

# UNIVERSIDADE FEDERAL DE SÃO CARLOS

Guilherme Fumagali Marques - 792182  
Guilherme Silva de Camargo - 792183  
Rodrigo Henrique Amaral - 792241  
Vinicius Gabriel Nanini da Silva - 795181

## Miniprojeto 2

Aplicação web MVC - Inclusão e visualização de pedidos de lanches

Relatório do Projeto - Miniprojeto 2  
CCSo - Desenvolvimento Web  
Orientação Profa. Dra. Luciana Zaina

Sorocaba  
06/03/2023

# Índice

<b>1 - Introdução</b>	<b>3</b>
<b>2 - Visão da estrutura do BD</b>	<b>3</b>
2.1 - MER	3
2.2 - Inicialização do banco de dados	4
<b>3 - Arquitetura MVC da aplicação</b>	<b>5</b>
3.1 - Visão geral	5
3.1.1 - Controller	5
3.1.2 - View	5
3.1.3 - Model	6
<b>4 - Conclusão</b>	<b>6</b>

# 1 - Introdução

O trabalho tem como objetivo o desenvolvimento de uma aplicação web para venda de lanches, utilizando o padrão MVC. No back-end, o foco da aplicação é o desenvolvimento de um servlet, que deve receber requisições, usar DAOs para fazer intermédio com os dados e o banco de dados e retornar o conteúdo solicitado. Por outro lado, no front-end a principal atenção está no uso do AJAX, para atualizar automaticamente os dados na tela.

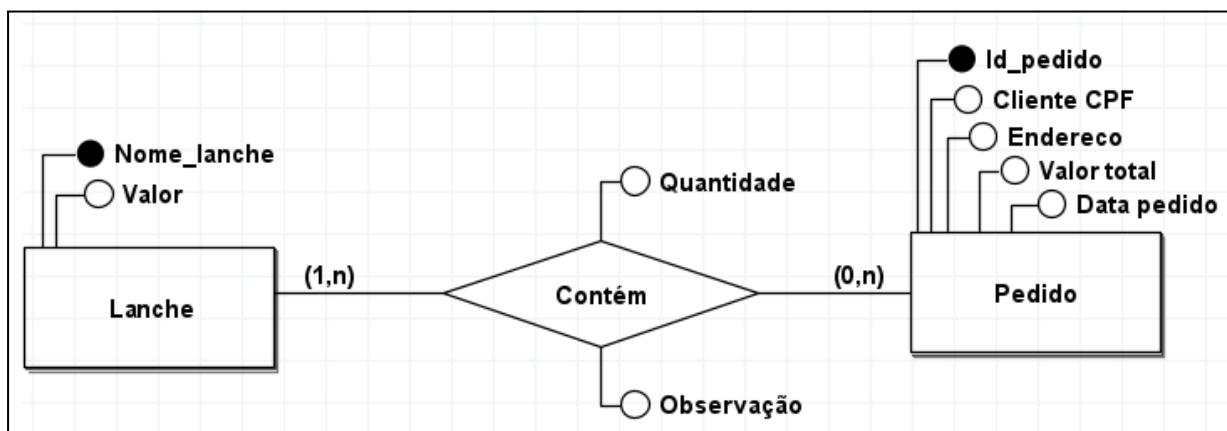
Ademais, os outros elementos do front-end não fazem parte do foco de desenvolvimento, tendo apenas uma simples interface, com as duas telas solicitadas: a inclusão de pedidos, feito pelo lado de quem prepara o lanche, e a visualização dos pedidos, que é feito pela parte do cliente.

## 2 - Visão da estrutura do BD

### 2.1 - MER

A seguir temos a representação do modelo relacional, explicitada na imagem abaixo.

Figura 1 - MER



A entidade *lanche* contém atributos de nome e valor, para guardar as informações necessárias para cada lanche, e a entidade *pedido* tem informações do cliente, cpf, endereço, id do pedido, valor total e data. Por fim, a entidade-relacionamento *contém* entre *lanche* e *pedido* guarda as informações sobre a quantidade e observações acerca do pedido, tal como retirada e adição de novos ingredientes.

## 2.2 - Inicialização do banco de dados

Primeiramente, criaremos um banco de dados chamado “*dev\_lanche*” (Figura 2).

Figura 2 - Criar banco de dados (phpMyAdmin)



The screenshot shows the 'Base de Dados' (Database) section of phpMyAdmin. It features a button labeled 'Criar base de dados' with a plus icon. Below this, there is a text input field containing 'dev\_lanche' and a dropdown menu set to 'latin1\_swedish\_ci'. A 'Criar' button is positioned to the right of the dropdown.

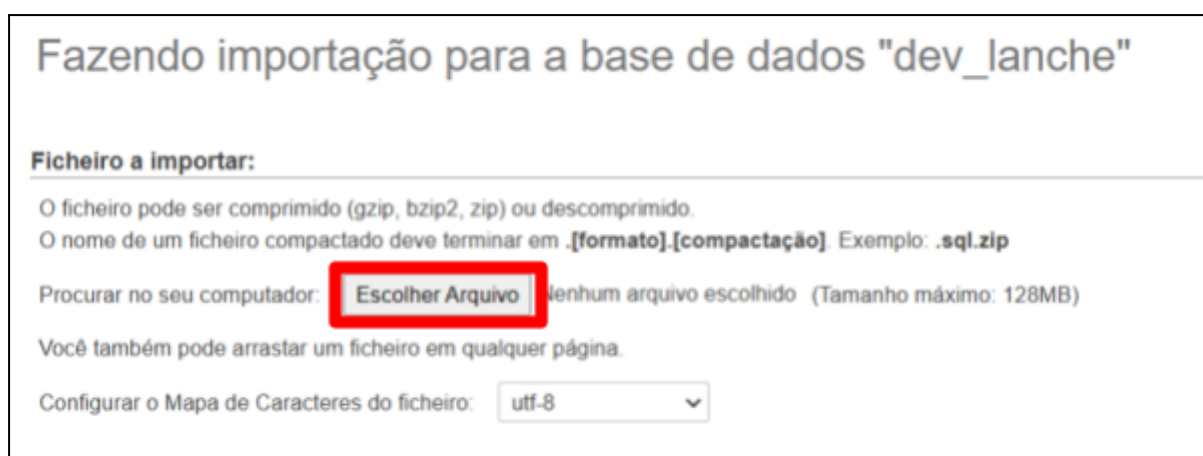
Após a criação do banco de dados é necessário importar a base de dados, selecionando a opção como descrito abaixo (Figura 3).

Figura 3 - Importar dados (phpMyAdmin)



Assim, será redirecionado para fazer a escolha do arquivo que deseja importar (Figura 4), no contexto, o arquivo “*dev\_lanche.sql*” estará na pasta do projeto.

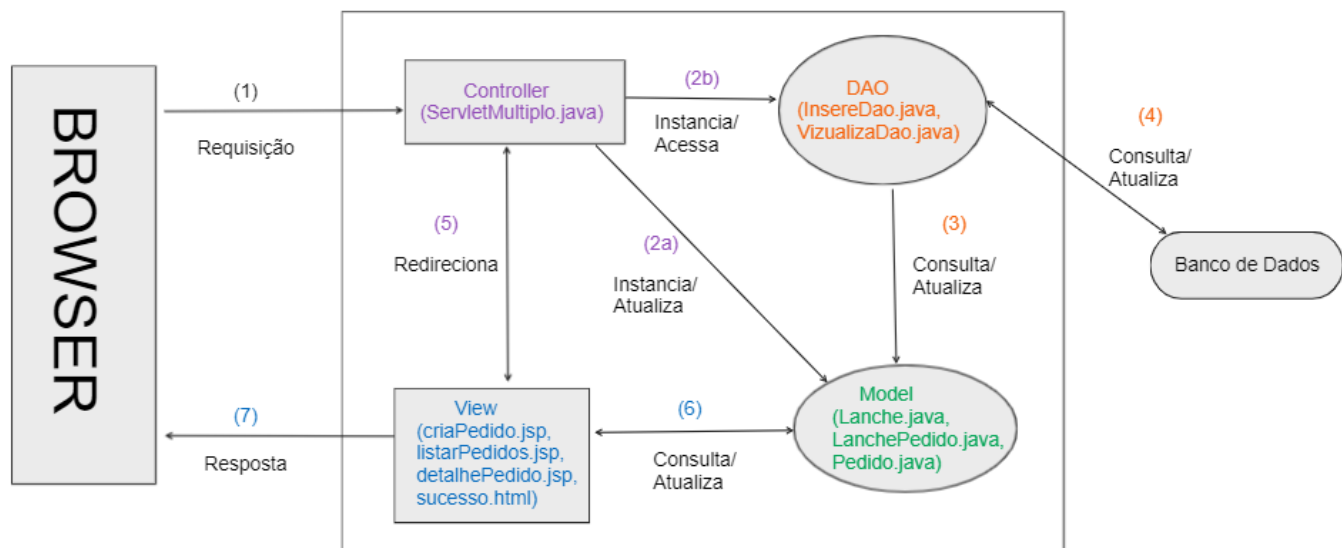
Figura 4 - Selecionar arquivo .sql (phpMyAdmin)



The screenshot shows the 'Fazendo importação para a base de dados "dev\_lanche"' page. Under the heading 'Ficheiro a importar:', there is explanatory text about file formats and a note that the filename must end in a specific format. Below this, the text 'Procurar no seu computador:' is followed by a button labeled 'Escolher Arquivo', which is highlighted with a red rectangular box. To the right of the button, it says 'nenhum arquivo escolhido (Tamanho máximo: 128MB)'. At the bottom, there is a section for 'Configurar o Mapa de Caracteres do ficheiro:' with a dropdown menu currently set to 'utf-8'.

## 3 - Arquitetura MVC da aplicação

Figura 5 - Arquitetura MVC



### 3.1 - Visão geral

#### 3.1.1 - Controller

Na aplicação, temos apenas um arquivo que faz o papel de controller: “*ServletMultiplo*”.

Esse arquivo espera requisições usando método POST, que incluem em seu corpo um parâmetro “botao”, que pode assumir quatro valores distintos, que selecionam qual função do servlet se deseja acessar. Os valores podem ser:

- “*registrarPedido*”: receberá os dados dos lanches, quantidade, observações e dados do cliente e insere em ambas tabelas do banco de dados. Após isso, redireciona para uma página indicando que a operação foi realizada com sucesso para o usuário, caso nenhum erro aconteça.
- “*consultarPedidos*”: Redireciona para a view que lista os pedidos, com os dados de todos os pedidos do banco de dados.
- “*registrarLanche*”: Redireciona para a view que registra pedidos de lanche. Essa view é gerada com base nos lanches presentes no banco de dados.
- “*detalhesPedido*”: Recebe o id de um pedido, e redireciona para uma View com detalhes do mesmo, que inclui dados como todos os lanches desse pedido, através das tabelas *Pedido* e *LanchePedido*.

#### 3.1.2 - View

Temos três páginas dinâmicas desenvolvidas com jsp:

- “*criarPedido*”: recebe uma lista de objetos do tipo *Lanche* e renderiza-os na tela em formato de lista, assim o usuário pode ver e selecionar o pedido desejado. Ao fim, os dados do pedido são enviados ao servlet, com a operação de “*registrarPedido*”.

- “*sucesso.html*”: página simples em que o servlet redireciona caso o pedido seja registrado com sucesso.
- “*listarPedidos*”: recebe uma lista de objetos do tipo Pedido e renderiza no formato de uma tabela. Em cada item da tabela, tem um botão “detalhes”, que envia ao servlet, com a opção de “detalhesPedido”, o id do pedido.
- “*detalhesPedido*”: receberá os dados do pedido e dos lanches e renderiza na tela.

### 3.1.3 - Model

Contém classes com a estrutura de dados necessária para carregar os dados das três tabelas do banco de dados: *Lanche*, *Pedido* e *LanchePedido*.

## 4 - Conclusão

Concluimos que a utilização das tecnologias presentes no projeto trás uma integração de grande valor, principalmente considerando o padrão MVC, que separa de forma clara os papéis de cada arquivo do projeto.

Ademais, o projeto tem o funcionamento ao qual era esperado pelas definições do projeto.