Code Quality Report

Design Choices made:

We found this code to be generally very clear and readable. In terms of design choices, we noticed that instead of having a separate namespace for triggers, there was a trigger function. One aspect of this function that we didn't like was that it required you to add the new key-word to create-minion each time a card with a new trigger was added. Since there is no namespace that shows all of the trigger key-words, we didn't know right away that we had to modify create-minion each time there is a card with a new trigger word.

A small annoyance we noticed in their design choice was in their "get-random-enemy-minion-id" function. We were getting an error while we were using this function and then realized that the function returns an array of the state and minion-id. The function name didn't seem very clear, because at first glance we assumed this function would just return the minion-id.

Level of Abstraction:

The code maintains abstraction such that the code is easily extensible, while maintaining readability. Helper functions are frequently used, and named well. Documentation for these functions was brief, but clear. Functions were sorted into their separate namespaces in a logical, understandable way.

Tests of the Cards:

As a first test, all of the card tests passed when run. Tests were for the most part understandable and comprehensive. We could not find an issue where a base card function went untested.

The biggest issue with the tests was not with the tests themselves, but that they were undocumented. It would have been helpful if the tests were commented such that people who were unfamiliar with the code could orient themselves better.

Tests of the Engine:

All basic functionality of the view works very well. Cards work as expected, with abilities showing up upon hover, and new cards being drawn from the deck. Secret functionality works, and is triggered by game events. Hero powers are operable. Mana decreases as expected when cards are played, and resets to 10 at the end of each turn. Minion powers all work. The "show opponents cards" and "reset" functionality also work as expected. When the end turn button is

pressed, the game switches to the other players turn. Supplemental effects, such as aesthetic qualities, sound functionality, and effect volume, work as expected.

If there is one issue that could be pointed to as imperfect, it would be the “show history functionality” As expected, the side-tab opens up, but there is no list of past events. Upon close consideration, there is a history, but the singular events are thin and nearly invisible, and contain no description.

There was one error that occurred during gameplay. It is occasionally possible to click the hero, and an arrow will come out of the hero as though it is preparing to attack. The only way to remove this arrow is to refresh the webpage of the view. This error was not reproducible.

Misplaced Logic

For the most part, this logic appears quite sound and well thought out. If we had to nitpick, this repository could benefit from more namespaces for specific game elements, such as a namespace for everything attack-related, or everything related to triggers. It would make the code easier to navigate when debugging entire features, rather than specific functions.

Modeling of the State:

The state is responsible for holding the cards, minions, decks, battlecries, dead minions and secrets for each player. This approach allowed for expressive and extensible functionality of things such as removing dead minions, by moving them into the dead-minions array. So, if you were to recall a dead minion, you could preserve characteristics it held when it was alive, such as damage-taken. Therefore, the state is responsible for preserving the current state of every single card in the game, and it is a very effective model.

Responsibilities of the Functions:

One thing we liked a lot about this groups function presentation was the brevity of the functions. This allowed for a specificity that was optimal for understanding the functions purpose and usage. Function tests were also comprehensive, usually larger than the functions themselves. Connections between functions were sensible and easily extensible. Sometimes the names of functions were not the most intuitive, but rarely did they provide any legitimate barrier to understanding.

Overall Analysis

Overall, we had a very positive impression of the code base. All tests passed as expected. There were a few issues, but they were generally minor and/or superficial.