Recipe Tracker Project Report

Katriel Coyoca, Li Wang, Vinhem Duong

Course: CS 157A section 03

Professor: Aravind Rokkam

Table of Contents:

Introduction

With the high costs of restaurants and food delivery services, many people prefer to cook their meals. However, it can be difficult to keep track of all your recipes. You may want to search for a certain recipe based on how long it will take to cook, or by what ingredients are available in your fridge, or whether you have an air fryer available to make something. A recipe tracking app will enable you to easily plan meals by offering a convenient place to save, view, and search for recipes.

Objective

The objective is to develop a recipe tracker desktop application that stores recipes, recipe category, ingredients, quantity of ingredients, necessary appliances (such as an oven or stove), preparation time, cooking time, and the instructions of the recipe. The user will be able to input a recipe by entering the name, ingredients used, appliances used, and instructions. The user has the choice to optionally save an image of the dish, the prep time, cook time, serving size, and to save the recipe under a category which the user has previously defined. A category is used instead of cuisine type since a category is more general, so users can save a variety of categories such as those named after cuisine type (Ex. Mexican, Chinese, Italian), after diet types (Ex. vegetarian, vegan, gluten-free), or even meal-type (Ex. breakfast, lunch, dinner). Additionally, ingredients areThe recipe and its corresponding information will be stored in the app's database. The user will also be able to view a search interface containing all saved recipes, and search for a recipe by either typing the recipe name, or filtering results based on the category, ingredients, appliances, or total time involved. The application will be developed with Python using Tkinter for the GUI and MySQL for the database.

Additions/changes to original design:

- Category instead of cuisine-type
- Serving size, description, imagepath, units
- Ingredient categories

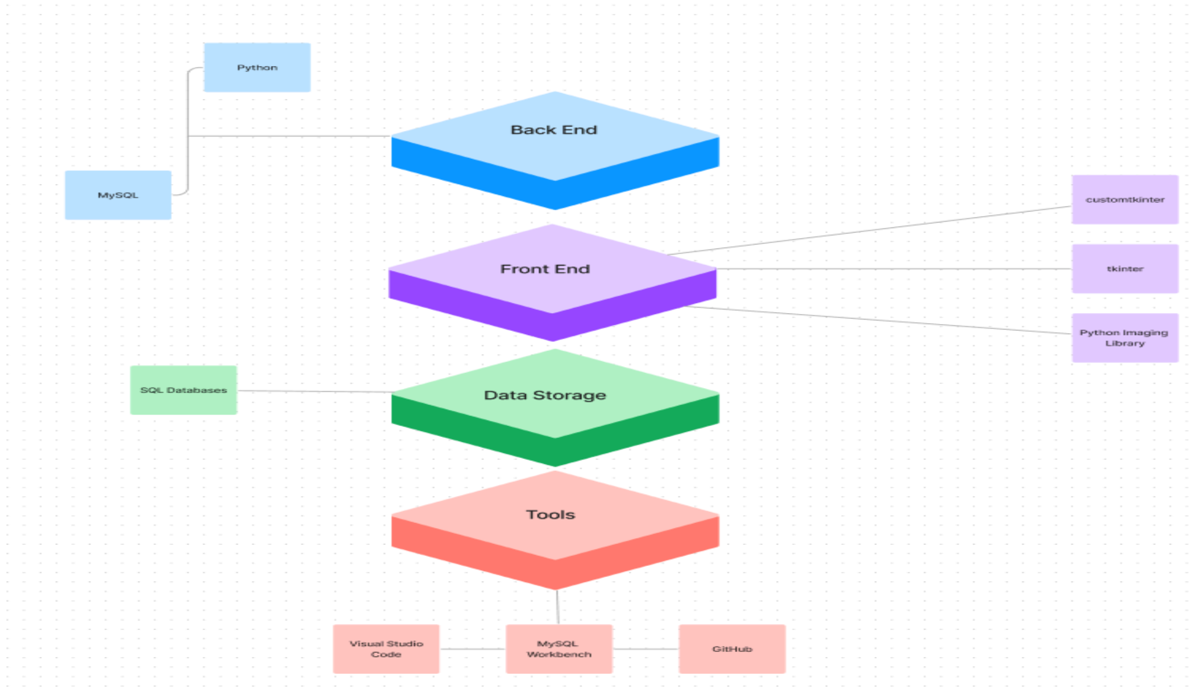
Features:

1. Add Category
2. Add Ingredient
3. Add Recipe
4. Search by name

5. Search by filter
    4a. Filter by Category
    4b. Filter by Ingredients
    4c. Filter by Appliances
    4d. Filter by Duration

High-level Design

Tech Stack:



Tkinter and CustomTkinter, a python GUI framework and library were used to create the UI of the application. A MySQL database is created in the user desktop using a script file and connected to through the app using MySQL Connector in Python. The rest of the back-end functionality is in python, where methods contain queries connected to certain functions, and the cursors connect to the database to carry out the queries.

Database Design

Normalization process

First Normal Form:

Recipe

- (**recipeID (P)**, name, preptime, cooktime, description, instruction, servingsize, imagepath, **categoryID (P)**, cName, **ingID (P)**, ingName, ingtypeID, ingtypeName, **applianceID (P)**, aName, unitID, uName, quantity)

<u>Functional Dependencies:</u>

recipeID -> name, preptime, cooktime, description, instruction, servingsize, imagepath	(partial)
categoryID -> cName	(partial)
ingID -> ingtypeID, ingName	(partial)
ingtypeID -> ingtypeName	(transitive)
recipeID, ingID -> unitID, quantity	(partial)
unitID -> uName	(transitive)
applianceID -> aName	(partial)

There are no multivalued attributes so the cells are atomic, however there may be a multivalued dependency for categoryID, ingreID, and applianceID. There is a composite primary key (recipeID, categoryID, ingreID, and applianceID). There are no repeating rows.

Second Normal Form:

Recipe

- (**recipeID (P)**, name, preptime, cooktime, description, instruction, servingsize, imagepath)

RecipeIngredients

- (**recipeID (P/F), ingID (P/F)**, unitID (F), uName, quantity)

RecipeAppliances

- (**recipeID (P/F), applianceID (P/F))**

RecipeCategories

- (**recipeID (P/F), categoryID (P/F))**

Ingredients

- **(ingID (P)**, ingName, ingtypeID (F), ingtypeName)

Appliance

- (**applianceID (P)**, aName)

Category

- (**categoryID (P)**, cName)

All previous partial dependencies were removed and placed in their own tables, so now all derivable attributes are fully dependent on the primary key of their table.

Third Normal Form:

Recipe

- (**recipeID (P)**, name, preptime, cooktime, description, instruction, servingsize, imagepath)

RecipeIngredients

- (**recipeID (P/F), ingID (P/F)**, unitID (F), quantity)

RecipeAppliances

- (**recipeID (P/F), applianceID (P/F)**)

RecipeCategories

- (**recipeID (P/F), categoryID (P/F)**)

Ingredients

- (**ingID (P)**, ingtypeID (F), ingName)

IngredientType

- (**ingtypeID (P)**, ingtypeName)

Appliance

- (**applianceID (P)**, aName)

Category

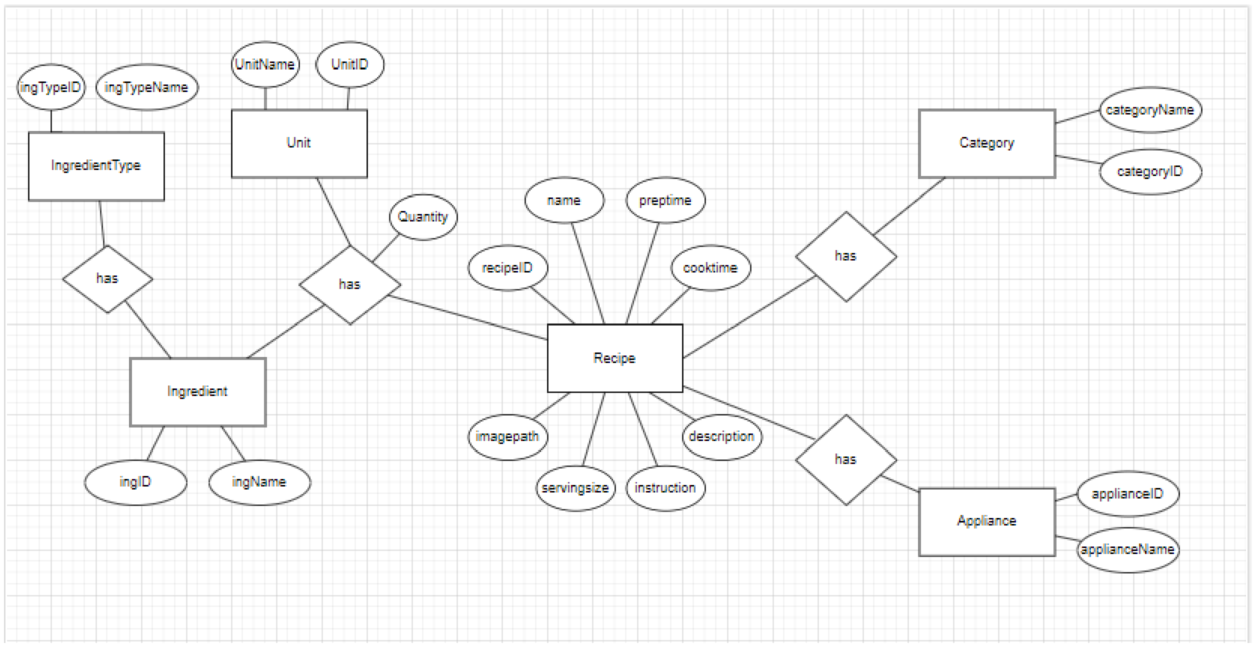
- (**categoryID (P)**, cName)

Unit

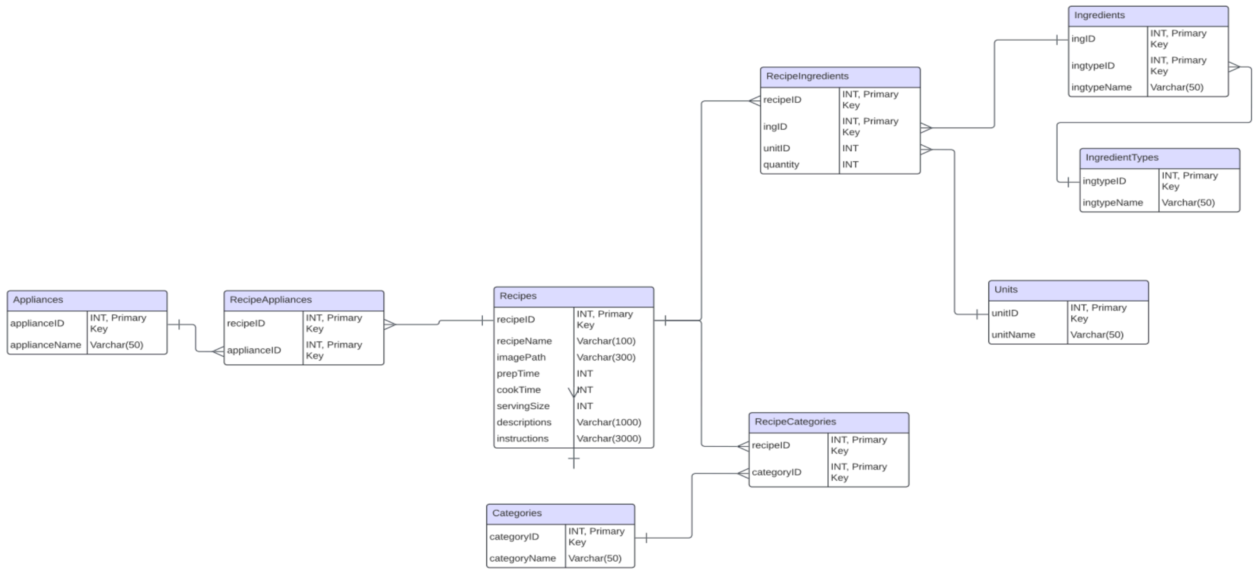
- (**unitID (P)**, uName)

All transitive dependencies were removed and placed in their own tables.
In this case, the schema that is in 3rd normal form also follows BCNF.

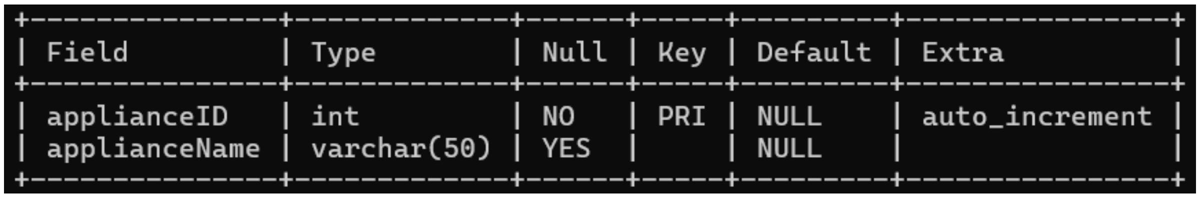
ER Diagram:



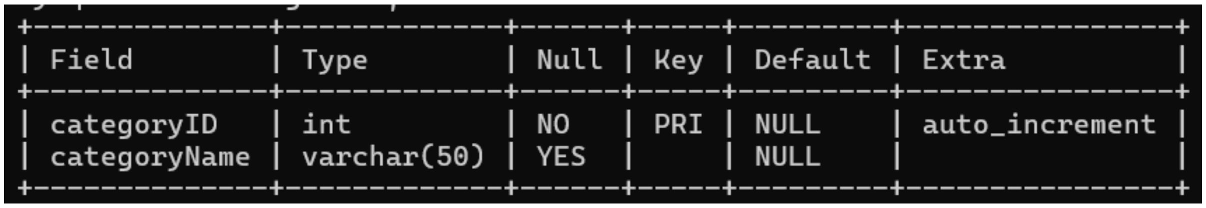
Relational Model:



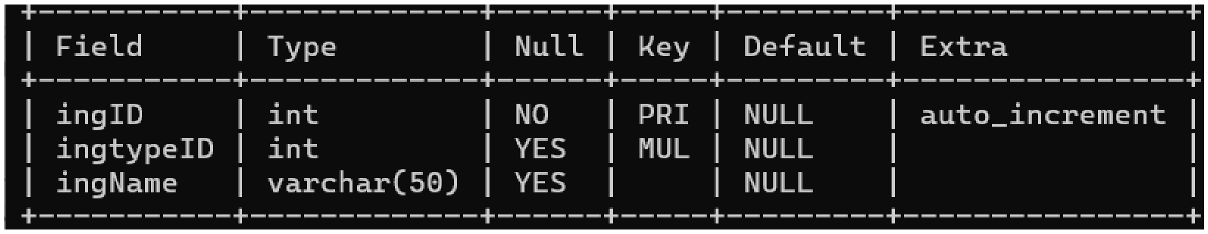
Final Tables



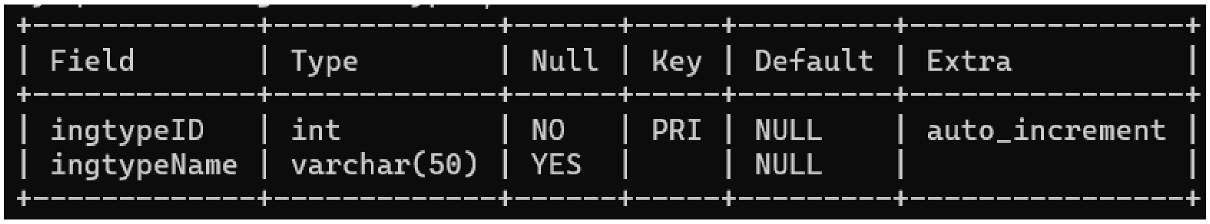
```
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| applianceID   | int         | NO   | PRI | NULL    | auto_increment |
| applianceName | varchar(50) | YES  |     | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
```



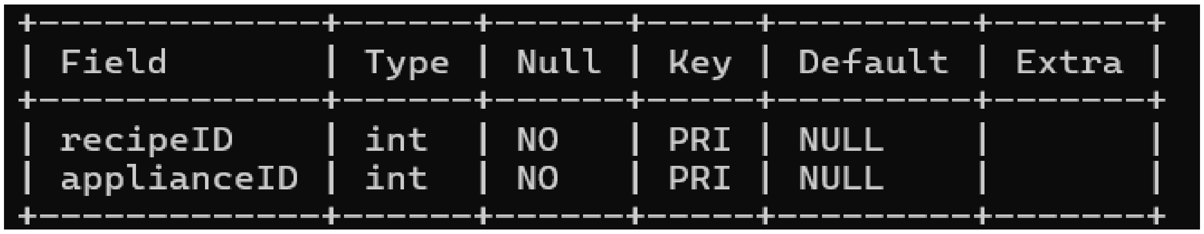
```
+--------------+-------------+------+-----+---------+----------------+
| Field        | Type        | Null | Key | Default | Extra          |
+--------------+-------------+------+-----+---------+----------------+
| categoryID   | int         | NO   | PRI | NULL    | auto_increment |
| categoryName | varchar(50) | YES  |     | NULL    |                |
+--------------+-------------+------+-----+---------+----------------+
```



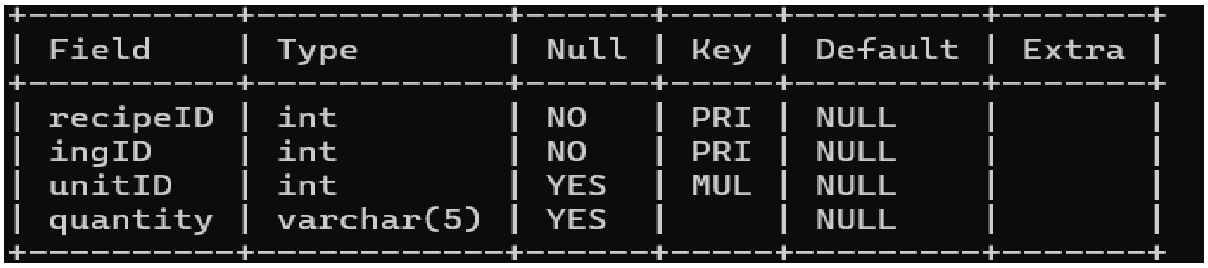
```
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| ingID      | int         | NO   | PRI | NULL    | auto_increment |
| ingtypeID  | int         | YES  | MUL | NULL    |                |
| ingName    | varchar(50) | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
```



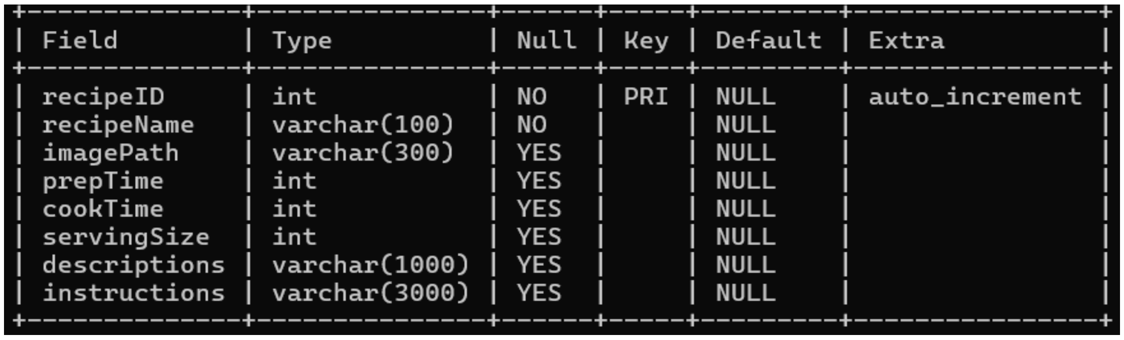
```
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| ingtypeID   | int         | NO   | PRI | NULL    | auto_increment |
| ingtypeName | varchar(50) | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
```



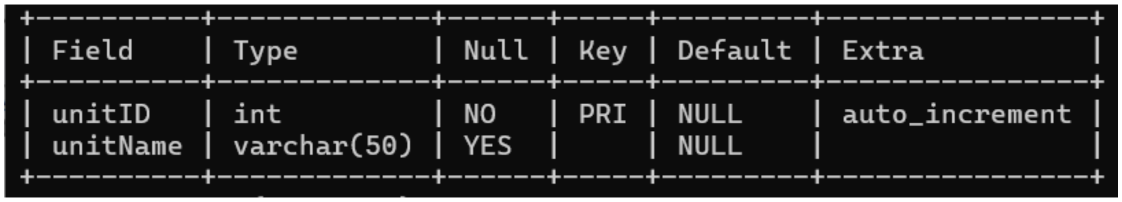
```
+-------------+------+------+-----+---------+-------+
| Field       | Type | Null | Key | Default | Extra |
+-------------+------+------+-----+---------+-------+
| recipeID    | int  | NO   | PRI | NULL    |       |
| applianceID | int  | NO   | PRI | NULL    |       |
+-------------+------+------+-----+---------+-------+
```



```
+----------+------------+------+-----+---------+-------+
| Field    | Type       | Null | Key | Default | Extra |
+----------+------------+------+-----+---------+-------+
| recipeID | int        | NO   | PRI | NULL    |       |
| ingID    | int        | NO   | PRI | NULL    |       |
| unitID   | int        | YES  | MUL | NULL    |       |
| quantity | varchar(5) | YES  |     | NULL    |       |
+----------+------------+------+-----+---------+-------+
```



```
+--------------+---------------+------+-----+---------+----------------+
| Field        | Type          | Null | Key | Default | Extra          |
+--------------+---------------+------+-----+---------+----------------+
| recipeID     | int           | NO   | PRI | NULL    | auto_increment |
| recipeName   | varchar(100)  | NO   |     | NULL    |                |
| imagePath    | varchar(300)  | YES  |     | NULL    |                |
| prepTime     | int           | YES  |     | NULL    |                |
| cookTime     | int           | YES  |     | NULL    |                |
| servingSize  | int           | YES  |     | NULL    |                |
| descriptions | varchar(1000) | YES  |     | NULL    |                |
| instructions | varchar(3000) | YES  |     | NULL    |                |
+--------------+---------------+------+-----+---------+----------------+
```



```
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| unitID   | int         | NO   | PRI | NULL    | auto_increment |
| unitName | varchar(50) | YES  |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
```

Results

We were able to implement all of the planned features described in the project proposal.
The majority of the bugs we came across were related to the “add recipe” functionality. There were some

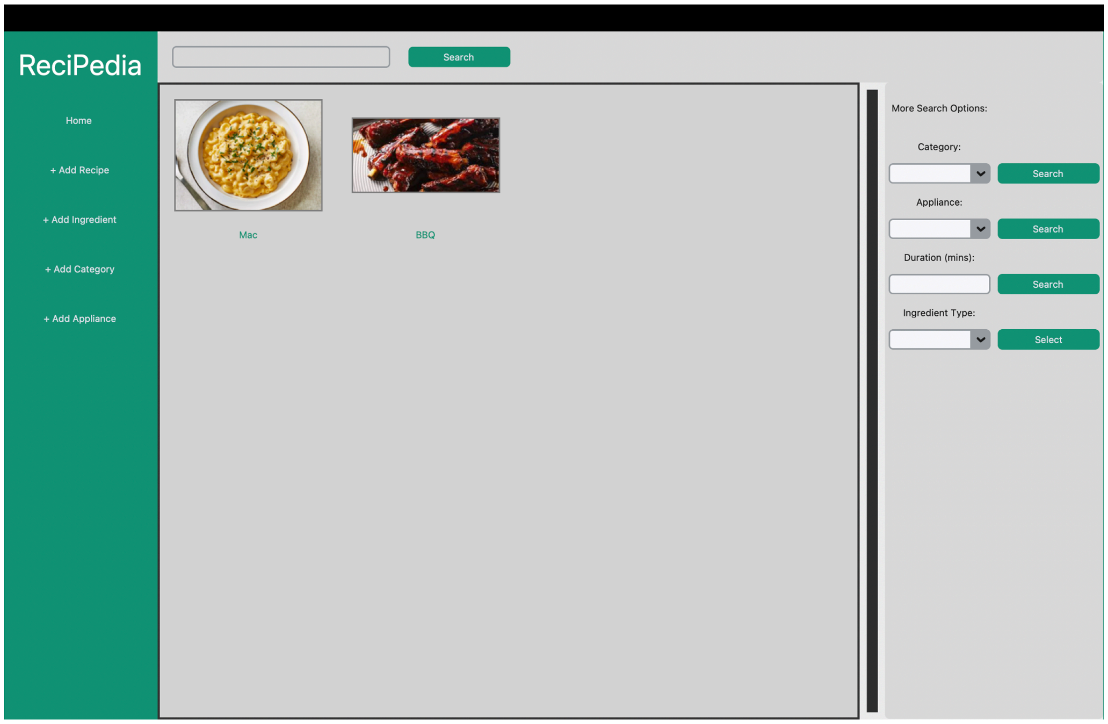
Through the project, we made some observations about our choice of tech stack. When the project team was formed and the tech stack was chosen, we thought our choice of a basic tech stack would help us complete the project more easily than if we had chosen more technologies and created a web-application. Because our group was largely unfamiliar with the process of actually creating software, we decided to go with the language we all commonly knew, and use Tkinter since one of us was somewhat familiar with it.
However, through the process of the project, we found that our simple tech stack actually meant that we had to create many functionalities ourselves when it related to the back-end. For example, packing and formatting each individual component, frame, and widget onto the home page was a more arduous task than anticipated. “Form handling,” such as inputting fields to add a recipe, ingredient, etc., was also difficult, whereas if we had created a web application, the form handling would have been more lightweight and had existing libraries that we could have used to help us out.
Another factor was, since this was the group’s collective first experience with creating an application, it was a learning process when writing the code and in finding the most concise, least bug-prone way to write the code. As a result the code is not as concise as it could be.

As a result, a lot of our code was more buggy than anticipated, especially the “Add Recipe” page, which we discovered when making improvements to the GUI. This was also a result of not testing functionality correctly.

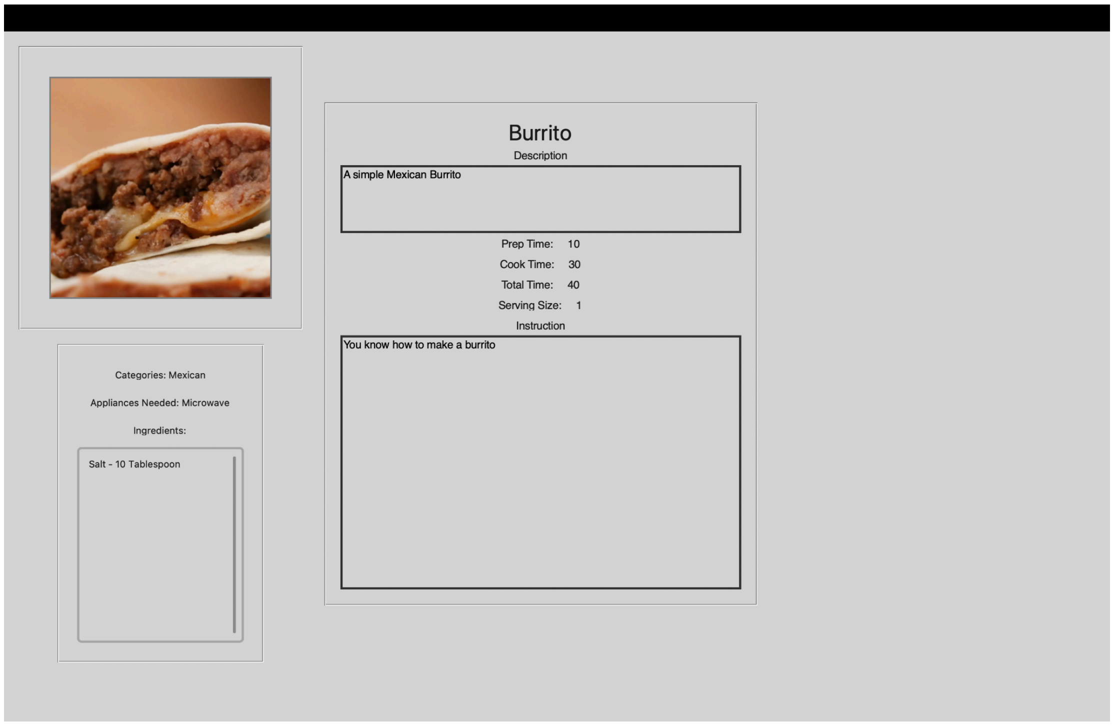
In conclusion, all features were implemented, and all discovered bugs were corrected.

Final Product

All recipes page (Search functionalities)



Individual recipe page:



Burrito

Description

A simple Mexican Burrito

Prep Time: 10
Cook Time: 30
Total Time: 40
Serving Size: 1

Instruction

You know how to make a burrito

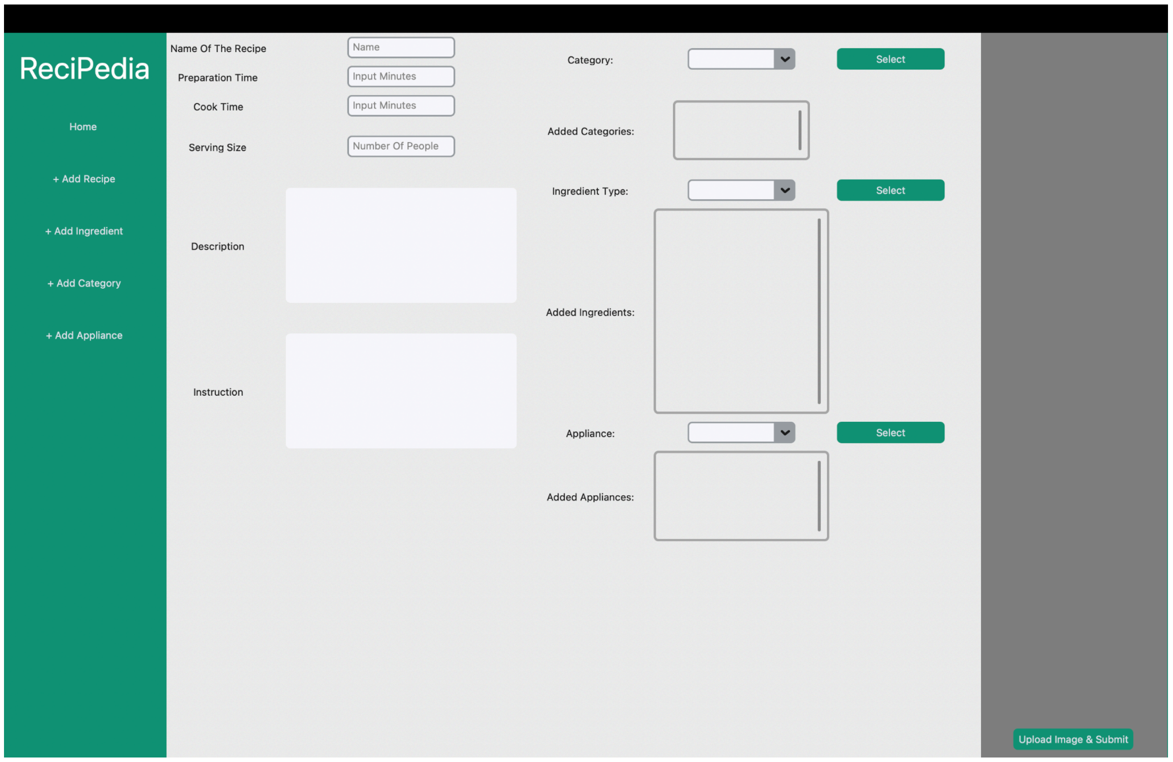
Categories: Mexican

Appliances Needed: Microwave

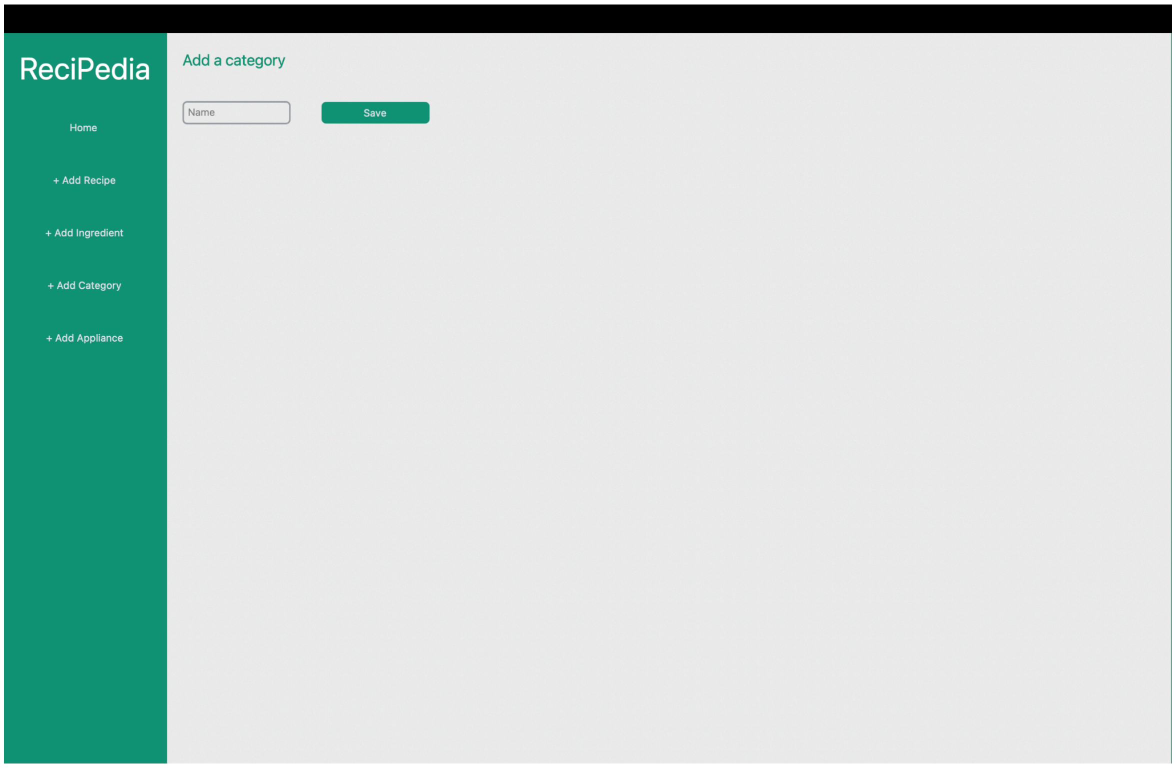
Ingredients:

Salt - 10 Tablespoon

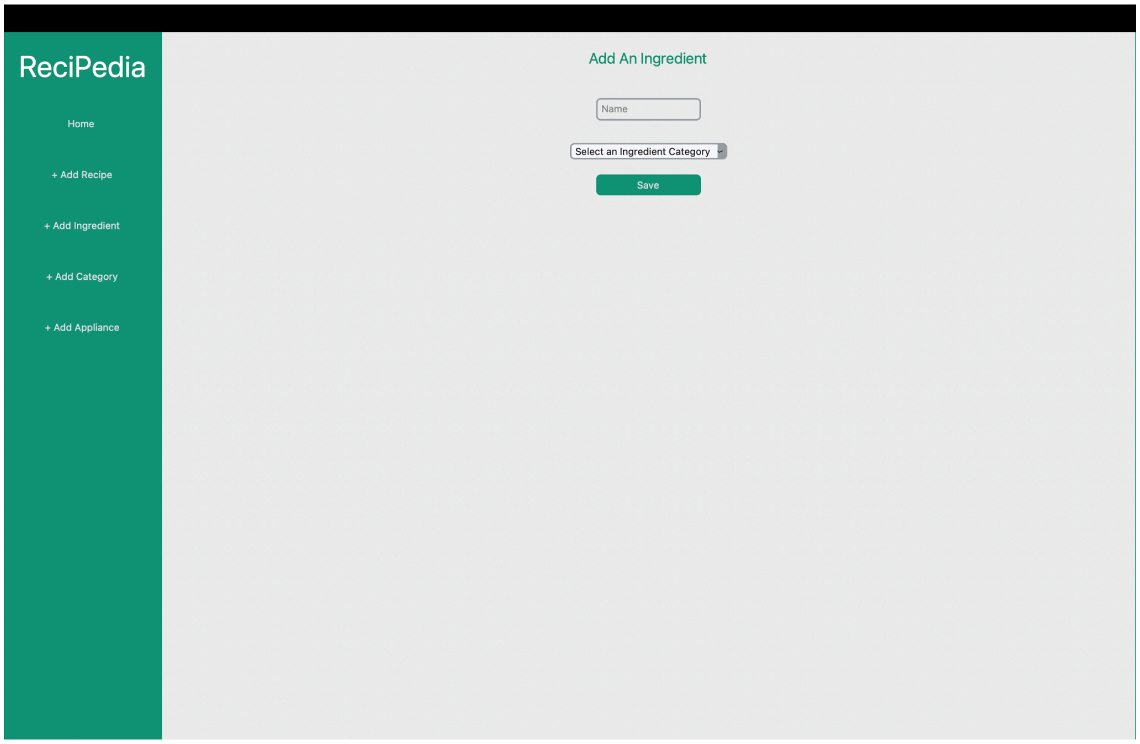
Add recipe page:



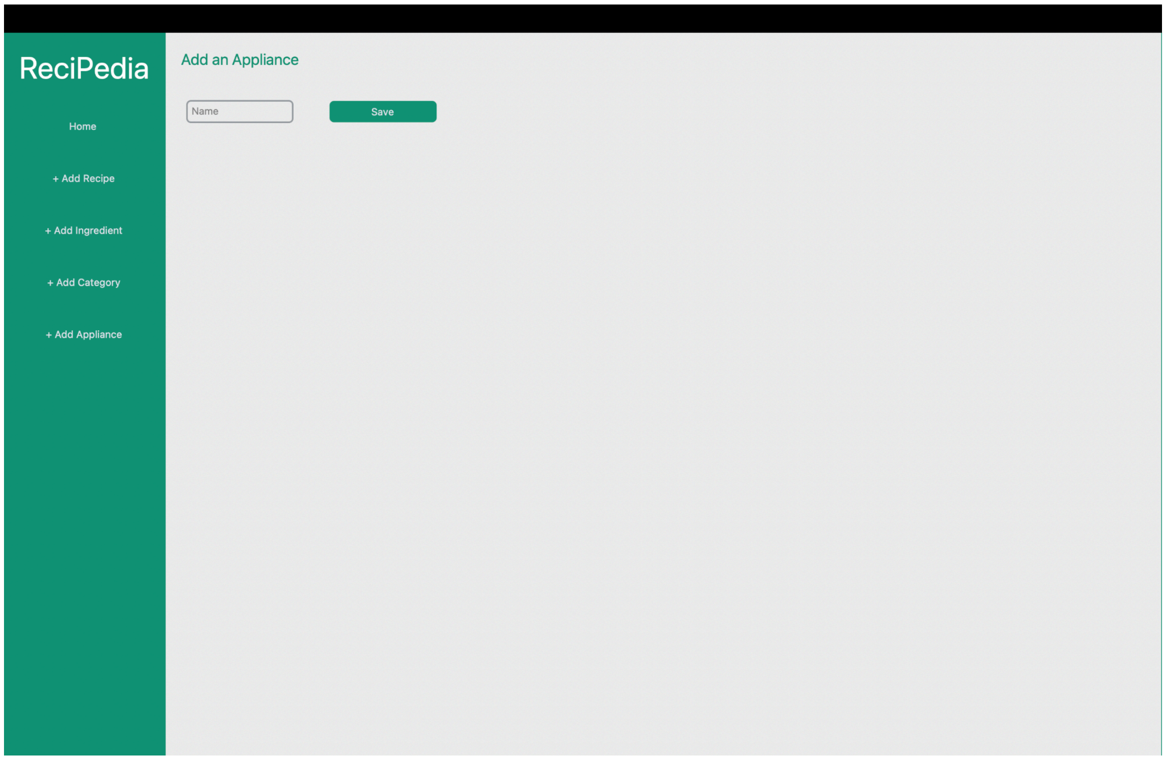
Add category page:



Add ingredient page:



Add Appliance Page:



Contribution:
Katriel:

- Mostly worked on back-end functionality.
- Connection with the DB, wrote majority of the sql script to create the database tables, did database normalization
- Features worked on:
  - All functionality to add categories, ingredients, and appliance individually and within a recipe
  - Display the categories, ingredients, and appliances on a recipe page
  - All search features

Li:

- I worked on the homepage, which displays all recipes. I also developed the add recipe page, created the Recipes table, and connected it to the database. I fetched data from the database, and for displaying individual recipe pages, I primarily worked on the front-end.
- Helped with database normalization

Vinhem:

- I worked on the front_end. I built the frames for the main window, which were the base of our program. I built the scrollable frame, which helped us add our images, without affecting our search bar and our window.
- I also worked on the GUI, making sure it looked better for the final product
- For the add recipe page, I built new frames, so adding items and using it will be much easier.
- I also assisted on back - end development. I noticed bugs and built functions that tried to stop those bugs. I also helped on search functionalities.
- Helped with database normalization

References

https://www.youtube.com/watch?v=yQSEXcf6s2I&list=PLCC34OHNcOtoC6GglhF3ncJ5rLwQrLGnV