**161. Configuring Route Parameters:**

* -: In the last lecture, we started adding our child routes and now loading the recipe detail component doesn't work anymore.
* The reason for this is that here we use recipe all over the place and that's a property of this component but it's a property set through property binding with @input in front of it.
* So it awaits to be set from outside the component.
* Since this component is set through routing, though, this won't work anymore.
* So we have to change how we receive the recipe here, how we set the recipe, and we can change it because keep in mind in our route, which we define here, we passed this ID, which is a dynamic parameter.
* So we should be able to load a recipe by the ID which will simply refer to the position in the array of recipes.

Text

Description automatically generated

* So here is your challenge.
* Try fixing the recipe detail component to somehow use that ID to update the recipe which is displayed to fetch it from the recipe service.
* Of course, you will need to implement some method in the recipe service, which allows you to get that.
* Here is your chance to pause the video.
* Thereafter, I will solve it together with you.
* So, were you successful? Let's fix it together.
* In the recipe detail component, we clearly can remove @input because that's not how we receive the value anymore.
* So we can also remove the import of @input.
* That of course also means, before we forget it, that in the recipe item component, we can remove the on selected click listener here because that will not be how we pass the value anymore.
* And of course we could also remove the on selected method here.
* That is something we don't need anymore.
* Of course, that also allows us to remove the recipes service from this component and remove the import.
* So we're cleaning up quite a lot here.
* In the recipe list component, we don't have to change anything.
* And in the recipe component itself here, we already did make a change and add the router outlet.
* So now in the recipe detail component, we somehow need to get our single recipe and we know that we will get the ID from the router.
* So let's first fetch this ID and you learned that for this we need to get access to the activated route.
* So let's inject it here, of type activated route.
* Make sure to add this import here from @angular router.
* This import is important to enable type script to identify the type you're trying to inject here.
* And if that angular is able to use it.

Text

Description automatically generated

* **So now we get this route and now we have two ways of retrieving the id.**
* The first one would be to simply bind it to route, snap, excuse me, this, route, snapshot, params, id.

Text

Description automatically generated

* But this will only work for the first time we load a detail component.
* Now keep in mind, the menu where we can click on one is on the left.
* So we will be able to choose a new recipe whilst we still see the details of another one.
* So we probably want to react to changes in our recipe ID because that's absolutely possible in our app.
* So that is not the approach I want to use.
* ***Instead, what I want to do is, I want to use my route, the params observable and subscribe to that observable.***
* Now with that, we can react to any changes in our route params and we will get them here of type params, as we learned, to be imported from at Angular router.

Text

Description automatically generated

* And now, now we can really react to a new ID.
* So we should probably store that ID here in a property of our class.
* It will be of type number.
* So we should set this ID equal to parents' ID though we know that what we get back here will be a string.
* So we should cast it to a number by adding a plus in front of it.
* So now we will have an ID in the form of a number which we can use to identify our recipe.
* Well with this information, we want to load our recipe from the recipe service and forward it.
* We need to change our recipe service because right now we got no way of loading a single recipe by ID.
* That's something we can easily implement though.
* We can simply implement a get recipe method which takes an ID of type number.
* And then we can simply return this recipes and select the item at this index.
* We could also rename ID to index to make really clear what that number will be in the end.
* Text

  Description automatically generated
* You could also call slice before doing that.
* Because slice would now give you a copy of your array, though it won't be a deep copy.
* So the objects still are the same anyways.
* So you can also directly return the object like this.
* You could of course also create a hard copy of that object with object assigned, for example.
* That's not what I want to do here though.
* So we return a single recipe, now back in the recipe detail component type script file.
* After fetching an ID and storing it, I want to fetch the new recipe and I will later also show why I needed to store the ID here.
* So let's fetch the recipe using our recipe service.
* Whenever the ID changes, then I will call get recipe and pass my ID as an argument.
* And with that, we should be able to indeed make our application to work again by going manually to recipes zero.

Text

Description automatically generated

* We indeed see the detail view for the schnitzel and with one for the burger.
* So that is working.
* Now, we also need to enable these links here to work again.
* Cause right now they're not doing anything.