**Log of optimizing an Unreal project**

Nicholas Lane /1904291

**Introduction**

For my optimization assignment I am going to be using an old team project from collage to profile and optimize. This is my most up together project in unreal and I know that when me and my team were creating this project, we did not focus at all on optimization. This means there is likely to be some optimization errors within this project I can find and work on.

This game is a 2D RPG where the main mechanics are a turn-based battle system, level up system and a couple of shops the character can purchase items from.

**Repository Of the unreal project:**

<https://falmouthac-my.sharepoint.com/:f:/g/personal/nl232500_falmouth_ac_uk/Esom5Zh_yAFOvQiApWOPjsgBdBP-5Tae7VwKCVdpQGR_vQ?e=8qOxzq>

**First Profiler Test**

The first thing I have done in the game is to set up a profiler test to check to see any points where the game will slow down. When running the game, I tested out the user interface for the shops, level transitions and the games combat. This Flagged up some points along the timeline that I can start to investigate and see if I can correct them.

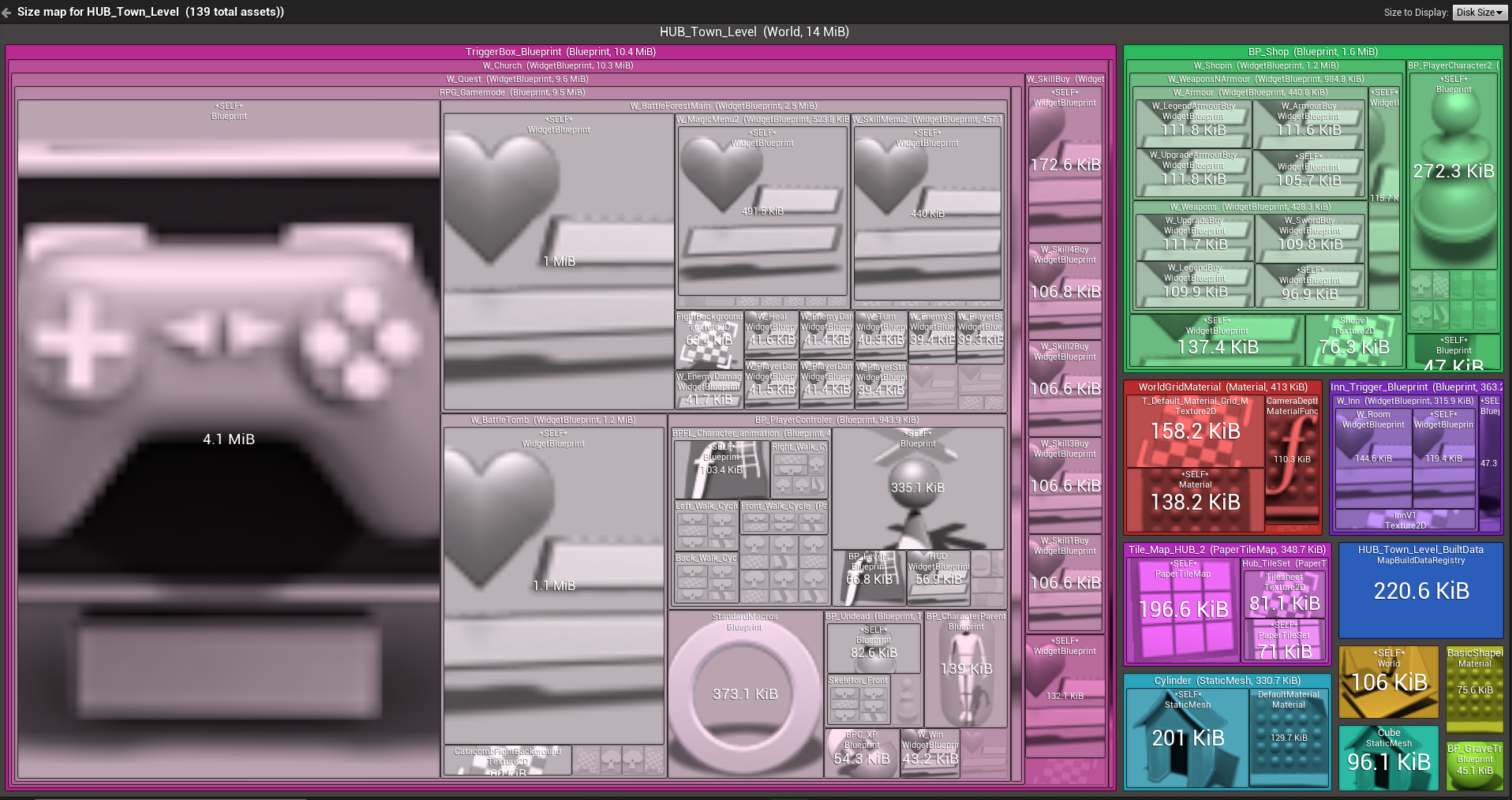
Profiler Timeline:



Where I have marked with black boxes on the timeline is where I should look and see what’s running. This will then let me see if these spikes on the graph are things I can modify and optimize. However, before I dive into this, I am also going to check the size map for any assets that are oversized.

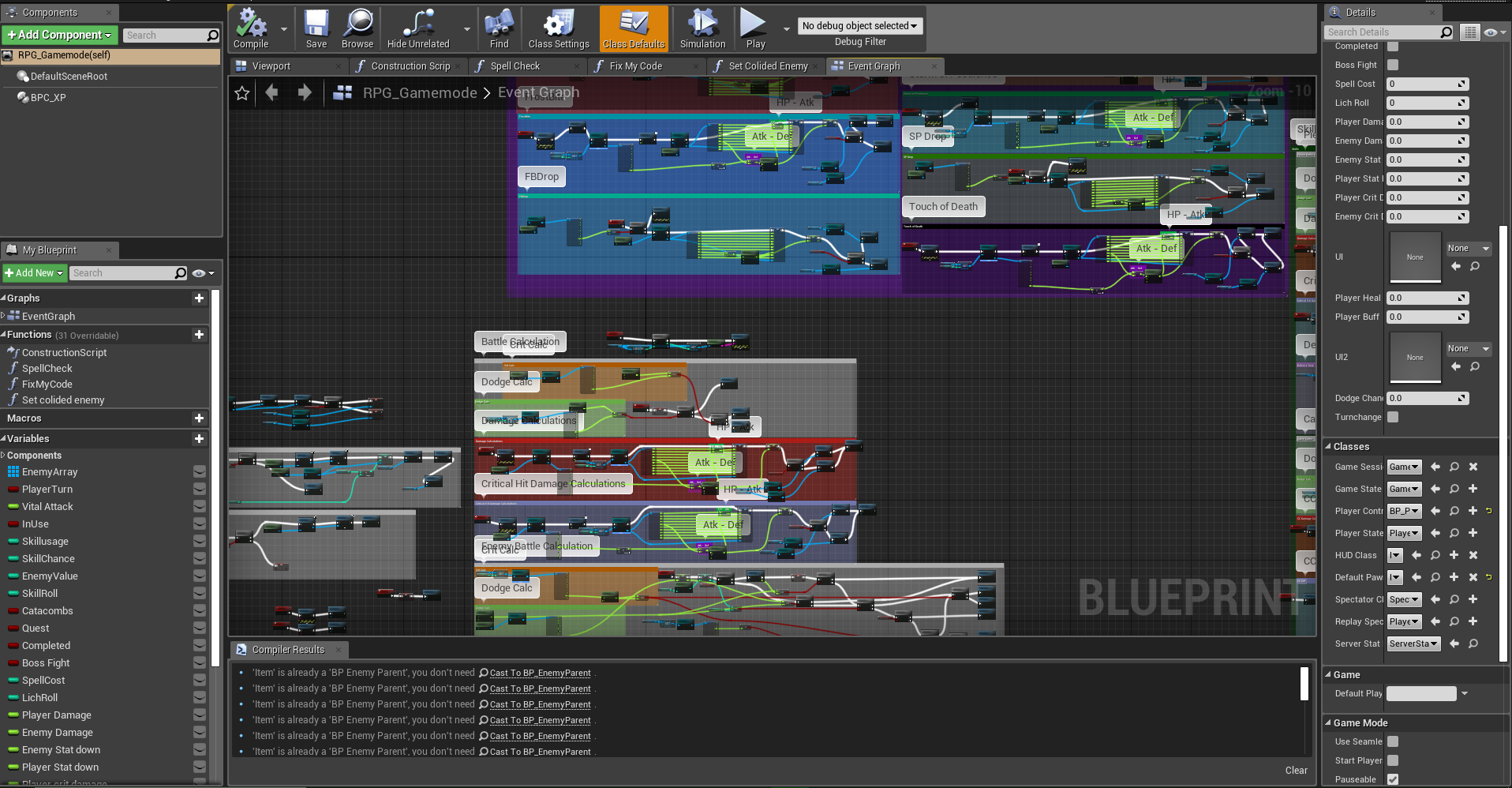
**Size Map Tests**

Before I go any further, I should check the size maps. I don’t expect there to be any assets that are out of proportion due to the art style of the game. I started with the HUB town level and found something that immediately stuck out.



As you can see the asset that shows the most is the game mode where a lot of the code for the main battle system is held. Opening it up to see what code it contains I can immediately see some problems.

Inside the game mode:



Firstly, there is a lot of code within the same window. A lot of these could be converted into functions to keep the window tidy. This However isn’t so much of an optimisation issue. A much larger issue is the amount of cast nodes used within this window. It is being called an unnecessary amount of times. A solution to this is to call the cast nodes at the beginning and make them variables. Another way to better optimise the cast nodes is to convert them to pure cast nodes when able for when it is just getting a value.

Other than these 2 issues I cannot see any other major issues. Though as the code is incredibly messy It is more than possible that I have missed parts that can be optimised.