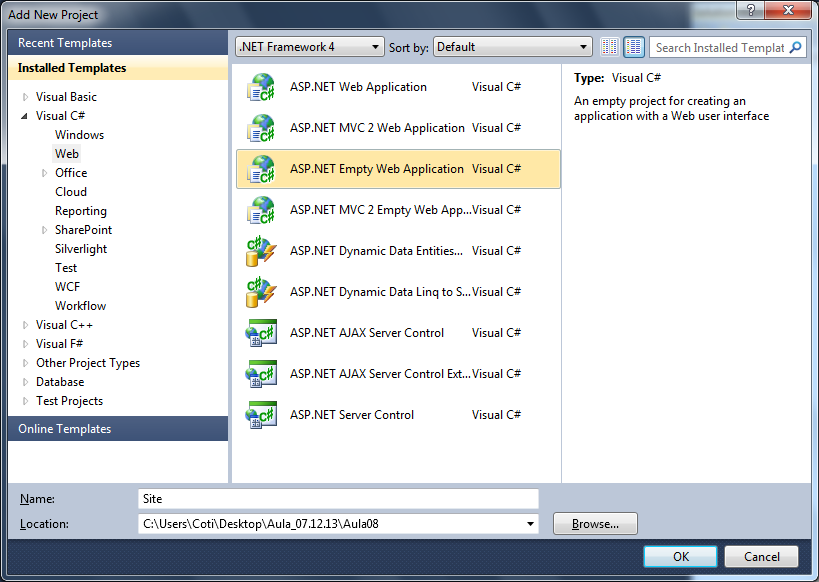
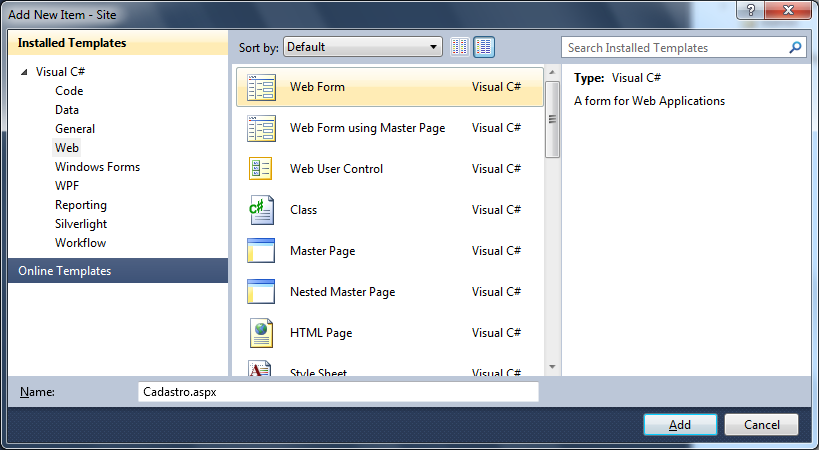


Criando o Projeto Asp.Net



Criando a página para cadastro de Usuários...



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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<h3>Cadastro de Usuários</h3>

<a href="Login.aspx">Acessar Sistema</a>

<hr />

Login de Acesso: <br />

<asp:TextBox ID="txtLogin" runat="server" />

<br /><br />

Senha: <br />

<asp:TextBox ID="txtSenha" runat="server" TextMode="Password" />

<br /><br />

Confirme sua Senha: <br />

<asp:TextBox ID="txtSenhaConfirm" runat="server" TextMode="Password" />

<br /><br />

<asp:Button ID="btnCadastro" runat="server" Text="Cadastrar Usuário" />

<p>

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

</p>

</div>

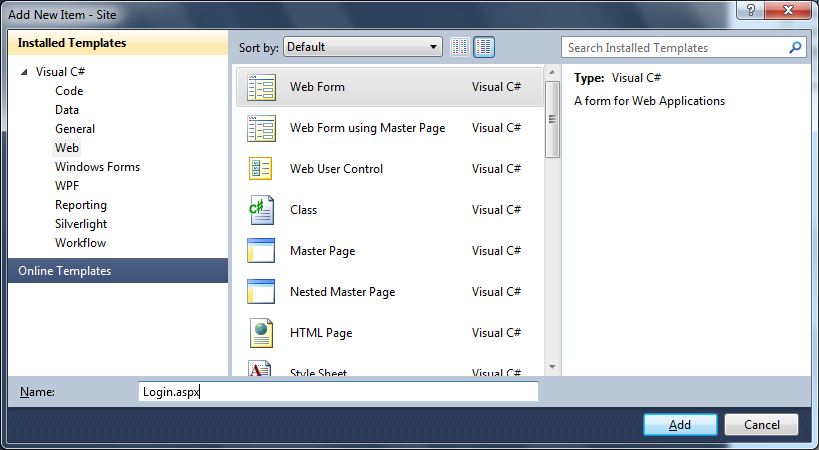
</form>

</body>

</html>



Criando a página de Login de Usuários



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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<h3>Login de Usuários</h3>

<a href="Cadastro.aspx">Criar Conta</a>

<hr />

<!-- Componente customizado para login -->

**<asp:Login**

**ID="formLogin"**

**runat="server"**

**OnAuthenticate="AutenticarUsuario"**

**/>**

</div>

</form>

</body>

</html>

O Acesso a uma área restrita de um site Asp.Net WebForms é concedida mediante a apresentação de um Ticket através do qual definiremos que o usuário foi reconhecido ou identificado pela aplicação.

AuthenticateEventArgs

Classe derivada de EventArgs que é utilizada para suporte a eventos de autenticação em sites Asp.Net

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 Login : System.Web.UI.Page

{

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

{

}

protected void AutenticarUsuario(object sender, **AuthenticateEventArgs** e)

{

//Negando o Ticket de Acesso

e.Authenticated = false;

}

}

}

Customizando o campo asp:Login

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<h3>Login de Usuários</h3>

<a href="Cadastro.aspx">Criar Conta</a>

<hr />

<!-- Componente customizado para login -->

**<asp:Login**

**ID="formLogin"**

**runat="server"**

**OnAuthenticate="AutenticarUsuario"**

**TitleText="Formulário de Login"**

**UserNameLabelText="Login de Acesso:"**

**PasswordLabelText="Senha:"**

**LoginButtonText="Acessar Sistema"**

**DisplayRememberMe="false"**

**FailureText="Acesso Negado. Tente novamente."**

**DestinationPageUrl="/Admin/Home.aspx"**

**/>**

</div>

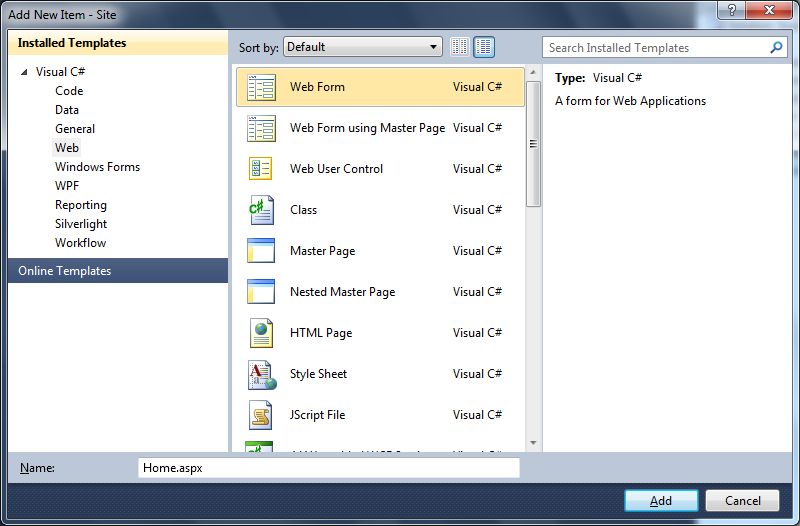
</form>

</body>

</html>



Criando a página de boas vindas do sistema



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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<h3>Bem vindo ao Sistema</h3>

<hr />

<p>

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

</p>

<asp:Button ID="btnLogout" runat="server" Text="Sair do Sistema" />

</div>

</form>

</body>

</html>



**1º Passo**: Mapear para o projeto Asp.Net qual o nome da página de autenticação de usuários

\Web.config.xml

<?xml version="1.0"?>

<configuration>

<system.web>

<!-- Mapeamento da página de Login de Usuários -->

<!-- Forms: Autenticação é feita por página web -->

**<authentication mode="Forms">**

**<!-- loginUrl: caminho da página de autenticação -->**

**<forms loginUrl="/Pages/Login.aspx"/>**

**</authentication>**

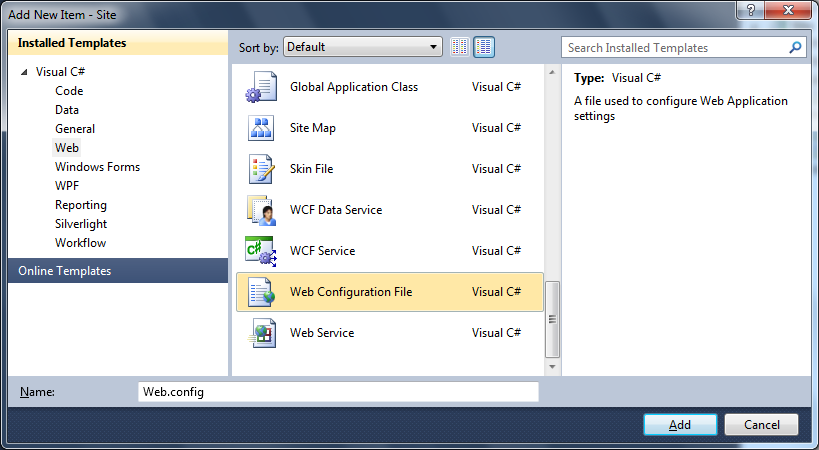
<compilation debug="true" targetFramework="4.0" />

</system.web>

</configuration>

**2º Passo**: Mapear para a área do projeto que terão acesso restrito a usuários.

\Admin\Web.config.xml



<?xml version="1.0"?>

<configuration>

<system.web>

<!-- Regra de Acesso -->

<!-- Bloquear Usuários Anônimos [?] -->

**<authorization>**

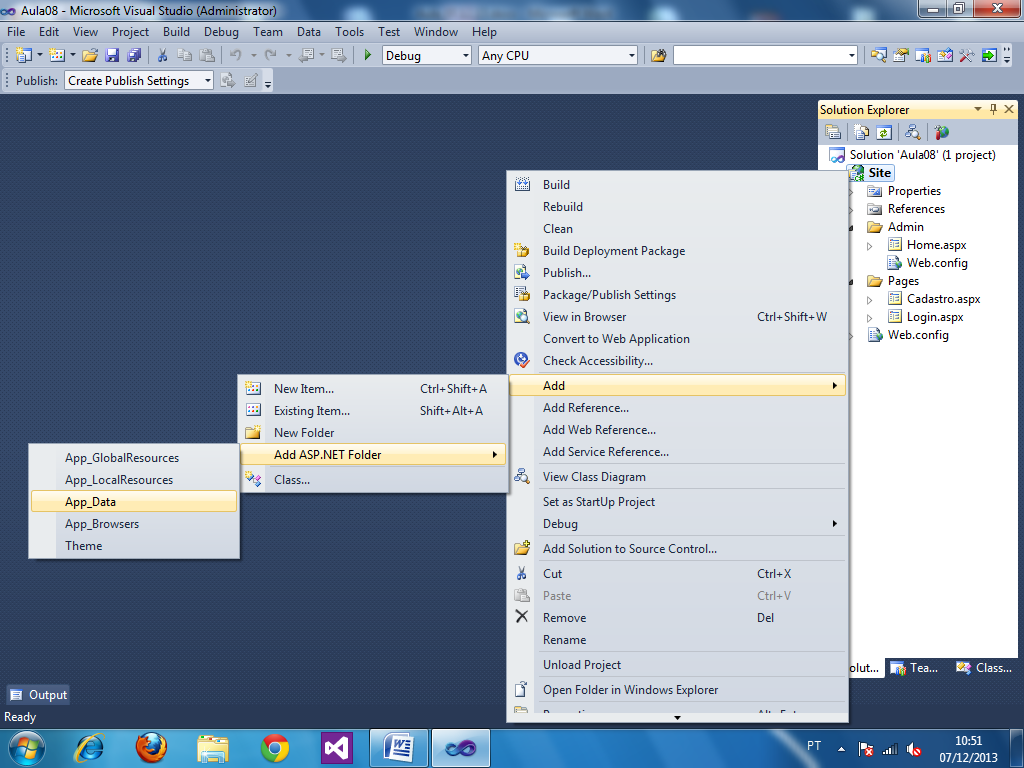
**<deny users="?"/>**

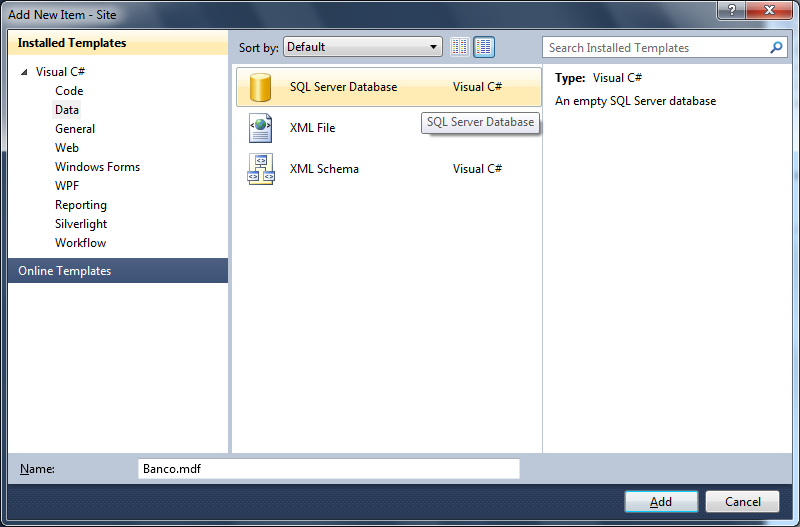
**</authorization>**

</system.web>

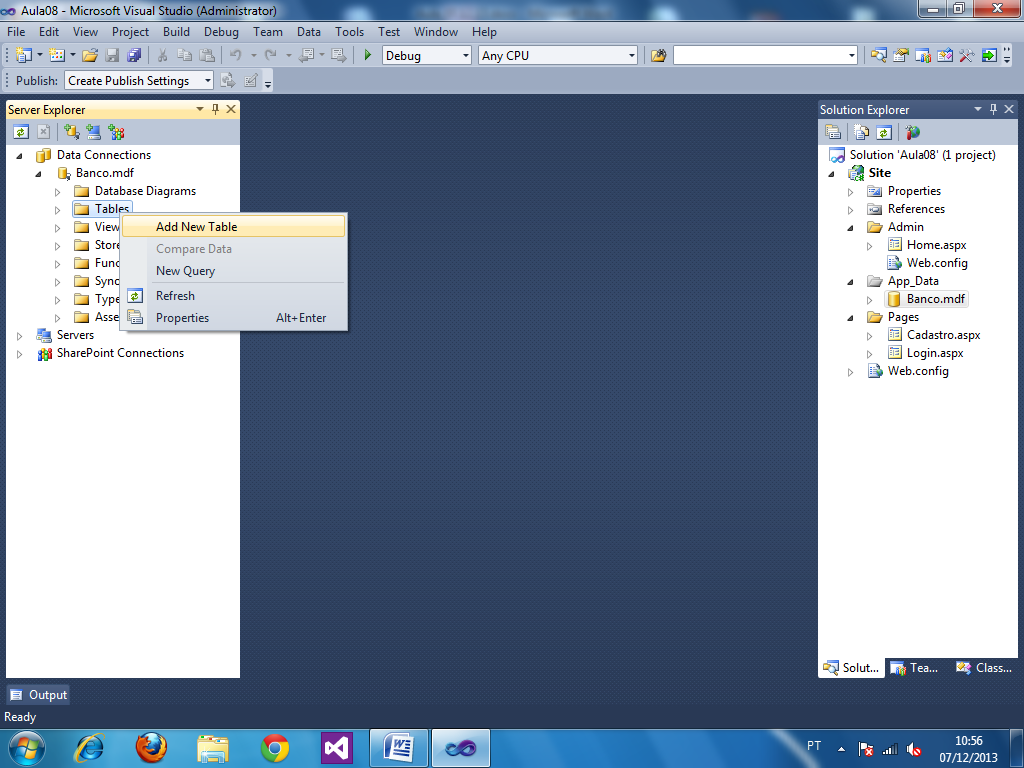
</configuration>

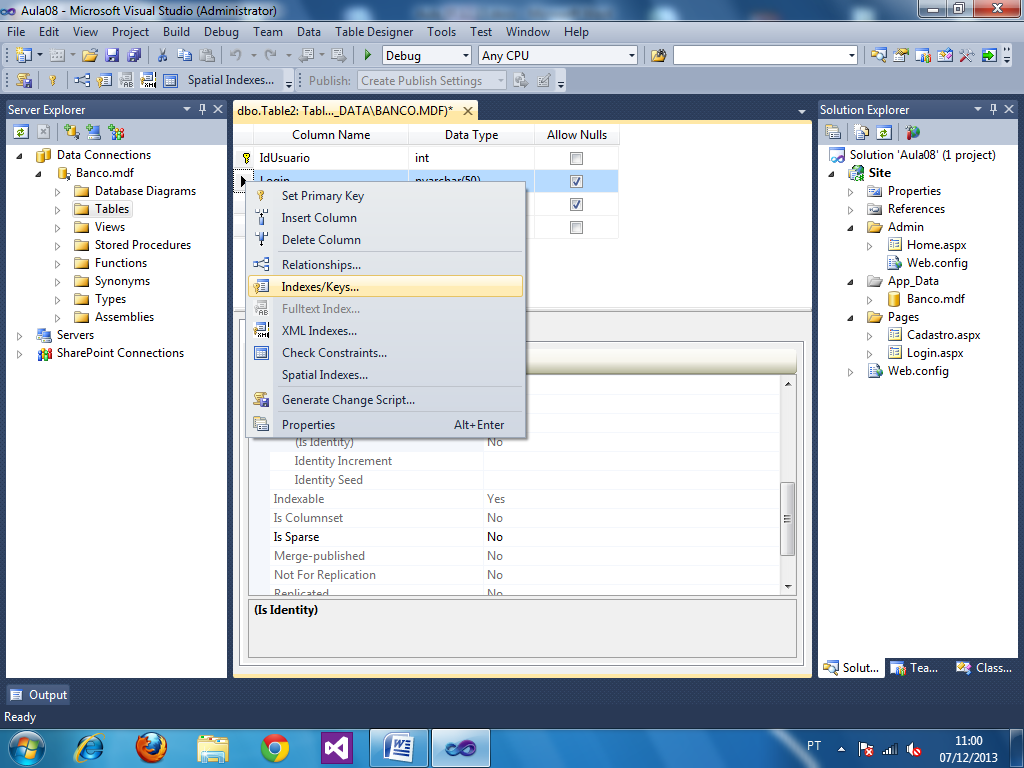
Criando a base de dados...

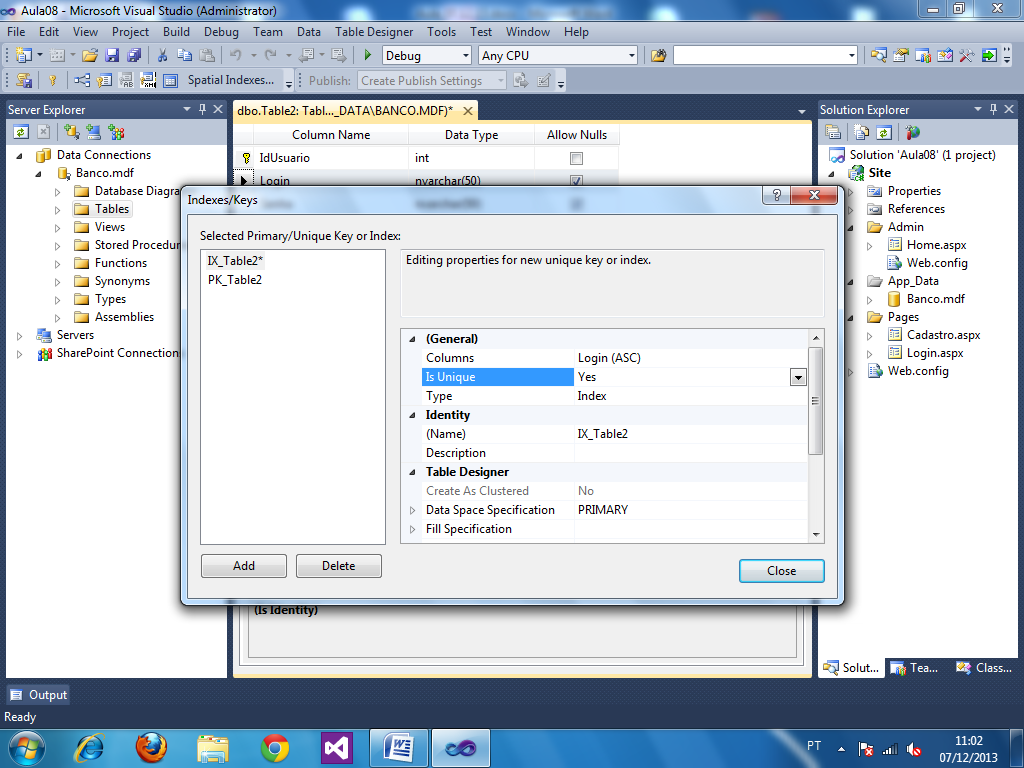


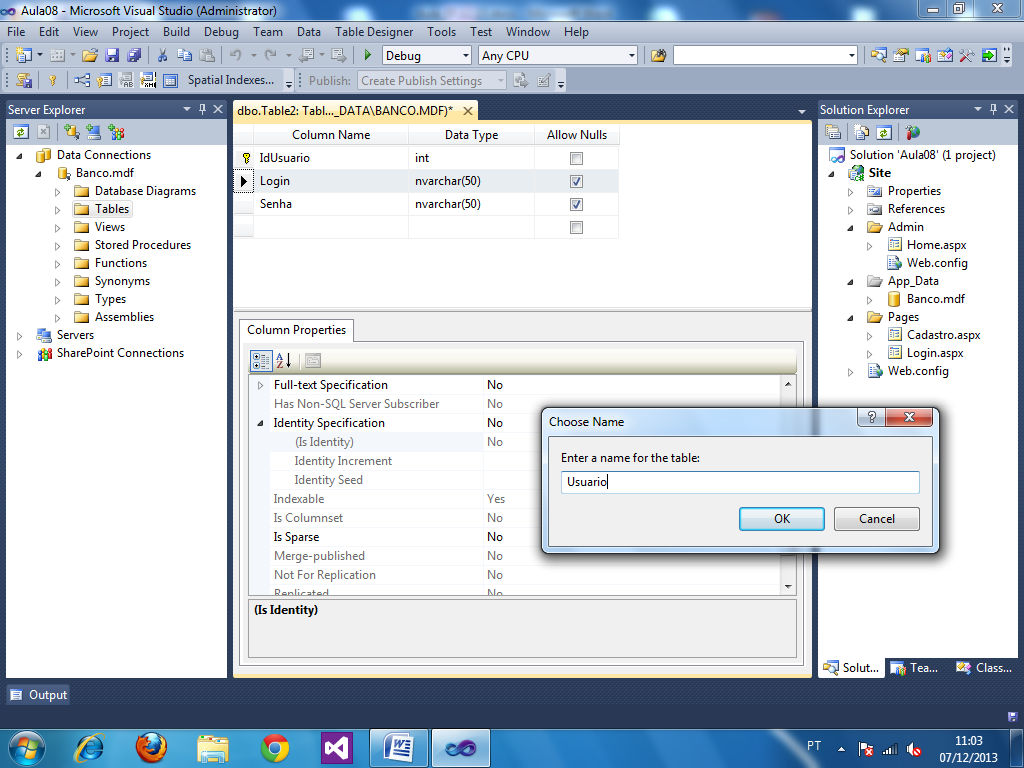


Criando a tabela de Usuários...



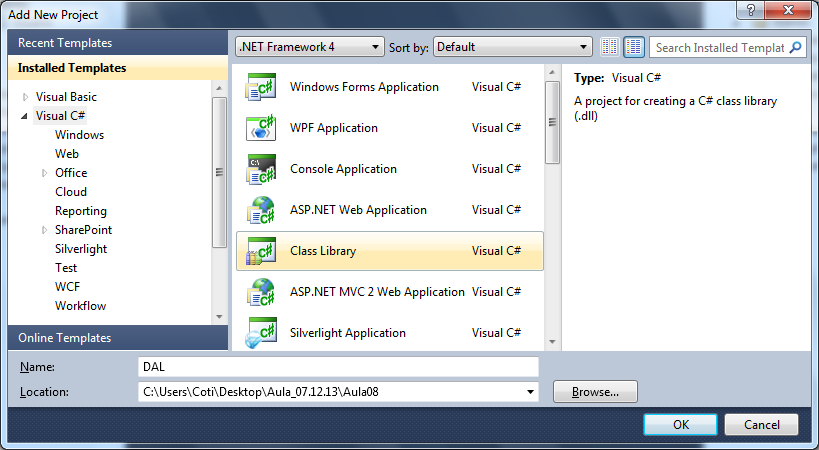




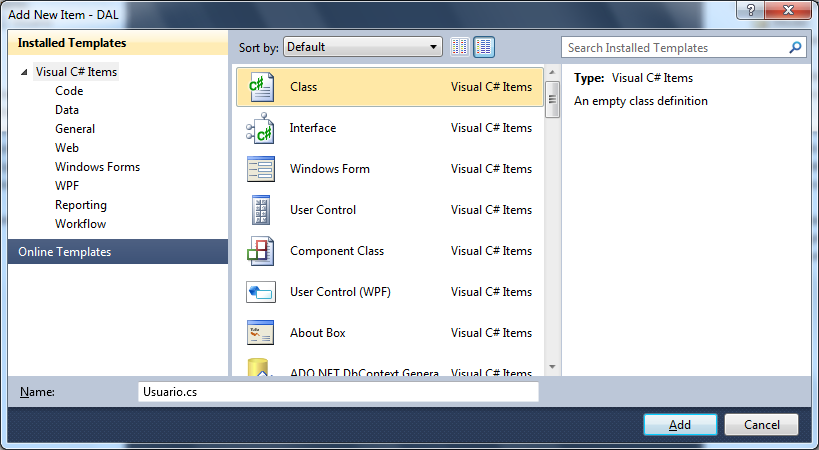


Data Access Layer (DAL)

Camada de acesso a dados



Classe para modelar Usuario...



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace DAL.Model

{

public class Usuario

{

public int IdUsuario { get; set; }

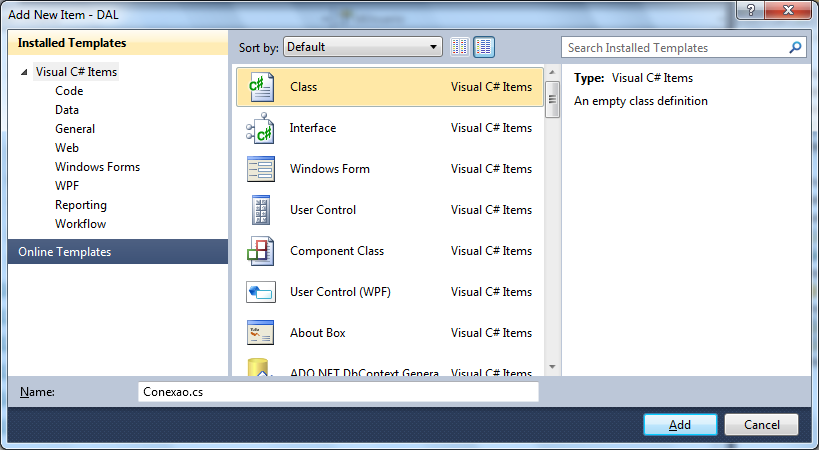
public string Login { get; set; }

public string Senha { get; set; }

}

}

Classe para Conexão com o BD



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

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

namespace DAL.Persistence

{

public class Conexao

{

protected SqlConnection Con;

protected SqlCommand Cmd;

protected SqlDataReader Dr;

protected void AbrirConexao()

{

Con = new SqlConnection();

Con.Open();

}

protected void FecharConexao()

{

Con.Close();

}

}

}

Mapeamento da String de Conexão...

<?xml version="1.0"?>

<configuration>

<!-- Mapeamento da String de Conexão -->

**<connectionStrings>**

**<add**

**name="AULA"**

**connectionString="Data Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\Coti\Desktop\Aula\_07.12.13\Aula08\Site\App\_Data\Banco.mdf;Integrated Security=True;User Instance=True"**

**/>**

**</connectionStrings>**

<system.web>

<!-- Mapeamento da página de Login de Usuários -->

<!-- Forms: Autenticação é feita por página web -->

<authentication mode="Forms">

<!-- loginUrl: caminho da página de autenticação -->

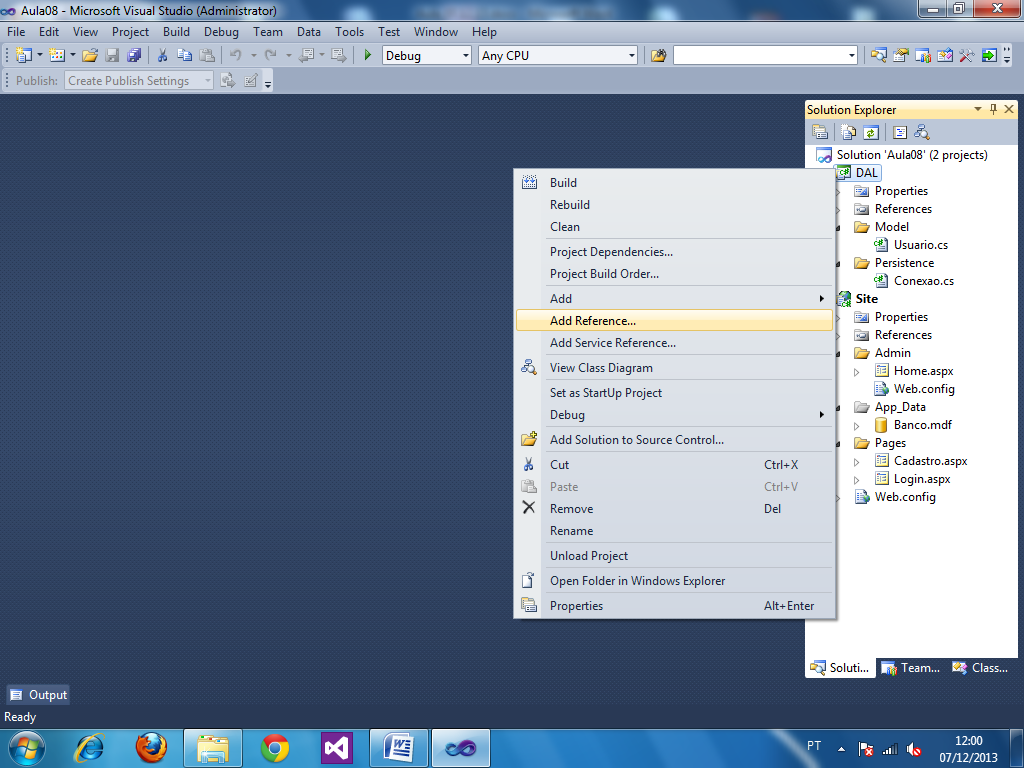
<forms loginUrl="/Pages/Login.aspx"/>

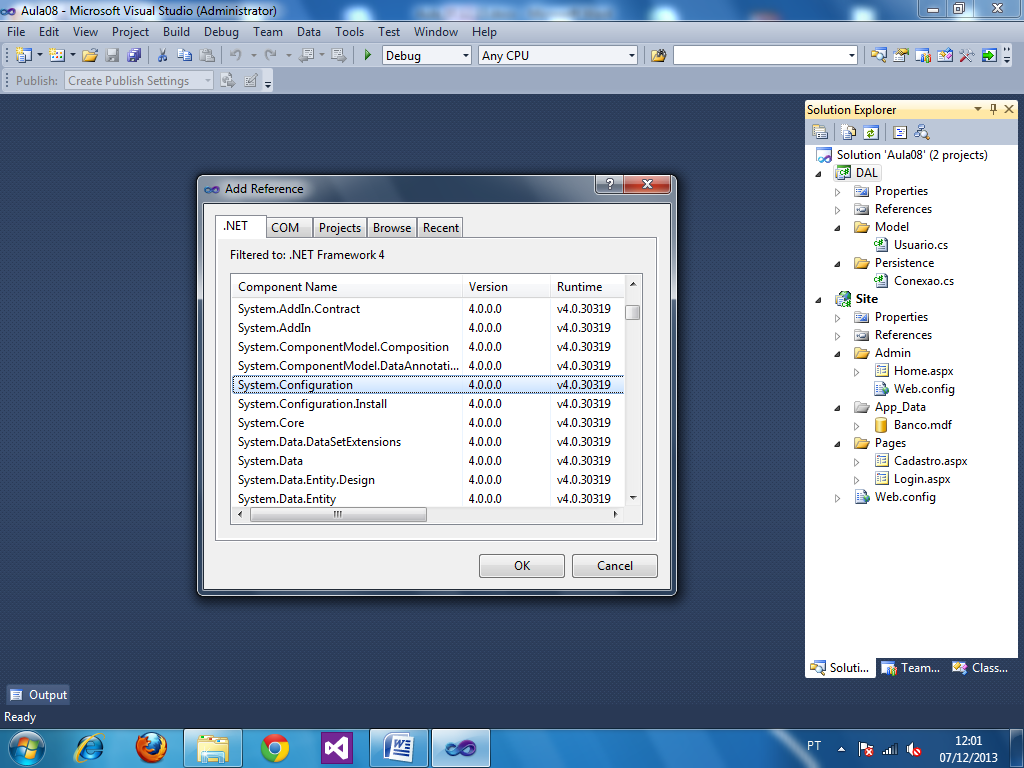
</authentication>

<compilation debug="true" targetFramework="4.0" />

</system.web>

</configuration>





using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

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

using System.Configuration;

namespace DAL.Persistence

{

public class Conexao

{

protected SqlConnection Con;

protected SqlCommand Cmd;

protected SqlDataReader Dr;

protected void AbrirConexao()

{

Con = new SqlConnection();

Con.ConnectionString = ConfigurationManager.ConnectionStrings

["AULA"].ConnectionString;

Con.Open();

}

protected void FecharConexao()

{

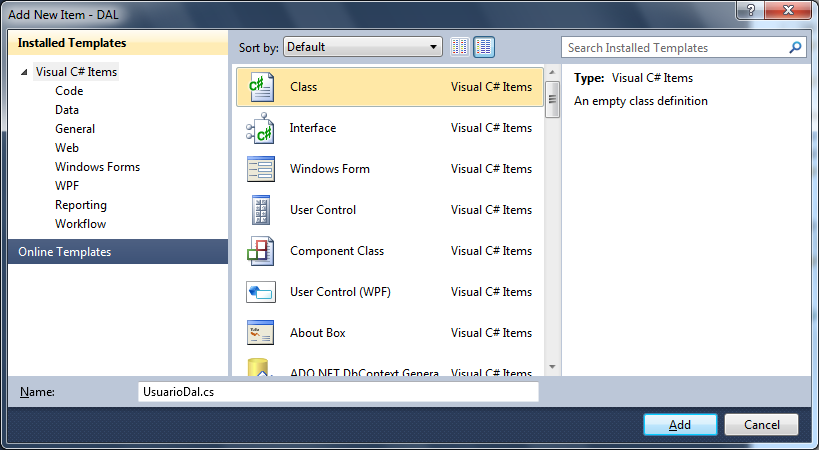
Con.Close();

}

}

}

Classe de persistência para a Classe Usuário...



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.SqlClient;

using DAL.Model;

namespace DAL.Persistence

{

public class UsuarioDal : Conexao

{

//Método para cadastrar um novo usuário

public void Salvar(Usuario u)

{

try

{

AbrirConexao();

Cmd = new SqlCommand("insert into Usuario(Login, Senha)

values(@v1, @v2)", Con);

Cmd.Parameters.AddWithValue("@v1", u.Login);

Cmd.Parameters.AddWithValue("@v2", u.Senha);

Cmd.ExecuteNonQuery(); //executar

}

catch(Exception e)

{

throw new Exception("Erro ao cadastrar usuário: " + e.Message);

}

finally

{

FecharConexao();

}

}

//Método para verificar se um usuario existe na base de dados

public bool UsuarioExistente(Usuario u)

{

try

{

AbrirConexao();

Cmd = new SqlCommand("select \* from Usuario where Login=@v1

and Senha=@v2", Con);

Cmd.Parameters.AddWithValue("@v1", u.Login);

Cmd.Parameters.AddWithValue("@v2", u.Senha);

Dr = Cmd.ExecuteReader();

//executa a consulta e obtem os resultados

//Verificar se o SqlDataReader obteve algum

//registro a partir da consulta

if(Dr.Read()) //indica se o SqlDataReader leu algum registro

{

return true; //Usuario existente!

}

else

{

return false; //Usuario não existe!

}

}

catch(Exception e)

{

throw new Exception("Erro ao consultar usuário: " + e.Message);

}

finally

{

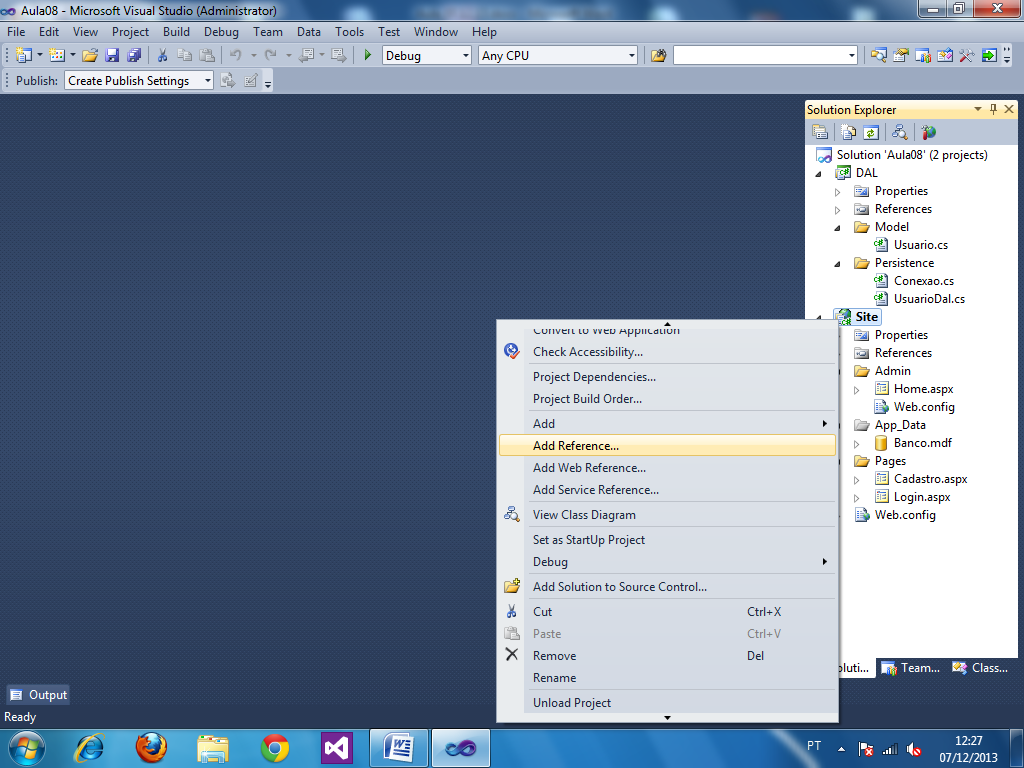
FecharConexao();

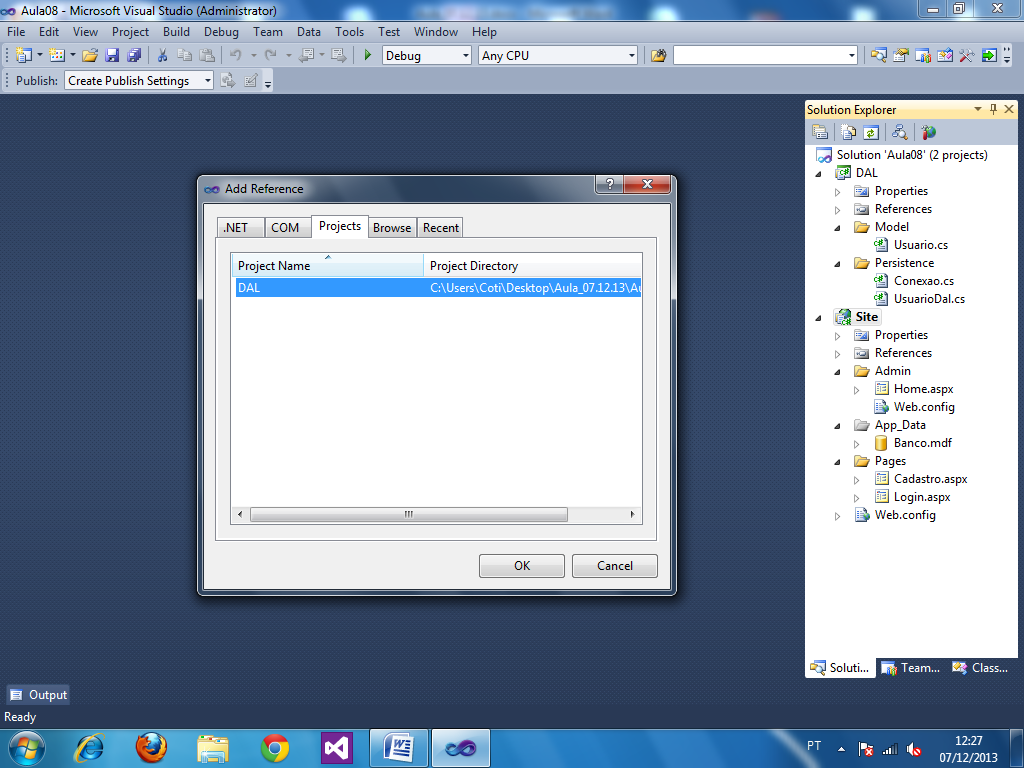
}

}

}

}





Aplicando validações aos campos da página de Cadastro...

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<h3>Cadastro de Usuários</h3>

<a href="Login.aspx">Acessar Sistema</a>

<hr />

Login de Acesso: <br />

<asp:TextBox ID="txtLogin" runat="server" />

**<asp:RequiredFieldValidator**

**ID="reqLogin"**

**runat="server"**

**ControlToValidate="txtLogin"**

**ErrorMessage="Por favor, informe o login de acesso."**

**ForeColor="Red"**

**/>**

<br /><br />

Senha: <br />

<asp:TextBox ID="txtSenha" runat="server" TextMode="Password" />

**<asp:RequiredFieldValidator**

**ID="reqSenha"**

**runat="server"**

**ControlToValidate="txtSenha"**

**ErrorMessage="Por favor, informe a senha de acesso."**

**ForeColor="Red"**

**/>**

<br /><br />

Confirme sua Senha: <br />

<asp:TextBox ID="txtSenhaConfirm" runat="server" TextMode="Password" />

**<asp:RequiredFieldValidator**

**ID="reqSenhaConfirm"**

**runat="server"**

**ControlToValidate="txtSenhaConfirm"**

**ErrorMessage="Por favor, confirme a senha de acesso."**

**ForeColor="Red"**

**/>**

<br /><br />

<asp:Button ID="btnCadastro" runat="server" Text="Cadastrar Usuário" />

<p>

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

**<asp:CompareValidator**

**ID="compareSenha"**

**runat="server"**

**ControlToValidate="txtSenha"**

**ControlToCompare="txtSenhaConfirm"**

**ErrorMessage="Por favor, confirme sua senha corretamente."**

**ForeColor="Red"**

**/>**

</p>

</div>

</form>

</body>

</html>

Criando o evento de cadastro...

<asp:Button ID="btnCadastro" runat="server"

Text="Cadastrar Usuário"

**OnClick="CadastrarUsuario"** />

CodeBehind

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using DAL.Model;

using DAL.Persistence;

using System.Web.Security; //criptografia

namespace Site.Pages

{

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

{

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

{

}

protected void CadastrarUsuario(object sender, EventArgs e)

{

try

{

Usuario u = new Usuario();

u.Login = txtLogin.Text;

u.Senha = FormsAuthentication.

HashPasswordForStoringInConfigFile

(txtSenha.Text, "MD5");

//MD5 -> Algoritmo de Criptografia baseado em chave de 128bits

UsuarioDal d = new UsuarioDal();

d.Salvar(u);

lblMensagem.Text = "Usuário cadastrado com sucesso.";

}

catch(Exception ex)

{

lblMensagem.Text = ex.Message;

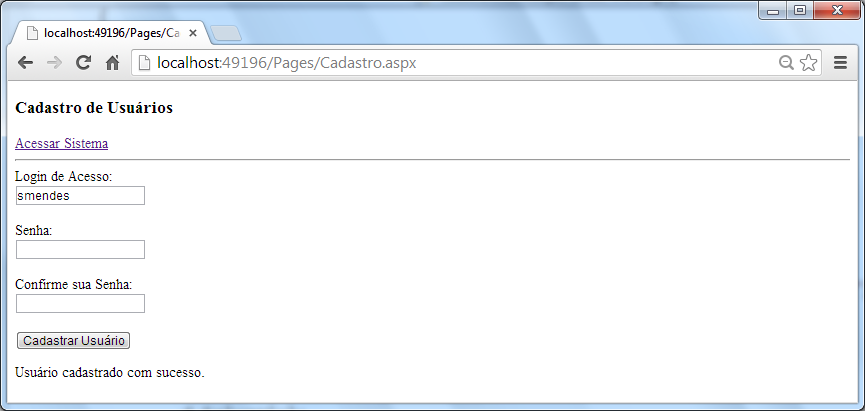
}

}

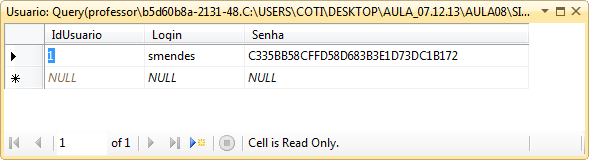
}

}

Executando...



No SqlServer...



Implementando a autenticação   
de Usuários...

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<h3>Login de Usuários</h3>

<a href="Cadastro.aspx">Criar Conta</a>

<hr />

<!-- Componente customizado para login -->

<asp:Login

**ID="formLogin"**

runat="server"

**OnAuthenticate="AutenticarUsuario"**

TitleText="Formulário de Login"

UserNameLabelText="Login de Acesso:"

PasswordLabelText="Senha:"

LoginButtonText="Acessar Sistema"

DisplayRememberMe="false"

FailureText="Acesso Negado. Tente novamente."

DestinationPageUrl="/Admin/Home.aspx"

/>

</div>

</form>

</body>

</html>

CodeBehind...

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using DAL.Model;

using DAL.Persistence;

using System.Web.Security;

namespace Site.Pages

{

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

{

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

{

}

protected void AutenticarUsuario(object sender, AuthenticateEventArgs e)

{

try

{

Usuario u = new Usuario();

u.Login = formLogin.UserName; //Resgato o nome do usuario (login)

u.Senha = FormsAuthentication.

HashPasswordForStoringInConfigFile

(formLogin.Password, "MD5");

UsuarioDal d = new UsuarioDal(); //Classe de persistência

**if (d.UsuarioExistente(u))**

**{**

**e.Authenticated = true; //Concedo o acesso**

**}**

**else**

**{**

**e.Authenticated = false; //Acesso Negado**

**}**

}

catch(Exception ex)

{

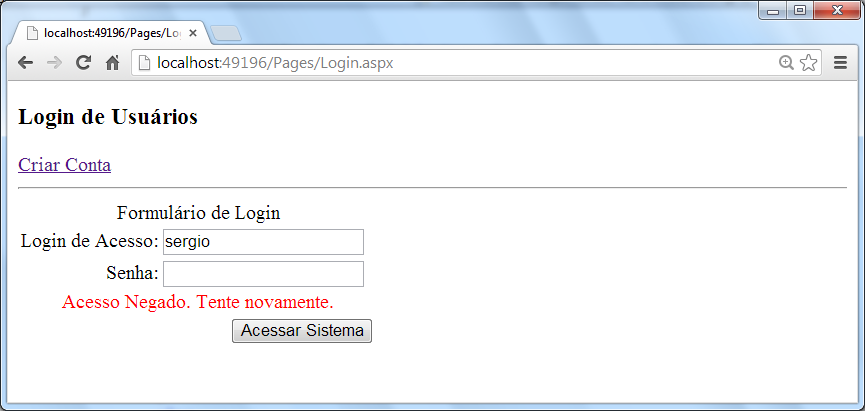
Response.Write(ex.Message);

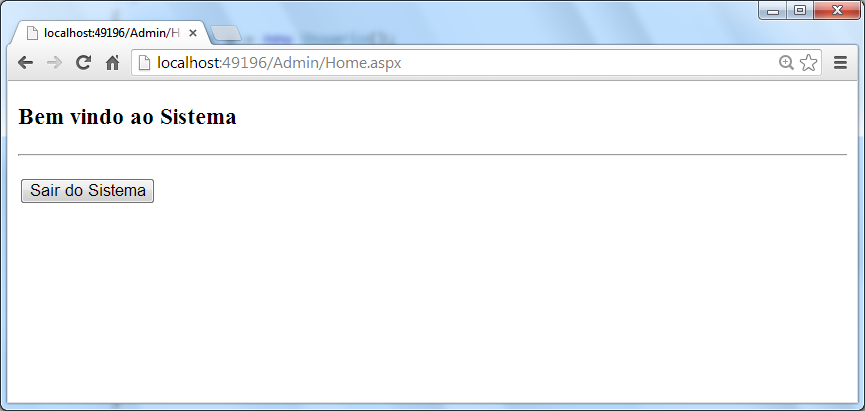
}

}

}

}





Método na Classe UsuarioDal para verificar se um determinado Login já existe cadastrado na tabela...

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.SqlClient;

using DAL.Model;

namespace DAL.Persistence

{

public class UsuarioDal : Conexao

{

//Método para cadastrar um novo usuário

public void Salvar(Usuario u)

{

try

{

AbrirConexao();

Cmd = new SqlCommand("insert into Usuario(Login, Senha)

values(@v1, @v2)", Con);

Cmd.Parameters.AddWithValue("@v1", u.Login);

Cmd.Parameters.AddWithValue("@v2", u.Senha);

Cmd.ExecuteNonQuery(); //executar

}

catch(Exception e)

{

throw new Exception("Erro ao cadastrar usuário: " + e.Message);

}

finally

{

FecharConexao();

}

}

//Método para verificar se um usuario existe na base de dados

public bool UsuarioExistente(Usuario u)

{

try

{

AbrirConexao();

Cmd = new SqlCommand("select \* from Usuario where Login=@v1

and Senha=@v2", Con);

Cmd.Parameters.AddWithValue("@v1", u.Login);

Cmd.Parameters.AddWithValue("@v2", u.Senha);

Dr = Cmd.ExecuteReader();

//executa a consulta e obtem os resultados

//Verificar se o SqlDataReader obteve algum

//registro a partir da consulta

if(Dr.Read()) //indica se o SqlDataReader leu algum registro

{

return true; //Usuario existente!

}

else

{

return false; //Usuario não existe!

}

}

catch(Exception e)

{

throw new Exception("Erro ao consultar usuário: " + e.Message);

}

finally

{

FecharConexao();

}

}

//Método para verificar se um Login informado já existe

public bool LoginExistente(string Login)

{

try

{

AbrirConexao();

Cmd = new SqlCommand("select \* from Usuario where

Login = @v1", Con);

Cmd.Parameters.AddWithValue("@v1", Login);

Dr = Cmd.ExecuteReader();

//Executa a consulta e retorna os dados obtidos

if (Dr.Read()) //Se o SqlDataReader obteve algum registro

{

return true; //Login existente

}

else

{

return false; //Login não existente

}

}

catch(Exception e)

{

throw new Exception("Erro ao verificar Login: " + e.Message);

}

finally

{

FecharConexao();

}

}

}

}

Cadastro.aspx > CodeBehind

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using DAL.Model;

using DAL.Persistence;

using System.Web.Security; //criptografia

namespace Site.Pages

{

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

{

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

{

}

protected void CadastrarUsuario(object sender, EventArgs e)

{

try

{

UsuarioDal d = new UsuarioDal();

//Se login não existe

if( ! d.LoginExistente(txtLogin.Text) )

{

Usuario u = new Usuario();

u.Login = txtLogin.Text;

u.Senha = FormsAuthentication.

HashPasswordForStoringInConfigFile

(txtSenha.Text, "MD5");

d.Salvar(u);

lblMensagem.Text = "Usuário cadastrado com sucesso.";

}

else

{

lblMensagem.Text = "Este login já foi cadastrado.

Tente outro.";

}

}

catch(Exception ex)

{

lblMensagem.Text = ex.Message;

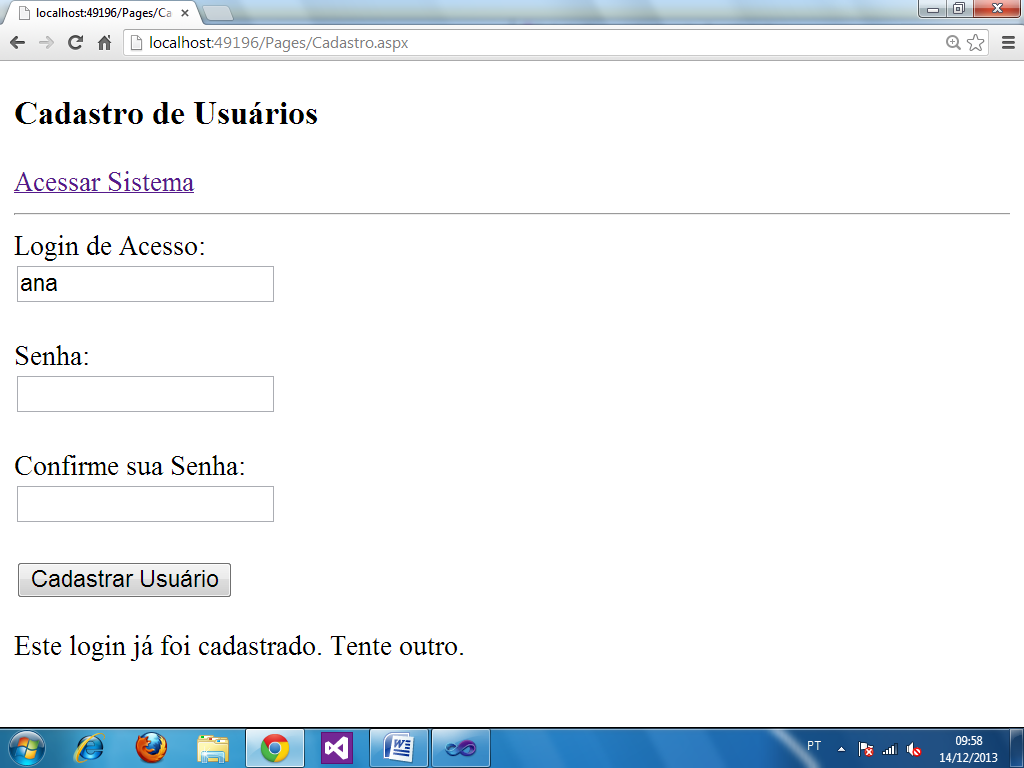
}

}

}

}

Executando...



/Admin/Home.aspx

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

</head>

<body>

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

<div>

<h3>Bem vindo ao Sistema</h3>

<hr />

<p>

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

</p>

**<asp:Button ID="btnLogout" runat="server"**

**Text="Sair do Sistema"**

**OnClick="Logout" />**

</div>

</form>

</body>

</html>

CodeBehind

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace Site.Admin

{

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

{

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

{

}

**protected void Logout(object sender, EventArgs e)**

**{**

**}**

}

}

Implementando o controle de Sessão

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using DAL.Model;

using DAL.Persistence;

using System.Web.Security;

namespace Site.Pages

{

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

{

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

{

}

protected void AutenticarUsuario(object sender, AuthenticateEventArgs e)

{

try

{

Usuario u = new Usuario();

u.Login = formLogin.UserName; //Resgato o nome do usuario (login)

u.Senha = FormsAuthentication.

HashPasswordForStoringInConfigFile

(formLogin.Password, "MD5");

UsuarioDal d = new UsuarioDal(); //Classe de persistência

if (d.UsuarioExistente(u))

{

**//Armazenar os dados do Usuário em uma Sessão**

**//Sessão é uma área de memória utilizada pelo navegador**

**//qualquer dado gravado em sessão estará disponivel**

**//enquanto o navegador estiver aberto**

**Session.Add("usuario", u);**

**Session.Timeout = 2; //tempo limite de ociosidade**

**//tempo máximo que esta sessão poderá ficar em**

**//memoria sem ser utilizada**

e.Authenticated = true; //Concedo o acesso

}

else

{

e.Authenticated = false; //Acesso Negado

}

}

catch(Exception ex)

{

Response.Write(ex.Message);

}

}

}

}

/Admin/Home.aspx > CodeBehind

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using DAL.Model;

namespace Site.Admin

{

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

{

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

**{**

**//Resgatar o Usuario contido na Sessão**

**//Verifique se de fato existe um registro de usuario em sessão**

**if( Session["usuario"] != null )**

**{**

**//Resgatar o Usuario contido na sessão**

**Usuario u = Session["usuario"] as Usuario; //casting**

**//Imprimir o nome do usuário**

**lblMensagem.Text = "Usuário Autenticado: " + u.Login;**

**}**

**}**

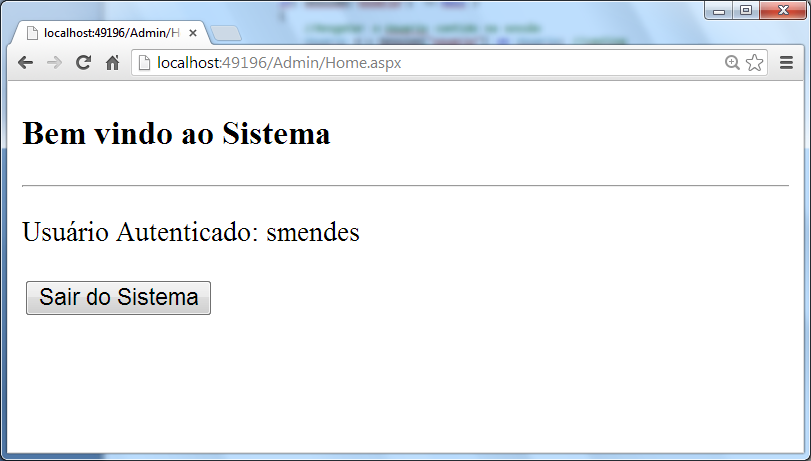
protected void Logout(object sender, EventArgs e)

{

}

}

}



Implementando o Logout de Usuários…

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using DAL.Model;

using System.Web.Security; //Autenticação

namespace Site.Admin

{

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

{

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

{

//Resgatar o Usuario contido na Sessão

//Verifique se de fato existe um registro de usuario em sessão

if( Session["usuario"] != null )

{

//Resgatar o Usuario contido na sessão

Usuario u = Session["usuario"] as Usuario; //casting

//Imprimir o nome do usuário

lblMensagem.Text = "Usuário Autenticado: " + u.Login;

}

}

**protected void Logout(object sender, EventArgs e)**

**{**

**//Destruir a Sessão**

**Session.Remove("usuario"); //remove o registro usuario**

**Session.Abandon(); //invalida e limpa toda a sessão**

**//Quando o usuário é autenticado, ele recebe um ticket de**

**//acesso concedido pelo [Authenticate = true], quando o seu**

**//logout for realizado deveremos destruir esta permissão**

**//fazendo com que o usuario volte a ser um anônimo [?]**

**FormsAuthentication.SignOut(); //destrói a permissão de acesso**

**//Redirecionar de volta para a página de Login mapeada**

**//pelo arquivo \Web.config.xml**

**FormsAuthentication.RedirectToLoginPage();**

**}**

}

}

Limpar o cache da página do Admin

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

<head runat="server">

<title></title>

**<!-- Hacks para que o navegador não guarde cache da página -->**

**<% Response.AddHeader("Last-Modified", DateTime.Now.ToString());%>**

**<% Response.AddHeader("Cache-Control", "no-store, no-cache, must-revalidate,**

**max-age=0, post-check=0, pre-check=0");%>**

**<% Response.AddHeader("Pragma", "no-cache"); %>**

</head>

<body>

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

<div>

<h3>Bem vindo ao Sistema</h3>

<hr />

<p>

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

</p>

<asp:Button ID="btnLogout" runat="server"

Text="Sair do Sistema"

OnClick="Logout" />

</div>

</form>

</body>

</html>

