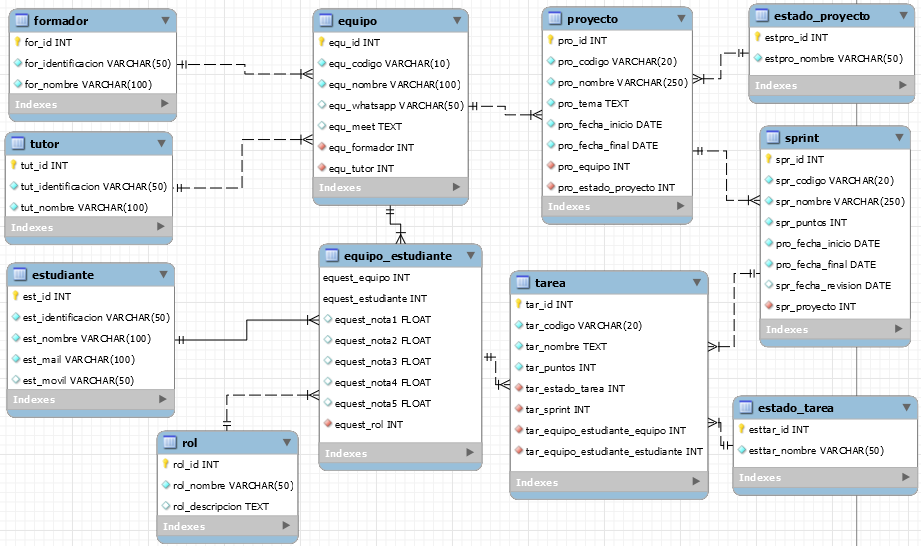
El Proyecto estará basado en un nivel general para el proyecto propuesto en clase llamado

Directorios color **Rojo**

Proyecto y sus componentes a crear color **azul**

Comando a ejecutar por consola color **verde**

****

|  |  |
| --- | --- |
|  |  |
|  | Verificar que contenga todas las instalaciones y referencias, en cada una de las capas, sea la versión 5.0.9Si ya tiene las instalaciones de paquetes con la versión 5.0.0, cambie en los archivos respectivos (ver gráfica) yComando en la capa de presentación dotnet restore |
|  |  |
|  | **COMPILACIÓN (dotnet build)**: Verificación de errores de sintaxis y referencias y paquetes incluidos, sobre las capas trabajadas:  Como la capa Presentación contiene la referencia a Dominio y Persistencia, al compilarla se compilan las tres.  **Verificar que se encuentre en la capa de Proyecto.App.Presentacion** |
|  |  |
|  | **VERIFICAR QUE FRONT-END CONSTRUIDO, CORRA EL PROYECTO WEB EN EL NAVEGADOR – RUN – EJECUTAR (dotnet watch run)**: Se obtienen los resultados de la primera página para adecuarla a un proyecto en particular, aplicando el CRUD completo sobre cada una de las entidades creadas. |
|  |  |
|  |  |
|  | **COMUNICACIÓN DEL FRONT-END CON EL BACK-END:** Se crean los Modelos de las clases en la capa de Dominio:   * Una sola carpeta **Entidades** * Una clase por cada entidad, nombre inicial mayúsculas, ejm **Formador.cs** * Se escribe el código de la clase con sus atributos y tipos de datos respectivos * El **namespace**, hasta el Entidades * Adicionar las directivas USING para validaciones DataAnnotatines. |
|  |  |
| 4.1. | **Código para la clase Formador.cs** |
|  | using System;  using System.ComponentModel.DataAnnotations;  namespace Proyectos.App.Dominio.Entidades  {      public class Formador      {          //atributos de la clase formador, obligado para cada entidad un id como PK          [Key]          [Display(Name = "ID")]          public int id { get; set; }          [Required]          [Display(Name = "Nro. Identificación")]          public string identificacion{ get; set; }          [Required]          [Display(Name = "Nombre del Formador")]          public string nombre{ get; set; }          [Required]          [Display(Name = "Correo Electrónico")]          public string mail{ get; set; }          [Required]          [Display(Name = "Celular")]          public string movil{ get; set; }          public bool vigente{ get; set; }      }    } |
| 4.2. | **Compilar y verificar que NO se tengan errores en la capa de dominio** |
|  | **CAPA DE PERSISTENCIA – INTERFACES – REPOSITORIOS - APPCONTEXT:** Se crean los archivos de las clases e interfaces en la capa de Persistencia:   * Una sola carpeta **AppRepositorios** * El alcance de los namespaces hasta AppRepositorios * Una clase para el **AppContext** * Una Interface **IRepositorios** para las firmas o contratos de los métodos CRUD * Una Clase **Repositorios** para la implementación de los métodos del CRUD |
|  |  |
| 6.1. | Código para la Clase **AppContext.cs**  La cadena de conexión varía de acuerdo al servidor, usuario, clave y base de datos, para este tutorial la Base de Datos a crear el **catalog BDProyectos84** |
|  | using System;  using Microsoft.EntityFrameworkCore;  using Proyectos.App.Dominio.Entidades;  namespace Proyectos.App.Persistencia  {      public class AppContext : DbContext      {          //poner aqui los modelos un DbSet por cada clase o tabla          public DbSet<Formador> formador{get; set;}          //crear el deContext para la creación y conexión con la Base de Datos en SQLServer          protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)          {              if (!optionsBuilder.IsConfigured)              {              optionsBuilder             .UseSqlServer("Server=localhost; user id=sa; password=12345; Initial Catalog=BDProyectos84;");              }          }      }  } |
| 6.2. | Código para la Interface IRepositorios, la misma van las firmas de los métodos del CRUD, sin su implementación, la cuál se implementa en la clase Repositorios |
|  | //Directivas  using System;  using System.Collections;  using System.Collections.Generic;  using System.Linq;  using Microsoft.EntityFrameworkCore;  using Proyectos.App.Dominio.Entidades;  using Proyectos.App.Persistencia.AppRepositorios;  namespace Proyectos.App.Persistencia.AppRepositorios  {      public interface IRepositorios      {          //contratos o firmas para los metodos Formador          Formador AddFormador(Formador formador);          IEnumerable<Formador> GetAllFormadores();          Formador GetFormador(int? idFormador);          Formador UpdateFormador(Formador formador);          void DeleteFormador(int idFormador);      }  } |
| 6.3. | Código para la clase Repositorios, en la misma hereda de la interface creada con las firmas y se implementa cada uno de los métodos del CRUD. |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using Microsoft.EntityFrameworkCore;  using Proyectos.App.Dominio.Entidades;  namespace Proyectos.App.Persistencia.AppRepositorios  {      public class Repositorios : IRepositorios      {         private readonly AppContext \_appContext;         public Repositorios(AppContext appContext)          {              \_appContext = appContext;          }          //AQUÍ CADA UNO DE LOS MÉTODOS DEL CRUD, REFERENCIADOS EN LA INTERFACE           //SIGUIENTE DIAPOSITIVA          Formador IRepositorios.AddFormador(Formador formador)          {          try           {              var FormadorAdicionado = \_appContext.formador.Add( formador );  //INSERT en la BD              \_appContext.SaveChanges();              return FormadorAdicionado.Entity;            }catch              {                  throw;              }          }          IEnumerable<Formador> IRepositorios.GetAllFormadores()          {              return \_appContext.formador;          }         Formador IRepositorios.GetFormador(int? idFormador)         {              return \_appContext.formador.FirstOrDefault(p => p.id == idFormador);         }         Formador IRepositorios.UpdateFormador(Formador formador)          {              var FormadorEncontrado = \_appContext.formador.FirstOrDefault(p => p.id == formador.id);              if (FormadorEncontrado != null)              {                  FormadorEncontrado.identificacion  = formador.identificacion;                  FormadorEncontrado.nombre          = formador.nombre;                  FormadorEncontrado.mail            = formador.mail;                  FormadorEncontrado.movil           = formador.movil;                  FormadorEncontrado.vigente         = formador.vigente;                  \_appContext.SaveChanges();              }              return FormadorEncontrado;          }          void IRepositorios.DeleteFormador(int idFormador)          {              var FormadorEncontrado = \_appContext.formador.FirstOrDefault(p => p.id == idFormador);              if (FormadorEncontrado == null)                  return;              \_appContext.formador.Remove(FormadorEncontrado);              \_appContext.SaveChanges();          }      }  } |
|  | **Compilar y verificar que NO se tengan errores en la capa de Persistencia** |
|  | **CONEXIÓN Y CREACIÓN DE LA BASE DE DATOS (MIGRACIÓN DE LOS MODELOS):** Se crean los archivos de migración para la base de datos con los comandos respectivos:   * Verificar en la capa de **consola**, las librerías y referencias requeridas |
|  |  |
|  | Para la creación y migración de la Base de datos, tener en cuenta:   * **Servidor de Bases de Datos** * **Usuario** * **Clave** * **Nombre de la Base de Datos** |
| 8.1. |  |
| 8.2. | **Comando para la Migración de los Modelos a la carpeta Migrations, todavía NO se crea la Base de Datos** |
|  | **dotnet ef migrations add Inicial --startup-project ..\Proyectos.App.Consola\** |
|  |  |
|  | **dotnet ef migrations add Inicial --startup-project ..\Proyectos.App.Consola\** |
|  |  |
| 8.3. | **Compilar y verificar que NO se tengan errores en la capa de Proyecto.App.Persistencia.** |
|  | **ENTITY-CORE CREARA LA BASE DE DATOS, A PARTIR DE LOS MODELOS CREADOS (CODE-FIRST) Comando para la Migración de los Modelos a la carpeta Migrations, todavía NO se crea la Base de Datos**  **En la misma capa de persistencia, aplicar el comando respectivo y verificar en SQLServer** |
|  | **dotnet ef database update --startup-project ..\Proyectos.App.Consola\** |
|  |  |
|  | En resumen para la migración, en **migrations** se crean los archivos de migración y en **update**, se crea la base de datos en el motor SQLServer.  Compilar y verificar que NO se tengan errores, posiblemente de compatibilidad de versiones de los paquetes instalados |
|  |  |
| 9.2. | Verificar la creación de la Base de Datos “BD**Proyectos84**” en el servidor de bases de datos SQLServer o Azure, se crean la tabla formadores y una adicional de migración controlada por el EntityFrameworkcore. Cuando se trabaja con Code First NO se debe alterar la estructura de la Base de Datos por el Servidor, sino recurrir a los scripts SQL que genera el EF, al correr una actualización. |
|  |  |
|  | Probar en la capa de presentación nuevamente la página web, inicialmente la lista se encuentra vacia, para empezar a insertar registros a la tabla respectiva |
|  |  |
|  | Complementar el código de la capa de presentación para las operaciones del CRUD, asociados a las Clases e Intefaces creadas en la capa de persistencia, tenga muy presente los alcances de los **namespace** |
| 13.1. | **List.cshtml.cs** |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Threading.Tasks;  using Microsoft.AspNetCore.Mvc;  using Microsoft.AspNetCore.Mvc.RazorPages;  using Microsoft.AspNetCore.Authorization;  using Microsoft.EntityFrameworkCore;  using Proyectos.App.Dominio.Entidades;  using Proyectos.App.Persistencia.AppRepositorios;  using Proyectos.App.Persistencia;  namespace Proyectos.App.Presentacion.Pages  {      //[Authorize]      public class ListModel : PageModel      {          private readonly IRepositorios \_appContext;          public IEnumerable<Formador> formadores {get; set;}          public ListModel()          {              this.\_appContext = new Repositorios(new Proyectos.App.Persistencia.AppContext());          }            public void OnGet()          {              formadores = \_appContext.GetAllFormadores();          }        }  } |
|  | **List.cshtml** |
|  | @page  @model Proyectos.App.Presentacion.Pages.ListModel  @{     //var searchString = Model.searchString;     ViewData["Title"] = "Lista de Formadores";  }  <div class="card">      <div class="card-header">          <div class="row">              <div class="col-sm-6">                  <h2><div class="text-info">Lista de Formadores</h2>              </div>              <div class="col-sm-6">                  <a class="btn btn-primary btn-sm" asp-page="Crear">Crear Nuevo Formador</a>              </div>          </div>      </div>      <div class="card-body">          @if (Model.formadores.Count() > 0) {              <br /><br />              <table class="table table-striped border">                  <tr class="table-secondary">                      <th>@Html.DisplayNameFor(m => m.formadores.First().id)</th>                      <th>@Html.DisplayNameFor(m => m.formadores.First().identificacion)</th>                      <th>@Html.DisplayNameFor(m => m.formadores.First().nombre)</th>                      <th>@Html.DisplayNameFor(m => m.formadores.First().mail)</th>                      <th>@Html.DisplayNameFor(m => m.formadores.First().movil)</th>                      <th>@Html.DisplayNameFor(m => m.formadores.First().vigente)</th>                      <th>ACCION1</th>                      <th>ACCION2</th>                      <th>ACCION3</th>                  </tr>                  @foreach (var item in Model.formadores)  {                  <tr>                      <td>@Html.DisplayFor(m => item.id)</td>                      <td>@Html.DisplayFor(m => item.identificacion)</td>                      <td>@Html.DisplayFor(m => item.nombre)</td>                      <td>@Html.DisplayFor(m => item.mail)</td>                      <td>@Html.DisplayFor(m => item.movil)</td>                      <td>@Html.DisplayFor(m => item.vigente)</td>                      <td>                          <a class="btn btn-primary btn-sm" asp-page="./Details" asp-route-formadorId="@item.id">                              Detalle</a>                      </td>                      <td>                          <a class="btn btn-success btn-sm" asp-page="./Edit" asp-route-formadorId="@item.id">                              Editar                          </a>                      </td>                      <td>                          <a class="btn btn-danger btn-sm" asp-page="./Delete" asp-route-formadorId="@item.id">                              Eliminar                          </a>                      </td>                  </tr>                  }              </table>          }          else{              <p>No hay Formadores para Listar</p>          }      </div>  </div> |
|  | **CRUD INSERT - CREAR :** Crear los archivos para **Crear**, que se encargará del INSERT, el formulario se muestra en blanco para ser completado por el usuario y al Guardar se almacenará el registro en la Base de Datos en la tabla respectiva. |
|  | **Código Crear.cshtml.cs** |
|  | using Microsoft.AspNetCore.Mvc;  using Microsoft.AspNetCore.Mvc.RazorPages;  using Proyectos.App.Dominio.Entidades;  using Proyectos.App.Persistencia.AppRepositorios;  using Proyectos.App.Persistencia;  namespace Proyectos.App.Presentacion.Formadores  {      public class CrearModel : PageModel      {          private readonly IRepositorios \_appContext;          [BindProperty]          public Formador formador { get; set; }          public CrearModel(){              //cargar desde la base de datos tabla formador              this.\_appContext = new Repositorios(new Proyectos.App.Persistencia.AppContext());              //cargarTemporales();          }            public IActionResult OnGet(int? formadorId)          {              if (formadorId.HasValue)              {                  formador = \_appContext.GetFormador(formadorId.Value);              }              else              {                  formador = new Formador();              }              if (formador == null)              {                  return RedirectToPage("./NotFound");              }              else                  return Page();          }          public IActionResult OnPost()          {              if(!ModelState.IsValid)              {                  return Page();              }              if(formador.id > 0)              {                 formador = \_appContext.UpdateFormador(formador);              }              else              {                 \_appContext.AddFormador(formador);              }              return Page();          }      }  } |
|  | **Código Crear.cshtml** |
|  | @page  @model Proyectos.App.Presentacion.Formadores.CrearModel  @{      ViewData["Title"] = "Crear Formador";  }  <div class="card">      <div class="card-header">          <h2 class="text-info">Crear un nuevo formador</h2>      </div>      <div class="card-body">          <form method="post">              <div asp-validation-sumary="ModelOnly" class="text-danger">                  <div class="container" style="padding: 50px">                      <div class="form-group">                          <div class="row">                              <div class="col-sm-2">                                  <label asp-for="formador.identificacion"></label>                              </div>                              <div class="col-sm-3">                                  <input asp-for="formador.identificacion" class="form-control" />                              </div>                              <span asp-validation-for="formador.identificacion" class="text-danger"></span>                          </div>                      </div>                      <div class="form-group">                          <div class="row">                              <div class="col-sm-2">                                  <label asp-for="formador.nombre"></label>                              </div>                              <div class="col-sm-6">                                  <input asp-for="formador.nombre" class="form-control" />                              </div>                              <span asp-validation-for="formador.nombre" class="text-danger"></span>                          </div>                      </div>                      <div class="form-group">                          <div class="row">                              <div class="col-sm-2">                                  <label asp-for="formador.mail"></label>                              </div>                              <div class="col-sm-6">                                  <input type="email" asp-for="formador.mail" class="form-control" />                              </div>                              <span asp-validation-for="formador.mail" class="text-danger"></span>                          </div>                      </div>                      <div class="form-group">                          <div class="row">                              <div class="col-sm-2">                                  <label asp-for="formador.movil"></label>                              </div>                              <div class="col-sm-3">                                  <input asp-for="formador.movil" class="form-control" />                              </div>                              <span asp-validation-for="formador.movil" class="text-danger"></span>                          </div>                      </div>                      <div class="form-group">                          <div class="row">                              <div class="col-sm-2">                                  <label asp-for="formador.vigente" class="text-primary"></label>                              </div>                              <div class="col-sm-6">                                  <input type="checkbox" asp-for="formador.vigente" class="form-check-input" value=@Model.formador.vigente />                              </div>                          </div>                      </div>                      <div class="form-group">                          <div class="row">                              <div class="col-sm-2 offset-sm-2">                                  <input type="submit" value="Crear" class="btn btn-success form-control" />                              </div>                              <div class="col-sm-4">                                  <a asp-page="List" class="btn btn-primary form-control">Volver a la Lista</a>                              </div>                          </div>                      </div>                  </div>              </div>          </form>      </div>  </div> |
|  | **Verificar Insertando varios registros por la Interfaz** |
|  |  |
|  |  |
|  | **Al Editar el registro, puede actualizar los campos del formulario y al guardar se actualiza el registro en la Base de Datos.** |
|  |  |
|  |  |
|  |  |
|  |  |
|  | **Al Detalle del registro, puede Visualizar los campos del formulario cargados desde la base de datos, es la oportunidad de utilizar un CARD para su presentación.** |
|  |  |
|  | **CRUD DELETE:** Es similar al Editar, crear los archivos para **Delete**, que se encargará del Eliminar dependiendo del ID de la lista, abriendo un formulario con los datos respectivos al ID. Y desactivándolos para que el usuario NO digite sobre los mismos. |
|  | **En List.cshtml**  Crear al frente da cada fila otra celda para el botón **Eliminar** respectivo |
|  | <td>                  <a class="btn btn-danger table-btn" asp-page="./Delete" asp-route-formadorId="@formador.id">                      Eliminar                  </a>              </td> |
|  |  |
|  | Crear los archivos html y cs respectivamente |
|  | **dotnet new page -n Delete -na Proyectos.App.Presentacion -o .\Pages\Formadores\** |
|  | **Código Delete.cshtml.cs** |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Threading.Tasks;  using Microsoft.AspNetCore.Mvc;  using Microsoft.AspNetCore.Mvc.RazorPages;  using Proyectos.App.Dominio;  using Proyectos.App.Persistencia.AppRepositorios;  namespace Proyectos.App.Presentacion.Pages  {      public class DeleteModel : PageModel      {         private readonly IRepositorios \_appContext;          [BindProperty]          public Formador formador  { get; set; }          public DeleteModel()          {              this.\_appContext  =new Repositorios(new Proyectos.App.Persistencia.AppRepositorios.AppContext());          }    //se ejecuta al presionar Eliminar en la lista          public IActionResult OnGet(int formadorId)          {              formador = \_appContext.GetFormador(formadorId);              if(formador == null)              {                  return RedirectToPage("./NotFound");              }              else              return Page();          }    //se ejecuta al presionar Eliminar en el formulario          public IActionResult OnPost()          {              if(formador.id > 0)              {                 \_appContext.DeleteFormador(formador.id);              }              return Page();          }      }  } |
|  | **Código Delete.cshtml** |
|  | @page  @model Proyectos.App.Presentacion.Pages.DeleteModel  @{      int formadorId = Model.formador.id;  }  <h2>Edición de los datos del formador @Model.formador.nombre @Model.formador.identificacion</h2>  <form method="post">      <input type="hidden" asp-for="formador.id" />      <div class="form-group">          <label asp-for="formador.identificacion"></label>          <input asp-for="formador.identificacion" class="form-control" disabled>          <span class="text-danger" asp-validation-for="formador.identificacion"></span>      </div>      <div class="form-group">          <label asp-for="formador.nombre"></label>          <input asp-for="formador.nombre" class="form-control" disabled>          <span class="text-danger" asp-validation-for="formador.nombre"></span>      </div>      <button type="submit" class="btn btn-danger">ELIMINAR</button>  </form>  <a asp-page="./List" class="btn-default">Regresar a la lista de formadores </a> |
|  | **Comprobar Eliminando un registro** |
|  |  |
|  |  |

HERRAMIENTA PARA LOS MANUALES: https://www.helpndoc.com/es/

