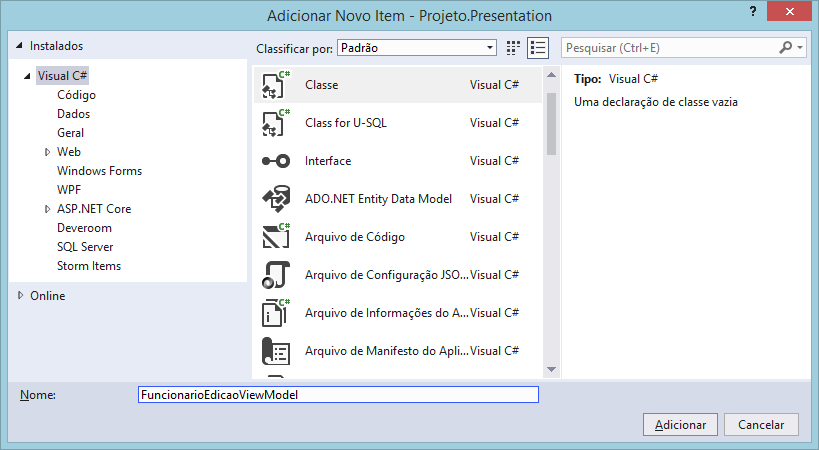
**Criando uma classe ViewModel para edição do funcionário:**

/Models/FuncionarioEdicaoViewModel.cs



using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations;

using System.Web.Mvc;

using Projeto.BLL;

using Projeto.Entities;

namespace Projeto.Presentation.Models

{

public class FuncionarioEdicaoViewModel

{

public int IdFuncionario { get; set; }

[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;

}

}

}

}

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

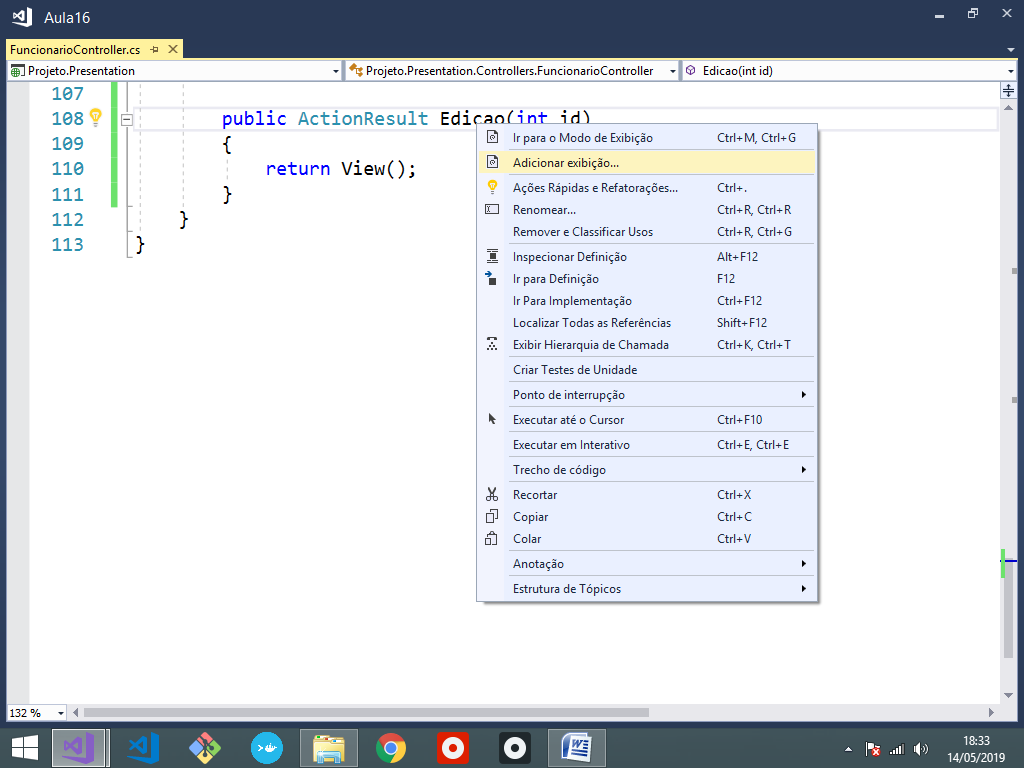
**Criando a página de Edição:**

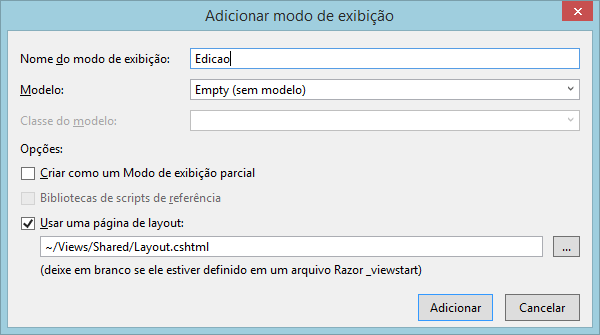
**public ActionResult Edicao(int id)**

{

return View();

}





@model Projeto.Presentation.Models.FuncionarioEdicaoViewModel

@{

ViewBag.Title = "Edicao";

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

}

<h4>Atualizar dados de Funcionários</h4>

<a href="/Funcionario/Consulta">Consulta de Funcionários</a>

<hr />

<div class="row">

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

@using (Html.BeginForm())

{

<!-- campo oculto -->

@Html.HiddenFor(model => model.IdFuncionario)

<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="Atualizar Funcionário"

class="btn btn-primary" />

<br />

<br />

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

}

</div>

</div>

**Exibindo os dados do funcionário na página de Edição:**

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;

}

//enviando a lista para a view

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");

}

public ActionResult Edicao(int id)

{

//criando um objeto viewmodel

FuncionarioEdicaoViewModel model = new FuncionarioEdicaoViewModel();

try

{

//buscar o funcionario na base de dados pelo id..

Funcionario funcionario = business.ConsultarFuncionarioPorId(id);

model.IdFuncionario = funcionario.IdFuncionario;

model.Nome = funcionario.Nome;

model.Salario = funcionario.Salario;

model.DataAdmissao = funcionario.DataAdmissao;

model.IdSetor = funcionario.IdSetor;

model.IdFuncao = funcionario.IdFuncao;

}

catch(Exception e)

{

//exibir mensagem de erro..

TempData["Mensagem"] = e.Message;

}

//enviando a model para a página

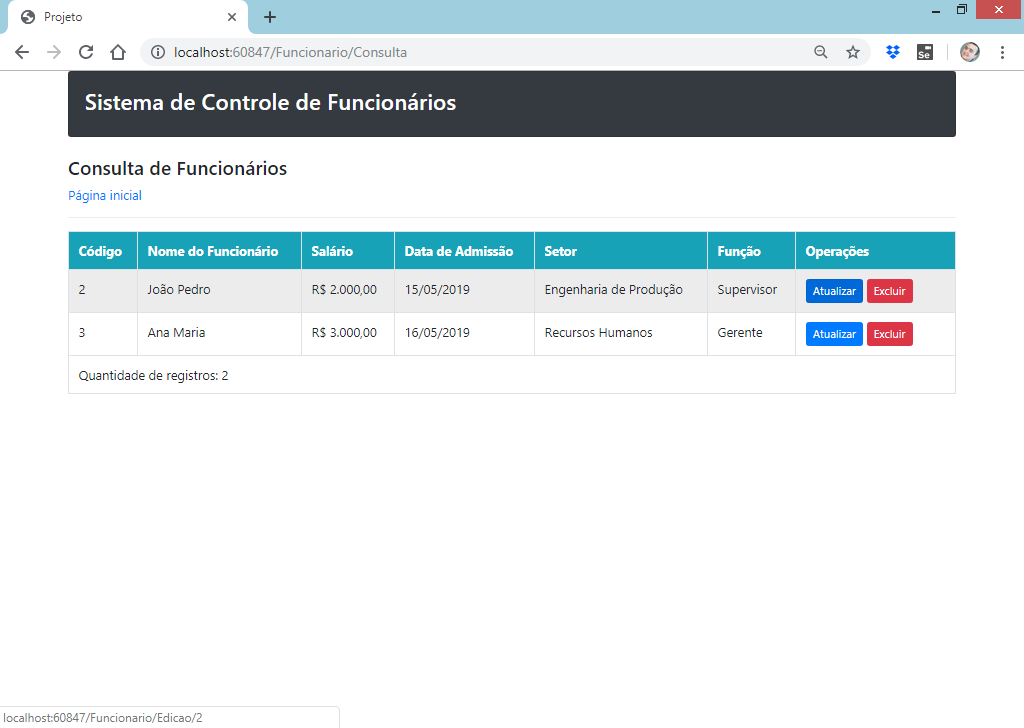
return View(model);

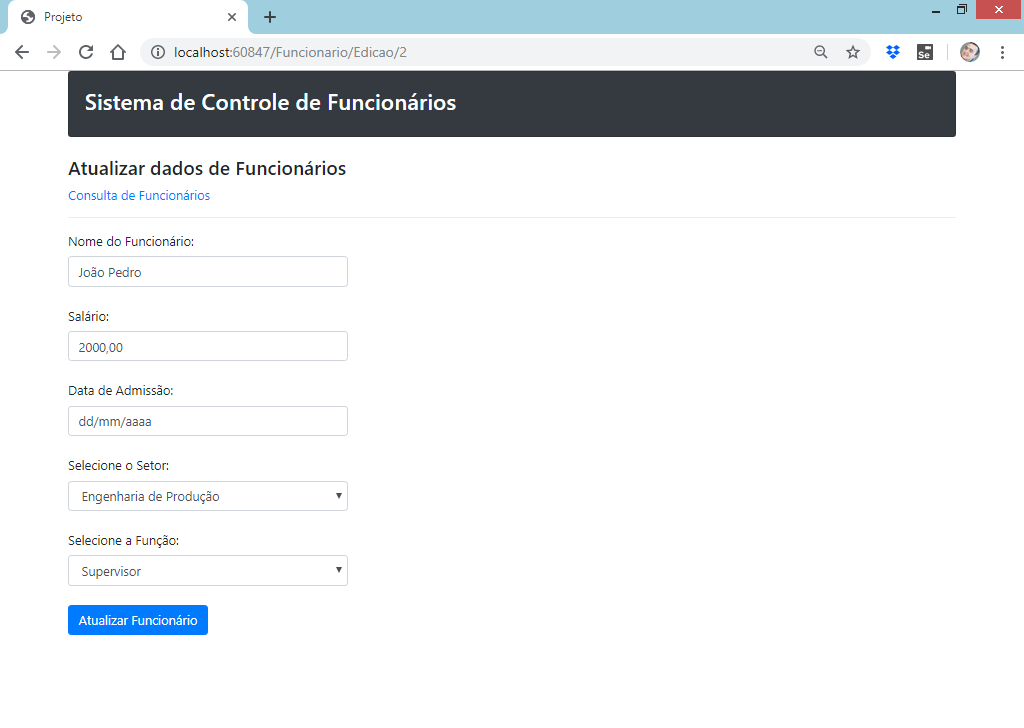
}

}

}

**Executando:**





**Exibindo a data corretamente:**

@model Projeto.Presentation.Models.FuncionarioEdicaoViewModel

@{

ViewBag.Title = "Edicao";

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

}

<h4>Atualizar dados de Funcionários</h4>

<a href="/Funcionario/Consulta">Consulta de Funcionários</a>

<hr />

<div class="row">

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

@using (Html.BeginForm())

{

<!-- campo oculto -->

@Html.HiddenFor(model => model.IdFuncionario)

<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,**

**Model.DataAdmissao.ToString("yyyy-MM-dd"),**

**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="Atualizar Funcionário"

class="btn btn-primary" />

<br />

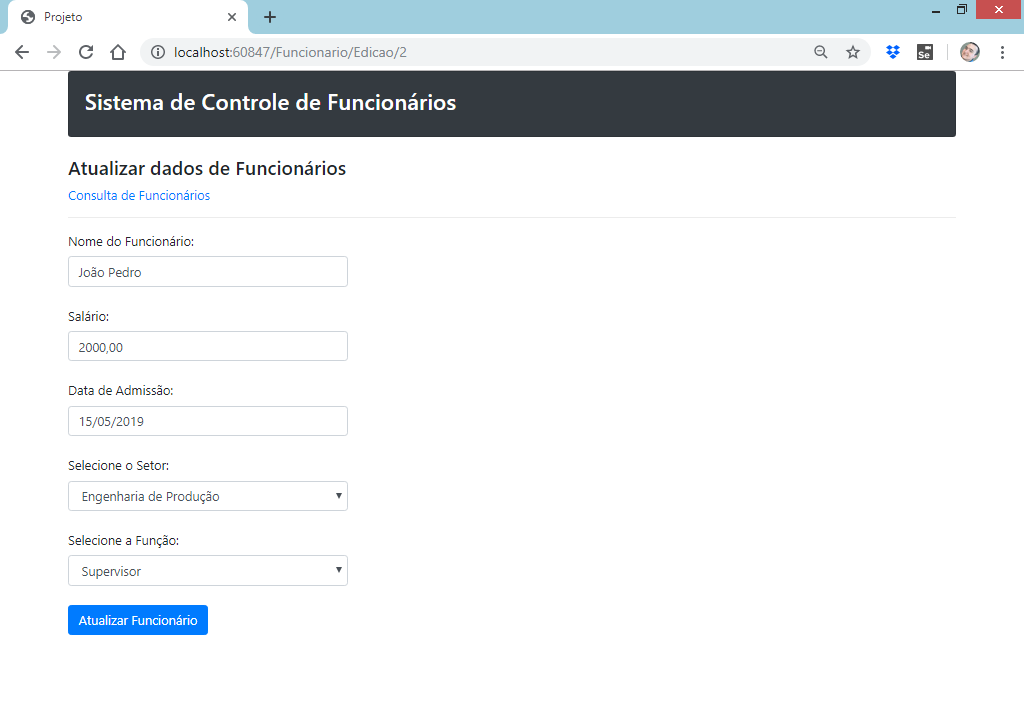
<br />

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

}

</div>

</div>



**Criando o método HttpPost para   
atualizar os dados 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()

{

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;

}

//enviando a lista para a view

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");

}

public ActionResult Edicao(int id)

{

//criando um objeto viewmodel

FuncionarioEdicaoViewModel model = new FuncionarioEdicaoViewModel();

try

{

//buscar o funcionario na base de dados pelo id..

Funcionario funcionario = business.ConsultarFuncionarioPorId(id);

model.IdFuncionario = funcionario.IdFuncionario;

model.Nome = funcionario.Nome;

model.Salario = funcionario.Salario;

model.DataAdmissao = funcionario.DataAdmissao;

model.IdSetor = funcionario.IdSetor;

model.IdFuncao = funcionario.IdFuncao;

}

catch(Exception e)

{

//exibir mensagem de erro..

TempData["Mensagem"] = e.Message;

}

//enviando a model para a página

return View(model);

}

**// POST: Funcionario/Edicao**

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

**public ActionResult Edicao(FuncionarioEdicaoViewModel model)**

**{**

**//verificar se os campos da model passaram nas validações**

**if (ModelState.IsValid)**

**{**

**try**

**{**

**Funcionario funcionario = new Funcionario();**

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

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

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

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

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

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

**business.AtualizarFuncionario(funcionario);**

**TempData["Mensagem"] = $"Funcionário {funcionario.Nome},**

**atualizado com sucesso";**

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

**}**

**catch (Exception e)**

**{**

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

**}**

**}**

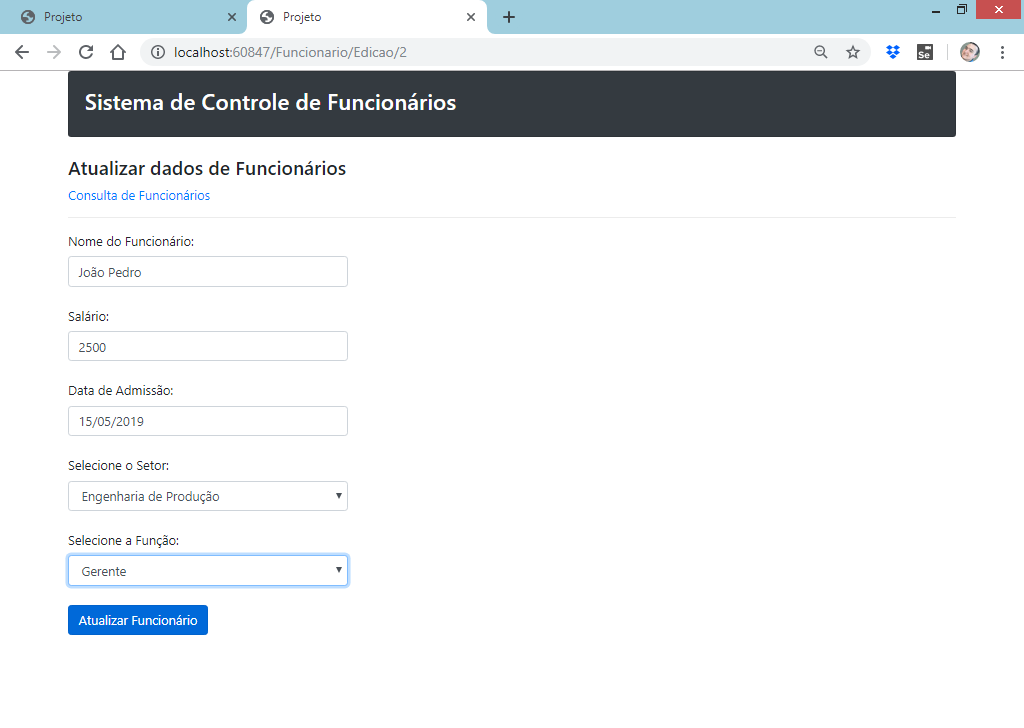
**return View(new FuncionarioEdicaoViewModel());**

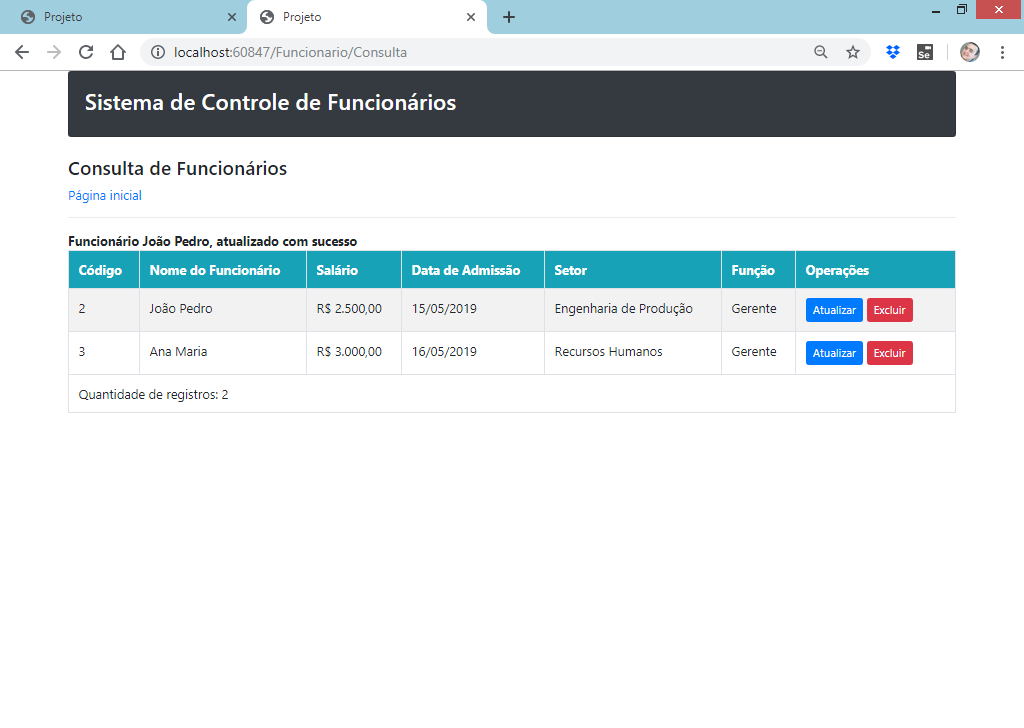
**}**

}

}

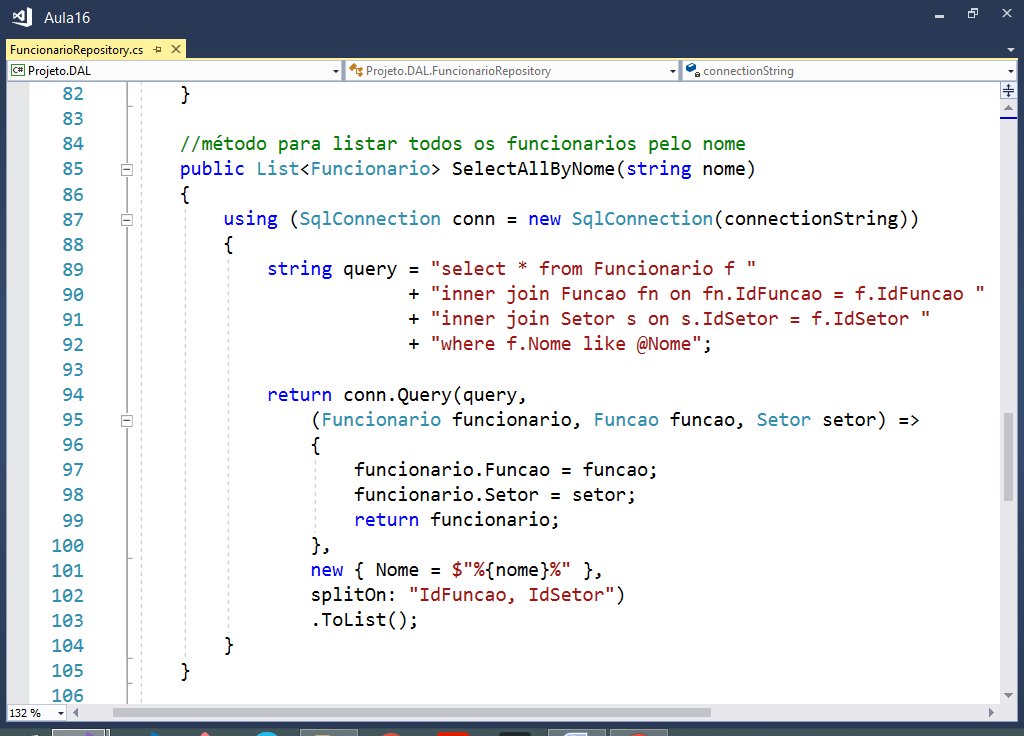
**Executando:**





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

**Filtrando a consulta de Funcionários:**



//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();

}

}

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

@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 class="row">**

**<div class="col-md-3">**

**@using (Html.BeginForm())**

**{**

**<label>Nome do Funcionário:</label>**

**<input type="text" class="form-control"**

**name="Nome" placeholder="Digite aqui"/>**

**<input type="submit" value="Pesquisar Funcionários"**

**class="btn btn-success btn-sm"/>**

**}**

**</div>**

**</div>**

<br />

<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:**

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;

}

//enviando a lista para a view

return View(lista);

}

**// POST: Funcionario/Consulta**

**[HttpPost] //recebe requisições do tipo POST**

**public ActionResult Consulta(string nome)**

**{**

**List<FuncionarioConsultaViewModel> lista**

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

**try**

**{**

**//varrer os funcionarios obtidos na base de dados**

**foreach (Funcionario funcionario**

**in business.ConsultarFuncionariosPorNome(nome))**

**{**

**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;**

**}**

**//enviando a lista para a view**

**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");

}

public ActionResult Edicao(int id)

{

//criando um objeto viewmodel

FuncionarioEdicaoViewModel model = new FuncionarioEdicaoViewModel();

try

{

//buscar o funcionario na base de dados pelo id..

Funcionario funcionario = business.ConsultarFuncionarioPorId(id);

model.IdFuncionario = funcionario.IdFuncionario;

model.Nome = funcionario.Nome;

model.Salario = funcionario.Salario;

model.DataAdmissao = funcionario.DataAdmissao;

model.IdSetor = funcionario.IdSetor;

model.IdFuncao = funcionario.IdFuncao;

}

catch(Exception e)

{

//exibir mensagem de erro..

TempData["Mensagem"] = e.Message;

}

//enviando a model para a página

return View(model);

}

// POST: Funcionario/Edicao

[HttpPost] //método recebe SUBMIT do formulário

public ActionResult Edicao(FuncionarioEdicaoViewModel model)

{

//verificar se os campos da model passaram nas validações

if (ModelState.IsValid)

{

try

{

Funcionario funcionario = new Funcionario();

funcionario.IdFuncionario = model.IdFuncionario;

funcionario.Nome = model.Nome;

funcionario.Salario = model.Salario;

funcionario.DataAdmissao = model.DataAdmissao;

funcionario.IdSetor = model.IdSetor;

funcionario.IdFuncao = model.IdFuncao;

business.AtualizarFuncionario(funcionario);

TempData["Mensagem"] = $"Funcionário

{funcionario.Nome}, atualizado com sucesso";

return RedirectToAction("Consulta"); //redirecionamento

}

catch (Exception e)

{

TempData["Mensagem"] = e.Message;

}

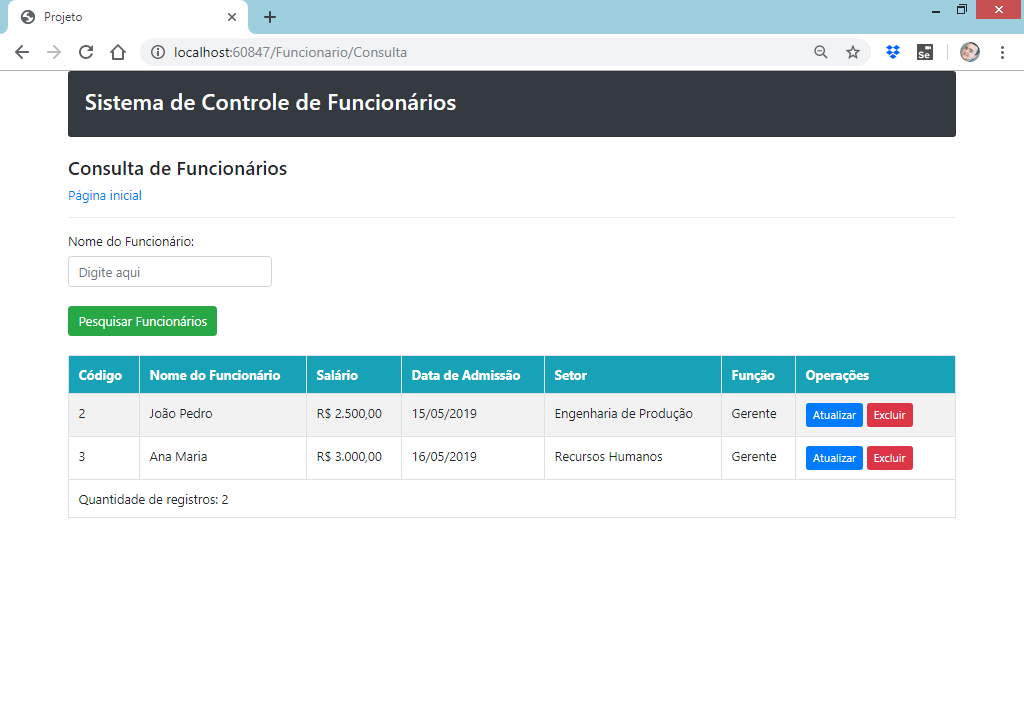
}

return View(new FuncionarioEdicaoViewModel());

}

}

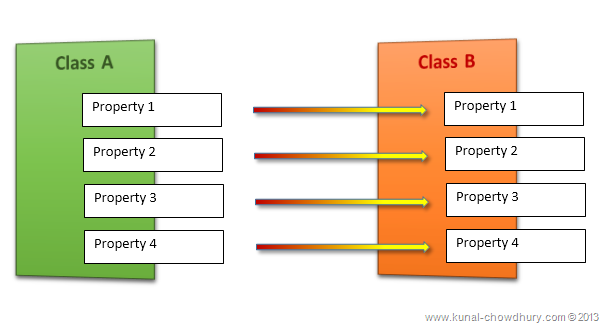
}



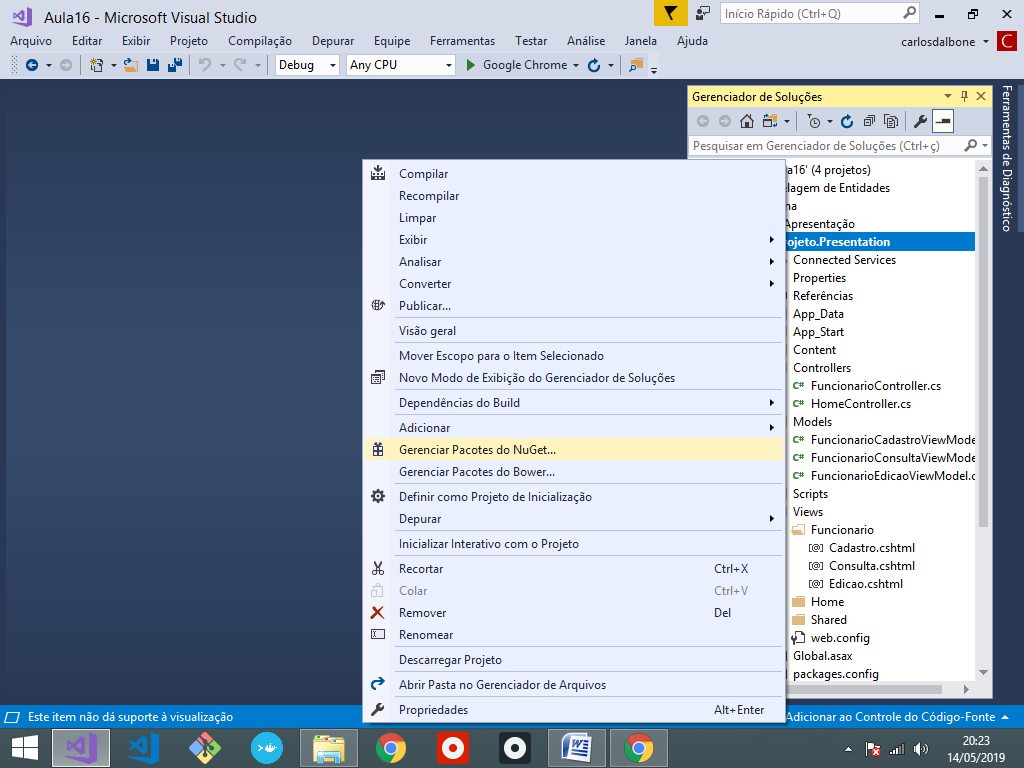
--------------------

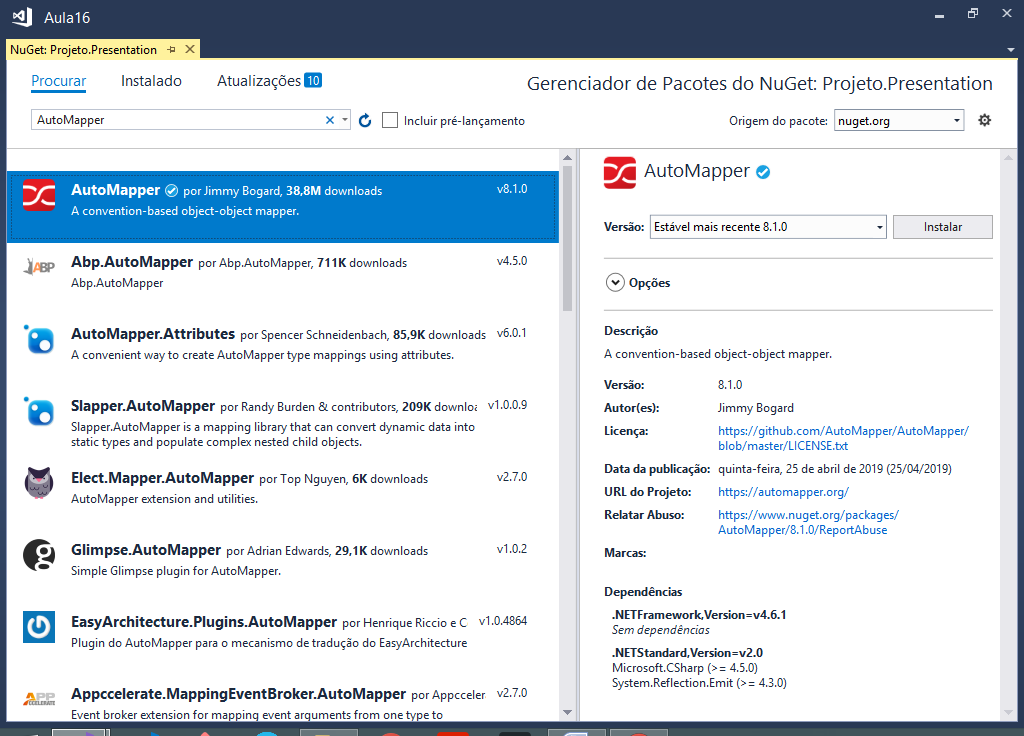
AutoMapper (<https://automapper.org/>)

Framework para realização de troca de dados entre objetos, iremos utiliza-lo para simplificar a transferencia dos dados entre as classes de entidade e viewmodels.



**Instalando o AutoMapper:**





**Criando 2 classes para mapear as trocas   
de dados que serão feitas pelo AutoMapper:**

\Mappings\ViewModelToEntityMap.cs

Mapeamento das trocas de dados entre ViewModel e Entidades

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using AutoMapper;

using Projeto.Entities;

using Projeto.Presentation.Models;

namespace Projeto.Presentation.Mappings

{

//classe de mapeamento do AutoMapper

public class ViewModelToEntityMap : Profile

{

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

public ViewModelToEntityMap()

{

CreateMap<FuncionarioCadastroViewModel, Funcionario>();

CreateMap<FuncionarioEdicaoViewModel, Funcionario>();

}

}

}

\Mappings\EntityToViewModelMap.cs

Mapeamento das trocas de dados entre Entidades e ViewModels

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using AutoMapper;

using Projeto.Entities;

using Projeto.Presentation.Models;

namespace Projeto.Presentation.Mappings

{

public class EntityToViewModelMap : Profile

{

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

public EntityToViewModelMap()

{

CreateMap<Funcionario, FuncionarioConsultaViewModel>();

CreateMap<Funcionario, FuncionarioEdicaoViewModel>();

}

}

}

Para que os mapeamentos feitos no AutoMapper possam funcionar é necessário configura-los na classe **Global.asax**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using System.Web.Routing;

using AutoMapper;

using Projeto.Presentation.Mappings;

namespace Projeto.Presentation

{

public class MvcApplication : System.Web.HttpApplication

{

//método executado quando o projeto

//é inicializado (similar ao Main())

protected void Application\_Start()

{

AreaRegistration.RegisterAllAreas();

RouteConfig.RegisterRoutes(RouteTable.Routes);

**//registrando as classes de mapeamento do automapper..**

**Mapper.Initialize(m =>**

**{**

**m.AddProfile<EntityToViewModelMap>();**

**m.AddProfile<ViewModelToEntityMap>();**

**});**

}

}

}

Refatoração

Melhoria / Otimização do código-fonte de uma aplicação.

<https://www.amazon.com.br/Refatora%C3%A7%C3%A3o-Aperfei%C3%A7oando-Projeto-C%C3%B3digo-Existente-ebook/dp/B019IZK89A?tag=goog0ef-20&smid=A18CNA8NWQSYHH&ascsubtag=go_1366271959_58245915327_265589414315_pla-644361093244_c_>



**Voltando na classe FuncionarioController:**

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

using AutoMapper;

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

{

var funcionario = Mapper.Map<Funcionario>(model);

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

{

lista = Mapper.Map<List<FuncionarioConsultaViewModel>>

(business.ConsultarFuncionarios());

}

catch(Exception e)

{

TempData["Mensagem"] = e.Message;

}

//enviando a lista para a view

return View(lista);

}

// POST: Funcionario/Consulta

[HttpPost] //recebe requisições do tipo POST

public ActionResult Consulta(string nome)

{

List<FuncionarioConsultaViewModel> lista

= new List<FuncionarioConsultaViewModel>();

try

{

lista = Mapper.Map<List<FuncionarioConsultaViewModel>>

(business.ConsultarFuncionariosPorNome(nome));

}

catch (Exception e)

{

TempData["Mensagem"] = e.Message;

}

//enviando a lista para a view

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");

}

public ActionResult Edicao(int id)

{

//criando um objeto viewmodel

FuncionarioEdicaoViewModel model = new FuncionarioEdicaoViewModel();

try

{

model = Mapper.Map<FuncionarioEdicaoViewModel>

(business.ConsultarFuncionarioPorId(id));

}

catch(Exception e)

{

TempData["Mensagem"] = e.Message;

}

return View(model);

}

[HttpPost] //método recebe SUBMIT do formulário

public ActionResult Edicao(FuncionarioEdicaoViewModel model)

{

if (ModelState.IsValid)

{

try

{

var funcionario = Mapper.Map<Funcionario>(model);

business.AtualizarFuncionario(funcionario);

TempData["Mensagem"] = $"Funcionário

{funcionario.Nome}, atualizado com sucesso";

return RedirectToAction("Consulta"); //redirecionamento

}

catch (Exception e)

{

TempData["Mensagem"] = e.Message;

}

}

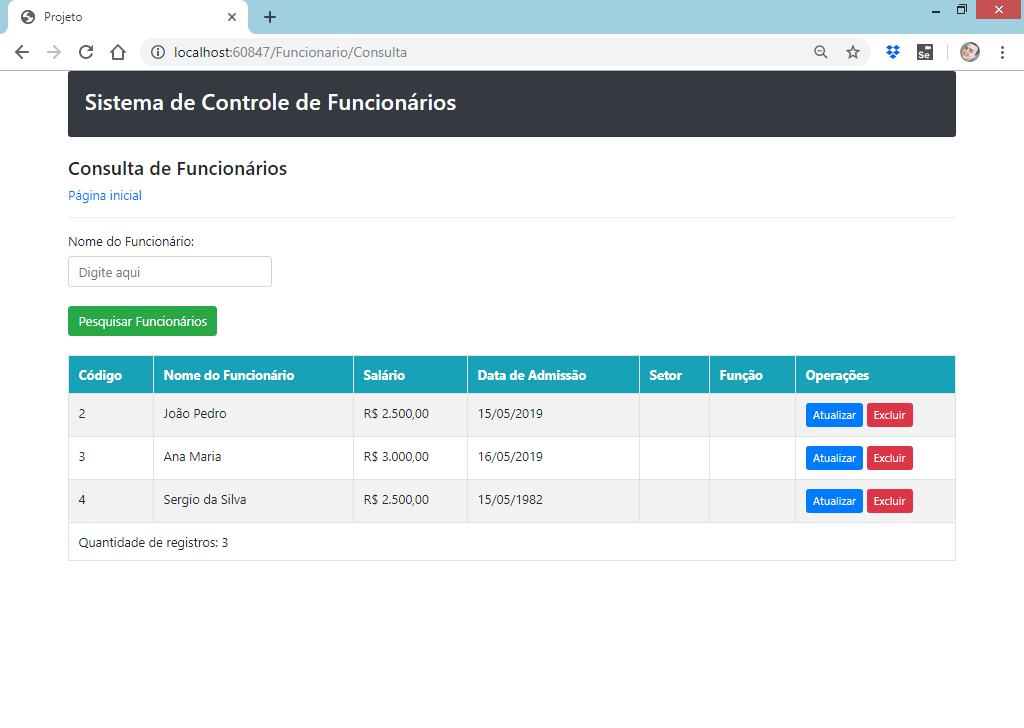
return View(new FuncionarioEdicaoViewModel());

}

}

}

**Mapeando os campos com nomes diferentes   
entre as viewmodels e entidades:**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using AutoMapper;

using Projeto.Entities;

using Projeto.Presentation.Models;

namespace Projeto.Presentation.Mappings

{

public class EntityToViewModelMap : Profile

{

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

public EntityToViewModelMap()

{

**CreateMap<Funcionario, FuncionarioConsultaViewModel>()**

**.AfterMap((from, to) => to.NomeSetor = from.Setor.Nome)**

**.AfterMap((from, to) => to.NomeFuncao = from.Funcao.Nome);**

CreateMap<Funcionario, FuncionarioEdicaoViewModel>();

}

}

}

**Executando:**

