Step by Step Implementation of Ocelot API Gateway in .NET 8

1) Create three projects

---

dotnet new web -n Gateway --framework net8.0
dotnet new webapi -n Products.Api --framework net8.0
dotnet new webapi -n Orders.Api --framework net8.0

Assign ports for Products.Api (7001) and Orders.Api (7002).

2) Add Ocelot to Gateway

---

dotnet add package Ocelot
dotnet add package Ocelot.Provider.Polly
dotnet add package MMLib.SwaggerForOcelot

3) Gateway configuration (ocelot.json)

---

```
{
"Routes": [
{
"DownstreamPathTemplate": "/api/products",
"DownstreamScheme": "http",
"DownstreamHostAndPorts": [{ "Host": "localhost", "Port": 7001 }],
"UpstreamPathTemplate": "/products",
"UpstreamHttpMethod": [ "GET", "POST" ],
"Key": "products-all"
},
{
"DownstreamPathTemplate": "/api/products/{id}",
"DownstreamScheme": "http",
"DownstreamHostAndPorts": [{ "Host": "localhost", "Port": 7001 }],
"UpstreamPathTemplate": "/products/{id}",
"UpstreamHttpMethod": [ "GET", "PUT", "DELETE" ],
"Key": "products-by-id"
},
{
"DownstreamPathTemplate": "/api/orders/{customerId}",
"DownstreamScheme": "http",
"DownstreamHostAndPorts": [{ "Host": "localhost", "Port": 7002 }],
"UpstreamPathTemplate": "/orders/{customerId}",
"UpstreamHttpMethod": [ "GET" ],
"Key": "orders-by-customer"
}
],
```

"GlobalConfiguration": {
"BaseUrl": "http://localhost:7000",
"RequestIdKey": "Oc-Request-Id"
}
}

4) Program.cs (Gateway)

---

using Ocelot.DependencyInjection;
using Ocelot.Middleware;

var builder = WebApplication.CreateBuilder(args);
builder.Configuration.AddJsonFile("ocelot.json", optional: false, reloadOnChange: true);
builder.Services.AddOcelot(builder.Configuration);
builder.Services.AddCors(p => p.AddDefaultPolicy(b =>
b.AllowAnyOrigin().AllowAnyHeader().AllowAnyMethod()));
builder.Services.AddSwaggerForOcelot(builder.Configuration);

var app = builder.Build();
app.UseCors();
app.UseSwaggerForOcelotUI(opt => { opt.PathToSwaggerGenerator = "/swagger/docs"; });
await app.UseOcelot();
app.Run("http://localhost:7000");

5) Controllers (Downstream APIs)

---

ProductsController with GET /api/products and GET /api/products/{id}
OrdersController with GET /api/orders/{customerId}

6) Optional Features

---

a) Load Balancing with RoundRobin
b) Rate Limiting with "RateLimitOptions"
c) JWT Authentication at Gateway
d) Aggregation using "Aggregates" section
e) Swagger Consolidation with SwaggerForOcelot

7) Run and Test

---

dotnet run --urls http://localhost:7001 (Products)
dotnet run --urls http://localhost:7002 (Orders)
dotnet run --urls http://localhost:7000 (Gateway)

curl http://localhost:7000/products
curl http://localhost:7000/products/1
curl http://localhost:7000/orders/alice
curl http://localhost:7000/dashboard/alice/1

8) Common Issues

---

- 404: check UpstreamPathTemplate, UpstreamHttpMethod, and await app.UseOcelot().
- CORS: configure in Gateway.
- Swagger empty: check SwaggerKey matches SwaggerEndPoints keys.