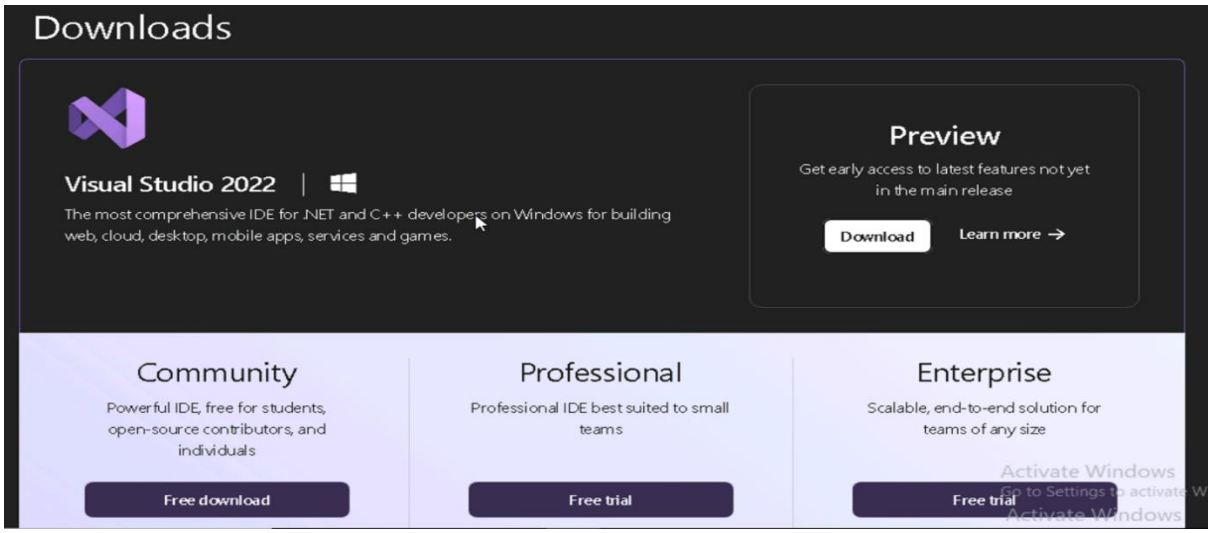
VISUAL STUDIO DOCUMENTATION

VISUAL STUDIO DOWNLOAD SITE LINK

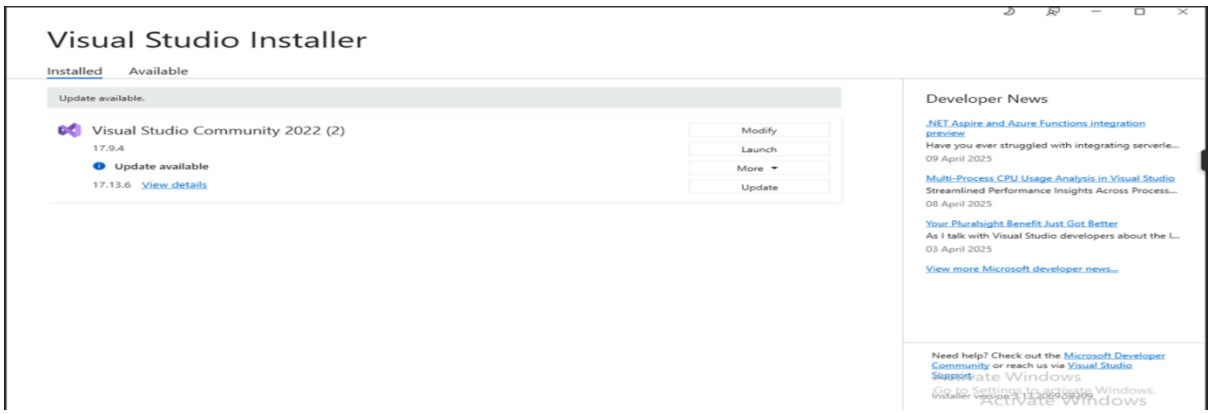
Download Visual Studio Tools - Install Free for Windows, Mac, Linux

Step 1: Click the above link to download visual studio

According to the usage choose the versions of it . For free version use Community version.



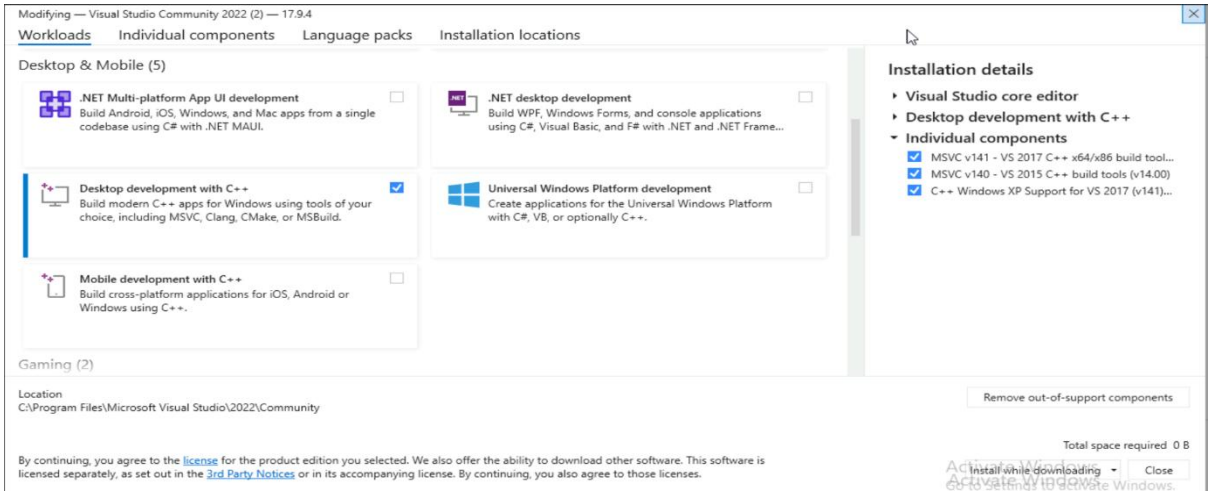
Step 2: Open Visual Studio Installer to install the required components and features.



Then select modify to make changes.

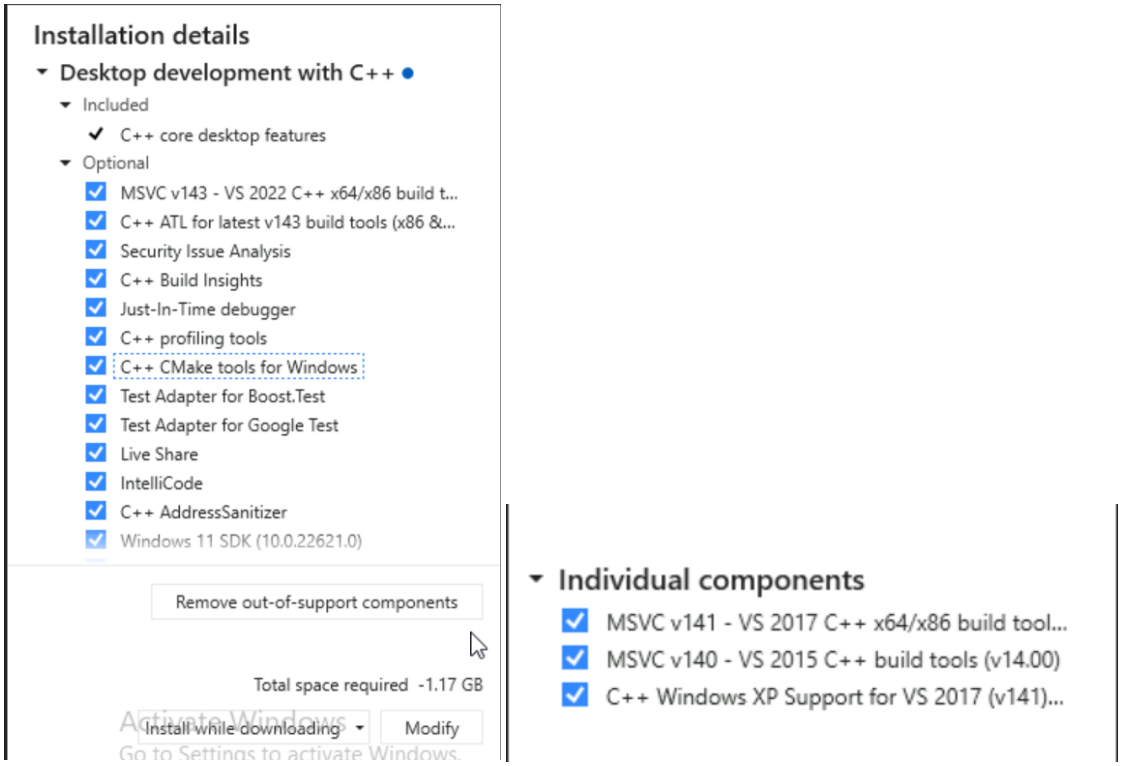
Step 3: Select Desktop development with C++ on Workloads.

which provides the tools needed for developing Windows desktop applications that can run on Windows.

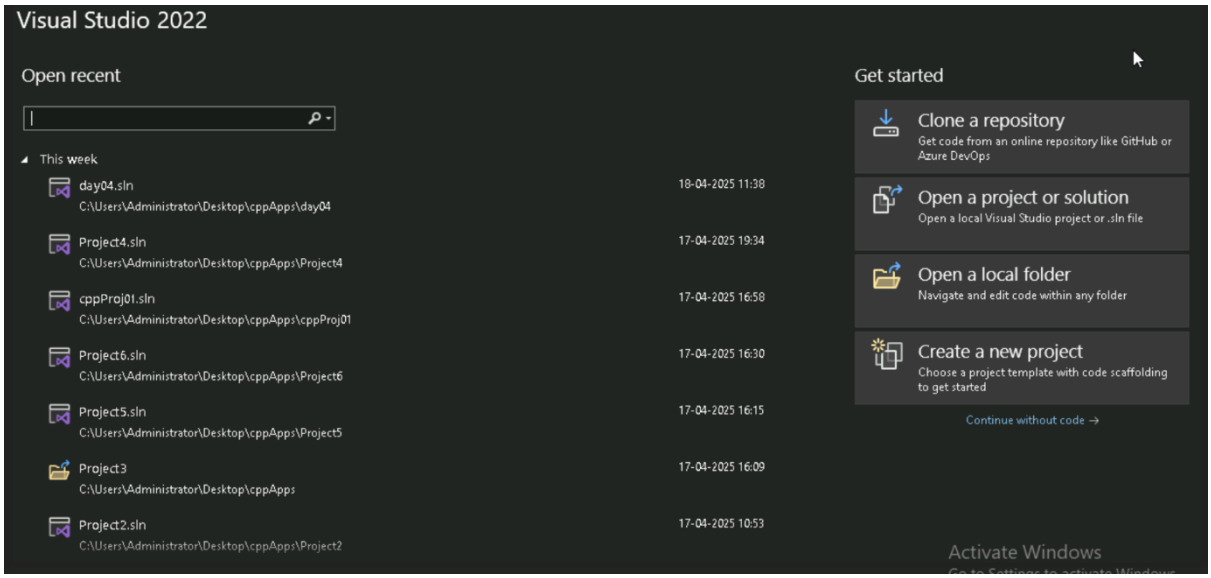


When setting up the C++ Windows desktop workload, you can proceed the installation by choosing specific tools, Windows SDKs, and additional features to suit your needs.

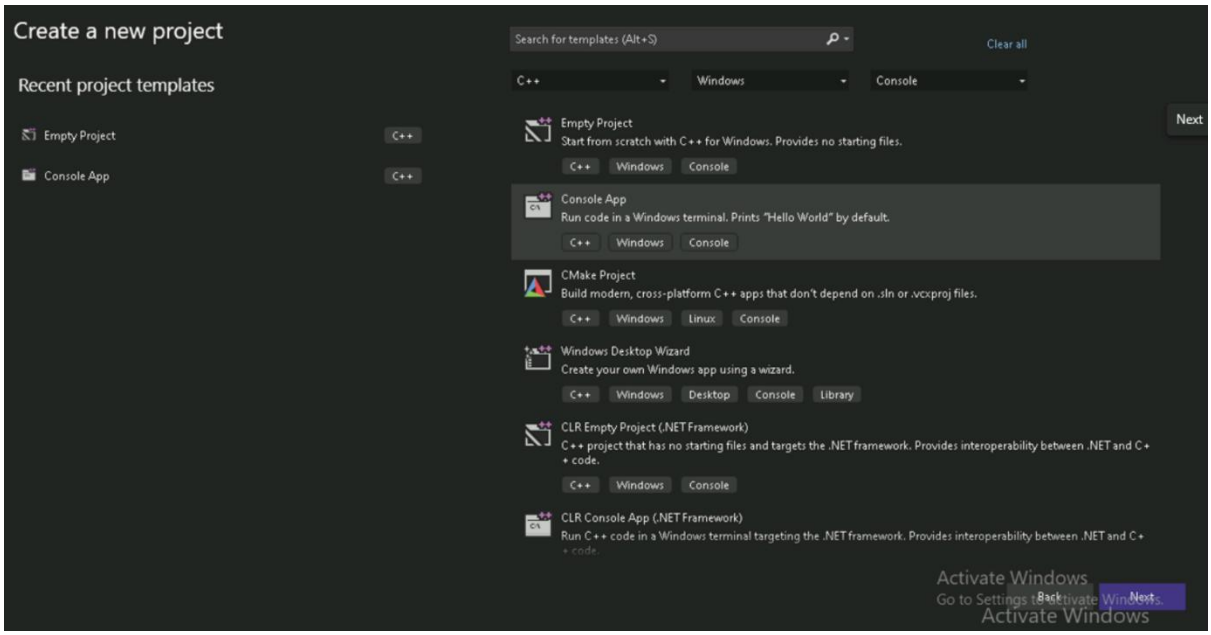
->After selecting the required features and components ->select Install to update and modify.



Step 4:Open Visual Studio->Select Create a new project->Console App/Empty project->select next

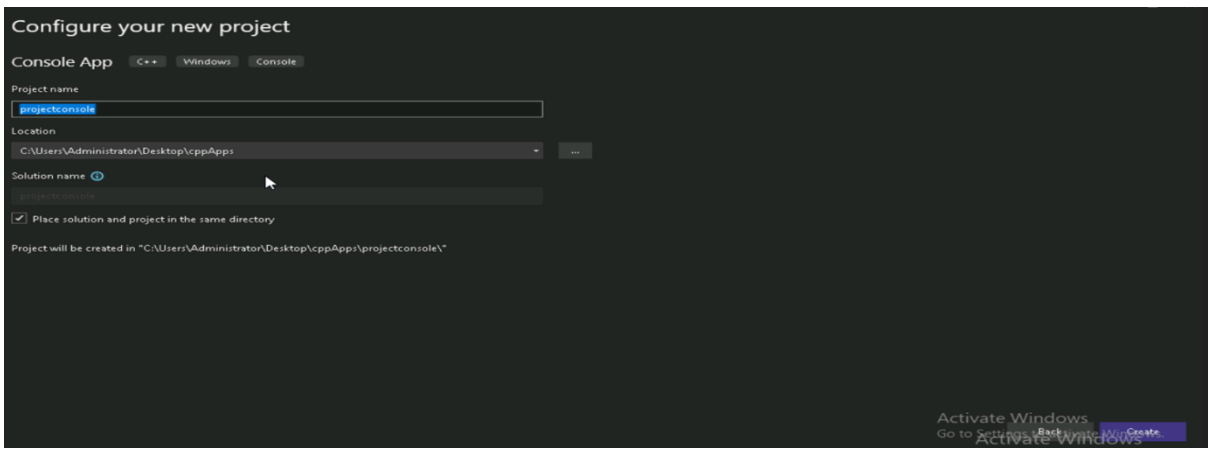


Console App-It consist of a basic structure of a CPP file

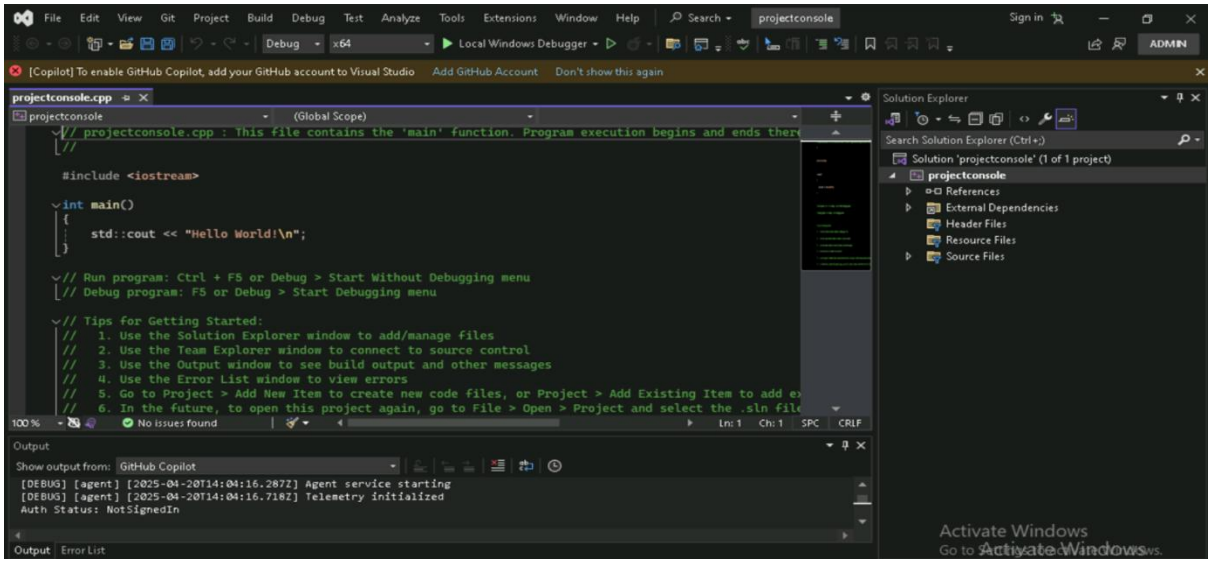


Step 5:Configure your New Project

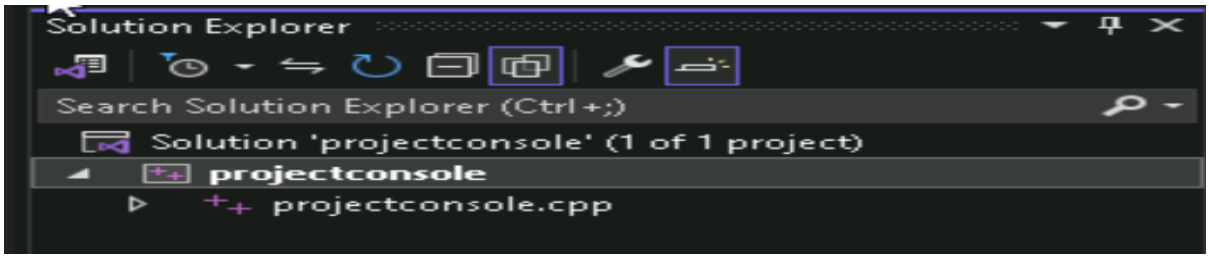
Add Project name->Add location ->Add Solution Name->select create.

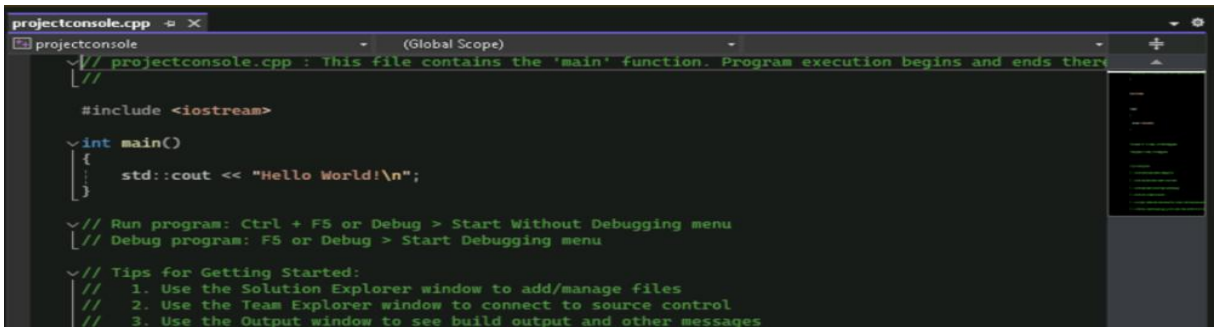


The below image Shows the basic structure of cpp file and the look of visual Studio.



Step 6:Add a CPP file to the project and type the code.

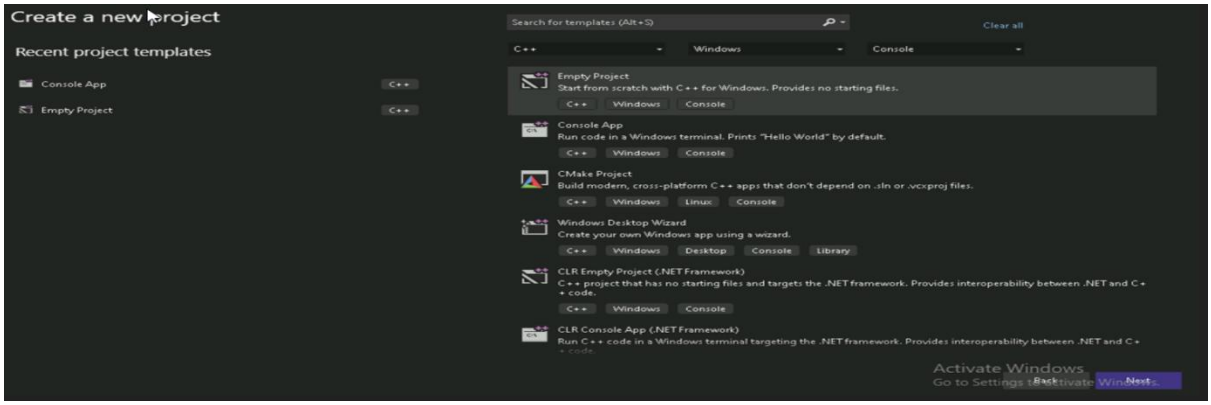




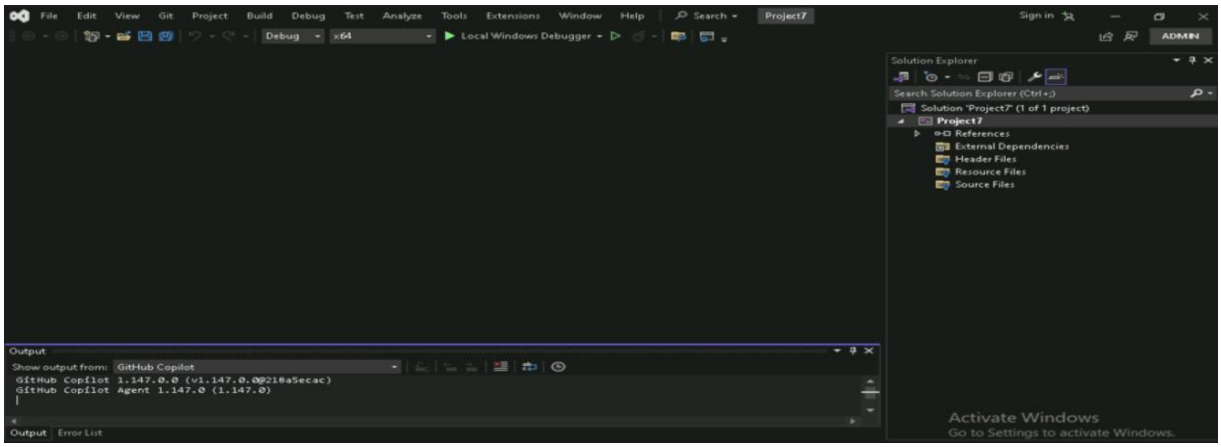
Step 7:Compile the code using Build Solution/using command prompt using the below commands

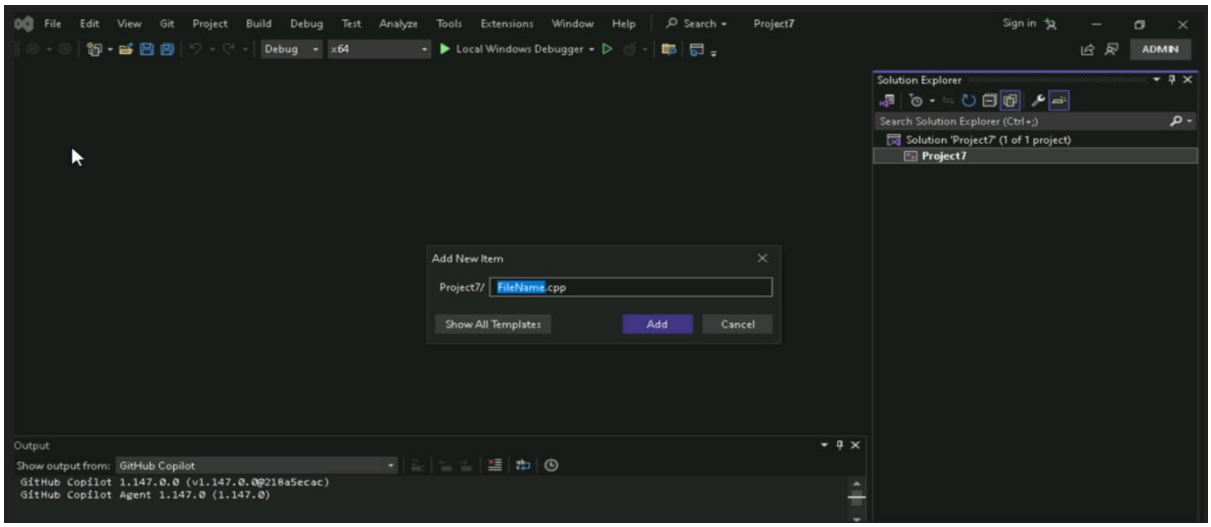
- g++ filename.cpp(To create an executable file(.exe)
- a.exe(To compile the executable file)

To Create an Empty project



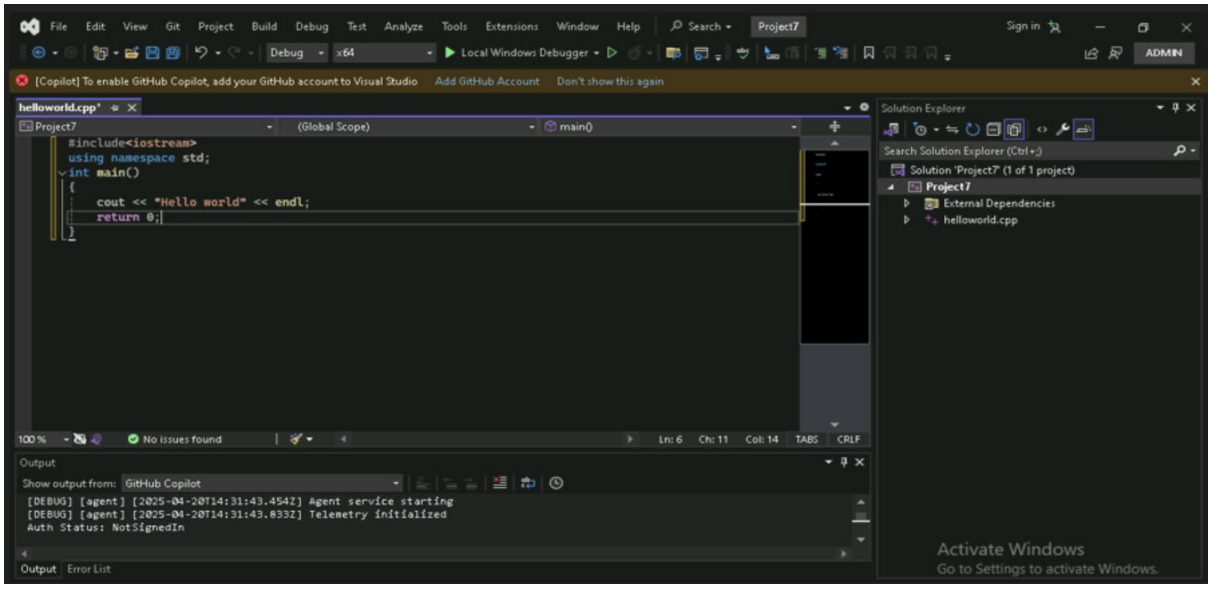
Add Project name->Add location ->Add Solution Name->select create-> Add a CPP file





How to create a Project

Step 1:Create a Empty project



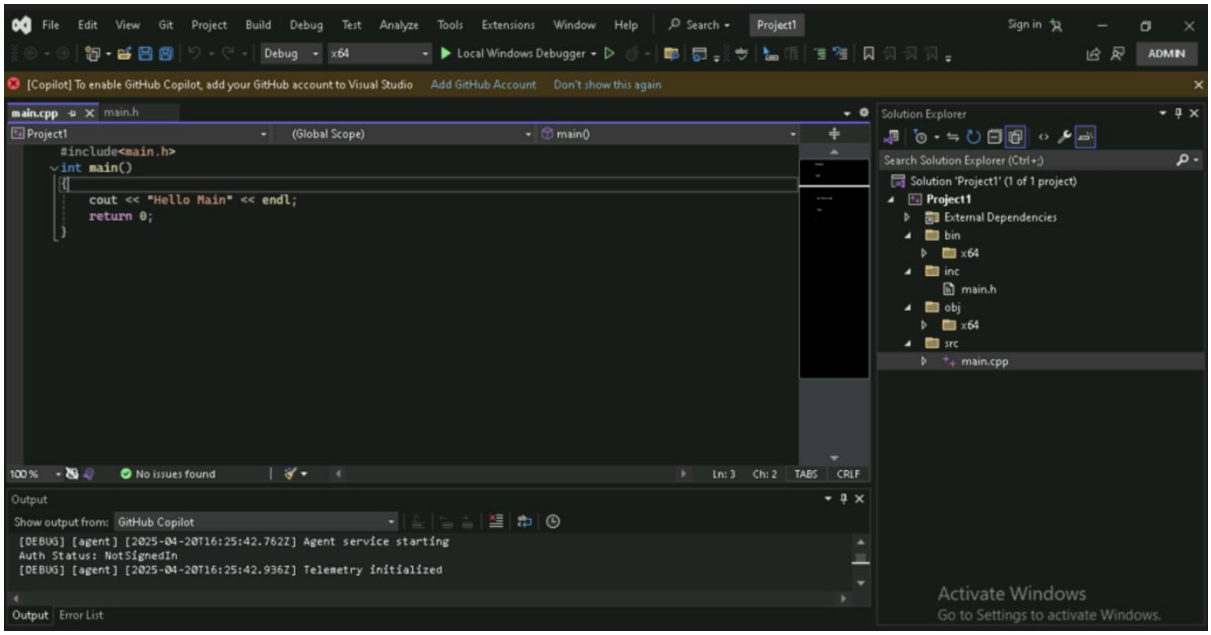
Step 2:Add different Folders inside the project as src, lib, bin and inc

src – Store the Source file (.cpp)

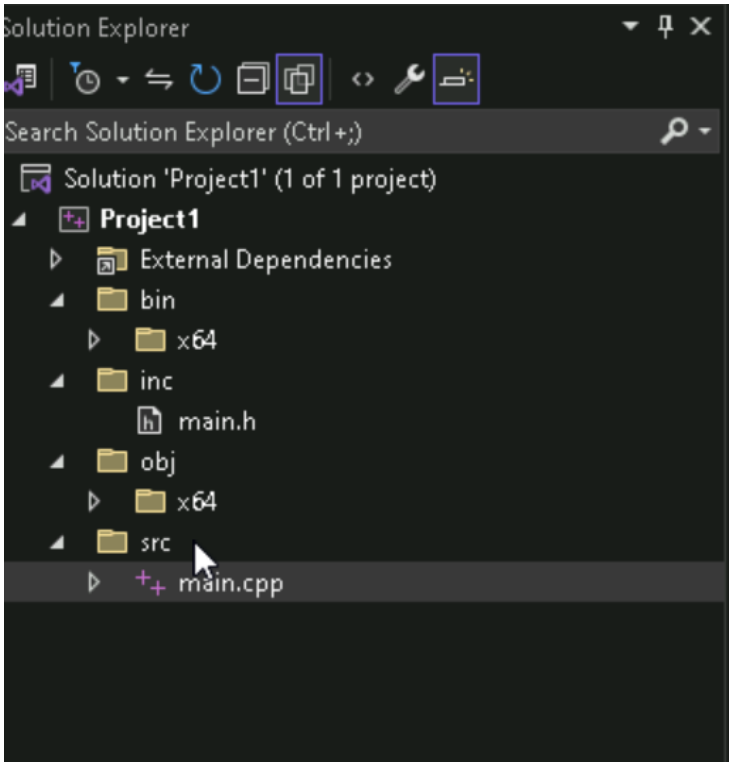
bin – Used to store the binary file or executable files(a.exe)

inc – Used to store the header files(.h)

lib – It is used to store object files, library files(.o , .s , .lib)

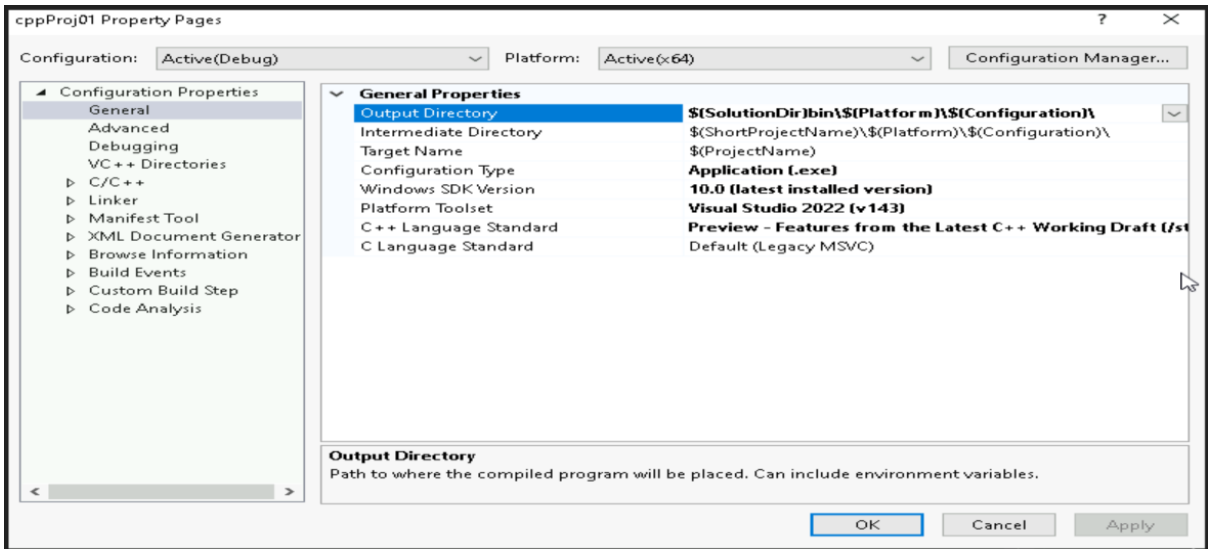


The below image illustrates how it is been created



Step 3: To modify the build output directory for a specific C++ project:

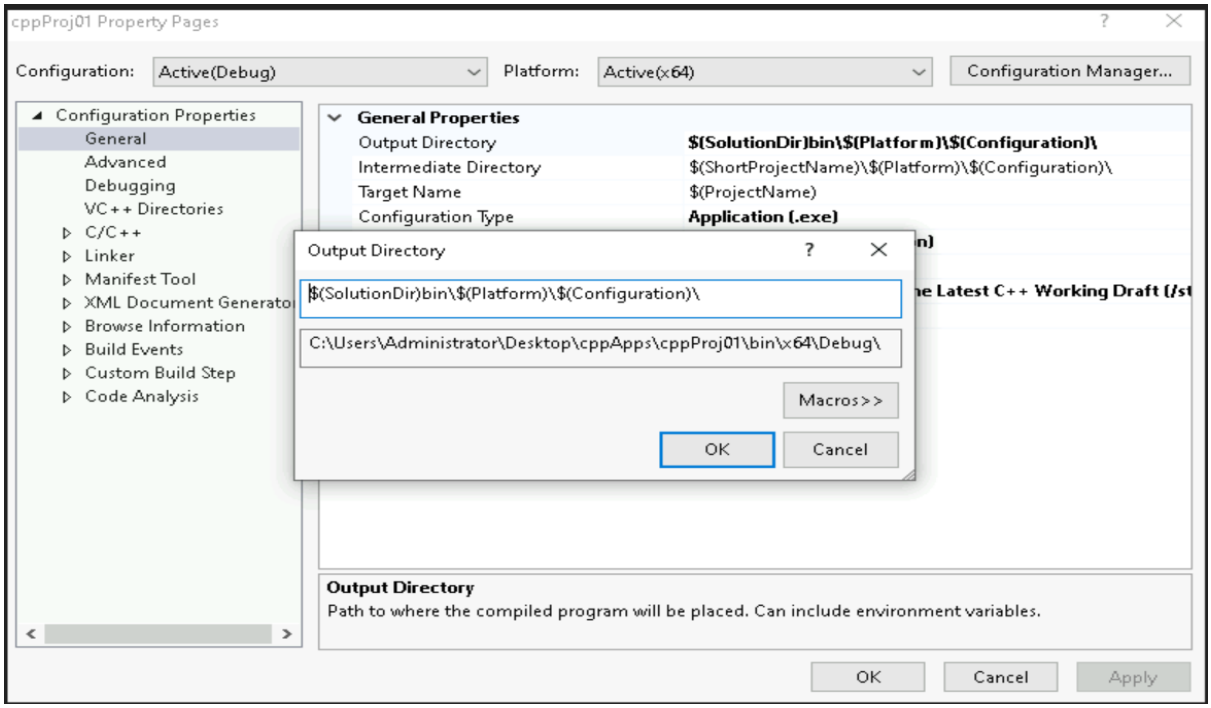
- Begin by cleaning the project to remove any existing output files (navigate to Build > Clean Solution).
- In Solution Explorer, right-click on the project node and select Properties.
- Go to the General tab.
- Using the configuration drop-down menu at the top, select the configuration for which you wish to change the output file location (such as Debug, Release, or All Configurations).



- To Add bin to Output Directory use the drop-down menu and select macros and edit it
- Next select ok ->Select Apply to make changes

Why Bin directory is used?

It is used to store the output files generated during the build process.

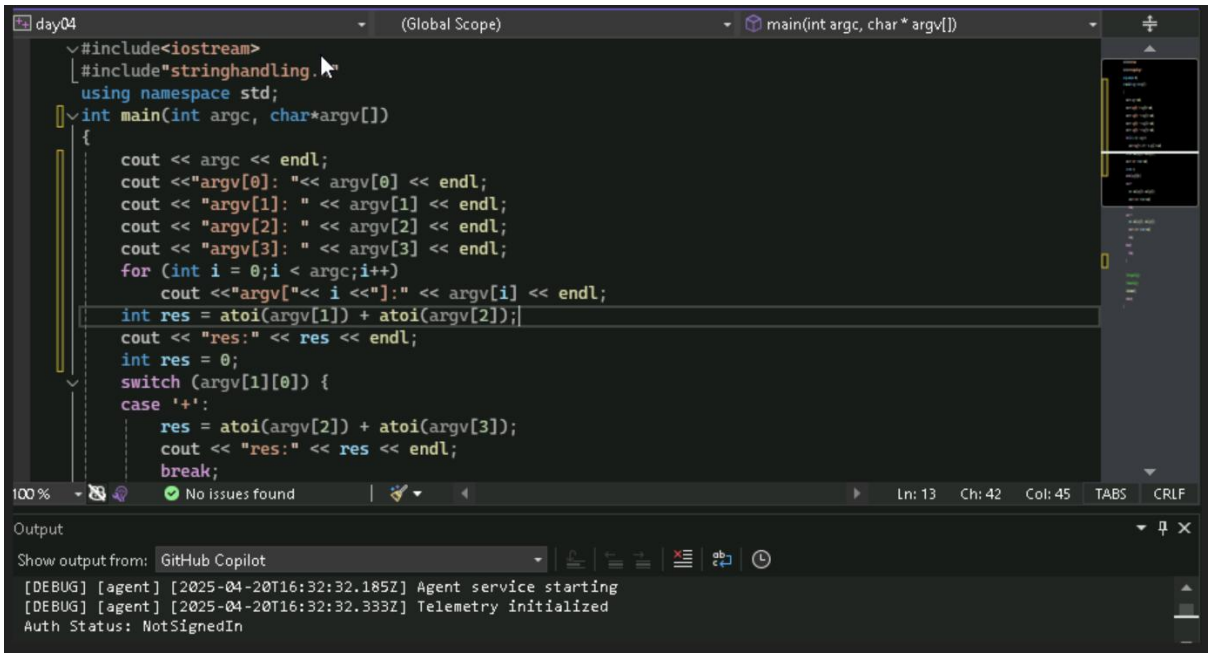


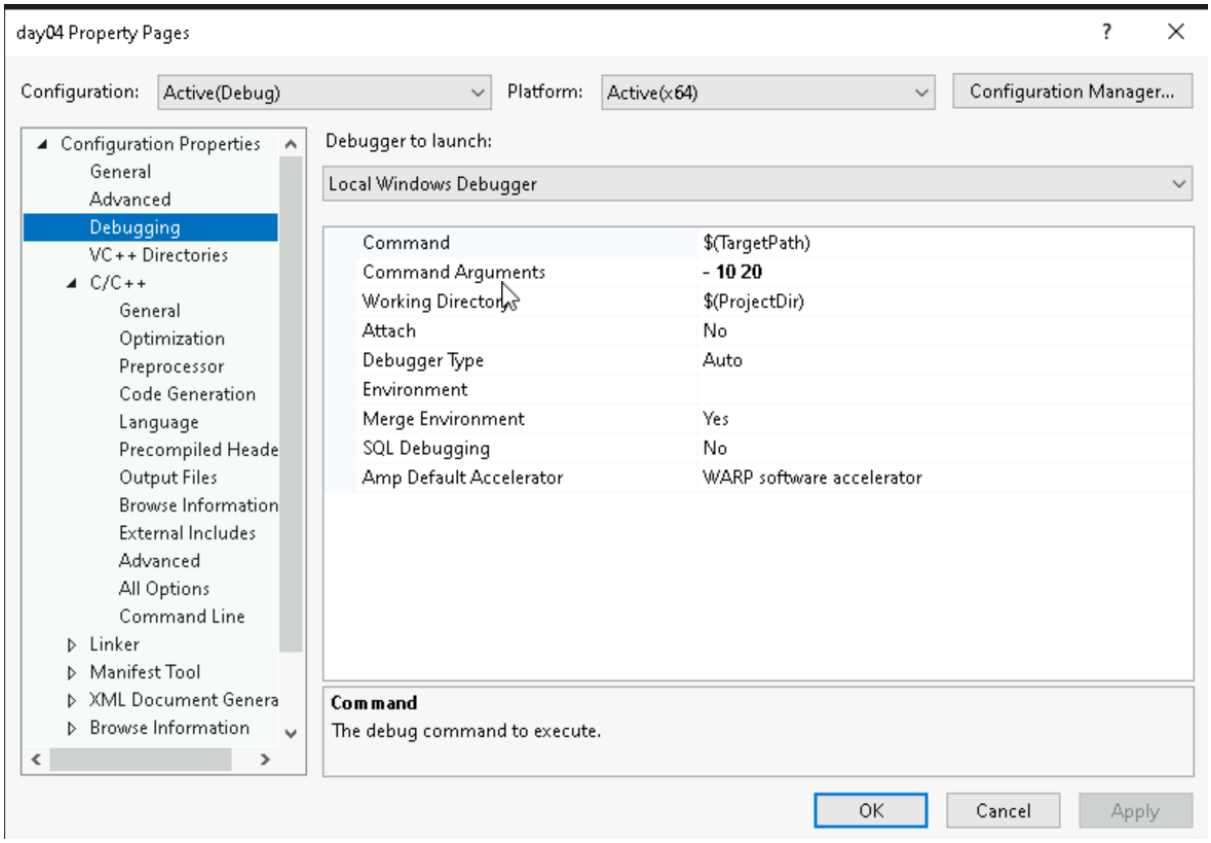
TO HANDLE OR TO IGNORE WARNINGS

To ignore the warnings there are two ways:

1. Copy the Warning/error number then,
   In Solution Explorer, right-click on the project node -> select Properties->C/C++->Advanced->Disable specific warnings->paste the copied error id ->Select Apply to make changes.

2. Copy the Warning(macro) then,
   In Solution Explorer, right-click on the project node -> select Properties->C/C++->PreProcessor->PreProcessor Definitions->Paste the copied Warning(macro).


day04
(Global Scope)
main(int argc, char * argv[])
#include<iostream>
#include"stringhandling.
using namespace std;
int main(int argc, char*argv[])
{
cout << argc << endl;
cout <<"argv[0]: "<< argv[0] << endl;
cout << "argv[1]: " << argv[1] << endl;
cout << "argv[2]: " << argv[2] << endl;
cout << "argv[3]: " << argv[3] << endl;
for (int i = 0;i < argc;i++)
cout <<"argv["<< i <<"]:" << argv[i] << endl;
int res = atoi(argv[1]) + atoi(argv[2]);
cout << "res:" << res << endl;
int res = 0;
switch (argv[1][0]) {
case '+':
res = atoi(argv[2]) + atoi(argv[3]);
cout << "res:" << res << endl;
break;
100 %
No issues found
Ln: 13
Ch: 42
Col: 45
TABS
CRLF
Output
Show output from:
GitHub Copilot
[DEBUG] [agent] [2025-04-20T16:32:32.185Z] Agent service starting
[DEBUG] [agent] [2025-04-20T16:32:32.333Z] Telemetry initialized
Auth Status: NotSignedIn


day04 Property Pages
Configuration:
Active(Debug)
Platform:
Active(x64)
Configuration Manager...
Configuration Properties
General
Advanced
Debugging
VC++ Directories
C/C++
General
Optimization
Preprocessor
Code Generation
Language
Precompiled Heade
Output Files
Browse Information
External Includes
Advanced
All Options
Command Line
Linker
Manifest Tool
XML Document Genera
Browse Information
Debugger to launch:
Local Windows Debugger
Command
$(TargetPath)
Command Arguments
- 10 20
Working Directory
$(ProjectDir)
Attach
No
Debugger Type
Auto
Environment
Merge Environment
Yes
SQL Debugging
No
Amp Default Accelerator
WARP software accelerator
Command
The debug command to execute.
OK
Cancel
Apply