TECHNICAL ASSESMENT

FULL-STACK ENGINEER

8 DE ENERO DE 2026

RAUL ANTONIO BATRES PEREZ
INGENIERO

Sección 1: Resolución de problemas

Problema 1

Se tiene una base de datos relacional ficticia la cual fue desarrollada para albergar una cantidad media a baja de datos, el uso del sistema fue más alto de lo esperado y se está teniendo una afluencia masiva de datos, el sistema se está volviendo lento y se notan tiempos de respuesta elevado. Tu trabajo es dar un approach inicial para lograr revertir esta situación, ¿qué recomendaciones darías y cual sería tu plan de trabajo?

Consideraciones:

- Base de datos está normalizada
- Base de datos está un motor actualizado
- Las consultas con el backend se hacen a través de un ORM

Respuesta 1

Antes de empezar a optimizar y a desmembrar código, lo cual podría ocasionar más problemas, realizaría una evaluación con respecto a las querys más frecuentes utilizadas y de igual forma cuales son las querys más lentas para tenerlas como primera alternativa de mejora.

Comenzaría a revisar dichas consultas para poder saber la lógica aplicada, pueda que alguna de las consultas retorne demasiados registros, esto es algo que se da con mucha frecuencia. Utilizaría la ejecución EXPLAIN para ver donde la consulta gasta más tiempo. Realizaría los cambios solicitando solo las columnas que se necesiten para evitar utilizar un SELECT *. Si es posible configuraría los LIMIT para retornar una cantidad adecuada de datos. Si es posible realizara SPs para poder almacenar las consultas frecuentes para mejorar el rendimiento.

En el caso de utilización de ORM realizaría Lazy Loading controlados y paginaciones para tener un mejor control de los registros que estarían siendo retornados. Hiciera uso de cache en Backend con Redis ya que es uno de los más utilizados por los desarrolladores

Problema 2

Tienes una página con muchos componentes que hacen fetch() al cargar. El rendimiento es bajo.

- ¿Cómo lo solucionarías?
- ¿Qué técnicas o tecnologías usarías?
- ¿Qué herramientas usarías para detectar los cuellos de botella?

Respuesta 2

Muchas veces los componentes necesitan algún fetch para poder obtener datos y trabajar. Al momento de realizar el montaje de los componentes este sería un gran problema de optimización. Peor desde mi punto de vista el problema no es el fetch sino más bien cuando y cuantas veces se hace el llamado. Para solucionarlo de manera rápida utilizaría un Hook de react bastante famoso llamado useCallback para evitar los renderizados indebidos y de esta manera realizar el llamado de un fetch solo una vez. Como solución más avanzada podría crear un lazy loading de componentes el cual se encargaría de cargar los componentes que solamente estén presentes en la pantalla. También utilizaría como una solución avanzada la herramienta llamada Tanstack React Query para una gestión eficiente de caché y estado para evitar llamadas repetidas innecesarias. Con respecto a las herramientas la más popular que me brinda información para verificar todos los renderizados de componentes en mi Frontend es React DevTools.

Problema 3

Sin hacer uso de compiladores, ¿qué resultado da el algoritmo de acá abajo?
function fibonacci(n: number): number {

```
if (n <= 1) return n;

let prev = 0;

let curr =1;

for (let i = 2; i <= n; i++) {

const next = prev + curr;

prev = curr;

curr = next;
```

```
 }
 return curr;
}
```

Respuesta 3

La secuencia Fibonacci es aquella que cada número es la suma de los 2 anteriores, en el algoritmo se muestra como inicio los valores de prev = 0 y curr = 1 como podemos observar dentro del For se recorren los valores y se realiza la suma de prev + curr y se va desplazando sin usar recursividad lo cual me parece bastante bien desde mi punto de vista porque esta versión con un bucle For es más eficiente que la forma recursiva.

N	Fibonacci
0	0
1	1
2	1
3	2
4	3
5	5

Problema 4

Se tiene un módulo interactivo en un sistema de educación el cual ayuda a la población no vidente generando notas de voz para que el estudiante escuche la respuesta a los cuestionarios generados; sin embargo, las respuestas largas generan un costo importante ya que se usan demasiados tokens para generar el audio ¿Cómo optimizamos la generación de audios para reducir el costo de tokens al modelo de inteligencia artificial? ¿Qué cambios de arquitectura propondrías?

Problema 5

Se tiene un módulo donde dado un archivo .csv se busca una subida masiva de datos, del lado del backend, considerando que los registros son dependientes entre sí

- ¿Cómo manejarías la subida masiva de estos datos?
- ¿Qué haces antes de las excepciones ocurridas?
- ¿Si un registro falla después de haber procesado 10,000 registros como lo manejarías?

Respuesta 5

Para darle solución a este problema realizaría lo siguiente con estos pasos:

1. Primero es la subida del archivo
2. Segundo aplico validaciones y parseo todo lo necesario
3. Tercero realizo un procesamiento por lotes o conocido por batches
4. Cuarto reporto los resultados

Pienso que esta manera seria una forma óptima, empezaría leyendo el CSV y lo parto por chunks por ejemplo obtengo los primeros 5000 registros y seguiría de esa manera hasta completar el archivo. En el procesamiento validaría si el archivo CSV no contiene datos duplicados, si hay campos obligatorios vacíos, si existe algún carácter especial que se deba eliminar y también verificaría tipos de datos válidos.

Si en algún caso dado falla el procesamiento, mi idea sería manejarlo por batches para saber si fallaron los registros de 1 a 1000 que sería mi primer lote, por ejemplo, luego del 1001 al 2000 y así de manera sucesiva, eso me brinda la posibilidad de detener el proceso en el batch que se vaya recorriendo sin intervenir en los registros que pudieron estar buenos previamente.

Problema 6

En una aplicación web, tienes un formulario repetido en múltiples páginas. ¿Cómo lo harías reutilizable? Responde con una estrategia concreta.

Respuesta 6

Mi idea sería crear un componente único que no conozca la página donde vive y se adapte a cualquier llamado que se le realice.

Por ejemplo, crearía mi propio componente llamado MyForm de la siguiente manera:

```
<MyForm
  schema={schema}
  initialValues={initialValues}
  onSubmit={handleSubmit}
/>
```

Este recibiría 3 props una donde pueda ir alojado el esquema por ejemplo si se necesita que el form tenga solo 3 campos Nombre, Apellido, Edad. O si se desea en otra página se pueden agregar diferentes campos para eso me serviría mi prop llamada Schema. La segunda seria para colocar valores iniciales o por defecto y la tercera simplemente que reciba una función cuando el formulario haya sido clickeado.

Problema 7

¿Cómo estructurarías un frontend grande para que escale con varios equipos?

Respuesta 7

En mi caso me gusta utilizar la infraestructura con Atomic Design para crear Iones, Átomos, Moléculas, Organismos y sus respectivas páginas. Recomiendo esta infraestructura que es trabajosa en el sentido que se deben crear los componentes por pequeños que sean. Un simple input o un botón, por ejemplo. Esta infraestructura permite escalar de manera organizada y sin que el equipo se pierda en el camino. De igual manera me gusta trabajar por Features con la infraestructura Feature-based ya que también es bien organizada al momento que se vayan añadiendo nuevos Features al Frontend.

Problema 8

¿Cómo aplicarías principios de Clean Architecture en frontend? Menciona una solución en concreto

Respuesta 8

Desde una perspectiva teórica, yo aplicaría Clean Architecture en frontend entendiendo que no se trata de copiar el diagrama de backend, sino de preservar principios: separación de responsabilidades, inversión de dependencias y aislamiento del dominio frente al framework.

Se pueden tener carpetas mal nombradas y buena arquitectura, o carpetas perfectas con acoplamiento total. Lo importante es quién depende de quién**.**

Mi solución sería:

- Feature-based architecture como base
- Dominio por feature, no global
- Estado global mínimo
- Frameworks tratados como plugins

Pregunta 9

Se tiene el siguiente escenario, para el manejo de sesión de un sistema se está usando un sistema de tokens en donde dentro trae la información del usuario, almacenados en cookies con la siguiente configuración: (secure:true, sameSite: ‘lax’, maxAge: 1000*60*15, httpOnly:true) Menciona una manera de acceder a esta cookie en el frontend y como manejarías su decodificación para obtener la información del token y así poder imprimir la información del usuario

Respuesta 9

Desde una perspectiva correcta a nivel de seguridad, la respuesta clave es esta: Una cookie con httpOnly: true NO puede ni debe ser accedida desde el frontend. Y eso no es una limitación técnica accidental, es una decisión de diseño de seguridad.

Pregunta 10

¿Cómo asegurarías idempotencia en acciones del usuario en React?

Respuesta 10

Para mí la idempotencia en acciones de usuario en React se asegura modelando intenciones, no eventos.

El frontend debe:

- Tratar cada acción crítica como una intención única, no como un click
- Asociar esa intención a un identificador estable o key
- Evitar ejecuciones concurrentes mediante estado declarativo

El backend es quien garantiza la idempotencia real, mientras el frontend:

- Previene duplicaciones locales
- Mantiene consistencia visual
- Reintenta sin cambiar el resultado