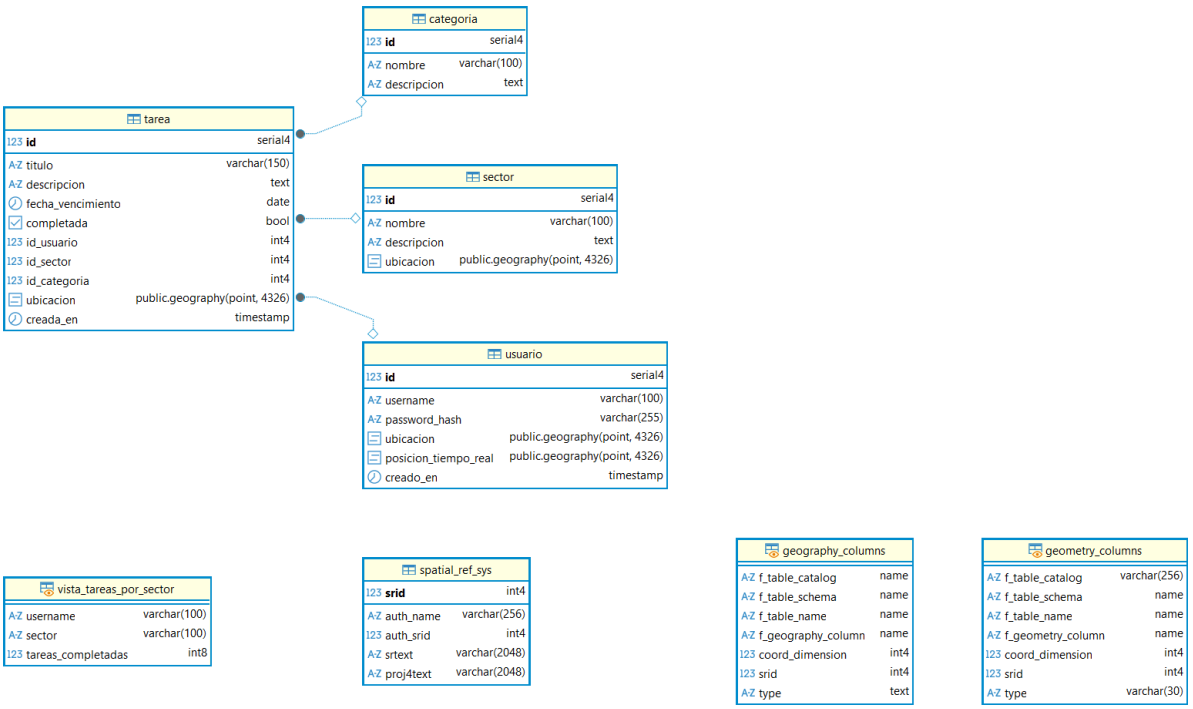
Documentación
Definición del proyecto

Este proyecto presenta una solución con el objetivo de permitir a los usuarios realizar un seguimiento de sus tareas (pendientes y terminadas). Además, el sistema permite la creación, edición y eliminación de tareas.
Asimismo, el sistema integra mecanismos que permiten la realización de una gran cantidad de consultas, tales como obtener promedios, encontrar la tarea más cercana, obtener el número de tareas pendientes, entre otros.
Por último, la aplicación incorpora autenticación JWT con el objetivo de prevenir ataques de inyección SQL y CSRF.

Tecnologías utilizadas

Capa	Tecnología
Frontend	Vue.js
Backend	Java
Seguridad	Spring Security y JWT
Base de datos	Postgresql 15
Extensión espacial	PostGIS

Diagrama



Modelo de datos

Entidad usuario: Se almacenan la información de los usuarios. Su objetivo es validar las credenciales de login en la base de datos.

Atributos	Tipo de dato
id	Serial
username	varchar(100)
password_hash	varchar(255)
ubicacion	public.geography(point, 4326)
creado_en	timestamp

Entidad sector: Corresponde al sector geográfico donde se encuentran las tareas.

Atributos	Tipo de dato
id	Serial

nombre	varchar(100)
descripcion	text
ubicacion	public.geography(point, 4326)

Entidad tarea: Corresponde a las actividades que puede realizar el usuario. Cada tarea esta vinculada a un usuario y a un sector geográfico.

Atributos	Tipo de dato
id	Serial
titulo	varchar(150)
descripcion	text
fecha_vencimiento	date
completada	bool
id_usuario	int
id_sector	int
ubicacion	public.geography(point, 4326)
creada_en	timestamp

Entidad categoría: Corresponde a las categorías en las que se engloban las distintas tareas. Estas categorías representan oficios generales, como mantenimiento ,instalación, etc.

Atributos	Tipo de dato
id	Serial
nombre	varchar(100)
descripcion	text

Funcionalidades

1. Registro de usuarios

Se implementó una funcionalidad que permite que los usuarios se registren con un nombre de usuario, contraseña y elijan su ubicación de registro en un mapa.
La contraseña sigue un cifrado Bcrypt y la ubicación es transformada a un objeto geoespacial point.

2. Gestión de tareas

Este módulo implementa un CRUD robusto que vincula cada tarea con una **ubicación espacial (SRID 4326)** y un usuario específico mediante **Spring Security**. Al crear o actualizar una tarea, el sistema utiliza la biblioteca **JTS** para convertir la latitud y longitud en un objeto `Point`, permitiendo un almacenamiento georeferenciado preciso. El controlador asegura la integridad de los datos validando que solo el propietario de la tarea (identificado por su token **JWT**) pueda modificarla o eliminarla.

3. Filtros y búsqueda

El sistema incorpora funcionalidades de filtrado y búsqueda de tareas por identificador, estado y prioridad.

4. Notificaciones

Se implementó una funcionalidad de notificaciones en la aplicación representada por un icono de una campana en la interfaz de usuario. Esta funcionalidad permite alertar al usuario cuando una tarea está próxima a vencer (2 días).
El funcionamiento de este módulo se implementó mediante la utilización de una consulta a la base de datos mediante la utilización de JDBC. El controlador recibe el token de autenticación (JWT) lo decodifica, obtiene el identificador del usuario, y lo utiliza para llamar al repositorio encargado de realizar la consulta.
Entradas: identificador del usuario
Salidas: lista de notificaciones con fecha de vencimiento próxima (2 días)

5. Asociación

Se implementó un módulo que permite asociar un sector a cada tarea. Para su implementación se creó una nueva tabla. Encargada de gestionar y visualizar dicha asociación mediante categorías generales para cada tipo de tarea u oficio. Esta tabla lleva el nombre categoría.

6. Preguntas a responder

1. ¿Cuántas tareas ha hecho el usuario por sector?

   Este endpoint devuelve la cantidad de tareas terminadas que ha realizado el usuario.
   URL: `/api/numerotareas/cantidadtareas`
   Método: `GET`
   Descripción: Se implementó una consulta que permite obtener las cantidad de tareas terminadas agrupadas por sector.
   Datos entrada: identificador de usuario.
   Datos salida: lista de tareas terminadas agrupadas por sector.

2. ¿Cuál es la tarea más cercana al usuario (que esté pendiente)?

   Este endpoint devuelve la tarea más cercana a la ubicación del usuario
   URL: `/api/tarea/mascercana`
   Método: `GET`
   Descripción: ordena las tareas que están pendientes de menor a mayor distancia y devuelve la más cercana.
   Datos entrada: identificador de usuario
   Datos salida: tarea más cercana

3. ¿Cuál es el sector con más tareas completadas en un radio de 2 kilómetros del usuario?

   Este endpoint devuelve la media aritmética de la distancia entre la posición actual d el usuario y sus tareas finalizadas.

   URL: `/api/promedios/distancia-tareas-completadas`

   Método: `GET`

   Descripción: Calcula qué tan lejos suele estar el usuario de los puntos donde completa sus tareas.

4. ¿Cuál es el promedio de distancia de las tareas completadas respecto a la ubicación del usuario?

   URL: `/api/sectores/tareas-completadas/radio-2km`
   Método: `GET`
   Descripción: Retorna una lista de sectores geográficos, ordenados por volumen de actividad, donde el usuario ha finalizado trabajos dentro de un perímetro de 2 kilómetros.

5. ¿En qué sectores geográficos se concentran la mayoría de las tareas pendientes? (utilizando agrupación espacial).

   URL: `/api/tarea/pendientes-por-sector`
   Método: `GET`
   Descripción: Retorna una lista de sectores geográficos, en conjunto con el número de tareas que hay en esa zona, se despliega una lista de las tareas.

6. ¿Cuántas tareas ha realizado cada usuario por sector?

   URL: `/api/tarea/reportes/tareas-por-usuario-sector`
   Método: `GET`
   Descripción: Obtiene la cantidad de tareas realizadas para cada usuario y en cada sector.

7. ¿Cuál es el sector con más tareas completadas dentro de un radio de 5 km desde la ubicación del usuario?

   URL: `/api/tarea/reportes/sector-top-5km`
   Método: `GET`
   Descripción: Obtiene el sector con mayor cantidad de tareas realizadas por el usuario logueado.

8. ¿Cuál es el promedio de distancia entre las tareas completadas y el punto registrado del usuario?

   URL: `/api/promedioubicacion/tarea/registro`
   Método: `GET`
   Descripción: Obtiene el promedio de distancia entre todas las tareas completadas con respecto a la posición en tiempo real del usuario.

Funcionamiento

En el frontend:
opción 1: utilizar el usuario username: usuario testeo y contraseña: prueba123
opción 2: se debe crear un usuario con un correo electrónico y una contraseña (de los usuarios que están cuando se crea la base de datos solo uno se puede utilizar). Luego se debe loguear con el correo y la contraseña, de esta forma se podrán ver las consultas.
Si elige la opción 2, para que funcione varias consultas que utilizan la posición en tiempo real se debe elegir otra ubicación en el mapa de ubicación registrada.
Para que funcione la funcionalidad de la notificación se debe crear una tarea por vencer en una fecha < 2 días.
Nota: para que funcione las contraseñas y el login el jwt.secret debe tener 32 caracteres o más.
En el backend:
utilizando .jar se debe ejecutar el siguiente comando en la terminal java -jar target/Gestor_de_tareas.jar