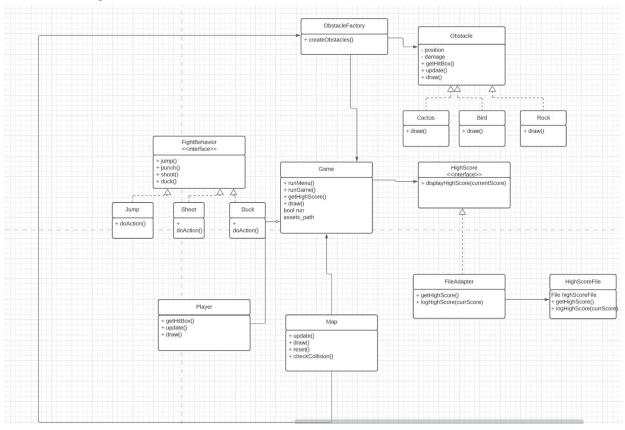Name of Project and Members: Rootin Tootin Yeehaw Simulator by Nick Lamonica, Bryce Woods, and Jaci Stickrod
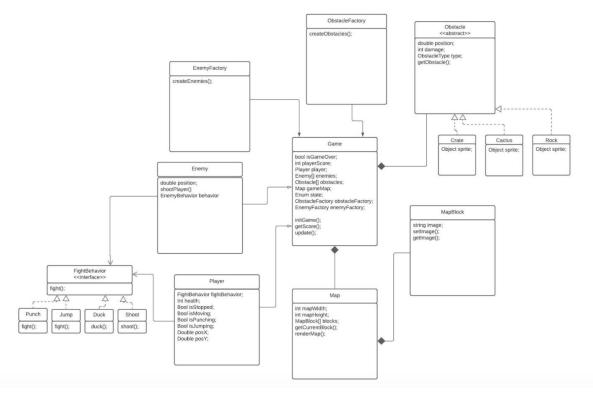
Final State of System Statement:

Our project has a simple menu and basic game functionality. We have assets for cacti, birds, and rock obstacles, and the cowboy player. The player can jump and duck. There is a health bar that keeps track of a player's health, and if the player runs out of health then the game ends. We also use a text file to keep track of all the scores. We did not implement enemies, the shooting behavior for the main player, or the loading screen. Unfortunately, we did not have enough time to create enemies for our game, and the shooting behavior was not well adapted to the way we created our obstacles. We didn't find the loading screen necessary for our project.

Final Class Diagram and Comparison Statement
Final UML diagram

UML Diagram from Project 4

Design Patterns Used
- Mediator
- Adapter
- Singleton
- Decorator

      The major changes to our class diagram structure since project 4 were the addition of the Adapter pattern to support getting the high score from a text file. The adapter pattern is useful here because if we decide in the future that we needed to use a database for our high score or other data, we could switch our adaptee and reuse the adapter code. Since we did not implement enemies, we do not have classes for those anymore. We took out the MapBlock class and now the Map class inherits from the obstacleFactory class so that it can check for collisions.

Third-Party code vs.Original code Statement
Our third-party code from a library call was PyGame.
View the library here: https://www.pygame.org/wiki/about. Other than Pygame we just used code from the Python standard library.

Statement on the OOAD process for your overall Semester Project

List three key design process elements or issues (positive or negative) that your team
Experienced in analysis and design of the OO semester project

1. We had very organized our sprints and distributed tasks fairly.
2. Our original UML design, while not 100% accurate, gave us a good understanding. of the project and what was needed.
3. Testing was minimal.