

# Prueba Técnica CMPC

Lea atentamente los enunciados. Si tiene que hacer supuestos explíquelos y fundamente sus respuestas. Considere todo lo necesario para que lo que se le pide funcione en un ambiente productivo real.

## Objetivo

Desarrollar una aplicación web completa para la tienda CMPC-libros que digitalice sus procesos de inventario, incluyendo funcionalidades avanzadas de gestión y análisis de datos.

Se debe considerar los siguientes datos en un libro: título, autor, editorial, precio, disponibilidad, y género.

## Requerimientos Funcionales

### Frontend (React con TypeScript)

1. Login de autenticación
2. Desarrollar un listado de libros con las siguientes características:
  - Filtrado avanzado por género, editorial, autor y disponibilidad.
  - Ordenamiento dinámico por múltiples campos.
  - Paginación del lado del servidor.
  - Búsqueda en tiempo real con debounce.
3. Formulario de alta/edición de libro con:
  - Validación reactiva de formularios.
  - Carga de una imagen por libro.
4. Visualizar los datos disponibles sobre un libro

### Backend (NestJS con TypeScript)

1. Diseñar una arquitectura modular y escalable utilizando los principios SOLID.
2. Implementar un sistema de autenticación JWT.
3. Desarrollar endpoints RESTful para todas las operaciones CRUD de libros.
4. Crear endpoints adicionales para:
  - Exportación de datos en formatos CSV.
5. Utilizar técnicas de soft delete para el manejo de eliminaciones.
6. Implementar un sistema de logging para auditoría de operaciones.

### Base de Datos (PostgreSQL con sequelize ORM)

1. Diseñar un modelo de datos normalizado que incluya:
  - Relaciones apropiadas entre las tablas.
  - Índices para optimizar consultas frecuentes.
2. Utilizar transacciones para garantizar la integridad de los datos en operaciones críticas.

## Testing

1. Implementar tests unitarios para componentes y servicios en Nest.
2. Crear tests unitarios para servicios y controladores en NestJS.
3. Alcanzar una cobertura de código de al menos 80%.

## **DevOps y Despliegue**

1. Proporcionar un docker-compose.yml para el despliegue local de todo el stack.

## **Documentación**

1. Crear un README.md detallado con:
  - Instrucciones de instalación y configuración.
  - Guía de uso de la aplicación.
  - Descripción de la arquitectura y decisiones de diseño.
2. Documentar la API utilizando Swagger/OpenAPI.
3. Proporcionar un diagrama de la arquitectura del sistema.
4. Incluir un modelo relacional de la base de datos (puede ser en formato imagen o utilizando herramientas como dbdiagram.io).

## **Requerimientos Adicionales**

1. Implementar manejo de errores tanto en frontend como en backend.
2. Utilizar interceptores en NestJS para transformación y manipulación de respuestas.

## **Criterios de Evaluación**

- Calidad y legibilidad del código.
- Arquitectura y escalabilidad de la solución.
- Rendimiento y optimización.
- Cobertura y calidad de los tests.
- Usabilidad y experiencia de usuario.
- Documentación y facilidad de despliegue.
- Uso apropiado de patrones de diseño y mejores prácticas.

## **Entrega**

- Código fuente en un repositorio Git (GitHub, GitLab, etc.).
- Documentación completa según lo especificado.