<https://www.youtube.com/watch?v=YoBFTSKNrD0>

SPRING BOOT + API REST + JPA + QUERY METHOD + DTO - YouTube

https://www.youtube.com/watch?v=YoBFTSKNrD0

Transcript:

(00:02) Welcome to this new video where we are going to be talking a little about the cuerit meto and how to work on them in a springbook project to make a Rest API. But before going to the code we are going to understand a little about the architecture of a network API developed in sprintbook like We can see the sanatation plays an important role when it comes to defining our apirez and what are the annotations, well basically they are annotations as their name says that are added to the classes, methods, Fields and

(00:41) so on to define or generate metadata for the different classes or methods. or whatever we are adding the annotation to in this way we can somehow dispense with the configuration xml files that we were previously working on in Java. As we can see, first we obtain an annotation called Rest controller that will basically allow us to indicate which one it is.

(01:11) The class that is going to be the controller, that is, the class that is going to expose the routes, that is, the empons, and it will be able to receive http requests and respond to them as well as perform the corresponding validations. Remember that although we perform validations of the On the frontend side, it is also necessary to perform the validations on the Baker side.

(01:40) Here we can also observe some other annotations that will allow us to specifically define how that network controller behaves. We have the specific methods of how to work with http requests through these annotations. We also have How the data is going to be received And it is going to be through the same route through parameters or in the body of the http request And also how it is going to respond Yes how are they going to in this the frontend or that application client How are you going to access the answer yes, let us remember that a

(02:25) network API is nothing more than an object or a service that will allow us in some way to facilitate the exchange of information and data between a service that is online through a simple url address and in this way it will be possible to do secondly, we have the service, that is, the Rest controller will invoke the service.

(02:52) As we can see in the image, the service is nothing more than another class that will basically be in charge of connecting. to one or several repositories and also what is related to the business logic of our application And there is some business logic that we should do before well finally we have the repository that is defined through the repository annotation, notice that Here, unlike the previous ones, we are talking about interfaces, remember that an interface is nothing more than a contract that the class that implements it must comply with, that is, we are

(03:40) not going to write the specification of the methods but we do have to define what those methods are. we are working with jpa as in the case of the example that we are going to see. So we Thanks to that repository given that it is going to implement Yes jpa we will be able to access methods such as Side fine all fine by ID among others that will help us allow in some way to manipulate the data without writing a single line of code obviously related to the manipulation of the data, that is to say that it will bring us to describe, for

(04:22) example, the SQL instructions that we know so much, that is to say that we are going to be working as if it were not specifically we are going to We are going to be working with objects and we are going to extract ourselves from the concept of tables and the concept of SQL instructions to manipulate the data in those tables as long as we are obviously talking about a relational database, of course jpa is going to provide us with these methods that I mentioned previously but also we are going to be able to create

(04:59) some other methods following some rules that we are going to see a little later based on our own needs Yes because we see that basically a basic club yes that is to say that I It will allow you to create and modify Search by identifier by identifier and not much else. So we are going to surely need some other more specific methods and for that there are the cuerinatos, which is part of what we are going to be talking about in this video.

(05:34) Well, finally, we have the entities yes the entities that are obviously going to be mapped to tables So to speak of the database and they are nothing more than classes that in some way And thanks to the mapping they will allow us to manipulate the data as I said previously without having to write code related to sql and so on, that is to say that we are going to work with domain-specific objects and we are going to manipulate them, the rest is going to be transparent for us here as you can see I am already

(06:12) in netbeans And in this case I already have a Sprint project specifically Following the architecture that we were talking about previously, I have the controller, the model, the repository and the service to work with our API. In this case, I will be working with the person entity and that is, this entity that for us will be objects, here we have as we saw previously.

(06:39) the annotation entity and notice that it is a very simple class that only has as attributes the identifier, the location, the position, full name and the image, it also has its corresponding constructors and its corresponding methods. In this case, for the detericenter methods I am using the longbooks library.

(07:05) But if you do not have it directly, you can generate them by clicking with the right mouse button and here the option that says insert code and select the option ghetter and setters in this case they do not appear because it is already implemented as you can see I also have some annotation specifically is that they are important for the mapping yes as being the entity the identifier generated valid basic in others more Yes then we could add others more That's what I mean yes this basically corresponds to thanks to that mapping this is going to be created

(07:43) the base of data in this case the database does not forgive it has to be previously created the table in this case person is going to be created as it appears here yes where each attribute in my class is corresponds to a column in my table person Yes and the entity It is going to be converted into a table of course the manipulation and access to the data of this is going to be transparent for us and now we are going to see it well let's go back to our project then I am going to be working basically this

(08:18) Api that I created previously and It will allow you to make a basic basic club with that person table, that is, I could obtain a person from their identifier, I could modify the data, I could add new records and so on, yes. So let's first see how it works and then we will go specifically to the cueritos good So first we said We have to have our controller Yes so here we have our red controller class yes as we said it is who is going to specify the routes yes notice that here we have the routes defined by the

(09:02) lanotation get mapping and there we are going to specify the route in this case we have get Martin to list all the people we have in our database in this case to create a new person notice that we are already working with the post method Yes to eliminate Yes and to obtain by identifier and to modify that is a basic club as we said if this controller basically has to connect So to speak no or in reality what it has to do is call the service here we can also see it and the service is the one that as we said previously is going

(09:48) to communicate with one or more repositories How can we see here and the repository in this case we have only one repository it is an interface as we had previously said that extends jpa Yes and jpa we had said that it provided us with those methods to do the raw like Side find old fine by ID and others For example, if we are going to click here on jpr repository, we select the Navigate option and then we are going to inspect the methods, we are going to see the methods that jpa provides us with, as you can see, I tried find all, find

(10:33) by ID, save all, 6. method 6 that I think was there and there are other methods Yes, the get by ID and these methods are methods that will practically allow me to manipulate the database but speaking of objects always from the perspective of objects that is to say that I'm not going to write sql as we had said to create a record in the person table or anything or anything like that, say I'm going to create a person and I'm going to search and so on all those methods to make that basic club I tried them.

(11:15) jpea And we can see that in the service. Yes, notice that we had said that the service in this case is the one that is practically going to communicate with the repository and the repository. I tried these methods. Yes, that is to say that I am simply going to put a here. point and there I also have all the methods.

(11:39) I do not write these methods but you can use them because thanks to the mapping and jpa we can solve it so here I can access 6 to create a new person. Note that it does not distinguish between creating and modifying. We simply have a save method that will allow us to create it if it does not exist and if it does exist, it will modify it to delete, it provides us with the delete method to search for all or to list all the people.

(12:11) What I have in my database is going to be the find all to search by identifier we have the find by ID among others And then let's see how this works for that I'm going to go to postman well and here I'm simply going to take the routes into account Yes, let's remember and we're going to see our controller again so we're looking at it and at the same time We're doing the tests, remember that to list all the people I would have to work with the get method And in this case person bar Yes this would be the route So

(12:49) first comes everything that is The domain and then the route that corresponds to ellenpoint Yes then I 'm going to go back to postman So here I have it I have already raised the service yes I raised it yes look here I have it raised to the service so I'm going to return here and we are going to send of course we do not have any record Yes then what we are going to do is create a new one for that Again I go to my controller and notice that here the route is the same but what changes is the method I

(13:25) I have to pass through parameters, yes in this case the person object, how am I going to pass it as boots, that is, I am going to use the J Jason format, which is a data transmission format, one of the most used, there are others, but this is the one we are going for. to be using us So here I simply have to change the post method and here remember in the body of the message because here it is also specified that it does not go in the body of the message The object will go Yes in json format so I come here I have to select this

(14:05) I have to first select what is going to go in the body I am going to select this option and of all the options I am going to choose the json option then we are going to write the json that corresponds well notice that I am not specifying absolutely all of them but yes It is important that the names that I specified here correspond to those of my entity.

(14:34) Yes, notice that I have location, position and full name. Yes, if not, you will not find them because the mapping is in the middle. Yes, then here we are going to send. Yes, notice that It returned a 200, which means that everything is fine. It is returning the record in the image and it put null because I specifically did not send it, but in theory it already created the record in the database. Let's see if that's the case.

(14:59) Let's warrench here. Let's go. refresh and as you can see yes the record has actually been created in the database Now I am going to go to list I come here and I am going to list then I am going to use the get to see all the records that I have yes this is no longer necessary that it is if they leave it Nothing happens and we are going to send and now I have a record What happens now if I want to modify it Well I have to go to the controller Yes, let's see in this case in the controller we have the modify person, notice

(15:40) what changes Although the route in this case is the same, what changes is the method Yes and the data also goes in the request Yes in the body of the rich it is good Then we are going to go to postman again Yes we are going to change the method we are going to put it put here I'm going to copy and what I'm going to do Obviously since it's modifying we have to pass it the identifier because it's going to search for it by identifier Yes then here I'm going to change it and I'm going to put

(16:14) chubut and we're going to send notice that again It returned a 200, that means that in theory the backend worked or at least there was no error, I'm going to go Sorry, to the database, let's refresh it and see if this was actually modified, yes, that's the essence of a Api Rest is going to expose functionality or it is going to expose functionality or also how to manipulate the data through URL Of course the entire Security part is also included here that we are not going to see in this video, that is not to say no.

(16:58) Anyone I could enter and access the data and it is configured But well for our example it is perfect understanding all that let's go to an example let's suppose that I now in my controller I need a method that is called for example login yes that is to say that I am going to go and search A person does exist in the database, yes, but based on their username and password, well, for that, obviously the username and password data must persist in the database.

(17:41) So initially I am going to leave in this case my person entity and I'm going to add those two attributes here. Of course you could create a new user entity and relate it with a one-to-one relationship to the person, but to make it easier and to understand the concept of cueritmétum, which is where we want to go.

(18:03) I'm going to work directly on the person entity, so I'm going to specify the user and password here, and I'm also going to add them in the constructor. Remember that it would probably be better to separate this data into a new user entity, yes, well, there I already have my person entity that It has the username and the password well, so I 'm going to go to my controller and I'm going to create a new effort.

(18:36) Yes, in this case, since we are going to pass sensitive data there, it is not advisable to use the get method. Therefore, I am going to use the post mapping. Well, I'm going to go get and I'm not going to make any modifications to the database, the post method is generic. Well, then we 're going to create, we already have the route.

(18:56) We're going to specify the method. I'm going to make it return a person object. So yes. find the username and password corresponding to the person who returns the record to me, we are going to pass it through parameters in this case the person object The idea is that the data comes Yes in the body of the request now well Yes We know that not attract all the data of person, but yes, the username and password could, if you want, use another specific object here to bring only those two data, but for the purposes of simplifying the coding here and going to the

(19:40) example, I am going to leave person. Yes, well, as you can see, then the controller is communicate with the service yes Here I am going to do a retune new personas simply so that it compiles but in this case let's remember that what we need is to call the service who is going to practically provide in some way the person object if the username and password exist But what happens? I don't have that method.

(20:16) Look, I'm going to go to the service and I have the delete person, the Search by ID and the create, yes. So I have to go to the service beforehand to create that method. Well, then I I'm going to go here to the service I'm going to create too, but this one works with the repo, let 's not remember, this one is going to communicate with the rest, so I'm going to create a new method, I'm going to specify the login that it's going to receive by parameters and we're going to make it receive by parameters the

(20:47) user and the password as we said in this case the service in some way is going to be connecting with the repositories which are the ones who are finally going to be the connection with the database and are going to manipulate the data And then I have I have to access my repo person in this case I am working with only one and Access the method that allows me to Search by username and password And what happens Yes I am going to put here fine I have no way yes that is to say I have as an identifier that the identifier a It won't come to me

(21:25) because only the username and password will come to me. Well, in theory, I could go to find all and then go through and return depending on when I find the username and password, but that wouldn't be the right thing to do. So what do we do in this case and here? It's not a Boy Sorry what this is going to do is return an object well then well here for it to compile we are going to put return New person But then we are going to come and modify this simply so that it doesn't break and it continues compiling Okay but then

(22:02) what's happening I have to go to the repo and the repo we had seen that it was an interface yes it basically extended from jpr repository well this is where the cuerinatos come in yes that is to say that as we said although JP a I tried those methods that we have previously Obviously they are basic methods when we already want to Search yes or have a slightly more complex search that is not for just an identifier we are already in trouble but the cuerimentos are going to help us for this and What is a good cuerimento look at the

(22:43) quilling metal basically from jpa They provide us with the possibility, as it says here, of generating queries using the name of the methods. I simply have to take into account that interface, as we said, it is an interface. I do not do the implementation but I do have to define the name of the methods and with the simple name of the methods jpa is going to take care of the implementation And it is going to solve it for us, for example if I want to search by username and password like the example we are seeing I could create a method with

(23:19) these words Yes there are words that are reserved like for For example, find, byte and or and so on, and I am going to use them depending on what I want to achieve. Yes, for example, if I want to search as in this example by user and password, the method will be find by user and password and it will we are going to pass the parameters if we go to the sprintbook specification specifically in relation to the cued engine in this case the reserved words for which we can find the fine by ridby get by si existbi look these are some but

(24:01) know that there are many more to work the logic we can use the Land or notice that we could also check if it is empty practically we can make those pueri as complex as we want and simply specifying the name of the method yes where In our repository well So let's do the test yes in this case I am going to define a method and it is going to return the person object well and how it is going to be called is going to be fine by user and password notice that the nomenclature that we have to use is camel Case yes the

(24:52) camel Case nomenclature for the methods true because notice that for the names of the classes and interfaces it is Pascal Case camel Case means that the first word yes the first letter of the first word is lowercase but in the next one notice they are capitalized this is super important for in this case for jpa so you can identify where a word begins and ends and it also helps us read better.

(25:26) Well, it is important that you write it this way. The username and password are basically the names as they exist in our database. Yes, it is. say in the column is the attribute it has to be as is yes So we are going to put here String user and String password yes Obviously these also have to be reflected in some way yes And here notice that I am not writing the implementation because this is a interface I simply put a semicolon and that's it yes the implementation who is going to do it jpa is going to do it for me yes I'm simply

(26:06) going to specify that I also want it to provide me with this method yes And in that way Then I'm going to go to the service that now yes I can delete this if I already have that method because I just created it Yes there is fine password and what the user is going to receive and the step we are going to return this and I already have the service Now we do Let's go to the controller that we had also left here unfinished in this case obviously it is going to Access the service and now I already have

(26:45) the login method Yes Obviously the user receives the password but they are part of my entity Yes so I can through the getter properties access and the user and the password as it says modifications in the entities, just in case I am going to download in this case I am going to stop running my API and I am going to raise it again and we are going to check with postman to see if this works I'm going to go to the database, we're going to refresh, as you can see, I already added the two columns for the

(27:22) mapping issue, we're going to put here as password 1 2 3 4 and as user Ivana and we're going to update and then do the tests correctly. We are going to go next to postman and here notice we had defined the post method and I am going to come here So we refresh the spos method the route is login and it will receive all the data in the body of the request well then we come here and here I have to change because now it's not a person but it's login again login and here I'm just going to give you the

(28:10) username and password. Yes, I'm going to modify this. We're going to put it in and use it. We said that the user was Ivana and the password will be. Let's go. let's get a two three four let's see if this actually works it is bringing me the data yes Obviously it brings me the entire record of people Yes if I write something else here and I put Send it is not finding it although it is not returning anything but no There is an error Yes, so up to that point everything is perfect but we have two problems. The first problem

(28:55) is that we are returning the password and the user, which is sensitive data, should not return that. On the one hand, Yes, if you used two, Two classes, one for the person and one for the users. They could easily solve this by returning person and they would be returning the user's data, which is not happening in my case and on the other hand, remember in the database, although This is in the database, it should not be able to be seen like this at first glance.

(29:28) and the password this should still be encrypted in the database Yes but well this case I leave it to you to solve now we are going to solve in this case and precisely to incorporate a new concept that has to do with discount to avoid that in this case these two data are returned And what is Teo dto basically it is a data transfer pattern Data transfer object and its purpose is basically to create an object from a series of attributes that are going to be sent or retrieved by the server in a single invocation, that is, at a single moment, these attributes can

(30:17) come from a single source or table or from multiple tables. A common mistake that in some cases developers make is to use entities that, as we saw here in the example, are mapped directly to the table. of the database to transfer the data if we should create the data Yes then in this way we will be adapting in this case to the frontend and not be sending unnecessary data what do we have to solve in this case But well the implementation is quite simple, we are simply going to create a new class that will not be

(31:05) mapped to the database, it is simply a class that we can put at the end the name of to to remind us of the concept, which is ultimately what is going to be returned in this case what our API is going to return let's go to the example. So yes, normally we are not going to work on it in the controller, the ideal is to work on it in the service since it is going to be in charge of the logic.

(31:36) Remember that the controller is going to be in charge of the requests and the validations regarding the requests but not the logic, the logic there is better to work on it in the service. So we are not going to go to the service and now we are not going to return the person object, we have to return another object that will not be mapped the database Yes so here inside my service what I am going to do I am going to create a new class and I am going to put Teo person Yes in this case because the Source is going to be a single Yes but there could be more Yes then

(32:13) I I am going to put well This is going to come from the person table Yes but it is not complete Yes because in this case I want to get the username and password Yes we are going to finalize this I am going to delete it I am not going to work on mapping nor am I going to put entities here or absolutely nothing this is not mapped we said so here I am going to create my attributes I am going to come here and I am going to copy to make it faster I am going to have all the data we said except for the username and password Only what

(32:50) we want so we are going to paste here I'm going to get it, we said, Oops, you added the annotation, we're going to get the annotation, we don't need it, that is, I want you to return all the data Yes, but no. In no way do I want you to return the username and password, then I'm going to create the username and password as well.

(33:11) constructor and we are going to add the properties and I am going to bring this from a longbox just like this one but if not, if you do not want to use this library directly, add them and that's it, I have my class which is the one that I am going to return in what I am doing here an error, remember that the nomenclature is Pascal Case for both classes and interfaces Obviously I have to rename the constructors And now what happened Ah it's asking me to rename the file Perfect there it's good I already have

(34:06) my Teo persona class and I just do it What do I have to do now? The login is no longer going to return the person, but rather it is going to return my Teo object. Yes, but I have to do the mapping now, that is, I have to start with the person who returns the fine by user and password that is going to return a person object that is an entity so I am going to create a variable of type person and what the method brings I am going to save there I am also going to create another one but now I deactivate it which is the one that I

(34:46) really need to return and now I am going to instantiate using the parameterized constructor that first had the ID but I have it in my object yes get ID then let's see let's see here Remember that the order is super important as well as the data types of each parameter At the time of instantiate or create an object of Teo's person class and the number of parameters also if no it will not work Yes then location position Yes then here person dot get location person dot get position if I'm not mistaken anyway now we are going to review it

(35:43) person dot name and I have the image left I'm going to download it so you can also see it comma person dot image okay Let's arrange a little and let's return Now yes our Teo object is perfect okay let's go to our controller remember that now the login method It no longer returns the person object but is going to return our Teo object.

(36:19) Yes, we are going to add the dependency directly. I don't know why it does n't appear there automatically. Let's remember that this is within the service. So here we should find it perfect. This is already compiling. we are going to lower the service we are going to raise it again and we are going to see if everything works well I already have a pencil up we are going to go to postman and we are going to try again notice that it is no longer returning the username and password well up to here We arrived as we could see, not only were we working

(36:58) with leather and method and trying to understand but we also reviewed a little what the architecture of a Rest API developed in springbook is and also this video was specially requested by my students from Argentina, a program that they I send a big hug that they will soon deliver their project So there are many successes let's go for it Bye bye

Github

<https://github.com/andresestebanvega/apirestPersona>