In developing my 3d scene I made a lot of choices and changes to the code to render the scene that I picked. The scene I chose was a desk image with coffee, keyboard, and a mouse in the scene. There were other things in the scene, but I focused mainly on those objects.

I chose that scene because it is something that I see every day, and I wanted to recreate that.

Starting off each of the objects in the scene I had to model them after 3D shapes such as planes, cylinders, toures, and boxes. I selected these shapes because I thought that when combined they could create the shapes that I needed to render my scene as close to the picture as I could.

I used the plane for my desk, the cylinder, and torus for the coffee mug, the keyboard used boxes, and I used some for loops to render the keyboards keys, and the mouse used tapered cylinders along with a box. These shapes let me experiment with different geometric forms and their transformations.

By combining shapes such as the torus and the cylinder for my coffee mug I was able to practice fundamental 3D modeling techniques, such as scaling, rotating, and translating these shapes to make more complex objects.

The overall functionality of the scene was implemented using shaders, materials, and transformations. For example I used the CreateGLTExture function to apply textures to various objects. This gave them a more realistic appearance. I did a wood texture for the desk, and did my best to find a white glossy texture for the coffee mug, and a black texture for the computer monitor.

I went further as well and utilized different surface properties, such as shiny, white, and dark, to ensure that the visual scene had more realistic qualities. This works with the lighting, and complements the lighting of the scene.

For my lighting I tried to set up sunlight, as well as a direct overhead light, to resemble light coming out of a window, as well as a ceiling light shining down onto the desk. I struggled with the lighting, and am not sure if I implemented it into its entirety. But never the less I got the lighting to work in some way. I think overall the scene looks good with the lighting.

Navigating my 3D scene was made possible by using a keyboard and mouse controls that allow for you to move back and forth, up and down, and control the camera angle with the mouse. The controls also allow for the switching between a 2d scene, and a 3d scene. I used the controls WASD for movement, QR for up and down, and O and P for the scene switching. The navigation resembled playing a video game on the computer.

To maintain modularity of the code I used various functions. Functions such as DefineObjectMaterials() , SetupSceneLights(), LoadSceneTextures(), loading all of the shapes and objects, as well as SetShaderMaterial() and

SetShaderTexture(). This gave me more efficient code, it made the code look better, and also made the data flow better. The functions of those functions were pretty self explanatory with the names. The DefineObjectMaterials, defined the object materials and let me create tags that could be used later in the code to give that object the specific material that I created. In this case I used it for shiny, white, and dark materials. The SetupSceneLights() helps with creating the lighting within the scene. My LoadSceneTextures, are loaded after all of my CreateGLTexture functions are binded by BindGLTextures. Within the CreateGLTexture functions I created more tags that are used later. I then called the LoadSceneTextures to bring in my images that I found online to use for the textures. They would then be set by the SetShaderTexture function, where I used the tags for the textures that I created. This is what gave my objects the specific textures that I have in my scene.

We had a lot of other functions that were already coded in the project, but some of them like the SetTransformations function takes parameters for scaling, rotation, and translation, allowing it to be reused for every object in the scene. It makes the code more reusable and eliminate the need to write repetitive code. We could also look at our LoadSceneTextures function again, which would load all the textures used in the scene, and is another perfect example of a modular function.

These modular functions help organize the codebase by grouping similar operations together, thereby reducing the redundancy and making the program easier to understand, as well as maintain.

Overall the project helped me practice all of the principles that we have learned in this course from rendering objects, applying textures, creating movement in the scene, creating lighting, and applying materials. I will also say that creating this scene was a lot of fun!