PROJECT WRITE-UP

ON

DEVELOPING AN IP-BASED CHAT APPLICATION

BY

MICHEAL TADESE

OCTOBER, 2012

ABSTRACT

Communication is the basis of Information acquisition, processing, gathering, exchange and distribution. Communication is the link between problems and solutions, the solution seeker and the solution provider.

Communication is practiced in every stage of life be it childhood, adolescence or adulthood. Every aspect of life like education, religion or socialization entails communication. Communication is also mainly practice in professions such as Medicine, Media, Law, Politics, Engineering and Science related professions amongst others.

The need for effective and efficient communication solutions is therefore very essential in our society. This need however created the idea of developing a communication application such as an IP-Based Chat Application.

The IP-Based Chat Application is designed to enable the exchange of textual data and information amongst users via IP (Internet Protocol) addressed computers within a local network. It is a Peer-to-Peer based Chat application which supports real-time communication between two PCs at a time using the IP Address of both computers as the communication channel.

Once the program is loaded on a certain PC, the user simply inputs the IP Address or the Host Name of the Recipient PC he/she wants to connect to, clicks the Connect button to activate the connection and once a connection confirmation is displayed on the screen, both users can start chatting.

TABLE OF CONTENTS

CHAPTERS		PAGE
ABSTRACT		ii
TABLE OF CONTENTS		iii
CHAPTER ONE: INTRODUCTION		**1**
1.1	INTRODUCTION	1
1.2	PROBLEM STATEMENT	2
1.3	OBJECTIVES AND AIMS OF RESEARCH	2
1.4	SIGNIFICANCE OF RESEARCH	3
1.5	SCOPE OF RESEARCH	3
1.6	METHODOLOGY	3
1.7	DEFINITION OF TERMS	4
CHAPTER TWO: LITERATURE REVIEW		**5**
2.1	REVIEW OF RELATED LITERATURE	5
2.2	REVIEW OF PAST PROJECTS	6
CHAPTER THREE: SYSTEM DESIGN AND ANALYSIS		**10**
3.1	**SYSTEM DESIGN**	10
3.2.1	OVERVIEW OF PROPOSED SYSTEM	10
3.2.2	BENEFITS OF THE PROPOSED SYSTEM	10
3.2.3	OUTPUT DESIGN	11
3.2.4	INPUT DESIGN	12
3.2.5	PROCESS DESIGN	14
CHAPTER FOUR: IMPLEMENTATION AND DOCUMENTATION		**15**
4.1	CHOICE OF PROGRAMMING LANGUAGE	15
4.2	SYSTEM TESTING	15

4.3 SYSTEM REQUIREMENTS 16
4.4 SYSTEM CONVERSION 16
4.5 SYSTEM DOCUMENTATION 17

CHAPTER FIVE: SUMMARY AND CONCLUSION 19
5.1 SUMMARY OF RESEARCH 19
5.2 PROBLEMS ENCOUNTERED 19
5.3 LIMITATIONS OF RESEARCH 19
5.4 AREAS OF FURTHER STUDY 20
5.5 CONCLUSION 20
5.6 RECOMMENDATIONS 20

REFERENCES 21
APPENDICES 22
APPENDIX 1- SAMPLE REPORTS (OR OUTPUT) 23
APPENDIX 2- PROGRAM SOURCE CODES 25

CHAPTER ONE: INTRODUCTION

1.8 INTRODUCTION

Chatting is a method of using technology to bring people and ideas "together" despite any barriers in-between. The technology has been available for years but the acceptance was quite recent.

A Chat Application is a system designed for a group of people to communicate with each other over a network (LAN or WAN). It comes as either a Server/Client Chat Application or a Peer-to-Peer Chat Application.

A server/client chat application is made up of two sub-applications, a server application which runs on a networked PC and a client application which runs on all other users' PCs within the same network. This technology enables a user-to-user communication and also a broadcast communication. The server version monitors the entire chatting sessions amongst all users and the client applications depends on the continuous operation of the server application for users chatting sessions to be to be active.

The Peer-to-Peer application however is made up of a single application which is installed across all networked PCs. This usually supports one-on-one communication and does not depend on the operation of the server application to enable users chatting session. This also adds the option of confidentiality between the two clients as the server cannot have access to the information being exchanged.

An IP-Based Chat Application however is a specialized network system designed for a specific group of people within a certain network to communicate with each other via IP addresses of the networked computers. It solves the limitations of conventional systems which is basically 'Communication'. It could be server/client or peer-to-peer based.

OTAD-IPChat Application is a program which will provide users with the interface to chat with other users by typing the IP address of the destination PC into the program. Once the IP address of the other PC is entered and the connection is verified, the user can start sending messages to the connected user. The connected user will see the sender's name (with IP Address) and message on

his/her system provided the chat program is running on that system. The other user can also send a feedback by entering the IP address of the sender's PC into the Address Field.

The software will use a Peer-to-Peer structure which means users can interact with each other without server system's intervention and communication is one-on-one based.

1.9 PROBLEM STATEMENT

Most conventional systems lack the capacity to communicate with each other. Even with the introduction of network facilities like hubs, switches and routers, locally-networked computers still have challenges passing information from one to the other without certain configurations routines performed on each computer such as setting it into a certain workgroup or domain, sharing a folder on each computer so as to pass information to others and so on.

In other cases, an internet facility has to be setup for more flexible communication options like electronic mails.

1.10 OBJECTIVES AND AIMS OF RESEARCH

The objective of this research is to design an IP-based chat application to be used within a locally-networked environment.

The research's aim is to bypass the need for technical configurations on individual computers for PC-to-PC communication to be achieved. Also there will be no need for in internet facility to enable users communicate within the networked environment.

With OTAD-IPChat Application running on each system, users can communicate easily with each other. All the user requires is the IP address detail of the PC to connect to which will be displayed on each computer once the program is started.

All networked computer users within a college department, library or a company will equally benefit from the system called "OTAD-IPChat" application for chatting purposes.

This will also be beneficial in the promotion of silence within libraries and study centers as users' communication will be more textual and less verbal to avoid disturbances in such locations.

1.4 SIGNIFICANCE OF RESEARCH

The significance of this research is the provision of alternative communication functionality in a Local Networked Environment compared to the regular Inter-Communication Phone Systems and face-to-face verbal communication. Its design will reduce the communication gaps between different users within a network.

1.5 SCOPE OF RESEARCH

This research will be based on the Computer-Science Lab department of an educational institution.

It will focus on enabling interactive communication amongst students and other users in the lab without disrupting other users' activities (such as reading) in the lab.

1.6 RESEARCH METHODOLOGY

The methodology that will be adopted in carrying out this research will include the following:

i. Interview with students of the department regarding their present mode of communication within the lab
ii. Interview with officials in the lab about challenges with maintaining order and avoiding disruption in the lab in terms of students' communication amongst each other.
iii. Surfing the internet for details regarding past chat projects
iv. Reviewing past projects relating to this study

Details gathered from the above method will be used in designing the mode of operation of the system.

1.7 DEFINITION OF TERMS

- **IP (INTERNET PROTOCOL):** it is a group of numbers made up of 32 bits divided into 4 four bytes. It is an IP address that uniquely identifies any computer connected to a network
- **TCP (TRANSMISSION CONTROL PROTOCOL):** has the responsibility for breaking up the message into datagrams, reassembling them at the other end, resending anything that gets lost, and putting things back in the right order.
- **LAN (LOCAL AREA NETWORK):** A self-contained network that spans a small area such as a single building, floor or room. The connected devices can only communicate with each other and cannot connect to any system outside the network range.
- **WAN (WIDE AREA NETWORK):** A network that spans multiple geographic locations. WANs typically connect multiple LANs using long range transmission media.
- **SERVER:** This is a networked computer that shares resources with other networked computers and responds to requests from those computers as well as from other servers.
- **CLIENTS:** A client is a networked computer that utilizes the resources of other networked computers including other clients.
- **PEER-TO-PEER:** A peer is a self-sufficient computer that acts as both a server and a client to other computers on the network.
- **HOST NAME:** this is a computer-assigned name which is used to recognise it within a certain network.
- **PC (PERSONAL COMPUTER):** This is a term used for all computers with a desktop or tower system unit, a keyboard and a mouse. A PC is usually a client or peer-to-peer computer.

CHAPTER TWO: LITERATURE REVIEW

2.1 REVIEW OF RELATED LITERATURE

Chat Applications provide an alternative means of communication in addition to older means such as the telephone, fax, postal service and even the recent electronic mail technology.

The use of nonverbal elements in text-based virtual interactions provides participants with some of the richness of real-time, face-to-face interactions. Chatting via programs is simple, direct, and unrestrained. While it contains many of the elements of face-to-face conversation, it differs from it in that it is a textual representation of conversation.

A chat application is a kind of technology that supports human-to-human communication in real-time. Over the past decade, with the continuous advancement of technology, there has been an increasing trend in the use of chat applications around the world for communication.

Programs like Yahoo! Messenger and ICQ enables chatting, sending of messages and files or even playing of games amongst users.

According to Michael Hauben (1997), an Internet Pioneer and Author, users are free to communicate without fear, limit nor abuse of confidentiality. This he pointed out as the main reason for the increasing use of chat technologies.

Joseph Licklider (1968), an Associate Professor at the University of Waterloo in Canada also claimed that with chat technology, people can communicate with others who have similar goals and interests thus enriching their lives and making communication more productive and enjoyable.

Other writers like Neil Randall (1997) cited the problems relating to chatting technologies. He wrote that though the technology is used for socializing, users often do not use their real identities for communication which somehow is a form of deceit undetectable by the technology. He also stated that with the increase in chat usage, people will be discouraged from normal social contact and adopt the social network contacts instead.

2.2 REVIEW OF PAST PROJECTS

2.2.1 DEVELOPING A CHAT SERVER

This project was developed by a student (**Waleed Farah)** of the Department of computer Science, Faculty of Natural and Applied Sciences from Notre Dame University, Indiana USA in September 2000.

The program was developed using Visual Basic. Visual Basic 6 was Microsoft's latest and greatest language for Visual programming at that time. It supports the use of a control called windows socket or WinSock.

This chatting system dealt only with LAN's (static IP address) and was made up of two applications, one runs on the server side (any computer on the network chosen to be the server) while the other was delivered and executed on the client PC.

Every time the client wants to chat, he/she runs the client application, enter user name, host name where the server application is running, and hits the connect button and start chatting. The system is many-to-many arrangement; every–one is able to "talk" to anyone else. Messages may be broadcasted to all receivers (recipients are automatically notified of incoming messages) or sent to special individuals (private chatting through server) where during this operation all messages are encrypted at the sender side and decrypted at the recipient to disallow any hackers to the server from reading these private messages.

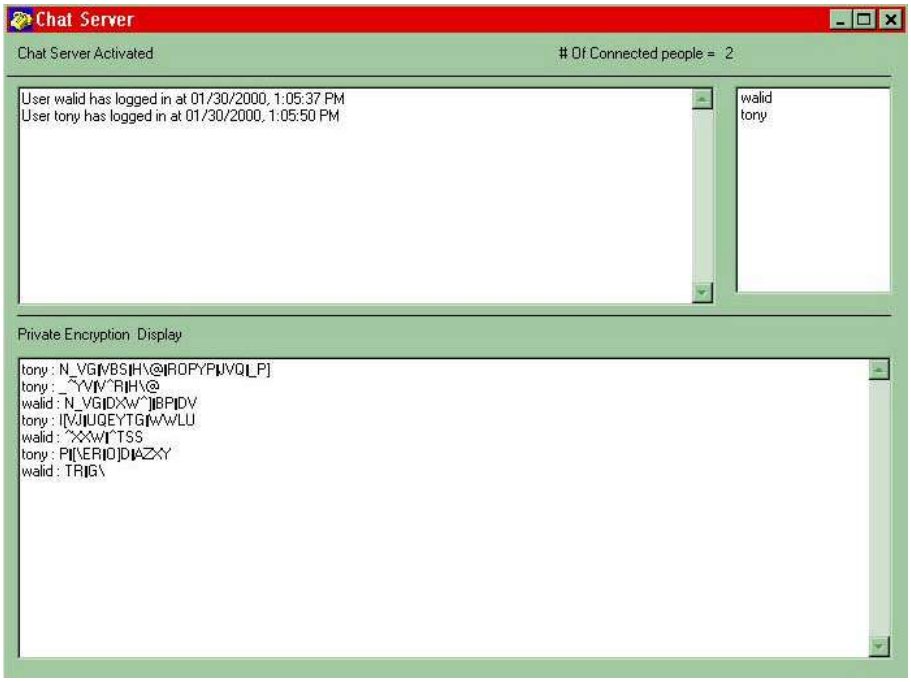
A TCP/IP technology was adopted for designing this program. TCP enables the user to create and maintain a connection to a remote computer. By using the connection, both computers can stream data between each other. An IP address uniquely identifies any computer connected to a network.

The mode of operation of the server version is listed below:

- Create a socket
- Listen for incoming connections from clients
- Accept the client connection
- Send and receive information
- Close the socket when finished, terminating the conversation

The mode of operation of the Client Version is:

- Create a socket
- Specify the address and service port of the server program
- Establish the connection with the server
- Send and receive information
- Close the socket when finished, terminating the conversation



SERVER FORM

CLIENT FORM

2.2.2 PERLCHAT – A CGI APPLICATION FOR HTML BASED CHAT

PerlChat is a web application, which utilizes the functionality of Common Gateway Interface (CGI) to provide users with a fast and easy-to-use chat based interaction facility.

It was designed in 2002 by the final year students of Computer Science and Engineering in R.V. (Rashtriya Vidyalaya) College of Engineering, Bangalore India.

PerlChat uses HTML based interaction sequences to provide its users with chat rooms which they can subscribe to, and after proper authentication, can chat with other users currently in the channel via instantly delivered messages.

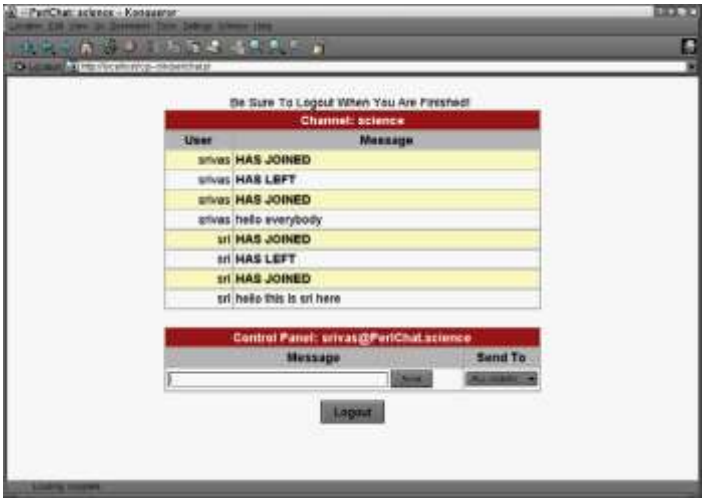
It entails a World Wide Web (WWW) programming which deals with the development of hypertext document interaction mechanisms, which provide the client with a rich and intuitive interface to the information that he or she desires to view.

The WWW works due to the process by which web servers generate and provide HTML encoded information, as and when requested by client browsers. CGI is the part of the Web server that can communicate with other programs running on the server. With CGI, the Web server can call up a program, while passing user-specific data to the program such as what host the user is connecting

from, or input the user has supplied using HTML form syntax etc. The program then processes that data and the server passes the program's response back to the Web browser. CGI turns the Web from a simple collection of static hypermedia documents into a whole new interactive medium, in which users can ask questions and run applications.

PerlChat provides HTML based chat room functionality to its users. It implements the following functionality.

- Users access the PerlChat facility by first signing up for the service. Once users have registered themselves, they are assigned usernames and passwords.
- PerlChat provides its users with a large collection of chat rooms on a variety of themes, and the user can add one or more chat rooms to his or her personal favourites.
- Using the assigned password, the user can access a collection of his or her favourite chat rooms.
- Once a user enters a chat room he can interact with other users currently in the chat room, via instant messages.
- PerlChat keeps track of all the users currently logged into the service. It ensures password secrecy, so that other users cannot access a user's password or profile information.



PERLCHAT MAIN SCREEN

PerlChat was implemented using Perl, short for **Practical Extraction and Reporting Language**. It uses the facilities of a CGI enabled web server to generate dynamic content, which is viewed by the user. Also, it provides for user authentication, and maintains information about its users in a database. This allows for customized content to be delivered to the user.

CHAPTER THREE: SYSTEM DESIGN AND ANALYSIS

3.1 SYSTEM DESIGN

3.1.1 OVERVIEW OF PROPOSED SYSTEM

The proposed chatting program called OTAD-IPChat Application will be based on a Local Area Network structure. It will be designed using a Peer-to-Peer protocol which means any user can instantly communicate with other users once the program is loaded on each PC. It will not depend on the existence operation of a server system to enable real-time communication.

Once the program is loaded, the user will be prompted for his/her name (User-1). Afterwards, the Chat window will be displayed which will automatically display the PC's Hostname and IP address. To connect to another user (User-2), User-1 will input the destination ("Buddy") PC's IP address or Hostname (provided both PCs are within the same LAN) and click the connect button. The destination PC (Buddy) will instantly get a notification about the User-1's connection to User-2's PC. The system will henceforth enable both User-1 and User-2 chat in a non-verbal (text-based) mode until either user log out of the connection.

Users will also have the option to save their chat discussions before closing the program. Furthermore, the option to view previously saved chat discussions will be implemented into the system.

3.1.2 BENEFITS OF THE PROPOSED SYSTEM

The following benefits will be achieved from using the system:

- Instant exchange of data and information amongst users within the laboratory
- Reduction in disruption or disturbance within the lab caused by verbal communication.
- Communication amongst all PCs in the lab without the need for configurations of file sharing and permissions on individual systems.

3.1.3 OUTPUT DESIGN

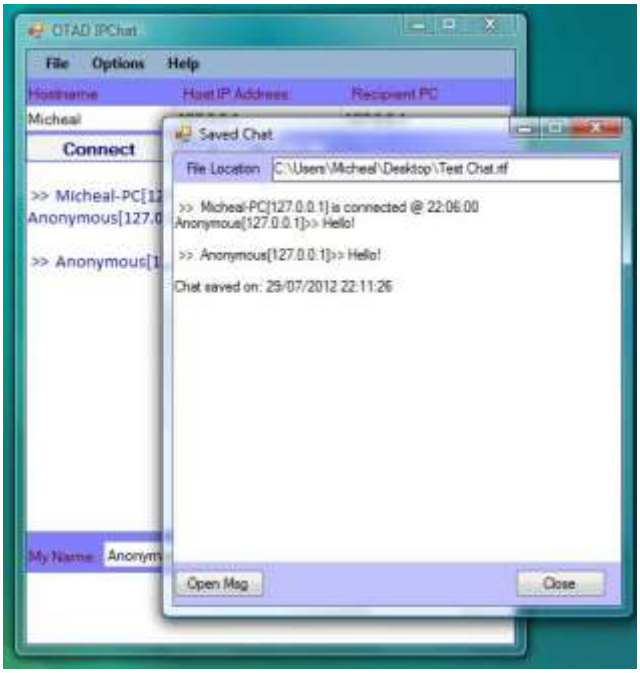
This is the display of chat discussions between connected users on the chat program. The discussions are displayed in Rich-Text, Read-Only format on a text box similar to the Windows Notepad but with a coloured font which makes it more user-interactive.

For every message typed and sent, a copy is instantly displayed on the destination PC's chat screen as well as the source PC's chat screen. A timer at the top-right side of the chat box is also continuously displaying the time each last message was received from both users.

In addition to the chat discussion that can be displayed on the chat box, users can as well open previously-saved chat discussions as a Text, Rich-Text or Document format. The saved chat discussion will display at the bottom of the discussion the date and time the discussion was saved initially.

Output Chat Screen

Saved Chat Screen



Saved Chat Screen

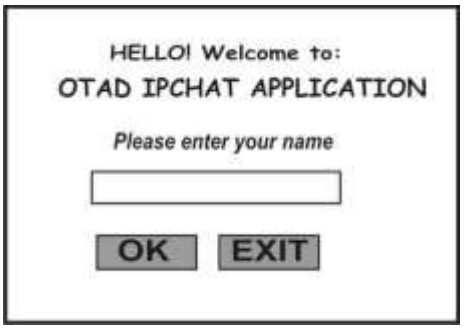
3.1.4 INPUT DESIGN

Once the program is started, the user is first prompted with a Start-up screen to enter his/her name into a textbox which will be used for identification when connected to other users. Afterwards, the program is opened displaying the User's name (identification), the Hostname and the Host IP Address of the PC.

On the Chat program, the user (User-1) is now expected to type into the required field, the IP Address or Hostname of the PC he/she wants to connect to. Once the 'CONNECT' button is clicked, the name of User-1 and the PC's IP address is displayed on the destination PC notifying the other user (User-2) that someone has connected to his/her PC. The user (User-2) will also need to type the Hostname or IP address of User-1's PC into the required field and clicks connect for the connection to be successful on both sides.

Both users can now start chatting by typing into the message box at the bottom of the chat application's interface and clicking the 'SEND' button.

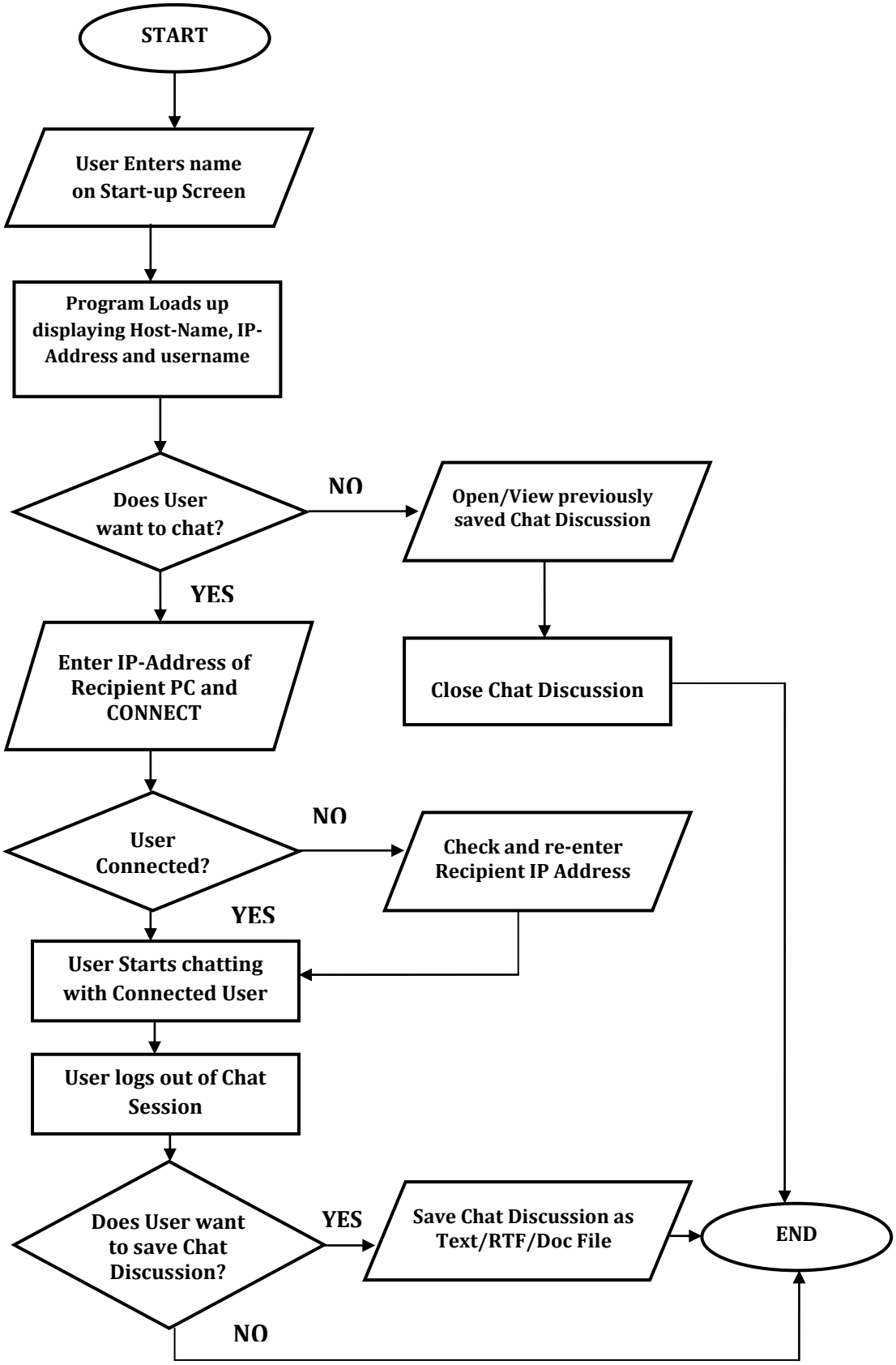
Furthermore, the users will have the option to save their chat discussion as a text or rich-text file into the system after logging out of the chat session.



Chat Startup Screen

Input Chat Screen

3.1.5 PROCESS DESIGN



CHAPTER FOUR: IMPLEMENTATION AND DOCUMENTATION

4.1 CHOICE OF PROGRAMMING LANGUAGE

The programming language used in designing this application is VISUAL BASIC 9. The tools used in developing the application are from Visual Studio 2008 Express Edition and the environment/framework used to design the application is the .NET Framework Version 3.5 Service Pack1.

The purpose of choosing this programming language is based on the wide range of graphical tools and features in Visual Basic for designing user-friendly applications such as the Textboxes, Labels, Command buttons, Timer Controls, Menu items, scrollbars and so on.

4.2 SYSTEM TESTING

Program testing discusses the testing procedures used in verifying the workability of the code and the entire system as desired.

These Testing procedures aim to ensure that the application performs its expected task which is the sending and receiving of textual data or information between PCs. The testing procedures will also ensure some validation rules are not violated while performing the expected task.

The special cases tested in this program include:

- Empty Text field (Username, Recipient IP Address, Chat Textbox)
- Invalid IP Address value (Recipient IP Address)
- Successful connection to a recipient's PC running OTAD IP-Chat Application
- Successful exchange of information between two networked PCs running OTAD IP-Chat Application
- Successful saving and re-opening of a (saved) chat discussion
- Logging-out and logging-in to deactivate and reactivate the Chat Application respectively

4.3 SYSTEM REQUIREMENTS

The hardware and software requirements listed below have been tested to run the program successfully.

The minimum, recommended hardware facilities required for the successful implementation and operation of this program are listed below:

- Pentium III 1.6Ghz Motherboard
- 20Gigabytes Hard Disk Drive
- 256Megabyte RAM Size
- 10/100MB Fast Ethernet Adapter (Network Interface Card, wired or wireless)
- 14 Inches SVGA Monitor
- Keyboard and Mouse

The minimum, recommended software requirements required for the successful implementation and operation of this program are listed below:

- Windows XP Professional SP2 Operating System or higher
- Microsoft Dot NET Framework 2.0 and 3.5 SP1

4.4 SYSTEM CONVERSION

The system conversion methodology that will be suitable for this project will be the **PARALLEL CONVERSION METHOD**.

PARALLEL CONVERSION

This involves keeping the old system running alongside the new system for the first couple of weeks or months after the introduction of the new system.

The Parallel Method activities include the following:

- Practicing the present method of communication (verbal) while the new system is being implemented and introduced to the Laboratory Computer users
- Adopting and getting familiar with the new method of communication (non-verbal or text mode) while practicing the verbal method at a minimal level
- Full acceptance and usage of the new system for communicating within the Computer Laboratory

4.5 SYSTEM DOCUMENTATION

System Documentation is the section of a project write-up which lists out all available modules in an application and describes the functions of these modules within the application.

FUNCTIONS OF APPLICATION MODULES

This section describes the activities of each module that makes up the Otad IP-Chat Application.

The modules entailed are:

- Start-up Screen Module
- Chat Screen Module
- Saved Message Module
- Help Topics Module
- Exit Module

START-UP SCREEN MODULE

This module acts as a gateway to other modules in the program. It is used to log a user into the system by displaying the user's name in the program interface. If a valid name is entered, it will load the next module. Otherwise, it will bring up an error message saying "Please input your Name"

CHAT SCREEN MODULE

This module performs the main activities of the application. Once the Start-up screen accepts the user's name entered, the Chat Screen is loaded and the various functions that are performed within this include:

- Automatically displays the host Computer Name and IP Address
- Displays the Name entered on the Start-up Screen
- Enables the user connect to another computer by entering the computer's Name and IP Address then clicking the CONNECT button
- Enables the user send text data to the connected recipient
- Displays text data received from the connected sender
- Logging in and out of chat sessions in the program
- Saving chat discussions as a Rich Text or Document file

SAVED MESSAGE MODULE

This module enables the user open chat discussions that have been saved previously on the system using the program.

It can process files in any of these file formats:

- Text File (*.txt)
- Rich Text File (*.rtf)
- Word Document File (*.doc)

HELP TOPICS MODULE

This module provides guidelines on how to perform expected tasks within the program. These tasks include the following:

- Connecting to other users
- Sending a message to a connected user
- Chatting with a connected user
- Logging out of a chat session
- Re-logging in to the chat environment
- Saving a chat discussion
- Open a saved chat discussion

EXIT MODULE

This module is simply used to terminate OTAD IP-Chat application.

CHAPTER FIVE: SUMMARY AND CONCLUSION

5.1 SUMMARY OF RESEARCH

The main goal of communication is to pass information from the source to the destination and vice-versa or to exchange ideas amongst individuals or group of people. This project makes use of intranet technology to automate the exchange of textual information between two or more individuals.

With the implementation of this project in the Laboratory Unit of the educational institution, an effective alternative to verbal communication will be achieved. Also, the Networked PCs in the laboratory will be more utilized as real-time communication will be enabled and operational on them.

Users will be able to communicate with other users using the networked computers in the lab. They would be able to share information among themselves without disrupting the activities of other users such as reading an e-book, designing or testing a program, creating or editing a document and so on. They would also have the option to save their chat discussions on the system for reference purposes

5.2 LIMITATIONS OF RESEARCH

Some expectations could not be met in this project due to unforeseen circumstances such as short time-frame for learning the programming language used for the project design as well as for completing the implementation tasks. These unmet expectations however created limitations on the application's features and operation. Some of these limitations include:

- Inability to add a Multi-Chat feature to the application due to the cumbersome amount of code involved in achieving this and the limited time-frame to complete the project

  A Multi-Chat Feature enables a user to chat with multiple users simultaneously from the same application

- Absence of Emoticons and smiley icons to make the chatting experience more visually-interactive
- Absence of Audio or Sound Alerts for new connections or incoming messages while the chat application is active

5.4 AREAS OF FURTHER STUDY

Further development to this project may involve the following:

- Creating and saving login and logout sessions of each user on a particular system.
- Enabling a connected user to chat with multiple users simultaneously from his/her screen.
- Addition of Smiley and emoticons to the chatting section of the application.
- Addition of Sound Alerts for new connections and incoming messages.

5.5 CONCLUSION

This Intranet based OTAD IP-Chat Application has been successfully designed and implemented on different Windows-Based Systems (XP, Vista, Win 7) with the .NET Framework Technologies (2.0 and 3.5) installed. The application's capabilities have also been fully tested on these Windows environments.

The application is designed to work within library PCs, college departments (locally networked), privately-networked companies and the likes.

5.6 RECOMMENDATIONS

With all the benefits achieved from this new system as listed in previous chapters, it would be even more beneficial and advantageous to maintain the operation and functionality of the project after implementation and project sign-off.

For the project to be sustainable after implementation, the following are recommended:

- Once a Local Area Network has been setup within the Computer Laboratory, continuous maintenance and effective administration of this network should be practiced.
- Regular diagnostics and maintenance of the Laboratory infrastructures should also be practiced.
- Frequent awareness of the implemented solution and its features should be made to every new and visiting scholar in the institution. This awareness may be in form of Notification Fliers around the lab promoting real-time PC communication over verbal communications within.

REFERENCES

Chennu S. N. (2002). *PERLCHAT – A CGI Application for HTML Based Chat.*
Department of Computer Science and Engineering in R.V. (Rashtriya Vidyalaya) College of Engineering, Bangalore India

Farah W. (2000). *Developing A Chat Server*
Department of Computer Science, Notre-Dame University, Indiana USA

Gajadhar J. and Green J. (2012). *Introduction to IP-Based Chat Application*.
http://www.educause.edu/importance-nonverbal-elements-online-chat.htm

Hauben M. (1997). *Review of Related Literature on IP-Based Chat Application [1]*.
The Net And Netizens: The Impact The Net Has on People's Lives, Chapter 1,
www.columbia.edu/~rh120/ch106.x01

Licklider J. and Taylor R. (1968). *Review of Related Literature on IP-Based Chat Application [2]*.
The Computer as a Communication Device in Science and Technology

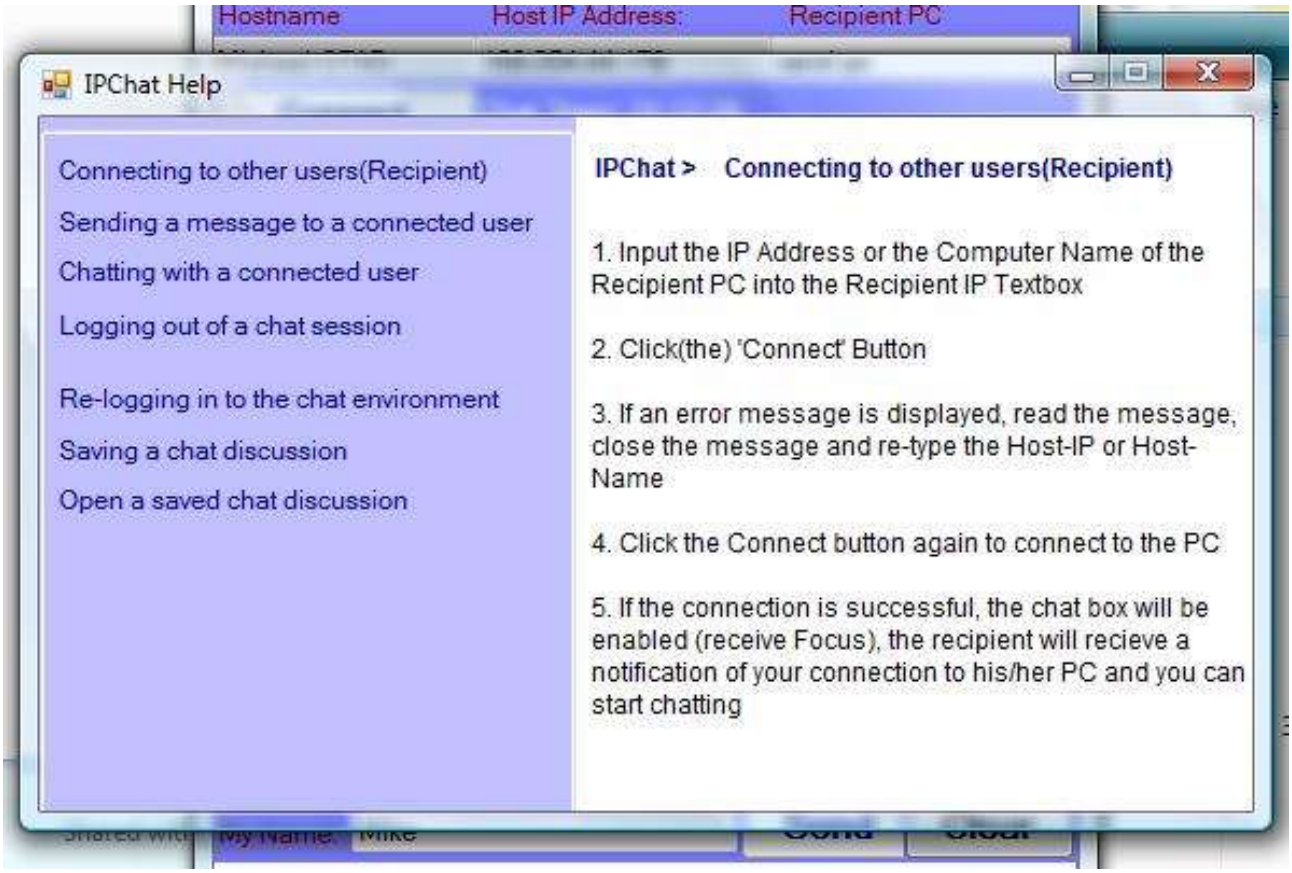
Neuage T. (2000). *Review of Related Literature on IP-Based Chat Application.*
http://www.angelfire.com/on/hypertextual/ethics.html

Petroutsos E. & Manfield R. (2004). *Implementation and Documentation: Choice of Programming Language.*
Visual Basic.NET Power Tools; Peer-to-Peer Programming, Chapter 12

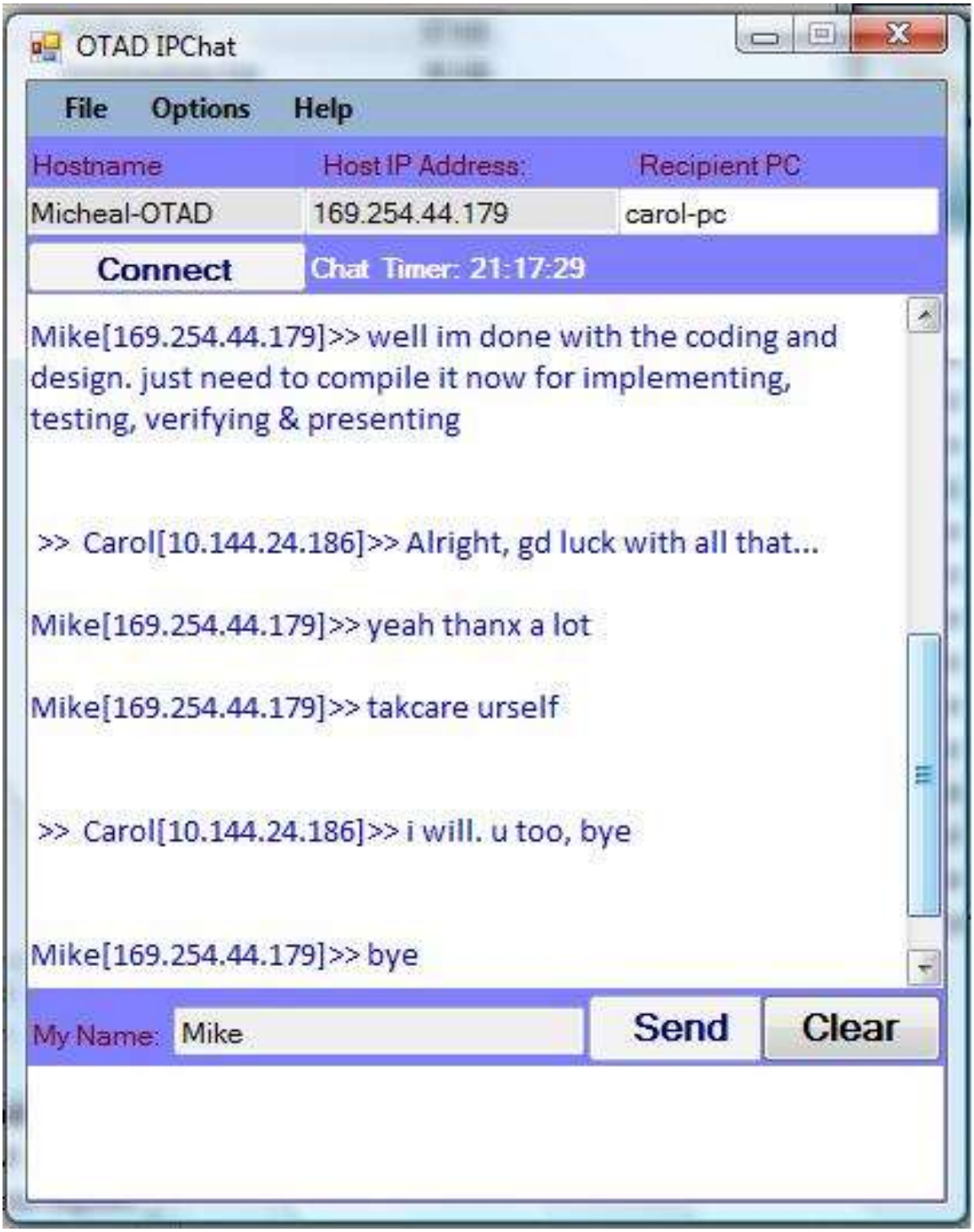
Randall N. (1997). *Review of Related Literature on IP-Based Chat Application [3]*.
Future Communities: Socializing And Educating in the Internet's Future, Chapter 19,
http://www4.caes.hku.hk/acadgrammar/litrev/examples/ICQ1-Impact.htm

APPENDICES

APPENDIX 1
SAMPLE REPORTS (OUTPUTS)



IPCHAT HELP SCREEN



IPCHAT MAIN SCREEN

APPENDIX 2
PROGRAM SOURCE CODES

- **Start-up Form**

```
Public Class frmstartup

    'Display Startup Name in IPChat Username Field
    Public Shared username As TextBox

    Private Sub frmstartup_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        ' txtname.Focus()
        username = txtname

    End Sub

    Private Sub btnok_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnok.Click
        'validating username field
        If txtname.Text = "" Then
            MsgBox("Please input your Name")
            txtname.Focus()

        Else
            'display username in the Chat Application; username field
            frmIPChat.txtusername.Text = username.Text

            'closing the startup screen and loading the chat application
            Me.Close()
            frmIPChat.Opacity = 100%
        End If
    End Sub

    Private Sub btnexit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnexit.Click
        'exit the entire application
        End

    End Sub

End Class
```

- **OTAD IP-Chat Main Form**

```
' defining vb methods

Imports System.Net.Sockets
Imports System.Threading
```

```
Imports System.IO
Imports System.Net
Imports System.Text
Imports System.Net.Dns


Public Class frmIPChat
    ' variable declarations

    Dim listener As New TcpListener(55555)
    Dim client As TcpClient
    Dim MySockect As Socket
    Dim message As String = ""
    Dim serveriplist As IPHostEntry =
System.Net.Dns.GetHostEntry(System.Net.Dns.GetHostName())
    Dim machineip As String
    Dim nameip As String
    Dim y As String = System.Net.Dns.GetHostName


    Private Sub frmIPChat_FormClosing(ByVal sender As Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles Me.FormClosing
        ' terminating the chat application network listener
        listener.Stop()

    End Sub

    Private Sub frmIPChat_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

        On Error Resume Next

        frmstartup.Show()
        ' display the PC's IPV4 address (xp, vista, win7...)

        machineip = serveriplist.AddressList(0).ToString()
        machineip = serveriplist.AddressList(1).ToString()
        machineip = serveriplist.AddressList(2).ToString()

        ' Display the system ip address
        txtuserip.Text = machineip

        'set the Listening port to start listening for connection
        'Dim listthread As New Thread(New ThreadStart(AddressOf listening))
        'listthread.Start()

        txthostname.Text = y

    End Sub
```

```
    Private Sub listening()
        ' Listening port subroutine
        listener.Start()

    End Sub

    Private Sub btnsend_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnsend.Click

        Try

            'Connect to the Client's PC with its IP Address and Port details

            client = New TcpClient(txtip.Text, 55555)
            Dim writer As New StreamWriter(client.GetStream())

            'Save the message the message unto the client's listening socket buffer

            writer.Write(txtusername.Text & "[" & txtuserip.Text & "]" & ">>" & " " &
txtsend1.Text)
            writer.Flush()

            RichTextBox1.Text += (txtusername.Text & "[" & txtuserip.Text & "]" & ">>" & " "
& txtsend1.Text) + vbCrLf

            'clears the inputbox after sending the last message
            txtsend1.Clear()
            txtsend1.Focus()

        Catch exc As Exception
            MsgBox(exc.Message)
        End Try

    End Sub


    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
        'set the listening port to continue listening for connections
        If listener.Pending = True Then
            message = ""
            client = listener.AcceptTcpClient()

            Dim reader As New StreamReader(client.GetStream())
            While reader.Peek > -1
                message = message + Convert.ToChar(reader.Read()).ToString
            End While

```

```
            'display the text stored in the Listening Port's buffer
            RichTextBox1.Text = RichTextBox1.Text + Environment.NewLine + " >> " + " " +
message + vbCrLf


            'set the cursor and scroolbar to the end of the text
            RichTextBox1.SelectionStart = Len(RichTextBox1.Text)
            RichTextBox1.ScrollToCaret()

        Else : End If
    End Sub

    Private Sub Connecterror()
        ' Error detection and handling subroutine

        On Error GoTo 50

50:     MsgBox("Pls check the client IP address is valid")
        txtip.Clear()
        txtip.Focus()

    End Sub


    Private Sub btnclear_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnclear.Click
        txtsend1.Clear()
        txtsend1.Focus()

    End Sub


    Private Sub AboutIPChatAppToolStripMenuItem1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles AboutIPChatAppToolStripMenuItem1.Click
        AboutForm.Show()
        Me.Enabled = False

    End Sub


    Private Sub btnConnect_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnConnect.Click
        'Error detection and handling
        'validating recipient ip address field
        Try

            If txtip.Text = "" Then
                MsgBox("Please enter the Client IP Address")
                txtip.Focus()
```

```
            txtsend1.Enabled = False

        Else

            'set the Listening port to start listening for connection
            Dim listthread As New Thread(New ThreadStart(AddressOf listening))
            listthread.Start()

            Timer1.Enabled = True

            client = New TcpClient(txtip.Text, 55555)
            Dim writer As New StreamWriter(client.GetStream())


            writer.Write(txthostname.Text & "-PC" & "[" & txtuserip.Text & "]" & " is connected @ " & TimeOfDay)
            writer.Flush()

            txtsend1.Enabled = True

            btnConnect.Enabled = False

        End If
    Catch exc As Exception
        MsgBox(exc.Message)
    End Try
End Sub

Private Sub SaveToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SaveToolStripMenuItem.Click
    'save a chat discussion on the chat box

    Try

        SaveFD.InitialDirectory = "C:\"
        SaveFD.Title = "Save Chat to File"
        SaveFD.Filter = "RTF Files|*.rtf|Text Files|*.txt|Word Files|*.doc|Wordx Files|*.docx"

        SaveFD.ShowDialog()
        SaveFD.OverwritePrompt = True

        'SaveFileDialog1.ShowDialog()
        System.IO.File.WriteAllText(SaveFD.FileName, RichTextBox1.Text & vbCrLf & "Chat saved on: " & Date.Now)

    Catch exc As Exception
        MsgBox(exc.Message)
    End Try
```

```
    End Sub

    Private Sub RichTextBox1_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RichTextBox1.TextChanged
        ' display time logs of the most recent chat message received

        lbltimer.Text = "Chat Timer: " & TimeOfDay
        'lbltimer.Text = "Chat Timer" & Date.Now

        'set the cursor and scroolbar to the end of the text
        RichTextBox1.SelectionStart = Len(RichTextBox1.Text)
        RichTextBox1.ScrollToCaret()

    End Sub

    Private Sub ExitToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ExitToolStripMenuItem.Click
        End
    End Sub

    Private Sub txtip_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtip.TextChanged
        btnConnect.Enabled = True
    End Sub

    Private Sub txtsend1_KeyPress1(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles txtsend1.KeyPress
        ' enabling the Enter key to auto-send chat message
        If e.KeyChar = ChrW(Keys.Enter) Then
            Try

                btnsend.Focus()

            Catch exc As Exception
                MsgBox(exc.Message)
            End Try

        End If

    End Sub

    Private Sub txtsend1_TextChanged_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtsend1.TextChanged
        ' sending a msg using the send button

        btnsend.Enabled = True
        If txtsend1.Text = "" Then
            btnsend.Enabled = False
        Else
```

```
            btnsend.Enabled = True
        End If

    End Sub

    Private Sub HelpTopicsToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles mnuOptionsLogout.Click
        ' logging out of a chat discussion

        Timer1.Enabled = False

        MsgBox(txthostname.Text & "-PC" & "[" & txtuserip.Text & "]" & " has logged out : " & TimeOfDay)
        RichTextBox1.Text += (txtusername.Text & "[" & txtuserip.Text & "]" & ">>" & " " & "has logged out: " & TimeOfDay) + vbCrLf

        txtsend1.Enabled = False
        mnuOptionsLogin.Enabled = True
        mnuOptionsLogout.Enabled = False
    End Sub

    Private Sub ToolStripMenuItem2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ToolStripMenuItem2.Click
        ' opening a saved chat discussion
        SavedChatForm.Show()
    End Sub

    Private Sub LoginToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles mnuOptionsLogin.Click
        're-logging into the chat session

        frmstartup.Show()
        Me.Opacity = 0
        btnConnect.Enabled = True
        mnuOptionsLogout.Enabled = True

    End Sub
End Class
```

- **Saved Chat Form**

```
Imports System.IO


Public Class SavedChatForm

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        ' open another saved chat from the saved chat screen
```

```
        Dim filereader As StreamReader

        OpenFD.InitialDirectory = "C:\"
        OpenFD.Title = "Open a Text File"
        OpenFD.Filter = "Text Files|*.txt|Word Files|*.doc|Wordx Files|*.docx"
        If OpenFD.ShowDialog = DialogResult.Cancel Then
            MsgBox("Cancel clicked")
        Else

            filereader = New StreamReader(OpenFD.FileName)

            RichTextBox1.Text = filereader.ReadToEnd()
            TextBox1.Text = OpenFD.FileName

            filereader.Close()
        End If
    End Sub

    Private Sub Form3_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        Dim filereader As StreamReader

        OpenFD.InitialDirectory = "C:\"
        OpenFD.Title = "Open a Text File"
        OpenFD.Filter = "RTF Files|*.rtf|Text Files|*.txt|Word Files|*.doc|Wordx Files|*.docx"
        If OpenFD.ShowDialog = DialogResult.Cancel Then
            MsgBox("Cancel clicked")
        Else

            filereader = New StreamReader(OpenFD.FileName)

            RichTextBox1.Text = filereader.ReadToEnd()
            TextBox1.Text = OpenFD.FileName
            filereader.Close()
        End If

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Me.Close()
    End Sub
End Class
```

- **Help Topics Form**

```
Public Class frmHelpTopics

Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label1.Click
        Label9.Text = "Connecting to other users(Recipient)"
        txthelp.Text = vbCrLf + "1. Input the IP Address or the Computer Name of the Recipient PC into the Recipient IP Textbox" + vbCrLf + vbNewLine + "2. Click(the) 'Connect' Button" + vbCrLf + vbNewLine + "3. If an error message is displayed, read the message, close the message and re-type the Host-IP or Host-Name" + vbCrLf + vbNewLine + "4. Click the Connect button again to connect to the PC" + vbCrLf + vbNewLine + "5. If the connection is successful, the chat box will be enabled (receive Focus), the recipient will recieve a notification of your connection to his/her PC and you can start chatting"

End Sub

Private Sub Label2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label2.Click
        Label9.Text = "Sending a message to a connected user"
        txthelp.Text = vbCrLf + "1. Once you are connected to a user, place your cursor in the Chat box and type your message" + vbCrLf + vbNewLine + "2. Click the send button and the message will be delivered to the recipient's message screen" + vbCrLf + vbNewLine + "3. A copy of the sent message will also be displayed on your message screen"

End Sub

Private Sub Label3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label3.Click
        Label9.Text = "Chatting with a connected user"
        txthelp.Text = vbCrLf + "1. After sending a message to the recipient, the user can choose to send you a feedback which will also be displayed on your chat screen" + vbCrLf + vbNewLine + "2. Once youe receive the feedback, you can further send more messages hence you chatting with the connected user"

End Sub

Private Sub Label4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label4.Click
        Label9.Text = "Logging out of a chat session"
        txthelp.Text = vbCrLf + "1. While chatting with a connected user, click on the Options Menu" + vbCrLf + vbNewLine + "2. Select and click Logout sub-menu" + vbCrLf + vbNewLine + "3. A message box will be displayed on your screen saying you have logged out of the chat Session hence you are automatically looged out and cannot chat with any user until you login again"
```

```
End Sub

Private Sub Label5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label5.Click
        Label9.Text = "Re-logging in to the chat environment"
        txthelp.Text = vbCrLf + "1. Click on the Options Menu" + vbCrLf + vbNewLine + "2. choose the Login sub-menu" + vbCrLf + vbNewLine + "3. The Startup Screen is displayed asking for your name" + vbCrLf + vbNewLine + "4. Input your name and click OK button" + vbCrLf + vbNewLine + "5. You have successfully logged in and ready to connect"
End Sub

Private Sub Label6_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label6.Click
        Label9.Text = "Saving a chat discussion"
        txthelp.Text = vbCrLf + "1. Click on the File Menu" + vbCrLf + vbNewLine + "2. Choose Save Chat sub-menu" + vbCrLf + vbNewLine + "3. On the appearing dialog box, choose which folder/directory on the PC to save your chat into" + vbCrLf + vbNewLine + "4. Type the FileName to save the chat with" + vbCrLf + vbNewLine + "5. The default file format will be RTF(Rich Text File) but you can choose to save as Text or Word Document(file)" + vbCrLf + vbNewLine + "6. Click Save button"
End Sub

Private Sub Label7_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label7.Click
        Label9.Text = "Open a saved chat discussion"
        txthelp.Text = vbCrLf + "1. Click on File Menu" + vbCrLf + vbNewLine + "2. Choose Open Saved Chat sub-menu" + vbCrLf + vbNewLine + "3. Browse to the location of the file you want to open (.rtf, .txt, .doc files)" + vbCrLf + vbNewLine + "4. click on the located file and click Open button" + vbCrLf + vbNewLine + "5. The message is opened on a seprate screen from the chat module" + vbCrLf + vbNewLine + "6. You can close the opened message after reading or open another saved message"
End Sub

Private Sub Label8_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label8.Click
        Label9.Text = ""
        txthelp.Text = vbCrLf + "Browse IPChat Help..." + vbCrLf + vbNewLine + "Click on the Help Topics for guidelines using this application."
End Sub

Private Sub frmHelpTopics_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub
End Class
```

- **About Form**
  - Public Class AboutForm
  -
  - Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
  - Me.Close()
  - frmIPChat.Enabled = True
  - frmIPChat.Focus()
  -
  - End Sub
  -
  - End Class