Prachee Sarkar & Aila Choudhary

CSC 22100 - Software Design Lab

Final Project Report

5/22/23

GitHub repository: https://github.com/pracheesarkar/Final-Project-RecipeMaster.git

I. Introduction

RecipeMaster is an app that provides a catalog of entry level recipes for beginner chefs or people who are looking for simple dishes to make. It is intended to have food options sorted into categories and list the ingredients, measurements, directions and a simple image of the recipe. All recipes were pulled from online websites such as AllRecipes.com. The format of the app itself is as follows: a home page with the RecipeMaster logo, followed by a page listing the categories of breakfast, lunch and dinner, followed by a page listing recipes of each type, and finally a page with the recipe itself.

II. Functionality

The backend for this app was written in Java Spring Boot using the IntelliJ IDE. Its intended functionality was to store data of the recipes and their attributes as well as create functions to retrieve recipes. The recipe class itself included the following attributes: id, name, category, yield, an ingredients list and a directions list. The ingredients and directions list call the ingredients and directions classes respectively. The ingredients class includes the attributes of amount and name. The directions class includes the attributes of step number and instruction. Using these classes, six recipes containing all the attributes of the recipe class were made and

stored in a JSON file. Within the RecipeService class we created multiple functions to access the recipe data. readRecipesFromFile reads the JSON file to get the data of all the recipes; getAllRecipes() retrieves a list of all the recipes; getRecipe(long id) retrieves a singular recipe by its id number; and getRecipesByCategory(String category) returns a list of all recipes of a desired category. The RecipeController class uses the RestController annotation with GET to map request data and return a JSON response. The functionality of each function was verified via PostMan. For example, the url localhost:8080/recipe/getRecipe?id=2 returned the recipe with id 2.

The frontend of the app included multiple activities and XML file layouts associated to each activity as well as a service interface, a Network class to connect with the backend via the Retrofit client, and an adapter class that assists the communication and functionality between the classes. These components combined create a straightforward and compatible interface presented to the user. The home screen is presented through the MainActivity.java file, which calls in the RecipeMaster logo and also implements a “Click to Enter” button by utilizing OnClickListener. When the enter button is clicked, a categories page is presented through the CategoryActivity.java file, which lists three separate buttons for “Breakfast,” “Lunch,” and “Dinner” options. Clicking on one of these buttons navigates to an activity corresponding to that respective button, and these activities are named BreakfastActivity.java, LunchActivity.java, and DinnerActivity.java. Each activity includes buttons with different menu options corresponding to the category. For example, LunchActivity.java includes OnClick buttons for pasta and chicken wraps. There are also activity files for each food option, such as WrapActivity.java, which display the recipe for the respective food item on the screen. These activities each have their own

XML file, and by utilizing TextView and ImageView, the XML files serve as placeholders for the recipes and images. When these XML files are combined with the activity files, the detailed recipe, instructions, and food images are generated on the screen. Lastly, each activity file contains a line of code that generates a “Back” button on each page, so that the user can easily navigate back and forth between the different pages.

III. Challenges

Our greatest challenge during the development of this app was being able to successfully access and retrieve data from the backend to the frontend. The intent was to use RecyclerView to create a page that would display only the recipe requested from the backend in the breakfast, lunch and dinner categories so that we would not need to manually create an activity for each recipe. However, we were unsure if there was an issue with our network connection using the Retrofit client because attempts at requesting and returning data from the backend resulted in the app either returning to the previous activity or crashing entirely. As of now, our app only exists with dummy data on the frontend to demonstrate its intended purpose with both the frontend and backend working, but separately.