Daniel Ruppert

Professor Steaanelli

CS-499

05/26/2025

Enhancement One: Software Design and Engineering

The software design and engineering artifact I selected for my ePortfolio is the EventSync Android application. This application was initially created during CS-360 in the Fall of 2024, and significantly enhanced throughout the CS-499 capstone experience. It is a full-featured scheduling tool that allows users to add, view, and manage both one-time and recurring events. The application includes user authentication, data persistence with SQLite, dynamic UI elements, and support for SMS notifications.

I chose this artifact because there were several features I had planned to implement during CS-360 but was ultimately unable to complete due to time constraints. The capstone provided the opportunity to return to this project and realize those enhancements. The improvements I envisioned aligned well with the focus areas of software engineering, data structures and algorithms, and database design. Key enhancements include refactoring core functionality into well-defined utility classes, introducing centralized logging for streamlined error tracking, and implementing persistent user sessions to maintain application state across activities. I also modularized lambda expressions to improve readability and reusability, resolved bugs in the registration process and SMS notification flow, and replaced ad hoc exception handling with a centralized and structured approach. Recurring event functionality was implemented both at the database schema level and within application logic, reflecting a scalable design strategy. Finally, the project’s package structure was reorganized with meaningful naming conventions, and new features, such as toggleable event views and weekday visualization, were added to enhance usability and maintainability.

With this enhancement, the outcome I listed in Module One actually ended up being more applicable to enhancement two, so I revised my outcome coverage plan accordingly. For this enhancement, the course outcomes align more closely with Outcomes 1, 2 and 4. Employing strategies for building collaborative environments, supporting quality communication, and demonstrating the use of well-founded and innovative techniques, respectively. For outcome 1, while the capstone was an individual effort, I leveraged Git and GitHub for full version control, enabling clear documentation of changes, commit history, and rationale behind architectural decisions. Outcome 2 is reflected through consistent in-line documentation, meaningful code comments, and organized project structure that together ensure the application is technically sound and easily understandable to a variety of audiences. Outcome 4 is demonstrated through the integration of persistent session management, centralized error logging, modular utility classes, and Android-specific UI components like RecyclerViews. These enhancements reflect not only strong foundational software engineering skills but also thoughtful application of industry-relevant tools and design patterns to deliver a robust, maintainable solution.

The process of enhancing and modifying this artifact was both challenging and rewarding, providing numerous opportunities to deepen my technical skill set. One of the key areas of growth was learning how to effectively use consumers to create more dynamic, callback-based functionality in my code, which helped modularize UI interactions. I also gained experience working with Android’s InputFilter framework to constrain user input at the UI level, improving both validation and user experience. Another major improvement was replacing System.out.println and e.printStackTrace() with Android’s built-in logging framework, which not only produces cleaner logs but also aligns with platform best practices for debugging and production readiness. I strengthened my understanding of UI design by exploring how component relationships affect layout behavior across different devices and screen sizes, ensuring a consistent and intuitive user experience. Lastly, implementing an LRU (Least Recently Used) cache taught me valuable lessons about memory-efficient data retrieval, especially for frequently accessed data like event lookups. These experiences collectively improved my confidence in Android development and sharpened my ability to build maintainable, user-friendly software.