**Task 1**

In task 1, I modified setTexture method in order to remove limitations of width and height values of power of two. So I allowed to any sized image with by using LINEAR as the minification filter and CLAMP\_TO\_EDGE as texture settings for wrapping. By blocking texture from repeating, CLAMP\_TO\_EDGE parameter fixes problems that arise when non power of two textures are used with normal wrapping approach. Smooth interpolation is feature of the LINEAR filter that enhance texture rendering at oblique angle.

Added:

A computer code with text

Description automatically generated

**Task 2**

In that part in order to lighting we first initilizate required variables in constructor they are lightPosLoc, ambientLoc and enableLightingLoc. Later we must update setMesh part to handle lighting with adding bindbuffer and bufferdata. Moreover we must update draw that should be concurrent with given values(LightX and LightY). So, gl.uniform3f(this.lightPosLoc, lightX, lightY, 1.0); with this code now gl will be concurrent with variables of LightX and LightY. So increasing or decreasing Ambient Light Density bar will change the lighting ratio.

A computer code with black text

Description automatically generated

With above code we can bind gl to functions and in the void main these features will be concurrent that will provide us to change the lighting options with using if else and let us to show the current lighting setting by changing light in sphere.

In meshFS, with extra we defined the lighPos in order to specify the lighining ratios. Also in void main we calculated lightposition and ambient color and removed texture2D(tex, v\_texCoord); and we replace it with our calculated variables.

**Before**

A red and white brick ball

Description automatically generated

**After**

****

**A red and black ball

Description automatically generated**