Design Decisions

**Justify development choices for your 3D scene.**

For my 3D scene I wanted to create a believable and familiar environment centered around a wooden shelf and a few everyday objects. I chose to model items like a decorative tree, an Xbox Series X, a picture frame, and a bowl because they are recognizable, simple to model, and together they form a small living room like set up. These objects helped me meet the requirement to build complex shapes by combining simpler components, such as scaled boxes, cylinders, and spheres.

I was also intentional about using transformations to scale and position each item inside the shelf cubbies making sure their placement looks natural and well balanced. For example, the xbox was made using a tall black box mesh, while the tree was made from a small cylinder for the pot and a textured sphere for the leaves on top. I used textures like wood grain and planks to create a consistent visual theme and applied appropriate material settings (Ambient, diffuse, specular) to reflect how these objects might interact with light.

For lighting I followed the requirements to include at least two light sources. I used one white overhead directional light and one purple-tinted side light.. These choices helped me achieve both even lighting and added some color to my scene.

**Explain how a user can navigate your 3D scene.**

The user can navigate the 3D scene using both the keyboard and the mouse. The keyboard allows movement in six directions. Pressing W moves the camera forward, S moves it backward, A moves it left, D moves it right, Q moves it down, and E moves it up. These inputs are handled through the ProcessKeyboard() method and update the camera’s position based on frame time. The mouse controls the direction of the camera view. As the user moves the mouse, the Mouse\_Position\_Callback() function updates the view using ProcessMouseMovement(), allowing the user to look around the scene. Additionally, the scroll wheel on the mouse adjusts the camera’s movement speed by passing input to the ProcessMouseScroll() method. This lets users zoom through the scene faster or slower depending on preference. Users can also switch between a perspective and orthographic view by pressing the P and O keys which can help when viewing objects.

**Explain the custom functions in your program that you are using to make your code more modular and organized.**

I created several custom functions to break the scene into manageable pieces. For example, RenderShelf() handles the shelfs structure and divides it into parts like the top panel, bottom panel, dividers, and middle shelf, RenderXbox() is a function just for drawing the Xbox model, and RenderTree() draws the little tree. These functions let me reuse transformation code while keeping everything organized by object.

By grouping the code this way, I can easily go back and update one part of the scene without affecting others. It also made my main rendering function cleaner, since each object can be drawn with a single function call. These functions are reusable because they accept transform parameters like position and scale, which makes it easy to add multiple instances if needed.