Bilkent University

Department of Computer Science

Senior Design Project

Project Name: PriceWise
Analysis and Requirements Report

Group: T2323

Tuğberk Dikmen, 21802480
Deniz Hayri Özay, 21803632
Mehmet Ali Öztürk, 21703425
Mete Arıkan, 21902316
Furkan Yıldırım, 21902514

Supervisor: Shervin Rahimzadeh Arashloo
Jury Members: Atakan Erdem, Mert Bıçakçı

December 08, 2023
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

Table of Contents

1. Introduction

In recent years, due to inflation in Turkey, the prices of daily used market products have been increasing very much and changing very frequently. Due to this situation, people now have difficulty learning what is cheaper or more expensive. With the PriceWise application, application users will be able to compare their grocery purchases between different markets, whether singular or plural. The biggest difference between PriceWise and similar applications is that users can compare products in their shopping cart across brands according to all products in the cart. Apart from this basic feature, the application will be able to give recommendations to users with machine learning methods. For example, if you want to buy brand A chocolate and there is a discount on brand B chocolate among the markets in the application, the user can see this as a suggestion. Another feature is that, with the user's approval, the user is notified if there is a discount on a frequently used product.

2. Current System

Currently, there are websites that compare different prices of the same product from various stores. One of the most popular ones is Epey.com [1]. However, Epey.com does price comparisons only in the scope of one item. It does not have any shopping list functionality or something similar. It has only the prices of durable products such as technological devices, cooking utensils, car oil, shampoo, and toothpaste. The users can not search the prices of routine grocery products like chocolate, milk or fruits. Another significant alternative, Cimri.com, displays both durable and non-durable products and has shopping list functionality [2]. However, it does not have huge convenience store brands like Migros or A101 as product sellers. Cimri.com shows the prices of grocery products from retail websites like HepsiBurada.com or Amazon.com which is out of their main scope and because of that the routine grocery products have much higher prices and longer delivery times than usual. The list functionality of Cimri is not sophisticated as PriceWise targets for, because it does not recommend similar products if the selected product of the user is out of stock. It also forces you to buy from only a single store.

As a team, we analyzed the issues and gaps in the current system. We wanted to design and implement PriceWise, a mobile shopping assistant app which will address the aforementioned problems with its unique functionalities that will be discussed in the next

section. We aim to enhance the shopping experiences of people and enable them to buy their needs while keeping their budgets in check.

3. Proposed System

3.1 Overview

PriceWise will be a mobile application that introduces a rich set of functionalities in shopping comparison applications. Users will have the capability to effortlessly create and manage shopping lists and organize their purchases efficiently. The mobile application provides real-time updates on prices and availability from various websites of markets and ensures users about the most economical prices of their preferred lists. The shopping list optimization feature goes beyond the traditional price comparison apps by suggesting the most economical products across different markets.

Unlike other product comparison platforms like Akakce and Cimri, PriceWise brings a new impulse to the product comparison apps by focusing on the comparison of entire shopping lists. While other apps compare products one by one, PriceWise optimizes the entire shopping experience by considering the collective cost of a user's shopping list from various markets. This distinctive approach not only saves time, but also provides a wide view of the most cost-effective way to make their entire shopping lists. Also, PriceWise will provide a functionality that shows the closest markets to the users based on their current locations and shopping lists. This feature facilitates efficient navigation for users and ensures quick access to the best prices.

3.2 Functional Requirements

- Secure login and authentication process.
- Tracking of user selections and preferences to offer more personalized shopping suggestions.
- Ability for users to create, edit, and manage their accounts.
- A feature to obtain explicit user consent for tracking their selections and preferences, in compliance with data privacy and protection regulations.
- Clear communication to users about how their data will be used to enhance their app experience and the measures taken to ensure data privacy and security.
- Users can create, edit, and save multiple shopping lists.
- Functionality to add, remove, or modify items in the shopping list.

- Capability to search for individual items or entire shopping lists.
- Retrieval of best available prices for each item from a range of websites and stores.
- Display of price comparison for similar products from different brands (e.g., suggesting cheaper milk from brand X over pricier brand Y).
- Option to view and select alternative products based on price and brand.
- Functionality to suggest optimized split shopping lists, indicating which items to buy from which sellers or websites for overall cost minimization.
- Real-time updating of prices and availability of items from various online sources.
- Push notifications for price drops, special offers, or other relevant alerts based on user preferences and shopping lists.
- Feature for users to provide feedback or report issues.
- Access to customer support for assistance with app-related queries.
- Limited functionality (viewing saved lists) available in offline mode.
- Incorporation of a machine learning model that learns from user choices and preferences.
- Tailoring future shopping list suggestions to align with individual user preferences.
- Continuous adaptation of the model to refine suggestions based on user interactions.

3.3 Nonfunctional Requirements

3.3.1 Usability

- The application should have an user friendly and intuitive interface to provide a positive user experience.
- The layout should be similar to other competitors as this will ease the transition process of the users to our application.
- In order to add many items to a shopping list more easily users will have a current list that they have chosen and they can add as many items as they wish.

3.3.2 Reliability

- The application should be accurate and up to date since the prices of many products in Turkey change frequently. It should inform the users about incoming sales for the products.
- In order to be accurate with the prices, the algorithm will work every 24 hours and admins can run the algorithm individually to update the prices if they think it is necessary.
- The application should be available for almost all times except for maintenance times.

3.3.3 Performance

- The application should have minimal response times under 2 seconds to ensure a positive user experience.
- The application should be able to handle high request traffic during special days such as holidays and Black Friday as these days are expected to be peak times for the usage of this application..

3.3.4 Supportability

- The application should be supported on different Android and IOS versions.

3.3.5 Scalability

- The application should be designed to handle an increasing number of users and data as the time goes on.

3.3.6 Security

- The users' information such as username, email, and password should be secured from 3rd party softwares.

3.4 Pseudo Requirements

- GitHub will be used as the version control system to manage source code and facilitate collaborative development.
- GitHub will also serve as the project management platform for issue tracking, task assignment, and overall project coordination.
- The mobile application will be developed using Flutter to ensure cross-platform compatibility, targeting both Android and iOS operating systems.
- MySQL database will be employed for efficient data management and storage, accommodating the dynamic nature of the application.

- Google Maps API will be integrated to provide mapping functionality, offering features related to location-based services within the application.
- Python will be used for the implementation.
- Beautifulsoup library will be used for web scraping
- Meetings will be held on Google Meet, Zoom, Discord.

3.5 System Models

3.5.1 Scenarios

Scenario 1

Use-case name: Sign In

Actors: Registered User

Entry Conditions: Application should be opened and the current user has to be not logged in.

Exit Conditions: Users either choose to sign in or continue as a non-registered user.

Main Flow Events:

1. Open the application.
2. Click the sign in button.
3. Fill the required area.
4. Either click to sign in or go to the search item page.

Scenario 2

Use-case name: Sign Up

Actors: Non-registered User

Entry Conditions: Application should be opened and the current user has to be not logged in.

Exit Conditions: Users either choose to register or cancel.

Main Flow Events:

1. Open the application.
2. Click the register button.
3. Fill the required area.
4. Either click the register button or cancel.

Scenario 3

Use-case name: Search an Item

Actors: Registered User, Non-registered User

Entry Conditions: Open the application, whether be registered user or non-registered user and click to the search item page.

Exit Conditions: Click another pages button.

Main Flow Events:

1. Open the application.
2. Click on the Search page's button.
3. Enter any item, which is desired.

Scenario 4

Use-case name: View Suggested Shopping List

Actors: Registered User, Non-registered User

Entry Conditions: Open the application

Exit Conditions: Click back button.

Main Flow Events:

1. Open the application.
2. Click to the Lists page button.
3. Click on the suggested list, which showed in the page.

Scenario 5

Use-case name: Modifying Shopping List

Actors: Registered User, Non-registered User

Entry Conditions: Open the application, have a shopping list.

Exit Conditions: Click back button.

Main Flow Events:

1. Open the application.
2. Click to the Lists page button.
3. Select and click on any list, which showed in the page.
4. Click on the Modify Shopping List button.

Scenario 6

Use-case name: View Sales

Actors: Registered User, Non-registered User

Entry Conditions: Open the application.

Exit Conditions: Click back button.

Main Flow Events:

1. Open the application.
2. Click to the Lists page button.
3. Click on the suggested list, which showed in the page.
4. Select any suggested list, which showed in the page.

Scenario 7

Use-case name: View Past Shopping Lists

Actors: Registered User

Entry Conditions: Open the application, User should be logged in.

Exit Conditions: User clicks on another page button.

Main Flow Events:

1. Open the application.
2. Sign in.
3. Click on Profile button.
4. Click on View Past Shopping Lists button.
5. Select and click on any past list, which showed in the page.

Scenario 8

Use-case name: View Preferences Personalized Shopping Lists

Actors: Registered User

Entry Conditions: Open the application, should have created list

Exit Conditions: Open the application

Main Flow Events:

1. Open the application.
2. Sign in.
3. Go to the Lists page.
4. Select the Suggested List.
5. Change the products according to desires.

Scenario 9

Use-case name: Update Database

Actors: Admin

Entry Conditions: Admin should open the application.

Exit Conditions: Click to Back button.

Main Flow Events:

1. Open the application
2. Go to Duty page.
3. Click the Update Database button.

Scenario 10

Use-case name: Send Notification

Actors: Admin

Entry Conditions: Admin should open the application.

Exit Conditions: Click to Back button.

Main Flow Events:

1. Open the application
2. Go to Duty page.
3. Click the Send Notification button.
4. Enter the message.
5. Click the Send button.

3.5.2 Use Case Model

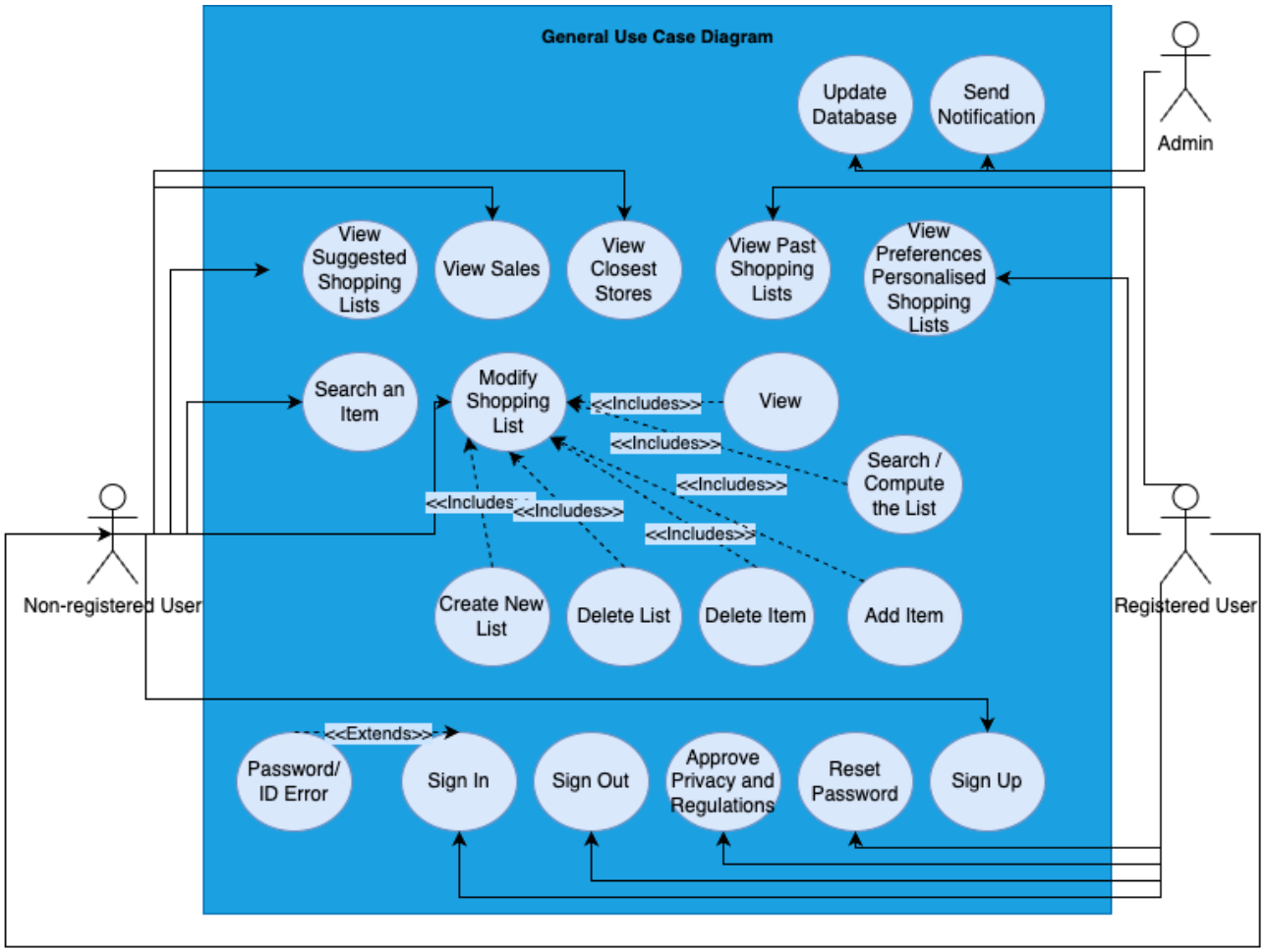

Figure 1. Use Case Diagram

The diagram represents a general use case model for a shopping platform, outlining the interactions a user can have with the system. It shows primary functions such as viewing suggested shopping lists, searching for items, and modifying shopping lists with options to create, delete, or add individual items. Secondary functions include account management features such as signing in and out, resetting passwords, and signing up. The model also incorporates administrative features like updating the database, sending notifications, and approving privacy regulations. This diagram serves as a blueprint for understanding the system's capabilities and the user's potential actions within the platform, aimed at ensuring a comprehensive and user-friendly experience.

3.5.3 Object and Class Model

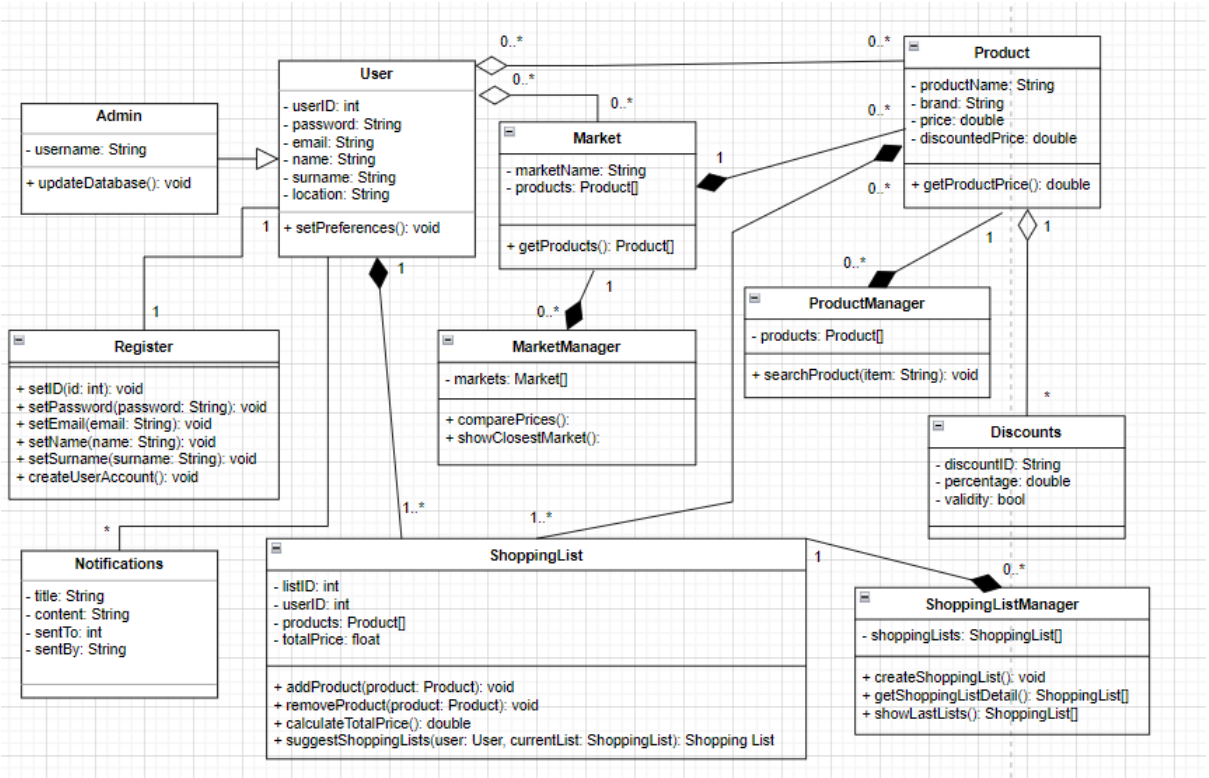

Figure 2. Object Class Diagram

- The Admin class represents an administrative user in the system. The admin has privileges for managing and overseeing various aspects of the application.
- The User class represents a regular user in the system, engaging with the application to create and manage their account, preferences, and other user-specific functionalities.
- The Register class is associated with user management, providing functionalities related to user registration.
- The Market class represents a physical market where products are available for purchase. It includes information about the market, such as its name and the products it offers.
- The MarketManager class manages and organizes multiple markets within the system. It handles the market operations and ensures the consistency of market data.
- The Product class represents individual products available to purchase within a market. It has details such as the product name, brand, and price.
- The ProductManager class manages the organization of various products within the system. It provides coherence in product operations.
- The Discounts class represents discounts or special offers that may be applicable to specific products. It holds details such as the discount name, percentage, and validity.
- The ShoppingList class represents a user's list of items they want to purchase. It includes information about the list, such as its ID, the associated user, the list of products, and the total price.

- The ShoppingListManager class manages and organizes multiple shopping lists within the system. It handles the lifecycle of shopping list operations.
- The Notifications class represents notifications that can be sent to users. It includes information such as the title, content, receiver, and sender of the notification.

3.5.4 Dynamic Models

3.5.4.1 Activity Diagram

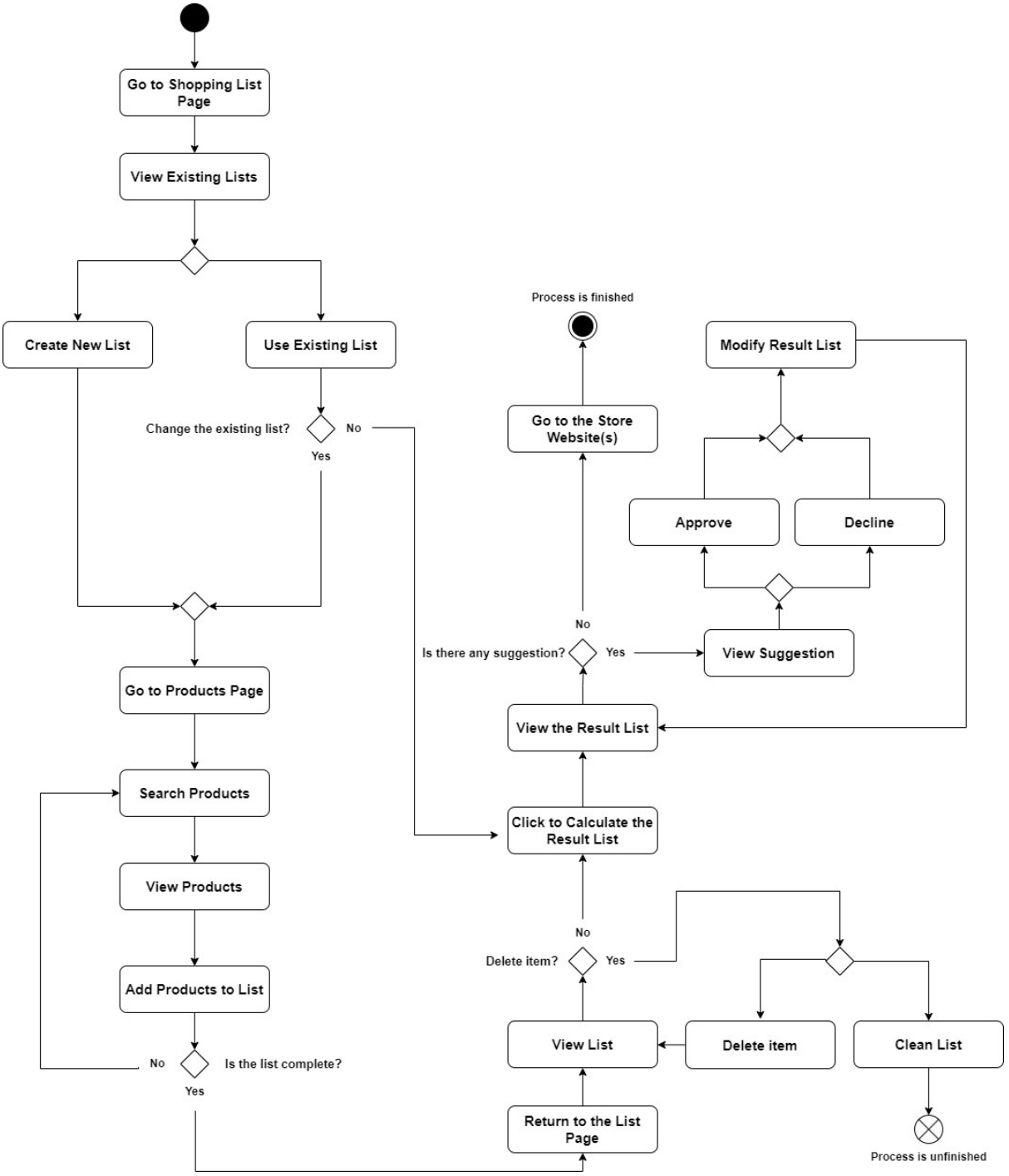

Figure 3. Activity Diagram

The activity diagram presents a workflow for managing a shopping list within a digital platform. The process starts when the user navigates to the Shopping List Page, where they can opt to view existing lists. Depending on the user's choice, they can either create a new list or use an existing one. If an existing list is selected, the user is presented with a decision point to either modify it or proceed without changes. Modifications involve going to the Products Page, searching for products, viewing them, and then adding selected products to the list. The user assesses whether the list is complete; if not, they loop back to add more products. Once the list is finalized, the user can opt to calculate the total, view suggestions if available, and either approve or modify the resultant list based on suggestions or personal preferences.

If the user decides to modify the result list, they can delete individual items or clean the list entirely and then return to the list page to review or alter further. The diagram illustrates two endpoints: one where the user finishes the process after approving the list, potentially proceeding to a store's website for purchase, and another where the process remains unfinished, indicating that the user may still want to make changes to the list. This activity diagram is designed to help system developers understand the step-by-step user interactions needed for the shopping list feature, ensuring the system supports a comprehensive user experience.

3.5.4.2 State Diagram

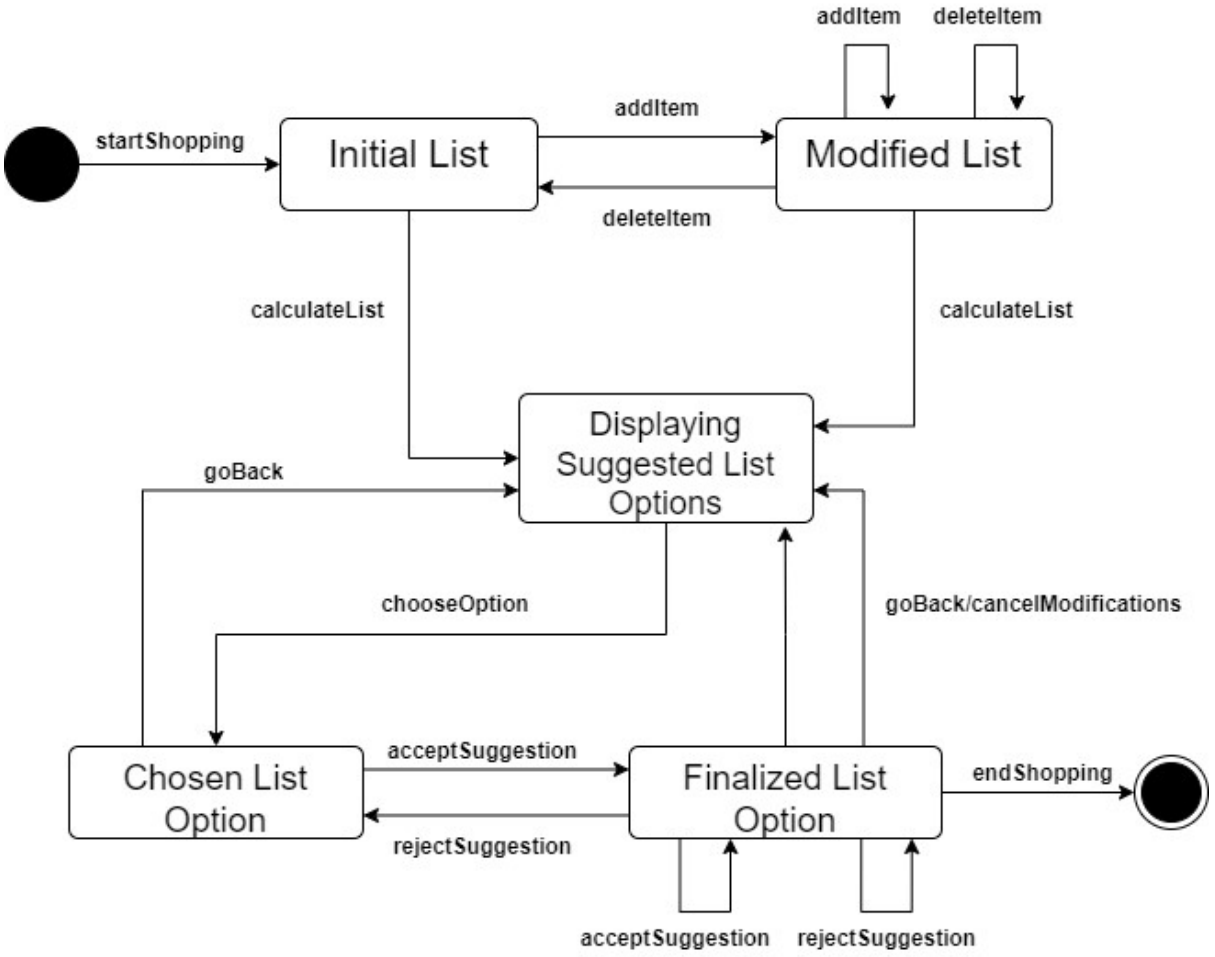

Figure 4. State Diagram

The state diagram serves as a visual representation of the various stages a user's shopping list undergoes within a system, detailing the transitions between states from the beginning of the shopping process to its conclusion. The process begins when the user initiates shopping, creating an 'Initial List'. From this point, the user can perform a series of actions such as adding items to the list, thus transitioning it to a 'Modified List', or removing items, potentially reverting it back to the initial state. At any stage, the user can calculate the total of the list, which is an operation that does not change the state of the list but provides valuable information to the user.

As the user interacts with the list, the system may offer 'Suggested List Options', which are generated based on the user's current list and possibly their past behavior or preferences. The user has the choice to 'chooseOption', which transitions the list to a 'Chosen

List Option' state, where they can 'acceptSuggestion' to proceed to a 'Finalized List Option', effectively marking the completion of their shopping list modification. Alternatively, the user can 'rejectSuggestion' at any point, which either leads to further modification of the current list or a return to the initial list. The process is marked as complete when the user ends shopping, finalizing the state of the list. This state diagram is critical for developers and analysts to understand the logic flow and the system's requirements to support user interactions with their shopping lists, ensuring a seamless and intuitive user experience.

3.4.5.3 Sequence Diagrams

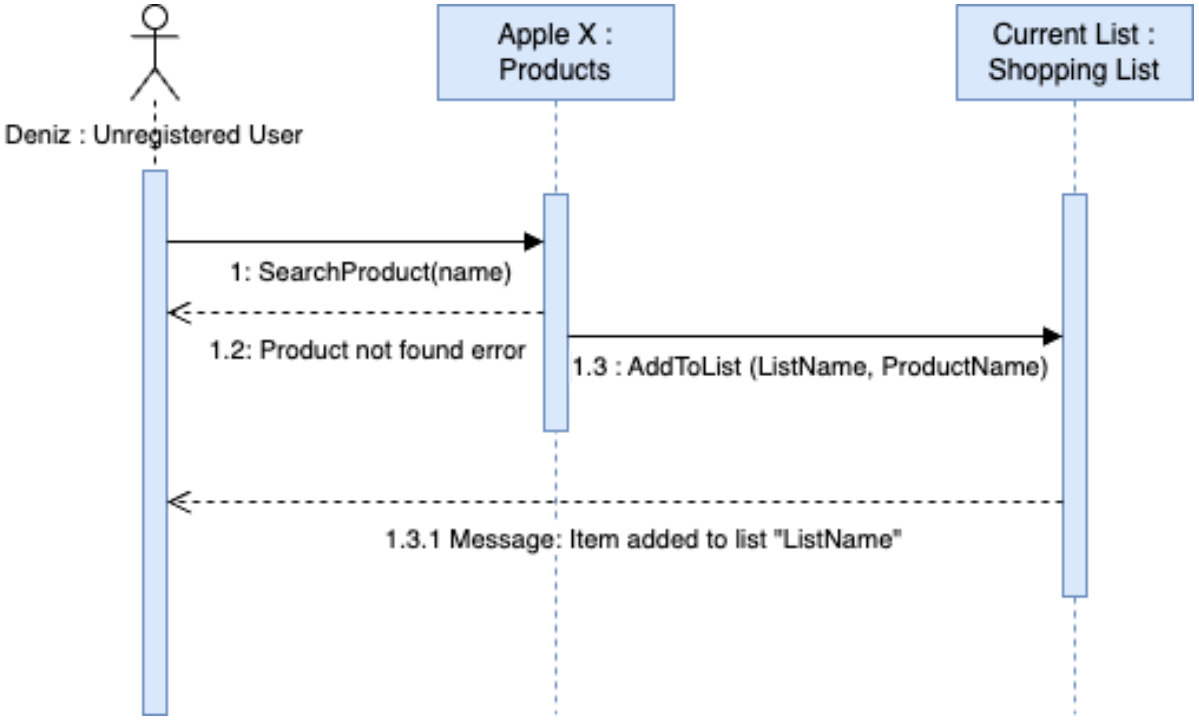

Figure 5. Sequence Diagram

The sequence diagram outlines the process flow for adding a product to a shopping list. It begins with a user-initiated search for a product by name. If the product is not found, an error message is generated. Upon successful location of the product, the system proceeds with the 'AddToList' function, where the product name is appended to a specified shopping list. The final step in the sequence is a confirmation message to the user indicating the successful addition of the item to the list. This diagram captures the interaction between the user and the system components, the sequence of actions and responses that facilitate the updating of a shopping list.

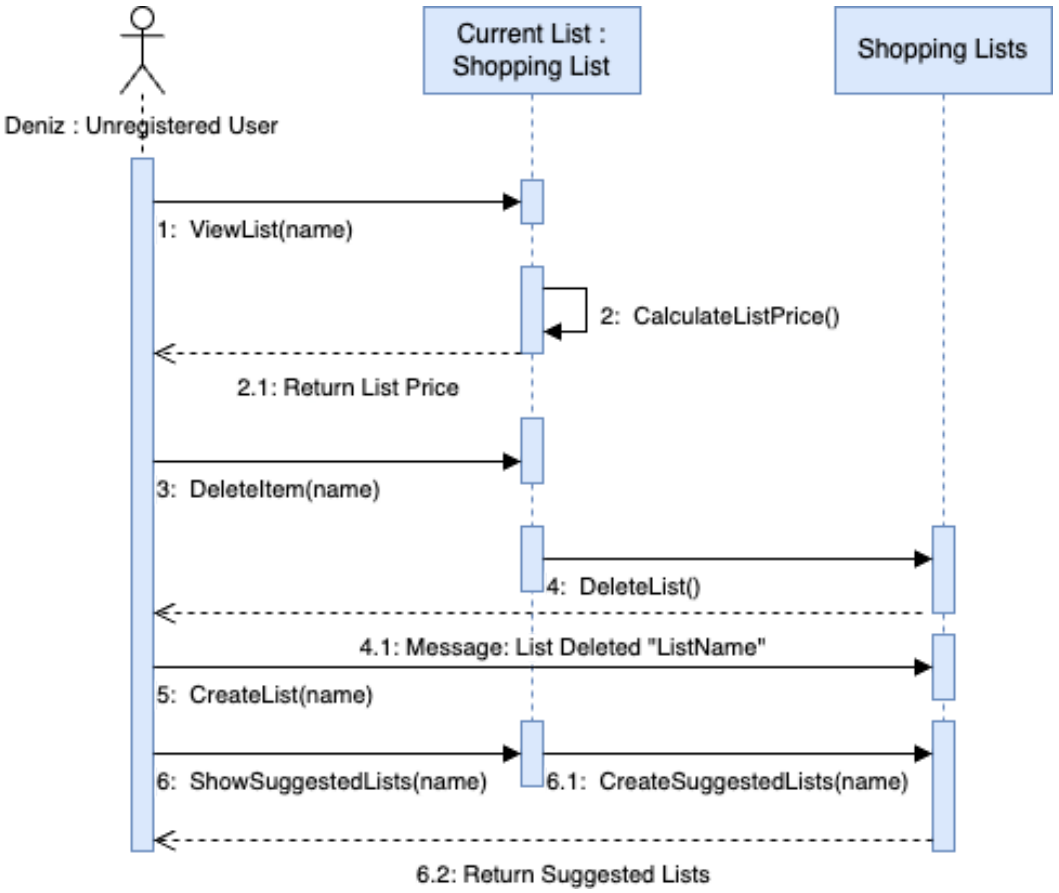

Figure 6. Sequence Diagram

The sequence diagram depicts the various operations a user can perform on the shopping lists within a system. It begins with the user viewing a list, followed by the system calculating and returning the total price of items in the list. The user can delete an item from the list, or delete the entire list itself, with the system confirming the deletion of the list. Additionally, the user has the option to create a new list and view suggested lists, which are generated by the system based on the user's input. This illustrates the flow and interaction between user commands and system responses for managing shopping lists, providing a clear guide for developers to understand the functionalities required for list management in the system.

3.5.5 User Interface - Navigational Paths and Screen Mock-ups

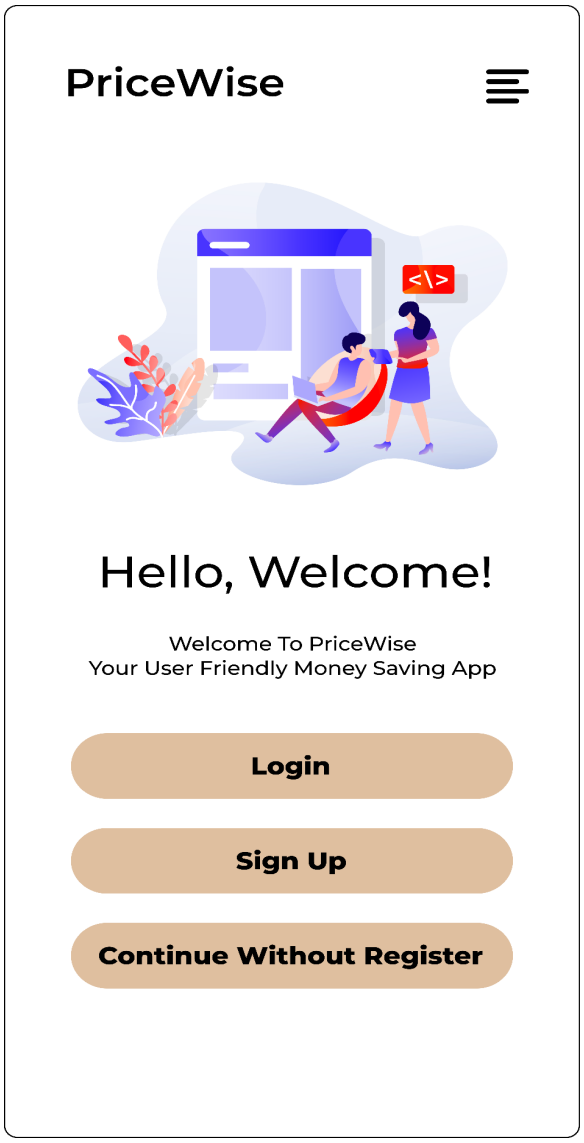

Figure 7. HomePage

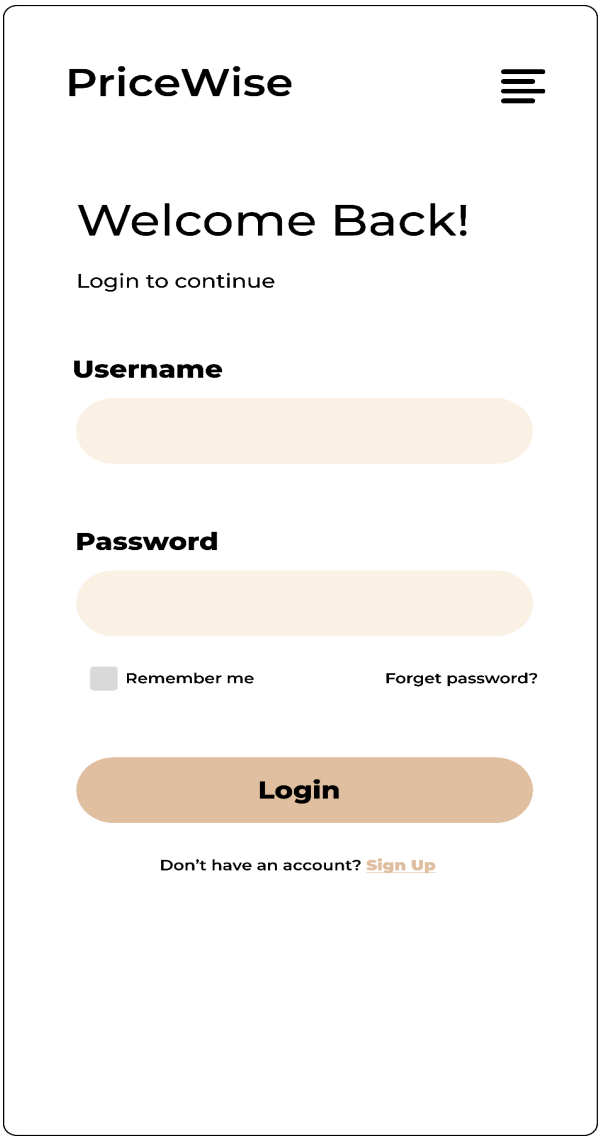

Figure 8. LoginPage

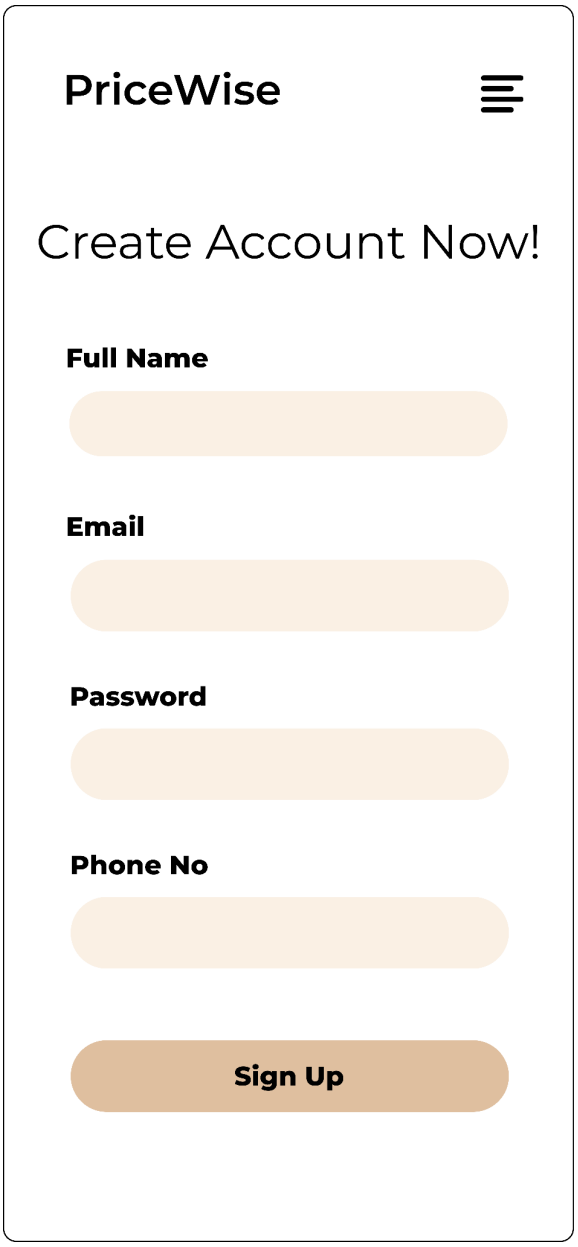

Figure 9. SignUpPage

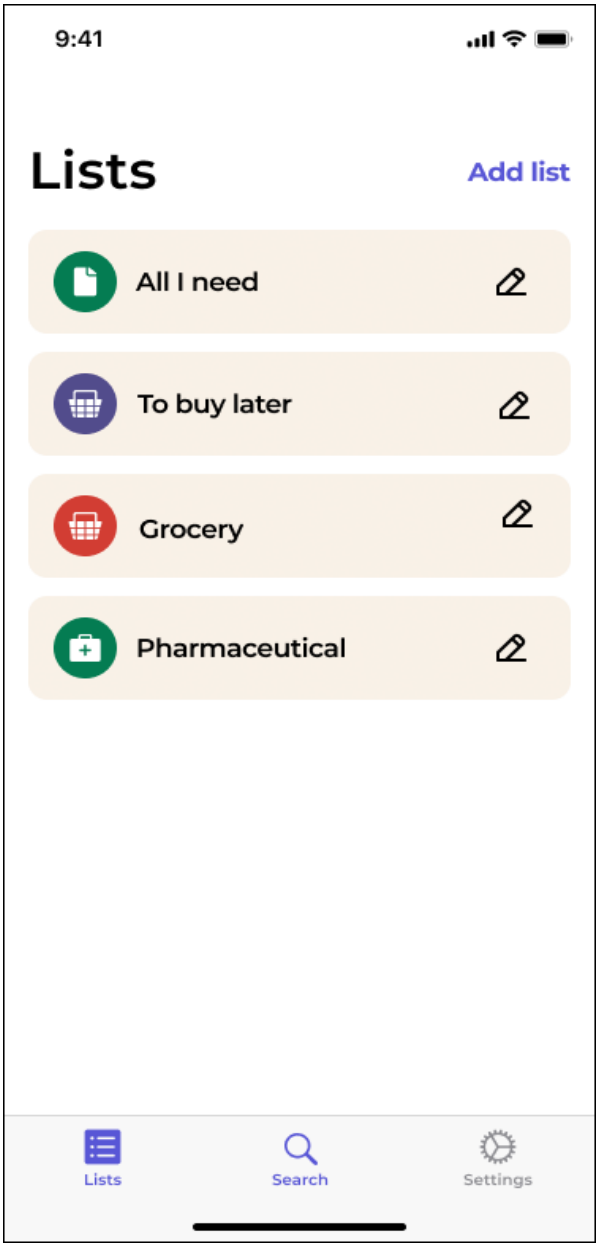

Figure 10. MyShoppingLists

Lists page shows all the shopping lists of the user. There is an add list button, where the user can add a new list and an edit button in case if the user wants to make changes later. In addition to these features there is a navigation bar at the bottom of the page that users can use to navigate important pages.

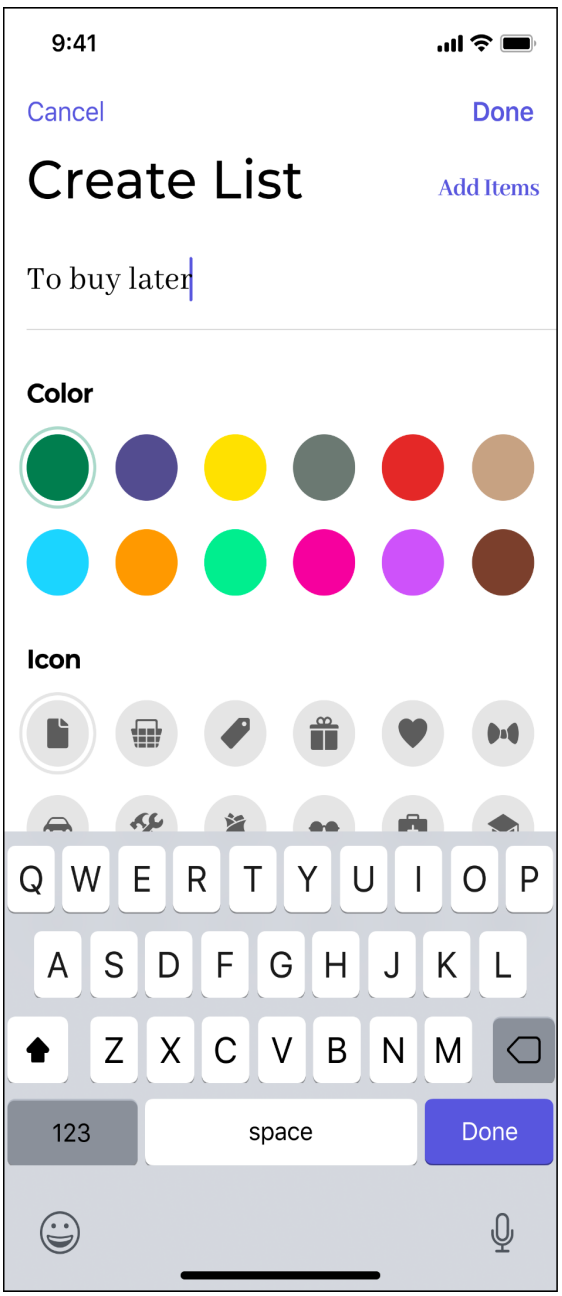

Figure 11. CreateShoppingList

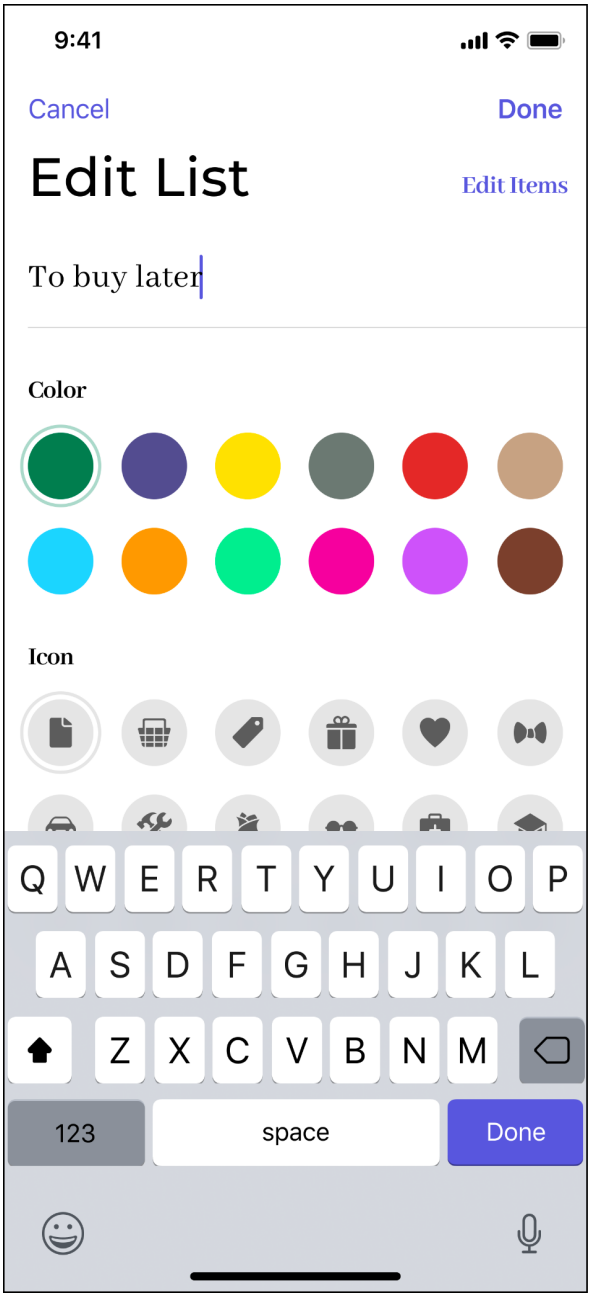

Figure 12. EditShoppingList

There are two pages. In the “Create List” page, when the user adds a new list, the user defines its name and lookup. In the “Edit list” page, when the user wants to change the current list’s name and lookup, the user can use this page.

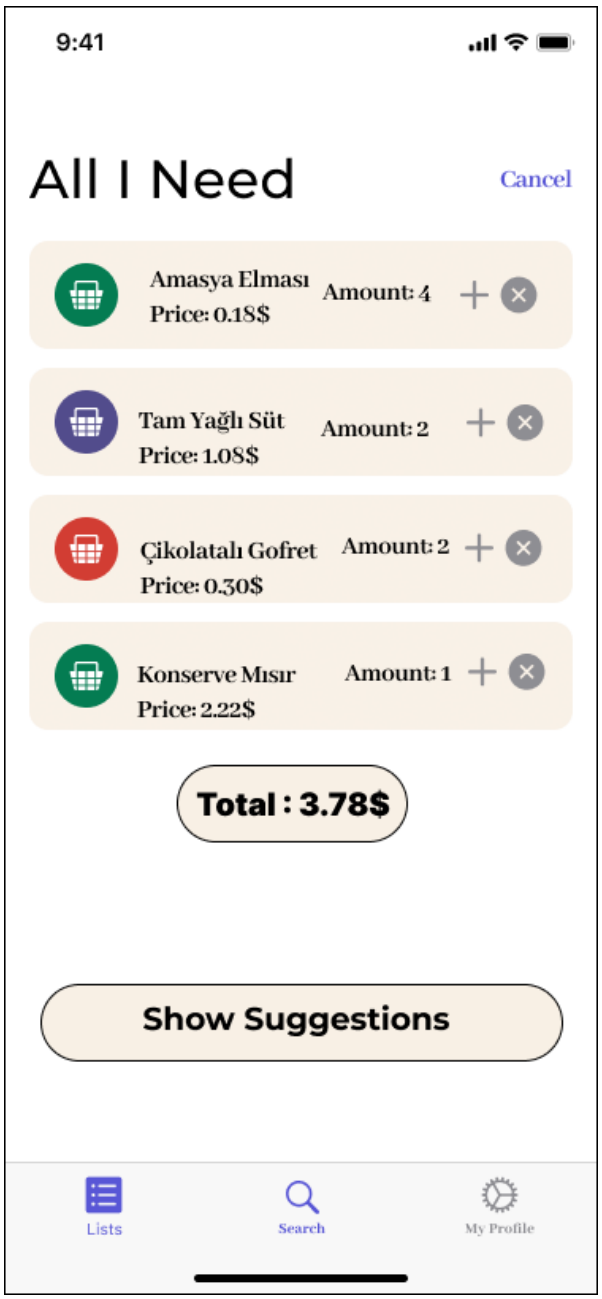

Figure 13. ShoppingListDetails

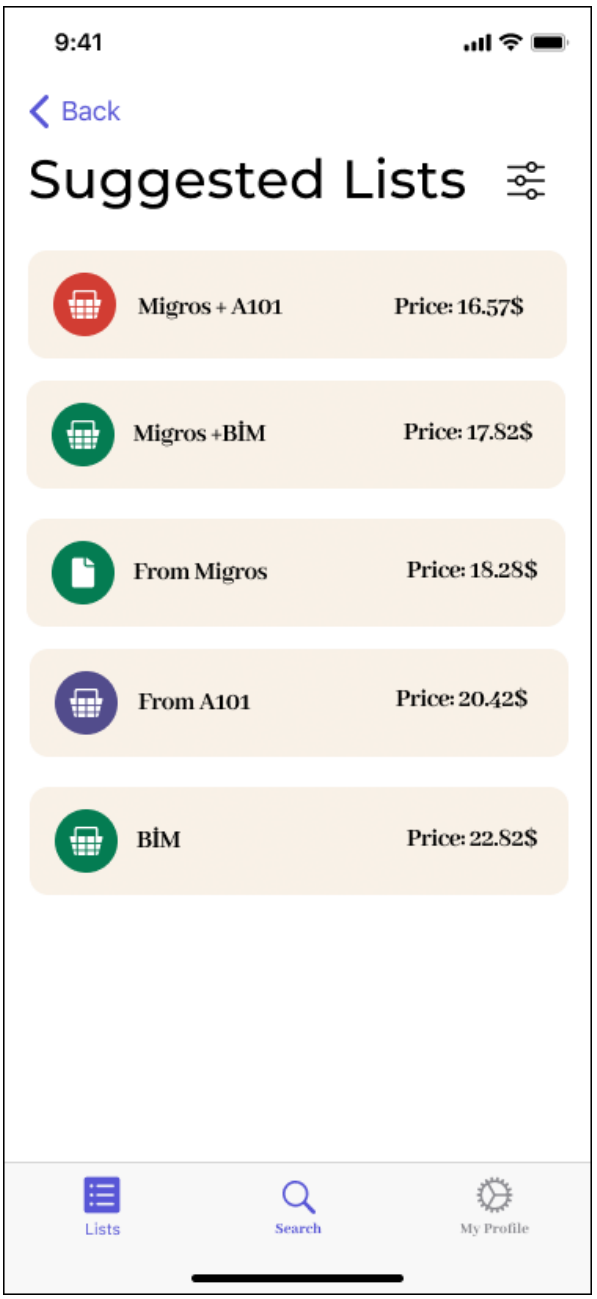

Figure 14. SuggestedShoppingLists

In the “ShoppingListDetails” page the user can see the products and details of the selected shopping list. In the “Suggested Lists” page, the user can see shopping lists which are recommended by the app.

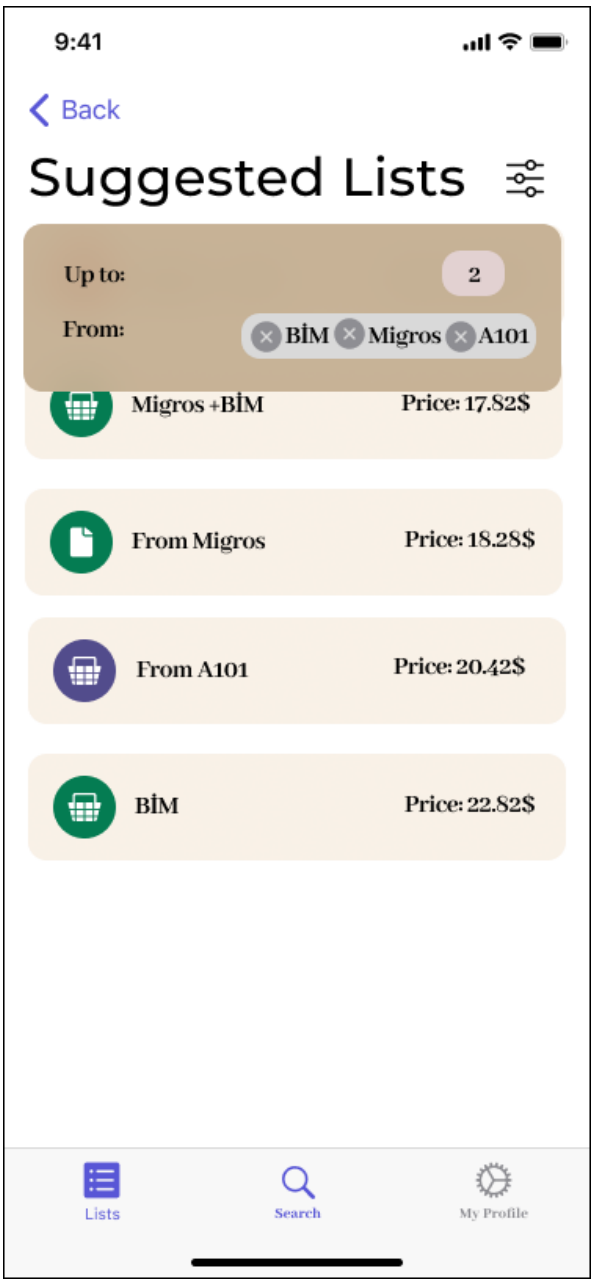

Figure 15. SuggestedShoppingListsFilter

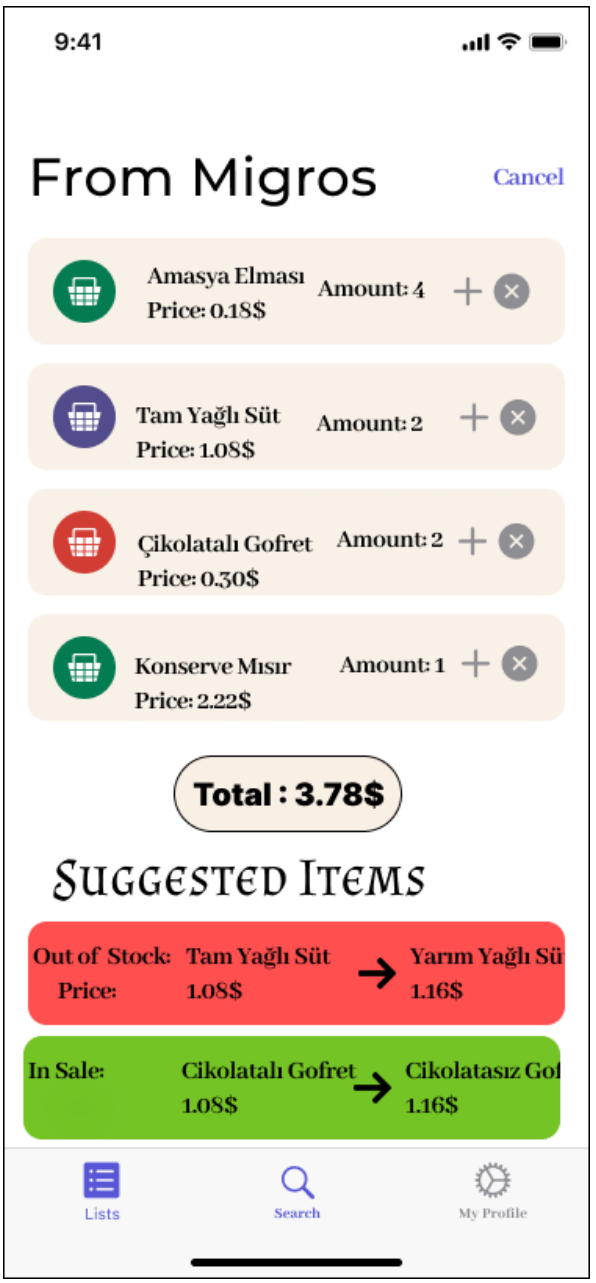

Figure 16. SuggestedShoppingItems

In the “Filtered Suggested Lists”, the user can filter the suggested lists, which is created by the app. In the “SuggestedShoppingItems”, the user can see the products that are recommended by the app. In addition there are suggested products which are suggested by the app, these are the products that are on instant sale.

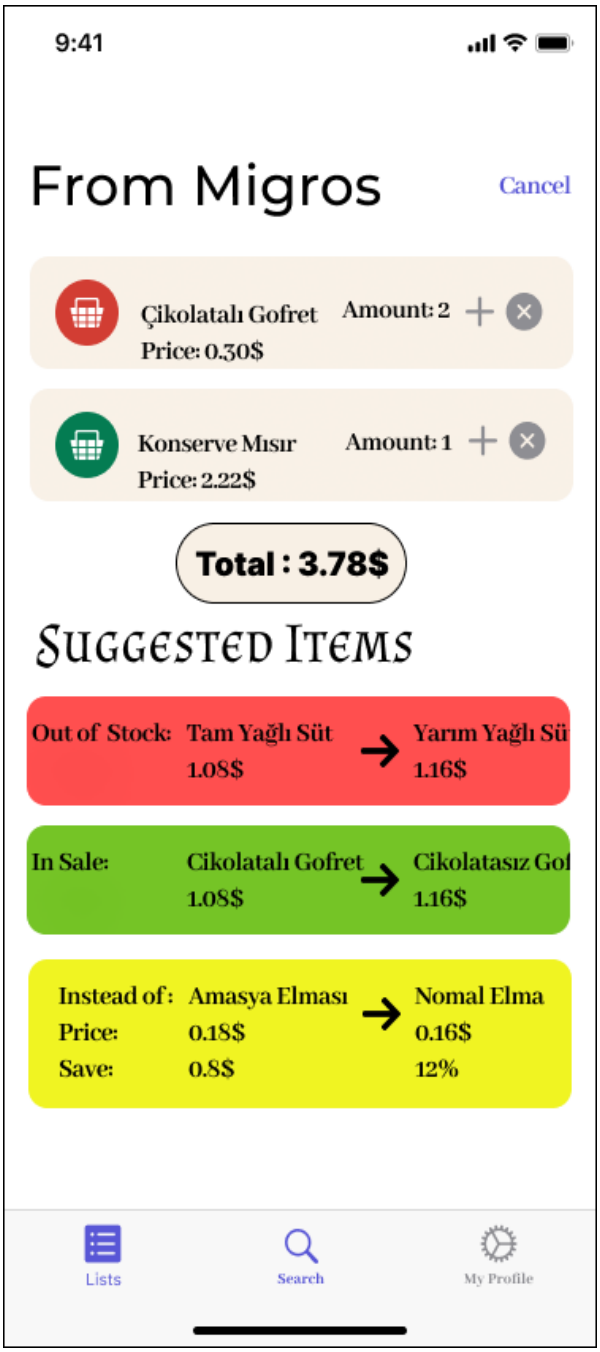

Figure 17. SuggestedShoppingItems

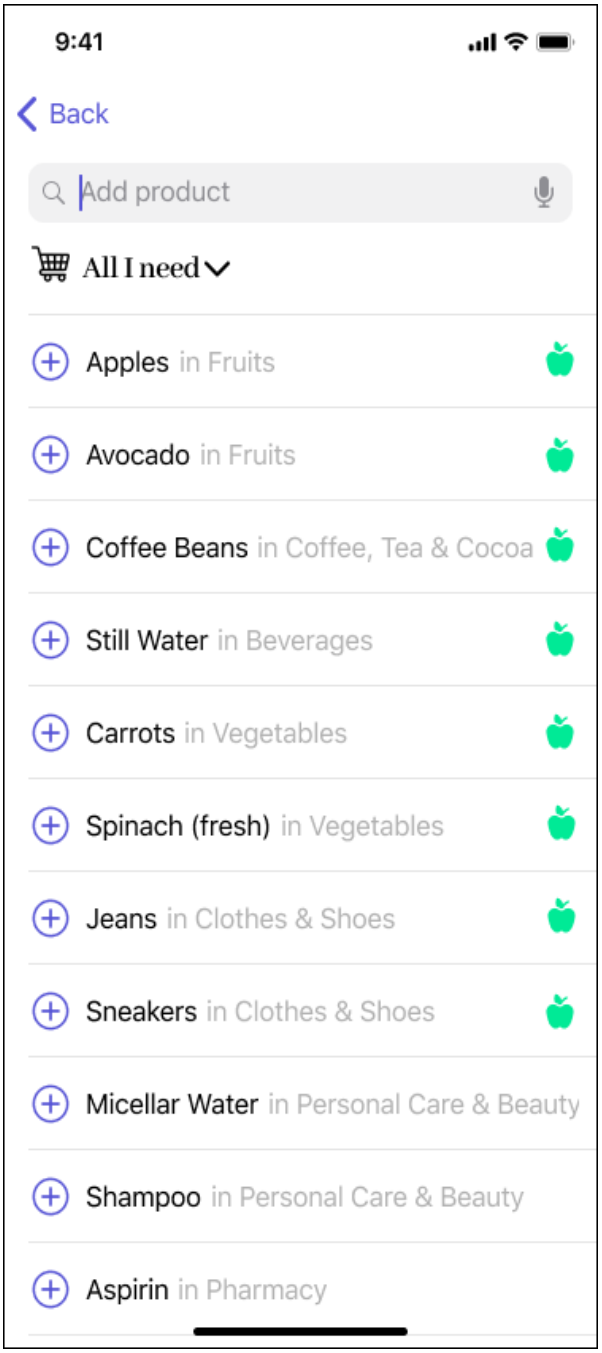

Figure 18. SearchProducts

In the “SuggestedShoppingItems”, there is another recommendation which shows the change in the product. In the “SearchProducts”, users can search items to add to their shopping list.

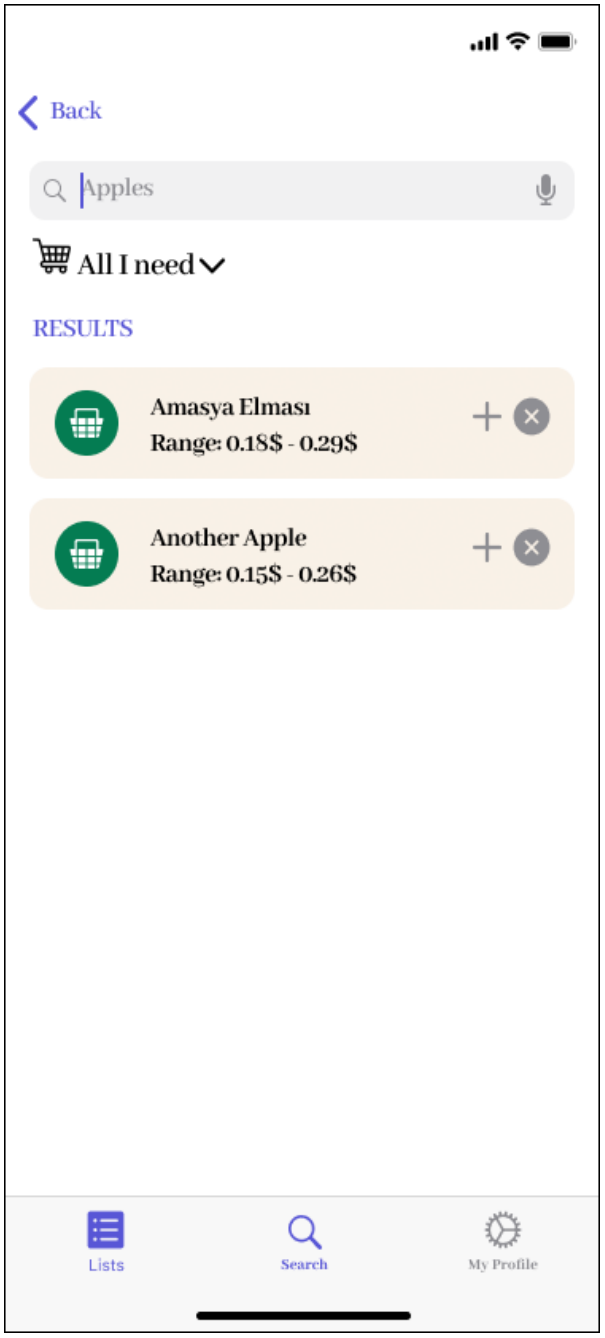

Figure 19. SearchProductResults

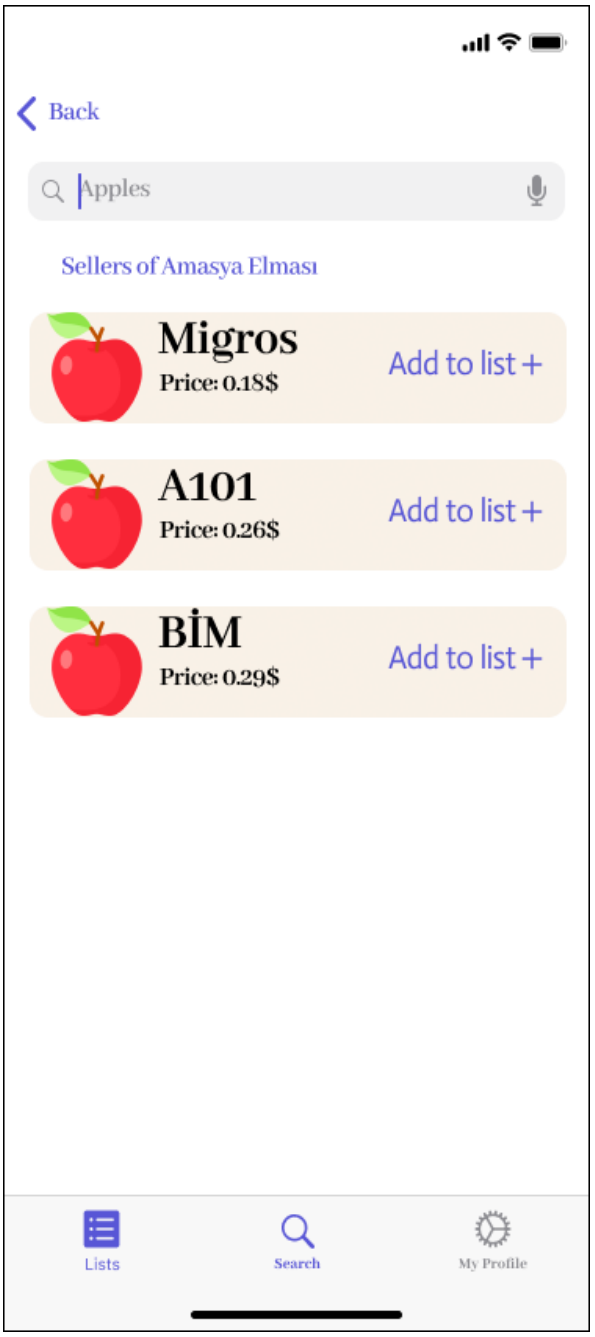

Figure 20. ProductSellers

In “SearchProductResults” the users can see the results for their searched product. In the “ProductSellers” page users can see the sellers of the specific product in detail.

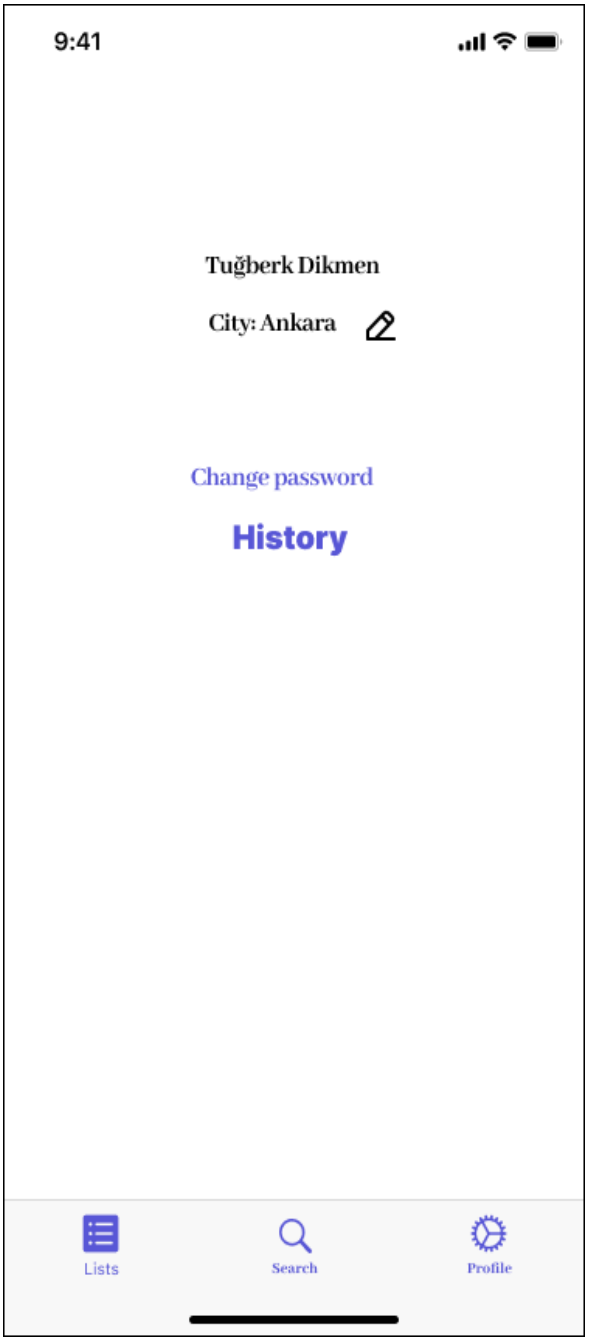

Figure 21. MyProfile

Figure 22. History

"MyProfile" page is one of the capstone pages in the application. The users can use this page to adjust their preferences and to see their history. In the "History", registered users can see their past lists.

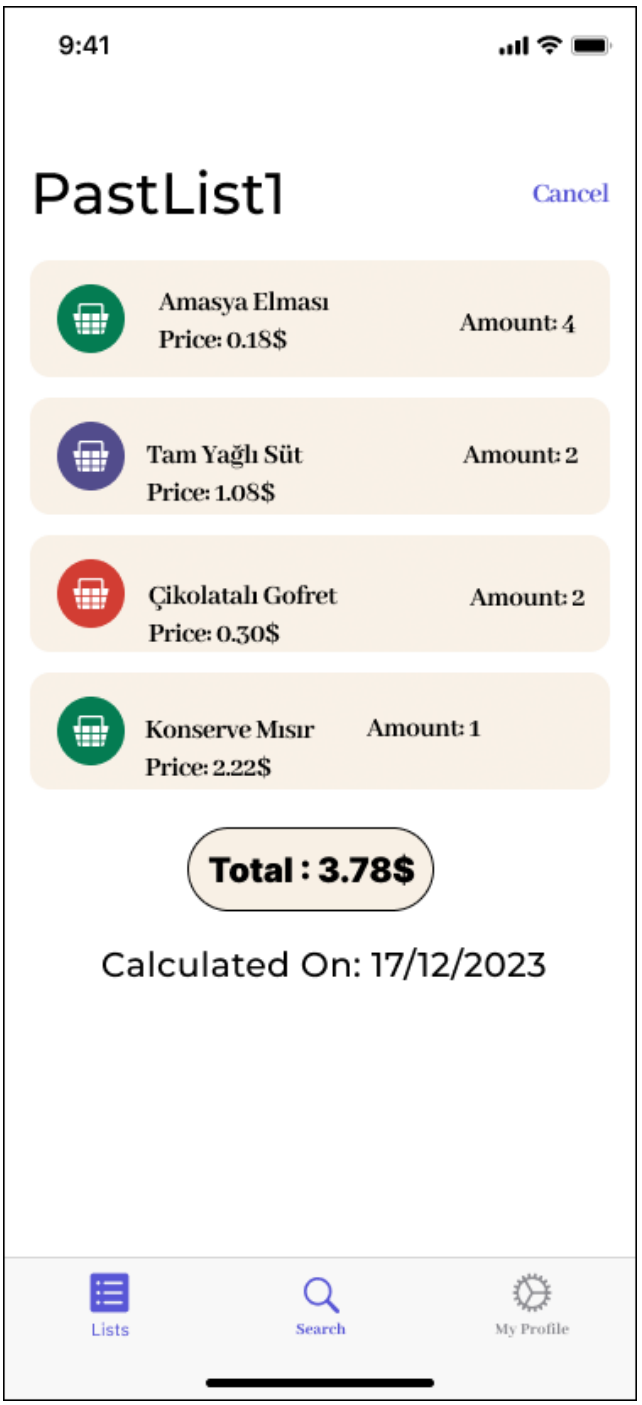

Figure 23. HistoryShoppingListDetails

In the “HistoryShoppingListDetails”, users can see their past list content in detail. There are products and their prices, which shows the price on the day the list was purchased.

4. Other Analysis Elements

4.1. Consideration of Various Factors in Engineering Design

4.1.1 Constraints

4.1.1.1 Time Constraints

This application is well suited for horizontal and vertical scalability, as there are countless products, markets, and very high potential of users and data. The most important constraints among all is the time constraint as there is a very limited time to include all these features and aspects.

4.1.1.2 Cost Constraints

As our budget and resources during the early development of this application is limited, it is hard to reach the full potential of this project during the time of this class.

4.1.1.3 Scope Constraints

At the beginning of the design process, this application is thought to include many kinds of products such as electronics, clothing, and healthcare, however, due to limitations it is decided that during the early development phases the application will only include grocery products.

4.1.1.4 Privacy and Data Protection Constraints

Privacy and data protection are one of the major issues for users and even though using the data of the users in different ways may have benefited the application in terms of progress and development, this constraint is one of the key limitations for this project.

4.1.2 Standards

4.1.2.1 Coding Standarts

In order for a project to be viable in the market and to have an easy development phase, reaching coding standards is one of the key factors that define the success of a software project. This includes applying coding conventions and adding comments for consistency, readability, and maintainability.

4.1.2.2 Testing Standarts

To provide a smooth user experience, an application should be bug-free and consistent. Applying inclusive and effective testing standards will ensure a better user experience, as this will decrease the number of bugs and critical errors.

4.1.2.3 Documentation Standarts

Reaching documentation standards is a key factor for our team to have a smooth and effective development experience, even though we use comments and version control systems to be more effective and consistent.Also it is not only important for the developers who work on this project to be consistent and effective. After the early development phases, if this application is decided to be an API for other developers to use on their projects documentation will be the key for them to provide a smooth development experience.

4.2. Risks and Alternatives

4.2.1. Market Risks and Alternatives

Problem: Changes in market trends and user preferences can impact the success of the application.

Plan: Conducting regular market research to stay informed about market changes to respond quickly to changing market conditions.

4.2.2. User Risks and Alternatives

Problem: Reduced user engagement can impact the success of the application which leads to lower user satisfaction and potentially affecting the app's use rate negatively.

Plan: Implement a user feedback mechanism to gather insights into user satisfaction and preferences. Actively address user concerns and suggestions. Also allowing users to control and customize their preferences. Provide options for users to manually update or reset their preferences if they wish.

4.2.3. Privacy Risks and Alternatives

Problem: Potential privacy concerns can shake user trust and damage the application's reputation.

Plan: Implement security measures, such as encryption, secure authentication for user data. Detailing how their data is collected, stored and used. Provide users with control over their data which will Enhance user trust by promoting transparency and giving users confidence in how their information is handled.

4.3. Project Plan

Name	Date	Leader	Member Involved
Documentation	1 October 2023->1 May 2024	Tuğberk Dikmen	All Members
Front End Development	1 November 2023->30 January 2024	Deniz Hayri Özay	Mete Arıkan, Tuğberk Dikmen
Web Scraping Back End Development	20 November 2023->15 April 2024	Furkan Yıldırım	Mehmet Ali Öztürk
Project Demo	1 December 2023->15 December 2023	Mete Arıkan	All Members
User and List Back End Development	5 December 2023->30 January 2024	Mehmet Ali Öztürk	Deniz Hayri Özay, Furkan Yıldırım
Database Implementation	1 January 2024->15 April 2024	Furkan Yıldırım	Mehmet Ali Öztürk, Tuğberk Dikmen
Machine Learning Back End Development	20 January 2024->20 April 2024	Tuğberk Dikmen	All Members
Admin Front End and Back End Development	1 January 2024->1 April 2024	Mehmet Ali Öztürk	Mete Arıkan, Deniz Hayri Özay
Testing	1 April 2024->1 May 2024	Deniz Hayri Özay	All Members
App Launch	1 May 2024->15 May 2024	Mete Arıkan	All Members

WP 1-Documentation
Duration: 1 October 2023->1 May 2024
Leader: Tuğberk Dikmen Members: All Members
Objectives: To make sure that developers and stakeholders are on the same page documents need to be written.
Tasks: Task 1.1 Writing Project Specifications Document Task 1.2 Writing Analysis and Requirement Report Task 1.3 Writing Detailed Design Report Task 1.4 Writing Final Report
Deliverables: Deliverable 1.1 Project Specifications Document Deliverable 1.2 Analysis and Requirement Report Deliverable 1.3 Detailed Design Report Deliverable 1.4 Final Report

WP 2 - Front End Development
Duration: 1 November 2023->30 January 2024
Leader: Deniz Hayri Özay Members: Mete Arıkan, Tuğberk Dikmen
Objectives: Maximizing end user experience, designing Registered and Non-registered User related screen mock-ups, developing and implementing User related interfaces.
Task: Task 2.1: Gather with group members and create designs together with mockups. Get design feedback possible end users. Task 2.2: Implement the Registered and Non-registered User interfaces.
Deliverables:

Deliverable 2.1: Design mockups Deliverable 2.2: UI development

WP 3-Web Scraping Back End Development
Duration: 20 November 2023->15 April 2024
Leader: Furkan Yıldırım Members: Mehmet Ali Öztürk
Objectives: Implementing fully functioning Web Scraping algorithm
Task: Task 3.1: Testing selected Python libraries on a small amount of web data Task 3.2: Checking the consistency of the data Task 3.3: Increasing range of web scraping to desired data size Task 3.4: Merging this feature to the backend
Deliverables: Deliverable 3.1: Web Scraping algorithm for desired data types

WP 4-Project Demo
Duration: 1 December 2023->15 December 2023
Leader: Mete Arıkan Members: All Members
Objectives: Project status presentation to the stakeholders
Tasks: Task 4.1: Preparing presentation Task 4.2: Rehearsal of presentation
Deliverables:

Deliverable 4.1: Project Presentation Deliverable 4.2: Project Demo

WP 5-User and List Back End Development
Duration: 5 December 2023->30 January 2024
Leader: Mehmet Ali Öztürk Members: Deniz Hayri Özay, Furkan Yıldırım
Objectives: Implementing Registered and Non-registered User related back end functionalities.
Task: Task 5.1: Building a backend server Task 5.2 Development of the APIs and related functionalities
Deliverables: Deliverable 5.1: Backend server Deliverable 5.2: Registered and Non-registered User related functionalities

WP 6-Database Implementation
Duration: 5 December 2023->30 January 2024
Leader:Furkan Yıldırım Members: Mehmet Ali Öztürk, Tuğberk Dikmen
Objectives: Design a database structure & preprocess it for test purposes
Task: Task 6.1: Specify the information and functionalities that are kept and used by the database. Task 6.2: Designing the DB using ER diagram
Deliverables: Deliverable 6.1: A preprocess functional database

WP 7-Machine Learning Back End Development
Duration: 20 January 2024->20 April 2024
Leader: Tuğberk Dikmen Members: All Members
Objectives: Implementing machine learning system to build an efficient recommendation system
Tasks: Task 7.1: Creating models Task 7.2: Training models Task 7.3: Testing models
Deliverables: Deliverable 7.1: Machine Learning Recommendation System

WP 8-Admin Front End and Back End Development
Duration: 1 January 2024->1 April 2024
Leader: Mehmet Ali Öztürk Members: Mete Arıkan, Deniz Hayri Özay
Objectives: Designing admin related screen mock-ups, developing and implementing
Task: Task 8.1: Gather with group members and create designs together with mockups. Task 8.2: Implement the admin user interfaces. Task 8.3: Development of the APIs and related functionalities
Deliverables: Deliverable 8.1: Admin design mockups Deliverable 8.2: Admin UI development Deliverable 8.3: Admin related functionalities

WP 9-Testing
Duration: 1 April 2024->1 May 2024
Leader: Deniz Hayri Özay Members: All Members
Objectives: Test the functionality of a program and prepare it for the beta release
Tasks: Task 9.1: Testing all the components Task 9.2: Testing usability of functions by real users Task 9.3: Bug fix Task 9.4: Beta release
Deliverables: Deliverable 9.1: Bug reports Deliverable 9.2: Testing beta version of the application

WP 10-App Launch
Duration: 1 May 2024->15 May 2024
Leader: Mete Arıkan Members: All Members
Objectives: Make product ready for the use
Tasks: Task 10.1: Code optimization Task 10.2: Executable creation Task 10.3: Uploading app to Google PlayStore Task 10.4: Uploading app to AppStore
Deliverables: Deliverable 10.1: Final source code Deliverable 10.2: Cross platform application

4.4. Ensuring Proper Teamwork

In order to ensure the success of our project each team member plays a significant role, to achieve this creating an emphasis on effective collaboration is a key element. It is vital that all team members are aligned regarding individual workloads and associated deadlines, with the goal of increasing each member's strengths to ensure balanced participation. So that we have organized the team into two specialized subgroups: the front-end team and the back-end team. These subgroups were formed based on the previous professional experiences of our team members. The teams are as follows:

Back-end Team: Tuğberk Dikmen, Mehmet Ali Öztürk, Furkan Yıldırım
Front-end Team: Deniz Hayri Özay, Mete Arıkan

The primary responsibility of the front-end team is the development of the client-side interface of our mobile application, while the back-end team focuses on crafting the server-side logic. To cultivate robust teamwork, we have established three distinct communication strategies, including intra-group and inter-group discussions.

For these communications, we have chosen to utilize the Discord platform. Meetings within each subgroup are scheduled on a weekly basis, whereas inter-group meetings will be convened as needed. These sessions will serve as a forum for discussing implementation, business logic, deadlines, and overall project progress.

In addition to scheduled meetings, our team utilizes WhatsApp for day-to-day communication and coordination. This platform serves as our primary channel for asynchronous interactions.

Furthermore, our team employs GitHub not only for version control but also for project management. GitHub's built-in projects and issues features are utilized in a manner similar to Jira. Each issue is tagged as either front-end or back-end, allowing team members to filter issues based on labels, assignees, and reviewers. This system provides transparency regarding each member's progress and tasks. Labels such as 'Backlog', 'In Progress', 'In Review', and 'Done' are instrumental in facilitating clear communication and understanding of each team member’s current status and contributions.

4.5. Ethics and Professional Responsibilities

Our primary professional responsibility is to display the most recent product prices to inform the users accurately. We frequently need to get the updated price data from the markets, preferably once every day. If we neglect this, all price comparisons, list optimization algorithms, and item suggestions of PriceWise would give wrong results. The app would not be able to find the optimum shopping list. Therefore, PriceWise would be unsuccessful in general. Promising to help people save their budgets and then giving false directions that cause them to lose money would be highly unethical. It would also kill the usability of PriceWise and the credibility of our team.

Furthermore, we must retrieve user data only if the user allows us to do so. Then, we must use this data to only suggest alternative items for their lists and nothing else. We should not share user data with third parties.

Lastly, we should not output any biased result in favor of a grocery store that offers a bribe to us to show the users their products in the optimum solutions even when there are better alternatives.

4.6. Planning for New Knowledge and Learning Strategies

In this project, we use these following technologies:

- Flutter
- Amazon Web Service
- Mobile App Development
- Machine Learning
- Web Scraping
- Database Management

In order to progress effectively in the project, we need to have a certain level of knowledge on these issues. We can learn this information from courses (Machine Learning, Database Management) as well as from sources available on the internet. We also need to

know how to proceed and what to do when developing the project. We all have experience in this regard with the CS 319 course. This will not be the only information we will use in the project. After a certain time, we will have to apply issues that we do not even know the names of yet. We will solve those issues as we do now.

5. References

[1] “Her şeyi Karşılaştır,” Epey, https://www.epey.com/ (accessed Nov. 16, 2023).

[2] “CIMRI - Fiyat Karşılaştırma,” Google, https://www.cimri.com/market (accessed Nov. 16, 2023).