Teste para dev C#

Perguntas técnicas (teoria prática)

Lógica / arquitetura

1- Você tem uma API lenta. Como investigaria o problema?
R: Analisaria logs, métricas de tempo de resposta, uso de CPU/memória, código mal otimizado (ex: Vários Includes no EF Core e retornando a classe inteira sem projetar um DTO, loops aplicando operações no Banco de Dados a cada interação).

2- Tenho um endpoint que é muito chamado. Ele não recebe parametros e retorna o mesmo resultado, mas a query no banco é pesada. Como melhorar isso?
R: Aplicaria uma solução com cache, no .NET eu aplicaria uma versão otimizada usando o HybridCache (Redis e ImMemory). Por padrão é consultado o ImMemory por ser mais rápido, caso não ache automaticamente busca no Redis.

3- Como garantir que uma operação crítica não seja executada duas vezes?
R: Usaria transações.

4- Como você lidaria com concorrência em atualização de dados?
R: Usaria transações.

5- O que você faria se um endpoint começar a receber 10x mais tráfego?
R: Otimizaria queries e utilizaria filas para processamento assíncrono como o broker RabbitMQ.

6- Tenho um endpoint dando erro de cors no Browser, mas funcionando pelo postman. Qual o motivo?
R: O navegador aplica política cors, já no Postman não. Provavelmente falta configurar CORS no servidor para a origem do front end.

.NET / C#

1- Qual a diferença entre Task, ValueTask e async void? Quando usar cada um?
- Task: Utilizada para operações assíncronas.
- ValueTask: Não utilizei com base nas minhas experiências.
- Async void: Não permite await.

2- Explique Dependency Injection no .NET Core.
– O que são Transient, Scoped e Singleton?

R: Dependency Injection é um padrão no .NET para injetar dependências evitando o acoplamento entre os objetos.

- Transient: Nova instância a cada solicitação.
- Scoped: Uma instância por escopo (ex: por request).
- Singleton: Uma única instância para toda a aplicação,seu valor muda apenas se a aplicação for restartada.

3- O que é Middleware no ASP.NET Core?
   R: Usado para interceptar requisições e respostas HTTP, podendo realizar uma série de fluxos antes de chegar ao Controller.

Dapper

1- Quando você escolheria Dapper em vez de EF Core?
   R: Quando precisa de performance máxima, controle total sobre SQL, ou em projetos simples pode ser uma boa abordagem.

2- Como evitar SQL Injection no Dapper?
   R: Usando parâmetros que se utiliza o caracter (@) em vez de concatenar strings. Ex:
   ("SELECT * FROM Users WHERE Id = @Id", new { Id }

REST API

1- Diferença entre PUT e PATCH
   - PUT: Substitui o objeto inteiro.
   - PATCH: Atualiza parcialmente o objeto.

2- Como versionar uma API?
   Pode ser versionada via URL ou via Header.

3- Cite códigos HTTP comuns e seus usos (200, 201, 400, 401, 403, 404, 500).
   - 200: OK – sucesso.
   - 201: Created – recurso criado.
   - 400: Bad Request – erro do cliente.
   - 401: Unauthorized – autenticação necessária.
   - 403: Forbidden – sem permissão.
   - 404: Not Found – recurso não existe.
   - 500: Internal Server Error – erro no servidor.

Banco de Dados

1- Diferença entre INNER JOIN e LEFT JOIN.
   R: Quando se aplica INNER JOIN você retorna somente os registros que estão em ambas as tabelas, já LEFT JOIN retorna todos os registros da tabela da esquerda mesmo não havendo correspondência na tabela da direita.

2- O que é índice? Quando pode piorar a performance?

R: Índice é uma estrutura que acelera consultas no banco. Pode piorar a performance quando há muitas escritas, pois cada insert, update ou delete precisa atualizar os índices, além de ocupar mais memória e espaço.

3- Diferença entre banco relacional e não relacional.

R: Bancos relacionais usam tabelas e relacionamentos, já bancos não relacionais usam modelos como documentos ou chave-valor.