I selected my Android inventory management application "Inventory Pro" as the artifact for my enhancement in the algorithms and data structure category. This application was originally developed during my CS-360 course and provided basic functionality for users to add, edit, and delete inventory items stored in a local SQLite database. The original implementation displayed inventory items in a simple, unsorted list based on the order they were retrieved from the database, which created significant usability issues as the inventory size grew. Users had no efficient way to locate specific items or analyze their inventory data, making the application impractical for real-world use cases where quick access to organized information is essential.

I chose this artifact because it provided an excellent opportunity to demonstrate my understanding of fundamental algorithmic principles and their practical application in solving real-world problems. The enhancement process involved implementing an efficient sorting system using the Merge Sort algorithm, which guaranteed O(n log n) performance regardless of dataset size. This implementation showcased my ability to analyze algorithmic complexity and select appropriate solutions that balance performance with scalability requirements. I created a comprehensive sorting framework that included multiple comparison criteria (item name, quantity, and creation date), bidirectional sorting options, and a generic utility class that could be extended for future enhancements. The solution required adding a new createdAt field to the database schema, implementing custom comparator classes following the Strategy design pattern, and creating an intuitive user interface with dropdown controls that allowed users to easily change sorting preferences.

I successfully met course outcomes related to designing and evaluating computing solutions using algorithmic principles and computer science best practices. The implementation demonstrated my ability to use well-founded techniques to deliver tangible value by transforming an inefficient, unsorted list into a highly organized and user-friendly interface. I learned valuable lessons about the importance of choosing appropriate algorithms for specific use cases, particularly how Merge Sort's stable sorting behavior and guaranteed performance made it superior to simpler alternatives like Bubble Sort or Selection Sort for this application. The process required more extensive database modifications than I initially realized, as I needed to implement proper schema migration to handle existing inventory items that lacked creation timestamps while ensuring backward compatibility.

One significant challenge I encountered involved refamiliarizing myself with the recursive nature of merge sort implementation and ensuring the algorithm worked correctly with custom comparator objects. I had to carefully debug the merging logic to handle edge cases where inventory items had null or empty values in their comparison fields. Another complexity came up during database schema modification, where I needed to design an upgrade strategy that would add the new createdAt column to existing installations without losing user data. This experience strengthened my understanding of both theoretical algorithmic concepts and practical software engineering considerations, particularly the importance of defensive programming when handling user data and the complexities involved in evolving database schemas in deployed applications.

Here are some screenshots from my enhancement:

