# CSCI 4448 Project 6 - The Survival Game

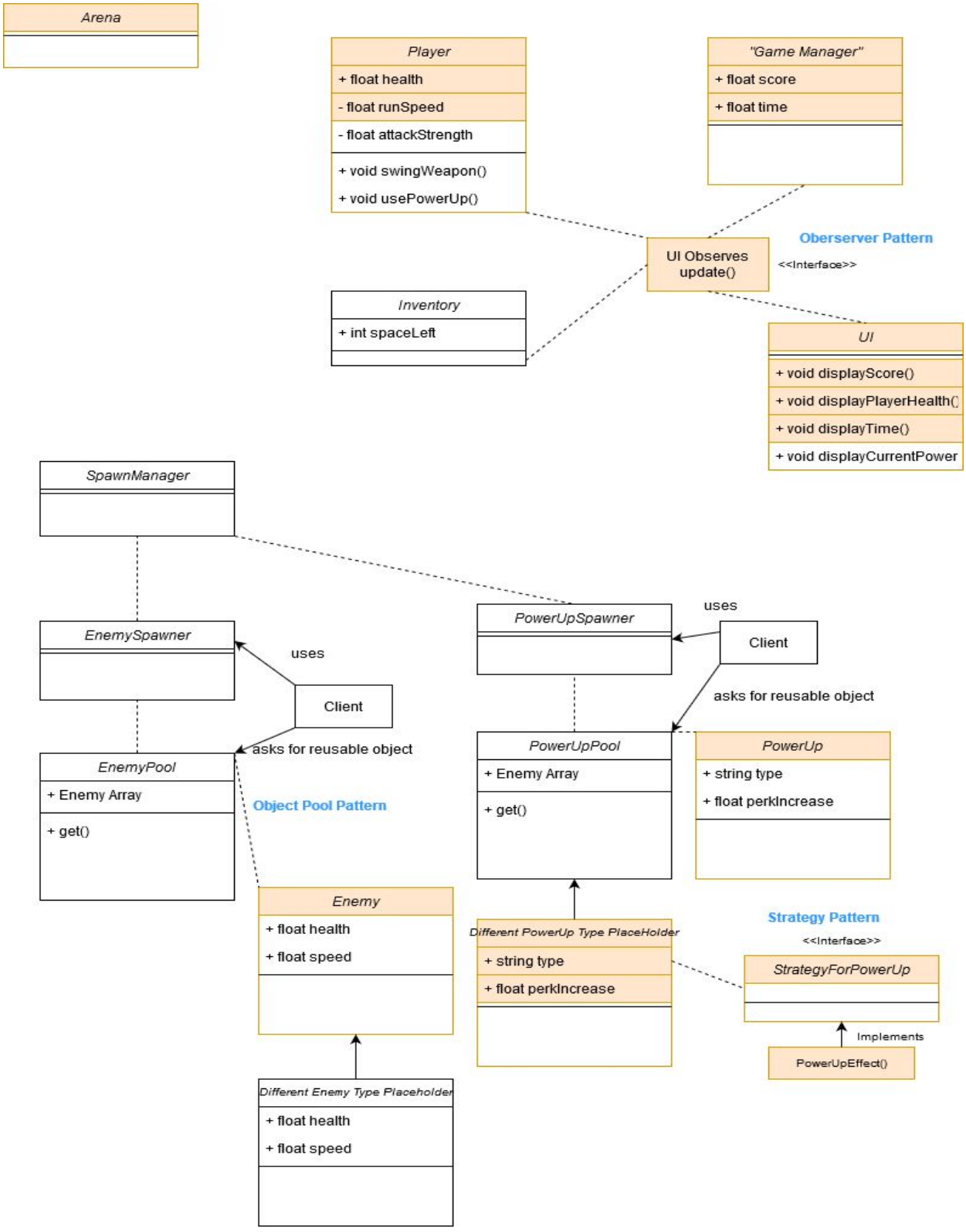
Luke Soguero, Kenny Schader, Austin Park

## Final State of System:

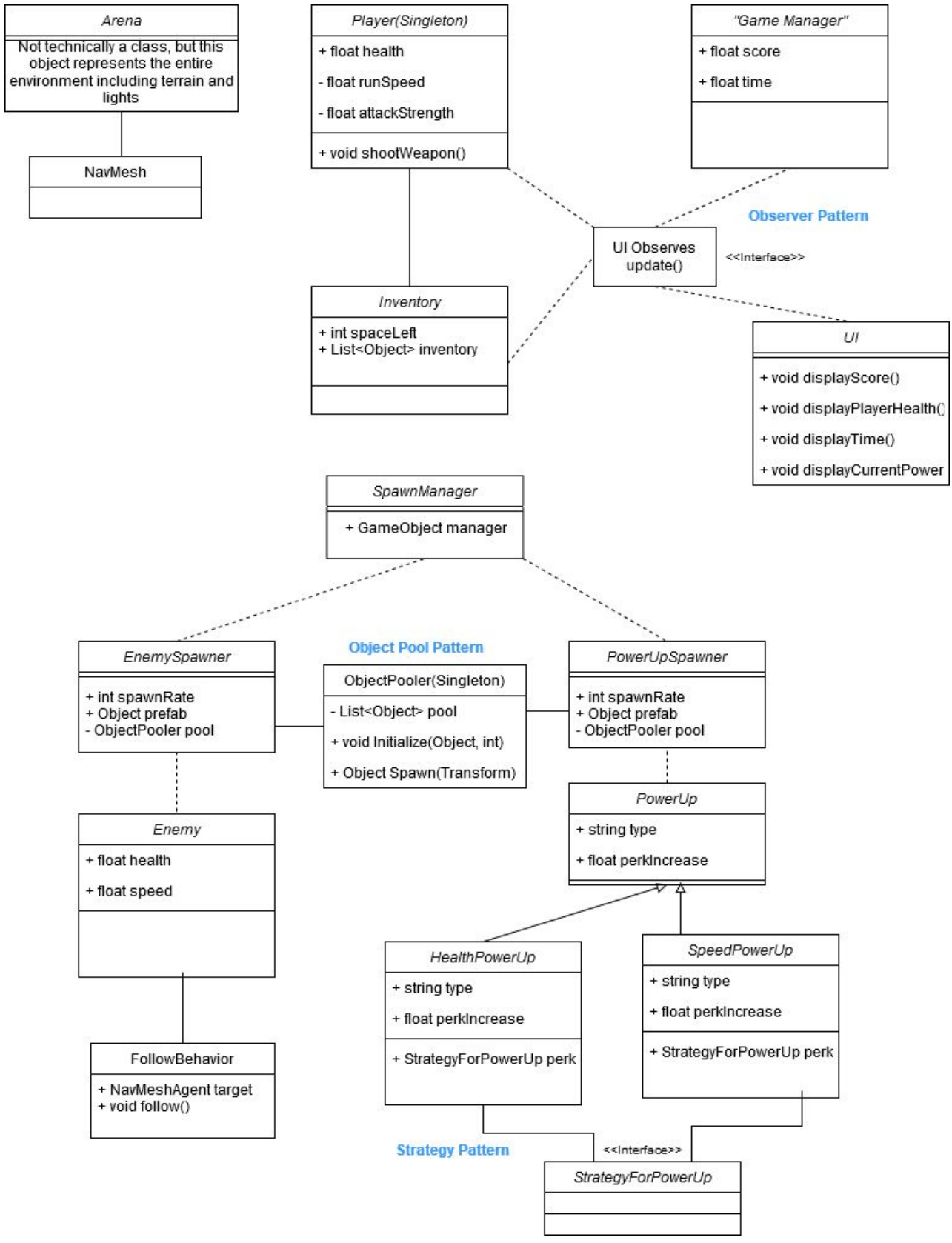
We were able to implement most of the features we originally planned on for our game. The user starts at the main menu where they have the option to view their high score, start the game, or quit the game. If the user starts the game, they spawn onto a 3D terrain with various obstacles. The user has a number of helpful indicators on their screen including a health bar and a mini map. Immediately, enemies begin spawning and moving towards the player. If an enemy gets close enough to the player, they will begin dealing damage. A player can attack an enemy by picking up a weapon and shooting the enemy. Additionally, the user can pick up helpful power ups that spawn randomly around the map including extra health and extra speed. As the game progresses, the enemies get faster, making them harder to outrun. The only feature we did not implement was the ability to pick up powerups and save them for later. Instead, power ups are used as soon as the player walks over them. Additionally, some changes were made to the class design, which you will see below.

# Final Class Diagram and Comparison:

## Original Diagram



Final Diagram



Changes to Diagram-

One difference between our original design and our final design is the Spawner and ObjectPooler layout. Instead of creating two different ObjectPooler classes for enemies and power ups, we decided it would be cleaner to create one general purpose ObjectPooler that could be delegated to each of the spawners. Another difference is that we used the singleton pattern for the Player object.

## Third Party Code:

Our project includes some third party assets downloaded from the Unity Store. These include the FastIK package and a terrain/vegetation pack. We also utilized online video tutorials from the YouTube channel Brackeys to complete some aspects of the project.
https://assetstore.unity.com/packages/tools/animation/fast-ik-139972
https://assetstore.unity.com/packages/3d/environments/low-poly-free-vegetation-pack-176906
https://www.youtube.com/channel/UCYbK_tjZ2OrIZFBvU6CCMiA

## OOAD Process:

When we first started we had a lot of ideas on what we wanted to do for a project. Once we all agreed on this survival game we needed a way to organize all our data. All of the UML diagrams helped a lot by getting out ideas and thoughts on a paper. Using the diagram it was easier to see how we could organize our code. A process we tried to follow was the “Thinking in Patterns Approach”. We started by thinking about which patterns could be applied to a particular problem in our system, and then we analyzed each of those patterns to find the one that would fit the best. This approach worked well for some problems, but it turned out to be difficult for others. We discovered that some problems didn’t require a strict object-oriented solution because the Unity engine already had built-in solutions or tools that solved the problem. For example, we ended up not having to implement the full Observer pattern, including the notifyListeners function, because Unity had an easier way of handling UI updates. We found that commonality and variability analysis was another good way of approaching our design. This approach allowed us to identify variations in our code and use design patterns (or

parts of design patterns) to encapsulate the differences. This type of analysis helped us decide how to design our spawner and object pool components.