API Aggregation service

# Overview

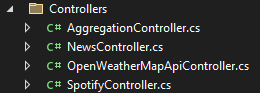
This API Aggregation Service is built using **ASP.NET Core** to provide an interface that consolidates data from multiple external APIs. The service integrates with three external APIs, **OpenWeatherMap**, **Spotify**, and **News API**, allowing clients to retrieve weather data, song information, and news articles through a single endpoint and also the option to aggregate the data in one request.

### APIs:

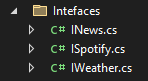
* **OpenWeatherMap API :** Provide weather data from a city.
* **Spotify API:** Provide song details like artist , duration and Spotify link
* **News API:** Provides articles based on a topic.

# Project Architecture

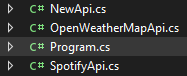
* **Controllers:** The controllers handle the incoming request and provide us the response of each API
  + **OpenWeatherApiController (**For Weather data**)**
  + **SpotifyApiController (**For song data**)**
  + **NewsApiController (**For news data**)**
  + **AggregationController(**For all together**)**



* **Interfaces :** 
  + **Iweather**
  + **ISpotify**
  + **INews**



* **Services:** The services Interact with the external APIs
  + **OpenWeatherMapApi** (implements **IWeather**)
  + **SpotifyApi** (implements **ISpotify**)
  + **NewsApi** (implements **INews**)



# Flow

1. **Client sends a request** to the API aggregation service.
2. **Controller processes the request** and passes it to the appropriate service based on the request (e.g., weather, news, or Spotify).
3. **Service makes a call** to an external API (like OpenWeatherMap, News API, or Spotify) to get the required data.
4. **External API responds** with the requested data or an error message.
5. **Service processes the response** and formats it, checking if the data is valid or if there's an error.
6. **Controller sends the response back** to the client (success or error).

The client receives either the requested data (like weather info, news articles, or song details) or an error message.

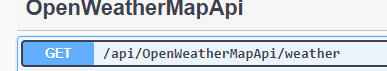
# API Endpoints

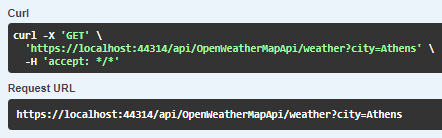
* **OpenWeatherMap API**

**Endpoint:** /api/OpenWeatherMapApi/weather

**HTTP Method:** Get

**Description:** Retrieves weather data for a city(parameter)





**Response(JSON):**

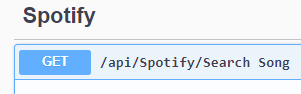


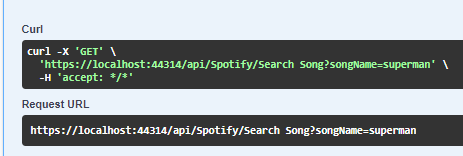
* **Spotify API**

**Endpoint:** /api/Spotify/Search Song

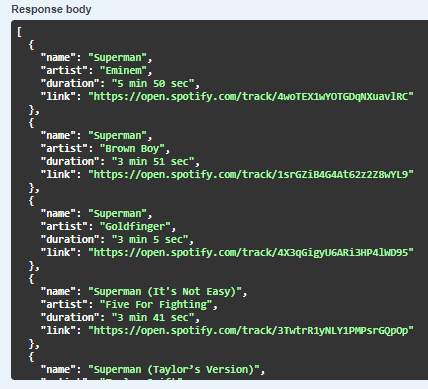
**HTTP Method:** Get

**Description:** Retrieves 5 song data with the name of the song(parameter)





**Response(JSON):**

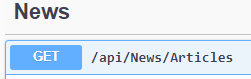


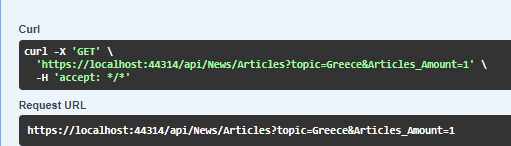
* **News API**

**Endpoint:** /api/News/Articles

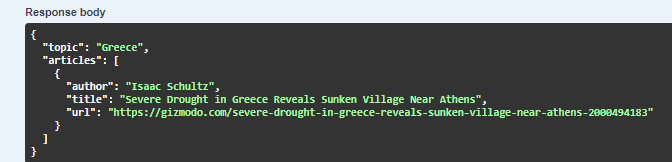
**HTTP Method:** Get

**Description:** Retrieves Atricles on a topic(parameter) and amount of articles(parameter)





**Response(JSON):**



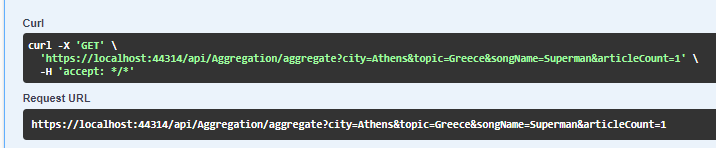
* **News API**

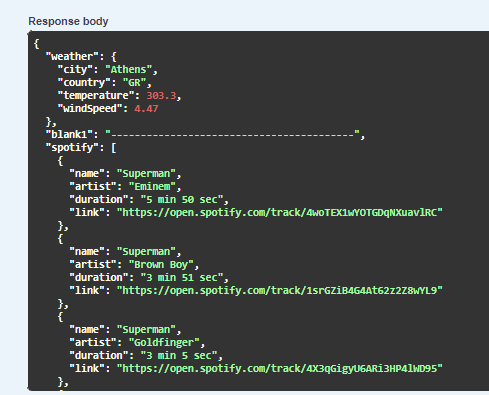
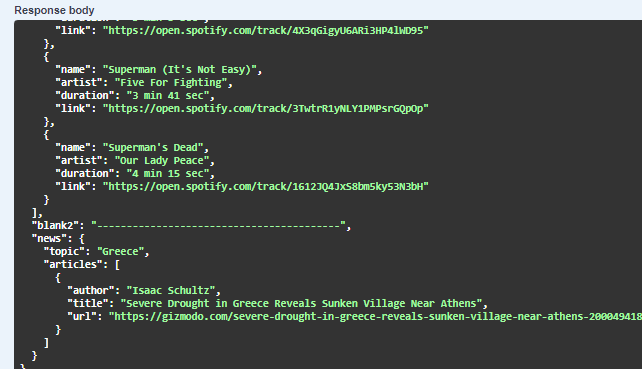
**Endpoint:** /api/Aggregation/aggregate

**HTTP Method:** Get

**Description:** Retrieves all the above together





**Response(JSON):**  

# Input/Output Formats

* **Input**
  + **Query Parameters:** All the API endpoints expect query parameters such as city, songName, and topic.
  + **Content Type:** application/json is used for the response.
* **Output**
  + **Format**: All responses are returned in JSON format.(Like above topic)
  + **Error Handling**: If the external API fails or the request is invalid or in a case of an error, a structured error message is returned in the following format for example:



# Setup and Configuration

* **Prerequisites**
  + **.NET 8 SDK**: Ensure you have the latest version of the .NET 8 SDK installed on your machine.
  + **NuGet Packages**:
    - **Newtonsoft.Json:** For handling JSON serialization and deserialization.
    - **xUnit:** For unit testing.
* **API Keys**

We will need to get our API Keys from:

* + **OpenWeatherMap API**: Get the key from OpenWeatherMap.
  + **Spotify API:** Get the client ID and secret from the [Spotify Developer Dashboard](https://developer.spotify.com/dashboard/applications).
  + **News API:** Sign up for a key at [News API](https://newsapi.org/).
* **NOTE:** You will need to add your key in the code (NewApi , SpotifyApi , etc)

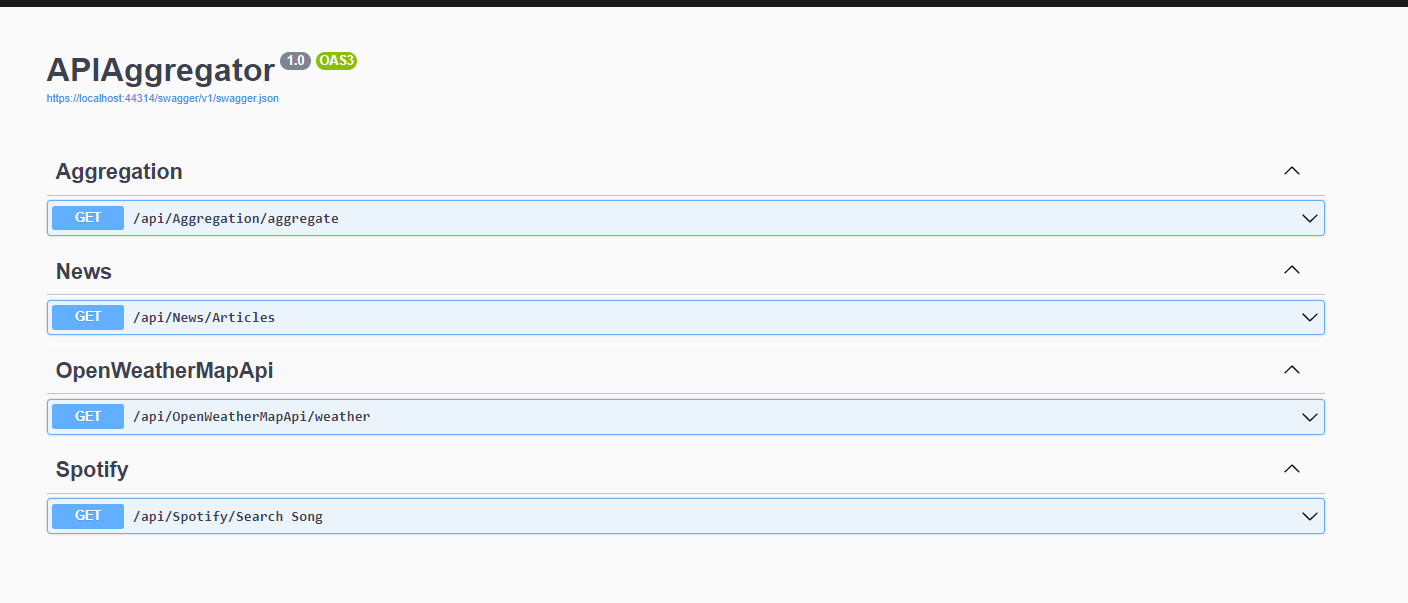


* **Run the Project**

To run the project you click on Run in IIS Express

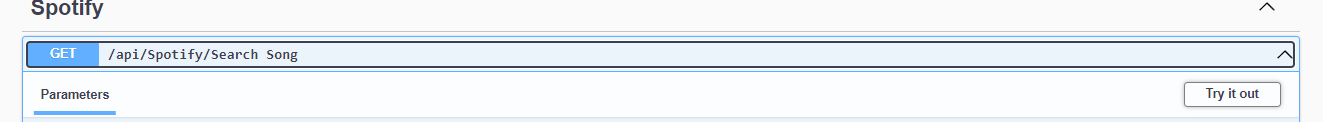


And it opens this Swagger UI

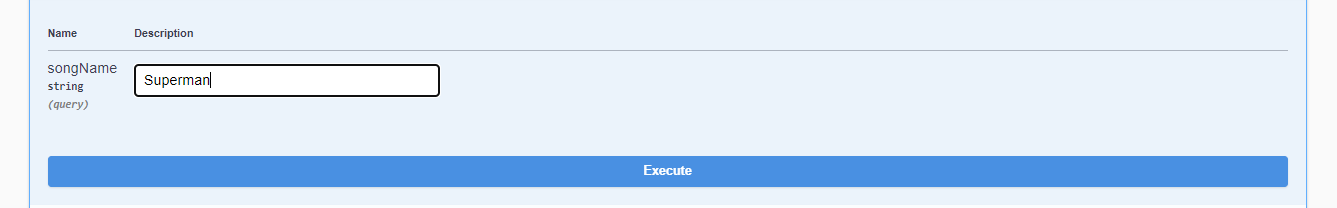


After the page loads you are ready to start fetching data from each API , for example Spotify:

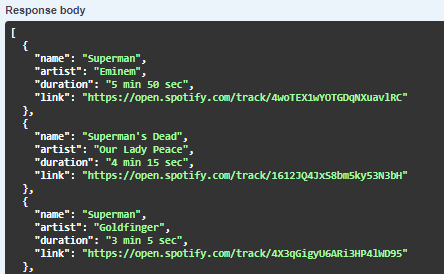
* You click **Try it out**



* You write the name of the song and click the **execute** Button



* And you have the results



# Conclusion

This project demonstrates how to build an API aggregation service in .NET, integrating multiple external APIs with error handling and unit testing. It can be extended to include additional services and improvements.