Final Project Report

Divya Venkatraman

April 2021

1 READ ME

1.1 Software Needed

- Python 3 compiler with the following libraries installed: `pymysql`, `os`, `string`, and `decimal`.
- Packages can be installed from the shell using `pip`: `python3 -m pip install package_name`
- `pip` can be installed from shell using `apt install python3-pip`
- MYSQL Server + Workbench

All the included python files must be in the interpreters' `$PYTHONPATH`. As long as they are all in the same folder as the script you are running (`PySQLConnection.py`) this should occur automatically.

1.2 Creating the Database

In the folder `create_schema`, you will find two `.sql` files; one a dump file containing the USDA Nutritional Database (`usda_national_nutrients.sql`) and the other a script (`final_project.sql`) that creates the database for our project - `meal_plan`.

- First, import `usda_national_nutrients.sql`, setting the default target schema to `usda`. The name is very important - the script relies on correct nomenclature to import data from the resultant schema.
- Second, open (`final_project.sql`) and run it. Remember, every time you run this script, the database `meal_plan` will reset and be filled with the default data, reversing any changes our program may make.

1.3 Running the Program

Run (`PySQLConnection.py`) from a shell using the command `python PySQLConnection.py`. Alternatively, open and run the script in a Python IDE such as Spyder or PyCharm.

The script must run on the same machine as your SQL server; virtual machines will not be able to connect to the MySQL Server.

1.4 Link to Video

https://youtu.be/kOC1QuUH_UU

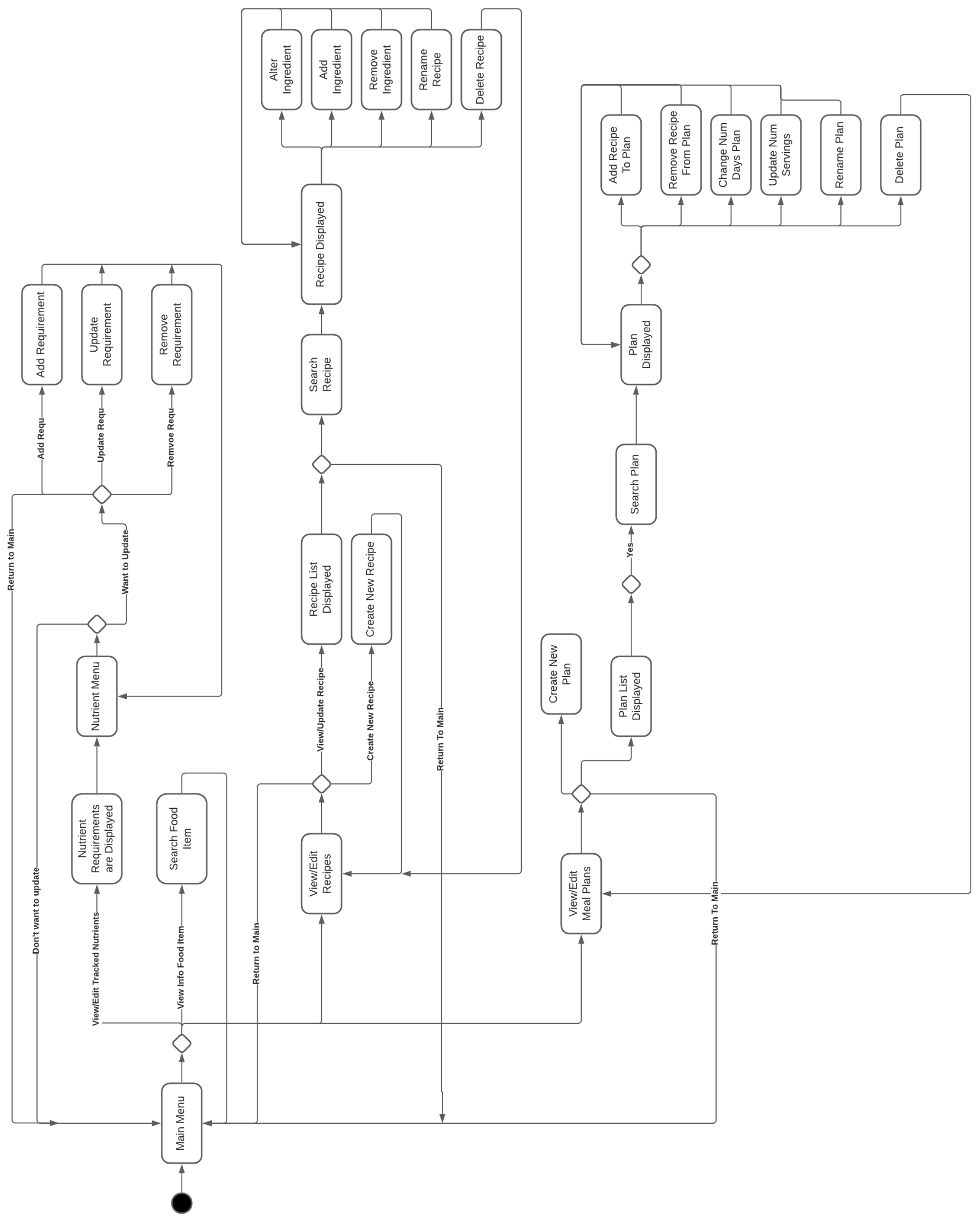
2 Program Description

Meal planning is deciding exactly what and how much you're going to cook over a certain time period. You buy exactly the groceries you need and the chore of cooking goes on autopilot and therefore becomes a lot easier. It's a very underutilized skill - it saves time and money, reduces food waste, and nutritionists testify that it helps people with portion control and eating healthier. But it's difficult to do, especially if you want to create nutritionally complete meal plans. Finding and calculating the nutritional value of a single ingredent, much less a recipe, is incredibly complicated.

This is a meal planning calculator that uses information from the most reliable and complete nutritional database out there, compiled by the USDA. You can choose what nutrient requirements you want to track, enter and save recipes, and test out meal plans. Search for nutrients, food items, recipes, and plans using either name or ID - and no need to be too careful. The program automatically detects what kind of input you've entered and does its best with it, showing you a list of closest matches if you've tried out a name, searching by ID if your input is all numerical, or simply asking you to enter again. All data validation is done in input parsing. For the nutrients you wish to track, the program automatically calculates the nutritional value of any recipe, and tells you if a meal plan over a certain number of days meets your desired daily average nutrient intake. Upon ending the program, all changes are committed to the database. The UI is a simple recursive menu interface. Simply run (`PySQLConnection.py`) and follow the prompts.

3 Action Flow

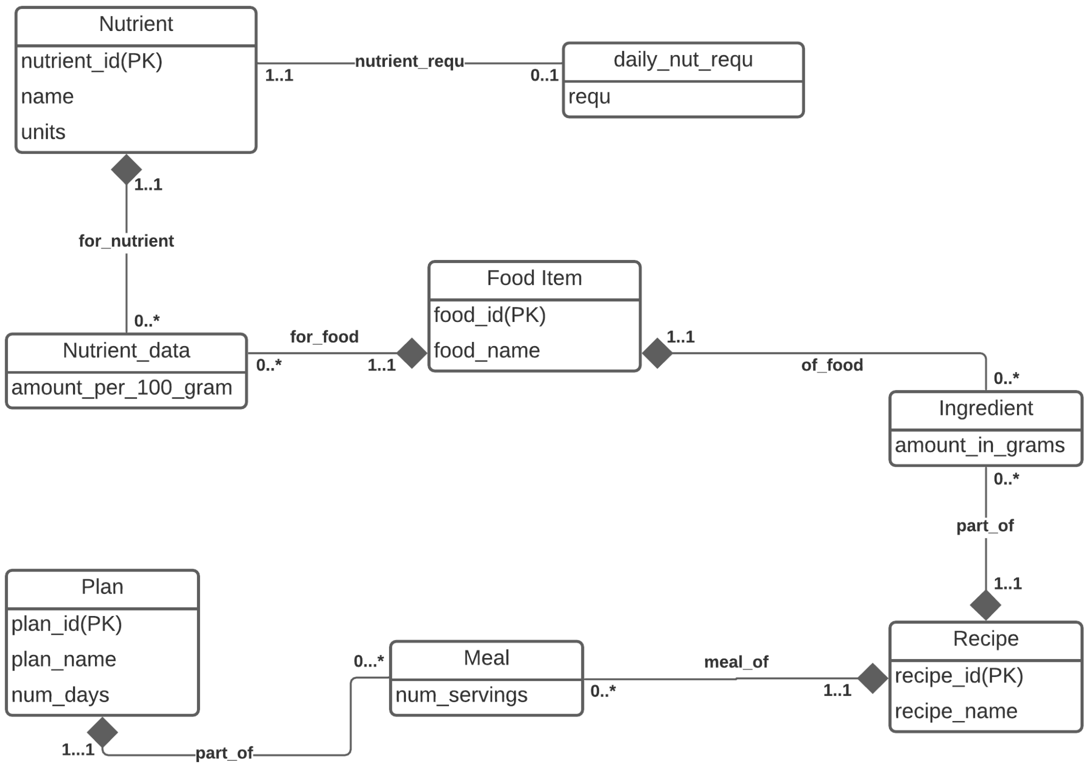
Note: this is only a general action flow. Each "state" is a function the user uses to interact with the program, though it is called automatically. Each of these functions have flows and recursions of their own in response to the results of other functions and the state of the database, but including them would make the graph too intricate to be readable. Their functionalities are very obvious when using the program.



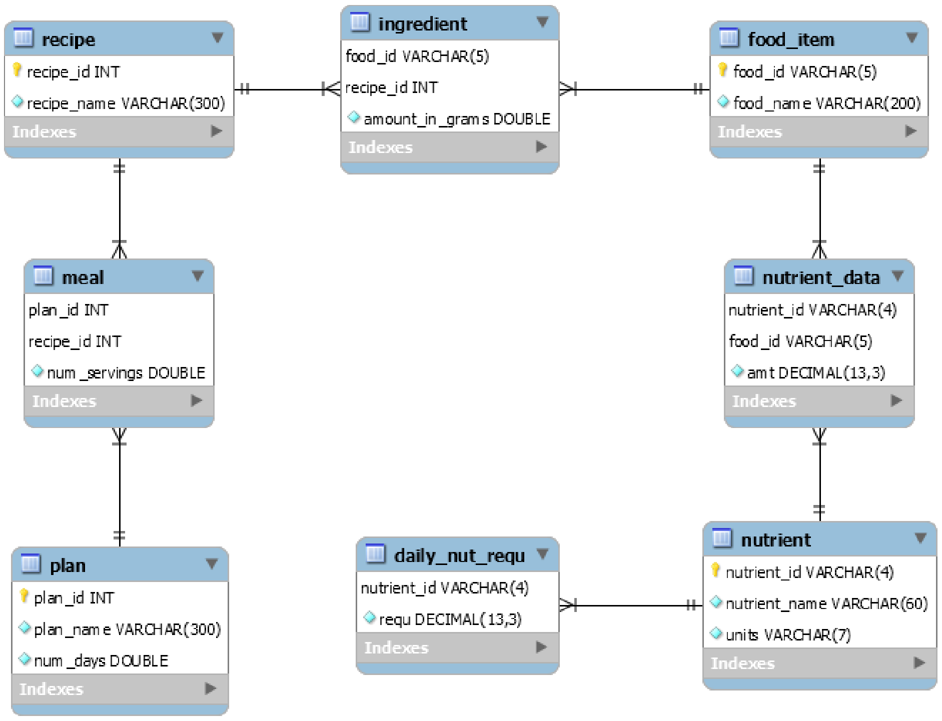
Main Menu
View/Edit Tracked Nutrients
View Info Food Item
Nutrient Requirements are Displayed
Search Food Item
Nutrient Menu
Don't want to update
Want to Update
Return to Main
Add Requ
Update Requ
Remvoe Requ
Add Requirement
Update Requirement
Remove Requirement
View/Edit Recipes
Return to Main
View/Update Recipe
Create New Recipe
Recipe List Displayed
Create New Recipe
Return To Main
Search Recipe
Recipe Displayed
Alter Ingredient
Add Ingredient
Remove Ingredient
Rename Recipe
Delete Recipe
View/Edit Meal Plans
Return To Main
Create New Plan
Plan List Displayed
Yes
Search Plan
Plan Displayed
Add Recipe To Plan
Remove Recipe From Plan
Change Num Days Plan
Update Num Servings
Rename Plan
Delete Plan

4 UML

Tables `food_item`, `nutrient_data`, and `nutrient` all contain data taken directly from the USDA database. The program does not allow this data to be deleted or changed. The user can, however, perform CRUD operations on the daily nutritional requirements, recipes, and plans. A recipe can have many food items and a food item can be part of many recipes, and the relation `ingredient` represents that relationship. Similarly, the relation meal functions as the logical representation of what is really the many to many relationship between recipes and plans.



5 Logical Design



6 Lessons Learned

Over the course of the design process, I learned to protect against duplicate entry errors, null pointer errors, division by zero, and circular imports, among many other errors. I also learned the value of making your code as modular as possible: if you find yourself writing coding a certain logic twice, always outsource it to another function. Modularizing in this way opens your eyes to symmetries in your code that provide opportunities for heightened efficiency, and also makes otherwise near-impossible debugging a breeze. It's also good to isolate all functions that can possibly return Null objects and keep track of their function calls, so catching Null pointer errors is a matter of prevention rather than arduous testing

I also realized that it's important to do data validation and comment thoroughly at the very beginning. It is easy to do so as you go but very hard to do at the end. Projects like this take more time than expected, always. Even if there are no unexpected hiccups and all the concepts are clear, code almost never comes out perfectly on the first try, and testing and debugging are difficult to account for in terms of time. It is important to always allot more time than one is inclined to estimate..

As for technical expertise, I gained a great deal of expertise working with with Python-SQL connectors and procedures and deepened my understanding of Python considerably.

7 Alternate Design Considerations

I decided to use SQL as the base for the project, which I think was the correct design choice considering that it was originally intended for single person use. However, while the nutritional data from the USDA is best suited to SQL, if this was expanded to commercial use, a No-SQL database would be necessary for at least part of the project: storing all individuals' individual recipes and data.

8 Future Work

The database has extensive functionality as a meal-planning aid. It takes all the work out of calculating nutritional completeness, a task which would be almost impossibly laborious to do by hand. The biggest shortcoming of this project currently is its unsophisticated user interface. I would like to extend it to a graphical user interface, and possibly extend the program to data to seamlessly create an easily accessible PDF of all recipes, complete with nutritional info, perhaps using a text formatter like Latex. I would also like a way to use recipes as ingredients for other recipes, because that happens quite often in a real kitchen.

9 Errors in Code

The program itself works fine, but I was unable to create a data dump of my schema. Somewhere in the process of exporting and re-importing, the dump randomly lost large chunks of data. Potential causes could have been the volume of data, a conflicting configuration of SQL Workbench, or very possibly just a glitch. After extensive but unsuccessful attempts to diagnose the issue, I submitted the necessary root schema and .sql file needed to construct my database from scratch.