UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

TALLER DE BASE DE DATOS DIURNO 2-2025
Enunciado 1

Ayudante: Pablo Macuada **Fernando Solis**	**Profesor:** Matías Calderón

Enunciado Laboratorio 1

Entrega: 20 octubre de 2025

Proyecto:

Mapa Colaborativo de Sitios Turísticos

Objetivo: Desarrollar una red social interactiva donde los usuarios puedan descubrir, compartir y calificar sitios de interés turístico. La plataforma debe integrar un mapa que muestre la ubicación de los lugares, permitiendo a la comunidad colaborar en la creación de un recurso de viaje dinámico y actualizado.

Tecnologías y Herramientas Requeridas

- **Base de Datos:** PostgreSQL
- **Backend:** Spring Boot con Java
- **Frontend:** Vue.js 3
- **Comunicación:** Axios para las llamadas HTTP
- **Seguridad:** JSON Web Tokens (JWT) para la autenticación y autorización
- **Control de Versiones:** Git y un repositorio en GitHub

Requisitos Específicos

1. Requisitos de la Base de Datos (PostgreSQL)

Deberás diseñar un **esquema de base de datos** normalizado que incluya, al menos, las siguientes tablas:

- **usuarios**: Información de los usuarios de la red social (nombre, email, contraseña hasheada, biografía).
- **sitios_turisticos**: Información de los lugares de interés (ID, nombre, descripción, tipo - p. ej., 'Parque', 'Museo', 'Restaurante', coordenadas de punto).
- **reseñas**: Reseñas y calificaciones de los usuarios sobre los sitios (ID, contenido, calificación, fecha, ID de usuario, ID de sitio).
- **fotografias**: Enlaces a las fotos subidas por los usuarios de los sitios turísticos (ID, URL, fecha, ID de usuario, ID de sitio).
- **listas_personalizadas**: Listas de lugares creadas por los usuarios (p. ej., 'Lugares para visitar en 2024').

No se permite el uso de JPA/Hibernate. La comunicación entre la aplicación y la base de datos debe ser exclusivamente a través de **sentencias SQL nativas**.

Deberás implementar los siguientes elementos en la base de datos:

- **Triggers:** Para automatizar procesos como la actualización de la calificación promedio de un sitio cuando se agrega una nueva reseña.
- **Procedimientos Almacenados:** Para encapsular la lógica de negocio compleja, como la búsqueda de sitios cercanos o la generación de perfiles de usuario.
- **Vistas Materializadas:** Para pre-calcular resúmenes y estadísticas que se consultan con frecuencia, mejorando el rendimiento de la aplicación.
- **Índices:** Para optimizar las consultas más comunes, especialmente las búsquedas por ubicación y atributos.

2. Requisitos del Backend (Spring Boot)

- Crear una **API RESTful** que exponga los endpoints necesarios para la gestión de usuarios, sitios, reseñas y fotos.

- Implementar un sistema de autenticación de usuarios utilizando **JWT**.

- Desarrollar un endpoint de **login** que devuelva un token válido para las subsiguientes peticiones.
- Proteger las rutas de la API, permitiendo el acceso solo a usuarios autenticados.
- Conectar a la base de datos PostgreSQL y ejecutar consultas utilizando sentencias SQL puras.

3. Requisitos del Frontend (Vue.js)

- Crear una **interfaz de usuario** intuitiva y atractiva.
- Implementar una página de **login** para que los usuarios puedan autenticarse.
- Una vez autenticados, los usuarios deben poder ver un **mapa interactivo** que muestre los sitios turísticos con sus detalles. (Para esta entrega solo es necesario tener en cuenta en el diseño el espacio adecuado para el mapa)
- La interfaz debe permitir a los usuarios agregar nuevos sitios, subir fotos, escribir reseñas y crear listas personalizadas.
- Utilizar **Axios** para todas las peticiones HTTP al backend.

Funcionalidades Clave del Sistema

- **Login de Usuario:** Autenticación de usuarios a través de un token JWT.
- **Mapa Interactivo:** Visualización de sitios turísticos en un mapa con la posibilidad de filtrar por tipo, calificación y distancia. (Para esta entrega no es necesario el mapa, solo planificar teniendo en cuenta que existirá más adelante)
- **Interacción Social:** Los usuarios pueden agregar sitios, subir fotos, escribir reseñas y seguir a otros usuarios.

- **Exploración:** Posibilidad de buscar y descubrir nuevos lugares basándose en la ubicación actual del usuario o un área seleccionada.
- **Perfiles de Usuario:** Los usuarios pueden ver sus contribuciones (reseñas, fotos, listas) y la actividad de otros usuarios.

Documentación y Entrega

Todo el proyecto debe ser subido a un **repositorio de GitHub**. La entrega debe incluir:

- **Documentación de la Base de Datos:** Un documento que describa el esquema, las relaciones entre tablas y el propósito de cada trigger, procedimiento almacenado, vista materializada e índice.
- **Script de Creación y Carga de Datos:** Un archivo .sql que permita recrear la base de datos completa, incluyendo las tablas, índices, triggers y un conjunto de datos de prueba para realizar las pruebas.
- **Código Fuente:** El código completo del backend (Spring Boot) y el frontend (Vue.js), subido al repositorio de GitHub.
- **README.md:** Un archivo README.md en el repositorio principal que contenga instrucciones claras sobre cómo clonar, configurar y ejecutar la aplicación.

Consultas SQL a desarrollar

1. **Cálculo de Calificación Promedio y Conteo de Reseñas:** Escribe una consulta SQL que, para cada tipo de sitio turístico, calcule la calificación promedio y el número total de reseñas. La consulta debe mostrar el tipo de sitio y ambos valores agregados.
2. **Identificación de los Reseñadores Más Activos:** Utilizando una subconsulta o una CTE (Common Table Expression), encuentra a los 5 usuarios que han escrito el mayor número de reseñas en los últimos 6 meses. Muestra el nombre de usuario y el conteo de reseñas.

3. **Análisis de Proximidad entre Sitios Turísticos:** Crea una consulta que encuentre todos los sitios turísticos de tipo 'Restaurante' que están a menos de 100 metros de un sitio de tipo 'Teatro'. La consulta debe devolver el nombre de ambos sitios y la distancia entre ellos.
4. **Detección de Sitios con Valoraciones Inusuales:** Escribe una consulta que identifique todos los sitios turísticos que tienen una calificación promedio superior a 4.5, pero que tienen menos de 10 reseñas. Muestra el nombre del sitio y su calificación promedio.
5. **Análisis de Popularidad por Región:** Calcula la cantidad total de reseñas por cada región o ciudad. La consulta debe mostrar el nombre de la región y el total de reseñas, ordenado de mayor a menor.
6. **Actualización Masiva de Calificaciones:** Implementa un trigger que, después de insertar una nueva reseña, actualice la calificación promedio del sitio turístico asociado en su tabla, para mantener un valor agregado.
7. **Listado de Sitios con Pocas Contribuciones:** Crea una consulta que muestre todos los sitios turísticos que no tienen reseñas o fotos asociadas en los últimos 3 meses. La consulta debe incluir el nombre del sitio, su tipo y la fecha de su última contribución.
8. **Análisis de Contenido de Reseñas:** Escribe una consulta que identifique las 3 reseñas más largas (por número de caracteres) escritas por un usuario con una calificación promedio de reseñas superior a 4.0. La consulta debe mostrar el nombre de usuario, el nombre del sitio y el contenido de la reseña.
9. **Resumen de Contribuciones por Usuario:** Crea una vista materializada llamada `resumen_contribuciones_usuario` que muestre la cantidad total de reseñas, fotos y listas creadas por cada usuario. Esta vista debe ser refrescada concurrentemente y debe poder ser consultada rápidamente por los perfiles de usuario.