Timothy McIntire

CS 499

Professor Hawk

June 10, 2025 (revised June 20, 2025)

Milestone Four Narrative

1. Briefly describe the artifact. What is it? When was it created?

The artifact selected for this category is the same artifact used in the previous categories—my final project from IT 145, a Java-based console application from Grazioso Salvare. The program was originally created to manage intake, reservation, and listing of rescue animals for a fictional rescue organization. The initial version of this artifact stored all data in memory using ArrayLists. No persistent storage was implemented, meaning that all animal records were lost when the program closed.

2. Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components showcase your skills and abilities in software development? How was the artifact improved?

I selected this artifact for my ePortfolio because it provides an excellent opportunity to demonstrate how I can enhance an existing Java application with real-world database integration. Prior to this enhancement, the program relied entirely on runtime memory storage. Through this enhancement, I implemented a persistent database layer using SQLite and the JDBC API, adding the ability to store and update animal records across sessions.

To achieve this, I created a new class called DatabaseUtility.java. This class establishes the connection to an SQLite database file (animals.db) and provides reusable methods for:

- Inserting new animal records when animals are intaked.
- Updating animal records (specifically, the reserved status) when animals are reserved.

These enhancements show my ability to:

- Design a normalized table structure for storing object data.
- Integrate SQL-based CRUD operations with existing object-oriented Java code.
- Implement secure database access using parameterized queries to prevent SQL injection.
- Ensure database persistence while preserving the overall architecture and flow of the original artifact.

Beyond basic data persistence, I evaluated security considerations inherent to database access. Parameterized queries protect against SQL injection, while the single-responsibility DatabaseUtility class centralizes error handling and minimizes the surface area for misuse. In future iterations I would extend these safeguards with stricter input validation (age ↔ numeric check), connection-pooling with least-privilege credentials, and encrypted at-rest storage if sensitive data were introduced.

3. Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

Yes, I met the planned course outcomes for this enhancement:

- **Course Outcome 3:** I designed and implemented a solution that allows animal data to be stored persistently in a relational database and updated securely through the application.

- **Course Outcome 4:** I applied real-world software development tools (SQLite and JDBC) to improve the capabilities and maintainability of the artifact.
- **Course Outcome 5:** I applied secure programming practices by using parameterized SQL statements to protect against injection attacks and ensured reliable database interactions through structured exception handling.

No updates to my outcome-coverage plans are needed. This enhancement has fully met my expectations for the Databases category.

4. Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

Throughout this enhancement, I learned how to bridge the gap between an object-oriented system and a relational database. This required careful thinking about how object attributes (such as name, species, and reserved status) map to columns in an SQL table. I also learned to decouple database logic from the main application flow by placing all database access methods in a dedicated utility class.

One challenge I faced was resolving a "No suitable driver found" error when first testing the database connection. After investigating, I learned that the required SQLite JDBC driver must be explicitly added to the project classpath for the connection to work properly in Eclipse. Once configured, the connection was successfully established, and database operations performed as expected.

Another challenge was ensuring that my database enhancements preserved the minimal change requirement of this project. I wanted to avoid restructuring the entire Driver.java class. Instead, I introduced lightweight calls to DatabaseUtility.insertAnimal() and

DatabaseUtility.updateReservedStatus() inside existing methods. This kept the structure of the original code highly recognizable and traceable for review.

Finally, I tested the program by intaking and reserving animals across multiple runs to verify that persistence was working. Seeing Animal [MAM001] inserted into database. and Reservation status updated for animal [MAM001]. confirmed that the new database logic integrated cleanly with the application.

Rubric Criteria Summary

Rubric / Milestone Question	How It Was Met
Briefly describe the artifact	Provided a summary of the original IT 145 project, its purpose, and how it evolved; now integrated with a persistent SQLite database backend.
Justify inclusion	Selected due to its need for persistent data storage and the opportunity to demonstrate database integration, which is a critical real-world software skill.
Skills showcased	Demonstrated ability to design and implement a relational database, use JDBC for SQL operations, manage transactions, and ensure secure database access.
Course outcomes met	Confirmed alignment with CO3 and CO4 from Module One; no update to outcome plan required. CO5 (secure programming / data integrity) also supported through input validation and secure database interaction.
Reflection on process	Shared lessons learned about database design, JDBC integration, SQL query building, and maintaining program structure while adding persistent storage. Also discussed troubleshooting and testing process.
Technical enhancement submitted	All Java files with database enhancements (Driver.java, DatabaseUtility.java,

	RescueAnimal.java, etc.) have been updated and tested with a working SQLite database.
Narrative reflects milestone goals	Narrative written with clear explanation of database enhancement process, decisions made, and technical challenges overcome.
Narrative is ready for feedback	Narrative drafted with clarity, aligned with milestone prompts and rubric expectations. Ready for instructor feedback.