Andrew Torrez

CS-499-10454

October 15th, 2025

Inventory App

Artifact 2:

Software Design and Engineering

**Artifact Description**

The artifact I selected for this milestone is my **Inventory App**, originally created in May during my Mobile Architecture & Programming course. The app enables users to create accounts, log in, add and manage inventory items, and receive SMS alerts when stock levels drop below a chosen threshold. It incorporates user authentication, data persistence, and notification handling. For my CS-499 Capstone, I enhanced this project to strengthen its design, usability, and maintainability in line with professional software engineering standards.

**Justification for Inclusion**

I chose this artifact because it demonstrates my ability to design software that integrates **user interface, data persistence, and user experience** principles cohesively. The Inventory App represents my growth in writing code that is both functional and thoughtfully engineered for real-world use.

In its enhanced version, I focused on improving the **architecture, flexibility, and reliability** of the app. Originally, the app relied on hardcoded values—such as a default SMS alert threshold of 5—which limited user customization. To resolve this, I designed and implemented a **Prefs.java helper class** that uses Android’s SharedPreferences API. This design choice was made after considering trade-offs between simplicity and scalability. SharedPreferences is lightweight and ideal for key-value storage, unlike a SQLite database which would have been more complex and unnecessary for small configuration data. By prioritizing maintainability and speed, this solution balanced **performance efficiency** and **development clarity** without introducing unnecessary complexity.

Another major enhancement involved separating the login and registration logic into distinct activities (LoginActivity and CreateAccountActivity). This decision improved modularity and followed the **Single Responsibility Principle**, reducing coupling between unrelated processes. During development, I encountered validation bugs and activity registration crashes, which I resolved through structured debugging and proper AndroidManifest configuration—an important lesson in dependency management and system design awareness.

**Reflection on the Process**

Enhancing the Inventory App taught me to think beyond simply making an app “work.” I learned to weigh **design trade-offs** and make architecture decisions that improve scalability and user experience. For example, I optimized UI performance by restructuring layout hierarchies to reduce overdraw and prevent overlapping toolbars. These small but intentional refinements demonstrate awareness of how UI and resource efficiency directly impact usability and app stability.

I also strengthened the app’s **security and reliability** by adding input validation and sanitization checks in the registration and login workflows. This prevents crashes and helps guard against improper or unsafe input, aligning with a security-first development mindset. Though this project didn’t involve network communication, the principles of validating data at the UI level and preventing unintended behavior mirror secure software engineering practices in larger systems.

In addition, I improved the repository organization, introduced clearer code comments, and practiced **version control branching in GitHub** for each feature. This approach reflects professional collaboration habits—separating new features into branches, writing meaningful commit messages, and ensuring code clarity for future contributors. These are real-world practices that support teamwork and maintainability in collaborative environments.

**Planned Enhancements and Improvements**

The enhancements I completed included:

* Implementing a **dedicated Create Account screen** separate from login for clarity and modularity.
* Adding a **Prefs.java class** to persist user SMS settings (threshold and phone number) across sessions using SharedPreferences.
* Refining UI layouts to fix toolbar overlaps and improve **visual consistency**.
* Adding **“up” navigation buttons** across all activities to enhance navigation flow.
* Writing **comprehensive documentation** and **inline comments** to explain logic and assist maintainers.

These changes reflect both the **technical** and **aesthetic** aspects of software design, demonstrating that I can integrate usability, structure, and maintainability cohesively.

**Connection to Course Outcomes**

This artifact primarily demonstrates my mastery in **Software Engineering and Design**, but it also supports outcomes in **Security**, **Collaboration**, and **Professional Communication**.

* **Software Engineering and Design:** Refactoring activities, improving layout hierarchies, and implementing persistent settings demonstrate thoughtful architecture and clean code design.
* **Security Mindset:** Robust validation, persistence integrity, and controlled user input enhance reliability and data safety.
* **Collaboration and Professional Practices:** Version control branching, repository reorganization, and documentation mirror real-world teamwork processes.
* **Communication:** Clear naming, documentation, and consistent UI design reflect user-centered thinking and professional communication through code.

Through this enhancement, I shifted my mindset from “making code work” to designing **scalable, maintainable, and user-focused applications**, showing measurable growth as a software engineer.