UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTY OF ENGINEERING

FINAL PROJECT
THREE-DIMENSIONAL RECONSTRUCTION

USER MANUAL

COMPUTER GRAPHICS AND HUMAN-COMPUTER INTERACTION
SEMESTER: 2025-02

PREPARED BY

GONZÁLEZ NAVA ALICIA AISLINN
ROMÁN JIMÉNEZ ADOLFO
LEE OBANDO ILEANA VERÓNICA

INSTRUCTOR:

ENG. LUIS SERGIO VALENCIA CASTRO

LAB GROUP: 10
THEORY GROUP: 03

PREFACE

This user manual accompanies the final project for the course *Computer Graphics and Human–Computer Interaction*, titled **Three-Dimensional Reconstruction**. The main objective of this project is to apply the theoretical and practical knowledge acquired throughout the semester to develop a high-quality graphical system, grounded in thoughtful analysis and design, while also promoting collaborative work as part of the students' professional training.

The project consists of the development of an interactive, three-dimensional virtual tour of the second floor of Building Q at the Faculty of Engineering, Universidad Nacional Autónoma de México (UNAM). Implemented using C++, OpenGL 3.0+, and GLFW, this immersive application enables users to explore the environment through first-person camera controls and perspective view navigation.

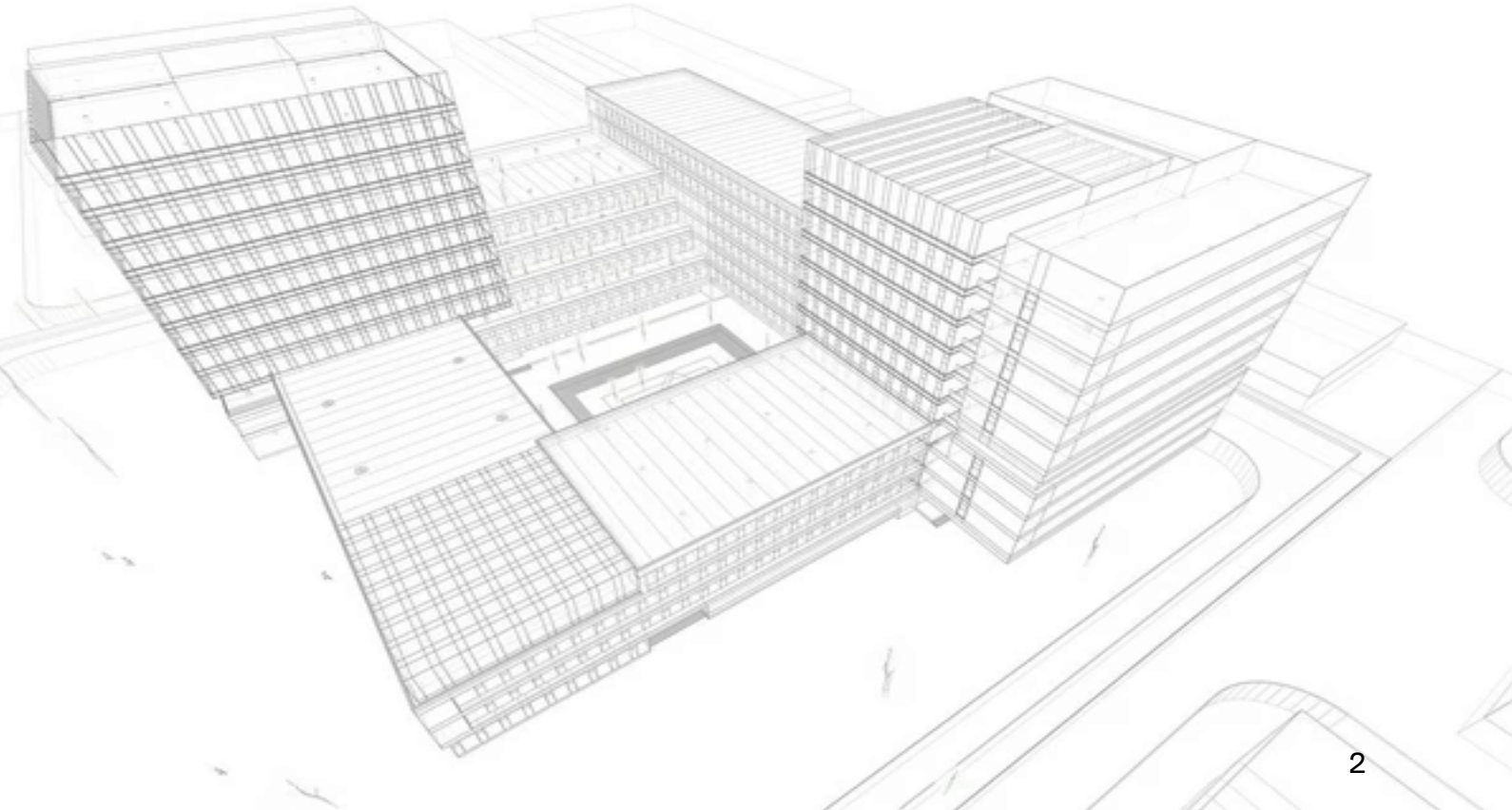
Among the project's specific goals are the geometric and hierarchical modeling, texturing, and animation of key spaces such as the Microsoft Artificial Intelligence Lab, the CISCO Classroom, and the Computer Graphics Laboratory. The user experience is enhanced by intuitive navigation shortcuts, isometric viewing options, environmental soundtracks, and animated character models representing students and professors interacting within the space.

This document also includes a detailed installation guide and a comprehensive explanation of the application's functionality. In addition, collaborative development tools like GitHub were used for team coordination and version control, and a technical and cost feasibility study was conducted to evaluate the potential for the application's future deployment as an educational tool on the faculty's website.

The model is designed to be visually appealing and structurally faithful, featuring realistic representations of architecture and furnishings alongside vibrant, playful colors and textures. While only three areas are fully accessible, the remaining spaces are accurately represented externally. Static environmental elements such as benches, plants, and sculptures, along with multiple complex animated avatars, contribute to a more lifelike and engaging experience.

Ultimately, this manual serves to guide users in the installation, exploration, and understanding of the software, ensuring a seamless and informative experience with the virtual tour.

CONTENTS



SYSTEM REQUIREMENTS

To ensure optimal performance and compatibility, the following system requirements are recommended for running the application:

Operating system

- Windows: Version 7 or later (64-bit recommended)
- macOS (*COMPATIBILITY HAS BEEN IMPLEMENTED, THOUGH UNTESTED. NO WARRANTY IS PROVIDED FOR MACOS SYSTEMS)*
- Linux: Not supported.

Hardware requirements

- Processor: Dual-core CPU with a minimum clock speed of 2.0 GHz (quad-core recommended).
- RAM: 4 GB or more.
- Graphics Card: A dedicated graphics card supporting OpenGL 3.0+ (e.g., NVIDIA GeForce 600 series or higher, AMD Radeon HD 7000 series or higher).
- Storage: At least 500 MB of available disk space for installation.

Software dependencies

- OpenGL: Version 3.0 or higher (for graphics rendering).
- GLFW: For window and input management, version 3.0 or higher.
- Compiler: C++11 or later (recommended: GCC, Clang, or MSVC).

NOTES

To ensure smooth performance, it is recommended to use a graphics card released after 2011. Older GPUs may not fully support all graphical features.

Installation instructions

Visual Studio is a professional integrated development environment (IDE) developed by Microsoft, widely used for building high-performance applications in multiple programming languages such as C++, C#, and Python. Its powerful suite of tools for code editing, debugging, project management, and integration with version control systems makes it an ideal platform for both academic and industry-level software development.

For this project, Visual Studio is required to compile and run the application, which is developed in C++ using OpenGL and GLFW. The following versions are supported:

Visual Studio 2017
Visual Studio 2019
Visual Studio 2022

Note: *It is only necessary to install one version, depending on your system's specifications and compatibility.*

Step-by-Step installation guide

1. **Download Visual Studio**

- Visit the official website: https://visualstudio.microsoft.com
- Choose the version 2022

2. **Run the installer**

- Launch the installer and wait for the setup window to open.

3. **Select workloads**

- In the Workloads tab, check the option: Desktop development with C++
  This is essential, as the application is written in C++ and relies on desktop application development tools.

4. **Complete installation**

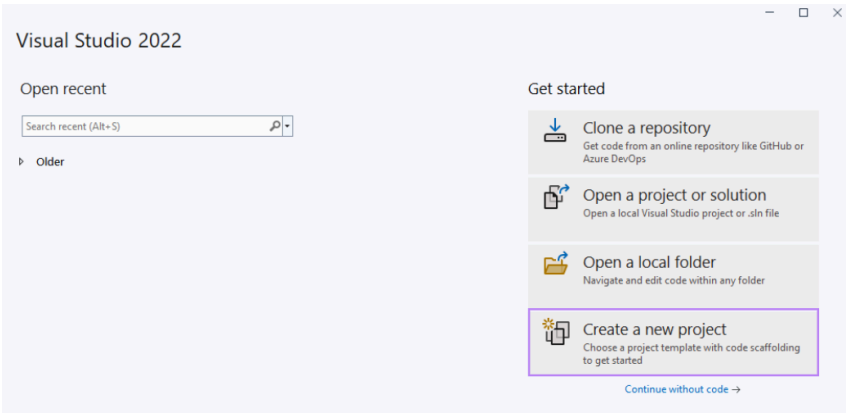
- Proceed with the installation and restart your computer if prompted.

Workspace configuration (Visual Studio 2022)

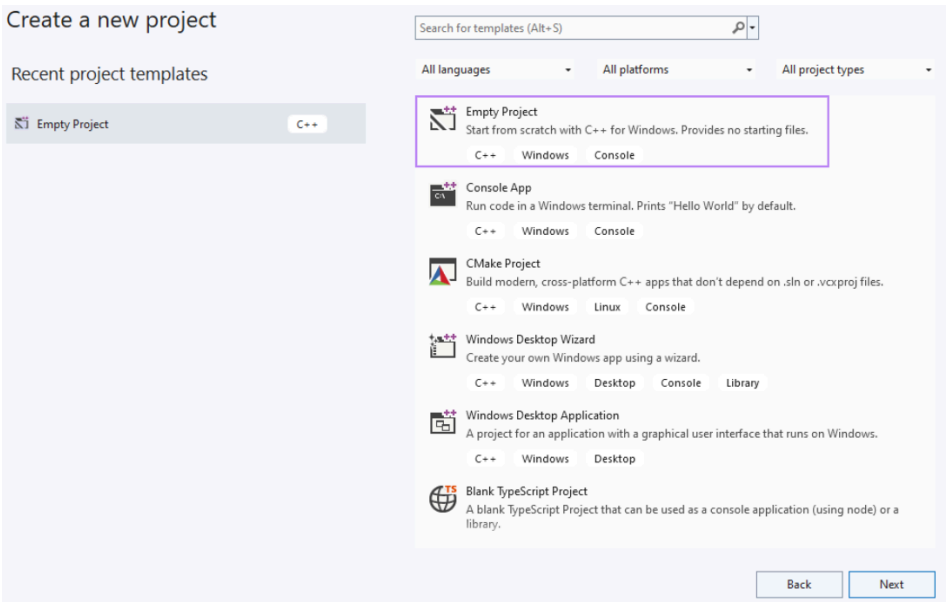
Follow the steps below to correctly set up your workspace and prepare the project for compilation and execution in Visual Studio 2022.

Step 1: Create a new project

1. Open Visual Studio 2022.
2. Select "Create a new project."



3. From the list of templates, under C++ > Windows > Console, choose Empty Project.



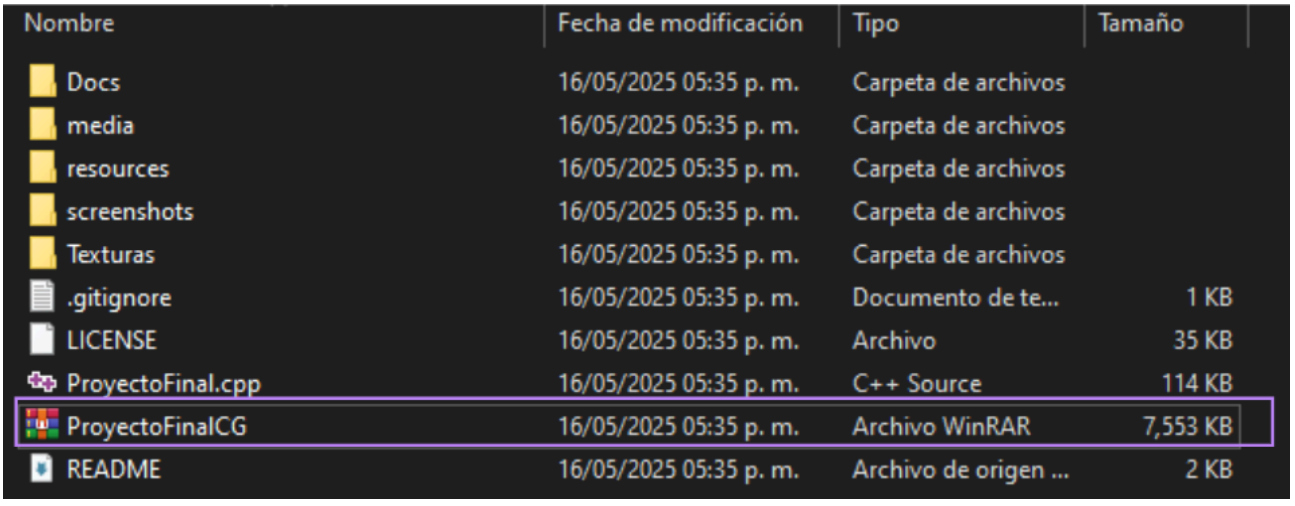
Important: This template is available only if the Desktop development with C++ workload was selected during installation.

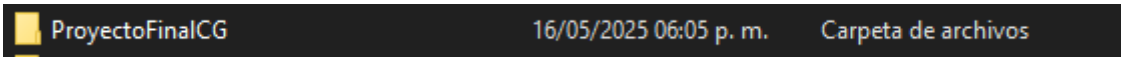
Step 2: Add project files

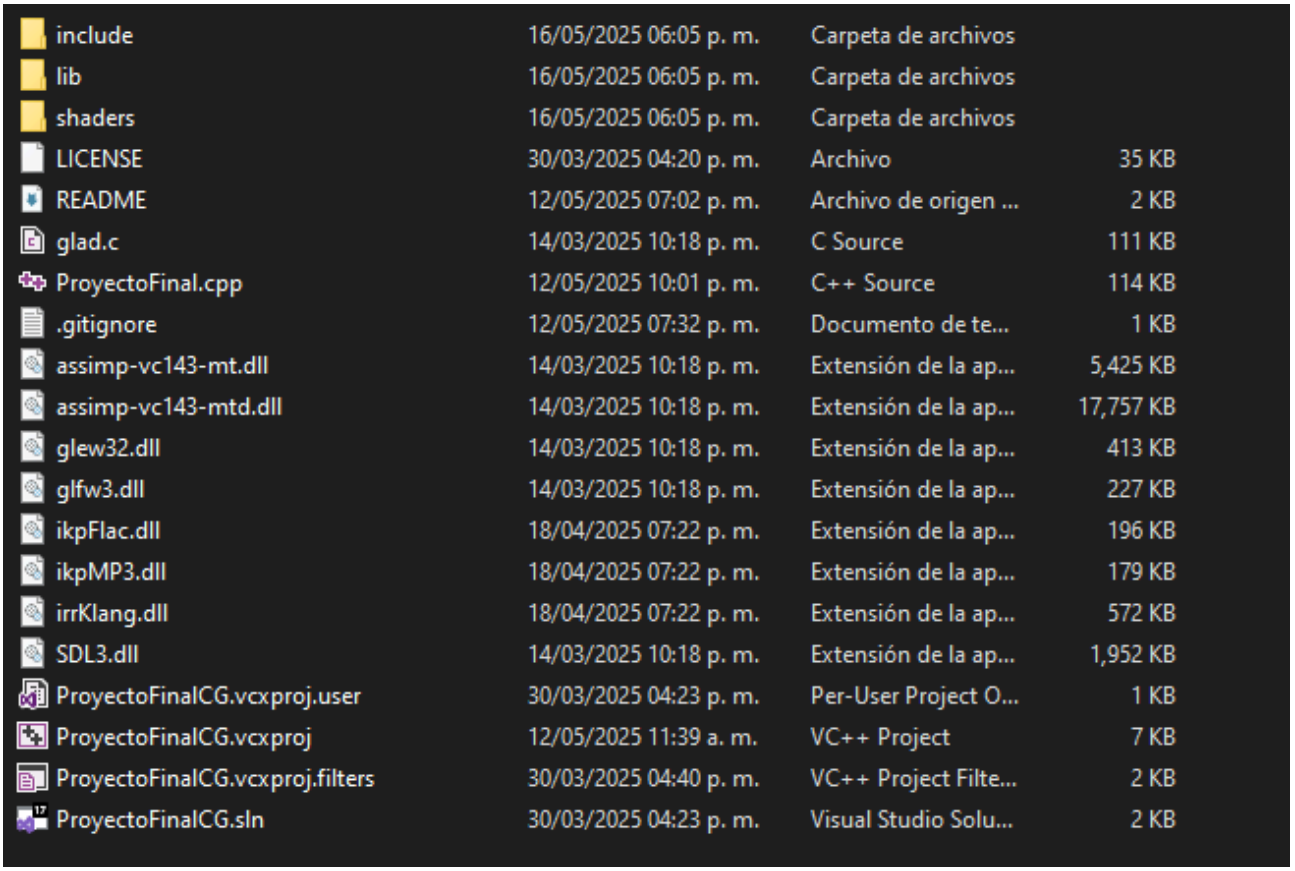
1. After downloading the project from the repository link on GitHub:

   Download our provided file for the project (ProyectoFinalCG.rar) and extract it's full content to the working folder of your preference. The files downloaded inside the project folder include things like:

   - Libraries (`.lib`)
   - Header files (`.h`)
   - `glad.c`
   - Dynamic link libraries (`.dll`)
   - irrKlang (audio library)
   - Shader programs located in the `shaders/` folder
   - Any other dependencies provided by the instructor







include	16/05/2025 06:05 p. m.	Carpeta de archivos	
lib	16/05/2025 06:05 p. m.	Carpeta de archivos	
shaders	16/05/2025 06:05 p. m.	Carpeta de archivos	
LICENSE	30/03/2025 04:20 p. m.	Archivo	35 KB
README	12/05/2025 07:02 p. m.	Archivo de origen ...	2 KB
glad.c	14/03/2025 10:18 p. m.	C Source	111 KB
ProyectoFinal.cpp	12/05/2025 10:01 p. m.	C++ Source	114 KB
.gitignore	12/05/2025 07:32 p. m.	Documento de te...	1 KB
assimp-vc143-mt.dll	14/03/2025 10:18 p. m.	Extensión de la ap...	5,425 KB
assimp-vc143-mtd.dll	14/03/2025 10:18 p. m.	Extensión de la ap...	17,757 KB
glew32.dll	14/03/2025 10:18 p. m.	Extensión de la ap...	413 KB
glfw3.dll	14/03/2025 10:18 p. m.	Extensión de la ap...	227 KB
ikpFlac.dll	18/04/2025 07:22 p. m.	Extensión de la ap...	196 KB
ikpMP3.dll	18/04/2025 07:22 p. m.	Extensión de la ap...	179 KB
irrKlang.dll	18/04/2025 07:22 p. m.	Extensión de la ap...	572 KB
SDL3.dll	14/03/2025 10:18 p. m.	Extensión de la ap...	1,952 KB
ProyectoFinalCG.vcxproj.user	30/03/2025 04:23 p. m.	Per-User Project O...	1 KB
ProyectoFinalCG.vcxproj	12/05/2025 11:39 a. m.	VC++ Project	7 KB
ProyectoFinalCG.vcxproj.filters	30/03/2025 04:40 p. m.	VC++ Project Filte...	2 KB
ProyectoFinalCG.sln	30/03/2025 04:23 p. m.	Visual Studio Solu...	2 KB

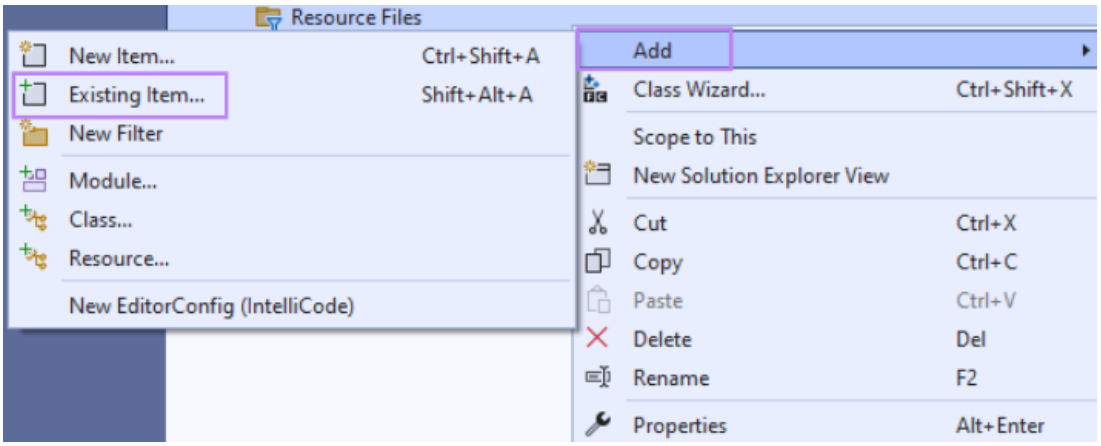
a) GitHub repository

The official GitHub repository hosts the main project structure and assets, including:

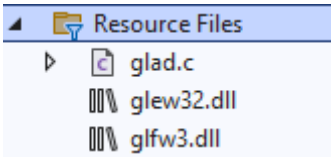
- Source code file (`.cpp`)
- `textures/` folder – containing all texture images used in the scene
- `resources/` folder – containing 3D models and environment assets

Important: Both sources are necessary. The project will only compile and run successfully if the contents from the compressed file and the GitHub repository are properly integrated into your Visual Studio workspace.

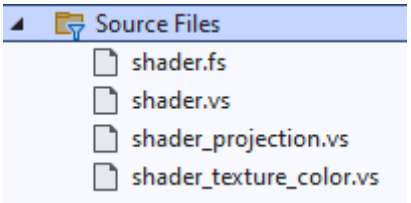
2. In the Solution Explorer, right-click on Resource Files and choose Add > Existing Item.



3. And select the following elements `glad.c`, glew32.dll, glfw3.dll and file named `ProyectoFinal.cpp`

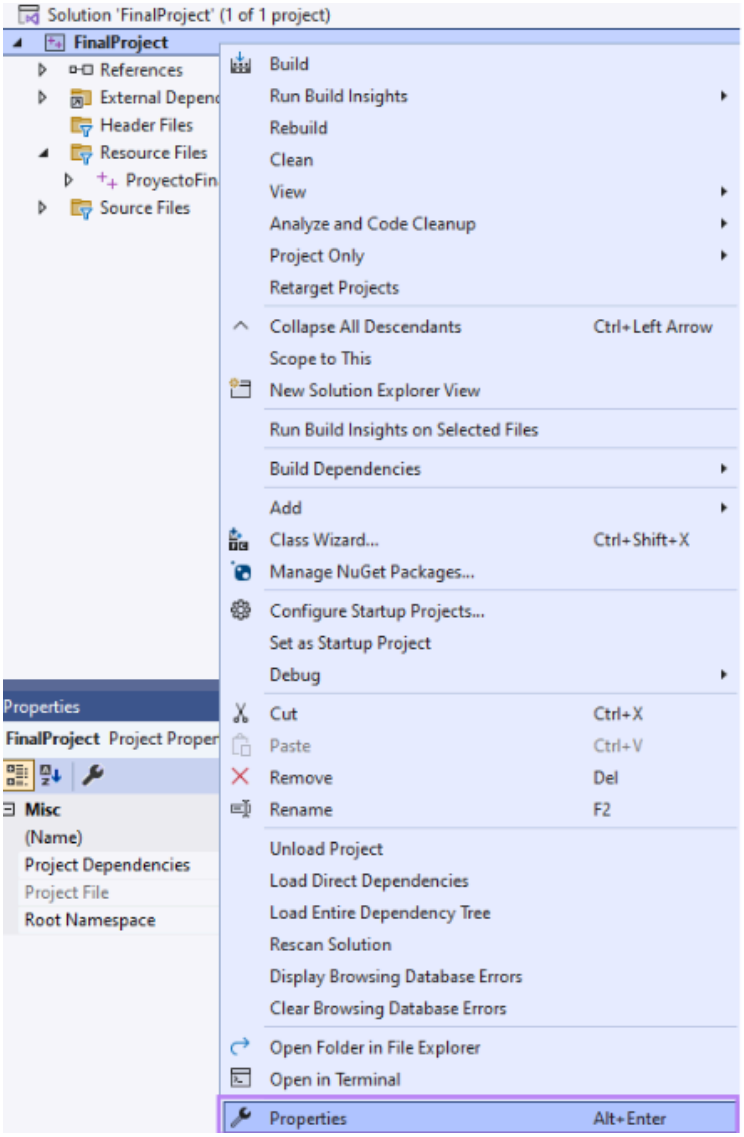


4. Repeat this process of adding the existing elements but with the Source Files section, for this you must add the following elements: shader, shader.vs, shader_projection and shader_projection.vs from the shaders folder.



Step 3: Configure project properties

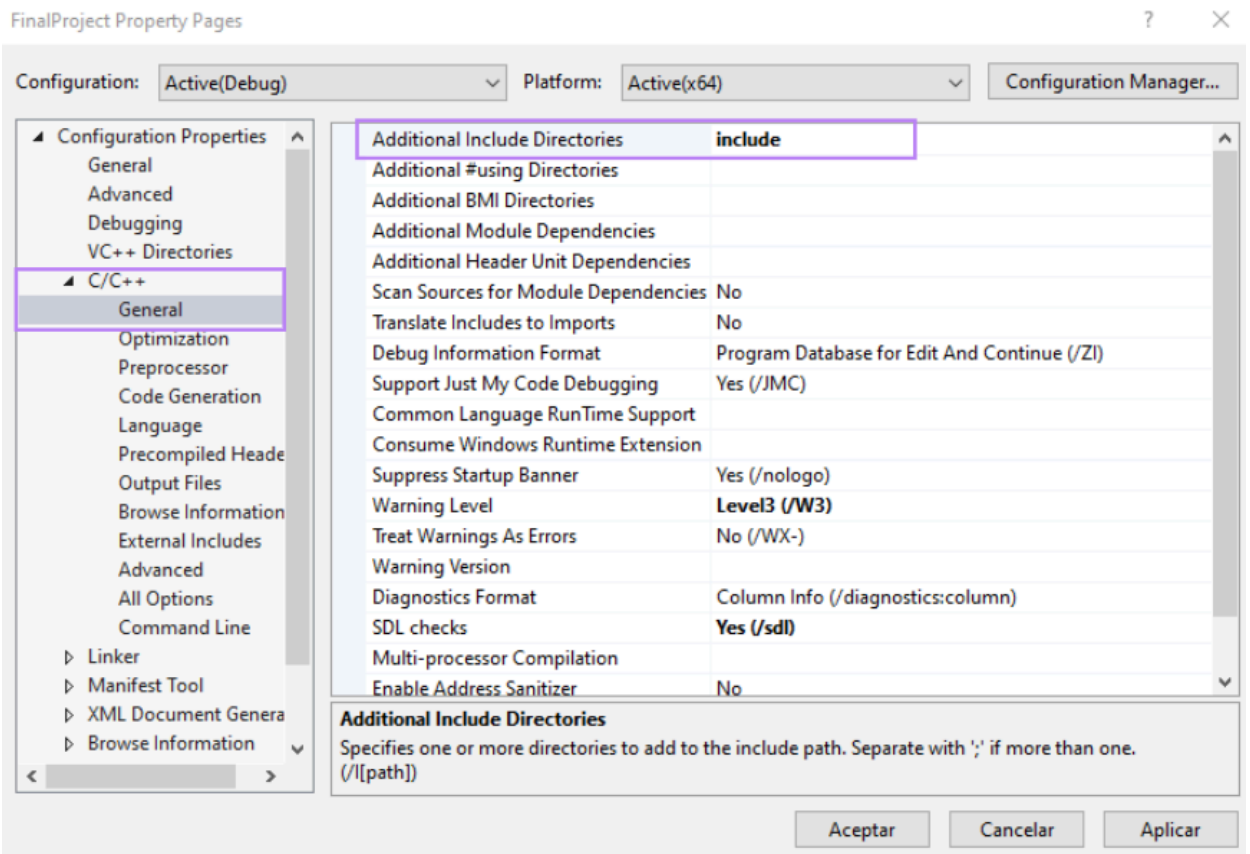
1. Right-click on the project file in Solution Explorer (`FinalProject`) and select Properties.



2. Then make the following changes:

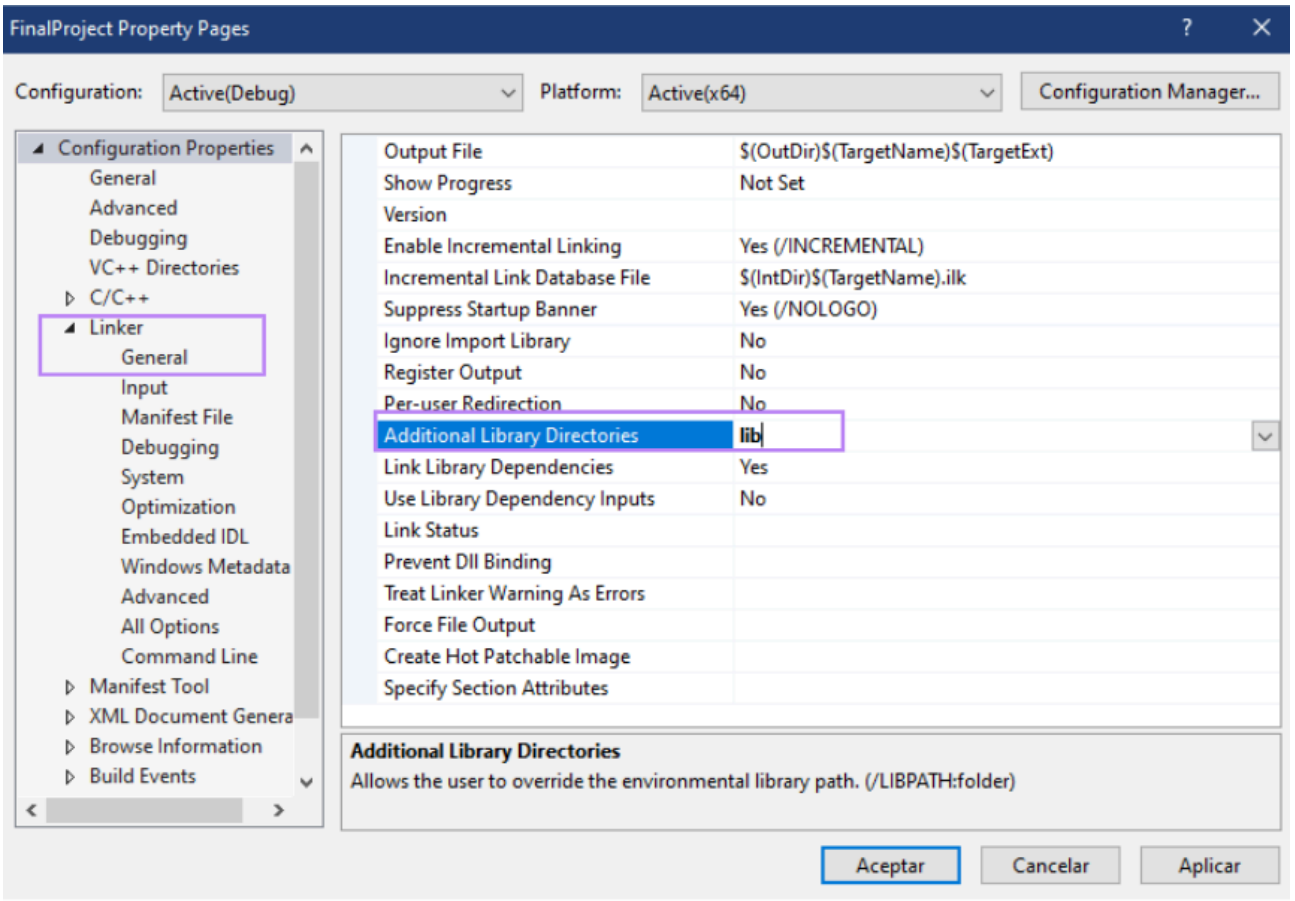
C/C++ > General

- Locate the section “C/C++” and then “General”, next add the path “`include`”.



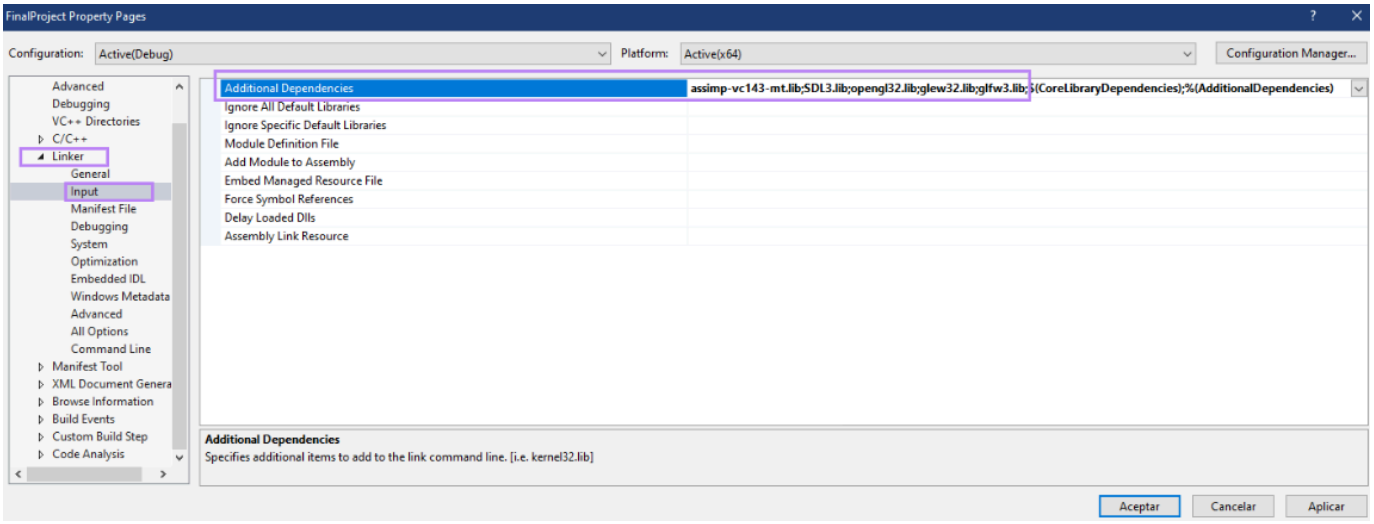
Linker > General

- Locate the section “Linker” and then “General”, next add the path “`lib`”.



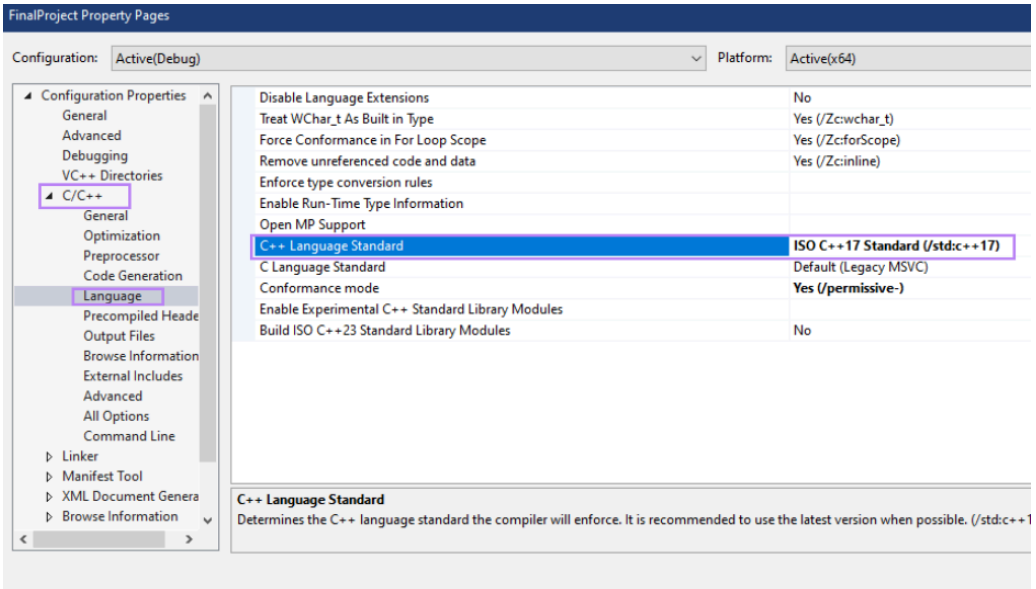
Linker > Input

- Locate the section “Linker” and then “Input”, next add the path assimp-vc143-mt.lib;SDL3.lib;opengl32.lib;glew32.lib;glfw3.lib; (Respect existing elements). And



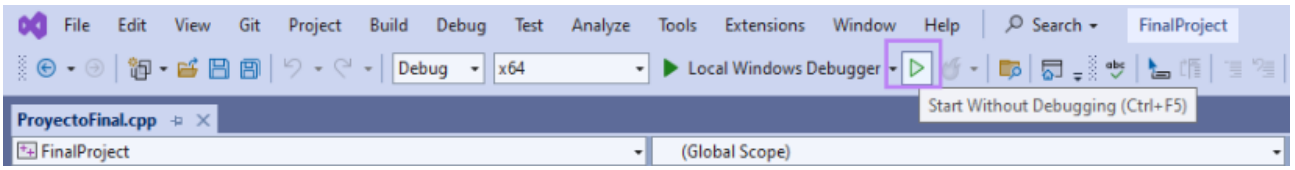
C/C++ > Language

Locate the section “Language” and then “C/C++ > Language standard”, next add the path ISO C++17 Standard (/std:c++17)



Step 4: Build and run

After completing these configurations the application should compile and run successfully. To do this, go to the navigation bar at the top of Visual Studio and select "Start without debugging" to run the program.



Controls and navigation

First-Person camera navigation

The application features a custom first-person camera system that allows users to freely explore the 3D environment. The orientation of the camera is controlled via mouse movement, while directional movement (forward, backward, left, right) is handled through keyboard input, enabling an immersive navigation experience.

Perspective and isometric views

Users can seamlessly switch between predefined camera perspectives using dedicated keyboard shortcuts:

- **Front view of the building**: The default camera position when the application launches, offering a frontal view of the structure.
- **Inside the computer graphics lab**: Positions the camera inside the lab area, providing an interior view of the environment.
- **Inside the CISCO hybrid classroom**: Moves the camera into the CISCO classroom to showcase its interior.
- **Isometric view of the scene**: Adjusts the camera to an elevated, angled position, offering a comprehensive overview of the entire scene.

Keyboard shortcuts

The keyboard is used both for movement and to switch between preset camera views. The following keys are used to access each camera preset:

- **Press 1** — Front view of the building (default).
- **Press 2** — Inside the Computer Graphics Lab.
- **Press 3** — Inside the CISCO Hybrid Classroom.
- **Press 4** — Isometric view of the scene.
- **Press 5** — Bike animation.
    - Press space and then “P”
- **Press 6** — Camera animation.
    - Press “C” to turn the animation on and off.
- **Note:** Up in the sky are the other two animations: a bird flying and the sun rotating on its own axis.