Introduction

My application example is of a Furniture Store Database which contains an ID, a manufacturer, a product ID, a product type, a furniture type, and its availability (whether it is in stock or not).

An example of a tuple:

A screen shot of a computer code

AI-generated content may be incorrect.

I chose this application product idea as there is a large variety of furniture products in the world, which contains a large possibility of examples I can store in my website. I also chose it because it is simplistic. I am definitely confident in creating sample data, searching through sample data, and changing sample data in regard to furniture.

It also appealed to me due to the fact its very user friendly, and not niche, anyone could look at my work and understand how it functions, and how to navigate it themselves.

The application is created VIA Spring Boot using Eclipse and is then sent to a local host of “8080”, where I can then send client commands VIA POSTMAN.

My application contains 4 different GET() methods, 3 DELETE() methods, and a POST() method.

Description

My application uses many various elements and tools within the Project, being the Rest API, POSTMAN, Spring Boot, Spring JPA, H2 + SQL, and basic Java methods such as exception handling.

* Spring Boot

What is used to setup my application and then inject into a host link of my choosing using a pom.xml file.

A computer screen shot of a program code

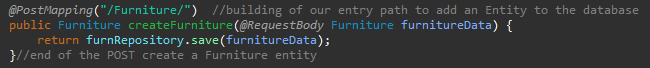
AI-generated content may be incorrect.

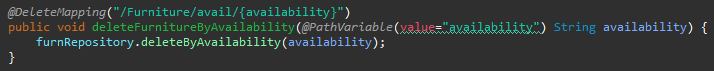
* Rest API

The API I use that follows the principles of the REST architectural style. In my app I use HTTP methods to Create, Read, Update, and Delete tuples of data in my database using the defined methods “get, post, delete”.

A screen shot of a computer code

AI-generated content may be incorrect.





* Spring JPA

Spring JPA allows me to create my database and methods more efficiently within Java, without the need to directly use queries from the other Coding languages. Using annotations, it let me create an interface with methods I can then reference to inside my Controller class, (for example, in the code directly above, “furnRepository” uses a method, “deleteByAvailability()”, which I created in my Interface class. A screen shot of a computer program

AI-generated content may be incorrect.

It can also be seen when I create my controller class and define methods:

A screen shot of a computer

AI-generated content may be incorrect.

* SQL and H2

A screen shot of a computer

AI-generated content may be incorrect.

This is when I create my database inside my Furniture.java class, as you can see, I use JPA annotations to set where each piece of information goes within the dataset I send off.

* Exception handling

Basic Java exception handling, used when a exception is raised, for example when a set of data is requested, but it does not exist.

My ResourceNotFoundException class:

A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.An example where its used:

Method

There are five main Java classes used in my application, all being inside one package named com.tus.license, however, four of them, are then stored inside their own package, within the package stated.

The four files in their own package are:

FurnitureController.java – which is where I define the methods used when the client sends their own queries.

Furniture.java – which is where I create and define the furniture object, including any variables, get() or set() methods, and the toString() method. I also define the database table and where each of the relevant data would map within the table.

ResourceNotFoundException.java – My exception handler class. Where I return any error messages if an error is met.

FurnitureRepository.java – where I create my repository interface, and create each method I reference in my controller class when a client sends a specific request.

My fifth java class is simply my main class where I run my application: A00320562Application.java, which can be seen above in my example for my use of Spring Boot.

A screenshot of a computer program

AI-generated content may be incorrect.This is how the classes look within my Java Project:

* Furniture.java

A screen shot of a computer program

AI-generated content may be incorrect.

The stating of where each piece of information entered into the table created “LICENSE” will be referring to.

A computer screen shot of a black screen

AI-generated content may be incorrect.

The constructors for the class, setting where inputted parameters will be stored within the specific object when it is called and defined.

A screen shot of a computer program

AI-generated content may be incorrect.Each get and set method for each variable within the Furniture object.

When a get() method is called, it will return the stored value it is requesting within the Furniture object.

When a set() method is called, it will replace the specific old information stored in the object with the new inputted information from the parameter.

This is the basic toString method I override, it is also the end of the Furniture class.

A black background with blue and green text

AI-generated content may be incorrect.

* FurnitureController.java

A screen shot of a computer code

AI-generated content may be incorrect.

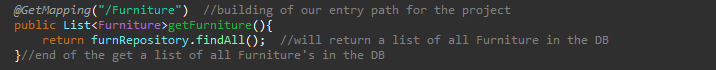
The creation of a reference to an instance of my Repository class.

Also stating what will appear in the default of my localhost link, without requesting any data from my application.

[http://localhost:8080/](http://localhost:8080/Furniture/id/4324) leading you to a blank screen with:



I will now go over the creation of my repository methods, I will provide examples of the code, but no output, as that will be covered later in the TESTING section of my word document:



The creation of my “get all” method, which will return every tuple in the database.

A screen shot of a computer code

AI-generated content may be incorrect.

The creation of my get method specifically returning the Id. The first line @GetMapping, establishes how to call the method, so an example I would need to follow to get a furniture with Id 2, would be: “http://localhost:8080/Furniture/id/2”.

The second line defines the method, it is a response entity with the type of <Furniture>, named “getFurnitureById”, with a parameter of type Long, named “Id”. The value for Id is then retrieved from “@PathVariable”, which is referring to the {id} in the line above.

The third line then creates an “Optional”, of furniture type, named furniture. An optional is a datatype that is used in case the variable would be empty and not return a null error. I then call an interface method on the created repository from the start of the class, and pass in the parameter retrieved from the client.

If the “furniture” variable is not empty, I then return the sorted list. If not, I throw an error and the request does not go through.

All the get() methods follow the same style as this one, so I will not re-explain anything said here and continue on showing the rest of my methods.

A screen shot of a computer program

AI-generated content may be incorrect.

These are my next 3 get methods to sort through availability, manufacturers, and product types. Example links to call these methods would be:

“http://localhost:8080/Furniture/avail/Availability”

“http://localhost:8080/Furniture/manu/Ikea”

“http://localhost:8080/Furniture/type/Bed\_Product”

A computer screen with colorful text

AI-generated content may be incorrect.

This is my get() method for furniture types, which works the same way as the other get methods, but required two parameters, the product type, and the furniture type. An example using this would be:

“http://localhost:8080/Furniture/type/Bed\_Product/King\_Bed”

This is the end of my get() methods.

A screen shot of a computer

AI-generated content may be incorrect.

This is my post method, which can create or update an existing entity in my database. An example link to this would look like:

<http://localhost:8080/Furniture/>, but with extra JSON code in the body of the request, such as:

A screen shot of a computer code

AI-generated content may be incorrect.

If the id does not exist, it will create a new entity with the id provided, but if it does exist, it will overwrite the existing entity.

This is my only post() method.

A screen shot of a computer program

AI-generated content may be incorrect.

The delete methods are very simple, deleting any data with the provided parameter.

* ResourceNotFoundException.Java

A screen shot of a computer program

AI-generated content may be incorrect.

Basic exception handler class.

* FurnitureRepository.Java

A java interface that helps the application carry out tasks via Rest JPA.

A screen shot of a computer program

AI-generated content may be incorrect.

It consists of the interface extending the imported JpaRepository, and defining any used methods in my controller class.

Screenshots (of database)

This is a screenshot of my “data.sql” file, which is where I insert my data into my LICENSE table I created in my Furniture.java class.

A screen shot of a computer

AI-generated content may be incorrect.

The following is the data portrayed in JSON via my browser, my JSON via POSTMAN, and then my JSON via the table provided by POSTMAN:

A screen shot of a computer screen

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect. A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Test Results

A black rectangular object with white lines

AI-generated content may be incorrect.Each get() Method in use (excluding getFurniture/findAll, as that is above):

The validation of the operation is located top right of the photo, (the 200 OK)

A black rectangular object with a black border

AI-generated content may be incorrect.Below will be an example of a not found request (404) and a bad query request (400), to make sure I cover all response messages. I will not do this for every method. (zoom may be recommended)

A black rectangular object with white lines

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.The rest of the get methods:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

This is now the showcasing the Post method:

A screenshot of a computer

AI-generated content may be incorrect.(data with id 0 before the post, and then the post Query with JSON):

A black screen with blue and white text

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.The aftermath of the POST query:

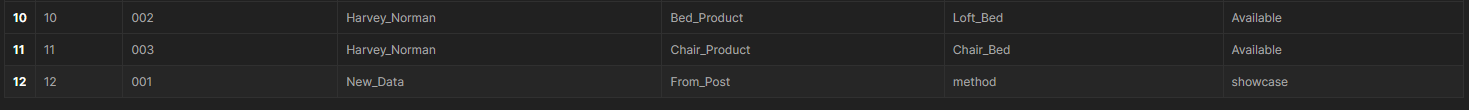
An example of a method not allowed error (405), as I attempted a get query with the post method:

A screenshot of a computer

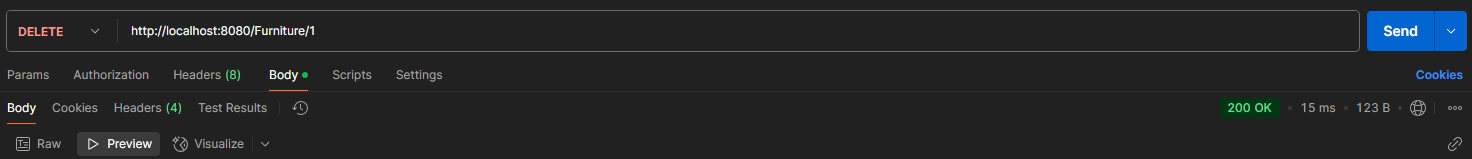
AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.This is an example of using post to create new data instead of updating as I have:

Added to end of table (this is after findAll get request):

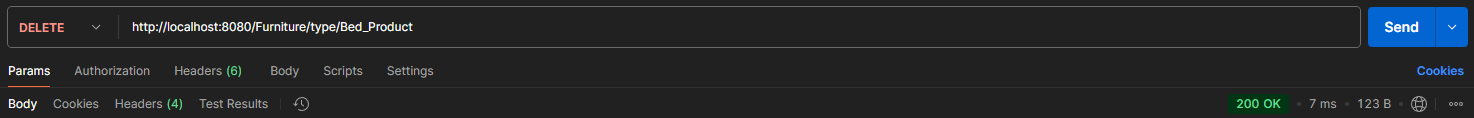
The delete methods here:

* via id:

After (id 1 is gone):

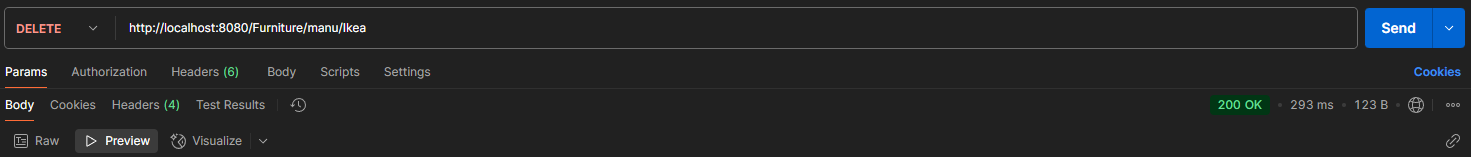
A screenshot of a computer

AI-generated content may be incorrect.

* Via type

A screenshot of a computer

AI-generated content may be incorrect.After (all beds are gone):

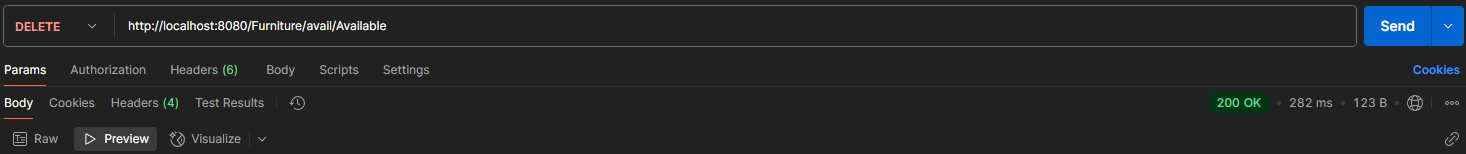
* Via manufacturer

After:

A screenshot of a computer

AI-generated content may be incorrect.

* Via availability



After:

A screenshot of a black screen

AI-generated content may be incorrect.

Evaluation

I believe I applied the seven application requirements well.

A screenshot of a computer program

AI-generated content may be incorrect.

Every requirement I have followed directly and have been applied in use in my application. I also have optional methods that were not required.

I encountered many problems but dealt with them accordingly. An example would be when creating the methods, not being consistent with naming schemes and variables across all files, leading to runtime errors when I would try to boot up my app.

A screenshot of a computer

AI-generated content may be incorrect.Another example would be when I tried to call my POST method, but had the wrong syntax for the JSON body, accidentally adding in [ ], resulting in an error I couldn’t wrap my head around as I thought I was right.

Further examples would be adding in an extra “/” at the end of my queries or not having @Modifying or @Transactional for my delete methods, which I didn’t need at first (or at least for my Id delete method), but it wouldn’t work without them later on.

Conclusion

I am happy with the app I created and the ending functionality of the app. The subject was simplistic, and the code works completely as intended. I put in an appropriate amount of time to get the result I was searching for, and I am happy with the methods I used and implemented. Each test resulted with the correct answer I needed and there were no bugs I could not fix.

Overall, once again, I am happy with the end result of my application and I had a lot of fun working with Spring boot and the Rest JPA/API.

Appendix

This will include mainly duplicate code, as I was sure to include code within my word document, but it is an appendix to view each file individually in its whole. Zooming in may be required, each page is a separate file (which can be seen on the left), unless otherwise specified.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.A screen shot of a computer program

AI-generated content may be incorrect.

-Exception handler

A screen shot of a computer program

AI-generated content may be incorrect.

-Interface

A screen shot of a computer program

AI-generated content may be incorrect.

-data.sql file

A screen shot of a computer

AI-generated content may be incorrect.