienza en el punto en el que el usuario conecta su cartera. A través de un observador o “watch” se hace un seguimiento del estado de la billetera conectada. La opción "watch" se utiliza para observar cambios en una propiedad específica del componente y tomar acciones en respuesta a esos cambios.

En este caso, el código está observando la propiedad "accAvailable". Cada vez que cambie el valor de "accAvailable", se ejecutará el código dentro del bloque de "watch".

```
watch: {
  accAvailable(newVal, old) {
    console.log(`actualizando deesde ${old} a ${newVal}`)
    this.getNumMensajes()
    this.getMensajes()
    this.getPositionsCoors()
```

ILUSTRACIÓN 27 - WATCHER PARA EL SEGUIMIENTO DE CAMBIO DE BILLETERA

Otra forma de obtener estos datos es través de un web hook del flujo de vida de un componente de Vue.js “mounted”, ocurre después de que el componente ha sido insertado en el DOM y es visible para el usuario. Se obtienen los mensajes y el número total de mensajes almacenados en la Blockchain. Previamente se comprueba que la billetera esté conectada.

```
mounted() {
  if (this.walletStore.walletData !== null) {
    console.log('Billetera conectada.')
    this.getNumMensajes()
    this.getMensajes()
    this.getPositionsCoors()
  }
},
```

ILUSTRACIÓN 28 - MÉTODO DE ACTUALIZACIÓN DE DATOS AL CONECTAR LA BILLETERA

La obtención de todos los mensajes se realiza a través de una función asíncrona en la que se invoca al método “getMensajes” el cual devuelve el array creado en el modelo donde se almacenan todos los reportes de estado introducidos por el usuario. Una vez obtenidos, se trasladan a una variable local, la cual va a ser procesada por la vista para mostrar cada reporte en el mapa.

```
const getMensajes = async function () {
  reportesEstado.value = []

  //@ts-expect-error Window.ethers not TS
  if (typeof window.ethereum !== 'undefined') {
    //@ts-expect-error Window.ethers not TS
    const provider = new ethers.providers.Web3Provider(window.ethereum)
    const contract = new ethers.Contract(
      contractAddress,
      ReportesEstado.abi,
      provider
    )
    try {
      const data = await contract.getMensajes({})
      console.log('Todos los reportes :>> ', data)
      data.forEach((rep: any) => {
        reportesEstado.value.push({
          from: rep.from,
          text: rep.text,
          lat: rep.lat,
          lng: rep.lng,
          //@ts-expect-error numbers...
          latLng: [parseFloat(rep.lat), parseFloat(rep.lng)],
          datetime: new Date(rep.datetime * 1000),
        })
      })
    } catch (error) {
      console.error(error)
    }
  }
}
```

ILUSTRACIÓN 29 - MÉTODO PARA LA OBTENCIÓN

Cabe destacar que, en la instanciación del contrato, se utiliza un “provider” y no un “signer” como se verá más adelante. Esto es debido a que solo consultamos los datos a través de vistas públicas declaradas en el contrato. Estas consultas no tienen coste alguno asociada, son totalmente gratuitas.

Para el flujo de creación de reportes, es necesario rellenar los 3 campos que se presentan debajo del mapa. Si no se conocen las coordenadas de la ubicación, al pulsar en el mapa, se captura ese evento con la latitud y longitud del punto seleccionado.

```
const getClickLatLngMap = function(event:LeafletMouseEvent){
  latitude.value = ''
  longitude.value = ''
  console.log(event.latlng)
  latitude.value = event.latlng.lat.toString()
  longitude.value = event.latlng.lng.toString()
}
```

ILUSTRACIÓN 30 - MÉTODO PARA CAPTURAR UN PUNTO SELECCIONADO EN EL MAPA

En su defecto, se puede dar permisos al navegador para conocer la localización del dispositivo con el que se está accediendo a la dApp.

Una vez se ha informado el campo de texto donde se introduce el mensaje a reportar y las coordenadas por cualquiera de los tres métodos mencionados, se puede enviar el reporte a la Blockchain a través del contrato inteligente desplegado para este fin.

```
const enviarReporte = async function () {
  //@ts-expect-error Window.ethers no es TS
  if (typeof window.ethereum !== 'undefined') {
    trxInProgress.value = true
    //@ts-expect-error Window.ethers no es TS...
    const provider = new ethers.providers.Web3Provider(window.ethereum)
    const signer = provider.getSigner()

    const contract = new ethers.Contract(
      contractAddress,    //direccion del contrato desplegado en la blockchain
      ReportesEstado.abi, //manual del contrato
      signer              // para fimar la transaccion
    )
    try {
      //se crea la trasaccion con la blockchain con el metodo del contrato inteligente
      const transaction = await contract.sendWave(message.value, latitude.value, longitude.value, {
        gasLimit: 300000,
      })

      console.log('transacción :>> ', transaction)
      await transaction.wait()
      message.value = ''
      trxInProgress.value = false
      notify({
        title: "El reporte se ha enviado correctamente",
        group: "dapp",
        type: "success"
      });
      //Se refrescan los mensajes de la vista
      //@ts-expect-error error por TS
      this.getMensajes()
      //@ts-expect-error error por TS
      this.getNumMensajes()

    } catch (error) {
```

ILUSTRACIÓN 31 - MÉTODO PARA ENVIAR UN NUEVO REPORTE

Se puede observar, que, en la inicialización del contrato, esta vez, se está utilizando un firmante y no un proveedor. Esto es debido a que los bloques minados en la Blockchain tienen una serie de transacciones y cada una de ella va a nombre de una dirección de cartera de Ethereum, este es el “signer” mostrado en esta parte del código. Por lo tanto, en este caso al realizar una inserción o modificación de datos en la Blockchain, se necesita saber qué dirección de cartera ha realizado esa acción a través de una transacción, el firmante.

El ABI del contrato es una especie de manual de instrucciones que le dice a otros programas cómo comunicarse con un contrato inteligente en la red Blockchain. Es un conjunto de reglas y especificaciones que define cómo se deben enviar y recibir datos desde y hacia el contrato inteligente. Este manual se genera después de la compilación del contrato y se almacena en el proyecto en formato JSON.

5.3.2. COMPONENTE: ÓRDENES DE ALEJAMIENTO

La gestión de órdenes de alejamiento presenta una estructura más compleja debido a la implementación de incidencias dentro de las órdenes de alejamiento, lo que conlleva a la creación de mapeos para indexar los DNI, que relaciona las órdenes por número de DNI. Esta decisión se ha tomado con el objetivo de abordar eficientemente la escalabilidad de la solución. A medida que el número de incidencias u órdenes de alejamiento aumenta con el tiempo, realizar búsquedas tradicionales con arreglos en el contrato inteligente sería costoso y poco viable. Por lo tanto, la utilización de un mapeo proporciona una solución óptima al relacionar directamente las órdenes de alejamiento con los DNI de las víctimas o agresores involucrados, asegurando una gestión eficiente y sostenible en el tiempo. Esta implementación es fundamental para garantizar un funcionamiento fluido y eficaz en la DApp a medida que la cantidad de datos registrados en la Blockchain aumenta.

Se obviarán algunos métodos de obtención e inserción de datos, como las órdenes de alejamiento y los “watchers” debido a que se realiza de forma parecida de la que se obtienen los reportes de estado descritos en el apartado anterior.

Para comenzar, se definen tres interfaces para ordenar la representación d datos teniendo una estructura constante a lo largo del proceso.

```
interface Persona {
  dni: string
  nombre: string
  apellidos: string
}

interface Orden {
  victima: Persona
  agresor: Persona
  distanciaMax: number
}
interface Incidencia {
    datetime: string;
    latLng: number[];
}
```

ILUSTRACIÓN 32 - ESTRUCTURA DE DATOS PARA LAS ÓRDENES DE ALEJAMIENTO

Como se observó en la vista, al consultar las incidencias de una orden de alejamiento, es necesario realizar una consulta a la Blockchain para poder visualizar sus incidencias. Primero se obtienen las órdenes de alejamiento de cada una de las partes y se busca la coincidente entre ellos. En este método se inicializa ya el contrato, al ser solo de consulta, se utiliza el “provider” para obtener los datos.

```
const ordenesV = await obtenerOrdenesPorDNI(val.victimaDni, contract);
const ordenesA = await obtenerOrdenesPorDNI(val.agresorDni, contract);

const coincidencia = ordenesV.find((orden) => ordenesA.includes(orden));
```

ILUSTRACIÓN 33 - EJEMPLO DE OBTENCIÓN DE ÓRDENES INDEXADAS POR DNI

La búsqueda de órdenes por DNI se muestra el resultado de la indexación por el posible caso de incremento de registros. Esta obtención se muestra en la siguiente figura.

```
async function obtenerOrdenesPorDNI(dni: string, contract:ethers.Contract): Promise<number[]> {

  try {
    const resultado = await contract.buscarOrdenesPorDNI(dni);

    const ordenes: number[] = resultado.map((valor:any) => valor.toNumber()); // se completa el array de ordenes en forma de number
    return ordenes;
  } catch (error) {
    console.log(error)
    notify({
        title: "Se ha producido un error al obtener las ordenes por DNI. Consulte con un adinistrador para obtener más información.",
        group: "dapp",
        type: "error"
      });
  }
  return []
}
```

ILUSTRACIÓN 34 - MÉTODO DE OBTENCIÓN DE ÓRDENES POR DNI Y NOTIFICACIÓN AL USUARIO

Una vez obtenido el número de Orden que relaciona a la víctima con el agresor, se obtienen sus incidencias asociadas. Estás estarán seguidas unas de otras, es decir, será necesario distinguir el número de fechas diferentes, por día, de todas las incidencias obtenidas. Esto se realiza con una instancia de la clase “Set” en TypeScrip donde se almacena una colección de valores únicos. Se utilizará para filtrar las incidencias mostradas y para el selector de fechas en la vista del componente, como se explicó en el apartado 5.3.

```
try {
  const data = await contract.obtenerIncidencias(coincidencia);
  //console.log('Incidencias :>> ', data);
  const fechasUnicasSet = new Set();
  data.forEach((inc: any) => {
    if(inc.latitudAgresor != "" && inc.longitudAgresor != "" && inc.latitudVictima!="" &&
       inc.longitudVictima !="" && inc.timestamp != null) {

      incidenciasV.value.push({
        latLng: [parseFloat(inc.latitudVictima), parseFloat(inc.longitudVictima)],
        datetime: new Date(inc.timestamp * 1000).toLocaleDateString('es-ES'),
      })
      incidenciasA.value.push({
        latLng: [parseFloat(inc.latitudAgresor), parseFloat(inc.longitudAgresor)],
        datetime: new Date(inc.timestamp * 1000).toLocaleDateString('es-ES'),
      })
    }
    const fecha = new Date(inc.timestamp * 1000).toLocaleDateString('es-ES');
    fechasUnicasSet.add(fecha);
  })
  //@ts-expect-error sets...
  fechasDisponibles.value = [...fechasUnicasSet] //solo fechas unicas en el selector
```

ILUSTRACIÓN 35 - MÉTODO PARA OBTENER LAS INCIDENCIAS

Las colecciones de incidencias “incidenciasA” siendo referentes a las incidencias del agresor y las “incidenciasV” a las de la víctima, serán el conjunto de incidencias que se utilizará para realizar el seguimiento de la infracción de ruptura de distancia de alejamiento en el mapa.

Al no existir el tipo de dato “Date” en Solidity, el timestamp obtenido es en formato de milisegundos, por lo que es necesaria una conversión a un formato viable para la representación de la fecha en el proyecto. Hay que destacar que, además que todas las funciones vistas para interactuar con contratos son funciones asíncronas que devuelven un objeto “Promise<T>” del tipo de valor T requerido para cada caso.

```
async function obtenerOrdenesPorDNI(dni: string, contract:ethers.Contract): Promise<number[]> {
```

ILUSTRACIÓN 36 - SIGNATURA TIPO DE MÉTODOS QUE INTERACTÚAN CON CONTRATOS INTELIGENTES

Además, se realizan para cada una de las llamadas el bloque “try-catch” para el manejo de excepciones, donde se imprime un mensaje por consola en la depuración de la herramienta y una notificación al usuario para que sea consciente del estado de la aplicación en todo momento.

```
} catch (error) {
  notify({
  title: "Se ha producido un error al obtener las incidencias. Consulte con un administrador para obtener más información.",
  group: "dapp",
  type: "error"
});
  console.error(error);
}
```

ILUSTRACIÓN 37 - EJEMPLO DE NOTIFICACIÓN DE ERROR

Y el usuario vería lo siguiente una vez se haya cometido el error.

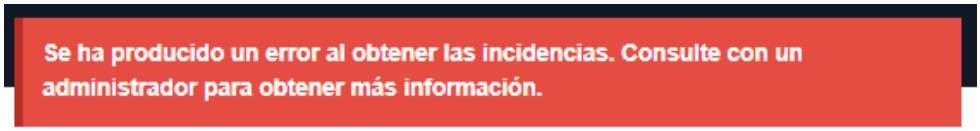

ILUSTRACIÓN 38 - NOTIFICACIÓN DE ERROR

6. VALIDACIÓN

En este apartado, se presentan las pruebas realizadas para validar el software desarrollado en la dApp. Las pruebas se han realizado con el objetivo de garantizar el correcto funcionamiento, la seguridad y la robustez de la aplicación descentralizada. Se han realizado cuatro tipos de pruebas: pruebas unitarias, de integración, de sistema y de aceptación.

6.1. UNITARIAS

En estas pruebas se han recopilado las diferentes funcionalidades de cada uno de los contratos inteligentes. Para los reportes de estado se ha creado el siguiente.

```
// SPDX-License-Identifier: UNLICENSED

pragma solidity ^0.8.4;

import "truffle/Assert.sol";
import "truffle/DeployedAddresses.sol";
import "../contracts/ReportesEstado.sol";

contract TestReportesEstado {
    ReportesEstado reportesContract;
    address usuarioPrueba; // Dirección de prueba para simular diferentes usuarios

    // Se ejecuta antes de cada prueba para desplegar el contrato y asignar una dirección de prueba
    function beforeEach() public {···
    }

    // Caso de Prueba 1: Crear Reporte con Datos Inválidos
    function testCrearReporteDatosInvalidos() public {···
    }

    // Caso de Prueba 2: Contador de reportes por usuario
    function testContadorReportesPorUsuario() public {···
    }

    // Caso de Prueba 3: obtener reportes vacios
    function testObtenerReportesVacios() public {···
    }

    // Caso de Prueba 4: obtener numero total de reportes
    function testObtenerNumeroTotalReportes() public {···
    }
```

ILUSTRACIÓN 39 - TESTS PARA LOS REPORTES DE ESTADO

Los resultados de las siguientes pruebas son procesados por la herramienta truffle, se vería de la siguiente forma.

```
Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.


  Contract: TestReportesEstado
    ✓ testCrearReporteDatosInvalidos (87ms)
    ✓ testContadorReportesPorUsuario (64ms)
    ✓ testObtenerReportesVacios (64ms)
    ✓ testObtenerNumeroTotalReportes (77ms)


  4 passing (2s)
```

ILUSTRACIÓN 40 - RESULTADOS DE PRUEBAS

Para la gestión de órdenes de alejamiento.

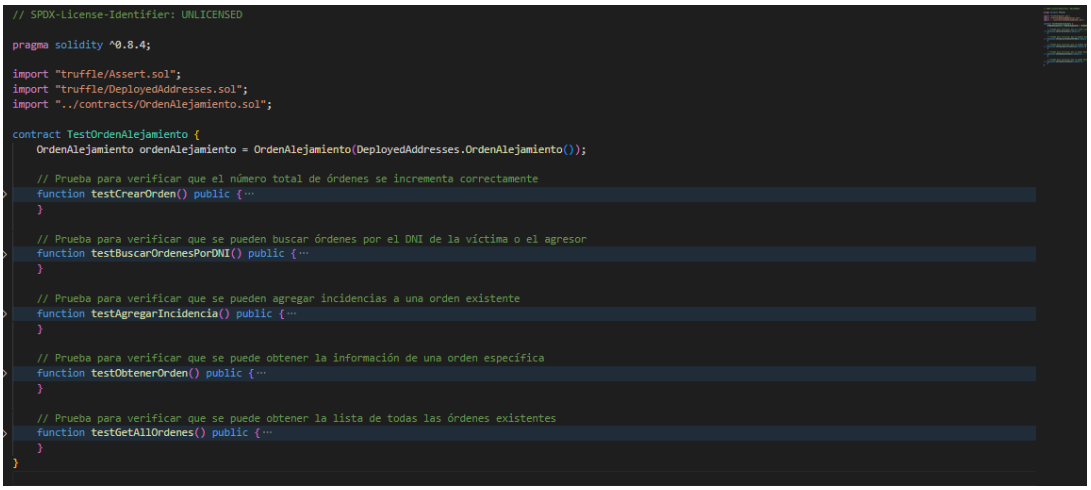

ILUSTRACIÓN 41 - TESTS PARA LAS ÓRDENES DE ALEJAMIENTO

Los resultados de las pruebas fueron los mostrados en la siguiente figura.

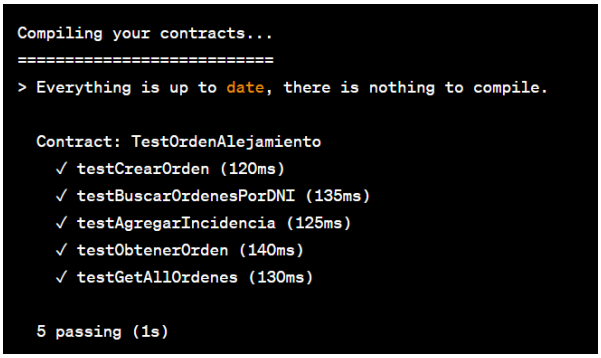

ILUSTRACIÓN 42 - RESULTADOS DE TESTS

6.2. INTEGRACIÓN

Las pruebas de integración se han realizado entre la capa controlador y la del modelo. Para llevar a cabo los casos de prueba se ha usado un sistema de log verificando los siguientes escenarios de forma manual.

- Prueba de creación y consulta de reportes de estado:
  - Crear un reporte de estado válido con datos válidos de latitud y longitud.
  - Verificar que el reporte se almacena correctamente en el contrato "ReportesEstado".
  - Consultar los reportes creados para asegurarse de que el nuevo reporte esté presente en la lista.
  - Intentar crear un reporte de estado con datos inválidos y verificar que se rechace adecuadamente.
- Prueba de creación y consulta de órdenes de alejamiento:
  - Crear una orden de alejamiento válida con datos válidos de víctima, agresor y distancia mínima.

- Verificar que la orden se almacena correctamente en el contrato "OrdenAlejamiento".
- Consultar las órdenes creadas para asegurarse de que la nueva orden esté presente en la lista.
- Intentar crear una orden de alejamiento con datos inválidos y verificar que se rechace adecuadamente.

- Prueba de asociación de Incidencias con órdenes de alejamiento:
  - Crear una orden de alejamiento.
  - Agregar varias incidencias asociadas a esa orden.
  - Verificar que las incidencias se agregan correctamente al contrato "OrdenAlejamiento" y que están asociadas a la orden correspondiente.

6.3. SISTEMA

Para la verificación de los requisitos no funcionales mencionados en el Capítulo 3, apartado 2, se han hecho una serie de pruebas para verificar su cumplimiento.

6.3.1. PRUEBAS DE SEGURIDAD Y PRIVACIDAD

- **Ataque de reentrada**: para el uso de la aplicación es condición necesaria conectar una billetera metamask. Para cada transacción realizada, se ha de confirmar la transacción por su extensión web o en la aplicación del teléfono móvil, por lo que se hace imposible la posibilidad de realizar este tipo de ataques.
- **Desbordamiento de enteros:** el único campo afectado es el que se encarga de recoger la distancia de alejamiento de una nueva orden. Este campo es de tipo numérico y tiene dos atributos adicionales en su etiqueta HTML, máximo y mínimo, por el cual se protege la entrada de números enteros negativos y demasiado grandes.
- **Pruebas de manejo de excepciones**: en cada petición realizada por el controlador al modelo, se hace mediante un bloque “try-catch” destinado al manejo de excepciones dentro de la aplicación. Cada uno de ellos se dispone un mensaje de error con el problema ocurrido.
- **Pruebas de privacidad**: para estas pruebas se ha identificado que ninguna parte del código almacena los datos personales de los usuarios finales. Solo se puede identificar a un usuario con su ID de billetera Metamask y, a su vez, esta no es vinculante con ningún tipo de dato personal. Con ello, se accede a la dApp de

forma totalmente anónima por lo que datos personales no se almacenan nunca a través de los contratos inteligentes.

6.3.2. RENDIEMIENTO Y ESCALABILIDAD

Para este tipo de prueba se ha realizado una carga de llamadas considerable en comparación con las que hace un usuario normal de la aplicación. En estas pruebas se ha saltado la verificación de metamask en el programa de carga de incidencias. Para estas pruebas se ha escogido un fichero de carga con 50 incidencias cada uno de los implicados y, posterior revisión por el programa de carga, se han introducido de forma continua a la Blockchain a través de transacciones. Ya es responsabilidad de la propia tecnología Blockchain soportar esta alta escalabilidad de transacciones, y de la dApp de tratarlas adecuadamente. En la Blockchain de Ethereum, en su versión actual, la 2.0 puede llegar a manejar 100.000 transacciones [22] por segundo, lo que habilita de una gran escalabilidad en la red.

6.4. ACEPTACIÓN

Se han realizado pruebas de aceptación utilizando diversos usuarios, que incluyen desde personas con poca experiencia en el mundo digital, como una ama de casa, hasta un funcionario jubilado del cuerpo nacional de policía. La participación de estos usuarios ha sido de gran ayuda para simular diferentes escenarios de uso y evaluar la funcionalidad de la dApp.

Durante las pruebas, se ha identificado una preocupación común entre los usuarios relacionada con la retroalimentación visual de las acciones realizadas en la aplicación. Para abordar esta preocupación, se han implementado elementos visuales adicionales, como sombras y resaltos, que permiten a los usuarios saber claramente si han pulsado un botón o interactuado con elementos específicos en la interfaz.

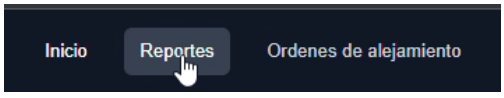

ILUSTRACIÓN 43 - EJEMPLO CORRECCIÓN DISEÑO INTERFAZ

Además, se ha recopilado valiosa retroalimentación sobre la disposición de los elementos en la pantalla y su usabilidad, lo que ha permitido realizar mejoras en la dApp. Se han tenido en cuenta las sugerencias proporcionadas por los usuarios para optimizar la experiencia de usuario y hacer que las diferentes acciones posibles en la aplicación sean más intuitivas y accesibles.

ILUSTRACIÓN 44 - EJEMPLO DE CORRECCIÓN INTERFAZ 2

Estas pruebas de aceptación con usuarios reales han sido fundamentales para detectar áreas de mejora en la dApp y garantizar una experiencia de usuario más satisfactoria. Gracias a la participación y aportes de los usuarios, se ha logrado ajustar la interfaz y la funcionalidad para asegurar que la aplicación sea más amigable y fácil de usar para diferentes tipos de usuarios.

6.5. PROGRAMA DE CARGA

Este apartado tiene como objetivo principal explicar cómo las incidencias son creadas, su lógica para discriminar posiciones en el mapa aptas y no aptas para su almacenamiento en la Blockchain. Este programa de carga se encarga de procesar y gestionar los datos recibidos del agresor y la víctima, desde un fichero, y realizar las operaciones necesarias para tomar decisiones basadas en esta información. Este programa se ha implementado en un archivo TypeScript, junto con un archivo JSON [21] que contiene los datos necesarios para simular la recepción de coordenadas del agresor y la víctima. Cuando se habla de simulación, se hace referencia que estos datos han sido creados y seleccionados especialmente para la realización de este proyecto, con el fin de proporcionar datos que simularían una interacción entre dos actores para guardar el registro con una lógica determinada.

Este programa de carga es totalmente externo a la dApp tratada hasta ahora. Su único propósito es proporcionar datos para probar el correcto funcionamiento de la aplicación, no forma parte de ninguna de las partes implicadas y explicadas anteriormente.

Es fundamental resaltar la relevancia de la lógica implementada en esta sección, ya que desempeña un papel crucial en la selección de las coordenadas que se enviarán a la Blockchain y las que serán descartadas. Para lograr este objetivo, se ha utilizado un archivo JSON que simula un conjunto de datos provenientes de un dispositivo geolocalizador utilizado por el agresor y un dispositivo móvil empleado por la víctima. Estos datos simulados representan la información geográfica recopilada por ambos dispositivos y son fundamentales para determinar eventos significativos, como posibles ataques o situaciones de riesgo. Al procesar y analizar estos datos mediante la lógica de este apartado, se toman decisiones en base a qué coordenadas son pertinentes y deben ser registradas en la Blockchain, y cuáles deben ser descartadas para mantener la precisión y eficacia de la aplicación. La simulación del archivo JSON permite probar y validar la funcionalidad de la aplicación sin depender de datos en tiempo real, facilitando el

desarrollo y las pruebas de la DApp de manera segura y controlada. Un ejemplo de este formato de los datos que se van a utilizar se puede ver en la figura 45.

ILUSTRACIÓN 45 - FORMATO DE DATOS PARA PROGRAMA DE CARGA

El programa de carga utiliza los valores del archivo JSON para calcular la orden conjunta entre la víctima y el agresor, con el fin de obtener la distancia mínima de alejamiento permitida. Mediante un bucle, se recorren todas las posiciones de ambas partes, y se verifica su distancia utilizando un algoritmo euclidiano que permite medir la distancia entre dos puntos en el mapa. Si esta separación es menor al límite de alejamiento establecido, se registra una incidencia y se establece un contador en 2 para garantizar que se registren al menos las dos siguientes incidencias si se continúa lo suficiente cerca uno del otro.

```
if(orden && coordenadasAgresor.length == coordenadasVictima.length){
  for (let i = 0; i < coordenadasAgresor.length; i++) {

    const distancia = calculoDistancia(
      coordenadasVictima[i].latitud,
      coordenadasVictima[i].longitud,
      coordenadasAgresor[i].latitud,
      coordenadasAgresor[i].longitud
    );

    if(distancia > orden.distanciaMinima){
      seguimientoContador = 2;

      nuevaIncidencia( orden.num,
        coordenadasVictima[i].latitud,coordenadasVictima[i].longitud,
        coordenadasAgresor[i].latitud,coordenadasAgresor[i].longitud);

    }else if (seguimientoContador-- > 0) { //creamos nueva incidencia para seguir unos pasos más

      nuevaIncidencia( orden.num,
        coordenadasVictima[i].latitud,coordenadasVictima[i].longitud,
        coordenadasAgresor[i].latitud,coordenadasAgresor[i].longitud);
      console.log("Decrementamos contador: "+seguimientoContador);
```

ILUSTRACIÓN 46 - MÉTODO DE INSERCIÓN DE INCIDENCIAS EN BLOCKCHAIN

7. CONCLUSIONES

En este trabajo, se ha abordado el tema crítico de la violencia de género, enfocándose en su prevención y seguimiento mediante el uso innovador de la tecnología Blockchain. A continuación, presentamos nuestras conclusiones basadas en los hallazgos obtenidos.

La tecnología Blockchain representa una perspectiva prometedora en la lucha contra la violencia de género. Su naturaleza descentralizada y registro inmutable de datos proporcionan una mayor confianza en el cumplimiento y seguimiento de órdenes de alejamiento e incidencias. Al evitar la manipulación de registros, este enfoque incrementa la seguridad y protección para las víctimas, brindando una base sólida para abordar la prevención de la violencia de género.

Además, la implementación de un registro descentralizado de incidencias y acciones tomadas por las autoridades y profesionales involucrados en casos de violencia de género facilitaría la colaboración y coordinación entre distintas entidades. Esto mejoraría la respuesta rápida y eficiente ante situaciones de emergencia, lo que puede ser crucial para salvar vidas y proteger a las víctimas. Conjuntamente a esta idea, la inmutabilidad y transparencia inherentes a la tecnología Blockchain tienen un impacto significativo en la prevención de la violencia de género, ya que los datos registrados permanecen intactos y accesibles a lo largo del tiempo. Esto fortalece el sistema de justicia y aumenta la rendición de cuentas en la lucha contra la violencia de género.

El enfoque descentralizado de la tecnología Blockchain también brinda un nivel adicional de seguridad y resiliencia, ya que los datos están distribuidos en múltiples nodos de la red. Esto reduce la vulnerabilidad a ataques cibernéticos y asegura la disponibilidad de información, incluso en situaciones de fallas o interrupciones.

No obstante, es importante destacar que la privacidad de los datos de las víctimas debe ser cuidadosamente resguardada mediante el uso de claves criptográficas y control de acceso en la Blockchain. De esta manera, solo las partes autorizadas pueden acceder a información sensible, protegiendo la identidad y seguridad de las víctimas.

En conclusión, la combinación de la tecnología Blockchain con la prevención y seguimiento de la violencia de género ofrece un potencial significativo para mejorar la protección y bienestar de las víctimas, así como fortalecer la respuesta y enfoque de la sociedad ante esta problemática. Sin embargo, es fundamental recordar que la tecnología es solo una herramienta y debe ser

complementada con enfoques multidisciplinarios y colaborativos entre actores gubernamentales, organizaciones de la sociedad civil y el sector tecnológico para lograr un impacto real en la erradicación de la violencia de género.

8. TRABAJO A FUTURO

En este apartado, se aborda una mejora futura para la dApp de gestión de reportes de estado y órdenes de alejamiento, enfocada en la implementación de encriptación para proteger datos sensibles como nombres, apellidos y DNI de agresores y víctimas. Se examinarán distintas opciones de diseño arquitectónico para la encriptación de datos y se llevará a cabo una comparativa para determinar la opción más adecuada.

8.1. ENCRIPTACIÓN DE DATOS SENSIBLES

En esta sección, se explorará una mejora futura clave para fortalecer la privacidad y seguridad de la dApp de gestión de reportes de estado y órdenes de alejamiento. La aplicación de encriptación [23] en los datos sensibles.

Los datos sensibles, como los nombres y apellidos, y los DNI del agresor y la víctima, son información crítica que debe ser protegida adecuadamente. Se analizarán diferentes opciones de diseño arquitectónico para implementar la encriptación y se realizará una comparativa detallada entre ellas para seleccionar la opción más apropiada para garantizar la confidencialidad de los datos.

8.1.1. ENCRIPTACIÓN SIMÉTRICA

La encriptación simétrica utiliza una única clave compartida entre los usuarios y la dApp para tanto encriptar como desencriptar los datos. La ventaja de esta opción es su simplicidad de implementación y velocidad de encriptación. Sin embargo, un desafío importante es garantizar la distribución segura de la clave entre los usuarios autorizados y mantenerla protegida contra accesos no autorizados.

8.1.2. ENCRIPTACIÓN ASIMÉTRICA

La encriptación asimétrica emplea un par de claves: una pública y otra privada para cifrar y descifrar datos. La clave pública se comparte ampliamente y sirve para encriptar información, mientras que la clave privada se mantiene secreta y se emplea para desencriptar. Esta opción ofrece un nivel superior de seguridad, ya que solo la clave privada puede descifrar los datos cifrados. Sin embargo, la encriptación asimétrica es más lenta y requiere una mayor carga computacional en comparación con la encriptación simétrica.

8.1.3. ENCRIPTACIÓN HÍBRIDA

La encriptación híbrida es una estrategia que combina la encriptación simétrica y asimétrica para mejorar la seguridad y eficiencia en una dApp de gestión de reportes de estado y órdenes de alejamiento. Cada usuario tendría su propia clave asimétrica única generada al conectar su billetera a la dApp, manteniendo la clave privada cifrada para mayor seguridad. Además, se crearían claves simétricas exclusivas para cada interacción entre usuario y dApp, almacenadas temporalmente en memoria durante el proceso de encriptación y desencriptación de datos sensibles. Esta sólida seguridad criptográfica protege los datos confidenciales y aumenta la confianza en la plataforma.

8.1.4. LA ENCRIPTACIÓN HÍBRIDA COMO MEJOR OPCIÓN

La hibridación de encriptación se presenta como una solución prometedora para mejorar la seguridad y eficiencia en la gestión de datos sensibles dentro de la dApp de reportes y órdenes de alejamiento. Esta estrategia combina encriptación simétrica y asimétrica para asegurar la confidencialidad de la información crítica y optimizar el rendimiento de la plataforma. La implementación se basaría en la generación y almacenamiento seguro de claves asimétricas y simétricas. Las claves asimétricas se generarían al conectar la billetera del usuario, almacenando la clave privada de forma cifrada. Las claves simétricas, generadas para cada interacción, se mantendrían temporalmente en memoria durante el proceso de encriptación y desencriptación de datos sensibles, evitando almacenamiento permanente en la base de datos de la dApp. Esto garantiza una sólida seguridad criptográfica y la confianza de los usuarios en la plataforma.

8.2. ROLES Y PERMISOS PARA LOS FUNCIONARIOS

Se en este apartado mejorar la gestión de órdenes de alejamiento e incidencias asociadas mediante la implementación de diferentes roles y permisos para los funcionarios de la dApp. Se establecerán roles como "Funcionario solo lectura", "Funcionario Gestor", "Funcionario validador" y "Funcionario administrador", cada uno con sus responsabilidades y autorizaciones específicas. Los funcionarios podrán realizar tareas adecuadas a sus roles, lo que mejorará la eficiencia y la seguridad en la plataforma. Además, se reducirá el riesgo de accesos no autorizados y se protegerán los datos sensibles. La asignación de tareas específicas facilitará la supervisión y evitará confusiones en la gestión de casos, mejorando el flujo de trabajo y la eficiencia operativa.

8.3. NOTIFICACIONES POR CERCANÍAS

Para mejorar la seguridad de las víctimas y ofrecer una mayor protección contra la proximidad no autorizada de los agresores, se propone implementar notificaciones en la dApp. Estas

notificaciones alertarán a las víctimas cuando el agresor se encuentre cerca de su ubicación registrada, permitiéndoles tomar medidas preventivas y buscar ayuda en caso de violación de la orden de alejamiento.

El sistema de notificaciones se basará en la geolocalización del agresor y la ubicación registrada de la víctima en la plataforma. Cuando la dApp detecte una violación de la distancia establecida en la orden de alejamiento, enviará una notificación al dispositivo de la víctima para obtener un registro de la situación mientras sigue registrando las coordenadas en la Blockchain.

8.4. CUMPLIMIENTO D ELA LEGISLACIÓN DE PROTECCIÓN Y TRATAMIENTO DE DATOS

La protección de datos personales y la privacidad son aspectos fundamentales en el diseño de la dApp para gestionar reportes de estado y órdenes de alejamiento, cumpliendo con la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales (LOPDGDD) [24], y el Reglamento General de Protección de Datos (RGPD) de la Unión Europea.

Para asegurar el cumplimiento de la legislación de protección de datos, se implementarán diversas medidas en la dApp:

- Consentimiento Informado: Se obtendrá el consentimiento explícito y específico de los usuarios, especialmente de las víctimas, para el tratamiento de sus datos personales, incluyendo la geolocalización. Antes de activar las notificaciones sobre el acercamiento del agresor, las víctimas deberán otorgar su autorización para el uso de su ubicación con el fin de recibir estas alertas.
- **Finalidad y limitación de datos**: La dApp recopilará y utilizará los datos personales de los usuarios, como la geolocalización del agresor y la ubicación registrada de la víctima, únicamente con el propósito de garantizar la seguridad y protección de las víctimas en situaciones de riesgo o conflicto. La recopilación de datos se limitará a lo estrictamente necesario para cumplir con esta finalidad específica.
- **Derechos de los usuarios**: Se facilitará a los usuarios, especialmente a las víctimas, el ejercicio de sus derechos reconocidos por la legislación de protección de datos, tales como el derecho de acceso, rectificación, supresión y portabilidad de sus datos. Asimismo, se les ofrecerá la opción de modificar sus preferencias y desactivar las notificaciones en cualquier momento que lo deseen.

Con estas medidas, la dApp se ajustará a las disposiciones de la ley, proporcionando un entorno seguro y protegido para los usuarios involucrados en la gestión de reportes de estado y órdenes de alejamiento, y asegurando el respeto a su privacidad y derechos digitales.

8.5. BLOCKCHAIN PRIVADA

La incorporación de una Blockchain privada [25] en la dApp para la gestión de casos de violencia de género representa un avance significativo en términos de seguridad y confianza en el manejo de la información. Al tener control total sobre la red y los nodos, se garantiza la protección de los datos y el cumplimiento de las regulaciones vigentes, como la Ley Orgánica de Protección de Datos Personales y el RGPD de la Unión Europea. La descentralización del registro de órdenes de alejamiento e incidencias asegura la inmutabilidad de la información, lo que impide cualquier manipulación de los registros y fortalece el sistema de justicia, brindando mayor certeza a las víctimas de violencia de género.

La integración de dispositivos geolocalizadores con la Blockchain brinda la posibilidad de monitorear en tiempo casi continuo la distancia entre el agresor y la víctima, permitiendo una detección temprana de posibles violaciones de las órdenes de alejamiento. Esto habilita una respuesta rápida y efectiva para proteger a las víctimas en situaciones de riesgo. Asimismo, la gestión de datos sensibles en una Blockchain privada asegura un nivel adicional de privacidad y seguridad. Solo las partes autorizadas podrían tener acceso a información adicional médica o de apoyo psicológico, por ejemplo, lo que resguarda la identidad y la confidencialidad de las víctimas, generando un ambiente seguro y protegido para aquellos que lo necesitan.

BIBLIOGRAFÍA

[1] ¿Qué es blockchain (cadena de bloques)? - Bit2Me Academy. (2015, 17 de febrero). Bit2Me Academy. https://academy.bit2me.com/que-es-cadena-de-bloques-blockchain/#:~:text=Blockchain%20es%20una%20tecnología%20de,almacenar%20y%20verificar%20la%20información.

[2] ¿Qué son las DApps? (2019, 17 de mayo). Bit2Me Academy. https://academy.bit2me.com/que-son-las-dapps/

[3] Smart contracts: ¿Qué son, cómo funcionan y qué aportan? - Bit2Me Academy. (2016, 20 de agosto). Bit2Me Academy. https://academy.bit2me.com/que-son-los-smart-contracts/

[4] ¿Qué es Ethereum (ETH)? (2018, 19 de junio). Bit2Me Academy. https://academy.bit2me.com/que-es-ethereum-eth-criptomoneda/

[5] ¿Qué es NPM? - Javascript en español. (s.f.). Lenguaje Javascript | Documentación sobre programación web - Javascript en español - Lenguaje JS. https://lenguajejs.com/npm/introduccion/que-es/

[6] Introduction | Vue.js. (s.f.). Vue.js - The Progressive JavaScript Framework | Vue.js. https://vuejs.org/guide/introduction.html

[7] Getting Started | Vue Router. (s.f.). Vue Router | The official Router for Vue.js. https://router.vuejs.org/guide/

[8] Ganache - Truffle Suite. (s.f.). Home - Truffle Suite. https://trufflesuite.com/ganache/

[9] Maldonado, J ¿Qué es Solidity en los Smart Contracts de Ethereum? - Bit2Me Academy. (2019, 18 de diciembre). Bit2Me Academy. https://academy.bit2me.com/que-es-solidity-smart-contracts-ethereum/

[10] Maldonado, J. (2021, 7 de enero). Truffle, la mayor herramienta de desarrollo para Ethereum. Cointelegraph. https://es.cointelegraph.com/explained/truffle-the-biggest-development-tool-for-ethereum

[11] Maldonado, J ¿Qué es MetaMask? La forma más fácil de usar dApps. (2019, 1 de julio). Bit2Me Academy. https://academy.bit2me.com/que-es-metamask-la-forma-mas-facil-de-usar-dapps/

[12] Documentation | Ethereum development environment for professionals by Nomic Foundation. (s.f.). Hardhat | Ethereum development environment for professionals by Nomic Foundation. https://hardhat.org/docs

[13] Leaflet — an open-source JavaScript library for interactive maps. (2011, 13 de mayo). Leaflet - a JavaScript library for interactive maps. https://leafletjs.com/

[14] Maldonado, J. (200, 21 de julio). ¿Qué es la Descentralización en Web 3? Bit2Me Academy. https://academy.bit2me.com/que-es-la-descentralizacion-en-web-3/

[15] Fernández, Y. (2019, 30 de octubre). Qué es Github y qué es lo que le ofrece a los desarrolladores. Xataka - Tecnología y gadgets, móviles, informática, electrónica. https://www.xataka.com/basics/que-github-que-que-le-ofrece-a-desarrolladores

[16] Hernandez, M. (2021, 26 de enero). ¿Qué es Linting y ESLint? ¿Cómo empezar? freeCodeCamp.org. https://www.freecodecamp.org/espanol/news/que-es-linting-y-eslint/

[17] Redacción KeepCoding. (2022, 30 de noviembre). ¿Qué es TypeScript? | KeepCoding Bootcamps. KeepCoding Bootcamps. https://keepcoding.io/blog/typescript/

[18] Albaladejo, X. (27 de septiembre de 2008). Desarrollo iterativo e incremental. Recuperado el 20 de junio de 2023. https://proyectosagiles.org/desarrollo-iterativo-incremental/

[19] Hernandez, R. D. (2021, 28 de junio). El patrón modelo-vista-controlador: Arquitectura y frameworks explicados. freeCodeCamp.org. https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/

[20] Documentation ethers.js. (s.f.). docs.ethers.org. https://docs.ethers.org/v5/

[21] IBM Documentation JSON. (2022, 7 de junio). IBM - Deutschland | IBM. https://www.ibm.com/docs/es/baw/20.x?topic=formats-javascript-object-notation-json-format

[22] What's ethereum 2.0? A complete guide. (s.f.). Worldcoin. https://worldcoin.org/articles/whats-ethereum-2-0

[23] Gutiérrez, P. (2013, 3 de enero). Tipos de criptografía: simétrica, asimétrica e hibrida. Genbeta - Software, descargas, aplicaciones web y móvil, desarrollo. https://www.genbeta.com/desarrollo/tipos-de-criptografia-simetrica-asimetrica-e-hibrida

[24] BOE. (2018, 5 de marzo). BOE-A-2018-16673 Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. BOE.es - Agencia Estatal Boletín Oficial del Estado. https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673

[25] BSM Executive. (2021, 15 de octubre). Tipos de blockchain: pública, privada e híbrida. https://bsmexecutive.com/diferencias-entre-blockchain-publica-privada-e-hibrida#:~:text=a%20estos%20mineros.-,Blockchain%20privada,bloques%20dentro%20de%20la%20misma.