I. Evaluación General de Expertise

¿Cuántos años de experiencia tienes programando profesionalmente?
Tengo tres años de experiencia trabajando en proyectos full-stack para empresas de distintos tamaños, abarcando desde mantenimiento de aplicaciones existentes hasta el diseño e implementación de nuevas funcionalidades críticas.

¿Cuál ha sido el proyecto más desafiante en el que has trabajado y cuál fue tu rol?
El mayor reto que he enfrentado ha sido en el desarrollo de la plataforma para Granja El Ceibillal, donde necesitábamos una aplicación móvil capaz de gestionar inventarios de insumos, control de alimentación, registros de mortalidad y natalidad de la granja avícola. Mi rol abarcó toda la cadena: desde la fase de diseño de la arquitectura hasta la implementación, pruebas y despliegue.

Tuvimos que garantizar que la app funcionara en entornos con conectividad intermitente, de modo que cada dispositivo guardara localmente todos los datos hasta que el trabajador recuperara señal; para ello empleé React Native con una capa de almacenamiento offline y desarrollo de sincronización diferida. Además, se optimizó la aplicación para que fuera muy ligera reduciendo al mínimo las dependencias y el uso de recursos de modo que corriera de forma fluida en teléfonos de gama baja.

En el backend diseñé y desplegué una API en Flask que validaba, almacenaba y sincronizaba la información con la base de datos en AWS cuando fuera posible. Coordiné las pruebas de campo con el equipo de operaciones y ajusté mecanismos de reintento y colas locales para evitar pérdida de datos. Gracias a esta solución integral, conseguimos un control fiable del ciclo de vida de los animales y optimizamos la gestión de insumos, reduciendo significativamente pérdidas y retrasos operativos.

¿Cómo te mantienes actualizado sobre nuevas tecnologías?
Dedico dos horas semanales a leer blogs especializados y documentación oficial, complemento con cursos cortos y aportes en proyectos open-source, y participo en comunidades técnicas como Discord para discutir novedades y buenas prácticas. También hago cursos prácticos en mi tiempo libre y veo vídeos relacionados a tecnología.

¿Has trabajado en equipos multidisciplinarios? ¿Cómo manejas la colaboración?
Sí, he colaborado con diseñadores, QA, product managers, equipos de marketing y de recursos humanos. Utilizo Jira para seguimiento de tareas, Figma para prototipos y establezco meetings diarios breves y revisiones de código constantes, garantizando que todos estén alineados y puedan detectar bloqueos rápidamente.

¿Cómo manejas situaciones en las que no tienes experiencia previa con una herramienta requerida?
Abordo primero la documentación oficial y ejemplos básicos, luego construyo un prototipo mínimo viable para entender su flujo, y si persisten dudas recurro a foros o colegas con experiencia, reduciendo así la curva de aprendizaje sin detener al equipo.

II. Frontend: React, Next.js, TypeScript, JavaScript + Redux

¿Qué ventajas ofrece Next.js respecto a CRA (Create React App)?
Next.js añade renderizado híbrido (SSR, SSG), enrutamiento basado en archivos y optimizaciones automáticas de performance, mejorando SEO y tiempos de carga sin configuración adicional.

¿Cómo organizas tu estructura de carpetas en una aplicación Next.js?
Suelo usar /pages para rutas, /components para componentes reutilizables, /styles para estilos, /hooks para lógica compartida y /lib para utilidades, manteniendo la separación clara de responsabilidades.

¿Cómo tiparías correctamente los props en un componente funcional?
Defino una interfaz o type en TypeScript para el tipo de datos que se espera de los props y lo pasaría de esta manera

interface ButtonProps { onClick: () => void; label: string; }
const Button: React.FC<ButtonProps> = ({ onClick, label }) => { … };

Así podría asegurar que el compilador de TS alerte sobre usos incorrectos.

¿Cómo manejarías el estado global de una app sin Redux?
Empleo Context de React con useReducer, creando un proveedor que expone el estado y el dispatch, logrando un flujo similar a Redux con menos dependencias.

¿En qué casos usarías JavaScript puro en lugar de React?
Cuando necesito un script aislado o una manipulación DOM muy ligera (por ejemplo, un widget sencillo), para evitar la sobrecarga de bundle que implica React.

¿Qué es un manejador de estado y cómo funciona?
Es una herramienta para centralizar el estado de la aplicación; suele consistir en un store que, al actualizarse, emite eventos a los componentes suscritos, garantizando un flujo de datos predecible.

III. Firebase

¿Cómo configurarías reglas de seguridad en Firestore para usuarios con diferentes roles?
Defino reglas en firestore.rules y reglas específicas para cada colección según rol, asegurando que solo usuarios autorizados accedan a datos sensibles.

¿Cuál es tu experiencia utilizando Firebase Authentication?
He implementado login con correo/contraseña y proveedores sociales (Google, Facebook), integrado el flujo de verificación de correo y restauración de contraseña, y validado tokens en mi backend para autorizar rutas protegidas para una aplicación de ejercicios.

¿Has utilizado Firebase Storage y Functions? Describe un caso real.

Sí: al subir imágenes de perfil en Storage, una Function generaba automáticamente versiones reducidas (thumbnails) y las almacenaba, notificando al frontend mediante eventos de Cloud Functions.

¿Cómo manejarías una estructura de base de datos escalable para una app con usuarios, productos y órdenes?
Creo colecciones separadas (users/{uid}, products/{id}, orders/{orderId}), en órdenes embebo referencias o datos clave de productos y usuario, y uso subcolecciones para historial de cambios, junto con índices compuestos para consultas eficientes.

IV. Shopify (Liquid + Integraciones JS)

¿Qué es Liquid y cómo funciona en Shopify?
Liquid es un lenguaje de plantillas que se ejecuta en el servidor para convertir etiquetas y filtros en HTML dinámico, permitiendo mostrar datos de productos, colecciones y configuraciones de tema.

¿Cómo modificarías una sección reutilizable en un tema de Shopify?
Editaría el archivo .liquid de la sección, ajustaría su schema para exponer opciones configurable en el editor de temas y actualizaría el markup y estilos asegurando compatibilidad con futuras versiones.

¿Cómo inyectarías funcionalidades externas con JavaScript en una tienda Shopify?
Incluyo scripts personalizados en theme.liquid o en plantillas específicas, y utilizo DOMContentLoaded para inicializar librerías de terceros sin bloquear el renderizado principal.

¿Qué experiencia tienes con personalización de checkouts o carritos?
He usado Shopify Script Editor para descuentos dinámicos basados en valor de carrito y la AJAX API de storefront para actualizar el carrito y totales en tiempo real, mejorando la experiencia de usuario.

V. Backend con Node.js

¿Cómo estructuras un proyecto Node.js que escale bien?
Divido en /controllers, /services, /models, /routes y /middleware, centralizando la configuración en un app.js o server.js para mantener el código modular y fácil de testar.

¿Has creado middleware personalizado? ¿Puedes dar un ejemplo?
Sí: un middleware de Express que extrae y verifica un token JWT del header, añade el usuario validado a req.user y retorna un 401 si el token es inválido.

¿Qué ORM has usado en tus proyectos? ¿Qué ventajas encuentras en uno como Prisma?
He trabajado con Sequelize y Prisma; Prisma destaca por su esquema tipado en TypeScript, generación automática de cliente y migraciones sencillas, lo que reduce errores de discrepancia con la BD.

¿Cómo implementarías autenticación y autorización en una API REST?
Validaría credenciales en un endpoint de login para emitir un JWT; luego protejo rutas con un middleware que verifica el token y comprueba los roles o permisos necesarios antes de continuar.

¿Cómo manejarías errores globales en una aplicación Express o Nest.js?
Defino un errorHandler central que captura excepciones, registra detalles en logs y envía una respuesta JSON uniforme con código y mensaje, evitando repetir lógica en cada ruta.

VI. Buenas Prácticas y Testing

¿Cómo aseguras la calidad del código que entregas?
Sigo principios SOLID, uso linters (ESLint) y formateadores (Prettier), revisiones de pull request obligatorias y escaneo estático de dependencias en CI para prevenir errores y vulnerabilidades.

¿Has implementado pruebas unitarias o de integración? ¿Qué librerías usas?
Sí: utilizo Jest, Vitest o Mocha + Chai, para tests unitarios y Supertest para pruebas de integración de endpoints REST, ejecutándolas en cada build para detectar regresiones.

¿Cómo configuras un pipeline de CI/CD básico para despliegue continuo?
En GitHub Actions configuro el workflow para que se dispare en cada push o pull request sobre la rama main y defino tres jobs secuenciales: uno de lint (checkout del código, setup de Node.js, cache de dependencias y ejecución de npm run lint), otro de tests (pasos similares pero con npm test) y, si ambos pasan y estamos en main, un job de deploy que carga los secretos desde los GitHub Secrets, autentica con la plataforma (por ejemplo AWS o Docker) y ejecuta un script de despliegue; de este modo cualquier cambio queda validado y solo lo que pase ambas comprobaciones llega automáticamente a staging o producción.

¿Qué estrategias sigues para mantener seguridad en aplicaciones web?
Valido y sanito entradas para evitar XSS e inyección SQL, uso HTTPS, configuro headers seguros , gestiono secretos con variables de entorno o Vault, y realizo auditorías periódicas de dependencias.