

ENTREGA PROYECTO #2

PARCHE LECTOR

Presentado por:

Nicolas Arciniegas

Juan Jose Alvarez Lozano

Julian Dario Colmenares Saenz

Julian Santiago Becerra Pulido

Sebastian Castañeda Garcia

Profesor:

Carlos Andres Sierra Virguez

Sabado 8 de Noviembre



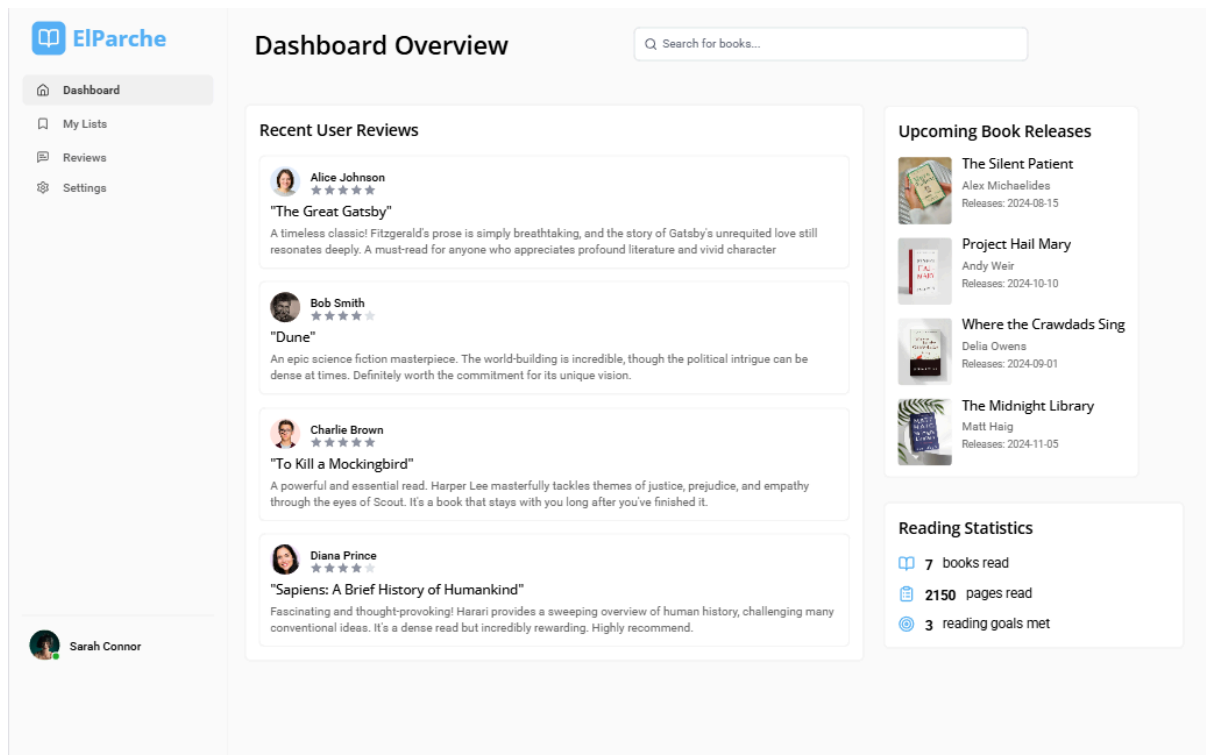
Universidad Nacional de Colombia
Departamento de Ingeniería de sistemas
2025

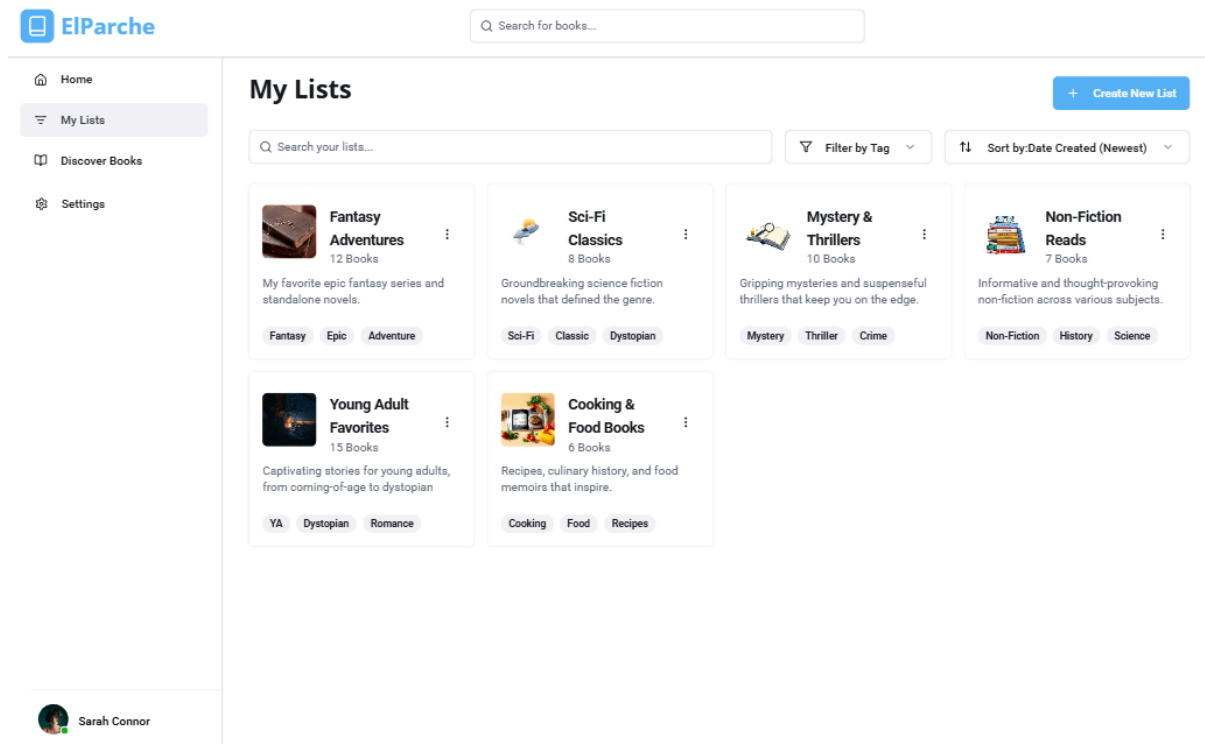
1. CRC Cards

Clase	Responsabilidades	Colaboradores
Usuario	- Registrar y autenticar usuarios- Editar perfil (bio, foto, géneros favoritos)- Seguir y dejar de seguir a otros usuarios- Gestionar sus listas de lectura y reseñas- Ver su feed y estadísticas personales	Autenticacion, Perfil, ListaLectura, Reseña, Estadísticas, Libro
Autenticación	- Registrar nuevos usuarios (email, Google, Apple)- Validar credenciales- Generar y manejar tokens JWT- Recuperar contraseñas	Usuario
Perfil	- Almacenar información personal y preferencias- Mostrar actividad reciente- Configurar privacidad y foto	Usuario, Estadísticas, Reseña, ListaLectura
Libro	- Almacenar datos bibliográficos (título, autor, año, ISBN, sinopsis)- Mostrar promedio de calificaciones y reseñas- Facilitar búsqueda y filtrado	Autor, Reseña, ListaLectura
Autor	- Almacenar nombre y biografía- Asociar sus libros publicados- Permitir a usuarios seguir autores	Libro, Usuario
Reseña	- Permitir crear, editar y eliminar reseñas- Guardar texto, calificación y fecha- Asociar reseñas a usuarios y libros- Calcular promedio de calificaciones del libro	Usuario, Libro
ListaLectura	- Crear, renombrar o eliminar listas personalizadas- Agregar o remover libros- Clasificar libros por estado ("Leyendo", "Leído", "Quiero leer")	Usuario, Libro
Estadísticas	- Calcular métricas (libros leídos, páginas leídas, géneros más leídos)- Actualizarse automáticamente cuando cambia la actividad del usuario- Generar visualizaciones de progreso	Usuario, Libro, ListaLectura

Feed	- Mostrar actividad de usuarios seguidos (reseñas, nuevas listas, calificaciones)- Ordenar cronológicamente las publicaciones- Permitir interacciones (me gusta, comentarios)	Usuario, Reseña, ListaLectura
Buscador	- Buscar libros, autores o usuarios- Filtrar resultados por género, año, calificación, etc.- Conectarse con APIs externas (por ejemplo Google Books API)	Libro, Autor, Usuario

2. Mockups





3. Business Model Processes

1. Registro y Onboarding de Usuarios

Objetivo: Ayudar a los usuarios a crear una cuenta, verificar su identidad, completar su perfil y familiarizarse con la plataforma para aumentar la retención desde el principio.

Desencadenante: El usuario accede a la app y elige crear una cuenta o iniciar sesión.

Participantes: Usuario - Frontend - Servicio de Autenticación - Email/Proveedor OAuth - Base de Datos de Usuario - Sistema de Recomendaciones - Sistema de Perfil.

Pasos Principales:

1. Usuario selecciona método de autenticación (Email, Google o Apple); sistema valida disponibilidad.
2. Validación de credenciales: Para email, verificar formato y confirmar por correo; para OAuth, validar token externo; crear cuenta en DB.
3. Configuración inicial de perfil: Ingresar nombre de usuario, bio (opcional), subir foto (opcional), seleccionar géneros favoritos (mínimo 3).
4. Sugerencias de seguimiento: Sistema sugiere usuarios populares o con gustos similares; usuario sigue o salta.
5. Tour de funcionalidades: Mostrar características principales, explicar creación de reseña y dirigir a la búsqueda de primer libro.
6. Generar JWT y devolver al frontend; guardar sesión/último acceso.

Excepciones/Compensaciones:

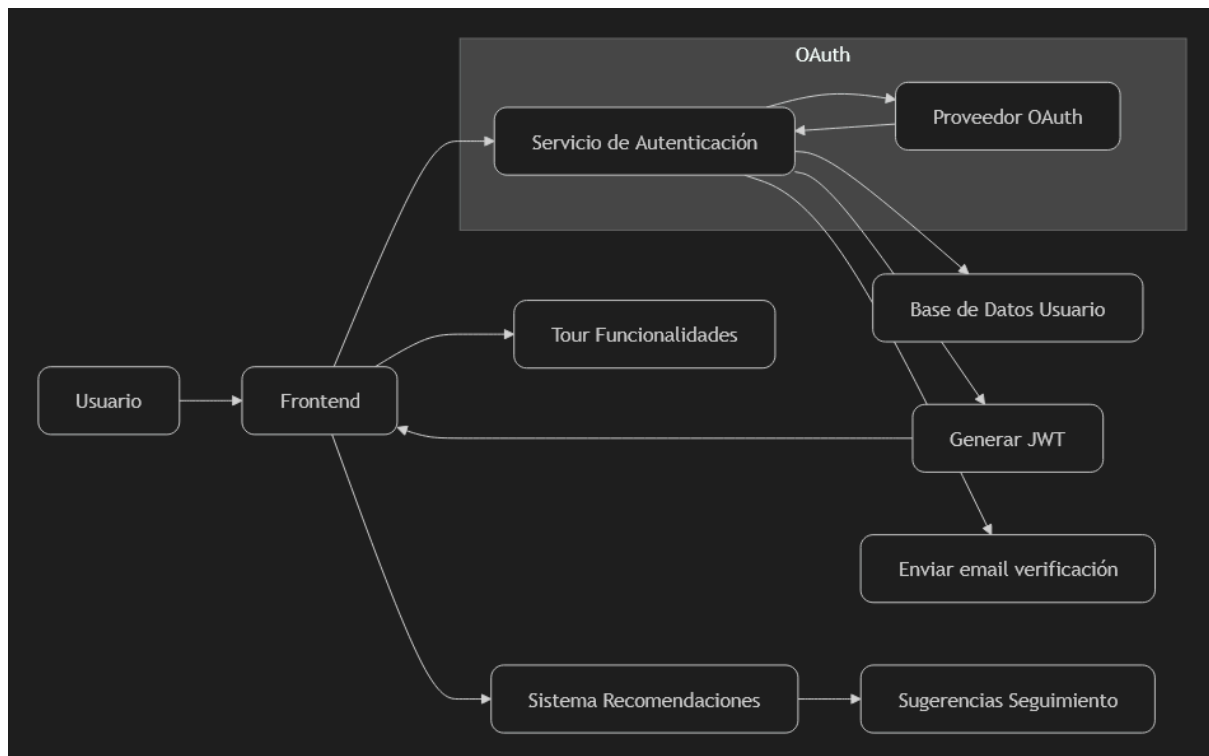
- Error de credenciales → Notificar y bloquear tras 5 intentos.
- Email existente → Ofrecer recuperación de contraseña.
- Proveedor OAuth falla → Fallback a registro con email.

Entradas/Salidas:

- Entrada: Email, contraseña, datos OAuth, datos de perfil inicial.
- Salida: JWT, usuario creado/validado, email de verificación, perfil inicial configurado, tour completado.

KPIs: Tasa de conversión de registro (usuarios que completan onboarding), tasa de verificación por email, intentos de login fallidos, tiempo medio de autenticación (<2s), % de usuarios que siguen al menos 1 sugerencia.

Diagrama:



2. Gestión de Perfil de Usuario

Objetivo: Permitir la gestión de bio, foto, géneros favoritos, privacidad y actualización de datos para personalizar la experiencia.

Desencadenante: Usuario solicita ver/editar su perfil.

Participantes: Usuario - Frontend - Servicio Perfil - Almacenamiento (imágenes) - Base de Datos Usuario - Sistema de Estadísticas.

Pasos Principales:

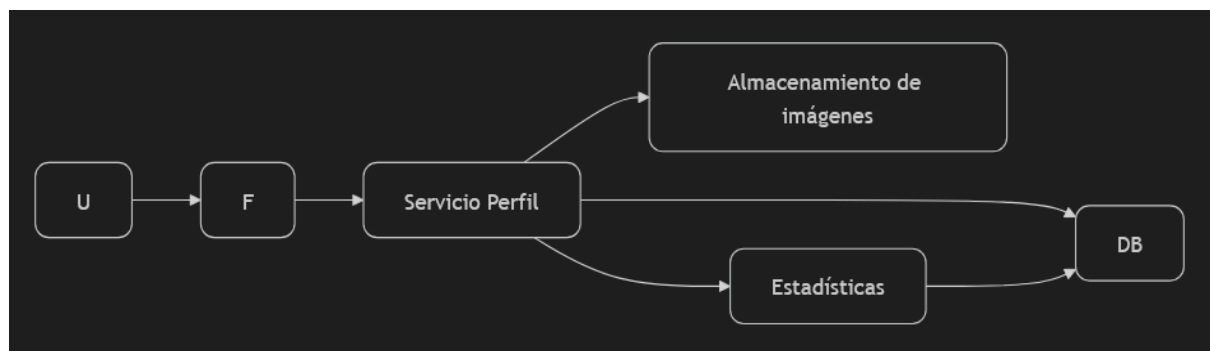
1. Usuario abre la sección Perfil; frontend solicita datos al Servicio Perfil.
2. Usuario edita campos (bio, géneros, foto) y envía.
3. Servicio Perfil valida, guarda en DB y almacena foto en CDN.
4. Actualizar estadísticas públicas y notificar a Feed si cambia la privacidad.
5. Mostrar estadísticas públicas, reseñas recientes y listas en visualización.

Excepciones/Compensaciones: Upload fallido → Reintentar o revertir cambios locales.

Entradas/Salidas:

- Entrada: Datos editados (bio, foto, géneros).
- Salida: Perfil actualizado, visualizaciones generadas.

KPIs: Ratio de completitud de perfiles (>70%), tiempo medio de edición (<3s de cambios visibles), uso de fotos de perfil.



3. Gestión de Libros y Autores

Objetivo: Mantener catálogo actualizado con metadatos, calificaciones e indexación para búsquedas eficientes.

Desencadenante: Nuevo libro identificado mediante búsqueda o entrada manual.

Participantes: Usuario/Editor - Servicio Libro - Servicio Autor - Base de Datos Libros - API Externa (Google Books/OpenLibrary) - Sistema de Búsqueda.

Pasos Principales:

1. Búsqueda/ingesta desde API externa (por título/autor/ISBN) o alta manual.
2. Servicio Libro normaliza metadatos (título, autor, sinopsis, portada).
3. Asociar o crear entidad Autor; agregar libro a DB si es nuevo.
4. Calcular/actualizar el promedio de calificaciones con reseñas existentes.
5. Indexar libro para Buscador.

Excepciones/Compensaciones: Datos incompletos → Marcar como 'pendiente' y solicitar enriquecimiento.

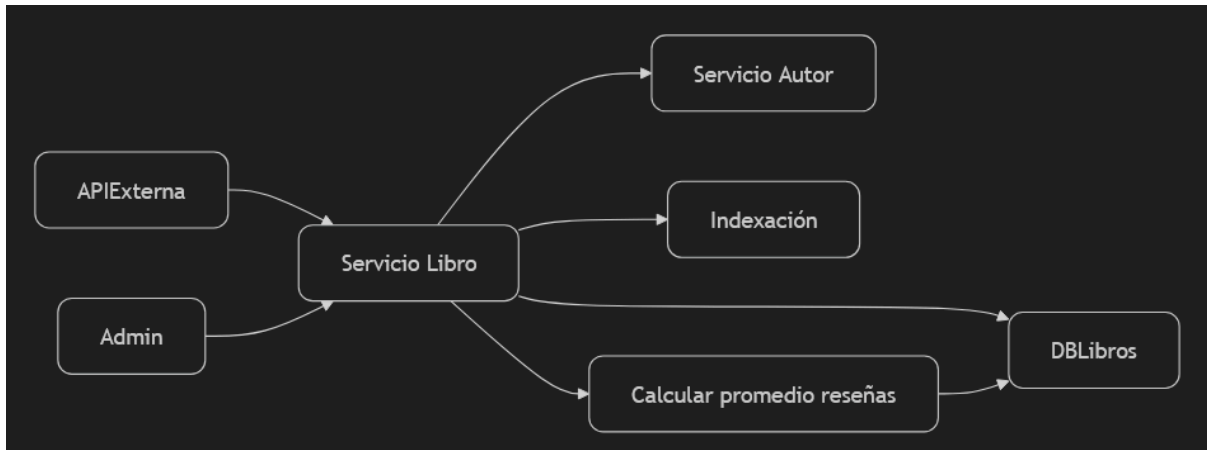
Entradas/Salidas:

- Entrada: Término de búsqueda o datos manuales.

- Salida: Libro/Autor agregado, metadatos normalizados, indexado.

KPIs: Cobertura del catálogo, tiempo de ingestión (<30s), % de libros con metadatos completos.

Diagrama:



4. Ciclo de Vida de Reseñas (Crear-Editar-Eliminar)

Objetivo: Permitir crear, publicar, editar y eliminar reseñas, actualizando métricas del libro y distribuyendo en feeds.

Desencadenante: Usuario decide reseñar un libro.

Participantes: Usuario - Frontend - Servicio Reseñas - Servicio Libro - Sistema de Moderación - Sistema de Feed - Sistema de Notificaciones - Sistema de Estadísticas.

Pasos Principales:

1. Búsqueda y selección de libro: Buscar por título/autor/ISBN; consultar DB local o API externa; agregar si es nuevo.
2. Creación de reseña: Asignar calificación (1-5), escribir texto, marcar spoilers, indicar fecha de lectura.
3. Actualización de estado: Marcar como "Leído", agregar a ListaLectura, actualizar estadísticas.
4. Validación y publicación: Validar longitud/calificación; guardar en DB; recalculando promedio del libro.
5. Distribución: Notificar seguidores; aparecer en feed, perfil y página del libro.
6. Permitir edición/borrado con moderación si aplica.

Excepciones/Compensaciones: Reseña reportada → Marcar para moderación, retractar si es inapropiado; moderación automática falla → Cola manual.

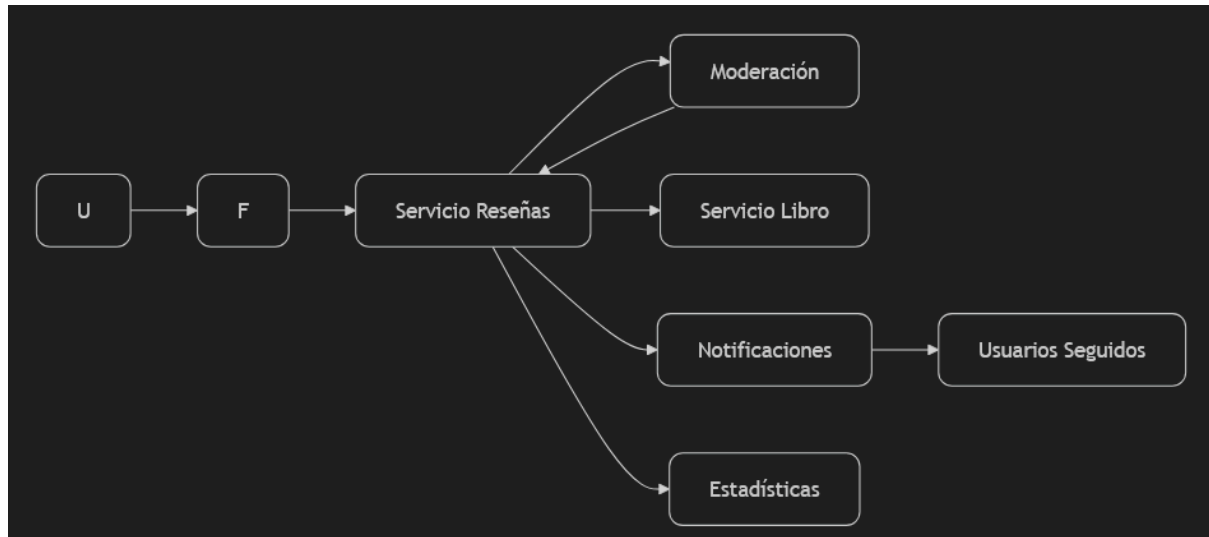
Entradas/Salidas:

- Entrada: Datos de reseña (texto, calificación, libro).

- Salida: Reseña publicada, métricas actualizadas, notificaciones enviadas.

KPIs: Reseñas por usuario/mes, tasa de edición, tiempo a publicación (<10s si automática), % moderado manualmente, % de usuarios que publican primera reseña.

Diagrama:



5. Gestión de Listas de Lectura

Objetivo: Organizar libros en listas automáticas o personalizadas, actualizar estados y compartir para interacción social.

Desencadenante: Usuario decide organizar libros o cambiar el estado.

Participantes: Usuario - Frontend - Servicio ListaLectura - Servicio Libro - Sistema de Estadísticas - Sistema de Feed.

Pasos Principales:

1. Selección de tipo: Automáticas ("Leyendo", "Leído", "Quiero leer") o personalizada.
2. Creación personalizada: Ingresar nombre, descripción, privacidad; crear lista vacía.
3. Adición de libros: Buscar/seleccionar libro, elegir lista; guardar cambios.
4. Actualización de estado: Remover de otras listas si es de estado; actualizar estadísticas.
5. Visibilidad: Si pública, aparecer en perfil; generar URL para compartir; opcional actividad en feed.

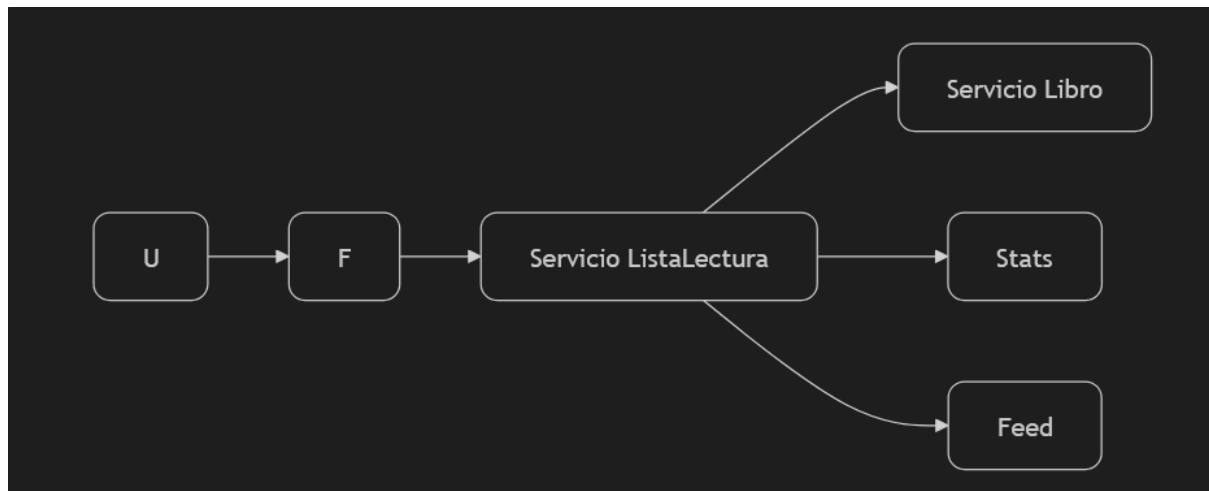
Excepciones/Compensaciones: Libro no encontrado → Ofrecer importación o sugerencia.

Entradas/Salidas:

- Entrada: Nombre de la lista, libros seleccionados.
- Salida: Lista actualizada, estados cambiados, compartible.

KPIs: Listas creadas por usuario, libros por lista, interacciones en listas compartidas, cambios visibles <3s.

Diagrama:



6. Feed Social e Interacciones

Objetivo: Agregar y presentar actividad de seguidos cronológicamente, permitiendo interacciones para aumentar engagement.

Desencadenante: Usuario accede al feed o publica actividad.

Participantes: Usuario - Frontend - Servicio Feed - Servicio Reseñas - Servicio ListaLectura - Servicio Interacciones - Sistema de Notificaciones.

Pasos Principales:

1. Solicitud de feed: Identificar seguidos; usuario refresca.
2. Agregación: Consultar actividad reciente (reseñas, listas, libros leídos, nuevos seguidos).
3. Ordenamiento: Por fecha (reciente primero); límite 20 por carga; eliminar duplicados.
4. Presentación: Mostrar con usuario, tipo, resumen, opciones (like, comentar).
5. Interacción: Registrar likes/comentarios; actualizar métricas.
6. Carga adicional: Infinite scroll para más lotes.

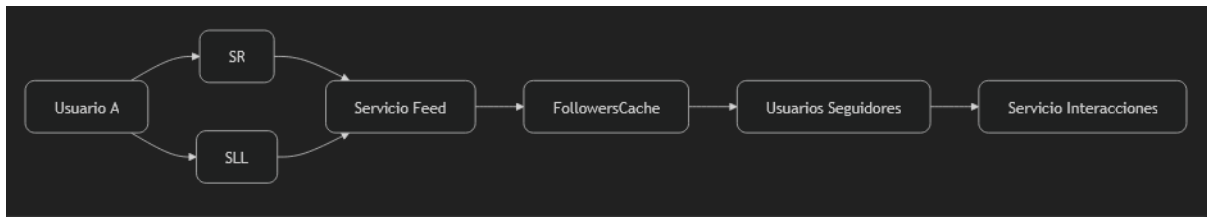
Excepciones/Compensaciones: Spam → Marcar para moderación.

Entradas/Salidas:

- Entrada: Acceso al feed.
- Salida: Feed generado, interacciones registradas.

KPIs: Engagement del feed (likes/comentarios), tiempo en plataforma tras ver feed, entrega a seguidores <5s, respuesta media por publicación.

Diagrama:



7. Buscador y Descubrimiento de Libros

Objetivo: Encontrar libros basados en criterios, integrando APIs externas para descubrimiento.

Desencadenante: Usuario inicia búsqueda.

Participantes: Usuario - Frontend - Servicio Buscador - Index (Elasticsearch) - API Externa - Sistema de Libro - Sistema de Reseña - Sistema de ListaLectura.

Pasos Principales:

1. Entrada: Término (título/autor/ISBN), filtros (género, año, calificación) o explorar trending.
2. Procesamiento: Consultar DB local; si <10 resultados, API externa; combinar/ordenar.
3. Presentación: Mostrar con portada, título, calificación promedio, reseñas, estado en listas del usuario.
4. Refinamiento: Ajustar filtros/orden (relevancia, popularidad); actualizar en real-time.
5. Selección: Mostrar página detallada; acciones (agregar lista, reseñar, ver reseñas).

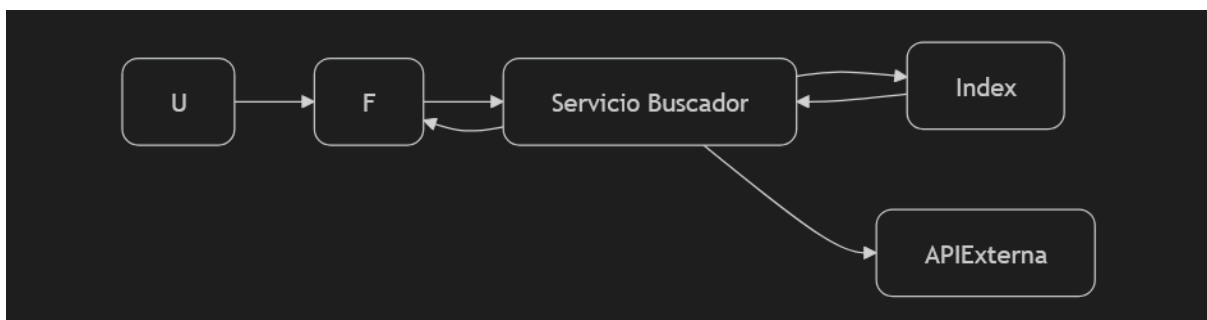
Excepciones/Compensaciones: Índice desincronizado → Fallback a DB o API.

Entradas/Salidas:

- Entrada: Consulta/filtros.
- Salida: Resultados paginados, acciones disponibles.

KPIs: Tasa de éxito de búsqueda, tiempo de respuesta (TTFB <200ms en 95%), tasa de click-through, libros buscados vs agregados.

Diagrama:



8. Cálculo de Estadísticas y Visualizaciones

Objetivo: Calcular métricas personales y generar visualizaciones para motivar usuarios.

Desencadenante: Cambio en actividad (libro leído, reseña, lista).

Participantes: Servicio Evento/Queue - Servicio Estadísticas - Data Warehouse - Frontend
- Sistema de ListaLectura - Sistema de Reseña - Sistema de Libro.

Pasos Principales:

1. Detección de evento: Cambio en listas/reseñas.
2. Recolección: Contar libros por estado, sumar páginas, agrupar por género/año.
3. Cálculo: Total de leídos, páginas, promedio mensual, % de géneros, autores más leídos, calificación promedio.
4. Generación: Gráficos (barras por mes, circular géneros, línea progreso, comparativas anuales).
5. Almacenamiento: Guardar en DB; actualizar dashboard; generar resumen anual.

Excepciones/Compensaciones: Evento duplicado → Deduplicación por ID.

Entradas/Salidas:

- Entrada: Eventos de actividad.
- Salida: Estadísticas calculadas, visualizaciones.

KPIs: Precisión de estadísticas, latencia de actualización (<1 min), número de visualizaciones por usuario.

Diagrama:



9. Notificaciones y Recuperación

Objetivo: Entregar notificaciones relevantes y gestionar la recuperación de acceso.

Desencadenante: Evento (nuevo seguidor, respuesta, verificación).

Participantes: Servicio Notificaciones - Proveedor Email/Push - Usuario - Servicio de Autenticación.

Pasos Principales:

1. Evento genera mensaje en cola.
2. Decidir canal (email/push/in-app) por preferencias.
3. Entregar y registrar.
4. En recuperación: Generar token temporal, enviarlo; validar al cambiar la contraseña.

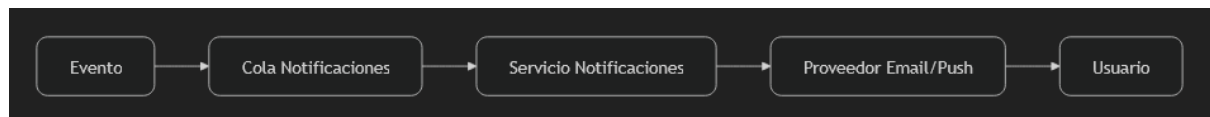
Excepciones/Compensaciones: Entrega fallida → Reintentar.

Entradas/Salidas:

- Entrada: Evento.
- Salida: Notificación entregada.

KPIs: Tasa de entrega, tasa de apertura, tiempo de entrega (push <10s, email <1 min).

Diagrama:



10. Administración, Moderación y Reporting

Objetivo: Moderar contenido, gestionar catálogo y revisar métricas operacionales.

Desencadenante: Incidente reportado o rutina administrativa.

Participantes: Admin - Herramienta Admin - Servicios (Libro, Reseñas, Usuarios) - Logs/Alertas.

Pasos Principales:

1. Revisar reports/cola de moderación.
2. Tomar acción (eliminar, suspender, corregir).
3. Registrar auditoría y notificar.
4. Generar reportes (KPIs, incidentes).

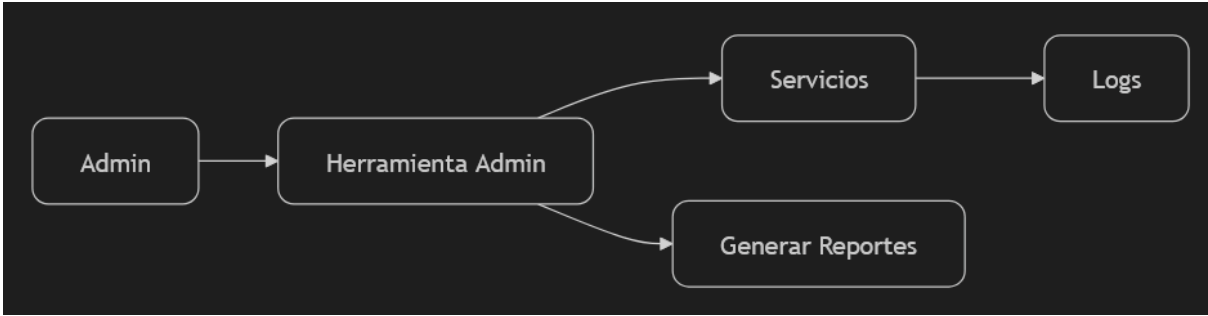
Excepciones/Compensaciones: Moderación automática imprecisa → Revisión manual.

Entradas/Salidas:

- Entrada: Reports/incidentes.
- Salida: Acciones tomadas, reportes generados.

KPIs: Tiempo de resolución, número de incidentes, precisión de moderación automática.

Diagrama:



Planta de Referencia: Owners, Triggers y SLAs

Proceso	Owner	Trigger	SLA Clave
Registro/Autenticación	Auth Team	Usuario crea/ingresa	<2s autenticación, bloqueo tras 5 fallos
Perfil	Product/UX	Usuario edita	Cambios visibles <3s
Libros/Autores	Catalog Team	Ingesta/API	Indexación <30s
Reseñas	Community	Usuario publica	Publicación <10s (automática)
Listas	Product	Usuario gestiona	Cambio visible <3s
Feed	Eng/Infra	Evento usuario	Entrega seguidores <5s (soft)
Búsqueda	Search Team	Query usuario	TTFB <200ms en 95%
Estadísticas	Data	Evento actividad	Actualización <1 min
Notificaciones	Ops	Evento	Push <10s, email <1 min
Administración	Ops/Community	Incidente/rutina	Resolución media <24h

4. Architecture Diagram

Diagrama

Este diagrama define una arquitectura de tres capas desacopladas (Frontend, Backend, Datos) alojada en AWS, ideal para un MVP que necesita escalar.

A. Capa de Presentación (Edge)

Esta capa es responsable de entregar la aplicación de cara al usuario de forma rápida y segura.

- **Usuario (Cliente):** El usuario final que accede a la aplicación "Parche Lector" a través de un navegador web.
- **Amazon S3 (Frontend):** Un bucket de S3 se configura para alojar los archivos estáticos de la aplicación **React con TypeScript** (HTML, CSS, JavaScript).
- **Amazon CloudFront (CDN):** Actúa como la red de distribución de contenido. Cuando un usuario accede al sitio, CloudFront entrega una copia en caché de la aplicación React desde la ubicación geográfica más cercana, reduciendo la latencia. También proporciona HTTPS (SSL/TLS) de forma nativa.

B. Capa de API y Aplicación (Backend)

Aquí reside toda la lógica de negocio. Está diseñada como un **monolito modular**, lo que significa que es una sola aplicación (NestJS) pero está organizada internamente siguiendo los "Servicios" que definimos en los procesos de negocio (como **Servicio de Autenticación**, **Servicio Perfil**, **Servicio Reseñas**, etc.).

- **Amazon API Gateway:** Es el punto de entrada único para todas las solicitudes de API del frontend. Gestiona el enrutamiento, puede aplicar seguridad (como WAF), controlar el tráfico (rate limiting) y manejar la autorización de las solicitudes antes de que lleguen a tu lógica de negocio.
- **AWS Elastic Beanstalk (con NestJS):** Para un MVP, Elastic Beanstalk (EB) es ideal. Simplemente subes tu código **NestJS** y EB gestiona automáticamente el aprovisionamiento de servidores (EC2), el balanceo de carga y el autoescalado. La aplicación NestJS única contendrá todos los módulos (controladores, servicios) que implementan la lógica de negocio.
- **Amazon SQS (Message Queue):** Se utiliza para desacoplar tareas pesadas o que no necesitan ser instantáneas. Por ejemplo, cuando un usuario publica una reseña, la API puede guardar la reseña en la DB (síncrona) y luego enviar un mensaje a SQS para que el **Servicio Estadísticas** recalcule el promedio del libro (asíncrono), asegurando que la API responda rápido.

C. Capa de Datos y Almacenamiento

Esta capa gestiona la persistencia de los datos de la aplicación.

- **Amazon RDS (PostgreSQL):** Se utiliza una instancia de Amazon Relational Database Service (RDS) con el motor **PostgreSQL**. RDS es un servicio administrado que maneja backups, parches y replicación, permitiendo enfocarse en el diseño del esquema de datos en lugar de en la administración de la base de datos.
- **Amazon S3 (Datos):** Se usa un segundo bucket de S3, separado del frontend, para almacenar archivos y contenido generado por el usuario, como fotos de perfil o portadas de libros, tal como se menciona en el proceso de "Gestión de Perfil".

D. Servicios Externos y Notificaciones

Estos son componentes de los que depende tu aplicación para funcionar.

- **Amazon SES (Simple Email Service):** Utilizado por el backend para enviar correos transaccionales, como la verificación de cuenta y la recuperación de contraseña.
- **Proveedores OAuth (Externos):** El **Servicio de Autenticación** se comunicará con Google y Apple para validar los tokens de inicio de sesión social.
- **APIs de Libros (Externas):** El **Servicio Libro** y **Buscador** consultarán APIs externas como Google Books para enriquecer el catálogo de libros.

5. Relational Database Model

<https://dbdiagram.io/d/RelationalDatabaseModel-El-Parche-Lector-690c199a6735e111707a2Odd>

