1- Yeni MVC projesi

dotnet new mvc -n EfMaterialApp

cd EfMaterialApp

2-EF Core paketleri

dotnet add package Microsoft.EntityFrameworkCore

dotnet add package Microsoft.EntityFrameworkCore.Sqlite

dotnet add package Microsoft.EntityFrameworkCore.Tools

dotnet tool install --global dotnet-ef

3-Model (Entity) Sınıfı -- Models/Material.cs

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
namespace EfMaterialApp.Models;

public class Material
{
    public int Id { get; set; }

    [Required, StringLength(50)]
    public string Name { get; set; }

    [Range(1, 10000)]
    public int Quantity { get; set; }

    [Column(TypeName = "decimal(18,2)")]
    [Range(0.01, 999999.99)]
    public decimal UnitPrice { get; set; }

    [Required, StringLength(50)]
    public string Supplier { get; set; }

    [DataType(DataType.Date)]
    public DateTime? PurchaseDate { get; set; }
}
```

4-DbContext ve Servis Kaydı ---Data/AppDbContext.cs

```
using EfMaterialApp.Models;
using Microsoft.EntityFrameworkCore;
namespace EfMaterialApp.Data;
public class AppDbContext : DbContext
{
    public AppDbContext(DbContextOptions<AppDbContext> options)
        : base(options) { }

    public DbSet<Material> Materials => Set<Material>();
}
```

5-appsettings.json (Connection String)---appsettings.json (mevcut dosyada "ConnectionStrings" ekle)

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=materials.db"
  },
  "Logging": { "LogLevel": { "Default": "Information", "Microsoft.AspNetCore": "Warning" } },
  "AllowedHosts": "*"
}
```

6-Program.cs (EF Core servisini ekleme)

```
using EfMaterialApp.Data;
using Microsoft.EntityFrameworkCore;
var builder = WebApplication.CreateBuilder(args);
builder.Services.AddControllersWithViews();

// EF Core + SQLite
builder.Services.AddDbContext<AppDbContext>(options =>
    options.UseSqlite(builder.Configuration.GetConnectionString("DefaultConnection")));

var app = builder.Build();
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Home/Error");
    app.UseHsts();
}
app.UseHttpsRedirection();
app.UseStaticFiles();
app.UseRouting();
app.UseAuthorization();
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Material}/{action=Index}/{id?}");

app.Run();
```

7-Migration ve Veritabanı Oluşturma

```
dotnet ef migrations add InitialCreate
dotnet ef database update
```

8-Controller (CRUD Aksiyonları)-- Controllers/MaterialController.cs

```
using EfMaterialApp.Data;
using EfMaterialApp.Models;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
namespace EfMaterialApp.Controllers;

public class MaterialController : Controller
{
    private readonly AppDbContext _context;
    public MaterialController(AppDbContext context) => _context = context;

    // GET: /Material
    public async Task<IActionResult> Index()
    {
        var items = await _context.Materials
            .OrderByDescending(x => x.Id)
            .ToListAsync();
```

```
    ViewBag.Success = TempData["Success"];
    return View(items);
}

// GET: /Material/Create
[HttpGet]
public IActionResult Create()
{
    return View(new Material { PurchaseDate = DateTime.Today });
}

// POST: /Material/Create
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create(Material model)
{
    if (!ModelState.IsValid) return View(model);

    _context.Materials.Add(model);
    await _context.SaveChangesAsync();

    TempData["Success"] = $"Material '{model.Name}' created.";
    return RedirectToAction(nameof(Index));
}

// GET: /Material/Edit/5
[HttpGet]
public async Task<IActionResult> Edit(int id)
{
    var entity = await _context.Materials.FindAsync(id);
    if (entity == null) return NotFound();
    return View(entity);
}

// POST: /Material/Edit/5
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Edit(int id, Material model)
{
    if (id != model.Id) return BadRequest();
    if (!ModelState.IsValid) return View(model);

    try
    {
        _context.Update(model); // state: Modified
        await _context.SaveChangesAsync();
        TempData["Success"] = $"Material '{model.Name}' updated.";
        return RedirectToAction(nameof(Index));
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!await _context.Materials.AnyAsync(m => m.Id == id))
            return NotFound();
        throw;
    }
}
```

```
    // GET: /Material/Delete/5
    [HttpGet]
    public async Task<IActionResult> Delete(int id)
    {
        var entity = await _context.Materials.FindAsync(id);
        if (entity == null) return NotFound();
        return View(entity); // onay sayfası
    }

    // POST: /Material/Delete/5
    [HttpPost, ActionName("Delete")]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> DeleteConfirmed(int id)
    {
        var entity = await _context.Materials.FindAsync(id);
        if (entity != null)
        {
            _context.Materials.Remove(entity);
            await _context.SaveChangesAsync();
            TempData["Success"] = $"Material '{entity.Name}' deleted.";
        }
        return RedirectToAction(nameof(Index));
    }

    // GET: /Material/Details/5
    [HttpGet]
    public async Task<IActionResult> Details(int id)
    {
        var entity = await _context.Materials.FirstOrDefaultAsync(m => m.Id == id);
        if (entity == null) return NotFound();
        return View(entity);
    }
}
```

9-Views (Razor)
Views/Shared/_Layout.cshtml

```
<li class="nav-item">
  <a class="nav-link text-dark" asp-controller="Material" asp-action="Index">Materials</a>
</li>
```

Views/Material/Index.cshtml

```
@model List<Material>
@{
    ViewData["Title"] = "Materials";
    var success = ViewBag.Success as string;
}

<h1>Materials</h1>

@if (!string.IsNullOrWhiteSpace(success))
{
    <div class="alert alert-success">@success</div>
}

<p>
    <a class="btn btn-primary" asp-action="Create">Add New</a>
```

```
</p>

@if (Model == null || !Model.Any())
{
  <p>No records.</p>
}
else
{
  <table class="table table-striped table-bordered">
    <thead>
      <tr>
        <th>Name</th>
        <th class="text-end">Qty</th>
        <th class="text-end">Unit Price</th>
        <th class="text-end">Total</th>
        <th>Supplier</th>
        <th>Purchase Date</th>
        <th style="width:180px">Actions</th>
      </tr>
    </thead>
    <tbody>
    @foreach (var m in Model)
    {
      <tr>
        <td>@m.Name</td>
        <td class="text-end">@m.Quantity</td>
        <td class="text-end">@m.UnitPrice.ToString("C")</td>
        <td class="text-end">@((m.Quantity * m.UnitPrice).ToString("C"))</td>
        <td>@m.Supplier</td>
        <td>@(m.PurchaseDate?.ToString("yyyy-MM-dd") ?? "-")</td>
        <td>
          <a class="btn btn-sm btn-info" asp-action="Details" asp-route-id="@m.Id">Details</a>
          <a class="btn btn-sm btn-warning" asp-action="Edit" asp-route-id="@m.Id">Edit</a>
          <a class="btn btn-sm btn-danger" asp-action="Delete" asp-route-id="@m.Id">Delete</a>
        </td>
      </tr>
    }
    </tbody>
  </table>
}
```

Views/Material/Create.cshtml

```
@model Material
@{
  ViewData["Title"] = "Add Material";
}
<h1>Add Material</h1>

<div asp-validation-summary="ModelOnly" class="text-danger"></div>

<form asp-action="Create" method="post" class="row g-3">
  <div class="col-md-6">
    <label asp-for="Name" class="form-label"></label>
    <input asp-for="Name" class="form-control" />
    <span asp-validation-for="Name" class="text-danger"></span>
  </div>
  <div class="col-md-3">
```

```
    <label asp-for="Quantity" class="form-label"></label>
    <input asp-for="Quantity" class="form-control" />
    <span asp-validation-for="Quantity" class="text-danger"></span>
  </div>
  <div class="col-md-3">
    <label asp-for="UnitPrice" class="form-label"></label>
    <input asp-for="UnitPrice" class="form-control" />
    <span asp-validation-for="UnitPrice" class="text-danger"></span>
  </div>
  <div class="col-md-6">
    <label asp-for="Supplier" class="form-label"></label>
    <input asp-for="Supplier" class="form-control" />
    <span asp-validation-for="Supplier" class="text-danger"></span>
  </div>
  <div class="col-md-6">
    <label asp-for="PurchaseDate" class="form-label"></label>
    <input asp-for="PurchaseDate" class="form-control" />
    <span asp-validation-for="PurchaseDate" class="text-danger"></span>
  </div>
  <div class="col-12">
    <button class="btn btn-success" type="submit">Save</button>
    <a class="btn btn-secondary" asp-action="Index">Back</a>
  </div>
</form>

@section Scripts { <partial name="_ValidationScriptsPartial" /> }
```

Views/Material/Edit.cshtml

```
@model Material
@{
  ViewData["Title"] = "Edit Material";
}
<h1>Edit Material</h1>

<div asp-validation-summary="ModelOnly" class="text-danger"></div>

<form asp-action="Edit" method="post" class="row g-3">
  <input type="hidden" asp-for="Id" />
  <div class="col-md-6">
    <label asp-for="Name" class="form-label"></label>
    <input asp-for="Name" class="form-control" />
    <span asp-validation-for="Name" class="text-danger"></span>
  </div>
  <div class="col-md-3">
    <label asp-for="Quantity" class="form-label"></label>
    <input asp-for="Quantity" class="form-control" />
    <span asp-validation-for="Quantity" class="text-danger"></span>
  </div>
  <div class="col-md-3">
    <label asp-for="UnitPrice" class="form-label"></label>
    <input asp-for="UnitPrice" class="form-control" />
    <span asp-validation-for="UnitPrice" class="text-danger"></span>
  </div>
  <div class="col-md-6">
    <label asp-for="Supplier" class="form-label"></label>
    <input asp-for="Supplier" class="form-control" />
    <span asp-validation-for="Supplier" class="text-danger"></span>
```

```
  </div>
  <div class="col-md-6">
    <label asp-for="PurchaseDate" class="form-label"></label>
    <input asp-for="PurchaseDate" class="form-control" />
    <span asp-validation-for="PurchaseDate" class="text-danger"></span>
  </div>
  <div class="col-12">
    <button class="btn btn-primary" type="submit">Update</button>
    <a class="btn btn-secondary" asp-action="Index">Back</a>
  </div>
</form>

@section Scripts { <partial name="_ValidationScriptsPartial" /> }
```

Views/Material/Details.cshtml

```
@model Material
@{
  ViewData["Title"] = "Material Details";
}
<h1>Material Details</h1>
<table class="table table-bordered">
  <tr><th>Id</th><td>@Model.Id</td></tr>
  <tr><th>Name</th><td>@Model.Name</td></tr>
  <tr><th>Quantity</th><td>@Model.Quantity</td></tr>
  <tr><th>Unit Price</th><td>@Model.UnitPrice.ToString("C")</td></tr>
  <tr><th>Total</th><td>@((Model.Quantity * Model.UnitPrice).ToString("C"))</td></tr>
  <tr><th>Supplier</th><td>@Model.Supplier</td></tr>
  <tr><th>Purchase Date</th><td>@(Model.PurchaseDate?.ToString("yyyy-MM-dd") ?? "-")</td></tr>
</table>
<p>
  <a class="btn btn-secondary" asp-action="Index">Back</a>
  <a class="btn btn-warning" asp-action="Edit" asp-route-id="@Model.Id">Edit</a>
</p>
```