# Atividade : testes em JPA Repository

**Implementar testes para algum projeto:**
**1 – Utilize algum projeto desenvolvido na disciplina Projeto Back-End.**

**2 – Utilize o projeto Cliente, disponível em : https://github.com/brunoqp78/cliente-teste-modelo.git**

**Cada aluno deverá criar no mínimo 4 testes. Abaixo algumas sugestões para o projeto Client.**

## Implementar funcionalidades na classe ClientRepository:

- criar um método personalizado para buscar um cliente pelo seu nome. (Sugestão: Aplicar a função LOWER)
  - Utilizando a clausula Lower o método se torna ignoreCase;
    - sintaxe: ……. Lower(campo)
    - Exemplo : "LOWER(obj.name) = LOWER(:nome)"
- criar um método personalizado para buscar os clientes pelo nome. (Sugestão: Aplicar a função LOWER)
  - Vocês deverão retornar uma List<Client> contendo todos os clientes que apresentam no nome a palavra passada como parâmetro ao método. (utilizar Like)
    - sintaxe: ……. campo LIKE %:parametro%
- criar métodos personalizados para buscar os clientes pelo salário. Criem três métodos:
    - um para buscar clientes com salários superiores a um valor.
    - um para buscar clientes com salários inferiores a um valor.
    - um para buscara clientes que tenham salários em uma determinada faixa de valores
  - Retornar uma List<Client> contendo todos os clientes que atendem a pesquisa cada uma das pesquisas.
- criar um método personalizado para buscar os clientes que tenham data de nascimento em uma determinada faixa de valores.
  - Não precisa criar a @Query;
  - necessário o método: List<Client> findClientBybirthDateBetween(Instant DataInicio, Instant DataTermino);

## Implemente novos testes na classe ClientRepositoryTests:

- Testar o método que retorna o cliente com nome existente;
  - Testar um nome existente;
  - Testar um nome não existente;
- Testar o método que retorna vários cliente com parte do nome similar ao texto informado;
  - Testar um texto existente;
  - Testar um texto não existente;
  - Testar find para nome vazio (Neste caso teria que retornar todos os clientes);
- Testar o método que retorna vários cliente baseado no salário;
  - Testar o método que busca clientes com salários superiores a um valor;
  - Testar o método que busca clientes com salários inferiores a um valor;

    - Testar o método que busca clientes com salários que esteja no intervalo entre dois valores informados.
- Testar o método que retorna vários cliente baseado na sua data de aniversário.
    - Teste o método buscando clientes que nasceram entre duas datas, sugestão uma data qualquer e a data atual. Códigos para criar uma classe que representa a data.
        - Instant dataI = Instant.*parse*("2017-12-25T20:30:50Z");
        - Instant dataT = Instant.*now*();
- Testar o update (save) de um cliente. Modifique o nome, o salário e o aniversário e utilize os métodos criados anteriormente para verificar se realmente foram modificados.

Observações:
● Fiquem à vontade para criar outros testes diferentes destes que foram sugeridos. Na dúvida, imaginem para aquele cenário que vocês estão testando, quais perguntas poderiam ser respondidas para validar determinado teste.
● Para a entrega me enviem o repositório no github ou o projeto compactado no classroom.
● 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 .

Valor: 20 pontos