To begin this project, I started by organizing several objects on my desk in an interesting way. Afterwards, I took several pictures of the scene from multiple angles. I used these photographs to study the different perspectives of my future render so that I might better understand how to recreate the scene in code. Ultimately, I decided to include the following objects in my scene: a lamp, a notebook, a tablet, a cup, and a plane for which to contain all of them. I selected these because they would allow me to use a total of seven different objects, which was more than sufficient to meet the project requirements, but I believed that making use of even more shapes would create a more interesting scene. Specifically, the shapes that I ended up using included the plane, cylinder, tapered cylinder, torus, sphere, prism, and box. The lamp object was the most difficult object in my scene and made use of five of those shapes on its own.

After I determined the shapes I wanted to use to represent my objects, I began working on introducing them into the application. I made use of some existing meshes by Professor Battersby in order to focus on the overall render and not the low-level mathematical details of shape vertices. For each object, I would first render a plain shape into the environment. Once I was satisfied with the position, scale, and angle of the shape, I would apply a texture and size it appropriately. Lastly, I would introduce lighting effects in order to fully bring the object to life.

I wanted controlling my scene to be as intuitive and straightforward as possible. To achieve this, I used the standard WASD control scheme to allow the user to move forward, backwards, left, and right. I also wanted the user to be able to move vertically up and down, so I decided that Q should be used for the upwards movement and E should be used for the downwards movement. The 3D render would not be nearly as interesting without a free camera, so I used mouse motion events to change the direction of the camera. Additionally, I used the mouse’s scroll wheel to affect the speed at which the user would navigate through the scene. While working on the render, however, I decided that I wanted a way to freely rotate the angle of the camera without necessarily changing the current position. To achieve this, I added functionality to allow the Z key to change only the current angle of the camera, which gave the impression of the camera spinning around its axis. Finally, the O key can be used to change from a perspective mode to an orthographic view, which instead displays the objects as if they were entirely flat. Pressing P afterwards will take the user back to projective mode with the same camera position they were in before.

As I worked through the project, I wanted to make sure that my code would stay organized so that my instructor would be able to understand my logic. I immediately knew that I would have a lot of rendering code for my objects, so I created render.h and render.cpp to contain all of the rendering logic for my objects. In this way, I would still be able to have top-level render function that I could call in main that would in return call individual render functions corresponding to all of my objects in the scene. Likewise, those objects were further broken down to contain individual render functions for all of their shape components. This was helpful, because I was then able to simply modify the position of the object, which would then apply the same mathematical operations to all of the contained shapes. In other words, I could simply supply a position for the “object” and all the shapes would seamlessly adjust their position around that origin, eliminating the need to move multiple shapes independently.

Another way I utilized code organization techniques was by creating functions such as loadTextures() and bindTextures(). As their name suggests, these functions were designed to load all of the textures from memory, then subsequently bind them to individual texture bits. By calling functions only once outside of the main loop, I made sure that my program would not be slowed down by any unnecessary excess loading or binding calls.