



TESTE ESTÁGIO BACKEND V3  
DOCUMENTAÇÃO DA APLICAÇÃO  
FELIPE RODRIGUES

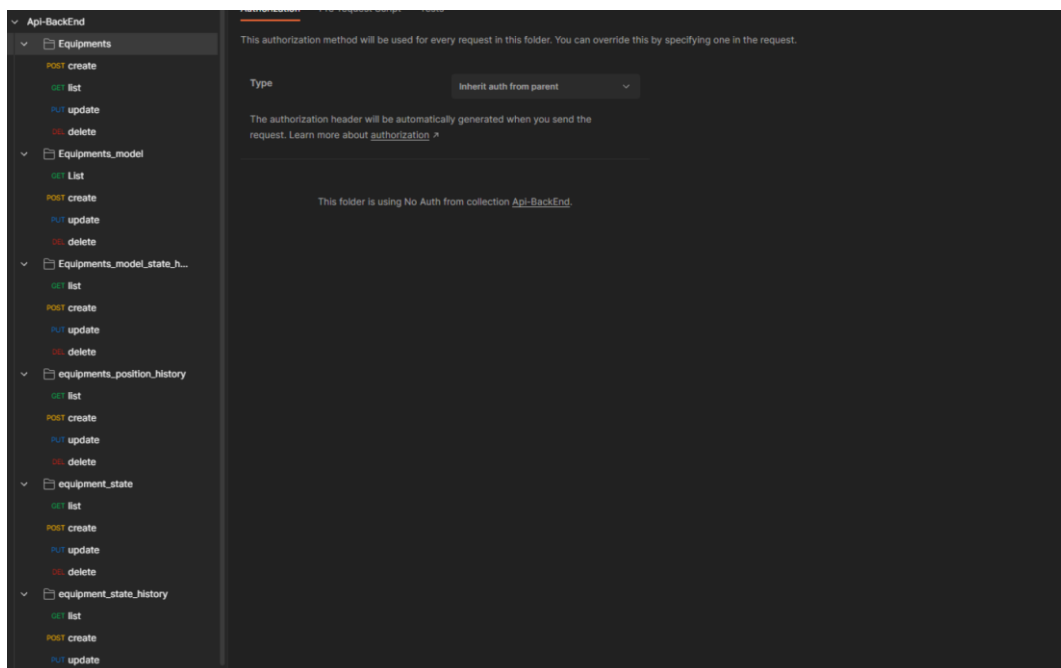
---

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:



- Entidades, repositórios e controladores

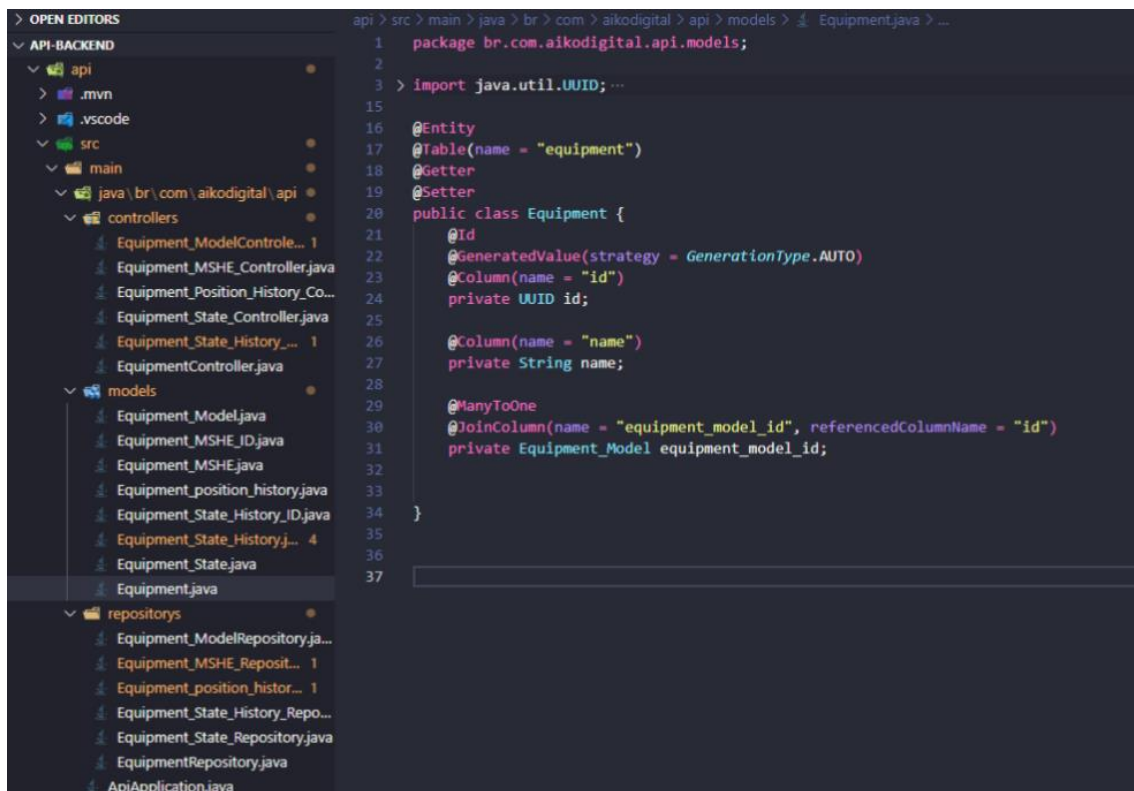
Partindo para as entidades e códigos relacionados. Temos a classe Equipment definida através das anotações do spring @Entity, como uma entidade e @table como referência a tabela descrita no banco de dados. Existe também as anotações do lombok, @Getter e @Setter que fornecem a geração automática dos metodos 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-existentis na tabela “equipment” da base de dados.

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



```
1 package br.com.aikodigital.api.models;
2
3 import java.util.UUID;
4
5
6
7
8
9
10
11
12
13
14
15
16 @Entity
17 @Table(name = "equipment")
18 @Getter
19 @Setter
20 public class Equipment {
21     @Id
22     @GeneratedValue(strategy = GenerationType.AUTO)
23     @Column(name = "id")
24     private UUID id;
25
26     @Column(name = "name")
27     private String name;
28
29     @ManyToOne
30     @JoinColumn(name = "equipment_model_id", referencedColumnName = "id")
31     private Equipment_Model equipment_model_id;
32
33
34 }
35
36
37
```

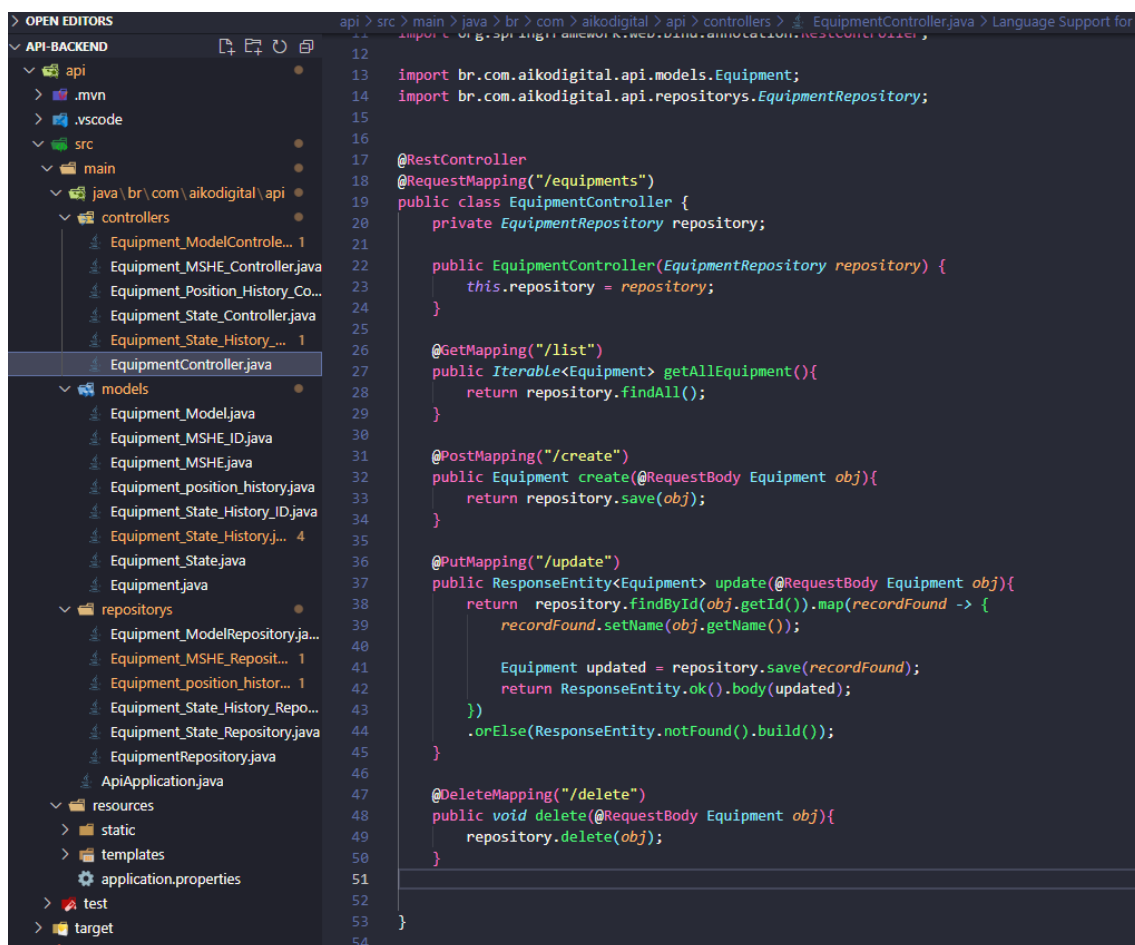
Repositório, que será utilizado para a implementação dos metodos de CRUD, sendo uma interface simples estendendo uma outra interface, CrudRepository, passando a entidade e o tipo de seu Id correspondente:



```
1 package br.com.aikodigital.api.repositories;
2
3 import java.util.UUID;
4
5 import org.springframework.data.repository.CrudRepository;
6 import org.springframework.stereotype.Repository;
7
8 import br.com.aikodigital.api.models.Equipment;
9
10 @Repository
11 public interface EquipmentRepository extends CrudRepository<Equipment, UUID> {
12
13 }
```

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.



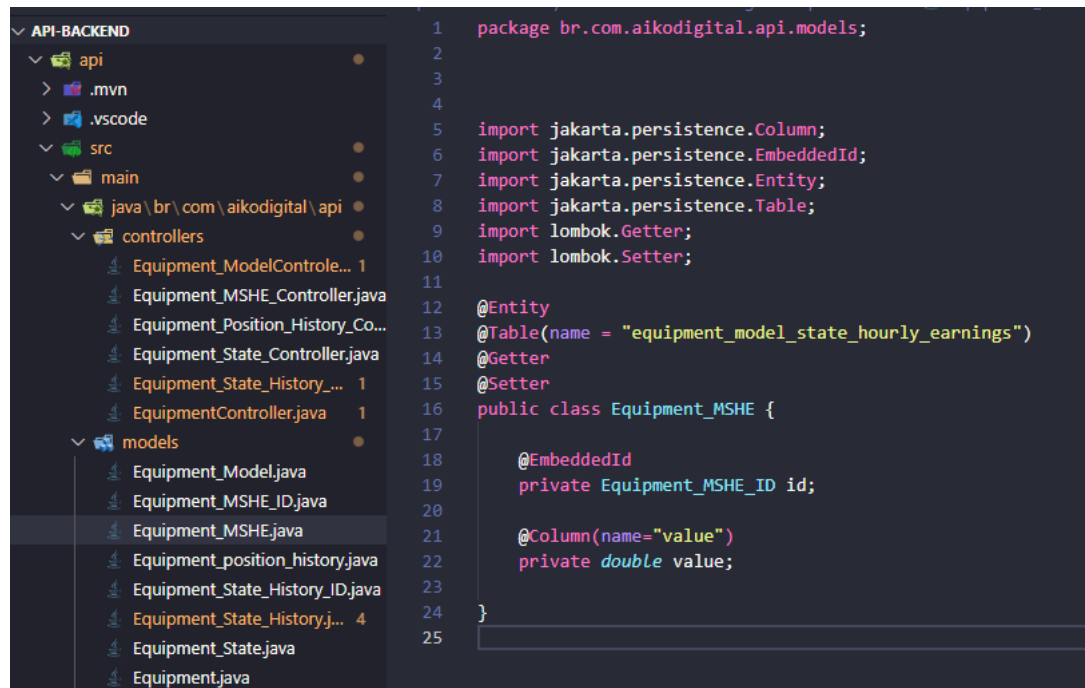
```
12 import org.springframework.web.bind.annotation.RestController;
13
14 import br.com.aikodigital.api.models.Equipment;
15 import br.com.aikodigital.api.repositories.EquipmentRepository;
16
17 @RestController
18 @RequestMapping("/equipments")
19 public class EquipmentController {
20     private EquipmentRepository repository;
21
22     public EquipmentController(EquipmentRepository repository) {
23         this.repository = repository;
24     }
25
26     @GetMapping("/list")
27     public Iterable<Equipment> getAllEquipment(){
28         return repository.findAll();
29     }
30
31     @PostMapping("/create")
32     public Equipment create(@RequestBody Equipment obj){
33         return repository.save(obj);
34     }
35
36     @PutMapping("/update")
37     public ResponseEntity<Equipment> update(@RequestBody Equipment obj){
38         return repository.findById(obj.getId()).map(recordFound -> {
39             recordFound.setName(obj.getName());
40
41             Equipment updated = repository.save(recordFound);
42             return ResponseEntity.ok().body(updated);
43         })
44         .orElse(ResponseEntity.notFound().build());
45     }
46
47     @DeleteMapping("/delete")
48     public void delete(@RequestBody Equipment obj){
49         repository.delete(obj);
50     }
51
52
53
54 }
```

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

No método 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.

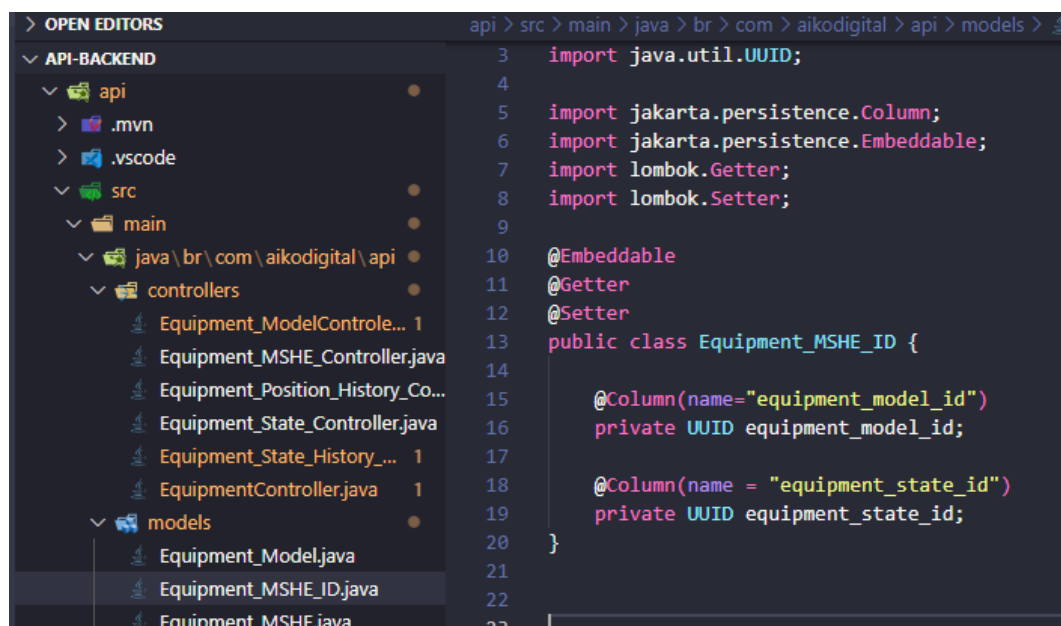
O restante dos métodos de CRUD, de forma simples, sendo realizados por meio dos métodos padrões do repositório criado.

A lógica 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.



```
1 package br.com.aikodigital.api.models;
2
3
4
5 import jakarta.persistence.Column;
6 import jakarta.persistence.EmbeddedId;
7 import jakarta.persistence.Entity;
8 import jakarta.persistence.Table;
9 import lombok.Getter;
10 import lombok.Setter;
11
12 @Entity
13 @Table(name = "equipment_model_state_hourly_earnings")
14 @Getter
15 @Setter
16 public class Equipment_MSHE {
17
18     @EmbeddedId
19     private Equipment_MSHE_ID id;
20
21     @Column(name="value")
22     private double value;
23
24 }
25
```

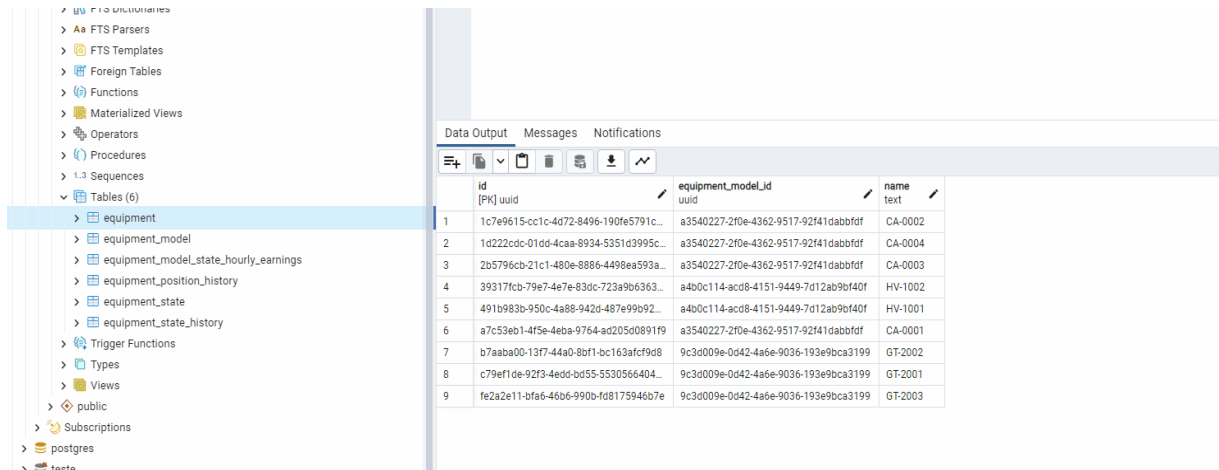
Então nestas duas classes optei por utilizar o @EmbeddedId, que é uma notação para utilização de chaves compostas como chave primária. E a criação de uma classe extra representante deste ID.



```
3 import java.util.UUID;
4
5 import jakarta.persistence.Column;
6 import jakarta.persistence.Embeddable;
7 import lombok.Getter;
8 import lombok.Setter;
9
10 @Embeddable
11 @Getter
12 @Setter
13 public class Equipment_MSHE_ID {
14
15     @Column(name="equipment_model_id")
16     private UUID equipment_model_id;
17
18     @Column(name = "equipment_state_id")
19     private UUID equipment_state_id;
20 }
21
22
23
```

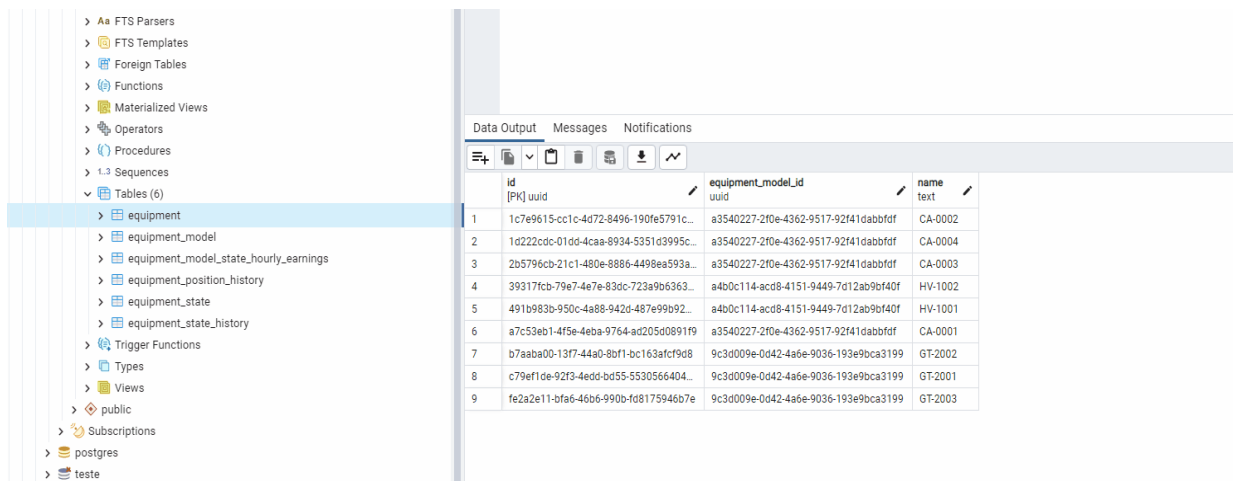
- Realização de Testes no Postman

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.



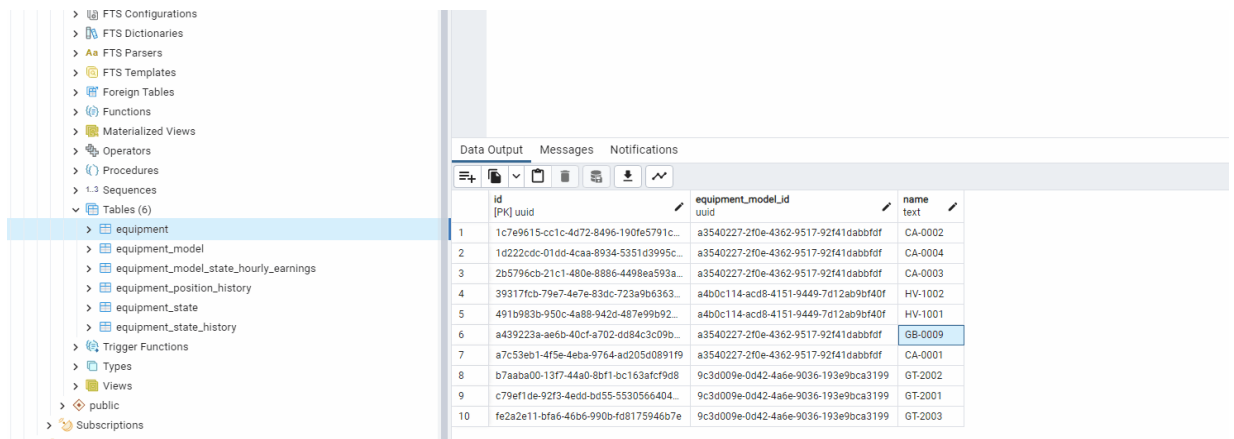
	id [PK] uid	equipment_model_id uid	name text
1	1c7e9615-cc1c-4d72-8496-190fe5791c...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0002
2	1d222cdc-01dd-4caa-8934-5351d3995c...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0004
3	2b5796cb-21c1-480e-8886-4498ea593a...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0003
4	39317fcb-79e7-4e7e-83dc-723a9b6363...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1002
5	491b983b-950c-4a88-942d-487e99b92...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1001
6	a7c53eb1-4f5e-4eba-9764-ad205d0891f9	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0001
7	b7aaba00-13f7-44a0-8bf1-bc163afc9d8	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2002
8	c79ef1de-92f3-4edd-bd55-5530566404...	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2001
9	fe2a2e11-bfa6-46b6-990b-f08175946b7e	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2003

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



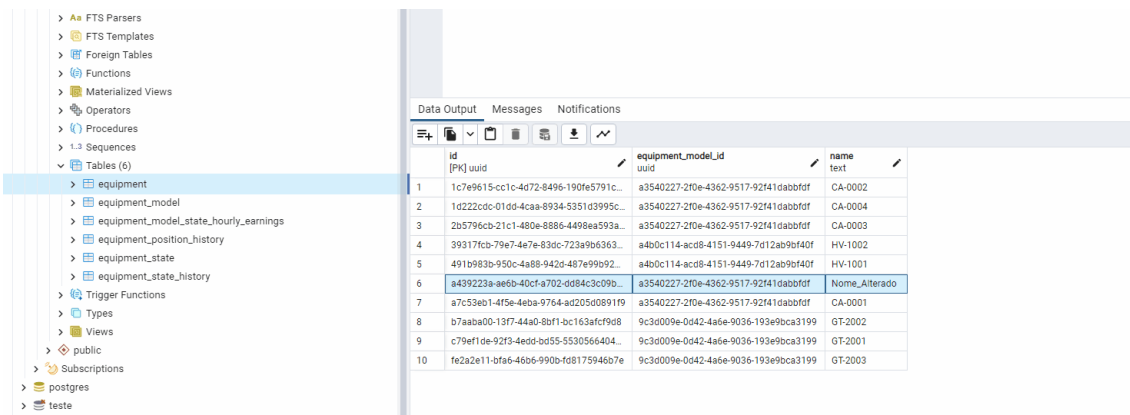
	id [PK] uid	equipment_model_id uid	name text
1	1c7e9615-cc1c-4d72-8496-190fe5791c...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0002
2	1d222cdc-01dd-4caa-8934-5351d3995c...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0004
3	2b5796cb-21c1-480e-8886-4498ea593a...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0003
4	39317fcb-79e7-4e7e-83dc-723a9b6363...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1002
5	491b983b-950c-4a88-942d-487e99b92...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1001
6	a7c53eb1-4f5e-4eba-9764-ad205d0891f9	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0001
7	b7aaba00-13f7-44a0-8bf1-bc163afc9d8	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2002
8	c79ef1de-92f3-4edd-bd55-5530566404...	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2001
9	fe2a2e11-bfa6-46b6-990b-f08175946b7e	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2003

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



	id [PK] uid	equipment_model_id uid	name text
1	1c7e9615-cc1c-4d72-8496-190fe5791c...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0002
2	1d222cdc-01dd-4caa-8934-5351d3995c...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0004
3	2b5796cb-21c1-480e-8886-4498ea593a...	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0003
4	39317fcb-79e7-4e7e-83dc-723a9b6363...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1002
5	491b983b-950c-4a88-942d-487e99b92...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1001
6	a439223a-ae6b-40cf-a702-dd84c3c09b...	a3540227-2f0e-4362-9517-92f41dabbbdf	GB-0009
7	a7c53eb1-4f5e-4eba-9764-ad205d0891f9	a3540227-2f0e-4362-9517-92f41dabbbdf	CA-0001
8	b7aaba00-13f7-44a0-8bf1-bc163afc9d8	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2002
9	c79ef1de-92f3-4edd-bd55-5530566404...	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2001
10	fe2a2e11-bfa6-46b6-990b-f08175946b7e	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2003

E para finalizar, o delete do equipamento:



	id [PK] uuid	equipment_model_id uuid	name text
1	1c7e9615-cc1c-4d72-8496-190fe5791c...	a3540227-2f0e-4362-9517-92f41daabf0f	CA-0002
2	1d222c0c-01d5-4caa-8934-5351d3995c...	a3540227-2f0e-4362-9517-92f41daabf0f	CA-0004
3	2b5796cb-21c1-480e-8886-4498ea593a...	a3540227-2f0e-4362-9517-92f41daabf0f	CA-0003
4	39317fcb-7967-4e7e-83dc-723a9b6363...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1002
5	491b583b-950c-4a88-942d-487e99b92...	a4b0c114-acd8-4151-9449-7d12ab9bf40f	HV-1001
6	a439223a-ae6b-40cf-a702-dd84c3c09b...	a3540227-2f0e-4362-9517-92f41daabf0f	Nome_Alterado
7	a7c53eb1-4f5e-4eba-9764-ad205d0891f9	a3540227-2f0e-4362-9517-92f41daabf0f	CA-0001
8	b7aabaw0-13f7-44a0-8bf1-bc163afc9d8	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2002
9	c79ef1de-92f3-4edd-bd55-5530566404...	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2001
10	fe2a2e11-bfa6-46b6-990b-fd8175946b7e	9c3d009e-0d42-4a6e-9036-193e9bca3199	GT-2003

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.