# CS 499 Milestone Three Narrative

Name: Caeli Firth

Date: July 27, 2025

## Artifact Overview

The artifact I selected for my algorithms and data structure enhancement is a 3D graphics engine prototype that was originally developed in a prior course. This artifact was created to allow user interaction within a 3D environment and focused heavily on camera movement, object rendering, and scene navigation. I selected this artifact because it presents a clear opportunity to improve algorithmic logic, introduce more advanced data structures, and enhance program stability and maintainability.

## Justification and Enhancements

This artifact showcases my skills in manipulating camera movement algorithms and structuring state logic for interactive 3D environments. The original implementation lacked collision detection, used inline input handling, and had no concept of physical velocity or acceleration. To improve it, I introduced axis-aligned bounding boxes for major scene objects, implemented velocity-based movement with acceleration smoothing, refactored the input logic into a dedicated subsystem, and revised conditional chains to eliminate side effects and ensure clean logic paths.

## Course Outcomes

These enhancements align with CS 499 Outcome 3, which emphasizes solving problems using algorithmic principles, and Outcome 4, which focuses on applying innovative computing techniques. The use of bounding box collision, modular input systems, and control flow restructuring demonstrates my ability to improve logic and structure at scale.

## Reflection and Challenges

Throughout this enhancement, I deepened my understanding of algorithm optimization and software design. One of the key lessons was how subtle flaws in input handling can create major usability issues. Debugging collision boundaries also proved to be challenging due to overlapping object states and rendering artifacts. I had to carefully test bounding logic and ensure all spatial dimensions matched the visual layout. This experience reinforced the value of modular design, unit testing, and abstraction in algorithm-heavy systems.