Olá, Estudante!

Nesta atividade, vamos dar continuidade à criação do site para a agência de viagens. Nesta entrega, contemple uma estrutura de layout com um menu horizontal na parte superior de todas as páginas: HOME, DESTINOS, PROMOCOES, CONTATO. Não esqueça que está atividade deve ser realizada individualmente.

Atente-se às seguintes orientações:

Modelagem de dados:

Criar o modelo conceitual, lógico usando uma ferramenta de modelagem de dados;

Criar as relações e cardinalidades entre as tabelas do modelo;

Criar o modelo físico (Tabelas e relações) usando SQL.

Back-end

Usando os conceitos de projeto MVC e C# .Net, criar um projeto aplicando o CRUD (Create, Read, Update, Delete) para todas as tabelas de seu modelo.

Front-end

1ª Passo:

Criar as páginas: Home, Destino, Promoções, Contato e as demais de acordo com seu modelo de dados usando HTML e CSS raiz, faça as formatações de acordo com a paleta de cores escolhida;

2º Passo:

Criar versão do frontend anterior usando o framework Bootstrap, com o conceito de responsividade. (Esta versão deve ser o frontend que vai fazer a conexão com banco de dados e o padrão MVC)

Passo a passo para apresentação da entrega:

O seu projeto deve ser entregue pelo GitHub. Você pode colar o link do seu projeto em um bloco de notas ou outro editor de texto, e fazer o upload desse arquivo.

Ou colar o link no campo "comentário", disponível na área de entrega.

Agora é só aguardar o feedback do seu monitor!

Bom estudo!

PASSO A PASSO PROJETO MVC

01 – dotnet new mvc

02 - dotnet add package Microsoft.AspNetCore.Mvc.Razor.RuntimeCompilation --version 5.0.8

03 - dotnet add package Microsoft.EntityFrameWorkCore.SqlServer --version 5.0.8

04 - dotnet add package Microsoft.EntityFrameWorkCore.Tools --version 5.0.8

05 - dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design --version 5.0.2

06 – Criar as classes.cs

07 – using System.ComponentModel.DataAnnotations

08 – Criar chaves estrangeiras

09 – Criar pasta data

10 – Na pasta data criar a classe

10.1 – nomeDoSeuContext.cs

11 – Herança

11.1 – public class nome\_classe\_Context : DbContext

12 – Criar construtor

12.1 – public NomeDoSeuContext(DbContextOptions<ClienteContext> opt) : base(opt){}

13 – CRIAR REFERENCIA PARA CADA CLASSE

13 – public DbSet<nome\_do\_modelo\_de\_classe> nome\_variável { get; set; }

14 – CONFIGURAÇÃO DO APPSETTING.JSON

14.1 – "ConnectionStrings": {

"ConnDB": "Server=NOME-DESKTOP-NO-SQLSERVER; Database=NomeDoBD; User ID=sa; Password=SenhaDoSeuSqlServer; Trusted\_Connection=false; MultipleActiveResultSets=True"

}

15 – CONFIGURAÇÃO DO SERVIÇO DE CONEXÃO DO STURTUP

15.1 – services.AddDbContext<NomeDoSeuContext>(opt => opt.UseSqlServer(Configuration.GetConnectionString("ConnDB")));

16 – GERAR MIGRATIONS:

16.1 – dotnet ef migrations add nomeDoProjeto

17 – SUBIR INFORMAÇÕES PARA O DB

17.1 – dotnet ef database update

18 – CRIAR PASTAS CONTROLERS E VIEWS

18.1 – dotnet aspnet-codegenerator --project "C:\Caminho\Do\Projeto\ " controller --force --controllerName NomeDoControlerIgualAModeloDeClasseController --model NomeDoProjeto.Models.NomeDaClasse --dataContext NomeDoProjeto.Data.NomeDoContext --relativeFolderPath Controllers --controllerNamespace NomeDoProjeto.Controllers

19 – INTEGRAR OS CSHTML COM O LAYOUT PRINCIPAL

19.1 – Cada cshtml deve ter seu proprio View

19.2 – ViewData["Title"] = "Comando Nome\_do\_modelo"

20 – Apagar HomeController e suas Views

21 – Mudar o nome do asp-controler no layout para ser o controle principal Cliente no \_layout.cshtml

22 – Criar os outros controllers no layout para as próximas paginas

23 – Mudar o controle principal “pattern” na Startup

23.1 – "{controller=NomeDoControlePrincipal}/{action=Index}/{id?}");