

Generation Brasil – Projeto Integrador

Projeto: Plataforma WEB – ColaborArt.

Tipo: E-commerce.

Representantes: Alana Rodrigues Teixeira / Bruno Falango / Gustavo Broch /

Laíse Lima Galvão de Souza / Vitor Carneiro Borela.

Back-End

Framework: Spring Boot

Link: <https://start.spring.io/>

Versão: 2.4.1

Projeto: Maven Project

Linguagem: Java

Versão: 8

Dependências Adicionadas:

- Spring Web
- Spring Boot DevTools
- Spring Data JPA
- Validation
- MySQL Driver

Banco de Dados: MySQL

Model

Produto - Atributos:

- **private long idProduto;**

Número de identificação do produto dentro da *tb_produto*.

(Exemplo: 1, 2, 3, etc.)

- **private String descricaoProduto;**

Descrição do produto dentro da *tb_produto*.

(Exemplo: Camiseta Personalizada, com estampa Star Wars)

- **private String tamanho;**

Tamanho do Produto.

(Exemplo: Camisas: PP, P, M, G, GG / Caneca: 250ml)

- **private BigDecimal valor;**

Valor do Produto.

(Exemplo: R\$22,50)

- **private boolean disponivel;**

Disponibilidade do Produto.

(Exemplo: true ou false)

- **private int estoque;**

Quantidade de produto no estoque.

(Exemplo: 20)

- **private String urlProduto;**

URL das imagens hospedadas.

(Exemplo: www.imgur.com/starwars)

- **private String nome;**

Nome do produto.

(Exemplo: Camiseta Star Wars Yoda)

- **private Categoria categoria;**

Relacionamento Produto/Categoria

Produto – Métodos:

- *getters / setters dos atributos do model;*

```
public long getIdProduto() {  
    return idProduto;  
}  
  
public void setIdProduto(long idProduto) {  
    this.idProduto = idProduto;  
}  
  
public String getDescricaoProduto() {  
    return descricaoProduto;  
}  
  
public void setDescricaoProduto(String descricaoProduto) {  
    this.descricaoProduto = descricaoProduto;  
}  
  
public String getTamanho() {  
    return tamanho;  
}  
  
public void setTamanho(String tamanho) {  
    this.tamanho = tamanho;  
}  
  
public BigDecimal getValor() {  
    return valor;  
}  
  
public void setValor(BigDecimal valor) {  
    this.valor = valor;  
}  
  
public boolean isDisponivel() {  
    return disponivel;  
}  
  
public void setDisponivel(boolean disponivel) {  
    this.disponivel = disponivel;  
}  
  
public int getEstoque() {  
    return estoque;  
}
```

```

public void setEstoque(int estoque) {
    this.estoque = estoque;
}

public String getUrlProduto() {
    return urlProduto;
}

public void setUrlProduto(String urlProduto) {
    this.urlProduto = urlProduto;
}

public Categoria getCategory() {
    return categoria;
}

public void setCategoria(Categoria categoria) {
    this.categoria = categoria;
}

public String getNome() {
    return nome;
}

public void setNome(String nome) {
    this.nome = nome;
}

```

Repository

```

9
10 @Repository
11 public interface ProdutoRepository extends JpaRepository<Produto, Long>{
12
13     public List<Produto> findAllByNameContainingIgnoreCase(String nome);
14 }
15

```

Interface – ProdutoRepository

Criação do repositório do produto, estendendo da Classe ***JpaRepository***, onde recebe a **Produto** Model e o tipo primitivo **Long**.

Método – Query Method

- findAllByNameContainingIgnoreCase

Lista todo tipo de produto pelo nome presente na *tb_produto*, de acordo com a texto passado pelo usuário.

Controller

```
@Autowired  
private ProdutoRepository repositorio;
```

Atributo para implementação do repositório, através da injeção de dependências (@Autowired).

```
@GetMapping()  
public ResponseEntity<List<Produto>> buscarTodosProdutos(){  
    return ResponseEntity.ok(repositorio.findAll());  
}
```

Retorna todos os produtos encontrados no banco de dados.

```
@GetMapping("/{id}")  
public ResponseEntity<Produto> buscarProdutoPorId(@PathVariable long id){  
    return repositorio.findById(id).map(resposta -> ResponseEntity.ok(resposta))  
        .orElse(ResponseEntity.notFound().build());  
}
```

Retorna o Produto de acordo com Id fornecido pelo usuário.

```
@GetMapping("/produto/{nome}")  
public ResponseEntity<List<Produto>> buscarProduto(@PathVariable String nome){  
    return ResponseEntity.ok(repositorio.findAllByNameContainingIgnoreCase(nome));  
}
```

Retorna valores dos Produtos, de acordo com o nome do produto fornecido (Query Method).

```
@PostMapping()  
public ResponseEntity<Produto> inserirProduto(@RequestBody Produto amostra){  
    return ResponseEntity.status(HttpStatus.CREATED).body(repositorio.save(amostra));  
}
```

Insere informações do Produto no banco de dados.

```
@PutMapping()
public ResponseEntity<Produto> atualizarProduto(@RequestBody Produto amostra){
    return ResponseEntity.status(HttpStatus.OK).body(repositorio.save(amostra));
}
```

Atualiza informações do Produto no banco de dados, de acordo com o Id.

```
@DeleteMapping("/{id}")
public void deleteProduto(@PathVariable long id) {
    repositorio.deleteById(id);
}
```

Apaga informações do Produto no banco de dados.

CRUD – Postman

GET

<http://localhost:8080/produto>

```
{
  "idProduto": 1,
  "descricaoProduto": "Produto Gamer,personagens de jogos",
  "tamanho": "P",
  "valor": 30.00,
  "disponivel": true,
  "estoque": 2,
  "urlProduto": "www.imagens.com.br",
  "categoria": {
    "idCategoria": 1,
    "tipoProduto": "Camiseta Gamer",
    "descricaoCategoria": "Camisetas produzidas com material 100% algodão, confortável e fácil de lavar, disponível em PP, P, M, G, GG, EXG e EXG2"
  },
  "nome": "Star Wars"
},
```

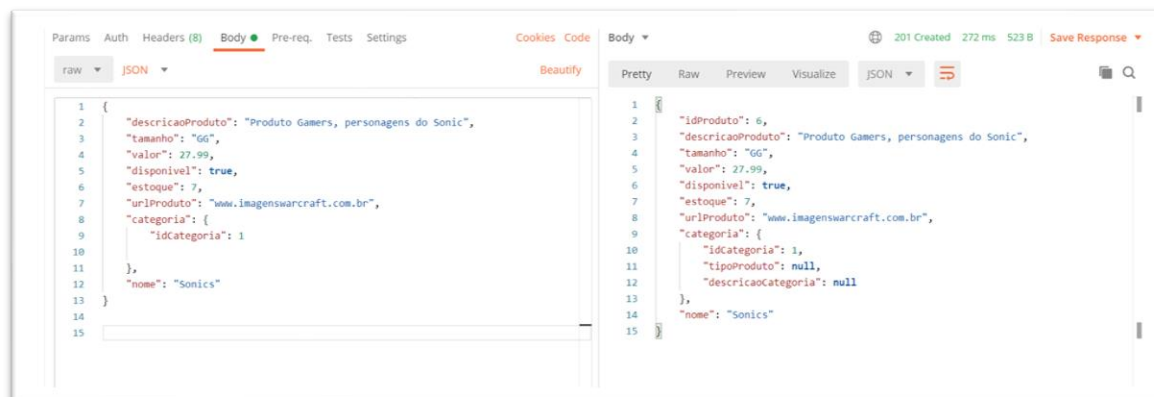
GET

<http://localhost:8080/produto/5>

```
{
  "idProduto": 5,
  "descricaoProduto": "Produto BodyTatto, personagens de jogos",
  "tamanho": "GG",
  "valor": 27.99,
  "disponivel": true,
  "estoque": 7,
  "urlProduto": "www.imagenswarcraft.com.br",
  "categoria": {
    "idCategoria": 2,
    "tipoProduto": "Camiseta BodyTattoo",
    "descricaoCategoria": "Camisetas produzidas com material 90% algodão mesmo e 10% poliéster, disponível em PP, P, M, G, GG, EXG e EXG2"
  },
  "nome": "Warcraft"
}
```

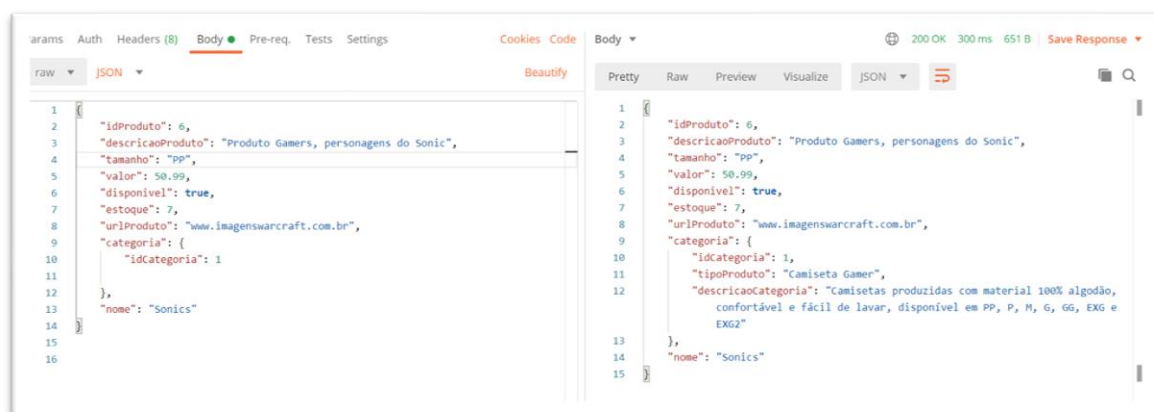
POST

<http://localhost:8080/produto/>



PUT

<http://localhost:8080/produto/>



DELETE

<http://localhost:8080/produto/6>

