SP-5-Orange - Dynamic Grocery List

SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

Course: CS 4850

Semester: SPRING 2023

Professor Perry

Date: 2/24/23

Team: SP-5-Orange

Table of Contents

1.0 Introduction

1.1 Overview

This document defines the requirements for our Dynamic Grocery list application. A grocery list app built with React Native that allows users to easily manage their shopping lists on their mobile devices. The app features a clean and intuitive user interface that allows users to quickly add items to their list, as well as organize and categorize items for efficient shopping. The app also includes a variety of useful features such as the ability to share lists with family and friends. Additionally, the app utilizes the power of React Native to provide a smooth and responsive user experience.

1.2 Project Goals

To effectively design an application that allows users to access a customizable grocery list that enables users to edit, organize and share lists with others.

1.3 Definitions and Acronyms

Firebase- A set of hosting services for many applications such as Android, iOS, Javascript. Firebase offers hosting of databases, content, social authentication, and notifications.

Auth2.0- adds authentication and authorization services to applications

React-Native- Framework used to develop application

1.4 Assumptions

It is assumed that to access this app, an Android or i0s operating system must be equipped to the user's devices.

2.0 Design Constraints

2.1 Environment

Dynamic grocery list application will target mobile environments of Android 5.0 and higher as well as iOS versions 12.4 and higher. A stable internet connection will also be required in order to view updated shopping lists and for the lists to be updated when adding or removing items for other users.

2.2 User Characteristics

The users of the application will be anyone with a mobile device who wishes to have a place to quickly see what needs to be purchased when shopping for groceries. This is also for users that shop for themselves as well as others with lists in the app being collaborative. The target users are very broad with the goal being to be accessible to both major mobile operating systems and a friendly easy to use user interface.

2.3 System

The system will be creating using React Native, Expo, and Firebase. Using these three technologies it will be possible to create native applications that run on both iOS and Android seamlessly. With the use of Expo it adds a set of tools on top of React Native to ease the process of creating the app as well as testing of React Native applications. To store the data Firebase can store the data inputted by users securely and sync data across all devices.

3.0 Functional Requirements

3.1 Frontend Requirements

- Startup screen while launching
- Login screen
- Account creation screen
- Authentication- auth0
- Forgot password screen
- Verification email (once user signs up)
- Home screen shows an organized view of grocery lists with option to create new lists
- Users can create and add other users in group
- All users that are in group can access and edit grocery list
- In-app text editor for lists
- Sharing screen with information about who the document is shared with and functionality to add or remove users

3.2 Backend Requirements

- Create, delete, and edit list entries
- Sync list entries with all users
- Add and remove users to a list
- Update list live when list collaborators edit lists
- User authentication and registration
- Store user account info

4.0 Non-Functional Requirements

4.1 Security

- All personal data will be encrypted
- Users will be verified upon creating an account

4.2 Capacity

- The app should have a capacity of at least 50 logged-in users at a time

4.3 Usability

- Upon launching the app, it should reach the home screen in under 15 second

4.4 Other

- The app should have at least a 99% uptime
- The app will be compatible with iOS 12.4 or Android 5.0 and above

5.0 External Interface Requirements

5.1 User Interface Requirements

The user interface should be simple and sleek. It will include all necessary features, but avoid overwhelming the user with too many options or settings. More specifically, this means avoiding adding features that aren't necessary, as well as making sure the features that we do have are laid out in a logical manner. The app should be intuitive, so that a user can make and share a list within 5 minutes of opening the app with no training.

5.2 Hardware Interface Requirements

The app should not use an excessive amount of system resources. It should efficiently use what is available, without taking up more than its fair share of CPU time or memory.

5.3 Software Interface Requirements

All code will be efficient and wee

5.4 Communication Interface Requirements

All communication involving sensitive user data will be encrypted. Data transfer, such as uploading and downloading lists, will be fast and efficient.

6.0 Test Plan

6.1 Sign Up/Log In Tests

- Create an account
- Verify account via email
- Log in to previously created account

6.2 List Creation/Organization Tests

- Create List
- Access pre existing lists
- Ability to add and remove items in list
- Display customized list

6.3 List Sharing Tests

- Share list amongst others in group

APPENDICES