

TESTE ESTÁGIO BACKEND V3

DOCUMENTAÇÃO DA APLICAÇÃO

FELIPE RODRIGUES

A listagem abaixo corresponde a todo conteúdo presente na realização da API requisitada.

Tecnologias: Java, Spring boot, Maven.

Dependências: Spring Data JPA, Spring Web, DevTools, Postgresql Driver e Lombok.

Para fins de efetuação de testes da API, foi utilizado a plataforma Postman, contendo todas as requisições necessárias para o CRUD:

Texto

Descrição gerada automaticamente

* Entidades, repositórios e controladores

Partindo para as entidades e códigos relacionados. Temos a classe Equipment definida através das anotações do spring @Entity, como uma entidade e @table como referência a tabela descrita no banco de dados. Existe também as anotações do lombok, @Getter e @Setter que fornecem a geração automática dos metodos get e set de cada atributo. Além destas, temos:

@Id – Definindo atributo id como chave primária desta entidade

@GeneratedValue – Gera um id automaticamente para o banco de dados, caso uma nova entidade seja criada.

@Column – referencia as colunas pré-existentes na tabela “equipment” da base de dados.

@ManyToOne e @ JoinColumn para definir chave estrangeira, “equipment\_model\_id”, da entidade “Equipment”.

Texto

Descrição gerada automaticamente

Repositório, que será utilizado para a implementação dos metodos de CRUD, sendo uma interface simples estendendo uma outra interface, CrudRepository, passando a entidade e o tipo de seu Id correspondente:

Tela de computador

Descrição gerada automaticamente

Por fim, a classe que representa o controle da entidade “Equipment”. Sendo mapeada para o container de “equipments”.

Além do mapeamento para recebimento de requisições Post (create), Get (read), Put (update), Delete (delete), que são equivalentes aos metodos doPost, doGet, doPut, doDelete, presentes em um servlet Java.

Texto

Descrição gerada automaticamente

@RequestBody – captura objeto Json enviado no corpo da requisição.

No metodo de update, é realizada a busca pelo id de um equipamento existente. Caso seja encontrado um registro deste equipamento, é realizada a atualização de nome dele. Caso não encontrado, é retornado um responseEntity informando o notFound.

O restante dos metodos de CRUD, de forma simples, sendo realizados por meio dos metodos padrões do repositório criado.

A logica para os outros componentes foi a mesma utilizada, com exceção para as entidades “Equipment\_model\_state\_hourly\_earnings” e “Equipment\_State\_History”, devido a serem classes onde suas chaves primárias são chaves compostas.

Texto

Descrição gerada automaticamente

Então nestas duas classes optei por utilizar o @EmbeddedId, que é uma notação para utilização de chaves compostas como chave primária. E a criação de uma classe extra representante deste ID.

Texto

Descrição gerada automaticamente

* Realização de Testes no Postman

O gif abaixo retrata a tabela equipment no banco de dados, e a realização da listagem no Postman, recuperando os dados da tabela em um formato json. Indicando o funcionamento do read.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Em seguida temos a criação de um novo equipamento “GB-0009”, demonstrando a tabela anteriormente e posteriormente a criação:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Seguindo para a alteração, deste mesmo equipamento “GB-0009”, mudando para seu nome para “Nome Alterado”:

Interface gráfica do usuário, Aplicativo, Tabela, Excel

Descrição gerada automaticamente

E para finalizar, o delete do equipamento:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

O teste foi realizado nas outras entidades, e obtiveram sucesso. Porém ocorreu uma exceção, que foi na criação de um novo “Equipment\_State\_History”, que utiliza uma estrutura um pouco diferente da maioria. O erro fornecido foi o seguinte:

"message": "Duplicate row was found and `ASSERT` was specified",

    "path": "/equipments\_state\_history/create"

A entidade Equipment\_MSHE utiliza a mesma estrutura e o “create” funcionou, então a única dúvida foi essa. Acho importante ser transparente, então deixo registrado meu único problema ao desenvolver.

* Conclusão

Todo o teste foi muito bem elaborado, gostei bastante e foi prazeroso de realiza-lo. Fico no aguardo de um possível retorno e agradeço a oportunidade de me incluírem nessa.