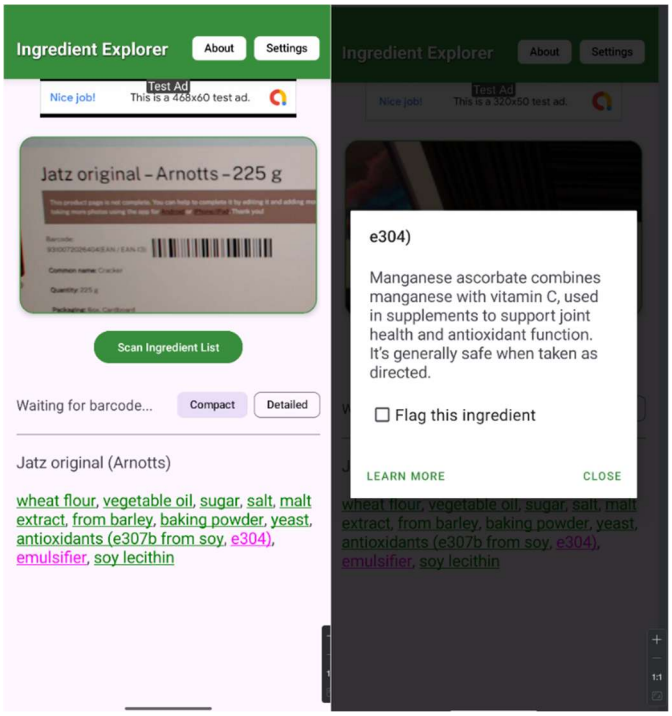
Artifact Three Narrative – E Numbers

My artifact for this enhancement is upgrading the functionality of the database in my android app, Ingredient Explorer. I started working on the app a couple months ago, using the UI design created during CS-319, and the Android Studio experience gained in CS-360. It has food ingredients with their associated descriptions and classifications in .json files, so when a user clicks an ingredient in the app it will popup it's description from the database. The food ingredients list comes from the user scanning a barcode with their phone camera. I've been using upc_ingredients.json for the ingredients and descriptions, and ingredient_aliases.json to store alternate name lists for ingredients to avoid duplicate entries in the main list. The update to the system is adding support for E numbers (i.e. "E213" corresponds to "Calcium benzonate") which are sometimes in ingredient lists especially from Europe and correspond to food additives, so users are able to view ingredient descriptions for ingredients using the European documenting convention where the name isn't plainly listed.

In-app incorrect description retrieved before enhancement:



I chose this artifact because it demonstrates my experience with maintaining a database using real-world data, and adapting it to meet new requirements. To update the database, I first did online research to find the most complete list of E numbers and their corresponding ingredients. I reformatted the data and added it to the aliases list, then I tested the app by scanning some products from "https://world.openfoodfacts.org/facets/countries/australia" which has product listings with barcodes that I can scan from a country that uses E numbers. During testing I noticed that around 70% of the codes would fetch the correct descriptions, so to get full coverage I needed to check that every new entry had a corresponding entry in upc_ingredients.json.

That was a challenge because the app matches ingredients to their closest database entry to accommodate spelling variations, so finding missing entries isn't as simple as searching

specific names. To address that, I made a python script and copied the ingredient retrieval logic from my apps IngredientDataLoader class, which uses approximate matching to find ingredients that aren't exact matches. That ensured the way the data was handled would match the app's behavior. The python script ingredient_checker.py retrieves only the ingredients that weren't found with the approximate matching and saves them in an output file. That let me retrieve the ~50 missing entries out of the ~350 total, and from there all I needed was to add entries for those in upc_ingredients.json to achieve full E number coverage.

Updated version retrieves correct ingredient description from E number: