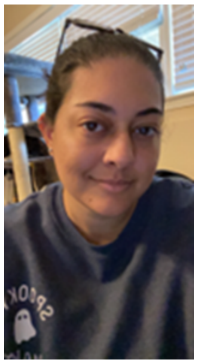
SP-5 Silver Grocery List App

Final Report

CS 4850
Fall 2023
Professor Perry
12/03/2023

Akinola Akintobi	Ana Upreti	Brian Bakor	Diana Calixto
			

Project Website:
https://silvergrocery.github.io/ksuprogrammer1684.github.io-grocerylistwebsite-/

Project Github:
https://github.com/SilverGrocery

Table of Contents

1.0. Introduction

Grocery shopping is an essential activity to all, but oftentimes, things are forgotten or missed during this trip to the store. This is especially true when grocery shopping is done for a group of people, like family or roommates, where it often occurs that items were not bought simply due to miscommunication. This grocery mobile application aims to solve this problem by providing people a way to create and manage grocery lists as groups and have them update across the devices of the members in the group. This document contains the requirements and design of the project as well as the testing performed on the application.

1.1. Version Control

Version control is a way for developers working in teams or groups to keep track of any changes made on the project by the various group members. This project used Github for version control. All project source code was uploaded into a repository and changes were tracked via Github. The project's website source code was also tracked via a Github repository.

2.0. Requirements

2.1. Log in and Register

2.1.0. Log In

2.1.0.0. The application will allow the user to enter their username to log in.

2.1.0.1. The application will allow the user to enter their password to log in.

2.1.1. Register

2.1.1.0. The application will require the user to enter a group name.

2.1.1.0.0. If the user already has a group they intend to join, the application will allow the user to bypass the group name entry.

2.1.1.1. The application will require the user to enter a unique username.

2.1.1.2. The application will require the user to enter a password.

2.1.1.3. The application will require the user to enter an email address.

2.1.1.3. The application will require the user's first and last name during registration

2.1.3. The application will allow the user to recover their log-in information using their email address

2.2. Authentication

2.2.0. The application will authenticate the user's entered username and password when logging in.

2.2.1. The application will verify if the group name is already in-use during registration.

2.2.2. The application will verify that the user's chosen username is unique.

2.2.3. The application will use the user's email to verify their account during registration

2.3. Grocery List and Group

2.3.0. The application will update to the most recent version when another user edits it.

2.3.1. The application will allow the user to add items to the list.

2.3.2. The application will allow the user to delete items from the list.

2.3.3. The application will allow the user to cross off items from the list.

2.3.4. The application will allow the user to create multiple lists
2.3.5. The application will allow the user to search for their group if they did not create one during registration
2.3.6. The application will allow the user to join their group after registration if they did not create one
2.3.7. If the user is new, the application must provide a tutorial explaining the details and features of the app.
2.3.8. The application will allow the user to join different groups.

2.4. Tutorial

2.4.1. The application will provide a tutorial for new users

2.5. Other Requirements

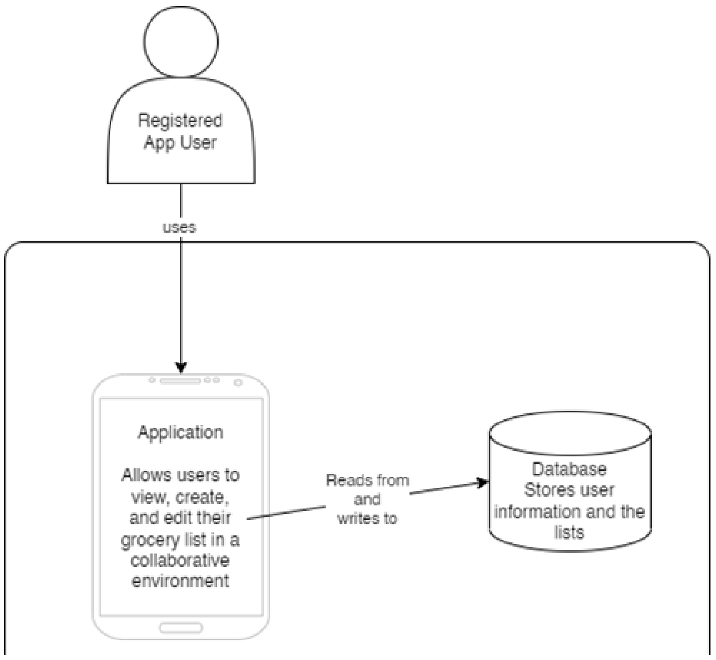
2.4.0. Operating System
    2.4.0.1. The application will be available on both Android and iOS, with the minimum versions for each operating system being 4.1 and 10.0 respectively
2.4.1. Usability
    2.4.0.1. When using the application, users must be able to complete all given tasks with a 98% rate.
2.4.2. Security
    2.4.0.1. The application will require the user’s password to be 8-14 characters long
    2.4.0.2. The application will require that the user’s password contains at least one of each: an uppercase letter, a lowercase letter, a number, a symbol.

3.0. Design

This section contains various diagrams and interfaces of the mobile application which allows users to create and edit a grocery list collaboratively.

3.1. Container Diagram of Application

Figure 1: High Level Application Diagram

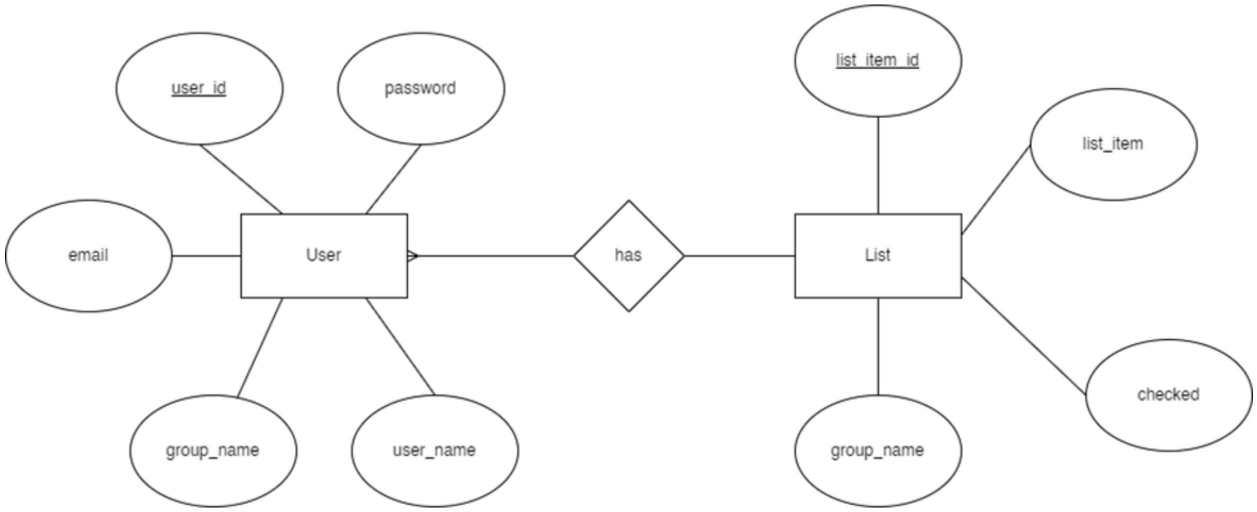

Registered App User
uses
Application
Allows users to view, create, and edit their grocery list in a collaborative environment
Reads from and writes to
Database Stores user information and the lists

The diagram above is a container diagram of the application that provides a high-level view of the application. It depicts the user of this application, one container (the mobile application), and the database. The user shown in the diagram has to be registered, the application provides all the functionality the user needs, and the application then reads and writes from/to the database whenever a new item is added unto the list or whenever a new user creates an account within the app.

3.2. Entity-Relation Diagram of Application's Database

The diagram below is an entity-relationship, ER, diagram of the application's database. It consists of only two entities, user and list. As seen in the app, many users can have one list (due to the group functionality of the app), and each user has five different attributes: user_id, password, email, group_name, user_name. The list entity on the other hand has only four attributes, sharing one of them (group_name) with the user entity. The four attributes of the list entity are list_item_id, list_item, checked, and group)name.

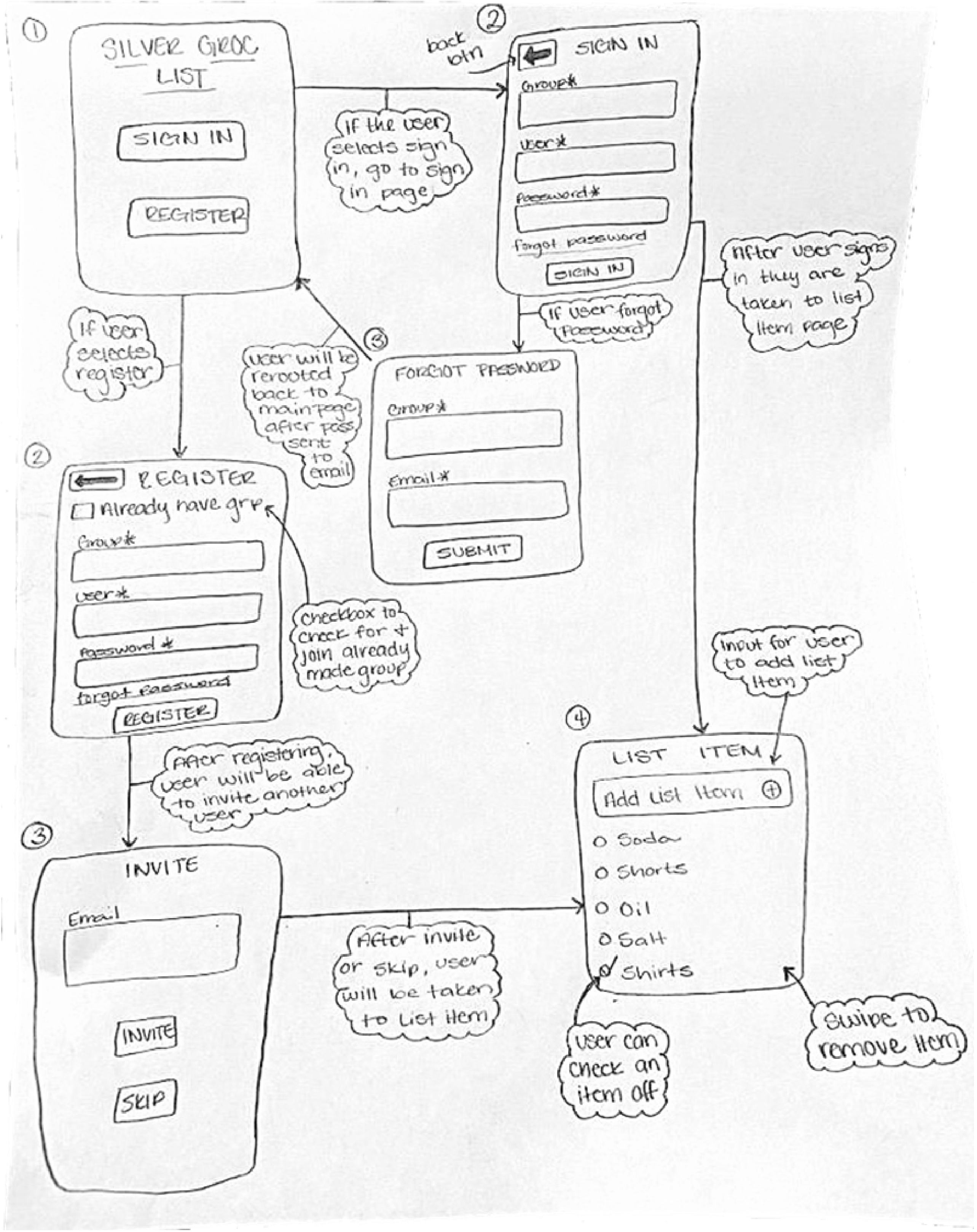
Figure 2: Application ER Diagram



3.3. Low-Level Application Flow Design

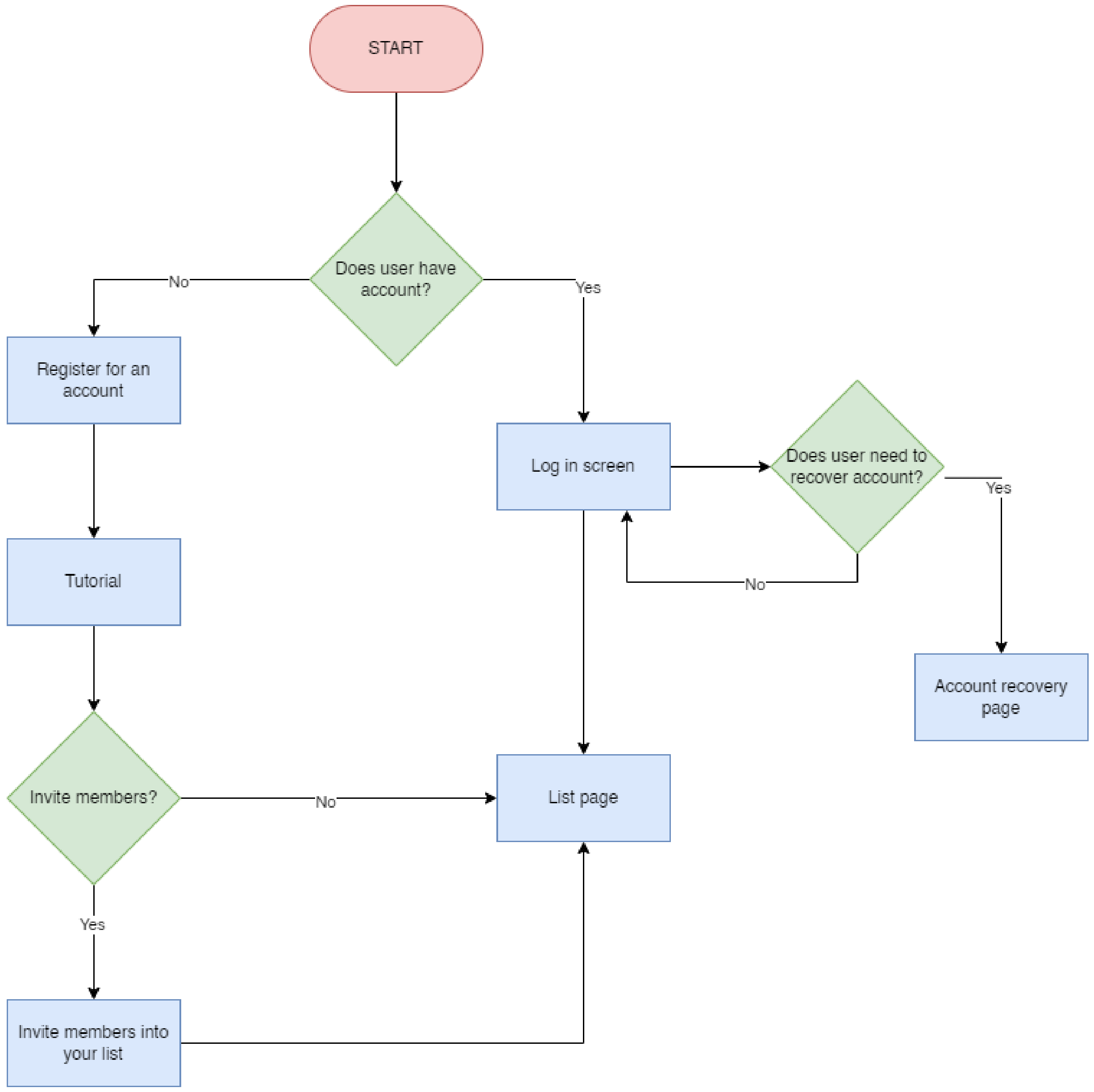
Figure 3 depicts the application design flow. The application starts off by showing a landing page where a user can sign in/register. If the user chooses to sign in, they are led to the sign in page, where their credentials are asked for. If the user taps on "register," then they are led to a different screen to register for an account. After registering for an account, they are asked whether they have members they want to add to the list. Regardless if the user chooses to sign in/register, they are lead to the list screen (screen #4), where they can view and edit their list.

Figure 3: Application Design Flow


1
SILVER GROC LIST
SIGN IN
REGISTER
If the user selects sign in, go to sign in page
2
back btn
SIGN IN
Group*
User*
Password*
forgot password
SIGN IN
After user signs in they are taken to list item page
If user forgot password
If user selects register
3
User will be rerouted back to main page after pass sent to email
FORGOT PASSWORD
Group*
Email*
SUBMIT
2
REGISTER
Already have grp
Group*
User*
Password*
forgot password
REGISTER
Checkbox to check for + join already made group
Input for user to add list item
4
LIST ITEM
Add list item
Soda
Shorts
Oil
Salt
Shirts
After registering, user will be able to invite another user
3
INVITE
Email
INVITE
SKIP
After invite or skip, user will be taken to List item
User can check an item off
Swipe to remove item

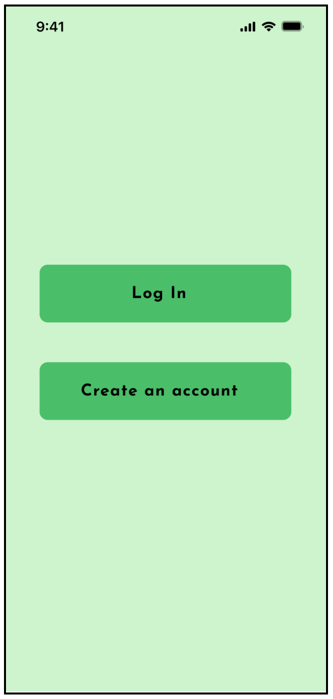
3.4. Detailed Application Flow Design

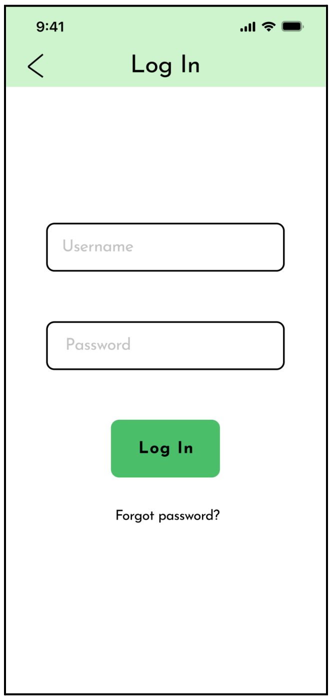
Figure 4. Updated Application Flow for Phase 2

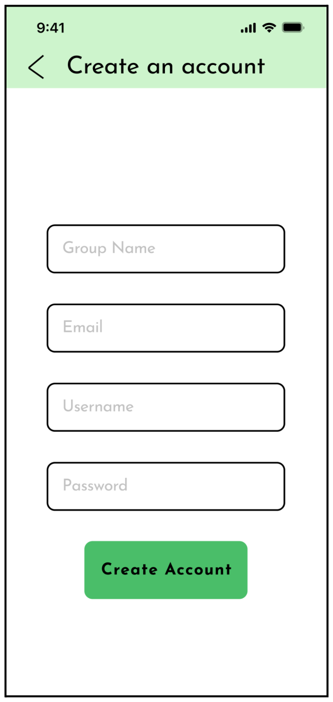


The figure above depicts the application's flow, updated for phase 2 (not currently implemented). This version of the app's flow includes a tutorial, included after receiving feedback on the previous design. The tutorial will aid new users in navigating the app. Everything else in the flowchart remains the same. Users will first either choose to register or log in. If they log in, they'll be taken to the list page. If they need to recover their account, then they can navigate into the "Forgot Password" screen. If a user instead chooses to register, the application will ask for their information, and after they have created their account, it will lead them straight into the tutorial. After the tutorial, they can choose to invite members or go straight into their list page.

3.5. Screen Mockup

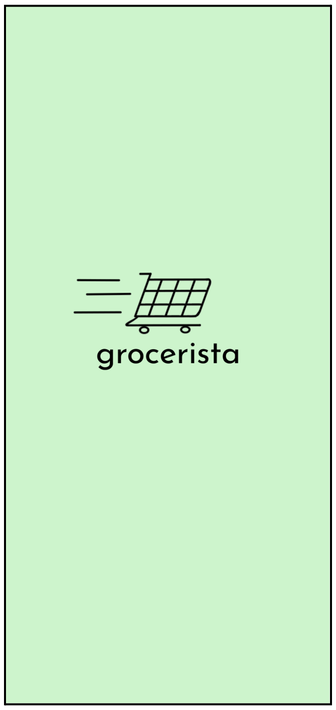


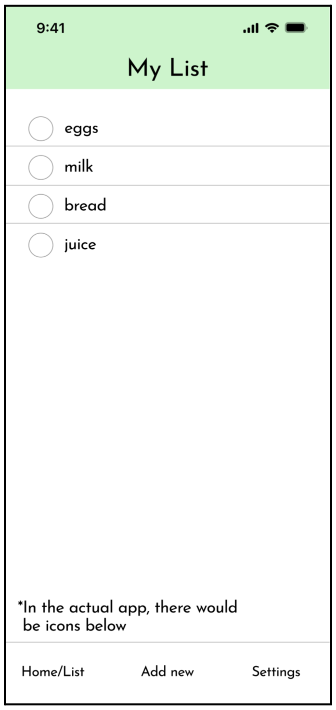




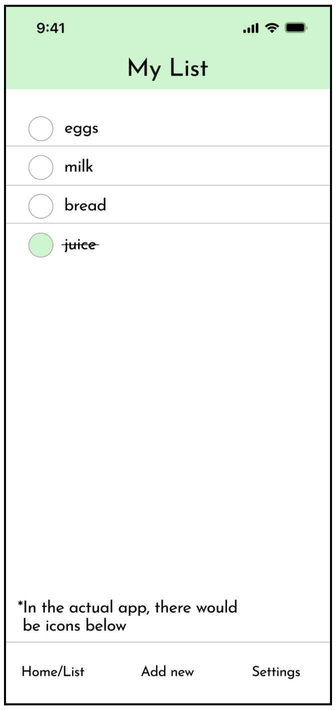
The pictures above depict a potential design for the application. This design uses green as the main color and features large textboxes and buttons for ease of use. The first screen depicted above is what the user will see immediately after opening the app. The second screen is what the user will view upon tapping “log in,” and the third screen is what the user will view if they tap “create an account.” Users can enter their username and password to log in, but if they have forgotten either or, they can also choose “Forgot password?” to lead to a recovery screen. In the “Create Account” screen, users will be asked for a “Group Name,” an “Email”, a “Username”, and a “Password”.

The images shown below are additional mockups of the application. It includes a splash screen, which would be shown to users when they open the app, and two different versions of the list page. The first version shows the list as it would be seen by the users, and the second shows the list again if the users were to cross off an item. The circle by the item would turn green and the text would be striked through.





*In the actual app, there would be icons below



*In the actual app, there would be icons below

4.0. Testing

4.1. Test Cases

Test ID#	Test Purpose	Expected Output	Actual Output	Pass/ Fail
001	Add Item To List	Users should be able to enter an item into the input and when they click the ”+” button, the item will be added to their list.	Users are able to add items to their list.	Pass
002	Remove Item From List	Users should be able to click the “x” button beside the item and it will be removed from their list.	Users are able to remove items to their list.	Pass

003	User Can Sign In	Users should be able to enter their information within the required sign in input fields. When the user activates the “Sign In” control, the user is navigated to the list page.	Users are able to add information into the sign in input fields. When the user activates the “Sign In” button, they are taken to the list page.	Pass
004	User Can Register	Users should be able to enter their information within registration input fields.	Users are able to add information into the register input fields.	Pass
005	Retrieve Forgotten Password	When users activate the “Forgot Password” button, they should be navigated to the Forgot Password page. Users should be able to enter their information within the forgotten password input fields and when the submit button is activated an email will be sent to the user.	User is navigated to the Forgot Password page and can enter their email into the input field, but there is no functionality to email the user their correct password.	Fail
006	Invite Another User	After a user registers, they would be taken to the invite page. Users should be able to enter an email and, on submission, an invite link would be sent to that email.	Users are navigated to the invite page, but there is no functionality to email another person to invite them to a group grocery list.	Fail
007	Authenticate User - SIgn In	When a user submits their sign in information, the user will be authenticated and taken to their grocery list.	There is no functionality to authenticate a user's sign in information.	Fail
008	Authenticate User - Register	When a user submits	There is currently no	Fail

		their registration information, the user data will be saved, authenticated, and taken to the invite page.	functionality to save or authenticate a user's registration, but users will be navigated to the invite page.	
009	Save User's List	User's list should be saved to a database so that their list is always up to date.	There is currently no database connected to the application so the list cannot be saved and updated but a user can input the list that can be viewed in real time.	Fail

4.2. Test Report

The purpose of this test report is to evaluate the performance, functionality, and usability of the Silver Grocery List App. The app aims to provide users with a convenient and efficient way to create and manage grocery lists. The testing was conducted to identify any potential issues, assess the app's adherence to requirements, and ensure a simple and high-quality user experience.

Tests focus on frontend user interface and backend database functionality. After testing it is clear that the frontend user interface such as input fields, buttons, and navigation pass and users are fully capable of activating all functionality. Unfortunately, the tests that required the use of saving and authenticating to a database failed. This means that users are not able to save data or create a usable login. Overall, the application frontend UI was implemented well but the database functionality needs additional work.

5.0. Development Discussion

5.1 Background

Grocery shopping can be a challenge! Sometimes you think of something in the moment and don't have a list in front of you; then it slips your mind while at the store and you're standing at the checkout line wondering what it was you have forgotten. That is why our team decided to create a mobile grocery list to help minimize this issue. We even realized that groups of people living together need to have one list so that the same item isn't bought multiple times or that an item is not forgotten. Because of this, we decided we wanted to add the capability to create a shared group list where multiple users can sign into the same list and edit it. Once we made the decision on what we wanted the application to be, we needed to determine the roles and responsibilities of each team member.

One of our first group discussions was about the strengths and weaknesses of each team member. During this conversation, we each discussed our interests, experiences, and what career goals we would have after graduation. Akinola mentioned his interest in a leadership position that uses his organization skills and knowledge of the project lifecycle and voiced his disinterest in writing code. This made him a good candidate to be our team lead, where he organized our weekly team meetings and kept up to date with the deadlines, so the team didn't fall behind. Originally, Brian was not interested in being a programmer for this project but expressed his experience in writing an Android application and offered to help with coding tasks. His contributions were creating the project website and testing the application on the Android emulator. Diana's adept organizational skills made her the best fit for keeping track of and writing project documents. Her experience and interest in project management gave her the ability to create a detailed requirements list and draw out project designs and diagrams. Ana came into this project wanting to be a programmer because of her industry experience. Her previous experience with React was a valuable skill in researching and building this application. She was able to build a fully functional front-end application using React Native and Expo. With the application and roles decided the team was ready to begin the project life cycle.

5.2 Design

Our planning process consisted of determining what we want our application to have, how it would function, and how it would flow. Through the research process we asked ourselves, friends, and family what they would want from a grocery list application. From this, we were able to compile a list of features and requirements for our application. We wanted to have a landing page where the user will be brought when the application opens. Since we wanted users to have personalized lists, we needed to allow them to log in or register an account. From the landing page, users should be able to navigate to either the registration page or the sign in page. If the user is new, and has registered, we want them to have the option to invite a user to their group so they can collaborate on their grocery list. Originally, we considered allowing users to be able to create multiple lists and having them navigate to a list overview page once a user logs in. After some consideration and group discussion, we decided to remove this feature to simplify the application. Instead, we decided to have users navigate to their group grocery list page after logging in.

Once we had an idea of how we wanted the application to flow and some requirements, we wanted to draw a diagram for reference. For the landing page, we designed two buttons that would allow the user to navigate to the sign in page or the registration page. For the registration page, we decided that users should have a switch button to indicate if they are already in an existing group or not. Our design also included four input fields that the user would be required to fill out. The group name field will be used to either search for the already existing group they want to join or what the name of the group they are creating will be. The username field is specific to the user only and will be required to sign in. The email address input field will be required for users in case they forget their password. The last input field will be for the user to create a password specific to this user and will be required to sign in. From the registration page, our design showed that the user would navigate to the invite page to invite anyone who should have access to their list. This page would only consist of an input field for the invitee's email address, a submit button to invite, and a sign in button to take the user to the sign in page. On the sign in page, there would be three input fields required to be filled out for the user to be logged in. The user would have to provide the group name, username, and their password. If a user forgot their password, there would be a button to navigate to the forgot password page. This page would consist of two input fields. For authentication purposes we would request the user's username and email address, and the idea would be for the password to be emailed to the user. When a user is logged in, they would be taken to their list page. On this page, a user would find an input field where they can type in an item they want to add to their list. We designed a button that when activated adds the item to the screen and clears out the input field. When an item is added, the user will need to be able to remove the item once it has been bought or if it is no longer needed. We originally wanted to build this feature so that the user could swipe the item and delete it. With the requirements, function, and flow drawn out, the developers could start their work on the application.

5.3 Brainstorming

With a plan in place on how the application should look and flow, the team needed to decide on how we would build the application. Ana suggested React Native due to her previous experience with React and knowing that one code base could build applications for the web, Android, and iOS. This is exactly what our team wanted. We would be able to write our application one way and the React Native would manage converting the application components into code that is required by iOS or Android. With further research, the team found that React Native would be a good platform to build the application, especially because it was recommended for first time application programmers.

During the React Native research, we noticed Expo was recommended for building iOS and Android mobile applications in conjunction with React Native. Expo is a tool that assists in the full build of setting up and creating an application. It has its own Command Line Interface and allows you to create the project directly in your terminal with a simple command. Expo has an application called Expo Go that allows developers to test their application in real time on a mobile device. Developers need to download the application onto their phone and ensure that their mobile device is connected to the same internet connection that their computer is connected to. In the terminal, a developer can enter "npx expo start" and a local server will start their application. A QR code will display in the terminal and the developer can scan this with their mobile device and it will automatically load their application. Any changes they make to their project will show right away on their mobile device. This was incredibly helpful for the developers to test their project in real time and on a mobile device making the user experience more easily accessible to the developer. With these findings, the team agreed to build the application with React Native and Expo.

For authentication and saving a user's list for later reference, we needed to have a database connected. In our original research phase, we planned to use MySQL. Some of the team had some experience with MySQL and it seemed like it would work well with React Native and Expo. When the time came to start connecting the database to the application, we realized that

MySQL was mainly used for local storage and connection. To have a fully usable application, we would need to find a more reliable database to use for authentication and storage. It was suggested to our team by Professor Perry to investigate using Firebase for our database.

We began research on Firebase and found that it had a lot of useful tools and that it was compatible with React Native applications. Because of this, we decided that we would move forward with integrating Firebase into our application. We decided we wanted to use the authentication tool for logging in users and the Realtime database for storing the list data. We began combing through the Firebase and Expo documentation regarding adding the database to our application. Following the documentation to get our application to connect to the Firebase console proved to be much more difficult than originally expected.

The first issue we encountered was not being able to find information within our file structure that the documentation claimed we should have already had. Eventually, we found that in order to have those files, we needed to do an expo prebuild which built out a file structure that included the data that was mentioned in the Firebase documentation. Though we eventually figured this out, this did not solve our connection issue to Firebase. More research was needed to understand the error message we were receiving which led us to learn that Firebase React Native, which was what we were originally using, was not compatible with Expo. Because of this, we found the Firebase JS SDK which claimed to be compatible with Expo.

We began working with this SDK and thought that a successful connection had been made between the Firebase console and our application. However, when trying to add the authentication tool code into our codebase, we received a new error message that we had not seen before. The error message we were getting was: "FirebaseError: Firebase: No Firebase App '[DEFAULT]' has been created - call initializeApp() first (app/no-app)., js engine: hermes". Even with this error, we still thought we were on the right path, but we did not know the cause or the fix for this error. After much research, many attempts to fix, request for help from others, and talking with Professor Perry, we were still not able to resolve this error. Overall, if we had been able to connect Firebase successfully, it would have been a great database tool for our application.

5.4 Development

We started writing the application in mid-September. The first task was to set up file structures for the project and start the UI for the application pages. We added the file structure for the project by creating a components folder with JavaScript files for each one of the application pages. Then we added simple JSX code to create the visuals for the Login Page, Registration Page, List Page, Invite Page, Landing Page, and Forgot Password Page. After the visuals for the application were complete, UI elements and functionality were added to all the pages. The last task in September was to create the navigation logic for the application. We decided on stack navigation functionality for the entire application and completed writing out the code to successfully navigate pages by the end of September.

October was the core of our development phase. To begin the month, we worked on the authentication logic for the project. The authentication will be responsible for determining which pages show when a user is not logged in, and which pages show when a user is logged in. The code for our authentication uses a ternary operator to only have access to the landing page, login page, forgot password page, registration page, and invite page when the user is not logged in and only have access to the list page when the user is logged in. We did this by making the navigation logic work with the authentication logic. We also finished the functionality for the user to add items to their grocery list. Once all the frontend functionalities had been coded, we moved to working on the backend. We researched how to connect our React Native Expo application to a MySQL database. After doing more research on MySQL database, we realized that this was not the best fit for our application since it focuses more on local storage. We then discussed this with Professor Perry who suggested that we investigate Firebase. Working through the Firebase documentation, we realized we were missing some file structures for iOS and Android so we created a development build for an Android emulator and iOS simulator in the hopes that these would produce the files we needed. It took awhile to successfully create the development builds but when we did, they ran successfully on the emulator and simulator. However, it did not produce the file structure we were looking for and we did not have a path to connect the application to Firebase. The next thing we tried was to create Android and iOS applications in the Firebase console and downloaded the connection files for those applications and inserted them into the root of the project. This route also did not work for us.

November mostly consisted of trying to fix the Firebase connection errors as well as finishing up miscellaneous frontend functionality. Continuing from our trials in October, we found a document that recommended we create a web firebase project and connect it to the Android and iOS builds. Because of this, we deleted the iOS and Android applications in the Firebase console and created a web application instead. We went through the tutorial but had the same outcome at the end. We were able to get the web Firebase to connect to the application, but we got errors. When running the app, it failed and threw an error: "FirebaseError: Firebase: No Firebase App '[DEFAULT]' has been created - call initializeApp() first (app/no-app)., js engine: hermes". After getting this error, we did a lot of research where we learned about Expo prebuild and ran it in the project which built iOS and Android file structures that were missing when we tried previous paths. At this point we recreated the Android and iOS Firebase projects and connected them to the application. This still produced the same error mentioned above. Some of the group members went to the Firebase help lecture and tried the student help document that referenced the same document we were previously following but the error persisted. To take a break from the Firebase errors, we went back to the frontend code to fix up a few items we had missed previously. We added the images into the asset folder in the project to keep the images from loading slowly. We added the functionality for users to remove a list item from their grocery list. We added keys to the list items so that the React Native logic would accurately load the list. The semester is coming to an end and although we were not able to get the database connected, we learned a lot and made a full functioning frontend application in the process.

5.5 Current State of Application

Although our project does not have a connected database, the application when run on an emulator or simulator has a fully functional frontend user interface. Currently when the application is opened by a user, they are brought to the landing page which has two buttons to navigate the user to the sign in page or the registration page. When the user goes through the registration flow, they are taken to a form that allows them to select if they are joining an existing group or creating a new group, create a personalized username, add their email address, and create a unique password. When the user has filled out this form, they are taken to an invite page where they can invite other users to the group grocery list. When the user goes through the sign in flow, they are taken to a form where they can input their username, email address, and password to log in. If they have forgotten their password, they can activate the forgot password control to navigate to the forgot password form. They will be required to provide their group name and email address to receive an email with their password (the email functionality does not currently work without the authentication tool from Firebase). When the user does login, they are taken to their group grocery list where they can see the current list, add new items to the list, and delete items from the list. Overall, the application does most of the actions we wanted it to do. We would just need a longer project cycle to make the application production ready.

5.6 How to Run the Application

Because we do not have a production ready application, we thought the best way to provide the application for review would be to use the Expo Go application, which is how we have run the testing throughout the project. You will first need to download the Expo Go application to your mobile device. Ensure that your device is connected to the same internet connection as your computer. To run our application, open our project files in Visual Studio Code and open a terminal window. In the terminal window, type the command “npx expo start”. This will start a local server for the application and a QR code will show in the terminal window. Scan this code with your mobile device’s camera and it will automatically load our application into Expo Go. Enjoy running our application!

5.7. Future Development

If we were to continue the project cycle for this application, our focus would be on figuring out how to successfully connect Firebase to the application without any errors. If we were to figure out the connection issue, we could implement the Firebase authentication feature into our application to log in users, register users, use email communication for forgotten passwords, and even add third party authentication such as Google, Apple, and more. We can also utilize the Realtime database to house group grocery lists and only allow users to access them when they are authenticated. We also want to add the capability for email communication from the application when a user is invited to a group’s grocery list. We have learned so much throughout this semester about the project life cycle and building fully functional mobile applications. Using this knowledge in the future will help prepare us for interviews and entering the workforce.

6.0. Challenges and Assumptions

The biggest challenge we had was connecting our application to Firebase. We wanted to use Firebase authentication for users to be able to sign in, register, and to retrieve their forgotten password. We wanted to use the Firebase Realtime database to store a user’s grocery list so that it was always up to date. The problem was that whenever we tried to connect the application to the Firebase console and run the application, an error was thrown “FirebaseError: Firebase: No Firebase App '[DEFAULT]' has been created - call initializeApp() first (app/no-app)., js engine: hermes”. We spent over a month researching and testing ways to fix the error but were unsuccessful. If we were to continue to work on this project in the future, we could do a deeper dive into both Firebase and Expo to make them work together.

7.0. Conclusion

The purpose of this project was to learn how to create an application within a standard project lifecycle. We began the cycle by planning what our application would be.Defining an issue to solve with our application was our first step, which is why we decided to create a grocery list application. We continued the planning cycle by creating designs and a requirements list before writing any code. The next step was to research the best technology to use for our project which is how we decided to pursue a React Native and Expo project. React Native is a great starter library for new application developers because you only have to write one code base for multiple platforms. For that same reason, we used the Expo platform to help us create our application because it has a lot of built in tools to help first time developers succeed. Finally we had to choose a database to house our user information and grocery list data. We decided on Firebase because it had built in tools for authentication and a Realtime database. Building the frontend user interface of the application went very well and we ended up with a fully functional front end application. Unfortunately, the backend proved to be more challenging and full of errors that we were unable to solve. Overall, this project allowed the team to learn all about the project life cycle even if we were unable to create a production build. If we were to continue to work on this in the future, we would be able to figure out our Firebase errors and create a fully functioning application.

Appendix

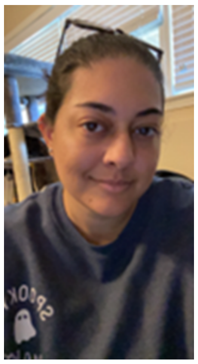
Project Plan

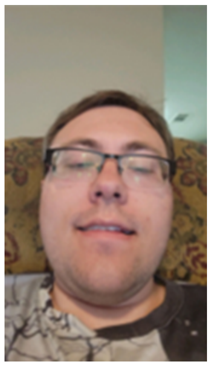
Grocery List App SP - 5 Silver
Date: September 3, 2023

Project Team

Roles	Name	Major Responsibilities	Contact (Cell Phone)
Project owner	Sharon Perry	Facilitate project progress; advise on project planning and management	Sperry46 in D2L is preferred
Team leader	Akinola Akintobi	Team leader	[redacted]
Team Members	Anastasia Upreti	Developer	[redacted]
	Diana Calixto	Documentation	[redacted]
	Brian Baker	Developer	[redacted]
Advisor/ Instructor	Sharon Perry	Facilitate project progress; advise on project planning and management	Sperry46 in D2L is preferred

Akinola


Ana


Brian

Diana

Project Overview
Our group will be completing a Mobile Grocery List App, that will allow users an easy solution to keeping an updated and ready to use list when they are at the store.

Our app will allow families or roommates to add, check, and remove items from a list straight from their phone. This will allow users to have an updated list of things needed when they are able to run to the store. Having this mobile app makes it convenient because you will have it wherever you go!

Project website

URL shown below:
https://aupreti1.github.io/SilverGroceryWebPage/**

** This website link is not the correct version. See cover page for updated website link.

Final Deliverables

1. GitHub Link
2. Source Code
3. Website
4. Final Application

Milestone Events

Define a few milestone events for your project – events that make sense for your project. In general, milestones like planning (RAD), Development, Prototype, Research Results, and Final Report.
#1 Planning - By **September 22nd**
#2 Prototype - By **October 28th**
#3 Final Application - By **November 4th**

Meeting Schedule Date/Time

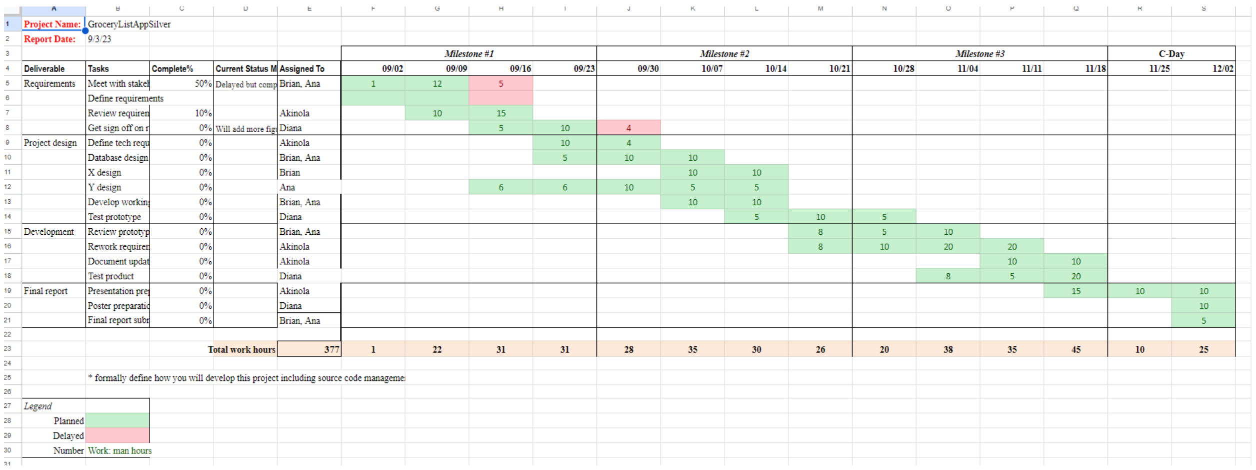
We will meet every Monday at 8pm on Zoom

Collaboration and Communication Plan

We will be using Zoom for meetings and GroupMe/Text for communications.

Project Schedule and Task Planning

See preview of Gantt chart below:



Project Name: GroceryListAppSilver
Report Date: 9/3/23

					Milestone #1				*Milestone #2*				*Milestone #3*				C-Day	
Deliverable	**Tasks**	**Complete%**	**Current Status M**	**Assigned To**	**09/02**	**09/09**	**09/16**	**09/23**	**09/30**	**10/07**	**10/14**	**10/21**	**10/28**	**11/04**	**11/11**	**11/18**	**11/25**	**12/02**
Requirements	Meet with stakel	50%	Delayed but comp	Brian, Ana	1	12	5											
	Define requirements				🟩	🟩	🟥											
	Review requiren	10%		Akinola		10	15											
	Get sign off on r	0%	Will add more fig	Diana			5	10	4									
Project design	Define tech requ	0%		Akinola				10	4									
	Database design	0%		Brian, Ana				5	10	10								
	X design	0%		Brian						10	10							
	Y design	0%		Ana			6	6	10	5	5							
	Develop workin	0%		Brian, Ana						10	10							
	Test prototype	0%		Diana							5	10	5					
Development	Review prototyp	0%		Brian, Ana								8	5	10				
	Rework requiren	0%		Akinola								8	10	20	20			
	Document updat	0%		Akinola											10	10		
	Test product	0%		Diana										8	5	20		
Final report	Presentation pre	0%		Akinola												15	10	10
	Poster preparatic	0%		Diana														10
	Final report subr	0%		Brian, Ana														5
			Total work hours	**377**	**1**	**22**	**31**	**31**	**28**	**35**	**30**	**26**	**20**	**38**	**35**	**45**	**10**	**25**

* formally define how you will develop this project including source code manageme

Legend

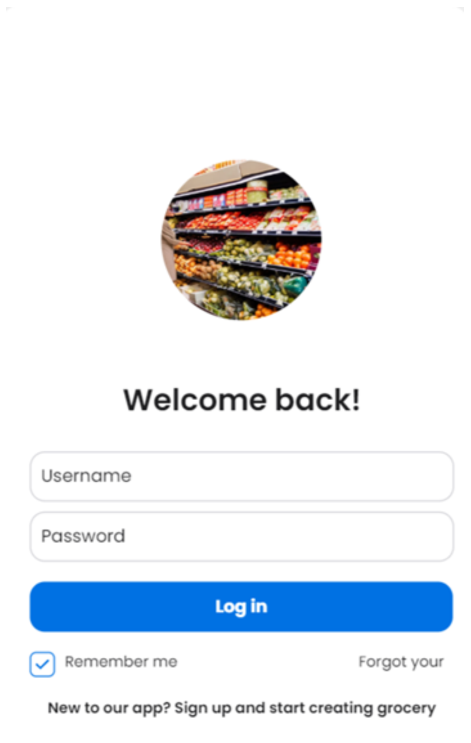
Planned	🟩
Delayed	🟥
Number	Work: man hours

Version Control Plan

Our group will be using GitHub for version control. We will have main that will only be updated when the project is at a clean and working version. From “main” we will have a branch called Dev where we will push our collaborative work. From Dev we will have branches for each student who is developing the application.

Proof of Completion of Tutorial

Akinola Akintobi



Welcome back!

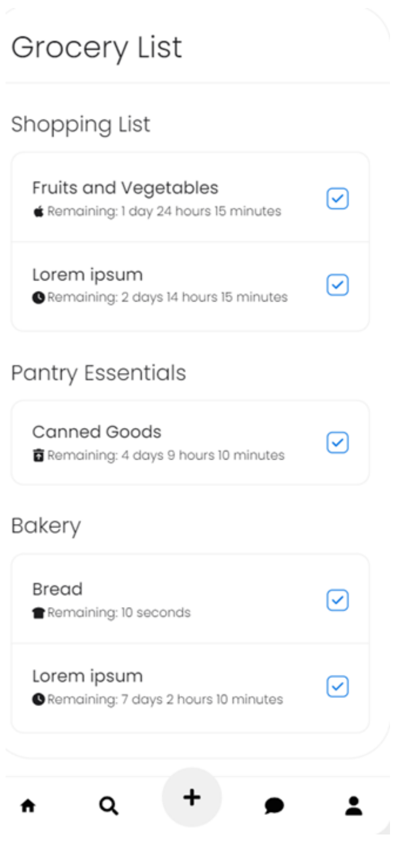
Username

Password

Log in

Remember me   Forgot your

New to our app? Sign up and start creating grocery



Grocery List

Shopping List

Fruits and Vegetables
Remaining: 1 day 24 hours 15 minutes

Lorem ipsum
Remaining: 2 days 14 hours 15 minutes

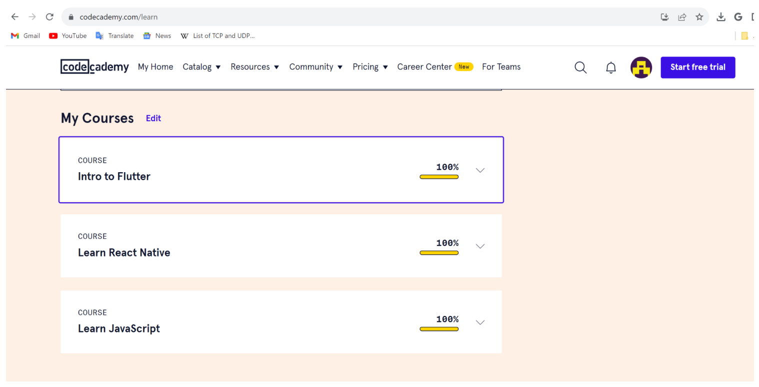
Pantry Essentials

Canned Goods
Remaining: 4 days 9 hours 10 minutes

Bakery

Bread
Remaining: 10 seconds

Lorem ipsum
Remaining: 7 days 2 hours 10 minutes



codecademy.com/learn

Gmail YouTube Translate News List of TCP and UDP...

codecademy My Home Catalog Resources Community Pricing Career Center New For Teams Start free trial

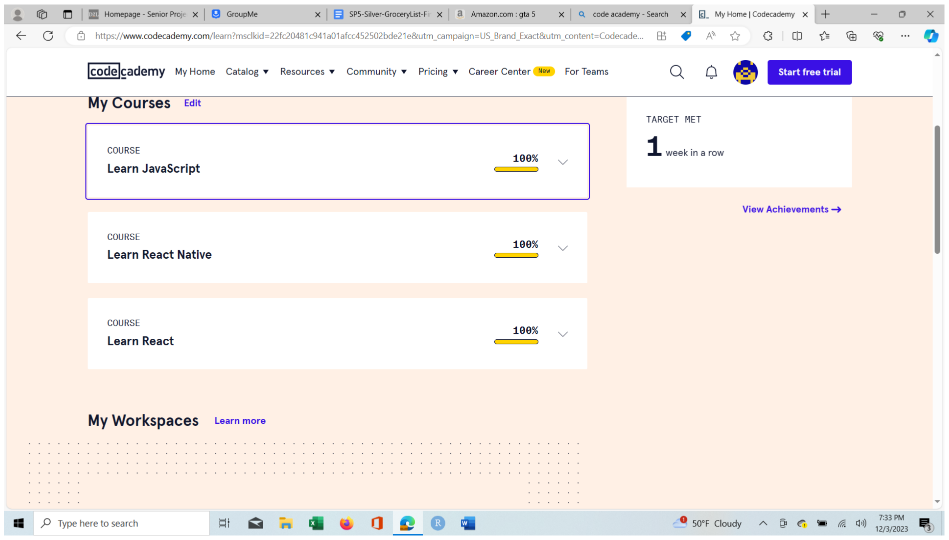
My Courses Edit

COURSE
Intro to Flutter 100%

COURSE
Learn React Native 100%

COURSE
Learn JavaScript 100%

Brian Baker



https://www.codecademy.com/learn?msclkid=22fc20481c941a01afcc452502bde21e&utm_campaign=US_Brand_Exact&utm_content=Codecade...

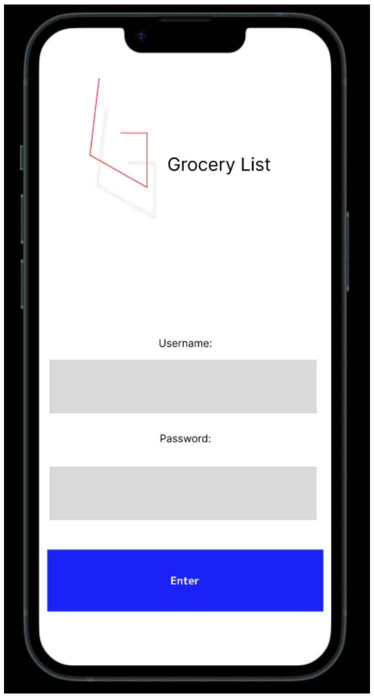
My Courses Edit

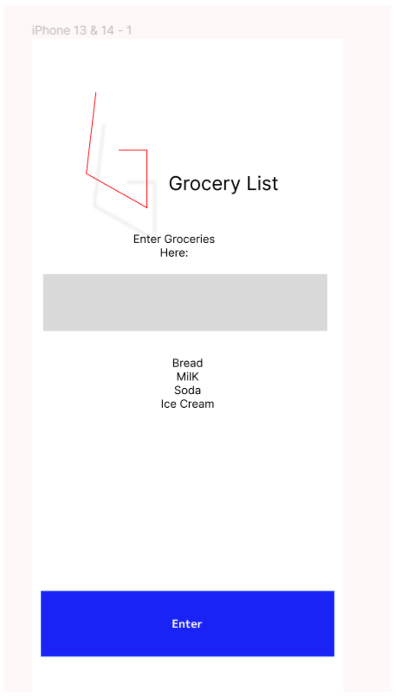
COURSE	Learn JavaScript	100%
COURSE	Learn React Native	100%
COURSE	Learn React	100%

TARGET MET
1 week in a row

View Achievements →

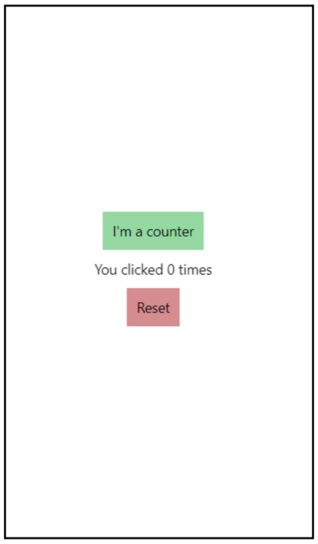
My Workspaces Learn more



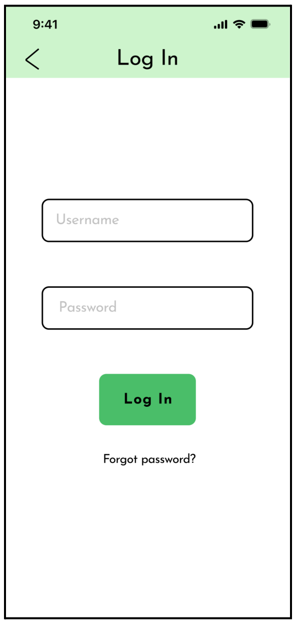


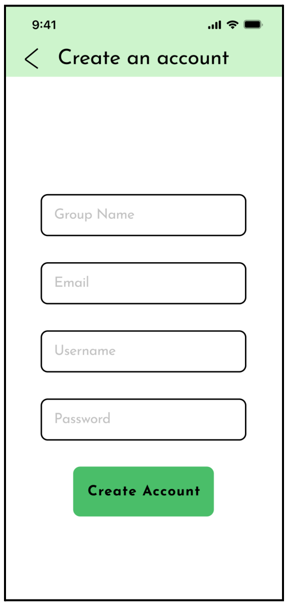
Diana Calixto
Tutorial results, following tutorial from official React Native site:
https://reactnative.dev/docs/tutorial?language=javascript

<u>React Native Tutorial Results:</u>

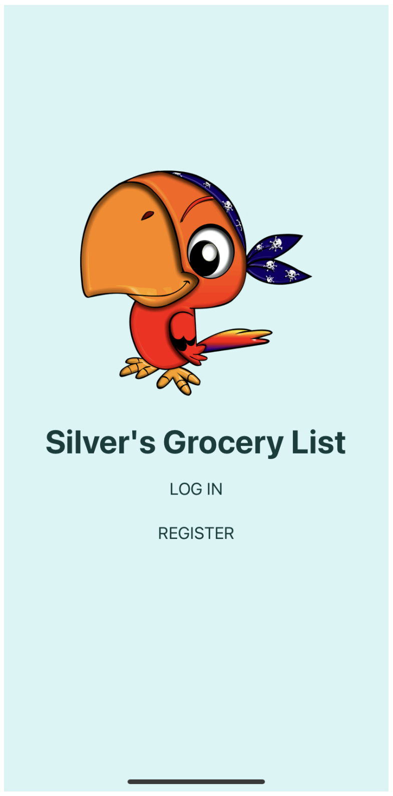


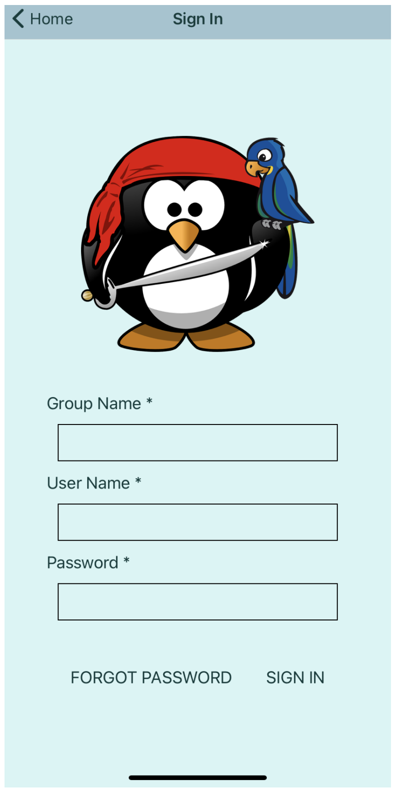
<u>Figma Mockups:</u>

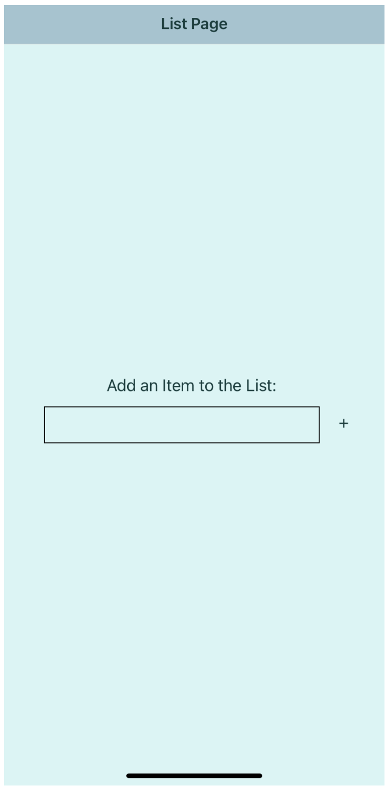




Anastasia Upreti


Silver's Grocery List
LOG IN
REGISTER


Home
Sign In
Group Name *
User Name *
Password *
FORGOT PASSWORD
SIGN IN


List Page
Add an Item to the List:
+