

Proyecto

CENTRO
COMERCIAL

Manual Técnico

Grupo. 03

Equipo 6

Integrantes:

- Hernández Nava Luisa Fernanda
- Martínez Romero Alejandro Javier
- Sánchez del Valle Escanero Alfonso



No. cuenta: 317333613

**Manual técnico de Centro Comercial
Gran Casa**

ÍNDICE

- 1. Objetivos**
- 2. Diagrama de Gantt**
- 3. Alcance del**
- 4. Limitantes**
- 5. Documentación del código**
- 6. Conclusiones.**

Manual técnico

Centro Comercial - Gran Casa

Enlace a repositorio:

<https://github.com/Alejandro-alt/ProyectoFinal-2024-GPO3-EQUIPO6.git>

Objetivos:

Emplear los conocimientos adquiridos en el laboratorio, así como el mejor uso de los programas y métodos usados desde el inicio del laboratorio, entender el uso de las herramientas para mejorar el modelado y entender la parte teórica, y su correcto uso y empleamiento.

Diagrama de Gantt

Cronograma de Actividades				
Actividades	Mes de realización (entregas / avances)			
	Marzo	Abril	Mayo	
Planeación de proyecto				
Busqueda de ideas para definir tematica de cada sala	■			
Ubicación geometrica de cada sala dentro del centro comercial	■			
Analisis de costos/presupuestos		■		
Propuesta de animaciones	■			
Busqueda de objetos a incluir en cada sala		■		
Busqueda de animaciones a incluir		■		
Modelado de area de la fachada			■	
Incluir textura en la fachada			■	
Busqueda de modelos/ objetos a animar				
Modelado de sala 1				■
Modelado de sala 2			■	
Modelado de sala 3			■	
Busqueda de texturas de los objetos de las animaciones				■
Texturizar animaciones			■	
Realización de 2 animaciones(movimientos)				■
Realización de 3 animaciones(movimientos)			■	
Modelado de area exterior del centro				■
Revision de detalles de cada sala				
Revisión de detalles de las animaciones				
Probar funcionamiento de todo el proyecto				
Realizar manual de usuario				■
Entrega de proyecto				■

Estimación de costos y precio de venta del proyecto

El equipo de este proyecto se compone por los siguientes roles:

- Líder de proyecto
- Programador
- Diseñador

Puesto	Sueldo por dia	Horas de trabajo	Días de trabajo	Fechas	Total
Líder de proyecto	\$1400	6	40	19 de marzo a 17 de mayo	\$56,000
Diseñador	\$700	6	30	19 de marzo a 17 de mayo	\$21,000
Programador	\$750	6	30	19 de marzo a 17 de mayo	\$22,500
Total					\$99,500

Costos Fijos (Servicios)		
Servicio	Costo por mes	Costo por duración del proyecto
Luz	\$200	\$1,800
Agua	\$200	\$1,800
Internet	\$400	\$3,600
Total:	\$800	\$7,200

Costos Variables		
Consumibles	Costo por mes	Costo por duración del proyecto
Licencia 3DMax	\$2,913	\$5,826
Licencia Visual S.C	-	-
Gimp	-	-
Total:	\$2,913	\$5,826

Cronograma de actividades

Link:

https://docs.google.com/spreadsheets/d/1NGM-Mk_u86Orwwi2Z7K75VsB1kvbPAdIQXIEgqJd84/edit?usp=sharing

Alcance del proyecto

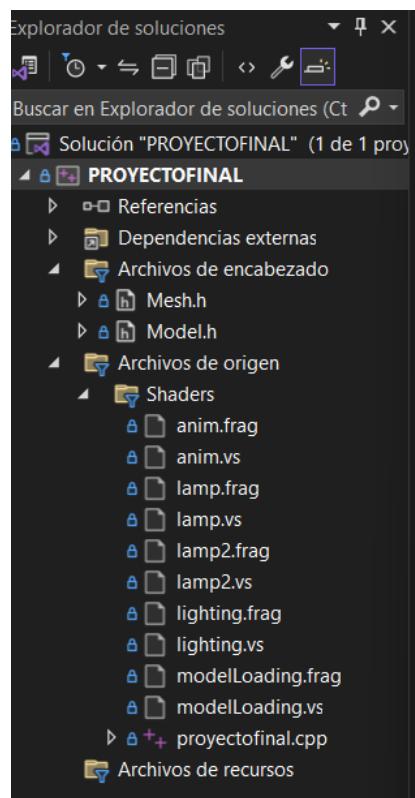
- Carga y visualización de modelos 3D en open gl
- Implementar diferentes animaciones usando lo visto en laboratorio
- Implementar transparencias a objetos
- Activar canal alpha a texturas y escalar
- Saber utilizar la estructura de código para usar transparencias y movimiento del agua
- Saber el orden de los los objetos a imprimir, si se desea transparencias.
- Aplicar transformaciones y acomodar el pivote de los objetos a animar, segun sea el caso.
- Aplicación de iluminación, así como el área que se desea abarcar.

Limitantes

- Algunas de las limitantes de encontre a realizar el proyecto fueron las siguientes:
- Algunos modelos contienen mucha geometría, lo cual ocasiona que sean muy pesados y en mi caso el equipo se trabó varias veces.
- Algunos modelos eran de paga

Documentación del código

Hacemos la carga y creación de nuestro cpp y de los códigos a usar, así como de nuestros shaders y demás elementos a usar.



Llamada de los shaders correspondientes

```
198
199
200     Shader lightingShader("Shaders/lighting.vs", "Shaders/lighting.frag");
201     Shader lampShader("Shaders/lamp.vs", "Shaders/lamp.frag");
202     Shader Anim("Shaders/anim.vs", "Shaders/anim.frag");
203
204 // CARGA DE MODELOS CENTRO COMERCIAL FACHADA-----
```

Variables a usar en mis animaciones / en ocasiones volvi a ocupar las mismas variables

```
// Light attributes
glm::vec3 lightPos(0.0f, 0.0f, 0.0f);
bool active;
float tiempo; // Variable para controlar el tiempo

//variable controlar
int currentLightIndex = 0;

//animaciones
float rot = 0.0f;
float rot1 = 0.0f;
float rot2 = 0.0f;
float rot3 = 0.0f;

//vidrio deslizante
float des = 0.0f;
float des1 = 0.0f;
bool anim4 = false;

//animacion perro
float CaballoTorso;
float CaballoTorso_offset;
float CaballoPatasT;
float CaballoPatasT_offset;
float CaballoPatasD;
float CaballoPatasD_offset;
float CaballoCabeza;
float CaballoCabeza_offset;

//camara
bool anim = false;
bool anim1 = false;
bool anim2 = true;
bool anim3 = false;
```

```
static bool direccion = true;
```

Carga de modelos de mi proyecto a usar

```
// CARGA DE MODELOS CENTRO COMERCIAL  
FACHADA-----
```

```
Model centrofachada((char*)"Models/centrofachada/centrofachada.obj");
```

```
/// CARGA DE MODELOS CENTRO COMERCIAL  
FACHADA-----
```

```
Model interior((char*)"Models/cafe/interior.obj");
```

```
Model interiorvidrio((char*)"Models/cafe/interiorvidrio.obj");
```

```
Model exterior((char*)"Models/cafe/exterior.obj");
```

```
Model hoja((char*)"Models/plantas/hoja.obj");
```

```
Model hojacom((char*)"Models/plantas/hojacom.obj");
```

```
Model hojatallo((char*)"Models/plantas/hojatallo.obj");
```

```
Model pasto((char*)"Models/plantas/pasto.obj");
```

```
//Model pasto2((char*)"Models/plantas/pasto2.obj");
```

```
Model patasade((char*)"Models/perro/patasade.obj");
```

```
Model patastra((char*)"Models/perro/patastra.obj");
```

```
Model torso((char*)"Models/perro/torso.obj");
```

```
///-----
```

```
Model cine((char*)"Models/cine/cine.obj");
```

```
Model sala((char*)"Models/cine/sala.obj");
```

```
Model sala1((char*)"Models/cine/sala1.obj");
```

```
Model anicarro((char*)"Models/anicarro/anicarro.obj");
```

```
Model anicarro2((char*)"Models/anicarro2/anicarro2.obj");
```

```
// CARGA DE MODELOS JUGUETERIA-----
```

```
Model Piso((char*)"Models/Piso/Piso.obj");
```

```
Model fachada((char*)"Models/fachada/fachada.obj");//toda la fachada
```

```
Model loquer((char*)"Models/loquer/loquer.obj");//loquer de entrada
Model mueble((char*)"Models/mueble/mueble.obj");// mueble de izquierda
Model mueble2((char*)"Models/mueble/mueble2.obj");// mueble de fondo
Model mueble3((char*)"Models/mueble/mueble3.obj");// mueble de derecha
Model sillón((char*)"Models/sillón/sillón.obj");// 
Model mesa((char*)"Models/mesa/mesa.obj");// mesa de entrada
Model mesa3((char*)"Models/mesa3/mesa3.obj");// mesa de entrada
Model puerta((char*)"Models/puerta/puerta.obj");//agregado
Model puerta2((char*)"Models/puerta/puerta2.obj");//agregado
Model puertac1((char*)"Models/puerta/puertac1.obj");//agregado
Model puertac2((char*)"Models/puerta/puertac2.obj");//agregado
Model puertaarriba((char*)"Models/puerta/puertaarriba.obj");//agregado
Model vidriod((char*)"Models/vidrio/vidriod.obj");//agregado
Model vidrioi((char*)"Models/vidrio/vidrioi.obj");//agregado
Model carro3VIDRIO((char*)"Models/carro3/carro3VIDRIO.obj");//agregado
Model escaleras((char*)"Models/escaleras/escaleras.obj");//agregado
```

//////7objetos-----

```
Model carro5lego((char*)"Models/carro5lego/carro5lego.obj");// carro colores
```

```
Model oso((char*)"Models/oso/oso.obj");// oso rojo
Model oso2((char*)"Models/oso/oso3.obj");// oso azul
Model osos((char*)"Models/osos/osos.obj");// oso azul
```

```
Model balon((char*)"Models/balon/balon.obj");// 3 balones de basquet
Model balonitalia((char*)"Models/balonitalia/balonitalia.obj");// 3 balones de basquet
Model pelotafut((char*)"Models/pelotafut/pelotafut.obj");// 3 balones de basquet
Model Brazuka((char*)"Models/brazuk/Brazuka.obj");// 3 balones de basquet
```

```
Model casa2((char*)"Models/casa2/casa2.obj");// casita MIA
```

```
Model car1((char*)"Models/car/car1.obj");// carro colores
Model car2((char*)"Models/car/car2.obj");// carro colores
Model carro3((char*)"Models/carro3/carro3.obj");// carro negro mio
Model carro4madera((char*)"Models/carro4madera/carro4madera.obj");// carro negro mio
```

```
Model piano1((char*)"Models/piano/piano1.obj");// piano
Model piano2((char*)"Models/piano/piano2.obj");//
```

```
Model sonaja2((char*)"Models/sonaja2/sonaja2.obj");//
```

////EXTRA-----

```
Model caballo2((char*)"Models/caballo2/caballo2.obj");//  
Model alfombra((char*)"Models/alfombra/alfombra.obj");//  
Model camara((char*)"Models/camara/camara.obj");//  
Model camara2((char*)"Models/camara/camara2.obj");//
```

```
Model juguetamarillo((char*)"Models/juguetamarillo/juguetamarillo.obj");//  
Model mario((char*)"Models/mario/mario.obj");//  
Model osocar((char*)"Models/osocar/osocar.obj");//  
Model varios((char*)"Models/varios/varios.obj");//osos  
Model planta((char*)"Models/planta/planta.obj");//  
Model caja((char*)"Models/caja/caja.obj");//  
Model mesa4((char*)"Models/mesa4/mesa4.obj");//
```

```
Model lego1y2((char*)"Models/legos/lego1y2.obj");//  
Model lego3y4((char*)"Models/legos/lego3y4.obj");//  
Model lego5y6y7((char*)"Models/legos/lego5y6y7.obj");//  
Model pecera((char*)"Models/pecera/pecera.obj");//  
Model peceravidrio((char*)"Models/pecera/peceravidrio.obj");//  
Model agua((char*)"Models/pecera/agua.obj");//
```

```
Model lampara1((char*)"Models/lampara/lampara1.obj");//
```

Incluimos nuestros comandos de iluminación a usar, así como el shader

```
//Load Model
// Use cooresponding shader when setting uniforms/drawing objects
lightingShader.Use();

glUniform1i(glGetUniformLocation(lightingShader.Program, "material.diffuse"), 0);
glUniform1i(glGetUniformLocation(lightingShader.Program, "material.specular"), 1);
GLint viewPosLoc = glGetUniformLocation(lightingShader.Program, "viewPos");
glUniform3f(viewPosLoc, camera.GetPosition().x, camera.GetPosition().y, camera.GetPosition().z);

// Directional light AMBIENTACION
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.direction"), -0.2f, -1.0f, -0.3f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.ambient"), 0.9f, 0.9f, 0.9f); //ambiente
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.diffuse"), 0.2f, 0.2f, 0.2f); //ambiente
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.specular"), 0.8f, 0.8f, 0.8f);

// Point light 1
glm::vec3 lightColor1;
lightColor1.x = Light1.x;
lightColor1.y = Light1.y;
lightColor1.z = Light1.z;

glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].position"), 9.12f, 13.335f, 13.533f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].ambient"), lightColor1.x, lightColor1.y, lightColor1.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].diffuse"), lightColor1.x, lightColor1.y, lightColor1.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].specular"), 1.0f, 1.0f, 1.0f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[0].constant"), 1.0f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[0].linear"), 0.35f); //ilu sobre superficie
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[0].quadratic"), 0.44f); //ILUMInacion sobre superficie

// Point light 2
glm::vec3 lightColor2;
lightColor2.x = Light2.x;
lightColor2.y = Light2.y;
lightColor2.z = Light2.z;

glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[1].position"), -20.446f, 6.227f, 10.998f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[1].ambient"), lightColor2.x, lightColor2.y, lightColor2.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[1].diffuse"), lightColor2.x, lightColor2.y, lightColor2.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[1].specular"), 1.0f, 1.0f, 1.0f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[1].constant"), 1.0f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[1].linear"), 0.7f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[1].quadratic"), 1.8f);

// Point light 3
glm::vec3 lightColor3;
lightColor3.x = Light3.x;
lightColor3.y = Light3.y;
lightColor3.z = Light3.z;

glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[2].position"), -5.876f, 6.227f, 10.998f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[2].ambient"), lightColor3.x, lightColor3.y, lightColor3.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[2].diffuse"), lightColor3.x, lightColor3.y, lightColor3.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[2].specular"), 1.0f, 1.0f, 1.0f);
```

💡 No se encontraron problemas. | 🔪 Línea: 401 Carácter: 25 Columna: 31 MIXTO

Aplicar transparencias // Para este punto es importante activar y al final desactivar el canal alpha, y recordar que estos objetos se colocan al último

//TRANSPARENCIAS DE VIDRIO DE MUEBLE-----

```
glEnable(GL_BLEND); //Activa la funcionalidad para trabajar el canal alfa
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA); //operaciones con canal alpha
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(0.0f, 0.0f, des));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glUniform1i(glGetUniformLocation(lightingShader.Program, "activaTransparencia"), 0);
glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlpha"), 1.0, 1.0, 1.0, 0.35);
vidriod.Draw(lightingShader);
glDisable(GL_BLEND); //Desactiva el canal alfa
glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlpha"), 1.0, 1.0, 1.0, 1.0);
```

Después de esto, seguimos con la llamada de nuestros modelos, en mi caso fui asignando un comentario a cada uno, por cada fachada que realizaba para tener un mejor orden.

```
//FACHADACENTROCOMERCIAL-----
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
centrofachada.Draw(lightingShader);

model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
hojacom.Draw(lightingShader);

//CAFE INTERIOR Y EXTERIOR-----
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
exterior.Draw(lightingShader);

model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
interior.Draw(lightingShader);

//-----
//perro

// Torso
model = glm::mat4(1.0f);
modelaux = glm::mat4(1.0f);
//model = glm::translate(model, glm::vec3(170.0f, 6.2f, (-170.0f + CaballoTorso)));
modelaux = model;
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
//CaballoTorso_M.RenderModel();
torso.Draw(lightingShader);

// Dato: Transparency
```

```

//CINE-----
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
cine.Draw(lightingShader);

model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
sala.Draw(lightingShader);

model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
salal.Draw(lightingShader);

model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
anicarro.Draw(lightingShader);

model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
anicarro2.Draw(lightingShader);

////JUGUETERIA
///Carga de modelos-----

model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
Piso.Draw(lightingShader);

//fachada
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
fachada.Draw(lightingShader);

//loquer DE ENTRADA
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
loquer.Draw(lightingShader);

//escaleras
model = glm::mat4(1);

```

Muy importante mencionar en que orden llamamos a nuestros modelos, ya que si queremos obtener una transparencias, esto va a influir mucho, ya que puede que no la obtengamos

```

//TRANSPARENCIAS DE PUERTA-----
 glEnable(GL_BLEND); //Activa la funcionalidad para trabajar el canal alfa
 glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA); //operaciones con canal alpha
 model = glm::mat4(1);
 glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
 glUniform1i(glGetUniformLocation(lightingShader.Program, "activaTransparencia"), 0);
 glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlpha"), 1.0, 1.0, 1.0, 0.65);
 puerta.Draw(lightingShader);
 glDisable(GL_BLEND); //Desactiva el canal alfa
 glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlpha"), 1.0, 1.0, 1.0, 1.0);

 glEnable(GL_BLEND); //Activa la funcionalidad para trabajar el canal alfa
 glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA); //operaciones con canal alpha
 model = glm::mat4(1);
 glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
 glUniform1i(glGetUniformLocation(lightingShader.Program, "activaTransparencia"), 0);
 glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlpha"), 1.0, 1.0, 1.0, 0.65);
 puerta2.Draw(lightingShader);
 glDisable(GL_BLEND); //Desactiva el canal alfa
 glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlpha"), 1.0, 1.0, 1.0, 1.0);

```

Como última parte el shader para animar nuestra agua, así como también nuestras plantitas de la entrada que dan la impresión de que hay aire.

```
//// Shader para simular el agua
Anim.Use();
glEnable(GL_BLEND); //Activa la funcionalidad para trabajar el canal alfa
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA); //operaciones con canal alpha
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glUniform1i(glGetUniformLocation(Anim.Program, "activaTransparencia"), 0);
glUniform4f(glGetUniformLocation(Anim.Program, "colorAlpha"), 1.0, 1.0, 1.0, 0.15);
tiempo = glfwGetTime() * 3.0f; // Definimos la intensidad del oleaje.
//glUniform1f(glGetUniformLocation(Anim.Program, "time"), tiempo);
modelLoc = glGetUniformLocation(Anim.Program, "model");
viewLoc = glGetUniformLocation(Anim.Program, "view");
projLoc = glGetUniformLocation(Anim.Program, "projection");
glUniformMatrix4fv(viewLoc, 1, GL_FALSE, glm::value_ptr(view));
glUniformMatrix4fv(projLoc, 1, GL_FALSE, glm::value_ptr(projection));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glUniform1f(glGetUniformLocation(Anim.Program, "time"), tiempo);

agua.Draw(lightingShader);
glDisable(GL_BLEND); //Desactiva el canal alfa
glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlpha"), 1.0, 1.0, 1.0, 1.0);

Anim.Use();

model = glm::mat4(1);
```

Todas nuestras animaciones las colocamos dentro de nuestro DoMovement

```
//-----
```

```
//ANIMACION PARA PUERTAS DE ENTRADA AMBAS-----
```

```
if (anim) {
    // Rotar puertac1 hacia afuera
    if (rot < 90.0f) {
        rot += 0.4f;
    }
    // Rotar puertac2 hacia adentro
    if (rot1 > -90.0f) {
        rot1 -= 0.4f;
    }
}
else {
    // Regresar puertac1 a 0
    if (rot > -0.0f) {
        rot -= 0.4f;
    }
}
```

```
}

// Regresar puertac2 a 0
if (rot1 < 0.0f) {
    rot1 += 0.4f;
}
}

//ANIMACION PARA CAJUELA-----
if (anim1) {
    // Rotar puertac1 hacia afuera
    if (rot2 < 60.0f) {
        rot2 += 0.4f;
    }
}

}

else {
    // Regresar puertac1 a 0
    if (rot2 > -0.0f) {
        rot2 -= 0.4f;
    }
}

}

//ANIMACION PARA CAMARA-----
if (anim2)
{
    if (direccion)
    {
        rot3 += 0.1f;
        if (rot3 >= 90.0f)
            direccion = false;
    }
    else
    {

```

```
    rot3 -= 0.1f;
    if (rot3 <= -5.0f)
        direccion = true;
    }
}
```

//ANIMACION PARA VIDRIO PUERTA DERECHA-----

```
if (anim3) {
    // Rotar puertac1 hacia afuera
    if (des < 3.9f) {
        des += 0.01f;
    }
}
```

}

```
else {
    // Regresar puertac1 a 0
    if (des > -0.0f) {
        des -= 0.01f;
    }
}
```

}

//ANIMACION PARA VIDRIO PUERTA izquierda-----

```
if (anim4) {
    // Rotar puertac1 hacia afuera
    if (des1 < 4.6f) {
        des1 += 0.01f;
    }
}
```

}

```
else {
    // Regresar puertac1 a 0
    if (des1 > -0.0f) {
```

```
    des1 -= 0.01f;  
}  
  
}
```

Con estas animaciones se obtienen resultados como los siguientes.
desde abrir una puerta, rotar un objeto, rotar una camara, incorporar iluminación y tambien recrear el viento que pasa sobre una hoja y movimiento que tienen tanto el fallo, como las hojas.

Nosotros manejamos las activaciones de algunas animaciones por medio de nuestro teclado numérico y asi es como lo asignamos.

```
//puertas  
if (keys[GLFW_KEY_2])  
{  
    anim = !anim;  
  
}  
//cajuela  
if (keys[GLFW_KEY_3])  
{  
    anim1 = !anim1;  
  
}  
//camara(es automaitco)  
if (keys[GLFW_KEY_4])  
{  
    anim2 = true;  
  
}  
  
//vidrio dezlizante  
if (keys[GLFW_KEY_5])
```

```
{  
    anim3 = !anim3;
```

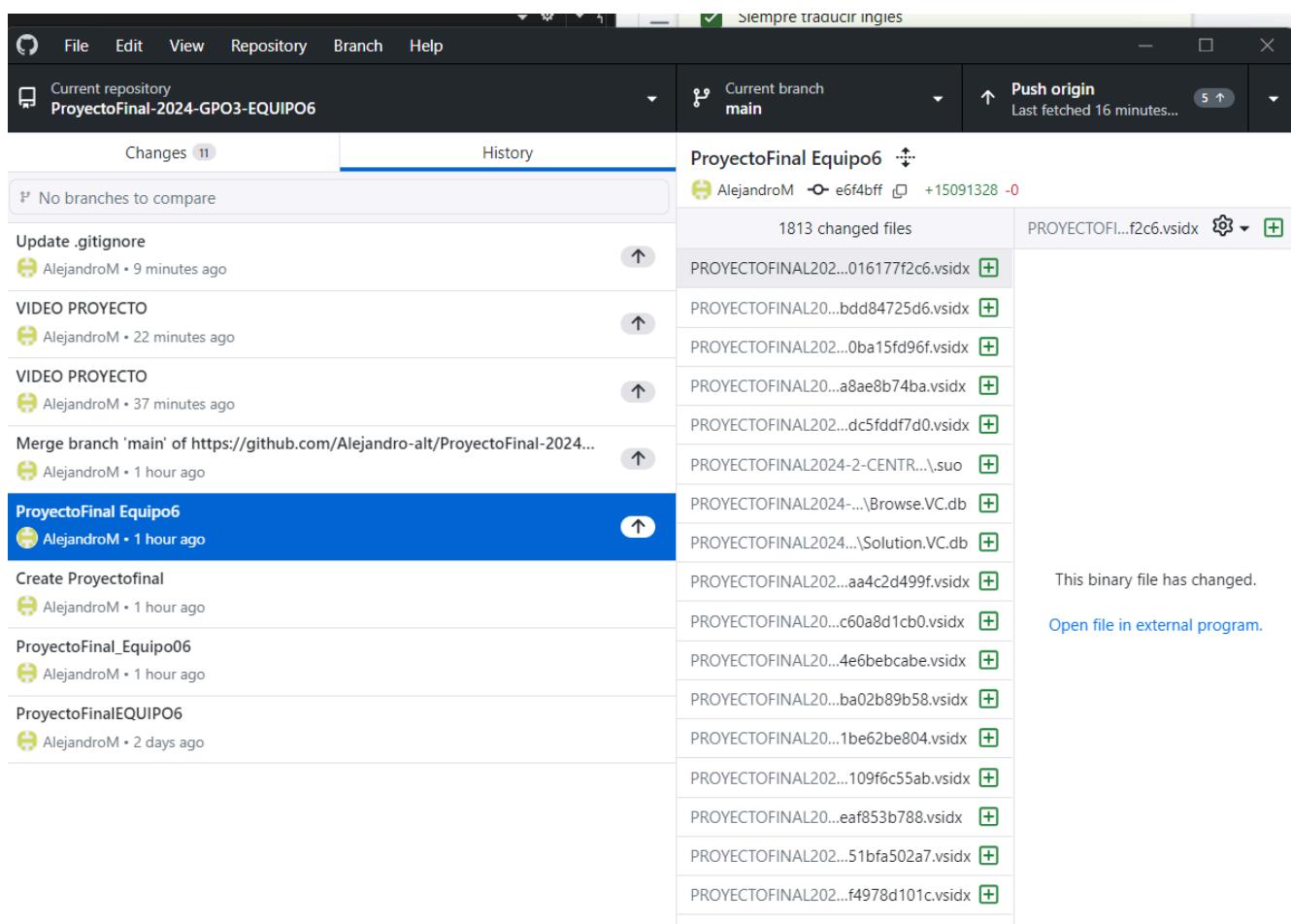
```
}
```

```
//vidrio dezlizante
```

```
if (keys[GLFW_KEY_6])  
{  
    anim4 = !anim4;
```

```
}
```

Repositorio local



Conclusiones.

En este proyecto aprendí a texturizar mejor, me costó mejorar la geometría de unos modelos, experimenté varios errores con las texturas, el tamaño, o en las transformaciones sencillas, ya que al momento de hacer animaciones no aplicaba bien las transformaciones o simplemente coloque mal el pivote.

Fue un proyecto que exige mucho de mi, ya que fui tomando más práctica en el manejo de las herramientas usadas y así poder disminuir varios errores.