Examen práctico de Analista de Desarrollo

Gerencia de Innovación

Nombre aspirante: Denis Angelino Gudiel Yanes

Teléfono: 5317-3668

**Contestar las siguientes preguntas:**

1. **Bases de datos:**
   1. ¿Qué hace un left join?

Un LEFT JOIN combina dos tablas y devuelve todas las filas de la tabla izquierda, junto con las filas coincidentes de la tabla derecha.

* 1. ¿Qué hace un inner join?

Un INNER JOIN combina dos tablas y devuelve únicamente las filas donde hay coincidencias en ambas tablas

* 1. ¿Cuál es la diferencia entre el cast y convert?

En que el CAST se usa para convertir datos de un tipo a otro y el CONVERT se usa cuando se necesita formatos específicos.

* 1. ¿Para qué sirve un índice?

Sirve para mejorar el rendimiento de las consultas que se realizan en la base de datos.

* 1. ¿Para qué sirve un Store Procedure?

Es un conjunto de instrucciones SQL precompiladas que se almacenan del lado del servidor de base de datos.

* 1. ¿Cuál es la diferencia entre un índice Clusteriado y no clusteriado?

En que Clusteriado ordena físicamente los datos ideal para búsquedas frecuentes por ID principal y no clusteriado mantiene los datos sin ordenarlos físicamente ideal para consultas especificas.

* 1. ¿Para qué sirve una tabla temporal y como se declara?

Sirve para manejar datos de manera eficiente durante operaciones complejas o temporales. En Sql Server se declara así

CREATE TABLE #TempTable (

Id INT,

Name VARCHAR(50),

CreatedDate DATETIME

);

1.8 ¿Para qué sirve una tabla doble temporal y como se declara?

Sirven para escenarios multiusuario(base de datos) para compartir datos temporales de forma controlada entre distintas sesiones. En Sql Server se declara así

CREATE TABLE ##GlobalTempTable (

Id INT PRIMARY KEY,

Name NVARCHAR(100),

CreatedDate DATETIME

);

* 1. ¿Para qué sirve la función set dateformat dmy?

Sirve para asegurar la interpretación del formato mes, día y año.

* 1. Escriba la sintaxis de un cursor:

DECLARE nombre\_cursor CURSOR FOR SELECT columna\_tabla FROM nombre\_tabla

OPEN nombre\_cursor

FETCH NEXT FROM nombre\_cursor INTO @variable\_columna\_tabla

WHILE @@FETCH\_STATUS = 0 BEGIN

-- OPERACIONES A REALZAR CON @variable\_columna\_tabla

FETCH NEXT FROM nombre\_cursor INTO @variable\_columna\_tabla

END

CLOSE nombre\_cursor

DEALLOCATE nombre\_cursor

* 1. ¿Diferencia entre un truncate table y un drop table?

En que el TRUNCATE sirve para eliminar todos los datos de la tabla y conservar la tabla para seguir utilizando y DROP elimina la tabla físicamente de la base de datos.

* 1. ¿Indique una herramienta para ver el performance de un gestor de base de datos y que encontramos?

Para Sql server se puede usar la herramienta **“SQL Server Profiler”** de Microsoft.

2 **Backend:**

2.1 Explica cómo implementarías una estrategia de cacheo a nivel de aplicación sin utilizar servicios externos (Se puede apoyar de una base de datos relacional como Postgresql o sql server al ser un escenario común donde no se cuenta con la disponibilidad de usar servicios de alguna base de datos dictionary in memory como Redis):

Dado lo mencionado se puede implementar una estructura de cache en memoria a nivel de aplicación. Por ejemplo para el ámbito en seguridad para Autenticación de dos pasos(OTP) en general se utilizan 6 dígitos para para permitirle acceso al usuario en un inicio de sesión, el problema aquí es que no se recomienda guardar el código(OTP) en la base de datos por que hay que encriptarlo y desencriptarlo y se pierde tiempo y rendimiento en recursos, lo ideal es implementar un mapa en memoria con los parámetros (email y código) para poder identificar a cada usuario, porque lo más seguro es memoria. Recordando que cada registro en cache es volátil, dependiendo el tiempo configurado de caducidad esto para poder optimizar recursos del servidor. Si usamos .NET se puede usar la dependencia “IMemoryCache” nativa de Microsoft.

2.2 Explica el concepto de versionado de APIs y cómo lo implementarías.

Esta práctica se usa para gestionar cambios en una API sin interrumpir a los clientes que dependen de versiones anteriores. Es decir, que permite la evolución de la Api mientras mantiene versiones anteriores de forma controlada.

2.3 ¿Qué son los endpoints idempotentes y por qué son importantes?

Los endpoints idempotentes son aquellos sin importar cuantas veces se envíen los mismos parametros el resultado siempre será el mismo.

2.4 ¿Qué es la inyección SQL y cómo prevenirla?

Es una vulnerabilidad de seguridad cuando el atacante insertar código sql malicioso por medio de las consultas que realiza la aplicación, logrando el atacante la manipulación de la base de datos. Para prevenir este ataque se recomienda usar las mejores prácticas, como Validar la entrada de parámetros y parametrizar el tipo de dato de cada parámetro.

2.5 Explica cómo implementar autenticación y autorización en una aplicación backend (incluyendo como manejarías el almacenamiento seguro de contraseñas) para evaluar su conocimiento de encriptación simétrica y asimétrica.

Usaría JWT(Json Web Token) con el esquema de autorización “BEARER”, cada token se debe de identificar con un identificar único que relacione al usuario para que pueda realizar las consultas respectivas a la base de datos. Para el cifrado de contraseñas usaría un Hash + Salt es un cifrado Unidireccional (es decir que no se puede descifrar por que no se usan llaves publicas y privadas). En este caso usaría el cifrado de 128 bits(SHA512). La contraseña se cifra en el Backend y luego se guarda cifrada en la base de datos. Y por ultimo para iniciar sesión la contraseña que ingrese el usuario se cifra en el backend para luego compararla con ambos cifrados, si todo esta bien se inicia sesión con éxito sin estar descifrando la contraseña.

2.6 ¿Cómo identificarías y resolverías un cuello de botella en una aplicación backend?

A nivel de código revisaría si se está usando programación asíncrona ya que si no se usa se estaría desperdiciando recursos por lo bloqueos de hilos(procesos) ya que la programación asíncrona permite procesos simultáneos.

2.7Explica las diferencias entre concurrencia y paralelismo y cómo afectan el diseño de aplicaciones backend.

La concurrencia permite multiples tareas de entrada y salida y el paralelismo permite multiples tareas en diferentes nucleos del CPU.

2.8 ¿Cómo manejarías la sincronización de hilos en un entorno multihilo para evitar condiciones de carrera?

\* Usando bloqueos (Lock, RLock) para proteger secciones críticas.

\* Emplea mecanismos avanzados como semáforos, barreras y condiciones según el caso.

\* Diseña cuidadosamente para minimizar bloqueos y evitar deadlocks.

2.9 ¿Cuáles son los problemas asociados con el uso de números de punto flotante en aplicaciones financieras? Proporcione ejemplos específicos.

Errores de Precisión, Problemas de Redondeo, Pérdida de Exactitud en Comparaciones y Problemas de Redondeo Internacional.

2.10 ¿Cómo afecta el uso excesivo de bloques try-catch y la implementación de recursión profunda al rendimiento de una aplicación backend? En tu respuesta, analiza el impacto en el uso de la CPU y la memoria RAM, y explica los conceptos de ciencias de la computación que sustentan estos efectos.

* El incremento del uso de CPU, cada excepción genera una pila de ejecución y requiere procesamiento adicional.
* Aumentar el uso de memoria RAM, cada llamada recursiva agrega un marco en la pila de ejecución.

1. **Desarrollo web y mobile**
   1. ¿Cuáles son las principales diferencias entre el Renderizado del Lado del Servidor (SSR) y el Renderizado del Lado del Cliente (CSR)? Analiza sus ventajas y desventajas en términos de rendimiento, SEO y experiencia de usuario.

En que en SSR el contenido HTML se genera en el servidor y se envía completamente al cliente. Y para el CSR el contenido HTML inicial es mínimo; el navegador ejecuta JavaScript para generar la página.

* 1. Explica qué es el Critical Rendering Path en el contexto del desarrollo frontend y describe las técnicas avanzadas que utilizarías para optimizarlo, reduciendo el tiempo de carga percibido por el usuario.

Es el proceso que el navegador sigue para convertir HTML, CSS y JavaScript en una página visualizada y funcional. Hay que comprimir archivos, imágenes o cualquier otro tipo de archivo para mejorar el rendimiento de carga.

* 1. Describe cómo implementarías un middleware personalizado en cualquier gestor de estados (Redux o Zustand) para manejar efectos secundarios complejos, como llamadas a APIs externas y gestión de autenticación, asegurando que el flujo de datos permanezca predecible y mantenible.

Para implementar un middleware personalizado en Redux, creas una función que intercepta acciones específicas (como llamadas a APIs), realiza los efectos secundarios necesarios (fetch, autenticación, etc.), y despacha nuevas acciones para manejar los resultados (éxito o error).

* 1. Explica las diferencias entre REST y GraphQL en el contexto del desarrollo frontend y describe cómo implementarías una solución híbrida que aproveche las ventajas de ambos enfoques.

REST ofrece endpoints fijos para operaciones estándar, mientras que GraphQL permite consultas dinámicas y personalizadas desde un único endpoint.

* 1. Explica los conceptos de Progressive Enhancement y Graceful Degradation en el desarrollo frontend y cómo los aplicarías para asegurar una experiencia de usuario consistente en diferentes navegadores y dispositivos.

Progressive Enhancement construye aplicaciones comenzando con funcionalidades básicas que funcionan en todos los navegadores, añadiendo características avanzadas para entornos modernos. Graceful Degradation parte de una versión completa, asegurando que funciones esenciales permanezcan disponibles en navegadores menos capaces.

* 1. Explica la importancia de la accesibilidad en el desarrollo y describe las mejores prácticas para asegurar que una aplicación web o mobile sea accesible para todos los usuarios, incluyendo aquellos con discapacidades o poca educación digital (usuarios de escasos recursos con dispositivos demasiado antiguos y con los que les cuesta interactuar).

La accesibilidad asegura que aplicaciones web o móviles sean utilizables por todos, incluyendo personas con discapacidades o usuarios con recursos limitados. Mejores prácticas incluyen: usar etiquetas semánticas, proporcionar navegación por teclado, agregar descripciones en imágenes (alt), contrastes adecuados, y diseñar interfaces simples y ligeras, compatibles con dispositivos antiguos y conexiones lentas.

* 1. Describe las Core Web Vitals (LCP, FID y CLS) y explica cómo optimizarías una aplicación frontend para cumplir con estos indicadores de rendimiento.

LCP (Largest Contentful Paint): Mide la velocidad de carga del contenido principal. Optimízalo cargando imágenes más rápido, usando lazy loading y preconexión a recursos.

FID (First Input Delay): Mide la interactividad inicial. Mejorarlo implica reducir el tamaño de los scripts y delegar tareas pesadas al backend.

CLS (Cumulative Layout Shift): Mide la estabilidad visual. Minimízalo reservando espacio para imágenes y evitando cambios dinámicos inesperados en el diseño.

* 1. En el contexto de desarrollar un chat frontend que interactúa con una inteligencia artificial (IA) a través de una API de un modelo de lenguaje grande (LLM), ¿qué tecnología utilizarías para manejar la comunicación en tiempo real: WebSockets o Server-Sent Events (SSE)? Detalla los beneficios de la tecnología seleccionada y explica cómo gestionarías la conexión, el envío y la recepción de mensajes, así como la gestión de eventos de conexión y desconexión, considerando que se trata de un chat con una IA y no entre dos personas reales.
  2. Chat en Tiempo Real, Manejo de Estado Offline y Seguridad en Flutter

En el contexto del desarrollo de una aplicación de chat en tiempo real con Flutter que debe funcionar tanto online como offline, y considerando los siguientes gestores de estado disponibles:

* BLoC (Business Logic Component)
* Riverpod

Por el manejo de estados asíncronos.

* Provider
* MobX
* GetX

A) Seleccione un gestor de estado y justifique detalladamente su elección basándose en:

* Manejo de estados asíncronos

Riverpod tiene un manejo nativo y fluido de estados asíncronos con AsyncValue, facilitando la implementación de tareas como la sincronización online/offline.

* Complejidad de implementación

Aunque requiere cierta curva de aprendizaje inicial, su diseño declarativo lo hace más sencillo de usar en proyectos complejos comparado con otras opciones como BLoC.

* Testabilidad

Riverpod facilita la inyección de dependencias, haciendo que las pruebas unitarias sean simples y desacopladas.

* Mantenibilidad

Su enfoque modular y la separación de lógica de negocio de la interfaz favorecen un código limpio y escalable.

* Rendimiento con actualizaciones frecuentes

Riverpod actualiza solo los widgets dependientes, optimizando el rendimiento en aplicaciones con cambios de estado continuos.

* Facilidad para manejar side effects

El StateNotifier y Provider permiten manejar efectos secundarios de manera estructurada y segura.

* Seguridad en el manejo de datos sensibles

Riverpod permite integrar prácticas seguras, como controlar accesos a datos sensibles mediante estados protegidos.

B) Con el gestor de estado seleccionado, explique cómo implementaría:

* La arquitectura general de la solución.

Modelo MVU (Model-View-Update): La lógica de negocio reside en StateNotifier y se inyecta a través de Provider en los widgets.

Capas:

Datos (Data Layer): Repositorios que manejan la base de datos local y la API remota.

Estado (State Layer): Gestión con StateNotifier y AsyncValue para estados de conexión, mensajes, y conflictos.

UI Layer: Componentes que reaccionan a cambios de estado usando Consumer.

* La sincronización de mensajes online/offline.

Usar un repositorio que detecte el estado de conexión (con connectivity\_plus) y sincronice automáticamente los mensajes almacenados en caché con el servidor remoto cuando la conexión esté disponible.

* El manejo de estados de conexión.

Crear un StateNotifier que escuche el estado de red y emita valores (online/offline), permitiendo que la UI se actualice dinámicamente.

* La estrategia de caché local.

Utilizar Hive o SQLite para almacenar mensajes localmente. Se asigna un estado de sincronización (syncPending) a los mensajes que aún no han sido enviados.

* La gestión de conflictos en mensajes editados.

Implementar un algoritmo de resolución basado en marcas de tiempo (Last Write Wins) y sincronizar los cambios con el servidor al reconectarse.

C) Explique las consideraciones y estrategias de seguridad para:

* Almacenamiento seguro de mensajes localmente (encryption at rest)

Usar AES-256 para cifrar mensajes en repositorios locales (con librerías como flutter\_secure\_storage).

* Protección de datos en tránsito (E2E encryption)

Implementar E2E encryption con bibliotecas de cifrado como libsignal.

* Manejo seguro de tokens y credenciales.

Almacenar tokens en el almacenamiento seguro del dispositivo (Keychain en iOS, Keystore en Android).

* Prevención de ataques de replay.

Añadir marcas de tiempo y nonces únicos a los mensajes enviados.

* Validación de integridad de mensajes.

Utilizar firmas digitales (HMAC) para verificar la autenticidad de los mensajes.

* Protección contra inyección de código malicioso.

Sanitizar toda entrada de usuario usando validadores robustos.

* Manejo de permisos y autenticación biométrica.

Integrar autenticación biométrica para acceder a los mensajes sensibles.

* Sanitización de datos de entrada/salida.

Normalizar y validar datos antes de almacenarlos o procesarlos.

D) ¿Qué consideraciones de rendimiento tendría en cuenta para:

* Listas largas de mensajes.

Usar widgets como ListView.builder con carga diferida (lazy loading) para manejar grandes volúmenes de datos.

* Actualizaciones frecuentes de estado.

Optimizar los Provider para emitir actualizaciones solo a widgets específicos.

* Manejo de memoria en caché de imágenes/archivos.

Usar cached\_network\_image para optimizar el almacenamiento y la carga de imágenes.

* Impacto del cifrado en el rendimiento.

Realizar el cifrado/descifrado en un hilo secundario para evitar bloqueos de la UI.

* Balance entre seguridad y experiencia de usuario

Priorizar operaciones en segundo plano para que la seguridad no degrade la fluidez de la aplicación.

E) ¿Cómo implementaría:

* Borrado seguro de mensajes/conversaciones

Sobrescribir datos antes de eliminar registros locales (data wiping).

* Backups cifrados de conversaciones

Implementar copias de seguridad cifradas usando AES-256 en almacenamiento seguro.

* Detección de manipulación de datos locales

Implementar validación con checksums o hashes.

* Revocación de acceso y gestión de sesiones.

Invalidar tokens en el servidor al cerrar sesión o revocar acceso.

* Logs seguros para auditoria.

Registrar eventos clave en logs cifrados con acceso restringido.

**No es necesario escribir el código, sino explicar la estrategia y estructura que utilizaría, los ejemplos de las partes más críticas es opcional, detallar los porqués de cada decision.**

1. **Prueba Practica**

**2.1 UNICAMENTE REACT y .NET (MVC o MINIMAL APIS)**

* En una empresa dedicada a la venta de MARISCOS desea llevar el control de inventario sobre cada uno de sus productos, los productos se venden por libra, o medía libra, debe considerar que el vendedor puede armar sus combos a precios accesibles a sus clientes, al vender un combo este se debe descontar del inventario de los productos específicos y el reporte debe salir como venta de combo con su respectivo precio.
* Debe tener la opción de realizar las ventas y el registro de productos nuevos o existentes.
* Adicional debe mostrar las ventas realizadas sea por libre media libra o combos
* Debe considerar los diferentes tipos de empaques, porque existen cajas de 4 o 3 libras y el sistema debe ser lo suficiente inteligente para calcular todo por libras existentes.
* Debe mostrar el listado de productos existentes
* Y mostrar alertas cuando un producto esta debajo de 5 libras en existencia.
* Debe mostrar las ventas por día y por mes, y las ganancias de la misma forma.

**Puntos extras (opcional):**

**Se otorgarán puntos extra si realizas lo siguiente:**

1. **Grabar un video corto usando la herramienta React Scan para analizar el desarrollo**
2. **Presentar un análisis de los resultados obtenidos por React Scan**
3. **Explicar de manera conceptual (sin código) cómo abordarías las mejoras sugeridas por la herramienta**

**No es necesario:**

* **Implementar las correcciones**
* **Mostrar código**
* **Hacer cambios en la aplicación**

**El enfoque debe estar en:**

* **La interpretación de los resultados del análisis**
* **La explicación clara de las estrategias de mejora**
* **El razonamiento detrás de las soluciones propuestas**

**Esta sección es completamente opcional y complementa la evaluación principal.**

**2.2 UNICAMEMENTE FLUTTER**

* Muestre el ejercicio anterior en el teléfono móvil,
* Con todas las funciones, pero adicional debe implementar la opción de agregar sucursales y que como usuario pueda entrar a gestionar las sucursales y ver los respectivos reporte
* Adicional puede registrar las ventas también por medio del móvil.
* Las push se deben enviar cuando un producto está debajo de la cantidad definida.

**IMPORTANTE: se estará calificando diseño, seguridad, rendimiento.**

**NOTA: Tiene 3 día para realizar la prueba.**

**FECHA MÁXIMA DE ENTREGA 14 DE DICIEMBRE A LA 11 AM**

Enviar un video, documentación y adjuntar su proyecto a los siguientes correos

[esantos@genesisempresarial.com](mailto:esantos@genesisempresarial.com)

[cfernandez@genesisempresarial.com](mailto:cfernandez@genesisempresarial.com)

[tdeleon@genesisempresarial.com](mailto:tdeleon@genesisempresarial.com)

aagonzalez@genesisempresarial.com

**DUDAS WHATSAPP: 33568980**