A Software Design Specification

Grocery List App

Course: Senior Project

Section: 02

Semester: Fall

Professor Perry

Date: 09/24/2023

Team: SP-5 Gold

Prepared by:

Jaedan

Nathan

Shawn

Table of Contents

1. Introduction

1.1. Document Outline

- Introduction
- System Overview
- Design Considerations
    - Assumptions and Dependencies
    - General Constraints
    - Goals and Guidelines
    - Development Methods
- Architectural Strategies
    - strategy-1 name or description
    - strategy-2 name or description
    - ...
- System Architecture
    - component-1 name or description
    - component-2 name or description
    - ...
- Policies and Tactics
    - policy/tactic-1 name or description
    - policy/tactic-2 name or description
    - ...

1.2. Document Description

Here is the description of the contents (by section and subsection) of the proposed template for software design specifications:

1.2.1. Introduction

The purpose of this document is to clearly communicate a basic understanding of the software design for a mobile grocery list. After reading this document there should be agreed upon design goals. The Software Design Specification plays a crucial role in ensuring that the software development process is well-organized, efficient, and results in a product that meets the intended requirements and quality standards. This document should guide and inform various stakeholders, including developers, testers, project managers, and clients, about the design, architecture, and functionality of the software. The scope of this document includes the Assumptions and Dependencies, General Constraints, Goals and Guidelines, Development Methods, Architectural Strategies, Policies and Tactics, System architecture and detailed design. Regarding system requirements, documentation can be found in the Software Requirements Specification document. This document should serve as a comprehensive document that provides the necessary information for development teams to implement the software system accurately and for stakeholders to understand the system's design and architecture.

Note:
For the remaining sections of this document, it is conceivable (and perhaps even desirable) that one or more of the section topics are discussed in a separate design document within the project. For each section where such a document exists, a reference to the appropriate design document is all that is necessary. All such external (or fragmented) design documents should probably be provided with this document at any design reviews.

1.2.2. System Overview

A sharable mobile grocery application is a convenient tool that allows users to create, manage, and share grocery shopping lists with family members, friends, or roommates. The mobile grocery application offers a comprehensive set of features for user convenience and collaboration. Users can register and create profiles, including personal details and preferred store locations. The app allows the creation and customization of multiple grocery lists, enabling users to organize items by categories and mark priority items. Collaboration is facilitated through shared lists with real-time synchronization and notifications for updates. The app offers intelligent item suggestions along with recipe integration for easy shopping. It boasts a user-friendly interface and cross-platform compatibility. Additionally, users can provide feedback for a better user experience.

2. Design Considerations

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

2.1. Assumptions and Dependencies

Assumptions and Dependencies can be found in the Software Requirements Specification document.

2.2. General Constraints

Describe any global limitations or constraints that have a significant impact on the design of the system's software (and describe the associated impact). Such constraints may be imposed by any of the following (the list is not exhaustive):

Platform Constraints:

- Operating System Compatibility: Developing for both iOS and Android platforms can be resource-intensive.
- Device Compatibility: Ensuring the app works well on various devices with different screen sizes and resolutions can be challenging.

Localization Constraints:

- Localization efforts to offer the app in multiple languages may be constrained by budget and resources.
- Adapting the app to local grocery store chains and products.

Technical Constraints:

- Internet Connectivity: The app's functionality may be limited in areas with poor or no internet connectivity.
- Device Capabilities: Some devices may lack necessary features like GPS for location-based services or barcode scanners.

User Experience Constraints:

- Design constraints related to providing a consistent and user-friendly experience across different devices and screen sizes.
- Accessibility constraints to ensure the app is usable by individuals with disabilities.

Data Constraints:

- Data Storage: Costs and limitations associated with storing and managing user data, especially if the app gains a large user base.
- Data Privacy: Compliance with data privacy regulations (e.g., GDPR) can pose constraints on data handling and storage.

Feedback and Iteration Constraints:

- Gathering and incorporating user feedback for continuous improvement can be challenging, especially with limited resources.
- Balancing feature requests with development capacity.

2.3. Goals and Guidelines

Described here are any goals, guidelines, principles, or priorities which dominate or embody the design of the system's software.

- The KISS principle ("Keep it simple stupid!")
- working, looking, or "feeling" like an existing product
- Efficient Shopping: Streamline the grocery shopping experience by allowing users to create, manage, and organize their shopping lists conveniently.
- Collaboration: Enable users to collaborate with family members, friends, or roommates in creating and maintaining shared grocery lists, ensuring everyone's needs are met.

- Organization: Help users categorize and prioritize items on their grocery lists, making it easier to find and purchase the items they need.
- Accessibility: Provide users with easy access to their grocery lists from their mobile devices anytime, anywhere, even while offline.
- Synchronization: Ensure real-time synchronization of shared lists across multiple devices, allowing users to stay updated on changes made by others.
- Item Suggestions: Offer intelligent item suggestions based on user history and popular items, simplifying the process of adding items to the list.
- Price Tracking: Enable users to track prices and expenses, helping them make informed shopping decisions and save money.
- Deal Notifications: Notify users of deals, discounts, and promotions at their preferred stores, helping them take advantage of cost-saving opportunities.
- Recipe Integration: Allow users to import recipes and generate shopping lists based on recipe ingredients, streamlining meal planning and shopping.
- User-Friendly Interface: Create an intuitive and visually appealing user interface to ensure a positive user experience and easy navigation.
- Cross-Platform Compatibility: Ensure the app is available on both iOS and Android platforms to reach a wider user base.
- Feedback and Support: Provide options for users to provide feedback, report issues, or seek assistance within the app, ensuring a responsive support system.

2.4. Development Methods

The approach decided for the mobile grocery list app is the Iterative and Incremental Development method. This approach divides the project into smaller increments or iterations. It allows for flexibility and adaptability as requirements evolve.

3. Architectural Strategies

For my team and I to meet our design goals while considering our experience with creating mobile applications we have decided to use Flutter to carry out most of our architectural strategies. Flutter can accommodate many of the factors and features we have agreed on for the mobile grocery list application. The app should feature efficient Shopping and Flutter's hot reload feature allows for quick iterations during development, which can speed up the process of creating and refining list creation and management features. It also has a rich set of UI widgets and customizable designs make it possible to create a user-friendly and efficient interface for managing lists and items. Flutter can be used to develop the frontend of the app, including collaborative features. For the backend, it is possible to integrate with cloud-based solutions or server-side technologies to manage shared lists and real-time synchronization. Provided by Flutter is the flexibility in designing the app's UI, making it possible to create well-organized and categorized lists and items. In addition, Flutter can interact with backend services and databases to retrieve and display organized lists for efficient data retrieval. It can also support offline capabilities by incorporating data caching and local storage. This allows users to access and modify their lists even when offline, with data synchronization occurring when an internet connection is available. Flutter can work with backend technologies that support real-time synchronization, such as WebSocket or Firebase Realtime Database. These technologies can ensure that shared lists are kept up to date across multiple devices. Flutter can integrate with backend services that provide item suggestions based on user history. User interactions with the suggestions can be implemented in Flutter's UI. The UI can be designed to display recipes and generate shopping lists based on ingredients. Flutter's extensive widget library and customizable themes enable the creation of visually appealing and user-friendly interfaces. Flutter excels in cross-platform development, allowing you to maintain a single codebase for both iOS and Android. It adapts to various screen sizes and orientations, ensuring compatibility. It can incorporate user feedback and support features, including communication channels, issue tracking, and user engagement components.

4. System Architecture

Below is a high level process control flow diagram (figure 1).

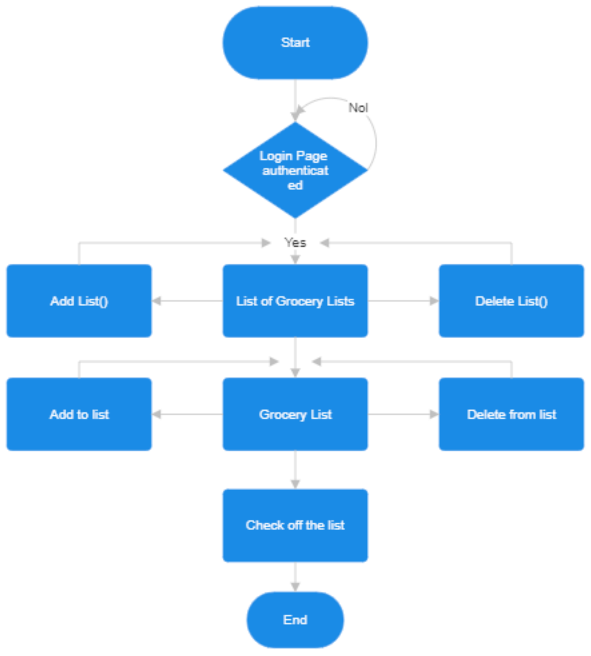
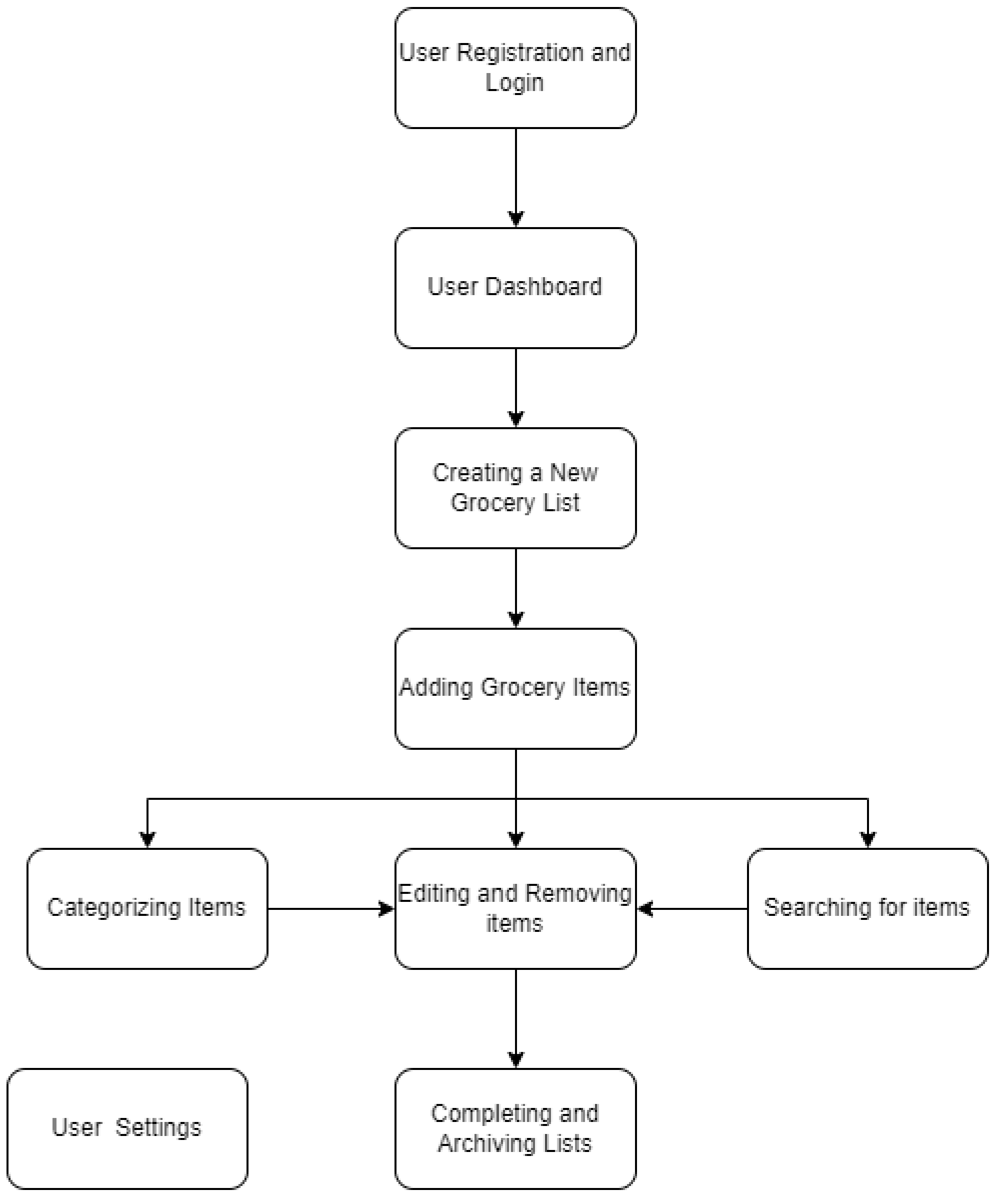


Figure 1.

5. Policies and Tactics

My team and I will be employing various policies and Tactics that we feel are necessary to carry out the desired goals. However, these factors will not have significant architectural implications on the overall organization of the system and its high-level structures. As my group has chosen Flutter to build our software application, it takes the responsibility of of which compiler and database to use. It uses a JIT compiler to enable my team to make changes to our code and see the results immediately without restarting the app. The JIT compiler is particularly useful for iterative development and debugging. While Flutter itself doesn't include a database system, my team can use third-party packages like sqflite to work with SQLite databases within Flutter apps. To ensure requirement traceability, document ids and labels will be utilized by assigning unique identifiers or labels to requirements, design documents, code modules, and test cases. These IDs facilitate cross-referencing and linking. The types of testing that will be employed is system testing, usability testing and acceptance testing.

Process flow and Mockup:


User Registration and Login
User Dashboard
Creating a New Grocery List
Adding Grocery Items
Categorizing Items
Editing and Removing items
Searching for items
User Settings
Completing and Archiving Lists

Users start by registering for an account or logging in if they already have one. User authentication ensures data security.

After login, users are directed to the dashboard where they can manage their grocery lists and access various app features.

Users can create a new grocery list by clicking on a "Create New List" button. They can give the list a name and choose a category (e.g., "Weekly Shopping").

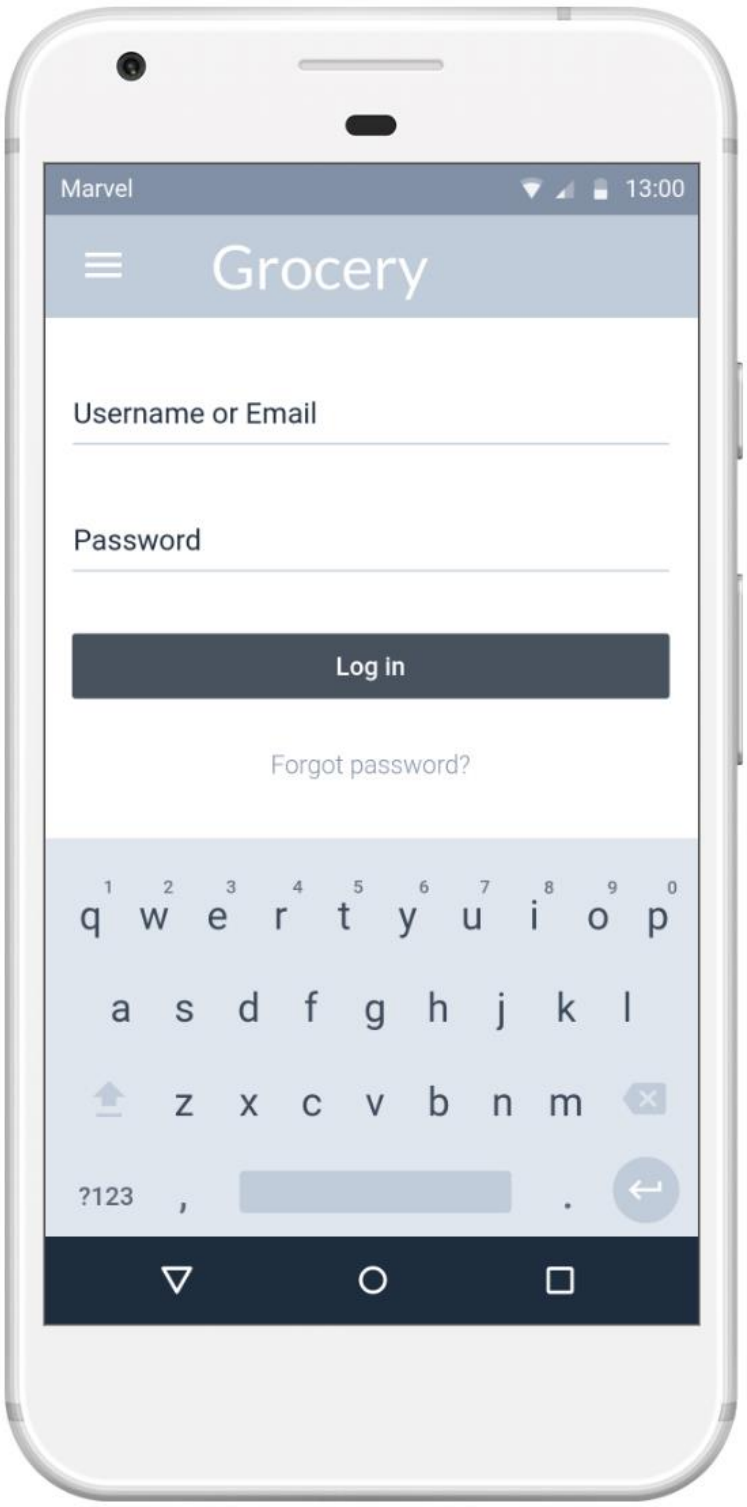
Within a grocery list, users can add grocery items by typing in the item name and specifying the quantity.

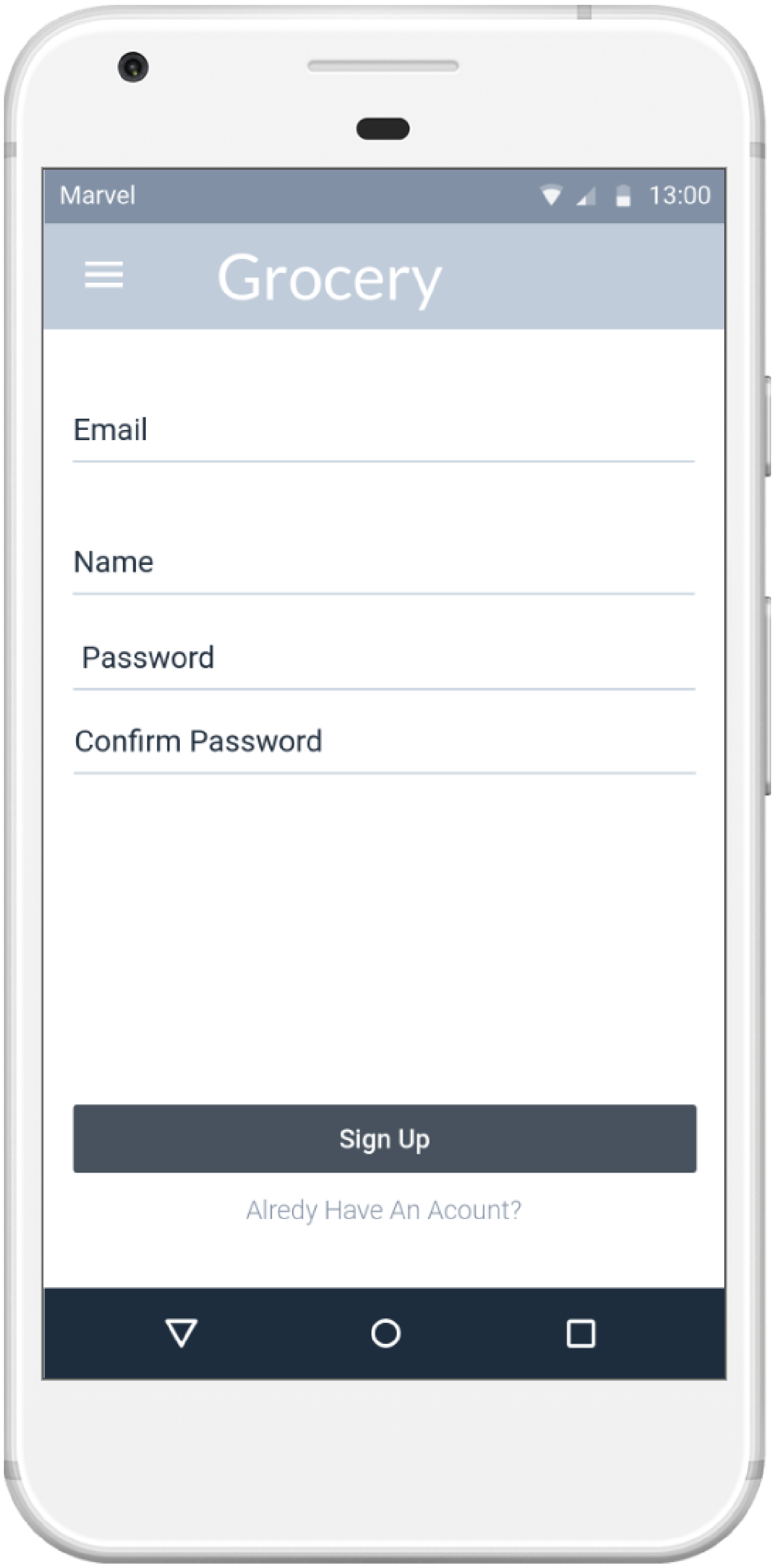
Users can categorize items within a list (e.g., "Produce," "Dairy," "Meat") for better organization.

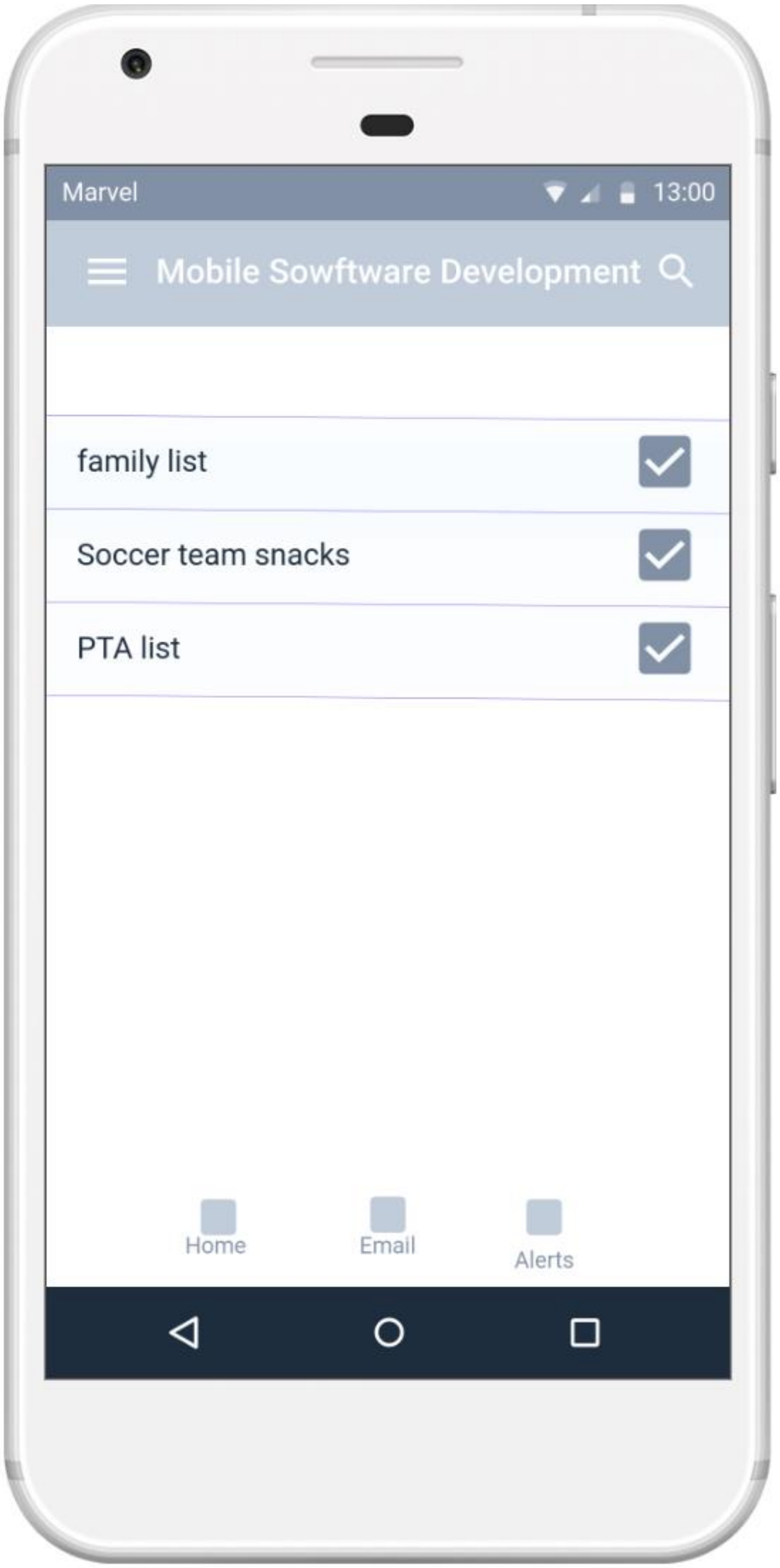
Users can edit or remove items from the list as needed. They can update the item name, quantity, category, or delete items entirely.

Users can use the search bar to quickly find specific items within a list, making it easier to locate items in a long list.

Users have access to settings to customize their experience, such as notification preferences, account management, and data privacy settings.


Marvel
13:00
Grocery
Username or Email
Password
Log in
Forgot password?

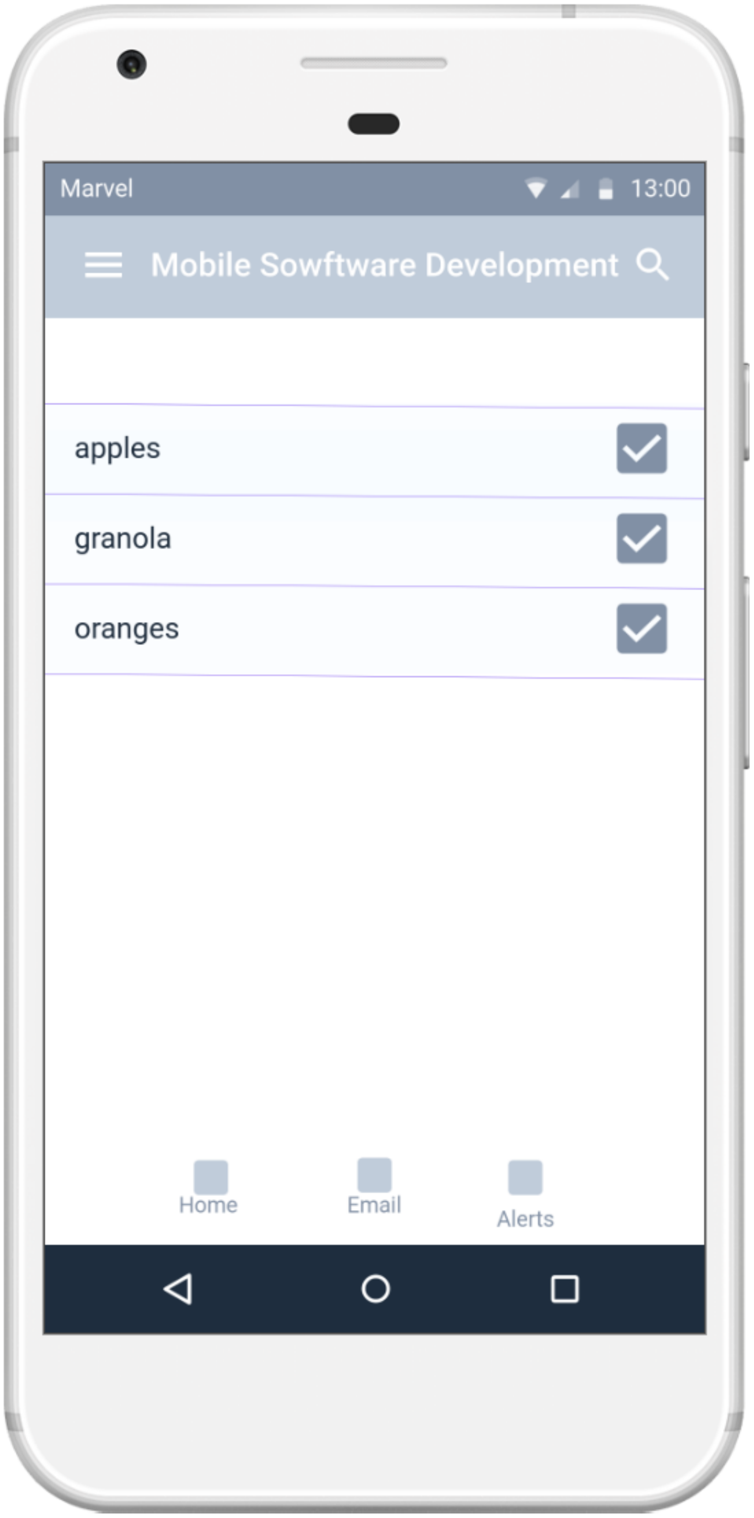

Marvel
13:00
Grocery
Email
Name
Password
Confirm Password
Sign Up
Alredy Have An Acount?



family list ✓

Soccer team snacks ✓

PTA list ✓



apples	✓
granola	✓
oranges	✓