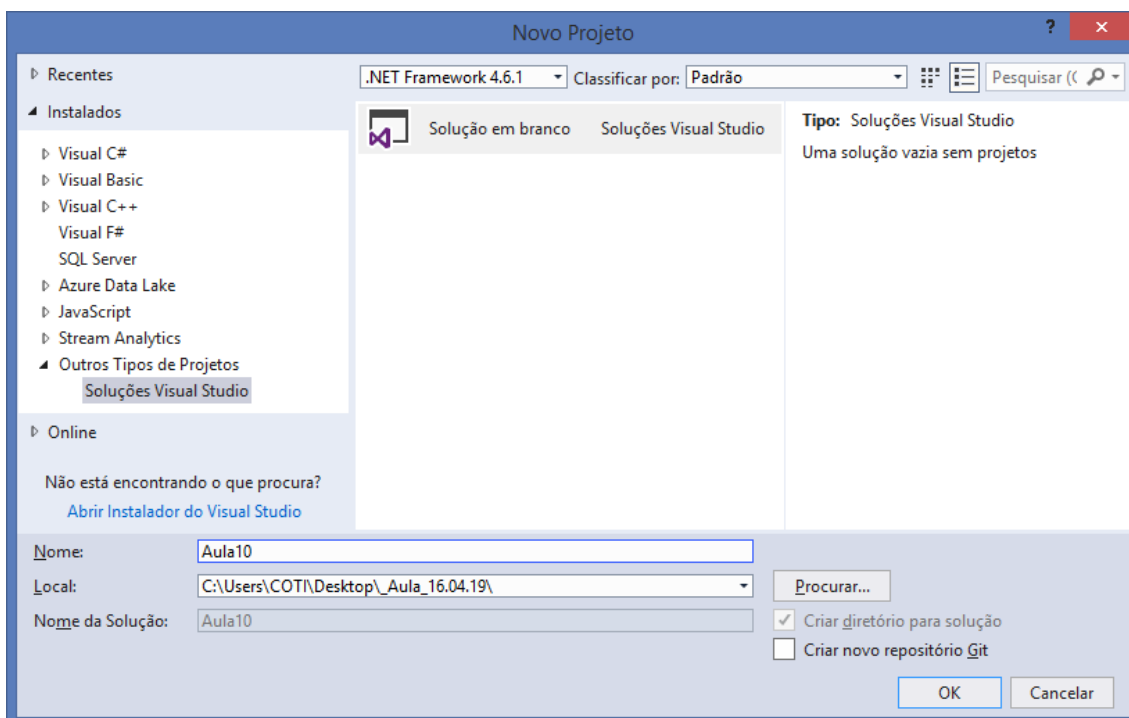
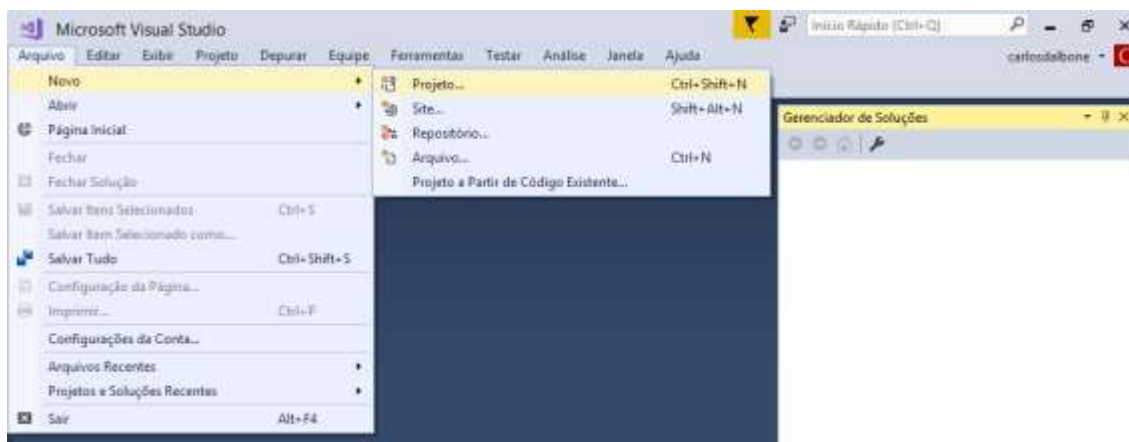
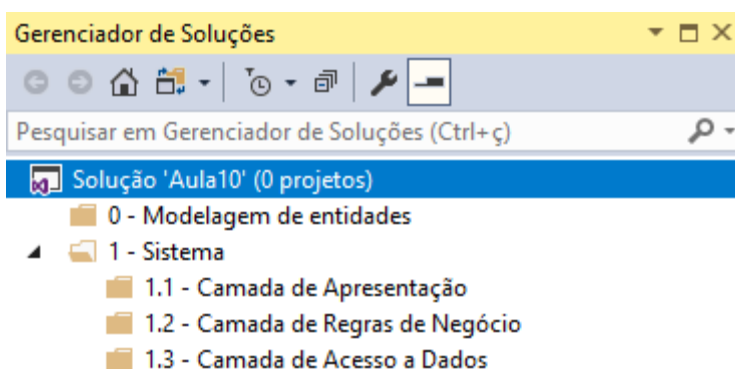


## Criando uma nova solution em branco:

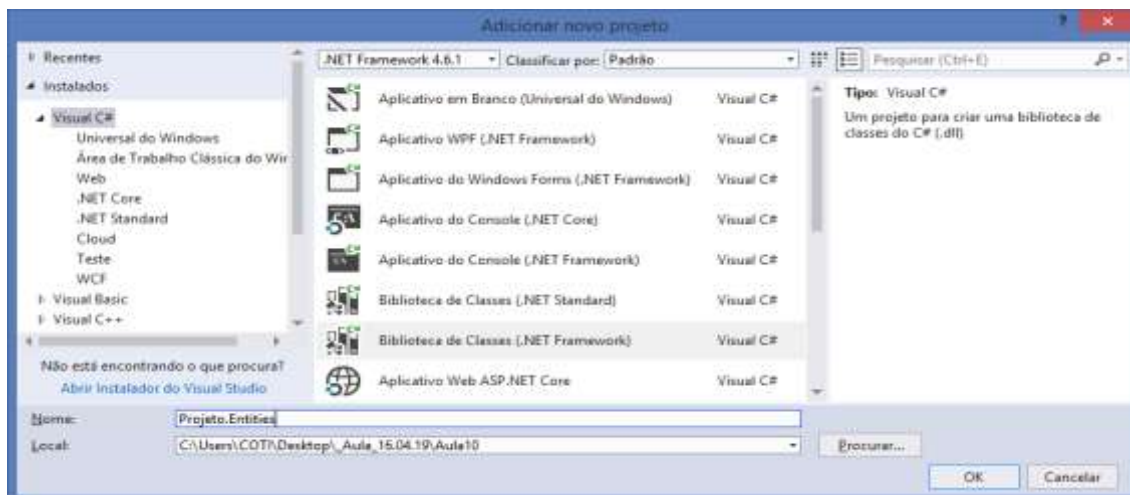


## Organização da Solution:



## 0 - Modelagem de entidades

### Biblioteca de classes



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

namespace Projeto.Entities
{
    public class Produto
    {
        //prop + 2x[tab]
        public int IdProduto { get; set; }
        public string Nome { get; set; }
        public decimal Preco { get; set; }
        public int Quantidade { get; set; }

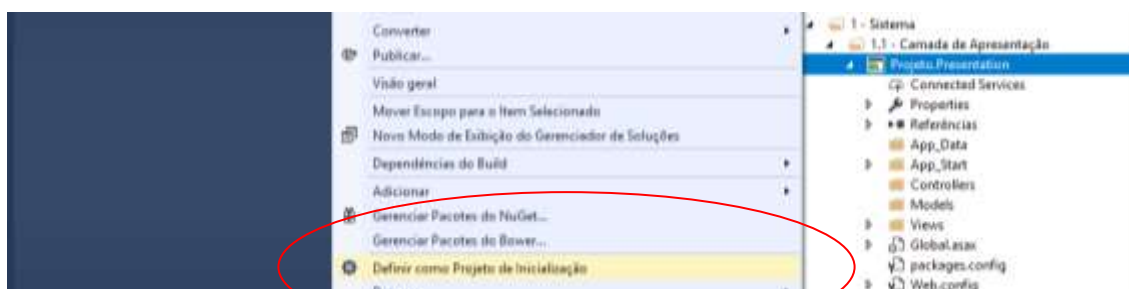
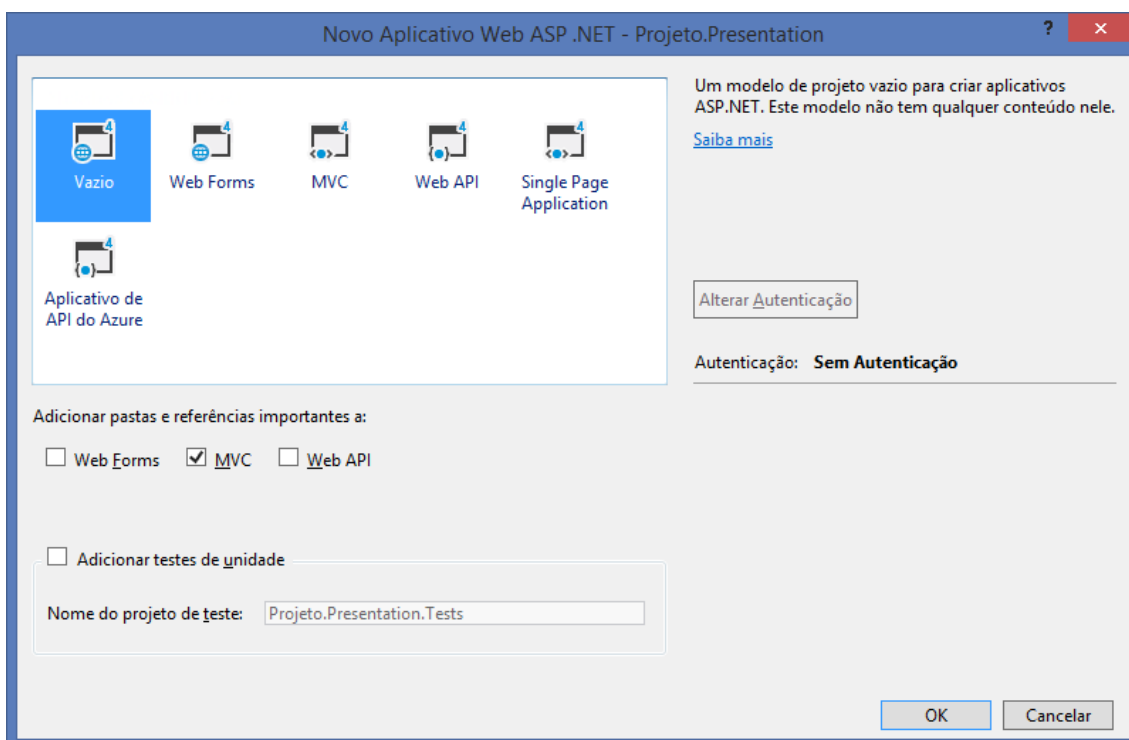
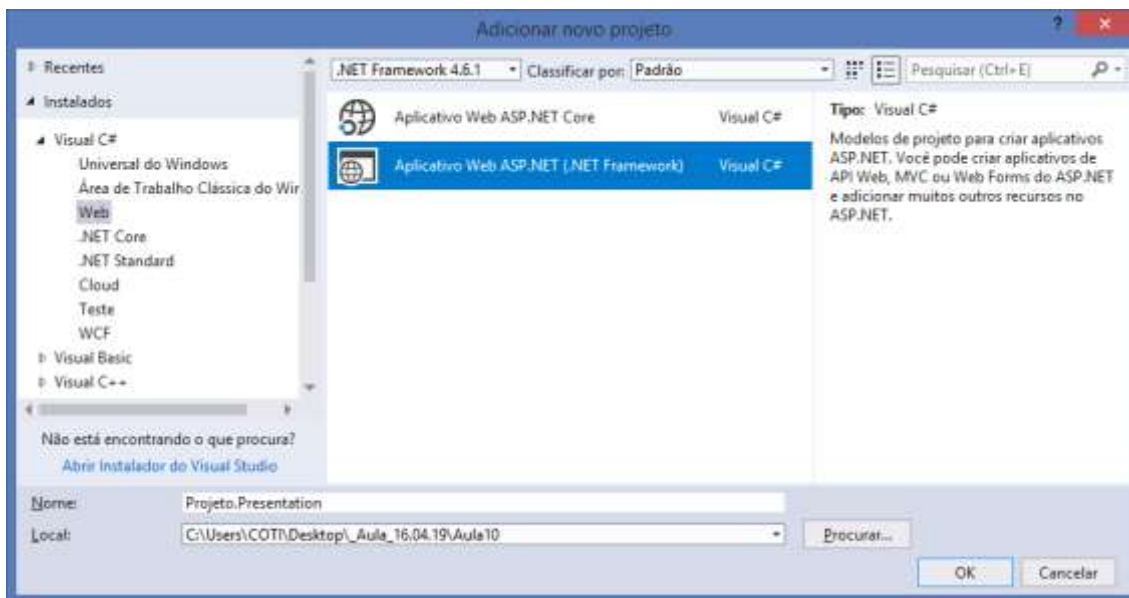
        //construtor default -> ctor + 2x[tab]
        public Produto()
        {
            //vazio
        }

        //sobrecarga de construtores (overloading)
        public Produto(int idProduto, string nome, decimal preco, int quantidade)
        {
            IdProduto = idProduto;
            Nome = nome;
            Preco = preco;
            Quantidade = quantidade;
        }

        //sobrescrita (override) do método ToString()
        public override string ToString()
        {
            return $"Id: {IdProduto}, Nome: {Nome},
                Preço: {Preco}, Quantidade: {Quantidade}";
        }
    }
}
```

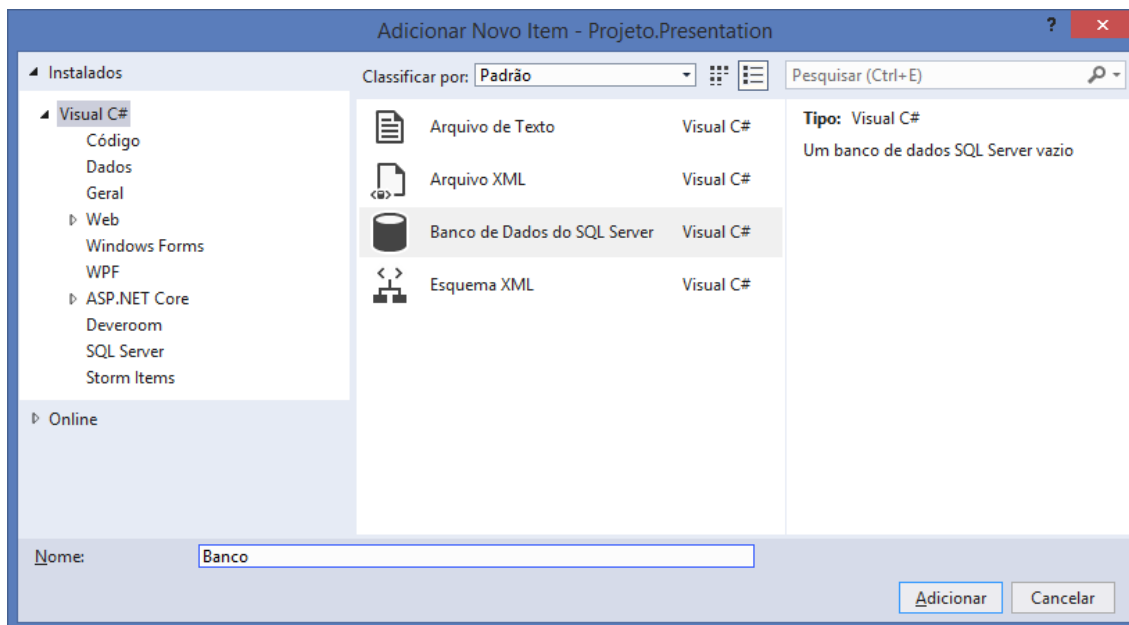
## 1.1 - Camada de Apresentação

### Projeto Asp.Net MVC

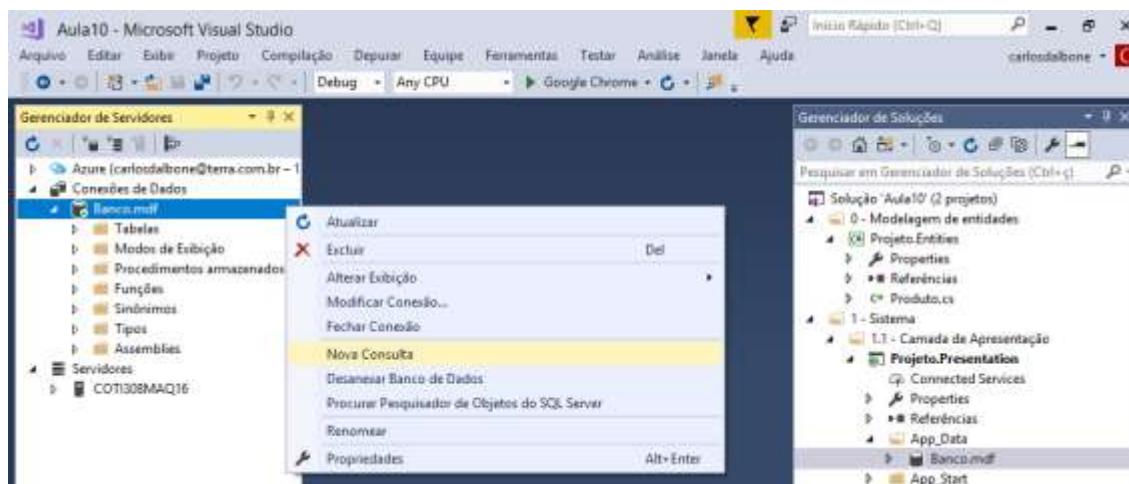


## App\_Data/Banco.mdf

Arquivo de banco de dados local do SqlServer



Criando a tabela de Produtos:



```
CREATE TABLE PRODUTO(
    IDPRODUTO          INTEGER          IDENTITY(1,1),
    NOME                NVARCHAR(150)   NOT NULL,
    PRECO               DECIMAL(18,2)   NOT NULL,
    QUANTIDADE          INTEGER          NOT NULL,
    PRIMARY KEY(IDPRODUTO))
```

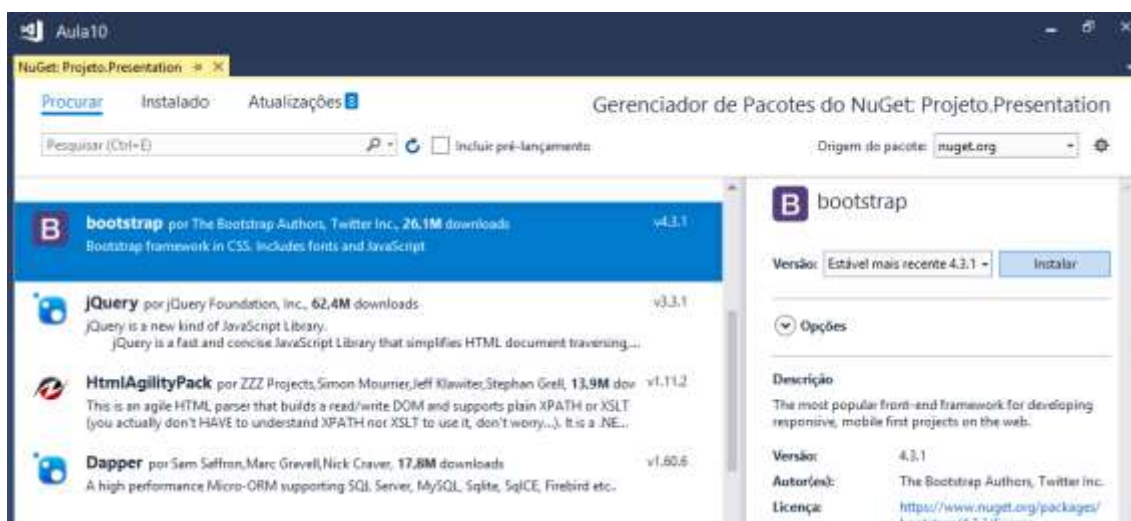
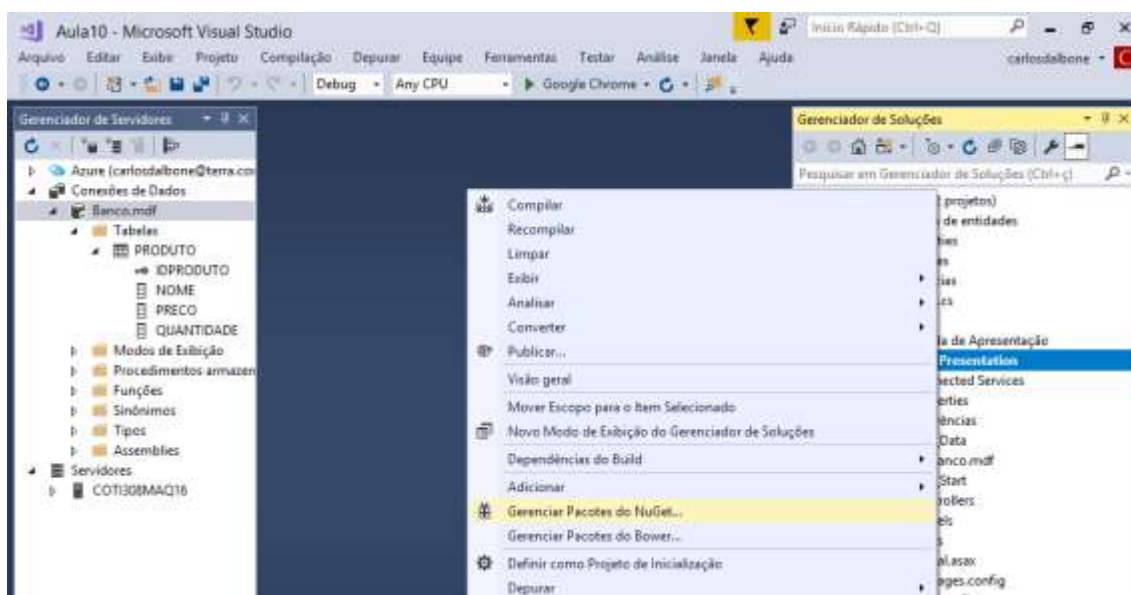
## \Web.config.xml

Mapeamento da connectionstring

```
<!-- Mapeamento da connectionstring -->
<connectionStrings>
  <add
    name="projeto"
    connectionString="Data Source=(LocalDB)\MSSQLLocalDB;
    AttachDbFilename=C:\Users\COTI\Desktop\_Aula_16.04.19\
    Aula10\Projeto.Presentation\App_Data\Banco.mdf;
    Integrated Security=True"
  />
</connectionStrings>
```

## Instalando o bootstrap no projeto Asp.Net

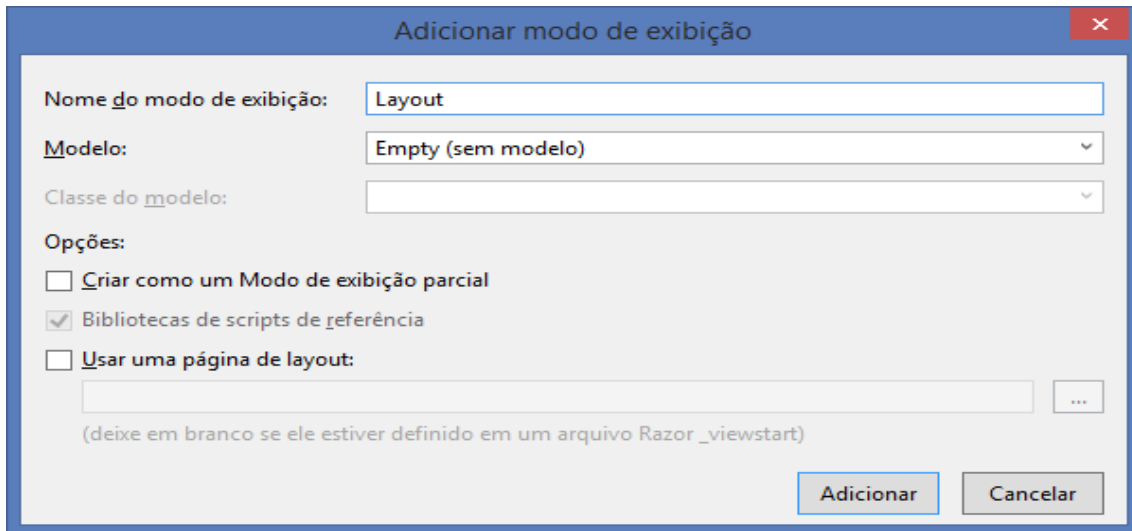
Gerenciador de pacotes do NuGet



## MasterPage

Página de Layout padrão

Geralmente estas páginas de layout (MasterPages) ficarão dentro da pasta Views, em uma subpasta chamada de **Shared**



```
<!DOCTYPE html>

<html>
<head>
  <meta name="viewport" content="width=device-width" />
  <title>Projeto</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 Produtos</h3>
      <p class="text-white">
        Aula de C# WebDeveloper - COTI Informática
      </p>
    </div>
    <br/>

    @RenderBody()

  </div>

  <!-- arquivos javascript -->
  <script src="/Scripts/jquery-3.0.0.min.js"></script>
  <script src="/Scripts/bootstrap.min.js"></script>

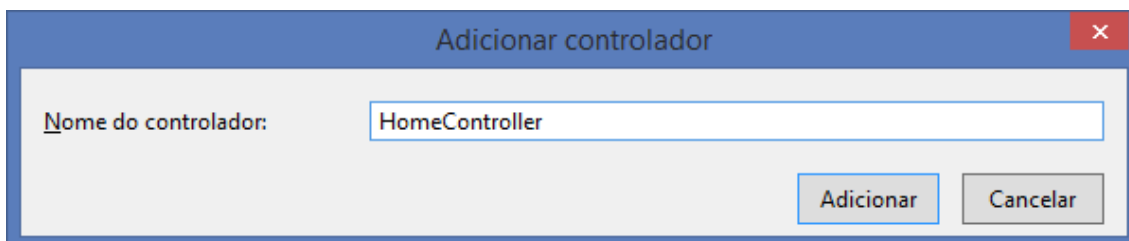
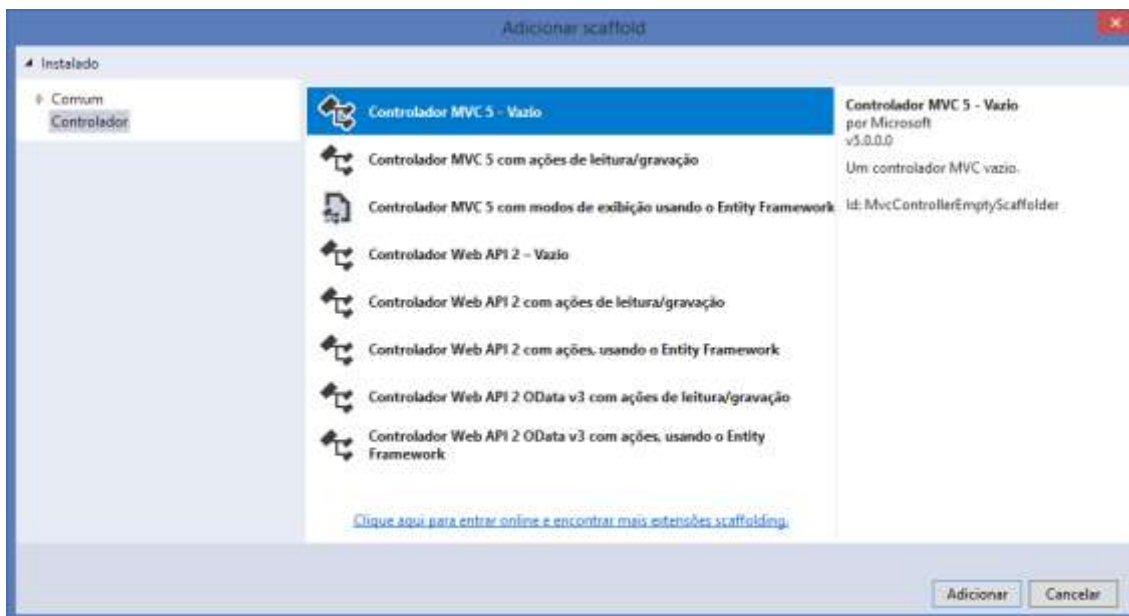
</body>
</html>
```



Criando a página inicial do projeto:

/Home/Index

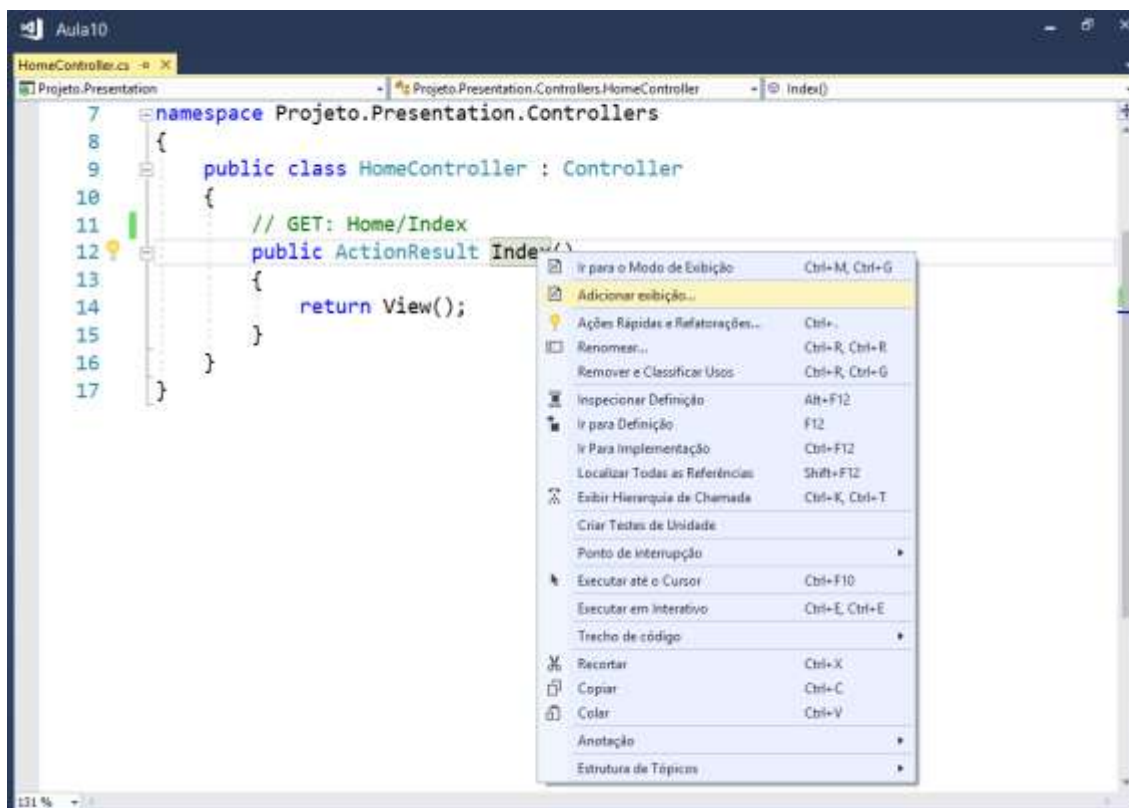
[Controller] [View]



```
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/Index
        public ActionResult Index()
        {
            return View();
        }
    }
}
```

## Criando a página:



Adicionar modo de exibição

Nome do modo de exibição:

Index

Modelo:

Empty (sem modelo)

Classe do modelo:

Opções:

☐ Criar como um Modo de exibição parcial

☐ Bibliotecas de scripts de referência

☒ Usar uma página de layout:

~/Views/Shared/Layout.cshtml

...

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

Adicionar

Cancelar

@{

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

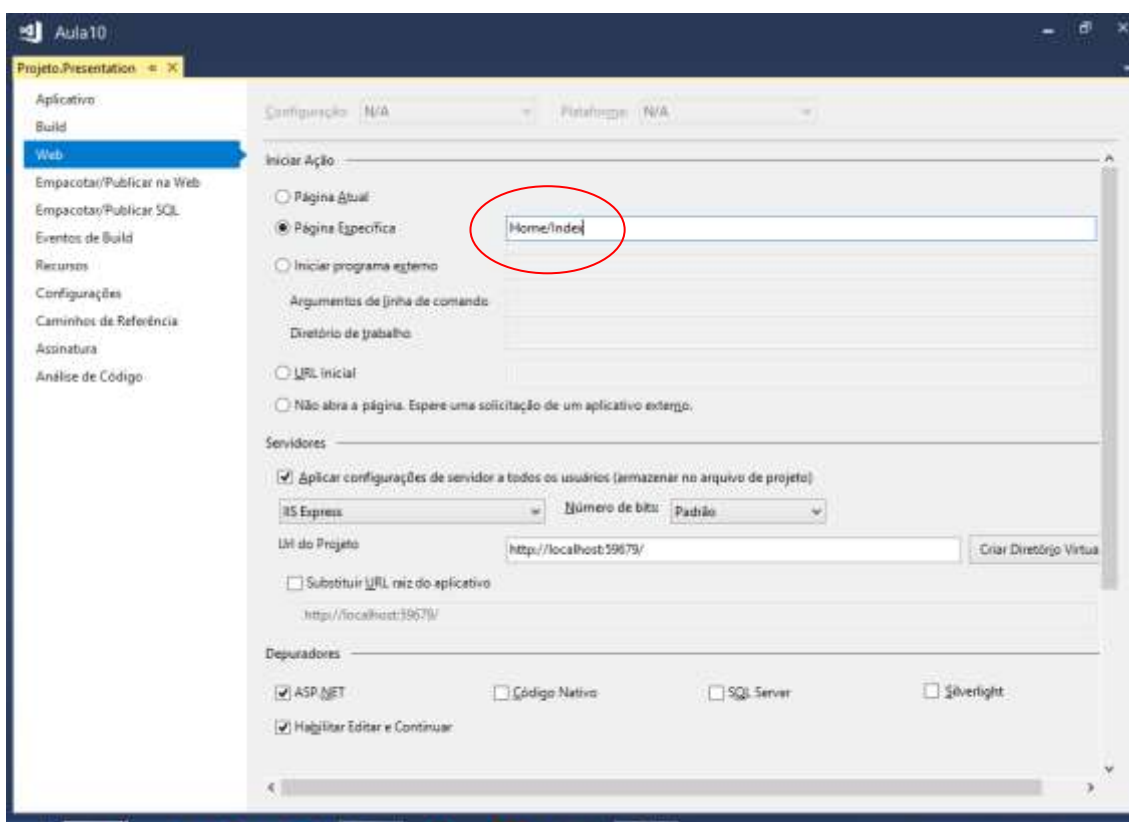
}

<h4>Seja bem vindo ao projeto</h4>  
Escolha a ação desejada:

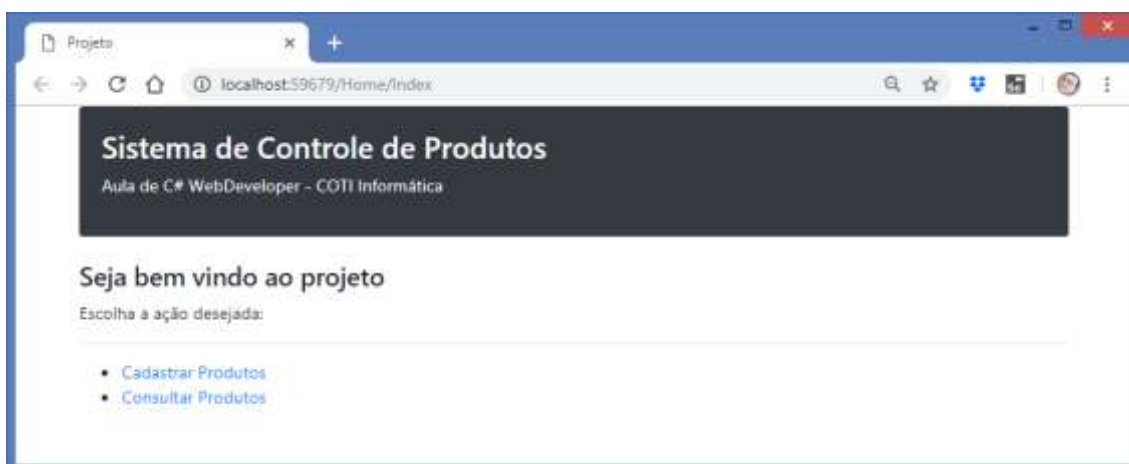


```
<hr/>
<ul>
  <li>
    <a href="/Produto/Cadastro">Cadastrar Produtos</a>
  </li>
  <li>
    <a href="/Produto/Consulta">Consultar Produtos</a>
  </li>
</ul>
```

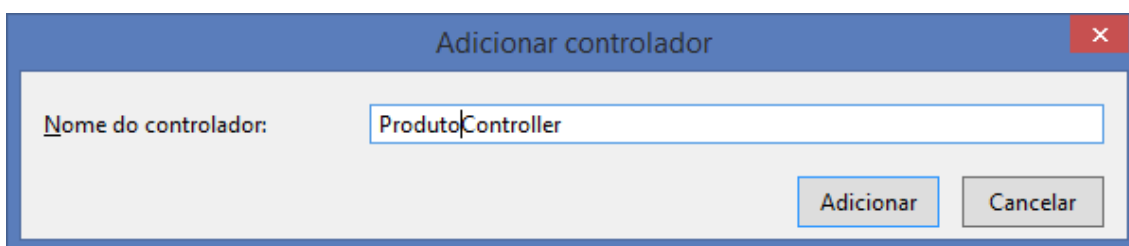
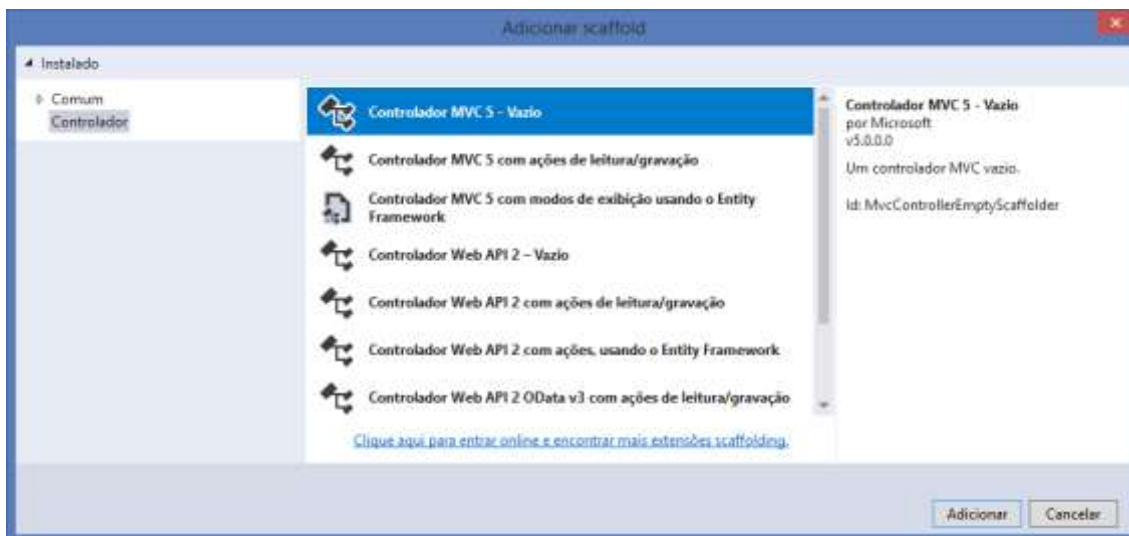
## Definindo a página inicial do projeto no VisualStudio:



<http://localhost:59679/Home/Index>



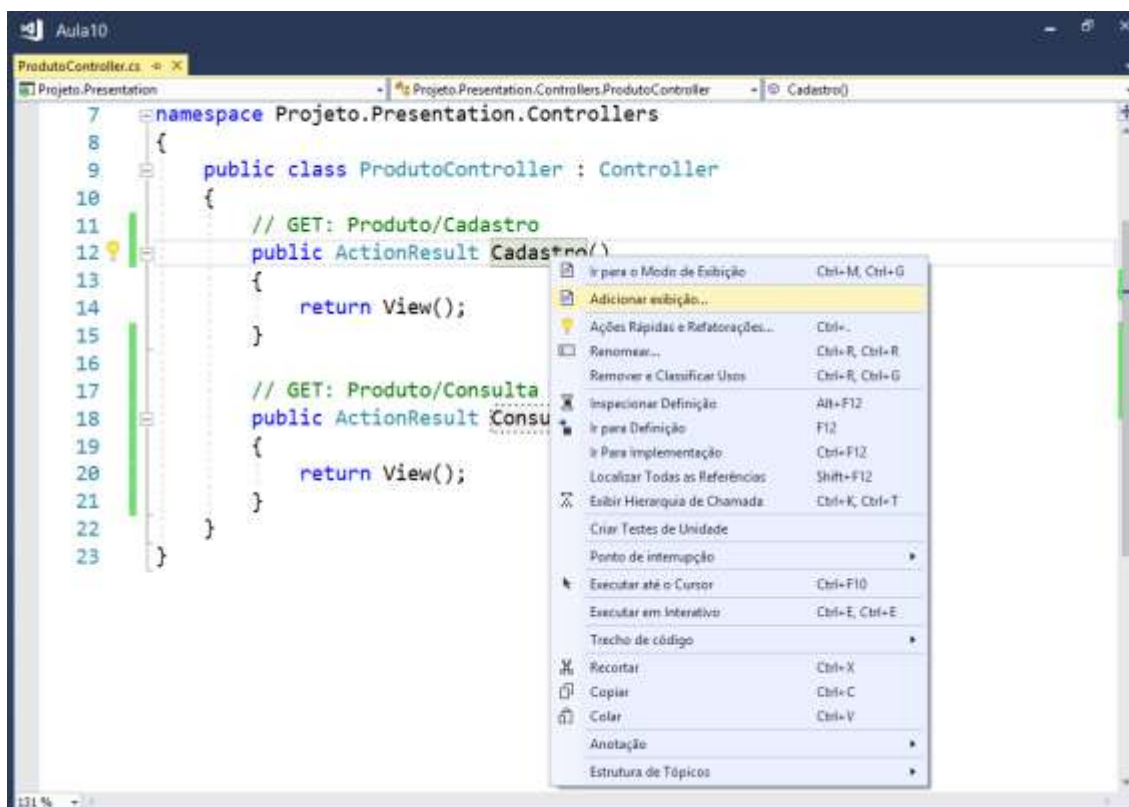
Criando a classe "**ProdutoController**"



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

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

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



Adicionar modo de exibição

Nome do modo de exibição:

Cadastro

Modelo:

Empty (sem modelo)

Classe do modelo:

Opções:

☐ Criar como um Modo de exibição parcial
 ☐ Bibliotecas de scripts de referência
 ☒ Usar uma página de layout:
 

~/Views/Shared/Layout.cshtml

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

Adicionar

Cancelar

```

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

```

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

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

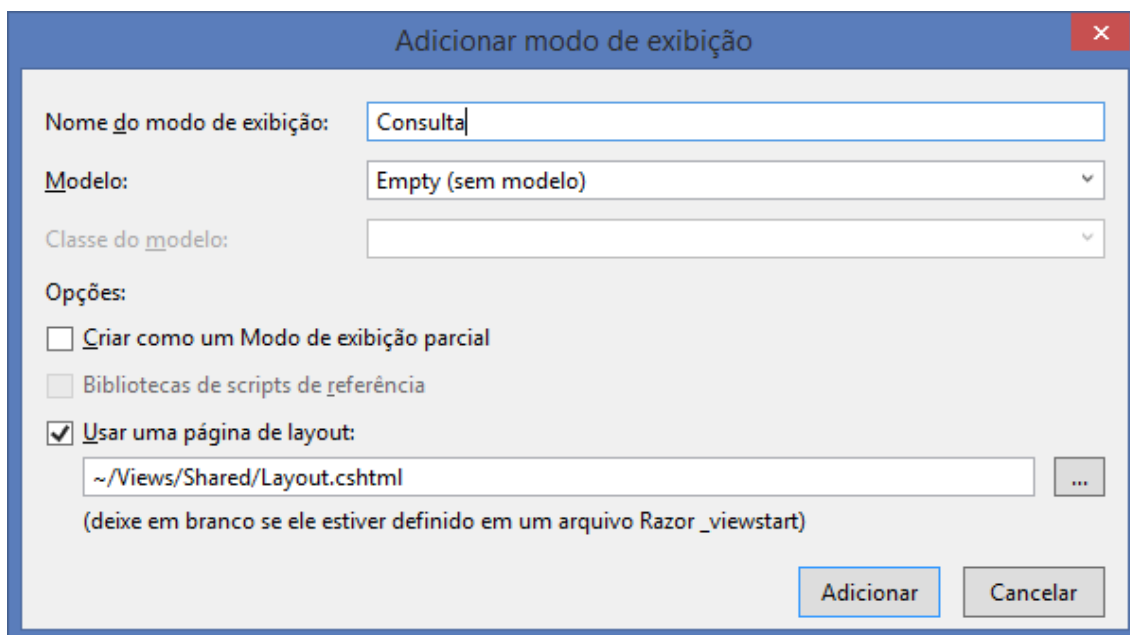
        <label>Nome do Produto:</label>
        <input type="text" id="nome" class="form-control"
            placeholder="Nome do Produto"/>
        <br/>

        <label>Preço:</label>
        <input type="text" id="preco" class="form-control"
            placeholder="Preço"/>
        <br/>

        <label>Quantidade:</label>
        <input type="text" id="quantidade" class="form-control"
            placeholder="Quantidade"/>
        <br/>

        <button id="btncadastro" class="btn btn-success">
            Cadastrar Produto
        </button>

    </div>
</div>
```



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

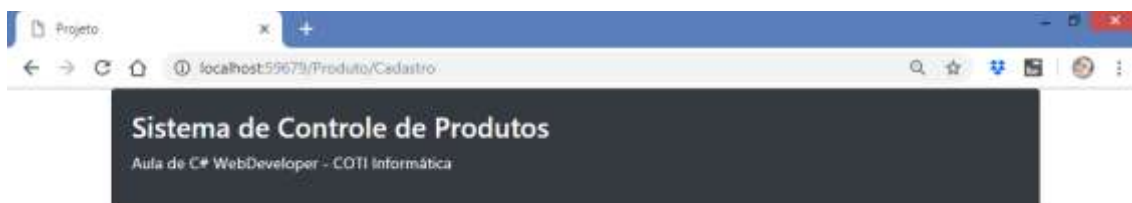
```
<h4>Consulta de Produtos</h4>
<a href="/Home/Index">Página inicial</a>
<hr/>

<table id="tabela" class="table table-bordered table-hover table-striped">
    <thead>
        <tr>
```

```
<th class="bg-info">Código</th>
<th class="bg-info">Nome do Produto</th>
<th class="bg-info">Preço</th>
<th class="bg-info">Quantidade</th>
<th class="bg-info">Total</th>
<th class="bg-info">Operações</th>
</tr>
</thead>
<tbody>

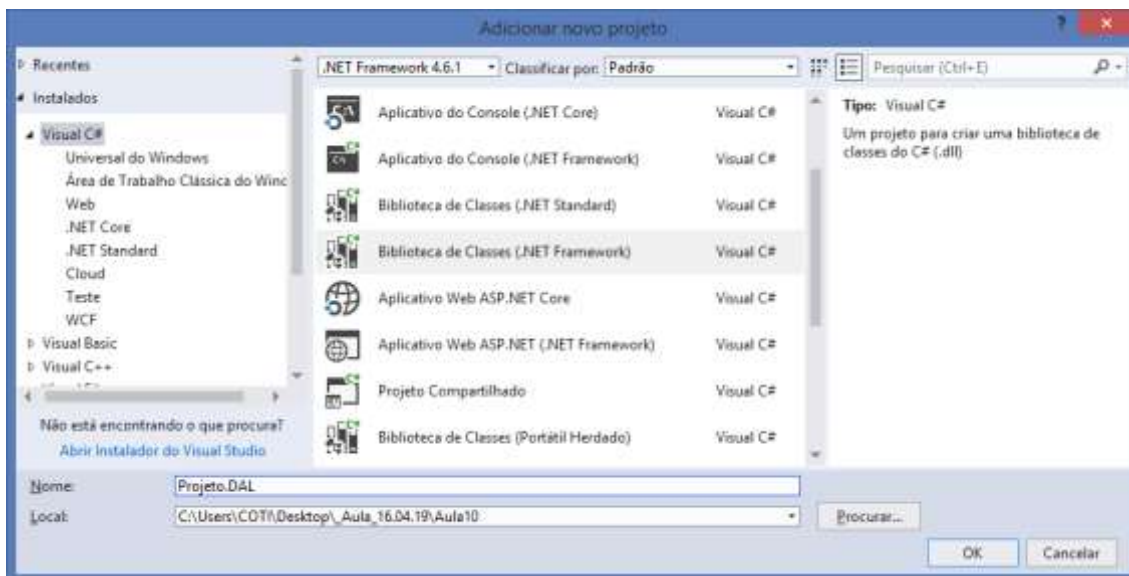
</tbody>
</table>
```

## Executando:

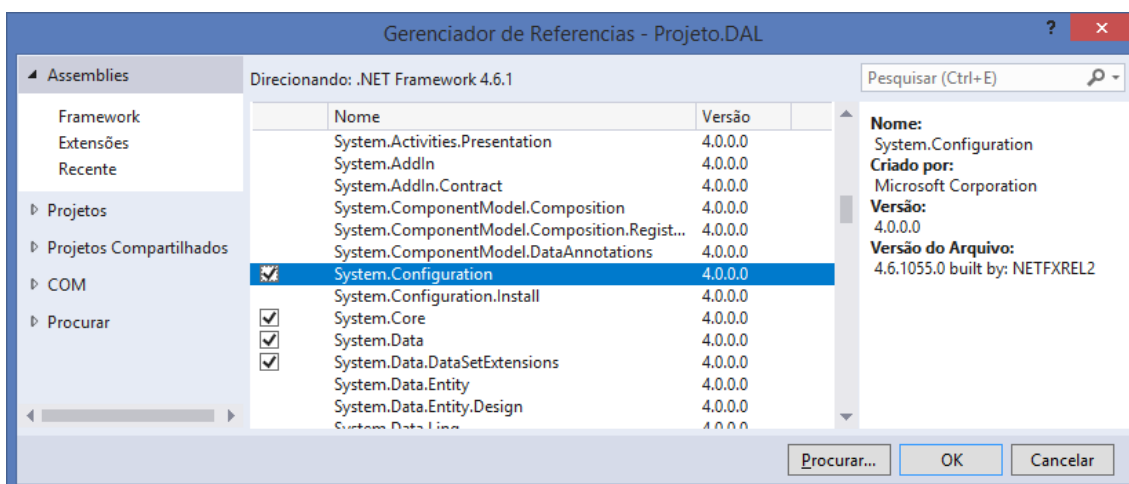
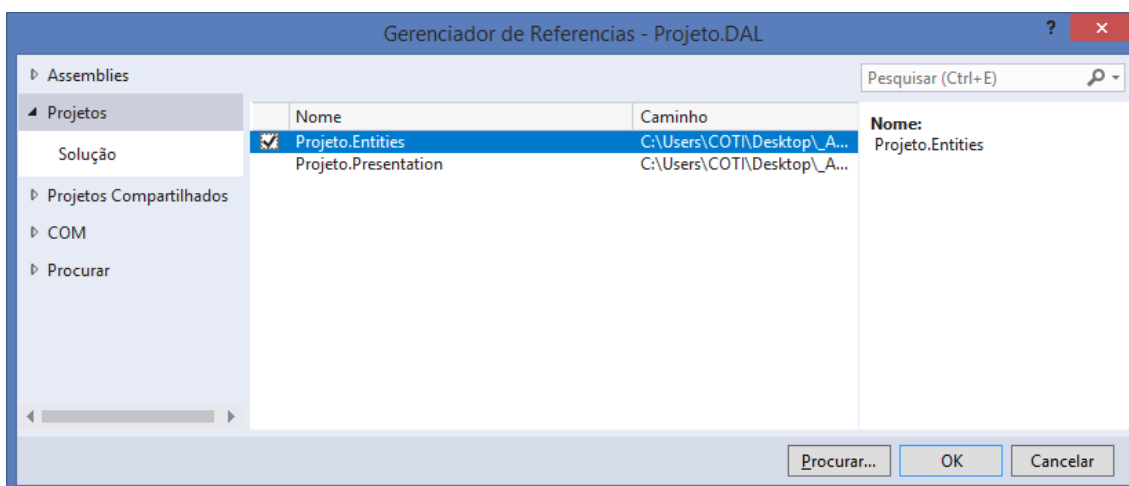


## 1.3 - Camada de Acesso a dados

Class Library .NET Framework

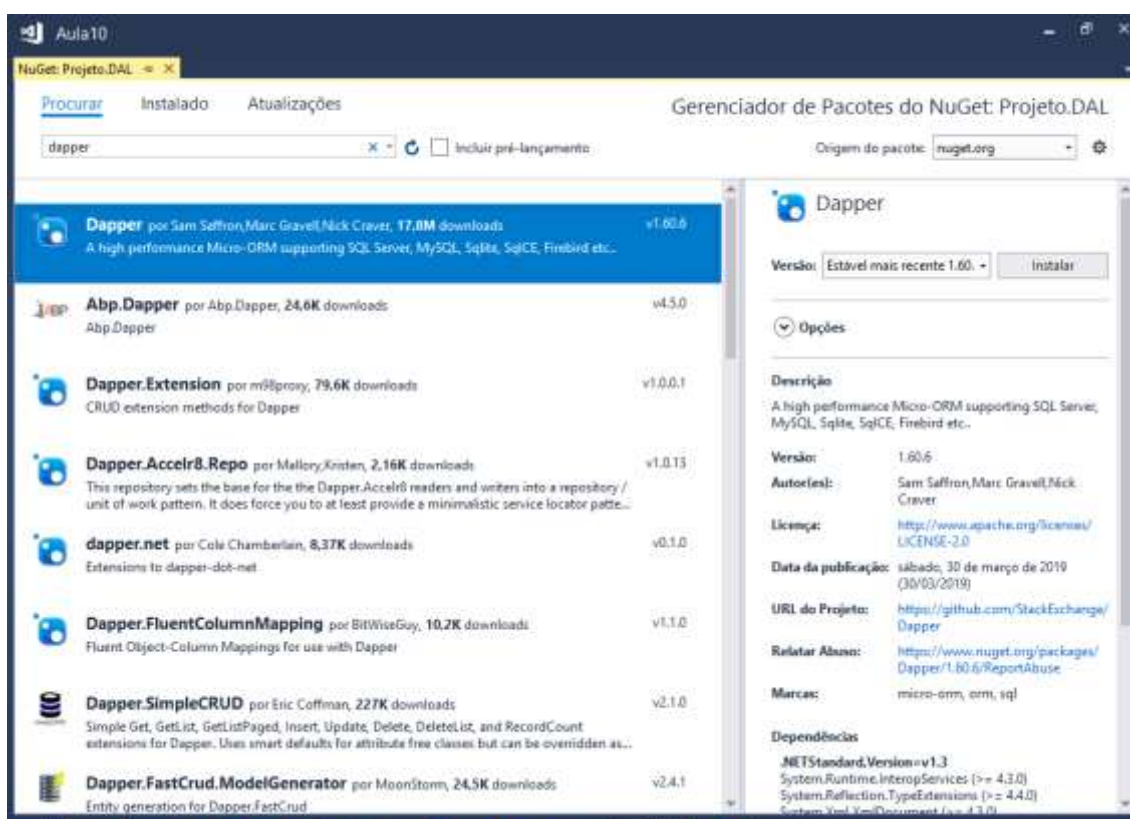
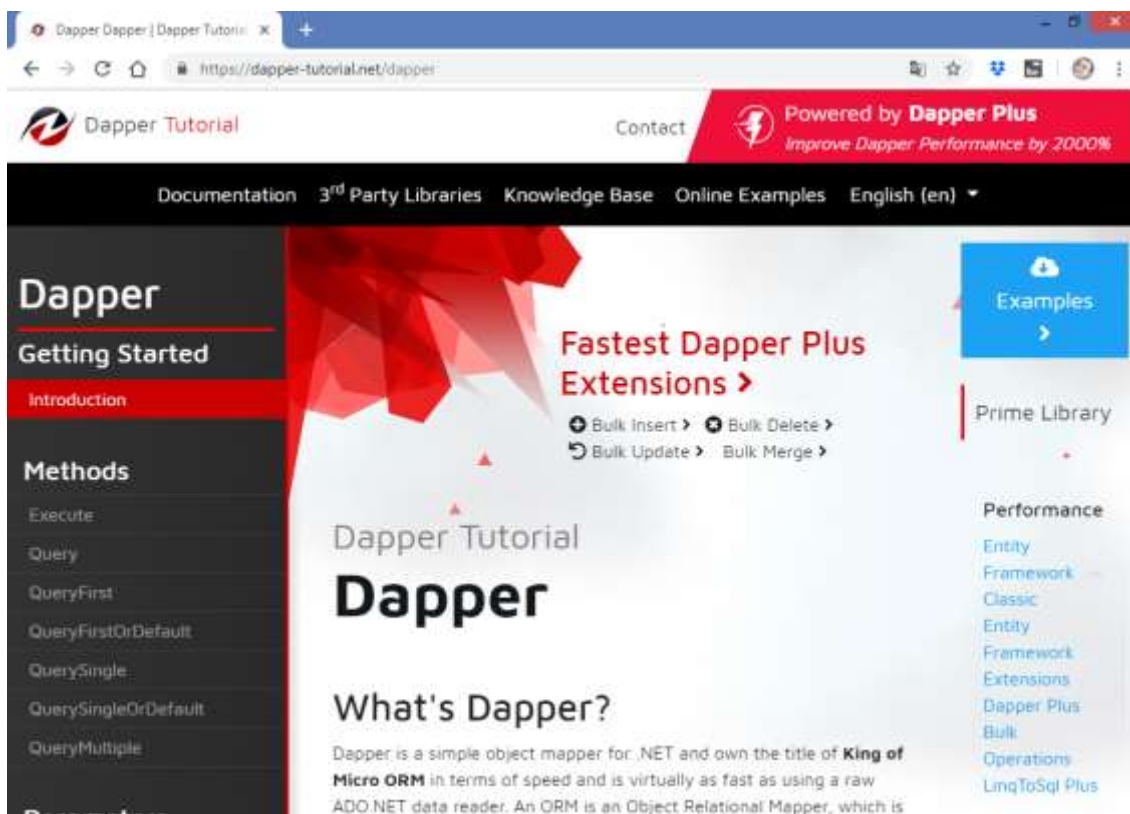


**Adicionando referências no projeto DAL:**





## Dapper (<https://dapper-tutorial.net/dapper>) Framework para acesso a banco de dados em .NET



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient; //SqlServer
using System.Configuration; //connectionstring
using Dapper; //framework para acesso ao BD
using Projeto.Entities; //classes de entidade

namespace Projeto.DAL
{
    public class ProdutoRepository
    {
        //atributo
        private string connectionString;

        //construtor -> ctor + 2x[tab]
        public ProdutoRepository()
        {
            connectionString = ConfigurationManager.ConnectionStrings["projeto"]
                .ConnectionString;
        }

        //método para inserir um novo produto no banco de dados
        public void Insert(Produto produto)
        {
            using (SqlConnection conn = new SqlConnection(connectionString))
            {
                string query = "INSERT INTO PRODUTO(NOME, PRECO, QUANTIDADE) "
                    + "VALUES(@Nome, @Preco, @Quantidade)";

                conn.Execute(query, produto);
            }
        }

        //método para atualizar um produto no banco de dados
        public void Update(Produto produto)
        {
            using (SqlConnection conn = new SqlConnection(connectionString))
            {
                string query = "UPDATE PRODUTO SET NOME = @Nome,
                    PRECO = @Preco, "
                    + "QUANTIDADE = @Quantidade
                    WHERE IDPRODUTO = @IdProduto";

                conn.Execute(query, produto);
            }
        }

        //método para excluir um produto no banco de dados
        public void Delete(int idProduto)
        {
            using (SqlConnection conn = new SqlConnection(connectionString))
            {
                string query = "DELETE FROM PRODUTO
                    WHERE IDPRODUTO = @IdProduto";

                conn.Execute(query, new { IdProduto = idProduto });
            }
        }
    }
}
```

```
//método para retornar todos os produtos do banco de dados
public List<Produto> FindAll()
{
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM PRODUTO";

        return conn.Query<Produto>(query)
            .ToList();
    }
}

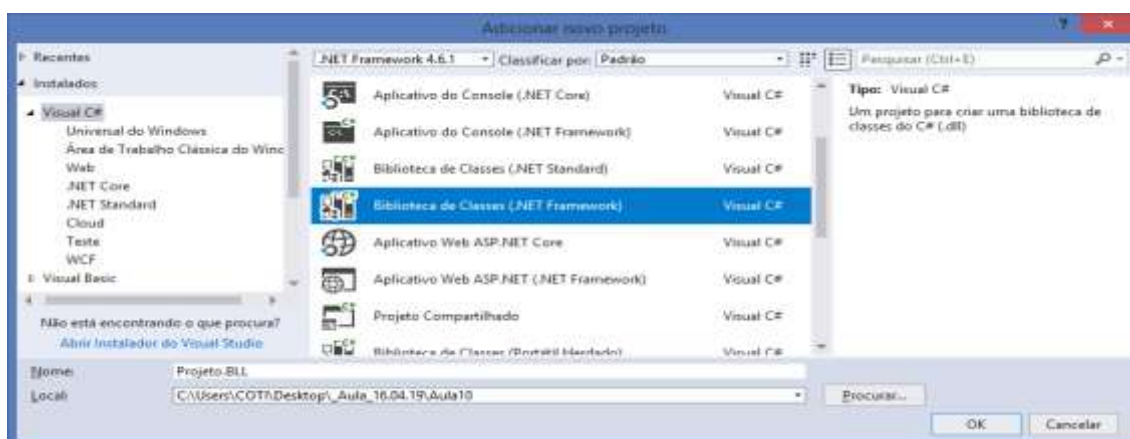
//método para retornar 1 produto pelo id
public Produto FindById(int idProduto)
{
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM PRODUTO
            WHERE IDPRODUTO = @IdProduto";

        return conn.Query<Produto>(query,
            new { IdProduto = idProduto })
            .SingleOrDefault();
    }
}

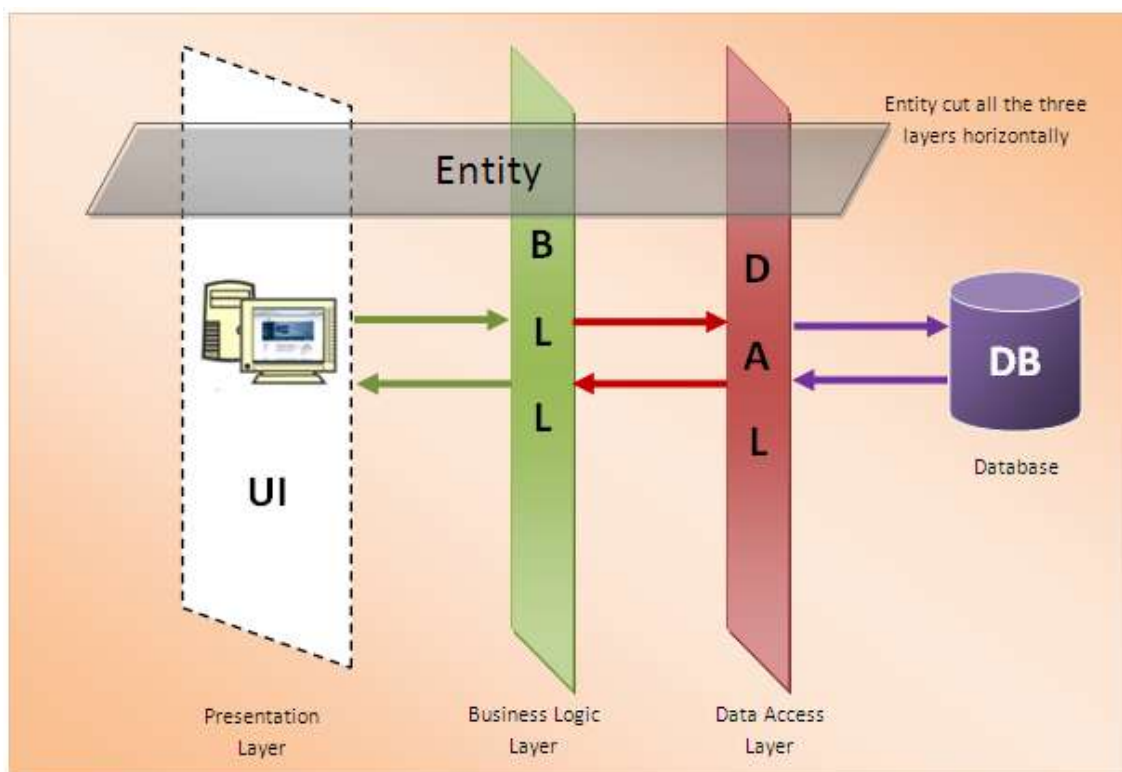
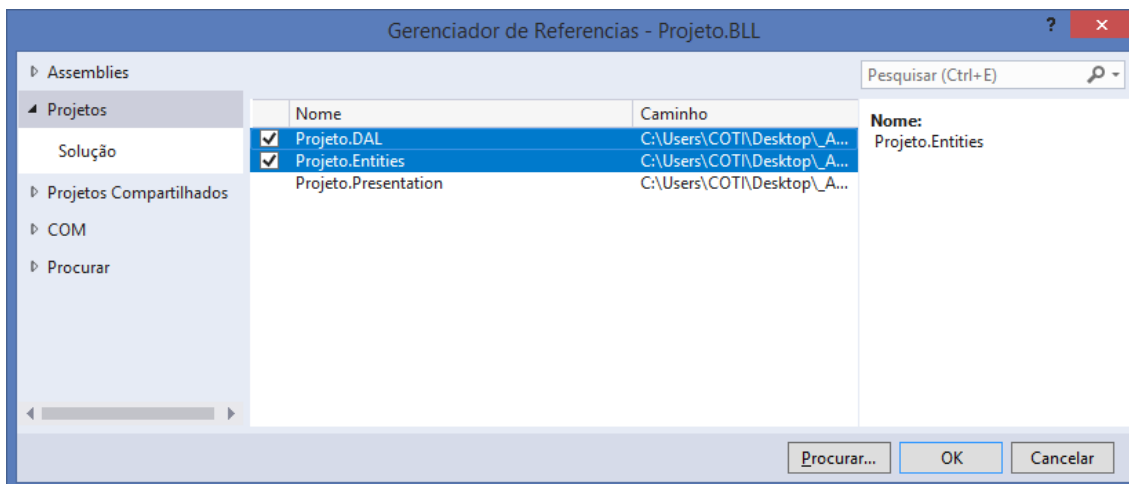
//método para retornar produtos pelo nome
public List<Produto> FindByName(string nome)
{
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        string query = "SELECT * FROM PRODUTO WHERE NOME LIKE @Nome";

        return conn.Query<Produto>(query,
            new { Nome = $"%{nome}%" })
            .ToList();
    }
}
}
```

## 1.2 - Camada de Regras de Negócio: Biblioteca de Classes .NET Framework



## Adicionando referencias:



[Basic 3-Tire architecture]

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

namespace Projeto.BLL
{
```

```
public class ProdutoBusiness
{
    public void CadastrarProduto(Produto produto)
    {
        ProdutoRepository repository = new ProdutoRepository();
        repository.Insert(produto);
    }

    public void AtualizarProduto(Produto produto)
    {
        ProdutoRepository repository = new ProdutoRepository();
        repository.Update(produto);
    }

    public void ExcluirProduto(int idProduto)
    {
        ProdutoRepository repository = new ProdutoRepository();
        repository.Delete(idProduto);
    }

    public List<Produto> ConsultarTodos()
    {
        ProdutoRepository repository = new ProdutoRepository();
        return repository.FindAll();
    }

    public Produto ConsultarPorId(int idProduto)
    {
        ProdutoRepository repository = new ProdutoRepository();
        return repository.FindById(idProduto);
    }

    public List<Produto> ConsultarPorNome(string nome)
    {
        ProdutoRepository repository = new ProdutoRepository();
        return repository.FindByName(nome);
    }
}
```

## Adicionando referencias no projeto Presentation

