iShnak IOS

iPhone Health Application

Author: Huw Williams

Number: 30051129

Lecturer: Alun King

Table of Contents

Introduction 3
Design Outline 4
Evaluation 7
Conclusion 8
References 8
Appendix I 9
Tab View 9
Home View 10
Ring Views 11
Weekly View 12
Figure 6: The four Ring Views 13
Profile View 13
Widget Extension 14
Appendix II 15
How To display a SwiftData (SQLite) DB 15

Introduction

iShnak iOS is a personal wellness iPhone application (app) focused on hydration and calorie tracking. Created using SwiftUI and SwiftData, the app enables users to monitor their daily intake of water, beverages, meals, and snacks through an interactive activity ring interface. The project explores key aspects of mobile development, including data persistence, user experience, and Apple HealthKit integration. By addressing real-world health management needs, iShnak iOS serves as both a practical application for users, and a learning opportunity in applying software engineering principles within the Apple development environment.

Design Outline

The design of iShnak iOS is inspired by the Apple Watch's activity ring interface first introduced in In WWDC23 (Apple Developer, 2024). Each activity ring on the *Home View* acts as a navigational element that allows users to access dedicated views for logging water, meals, snacks, and beverages. These rings are located in the upper-right quadrant of the screen for easy access via the user's right thumb, aligning with observations from informal user testing and Apple's Human Interface Guidelines (Apple Inc, 2024).

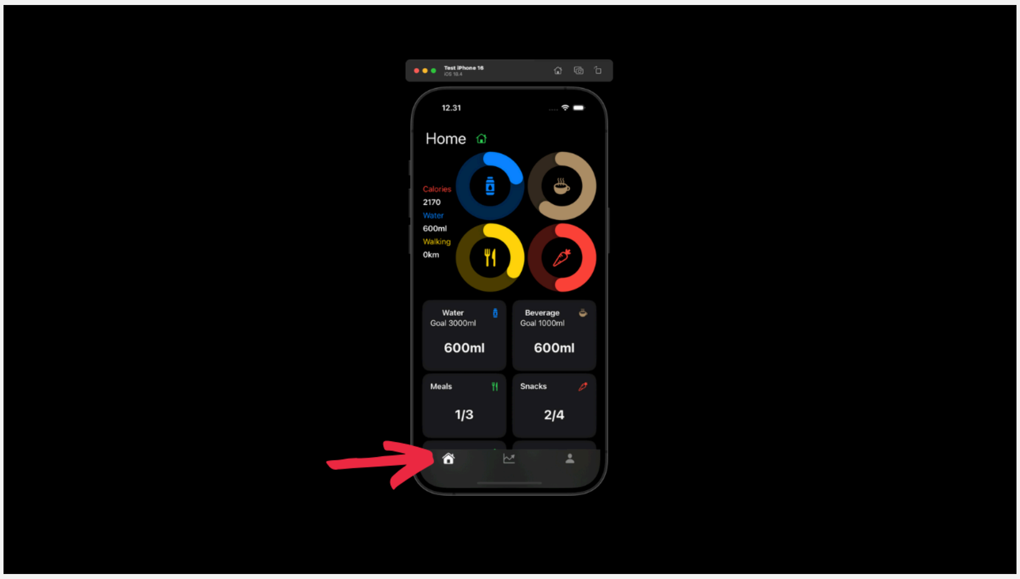


Figure 1: TabView displaying HomeView.

A *TabView* serves as the primary navigation framework, giving users access to the Home View, Weekly View, and Profile View. Below the activity rings on the Home View are quick-glance panels that display up-to-date user metrics such as calories consumed and HealthKit-provided step counts. The ring views have been enhanced beyond the WatchOS implementations by allowing users to select input values before tapping the ring, and by providing "Reset" and "Undo" buttons. Features not available in the original watch implementation.

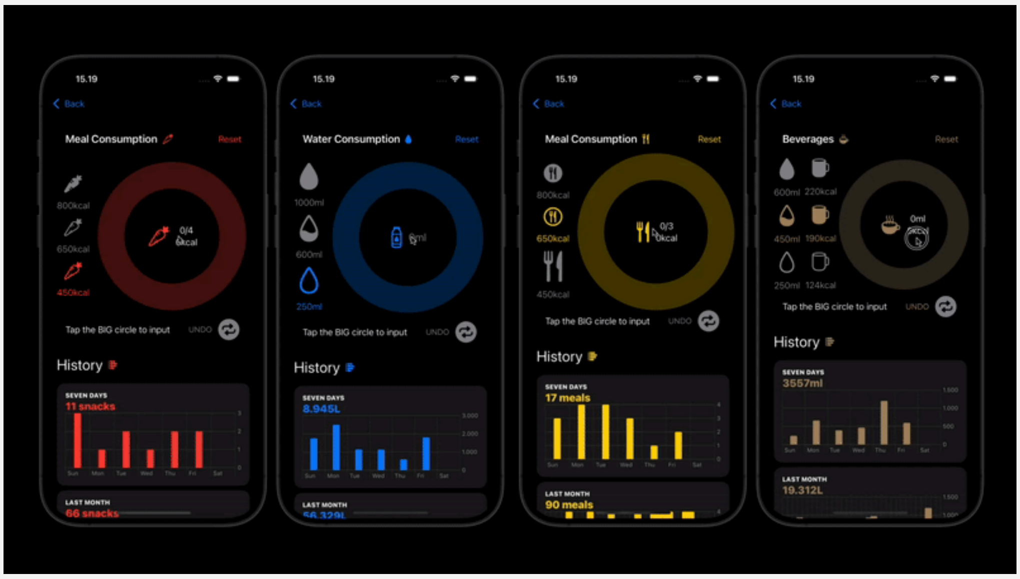


Figure 2: Enhanced Ring View.

The *Weekly View* offers a scrollable list of historical data, where each day is represented as a tappable card that links to a detailed chart of the day's consumption. Finally, the *Profile View* gives users control over their goals and notification settings for meals and hydration reminders. These improvements attempt to provide both convenience and improved data browsing.

The app includes a Widget Extension built with SwiftUI and WidgetKit. This widget offers users a quick overview of hydration and meal progress directly from the home screen. By displaying simplified activity rings and key metrics, it encourages users to stay engaged with their health goals throughout the day. The widget pulls data from the app's SwiftData container using a shared App Group, ensuring consistent updates between the app and widget.

Full Design Outline: Appendix I

Evaluation

Firstly, it is important to mention that a trial period was set aside to attempt a connection between the WatchOS app and an iOS app which failed. Therefore, iShnak iOS is a standalone application and is not influenced by a companion watch app.

The app was designed to visualise enough data to satisfy without overwhelming the user. A session was expected to exceed the WatchOS app session by about ten seconds, with the option to stay on the app to browse progress and history data.

The iOS app has succeeded in expanding on the original Activity Ring idea by implementing a system by which users can select the incrementation amount before tapping the ring, along with the options to undo and reset. Which was not available in the watch app.

A Widget Extension was successfully completed to enhance the app, it is updated every 3 minutes to represent the data available on the app and uses a shared SwiftData container with iShnak iOS. However, the Widget Extension will not deploy to an actual device when used in place of a simulator. There seems to be a setting within Build Settings or Build Phases that is likely incorrect or missing.

SwiftData caused many delays until a directory location for the database was printed to the console. This allowed access via VSCode using the SQLite and SQLite Reader extensions. Troubleshooting improved once the database could be displayed as tables with rows and columns.

Appendix II: How to Use VSCode to View SwiftData Database.

A bug remains on the *Profile View*. When the screen is first initialised, the "changed" overlay animation executes. This will be fixed in future developments.

Conclusion

In conclusion, iShnak iOS successfully delivers an iphone application for tracking hydration and calorie intake. Built using Apple design concepts, HealthKit integration, and user-defined features like customisable reminders and activity rings. The development process has resulted in a better understanding of SwiftUI, SwiftData, and system architecture. Overall, iShnak showcases functional and User Interface improvements spanning both iShack WatchOS and iShnak iOS.

References

Apple Inc. (2024) ‘Human Interface Guidelines: Designing for iOS’, *Apple Developer Documentation*. Available at: https://developer.apple.com/design/human-interface-guidelines/designing-for-ios, (accessed: 17th April 2025).

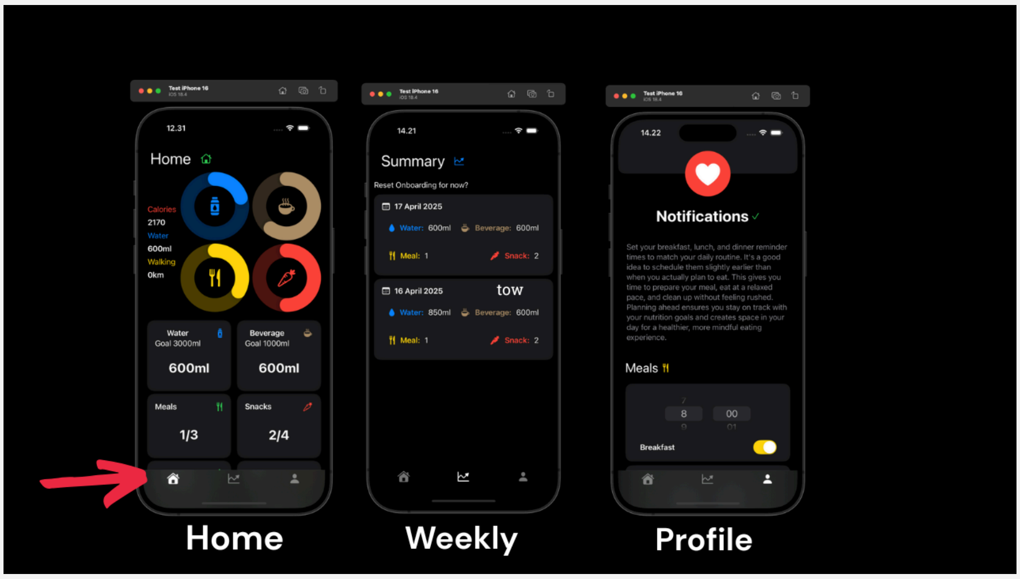
Apple Inc. (2024) ‘iOS: Planning your iOS app’ *Apple Developer Documentation*. Available at: https://developer.apple.com/ios/planning/, (accessed: 10th April 2025).

Apple Inc. (2024) ‘HealthKit’ *Apple Developer Documentation*. Available at: https://developer.apple.com/documentation/healthkit (accessed: 10th April 2025).

Appendix I

Tab View

The design of iShnak iOS follows the Apple Watch interactive rings theme to record user input, which aligns with dial-based views (Apple Developer, 2024). However, each ring on the *HomeView* navigates to a separate view that offers further options for recording entries.



Set your breakfast, lunch, and dinner reminder times to match your daily routine. It's a good idea to schedule them slightly earlier than when you actually plan to eat. This gives you time to prepare your meal, eat at a relaxed pace, and clean up without feeling rushed. Planning ahead ensures you stay on track with your nutrition goals and creates space in your day for a healthier, more mindful eating experience.

Figure 3: All views nested in the TabView.

A *TabView* offers another navigation framework that lets the user move between the *HomeView*, *WeekView* and the *ProfileView*.

Home View

Users interact with their iPhone device using both hands[1], at a distance of a foot. This led to the decision of placing the tappable activity rings at the top-trailing (right) side of the screen, so users activate the ring using their right thumb[2]. Text and icons were placed on the leading edge to satisfy English-language users[3].

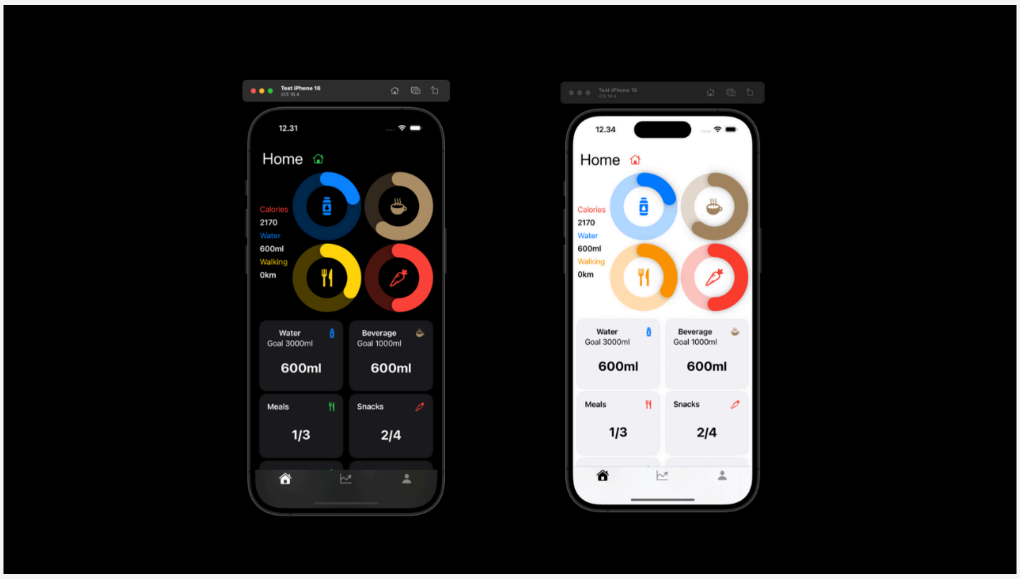


Figure 4: Home View in Dark and Light mode.

Each ring within the Home View navigates to a specific view.

Quick-glance panels sit below the activity rings and display relevant user data. Daily goals and HealthKit[4] metrics like step count, are also available. Where the iShnak Apple Watch (WatchOs) app relies on icons to interpret data categories, the iOS app

[1] According to Apple's Human Interface Guidelines (Apple Inc, 2024)
[2] Interestingly, 4 students in the room were asked to interact with the app using both hands. All of them naturally used their right thumb to initiate the activity ring navigation, even a left handed person.
[3] English users naturally read from the left.
[4] Healthkit SDK lets iPhone apps access native fitness data and display it as required.

benefits from text headings and more detailed information for the user to monitor.

Ring Views

The ring views are a marked improvement to the Apple Watch app views. Each activity ring view has a layout that displays a tappable activity ring similar to the WatchOS App. The Water, Meal, and Snack views display a single row of icons representing an amount millilitres or calories. The user is able to select an icon before tapping the activity ring, which increments the user's daily consumption by that amount.

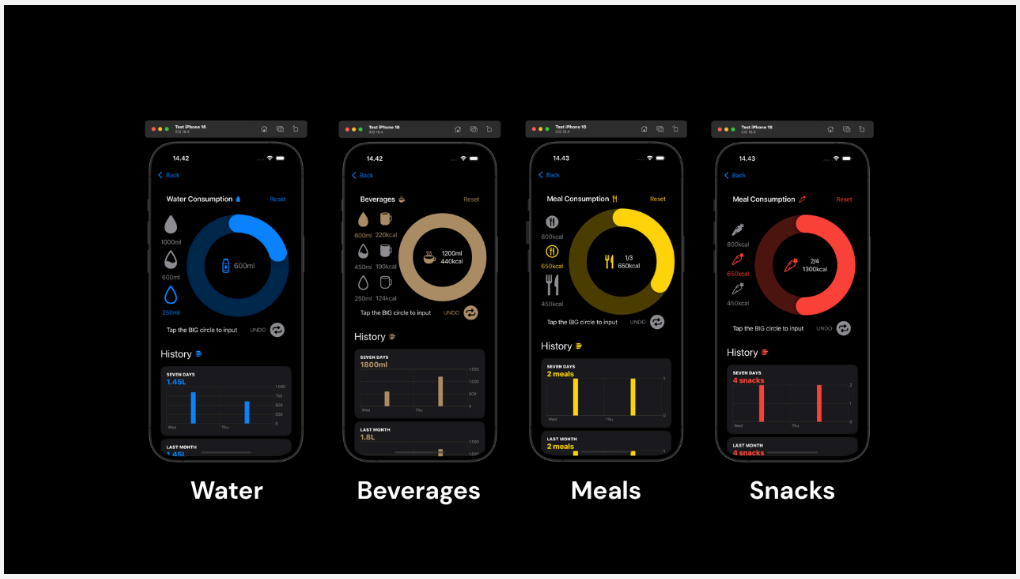


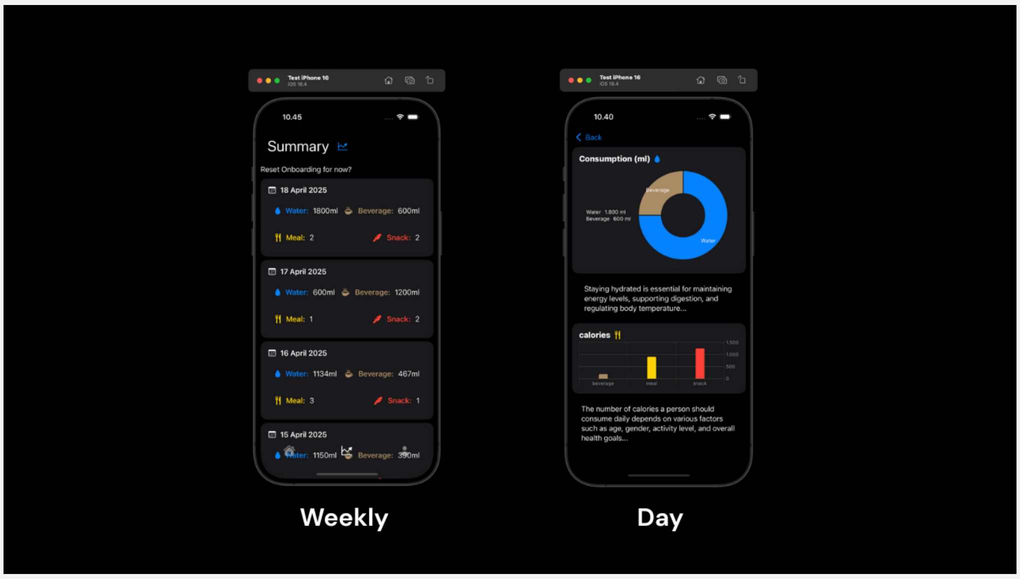
Figure 5: The Four Ring Views .

The ring has a “reset” text button that completely resets the ring, and an “UNDO” button that decrements the last input. Both buttons decrement the daily total. This improves on the WatchOS app by offering both features where the WatchOS app offered none.

The iOS app further improves the activity ring views by displaying weekly and monthly history data for multiple categories.

Weekly View

The *Weekly View* is a List embedded in a ScrollView that displays the user's daily history. Originally the view was designed to show seven days of data, however it was extended to display all past data.



Reset Onboarding for now?

18 April 2025
Water: 1800ml Beverage: 600ml
Meal: 2 Snack: 2

17 April 2025
Water: 600ml Beverage: 1200ml
Meal: 1 Snack: 2

16 April 2025
Water: 1134ml Beverage: 467ml
Meal: 3 Snack: 1

15 April 2025
Water: 1150ml Beverage: 390ml

Water 1.800 ml
Beverage 600 ml

Staying hydrated is essential for maintaining energy levels, supporting digestion, and regulating body temperature...

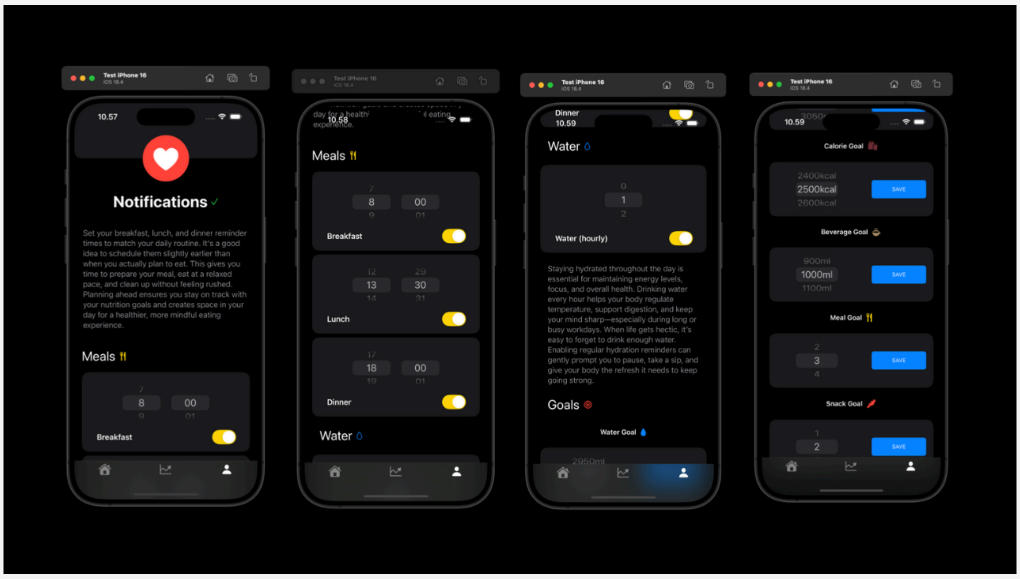
The number of calories a person should consume daily depends on various factors such as age, gender, activity level, and overall health goals...

Figure 6: Daily Consumption View .

Each card represents a day and is a navigation link that opens a detailed record of the day's consumption, which is visualised by charts.

Profile View

The *Profile View* lets the user set notifications for breakfast, lunch, and dinner. Reminders can also be set for water consumption throughout the day.



Set your breakfast, lunch, and dinner reminder times to match your daily routine. It's a good idea to schedule them slightly earlier than when you actually plan to eat. This gives you time to prepare your meal, eat at a relaxed pace, and clean up without feeling rushed. Planning ahead ensures you stay on track with your nutrition goals and creates space in your day for a healthier, more mindful eating experience.

Staying hydrated throughout the day is essential for maintaining energy levels, focus, and overall health. Drinking water every hour helps your body regulate temperature, support digestion, and keep your mind sharp—especially during long or busy workdays. When life gets hectic, it's easy to forget to drink enough water. Enabling regular hydration reminders can gently prompt you to pause, take a sip, and give your body the refresh it needs to keep going strong.

Figure 7: Profile View.

The iOS app improves this feature by offering more control to set and remove mealtime notifications. The iOS app further improves user experience by providing the option to reset all goals.

Widget Extension

The *iShnak* app includes a Widget Extension that provides glanceable insights into the user's hydration and calorie progress. Built with SwiftUI, the widget displays simplified activity rings for water and beverage intake, using familiar icons and colors consistent with the main app. It reads data from the shared SwiftData container via App Groups, displaying updates across both app and widget. By offering access to daily goals, the widget improves user engagement while encouraging healthy habits effortlessly.

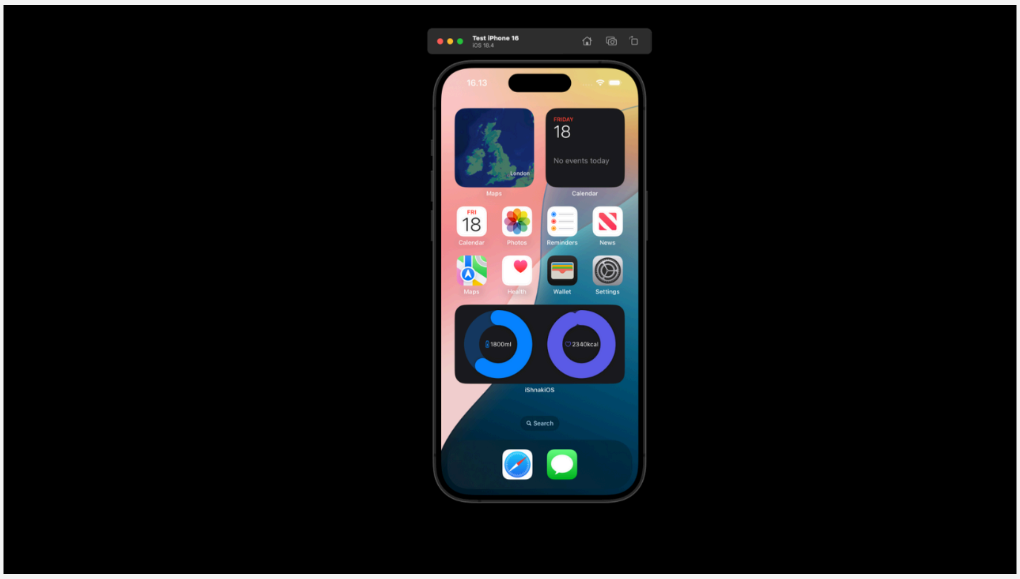


Figure 8: Widget Extension.

Appendix II

How To display a SwiftData (SQLite) DB.

If the developer is using the default location for SwiftData with this code snippet:

```
let sharedURL =
FileManager.default.containerURL(forSecurityApplica
tionGroupIdentifier: "group.com.iShnak.shared")!
```

A print statement can be placed in the init(),

```
if let url =
FileManager.default.containerURL(forSecurityApplicatio
nGroupIdentifier: "group.com.iShnak.shared") {

print("Shared container is at: \(url.path())")
}
```

which will display:

```
/Users/huwwilliams/Library/Developer/CoreSimulator/Devices
/51828229-AF54-47DE-BE27-9F106FD5F04B/data/Containers/Sha
red/AppGroup/8B600A61-B596-40B1-91C2-09EF8047206D/
```

In the Console.

Highlight this address, scroll down to Services/Show-in-finder.

This will open the location. Right-click on the database file, and open with. Find VSCode in the options.

Ensure you have installed both extensions.

If a bunch of binary appears, click on the alert bar above and trust the file. This can be difficult to find.