My scene is set on a marble countertop with various items sitting on top. These items include a bottle, wooden spatula, towel/potholder, and salt and pepper shakers. First, the plane was developed by declaring the 6 vertices which represent the two triangles that create the plane. I applied the texture to make it look like a marble countertop. I created the bottle using a cube with a cylinder connected to the top. I elongated both on the y-axis to shape them appropriately. I then applied textures to the bottle to show that the cube was glass, and the cylinder was covered in plastic as if the bottle wasn’t opened yet. For the wooden spatula I used two cubes together. The cube that represented the handle was scaled to make the y-axis much larger than the x and z axes which resulted in an elongated handle. The other cube was made to be flat by scaling the z-axis to be much smaller than the y and x axes, then rotated to make it lay correctly. Then the wood grain texture was applied to both. For the towel/potholder, I used another cube and scaled the y-axis to be small, making the cube very flat. I then translated it to make it lay flush with the marble countertop plane and added the towel texture. The salt and pepper shakers are both represented using cylinders. The texture was tricky for these because I could not find a texture that would work that looked like an actual salt and pepper shaker, so instead I went with white and black textures which has light grain to them to represent the salt and pepper.

It is very simple for a user to navigate the program using a keyboard and mouse. The user can look around using the mouse cursor to change the orientation of the camera. Using the WASD keys, the user can move forward, backward, left, and right. Then, if the user would like to move up and down, they can use the Q key to move down, and the E key to move up. Using the mouse scroll, the user may change the speed in which the camera moves. If they want to reset the speed back to default, they can press the spacebar. To change between the orthographic and perspective view, the user can press O for orthographic, and P for perspective.

This program uses a few very important functions to allow for the display to work properly. First, I had to create the UCreateTexturedMesh function. This function was used to create the vertices for each shape. I created vertices to display a plane, cube, and cylinder which I then used to create each of my objects. I was able to create the mesh and activate the VAO and VBO for each object. I also created the vertex attribute pointers for each object. This function is very reusable since I have basic vertices for three shapes that would just need to be scaled, rotated, or translated to create the object that I want. I can also use the following code as a guide on how to create a mesh for an object. The next function is the URender function which allows me to render each object. In this function, I scaled, rotated, and translated each object, activated the shader program, bound the vertex array, bound the textures, and then drew each object. I wouldn’t be able to reuse this entire function, but I can use this function as a guide in the future if I need to create another project similar to this one. Another vital function that I used was the UProcessInput function. This allowed the system to process all input from the user. In combination with the camera.h file, I was able to tell the system what to do when the user produces a certain input. This function will be reusable in future OpenGL projects as long as the camera.h file is imported as well.