# **Milestone Four Narrative**

**CS 499 – Computer Science Capstone**

### **Artifact Description**

The artifact I selected is my **CS 330 Computational Graphics and Visualization project**, originally created in Fall 2024. The application uses C++ and OpenGL to render a 3D desk scene with multiple objects, textures, and lighting. While the original project demonstrated graphics programming skills, for this milestone I enhanced it by integrating a **database layer** to improve persistence, configurability, and monitoring.

### **Justification for Inclusion in ePortfolio**

I chose this artifact because it demonstrates how a graphics-focused program can be extended into a more **robust, professional-grade system** by adding database support. Originally, the application relied on hard-coded constants and in-memory state for settings like camera configuration, projection mode, and lighting. This design limited flexibility and reproducibility.

To enhance the artifact, I integrated an **SQLite database helper module**. The enhancements included:

* **Profiles Table**: Stores camera position, projection mode, and field of view for reproducible scene setups.
* **Lighting Table**: Stores light intensity and material parameters, allowing scene profiles to be loaded from a structured source rather than hard-coded values.
* **Telemetry Table**: Logs frame rates and frame times, providing insight into application performance over time.
* **Error Table**: Records failed resource loads (e.g., missing textures), creating an audit trail of runtime issues.

This improvement demonstrates my ability to connect C++ code with persistent data management, applying real-world practices that bridge system-level code and database integration.

### **Course Outcomes Coverage**

In my Module One plan, I intended to demonstrate database integration and security-minded design. I met these outcomes through the following enhancements:

* **Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices**: By embedding SQLite and writing a C++ helper class, I showed how to extend an application with a lightweight but powerful database backend.
* **Develop a security mindset**: The schema design validated and restricted inputs to known tables, reducing risks such as unvalidated configuration files. In addition, error logging ensures failures are recorded safely for review.
* **Professional-quality communication**: I documented the schema, added code comments, and provided a narrative explaining the purpose and results of these enhancements.

No updates to my outcome coverage plan were necessary, the database work aligned exactly with my plan.

### **Reflection on the Enhancement Process**

Through this enhancement, I learned how valuable persistence is even in applications that are not traditionally database-driven. Managing camera profiles and lighting configurations in a database provided a **more professional, configurable workflow** than hard-coded values. Logging telemetry data gave me practical experience with **instrumentation**, which is a critical part of modern software engineering.

One of the challenges was designing a schema that was simple but flexible. I wanted to avoid over-engineering but still provide enough structure for camera, lighting, and telemetry data. Another challenge was error handling, ensuring the database integration failed gracefully if the database file could not be created or opened.

Overall, this process taught me how database integration can make applications **more secure, reproducible, and user-friendly**. It was rewarding to see how this enhancement transformed the project into something closer to an industry-ready system.