Department of Computer Engineering

Senior Design Project

Farketti

Analysis Report

Deniz Alkışlar | H. Buğra Aydın | M. Erim Erdal | M. Enes Keleş | Hakan Türkmenoğlu

Supervisor: Eray Tüzün

Jury Members: Mustafa Özdal and Özcan Öztürk

Innovation Expert: Barış Misman

November 12, 2018

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Content

1. Introduction

Finding the best price is always a daunting task. Comparing prices between stores is both time and energy inefficient especially on essential items such as food, cleaning products, and cosmetics. This problem highly affects lives of people with low income including students, homemakers, working class families, especially in countries that have high inflation which results market price fluctuations nearly on a daily basis. These people have to constantly track prices in order to cut off expenses. To this end they buy from gross markets and online stores, and collect discount coupons. Both of these solutions have downsides, collecting coupons is time and energy consuming and going to a gross market is not the best option in most times which we discuss in the following table.

We conducted a case study on grocery prices in order to see the price difference using a prepared list of most crucial items corresponding to basic needs such as provisions and cleaning products. When purchased from the overall cheapest grocery store, the total cost is 109 Turkish Liras. However when the cheapest option of every item is bought the total is 99 Turkish Liras which is about 10% cheaper than the former *(Table I)*

	Carrefour	**A-101**	**BIM**	**Akakçe**	**Cheapest option selected**
500g Spagetti	2.35 TL	1.15 TL	1.15 TL	3 TL	1.15
1 kg potatoes	2.79 TL	*	2.73 TL	*	2.73
1 kg tomatoes	6.95 TL	*	6.95 TL	*	6.95
1 kg onions	2.79 TL	*	2.98 TL	*	2.79
15 eggs	11.99 TL	8 TL	8 TL	10 TL	8
1 lt milk	2.75 TL	3.25 TL	3.25 TL	*	2.75
Price / diaper**	0.67 TL	0.36 TL	0.365 TL	0.342 TL	0.342
32x toilet papers**	31.90 TL	31 TL	31 TL	24 TL	24

Detergent / KG**	5.28 TL	4.18 TL	3.75 TL	*	3.75
Disher Capsule**	1.01 TL	0.79 TL	0.35 TL	0.83 TL	0.35
-TOTAL-	**169 TL**	**131 TL**	**107 TL**	**128 TL**	**99 TL**

Table 1. Price comparison between different markets
*note that missing prices are plugged with the price of the cheapest option
**prices are normalized for quantities

2. Current system

2.1 cimri/akakce

Website and mobile app which compares prices among many online stores and lists the best options for the user. It compares for one product at a time [1].

2.2 Basket

Mobile app which shows consumer prices in grocery stores. The application gathers data by crowdsourcing, namely its users entering prices of products into the system and validating these prices by scanning receipt barcodes [2].

2.3 Grocery Pal

Mobile app that points the user towards weekly sales at local supermarkets and discount stores, and matches shopping list items against products online with lower prices [3].

2.4 GroceryIQ

A mobile app application that allows the user to build shopping lists with features like predictive search and barcode scanning [4].

2.5 Bring!

Grocery shopping list app allows users to create, sync, and share shopping lists [5].

2.6 Favado

Mobile app showing sales and coupons on nearby stores. The user can collect

coupons by emailing to themselves and printing out [6].

3. Proposed system

Farkett is a cross platform mobile application which shows up-to-date grocery prices and recommends optimal routes for shopping lists. This task requires grocery price data, the majority of which will be collected by scraping online stores. However some grocery stores don't have online stores. Therefore we have another data collection method that is receipt scanning. In this method receipt information is recognized using OCR and consumer prices are extracted. To encourage users to scan receipts we consider giving rewards which will be determined later in the process. We aim to keep personalized receipt information in order to offer targeted advertisement.

The novel feature in our product is shopping route recommendation. Farkett scans prices in nearby grocery stores and offers optimized routes for shopping lists with criterias determined by user. This is not an easy task because sometimes the cheapest route is not the most desirable. Most people don't want to walk kilometers and visit many stores. Therefore Farkett takes distance, prices, and number of stores into account while offering routes. Furthermore, upon entering a store in the route Farkett will show the items to buy in that store, for convenience. Shopping route recommendation will be customizable to user's will, namely range and product criterias can be selected. This makes Farkett a useful tool for many people.

3.1 Functional Requirements

- User can see detailed information (price, location, category, etc.) about products that we store in our database.
- User can filter products according to their preferences.
- User can get shopping route recommendations by entering a shopping list which will be modified in real time.
- User can scan shopping receipt to remove purchased items from shopping list.
- The application shows user which items to buy upon entering a store.
- The application notifies users about discounts and markups.

- The application gives awards to users who scan their receipts.

3.2 Nonfunctional Requirements

- **Usability:** The application must be easy to use for all age groups.
- **Scalability:** The application must serve up to millions of users during its runtime and it must accommodate the growth of its database.
- **Response time:** The application must respond quickly - 300 milliseconds at most - to receipt scan so that the users would not wait for long durations.
- **Data integrity:** The product data must be accurate (90%) with the real markets.
- **Availability:** The product data must be accessible when required for use.
- **Robustness:** The application must cope with the errors during its runtime and in case of a failure no data should be lost.
- **Extensibility:** The application must be extensible such that it can handle adding a new functionality or modifying an existing one.
- **Security:** The application must secure user data from potential external threats and it must detect frauds both in receipts and user profiles.

3.3 Pseudo requirements

- The application must operate on existing iOS and Android smartphones.
- The application must use Google's Directions, Places, Cloud Vision API, and OpenCV.
- The application's server side will run on AWS and developed by using Java.
- The application's client side will be implemented by using React Native.

3.4 System models

3.4.1 Personas

3.4.1.1 Enes

At the age of 43, Enes likes to enjoy the luxury in life: expensive travels, wine tasting, gourmet foods, luxury sports cars. He is very busy because of his overloaded work schedule and meetings which makes online shopping a necessity even if he does not prefer to do so. He holds a MBA Degree and is currently owner of a high end company.

3.4.1.2 Emre & İrem

Emre and İrem is married couple with no kids because of career concerns. They are very well-educated and well-compensated for their age. They do not have much time because of their focuses on their careers.

3.4.1.3 Buğra

Buğra is a 21 year old college student. He stays in rent and tries to keep ends meet from month to month, doing careful spendings. He has no income and compensated just enough by his father. Being a student at a demanding college, he is almost always overloaded by the workload of the classes thus he has very little time and money.

3.4.1.4 Erim

Erim is earning a lower-income than other personas, married 37 living in a small apartment with his wife and 3 toddlers one of which is has recently started to a state school. He currently works in a blue-collar job. His wife is a housewife and earning money is hard for Erim and requires working long hour shifts with minimum wage. He has below average education since he never finished high-school. He needs discount shopping destinations to help him stretch his lower income such as Şok, A-101 and BİM. He has spare time for finding discounts when he is not working.

3.4.1.5 Sena and Mehmet

They are an elderly couple who has retired for a long time now. They have 3 grandkids from their 2 kids which they actively spare time to be with them for rather slow activities. Their credit card spending is high. They live together without any other family members. They have a lot of free-time but not much energy which restricts the activities they are able to do.

3.4.1.6 Ersan and Seda

Seda is currently a teacher and Ersan is a retired officer. They live in rural areas of the country. They find it hard to move to another place because they have long roots in these rural areas. They have 2 children which they are trying to give the best education opportunities they can by sparing money for their education, but is nonetheless limited because they do not have a very high income. They regularly do religious activities and define themselves as such.

3.4.2 Use case models and Scenarios

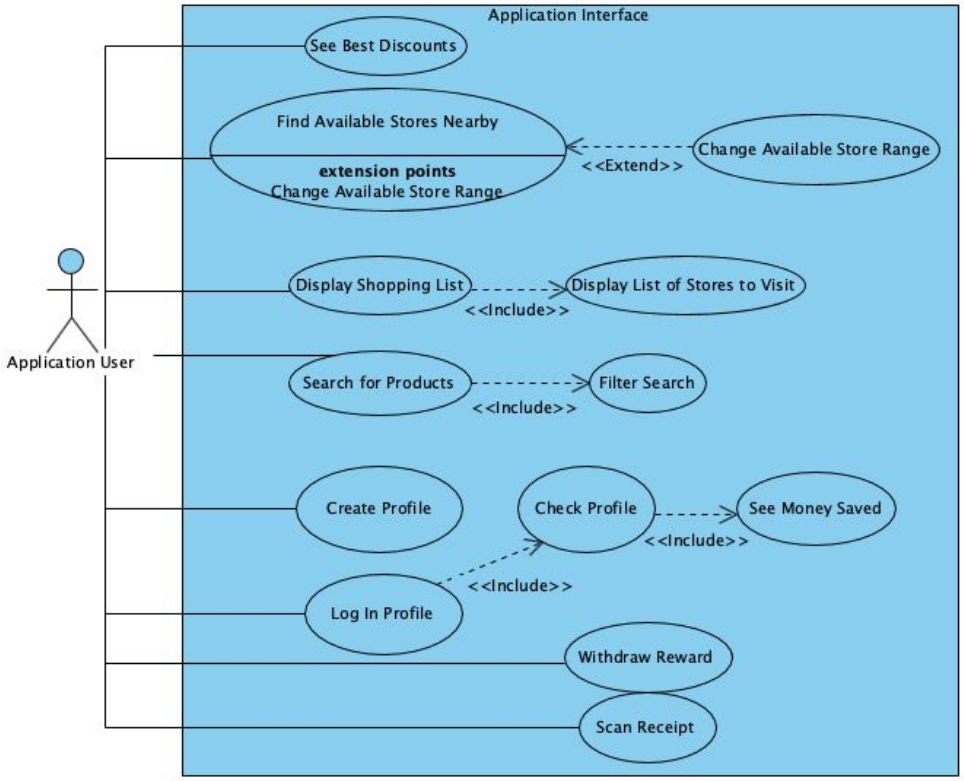

Figure 2. General use case model

3.4.2.1 Scenario 1

Sena and Mehmet are not really fancy phone users but they both own a smartphone so their children downloads the app for them and creates and account. Sena and Mehmet try to use the app and because of the simple UI, they are able to create a small shopping list and they search for the discounts after creating the shopping list. They are amazed by some discounts and they are in need so they also search for the nearby markets where these discounts exist. They can only use the simplest form of the app so they do not filter products, they do not widen the range of stores and they do not scan their receipts. They also do not create user accounts.

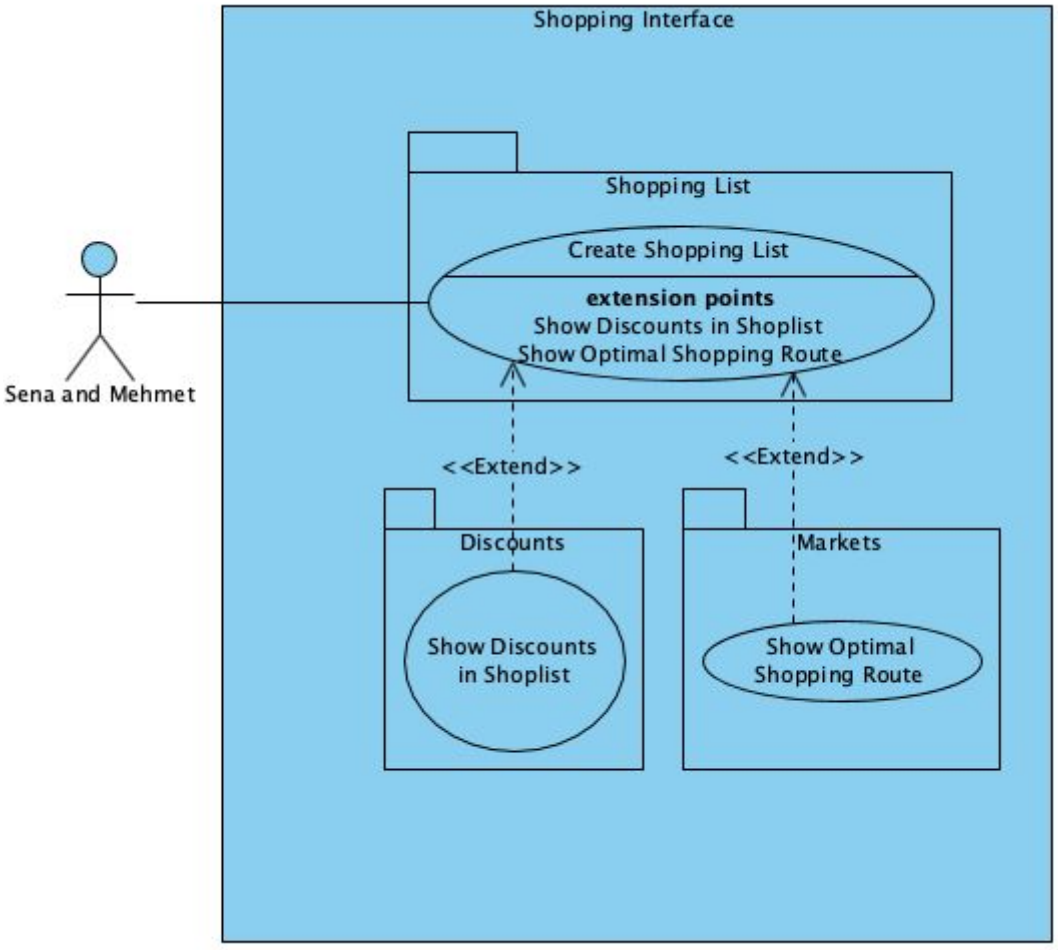

Figure 3. Use case model for Scenario 1

3.4.2.2 Scenario 2

Ersan and Seda are trying to adjust their budget this month and they understand from technology like an average middle-aged. They register to the system by creating an account and since they live in rural area they see there is no nearby store available in close range. So they widen the range and they find available stores. They fill their shopping list and use filters for several products. Finally they check for the discounts. They will later buy from the stores that has an online market available.

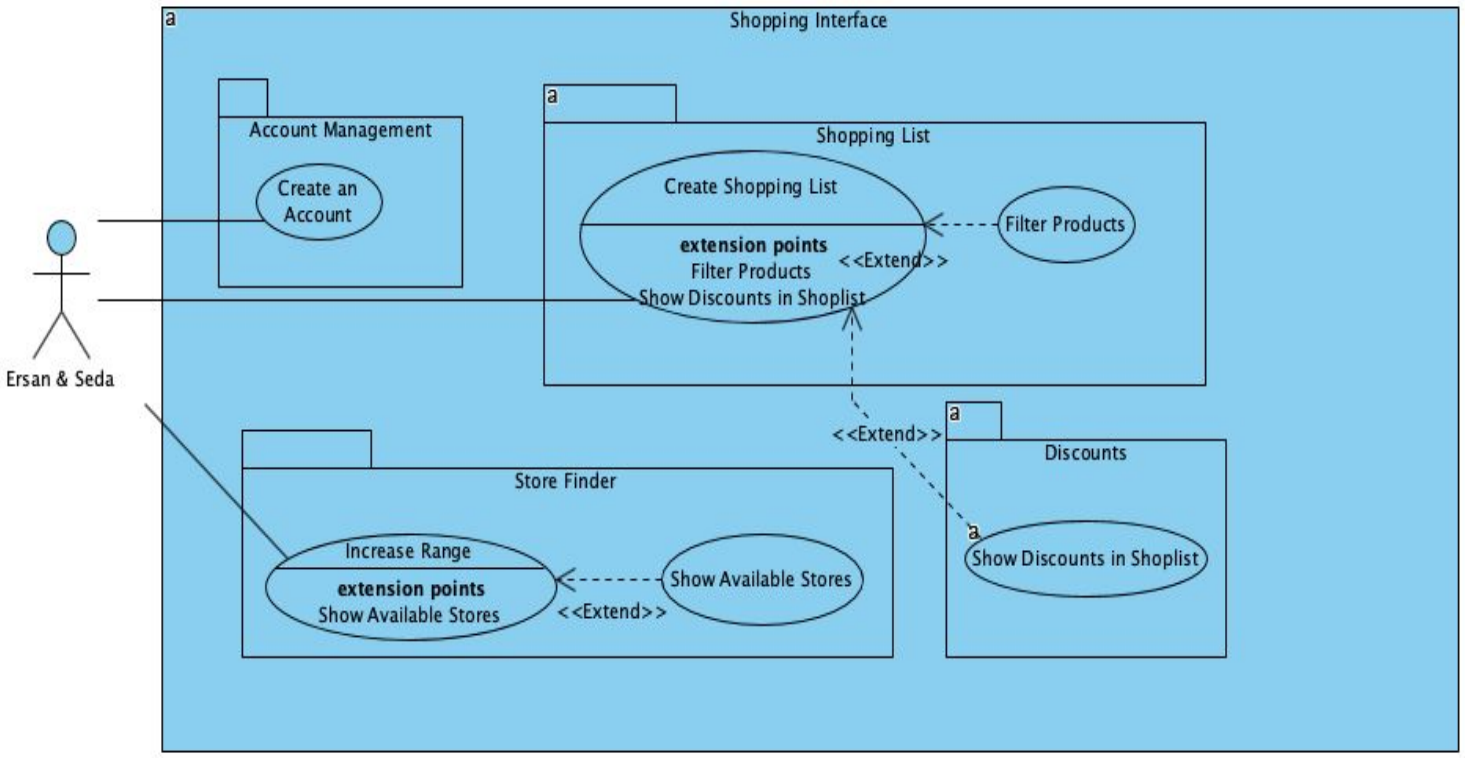

Figure 4. Use case model for Scenario 2

3.4.2.3 Scenario 3

Erim is into technology and he has found this app from an advertisement. He had already stocked some receipts recently for tracking his spendings and he wants to check if the system really works and if he will actually win some rewards for it. He creates an account to collect the rewards and he goes directly to scanning receipts section. He scans several receipts some of which do not generate any rewards sadly but one of the receipts was a rare one hence it gives him reward. He tries to withdraw the reward but application warns him that he can only collect his reward after collecting a specific amount of reward which is not yet sufficient for him to collect.

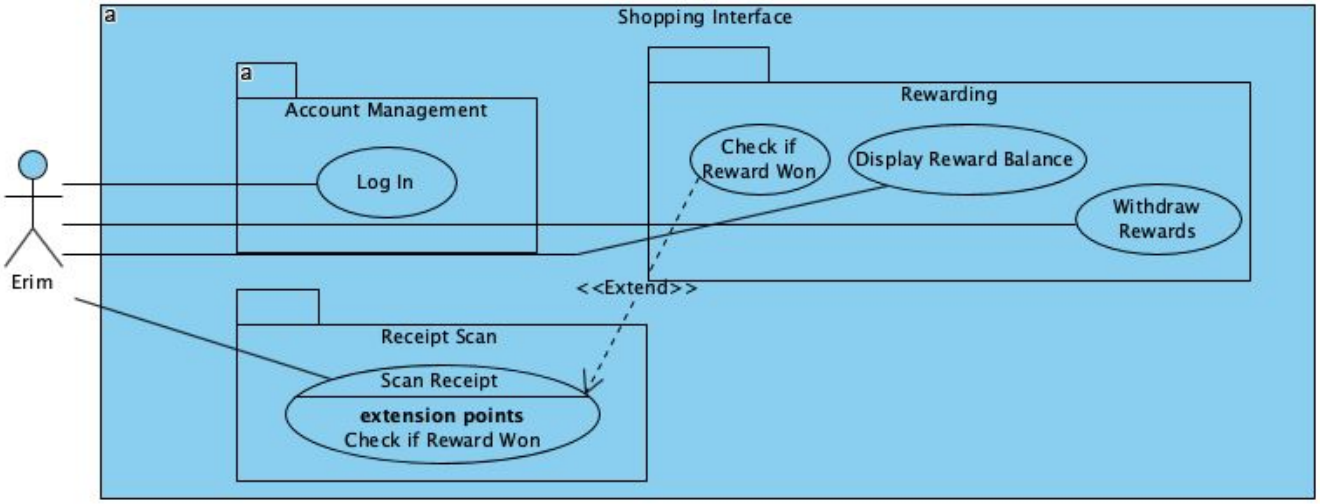

Figure 5. Use case model for Scenario 3

3.4.2.4 Scenario 4

Buğra has some spare time after a long day of hard work and wants to travel the optimal route of the app to gather the cheapest possible groceries for his kitchen. He wants to find the route fast so he does not have to create a user yet, he will create a shopping list and check the optimal routes and what would be the best transportation usage to efficiently gather all the groceries.

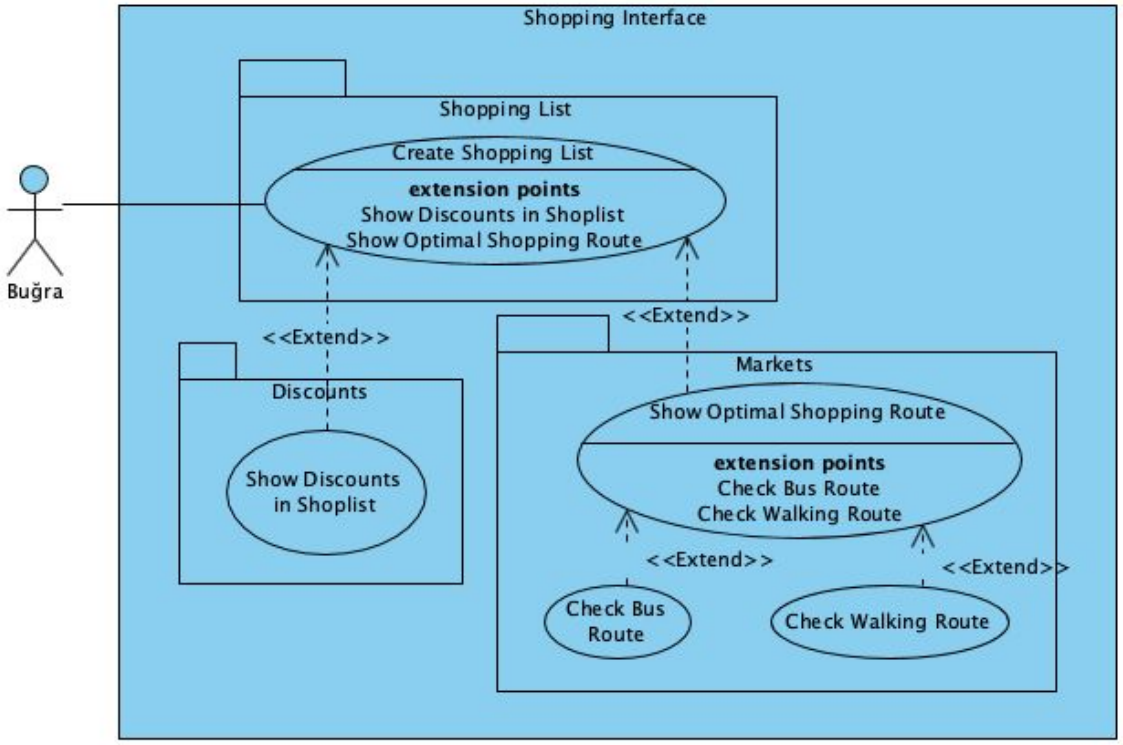

Figure 6. Use case model for Scenario 4

3.4.3 Object and class model

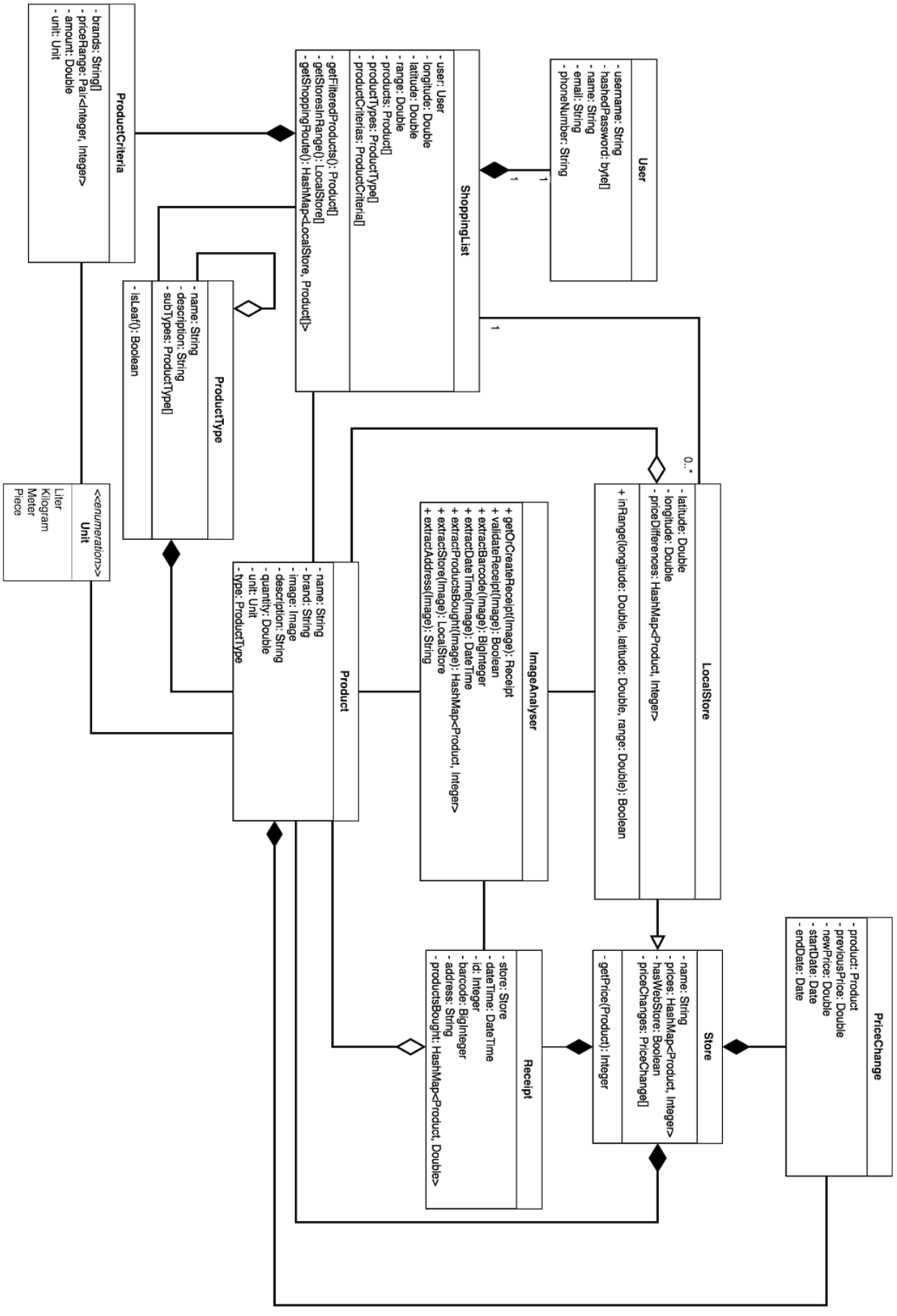

Figure 7. General Object-Class model for the application

3.4.3.1 User

Represents a single user's information.

3.4.3.2 Product

Represents a single product with name, brand, quantity, unit etc.

3.4.3.3 Unit

An enumerator for representing measurement units.

3.4.3.4 Store

Represents a grocery store franchise. Holds all relevant information such as products and prices, price changes, if the franchise has a web store.

3.4.3.5 LocalStore

Represents a local instance of a franchise. Holds coordinates of the local store and price differences with the main franchise. (In some franchises, prices differ depending on the area for instance fish is cheaper in stores that are close to seaside.)

3.4.3.6 ProductType

This is a Tree Node-like class that will be used to represent the category tree of products.

3.4.3.7 ProductCriteria

This class represents criterias for a single product. For every product in a shopping list user will be able to determine some criterias such as brand, price range, etc.

3.4.3.8 ShoppingList

This the major class in our system. Represents a shopping list, coming from a user, with products and product criterias. Major functionality is kept in this class - finding nearby stores, route recommendation, and filtered products.

3.4.3.9 PriceChange

This class represents discounts and markups.

3.4.3.10 Receipt

This is keeps receipt information which is parsed by ImageAnalyzer class.

3.4.3.11 ImageAnalyzer

This class is used to scan a receipt image and to extract information from it. To scan receipts it will send HTTP requests to Google's Cloud Vision API.

3.4.4 Dynamic models

3.4.4.1 Sequence Diagrams

Scenario 1:

After logging in to the system, the application greets the user with shopping list and remembers the items actor added last time. In this case the actor has moved to a remote place for a temporary visit which results in no stores available within the actor's last saved preferred range. Upon the application's warning, the actor widens the preferred range where it collects 7 stores nearby. The actor continues by adding "1 lt milk" and "2 cardboard of eggs" to the list. The application automatically deduces what these products are, including their specific criterias (unit, quantity, preferred brand etc.). Afterwards the application receives an optimal route from the server for each product. Using this optimal route the application retrieves the best price available from the stores. Then, user presses "Start Shopping" button and the application switches to the navigation screen. Navigation screen helps user navigate to the stores. Upon entering a store, the application reminds the actor what products they should buy inside the store. After leaving the store, application crosses out (but not delete) the items inside the store, because they are assumed to be purchased.

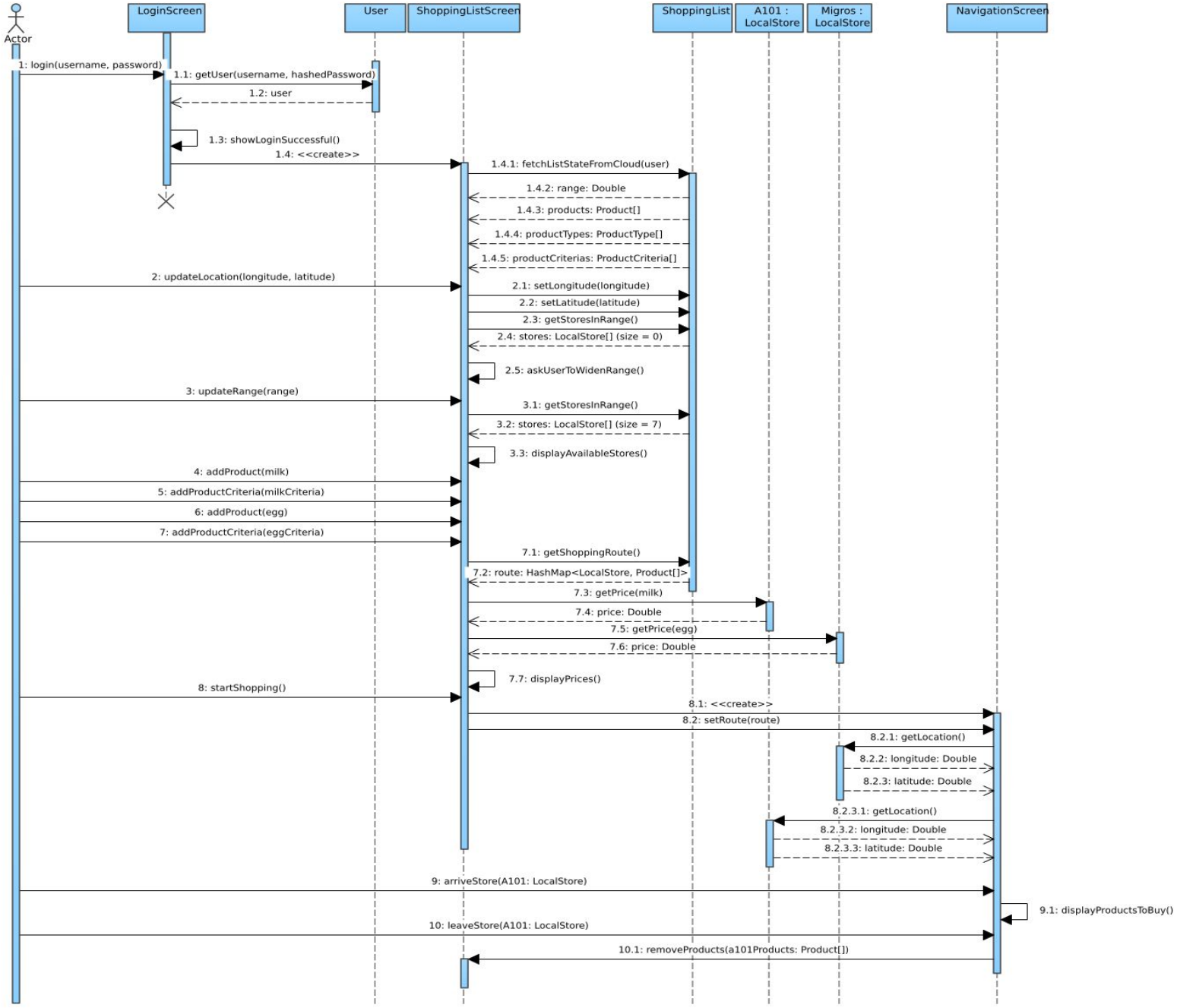

Figure 8. Sequence diagram of shopping with the application

Scenario 2:

After finishing their shopping, the actor opens the receipt scanning screen. In this view, the client application validates that the user is pointing the camera at the receipt visually clearly. After the client detects the receipt image realtime, it validates the receipt and if it succeeds, sends an HTTP request to further process the receipt to increase accuracy and fraud detection. Then a receipt object is created and stored in the database.

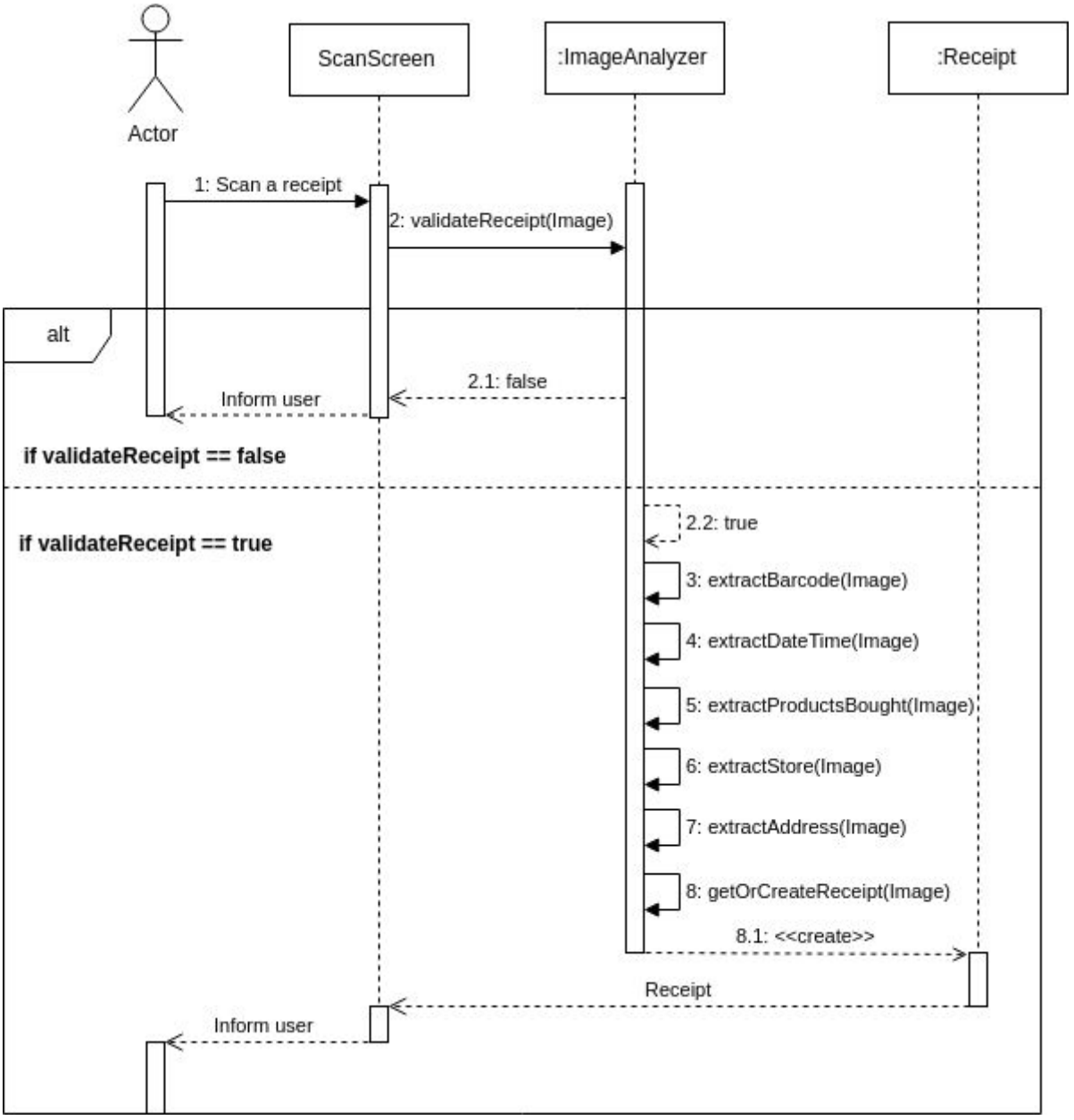

Figure 9. Sequence diagram of receipt scanning

3.4.4.2 Activity Diagram

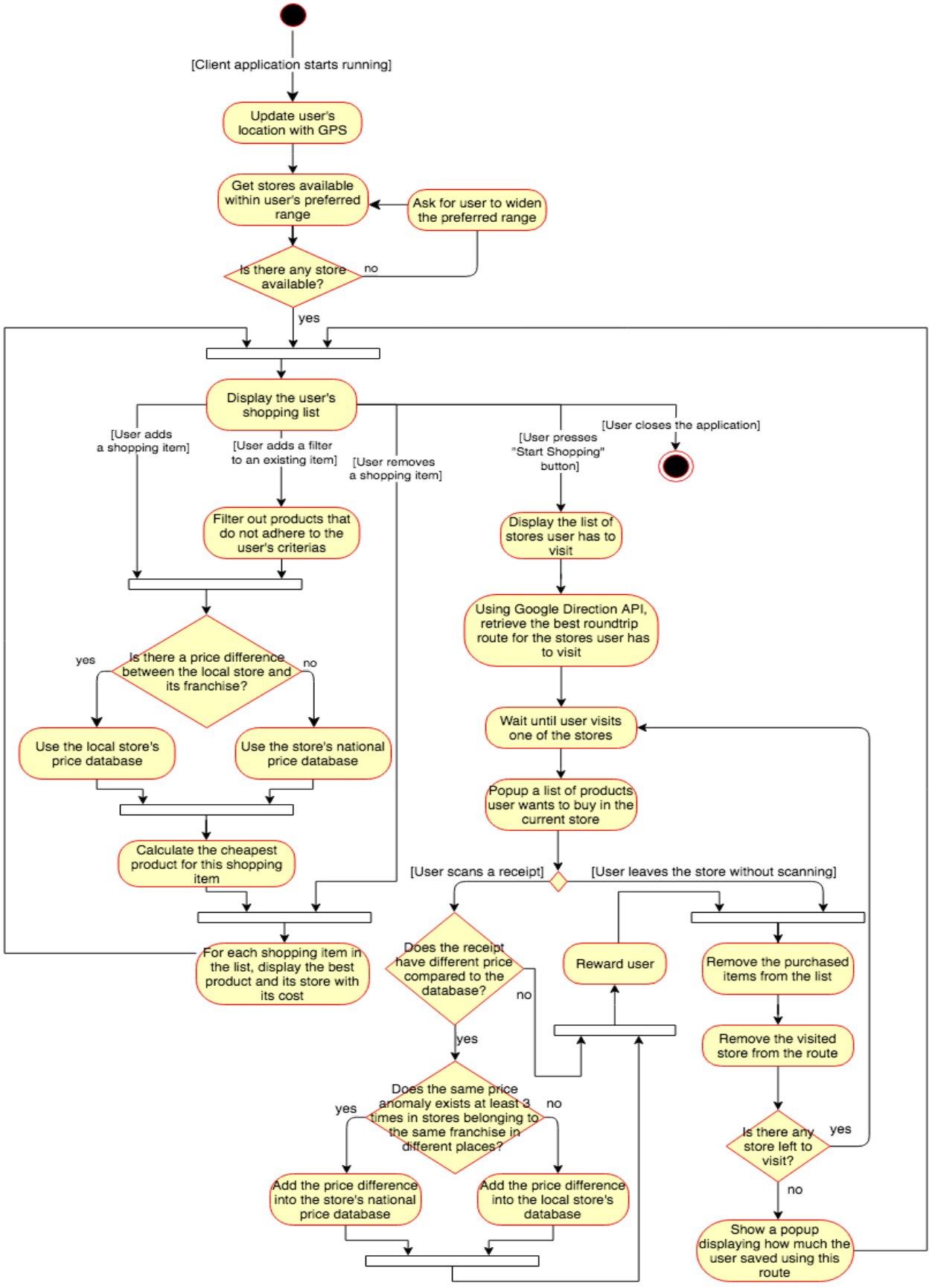

Figure 10. Activity diagram of shopping and receipt scanning system

3.4.5 User interface - navigational paths and screen mock-ups

Figure 11 illustrates the main navigational path of the user interface.

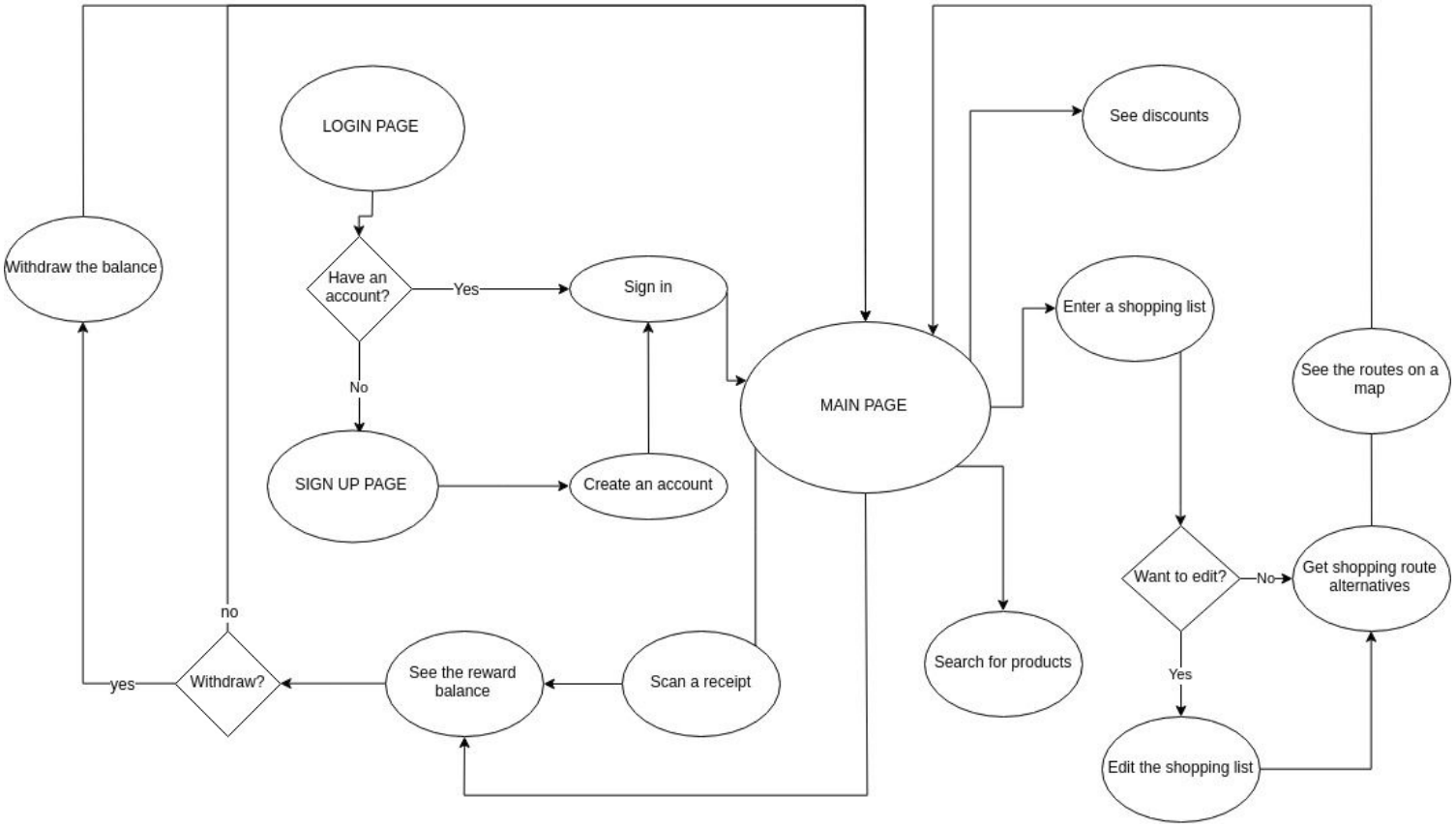

Figure 11. Main navigational path for client application

The figure 12 demonstrates the login screen. Users are able to enter their username and password and enter to their account. A button directs them to sign up screen if the user has no account. The figure 13 demonstrates the sign up screen. Users will be asked to enter their username, password, e-mail and their address. By swiping left, the sign up operation will be completed.

Figure 12. The login screen mock-up

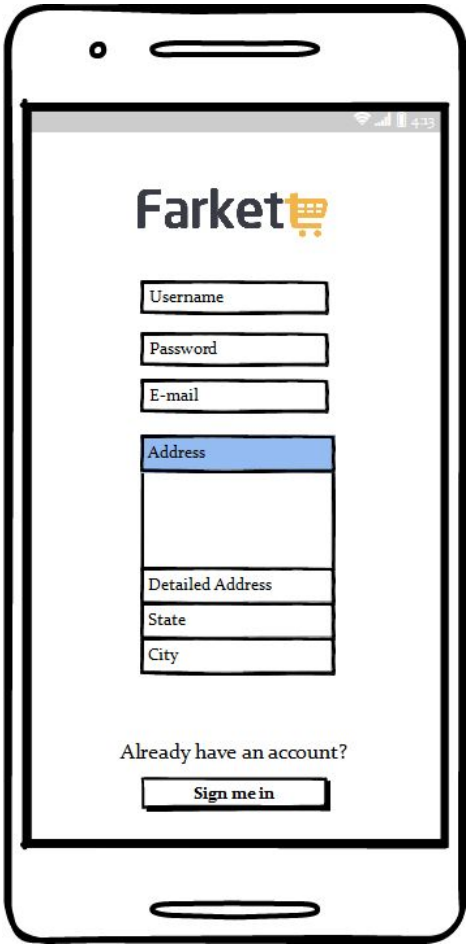

Figure 13. The sign up screen mock-up

The figure 14 demonstrates the main screen of the app. Users are presented with targeted advertisements according to their previous shopping lists. It is possible to swipe items up and add them to the shopping lists. It is also possible for users to click on the search bar as illustrated in figure 15 and the relevant items will take place of the targeted advertisement part. User are also able to obtain informations about the changes to prices of the products in the main screen. So a user will be informed about the sales of a product that he was not planning to add the list and the user will be notified about the markups to the products that is planned to be added to the list. User are also provided with the option to logout and redirected to the login page.

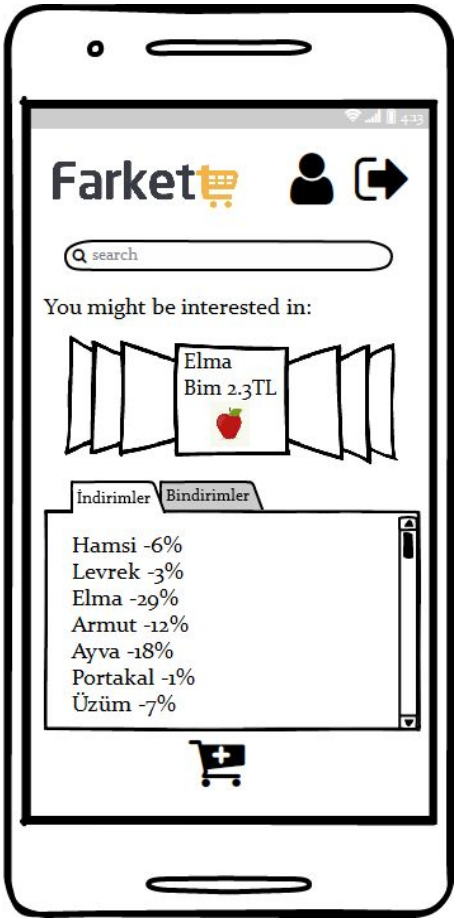

Figure 14. Main screen mock-up

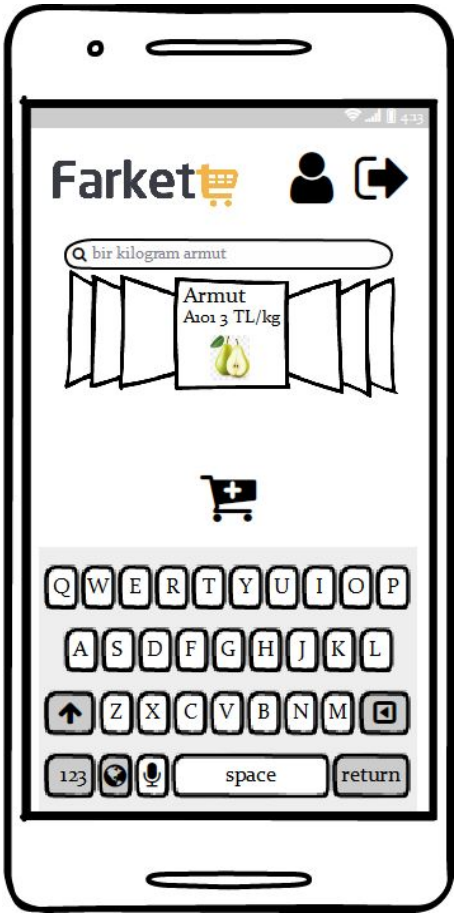

Figure 15. Main screen mock-up with search bar activated

By clicking the profile icon, the users will be directed to the profile screen illustrated in the figure 16. User can click on the picture icon again to be directed to the change profile picture screen illustrated in figure 17. In this page, user will be able to select a profile picture and click on confirmation button, or delete the current profile pic by clicking remove button. Users can also change their address range demonstrated in the figure 18. In this screen users will be able to search a location and change the radius of the range circle. There will be a option to view previous shopping routes and lists by clicking previous orders, illustrated in the figure 19. It will be possible to change the account information by clicking manage account button, illustrated in the figure 20.

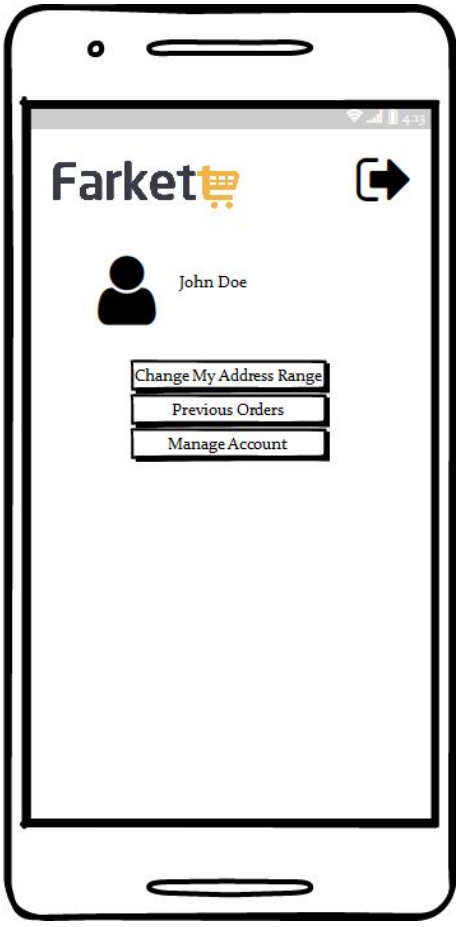

Figure 16. Profile screen mock-up

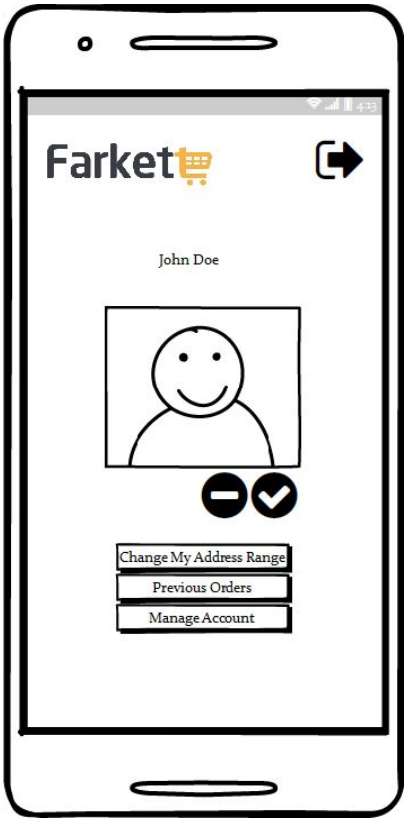

Figure 17. Change profile screen mock-up

Figure 18. Change address range mock-up

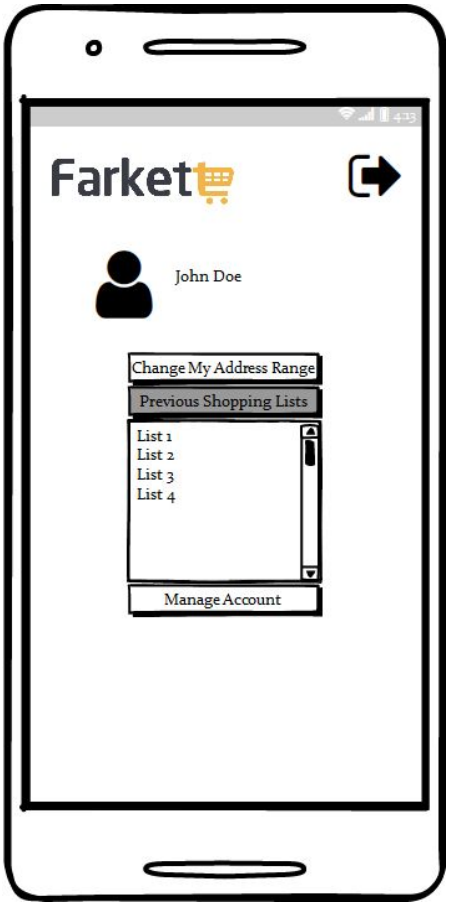

Figure 19. Previous shopping lists mock-up

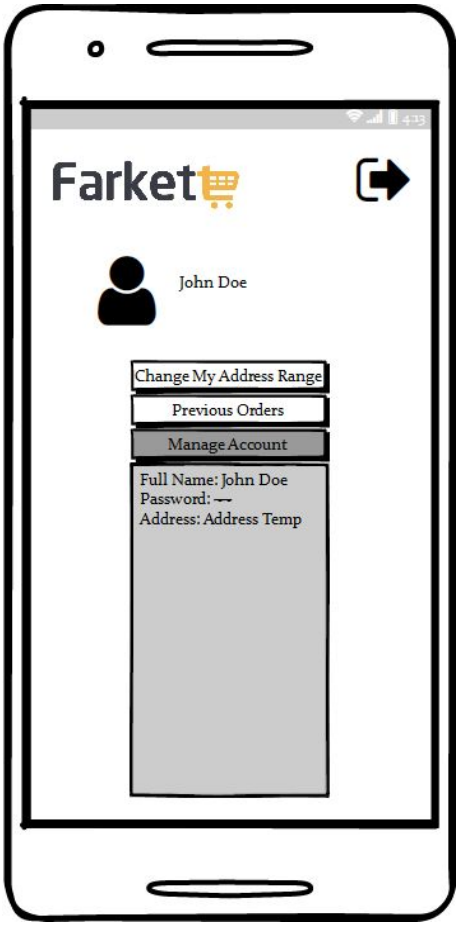

Figure 20. Manage Account screen mock-up

In the main screen, user can click on the shopping cart icon and go to their current shopping list, illustrated in figure 21. The shopping list will show the current items, their prices and amounts. User can click on an item and delete it. By clicking the confirmation button, users will be redirected to the confirmed shopping list screen, illustrated in the figure 22.

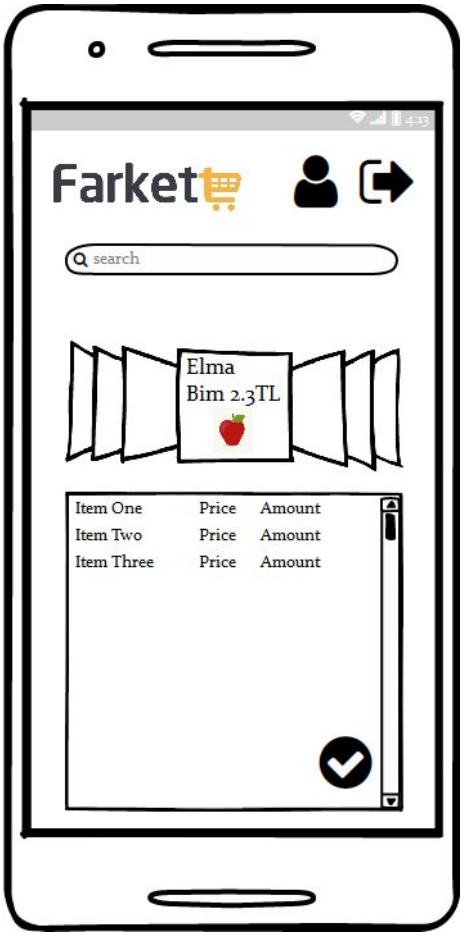

Figure 21. The shopping list screen mock-up

In the confirmed shopping list screen illustrated in figure 22, users will be provided with the information about the distribution of the shopping. A compass icon will be provided to direct to the shopping route screen, illustrated in figure 23. Users will be able to go back to the confirmed shopping list screen by clicking the shopping cart icon. In that screen, user will be provided with a shopping route demonstrating the optimal path to complete the shopping list with minimal spending.

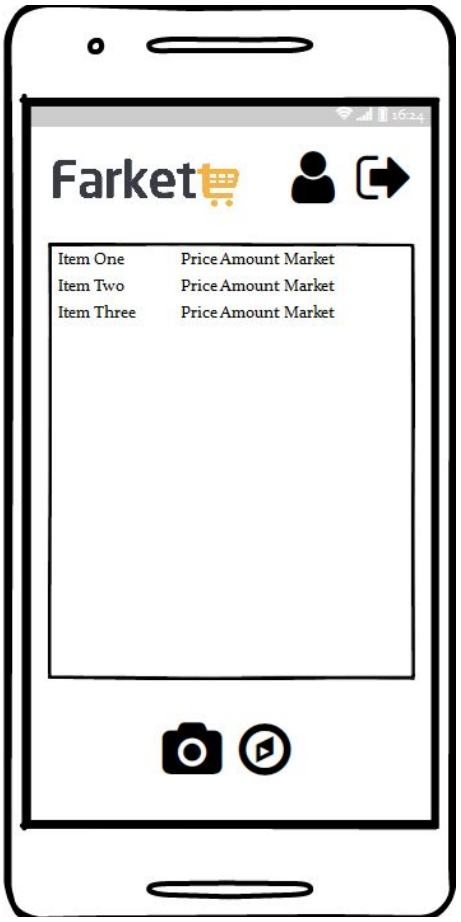

Figure 22. Confirmed shopping screen mock-up

It will be possible to click on market location in the map and obtain further information about the corresponding part of the shopping, illustrated in figure 24. Users will be able to scan receipts by clicking the camera icon. The scan receipt screen mock up is illustrated in figure 25.

Figure 23. Shopping route mock-up

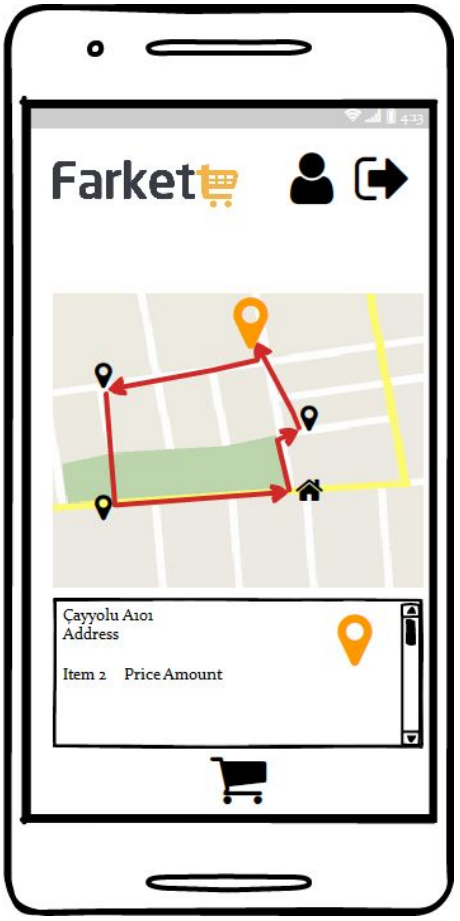

Figure 24. Location information mock-up

Figure 25. Scan receipt screen mock-up

4. Glossary

Product: Anything that can be purchased from a store.

Shopping Item: A set of products corresponding to a minimal category such as milk, cola, dish soap etc. Notice that their brands and quantities are not specified.

Gross Market: Convenience store.

Shopping Route: The route that contains all stores to buy items in a shopping list in the optimal way.

Discount: Decrease of price.

Markup: Increase of price.

Predictive Search: A type of search that uses prediction based on users previous queries.

Sales: Activities related to selling or the number of goods or services sold in a given time period.

Crowdsourcing: Gathering data from a large number of internet users.

5. References

[1] cimri.com.(2018) HomePage. [Online] Available at: https://www.cimri.com/ [Accessed 11 November. 2018].

[2] Basket. (2018) HomePage. [Online] Available at: https://basket.com/ [Accessed 11 November. 2018].

[3] Grocery Pal. (2018) on Play Store. [Online] Available at: https://play.google.com/store/apps/details?id=com.twicular.grocerypal.android [Accessed 11 November. 2018].

[4] Grocery IQ. (2018) HomePage. [Online] Available at: http://www.groceryiq.com/ [Accessed 11 November. 2018].

[5] Bring!. (2018) HomePage. [Online] Available at:https://getbring.com/#!/app [Accessed 11 November. 2018].

[6] Favado. (2018) on Play Store. [Online] Available at:https://play.google.com/store/apps/details?id=com.savings.android.grocery&hl=tr [Accessed 11 November. 2018].