

**GitHub Username:** [juandgaines](#)

# Body tracker

## Description

Body Tracker is your personal calories food tracker leading you through your dream's body available for Android devices. It will guide you through thousands of healthy recipes and suggestions to stay focus on your goal to obtain the body you want.

## Intended User

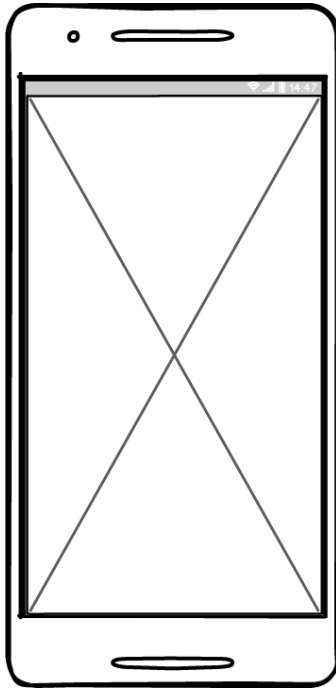
This App is designed for health enthusiast, fitness enthusiast or persons who wants to improve their dietary.

## Features

- Save favorite recipes.
- Natural language translator to update your daily calories consumption.
- Bar code reader to upload groceries (UPC reader)
- Daily recipe suggestion based on your preferences to keep you on track.
- Daily and monthly graphical data of your progress with proteins, carbs, and fats.

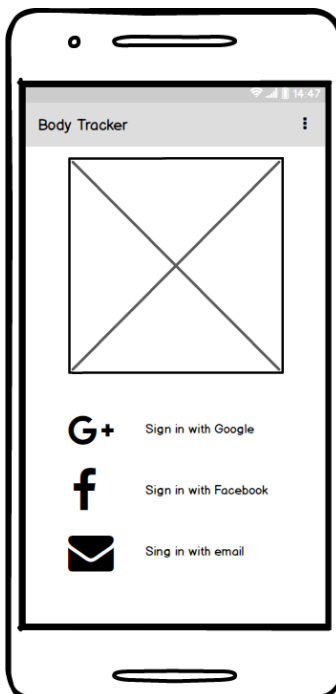
# User Interface Mocks

## Screen 1-Animation activity



This Activity will provide a welcome animation to the user.

## Screen 2-Log in Activity



Login will be implemented in order to keep preferences from the user and his/her personal data.

### Screen 3-First time login questionnaires (Viewpager)

- First page

Body Tracker

Questionnaires

Name

Birthday  /  /

Gender

Weight  Kg

Height  cm

Sedentary ☐

Moderately active ☐

Highly active ☐

Super active ☐

Next

Body Tracker

Questionnaires

Name

Birthday  /  /

Gender

Weight  Kg

Height  cm

Sedentary ☐

Moderately active ☐

Highly active ☐

Super active ☐

Next

Body Tracker

Questionnaires

Name

Birthday  /  /

Gender

Weight  Kg

Height  cm

Sedentary ☐

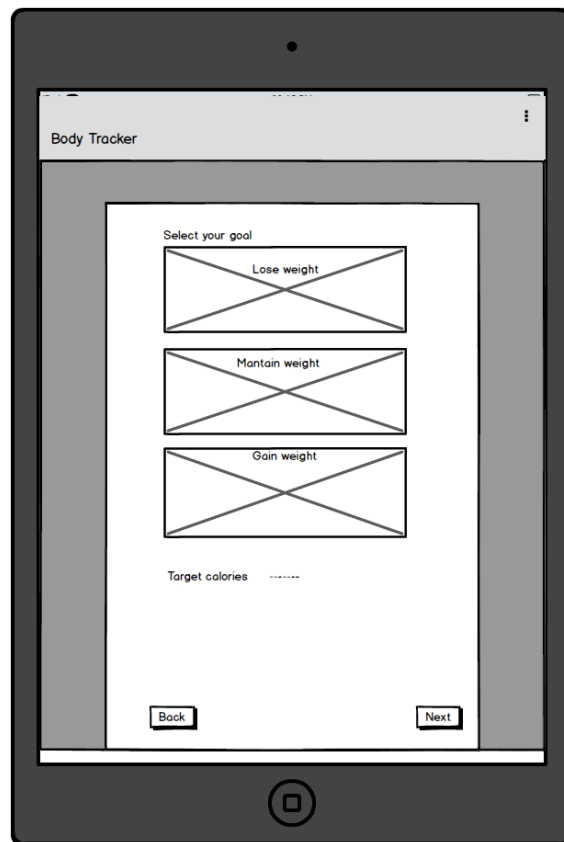
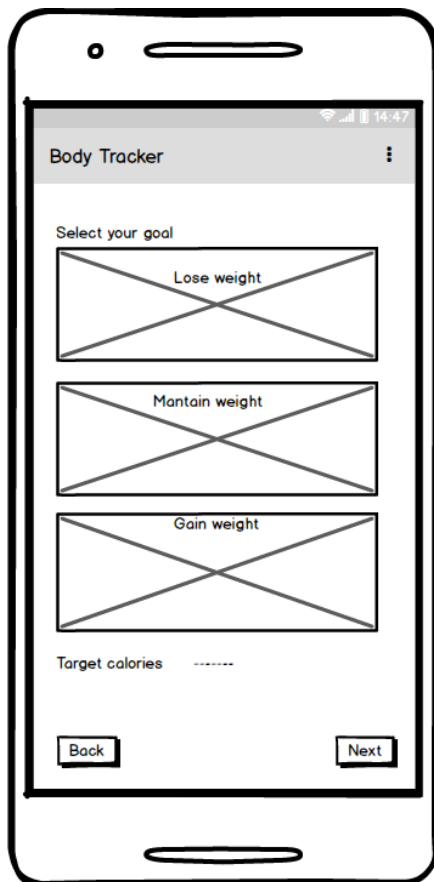
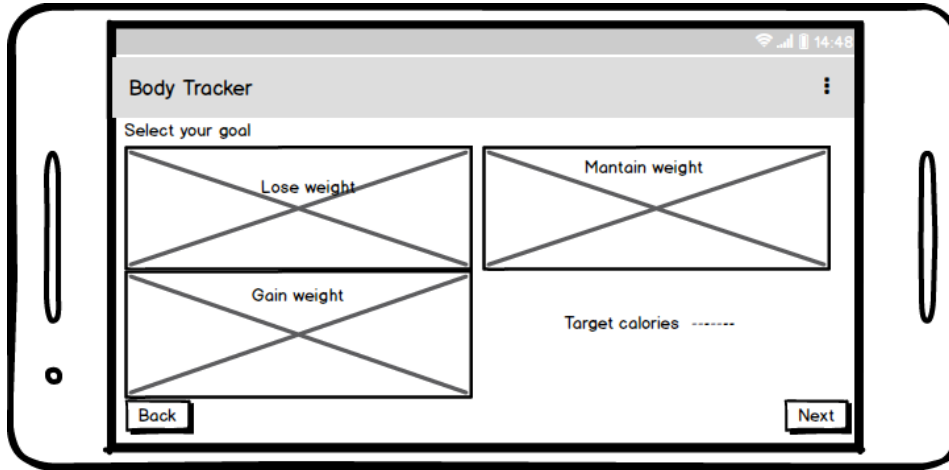
Moderately active ☐

Highly active ☐

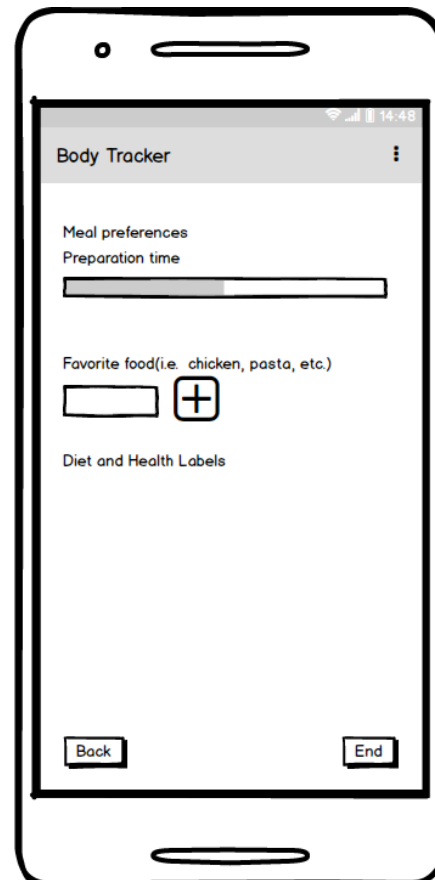
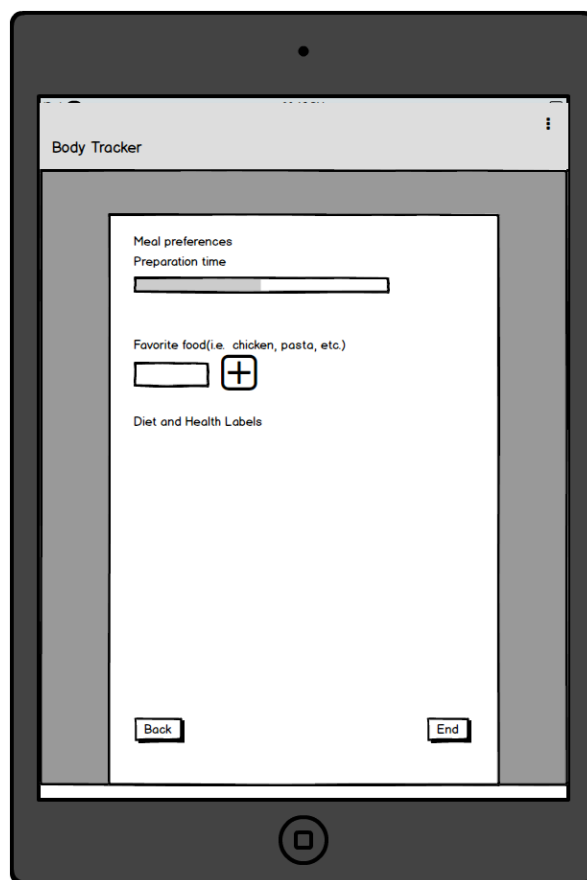
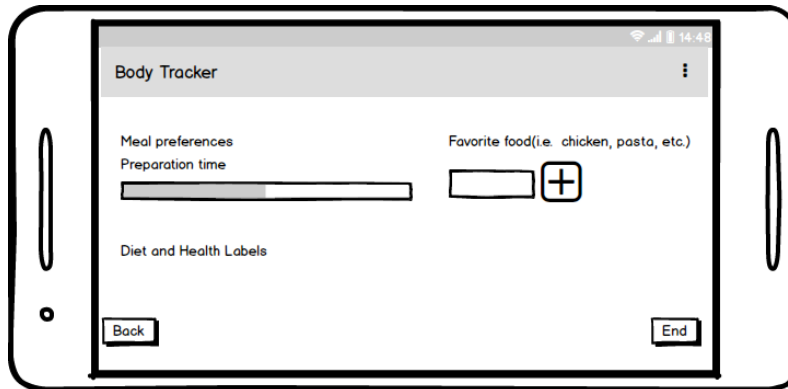
Super active ☐

Next

- Second page



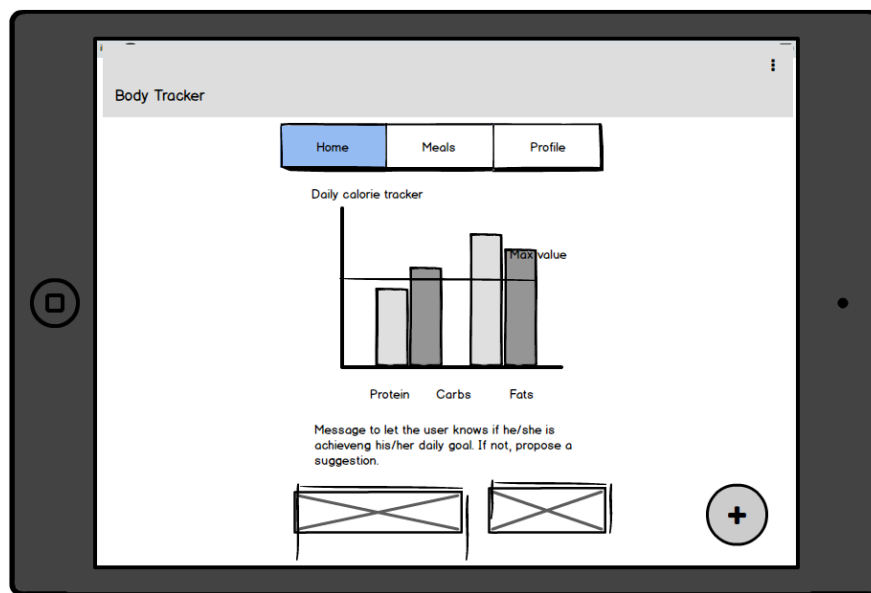
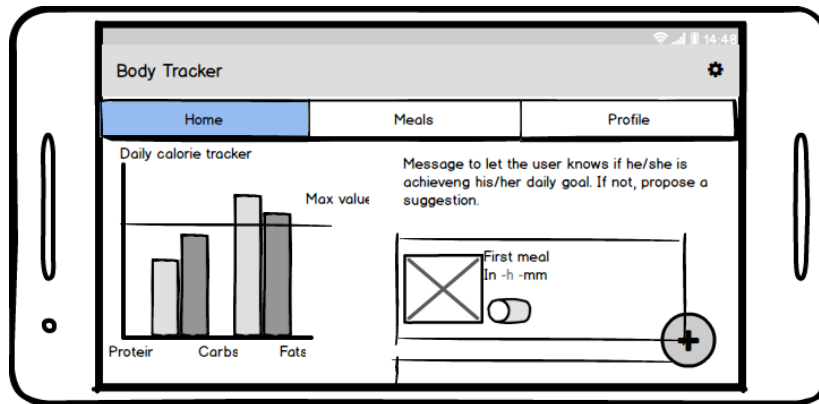
- Third page



The following activity will ask for the basic data from the user and calculate his/her caloric needs based on his dietary preferences.

Screen 4-Home

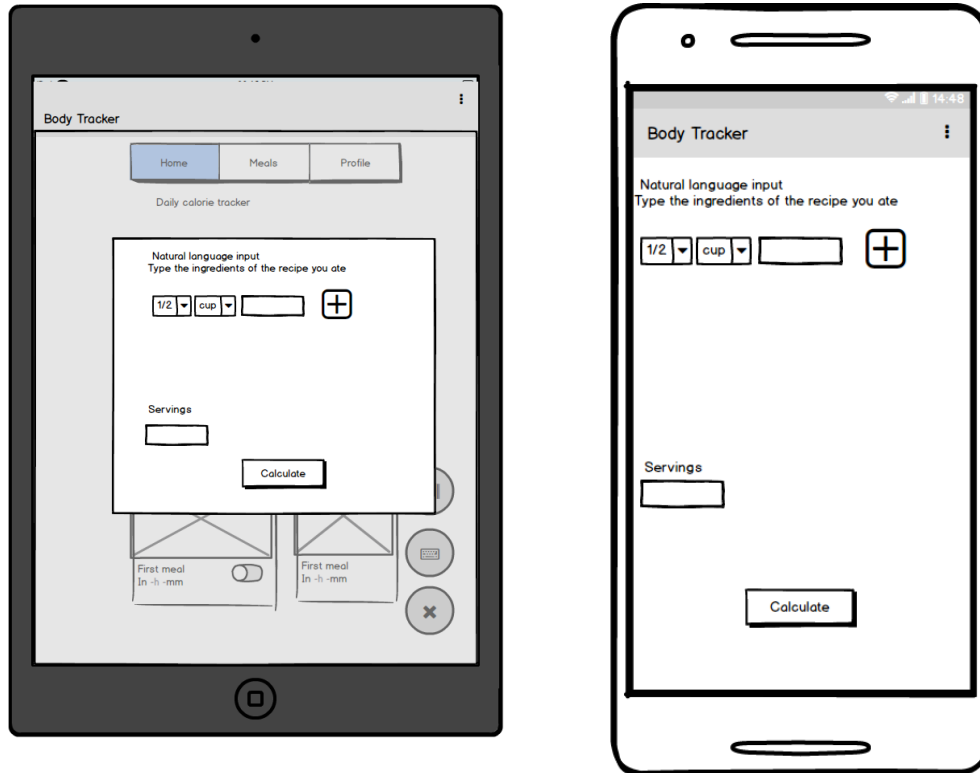




The activity will have a view pager which is going to be navigated through tabs. The home tab will provide the relevant daily and monthly data of the user's progress. In the other hand it will display the suggestions to improve the habits.

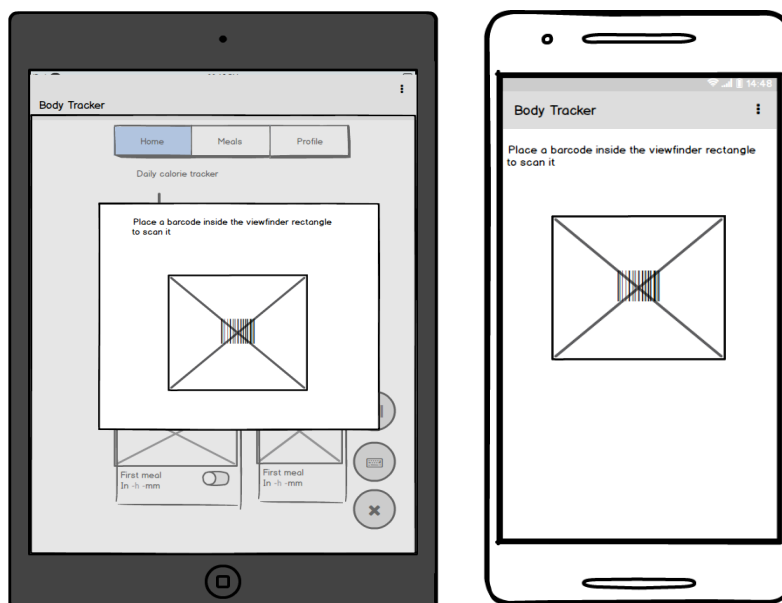
In addition, the food suggestion section will let the user add the suggested meals while the FAB will let the user to insert the unexpected food during the day. The FAB will contain two different options for two different input methods: natural language and bar code reader (UPC).

## Screen 5-Natural language



The natural language input method, triggered by the FAB, will let the user insert recipes manually to be updated and inserted by the app.

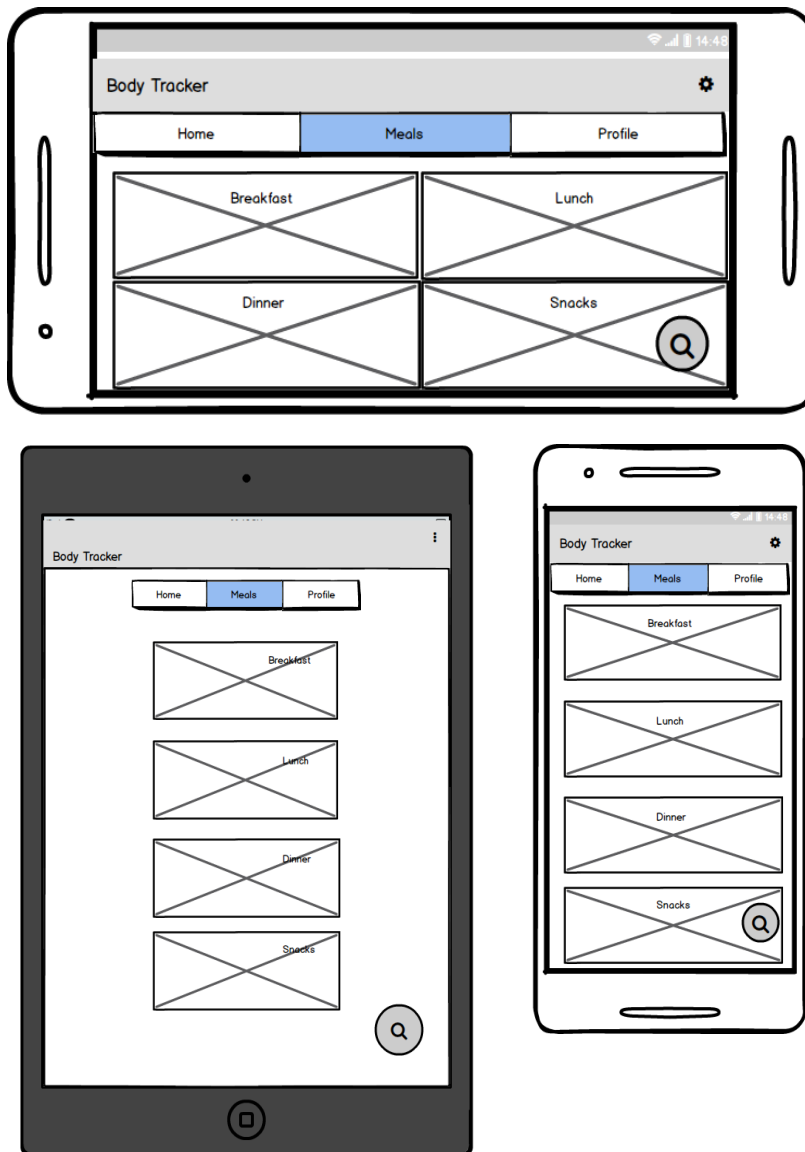
## Screen 6-Bar code reader (UPC reader)





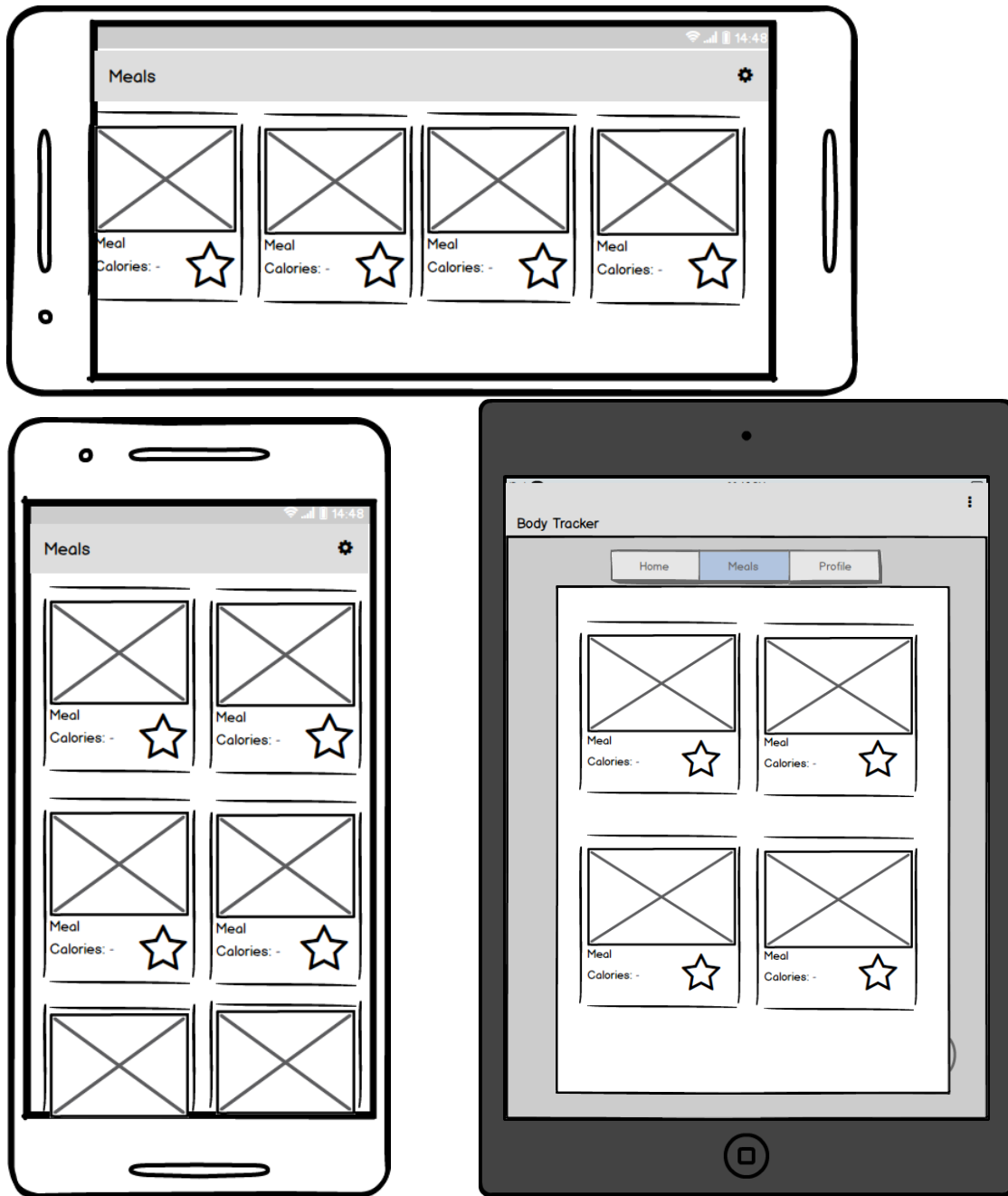
The bar code reader activity will read the UPC (Universal Product Code) from the label. If the product exists from the EDAMAN API, the product will be added automatically after confirmation. If the product does not exist inside the database, a manual input option to insert the macronutrients will be launched.

## Screen 7 Meals



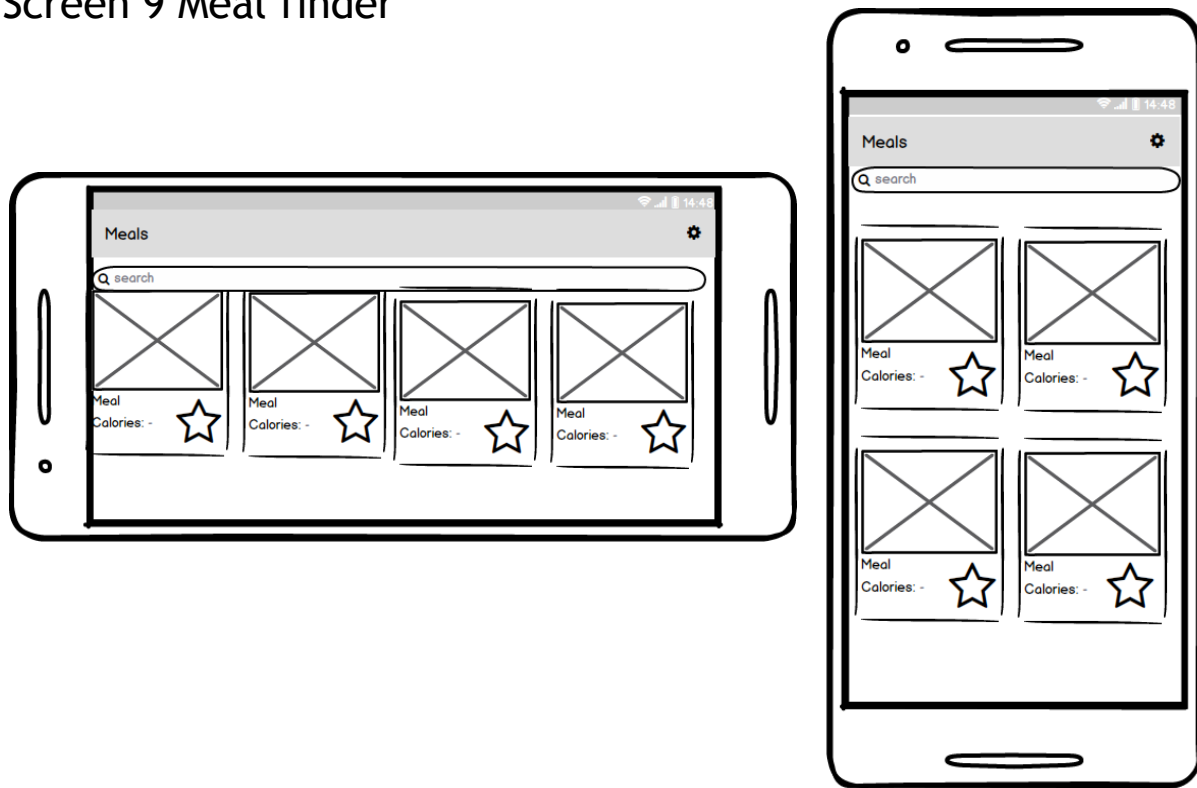
The meal tab (related with screen 4) will provide four options of meal to be selected for the personal database of the user: breakfast, lunch, dinner, and snacks. Also, it will have a FAB to launch a finder activity (Screen 9).

## Screen 8 Meals- Gridviews /breakfast/lunch/dinner/snacks



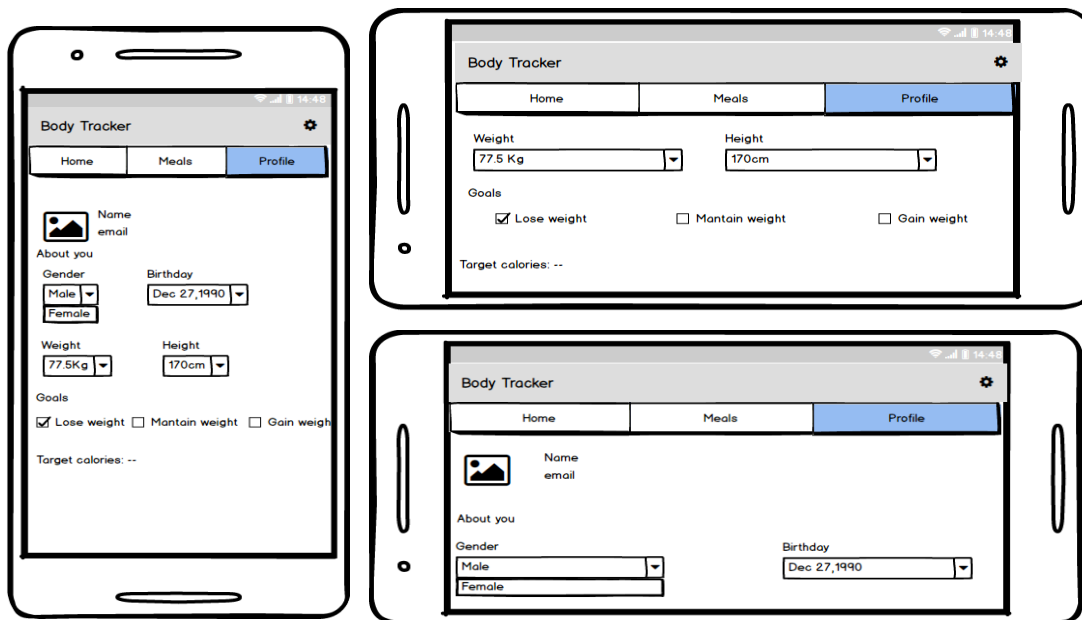
Depending of the user's choice (breakfast, lunch, dinner, and snacks) , RecyclerView will be displayed with a grid depicting all the options available based on user preferences. The intended fetching data will be handle through an AsyncTask(on-demand request), requesting the desired data.

## Screen 9 Meal finder



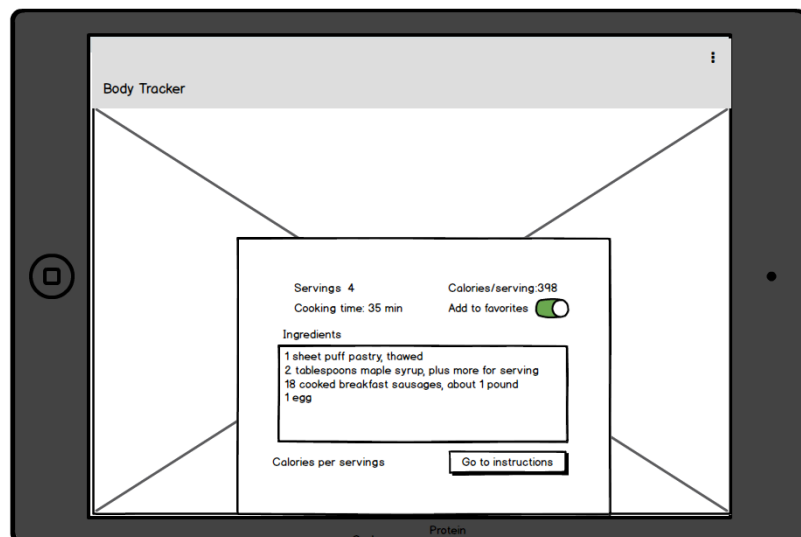
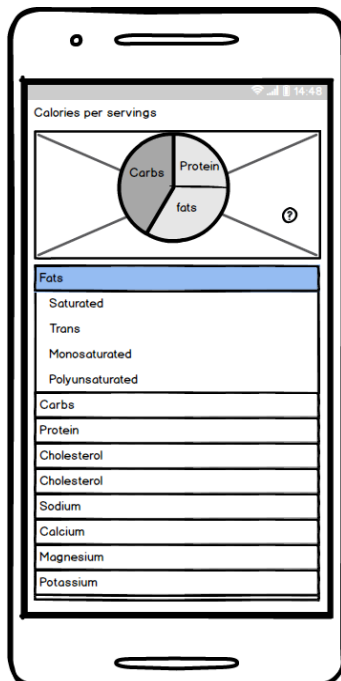
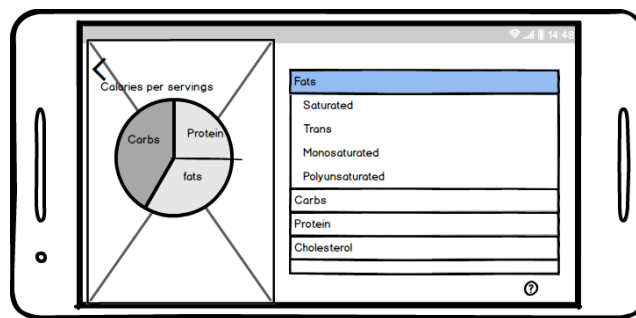
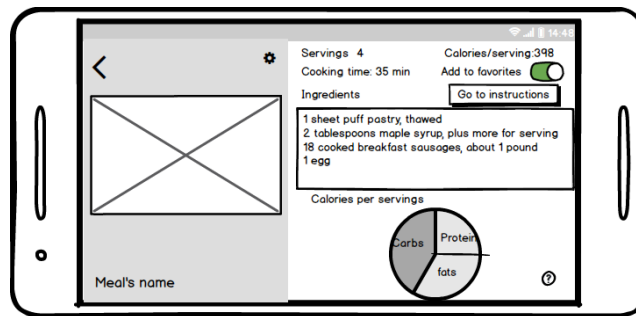
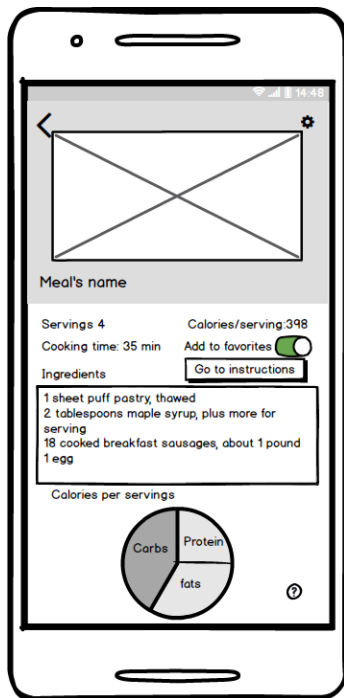
The meal finder will help the user find specific food with or without preferences (See screen 7). The intended fetching data will be handled through an AsyncTask (on-demand request), requesting the desired data through the user's keywords.

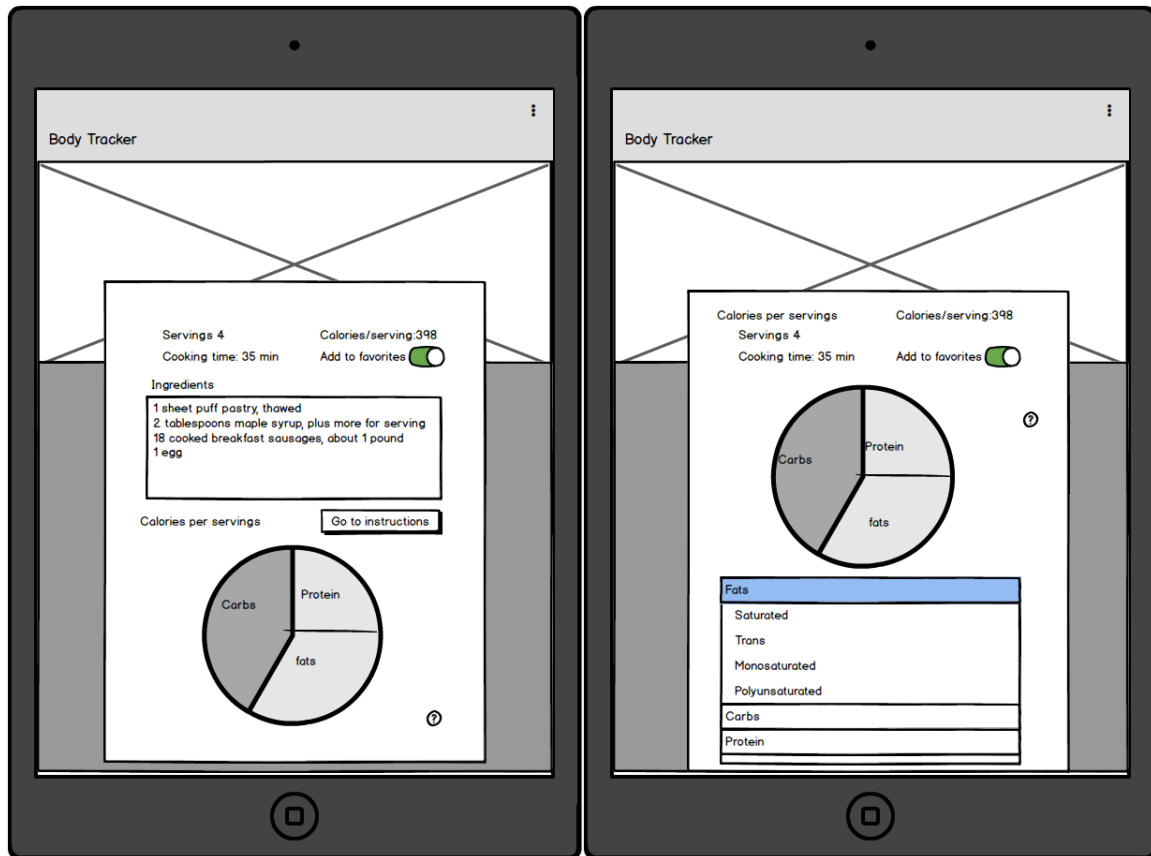
## Screen 10 Profile



The profile section will keep the personal data of the user such: email, gender, birthday, weight, height and goals.

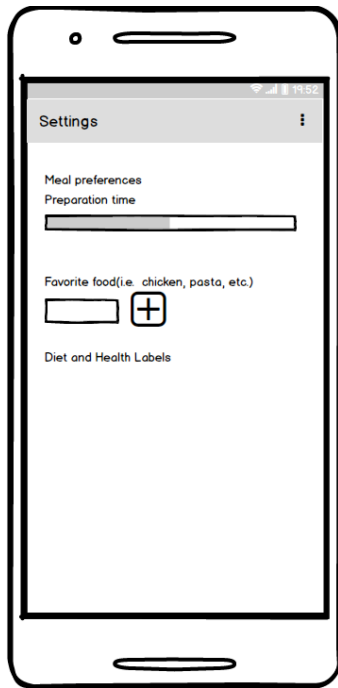
## Screen 11 Recipe Details Activity





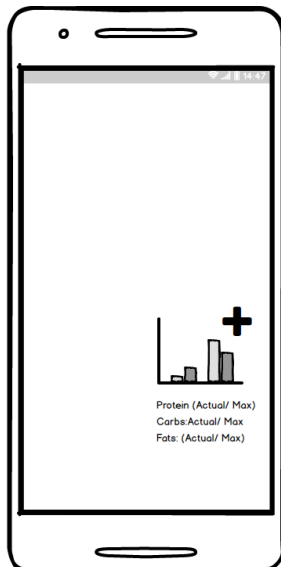
The Recipe details will depict the relevant data (servings, calories, ingredients and macronutrients) of the recipe selected by the user. The activity will be able to expand the data when a question symbol has been clicked.

## Screen 12 Settings activity



Settings activity will store the preferences of the user that apply in his/her query when searching for data.

## Screen 13 Screen widget



The widget will contain the daily data of the calories consumed during the day, and it will redirect to home page to add new data.

# Key Considerations

## How will your app handle data persistence?

The user will save the preferred recipes using a local database implemented in room when connection to internet is not available. The fields that the recipe tables will expect are: name, uri, image url, instructions url, source, servings, diet labels, health labels, ingredients, calories, proteins, fats and carbs. In the other hand, the recipes will be stored in a Firebase database when connection to internet is granted. The local and remote database will be synchronized through background tasks using JobDispatcher.

The calories consumption table will work in a similar way. The fields needed for this will be: id, date, calories, meal's name, carbs, fats, proteins.

## Describe any edge or corner cases in the UX.

- When the user generates a typo accidentally in the natural language input method, the app will ask him/her to enter the recipe again.
- If the bar code wasn't found in the barcode input method, the data will be inserted manually.
- If no recipe has been selected, a couple of suggestions will be depicted to the user.

## Describe any libraries you'll be using and share your reasoning for including them.

[Picasso\('com.squareup.picasso:picasso:2.7.1828'\)](#): It will handle the loading and caching of images.

[MPAndroidChart\('com.github.PhilJay:MPAndroidChart:v3.0.3'\)](#): Chart drawing to display the daily and monthly calorie tracker and the recipe macronutrients proportion .

[Retrofit\('com.squareup.retrofit2:retrofit:2.5.0'\)](#): HTTP client to fetch data from the API servers (EDAMAN).

[Mobile Vision\(com.google.android.gms:play-services-vision:17.0.2\)](#): The Vision library, from Google play services will help to scan the barcodes from the grocery products.

[CardView\('com.android.support:cardview-v7:28.0.0'\)](#): The CardView will be used to contain the view from the recipes to even relevant options.

[FirebaseUI-Android\('com.firebaseui:firebase-ui-auth:4.3.1'\)](#): FirebaseUI allows you the user quickly connect common UI elements to Firebase APIs.

## Describe how you will implement Google Play Services or other external services.

Firestore Authentication: Authentication will be required to associate the preferences and personal data of the users with their email, Facebook or Gmail accounts. All the data will be store locally and synchronized with a Database built with Firestore.

Firestore Real-time Database: The app will use Firestore database in order to save two main groups of data: favorite recipes and calories consumed by the user. The Real-time Database will store the data as follows:

The fields that the recipe tables will expect are: name, uri, image url, instructions url, source, servings, diet labels, health labels, ingredients, calories, proteins, fats and carbs. The calories consumption table will work in a similar way. The fields needed for this will be: id, date, calories, meal's name, carbs, fats, proteins.

## Required Tasks

In order to complete the mentioned App, the following task must be implemented:

### Task 1: Project Setup

- Configuration needed libraries for the project in the build.gradle files
- Set API keys on build.gradle file
- Set-up Firestore project on Firestore Console.
- Enable Firestore Realtime Database
- Enable Firestore Authentication.
- Prepare and upload imagery for the project.
- Prepare Spanish and English version for the App in the strings.xml file.

### Task 2: Implement UI for Each Activity and Fragment

- Build UI for Animation Activity (Phone with landscape and portrait, tablet landscape and portrait) with content description and RTL support.
- Build UI for Login Activity (Phone with landscape and portrait , tablet landscape, etc.) with content description and RTL support.
- Build UI for First time login activity (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build UI for Home Activity (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build Home's FAB animation layouts (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build UI for Home tab (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build UI for Meals tab (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build UI for Profile tab (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build UI for Settings (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build UI for Bar scanner reader Activity (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.



- Build UI for Natural language Activity (Phone with landscape and portrait, tablet landscape, etc.) with content description and RTL support.
- Build UI for Widget.

### **Task 3: Validate and test data from API servers**

Fetch data from server and validate the purpose and usage.

- Fetch data using Retrofit
- Validate mathematical operations to get the macronutrients proportions
- Filter unnecessary data.

### **Task 4: Implement Java code for Activities.**

- Build code for Animation Activity (Phone with landscape and portrait, tablet landscape and portrait).
- Build code for Login Activity (Phone with landscape and portrait, tablet landscape, etc.).
- Build code for First time login activity (Phone with landscape and portrait, tablet landscape and portrait).
- Build code for Home Activity (Phone with landscape and portrait, tablet landscape and portrait).
- Build code for Home tab (Phone with landscape and portrait, tablet landscape and portrait).
- Build code for Meals tab (Phone with landscape and portrait, tablet landscape and portrait).
- Build code for Profile tab (Phone with landscape and portrait, tablet landscape and portrait).
- Build code Settings (Phone with landscape and portrait, tablet landscape and portrait).
- Build code Bar scanner reader Activity (Phone with landscape and portrait, tablet landscape and portrait).
- Build code Natural language Activity (Phone with landscape and portrait, tablet landscape and portrait).
- Build code for Widget
- Handle exceptions and errors.

### **Task 5: Material design implementation**

- Build code for Home's FAB animation.
- Develop enter and exits transitions where needed
- Parallax for Recipe details activity.

### **Task 6: Notifications**

- Build notifications of remaining time for next meal using Jobdispatcher.
- Implement a suggestions of recipes every day using Jobdispatcher.