**7-1 Final Project Submission**

## Development Choices

For my final project, I chose to recreate a 3D yoga and fitness scene that includes a mat, kettlebell, water bottle, and yoga block placed on a textured floor with a background wall. This setup allowed me to demonstrate texturing, lighting, and transformation concepts while keeping polygon counts efficient and under 1,000 triangles per object. Each model was built from primitive shapes: the kettlebell combines a sphere and torus, the yoga mat uses planes and cylinders, and the water bottle uses a tapered cylinder and cone for the cap.

All textures were sourced from royalty-free 1024×1024 images, including materials like wood, metal, marble, mosaic, and brick. I used OpenGL’s mipmapping and GL\_LINEAR filtering for smooth texture rendering. Lighting in the scene includes both a warm directional light and a cool blue point light to satisfy the Phong illumination model with ambient, diffuse, and specular components. Each material was given specific shininess values for realistic highlights across different surfaces.

## Navigation and Camera Controls

The scene supports six-axis camera navigation. The user can move forward, backward, left, and right using the W, A, S, and D keys, and move vertically using Q and E. The mouse controls pitch and yaw to change orientation, allowing smooth first-person exploration of the environment. The scroll wheel adjusts movement speed dynamically, giving users finer or faster control. A keyboard toggle allows switching between orthographic and perspective views without altering the camera’s position, making it easier to analyze spatial relationships or view the scene with depth perception.

## Custom Functions and Modularity

To maintain modular and efficient code, the program architecture is divided into distinct classes: ‘SceneManager’ for scene rendering, ‘ShaderManager’ for handling GLSL uniform communication, and ‘ShapeMeshes’ for managing reusable mesh geometries. Helper methods such as `EnableLighting()`, `SetMaterial()`, and `SetShaderTexture()` simplify shader and lighting setup across multiple objects. This modular approach makes the codebase easy to maintain, extend, and reuse for future projects.

These design choices follow software engineering best practices, emphasizing readability, modularity, and functional logic. The final 3D scene effectively demonstrates texturing, realistic lighting, accurate object placement, and smooth navigation controls consistent with the CS-330 Final Project Rubric.