Here’s a **complete step-by-step guide and code structure** to help you implement the **Gym Management Web API** project using **ASP.NET Core 6 Web API with EF Core**.

**Project Title:**

**Gym Management API**

**Objective:**

Create a simple API to manage workout routines where users can:

* Add a new workout
* Mark a workout as completed
* Delete a workout

**Step-by-Step Implementation**

**Step 1: Create the Project**

dotnet new webapi -n dotnetapp

cd dotnetapp

**Step 2: Add Required Packages**

Install EF Core and SQL Server packages:

dotnet add package Microsoft.EntityFrameworkCore --version 6.0

dotnet add package Microsoft.EntityFrameworkCore.SqlServer --version 6.0

dotnet add package Microsoft.EntityFrameworkCore.Tools --version 6.0

**Step 3: Create the Model**

Create a folder named **Models** and add a file **Workout.cs**:

namespace dotnetapp.Models

{

public class Workout

{

public int WorkoutId { get; set; }

public string Name { get; set; }

public string Type { get; set; }

public int Duration { get; set; } // in minutes

public bool IsCompleted { get; set; } = false;

}

}

**Step 4: Create the DbContext**

Create a folder named **Data** and add a file **AppDbContext.cs**:

using Microsoft.EntityFrameworkCore;

using dotnetapp.Models;

namespace dotnetapp.Data

{

public class AppDbContext : DbContext

{

public AppDbContext(DbContextOptions<AppDbContext> options) : base(options)

{

}

public DbSet<Workout> Workouts { get; set; }

}

}

**Step 5: Configure Database Connection**

Open **appsettings.json** and add the connection string:

{

"ConnectionStrings": {

"DefaultConnection": "User ID=sa;password=examlyMssql@123;server=localhost;Database=appdb;trusted\_connection=false;Persist Security Info=False;Encrypt=False"

},

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning"

}

},

"AllowedHosts": "\*"

}

**Step 6: Register DbContext in Program.cs**

Edit **Program.cs** to include EF Core setup:

using dotnetapp.Data;

using Microsoft.EntityFrameworkCore;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

builder.Services.AddDbContext<AppDbContext>(options =>

options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

var app = builder.Build();

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseAuthorization();

app.MapControllers();

app.Run();

**Step 7: Create the Controller**

Create a folder named **Controllers** and add **WorkoutController.cs**:

using Microsoft.AspNetCore.Mvc;

using dotnetapp.Data;

using dotnetapp.Models;

using Microsoft.EntityFrameworkCore;

namespace dotnetapp.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class WorkoutController : ControllerBase

{

private readonly AppDbContext \_context;

public WorkoutController(AppDbContext context)

{

\_context = context;

}

// GET: api/Workout

[HttpGet]

public async Task<ActionResult<IEnumerable<Workout>>> GetWorkouts()

{

return await \_context.Workouts.ToListAsync();

}

// POST: api/Workout

[HttpPost]

public async Task<ActionResult<Workout>> AddWorkout(Workout workout)

{

\_context.Workouts.Add(workout);

await \_context.SaveChangesAsync();

return CreatedAtAction(nameof(GetWorkouts), new { id = workout.WorkoutId }, workout);

}

// PUT: api/Workout/{id}/complete

[HttpPut("{id}/complete")]

public async Task<IActionResult> MarkAsCompleted(int id)

{

var workout = await \_context.Workouts.FindAsync(id);

if (workout == null)

return NotFound();

workout.IsCompleted = true;

await \_context.SaveChangesAsync();

return NoContent();

}

// DELETE: api/Workout/{id}

[HttpDelete("{id}")]

public async Task<IActionResult> DeleteWorkout(int id)

{

var workout = await \_context.Workouts.FindAsync(id);

if (workout == null)

return NotFound();

\_context.Workouts.Remove(workout);

await \_context.SaveChangesAsync();

return NoContent();

}

}

}

**Step 8: Run EF Core Migrations**

Initialize EF and create the database:

dotnet new tool-manifest

dotnet tool install --local dotnet-ef --version 6.0.6

dotnet ef migrations add "InitialSetup"

dotnet ef database update

**Step 9: Set Port to 8080**

In **Properties/launchSettings.json**, update the URL:

"applicationUrl": "http://0.0.0.0:8080"

**Step 10: Run the Application**

dotnet restore

dotnet build

dotnet run

Then open:  
[**http://localhost:8080/swagger**](http://localhost:8080/swagger)

**Sample API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| GET | /api/workout | Get all workouts |
| POST | /api/workout | Add a new workout |
| PUT | /api/workout/{id}/complete | Mark a workout as completed |
| DELETE | /api/workout/{id} | Delete a workout |

**Sample JSON Payload**

**Add Workout:**

{

"name": "Morning Cardio",

"type": "Cardio",

"duration": 30

}

Would you like me to include **NUnit test cases** for this API as well (to test Add, Get, Complete, and Delete operations)?

Top of Form

Bottom of Form