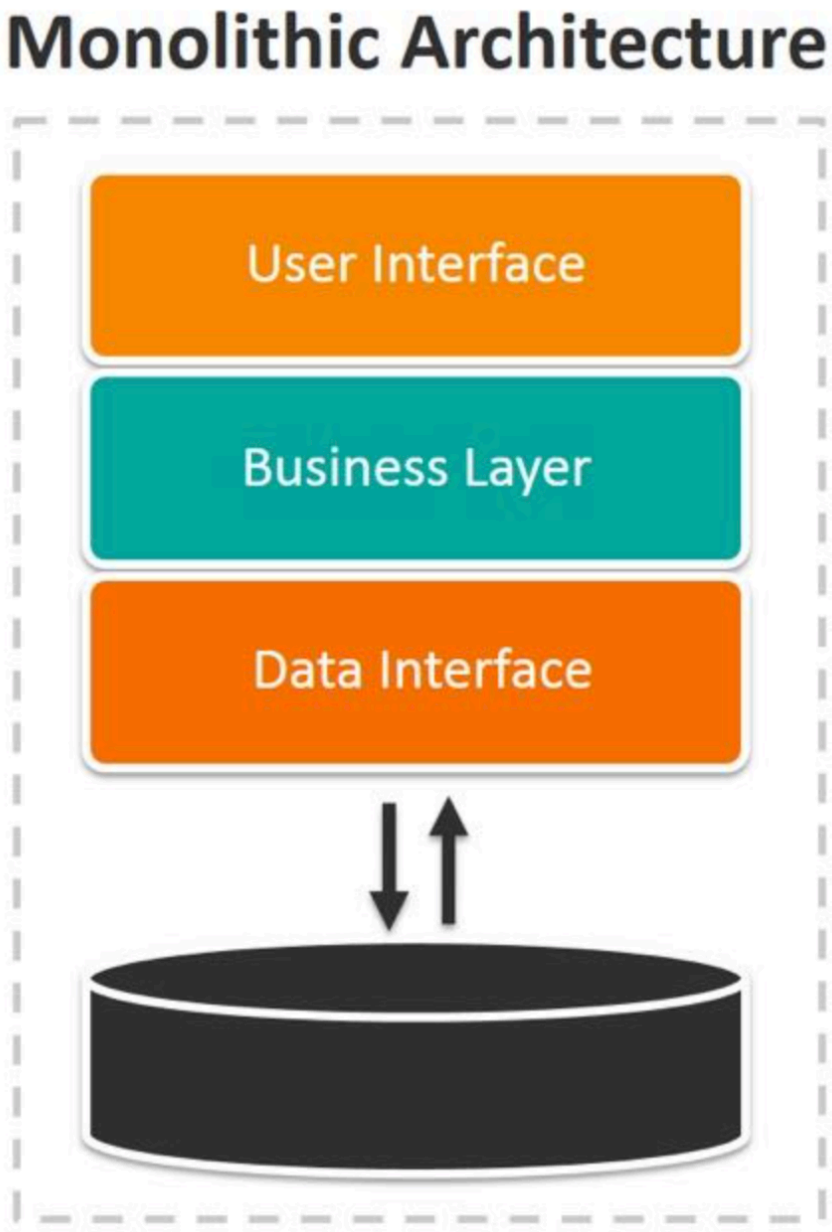
15 Casos Monolíticos para Actividad ✅ Actividad Preparatoria:

"Detectives del Monolito: Analiza, Divide y Propón"



🔧 CASOS MONOLÍTICOS PARA LA ACTIVIDAD

🎓 Caso 1: EduConnect - Plataforma educativa centralizada

Descripción del sistema:
EduConnect es una plataforma monolítica para colegios y liceos que reúne gestión de notas, comunicaciones con apoderados, chat interno, calendario escolar y evaluaciones online.

Problemas actuales:

- El sistema tiene tiempos de carga lentos durante la semana de evaluaciones.
- Cada vez que se actualiza el sistema de notas, se cae también el chat interno.
- La base de datos es compartida entre todos los módulos.

Problemas éticos detectados:

- Docentes pueden acceder a toda la información de estudiantes aunque no sean sus alumnos.
- El sistema guarda historial de chats sin notificación a los usuarios.
- No existe un mecanismo claro para que un estudiante solicite eliminar su información.

Datos técnicos:

- Aplicación PHP con base de datos MySQL.
- Hospedada en un servidor on-premise.
- Sin logs ni auditoría de accesos.

Desafíos a resolver:

- Rediseñar la arquitectura para que cada módulo funcione de forma independiente.
- Diseñar un sistema de roles y permisos para docentes.
- Incorporar herramientas modernas de gestión de identidades y logs de auditoría.

💼 Caso 2: MediBase - Sistema hospitalario regional

Descripción del sistema:
MediBase es un software centralizado que gestiona pacientes, recetas,

exámenes, hospitalizaciones y horarios médicos para hospitales y consultorios de una región.

Problemas actuales:

- Cualquier funcionario con acceso puede ver el historial completo de todos los pacientes.
- No hay separación entre módulos: los reportes de laboratorio están alojados en el mismo backend que la facturación.
- El sistema se cae cuando hay más de 500 usuarios concurrentes.

Problemas éticos detectados:

- Violación de la confidencialidad de datos médicos.
- No se solicita consentimiento explícito para compartir los datos entre especialidades.
- Se han detectado casos de uso indebido de datos por personal administrativo.

Datos técnicos:

- Sistema .NET monolítico.
- Base de datos SQL Server.
- Sin cifrado de datos sensibles.

Desafíos a resolver:

- Diseñar un sistema de microservicios por área funcional: pacientes, laboratorios, facturación, etc.
- Aplicar principios de confidencialidad (modelo CIA).
- Incorporar cifrado en reposo y en tránsito.

🏠 Caso 3: CityPlan - Gestión urbana y reclamos ciudadanos

Descripción del sistema:
CityPlan es una plataforma monolítica de gobiernos locales para gestionar

reclamos ciudadanos, solicitudes de obras, agenda municipal, pagos de permisos y licencias.

Problemas actuales:

- El sistema prioriza reclamos por "volumen histórico", lo que invisibiliza barrios de menor participación.
- Usuarios pueden ver el estado de cualquier reclamo, incluso de otros vecinos.
- Todos los datos se procesan en una sola base de datos.

Problemas éticos detectados:

- Inequidad territorial digital.
- Falta de anonimato en denuncias ciudadanas.
- Asignación desigual de recursos por criterios algorítmicos opacos.

Datos técnicos:

- Java Spring Boot monolítico.
- Sistema legado con base de datos Oracle.
- No tiene APIs ni integraciones externas.

Desafíos a resolver:

- Crear servicios independientes para reclamos, pagos, agenda y licencias.
- Implementar algoritmos de prioridad transparentes.
- Diseñar interfaces de acceso con autenticación y autorización robustas.

⌚ Caso 4: FitLife - Plataforma de salud y bienestar

Descripción del sistema:
FitLife ofrece seguimiento de actividad física, dieta, ciclos de sueño y acceso a entrenadores personales. Todos los datos están centralizados y accesibles para el equipo de marketing.

Problemas actuales:

- Datos de salud (peso, IMC, horas de sueño) son compartidos internamente sin consentimiento.
- El sistema falla con frecuencia al sincronizar con dispositivos externos.
- Una sola falla en el backend interrumpe toda la plataforma.

Problemas éticos detectados:

- Monetización encubierta de datos personales.
- Falta de transparencia en el uso de los datos.
- Riesgo de discriminación por perfiles de salud.

Datos técnicos:

- Aplicación Node.js monolítica.
- Almacenamiento en MongoDB sin reglas de acceso.
- Sin cumplimiento de normativas tipo GDPR.

Desafíos a resolver:

- Segmentar el sistema en servicios de usuarios, salud, dispositivos y recomendaciones.
- Implementar consentimiento explícito y dashboard de privacidad.
- Aplicar políticas de acceso por roles (RBAC).

⚙️ Caso 5: JobMatch - Plataforma de búsqueda laboral

Descripción del sistema:
JobMatch es una plataforma monolítica donde empresas publican ofertas laborales y los usuarios postulan con un perfil. Cuenta con un motor de recomendaciones automático que filtra candidatos según el "historial de contratación exitoso".

Problemas actuales:

- Las recomendaciones tienden a excluir candidatos sin experiencia previa o de regiones rurales.

- Las postulaciones se almacenan en una sola tabla sin segmentación por empresa o tipo de cargo.
- El sistema se vuelve lento cuando más de 50 empresas utilizan filtros de búsqueda simultáneamente.

Problemas éticos detectados:

- Sesgo algorítmico contra perfiles no tradicionales.
- Falta de acceso equitativo a oportunidades.
- Filtros ocultos que penalizan a minorías sin justificación.

Datos técnicos:

- Plataforma Ruby on Rails monolítica.
- Base de datos PostgreSQL.
- Algoritmo de recomendación no auditable.

Desafíos a resolver:

- Separar los módulos de empresas, candidatos y motor de búsqueda.
- Diseñar un algoritmo justo y explicable.
- Crear filtros personalizables visibles para el usuario.

 Caso 6: SafeBank – Plataforma bancaria de servicios múltiples

Descripción del sistema:

SafeBank es una plataforma bancaria unificada que permite transferencias, pagos de servicios, revisión de saldo, solicitudes de crédito y gestión de inversiones. Todo se maneja desde una única aplicación central.

Problemas actuales:

- Cambios menores en un módulo afectan a todo el sistema.
- Los tiempos de respuesta son lentos los fines de mes.

- Fallas en el módulo de inversiones bloquean el acceso a transferencias y pagos.

Problemas éticos detectados:

- No se informa adecuadamente a los usuarios sobre cómo se usan sus datos financieros.
- No hay auditoría clara sobre accesos a datos sensibles.
- Se comparten comportamientos de consumo con terceros sin consentimiento informado.

Datos técnicos:

- Backend monolítico en Java.
- Base de datos relacional con más de 40 tablas interconectadas.
- Código acoplado, sin pruebas unitarias.

Desafíos a resolver:

- Separar los módulos financieros en microservicios autónomos.
- Implementar encriptación y auditoría detallada.
- Crear un centro de control de privacidad para usuarios.

📚 Caso 7: UniManager – Sistema de gestión universitaria

Descripción del sistema:

UniManager centraliza procesos académicos, administrativos y financieros de una universidad: matrículas, notas, pagos, carreras, horarios, etc.

Problemas actuales:

- Los estudiantes pueden ver notas de cursos que no están inscritos.
- Si un docente sube un archivo, puede reemplazar accidentalmente uno de otro curso.
- Se demora horas en cerrar el proceso de matrícula.

Problemas éticos detectados:

- Fallos de privacidad de datos académicos.
- Posible manipulación de evaluaciones.
- Falta de control de acceso segmentado.

Datos técnicos:

- Aplicación ASP.NET con arquitectura en capas, pero acoplada.
- Base de datos SQL Server sin respaldo automatizado.

Desafíos a resolver:

- Separar funciones en servicios: matrícula, evaluación, finanzas, usuarios.
- Asignar permisos diferenciados por rol y curso.
- Diseñar auditoría para todas las acciones académicas.

🛒 Caso 8: MarketOne – Plataforma de e-commerce multivendedor

Descripción del sistema:

MarketOne permite a vendedores subir productos, gestionar envíos y recibir pagos. Los compradores pueden buscar, comparar y comprar en un solo flujo. Todo se ejecuta en una única app con código compartido.

Problemas actuales:

- Una falla en el módulo de pagos deja inactiva toda la tienda.
- Todos los productos de todos los vendedores son gestionados en la misma tabla.
- Hay demoras frecuentes en días de alto tráfico (cyberdays).

Problemas éticos detectados:

- Recolección automática de datos de navegación sin informar al usuario.
- Promoción de productos pagados sin aviso (publicidad encubierta).
- No hay canal claro de eliminación de datos personales.

Datos técnicos:

- Backend Node.js con Express monolítico.
- Base de datos MongoDB.

Desafíos a resolver:

- Separar servicios por dominios: catálogo, pagos, usuarios, pedidos.
- Implementar dashboards de privacidad para compradores y vendedores.
- Agregar logging y manejo de errores por módulo.

🎮 Caso 9: GameNet – Red social para gamers

Descripción del sistema:

GameNet permite crear perfiles de jugadores, registrar logros, publicar videos, participar en foros y torneos. Todo está centralizado en una aplicación web con funciones mezcladas.

Problemas actuales:

- Lentitud en el inicio de sesión cuando hay torneos activos.
- Publicaciones de un usuario se mezclan con logs internos del sistema.
- Bugs frecuentes por cambios cruzados entre módulos.

Problemas éticos detectados:

- Sin filtros para mensajes abusivos o discursos de odio.
- Cualquier moderador puede editar perfiles ajenos.
- No se informa sobre uso de datos de navegación y chat.

Datos técnicos:

- Aplicación PHP monolítica.
- Base de datos MySQL.
- Sin políticas de moderación algorítmica ni manual.

Desafíos a resolver:

- Dividir en microservicios: perfil, contenido, moderación, eventos.
- Incluir mecanismos de reporte y sanción automáticos.
- Diseñar filtros de lenguaje y protocolos éticos de comunidad.

🎬 Caso 10: CineFlex – Plataforma de streaming independiente

Descripción del sistema:

CineFlex ofrece películas y series bajo demanda. Todo el backend se encuentra en un solo servidor que maneja el catálogo, el sistema de pagos, las cuentas de usuario y la entrega de video.

Problemas actuales:

- Una sobrecarga en el streaming afecta el acceso al sistema de usuarios.
- El catálogo no se actualiza correctamente si hay un fallo en el proceso de pago.
- No hay redundancia ni balanceo de carga.

Problemas éticos detectados:

- Se rastrea el comportamiento del usuario sin informar (tracking de reproducciones y pausas).
- Algoritmos que priorizan contenido de ciertos estudios sin transparencia.
- No existe opción de descargar o eliminar datos de usuario.

Datos técnicos:

- Backend Python (Django) monolítico.
- Base de datos PostgreSQL.
- Almacenamiento de videos en un NAS local.

Desafíos a resolver:

- Crear microservicios independientes: video, pagos, usuarios, catálogos.
- Añadir políticas de consentimiento y privacidad visibles.

- Implementar streaming adaptativo separado del backend general.

🏥 Caso 11: ClinicaPlus – Sistema de gestión clínica privada

Descripción del sistema:

ClinicaPlus centraliza agendas médicas, historiales clínicos, gestión de recetas, pagos y fichas administrativas en un solo sistema usado por distintas clínicas privadas.

Problemas actuales:

- Toda la información de pacientes está en una única base de datos sin separación por clínica o especialidad.
- No existen controles sobre qué profesional puede acceder a qué ficha médica.
- El sistema se cae cuando se hacen procesos de facturación masiva.

Problemas éticos detectados:

- Exposición indebida de información médica sensible.
- Posible cruce de información entre clínicas competidoras.
- No se aplica consentimiento informado sobre almacenamiento y uso de datos.

Datos técnicos:

- Sistema Java EE monolítico.
- Base de datos Oracle compartida.
- Sin bitácora de accesos.

Desafíos a resolver:

- Diseñar microservicios por dominio funcional: pacientes, recetas, pagos, agendas.

- Separar datos por clínica o especialidad médica.
- Aplicar reglas estrictas de control de acceso (RBAC).

🏨 Caso 12: ReserHotel – Plataforma de reservas hoteleras

Descripción del sistema:

ReserHotel permite reservar habitaciones, revisar disponibilidad, gestionar pagos, y realizar comentarios. Todo se encuentra integrado en un único backend.

Problemas actuales:

- Una reserva errónea bloquea la disponibilidad general del hotel.
- Las evaluaciones y comentarios pueden ser editadas por los administradores sin dejar rastro.
- El sistema permite ver datos de reserva de otros usuarios por error de permisos.

Problemas éticos detectados:

- Vulneración de la privacidad del historial de viajes.
- Posible manipulación de reputación por parte del hotel.
- Falta de trazabilidad en cambios de información.

Datos técnicos:

- Backend en PHP monolítico.
- Base de datos MySQL.
- Sin control de versiones ni backups diarios.

Desafíos a resolver:

- Crear microservicios para reservas, usuarios, pagos y comentarios.
- Diseñar un historial de auditoría de cambios.
- Implementar privacidad de datos por usuario.

📺 Caso 13: EduTV – Plataforma de contenidos educativos en video

Descripción del sistema:

EduTV ofrece lecciones en video, cursos en línea, evaluaciones automáticas y foros de discusión. Todo esto es gestionado desde un backend único para todos los cursos y usuarios.

Problemas actuales:

- Todos los videos están accesibles para cualquier usuario autenticado, sin segmentación por curso.
- El sistema de evaluaciones se cae cuando se publican nuevos contenidos.
- No hay separación entre funciones administrativas y académicas.

Problemas éticos detectados:

- Violación de propiedad intelectual de contenidos educativos.
- Evaluaciones automatizadas sin validación humana.
- Acceso no controlado a conversaciones de foro privadas.

Datos técnicos:

- Plataforma Python Django monolítica.
- Videos servidos desde almacenamiento local.

Desafíos a resolver:

- Separar módulos en servicios independientes: contenidos, usuarios, evaluaciones, foros.
- Aplicar permisos por tipo de usuario y curso.
- Incorporar trazabilidad y validación en calificaciones.

👨‍⚖️ Caso 14: LegalSoft – Sistema jurídico de gestión de casos

Descripción del sistema:

LegalSoft administra procesos judiciales, documentos, plazos, audiencias y notificaciones. Todo el flujo legal está contenido en un único software compartido por múltiples estudios jurídicos.

Problemas actuales:

- Acceso cruzado entre casos de distintos estudios por falta de separación de usuarios.
- Documentos confidenciales pueden ser abiertos sin restricciones.
- No hay encriptación de archivos sensibles.

Problemas éticos detectados:

- Violación de secreto profesional.
- Riesgo de corrupción o manipulación judicial.
- Uso no controlado de evidencia digital.

Datos técnicos:

- Sistema en Ruby monolítico.
- Base de datos PostgreSQL.

Desafíos a resolver:

- Separar servicios por funcionalidad (documentos, plazos, notificaciones).
- Implementar políticas estrictas de privacidad jurídica.
- Añadir cifrado punto a punto y registro de auditoría.

🧬 Caso 15: BioLabSys – Gestión de laboratorios de investigación

Descripción del sistema:

BioLabSys centraliza la gestión de muestras biológicas, protocolos de experimentos, resultados y reportes científicos en una única aplicación compartida por varias instituciones de investigación.

Problemas actuales:

- Muestras y resultados de distintos proyectos se almacenan juntos, sin clasificaciones por estudio o investigador.
- Cualquier investigador puede modificar resultados.
- Las actualizaciones del sistema deben hacerse en horario nocturno por la fragilidad del backend.

Problemas éticos detectados:

- Riesgo de manipulación o falsificación de datos.
- Falta de trazabilidad en resultados de alto impacto científico.
- Confusión entre estudios privados y públicos.

Datos técnicos:

- Backend en C# ASP.NET monolítico.
- Base de datos relacional SQL Server.

Desafíos a resolver:

- Separar servicios por áreas: muestras, protocolos, resultados, usuarios.
- Control de versiones y trazabilidad de modificaciones.
- Establecer políticas de ética científica y control de acceso por rol.