Hello everyone. Thank you for taking time to look at my final project for Cycle 14, ExpressO. The project, as requested in the document, was a proof of concept application which tracked coffee recipes and baked goods for a small coffee shop. The core of the program gives the user access to the create, read, and update functions for the ingredients, the create, read, update, and delete functions for the baked goods, and the create, read, update, and delete functions for the recipes themselves. This is all done in Java and PostGres on the backend, and the front end employs AngularJS to create a single page application. So, let’s begin.

We’re going to begin, then, by typing in localhost:8080 to show that the Recipe Home Page acts as the front of the program as I decided that the user was likely to frequent this page most often. The background is intentionally simple in order that the user can more easily and quickly navigate the UI – though I couldn’t resist a nod to the underlying programming language in a coffee application.

We’ll move forward, however, to the Baked Goods tab. As you can see, we have the name and the price on the list page. I made this design decision under the impression that these two fields will be the ones most frequently referenced. However, if the user requires more information, he or she can click on the red link and go the view page.

At the bottom, you’ll notice the Add New Baked Good Button – I preferred it here, but as the list grows longer, it might be preferable for a user to see it at the top so that they don’t have to scroll – and when we click it, we are directed to the Add page. The three fields from the previous page are available, and if we type in Demo Baked Good, 4.50, from La Provence, it contains peanuts, and is a bagel, we would be allowed to submit. However, if we delete each of these, we’ll see the validation show. In order to prove the concept, I chose to validate all 5 fields; on client work, of course, some fields will be optional. Retyping and submitting, then, we can now see the field in the list.

Let’s say however, that it’s not a Demo Baked Good, but a dirty wu donut from Pip’s. We can click on the link, click the edit button, and make our corrections. As you can see, it shows on the list in order.

But, unfortunately, those donuts are delicious and quickly sell out, so we must delete them. If we go into the detail view, and only the detail view, we can delete these doughnuts forever. Or until they’re back in stock.

Let’s move then to Ingredients. We’ll go through the same process, a little bit faster as the functionality here is more of the same. So, let’s put in the pricey ingredient Civet Poop Coffee which is 26.50 an oz.

However, that’s not appetizing, so let’s change it to its more common name, Kopi Luwak.

And then let’s find out that Kopi Luwak is both too expensive to sell and too adventurous for the average purchaser. So let’s delete it, so that there’s no aftertaste.

Finally, let’s go to Recipes. The same function goes as before, but let’s start here with the Add. The user can add both a name and the necessary instructions before adding the ingredients from the provided list. As you can see, an empty click results in an error message, and when we add, we have a remove button. You will also notice that the user cannot select the same ingredient twice as the chosen one has been filtered out. When we are done, we submit, and the recipe shows on the homepage. The Instructions here are the same, with the exception that, as you can see in the filter, the chosen ingredients are not listed in the available ingredients. This stops the user from double adding.

As for test coverage, as you can see, if we go into the Jenkins, there’s x amount of test coverage, and if we navigate into Eclipse, we can see multiple Selenium tests, including 3 happy path tests for the Ingredients, Baked Goods, and Recipes.

Any questions?