Olá Pabricio, bom dia, tudo bem?

Primeiramente agradecemos seu contato e interesse em trabalhar conosco. A primeira etapa do nosso processo seletivo consiste na realização de um teste prático de programação, segue abaixo:

**Premissas:**

Criar uma API(é um conjunto de normas que possibilita a comunicação entre plataformas através de uma série de padrões e protocolos.) usando Spring Boot, que faz parte da nossa stack.(Nosso conjunto de tecnologia)  
Utilizar um banco de dados em memória (HSQLDB ou H2)  
Não é necessário desenvolver front-end, apenas o back-end.

**Teste:**

Desenvolver um serviço que seja capaz de gerar uma venda.  
Uma venda é composta por id, data da venda, valor, vendedor id e vendedor nome.

**Sua tarefa é desenvolver os serviços REST(**Transferência de Estado Representacional**) abaixo:**

- Criar uma nova venda  
- Retornar a lista de vendedores com os campos: nome, total de vendas do vendedor e média de vendas diária, conforme o período informado por parâmetro

**Instruções:**

Gostaríamos de ver o histórico de seus commits.  
Vamos avaliar a maneira que você escreveu seu código, a solução apresentada e a cobertura de testes automatizados.  
Caso encontre algum impedimento no decorrer do desenvolvimento, entregue da maneira que preferir e faça uma explicação sobre o impedimento.  
Avaliaremos também sua postura, honestidade e a maneira que resolve problemas.  
Não é necessário implementar nenhum tipo de autenticação  
Ao finalizar subir o projeto no github e nos enviar o link por email.  
**Nos retornar dentro de 7 dias**, iremos analisar o projeto e agendar uma entrevista.

Atenciosamente,



Como compilar toda API(ela vai funcionar como banco de dados para ser acessado pelo seu serviço http)

Vai na pasta principal clica com direito>>>

Vai em Rus as

Vai em mavie build

Em goals: digite clean package

O arquivo será Gerado na pastar target

Como nome dado inicialmente a sua api .jar

Teste de implementaçãp

Jpa para ajuda cirar repositório

server.port=${port:8081}

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

spring.datasource.url=jdbc:h2:file:~/test

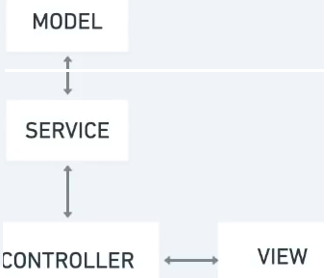
spring.datasource.username=sa

spring.datasource.password=

spring.datasource.driver-class-name=org.h2.Driver

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true



Model é a cara do objeto {cliente: nome endereço, idade}

**Service**: O comando que vai consulta o banco de dados;

**Controller**: Vai fazer as requisições de serviços

stacks= empilhar

@GetMapping("/{id}")

**public** Venda listarId(@PathVariable(value="id") **long** id) {

**return** vendaRepository.getById(id);

}

Venda getById();

=== (1dia(valor de vendas na quele dia) + 2dia + 3 dia) / quantidade de dia( primeiro dia até o ult dia)

List<lista> listas = **new** ArrayList<>();

lista actual= **new** lista(vendedorRepository.findAll().get(i));

listas.add(actual)

**for** (**int** r=0; r < 6; ++r) {

System.***out***.print(lista.get(r));

}

}

**for** (**int** r=0; r < 6; ++r) {

}

/\*

@GetMapping("/a")

public long contar( Cliente cliente) {

System.out.println("essseeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee"+clienteRepository.count());

return 11111;//Listando tudo que está dentro da tabela cliente

}\*/

**package** com.algaworks;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**public** **class** AlgacrmApi1Application {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(AlgacrmApi1Application.**class**, args);

}

}