GlobalWeather RESTful API -
Design Decision, Challenges and Issues

Design Decision

The task is to implement a RESTFul API which has to fill the gap between an existing legacy SOAP API and external applications.

Firstly, I had to understand the problem well and analyse all parts which will be involved in the project:

1. To understand how the GlobalWeather SOAP API works
2. To determine what data or operations the API will expose / consume
3. To determine how these elements relate to each other, what parameters they might allow and how results are returned
4. To understand what modules, what kind of relationships between the modules, functionality of each module
5. Design classes, interfaces of each module that communicate with each other

Then, I did research on the development frameworks, to find out which one would be more suitable for the development of this project:.

- MuleSoft,
- Java/Spring Boot/WebFlux/Miconouts or
- .Net Core.

Challenges

The project was challenging but having a development background helped me overcome them. I have professional experience with building RESTful APIs and SOAP web services.

The main challenge for me was learning the Spring Boot framework, WebFlux and Swagger as I did not have any experience with all of them. It was very interesting for me to see how

these technologies worked and I was excited to have some hands-on experience with all of them.
Good source for learning was Tutorials and Github, as there are many practical examples:

- https://github.com/eugenp/tutorials/tree/master/spring-boot-modules
- https://github.com/eugenp/tutorials/tree/master/spring-5-reactive

Issues

I had many issues regarding dependencies incompatibilities (maven), probably gradle would have been a better choice to use for my first Spring Boot project.

I did have some hiccups here and there regarding SOAP connections, Swagger Settings, Unit testing and other small issues. Stackoverflow and Google mainly helped me with most of them.