

2021

Universidad Tecnológica de Puebla

**Producto 3 (MVC).**

**Carrera: Tecnologías de la información (desarrollo de software multiplataforma).**

**Leonardo Saínos Pérez**

**Aplicaciones web para I.4.0**

Contenido

[INTRODUCCIÓN. 4](#_Toc69934188)

[CAPTURAS DE PANTALLA. 4](#_Toc69934189)

[ASISTE 4](#_Toc69934190)

[Indice 4](#_Toc69934191)

[Detalles 5](#_Toc69934192)

[Actualizar 6](#_Toc69934193)

[Eliminar 7](#_Toc69934194)

[EMPLEADO 8](#_Toc69934195)

[Indice 8](#_Toc69934196)

[Detalles 8](#_Toc69934197)

[Crear 9](#_Toc69934198)

[Actualizar 10](#_Toc69934199)

[Eliminar 11](#_Toc69934200)

[ROL. 11](#_Toc69934201)

[Indice 12](#_Toc69934202)

[Detalles 12](#_Toc69934203)

[Crear 12](#_Toc69934204)

[Actualizar 14](#_Toc69934205)

[Eliminar 14](#_Toc69934206)

[CÓDIGO DEL MODELO 15](#_Toc69934207)

[CÓDIGO DEL CONTROLADOR. 16](#_Toc69934208)

[CÓDIGO DE LAS VISTAS 32](#_Toc69934209)

[VISTAS DE EMPLEADO 32](#_Toc69934210)

[Empleado 32](#_Toc69934211)

[Actualizar empleado 33](#_Toc69934212)

[Crear Nuevo 35](#_Toc69934213)

[Detalles empleado 38](#_Toc69934214)

[Eliminar empleado 39](#_Toc69934215)

[VISTAS DE ASISTE 41](#_Toc69934216)

[Asiste 41](#_Toc69934217)

[Dar asistencia 42](#_Toc69934218)

[Detalles 43](#_Toc69934219)

[Actualizar\_asistencia 44](#_Toc69934220)

[Eliminar\_Asistencia 45](#_Toc69934221)

[VISTAS DE ROL 47](#_Toc69934222)

[Index 47](#_Toc69934223)

[Detalles 48](#_Toc69934224)

[Crear 49](#_Toc69934225)

[Actualizar 50](#_Toc69934226)

[Eliminar 51](#_Toc69934227)

[CONCLUSIÓN 52](#_Toc69934228)

# 

# INTRODUCCIÓN.

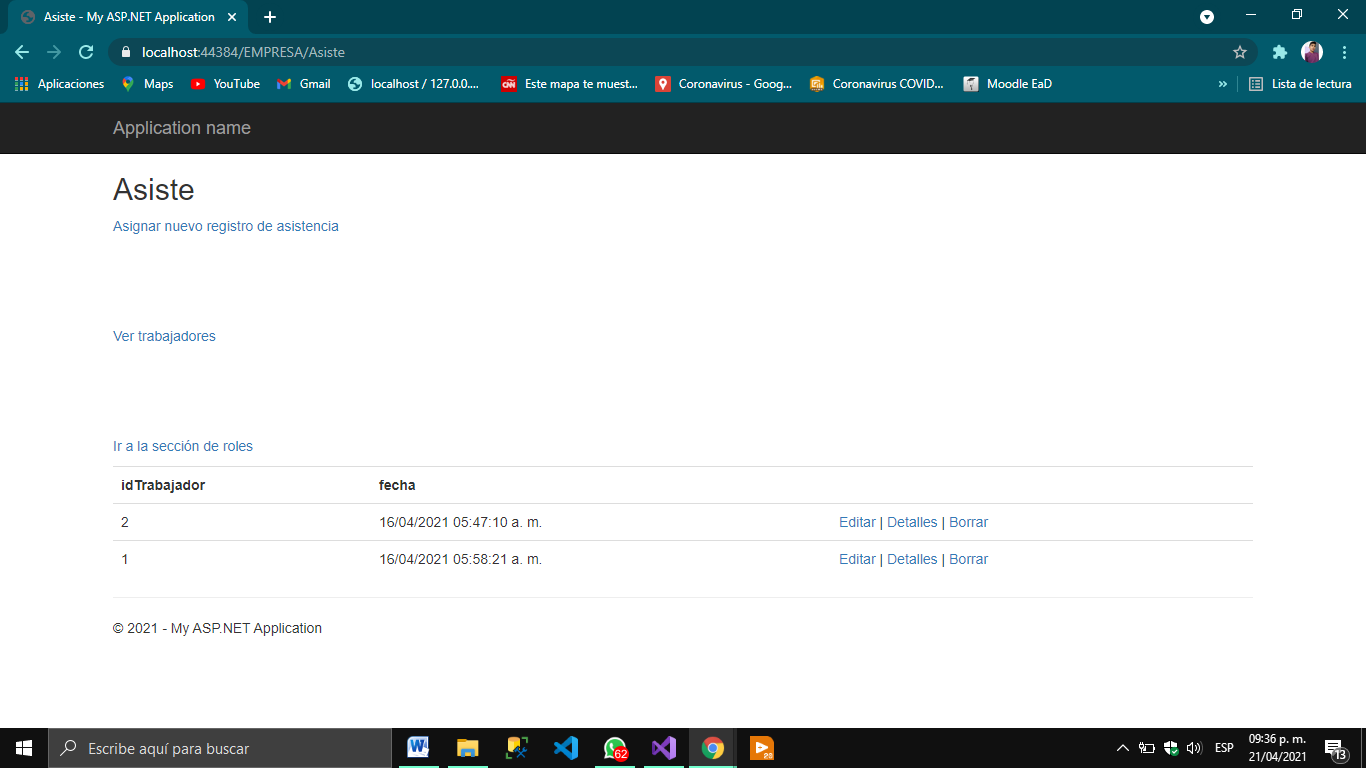
En esta aplicación MVC con c# crearemos una prueba con tablas relacionadas en entity framework la cual como conocemos, se encarga de apoyarnos con las vistas.La cual yo defino las vistas como el código html sólo que entity ya se hace cargo de crearlo por ti incluso de las funciones y los valores que captura.

En este programa haremos el crud de tres tablas relacionadas, las cuales son Trabajador, Empleado y Asiste de la base de datos llamada Fabrica, iremos creando poco a poco lo que es las acciones que son parecidas pero no iguales a los métodos que conocemos en POO y también se estarán usando expresiones Landa sin usar directamente consultas SQL ya que entity nos estará apoyando en esto.

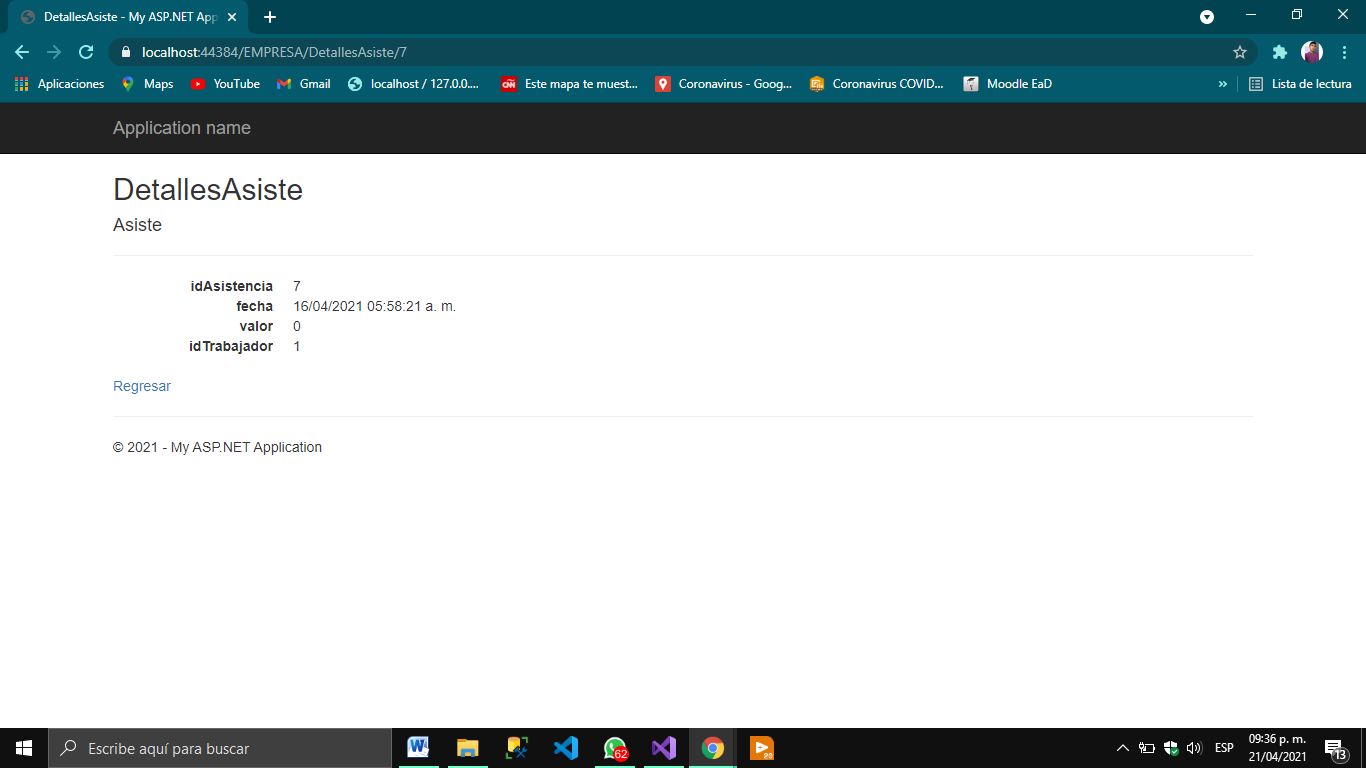
# CAPTURAS DE PANTALLA.

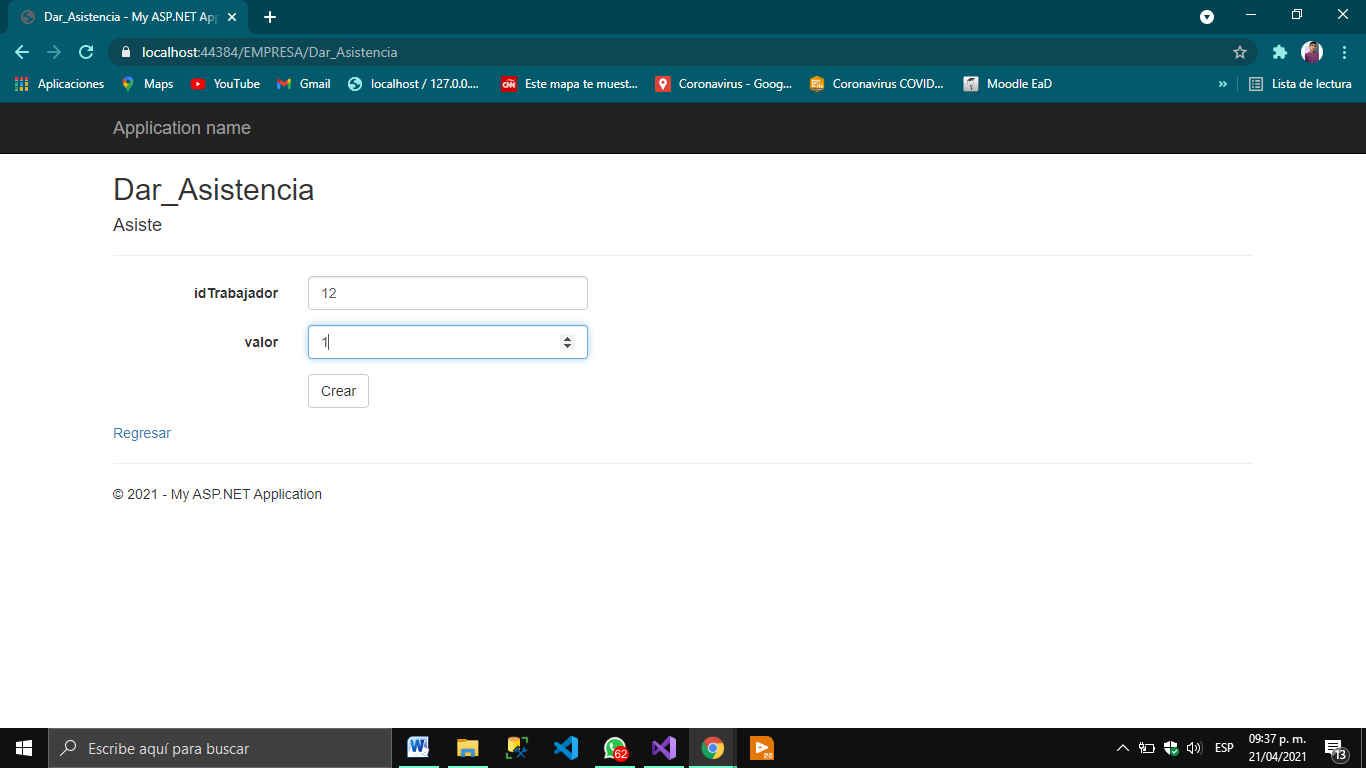
## ASISTE

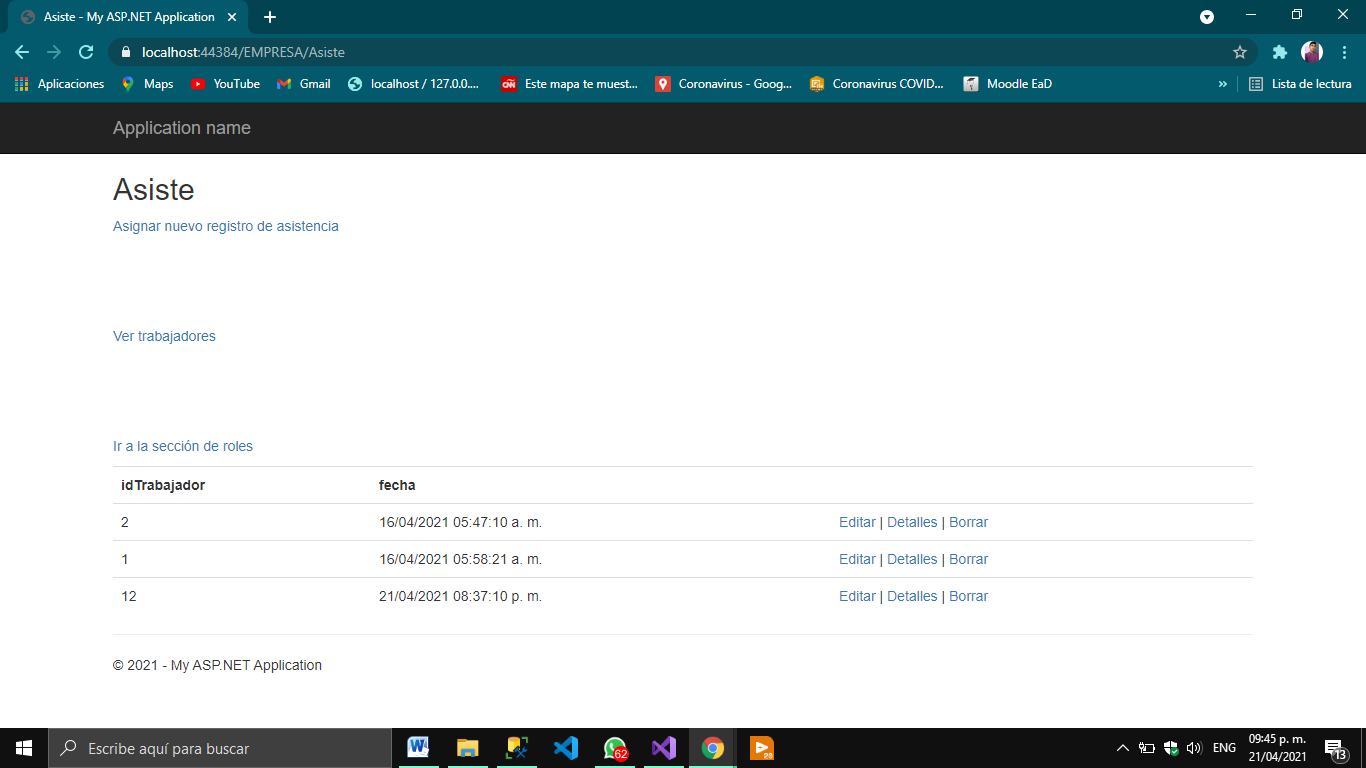
### Indice



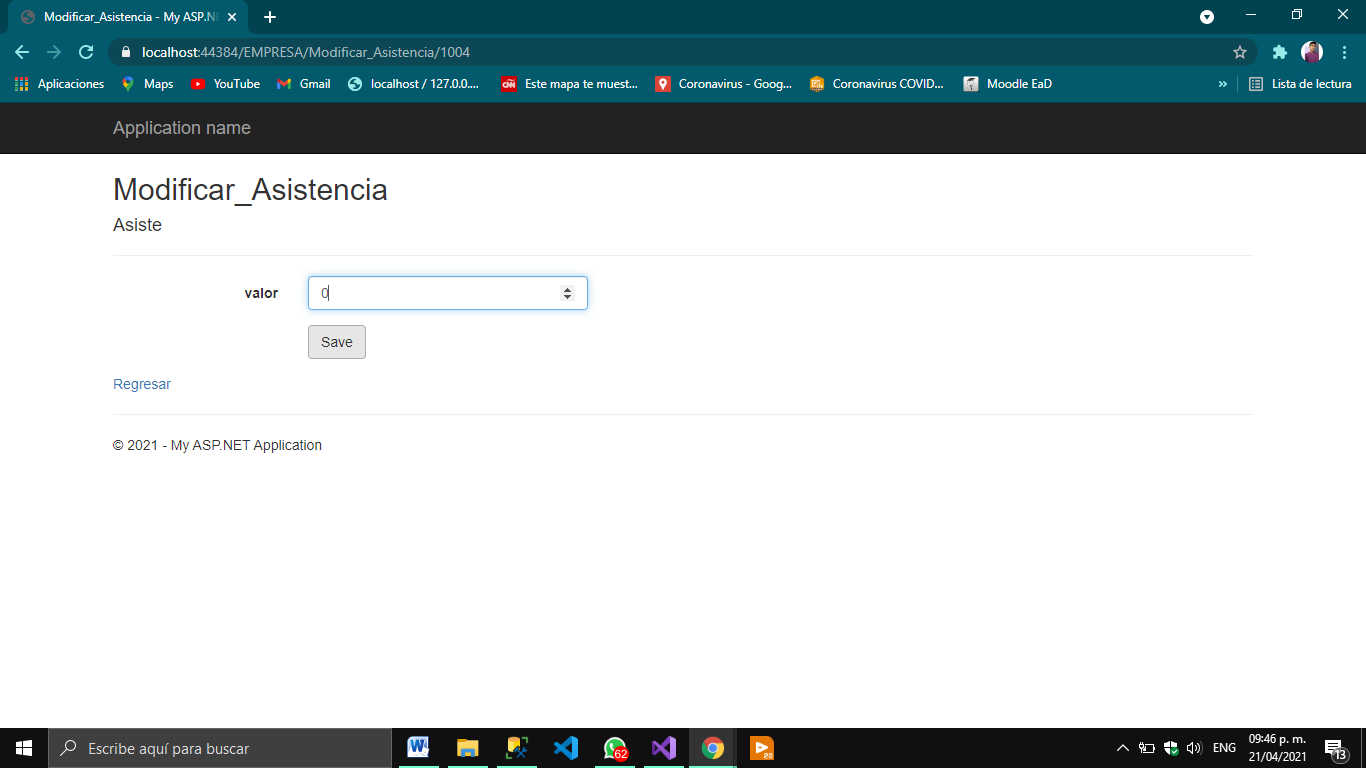
### Detalles

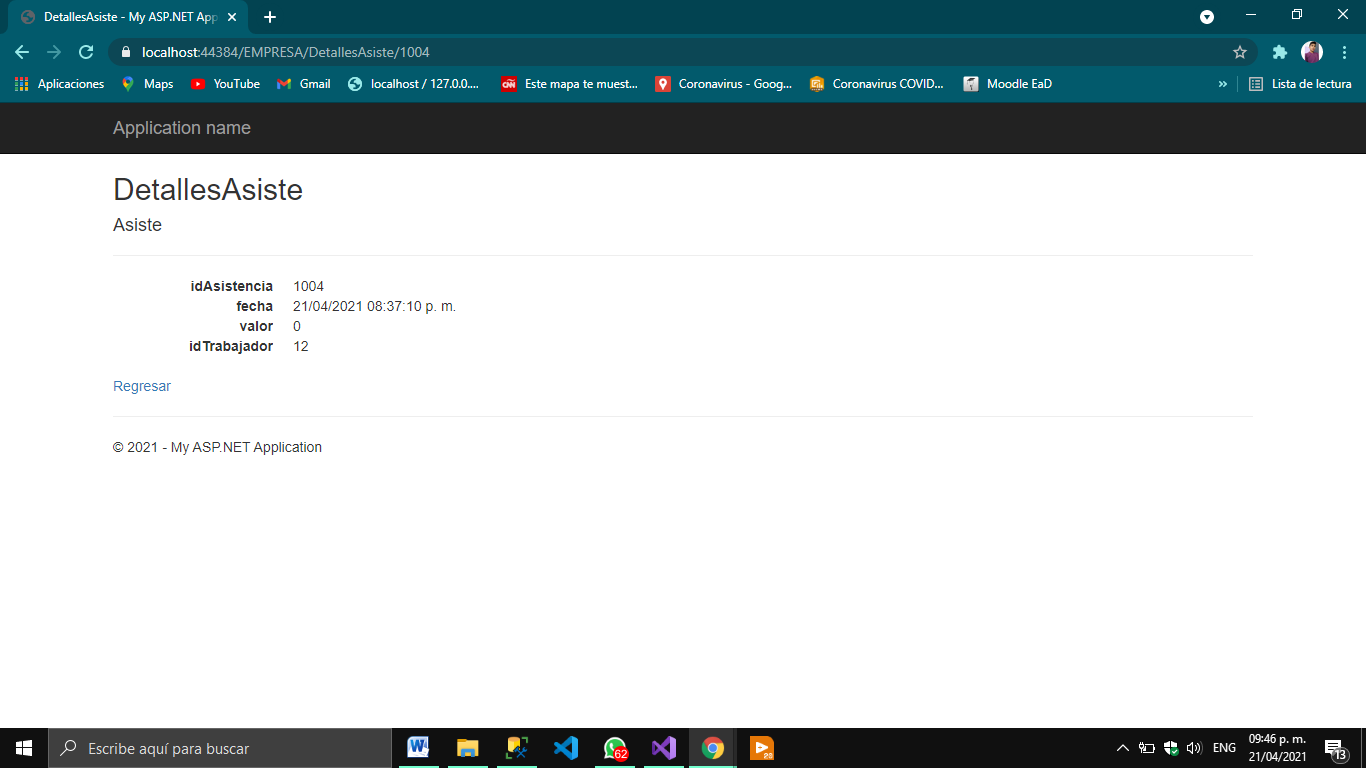
Crear



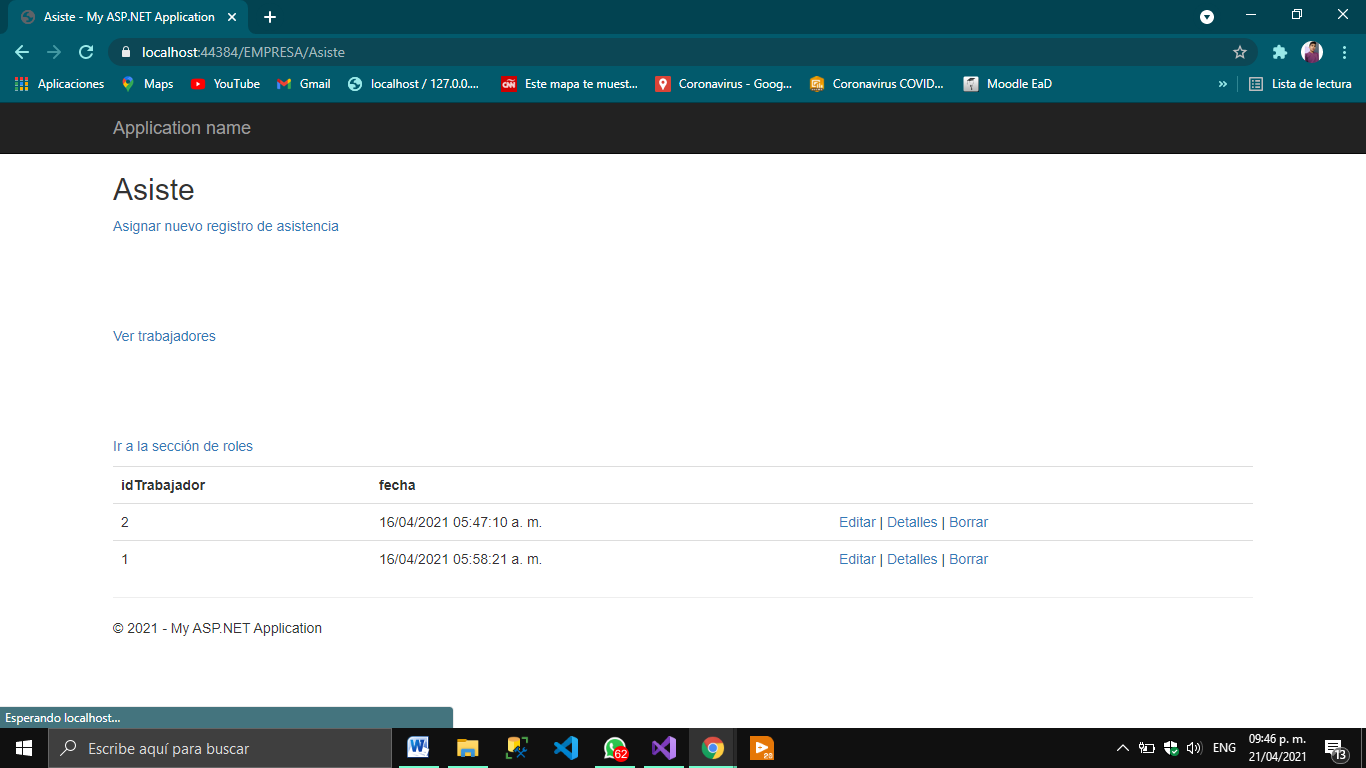


### Actualizar



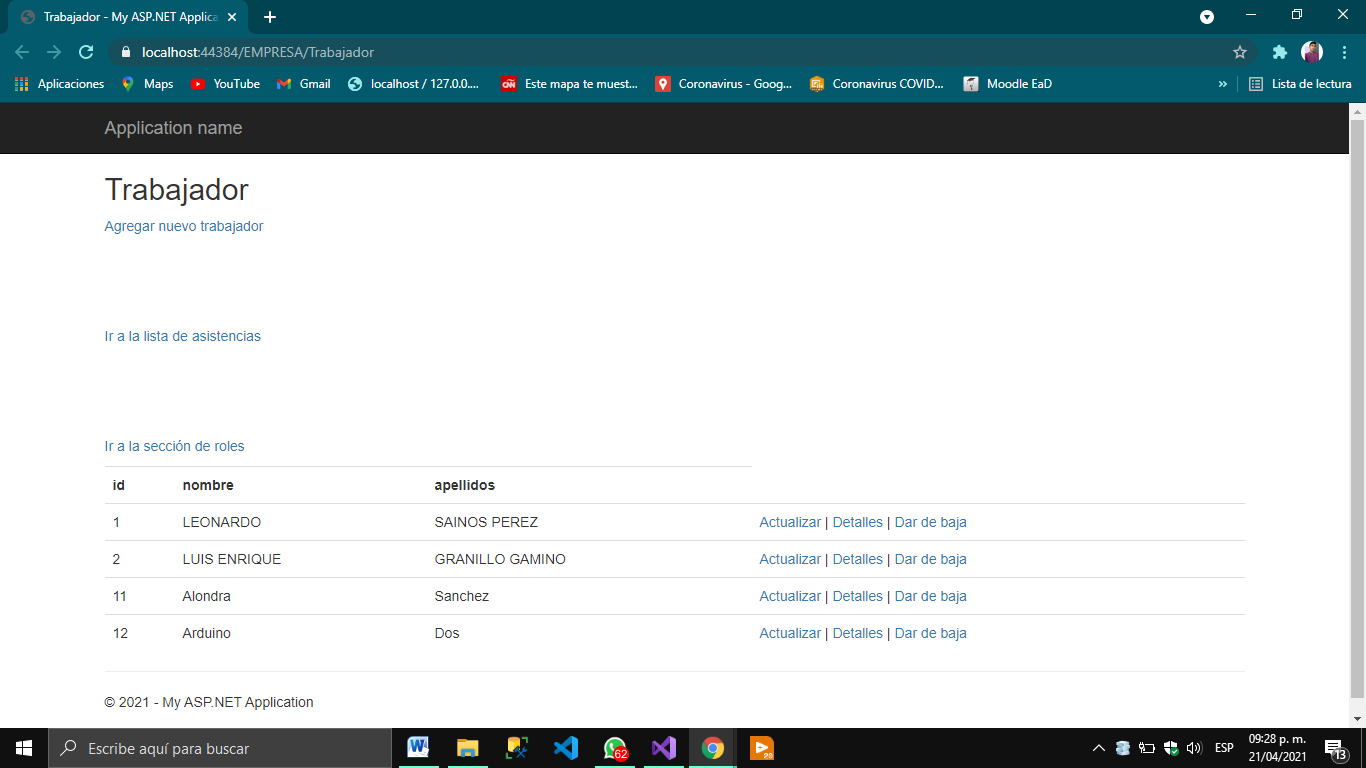


### Eliminar

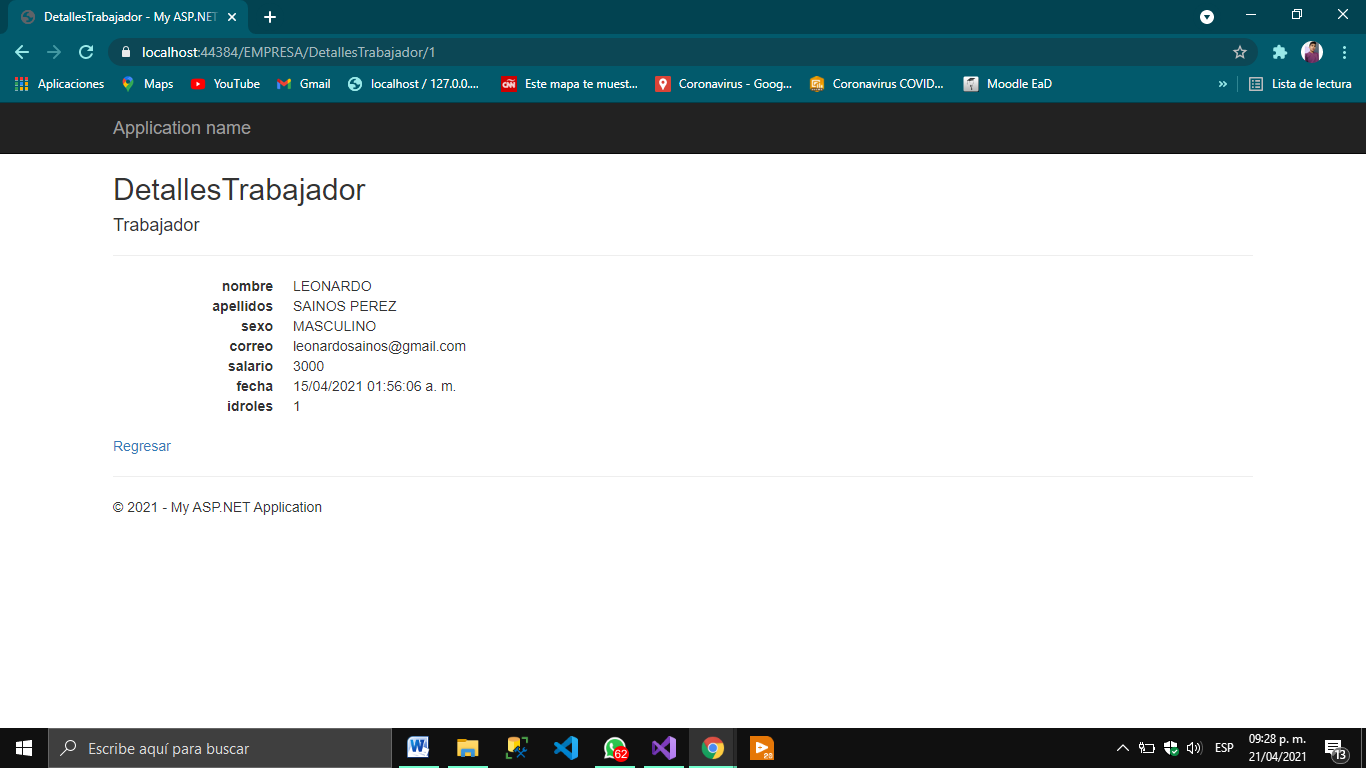


## EMPLEADO

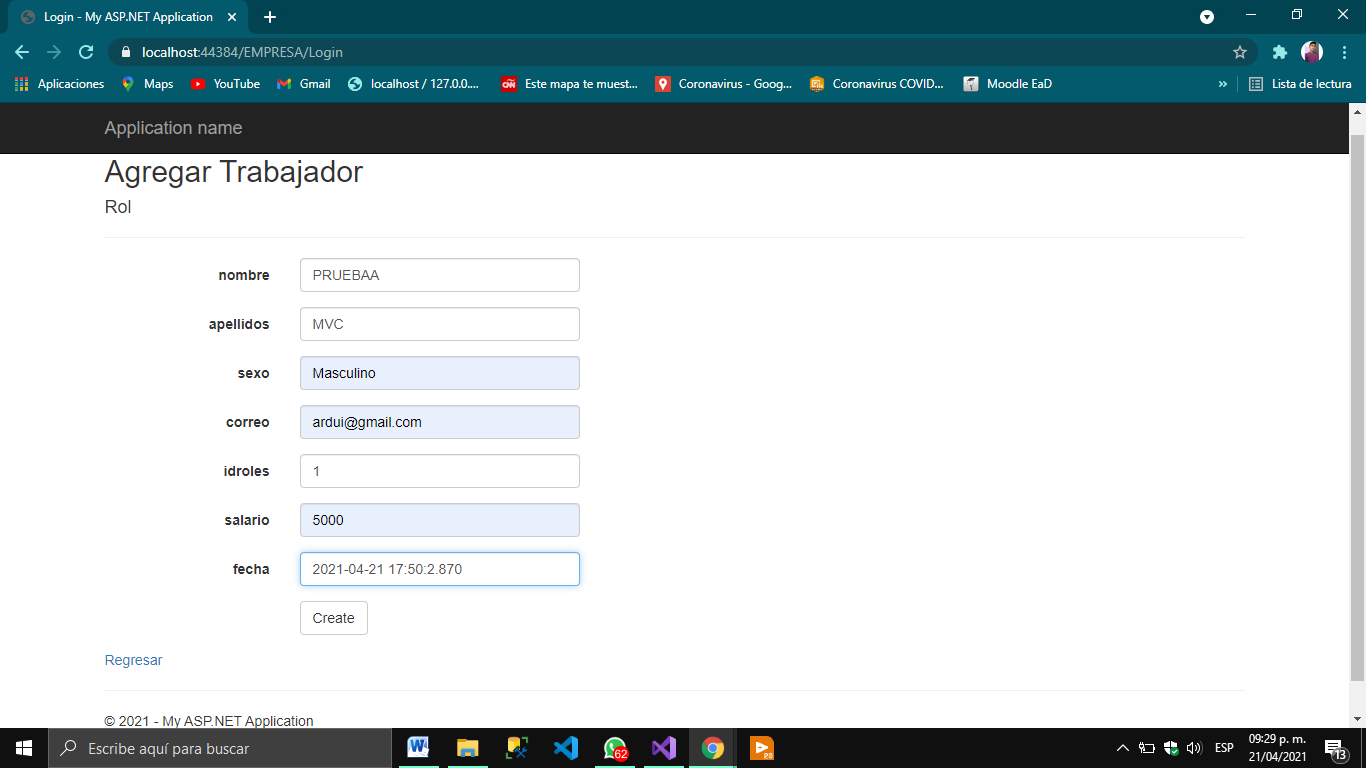
### Indice

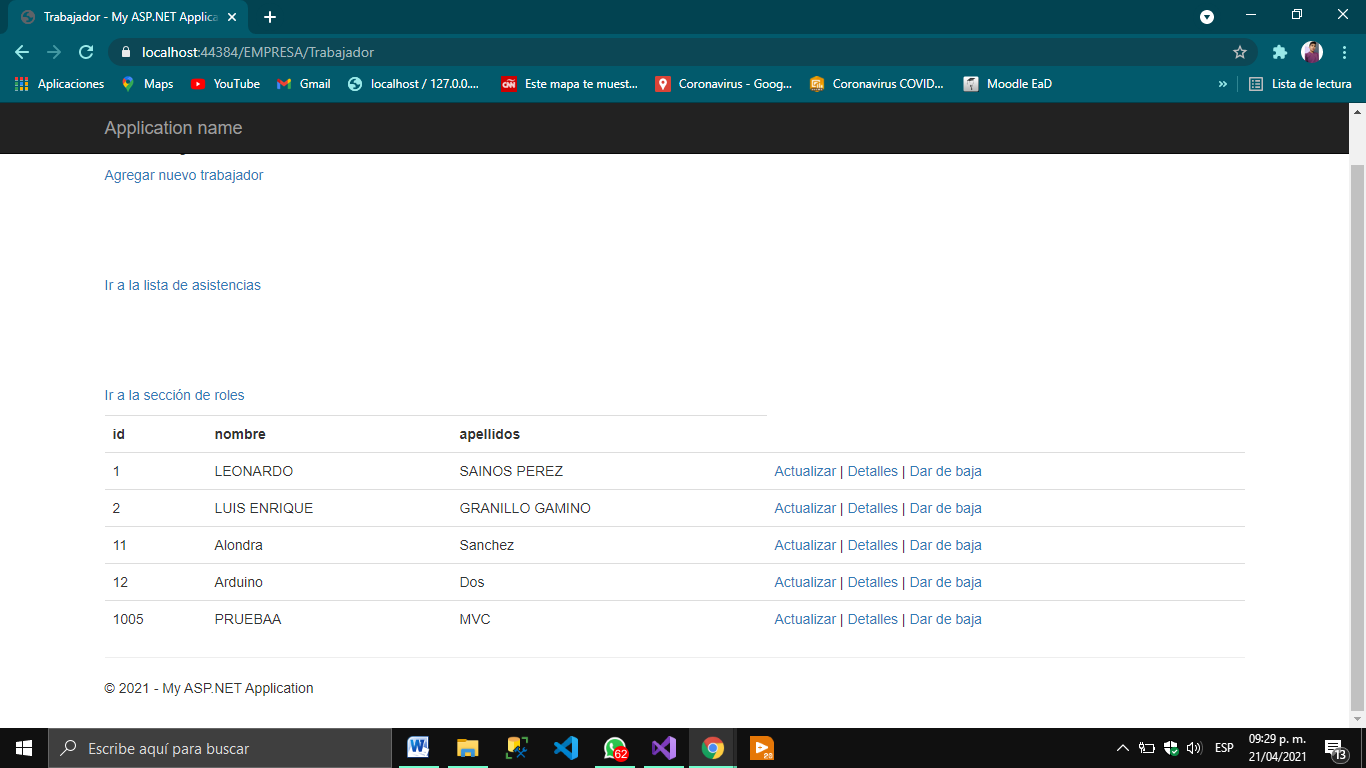


### Detalles

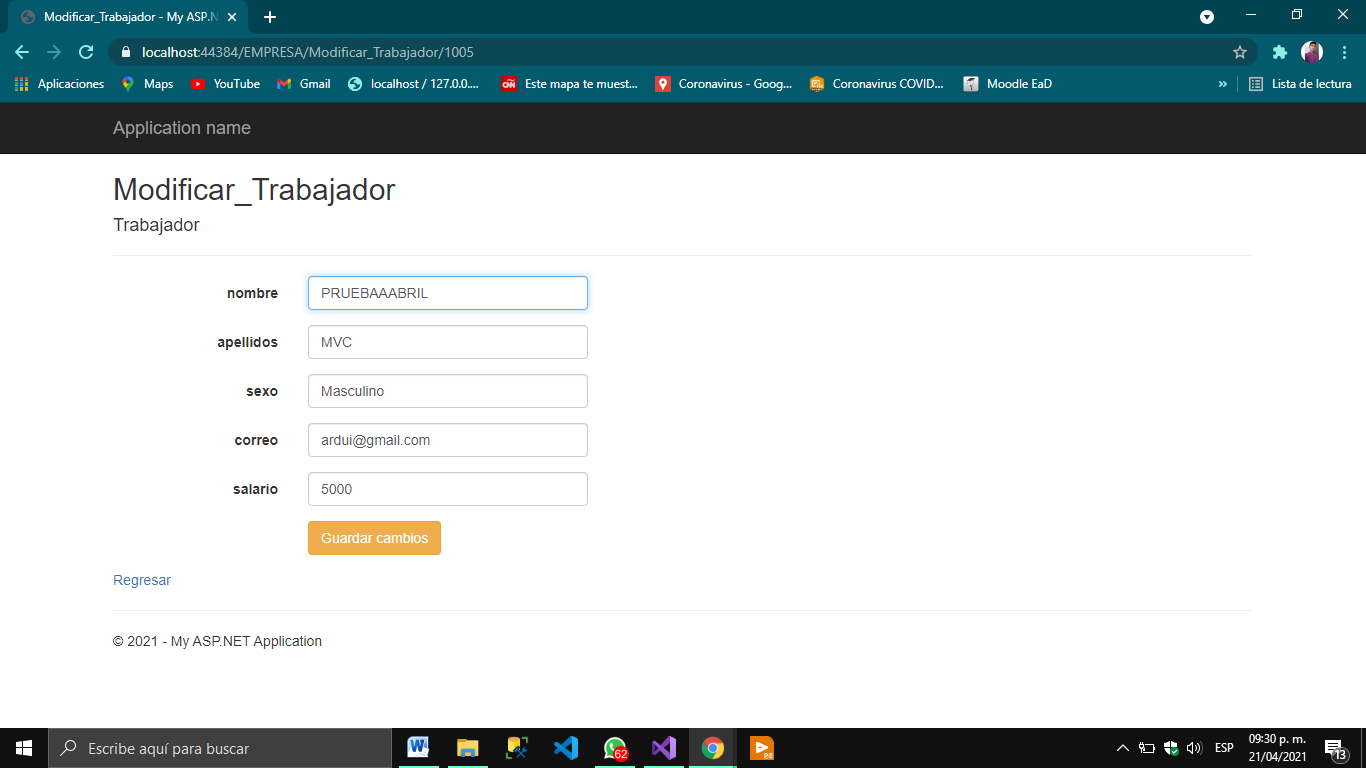


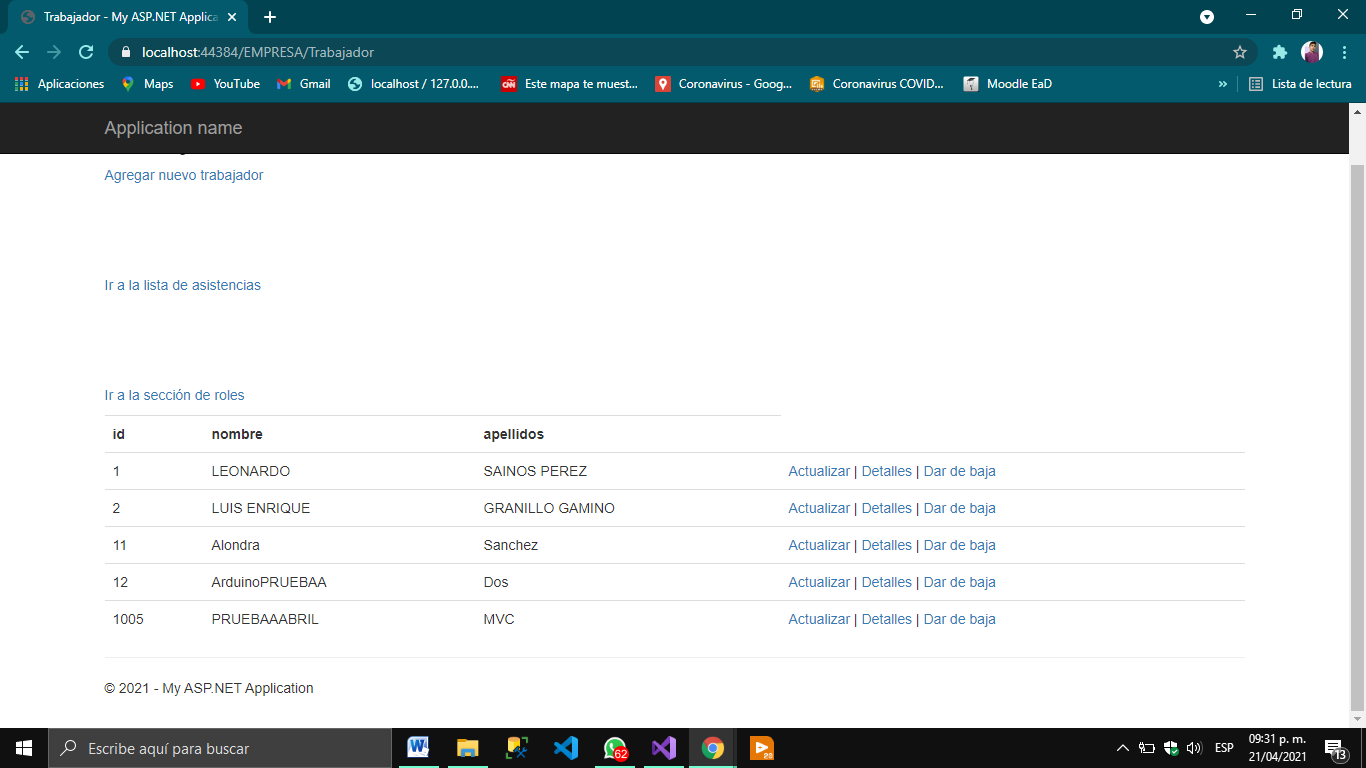
### Crear



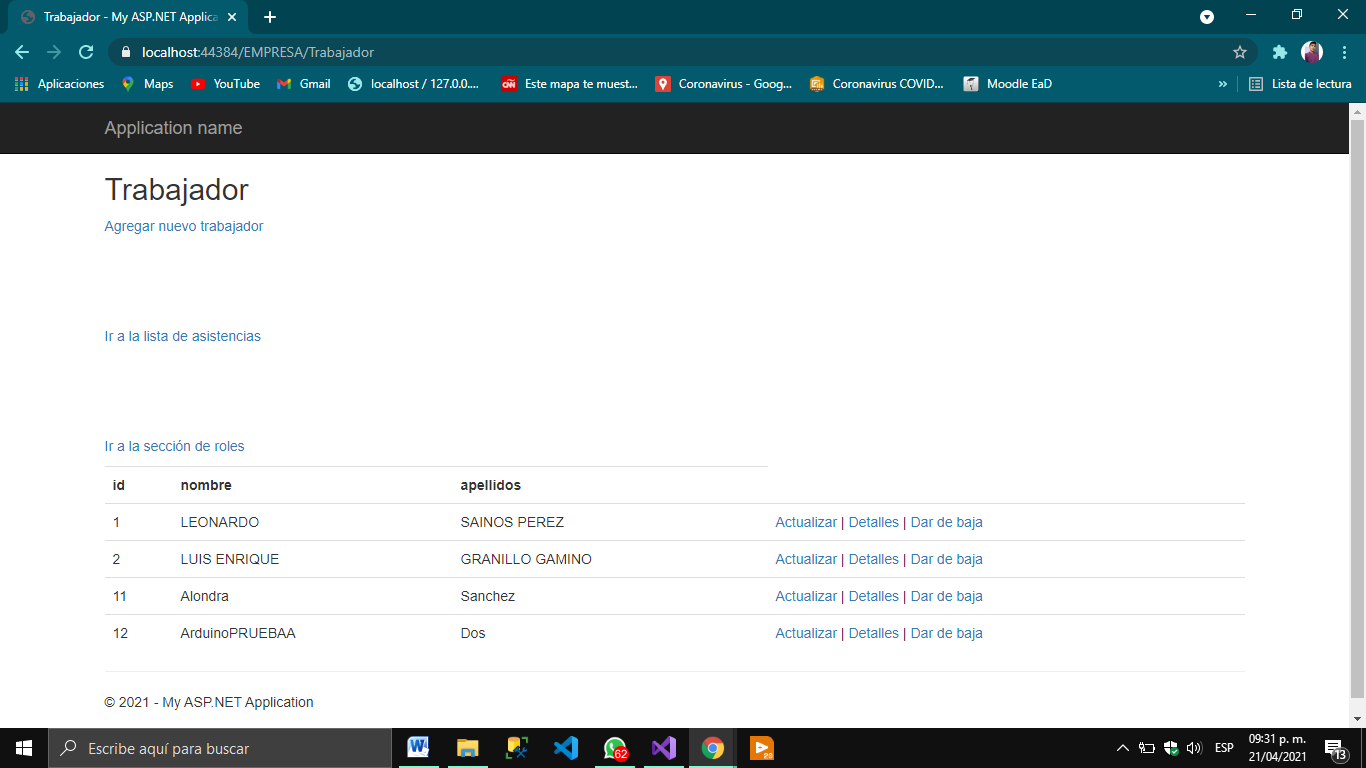


### Actualizar



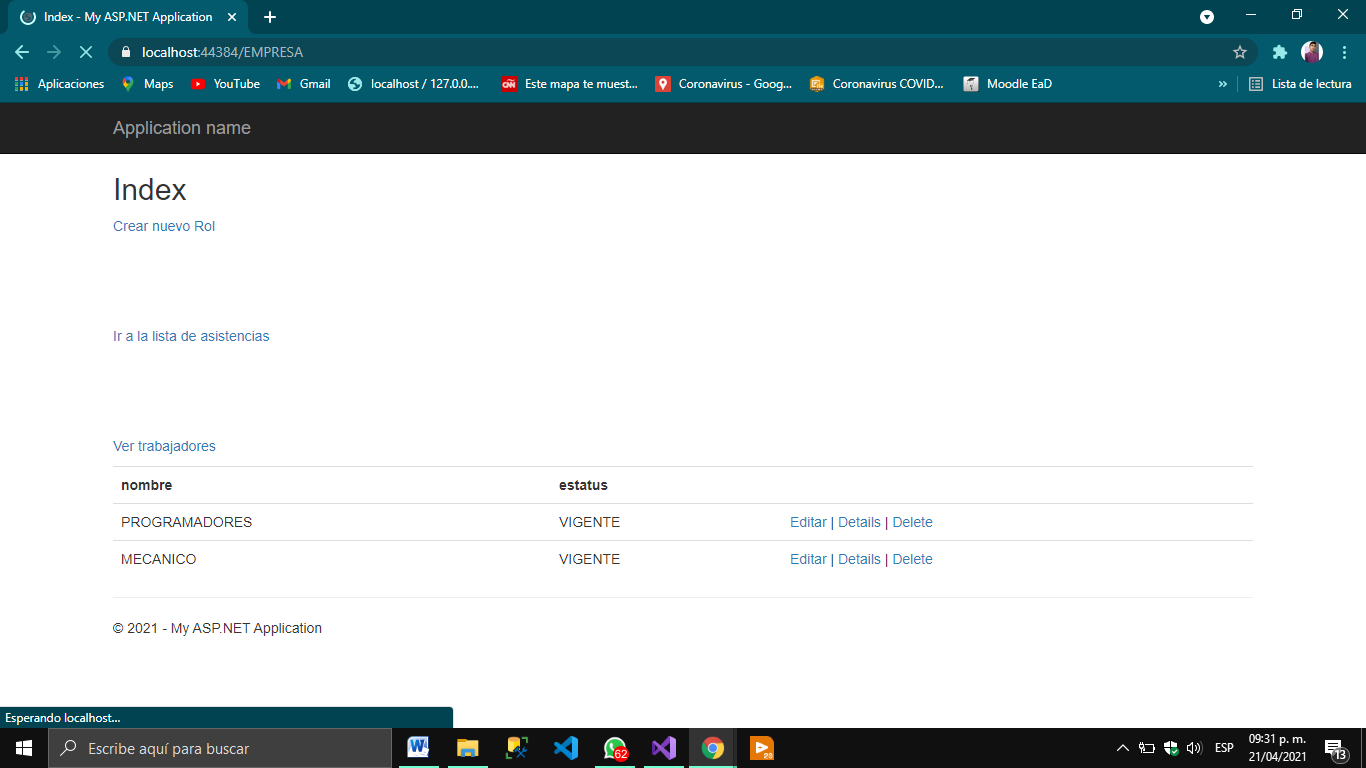


### Eliminar

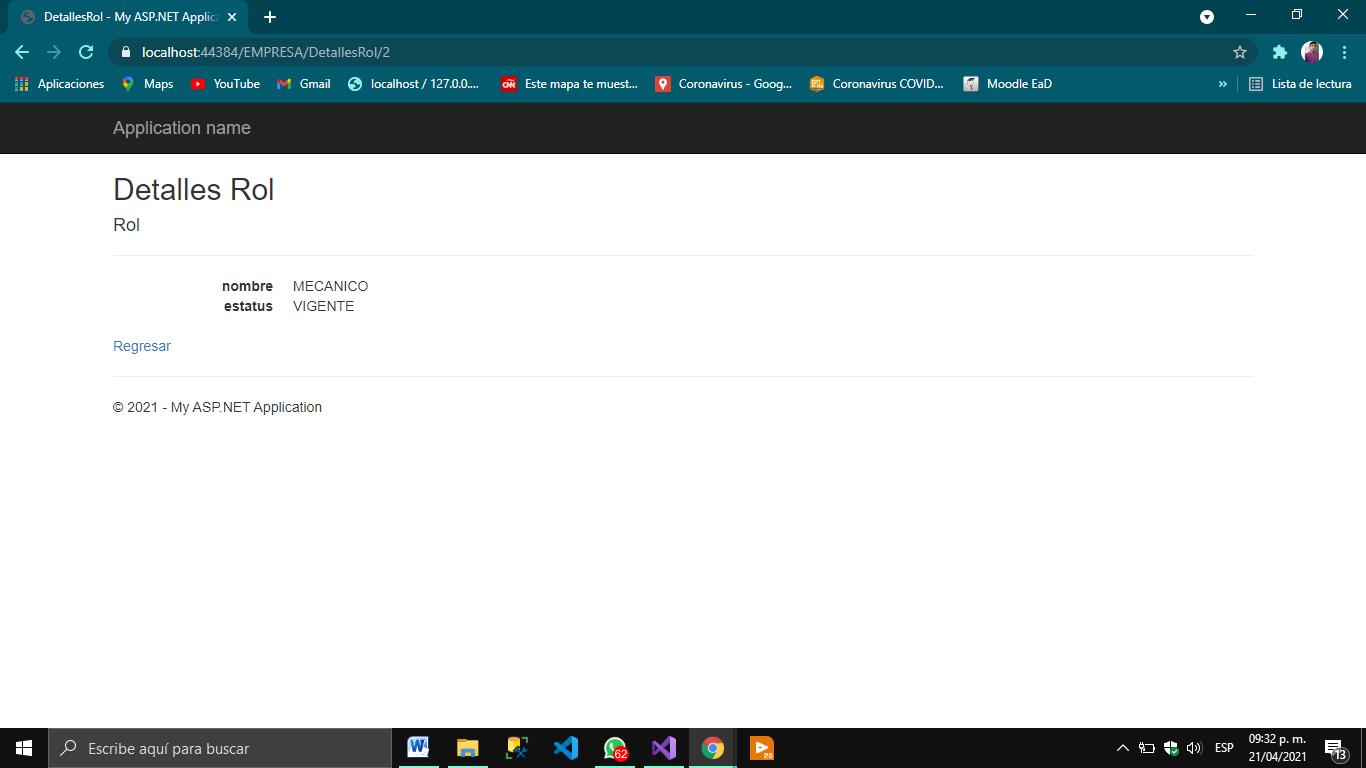


## ROL.

### Indice

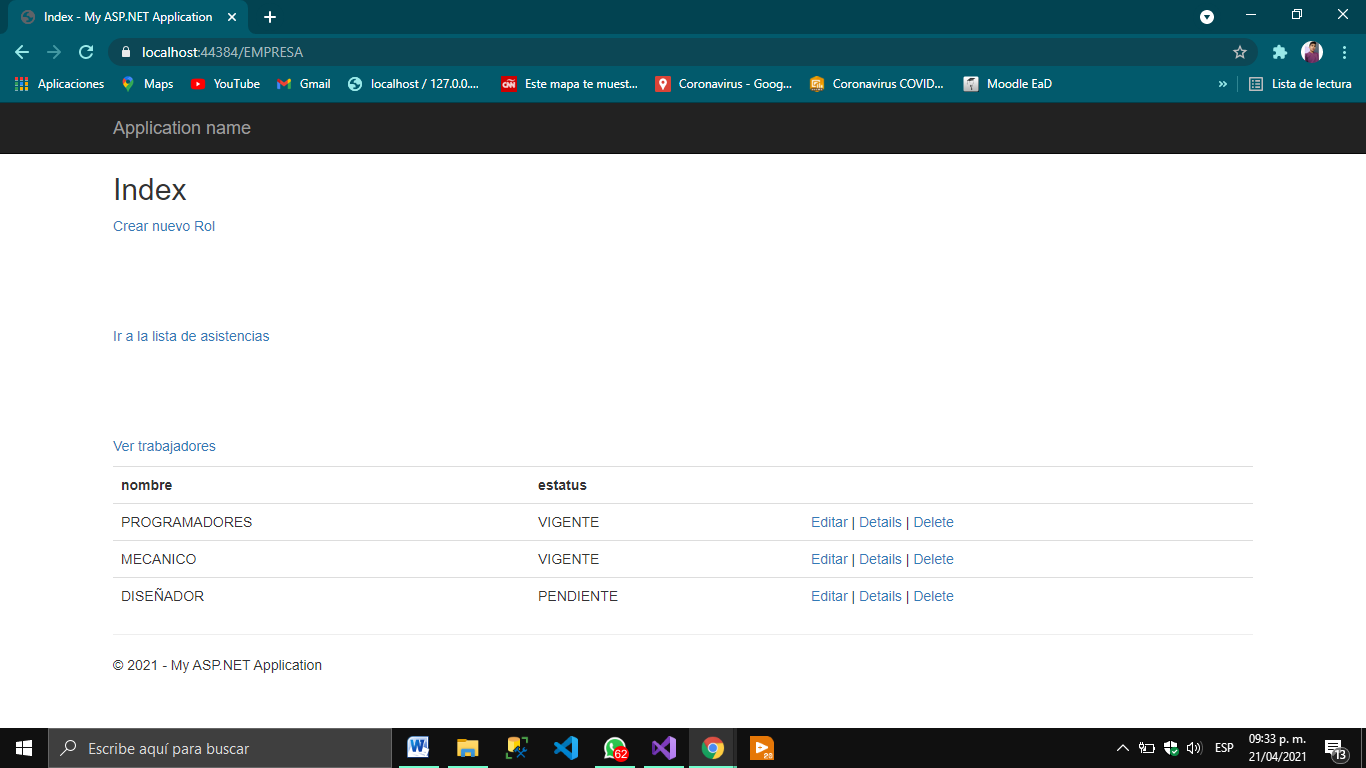


### Detalles

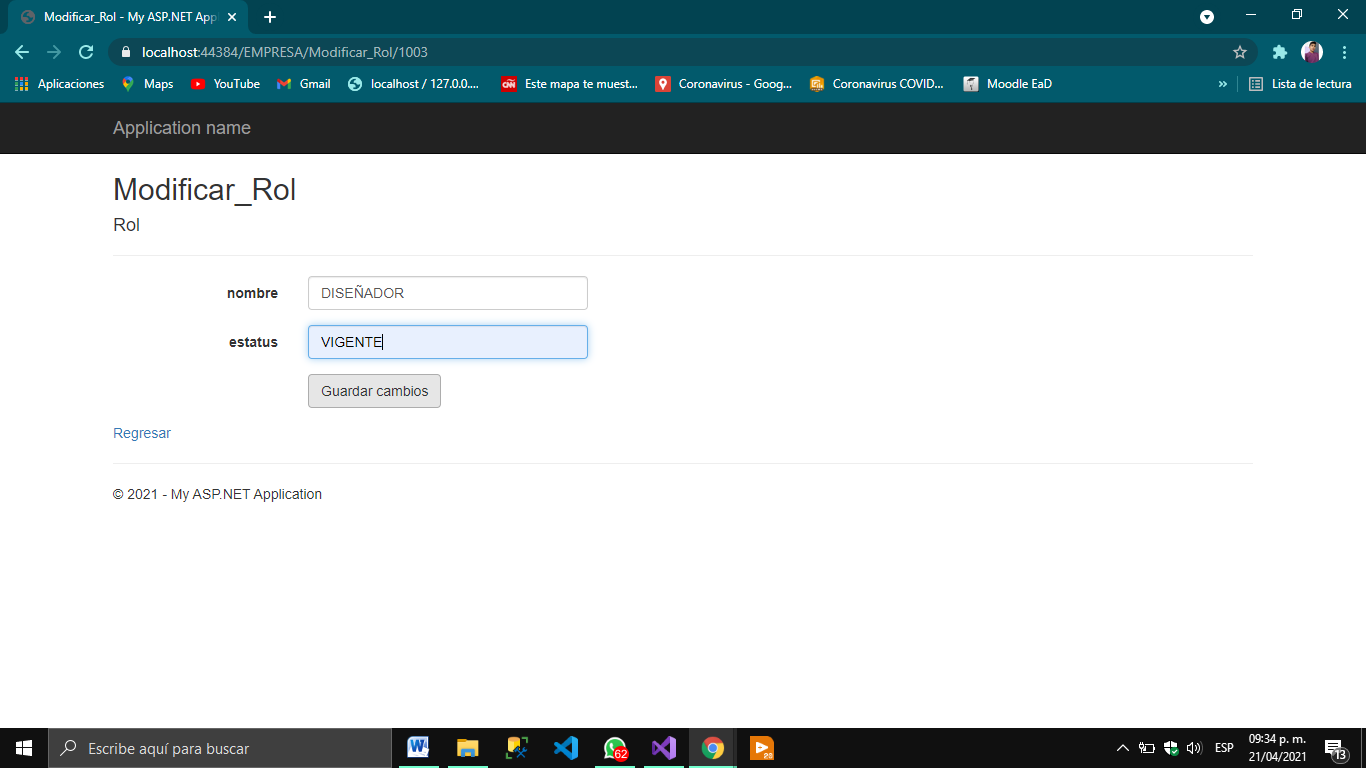


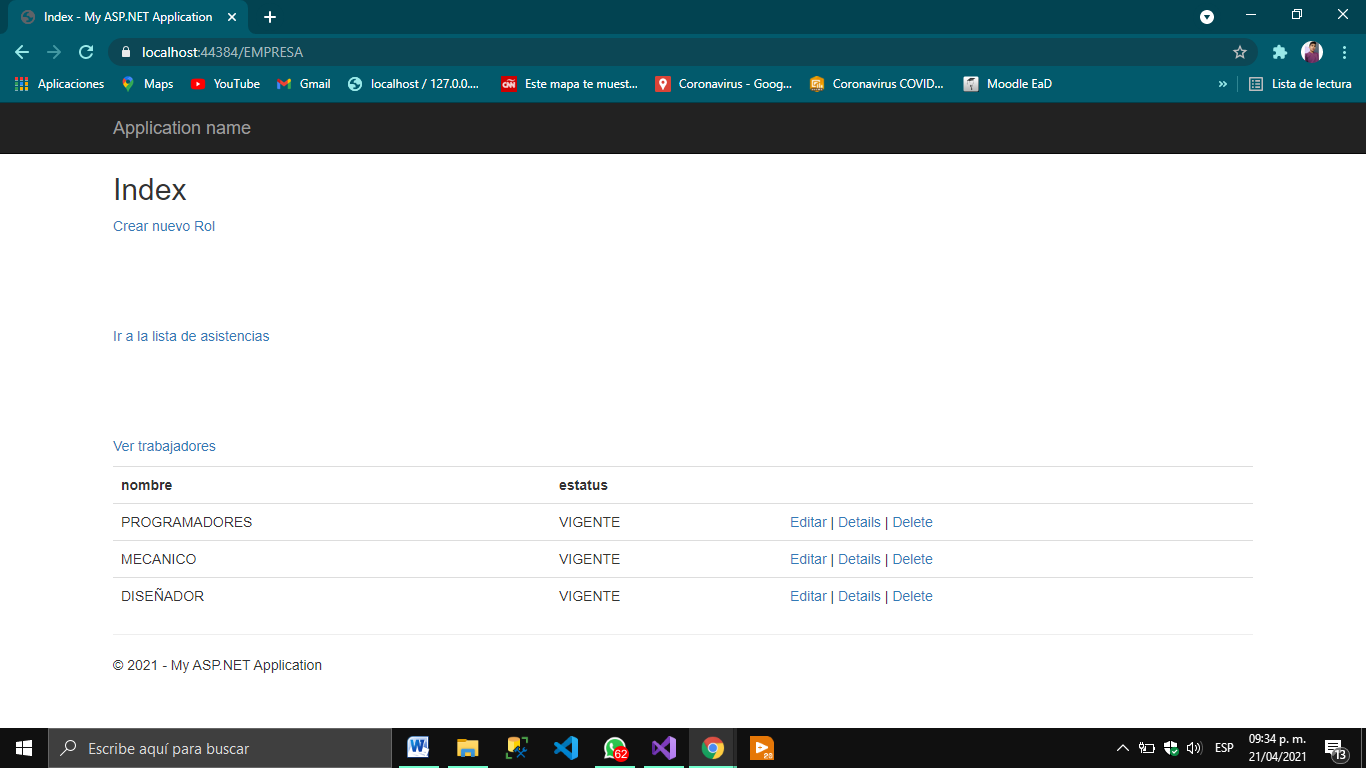
### Crear



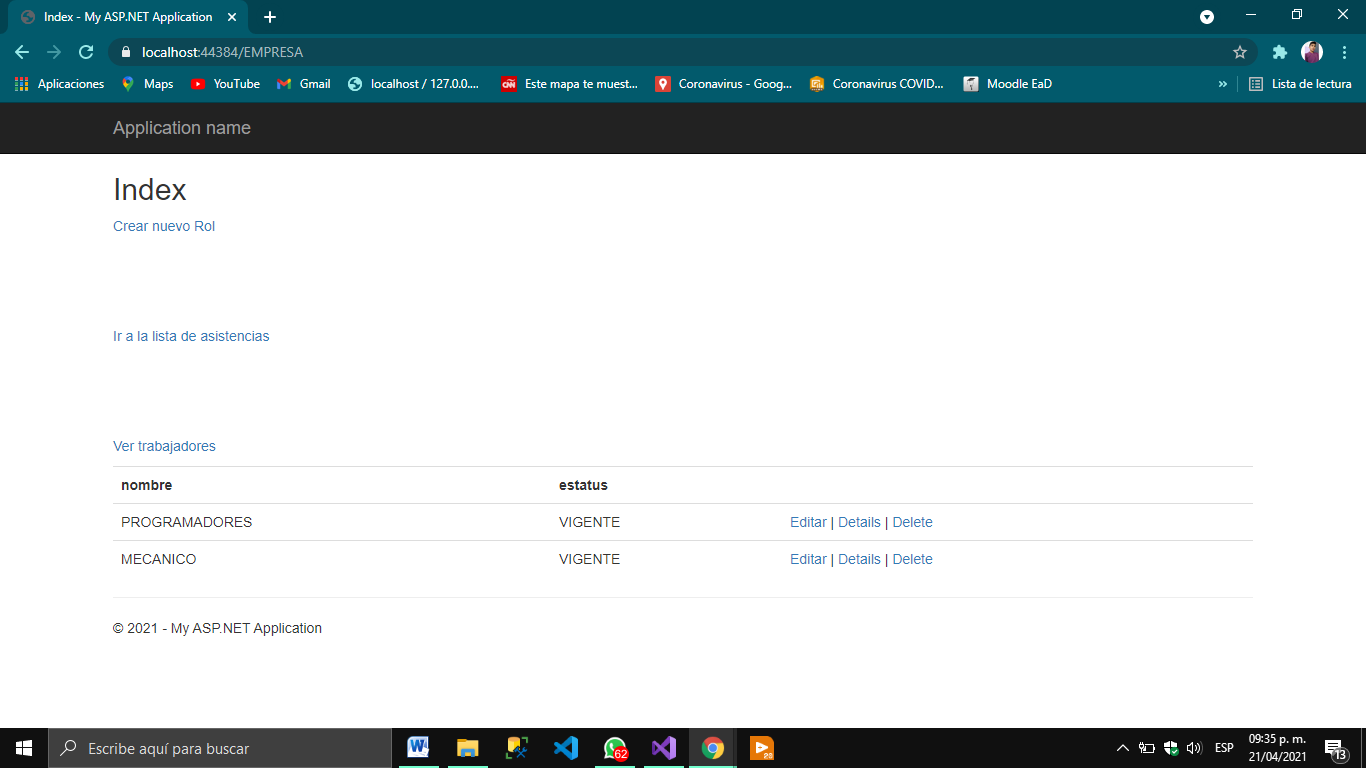


### Actualizar





### Eliminar



# CÓDIGO DEL MODELO

//------------------------------------------------------------------------------

// <auto-generated>

// Este código se generó a partir de una plantilla.

//

// Los cambios manuales en este archivo pueden causar un comportamiento inesperado de la aplicación.

// Los cambios manuales en este archivo se sobrescribirán si se regenera el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace PRODUCTO3.Models

{

using System;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

public partial class EMPRESAEntities5 : DbContext

{

public EMPRESAEntities5() : base("name=EMPRESAEntities5")

{

this.Configuration.LazyLoadingEnabled = false;

}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

public virtual DbSet<Asiste> Asiste { get; set; }

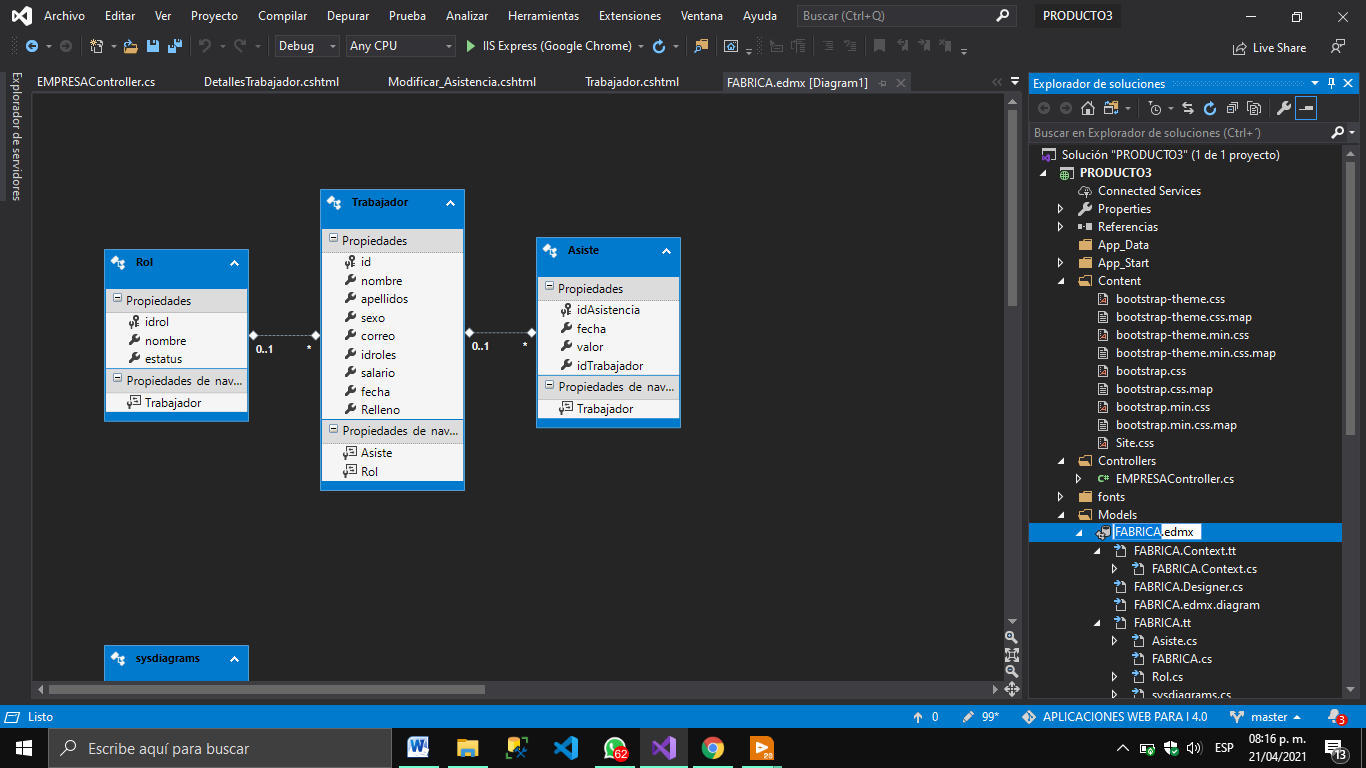
public virtual DbSet<Rol> Rol { get; set; }

public virtual DbSet<sysdiagrams> sysdiagrams { get; set; }

public virtual DbSet<Trabajador> Trabajador { get; set; }

}

}



# CÓDIGO DEL CONTROLADOR.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using PRODUCTO3.Models;

namespace PRODUCTO3.Controllers

{

public class EMPRESAController : Controller

{

public object Configuration { get; private set; }

// GET: EMPRESA

// GET: Persona

public ActionResult Index()

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

List<Rol> lista = bd.Rol.Where(a => a.idrol >= 0).ToList();

return View(lista);

}

}

catch (Exception t)

{

throw ;

}

}

[HttpGet]

public ActionResult Agregar\_Rol()

{

return View();

}

[HttpPost]

public ActionResult Agregar\_Rol(Rol per)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

bd.Rol.Add(per);

bd.SaveChanges();

return RedirectToAction("Index");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al crear el Rol ", t);

return View();

}

}

[HttpGet]

public ActionResult Modificar\_Rol(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Rol obj = bd.Rol.Find(id); // Debe ser Orimary key afuerza

return View(obj);

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al actualizar el Rol ", t);

return View();

}

}

[HttpPost]

public ActionResult Modificar\_Rol(Rol obj)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Rol obje = bd.Rol.Find(obj.idrol);

obje.nombre = obj.nombre;

obje.estatus = obj.estatus;

bd.SaveChanges();

return RedirectToAction("Index");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al actualizar persona ", t);

return View();

}

}

public ActionResult Eliminar\_Rol(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Rol Existe = bd.Rol.Find(id);

bd.Rol.Remove(Existe);

bd.SaveChanges();

return RedirectToAction("Index");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al eliminar ", t);

return View();

}

}

public ActionResult DetallesRol(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Rol Existe = bd.Rol.Find(id);

return View(Existe);

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al mostrar los datos de este registro ", t);

return View();

}

}

public ActionResult Asiste()

{

try

{

//this.Configuration.LazyLoadingEnabled = false;

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

List<Asiste> lista = bd.Asiste.Where(a => a.idAsistencia >= 0).ToList();

return View(lista);

}

}

catch (Exception t)

{

throw;

}

}

[HttpGet]

public ActionResult Dar\_Asistencia()

{

return View();

}

[HttpPost]

public ActionResult Dar\_Asistencia(Asiste asi)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

bd.Asiste.Add(asi);

bd.SaveChanges();

return RedirectToAction("Asiste");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al dar la asistencia ", t);

return View();

}

}

public ActionResult DetallesAsiste(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Asiste Existe = bd.Asiste.Find(id);

return View(Existe);

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al mostrar los datos de este registro ", t);

return View();

}

}

public ActionResult BorrarAsistencia(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Asiste Existe = bd.Asiste.Find(id);

bd.Asiste.Remove(Existe);

bd.SaveChanges();

return RedirectToAction("Asiste");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al eliminar ", t);

return View();

}

}

public ActionResult Trabajador()

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

List<Trabajador> lista = bd.Trabajador.Where(a => a.id >= 0).ToList();

return View(lista);

}

}

catch (Exception t)

{

throw;

}

}

public ActionResult DetallesTrabajador(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Trabajador Existe = bd.Trabajador.Find(id);

return View(Existe);

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al mostrar los datos de este registro ", t);

return View();

}

}

public ActionResult DarBaja(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Trabajador Existe = bd.Trabajador.Find(id);

bd.Trabajador.Remove(Existe);

bd.SaveChanges();

return RedirectToAction("Trabajador");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al eliminar ", t);

return View();

}

}

[HttpGet]

public ActionResult Login()

{

return View();

}

[HttpPost]

public ActionResult Login(Trabajador asi)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

bd.Trabajador.Add(asi);

bd.SaveChanges();

return RedirectToAction("Login");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al loguearse ", t);

return View();

}

}

[HttpGet]

public ActionResult Modificar\_Trabajador(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Trabajador obj = bd.Trabajador.Find(id); // Debe ser Orimary key afuerza

return View(obj);

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al actualizar el Rol ", t);

return View();

}

}

[HttpPost]

public ActionResult Modificar\_Trabajador(Trabajador obj)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Trabajador obje = bd.Trabajador.Find(obj.id);

obje.nombre = obj.nombre;

obje.apellidos = obj.apellidos;

obje.sexo = obj.sexo;

obje.correo = obj.correo;

obje.salario = obj.salario;

bd.SaveChanges();

return RedirectToAction("Trabajador");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al actualizar persona ", t);

return View();

}

}

[HttpGet]

public ActionResult Modificar\_Asistencia(int id)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Asiste obj = bd.Asiste.Find(id); // Debe ser Orimary key afuerza

return View(obj);

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al actualizar el Rol ", t);

return View();

}

}

[HttpPost]

public ActionResult Modificar\_Asistencia(Asiste obj)

{

try

{

using (EMPRESAEntities5 bd = new EMPRESAEntities5())

{

Asiste obje = bd.Asiste.Find(obj.idAsistencia);

obje.valor = obj.valor;

bd.SaveChanges();

return RedirectToAction("Asiste");

}

}

catch (Exception t)

{

ModelState.AddModelError("Error al actualizar persona ", t);

return View();

}

}

}

}

# CÓDIGO DE LAS VISTAS

## VISTAS DE EMPLEADO

### Empleado

@model IEnumerable<PRODUCTO3.Models.Trabajador>

@{

ViewBag.Title = "Trabajador";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Trabajador</h2>

<p>

@Html.ActionLink("Agregar nuevo trabajador ", "Login")

</p>

<br />

<br />

<br />

<br />

<p>

@Html.ActionLink("Ir a la lista de asistencias ", "Asiste")

</p>

<br />

<br />

<br />

<br />

<p>

@Html.ActionLink("Ir a la sección de roles", "Index")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.id)

</th>

<th>

@Html.DisplayNameFor(model => model.nombre)

</th>

<th>

@Html.DisplayNameFor(model => model.apellidos)

</th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.id)

</td>

<td>

@Html.DisplayFor(modelItem => item.nombre)

</td>

<td>

@Html.DisplayFor(modelItem => item.apellidos)

</td>

<td>

@Html.ActionLink("Actualizar", "Modificar\_Trabajador", new { id = item.id }) |

@Html.ActionLink("Detalles", "DetallesTrabajador", new { id = item.id }) |

@Html.ActionLink("Dar de baja", "DarBaja", new { id = item.id })

</td>

</tr>

}

</table>

### Actualizar empleado

@model PRODUCTO3.Models.Trabajador

@{

ViewBag.Title = "Modificar\_Trabajador";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Modificar\_Trabajador</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Trabajador</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.id)

<div class="form-group">

@Html.LabelFor(model => model.nombre, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.nombre, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.nombre, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.apellidos, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.apellidos, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.apellidos, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.sexo, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.sexo, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.sexo, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.correo, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.correo, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.correo, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.salario, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.salario, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.salario, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Guardar cambios" class="btn btn-warning" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Regresar", "Trabajador")

</div>

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

### Crear Nuevo

@model PRODUCTO3.Models.Trabajador

@{

ViewBag.Title = "Login";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Agregar Trabajador</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Rol</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.nombre, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.nombre, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.nombre, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.apellidos, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.apellidos, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.apellidos, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.sexo, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.sexo, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.sexo, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.correo, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.correo, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.correo, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.idroles, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.idroles, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.idroles, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.salario, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.salario, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.salario, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.fecha, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.fecha, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.fecha, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Create" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Regresar", "Trabajador")

</div>

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

### Detalles empleado

@model PRODUCTO3.Models.Trabajador

@{

ViewBag.Title = "DetallesTrabajador";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>DetallesTrabajador</h2>

<div>

<h4>Trabajador</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.nombre)

</dt>

<dd>

@Html.DisplayFor(model => model.nombre)

</dd>

<dt>

@Html.DisplayNameFor(model => model.apellidos)

</dt>

<dd>

@Html.DisplayFor(model => model.apellidos)

</dd>

<dt>

@Html.DisplayNameFor(model => model.sexo)

</dt>

<dd>

@Html.DisplayFor(model => model.sexo)

</dd>

<dt>

@Html.DisplayNameFor(model => model.correo)

</dt>

<dd>

@Html.DisplayFor(model => model.correo)

</dd>

<dt>

@Html.DisplayNameFor(model => model.salario)

</dt>

<dd>

@Html.DisplayFor(model => model.salario)

</dd>

<dt>

@Html.DisplayNameFor(model => model.fecha)

</dt>

<dd>

@Html.DisplayFor(model => model.fecha)

</dd>

<dt>

@Html.DisplayNameFor(model => model.idroles)

</dt>

<dd>

@Html.DisplayFor(model => model.idroles)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Regresar", "Trabajador")

</p>

### Eliminar empleado

@model PRODUCTO3.Models.Trabajador

@{

ViewBag.Title = "DarBaja";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>DarBaja</h2>

<h3>¿Seguro que quieres dar de baja a este empleado?</h3>

<div>

<h4>Trabajador</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.nombre)

</dt>

<dd>

@Html.DisplayFor(model => model.nombre)

</dd>

<dt>

@Html.DisplayNameFor(model => model.apellidos)

</dt>

<dd>

@Html.DisplayFor(model => model.apellidos)

</dd>

<dt>

@Html.DisplayNameFor(model => model.sexo)

</dt>

<dd>

@Html.DisplayFor(model => model.sexo)

</dd>

<dt>

@Html.DisplayNameFor(model => model.correo)

</dt>

<dd>

@Html.DisplayFor(model => model.correo)

</dd>

<dt>

@Html.DisplayNameFor(model => model.salario)

</dt>

<dd>

@Html.DisplayFor(model => model.salario)

</dd>

<dt>

@Html.DisplayNameFor(model => model.fecha)

</dt>

<dd>

@Html.DisplayFor(model => model.fecha)

</dd>

<dt>

@Html.DisplayNameFor(model => model.idroles)

</dt>

<dd>

@Html.DisplayFor(model => model.idroles)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Dar de baja" class="btn btn-warning" /> |

@Html.ActionLink("Regresar", "Trabajador")

</div>

}

</div>

## VISTAS DE ASISTE

### Asiste

@model IEnumerable<PRODUCTO3.Models.Asiste>

@{

ViewBag.Title = "Asiste";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Asiste</h2>

<p>

@Html.ActionLink("Asignar nuevo registro de asistencia ", "Dar\_Asistencia")

</p>

<br />

<br />

<br />

<br />

<p>

@Html.ActionLink("Ver trabajadores ", "Trabajador")

</p>

<br />

<br />

<br />

<br />

<p>

@Html.ActionLink("Ir a la sección de roles", "Index")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.idTrabajador)

</th>

<th>

@Html.DisplayNameFor(model => model.fecha)

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.idTrabajador)

</td>

<td>

@Html.DisplayFor(modelItem => item.fecha)

</td>

<td>

@Html.ActionLink("Editar", "Modificar\_Asistencia", new { id = item.idAsistencia }) |

@Html.ActionLink("Detalles", "DetallesAsiste", new { id = item.idAsistencia }) |

@Html.ActionLink("Borrar", "BorrarAsistencia", new { id = item.idAsistencia })

</td>

</tr>

}

</table>

### Dar asistencia

@model PRODUCTO3.Models.Asiste

@{

ViewBag.Title = "Dar\_Asistencia";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Dar\_Asistencia</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Asiste</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.idTrabajador, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.idTrabajador, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.idTrabajador, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.valor, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.valor, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.valor, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Crear" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Regresar", "Asiste")

</div>

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

### Detalles

@model PRODUCTO3.Models.Asiste

@{

ViewBag.Title = "DetallesAsiste";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>DetallesAsiste</h2>

<div>

<h4>Asiste</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.idAsistencia)

</dt>

<dd>

@Html.DisplayFor(model => model.idAsistencia)

</dd>

<dt>

@Html.DisplayNameFor(model => model.fecha)

</dt>

<dd>

@Html.DisplayFor(model => model.fecha)

</dd>

<dt>

@Html.DisplayNameFor(model => model.valor)

</dt>

<dd>

@Html.DisplayFor(model => model.valor)

</dd>

<dt>

@Html.DisplayNameFor(model => model.idTrabajador)

</dt>

<dd>

@Html.DisplayFor(model => model.idTrabajador)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Regresar", "Asiste")

</p>

### Actualizar\_asistencia

@model PRODUCTO3.Models.Asiste

@{

ViewBag.Title = "Modificar\_Asistencia";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Modificar\_Asistencia</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Asiste</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.idAsistencia)

<div class="form-group">

@Html.LabelFor(model => model.valor, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.valor, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.valor, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Regresar", "Asiste")

</div>

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

### Eliminar\_Asistencia

@model PRODUCTO3.Models.Asiste

@{

ViewBag.Title = "BorrarAsistencia";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>BorrarAsistencia</h2>

<h3>¿Estás seguro que deseas eliminar este registro?</h3>

<div>

<h4>Asiste</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.fecha)

</dt>

<dd>

@Html.DisplayFor(model => model.fecha)

</dd>

<dt>

@Html.DisplayNameFor(model => model.idAsistencia)

</dt>

<dd>

@Html.DisplayFor(model => model.idAsistencia)

</dd>

<dt>

@Html.DisplayNameFor(model => model.valor)

</dt>

<dd>

@Html.DisplayFor(model => model.valor)

</dd>

<dt>

@Html.DisplayNameFor(model => model.idTrabajador)

</dt>

<dd>

@Html.DisplayFor(model => model.idTrabajador)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Delete" class="btn btn-default" /> |

@Html.ActionLink("Back to List", "Asiste")

</div>

}

</div>

## VISTAS DE ROL

### Index

@model IEnumerable<PRODUCTO3.Models.Rol>

@{

ViewBag.Title = "Index";

}

<h2>Index</h2>

<p>

@Html.ActionLink("Crear nuevo Rol", "Agregar\_Rol")

</p>

<br />

<br />

<br />

<br />

<p>

@Html.ActionLink("Ir a la lista de asistencias ", "Asiste")

</p>

<br />

<br />

<br />

<br />

<p>

@Html.ActionLink("Ver trabajadores", "Trabajador")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.nombre)

</th>

<th>

@Html.DisplayNameFor(model => model.estatus)

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.nombre)

</td>

<td>

@Html.DisplayFor(modelItem => item.estatus)

</td>

<td>

@Html.ActionLink("Editar", "Modificar\_Rol", new { id = item.idrol }) |

@Html.ActionLink("Details", "DetallesRol", new { id = item.idrol }) |

@Html.ActionLink("Delete", "Eliminar\_Rol", new { id = item.idrol })

</td>

</tr>

}

</table>

### Detalles

@model PRODUCTO3.Models.Rol

@{

ViewBag.Title = "DetallesRol";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Detalles Rol</h2>

<div>

<h4>Rol</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.nombre)

</dt>

<dd>

@Html.DisplayFor(model => model.nombre)

</dd>

<dt>

@Html.DisplayNameFor(model => model.estatus)

</dt>

<dd>

@Html.DisplayFor(model => model.estatus)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Regresar", "Index")

</p>

### Crear

@model PRODUCTO3.Models.Rol

@{

ViewBag.Title = "Agregar\_Rol";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Agregar\_Rol</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Rol</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.nombre, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.nombre, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.nombre, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.estatus, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.estatus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.estatus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Create" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

### Actualizar

@model PRODUCTO3.Models.Rol

@{

ViewBag.Title = "Modificar\_Rol";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Modificar\_Rol</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Rol</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

@Html.HiddenFor(model => model.idrol)

<div class="form-group">

@Html.LabelFor(model => model.nombre, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.nombre, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.nombre, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.estatus, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.estatus, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.estatus, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Guardar cambios" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Regresar", "Index")

</div>

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

### Eliminar

@model PRODUCTO3.Models.Rol

@{

ViewBag.Title = "Eliminar\_Rol";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Eliminar\_Rol</h2>

<h3>Are you sure you want to delete this?</h3>

<div>

<h4>Rol</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.nombre)

</dt>

<dd>

@Html.DisplayFor(model => model.nombre)

</dd>

<dt>

@Html.DisplayNameFor(model => model.estatus)

</dt>

<dd>

@Html.DisplayFor(model => model.estatus)

</dd>

</dl>

@using (Html.BeginForm()) {

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Delete" class="btn btn-default" /> |

@Html.ActionLink("Regresar", "Index")

</div>

}

</div>

# CONCLUSIÓN

Como conslusión puedo rescatar que me agrado mucho la parte de entity ya que nos facilita muchísimo el trabajo para trabajar con SQL , lo importante es que ahorra tiempo y parece fácil de usar, vi como desventaja que el proyecto no puedes usarlo directamente de otra computadora ya que tienes que mover unas cosas del proyecto ya que desde un inicio está configurado a tu máquina, pero de ahí en fuera todo me agrado, sólo se me complico una parte al crear mis vistas de las tablas relacionadas y el poder implementar bootstrap o mejor dicho hacer referencias a tu archivo de trabajo ya que intenté jalar algunas clases para estilo y no se podía, pero al final lo logré y comenzare a usar más entity.