For this project, I wanted to take an effect I had realized before from the flat plane into a higher dimension for better viewing value, and explore a new effect, then add basic tweaks to both. Therefore, I chose "Game Of Life" and "Perlin Noise Based Terrain Generation" as my goals, and set the target engine to be Unity, and after a period of development and tweaking, I have roughly achieved my goals. After a period of development and adjustment, I have roughly realized my goal.

I first used the Perlin Noise process to build a system for Unity that adjusts the mesh using code. I used Perlin Noise to generate a noise map, then I generated different mesh heights for it based on its grayscale, and I set the color for the mesh in the different height ranges (so that the mesh would look like the underlying terrain). At the same time, I set up real-time changes to its offsets so that the terrain is constantly moving and updating. I also put the control options for the different variables into the UI, so that some of the properties can be adjusted in real time, intuitively, while the program is running. In practice, I accidentally realized that setting the Noise Scale variable to an extremely large value resulted in a pixel cube style undulation effect, which looked pretty cool, so I added an additional Noise Scale Multiplier to allow users to reproduce this effect.

Then there's Game Of Life, whose logic I converted to a C# script and then tweaked using the mesh generation script mentioned above, changing the 2D float array to a 2D bool array and tweaking the rest of the logic to convert the original 2D pixel effect into 3D space. At the same time, I also added properties like meshHeight for tweaking. After the cells in the mesh have stopped transforming, use the reset button to restart the simulation in a different seed.

In the user interface, in addition to the respective property adjustments I mentioned earlier, I also provided a toggle function for both functions so that the user can switch to Perlin Noise or Game Of Life at any time, and since both simulations are performed on the same mesh, the mesh transformations caused by resizing the mesh are preserved even when switching between simulations. The player can drag and rotate this grid to view the effect from different angles for a more intuitive view.

With this project, I not only reviewed my past knowledge, but also combined it with the Unity engine to create three-dimensional effects and adjustments, and viewing capabilities. As a result, I became more experienced in implementing visual effects and utilizing the Unity engine, which will benefit my future learning and project development.