Timothy McIntire

CS 499

Professor Hawk

June 21, 2025

**Professional Self-Assessment**

**Coursework, ePortfolio & Professional Growth**

Building this ePortfolio around a single codebase—the IT-145 “Grazioso Salvare” animal-rescue application—forced me to iterate exactly the way production teams do: refactor, optimize, then persist data securely. Although every visible artifact comes from that project, each enhancement leverages concepts first mastered across the wider CS curriculum:

| **Strength (learned in …)** | **How it appears in the capstone** |
| --- | --- |
| Modular OOP Design (*IT-145*, *CS-230*) | Abstract RescueAnimal superclass, kingdom subclasses, reusable intake logic |
| Algorithmic Efficiency (*CS-300*, *CS-305*) | Swapped ArrayList linear scans for HashMap O(1) look-ups |
| Secure Data Persistence (*CS-340*, *CS-470*) | JDBC + SQLite with parameterised queries and exception handling |

**Key Professional Competencies**

| **Domain** | **Demonstrated Ability** |
| --- | --- |
| **Collaborating in Team Environments** | Led Git workflows and peer reviews in *CS-250* & *CS-255*; comfortable with Agile boards and GitHub Projects. |
| **Communicating With Stakeholders** | Produced UML diagrams, sprint demos, and the capstone code-review video that explains technical trade-offs to non-developers. |
| **Data Structures & Algorithms** | Reduced search time from *O(n)* → *O(1)* using HashMap; earlier labs implemented recursion, BSTs, and dynamic programming. |
| **Software Engineering & Database** | Refactored for SOLID design; added a secure SQLite layer that persists data across sessions. |
| **Security Mindset** | Parameterized SQL, rigorous input validation, exception handling, and separation of concerns mitigate injection and logic attacks. |

**Career Aspirations & Values**

I’m transitioning from 10+ years as a Fire Medic to a Software/Full-Stack Developer role—ideally with companies that serve public-safety or civic domains (FirstDue, emergency-services SaaS providers, or similar mission-driven teams). I value:

* Solutions that improve community safety
* Cloud-native, scalable architectures (studying for AWS/Azure certs)
* Ethical, explainable AI (foundational exposure in *CS-370*)

**How the Three Enhancements Fit Together**

1. Software Design & Engineering – establishes a clean, extensible OOP foundation.
2. Algorithms & Data Structures – boosts performance without altering external behaviour.
3. Database Integration – adds secure, persistent storage, completing the data-flow.

By evolving *one* artifact through these stages, reviewers see a clear narrative of problem-solving: from architecture → efficiency → persistence—exactly the lifecycle of real-world software.