RestAPI

* Problems:
  + Get XML
    - While fetching the XML, the provided methods did not work properly – every one of the ended with a CORS error for no reason.
* Problem Solution:
  + To solve this problem, the provided methods weren’t used anymore. Instead of them, a normal HTTP client was used. With the usage of the HTTP client, the CORS errors were resolved. At the end, it took quite a lot of time to investigate and resolve this problem

XML-Reader

* Problems
  + Reading fetched XML causing exception
    - After the XML File has been downloaded, it was not possible to use the XMLReader class to iterate through the XML File. The content couldn’t be read with the reader and therefore causing exceptions.
* Problem Solution:
  + After multiple attempts to solve this problem, I switched to another alternative “Syndication”. The Syndication class worked immediately without any further code changes. It would be also possible to create a Parser for that but at the end, this is not the goal of this thesis statement.

MudBlazor

* Problems:
  + IMG Component is not working
    - MudBlazor is the most used Material Design Plugin for Blazor with tons of Components. But at the end, not every component is working properly. The MudImage Component is documented from MudBlazor but it can’t be called as it does not exist. Therefore, it might cause some frustrating moments.
* Problem Solution:
  + Used the default img tag of HTML

Dependency Injection

* Problems:
  + Couldn’t Inject value with the Tag [Inject]
    - The Tag [inject] was suggested by the community to inject a service but at the end this caused null pointer exceptions.
* Problem Solution:
  + It is not possible to use [inject] within .cs files therefore used Automatic Injection via constructor.

What was positive:

* + Authorization
    - It is simple to setup authorization on a page even if it was just complex to setup the login. There is no plugin available for that and it is a little bit complex as blazor wasm is running on client side.
  + Component
    - Creation of a Component and passing data to it is really easy.

What was negative:

* + Debugger
    - The debugger does not get fired immediately causing confusion as the task is already finished but the bullet point wasn’t hit. It takes a while until the Debugger is ready. Therefore, you must wait at least 10 seconds to use the debugger otherwise it will just skip the debug points as the debugger is not ready at that time.
  + CORS
    - Blazor WASM especially in this example caused a lot of CORS errors which are really frustrating as there is nothing wrong with the code and gets suggested from the community.
  + Compiler
    - Found out, that the compiler is a little bit to slow as the changes from the static files and for example in program.cs and imports.razor are not getting detected immediately. After building the project the changes get recognized.