In creating my 3D scene throughout CS330 leading into the final submission, I wanted to make an environment that felt familiar, this is why I chose to build a small workspace setup consisting of a laptop, mug, desk lamp, and Rubik’s cube sitting on a marble-textured table. These objects were meant to reflect items often found in a study or desktop environment, and they provided enough variety in shape and material to fully explore OpenGL’s rendering capabilities. Each object was created by combining and transforming basic mesh shapes like boxes, cylinders, and planes. For example, the desk lamp is built with a tapered cylinder for the arm and head, and a low, wide cylinder for the base. Another example is that the mug uses a tapered cylinder for the base, a torus for the handle, and a torus for the rim.

To help users explore the scene, I added full camera movement and switching between projections. The scene starts with a perspective view, WASD have been mapped as the forward, back, left, right movement and the Q & E keys make the camera move up and down. Mouse movement rotates the camera direction, and scrolling adjusts the speed at which the camera moves. For projection control, I used the P key to switch to a perspective view and the O key to swap to orthographic mode. I also added extra orthographic views for front, top (T key), and left side (L key) orientations.

One of the main strengths of the project is modularity. Being able to use re-usable functions to help keep the code clean and manageable. For example, the SetTransformations() function simplifies how position, rotation, and scaling are applied to objects in the scene. By calling this function with different parameters, I could place each object precisely without repeating the transformation logic. Another helpful function is SetShaderMaterial(), which applies predefined lighting properties to the shape or object it is being applied to. These modular functions make the code easier to maintain and extend in future versions.

Lighting plays a very important role in making the scene visually appealing and realistic. I used three types of light sources: a directional light for overall ambient fill, a point light to simulate the desk lamp’s bulb, and a spotlight to represent glow coming from the laptop screen. According to the official OpenGL documentation, using different light types in combination helps create realistic shading, especially when each light contributes ambient, diffuse, and specular components to the scene (OpenGL, n.d.). The point light near the Rubik’s cube is particularly effective at casting soft, warm light and creating highlights on nearby objects.

Texture mapping was also important. I used high-resolution textures to make sure that materials looked clean and realistic. The laptop screen texture, for example, is a Windows XP background. The Rubik’s cube uses different textures for each face to demonstrate multi-sided mapping, a technique supported in OpenGL using glBindTexture() and proper UV coordinates (LearnOpenGL, n.d.). I made sure to avoid texture stretching or distortion by adjusting UV scaling appropriately for each shape.

Overall, I’m proud of how the scene turned out. It includes accurate models, camera controls, detailed lighting, and well-applied textures. This project helped me better understand how OpenGL handles 3D rendering and how to structure code for better readability and reuse. I learned how small lighting changes can impact realism, how to manage complex objects through modular design, and how to offer a smooth, interactive experience for users navigating the scene.

**References**

OpenGL. (n.d.). *OpenGL 4 Reference Pages*. Retrieved from

https://registry.khronos.org/OpenGL-Refpages/gl4/

LearnOpenGL. (n.d.). *Lighting, Textures, and Camera*. Retrieved from

<https://learnopengl.com/>