```
const Canvas =document.querySelector("Canvas");
const webgl =Canvas.getContext("webgl");

if(!webgl){
    throw Error("This is not related to webgl")
}

var Vertex = new Float32Array([
        //front triangle bottom
             -1,1,1,
             -1,-1,1,
             1,-1,1,
        //front triangle top
             -1,1,1,
             1,1,1,
             1,-1,1,

        //back triangle top
        -1,1,-1,
        1,-1,-1,
        -1,-1,-1,
        //back triangle bottom
        -1,1,-1,
        1,-1,-1,
        1,1,-1,

        //left triangle
        -1,1,1,
        -1,-1,1,
        -1,-1,-1,
        //left triangle
        -1,1,1,
        -1,-1,-1,
        -1,1,-1,

        //rigt triangle
        1,1,1,
        1,-1,-1,
        1,-1,1,
        //right triangle
        1,1,1,
        1,-1,-1,
        1,1,-1,

        //bottom triangle
        -1,-1,1,
        1,-1,1,
        -1,-1,-1,

        -1,-1,-1,
        1,-1,-1,
        1,-1,1,


        //top triangle
        -1,1,1,
        -1,1,-1,
        1,1,1,

        -1,1,-1,
        1,1,1,
        1,1,-1

])

var Colour = new Float32Array([
    //front triangle top
         1,0,0,
         1,0,0,
         1,0,0,
    //front triangle bottom
         0,1,0,
         0,1,0,
         0,1,0,

    //back triangle top
    0.2,0.8,0,
    0.5,0.8,0,
    0.2,0,0.8,
//back triangle bottom
    0,0.8,0,
    0,0.2,0.8,
    0,0.1,0.8

]);

const textures = new Float32Array([
    0,1, 0,0, 1,0,
    0,1,1,1,1,0,

    0,1 ,1,0 ,0,0,
    0,1 ,1,0, 1,1,

    0,1, 0,0, 1,0,
    0,1,1,1,1,0,

    0,1 ,1,0 ,0,0,
    0,1 ,1,0, 1,1,

    0,1, 0,0, 1,0,
    0,1,1,1,1,0,

    0,1 ,1,0 ,0,0,
    0,1 ,1,0, 1,1,





]);

const vBuffer=webgl.createBuffer();
webgl.bindBuffer(webgl.ARRAY_BUFFER, vBuffer);
webgl.bufferData(webgl.ARRAY_BUFFER, Vertex,webgl.STATIC_DRAW);

const cBuffer=webgl.createBuffer();
webgl.bindBuffer(webgl.ARRAY_BUFFER, cBuffer);
webgl.bufferData(webgl.ARRAY_BUFFER, Colour,webgl.STATIC_DRAW);

const texturebuffer = webgl.createBuffer();
webgl.bindBuffer(webgl.ARRAY_BUFFER, texturebuffer);
webgl.bufferData(webgl.ARRAY_BUFFER, textures, webgl.STATIC_DRAW);
let imagedata = new Image();
imagedata.src ="1.png";


const vSource =`
        attribute vec3 position;
        attribute vec3 colour;
        varying vec3 fragcolour;
        uniform float angle,dx,dy,dz;
        attribute vec2 texCoord;
        varying vec2 fTexCoord;
        float x,y,z,m,j;

        void main(){
            x=position.x*cos(angle)-position.z*sin(angle);
            z=position.x*sin(angle)+position.z*cos(angle);
            y=position.y;

            m=y;
            y=m*cos(angle)-z*sin(angle);
            y=z*cos(angle)+m*sin(angle);



gl_Position=vec4(x*0.25*cos(angle),y*0.25*cos(angle),z*0.25*cos(angle),1)+vec4(dx,d
y,dz,0);
            fragcolour=colour;
            fTexCoord = texCoord;
        }

`
const cSource =`
        precision mediump float;
        varying vec3 fragcolour;
        varying vec2 fTexCoord;
        uniform sampler2D uSampler;

        void main(){
            gl_FragColor=vec4(fragcolour,1);
            gl_FragColor = texture2D(uSampler,fTexCoord);

        }

`

const vShader= webgl.createShader(webgl.VERTEX_SHADER);
webgl.shaderSource(vShader, vSource);
webgl.compileShader(vShader);


const cShader= webgl.createShader(webgl.FRAGMENT_SHADER);
webgl.shaderSource(cShader, cSource);
webgl.compileShader(cShader);

const program =webgl.createProgram();
webgl.attachShader(program, vShader);
webgl.attachShader(program,cShader);
webgl.linkProgram(program);
webgl.useProgram(program);

webgl.bindBuffer(webgl.ARRAY_BUFFER, vBuffer);
const positionLocation=webgl.getAttribLocation(program, "position");
webgl.enableVertexAttribArray(positionLocation);
webgl.vertexAttribPointer(positionLocation, 3, webgl.FLOAT, false, 0, 0);

webgl.bindBuffer(webgl.ARRAY_BUFFER, cBuffer);
const colorLocation=webgl.getAttribLocation(program, "colour");
webgl.enableVertexAttribArray(colorLocation);
webgl.vertexAttribPointer(colorLocation, 3, webgl.FLOAT, false, 0, 0);

webgl.bindBuffer(webgl.ARRAY_BUFFER, texturebuffer);
    const textureLocation = webgl.getAttribLocation(program, `texCoord`);
    webgl.enableVertexAttribArray(textureLocation);
    webgl.vertexAttribPointer(textureLocation, 2, webgl.FLOAT, false, 0, 0);

const texture = webgl.createTexture();
webgl.bindTexture(webgl.TEXTURE_2D,texture);
webgl.texParameteri(webgl.TEXTURE_2D, webgl.TEXTURE_MAG_FILTER, webgl.LINEAR);
webgl.texParameteri(webgl.TEXTURE_2D, webgl.TEXTURE_MIN_FILTER, webgl.LINEAR);
webgl.texParameteri(webgl.TEXTURE_2D, webgl.TEXTURE_WRAP_S, webgl.CLAMP_TO_EDGE);
webgl.texParameteri(webgl.TEXTURE_2D, webgl.TEXTURE_WRAP_T, webgl.CLAMP_TO_EDGE);

webgl.texImage2D(webgl.TEXTURE_2D,0,webgl.RGBA,webgl.RGBA,webgl.UNSIGNED_BYTE,image
data);
//webgl.generateMipmap(webgl.TEXTURE_2D);



var angle=0.06;
var angleSpeed=0;
var dx=0.0,dy=0.0,dz=0.0;
var move_X=0.017526;
var move_Y=0.01248546;
var move_Z=0.002416;
var bounce = false;

function draw(){
  //angle+=0.02;
  angle=angle+angleSpeed;

  if(bounce)
  {dx+=move_X;
    if(dx>1 || dx<-1){
      move_X=-move_X;
    }

    dy+=move_Y;
    if(dy>1 || dy<-1){
      move_Y=-move_Y;
    }

    dz+=move_Z;
    if(dz>1 || dz<-1){
      move_Z=-move_Z;
    }}

    webgl.clearColor(0.8, 0.8, 0.8, 1.0);
    webgl.clear(webgl.COLOR_BUFFER_BIT);
    webgl.enable(webgl.DEPTH_TEST);
    webgl.uniform1f(webgl.getUniformLocation(program, "angle"),angle);
    webgl.uniform1f(webgl.getUniformLocation(program, 'dx'),dx);
    webgl.uniform1f(webgl.getUniformLocation(program, 'dy'),dy);
    webgl.uniform1f(webgl.getUniformLocation(program, 'dz'),dz);
    window.requestAnimationFrame(draw);
    webgl.drawArrays(webgl.TRIANGLES, 0, Vertex.length/3)
}
```

```
draw();

function Increase(){
    angleSpeed+=0.02;

}
function Decrease(){
    angleSpeed-=0.02;

    if(angleSpeed<0){
        angleSpeed=0;
        }

}

function Rotate(){
    if(angleSpeed==0){
        angleSpeed+=0.02;

    }


}

function Shift_All_Sides(){
    bounce=true;
}
```