TESTE TÉCNICO DEV SENIOR

Digitas B3

Sumário

Digitas B3

Escopo

Escreva um micro-serviço em netcore (net6 ou net7) utilizando as melhores práticas arquiteturais para esse tipo de aplicação e considerando que vai rodar em container Linux na cloud (AKS/EKS).

O que o serviço precisa fazer:

- Conectar com o websocket de Order Book público da Bitstamp: WebSocket API v2 - Bitstamp (olhar para a especificação **Live Order Book** e para o mecanismo de subscrição pública em **Subscriptions > Public channels**). Não é necessário criação de tokens ou qualquer outro cadastro.

- Manter a ingestão e tratamento dos dados de 2 instrumentos: BTC/USD e ETH/USD, persistindo os dados em alguma base NoSQL ou SQL para consultas

- A cada 5 segundos, printar na tela (pode ser no console mesmo, se conseguir alguma tela web ou Windows forms simples, melhor ainda):

    - Maior e menor preço de cada ativo naquele momento ○ Média de preço de cada ativo naquele momento ○ Média de preço de cada ativo acumulada nos últimos 5 segundos ○ Média de quantidade acumulada de cada ativo

- Expor uma API de simulação de melhor preço:
    - Deve ser informado a operação (compra ou venda), o instrumento e a quantidade ○ Calcular o melhor preço para a quantidade total, ou seja:
        - Se foi solicitado uma compra de 100 BTCs:
            - Ordenar todos os itens da coleção de “asks” do JSON contido na última atualização recebida em ordem crescente de preço.
            - Calcular o valor correspondente a 100 BTCs varrendo essa coleção e multiplicando quantidade x valor até chegar a 100 BTC. Nesse caso serão necessários, provavelmente, vários itens da coleção para cumprir a quantidade de 100 BTC dado que os itens isoladamente tendem a ser de pequena quantidade.
            - Caso a quantidade seja atendida já com o primeiro item da coleção, somente retornar o preço de quantidade desejada x valor.
        - Se for solicitado uma venda, fazer a mesma operação, mas na coleção de “bids” ordenados em ordem decrescente de preço.
    - O payload de retorno deve conter:
        - Um ID único que represente essa cotação
        - A coleção usada no cálculo (somente os itens que foram efetivamente utilizados)
        - A quantidade solicitada
        - O tipo de operação solicitado (compra/venda)
        - O resultado do cálculo ○ Gravar a memória de cálculo no banco de dados ○ A API de simulação não pode interferir na ingestão de dados sendo realizada

Detalhes

Prazo

Teste Técnico

Digitas B3

1 semana

Teste:

- Cobertura de 80% evitando de usar "ExcludeFromCodeCoverage"

Será avaliado:

- SOLID, KISS, YAGNI e DRY
- Conhecimento em DDD
- Design Pattern, motivo de usar ou não usar

Implementação

Fique à vontade para escolher frameworks e bibliotecas de aceleração.

Justifique suas escolhas arquiteturais e técnicas.

Pode rodar localmente em Docker na sua máquina. Porém, será um diferencial se você construir um pipeline simples em Azure DevOps e fizer o deploy em AKS (*) ou então em outra solução opensource que demonstre seu conhecimento em pipeline e deploy.

(*) Importante: caso deseje ir por esse caminho, favor utilizar contas gratuitas. Não haverá reembolso de valores em virtude do teste.

Apresentação

- Reunião de 1h.
    - 40 min: apresentação e demo
    - 20 min: Q&A
- Utilizar o Postman para demonstrar essa API
- Mostrar a base de dados ou por API no Postman as memórias de cálculo

Diferencial:

- Conhecimento em CosmosDB
- Experiencia com Testes não functionais
- Experiencia com Sistema escalável horizontalmente
- Experiencia com Baixa latencia