**Modularización y componetización.**

Elabore para cada uno de los siguientes enunciados:

1. Identifique las funcionalidades (requisitos funcionales)
2. Modularice el sistema que dará solución a las funcionalidades identificando componentes
3. Identifique las interfaces – conectores que permitirán comunicar los componentes
4. Agregue un análisis breve que justifique las decisiones de diseño para el modelo propuesto.
5. Socialice en clase sus propuestas de diseño.

**Ejercicio 1.** Un centro médico requiere un sistema web para que pacientes puedan reservar consultas, médicos gestionen su agenda y los administradores controlen el sistema.

**Ejercicio 2.** Una startup quiere desarrollar una aplicación móvil y web tipo Rappi o Uber Eats, donde los usuarios pueden pedir comida, los restaurantes gestionan pedidos y los repartidores hacen las entregas.

**Ejercicio 3.** Un colegio tiene un sistema heredado de biblioteca que mezcla lógica de préstamo, usuarios, catálogo y reportes en un solo módulo monolítico. El sistema debe ser **refactorizado y rediseñado** usando principios modernos de diseño.

**EJERCICIO 1**

1. R/ **Requisitos funcionales**:

**RF-01 registro de usuario:** Los pacientes deben poder crear una cuenta con nombre, correo, teléfono y contraseña.

**RF-02 iniciar sección:** Pacientes, médicos y administradores pueden entrar al sistema con su usuario y contraseña.

**RF-03 reservar consulta:** Un paciente puede buscar un médico y reservar una cita en una fecha y hora disponibles.

**RF-04 Gestión de agenda:** Los médicos pueden ver su agenda, aceptar o rechazar citas, y bloquear horas (vacaciones, consultas externas).

**RF-05 Funciones Administrativas:** los administradores controlan el sistema (crear cita, agendar cita, modificar cita, eliminar cita, paciente y medico)

**RF-06 Notificaciones:** El sistema envía recordatorios de cita por correo o mensaje (cuando se confirme o cambie una cita).

**RF-07 Historial:** El médico puede guardar una nota breve de la consulta en el historial del paciente.

1. **Modularización:**

**Hospital Web**

Web donde pacientes, médicos y administradores interactúan (buscar médicos, reservar cita, ver agenda).

**Servicio de Autenticación**

Maneja registro, inicio de sesión y permisos (quién es paciente, médico o admin).

**Servicio de Citas**

Encargado de crear, modificar y cancelar citas. Valida que la hora esté libre antes de confirmar.

**Servicio de Agenda (por médico)**

Mantiene la disponibilidad del médico: días hábiles, horarios, bloqueos por vacaciones.

**Servicio de Recursos (Capacidad)**

Controla salas o equipos necesarios y verifica que haya espacio para la cita.

**Servicio de Notificaciones**

Envía correos o mensajes cuando se crea, confirma o cambia una cita.

**Base de datos**

Guarda usuarios, citas, agendas y el historial de consultas.

1. **Interfaz:**

**Hospital Web -> Servicio de Autenticación**

El portal pide “login” o “registro” al servicio de autenticación.

**Hospital Web -> Servicio de Citas**

Cuando un paciente quiere reservar, el portal solicita crear la cita al servicio de citas.

**Servicio de Citas -> Servicio de Agenda**

Antes de confirmar la cita, el servicio de citas consulta si el médico tiene esa hora libre.

**Servicio de Citas -> Servicio de Recursos/Capacidad**

Si la cita requiere sala o equipo, se valida disponibilidad.

**Servicio de Citas -> Servicio de Notificaciones**

Cuando una cita se confirma o cambia, se le dice al servicio de notificaciones para que envíe el correo o SMS al paciente y al médico.

**Todos los servicios -> Base de datos**

Guardan y leen la información (usuarios, citas, agendas, historial).

Separar la autenticación, la gestión de citas y la agenda evita que una sola parte tenga demasiadas funciones.

Evitar errores de reserva, al consultar la agenda antes de confirmar, se reduce la posibilidad de dar dos citas para la misma hora.

Si en el futuro se requiere agregar salas o equipos, el servicio de recursos ya existe para integrarlos sin cambiar todo el sistema.

Seguridad básica centralizar el inicio de sesión mejora el control de permisos (solo administradores pueden eliminar citas).

**EJERCICIO 2**

1. **Requerimientos funcionales:**

**RF-01:** Registrarse (usuarios, restaurantes y repartidores).

**RF-02:** Iniciar sesión (usuarios, restaurantes y repartidores).

**RF-03:** Ver menú de restaurantes.

**RF-04:** Realizar pedido de comida.

**RF-05:** Pago en línea (tarjeta, PSE, efectivo contra entrega).

**RF-06:** Notificaciones al usuario (pedido confirmado, en camino, entregado).

**RF-07:** Gestión de pedidos por parte de los restaurantes (aceptar, preparar, actualizar estado).

**RF-08:** Gestión de entregas por repartidores (aceptar pedido, marcar en camino, entregar).

1. **Modularización:**

**App Móvil/Web**

* Registro e inicio de sesión.
* Exploración de restaurantes.
* Selección de productos y carrito de compras.
* Pago y seguimiento del pedido.

**Módulo de Restaurantes**

* Registro del restaurante.
* Publicar menú y precios.
* Recibir pedidos.
* Actualizar estado de los pedidos (preparando, listo, entregado al repartidor).

**Módulo de Repartidores**

* Registro de repartidor.
* Aceptar pedidos disponibles.
* Marcar estado de entrega (en camino, entregado).

1. **Interfaz:**

**App Móvil/Web -> Módulo de Restaurantes**

* El usuario explora el menú y selecciona productos.
* La App envía el pedido al restaurante.
* El restaurante actualiza el estado del pedido (preparando, listo).

**App Móvil/Web -> Módulo de Repartidores**

* Una vez que el pedido está listo, el sistema asigna un repartidor.
* El usuario puede ver el estado de la entrega (en camino, entregado).

**Módulo de Restaurantes -> Módulo de Repartidores**

* El restaurante notifica al repartidor cuando un pedido está listo para ser recogido.
* El repartidor confirma la aceptación del pedido y actualiza su estado.

1. La **App Móvil/Web** concentra la experiencia del usuario final (pedidos, pagos, seguimiento). El **Módulo de Restaurantes** se enfoca únicamente en la gestión de menús y pedidos, lo que facilita que cada restaurante maneje su propia información sin afectar a otros. El **Módulo de Repartidores** administra la logística de entrega de forma independiente, permitiendo asignar pedidos a diferentes repartidores según disponibilidad.

**EJERCICIO 3**

**Ejercicio 3.** Un colegio tiene un sistema heredado de biblioteca que mezcla lógica de préstamo, usuarios, catálogo y reportes en un solo módulo monolítico. El sistema debe ser **refactorizado y rediseñado** usando principios modernos de diseño.

**1. REQUERIMIENTOS FUNCIONALES:**

**RF-01:** Registro de usuario El estudiante puede crear una cuenta con nombre, correo, matrícula y contraseña.

**RF-02:** Inicio de sesión estudiantes y administradores entran con usuario y contraseña.

**RF-03:** Préstamo de libros buscar libro, solicitar préstamo, renovar y devolver.

**RF-04:** Catálogo consultar catálogo por título, autor, categoría y disponibilidad.

**RF-05:** Reportes generar reportes: préstamos activos, historial por estudiante, libros más prestados, sanciones.

**RF-06:** Gestión administrativa administrador puede agregar/editar libros, gestionar sanciones y consultar reportes.

**2. Modularización:**

BibliotecaWeb

* Interfaz para estudiantes y administradores: registro, búsqueda, pedidos de préstamo, renovaciones, devoluciones y ver reportes.

Autenticación y autorización

* Registro, login, gestión de contraseñas y roles (estudiante, admin).

Catálogo

* Almacena información de libros (título, autor, ISBN, copias disponibles, ubicación).
* Funciones: búsqueda, ficha de libro, actualización por admin.

Préstamos

* Lógica de préstamo: crear préstamo, renovar, devolver, calcular fechas de vencimiento y multas básicas.
* Verifica disponibilidad consultando al CatalogService.

Usuarios/Estudiantes

* Datos del estudiante: perfil, historial de préstamos, sanciones.

Reportes

* Genera reportes periódicos y bajo demanda (PDF o tabla).

**3. Interfaz:**

**BibliotecaWeb → Autenticación y autorización**  
El portal solicita registro o login al servicio de autenticación; recibe un token si el usuario queda validado.

**BibliotecaWeb → Catálogo**  
Cuando un usuario busca o abre la ficha de un libro, el portal pide la lista o los detalles al Catálogo (título, autor, copiasDisponibles, ubicación).

**BibliotecaWeb → Préstamos**  
Cuando un estudiante solicita, renueva o devuelve un libro, el portal envía la petición al servicio de Préstamos.

**Préstamos → Catálogo**  
Préstamos consulta disponibilidad antes de crear un préstamo y solicita actualizar la cantidad de copias (decrementar al prestar, incrementar al devolver).

**Préstamos → Usuarios/Estudiantes**  
Antes de autorizar un préstamo, Préstamos verifica el estado del estudiante (sanciones, préstamosActivos) y, al crear/retornar un préstamo, actualiza el historial del usuario.

**Reportes → Préstamos / Catálogo / Usuarios**  
El módulo Reportes solicita datos agregados para generar informes (préstamos activos, libros más prestados, historial de un estudiante) y devuelve tablas o archivos.

**4.** El diseño propuesto separa el sistema en partes claras (Catálogo, Préstamos y Autenticación), de modo que cada una se encargue de una sola tarea; esto facilita arreglar errores, mejorar funciones y probar cada parte por separado. Un problema importante a cuidar es que dos estudiantes no se queden con la misma copia al mismo tiempo; para evitarlo conviene que, al pedir un libro, el sistema reserve o compruebe la copia y la registre en un solo paso, evitando que otra persona la tome a la vez. También es recomendable concentrar en el módulo de Préstamos las reglas que definen cuántos libros puede tener un estudiante, cuánto tiempo los puede mantener y cómo se aplican las multas, y centralizar el inicio de sesión para controlar quién puede hacer qué. Mantener un registro simple de las acciones (quién hizo qué y cuándo) ayuda para reportes y sanciones.