**Boas práticas de construção de uma API rest.**

0 configure o applicationProperties para o seu banco de dados.

1 Crie uma model coloque todos os atributos necessários + get seters+ anotações

2 Teste sua aplicação (verifique se a tabela foi criada no mysql workbench)

3 Crie um repository e um controller com um método findAll

3 insira dados via o mysql workbench

4 teste o método findAll através do postman.

5 Crie um método findById no controller

6 Teste o método findByID via postman.

7 crie um método post no controller

8 teste o método post via postman.

9 crie um método put no controller.

10 teste o método put no postman.

11 crie um método delete no controller

12 teste o método delete no postman.

**Atividade CRUD Back-end minhaLojaDeGames.**

construa um back-end para uma rede social onde ele tenha a capacidade de manipular os dados dos Produtos e usuários da loja.

**Parte 1 criação da tabela Categoria.**

* Camada de model com o nome de **Categoria** com os seus atributos. (id, genero)
* Uma camada de repository com o nome **CategoriaRepository** (com a capacidade de se comunicar com o banco de dados mysql).
* Uma camada de Controller com o nome de **CategoriaController** Com 5 endpoints:
* findAllCategoria = um endPoint com a capacidade de trazer todas as categorias.
* findByIDCategoria = um endPoint com a função de trazer uma única categoria por id.
* findByDescricaoCategoria = um endPoint com a função de trazer uma categoria turma por Descricao.
* postCategoria = um endPoint com a função de gravar uma nova categoria no banco de dados.
* putCategoria = um endPoint com a função de atualizar dados de uma categoria.
* deleteCategoria = um endPoint com a função de apagar uma categoria do banco de dados).

**Parte 2 relacionamento com a tabela Produto.**

* Camada de model com o nome de **Produto** com os seus atributos. (id, titulo)

5. Crie um relacionamento de um para muitos/muitos para um, para essas models (uma categoria para muitos produtos e muitos produtos para uma categoria)

**IMPORTANTE: Não esqueça de colocar as anotações corretamente para o relacionamento e inibir recursividade através da anotação @JsonIgnoreProperties.**

* Uma camada de repository com o nome **ProdutoRepository** (com a capacidade de se comunicar com o banco de dados mysql).
* Uma camada de Controller com o nome de **ProdutoController** Com 5 endpoints:
* findAllProduto = um endPoint com a capacidade de trazer todos os Produtos .
* findByIDProduto = um endPoint com a função de trazer uma único Produto por id.
* findByDescricaoTitulo = um endPoint com a função de trazer um único Produto por Titulo.
* postProduto = um endPoint com a função de gravar um novo Produto no banco de dados.
* putProduto = um endPoint com a função de atualizar dados de um Produto .
* deleteProduto = um endPoint com a função de apagar um Produto do banco de dados).

**Parte 3 tabela Usuario.**

Camada de model com o nome de **Usuario** com os seus atributos. (id, nome)

Uma camada de repository com o nome **UsuarioRepository** (com a capacidade de se comunicar com o banco de dados mysql).

Uma camada de Controller com o nome de **UsuarioController** Com 5 endpoints:

* findAllUsuario = um endPoint com a capacidade de trazer todos os Usuarios .
* findByIDUsuario = um endPoint com a função de trazer uma único Usuariopor id.
* findByNome = um endPoint com a função de trazer um único Usuario por Nome.
* postUsuario = um endPoint com a função de gravar um novo Usuario no banco de dados.
* putUsuario = um endPoint com a função de atualizar dados de um Usuario .
* deleteUsuario = um endPoint com a função de apagar um Usuario do banco de dados).