Dynamic Grocery List

SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

CS 4850-02

Spring 2023

Professor Perry

02/10/2023

SP5-Yellow-Dynamic Grocery List

Table of Contents

1.0 Introduction

1.1 Abstract

The goal of the Dynamic Grocery List application is to ease the process of grocery shopping for roommates, families, friend groups, or individuals. The core objective of the project will be to develop a mobile application to allow users to create a shared grocery list. Additionally, the application should grant users the capabilities to view or edit the list, and to save the lists both locally as well as in a database. The application itself will be developed using the React framework while the associated database will be developed using Microsoft Azure. The primary objectives of this project will be: list design, user login, database design, website design, and app design/mockup. Once completed, secondary objectives of the application will be to explore monetization options, searching for coupons, and price comparison.

1.2 Project Goals

- Develop an app that allows for the base functions of creating, editing, sharing, saving, and deleting a grocery list.
- Develop a database to store created lists and information regarding connected users.
- Design a well-designed user interface which efficiently carries out the user's actions.

1.3 Definitions and Acronyms

VS Code - Visual Studio Code

SQL - Structured Query Language

RDP - Remote Desktop Protocol

HTTP - Hypertext Transfer Protocol

iOS - iPhone Operating System

1.4 Assumptions

Assumptions regarding the development of the project are the following:

- The development team will have access to and utilize the resources needed to complete this project (e.g., time, electricity, access to project software, etc.).
- The scope of the project will not change significantly throughout development.
- The database and application servers will be readily available at all times.

Assumptions regarding the users of this project are the following:

- Users will forget their username and/or password.
- Users will provide invalid data (empty lists, lists beyond capacity, etc.).
- Users will perform accidental actions (delete a list, share permissions, etc.).

2.0 Design Constraints

Anticipated constraints that will be needed to plan for through development include: Storage and cache sizes (offline use), screen size, appropriate gestures/interactions, battery usage, memory usage, and database communications.

2.1 Environment/System

The operating environment for the Dynamic Grocery List is as follows

- Operating System compatibility - iOS and Android
- Development platform - VS Code, React, JavaScript
- Database design - Microsoft Azure

2.2 User Characteristics

Application users should be able to create a list, add any desired items, and then save the list. The user should also have the additional options to delete, undo delete, and share a list. List sharing also allows the user to grant view only or editor privileges to any users they share a list to. Created lists will then be saved locally as well as on the database Users will also have the option to mark lists as completed or incomplete and sort them as such.

3.0 Requirements (Include screen mockups and requirement details)

- A user must be able to login to the application using an user defined username and password.
    - Each user must have a unique ID associated with this login.
    - See Mockup in Appendix, Figure 1: Login Screen.
- The program must display created user lists upon login.
    - The program must display any previously created lists or display “no lists available”.
    - See Mockup in Appendix, Figure 2: Home Screen.
- The program must allow the user to create a list for desired items.
    - The user must be able to create/name lists and add up to 50 items.
    - See Mockups in Appendix, Figure 3: Create Function and Figure 4: Create Result.
- The program must allow the user to edit a list itself and items within a list.
    - Edits should include items, list properties (e.g., title, access, etc.)
    - See Mockups in Appendix, Figure 5: Edit Function and Figure 6: Edit Result.
- The user must be able to delete items from the list as well as delete the list itself.
    - Items should be deleted immediately while lists should be moved to a trash bin.
    - See Mockups Appendix, Figure 7: Delete Function and Figure 8: Delete Result.
- The user must be able to share their list to other users.
- Users must be able to designate roles for shared lists (e.g., viewer/editor).
    - Viewer and editors will have distinct permissions to act on a shared list.
- After creation, lists must be saved both locally and in the database.
- Users must be able to search for a certain list as well as items in a list.
    - The program should display results in order of relevance according to input.

- Users may enable notifications for actions performed on a list.
    - i.e., user should be notified when a list has been shared/unshared with them
- The users' information must be encrypted.
    - The user's personal information must be secured to prevent outside access.
- Access to the program database must be protected and recorded.
    - Access should only be granted to specific individuals, and any actions taken by said individuals must be kept in a log.
- The program database must backup data at least daily.
    - Backups will maintain a screenshot of the database which can be used to recover lost or damaged information.
- The integrity of the data should be maintained by the database.
    - Data integrity will be maintained by handling synchronous actions to a shared list.
- The size of the database must grow in accordance with the number of users.
- The user's device should contain local storage of at least 500 MB.
- The user's allocated space on the database must adhere to the project's policies.
- The user interface should be clear, consistent, and responsive.
    - The interface must be designed so that it is simple and intuitive to users.
- The user interface should disguise the inner workings of the program.
- The program should be compatible with the latest versions iOS and Android Operating Systems.
- Under normal operation, each given page must load within 2 seconds.
    - The program should not have excessive load times that hinder user experience
- After the app crashes, the app must load in between within 10 seconds.
    - Extra time should be allowed for the program to reload/resync data
- The server hosting the database should be able to carry the load of all active users.
    - The server database should be able to allocate/deallocate space as needed.

APPENDICES

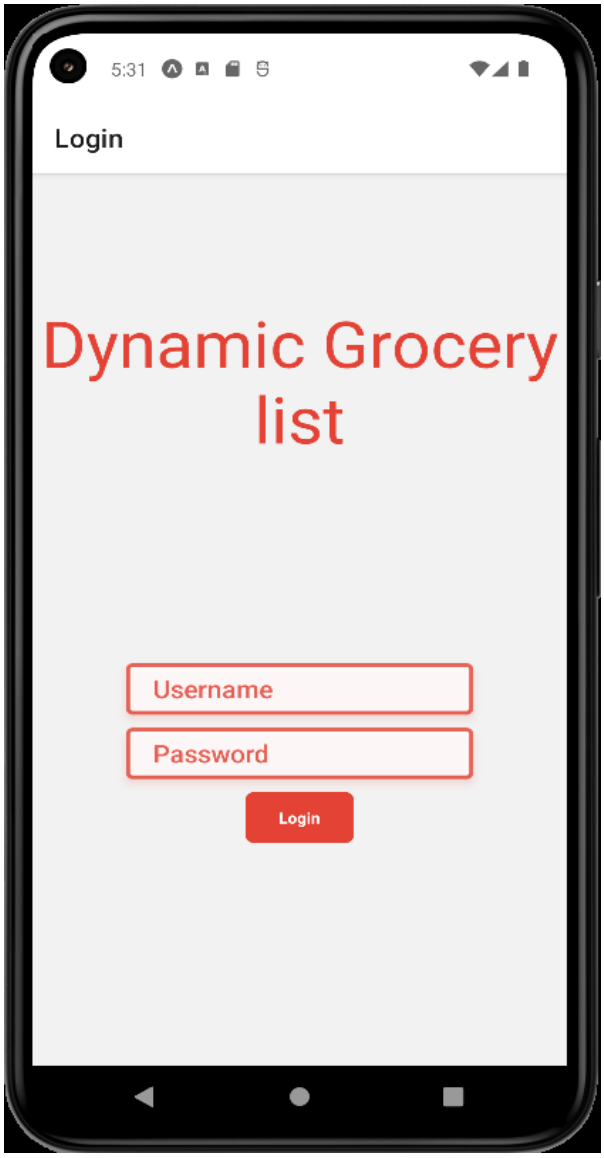

Figure 1: Login Screen

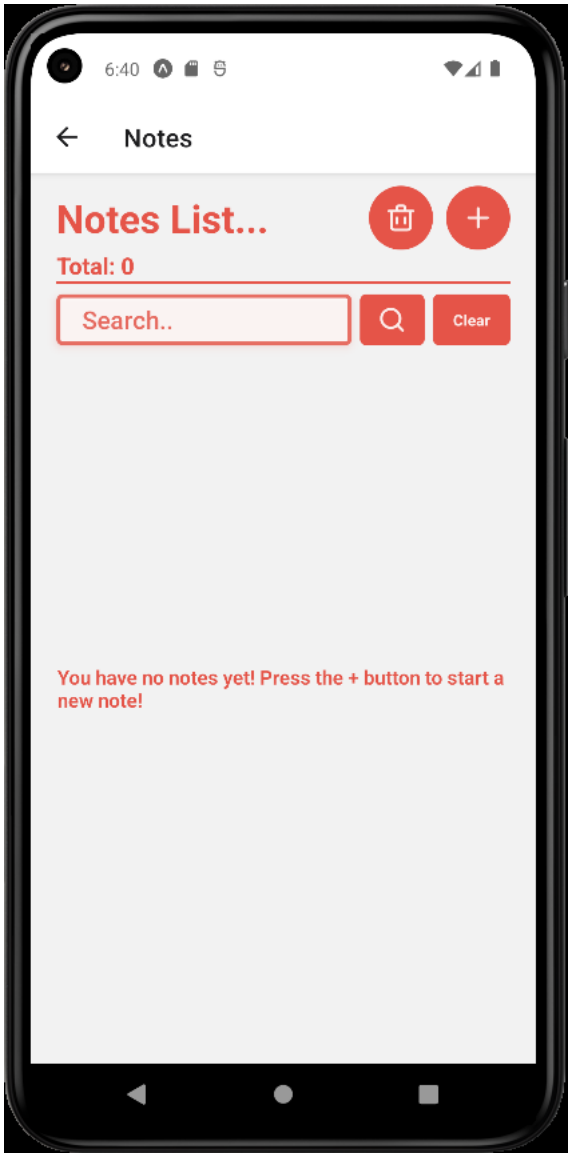

Figure 2: Home Screen

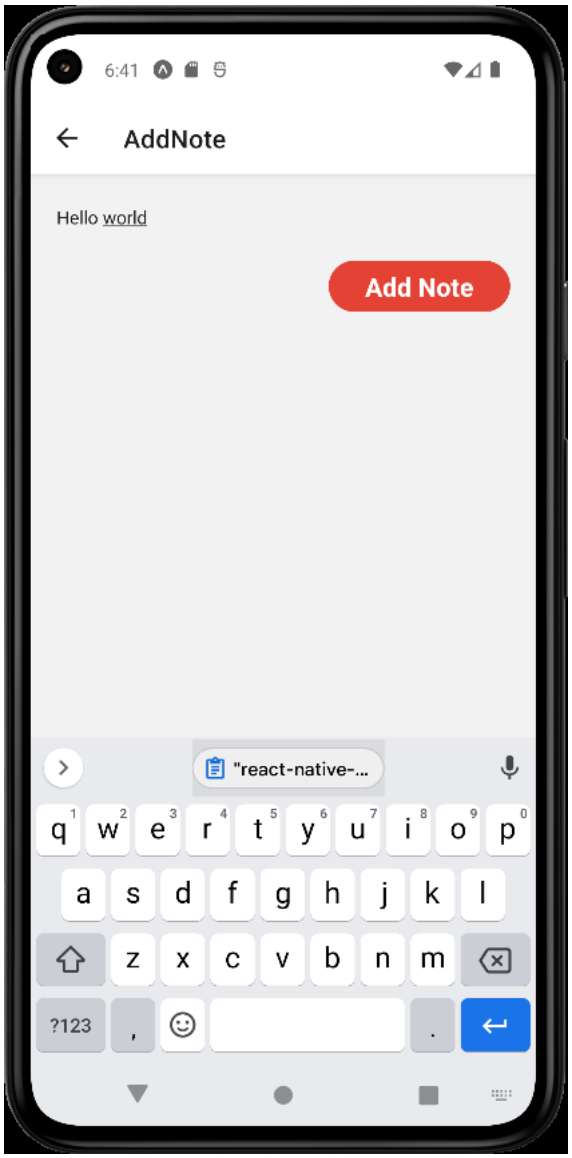

Figure 3: Create Function

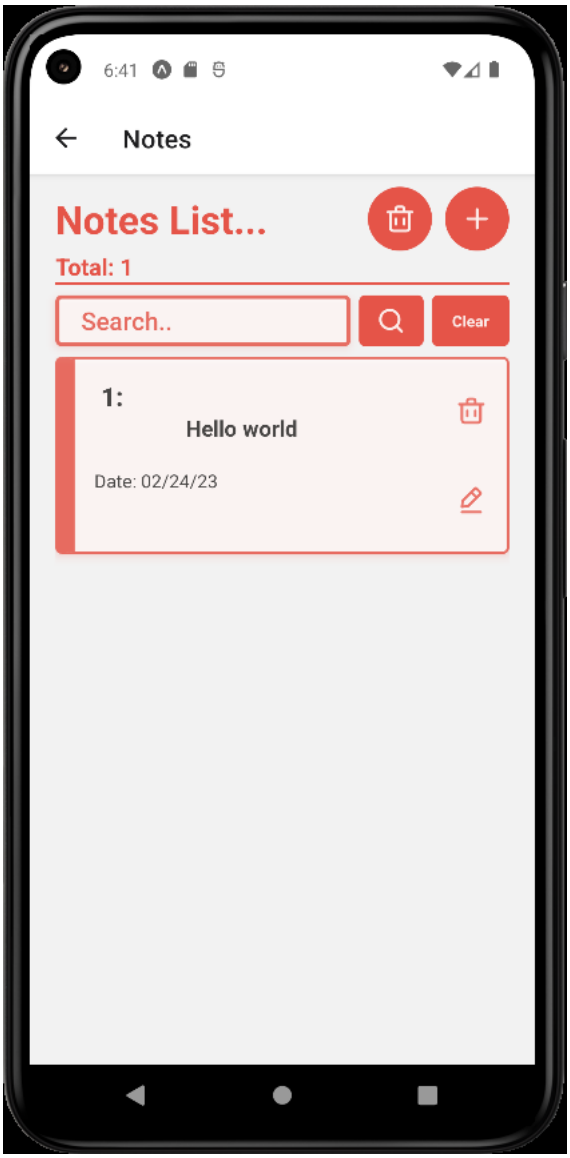

Figure 4: Create Result

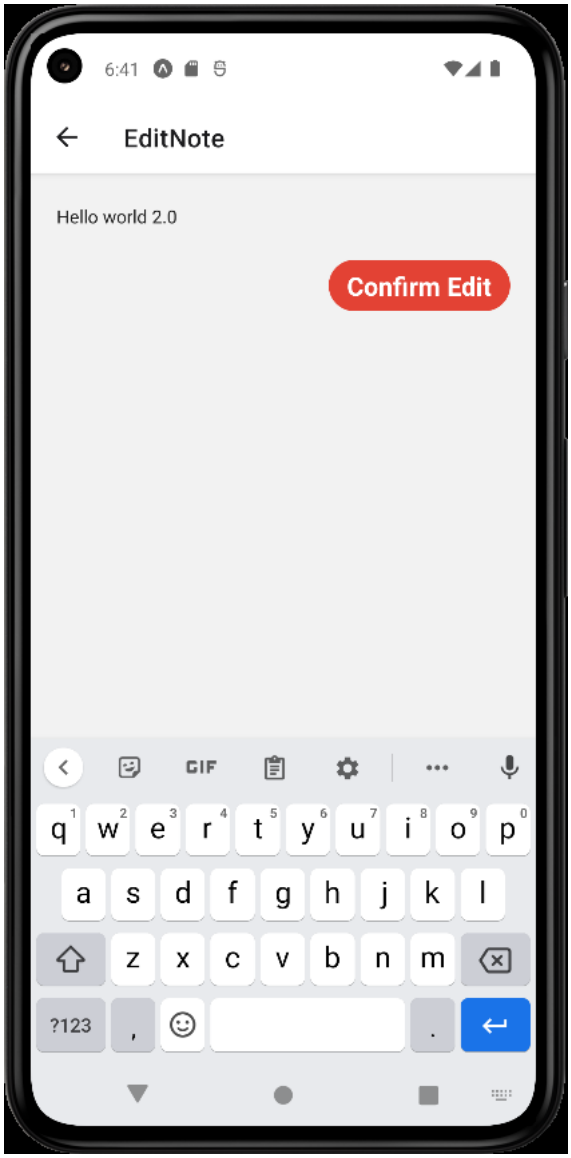

Figure 5: Edit Function

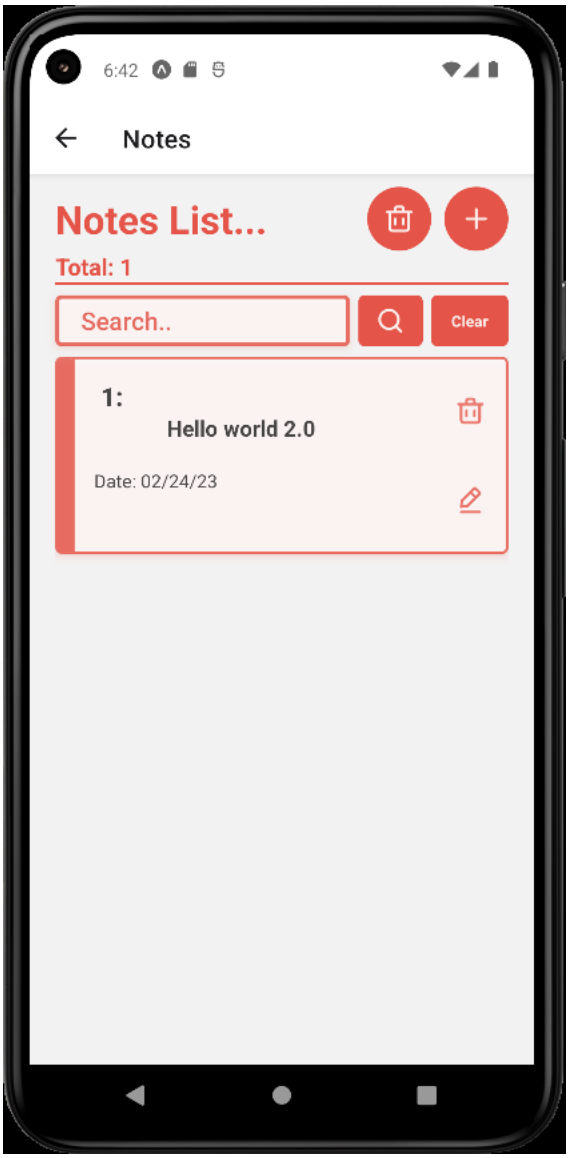

Figure 6: Edit Result

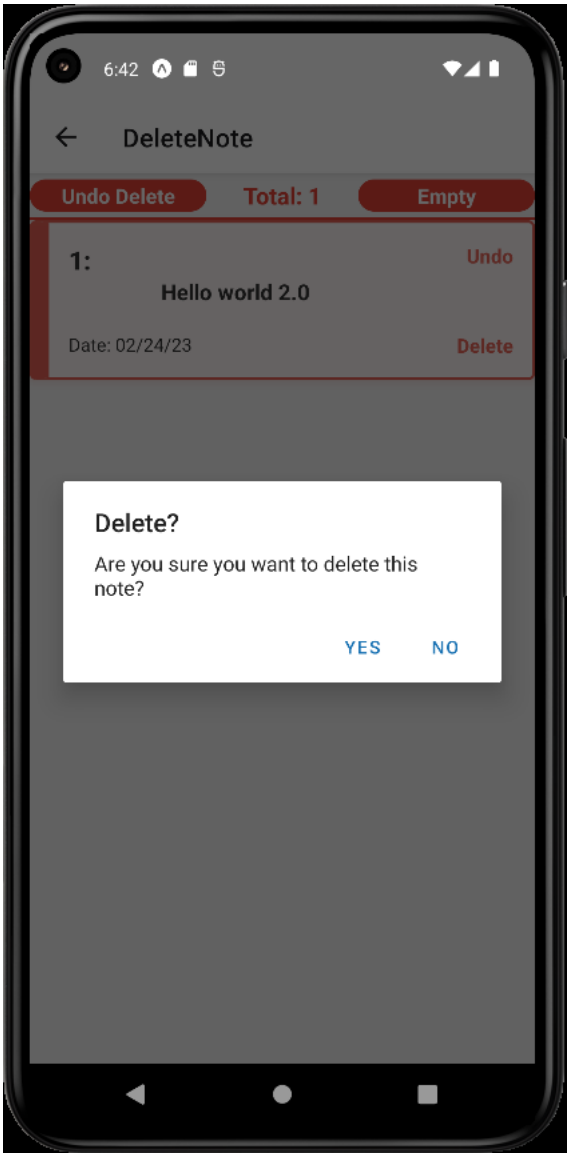

Figure 3: Delete Function

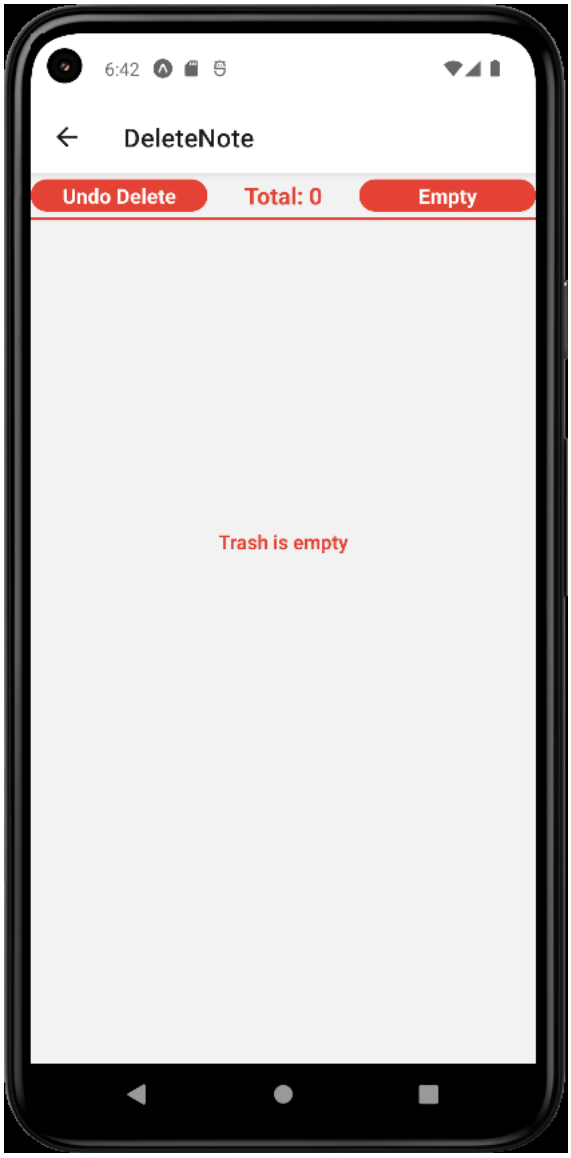

Figure 4: Delete Result