**Things which the rendering engine should have**

**Features to add now (V0.0):**

* Release on 7/11/2023
* converting a PNG to a mesh
* ability to animate shapes
* Able to animate characters using rigging and skeletal animation
* add ability to set solid colours to shapes
* add ability to delete shapes
* text support
* saving settings to a file
* post processing implements using Vulkan

**Features to add later (V0.1):**

* Release on 11/11/2023
* add ability to change textures from Gui
* add ability to add offsets to textures
* ability to toggle V-Sync
* render circles
* Render heath bars

**Features to add later (V0.2):**

* GPU accelerated particle physics
* multiple textures support
* Multi-threaded rendering
* instancing
* pause rendering when nothing is happening and duplicate frames

**Features to add later (V0.3):**

* direct storage
* texture compression
* use VMA for memory allocation
* mesh shaders

**After creating this engine:**

* physics
* input system
* sound
* making the gamE

**To render Fonts using stb\_truetype**

* Create texture for each character using truetype
* store that text and its data needed
* pass them to the GPU and create all the Vulkan shit needed
* when needed using instanced rendering, render that text

**In order to generate the vertices of a PNG**

* create an outline

1. create a matte
2. trace the edge of the matte by getting the pixels
3. which are outside the matte

* at regular intervals in this outline, add vertices to it

1. split the outline into two lines of edges, alternating
2. repeat this again recursively till the desired number of intervals are created
3. at the mean of these lines, create a vertex

* use those vertices to generate a mesh using shaders in Vulkan
* https://github.com/Half-People/ImGui-2D-HArrow

**In order to implement post processing techniques:**

**Formulas/Algorithms for post processing:**

* Saturation

https://www.geeksforgeeks.org/program-change-rgb-color-model-hsv-color-model/

* Contrast
* https://www.geeksforgeeks.org/changing-the-contrast-and-brightness-of-an-image-using-python-opencv/
* Gamma
* Offset
* Highlights
* Blue correction
* Vignette
* Exposure
* Chromatic aberration

**Implementation:**

* + 1. Pass it through until we get final buffer
    2. Take that buffer and put ti thorugh a pipeline
    3. Modify the fragment shader of that pipeline to apply post processing effects