Arquitetura do Projeto

Baseado em 3 camadas

* Data Access Layer (DAL)

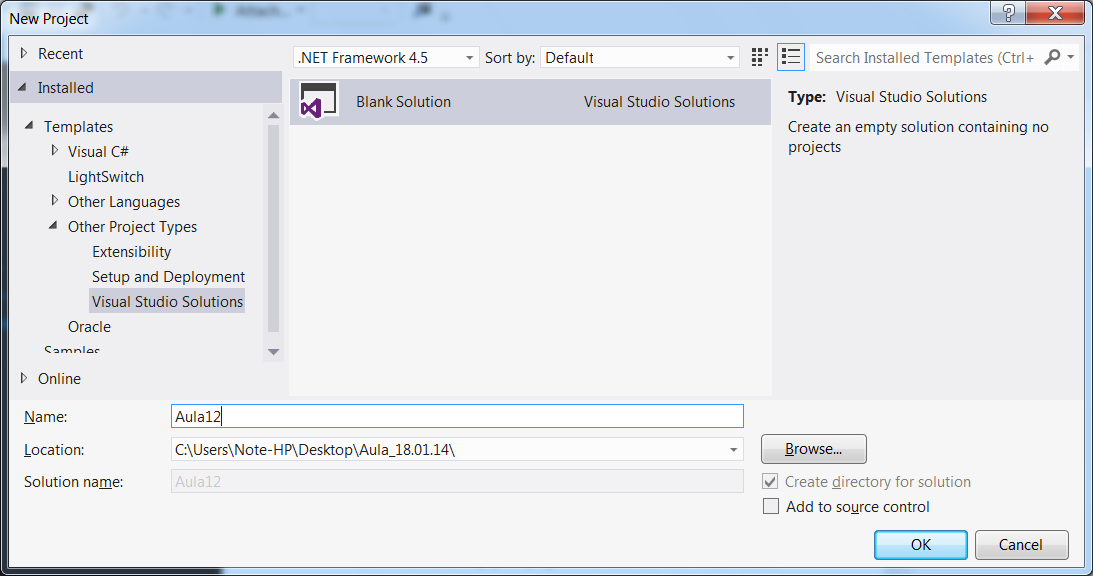
Camada para acesso a base de dados

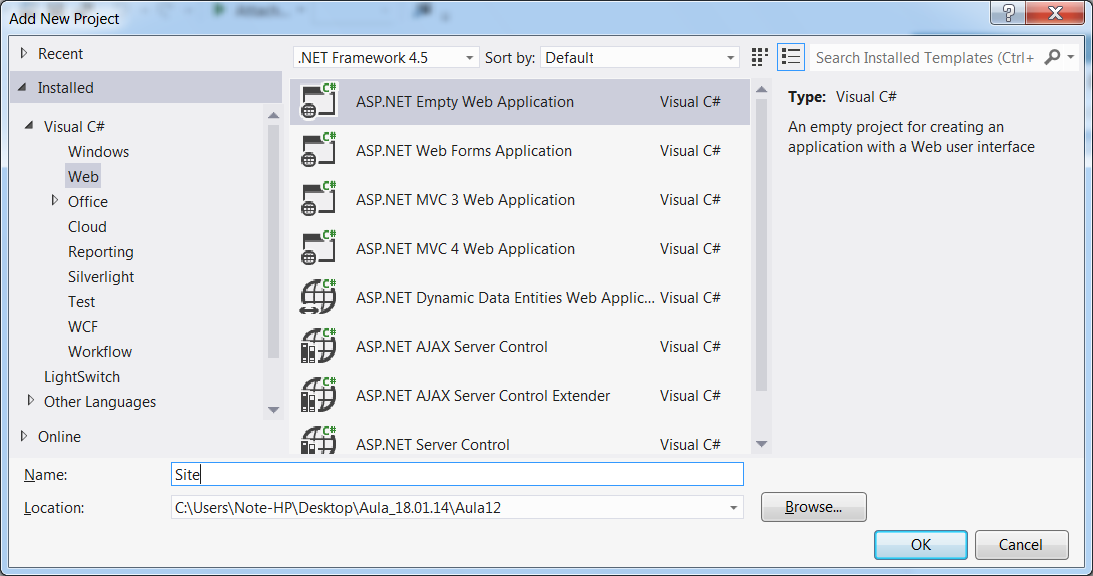
* Business Logic Layer (BLL)

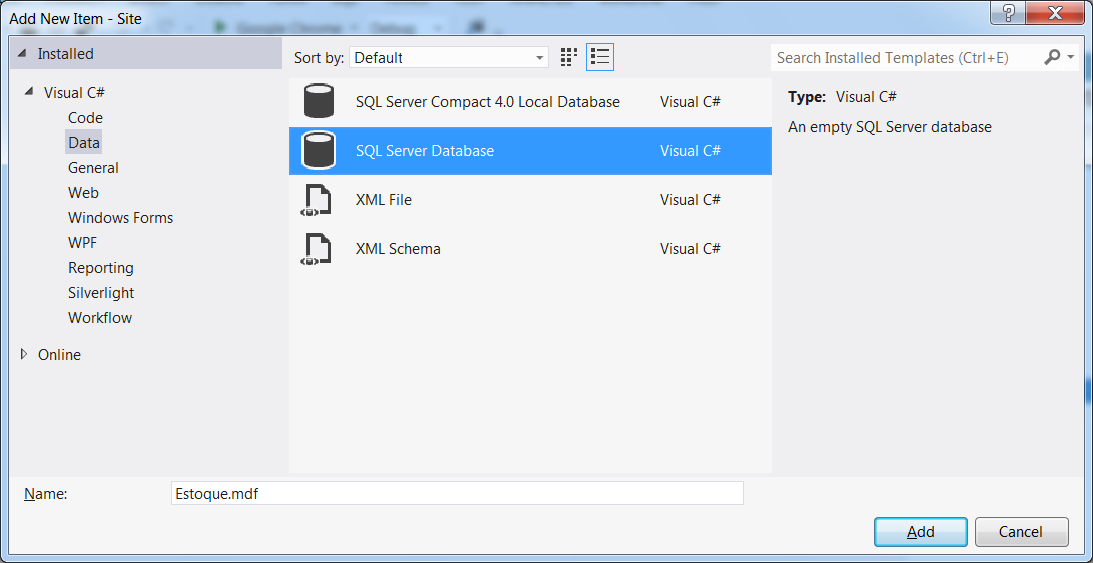
Camada de regras de negócio

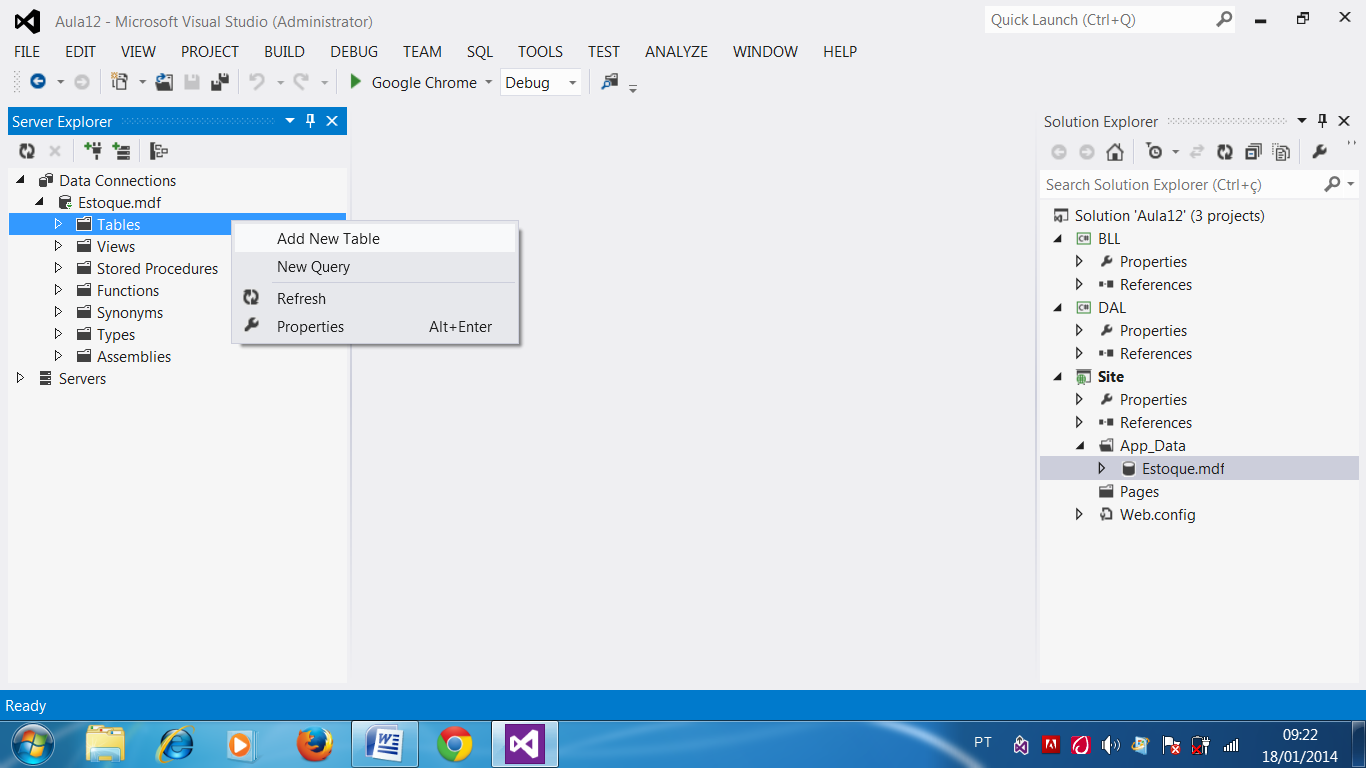
* Presentation Layer

Projeto Web (Interface do Sistema com o usuario)









go

create table Fornecedor(

IdFornecedor integer identity,

Nome nvarchar(50) not null,

primary key(IdFornecedor))

go

create table Produto(

IdProduto integer identity,

Nome nvarchar(50) not null,

Preco float not null,

Quantidade integer not null,

IdFornecedor integer not null,

primary key(IdProduto),

foreign key(IdFornecedor)

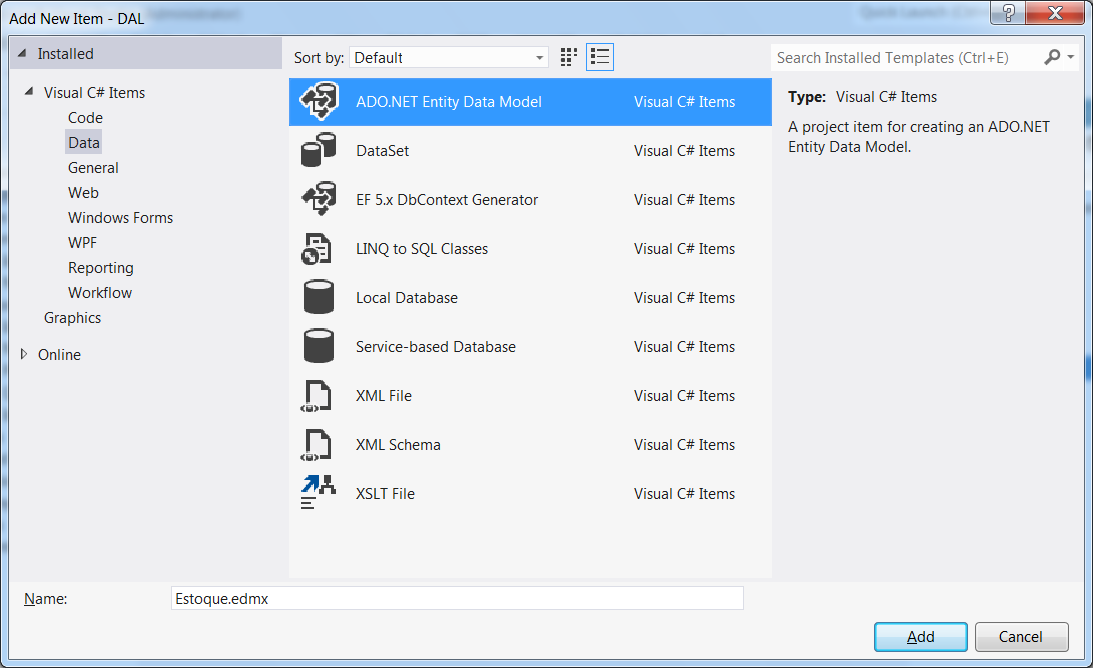
references Fornecedor(IdFornecedor))

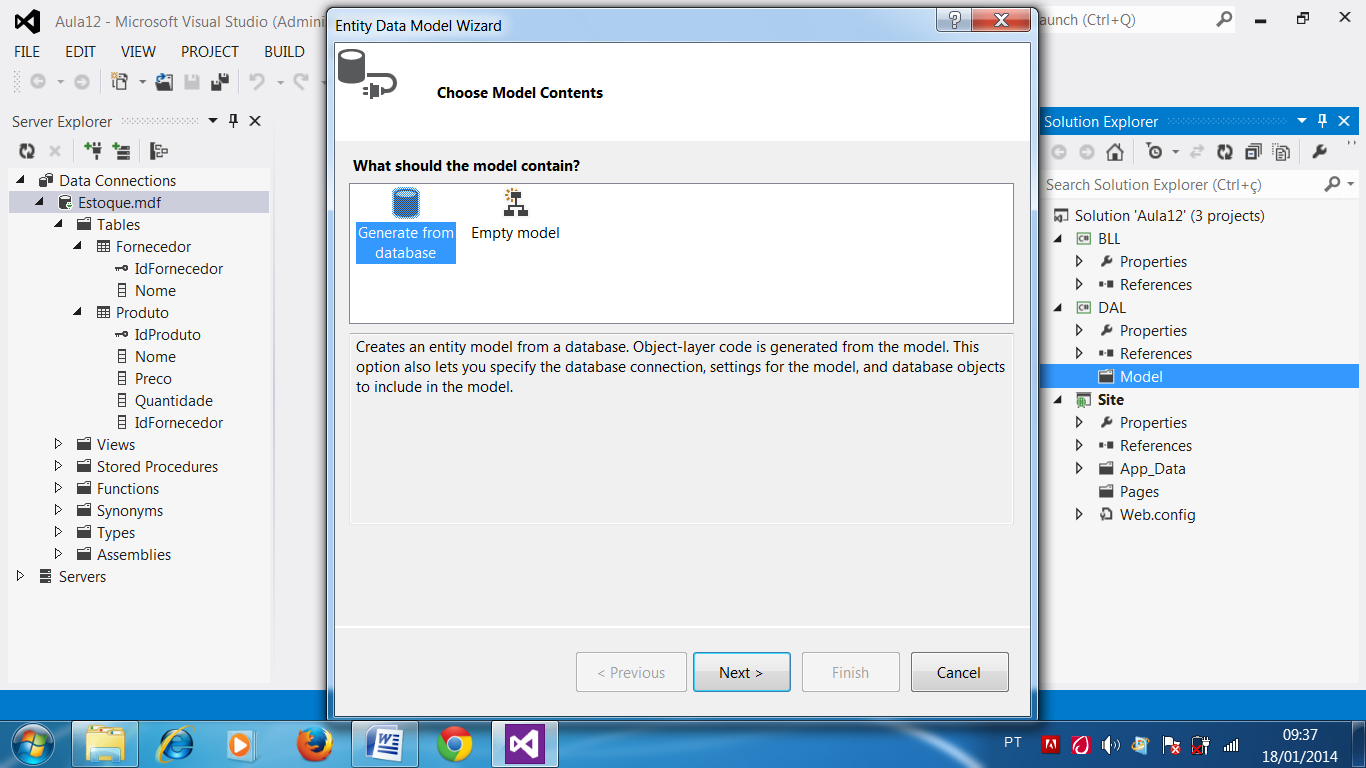


EntityFramework

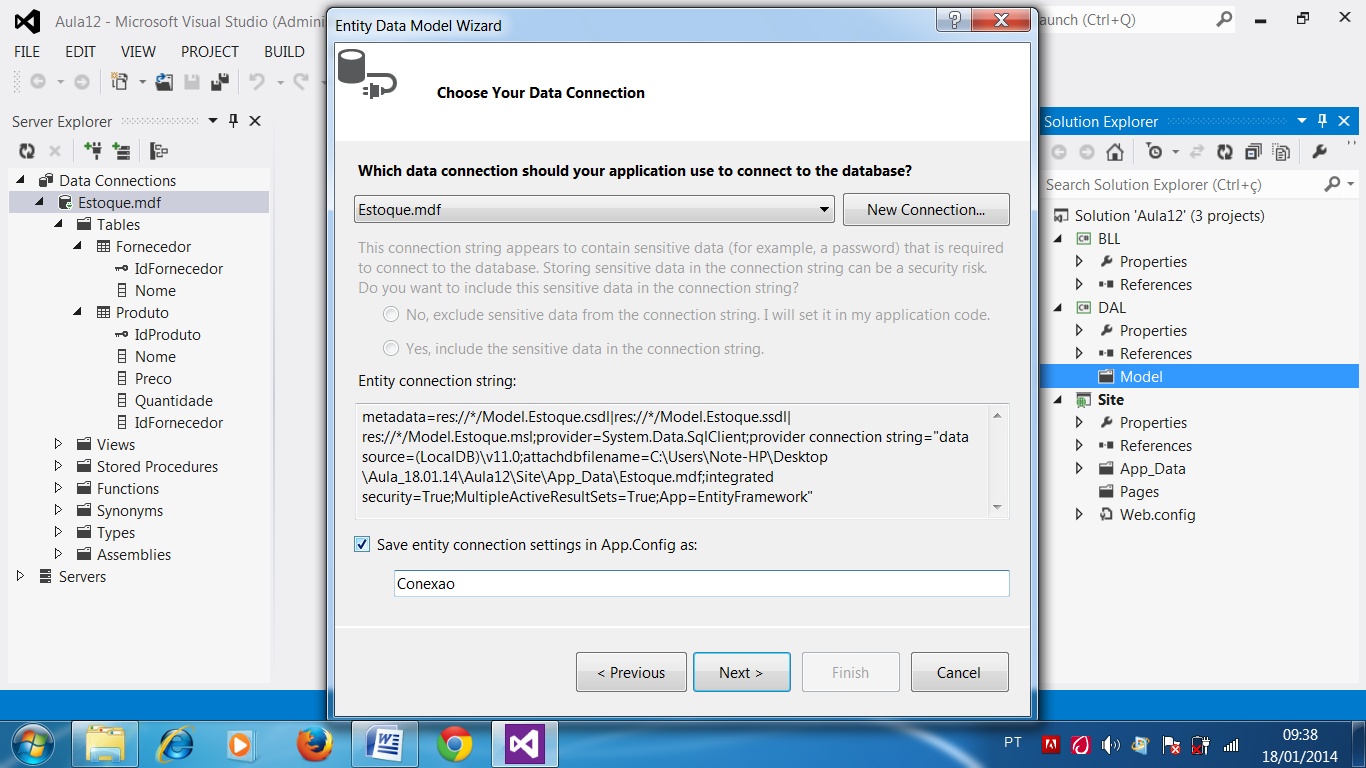
**.edmx** (Entity Data Model)

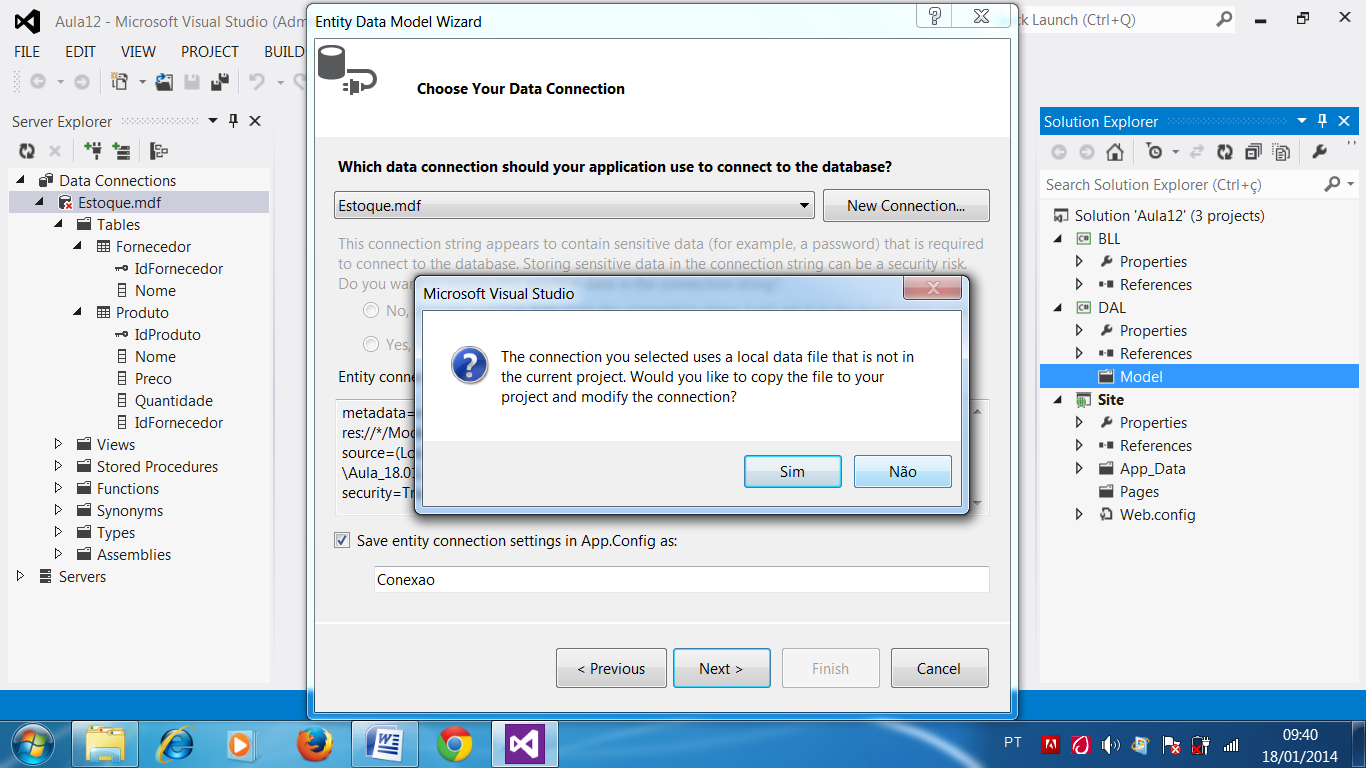
* ModelFirst
* DatabaseFirst (Engenharia reversa)

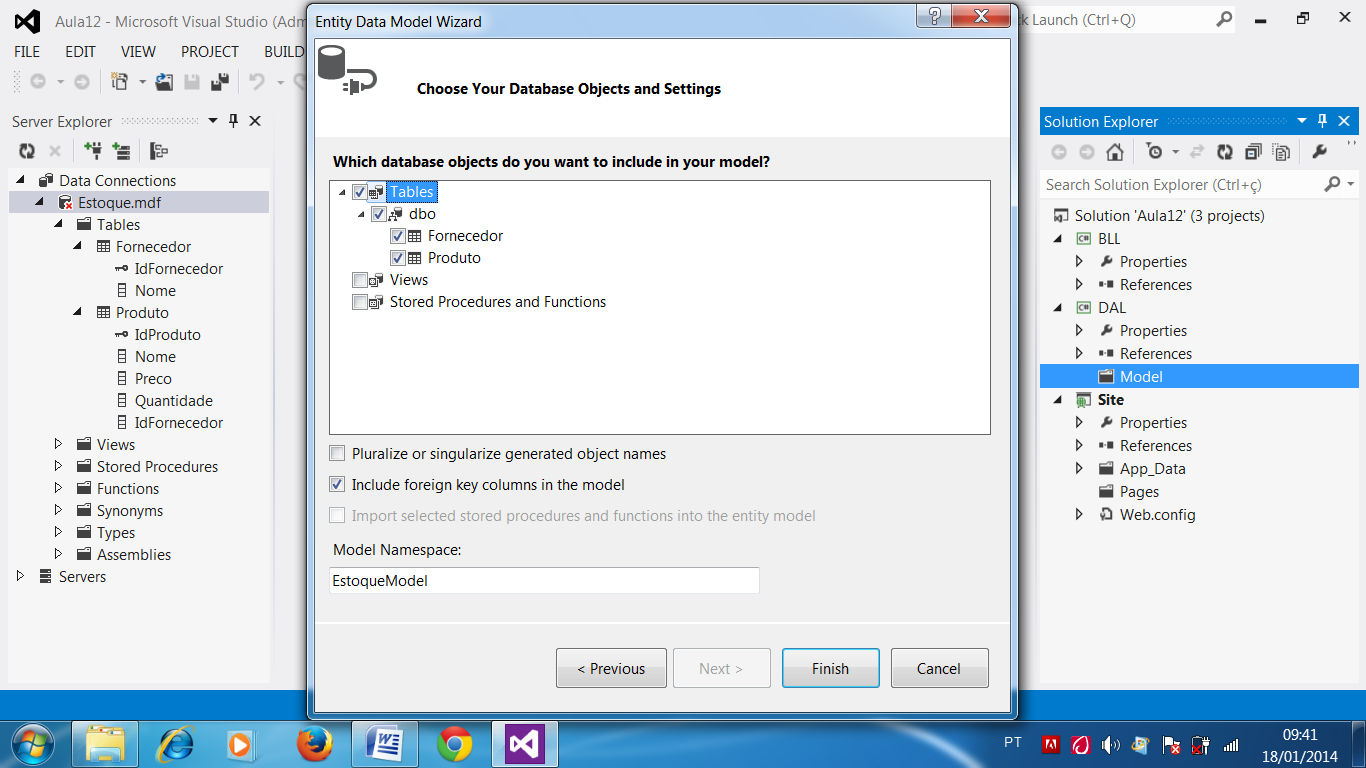


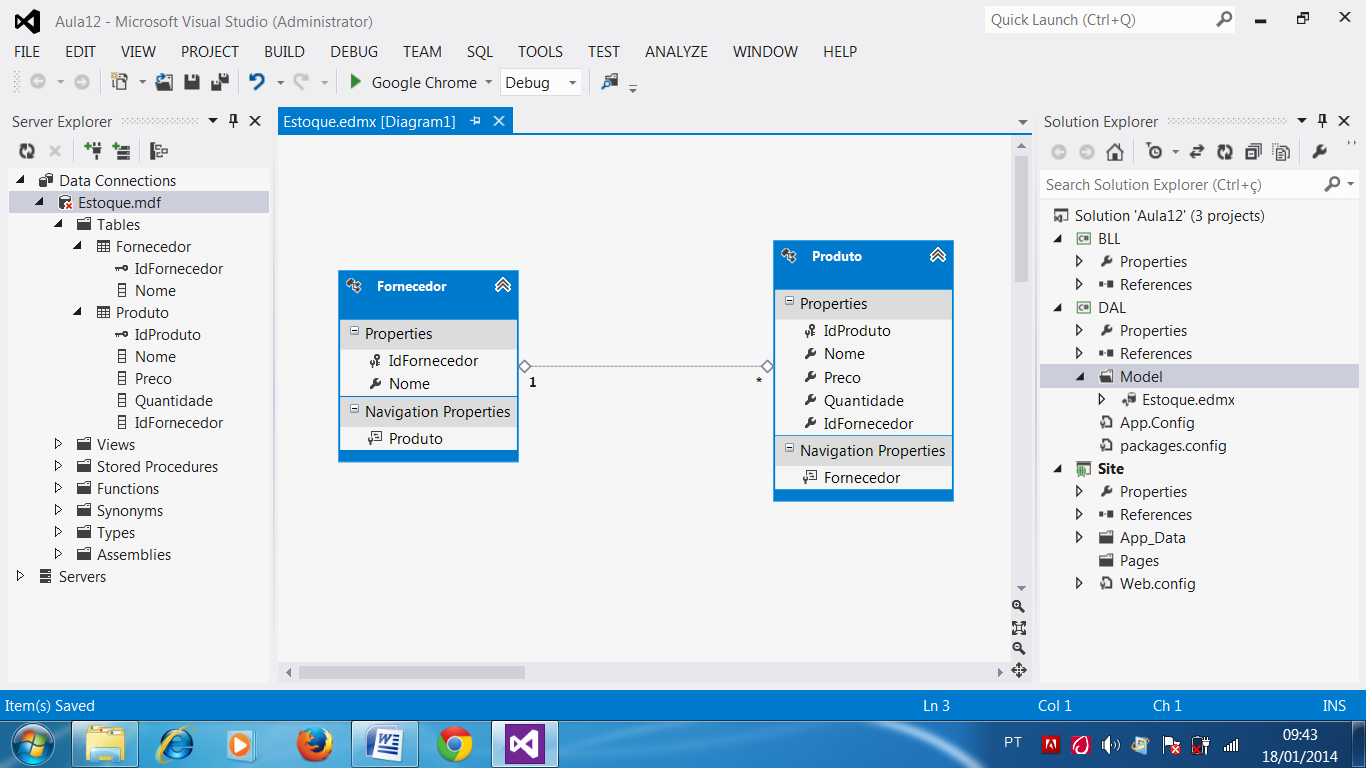


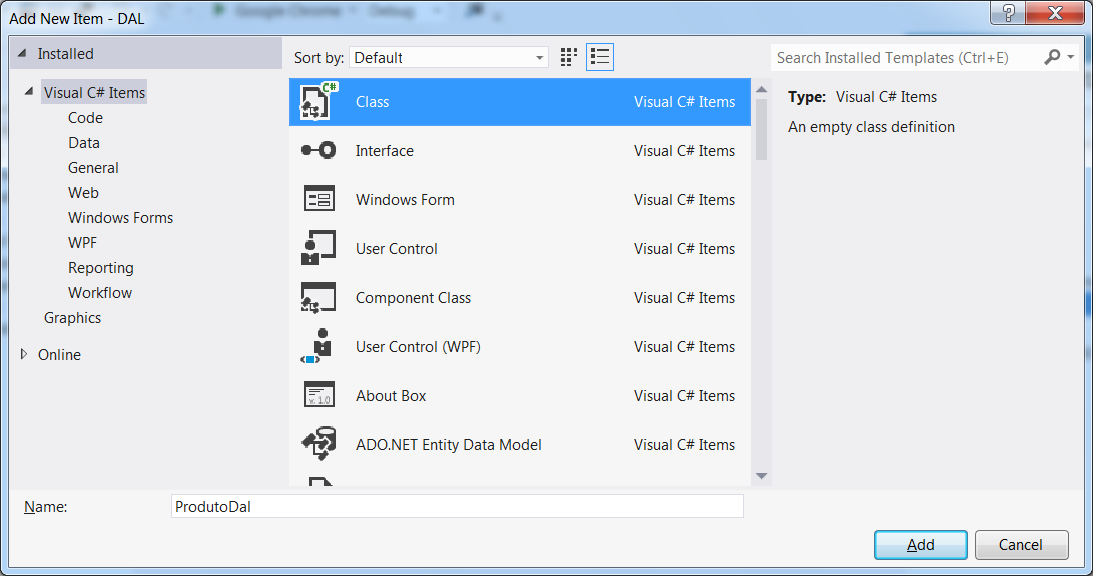
Nome da Classe de Conexão com a base de dados que será gerada pelo Entity Data Model

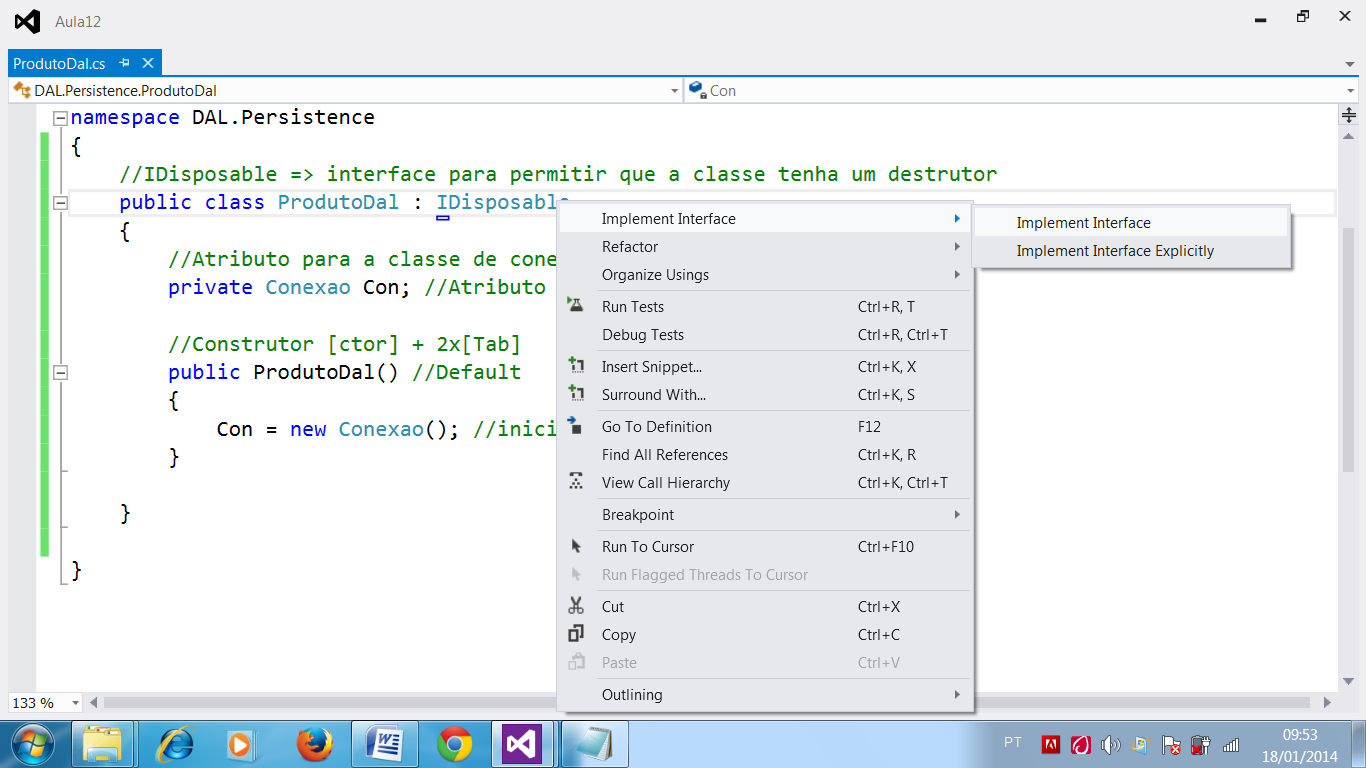












using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using DAL.Model;

namespace DAL.Persistence

{

//IDisposable => interface para permitir que a classe tenha um destrutor

public class ProdutoDal : IDisposable

{

//Atributo para a classe de conexão com o banco de dados

private Conexao Con; //Atributo (null)

//Construtor [ctor] + 2x[Tab]

public ProdutoDal() //Default

{

Con = new Conexao(); //inicializando o atributo

}

//Método para listar todos os Produtos da base de dados

public List<Produto> ListarTodos()

{

try

{

//SQL => select \* from Produto

return Con.Produto.ToList();

}

catch(Exception e)

{

throw new Exception("Erro ao listar produtos: " + e.Message);

}

}

public void Dispose()

{

//liberar o atributo Conexão de memória

Con.Dispose();

}

}

}

String de Conexão...

<?xml version="1.0"?>

<configuration>

<connectionStrings>

<add name="Conexao" connectionString="metadata=res://\*/Model.Estoque.csdl|res://\*/Model.Estoque.ssdl|res://\*/Model.Estoque.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=(LocalDB)\v11.0;attachdbfilename=C:\Users\Note-HP\Desktop\Aula\_18.01.14\Aula12\Site\App\_Data\Estoque.mdf;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

</connectionStrings>

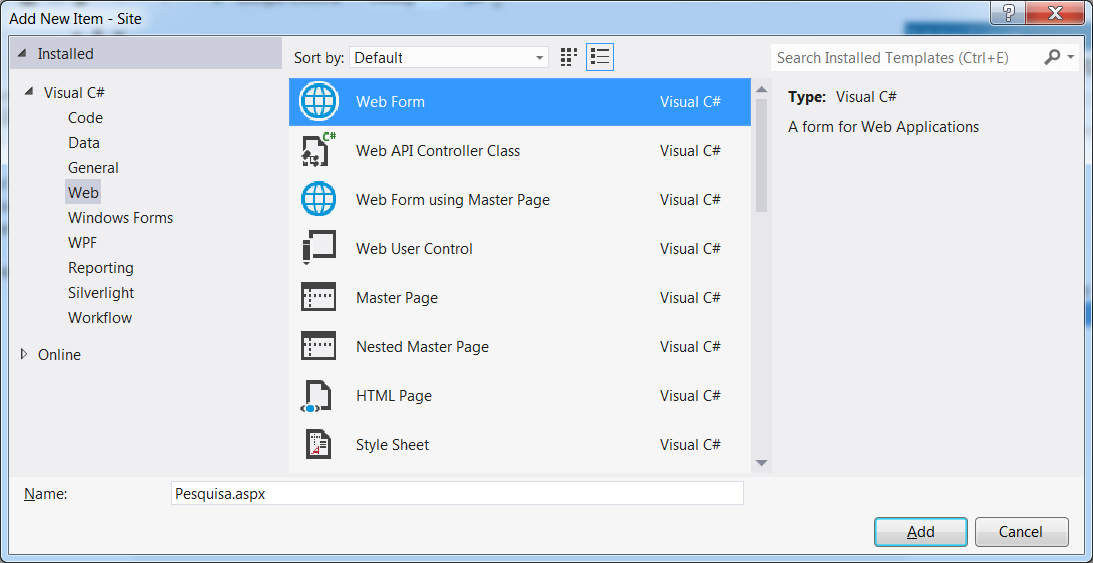
<system.web>

<compilation debug="true" targetFramework="4.5" />

<httpRuntime targetFramework="4.5" />

</system.web>

</configuration>



<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Pesquisa.aspx.cs" Inherits="Site.Pages.Pesquisa" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

</head>

<body>

<form id="form1" runat="server">

<div>

<h3>Pesquisa de Produtos</h3>

<hr />

<p>

Relação de Produtos em Estoque:

</p>

<asp:GridView ID="gridEstoque" runat="server" Width="100%">

</asp:GridView>

<p>

Selecione o Formato do Relatório:

<asp:DropDownList ID="ddlFormato" runat="server">

<asp:ListItem Value="0" Text="- Escolha uma Opção -" />

<asp:ListItem Value="1" Text="PDF" />

<asp:ListItem Value="2" Text="Word" />

<asp:ListItem Value="3" Text="Excel" />

</asp:DropDownList>

<asp:Button ID="btnDownload" runat="server" Text="Gerar Relatório" />

</p>

<p>

<asp:Label ID="lblMensagem" runat="server" />

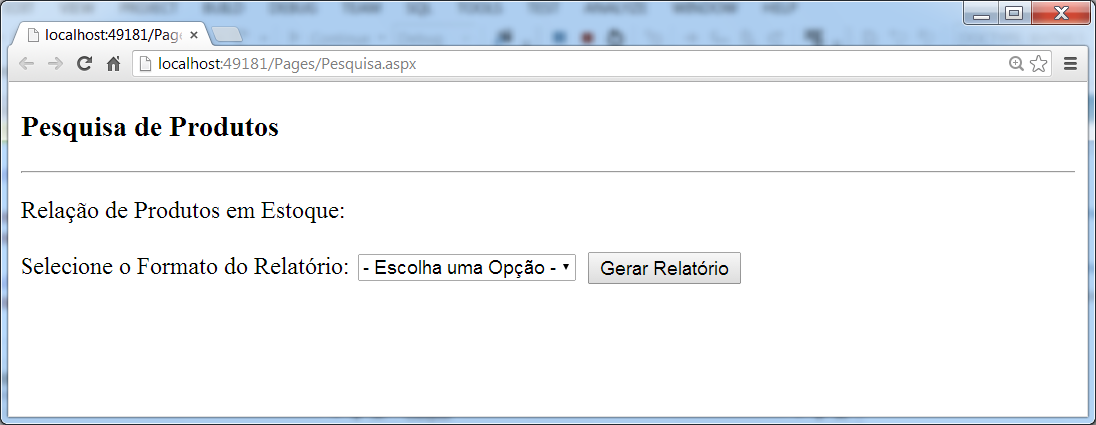
</p>

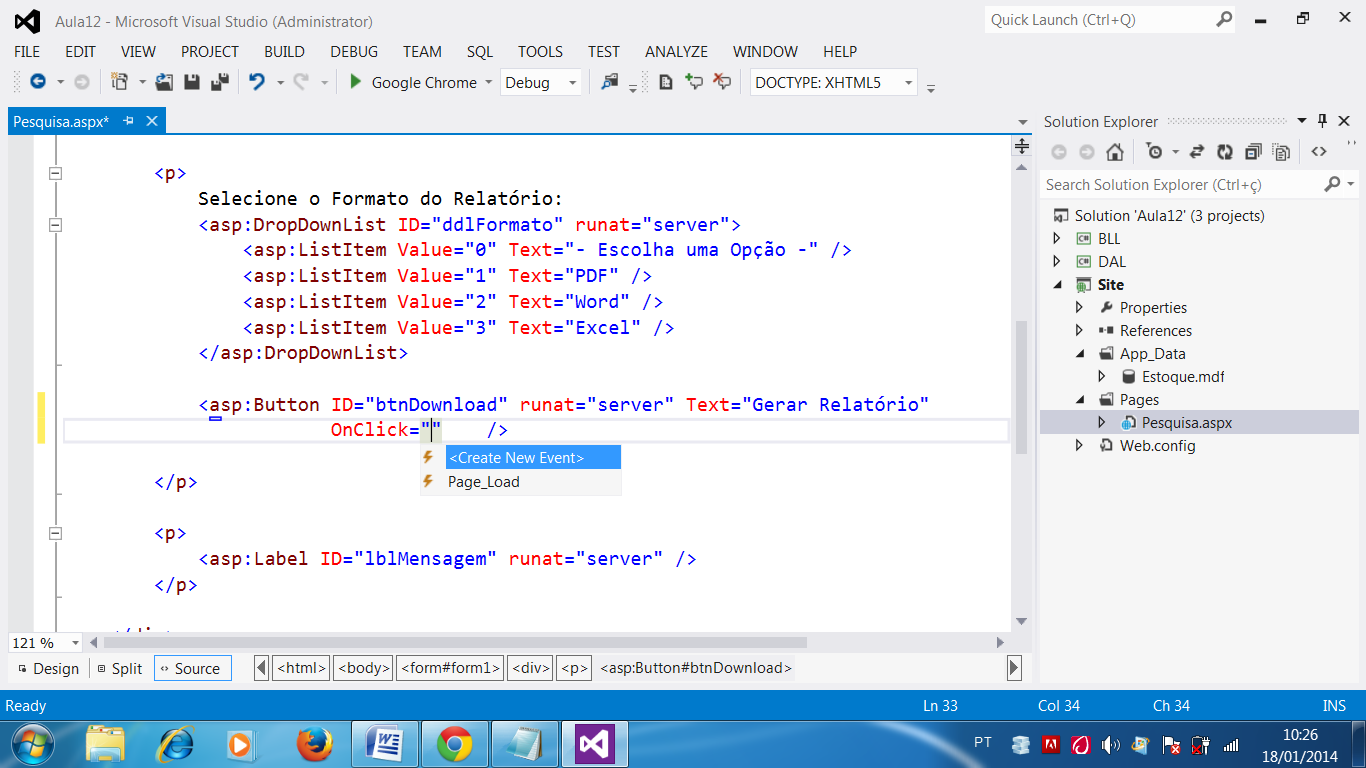
</div>

</form>

</body>

</html>





**<asp:Button ID="btnDownload" runat="server" Text="Gerar Relatório"**

**OnClick="btnDownload\_Click" />**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace Site.Pages

{

public partial class Pesquisa : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void btnDownload\_Click(object sender, EventArgs e)

{

}

}

}

Business Logic Layer

Camada de Regras de Negócio

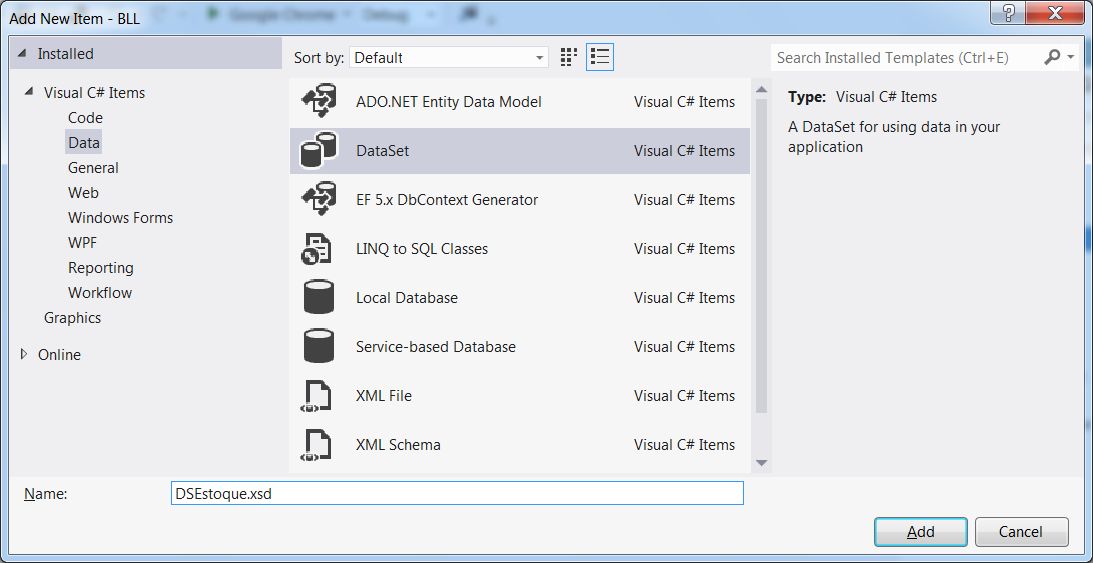
* **Modelagem de dados de negócio**
  + RelatorioEstoque
    - Codigo Id do Produto
    - Produto Nome do Produto
    - Fornecedor Nome do Fornecedor
    - Preco Preco
    - Quantidade Quantidade
    - Total Preco \* Quantidade

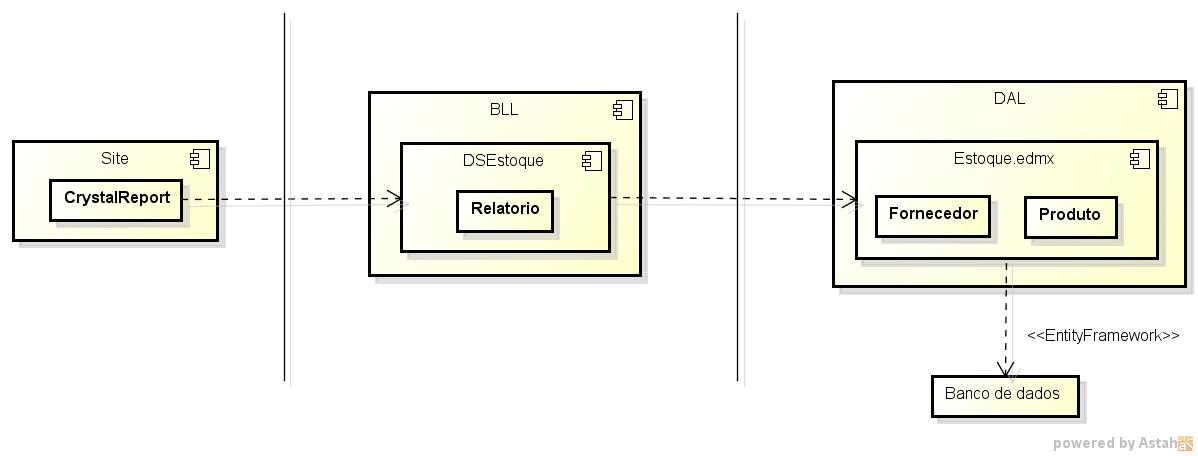
DataSet

Ferramenta para modelagem de dados que, em versões anteriores do .NET, era utilizado para modelagem de bases de dados, como no EntityFramework.

Hoje, podemos utilizar os DataSets para modelagem de negocio, ou seja, criar neles entidades que sirvam à regras de negocio da aplicação.

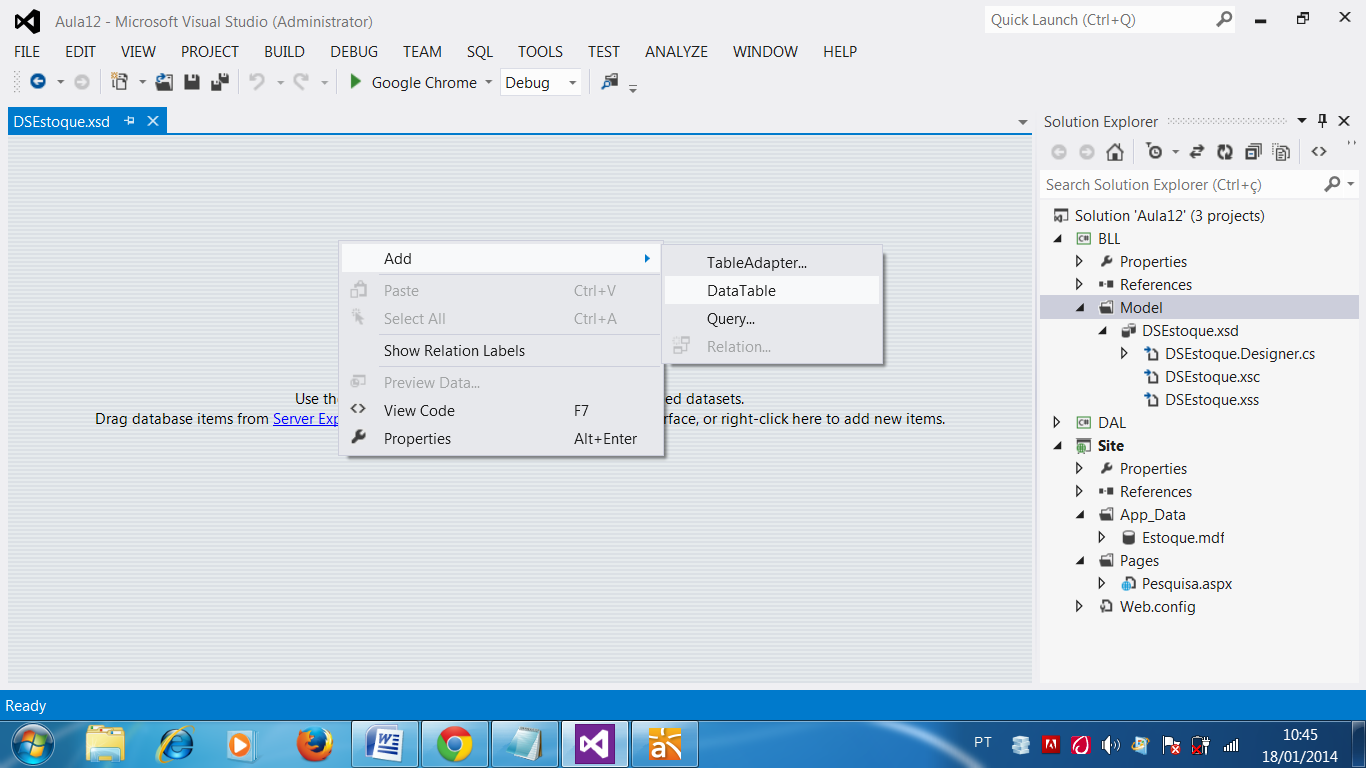


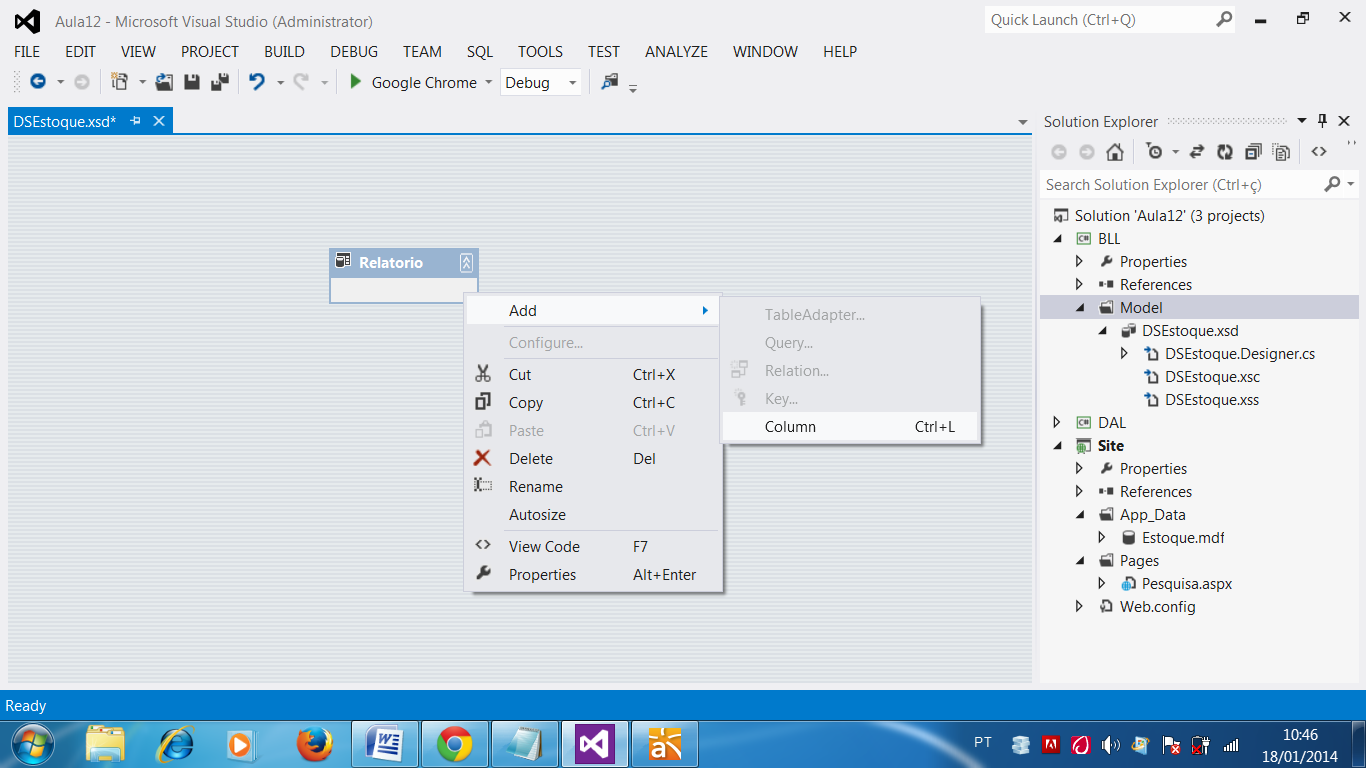




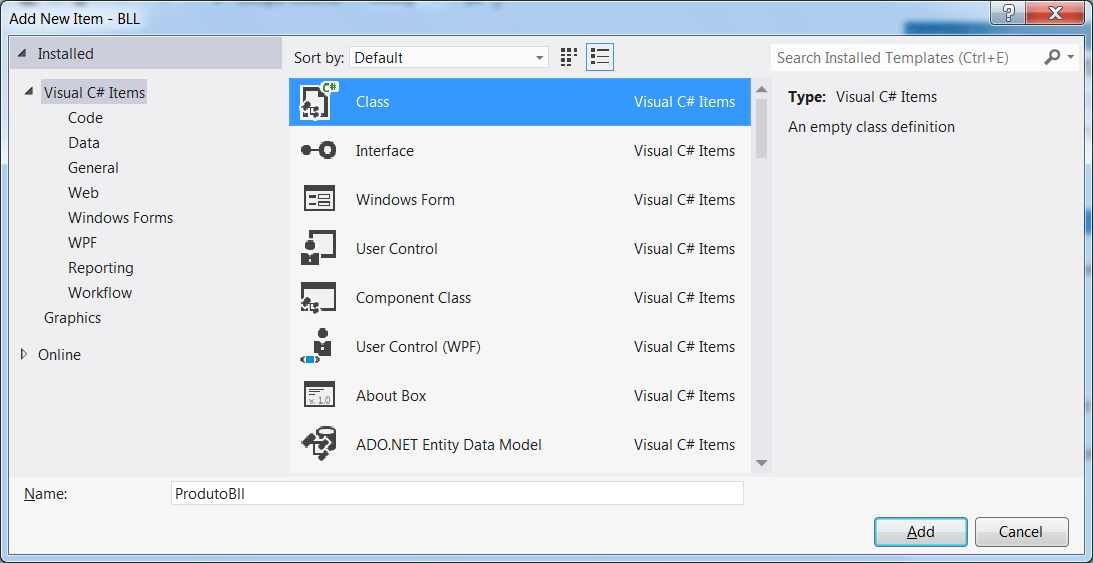
DataTables

Tabelas dentro de um DataSet utilizadas para representação de entidades de dados.









using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data; //DataSet

using DAL.Model;

using DAL.Persistence;

namespace BLL.Business

{

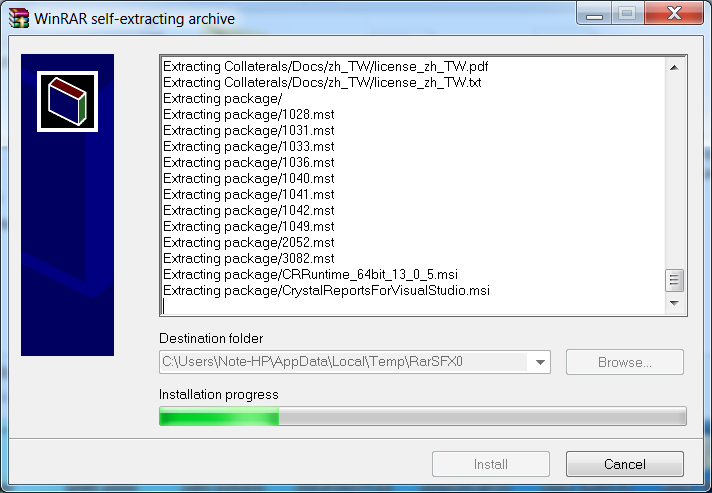
public class ProdutoBll

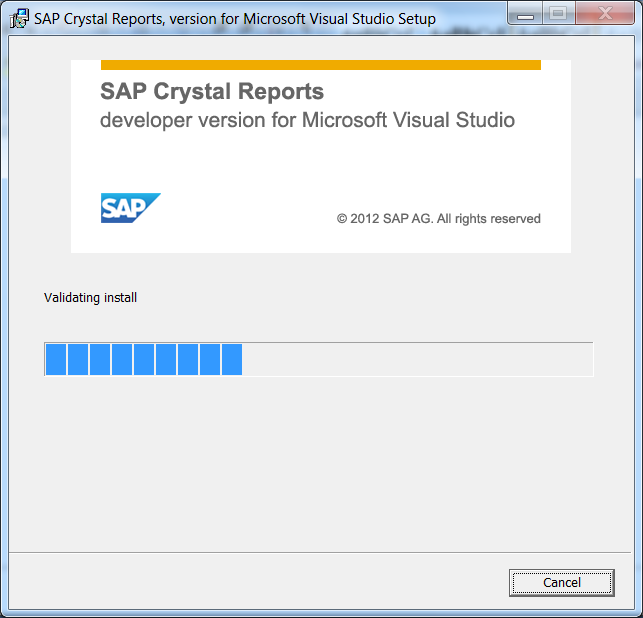
{

}

}

Instalando o CrystalReports





Observação...

Namepsace

System.Data

**DataSet**

Representa a Modelagem, que contem as DataTables

**DataTable**

Representa uma entidade contida em um dataset, como uma tabela, por exemplo.

**DataRow**

Representa uma linha, ou seja, um registro dentro da DataTable

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data; //DataSet

using DAL.Model;

using DAL.Persistence;

using BLL.Model;

namespace BLL.Business

{

public class ProdutoBll

{

//Método para gerar a DataTable 'Relatorio' populada com os dados da base

public DataTable ObterDadosRelatorio()

{

try

{

using(ProdutoDal d = new ProdutoDal())

{

//Executar a consulta e retornar todos os registros do banco

var dados = d.ListarTodos();

DSEstoque ds = new DSEstoque();

//instanciando o DataSet (Conjunto de DataTables)

DataTable dt = ds.Tables["Relatorio"];

//instanciando a DataTable

//Percorrer cada registro contido na variavel

//dados (List<Produto>)

foreach(var item in dados)

//foreach(Produto item in dados)

{

//DataRow => criar cada registro dentro da DataTable

DataRow registro = dt.NewRow();

registro["Codigo"] = item.IdProduto;

registro["Produto"] = item.Nome;

registro["Fornecedor"] = item.Fornecedor.Nome;

registro["Preco"] = item.Preco;

registro["Quantidade"] = item.Quantidade;

registro["Total"] = item.Preco \* item.Quantidade;

//Gravar o registro (DataRow) na DataTable

dt.Rows.Add(registro);

}

//retornar a DataTable

return dt;

}

}

catch(Exception e)

{

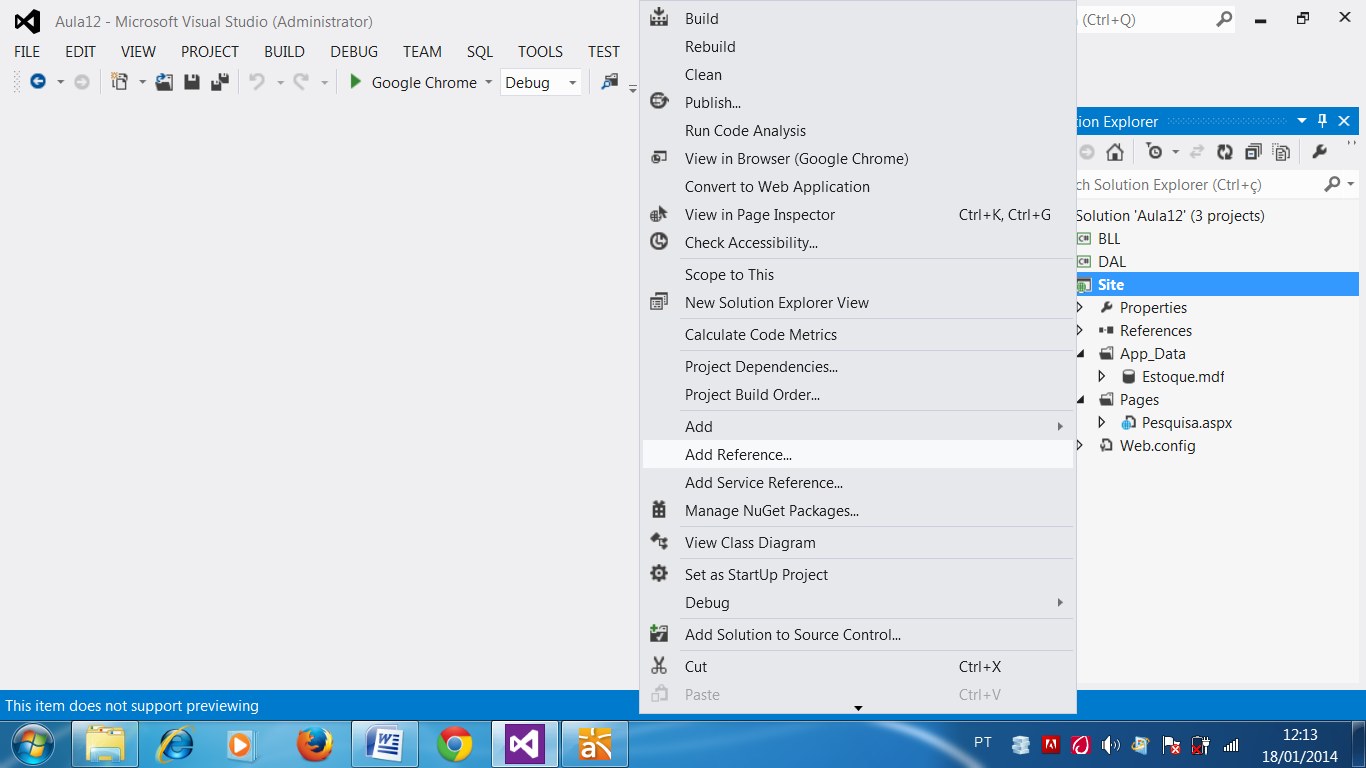
throw new Exception("Erro ao obter dados: " + e.Message);

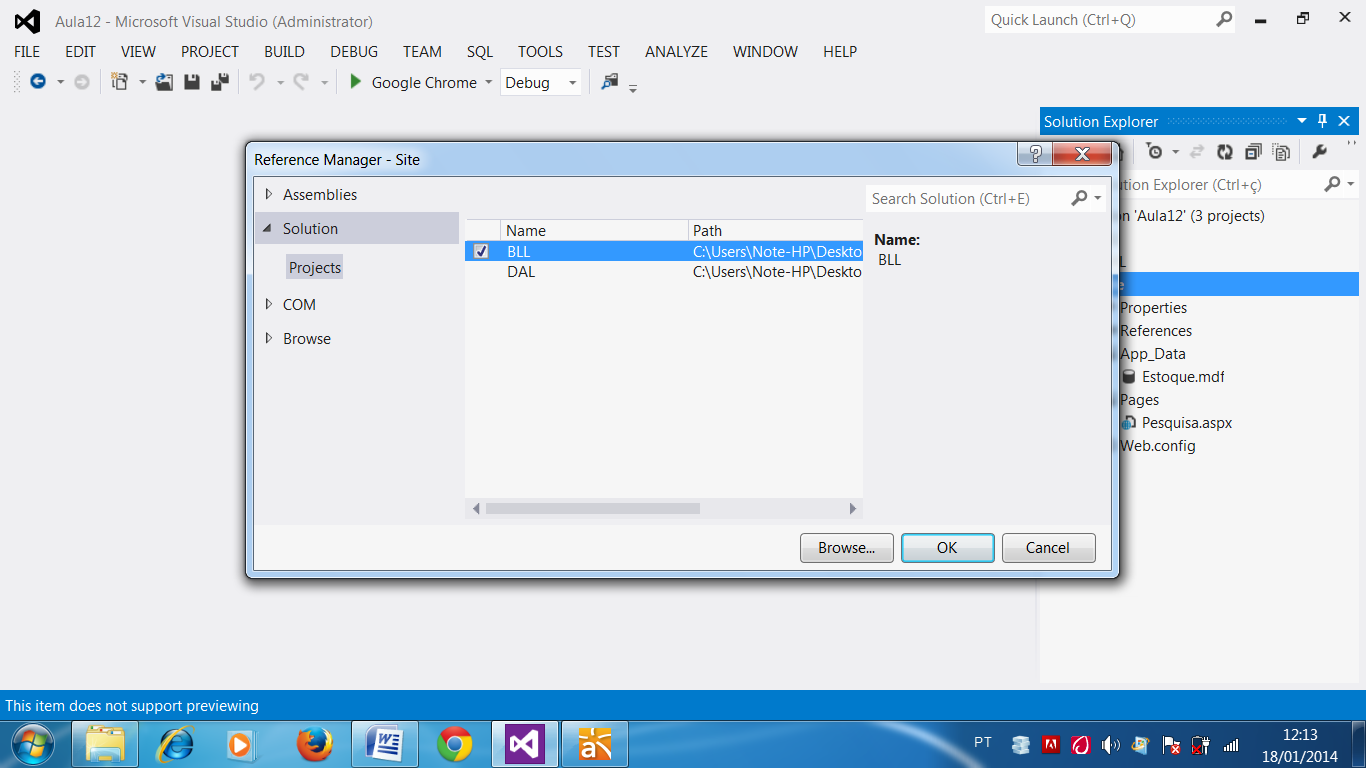
}

}

}

}





using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using BLL.Business;

namespace Site.Pages

{

public partial class Pesquisa : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if( ! IsPostBack )

{

try

{

ProdutoBll p = new ProdutoBll();

gridEstoque.DataSource = p.ObterDadosRelatorio(); //Popular

gridEstoque.DataBind(); //Exibir

}

catch(Exception ex)

{

lblMensagem.Text = ex.Message;

}

}

}

protected void btnDownload\_Click(object sender, EventArgs e)

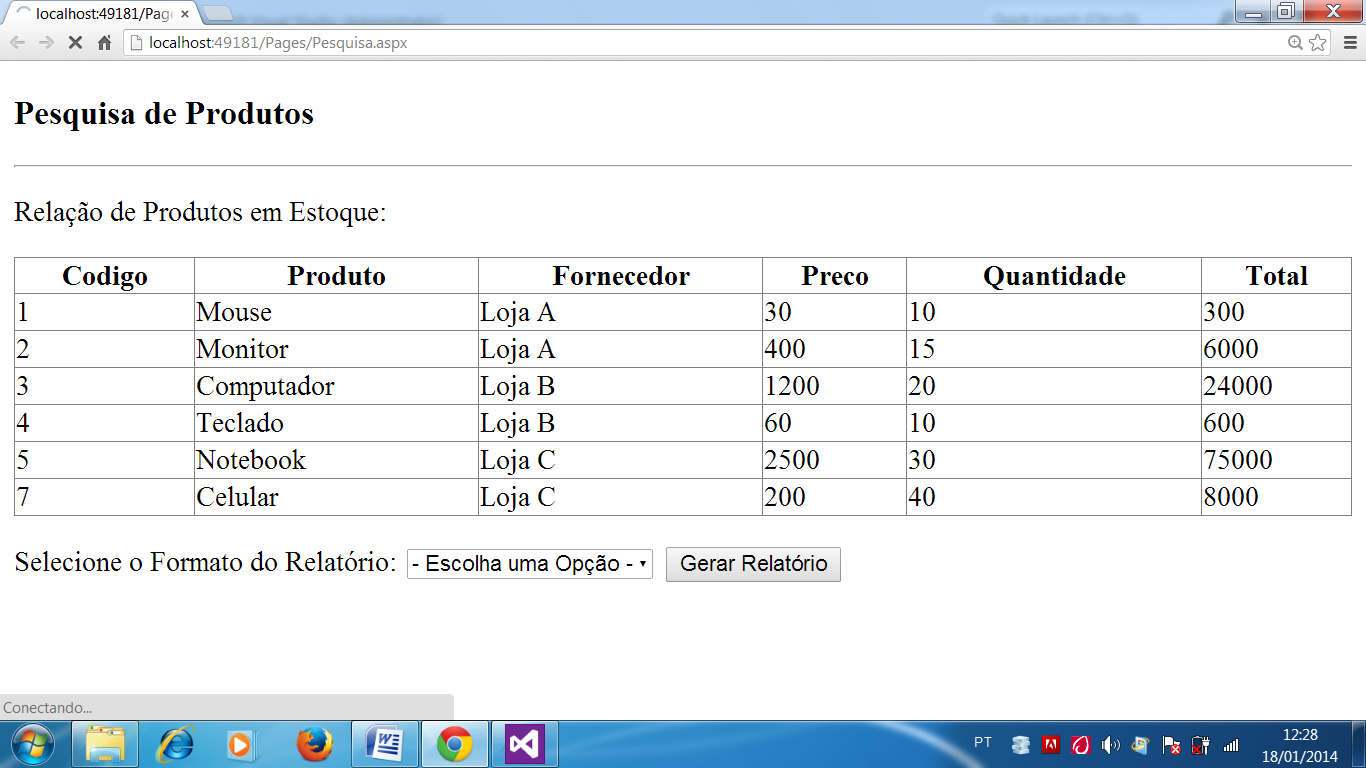
{

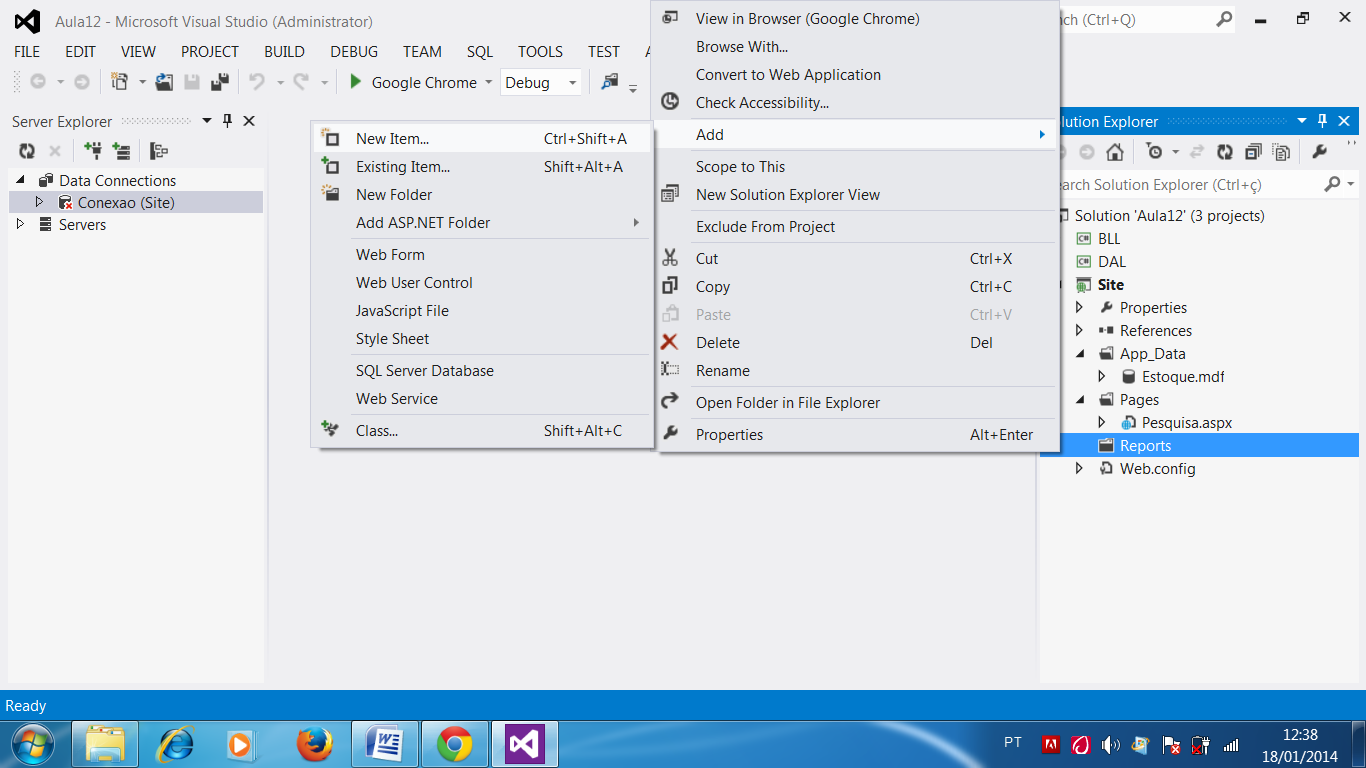
}

}

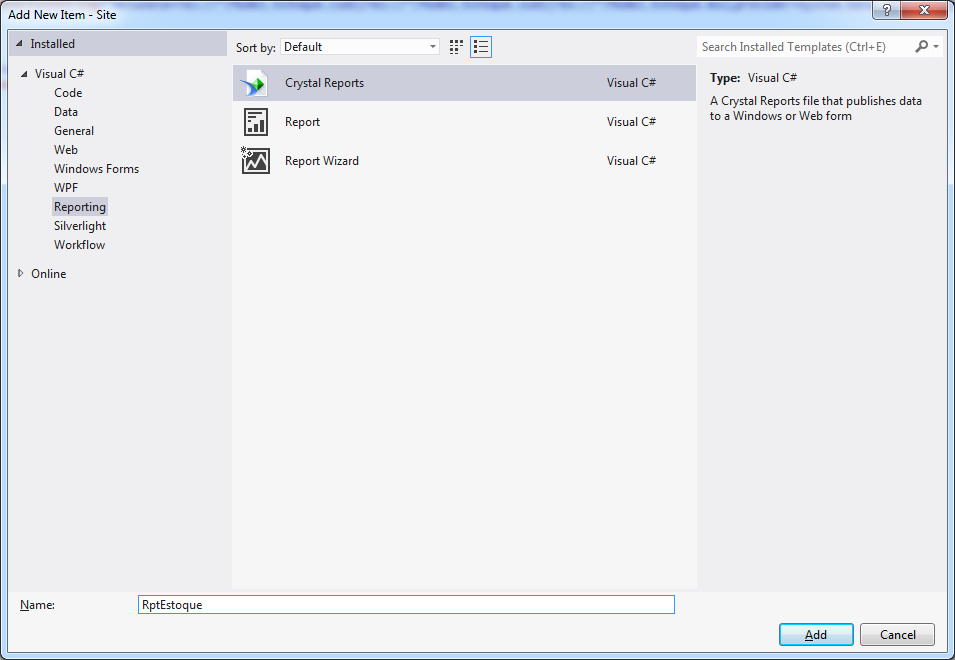
}

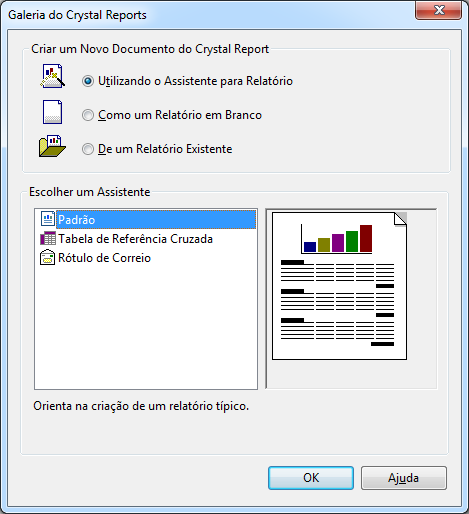
Resultado...

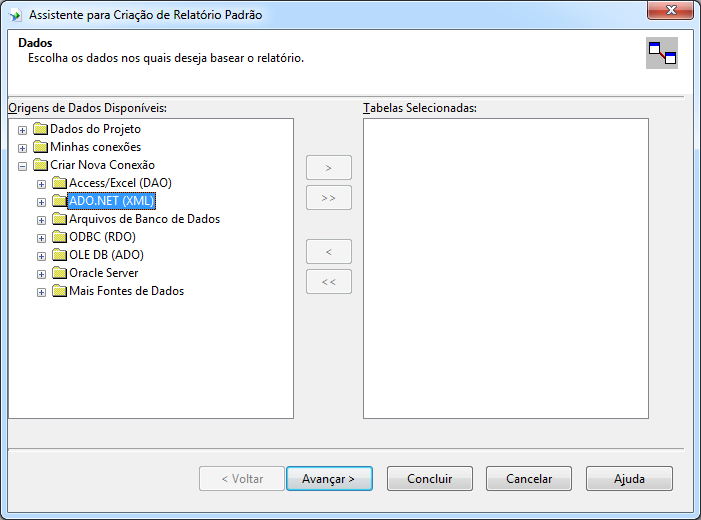
****

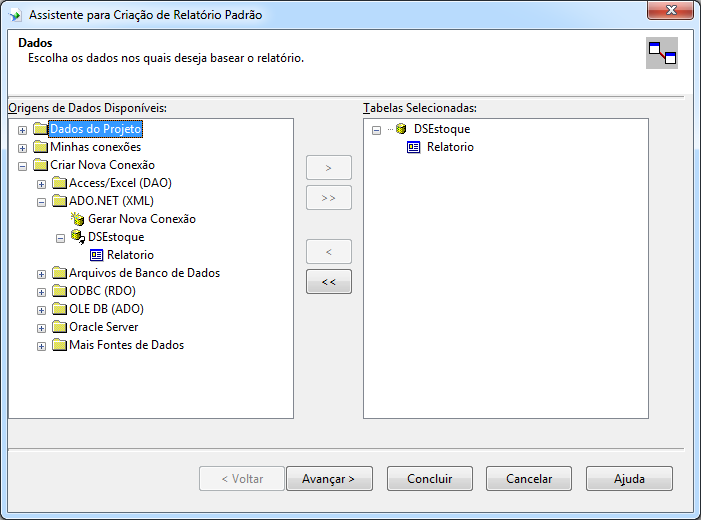
****

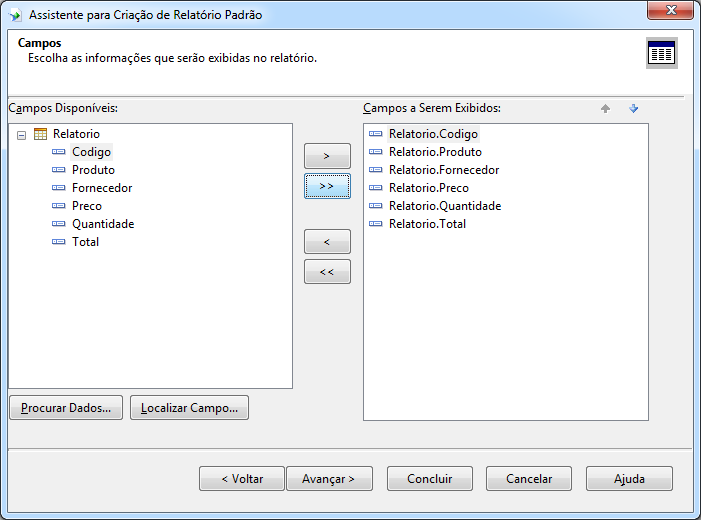
Criando o Relatório através do CrystalReports

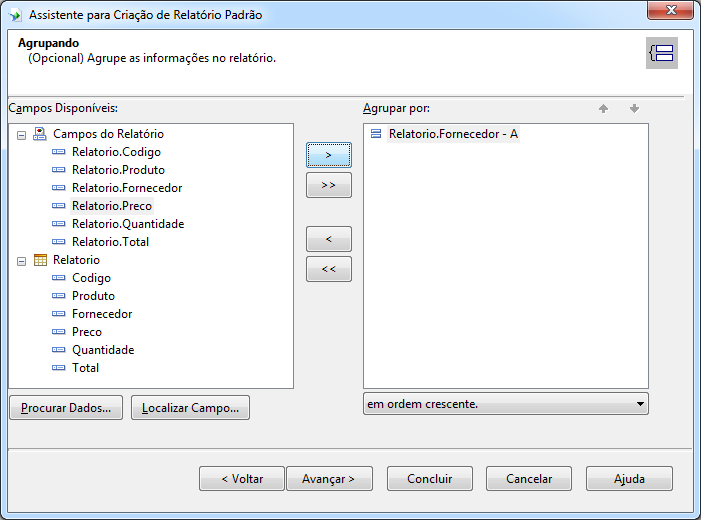
****

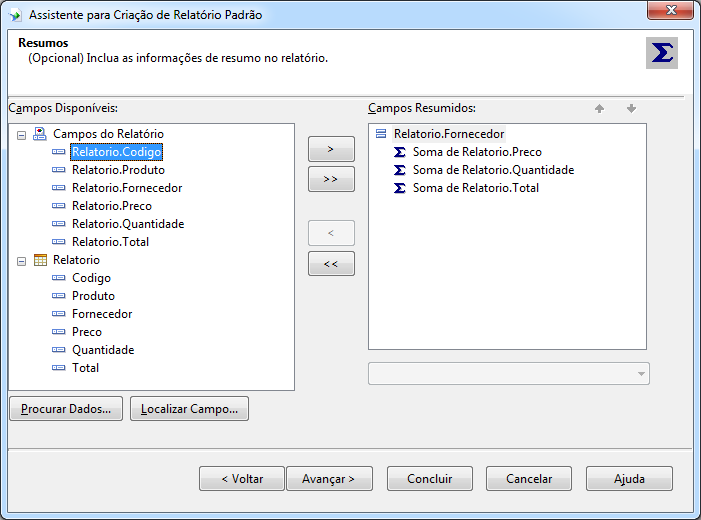
****

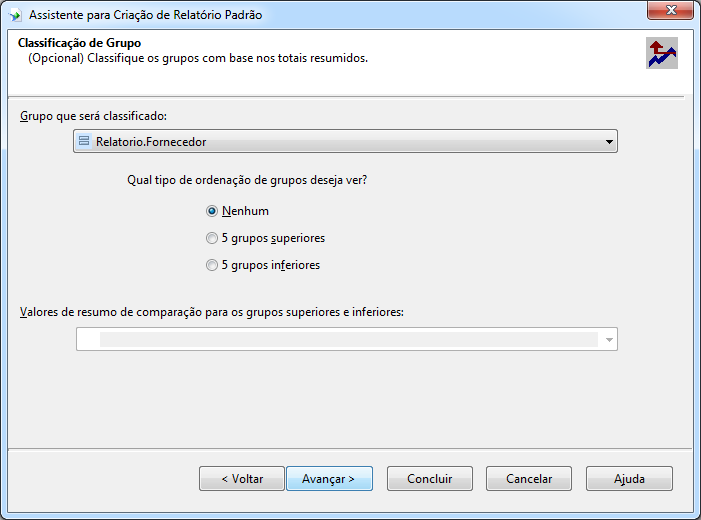
****

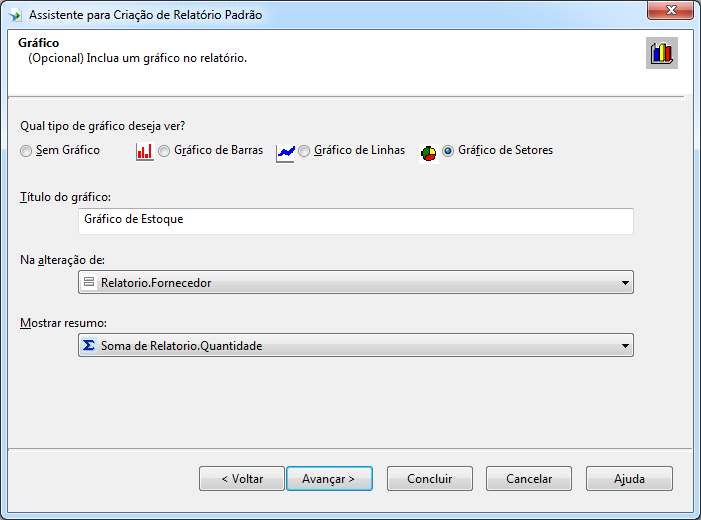
****

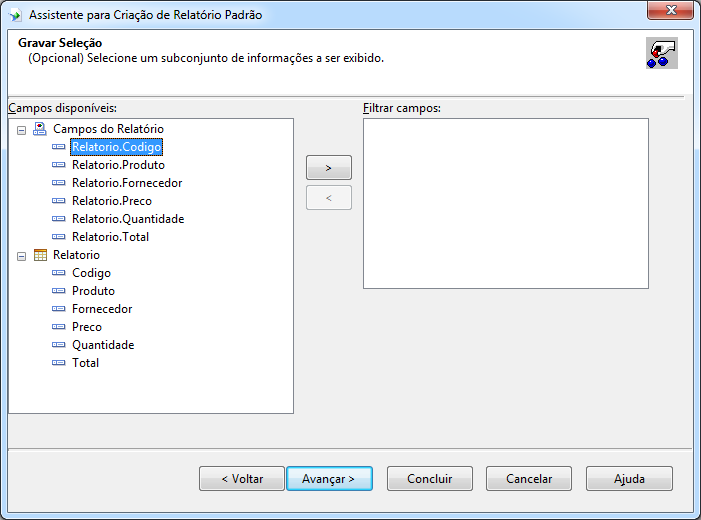
****

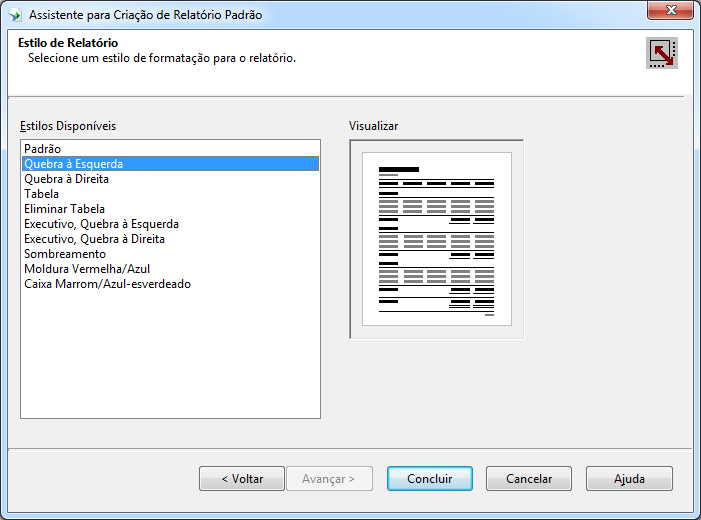
****

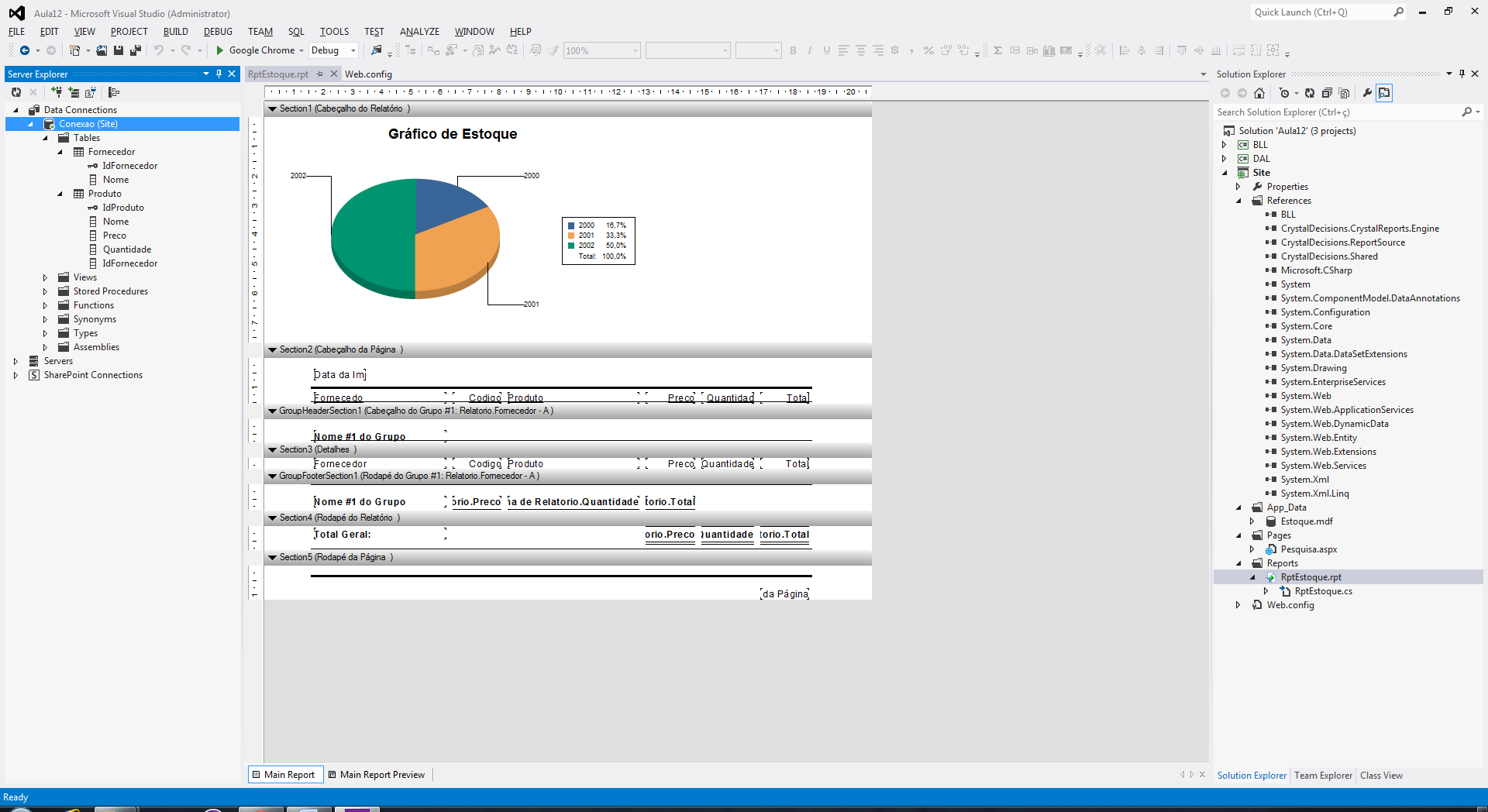
****

****

****

****

****



Criando o evento para realizar o Download do Relatório...

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using BLL.Business;

using Site.Reports;

using CrystalDecisions.CrystalReports.Engine;

using CrystalDecisions.Shared;

namespace Site.Pages

{

public partial class Pesquisa : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if( ! IsPostBack )

{

try

{

ProdutoBll p = new ProdutoBll();

gridEstoque.DataSource = p.ObterDadosRelatorio(); //Popular

gridEstoque.DataBind(); //Exibir

}

catch(Exception ex)

{

lblMensagem.Text = ex.Message;

}

}

}

protected void btnDownload\_Click(object sender, EventArgs e)

{

try

{

ProdutoBll pb = new ProdutoBll();

RptEstoque es = new RptEstoque();

es.SetDataSource(pb.ObterDadosRelatorio());

switch(ddlFormato.SelectedValue)

{

case "1":

es.ExportToHttpResponse(ExportFormatType.

PortableDocFormat, Response, true, "relatorio");

break;

case "2":

es.ExportToHttpResponse(ExportFormatType.

WordForWindows, Response, true, "relatorio");

break;

case "3":

es.ExportToHttpResponse(ExportFormatType.

Excel, Response, true, "relatorio");

break;

}

}

catch(Exception ex)

{

lblMensagem.Text = ex.Message;

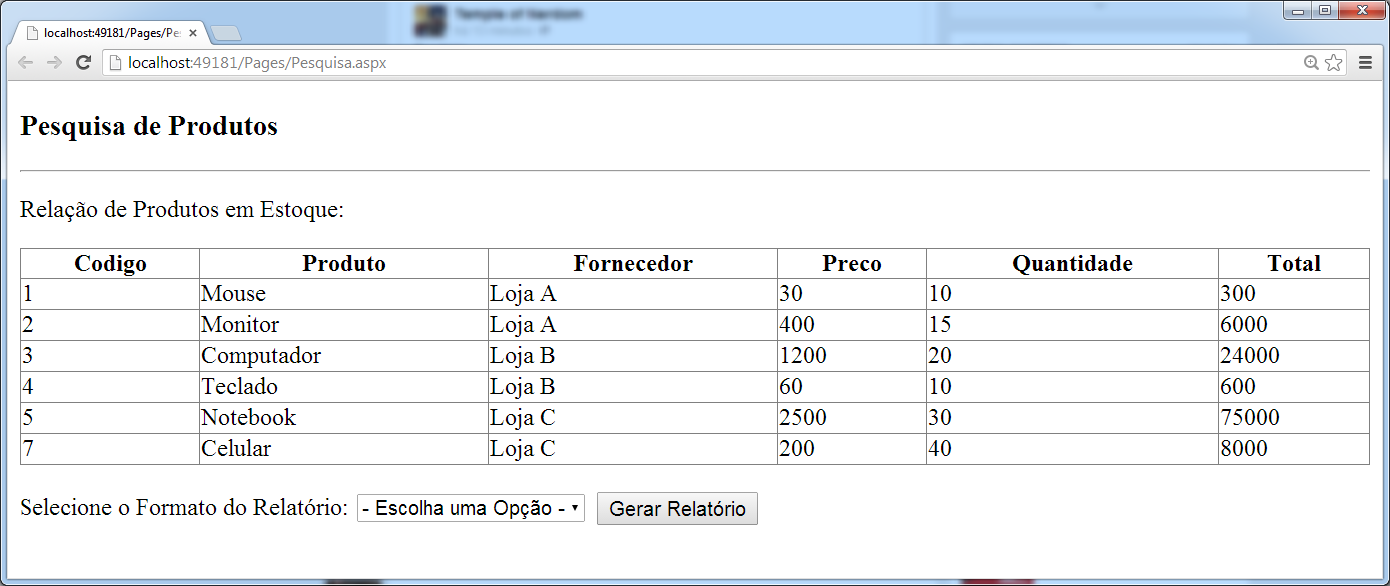
}

}

}

}

Executando...



Relatório...



