**Software Design and Engineering**

**Artifact Title:** CS-330 OpenGL Final Project (3D Scene Renderer)

**Creation Date:** Originally developed in [April, 2025] for CS-330 Computational Graphics and Visualization  
**Enhancement Date:** June 2025

**Artifact Description**

This artifact is a C++ OpenGL application built during CS-330, which displays a textured 3D scene using modern graphics programming techniques. It incorporates GLFW for window and input management, GLEW for accessing modern OpenGL functionality, and GLM for mathematics.

**Justification for Inclusion**

I selected this artifact because it demonstrates my understanding of real-time rendering, modular architecture (with classes like SceneManager, ViewManager, and ShaderManager), and how to structure a professional graphics application. This artifact highlights my skills in:

* Object-oriented software design.
* OpenGL rendering pipeline.
* Performance optimization.
* Debugging and code enhancement.

**Enhancements Performed**

Based on feedback and a code review in Module One, I made the following key enhancements:

* **Added a real-time FPS counter** using std::chrono, which calculates and displays the current frame rate in the window title.
* **Refactored timing logic** to be more efficient and modular.
* **Improved code readability** by removing unused or outdated functions.
* **Addressed initialization warnings** related to uninitialized member variables.

These enhancements improved the artifact’s usability, performance monitoring, and code maintainability.

**Course Outcomes Addressed**

This enhancement contributes to the following program outcomes:

* **Software Engineering & Tools**: Used modern C++ with OpenGL, GLFW, GLEW, and GLM to build and improve a functional 3D application.
* **Communication**: The clean window title update with FPS info is a user-facing communication improvement.
* **Problem Solving**: Implemented an accurate FPS counter that required analyzing real-time frame timing and integrating it without disrupting the rendering loop.

**Reflection**

While enhancing this artifact, I deepened my understanding of frame-based timing and real-time rendering constraints. A key challenge was determining how to implement an FPS counter without affecting performance or OpenGL state — especially with limitations on available libraries like GLUT.

This experience reinforced the importance of debugging, precise measurement in graphics programming, and writing clean, maintainable code. I also recognized how small usability improvements, like showing FPS, greatly enhance the developer and user experience.

**Conclusion**

This artifact and its enhancements showcase my ability to build and improve well-structured software using modern development tools and C++ best practices. It represents a meaningful step toward professional competency in software design and engineering.