GitHub Username: dofukuhara

Nutritional Assistant

Description

Want to know the nutrition facts information about the ingredients of the meal that you are going to prepare?
With Nutritional Assistant you can find all of this information in one place, so you can use this information to prepare a tasteful fitness food and improve your quality of your food! =)

Intended User

This app is intended for people who are concerned about the quality of their food and pretend to have a more healthier food.

Features

With this app the user will be capable of:

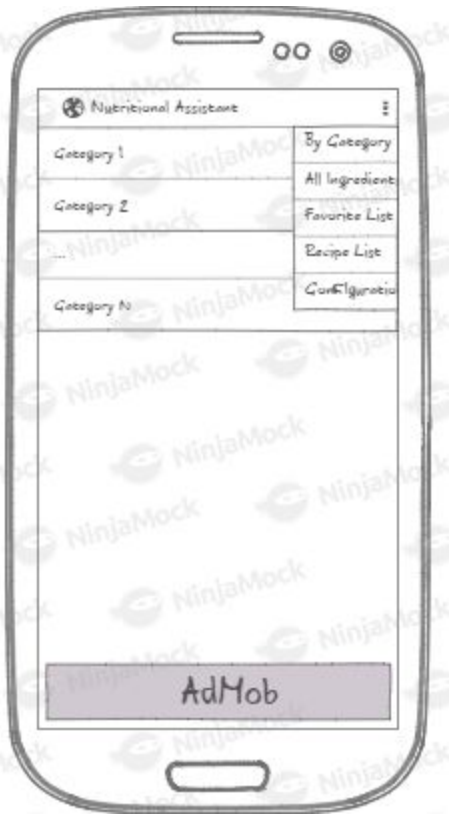
- List all the available ingredients;
- List all ingredients, sorted by “category”
- Create an offline ingredient favorite list, for a more convenient listing
- Sync the favorite list in he server, so the user can retrieve the list in any device
- Create a “recipe” item, where ingredients can be combined and their nutritional facts be added

User Interface Mocks

Screen 1

	This is the configuration screen If the user opens the application for the first time, this activity will be launched prior to Main Activity to handle Mobile x Wifi data access and Authentication.

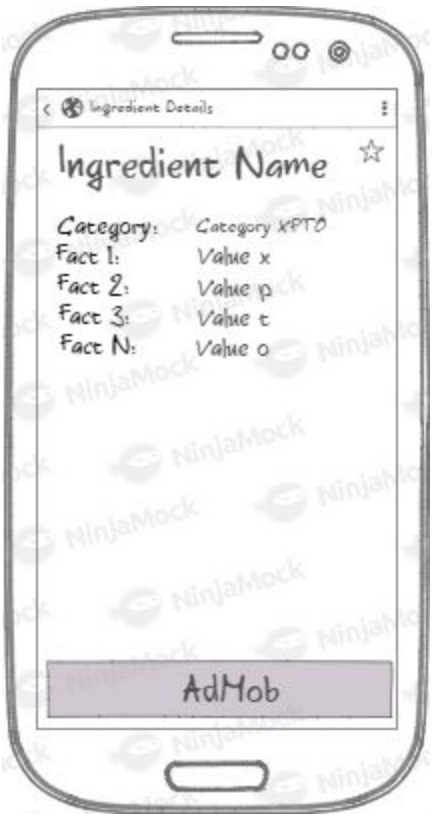
Screen 2

	The Category grouping ingredients will be the Main Activity. To use see other activities, the user can use the Menu option. This menu will be the same for all activities. Also, in every screen, there will be an AdMob banner in the bottom side of the activity.

Screen 3

	When the user decides to open the Ingredient list (either by Category sort or All Ingredient List), a list with all available ingredients will be displayed, so the user can click on it to check for more details

Screen 4

	In the Ingredient Details activity, the user will be able to see all the information available to the target ingredient, as well as mark it as a favorite one by clicking in the start button.

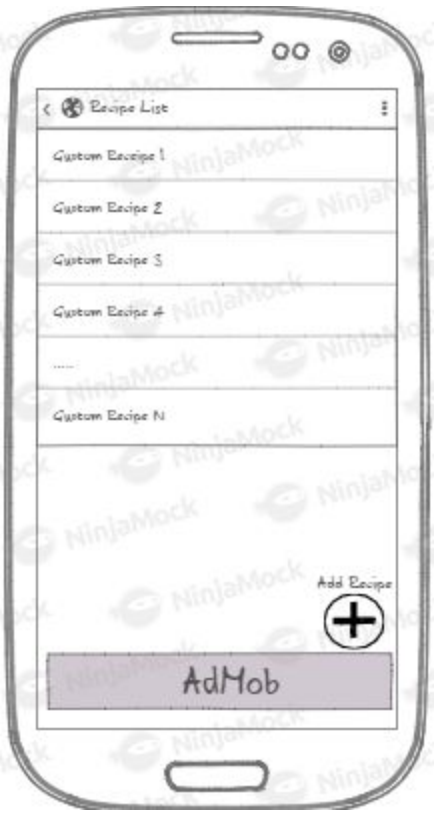
Screen 5

	All the ingredients marked with a start in the Ingredient Details activity will be listed in here

Screen 6

	User can swipe the item to remove it from Fav list. If swiped, a Snack Bar will be displayed, asking if the user wants to undo this operation.

Screen 7

	In this Recipe List activity, the user will be able to see all the custom recipes created, as well as clicking in the FAB button to create another one.

Screen 8

	When clicked in the FAB button, a dialog button will be displayed, asking for the Custom Recipe's name.

Screen 9

	In the Custom Recipe Details activity, it will be able to see all the ingredients (as well as clicking on it to jump to its details screen), see the nutritional facts combined and click in the delete recipe button.

Screen 9

	When the user goes to create a widget, a configuration screen is displayed, in order to choose which recipe the user wants to see the information.

Widget 01

	The widget will show all the information of the Recipe, as in the Recipe Details activity.

Key Considerations

How will your app handle data persistence?

In order to handle Data Persistence, it will be used Content Provider to store Favorite and Custom Recipe information, as well as saving both Category and Ingredients list in the Content Provider, in order to provide an offline experience to the user.
Also, all of these information will be also saved in the cloud using Firebase.

Describe any edge or corner cases in the UX.

Unfortunately, this Rest API only return the data in portuguese, so the target users will need to know portuguese in order to understand the ingredient names.
As the Json result doesn't has any version on it, a "refresh" condition, such as reload only when user explicitly wants or when the data reaches 1-week age to be refreshed.

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

- RetroFit, in order to handle the REST Json handling
- Firebase to handle user authentication and server side data persistence

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

- Firebase to handle user authentication and server side data persistence
- AdMob: Make use of AdMob advertisement banners in the bottom of the screens, as well as shown ads when returning from Favorites and Recipes lists.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

- Create the “Project Capstone” project in GitHub
- Configure project’s libraries
- Setup Firebase account

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Category List, that will be used as the Main Activity
- Build UI for Configuration screen, that will be used as the first entrypoint in case that if it is the first time the user had opened this application
- Build UI for Ingredient List, where it will be displayed a list with Ingredients, sorted by Category or All Ingredients
- Build UI for Ingredient Details
- Build UI for the Favorite List activity
- Build UI for the Custom Recipe List activity
- Build UI for the Custom Recipe Details activity

Task 3: First Time App Entry

Create a mechanism to identify if the user was firstly opening the application, so instead of opening the Main Activity, it will launch the configuration screen.

Task 4: REST Api

In this task, it will be developed the REST query by using RetroFit.

- Create the Model objects of Category and Ingredient
- Configure RetroFit with server endpoint
- Create the queries to retrieve (https://github.com/raulfdm/taco-api):
    - All Categories
    - Categories per ID
    - All Ingredients
    - Ingredients per ID
- Create the necessary callbacks to handle server response with JSON results

Task 5: Server and User Data Persistence

Create a Content Provider provider in order to:

- Store the data acquired from REST query
- Save Favorite List
- Save Recipe List

Task 6: Recipe Combination Calculation

In order to build the Recipe information and nutrition facts calculation, it will be used an AsyncTask, in order to fetch the ingredients information and the necessary calculation, as well as the data persistence.

Task 7: Authentication

Configure the application to use Firebase in order to properly authenticate the user.
If the user's account already have information stored in the server, download them and update the app's Content Provider.

Task 8: Firebase data synchronization

Configure the application to push Favorite and Custom Recipe information to Firebase.

Task 9: Adding error handling cases and bugfixes

Inspect the application source code and perform the necessary fixes, as well as including missing error handlings (no network connection, response without valid json data, etc.)