**CS - 330**

**Comp Graphic and Visualization**

**Southern New Hampshire University**

**Andrea C. Sherry**

**Reflection**

1. **Justify development choices for your 3D scene.**My development choices for the 3D scene were led by a desire to create a realistic environment. I like that Virtual Studio 2022 has a strong set of tools for 3D modeling, texturing, and animation, all within a single integrated platform. Achieving a high level of realism was vital, so I used high-resolution textures and physically based rendering materials to increase the visual experience. Performance was also a critical consideration thus, I improved models to reduce polygon count where possible and employed efficient lighting techniques to keep smooth operation without compromising visual quality. Interactivity played a significant role in making the scene engaging. I incorporated dynamic camera movements to help an interactive experience. Additionally, I organized the project in a modular way, using reusable components and scripts to ensure easy updates and scalability. This modularity allows for the uniform addition of new elements or modifications to existing ones, minimizing the need for extensive rework and ensuring a flexible and maintainable codebase.
2. **Explain how a user can navigate your 3D scene.**Users can navigate the 3D scene through intuitive controls that mimic real-world movement and interactions. The keyboard allows for movement using the WASD keys for forward, backward, and lateral movement, while the mouse is used for looking around. The camera is designed to follow the user’s inputs smoothly, with adjustable speed.
3. **Explain the custom functions in your program that you are using to make your code more modular and organized.**To ensure the codebase is modular and organized, I implemented several custom functions and scripts. The `InitializeScene()` function sets up the initial state of the scene by loading all necessary assets, setting the initial camera position, and configuring the lighting. The `LoadTextures(material, texturePath)` function is reusable for loading textures onto materials, allowing for easy updates without changing the core logic. Additional functions like `HandleUserInput()` process all user inputs (keyboard and mouse) and trigger the appropriate responses in the scene. `UpdateCameraPosition()` calculates and updates the camera’s position and orientation based on user inputs and predefined paths, ensuring smooth and natural camera movements. These functions help keep the code clean, readable, and maintainable. By breaking down the code into smaller, reusable components, it becomes easier to debug and extend the project's functionality.