

Herança

Abrir o Visual Studio

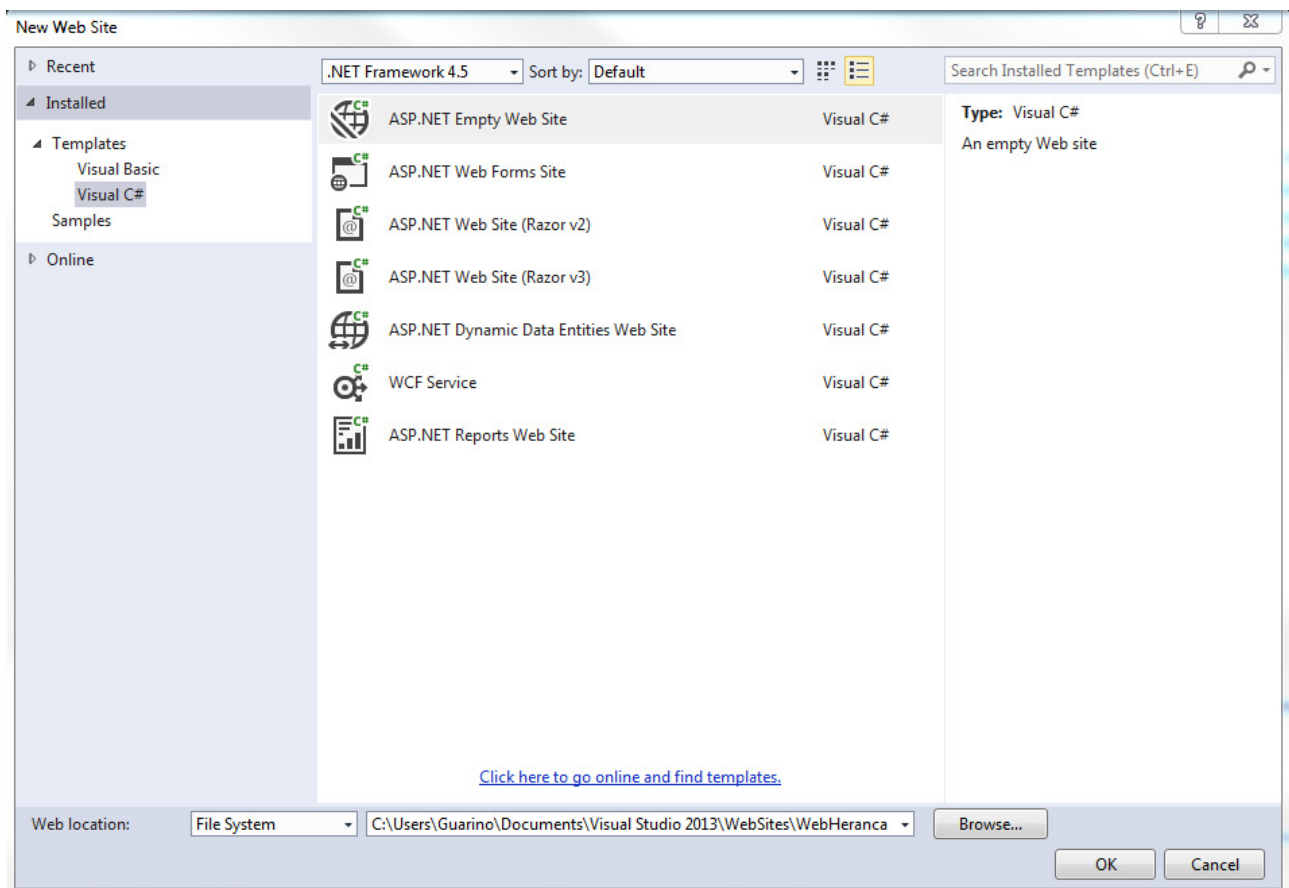
Criar um novo projeto. File > New > WebSite

ASP.NET Empty Web Site

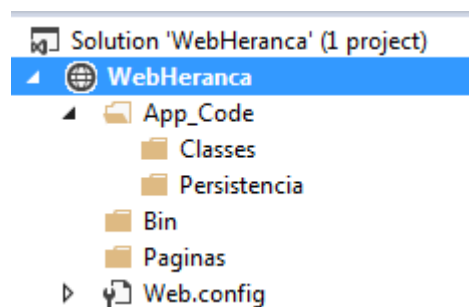
.NET Framework 4.5

Visual C#

Nome: WebHeranca



Criar as pastas na Solution Explorer:



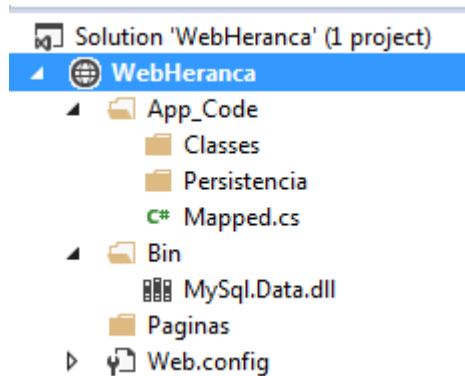
Adicionando a classe de Mapeamento

- Faça o download do arquivo <http://www.4learn.pro.br/guarino/pi/Mapped.zip>
- Descompacte-o

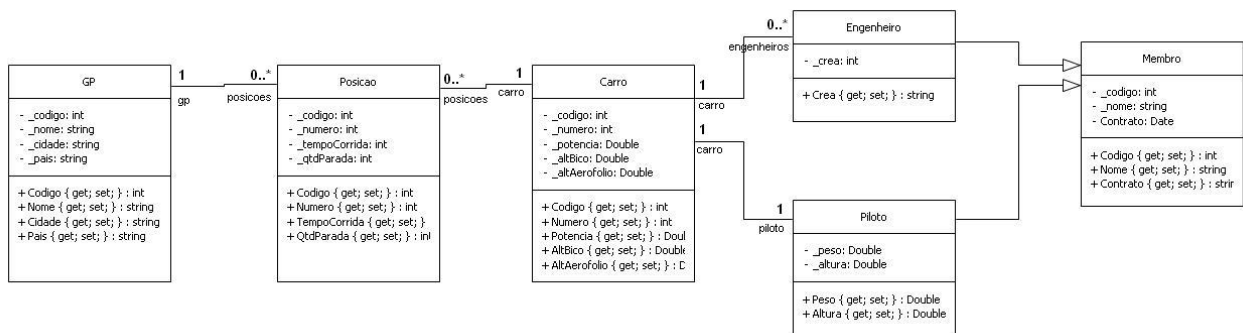
- Adicione-o na pasta App_Code.
 - Botão direito na pasta App_Code > Adicionar Item Existente.
 - Selecionar o arquivo Mapped.cs

Adicionar DLL para acesso ao MySQL

- Baixe o arquivo <http://www.4learn.pro.br/guarino/pi/MySQL.Data.dll>
- Adicione-o na pasta Bin



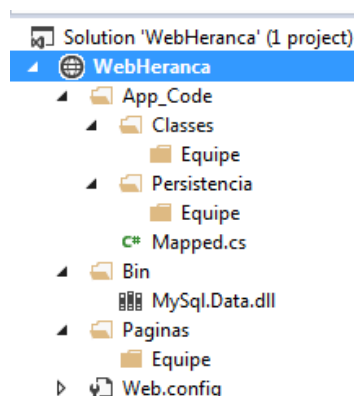
Baseado no Diagrama de Classe a seguir, iremos criar as classes de Modelagem que compõem a Herança.



Para isso, serão criadas as classes Membro, Engenheiro e Piloto.

A fim de facilitar a organização do projeto, estas classes serão agrupadas na pasta Equipe.

Crie a pasta Equipe dentro da pasta Classes, dentro da pasta Persistencia e dentro da pasta Paginas.



Crie a classe Membro.cs dentro da pasta Classes/Equipe

- Defina o namespace
- Crie as propriedadesCodigo, Nome, Contrato
- Crie a propriedade Tipo – será usada para diferenciar as subclasses.

```
using System;

namespace WebHeranca.Classes.Equipe
{
    /// <summary>
    /// Summary description for Membro
    /// </summary>
    public class Membro
    {
        public int Codigo { get; set; }
        public string Nome { get; set; }
        public string Contrato { get; set; }
        public int Tipo { get; set; }

        public Membro()
        {
            //
            // TODO: Add constructor logic here
            //
        }
    }
}
```

Crie a classe Engenheiro

- Defina o namespace
- Defina a herança
- Crie a propriedade CREA

```
using System;

namespace WebHeranca.Classes.Equipe
{
    /// <summary>
    /// Summary description for Engenheiro
    /// </summary>
    public class Engenheiro:Membro
    {
        public string CREA { get; set; }

        public Engenheiro()
        {
            //
            // TODO: Add constructor logic here
            //
        }
    }
}
```

Crie a classe Piloto

- Defina o namespace
- Defina a herança
- Crie as propriedades Peso, Altura

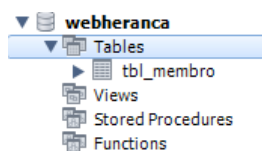
```
using System;
```

```
namespace WebHeranca.Classes.Equipe
{
    /// <summary>
    /// Summary description for Piloto
    /// </summary>
    public class Piloto:Membro
    {
        public double Peso { get; set; }
        public double Altura { get; set; }

        public Piloto()
        {
            //
            // TODO: Add constructor logic here
            //
        }
    }
}
```

Criando o Banco de Dados

- No MySQL Query Browser ou MySQL Workbench, crie o schema (database) webheranca
- Crie a tabela tbl_membro
 - A regra para criação da tabela para Herança é:
 - Todas as classes da Herança formam uma única tabela
 - Todos os campos da classe pai são Not Null
 - Todos os campos das demais classes da herança são Null
 - Adicionar um campo Tipo (INT) para diferenciar as classes. Para o nosso caso, Engenheiro será 0 (ZERO) e Piloto será 1 (UM). Não tem porque adicionarmos Membro.



Observe a tabela




Table Name:

Schema: **webheranca**

Collation:

Engine:

Comments:

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
mem_codigo	INT(10)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
mem_nome	VARCHAR(255)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
mem_contrato	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
mem_tipo	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
mem_crea	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
mem_peso	DECIMAL(6,2)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
mem_altura	DECIMAL(6,2)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Adicionando a string de conexão na aplicação

- Abra o arquivo web.config na Solution Explorer
- Adicione a tag appSettings dentro da tag Configuration, conforme mostrado a seguir
 - Em Database, informe o nome do seu schema (ex: webheranca)
 - Em Data Source, informe o local ou IP do servidor (ex: localhost)
 - Em User Id, informe o usuário que acessará o banco de dados (ex: root)
 - Em Password, informe a senha de acesso ao banco de dados. Caso seu banco de dados não tenha senha, deixe Password=;

```
<?xml version="1.0"?>

<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
-->

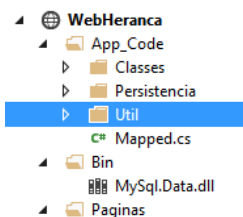
<configuration>

  <appSettings>
    <add key="strConexao" value="Database=webheranca;Data Source=localhost;User
Id=root;Password=; pooling=false"/>
  </appSettings>

  <system.web>
    <compilation debug="true" targetFramework="4.5" />
    <httpRuntime targetFramework="4.5" />
  </system.web>

</configuration>
```

Crie mais uma pasta dentro de App_Code, chamada Util



Nessa pasta, adicione uma classe, chamada Membro.

- Adicione o namespace
- Altere de classe para enum
- Defina o enum com ENGENHEIRO, PILOTO.

```
namespace WebHeranca.Util
{
    public enum MEMBRO
    {
        ENGENHEIRO,
        PILOTO
    }
}
```

Criando a classe que terá os métodos de acesso ao banco de dados relacionados a classe Engenheiro

- Botão direito na pasta Persistencia
- Adicione Nova Classe
- Nome da classe: EngenheiroBD.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using FATEC; //para acesso a classe Mapped
using WebHeranca.Classes.Equipe; //para acesso a classe de modelagem
using System.Data; // para acesso ao DataSet
using WebHeranca.Util; //para acesso ao enum

namespace WebHeranca.Persistencia.Equipe
{
    /// <summary>
    /// Summary description for EngenheiroBD
    /// </summary>
    public class EngenheiroBD
    {
        //métodos
        //insert

        //selectall

        //select

        //update

        //delete

        //construtor
        public EngenheiroBD()
        {
            //
            // TODO: Add constructor logic here
            //
        }
    }
}
```

Dentro da classe EngenheiroBD, adicione o método Insert

```
public int Insert(Engenheiro engenheiro)
{
    int retorno = 0;

    try
    {
        System.Data.IDbConnection objConexao;
        System.Data.IDbCommand objCommand;
        string sql = "INSERT INTO tbl_membro(mem_nome, mem_contrato, mem_tipo,
mem_crea) VALUES (?nome, ?contrato, ?tipo, ?crea)";

        objConexao = Mapped.Connection();
        objCommand = Mapped.Command(sql, objConexao);
```

```

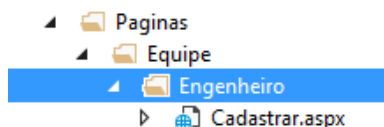
        objCommand.Parameters.Add(Mapped.Parameter("?nome", engenheiro.Nome));
        objCommand.Parameters.Add(Mapped.Parameter("?contrato",
engenheiro.Contrato));
        objCommand.Parameters.Add(Mapped.Parameter("?tipo", MEMBRO.ENGENHEIRO));
        objCommand.Parameters.Add(Mapped.Parameter("?crea", engenheiro.CREA));

        objCommand.ExecuteNonQuery();
        objConexao.Close();
        objCommand.Dispose();
        objConexao.Dispose();
    }
    catch (MySql.Data.MySqlClient.MySqlException)
    {
        retorno = -1;
    }
    catch (Exception)
    {
        retorno = -2;
    }

    return retorno;
}

```

Adicione uma pasta dentro de Paginas/Equipe, chamada Engenheiro.
Dentro dela, adicione um WebForm chamado Cadastrar.aspx.



Crie a tela como:

Cadastro de Engenheiro (Label - ID=lblTitulo, Text=Cadastro de Engenheiro)	
Nome:	(Label - ID=lblNome, Text=Nome:)
<input type="text"/>	(Textbox - ID=txtNome)
Contrato:	(Label - ID=lblContrato, Text=Contrato:)
<input type="text"/>	(Textbox - ID=txtContrato)
CREA:	(Label - ID=lblCREA, Text=CREA:)
<input type="text"/>	(Textbox - ID=txtCREA)
<input type="button" value="Salvar"/>	(Button - ID=btnSalvar, Text=Salvar)
[lblMensagem]	(Label - ID=lblMensagem, Text=vazio)

Acesse o código fonte da tela (Cadastrar.aspx.cs)

Importe os namespaces que serão usados. Coloque-os no começo do código:

```

using WebHeranca.Classes.Equipe;
using WebHeranca.Persistencia.Equipe;

```

No evento Page_Load

```

protected void Page_Load(object sender, EventArgs e)
{
    txtNome.Focus();
}

```

Crie um método privado para Limpar os campos após Salvar

```
private void LimparCampos()
{
    txtNome.Text = "";
    txtContrato.Text = "";
    txtCREA.Text = "";
}
```

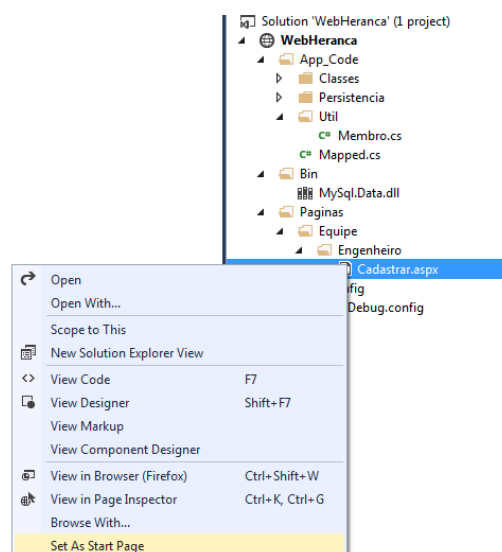
No evento click do Botão

```
protected void btnSalvar_Click(object sender, EventArgs e)
{
    Engenheiro engenheiro = new Engenheiro();
    engenheiro.Nome = txtNome.Text;
    engenheiro.Contrato = txtContrato.Text;
    engenheiro.CREA = txtCREA.Text;

    EngenheiroBD bd = new EngenheiroBD();
    int retorno = bd.Insert(engenheiro);

    switch (retorno)
    {
        case 0:
            LimparCampos();
            txtNome.Focus();
            lblMensagem.Text = "Cadastro realizado com sucesso";
            break;
        case 1:
            //Erro no banco de dados
            lblMensagem.Text = "Não foi possível realizar o cadastro.";
            break;
        case 2:
            //Erro geral
            lblMensagem.Text = "Não foi possível realizar o cadastro.";
            break;
        default:
            break;
    }
}
```

Coloque essa página para ser a primeira a ser executada.



Execute a aplicação e cadastre um Engenheiro.

Listar Engenheiros

Crie um novo método em EngenheiroBD

```
public DataSet SelectAll()
{
    DataSet ds = new DataSet();

    System.Data.IDbConnection objConexao;
    System.Data.IDbCommand objCommand;
    System.Data.IDataAdapter objDataAdapter;

    objConexao = Mapped.Connection();
    objCommand = Mapped.Command("SELECT * FROM tbl_membro WHERE mem_tipo = ?tipo",
objConexao);
    objCommand.Parameters.Add(Mapped.Parameter("?tipo", MEMBRO.ENGENHEIRO));

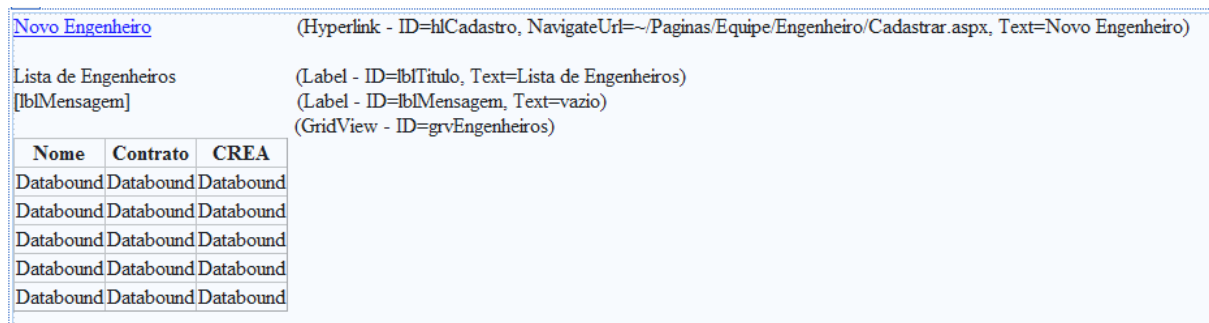
    objDataAdapter = Mapped.Adapter(objCommand);
    objDataAdapter.Fill(ds);

    objConexao.Close();

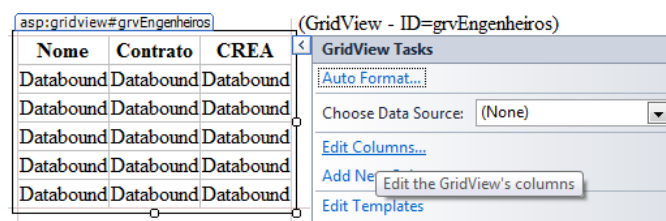
    objCommand.Dispose();
    objConexao.Dispose();

    return ds;
}
```

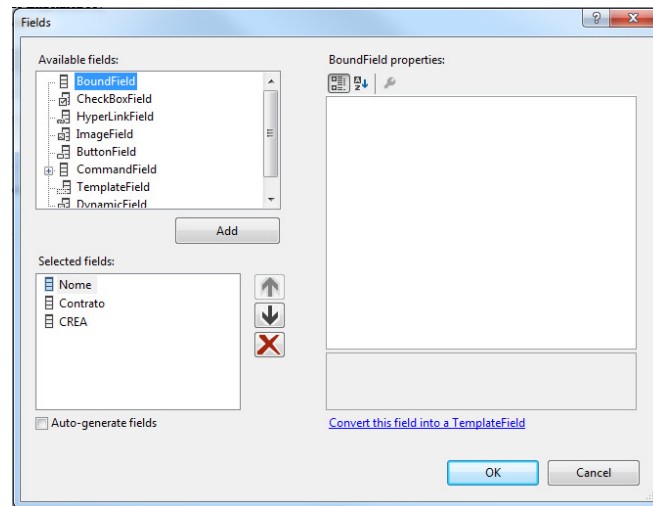
Crie uma nova página, dentro de Paginas/Equipe/Engenheiro, chamada Listar.aspx



Altere o GridView para deixar apenas as colunas Nome, Contrato e CREA.
Para isso, selecione o GridView, clique na opção “<” e depois em Edit Columns...



A tela a seguir irá aparecer:



Desmarque a opção “Auto generate fields” (parte inferior da tela)

Selecione um BoundField e clique em Add.

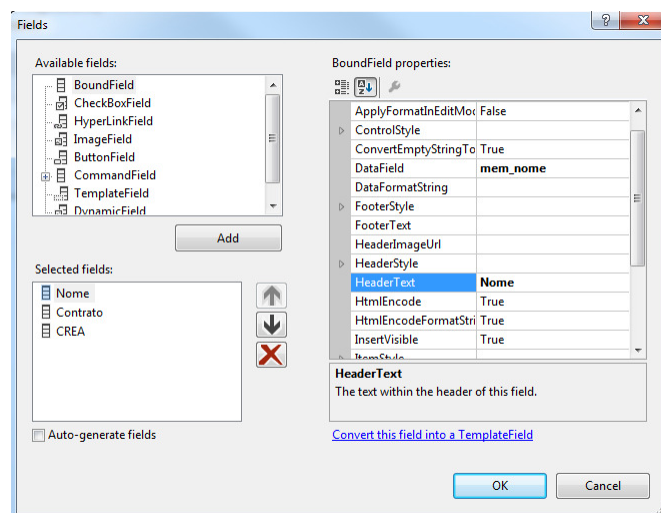
Faça isso 3 vezes.

Cada BoundField corresponde a um campo que será exibido.

Na caixa Selected Fields, selecione o primeiro BoundField e altere as propriedades:

- HeaderText – cabeçalho da coluna
- DataField – campo no banco de dados que será exibido.

Repita isso para os 3 campos.



Acesse o código fonte da tela (Listar.aspx.cs)

Importe os namespaces

```
using WebHeranca.Persistencia.Equipe;  
using System.Data;
```

No evento Page_Load

```
protected void Page_Load(object sender, EventArgs e)
{
    EngenheiroBD bd = new EngenheiroBD();
    DataSet ds = bd.SelectAll();

    //verifica a quantidade de engenheiros no dataset
    int quantidade = ds.Tables[0].Rows.Count;
    if (quantidade > 0)
    {
        grvEngenheiros.DataSource = ds.Tables[0].DefaultView;
        grvEngenheiros.DataBind();
        lblMensagem.Text = "Existem " + quantidade + " engenheiros cadastrados";
    }
    else
    {
        lblMensagem.Text = "Nenhum engenheiro cadastrado";
    }
}
```

Coloque a página Listar.aspx para ser a primeira a ser executada.

Execute a aplicação.

O processo de alteração e exclusão é semelhante ao do tutorial CRUD + MySQL.

Exercícios

1. Faça a alteração de um Engenheiro.
2. Faça a exclusão de um Engenheiro.
3. Faça o CRUD para Piloto.