Proyecto de Semestre  
Sistema de Gestión de Sesiones de Bienestar

Curso: Programación 2

Universidad Mariano Gálvez de Guatemala

# 1. Contexto

Los centros de bienestar enfrentan dificultades para organizar sus citas, facturación e historial de clientes. Procesos manuales generan errores, retrasos y falta de trazabilidad. Este proyecto propone un Sistema de Gestión de Sesiones de Bienestar que incluye:

* Módulo web administrativo (React) para administrador/recepcionista.
* Módulo móvil (Android) para clientes.
* Backend (Spring Boot + PostgreSQL en la nube) con API REST.

# 2. Problemática

Los 16 casos de uso definen la funcionalidad clave del sistema: registrar clientes, agendar/cancelar citas, gestionar servicios, consultar historial, facturar, generar reportes y enviar notificaciones. Estos solucionan la falta de centralización, trazabilidad, comunicación con clientes y eficiencia administrativa.

# 3. Restricciones del Proyecto

* Trabajo individual.
* Tecnologías obligatorias: Spring Boot, PostgreSQL, React, Android Studio (Java).
* Buenas prácticas: POO avanzada (encapsulamiento, herencia, interfaces, polimorfismo).
* Principios SOLID aplicados y documentados.
* Patrones de diseño vistos en clase (DAO, Singleton, Observer, Facade, Decorator, MVC).
* Seguridad (validación de datos, autenticación).
* Gestión del código: uso de Git y GitHub para versionamiento.
* Pruebas: al menos pruebas unitarias básicas con JUnit.

# 5. Entrega Final (Defensa Técnica)

## 5.1 Repositorio en GitHub

* El proyecto debe estar versionado en GitHub, con historial de commits individuales.
* Debe incluir un README.md con instrucciones claras de instalación y ejecución.
* Se valorará el uso de issues, branches o pull requests.

## 5.2 Configuración en entorno local (localhost)

* Backend ejecutándose en localhost:8080.
* Frontend web en localhost:3000.
* App móvil configurada para consumir endpoints desde localhost o la IP local.

## 5.3 Conexión a base de datos en la nube (PostgreSQL)

* El backend debe conectarse a PostgreSQL en la nube (ej. Railway, Render, ElephantSQL, Supabase).
* Documentar la URL de conexión (sin exponer credenciales sensibles).
* Instrucciones para cambiar entre BD en nube y H2 en memoria (opcional).

## 5.4 Manual Técnico

* Arquitectura del sistema (diagrama de capas, módulos, componentes).
* Modelo entidad–relación y script de creación de la BD.
* Descripción de clases principales y cómo aplican POO/SOLID.
* Patrones de diseño utilizados (ejemplos con código).
* Configuración del entorno (backend, frontend web, móvil).
* Pruebas unitarias implementadas.

## 5.5 Manual de Usuario

* Guía de acceso a las aplicaciones (web y móvil).
* Descripción de roles y sus funcionalidades.
* Instrucciones paso a paso con capturas de pantalla: registro/login, agendar/cancelar cita, consultar historial, ver facturas, recibir notificaciones.

## 5.6 Presentación Final

* Demo técnica mostrando la integración completa (backend + web + móvil).
* Justificación de cómo aplicaron POO, SOLID, JPA, patrones de diseño, seguridad y pruebas.
* Tiempo estimado: 15–20 minutos.