

Atividade

Projeto Web LH-Pets

Nesta atividade você vai implementar um sistema web de cadastro de clientes, criar rotas e executar seu projeto em hospedagem local.

Para isso, você deve ter instalado e configurado em seu computador os seguintes softwares:

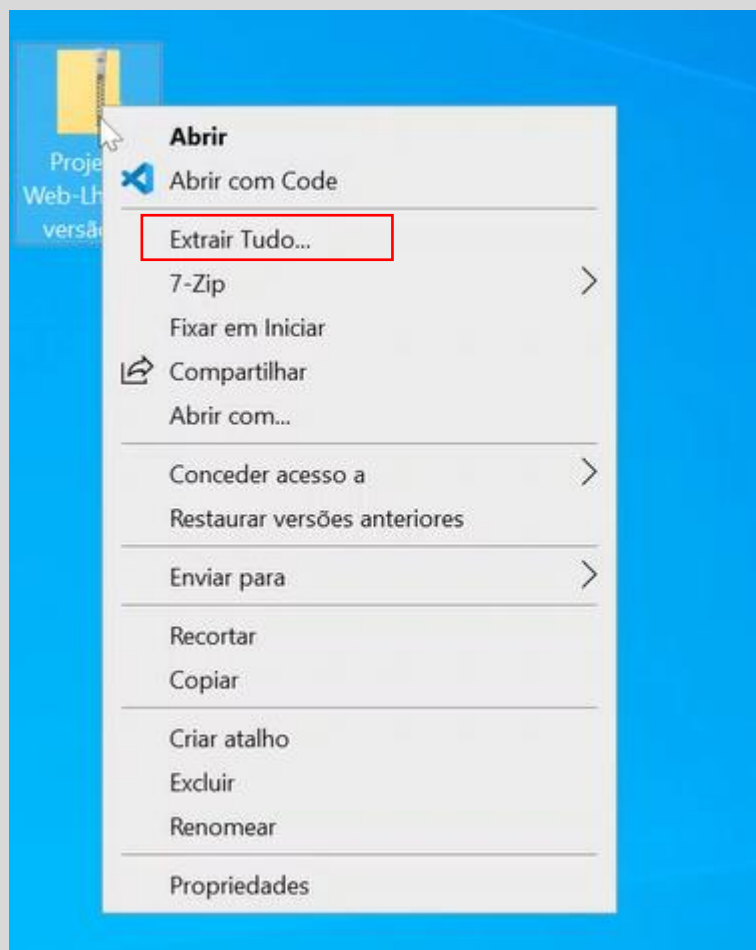
- SQL Server
- SQL Server Management Studio
- VSCode

Resumidamente, você completará as seguintes etapas:

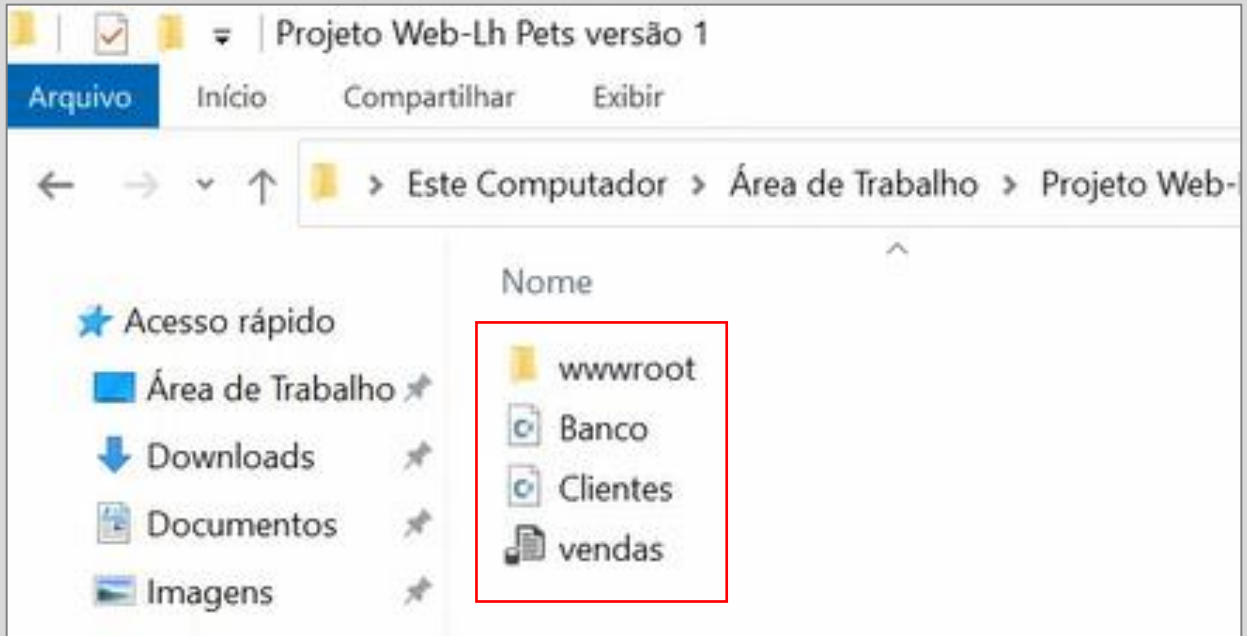
- Preparação do ambiente
- Criação do projeto
- Criação das rotas
- Criação do banco de dados
- Implementação do back-end
- Publicação em hospedagem local

Preparação do ambiente

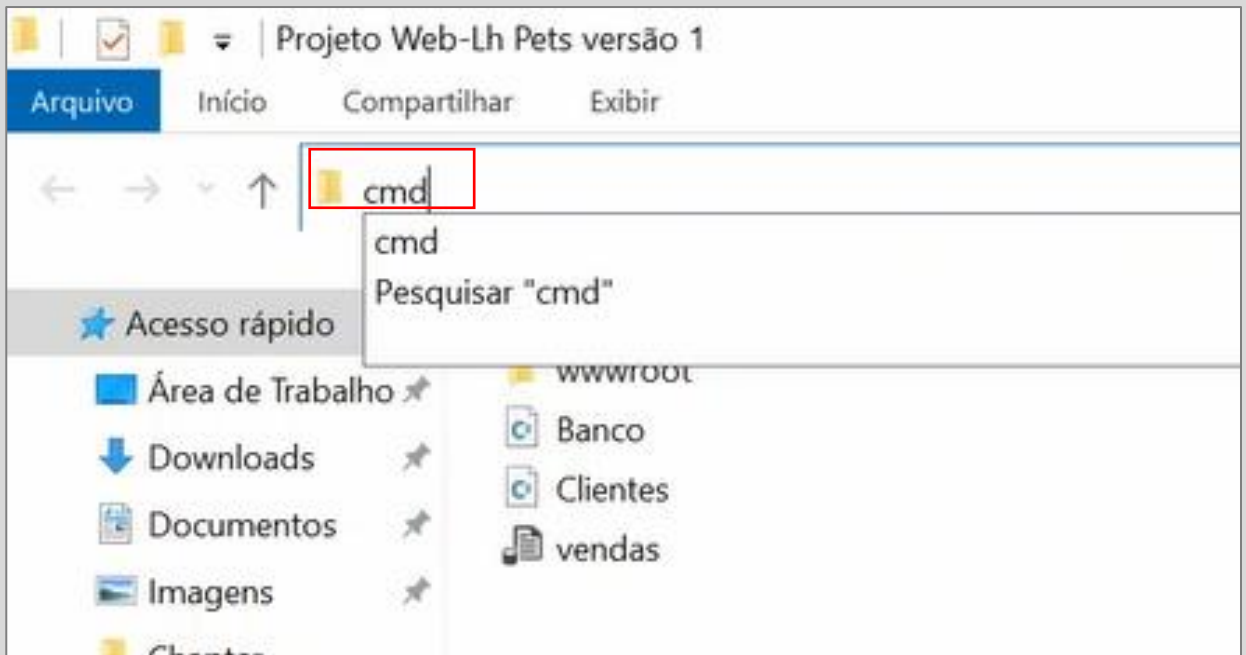
1. Baixe o zip anexo **Projeto Web-Lh Pets versão 1.zip**, clique com o botão direito sobre o arquivo baixado e selecione **Extrair tudo**.



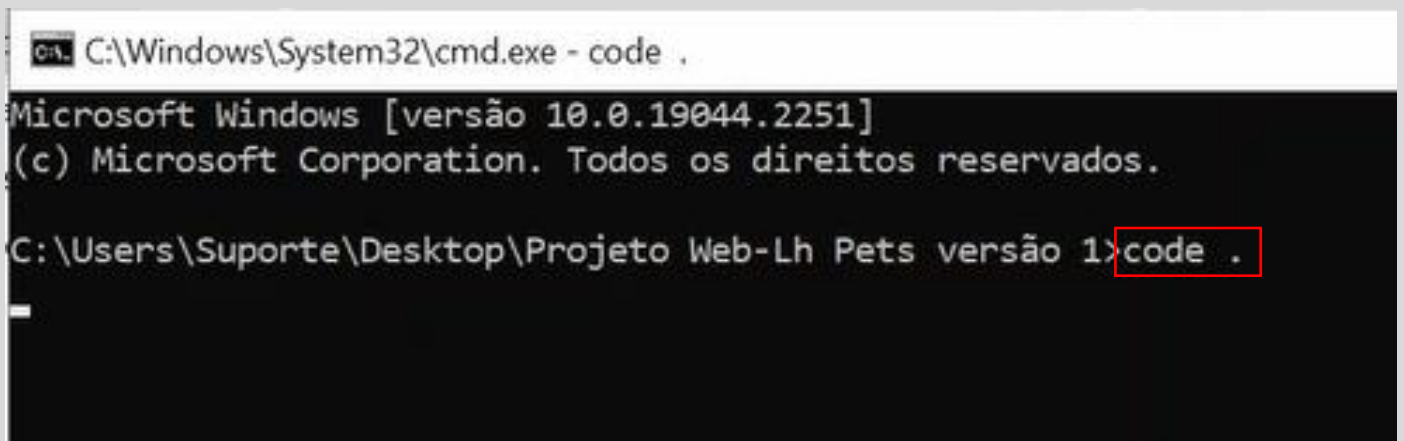
2. A pasta descompactada, chamada Projeto Web-Lh Pets versão 1, deve possuir uma subpasta wwwroot, um arquivo Banco.cs, um arquivo Clientes.cs e um script vendas.sql.



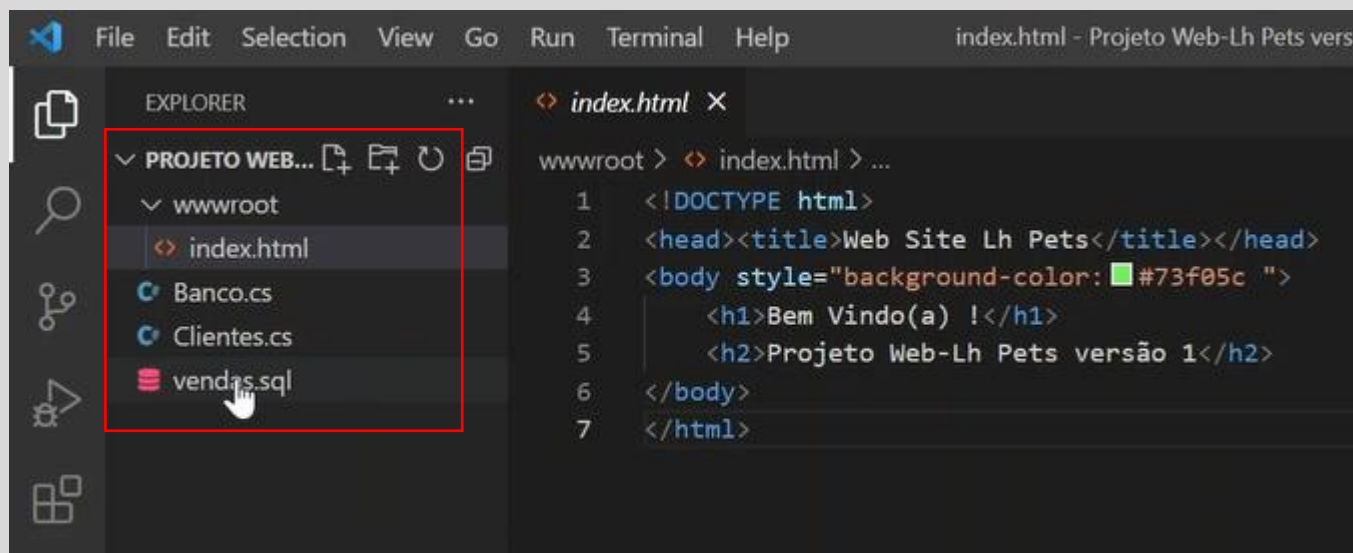
3. Na barra de navegação, digite `cmd` e dê Enter.



4. Um terminal será aberto. Nele, digite **code .** e dê Enter.



5. O VS Code abrirá com a estrutura da pasta do projeto.

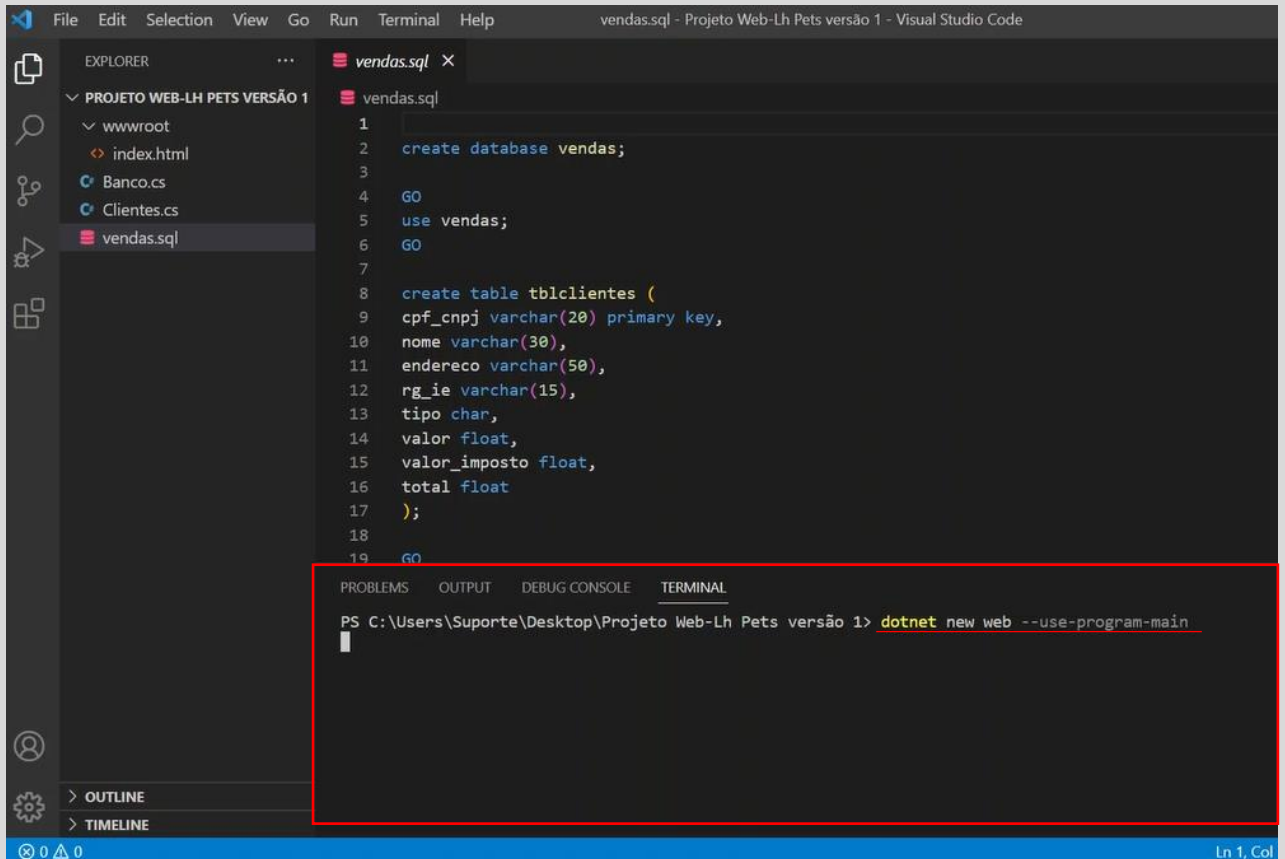


Dentro da pasta wwwroot, há um arquivo chamado index.html, uma página estática já pronta, com uma estrutura básica para a atividade.

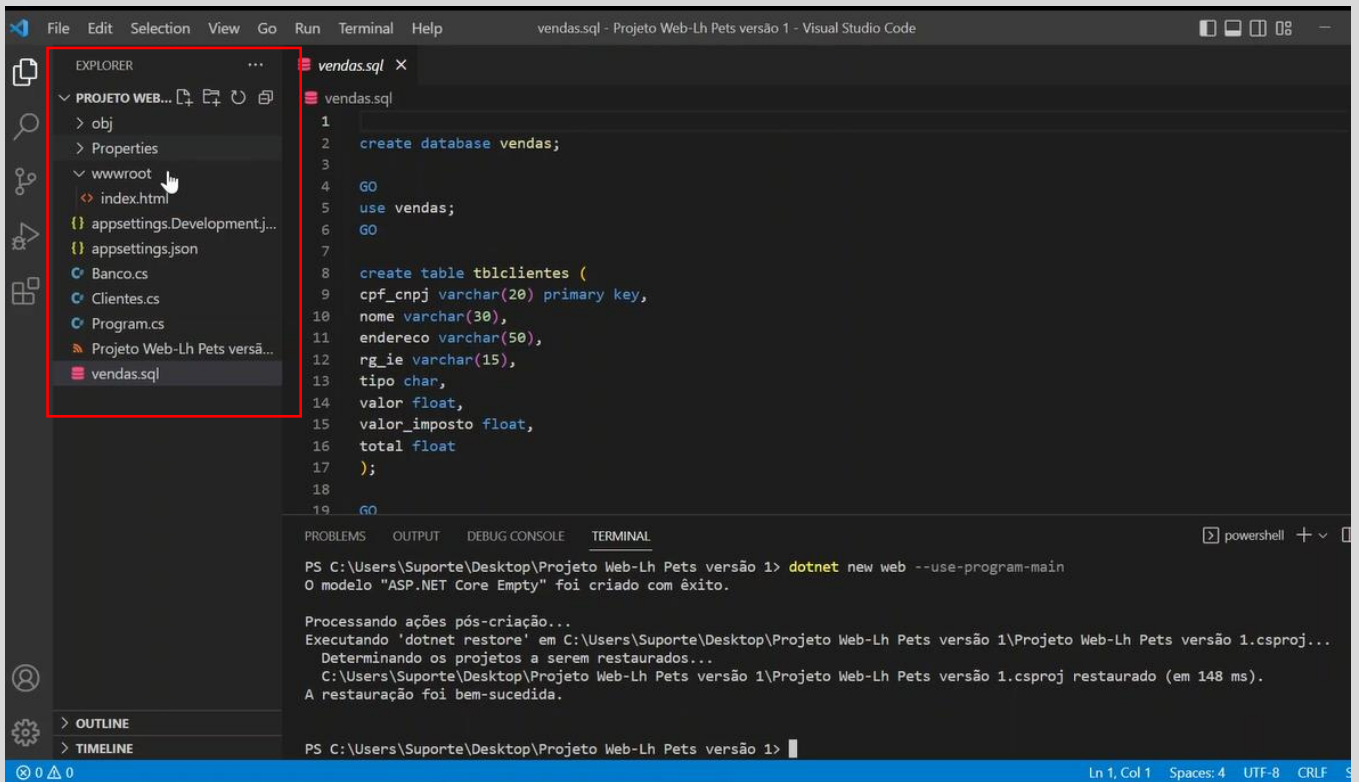
Os arquivos com extensão .cs são classes. Vamos implementar a classe Banco.cs, e a classe Clientes.cs será utilizada na classe Banco.cs

O arquivo vendas.sql é um script pronto, com dados básicos, para rodar e criar um banco de dados.

6. Use o atalho **ctrl +** para abrir o terminal e digite **dotnet new web --use-program-main** para criar a estrutura básica de um projeto web.

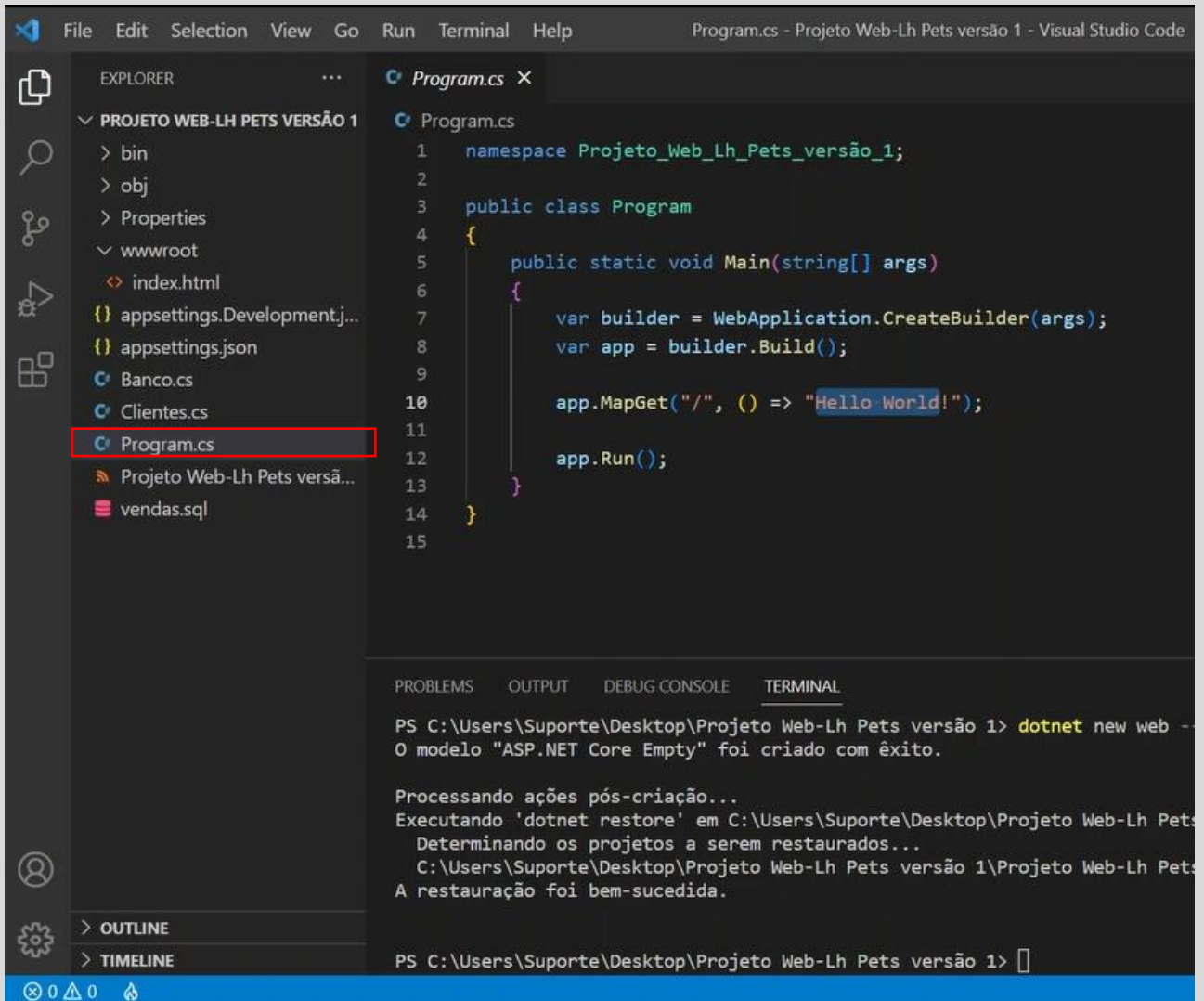


7. O VS Code carregará todos os pacotes necessários para o funcionamento do projeto web.



Desenvolvimento do projeto

1. O primeiro arquivo a ser modificado é o **Program.cs**.



```
File Edit Selection View Go Run Terminal Help
Program.cs - Projeto Web-Lh Pets versão 1 - Visual Studio Code

EXPLORER
PROJETO WEB-LH PETS VERSÃO 1
  bin
  obj
  Properties
  wwwroot
    index.html
  appsettings.Development.j...
  appsettings.json
  Banco.cs
  Clientes.cs
  Program.cs
  Projeto Web-Lh Pets versã...
  vendas.sql

Program.cs
1 namespace Projeto_Web_Lh_Pets_versão_1;
2
3 public class Program
4 {
5     public static void Main(string[] args)
6     {
7         var builder = WebApplication.CreateBuilder(args);
8         var app = builder.Build();
9
10        app.MapGet("/", () => "Hello World!");
11
12        app.Run();
13    }
14 }
15

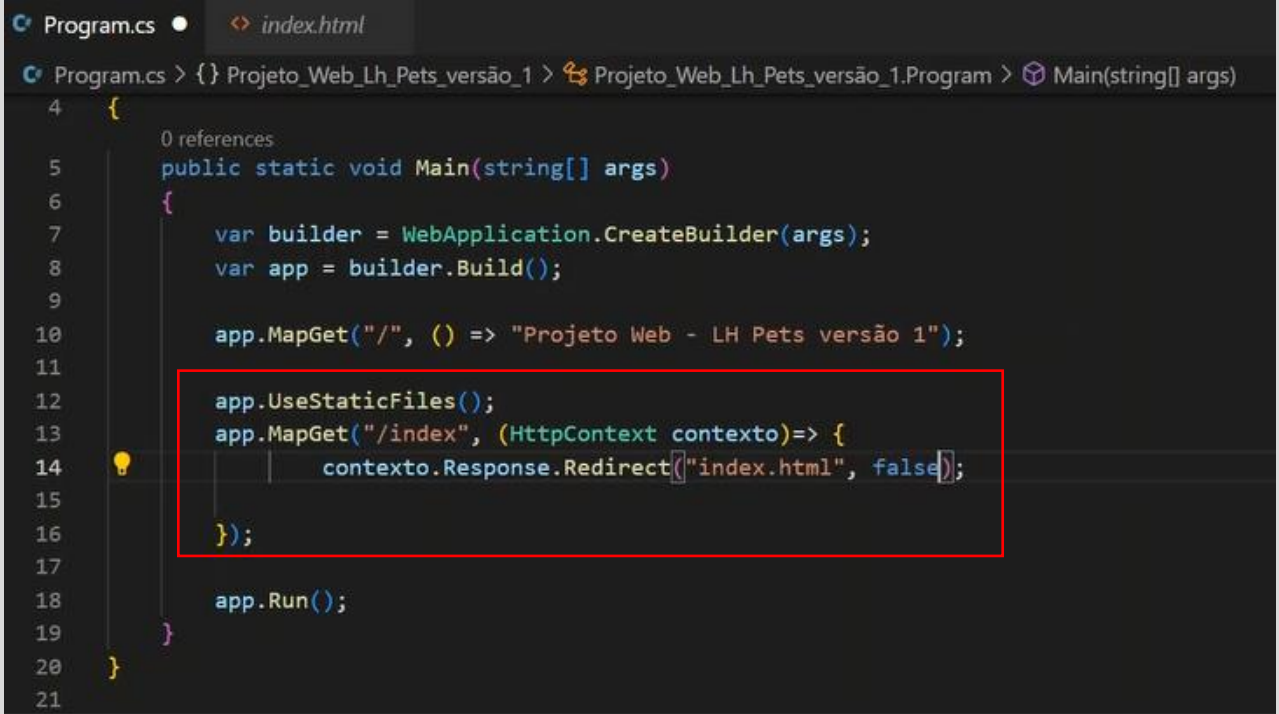
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1> dotnet new web --
O modelo "ASP.NET Core Empty" foi criado com êxito.

Processando ações pós-criação...
Executando 'dotnet restore' em C:\Users\Suporte\Desktop\Projeto Web-Lh Pets
Determinando os projetos a serem restaurados...
C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1\Projeto Web-Lh Pets
A restauração foi bem-sucedida.

PS C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1>
```

Na linha 10, troque **Hello World!** por **Projeto Web – LH Pets versão 1**. Salve a alteração com o atalho Ctrl + s.

2. Para que usuários externos possam solicitar os arquivos armazenados dentro da pasta wwwroot, devemos habilitar seu uso e mapear a sua rota.



```
Program.cs > {} Projeto_Web_Lh_Pets_versão_1 > Projeto_Web_Lh_Pets_versão_1.Program > Main(string[] args)
4 {
5     0 references
6     public static void Main(string[] args)
7     {
8         var builder = WebApplication.CreateBuilder(args);
9         var app = builder.Build();
10
11         app.MapGet("/", () => "Projeto Web - LH Pets versão 1");
12
13         app.UseStaticFiles();
14         app.MapGet("/index", (HttpContext contexto)=> {
15             contexto.Response.Redirect("index.html", false);
16         });
17
18         app.Run();
19     }
20 }
21
```

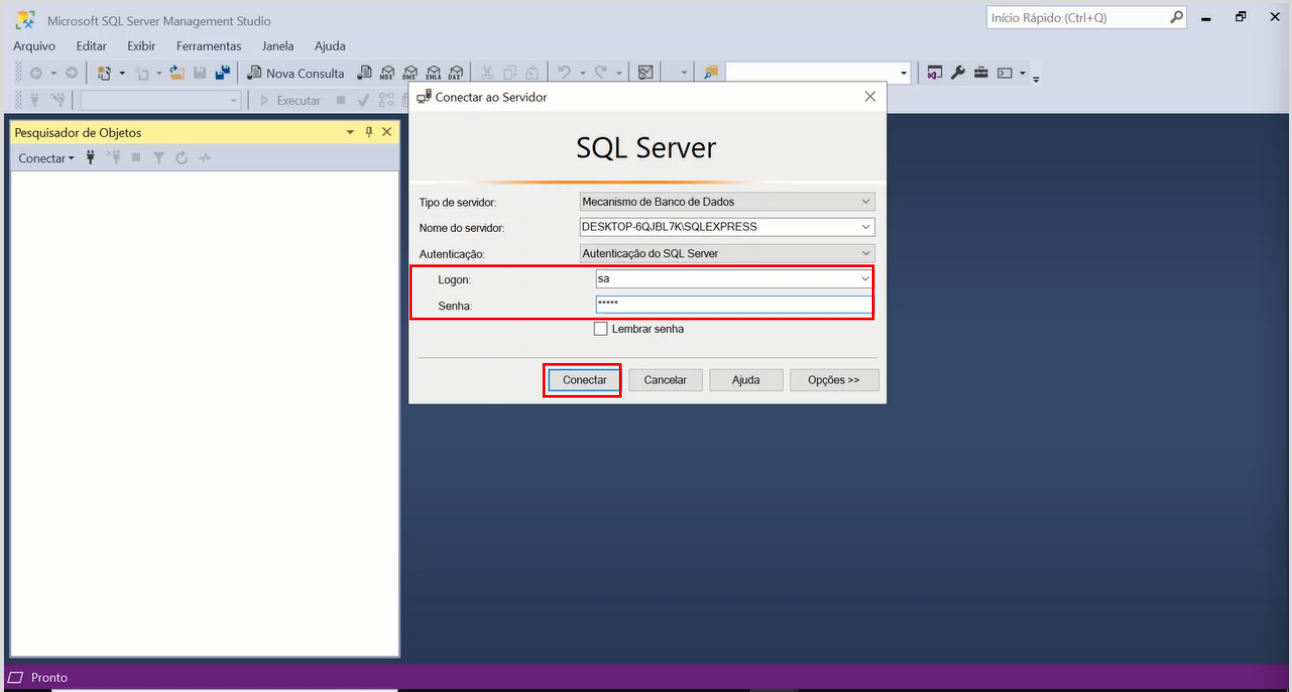
Digite `app.UseStaticFiles();` para habilitar o uso das páginas estáticas, como mostrado na linha 12.

Digite:

```
app.UseStaticFiles();
app.MapGet("/index", (HttpContext contexto)=> {
    contexto.Response.Redirect("index.html", false);
});
```

Assim, você criará a rota `/index`, que redirecionará o usuário para a página desejada (`index.html`).

3. A próxima etapa é inserir o script vendas.sql no SSMS. Abra o SSMS (SQL Server Management Studio) e use **sa** como login, a senha **12345** e clique em **Conectar**.

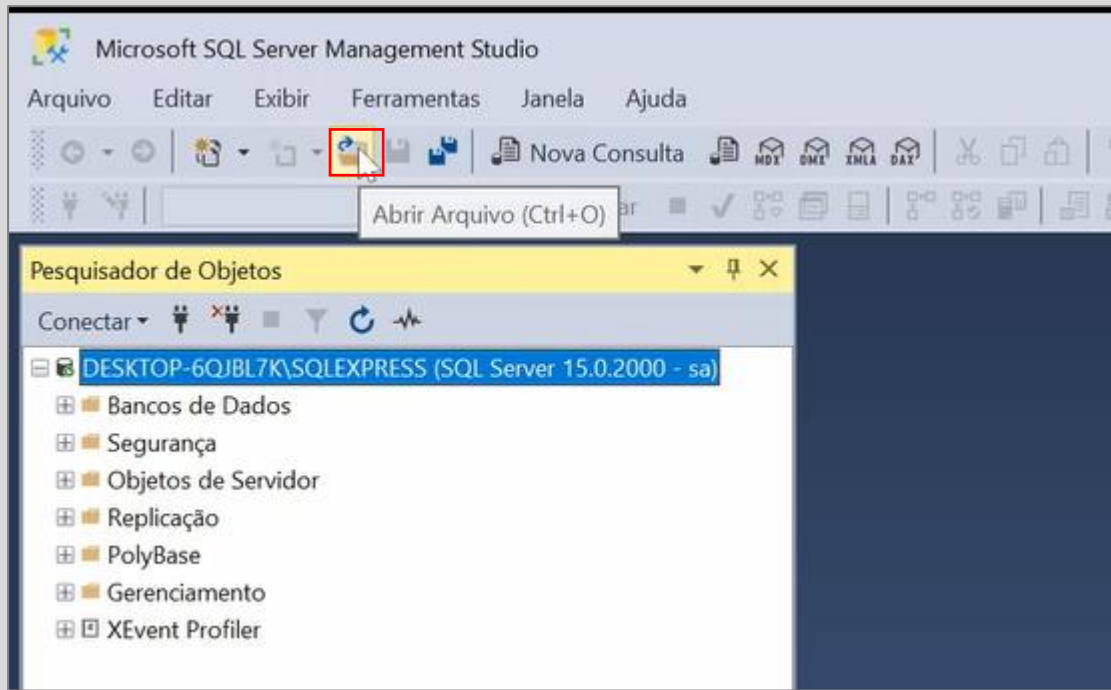


Importante

As credenciais (login e senha) são muito importantes, pois serão usadas no arquivo Banco.cs.

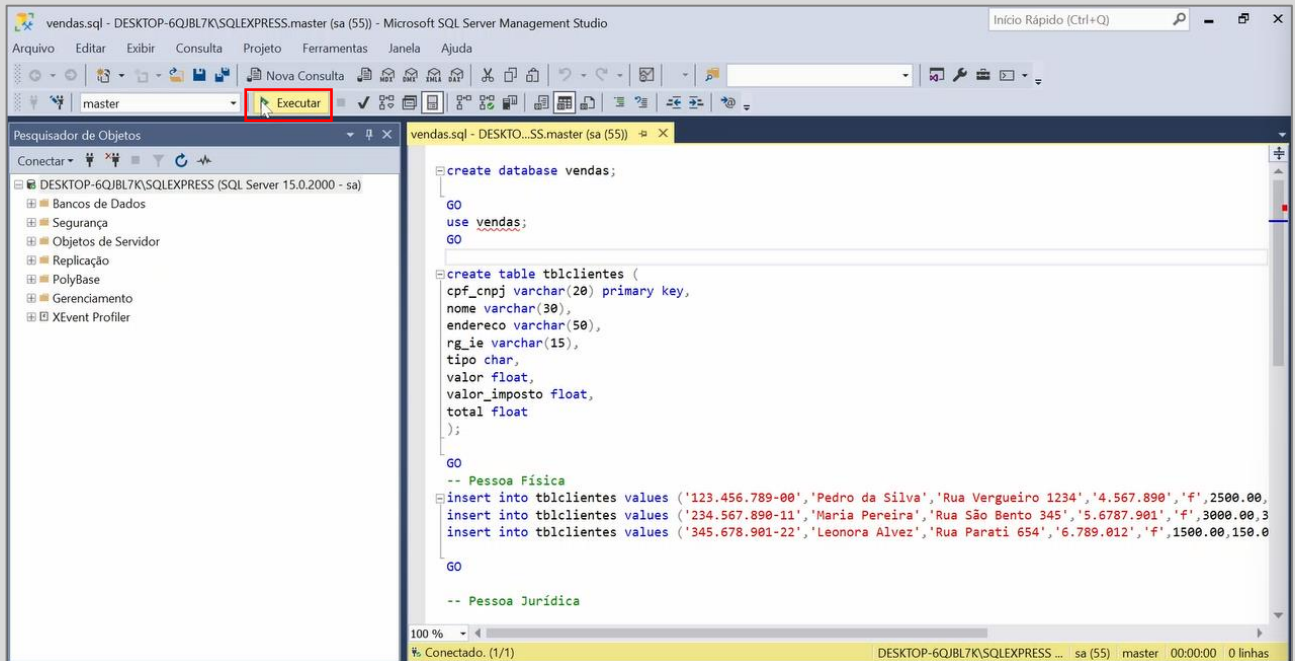


4. Conexão completada, abra o arquivo vendas.sql, clicando em **Abrir Arquivo**.

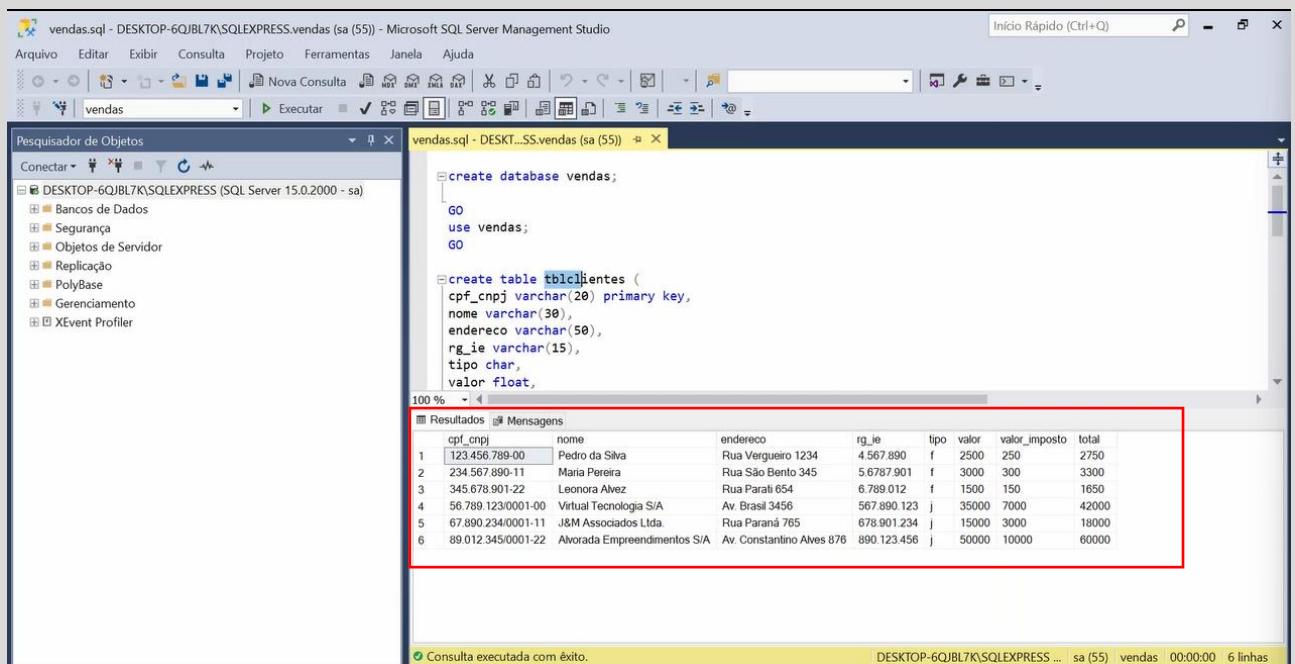


Em seguida, encontre o arquivo vendas.sql dentro da sua pasta e dê um duplo clique.

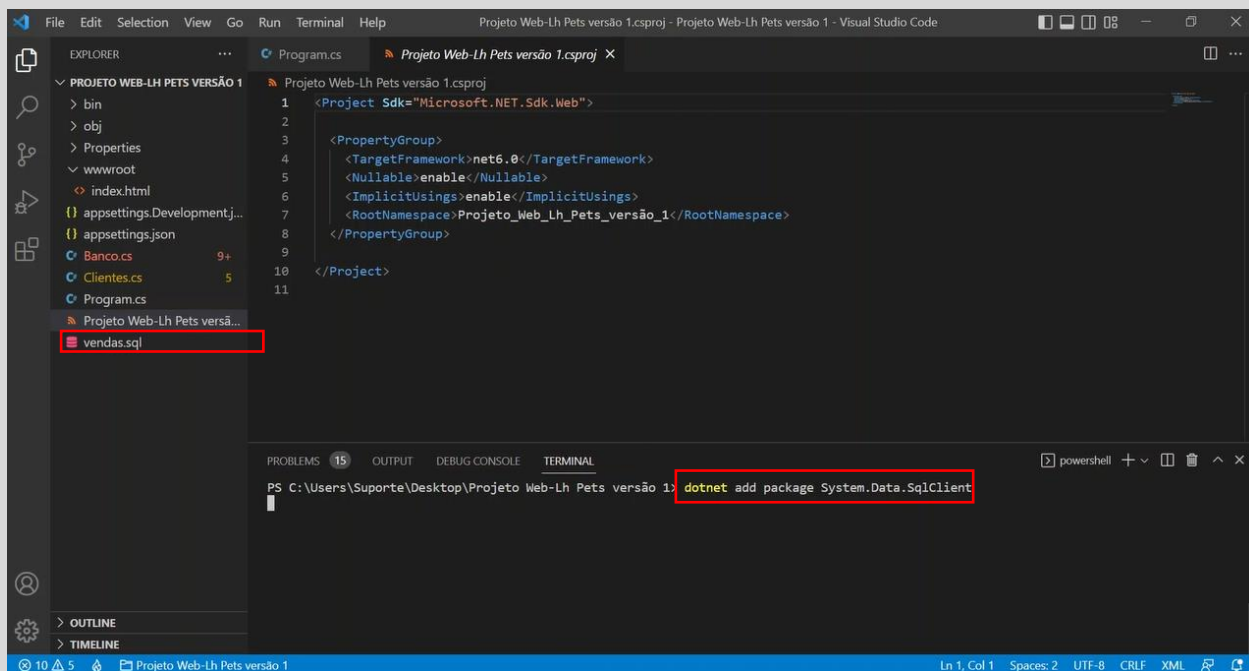
5. O script será carregado. Clique em **Executar**.



6. O banco de dados **vendas** será criado, junto com a tabela **tblclientes**. Os dados inseridos serão mostrados na tabela.



7. Agora volte para o VS Code e abra o arquivo Projeto Web-Lh Pets versão 1.csproj para instalar o conector com o banco de dados.



Digite no terminal **dotnet add package System.Data.SqlClient** e dê **Enter**. O sistema baixará todos os pacotes necessários para instalar o conector.

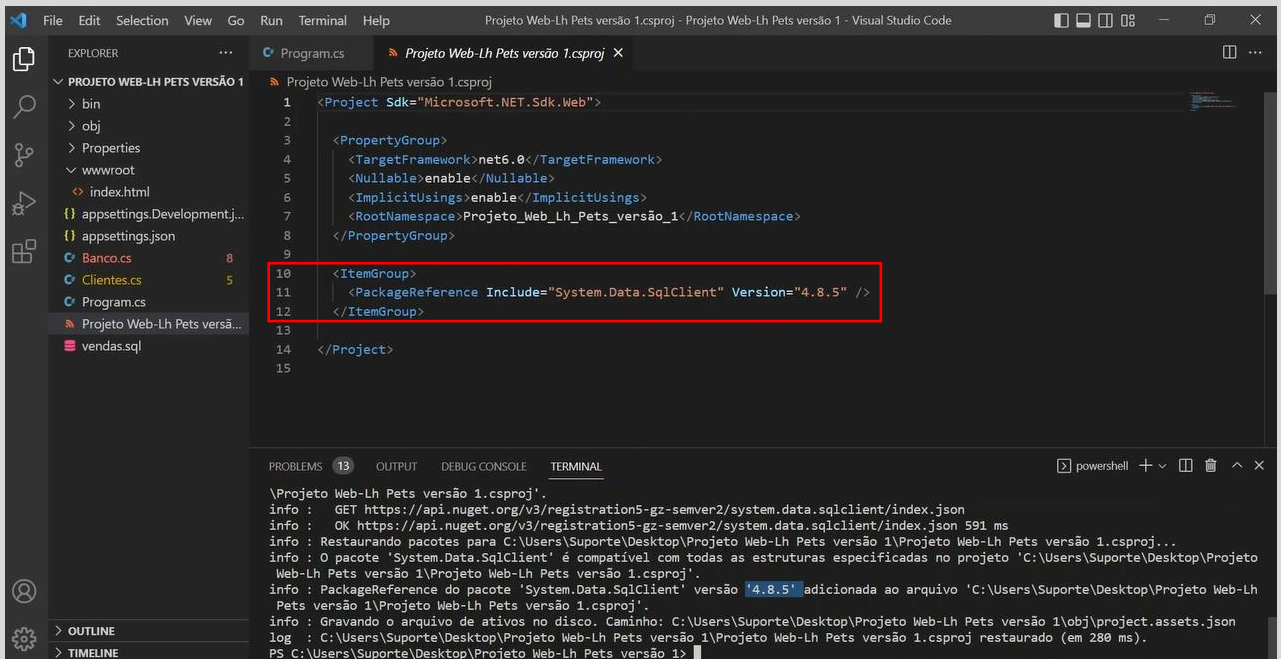
Importante

Você deve estar conectado(a) à internet para executar essa etapa.



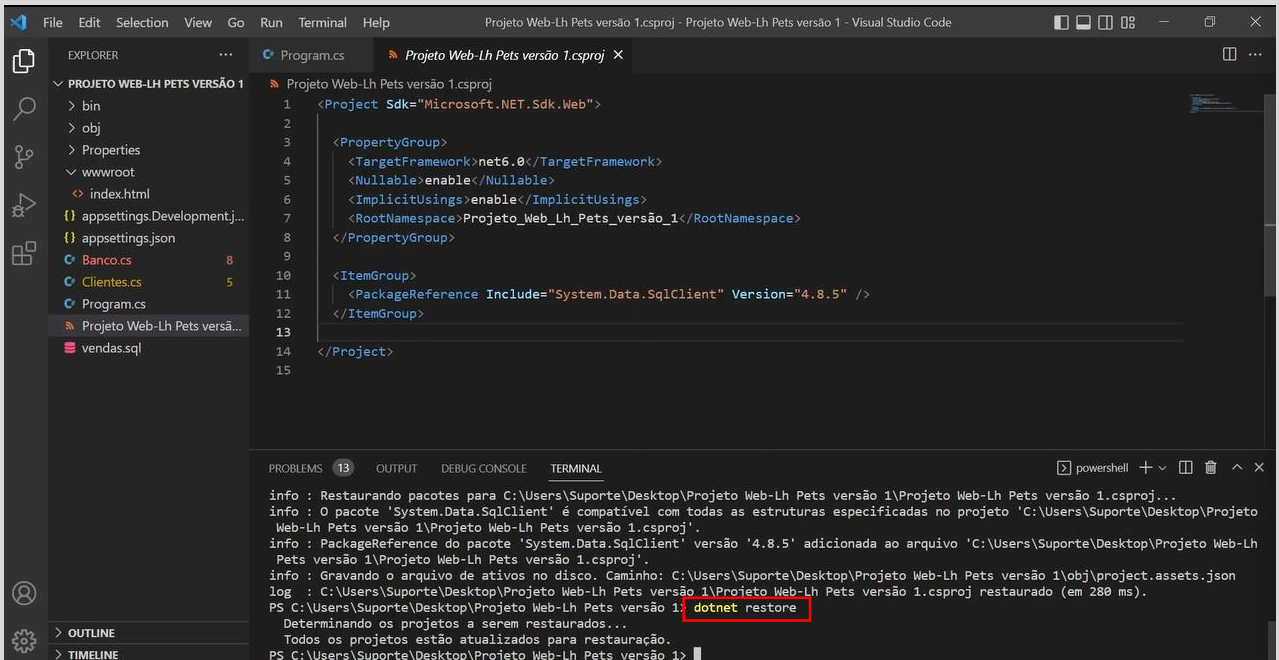
8. A referência do conector será inserida no arquivo, como mostram as linhas de 10 a 12.

```
<ItemGroup>
  <PackageReference Include="System.Data.SqlClient"
Version="4.8.5" />
</ItemGroup>
```



Agora com o conector instalado, basta referenciá-lo para que ele funcione.

9. Faça a atualização do projeto, digitando dotnet restore no terminal.



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left displays the project structure for 'PROJETO WEB-LH PETS VERSÃO 1', including folders like 'bin', 'obj', 'Properties', and 'wwwroot', and files like 'index.html', 'appsettings.json', 'Banco.cs', 'Clientes.cs', 'Program.cs', and 'vendas.sql'. The main editor area shows the 'Projeto Web-Lh Pets versão 1.csproj' file with the following XML content:

```
1 <Project Sdk="Microsoft.NET.Sdk.Web">
2
3
4 <PropertyGroup>
5   <TargetFramework>net6.0</TargetFramework>
6   <Nullable>enable</Nullable>
7   <ImplicitUsings>enable</ImplicitUsings>
8   <RootNamespace>Projeto_Web_Lh_Pets_versão_1</RootNamespace>
9 </PropertyGroup>
10
11 <ItemGroup>
12   <PackageReference Include="System.Data.SqlClient" Version="4.8.5" />
13 </ItemGroup>
14
15 </Project>
```

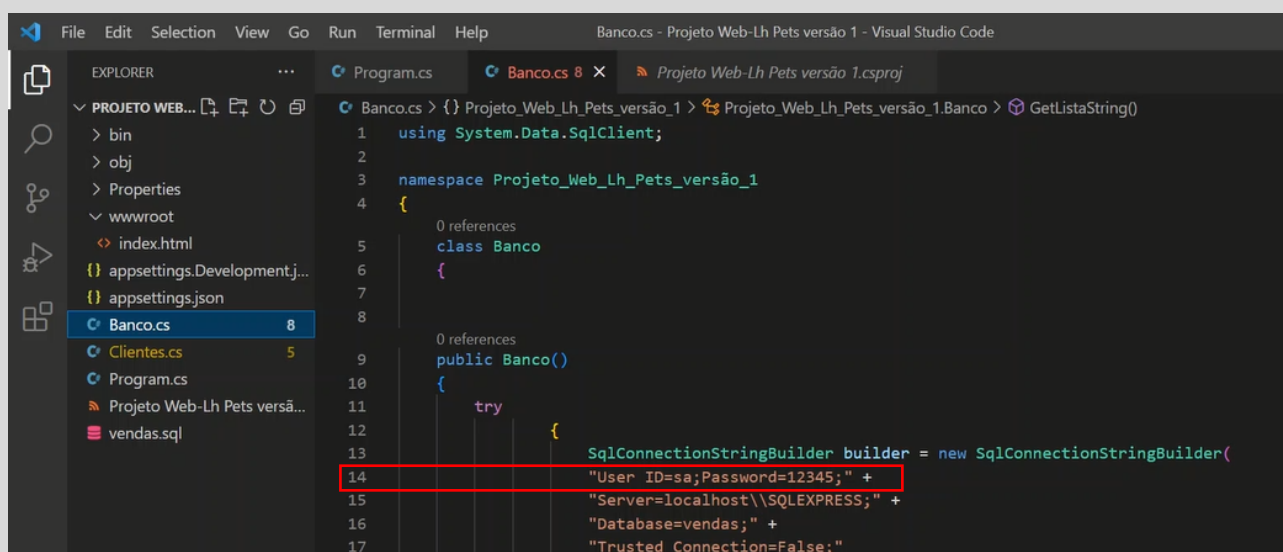
The bottom panel shows the 'TERMINAL' tab with the following output:

```
info : Restaurando pacotes para C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1\Projeto Web-Lh Pets versão 1.csproj...
info : O pacote 'System.Data.SqlClient' é compatível com todas as estruturas especificadas no projeto 'C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1\Projeto Web-Lh Pets versão 1.csproj'.
info : PackageReference do pacote 'System.Data.SqlClient' versão '4.8.5' adicionada ao arquivo 'C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1\Projeto Web-Lh Pets versão 1.csproj'.
info : Gravando o arquivo de ativos no disco. Caminho: C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1\obj\project.assets.json
log : C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1\Projeto Web-Lh Pets versão 1.csproj restaurado (em 280 ms).
PS C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1> dotnet restore
Determinando os projetos a serem restaurados...
Todos os projetos estão atualizados para restauração.
PS C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1>
```

A atualização do pacote que fará a conexão com o banco de dados estará pronta.

Conexão com o banco de dados

1. Abra o arquivo Banco.cs e verifique se o login e a senha são os mesmos usados no SSMS. No nosso exemplo, o login é **sa** e a senha **12345**.



```
1 using System.Data.SqlClient;
2
3 namespace Projeto_Web_Lh_Pets_versão_1
4 {
5     0 references
6     class Banco
7     {
8
9     0 references
10    public Banco()
11    {
12        try
13        {
14            SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder(
15                "User ID=sa;Password=12345;" +
16                "Server=localhost\\SQLEXPRESS;" +
17                "Database=vendas;" +
18                "Trusted_Connection=False;"
```

Importante

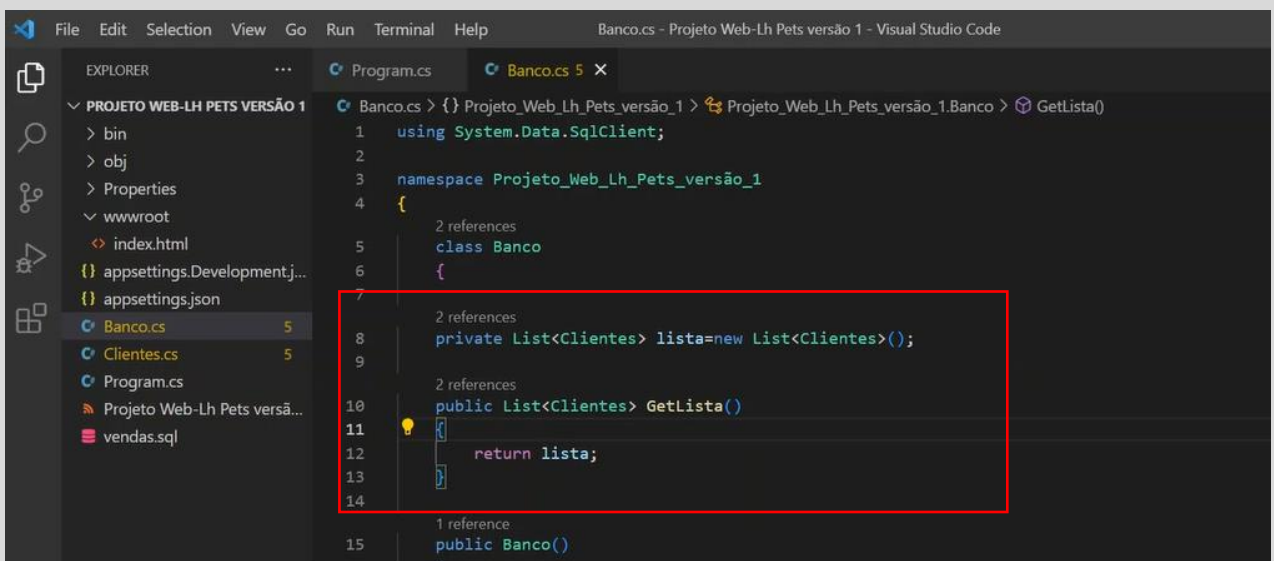
Usamos o login **sa**, pois estamos em hospedagem local. Se o banco for executado em rede, é necessário usar outro tipo de login.



2. Ainda em Banco.cs, digite o seguinte dentro da classe Banco:

```
private List<Clientes> lista=new List<Clientes>();

public List<Clientes> GetLista()
{
    return lista;
}
```



Nesse bloco, criamos e instanciamos uma lista, chamada **lista**, com os dados da classe **Clientes**. As classes **Banco** e **Clientes** estão dentro do mesmo **namespace** **Projeto_Web_Lh_Pets_versão_1**.

Dentro do método construtor **public Banco()** da classe **Banco**, há comandos para abrir as conexões com o banco de dados.

A seguir, acompanhe o bloco de código que recebe os dados da tabela e insere na lista de clientes.

```
while(tabela.Read())
{
    lista.Add(new Clientes()
    {
        cpf_cnpj = tabela["cpf_cnpj"].ToString(),
        nome = tabela["nome"].ToString(),
        endereco = tabela["endereco"].ToString(),
        rg_ie = tabela["rg_ie"].ToString(),
        tipo = tabela["tipo"].ToString(),
        valor = (float)Convert.ToDecimal(tabela["valor"]),
        valor_imposto = (float)Convert.ToDecimal(tabela["valor_imposto"]),
        total = (float)Convert.ToDecimal(tabela["total"])
    });
}
```

Na linha 29, há um laço while para montar uma tabela. Então, enquanto houver dados na classe Clientes, o laço vai montar uma lista.

Na linha 37, a variável lista vai receber novos objetos, chamados Clientes, com os atributos carregados do banco de dados.

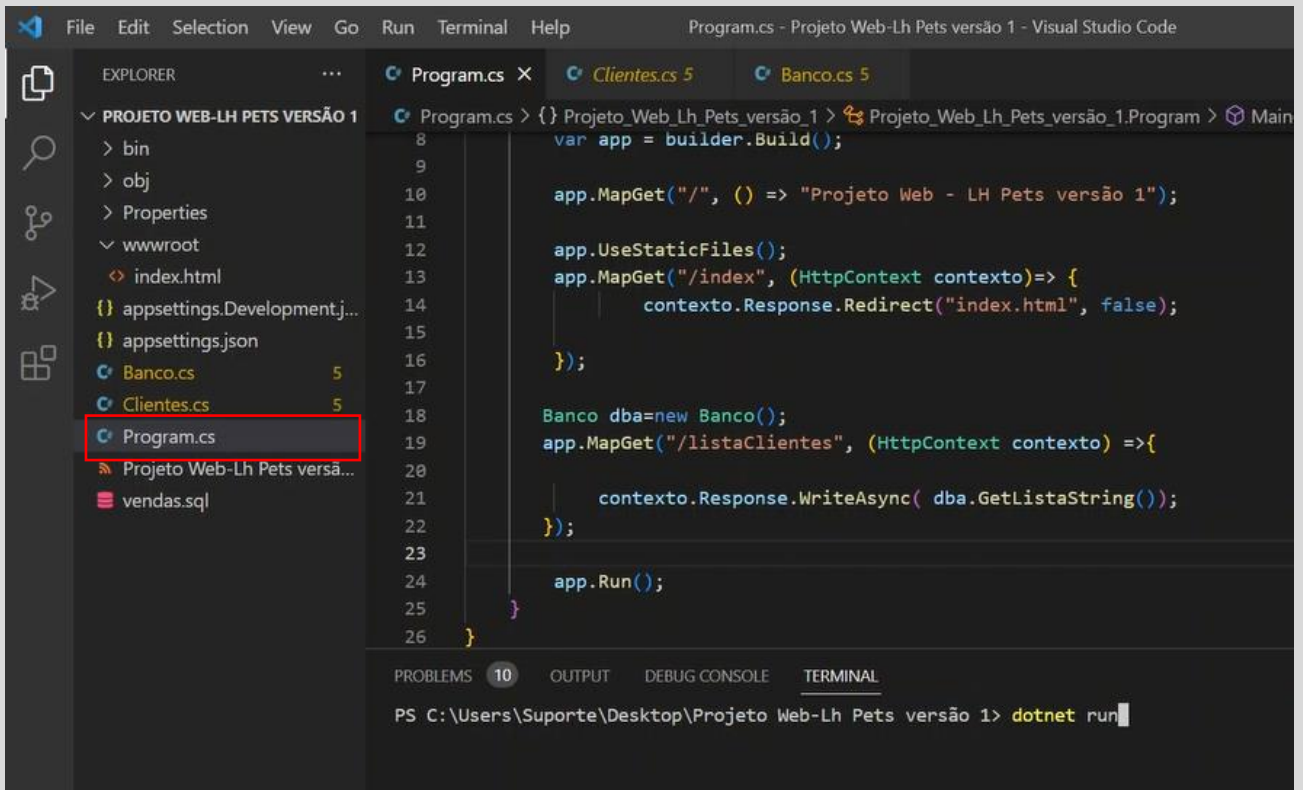
Com o comando `GetListaString()`, na linha 61, os dados da lista são montados dentro de um HTML. Esse método constrói uma página HTML através do back-end.

A função `imprimirListaConsole()`, na linha 91, vai imprimir dentro do servidor, ou seja, o cliente não tem acesso.

Então, a lista de clientes é criada através da conexão com o banco de dados, ou seja, a lista carrega os dados do banco. Em seguida, essa lista é mostrada em HTML ou no console.

3. O próximo passo é criar a rota para a lista criada. Acesse o arquivo **Program.cs** e abra a rota digitando:

```
Banco dba=new Banco();  
app.MapGet("/listaClientes", (HttpContext contexto) =>{  
    contexto.Response.WriteAsync( dba.GetListaString());  
});
```



Em seguida, vamos abrir a rota **/listaClientes** com o método **MapGet**.

Usaremos novamente o método **HttpContext** para escrever o que vem do banco para a página dentro da variável **contexto**.

O método **WriteAsync** escreve na página que está sendo aberta. Em nosso exemplo, o retorno vai ser uma lista com os dados do banco **dba** já dentro da estrutura HTML (**GetListaString**).

Dica!

Não se esqueça de salvar a cada alteração feita.



4. Para trazer o conteúdo pedido para a página, digite **dotnet run** no terminal do **Program.cs**.

The screenshot shows the Visual Studio Code interface. On the left, the Explorer pane shows the file structure of 'PROJETO WEB-LH PETS VERSÃO 1'. The file 'Program.cs' is selected and highlighted with a red box. The main editor displays the code in 'Program.cs', which includes a MapGet method for '/listaClientes' that calls 'contexto.Response.WriteAsync(dba.GetListaString())'. The terminal at the bottom, also highlighted with a red box, shows the command 'dotnet run' being executed in the project directory.

```
File Edit Selection View Go Run Terminal Help
Program.cs - Projeto Web-Lh Pets versão 1 - Visual Studio Code

EXPLORER
PROJETO WEB-LH PETS VERSÃO 1
  bin
  obj
  Properties
  wwwroot
    index.html
  appsettings.Development.j...
  appsettings.json
  Banco.cs
  Clientes.cs
  Program.cs
  Projeto Web-Lh Pets versã...
  vendas.sql

Program.cs
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26

var app = builder.Build();

app.MapGet("/", () => "Projeto Web - LH Pets versão 1");

app.UseStaticFiles();
app.MapGet("/index", (HttpContext contexto)=> {
    contexto.Response.Redirect("index.html", false);
});

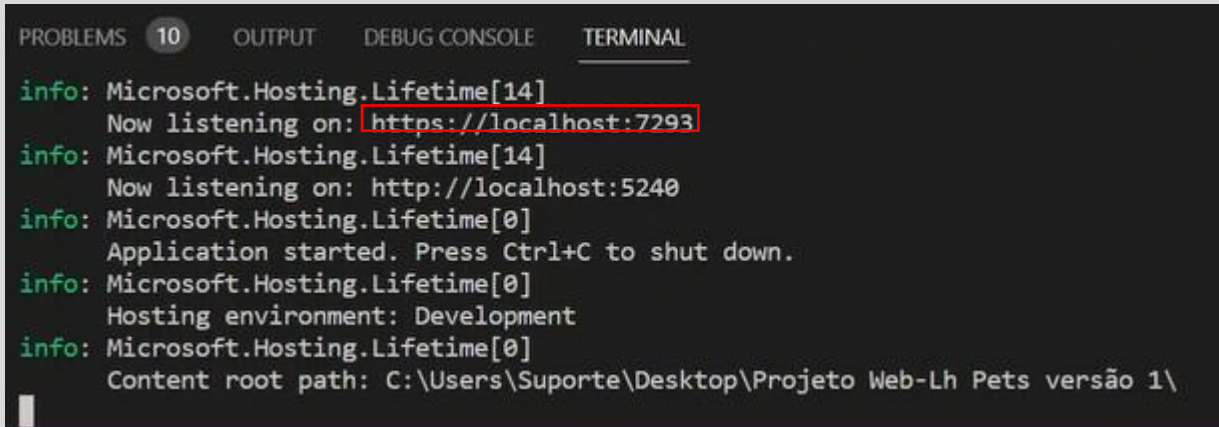
Banco dba=new Banco();
app.MapGet("/listaClientes", (HttpContext contexto) =>{
    contexto.Response.WriteAsync( dba.GetListaString());
});

app.Run();
}
```

PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL

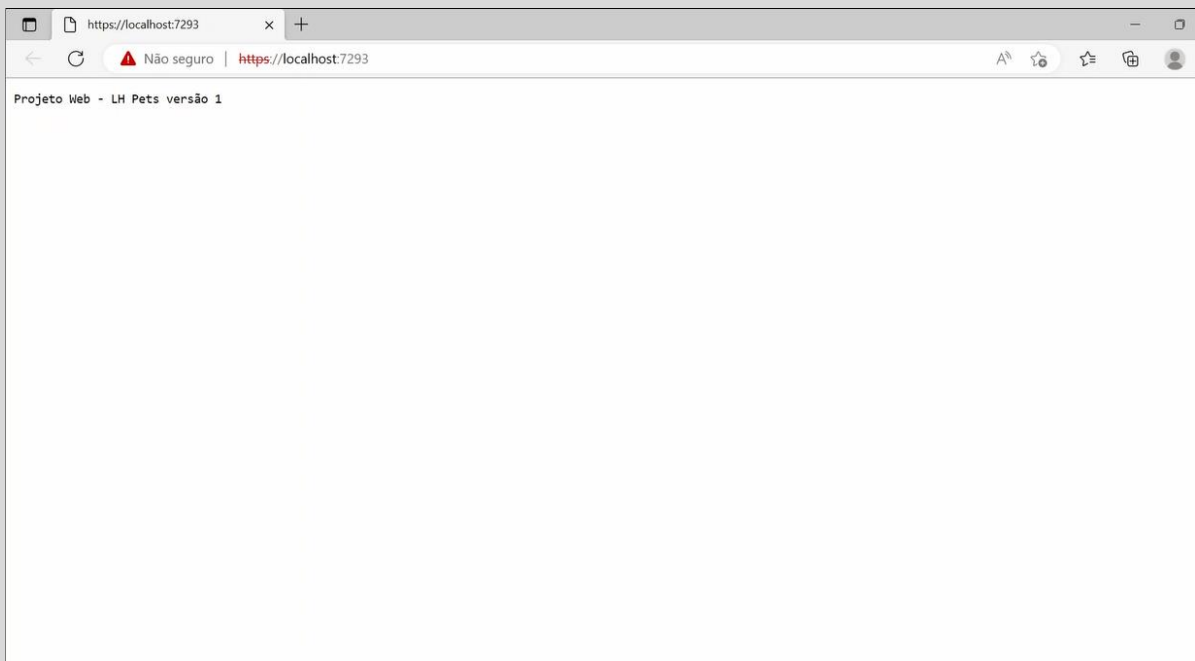
PS C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1> dotnet run

5. Aguarde a compilação completa e abra o primeiro link em seu navegador.

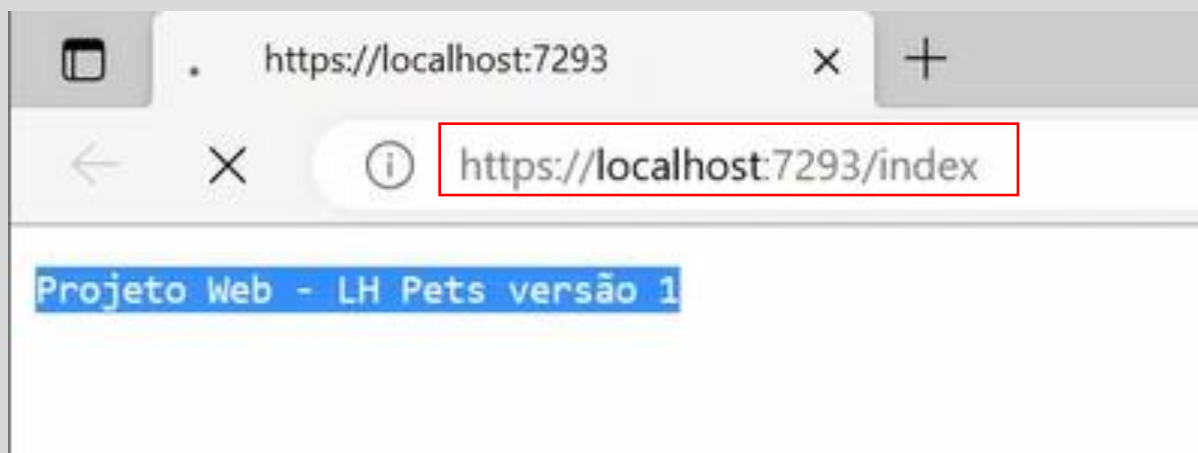


```
info: Microsoft.Hosting.Lifetime[14]
  Now listening on: https://localhost:7293
info: Microsoft.Hosting.Lifetime[14]
  Now listening on: http://localhost:5240
info: Microsoft.Hosting.Lifetime[0]
  Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
  Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
  Content root path: C:\Users\Suporte\Desktop\Projeto Web-Lh Pets versão 1\
```

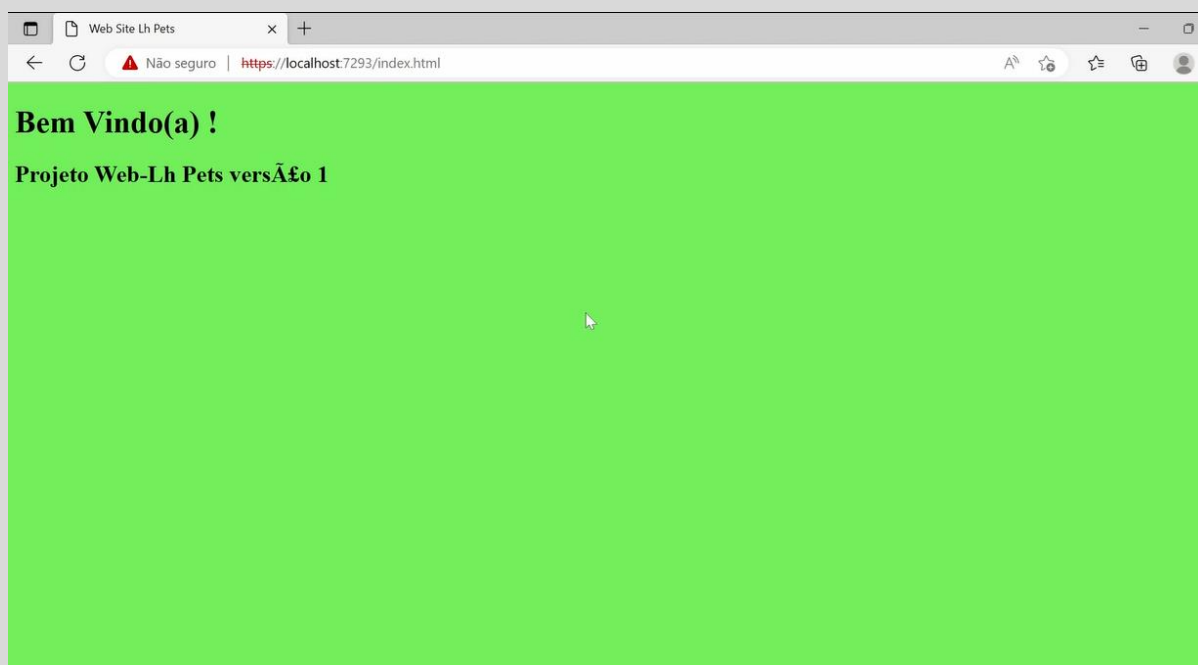
A renderização no navegador deve ser apenas o nome da pasta do projeto.



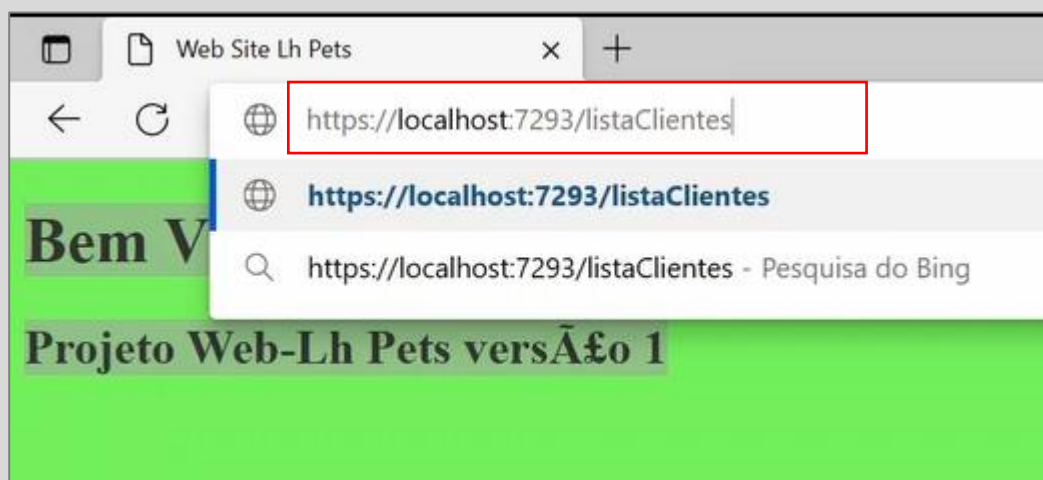
6. Na barra de endereços do navegador, digite a rota **/index** no final do endereço e dê **Enter**.



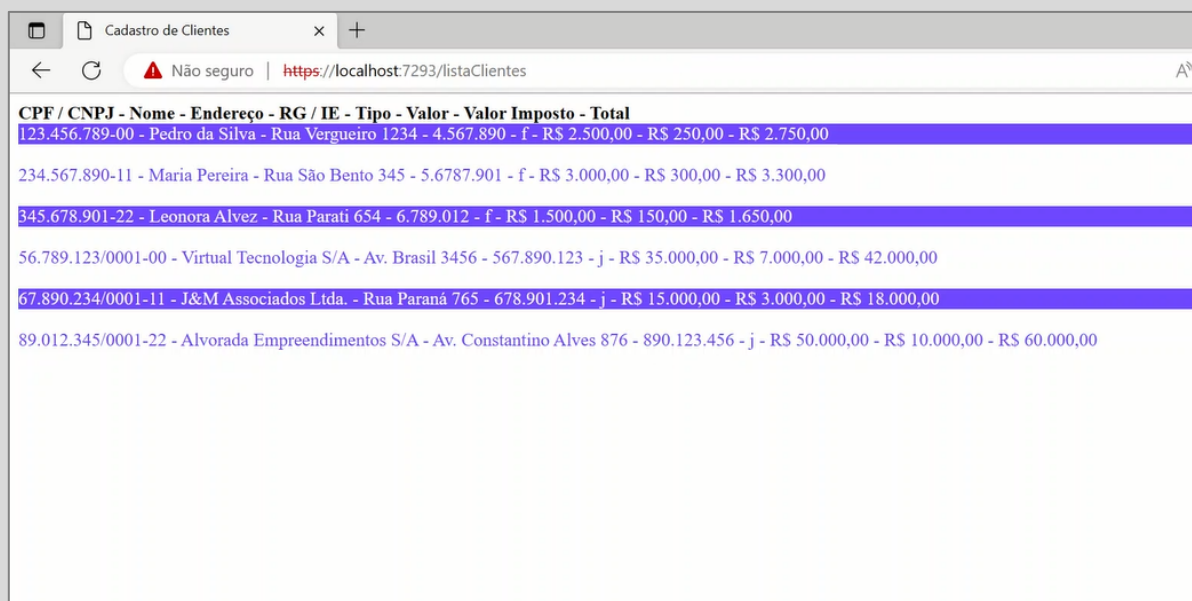
A renderização no navegador deve mostrar a página estática dentro da pasta wwwroot.



7. Agora digite a rota /listaClientes na barra de endereços do navegador e dê **Enter**.

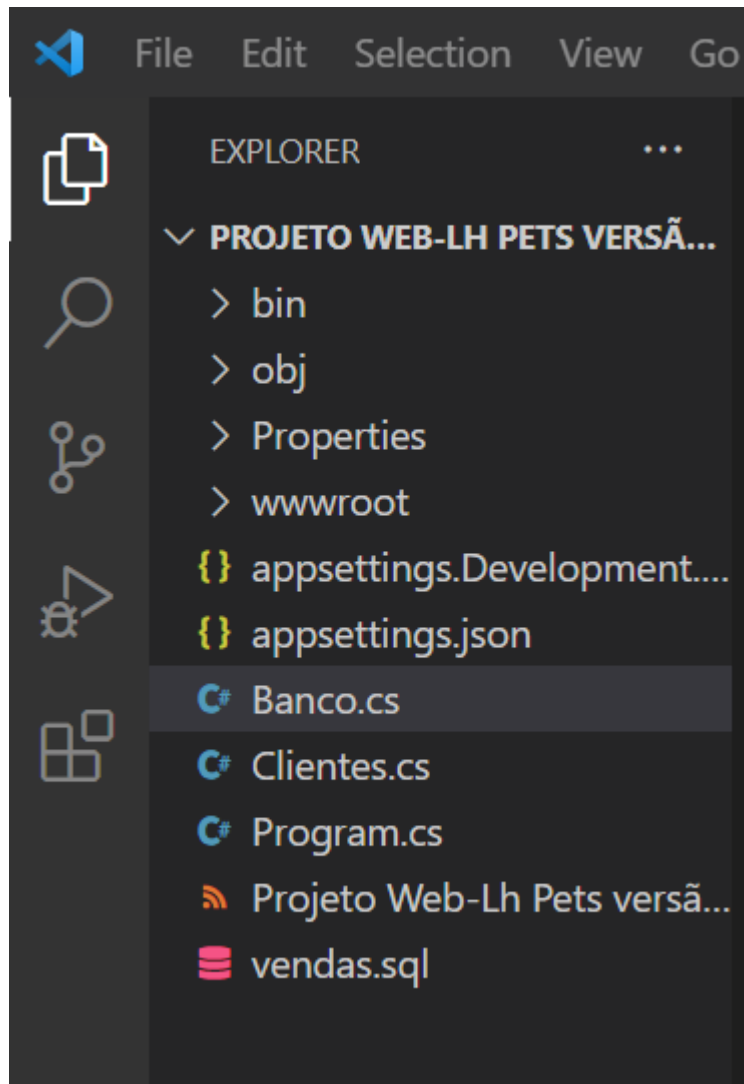


A renderização no navegador deve mostrar a página criada através do back-end, com os dados carregados do banco de dados.



| CPF / CNPJ | Nome | Endereço | RG / IE | Tipo | Valor | Valor Imposto | Total |
|--------------------|------------------------------|---------------------------|-------------|------|---------------|---------------|---------------|
| 123.456.789-00 | Pedro da Silva | Rua Vergueiro 1234 | 4.567.890 | f | R\$ 2.500,00 | R\$ 250,00 | R\$ 2.750,00 |
| 234.567.890-11 | Maria Pereira | Rua São Bento 345 | 5.678.901 | f | R\$ 3.000,00 | R\$ 300,00 | R\$ 3.300,00 |
| 345.678.901-22 | Leonora Alvez | Rua Parati 654 | 6.789.012 | f | R\$ 1.500,00 | R\$ 150,00 | R\$ 1.650,00 |
| 56.789.123/0001-00 | Virtual Tecnologia S/A | Av. Brasil 3456 | 567.890.123 | j | R\$ 35.000,00 | R\$ 7.000,00 | R\$ 42.000,00 |
| 67.890.234/0001-11 | J&M Associados Ltda. | Rua Paraná 765 | 678.901.234 | j | R\$ 15.000,00 | R\$ 3.000,00 | R\$ 18.000,00 |
| 89.012.345/0001-22 | Alvorada Empreendimentos S/A | Av. Constantino Alves 876 | 890.123.456 | j | R\$ 50.000,00 | R\$ 10.000,00 | R\$ 60.000,00 |

Confira a seguir o print do projeto completo e os códigos dos arquivos necessários.



Banco.cs

```

using System.Data.SqlClient;

namespace Projeto_Web_Lh_Pets_versão_1
{
    class Banco
    {
        private List<Clientes> lista=new List<Clientes>();

        public List<Clientes> GetLista()
        {
            return lista;
        }

        public Banco()
        {
            try
            {
                SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder(
                    "User ID=sa;Password=12345;" +
                    "Server=localhost\\SQLEXPRESS;" +
                    "Database=vendas;" +
                    "Trusted_Connection=False;"
                );

                using (SqlConnection conexao = new
                SqlConnection(builder.ConnectionString))
                {
                    String sql = "SELECT * FROM tblclientes";
                    using (SqlCommand comando = new SqlCommand(sql, conexao ))
                    {
                        conexao.Open();
                        using (SqlDataReader tabela = comando.ExecuteReader())
                        {
                            while(tabela.Read())
                            {
                                lista.Add(new Clientes()
                                {
                                    cpf_cnpj = tabela["cpf_cnpj"].ToString(),
                                    nome = tabela["nome"].ToString(),
                                    endereco = tabela["endereco"].ToString(),
                                    rg_ie = tabela["rg_ie"].ToString(),
                                    tipo = tabela["tipo"].ToString(),
                                    valor = (float)Convert.ToDecimal(tabela["valor"]),
                                    valor_imposto =
                                    (float)Convert.ToDecimal(tabela["valor_imposto"]),
                                    total = (float)Convert.ToDecimal(tabela["total"])
                                });
                            }
                        }
                    }
                }
            }
        }
    }
}

```

```

        catch (Exception e)
        {
            Console.WriteLine(e.ToString());
        }
    }

    public String GetListaString()
    {
        string enviar= "<!DOCTYPE html>\n<html>\n<head>\n<meta charset='utf-8'
/>\n"+
            "<title>Cadastro de Clientes</title>\n</head>\n<body>";
        enviar = enviar + "<b>    CPF / CNPJ        -        Nome        -        Endereço        -
RG / IE        -        Tipo        -        Valor        - Valor Imposto -        Total  </b>";

        int i=0;
        string corfundo="",cortexto="";

        foreach (Clientes cli in GetLista())
        {
            if (i % 2 == 0)
            {
                corfundo = "#6f47ff"; cortexto="white";}
            else
            {
                corfundo = "ffffff";cortexto="#6f47ff";}
            i++;

            enviar = enviar +
                $"<br><div style='background-
color:{corfundo};color:{cortexto};'>" +
                cli.cpf_cnpj + " - " +
                cli.nome + " - " + cli.endereco + " - " + cli.rg_ie + " - " +
                cli.tipo + " - " + cli.valor.ToString("C") + " - " +
                cli.valor_imposto.ToString("C") + " - " +
                cli.total.ToString("C") + "<br>"+
                "</div>";
        }
        return enviar;
    }
}

```

```
public void imprimirListaConsole(){

    Console.WriteLine("    CPF / CNPJ    " + " - " + "    Nome    " +
        " - " + "    Endereço    " + " - " + "    RG / IE    " + " - " +
        "    Tipo    " + " - " + "    Valor    " + " - " + "Valor Imposto" +
        " - " + "    Total    ");

    foreach (Clientes cli in GetLista())
    {
        Console.WriteLine(cli.cpf_cnpj + " - " +
            cli.nome + " - " + cli.endereco + " - " + cli.rg_ie + " - " +
            cli.tipo + " - " + cli.valor.ToString("C") + " - " +
            cli.valor_imposto.ToString("C") + " - " +
            cli.total.ToString("C"));
    }

}
```

Clientes.cs

```
namespace Projeto_Web_Lh_Pets_versão_1
{
    class Clientes
    {
        public string? cpf_cnpj {get; set;}
        public string nome {get; set;}
        public string endereco {get; set;}
        public string rg_ie {get; set;}
        public string tipo {get; set;}
        public float valor {get; set;}
        public float valor_imposto {get; set;}
        public float total {get; set;}
    }
}
```

Program.cs

```
namespace Projeto_Web_Lh_Pets_versão_1;

public class Program
{
    public static void Main(string[] args)
    {
        var builder = WebApplication.CreateBuilder(args);
        var app = builder.Build();

        app.MapGet("/", () => "Projeto Web - LH Pets versão 1");

        app.UseStaticFiles();
        app.MapGet("/index", (HttpContext contexto)=> {
            contexto.Response.Redirect("index.html", false);

        });

        Banco dba=new Banco();
        app.MapGet("/listaClientes", (HttpContext contexto) =>{
            contexto.Response.WriteAsync( dba.GetListaString());
        });

        app.Run();
    }
}
```

Projeto Web-Lh Pets versao 1.csproj

```
<Project Sdk="Microsoft.NET.Sdk.Web">

  <PropertyGroup>
    <TargetFramework>net6.0</TargetFramework>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
    <RootNamespace>Projeto_Web_Lh_Pets_versão_1</RootNamespace>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="System.Data.SqlClient" Version="4.8.5" />
  </ItemGroup>

</Project>
```

vendas.sql

```
create database vendas;

GO

use vendas;
GO

create table tblclientes (
  cpf_cnpj varchar(20) primary key,
  nome varchar(30),
  endereco varchar(50),
  rg_ie varchar(15),
  tipo char,
  valor float,
  valor_imposto float,
  total float
);

GO
-- Pessoa Física
insert into tblclientes values ('123.456.789-00','Pedro da Silva','Rua Vergueiro
1234','4.567.890','f',2500.00,250.00,2750.00);
insert into tblclientes values ('234.567.890-11','Maria Pereira','Rua São Bento
345','5.6787.901','f',3000.00,300.00,3300.00);
insert into tblclientes values ('345.678.901-22','Leonora Alvez','Rua Parati
654','6.789.012','f',1500.00,150.00,1650.00);

GO
```

```
-- Pessoa Jurídica

insert into tblclientes values ('56.789.123/0001-00','Virtual Tecnologia S/A','Av.
Brasil 3456','567.890.123','j',35000.00,7000.00,42000.00);
insert into tblclientes values ('67.890.234/0001-11','J&M Associados Ltda.','Rua
Paraná 765','678.901.234','j',15000.00,3000.00,18000.00);
insert into tblclientes values ('89.012.345/0001-22','Alvorada Empreendimentos
S/A','Av. Constantino Alves 876','890.123.456','j',50000.00,10000.00,60000.00);

select * from tblclientes;
```