Truck CRUD Challenge

Business Requirements

For our platform we want to provide an application to maintain truck data for MAN. For this particular exercise we assume to have the following products:

- The TGX 18.440 which is a heavy range truck with a 440 HP engine power, 10.518 cm³ engine consuming diesel fuel. This truck is available in the colors red, white and blue. There are different configurations of this truck available for the following application segments: "Long haul", "Construction", "Firedepartment" (For a better imagination have a look at https://www.truck.man.eu/de/en/trucks/tgx/overview/tgx.html, but for your design & implementation only consider the options written here).
- The TGS 18.320 which is a light range truck with a 320 HP engine power, 10.518 cm³ engine consuming diesel fuel. This truck is available in the colors white and blue. Here are different configurations of this truck available for the following application segments: "Distribution (Food)", "Long haul", "Wastedisposal" (For a better imagination have a look at https://www.truck.man.eu/de/en/trucks/tgs/tgs.html, but for your design & implementation only consider the options written here)

Technical Requirements

- Define a Rest API with the functionalities:
  - List all Trucks
  - Create one Truck
  - Read one Truck
  - Update one Truck
  - Delete one Truck
- The definition should be done with OpenAPI 3 (use https://editor.swagger.io/ for this).
- In case you pass information in the request body, please define also the JSON schema definition in the specification for this request body.
- Generate with the maven plugin java classes from the OpenAPI specification.
  - Generate for the Angular frontend the type script classes and files.
  - Generate for the server side the java classes.
- Implement a simple Angular application with Angular with at least version 7.X.
  - The application should have different components for be able to list and edit trucks.
  - Use the generated typescript files to communicate with the server.
  - Use Angular CLI to generate the different kind of artifacts.
  - Write some unit tests.
- Implement a simple Spring Boot (Version >= 2.0) application for the Rest APIs.
  - Use the generated java classes to provide data.
  - Store the truck objects in a HSQLDB or H2 DB.
  - Write some unit tests.
- Use Apache Maven as build tool.
  - Use multi module setup.

Demonstration

- Explain why you designed the API the way you did and what assumptions did you take.
- Explain why you have implemented the spring application as you did.
    - What kind of layers did you use?
    - Which approach has been used to store the data?
- Explain the Angular components and how you did the integration with the OpenAPI files.
- Explain if you did use a version control system like Git to save intermediate states.

Result format

- Please provide the code in a zip/tar file or in any public Git repository.
- Descriptions and explanations should be delivered in short bulleted lists.