Jeffrey Flores

September 21, 2025

CS-499 – Milestone Two – Narrative

Weight Trackit is a mobile application designed to make weight logging and goal tracking fast, private, and motivating. Core features include daily weight entry, goal setting, progress visualization, real-time sync via Firebase (so entries persist across devices), and milestone notifications when targets are hit. The product features a clean, low-friction UI designed for users who prioritize functionality over clutter. The initial Android version launched during my Mobile Architect and Programming course in April 2025, and I created the iOS version in September 2025, bringing the project to a multi-platform state.

I selected this app because it represents the kind of work I want to do professionally, mobile engineering tied to cloud data and user-centric design. It showcases my mobile development skills with me doing an Android to iOS port. Also, it shows my ability to use cloud integration with Firebase Realtime Database (NoSQL modeling, live updates) and secure read/write rules. I am also showcasing my ability to work with notifications, sending alerts using platform-appropriate implementations. Finally, it shows my software engineering craft by allowing me to show my use of modular architecture, version control discipline, and iterative refinement from a single-platform MVP to a polished multi-platform app.

The largest improvement was porting from Android to iOS, which required rethinking layout (Auto Layout, safe areas, dynamic type) and replacing Android-specific behaviors with native iOS approaches. I also refined the data model and synchronization logic to ensure consistent cross-platform behavior and added small UX touches (empty-state screens, friendlier error messages, and guardrails on input).

One of the course outcomes that I met with this enhancement and with this artifact was the algorithm design and evaluation. This allowed me to model time series entries, compute trends/targets, and choose NoSQL schemas that optimize reads for charts. I was also able to hit the Engineering tools and practice outcome by using Firebase SDKs, dependency managers, linting, and crash/analytics dashboards. I was able to organize code into testable modules. I also developed a security mindset, which is another course outcome. I enforced Firebase security rules, least-privilege access, Keychain storage for sensitive tokens, and also added 2FA to the mobile app through email verification to avoid misuse of the app and its data.

Porting taught me that features that feel “free” on one platform often require different primitives on the other. For example, Android-style SMS triggers aren’t appropriate on iOS; the iOS version instead uses local/push notifications routed through platform channels. I also learned that Auto Layout demands more upfront work than some Android layouts but presents better results across device sizes when done right. On the tooling side, integrating the latest Firebase libraries with a new Xcode version caused instability; I resolved this by cleaning the build, re-installing dependencies, and restarting Xcode (the issues disappeared after updates. Finally, I strengthened my privacy posture: I audited what data truly needs to sync, reduced over-collection, and validated rules to ensure users can access their own records.