**Development Choices**

While writing the proposal, I recognized that quite a few primitive shapes would be needed to create the final scene, which led to me wanting to create code that could be easily reused to make those shapes. This resulted in the various primitive shape files such as Cube.cpp, though I soon noticed that they shared some data and logic that could be placed in the Shape.h file. One thing that I struggled with during the first few weeks was placing objects within the scene, as they didn’t quite appear where I expected them based on the transformations I used. I eventually tracked down the cause of this to the generateModel function in Shape.h, where the transformations were not in the correct order. Though I initially tried to implement the primitive shapes without referencing other code, I ran into a roadblock when attempting to create a torus since I couldn’t quite understand how to translate the math into working code. After textures and lighting were introduced, I decided to reference other code again to quickly discover the correct UV coordinates and normals, especially for the curved shapes. However, using those normals for the cylinder produced an effect where two quarters of the cylinder sides were facing out, and the other two were facing in, which I eventually solved by switching the sine and cosine normals.

**Navigation**

The camera controls follow the common model for navigating a 3D scene on a desktop computer, where the mouse changes the camera facing, and the keyboard changes the camera position based on the facing. Moving the mouse horizontally pans the camera, and moving it vertically tilts the camera, though it cannot be tilted past straight up or straight down (ProcessMouseMovement in camera.h & UMouseCallback in Source.cpp). The mouse scroll wheel can be used to increase or decrease how quickly the camera position is changed (ProcessMouseScroll in camera.h & UScrollCallback in Source.cpp). The keyboard inputs include the ‘W’ key to move the camera forward, the ‘S’ key to move the camera backwards, the ‘A’ key to move the camera to the left, the ‘D’ key to move the camera to the right, the ‘Q’ key to move the camera up and the ‘E’ key to move the camera down (ProcessKeyboard in camera.h & UProcessInput in Source.cpp). As mentioned before, these directions are all relative to the current facing of the camera. Additionally, the speed at which the mouse changes the camera facing can be increased with the ‘V’ key or decreased with the ‘C’ key (UpdateSensitivity in camera.h & UProcessInput in Source.cpp). Finally, the view can be changed between perspective and orthogonal by pressing the ‘P’ key (UProcessInput & URender in Source.cpp).

**Custom Functions**

The function UpdateSensitivity in camera.h provides an input anonymous way to adjust the sensitivity, making it easy to adjust to different keys or even mouse button input. The function parseVertex in ShapeTools.cpp provides a clean way to insert vec3 data for a mesh into an array, which could be useful when creating primitive shapes. The remaining custom functions in Shape.h and each primitive shape source file are limited in reusability, as many are dependent on the shader I have used. If the intent was to create another simple 3D scene using the same shader, then they could be easily reused for that purpose since they provide a quick way to create primitive shape meshes and store several transformations for each one. The new developer would only need to provide transformations and textures, then they could be immediately rendered in the new scene. If a different shader was required, then the basic functionality of creating shape meshes and storing transformations could be useful as well but would require some modification to support the new shader. Although I could have encapsulated the code in Source.cpp that creates the primitive shapes that the objects are made of, it did not seem necessary based on the scope of this project. These objects are unlikely to be reused in this project or other projects, and the comments seem to provide enough organization.