For my final project for the CS 330 course at SNHU, I selected the image of a kitchen counter with some objects on it.



This image was selected because the objects in it were spread apart and required the use of multiple 3D shapes to render. As none of the objects overlapped with each other, making them separately was a bit easier as joining together different 3D shapes did not get in the way of one another. Also, the objects in the image were not too complex in terms of their shape, so it was possible to render each of them without needing too many shapes, textures or shaders, which kept the load times and processing time low, making the scene render smoothly. My approach to programming the scene started with a pane that would represent the counter, and I chose the goblet as the first object to render as it was the right most object in the image. As I followed the milestones, I applied camera controls, textures, shaders and lighting to the scene. For the final scene, I created all the remaining objects first and ensured their position was similar to the real image. Once their 3D representation was complete, I applied textures and shaders to each object one at a time and tweaked some texture scaling and shaders for consistency.

For navigation, by default the scene allowed for controlling the camera via moving the cursor around. I have disabled the use of cursor and made it so the entire mouse acts as the camera. This allows for more fine-tuned camera movement and is more intuitive for users. The W and S keys in the keyboard allow for zooming or moving forward the scene, and the latter key zooms out or walks backwards. Keys A and D allow the user to strafe left and right respectively. The Q and E keys are used to move the height of the camera angle, allowing the user to pitch the camera up and down. Finally, the view can be switched between orthographic (2D) and perspective (3D) by pressing the O and P keys on the keyboard. The camera speed can also be manipulated by scrolling up or down on the mouse based on user needs.

Aside from good coding practices such as inline comments and local scope variables, I have made an individual function for each object. My prior approach was to have all the 3D object codes be in the function responsible for rendering the scene. But the amount of 3D shapes that was required for the scene, this monolithic approach would have made the code very hard to read and make changes to. Instead, I separated each object into their own functions, so if I wanted to make changes to an object, I would only need to tweak the function that was responsible for it. This allows more modularity across the application and helps in testing and debugging the application. Aside from the shapes rendering, other features such as camera controls, texture loading and lighting are also in their individual functions, so their function code blocks can easily be copied to a different project, and they would still work as expected as they all have local scoped variables and does not rely on any globally declared variables to work.