Gestión de Alerta de Incidentes

Integrantes:

- Nayeli Belen Guerrero Gutierrez
- Julio Cesar Portugal Vilca
- Carlos Renato Guerra Marquina
- Yeimi Adelmar Varela Villarreal

1. Introducción

Objetivo del proyecto

Desarrollar una plataforma web y aplicativo moderno para la gestión de alertas de incidentes y accidentes en entornos laborales o comunitarios. Este proyecto está dirigido al público en general pues, esta solución busca mejorar la seguridad y la capacidad de respuesta frente a emergencias, permitiendo registrar y monitorear incidentes de manera activa.

Contexto

En el contexto peruano, la gestión de incidentes sigue siendo informal y dispersa, lo que genera demoras en la atención y resolución de emergencias. Este proyecto se desarrollará en un contexto comunitario, donde vecinos, líderes, responsables de seguridad y el público en general podrán reportar situaciones que afecten la integridad de las personas o el entorno; con el objetivo de atender necesidades reales de coordinación, registro histórico, y generación de estadísticas, se busca implementar una plataforma que modernice los procesos y fomente comunidades más seguras y conectadas.

2. Features a implementar

Descripción de funcionalidades

- *Registro de usuario*: Autenticación segura para el registro de los datos del usuario (sólo será la primera vez que inicie sesión).

- *Formulario de reporte de incidentes*: Captura de datos como tipo, ubicación, prioridad, evidencia (fotos/videos). Esto puede variar dependiendo de la prioridad, pues se habilitará un registro de opción rápida para aquellos incidentes/accidentes considerados “urgentes”.

- *Dashboard de seguimiento*: Visualización interactiva de incidentes activos, resueltos y métricas generales. Deberá quedar un registro histórico en la plataforma. Adicionalmente, el usuario puede escoger si desea visualizar la información por tipo (incendio, choque, robo, etc), por zona o por gravedad (incidente, accidente o emergencia).

- *Notificaciones*: Alertas automáticas vía SMS para alertar a los usuarios que se encuentren cerca de alguna emergencia.

- *Geolocalización*: Integración con Google Maps API para visualizar ubicación exacta del incidente y de los distintos usuarios.

- *Historial de cambios*: Trazabilidad del cambio de estatus de los incidentes.

Prioridades y Fases de Implementación

Fase	Descripción
Backend - MVP	En la primera fase se desarrollará el backend básico que permita cubrir el flujo principal de la plataforma. Se implementarán los servicios de autenticación de usuarios para registrar y validar el acceso de las personas que usarán la plataforma. Asimismo, se desarrollarán las APIs necesarias para permitir el registro de incidentes, incluyendo la captura de datos como tipo de incidente, ubicación, prioridad y evidencia. Además, se integrará de forma inicial la API de Google Maps para almacenar coordenadas asociadas a los reportes. Se diseñará y estructurará la base de datos para asegurar la persistencia adecuada de todos los datos. Esta fase garantizará un backend funcional que permita la creación y consulta básica de incidentes.
Backend - Expandido	En la segunda fase se ampliarán las funcionalidades del backend para cubrir características más específicas. Se implementarán los servicios de notificaciones automáticas vía SMS mediante la integración de un proveedor externo como Twilio. También se habilitará la subida y almacenamiento seguro de archivos multimedia (fotos y videos) que acompañen los reportes de incidentes. Se desarrollará un endpoint especial de registro rápido para emergencias, que priorice la velocidad del reporte. Asimismo, se trabajará en el sistema de historial de cambios para rastrear cualquier modificación en los incidentes registrados y en las APIs de filtros avanzados que permitan al usuario consultar incidentes por tipo, gravedad o zona geográfica. Finalmente, se implementarán endpoints para generar estadísticas y métricas generales de los incidentes reportados.
Fronted - MVP	El desarrollo del frontend básico permitirá la interacción del usuario con el sistema. Se diseñarán las pantallas de registro e inicio de sesión para que los usuarios puedan autenticarse de manera segura. También se construirá un formulario de reporte de incidentes que capture todos los datos esenciales. Se desarrollará un dashboard sencillo que permita a los usuarios visualizar los incidentes activos y no activos, junto con una vista de mapa que muestre la ubicación de cada incidente utilizando Google Maps. En esta etapa, se priorizará la conexión con las APIs existentes para asegurar que todas las operaciones principales puedan ser realizadas desde la interfaz de usuario.
	Se integrarán los filtros avanzados para que los usuarios puedan buscar y visualizar incidentes de manera más

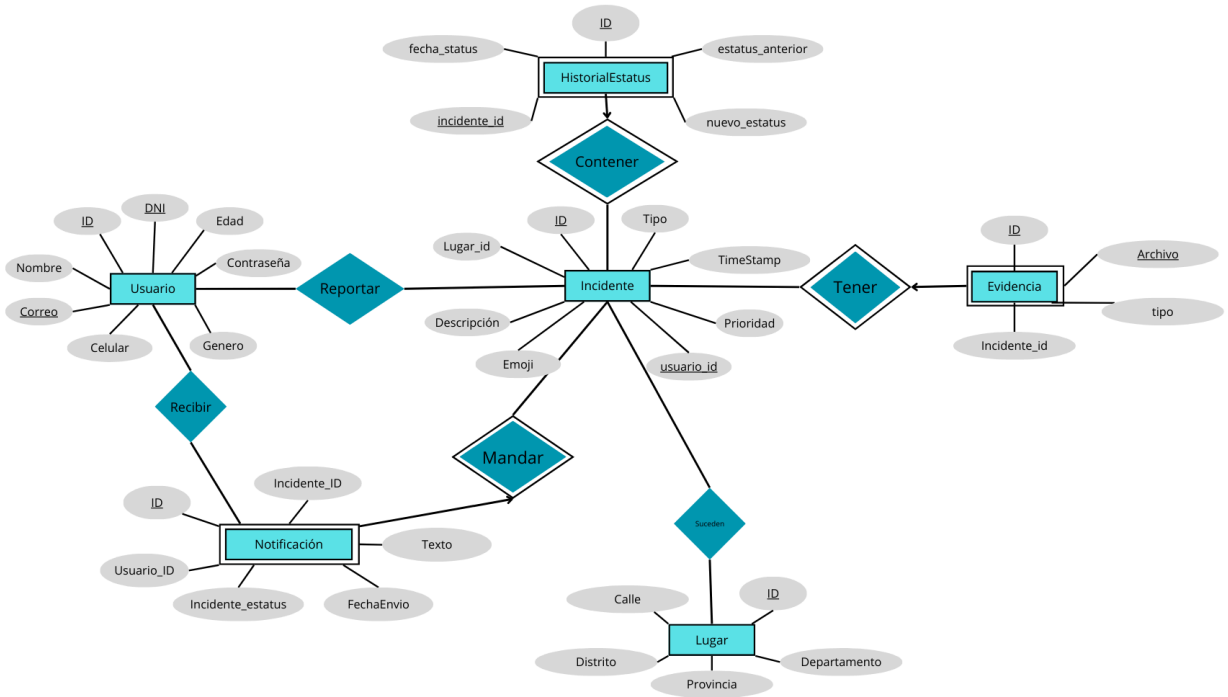
Fronted - Expandido	precisa. Se habilitará la visualización de los historiales de cambios asociados a cada incidente, permitiendo ver su evolución. Se implementará la visualización de evidencia multimedia (fotos y videos) en los detalles de los reportes. Asimismo, se conectará la funcionalidad de notificaciones mediante alertas SMS y mensajes en la plataforma. Por último, se diseñarán gráficas dinámicas que muestren estadísticas relevantes sobre la actividad de incidentes en la comunidad.

3. **Entidades**

Modelo Relacional

- **Usuario**: id, dni, nombre, correo, contraseña, edad, género, celular
- **Incidente**: id, Tipo, descripcion, prioridad, estatus, usuario_id, emoji, TimeStamp, lugar_id
- **Evidencia**: id, archivo, incidente_id, tipo
- **Lugar**: id, departamento, provincia, distrito, calle_jiron
- **HistorialEstatus**: id, estatus_anterior, nuevo_estatus, fecha_status, incidente_id.
- **Notificación:** id, usuario_id, incidente_id, estatus_actual, texto, fecha_envio

Explicación de la relación entre entidades (MER)



4. **Manejo de errores y testing**

Manejo de errores:

- Formato correcto del correo electrónico (`correo`)
- Validación del DNI (mediante verificación facial y comparación de datos de `nombre`, y número de DNI `dni`)
- Conflictos de datos (409 Conflict), como usuarios duplicados.
- Validación de formatos y límite de tamaño para las evidencias (.JPG, .JPEG, .PNG, .HEIC, .MP4, .MOV, etc)
- Fallo en el guardado o lectura del archivo de `evidencia`
- Validación del número de celular (mediante códigos de validación)
- Contraseñas seguras
- Rellenar Campos obligatorios al realizar un reporte
- Validación de fechas (para que no existan reportes con fechas futuras)
- Validación de la entidad `Lugar` (Solo para áreas geográficas dentro del Perú)
- Manejo de errores cuando la API de mapas no responde (Se puede alternar entre Google Maps y Mapbox)
- Límite de notificaciones por usuario para evitar spam de SMS's

Testing

- Pruebas unitarias para cada entidad (Usuario, Incidente, Lugar, etc.)
- Validación de relaciones entre entidades
- Control de acceso entre usuarios (no poder modificar el incidente de otros)
- Protección de datos personales como DNI y celular
- Integración con los servicios de terceros (API's externas)
- Carga y visualización de elementos multimedia de las evidencias
- Transiciones de estado en el historial (HistorialEstatus)
- Permisos concedidos de GPS, Notificaciones, etc…, por los usuarios
- Cambio de contraseña para cuentas de usuario
- Pruebas E2E: Postman
- Pruebas de Usabilidad: Feedback de usuarios sobre la interfaz React.
- Pruebas Unitarias: JUnit + Mockito para clases de servicio.
- Pruebas de Integración: Spring Boot Test para verificar relaciones entre capas.

5. Eventos y asincronía

Eventos

- Clic en "Registrar usuario" : Envía el formulario de registro al servidor y un correo de confirmación una vez terminado.
- Clic en "Iniciar sesión" : Envía credenciales al servidor para autenticar.
- Clic en "Reportar incidente" : Abre el formulario de reporte y selecciona prioridad (urgente/no urgente).
- Envío del formulario de reporte de incidente: Captura y envía la información al servidor.
- Selección de tipo de incidente (incendio, robo, etc.): Para marcar.
- Clic en "Ver detalles de incidente" en el dashboard : Abre información detallada.

- Recepción de notificación de emergencia: Envía un correo electrónico confirmando el registro del incidente.
- Cambio de estado de un incidente (activo/resuelto) : Actualiza el historial.
- Movimiento o desplazamiento en el mapa : Actualiza la ubicación mostrada.
- Cierre de sesión: Finaliza la sesión.

Asincronía

- Envío de datos de registro/inicio de sesión: Solicitud HTTP al servidor (autenticación).
- Envío del reporte de incidente: Guardar en la base de datos vía API.
- Carga de incidentes en el dashboard : Llamadas a la API para traer datos actualizados.
- Envío y recepción de notificaciones automáticas: Integración con servicios externos tipo Twilio.
- Petición a Google Maps API: Obtener mapas y geolocalización en tiempo real.
- Actualización del estado de un incidente: Solicitud API para modificar el incidente.
- Carga del historial de cambios de un incidente: Petición a la base de datos para traer la trazabilidad.
- Subida de evidencias (fotos/videos): Carga de archivos al servidor.
- Recepción de alertas para usuarios cercanos: Notificar mediante SMS a usuarios cercanos a algún incidente.

6. Integración con servicio de tercero

Mencionar el uso de API's y plataformas externas de ser necesario

- Google Maps API / Mapbox**:** Para geolocalización de incidentes y usuarios y mostrar zonas de riesgo.
- API de Reniec/ Onfido**:** para validar datos del DNI.