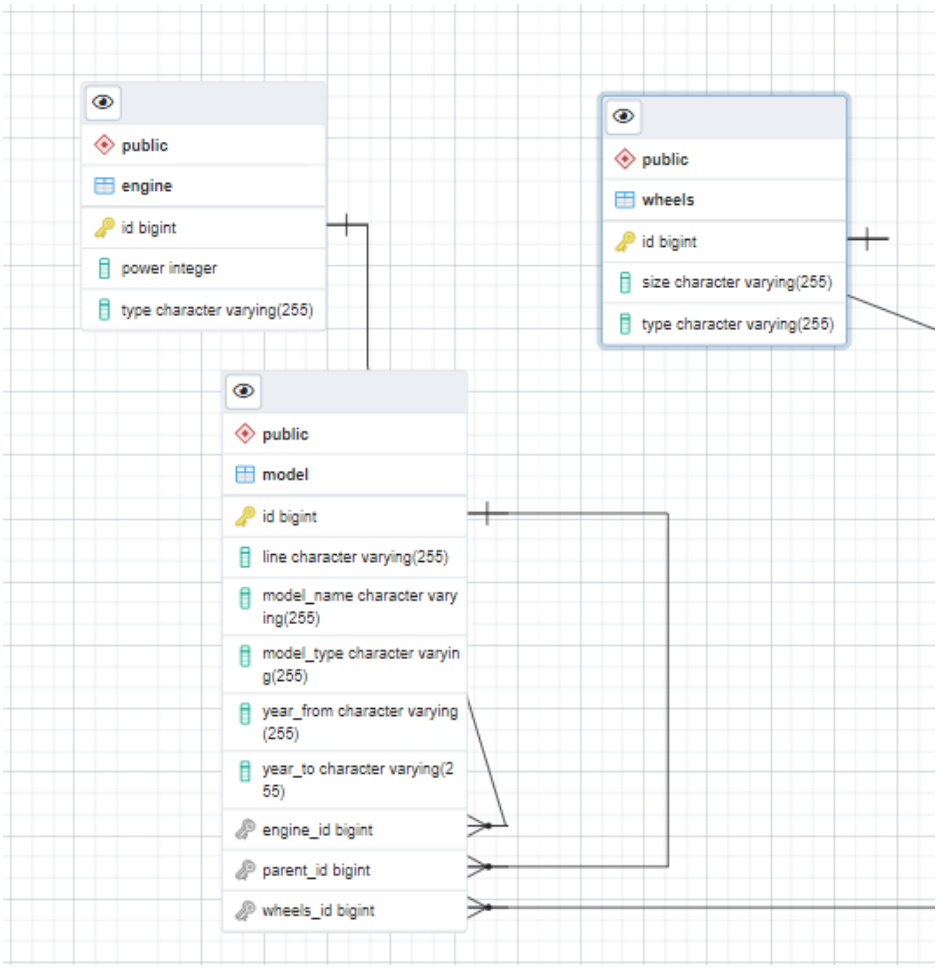
Technincal Document CarModel

1. Entity



In the UML diagram there are presented three entities where the components of the ford.xml file will be written. There is created a base abstract class with only one attribute “Long id” that all the subclasses should inherit. The “model” entity will contain the characteristics of the cars. Since the models and submodels have very characteristics in common the “model” entity have a @OneToMany relationship with itself to map models with submodels. Different car models could have the same engine so the “engine” entity is created with two attributes type and power and has a @ManyToOne relationship with the “model” entity. As well the wheels components could be the same for different models so the “wheels” entity has two attributes type and size and to have a @ManyToOne relationship with the “model” entity. **Model**(**PK(Long id)**,model_name, model_type, model_line, year_from, year_to, **FK(parent_id)**, **Fk(engine_id),** **FK(wheels_id)**) **Engine** **(PK(Long id)**, power, type) **Wheels(PK(Long id),** size, type)

2. Entering data

I have used DOM parser to read data from the XML file. The Document Object Model (DOM) uses nodes to represent the HTML or XML document as a tree structure. First I have taken the list with the engine and wheels component. Parsed the attributes inside the components in WheelsDto and EngineDto and after that into the entity. For saving the Lists with EngineEntity and WheelsEntity it is used saveAll method from the repositories that extend the JpaRepository. After inserting these two components it is taken the node list with the model component. There is made a check to separate models and submodels from the type attribute. After converting ModelDto to Entity there is made a call to get all the wheels and engine entitys from the database and associate them with their corresponding model. Using Cascade.ALL to insert all the models and submodels at the same time and associating them with their parent_id.

3. Exposing Data with RESTful API

Creating a CarModelController and annotating the class with @RestController annotation in order to create RESTful web services. Data is exposed through three methods that have the @GetMapping annotation to show that this is a get request with the defined path for the request. There are three methods for exposing data. 1. to get all the car models in a response dto from the database. 2. to get the car by the model name. 3. to get the car model by their id. All this methods call the ModelService interface which is implemented in the Service implementetion class where goes the business logic of this application. In the service implementation layer there are made calls from the repositories and handled the errors that could be thrown the custom Not Found exception or to map the returned entities to Response Dto. In the repositories interface which are overriden the findByModelName(String modelName) method where jpa creates the select query with the condition of the model_name= modelName.

3.Inserting image

Inseritng images could be done through adding the image url to a xml component so the image could be entered in the databse as a url String and then be returned the same to the Response so the frontend could render it as an image link.

4. Improvements

There could be improvements in reading data from the XML file because the DOM parser could create latency problems in reading from a bigger file. And making a use more of the generic types.