Assignment Problem:

Problem: Implement a phong shading model to a 3D cube by implementing the GetViewMatrix() function in camera.h and completing the projection matrix in main.cpp
The phong.vs and phong.frag will be the 2 files that will calculate the lighting and the actual color and shading of the cube model

Methods:

The first step is to set up the camera.h function in order for the camera to actually work properly. In the GetViewMatrix() function, the function returns glm::lookAt(Position, Position + Front, Up);
Lookat controls where the camera will be pointed towards, and it returns the matrix that will give the specific coordinates that will point to where you want it to look by moving the mouse. The parameters basically take the current position, the X-axis in front of your position, and the Y-Axis of that position.

The actual color shading occurs in phong.vs where you're mainly calculating the normal vector, gl_position (position of the lighting) and normal vector (see. https://mathworld.wolfram.com/NormalVector.html#:~:text=The%20normal%20vector%2C%20often%20simply,pointing%20normal%20are%20usually%20distinguished.)
Phong.frag references the variables from phong.vs and actually combines 3 types of lighting in order to create the phong shader effect: **Ambient lighting**, which makes the cube a solid color, **Diffuse**, which gives the cube a shader effect (makes one side bright and the others darker), and **Specular**, which adds the spotlight reflection on the surface. The three combined create the phong effect, and can be reproduced by following the steps here: https://learnopengl.com/Lighting/Basic-Lighting

Once the calculations are done, you can start the program and the result will be like the one in results.jpg. The cube will be rendered at a specific coordinate and the lighting will be added to the cube through the functions of camera.h and the calculations of the lighting shader.