1. Justify development choices for your 3D scene. Think about why you chose your selected objects. Also consider how you were able to program for the required functionality.

For my 3D scene, I recreated the spice rack, pen, ChapStick, masking tape, Solo cup, and book from my original image. Starting with the simplest, the book was included because it was a simple box. To make it look more identifiable, I found the cover of the book online and applied it as a texture of the book. Because the shape was already square, the texture applied neatly to the front of the book. An unintentional effect that occurred was the texture stretching on the skinny faces of the box, which in turn looked similar to pages compressed between the covers of the book. I turned the book slightly for aesthetics. Next, is the ChapStick. I simply made it a white cylinder. Truthfully, I had a large number of difficulties with this object because of the rotation, leading me to sit at my desk with the same ChapStick, envisioning how to orient the shape correctly. Originally, I planned to use a second cylinder to make the cap of the ChapStick but after doing so, I did not feel like it added much to the scene. Next is the masking tape. I started off by creating a blue cylinder to create the general shape. Then I created another cylinder in the middle of the blue cylinder to represent the cardboard liner. I made the inner cylinder slightly taller than the blue cylinder to avoid clipping issues and applied a cardboard-like texture over it. The Solo cup was another object I created in my scene. The cup is simply a tapered cylinder that I applied a red texture to and made it slightly shiny so I can replicate the reflectiveness of the red plastic. Next is the black pen. To start, I used a long, thin cylinder to create the pen body, followed by a cone to create the tip. Looking at the rendered scene, I felt that the pen looked more like a pencil, so I decided to add an elongated sphere to recreate the clicker at the top of a pen. Lastly, there is the three-tier spice rack. This is easily the most complex object in the scene. I designed it from the bottom up, as it felt the most logical while ensuring everything lined up perfectly. Each tier of the spice rack is represented with cylinders, with each sequential cylinder shrinking in width. For the pole running through each tier, I could have just used a cylinder running through the whole structure. However, looking at the spice rack, I noticed the pole was not straight but was skinnier in between each tier, creating an almost hourglass shape. To replicate this, I used two cones, with one inverted and intersecting each other. On the top tier, I used a cone topped with a cylinder to create the top piece of the entire structure. Lastly, I applied two different wood grain textures. I opted for using two textures because I disliked how the texture used on the cylinders looked on the cones. Overall, I feel this also adds more depth to the structure.

1. Explain how a user can navigate your 3D scene. Explain how you set up to control the virtual camera for your 3D scene using different input devices.

Users can navigate my 3D scene using WASD controls to move the camera. To rotate the camera, the user can use their mouse. Additionally, users can use the Q and E keys to shift the camera up and down. If a user would like to adjust the speed of the camera for finer camera controls, the user can use their mouse scroll wheel to increase and decrease the camera movement speed.

1. Explain the custom functions in your program that you are using to make your code more modular and organized. Ask yourself, what does the function you developed do and how is it reusable?

I created custom functions for rendering all the shapes in my scene. These functions allow me to render any basic 3D shape by simply passing transformation parameters and rendering settings, instead of repeating the same OpenGL setup code for every object. These functions encapsulate the logic for setting up object transformations (SetTransformations()), applying materials and textures, and calling the appropriate mesh draw command. For example, instead of writing several lines to draw a wooden cylinder every time, I can now just call:

This modular approach improves readability and allows me to reuse the same rendering logic for different objects (like a spice rack tier, a solo cup, or a roll of masking tape) just by changing the parameters. It also makes future updates easier, for example, if I want to change how all cones are drawn, I only need to update RenderCone() once instead of modifying every instance across the codebase.