3-2 Milestone Two: Enhancement One: Software Design and Engineering

Jorgo Qendro

Southern New Hampshire University

CS-499: Computer Science Capstone

Professor Nembhard

June 12th, 2025

**Milestone Two: Software Design/Engineering Enhancement Narrative**

1. Briefly describe the artifact. What is it? When was it created?

The artifact I selected is my Weight Tracker Android application, which I originally created during CS-360: Mobile Architecture and Programming. This application enables users to track their weight entries over time in a mobile environment and was designed as part of a course that focused on building native Android apps using best practices. It was developed using Kotlin and Android Studio and includes basic functionality such as form entry, date selection, and storage of weight data. At its core, the app serves a simple but clear purpose, making it an ideal candidate for targeted software design enhancements.

**2.** Justify the inclusion of the artifact in your ePortfolio. Why did you select this item?What specific components of the artifact showcase your skills and abilities in softwaredevelopment? How was the artifact improved**?**

I selected this artifact for my ePortfolio because it provides an excellent opportunity to demonstrate my software engineering and design skills. Compared to my more complex project choice, aka the Animal Shelter dashboard, the Weight Tracker app is focused on scope, allowing for deeper, more measurable enhancements in code structure and reliability. This simplicity made it easier to apply and highlighted professional practices, particularly in improving architectural design and code maintainability.

The primary improvement made to the application was restructuring its code using the Model-View-ViewModel (MVVM) architecture. Previously, the app followed a more monolithic structure, with UI components directly handling data logic. By introducing ViewModel and Repository components, I created a clear separation of concerns between the user interface, business logic, and data handling. This makes the codebase significantly more testable, readable, and scalable. I also implemented comprehensive input validation to ensure that users cannot enter unreasonable weight values or incorrectly formatted dates. These validations enhance reliability and prevent app crashes while providing meaningful feedback to users. Additionally, I introduced dependency injection using Hilt, improving how components are instantiated and managed across the app, which further contributes to modularity and maintainability.

3. Did you meet the course outcomes you planned to meet with this enhancement inModule One? Do you have any updates to your outcome-coverage plans?

Yes, I met the planned course outcomes outlined in Module One. Specifically, this enhancement aligns with Course Outcome #4: “Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals.” The implementation of the MVVM pattern demonstrates a well-founded software architecture approach, while the validation improvements and dependency injection show my ability to apply tools and techniques that improve reliability and maintainability. There are no updates to my outcome-coverage plans at this time, as the enhancements align fully with the intended goals.

**4.** Reflect on the process of enhancing and modifying the artifact. What did you learnas you were creating it and improving it? What challenges did you face?

The process of enhancing this artifact taught me valuable lessons in applying software architecture patterns within a live project. Implementing the MVVM pattern helped me understand how separating responsibilities across components improves clarity and reduces the risk of tightly coupled code. It also gave me practical experience in integrating Android architecture components like LiveData and ViewModel in a meaningful way.

One challenge I faced was updating the UI components to interact correctly with the new ViewModel layer without introducing bugs. Since the original code was structured in a tightly coupled way, I had to carefully refactor the logic in stages to avoid breaking existing functionality. Implementing form validation across both the UI and ViewModel layers was also tricky, particularly ensuring that the validation messages were intuitive and the user experience remained smooth. These challenges helped me refine my debugging skills and reinforced the importance of planning architectural changes carefully. Overall, this enhancement process gave me firsthand experience in how professional-grade software design improves the quality and longevity of an application.