Aula 5 - Validação e Testes

Tempo estimado para esta prática: 30 min

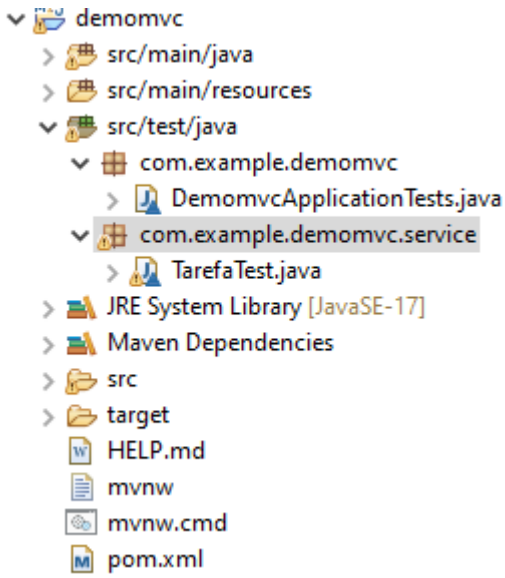
IDE utilizada: Eclipse JavaEE

Banco de Dados: Mysql Workbench

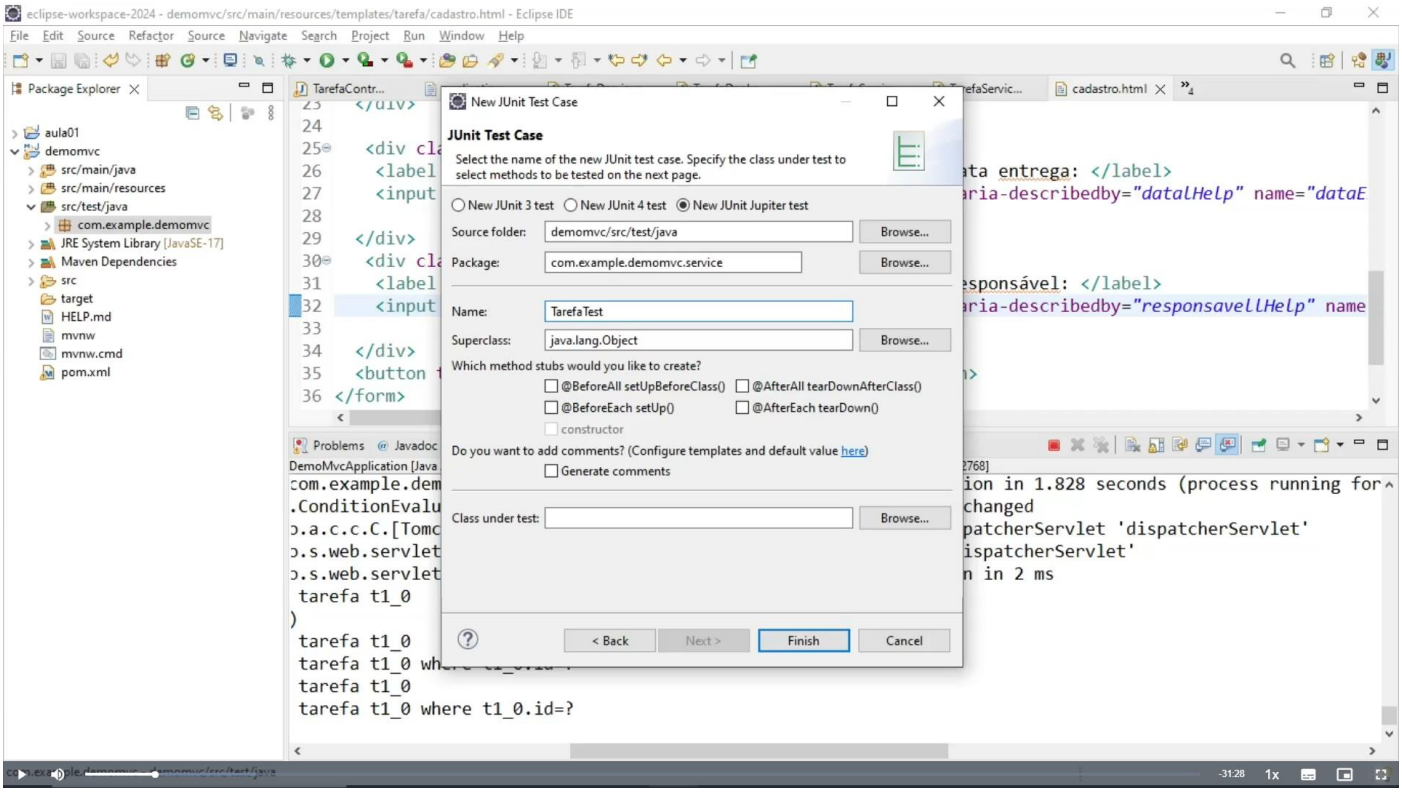
Nessa aula iremos fazer os testes e as validações na nossa aplicação para gerenciar tarefas. Os arquivos necessário para essa aula estão disponíveis na rota de aprendizagem.

1. Teste com junit

Criar um pacote chamado service em teste/java. Tem uma pasta especifica para testes em src/test/java



Criar um arquivo de teste (TarefaTest.java) para testar os métodos da classe de serviço.Clicar com botão direito do mouse e escolher new > Other >Java>Junit>JUnit Test Case



TarefaTest.java

```java
package com.example.demomvc.service;
import static org.junit.jupiter.api.Assertions.*;
import java.time.LocalDate;
import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.TestMethodOrder;
import org.junit.jupiter.api.extension.ExtendWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.junit.jupiter.api.MethodOrderer.OrderAnnotation;
import org.junit.jupiter.api.Order;
import org.springframework.test.context.junit.jupiter.SpringExtension;
import com.example.demomvc.entity.Tarefa;
@SpringBootTest
@ExtendWith(SpringExtension.class)
@TestMethodOrder(OrderAnnotation.class)
class TesteTarefa {
	@Autowired
	private TarefaServiceImpl ts;
	@Test
	@Order(1)
	public void insere() {
		Tarefa tarefa = new Tarefa();
		tarefa.setNome("revisao");
		tarefa.setDataEntrega(LocalDate.of(2024, 06, 20));
		tarefa.setResponsavel("Luciane");
		ts.salvar(tarefa);
	}
```

A seguir vamos testar também os métodos de pesquisa, atualização e remoção.

```java
	@Test
	@Order(2)
	public void pesquisaPeloId() {
		Tarefa tarefa = ts.buscarPorId(2L);
```

```
                        System.out.println(tarefa);
        }

        @Test
    @Order(3)
        public void atualiza() {
                 Tarefa tarefa = ts.buscarPorId(2L);
                tarefa.setNome("Review");
                 ts.editar(tarefa);
           }
        @Test
        @Order(4)

        public void remove() {
               Tarefa tarefa = ts.buscarPorId(2L);

                 ts.excluir(tarefa.getId());


        }


}
```

Testar com run > Junit Test

Colocar uma tela com junit test

2. VALIDAÇÂO

1. Vamos trabalhar validação de campos. Para isso é necessário incluir dependência no pom:

```
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-validation</artifactId>
</dependency>
```

2. Em Tarefa.java

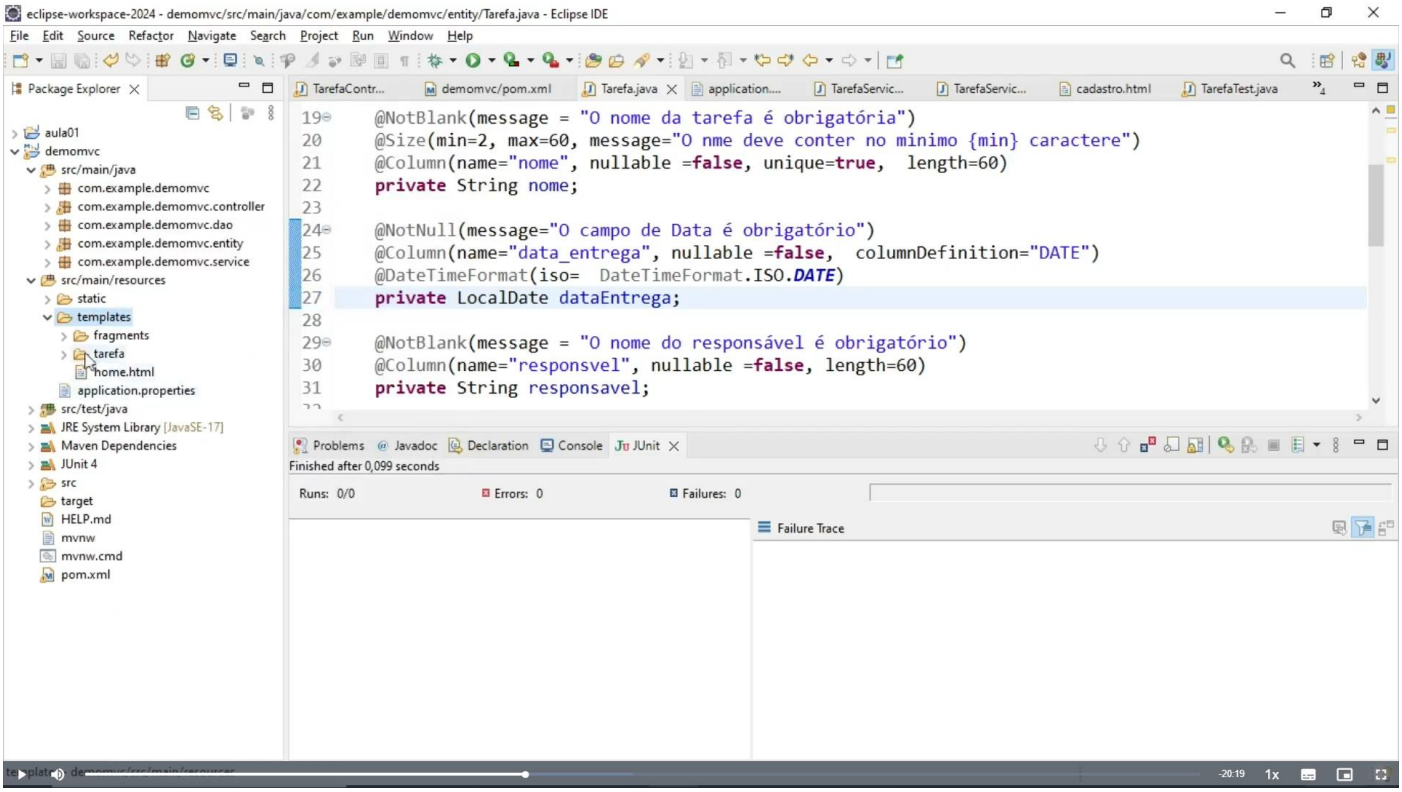
Colocar as anotações para validação de campo(em amarelo)

```
@NotBlank(message="nome da tarefa é obrigatória")
@Size(min=2, max=60, message="O nome deve conter no minimo {min} caractere")
@NotNull(message="campo de data é obrigatório")
```

```
package com.example.demomvc.entity;
import java.time.LocalDate;
import org.springframework.format.annotation.DateTimeFormat;
import jakarta.persistence.*;
import jakarta.validation.constraints.NotBlank;
import jakarta.validation.constraints.NotNull;
import jakarta.validation.constraints.Size;
@Entity
```

```java
@Table(name="TAREFA")
public class Tarefa extends AbstractEntity<Long>{

	@NotBlank(message="nome da tarefa é obrigatória")
	@Size(min=2, max=60, message="O nome deve conter no minimo {min} caractere")
	@Column(name="nome", nullable=false, unique=true, length=60)
	private String nome;

	@NotNull(message="campo de data é obrigatório")
	@Column( nullable=false, name="data_entrega", columnDefinition="DATE")
	@DateTimeFormat(iso = DateTimeFormat.ISO.DATE)
	private LocalDate dataEntrega;

	@NotBlank(message="nome do  responsável é obrigatória")
	@Column(name="responsavel", nullable=false, length=60)
	private String responsavel;



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

	public LocalDate getDataEntrega() {
		return dataEntrega;
	}
	public void setDataEntrega(LocalDate dataEntrega) {
		this.dataEntrega = dataEntrega;
	}
	public String getResponsavel() {
		return responsavel;
	}
	public void setResponsavel(String responsavel) {
		this.responsavel = responsavel;
	}


}
```

3. Copiar a pasta fragments. Nela temos a pagina validacao.html, que contém o seguinte conteúdo:



Codigo de validacao.html

```
<html>
<head><meta charset="UTF-8"></head>
<body>
	<div th:if="${#fields.hasAnyErrors()}" th:fragment="validacao">
		<div class="alert alert-danger alert-dismissible fade show" role="alert">
		<div th:each="error : ${#fields.detailedErrors()}">
		  <i class="oi oi-warning"></i>
		    <span th:text="${error.message}"></span>
		</div>
		<button type="button" class="close" data-dismiss="alert" aria-label="Close">
		  <span aria-hidden="true">&times;</span>
		</button>
	    </div>
	</div>
</body>
</html>
```

4. EmTarefaController fazer as modificações nos Método salvar e editar

```
package com.example.demomvc.controller;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
import com.example.demomvc.entity.Tarefa;
import com.example.demomvc.service.TarefaService;
import jakarta.validation.Valid;
@Controller
@RequestMapping("/tarefas")
public class TarefaController {
```

```
	@Autowired
	private	TarefaService service;
	@GetMapping("/cadastro")
	public String cadastro(Tarefa tarefa) {
		return "/tarefa/cadastro";
	}



	@PostMapping("/salvar")
	public String salvar(@Valid Tarefa tarefa,BindingResult result) {
		if (result.hasErrors()) {
			return "/tarefa/cadastro";
		}
		service.salvar(tarefa);

		return "redirect:/tarefas/cadastro";
	}

	@GetMapping("/lista")
	public String listar(ModelMap model ) {
		model.addAttribute("tarefas", service.buscaTodos());
		return "tarefa/lista";
	}

	// editar e excluir
	@PostMapping("/editar")
	public String editar(@Valid Tarefa tarefa) {
		if (result.hasErrors()) {
			return "/tarefa/cadastro";
		}
		service.editar(tarefa);

		return "redirect:/tarefas/cadastro";
	}

	@GetMapping("/editar/{id}")
	public String preEditar(@PathVariable("id") Long id, ModelMap model) {

		return "/tarefa/cadastro";
	}

	@GetMapping("/excluir/{id}")
	public String excluir(@PathVariable("id") Long id, ModelMap model) {
		service.excluir(id);

		return listar(model);
	}
}
```

5. Colocar a tag de validação no cadastro.hml

* Está no bloco de notas

```
<div th:replace="fragments/validacao :: validacao"></div>
```

```
<!DOCTYPE html>
```

```
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Cadastro de Tarefas</title>

<!-- Bootstrap core CSS -->
<link href="/webjars/bootstrap/css/bootstrap.min.css" rel="stylesheet" />
<link rel="stylesheet" type="text/css" media="all" href="../../css/style.css"  />
</head>
<body>
<div th:replace="fragments/alert"></div>
<form th:action="${tarefa.id == null} ? @{/tarefas/salvar} : @{/tarefas/editar}"
th:object="${tarefa}" method="POST">
<div th:replace="fragments/validacao :: validacao"></div>

<input type="hidden"  th:field="*{id}">


 <div class="mb-3">
   <label for="exampleInputTarefa" class="form-label">Nome da Tarefa</label>
   <input type="text" class="form-control" id="nome" name = "nome" th:field="*{nome}">
 </div>

  <div class="mb-3">
   <label for="exampleInputDataEntrega" class="form-label">Data entrega: </label>
   <input type="date" class="form-control" id="datacriacao" aria-
describedby="datalHelp" name="dataEntrega"  th:field="*{dataEntrega}">

 </div>
  <div class="mb-3">
   <label for="exampleInputResponsavel" class="form-label">Responsável: </label>
   <input type="text" class="form-control" id="responsavel" aria-
describedby="responsavellHelp" name="responsavel" th:field="*{responsavel}">

 </div>
 <button type="submit" class="btn btn-primary">Submit</button>
</form>
<div th:insert="~{footer::copy}">&copy; 2024 modelo estático</div>
</body>
</html>
```

6.Testar a aplicação

Verificar que se deixar os campos em branco vai mostrar a mensagem. Reparar que quando editamos, cadastramos, não temos mensagem indicando que deu certo ou não.

7. Vamos trabalhar com Fragmento de mensagem de alerta quando incluir/excluir/editar

Na pasta fragments temos o fragmento alert.html

```
      <div th:if="${success} != null">
```

```
            <div class="alert alert-success alert-dismissible fade show" role="alert">
              <i class="oi oi-check"></i>
              <span>
                  <strong th:text="${success}"></strong>
              </span>
              <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                <span aria-hidden="true">&times;</span>
              </button>
            </div>
      </div>

      <div th:if="${fail} != null">
            <div class="alert alert-danger alert-dismissible fade show" role="alert">
              <i class="oi oi-check"></i>
              <span>
                  <strong th:text="${fail}"></strong>
              </span>
              <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                <span aria-hidden="true">&times;</span>
              </button>
            </div>
      </div>
```

7.1.colocar em cadastro.html e lista.html o div:

```
<div th:replace="fragments/alert"></div>
```

em cadastro.html:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Cadastro de Tarefas</title>
<link href="/webjars/bootstrap/css/bootstrap.min.css" rel="stylesheet" />
<link rel="stylesheet" type="text/css" media="all" href="../../css/style.css"  />
</head>
<body>
<div th:replace="fragments/alert"></div>
<form th:action="${tarefa.id == null} ? @{/tarefas/salvar} : @{/tarefas/editar}"
                        th:object="${tarefa}" method="POST">
            <div th:replace="fragments/validacao :: validacao"></div>
            <input type="hidden"  th:field="*{id}">
 <div class="mb-3">
   <label for="exampleInputTarefa" class="form-label">Nome da Tarefa</label>
   <input type="text" class="form-control" id="nome" name = "nome" th:field="*{nome}">
 </div>

  <div class="mb-3">
   <label for="exampleInputDataEntrega" class="form-label">Data entrega: </label>
   <input type="date" class="form-control" id="datacriacao" aria-describedby="datalHelp"
name="dataEntrega" th:field="*{dataEntrega}" >

 </div>
  <div class="mb-3">
   <label for="exampleInputResponsavel" class="form-label">Responsável: </label>
   <input type="text" class="form-control" id="responsavel" aria-describedby="responsavellHelp"
name="responsavel"  th:field="*{responsavel}">
```

```
 </div>
 <button type="submit" class="btn btn-primary">Submit</button>
</form>
</body>
</html>
```

colocar na lista.html também:

```
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Gerenciador de Tarefas</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<link href="/webjars/bootstrap/css/bootstrap.min.css" rel="stylesheet" />
<link rel="stylesheet" type="text/css" media="all"  href="../../../css/style.css" />
</head>
<body>
      <h1>Lista de Tarefas</h1>
<div th:replace="fragments/alert"></div>
      <table>
            <thead>
                  <tr>
                        <th>Identificação da Tarefa</th>
                        <th>Nome da Tarefa</th>
                        <th>Data entrega</th>
                        <th>Responsável</th>
                  </tr>
            </thead>
            <tbody>
                  <tr th:each="tarefa : ${tarefas}">
                        <td th:text="${tarefa.id}">1</td>
                        <td th:text="${tarefa.nome}">Criação de conteúdo</td>
                        <td th:text="${tarefa.dataEntrega}">15/08/2024</td>
                        <td th:text="${tarefa.Responsavel}">Luciane</td>
                        <td colspan="2"><a class="btn btn-info btn-sm"
                              th:href="@{/tarefas/editar/{id} (id=${tarefa.id}) }"
role="button">
                                    <span class="oi oi-brush" title="Editar" aria-
hidden="true">Editar</span>
                        </a> <a class="btn btn-info btn-sm"
                              th:href="@{/tarefas/excluir/{id} (id=${tarefa.id}) }"
role="button">
                                    <span class="oi oi-brush" title="Excluir" aria-
hidden="true">Excluir</span>
                        </a></td>
                  </tr>
            </tbody>
      </table>
      <p>
            <a href="../">Return to home</a>
      </p>

      <div >&copy; 2024 modelo estático</div>
</body>
</html>
```

8. colocar o redirectAttributes no TarefaController

```
package com.example.demomvc.controller;
```

```java
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
import com.example.demomvc.entity.Tarefa;
import com.example.demomvc.service.TarefaService;
import jakarta.validation.Valid;
@Controller
@RequestMapping("/tarefas")
public class TarefaController {
	@Autowired
	private TarefaService service;
	@GetMapping("/cadastro")
	public String cadastro(Tarefa tarefa) {
		return "/tarefa/cadastro";
	}
	@PostMapping("/salvar")
	public String salvar(@Valid Tarefa tarefa, BindingResult result,
RedirectAttributes attr) {
		if (result.hasErrors()) {
			return "/tarefa/cadastro";
		}
		service.salvar(tarefa);
		attr.addFlashAttribute("success", "Tarefa salva com sucesso.");
		System.out.println("ola");
		return "redirect:/tarefas/lista";
	}
	@GetMapping("/lista")
	public String listar(ModelMap model) {
		model.addAttribute("tarefas", service.buscaTodos());
		return "tarefa/lista";
	}
	// editar e excluir
	@PostMapping("/editar")
	public String editar(@Valid Tarefa tarefa, BindingResult result,
RedirectAttributes attr) {
		if (result.hasErrors()) {
			return "/tarefa/cadastro";
		}
		service.editar(tarefa);
		attr.addFlashAttribute("success", "Tarefa Editada com sucesso.");
		return "redirect:/tarefas/cadastro";
	}
	@GetMapping("/editar/{id}")
	public String preEditar(@PathVariable("id") Long id, ModelMap model) {
		model.addAttribute("tarefa", service.buscarPorId(id));
		return "/tarefa/cadastro";
	}
	@GetMapping("/excluir/{id}")
	public String excluir(@PathVariable("id") Long id, ModelMap model) {
		service.excluir(id);
		model.addAttribute("success", "Tarefa excluída com sucesso.");
		return listar(model);
```

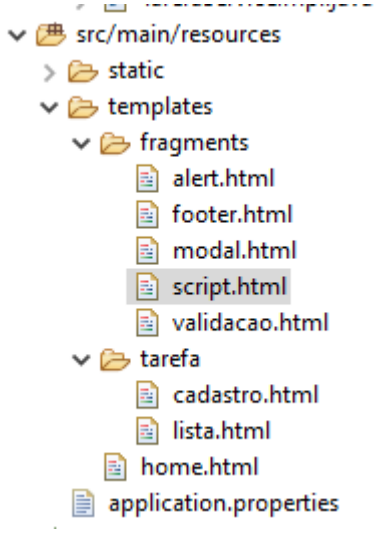
```
        }
}
```

Validação com Javascript

Use o plugin modal JavaScript do Bootstrap para adicionar caixas de diálogo ao seu site para lightboxes, notificações de usuário ou conteúdo totalmente personalizado.

Vamos fazer a validação antes de excluir com modal do bootstrap.

Temos dois arquivos na pasta fragments: o arquivo modal.html e script.html



vamos modificar o botão de exclusão

1. colocar o webjar do bootstrap (disponível no bloco de notas)

```
<link href="/webjars/bootstrap/css/bootstrap.min.css" rel="stylesheet" />
```

2. em lista.html colocar os scripts na ordem:

```
<script src="/webjars/jquery/3.5.1/jquery.slim.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></scrip
t>
<script src="/webjars/bootstrap/4.5.2/js/bootstrap.min.js"></script>
```

Lembrando que as dependência do webjars já estão no pom.

3. modificar o código do botão excluir em lista.html

```
<button th:id="${'btn_tarefas/excluir/' + tarefa.id}" type="button"
class="btn btn-danger btn-sm"   data-toggle="modal"  data-target="#myModal">Excluir
</button>
```

4. Colocar o fragmento do modal e do script

```
<div th:replace="~{fragments/modal :: modal}"></div>
<div th:replace="~{fragments/script}"></div>
```

Código da lista.html

```
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Gerenciador de Tarefas</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<link href="/webjars/bootstrap/css/bootstrap.min.css" rel="stylesheet" />
<link rel="stylesheet" type="text/css" media="all"
      href="../../../css/style.css" />
<script src="/webjars/jquery/3.5.1/jquery.slim.min.js"></script>
<script
      src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"><
/script>
<script src="/webjars/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
      <h1>Lista de Tarefas</h1>
      <div th:replace="~{fragments/alert}"></div>
      <table>
            <thead>
                  <tr>
                        <th>Identificação da Tarefa</th>
                        <th>Nome da Tarefa</th>
                        <th>Data entrega</th>
                        <th>Responsável</th>
                  </tr>
            </thead>
            <tbody>
                  <tr th:each="tarefa : ${tarefas}">
                        <td th:text="${tarefa.id}">1</td>
                        <td th:text="${tarefa.nome}">Criação de conteúdo</td>
                        <td th:text="${tarefa.dataEntrega}">15/08/2024</td>
                        <td th:text="${tarefa.Responsavel}">Luciane</td>
                        <td colspan="2"><a class="btn btn-info btn-sm"
                              th:href="@{/tarefas/editar/{id} (id=${tarefa.id}) }"
role="button">
                                    <span class="oi oi-brush" title="Editar" aria-
hidden="true">Editar</span>
                        </a>
      <button th:id="${'btn_tarefas/excluir/' + tarefa.id}" type="button"
class="btn btn-danger btn-sm"   data-toggle="modal"  data-target="#myModal">Excluir
</button></td>
                  </tr>
            </tbody>
      </table>
      <div th:replace="~{fragments/modal :: modal}"></div>
      <p>
            <a href="../">Return to home</a>
      </p>
      <div>&copy; 2024 modelo estático</div>
      <div th:replace="~{fragments/script}"></div>
</body>
</html>
```