Prueba Técnica Full Stack

Duración máxima: 2 días

Objetivo:

Desarrollar una solución completa que incluya una API basada en microservicios y una interfaz frontend que consuma sus datos.

Parte 1: Backend

Objetivo

Crear dos microservicios independientes que interactúen entre sí, siguiendo el estándar JSON API.

Requisitos Técnicos

- Utilizar el lenguaje de la vacante aplicada.
- Implementar JSON API (https://jsonapi.org/) para todas las respuestas.
- Usar Docker para containerizar los servicios.
- Seleccionar entre base de datos SQL, NoSQL o en memoria (con justificación).
- Implementar autenticación básica entre microservicios mediante API keys.
- Manejar errores y fallos en la comunicación con reintentos y timeouts.
- Documentar la API con Swagger/OpenAPI.

Microservicio 1: Productos

Gestionará un recurso llamado productos con los siguientes endpoints:

- **Crear un producto**.
- **Obtener un producto por ID**.
- **Actualizar un producto por ID**.
- **Eliminar un producto por ID**.
- **Listar todos los productos con paginación**.

Microservicio 2: Inventario

Gestionará un recurso inventarios y permitirá:

- **Consultar la cantidad de un producto** llamando al microservicio de productos.
- **Actualizar la cantidad disponible tras una compra**.
- **Emitir un evento básico cuando cambia el inventario** (puede ser un log en consola).

Pruebas Backend

Implementa pruebas unitarias y de integración que cubran:

- Creación y actualización de productos.
- Comunicación entre microservicios.
- Manejo de errores como producto no encontrado o fallos en la API.

Parte 2: Frontend

Objetivo

Desarrollar una interfaz web que consuma la API creada en la parte backend.

Requisitos Técnicos

- Usar el framework de la vacante (React, Angular, Vue, etc.).
- Implementar una interfaz visual limpia y funcional.
- Manejar errores de API y estados de carga.
- Versionar el código con Git.
- Implementar pruebas unitarias básicas.

Funcionalidades

- **Listar productos disponibles con paginación**.
- **Consultar detalles de un producto específico**.
- **Mostrar la cantidad disponible de un producto desde el microservicio de inventario**.
- **Actualizar la cantidad disponible tras una compra**.

Pruebas Frontend

- Pruebas unitarias de los componentes clave.
- Simulación de fallos en la API y validación del manejo de errores.

Expectativas según Seniority

Junior

✅ Backend funcional con Docker básico.

✅ Frontend funcional con diseño simple.

✅ Pruebas unitarias mínimas (40% backend, 30% frontend).

✅ Documentación básica (README con instrucciones).

Mid-Level

✅ Backend estructurado con buenas prácticas y Docker Compose.

✅ Frontend responsivo y bien estructurado.

✅ Buen manejo de errores y logs.

✅ Pruebas unitarias con 60% de cobertura.

✅ Documentación completa con diagramas de arquitectura y componentes.

Senior

✅ Backend robusto con autenticación entre servicios y logs estructurados.

✅ Frontend avanzado con manejo óptimo de estados.

✅ 80% de cobertura en pruebas.

✅ Documentación exhaustiva y guía técnica para futuros desarrolladores.

Líder Técnico

✅ Todo lo anterior más patrones de diseño bien implementados.

✅ Estrategia de versionado de API.

✅ Propuesta de mejoras para escalabilidad futura.

Entrega

- Plazo: **2 días** desde la recepción de la prueba.
- Comparte un repositorio público con:
- Código fuente bien estructurado.
- README con instrucciones de instalación y ejecución.
- Justificación de decisiones técnicas.
- Diagramas de arquitectura y flujo de datos.
- Pruebas unitarias e integración documentadas.

Nota Final

La calidad del código, la organización del proyecto y la documentación son más importantes que la cantidad de funcionalidades implementadas. ¡Prioriza buenas prácticas y entrega un producto bien estructurado! 🚀