CHAPTER – 1

INTRODUCTION

1.1 Introduction

In this era of computing, computer graphics has become one of the most powerful and interesting fact of computing. It all started with display of data on hardcopy and CRT screen.
Now computer graphics is about creation, retrieval, manipulation of models and images. Graphics today is used in many different areas. Graphics provides one of the most natural means of communicating witha computer, since our highly developed 2D and 3D pattern recognition abilities allow us to perceive and process pictorial data rapidly and efficiently. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television. It has the added advantage that, with the computer, we can make pictures not only of concrete real world objects but also of abstract, synthetic objects, such as mathematical surfaces and of data that have no inherent geometry, such as survey results.

The term computer graphics has been used in a broad sense to describe " almost everything on computers that is not text or sound ". Typically the term computer graphics refers to several different things :

- The representation and manipulation of image data by a computer.
- The various technologies used to create and manipulation of image.
- The sub field of computer science which studies methods for digitally synthesizing visual content.

Data communication refers to the transmission of digital data between two or more computers and A computer network or data network is a telecommunications network that allows computers to exchange data. The physical connection between networked computing devices is established using either cable media or wireless media. The effectiveness of a data communications system depends on four fundamental characteristics: delivery, accuracy, timeliness, and jitter. A network is two or more devices connected through links. A link is a communications pathway that transfers data from one device to another. This process involves a communication system which is made up of hardware and software. The hardware part involves the sender and receiver devices and the intermediate devices through which the data passes. The software part involves certain rules which specify what is to be communicated, how it is to be communicated and when. It is also called as a Protocol.

Data Communication Over Internet is developed with the use of procedural language like C++ and OPENGL glut libraries has also been used as an API.

1.2 Aims and Objectives

The main objective of Data Communication Over Internet is to enable seamless exchange of data between any two points in the world. This exchange of data takes place over a computer network.

The Fundamental characteristics of Data communication systems are:

- **Delivery**: The data should be delivered to the correct destination and correct user.
- **Accuracy:** The communication system should deliver the data accurately, without introducing any errors. The data may get corrupted during transmission affecting the accuracy of the delivered data.
- **Timeliness:** Audio and Video data has to be delivered in a timely manner without any delay; such a data delivery is called real time transmission of data.
- **Jitter:** It is the variation in the packet arrival time. Uneven Jitter may affect the timeliness of data being transmitted.

1.3 About OpenGL

OpenGL (Open Graphics Library) is a standard specification defining a cross language cross platform API for writing applications that produce 2D and 3D computer graphics. The interface consists of over 250 different function calls which can be used to draw complex 3D scenes from simple primitives. OpenGL was developed by Silicon Graphics Inc. (SGI) in 1992 and is widely used in CAD, virtual reality,scientific visualization, information visualization and flight simulation. It is also used in video games, where it competes with direct 3D on Microsoft Windows Platforms. OpenGL is managed by the non profit technology consortium, the Khronos group.INC.

OpenGL serves two main purposes :

- To hide the complexities of interfacing with different 3D accelerators, by presenting programmerwith a single, uniform API
- To hide the differing capabilities of hardware platforms , by requiring that all implementationssupport the full OpenGL feature set.

1.3.1 OpenGL Operation

OpenGL has become a widely accepted standard for developing graphics application. OpenGL is easy tolearn, and it possesses most of the characteristics of other popular graphics system. It is top-down approach. OpenGL is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. The interface consists of over 250 different function calls which can be used to draw complex three-dimensional scenes from simple primitives.

OpenGL has become a widely accepted standard for developing graphics application. OpenGL is easy tolearn, and it possesses most of the characteristics of other popular graphics system. It is top-down approach. OpenGL is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. The interface consists of over 250 different function calls which can be used to draw complex three-dimensional scenes from simple primitives.

OpenGL was developed by Silicon Graphics Inc. (SGI) in 1992 and is widely used in CAD, virtual reality,scientific visualization, information visualization, and flight simulation. It is also used in video games, where it competes with Direct3D on Microsoft Windows platforms.

The interface between the application program and the graphics system can be specified through that set of function that resides in graphics library. The specification is called the APPLICATION PROGRAM INTERFACE (API). The application program sees only the API and is thus shielded from the details boththe hardware and software implementation of graphics library. The software driver is responsible for interpreting the output of an API and converting these data to a form that is understood by the particularhardware.

Most of our applications will be designed to access openGL directly through functions in three libraries.Function in the main GL library have name that begin with the letter gl and stored in the library. The second is the openGL utility Library (GLU). This library uses only GL function but contains codes for creating common object and viewing. Rather than using a different library for each system we used available library called openGL utility toolkit (GLUT). It is used as #include<glut.h>. A graphics editor is a computer program that allows users to compose and edit pictures interactively onthe computer screen and save them in one of many popular “bitmap” or “raster” a format such as TIFF, JPEG, PNG and GIF.

My project named " DATA COMMUNICATION OVER INTERNET " uses OpenGL software interface and develops 2D images. Thisproject uses the techniques like Translation, motion, display list, transformation techniques, etc.

The graphics in OpenGL provides a wide variety of built-in function. The computer graphics remains oneof the most exciting and rapidly growing computer fields. It has become a common element in user interface, data visualization, TV commercials, motion picture and many other applications. The current trend of computer graphics is to incorporate more physics principles into 3D graphics algorithm to bettersimulate the complex interactions between objects and lighting environment.

1.3.2 Primitives and Commands

OpenGL draws primitives—points, line segments, or polygons—subject to several selectable modes. Youcan control modes independently of each other; that is, setting one mode doesn't affect whether other modes are set (although many modes may interact to determine what eventually ends up in the frame buffer). Primitives are specified, modes are set, and other OpenGL operations are described by issuing commands in the form of function calls. Primitives are defined by a group of one or more vertices. A vertex defines a point, an endpoint of line, or a corner of a polygon where two edges meet. Data (consisting of vertex coordinates, colors, normal, texture coordinates, and edge flags) is associated with a vertex, and each vertex and its associated data are processed independently.

Commands are always processed in the order in which they are received, although there may be an indeterminate delay before a command takes effect. This means that each primitive is drawn completely Before any subsequent command takes place. It also means effect that state-querying commands return data that's consistent with complete execution of all previously issued OpenGL commands.

1.4 About C ++

C++ is one of the world's most popular programming languages. It can be found in today's operating systems, Graphical User Interfaces, and embedded systems. C++ is an object-oriented programming language which gives a clear structure to programs and allows code to be reused, lowering development costs. C++ is portable and can be used to develop applications that can be adapted to multiple platforms. C++ is fun and easy to learn!

As C++ is close to C# and Java, it makes it easy for programmers to switch to C++ or vice versa.

CHAPTER 2

SYSTEM REQUIREMENTS

2.1 Hardware Requirements:

The selection of hardware is very important in the existence and proper working of any software. In the selection of hardware, the size and the capacity requirements are also important. The Web Based Manufacturing System can be efficiently run on Pentium system with at least 128 MB RAM and Hard disk drive having 20 GB. Floppy disk drive of 1.44 MB and 14 inch color monitor suits the information system operation.(A Printer is required for hard copy output).

Processor	: Intel-Core i5,i7
Processor Speed	: 2.0 GHz
RAM	: 1GB or more
Hard Disk	: 40GB to 80GB

2.2 Software Requirements:

Data Communication Over internet is graphically designed in such a way that the user can get the better visual experience. The Sender and the receiver are the two users of the project where the data packet and the acknowledgement would transmit from one another. This requirements has made the programming and the communication easier.

Operating System	: Windows (7,10,11)
IDLE	: CODE BLOCKS
Programming Language	: C++
Graphics Library	: glut.h
API	: OpenGL 2.0

CHAPTER 3

SYSTEM DESIGN

3.1 OpenGL Graphics Architecture

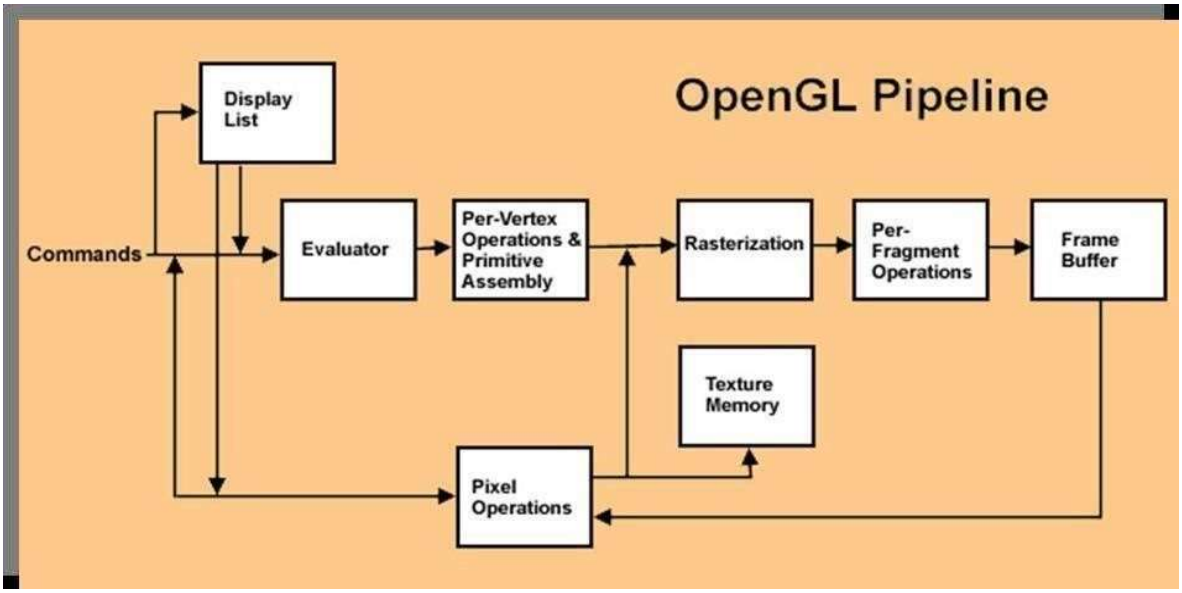


Fig 3.1 OpenGL Graphics Architecture

In proposed system, the OpenGL is a graphic software system designed as a streamlined, hardware-independent interface to be implemented on many different hardware platforms. To achieve these qualities, no commands for performing windowing tasks or obtaining user input are included in OpenGL; instead, you must work through whatever windowing system controls the particular hardware you're using.

OpenGL doesn't provide high-level commands for describing models of three-dimensional objects. Such commands might allow you to specify relatively complicated shapes such as automobiles, parts of the body, airplanes, or molecules. With OpenGL, you must build up your desired model from a small set of geometric primitives - points, lines, and polygons .

CHAPTER 4

IMPLEMENTATION

This program is implemented using various OpenGL functions which are shown below.

4.1 Various functions used in this program.

- **glutInit () :** interaction between the windowing system and OPENGL is initiated
- **glutInitDisplayMode () :** used when double buffering is required and depth information is required
- **glutCreateWindow () :** this opens the OPENGL window and displays the title at top of the window
- **glutInitWindowSize () :** specifies the size of the window
- **glutInitWindowPosition () :** specifies the position of the window in screen co-ordinates
- **glutKeyboardFunc () :** handles normal ASCII symbols
- **glutSpecialFunc () :** handles special keyboard keys
- **glutReshapeFunc () :** sets up the call-back function for reshaping the window
- **glutIdleFunc () :** this handles the processing of the background
- **glutDisplayFunc () :** this handles redrawing of the window
- **glutMainLoop () :** this starts the main loop, it never returns
- **glViewport () :** used to set up the viewport
- **glVertex3fv() :** used to set up the points or vertices in three dimensions
- **glColor3fv() :** used to render color to faces
- **glFlush () :** used to flush the pipeline
- **glutPostRedisplay () :** used to trigger an automatic redrawal of the object
- **glMatrixMode () :** used to set up the required mode of the matrix
- **glLoadIdentity () :** used to load or initialize to the identity matrix
- **glTranslatef () :** used to translate or move the rotation centre from one point to another in three dimensions
- **glRotatef () :** used to rotate an object through a specified rotation angle

4.2 Interaction With the Program

- We can have an interaction with this project using popular input device like mouse and key board are used to interact with this program.
- Some keys of key board have specific function, we mentioned them below,
  - 's/S' - Start
  - 'n/N' – Network
  - 'a/A' – About Project
- With these above mentioned keys, we can interact with the program.
- Also we can interact with this program by using mouse (input device),

  By click on right and left button
  - On right click of mouse menu will occur.
  - Select the option by left click of mouse.

4.3 Source Code

4.3.1 Include Libraries

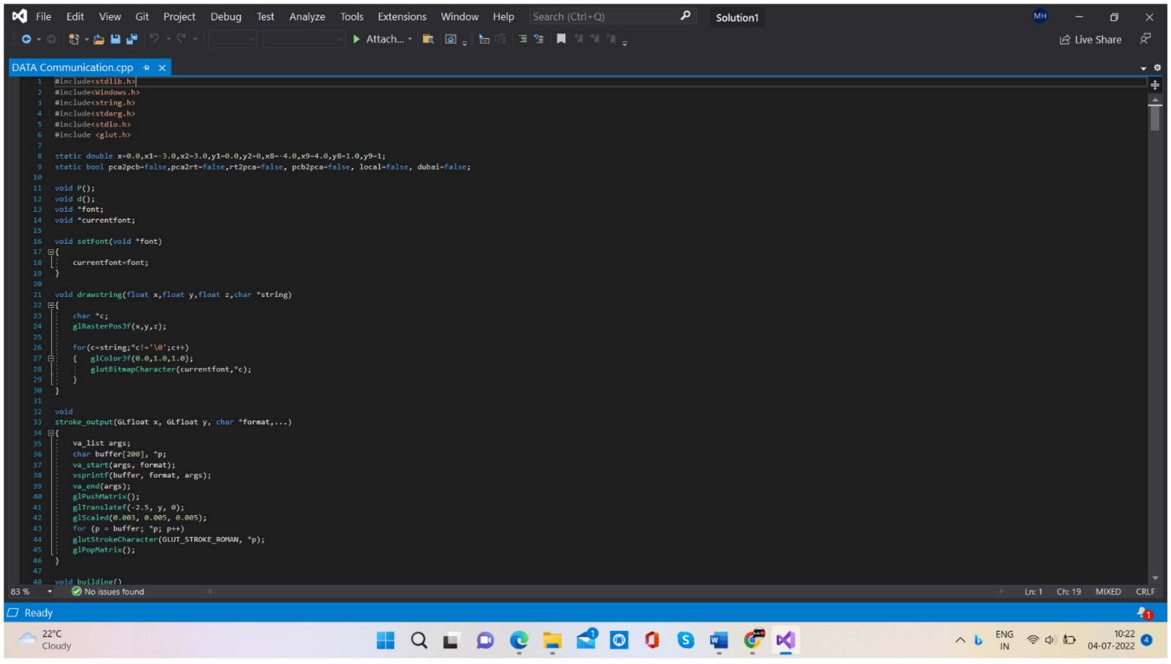
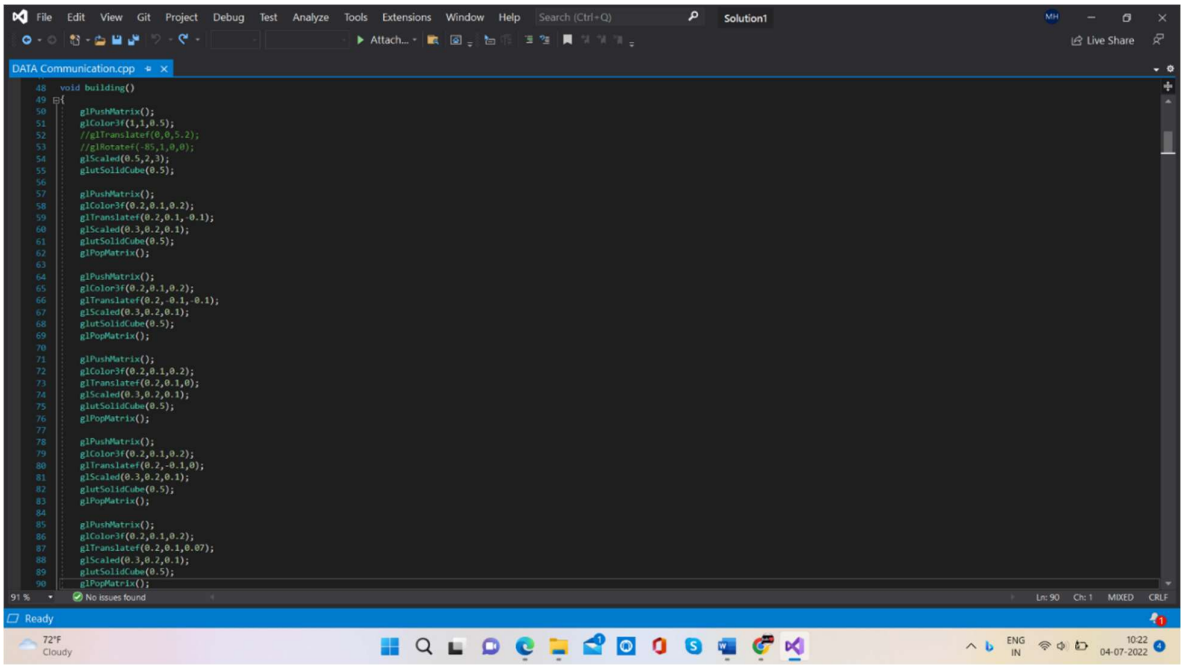


Fig 4.3.1: include libraries

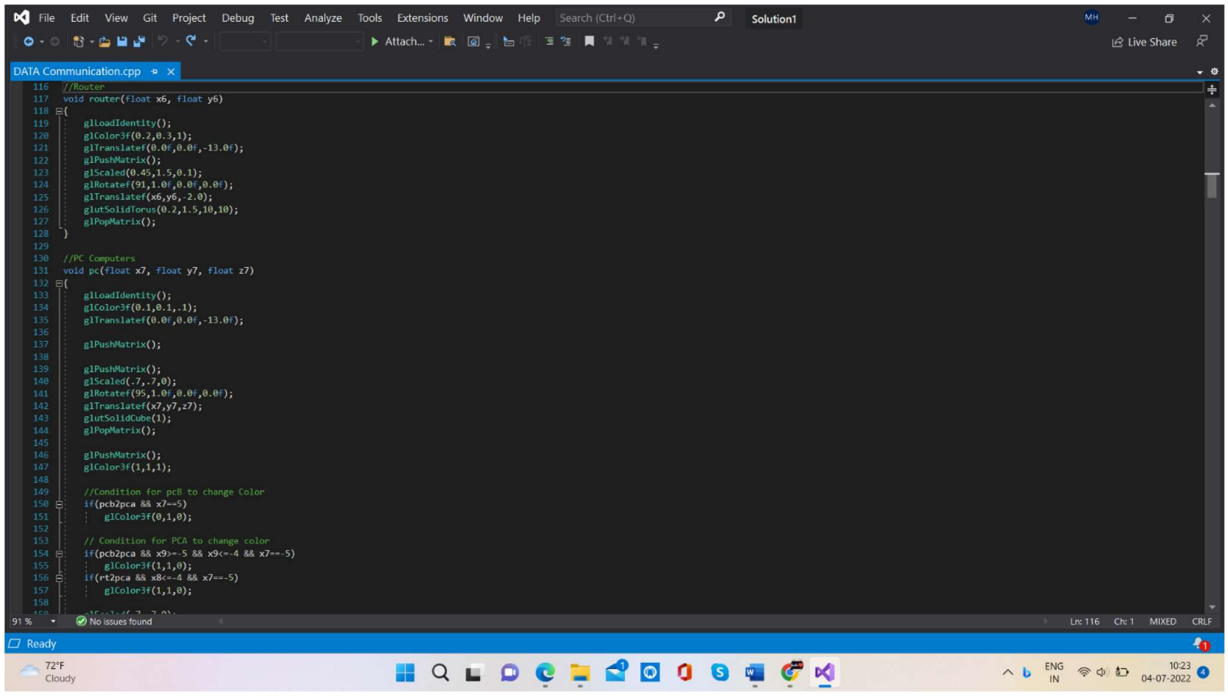
4.3.2 Void building method



```
void building()
{
    glPushMatrix();
    glColor3f(1,1,0.5);
    //glTranslatef(0,0,5.2);
    //glRotatef(-85,1,0,0);
    glScaled(0.5,2,3);
    glutSolidCube(0.5);

    glPushMatrix();
    glColor3f(0.2,0.1,0.2);
    glTranslatef(0.2,0.1,-0.1);
    glScaled(0.3,0.2,0.1);
    glutSolidCube(0.5);
    glPopMatrix();

    glPushMatrix();
    glColor3f(0.2,0.1,0.2);
    glTranslatef(0.2,-0.1,-0.1);
    glScaled(0.3,0.2,0.1);
    glutSolidCube(0.5);
    glPopMatrix();

    glPushMatrix();
    glColor3f(0.2,0.1,0.2);
    glTranslatef(0.2,0.1,0);
    glScaled(0.3,0.2,0.1);
    glutSolidCube(0.5);
    glPopMatrix();

    glPushMatrix();
    glColor3f(0.2,0.1,0.2);
    glTranslatef(0.2,-0.1,0);
    glScaled(0.3,0.2,0.1);
    glutSolidCube(0.5);
    glPopMatrix();

    glPushMatrix();
    glColor3f(0.2,0.1,0.2);
    glTranslatef(0.2,0.1,0.07);
    glScaled(0.3,0.2,0.1);
    glutSolidCube(0.5);
    glPopMatrix();
```

Fig 4.3.2 Void building method

4.3.3 Router and PC



```
//Router
void router(float x6, float y6)
{
    glLoadIdentity();
    glColor3f(0.2,0.3,1);
    glTranslatef(0.0f,0.0f,-13.0f);
    glPushMatrix();
    glScaled(0.45,1.5,0.1);
    glRotatef(91,1.0f,0.0f,0.0f);
    glTranslatef(x6,y6,-2.0);
    glutSolidTorus(0.2,1.5,10,10);
    glPopMatrix();
}

//PC Computers
void pc(float x7, float y7, float z7)
{
    glLoadIdentity();
    glColor3f(0.1,0.1,.1);
    glTranslatef(0.0f,0.0f,-13.0f);

    glPushMatrix();

    glPushMatrix();
    glScaled(.7,.7,0);
    glRotatef(95,1.0f,0.0f,0.0f);
    glTranslatef(x7,y7,z7);
    glutSolidCube(1);
    glPopMatrix();

    glPushMatrix();
    glColor3f(1,1,1);

    //Condition for pcB to change Color
    if(pcb2pca && x7==5)
        glColor3f(0,1,0);

    // Condition for PCA to change color
    if(pcb2pca && x9>=-5 && x9<=-4 && x7==-5)
        glColor3f(1,1,0);
    if(rt2pca && x8<=-4 && x7==-5)
        glColor3f(1,1,0);
```

Fig 4.3.3 Router and PC

4.3.4 Data Transfer and Acknowledgement

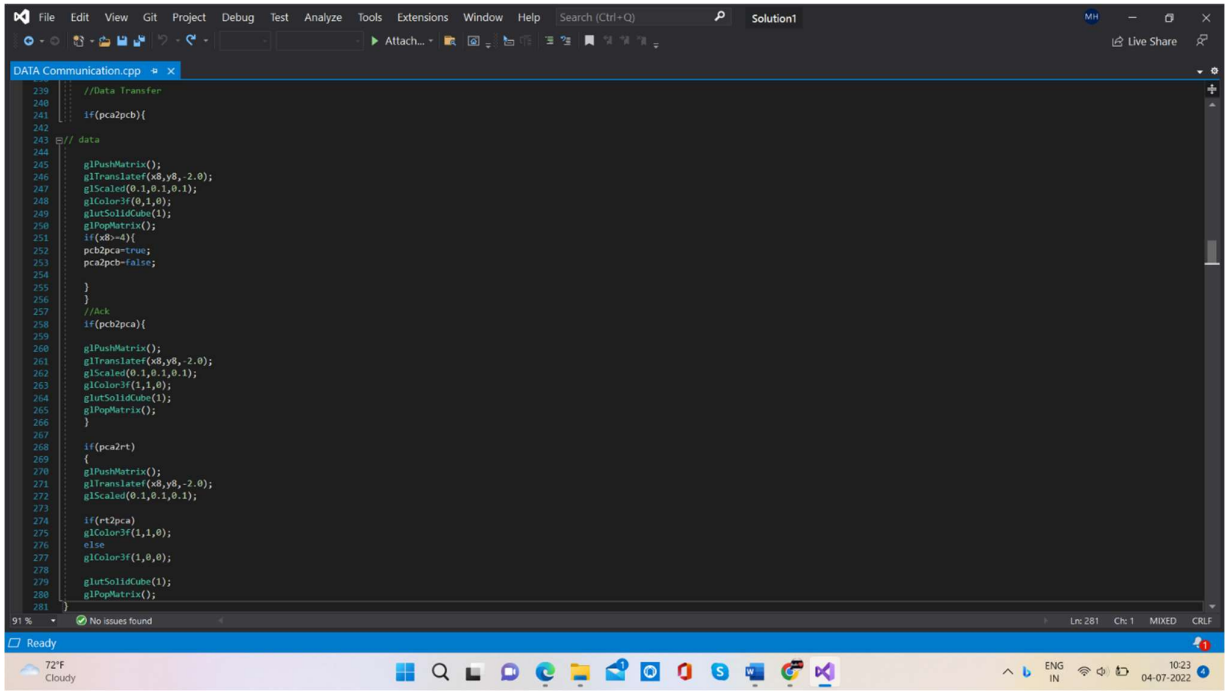


Fig 4.3.4 Data Transfer and Acknowledgement

4.3.5 Switch and Draw Network

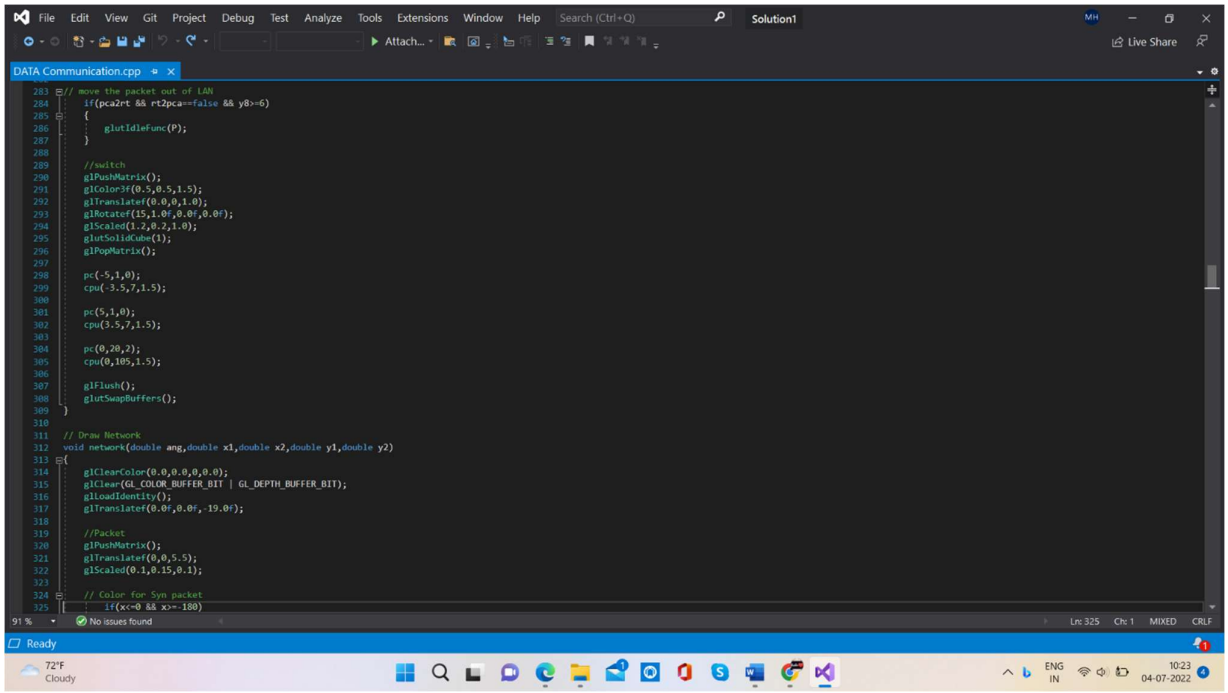


Fig 4.3.5 Switch and Draw Network

4.3.6 Earth Texture Code

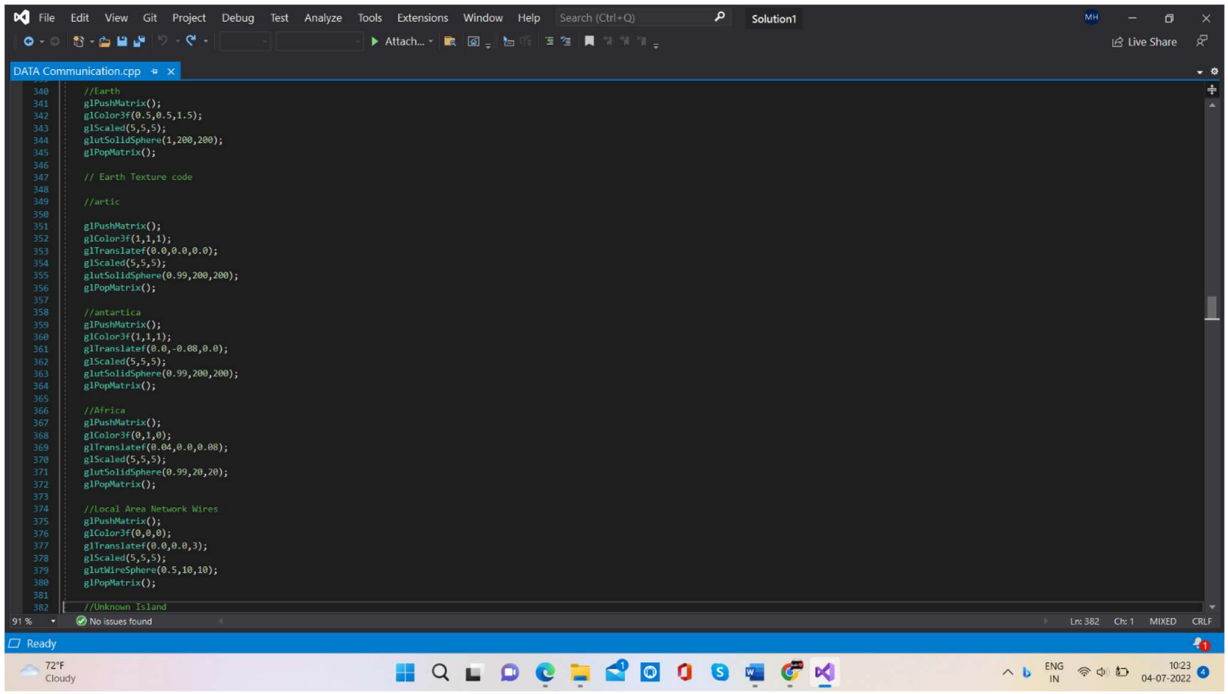


Fig 4.3.6 Earth Texture Code

4.3.7 LAN -A wires and Buildings

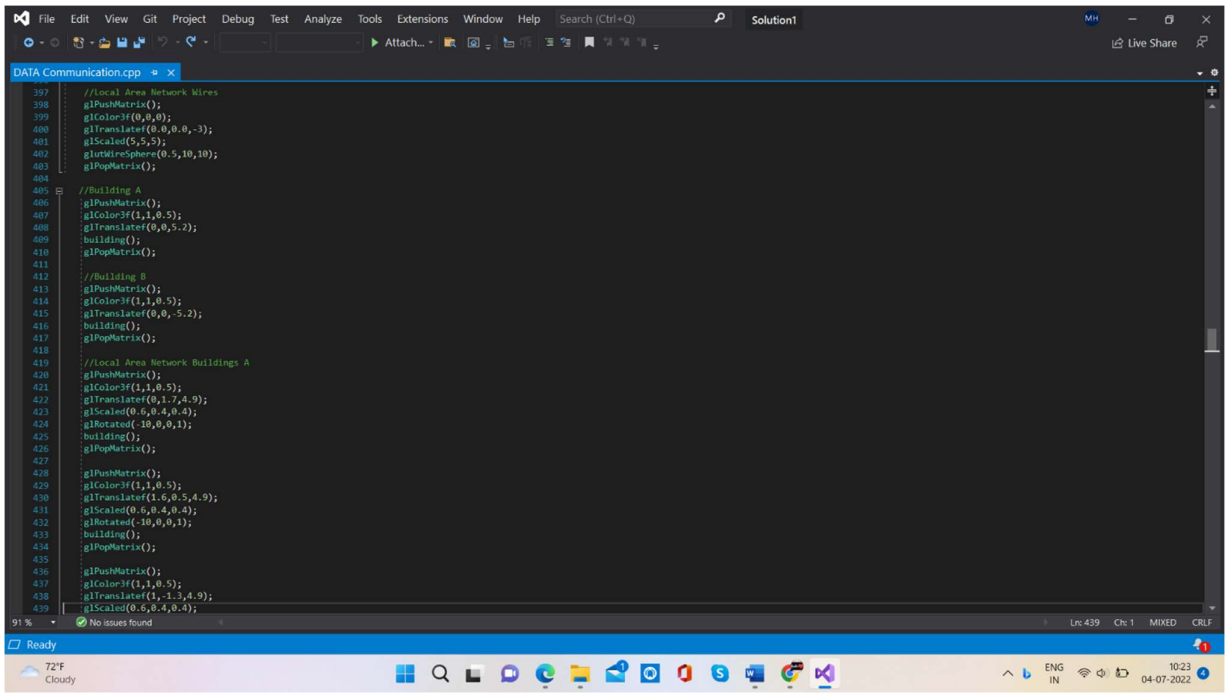


Fig 4.3.7 LAN wires and Buildings

4.3.8 LAN -B Buildings

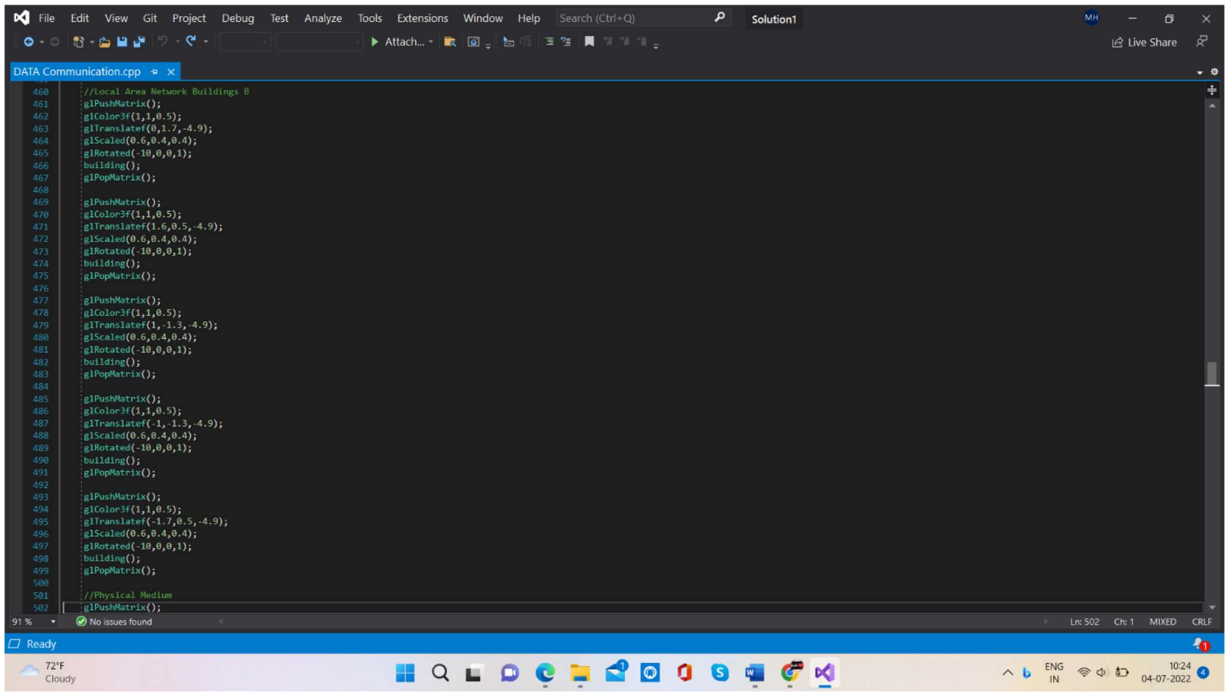


Fig 4.3.8 LAN-B Buildings

4.3.9 About section

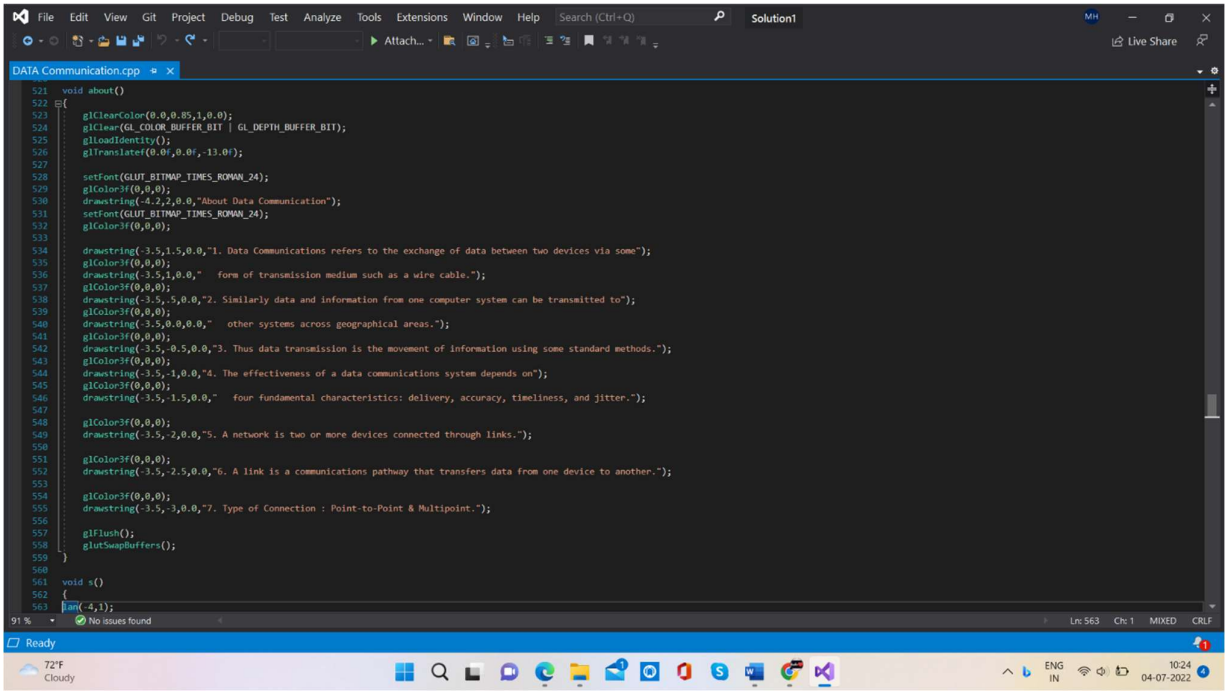
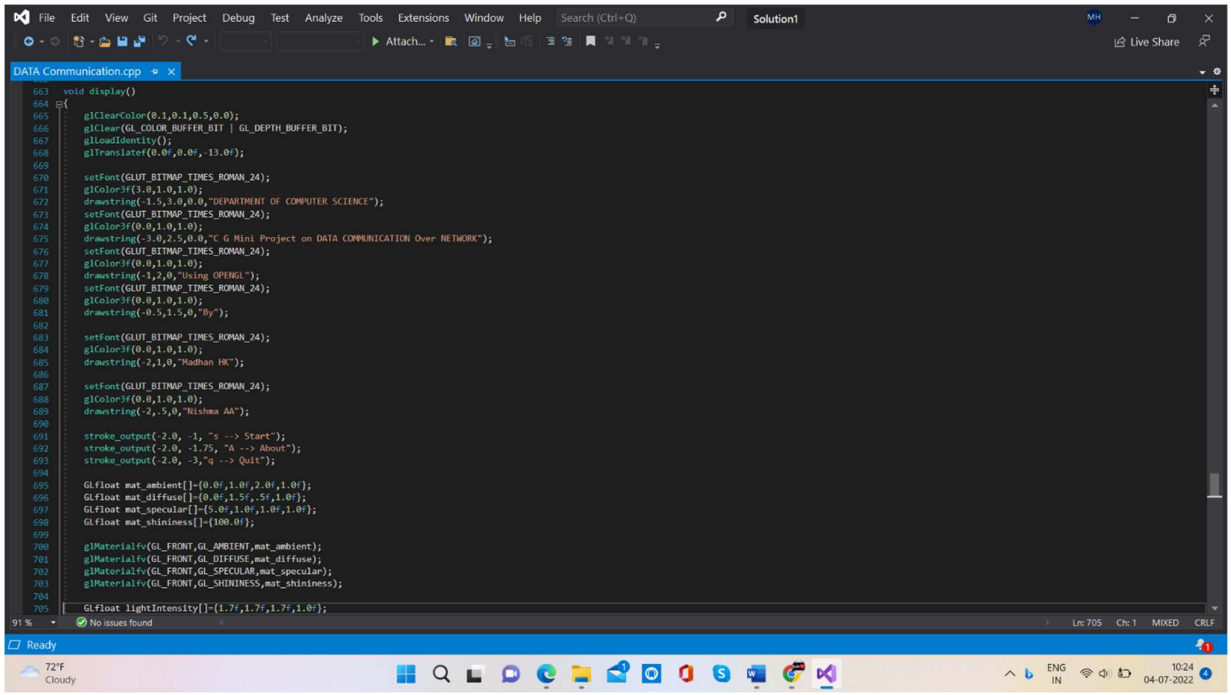


Fig 4.3.9 About section

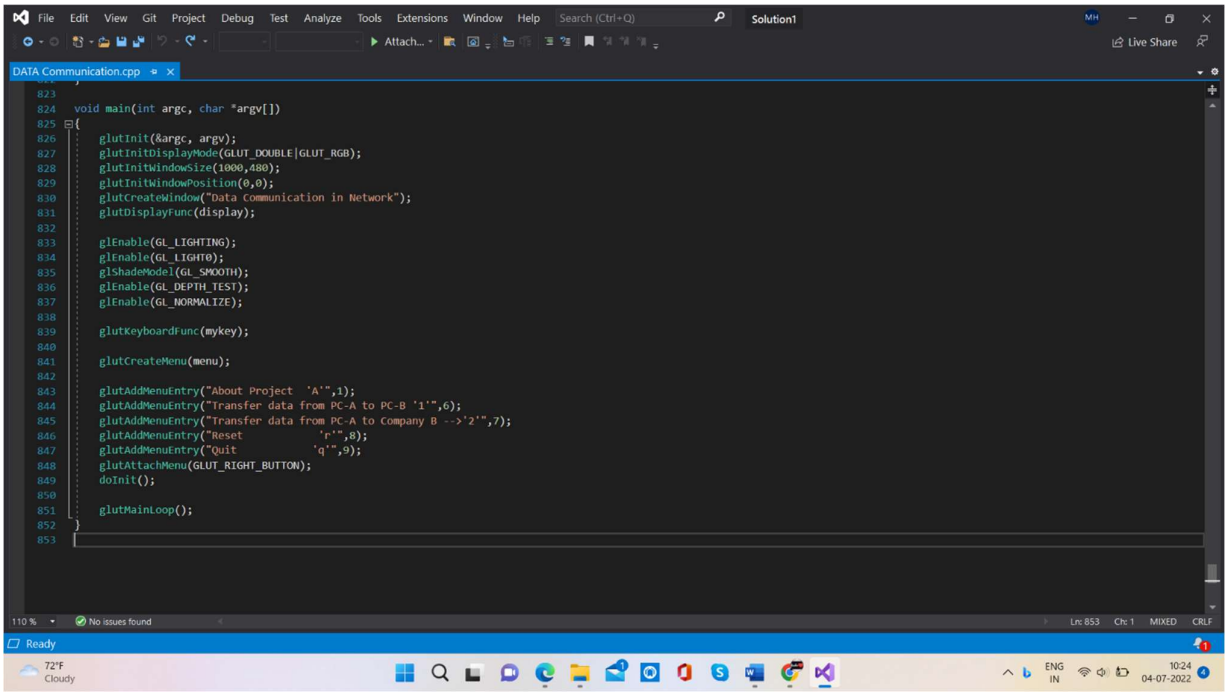
4.3.10 Display Section



```
void display()
{
    glClearColor(0.1,0.1,0.5,0.0);
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    glTranslatef(0.0f,0.0f,-13.0f);

    setFont(GLUT_BITMAP_TIMES_ROMAN_24);
    glColor3f(3.0,1.0,1.0);
    drawstring(-1.5,3.0,0.0,"DEPARTMENT OF COMPUTER SCIENCE");
    setFont(GLUT_BITMAP_TIMES_ROMAN_24);
    glColor3f(0.0,1.0,1.0);
    drawstring(-3.0,2.5,0.0,"C G Mini Project on DATA COMMUNICATION Over NETWORK");
    setFont(GLUT_BITMAP_TIMES_ROMAN_24);
    glColor3f(0.0,1.0,1.0);
    drawstring(-1,2,0,"Using OPENGL");
    setFont(GLUT_BITMAP_TIMES_ROMAN_24);
    glColor3f(0.0,1.0,1.0);
    drawstring(-0.5,1.5,0,"By");

    setFont(GLUT_BITMAP_TIMES_ROMAN_24);
    glColor3f(0.0,1.0,1.0);
    drawstring(-2,1,0,"Madhan HK");

    setFont(GLUT_BITMAP_TIMES_ROMAN_24);
    glColor3f(0.0,1.0,1.0);
    drawstring(-2,.5,0,"Nishma AA");

    stroke_output(-2.0, -1, "s --> Start");
    stroke_output(-2.0, -1.75, "A --> About");
    stroke_output(-2.0, -3,"q --> Quit");

    GLfloat mat_ambient[]={0.0f,1.0f,2.0f,1.0f};
    GLfloat mat_diffuse[]={0.0f,1.5f,.5f,1.0f};
    GLfloat mat_specular[]={5.0f,1.0f,1.0f,1.0f};
    GLfloat mat_shininess[]={100.0f};

    glMaterialfv(GL_FRONT,GL_AMBIENT,mat_ambient);
    glMaterialfv(GL_FRONT,GL_DIFFUSE,mat_diffuse);
    glMaterialfv(GL_FRONT,GL_SPECULAR,mat_specular);
    glMaterialfv(GL_FRONT,GL_SHININESS,mat_shininess);

    GLfloat lightIntensity[]={1.7f,1.7f,1.7f,1.0f};
```

Fig 4.3.10 Display Section

4.3.11 Main Function



```
void main(int argc, char *argv[])
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB);
    glutInitWindowSize(1000,480);
    glutInitWindowPosition(0,0);
    glutCreateWindow("Data Communication in Network");
    glutDisplayFunc(display);

    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glShadeModel(GL_SMOOTH);
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_NORMALIZE);

    glutKeyboardFunc(mykey);

    glutCreateMenu(menu);

    glutAddMenuEntry("About Project  'A'",1);
    glutAddMenuEntry("Transfer data from PC-A to PC-B '1'",6);
    glutAddMenuEntry("Transfer data from PC-A to Company B -->'2'",7);
    glutAddMenuEntry("Reset           'r'",8);
    glutAddMenuEntry("Quit           'q'",9);
    glutAttachMenu(GLUT_RIGHT_BUTTON);
    doInit();

    glutMainLoop();
}
```

Fig 4.3.11 Main Function

CHAPTER 5

RESULT

5.1 Front Page

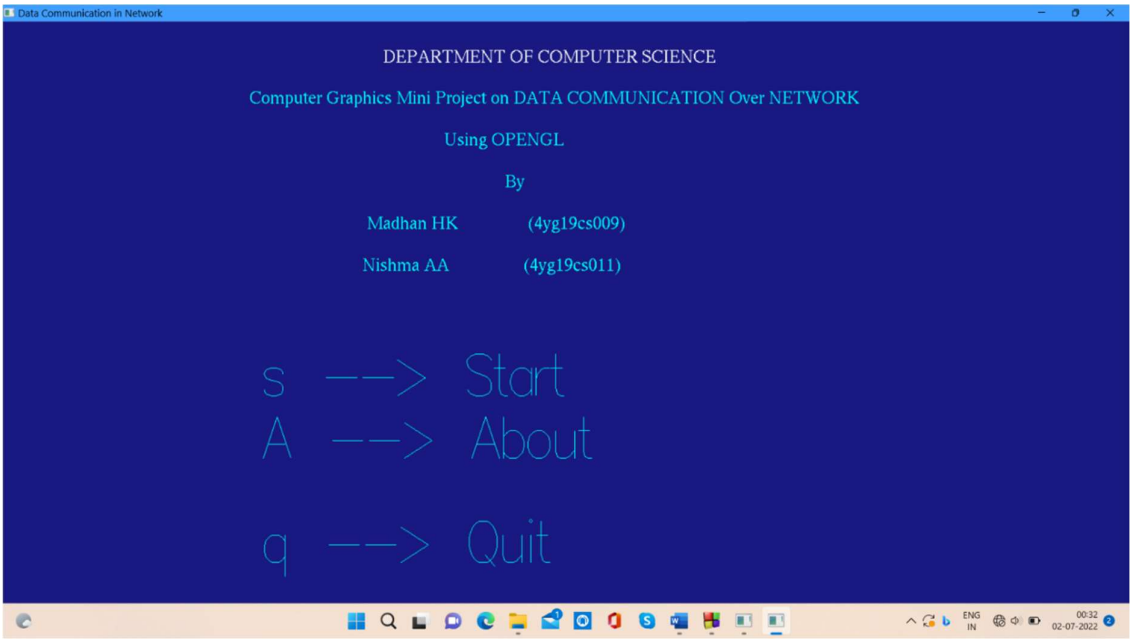


Fig 5.1 Front Page

5.2 About Page

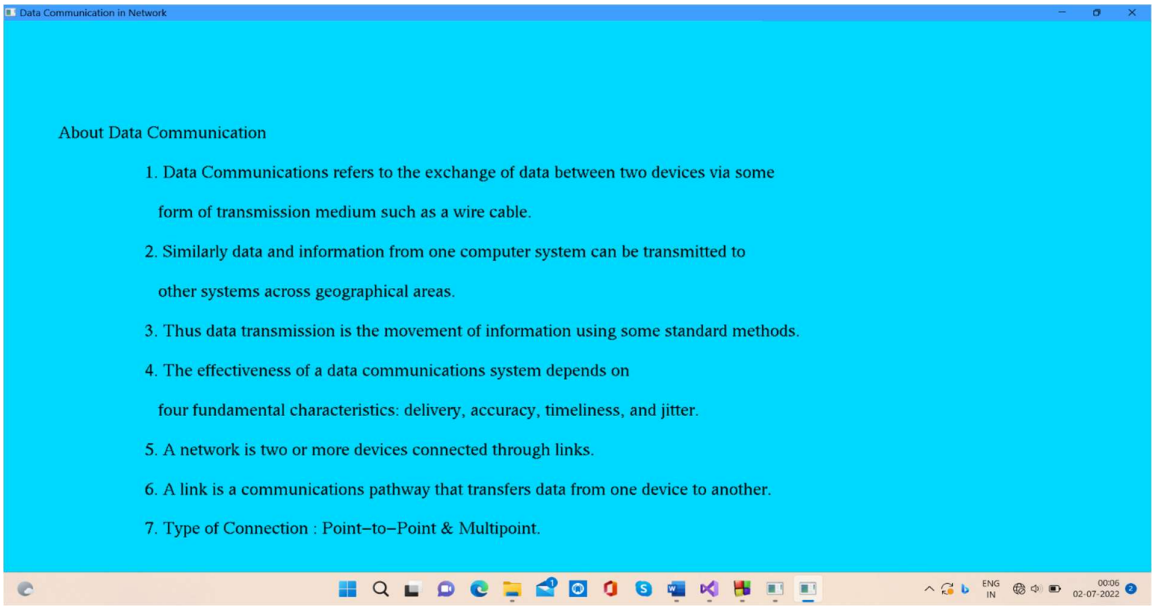


Fig 5.2 About Page

5.3 Local Area Network

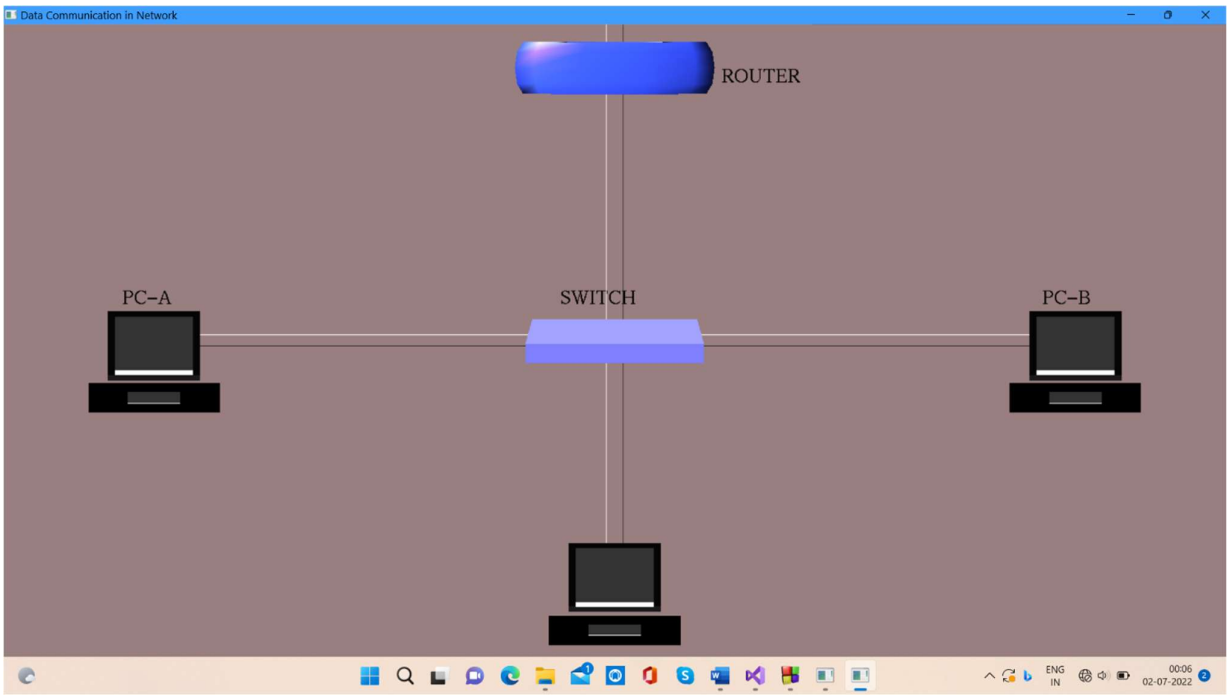


Fig 5.3 Local Area Network

5.4 Data Packet Transfer from PC-A to PC-B

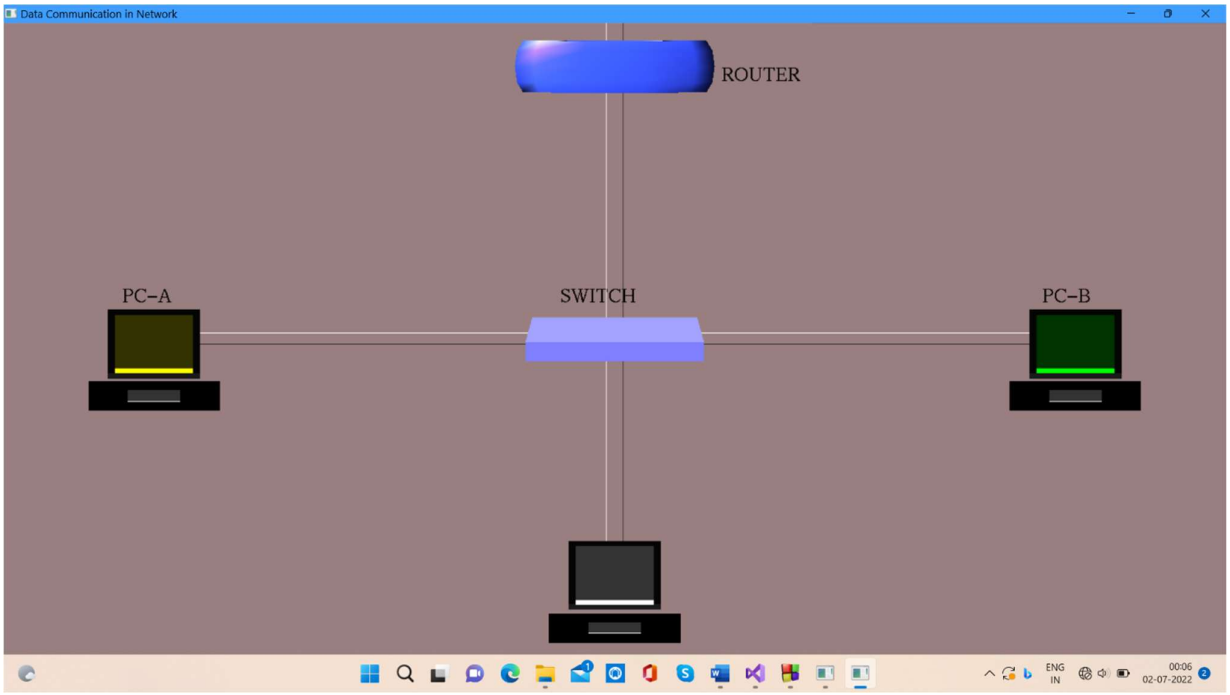


Fig 5.4 Data Packet Transfer from PC-A to PC-B

5.5 Receipt of Acknowledgement from PC-B to PC-A

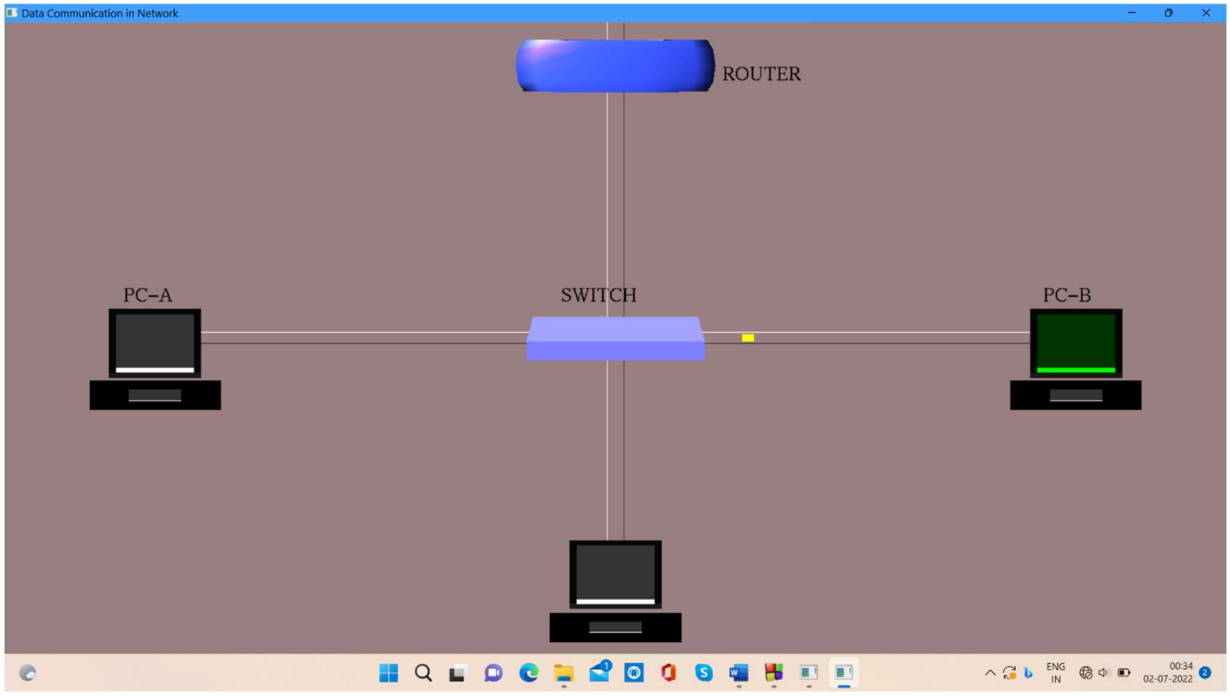


Fig 5.5 Receipt of Acknowledgement from PC-B to PC-A

5.6 On right Click menu bar

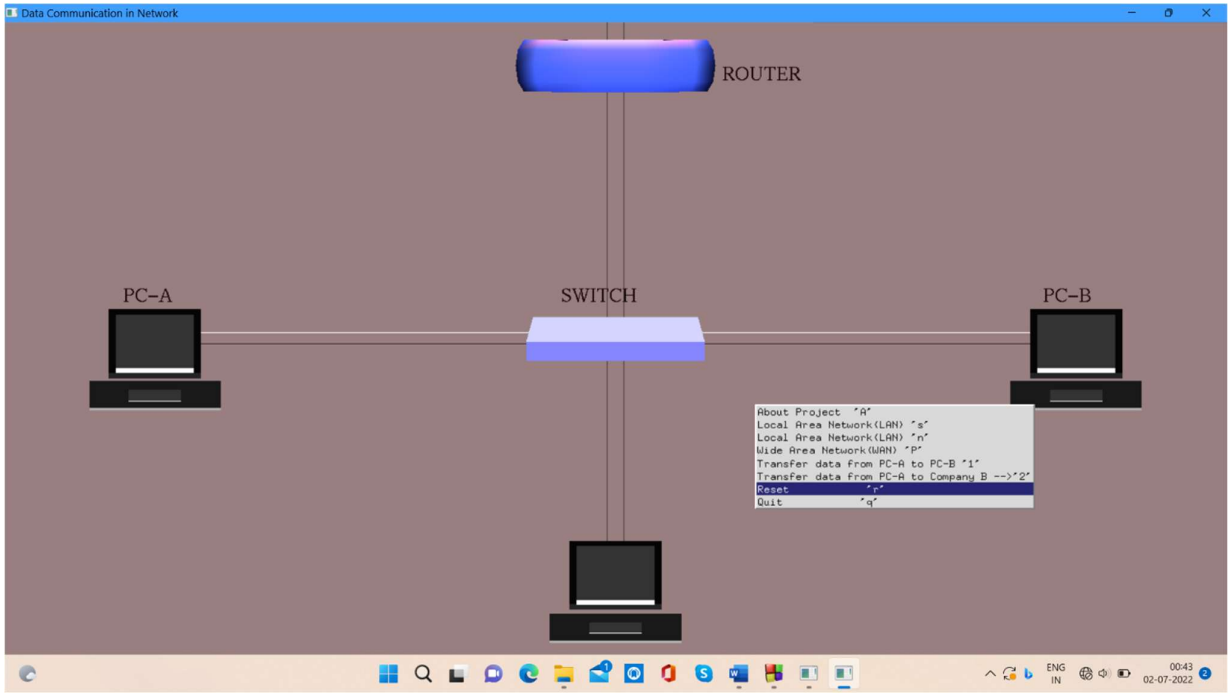


Fig 5.6 On right Click menu bar

5.7 Wide Area Network

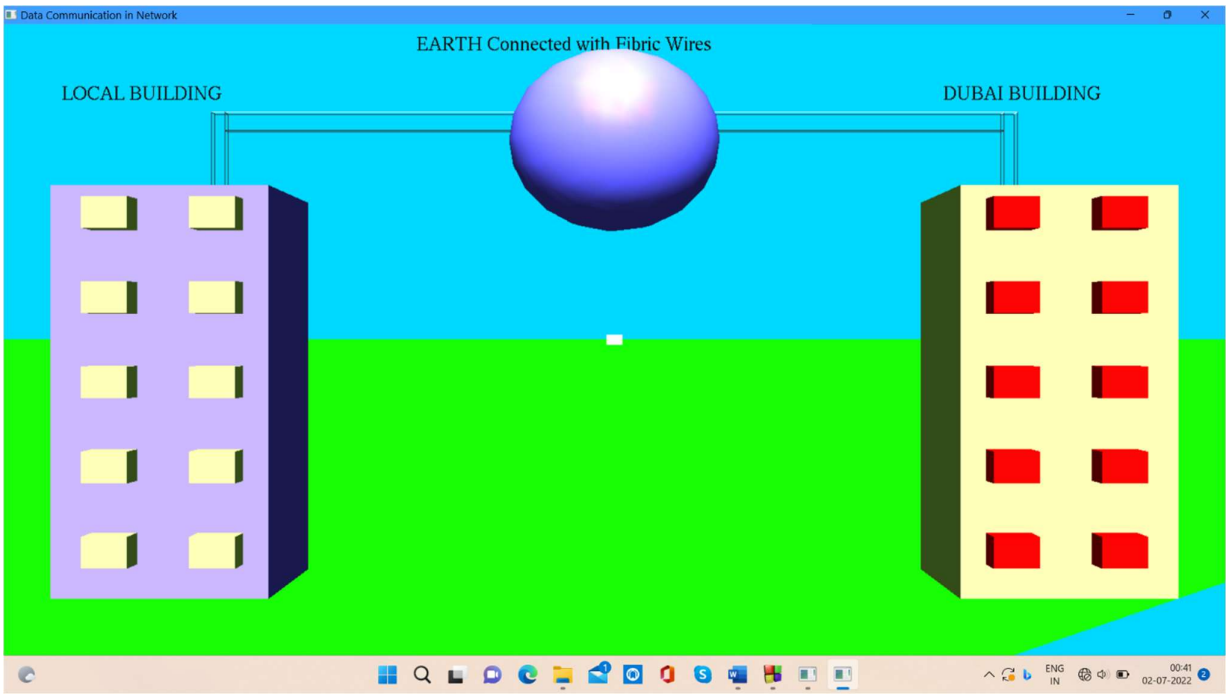


Fig 5.7 Wide Area Network

5.8 Transfer of Data from PC-A to Company-B

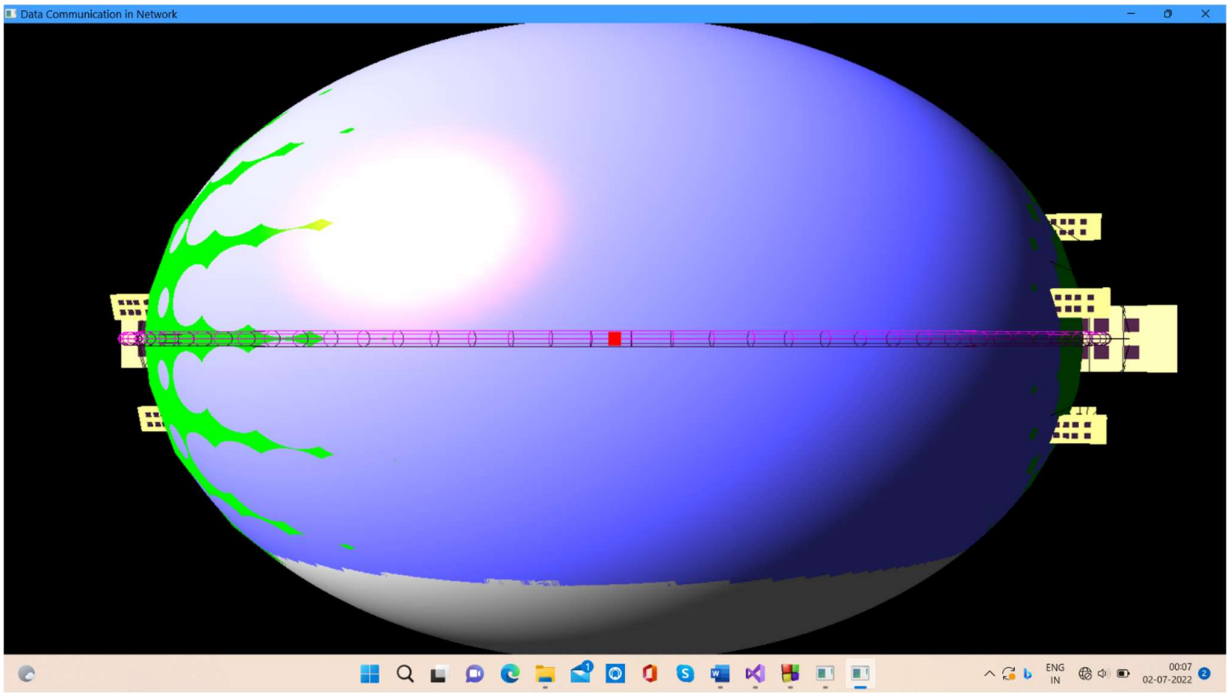


Fig 5.8 Transfer of Data from PC-A to Company-B

CONCLUSION

An attempt is made to develop an OpenGL package which means necessary requirements of the user successfully. Since it is user friendly, it enables the user to interact efficiently and easily. The development of the mini project which has been built by using point as a source, has given a good exposure towards OpenGL through which some of the techniques which help in development of animations, gaming was wellunderstood. It helps to understand the basic implementation of functions and basics of OpenGL. The code we have implemented for our project is working well to the best of our knowledge.

In this project the Data Transfer of packets act as per the user's command. This project is both informative and entertaining . This project provided an opportunity to learn the various concepts of the subject in detail and provided us a platform to express our creativity and imagination come true.

REFERENCES

1. https://www.w3schools.com
2. https://www.opengl.org
3. http://google.com