

FIAP GRADUAÇÃO

SISTEMAS DE INFORMAÇÃO

MICROSERVICE AND WEB ENGINEERING

Prof^a. Aparecida Castello Rosa
profaparecida.rosa@fiap.com.br

■ Agenda

- Continuar com o projeto produto-mvc
- Camada de Serviço - Produto

Objetivos

- Continuar com o projeto produto-mvc
- Entender o funcionamento da camada de serviços.
- Delegar e implementar as reponsabilidades do controller Produto para a classe de serviço.

I Responsabilidades

O Design Pattern **Front Controller**

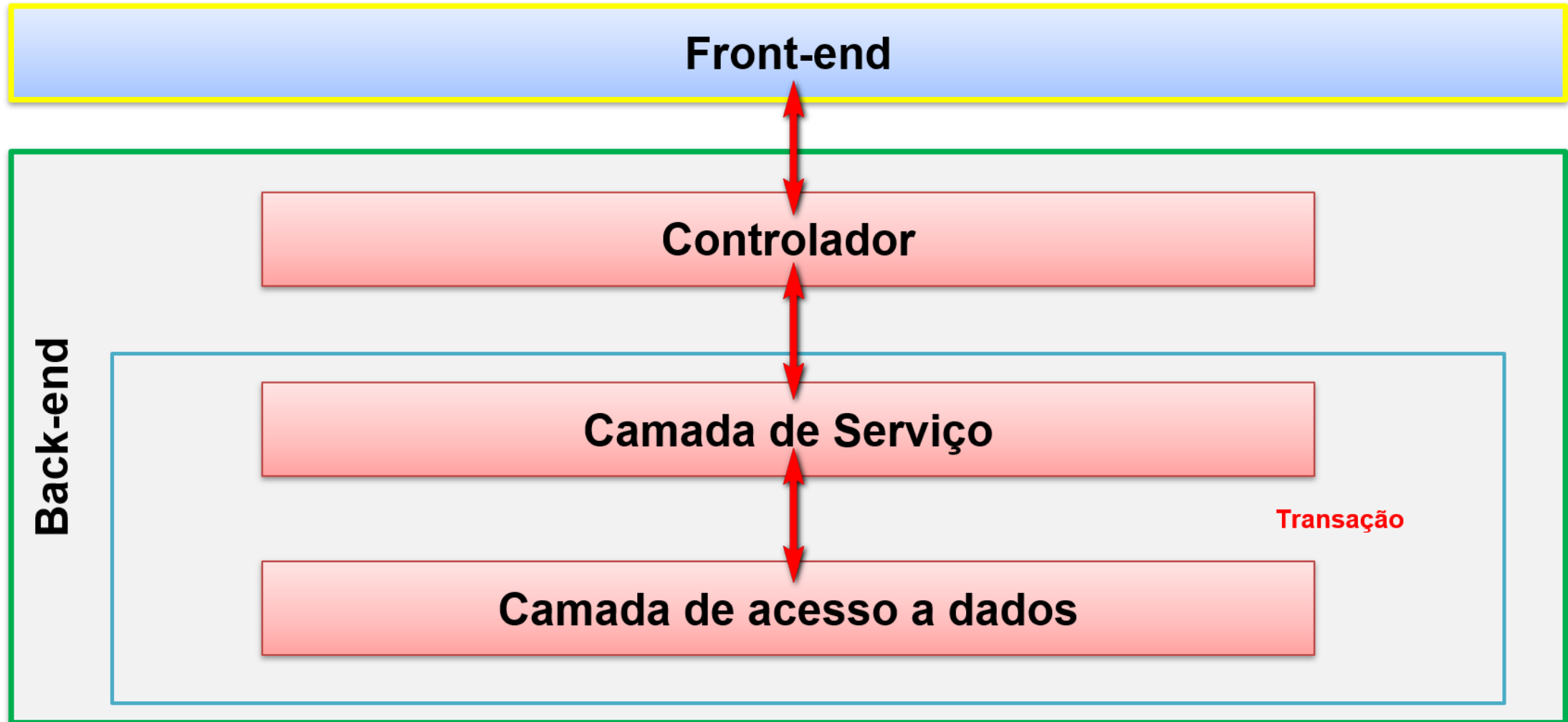
Definição de **MVC**

- **Model:** Define o modelo ou domínio da aplicação
 - Regras de negócios
 - Persistência de dados
- **View:** Interação com usuário
- **Controller:** Componente intermediário entre View e Model
 - Recebe requisições (requests)
 - Envia respostas (responses)
 - Interage com a camada model

Responsabilidades

- **Controller:** responder interações do usuário
 - No caso do MVC: recebe requisições (requests); enviar respostas (responses); Interage com a camada model.
- **Service:** realizar operações de negócio.
 - Um método da camada de serviços deve ter um significado relacionado ao negócio, podendo executar várias operações.
- **Repository:**
 - Realizar operações “individuais” de acesso ao banco de dados.

Responsabilidades



Camada de Serviços para Produto

Exercício

■ Criar class ProdutoService

```
ProdutoService.java x
1  package br.com.fiap.produtomvc.services;
2
3  import org.springframework.stereotype.Service;
4
5  @Service
6  public class ProdutoService {
7
8
9  }
```


■ Implementar findAll class ProdutoService

```
ProdutoService.java x CategoriaService.java x
11 @Service
12 public class ProdutoService {
13
14     // Injeção de dependência de repository
15     @Autowired
16     private ProdutoRepository repository;
17
18     //Método que retorna uma lista de Produtos
19     @Transactional(readOnly = true)
20     public List<Produto> findAll(){
21         return repository.findAll();
22     }
}
```

Alterações ProdutoController

```
ProdutoService.java x ProdutoController.java x CategoriaService.java x CategoriaController.java x
22 public class ProdutoController {
23
24     // Injeção de dependência de service
25     @Autowired
26     private ProdutoService service;
27
28     @Autowired
29     private ProdutoRepository repository;
30
31     @Autowired
32     //private CategoriaRepository categoriaRepository;
33     private CategoriaService categoriaService;
34
35     @ModelAttribute("categorias")
36     public List<Categoria> categorias(){
37         //return categoriaRepository.findAll();
38         return categoriaService.findAll();
39     }
```

Alterar findAll ProdutoController



The screenshot shows an IDE with two tabs: 'ProdutoService.java' and 'ProdutoController.java'. The 'ProdutoController.java' tab is active and shows the following code:

```
61     @GetMapping()  
62     //@Transactional(readOnly = true)  
63     @RequestMapping("/produtos")  
64     public String findAll(Model model){  
65         //model.addAttribute("produtos", repository.findAll());  
66         model.addAttribute("produtos", service.findAll());  
67         return "/produto/listar-produtos";  
68     }
```

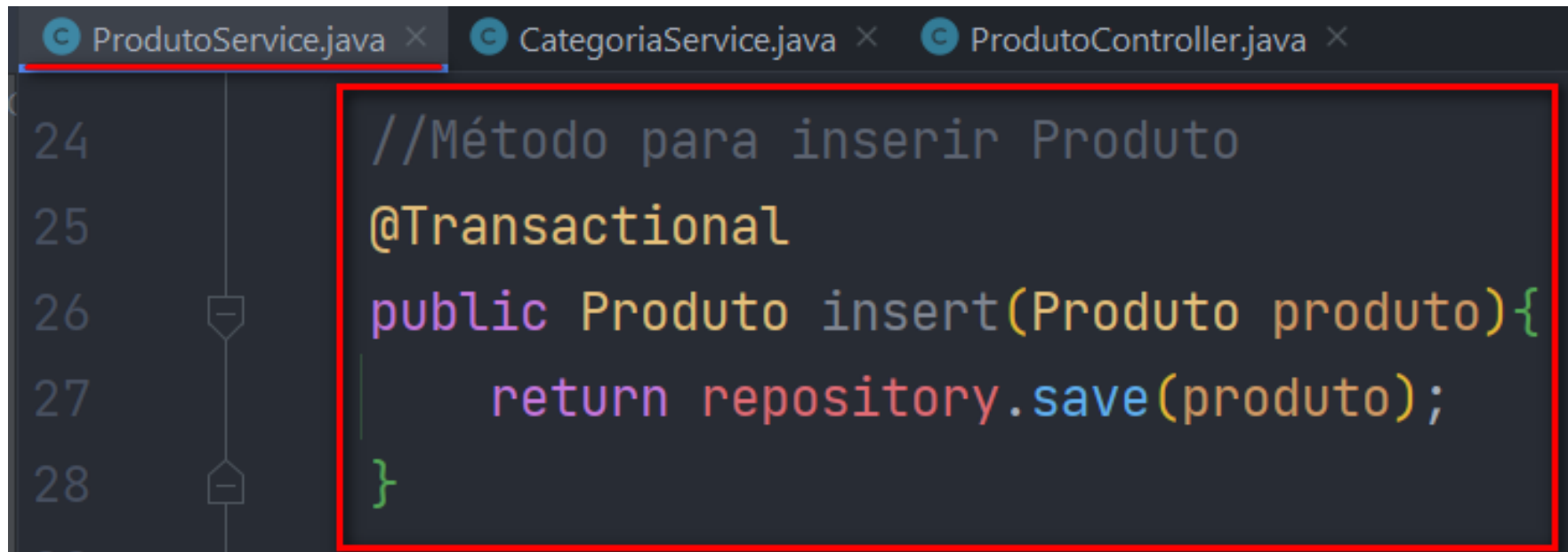
■ Testar aplicação

FIAP Home Categoria ▾ Produto ▾

Lista de Produtos

Nome	Descrição	Categoria	Valor	Ações	
Mouse Microsoft	Mouse sem fio	Mouse	250.0	Excluir	Editar
Smartphone Samsung Galaxy A54 5G	Samsung Galaxy A54 5G	Smartphone	1799.0	Excluir	Editar
Smart TV	Smart TV LG LED 65 polegadas	Smart TV	3999.0	Excluir	Editar

■ Implementar insert class ProdutoService



The screenshot shows an IDE with three tabs: ProdutoService.java, CategoriaService.java, and ProdutoController.java. The ProdutoService.java tab is active, and the following code is visible, with the insert method implementation highlighted by a red rectangle:

```
24 //Método para inserir Produto
25 @Transactional
26 public Produto insert(Produto produto){
27     return repository.save(produto);
28 }
```

Alterar insert ProdutoController

```
ProdutoService.java x CategoriaService.java x ProdutoController.java x CategoriaController.java x
48      @PostMapping()
49      //@Transactional
50      @ public String insert(@Valid Produto produto,
51                           BindingResult result,
52                           RedirectAttributes attributes) {
53          if(result.hasErrors()){
54              return "produto/novo-produto";
55          }
56          //repository.save(produto);
57          produto = service.insert(produto);
58          attributes.addFlashAttribute("mensagem", "Produto salvo com sucesso");
59          return "redirect:/produtos/form";
60      }
```

Testar aplicação

FIAP Home Categoria ▾ Produto ▾

Cadastro de Produtos

Nome: Campo requerido
O nome deve ter no mínimo 3 caracteres

Descrição: Campo requerido

Categoria:

Valor: Campo requerido

Salvar

FIAP Home Categoria ▾ Produto ▾

Cadastro de Produtos

Nome: Campo requerido
O nome deve ter no mínimo 3 caracteres

Descrição: Campo requerido

Categoria:

Valor: Campo requerido

Salvar

FIAP Home Categoria ▾ Produto ▾

Lista de Produtos

Nome	Descrição	Categoria	Valor	Ações	
Mouse Microsoft	Mouse sem fio	Mouse	250.0	Excluir	Editar
Smartphone Samsung Galaxy A54 5G	Samsung Galaxy A54 5G	Smartphone	1799.0	Excluir	Editar
Smart TV	Smart TV LG LED 65 polegadas	Smart TV	3999.0	Excluir	Editar
Teste	Teste	Tablet	12.0	Excluir	Editar

Implementar findById class ProdutoService

```
ProdutoService.java x CategoriaService.java x ProdutoController.java x CategoriaController.java x
30 //Método para buscar Produto por Id
31 @Transactional(readonly = true)
32 public Produto findById(Long id){
33
34     Produto produto = repository.findById(id).orElseThrow(
35         () -> new IllegalArgumentException("Recurso inválido - " + id)
36     );
37     return produto;
38 }
```


Alterar findById ProdutoController

```
ProdutoService.java x ProdutoController.java x CategoriaService.java x CategoriaController.java x
70 @GetMapping("/{id}")
71 //@Transactional(readOnly = true)
72 @ public String findById(@PathVariable ("id") Long id, Model model ){
73
74     // Produto produto = repository.findById(id).orElseThrow(
75     //     () -> new IllegalArgumentException("Produto inválido - id: " + id)
76     // );
77     Produto produto = service.findById(id);
78     model.addAttribute("produto", produto);
79     return "/produto/editar-produto";
80 }
```

Implementar update class ProdutoService

```
ProdutoService.java x ProdutoController.java x CategoriaService.java x CategoriaController.java x
41 @Transactional
42 public Produto update(Long id, Produto entity) {
43     try {
44         Produto produto = repository.getReferenceById(id);
45         copyToProduto(entity, produto);
46         produto = repository.save(produto);
47         return produto;
48     } catch (EntityNotFoundException e) {
49         throw new IllegalArgumentException("Recurso não encontrado");
50     }
51 }
52
53 @private void copyToProduto(Produto entity, Produto produto) {
54     produto.setNome(entity.getNome());
55     produto.setDescricao(entity.getDescricao());
56     produto.setValor(entity.getValor());
57     produto.setCategoria(entity.getCategoria());
58 }
```

Alterar update ProdutoController

```
ProdutoService.java x ProdutoController.java x CategoriaService.java x CategoriaController.java x
82      @PutMapping("/{id}")
83      //@Transactional
84      @ public String update(@PathVariable("id") Long id,
85                           @Valid Produto produto,
86                           BindingResult result){
87          if(result.hasErrors()){
88              produto.setId(id);
89              return "/produto/editar-produto";
90          }
91          //repository.save(produto);
92          service.update(id, produto);
93          return "redirect:/produtos";
94      }
```

Testar Aplicação

FIAP Home Categoria ▾ Produto ▾

Lista de Produtos

Nome	Descrição	Categoria	Valor	Ações
Mouse Microsoft	Mouse sem fio	Mouse	250.0	Excluir Editar
Smartphone Samsung Galaxy A54 5G	Samsung Galaxy A54 5G	Smartphone	1799.0	Excluir Editar
Smart TV	Smart TV LG LED 65 polegadas	Smart TV	3999.0	Excluir Editar
Teste	Teste	Tablet	12.0	Excluir Editar

FIAP Home Categoria ▾ Produto ▾

Alteração de Produto

Nome:

Descrição:

Categoria:

Valor:

[Alterar](#)

FIAP Home Categoria ▾ Produto ▾

Lista de Produtos

Nome	Descrição	Categoria	Valor	Ações
Mouse Microsoft	Mouse sem fio	Mouse	250.0	Excluir Editar
Smartphone Samsung Galaxy A54 5G	Samsung Galaxy A54 5G	Smartphone	1799.0	Excluir Editar
Smart TV	Smart TV LG LED 65 polegadas	Smart TV	3999.0	Excluir Editar
Teste 1	Teste	Smartphone	12.0	Excluir Editar

■ Implementar delete class ProdutoService

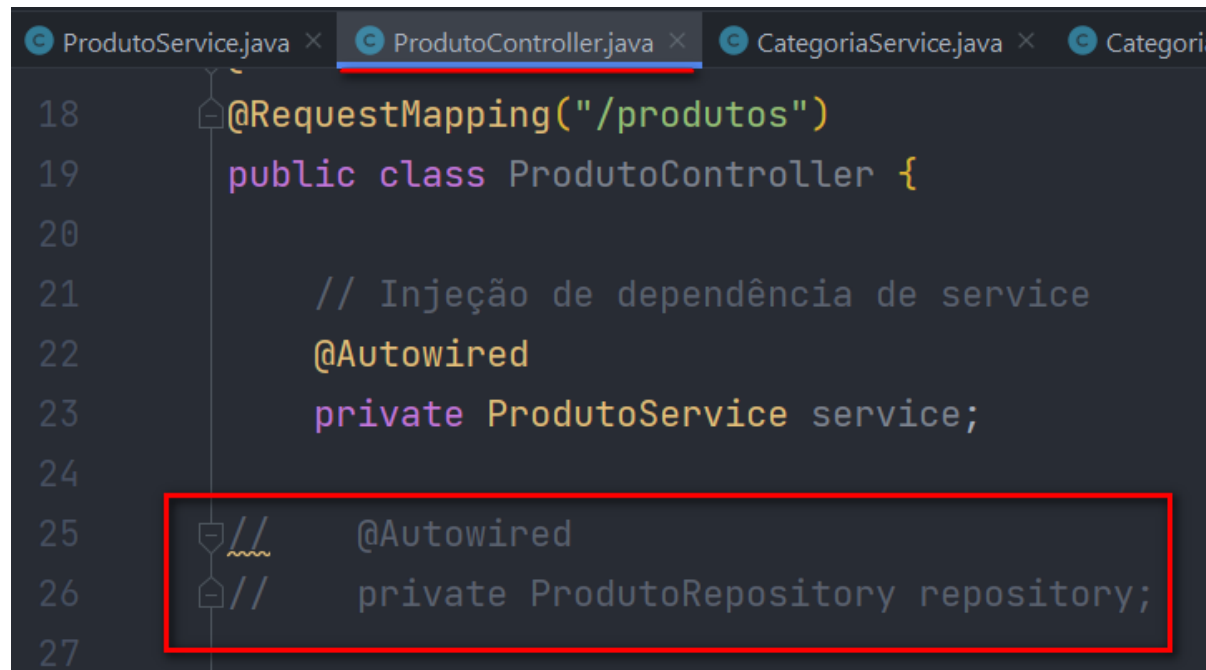
```
ProdutoService.java x ProdutoController.java x CategoriaService.java x CategoriaController.java x
53 // Método para excluir Produto
54 @Transactional
55 public void delete(Long id){
56     if(!repository.existsById(id)){
57         throw new IllegalArgumentException("Produto inválido - id: " + id);
58     }
59     try {
60         repository.deleteById(id);
61     } catch (Exception e){
62         throw new IllegalArgumentException("Produto inválido - id: " + id);
63     }
64 }
```

Alterar delete ProdutoController

```
ProdutoService.java x ProdutoController.java x CategoriaService.java x CategoriaController.java x
95
96     @DeleteMapping("/{id}")
97     //@Transactional
98     public String delete(@PathVariable("id") Long id, Model model){
99         //         if(!repository.existsById(id)){
100             //             throw new IllegalArgumentException("Produto inválido - id: " + id);
101             //         }
102         //         try {
103             //             repository.deleteById(id);
104             //         } catch (Exception e){
105             //             throw new IllegalArgumentException("Produto inválido - id: " + id);
106             //         }
107         service.delete(id);
108
109         return "redirect:/produtos";
110     }
```

Finalizando class CategoriaController

- Organizar os imports.
- Não estamos utilizando atributo *repository* então podemos excluir ou deixar comentado.



```
18 @RequestMapping("/produtos")
19 public class ProdutoController {
20
21     // Injeção de dependência de service
22     @Autowired
23     private ProdutoService service;
24
25     // @Autowired
26     // private ProdutoRepository repository;
27 }
```

- Testar todas as funcionalidades de categoria e produto da aplicação .



Copyright © 2024
Prof^a. Aparecida de Fátima Castello Rosa

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).