I chose to use Unity for my game engine since I had already been using it for my game from a tutorial I had found. I used that tutorial to build upon and have completely revamped it from a 2-players-on-1-computer game, to a PVE game where the player fights against AI enemies. the tutorial gave me the prefabs and map and allowed me to do the programming aspect of the game development since that is what I want to make my career. It gave me the opportunity to take something that already exists and make it more like what I want it to be.

Many things worked well for me, especially being able to manipulate the code because the code itself could be taken apart and put back together. The game engine and the link between Visual Studio, which I used for coding, and Unity allowed me to seamlessly make the game go from a PVP aspect to a PVE aspect. Another very helpful aspect of Unity was its ability to add and remove components of objects, which allowed me to revamp the camera system and be able to experiment with mouse tracking. Unity made the greatest impact on being able to take apart the whole and then learn how each piece worked, which allowed me to better understand the whole project and the way the system itself worked. I did struggle with allowing for a mouse following part of the tank prefab, but this was due to the limitations of the prefab system and the way the models were spawned in the world and how they had certain access permissions. Another issue I encountered was dealing with spawning multiple tanks, this was a simple solution that I found when exploring the GameManager item within the world and allowed me to put a near infinite amount of enemies in and can choose where they spawn through some code. Many other difficulties were small and required some rethinking and the idea of how to fix it came at random times so there was no concrete method used to solving these other than the skills I have learned from programming.

The outline in the mini proposal was more refined and complete than the final product. Most of the changes from the mini-proposal was limitations with the way the tutorial final product was set up and the limitations of Unity's code to interface relations. Working with the Unity allowed me to see how the starting idea needs to be broad, but specific, and allow for the programming and in-engine work to be done in order to allow for a simpler and less mentally draining way of making the game. Taking the abilities of what can be done can prevent failed promises, work that won't be used, or features that won't be available. I definitely learned about the difficulty of game programming, but also how rewarding it is to see the product work as you wanted it to, or how you changed it to work better. These lessons have helped me really want to continue my experience making games, especially from a programming side, where I can collaborate with other programmers to really make an engine reach its limit.