Taller: Modularización y componetización

**Integrante:**

David Alejandro Luna

# Sistema Web para Centro Médico

1. Requisitos funcionales

**Pacientes:**

RF01 Registrarse e iniciar sesión

- Qué: Permitir a los pacientes crear una cuenta y autenticarse.
- Cómo: Mediante formulario con datos personales y credenciales seguras (usuario/contraseña).
- Para qué: Garantizar el acceso personalizado y seguro a los servicios del sistema.

RF02 Reservar, cancelar o reprogramar citas

- Qué: Gestionar citas médicas de manera autónoma.
- Cómo: A través de un calendario interactivo con disponibilidad de médicos.
- Para qué: Facilitar la autogestión de la atención médica y reducir tiempos administrativos.

RF03 Consultar historial de consultas

- Qué: Acceder al registro de citas pasadas y observaciones médicas.
- Cómo: Mediante un panel de historial asociado al perfil del paciente.
- Para qué: Brindar continuidad en la atención y acceso a información relevante para el paciente.

**Médicos:**

RF04 Gestionar agenda

- Qué: Administrar horarios de disponibilidad y confirmar citas.
- Cómo: Por medio de un módulo de agenda sincronizado con las reservas de pacientes.
- Para qué: Organizar eficientemente el tiempo de atención y evitar conflictos de horarios.

RF05 Registrar observaciones médicas en cada cita

- Qué: Ingresar diagnósticos, tratamientos o notas en cada consulta.
- Cómo: Usando formularios electrónicos vinculados a la cita correspondiente.
- Para qué: Documentar el proceso clínico y dar soporte al historial del paciente.

**Administradores:**

RF06 Crear y editar perfiles de médicos

- Qué: Gestionar la información profesional de los médicos.
- Cómo: Mediante un módulo administrativo con permisos avanzados.
- Para qué: Mantener actualizada la base de datos de especialistas.

RF07 Monitorear estadísticas de citas

- Qué: Visualizar reportes de uso del sistema (número de citas, cancelaciones, asistencias).
- Cómo: A través de dashboards e informes exportables.
- Para qué: Evaluar el desempeño del sistema y optimizar la gestión del servicio.

RF08 Control de usuarios y permisos

- Qué: Administrar roles de pacientes, médicos y administradores.
- Cómo: Asignando niveles de acceso diferenciados en el sistema.
- Para qué: Garantizar seguridad y control en el manejo de la información.

Módulos del sistema

- Gestión de pacientes
- Gestión de médicos y agenda
- Gestión de citas
- Gestión administrativa
- Módulo de notificaciones (email/SMS)
- Base de datos central (pacientes, médicos, citas, usuarios)

1. Interfaces – Conectores

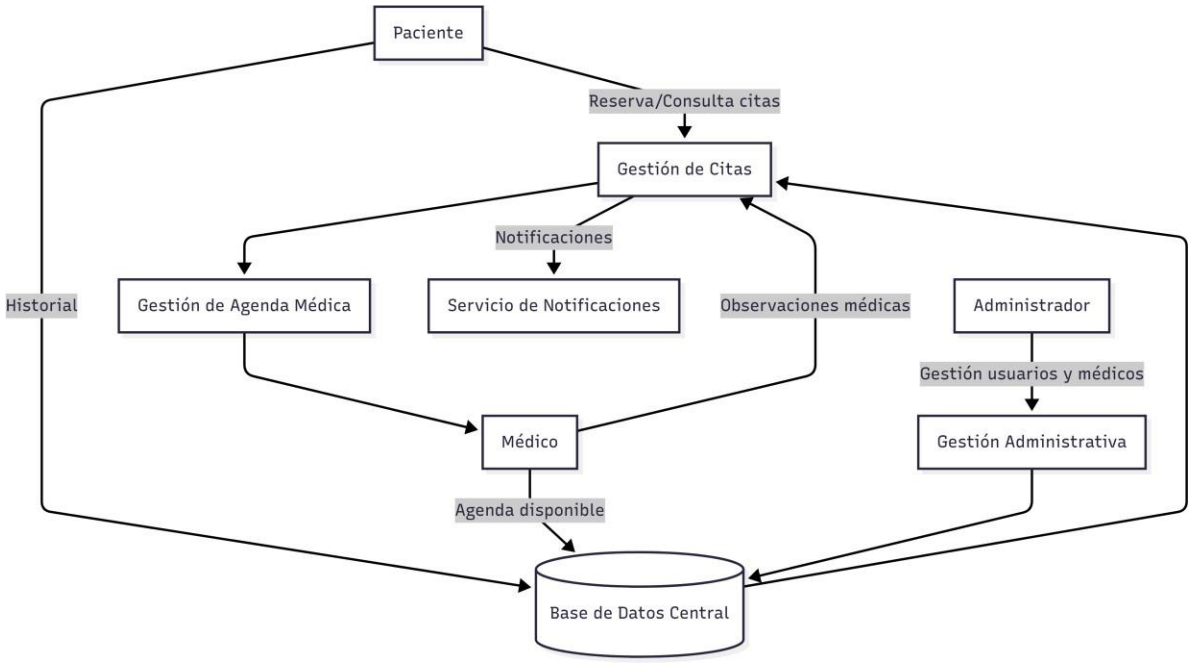
- API REST entre frontend web y backend.
- Base de datos relacional (PostgreSQL).
- Conector de notificaciones vía servicios externos (SendGrid).

2. Análisis de diseño

Se propone arquitectura multicapa (MVC):

- Frontend web para pacientes, médicos y admin.
- Backend que expone APIs seguras.
- Separación clara permite escalabilidad (ej. agregar telemedicina a futuro).

3. Flowchart




# Aplicación tipo Rappi/Uber Eats

1. Requisitos funcionales

**Usuario cliente:**

RF01 Registrarse e iniciar sesión

- Qué: Permitir a los clientes crear una cuenta y autenticarse en la plataforma.

- Cómo: Mediante un formulario con datos personales y credenciales seguras (usuario/contraseña o redes sociales).
- Para qué: Garantizar acceso personalizado, seguro y con historial de pedidos.

RF02 Ver restaurantes y menús

- Qué: Mostrar el listado de restaurantes disponibles y sus menús.
- Cómo: A través de un catálogo filtrable por ubicación, categoría o disponibilidad.
- Para qué: Facilitar la elección de opciones gastronómicas según la preferencia del cliente.

RF03 Realizar pedidos y pagos en línea

- Qué: Permitir al cliente seleccionar productos, confirmar pedido y pagar digitalmente.
- Cómo: Mediante carrito de compras con integración a pasarelas de pago (tarjeta, PSE, billeteras digitales).
- Para qué: Agilizar la compra y brindar comodidad en la experiencia de usuario.

RF04 – Seguir estado de la entrega

- Qué: Dar seguimiento en tiempo real al pedido desde la preparación hasta la entrega.
- Cómo: Con notificaciones y rastreo de ubicación del repartidor en un mapa.
- Para qué: Mantener informado al cliente y generar confianza en el servicio.

**Restaurante:**

RF05 Gestionar menús y disponibilidad

- Qué: Administrar platos, precios y disponibilidad de productos.
- Cómo: A través de un panel de control editable en tiempo real.
- Para qué: Mantener actualizado el menú y evitar pedidos de productos no disponibles.

RF06 Recibir y aceptar/rechazar pedidos

- Qué: Visualizar pedidos entrantes y tomar decisiones de aceptación o rechazo.
- Cómo: Con un sistema de notificaciones inmediatas y botones de gestión en el panel del restaurante.

- Para qué: Asegurar un flujo de pedidos eficiente y adaptado a la capacidad de atención.

**Repartidor:**

RF07 – Ver pedidos asignados

- Qué: Acceder a la lista de pedidos que debe entregar.
- Cómo: Desde una aplicación móvil con detalle de dirección, cliente y restaurante.
- Para qué: Organizar la ruta de trabajo y optimizar tiempos de entrega.

RF08 Rastrear ruta de entrega

- Qué: Seguir la mejor ruta para realizar la entrega.
- Cómo: Usando integración con mapas y GPS en la aplicación móvil.
- Para qué: Reducir tiempos de traslado y asegurar puntualidad.

RF09 Confirmar entrega

- Qué: Validar que el pedido fue entregado correctamente al cliente.
- Cómo: A través de botón de confirmación en la aplicación con firma digital o código de verificación.
- Para qué: Cerrar el ciclo de pedido y generar trazabilidad en el sistema.

2. Módulos del sistema

- Gestión de usuarios y autenticación
- Módulo de pedidos
- Módulo de pagos
- Gestión de restaurantes
- Gestión de repartidores
- Geolocalización y tracking
- Notificaciones push
- Base de datos distribuida (usuarios, pedidos, menús, entregas)

3. Interfaces – Conectores

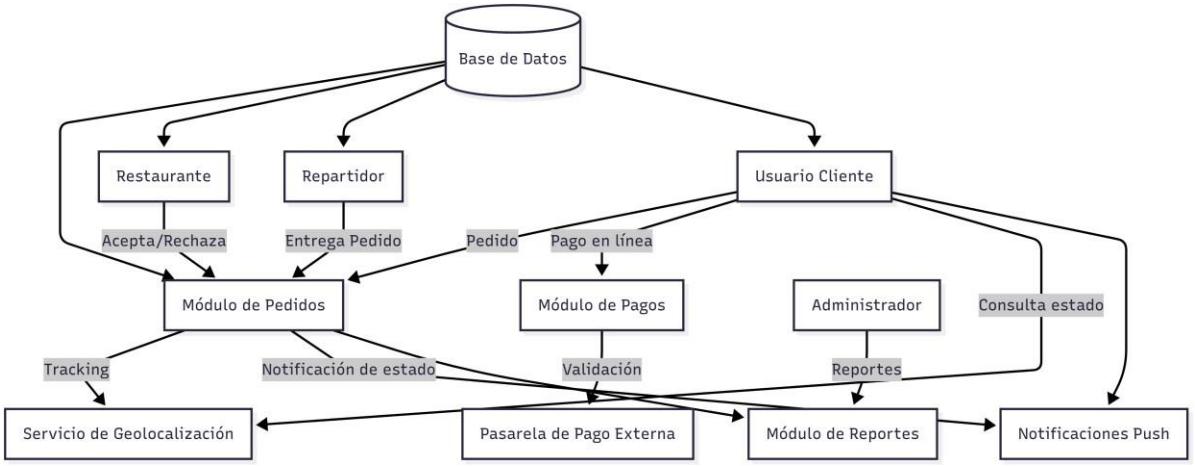
- APIs REST/GraphQL para comunicación móvil/web con backend.
- Pasarela de pago (ej. Stripe, PayU).
- Servicios de mapas (Google Maps API).
- Notificaciones push (Firebase Cloud Messaging).

4. Análisis de diseño

Se recomienda arquitectura de microservicios porque:

- Permite escalar servicios como pagos o pedidos de manera independiente.
- Uso de caché (Redis) para mejorar la velocidad de búsqueda de restaurantes.
- Flexibilidad para soportar app móvil + web en paralelo.

5. Flowchat




# Sistema de Biblioteca

1. Requisitos funcionales

RF01 Gestión de usuarios (estudiantes, profesores, administradores)

- Qué: Administrar el registro, edición y control de usuarios con diferentes roles.
- Cómo: Mediante un módulo de usuarios con niveles de permisos diferenciados (acceso a préstamos, reportes o gestión administrativa).
- Para qué: Garantizar la organización y seguridad en el uso del sistema según el perfil del usuario.

RF02 – Gestión de catálogo (libros físicos y digitales)

- Qué: Registrar, clasificar y actualizar la información de los materiales disponibles en la biblioteca.
- Cómo: A través de una base de datos con búsqueda por título, autor, categoría o formato.
- Para qué: Facilitar el acceso a los recursos y mantener actualizado el inventario bibliográfico.

RF22 – Préstamos y devoluciones

- Qué: Permitir a los usuarios solicitar préstamos y registrar devoluciones de libros.
- Cómo: Mediante un sistema automatizado que controle fechas de préstamo, renovaciones y sanciones por retrasos.
- Para qué: Optimizar la gestión de circulación del material y garantizar disponibilidad a todos los usuarios.

RF23 – Reportes de uso y disponibilidad

- Qué: Generar reportes sobre cantidad de préstamos, devoluciones, materiales más consultados y disponibilidad actual.
- Cómo: A través de paneles de estadísticas e informes exportables.
- Para qué: Apoyar la toma de decisiones administrativas y mejorar la eficiencia del servicio bibliotecario.

2. Módulos del sistema

- Módulo de usuarios
- Módulo de catálogo
- Módulo de préstamos/devoluciones
- Módulo de reportes
- Base de datos modernizada

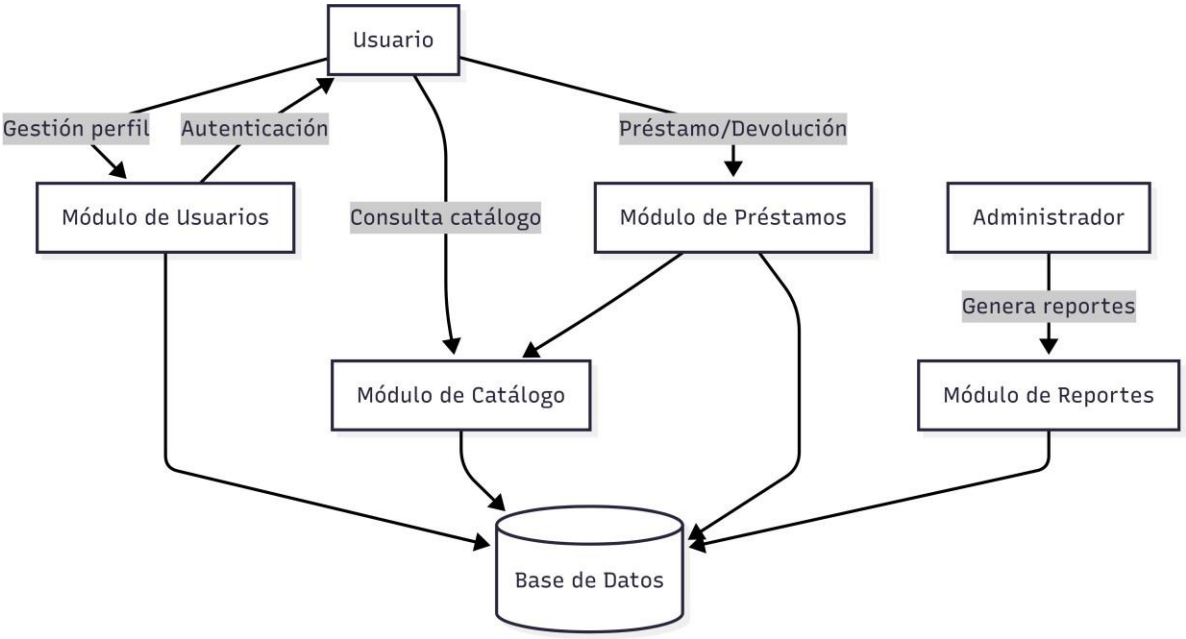
3. Interfaces – Conectores

- APIs internas para que los módulos se comuniquen.
- Frontend web que consume las APIs.
- Base de datos modularizada con relaciones claras (usuarios, libros, préstamos).

4. Análisis de diseño

- Se aplica arquitectura orientada a servicios (SOA):
- Cada funcionalidad aislada en su módulo, evitando el monolito original.
- Facilita mantenimiento y escalabilidad (ej. agregar libros digitales o integración con bibliotecas externas).
- Se prepara el sistema para despliegue en la nube y uso de analítica de datos.

5. Flowchat


Usuario
Gestión perfil
Autenticación
Préstamo/Devolución
Módulo de Usuarios
Consulta catálogo
Módulo de Préstamos
Administrador
Genera reportes
Módulo de Catálogo
Módulo de Reportes
Base de Datos