**Guía de Entrevista Técnica**

1. Evaluación General de Expertise:

✅ **¿Cuántos años de experiencia tienes programando profesionalmente?**

Tengo alrededor de **cuatro años de experiencia profesional**, trabajando en el desarrollo de sistemas internos, aplicaciones móviles y herramientas web, tanto en el sector público como privado.

**✅ ¿Cuál ha sido el proyecto más desafiante en el que has trabajado y cuál fue tu rol?**

Uno de los más desafiantes fue en **Polyproductos**, donde desarrollé una **app móvil para generar certificados** y otra para organizar bodegas en Guatemala y Nicaragua. Yo fui el encargado del desarrollo completo, desde el diseño hasta la implementación, y también daba soporte a los usuarios finales.

**✅ ¿Cómo te mantienes actualizado sobre nuevas tecnologías?**

Normalmente veo cursos en línea, leo documentación oficial y me gusta probar cosas nuevas por mi cuenta. Por ejemplo, estoy aprendiendo **Angular y Flutter** en mi tiempo libre para seguir creciendo.

**✅ ¿Has trabajado en equipos multidisciplinarios? ¿Cómo manejas la colaboración?**

Sí, he trabajado con personal técnico, usuarios finales y áreas administrativas. Me gusta colaborar siendo claro, escuchando, y buscando soluciones en conjunto. Si hay algo técnico, trato de explicarlo sencillo para que todos podamos entendernos.

**✅ ¿Cómo manejas situaciones en las que no tienes experiencia previa con una herramienta requerida?**

Primero investigo, busco la documentación y si es necesario veo algún tutorial rápido. Me adapto rápido porque me gusta resolver y no me quedo trabado esperando que alguien me lo diga. Prefiero intentarlo y aprender en el camino.

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

**✅ ¿Qué ventajas ofrece Next.js respecto a CRA (Create React App)?**

Next.js ya viene con muchas cosas listas como **renderizado del lado del servidor (SSR)**, **ruteo automático**, y buena performance. En cambio, CRA es más básico, es solo para aplicaciones que se renderizan en el cliente.

**✅ ¿Cómo organizas tu estructura de carpetas en una aplicación Next.js?**

Uso la carpeta /pages para las rutas, /components para los componentes reutilizables, /styles si manejo CSS o módulos, y a veces /hooks o /utils para lógica personalizada. Me gusta que todo esté bien ordenado para trabajar en equipo.

✅ **¿Cómo tiparías correctamente los props en un componente funcional?**

Uso **TypeScript** y creo una interfaz para los props.

**✅ ¿Cómo manejarías el estado global de una app sin Redux?**

Podría usar el **Context API de React**, que funciona bien para pasar datos a muchos componentes. También se puede usar algún manejador liviano como **Zustand** si no quiero tanta configuración.

**✅ ¿En qué casos usarías JavaScript puro en lugar de React?**

Cuando el proyecto es muy simple, como una página estática o algo que no necesita estado ni componentes.

**✅ ¿Qué es un manejador de estado y cómo funciona?**

Es una herramienta para controlar y compartir datos entre componentes. Ayuda a que el estado esté centralizado y sea más fácil de mantener.

III. Firebase

**✅ ¿Cómo configurarías reglas de seguridad en Firestore para usuarios con diferentes roles?**

Uso las **reglas de seguridad de Firestore** con condiciones basadas en los campos del usuario autenticado.

**✅ ¿Cuál es tu experiencia utilizando Firebase Authentication?**

He usado Firebase Auth para registrar y autenticar usuarios con **correo/contraseña**, y también para proteger rutas. Me gusta porque es fácil de implementar y se conecta bien con Firestore y otras herramientas.

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

Sí. En un proyecto subíamos **imágenes de productos** a Firebase Storage desde el frontend, y con las **Cloud Functions** hacíamos validaciones o guardábamos logs automáticos cuando se subía un archivo. Me sirvió mucho para automatizar tareas.

**✅ ¿Cómo manejarías una estructura de base de datos escalable para una app con usuarios, productos y órdenes?**

Usaría **subcolecciones** y documentos bien separados para evitar lecturas innecesarias.

IV. Shopify (Liquid + Integraciones JS)

**✅ ¿Qué es Liquid y cómo funciona en Shopify?**

Liquid es el lenguaje de plantillas que usa Shopify para construir temas. Es parecido a HTML, pero con variables, condicionales y bucles. Se usa para mostrar productos, colecciones o datos del carrito en las vistas.

**✅ ¿Cómo modificarías una sección reutilizable en un tema de Shopify?**

Editar una sección se hace desde un archivo .liquid dentro del tema, normalmente en la carpeta /sections. Ahí puedes modificar el HTML y usar variables de Shopify para que sea dinámica.

**✅ ¿Cómo inyectarías funcionalidades externas con JavaScript en una tienda Shopify?**

Puedes usar etiquetas <script> en archivos .liquid, o usar archivos .js personalizados en la carpeta de assets. También se puede usar Shopify ScriptTag API si es una app externa.

**✅ ¿Qué experiencia tienes con personalización de checkouts o carritos?**

Aún no he trabajado directamente con la personalización del checkout de Shopify, pero tengo experiencia integrando carritos y flujos similares en otras plataformas.

V. Backend con Node.js

**✅¿Cómo estructuras un proyecto Node.js que escale bien?**

Organizo por capas: /routes, /controllers, /services, /models, y /middlewares. Así el código está separado y es fácil de mantener.

**✅¿Has creado middleware personalizado? ¿Puedes dar un ejemplo?**

Sí. Uno simple fue para verificar si el usuario está autenticado antes de permitir acceso a ciertas rutas. Revisaba el token y si no era válido, respondía con error.

**✅¿Qué ORM has usado en tus proyectos? ¿Qué ventajas encuentras en uno como Prisma?**

He trabajado más con Entity Framework en .NET, pero conozco Prisma por su simplicidad. Me gusta que tiene autocompletado, migraciones claras y buen manejo de relaciones.

**✅¿Cómo implementarías autenticación y autorización en una API REST?**

Usaría JWT. Al iniciar sesión, se genera el token. Luego, en cada petición protegida, un middleware lo valida. Para autorización, reviso el rol del usuario según la ruta.

**✅¿Cómo manejarías errores globales en una app Express o Nest.js?**

En Nest.js usaría filtros de excepciones con @Catch.

VI. Buenas Prácticas y Testing

**✅¿Cómo aseguras la calidad del código que entregas?**

Reviso bien mi código antes de subirlo, uso lint o extensiones como Prettier, y si es en equipo, prefiero usar pull requests y revisar entre compañeros.

**✅¿Has implementado pruebas unitarias o de integración? ¿Qué librerías usas?**

He hecho pruebas simples con Jest y también usando xUnit en .NET. No soy experto en testing todavía, pero sí entiendo su importancia y estoy aprendiendo más.

**✅¿Cómo configuras un pipeline de CI/CD básico para despliegue continuo?**

He visto configuraciones usando **GitHub Actions**, donde al hacer push a la rama principal se corre el build y se despliega automáticamente. Aún estoy aprendiendo a usarlo bien.

✅**¿Qué estrategias sigues para mantener seguridad en aplicaciones web?**

Valido todos los inputs del usuario, uso HTTPS, manejo bien los permisos y evito exponer datos sensibles. También uso autenticación segura y tokens con expiración.