Teste para Engenheiro de Software

Instruções

1. Leia esse documento antes de iniciar as atividades.

2. Você tem 1 dia para entregar o plano de trabalho previsto (item 10), após finalização do trabalho atualizar o plano com a visão do trabalho realizado antes da entrevista.

3. Você tem até 7 dias corridos para concluir as atividades aqui solicitadas e apresentar o resultado na entrevista (O Resultado deverá ser enviando após os 7 dias, independente do ponto do desenvolvimento);

- Caso não consiga concluir todas as atividades, por favor, entregue o que foi feito até a data solicitada.

4. Fique à vontade para utilizar tecnologias, frameworks e técnicas não citadas nas atividades.

5. Recomendamos a utilização do docker (http://www.docker.com) para montagem do ambiente;

- Caso opte pela utilização do docker, publique os Dockerfiles no repositório do projeto ou deixe a imagem publicada no Dockerhub.

Atividades

1. Elabore um plano de trabalho.

2. Crie uma aplicação em uma linguagem de sua preferência para coletar as últimas postagens do Twitter, dada uma determinada #tag.
a. Por padrão o Twitter disponibiliza as 100 últimas postagens.
b. Caso não tenha 100 twittes, colete todas que vierem.
c. Não há necessidade de coletar mais do que 100 twittes, dada um #tag.

3. Modele e implemente uma base de dados para armazenar as informações.

4. Colete e armazene as mensagens, em uma base de dados da sua preferência, para as #tags listadas abaixo: #openbanking, #apifirst, #devops, cloudfirst, #microservices, #apigateway, #oauth, #swagger, #raml, #openapis

5. Utilizando uma linguagem de sua preferência, sumarize e grave os dados para conseguir listar as informações:
a. Quais são os 5 (cinco) usuários, da amostra coletada, que possuem mais seguidores?
b. Qual o total de postagens, agrupadas por hora do dia (independentemente da #hashtag)? c. Qual o total de postagens para cada uma das #tag por idioma/país do usuário que postou;

6. Crie uma API REST que permita o consumo dos três itens anteriores. A Api deverá expor métricas de execução.

7. Crie uma página uma linguagem de sua preferência que chame as API's e mostre os resultados.

8. **Muito importante 1:** Utilizando uma ferramenta de logging (exemplos: Elastic Search, Splunk, Graylog ou similar), crie uma query que mostre em tempo real os eventos que acontecem na execução da api criada no item 6, exemplos (Warning, Erro, Debug, Info, etc). Importante ter ao menos uma situação de execução com erro.

9. **Muito importante 2:** Utilizando uma ferramenta de métricas (exemplos: Prometheus, Zabbix, cloudwatch ou

similar), crie 3 dashboards que mostre em tempo real a quantidade de execução, a latência (tempo de execução) e quantidade de erros da api criada no item 6. Importante ter ao menos uma situação de execução com erro.

10. Plano de Trabalho (previsto e realizado)
a. Caso haja algum desvio entre o planejamento original e a execução, explique o que houve.
b. Caso o plano de trabalho foi seguido sem desvio, comente os motivos para esse resultado.

11. Publique o projeto no Github e documentar em um README.md os itens abaixo:

a. Documentação do projeto

b. Documentação das APIs

c. Documentação de arquitetura

d. Documentação de como podemos subir uma cópia deste ambiente localmente

e. Manual com prints dos Logs (item 8) e os 3 Dashboards (item 9) explicando cada um deles.

Peso utilizado na avaliação:

- Código: 20%

- Testes: 20%

- Logging: 20%

- Métricas: 20%

- Facilidade de Deploy (Item 10.d): 20%

Obs.: Para todos os itens iremos considerar a documentação como parte da entrega.

Importante: Trazer o case funcionando no dia da entrevista.