DESKTOP SHARING APPLICATION USING RFB PROTOCOL

This project report is submitted to
Rashtrasant Tukadoji Maharaj Nagpur University
in the partial fulfilment of the requirement for the award of the degree
of

Bachelor of Engineering in Computer Technology

by

Mr. Akash S. Natkar (CT14057)
Mr. Yugant P. Kadu (CT14060)
Ms. Nilu Kant (CT14061)
Mr. Deepak Kumar Gupta (CT14068)
Mr. Satish W. Wanjari (CT14078)

Under the guidance of

Mr. Bhushan Deshpande
Assistant Professor

2017-2018

DEPARTMENT OF COMPUTER TECHNOLOGY
KAVIKULGURU INSTITUTE OF TECHNOLOGY AND SCIENCE
RAMTEK – 441106

DEPARTMENT OF COMPUTER TECHNOLOGY

KAVIKULGURU INSTITUTE OF TECHNOLOGY AND SCIENCE

RAMTEK-441106

This is to certify that the project report entitled '**Desktop Sharing Application using RFB Protocol** ' carried out by Mr. Akash S. Natkar (CT14057), Mr. Yugant P. Kadu (CT14060), Ms. Nilu Kant (CT14061), Mr. Deepak Kumar Gupta (CT14068) and Mr. Satish W. Wanjari (CT14078) of the B.E. final year of computer technology, during the academic year 2017-2018, in the partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering (Computer Technology)** offered by the **Rashtrasant Tukadoji Maharaj Nagpur University**, Nagpur**.**

Mr. Bhushan Deshpande

Guide

Mr. V.P.Mahatme

Head of the Department

Dr. B. Ram Rathan Lal

Principal

Date:

Place: Ramtek

DECLARATION

We declare that

a. The work contain in this project report has been done by us under the supervision of our guide.
b. The work has not been submitted to any other institute for any degree or diploma.
c. We have followed the guideline provided by institute in preparing the project report.
d. We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the institute.
e. Whenever we have used material (data, theoretical analysis, figures and text) from other sources, we have given due credit to them by citing giving their details in the references.

Project-mates

ACKNOWLEDGEMENT

We are grateful to our respected guide **Mr. Bhushan Deshpande** for his kind, disciplined and invaluable guidance which inspired us to solve all the difficulties that came across during completion of project.

We express our special thanks to **Mr. V. P. Mahatme**, Head of the Department, for his kind support, valuable suggestion and allowing us to use all facilities that are available in the Department during this project.

Our sincere thanks are due to **Dr. B. Ram Rathan Lal**, Principal, for extending all the possible help and allowing us to use all resources that are available in the Institute.

We are also thankful to our parents and friends for their valuable cooperation and standing with us in all difficult conditions.

Project-mates

ABSTRACT

Desktop sharing is a common name for technologies and product that allow remote access and remote collaboration on individual computer screen. It is an online technology that allows to share concurrent session with user on another computer at remote location.

The main objective of this desktop sharing application is to monitor the clients system so that server can see what is going on clients system. Sharing the desktop of one machine with other machines is useful in corporate offices and large establishments to make a remote presentation. When all the machines use the same operating system, it is easy to view the display of one computer from the other. This application presents a client-server framework in python where the display of the client computer is captured at periodic intervals and sent to the server over a network. The sharing system should be efficient in the sense that it should transmit only the changed parts of the screen and it should not consume all the resources while doing. This application can also be used by parents for tracking their children's activity. Moreover this application may be useful in recognizing and monitoring threats, and also for preventing and investigating criminal activity.

RFB ("remote framebuffer") is a simple protocol for remote access to graphical user interfaces. Because it works at the framebuffer level, it is applicable to all windowing systems and applications, including Windows and Macintosh. RFB is the protocol used in VNC (Virtual Network Computing).The remote endpoint where the user sits (i.e. the display plus keyboard and/or pointer) is called the RFB client or viewer. The endpoint where changes to the framebuffer originate (i.e. the windowing system and applications) is known as the RFB server.

This application will be implemented using Python, SQLite data base software, Remote Frame Buffer protocol. Display sharing application required Router, LAN cable for communication python platform.

KEYWORDS: RFB, Macintosh, VNC, SQLite,, Router

CONTENTS

ABBREVIATIONS

Abbreviations	Meaning
RFB	Remote Frame Buffer
VNC	Virtual Network Computing
IP	Internet Protocol
PIL	Python Imaging Library
GUI	Graphical User Interface
GNU	General Public License
LGPL	Lesser General Public License
IDLE	Integrated Development and Learning Environment
I/O	Input/output
IANA	Internet Assigned Numbers Authority
RFB	Remote Frame Buffer
VNC	Virtual Network Computing
IP	Internet Protocol

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

RFB ("remote framebuffer") is a simple protocol for remote access to graphical user interfaces. Because it works at the framebuffer level, it is applicable to all windowing systems and applications, including Windows and Macintosh. RFB is the protocol used in VNC (Virtual Network Computing).

The remote endpoint where the user sits (i.e. the display plus keyboard and/or pointer) is called the RFB client or viewer. The endpoint where changes to the framebuffer originate (i.e. the windowing system and applications) is known as the RFB server.

The protocol also makes the client stateless. If a client disconnects from a given server and subsequently reconnects to that same server, the state of the user interface is preserved. Furthermore, a different client endpoint can be used to connect to the same RFB server. At the new endpoint, the user will see exactly the same graphical user interface as at the original endpoint. In effect, the interface to the user's applications becomes completely mobile. Wherever suitable network connectivity exists, the user can access their own personal applications, and the state of these applications is preserved between accesses from different locations. This provides the user with a familiar, uniform view of the computing infrastructure wherever they go.

Sharing the desktop of one machine with other machines may be needed in corporate offices and large establishments to make a remote presentation. When all the machines use the same operating system, it is easy to view the display of one computer from the other. But, it is difficult over a heterogeneous network running different operating systems on different hardware. This application presents a client-server framework in Java where the display of the client computer is captured at periodic intervals and sent to the server over a network. The sharing system should be efficient in the sense that it should transmit only the changed parts of the screen and it should not consume all the resources while doing.

Computer surveillance is the monitoring of computer activity. The monitoring is often carried out covertly and may be completed by governments, corporations, criminal organizations, or individuals.

Desktop sharing allows sharing an application with remote users. All participants see the same screen view. The main challenges of desktop sharing are scalability, reliability,operating system independence and performance. It is believed that desktop sharing system should be operating system independent because participants can use different operating systems. Also, the system should scale well.

1.1 Purpose

The main objective of this desktop sharing application is to monitor the clients system so that server can see what is going on clients system.Sharing the desktop of one machine with other machines may be needed in corporate offices and large establishments to make a remote presentation. When all the machines use the same operating system, it is easy to view the display of one computer from the other.This application presents a client-server framework in Python where the display of the client computer is captured at periodic intervals and sent to the server over a network. The sharing system should be efficient in the sense that it should transmit only the changed parts of the screen and it should not consume all the resources while doing.This application can also be used by parents for tracking their children's activity.Moreover this application may be useful in recognizing and monitoring threats, and also for preventing and investigating criminal activity.

1.2 Features

- Platform independent: - Platform independence in software means that some code can run with little or no modification on multiple platforms.
- Online message passing: -The invoking program sends a message to a process (which may be an actor or object) and relies on the process and the supporting infrastructure to select and invoke the actual code to run. Message passing differs from conventional programming where a process, subroutine, or function is directly invoked by name. Message passing is key to some models of concurrency and object-oriented programming.

1.2Applications

Application of project are discussed below:

- Desktop sharing-Desktop sharing works by sending packets of information from a host computer to a remote computer describing what's on the host computer's screen at any given time. The encrypted data travels over the Internet. Some data arrives as image files, while others arrive as individual pixels assigned to a particular X and Y coordinate. Desktop sharing software is smart enough to only send information updates on the sections of the screen that have changed and to compress the data significantly, minimizing the amount of necessary bandwidth
- To monitor employee's activity- Employee monitoring allows a business to track employee activities and monitor worker engagement with workplace related tasks. A business using employee monitoring on a computer can measure productivity, track attendance, ensure security and collect proof of hours worked.
- To track children's activity-The internet is a wonderful educational tool that children are increasingly expected to use to support their learning. However, the online world does have a dark side. As a modern parent, it is the responsibility to protect children from online as well as offline dangers.
- To recognize and monitor threats, and prevent and investigate criminal activity- It is the monitoring of computer activity. The monitoring is often carried out covertly and may be completed by governments, corporations, criminal organizations, or individuals. It may or may not be legal and may or may not require authorization from a court or other independent government agencies.

1.4 Problem Statement

To implement remote desktop sharing using RFB protocol on Python platform in which the clients use a very simple and small Python application for connecting to server and they do not need the shared application. The server and receive screen updates from the client and send keyboard and mouse events to the server. The connection between client and server is established using socket. The Graphical User Interface (GUI) is developed using PyQt4 in Python.

1.5 Motivation

This application is developed to provide reliable connectivity. Desktop sharing application is an application through which one can share desktop screen contain live over Ethernet. It is a powerful application applicable in various web conferences, online presentation, remote training and support. With the growth of Multination Corporation and the requirement for large network, companies need and internet technology professionals for distributed offices. This project provide more efficient way to manage take support issues. Using this application the employee's activity can be monitored which can increase measure productivity, track attendance, ensure security and collect proof of hours worked. Using this application child's activity can also be monitored because when it comes down to it, parental monitoring isn't about privacy, it's about safety. And also to recognize and monitor threats and prevent and investigate criminal activity this application is developed. This project is based on distributed computing and networking.

1.6 Organization of Report

Chapter one describes introduction of the project including the main objective.

Chapter two explains literature survey which includes the literature study and reference for the project.

Chapter three presents proposed Approach and System Architecture which includes brief implementation of the project.

Chapter four describes system description which contains the description of the system,

Chapter five contains the result and discussion

Chapter six contains conclusion and future scope.

CHAPTER 2

LITERATURE REVIEW

In desktop sharing application, the server is able to view the screen of its multiple clients. To develop application various literature have been studied.

2.1 Design of a modified RFB protocol and its implementation in an ultra-thin client

Daniel Zinca (2010) implemented modules related to the network communications (the implementation of the proposed modified RFB protocol and the other protocols below), the keyboard interface and mouse interface. This paper describes a modified RFB (Remote Frame Buffer) protocol intended to be used by LAN ultra-thin clients. The prototype implementation is based on the ATMEL AVR microcontroller family and on the Microchip's Ethernet integrated circuit ENC28J60, showing that a low cost microcontroller can implement the network functions. The main differences consist of the use of UDP as the transport protocol instead of TCP and in additional fields to the RFB header.

This paper describes a prototype implementation of an ultra-thin client (called also a zero-client) and focuses on the networking aspects. Generally, the protocols developed for remote access fall in 2 categories:

- Image frame buffers
- High level API display functions

UDP can be used as the replacement of TCP as the transport protocol in an RFB architecture. The advantages of using UDP (compared with TCP) when implementing a thin client are the following:

- Simpler hardware implementation.
- Decreased redundancy.

The drawbacks of using RFB on top of UDP:

- Lower security: it is easy to initiate spoofing attacks.

- No error control: lost packets can have an important influence on the information that is displayed.
- Possible lost packets during congestion periods.

Fig.2.1 RFB protocol sequence flow

The server starts operating as a service. It will wait for connections from hosts. The number of simultaneous connections depends on the operating system. If the operating system is a desktop one (like Microsoft Windows XP or Windows 7) then one remote connection is allowed. For server operating systems, multiple simultaneous connections are available.

After establishing the connection, the protocol enables the transmission of screen contents from the server. In order to improve the use of the network channel, detection of the modified screen at the server was performed and a packet with that modified screen is sent to the client. The transfer mode is Raw, leaving for further

implementation the RRE, HExtile and ZRLE because it requires more processing power in the embedded client.

The client sends the keys (using the KeyEvent message) and the mouse positions (PointerEvent message) back to the server. At the server side, the commands are run and the result is displayed on the screen and then sent to the client. Periodically the client sendsFrameBufferUpdateRequest. As a response, the server detects the portions of the image that are modified and send them in anFrameBufferUpdate. In order to test the communications between the server and the embedded client, all packets that were received over the network connection were sent to the spare serial V.24 interface. From here the characters were captured with a terminal emulation software and saved to a file for later comparison. The buffer contents is limited to 1024 bytes but is sufficient to test the transfer of the FrameBuffer.

2.2 RFB Protocol

RFB ("remote framebuffer") is a simple protocol for remote access to graphical user interfaces. Because it works at the framebuffer level it is applicable to all windowing systems and applications, including X11, Windows and Macintosh. RFB is the protocol used in VNC (Virtual Network Computing).

An RFB server is typically a long-lived process that maintains the state of a framebuffer. RFB clients typically connect, communicate with the server for a period of time to use and manipulate the framebuffer, and then disconnect. A subsequent RFB session will then pick up where a prior session left off, with the state of the framebufferintact.An RFB client contacts the server on TCP port 5900. On systems with multiple RFB servers, server N typically listens on port 5900+N, analogous to the way that X Window servers listen on port 6000+N.

Some browser-based clients use a Python application to run the RFB protocol. .In some cases, the initial roles of the client and server are reversed, with the RFB client listening on port 5500, and the RFB server contacting the RFB client. Once the connection is established, the two sides take their normal roles, with the RFB server sending the first handshake message.

Note that the only port number assigned by IANA for RFB is port 5900, so RFB clients and servers should avoid using other port numbers unless they are communicating with servers or clients known to use the non-standard ports.

2.2.1 Display Protocol

The display side of the protocol is based around a single graphics primitive: “put a rectangle of pixel data at a given x,y position”. At first glance this might seem an inefficient way of drawing many user interface components. However, allowing various different encodings for the pixel data gives us a large degree of flexibility in how to trade off various parameters such as network bandwidth, client drawing speed and server processing speed.

2.2.2 Screen Model

In its simplest form, the RFB protocol uses a single, rectangular framebuffer. All updates are contained within this buffer and may not extend outside of it. A client with basic functionality simply presents this buffer to the user, padding or cropping it as necessary to fit the user’s display.

More advanced RFB clients and servers have the ability to extend this model and add multiple screens. The purpose being to create a server-side representation of the client’s physical layout. Applications can use this information to properly position themselves with regard to screen borders.

In the multiple-screen model, there is still just a single framebuffer and framebuffer updates are unaffected by the screen layout. This assures compatibility between basic clients and advanced servers. Screens are added to this model and act like viewports into the framebuffer. A basic client acts as if there is a single screen covering the entire framebuffer.

The server may support up to 255 screens, which must be contained fully within the current framebuffer. Multiple screens may overlap partially or completely.

2.2.3 Server To Client Message

The server to client message types that all clients must support are:

Table 2.1 Exchange protocol

Number	**Name**
1	SetColourMapEntries

Table 2.1 Exchange protocol (continued)

Number	Name
	When the pixel format uses a "color map", this message tells the client that the specified pixel values should be mapped to the given RGB values.
2	Bell A Bell message makes an audible signal on the client if it provides one.
3	ServerCutText The server has new ISO 8859-1 (Latin-1) text in its cut buffer. Ends of lines are represented by the newline character (hex 0a) alone.

The official, up-to-date list is maintained by IANA (Internet Assigned Numbers Authority).Note that before sending a message with an optional message type a server must have determined that the client supports the relevant extension by receiving some extension-specific confirmation from the client; usually a request for a given pseudo-encoding.

2.3 Parallel Applications

During the development of the application, the following similar type of applications have been studied and detailed information is given below:-

2.3.1 TeamViewer

TeamViewer is developed by TeamViewerGmbHfor remote control of computers and take full control access of remote desktop. TeamViewer is a proprietary computer software package for remote control, desktop sharing, online meetings, webconferen- -cingand file transfer between computers. TeamViewer may be installed with an installation procedure, although the 'Quick Support' version will run without installation.To connect to another computer, TeamViewer has to be running on both machines. To install TeamViewer, administrator accessis required, but once installed it can be run by any user. When TeamViewer is started on a computer, it generates a

partner ID and password (user-defined passwords are also supported). To establish a connection between a local client and a remote client, TeamViewer generated ID and password of either client are required. The local client requires the remote client's ID and password to gain control over the remote client, whereas the remote client requires the local client's ID and password to gain control over the local client.
To start an online meeting, the presenter gives the Meeting ID to the participants. They join the meeting by using the TeamViewer full version or by logging on to http://go.teamviewer.com/ and entering the Meeting ID. It is also possible to schedule a meeting in advance.TeamViewer uses RSA private/public key exchange (2048-bit) and AES (256-bit) session encryption.

In the default configuration, TeamViewer uses one of the servers of TeamViewer.com to start the connection and the routing of traffic between the local client and the remote host machine. The software then determines how to establish a connection. In 70% of the cases after the handshake, a direct connection via UDP or TCP is established; the other connections are routed through TeamViewer GmbH's router network (via TCP or HTTP-tunneling).

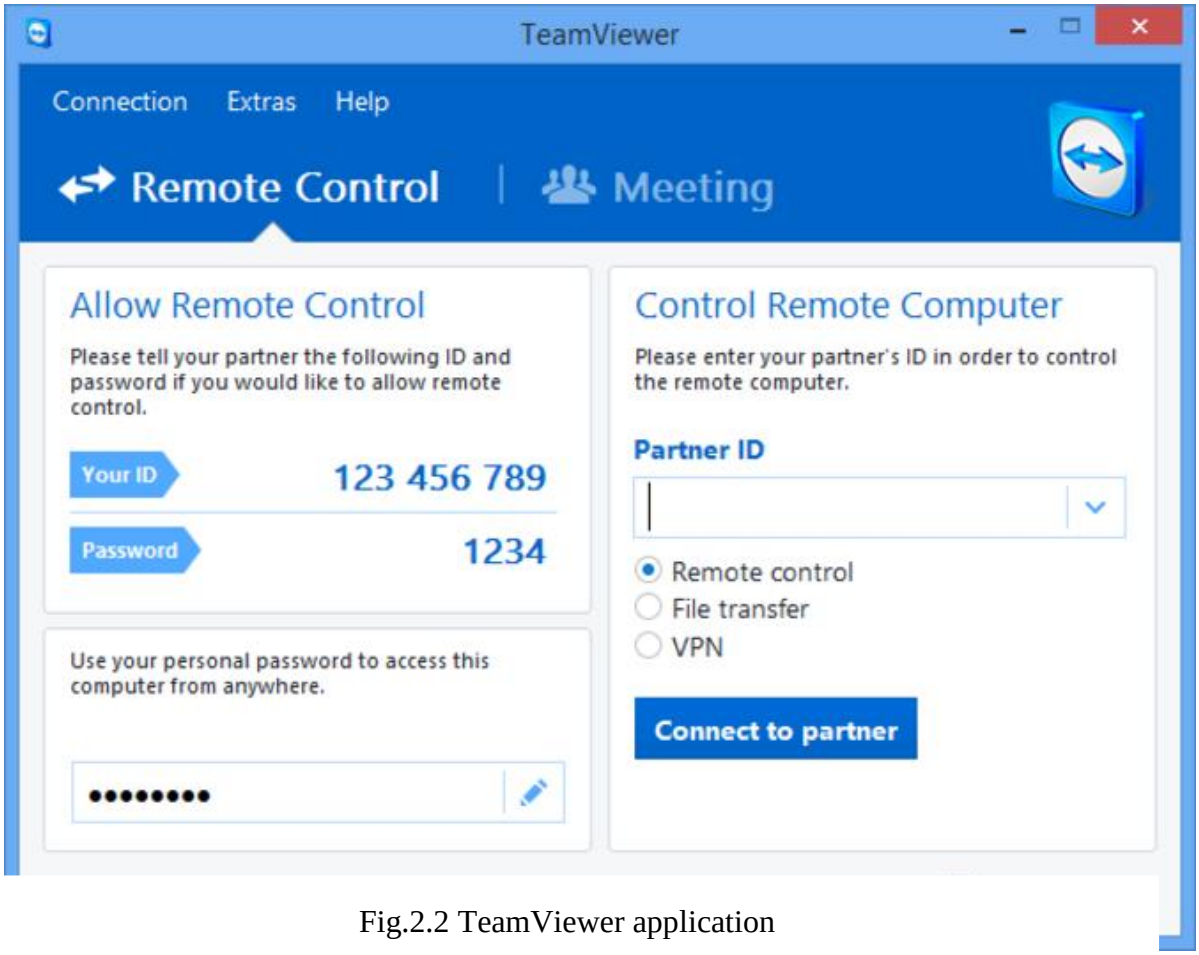

Fig.2.2 TeamViewer application

Table 2.2 Version history

Version	Date	Description
12	December 2016	New features: Windows 10 and Chrome OS support, mass deployment for TeamViewer host on Android Nougat and iOS 10, Smoother Remote Sessions, Remote Control Tabs for macOS, Remote QuickSteps, Service Case Notifications, Remote Sticky Note, High Frame Rate Connections, Simplified Client Interface, Intelligent Connections Setup, OneDrive for Business, Faster File Transfer, Silent Host Roll Out, Remote Device Dashboard, Code Base Re-write for Performance, Device Connection Reports, Mobile to Mobile Remote Connections, Windows Phone Support

Table 2.2 Version history (Continued)

11	December 2015	New features: Universal app for Windows 10, Chrome OS support, mass deployment for TeamViewer host on Android, improved performance, Overhauled toolbar, Unattended access for Android devices, SOS button for support requests, Customer satisfaction form, Connections to Linux console, remote installation of TeamViewer Host, channel groups, group sharing improvements, TeamViewerinbrowser, Chat Widget, Mobile & Web-based TeamViewer Chats, Client redesign.
10	December 2014	New features: Central setting policies, Master whitelist, Chat history, One-click video calls, Profile picture, find nearby contacts, 4K display support, Real-time session notes, Optimized Performance, Computers & Contacts API, idle session timeout, Door lock for online meetings, integration with Dropbox, Google Drive and more, whiteboard for remote control, Improved design
9	December 2013	New features: Open multiple connections in different tabs, Wake-on-LAN, customer modules, copy and paste files via the clipboard, simplified file transfer, service queue, TeamViewer API, desktop shortcuts, notifications, optimized video, redesigned compatibility: Remote control of TeamViewer 9 by TeamViewer 9 clients only, meetings are accessible by TeamViewer 8,7 clients. TeamViewer 9 users can remote control TeamViewer 3, 4, 5, 6, 7 and 8[37]
8	December 2012	New features: Session handover from one expert to another, comment on sessions for billing documentation, share selected groups with other TeamViewer accounts, easy remote printing in the home office, schedule online meetings easily in Microsoft Outlook, session recording, including sound and video for perfect documentation, remote sound and video, remote account logout, automatically lock of the computer after remote access, TeamViewer Management

Table 2.2 Version history (Continued)

		Console compatibility: Remote control of TeamViewer 8 by TeamViewer 9, 8 clients only, meetings are accessible by TeamViewer 7 clients. TeamViewer 8 users can remote control TeamViewer 3, 4, 5, 6 and 7

2.3.2 REAL VNC(Virtual Network Computing)

RealVNC is developed by VNC team at AT&T founded RealVNC Limited. RealVNC is a company that provides remote access software. The software consists of a server and client application for the Virtual Network Computing (VNC) protocol to control another computer's screen remotely. For a desktop-to-desktop connection RealVNC runs on Windows, on Mac OS X, and on many Unix-like operating systems. A Windows-only client, VNC Viewer Plus is now available, designed to interface to the embedded server on Intel AMT chipsets found on Intel vPro motherboards.

For remote access to view one computer desktop on another, RealVNC comes in one of three editions:

1. Open Edition (formerly "Free Edition") – free registration and activation required, Free software version distributed under the GNU General Public License; runs on various flavors of Unix (Linux, Solaris, HP-UX, AIX) and versions of Windows prior to Windows Vista (i.e. Windows NT 4, 2000, XP, Server 2003), newer Windows operating systems must use the Personal or Enterprise editions.[1] Note that users who wish to use this free version, as supplied by the realvnc.com site, may need (for example) to patch and compile the XFree86 source code by themselves, as the free binaries available for download are outdated and typically fail to operate in modern environments.[2] Fortunately, almost all Linux distributions include an updated/customised version of the RealVNC free edition. For example, in Debian and its derivatives, the RealVNC server and client appear under the packages named vnc4server and xvnc4viewer, respectively. (Until June 2012, the opensource "Free Edition 4.1.3" was downloadable without any registration at realvnc.com. As of June 2012, the "free edition 4.1.3" has been renamed to "open edition 4.1.3" and now requires a registration at realvnc.com. There is also a new "Free Edition 5.x" that requires a "free licence key".)

2. Personal Edition – commercial version geared towards home or small-business users, with authentication and encryption, remote printing, chat and file transfer; runs on Windows, Mac OS X and various flavours of Linux and UNIX.
3. Enterprise Edition – commercial version geared towards enterprises, with enhanced authentication and encryption, remote printing, chat, file transfer and a deployment tool for Windows; runs on Windows, Mac OS X and various flavours of Linux and UNIX.

As of release 4.3 (released August 2007) separate versions of both the Personal and Enterprise editions exist for 32-bit and 64-bit systems. Release 4.6 included features such as HTTP proxy support, chat, an address book, remote printing, unicode support, and connection notification.

Users must activate each of the server versions ("Free", "Personal", "Enterprise").

With the release of VNC 5.0 late December 2013, RealVNC software editions used a single binary which superseded VNC Enterprise Edition and VNC Personal Edition.

In November 2016, RealVNC released the updated version of their software, now called VNC Connect (version 6.0). The new version introduces a cloud connection option using a subscription-based pricing model. Users can choose between three subscription levels; Home (free for non-commercial use), Professional and Enterprise. Home and Professional subscriptions are cloud connections only. The Enterprise subscription supports hybrid connections that include the traditional direct (peer to peer) connections and/or cloud connections.

RealVNC clients using vncviewer can run in full-screen mode; they use the F8 function-key as the default key for bringing up an options menu (which includes the option to, among other things, switch off full screen mode or to forward a Control-Alt-Delete key-sequence). The server component of RealVNC allows a computer to be remotely controlled by another computer. RealVNC uses the RFB protocol. It defaults to TCP port 5900. When making a connection over the Internet, the user must open this port in the local firewall as well as configure port forwarding to forward TCP Port 5900 (or the customized port respectively) to the local machine address if behind a NAT Router. As an alternative, one can tunnel VNC through SSH, avoiding the opening of additional ports and automatically traversing the NAT router. SSH also provides encryption of the connection between the VNC server and viewer.

After proposing remote access interface for Weston in October 2013, RealVNC published a Wayland developer preview in July 2014.The VNC protocol is pixel based. Although this leads to great flexibility (e.g., any type of desktop can be displayed), it is often less efficient than solutions that have a better understanding of the underlying graphic layout, like X11. Those protocols send graphic primitives or high-level commands in a simpler form (e.g., open window), whereas RFB just sends the raw pixel data.

Fig.2.3 Real VNC

.

2.4 Python (Programming Language)

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, and a syntax that allows programmers to express concepts in fewer lines of code, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open sources oftware and has a community–based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are optional. It has fewer syntactic exceptions and special cases than C or Pascal.

Features of Python Programming

1. Python is a simple programming language which is very easier to learn: Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows to focus on the solution rather than syntax.
2. Free for everyone, installed patches, community version and open-source: one can freely use and distribute Python, even for commercial use. Not only can one use and distribute software written in it, one can even make changes to the Python's source code. Python has a large community constantly improving it in each iteration.
3. Portable and cross platform that run code anywhere and anytime: Python programs can be moved from one platform to another, and run it without any changes. It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.
4. An extensible architecture designed to accommodate changes and embeddable: Suppose an application requires high performance. One can easily combine pieces of C/C++ or other languages with Python code. This will give an application high performance as well as scripting capabilities which other languages may not provide out of the box.
5. A high-level, interpreted language containing advanced programming features: Unlike C/C++, tasks like memory management, garbage collection and so on is not performed. Likewise, when Python code is run, it automatically converts code to the language computer understands.

6. Large standard libraries to solve common tasks for simplicity and flexibility: Python has a number of standard libraries which makes life of a programmer much easier. For example: Need to connect MySQL database on a Web server? import MySQLdb. Standard libraries in Python are well tested and used by hundreds of people..

7. Python supports object-oriented programming with classes and inheritance: Everything in Python is an object. Object oriented programming (OOP) helps to solve a complex problem intuitively. OOP helps to break complex problem into smaller set.

Applications of Python

1. Web Applications:

Web application helps to create scalable Web Apps using frameworks and CMS (Content Management System) that are built on Python. Some of the popular platforms for creating Web Apps are: Django, Flask, Pyramid, Plone, Django CMS.Sites like Mozilla, Reddit, Instagram and PBS are written in Python.

2. Scientific and Numeric Computing:

There are numerous libraries available in Python for scientific and numeric computing. There are libraries like: SciPy and NumPy that are used in general purpose computing. And, there are specific libraries like: EarthPy for earth science, AstroPy for Astronomy and so on.Also, the language is heavily used in machine learning, data mining and deep learning.

3. Creating software Prototypes:

Python is slow compared to compiled languages like C++ and Java. It might not be a good choice if resources are limited and efficiency is a must.However, Python is a great language for creating prototypes. For example: one can use Pygame (library for creating games) to create game's prototype first.

4. Good Language to Teach Programming:

Python is used by many companies to teach programming to kids and newbies.It is a good language with a lot of features and capabilities. Yet, it's one of the easiest language to learn because of its simple easy-to-use syntax.

2.5 Screenshot Module-pyscreenshot

The pyscreenshot module can be used to copy the contents of the screen to a PIL or Pillow image memory using various back-ends.Replacement for the ImageGrab Module, which works on Windows only, so Windows users don't need this library.

The pyscreenshot tries to allow taking screenshots without installing 3rd party libraries. It is cross-platform but useful for Linux based distributions. It is only a pure Python wrapper, a thin layer over existing back-ends. Its strategy should work on most Linux distributions: a lot of back-ends are wrapped, if at least one exists then it works, if not then one back-end should be installed. Performance and interactivity are not important for this library.

2.6 GUI Toolkit-PYQT4

PyQt is a Python binding of the cross-platform GUI toolkit Qt, implemented as a Python plug-in. PyQt is free software developed by the British firm Riverbank Computing. It is available under similar terms to Qt versions older than 4.5; this means a variety of licenses including GNU General Public License (GPL) and commercial license, but not the GNU Lesser General Public License(LGPL). PyQt supports Microsoft Windows as well as various flavours ofUNIX, includingLinuxandmacOS.

PyQt implements around 440 classes and over 6,000 functions and methods including:

- a substantial set of GUI widgets
- classes for SQL databases (ODBC, MySQL, PostgreSQL, Oracle, SQLite)
- QScintilla, Scintilla-based rich text editor widget
- data aware widgets that are automatically populated from a database
- an XML parser
- SVG support
- classes for embedding ActiveX controls on Windows (only in commercial version)

PyQt4 comprises a number of different components. First of all there are a number of Python extension modules. These are all installed in the PyQt4 Python package.

- The QtCore module: This contains the core non-GUI classes, including the event loop and Qt's signal and slot mechanism. It also includes platform independent abstractions for Unicode, threads, mapped files, shared memory, regular expressions, and user and application settings.
- The QtGui module: This contains the majority of the GUI classes.
- The QtHelp module: This contains classes for creating and viewing searchable documentation.
- The QtNetwork module: This module contains classes for writing UDP and TCP clients and servers. It includes classes that implement FTP and HTTP clients and support DNS lookups.
- The QtOpenGL module: This module contains classes that enable the use of OpenGL in rendering 3D graphics in PyQt4 applications.
- The QtScript module: This module contains classes that enable PyQt4 applications to be scripted using Qt's JavaScript interpreter.
- The QtScriptTools module: This module contains classes that contain additional components (e.g. a debugger) that are used with Qt's JavaScript interpreter.
- The QtSql module: This module contains classes that integrate with SQL databases. It includes editable data models for database tables that can be used with GUI classes. It also includes an implementation ofSQLite.
- The QtSvg module: This module contains classes for displaying the contents of SVG files.
- The QtTest module: This module contains functions that enable unit testing of PyQt4 applications. (PyQt4 does not implement the complete Qt unit test framework. Instead it assumes that the standard Python unit test framework can be used and implements those functions that simulate a user interacting with a GUI.)
- The QtWebKit module: This module implements a web browser engine based on the WebKit open source browser engine.
- The QtXml module: This module contains classes that implement SAX and DOM interfaces to Qt's XML parser.
- The QtXmlPatterns module: This module contains classes that implement XQuery and XPath support for XML and custom data models.

- The phonon module: This module contains classes that implement a cross-platform multimedia framework that enables the use of audio and video content in PyQt4 applications.
- The QtDBus module: This Unix-only module provides classes that support Inter-Process Communication using the D-Bus protocol.
- The QtDeclarative module: This module provides a declarative framework for building highly dynamic, custom user interfaces using QML.
- The QtMultimedia module: This module provides low-level multimedia functionality. Application developers would normally use the phonon module.
- The QtAssistant module: This module contains classes that allow Qt Assistant to be integrated with a PyQt4 application to provide online help. This module is not available with Qt v4.7 and later - use the QtHelpmodule instead.
- The QtDesigner module: This module contains classes that allow Qt Designer to be extended using PyQt4. See Writing Qt Designer Plugins for a full description of how to do this.
- The QAxContainer module: This Windows-only module contains classes that allow access to ActiveX controls and COM objects.
- The Qt module: This module consolidates the classes contained in all of the modules described above into a single module. It has the disadvantage that it loads the whole of the Qt framework, thereby increasing the memory footprint of an application. The DBus support module is installed as dbus.mainloop.qt. This module provides support for the Qt event loop in the same way that the dbus.mainloop.glib included with the standard dbus-pythonbindings package provides support for the GLib event loop. The API is described in DBus Support. It is only available if the dbus-python v0.80 (or later) bindings package is installed. The QtDBus module provides a more Qt-like interface to DBus.
- The uic module: This module contains classes for handling the .ui files created by Qt Designer that describe the whole or part of a graphical user interface. It includes classes that load a .ui file and render it directly, and classes that generate Python code from a .ui file for later execution.
- The pyqtconfig module is an extention of the SIP build system and is created when PyQt4 is configured. It encapsulates all the necessary information about Qt installation and makes it easier to write installation scripts for bindings built on top of PyQt4. It is covered in detail in The PyQt4 Build System.

PyQt4 also contains a number of utility programs:

- pyuic4 corresponds to the Qt uic utility. It converts GUIs created using Qt Designer to Python code.
- pyrcc4 corresponds to the Qt rcc utility. It embeds arbitrary resources (eg. icons, images, translation files) described by a resource collection file in a Python module.
- pylupdate4 corresponds to the Qt lupdate utility. It extracts all of the translatable strings from Python code and creates or updates .ts translation files. These are then used by Qt Linguist to manage the translation of those strings.

2.7 IDLE an Integrated Development Environment for Python

IDLE (short for integrated development environment or integrated development and learning environment) is an integrated development environment for Python, which has been bundled with the default implementation of the language since 1.5.2b1 It is packaged as an optional part of the Python packaging with many Linux distributions. It is completely written in Python and the Tkinter GUI toolkit (wrapper functions for Tcl/Tk).

IDLE is intended to be a simple IDE and suitable for beginners, especially in an educational environment. To that end, it is cross-platform, and avoids feature clutter.

IDLE has been criticized for various usability issues, including losing focus, lack of copying to clipboard feature, lack of line numbering options, and general user interface design; it has been called a "disposable" IDE, because users frequently move on to a more advanced IDE as they gain experience.

Author Guido van Rossum says IDLE stands for "Integrated DeveLopment Environment", and since van Rossum named the language Python partly to honor British comedy group Monty Python, the name IDLE was probably also chosen partly to honor Eric Idle, one of Monty Python's founding members.

IDLE has the following features:

- coded in 100% pure Python, using the tkinter GUI toolkit
- cross-platform: works mostly the same on Windows, Unix, and Mac OS X
- Python shell window (interactive interpreter) with colorizing of code input, output, and error messages

- multi-window text editor with multiple undo, Python colorizing, smart indent, call tips, auto completion, and other features
- search within any window, replace within editor windows, and search through multiple files (grep)
- debugger with persistent breakpoints, stepping, and viewing of global and local namespaces
- configuration, browsers, and other dialogs
- Multi-window text editor with syntax highlighting, auto completion, smart indent and other.
- Python shell with syntax highlighting.
- Integrated debugger with stepping, persistent breakpoints, and call stack visibility.

IDLE has two main window types, the Shell window and the Editor window. It is possible to have multiple editor windows simultaneously. Output windows, such as used for Edit / Find in Files, are a subtype of edit window. They currently have the same top menu as Editor windows but a different default title and context menu.IDLE's menus dynamically change based on which window is currently selected. Each menu documented below indicates which window type it is associated with.

2.8 Database Environment-SQLite

SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world with more applications than we can count, including several high-profile projects.

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. The database file format is cross-platform – can be freely copy a database between 32-bit and 64-bit systems or between big-endian and little-endianarchitectures. These features make SQLite a popular choice as an Application File Format. Think of SQLite not as a replacement for Oracle but as a replacement for fopen()

SQLite is a compact library. With all features enabled, the library size can be less than 500KiB, depending on the target platform and compiler optimization settings. (64-bit code is larger. And some compiler optimizations such as aggressive function inlining and loop unrolling can cause the object code to be much larger.) There is a trade-off between memory usage and speed. SQLite generally runs faster the more memory. Nevertheless, performance is usually quite good even in low-memory environments. Depending on how it is used, SQLite can befaster than direct filesystem I/O.

SQLite is very carefully tested prior to every release and has a reputation for being very reliable. Most of the SQLite source code is devoted purely to testing and verification. An automated test suite runs millions and millions of test cases involving hundreds of millions of individual SQL statements and achieves 100% branch test coverage. SQLite responds gracefully to memory allocation failures and disk I/O errors. Transactions are ACID even if interrupted by system crashes or power failures. All of this is verified by the automated tests using special test harnesses which simulate system failures. Of course, even with all this testing, there are still bugs. But unlike some similar projects (especially commercial competitors) SQLite is open and honest about all bugs and provides bugs lists and minute-by-minute chronologies of code changes.

The SQLite code base is supported by an international team of developers who work on SQLite full-time. The developers continue to expand the capabilities of SQLite and enhance its reliability and performance while maintaining backwards compatibility with the published interface spec, SQL syntax, and database file format. The source code is absolutely free to anybody who wants it, but professional support is also available.

The SQLite project was started on 2000-05-09. The future is always hard to predict, but the intent of the developers is to support SQLite through the year 2050. Design decisions are made with that objective in mind.

2.9 PY2exe an exe converter

py2exe is a Python extension which converts Python scripts (.py) intoMicrosoft Windows executables (.exe). These executable can run on a system without Python installed.It is the most common tool for doing so.py2exe was used to distribute the official BitTorrent client (prior to version 6.0) and is still used to distribute SpamBayesas well as other projects.Since May 2014, there is a version of

py2exe available for Python 3.Before then, py2exe was made only for Python 2, and it was necessary to use an alternative like cx_Freeze for Python 3 code.

Although this program transforms a .py file to an .exe, it does not make it run faster as py2exe just bundles the Python bytecode rather than converting it to machine-code. It may even run slower than using the Python interpreter directly because of startup overhead. Py2exe is a simple way to convert Python scripts into Windows .exe applications. It is an utility based in Distutils that allows user to run applications written in Python on a Windows computer without requiring the user to install Python. It is an excellent option when user need to distribute a program to the end user as a standalone application. Py2exe currently only works in Python 2.x.

Now, in order to be able to create the executable we need to create a file called setup.py in the same folder where the script user want to be executable is located:

**********Setup for py2exe**********

```
from distutils.core import setup

import py2exe

setup(console=['myscript.py'])
```

****

In the code above, we are going to create an executable for myscript.py. The setup function receives a parameter console=['myscript.py'] telling py2exe that we have a console application called myscript.py.

Then in order to create the executable just run python setup.py py2exe from the Windows command prompt (cmd). User will see a lot of output and then two folders will be created: dist and build. The build folder is used by py2exe as a temporary folder to create the files needed for the executable. The dist folder stores the executable and all of the files needed in order to run that executable. It is safe to delete the build folder.

CHAPTER 3
PROPOSED APPROACH AND SYSTEM ARCHITECTURE

A system architecture is a conceptual model that defines the structure, behavior and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. In this chapter different modules of the application are discussed.

3.1 System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. A system design and architecture of desktop sharing application is proposed as follows:

Physical Layer

Lowest layer is a physical layer. It activates, maintains and deactivates the physical connection. It is responsible for transmission and reception of the unstructured raw data over network. Data rates needed for transmission is defined in the physical layer. Data encoding is also done in this layer. In this project, Ethernet plays can important in establishing Server-Client connection.

Network Layer

Network layer routes the signal through different channels from one node to other. It acts as a network controller. It manages the Subnet traffic. It decides by which route data should take. It divides the outgoing messages into packets and assembles the incoming packets into messages for higher levels. IP (Internet Protocol) is one of the component of the Network Layer. IP has the task of delivering packets from the source host to the destination host solely based on the IP addresses in the packet headers. For this purpose, IP defines packet structures that encapsulate the data

to be delivered. It also defines addressing methods that are used to label the datagram with source and destination information. In this application, client uses IP address of server for reliable connectivity. When connection is established, data packet from network layer are transferred to transport layer.

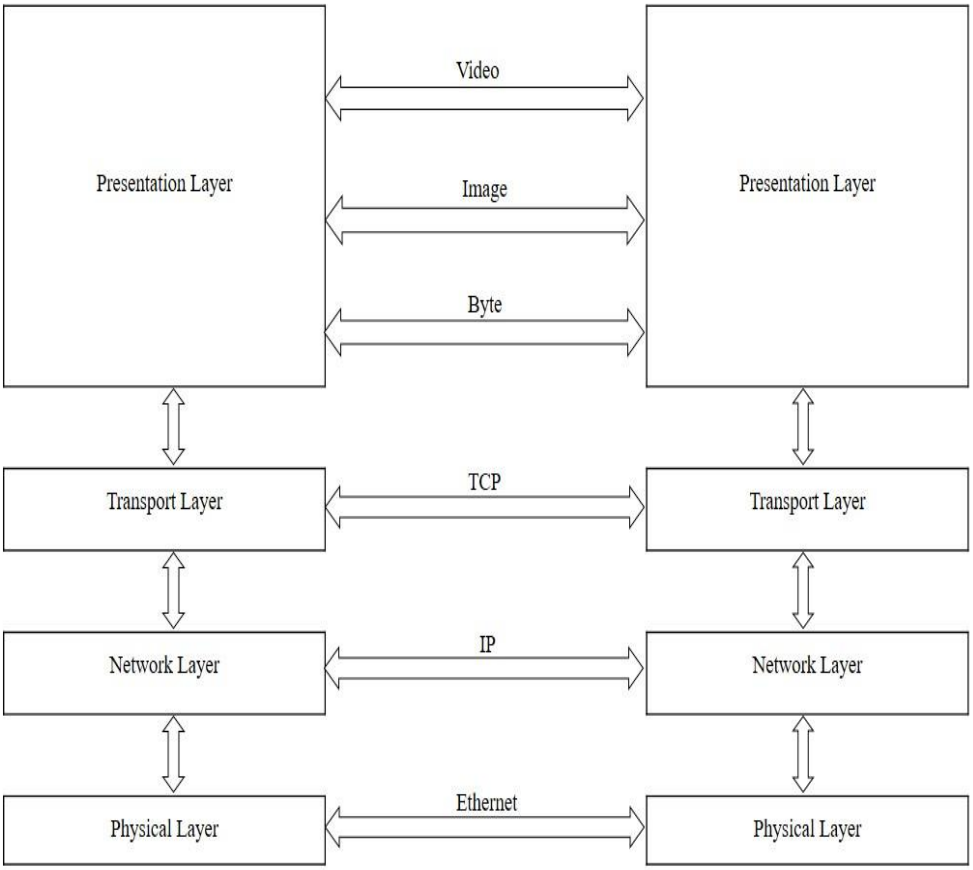

Fig.3.1 Protocol stack

Transport Layer

Transport Layer decides if data transmission should be on parallel path or single path. Functions such as Multiplexing, Segmenting or Splitting on the data are done by this layer. Transport layer can be very complex, depending upon the network requirements. Transport layer breaks the message (data) into small units so that they are handled more efficiently by the network layer.TCP (Transmission Control Protocol) provides reliable, ordered, and error-checked delivery of a stream of octets between applications running on hosts communicating by an IP network. Major Internet applications such as the World Wide Web, email, remote administration, and file transfer rely on TCP. TCP/IP Sockets are used for client server connection. TCP is

a connection-oriented protocol, which means a connection is established and maintained until the application programs at each end have finished exchanging messages. Data from transport layer are transferred to presentation layer.

Presentation Layer

Presentation layer takes care that the data is sent in such a way that the receiver can understand the information (data) and can be able to use the data. It performs Data compression, Data encryption, Data conversion etc. Byte Stream, Images and Video takes part in presentation layer.

3.1.1 Peer to Peer

In the client/server programming model, a server program awaits and fulfils requests from client programs, which may be running in the same or other computers. The server can take the request and make sure that the request is valid. If everything checks out okay, then the server can fetch the request and serve the client.Server gets connected to Ethernet. Then server gets online and wait for client to respond.

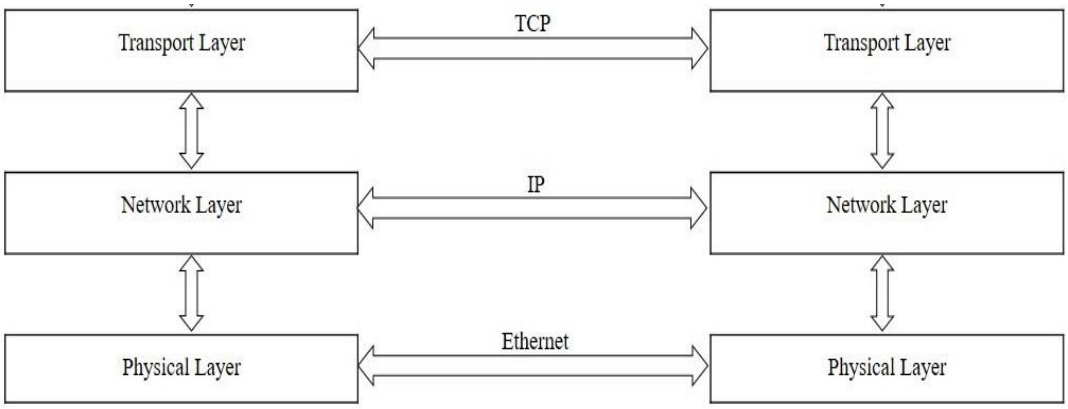

Fig.3.2 Peer to peer connection

3.1.2 ScreenCapture

A frame typically includes frame synchronization features consisting of a sequence of bits or symbols that indicate to the receiver, the beginning, and end of the payload data within the stream of symbols or bits it receives. If a receiver is connected to the system in the middle of a frame transmission, it ignores the data until it detects a new frame synchronization sequence.The Frame which has been captured by Client is being received at Server side. While transferring image from client to server, if

eventually session of the client is over due to technical reason then alert message is shown on server window. And it shows the size of the image.

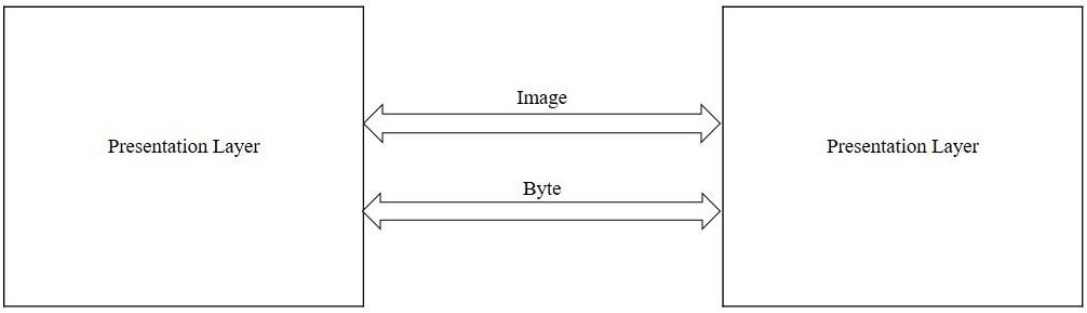

Fig.3.3 File sharing

3.1.3 I/O Stream

A frame is usually transmitted serial bit by bit and contains a header field and a trailer field that "frame" the data. (Some control frames contain no data.)The received image saved at buffer location. And then image is display on server screen. In desktop client server application, in which the capturing of the frames is done, in jpg images, rendered by the renderer and displaying them on the server side. Now to upload the images to the server. After getting connected, Client's current screen gets captured and is sent to Server.

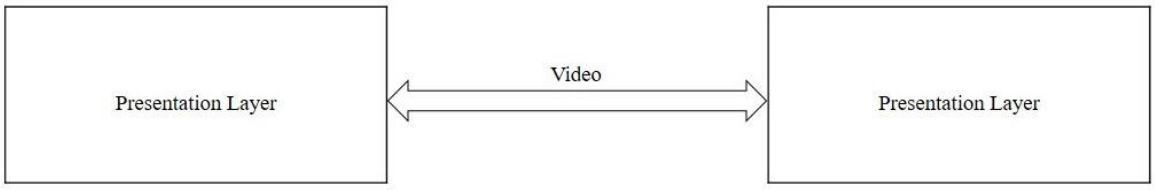

Fig.3.4 Video streaming

3.2 Sequence Diagram

RFB protocol specification is pretty well defined. According to Wikipedia, RFB protocol has several versions. For this article, the focus can be on common messages that should be understood properly by most VNC implementations regardless of

protocol version. After a VNC viewer (client) establishes a TCP connection to a VNC server (RFB service), the first phase involves the exchange of protocol version:

RFB Service ---------- "RFB 003.003\n" -------> VNC viewer

RFB Service <---------- "RFB 003.008\n" -------- VNC viewer

It's a simple stream of bytes which can be decoded into ASCII characters, such as "RFB 003.008\n".

Once that is done, the next step is authentication. VNC server sends an array of bytes to indicate what type of authentications it supports. For example:

RFB Service ---------- 0x01 0x02 ----------> VNC viewer

RFB Service <---------- 0x02 ---------- VNC viewer

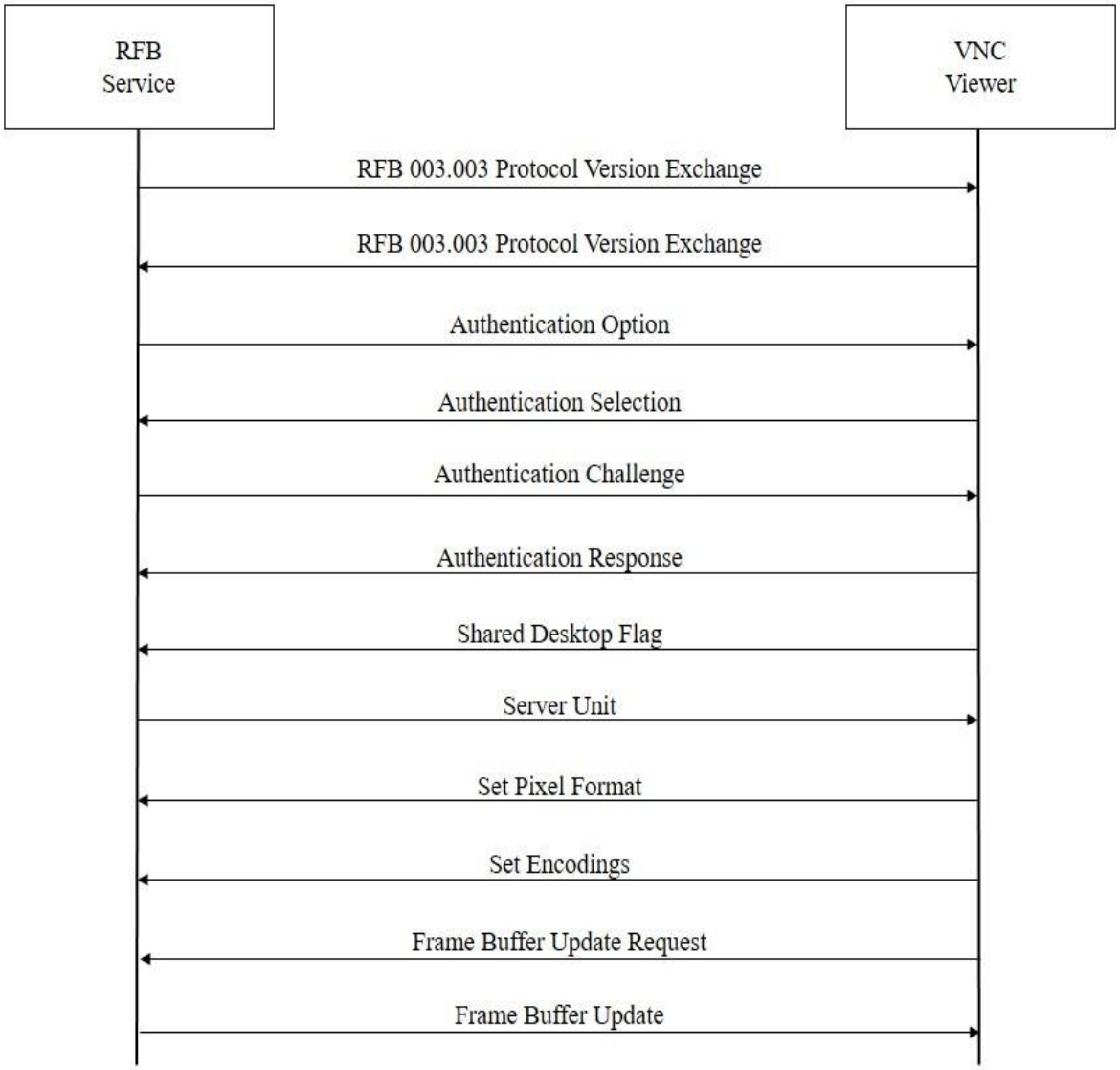

Fig.3.5 Sequence diagram

Here the VNC server sent only 1 possible authentication type (0x02). The first byte 0x01 denotes the number of authentication types available. VNC viewer has to reply with value 0x02, since that's the only possible type supported by the server in this example.Next, server can send authentication challenge (depending on which algorithm, there are several), and the client has to respond with proper challenge response message and wait for the server to confirm the response. Once the client is authenticated, they can continue with the process of session establishment.

The simplest way here is to choose no authentication at all. RFB protocol is insecure anyway, regardless of authentication mechanism. If security is important, the proper way would be to tunnel RFB sessions via VPN or SSH connections.At this point, VNC viewer sends a shared desktop message which tells if the client can share and allow other VNC viewers to connect to the same desktop. It's up to RFB service implementation to consider that message and possibly prevent multiple VNC viewers from sharing the same screen. This message is only 1 byte in length, and a valid value is either 0x00 or 0x01.

Finally the RFB server sends a server init message, which contains screen dimension, bits per pixel, depth, big endian flag and true color flags, maximum values for red, green and blue colors, bit positions in pixel for red, green and blue colors, and desktop string/title. First two bytes represent screen width in pixels, next two bytes are screen height. After screen height bytes, bits per pixel byte should be present in message. The value is usually 8, 16, or 32. On most modern systems with full color range, bits per pixel byte has value 32 (0x20). It tells client that it can request full color for each pixel from server. Big endian byte is non-zero only if pixels are in big endian order. If true color byte is non-zero (true) then the next six bytes specify how to extract red, green and blue color intensities from the pixel value. Next six bytes are maximum allowed values for red, green and blue component of pixel. This is important in 8-bit color mode, where only few bits are available for each color component. Red, green, and blue shifts determine bit positions for each color. Last three bytes are padding and should be ignored by the client. After pixel format, there is a byte that defines length of a string for desktop title. Desktop title is an ASCII encoded string in byte array of arbitrary length.

3.3 Data Flow Diagram

Desktop sharing application is proposed as follows: TCP/IP Socket helps in Client-Server connection. RFB protocol provide authentication to the Client by exchange of protocol version of RFB Service and VNC Viewer. It's a simple stream of bytes which can be decoded into ASCII characters, such as "RFB 003.008\n" as shown fig. 3.2. Once Client is authenticated, desktop sharing application captures Client-side display. The image file or frame is converted into byte stream. This byte stream data is encrypted to provide security and send to Server-side.

After receiving encrypted byte stream data, Server-side decrypt it and regain the file as a frame. Then Server display the image. To render this image as video, multiple screenshots of the Client-side desktop is captured continuously and transferred it to Server-side.

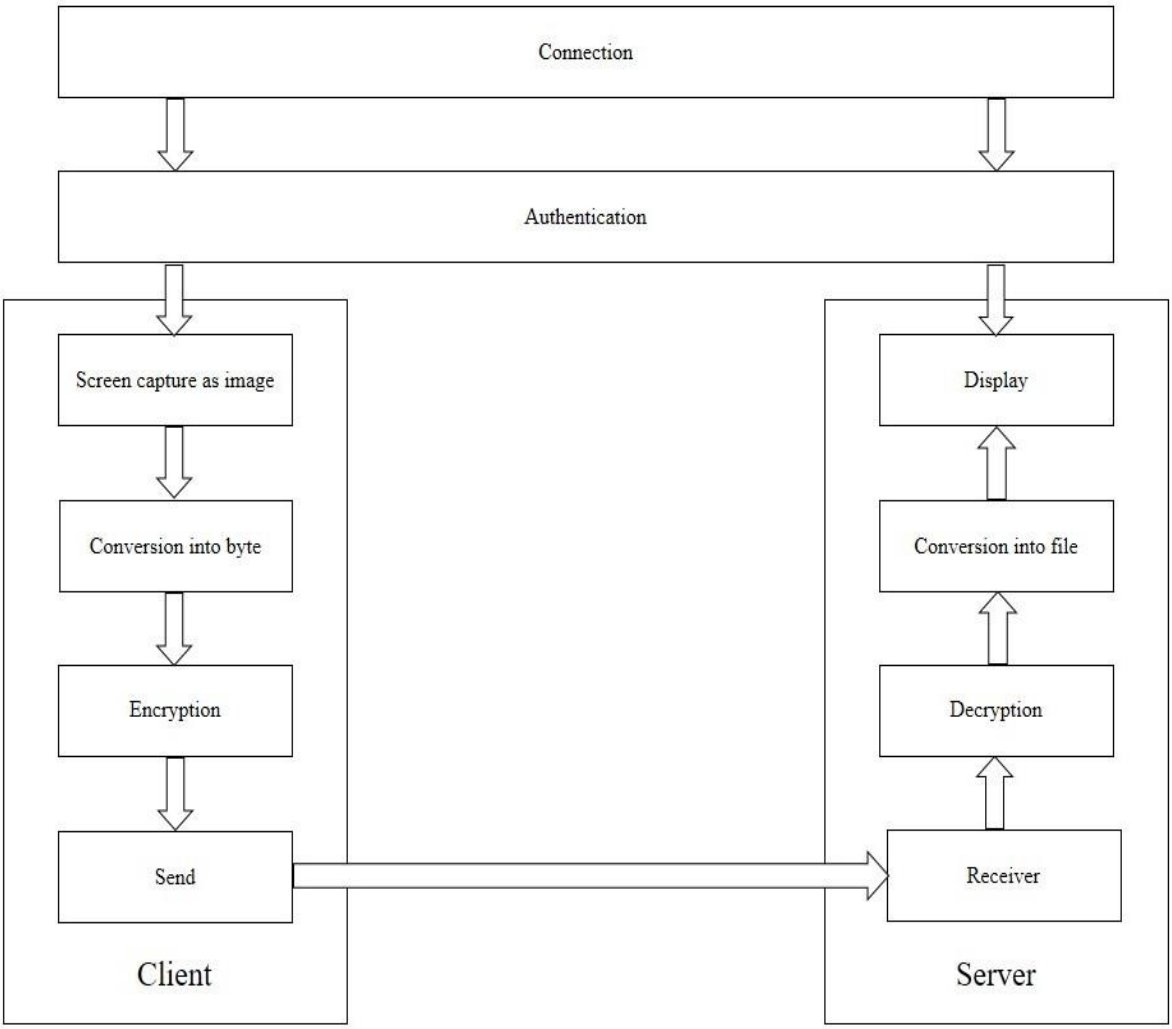

Fig.3.6 Data flow diagram

3.4 Development and Deployment Requirements

This application has been developed using SQLite data base software on Python 2.7 platform. The Python script is used to write program and PyQT4 tool is used to develop elegant GUI. This application is developed using Python because it provide more flexibility and bundles of standard libraries support. The software and hardware required for this application are as follows:

3.4.1 Software Requirements

Software requirements refer to the minimum capability of a system must have to execute this particular application.

- Operating System: Windows 10 / 8.1 / 8 / 7 / Vista / XP / Mac / Linux / Ubuntu / Linux / Any Linux compatible.
- Languages: Python with bundled packages.
- Graphical User Interface: PyQT4.
- Database Support : SQLite3.

3.4.2 Hardware Requirements

Hardware requirements refer to the minimum capability of a system must have to execute this particular application.

- Processor: 2.4GHz Intel Core i3-7100U processor / 1.8GHz AMD AMD E2-9000e processor.
- RAM (SD/DDR): Up to 4 GB .
- Hard Disk: 50 GB.
- Local Area Network: 100/1000 MB Ethernet Support.

CHAPTER 4
IMPLEMENTATION

Implementation is the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy. The application aims at giving prior information about remote desktop. The implementation of this project requires client server connectivity followed by image capturing. Captured image displayed on server side using PYQT4.

4.1 Socket Establishment

Sockets are the endpoints of a bidirectional communications channel. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.

Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on. The socket library provides specific classes for handling the common transports as well as a generic interface for handling the rest.

4.1.1 Socket Connection for Server

To write Internet servers, we use the socket function available in socket module to create a socket object. A socket object is then used to call other functions to setup a socket server. Now call bind(hostname, port) function to specify a port for service on the given host.

Next, call the accept method of the returned object. This method waits until a client connects to the port specified, and then returns a connection object that represents the connection to that client.

4.1.2 Socket Connection for Client

To write a very simple client program which opens a connection to a given port 12345 and given host. This is very simple to create a socket client using Python's socket module function. The socket.connect(hosname, port) opens a TCP connection to hostnameon the port. Once have a socket open, one can read from it like any IO object. When done, remember to close it, as one would close a file.

The following code is a very simple client that connects to a given host and port, reads any available data from the socket, and then exits.

Table 4.1Server socket methods

Sr.No.	Method & Description
1	s.bind() This method binds address (hostname, port number pair) to socket.
2	s.listen() This method sets up and start TCP listener.
3	s.accept() This passively accept TCP client connection, waiting until connection arrives (blocking).

Table 4.2 Client socket methods

Sr.No.	Method & Description
1	s.connect() This method actively initiates TCP server connection.

Table 4.3 General socket methods

Sr.No.	Method & Description
1	s.recv() This method receives TCP message
2	s.send() This method transmits TCP message
3	s.recvfrom() This method receives UDP message
4	s.sendto() This method transmits UDP message
5	s.close() This method closes socket
6	socket.gethostname() Returns the hostname.

4.2 Delay Function

Delay function helps to delay the execution of desired program. Python's time module has a handy function called sleep(). Essentially, as the name implies, it pauses Python program. time.sleep() is the equivalent to the Bash shell's sleep command. Almost all programming languages have this feature, and is used in many use-cases.

4.2.1 time.sleep()

The method sleep() suspends execution for the given number of seconds. The argument may be a floating point number to indicate a more precise sleep time.

The actual suspension time may be less than that requested because any caught signal will terminate the sleep() following execution of that signal's catching routine.

Syntax

Following is the syntax for sleep() method −time.sleep(t)

Parameters

- t − This is the number of seconds execution to be suspended.

Suspend execution for the given number of seconds. The argument may be a floating point number to indicate a more precise sleep time. The actual suspension time may be less than that requested because any caught signal will terminate the sleep() following execution of that signal's catching routine. Also, the suspension time may be longer than requested by an arbitrary amount because of the scheduling of other activity in the system.

**********Delay method example**********

```
import time                    #importing time module from python library
def sleeper():                 #declaration of method
   while True:
      num = raw_input('How long to wait: ')
      try:
         num = float(num)
      except ValueError:
         print('Please enter in a number.\n')
         continue
      print('Before: %s' % time.ctime())
      time.sleep(num)                        #method used for delay
      print('After: %s\n' % time.ctime())
try:
   sleeper()
```

```
except KeyboardInterrupt:
   print('\n\nKeyboard exception received. Exiting.')
   exit()
**************************************************************************
```

4.3 pyscreenshot

The pyscreenshot module can be used to copy the contents of the screen to a PIL or Pillowimage memory. Replacement for the ImageGrab Module, which works on Windows onlygrab and show the whole screen

```
**************************pyscreenshot example*****************************
import pyscreenshot as ImageGrab    #import pyscreenshot module
if __name__ == "__main__":          #creation of the constructor
        im=ImageGrab.grab()         #to capture the screen
        im.show()                   #show() method is used to show captured image
```

to start the example:

```
python -m pyscreenshot.examples.showgrabfullscreen
```

grab and show the part of the screen

```
#-- include('examples/showgrabbox.py')--#
importpyscreenshot as ImageGrab
if __name__ == "__main__":
        im=ImageGrab.grab(bbox=(10,10,510,510)) # X1,Y1,X2,Y2
        im.show()
```

to start the example:

```
python -m pyscreenshot.examples.showgrabbox
```

4.3.1 makedirs()

The method makedirs() is recursive directory creation function. Like mkdir(), but makes all intermediate-level directories needed to contain the leaf directory.

Syntax:

Following is the syntax for makedirs() method −os.makedirs(path[, mode])

Parameters

- path − This is the path, which needs to be created recursively.
- mode − This is the Mode of the directories to be given.

Return Value

This method does not return any value.

Example

The following example shows the usage of makedirs() method.

```
**************************makedirs() example*****************************
import os, sys                          #import os and system module
path = "/tmp/home/monthly/daily"        #set the path
os.makedirs( path, 0755 );              #call makedirs() method for create directory
print "Path is created"
*************************************************************************
```

When we run above program, it produces following result −

```
Path is created
```

4.4 Database Environment-SQLite

SQLite is a self-contained, server-less, config-free transactional SQL database engine. Python gained the sqlite3 module all the way back in version 2.5 which means that one can create SQLite database with any current Python without downloading any additional dependencies. Mozilla uses SQLite databases for its popular Firefox browser to store bookmarks and other various pieces of information.To use sqlite3 module, one must first create a connection object that represents the database and then optionally create a cursor object, which will help to executing all the SQL statements.

Following Python code shows how to connect to an existing database. If the database does not exist, then it will be created and finally a database object will be returned.

**********Database creation**********

```
import sqlite3                          #importing sqlite3 database module
conn = sqlite3.connect('test.db')       ''' crating database if database not exist simultaneously creating connection between database named test.db and assign reference to conn variable '''
print "Opened database successfully";
```

****

To create a new table in an SQLite database from a Python program, use the following steps:

First, create a Connection object using the connect() function of the sqlite3 module. Second, create a Cursor object by calling the cursor() method of the Connection object. Third, pass the CREATE TABLE statement to the execute() method of the Cursorobject, and execute this method.

Insert operation

To insert rows into a table in SQLite database, use the following steps:

First, connect to the SQLite database by creating a Connection object. Second, create a Cursor object by calling the cursor method of the Connection object. Third, execute an INSERT statement. If user want to pass arguments to the INSERT statement, user use the a question mark (?) as the placeholder for each argument.

**********Data insertion operation**********

```
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
    VALUES (1, 'Paul', 32, 'California', 20000.00 )");
conn.commit()
print "Records created successfully";
conn.close()
```

****

Update operation

To update data in a table from a Python program, follow the steps below:

First, create a database connection to the SQLite database using the connect() function. Once the database connection created, user can access the database using the Connection object.

Second, create a Cursor object by calling the cursor() method of the Connection object.

Third, execute the UPDATE statement by calling the execute() method of the Cursor object.

********** Data updation operation **********

```
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
conn.execute("UPDATE COMPANY set SALARY = 25000.00 where ID = 1")
conn.commit
print "Total number of rows updated :", conn.total_changes
print "Operation done successfully";
conn.close()
```

****

Delete operation

In order to delete data in the SQLite database from a Python program, user use the following steps:

First, establish a connection the SQLite database by creating a Connection object using the connect() function. Second, to execute a DELETE statement, user need to create a Cursor object using the cursor()method of the Connection object. Third step is to execute the DELETE statement using the execute() method of theCursor object. In case user want to pass the arguments to the statement, user use a question mark (?) for each argument.

********** Data deletion operation **********

```
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
conn.execute("DELETE from COMPANY where ID = 2;")    ‘’’execution of queries on database ‘’’
```

```
conn.commit()
print "Operation done successfully";
conn.close()
```

Select operation

To query data in an SQLite database from Python, use these steps:

First, establish a connection to the the SQLite database by creating a Connection object.Next, create a Cursor object using the cursor method of the Connection object.Then, execute the SELECT statement.After that, call the fetchall() method of the cursor object to fetch the data.Finally, loop the cursor and process each row individually.

```
********** Data fetch operation **********
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
cursor = conn.execute("SELECT id, name, address, salary from COMPANY")
‘’’ execution of select query on database and store data cursor variable with execution of select query’’’
for row in cursor:
    print "ID = ", row[0]
    print "NAME = ", row[1]
    print "ADDRESS = ", row[2]
    print "SALARY = ", row[3], "\n"
    print "Operation done successfully";
conn.close()
**********
```

Table 4.4 Database for registered users

ID	First	Last	Username	Password
1	Sahil	Tale	sahil@gmail.com	tale1234
2	Mark	Joseph	mark@gmail.com	joseph567
3	Sooraj	Hasan	sooraj@rediff.com	hasan256

Table 4.4 Database for registered users(continued)

4	Fatima	Sheikh	fatima@yahoo.com	sheikh278

4.5 Python Imaging Library-Pillow

Python Imaging Library (abbreviated as PIL) (in newer versions known as Pillow) is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X and Linux.

Pillow offers several standard procedures for image manipulation. These include:

- per-pixel manipulations
- masking and transparency handling
- image filtering, such as blurring, contouring, smoothing, or edge finding
- image enhancing, such as sharpening, adjusting brightness, contrast or color
- adding text to images and much more.

The Python Imaging Library adds image processing capabilities to Python interpreter. This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities.The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.
Let's look at a few possible uses of this library.

Image Archives

The Python Imaging Library is ideal for image archival and batch processing applications. User can use the library to create thumbnails, convert between file formats, print images, etc.The current version identifies and reads a large number of formats. Write support is intentionally restricted to the most commonly used interchange and presentation formats.

Image Display

The current release includes Tk PhotoImage and BitmapImage interfaces, as well as a Windows DIB interface that can be used with PythonWin and other Windows-based toolkits. Many other GUI toolkits come with some kind of PIL support.

For debugging, there's also a show() method which saves an image to disk, and calls an external display utility.

Image Processing

The library contains basic image processing functionality, including point operations, filtering with a set of built-in convolution kernels, and colour space conversions. The library also supports image resizing, rotation and arbitrary affine transforms. There's a histogram method allowing user to pull some statistics out of an image. This can be used for automatic contrast enhancement, and for global statistical analysis.

4.6 Image Module

The Image module provides a class with the same name which is used to represent a PIL image. The module also provides a number of factory functions, including functions to load images from files, and to create new images.

Examples

The following script loads an image, rotates it 45 degrees, and displays it using an external viewer (usually xv on Unix, and the paint program on Windows).

Open, rotate, and display an image (using the default viewer)

**********Example of displaying an image**********

```
from PIL import Image

im=Image.open("bride.jpg")

im.rotate(45).show()
```

Functions

PIL.Image.open(fp, mode='r')

Opens and identifies the given image file.

This is a lazy operation; this function identifies the file, but the file remains open and the actual image data is not read from the file until user try to process the data (or call the load() method). See new().

Parameters: fp – A filename (string), pathlib.Path object or a file object. The file object must implement read(), seek(), and tell() methods, and be opened in binary mode. mode – The mode. If given, this argument must be “r”.
Returns: An Image object.
Raises: IOError – If the file cannot be found, or the image cannot be opened.

4.7 Multithreading

Running several threads is similar to running several different programs concurrently, but with the following benefits −

- Multiple threads within a process share the same data space with the main thread and can therefore share information or communicate with each other more easily than if they were separate processes.
- Threads sometimes called light-weight processes and they do not require much memory overhead; they are cheaper than processes.

A thread has a beginning, an execution sequence, and a conclusion. It has an instruction pointer that keeps track of where within its context it is currently running.

- It can be pre-empted (interrupted)
- It can temporarily be put on hold (also known as sleeping) while other threads are running - this is called yielding.

4.7.1 Starting a New Thread

To spawn another thread, user need to call following method available in *thread* module −

thread.start_new_thread(function, args[, kwargs])

This method call enables a fast and efficient way to create new threads in both Linux and Windows.

The method call returns immediately and the child thread starts and calls function with the passed list of *args*. When function returns, the thread terminates.

Here, *args* is a tuple of arguments; use an empty tuple to call function without passing any arguments. *kwargs* is an optional dictionary of keyword arguments.

4.7.2 The Threading Module

The newer threading module included with Python 2.4 provides much more powerful, high-level support for threads than the thread module discussed in the previous section.

The *threading* module exposes all the methods of the *thread* module and provides some additional methods –

- **threading.activeCount()** − Returns the number of thread objects that are active.

- **threading.currentThread()** − Returns the number of thread objects in the caller's thread control.

- **threading.enumerate()** − Returns a list of all thread objects that are currently active.

In addition to the methods, the threading module has the *Thread* class that implements threading. The methods provided by the *Thread* class are as follows −

- **run()** − The run() method is the entry point for a thread.
- **start()** − The start() method starts a thread by calling the run method.
- **join([time])** − The join() waits for threads to terminate.
- **isAlive()** − The isAlive() method checks whether a thread is still executing.
- **getName()** − The getName() method returns the name of a thread.
- **setName()** − The setName() method sets the name of a thread.

4.7.3 Creating Thread Using Threading Module

To implement a new thread using the threading module, do the following −

- Define a new subclass of the *Thread* class.
- Override the *__init__(self [,args])* method to add additional arguments.
- Then, override the run(self [,args]) method to implement what the thread should do when started.

Once created the new *Thread* subclass, create an instance of it and then start a new thread by invoking the *start()*, which in turn calls *run()* method.

4.8 PYQT4

PyQt is widely used for developing graphical interfaces that can be run on various operating systems. It's one of the most popular GUI choices for Python programming.

4.8.1 QT4 Window

********** Adds a window to a PyQt4 window **********

To import PyQT4 module:

```
from PyQt4.QtGui import *
```

To create the PyQT4 application object using QApplication():

```
a = QApplication(sys.argv)
```

To create the window (QWidget), resize, set the title and show it with this code:

```
w = QWidget()
w.resize(320, 240)
w.setWindowTitle("Hello World!")
```

To show the window:

```
w.show()
```

****

4.8.2 QT4 Buttons

PyQt4 (Qt4) supports buttons through the QPushButtonwidget. Code can be extended to display a button in the center of the window. The button will show a tooltip if hovered and when pressed will close the program

********** Adds a button to a PyQt4 window**********

```
import sys
from PyQt4.QtGui import * #importing pyqt4 module from python library
a = QApplication(sys.argv)
w = QWidget()
w.resize(320, 240)
w.setWindowTitle("Hello World!")
```

```
btn = QPushButton('Hello World!', w)
btn.setToolTip('Click to quit!')
btn.clicked.connect(exit)
btn.resize(btn.sizeHint())
btn.move(100, 80)
w.show()
sys.exit(a.exec_())
```

****

4.8.3 PyQt4 signals and slots

A button click should do something. To do so, one must use signals and slots. If a user does an action such as clicking on a button, typing text in a box – the widget sends out a signal. Signals can be connected with a slot, that acts as a receiver and acts on it.

**********Example for signals and slots**********

```
import sys
from PyQt4.QtCore import pyqtSlot
from PyQt4.QtGui import *
app = QApplication(sys.argv)
w = QWidget()
w.setWindowTitle('Button click example @pythonspot.com')
btn = QPushButton('Click me', w)
@pyqtSlot()
defon_click():
print('clicked')
@pyqtSlot()
defon_press():
print('pressed')
@pyqtSlot()
defon_release():
print('released')
# connect the signals to the slots
btn.clicked.connect(on_click)
btn.pressed.connect(on_press)
```

```
btn.released.connect(on_release)
w.show()
app.exec_()
```

4.8.4 QT4 Widgets

Various widgets that can be access with PyQT. Including:

- Textbox
- Combobox
- Calendar

Textbox Widget

Input fields are present in nearly every application. In PyQT4 an input field can be created using the QLineEdit() function.

**********Example for textbox widget**********

```
import sys
from PyQt4.QtGui import *
a = QApplication(sys.argv)
w = QMainWindow()
w.resize(320, 100)
w.setWindowTitle("PyQT Python Widget!")
textbox = QLineEdit(w)
textbox.move(20, 20)
textbox.resize(280,40)
w.show()
sys.exit(a.exec_())
```

4.8.5 QT4 Pixmaps (Images)

The images can be displayed in a PyQT window using the Pixmapwidget. The constructor of Pixmap takes the image path as parameter:

```
pixmap = QPixmap(os.getcwd() + 'https://pythonspot-9329.kxcdn.com/logo.png')
```

This image needs to be in the same directory as program. The QPixmap widget supports png and jpeg. Example code below

PyQT load image in Pixmap

The Pixmap is attached to a label which is drawn to the screen.

**********Example for pixmap**********

```
import os
import sys
from PyQt4.QtGui import *
app = QApplication(sys.argv)
w = QWidget()
w.setWindowTitle("PyQT4 Pixmap @ pythonspot.com ")   #set title of window
label = QLabel(w)
pixmap = QPixmap(os.getcwd() + 'https://pythonspot-9329.kxcdn.com/logo.png')
label.setPixmap(pixmap)
w.resize(pixmap.width(),pixmap.height())
w.show()
app.exec_()
```

****

CHAPTER 5

RESULT AND DISCUSSION

In the previous chapter the information related to the application is provided which consists of each and every details of the project. The software tools required to build the project along with its functional and non-functional requirements. Study about python for coding required for the implementation.

Following are the screen snapshots of some of application parts shown on the Server side and Client side desktop at different interval of time:

5.1 Authentication

The process or action of verifying the identity of a user is called as authentication. Authentication is a process in which the credentials provided are compared to those on file in a database of authorized users' information on a local operating system or within an authentication server. If the credentials match, the process is completed and the user is granted authorization for access. The permissions and folders returned define both the environment the user sees and the way he can interact with it, including hours of access and other rights such as the amount of allocated storage space.

The process of an administrator granting rights and the process of checking user account permissions for access to resources are both referred to as authorization. The privileges and preferences granted for the authorized account depend on the user's permissions, which are either stored locally or on the authentication server. The settings defined for all these environment variables are set by an administrator.

5.1.1 Login Form

Login form is used for providing authentication that is for recognizing a user's identity. It contains two text label that is username and password which accepts username and password of already registered user. Login form is the very first step of this application to provide authentication to user.

Text input for Login form are as follows:

- Username: Registered user enters allocated username.
- Password: Registered user enters password.

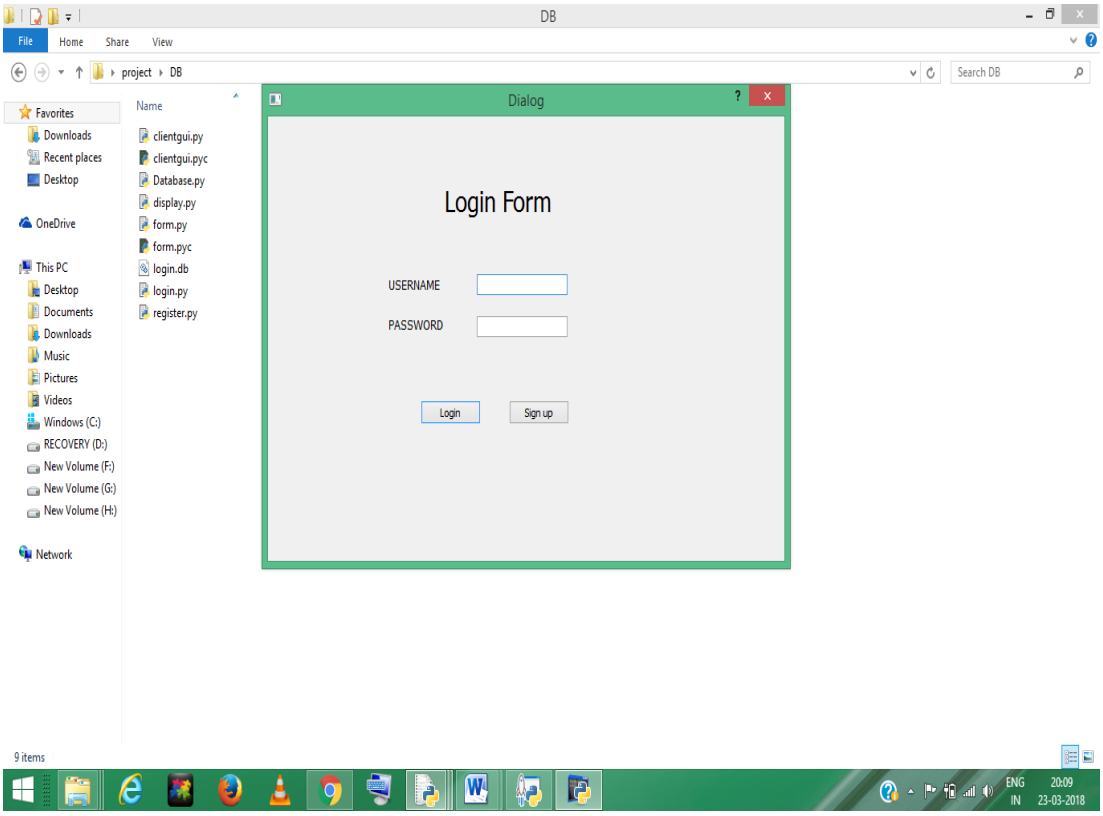

Fig. 5.1 Login form

5.1.1 Registration Form

If user is not registered then after clicking on register button a new window will be opened requesting for registering. Registration form contains First Name, Last Name, Email ID, Password, Re-enter Password and IP Address field. After filling the required fields in the registration window, user need click on submit button and logs into the application.

Text input for Registration form are as follows:

- First Name: First Name of the registering user
- Last Name: Last Name of the registering user
- Email ID: Email ID for registration.
- Password: Password for providing authentication.
- Re-enter Password: Rechecking pre-entered password

Below image shows GUI on server side which consist of three buttons i.e Submit, Reset and Cancel.

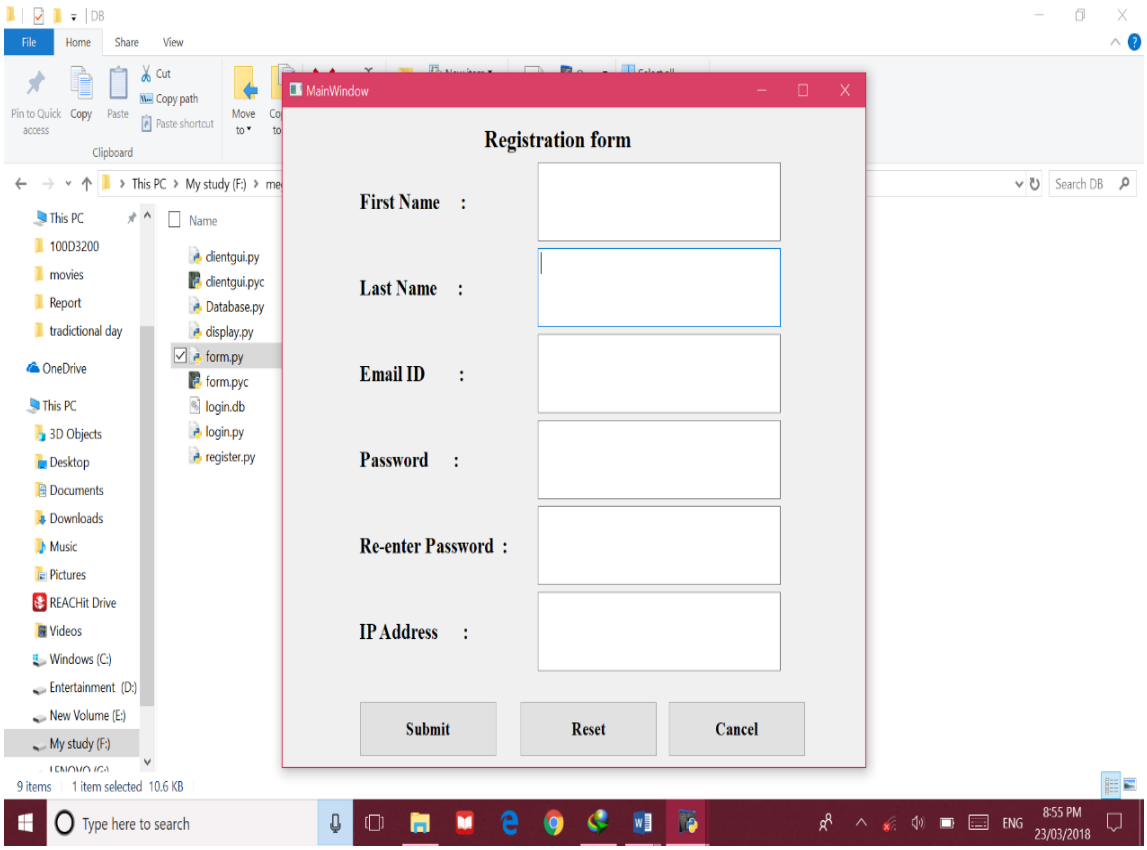

Fig.5.2 Registration form

5.2 Peer to Peer

Connection between Server and Client are established using sockets. The Internet protocol suite provides end-to-end data communication specifying how data should be packetized, addressed, transmitted, routed, and received.

5.2.1 Server Side

A Server is a computer program that provides services to other computer programs in the same or other computers. In the client/server programming model, a server program awaits and fulfils requests from client programs, which may be running in the same or other computers. Below image shows GUI on server side which consist of four buttons i.e. Connect, Start, Stop, Multi and Refresh.

- Connect: Connect button is used to start Socket connection with specific client on server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the incoming connection.
- Multi: Multi Button is used to broadcast images.
- Refresh: Refresh button is used for update the connected clients to check if client is online or not.

When user starts the server.py application, Socket connection on server side gets started. In this case, Server waits for incoming connections from client side.

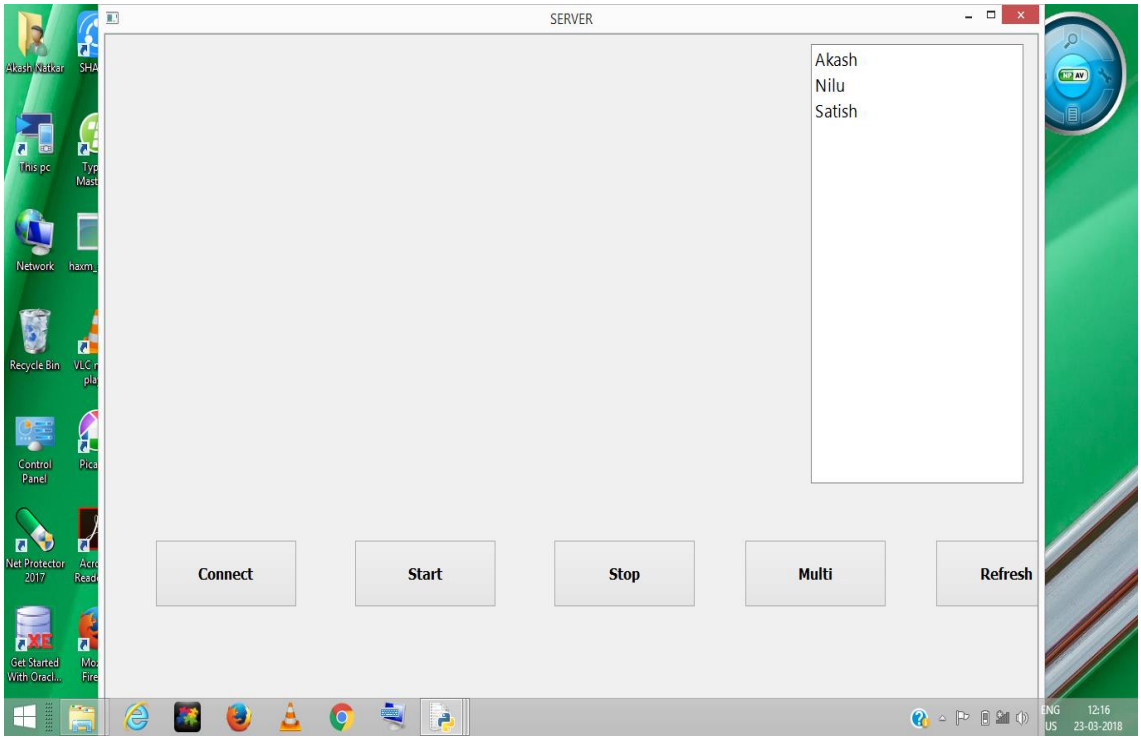

Fig.5.3 Server application online

5.2.2 Client Side

A client is a computer that connects to and uses the resources of a remote computer, or server. Many corporate networks comprise a client computer for each employee, each of which connects to the corporate server. The server provides resources like files, information, Internet and intranet access, and external processing power. In the case of processing, any work done on the server is referred to as "server-side" work. Any work done on the local client is similarly called "client-side". A client is the requesting program or user in a client/server relationship. Below image shows GUI on client side which consist of four buttons i.e. Send, Start and Stop.

- Send: Send button is used to send image to server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the connection.

After getting acknowledgment that server is online, client starts the client.py application and get connected to server. Following successful connection, server selects the online

user present in the list and then clicks on Connect button to request image from client. Then client clicks on Send button to invoke the Screenshot event and send it to server.

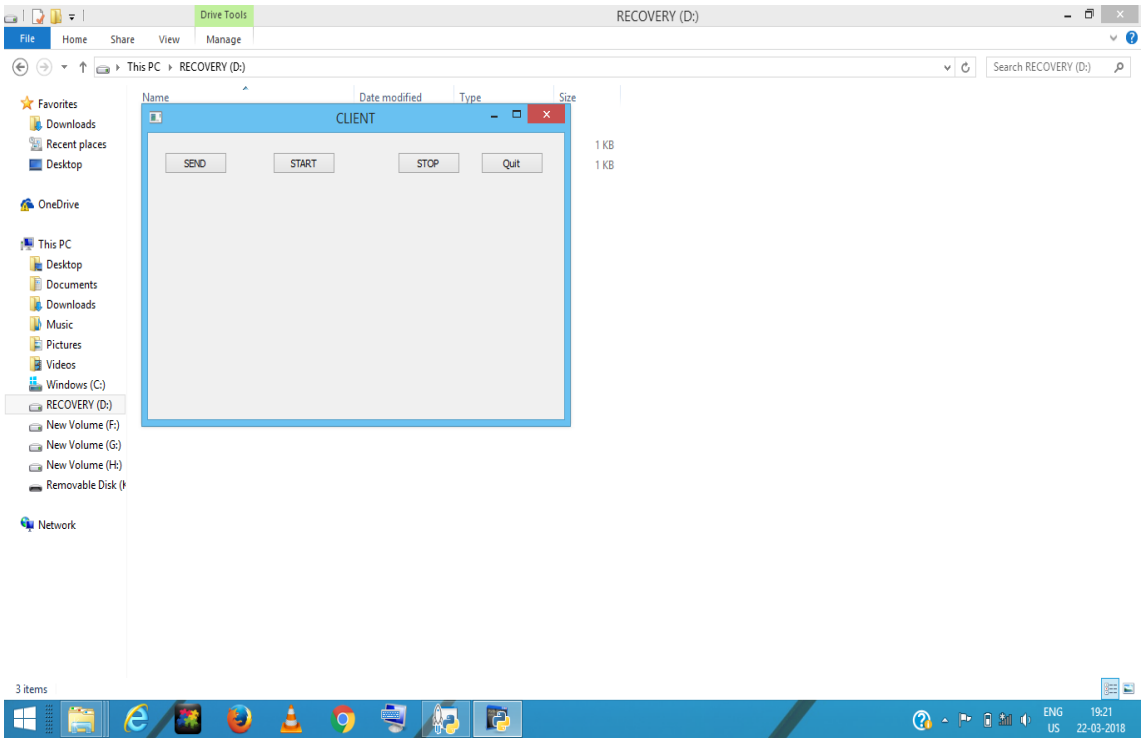

Fig.5.4 Online client application

5.2.3 After Receiving of Image

After successfully receiving image from client side, server grabs this image and display it on GUI. This scenario reflects the real time activity of client at remote location.

Below image shows GUI on server side which consist of four buttons i.e. Connect, Start, Stop, Multi and Refresh.

- Connect: Connect button is used to start Socket connection with specific client on server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the incoming connection.
- Multi: Multi Button is used to broadcast images.
- Refresh: Refresh button is used for update the connected clients.

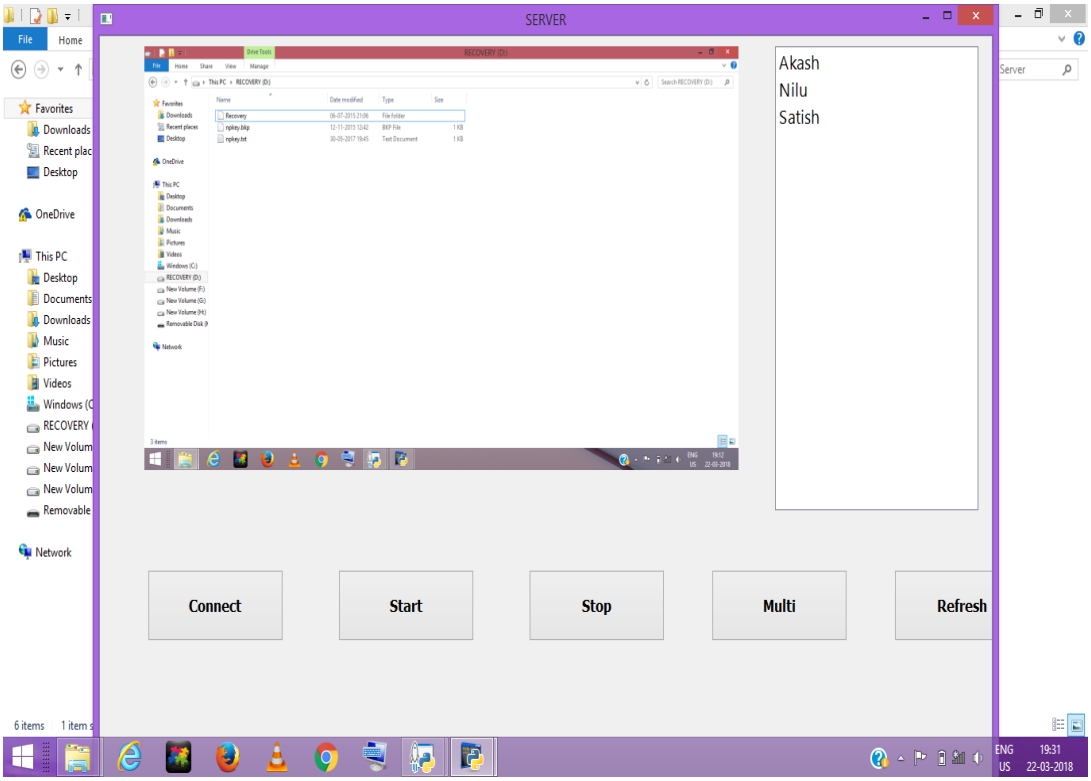

Fig. 5.5 Retrieval of image

5.3 Multiple Images

In multiple images, images at certain interval of time is captured to track the client activity at real time.

5.3.1 Server

To start multiple image activity, user at server side choose the client for this operation. Then user clicks on Start button. Connect: Connect button is used to start Socket connection with specific client on server side. Following buttons are present :

- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the incoming connection.
- Multi: Multi Button is used to broadcast images.
- Refresh: Refresh button is used for update the connected clients.

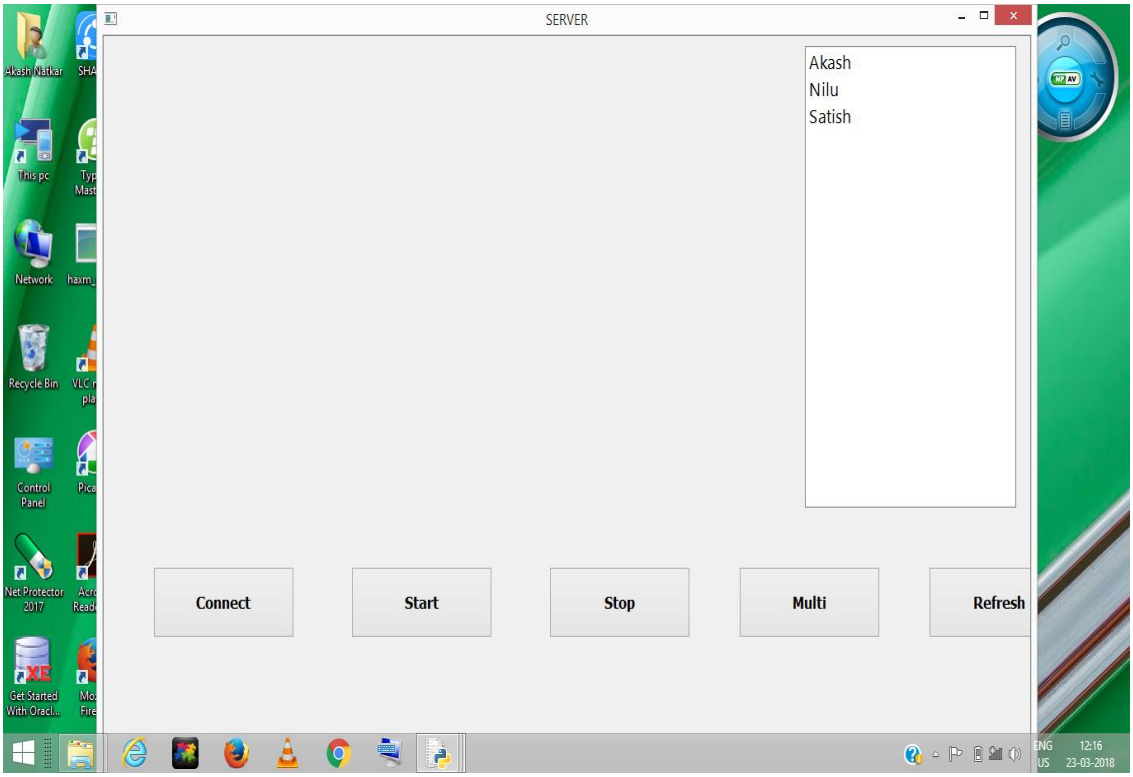

Fig. 5.6 Server application online for multiple images

5.3.2 Client

After getting acknowledgement that server wants to start communication, client clicks on start button. Below image shows GUI on client side which consist of four buttons i.e. Send, Start and Stop.

- Send: Send button is used to send image to server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the connection.

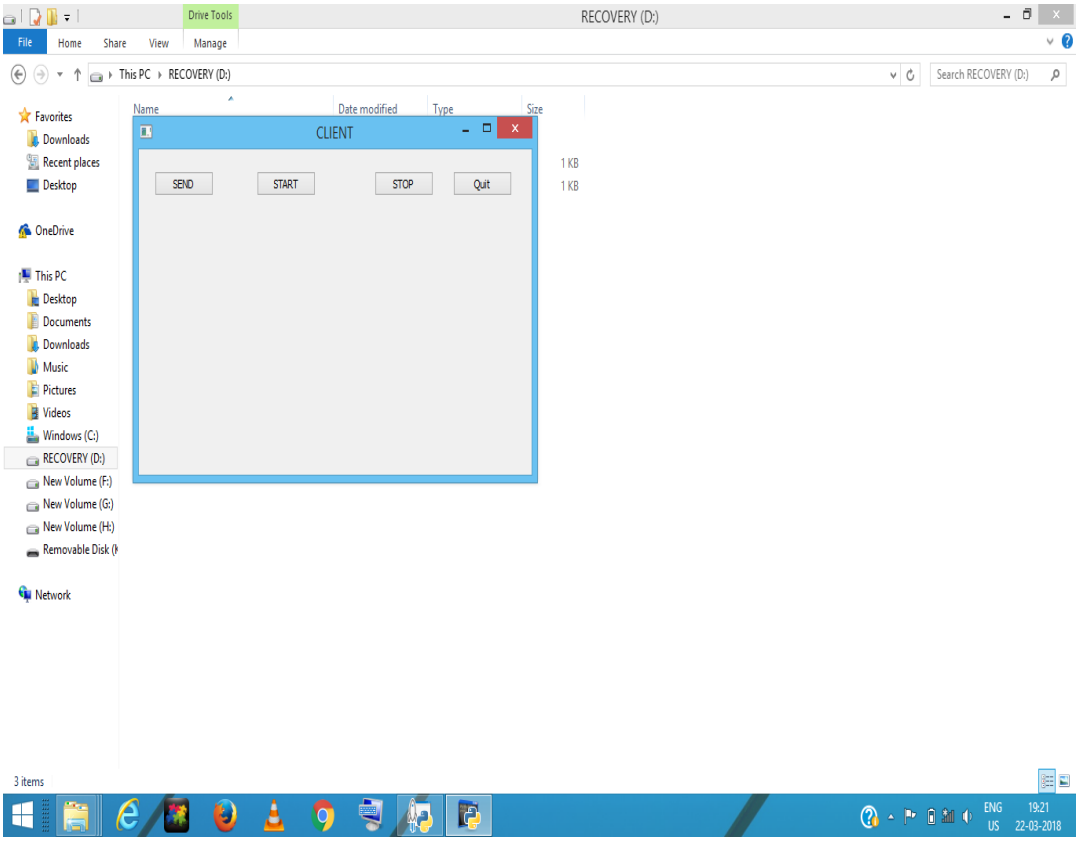

Fig. 5.7 Client application online for multiple image

5.3.3 After Receiving Multiple Images

As soon as client clicks on start button, server receives this image one by one and display this image on GUI. To get better result, multiple image are made overlapped on each other one by one to get show result as video streaming.

Below image shows GUI on server side which consist of four buttons i.e. Connect, Start, Stop, Multi and Refresh.

- Connect: Connect button is used to start Socket connection with specific client on server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the incoming connection.
- Multi: Multi Button is used to broadcast images.
- Refresh: Refresh button is used for update the connected clients.

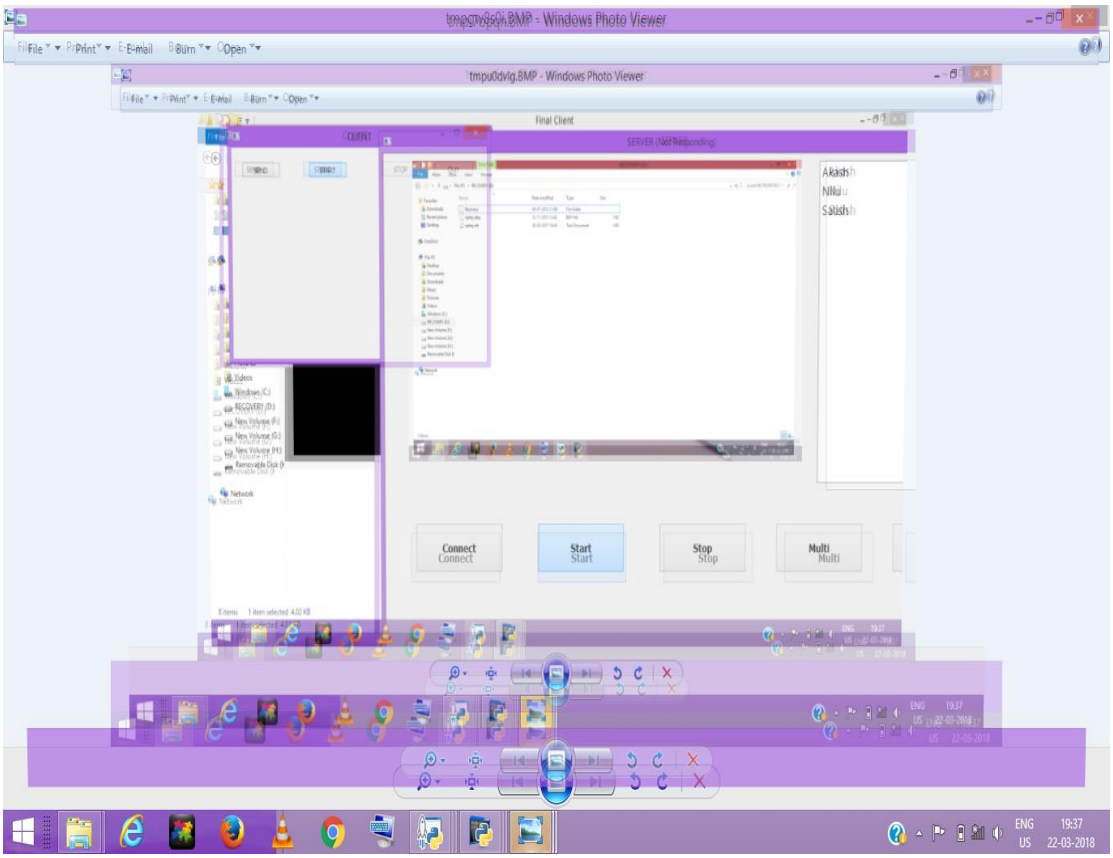

Fig. 5.8 Receiving multiple images

5.4 Broadcast

In computer networking, broadcasting refers to transmitting a packet that will be received by every device on the network. In practice, the scope of the broadcast is limited to a broadcast domain. Broadcasting a message is in contrast to unicast addressing in which a host sends datagrams to another single host identified by a unique IP address.

Broadcasting is the most general communication method, and is also the most intensive in the sense that a large number of messages are required

If user at server side wants to share its desktop screen to multiple users for presentation or other purpose, broadcast method helps to carry out such operation. In broadcast method, one server sends message to multiple client at same time.

5.4.1 Server

When client wants to initiate broadcasting feature, he clicks on multi button. Below image shows GUI on server side which consist of four buttons i.e. Connect, Start, Stop, Multi and Refresh.

- Connect: Connect button is used to start Socket connection with specific client on server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the incoming connection.
- Multi: Multi Button is used to broadcast images.
- Refresh: Refresh button is used for update the connected clients.

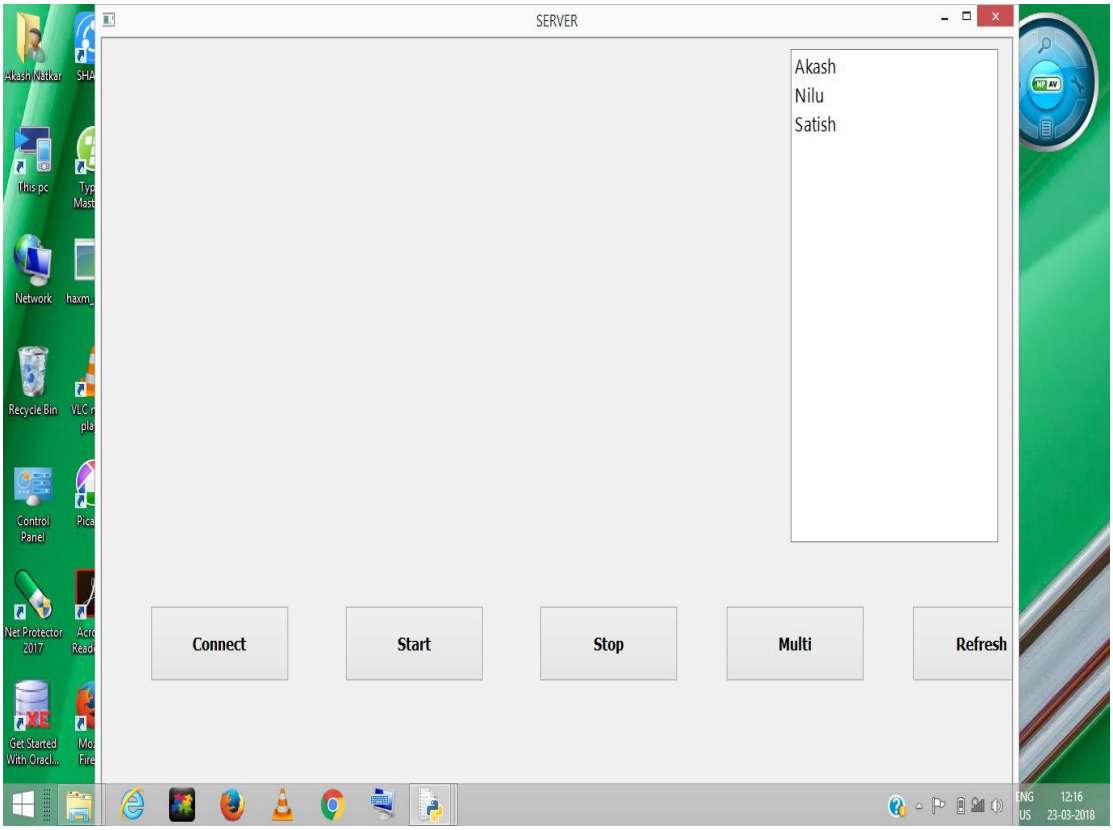

Fig. 5.9 Server application online

5.4.2 Client

To see the presentation on client side, user at client side clicks on multi-client.py application. This application differs from previous application because it displays image on client side.

When user starts this application, connection is established with the server. As soon as connection is established, image gets received at client side. Below image shows GUI on client side which consist of four buttons i.e. Send, Start and Stop.

- Send: Send button is used to send image to server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the connection.

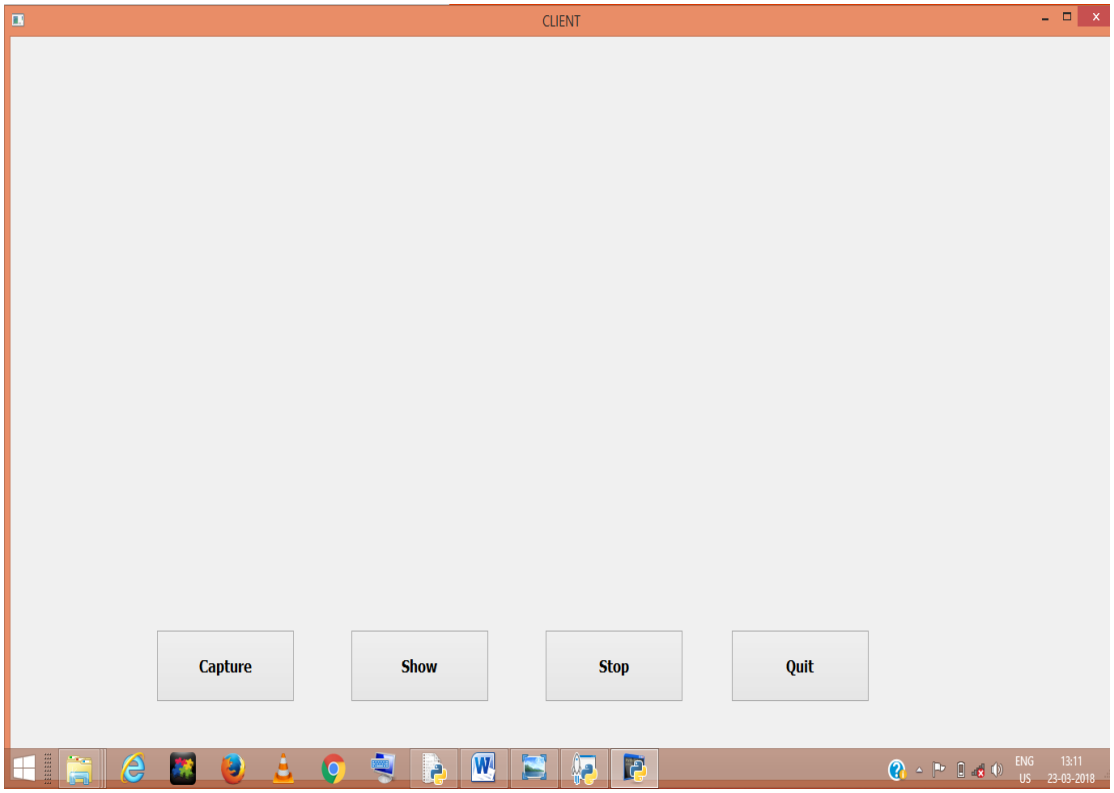

Fig. 5.10 Client application online

5.4.3 With Image

A QLabel object acts as a placeholder to display non-editable text or image, or a movie of animated GIF. It can also be used as a mnemonic key for other widgets. Plain text, hyperlink or rich text can be displayed on the label. To display this image on Client side, user clicks on Show button.

Below image shows GUI on server side which consist of four buttons i.e. Connect, Start, Stop, Multi and Refresh.

- Connect: Connect button is used to start Socket connection with specific client on server side.
- Start: Start Button is used to send/receive multiple image. The images are send/received at certain interval of time.
- Stop: Stop Button is used to stop the incoming connection.
- Multi: Multi Button is used to broadcast images.
- Refresh: Refresh button is used for update the connected clients.

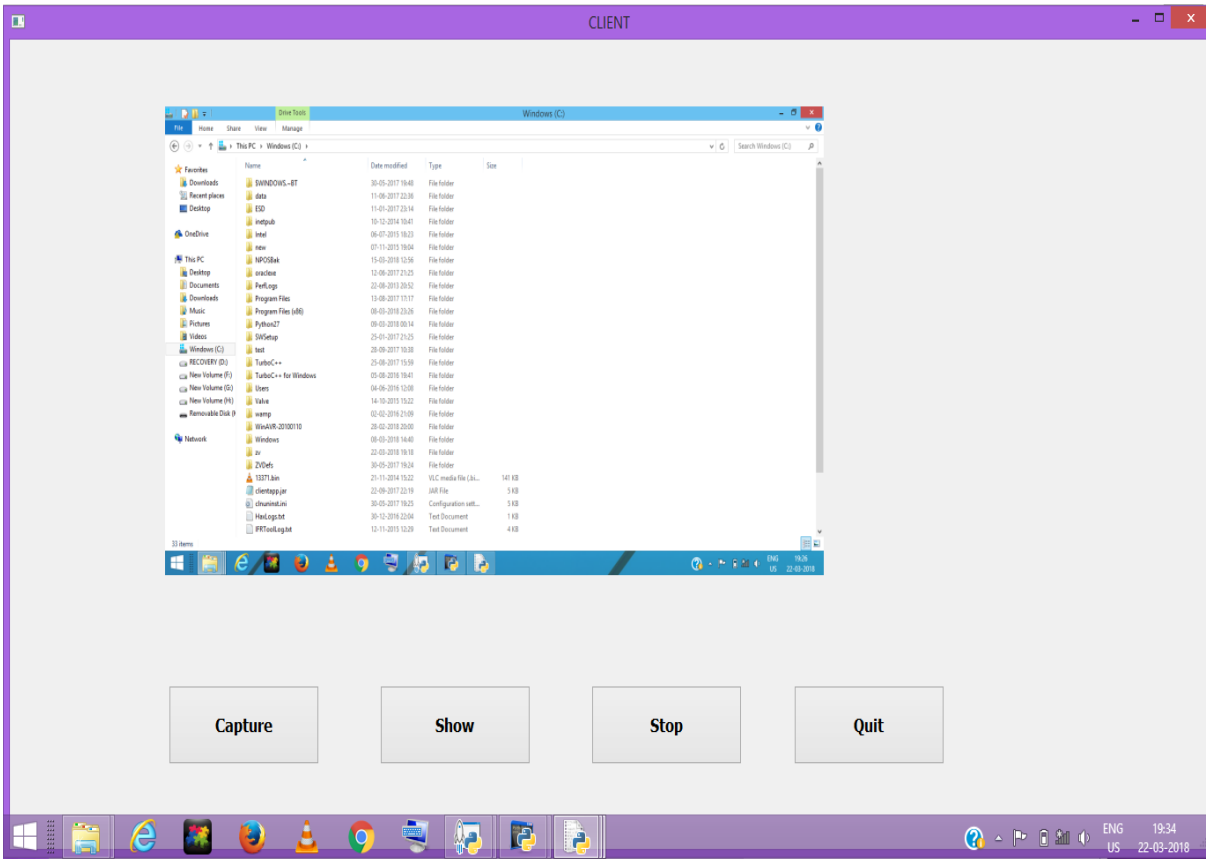

Fig. 5.11 Broadcast successful

CHAPTER 6

CONCLUSION

Desktop sharing application allows the server to monitor the client activity. This application may be used for various purposes like monitor child activity, to monitor the employee's activity and recognizing and monitor threads and present and investigation criminals' activity. This application has features like platform independency and online message passing. This application may be used by government corporations, anti-criminal organization and by individuals.

6.1 Limitations of the Study

Every project has got some limitations. This sharing application has also some limitations. Without concern of an individual their activities can be monitor so, their privacy may get hampered. This application allows only client activity monitoring. It does not allow the whole control access over the client side. So, drawback of this project is that it won't stop ongoing unwanted activity.

6.2 Future Scope of Work

Every project has future scope for further development. To overcome the limitations of this desktop sharing application this project can be extended in future. The privacy issue suffered by client may be overcome by providing appropriate security factor and encryption.

More users it can be extended to record the client's activity that is may be used for video record and also for voice record. In future this application may be used for network surveillance also. The limitations of not accessing client side may be overcome i.e. in future access provision may be given to server to get full access of client side providing appropriate security factor.

References

1. Daniel Zinca (2010). “Design of a modified RFB protocol and its implementation in an ultra - thin client”, *Ninth International Symposium on Electronics and Telecommunications*, Timisoara, Romania.
2. Mark Lutz (2013). “Learning Python”, *O'reilly*, Fifth Edition, ISBN: 978-1-449-35573-9.
3. http://vncdotool.readthedocs.io/en/latest/rfbproto.html#server-to-client-message accessed on 27/06/2017 at 8.35p.m.
4. http://www.realvnc.com/docs/rfbproto.pdf accessed on 10/07/2017 at 9.00 p.m.
5. http://www.techbeamers.com/python-tutorial-write-multithreaded-python-server/ accessed on 19/12/2017 at 9.55 p.m.
6. http://schurpf.com/python/python-hotkey-module/pyhk-end-user-documentation/ accessed on 21/12/2017 at 6.05 p.m.
7. http://zetcode.com/db/sqlitepythontutorial/ accessed on 28/12/2017 at 7.20 p.m.
8. https://www.tutorialspoint.com/python/python_files_io.htm accessed on 30/12/17 at 8 p.m.
9. http://pymbook.readthedocs.io/en/latest/file.html accessed on 10/01/2018 at 5.30 p.m.
10. http://euanfreeman.co.uk/pyqt-qpixmap-and-threads/ accessed on 12/02/2018 at 9.30 p.m.