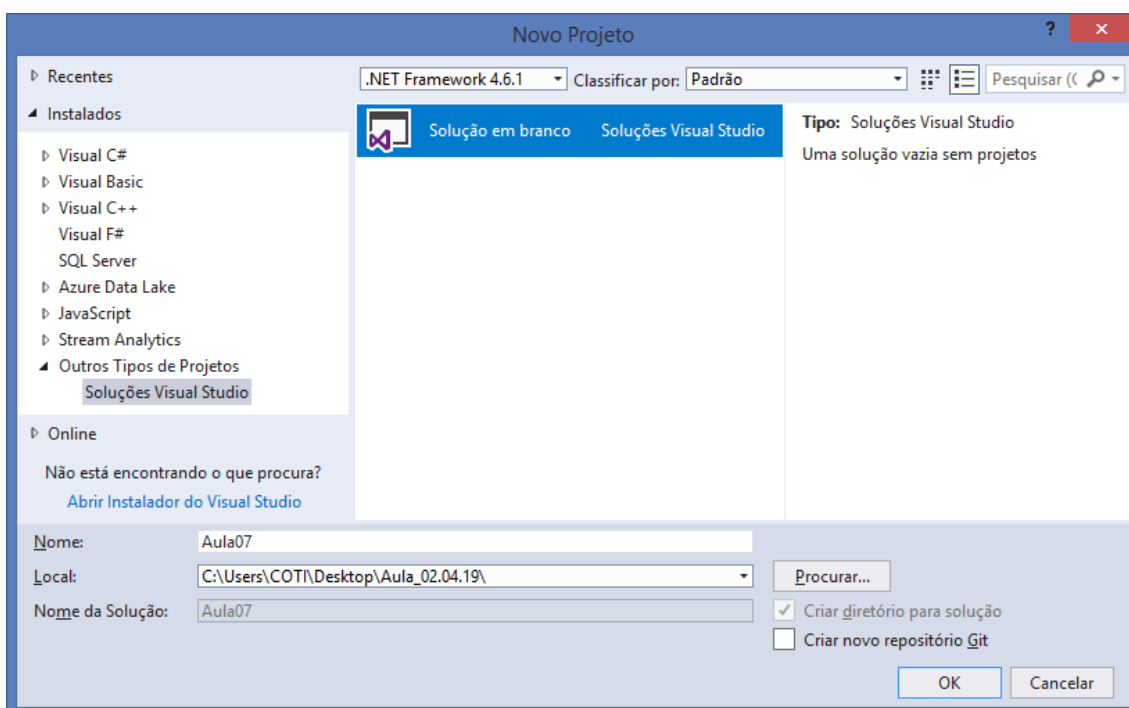
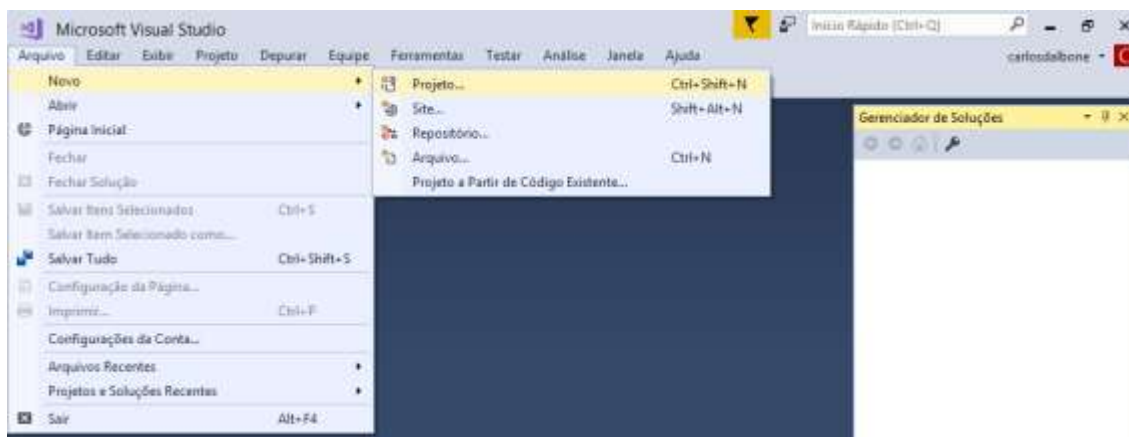


## Criando uma nova solution em branco:



## Arquitetura baseada em camadas

Todo sistema criado em .NET deverá ser composto de, no mínimo, 3 camadas (partes)

### Presentation Layer

Camada de Apresentação (Projeto Asp.Net).

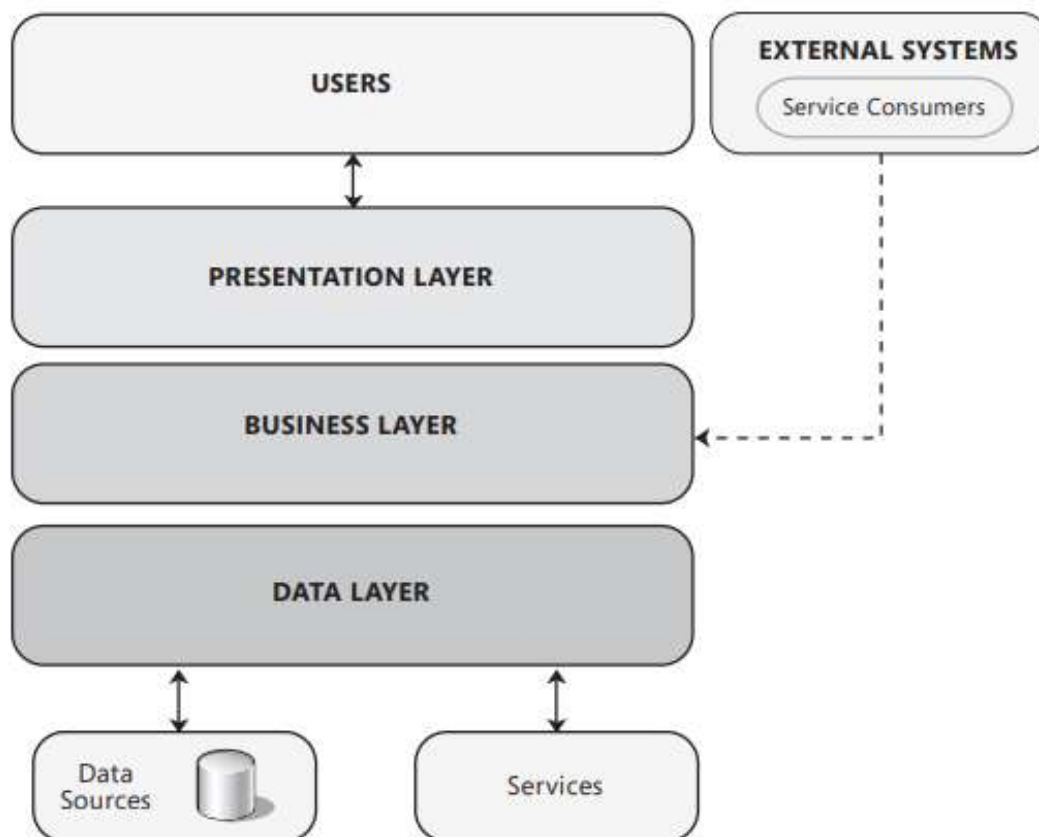
### Business Logic Layer (BLL)

Camada utilizada para implementar as regras de negócio do sistema.

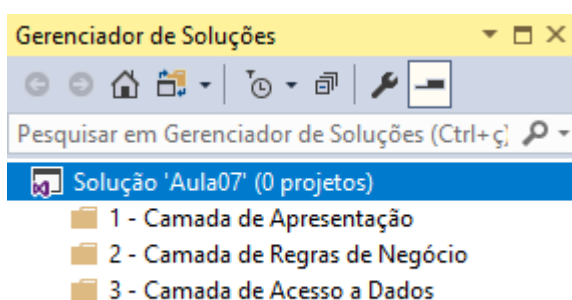
### Data Access Layer (DAL)

Camada utilizada para acesso a base de dados.

<https://www.intertech.com/Downloads/eBook/ApplicationArchitectureGuide.pdf>



## Organização da Solution



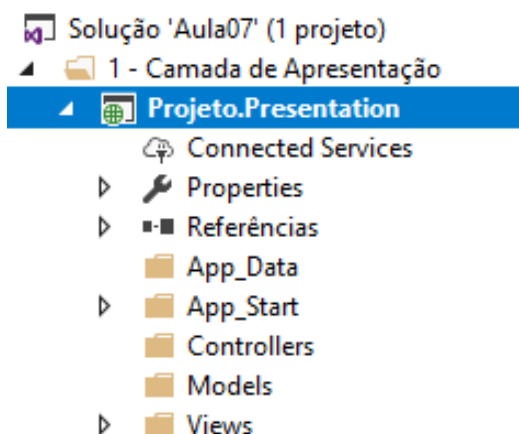
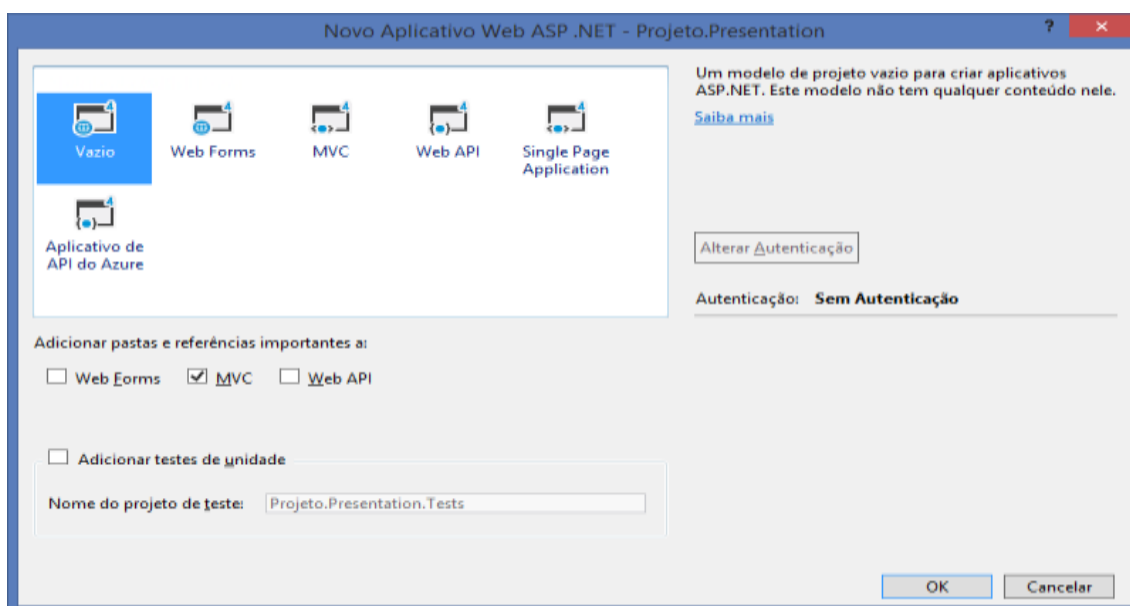
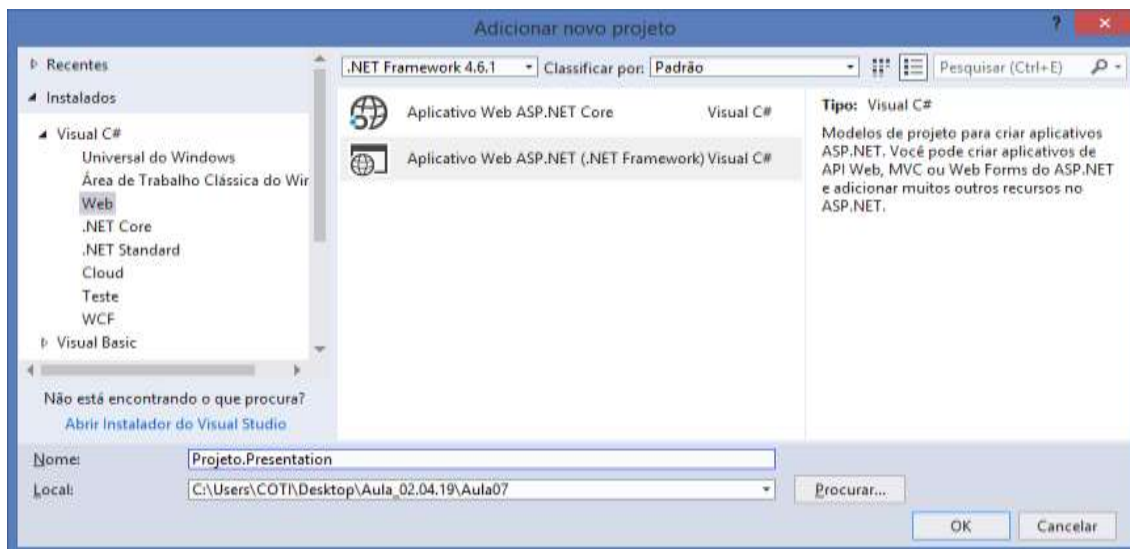
## Asp.Net

Conjunto de tecnologias da plataforma .NET voltado para desenvolvimento de aplicações web. O Asp.Net pode ser dividido em 3 principais tecnologias:

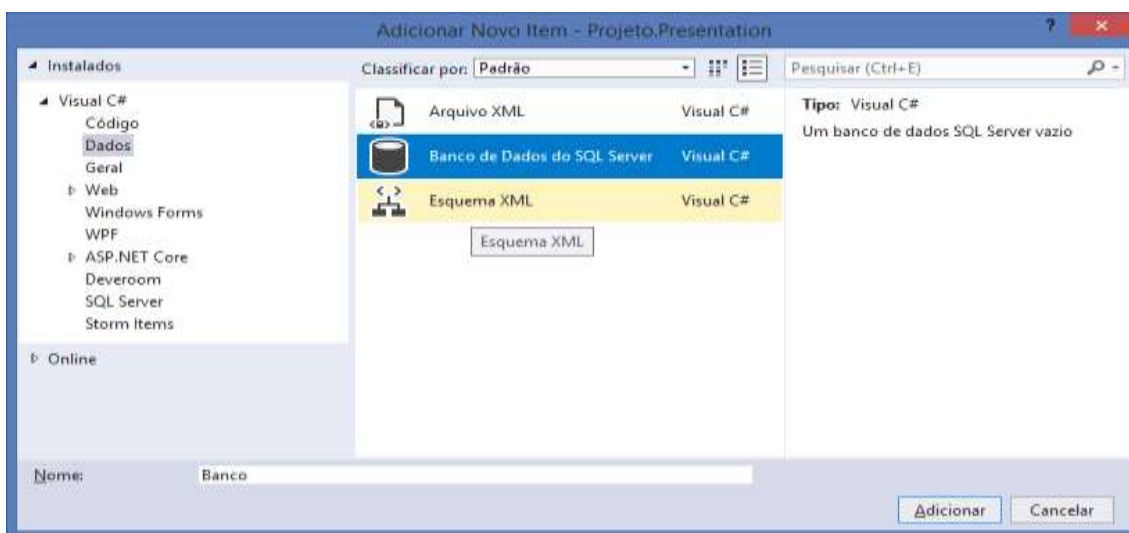
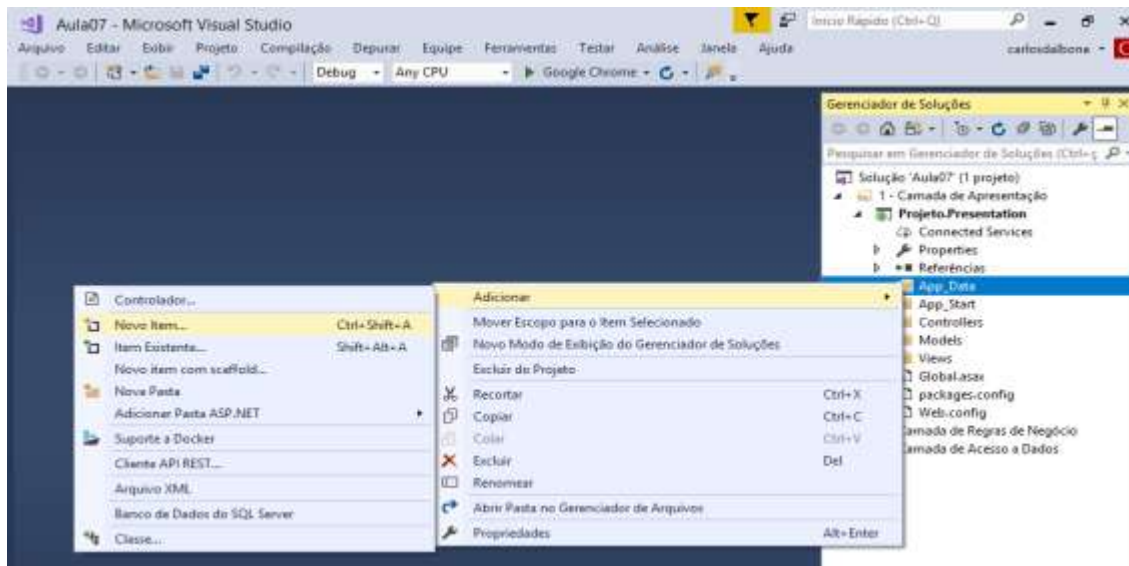
- Asp.Net WebForms
- Asp.Net MVC
- Asp.Net WebApi

## 1 - Camada de Apresentação

Criando o projeto Asp.Net



## Criando um banco de dados local do SqlServer: App\_Data/Banco.mdf

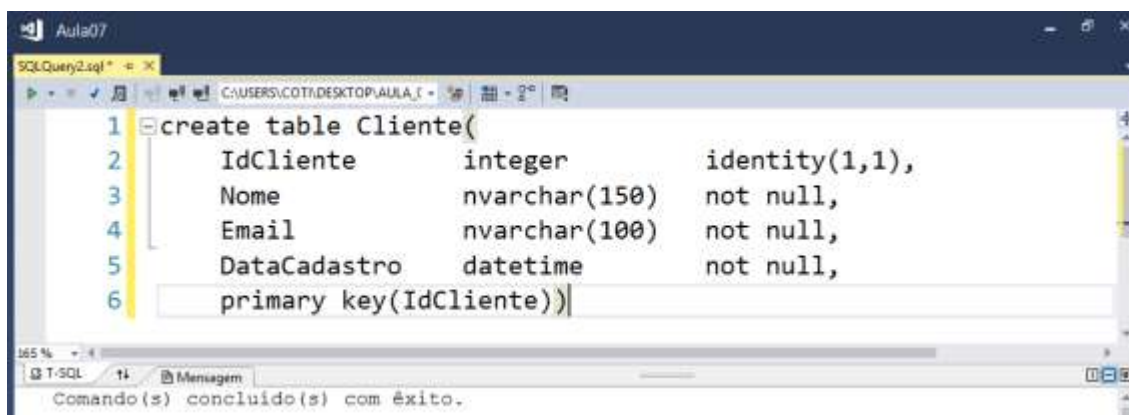


## Criando uma tabela de **Cliente** no banco de dados

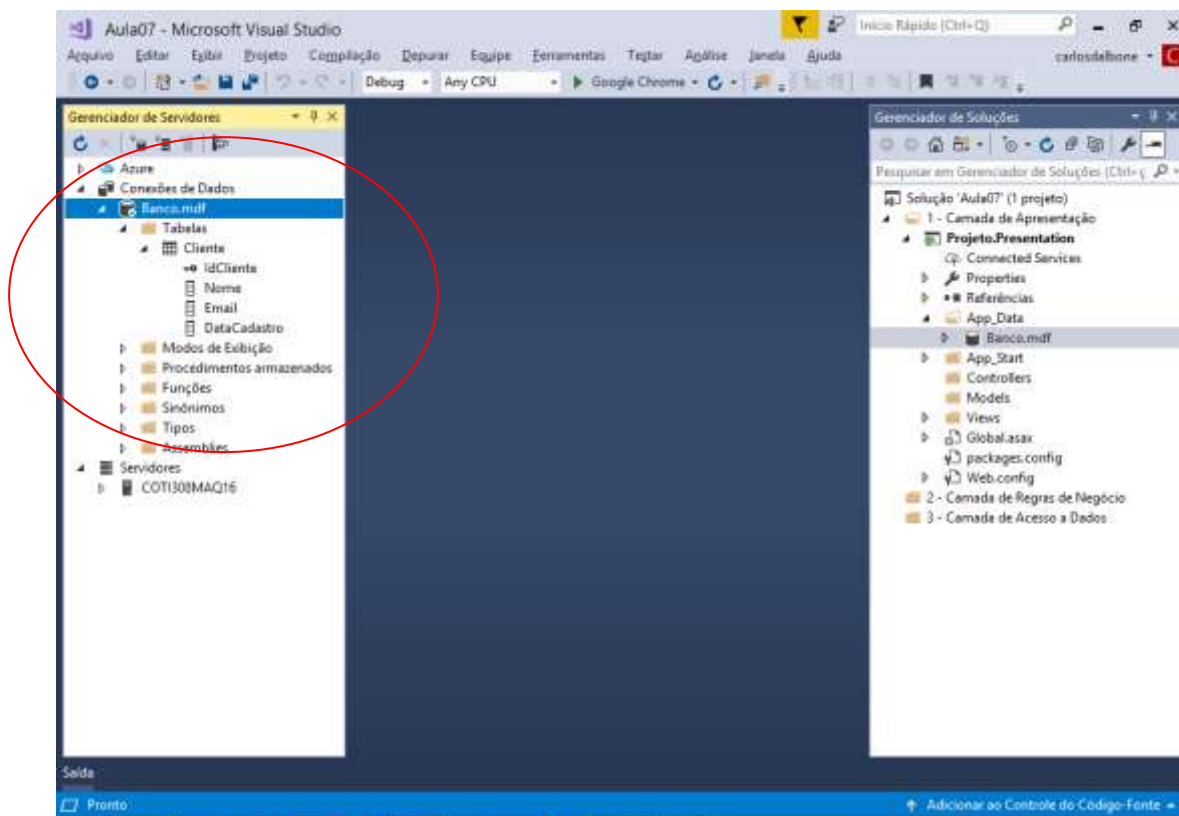


## Script para criação da tabela:

```
create table Cliente(
    IdCliente      integer      identity(1,1),
    Nome           nvarchar(150) not null,
    Email          nvarchar(100) not null,
    DataCadastro   datetime     not null,
    primary key(IdCliente))
```

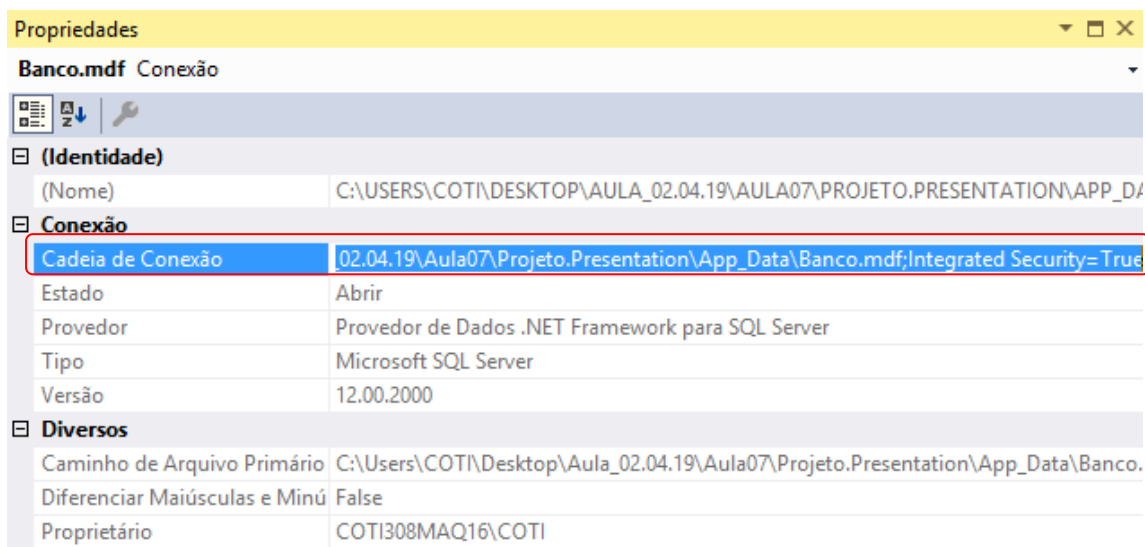
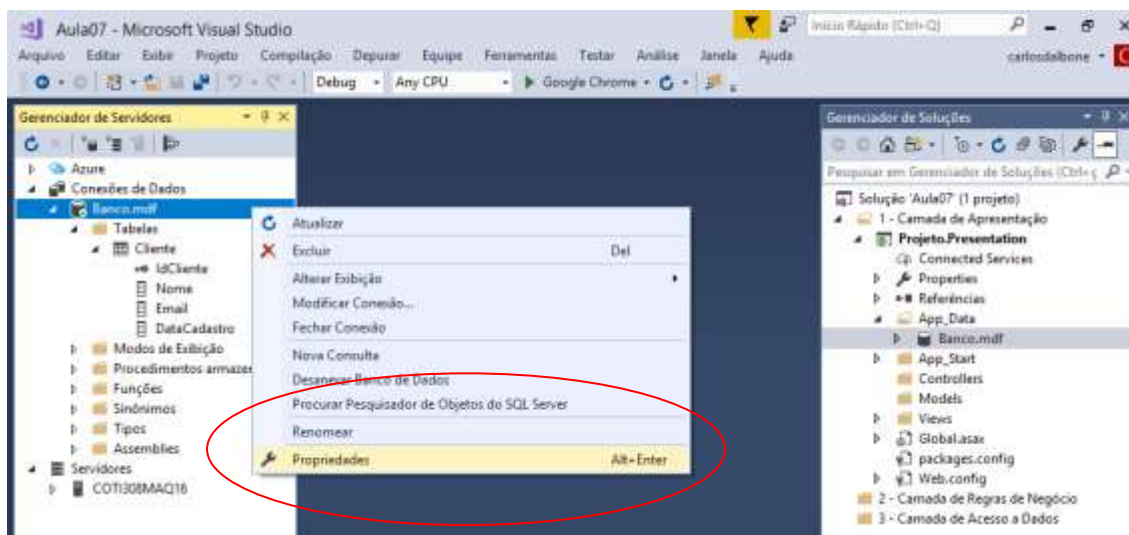


## Tabela criada:





## Mapeamento daConnectionString do banco de dados



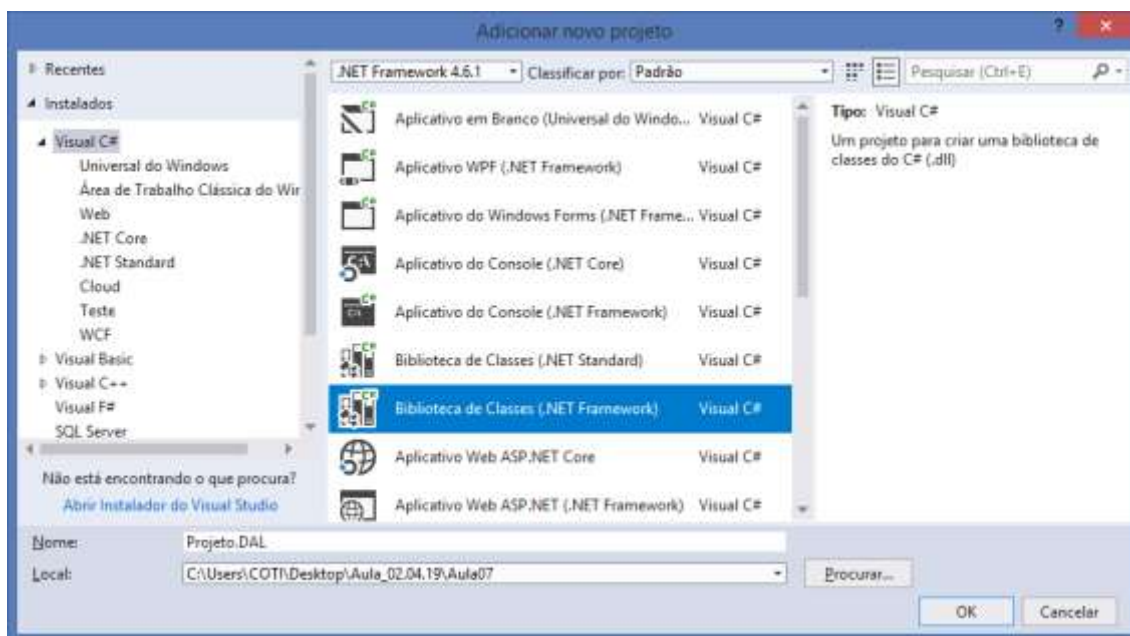
## Web.config.xml

Mapeando a string de conexão com o banco de dados:

```
<!-- Mapeamento da connectionstring -->
<connectionStrings>
  <add
    name="projeto"
    connectionString="Data Source=(LocalDB)\MSSQLLocalDB;
      AttachDbFilename=C:\Users\COTI\Desktop\
      Aula_02.04.19\Aula07\Projeto.Presentation\
      App_Data\Banco.mdf;Integrated Security=True"
  />
</connectionStrings>
```

## 3 - Camada de Acesso a Dados

Criando um projeto voltado somente para implementação de classes com o objetivo de realizar acesso e persistência (CRUD) em uma base de dados.



### Criando uma classe Cliente:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto.DAL.Entities
{
    public class Cliente
    {
        //propriedades -> prop + 2x[tab]
        public int IdCliente { get; set; }
        public string Nome { get; set; }
        public string Email { get; set; }
        public DateTime DataCadastro { get; set; }

        //construtor -> ctor + 2x[tab]
        public Cliente()
        {
            //vazio
        }

        //construtor com entrada de argumentos (sobrecarga de construtores)
        public Cliente(int idCliente, string nome, string email,
            DateTime dataCadastro)
        {
        }
    }
}
```

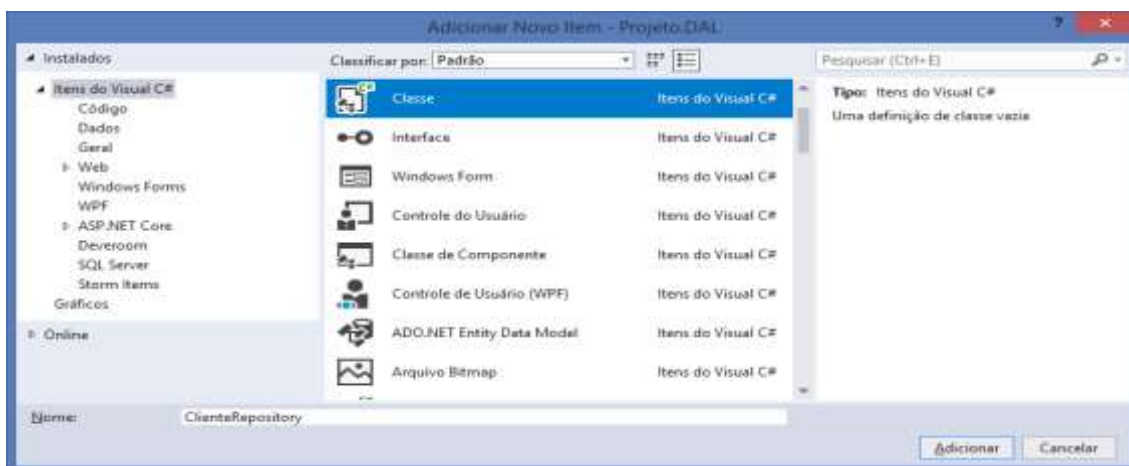
```

        IdCliente = idCliente;
        Nome = nome;
        Email = email;
        DataCadastro = dataCadastro;
    }

    //Sobrescrita do método ToString()
    public override string ToString()
    {
        return $"Id: {IdCliente}, Nome: {Nome}, Email: {Email},
            Data de Cadastro: {DataCadastro}";
    }
}
}

```

Criando a classe para armazenar os registros de Cliente em banco de dados



```

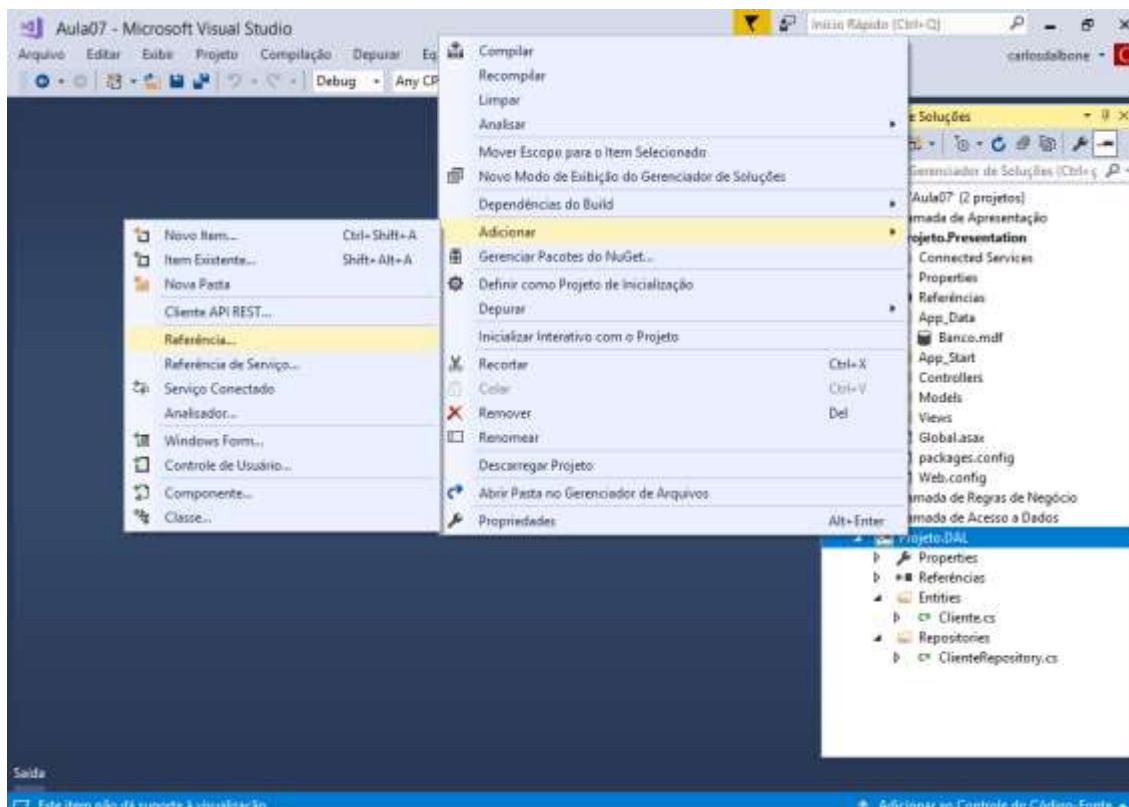
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient; //acesso ao sqlserver
using Projeto.DAL.Entities; //entidades

namespace Projeto.DAL.Repositories
{
    public class ClienteRepository
    {
        //atributos
        private SqlConnection connection;
        private SqlCommand command;
        private SqlDataReader dataReader;
        private string connectionString;
    }
}

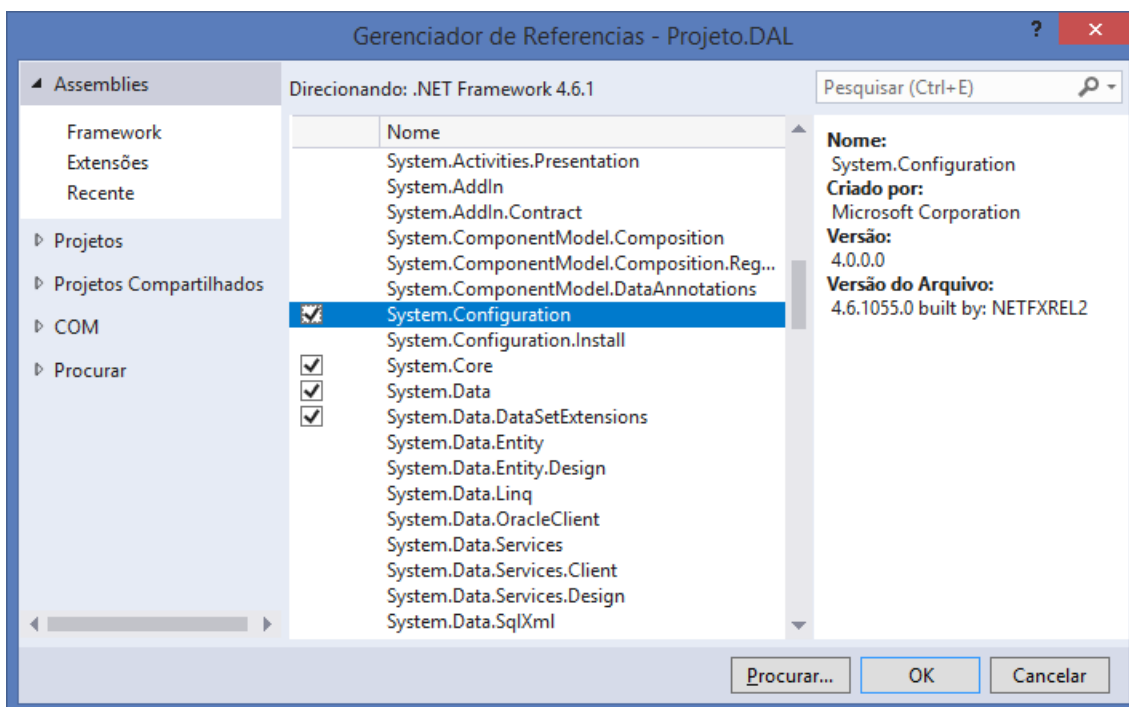
```



Para que possamos obter a string de conexão do arquivo **Web.config.xml** é necessário adicionar no projeto uma referência para **System.Configuration**



Selecione: **System.Configuration**



```
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..

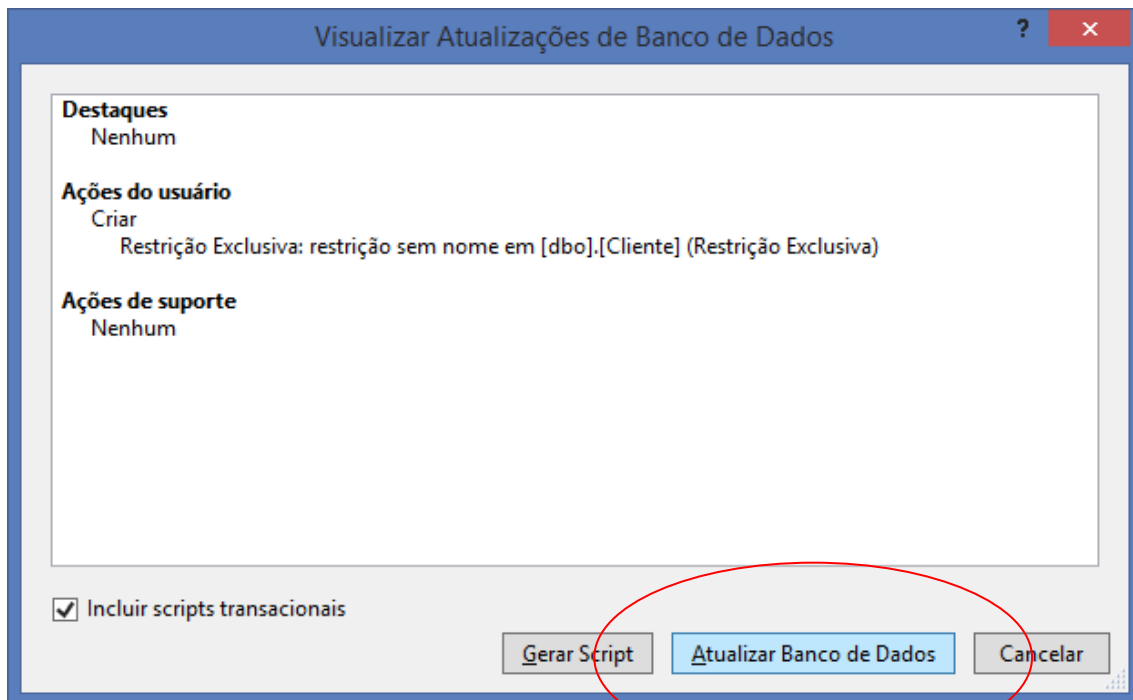
                string query = "insert into Cliente(Nome, Email, DataCadastro) "
                    + "values(@Nome, @Email, @DataCadastro)";

                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();
            }
        }
    }
}
```

## Voltando na base de dados:

```
CREATE TABLE [dbo].[Cliente] (
    [IdCliente] INT IDENTITY (1, 1) NOT NULL,
    [Nome] NVARCHAR (150) NOT NULL,
    [Email] NVARCHAR (100) NOT NULL UNIQUE,
    [DataCadastro] DATETIME NOT NULL,
    PRIMARY KEY CLUSTERED ([IdCliente] ASC)
);
```



## Método para verificar se um email já está cadastrado na tabela de cliente:

```
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();
    }
}

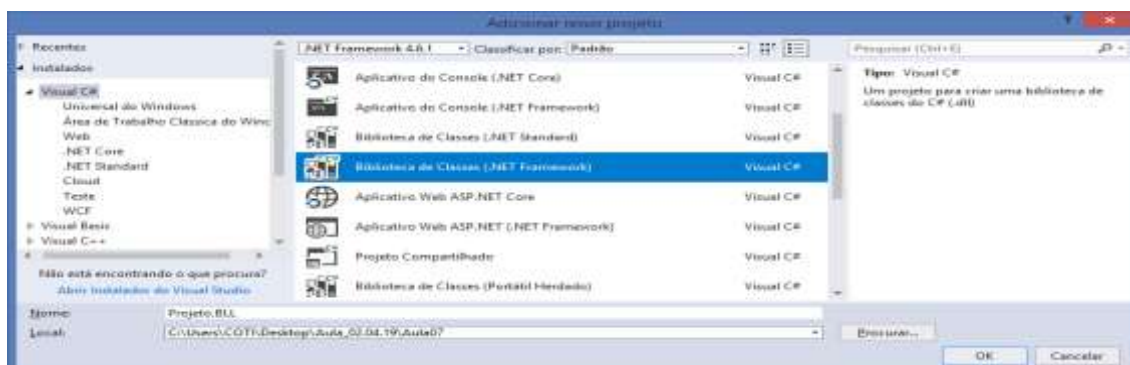
//método para verificar se um email já está cadastrado na tabela
public bool HasEmail(string email)
{
    using (connection = new SqlConnection(connectionString))
    {
        connection.Open(); //conectado..

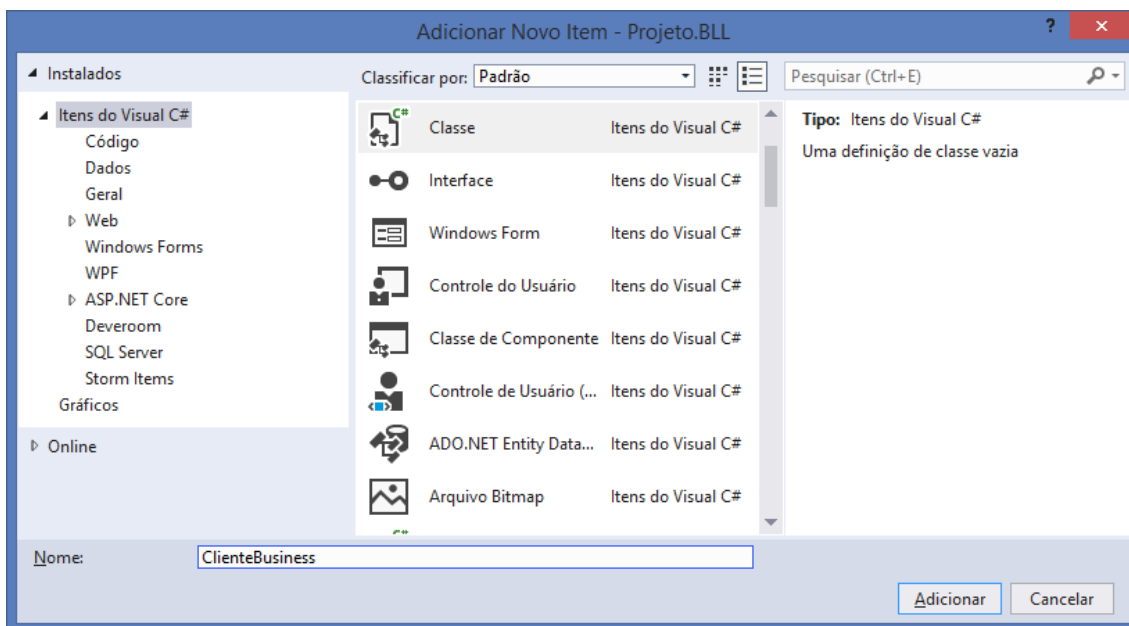
        string query = "select Email from Cliente where Email = @Email";

        command = new SqlCommand(query, connection);
        command.Parameters.AddWithValue("@Email", email);
        dataReader = command.ExecuteReader();

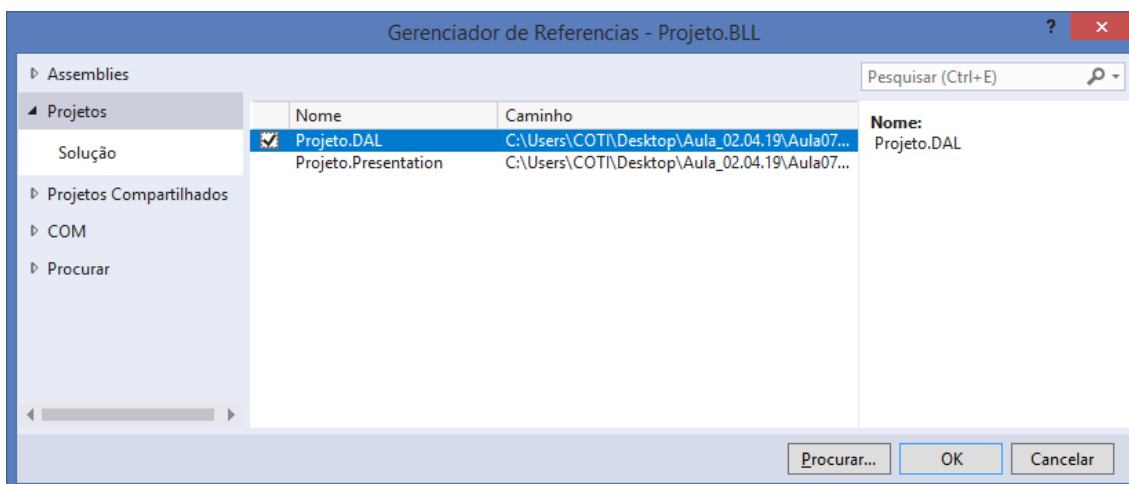
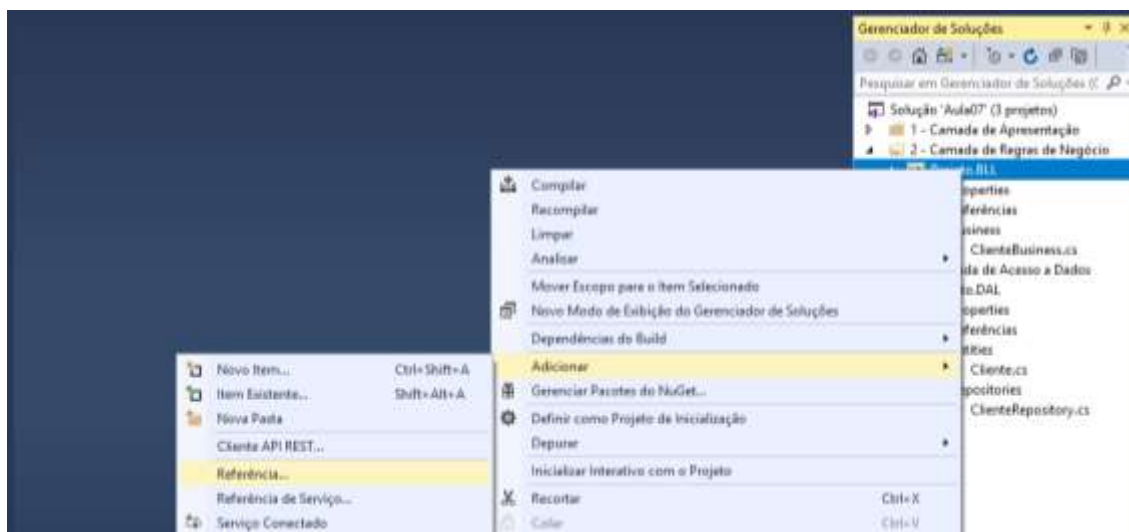
        return dataReader.HasRows;
    }
}
}
```

## 1.2 - Camada de Regras de Negócio: Class Library .NET Framework





## Adicionando referencia para o projeto DAL no projeto BLL:



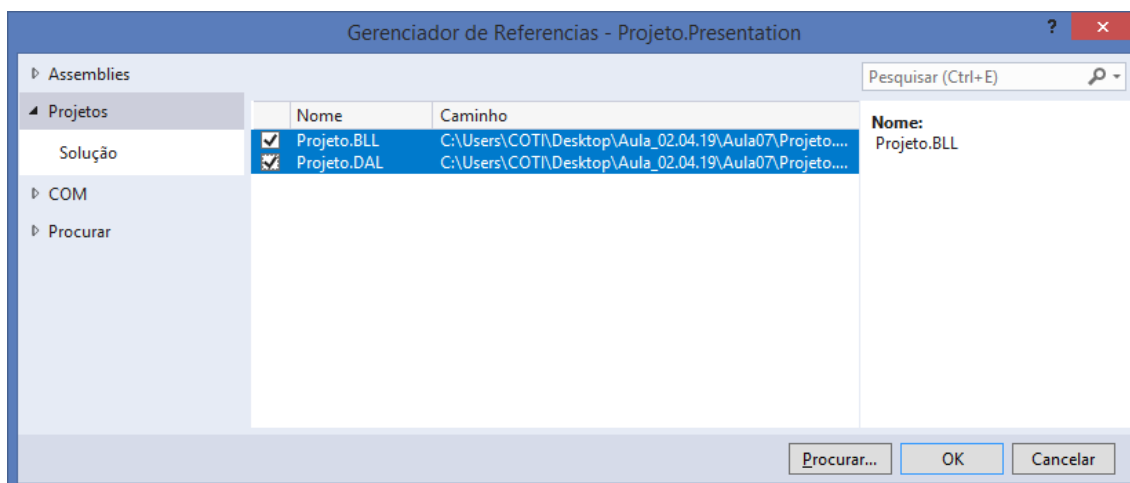
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto.DAL.Entities; //importando..
using Projeto.DAL.Repositories; //importando..

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.");
            }
        }
    }
}
```

## Voltando para o Projeto Asp.Net

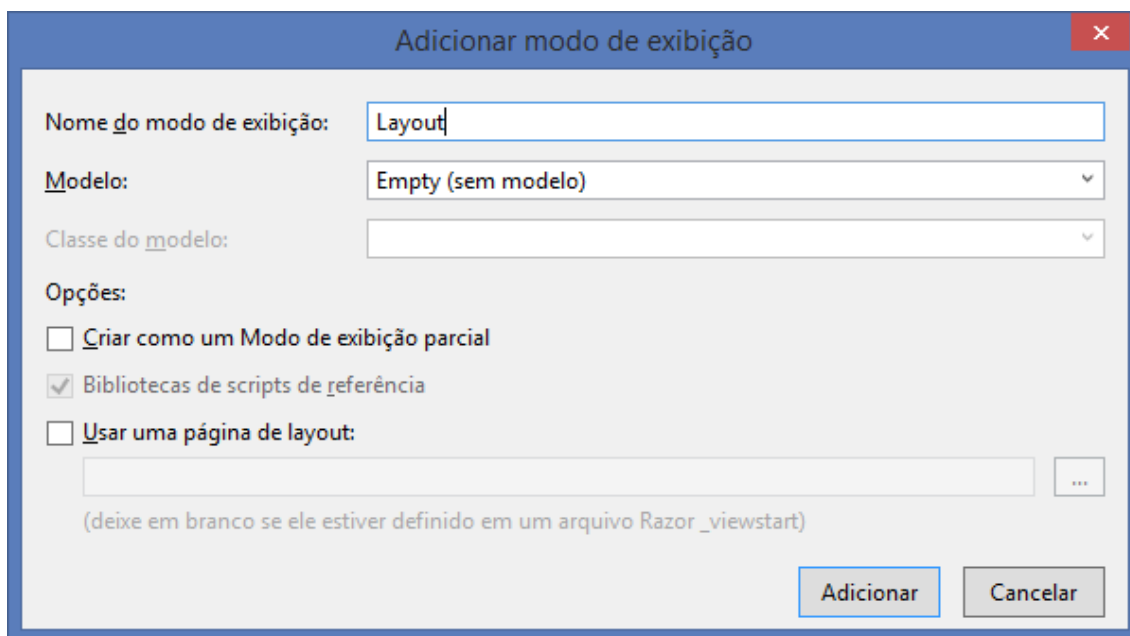
Adicionando referência para os demais projetos:





## MasterPage

Criando uma página de layout padrão. Geralmente em Asp.Net MVC as páginas de layout mestre ficam dentro de **\Views\Shared**



## .cshtml

Extensão de páginas utilizadas em projetos Asp.Net MVC, consiste de uma combinação de código HTML, CSS e JavaScript com CSharp

```
<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>COTI Informática</title>
</head>
<body>
    <div>

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

        <!-- INICIO DO CONTEUDO PRINCIPAL -->
        @RenderBody()
        <!-- FIM DO CONTEUDO PRINCIPAL -->

    </div>
</body>
</html>
```

## @Razor

Recurso utilizado em Asp.Net MVC para escrever pequenos trechos de código C# dentro de páginas HTML.

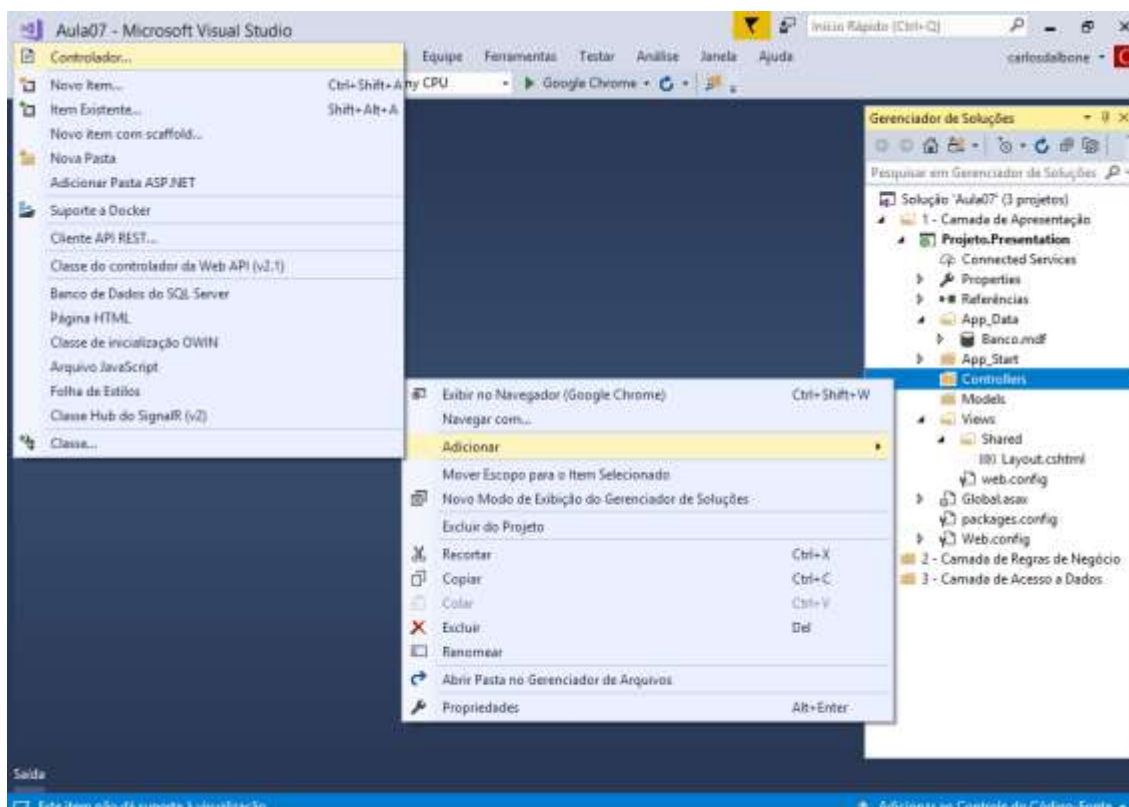
## Criando a página inicial do projeto:

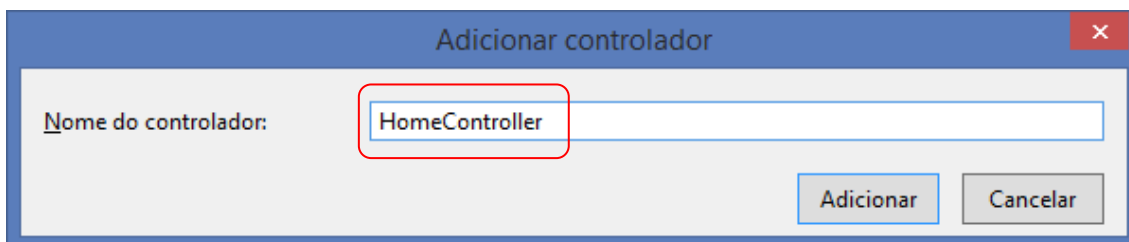
Em MVC, toda página web é composta de um endereço abaixo:

/Home/Index

[Controller] [View]

\*\* Toda página (View) em MVC é criada a partir de uma classe de controle. Para toda página que criamos em MVC precisamos de uma classe de controle que gerencie o conteúdo desta página.





```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
```

```
namespace Projeto.Presentation.Controllers
{
```

```
    public class HomeController : Controller
```

```
    {
```

```
        // GET: Home
```

```
        public ActionResult Index()
```

```
        {
```

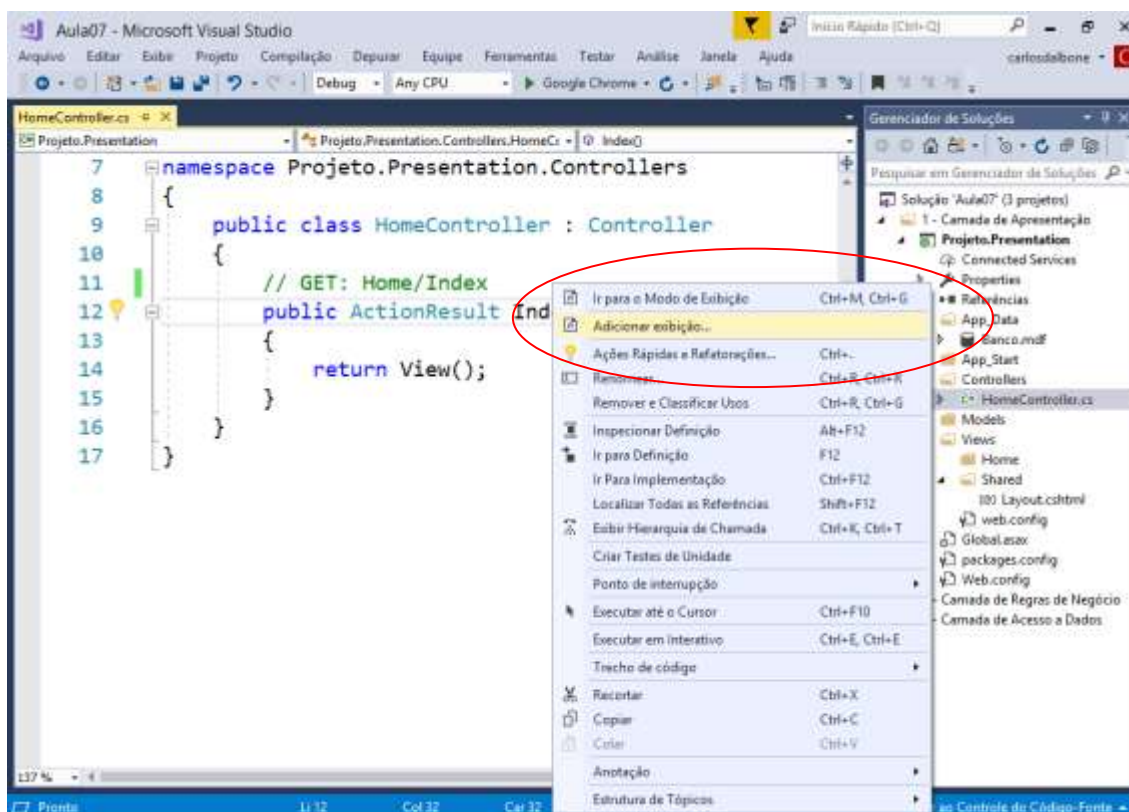
```
            return View();
```

```
        }
```

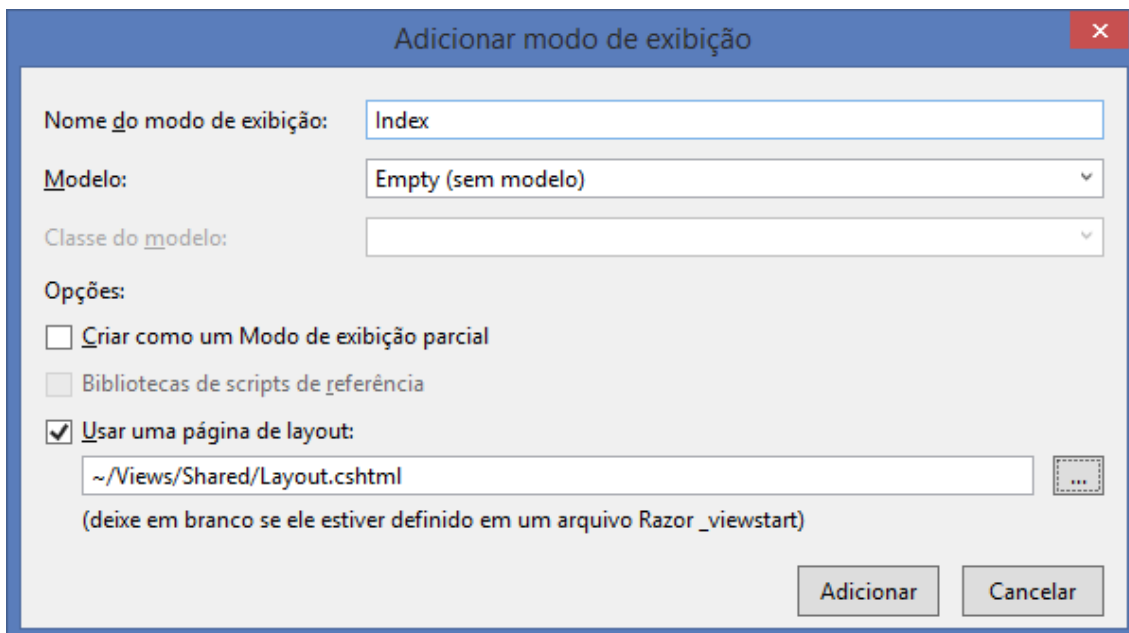
```
    }
```

```
}
```

**Criando a página:**



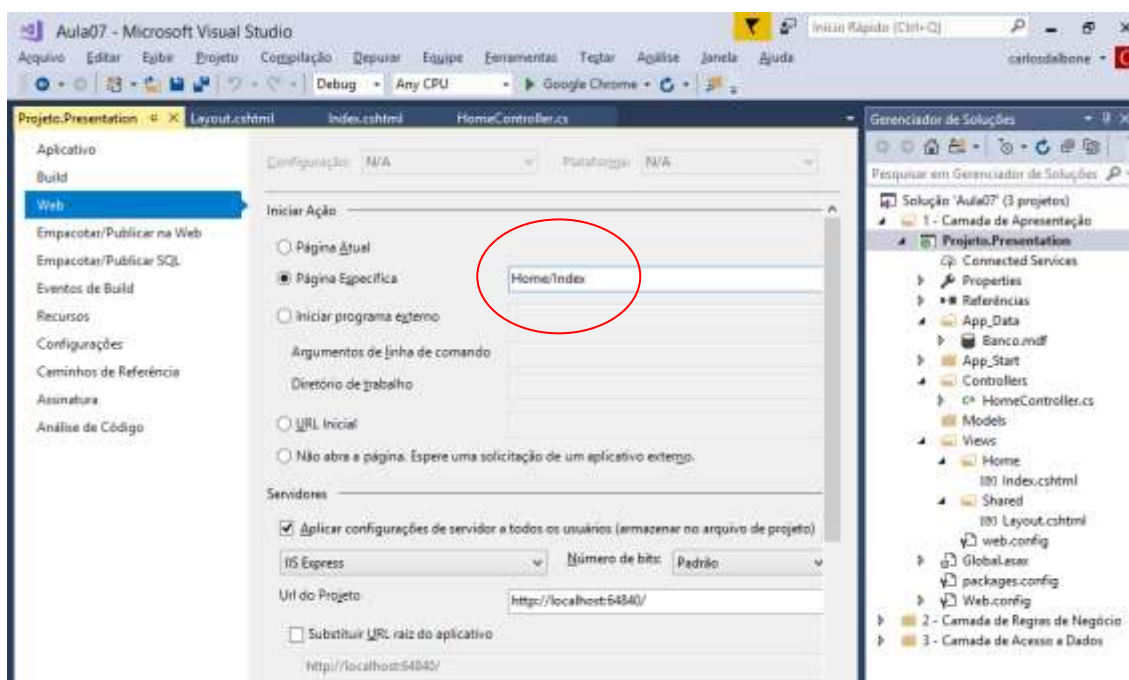
Selecione a opção **"Usar uma página de layout"**



```
@{
    ViewBag.Title = "Index";
    Layout = "~/Views/Shared/Layout.cshtml";
}
```

<h4>Seja bem vindo ao projeto!</h4>

**Definindo a página inicial do projeto no VisualStudio:**



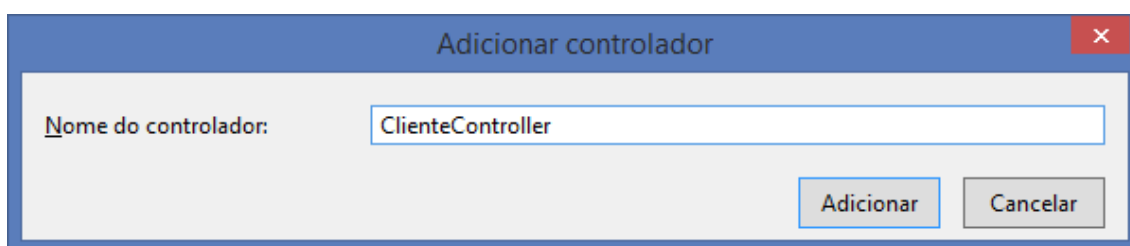
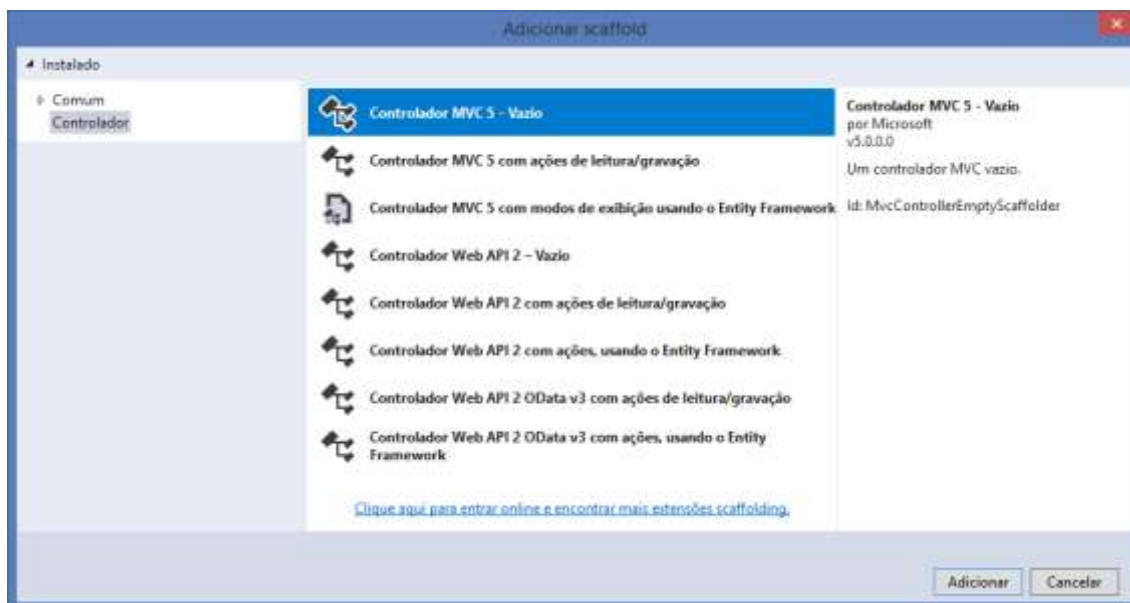
<http://localhost:64840/Home/Index>



Criando mais uma classe de controle no projeto MVC:

Rotas:

- **/Cliente/Cadastro** (Página para cadastro dos clientes)
- **/Cliente/Consulta** (Página para consulta dos clientes)



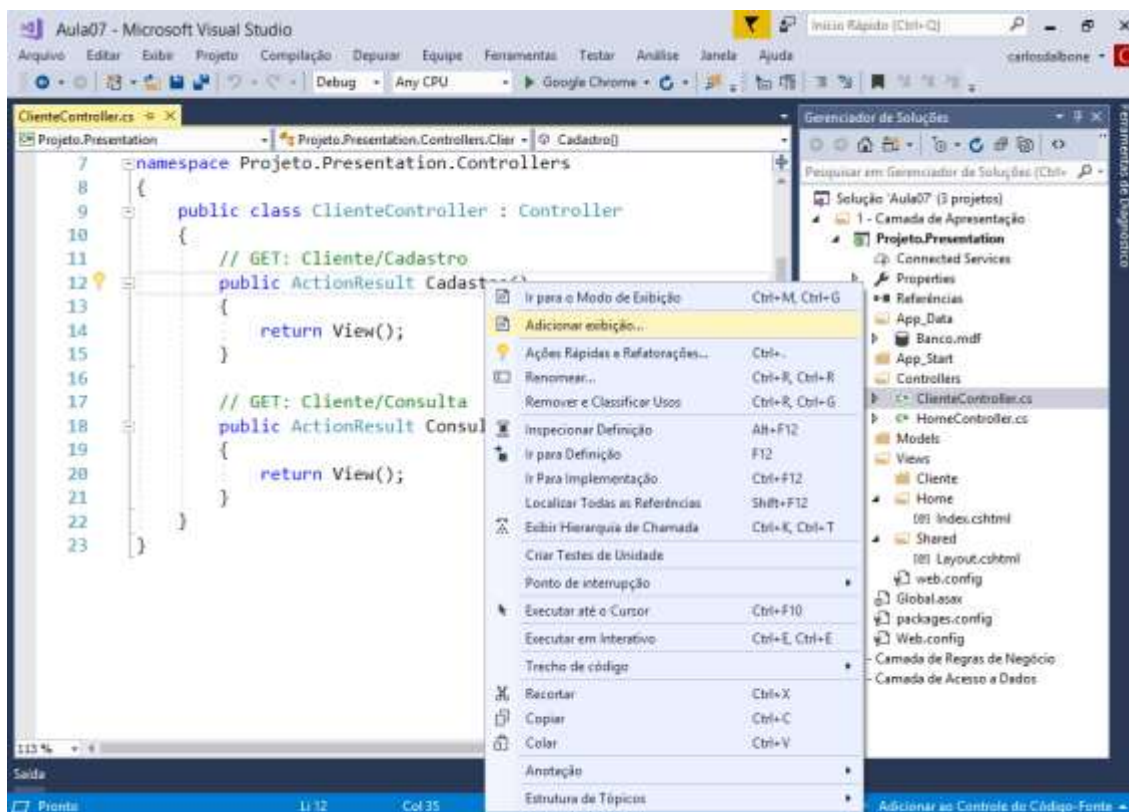


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace Projeto.Presentation.Controllers
{
    public class ClienteController : Controller
    {
        // GET: Cliente/Cadastro
        public ActionResult Cadastro()
        {
            return View();
        }

        // GET: Cliente/Consulta
        public ActionResult Consulta()
        {
            return View();
        }
    }
}
```

## Gerando as páginas de cadastro e consulta:





**Adicionar modo de exibição**

Nome do modo de exibição:

Modelo:

Classe do modelo:

Opções:

☐ Criar como um Modo de exibição parcial

☐ Bibliotecas de scripts de referência

☒ Usar uma página de layout:

...

(deixe em branco se ele estiver definido em um arquivo Razor \_viewstart)

**Adicionar modo de exibição**

Nome do modo de exibição:

Modelo:

Classe do modelo:

Opções:

☐ Criar como um Modo de exibição parcial

☐ Bibliotecas de scripts de referência

☒ Usar uma página de layout:

...

(deixe em branco se ele estiver definido em um arquivo Razor \_viewstart)

**Projeto.Presentation**

Connected Services

Properties

Referências

App\_Data

Banco.mdf

App\_Start

Controllers

C# ClienteController.cs

C# HomeController.cs

Models

Views

Cliente

@1 Cadastro.cshtml

@1 Consulta.cshtml

Home

@1 Index.cshtml

Shared

@1 Layout.cshtml

web.config

## /Home/Index

```
@{
    ViewBag.Title = "Index";
    Layout = "~/Views/Shared/Layout.cshtml";
}

<h4>Seja bem vindo ao projeto!</h4>

<ul>
    <li> <a href="/Cliente/Cadastro">Cadastrar Clientes</a> </li>
    <li> <a href="/Cliente/Consulta">Consultar Clientes</a> </li>
</ul>
```

## /Cliente/Cadastro

```
@{
    ViewBag.Title = "Cadastro";
    Layout = "~/Views/Shared/Layout.cshtml";
}

<h4>Cadastro de Clientes</h4>
<a href="/Home/Index">Página inicial</a>
```

## /Cliente/Consulta

```
@{
    ViewBag.Title = "Consulta";
    Layout = "~/Views/Shared/Layout.cshtml";
}

<h4>Consulta de Clientes</h4>
<a href="/Home/Index">Página inicial</a>
```

