**Challenges, changes and requirements of CPPPR**

**Issues encountered while integrating this modular complex system:**

* Initial difficulty designing data structures which are modular and performant enough.
* Difficulty learning how to properly set up a multi-project c++ solution and a library based project.
* Difficulty learning new versions of the required libraries (GLFW, GLEW, GLM).

**Performance of the system:**

At least on the machine used to develop this system, the performance of this system is satisfactory, even under extreme testing conditions. This test was conducted by using the demo app to create thousands of primitives while witnessing the change in performance over time. Despite thousands of primitives being created and rendered in real time, the program did not stutter or slow down to any noticeable degree on my machine.

It took upwards of 3 minutes of non-stop primitive spawning every frame to cause a noticeable drop in performance on my machine. This performance can be further improved by implementing an upper limit in how many primitives can be drawn in one frame, splitting the load up into smaller batches (currently each primitive type are all rendered at once regardless of number).

**Required Changes for the system to function as intended:**

Although the system functions fine at a basic level, there are multiple features which would need to be added in order for this complex system to be practical in any real application.   
The required features are as follows:

* Support for textures and texture mapping
* Support for model instancing
* basic lighting
* basic particle physics
* basic post processing options
* numerous additional primitive types
* Multi-threaded support

However, for the proof of concept required for this assignment submission, the currently implemented features are satisfactory (these are already implemented):

* Ability to render many primitives with very few function calls
* At least two different primitive types
* Ability to customize individual primitives with generic attributes
* Ability to be comfortably integrated into most suitable applications