



Universidad Nacional Autónoma de México
Facultad de Ingeniería
División de Ingeniería Eléctrica



Carrera: Ingeniería en Computación

Curso: Computación Gráfica e Interacción Humano-Computadora

(Clave: 1590)

Proyecto 2. Final

Profesor: Ing. Carlos Aldair Roman Balbuena

Alumno:

- 318192709 -

Grupo: 05

Fecha de entrega: 20/Mayo/25

Semestre: 2025-2

Índice

1. Introducción.....	1
2. Justificación del proyecto.....	2
3. Objetivo.....	2
4. Metodología.....	3
5. Diagrama de Gantt.....	5
6. Imágenes de referencia.....	8
7. Manual técnico.....	10
○ 7.1 Objetivo técnico y alcances del sistema.....	10
○ 7.2 Modelado.....	11
○ 7.3 Texturizado y mapeado UV.....	21
○ 7.4 Herramientas, librerías y estructura del código.....	28
○ 7.5 Carga de modelos y texturas en el sistema.....	30
○ 7.6 Implementación de la cámara sintética.....	37
○ 7.7 Animaciones.....	38
○ 7.8 Carga e implementación de sonido.....	41
○ 7.9 Diagrama de flujo del sistema.....	42
○ 7.10 Diccionario de funciones y variables clave.....	43
8. Manual de usuario.....	46
- 8.1. Ejecutable.....	47
9. Análisis de costos.....	62
10.Repositorio en GitHub.....	65
11.Video demostrativo Fachada.....	65
12.Resultado final.....	66
13.Conclusiones finales.....	72

1. INTRODUCCIÓN

El presente documento forma parte del proyecto final de la asignatura de Teoría, cuyo propósito fue diseñar, modelar y animar un espacio virtual a través de herramientas de diseño 3D, particularmente Blender, e integrarlo posteriormente en un entorno programado con OpenGL. La idea central consistió en recrear un museo, el cual está dentro de un parque ecológico.

A lo largo del proyecto se abordaron distintas fases, desde la planeación temporal mediante un diagrama de Gantt, hasta la implementación técnica de modelos y animaciones, incluyendo una cámara sintética y al menos cuatro animaciones funcionales. Se trabajó en el modelado de habitaciones con un enfoque artístico coherente, recreando objetos específicos para cada espacio interior y exterior, buscando alcanzar un nivel de realismo aceptable para una experiencia atractiva.

Además, se desarrollaron recursos complementarios como un análisis de costos, documentación técnica, un manual de usuario para comprender la funcionalidad del sistema y la publicación del proyecto en un repositorio público como parte de la entrega final. Esta documentación reúne todos estos aspectos y evidencia el proceso de desarrollo integral del proyecto.

2. JUSTIFICACIÓN DEL PROYECTO

La realización de este proyecto surge de la necesidad de aplicar de forma integral los conocimientos adquiridos en el área de gráficos computacionales, modelado 3D y programación visual. A través del desarrollo de un entorno virtual, en este caso, un museo interactivo dentro de un parque ecológico, se busca representar un espacio que pueda ser explorado por el usuario, integrando elementos de modelado, animación, interacción y lógica computacional.

Este museo ha sido concebido no solo como una experiencia visual, sino también como una representación simbólica de la cultura de Xochimilco, incorporando elementos tradicionales como las trajineras y las chinampas dentro de una sala de exposición. Además, se incluye un baño público como segundo cuarto, aportando

realismo funcional y coherencia arquitectónica al diseño. Ambos espacios permiten una interacción significativa con el entorno, al mezclar lo cultural con lo práctico.

El proyecto responde al reto de simular un entorno con un alto grado de realismo, donde se contemplan tanto aspectos técnicos (como el uso de cámaras sintéticas, materiales, texturizado y ejecución funcional en OpenGL) como artísticos (referencias visuales, narrativa espacial y estilo gráfico definido). A través del uso de Blender, se fortalece la comprensión del pipeline de creación 3D, desde el modelado hasta la animación por keyframes.

Finalmente, este proyecto también contribuye al desarrollo de competencias profesionales como la planeación estructurada (mediante el diagrama de Gantt), el control de versiones (a través de GitHub) y la documentación del proceso (manuales técnico y de usuario). Por todo esto, el proyecto representa una experiencia formativa integral, con valor académico y potencial aplicación profesional.

3. OBJETIVO

El objetivo de este proyecto es diseñar y desarrollar un museo virtual interactivo ubicado dentro del Parque Ecológico de Xochimilco, inspirado en la riqueza cultural y natural de esta zona. El entorno virtual integra técnicas de modelado 3D, animación, cámaras sintéticas y programación visual con ejecución en OpenGL. Para ello, se contemplan dos espacios principales del museo: una sala de exposición cultural y un baño público, cada uno con cinco objetos modelados según un estilo artístico definido. El proyecto busca aplicar de manera integral los conocimientos adquiridos durante el curso, abarcando desde la planeación (diagrama de Gantt) hasta su documentación completa (manual técnico y de usuario), incluyendo el análisis de costos y su publicación en un repositorio funcional en GitHub.

4. **METODOLOGÍA**

Para el desarrollo del museo virtual se utilizó una metodología incremental, la cual resultó ser la más adecuada para organizar y construir el sistema de manera progresiva, eficiente y controlada.

Este enfoque permitió dividir el trabajo en etapas consecutivas, donde cada módulo del sistema (modelado, texturizado, interacción, sonido, etc.) fue implementado y probado antes de avanzar al siguiente. Gracias a esto, fue posible detectar errores tempranos, validar el funcionamiento visual y lógico de cada componente, y aplicar mejoras sin comprometer el resto del desarrollo.

Etapas principales de implementación:

1. Planeación general del entorno
 - Se definirá la temática del museo, el espacio a modelar y la lista de objetos clave a desarrollar.
2. Modelado 3D
 - Se utilizará Blender para crear la fachada, los cuartos interiores y los objetos decorativos y funcionales del museo.
3. Texturizado
 - Se aplicarán texturas y materiales a los modelos mediante mapeo UV, usando Blender y GIMP.
4. Programación en OpenGL
 - Se configurará el entorno de desarrollo en Visual Studio Code.
 - Se integrarán librerías como GLAD, GLFW, GLM y stb_image.h.
5. Carga de modelos y texturas
 - Los modelos .fbx se exportarán desde Blender y serán cargados en el proyecto mediante clases personalizadas.

6. Implementación de cámara sintética

- Se implementará una cámara en primera persona que responde a teclado y mouse, permitiendo la exploración libre del entorno.

7. Animaciones e interacciones

- Se programarán interacciones como:
 - Puerta del museo (teclas H/J).
 - Ventiladores giratorios.
 - Puerta del baño automática por proximidad.
 - Trajinera que gira al acercarse.
 - Fantasma que aparece al hacer clic en un libro.

8. Efectos visuales y sonoros

- Se integrará un cubemap para ambientación y un shader procedural para simular el agua.
- Se añadirá música ambiental con la librería irrKlang.

9. Pruebas, validación y ajustes

- Se verificó el comportamiento de los modelos, texturas y animaciones.
- Se realizaron ajustes de posición, iluminación y escala para mejorar la experiencia visual.

10. Documentación

- Se elaboraron el manual técnico y el manual de usuario, incluyendo capturas, diagramas de flujo y análisis de costos y se exportará en github.

5. DIAGRAMA DE GANTT

Para llevar a cabo una correcta planeación del desarrollo de este proyecto, se optó por dividir el cronograma en tres diagramas de Gantt, distribuidos de forma mensual y abarcando el periodo total de trabajo comprendido entre el 20 de marzo y el 20 de mayo. Esta división permite observar con mayor claridad el avance progresivo, las fases del proceso creativo y técnico, así como la gestión del tiempo en cada etapa.

Mes Marzo

Durante el mes de marzo se llevó a cabo la fase inicial del proyecto, enfocada principalmente en la organización, conceptualización y preparación técnica necesaria para comenzar el desarrollo del museo. En esta etapa se definieron los requerimientos generales, se exploraron ideas para la fachada y se eligió el diseño definitivo que serviría como base visual del proyecto. Paralelamente, se seleccionaron los objetos y habitaciones que formarían parte del entorno, al mismo tiempo que se planificó de manera general su distribución.

[illegible]

Mes Abril

Durante el mes de abril, las actividades se enfocaron en el modelado, texturizado y organización del museo. Se inició con la fachada y los cuartos principales, y se continuó con objetos como la trajinera, esculturas, puertas y accesorios.

Inicio del proyecto	20-mar-25	PROYECTO FACHADA MUSEO PARQUE ECOLOGICO XOCHIMILCO											
Fin del proyecto	20-may-25												
ACTIVIDAD	01-abr-25	02-abr-25	03-abr-25	04-abr-25	05-abr-25	06-abr-25	07-abr-25	08-abr-25	09-abr-25	10-abr-25	11-abr-25	12-abr-25	
Modelado fachada													
Ajustes fachada													
Modelado 2 cuartos													
Modelado y textura trajinera													
Modelado y textura banca													
Modelado y textura cuadros													
Modelado y textura esculturas													
Modelado y texturaventilador													
Modelado y textura WC													
Modelado y textura espejo													
Modelado y textura lampara													
Modelado y textura puertas													
Modelado y textura lavamanos													
Acomodar cuartos con objetos													

ACTIVIDAD	13-abr-25	14-abr-25	15-abr-25	16-abr-25	17-abr-25	18-abr-25	19-abr-25	20-abr-25	21-abr-25	22-abr-25	23-abr-25	24-abr-25	25-abr-25	26-abr-25	27-abr-25	28-abr-25	29-abr-25	30-abr-25
Modelado fachada																		
Ajustes fachada																		
Modelado 2 cuartos																		
Modelado y textura trajinera																		
Modelado y textura banca																		
Modelado y textura cuadros																		
Modelado y textura esculturas																		
Modelado y texturaventilador																		
Modelado y textura WC																		
Modelado y textura espejo																		
Modelado y textura lampara																		
Modelado y textura puertas																		
Modelado y textura lavamanos																		
Acomodar cuartos con objetos																		

Mes Mayo

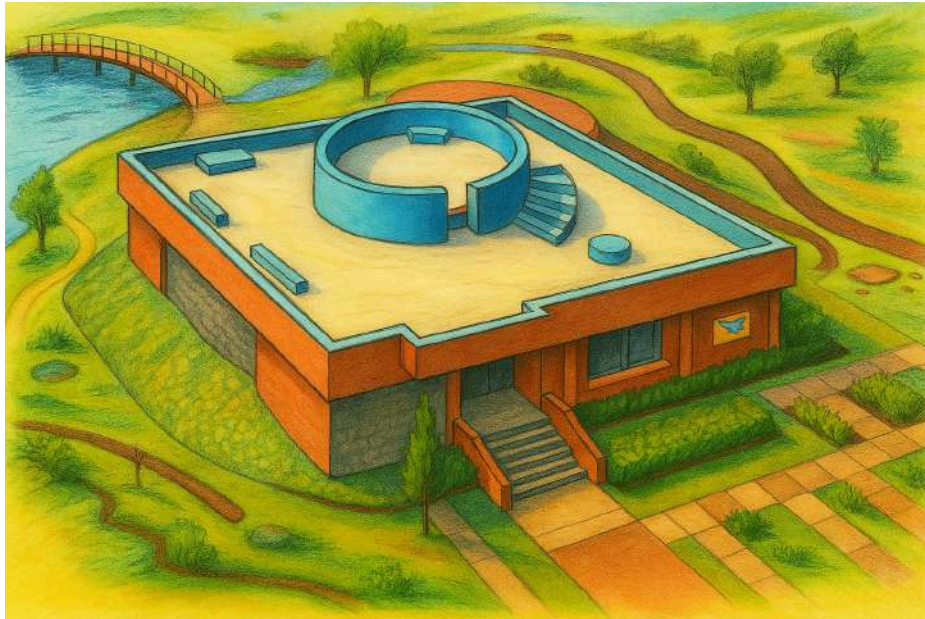
El mes de mayo muestra la fase final del desarrollo del museo virtual, centrada en la programación del código fuente, la integración de animaciones, el exportado del entorno completo a OpenGL y la generación del ejecutable funcional.

Durante este mes también se realizaron tareas clave como la creación de la cuenta en GitHub, la subida del proyecto, así como las pruebas finales y la documentación completa para entregar el proyecto en condiciones óptimas.

[illegible][illegible]

6. IMÁGENES DE REFERENCIA

Fachada



Cuarto 1 - Sala de exhibición





Elementos a recrear:

- Cuadros
- Banca
- ventilador
- trajinera
- Esculturas

Cuarto 2 - Sanitario Museo



Elementos a recrear:

- Lámparas colgantes
- lavamanos
- Puertas
- Retretes
- Espejo

7. MANUAL TÉCNICO

El presente manual técnico documenta a detalle el funcionamiento interno del proyecto “Museo Virtual del Parque Ecológico de Xochimilco”, desarrollado con tecnologías de código abierto. Su propósito es brindar una visión clara y ordenada de los componentes técnicos que integran el sistema, desde la creación de los modelos 3D hasta la implementación del código fuente, la cámara sintética, las animaciones interactivas y los efectos sonoros.

Aquí se explican tanto las herramientas utilizadas como la lógica de funcionamiento, las estructuras de código clave y los aspectos técnicos más relevantes para su correcta operación.

7.1 Objetivo técnico y alcances del sistema

El objetivo técnico principal de este proyecto fue desarrollar un entorno virtual interactivo y funcional que represente al Museo que está dentro del Parque Ecológico de Xochimilco. Se buscó no solo mostrar visualmente el espacio, sino permitir que el usuario interactúe en tiempo real con distintos elementos del entorno, integrando animaciones, detección por proximidad, cámara sintética y ambientación sonora.

Desde el punto de vista técnico, el sistema fue diseñado con los siguientes alcances:

- Permitir la visualización en tiempo real del entorno 3D a través de OpenGL.
- Implementar una cámara sintética libre controlada por teclado y mouse.

- Integrar animaciones reactivas, como puertas que se abren automáticamente, trajineras que giran al acercarse o fantasmas que aparecen al hacer clic en objetos específicos.
- Incorporar efectos sonoros ambientales para reforzar la inmersión del recorrido.
- Utilizar modelos propios, creados en Blender y exportados a formatos compatibles.

7.2 Modelado

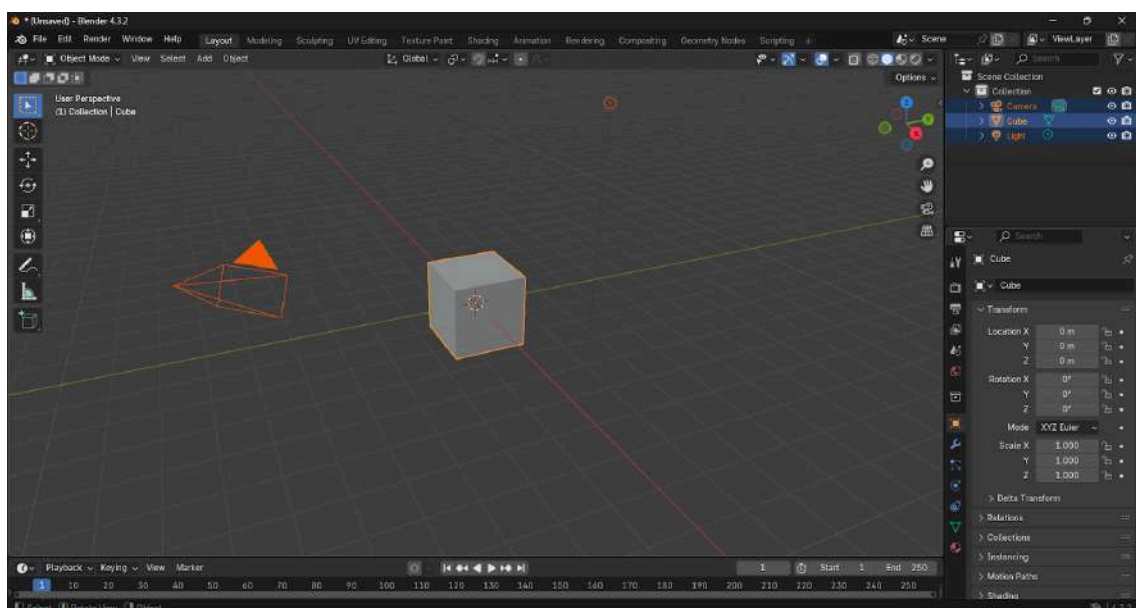
El modelado 3D del museo fue realizado utilizando el software Blender. En esta etapa se construyeron todos los elementos del entorno virtual a partir de geometría limpia, evitando intersecciones innecesarias y asegurando un orden lógico en la escena.

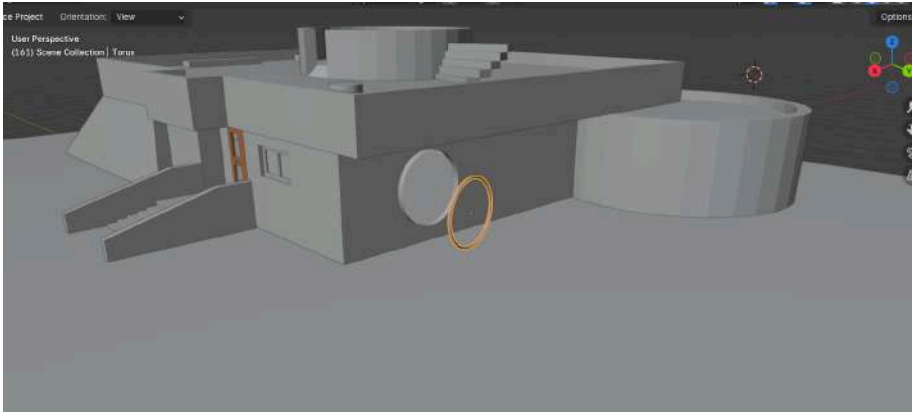
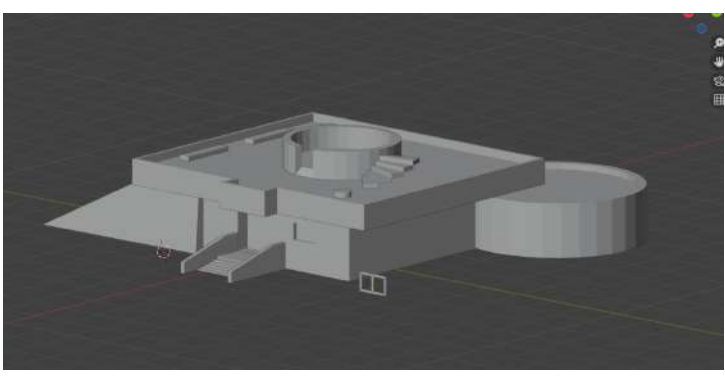
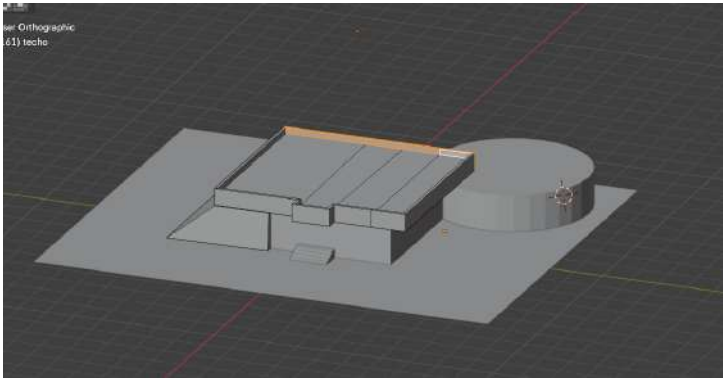
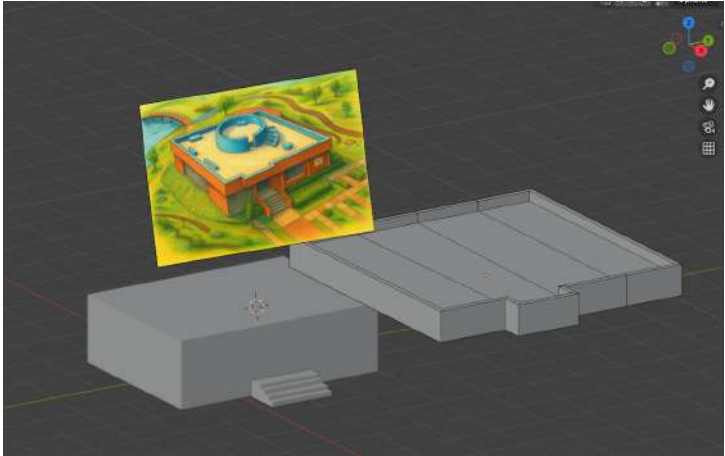
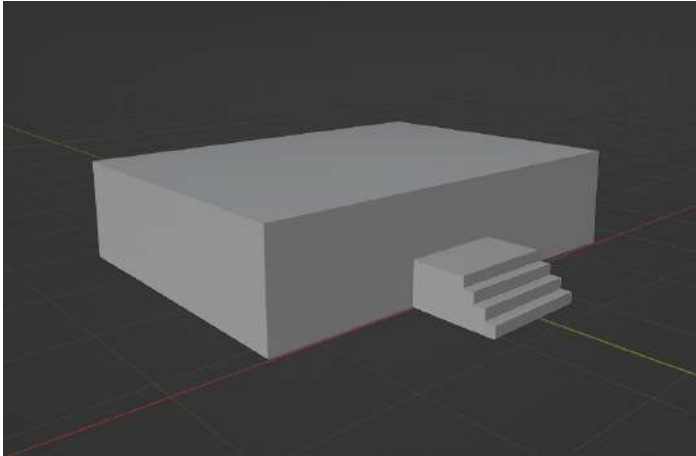
Se modelaron los siguientes componentes principales del entorno:

Diseño del museo

Fachada

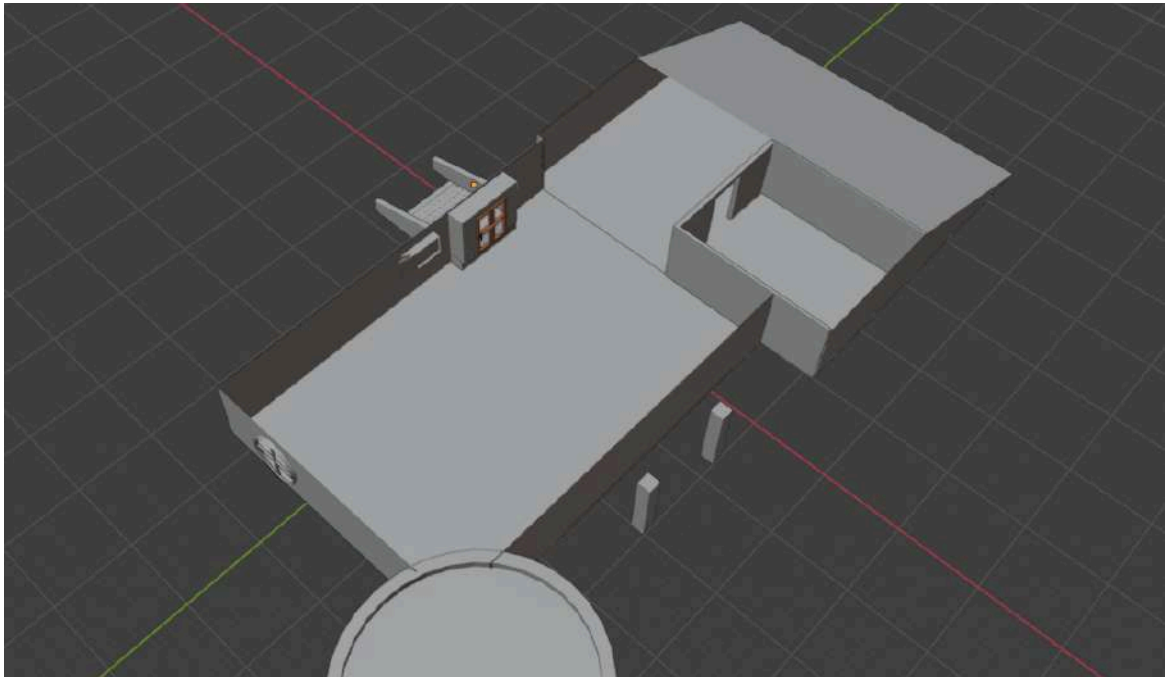
Para realizar nuestro museo, tomamos como referencia el cubo que viene por defecto al abrir el blender. Lo tomamos y lo deformamos con una forma rectangular como puede verse en las siguiente imágenes, este sería nuestro cuerpo del museo.





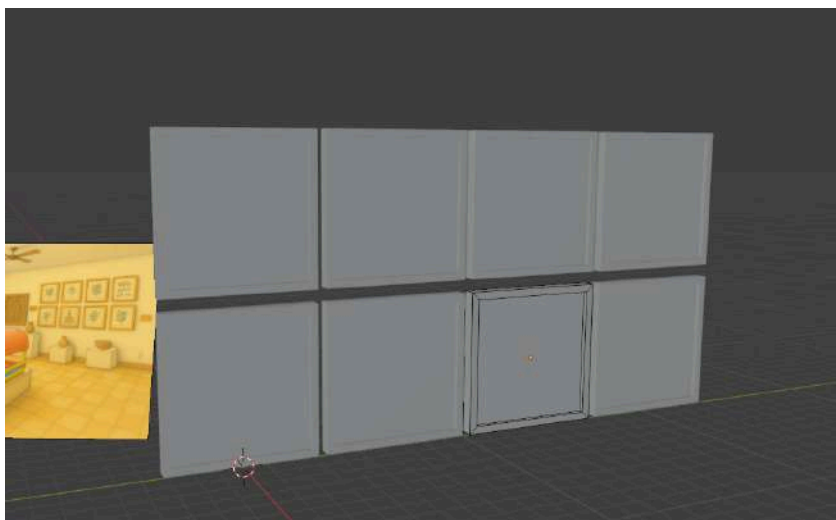
Parte interior del museo

Se puede ver la división de los 2 cuartos, que se utilizarán para nuestro museo, el cuarto de sala de exhibición del museo y posteriormente el baño.

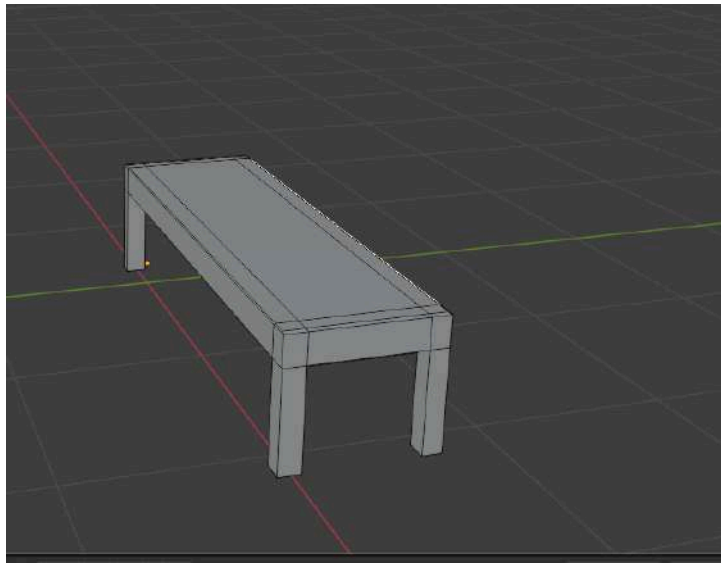


Modelado de los 5 objetos para la sala de exhibición.

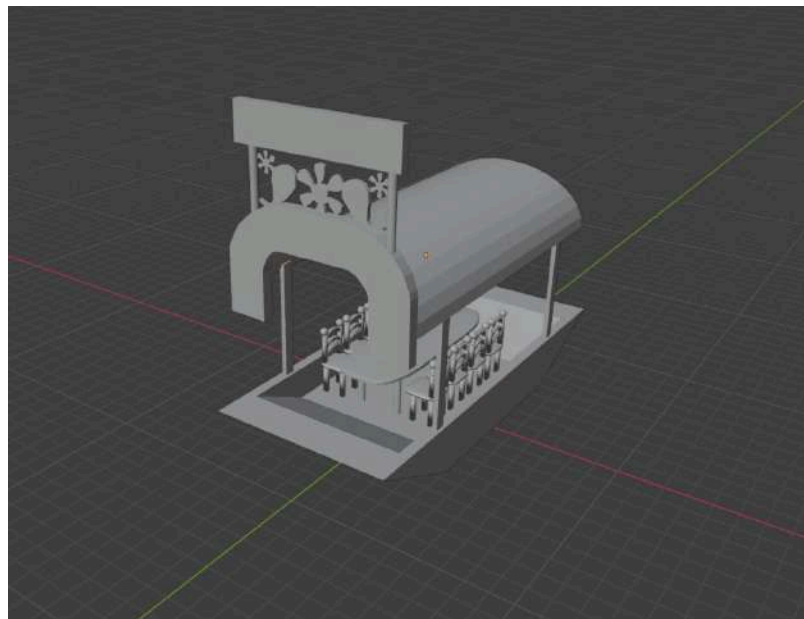
1. Cuadros



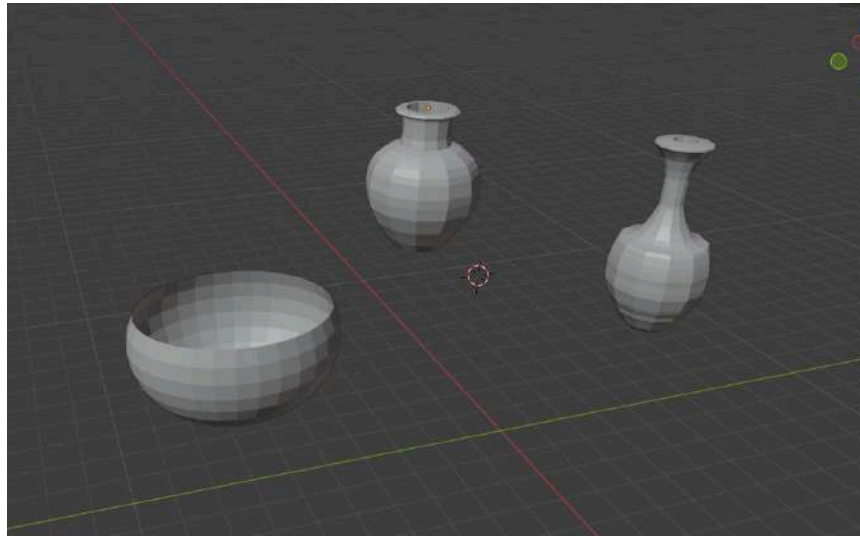
2. Banca



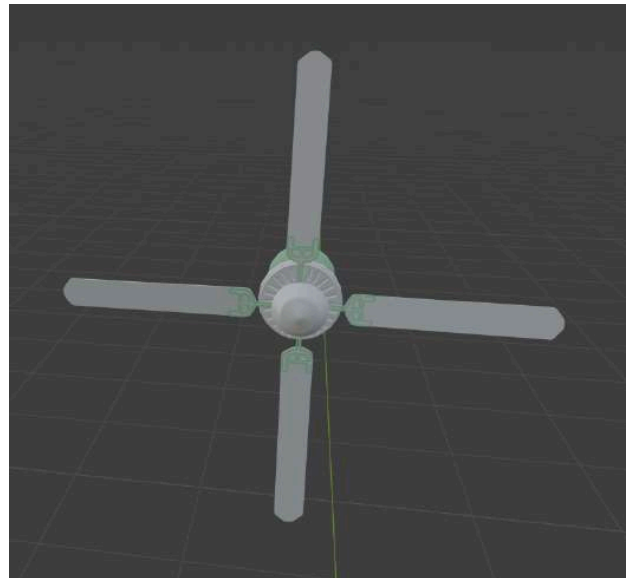
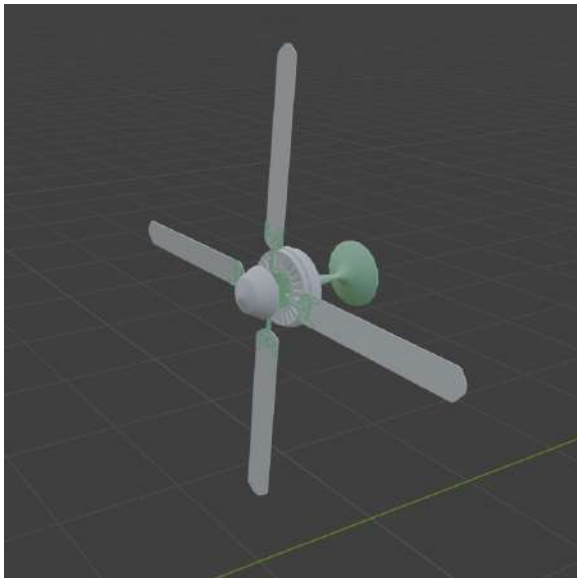
3. Trajinera



4. Esculturas



- Ventilador

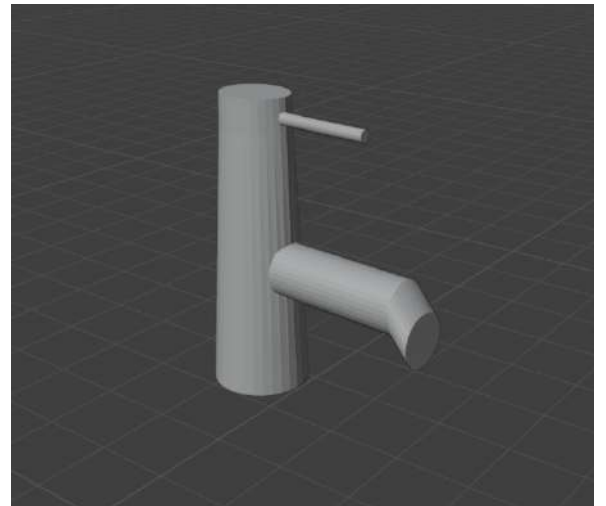
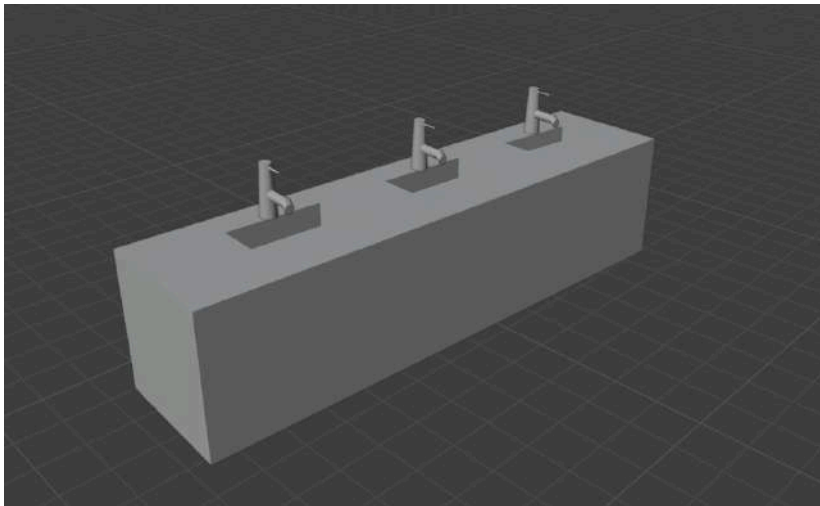


Modelado de los 5 objetos para el baño.

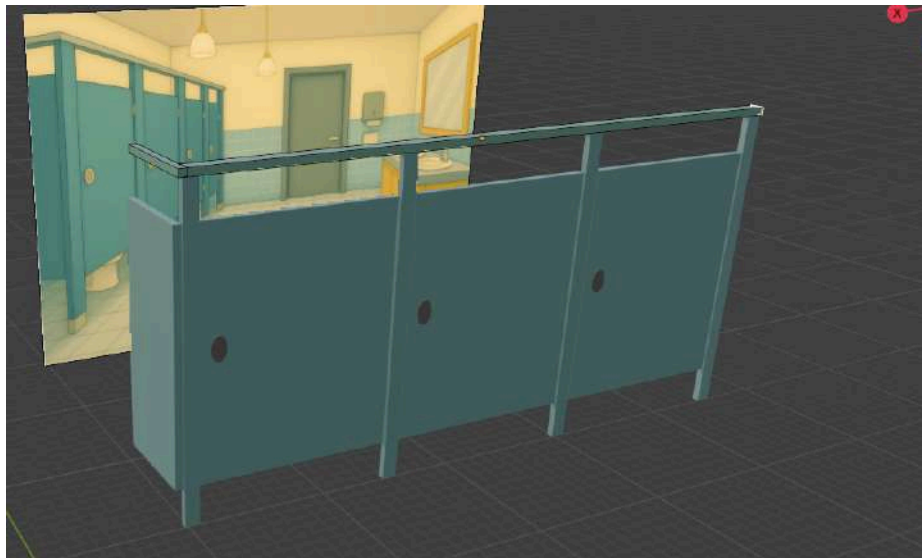
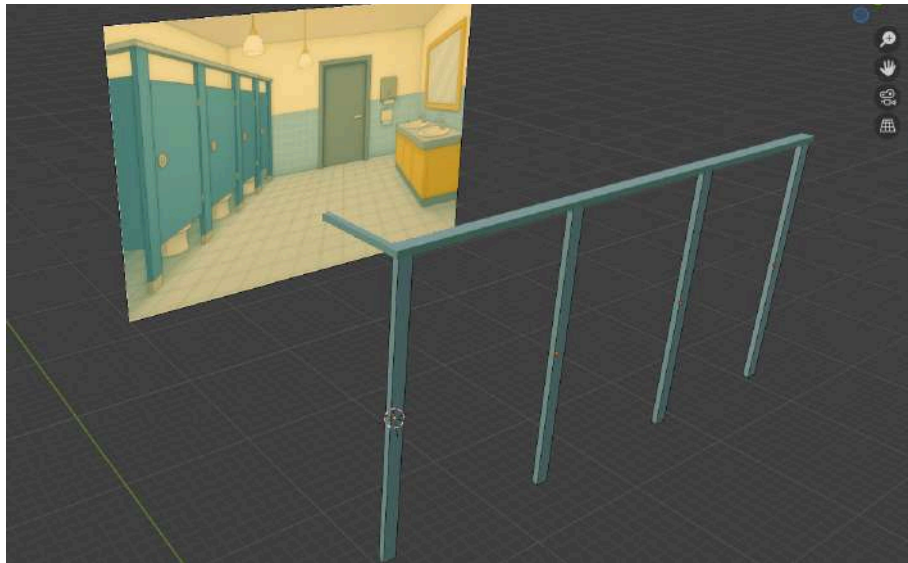
1. Lámparas colgantes



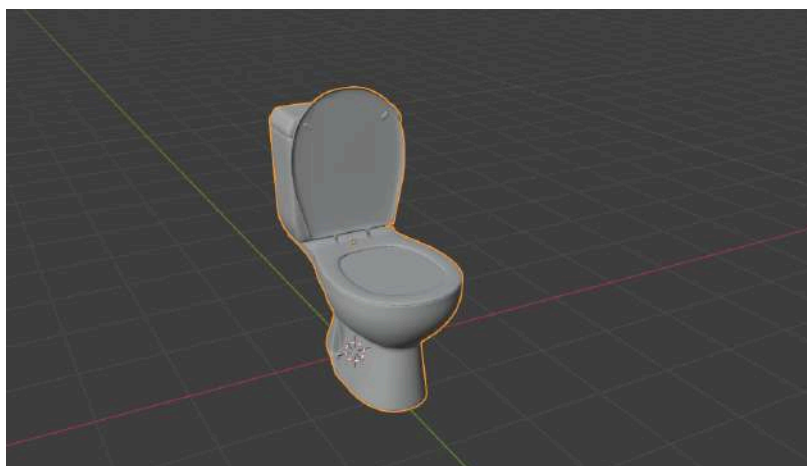
2. Lavamanos



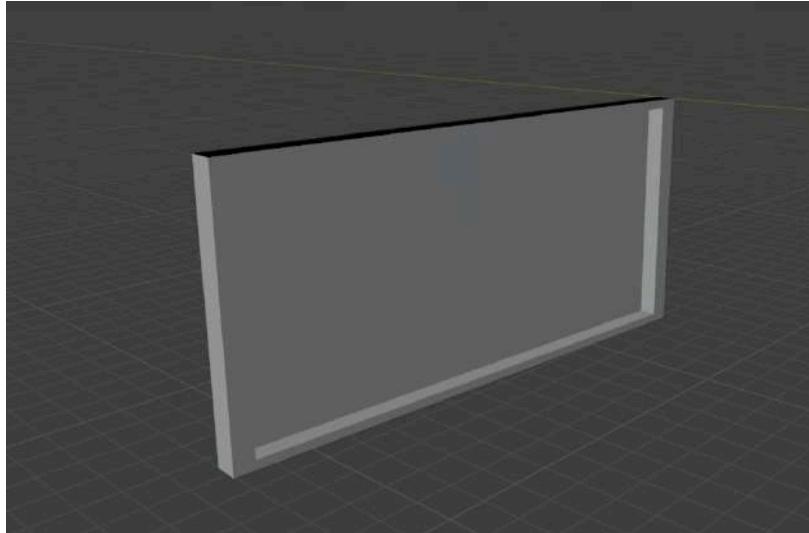
3. Puertas



4. WC



5. Espejo



Se modelaron más objetos, para nuestro complemento de nuestro museo, así como el entorno donde estará nuestro museo, para que no se vea tan vacío.

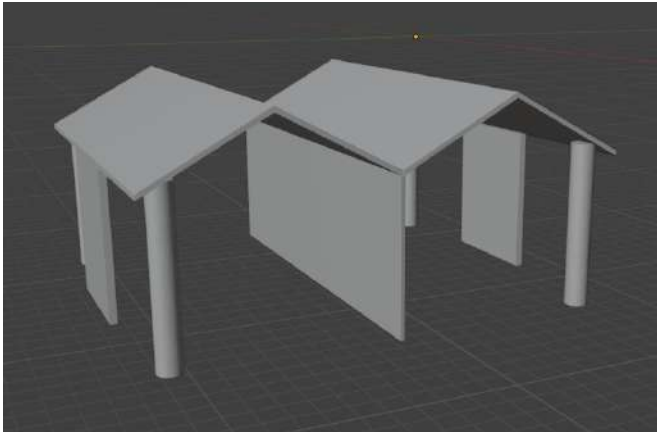
- Aguila



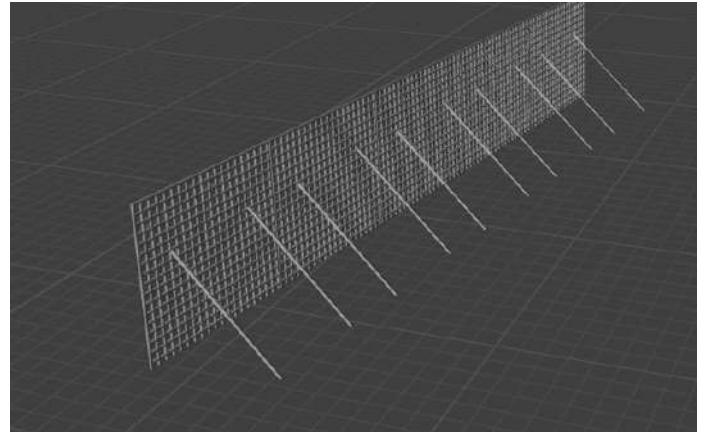
- Letrero



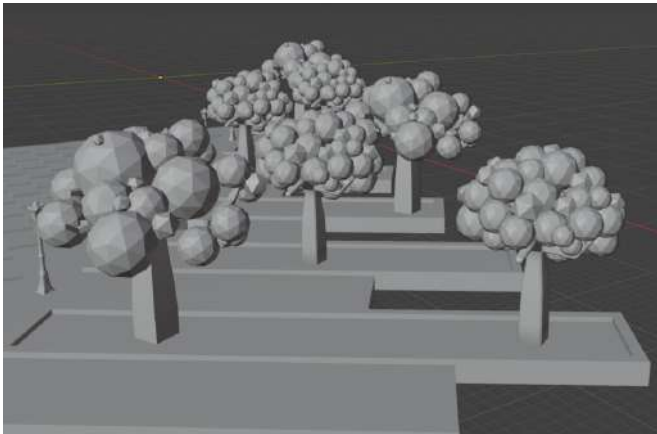
- Entrada Parque



- Rejas



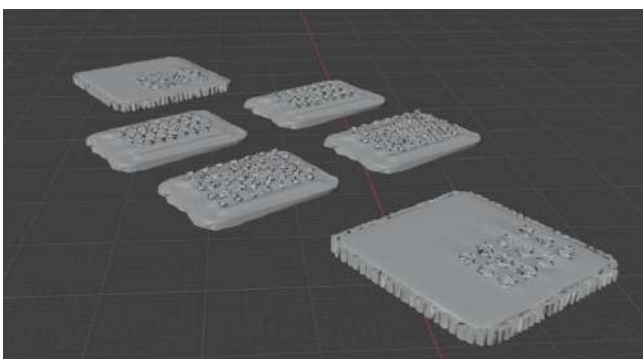
- Árboles



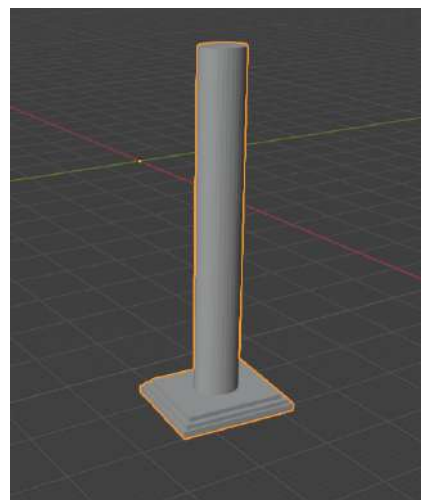
- Mesas



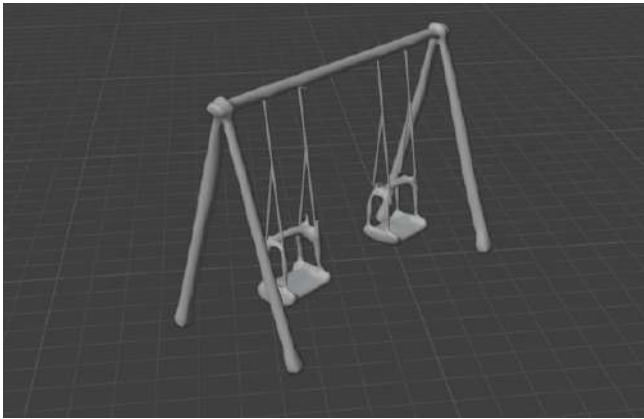
- Chinampas



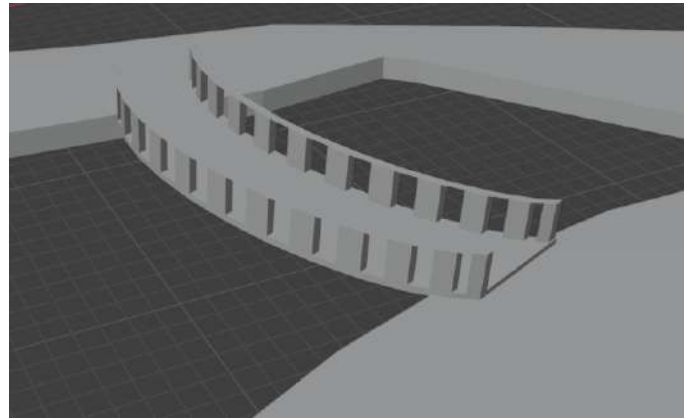
- Torre



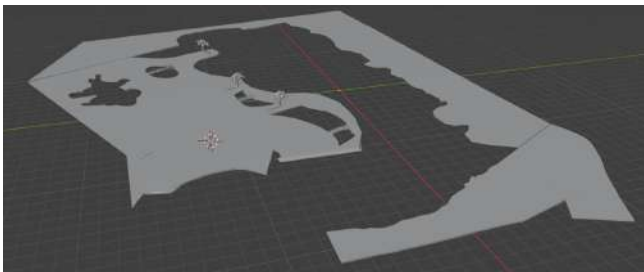
- Columpios



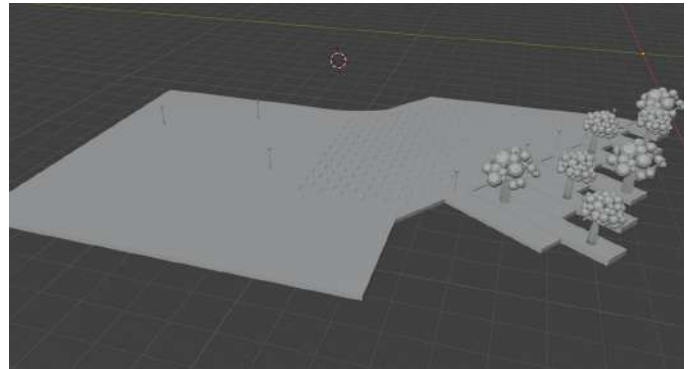
- Puentes



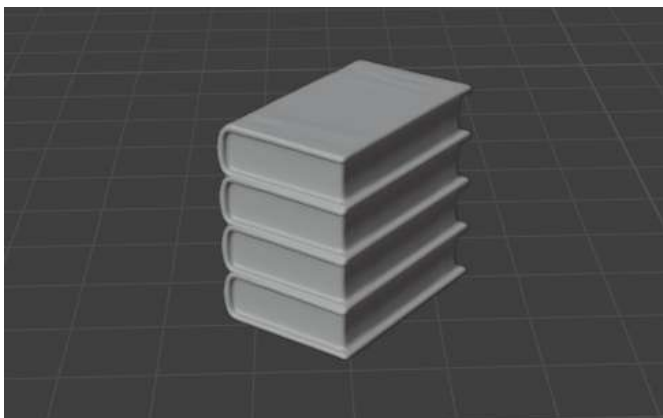
- Tierra



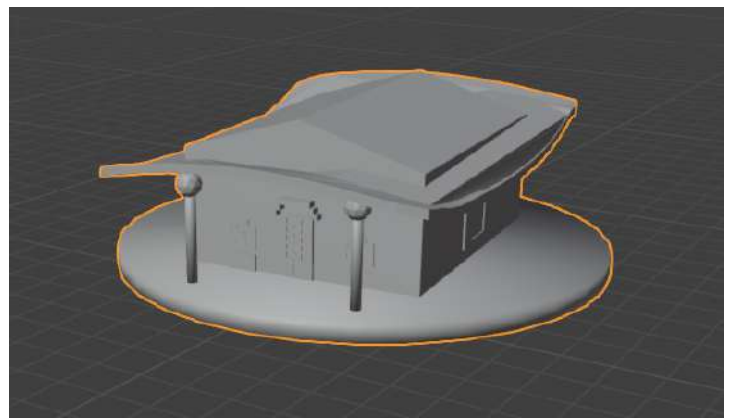
-Terraza



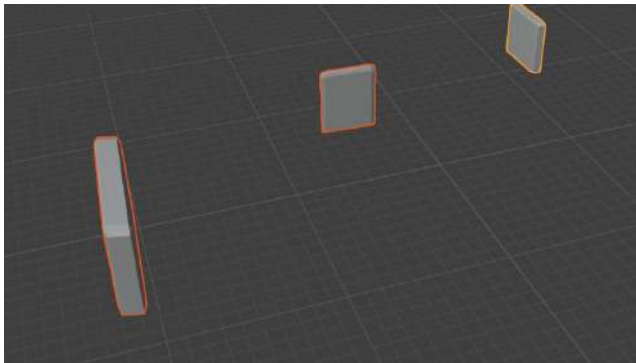
- Libros



- Escuela



- Murales



- Fantasma

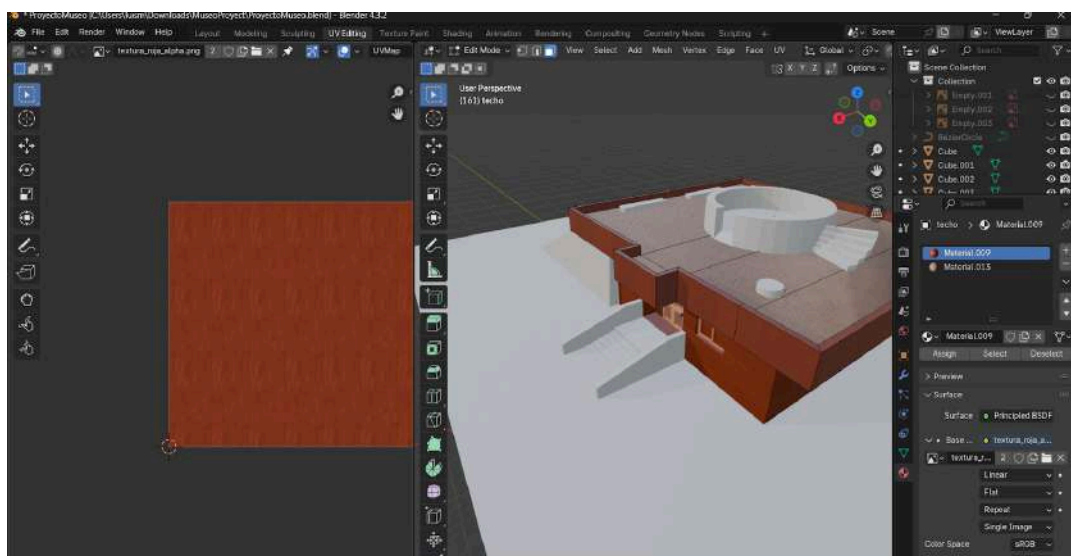


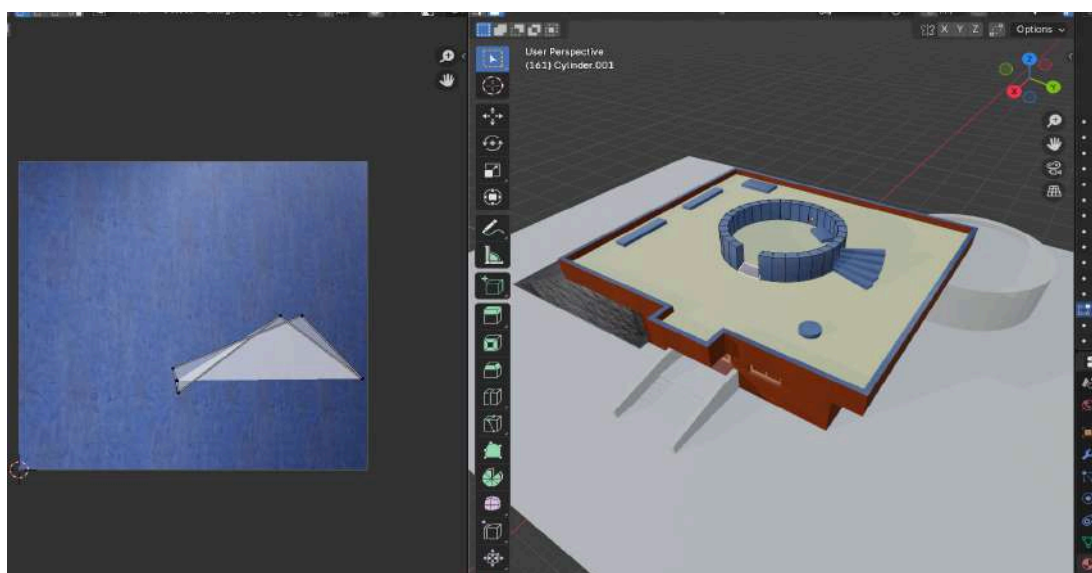
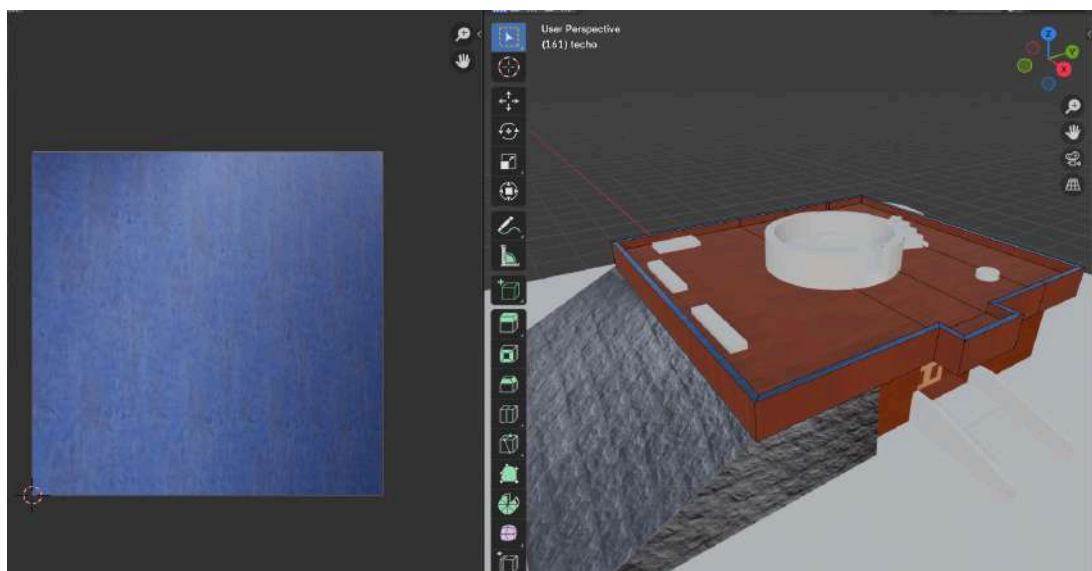
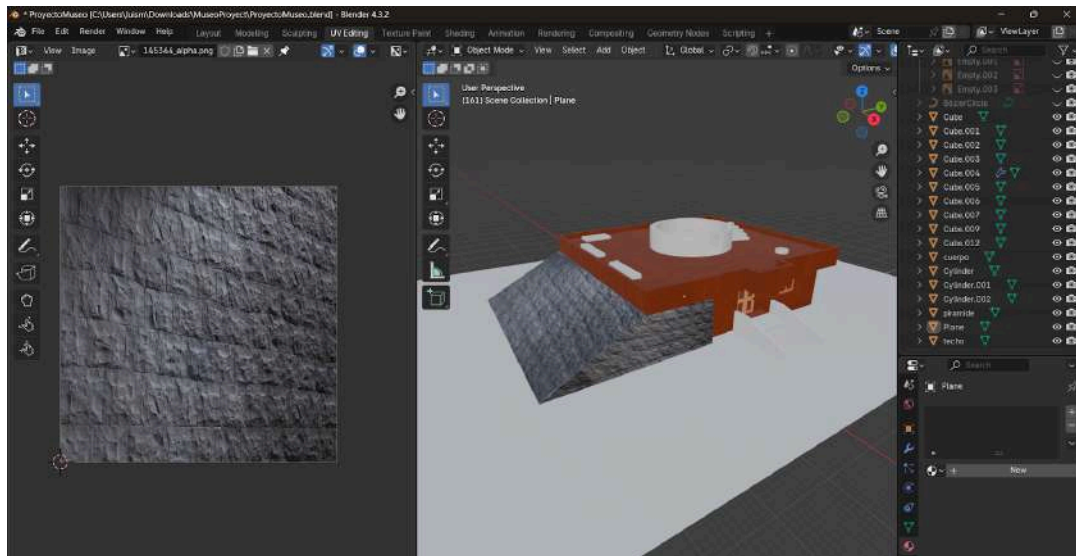
7.3 Texturizado y mapeado UV

El proceso de texturizado se realizó utilizando el software Blender, con el objetivo de aplicar materiales visualmente coherentes y funcionales a cada objeto modelado. Se utilizó mapeo UV manual para un mayor control sobre la distribución de las texturas, evitando estiramientos, repeticiones excesivas o problemas de escala.

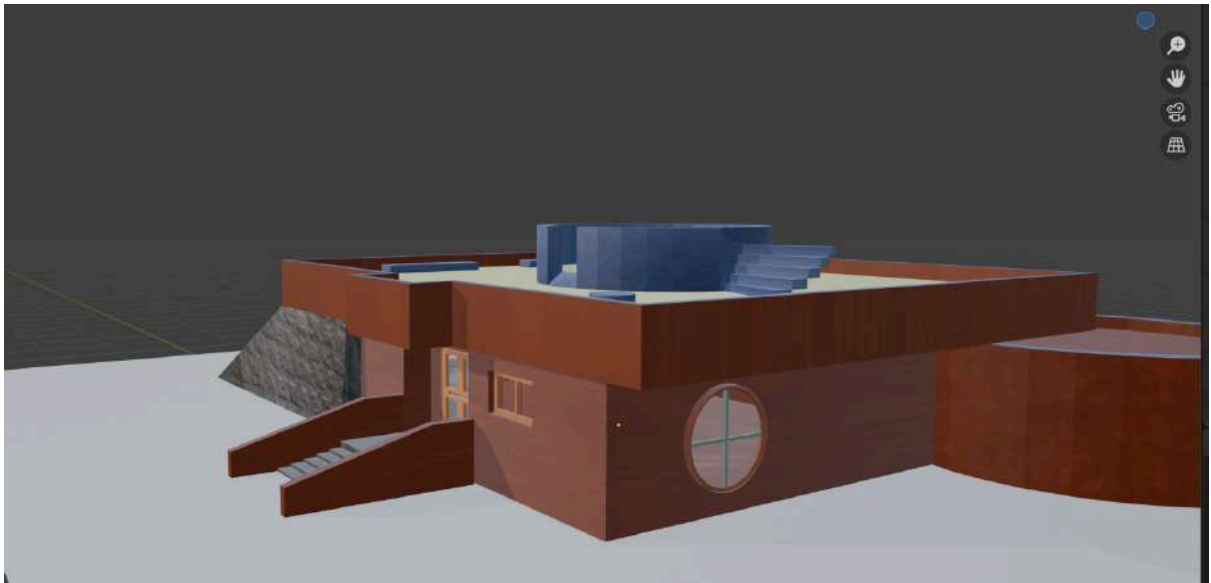
Las texturas utilizadas fueron imágenes en formato .png y .jpg, editadas con GIMP para reducir peso y mejorar contraste sin perder calidad. Se utilizaron resoluciones entre 512x512 px. A continuación se muestra el proceso de texturizado.

Textura fachada





Así quedaría la fachada con sus texturas correspondientes



Textura objetos

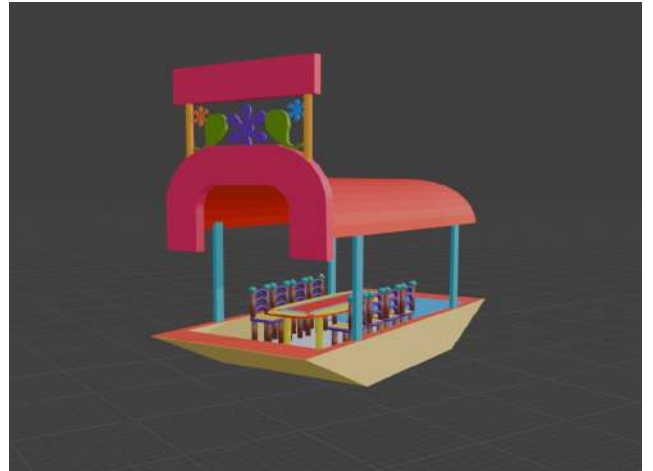
1. Cuadros



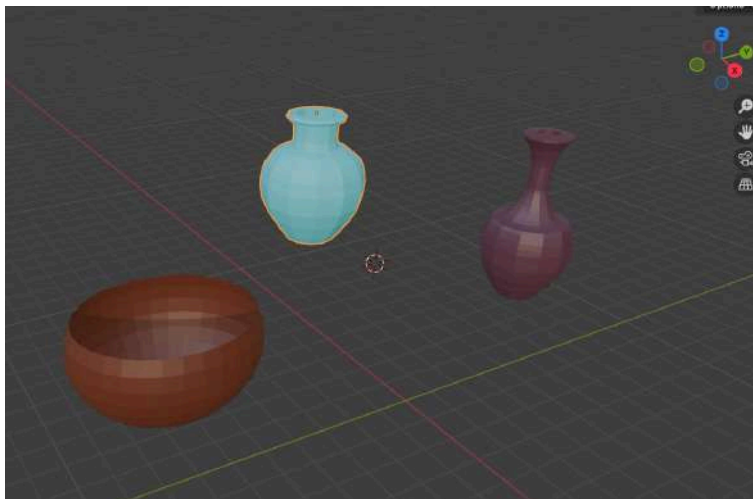
2. Banca



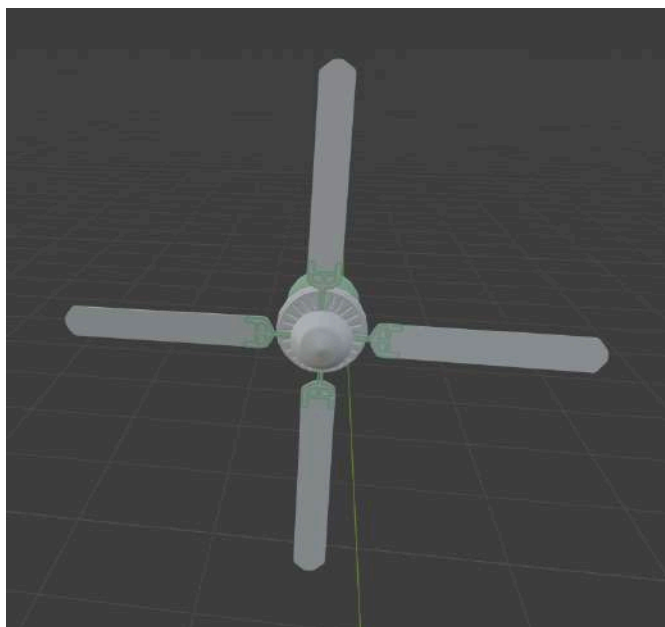
3. Trajinera



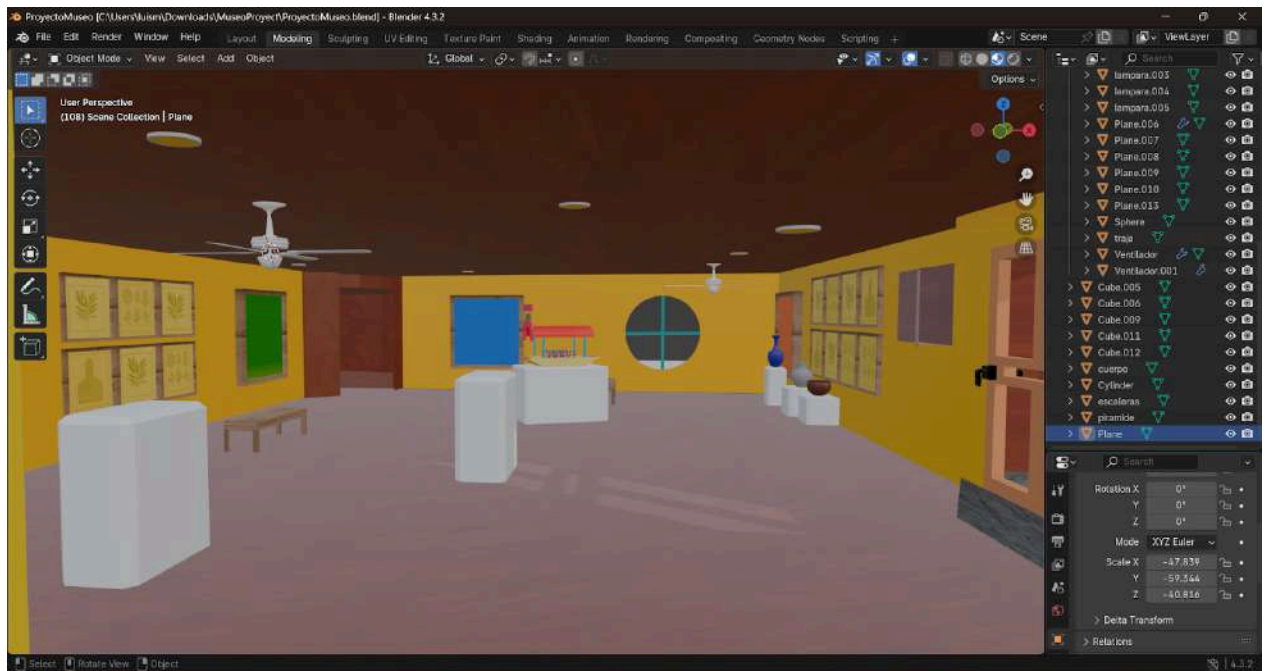
4. Esculturas



5. Ventilador



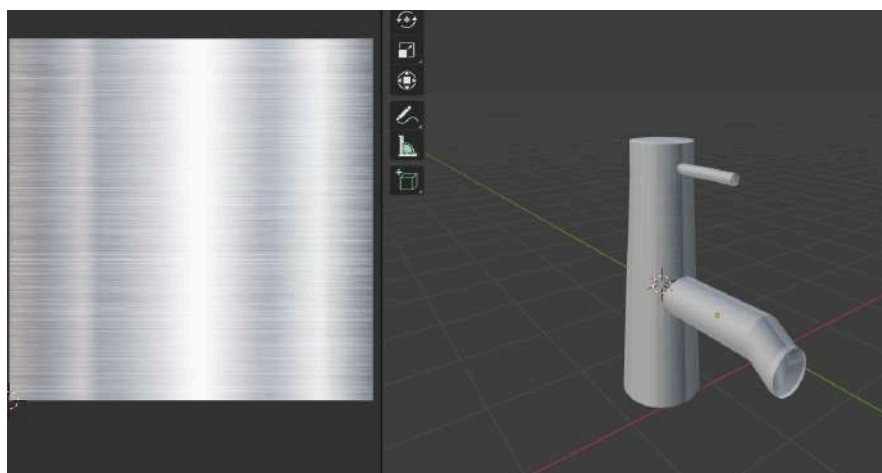
Cuarto con los 5 objetos modelados y su textura



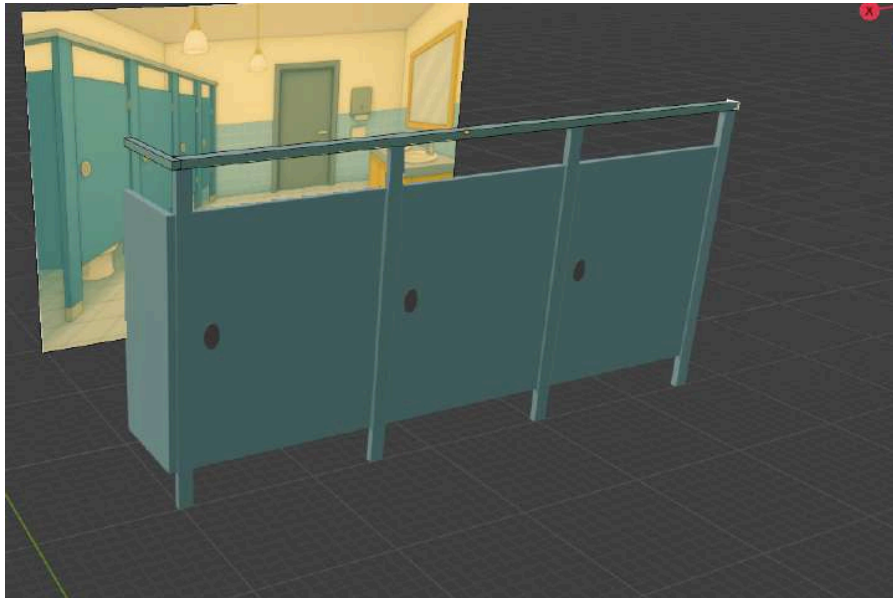
6. Lámparas



7. Lavamanos



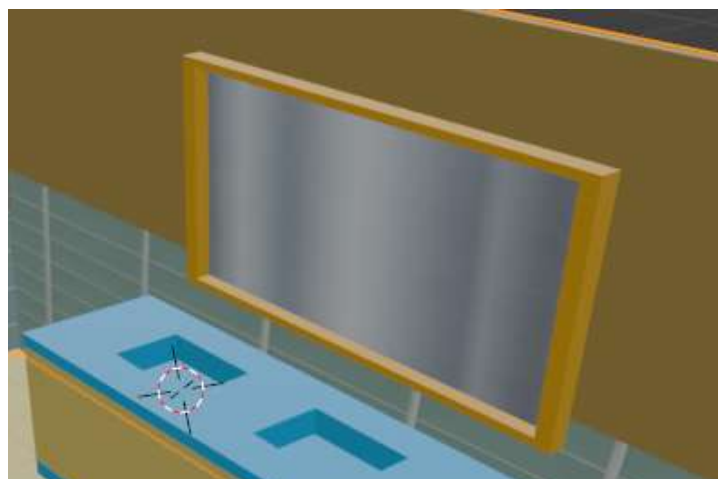
8. Puertas



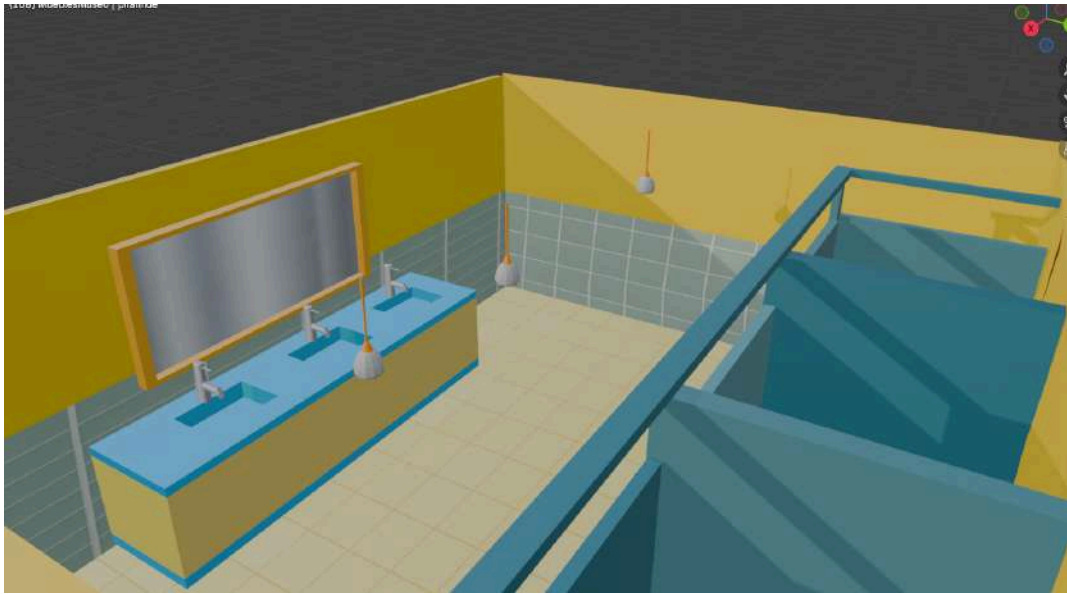
9. WC



10. Espejo

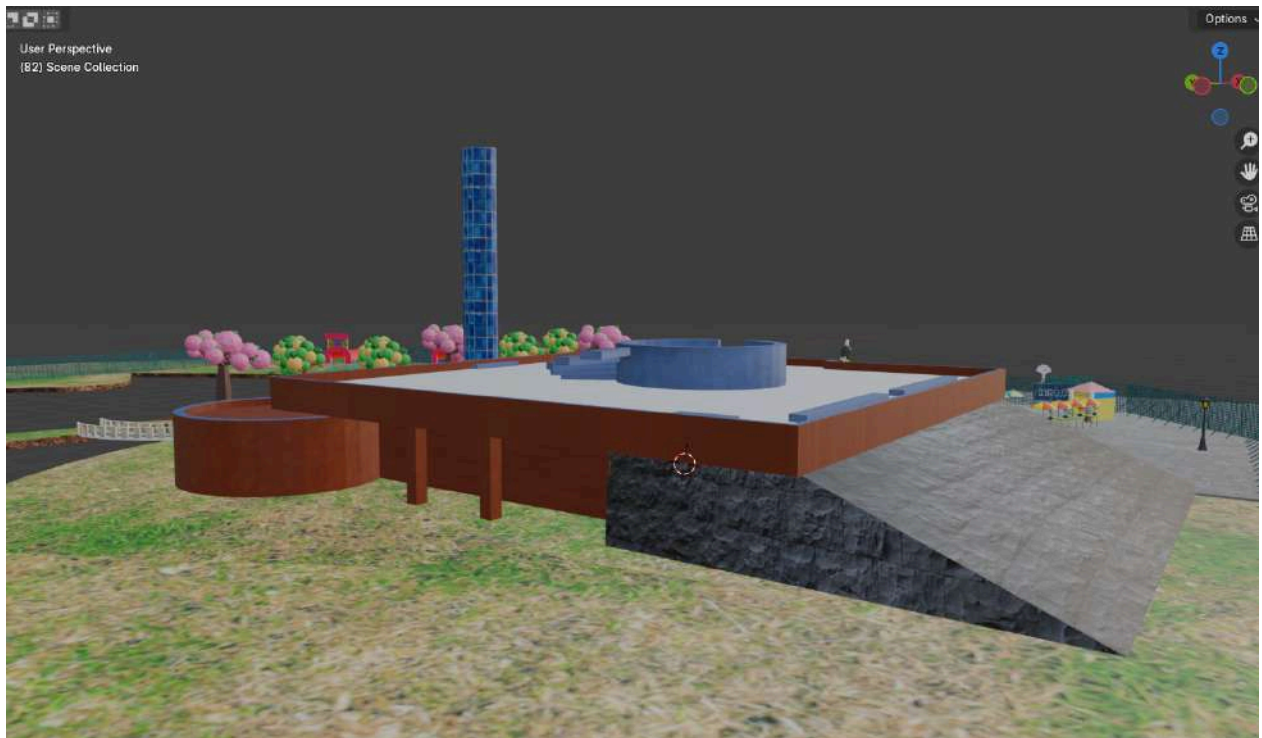


Baño con objetos



El museo finalmente quedaria asi, despues del su modelado y sus texturas correspondientes



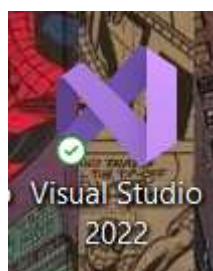


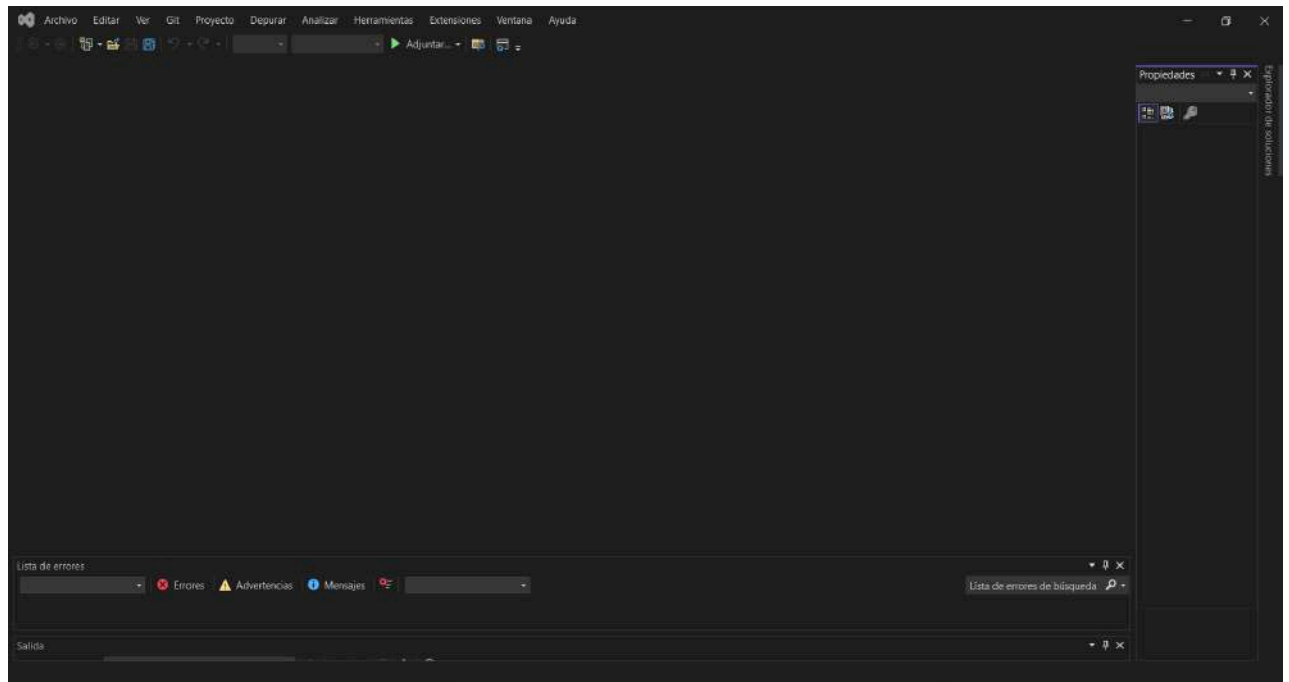
7.4 Herramientas, librerías y estructura del código

Para el desarrollo del museo se utilizaron herramientas y librerías de software libre que facilitaron tanto el modelado 3D como la programación en C++ con OpenGL. A continuación, se detallan las principales herramientas utilizadas y la organización interna del proyecto.

Herramienta utilizada

Visual Studio Code: Fue el entorno de desarrollo empleado para escribir, compilar y organizar el código fuente del proyecto. Se utilizó con una estructura de carpetas clara que permitiera separar el código, modelos, texturas y sonidos. Además, se integraron extensiones para facilitar la escritura en C++ y la navegación entre archivos.





Librerías integradas en el proyecto

- **OpenGL**: Librería principal para renderizar el entorno 3D en tiempo real.
- **GLFW**: Permite crear ventanas, detectar teclado y mover el mouse.
- **GLAD**: Cargador de funciones de OpenGL.
- **GLM**: Para manejar vectores y matrices en 3D.
- **stb_image.h**: Para cargar las texturas en formato .png o .jpg.

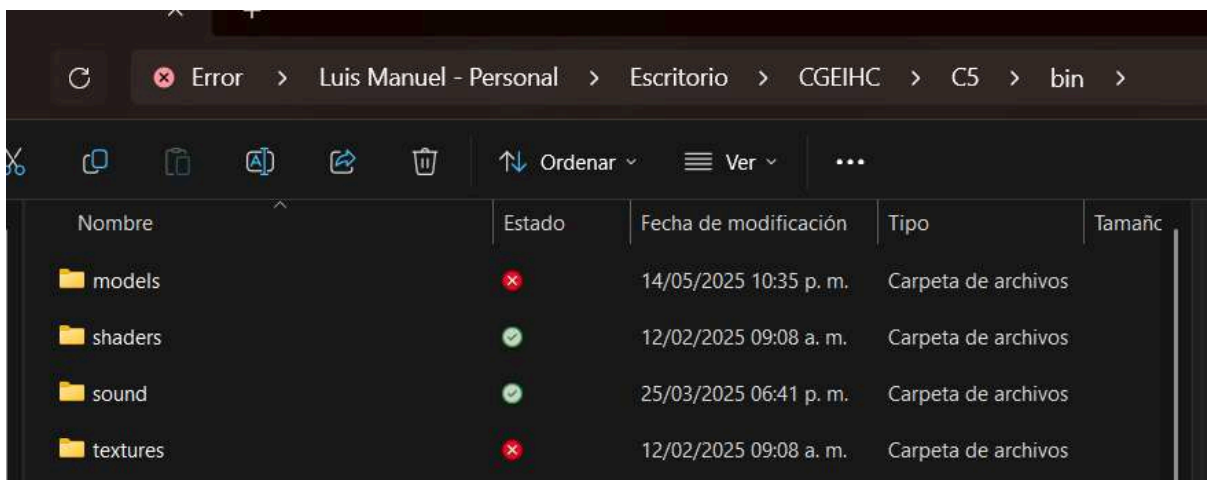
```
1
2
3  ✓ #include <iostream>
4    #include <stdlib.h>
5
6  ✓ // GLAD: Multi-Language GL/GLES/EGL/GLX/WGL Loader-Generator
7    // https://glad.davld.de/
8    #include <glad/glad.h>
9
10   // GLFW: https://www.glfw.org/
11   #include <GLFW/glfw3.h>
12
13   // GLM: OpenGL Math library
14   ✓ #include <glm/glm.hpp>
15     #include <glm/gtc/matrix_transform.hpp>
16     #include <glm/gtc/type_ptr.hpp>
17
18   // Model loading classes
19   ✓ #include <shader_m.h>
20     #include <camera.h>
21     #include <model.h>
22     #include <animatedmodel.h>
23     #include <material.h>
24     #include <light.h>
25     #include <cubemap.h>
26
27   #include <irrKlang.h>
```

Estructura de carpetas del proyecto

El proyecto fue organizado mediante carpetas específicas que agrupan los recursos gráficos, sonoros y los modelos utilizados en la construcción del museo virtual. Esta organización permite mantener el proyecto ordenado y facilitar el acceso a los archivos durante la programación.

A continuación, se describen las carpetas principales:

- **/models**: Contiene los modelos exportados desde Blender en formato .fbx junto con sus texturas.
- **/textures**: Carpeta donde se encuentran el cubemap de nuestro proyecto, para darle un fondo más realista
- **/shaders**: Incluye los archivos de vertex y fragment shaders (en caso de usarse iluminación o efectos personalizados).
- **/sound**: Almacena los archivos de sonido ambiental utilizados.



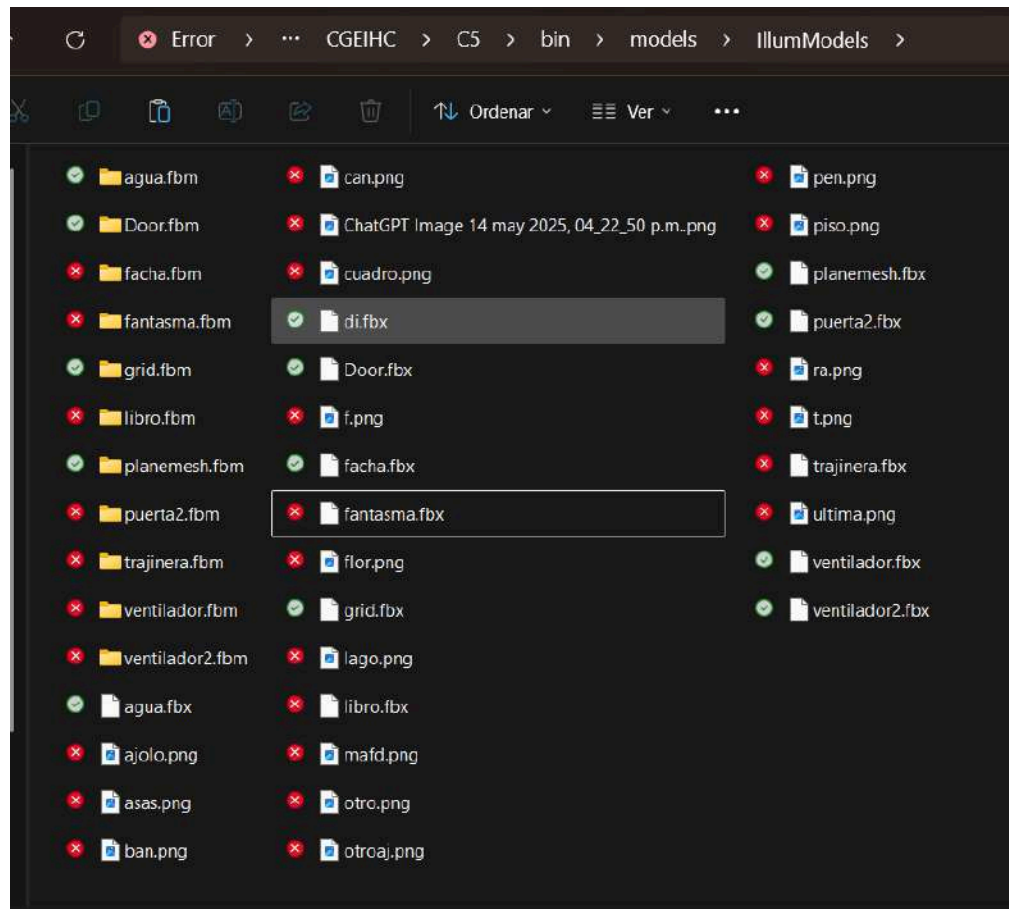
Nombre	Estado	Fecha de modificación	Tipo	Tamaño
models	✗	14/05/2025 10:35 p. m.	Carpeta de archivos	
shaders	✓	12/02/2025 09:08 a. m.	Carpeta de archivos	
sound	✓	25/03/2025 06:41 p. m.	Carpeta de archivos	
textures	✗	12/02/2025 09:08 a. m.	Carpeta de archivos	

7.5 Carga de modelos y texturas en el sistema

Una vez finalizado el proceso de modelado y texturizado en Blender, los modelos fueron exportados en formato .fbx para su integración en OpenGL. El sistema se encargó de cargar estos archivos, interpretarlos y renderizarlos en tiempo real.

Los modelos 3D fueron creados en Blender y posteriormente exportados en formato .fbx, debido a su compatibilidad con estructuras jerárquicas, texturas embebidas y

ejes compatibles con OpenGL. Estos archivos se almacenan en la carpeta /models/IllumModels.



En el código, la carga de modelos se realiza mediante las siguientes instrucciones

```
// Carga la información del modelo
Model *house;
Model *door;
Model *moon;
Model *gridMesh;
Model* fan1;
Model* fan2;
Model* door_bathroom;
Model* trajinera;
Model* book_model;
Model* ghost_model;
```

```

house = new Model("models/IllumModels/facha.fbx");
door = new Model("models/IllumModels/di.fbx");
fan1 = new Model("models/IllumModels/ventilador.fbx");
fan2 = new Model("models/IllumModels/ventilador2.fbx");
gridMesh = new Model("models/IllumModels/grid.fbx");
door_bathroom = new Model("models/IllumModels/puerta2.fbx");
trajinera = new Model("models/IllumModels/trajinera.fbx");
ghost_model = new Model("models/IllumModels/fantasma.fbx");
book_model = new Model("models/IllumModels/libro.fbx"); //

```

Cube Map

Para ambientar visualmente el entorno del museo, se integró un cube map que actúa como fondo de cielo, brindando profundidad y realismo a la escena. Este cube map fue cargado utilizando una clase personalizada CubeMap, y las imágenes individuales fueron ubicadas en la carpeta /textures/cubemap/02.

Se utilizaron 6 imágenes, una para cada cara del cubo:

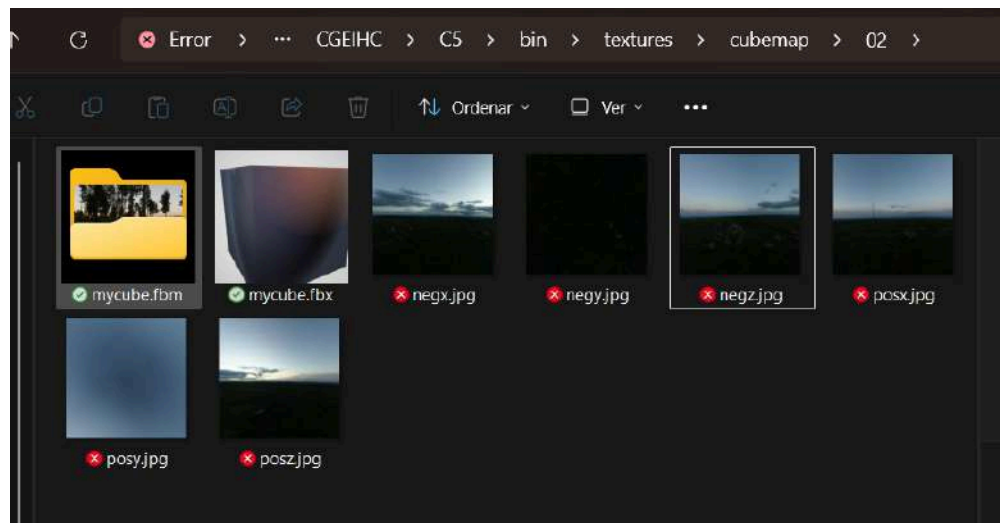
```

// Cubemap
vector<std::string> faces
{
    "textures/cubemap/02/posx.jpg",
    "textures/cubemap/02/negx.jpg",
    "textures/cubemap/02/posy.jpg",
    "textures/cubemap/02/negy.jpg",
    "textures/cubemap/02/posz.jpg",
    "textures/cubemap/02/negz.jpg"
};
mainCubeMap = new CubeMap();
mainCubeMap->loadCubemap(faces);

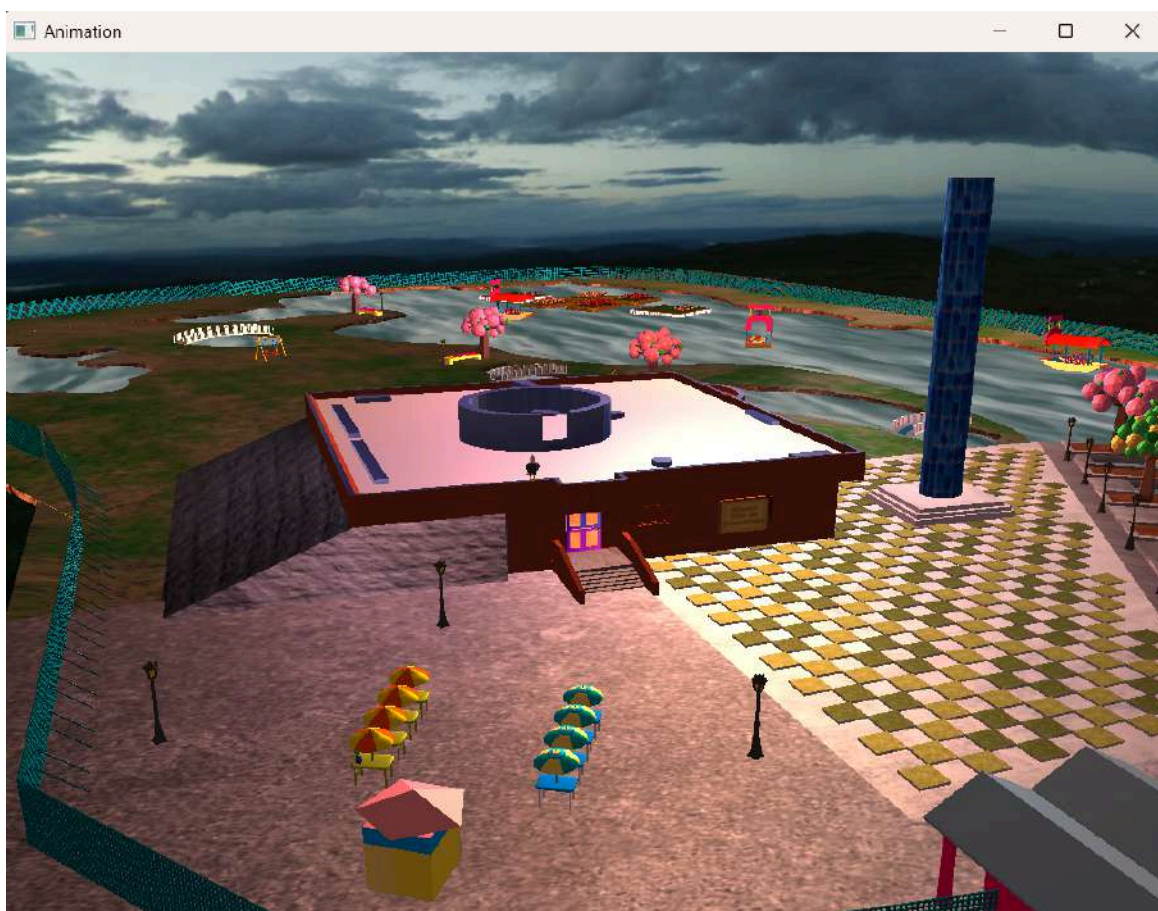
```

Estas imágenes se aplican como textura a una malla cúbica que rodea toda la escena. La visualización del cubemap se realiza antes del renderizado de los

modelos 3D, garantizando que siempre permanezca al fondo sin interferir con los objetos del museo.



Al ejecutar podemos ver nuestros modelos con sus textura cargados en opengl



Museo por dentro

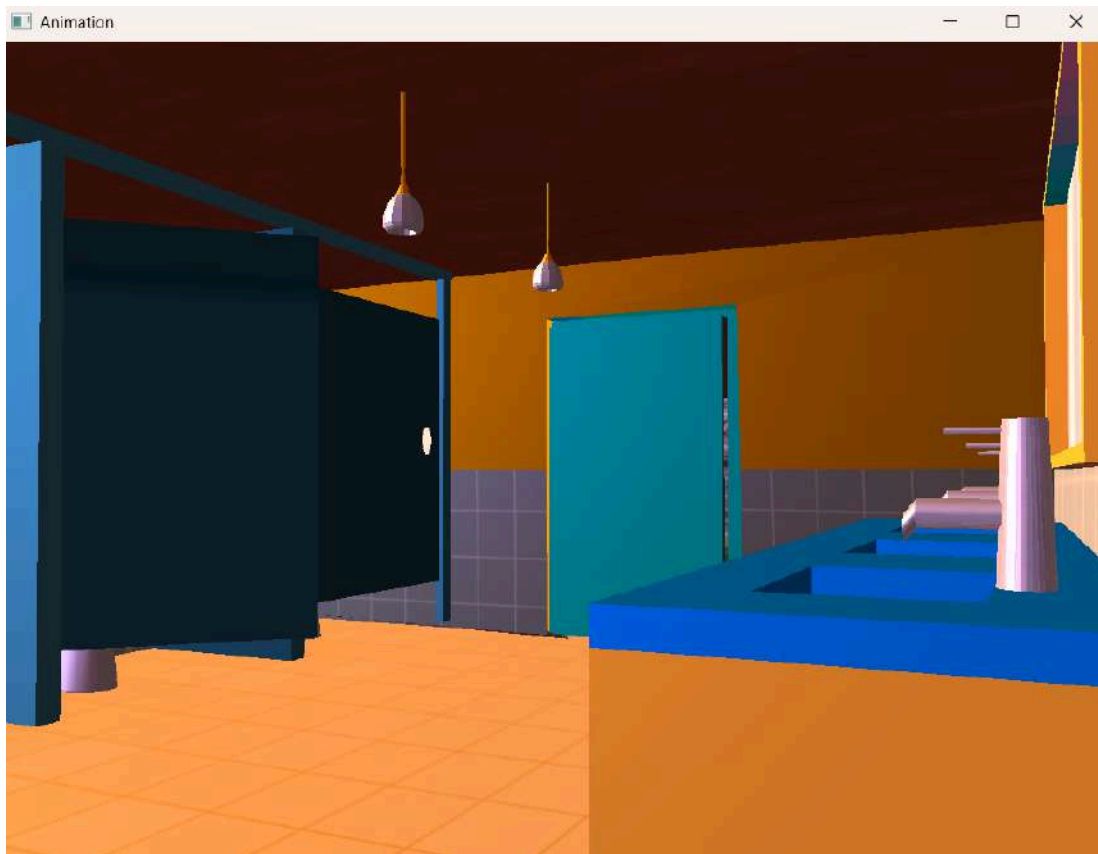
Cuarto 1





Cuarto 2 baño





7.6 Implementación de la cámara sintética

El proyecto incluye una cámara sintética en primera persona implementada manualmente con ayuda de la librería GLM para el manejo de vectores y matrices, y gestionada mediante una clase personalizada llamada Camera. Esta cámara permite al usuario desplazarse por el museo de forma libre con controles precisos y naturales.

```
✓ #include <shader_m.h>
#include <camera.h>
#include <model.h>
#include <animatedmodel.h>
#include <material.h>
#include <light.h>
#include <cubemap.h>
```

Se inicializó la cámara principal en primera persona con una posición y orientación personalizadas:

```
Camera camera(glm::vec3(210.75f, -4.3073f, 108.293f),
    glm::vec3(0.0f, 1.0f, 0.0f), -185.0f, -10.0f);
```

Con el callback `mouse_callback()` se controla la rotación de la vista horizontal y vertical:

```
void scroll_callback(GLFWwindow* window, double xoffset, double yoffset)
{
    camera.ProcessMouseScroll((float)yoffset);
}
```

Podemos obtener esta vista



7.7 Animaciones

Para lograr la animación de cada objeto que usamos, fue necesario exportarlos en .fbx como se mencionó, posteriormente mover su posición para que coincidieran donde debían ir.

1. Puerta principal del museo

La puerta principal del museo se anima rotando sobre su eje vertical mediante transformaciones GLM. El usuario puede abrirla presionando H o cerrarla con J, lo que modifica en tiempo real la variable `door_rotation`, usada para girar el modelo durante el renderizado.

```
model = glm::mat4(1.0f);
model = glm::translate(model, door_position);
model = glm::rotate(model, glm::radians(door_rotation), glm::vec3(0.0f, 1.0f, 0.0f));
model = glm::rotate(model, glm::radians(-90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
mLightsShader->setMat4("model", model);
door->Draw(*mLightsShader);
```

```
if (glfwGetKey(window, GLFW_KEY_H) == GLFW_PRESS)
    door_rotation += 1.f;
if (glfwGetKey(window, GLFW_KEY_J) == GLFW_PRESS)
    door_rotation -= 1.f;
```

2. Ventiladores

Ambos ventiladores giran mediante una animación continua basada en el tiempo (`deltaTime`). La rotación se activa con la tecla T y se detiene con G

```
glm::mat4 model_fan = glm::mat4(1.0f);
model_fan = glm::translate(model_fan, fan1_position);
model_fan = glm::rotate(model_fan, glm::radians(-90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
model_fan = glm::rotate(model_fan, glm::radians(fan_rotation), glm::vec3(0.0f, 0.0f, 1.0f));

mLightsShader->setMat4("model", model_fan);
fan1->Draw(*mLightsShader);
```

```
if (glfwGetKey(window, GLFW_KEY_T) == GLFW_PRESS) {
    fan_rotate = true;
    fan2_rotate = true;
}
if (glfwGetKey(window, GLFW_KEY_G) == GLFW_PRESS) {
    fan_rotate = false;
    fan2_rotate = false;
}
```


3. Trajinera

Si el usuario está a menos de 10 unidades, la variable `trajinera_rotation` se incrementa, haciendo que gire constantemente mientras permanezca cerca.

```
float dist_trajinera = glm::distance(camera.Position, trajinera_position);

if (dist_trajinera < 10.0f) {
    trajinera_rotation += 60.0f * deltaTime;
    if (trajinera_rotation > 360.0f) trajinera_rotation -= 360.0f;
}
```

```
glm::mat4 model_trajinera = glm::mat4(1.0f);
model_trajinera = glm::translate(model_trajinera, trajinera_position);
model_trajinera = glm::rotate(model_trajinera, glm::radians(trajinera_rotation), glm::vec3(0.0f, 1.0f, 0.0f));
model_trajinera = glm::rotate(model_trajinera, glm::radians(-90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
mLightsShader->setMat4("model", model_trajinera);
trajinera->Draw(*mLightsShader);
```

4. Aparicion fantasma

El fantasma aparece sobre el libro cuando se activa la variable `show_book_info`. Su movimiento ascendente se logra incrementando `popup_height` con el tiempo (`deltaTime`). Luego, el modelo se traslada en el eje Y, se rota y se escala para mostrarse correctamente en pantalla.

```
if (show_book_info) {
    popup_height += deltaTime * 1.5f;
    if (popup_height > 1.5f) popup_height = 1.5f;

    glm::vec3 ghost_pos = book_position + glm::vec3(0.0f, popup_height + 1.0f, 0.0f);
    glm::mat4 ghost_model_mat = glm::mat4(1.0f);
    ghost_model_mat = glm::translate(ghost_model_mat, ghost_pos);
    ghost_model_mat = glm::rotate(ghost_model_mat, glm::radians(-90.0f), glm::vec3(1.0f, 0.0f, 0.0f));
    ghost_model_mat = glm::scale(ghost_model_mat, glm::vec3(0.45f));

    mLightsShader->setMat4("model", ghost_model_mat);
    ghost_model->Draw(*mLightsShader);
}
```

5. Puerta del baño se abre sola

La puerta del baño se abre automáticamente cuando el usuario se acerca a menos de 10 unidades de distancia, y se cierra al alejarse. Esto se logra

midiendo la distancia con `glm::distance` y ajustando el ángulo `door_bath_rotation` en cada frame usando `deltaTime`

```
float dist = glm::distance(camera.Position, door_bath_position);

// Abrir si está cerca
if (dist < 10.0f && door_bath_rotation < 100.0f) {
    door_bath_rotation += 60.0f * deltaTime;
}
// Cerrar si ya se alejó
else if (dist >= 10.0f && door_bath_rotation > 0.0f) {
    door_bath_rotation -= 60.0f * deltaTime;
}
```

6. Movimiento del lago

La superficie de agua se anima usando un shader procedural que simula movimiento ondulatorio. En cada frame, se incrementa una variable de tiempo (`wavesTime`), la cual es enviada al shader como uniform para modificar dinámicamente los vértices y crear el efecto de olas

```
wavesShader->setFloat("time", wavesTime);
wavesShader->setFloat("radius", 5.0f);
wavesShader->setFloat("height", 5.0f);

gridMesh->Draw(*wavesShader);
wavesTime += 0.0003;
```



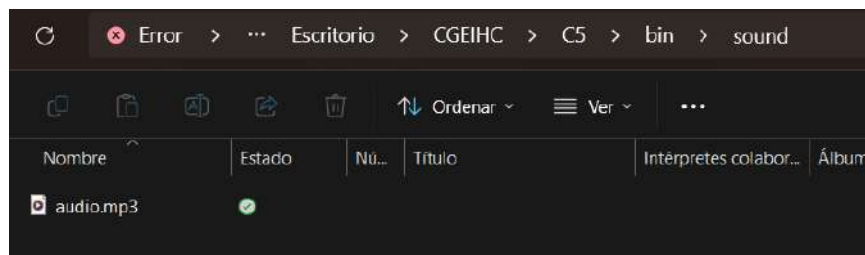
7.8 Carga e implementación de sonido

El sistema reproduce un sonido ambiental desde el inicio del programa utilizando IrrKlang.

```
#include <irrKlang.h>
using namespace irrklang;
```

```
SoundEngine->play2D("sound/audio.mp3", true);
```

El parámetro true indica que se reproduce de forma continua mientras se ejecuta la aplicación.

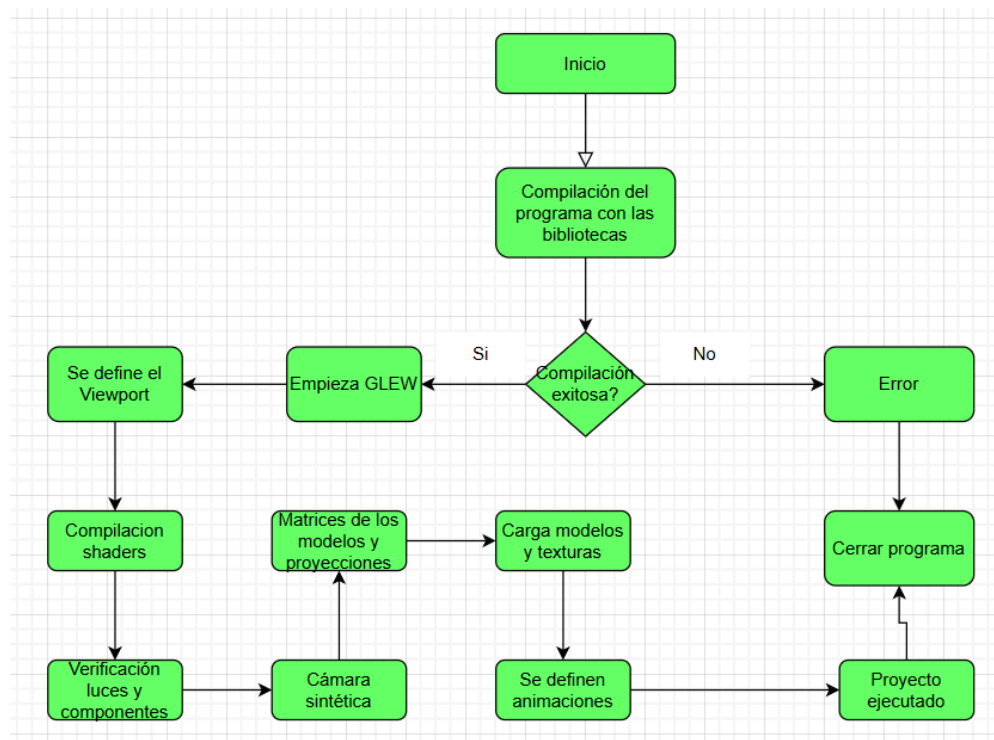


7.9 Diagrama de flujo del sistema

El diagrama de flujo representa la arquitectura lógica del ciclo de vida del programa, desde su inicialización hasta su cierre. Se inicia con la compilación del programa junto con las bibliotecas necesarias (GLFW, GLAD, GLM, irrKlang), seguida por una validación de compilación. En caso de error, el sistema termina su ejecución de forma controlada.

Si la compilación es exitosa, se procede con la configuración del entorno gráfico: inicialización de GLEW, definición de VAO y VBO, compilación de shaders, y definición del viewport. A continuación, se configuran los sistemas de iluminación y se declara la cámara sintética para establecer la perspectiva de renderizado.

Posteriormente, se definen las matrices de modelo, vista y proyección, y se realiza la carga de modelos 3D en formato .fbx, junto con sus texturas correspondientes. Una vez cargados los recursos, se inicializa el sistema de animaciones, y comienza el bucle principal de renderizado, donde se actualizan continuamente los elementos interactivos del entorno, como la puerta del museo, ventiladores, la trajinera, la puerta del baño, y el efecto emergente del libro y el fantasma.



7.10 Diccionario de funciones y variables clave

Funciones principales

Función	Descripción
main()	Función principal. Inicia el programa, ejecuta Start() y entra en el ciclo Update().
Start()	Inicializa GLFW, GLAD, shaders, luces, cámara, modelos, texturas, cubemap y sonido.
Update()	Bucle principal. Procesa entradas, actualiza animaciones, renderiza modelos y efectos.
processInput()	Captura y ejecuta entradas del teclado (movimiento, acciones, selección de cámara).
mouse_callback()	Controla la rotación de la cámara en primera persona mediante movimiento del mouse.
scroll_callback()	Controla el zoom de la cámara mediante la rueda del mouse.
mouse_button_callback()	Detecta clics del usuario. Activa aparición del fantasma al hacer clic en el libro.
SetLightUniform*()	Funciones que cargan parámetros de luces al shader (vec3, vec4, int, float).

Variables clave del sistema

Variable	Tipo	Descripción / Función
camera	Camera	Cámara en primera persona. Controlada con teclado y mouse.
camera3rd	Camera	Cámara en tercera persona (no activa por defecto).
deltaTime	float	Tiempo entre frames, usado para animaciones suaves.
lastFrame	float	Marca de tiempo del frame anterior.
door_rotation	float	Ángulo de apertura de la puerta del museo (teclas H/J).
fan_rotation	float	Rotación del ventilador 1 (tecla T/G).
fan2_rotation	float	Rotación del ventilador 2. Sincronizado con el primero.
door_bath_rotation	float	Ángulo de la puerta del baño (automática por cercanía).
trajinera_rotation	float	Rotación de la trajinera al acercarse.
book_position	glm::vec3	Posición del libro en la escena.
popup_height	float	Altura de aparición del fantasma.
show_book_info	bool	Activa la animación del fantasma.

water_position	glm::vec3	Posición del plano que simula el agua.
wavesTime	float	Valor de tiempo en el shader procedural del agua.
SoundEngine	ISoundEngine*	Motor de audio de irrKlang.
window	GLFWwindow*	Ventana principal del sistema.
mainCubeMap	CubeMap*	Cubemap de cielo que rodea la escena.
gLights	std::vector<Light>	Vector de luces en la escena.

MANUAL DE USUARIO

El presente manual de usuario tiene como objetivo guiar al visitante virtual en el recorrido por el Museo del Parque Ecológico de Xochimilco, un entorno tridimensional desarrollado con tecnologías de computación gráfica. Este espacio ha sido diseñado para ofrecer una experiencia inmersiva e interactiva, permitiendo explorar contenidos relacionados con la riqueza natural, cultural y agrícola de la región de Xochimilco.

Requisitos recomendados

- **Sistema operativo:** Windows 10, 11 Home o Pro
- **Procesador:** Intel Core i7 (9.ª gen o superior) / AMD Ryzen 5 3600 o superior
- **Memoria RAM:** 16 GB o más
- **Tarjeta gráfica:** NVIDIA GeForce GTX 1660 Ti / RTX 2060 o AMD Radeon RX 6600 o superior
- **Almacenamiento libre:** 1 GB en unidad SSD (NVMe ideal)
- **Resolución de pantalla:** Full HD (1920x1080) o superior, preferentemente con monitor de 60 Hz o más
- **Periféricos:** Mouse con sensor óptico y teclado físico para mejor control
- **Audio:** Altavoces o audífonos para disfrutar del sonido ambiental e interacciones

REQUISITOS DEL SISTEMA

MÍNIMOS

	Sistema Operativo: Windows 10
	Procesador: Intel Core i5
	Memoria RAM: 8 GB
	Tarjeta Gráfica: OpenGL 3.3
	Espacio en Disco 500 MB

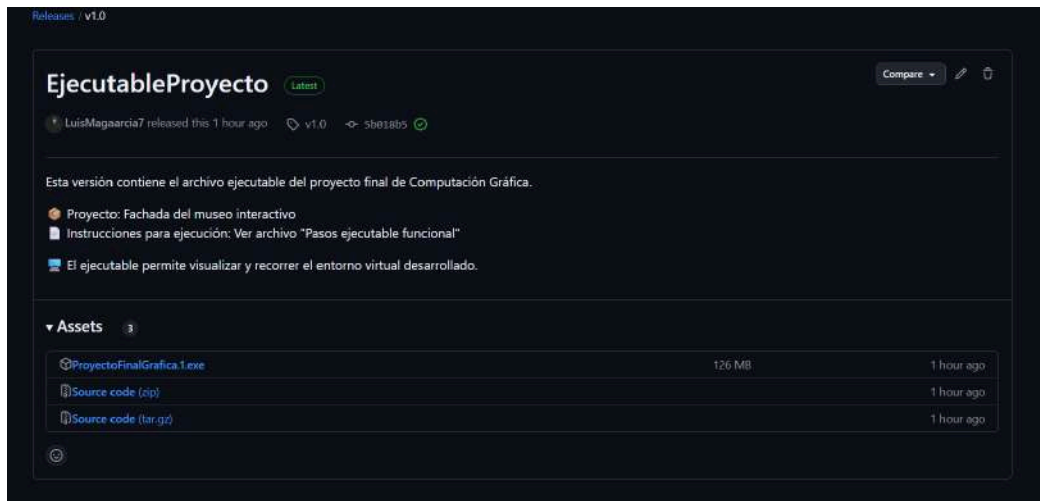
RECOMENDADOS

	Sistema Operativo: Windows 11
	Procesador: Intel Core i7
	Memoria RAM 16 GB
	Tarjeta Gráfica GTX 1660 Ti
	Resolución de Pantalla 1920x1080

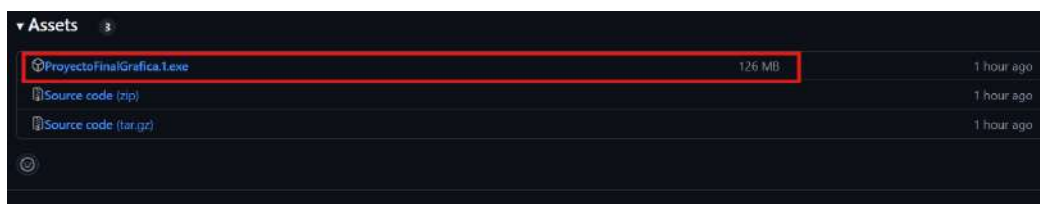
Ejecutable del proyecto.

Para ejecutar el programa del proyecto es necesario entrar al siguiente link:

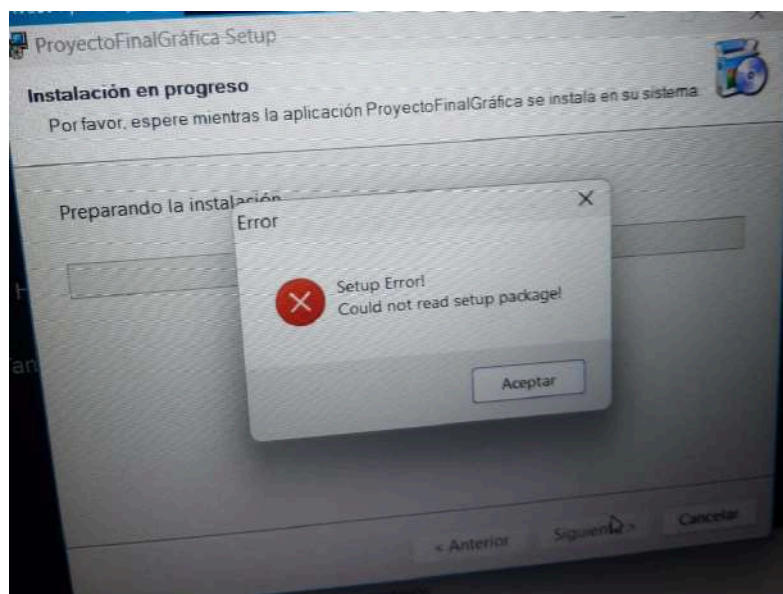
<https://github.com/LuisMagaarcia7/ProyectoComputaci-nGr-fica/releases/tag/v1.0>



Dentro del link podemos ver el ejecutable, **únicamente descargamos el archivo .exe “ProyectoFinalGráfica.1.exe”**

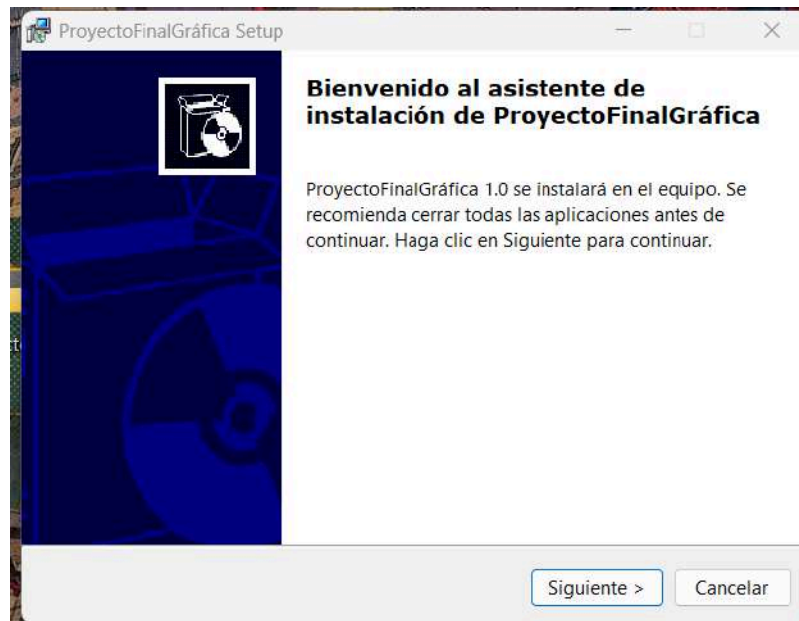


Cuando se haya descargado correctamente, se recomienda ejecutarlo como administrador y no tener algún antivirus activado, pues corre el riesgo de no dejar descargar el ejecutable adecuadamente, como se muestra en la imagen:

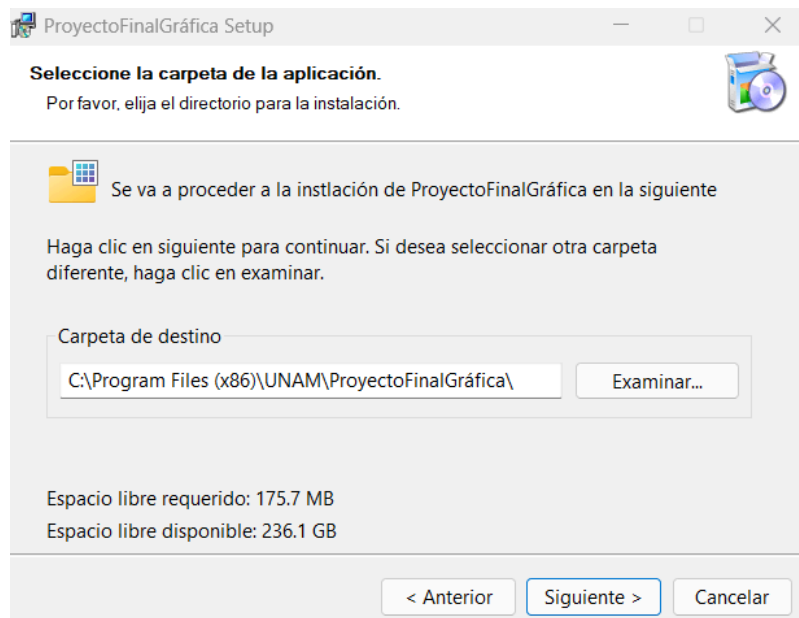


Después de ejecutar el .exe seguimos los siguientes pasos

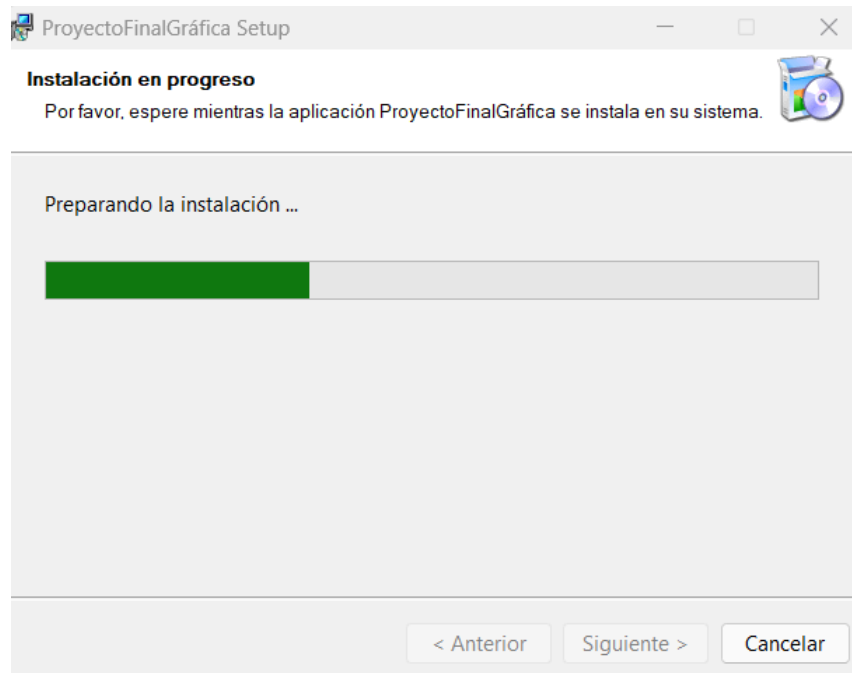
Click en siguiente



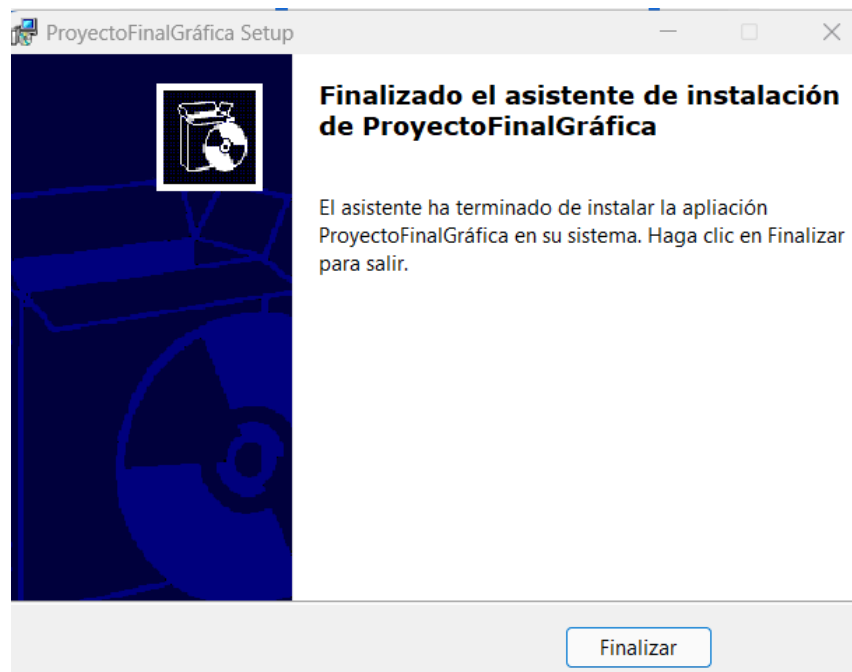
Elegimos el directorio donde se instalará el programa, en este caso, el mio es el siguiente, damos lo siguiente.



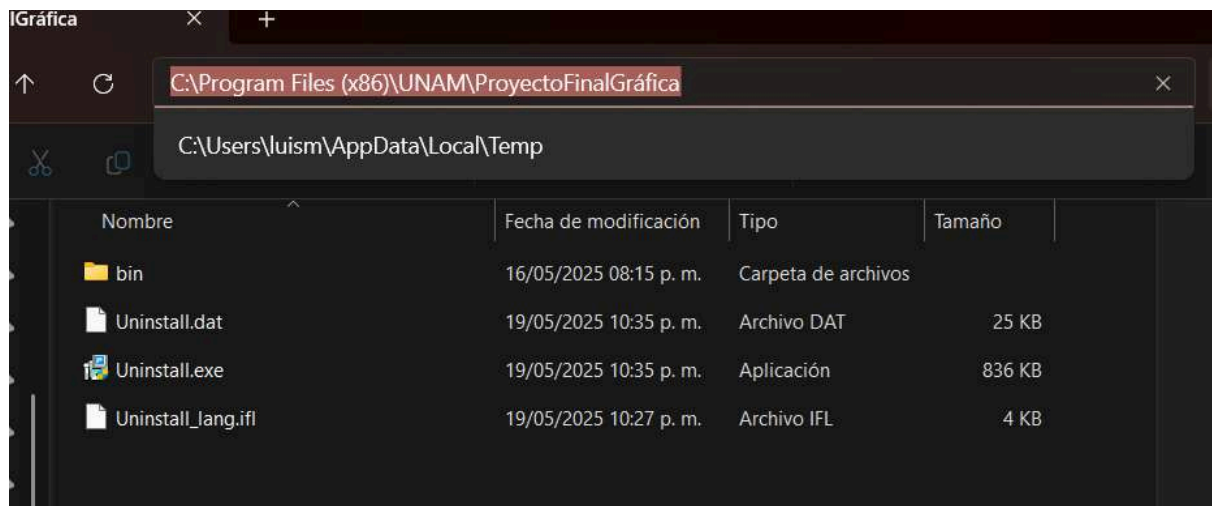
Comienza la instalación



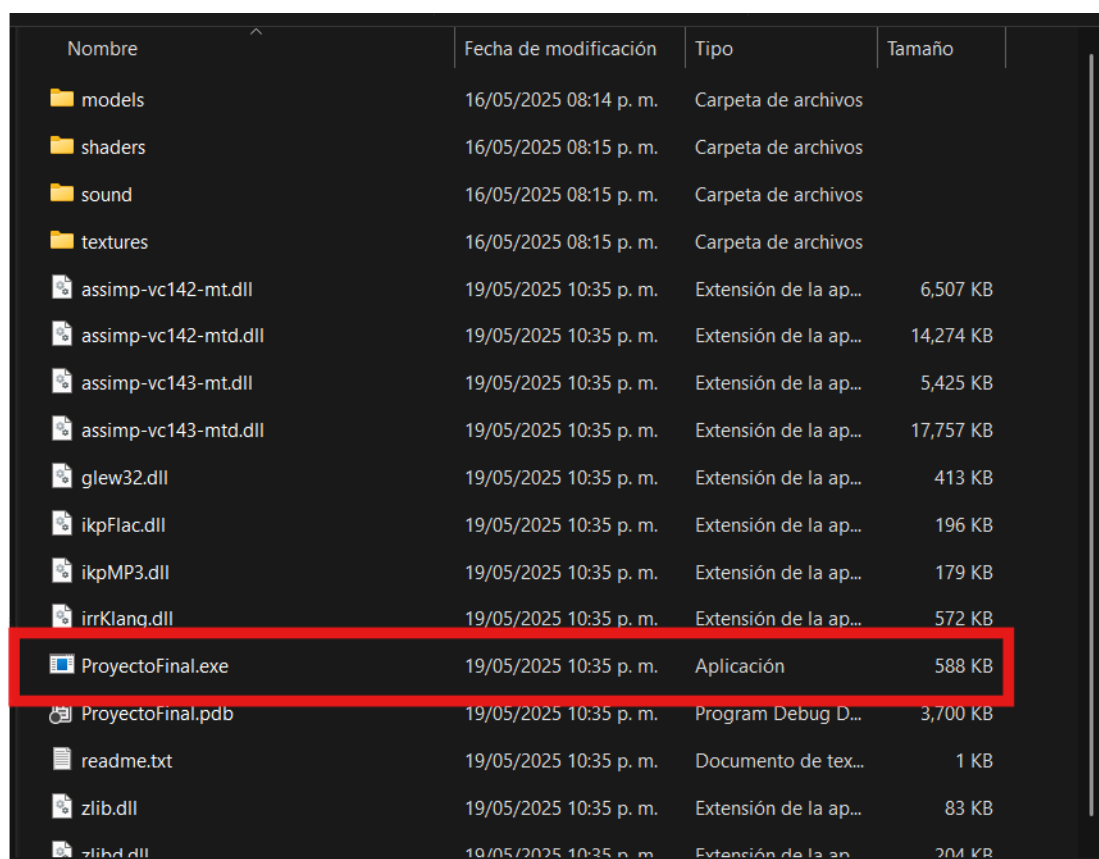
Damos finalizar



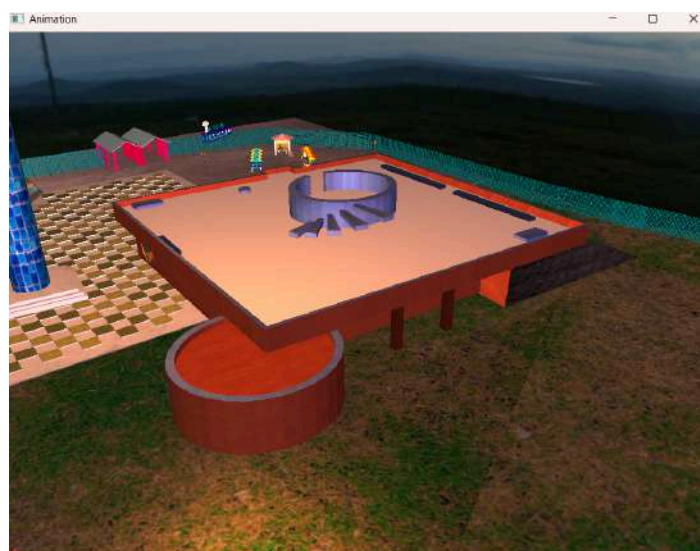
Cuando se haya instalado el ejecutable, nos vamos al directorio donde elegimos que se instalará, podemos ver que cuenta con un desinstalador también, por si el usuario desea desinstalar el ejecutable. Después, damos click en la carpeta “bin”



En la carpeta bin, podemos ver diversas carpetas y archivos, pero el que nos interesa, es el que no va a abrir el proyecto, damos click “ProyectoFinal.exe” esto nos abrirá el proyecto.



Podemos ver como se abre correctamente el ejecutable, para empezar a navegar por nuestro proyecto final.



1. Movernos por el entorno

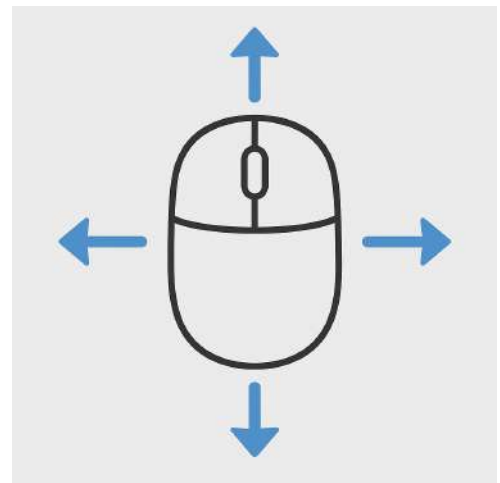
Para poder movernos por todo el entorno es necesario usar las teclas A, S, W, D, Q y E de nuestro teclado.

- W → Avanzar
- S → Retroceder
- D → Movimiento a la derecha
- A → Movimiento a la izquierda
- Q → Bajar
- E → Subir



Además del desplazamiento con el teclado, la cámara se controla con el movimiento del mouse, lo que permite al usuario observar libremente en todas las direcciones.

- **Mover el mouse hacia la derecha o izquierda:** rota la vista horizontalmente (giro de cámara).
- **Mover el mouse hacia arriba o abajo:** cambia el ángulo vertical de la cámara.



2. Abrir y cerrar puerta del Museo

Para esta primera animación, podemos observar la puerta del museo, para poder abrir y cerrar la puerta, es necesario usar las teclas H y J del teclado.

- J → Abrir puerta
- H → Cerrar puerta



Puerta normal



Puerta abierta presionando “J”



Puerta cerrada presionando “H”



3. Accionar ventiladores del museo

Para activar los 2 ventiladores que están dentro del museo es necesario presionar las teclas T y G de nuestro teclado.

- T → Activar los ventiladores (empezarán a girar)
- G → Detener el movimiento de los ventiladores (Se detiene el giro de los ventiladores)



Al presionar la tecla T, estos van a empezar a girar, si queremos detener el giro de los ventiladores presionamos la tecla G.



4. Movimiento de la trajinera

Dentro del museo, el usuario encontrará una trajinera decorativa como parte de la sala dedicada a la cultura agrícola y lacustre de Xochimilco.

Esta trajinera cuenta con una animación interactiva que se activa de forma automática con base en la proximidad visual del usuario:

- Cuando el usuario se acerca a la trajinera con la cámara, esta comienza a girar lentamente, permitiendo observar sus detalles desde distintos ángulos sin necesidad de interacción adicional.
- Al alejarse de la trajinera, la animación se detiene automáticamente, simulando una pausa natural en su movimiento.



Trajinera sin acercamiento



Al acercarnos a la trajinera desde cualquier ángulo empieza a girar, para apreciarla mejor



Al alejarnos de la trajinera, podemos observar que está detiene su giro, pues damos a entender que ya no la vamos a apreciar.



5. Animación fantasma

Dentro del museo podemos observar que tenemos 3 secciones de libros. La animación está directamente en estos libros de aquí, los de color naranja.



Esta interacción tiene como finalidad añadir un momento de sorpresa y fantasía dentro del entorno museográfico, sin alterar el contenido educativo del resto del recorrido.

- **Acción para activarlo:** clic izquierdo del mouse sobre los libros.
- **Condición de activación:** solo se activa si el usuario apunta y da clic sobre ellos
- **Resultado:** el fantasma aparece y realiza una animación predefinida



Podemos observar los libros, al momento de darle click a los libros podemos ver como sale la animación de un fantasma.

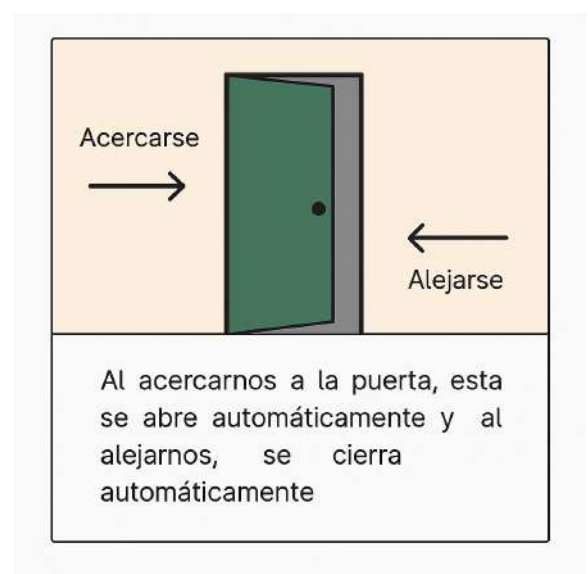


6. Abrir y cerrar puerta del baño

Dentro del entorno del museo, el usuario encontrará una puerta automática que da acceso al área del baño. Esta puerta cuenta con un sistema de apertura y cierre completamente autónomo, basado en la proximidad del usuario, lo que permite un recorrido fluido y sin necesidad de interacción manual.

Las acciones funcionan de la siguiente forma:

- Al acercarse a la puerta desde el exterior, esta se abre automáticamente, permitiendo el acceso al baño.
- Una vez que el usuario ha ingresado, la puerta se cierra sola detrás de él.
- Para salir del baño, basta con acercarse nuevamente a la puerta desde el interior; esta se abre de forma automática.



- Al alejarse de la puerta tras salir, esta se vuelve a cerrar sola.

Puerta normal



Puerta se abre sola al acercarme



Puerta se cierra al entrar al baño



7. Ver geometría del museo y parque

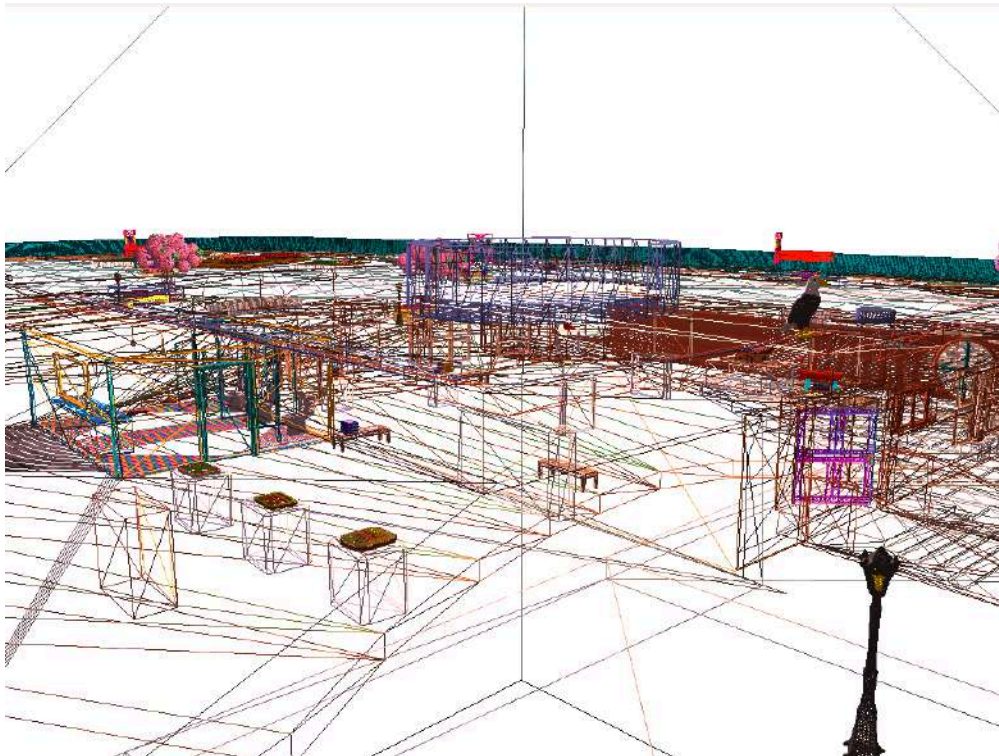
Para poder apreciar la geometría completa del museo y del parque, el usuario puede activar el modo de visualización de malla presionando la tecla M.

Este modo muestra la estructura interna de los modelos, permitiendo observar:

- La distribución espacial de los objetos
- La construcción poligonal de los elementos
- La organización técnica de cada sala y del entorno natural



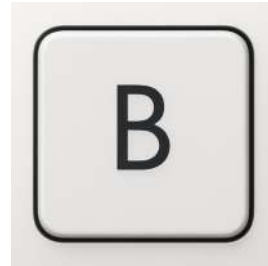
Con la tecla N se regresa al modo normal del entorno.



8. Ver puntos del Museo y parque

El sistema permite activar un modo especial que muestra todos los puntos geométricos (vértices o vértices) del museo y del parque. Esto permite observar con precisión cómo están distribuidos los objetos y su estructura detallada en el espacio 3D.

- Tecla B → Activar la vista de puntos.
- Tecla N → Regresar a la normalidad.



ANÁLISIS DE COSTOS

Costos Fijos

La siguiente tabla muestra los elementos esenciales utilizados durante todo el desarrollo del proyecto, como el equipo de cómputo, software y servicios básicos. La mayoría de las herramientas fueron gratuitas, lo que permitió optimizar recursos, y se consideraron también los gastos en electricidad e internet.

Categoría	Elemento	Costo aproximado (MXN)	Notas
Hardware	Laptop personal (HP Victus)	\$14,000	Equipo de uso propio, 16 GB RAM, Ryzen 5, SSD y GPU integrada
Software	Blender	\$0	Software libre para modelado 3D
	Visual Studio Code / OpenGL	\$0	IDE gratuito + API gráfica de código abierto
	stb_image.h / GLM / GLFW	\$0	Librerías gratuitas para carga de texturas, cámaras y ventanas
	GitHub	\$0	Plataforma gratuita para control de versiones
	Audacity / GIMP	\$0	Software libre para edición de audio e imágenes
Servicios	Electricidad (2 meses)	\$750	Estimado por consumo en equipo de desarrollo

Internet (2 meses)	\$500	Conexión necesaria para documentación, actualizaciones y pruebas
--------------------	-------	--

Costos Variables

Esta tabla presenta los costos relacionados con elementos que fueron creados específicamente para este proyecto, como la fachada del museo, objetos modelados y el diseño sonoro. Aunque muchos recursos fueron desarrollados internamente, se asigna un valor comercial estimado para reflejar su aporte real. El sonido se generó con herramientas libres, por lo que no representó un gasto adicional.

Categoría	Elemento	Costo aproximado (MXN)	Notas
Modelado 3D	Fachada principal	\$900	Equivalente comercial aproximado
	Objetos modelados (20 elementos aprox)	\$3,600	\$180 c/u
Sonido	Efectos ambientales y de objetos	\$0	Grabados con herramientas libres, sin costo comercial

Mano de Obra

Categoría	Actividad	Tiempo estimado	Tarifa	Subtotal
Desarrollo	Programación e interacción	60 horas	\$100/hora	\$6,000
Documentación	Manual técnico,	10 horas	\$100/hora	\$1,000

	manual de usuario			
Diseño gráfico	Texturizado y generación de imágenes	20 horas	\$100/hora	\$2,000

Costo Total Estimado

A continuación se presenta el total acumulado del proyecto, sumando los costos fijos, variables y la mano de obra. Esta cifra representa una estimación general del valor real de producción del museo virtual, considerando tanto recursos materiales como el tiempo invertido.

Concepto	Costo (MXN)
Hardware y servicios	\$15,250
Modelado y recursos gráficos	\$4,500
Mano de obra (dev + docs + diseño)	\$9,000
Total estimado del proyecto	\$28,750

Precio de Venta Sugerido

Con base en el costo total estimado del proyecto, se calcula un margen de ganancia del 25% para establecer un precio competitivo y justo

Concepto	Costo (MXN)
Costo base del proyecto	\$28,750
Margen de ganancia (25%)	\$7,187
Precio de venta final	\$35,937

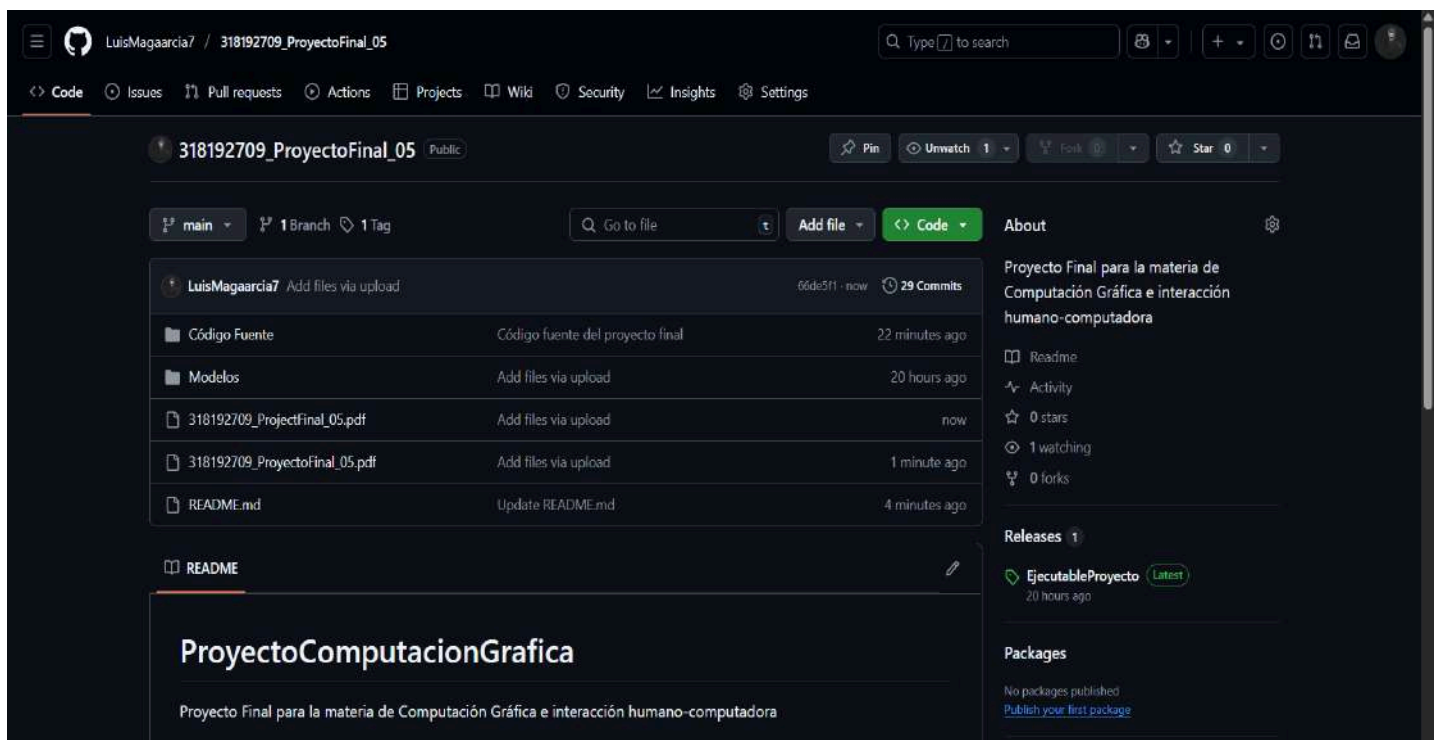
REPOSITORIO EN GITHUB

Link:

<https://github.com/LuisMagaarcia7/ProyectoComputacionGrafica/tree/main>

Dentro del repositorio de GitHub se encuentra todo el proyecto:

- Modelos utilizados
- Código fuente
- Ejecutable (ubicado en releases)
- Documentación en español
- Documentación en inglés

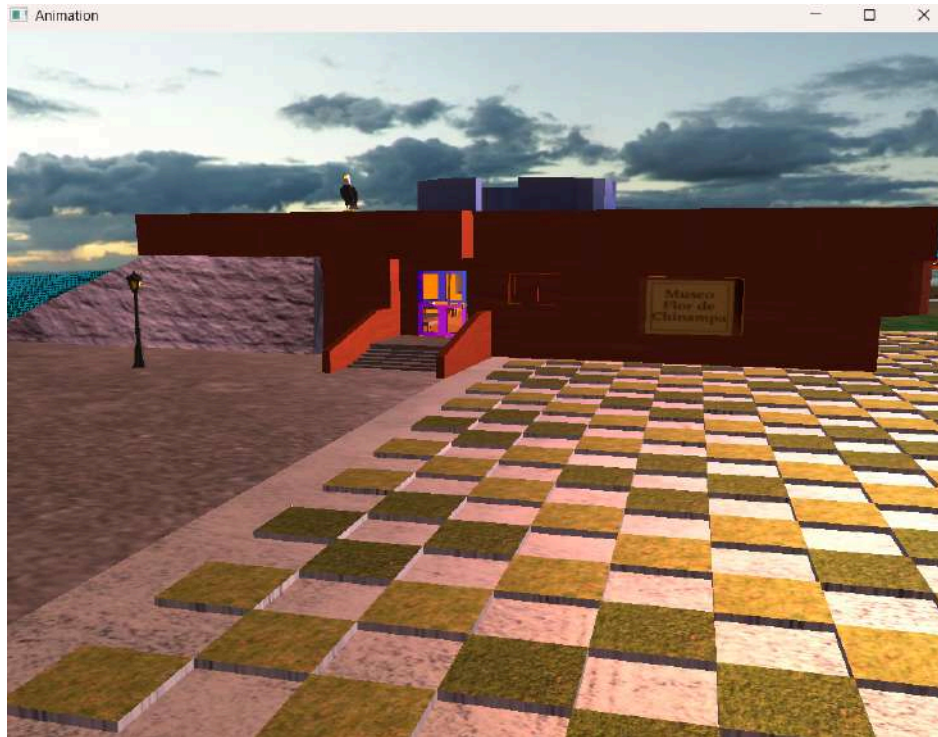


VIDEO DEMOSTRATIVO FACHADA

<https://drive.google.com/file/d/1YgZOBdug4wkQrNEltVO0BVuHVG84w4yE/view?usp=sharing>

RESULTADO FINAL

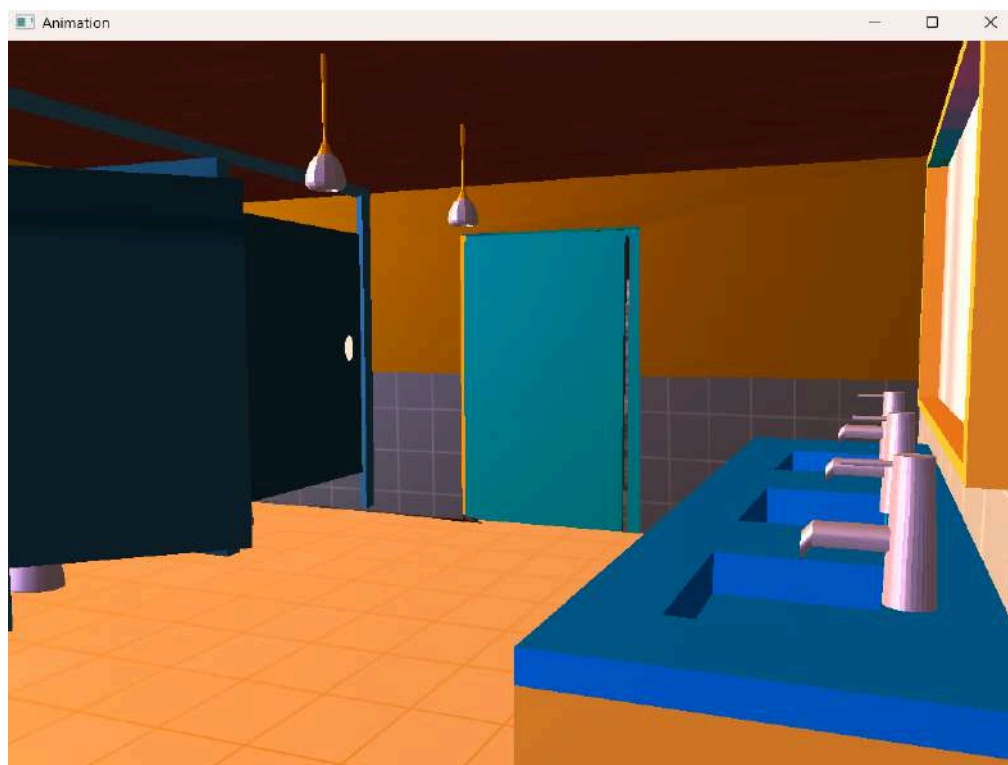
En esta sección se subirán capturas de pantalla del proyecto en su etapa final, mostrado como se ve ya todo lo trabajado en el ejecutable.

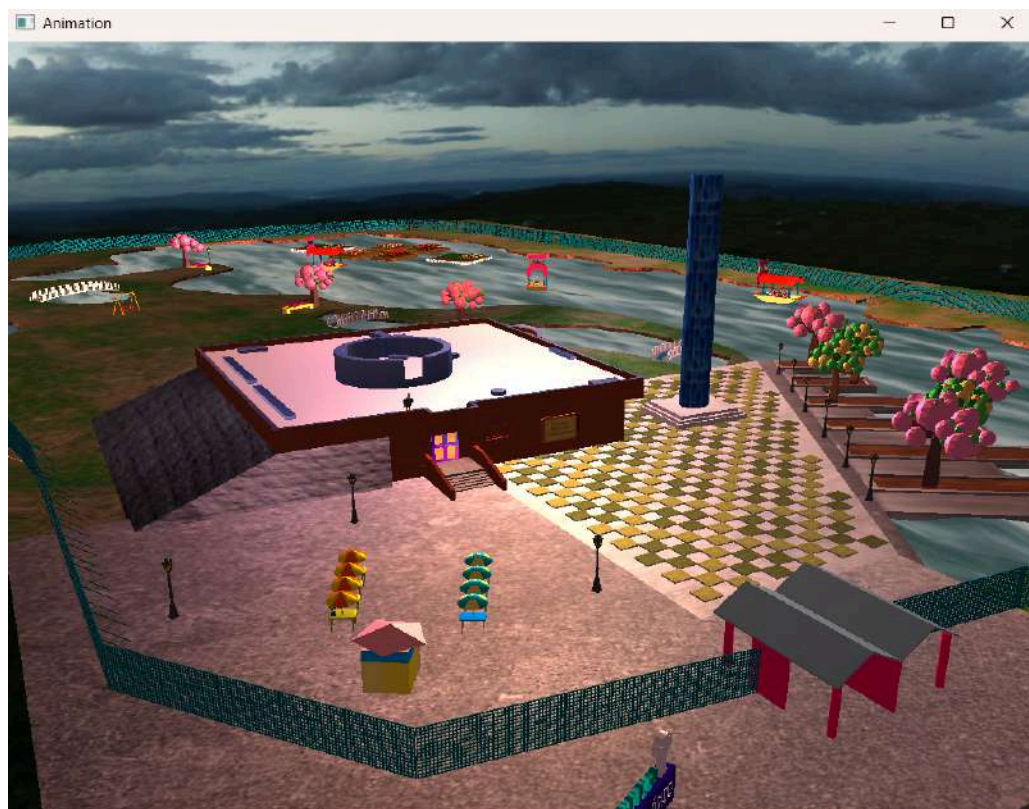
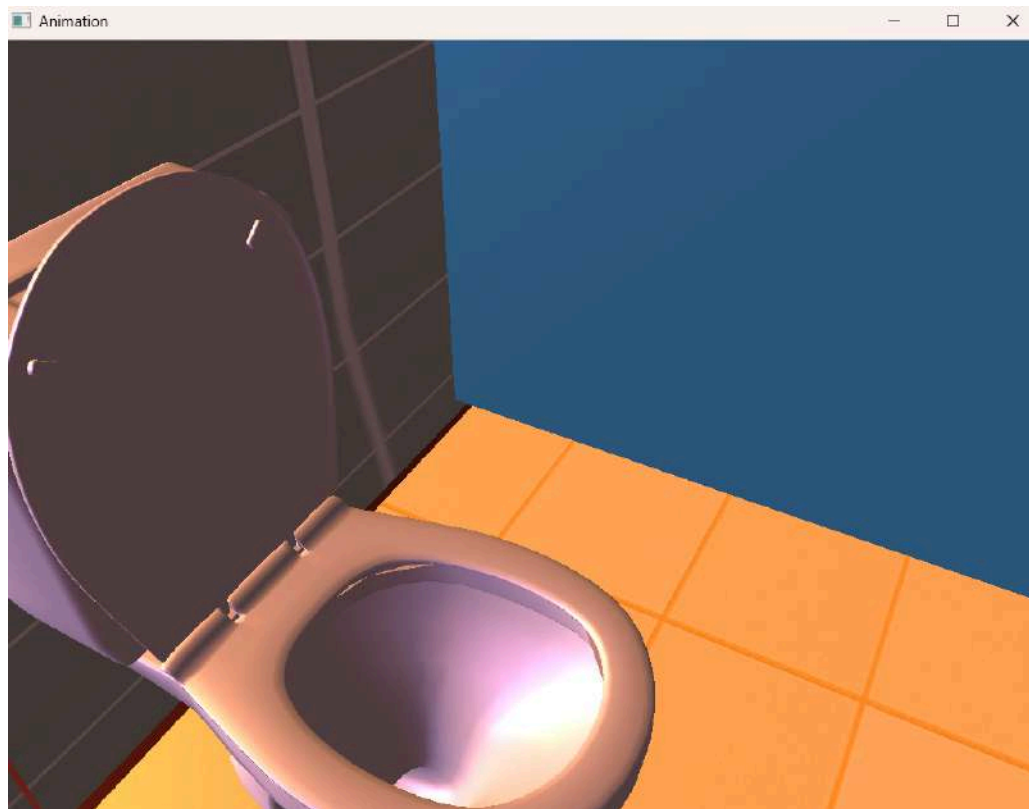












CONCLUSIONES

Este ha sido uno de los proyectos más largos y desafiantes que he llevado a cabo en mi carrera. Sin embargo, también ha sido uno de los más fascinantes, tanto por su complejidad como por la oportunidad que me brindó de aplicar de manera integral todo lo que he aprendido en la materia de Computación Gráfica, tanto en teoría como en práctica. No se trató solo de cumplir con un requisito, sino de entender y ejecutar todo un pipeline gráfico completo, desde la conceptualización hasta la ejecución técnica.

Durante el desarrollo del proyecto, me enfrenté al reto de aprender a modelar objetos desde cero en Blender, gestionar mi tiempo para completar cada etapa, y aplicar técnicas específicas que vimos en clase, como el mapeo UV, el texturizado, la cámara sintética, la iluminación, la carga de modelos .fbx, la implementación de shaders y la integración de sonido ambiental. Cada componente, desde la fachada del museo hasta los elementos interactivos y las animaciones, fue desarrollado con base en lo aprendido, siempre buscando una coherencia visual y funcional.

Uno de los aspectos que más disfruté fue ver cómo todo iba tomando forma a medida que se integraban los modelos con sus texturas, los efectos de iluminación y las animaciones reactivas, como la trajinera que gira al acercarse o la aparición del fantasma al interactuar con el libro. Todo esto fue posible gracias al uso de OpenGL, que, con sus librerías (GLFW, GLAD, GLM, stb_image, irrKlang), nos permitió construir un entorno 3D completo y visualmente atractivo. La lógica implementada para controlar la cámara, las teclas de interacción y las respuestas visuales y sonoras resultó en una experiencia muy satisfactoria.

Más allá del desarrollo técnico, el proyecto me ayudó a comprender el esfuerzo que implica crear un entorno gráfico interactivo. Me hizo ver con nuevos ojos el mundo de los videojuegos y las simulaciones, valorando lo complejo que puede ser lograr que algo tan "simple" como abrir una puerta o encender una luz se sienta tan natural.