ShopAssist Mobile Application

CM3050 Mobile Development

[Expo Link for the project]

[Youtube link with the demo presentation]

Student: Cristina DeLisle
Student number: 190390475

Table of contents

Expo link for the project:
https://snack.expo.dev/@redchrision/shopassist

1. Concept development

The ShopAssist mobile application aims to be a companion for users that shop and need a shopping list with the price added next to the items. As a base, it represents a productivity application and the concept is evolved towards the shopping theme. The most common use case for ShopAssist is having the user in a store, before going to the counter and using the application for inserting items one by one, while also keeping track of their price. The advantage of this method is that the user has the possibility:

- To know in advance what is the total amount of the items to buy
- To have an overview of the shopping items

The area of user needs can expand to various other ways in which the application ShopAssist could be used. However, as concept development, the idea of this application came with the specified use case in mind.

As a productivity application, the ShoppAssist product has the following objective:

- Improving the shopping experience of users by providing them with the relevant information in order for them to make the necessary decisions about their purchases.

2. Wireframing

2.1. Visual interface of ShopAssist

Shop assist has two screens which display to the user the following elements:

- A main screen with all the items displayed as presented in Figure 1.
- An Item manager screen that is presented in two modes:
    - An Add item mode, which is displayed in the wireframe from Figure 2.
    - An Edit item mode, which is shown in the wireframe from Figure 3.

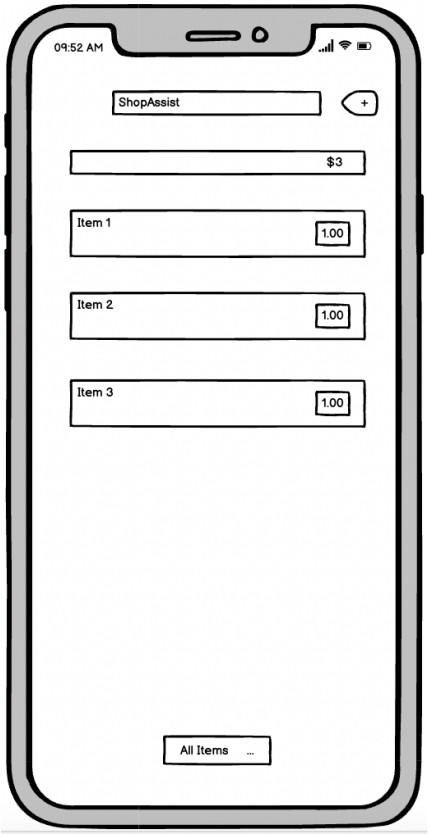


Figure 1


09:52 AM
Add Item
Your Item
Amount
$
Description
Item Description
Cancel
Add

Figure 2

Figure 3

2.2. User flow diagram

For showing the ways in which the user can navigate in the application and access different elements, a user flow diagram was created in Figure 4.

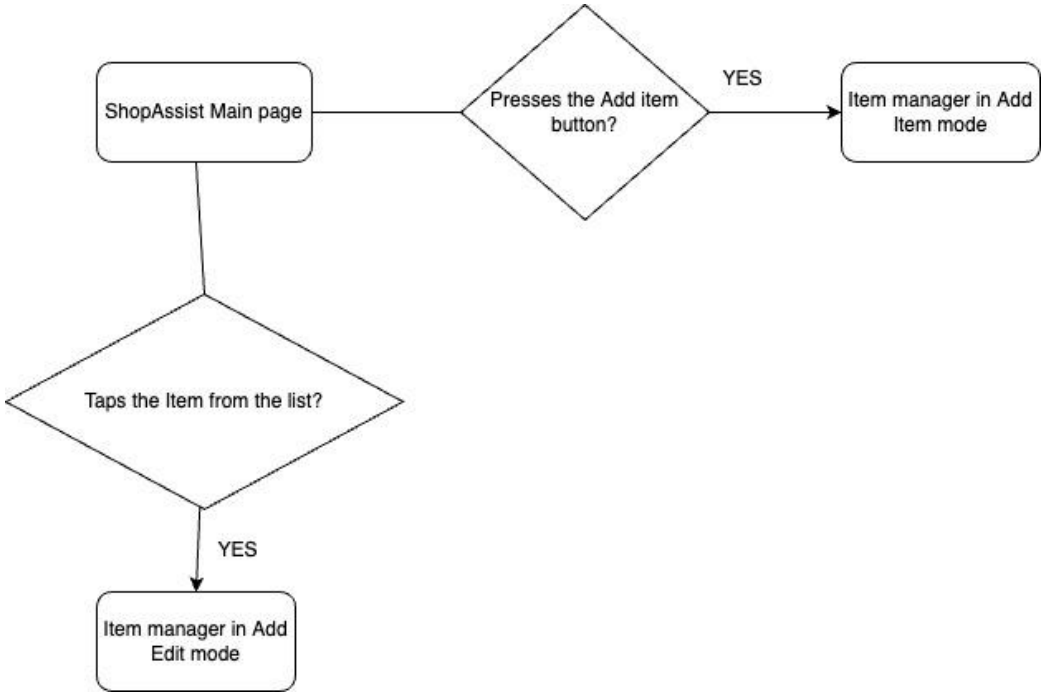


Figure 4

Here we can see that from the ShopAssist main page, the user can access for the following pages:

- By clicking the Add item button from the top of the page, the user is taken to the Item manager page which is displayed in Edit mode;
- By clicking on one of the items already present in the list, the user gets to the Edit mode (please note that the Figure 4 has an error - instead of ‘Add Edit mode’ it should be ‘Edit mode’).

3. User feedback

In order to test the application from an user perspective, so far the following methods were conducted:

- Testing them on IoS and Android platform in terms of main functionalities. The project is running on both platforms and also on Expo / Snack, without errors. These user tests were performed by:
    - The developer of the application who had access to the source code on the local environment, by using IOS and Android simulators while doing the programming work.
- User feedback gathered for the prototype version, by having three users test the application on the local device and answer:
    - a short questionnaire about its usability. The scales used for this were the ones from the System Usability Scale (SUS) Score Calculator, accessible here:

https://stuart-cunningham.github.io/sus/ The project overall received a score of 94% , which is explicable considering the simplicity of the application.

- Interviewing the three respondents to the previous survey and getting more user feedback directly from them, by addressing open questions about their user experience.

4. Prototyping

As seen in Figure 5, the prototype version of the application has the following features on the main screen:

- Add items button on top, on the right, which is a customized button.
- Calculation of the amount of the items displayed.
- Displaying the items which were previously inserted in the application.

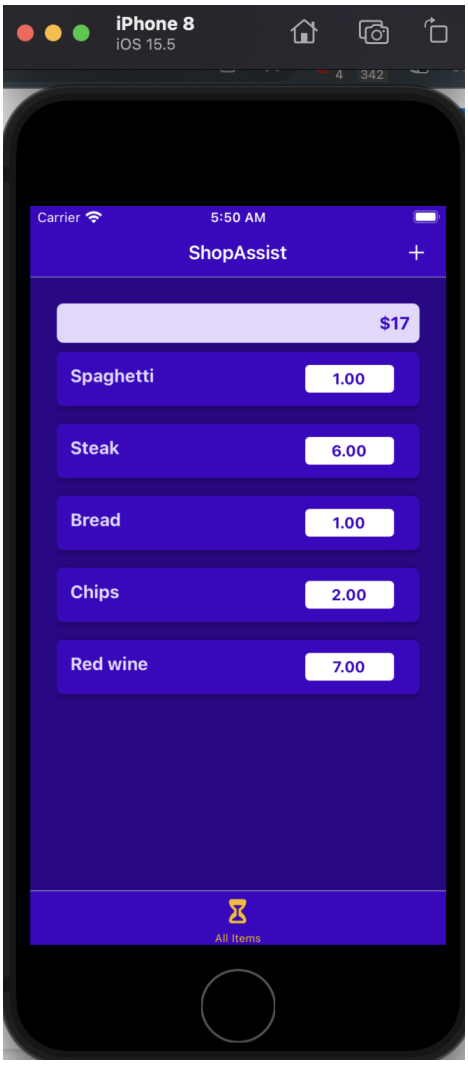


Figure 5

As seen in Figure 6, the Item Manager page is presented in the Add item mode. Here we see displayed:

- A form with a title, an amount field and a description field. All fields are configured based on their specific needs.
- A button for Adding the item and another for Canceling the insertion.

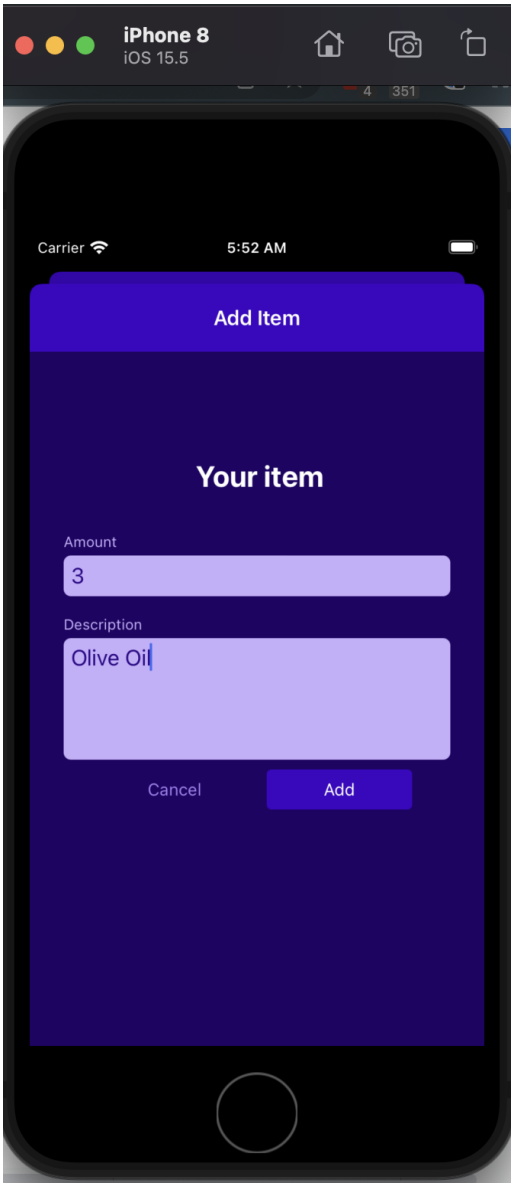


Figure 6

In Figure 7, we notice the same Item manager page, this time displayed in Edit mode. It consists of:

- The same form, having an amount and a description.
- The same Cancel button as in Figure 6.
- A button for Updating the item with the new information displayed.

- A delete button, which is accessible only from this place in the application. This was a user experience design choice, in order to make sure that the user is not tapping by mistake on the main page and deletes an item from there.

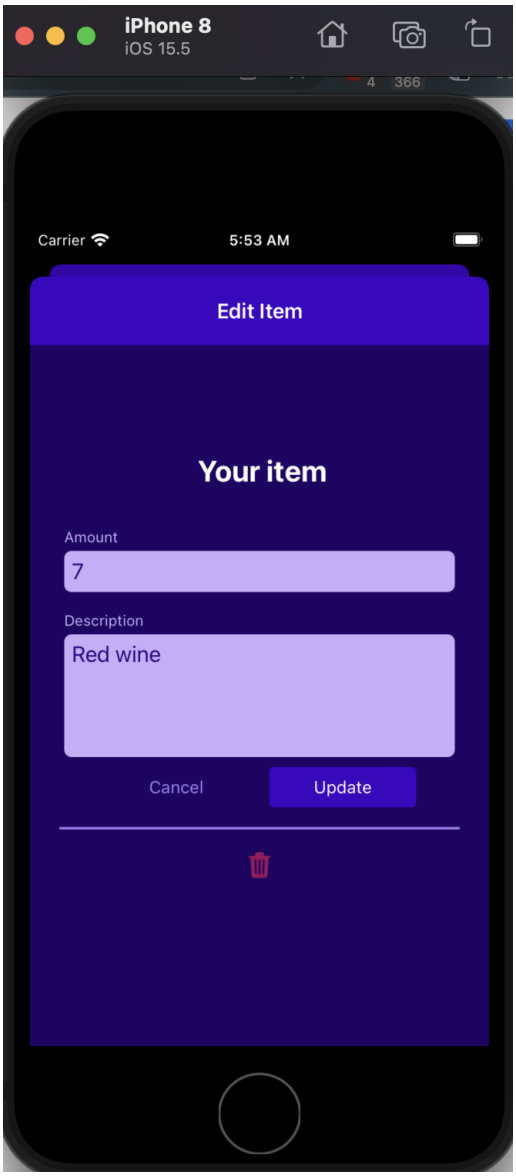


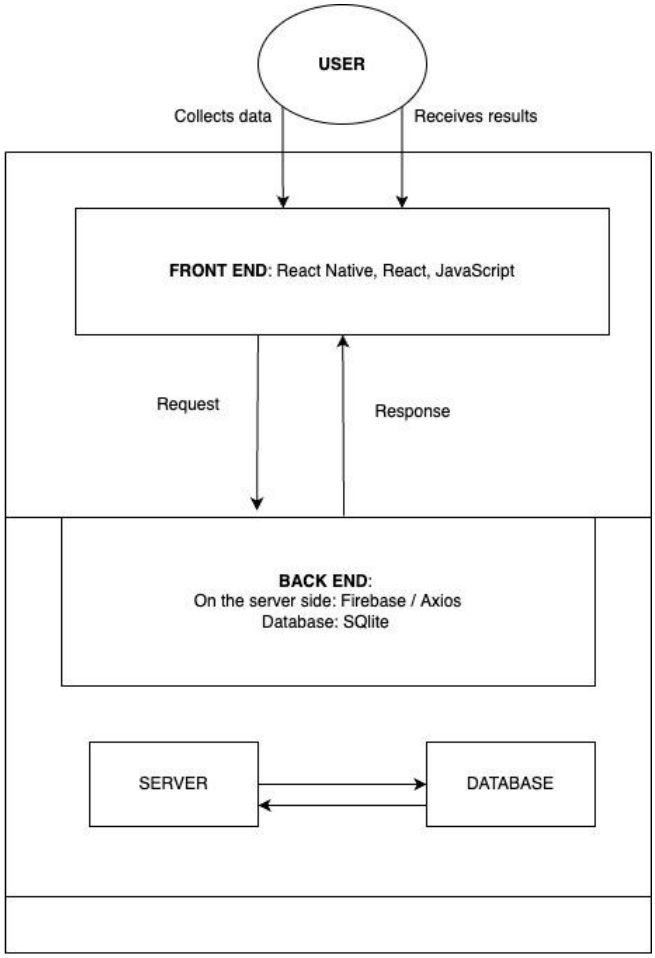
Figure 7

5. Development

From the perspective of the development of the application, we can see in Figure 8 an overview of the technologies used in ShopAssist, with the mention that the backend was only created on

local environment and is not accessible from the submitted files presented in the application. The diagram presents a best case scenario of an application that is connected to a server and a database.

On the front-end side, the application makes use of React Native mainly, with elements also from React and Vanilla JavaScript, as a base.



ShopAssist architecture

Figure 8

The application is using elements from React Native in terms of navigation, like the createBottomTabNavigator component. We can also notice the GlobalStyles constant which is exported in multiple places, in order to apply an uniform styling in the application, in a more effective way. The DUMMY_DATA inserted in the application is meant to offer to the user a more visual display of how the items would look if the application would be connected to a real server - by remaining on the screen after the application is closed.

In the ItemManager.js file, we can notice that the Add Item and the Edit modes for the Item manager page is being activated from there, by using isEditing and detecting if the user is editing or adding a new item.

The application is using the Context API in order to avoid passing props in a manual manner at each necessary time in the application. The main file which is carrying on this overall management of the data is items-context.js. The functions take into consideration the various cases relevant in the application: adding, updating and deleting the item data.

In the form present in the folder components, in the ItemForm.js file, we can see that the elements of the form have specific characteristics. The description is a text input which can be added on multiple lines, for example. Also, we can notice the visual validation elements added to the Item manager page. If the user is adding wrongly the amount or the description, then error messages will be displayed and also color visual elements will be activated in order to point out to the user that a change in the input is required in order to add / edit the item.

6. Unit testing

In terms of unit testing, this was performed on the local environment using Jest and aimed to test the main functionalities of the application and make sure all tests are passing.

7. Evaluation and conclusion

The evaluation of the ShopAssist application can be done from the perspective of the objective of this project, which is to offer users a tool to accompany them in their shopping experience. This objective was achieved through the main features of the application, given that the adding an item, updating it, deleting it represent core components of a productivity application. Furthermore, the amount sum feature helps the user with a detail which otherwise would have had to be calculated differently.

In terms of further development for extending this application, I would connect it to a backend - a server and a database, outside the context of the local environment. More testing needs to be performed and currently the report lacks the relevant information about the actual unit tests carried out. These elements related to a server connection and performing testing took into consideration the limitations which might have appeared by using Snack. Currently, the project is running on all display / simulator platforms from Snack, which is a positive aspect.

The application can also be extended with adding authentication to it, for a more enhanced level of privacy / security protection. More features can be added, like the possibility to take photos and add the image in the form from the Item manager page.

8. Bibliography

1. Schwarzmüller M., 2022, React Native - The Practical Guide [2022], https://www.udemy.com/course/react-native-the-practical-guide/

2. BROOKE J., *System Usability Scale (SUS) Score Calculator*, https://stuart-cunningham.github.io/sus/