Adicionando na classe Model as propriedades

para gerar os campos DropDownList

/Models/FuncionarioCadastroViewModel.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations; //validações

namespace Projeto.Presentation.Models

{

public class FuncionarioCadastroViewModel

{

[Required(ErrorMessage = "Campo obrigatório")]

public string Nome { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public decimal Salario { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public DateTime DataAdmissao { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public int IdSetor { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public int IdFuncao { get; set; }

}

}

**SelectListItem**

Classe do Asp.Net MVC utilizada para gerar campos de seleção em formulários como: DropDownList, RadioButtonList, CheckBoxList, etc

Esta classe é composta de 2 propriedades principais:

* **Value**: Armazena o valor que será enviado para o servidor quando o campo for selecionado.
* **Text**: Exibe o texto que é mostrado no campo para o usuario

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations; //validações

using System.Web.Mvc;

using Projeto.Entities;

using Projeto.BLL;

namespace Projeto.Presentation.Models

{

public class FuncionarioCadastroViewModel

{

[Required(ErrorMessage = "Campo obrigatório")]

public string Nome { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public decimal Salario { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public DateTime DataAdmissao { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public int IdSetor { get; set; }

[Required(ErrorMessage = "Campo obrigatório")]

public int IdFuncao { get; set; }

**//propriedade para popular o campo**

**//DropDownList referente a Setor**

**public List<SelectListItem> Setores**

**{**

**get**

**{**

**List<SelectListItem> lista = new List<SelectListItem>();**

**SetorBusiness business = new SetorBusiness();**

**foreach(Setor setor in business.ConsultarSetores())**

**{**

**SelectListItem item = new SelectListItem();**

**item.Value = setor.IdSetor.ToString();**

**item.Text = setor.Nome;**

**lista.Add(item);**

**}**

**return lista;**

**}**

**}**

**//propriedade para popular o campo**

**//DropDownList referente a Função**

**public List<SelectListItem> Funcoes**

**{**

**get**

**{**

**List<SelectListItem> lista = new List<SelectListItem>();**

**FuncaoBusiness business = new FuncaoBusiness();**

**foreach(Funcao funcao in business.ConsultarFuncoes())**

**{**

**SelectListItem item = new SelectListItem();**

**item.Value = funcao.IdFuncao.ToString();**

**item.Text = funcao.Nome;**

**lista.Add(item);**

**}**

**return lista;**

**}**

**}**

}

}

Para que a página de cadastro de Funcionario possa exibir os campos DropDownList carregados de forma correta, é necessário enviar para a página uma instância da classe **FuncionarioCadastroViewModel** quando a página for carregada.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Projeto.Entities; //importando

using Projeto.BLL; //importando

using Projeto.Presentation.Models; //importando

namespace Projeto.Presentation.Controllers

{

public class FuncionarioController : Controller

{

//atributo

private FuncionarioBusiness business;

//construtor -> ctor + 2x[tab]

public FuncionarioController()

{

business = new FuncionarioBusiness();

}

**// GET: Funcionario/Cadastro**

**public ActionResult Cadastro()**

**{**

**return View(new FuncionarioCadastroViewModel());**

**}**

**// POST: Funcionario/Cadastro**

**[HttpPost] //método recebe SUBMIT do formulário**

**public ActionResult Cadastro(FuncionarioCadastroViewModel model)**

**{**

**return View(new FuncionarioCadastroViewModel());**

**}**

// GET: Funcionario/Consulta

public ActionResult Consulta()

{

return View();

}

}

}

**Na página, iremos desenhar os campos DropDownList:**

@model Projeto.Presentation.Models.FuncionarioCadastroViewModel

@{

ViewBag.Title = "Cadastro";

Layout = "~/Views/Shared/Layout.cshtml";

}

<h4>Cadastro de Funcionários</h4>

<a href="/Home/Index">Página inicial</a>

<hr/>

<div class="row">

<div class="col-md-4">

@using (Html.BeginForm())

{

<label>Nome do Funcionário:</label>

@Html.TextBoxFor(model => model.Nome,

new { @class = "form-control" })

<span class="text-danger">

@Html.ValidationMessageFor(model => model.Nome)

</span>

<br/>

<label>Salário:</label>

@Html.TextBoxFor(model => model.Salario,

new { @class = "form-control" })

<span class="text-danger">

@Html.ValidationMessageFor(model => model.Salario)

</span>

<br />

<label>Data de Admissão:</label>

@Html.TextBoxFor(model => model.DataAdmissao,

new { @class = "form-control", @type = "date" })

<span class="text-danger">

@Html.ValidationMessageFor(model => model.DataAdmissao)

</span>

<br />

<label>Selecione o Setor:</label>

@Html.DropDownListFor(model => model.IdSetor, Model.Setores,

"Escolha uma opção", new { @class = "form-control" })

<span class="text-danger">

@Html.ValidationMessageFor(model => model.IdSetor)

</span>

<br/>

<label>Selecione a Função:</label>

@Html.DropDownListFor(model => model.IdFuncao, Model.Funcoes,

"Escolha uma opção", new { @class = "form-control" })

<span class="text-danger">

@Html.ValidationMessageFor(model => model.IdFuncao)

</span>

<br />

<input type="submit" value="Cadastrar Funcionário"

class="btn btn-success"/>

<br />

<br />

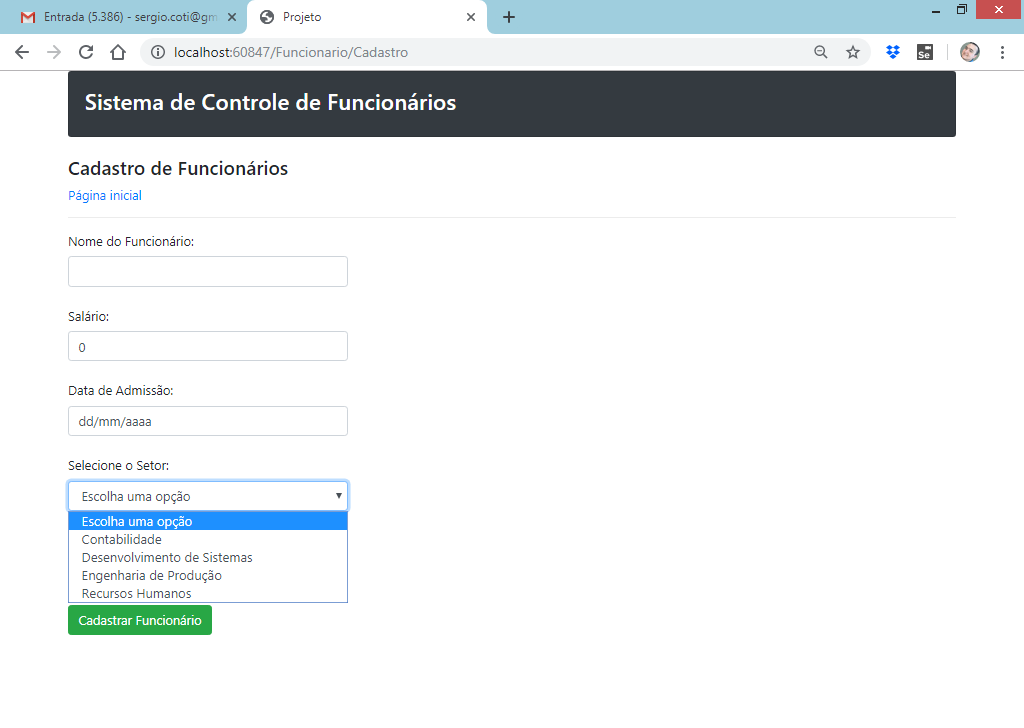
<strong>@TempData["Mensagem"]</strong>

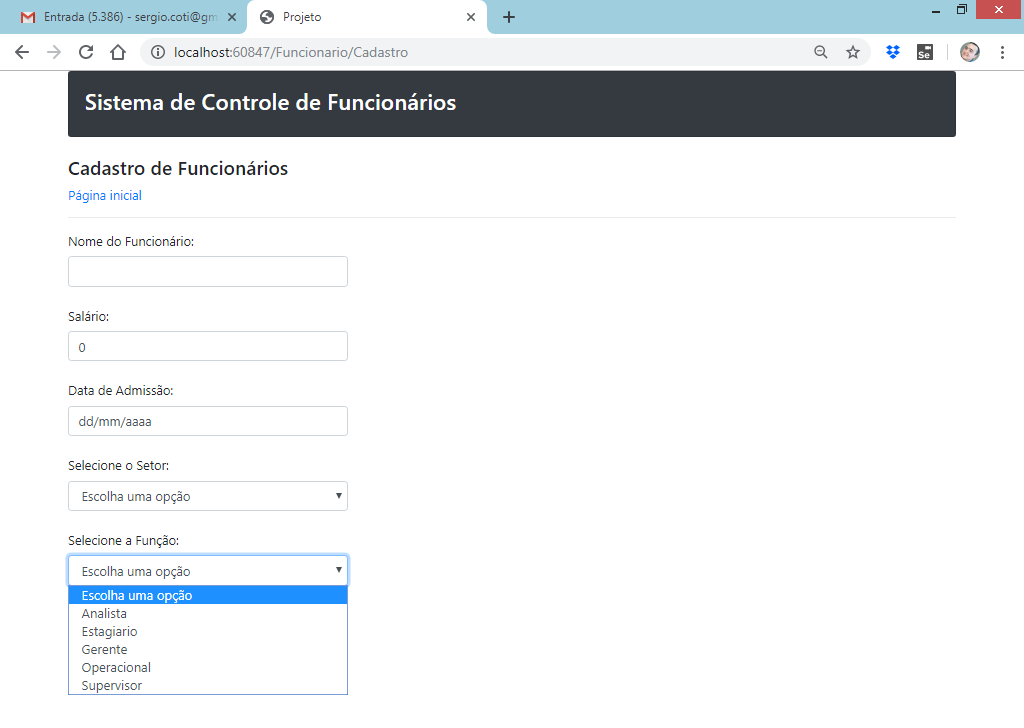
}

</div>

</div>

**Executando:**





**Implementando o cadastro do funcionário:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Projeto.Entities; //importando

using Projeto.BLL; //importando

using Projeto.Presentation.Models; //importando

namespace Projeto.Presentation.Controllers

{

public class FuncionarioController : Controller

{

//atributo

private FuncionarioBusiness business;

//construtor -> ctor + 2x[tab]

public FuncionarioController()

{

business = new FuncionarioBusiness();

}

// GET: Funcionario/Cadastro

public ActionResult Cadastro()

{

return View(new FuncionarioCadastroViewModel());

}

// POST: Funcionario/Cadastro

[HttpPost] //método recebe SUBMIT do formulário

public ActionResult Cadastro(FuncionarioCadastroViewModel model)

{

//verificar se os campos da model passaram nas validações

if (ModelState.IsValid)

{

try

{

Funcionario funcionario = new Funcionario();

funcionario.Nome = model.Nome;

funcionario.Salario = model.Salario;

funcionario.DataAdmissao = model.DataAdmissao;

funcionario.IdSetor = model.IdSetor;

funcionario.IdFuncao = model.IdFuncao;

business.CadastrarFuncionario(funcionario);

TempData["Mensagem"] = $"Funcionário

{funcionario.Nome}, cadastrado com sucesso";

ModelState.Clear();

}

catch(Exception e)

{

TempData["Mensagem"] = e.Message;

}

}

return View(new FuncionarioCadastroViewModel());

}

// GET: Funcionario/Consulta

public ActionResult Consulta()

{

return View();

}

}

}

---------------------------

**Criando a página de**

**consulta de funcionários:**

@{

ViewBag.Title = "Consulta";

Layout = "~/Views/Shared/Layout.cshtml";

}

<h4>Consulta de Funcionários</h4>

<a href="/Home/Index">Página inicial</a>

<hr/>

<div>

<strong>@TempData["Mensagem"]</strong>

</div>

<table class="table table-bordered table-hover table-striped">

<thead>

<tr>

<th class="bg-info text-white">Código</th>

<th class="bg-info text-white">Nome do Funcionário</th>

<th class="bg-info text-white">Salário</th>

<th class="bg-info text-white">Data de Admissão</th>

<th class="bg-info text-white">Setor</th>

<th class="bg-info text-white">Função</th>

<th class="bg-info text-white" width="200">Operações</th>

</tr>

</thead>

<tbody>

</tbody>

<tfoot>

<tr>

<td colspan="7">

Quantidade de registros:

</td>

</tr>

</tfoot>

</table>

Resultado:

<http://localhost:60847/Funcionario/Consulta>



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient; //importando

using System.Configuration; //importando

using Projeto.Entities; //importando

using Dapper; //importando

namespace Projeto.DAL

{

public class FuncionarioRepository

{

//atributo

private string connectionString;

//construtor

public FuncionarioRepository()

{

connectionString = ConfigurationManager.ConnectionStrings

["projeto"].ConnectionString;

}

//método para inserir um funcionario na base de dados

public void Insert(Funcionario funcionario)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "insert into Funcionario(Nome, Salario, "

+ "DataAdmissao, IdSetor, IdFuncao) "

+ "values(@Nome, @Salario, @DataAdmissao, "

+ "@IdSetor, @IdFuncao)";

conn.Execute(query, funcionario);

}

}

//método para atualizar um funcionario na base de dados

public void Update(Funcionario funcionario)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "update Funcionario set Nome = @Nome,

Salario = @Salario, "

+ "DataAdmissao = @DataAdmissao,

IdSetor = @IdSetor, "

+ "IdFuncao = @IdFuncao

where IdFuncionario = @IdFuncionario";

conn.Execute(query, funcionario);

}

}

//método para excluir um funcionário na base de dados

public void Delete(int id)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "delete from Funcionario

where IdFuncionario = @IdFuncionario";

conn.Execute(query, new { IdFuncionario = id });

}

}

//método para listar todos os funcionarios da base de dados

public List<Funcionario> SelectAll()

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "select \* from Funcionario f "

+ "inner join Funcao fn on fn.IdFuncao = f.IdFuncao "

+ "inner join Setor s on s.IdSetor = f.IdSetor";

return conn.Query(query,

(Funcionario funcionario, Funcao funcao, Setor setor) =>

{

funcionario.Funcao = funcao;

funcionario.Setor = setor;

return funcionario;

},

splitOn: "IdFuncao, IdSetor")

.ToList();

}

}

//método para listar todos os funcionarios pelo nome

public List<Funcionario> SelectAllByNome(string nome)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "select \* from Funcionario f "

+ "inner join Funcao fn on fn.IdFuncao = f.IdFuncao "

+ "inner join Setor s on s.IdSetor = f.IdSetor "

+ "where f.Nome like @Nome";

return conn.Query(query,

(Funcionario funcionario, Funcao funcao, Setor setor) =>

{

funcionario.Funcao = funcao;

funcionario.Setor = setor;

return funcionario;

},

new { Nome = $"%{nome}%" },

splitOn: "IdFuncao, IdSetor")

.ToList();

}

}

//método para obter 1 funcionario pelo id

public Funcionario SelectById(int id)

{

using (SqlConnection conn = new SqlConnection(connectionString))

{

string query = "select \* from Funcionario f "

+ "inner join Funcao fn on fn.IdFuncao = f.IdFuncao "

+ "inner join Setor s on s.IdSetor = f.IdSetor "

+ "where f.IdFuncionario = @IdFuncionario";

return conn.Query(query,

(Funcionario funcionario, Funcao funcao, Setor setor) =>

{

funcionario.Funcao = funcao;

funcionario.Setor = setor;

return funcionario;

},

new { IdFuncionario = id }, //filtros

splitOn: "IdFuncao, IdSetor")

.SingleOrDefault();

}

}

}

}

Camada de Regras de Negócio:

**FuncionarioBusiness.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto.Entities; //importando

using Projeto.DAL; //importando

namespace Projeto.BLL

{

public class FuncionarioBusiness

{

//atributo

private FuncionarioRepository repository;

//construtor -> ctor + 2x[tab]

public FuncionarioBusiness()

{

repository = new FuncionarioRepository();

}

//método para cadastrar funcionario

public void CadastrarFuncionario(Funcionario funcionario)

{

repository.Insert(funcionario);

}

//método para atualizar funcionario

public void AtualizarFuncionario(Funcionario funcionario)

{

repository.Update(funcionario);

}

//método para excluir funcionario

public void ExcluirFuncionario(int id)

{

repository.Delete(id);

}

//método para listar todos os funcionarios

public List<Funcionario> ConsultarFuncionarios()

{

return repository.SelectAll();

}

//método para listar todos os funcionarios por nome

public List<Funcionario> ConsultarFuncionariosPorNome(string nome)

{

return repository.SelectAllByNome(nome);

}

//método para obter 1 funcionário pelo id

public Funcionario ConsultarFuncionarioPorId(int id)

{

return repository.SelectById(id);

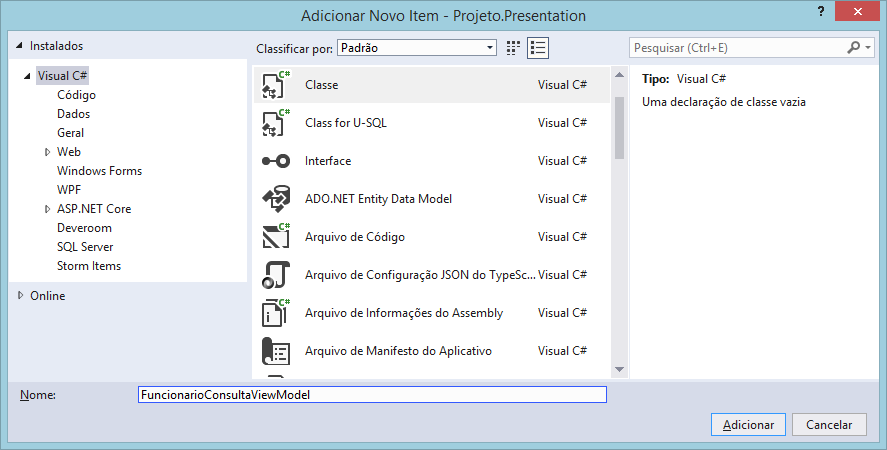
}

}

}

**Criando a classe de modelo para realizar a consulta de funcionários:**

/Models/FuncionarioConsultaViewModel.cs



using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace Projeto.Presentation.Models

{

public class FuncionarioConsultaViewModel

{

public int IdFuncionario { get; set; }

public string Nome { get; set; }

public decimal Salario { get; set; }

public DateTime DataAdmissao { get; set; }

public int IdSetor { get; set; }

public string NomeSetor { get; set; }

public int IdFuncao { get; set; }

public string NomeFuncao { get; set; }

}

}

------------------------------------------------

Exibindo os dados na página de consulta:

**FuncionarioController.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Projeto.Entities; //importando

using Projeto.BLL; //importando

using Projeto.Presentation.Models; //importando

namespace Projeto.Presentation.Controllers

{

public class FuncionarioController : Controller

{

//atributo

private FuncionarioBusiness business;

//construtor -> ctor + 2x[tab]

public FuncionarioController()

{

business = new FuncionarioBusiness();

}

// GET: Funcionario/Cadastro

public ActionResult Cadastro()

{

return View(new FuncionarioCadastroViewModel());

}

// POST: Funcionario/Cadastro

[HttpPost] //método recebe SUBMIT do formulário

public ActionResult Cadastro(FuncionarioCadastroViewModel model)

{

//verificar se os campos da model passaram nas validações

if (ModelState.IsValid)

{

try

{

Funcionario funcionario = new Funcionario();

funcionario.Nome = model.Nome;

funcionario.Salario = model.Salario;

funcionario.DataAdmissao = model.DataAdmissao;

funcionario.IdSetor = model.IdSetor;

funcionario.IdFuncao = model.IdFuncao;

business.CadastrarFuncionario(funcionario);

TempData["Mensagem"] = $"Funcionário

{funcionario.Nome}, cadastrado com sucesso";

ModelState.Clear();

}

catch(Exception e)

{

TempData["Mensagem"] = e.Message;

}

}

return View(new FuncionarioCadastroViewModel());

}

// GET: Funcionario/Consulta

**public ActionResult Consulta()**

**{**

**List<FuncionarioConsultaViewModel> lista**

**= new List<FuncionarioConsultaViewModel>();**

**try**

**{**

**foreach(Funcionario funcionario**

**in business.ConsultarFuncionarios())**

**{**

**FuncionarioConsultaViewModel model**

**= new FuncionarioConsultaViewModel();**

**model.IdFuncionario = funcionario.IdFuncionario;**

**model.Nome = funcionario.Nome;**

**model.Salario = funcionario.Salario;**

**model.DataAdmissao = funcionario.DataAdmissao;**

**model.IdSetor = funcionario.Setor.IdSetor;**

**model.NomeSetor = funcionario.Setor.Nome;**

**model.IdFuncao = funcionario.Funcao.IdFuncao;**

**model.NomeFuncao = funcionario.Funcao.Nome;**

**lista.Add(model); //adicionando na lista**

**}**

**}**

**catch(Exception e)**

**{**

**TempData["Mensagem"] = e.Message;**

**}**

**return View(lista);**

**}**

}

}

**Exibindo os dados na página:**

@model List<Projeto.Presentation.Models.FuncionarioConsultaViewModel>

@{

ViewBag.Title = "Consulta";

Layout = "~/Views/Shared/Layout.cshtml";

}

<h4>Consulta de Funcionários</h4>

<a href="/Home/Index">Página inicial</a>

<hr/>

<div>

<strong>@TempData["Mensagem"]</strong>

</div>

<table class="table table-bordered table-hover table-striped">

<thead>

<tr>

<th class="bg-info text-white">Código</th>

<th class="bg-info text-white">Nome do Funcionário</th>

<th class="bg-info text-white">Salário</th>

<th class="bg-info text-white">Data de Admissão</th>

<th class="bg-info text-white">Setor</th>

<th class="bg-info text-white">Função</th>

<th class="bg-info text-white" width="200">Operações</th>

</tr>

</thead>

<tbody>

@foreach(var item in Model)

{

<tr>

<td>@item.IdFuncionario</td>

<td>@item.Nome</td>

<td>@item.Salario.ToString("c")</td>

<td>@item.DataAdmissao.ToString("dd/MM/yyyy")</td>

<td>@item.NomeSetor</td>

<td>@item.NomeFuncao</td>

<td>

<a href="/Funcionario/Edicao/@item.IdFuncionario"

class="btn btn-primary btn-sm">

Atualizar

</a>

<a href="/Funcionario/Exclusao/@item.IdFuncionario"

class="btn btn-danger btn-sm">

Excluir

</a>

</td>

</tr>

}

</tbody>

<tfoot>

<tr>

<td colspan="7">

Quantidade de registros: @Model.Count

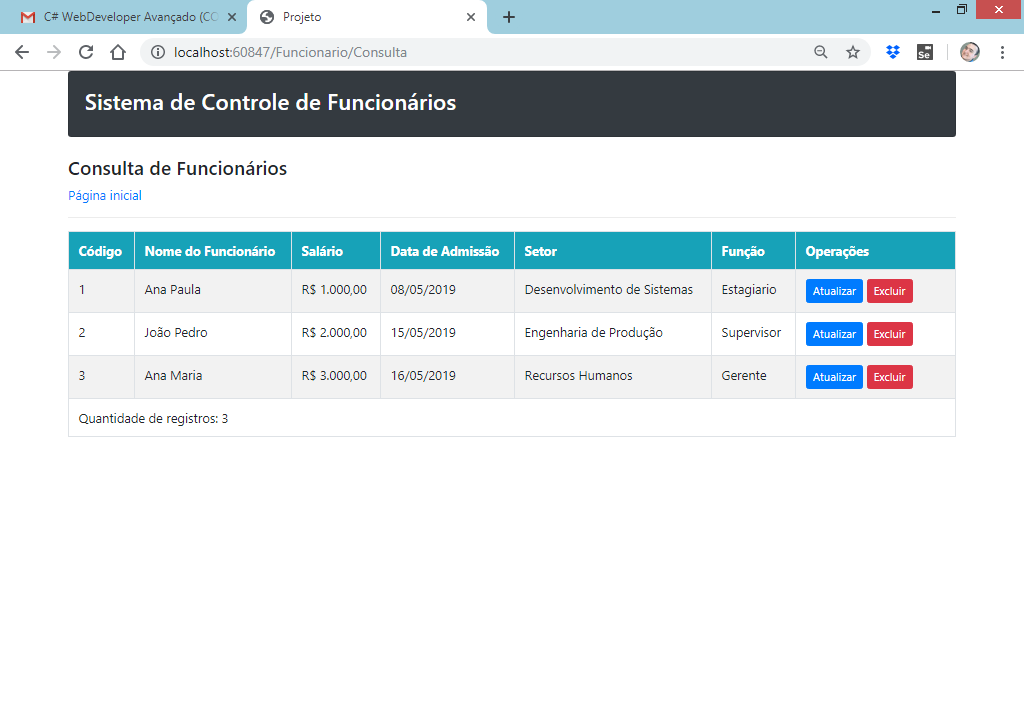
</td>

</tr>

</tfoot>

</table>

**Resultado:**



**Implementando a exclusão:**

@model List<Projeto.Presentation.Models.FuncionarioConsultaViewModel>

@{

ViewBag.Title = "Consulta";

Layout = "~/Views/Shared/Layout.cshtml";

}

<h4>Consulta de Funcionários</h4>

<a href="/Home/Index">Página inicial</a>

<hr/>

<div>

<strong>@TempData["Mensagem"]</strong>

</div>

<table class="table table-bordered table-hover table-striped">

<thead>

<tr>

<th class="bg-info text-white">Código</th>

<th class="bg-info text-white">Nome do Funcionário</th>

<th class="bg-info text-white">Salário</th>

<th class="bg-info text-white">Data de Admissão</th>

<th class="bg-info text-white">Setor</th>

<th class="bg-info text-white">Função</th>

<th class="bg-info text-white" width="200">Operações</th>

</tr>

</thead>

<tbody>

@foreach(var item in Model)

{

<tr>

<td>@item.IdFuncionario</td>

<td>@item.Nome</td>

<td>@item.Salario.ToString("c")</td>

<td>@item.DataAdmissao.ToString("dd/MM/yyyy")</td>

<td>@item.NomeSetor</td>

<td>@item.NomeFuncao</td>

<td>

<a href="/Funcionario/Edicao/@item.IdFuncionario"

class="btn btn-primary btn-sm">

Atualizar

</a>

**<a href="/Funcionario/Exclusao/@item.IdFuncionario"**

**onclick="return confirm**

**('Deseja excluir este funcionário?');"**

**class="btn btn-danger btn-sm">**

**Excluir**

**</a>**

</td>

</tr>

}

</tbody>

<tfoot>

<tr>

<td colspan="7">

Quantidade de registros: @Model.Count

</td>

</tr>

</tfoot>

</table>

**No controller:**

/FuncionarioController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Projeto.Entities; //importando

using Projeto.BLL; //importando

using Projeto.Presentation.Models; //importando

namespace Projeto.Presentation.Controllers

{

public class FuncionarioController : Controller

{

//atributo

private FuncionarioBusiness business;

//construtor -> ctor + 2x[tab]

public FuncionarioController()

{

business = new FuncionarioBusiness();

}

// GET: Funcionario/Cadastro

public ActionResult Cadastro()

{

return View(new FuncionarioCadastroViewModel());

}

// POST: Funcionario/Cadastro

[HttpPost] //método recebe SUBMIT do formulário

public ActionResult Cadastro(FuncionarioCadastroViewModel model)

{

//verificar se os campos da model passaram nas validações

if (ModelState.IsValid)

{

try

{

Funcionario funcionario = new Funcionario();

funcionario.Nome = model.Nome;

funcionario.Salario = model.Salario;

funcionario.DataAdmissao = model.DataAdmissao;

funcionario.IdSetor = model.IdSetor;

funcionario.IdFuncao = model.IdFuncao;

business.CadastrarFuncionario(funcionario);

TempData["Mensagem"] = $"Funcionário

{funcionario.Nome}, cadastrado com sucesso";

ModelState.Clear();

}

catch(Exception e)

{

TempData["Mensagem"] = e.Message;

}

}

return View(new FuncionarioCadastroViewModel());

}

// GET: Funcionario/Consulta

public ActionResult Consulta()

{

List<FuncionarioConsultaViewModel> lista

= new List<FuncionarioConsultaViewModel>();

try

{

//varrer os funcionarios obtidos na base de dados

foreach(Funcionario funcionario

in business.ConsultarFuncionarios())

{

FuncionarioConsultaViewModel model

= new FuncionarioConsultaViewModel();

model.IdFuncionario = funcionario.IdFuncionario;

model.Nome = funcionario.Nome;

model.Salario = funcionario.Salario;

model.DataAdmissao = funcionario.DataAdmissao;

model.IdSetor = funcionario.Setor.IdSetor;

model.NomeSetor = funcionario.Setor.Nome;

model.IdFuncao = funcionario.Funcao.IdFuncao;

model.NomeFuncao = funcionario.Funcao.Nome;

lista.Add(model); //adicionando na lista

}

}

catch(Exception e)

{

TempData["Mensagem"] = e.Message;

}

return View(lista);

}

**public ActionResult Exclusao(int id)**

**{**

**try**

**{**

**business.ExcluirFuncionario(id);**

**TempData["Mensagem"] = "Funcionário excluído com sucesso.";**

**}**

**catch(Exception e)**

**{**

**TempData["Mensagem"] = e.Message;**

**}**

**//redirecionamento..**

**return RedirectToAction("Consulta");**

**}**

}

}

**Resultado:**

