Prof. Roberto de Castro

Escola Técnica Estadual "São Paulo" - ETESP
Linguagem C# - Visual Studio 2019

Banco de Dados (Continuação)

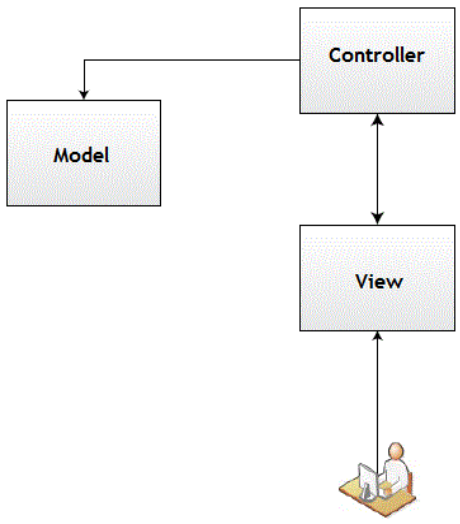
No projeto anterior "dividimos" a solução em "duas camadas": a camada de apresentação (View), que também incorporou a camada da "regra de negócios" (testes de validação dos campos...) e a camada de acesso a dados, que incorporou a "camada de modelagem dos dados" e a manipulação destes ao banco de dados.

No projeto que desenvolveremos nesta etapa separaremos a "regra de negócios" (BLL - Business Logic Layer) da camada de apresentação (View ou User Interface) e separaremos a "camada de modelo de dados" (Entidade/Tabela) da camada de "manipulação dos dados" (DAL → Data Access Layer), criando desta forma uma solução com a seguinte estrutura (projetos):

- **UI (User Interface ou View)** → Projeto "Windows Forms Application" que irá conter os "formulários" de interação/visualização com o usuário;
- **Entidade (ou tabela)** → Projeto do tipo Class Library (Biblioteca de Classes) que irá conter as classes com as entidades (tabelas e campos) que o projeto manipulará;
- **BLL (Business Logic Layer)** → Projeto do tipo Class Library (Biblioteca de Classes) que irá conter as classes com "regras do negócio" (faixa de valores de um campo, obrigatoriedade (ou não), testes de validação....)
- **DAL (Data Access Layer)** → Projeto do tipo Class Library (Biblioteca de Classes) que irá conter as classes com os métodos para manipulação das tabelas na Base de Dados: Inclusão, consulta, alteração, exclusão....

  OBSERVAÇÃO: "DAL" ou "DAO"??? → DAL se refere à camada de acesso a dados como um todo, enquanto DAO são os objetos que compõem essa camada.

Esta proposta de "dividir" as responsabilidades pela execução de cada tarefa (Interface gráfica, Lógica, Acesso a dados...) não é nova. O então funcionário da corporação Xerox PARC, Trygve Reenskaug, iniciou em 1979 o que seria o nascimento do padrão de projeto MVC (Model, View e Controller). A implementação original foi descrita no artigo "Applications Programming in Smalltalk-80: How to use Model-View-Controller": separar o projeto em três camadas independentes: o modelo (Model), a visão (View) e o controlador (Controller), podendo facilitar a manutenção do código e sua reutilização em outros projetos.



Onde:

Model → é a camada responsável pela disponibilização dos dados e a realização das ações de atualização, inserção e exclusão destes dados no Banco de Dados. OBS: Neste modelo, a camada integra (reúne) a declaração das "entidades" (campos/tabelas) e dos "serviços de acesso aos dados".

View → representa a interface com o usuário.

Controller → implementa as regras do negócio, por exemplo. Receber a solicitação do usuário e encaminhar o pedido ao banco de dados para retornar uma lista de produtos em estoque e devolver o resultado na tela do computador (formulário).

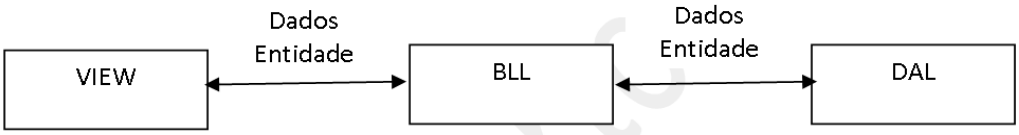
O modelo acima, se comparado com a nossa proposta de desenvolvimento, teremos:

➔ UI (user interface) = View.

➔ Entidade (ou tabela) = Model (somente com a declaração das entidades (campos/tabelas, SEM os serviços de acesso aos dados).

➔ BLL (Business Logic Layer) = Controller.

➔ DAL (Data Access Layer) = Model (com os serviços de acesso aos dados).



Notas explicativas complementares:

1. Programação em "camadas" é a "técnica" de refinamento (divisão/separação) do programa em "partes menores". É o "meio" para a implementação do "padrão MVC".
   As camadas poderão ser "físicas" (cada camada é criada em um projeto separado, compilado na forma de "DLL") ou "lógicas" (criadas dentro do mesmo projeto, podendo, por exemplo, serem separadas em pastas). Embora possamos definir todas genericamente por "camadas", alguns artigos/autores separam o conceito em Tier (camada física) e Layer (camada lógica).
2. MVC estabele (define) o que cada camada deverá conter e qual será a forma de "comunicação/interação" entre as camadas.
3. A "camada de dados" (entidade/tabela) é considerada como uma "camada auxiliar", utilizada para transferir/transportar os dados entre as demais camadas. É chamada tecnicamente de **DTO** (Data Transfer Object - Objeto de transferência de dados), normalmente utilizada para transferir/recuperar dados de um banco de dados. A grande diferença entre esta camada e as demais é que não possuirá nenhum tipo de "comportamento" (método).
4. Não há uma regra rígida e inflexível. Boa parte da decisão fica a critério dos projetistas de sistemas que adaptam os modelos existentes às necessidades específicas de cada software, podendo, inclusive, criar camadas adicionais.

Iniciaremos, a partir deste ponto, o desenvolvimento do projeto de "Cadastro de Clientes".

Etapa I

Criação do Banco de Dados e Tabela

Nome do Banco de Dados: **"bdMVC"**

Tabelas (entidades) → Regras IMPORTANTES para "nomear" os campos das tabelas:

- Não utilize "espaços", por exemplo: "Tabela de Clientes" (sugestão para nomear o campo: "**TabClientes**");

- Adote um "padrão" na escrita dos nomes dos campos, pois na programação o campo "Descricao" é diferente do campo "descricao";

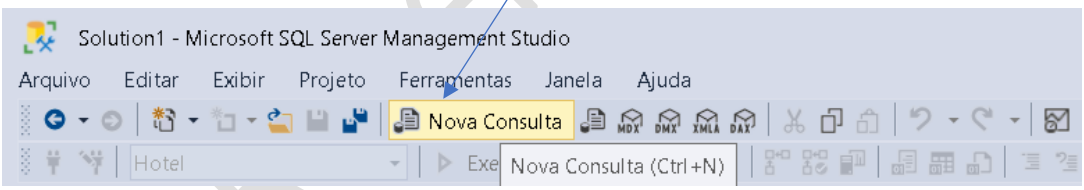
- Não utilize "caracteres da língua portuguesa" (acentos, cedilhas....), como por exemplo: descrição.

Lay-out da tabela de Clientes: TabClientes

Campo	Tipo	Observação
CPFCliente	Número BigInt	Campo Chave
NomeCliente	Texto - Varchar	Tamanho 100 / Requerido
ContatoCliente	Texto - Varchar	Tamanho 100 / Requerido
Observacoes	Texto - Varchar	200

Com base nos dados acima, vamos acessar o SQL Server Management Studio (**SSMS**) para a criação do Banco de Dados, tabela e respectivos campos.

SQL SERVER – SCRIPTS
(clicar no botão "Nova Consulta" para exibir a área de script)



SCRIPTS

1. **Criação do Banco de Dados**:

```
create database bdMVC
go (clicar em executar)
```

2. **Criação da tabela "TabClientes"**

```
Use bdMVC
Create table TabClientes
(
   CPFCliente Bigint primary key not null,
   NomeCliente Varchar(100) not null,
   ContatoCliente varchar(100) not null,
   Observacoes Varchar(200)
);
Go
```

3. Inclusão de Dados na tabela "TabClientes"

ATENÇÃO: O campo CPF precisa conter 11 dígitos, para ser corretamente exibido mascarado pela aplicação: 999.999.999-99!!

```
use bdMVC
Insert into TabClientes (CPFCliente, NomeCliente, ContatoCliente, Observacoes) values
(11111111111,'Cliente A','Contato@clientea.com','Excelente cliente')
Go

Insert into TabClientes (CPFCliente, NomeCliente, ContatoCliente, Observacoes) values
(22222222222,'Cliente B','Contato@clienteabcom','Excelente cliente')
Go


Insert into TabClientes (CPFCliente, NomeCliente, ContatoCliente, Observacoes) values
(33333333333,'Cliente C','Contato@clienteac.com','Cliente mal pagador')
Go
```

	CPFCliente	NomeCliente	ContatoCliente	Observacoes
1	11111111111	Cliente A	Contato@clientea.com	Excelente cliente
2	22222222222	Cliente B	Contato@clienteabcom	Excelente cliente
3	33333333333	Cliente C	Contato@clienteac.com	Cliente mal pagador

Chegamos ao final desta primeira etapa de desenvolvimento do projeto (criação do Banco de Dados, Tabelas e Relacionamentos. Podemos finalizar o SQL Server).

Vamos avançar para a segunda etapa!!

Etapa II

Criação da estrutura básica da solução

A solução do projeto será composta por 4 (quatro) partes distintas. Nesta etapa criaremos a estrutura básica dele.

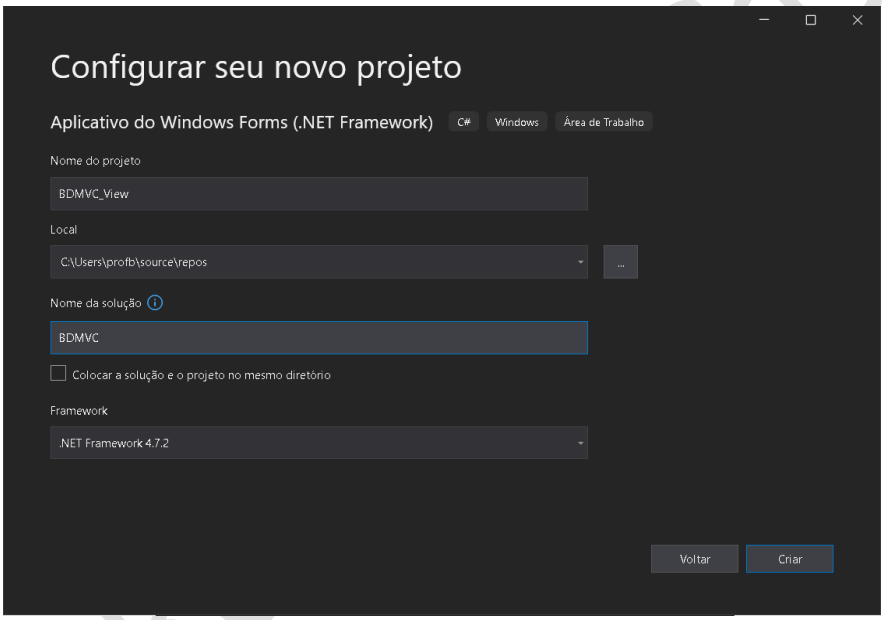
Abra o Visual Studio, selecione o tipo de projeto "Aplicativo do Windows Forms (.Net Framework)" C# - Área de Trabalho (idêntico aos projetos anteriores) e informe os dados abaixo:

Nome do Projeto → **BDMVC_View**

Local → defina o local para a gravação do projeto

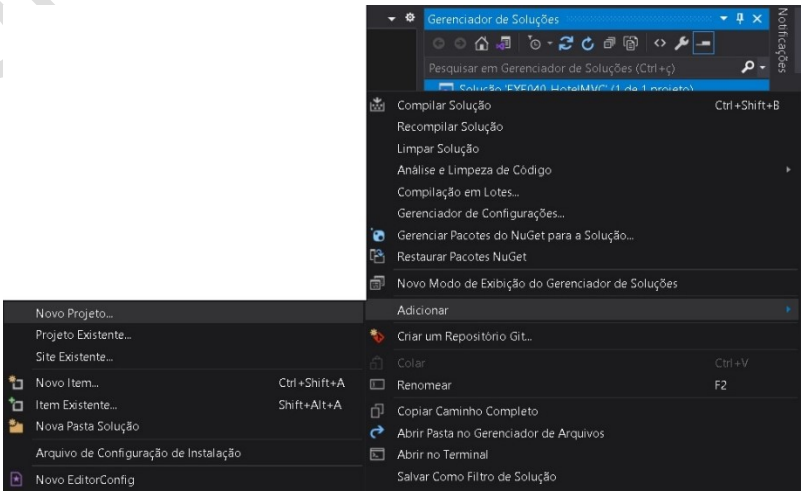
Nome da Solução → **BDMVC**

Observe que **NÃO estamos** utilizando o mesmo nome do projeto para a solução. A razão disto é que nossa solução será composta por quatro projetos... Vamos caminhar mais!



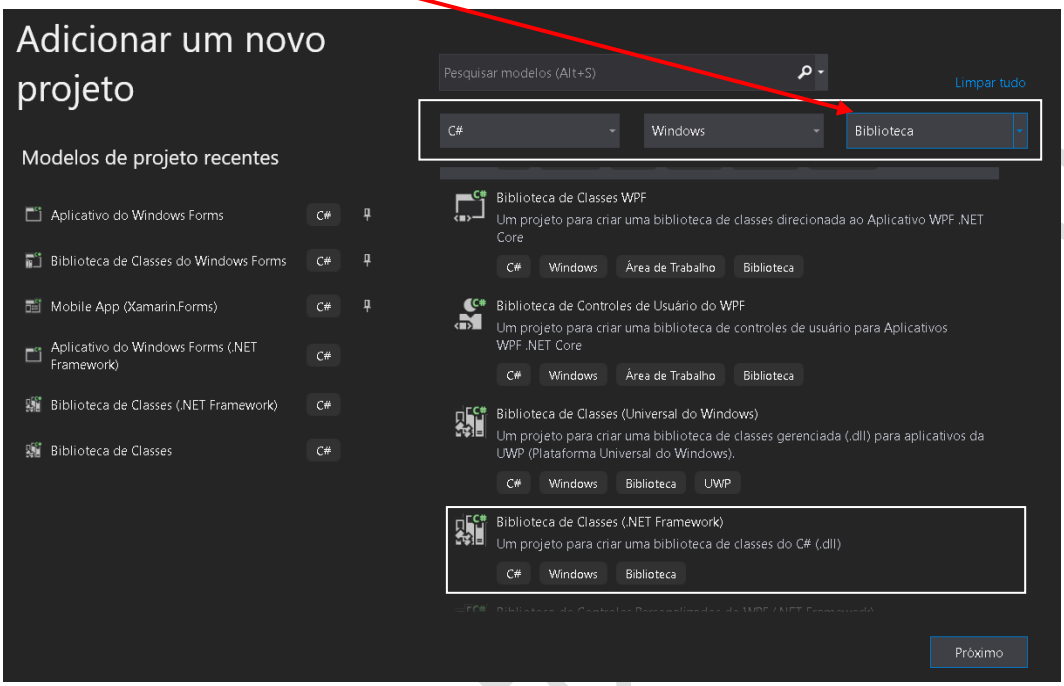
Clique em "Criar". Este primeiro projeto será o responsável pela apresentação da interface gráfica (formulários).

Agora vamos criar a segunda camada. Clique com o botão direito sobre o **nome da solução** (no gerenciador de soluções), selecione Adicionar → Novo Projeto (veja a imagem). **MUITA ATENÇÃO AO PROCEDIMENTO!!!**



Este novo projeto, que será incluído na mesma solução, será o responsável pela declaração das entidades (tabelas e campos) e será diferente dos projetos anteriores. *Vamos incluir uma "Biblioteca de Classes" e não um projeto do tipo "Windows Forms".*

Faça a seleção: "C#" → Windows → **Biblioteca** (conforme o modelo abaixo):



Selecione o tipo "**Biblioteca de Classes (.Net Framework)**" e clique em proximo.

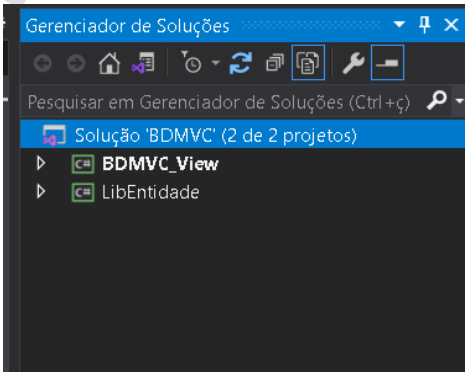
Informe os dados abaixo:

Nome do Projeto → LibEntidade ("Lib" abreviação de Library - Biblioteca)

Local → **DEVERÁ SER O MESMO LOCAL INFORMADO PELO SISTEMA!!!**

Clique em Criar.

Veja no "Solution Explorer" (Gerenciador de Soluções) o "quadro" de como está a "solução" até o momento:



Observe que o projeto "BDMVC_View" é exibido em "negrito", indicando ser o projeto "´principal".

Vamos repetir o mesmo procedimento para incluir a terceira camada ao projeto, responsável pela manipulação dos dados no Banco de Dados (a camada DAL → Data Access Library).

Clique com o botão direito do mouse sobre o **nome da solução** (no gerenciador de soluções), selecione Adicionar → Novo Projeto .

Selecione o tipo "**Biblioteca de Classes (.Net Framework**) e clique em proximo.

Informe os dados abaixo:

Nome do Projeto → **LibDAL**

Local → **DEVERÁ SER O MESMO LOCAL INFORMADO PELO SISTEMA!!!**

Clique em Criar.

Vamos repetir o mesmo procedimento para incluir a quarta e última camada ao projeto, responsável pela "lógica" de funcionamento do sistema (a camada BLL).

Clique com o botão direito do mouse sobre o **nome da solução** (no gerenciador de soluções), selecione Adicionar → Novo Projeto .

Selecione o tipo "Biblioteca de Classes (.Net Framework) e clique em proximo.

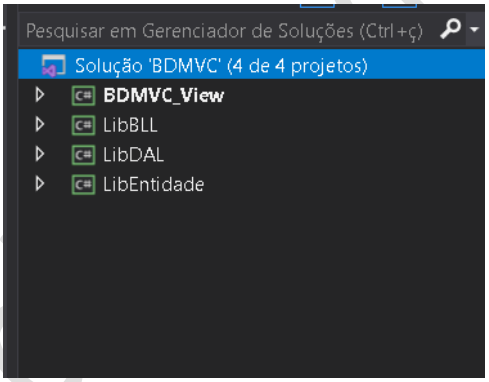
Informe os dados abaixo:

Nome do Projeto → **LibBLL**

Local → **DEVERÁ SER O MESMO LOCAL INFORMADO PELO SISTEMA!!!**

Clique em Criar.

Veja no "Solution Explorer" (Gerenciador de Soluções) o "quadro" da situação até o momento:



Podemos confirmar se a estrutura do projeto está de acordo com a proposta apresentada até o momento, clique com o botão direito do mouse sobre a solução (selecione a opção "Abrir Pasta" no Gerenciador de Arquivos) e observe a estrutura criada:

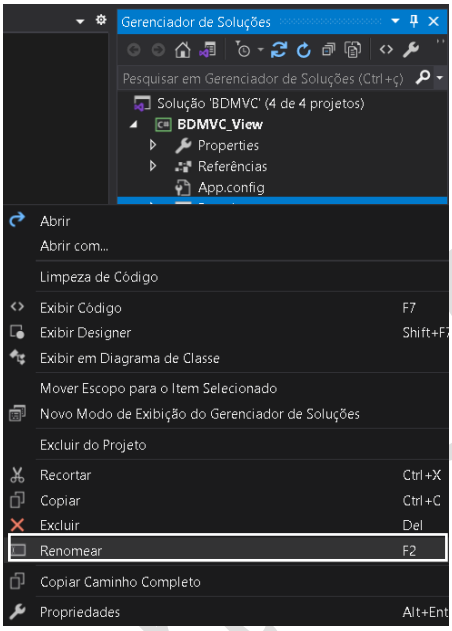
Nome	Data de modificação	Tipo	Tamanho
.vs	18/11/2022 09:29	Pasta de arquivos	
BDMVC_View	18/11/2022 09:28	Pasta de arquivos	
LibBLL	18/11/2022 09:36	Pasta de arquivos	
LibDAL	18/11/2022 09:34	Pasta de arquivos	
LibEntidade	18/11/2022 09:31	Pasta de arquivos	
BDMVC.sln	18/11/2022 09:29	Visual Studio Solut...	2 KB

Etapa III

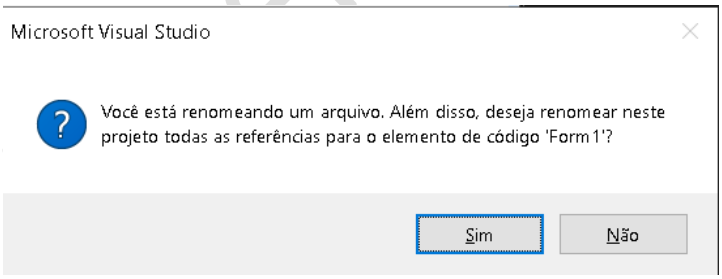
Criação da estrutura básica da solução - Formulários

Neste momento iremos "desenhar" o formulário existente na camada "View" (BDMVC_View), que será utilizado para o "Cadastro de Clientes".

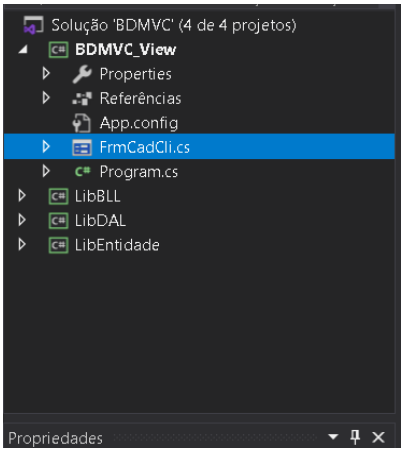
No "Gerenciador de Soluções", selecione e "expanda" **o projeto "BDMVC_View"** e altere o nome do formulário existente para "FrmCadCli".



IMPORTANTE: Após a digitação do nome "FrmCadCli" o sistema exibirá a mensagem:



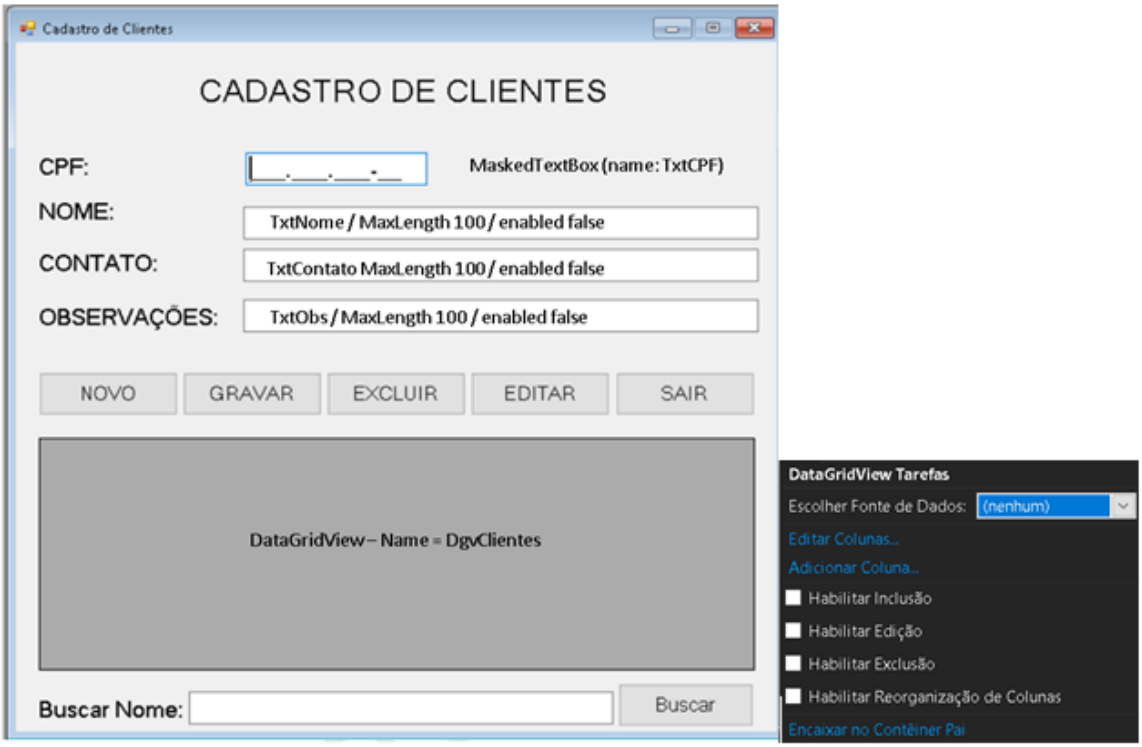
Clique em "SIM" e certifique-se de que o formulário foi renomeado corretamente, conforme a imagem abaixo:



Vamos configurar este segundo formulário, seguindo as orientações abaixo:

Propriedade	Conteúdo
Name	FrmCadCli (já deverá estar alterada)
Text	Cadastro de Clientes
StartPosition	CenterScreen
MaximizeBox	False

"Design" do formulário de "Cadastro de Clientes":



Inicialmente os "TextBox" deverão estar desabilitados (enabled = false).

Enabled "false" também para os botões "GRAVAR" "EXCLUIR" E "EDITAR". Desabilite também as propriedades do "DataGridView (Habilitar Inclusão, Edição....)

A propriedade "SelectionMode" do DatagridView (DgVClientes) foi alterada para "FullRowSelect" (indica que ao clicar na linha do "grid", toda a linha é selecionada. Desmarque as opções "Habilitar Inclusão", "Habilitar Edição". NÃO EXISTE NENHUMA FONTE DE DADOS VINCULADA!!!

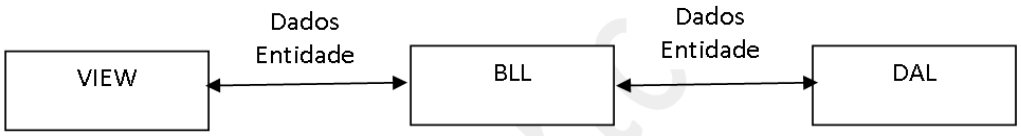
A proposta é de, ao carregar o formulário, exibir os clientes cadastrados no "DataGridView" e caso o usuário desejar fazer alguma manipulação em algum cliente (Excluir / Editar), deverá selecionar no "DataGridView" (clicar na linha desejada) e os dados serão exibidos nos "TextBox". Caso desejar incluir um novo cliente, o usuário deverá clicar em "NOVO", que destravará os TextBox e habilitará o botão "GRAVAR". Mas veremos com mais detalhes em outro momento, por enquanto é somente o "design".

ETAPA IV

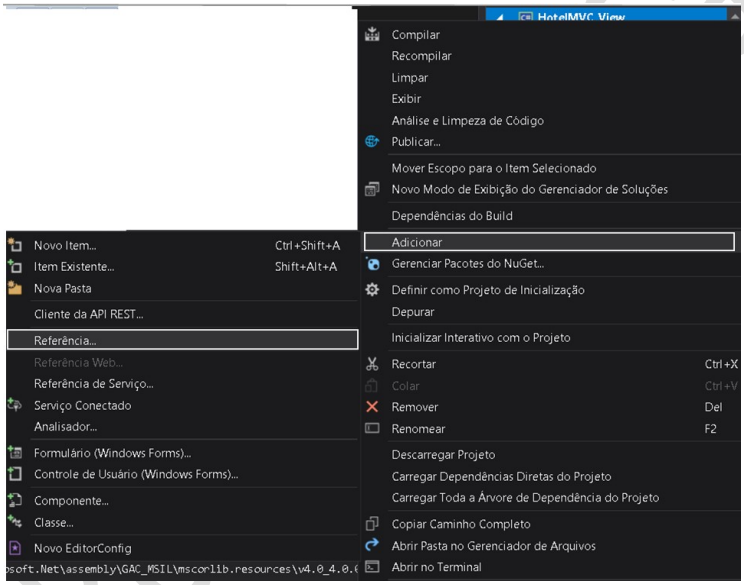
CONFIGURAÇÃO DOS RELACIONAMENTOS ENTRE AS CAMADAS

A solução do Sistema está composta por 4 (quatro) camadas/projetos, porém, até o momento uma camada não "sabe" nada sobre a existência das demais.

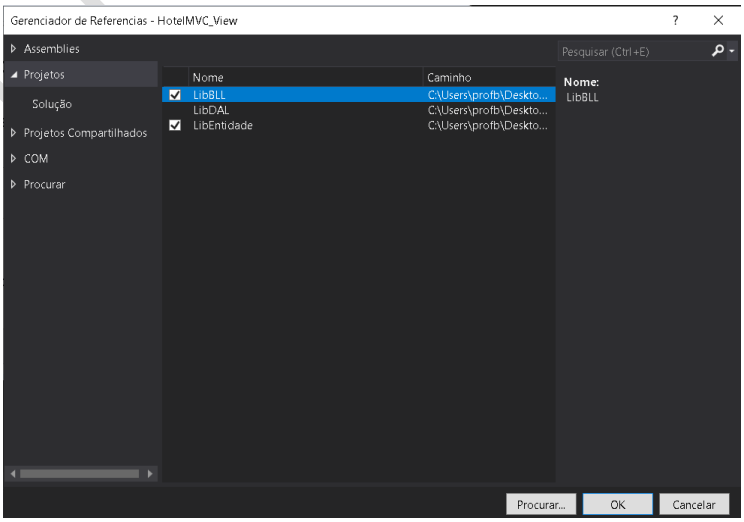
A inclusão das referências permitirá o relacionamento/comunicação entre as camadas do projeto.



No "Gerenciador de Soluções", clique com o botão direito do mouse no projeto "BDMVC_View" e selecione Adicionar → Referência

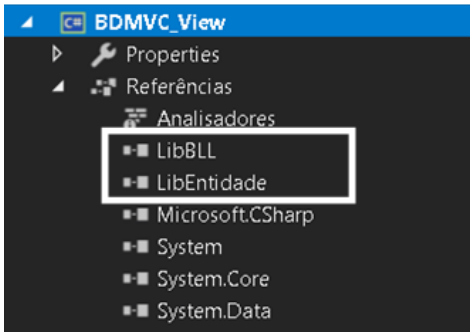


Selecione a LibBLL e a LibEntidade:



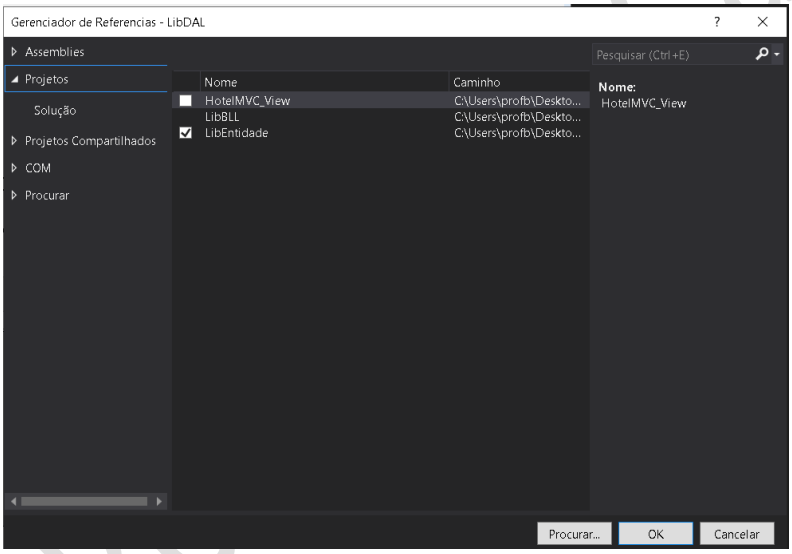
Clique em "OK".

Observe no "Gerenciador de Soluções" a inclusão das referências no projeto "BDMVC_View":



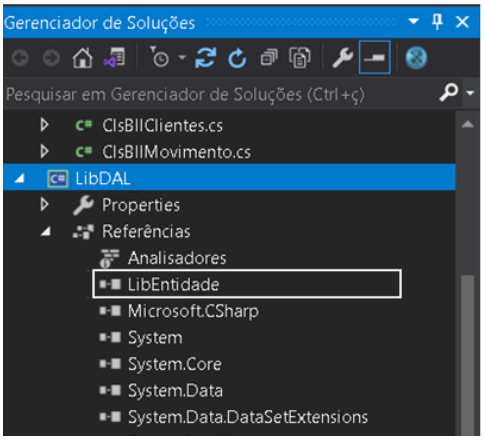
Vamos repetir o processo, agora para incluir os relacionamentos (referencias) no projeto "LibDAL". Selecione o projeto "LibDAL" no "Gerenciador de Soluções", pressione o botão direito do mouse sobre este projeto → Adicionar → Referencia.

ATENÇÃO: Selecione somente a "LibEntidade"!!!



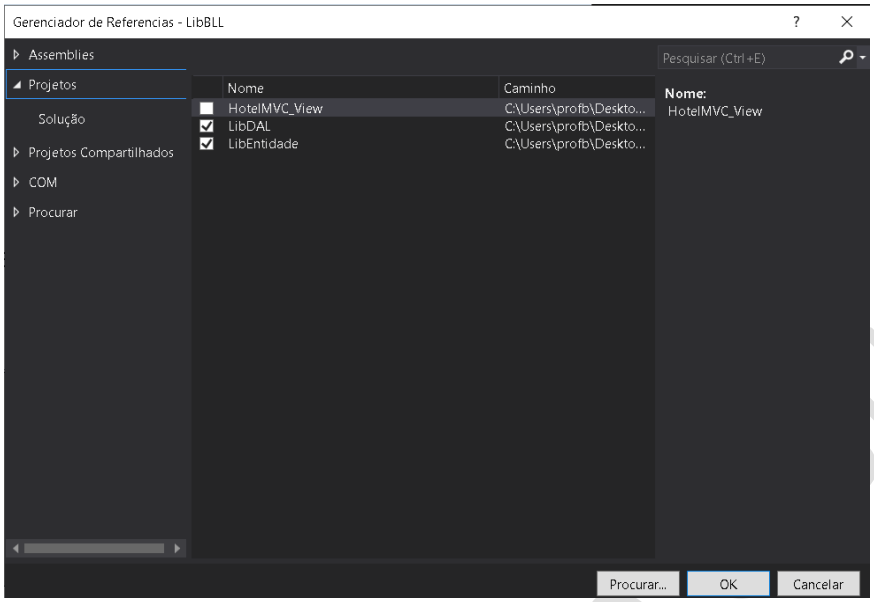
Clique em "OK".

Observe no "Gerenciador de Soluções" a inclusão da referência no projeto "LibDAL":



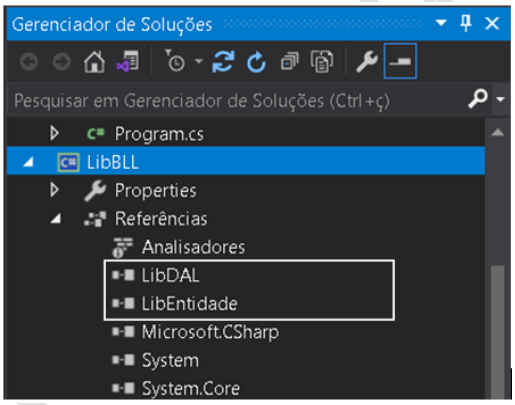
Vamos repetir o processo, agora para incluir os relacionamentos no projeto "LibBLL". Selecione o projeto "LibBLL" no "Gerenciador de Soluções", pressione o botão direito do mouse sobre este projeto → Adicionar → Referencia.

Selecione: "LibEntidade" e "LibDAL" .



Clique em "OK"

Observe no "Gerenciador de Soluções" a inclusão das referências no projeto "LibBLL":



IMPORTANTE descatar que a camada "LibEntidade", por ser uma camada de transferência de dados, NÃO faremos nenhuma inclusão de referência dentro dela, porém, ela é referenciada por todas as demais!!