

controladores de Ciudades y Categorías

Controlador de Categorías

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Rendering;
using Microsoft.EntityFrameworkCore;
using EventoTec.web.Models;
using EventoTec.web.Models.entities;

namespace EventoTec.web.Controllers
{
    public class CategoriesController : Controller
    {
        private readonly DataDbContext _context;

        public CategoriesController(DataDbContext context)
        {
            _context = context;
        }

        // GET: Categories
        public async Task<IActionResult> Index()
        {
            return View(await _context.Category.ToListAsync());
        }

        // GET: Categories/Details/5
        public async Task<IActionResult> Details(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var category = await _context.Category
                .FirstOrDefaultAsync(m => m.Id == id);
            if (category == null)
            {
                return NotFound();
            }

            return View(category);
        }

        // GET: Categories/Create
        public IActionResult Create()
        {
            return View();
        }

        // POST: Categories/Create
        // To protect from overposting attacks, please enable the specific properties you want to bind to, for
        // more details see http://go.microsoft.com/fwlink/?LinkId=317598.
        [HttpPost]
        [ValidateAntiForgeryToken]
```

```

public async Task<ActionResult> Create([Bind("Id,Name,Description")] Category category)
{
    if (ModelState.IsValid)
    {
        _context.Add(category);
        await _context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    return View(category);
}

// GET: Categories/Edit/5
public async Task<ActionResult> Edit(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var category = await _context.Category.FindAsync(id);
    if (category == null)
    {
        return NotFound();
    }
    return View(category);
}

// POST: Categories/Edit/5
// To protect from overposting attacks, please enable the specific properties you want to bind to, for
// more details see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Edit(int id, [Bind("Id,Name,Description")] Category category)
{
    if (id != category.Id)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(category);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!CategoryExists(category.Id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(category);
}

```

```

// GET: Categories/Delete/5
public async Task<IActionResult> Delete(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var category = await _context.Category
        .FirstOrDefaultAsync(m => m.Id == id);
    if (category == null)
    {
        return NotFound();
    }

    return View(category);
}

// POST: Categories/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public async Task<IActionResult> DeleteConfirmed(int id)
{
    var category = await _context.Category.FindAsync(id);
    _context.Category.Remove(category);
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}

private bool CategoryExists(int id)
{
    return _context.Category.Any(e => e.Id == id);
}
}

```

Controlador de Ciudades

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Rendering;
using Microsoft.EntityFrameworkCore;
using EventoTec.web.Models;
using EventoTec.web.Models.entities;

namespace EventoTec.web.Controllers
{
    public class CitiesController : Controller
    {
        private readonly DataDbContext _context;

        public CitiesController(DataDbContext context)
        {
            _context = context;
        }

        // GET: Cities
        public async Task<IActionResult> Index()
        {
            return View(await _context.City.ToListAsync());
        }

        // GET: Cities/Details/5
        public async Task<IActionResult> Details(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var city = await _context.City
                .FirstOrDefaultAsync(m => m.id == id);
            if (city == null)
            {
                return NotFound();
            }

            return View(city);
        }

        // GET: Cities/Create
        public IActionResult Create()
        {
            return View();
        }

        // POST: Cities/Create
        // To protect from overposting attacks, please enable the specific properties you want to bind to, for
        // more details see http://go.microsoft.com/fwlink/?LinkId=317598.
        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Create([Bind("id,Name,Descripcion,Slung")] City city)
        {
            if (ModelState.IsValid)
            {

```

```

        _context.Add(city);
        await _context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    return View(city);
}

```

// GET: Cities/Edit/5

```

public async Task<ActionResult> Edit(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var city = await _context.City.FindAsync(id);
    if (city == null)
    {
        return NotFound();
    }
    return View(city);
}

```

// POST: Cities/Edit/5

// To protect from overposting attacks, please enable the specific properties you want to bind to, for
 // more details see <http://go.microsoft.com/fwlink/?LinkId=317598>.

[HttpPost]

[ValidateAntiForgeryToken]

```

public async Task<ActionResult> Edit(int id, [Bind("id,Name,Descripcion,Slung")] City city)

```

```

{
    if (id != city.id)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(city);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!CityExists(city.id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(city);
}

```

// GET: Cities/Delete/5

```

public async Task<ActionResult> Delete(int? id)
{
    if (id == null)

```

```

    {
        return NotFound();
    }

    var city = await _context.City
        .FirstOrDefaultAsync(m => m.id == id);
    if (city == null)
    {
        return NotFound();
    }

    return View(city);
}

// POST: Cities/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> DeleteConfirmed(int id)
{
    var city = await _context.City.FindAsync(id);
    _context.City.Remove(city);
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}

private bool CityExists(int id)
{
    return _context.City.Any(e => e.id == id);
}
}

```

Controlador Evento

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Rendering;
using Microsoft.EntityFrameworkCore;
using EventoTec.web.Models;
using EventoTec.web.Models.entities;

namespace EventoTec.web.Controllers
{
    public class EventsController : Controller
    {
        private readonly DataDbContext _context;

        public EventsController(DataDbContext context)
        {
            _context = context;
        }

        // GET: Events
        public async Task<IActionResult> Index()
        {
            var dataDbContext = _context.Events.Include(a => a.Category).Include(a => a.City);
            return View(await dataDbContext.ToListAsync());
        }

        // GET: Events/Details/5
        public async Task<IActionResult> Details(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var @event = await _context.Events
                .Include(a => a.Category)
                .Include(a => a.City)
                .FirstOrDefaultAsync(m => m.Id == id);
            if (@event == null)
            {
                return NotFound();
            }

            return View(@event);
        }

        // GET: Events/Create
        public IActionResult Create()
        {
            ViewData["CategoryId"] = new SelectList(_context.Category, "Id", "Name");
            ViewData["CityId"] = new SelectList(_context.City, "id", "Name");
            return View();
        }

        // POST: Events/Create
        // To protect from overposting attacks, please enable the specific properties you want to bind to, for
        // more details see http://go.microsoft.com/fwlink/?LinkId=317598.
```

```

[HttpPost]
[ValidateAntiForgeryToken]
public async Task<ActionResult>
Create([Bind("Id,Name,EventDate,Descripcion,Picture,People,Duretion,CityId,CategoryId")] Event @event)
{
    if (ModelState.IsValid)
    {
        _context.Add(@event);
        await _context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    ViewData["CategoryId"] = new SelectList(_context.Category, "Id", "Name", @event.CategoryId);
    ViewData["CityId"] = new SelectList(_context.City, "id", "Name", @event.CityId);
    return View(@event);
}

// GET: Events/Edit/5
public async Task<ActionResult> Edit(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var @event = await _context.Events.FindAsync(id);
    if (@event == null)
    {
        return NotFound();
    }
    ViewData["CategoryId"] = new SelectList(_context.Category, "Id", "Description", @event.CategoryId);
    ViewData["CityId"] = new SelectList(_context.City, "id", "Descripcion", @event.CityId);
    return View(@event);
}

// POST: Events/Edit/5
// To protect from overposting attacks, please enable the specific properties you want to bind to, for
// more details see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Edit(int id,
[Bind("Id,Name,EventDate,Descripcion,Picture,People,Duretion,CityId,CategoryId")] Event @event)
{
    if (id != @event.Id)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(@event);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!EventExists(@event.Id))
            {
                return NotFound();
            }
            else
            {
                {

```



```

        throw;
    }
}
return RedirectToAction(nameof(Index));
}
ViewData["CategoryId"] = new SelectList(_context.Category, "Id", "Description", @event.CategoryId);
ViewData["CityId"] = new SelectList(_context.City, "id", "Descripcion", @event.CityId);
return View(@event);
}

// GET: Events/Delete/5
public async Task<IActionResult> Delete(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var @event = await _context.Events
        .Include(a => a.Category)
        .Include(a => a.City)
        .FirstOrDefaultAsync(m => m.Id == id);
    if (@event == null)
    {
        return NotFound();
    }

    return View(@event);
}

// POST: Events/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public async Task<IActionResult> DeleteConfirmed(int id)
{
    var @event = await _context.Events.FindAsync(id);
    _context.Events.Remove(@event);
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}

private bool EventExists(int id)
{
    return _context.Events.Any(e => e.Id == id);
}
}
}

```

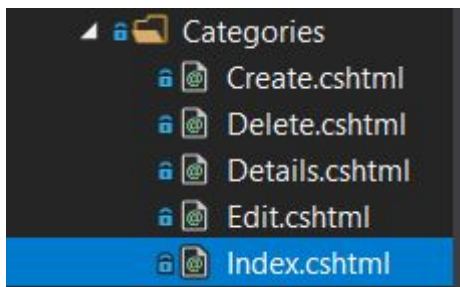
Data Context

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using EventoTec.web.Models.entities;
using Microsoft.EntityFrameworkCore;

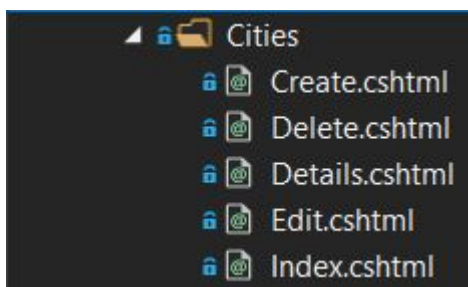
namespace EventoTec.web.Models
{
    public class DataDbContext : DbContext
    {
        public DataDbContext(DbContextOptions<DataDbContext> options) : base(options)
        {
        }

        public DbSet<Client> Clients { get; set; }
        public DbSet<City> City { get; set; }
        public DbSet<Event> Events { get; set; }
        public DbSet<EventoTec.web.Models.entities.Category> Category { get; set; }
    }
}
```

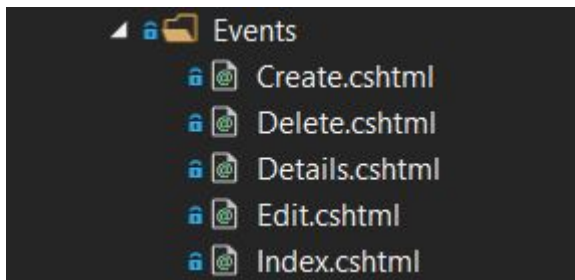
El código de las vistas de los HTML de Categorías



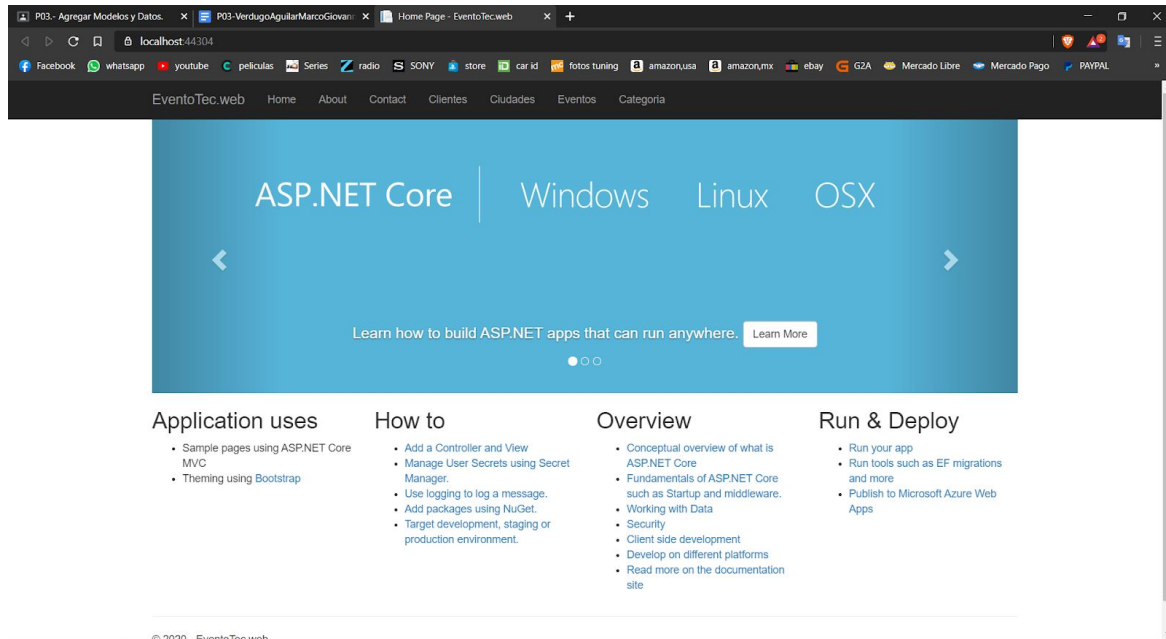
El código de las vistas de los HTML de Ciudades



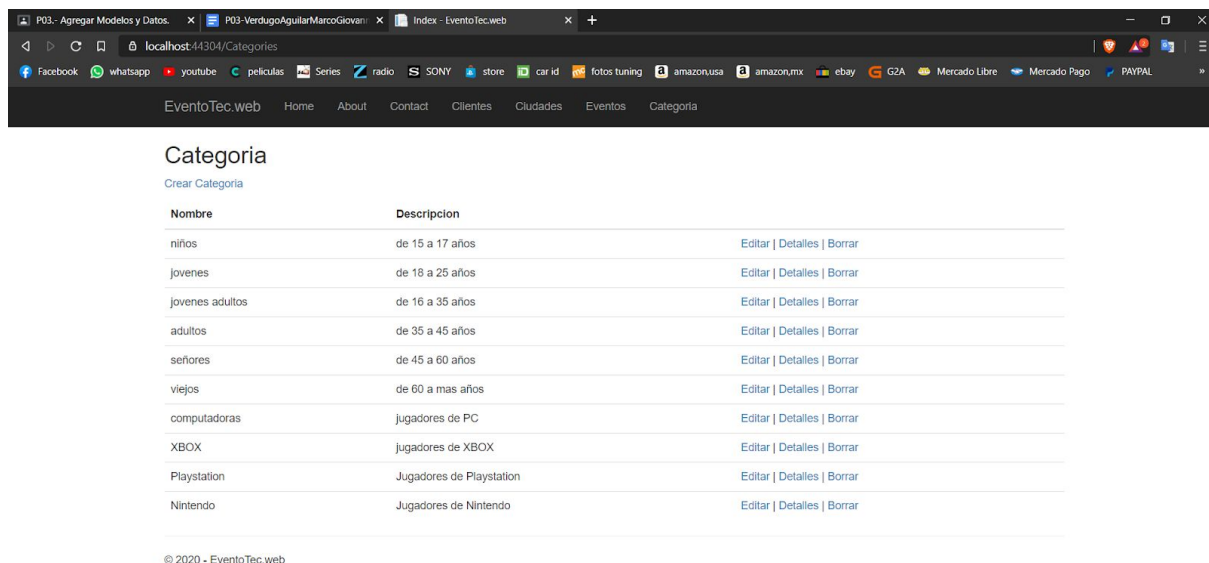
El código de las vistas de los HTML de Eventos



Diseño HTML



Categoría



Ciudades

Ciudades

[Crear Ciudad](#)

Ciudad	Descripcion	Pais	
tijuana	es raro	mexico	Editar Detalles Borrar
ensenada	pequeño	mexico	Editar Detalles Borrar
rosarito	no hay nada	mexico	Editar Detalles Borrar
mexicali	mucho calor	mexico	Editar Detalles Borrar
las vegas	casinos	estados unidos	Editar Detalles Borrar
san francisco	muy francisco	estados unidos	Editar Detalles Borrar
la paz	muchos peces	mexico	Editar Detalles Borrar
buenos aires	mi libro	argentina	Editar Detalles Borrar
mochis	comida	mexico	Editar Detalles Borrar
san ysidro	tiendas	estados unidos	Editar Detalles Borrar

© 2020 - EventoTec.web

Eventos

Eventos

[Crear Evento](#)

Nombre	Fecha del Evento	Descripcion	Foto	Asistentes	Duracion	City	Category	
fornte	30/07/2022 01:00:00 p. m.	Jugadores de fornte		10000	2	es raro	de 15 a 17 años	Editar Detalles Eliminar
fornte	30/01/2021 11:56:00 a. m.	Jugadores fornte		500	5	comida	Jugadores de Playstation	Editar Detalles Eliminar
apex	03/04/2024 04:03:00 p. m.	Jugadores de apex		15	10	tiendas	Jugadores de XBOX	Editar Detalles Eliminar
Call of duty	11/12/2032 10:10:00 a. m.	jugadores de Cod		15	8	muy francisco	jugadores de PC	Editar Detalles Eliminar
need for speed	10/11/2032 11:11:00 p. m.	carreras		11	11	muchos peces	de 45 a 60 años	Editar Detalles Eliminar
gays	11/11/2033 10:09:00 a. m.	todos lo gays		11	11	casinos	de 60 a mas años	Editar Detalles Eliminar
apex	14/12/2030 09:08:00 p. m.	Jugadores de apex		14	14	tiendas	jugadores de PC	Editar Detalles Eliminar
need for speed	12/11/2030 12:08:00 a. m.	carreras		14	14	pequeño	de 16 a 35 años	Editar Detalles Eliminar
Call of duty	10/12/2030 01:10:00 a. m.	jugadores de Cod		15	12	muy francisco	de 18 a 25 años	Editar Detalles Eliminar
Call of duty	10/10/2028 09:05:00 a. m.	jugadores de Cod		9	11	mucho calor	Jugadores de Nintendo	Editar Detalles Eliminar

© 2020 - EventoTec.web