Frigo
Design document

Tucci Francesco 10607818
Verzeni Marco 10577271

Design and Implementation of Mobile Applications

Summary

1.0 Introduction

1.1 Idea

The idea of this app is to help people fight food waste. We often buy several products that we do not use immediately and end up forgetting having them in the fridge with the risk of letting them expire.

Thanks to Frigo, people will be able to check the products in the fridge at any time, thus avoiding buying products they already have, and keep under control the expiration dates of all the products in the fridge or pantry. The app will show the list of available products in order of expiration date and will also show recipes that maximize the use of expiring foods to always discover new dishes and at the same time avoid throwing away products that are still edible. The app has been realized to be used by every type of person given its simplicity, the clear interface and its usefulness in day-to-day life.

1.2 Functionalities

The app currently implements the following features:

- **Ingredients insertion**: an ingredient can be inserted in the virtual fridge through the home screen by tapping the add button. The name of the product can then be specified by the user or retrieved by the app itself though a barcode scanner. In particular, the scanner will be able to retrieve the name and the image of the product. Lastly, the user will need to specify the quantity and expiration date of the product in question and possibly a custom photo.
- **Ingredients update**: the info related to an inserted product can be modified by the user at any time by selecting the edit option of an ingredient from the Frigo home screen.
- **Ingredients removal**: an ingredient can be removed from the virtual fridge by the user through the contextual menu in the home screen.
- **Recipes scrolling:** from the recipe screen it is possible to observe all the recipes suggested by the app given the content of the virtual fridge. The different dishes are presented in such and order that prioritizes the ones that make use of the largest amount of available ingredients.
- **Recipe visualization**: through this screen the user can read the various info about the chosen recipe like its summary, the tags, the number of servings, the time of preparation and lastly all the steps required to prepare it. An image is also available to show the finished product.

1.4 Implementation choices

The whole app has been developed by focusing on the simplicity of the interface and an intuitive usage. All actions can be performed through single taps to not complicate the experience and hide any functions. The graphic aspect has been particularly curated to be easy to grasp by any user, pleasing to the eye and functional, by showing all the relevant information about ingredients and the recipes.

A lot of importance has been given to the idea of not wasting food, by giving precedence to recipes that consume expiring products, and to the responsiveness of the whole app through the enforcement of updates to the shown recipes every time the ingredients list is modified.

2.0 Design

2.1 Implementation tools

Frigo has been implemented through the React native framework to allow the deployment of the app on both Android and IOS devices. Typescript was chosen as the base programming language to exploit the basic Javascript functionalities combined with type checking. This allowed to reduce the number of errors at execution time by making sure of the type of all the data that travelled throughout the app at each moment.

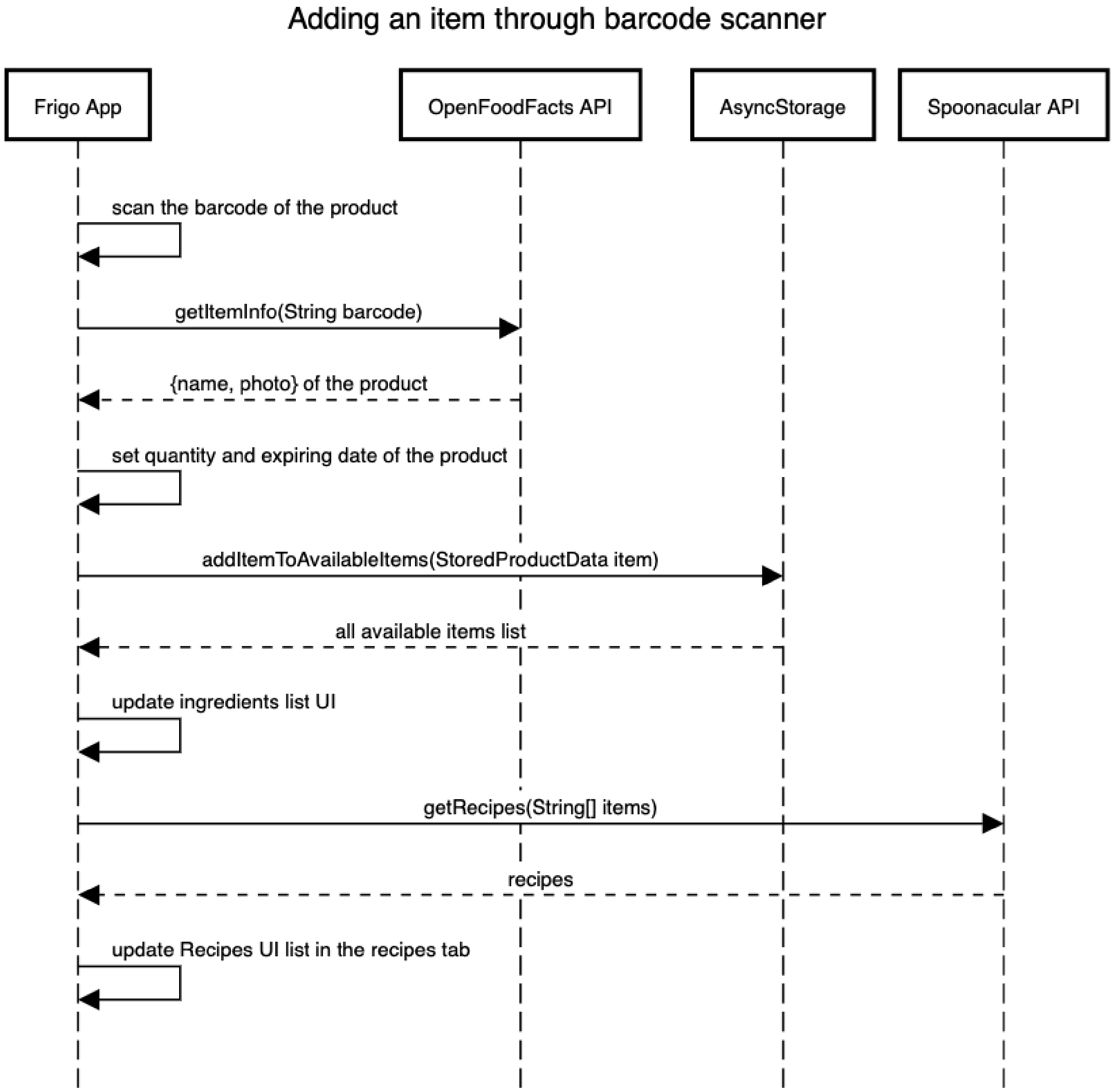
The whole project was handled through the Expo platform that facilitated the management of packets, their download, integration in the project and the deployment of the app on devices at testing time.

2.2 Architecture

The functional side of the app has been realized by making use of three main elements:

- **AsyncStorage:** an asynchronous, persistent and key-centered storage system that is global to the app. It is part of the react-native main libraries and it is used to store and retrieve the information about the ingredients entered by the user or obtained through the barcode scanner. All the items info is formatted through the StoredProductData data type and inserted into an array that gets saved in persistent memory.
- **Open food facts API:** this API handles the retrieval of ingredients information when the user inserts one through the barcode scanner. It takes the code of the ingredient as parameter and if it corresponds to a product in their database the API will answer with all the product related information.
- **Spoonacular API:** through this API it has been possible to retrieve all the recipes shown in the application. It takes the list of ingredients as parameter and returns a list of 10 recipes that maximize the number of used ingredients among the ones indicated. To minimize the amount of requests sent to the API the “complexSearch” endpoint has been used, which directly returns all the useful data.

2.3 Interaction sequence diagram


Adding an item through barcode scanner
Frigo App
OpenFoodFacts API
AsyncStorage
Spoonacular API
scan the barcode of the product
getItemInfo(String barcode)
{name, photo} of the product
set quantity and expiring date of the product
addItemToAvailableItems(StoredProductData item)
all available items list
update ingredients list UI
getRecipes(String[] items)
recipes
update Recipes UI list in the recipes tab

2.4 State management

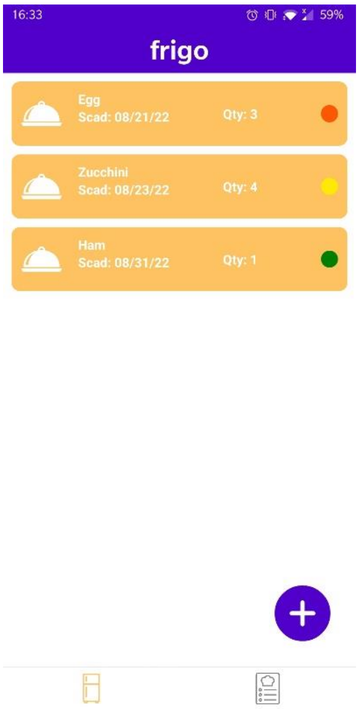
The app uses a central state, implemented through the useContext React Hook.

We always keep a single source of truth for the available items list in the central state, so at every change of this list (when adding, editing, removing an item) all the screens update correctly. The first tab screen updates the UI to show the updated list of available items and the second tab screen updates the list of recipes available after the changes in the items list.

2.5 User interface

The app has been designed starting from a BottomTabNavigator that handles the shift from the main screen, used for the ingredients list, and the recipes screen. From both, the navigation stack develops going to the ingredient insertion screen from the former and the recipe detail screen from the latter. The layout has been realized both for smartphones and tablets to make the experience more enjoyable and responsive on all devices. As a plus, two different colour themes have been included: a light one, more colourful, and a darker one for users that prefer a less flashy look.

2.5.1 Home screen



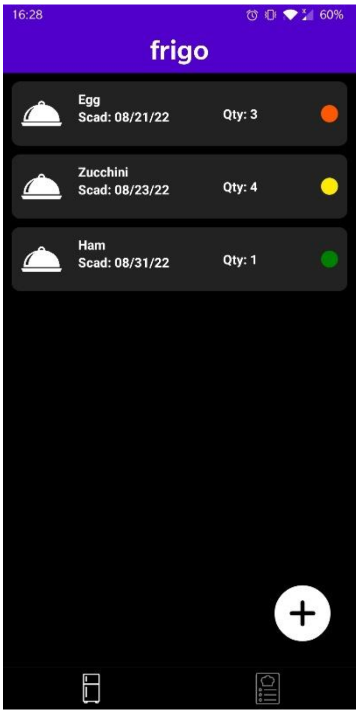
16:33 59%

frigo

Egg
Scad: 08/21/22 Qty: 3

Zucchini
Scad: 08/23/22 Qty: 4

Ham
Scad: 08/31/22 Qty: 1



16:28 60%

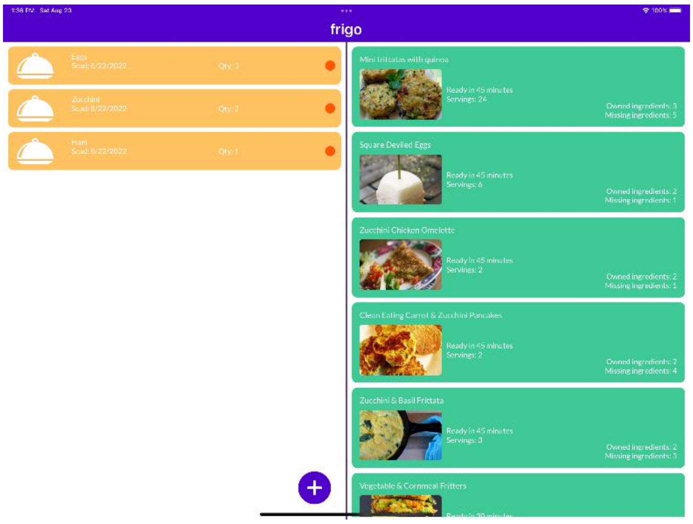
frigo

Egg
Scad: 08/21/22 Qty: 3

Zucchini
Scad: 08/23/22 Qty: 4

Ham
Scad: 08/31/22 Qty: 1

In the home screen it is possible to scroll through the virtual fridge and observe all the items added by the user, their name, the expiration date and the quantity or press the add button to insert a new ingredient. As a visual aid, it has also been added a coloured point at the right end of each product block. It represents the number of days between now and the expiration date of the product in question: it is green if the product is still some days away from expiring, yellow if the expiration date is between 3 and 5 days from now and orange if it is less than two days away. It finally becomes red when the product has expired.



1:56 PM Sat Aug 20 100%

frigo

Eggs
Scad: 8/22/2022 Qty: 3

Zucchini
Scad: 8/22/2022 Qty: 2

Ham
Scad: 8/22/2022 Qty: 1

Mini frittatas with quinoa
Ready in 45 minutes
Servings: 24
Owned ingredients: 3
Missing ingredients: 5

Square Deviled Eggs
Ready in 45 minutes
Servings: 6
Owned ingredients: 2
Missing ingredients: 1

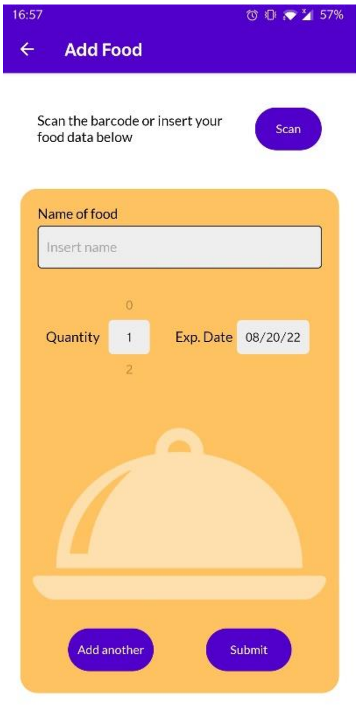
Zucchini Chicken Omelette
Ready in 45 minutes
Servings: 2
Owned ingredients: 2
Missing ingredients: 1

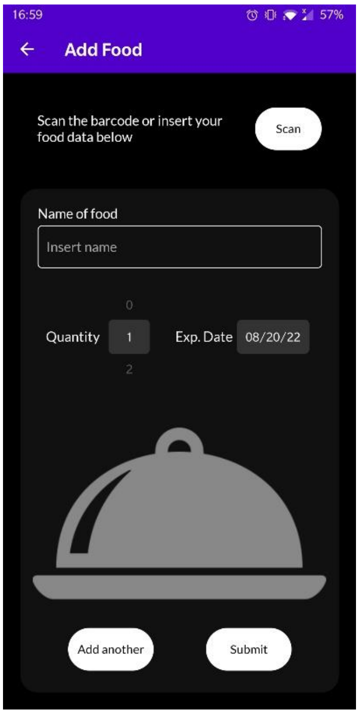
Clean Eating Carrot & Zucchini Pancakes
Ready in 45 minutes
Servings: 2
Owned ingredients: 2
Missing ingredients: 4

Zucchini & Basil Frittata
Ready in 45 minutes
Servings: 3
Owned ingredients: 2
Missing ingredients: 3

Vegetable & Cornmeal Fritters
Ready in 30 minutes

2.5.3 Food insertion screen

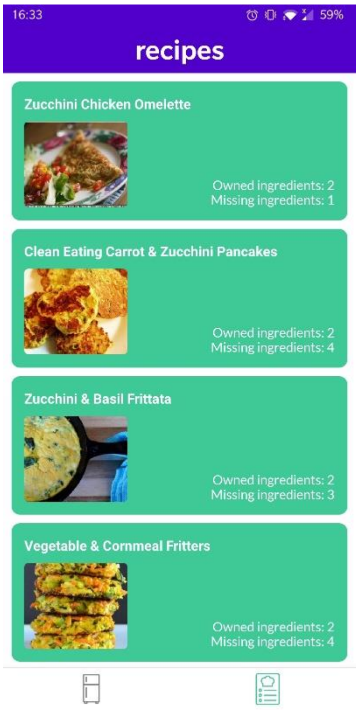


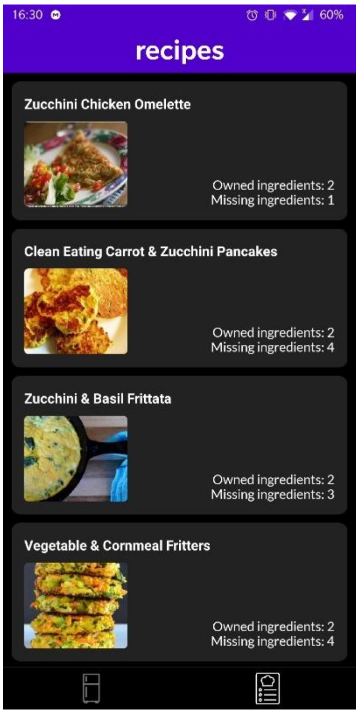


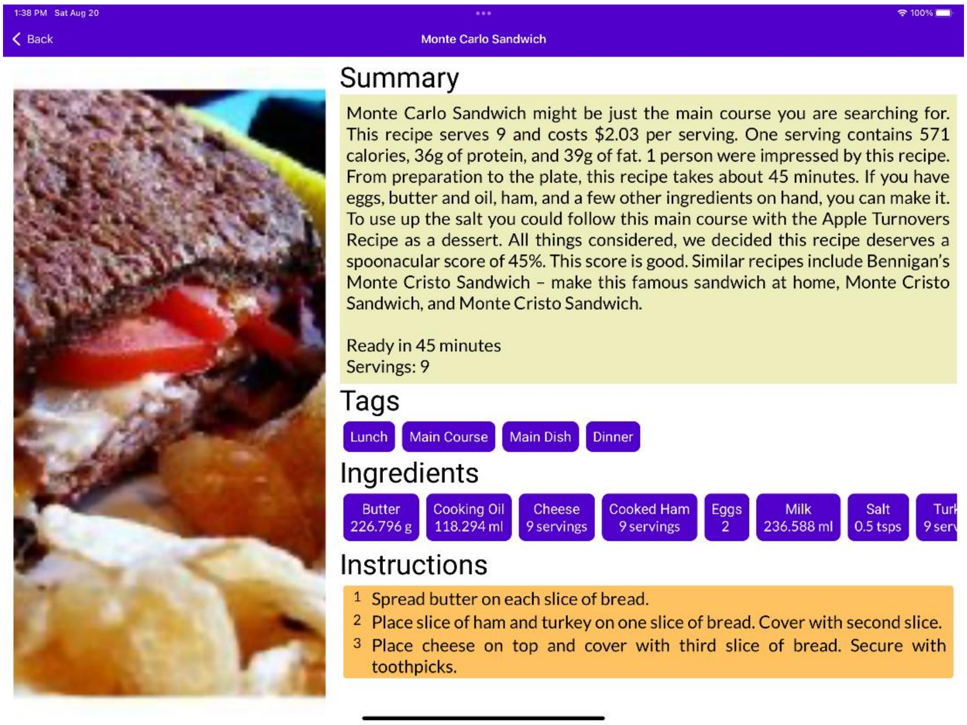
In this screen it is possible to specify all the ingredients details like its name, quantity, expiration date and image. It is possible to resort to the barcode scanner to retrieve the name and the image of the product through the Open food facts API in a more straightforward way. The user can also tap on the image to take a picture of the product in question or to modify the already inserted one, otherwise a default one will be used.

2.5.2 Recipes screen

The recipes screen displays the recipes suggested by the app based on the ingredients inserted in the virtual fridge. For each recipe it will be shown the number of ingredients that are present in the fridge and the ones that are missing. To try to consume first the products that are expiring the app will suggest recipes that consume those ingredients and only in case there are none it will suggest recipes containing the other items.







Summary

Monte Carlo Sandwich might be just the main course you are searching for. This recipe serves 9 and costs $2.03 per serving. One serving contains 571 calories, 36g of protein, and 39g of fat. 1 person were impressed by this recipe. From preparation to the plate, this recipe takes about 45 minutes. If you have eggs, butter and oil, ham, and a few other ingredients on hand, you can make it. To use up the salt you could follow this main course with the Apple Turnovers Recipe as a dessert. All things considered, we decided this recipe deserves a spoonacular score of 45%. This score is good. Similar recipes include Bennigan's Monte Cristo Sandwich – make this famous sandwich at home, Monte Cristo Sandwich, and Monte Cristo Sandwich.

Ready in 45 minutes
Servings: 9

Tags

Ingredients

Instructions

1 Spread butter on each slice of bread.
2 Place slice of ham and turkey on one slice of bread. Cover with second slice.
3 Place cheese on top and cover with third slice of bread. Secure with toothpicks.

3.0 Testing

The app has been tested during the whole development process: particular attention has been given to the correctness of the interface during manual testing whilst the main functions implementing the logic of the app have been tested using Jest framework.

In particular, the following units have been tested:

- AsyncStorage initialization
- Functions handling expiration dates and calculation of days left till expiration date
- Adding an item
- Editing an item info
- Removing an item

The whole app has also been distributed to a small number of people to test its functionalities and use in day to day life and gain critics.

4.0 Conclusions and observations

All the wanted functionalities of the app have been implemented and work as intended. The app does its job of memorizing ingredients and suggesting recipes to the users based on those. Its ease of use and clear interface make it appealing to the eye and helps people to get the most of it at a single glance.

Some more elements could be added to it in the future, like notifications for expiring products, multiple language support and a cloud service to save people's data into online accounts to make the app more portable from one device to another. As a first release, there was no particular need for the latter but it could be implemented in future versions to make the app more useful and accessible.