Ryan Mackintosh

21171466

Assignment One: App Design and Dev Basics

COMP 2010: App Development for the Apple Ecosystem

Curtin University

5 May 2024

Neppatsrev Coffee iOS App Development

Proof-of-Concept Application

Contents

[1.0 Introduction 3](#_Toc165843265)

[1.1 Background 3](#_Toc165843266)

[1.2 Objectives 3](#_Toc165843267)

[2.0 View Identification 4](#_Toc165843268)

[2.1 Required Views 4](#_Toc165843269)

[2.1.1 ContentView 4](#_Toc165843270)

[2.1.2 MenuView 4](#_Toc165843271)

[2.1.3 VIPView (Loyalty Card View) 4](#_Toc165843272)

[2.1.4 MapView (Store Locator View) 4](#_Toc165843273)

[2.1.5 CartView 4](#_Toc165843274)

[2.1.6 SettingsView 5](#_Toc165843275)

[2.1.7 DrinkView 5](#_Toc165843276)

[2.1.8 SnackView 5](#_Toc165843277)

[2.2 Navigation Methods and Justification 5](#_Toc165843278)

[2.2.1 TabView Navigation 5](#_Toc165843279)

[2.2.2 NavigationView and NavigationLink 5](#_Toc165843280)

[2.2.3 Environmental Object Passing 6](#_Toc165843281)

[3.0 Design of Views 7](#_Toc165843282)

[3.1 MenuView 7](#_Toc165843283)

[3.2 VIPView 8](#_Toc165843284)

[3.3 MapView 9](#_Toc165843285)

[3.4 CartView 10](#_Toc165843286)

[3.5 SettingsView 11](#_Toc165843287)

[3.6 DrinkView 12](#_Toc165843288)

[3.7 SnackView 13](#_Toc165843289)

[4.0 App Construction 14](#_Toc165843290)

[5.0 Reflection and iPad Design Considerations 14](#_Toc165843291)

[5.1 Reflections on Design 14](#_Toc165843292)

[5.2 iPad Design Adaptations 15](#_Toc165843293)

[6.0 References 17](#_Toc165843294)

[Appendices 18](#_Toc165843295)

[1. MenuView: 18](#_Toc165843296)

[2. VIPView: 20](#_Toc165843297)

[3. MapView: 21](#_Toc165843298)

[4. CartView: 22](#_Toc165843299)

[5. SettingsView: 23](#_Toc165843300)

[6. DrinkView 24](#_Toc165843301)

[7. SnackView: 25](#_Toc165843302)

[8. Draft Designs 26](#_Toc165843303)

# 1.0 Introduction

## 1.1 Background

Neppatsrev Coffee (the Client) is a startup café attempting to gain market share in the highly competitive Western Suburbs coffee market in Perth, Western Australia. A problem new café brands often face is attracting new customers and creating a loyal client base. To solve this problem, the Client is launching an iOS app-based ordering system to allow customers to make and customise their order on-the-go, which means the yummy mummies of the GT can spend less time waiting in line, and more time discussing the latest gossip with their girls. The Client will take inspiration and design ideas from the large drink chains Starbucks and Boost Juice.

## 1.2 Objectives

The primary objective of this project is to create a proof-of-concept app that embodies the Client’s vision of a seamless and customisable ordering experience. The app will feature:

* A **user-friendly menu for hot drinks**, with detailed customisation options including choice of milk, number of shots, drink size, and whether the coffee is decaf.
* A **digital loyalty card system** that stores and displays how many coffees the customer has purchased and their loyalty points, enhancing their retention by encouraging repeat visits.
* A **comprehensive store locator** integrated with Apple Maps marking the locations of the Client’s four concept stores across Cottesloe, Claremont and Swanbourne, aiding customers in finding their nearest coffee haven with ease.

This project will demonstrate technical feasibility for a small startup in the coffee market and provide insights into how digital tools can be used to enhance customer engagement and drive business growth.

# 2.0 View Identification

## 2.1 Required Views

### 2.1.1 ContentView

Serves as the main container view that manages navigation between different sections of the app via a tab bar. This view includes tabs for the menu, VIP section, store locator, cart, and settings.

### 2.1.2 MenuView

Displays a list of drinks and snacks available for order in a scrollable view. It's structured to allow users to browse through different categories (Drinks and Snacks) and select items to view more details or add to the cart (*See appendix 1).*

### 2.1.3 VIPView (Loyalty Card View)

Displays the loyalty card details including a barcode, member name, member number, and points earned (coffees purchased). It promotes customer engagement through a loyalty rewards program (*See appendix 2).*

### 2.1.4 MapView (Store Locator View)

Shows a map pinpointing the locations of Neppatsrev Coffee stores (Cottesloe, Swanbourne and two in Claremont). It allows users to see the store locations with pins on a map, enhancing the user experience by integrating geographical navigation (*See appendix 3)*.[[1]](#footnote-1)

### 2.1.5 CartView

Manages the items the user has added to their cart. It provides functionality for adjusting item quantities, removing items, and viewing the total cost. This view also includes playful prompts and access to adding snacks directly from the cart view (inspired by the Boost Juice CartView) (*See appendix 4)*.[[2]](#footnote-2)

### 2.1.6 SettingsView

Offers general settings and account options like updating profile information, FAQs, customer service access, and logout functionality (inspired by the Boost Juice SettingsView) (*See appendix 5)*.[[3]](#footnote-3)

### 2.1.7 DrinkView

Offers detailed customisation options for drinks, such as size, number of espresso shots, type of milk, and whether the drink is decaf. It allows users to add customised drinks to their cart, although prices remain constant for this Proof-of-Concept design (*See appendix 6)*.

### 2.1.8 SnackView

Separate to DrinkView because no customisation is available. It provides details for snacks and allows users to add them to their cart directly from the snack details page (*See appendix 7)[[4]](#footnote-4)*.

## 2.2 Navigation Methods and Justification

### 2.2.1 TabView Navigation[[5]](#footnote-5)

**Implementation:** ContentView serves as the root view of the application, implementing a TabView that allows users to switch between the main sections of the app (Menu, VIP (loyalty card), Store Locator, Cart, and Settings).

**Justification:** Inspired by Starbucks and Boost Juice’s use of TabView, it enables efficient and straightforward navigation between the app's primary functionalities. It provides an intuitive interface for users to access major features quickly, enhancing user experience by keeping essential services accessible with minimal interaction. This approach is particularly effective for mobile applications where ease of access to various functionalities is prioritised.

### 2.2.2 NavigationView and NavigationLink

**Implementation:** NavigationView was my most used navigation strategy. MenuView and CartView used NavigationView combined with NavigationLink. NavigationLink was used to navigate to detailed views from a menu item in the MenuView to its detailed order page in DrinkView or SnackView.

**Justification:** Initially, I had nested NavigationViews but this made the user experience less desirable. NavigationView and NavigationLink provide a hierarchical navigation option that allows users to drill down into details from a list, form, etc. This method is especially beneficial for presenting detailed information and customisation options for items without cluttering the main list view. It also supports intuitive back-navigation by swiping or tapping the back button, which is a familiar pattern for iOS users (Apple Inc., n.d.-a; Apple Inc., n.d.-b).

### 2.2.3 Environmental Object Passing[[6]](#footnote-6)

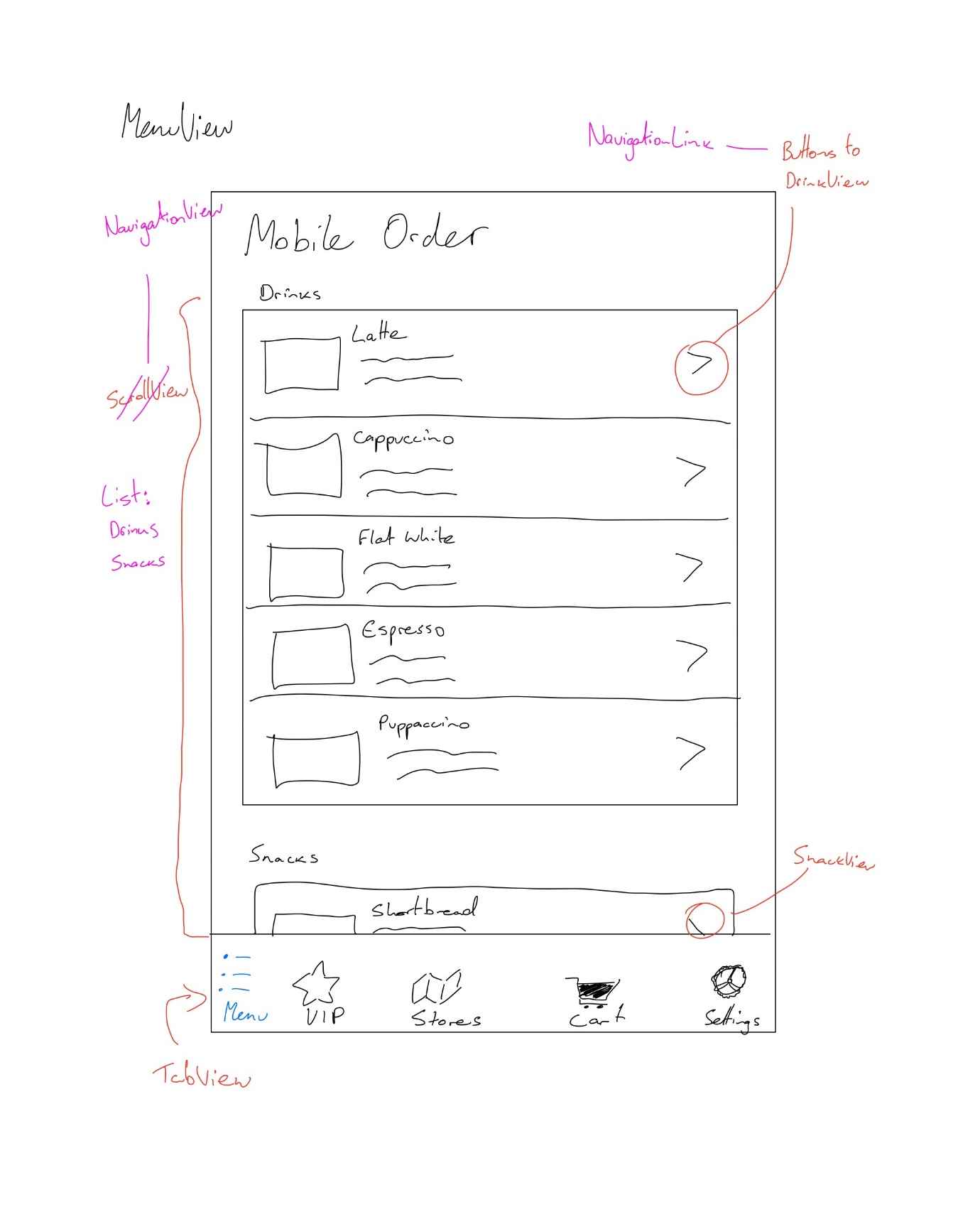
**Implementation:** I structured the app to pass environment objects (CartManager and ProductManager) across the views, ensuring that all views have access to shared data without manual propagation.

**Justification:** Although beyond the scope of this assignment, I wanted to maintain a clean architecture by reducing dependency chains and simplifying data access across the app's components. I implemented it in Module 5’s MenuApp extension activity. This approach is excellent for data consistency and state management, especially in complex apps where multiple views need to access and modify shared data (Apple Inc., n.d.; Hudson, 2024).

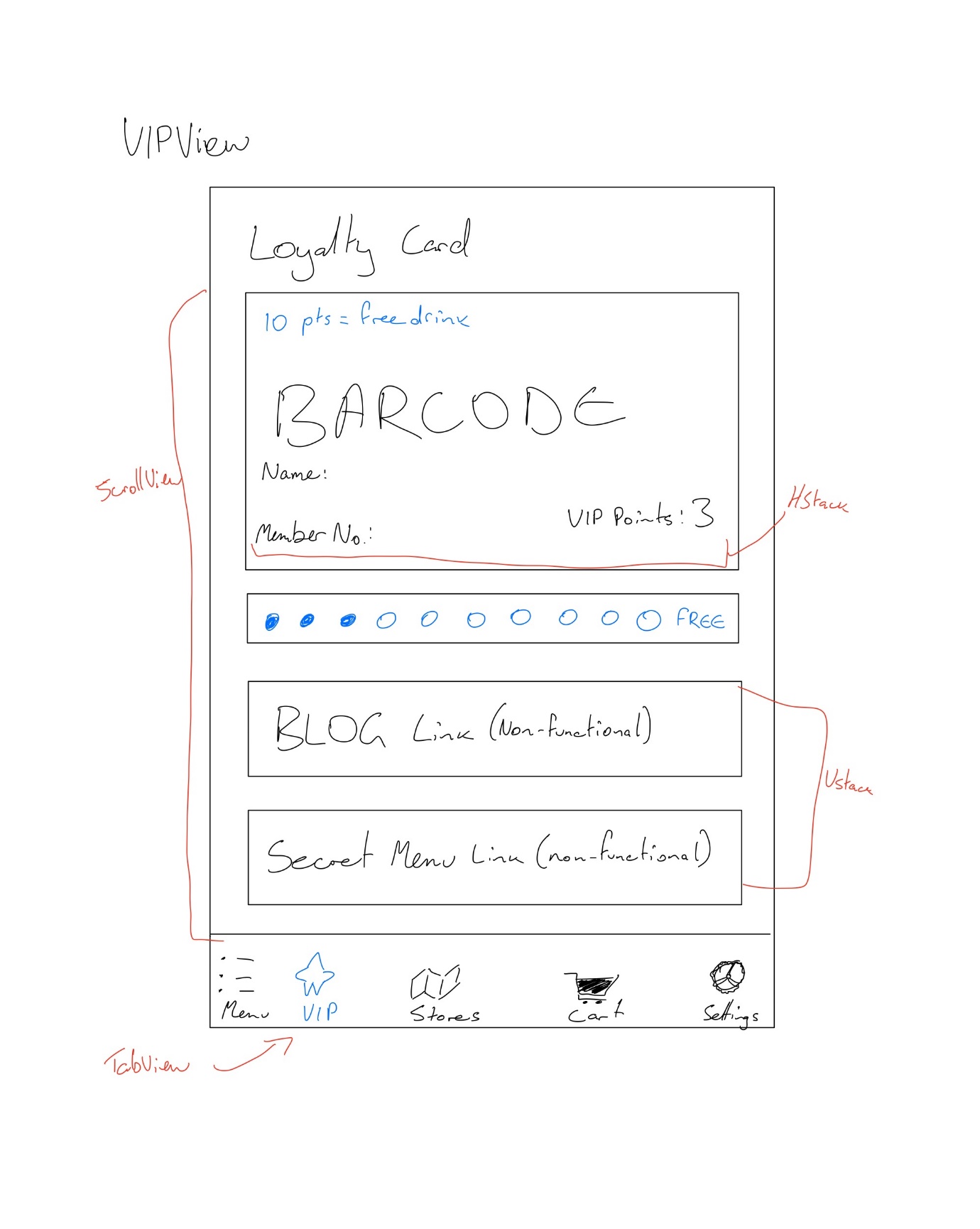
# 3.0 Design of Views

View, Control and Navigation Item from the SwiftUI user interface library which the View (screen) is composed of is annotated on each design *(See Appendix 8 for draft designs)*.

## 3.1 MenuView



## 3.2 VIPView



## 3.3 MapView

A drawing of a waterfall

Description automatically generated

## 3.4 CartView

A white paper with red writing

Description automatically generated

## 3.5 SettingsView

A close-up of a form

Description automatically generated

## 3.6 DrinkView

A close-up of a whiteboard

Description automatically generated

## 3.7 SnackView

A close-up of a document

Description automatically generated

# 4.0 App Construction

Please refer to the NeppastrevCoffee directory in the submitted zip file for App Construction.[[7]](#footnote-7)

# 5.0 Reflection and iPad Design Considerations

## 5.1 Reflections on Design

Firstly, I would advise the Client that their clientele will not be using iPads to order their coffees on-the-go and this would be a poor business decision. I would recommend the Client invests their iOS budget into improving and enhancing the phone app.

However, the primary difference between iPhones and iPads is that iPads have more screen space and changes in design should take advantage of this extra real estate:

1. **Multi-Column Layouts:** Utilising a split view or multi column layout on iPad would enable the app to show more content simultaneously. For example, in MenuView, the drinks and snacks could be displayed side by side using a two VStacks nested within an HStack, reducing the need for navigation and improving user engagement.
2. **Popover Menus and Detail Views:** For detailed views like DrinkView and SnackView, using popup presentations or slide-over panels (very common on retail websites to display cart) could enhance usability. This approach allows users to select and customise their orders without losing the context of the menu, which is particularly beneficial on larger screens.
3. **Additional Features:** It's possible to add more detailed descriptions, richer images, and additional information like nutritional facts, reviews or recommendations without ruining user experience.

## 5.2 iPad Design Adaptations

A white paper with writing on it

Description automatically generated

From here, when a user clicks on an item, the DrinkView/SnackView could popup rather than change views:

A close-up of a paper

Description automatically generated

A much more seamless feel and integration that takes advantage of extra screen space and would not be viable on iPhone.

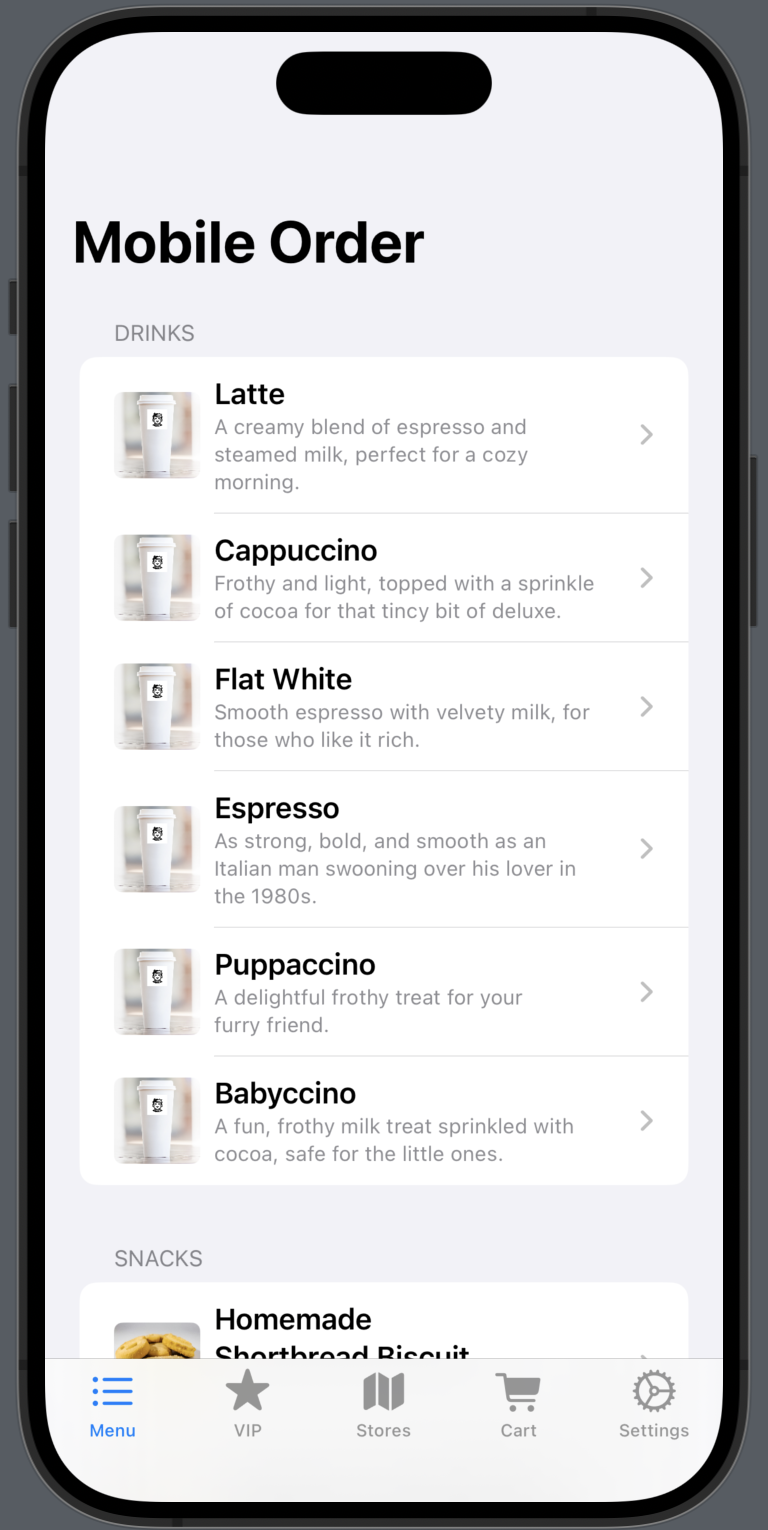
# 6.0 References

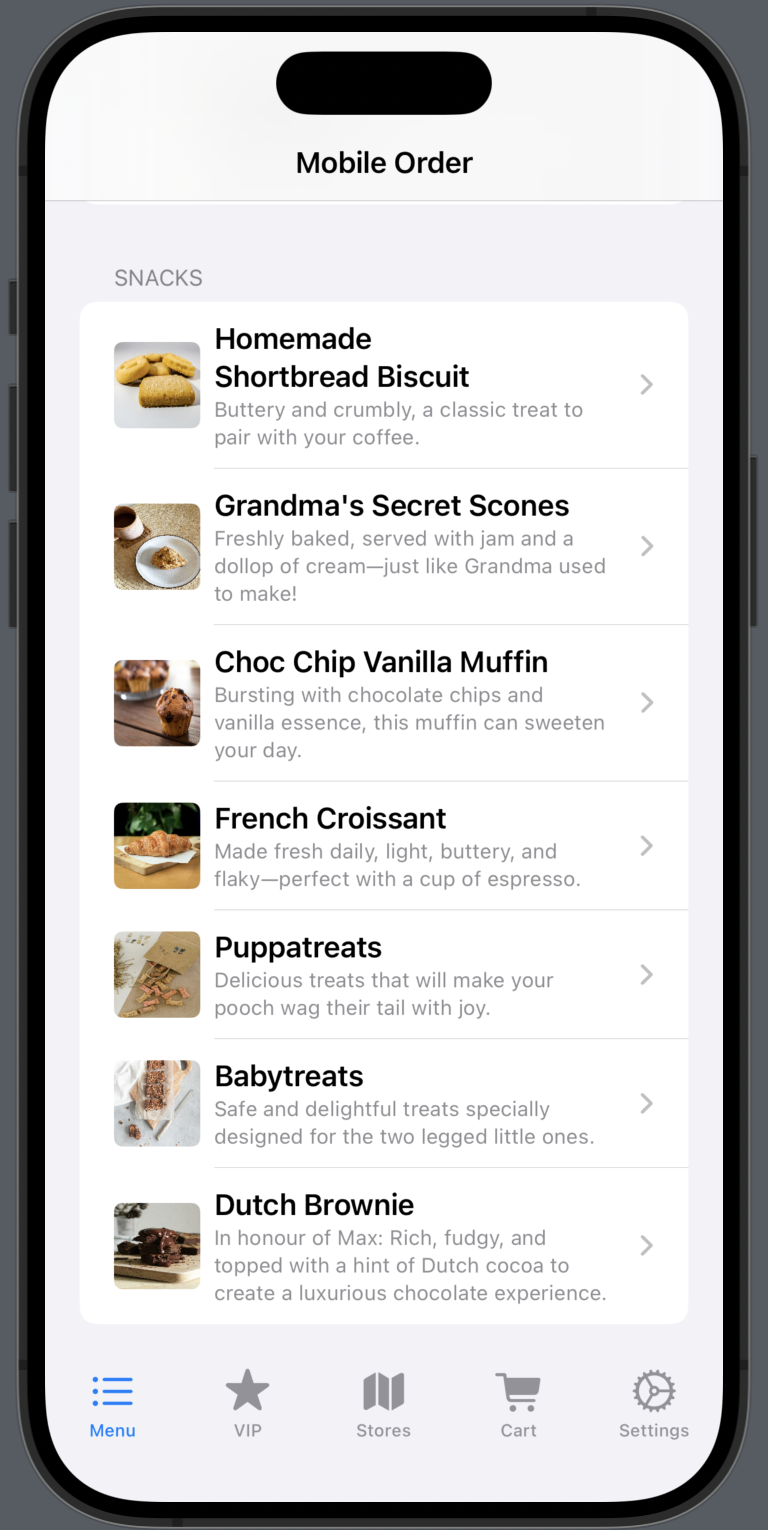
APA 7th Edition Referencing Style

* Apple Inc. (n.d.). Building lists and navigation. SwiftUI Tutorials. Retrieved from <https://developer.apple.com/tutorials/swiftui/building-lists-and-navigation>
* Apple Inc. (n.d.). EnvironmentObject. Retrieved from <https://developer.apple.com/documentation/swiftui/environmentobject>
* Apple Inc. (n.d.). NavigationLink. SwiftUI. Retrieved from <https://developer.apple.com/documentation/swiftui/navigationlink>
* Apple Inc. (n.d.). NavigationView. SwiftUI. Retrieved from <https://developer.apple.com/documentation/swiftui/navigationview>
* Hudson, P. (2024, February 1). Sharing data across tabs using @Environment. Hacking with Swift. Retrieved from <https://www.hackingwithswift.com/articles/202/sharing-data-across-tabs-using-environment>

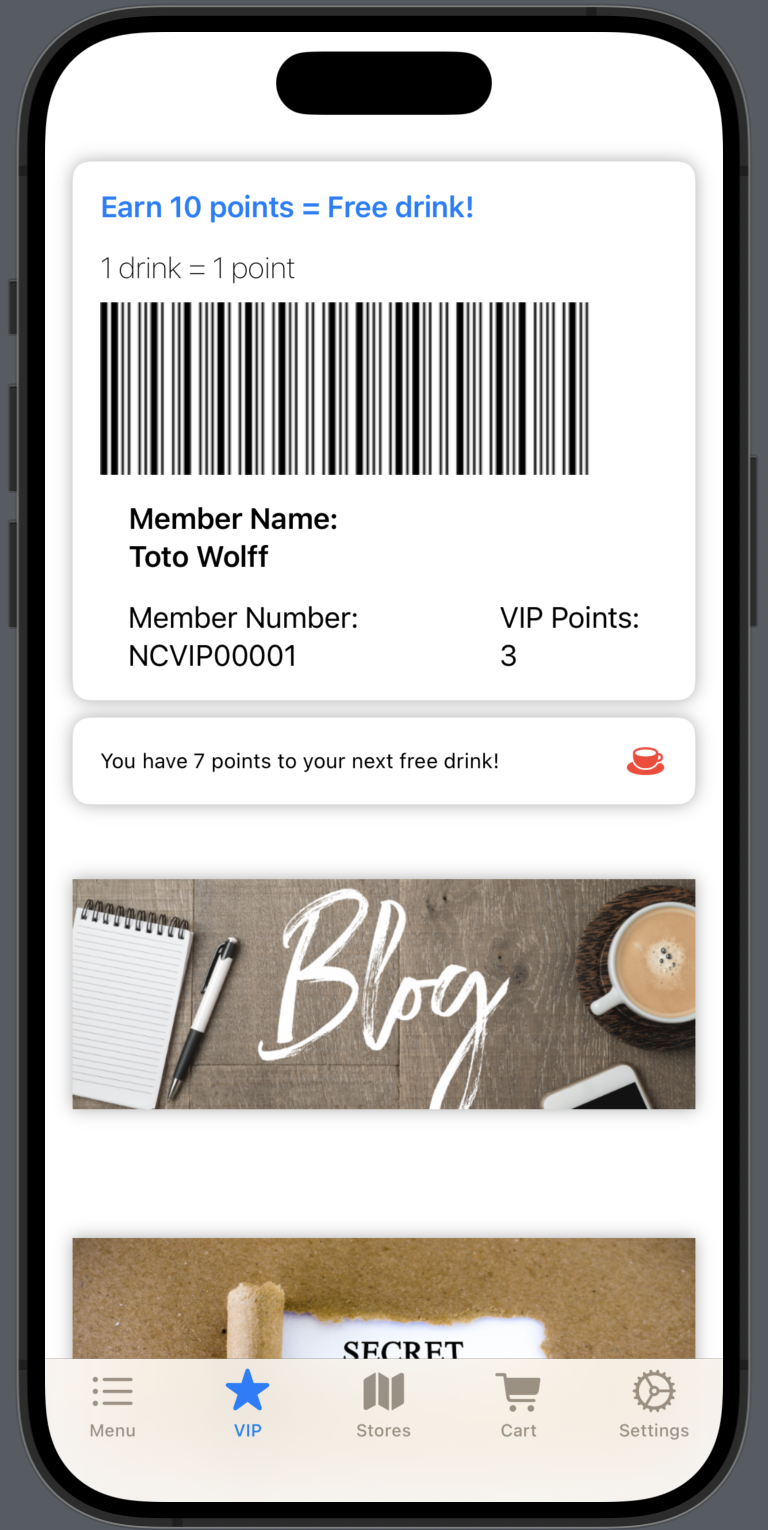
# Appendices

## MenuView:

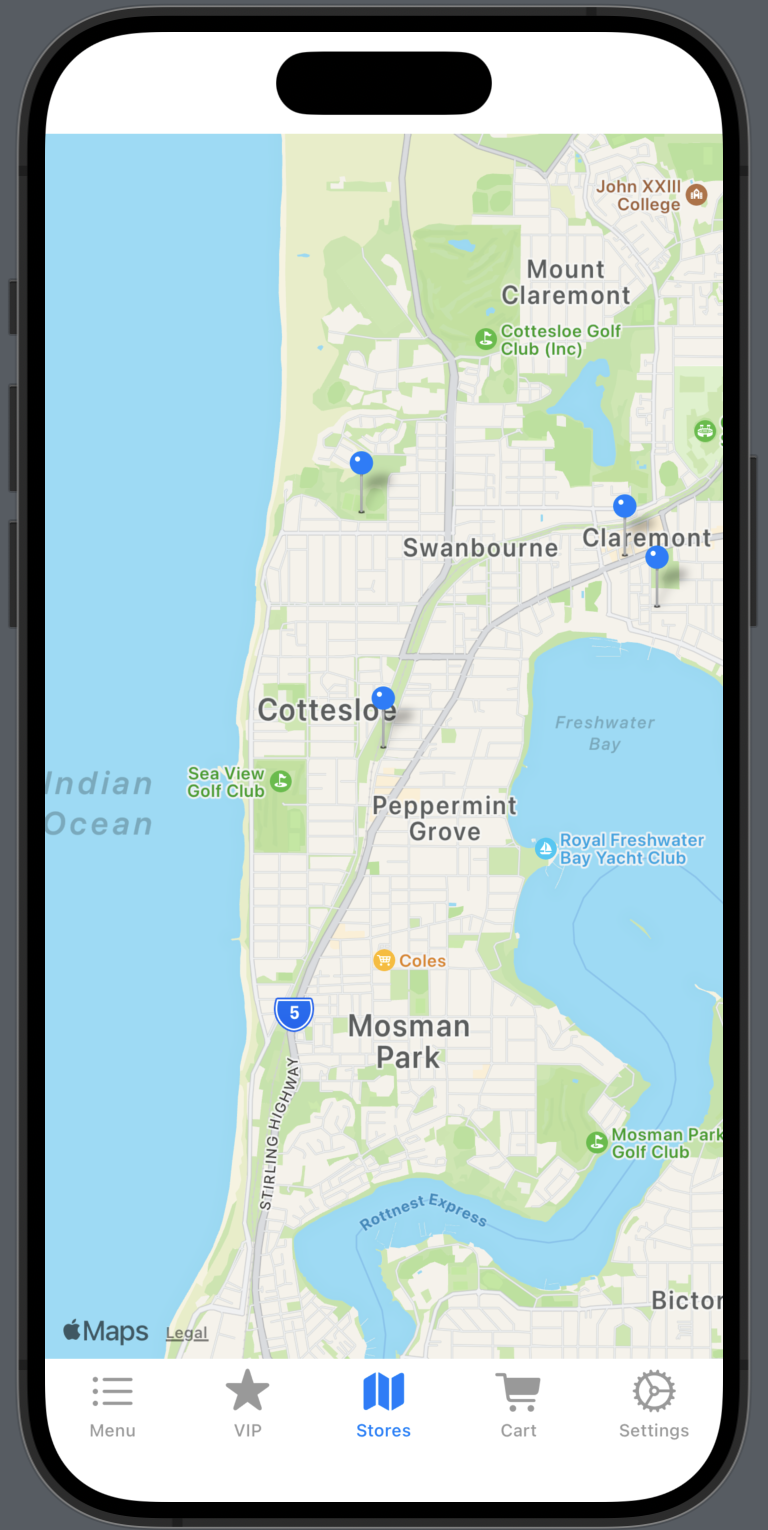




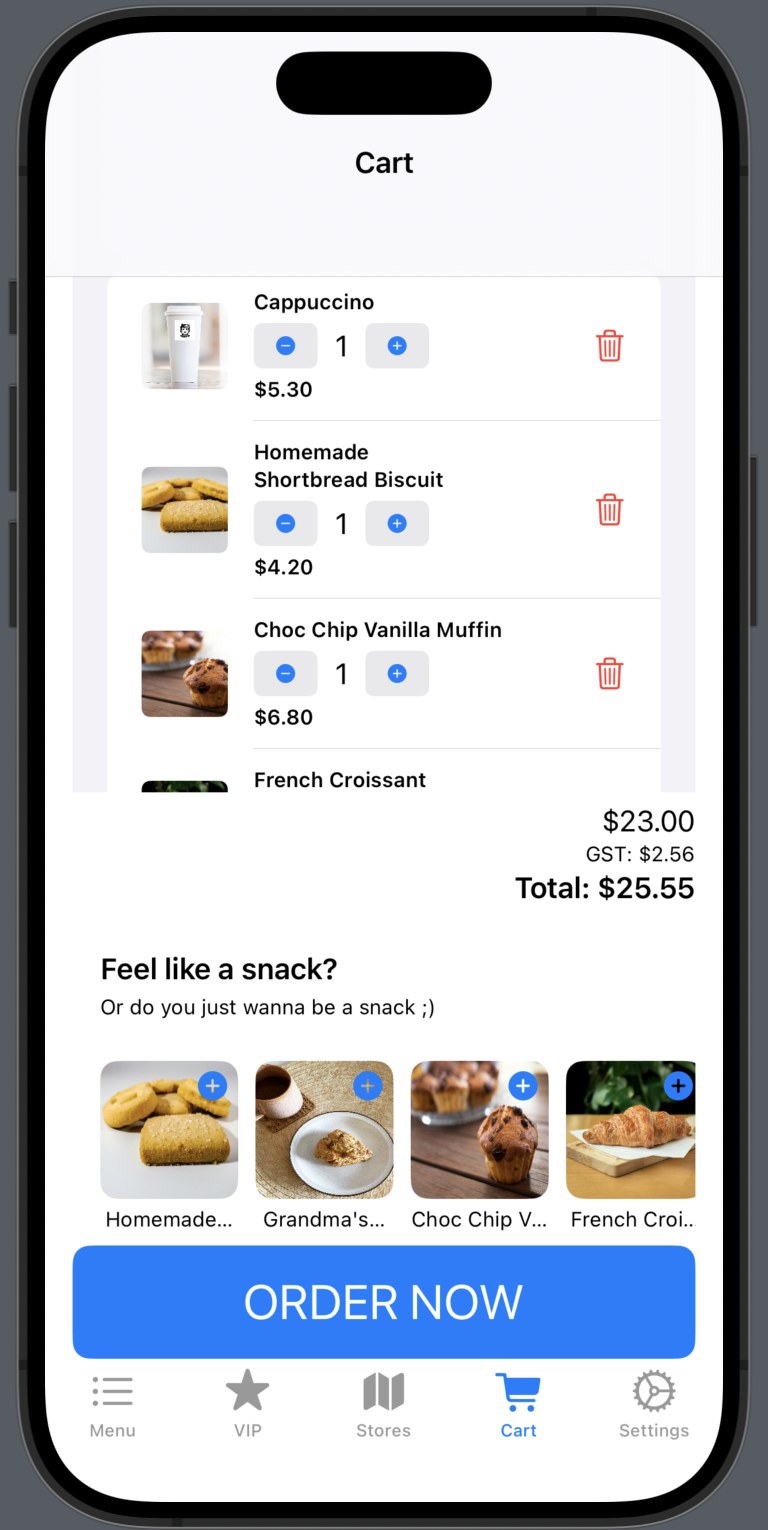
## VIPView:



## MapView:



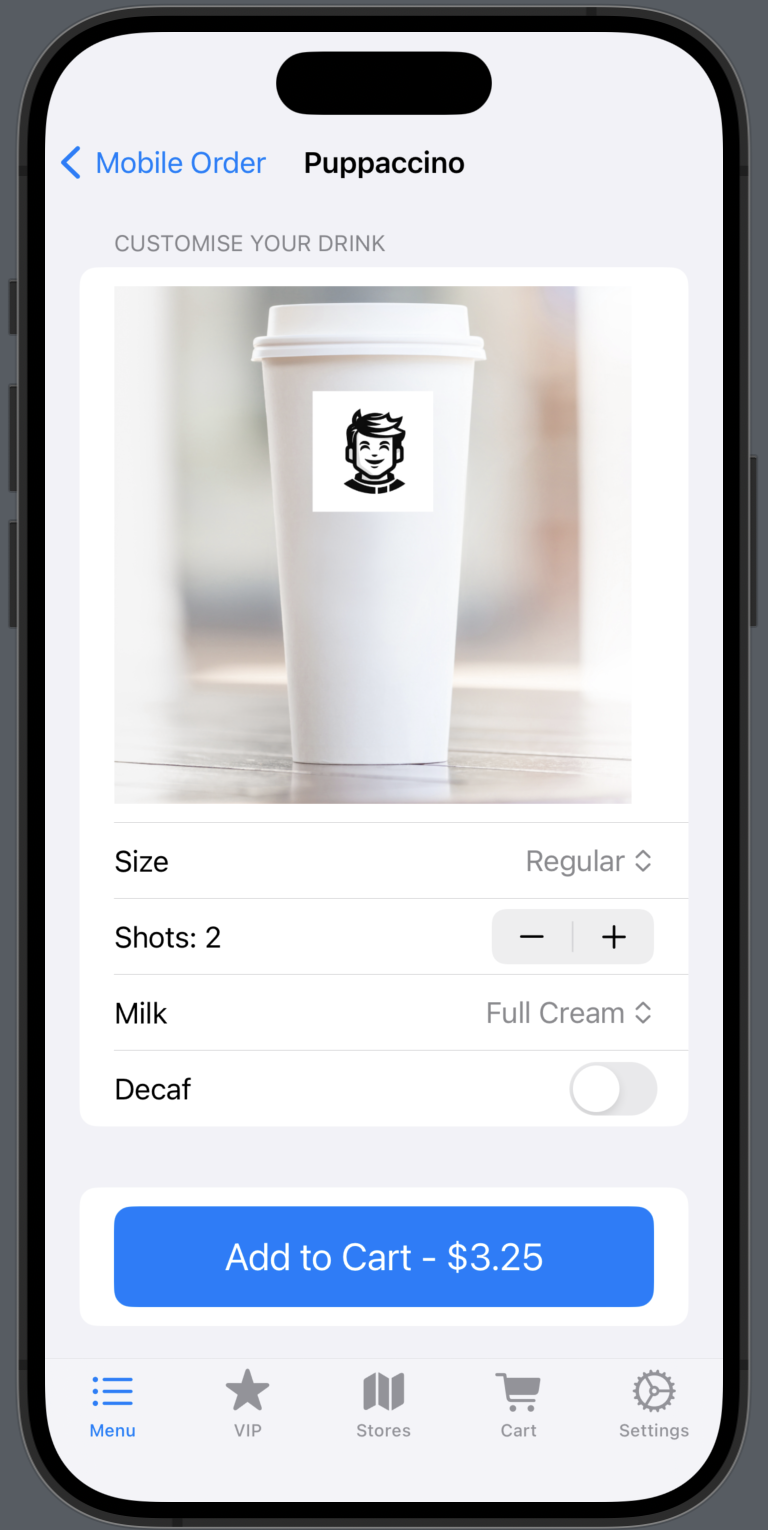
## CartView:



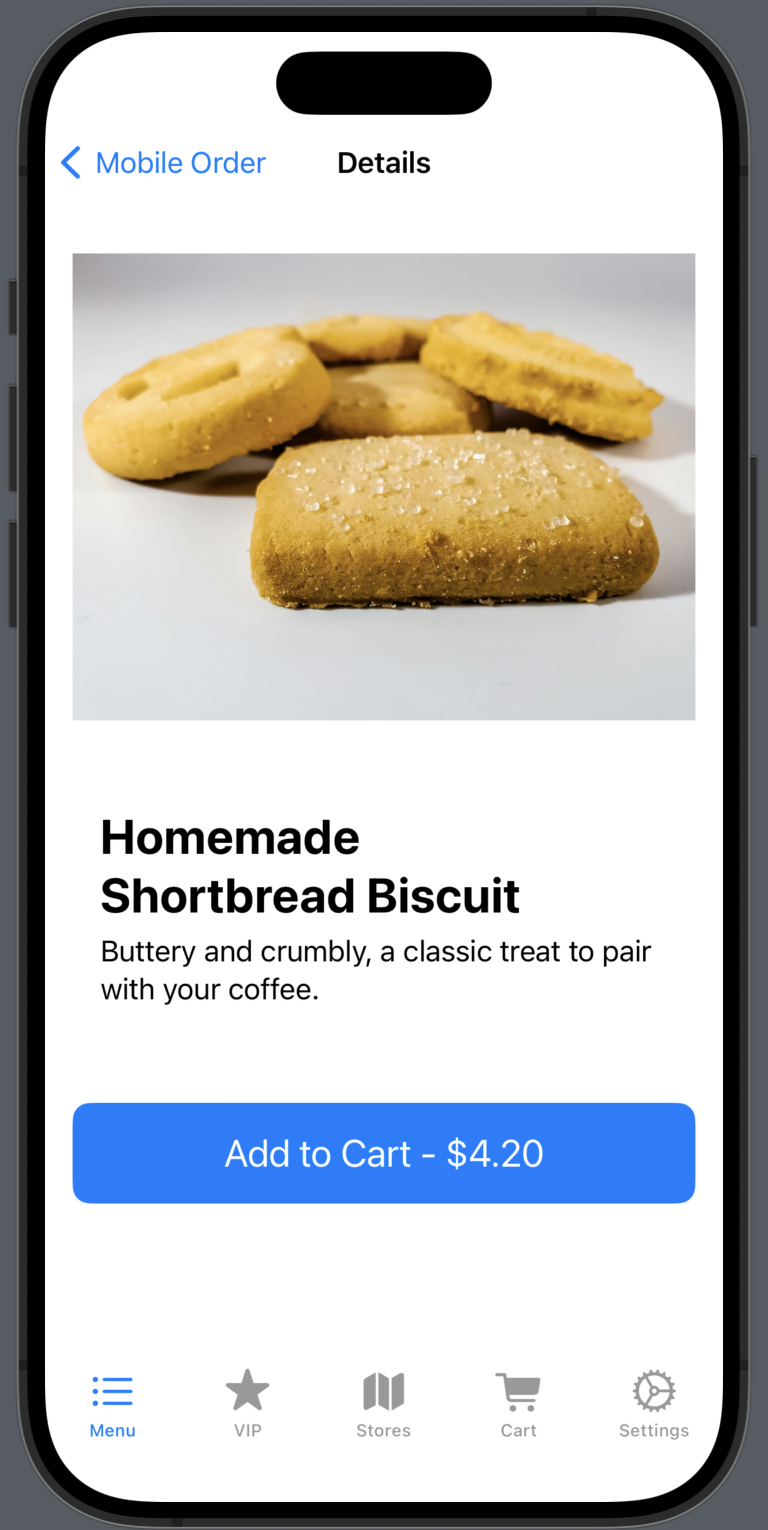
## SettingsView:



## DrinkView



## SnackView:



## Draft Designs

A line drawing of a computer

Description automatically generated with medium confidence

A close-up of a paper

Description automatically generated

1. As per task specifications, there is no functionality of the map other than displaying the pins of the four locations. [↑](#footnote-ref-1)
2. Although not included in the task specifications, Cart functionality was deemed an essential view to be included in the Proof-of-Concept design. [↑](#footnote-ref-2)
3. Although not included in the task specifications, a Settings Screen was deemed an essential view to be included in the Proof-of-Concept design. Note, this view is only hardcoded for display purposes and there is no functionality. [↑](#footnote-ref-3)
4. Although not included in the task specifications, Market research (my wife) strongly suggests snacks are essential for a café to be successful, so SnackView has been included in the Proof-of-Concept design. [↑](#footnote-ref-4)
5. A benefit of tearing ligaments in my ankle and requiring an extension is that I discovered the amazing (and very popular) TabView! I first used TabView in my Module6 AppControls application. [↑](#footnote-ref-5)
6. This was inspired by the Student Object Class we had PDI students create during this week’s lab (Worksheet 9). [↑](#footnote-ref-6)
7. Please also note NeppatsrevCoffeeApp and ProductManager which are not SwiftUI files. [↑](#footnote-ref-7)