CHRIST UNIVERSITY

BANGALORE, INDIA

Declared as Deemed to be University under Section 3 of UGC Act 1956

Code Sharer

by

Akshay Sadarangani (1115905)
Diwani Tejal (1115919)
Varsha Kedia (1115959)

Under the guidance of
Ms.Vaidhehi V

A Specialization project report submitted in partial fulfillment of the requirements of V Semester Bachelor of Computer Applications of Christ University

September - 2013

CHRIST
UNIVERSITY
BANGALORE, INDIA

Declared as Deemed to be University under Section 3 of UGC Act 1956

CERTIFICATE

This is to certify that the report titled ***Code Sharer*** *is a bona fide record of work done by* ***Akshay Sadarangani (1115905), Diwani Tejal (1115919), Varsha Kedia (1115959)*** *of Christ University, Bangalore, in partial fulfillment of the requirements of V Semester BCA during the year 2013.*

Head of the Department **Project Guide**

Valued-by:

	Name	:Akshay Sadarangani
1.	Register Number	:
	Examination Centre	:Christ University
2.	Date of Exam	:25 September 2013

ACKNOWLEDGEMENTS

Code Sharer (CS) would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We would like to thank our parents, for their care, encouragement and being our moral support at all stages of the project. We are thankful to Ms. Vaidhehi V. for her guidance and constant supervision as well as for providing necessary information regarding the project & also for her support in completing the project.

Sincere thanks to all the faculty members of the Department of Computer Science, Christ University for their kind co-operation and encouragement which helped us in the completion of this project.

We would like to express our special gratitude and thanks to the industry people for giving us their time and attention. Our thanks and appreciation also goes to all our colleagues in developing the project and people who have willingly helped us out with their abilities.

Last but not the least, a special and sincere thanks to the Christ University Lab administrators and lab staff for allowing us to use the systems in complete freedom and supplying us with all the software needed and their technical support.

This project would not have been possible without every contribution, support and guidance of everyone mentioned.

ABSTRACT

Code Sharer is a software to aid programmers all around especially beginners. Currently the best help available to programmers is in the form of books, internet or forums. With Code Sharer programmers will get a new way to interactively communicate directly to their peers and share code snippets with marked compiler errors.

Code Sharer is a very easy to use text editor and is extremely easy to get started with. Hidden beneath the user-friendly interface, though, is a very powerful text editor that can make programmers smile in delight. It is developed to support all the features of the traditional compiler and more. Code Sharer also provides a chat interface for programmers to communicate and get help from other users. Programmers can also share screenshots of their codes with other programmers. Code Sharer is a high performance, sleek editor, with features that make a programmer's work even easier.

This software brings a new evolution to programming in a network and the possibilities of future developments are endless. Using Code Sharer, a team of programmers or a class can share their source code with each other and come up with possible solutions together. Code Sharer aims to be a handy tool for every programmer and help them communicate and share their programs with others without hassle.

LIST OF FIGURES

Figure Number	**Figure Name**	**Page Number**
3.1	Block Diagram	9
4.1	Two-tier Architecture	13
4.2	Detailed Flowchart	14
4.3	Editor Flowchart	15
4.4	Chat Flowchart	16
4.5	Snippet Share Flowchart	17
4.6	Screenshot Flowchart	18
4.7	Group Chat	21
4.8	Chat Interface	21
4.9	File Sharing	22
5.1	Chat Screen	50
5.2	Editor Screen	51
5.3	File Transfer Screen	52
5.4	Server Screen	52
6.1	Test Report: User Login Screen	55
6.2	Test Report: File Transfer	56

LIST OF TABLES

Table Number	**Table Name**	**Page Number**
1.1	Project Plan	2
3.1	Functional Requirements	10
3.2	Hardware Requirements	11
3.3	Software Requirements	12
6.1	Test Report	54

TABLE OF CONTENTS

ACKNOWLEDGEMENTS .... **iii**

ABSTRACT .... **iv**

LIST OF FIGURES .... **v**

LIST OF TABLES .... **vi**

1. INTRODUCTION 1
   - 1.1 OVERVIEW OF THE SYSTEM 1
   - 1.2 PROJECT PLAN 2
2. SYSTEM ANALYSIS 5
   - 2.1 EXISTING SYSTEM 5
   - 2.2 PROPOSED SYSTEM 5
   - 2.3 SOFTWARE TOOLS USED 7
3. SYSTEM REQUIREMENTS 8
   - 3.1 SYSTEM MODEL 8
   - 3.2 FUNCTIONAL REQUIREMENTS 10
   - 3.3 HARDWARE REQUIREMENTS 11
   - 3.4 SOFTWARE REQUIREMENTS 12
4. DESIGN SPECIFICATION 13
   - 4.1 ARCHITECTURAL DESIGN 13
   - 4.2 FLOW CHARTS 13
   - 4.3 NETWORK SPECIFICATION 19
   - 4.4 ALGORITHM SPECIFICATION 19
     - 4.4.1 NETWORK CONFIGURATION 19
     - 4.4.2 TOPOLOGY DETAILS 19
     - 4.4.3 PROTOCOL DETAILS 20
     - 4.4.4 ROUTER DETAILS 20
   - 4.5 USER INTERFACE DESIGN 21

5. IMPLEMENTATION..23

5.1 SOURCE CODE..23

5.2 SCREENSHOT..50

6. TESTING..53

6.1 TEST PLAN..53

6.2 TEST CASES..54

6.3 TEST REPORTS..55

7. CONCLUSION..57

7.1 ADVANTAGES..57

7.2 LIMITATIONS..58

7.3 FUTURE ENHANCEMENT..58

REFERENCES..59

1. INTRODUCTION

1.1 OVERVIEW OF THE SYSTEM

Code Sharer (CS) is a software which is meant to aid programmers especially beginners. This software helps programmers resolve bugs in the source code by sharing code snippets with their peers at the touch of a button. Code Sharer (CS) is an easy to use text-editor which provides an interactive Integrated Development Environment (IDE) to compile simple codes and detect errors with ease. With features like bug highlighting, syntax correction and a chat interface, Code Sharer (CS) is the idle solution to a programmer's woes.

With the help of Code Sharer (CS), a teacher or a team member can demonstrate and share their programs with a group of other programmers for learning and understanding all at runtime. Programmers can also share screenshots of their codes with others. This software brings a new evolution to programming in a network and the possibilities of future developments are endless. Code Sharer aims to be a handy tool for every programmer and help them communicate and share their programs with others without hassle.

Code Sharer (CS) makes use of two-tier client-server architecture. The client sends a request while the server responds. On the server-side, there is a database which is remotely accessed by authorized clients. This database contains details like the chat history and the user's friends.

1.2 PROJECT PLAN

Table 1.1 Project Plan

Student Name : Akshay Sadarangani	Register Number : 1115905
Title: Code Sharer	
Department : Computer Science	Guide Name : Vaidhehi V

Date	Phase	Start Time	End Time	Regular Hours	Overtime Hours	**Total Hours**
5/06/13	TITLE DISCUSSION	11:00am	12:00pm	1	0	1
6/06/13	INTRODUCTORY PHASE	9:00am	11:00am	2	0	2
7/06/13	INTRODUCTORY PHASE	02:00pm	04:00pm	2	0	2
13/06/13	SYNOPSIS SUBMISSION	9:00am	11:00am	2	1	3
14/06/13	REQUIREMENT ANALYSIS PHASE	02:00pm	04:00pm	2	1	3
20/06/13	REQUIREMENT ANALYSIS PHASE	9:00am	11:00am	2	1	3
21/06/13	REQUIREMENT ANALYSIS PHASE	02:00pm	04:00pm	2	1	3
27/06/13	REQUIREMENT ANALYSIS PHASE	9.00am	11.00am	2	2	4
28/06/13	REQUIREMENT ANALYSIS PHASE	02:00pm	04:00pm	2	4	6
04/07/13	SYSTEM DESIGN PHASE	09:00am	11:00am	2	1	3

05/07/13	SYSTEM DESIGN PHASE	02:00pm	04:00pm	2	1	3
11/07/13	SYSTEM DESIGN PHASE	09.00am	11:00am	2	1	3
12/07/13	SYSTEM DESIGN PHASE	09.00am	11.00am	2	2	4
18/07/13	SYSTEM DESIGN PHASE	09.00am	11.00am	2	2	4
19/07/13	FIRST DRAFT	02.00pm	04.00pm	2	3	5
25/07/13	DEVELOPMENT PHASE	09.00am	11:00am	2	2	4
26/07/13	DEVELOPMENT PHASE	02:00pm	04:00pm	2	3	5
01/08/13	DEVELOPMENT PHASE	09.00am	11:00am	2	1	3
02/08/13	DEVELOPMENT PHASE	02:00pm	04:00pm	2	1	3
16/08/13	DEVELOPMENT PHASE	02:00pm	04:00pm	2	1	3
22/08/13	DEVELOPMENT PHASE	09.00am	11:00am	2	4	6
23/08/13	DEVELOPMENT PHASE	2:00pm	4:00pm	2	4	6
05/09/13	DEVELOPMENT PHASE	09.00am	11:00am	2	4	6
06/09/13	DOCUMENTATION PHASE	2:00pm	4:00pm	2	2	4

12/09/13	PROJECT DRAFT REPORT SUBMISSION	09.00am	11:00am	2	4	6
13/09/13	DOCUMENTATION PHASE	2:00pm	4:00pm	2	2	4
16/09/13	FINAL REPORT SUBMISSION	09.00am	1:00pm	0	1	1
19/09/13	TEST PHASE	09.00am	11:00am	2	4	6
20/09/13	TEST PHASE	2:00pm	4:00pm	2	4	6
	TOTALS:			54	57	111

Student Signature:	Date:
Guide Signature:	Date:

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

There are many text editors that provide useful functionality for programmers. Most support syntax highlighting for many programming languages, multiple document editing, and are extendable with plug-ins. Some also allow editing of remote files through FTP. Some of the short comings we found in the existing systems are pointed below. We intend to overcome these in our own implementation of the IDE software.

2.1.1 LIMITATIONS OF EXISTING SYSTEM

- No auto tag closing and matching tag highlight.
- Heavyweight and slower start-up.
- No collaborative editing
- No large file support
- No plug-in or macros
- No syntax highlighting
- No shared editing support
- Does not support server-side scripting
- No built-in FTP client

2.2 PROPOSED SYSTEM

The system we propose is a comprehensive code developing and sharing system that would encompass FTP support, open source, user friendly and intuitive interface. We will try to overcome the current problems faced by the existing text editors and at the same time increase the efficiency of all other aspects by performing small tweaks and changes. Some of these include providing instant code sharing mechanism and seeking aid from friends and teachers to resolve code errors quickly. This will help in the overall efficiency of the software. Our main focus is towards establishment of a network in order to make user communication easy.

2.2.1 BENEFITS OF PROPOSED SYSTEM

BEGINNER FRIENDLINESS

Meant to be easy to use and at the same time expose powerful features so that developers go through writing code, compiling, testing and deploying from a single application.

EXTENSIBILITY

The editor supports plug-in, bundles and extensions.

SPEED

Quick startup and faster performance time when opening large files.

CODE SHARING

Helps programmers resolve bugs in the source code by sharing code snippets with their peers at the touch of a button.

WIRELESS NETWORK

Code Sharer (CS) makes use of wireless LAN and avoids the hassle of a physical connection. Thus, it’s easy for clients to connect to the server from anywhere anytime.

2.3 SOFTWARE TOOLS

2.3.1 WINDOWS PRESENTATION FOUNDATION (WPF)

(WPF) is a next-generation presentation system for building Windows client applications with visually stunning user experiences. With WPF, you can create a wide range of both standalone and browser-hosted applications.

WPF application paves the way for the developers to define the thick client UI in a declarative way, which was never supported by the traditional .NET windows forms. Some of important features are declarative programming, independent of screen resolution, control inside control, control templates, control transforms, availability of different layouts like stack panel, wrap panel, dock panel, grid layout, canvas layout, etc.

2.3.2 MICROSOFT VISUAL STUDIO

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop console and graphical user interface applications along with Windows Forms or WPF applications, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and Microsoft Silver light.

Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source-control systems (like Subversion and Visual SourceSafe) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

3. SYSTEM REQUIREMENTS

3.1 SYSTEM MODEL

EDITOR

Code Sharer (CS) gives an interactive IDE which comprises of a very sophisticated editor. The main feature of the given editor is to provide programmers with a well-known and easy to use environment with features like syntax highlighting, bug highlighting, error indication with the exact error message, a tabbed interface to seamlessly switch between programs and a GCC compiler to compile all C/C++ codes on the go.

CHAT

A chat interface provides the easiest and fastest way of communication between peers. Code Sharer (CS) provides two modes of chatting viz. Single Chat and Multi Chat which provides a group chat between many users. With the help of this chat feature, users can acknowledge and address the problems of other users and explain their problems better.

SNIPPET SHARE

The snippet share feature is perhaps one of the most useful features of Code Sharer (CS). The main use of this feature is the sharing of code snippets either with another user or a group of other users. Code Sharer (CS) not only shares the code snippets with the desired receiver(s) but also highlights the bug(s) if any, on their screens. This helps them know exactly what the error is and saves plenty of time and effort.

SCREENSHOT

Users can not only share snippets but share screenshots as well. This can be useful especially in situations of graphic outputs where the user can just share the screenshot of the output with other users. Code Sharer (CS) provides an automatic screen detector which captures the entire screen with the drag of a mouse. Code Sharer (CS) also provides a manual mode for those moments when only a part of the screen needs to be captured.

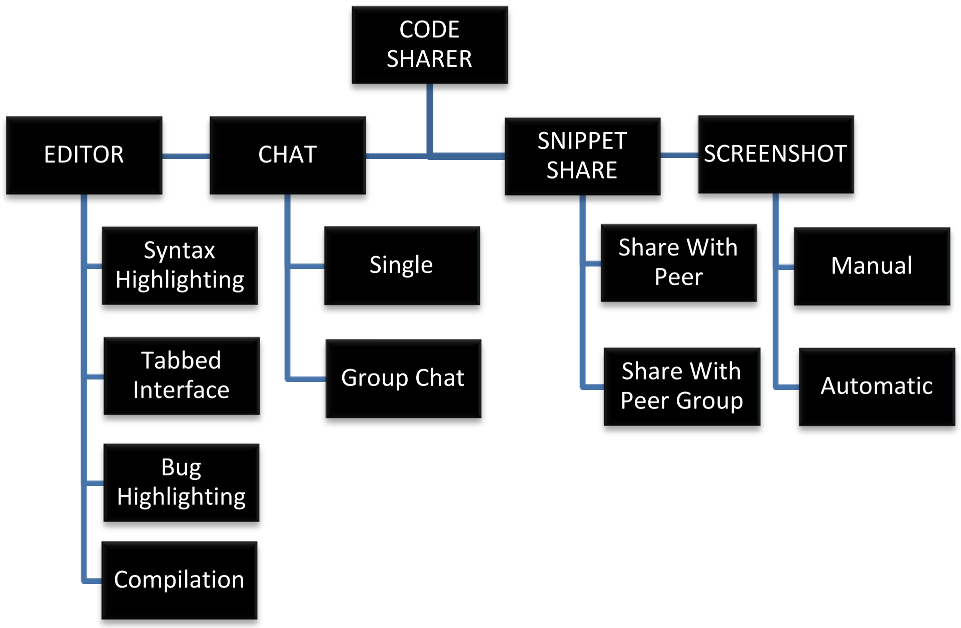


Fig 3.1 Code Sharer Block Diagram

3.2 FUNCTIONAL REQUIREMENTS

Table 3.1 Functional Requirements

MODULE NAME	INPUT	FUNCTIONALITY	REQUIREMENT
Snippet share / File share	Code snippets and files to be transferred on the network.	Programmers can share their programming codes and files easily with peers or groups by connecting to the network.	Built-in FTP client and server side scripting.
Screenshot/ Print Screen	Image file (JPEG or PNG) that displays the application status or the program's output screen.	The computer takes the image of the visible items on the monitor. Screen capture is used to demonstrate a program, a particular problem a user might be having, or generally when the display output needs to be shown to others or archived.	.NET Framework using the Graphics.CopyFromScreen method.
Chat	Text message	Allows participants to have a real-time synchronous discussion with peer or in groups. Includes features for managing and reviewing chat discussions.	Connection to local network and reference to NetworkCommsDotNet.
Editor	Programming languages like C and C++.	Development of applications.	A well designed GUI using WPF.

3.3 HARDWARE REQUIREMENTS

Table 3.2 Hardware Requirement

Requirement	
Processor	600 MHz processor Recommended: 1 gigahertz (GHz) processor
RAM	192 MB Recommended: 256 MB
Available Hard Disk Space	▪ 1 GB of available space required on system drive ▪ 2 GB of available space required on installation drive
Operating System	Windows 2000 Service Pack 4, Windows XP Service Pack 2, Windows Server 2003 Service Pack 1, or Windows Vista, Windows 7 or Windows 8. For a 64-bit computer, the requirements are as follows: ▪ Windows Server 2003 Service Pack 1 x64 editions ▪ Windows XP Professional x64 Edition ▪ Windows 7 ▪ Windows 8

3.4 SOFTWARE REQUIREMENTS

Table 3.3 Software Requirement

Front End	Visual Basic .NET
Operating System	Windows XP and above
Reference Files	• NetworkCommsDotNet_v2.3.0_DLLs • Winsock

4. DESIGN SPECIFICATION

4.1 ARCHITECTURAL DESIGN

The ‘Code Sharer’ software uses the two-tier architecture. The two-tier architecture here is divided into a client-services tier which here is the User and a server-services tier which here is the database. The Client Interface is made using C#, WPF and WCF while the database makes use of SQL Server. The two-tier architecture of Code Sharer (CS) is based on wireless network. The two-tier architecture of Code Sharer (CS) is aimed to connect one server to three clients.

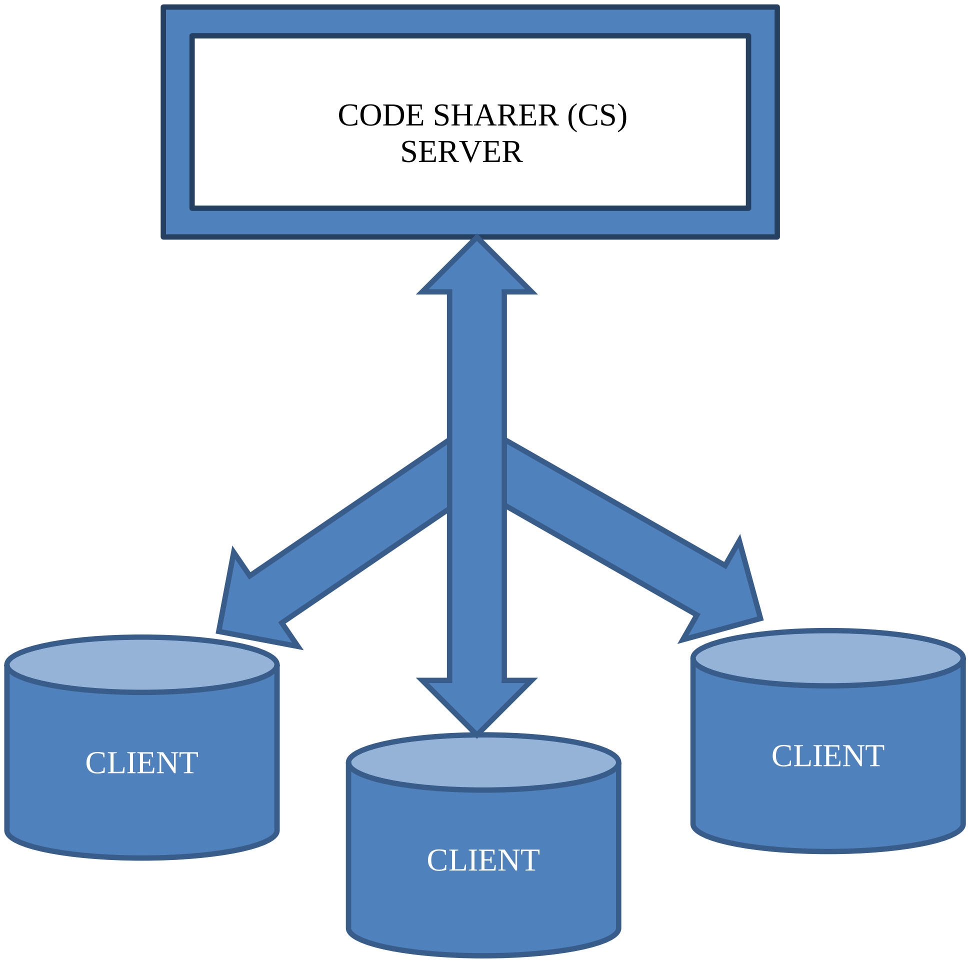


Fig 4.1 Two-tier architecture

4.2 DETAILED FLOW CHART

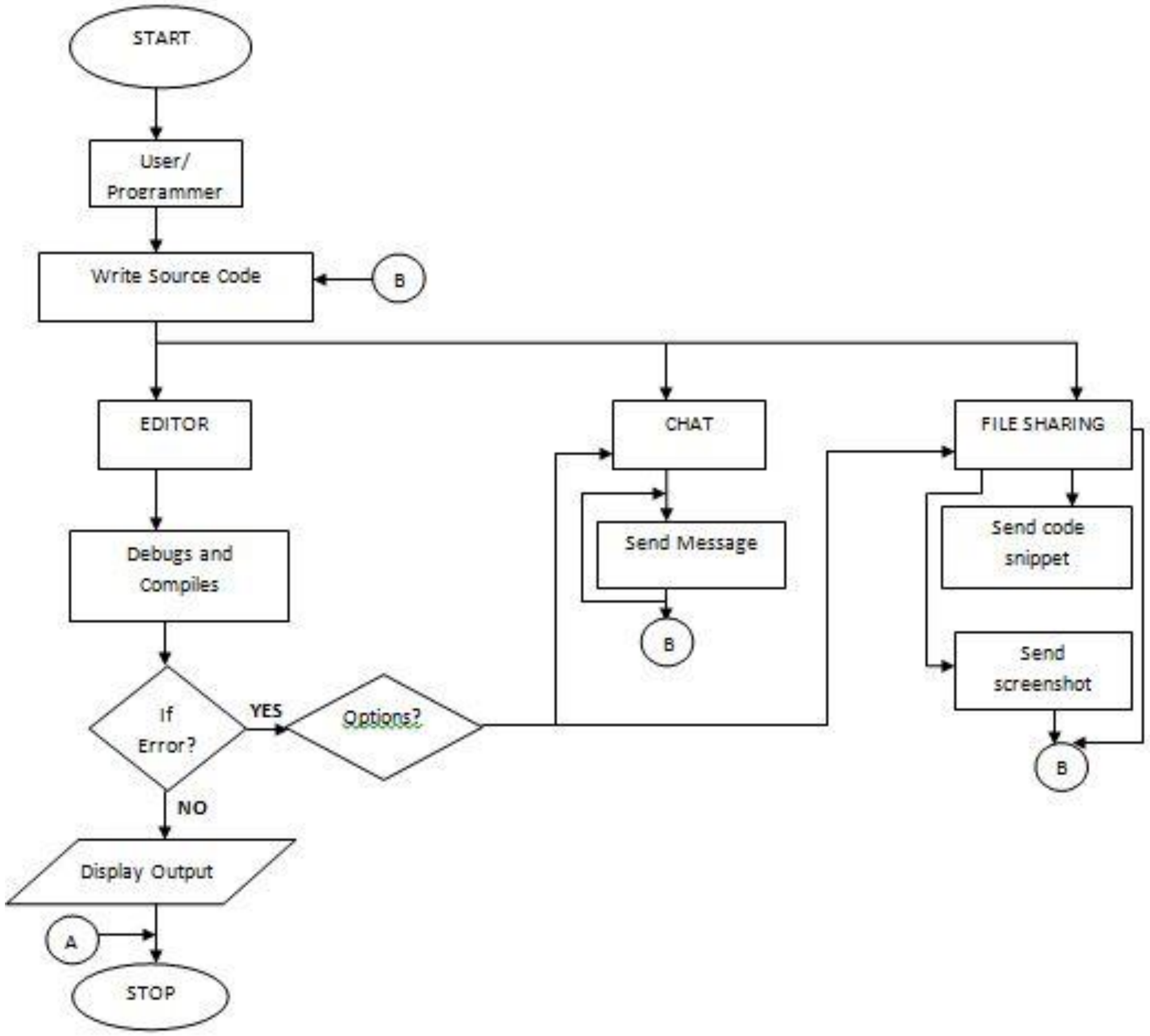


Fig 4.2 Detailed Flow Chart

4.2.1 EDITOR FLOW CHART

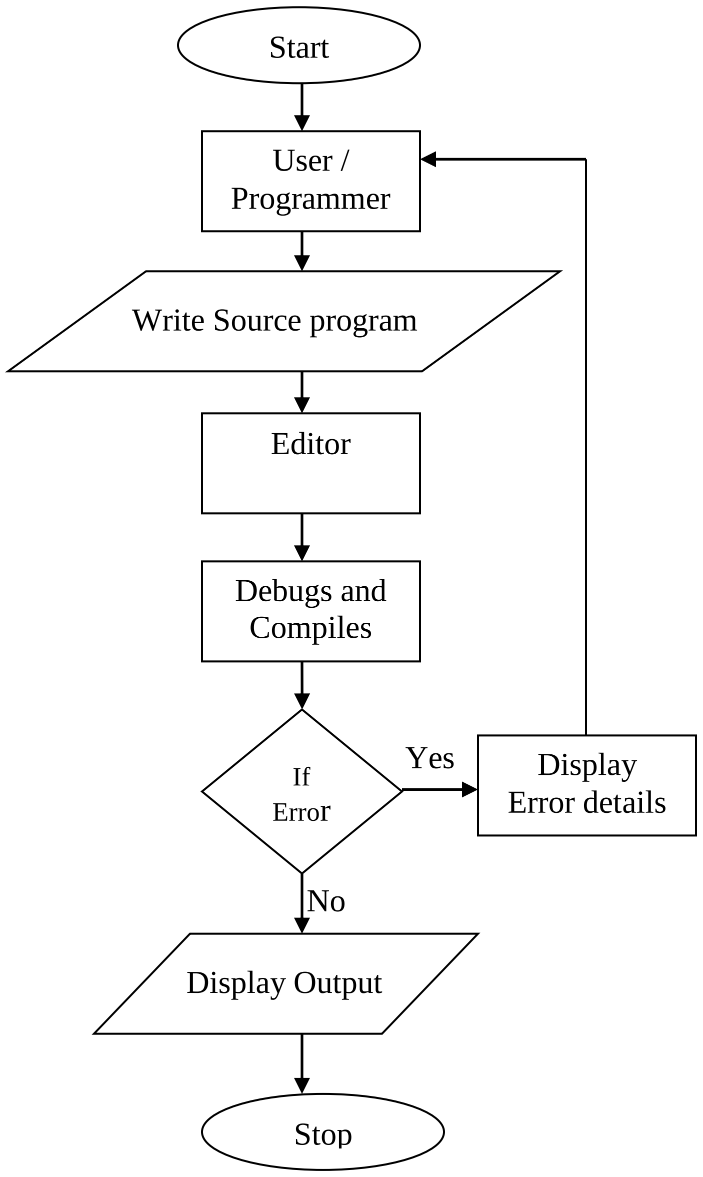


Fig 4.3 Editor Flow chart

4.2.2 CHAT FLOW CHART

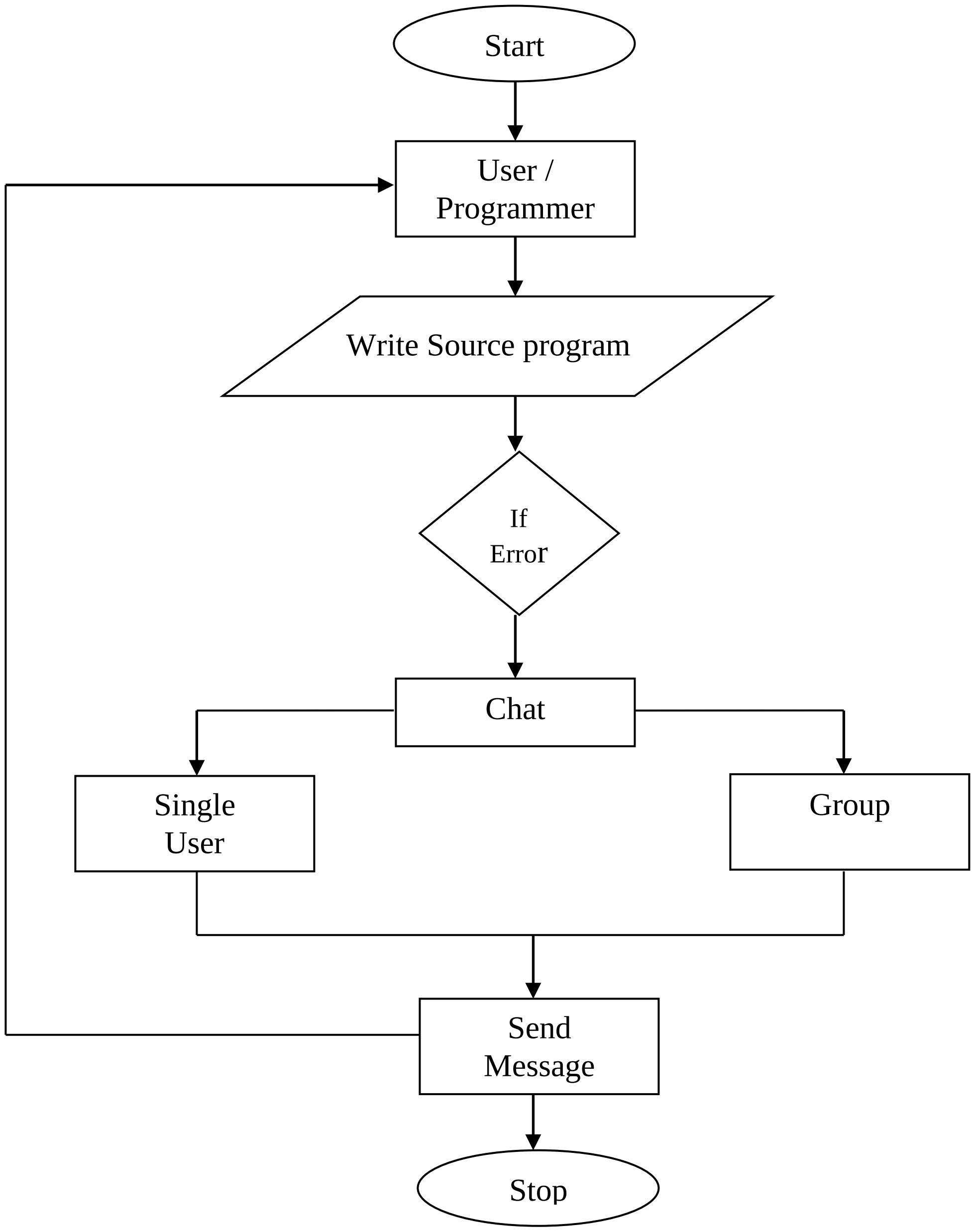


Fig 4.4 Chat Flow Chart

4.2.3 SNIPPET SHARE FLOW CHART

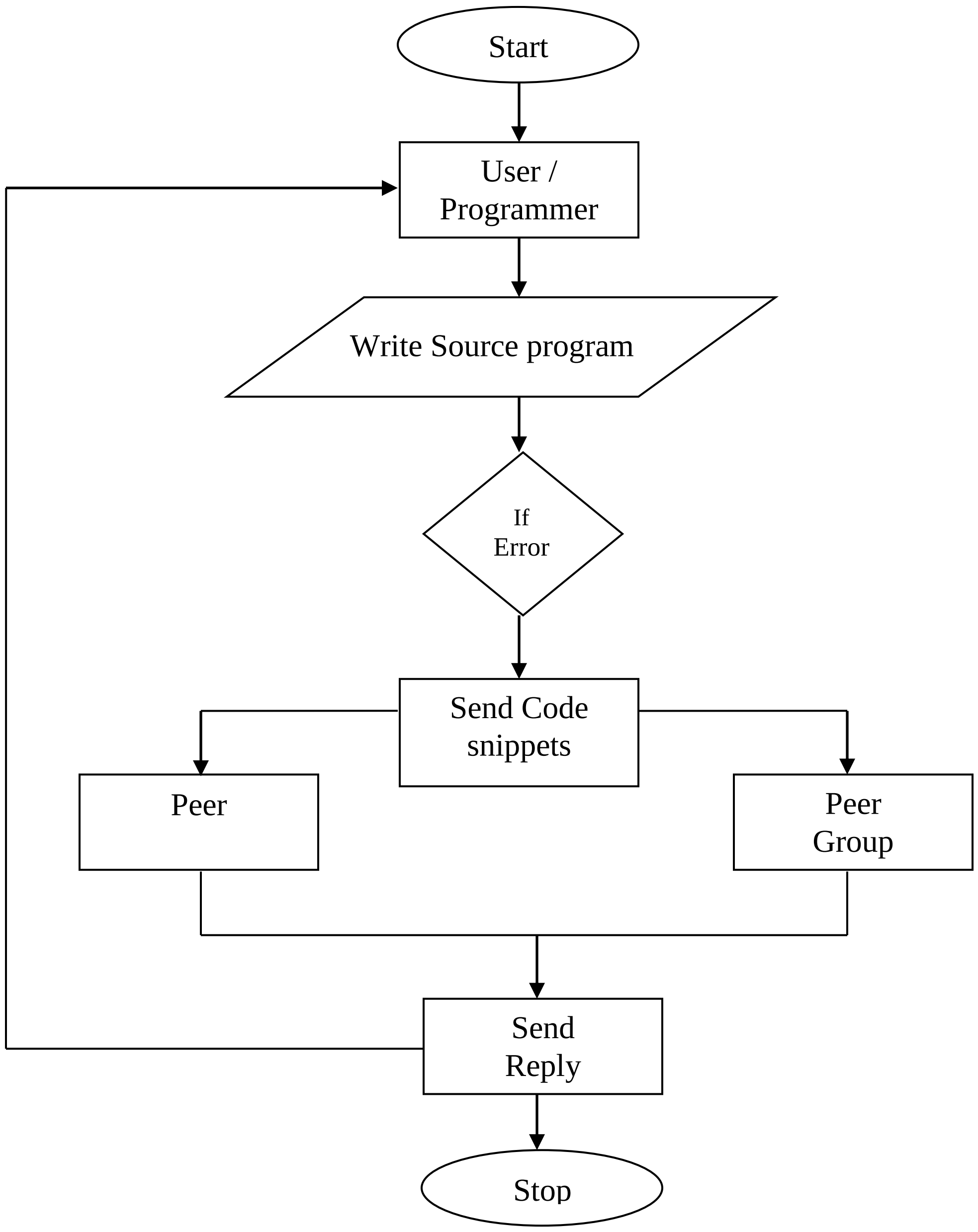


Fig 4.5 Snippet Share Flow Chart

4.2.4 SCREENSHOT FLOW CHART

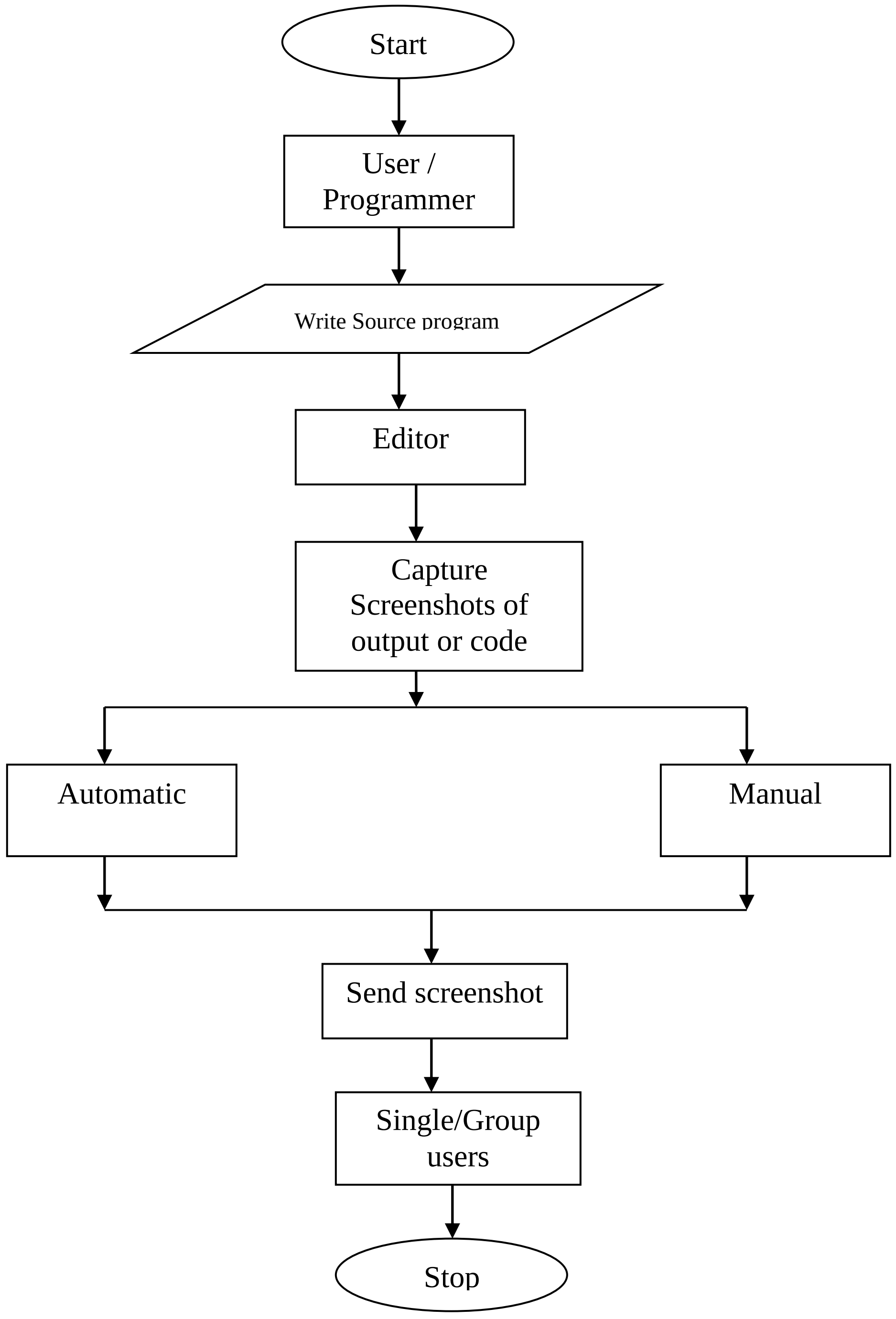


Fig 4.6 Screenshot Flow Chart

4.3 NETWORK SPECIFICATION

4.3.1 ALGORITHM SPECIFICATION

The algorithm used in Code Sharer (CS) is Selective Automatic Repeat Request (ARQ). Selective ARQ is an algorithm which sends multiple packets using the concept of sliding window. This algorithm makes use of a timer to send these packets and at the end of the timer, it checks for all the acknowledgements received for each packet. The packets for which acknowledgements are not received are re-sent.

The sliding window concept makes use of two pointers. One indicates the current packet to be sent and the other indicates the packets for which acknowledgement has been received. This algorithm is thus reliable and effective.

4.3.2 NETWORK CONFIGURATION

Computer should be wireless enabled with active port listeners.

4.3.3 TOPOLOGY DETAILS

The network setup of Code Sharer (CS) makes use of the star topology. This is because of the fact that there is one main server and several clients attached to the server. The star topology has the following advantages:

- Central hub can control all connected devices
- Simple setup and installation
- Easy to detect network failure

- Easy to add more clients to the network
- Minimum number of connections between devices
- Direct contact between server and client
- Failure of one client does not affect the others

The star topology has some disadvantages too such as:

- Too much dependency on the central hub
- Hub failure leads to downfall of entire network
- No direct contact between clients
- Number of clients that can be added depends upon the capacity of the server

4.3.4 PROTOCOL DETAILS

The protocols used in Code Sharer (CS) are File Transfer Protocol (FTP) and Internet Relay Chat Protocol (IRC). FTP, as the name suggests is used for file transfer and IRC is used for interactive text messaging over the internet.

4.3.5 ROUTER DETAILS

- Standard 150 Mbps wireless N
- Standard IEEE 802.11b and IEEE 802.3
- Frequency range of 2.4GHz-2.4835GHz
- Minimum of 5dBl antenna capacity to take load of 3 systems within a range of 100m
- Standard encryption and VPN support
- Additional: Firewall and port forwarding options

4.4 USER INTERFACE DESIGN

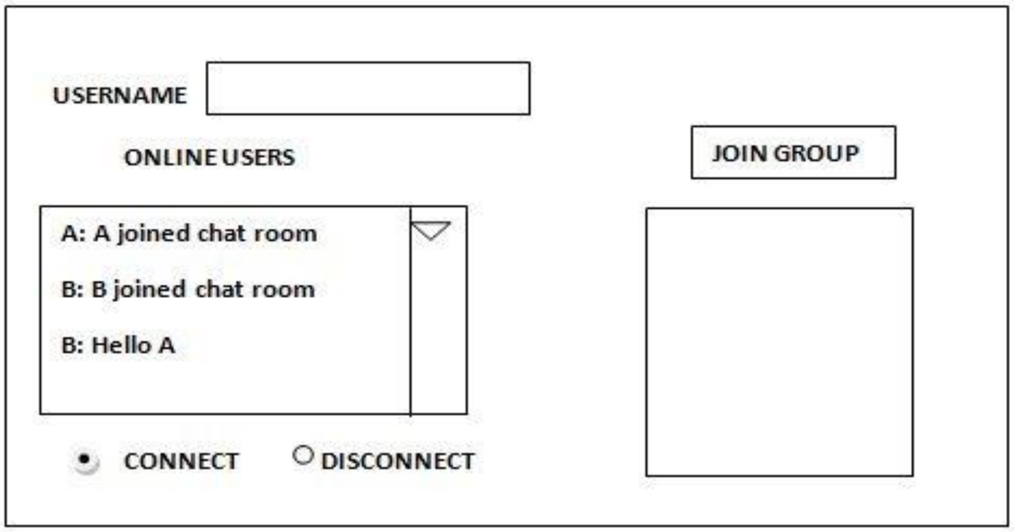


Fig 4.7 Group Chat

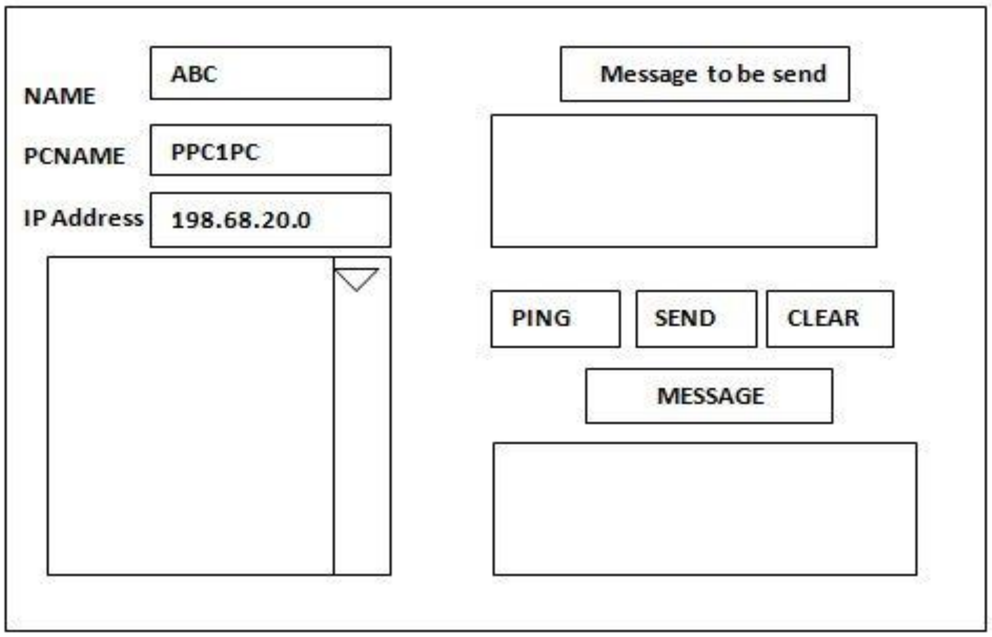


Fig 4.8 Chat Interface

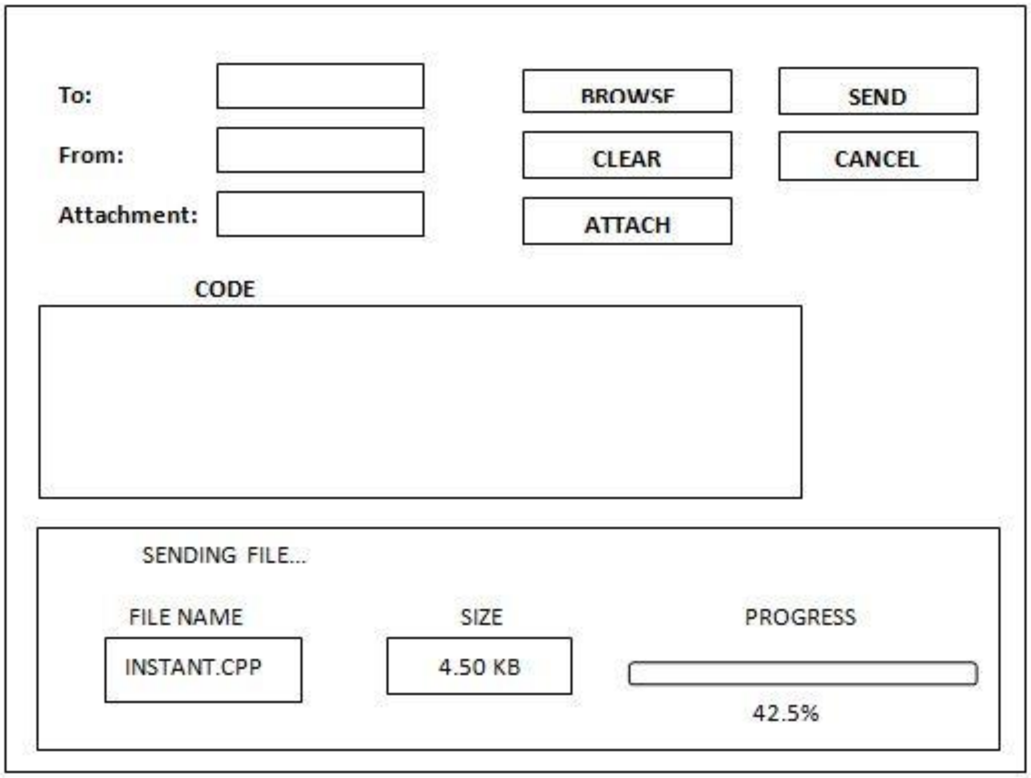


Fig 4.9 File Sharing

5. IMPLEMENTATION

5.1 SOURCE CODE

CHAT SERVER CODE

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Net;
using System.Net.Sockets;
using Proshot.CommandServer;
using System.ComponentModel;
namespace ConsoleServer
{
    class Program
    {
        private System.Collections.Generic.List<ClientManager> clients;
        private BackgroundWorker bwListener;
        private Socket listenerSocket;
        private IPAddress serverIP;
        private int serverPort;

        static void Main(string [] args)
        {
            Program progDomain = new Program();
            progDomain.clients = new List<ClientManager>();

            if ( args.Length == 0 )
            {
                progDomain.serverPort = 8000;
                progDomain.serverIP = IPAddress.Any;
            }
            if ( args.Length == 1 )
            {
                progDomain.serverIP = IPAddress.Parse(args [0]);
                progDomain.serverPort = 8000;
            }
            if ( args.Length == 2 )
            {
                progDomain.serverIP = IPAddress.Parse(args [0]);
                progDomain.serverPort = int.Parse(args [1]);
            }
         progDomain.bwListener = new BackgroundWorker();
            progDomain.bwListener.WorkerSupportsCancellation = true;
            progDomain.bwListener.DoWork += new
DoWorkEventHandler(progDomain.StartToListen);
            progDomain.bwListener.RunWorkerAsync();

            Console.WriteLine("*** Listening on port {0}{1}{2} started.Press ENTER to
shutdown server.
***\n",progDomain.serverIP.ToString(),":",progDomain.serverPort.ToString());

            Console.ReadLine();

            progDomain.DisconnectServer();
        }
        private void StartToListen(object sender , DoWorkEventArgs e)
        {
            this.listenerSocket = new Socket(AddressFamily.InterNetwork ,
SocketType.Stream , ProtocolType.Tcp);
            this.listenerSocket.Bind(new IPEndPoint(this.serverIP , this.serverPort));
            this.listenerSocket.Listen(200);
```

```
while ( true )
            this.CreateNewClientManager(this.listenerSocket.Accept());
      }
      private void CreateNewClientManager(Socket socket)
      {
         ClientManager newClientManager = new ClientManager(socket);
         newClientManager.CommandReceived += new
CommandReceivedEventHandler(CommandReceived);
         newClientManager.Disconnected += new
DisconnectedEventHandler(ClientDisconnected);
         this.CheckForAbnormalDC(newClientManager);
         this.clients.Add(newClientManager);
         this.UpdateConsole("Connected." , newClientManager.IP ,
newClientManager.Port);
      }

      private void CheckForAbnormalDC(ClientManager mngr)
      {
         if ( this.RemoveClientManager(mngr.IP) )
            this.UpdateConsole("Disconnected." , mngr.IP , mngr.Port);
      }
 void ClientDisconnected(object sender , ClientEventArgs e)

      {
         if ( this.RemoveClientManager(e.IP) )
            this.UpdateConsole("Disconnected." , e.IP , e.Port);
      }
 private bool RemoveClientManager(IPAddress ip)
        {
            lock ( this )
            {
                int index = this.IndexOfClient(ip);
                if ( index != -1 )
                {
                    string name = this.clients [index].ClientName;
                    this.clients.RemoveAt(index);
                    //Inform all clients that a client had been disconnected.
                    Command cmd = new Command(CommandType.ClientLogOffInform ,
IPAddress.Broadcast);
                    cmd.SenderName = name;
                    cmd.SenderIP = ip;
                    this.BroadCastCommand(cmd);
                    return true;
                }
                return false;
            }
        }
        private int IndexOfClient(IPAddress ip)
        {
            int index = -1;
            foreach ( ClientManager cMngr in this.clients )
            {
                index++;
                if ( cMngr.IP.Equals(ip) )
                    return index;
            }
            return -1;
        }
 private void CommandReceived(object sender , CommandEventArgs e)
        {
          if ( e.Command.CommandType == CommandType.ClientLoginInform )
               this.SetManagerName(e.Command.SenderIP , e.Command.MetaData);



            if ( e.Command.CommandType == CommandType.IsNameExists )
            {
                bool isExixsts = this.IsNameExists(e.Command.SenderIP ,
e.Command.MetaData);
                this.SendExistanceCommand(e.Command.SenderIP , isExixsts);
                return;
            }
            //If the client asked for list of a logged in clients,replay to it's request.
            else if ( e.Command.CommandType == CommandType.SendClientList )
            {
                this.SendClientListToNewClient(e.Command.SenderIP);
                return;
            }
            if ( e.Command.Target.Equals(IPAddress.Broadcast) )
                this.BroadCastCommand(e.Command);
            else
                this.SendCommandToTarget(e.Command);


        }


        private void SendExistanceCommand(IPAddress targetIP , bool isExists)
        {
            Command cmdExistance = new Command(CommandType.IsNameExists ,
targetIP , isExists.ToString());
            cmdExistance.SenderIP = this.serverIP;
            cmdExistance.SenderName = "Server";
            this.SendCommandToTarget(cmdExistance);
        }

        private void SendClientListToNewClient(IPAddress newClientIP)
        {
            foreach ( ClientManager mngr in this.clients )
            {
                if ( mngr.Connected && !mngr.IP.Equals(newClientIP) )
                {
                    Command cmd = new Command(CommandType.SendClientList ,
newClientIP);
                    cmd.MetaData = mngr.IP.ToString() + ":" + mngr.ClientName;
                     cmd.SenderIP = this.serverIP;
                    cmd.SenderName = "Server";
                    this.SendCommandToTarget(cmd);
                }
            }
        }

        private string SetManagerName(IPAddress remoteClientIP , string nameString)
        {
            int index = this.IndexOfClient(remoteClientIP);
            if ( index != -1 )
            {
                string name = nameString.Split(new char [] { ':' }) [1];
                this.clients [index].ClientName = name;
                return name;
            }
            return "";
        }
```

```
    private bool IsNameExists(IPAddress remoteClientIP , string nameToFind)
    {
       foreach ( ClientManager mngr in this.clients )
          if ( mngr.ClientName == nameToFind && !mngr.IP.Equals(remoteClientIP) )
             return true;
       return false;
    }

    private void BroadCastCommand(Command cmd)
    {
       foreach ( ClientManager mngr in this.clients )
          if ( !mngr.IP.Equals(cmd.SenderIP) )
             mngr.SendCommand(cmd);
    }

    private void SendCommandToTarget(Command cmd)
    {
       foreach ( ClientManager mngr in this.clients )
          if ( mngr.IP.Equals(cmd.Target) )
          {
             mngr.SendCommand(cmd);
             return;
          }
    }
   private void UpdateConsole(string status , IPAddress IP , int port)
    {
       Console.WriteLine("Client {0}{1}{2} has been {3} ( {4}|{5} )" ,
IP.ToString(),":" , port.ToString() ,
status,DateTime.Now.ToShortDateString(),DateTime.Now.ToLongTimeString());
    }
```

```
 public void DisconnectServer()
     {
         if ( this.clients != null )
         {
             foreach ( ClientManager mngr in this.clients )
                 mngr.Disconnect();

             this.bwListener.CancelAsync();
             this.bwListener.Dispose();
             this.listenerSocket.Close();
             GC.Collect();
         }
     }
  }
}
```

CHAT CLIENT CODE

Login

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Net;
using Proshot.UtilityLib.CommonDialogs;
```

```
namespace ChatClient
{
    public partial class frmLogin : Form
    {
        private bool canClose;
        private Proshot.CommandClient.CMDClient client;
        public Proshot.CommandClient.CMDClient Client
{

            get { return client; }
        }
        public frmLogin(IPAddress serverIP,int serverPort)
        {
            InitializeComponent();
            this.canClose = false;
            Control.CheckForIllegalCrossThreadCalls = false;
            this.client = new Proshot.CommandClient.CMDClient(serverIP , serverPort ,
"None");
            this.client.CommandReceived += new
Proshot.CommandClient.CommandReceivedEventHandler(CommandReceived);
            this.client.ConnectingSuccessed += new
Proshot.CommandClient.ConnectingSuccessedEventHandler(client_ConnectingSuccesse
d);
            this.client.ConnectingFailed += new
Proshot.CommandClient.ConnectingFailedEventHandler(client_ConnectingFailed);
        }
        private void client_ConnectingFailed(object sender , EventArgs e)
        {
            frmPopup popup = new frmPopup(PopupSkins.SmallInfoSkin);
            popup.ShowPopup("Error" , "Server Is Not Accessible !" , 200 , 2000 , 2000);
            this.SetEnablity(true);
        }
        private void client_ConnectingSuccessed(object sender , EventArgs e)
        {
          this.client.SendCommand(new
Proshot.CommandClient.Command(Proshot.CommandClient.CommandType.IsNameExi
sts,this.client.IP,this.client.NetworkName));
        }
        void CommandReceived(object sender ,
Proshot.CommandClient.CommandEventArgs e)
        {
            if ( e.Command.CommandType ==
Proshot.CommandClient.CommandType.IsNameExists )
            {
                if ( e.Command.MetaData.ToLower() == "true" )
                {
                    frmPopup popup = new frmPopup(PopupSkins.SmallInfoSkin);
                    popup.ShowPopup("Error" , "This username is already taken !" , 300 , 2000 ,
2000);
this.client.Disconnect();
                    this.SetEnablity(true);
                }
                else
                {
                    this.canClose = true;
                    this.Close();
                }
            }
        }
        private void LoginToServer()
        {
            this.UserName.Text = Environment.UserName;
            this.client.NetworkName = Environment.UserName;
            this.client.ConnectToServer();
        }
        private void btnEnter_Click(object sender , EventArgs e)
        {
            this.SetEnablity(false);
            this.LoginToServer();
        }
        private void SetEnablity(bool enable)
        {
            this.btnEnter.Enabled = enable;
            this.UserName.Enabled = enable;
            this.btnExit.Enabled = enable;
        }

        private void btnExit_Click(object sender , EventArgs e)
        {
            this.canClose = true;
        }

        private void frmLogin_FormClosing(object sender , FormClosingEventArgs e)
        {
            if ( !this.canClose )
                e.Cancel = true;
            else
                this.client.CommandReceived -= new
Proshot.CommandClient.CommandReceivedEventHandler(CommandReceived);
        }
private void frmLogin_Load(object sender, EventArgs e)
        {
            this.UserName.Enabled = false;
            //this.UserName.Text =
System.Security.Principal.WindowsIdentity.GetCurrent().Name;
            this.UserName.Text = Environment.UserName;
            frmPopup popup = new frmPopup(PopupSkins.SmallInfoSkin);
            popup.ShowPopup("Login ID", this.UserName.Text, 1000, 2000, 2000);
            //serverip=
        }
    }
}
```

PRIVATE CHAT

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Proshot.CommandClient;
using System.Net;
using System.Runtime.InteropServices;
namespace ChatClient
{
    internal enum FlashMode
    {
        FLASHW_CAPTION = 0x1 ,
        FLASHW_TRAY = 0x2 ,
        FLASHW_ALL = FLASHW_CAPTION | FLASHW_TRAY
    }
    internal struct FlashInfo
    {
        public int cdSize;
        public System.IntPtr hwnd;
        public int dwFlags;
        public int uCount;
        public int dwTimeout;
    }
    public partial class frmPrivate : Form
    {
        private CMDClient remoteClient;
        private IPAddress targetIP;
        private string remoteName;
        private bool activated;
        public string RemoteName
        {
            get { return this.remoteName; }
        }
        protected override bool ProcessCmdKey(ref Message msg , Keys keyData)
        {
            if ( keyData == Keys.Enter )
                this.SendMessage();
            if ( this.txtMessages.Focused &&
!ShareUtils.IsValidKeyForReadOnlyFields(keyData) )
                return true;
            return base.ProcessCmdKey(ref msg , keyData);
        }
        public frmPrivate(CMDClient cmdClient,IPAddress friendIP,string name,string
initialMessage)
```

```
 {
            InitializeComponent();
            this.remoteClient = cmdClient;
            this.targetIP = friendIP;
            this.remoteName = name;
            this.Text += " With " + name;
            this.txtMessages.Text = this.remoteName + ": " + initialMessage
+Environment.NewLine;
            this.remoteClient.CommandReceived += new
CommandReceivedEventHandler(private_CommandReceived);
        }

        public frmPrivate(CMDClient cmdClient , IPAddress friendIP , string name)
        {
            InitializeComponent();
            this.remoteClient = cmdClient;
            this.targetIP = friendIP;
            this.remoteName = name;
            this.Text += " With " + name;
            this.remoteClient.CommandReceived += new
CommandReceivedEventHandler(private_CommandReceived);
        }

        private void private_CommandReceived(object sender , CommandEventArgs e)
        {
            switch ( e.Command.CommandType )
            {
                case ( CommandType.Message ):
                    if ( !e.Command.Target.Equals(IPAddress.Broadcast) &&
e.Command.SenderIP.Equals(this.targetIP))
                    {
                        this.txtMessages.Text += e.Command.SenderName + ": " +
e.Command.MetaData + Environment.NewLine;
                        if ( !this.activated)
                        {
                            if(this.WindowState == FormWindowState.Normal || this.WindowState
== FormWindowState.Maximized)

ShareUtils.PlaySound(ShareUtils.SoundType.NewMessageReceived);
                            else
ShareUtils.PlaySound(ShareUtils.SoundType.NewMessageWithPow);
                            this.Flash(this.Handle , FlashMode.FLASHW_ALL , 3);
                        }
                    }
                    break;
            }
        }
        [DllImport("user32.dll")]
        private static extern int FlashWindowEx(ref FlashInfo pfwi);
        private void Flash(System.IntPtr hwnd , FlashMode flashMode , int times)
        {
            unsafe
            {
                FlashInfo FlashInf = new FlashInfo();
                FlashInf.cdSize = sizeof(FlashInfo);
                FlashInf.dwFlags = (int)flashMode;
                FlashInf.dwTimeout = 0;
                FlashInf.hwnd = hwnd;
                FlashInf.uCount = times;
                FlashWindowEx(ref FlashInf);
            }
        }
        private void btnSend_Click(object sender , EventArgs e)
        {
            this.SendMessage();
        }

        private void SendMessage()
        {
            if ( this.remoteClient.Connected && this.txtNewMessage.Text.Trim() != "")
            {
                this.remoteClient.SendCommand(new
Proshot.CommandClient.Command(Proshot.CommandClient.CommandType.Message ,
this.targetIP , this.txtNewMessage.Text));
                this.txtMessages.Text += this.remoteClient.NetworkName + ": " +
this.txtNewMessage.Text.Trim() + Environment.NewLine;
                this.txtNewMessage.Text = "";
                this.txtNewMessage.Focus();
            }
        }

        private void frmPrivate_FormClosing(object sender , FormClosingEventArgs e)
        {
            this.remoteClient.CommandReceived -= new
CommandReceivedEventHandler(private_CommandReceived);
        }

        private void mniExit_Click(object sender , EventArgs e)
        {
            this.Close();
        }

        private void mniSave_Click(object sender , EventArgs e)
        {
            SaveFileDialog saveDlg = new SaveFileDialog();
            saveDlg.Filter = "HTML Files(*.HTML;*.HTM)|*.html;*.htm";
            saveDlg.FilterIndex = 0;
            saveDlg.RestoreDirectory = true;
            saveDlg.CheckPathExists = true;
            saveDlg.OverwritePrompt = true;
            saveDlg.FileName = this.Text;
            if ( saveDlg.ShowDialog() == DialogResult.OK )
                ShareUtils.SaveAsHTML(saveDlg.FileName , this.txtMessages.Lines ,
this.Text);
        }
private void frmPrivate_Load(object sender , EventArgs e)
        {
            this.Location = new Point(Screen.PrimaryScreen.Bounds.Width - this.Width ,
Screen.PrimaryScreen.WorkingArea.Height - this.DesktopBounds.Height);
        }

        private void frmPrivate_Activated(object sender , EventArgs e)
        {
            this.activated = true;
        }

        private void frmPrivate_Deactivate(object sender , EventArgs e)
        {
            this.activated = false;
        }

        private void txtMessages_TextChanged(object sender, EventArgs e)
        {
            txtMessages.SelectionStart = txtMessages.Text.Length;
            txtMessages.ScrollToCaret();
        }

        private void mniCopy_Click(object sender, EventArgs e)
        {
            this.txtNewMessage.Copy();
        }

        private void mniPaste_Click(object sender, EventArgs e)
        {
            this.txtNewMessage.Paste();
        }

        private void mniCopyText_Click(object sender, EventArgs e)
        {
            this.txtMessages.Copy();
        }
    }
}
```

SCREEN CAPTURE

```
using System;
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using System.Windows.Navigation;
using System.Drawing;
using System.Runtime.InteropServices;
using System.Drawing.Imaging;
using System.Windows.Forms;

namespace ScreenShot
{
    /// <summary>
    /// Interaction logic for Window1.xaml
    /// </summary>
    public partial class Window1 : Window
    {
        //Global variables
        public double x;
        public double y;
        public double width;
        public double height;
        public bool isMouseDown = false;
        public Window1()
        {
            InitializeComponent();
        }

        private void Window_MouseDown(object sender, MouseButtonEventArgs e)
        {
            isMouseDown = true;
            x = e.GetPosition(null).X;
            y = e.GetPosition(null).Y;
        }
        private void Window_MouseMove(object sender,
System.Windows.Input.MouseEventArgs e)
        {
            if (this.isMouseDown)
        {
            double curx = e.GetPosition(null).X;
            double cury = e.GetPosition(null).Y;

             System.Windows.Shapes.Rectangle r = new
             System.Windows.Shapes.Rectangle();
            SolidColorBrush brush = new SolidColorBrush(Colors.White);
            r.Stroke = brush;
            r.Fill = brush;
            r.StrokeThickness = 1;
            r.Width = Math.Abs(curx - x);
            r.Height = Math.Abs(cury - y);
            cnv.Children.Clear();
            cnv.Children.Add(r);
            Canvas.SetLeft(r, x);
            Canvas.SetTop(r, y);
            if (e.LeftButton == MouseButtonState.Released)
            {
                cnv.Children.Clear();
                width = e.GetPosition(null).X - x;
                height = e.GetPosition(null).Y - y;
                this.CaptureScreen(x, y, width, height);
                this.x = this.y = 0;
                this.isMouseDown = false;
                this.Close();
            }
        }
    }
```

```
  public void CaptureScreen(double x, double y, double width, double height)
      {
          int ix, iy, iw, ih;
          ix = Convert.ToInt32(x);
          iy = Convert.ToInt32(y);
          iw = Convert.ToInt32(width);
          ih = Convert.ToInt32(height);
          Bitmap image = new Bitmap(iw, ih,
System.Drawing.Imaging.PixelFormat.Format32bppArgb);
          Graphics g = Graphics.FromImage(image);
          g.CopyFromScreen(ix, iy, ix, iy, new System.Drawing.Size(iw, ih),
CopyPixelOperation.SourceCopy);
          SaveFileDialog dlg = new SaveFileDialog();
          dlg.DefaultExt = "png";
          dlg.Filter = "Png Files|*.png|JPEG Files|*.jpeg|BMP File|*.bmp";

          DialogResult res = dlg.ShowDialog();
          if (res == System.Windows.Forms.DialogResult.OK)
              image.Save(dlg.FileName, ImageFormat.Png);
  /*        Bitmap myImage = new Bitmap(iw, ih);


    ImageFormat format = ImageFormat.Png;
if (dlg.ShowDialog() == System.Windows.Forms.DialogResult.OK)
{
    string ext = System.IO.Path.GetExtension(dlg.FileName);
    switch (ext)
    {
        case ".jpg":
            format = ImageFormat.Jpeg;
            break;
      case ".bmp":
          format = ImageFormat.Bmp;
          break;
    }
    myImage.Save(dlg.FileName, format);
}*/
      }

      public void SaveScreen(double x, double y, double width, double height)
      {
          int ix, iy, iw, ih;
          ix = Convert.ToInt32(x);
          iy = Convert.ToInt32(y);
          iw = Convert.ToInt32(width);
          ih = Convert.ToInt32(height);
          try
          {
              Bitmap myImage = new Bitmap(iw, ih);
              Graphics gr1 = Graphics.FromImage(myImage);
              IntPtr dc1 = gr1.GetHdc();
              IntPtr dc2 =
NativeMethods.GetWindowDC(NativeMethods.GetForegroundWindow());
              NativeMethods.BitBlt(dc1, ix, iy, iw, ih, dc2, ix, iy, 13369376);
              gr1.ReleaseHdc(dc1);
SaveFileDialog dlg = new SaveFileDialog();
              dlg.DefaultExt = "png";
              dlg.Filter = "Png Files|*.png|JPEG File|*.jpeg|BMP file|*.bmp";
              /*DialogResult res = dlg.ShowDialog();
              if (res == System.Windows.Forms.DialogResult.OK)
                  myImage.Save(dlg.FileName, ImageFormat.Png);
               */
```

```
ImageFormat format = ImageFormat.Png;
                if (dlg.ShowDialog() == System.Windows.Forms.DialogResult.OK)
                {
                    string ext = System.IO.Path.GetExtension(dlg.FileName);
                    switch (ext)
                    {
                        case ".jpg":
                            format = ImageFormat.Jpeg;
                            break;
                        case ".bmp":
                            format = ImageFormat.Bmp;
                            break;
                    }
                    myImage.Save(dlg.FileName, format);
                }catch { }
        }
        internal class NativeMethods
        {
            [DllImport("user32.dll")]
            public extern static IntPtr GetDesktopWindow();
            [DllImport("user32.dll")]
            public static extern IntPtr GetWindowDC(IntPtr hwnd);
            [DllImport("user32.dll", CharSet = CharSet.Auto, ExactSpelling = true)]
            public static extern IntPtr GetForegroundWindow();
            [DllImport("gdi32.dll")]
            public static extern UInt64 BitBlt(IntPtr hDestDC, int x, int y, int nWidth, int
nHeight, IntPtr hSrcDC, int xSrc, int ySrc, System.Int32 dwRop);
        }
    }
}
```

FILE TRANSFER

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using NetworkCommsDotNet;
using System.ComponentModel;
using System.IO;

namespace FileTransfer
{
    class ReceivedFile : INotifyPropertyChanged
    {
        public string Filename { get; private set; }
        public ConnectionInfo SourceInfo { get; private set; }
        public long SizeBytes { get; private set; }
        public int ReceivedBytes { get; private set; }
        public double CompletedPercent
        {
            get { return (double)ReceivedBytes / SizeBytes; }
            set { throw new Exception("An attempt to modify readonly value."); }
        }

        public string SourceInfoStr
        {
            get { return "[" + SourceInfo.RemoteEndPoint.Address + ":" +
SourceInfo.RemoteEndPoint.Port + "]"; }
        }
        public bool IsCompleted
        {
            get { return ReceivedBytes == SizeBytes; }
        }
        object SyncRoot = new object();
        MemoryStream data;
        public event PropertyChangedEventHandler PropertyChanged;
   public ReceivedFile(string filename, ConnectionInfo sourceInfo, long sizeBytes)
        {
            this.Filename = filename;
            this.SourceInfo = sourceInfo;
            this.SizeBytes = sizeBytes;

            if (this.SizeBytes > int.MaxValue)
                throw new NotSupportedException("The provided sizeBytes is not supported
for this example.");

            data = new MemoryStream((int)sizeBytes); ;
        }
memoryStream</param>
        public void AddData(long dataStart, long bufferStart, long bufferLength, byte[]
buffer)
        {
            if (bufferStart > int.MaxValue)
                throw new NotSupportedException("The provided bufferStart is not supported
for this example.");

            if (bufferLength > int.MaxValue)
                throw new NotSupportedException("The provided bufferLength is not
supported for this example.");
```

```
            lock (SyncRoot)
            {
                data.Seek(dataStart, SeekOrigin.Begin);
                data.Write(buffer, (int)bufferStart, (int)bufferLength);

                ReceivedBytes += (int)(bufferLength - bufferStart);
            }
            NotifyPropertyChanged("CompletedPercent");
            NotifyPropertyChanged("IsCompleted");
        }
        public void SaveFileToDisk(string saveLocation)
        {
            if (ReceivedBytes != SizeBytes)
                throw new Exception("Attempted to save out file before data is complete.");

            lock (SyncRoot)
                File.WriteAllBytes(saveLocation, data.ToArray());
        }

        private void NotifyPropertyChanged(string propertyName = "")
        {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs(propertyName));
        }
    }
}
```

```
//SEND INFORMATION CLASS
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using ProtoBuf;
namespace FileTransfer
{
   [ProtoContract]
   class SendInfo
   {
      [ProtoMember(1)]
      public string Filename { get; private set; }
      [ProtoMember(2)]
      public long BytesStart { get; private set; }
      [ProtoMember(3)]
      public long TotalBytes { get; private set; }
      [ProtoMember(4)]
      public long PacketSequenceNumber { get; private set; }
      private SendInfo() { }
     public SendInfo(string filename, long totalBytes, long bytesStart, long
packetSequenceNumber)
      {
         this.Filename = filename;
         this.TotalBytes = totalBytes;
         this.BytesStart = bytesStart;
         this.PacketSequenceNumber = packetSequenceNumber;
      }
   }
}
```

5.2 SCREENSHOTS

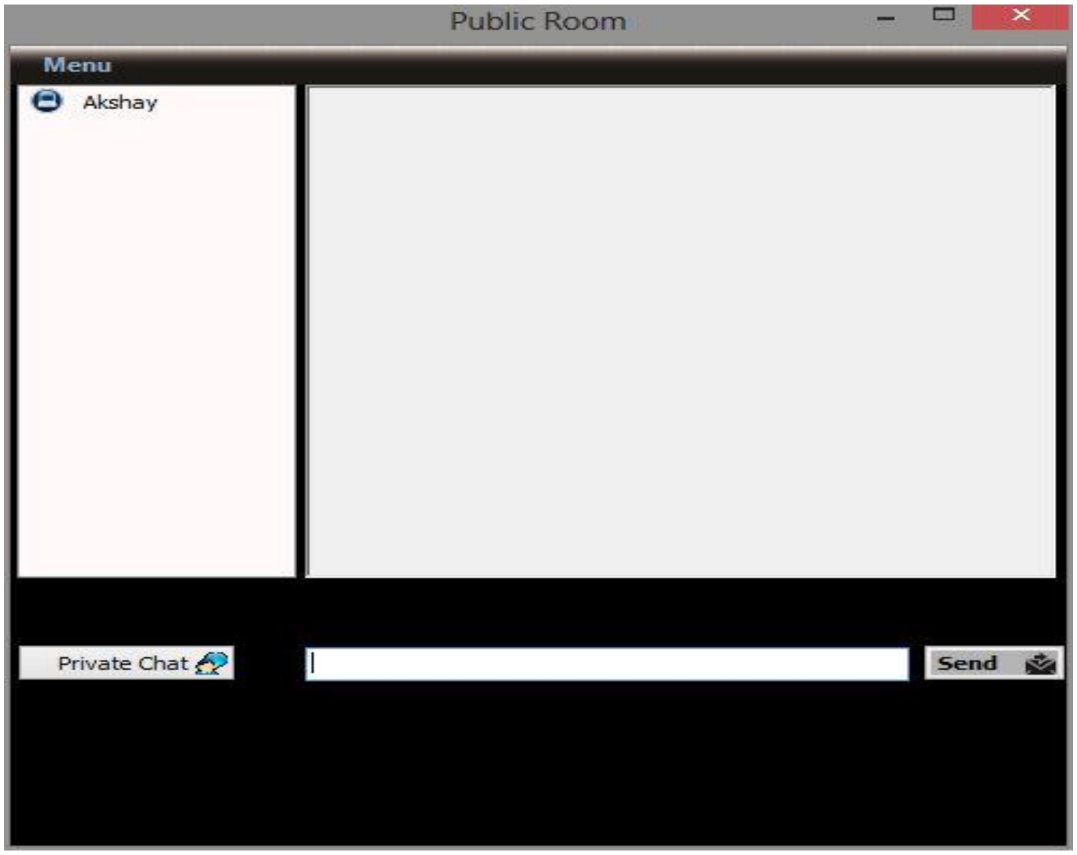


Fig 5.1 Chat Screen

The chat component enables the users to communicate with peers easily and more efficiently.

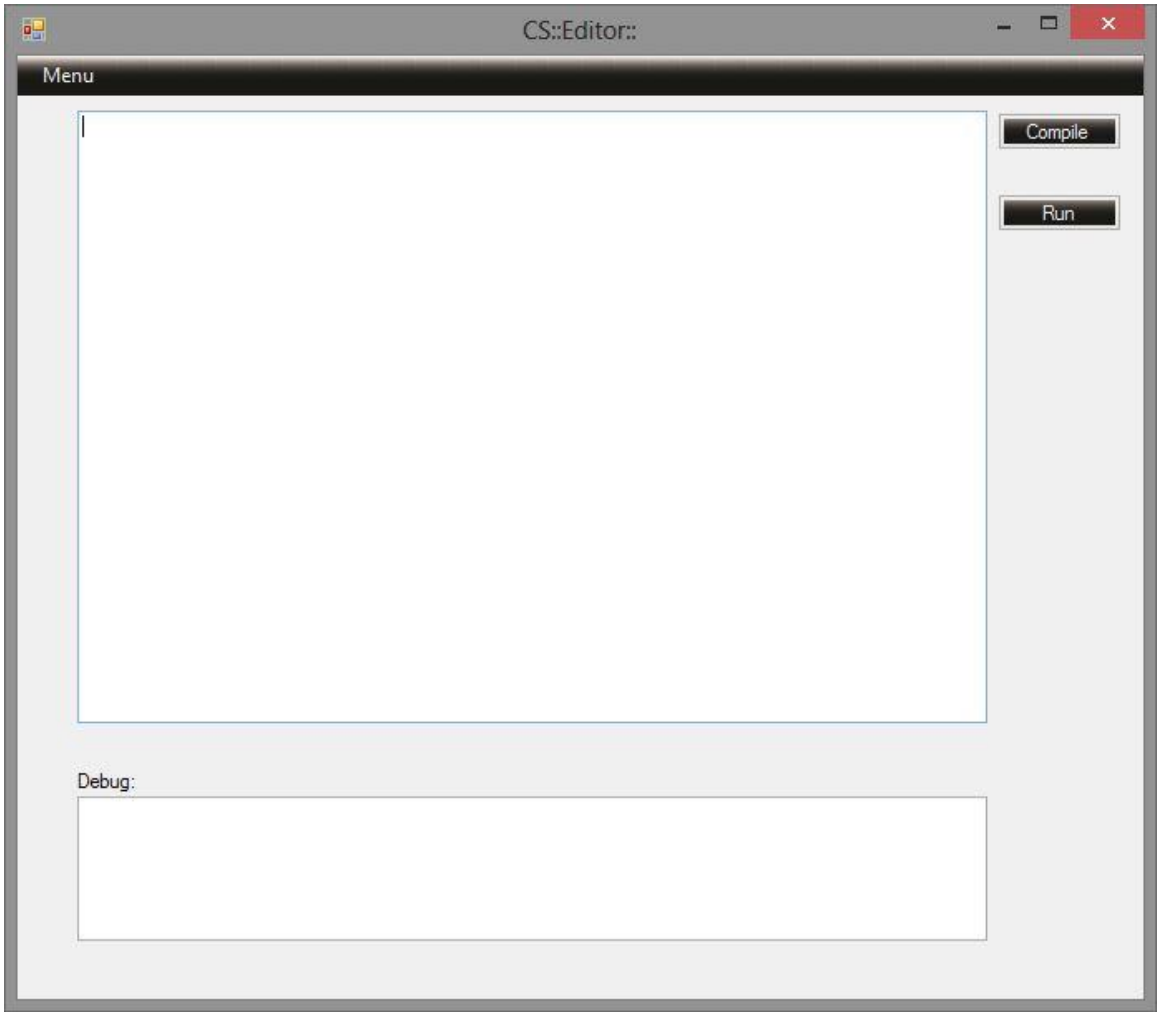
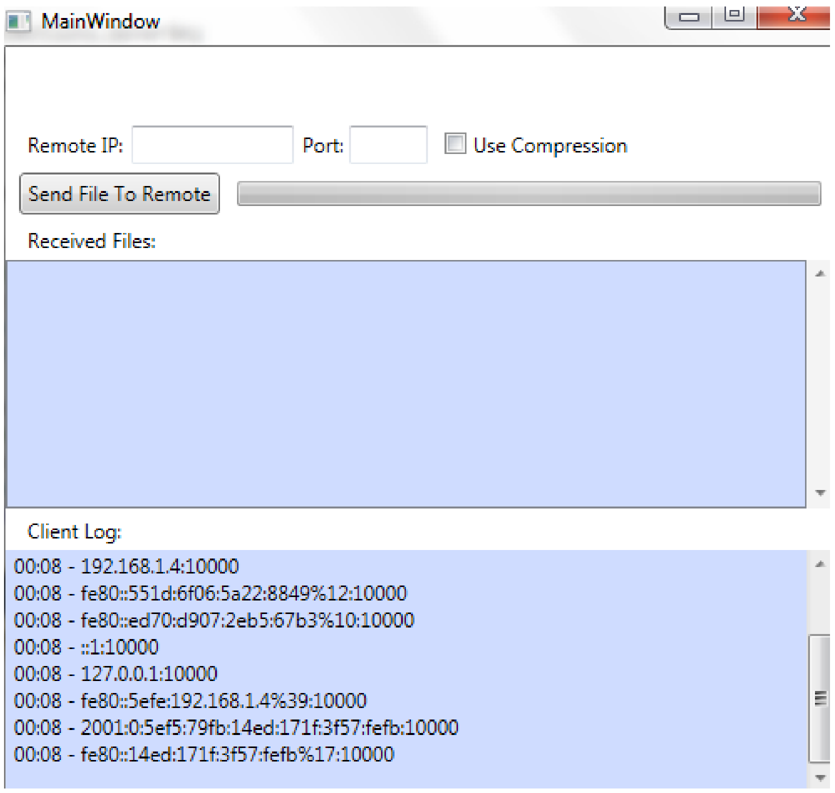


Fig 5.2 Editor Screen

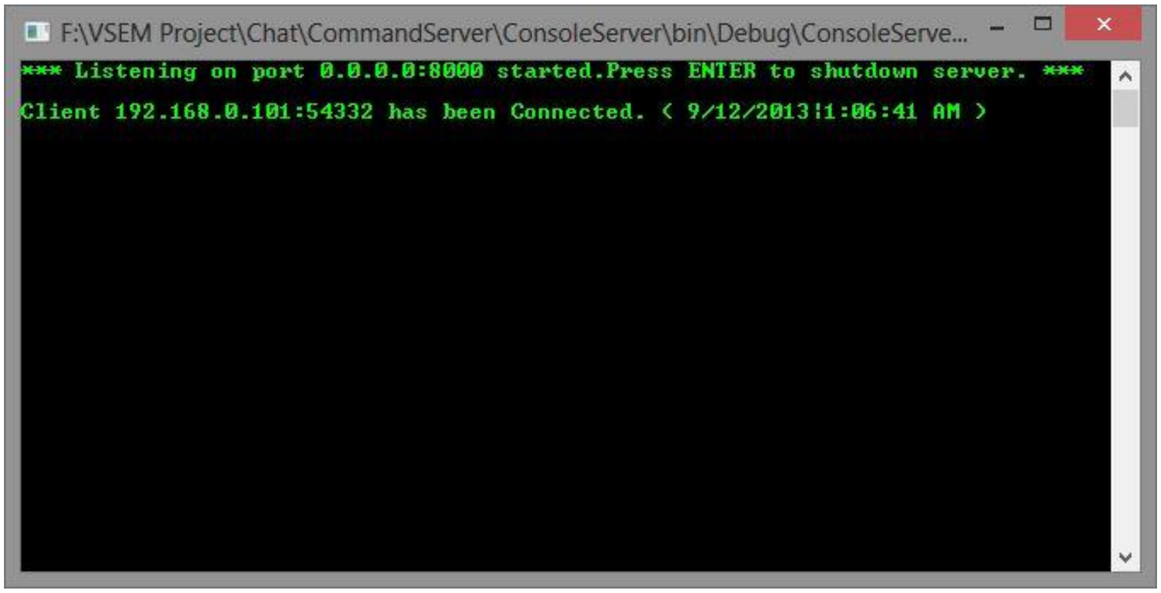
Editor is designed for programmers to write their source code and compile the program to get the desired output from the system.



```
00:08 - 192.168.1.4:10000
00:08 - fe80::551d:6f06:5a22:8849%12:10000
00:08 - fe80::ed70:d907:2eb5:67b3%10:10000
00:08 - ::1:10000
00:08 - 127.0.0.1:10000
00:08 - fe80::5efe:192.168.1.4%39:10000
00:08 - 2001:0:5ef5:79fb:14ed:171f:3f57:fefb:10000
00:08 - fe80::14ed:171f:3f57:fefb%17:10000
```

Fig 5.3 File Transfer Screen

The user can share files with peers by specifying their IP address and port number.


F:\VSEM Project\Chat\CommandServer\ConsoleServer\bin\Debug\ConsoleServe...

```
*** Listening on port 0.0.0.0:8000 started.Press ENTER to shutdown server. ***
Client 192.168.0.101:54332 has been Connected. ( 9/12/2013 1:06:41 AM )
```

Fig 5.4 Server Screen

Starts listener and displays the clients who are currently connected to the network.

6. TESTING

6.1 TEST PLAN

Software testing can be stated as the process of validating and verifying that a computer program/application/product:

- Meets the requirements that guided its design and development,
- Works as expected,
- Can be implemented with the same characteristics,
- Satisfies the needs of stakeholders.

Testing methods used in the development of this project are:-

1) Black-Box Testing
2) White-Box Testing

BLACK-BOX TESTING

Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings (white-box testing). This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well.

WHITE-BOX TESTING

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

6.2 TEST CASES

Table 6.1 Test Cases

Test Case	**Project Response**
Successful login credentials	User is granted access to the dashboard
Different file modes user can handle in the editor	User can open, close, modify and save source code successfully
Editor Features	Editor is enabled with bug highlighting, syntax highlighting and error indication.
Successfully capture screen	User can select the area to be captured and save the image in .png, .jpeg or .bmp format.
Open and Transfer files from drive	User can successfully open and share files from their memory to other users.
Successful transfer of file to the correct recipients	The files are sent to the recipients using their IP address.
Compilation of the source code	The source code is successfully compiled using gcc compiler
Send message in chat	User can send a text message to
Receive message in chat	User can receive messages from others

6.3 TEST REPORT

EVENT: Chat User Login

PROJECT RESPONSE:

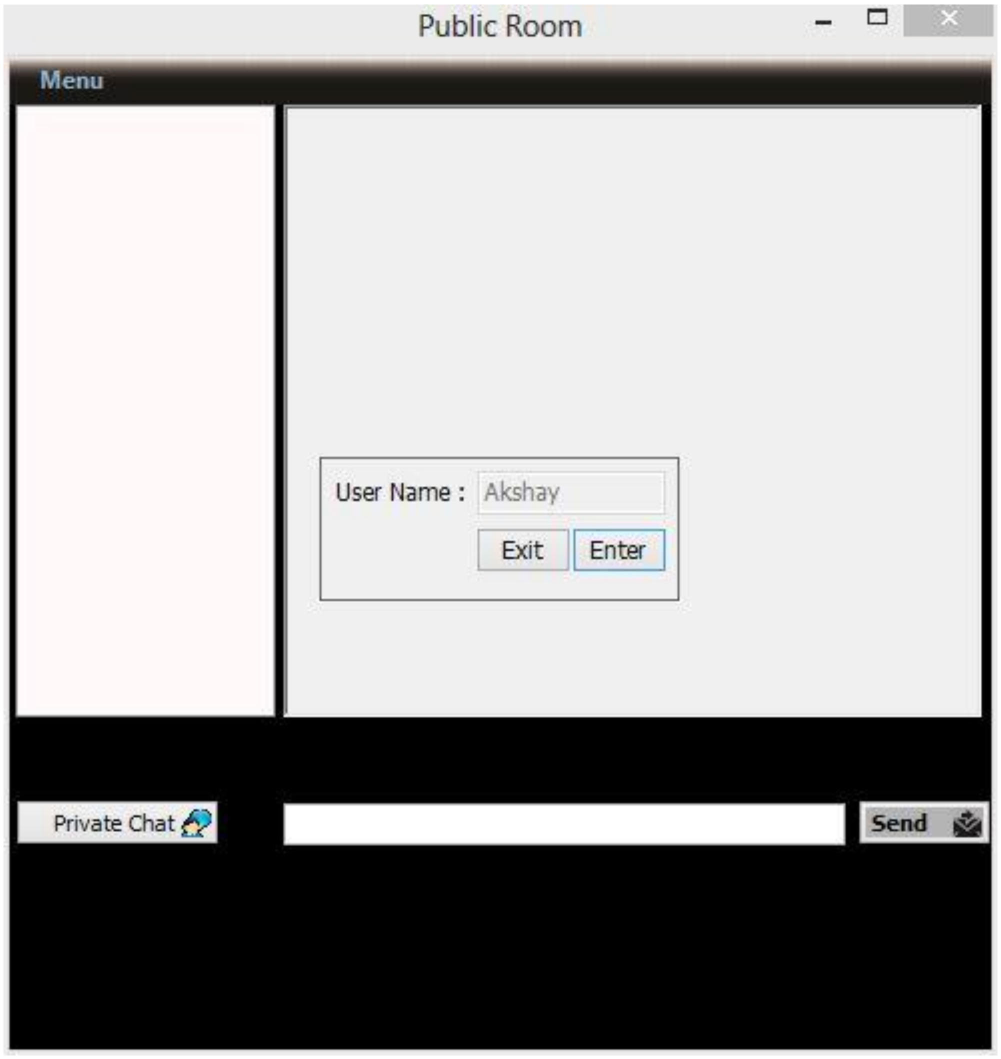
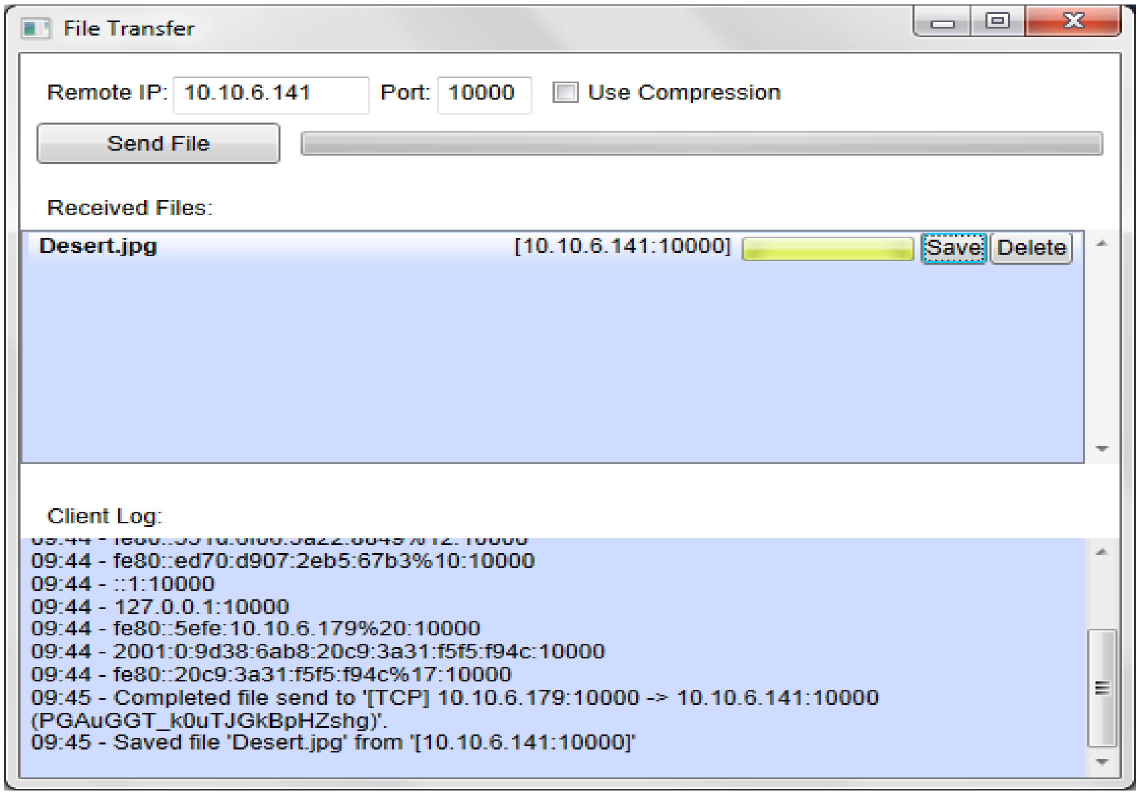


Fig 6.1 Chat User Login Screen

The user is able to login if the server is up and running.

EVENT: File Transfer

PROJECT RESPONSE:



```
09:44 - fe80::ed70:d907:2eb5:67b3%10:10000
09:44 - ::1:10000
09:44 - 127.0.0.1:10000
09:44 - fe80::5efe:10.10.6.179%20:10000
09:44 - 2001:0:9d38:6ab8:20c9:3a31:f5f5:f94c:10000
09:44 - fe80::20c9:3a31:f5f5:f94c%17:10000
09:45 - Completed file send to '[TCP] 10.10.6.179:10000 -> 10.10.6.141:10000
(PGAuGGT_k0uTJGkBpHZshg)'.
09:45 - Saved file 'Desert.jpg' from '[10.10.6.141:10000]'
```

Fig 6.2 File Transfer

The user can send and receive files once the connection has been established

7. CONCLUSION

Technology is an ever-growing field that will never cease to amaze us. In our project we have implemented what we had learnt in the semesters gone by. As we did so we gained further knowledge into the field of technical coding as well as project management. All that is put down theoretically in ink is much more difficult to implement in real life. The management of time and division of work amongst the team members was truly challenging in itself. All in all we did the best that we possibly could by putting in everything into the project, to make into a success. The challenges faced and overcome during developing the project helped us grow as analysts as well as programmers. The executable project can be implemented by any business concern as a point of sale software. I do know that in time technology will outdate our project but what we learnt while developing it will never be forgotten.

7.1 ADVANTAGES

Code Sharer is software designed for programmers especially beginners who are new to coding. This software helps programmers resolve bugs in the source code by seeking help from peers through sharing of code snippets. Code Sharer (CS) provides an easy to use text-editor which provides an interactive Integrated Development Environment (IDE) to compile simple codes and detect errors with ease. With features like bug highlighting, syntax correction and a chat interface, Code Sharer (CS) is the idle solution to a programmer’s woes.

With the help of Code Sharer (CS), a teacher or a team member can demonstrate and share their programs with a group of other programmers for learning and understanding all at run time. Users can also share screenshots of their codes with others and communicate with each other using the chat interface.

7.2 LIMITATIONS

Unfortunately, our software has a few limitations which we were not able to fix due to our limited resources. The current editor has problems like no auto generation of code and bracket matching functionality available to the user. Also the editor is not cross-platform. The editor can work only in Windows operating system.

7.3 FUTURE ENHANCEMENT

- Improve the visibility of the GUI for the users
- Enable auto code generation functionality
- Create a file upload module for users to store their source codes in cloud with ease.

REFERENCES

[1] Programming WPF, 2nd Edition - O'Reilly Media.

[2] Windows Presentation Foundation Unleashed (WPF) by Adam Nathan.

[3]< www.wpftutorial.net/ >.

[4] <msdn.microsoft.com/en-us/library/ms752299.aspx> .

[5] *<http://www.codegain.com/codesnippets/csharp/windowsforms/uploading-files-to-ftp-using-c-sharp-net.aspx>*.

[6] <http://www.codeproject.com/>.

[7]< http://geekswithblogs.net/>.

[8]< http://www.networkcomms.net/>.