Movie API

You are required to create a simple JSON API for a Movie Studio.
The list of endpoints you have to implement is given below:

Rules

- You should solve this problem using .NET Core 3.1, please create a .NET Core Web API.
- You should provide any required instructions to get the API up and running.
- You should try and spend no more than 1 hour on this challenge.
- Please upload the solution to a public GitHub repo. (They are free!)
- You have been provided with a data file containing all the metadata for some Sony movies, use this to test your API.
- You do not need to hook your solution up to a database. Reading directly from the supplied csv documents is sufficient. Results “saved to a database” can just be added to a list called database.
- **Your request and response formats MUST match the examples below.**
- **You do not need to create anything other than the three defined endpoints. You do not get bonus points for providing anything other than what we have asked.**

We run a set of automated tests against your solution, failing to follow the rules above will result in a failure.

The three endpoints are:

POST /metadata
GET /metadata/:movieId
GET /movies/stats

POST /metadata

Saves a new piece of metadata.. The body contains the following information:

1. Movie Id - integer representing unique movie identifier
2. Title - string representing the title of the Movie
3. Language - string representing the translation of the metadata
4. Duration - string representing the length of the movie
5. Release Year - integer representing the year the movie was released

For this example, you can assume no validation is necessary.
I.e. The client will only ever POST valid content.

Example:

```
{
  "movieId": 3,
  "title": "Elysium",
  "language": "EN",
  "duration": "1:49:00",
  "releaseYear": 2013
}
```

GET /metadata/:movieId

Returns all metadata for a given movie.

- Only the latest piece of metadata (highest Id) should be returned where there are multiple metadata records for a given language.
- Only metadata with all data fields present should be returned, otherwise it should be considered invalid.
- If no metadata has been POSTed for a specified movie, a 404 should be returned.
- Results are ordered alphabetically by language.

Example:


```
[
  {
    "movieId": 3,
    "title": "Elysium",
    "language": "EN",
    "duration": "1:49:00",
    "releaseYear": 2013
  },
  {
    "movieId": 3,
    "title": "Kỷ Nguyên Elysium",
    "language": "VN",
    "duration": "1:49:00",
    "releaseYear": 2013
  }
]
```

GET /movies/stats

Returns the viewing statistics for all movies.

- The movies are ordered by most watched, then by release year with newer releases being considered more important.
- The data returned only needs to contain information from the supplied csv documents and does not need to return data provided by the POST metadata endpoint.

Example:


```
[
  {
    "movieId": 3,
    "title": "Elysium",
    "averageWatchDurationS": 3600,
    "watches": 4000,
    "releaseYear": 2013
  },
  {
    "movieId": 6,
    "title": "Total Recall",
    "averageWatchDurationS": 5400,
    "watches": 3000,
    "releaseYear": 2012
  }
]
```