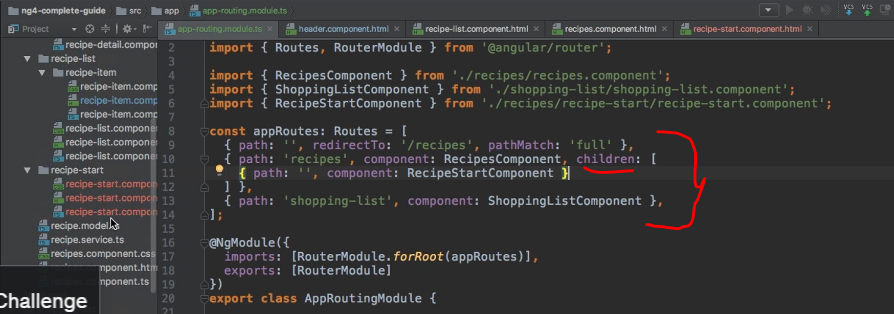
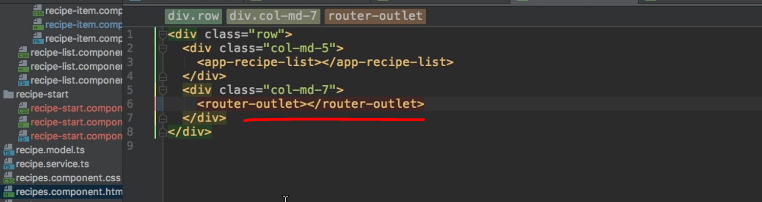
**160. Adding Child Routing Together**

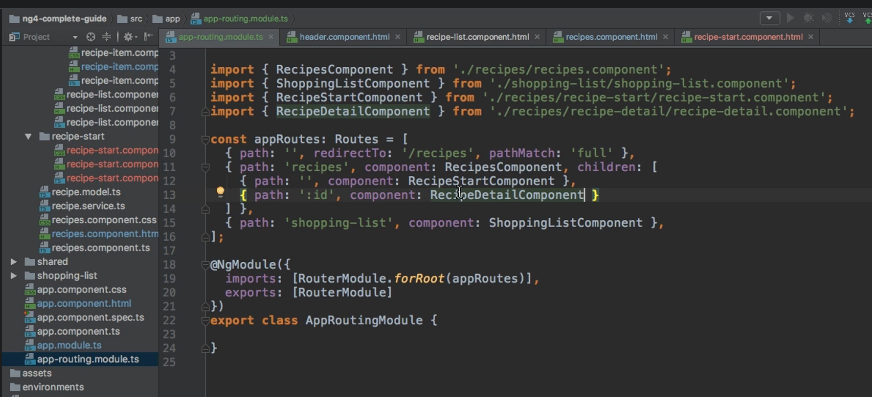
* -: So were you successful? Here is my solution.
* As I said, I want to implement child routes.
* So I will do this by going to the app routing module here and on my recipes I'll add to **children** property.
* Now, children also takes an array of routes and each route still is just a JavaScript object.
* Here the first route I want to have is the empty path so that we just visit slash recipes slash nothing.
* the default case.
* Here I want to load that new component I was referring to which will simply hold the text we have in this ng-template for now.
* Now for this, I will quickly create this component and I'll use the CLI for that.
* With ng g c, I can generate a new component and this component should now go into the recipes folders.
* So recipes and then slash and here you may simply choose that folder.
* So recipes and I will name it recipe-start.
* So the recipe-start component.
* With spec false I prevent this spec file from being generated.
* So with that we got our recipe-start component here.
* And as I mentioned before all I will do is I will output here maybe in a a tree tag now.
* Please select a recipe.
* So really simple.
* With that set up, with our start component set up here I now want to use that.
* It has been added to app module automatically by the cli.
* So make sure that this is the case.
* And now in the app routing module here I will load this component.
* So the RecipeStartComponent, this one here.
* Make sure to add the import at the top from .
* /recipes/recipe-start/recipe-start.
* component.
* And with that added, we got our first component which can be displayed inside of our recipes component.



* Let's try this out because that should be all we need to get started with it.
* In the recipes component, because again, this is and that's important, this is the component, the recipe component here is the component where our child routes are registered.
* So it is in this component's template that we need to provide a router outlet for the child routes to be rendered.
* So I will do this here.



* I will add a router outlet here, and when I save this with ng serv running, this looks pretty good.
* We see, please select a recipe and clearly that is not a paragraph anymore.
* So it seems to be our new component.
* Of course, if we click somewhere, that doesn't work anymore because we only have the router outlet there.
* Our old logic of loading a recipe is completely broken but we will restore it soon.
* Actually, we can already take the first step for restoring that functionality.
* I'll add another path here to the child routes of recipe component.
* That path should be :id.
* So a dynamic segment added after slash recipes.
* That's of course something you have to keep in mind.
* All child routes always come after the path of the parent route.
* So all these routes here will have slash recipes in front of it.
* So @ slash recipes slash id with id being a dynamic parameter, here I want to load my RecipeDetail component.
* We also need to add the import to that component up here of course.
* So from the recipes folder, then recipe detail and then the recipe detail component file.



* With that, we now at least have a route which would allow us to load this component and we can kind of see this in action.

Graphical user interface, text, application, chat or text message

Description automatically generated

* If we manually enter one here let's say it tries to load the recipe detail component.

Graphical user interface, text, application

Description automatically generated

* Of course it will break because their we try to output data which we no longer get because our old logic of transferring data from the list item to the detail component doesn't work anymore.
* But we'll fix it and we'll actually start fixing it in the next lecture.