Mitchell Fontaine

CS-499

03/24/2025

4-2 Milestone 3: Narrative

The event tracker application I chose is a mobile app designed to help users manage their events, reminders, and tasks efficiently. It was created a year ago during my course CS-360. It allows users to create, update, and delete events, set reminders with alarms, and organize their schedules. It was chosen as an artifact for my ePortfolio because it demonstrates my skills in Java and the use of Android Studio. The app features a clean and intuitive user interface built with ConstraintLayout, CardView, and RecyclerView, demonstrating my ability to create visually appealing and user-friendly designs. I implemented a notification system using AlarmManager and NotificationManager, showcasing my ability to integrate Android components. I created a SQLite database to store and manage event data as well as CRUD operations, demonstrating my proficiency in database design and management.

The artifact has improved with my optimizing data structures and algorithms. I replaced my linear search with a hash map to find reminders by ID and provide faster access when editing or deleting events. Sorting was made more efficient with Comparator which ensures a logical display order of earliest first. I used a priority queue for priority-based alarms so that alarms are processed in chronological order and not insertion order. An update to RecyclerView was then made in Custom.java so it avoids any unnecessary redraws.

The course outcomes I planned to meet from module one were met. I optimized data structures for event filtering and sorting using a hash map, priority queue, and Comparator sorting. Overall efficiency was improved with RecyclerView updates and reduced some redundancy. I then added synchronized blocks in RemindManager to increase security.

While I was creating this update to my artifact, I learned that some trade-offs are unavoidable. Priority queue improved scheduling but it also required some thread-safety measures. I also learned that the optimization of algorithms for Outcome 3 led to a better overall design which helped satisfy Outcome 4. One of the challenges I faced was thread safety when implementing the priority queue, but I was able to overcome it by synchronizing blocks. Another challenge I had was maintaining sync between the hash map and RecyclerView. This was overcome through the use of state management where I implemented a single source of truth.