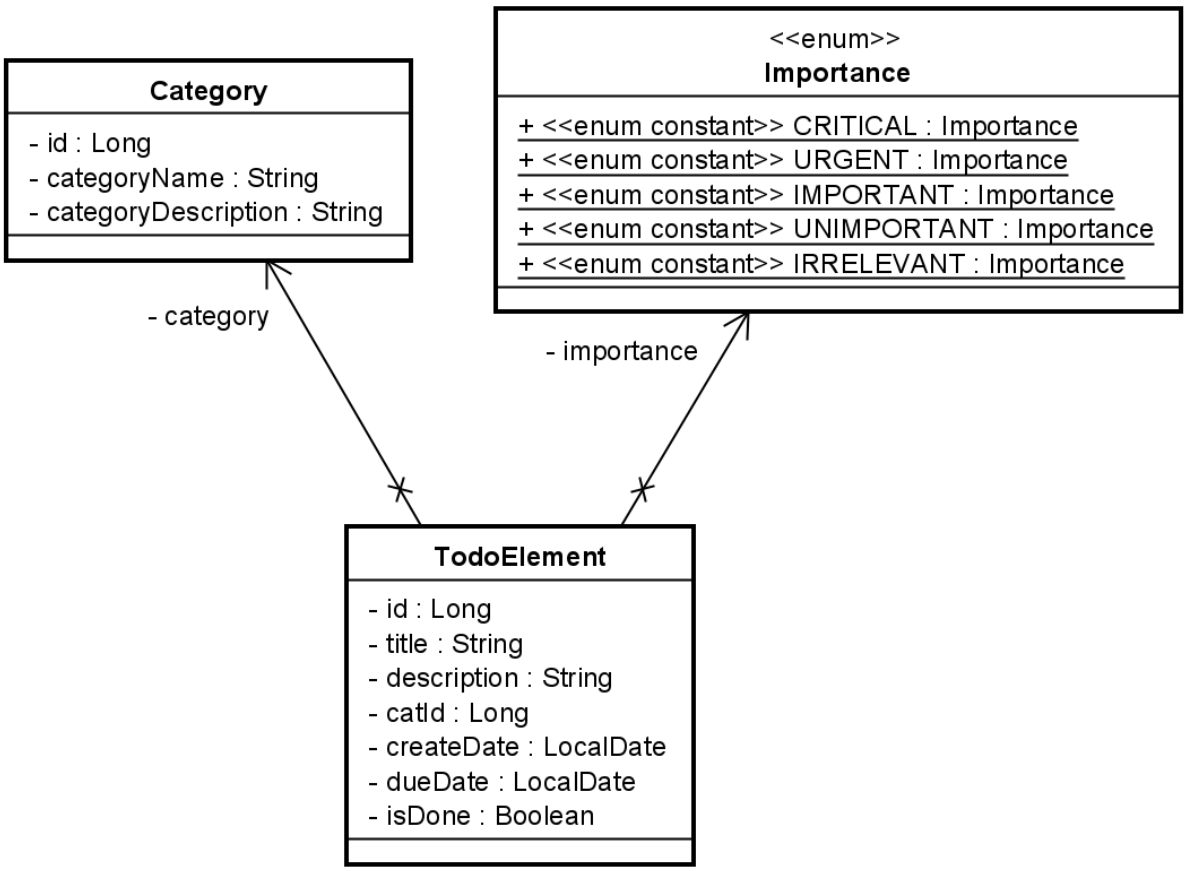
Todo List

SPRING BOOT, JSON, LOMBOK, REFLECTIONS, JACKSON

Your task is to create a Sprint Boot application to manage TODO elements within a mock database.

Create the pojos classes according to the class diagram:



`database.MockDataBase`
This class holds two lists: **`categories`** and **`elements`**. With the help of **`readCategories()`** and **`readTodoElements()`**, the **`initDatabase()`** method is responsible for reading the two JSON files (**`categories.json, todos.json`**) and correctly persist them to the lists. This method is called as soon as the class was created and initialized in the IoC Container.

`web.CategoryController`
This controller handles requests mapped to the path **`/categories`** and defines the following endpoints:

`GET`
`getAllCategories()`
This method returns a list containing all the categories.

`GET /categoryId`
`getCategoryById()`
This method returns a category identified by the **`id`** given in the path.

POST
addCategory()
This method inserts the passed category, if it does not exist already and returns the new category. Additionally, the path in the header of the response needs to be set pointing to the new category.

PUT /replace?categoryId=…
replaceCategory()
An existing category identified by the **categoryId** can be replaced by a new one. The new category including the set header path is returned.

PATCH /update/categoryId
updateCategory()
An existing category is updated. All non-null fields are updated (except the **id**). Make sure to change the object itself and not to copy it. Return the updated category.

---

web.TodoElementController
This controller handles requests mapped to the path **/todos** and defines the following endpoints:

GET
getAllTodos()
This method returns a collection of all todos.

GET /category?catID = …
getAllTodosByCategory()
This method returns all todos assigned to the passed category.

GET /categories?ids=…**&ids=**…**&**…
getAllTodosByCategories()
This method returns all todos assigned to the categories passed.

GET /priority?prio=…
getAllTodosByPriority()
This method returns all todos assigned to a defined priority.

GET /today
getAllTodosDueToToday()
Returns all todos, which are due to the current date.

GET /due?date=…
getAllTodosDueToDate()
This method returns all todos, which are due to the passed date.

GET /future?days=…
getAllTodosDueDatesWithinTheNextDays()
Returns all todos, where the due date is within the next passed days. The first day to count is today. So, passing 1 would result in all todos which are due today.

GET /todoId
getTodoById()
This method returns one todo element identified by the **id**.

POST
addTodo()
This method adds a new todo element to the list. Make sure that all values are correct. The new todo including the set header path is returned.

PATCH /change?id=…
changeTodo()
This method can change all non-null values (expect the **id**) of an existing todo element. The new todo element is returned.

PUT /replace?id=…
replaceTodo()
This method replaces an existing todo with a new one. The new todo element is returned.

PATCH /todoId/done
setTodoToDone()
This method sets the todo with the passed **id** to be done. The element is returned afterwards.

annotations.GeneratedValue
This annotation can be applied to any of the **ID** attributes in the pojos classes. When set, the read **id** in the JSON file is ignored and an autoincrement **id** is created. The start value and the increment value are set to 1 by default and might be changed by the user.

Use Postman and the IntelliJ HTTP Client to test all your endpoints.