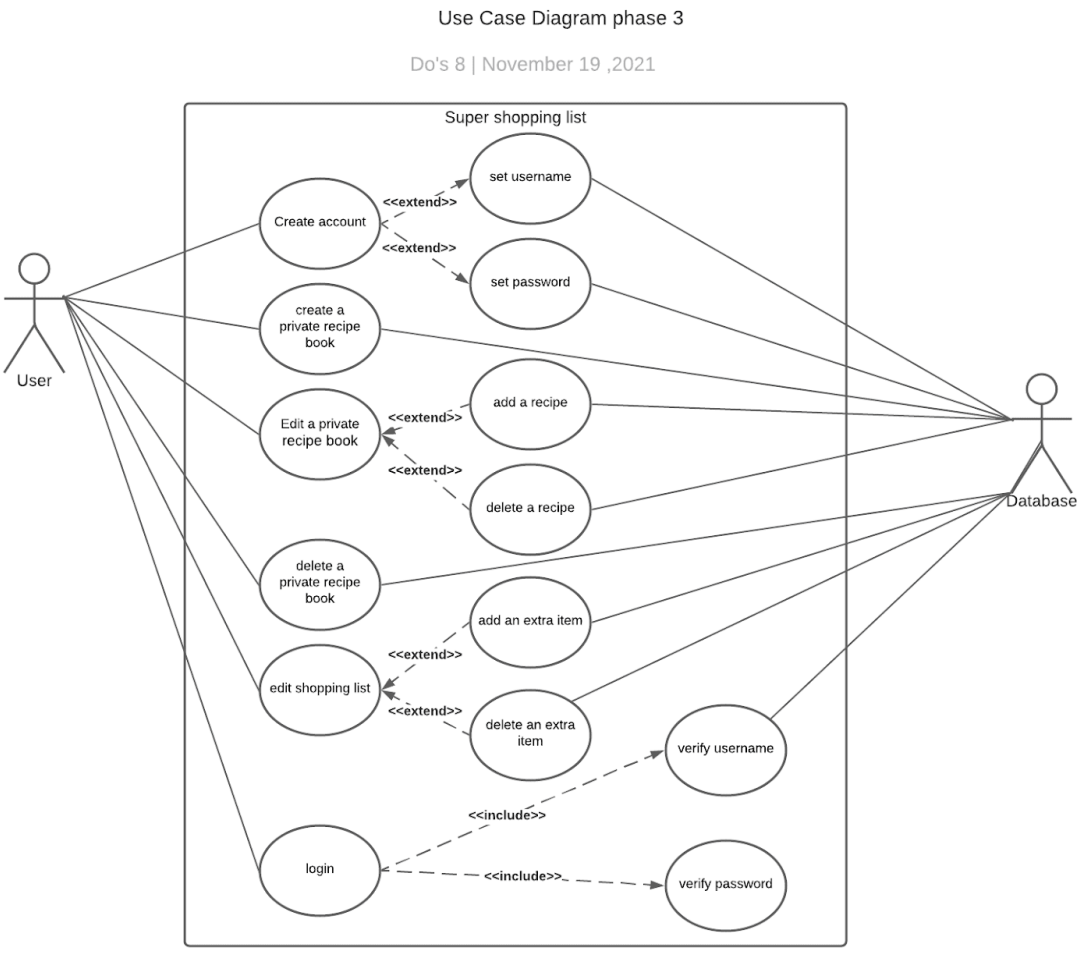
Deliverable 5 Report

Requirements

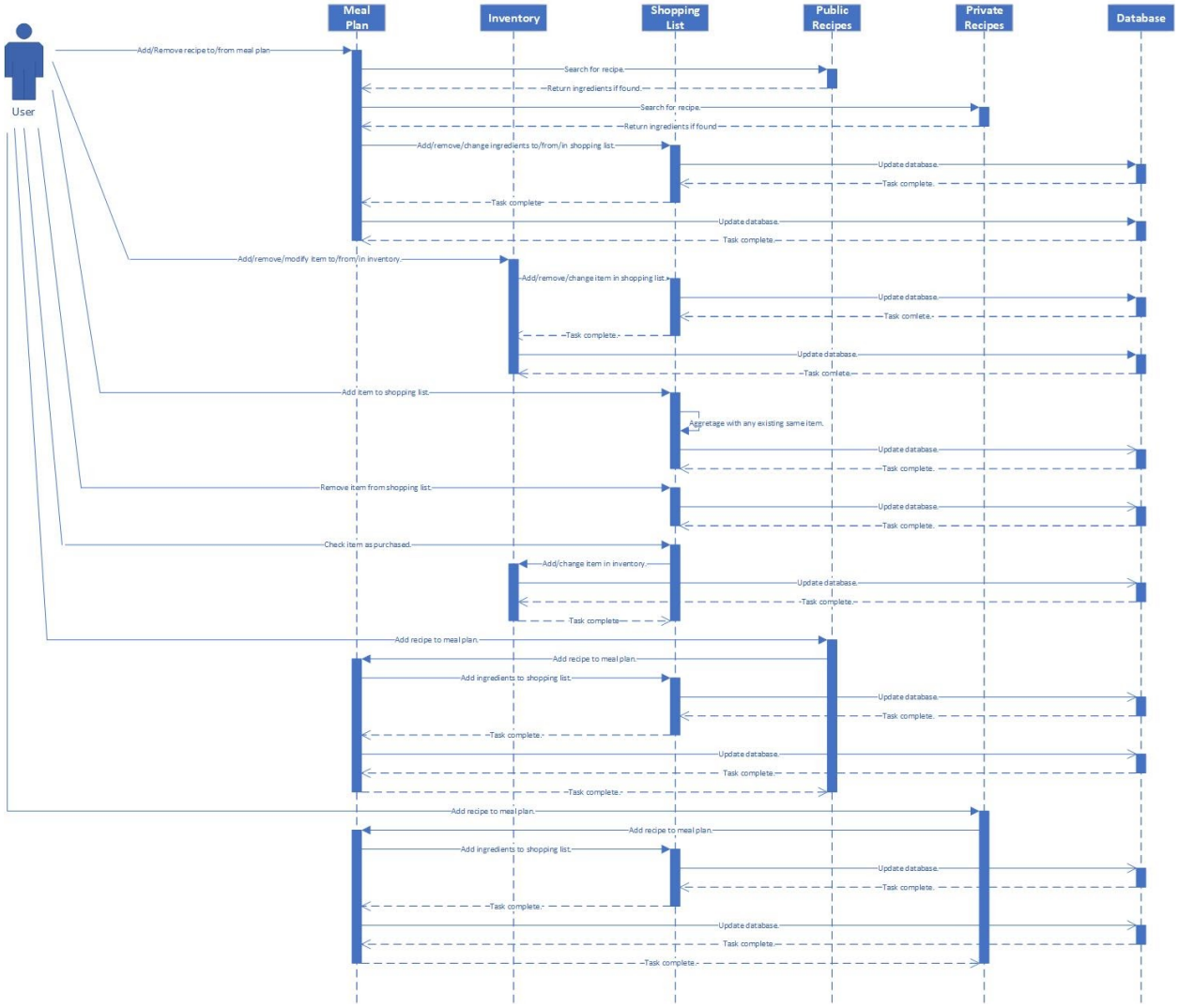
- User create recipe books
    - The user can create custom recipe books for recipe organization.
- User can delete recipe books
    - The user can delete any of their custom recipe books.
- User can view created recipe books
    - The user can view any of their recipe books. The viewer should function the same as the public and private recipe viewers, without the option to add the recipe to a recipe book.
- User can remove recipes from a created recipe book
    - The user can remove recipes from any of their custom recipe books.
- Add extra items to shopping list
    - The shopping list has a button to add items that are not part of the ingredient list of a selected recipe
- Remove extra items from shopping list
    - Those specific items that are added are able to be deleted.
    - This is different than “checking off” a shopping list item
- Items in the shopping list can be checked off
    - When a user purchases an item they can check the item off which moves it to the inventory to be removed from that later upon the meal being cooked.
- Unit conversion
    - Unit conversion between common types of measurements that are used in cooking.
- Shopping list generation algorithm
    - The shopping list generation algorithm accounts for the extra items as well as the items that are generated from the selected recipes.
    - It also considers potential unit conversions that may exist between items that have the same name.
- User can log in
    - The user can log in, giving them access to their recipes. This unlocks all tabs so the user can access the application.
- User can create account
    - The user can create an account, which also logs in the created account.This unlocks all tabs so the user can access the application.

UML Designs

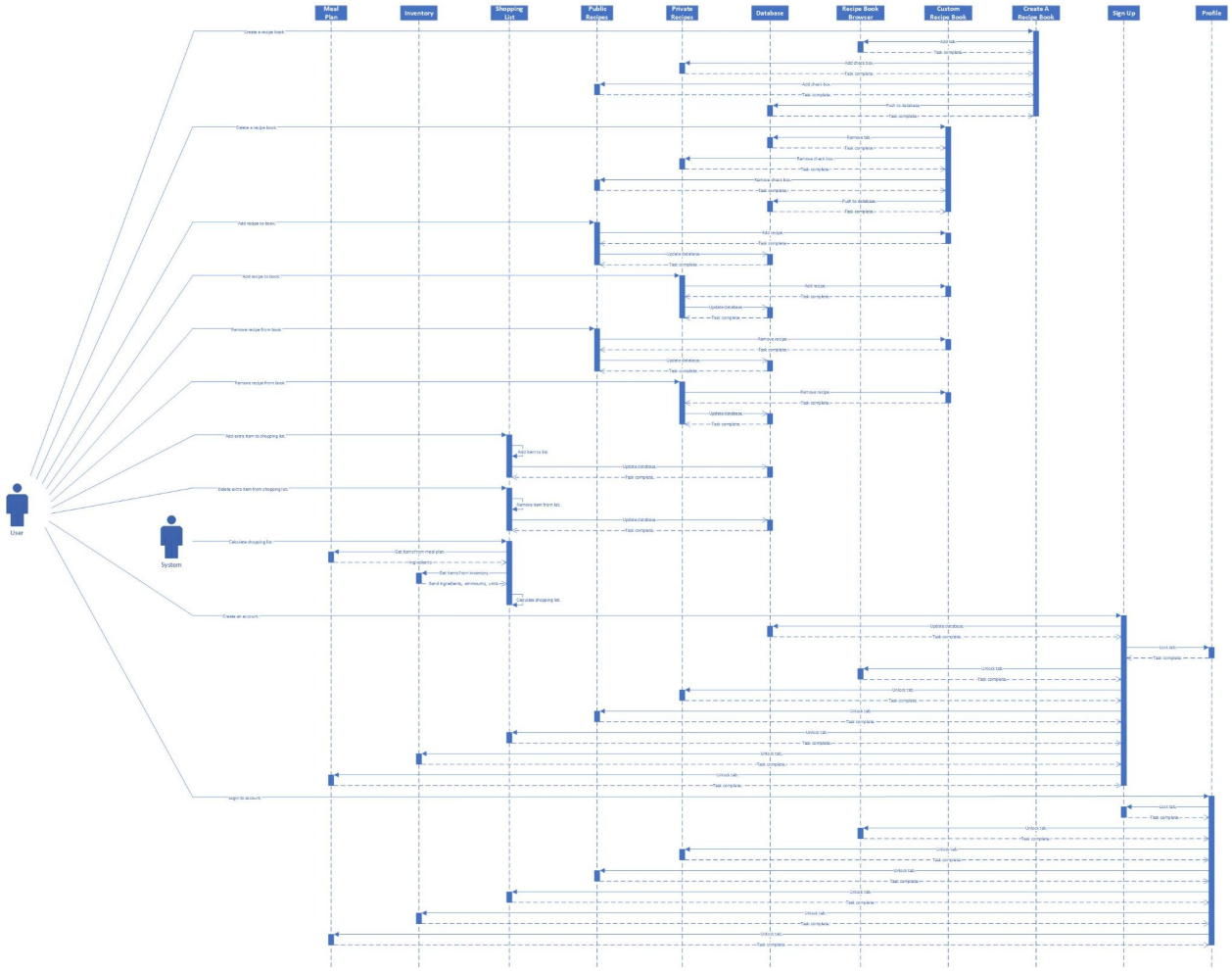
Use Case Diagram



Sequence Diagrams

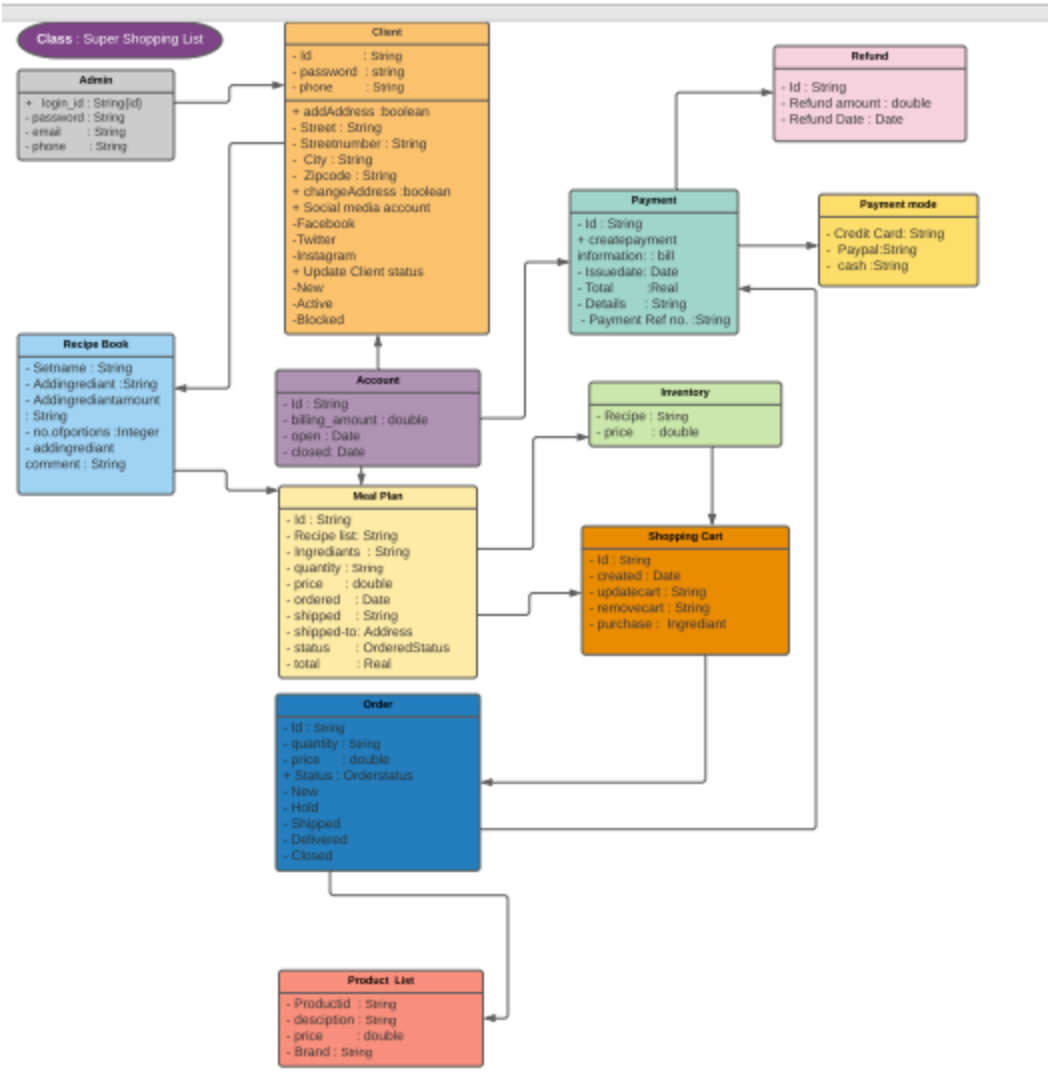


Mica's Sequence Diagram 1



Mica’s Sequence Diagram 2

UML Class Diagram



Test Cases

This phase testing focuses on writing (and passing) unit tests for the database functionality as well as UI functionality. The goal was to get 100% code coverage of the database code so that we know all the queries that are / might be used by the code base are correct.

The UI testing simulates button presses and checks displayed data for expected output.

Database Tests

Database unit tests just aim to get 100% or near 100% code coverage.

1. Test add recipe
    a. Add new recipe
        i. Expected output - search and fine the new recipe
    b. Add existing recipe
        i. Expected output - adding returns -1 to indicate it failed
    c. Add recipe with user id that doesn't exist
        i. Expected output - returns -1 to indicate it faile
2. Test remove recipe
    a. Remove existing recipe
        i. Expected output - query for same recipe returns nothing
    b. Removing non-existing recipe
        i. Expected output - return -1 from function indicating that recipe doesn't exist
3. Test add to meal plan
    a. Add existing recipe to meal plan
        i. Expected output - find meal plan in query
    b. Add non-existing recipe to meal plan
        i. Expected output - check false assertion for return value; empty list from query on that recipe
4. Test remove from meal plan
    a. Remove existing recipe
        i. Expected output - don't find that in the query
5. Test add to extra items
    a. Add item with existing user
        i. Expected output -
    b. Add item with non-existent user
6. Test remove from extra items
    a. Remove item with existing user
    b. Remove item with non-existent user

UI Tests

The idea with these tests is to run the actual app and simulate button presses and check that all the outputs / actions that should occur under those events actually happen and display the correct data. The implementation of these types of unit tests were done with a hand-made test framework rather than an existing framework. Those frameworks are annoying and restrictive and testing the Qt framework is sort of finicky so we needed the control.

1. Test user login
    a. Check that tabs are disabled before login and enabled after login

2. Add recipe to meal plan
    a. Simulate clicking the checkbox and check the meal plan
3. Remove from meal plan
    a. Simulate clicking the checkbox and then unclicking check box, check meal plan is empty
    b. Simulate clicking the checkbox and then *misclicking* unclicking the checkbox; check that meal plan is not empty
4. Add / remove from meal plan integration test
    a. Add meal from dashboard and check values in meal plan tab, check meal plan tab in dashboard as well
    b. Add meal from dashboard and remove from meal plan tab, check empty meal plan tab in dashboard

Manual

Mobile

Once you install the mobile application on an android device, the first page you see is the dashboard. From there you can read the instructions on how you can use the application and what functionalities the application provides. From the dashboard, you can navigate to one of the six pages which include inventory, meal plan, recipe book root, public recipes, favorite recipes, and the shopping list.

For phase one, there is no actual functionality in the app, for now, you can access different activities and the layouts consist of data that is not pulled from the database (will be implemented in phase 2). The inventory activity helps the user to keep track of their inventory for different ingredients. In this activity, there are three sections being fridge, pantry, and spice cabinet. Secondly, the meal plan activity consists of meals planned for different days, for example, Cake for Tuesday and Rice and beans for Thursday. Thirdly, the recipe book root allows you to access the recipe book pages which are the public recipes and the favorite recipes. In the public recipes, you have access to all the different recipes uploaded by different users. And from there you can select recipes to be your favorite and they will appear in the favorite recipes activity. While the shopping list activity pulls ingredients from the recipes of the meals you have in your meal plan and suggests you shop for them. The shopping list can be manually edited.

Desktop

The desktop app has a tab layout to navigate through the different views, these are:

- Dashboard
- Meal plan
- Shopping list

- Public Recipes
- Private Recipes
- Create a Recipe
- User’s Recipe Books
- Inventory
- Sign Up Page
- User Profile

The dashboard has a list of recipes that you can browse through, collapse with the arrows on the left, and select your meal plan with the select box on the right. When you select a recipe, you will see it populate in the meal plan window in that same tab.

If you change tabs with the tab buttons at the top, you can change to the meal plan view. The meal plan view shows the same recipe browser and a meal plan browser that are editable in the same way. If you edit from here, you will see the changes populate in the other tab.

If the users click on the Inventory tab which is located in between the Shopping list and Profile tabs on the top in the main window, then the users can save a lot of time and money by adding what they have at their house to the Inventory table since the system will subtract the ingredients in the inventory from the ingredients in the meal plan to get the shopping list. All the ingredients in the database are shown on the sorted ABC list on the left.Thus, the users do not need to type but just click on the item name which they have at their home and click on the green (Add) button then this ingredient will be added to the Inventory table. The users do not worry that the users will add an item many times to the Inventory table, since the system just lets the users add an item only one time. Users also can delete the item in the Inventory table by choosing that item then click on the red (Delete) button.

The login page allows the user to login with an existing username, and the signup page allows the user to sign up with a unique username.

This is a Qt application written with qt for python - to run it you will need to install pyside6 and qt creator.

Getting qt-creator will require an account (it's free).

- Go to qt.io
- Click download try
- Go open source
- Scroll to and click download the qt installer
- Run the installer
- Install qt 5.15
    - Include qtcore
    - Qtdesigner
    - You can exclude anything you already have (like visual studio or mingw)

    - You can disregard anything like mobile/android/web/etc (this is a qt app made with python - let that guide what you install to keep the install to keep it as small as possible)

Dependencies

- Python 3.8 or later
- Install (using pip) pyside6
- Install (using pip) mysql-connector
- Ensure that your environment is set up correctly such that when you run a python command it is indeed the python for which pyside was installed
- If you are not using python3.9 you will need to pip install data classes (they are built into 3.9)

Running

- You can run the program by using `python mainwindow.py` or by opening the project in an ide and running / pointing config at mainwindow.py.
- Using the build.bat script - this will compile all the ui files into python code (these files are committed to the git so you shouldn't *need* to do this, but you can)
    - For linux systems, you can use a shell script (just remove that "@echo off")
- If you prefer to use an ide to run your code rather than the command line, you can have your ide of choice point its run configuration towards that build script.

Mobile

This is an Android app written with Java. To run it you will need to install the latest version of Android Studio, which can be found on developer.android.com. Once the android studio is opened, you have to open the project with the source code of this project. Make sure to separate the mobile app folder from the git repository so that android can run it. Once the app source code is opened in Android Studio, you can hit run. If all dependencies are up-to-date, an Android phone emulator will run the app. One thing to always make sure of is to have all activities stated in the AndroidManifest.xml. If not, the respective activity would not open once directed from within the emulator. If something fails, the most likely culprit is Gradle. Deleting the .gradle file and rebuilding the app may help. Also, make sure that Gradle is up to date as well.

Features and Future Work

Desktop

Completed Features

- Dashboard has working viewers of recipes, meal plan, and shopping list.
- The meal plan compiles recipes that the user has selected.

- The public recipe book viewer can display all public recipes.
- The public recipe book viewer can add recipes to the meal plan.
- The public recipe book viewer can remove recipes from the meal plan.
- The public recipe book viewer can add recipes to any custom recipe book.
- The public recipe book viewer can remove recipes from any custom recipe book.
- The private recipe book viewer can display all of the user's recipes.
- The private recipe book viewer can add recipes to the meal plan.
- The private recipe book viewer can remove recipes from the meal plan.
- The private recipe book viewer can add recipes to any custom recipe book.
- The private recipe book viewer can remove recipes from any custom recipe book.
- The recipe book browser contains tabs for creating a custom recipe book and any of the user's created custom recipe books.
- The create a recipe book subtab creates an empty recipe book with the title the user entered.
- The custom recipe book viewer can display all of the recipes the user has added to that book.
- The custom recipe book viewer can delete the current recipe book.
- The custom recipe book viewer can add recipes to the meal plan.
- The custom recipe book viewer can remove recipes from the meal plan.
- The create a recipe form creates a recipe with a title, public or private status, ingredient list, and instructions.
- The shopping list compiles from the meal plan's ingredient list.
- Extra items not in any recipe can be added to the shopping list.
- Items that are not in a recipe can be removed from the shopping list.
- The inventory can add any ingredient listed in the database.
- The inventory's quantities and units can be edited.
- The inventory can remove any item from the inventory.
- The inventory does not have the option of creating subinventories for organization.
- Basic login functionality - user can login, password verification making sure user cannot enter wrong password
- Basic signup functionality - user can sign up with unique username
- User database functionality - users are stored in SQL database when signing up
- Sign up error message - users are displayed with sign up error message when entering existing username
- Login error message - users are displayed with error message when logging in with non existent username
- Password error message - users are displayed with wrong password message when providing right login, but incorrect password
- Tab disabling - disabled tabs login and signup on app startup, all tabs except login enabled on successful signup, all tabs except signup enabled on successful login, tabs disabled on log out

Incomplete Features

- Dashboard does not have a working viewer of the inventory.

- The meal plan does not have an option to say that the meal was cooked, which would remove the ingredients and their amounts from the user's inventory.
- The meal plan does not allow the user to assign any date or mealtimes to the selected recipes.
- The meal plan does not allow the user to adjust how many servings they want and have the meal plan and shopping list update as a result.
- The public recipe book viewer does not support searching for specific titles, ingredients, or instructions.
- The public recipe book viewer does not support tabs filtering.
- The private recipe book viewer does not support searching for specific titles, ingredients, or instructions.
- The private recipe book viewer does not support tabs filtering.
- The private recipe book viewer does not allow the user to delete their recipes.
- The private recipe book viewer does not allow the user to edit their recipes.
- The create a recipe form does not add tags - predefined or custom - to the recipe.
- The shopping list does not take the inventory into account in its calculations.
- The shopping list does not aggregate items of identical name and unit.
- The shopping list cannot remove items from itself and add them to the inventory in a "purchased" operation.
- The inventory does not store quantities and units in the database.
- The inventory cannot add any items that are not ingredients in some recipe.
- Forgot Password - would really like to set up a gmail/SendGrid api account so that user can get access code when they press forgot password button, to get back into their account
- Update username and password - Because of time constraints we believed this feature was not as important, but would definitely have loved to complete it.
- Linking the desktop app to the user's mobile phone. The app had to be dropped and sending an email was too complex to be done in the time remaining.

Mobile

Completed Features

- A dashboard with buttons for the meal plan, inventory, shopping list, public recipes viewer, private recipes viewer, and custom recipe books browser..
- An inventory UI.
- A meal plan UI.
- A private recipes UI.
- A public recipes UI.
- A custom recipe book browser UI.
- A shopping list UI.

Incomplete Features

- All functionality beyond the dashboard's buttons linking to their respective UIs.

Future Work

After this iteration of the project and the presentation, we would like to continue working on this project. Firstly we want to try adding all the functionalities to the android application that already exist in the desktop application. After that, we want to continue working on the desktop application and integrate both versions so well that there are no bugs. Additionally, we want to work as a team to decide on what more functionalities we need to add to our project further on. The main thing that comes to our mind is to make the desktop application look prettier because right now it has all the functionality but the user interface is not very appealing. We will be looking into further options once we are done with this iteration.

The code could use some refactoring to improve object oriented design principles. For example, the database methods could return user defined objects instead of a list of data. This would allow for easier manipulation and display of information.We would also like to package and deploy our apps so they are publishable on the app store or as a desktop app. It would also be really interesting to add a food recommendation engine built using Pytorch so that users can get food recommended to them based on recipes they have made, so it is easier to plan their future shopping.

Reflection

We were able to complete a lot of work. What we planned earlier was to work on having two versions of this project, the android application and the desktop application. Unfortunately, due to not having enough manpower and experience for the android project, we could not add all functionality to it so after talking to the professor, we decided to drop the android application and jump on the desktop application. Also, we did lose a team member so that was kind of inconvenient for us.

To prevent this type of thing from happening it would help to make sure we have the necessary skills in the team to create what is necessary. Only two people have had experience with Android. Though this experience was not comprehensive enough to develop the app past a certain point.

Contribution Table

Member Name	Contribution Description	Contribution (%)	Notes
Mica	Report - Sequence diagram, some of the requirements, 75% of the complete and incomplete features list	15%	

	Presentation - Project overview, login demo Coding - Recipe book browser, custom recipe book viewer, delete recipe book, add and remove recipes from custom books		
Brice	Project Overview, Tests (Report) Presentation - wrote and presented test slides and slides on the dashboard and meal plan pages. Coding - wrote unittests for database and UI. added unit conversion to the shopping list generation algorithm. Contributed to	15%	
Mustafa	Report - Manual, Compiling, Future Work, Reflection. Presentation - Overview, Android demo, my recipes. Coding - inventory.py	13%	
Nestor	Report - Reflection, Future Work, Requirements Presentation - Public Recipes slide Coding - databasep.py, login.py	15%	
Tam	Report -Use case diagram, part of manual Presentation - the Inventory slide Coding - Inventory.py , prevent a user adding an item many times.	13%	
Vandana	Report -UML class diagram for this iteration Presentation -shopping list overview Coding - ShoppingList.py	14%	
Nikhil	Report - Future Work, Desktop, Manual, Features Presentation - SignUp slide Coding - databases.py, login.py, signup.py,	15%	

	maindwindow.py		
Naveen	Report - NULL Presentation - NULL Coding - NULL	0%	