

Gemini said

Para aprender de verdade, o segredo é manter o código simples e usar as ferramentas que o ASP.NET Core oferece para facilitar sua vida.

Aqui está o projeto completo, refeito do zero, focado em ser **fácil de entender** e **impossível de errar**.

1. O Modelo (A Base)

O arquivo `Models/Contatos.cs`. Aqui definimos o que o banco de dados vai guardar. Usamos `?` para dizer que o e-mail e a categoria não são obrigatórios, evitando erros de SQL.

```
C#
using System.ComponentModel.DataAnnotations;

namespace MeuGerenciador.Models
{
    public class Contatos
    {
        public int Id { get; set; } // O banco cria sozinho

        [Required(ErrorMessage = "Digite o nome!")]
        public string Nome { get; set; } = string.Empty;

        [Required(ErrorMessage = "Digite o telefone!")]
        public string Telefone { get; set; } = string.Empty;

        public string? Email { get; set; } // O '?' permite ser vazio

        public string? Categoria { get; set; }
    }
}
```

2. O Controlador (O Cérebro)

O arquivo `Controllers/GerenciadorController.cs`. Ele tem apenas as funções essenciais. Note que para **Criar** e **Editar**, temos dois métodos: um para abrir a página e outro para salvar os dados.

```
C#
using MeuGerenciador.Data;
using MeuGerenciador.Models;
using Microsoft.AspNetCore.Mvc;

namespace MeuGerenciador.Controllers
{
    public class GerenciadorController : Controller
    {
        private readonly ContatoContext _db;
    }
}
```

```
public GerenciadorController(ContatoContext contexto) { _db =
contexto; }

// 1. LISTAR
public IActionResult Index()
{
    return View(_db.Contato.ToList());
}

// 2. CRIAR (Abrir tela)
[HttpGet]
public IActionResult Criar() => View();

// 2. CRIAR (Salvar no banco)
[HttpPost]
public IActionResult Criar(Contatos contato)
{
    if (ModelState.IsValid) {
        _db.Contato.Add(contato);
        _db.SaveChanges();
        return RedirectToAction("Index");
    }
    return View(contato);
}

// 3. EDITAR (Abrir tela com os dados)
[HttpGet]
public IActionResult Editar(int id)
{
    var contato = _db.Contato.Find(id);
    return View(contato);
}

// 3. EDITAR (Salvar alteração)
[HttpPost]
public IActionResult Editar(Contatos contato)
{
    if (ModelState.IsValid) {
        _db.Contato.Update(contato);
        _db.SaveChanges();
        return RedirectToAction("Index");
    }
    return View(contato);
}

// 4. EXCLUIR (Direto)
[HttpPost]
public IActionResult Excluir(int id)
{
    var contato = _db.Contato.Find(id);
    if (contato != null) {
        _db.Contato.Remove(contato);
        _db.SaveChanges();
    }
    return RedirectToAction("Index");
}
}
```

3. A Tela Principal (Index.cshtml)

Aqui listamos os contatos. Usei uma tabela simples para você não se perder no design agora.

```
HTML
@model IEnumerable<Contatos>

<h2>Meus Contatos</h2>
<a asp-action="Criar" class="btn btn-success"> + Novo Contato </a>

<table class="table mt-3">
  <thead>
    <tr>
      <th>Nome</th>
      <th>Telefone</th>
      <th>Ações</th>
    </tr>
  </thead>
  <tbody>
    @foreach (var c in Model) {
      <tr>
        <td>@c.Nome</td>
        <td>@c.Telefone</td>
        <td>
          <a asp-action="Editar" asp-route-id="@c.Id"
class="btn btn-primary">Editar</a>

          <form asp-action="Excluir" asp-route-id="@c.Id"
method="post" style="display:inline;">
            <button type="submit" class="btn btn-danger"
onclick="return confirm('Excluir?')">Apagar</button>
          </form>
        </td>
      </tr>
    }
  </tbody>
</table>
```

4. A Tela de Formulário (Criar.cshtml e Editar.cshtml)

O código abaixo serve para as duas! O segredo de iniciante é: **Sempre use asp-for**.

```
HTML
@model Contatos

<h3>Formulário de Contato</h3>

<form asp-action="@ViewContext.RouteData.Values["action"]"
method="post">
  @* ESSENCIAL: O ID oculto para o banco saber quem editar *@
  <input type="hidden" asp-for="Id" />

  <div class="mb-3">
    <label>Nome:</label>
```

```
<input asp-for="Nome" class="form-control" />
<span asp-validation-for="Nome" class="text-danger"></span>
</div>

<div class="mb-3">
  <label>Telefone:</label>
  <input asp-for="Telefone" class="form-control" />
  <span asp-validation-for="Telefone" class="text-
danger"></span>
</div>

<button type="submit" class="btn btn-success">Salvar</button>
<a asp-action="Index" class="btn btn-secondary">Voltar</a>
</form>
```