**Voltando no repositório:**

/Repository/ClienteRepository.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient; //acesso ao sqlserver

using System.Configuration; //connectionstring

using Projeto.DAL.Entities; //entidades

namespace Projeto.DAL.Repositories

{

public class ClienteRepository

{

//atributos

private SqlConnection connection;

private SqlCommand command;

private SqlDataReader dataReader;

private string connectionString;

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

public ClienteRepository()

{

connectionString = ConfigurationManager

.ConnectionStrings["projeto"].ConnectionString;

}

//método para inserir um cliente na base de dados

public void Insert(Cliente cliente)

{

using (connection = new SqlConnection(connectionString))

{

connection.Open(); //conectado..

string query = "insert into Cliente(Nome, Email, DataCadastro) "

+ "values(@Nome, @Email, @DataCadastro)";

command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@Nome", cliente.Nome);

command.Parameters.AddWithValue("@Email", cliente.Email);

command.Parameters.AddWithValue("@DataCadastro",

cliente.DataCadastro);

command.ExecuteNonQuery();

}

}

//método para verificar se um email já está cadastrado na tabela

public bool HasEmail(string email)

{

using (connection = new SqlConnection(connectionString))

{

connection.Open(); //conectado..

string query = "select Email from Cliente where Email = @Email";

command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@Email", email);

dataReader = command.ExecuteReader();

return dataReader.HasRows;

}

}

//método para atualizar os dados de um cliente

public void Update(Cliente cliente)

{

using (connection = new SqlConnection(connectionString))

{

connection.Open(); //conectado

string query = "update Cliente set Nome = @Nome, Email = @Email "

+ "where IdCliente = @IdCliente";

command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@IdCliente", cliente.IdCliente);

command.Parameters.AddWithValue("@Nome", cliente.Nome);

command.Parameters.AddWithValue("@Email", cliente.Email);

command.ExecuteNonQuery();

}

}

//método para excluir um registro de cliente

public void Delete(int idCliente)

{

using (connection = new SqlConnection(connectionString))

{

connection.Open();

string query = "delete from Cliente

where IdCliente = @IdCliente";

command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@IdCliente", idCliente);

command.ExecuteNonQuery();

}

}

//método para consultar todos os clientes cadastrados

public List<Cliente> FindAll()

{

using (connection = new SqlConnection(connectionString))

{

connection.Open();

string query = "select \* from Cliente";

command = new SqlCommand(query, connection);

dataReader = command.ExecuteReader();

//declarando uma lista de clientes..

List<Cliente> lista = new List<Cliente>();

//percorrendo os registros obtidos na consulta..

while (dataReader.Read())

{

Cliente cliente = new Cliente();

cliente.IdCliente = Convert.ToInt32(dataReader["IdCliente"]);

cliente.Nome = Convert.ToString(dataReader["Nome"]);

cliente.Email = Convert.ToString(dataReader["Email"]);

cliente.DataCadastro = Convert.ToDateTime

(dataReader["DataCadastro"]);

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

}

//retornando a lista..

return lista;

}

}

//método para retornar 1 cliente pelo id

public Cliente FindById(int idCliente)

{

using (connection = new SqlConnection(connectionString))

{

connection.Open(); //abrindo conexão..

string query = "select \* from Cliente

where IdCliente = @IdCliente";

command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@IdCliente", idCliente);

dataReader = command.ExecuteReader();

//verificando se algum registro foi encontrado

if(dataReader.Read())

{

Cliente cliente = new Cliente();

cliente.IdCliente = Convert.ToInt32(dataReader["IdCliente"]);

cliente.Nome = Convert.ToString(dataReader["Nome"]);

cliente.Email = Convert.ToString(dataReader["Email"]);

cliente.DataCadastro = Convert.ToDateTime

(dataReader["DataCadastro"]);

return cliente;

}

else

{

return null; //vazio (sem espaço de memória)

}

}

}

}

}

**Camada de Regras de Negócio:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto.DAL.Entities; //importando..

using Projeto.DAL.Repositories; //importando..

namespace Projeto.BLL.Business

{

public class ClienteBusiness

{

//método para realizar o cadastro de um cliente

//enviado pelo projeto 'Presentation'

public void CadastrarCliente(Cliente cliente)

{

//instanciando o repositório

ClienteRepository repository = new ClienteRepository();

//verificar se o email não está cadastrado

if( ! repository.HasEmail(cliente.Email))

{

repository.Insert(cliente); //gravando

}

else

{

throw new Exception($"O email '{cliente.Email}'

já está cadastrado no sistema.");

}

}

//método para atualizar os dados do cliente

public void AtualizarCliente(Cliente cliente)

{

ClienteRepository repository = new ClienteRepository();

repository.Update(cliente);

}

//método para excluir os dados do cliente

public void ExcluirCliente(int idCliente)

{

ClienteRepository repository = new ClienteRepository();

repository.Delete(idCliente);

}

//método para retornar todos os clientes

public List<Cliente> ConsultarTodos()

{

ClienteRepository repository = new ClienteRepository();

return repository.FindAll();

}

//método para retornar 1 cliente pelo id

public Cliente ConsultarPorId(int idCliente)

{

ClienteRepository repository = new ClienteRepository();

Cliente cliente = repository.FindById(idCliente);

//verificar se o cliente foi encontrado

if(cliente != null)

{

return cliente; //retornando o cliente

}

else

{

throw new Exception("Cliente não encontrado.");

}

}

}

}

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

MVC - Model, View e Controller

Padrão para desenvolvimento de aplicações web separadas por 3 papéis:

View

Representa a página HTML, composta também de código CSS (Folha de Estilo) e também JavaScript. Em Asp.Net MVC estas páginas possuem a extensão **.cshtml** pois também permitem a escrita de código C#.

Controller

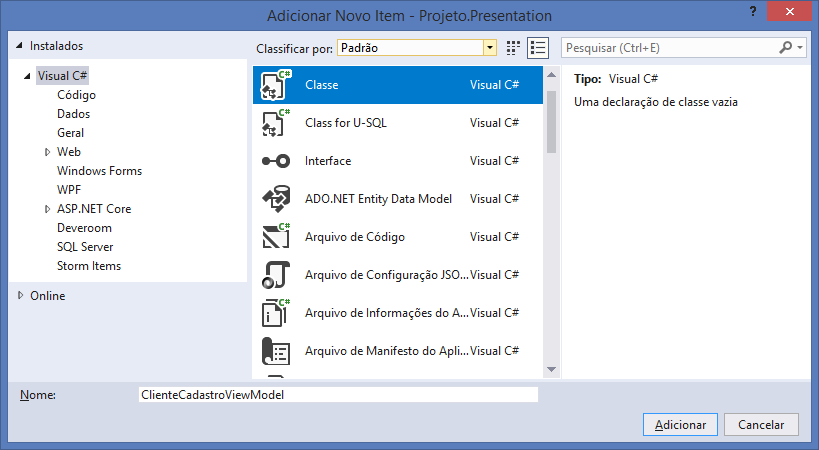
Representa a classe que gerencia uma ou mais páginas (Views) do projeto Asp.Net MVC. Também tem a responsabilidade de receber dados enviados por formulários contidos nas páginas.

Model

Representa a classe que irá receber ou retornar os dados dos formulários ou consultas exibidas nas páginas. Será de responsabilidade das Models validar também estes dados de entrada / saida.

Tarefa: Criar um formulário   
para cadastro de cliente:

**Passo 1)** Criar uma classe na pasta "**Models**" contendo   
os campos que irão servir para o formulário (**Nome**, **Email**)



using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations; //validações

namespace Projeto.Presentation.Models

{

public class ClienteCadastroViewModel

{

[MinLength(6, ErrorMessage = "Informe no mínimo {1} caracteres.")]

[MaxLength(100, ErrorMessage = "Informe no máximo {1} caracteres.")]

[Required(ErrorMessage = "Por favor, informe o nome do cliente.")]

public string Nome { get; set; }

[EmailAddress(ErrorMessage = "Informe um endereço de email válido.")]

[Required(ErrorMessage = "Por favor, informe o email do cliente.")]

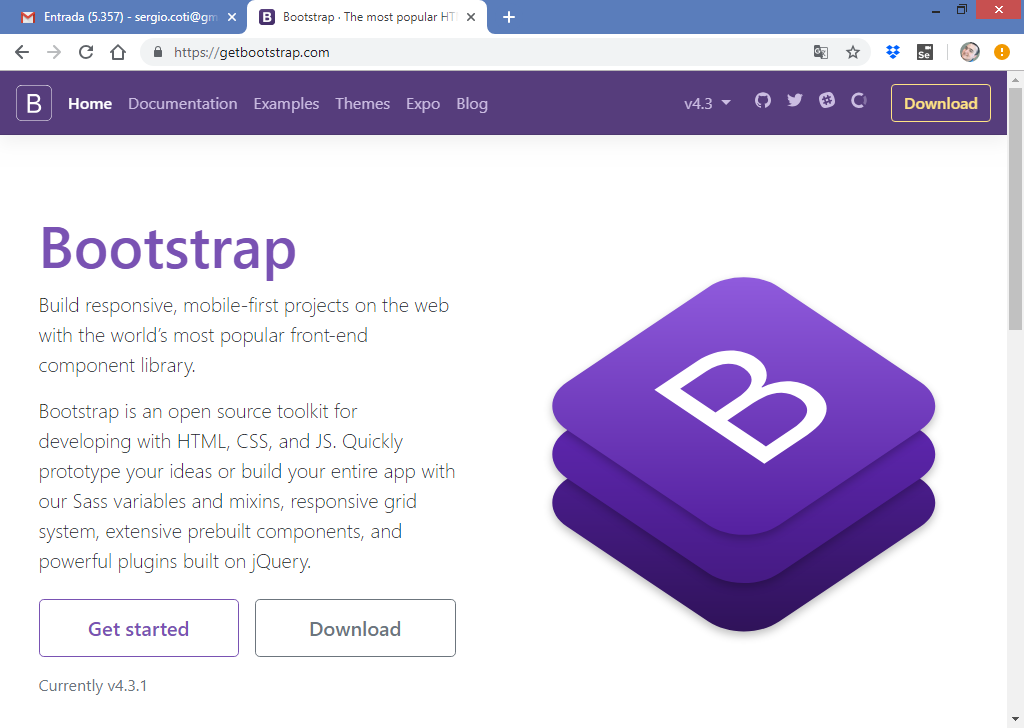
public string Email { get; set; }

}

}

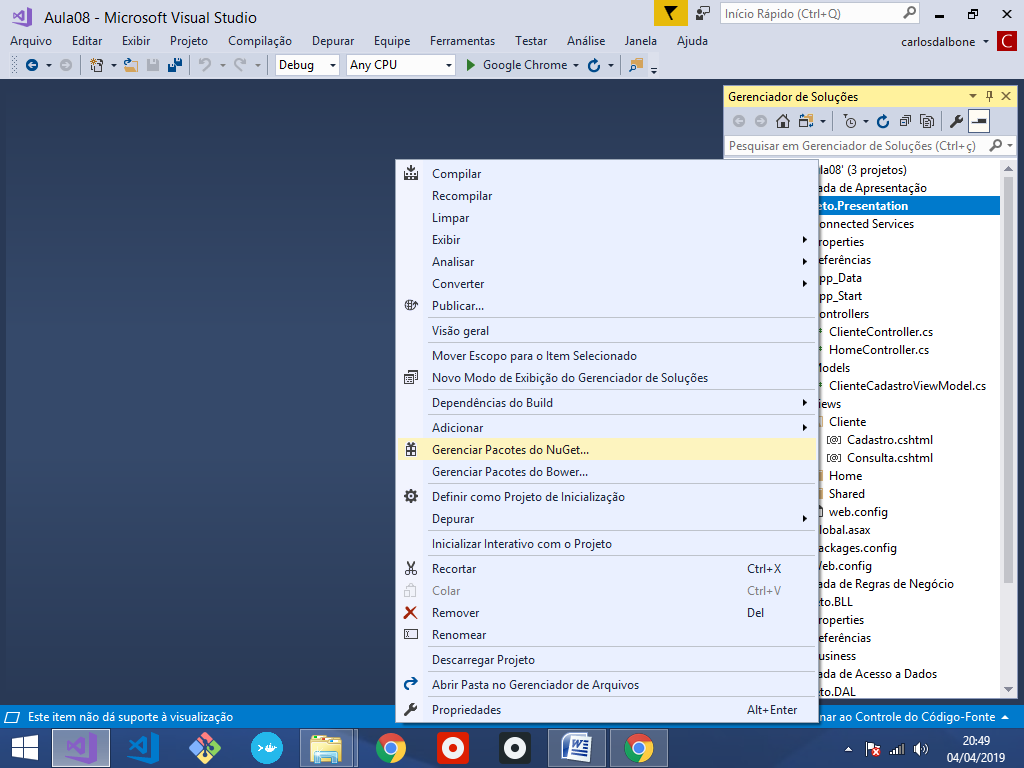
Bootstrap (<https://getbootstrap.com/>)

Conjunto de bibliotecas formadas por arquivos CSS e JavaScript   
para desenvolvimento de interface web de páginas (**FrontEnd**)

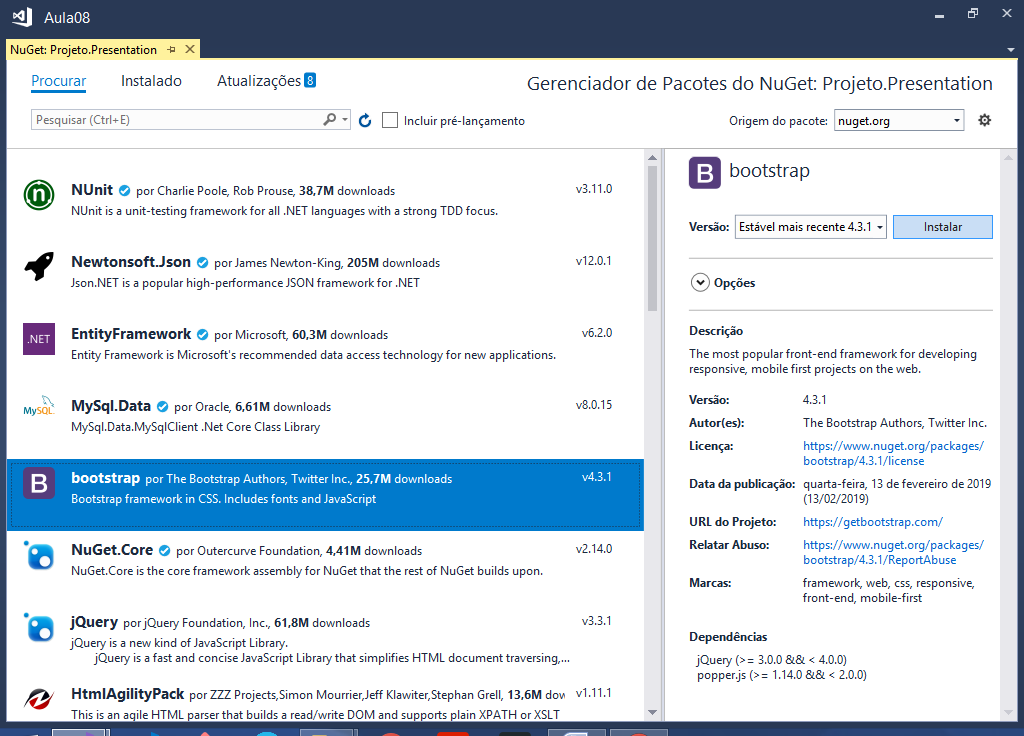


**Instalando o bootstrap no projeto Asp.Net MVC**

Gerenciador de pacotes do NuGet

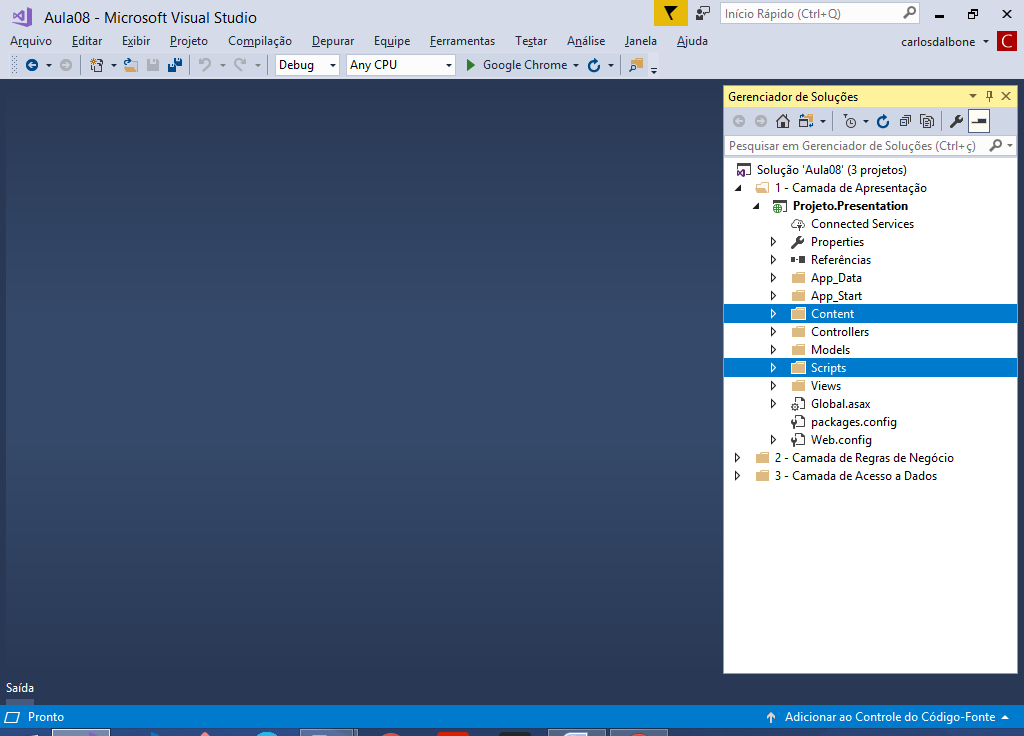


**Selecione bootstrap:**



**Pastas adicionadas no projeto:**

* **Content** Composta de arquivos .CSS (Folha de estilo)
* **Scripts** Composta de arquivos .JS (JavaScript)



Voltando na MasterPage

/Views/Shared/Layout.cshtml

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>COTI Informática</title>

<!-- folhas de estilo CSS -->

**<link href="~/Content/bootstrap.min.css" rel="stylesheet" />**

</head>

<body>

<div>

<h3>Sistema de Controle de Clientes</h3>

<hr/>

<!-- INICIO DO CONTEUDO PRINCIPAL -->

@RenderBody()

<!-- FIM DO CONTEUDO PRINCIPAL -->

</div>

<!-- Arquivos javascript -->

**<script src="~/Scripts/jquery-3.0.0.min.js"></script>**

**<script src="~/Scripts/bootstrap.min.js"></script>**

</body>

</html>

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

**Utilizando classes do bootstrap:**

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>COTI Informática</title>

<!-- folhas de estilo CSS -->

<link href="~/Content/bootstrap.min.css" rel="stylesheet" />

</head>

<body>

**<div class="container">**

**<div class="card card-body bg-dark">**

**<h3 class="text-white">Sistema de Controle de Clientes</h3>**

**</div>**

<br/>

<!-- INICIO DO CONTEUDO PRINCIPAL -->

@RenderBody()

<!-- FIM DO CONTEUDO PRINCIPAL -->

</div>

<!-- Arquivos javascript -->

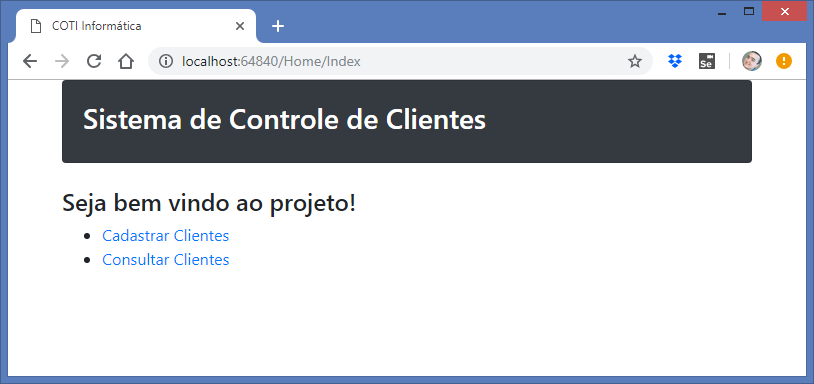
<script src="~/Scripts/jquery-3.0.0.min.js"></script>

<script src="~/Scripts/bootstrap.min.js"></script>

</body>

</html>

Executando:



**Criando o formulário para cadastro de clientes:**

@model Projeto.Presentation.Models.ClienteCadastroViewModel

@{

ViewBag.Title = "Cadastro";

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

}

<h4>Cadastro de Clientes</h4>

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

<br/>

<br/>

@using (Html.BeginForm())

{

<div class="text-danger">

@Html.ValidationSummary()

</div>

<label>Nome do Cliente:</label>

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

new { @class = "form-control col-md-4" })

<br/>

<label>Email do Cliente:</label>

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

new { @class = "form-control col-md-4" })

<br/>

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

class="btn btn-success"/>

}

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

Criando o método na classe de controle para receber os dados enviados pelo formulário:

/ClienteController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Projeto.DAL.Entities; //importando..

using Projeto.BLL.Business; //importando..

using Projeto.Presentation.Models; //importando..

namespace Projeto.Presentation.Controllers

{

public class ClienteController : Controller

{

// GET: Cliente/Cadastro

public ActionResult Cadastro()

{

return View();

}

**// POST: Cliente/Cadastro**

**[HttpPost] //requisições de formulário**

**public ActionResult Cadastro(ClienteCadastroViewModel model)**

**{**

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

**//nas regras de validação**

**if(ModelState.IsValid)**

**{**

**try**

**{**

**Cliente cliente = new Cliente();**

**cliente.Nome = model.Nome;**

**cliente.Email = model.Email;**

**cliente.DataCadastro = DateTime.Now;**

**ClienteBusiness business = new ClienteBusiness();**

**business.CadastrarCliente(cliente);**

**ViewData["Mensagem"] = "Cliente cadastrado com sucesso.";**

**ModelState.Clear(); //limpar os campos do formulário**

**}**

**catch(Exception e)**

**{**

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

**}**

**}**

**return View();**

**}**

// GET: Cliente/Consulta

public ActionResult Consulta()

{

return View();

}

}

}

**Exibindo na página:**

@model Projeto.Presentation.Models.ClienteCadastroViewModel

@{

ViewBag.Title = "Cadastro";

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

}

<h4>Cadastro de Clientes</h4>

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

<br/>

<br/>

@using (Html.BeginForm())

{

<div class="text-danger">

@Html.ValidationSummary()

</div>

<label>Nome do Cliente:</label>

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

new { @class = "form-control col-md-4" })

<br/>

<label>Email do Cliente:</label>

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

new { @class = "form-control col-md-4" })

<br/>

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

class="btn btn-success"/>

<br/>

<br/>

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

}

**Executando:**

