Subsystem interfaces

General part

Splitting software systems into subsystems with well-defined interfaces is one of the most effective and widely used techniques for reducing complexity and improving maintainability. During your study, you have seen many examples of this, both intra-language and inter-language.

- Briefly explain the idea of having subsystems behind interfaces.
- Explain some of the techniques you have seen for splitting software into subsystems.
- Explain how you can build subsystems in different programming languages and on different platforms and have them work together using a REST API combined with JSON or XML for DTO's.

Practical part

1) Design a simple REST API that can manage a "TODO" list. It must be possible to:

- add items to the list
- remove items from the list
- get all the items on the list.

You must use either JSON or XML for the Data Transfer Objects (DTOs).

2) Use Node.js and express to create a simple implementation of the REST api from 1). For this exercise, you do not need to use a database; it is okay to just store the list in memory (of course it will be lost when the server is restarted).

3) Use java to create a simple implementation of the REST API from 1). Again, it is ok to just store the list in memory, you do not need to spend time creating a database.

4) Create a simple client that uses the API from 1). You may use any technology you like for this. You must be able to add items, remove items and list all items. Just focus on this simple functionality, only focus on styling if you have time.

5) Verify that your client works with both your implementations.