EXPLICACION CODIGO MUNDO:

Para la estructuración de los grafos se creó una especie de formato JSON en el que cada nodo corresponde a un país, este contiene información como lo es la id, el número de contagiados y muertos por país, su posición en (x, y, z) y una etiqueta la cual corresponde al nombre de cada país en este caso solo realizamos los países latinoamericanos (13).

```
var nodes = [{
id: 0 ,contagiados:0,muertos:0,value: 2.0,x: 0.0,y: 0,z: 0 ,label: "Nucleo"},
id: 1 ,contagiados:9931,muertos:419,value: 2.0,x: 14.57,y:-12.23,z:-
6.18,label:Argentina"},];
```

Para la creación de las aristas se debe estipular el origen y destino de cada nodo

```
 var edges = [{  from: 0,    to: 1,  },{
                 from: 0,    to: 2,  },];
```

Se crea la escene, camera, aspect, se establece el ancho y alto de la cámara su Angulo, se activa el antialias, y se le da a la cámara un modo de Domo.

```
 renderer = new THREE.WebGLRenderer({ antialias: true });
 scene = new THREE.Scene();
 aspect = window.innerWidth / window.innerHeight;
 camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.0
1, 4500);
 renderer.setSize(window.innerWidth, window.innerHeight);
 document.body.appendChild(renderer.domElement);
```

Los crean los controles de la cámara los cuales permiten trasladarse en modo Domo, se estable un Angulo de min de PI/3 para su apertura, como la distancia mínima y máxima al hacer zoom.

```
 controls = new THREE.OrbitControls( camera, renderer.domElement );
 controls.minPolarAngle=Math.PI/3;
 controls.minDistance = 23.0;
 controls.maxDistance = 80.0;
```

Para la creación del fondo en sus 360° se usa una imagen de galaxia la cual esta ubicada en la carpeta de textures/FONDO2/ se hace su respectiva carga para su uso.

```
     var fondo1 = new THREE.CubeTextureLoader()
        .setPath('textures/FONDO2/')
        .load([  'px.png', 'nx.png',  'py.png', 'ny.png',  'pz.png', 'nz.png']);
        fondo1.format=THREE.RGBFormat;
        scene.background=fondo1;
```

Se crean 4 luces direccionales hacia la esfera central(mundo) con la siguiente estructura cambiamos únicamente la ubicación estas.

```
    var directionalLight1 = new THREE.DirectionalLight(0xffffff, 0.5);
    directionalLight1.position.set(-1, 1, 1).normalize();
    scene.add(directionalLight1);
```

Se establecen Materiales para la estructura de los grafos.

```
var linematerial = new THREE.LineBasicMaterial({ color: 0x0000ff });
var material2 = new THREE.MeshLambertMaterial( { color:0xE01B22,
    metalness: 0.5, roughness: 0.1, transparent:false, opacity: 0.5  } );
```

Creamos un parent para poder proporcionar un conjunto de propiedades y métodos para manipular objetos en el espacio 3D. Luego se le da inicio a un ciclo para la creación de los nodos llamando a la estructura que hicimos al inicio del código, y se crea la visualización de cada uno de estos nodos.

```
var parent = new THREE.Object3D();
            for(var i=0; i<nodes.length; i++){
var sphereGeometry=new THREE.CylinderGeometry(0.5,0.5,nodes[i].contagiados/6000,32);
sphereGeometry.rotateZ(Math.PI/2);
var sphere = new THREE.Mesh( sphereGeometry );
sphere.applyMatrix(new THREE.Matrix4().makeTranslation(nodes[i].x+0.2, nodes[i].y,
nodes[i].z));
                sphere.material = material2;
                parent.add(sphere);
            }
```

Se le da inicio a otro ciclo para la creación de las aristas llamando a la estructura que hicimos al inicio del código, y se crea la visualización de cada uno de estos caminos o aristas.

```
for(var i=0; i<edges.length; i++){
                var points = [];
    points.push( new THREE.Vector3( nodes[edges[i].from].x,
                                    nodes[edges[i].from].y,
                                    nodes[edges[i].from].z ) );
    points.push( new THREE.Vector3( nodes[edges[i].to].x,
                                    nodes[edges[i].to].y,
                                    nodes[edges[i].to].z ) );
var geometry = new THREE.BufferGeometry().setFromPoints( points );
               var line = new THREE.Line( geometry, linematerial );
               parent.add(line);
            }
            scene.add( parent );
```

Creamos un base_globe para poder proporcionar un conjunto de propiedades y métodos para manipular objetos en el espacio 3D, añadimos texturas de normales y un UVMapping a la esfera se hace una repetición de 5x5 de esta y de la normal de 2000x2000 tal vez un poco exagerado agregamos al objeto 3D la esfera con su respectivo material el cual junta estas texturas y aspectos de transparencia, opacidad entre otros.

```
  base_globe = new THREE.Object3D();
  base_globe.scale.set(20, 20, 20);
  scene.add(base_globe);

//AÑADIR TEXTURAS A ESFERA
   var Normalagua=new THREE.TextureLoader().load('textures/Normal_agua.jpg');
mar=THREE.ImageUtils.loadTexture('textures/mar.jpg',THREE.UVMapping,function(){
        mar.wrapS = THREE.RepeatWrapping;
        mar.wrapT = THREE.RepeatWrapping;
        mar.repeat.set(5, 5);
```

```
        Normalagua.wrapS = THREE.RepeatWrapping;
        Normalagua.wrapT = THREE.RepeatWrapping;
        Normalagua.repeat.set(2000, 2000);
        base_globe.add(new THREE.Mesh(
        new THREE.  SphereGeometry(radius, 32, 32),
        new THREE.MeshPhongMaterial({
        //normalMap:Normalagua,
        transparent: true,
        depthTest: true,
        depthWrite: false,
        opacity: 0.98,
        map: mar,
        color: 0x6699ff,
        })));
```

Se uso una función originalmente de https://github.com/udmani/tessalator/ el cual llama a los vértices de cada país crea caras y luego geometrías mapeando sobre la esfera ya que los vértices de cada país están en coordenadas cartesianas en 2d el archivo que contiene todos estos vértices se llama country_data.js este esta organizado por continentes y su respectivos países aclarando esto se crea un ciclo para la visualización de cada continente y sus respectivos países, con colores aleatorios para cada país, y se mapea una textura en estos todo esto se añade a el objeto 3D base_globe.

```
  for (var name in country_data) {
            geometry = new Tessalator3D(country_data[name], 0);
            var continents = ["EU", "AN", "AS", "OC", "SA", "AF", "NA"];

//GENERA COLORES ALEATORIOS A CADA PAIS
            var color = new THREE.Color();
            color.setRGB(0, Math.random(nodes[i].contagiados/999999), 0);
            var mesh = country_data[name].mesh = new THREE.Mesh(geometry, new THREE.M
eshPhongMaterial({
                normalMap:Normalagua,
                 color: color,
                 map: mar,
            }));
            mesh.name = "land";
            mesh.userData.country = name;
            base_globe.add(mesh);
        }
    });

    document.addEventListener('mousemove', onDocumentMouseMove, false);

    camera.position.z = -40;
    camera.position.x = 80;
    camera.position.y =0;
}
```

En esta función se usa el ratón, esto para comodidad de la interfaz con el usuario se escala al intersecar algún objeto en este caso con los países se hace uso del raycaster

para saber si se hace intersección y de ser así se llama el nombre de cada país establecido en country_data.js

```
function onDocumentMouseMove(event) {
    if (intersected_object !== 0) {
        intersected_object.scale.set(1.0, 1.0, 1.0);
    }

    event.preventDefault();
    var mouseX = (event.clientX / window.innerWidth) * 2 - 1;
    var mouseY = -(event.clientY / window.innerHeight) * 2 + 1;
    var vector = new THREE.Vector3(mouseX, mouseY, -1);
    vector.unproject(camera);
    var raycaster = new THREE.Raycaster(camera.position, vector.sub(camera.position).
normalize());
    var intersects = raycaster.intersectObject(base_globe, true);
    if (intersects.length > 0) {
        if (intersects[0].point !== null) {
            if (intersects[0].object.name === "land") {
            console.log(intersects[0].object.userData.country);
                if (overlay_element === 0) {
                    overlay_element = document.getElementById("overlay");
                }
                overlay_element.innerHTML = intersects[0].object.userData.country;

                intersects[0].object.scale.
                set(hover_scale, hover_scale, hover_scale);
                intersected_object = intersects[0].object;
            }}}}
```

Se crea una función animación para actualizar controles y animar frames, se llama a la función render

```
function animate() {
            requestAnimationFrame(animate);
            controls.update();
            render();
        }
```

Se crea un renderizado encargado la escena y la cámara.

```
 function render(){
          renderer.render(scene, camera);
         }
```

VISUALIZACIÓN DE PASOS GRAFICOS:

Ilustración 4 Creación Grafos

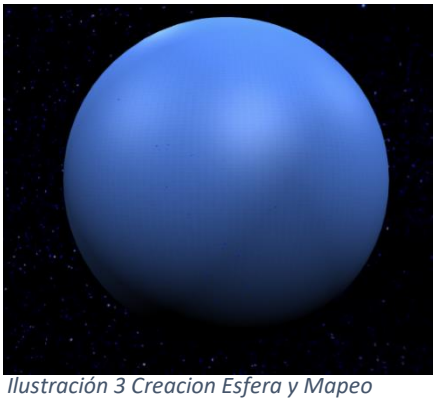

Ilustración 3 Creacion Esfera y Mapeo

Ilustración 2 Geometrias Mundo Mapeadas

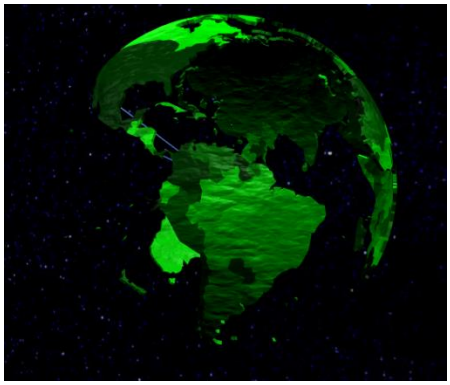

Ilustración 1 Textura en países

Ilustración 6 Unión esfera y países

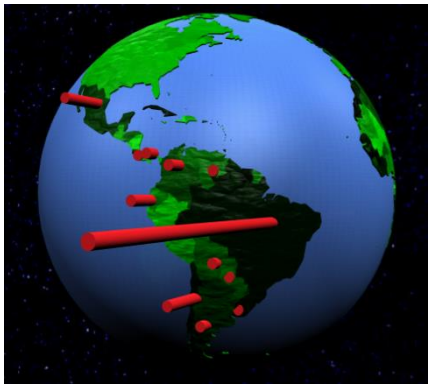

Ilustración 7 Visualización con Datos

Ilustración 8 Interfaz