Atividade
Testes da camada Web

Parte1: Utilizando o projeto base que estamos fazendo em aula, favor implementar os seguintes testes na classe ClientResourcesTests(opcional : utilizar o mockbean da classe ClientService) :

- **insert** deveria retornar “created” (código 201), bem como o produto criado, verifique no mínimo dois atributos.
  - (Opcional) será necessário programar a simulação do método insert da classe MockBean service. Não passe o objeto esperado e sim o método any(), que retorna uma simulação de algum objeto qualquer, assim evitamos alguns erros. Caso deseje passar um objeto DTO como parametro do método insert, você precisará criar os métodos equals e hash da classe DTO.

  - Como criar um Json no java (clientDTO é um objeto já instanciado):

```
// Junto com os atributos da classe de teste, faça a injeção de dependência abaixo
@Autowired
private ObjectMapper objectMapper;

// no método de teste insira os códigos de criação de JSON
String json = objectMapper.writeValueAsString(clientDTO);
ResultActions result =
                    mockMvc.perform(post("/clients/")
                    .content(json)
                    .contentType(MediaType.APPLICATION_JSON)
                    .accept(MediaType.APPLICATION_JSON));
```

- **delete** deveria
  - retornar “no content” (código 204) quando o id existir
  - retornar “not found” (código 404) quando o id não existir
  - (Opcional) A service simulada precisa retornar a exception not found. Utilizar o `doThrow` de modo similar ao teste realizado na service.
- **findByIncome** deveria
  - retornar OK (código 200), bem como os clientes que tenham o Income informado. Verificar se o Json Paginado tem a quantidade de clientes correta e se os clientes retornados são aqueles esperados. (similar ao exemplo feito em sala de aula).
  - Cuidado com os valores para teste, pois o delete apagou algum registro:
  - Código para passar o parâmetro income:

```
mockMvc.perform(get("/clients/income/")
          .param("income", String.valueOf(salarioResultado))
          .accept(MediaType.APPLICATION_JSON));
```

- **update** deveria
  - retornar “ok” (código 200), bem como o json do produto atualizado para um id existente, verifique no mínimo dois atributos. (similar ao insert, precisa passar o json modificado).
  - retornar “not found” (código 404) quando o id não existir. Fazer uma assertion para verificar no json de retorno se o campo “error” contém a string “Resource not found”.

Parte2: Opcional – criar a simulação (mockbean) da classe service.

Observações:

- Para a entrega me enviem o repositório no github.
- Vou deixar o link do projeto base que estamos usando em aula, porém nada impede que vocês usem outro projeto. Neste caso eu peço que vocês disponibilizem uma descrição básica do sistema e os endpoints daquele sistema.
- Link projeto base: https://github.com/brunoqp78/cliente-teste-aula

Data de entrega: **07 de julho de 2022**
Valor: **15 pontos**