Engenharia de Software III

Aula 4
Padrões de Arquitetura
Model-View-Controller

l.bertholdo@ifsp.edu.br

Conteúdo

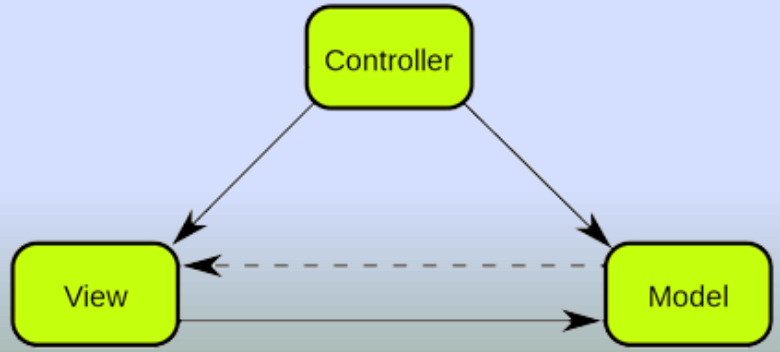
- Padrão de Arquitetura MVC
- Padrão de Projeto DAO
- JDBC – Java Database Connectivity
- Exemplo
  - Cadastro de Funcionários

l.bertholdo@ifsp.edu.br

Padrão de Arquitetura MVC

- **MVC (*Model-View-Controller*)** é um padrão de arquitetura de software cujo objetivo é a **separação** clara entre o código responsável pelo mapeamento das requisições do usuário (***controller***), o modelo de dados (***model***) e a interface gráfica (***view***) de uma aplicação.

- Assim como outros padrões de arquitetura, o padrão MVC também prioriza as noções de **separação** e **independência** de componentes de software, fundamentais para a arquitetura, pois permitem que alterações no sistema sejam feitas com menor impacto.



l.bertholdo@ifsp.edu.br

Padrão de Arquitetura MVC

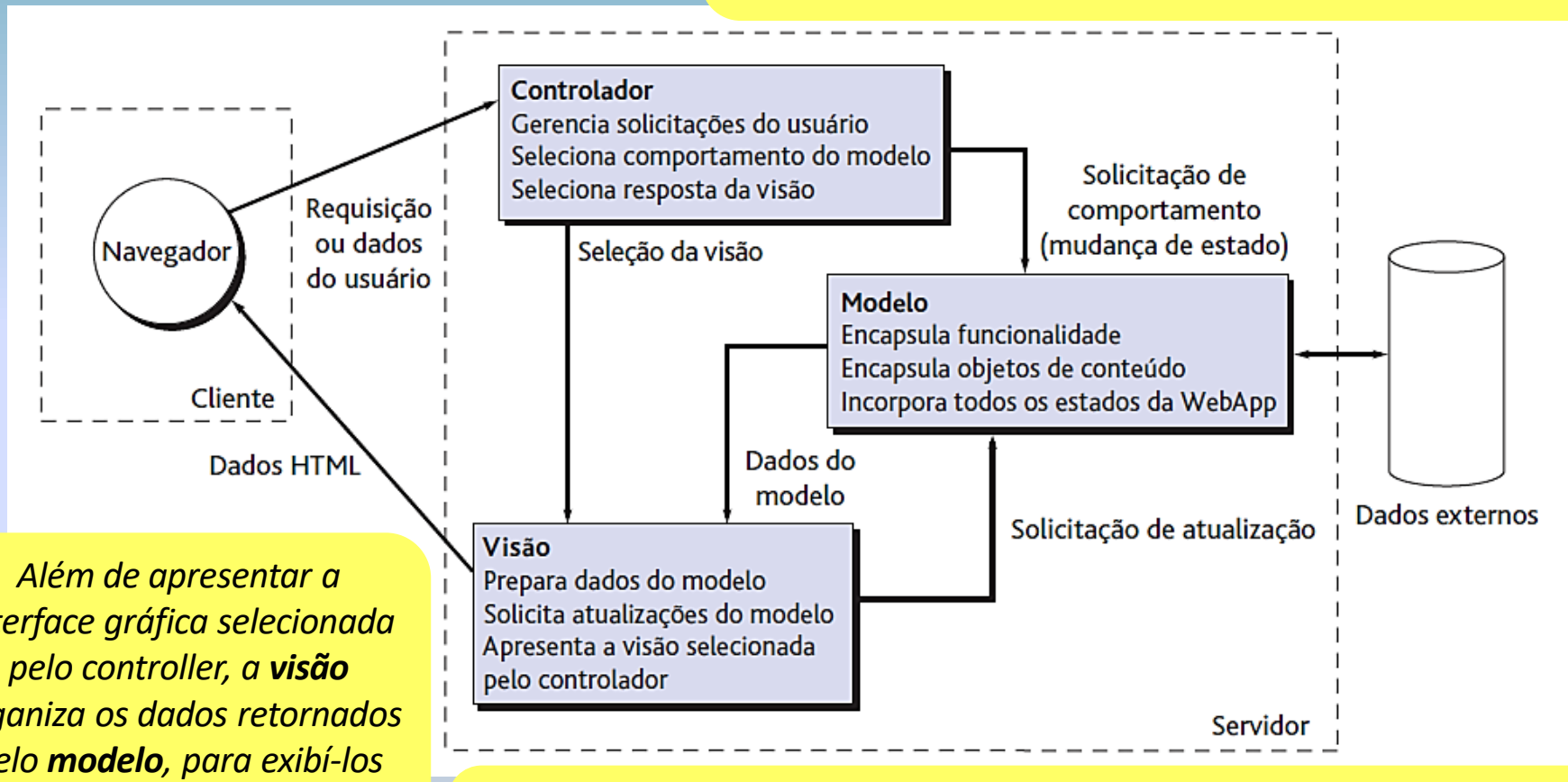
- Descrição do Padrão de Arquitetura MVC

Nome	MVC (*Model-View-Controller*)
Descrição	Separa as camadas de **apresentação** e **interação** da camada de **dados** do sistema, as quais interagem entre si: o **Modelo** gerencia os dados e as operações associadas a esses dados. A **Visão** define e gerencia como os dados são apresentados ao usuário. O **Controlador** gerencia a interação do usuário (por exemplo, teclas, cliques do mouse etc.) e passa essas interações para a Visão e o Modelo.
Exemplo	Sistemas Web em geral.
Quando é usado	• Quando há muitas maneiras de se visualizar e interagir com os dados do sistema; • Quando os requisitos de apresentação e interação do sistema são pouco detalhados, permitindo projetar apenas a camada de dados inicialmente.
Vantagens	Independência entre modelos, controles e visões do sistema, permitindo alterá-los de forma independente. Por exemplo, pode-se adicionar uma visão ou alterar uma visão existente sem quaisquer alterações na camada de dados. Melhor organização do código e gerenciamento da complexidade, o que facilita a manutenção.
Desvantagens	Requer mais tempo e esforço para analisar e modelar o sistema, além de demandar conhecimento especializado para implementação. Por isso, seus benefícios são mais nítidos para aplicações de médio e grande porte.

Padrão de Arquitetura MVC

*As requisições do usuário são gerenciadas pelo **controlador**. Ele também define a **visão** apropriada para a requisição e seleciona o comportamento relacionado junto ao **modelo**.*

Exemplo: Sistema Web genérico



*Além de apresentar a interface gráfica selecionada pelo controller, a **visão** organiza os dados retornados pelo **modelo**, para exibí-los no navegador do usuário.*

*O **modelo** executa a funcionalidade solicitada pela visão e recupera (ou atualiza) os dados necessários para atender à requisição, acessando uma fonte de dados.*

Padrão de Arquitetura MVC

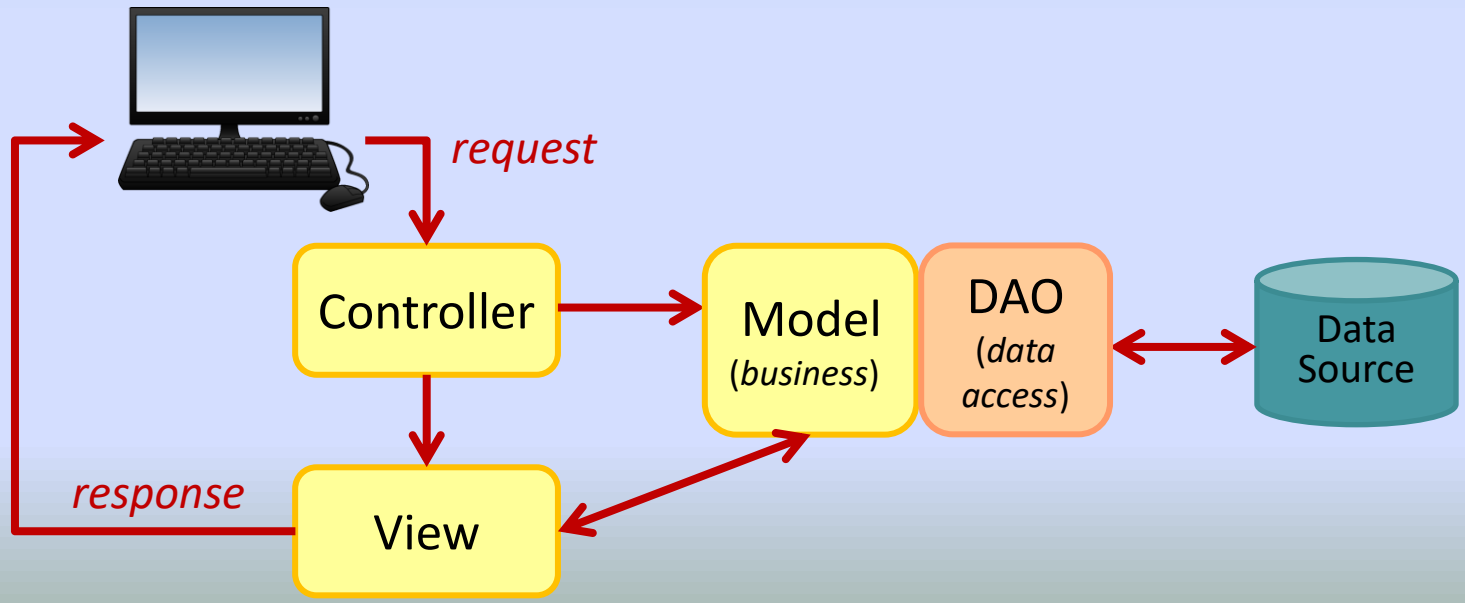
- Camada Model
  - Contém as classes de domínio que representam o **modelo de dados** da aplicação, bem como as classes responsáveis pelo **acesso e manipulação dos dados** da aplicação.
  - Normalmente, as classes **model** também definem as regras de validação e de negócio da aplicação. Exemplos:
    - **Regra de validação:** O endereço do funcionário deve ter no máximo 80 caracteres.
    - **Regra de negócio:** O salário de um funcionário não pode ser aumentado enquanto ele não completar um ano na empresa.

Padrão de Arquitetura MVC

- Camada Controller
  - Contém as classes responsáveis por encaminhar as requisições do usuário para as outras camadas da aplicação (***models*** e ***views***).
  - Por convenção, geralmente as classes ***controller*** têm a palavra "Controller" no final de seu nome. Exemplos: ClienteController, FuncionarioController, ProdutoController.
- Camada View
  - Contém as interfaces gráficas da aplicação, responsáveis por apresentar as funcionalidades ao usuário, bem como os resultados de suas requisições.

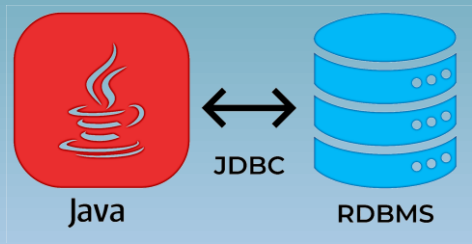
Padrão de Projeto DAO

- **DAO (*Data Access Object*)** é um padrão de projeto (*design pattern*) que permite a divisão de responsabilidades na camada Model, separando as classes de **negócios** e de **dados** da aplicação.
- As classes DAO são responsáveis exclusivamente pelo **acesso e manipulação dos dados** armazenados na fonte de dados.



JDBC – Java Database Connectivity

- API que contém os recursos necessários para que uma aplicação Java consiga acessar fontes de dados locais ou remotas.



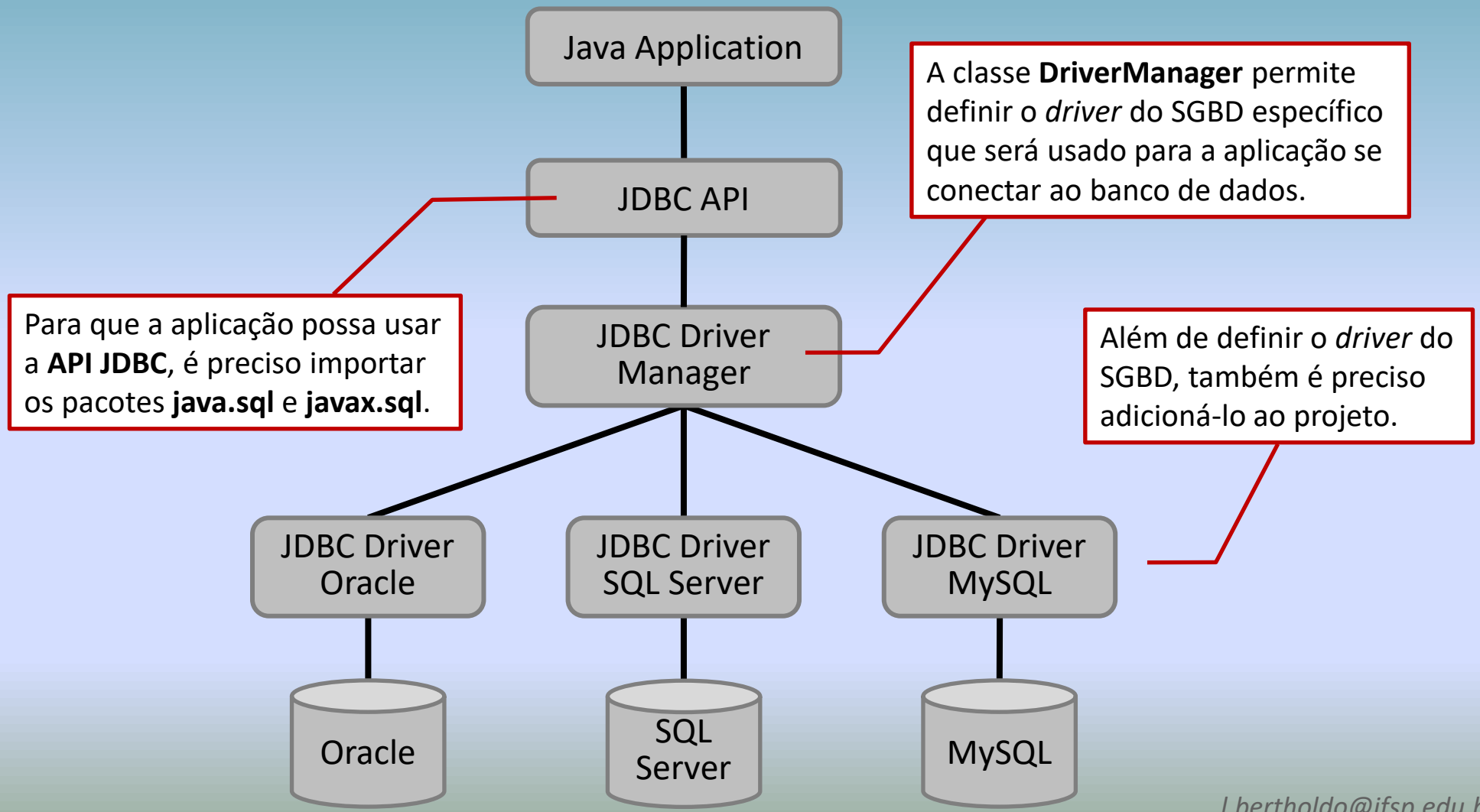
- É composta pelos pacotes **java.sql** e **javax.sql**, cujas classes e interfaces fornecem um padrão para que aplicações Java possam acessar qualquer tipo de fontes de dados (bancos de dados, arquivos, planilhas etc).

- Para acessar um SGBD, basta implementar uma interface JDBC para conexão com o banco de dados e, em seguida, usar instruções SQL *(Structured Query Language*) para acessar e manipular seus dados.

l.bertholdo@ifsp.edu.br

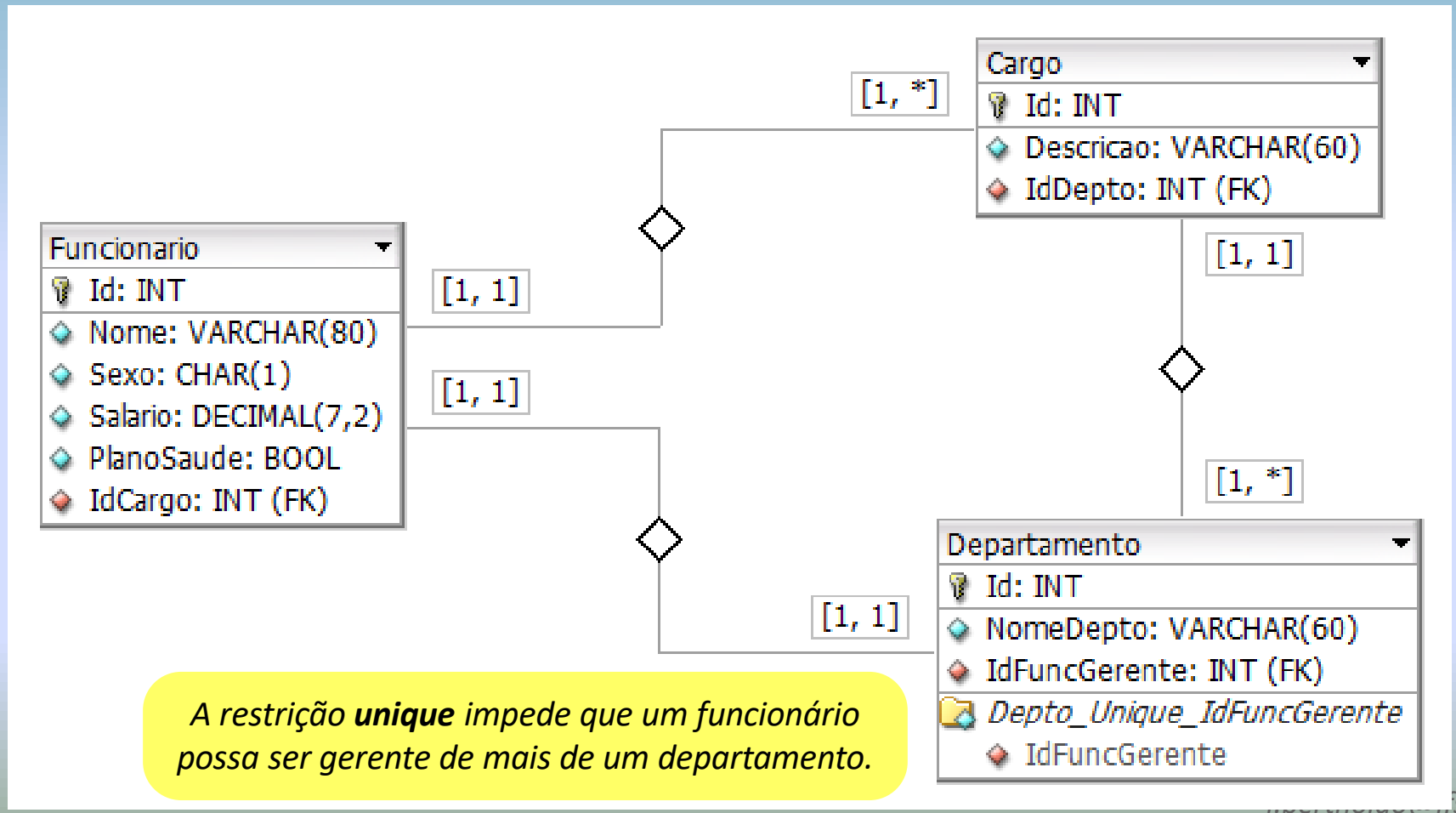
JDBC – Java Database Connectivity

- Arquitetura JDBC



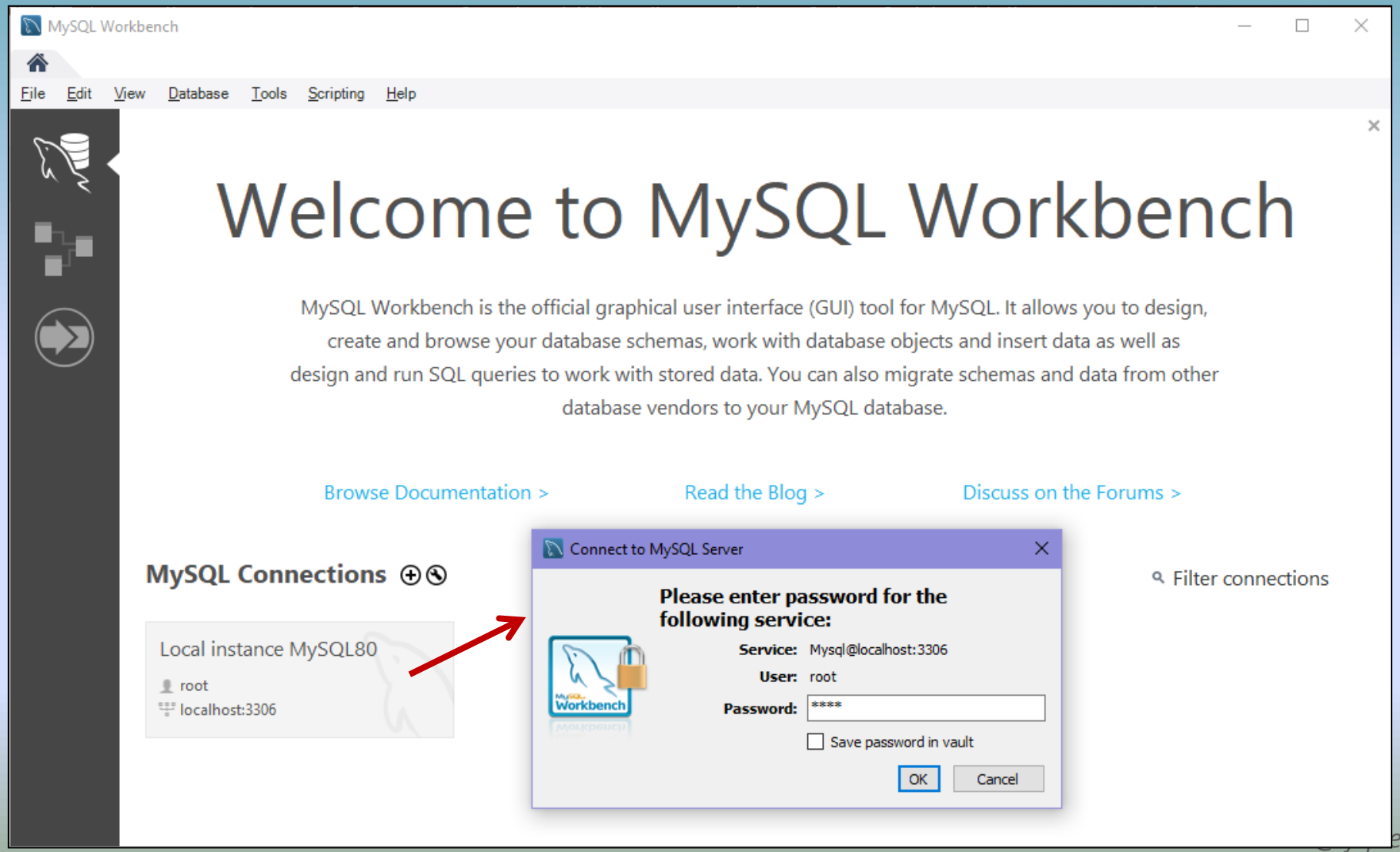
Exemplo

- Exemplo – Sistema de Departamento Pessoal
  - Diagrama Entidade-Relacionamento



Exemplo

- Criando o banco de dados **dpbd** no MySQL.



Exemplo

- Criando o banco de dados **dpbd** no MySQL.

```
CREATE DATABASE dpbd;
USE dpbd;

CREATE TABLE Departamento (
    Id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    NomeDepto VARCHAR(60) NOT NULL,
    IdFuncGerente INT
);

CREATE TABLE Cargo (
    Id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    Descricao VARCHAR(60) NOT NULL,
    IdDepto INT NOT NULL,
    FOREIGN KEY(IdDepto) REFERENCES Departamento(Id)
);

CREATE TABLE Funcionario (
    Id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    Nome VARCHAR(80) NOT NULL,
    Sexo CHAR(1) NOT NULL,
    Salario DECIMAL(7, 2) NOT NULL,
    PlanoSaude BOOL default true,
    IdCargo INT NOT NULL,
    FOREIGN KEY(IdCargo) REFERENCES Cargo(Id)
);
```

```
ALTER TABLE Departamento ADD CONSTRAINT FOREIGN KEY(IdFuncGerente) REFERENCES Funcionario(Id);
ALTER TABLE Departamento ADD CONSTRAINT UNIQUE(IdFuncGerente);
```

Exemplo

- Inserindo **departamentos** e **cargos** para carregar os valores no campo **Cargo** do formulário de cadastro de funcionários.

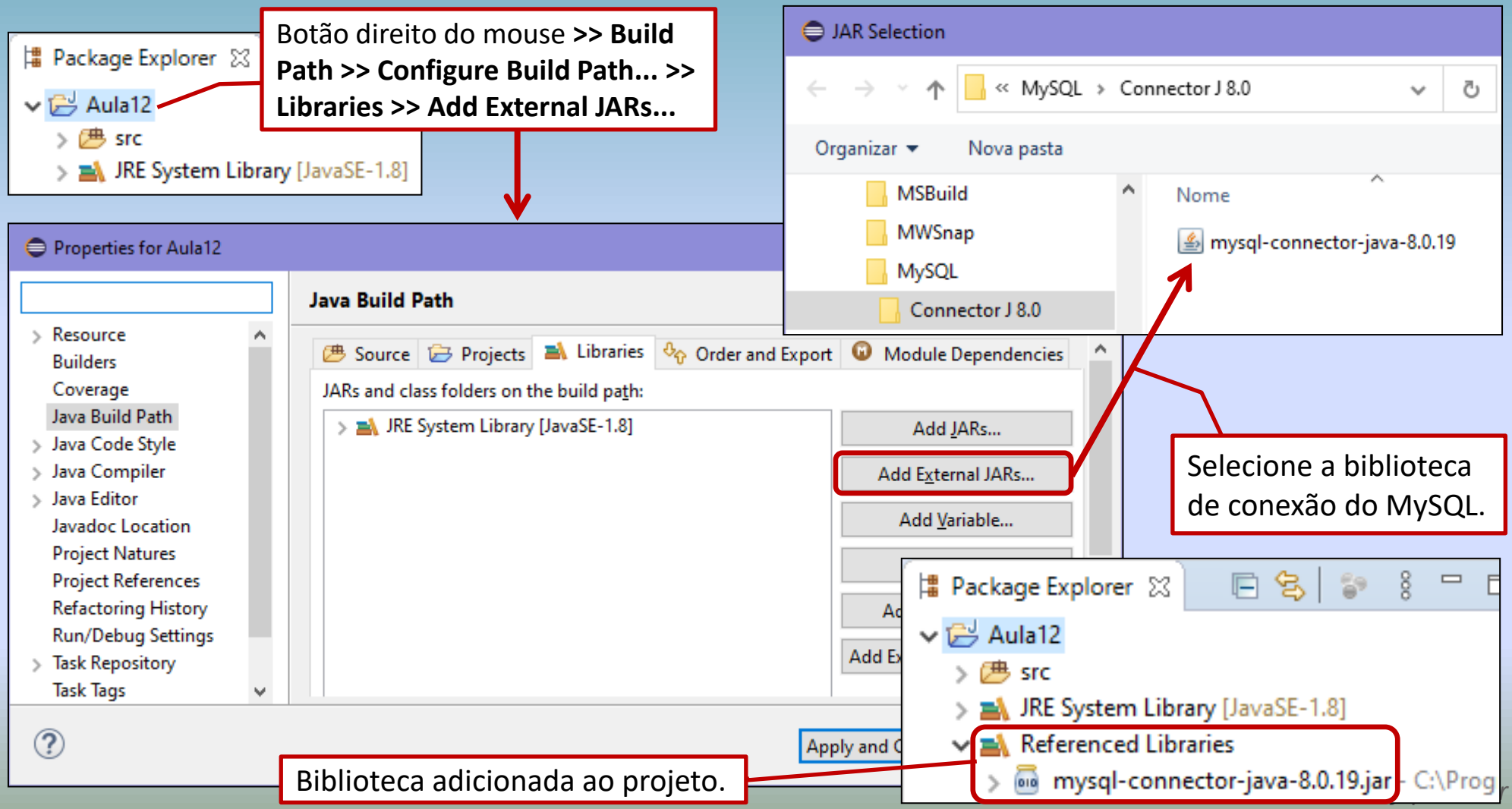
```
INSERT INTO Departamento VALUES (null, 'Gerência de Negócios', null);
INSERT INTO Departamento VALUES (null, 'Gerência de Requisitos', null);
INSERT INTO Departamento VALUES (null, 'Gerência de Desenvolvimento', null);
INSERT INTO Departamento VALUES (null, 'Gerência de Qualidade', null);

INSERT INTO Cargo VALUES (null, 'Analista de Negócios', 1);
INSERT INTO Cargo VALUES (null, 'Analista de Requisitos', 2);
INSERT INTO Cargo VALUES (null, 'Analista de Sistemas', 3);
INSERT INTO Cargo VALUES (null, 'Analista de Banco de Dados', 3);
INSERT INTO Cargo VALUES (null, 'Analista de Testes', 4);
```

Exemplo

No Windows, normalmente, a biblioteca de conexão do MySQL está disponível no caminho ***C:\Program Files (x86)\MySQL\Connector J <versão>****.*

- Para se conectar ao banco de dados, é preciso adicionar a biblioteca (ou *driver*) de conexão do MySQL ao projeto.

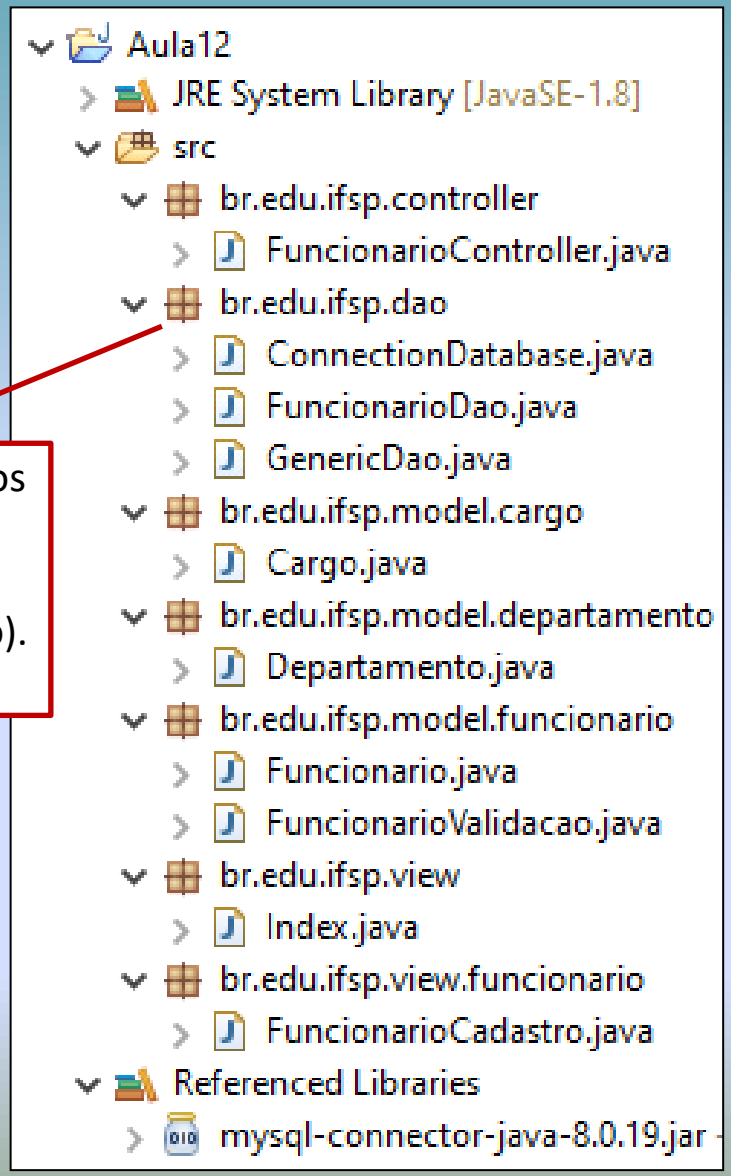


Exemplo

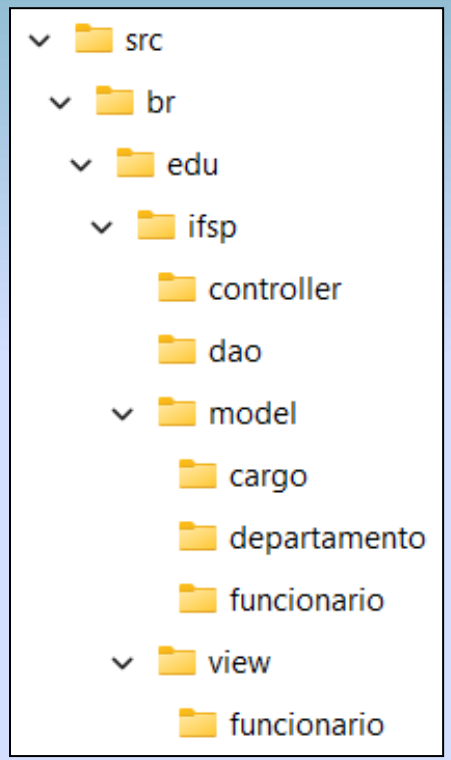
- Cadastro de Funcionários
  - Vide código-fonte

Por padrão, em projetos MVC, os diretórios do projeto são nomeados com o domínio da organização de forma invertida (país.tipo_organização.nome_organização). Por exemplo: **br.edu.ifsp**.

Estrutura do Projeto



Windows Explorer



l.bertholdo@ifsp.edu.br

Exemplo

- Aplicação em execução

Tabela **Funcionario** no MySQL

Id	Nome	Sexo	Salario	PlanoSaude	IdCargo
1	Antonio Gonçalves	M	5500.00	1	1
2	Carolina Almeida	F	4500.00	1	2
3	Carlos Santana	M	5000.00	0	2
4	João da Silva	M	4000.00	1	3
5	Henrique Pereira	M	3000.00	1	3
6	Tania Alves	F	4000.00	0	3
7	Julio Martins	M	4500.00	1	4
8	Simone Santos	F	3500.00	1	5
9	João da Silva	M	4580.00	1	3

```
CADASTROS:
1) Cadastro de Departamentos
2) Cadastro de Cargos
3) Cadastro de Funcionários
Digite uma opção (0 para sair): 3

OPERAÇÕES:
1) Cadastrar
2) Consultar
Digite uma opção: 1

CADASTRO DE FUNCIONÁRIO:
NOME: João da Silva
SEXO (Digite 'm' ou 'f'): m
SALÁRIO (R$): 4580
PLANO DE SAÚDE (Digite 's' ou 'n'): s
CARGOS CADASTRADOS:
1 - Analista de Negócios
2 - Analista de Requisitos
3 - Analista de Sistemas
4 - Analista de Banco de Dados
5 - Analista de Testes
CARGO (Digite o código do cargo): 3
Funcionário cadastrado com sucesso.

CADASTROS:
1) Cadastro de Departamentos
2) Cadastro de Cargos
3) Cadastro de Funcionários
Digite uma opção (0 para sair):
```

l.bertholdo@ifsp.edu.br

Referências

- PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software:** uma abordagem profissional. 8. ed. Porto Alegre: AMGH, 2016.

- SOMMERVILLE, Ian. **Engenharia de software**. 9. ed. São Paulo: Pearson Prentice Hall, 2011.