https://github.com/CSCI-4448-5448-OOAD/final-project-asbo8637

Decatur

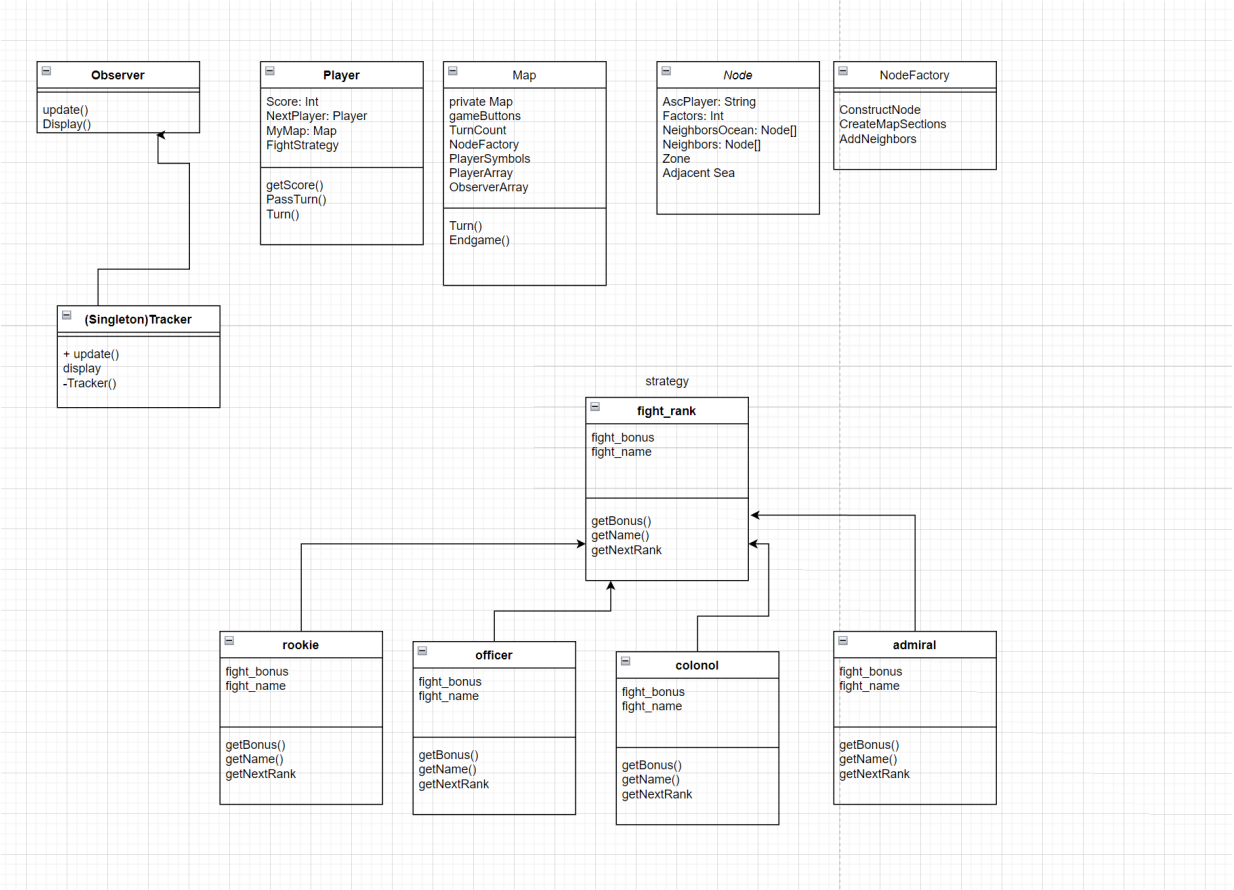
Project Team Members: Saulo Guzman, Assah Boneh

Video Link:
https://drive.google.com/file/d/1qOgjwHnqdWuhZgFeHHlkvGh1AXKe3Eyn/view?usp=sharing

2. Final State of system Statement

The final statement in the project was that most of the features in the board game were demonstrated. We were able to show the board game map with nodes and button options for the player. Each color of the node demonstrated a player with a number of factors. The goal for each player is still to attain the most factor on the map, by expanding or fighting to reduce other factors. Many features for the board were unfortunately not implemented due to the time constraints of the project. Some implementations that the player could interact with on the map to expand, move, or fight. The four patterns that were used in the code were strategery, observer, singleton, and factory. In projects 5 and 6 we were planning to use the command pattern where the player was given the options to execute the decision. Over time, we learn that Godot is limited in that feature, so the strategy pattern would of best for calling the algorithm during run time. However, we implemented something similar to the command pattern but decided to declare our four patterns as something else. Another pattern that was disbanded from the project was having a Decorator for the card system. During the interview with Professor Jim, he mentioned that is more important to get 4 patterns with a functional game and that there wasn't enough time to do 100% Replication of the board game. So the card system and port system were dismantled in projects 5 and 6.

3. Updated UML



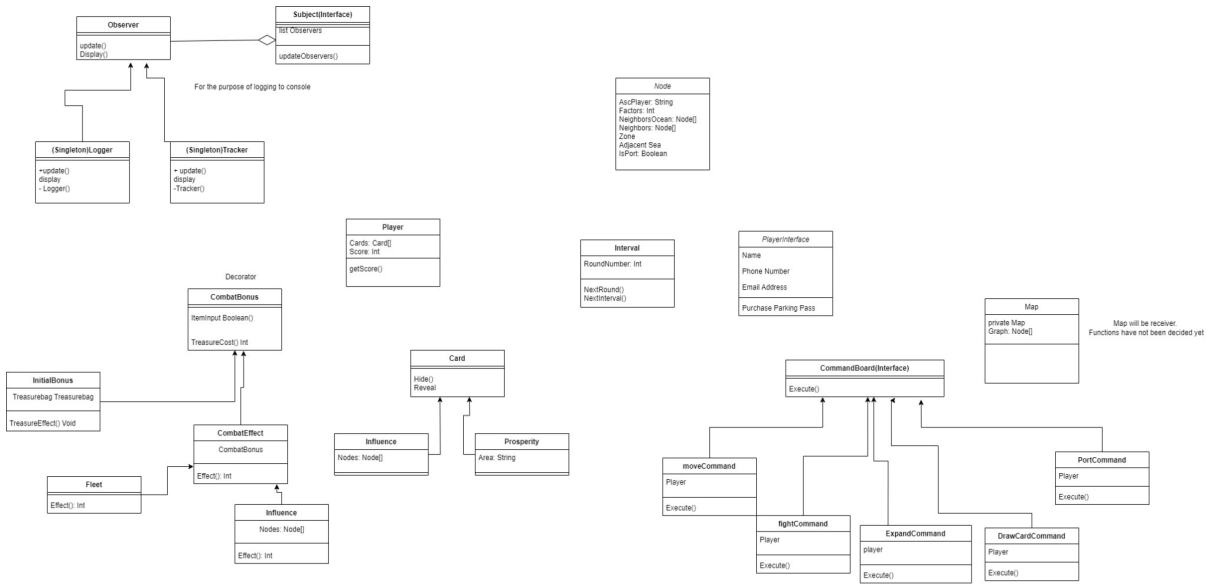
Some objects we decided not to include as they are only used for displaying the board or the code was already part of Godot. For example, we did not include the buttons and the signals they sent, because that code was provided by Godot.

The strategy pattern was used with fight rank. That changed the bonuses each player receives to fighting and defending.

The observer and singleton patterns were used together as we did in the arcane project.

We also created a factory class that creates nodes, to be used. It also connects nodes with all their neighbors.

Project 5 uml



As you can see a lot of ideas were changed from project 5. We removed the player interface, interval, card, command board, and combat bonus classes.

Node Factory was added, and fight strategy was added. Additionally, a lot more functionality was added to the map.

4. Third Party Code:

All the code in the repository is original. No frameworks were used, only the base godot package.

However, we did use some tutorials and documentation such as: https://godottutorials.com/courses/introduction-to-gdscript/godot-tutorials-gdscript-15/#:~:text=Classes%20can%20also%20have%20class,in%20the%20entire%20class%20file.

https://www.digitalocean.com/community/tutorials/gangs-of-four-gof-design-patterns

And a lot of syntax was used from the official godot documentation: https://docs.godotengine.org/en/stable/index.html

5. Overall project statement

-An issue in the project was that GitHub was not fully integrated with the terminal of Godot. There is a way to integrate godot with github, but we found it too big of a hassle to be worth it. This unfortunately led to many merging errors with scene files causing a code break. We decided to just drop the updated files, and clone after we changed the files. Meaning a lot of code had to be created together on one device.

-Another issue was the fact that gd code files are very similar to python syntax. This requires us to be more creative about a lot of pattern designs because Godot doesn't have all the features to make every pattern.

-A positive occurrence for the semester project was a lot of integrated features of Godot. The software was meant for creating a 2D video game. This made creating the map, clicking buttons, and interacting with the player incredibly easy.