### **Profiling and optimisation:**

Throughout the development of the artifact profiling was carried out using the performance profiler that is built into visual studio. This was used to measure CPU usage. I also tried measuring the GPU usage however this never changed no matter how many models were added, I assume this is because the models were not detailed enough to require much of the GPU’s power.

### **Objectives:**

The objective when making this artifact was to create a graphic simulation of particles hitting glass and then disappearing a few seconds after to simulate dust or sand particles hitting a glass pane.

### **Software and development approach:**

The artifact aims to simulate collision of dust/sand particles with a glass pane, this is simulated using boxes to represent the particles and a transparent cuboid to represent the glass.

The particles spawn in random positions that is bound so each will hit the glass in different places. They approach the glass at a constant speed. An AABB collision test is run in the update function that tests the bounds of each particle against the bounds of the glass. This allows detection of when each particle collides with the glass, when this happens the particle is stopped, a delay function is triggered which then stops the particle rendering after a couple seconds of hitting the glass.

After deciding on what to create research was done into OpenGL mainly using learnopenGL [1] to try and teach myself how to use it. I also looked at a video on transparency and blending in OpenGL [2] which I used to make the glass pane transparent.

I modified the template code that came with this module to get two models loading with one being the transparent glass and one being a box to represent a particle, I then made this approach the glass, this was then modified again for worksheet 3 to allow a larger number of boxes to be loaded.

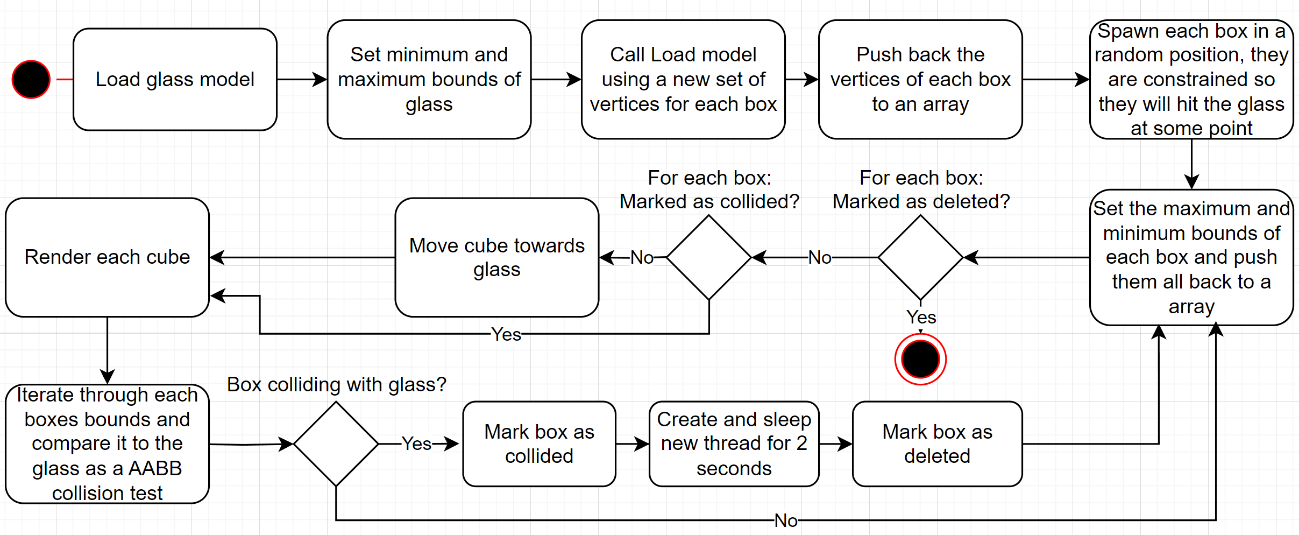
For worksheet 4 I added AABB collision to the boxes. This took some more research from learnopenGL and an explanation that I found here [3]. I then added a delay function as well as a check when rendering which allowed each cube to stop being rendered a few seconds after hitting the glass, essentially deleting them.

### **References:**

[1] *Learn OpenGL, Extensive Tutorial Resource for Learning Modern OpenGL*. <https://learnopengl.com/>.

[2] *Gordan, Victor. “OpenGL Tutorial 17 - Transparency & Blending” YouTube, 28 May 2021* [*https://www.youtube.com/watch?v=crOfyWiWxmc*](https://www.youtube.com/watch?v=crOfyWiWxmc)*.*

[3] *3D Collision Detection - Game Development | MDN*. 1 Aug. 2024, https://developer.mozilla.org/en-US/docs/Games/Techniques/3D\_collision\_detection.



**Artifact UML Activity Diagram Diagram:**

Figure 2 Artifact UML diagram

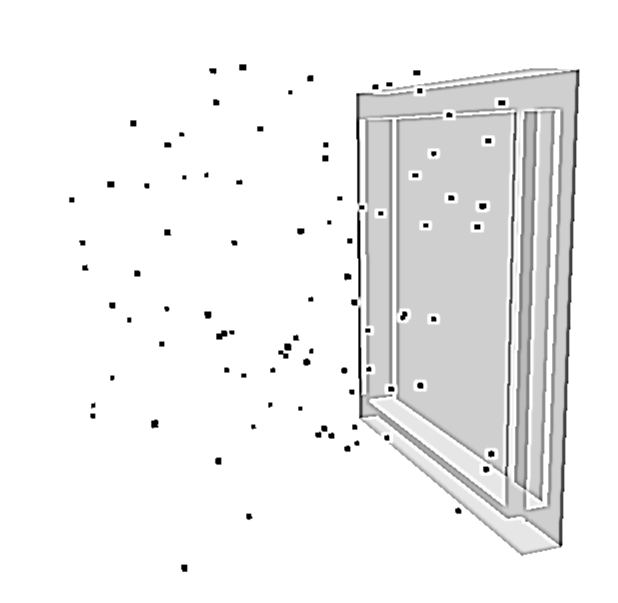


Figure Image of artifact