FIT3178_A1: Design Specifications

Application concept: MeRecipe

MeRecipe is an iOS mobile application that aims to provide an easy and intuitive way to store personal recipes, whilst providing the necessary tools to create and use the recipe.

This app is designed for those who are casual cookers and want to quickly create, store, and access the recipes they have made without creating an account or purchasing any features. Many top-chart recipe apps on the app store offer a social/community platform built into that app that clutters the page with tips, reviews, and tutorials, making it very hard to find where to create a recipe at first glance. This app aims to provide a platform free of that content, targeting those who purely want to focus on creating their recipe, storing it, and accessing it on demand.

This app has a heavy emphasis on simplicity, meaning that the core delivery of this app is to allow users to easily navigate to their recipe and retrieve its details without too many button presses. Very much like a traditional recipe book, this app attempts to make use of visuals as much as possible, while keeping texts to a minimum. That is, the ingredients, instructions, and nutrients of a recipe are the main components.

Finally, not only does the app provide the basic tools to create and use recipes, but it also has many small inbuilt features that attempt to speed up this process. Creating the recipe will be supported by a couple of features to make this easier. A document scanner will be provided if the user wants to convert ingredients or instructions into text. Moreover, the nutrition section will be automatically generated for the user using external web services, saving them the hassle of manually searching up nutrition values and inputting them in the recipe. Bar charts and pie charts will be used to represent these details, adhering to the core goal of the app, which is to reduce text cluttering.

Also, a basic shopping list is provided, allowing the user to instantly add the ingredients required for a recipe straight into the list with a single press of a button. Manual inputting of ingredients is also supported. A recipe scheduler will also be provided, allowing users to input their recipes in a calendar to create a meal plan. There will also be an automatic meal scheduler feature, eliminating the struggle of not knowing what to cook.

Competition and Innovation

MeRecipe's competitors are two apps called 'Yummly Recipes & Meal Planning', and 'Tasty: Recipes, Cooking Videos'. They are ranked among the top 150 charts in Food & Drinks category. Both apps are cooking recipe apps that aim to provide a platform for people to create and share their recipes. Like MeRecipe, users can create their recipe, including detailing the ingredients and instructions when creating a recipe.

Looking at Yummly, the app provides a very beautiful interface upon entering the app. There are large tiles depicting the food that can be created from the recipe, which is one of its strengths. It also provides a meal planner and shopping list, much like the features that my app is going to have, making it easy for users to create and use their recipes. Another core feature is the ability to connect smart home devices to make the cooking experience of the recipe more enjoyable. A cool feature is that within the shopping list, it also provides an average cost of the ingredient, providing a rough estimate to the user on how much they need to spend to create this meal. Also, another unique feature of this app is its ability to recommend popular recipes as this is a platform to share recipes more so rather than to just store recipes. One issue that I have found with this app is that it is not obvious how to navigate to create a recipe. There is no shortcut button that leads to the creation of your private recipe. The only way to do this is to scroll down from the main interface and find a cell that says create recipe. This can be easily missed as there are many photos of recipes and a lot of coloring that can overwhelm the user. Moreover, users are required to create an account to use some features such as making their own recipe or accessing premium recipes. For those who want a quick and easy experience in creating their recipe, this may discourage some users from using this app. So overall, one must create an account, and find the create recipe section before being able to create their recipe. (In total, 3 button clicks are required to make a recipe.)

Looking at Tasty, it is very similar the Yummly. It provides a beautiful landing interface, making it easy to select other people's recipes. The search bar on the landing page is powered by an AI, making it easy for users to navigate to certain sections of the app without having to click many buttons. The app also provides a community page, making it easy to see what people are cooking and what is popular, very much like an Instagram feed page. Finally, there is also a meal planner page, which seems like a feature that most recipe apps have. One major downside is that to create your recipes, you need to sign up for a monthly subscription to make an unlimited number of private recipes. The shopping list is also a premium feature that needs to be paid.

Looking at the downsides of these very popular recipe apps, my app has a slight competitive edge in some features. First, it is free and does not require the creation of an account. This makes it very easy for users to create and use their recipes straight from the moment the app is downloaded and opened. Both apps seem to have some form of obstacle to create a recipe, from it being a paywall or numerous button clicks. My app priorities a quick and easy recipe creation experience, which is achieved by my landing page having an obvious create recipe button. Also, I have noticed that Yummly doesn't offer a nutrition section and Tasty does, but it's cluttered with values. My nutrition section aims to simplify this by using charts and using external web services to auto-generate these values based on the ingredients. Moreover, none of these apps have a PDF scanner, meaning users would have to manually transcribe handwritten recipes onto the app. MyRecipe will have a document scanner for handwritten instructions.

Feasibility and technology

MyRecipe will be developed on XCode using Swift and UIkit. The following describes how each section of the app functions and what kind of technology and features in UIkit will allow for its implementation.

View Recipe List Page - The landing page

This will be landing page of my app, which features a collection of cards that details a recipe. Users will need to be able to quickly select a recipe they want to view while being able to visualize what kind of dish it is, the difficulty and time required to create it. Moreover, there can be many recipes, so the user needs to be able to freely scroll through the recipe or have the option to narrow down the search by using a filter. Finally, we will have a navigation bar at the bottom of the screen that is able to navigate to the recipe view list, recipe creation page, shopping list and the recipe planner. To implement the list of all recipes, it is likely that a **UICollectionView** will be used, which hosts many **UICollectionViewCells**. Clicking on the **UICollectionViewCell** will navigate to the Recipe detail page. To filter for recipes, a search bar will be present at the top, which will be implemented with a **SearchBarController** along with the **UISearchResultsUpdating** protocol to enable the filtering feature. Finally, for the navigation bar at the bottom of the page, a **UITabBarController** will be used, holding small images with texts to help readers navigate around the app.

View Recipe Detail Page

This page is opened when a recipe has been clicked from the View Recipe List Page. This page will be likely created with an **UIScrollView** to enable the user to see the full content of recipe by scrolling vertically. Firstly, the title of the recipe and brief description will be at the top of the scroll view. The time taken to cook, and the difficulty will be labelled up there also. Scrolling down, we will see the required ingredients, instructions to cook, nutrition and then additional notes. The nutrition details will be represented by **swiftCharts**. Very likely that the app will use the **PieChartView** to display the carbohydrates, protein and fat percentages of the meal. There will be a button besides the ingredients section that will

allow for the automatic adding of all ingredients to the shopping list. Finally, there will be an editing button that will allow the user to edit the recipe by leading them to the create recipe page where their input fields are saved and are made available for modification.

Create Recipe Page

The create recipe page will be accessed by the navigation bar at the bottom of the screen as mentioned before. This section is going to be controlled by a **UIPageViewController**, allowing the user the swipe horizontally for inputting different sections of a recipe. The first "page" will be the overview section, where users can name the recipe, set the time required to cook it and its difficulty. There will also be a brief description and an available course to describe what kind of meal this recipe makes (e.g. lunch, breakfast, etc.) Depending on the field, we may have a text box, or we may segue to another view that has a **UITableView** of options cells to click on. Following this structure, we will have the ingredients page, likely to be composed cells in a **UITableView** that we can add the user can add on by allowing them to create more cells. Again, we may segue into another page to allow for the entry of quantity or ingredient type. The direction page will also follow the same structure as the ingredient page, except it will have an option to scan a document of instructions to text. We will use the **VisionKit** for this. There will be an extra notes page for the user to type whatever they feel necessary and a page for the user to upload a photo of their food to be displayed in the View Recipe List page by importing the **Photos module**. I plan to have a nutrition page that if left blank, an API is called (**API Ninja's nutrition API**) to auto fill the nutrition detail for the user. If they fill in the page, then the API is not called. However, in worst case, there will be no feature for the user to manually enter their nutrition value in. It is highly likely that **Firebase** will be used to store the recipes to allow users to save their recipe to view for next time they open the app.

Shopping List Page

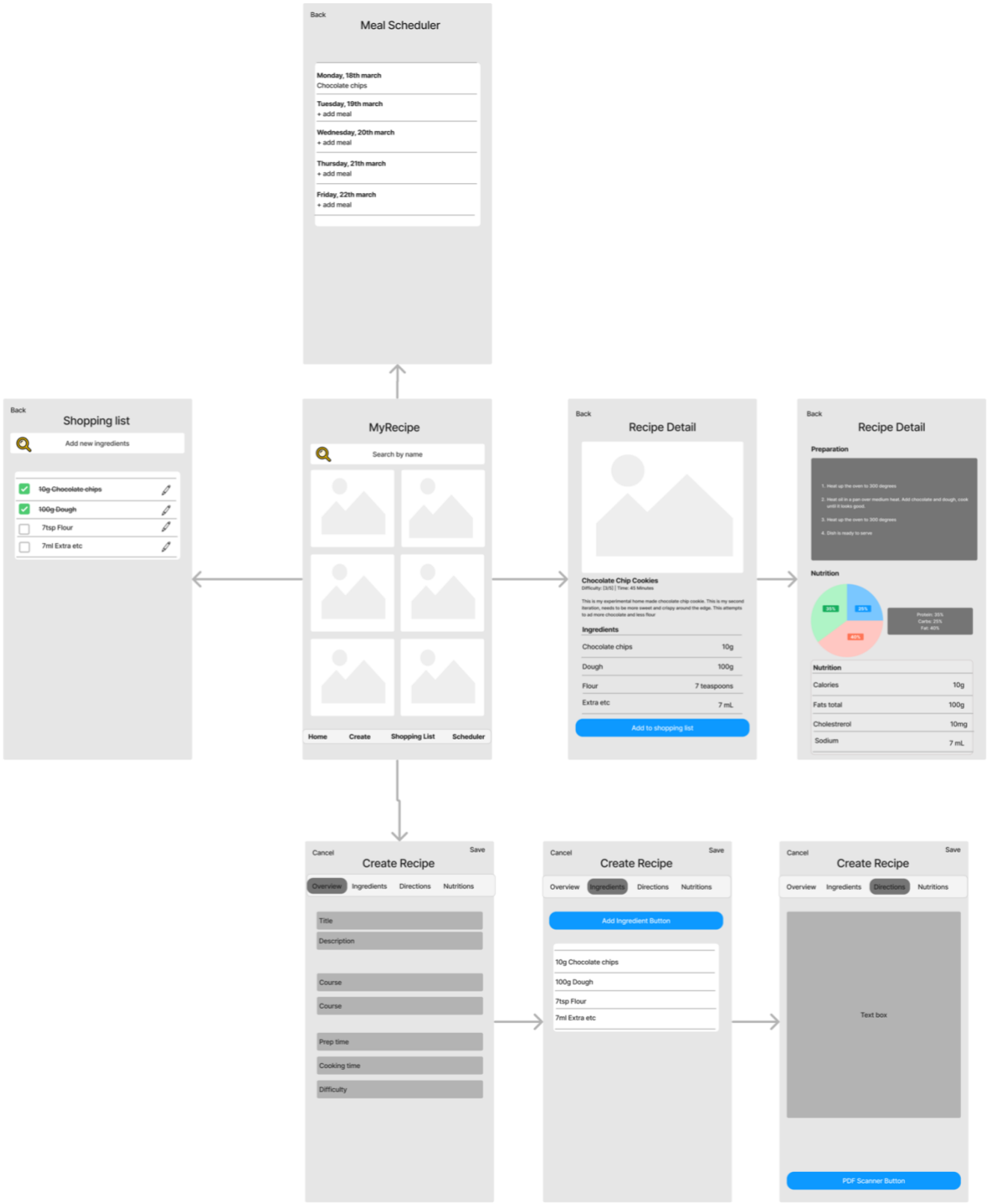
The shopping list page will be made with **UITableView**, where each cell will represent the ingredients needed to make the recipe. The cell will be styled with a **check mark style**, allowing users to know whether they have acquired it or not. They will be able to swipe to the left on the cell to delete the ingredient. It is also planned that ingredients will be stored with **Firebase** so that users can save their data.

Meal Schedule Page

This page allows the user to select as many recipes as possible they want to schedule on a certain day. The scheduler page will provide up to a week of available planning. This is likely going to be implemented with a **UITableView** where cells represent the recipes that are going to be made that day. Alternatively, if time permits, this page can be done similarly to the recipe view page where a **UICollectionView** table display the recipes as **UICollectionViewCells** rather than table cells with text. Once again, **Firebase** will help store these recipes in their respective dates. Once a day elapses, the recipe set for that day will be removed. Also, if time permits, a random meal scheduler will be implemented so that the user does not worry about planning what to eat for the week.

Interface Design and storyboard mock-ups

This page details the high-level app hierarchy and navigation for the user. Below, I will go into detail on what we are looking at, and why reasoning for this interface and how it adheres to Apple's Human Interface Guidelines. (Arrows show segue relationships / areas we can scroll/swipe to)



View Recipe List Page (MVP)

The View Recipe List page is also known as the home of the MyRecipe app. The image on the right is my rough plan on how the interface is going to be organized.

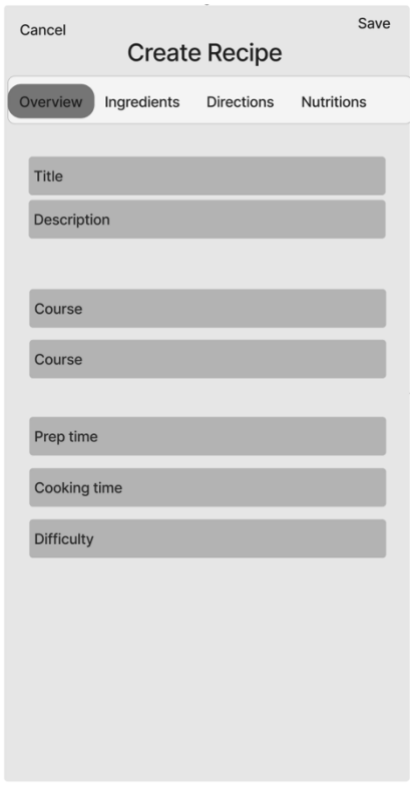
Starting from the top is the search bar to narrow down the recipes by name. As per Apple's HIG, placeholder text is shown to help prompt users on what to type into the bar. Moreover, the search bar's target is just listed below, which adheres to Apple's HIG as they recommend that items that people would want to search for should be near the search bar.

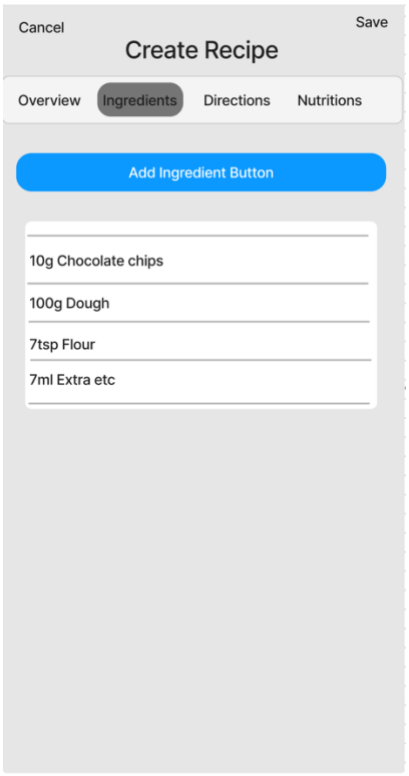
The collection view aims to offer cards that represent recipes as seen in the image on the far right for "Breakfast Bites". The style is based on Apple's Photo Album, allowing users to draw on their experience so that they know those cards are clickable. Moreover, the image displays the recipe name and image, hinting to the user that more about the recipe can be found by clicking it. This adheres to Apple's HIG as the visual background most likely mimics the action or page that is about to pop up.

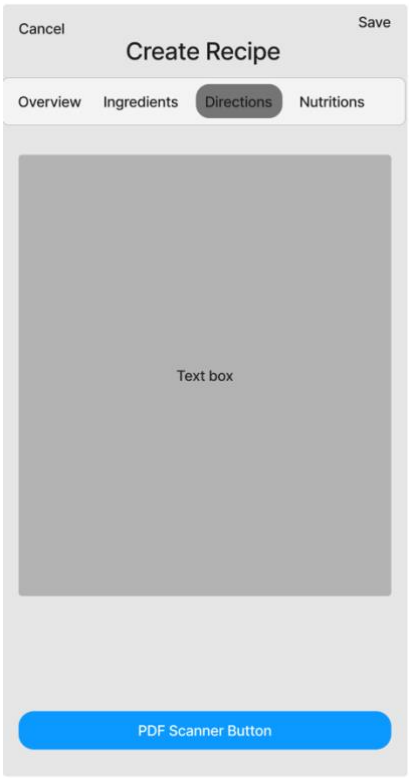
Finally, a tool bar is placed on the bottom section of the screen. According to Apple's HIG, it prioritizes commands that people use the most, which in our case, is the create recipe page and the home page to select the recipe. It consists of only 4 icons (the text in the image will be changed into icons), which we keep to the minimum to prevent cluttering. We will be sure to try use system icons so that users are familiar with the functions it will perform.

Create Recipe Page (MVP)

Clicking on the create button on the home page toolbar brings us to the Create Recipe page. On the top left and right of the screen, it features saves and cancel buttons, which is similar to most iOS apps. There is a segmented bar that details which portion of the recipe creation the user is up to. Swiping or clicking on the segmented bar will allow users to navigate through the different views. In the Overview section, users will be prompted with text fields that have a placeholder text, hinting what needs to be inputted there, adhering to the HIG. The ingredients section will have button placed at the upper-half center of the screen, prompting users to input their ingredients along with their quantity. Adding ingredients will populate a table as seen below. As per Apple's HIG, the button is constrained and positioned in such a way so that it is easily clickable by users. Finally in the direction section has a large text box. It is sized that way to anticipate cooking instructions as they can be quite long, adhering to Apple's HIG. (The image should say Recipe Instructions rather than text box)

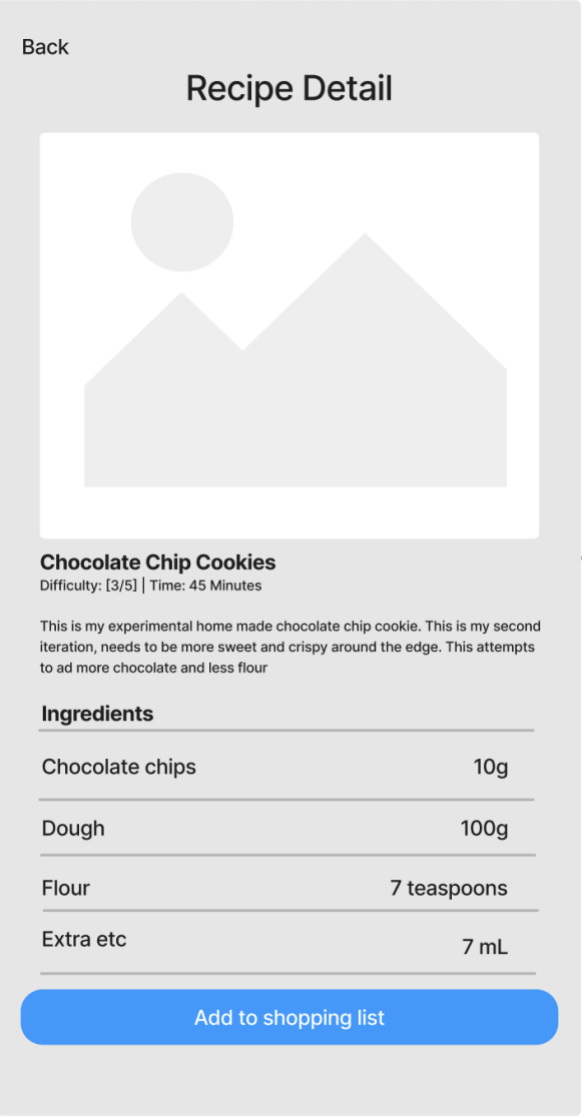


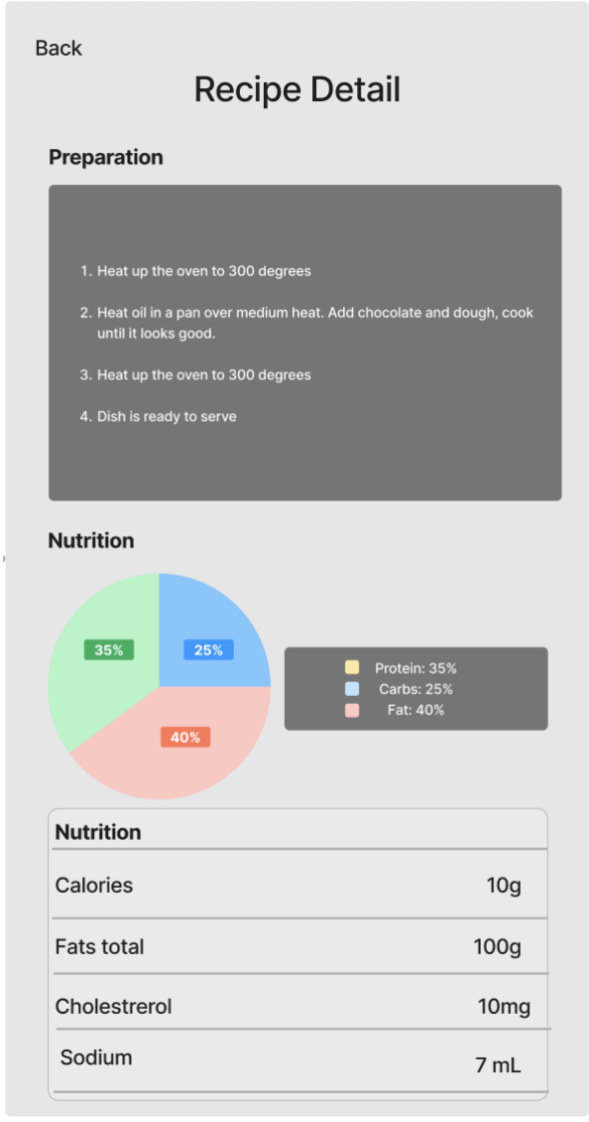




View Recipe Detail Page (MVP)

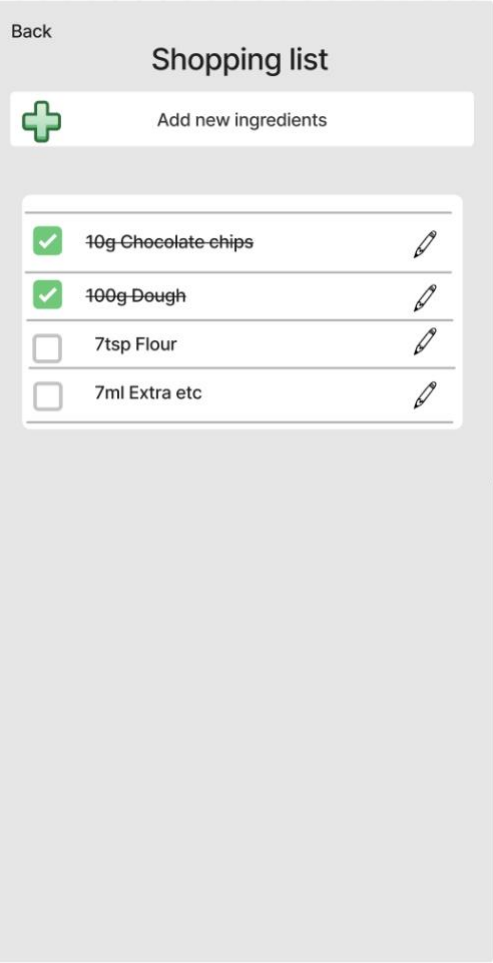
Clicking the recipe cards in the View Recipe List page (home page) brings you to this page. By scrolling down on image 1, users will find the rest of the recipe details as seen in image 2. This is implemented by a scroll view. The order in which the data is displayed is identical to the input orderings when the recipe was created, making navigating through this a more familiar experience for the user. Moreover, there is an obvious button below the ingredients that adds it to the shopping list. As per the Apple's HIG, its purpose is defined clearly with text and is positioned just below the ingredients section, which reassures the user that the button does exactly what it says it does. Finally, there is a Pie chart that represents the proportion of carbs, protein and fats there are in the recipe. There is a color legend used followed by a small table below that displays other nutrients details, adhering to Apple's HIG such that users do not need to interact with the chart to reveal critical information; it is nearly presented below the chart.





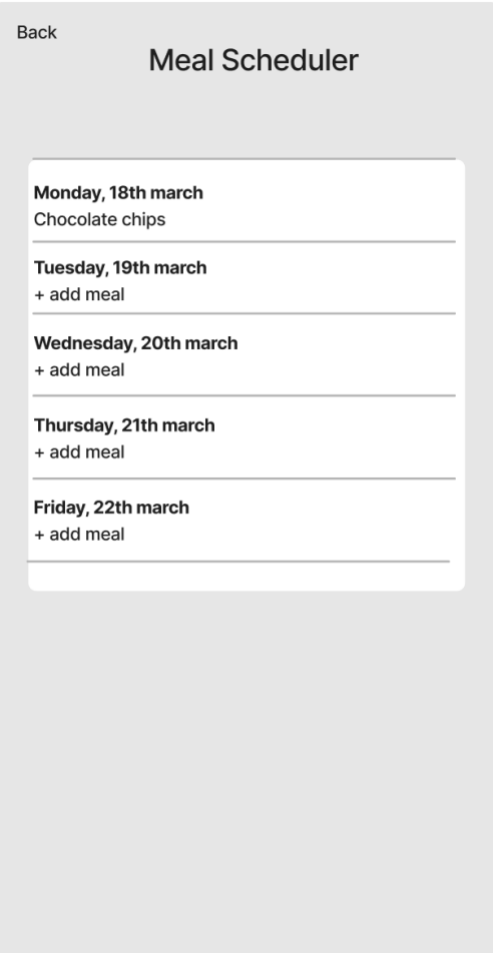
Shopping List Page

Starting from the top, there is a text field prompting the user to add ingredients. There is a system add icon, indicating the users can add ingredients and there is a placeholder text that helps guides users, conforming to Apple's HIG. Below this, there is a familiar check list that users can draw their previous experience from. Ticking the ingredient crosses off the item while there is a pencil icon indicating the cell is editable. These features are all recommended by Apple's HIG, particularly, selecting (ticking) the cell places a tick and a cross over the text which provides the appropriate feedback when users select the item.



Meal Scheduler Page

The meal scheduler page displays a table that shows the next 7 days/date from the present (for simplicity I only showed the next 5 days in the image on the right). This section is pretty straight forward. It draws on the user's experience with Apple's calendar as recommended by Apple's HIG and usability concepts. As they would probably guess, selecting a cell will enable them to select an ingredient that they have already made and add it to the calendar.



Scope and limitation

In this section, the minimal viable product (MVP) for each feature in the app will be defined. Overall, the most important features for the app to function is the View Recipe Page, Create Recipe Page and the View Recipe Detail Page. Firebase and the nutrition API are vital for the retrieval and storage of data. The shopping list and the recipe scheduler, in that respective order, will be the next most important features.

Note that since the whole purpose of the app is to allow for a quick and convenient pathway for users to create and retrieve recipes, these last two features are not necessary but are important details to keep this app competitive to other recipe apps as they tend to have one or both features. Now, we will detail specifically the 3 features that are necessary for the app to function and achieve its goal.

Create Recipe Page

- Be able to take inputs from the user when creating the recipe
- Be able to save ingredient details to create the recipe (name, ingredients, instructions)

View Recipe List Page

- Be able to clearly see the recipes. At bare minimum, be able to identify the differences between recipes by title or image.
- Users can navigate between recipes by scrolling

View Recipe Detail Page

- Be able to clearly see all the recipe details. That is, the user can scroll vertically to access this
- Users can delete the recipe

Estimated project timeline

This section will describe the development timeline for MyRecipe. As of writing this section, it is week 4, meaning that the developmental timeline starts week 5.

Considering that there are weekly iOS Portfolio exercises that extends all the way up till week 6, there may be an initial slow start for developing MyRecipe. Moreover, I am undertaking 4 heavily assignment-based units (no exams), meaning that I will be stretched for time in semester. Thus, this timeline will be more on the conservative side in terms of how much features are developed. That being said, here are my proposed timeline

Week 5 (learning webservices)

- Make an initial start on MyRecipe
- Set up project files and git.
- Aim to set up the View Recipe List page. Essential UIs are placed and constrained. Particularly, the UICollectionView and UICollectionViewCell may take up the most time and are the main feature of this page

Week 6 (Learning firebase)

- Attempt to make a start on the Create Recipe Page.
- Ensure navigation bar works; can switch between the create recipe page and View Recipe List page
- Try to see if we can create recipes, and display this as a card on the View Recipe List page. This may use persistent data storage. (Maybe firebase as we learn it this week)
- By now, really ensure that the View Recipe List page looks great at this is one of the core deliveries of the app.

Week 7 (learning maps and geolocation)

- Assuming week 5 and 6's task are finished, we can implement the View Recipe Detail page.
- Hopefully by now, we have implemented the features for the MVP
- Although the MVP may be done, we may need to do some conversions from data storage to firebase or keep the persistent.
- This would be an ideal week to do a demonstration

Week 8 (Working with images, audio and video)

- Clean up on UI. Navigation bar should be linked so that we can create recipes.
- If not done already, ensure that data storage is all set.
- If not already done, the View Recipe List page should be able to display recipe cards that can be clicked on to view the recipe.
- Begin on the shopping list
- If we have time, try implement the API to get nutrients and use swift charts to display this. Also try to get the PDF scanner working.

Week 9 (Motion sensor and swift charts)

- By now, we should have most of the knowledge required to implement the app (hopefully)
- Continue working on shopping list, API, swift charts and PDF Scanner
- This week would be an ideal time to do a demonstration.

Week 10 (Page scroll and notification)

- If time permits, we can now start on the meal scheduler (the least most important feature of the app)
- Clean up on unfinished features.

Week 11 / 12

- This week is left mainly empty for finishing up on uncompleted tasks. Busy weeks may have slowed developmental process
- Otherwise, hopefully the app should be completed around here.
- Finish up on meal scheduler if possible -> Implement auto scheduler feature
- Time for thorough bug searching
- Search for any UI improvements to make the app look better
- Prepare for final demonstration