



Facultad de Ingeniería
Escuela de Computación

Desarrollo de Software Empresarial

Guía 7: Desarrollo de habilidades

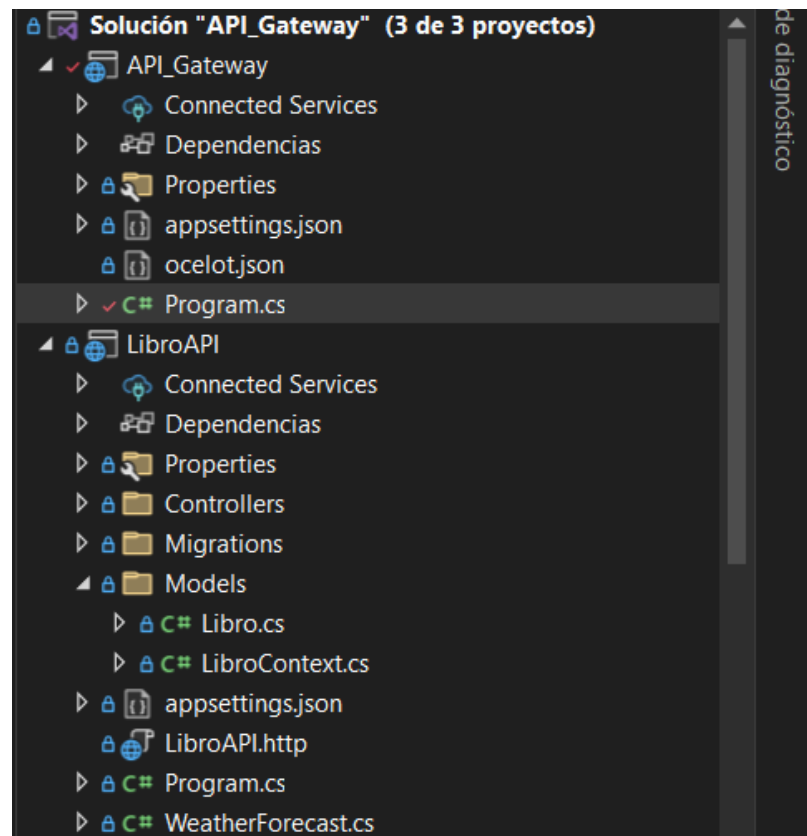
PRESENTADO POR:

Nombre	Código
Eduardo Josué Deras Mancia	DM201736

Catedrático: Mg. Emerson Cartagena

Desarrollo de habilidades

1. Estructura del proyecto



2. Modificando el program.cs del Api Gateway

```
using LibroAPI.Models;
using Microsoft.EntityFrameworkCore;
using Ocelot.DependencyInjection;
using Ocelot.Middleware;

namespace API_Gateway
{
    1 referencia
    public class Program
    {
        0 referencias
        public static async Task Main(string[] args) // Cambiar a Task y agregar async
        {
            var builder = WebApplication.CreateBuilder(args);

            builder.Configuration.AddJsonFile("ocelot.json", optional: false, reloadOnChange: true);
            builder.Services.AddOcelot();

            builder.Services.AddDbContext<LibroContext>(options =>
                options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

            builder.Services.AddControllers();

            builder.Logging.AddConsole();
            builder.Logging.AddDebug();

            var app = builder.Build();

            // autenticación para API Key
            app.Use(async (context, next) =>
            {
                const string HeaderName = "Auth";
                const string apiKey = "terrestre";

                if (context.Request.Headers.TryGetValue(HeaderName, out var extractedApiKey))
                {
                    if (extractedApiKey == apiKey)
                    {
                        await next();
                        return;
                    }
                }

                context.Response.ContentType = "text/plain";
                context.Response.StatusCode = StatusCodes.Status401Unauthorized;
                await context.Response.WriteAsync("La clave API es incorrecta");
                return;

                context.Response.ContentType = "text/plain";
                context.Response.StatusCode = StatusCodes.Status401Unauthorized;
                await context.Response.WriteAsync("No autenticado");
            });

            app.UseRouting();
            app.UseAuthorization();

            await app.UseOcelot(); // Ahora await es válido dentro del método async

            var logger = app.Services.GetRequiredService<ILogger<Program>>();
            logger.LogInformation("Ocelot on port {Port}", builder.Configuration["GlobalConfiguration:BaseUrl"]?.Split(':').Last());

            app.Run();
        }
    }
}
```

3. Validando las configuraciones atraves de thunderclient

The screenshot displays the Thunder Client interface. The top bar shows the method **GET** and the URL **https://localhost:7193/libros**, with a **Send** button. Below this, the **Headers** tab is active, showing a table with two headers:

Header	Value
<input checked="" type="checkbox"/> Auth	terrestre
<input type="checkbox"/> header	value

The right panel shows the **Response** tab with the following JSON data:

```
[
  {
    "id": 1,
    "titulo": "Clean Code",
    "autor": "Robert C. Martin",
    "anioPublicacion": 2008
  },
  {
    "id": 2,
    "titulo": "The Pragmatic Programmer",
    "autor": "Andrew Hunt, David Thomas",
    "anioPublicacion": 1999
  },
  {
    "id": 3,
    "titulo": "Design Patterns: Elements of Reusable
      Object-Oriented Software",
    "autor": "Erich Gamma, Richard Helm, Ralph Johnson,
      John Vlissides",
    "anioPublicacion": 1994
  }
]
```

At the bottom right of the response panel, there are buttons for **Response** and **Chart**.