## Peer review

The code is well structured and the structure makes sense. It feels natural to navigate. The MVC structure is very apparent and well-made. The model works independently from the view and controller.

The SOLID design principles are followed in most parts of the project. Single responsibility Principle is implemented in many parts, as an example in the model module they have divided code which are handling collisions into separate classes depending on which type of collision it handles instead of putting everything in one class.

The project uses mostly private attributes with getters and setters to make the code open for extension but closed for modification. Several abstract classes are used when needed to add functionality used by several classes. By doing so, the functions can be adapted to each class without having to compromise with other classes that need to use the function.

They also use the factory pattern, which helps follow LSP. Even if there is only one fighter right now, it seems like it would be easy to add more in the future while still following this principle. Another example of abstract classes used in the project is Fighter.java. A class that aims to reduce coupling between Fighter and the class inherits it “EvilWizard”.

It seems like they will be using the Chain of responsibility patterns later as well. (specifically, performing attacks, though this has not been finished yet).

The project also uses some interface, mostly for observers which helps to reduce dependencies. However some of the interfaces, like UpdatableTimer and Callback, don’t seem to do a lot. They’re only implemented by one class, but there might be a good reason for using these. Also the interface IAttack is not implemented anywhere and is very poor, but it is something we expect to be expanded on since the attacks are not functional at this time.

Most parts of the project are clear, and the code seems consistent and written in similar ways. There are no files which we can see that are written in a drastically different way.

Like with most programs it takes a bit of time to get a general understanding of the program, but since the files are divided in a logical way and proper naming of methods and attributes are used we could understand and follow along with the code pretty fast. It helped that the names were intuitive. However there are two folders named view (the MVC view and a view in Utils), maybe re-name the one in utils.

The parts of the code that are documented are well documented but the project is still missing a lot of documentation, although this seems to be on the todo-list already. We would have liked to have a README file to get a faster understanding of the project from the start.

We believe the code will be relatively easy to maintain. New fighters can be introduced easily because of the abstract class Fighter and the FighterFactory, more players could probably be added without much problems, and new objects and obstacles also seem to be easy to add.

The model contains dependencies to the libgdx library. If possible try to work around that and avoid dependencies between model and libgdx .

The code does use abstractions in many cases through interfaces but also through inheritance. Although some kind of interface to guarantee a base set of moves for Fighters on top of moves that are special to a specific fighter. It would make the code more expandable. Let's say you want to add a Fighter selection screen, or even a Fighter customization menu in the future it could be quite useful.

Looking through the tests, and judging by the names it seems that the tests are reasonable, useful and thought through. Things that can go wrong, for example hitboxes and collision detection are well tested and it feels like the author has thought of what can go wrong, instead of just testing what can go right.

Because the program does not access or store any information from the user and is run directly on the user's computer we don't think there are any security issues. The program was runnable without any noticeable issues with the performance for us.

Many user stories are still not implemented, for example fighters can only move around the screen,stamina bar and "2.1.3"  User story .

About 2.1.2 User story ; the timer is visible and running but the game does not finish or restart.

User story 2.1.4 with less priority is implemented before other user stories with higher priority(not implemented) , for example US 2.1.1.

The game is supposed to have a start screen according to the RAD document. in order to choose a character.

Looking at the SDD, the code appears to reflect what is written in the SDD, and if something is off, it is probably very minor.

Besides the small details we have mentioned previously, the group seems to know what they are doing and only need to finish the rest of the program.