My work throughout this project has been a journey from technical learning to creative expression, a gradual awakening of how geometry, light, and perspective combine to form meaning in a digital world. I began with simple forms: planes, cylinders, and spheres. By arranging them into a room and a bookshelf, I tried to give structure to space, and by shaping a crescent moon lamp, I aimed to give that space emotion. My goal was not only to complete a scene but to make it *feel* inhabited, warm, and quietly alive.

The process, however, was far from effortless. One of my most persistent struggles was rendering the books on the shelves in a visible and believable way. At first, they were invisible, buried in the scene like shadows behind the geometry. Their colors blended into the background, or they vanished entirely from the camera’s forward view. I adjusted the scale and position countless times, convinced that the problem lay in the transformation values. But the real issue was the camera itself. I was looking at the shelf straight on, flattening every object into a single plane. When I finally shifted the camera to a side view, rotating it just enough to give the scene depth, the books emerged clearly. That small rotation changed everything. What had been a wall of color became a layered, three-dimensional structure, and suddenly, the bookshelf looked real.

That moment of discovery reshaped the way I thought about perspective and visibility in 3D space. It wasn’t only about coding or adjusting coordinates, it was about seeing. The lesson echoed beyond the scene: sometimes, progress is not about adding more but about changing how you look. In graphics, as in creativity, the right perspective reveals what was always there, waiting to be seen.

My second major challenge was adding texture. I wanted the shelves to carry the grain of wood and the books to display the warmth of printed covers. I successfully loaded texture files using **stb\_image**, but when I implemented texture creation in both *main.cpp* and *SceneManager.cpp*, Visual Studio reported multiple symbol definitions for the stb functions. The linker errors (*stbi\_load already defined*, *stbi\_image\_free already defined*) stopped the build completely. It was a frustrating experience that tested my patience and my understanding of how C++ manages compilation units.

I learned the root cause: the directive #define STB\_IMAGE\_IMPLEMENTATION should appear in only one file. Placing it in multiple source files caused every texture-related function to be compiled twice. Although I did not fully integrate both textures this time, the failure taught me the deeper architecture of how libraries, shaders, and OpenGL communicate. I now know that the *SceneManager* should be the single point of responsibility for texture loading and binding, while other components simply reference those textures by name.

When I revisit this project, I plan to rebuild texture mapping from the ground up. My next step will be to load just one texture, a simple test image, verify it displays correctly, and then apply additional textures to the bookshelf and books once the shader bindings are stable. I’ll confirm that each texture is assigned its own tag and bound to the correct sampler slot, allowing for independent control within the shader. This careful, step-by-step approach will ensure that the wood texture, the book covers, and eventually the floor and walls each carry their own visual identity.

Despite the setbacks, I am proud of what I’ve learned. I’ve come to understand not just *how* to draw in 3D, but how to *see* in 3D, how light, texture, and perspective combined to create a sense of realism. The scene I built may still be simple, but it’s grounded in patience and discovery. Every challenge, from invisible books to broken texture loaders, was a lesson in persistence and perspective. I now look at this project not as a collection of code, but as a practice in observation, an experiment in turning space into story.

References

Khronos Group. (2025). *OpenGL 4.6 specification.* <https://www.khronos.org/opengl/>  
Noth, S. (n.d.). *stb\_image.h documentation.* Retrieved from <https://github.com/nothings/stb>