AspNetCore Fundamentals:
Step1: Create a new WebApi Project named "ServiceA".
Create an api method in ServiceA, that returns the MapSettings defined in ServiceB.
Make an **http request to ServiceB** to call MapSettings api of ServiceB. Set the "X-API-KEY" header to 123 to be able to call ServiceB apis.

Step2: Create a new WebApi Project named "ServiceB".
For all requests, create a **middleware** in this service that returns a 400 BadRequest response saying "API key is not valid." if the request header does not have a header **"X-API-KEY"** with value 123.
If the request has the header with the correct value(123) continue the middleware pipeline.
Read: https://docs.microsoft.com/en-us/aspnet/core/fundamentals/middleware/write?view=aspnetcore-5.0

Step3: Add the following setting in appSettings.json of Project ServiceB
```
"MapSettings": {
  "Provider":"OpenStreetMap",
  "ZoomLevel": 6,
  "Center": {
    "Latitude": 39.123,
    "Longitude": 32.987
  }
}
```
Create an api method in ServiceB, that returns this MapSettings. When this setting is changed in file, the api should return the latest values, without the need to restart the service.
Remember that this method is only accessible if the request has correct api key.
Read: https://docs.microsoft.com/en-us/aspnet/core/fundamentals/configuration/options?view=aspnetcore-5.0

Step4: Instead of a middleware to check the X-API-KEY header for all requests to ServiceB, implement an **ActionFilter**, and apply the attribute only to the GetMapSettings api method.
Comment out the middleware addition(in Startup.cs Configure method), and see how actionfilter works.
Read: https://docs.microsoft.com/en-us/aspnet/core/mvc/controllers/filters?view=aspnetcore-5.0