Project report for 2D RPG

We decided to work on the open source project called “Open Source A-RPG Demo.” We found this project on Github and it runs under the game engine Godot. As a team we have not heard of this game engine before so we had to read about it. This games engine is fairly new because it was released in 2014. There is very little documentation compared to more widely popular engines so we had some trouble with it. The other problem we ran into is that it uses its own scripting language called GDscript. We have not heard of this language so we had to take a crash course in it for our project. The open source project we found was building upon a RPG style game. It has a simple gameplay, core systems,user interface,and visual effects. It is in the very early basic stages of a game.So as a team we wanted to work on the games look and weapons.

When we first started it up we found some bugs right away. One bug we found and were able to fix was the error before we could even run the game. That was fixed with changing a line in the file called “project” in the repository we cloned. Another bug we ran into was that the player model would not move and the screen looked zoomed in. We fixed this by going in the project settings and changing the key bindings. The original project does not tell you this so it took us a while to figure out what the keys were to play the game properly. So our solution was to include an instruction document in the repository. Once we got the main project to work correctly then we were able to tackle our proposal goals. Those goals where to change the games sprites and to add a new weapon.

Traci Trojan’s contribution

Github was pretty new for me, so learning how to change the structure of the files turned out to be a challenge. At first, I was going to change the file path of each individual file, but there are hundreds of files in this project. As a result, I searched online for a way to create folders and put the files into the folders, along with duplicating them into other folders in a much more efficient way. After changing the structure of the files on github, other team members and I were able to add and change the files on github. My other big task was updating the read me. I kept the original read me and style. I put the information right above the original read me. I made sure to include all of our names along with every single file we changed and added, which I had to look up in all the commits. I also included the ideas we changed from the original code and an important note to look at our instructions document.

Victoria Naranjo’s contribution

For my part in the project I wanted to tackle the player models and the mob models. The base game had a nice artistic style to it but it was missing a theme in my opinion. So as a team we came up with having the player models have a medieval style theme. So the main character that was originally a long green rectangle was then changed to a wizard cat. I chose this because i wanted to make the game comical and fun. The next mob I changed was called porcupine in the scripts and this mob job was to charge at the main player. This mob was depicted as a tiny pink cube, so to fit our new theme I changed it to a goblin instead. The last mob i changed was called nest on the scripts and it shot arrows. This gave me the idea to change what was a blue cube to a dragon. The cube then shot out triangles that i changed into fireballs. The game has a little more style and fun aspects to it.

Damien Mousavi’s contribution

My part of the project included fixing the game crashing after dying. The issue was caused by the input registering as invalid once player object was in “die” state. This was fixed by adding a method to reload the game after player’s death. Another part was that I also added new weapon. The new weapon I chose was a hammer. I was able to use the same template as the already present sword weapon. The difference was that I tweaked some attributes,changed the in-game model and the animation speed. I then linked the weapon scripts to statemachine to allow proper functioning while using combo attacks. This was hard to figure out since the game engine and code language was all knew to me.

In the end this whole experience was new to each of us. We did our best to make a contribution to the open source community. We applied the knowledge we got from class on our open source project. We could not separate the folders that we did not change since the engine relied on the folder structure to find and use different component. Doing so would’ve meant us re-importing and mapping every asset in the game which we did not have enough time for. A summary of folders that we changed or added can be found in the README file. There is also an easy to use executable in the source folder labeled executable. This will let you play the game without the godot engine.