4850-03

Fall 2024

SP5– Monetized Grocery App

Professor Sharon Perry

Date: December 2nd, 2024

Group Members: Jarely Sanchez, Eric Sinkler, Yaya Barry, Seth Walker, Kemi Araba-Owoyele

Website Link

GitHub Link

lines of code:

of tools used

# of man hours:

Table of Contents

Introduction

Basket, the grocery shopping app, is crafted to redefine and elevate the grocery shopping experience. This app aims to make shopping more organized, efficient, and enjoyable. This app provides a suite of features that enables users to create, edit, and manage shopping lists effortlessly while seamlessly collaborating with family or friends in real-time. Through real-time notifications, users can stay updated on any list changes, such as items added, modified, or purchased by others, fostering effective coordination.

Basket also offers customizable reminders, allowing users to set alerts for list preparation or store visits, making it easier to stay on top of shopping routines. In addition, a range of personalization options allows users to tailor the app to suit their unique shopping preferences. Built with a strong emphasis on reliability and data security, the app is designed to provide a smooth experience across both iOS and Android platforms. Whether for individual users managing their weekly groceries or groups coordinating long lists in real-time, Basket ensures that users can easily track changes, stay informed, and simplify every aspect of their grocery shopping journey.

Requirements

Functional Requirements

User Registration

- Users will be able to register for an account with a valid email address and create a secure login/password.
- Users will have the option to log in using their email accounts
- Secure passwords will be user created

Home Page

- The app will display a user-friendly home page with options to navigate to their lists, notifications, user profile, settings
- Recently accessed lists will be prominently displayed on the home page.

Shopping List Management

- Users will be able to create, view, and edit shopping lists.
- Real-time synchronization of lists between users must be supported.
- Users will be able to add, remove, and mark items as purchased within a list.
- Users will be able to collaborate on a list in real time

Notifications

- Users will receive notifications when items are added or checked off or removed in shared lists.
- Notifications will be toggled on or off in app settings.

Reminders

- Users will have the ability to set reminders for creating new shopping lists or store visits.
- Reminders will be customizable with date and time settings.

User Profile

- Users will be able to view and update their profile, including a profile picture and name.
- Users will be able to change their password.

Settings

- Users will have the option to log out from the app.
- Users will be able to customize app settings, including notification preferences.

Non-Functional Requirements

Security

- Password requirement- must be at least 8 characters, contain a number, capital letter, and symbol

Capacity

- Max of 10 lists
- Max of 500 entries in a list
- Max of 100 characters per entry
- Max of 10 people can have access to a list

Usability

- Users should easily be able to navigate the user interface with ease
- Easily be able to share/ join lists
- 80% of users should be able to use 75% of features

Performance

- The app must respond to user interactions within, at most, 4 seconds.
- Real-time list synchronization should occur with minimal delay (less than 5 seconds).

Compatibility

- The app must be compatible with iOS (version 12 and above) and Android (version 8 and above) devices.

Analysis

Methodology

Basket was created using the Agile Methodology. Agile is a methodology that one can see and expect often in real world scenarios. Agile is iterative and allows teams to deliver a product in chunks or “sprints”. These sprints allow teams to adapt and make necessary changes to a project while still maintaining core functionality or scope. Agile also allows for end users to give developers feedback in real time that allows developers to push fixes and new features as quickly as needed. Our team decided to use Agile for its efficiency and to get more acquainted to a methodology that we will most likely employ in the real world.

App System Overview

Basket has been divided into three core systems:

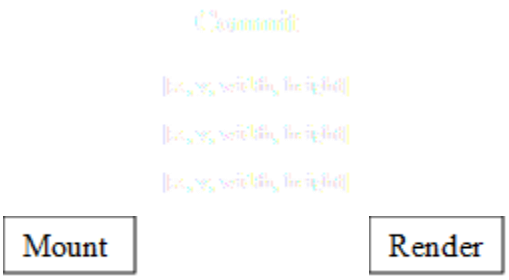
- Account Management
    - This system handles user authentication, account security, and sensitive information protection. Key functionalities include error detection and password recovery, ensuring a secure user experience.
- List Management
    - List Management system enables users to create, edit, and share shopping lists through an intuitive interface. By focusing on simplicity and functionality, this system aims to enhance productivity and collaboration among users.
- Notification Management
    - Notifications keep users informed about real time updates to shared lists, ensuring synchronization across devices. This system supports real-time communication, enhancing usability and convenience.

Implemented Technology

Our team decided to employ the use of React Native which is a framework well known for its ability to be efficient and able to be adapted cross-platform. React Native uses a combination of JavaScript and C++, which allows developers to create responsive and high-performance applications. Its architecture includes the following phases:

- Render Phase
    - Application logic is executed, producing a React Element Tree in JavaScript, which is transformed into a React Shadow Tree (RST) in C++.
- Commit Phase
    - The RST and React Element Tree are promoted for rendering.
- Mount Phase
    - The RST layout is calculated and converted into a Host View Tree, which is then displayed to the user.

Native React Architecture



Future Expectations

At current state, the functionality and features of Basket are simple. Our team has designed the app this way to be scalable. The scope of app can be expanded based on user feedback. Additional features and systems can be incorporated based on user feedback and emerging requirements. This flexibility ensures that the app remains relevant and valuable to its users over time.

Design

Design Interface

Account Management

- Registration: Registers a new user in the app
    - Parameters: Username, password
    - Returns: Home page or error notification
- Update Password: Updating a user's profile password
    - Parameters: Username, current password
    - Returns: successful update or error
- Get User Profile: retrieving the user's information
    - Parameters: Username
    - Returns: Profile data

Grocery List Management

- Create List: Creates a new grocery list
    - Parameters: Username, list name
    - Returns: A new list ID
- Delete List:
    - Parameters: List ID
    - Returns: Successful delete or error
- Share List: Shares a grocery list with different users
    - Parameters: Username, list ID
    - Returns: Successful share or error
- Add Item: Adds a new item to the grocery list
    - Parameters: List ID, item name/description
    - Returns: Item ID
- Update Item: Changes the description of an item
    - Parameters: List ID, item ID, new item name/description
    - Returns: Successful update or error
- Delete Item: Deletes an item from the grocery list
    - Parameters: List ID, item ID
    - Returns: Successful update or error

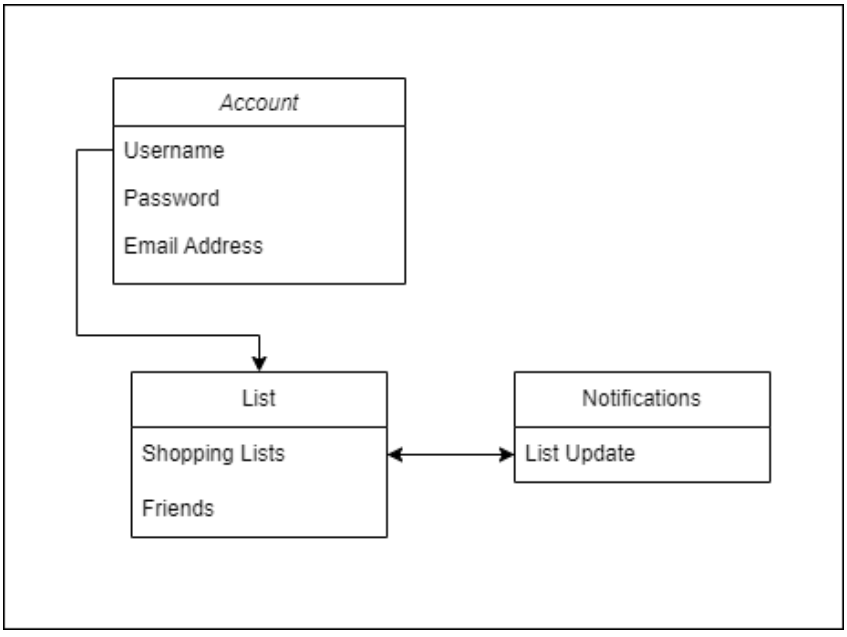
Notification Management

- Send Notification: Sends a notification to all of the users that have access to a list when any changes are made
    - Parameters: List ID

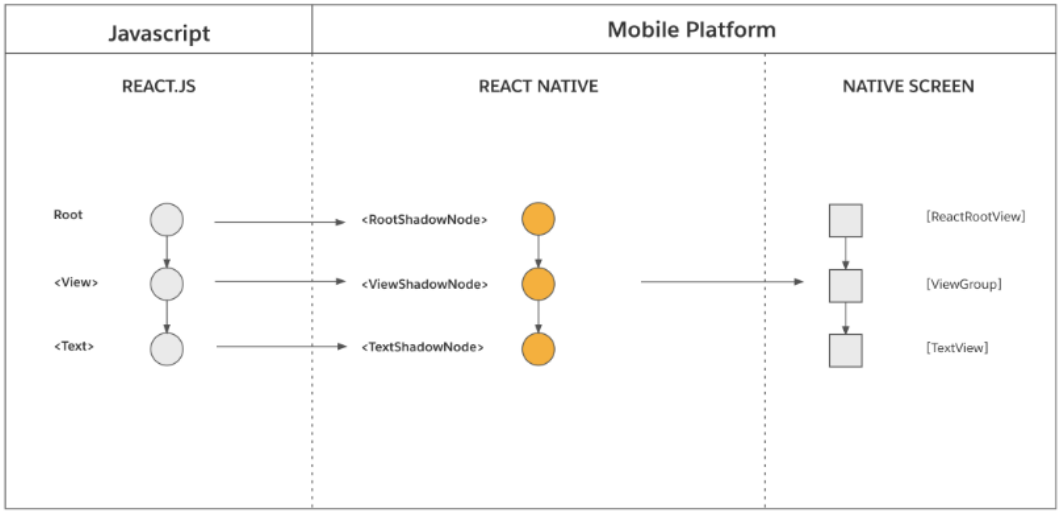
System Design

This system will contain the following modules

- **Account Management**
    - Purpose: Allowing the user to register, log in, and manage their profile and the lists that they have access to.
    - Semantic: Users will be able to access the app features with their account. This module also ensures that they will be able to view and edit the lists that they have the proper permissions for.
- **Grocery List Management**
    - Purpose: Managing the creation of new lists. This also involves sharing, modifying, and deleting these lists.
    - Semantic: This will cover the main functions of the app, allowing the users to create new lists and share them with other users.
- **Notification Management**
    - Purpose: Managing the notifications that result from sharing lists and making changes to them.
    - Semantic: Allowing notifications helps with the collaboration feature of this app by keeping the users informed about the changes that have been made to the lists.



System Interactions Diagram



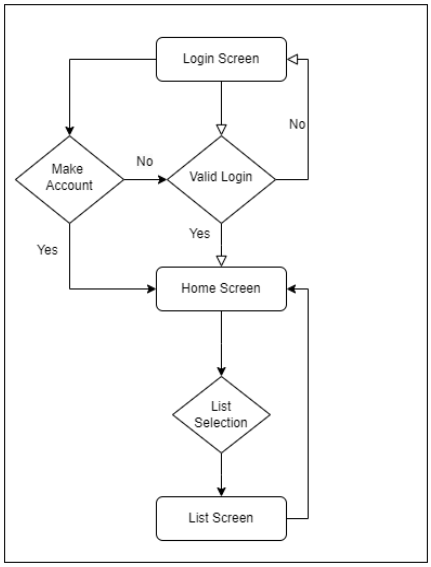
Design Resources

- **Account Management**
  - Database to store user credentials that are used when creating accounts and logging in
- **Grocery List Management**
  - Database to store list data and permission
- **Monetization**
  - Google AdMob
  - Google Play Store
  - Apple App Store
- **Software/IDE's**
  - Android Studio
  - Visual Studio

Design Constraints

Account Management

- Must handle errors when the user creates an account. This means no duplicate usernames and passwords must match the criteria.
- Must handle errors if the user makes mistakes when trying to log in to their accounts.
- Must synchronize the accounts promptly, allowing users to view newly shared lists.



Grocery List Management

- Must allow the storage of lists properly regardless of how many items there are.
- Ensure that lists are synchronized in real-time when any changes are made.

Platform-Specific Guidelines

- Must adhere to Apple's Human Interface Guidelines for a consistent and intuitive user experience on iOS devices.
- Must adhere to Google's material design principles for Android for a cohesive and familiar user experience.

Data Privacy and Security

- All sensitive data must be encrypted in both transit and at rest to protect against unauthorized access from attacks.
- Must implement secure authentication mechanisms to ensure that only authorized users can access their accounts.

Resource Constraints

- Must be optimized to minimize battery consumption, particularly during real-time synchronization and background notifications.
- Must optimize data usage to accommodate users with limited data plans or poor connectivity.

Accessibility

- Must comply with the WCAG, Web Content Accessibility Guidelines, to ensure users with disabilities can fully utilize the app and all its capabilities.

Integration with Third-Party Services

- Respect rate limits of third-party service APIs by implementing thoughtful design and proper error handling of any requests sent.
- Minimize dependency on unnecessary external libraries to reduce risk of vulnerabilities, performance issues, and deprecations.

Legal and Regulatory Constraints

- Must comply with the guidelines and policies of the Apple App Store and Google Play Store.
- Must adhere to consumer protection laws in regions where the app is accessible.

Development

Development Details

User Interface:

- All screens implemented using React Native and React libraries.
    - Login Screen:
        - Title of the app and picture of bag of groceries.
        - Two text inputs to type in username and password.
        - Login button: to login once username and password have been entered.
        - Create Account button: if the user does not already have one.

List Screen:

- Container: designed like a notepad with title of list at the top.
- Add Item button: when clicked creates a textbox where user can type in the item and creates a checkbox next to the item.
- Checkbox button: Togglable checkbox to mark whether an item has been bought or not. Will also add a strikethrough to the item text associated with the checkbox
- Delete Item: Tap on text to edit, then backspace. If backspace is pressed again when there is no longer any text, item will be deleted from the list.
- Back button: Takes the user back to the home screen when pressed.
- Share button: Gives user option to select a contact to share current list.

Home Screen:

- Title: Header at top of page “My Lists”
- Lists:
    - Separate buttons in for each list the user has created, designed to look like a sticky note.
    - Can be clicked to take user to the list screen for that list.
    - Can be long pressed for options to edit the name of the list, delete the list, or share the list
    - Icon appears in corner of list container if it has been shared
- Add list button: when clicked adds a new list with a default title
- Settings button: opens the setting menu

Settings:

- Account settings:
    - Option to edit username and password
    - Logout button
- Option to change the background color of the app

Account Management:

- Uses Firebase account management system

Create Account:

- Prompt the user for an email, username, and password.
    - User must verify their password
        - Password must be at least 8 characters, contain a number, capital letter, and symbol
- Take information provided by user and use firebase method "createUserWithEmailAndPassword" passing the information to create a new user

Delete Account:

- Get instance of current user and then call firebase "delete" method on current user.

List Management:

- Uses Firebase Realtime Database which is structured as a JSON tree
- Create Lists:
    - When a user creates an account, they are added to the tree, then when they go to create a list, that list gets created as a child of their account with a unique ID.
- Delete Lists:
    - When a user chooses to delete a list, the data is moved to a recycle bin for a set period before being permanently removed from the tree
- Share Lists:
    - With each list having a unique ID in the Realtime database, that can be shared with other users allowing them to access the list.
    - Using Firebase they can reference the list using "database.child("users").child(userId).child(listId)"
    - Then users can use setValue methods to edit the list.

Monetization:

- Google Mobile Ads with React Native and Expo
- Banner Ads:
    - Create a banner ad using Google AdMob and give it an ID

- Insert banner ad into view of home screen/list screen using ID from AdMob

App Creation Steps

1. **Install necessary tools:**
   - Node.js
   - Android studio
   - Expo CLI
   - VS-code (IDE)
2. **Create a React Native application using Command Prompt:**
   - Type:
     - "npx create-expo-app "GroceryListApp"
     - " npm run android" on windows
     - "npm run ios" on mac
3. **Set up Firebase or MongoDB**
   - Install dependencies
   - Navigate to Firestore Database in the Firebase Console
   - Add Firebase to the Expo project
4. **Set up User interface basics**
   - Create user login screen
   - Create user home screen
   - Create user list screen
5. **Set up AdMob**
   - Create a Google AdMob account
   - Publish app on App store and Google play store
   - Link App ID to Google AdMob
   - Create Banner ad in AdMob
   - Add banner ad to different screens

Test Plan/Report

Requirements	Pass	Fail	Severity
Create Account		x	major
Login		x	major
View Lists	x		
Add Lists	x		
Delete Lists	x		
Open Lists	x		
Add Items	x		
Delete Items	x		
Checkmark Items	x		
Share List		x	major
View Collaborators		x	major
Edit Shared List		x	major
View Shared Changes		x	major
Receive notifications on changes to shared lists		x	minor
Creating reminders		x	minor
Return to Homepage	x		
Open Settings	x		
View profile	x		
Update password		x	minor
Logout	x		
Customization		x	minor

Version Control

What is Version Control?

Version Control, or VC, is a system that allows us to collaborate on the project, keeps track of every modification and can restore the previous version if needed. For example, Git is the most popular version control, that is the main reason we used GitHub to collaborate which allowed us to work on the same project without conflicting changes.

GitHub

Our team decided to use GitHub for our version control. Each change that a team member made to the code meant a new branch was to be created. This allows for the team to keep a stable copy of our original work separate from any untested changes that could affect app functionality. Currently our team is using three repositories. Two of those repositories help with tracking both front end and back-end changes. The third repository acts as a middleman for our API server.

Conclusion

In wrapping up our project, Basket, we're proud of how far we've come in creating an app that makes grocery shopping simpler and more efficient. From real-time list updates and reminders to easy collaboration with family or friends, Basket combines a lot of useful features that we believe will help people manage their shopping lists better. Our group focused on making the UI and features user-friendly and accessible so whether someone's grocery shopping for themselves or collaborating on a list with others, they can rely on the app for a smooth experience.

Building Basket gave us real-world experience in several areas. Our team approached this project with a focus on practicality and usability, designing a user interface in Figma to prioritize easy navigation and responsiveness. We employed and ensured data security using Firebase for seamless, real-time syncing. Through the creation of this app, we also have explored monetization through Google AdMob. This has taught us how we could balance a positive user experience with ads in the app.

Working on Basket has been a valuable learning experience. We were able to put into practice many of the skills we've learned in our courses, like teamwork, coding, and problem-solving. Basket reflects our team's hard work, and we're excited about the potential it has to genuinely make grocery shopping easier for the everyday person.

Appendix

Project Plan

Deliverables

- Team/Project Selection document (Individual Assignment)
- Project Plan (Group Assignment)
- SRS, SDD, STP & Dev Doc (Group Assignment)
- Weekly Activity Reports (WARs – Individual Assignment)
- Team Status Report (TSR – Group Assignment)
- Peer Reviews (Individual Assignment)
- Present Prototype for Peer Review (Group Assignment)
- Final Report Package (Group Assignment)
- Final Report (Group Assignment)
    - Screen Mockups (Individual Assignment)
    - Tutorial (React) Evidence (Individual Assignment)
    - Source Code (Group Assignment)
    - Website (Group Assignment)
    - Video Demo (Group Assignment)
    - iOS and Android compatible mobile grocery list app

Milestone Events (Prototypes, Draft Reports, Code Reviews, etc)

1. Planning - By ____9/09/24____
2. Project Design - By ___9/30/24_____
3. Prototype - By ____10/14/24_____
4. Development - By ____11/04/24_____
5. Final Report - By ____12/02/24_____

Meeting Schedule Date/Time

- Tuesdays and Thursdays from 2:00pm-4:30pm

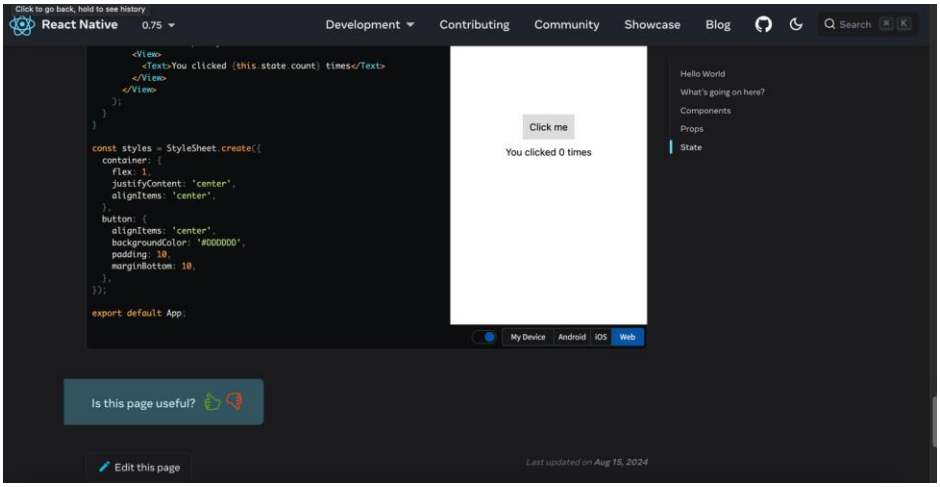
Collaboration and Communication Plan

- We conduct meetings in teams 2-3 times a week and communicate throughout the week on GroupMe. We also discussed meeting in person biweekly after classes.

Project Schedule and Task Planning

- Planning
    - Project Plan
    - Software Requirement Spec
    - Software Design Doc
    - Dev Document
- Project Design
    - React Native tutorial
    - Define tech required
    - Database design
    - UI Design
- Development
    - Develop working prototype
    - Test prototype
    - Review requirements
    - Document updated design
    - Test product
- Final Report
    - Presentation preparation
    - Draft report
    - Video preparation
    - Final report submission to D2L

React Native Training



App Mockups


Basket
Username
Password
LOGIN
Don't have an account yet?
Sign up here
← Lists
Groceries
Milk
Eggs
Bacon