Chatting Application

A PROJECT REPORT SUBMITTED

by

Rajat Saxena

University Roll No – 2000290140097

Submitted in partial fulfillment of the

Requirements for the Degree of

Master of Computer Application

Under the Supervision of

Mr. Ankit Verma

Assistant Professor

Submitted to

Faculty of MCA

DR. APJ ABDUL KALAM TECHNICAL UNIVERSITY
LUCKNOW
(Formerly Uttar Pradesh Technical University, Lucknow)

(JAN 2022)

STUDENT DECLARATION

We declare that minor project entitled **"CHATTING APPLICATION"** is our own work conducted under the supervision of **"Mr. Ankit Verma"**, Department of Master of Computer Application, **KIET Group of Institutions, Ghaziabad.**

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

Date: **Signature of Student**

RAJAT SAXENA (2000290140097)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Signature of HOD

Prof (Dr.) Ajay Shrivastava

HOD-MCA
Date:

Signature of Internal

Dr. Arun Tripathi

Associate Professor

ACKNOWLEDGEMENT

Any software project is not a work of an individual. It combines efforts, ideas, suggestions, reviews and hard work. We express our sincere gratitude to all those who initiated and helped us in the successful completion of our project. One of the most pleasing aspect in collecting the necessary information and compiling it is the opportunity to thanks those who have actively contributed to it.

I take this occasion to thank God, almighty for blessing us with his grace and taking our endeavor to a successful culmination. I extend my sincere and heartfelt thanks to our esteemed guide, **Mr. Ankit Verma**, for providing me with the right guidance and advice at the crucial junctures and for showing me the right way.

I'm highly indebted to **Prof (Dr.) Ajay Shrivastava (HOD- MCA)** for his continuous effort in building a good infrastructure and develop a professional attitude within ourselves during the academic period of MCA.

I would like to thank the other faculty members also, at this occasion. Last but not the least, I would like to thank my friends and family for the support and encouragement they have given me during our work.

Signature of Student

RAJAT SAXENA

(2000290140097)

MCA-III Semester

CERTIFICATE

Certified that **RAJAT SAXENA (2000290140097)** have carried out the project work having "**CHATTING APPLICATION"** for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself / herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

RAJAT SAXENA (2000290140097)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Mr. Ankit Verma

Assistant Professor

Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Signature of Internal Examiner **Signature of External Examiner**

Dr. Ajay Shrivastava
Head, Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Abstract

Messaging apps now have more global users than traditional social networks—which means they will play an increasingly important role in the distribution of digital journalism in the future. Drawing upon our interviews and case studies, we identify a number of opportunities and challenges for organizations using—or hoping to use—messaging apps for news.

This project is a web based chat application for the employees of an office. The objective of the project is to provide an easy way of communication between the users of the application.

Teleconferencing or chatting refers to any kind of communication that offers a real-time transmission of messages from sender to the receiver.

Chatting is a method of using technology to bring people and ideas together despite the geographical barriers. The technology to provide the chatting facility has been available for years, but the acceptance is quite recent.

Analysis of chatting provides an overview of the technologies used, available features, functions, system of the architecture of the application, the structure of database of an Instant Messaging application: IChat (IC). The objective of IC application is to facilitate text messaging, group chatting option, data transfer without size restriction which is commonly seen in most of the messaging applications.

Chatting Application is a social-networking tool that leverages on technology advancement thereby allowing its users communicate and share media. It offers a wonderful one stop shop experience for keeping in touch with people you know.

TABLE OF CONTENT

INTRODUCTION

1.1Introduction:

Chatting Application is a Desktop based application.

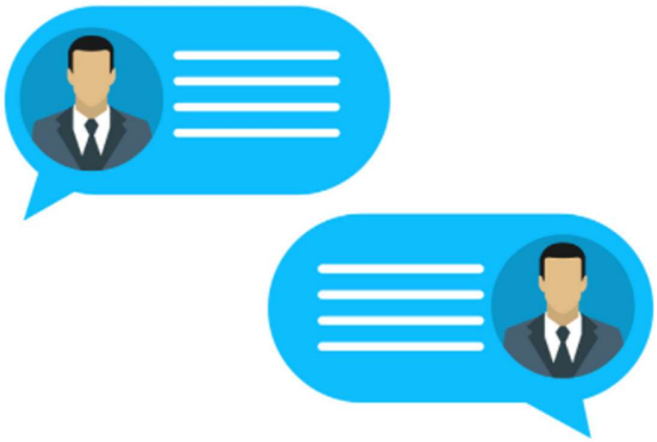
This client server chat application is based on java swing and used socket package. It's simple and easy and require only core java knowledge. I have taken this program from internet and modified a little bit to make it simpler and more elegant.

This application/program is a good example of using java.io, java.net package to create a chat application. A beginner of java language, who is familiar with this packages can able, be beneficiate.

Chatting is a method of using technology to bring people and ideas "together" despite of the geographical barriers. The technology has been available for years but the acceptance it was quit recent. Our project is an example of a multiple client chat server.

It is made up of 2 applications the client application, which runs on the user's Pc and server application, which runs on any Pc on the network. To start chatting client should get connected to server. We will focus on TCP and UDP socket connections which are a fundamental part of socket programming.

Keywords: sockets, client-server, Java network programming-socket functions, Multicasting etc.



1.2 System Objective

Communication over a network is one field where this tool finds wide ranging application. Chat application establishes a connection between 2 or more systems connected over an intra-net or ad-hoc.

This tool can be used for large scale communication and conferencing in an organization or campus of vast size, thus increasing the standard of co-operation. In addition, it converts the complex concept of sockets to a user-friendly environment. This application also provides facility to share files.

1.3Sockets Overview

A socket is an object that represents a low-level access point to the IP stack. This socket can be opened or closed or one of a set number of intermediate states.

A socket can send and receive data down disconnection. Data is generally sent in blocks of few kilobytes at a time for efficiency; each of these blocks are called a packet.

All packets that travel on the internet must use the Internet Protocol. This means that the source IP address, destination address must be included in the packet. Most packets also contain a port number.

A port is simply a number between 1 and 65,535 that is used to differentiate higher protocols. Ports are important when it comes to programming your own network applications because no two applications can use the same port.

Packets that contain port numbers come in two flavours: UDP and TCP/IP. UDP has lower latency than TCP/IP, especially on startup. Where data integrity is not of the utmost concerned, UDP can prove easier to use than TCP, but it should never be used where data

integrity is more important than performance; however, data sent by UDP can sometimes arrive in the wrong order and be effectively useless to the receiver.

TCP/IP is more complex than UDP and has generally longer latencies, but it does guarantee that data does not become corrupted when travelling over the internet. TCP is ideal for file transfer, where a corrupt file is more unacceptable than a slow download; however, it is unsuited to internet radio, where the odd sound out of place is more acceptable than long gaps of silence.

1.4 Design Approach

The application has been designed using :

1.4.1 Programming language Java

Android applications are developed using the Java language. As of now, that's really your only option for native applications. Java is a very popular programming language developed by Sun Microsystems (now owned by Oracle). Developed long after C and C++, Java incorporates many of the powerful features of those powerful languages while addressing some of their drawbacks. Still, programming languages are only as powerful as their libraries. These libraries exist to help developers build applications.

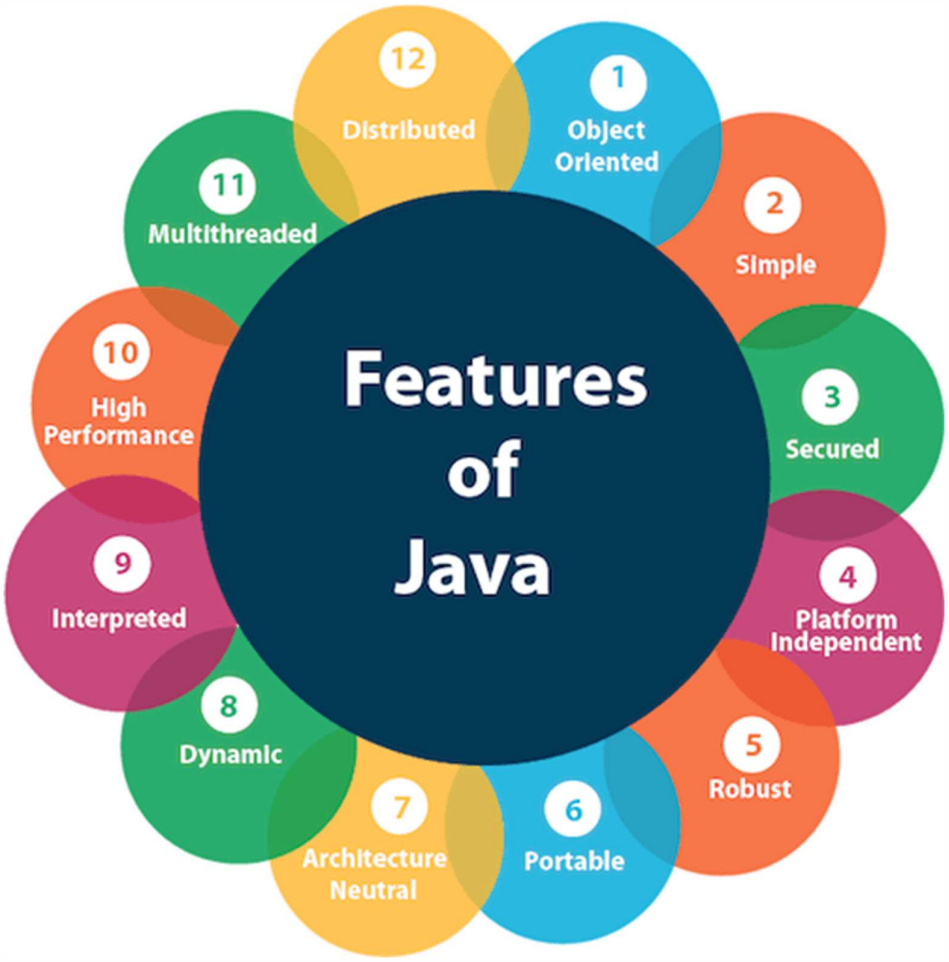
Some of the Java's important core features are:

- It's easy to learn and understand

- It’s designed to be platform-independent and secure, using virtual machines
- It’s object-oriented

Android relies heavily on these Java fundamentals. The Android SDK includes many standard Java libraries (data structure libraries, math libraries, graphics libraries, networking libraries and everything else you could want) as well as special Android libraries that will help you develop awesome Android applications.

Java Features:



1. Platform Independent: The Write-Once-Run-Anywhere ideal has not been achieved (tuning for different platforms usually required), but closer than with other languages.

2. Object Oriented : Java is object oriented throughout i.e. there is no coding outside of class definitions, including main(). There is an extensive class library available in the core language packages.

3. Robust: Exception handling built-in, strong type checking (that is, all data must be declared an explicit type), local variables must be initialized.

4. Automatic Memory Management: Automatic garbage collection - memory management handled by JVM

5. Security:

- No memory pointers.
- Programs run inside the virtual machine sandbox.
- Array index limit checking

6. Good Performance: Interpretation of byte codes slowed performance in early versions, but advanced virtual machines with adaptive and just-in-time compilation and other techniques now typically provide performance up to 50% to 100% the speed of C++ programs.

7. Dynamic Binding: The linking of data and methods to where they are located is done at run-time.

8. Threading: Lightweight processes, called threads, can easily be spun off to perform multiprocessing.

9. Built-in Networking: Java was designed with networking in mind and comes with many classes to develop sophisticated Internet communications [8].

Java package:

1.1 Java Packages & API

A package in Java is used to group related classes. Think of it as **a folder in a file directory**. We use packages to avoid name conflicts, and to write a better maintainable code. Packages are divided into two categories:

- Built-in Packages (packages from the Java API)
- User-defined Packages (create your own packages)

1.2 Built-in Packages

The Java API is a library of prewritten classes, that are free to use, included in the Java Development Environment.

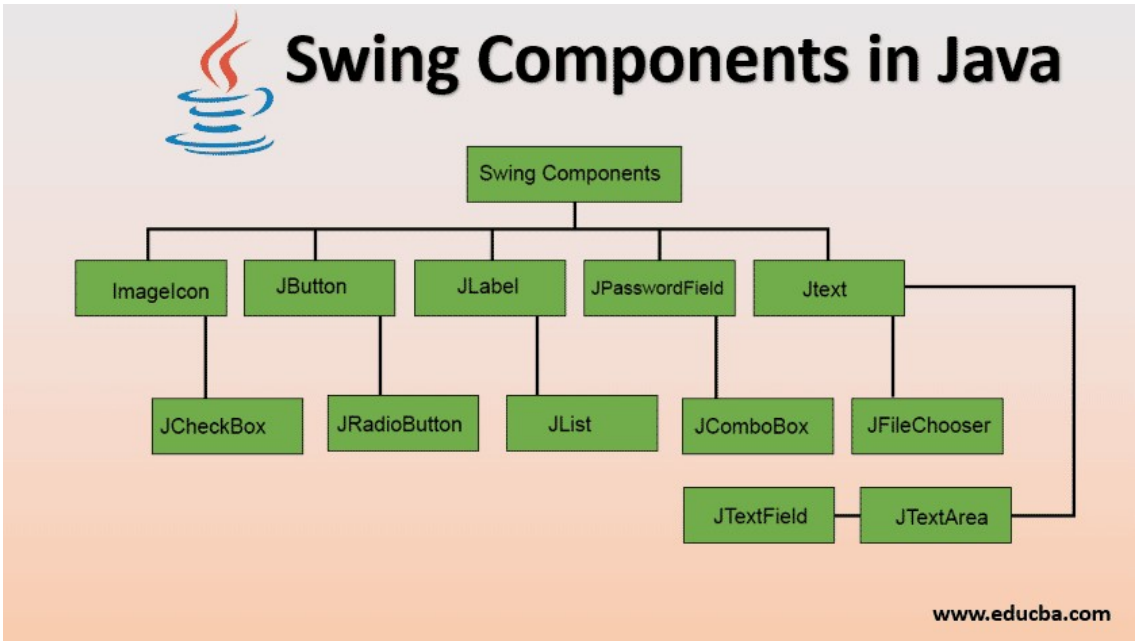
The library contains components for managing input, database programming, and much much more. The complete list can be found at Oracles website: https://docs.oracle.com/javase/8/docs/api/.

The library is divided into **packages** and **classes**. Meaning you can either import a single class (along with its methods and attributes), or a whole package that contain all the classes that belong to the specified package.

To use a class or a package from the library, you need to use the import keyword:

```
package chatting.application;
import javax.swing.*;
import javax.swing.border.*;
import java.awt.event.*;
import java.awt.*;
import java.net.*;
import java.io.*;
import java.util.Calendar;
import java.text.SimpleDateFormat;
```

1.4.2Swing API



Java Swing is a part of Java Foundation Classes (JFC) which was designed for enabling large-scale enterprise development of Java applications.

Java Swing is a set of APIs that provides the graphical user interface (GUI) for Java programs. Java Swing is also known as the Java GUI widget toolkit.

Java Swing or Swing was developed based on earlier APIs called Abstract Windows Toolkit (AWT). Swing provides richer and more sophisticated GUI components than AWT.

The GUI components are ranging from a simple label to a complex tree and table. Besides emulating the look and feel of various platforms, Swing also provides *a pluggable look and feel* to allow the look and feel of Java programs independent from the underlying platform.

Swing Architecture

Swing is a platform-independent and enhanced MVC (Model –View – Controller) framework for Java applications. Here are the most important features in Swing architecture.

- **Pluggable look and feel**: Swing supports several looks and feels and currently supports Windows, UNIX, Motif, and native Java metal look and feel. Swing allows users to switch look and feel at runtime without restarting the application. By doing this, users can make their own choice to choose which look and feel is the best for them instantly.

- **Lightweight components**: All swing components are lightweight except for some top-level containers. Lightweight means the component renders or paints itself using drawing primitives of the Graphics object instead of relying on the host operating system (OS). As a result, the application presentation is rendered faster and consumed less memory than previous Java GUI applications.

- **Simplified MVC**: Swing uses simplified model-view-architecture (MVC) as the core design behind each component called model-delegate. Based on this architecture, each swing component contains a model and a UI delegate. A UI delegate

wraps a view and a controller in MVC architecture, as in the picture below. UI delegate is responsible for painting screens and handling GUI events. Model is in charge of maintaining information or states of the component.

1.4.3Socket Programming

A *socket* is a communications connection point (endpoint) that you can name and address in a network. Socket programming shows how to use socket APIs to establish communication links between remote and local processes.

The processes that use a socket can reside on the same system or different systems on different networks. Sockets are useful for both stand-alone and network applications. Sockets allow you to exchange information between processes on the same machine or across a network, distribute work to the most efficient machine, and they easily allow access to centralized data. Socket application program interfaces (APIs) are the network standard for TCP/IP. A wide range of operating systems support socket APIs. i5/OS sockets support multiple transport and networking protocols. Socket system functions and the socket network functions are threadsafe.

Programmers who use Integrated Language Environment® (ILE) C can refer to this topic collection to develop socket applications. You can also code to the sockets API from other ILE languages, such as RPG.

The Java™ language also supports a socket programming interface.

Java Environment:

Java environment includes large number of development tools and hundreds of classes and methods. The development tools are part of the system known as Java Development Kit (JDK)and the classes and methods are part of the Java Standard Library (JSL), also known as the Application Programming Interface

After this detail we summarize the project requirements from following software

1.4.4File handling

As mentioned, **file handling in Java** allows you to work with files. It is a Java functionality defined in the File class of the Java.io package. The io in Java.io package stands for Input and Output.

You will find all the classes you will ever need to perform any input or output operations in the Java.io package.

You can access and use this File class to perform various functions by creating an object and providing the file or directory name. The syntax to create an object and use the File class is:

import java.io.File;

File obj = new File("name.txt");

As seen in the above syntax, you first need to import the File class from the Java.io package. Once imported, all you need to do is create an object and specify the file or directory name you want to work with.

To mention the directory name in a Windows OS, you have to use "\\." For instance, you have to use
"**C:\\Users\\MyFolder\\MySubFolder**" and not
"**C:\Users|MyFolder|MySubFolder**."

However, for other operating systems such as Linux, you can use the single \ as usual.

All the I/O (Input/Output) operations of the **file concepts in Java** are done with the help of a stream. Let's delve deep into what is a stream, and how are these operations performed with it?

```
public void sendTextToFile(String message) throws FileNotFoundException{

        try{

            File out = new File("chat.txt");
           FileWriter writer = new FileWriter(out,true);

            writer.write("Chirag:"+message);
            writer.write("\n");
              writer.flush();
           System.out.println("Data successfully written in the specified file");
           }catch(Exception e)
        {
           System.out.println("Exception:" +e);
        }
    }
```

1.4.5 Socket.IO

Socket.IO is a library that enables real-time, bidirectional and event-based communication between the browser and the server.

This application uses Socket.IO which enables real-time bidirectional event-based communication. Socket.IO is built on top of the Web Sockets API (Client side) and Server side.

1.4.6 API

The APIs for developing applications using the DHTML client in DB2 Alphablox are available on the server-side, where a developer accesses them through Java™ calls (for example, in a Java scriptlet on a JSP page). The reason the Java APIs are called *server-side* APIs is because the code executes on the server before it is sent to the browser.

Executing code on the server is often more efficient, and also makes it easier to create web pages that work correctly on multiple browsers. The DHTML client is designed to keep the client and the server in sync without page refreshing. When you execute code on the server, only affected areas in the Blox UI is refreshed, not the whole page.

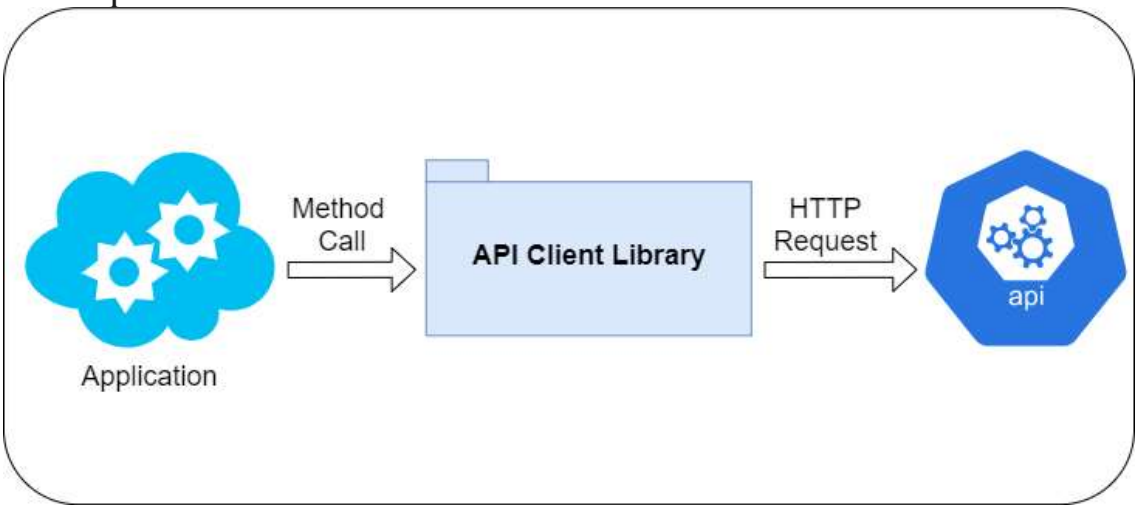
There are also times when you want to use the DHTML client's Client API for tasks that are best handled on the client. These are called client-side APIs because they are interpreted by the browsers. Often times you want to call some server-side code to change Blox properties on the server via some JavaScript™ code on the client when a user clicks a button or link on the page.

The DHTML client has a relatively straightforward API on the client-side.

Client API

When writing client-side JavaScript for web sites or applications, you will quickly encounter **Application Programming Interfaces** (**APIs**). APIs are programming features for manipulating different aspects of the browser and operating system the site is running on, or manipulating data from other web sites or services.

In this module, we will explore what APIs are, and how to use some of the most common APIs you'll come across often in your development work.



Server API

A server-side web API is a programmatic interface consisting of one or more publicly exposed endpoints to a defined request–response message system, typically expressed in JSON or XML, which is exposed via the web—most commonly by means of an HTTP-based web server.

Mashups are web applications which combine the use of multiple server-side web APIs. Webhooks are server-side web APIs that take input as a Uniform Resource Identifier (URI) that is designed to be used like a remote named pipe or a type of callback such that the server acts as a client to dereference the provided URI and trigger an event on another server which handles this event thus providing a type of peer-to-peer IPC.

1.5 Project Scope

This project can be mainly divided into two modules:

1. Server
2. Client

This project is mainly depended on client/server model. The client requests the server and server responses by granting the Clients request. The proposed system should provide both of the above features along with the followed ones:

Server:

A server is a computer program that provides services to other computer programs (and their users) in the same or other computers. The computer that a server program runs in is also frequently referred to as a server. That machine may be a dedicated server or used for other purposes as well. Example Server, Database, Dedicated, Fileserver, Proxy Server, Web Server. The server is always waiting for client's requests. The client come and go down but the server remains the same.

A server application normally listens to a specific port waiting for connection requests from a client. When a connection request arrives,

the client and the server establish a dedicated connection over which they can communicate.

During the connection process, the client is assigned a local port number, and binds a socket to it. The client talks to the server by writing to the socket and gets information from the server by reading from it. Similarly, the server gets a new local port number (it needs a new port number so that it can continue to listen for connection requests on the original port). The server also binds a socket to its local port and communicates with the client by reading from and writing to it. The client and the server must agree on a protocol that is, they must agree on the language of the information transferred back and forth through the socket. Normally, a server runs on a specific computer and has a socket that is bound to a specific port number. The server just waits, listening to the socket for a client to make a connection request.

Client:

On the client site the client knows the hostname of the machine on which the server is running and the port number on which the server is listening.
To make a connection request, the client tries to rendezvous with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number that it will use during this connection. This is usually assigned by the system.

The model used for this project is the single server following specifications must be implemented: multiple client models. The

1. The server and client are two separate programs.

2. Multiple clients must be able to connect to a single server.

3. All input and output is via I/O Interface (GUI is Required)

4. Client:

- Client must be able to choose a nickname on connection.

- Client must show when another client connects or disconnects with
  the server.

- Client must be able to send messages to the server before arriving in other client.

- Client must be able to receive messages while writing.

- Client must be able to print out any messages received from the
- server.

5. Server:

- Server must be able to print information in the event of the following cases: (connect, disconnect, send and receive messages).

- The server does not allow for more than one client to get the same nickname.
- Server must be able to return messages again to all clients (including source).

- Client connected and disconnected with the server does not crash the server.

1.6 Problem Specification

Represent online chat, a large area of the packet data that is exchanged between users of these web, but many users do not see the Internet only way to get to the other by means of various

communication offered by various Internet sites, such as your e-mail, forums, and programs Instant (chatting alternators), and chat sites, and although the chat, a means of communication are only a fraction of the possibilities that can be provided by the internet, but it is considered the main motive behind the connection of more than 25% of the users of the network.

Despite the effort made by Staff, to interact with the students during the lecture, but the time the lecture Limited, reduces the chance of participation and discussion with students.

So there pressing need for system Communication electronic between students and staff. The idea of creating chatting be the most important electronic means.

That help to enrich the discussion and communication, which will help to increase communication between students on the one hand, and between students and the university on the other hand, which will help to save the effort and time, and reduces the administrative burdens for staff member, contributing to the different viewpoints between the students.

1.7 Hardware / Software used in Project

This section describes the software and hardware requirements of the system.

Software requirements

Software's can be defined as programs which run on our computer .It act as petrol in the vehicle . It provide the relationship between the human and a computer .It is very important to run software to

function the computer. Various software's are needed in this project for its development.

Which are as follows--?

Operating System : Window 7 or later

Software : NetBeans IDE 6.5.1

- Language: Java
- Front-end design: Html, CSS, JavaScript, Php, MySQL
- Database: File Handling
- Socket Programming

Hardware requirements

In hardware requirement we require all those component which will provide us the plate form for the development of the project. The minimum hardware required for the development this project as follows-

- System operation: Windows 7 or later
- Ram: 2 GB
- Hardware capacity: 2 GB
- Processor: Intel Core i3

Chapter 2: System Study

2.1 Existing System Study

Currently there Is no so such system available. The users earlier need to login into official website .Therefore the need for a chat system facility was necessary.

2.2 System Analysis

System analysis is the way of studying a system with an eye on solving its problem using computer. It is the most essential part of the development of a project of a system analysis. System analysis consists of system element, process and technology. To analyze a system, has to study the systems in details.

The analyst has to understand the functioning and concept of the system in detail, before design the appropriate computer based system that will meet all the requirements of the existing system. The system analyst has to carry out a customary approach to use the computer for problem solving.

The above steps constitute the logical framework for the system analysis.
After the preliminary investigation and feasibility study, the scope of the defined and comparable items are set forth and hence detailed investigation is executed. This allows the system analyst to comprehend the full scope of the project. Soon after the implementation of the newly developed system, followed by the training of the users, the system analysis is included.

PRELIMINARY INVESTIGATION:

A request to receive assistance from information system can be made for many reasons, but in case a manager, employee or system specialist initiates the request. When that request is made, the first system activity preliminary investigation begins. The activity has three parts.

Request clarification: -

the request from employee may not be well stated. Sometimes the request may not be well defined. Therefore before any system. investigation can be considered, the project request must be examined to determine precisely the actual requirements of the organization.

2.3 Feasibility study

the basic idea of feasibility study is to determine whether the requested project is feasible.

Request approval:

all projects that are requested are not desirable or feasible .some organization receive so many projects requests from employee that only a few of them can be pursued. However those projects that are feasible and desirable should put into a schedule. The management decides request that are most important. After a project request is

approved the cost priority, the completion time and the personal required are estimated. Once the request is approved, the collection of data and determination of requirements can be started.

NEEDS FOR FEASIBILITY STUDY:

The feasibility study is needed for following things:

- Answer the questions whether new system is to be installed or not?
- Determine the potential of the existing system.
- Improve the existing system.
- Know what should be embedded in the new system.
- Define the problems and objectives involved.
- Avoid costly repairs at later stage when system is implemented.
- Avoid crash implementation of the new system.
- Avoid the xHardware approachx i.e. getting a computer first and then deciding how to use it.

STEPS IN FEASIBILITY ANALYSIS:

1. Form a project team and appoint a project leader.

2. Prepare system flowcharts.

3. Enumerate potential candidate system.

4. Describe and identify characteristics of candidate system.

5. Determine and evaluate performance and cost effectiveness of each candidate system.

6. Weight system performance and cost data.

7. Select the best candidate system.

8. Prepare and report final project directive to management[6].

2.3.1 ECONOMIC FEASIBILITY

Economic analysis is most frequently used for evaluation of the effectiveness of the system. More commonly knows as cost/benefit analysis, the procedure is to determine the benefit and saving that are expected from a system and compare them with costs, decisions is made to design and implement the system.

This part of feasibility study gives the top management the economic Justification for the new system. This is an important input to the management the management, because very often the top management does not like to get confounded by the various technicalities that bound to be associated with a project of this kind.

A simple economic analysis that gives the actual comparison of costs onomic and benefits is much more meaningful in such cases.

In the system, the organization is most satisfied by economic

feasibility.

Because, if the organization implements this system, it need not

require any additional hardware resources as well as it will be saving

lot of time.

2.3.2 TECHNICAL FEASIBILITY

Technical feasibility centers on the existing manual system of the test management process and to what extent it can support the system.

According to feasibility analysis procedure the technical feasibility of the system is analyzed and the technical requirements such as software facilities, procedure, inputs are identified. It is also one of the important phases of the system development activities.

The system offers greater levels of user friendliness combined with greater processing speed. Therefore, the cost of maintenance can be reduced.

Since, processing speed is very high and the work is reduced in the maintenance point of view management convince that the project is operationally feasible.

2.3.3 BEHAVIOURAL FEASIBILITY:

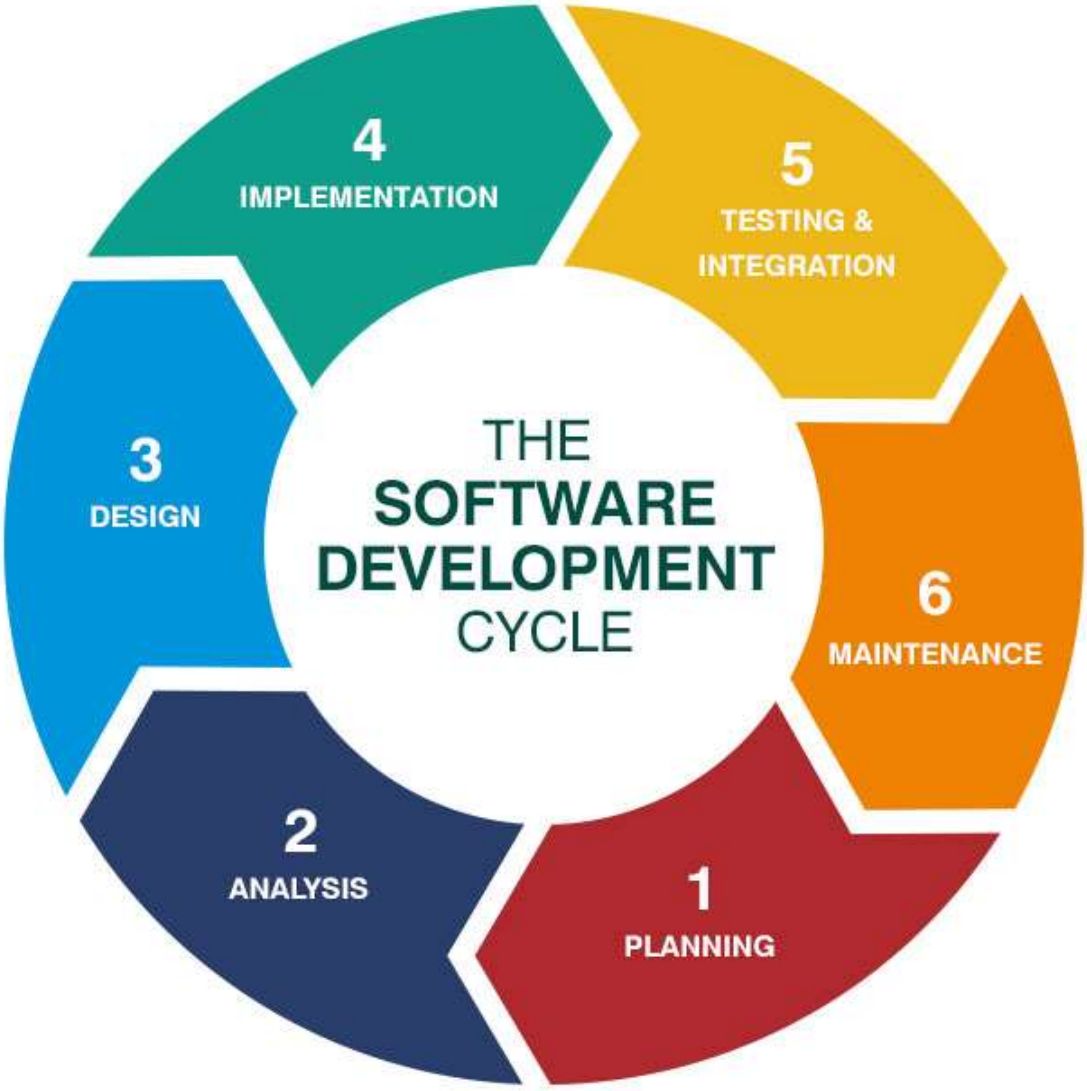
People are inherently resistant to change and computer has been known to facilitate changes. An estimate should be made of how strong the user is likely to move towards the development of computerized system.

These are various levels of users in order to ensure proper authentication and authorization and security of sensitive data of the organization.

2.4 Methodology (SDLC)

The Systems Development Life Cycle (SDLC):

Like Nitrogen & Production cycle ± a Software system is just like a cycle which passes the following stages like!



1. Planning phase.

2. Analysis phase.

3. Design phase.

4. Development phase.

5. Testing Phase.

6. Implementation & Maintenance phase.

The planning phase:

It is the process is to understand why should the system should be built and determine its requirements. It also includes a feasibility study from the different perspectives and the technical and economic, and feasibility aspects of the organization[11].

Through our project, has been studying technical requirements of project from hardware and software.

These requirements have discussed in chapter1. Since our will install on devices the university, will there is no additional cost on the project. also software required him Free.

The analysis phase:

This phase includes activities such determine and analysis problems, and even forecasting potential problems that may arise in future with regard the system. The deliverables / products of this phase will drive how the system will be built and guide the developers' works[11].

The basic requirements for this project can be decided according to the needs of the users who will use this application. Some basic requirements have been listed here:

Access to the project should be protected from unauthorized users. So, any attempt to access the project must go through some login process. It is for both types of users, that Is administrators and common chat users.

The project must provide user interfaces to create a new user, search a specific user, update the information of any existing user, and delete any existing user.

The project must provide an interface having a list of members of Instant Chat. The application must also show the online or offline status of the existing users.

Only administrators must have a right to maintain the list of existing members of Instant Chat by adding or deleting members from it.

The design phase:

It is the most creative and challenging phase of system development. It deals with converting input into output. It contains output design, input design, file or database design and processing design[11]. The user interfaces of this project are categorized into various modules. These modules are:

- Admin Module
- User Module

The development phase:

The development phase involves writing the source code based on the required functionality adhering the coding standards, code optimization, etc. It takes its primary input from the design elements described in the software design phase[12]. The development of this project involves creating windows forms, class files, and user controls in Java. The code files are covering: connecting to Mysql Server database, sending requests lo the server and responses to the clients,

executing queries. firing trigger, filtering data, and reflecting changes in a database.

The Testing Phase:

After the designing and developing phases, the application is tested for any logical flaws and functionality of all operations[12]. The project is also tested lo ensure that all methods and modules designed and developed are functioning properly, along with the navigation links provided in all the user interfaces, finally, the project is tested to ensure that all the requirements listed during the requirements analysis phase are being fulfilled.

The Implementation and Maintenance Phase:

Finally, the project Is Implemented In a distributed environment, where it is used by users logged In from different computer nodes on a network [12]. The project is maintained thereafter if any requests for changes are forwarded by the users.

Chapter 3: Database

In this chatting application I use File Handling for storing our chat in file.

3.1 File Handling

File handling in Java is defined as reading and writing data to a file. The particular file class from the package called **java.io** allows us to handle and work with different formats of files.

Thus, if we want to use a file class, we need to create an object of that particular class and should specify the filename or directory name.

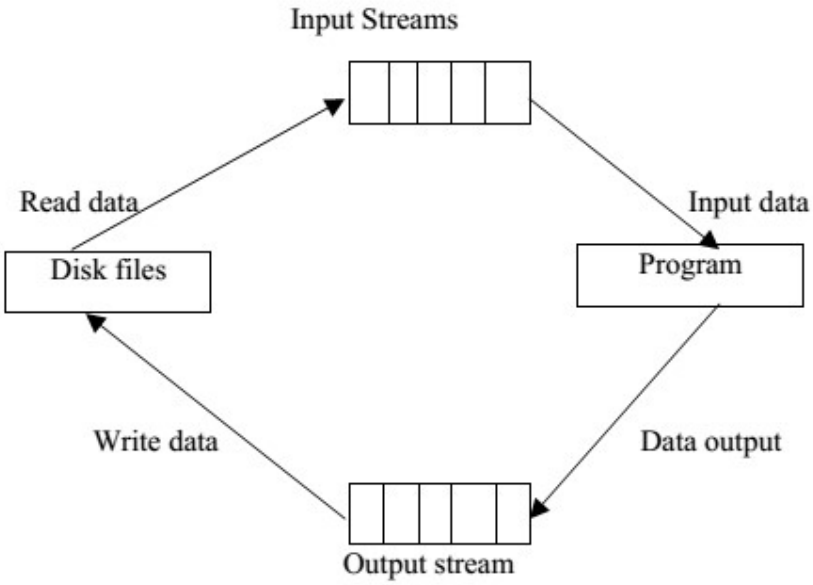
3.1.1 Streams

As discussed above, the package that supports stream input/output is java.io, and it is vast. This package defines over seventy classes and interfaces, many of which have a large number of methods.

- It will be able to give you a chance to read files from the data.
- It will give us a chance to create a formatted line for the command.
- It will give us the chance to read or write basic data.
- It will give us a chance to read or write files containing objects.

3.1.2 Understanding Streams

The stream is the only representation of input or output that is maybe a source or destination of the data. We can also write the data in the stream or read the particular data from the stream. We can also visualize the stream of data as a sequence of bytes that flow out of the program.



File Input and output stream

3.1.3 Input and Output Streams

Writing data in the stream or to a stream is called **an output stream.** The particular output stream can go to any device which can connect through a hard disk or maybe any stream which can contain the sequence of bytes. An output stream also can be displayed on any output screen, which has its true capability. Stream output to your display is output to your command line. When you start writing something for your display screen, it can only display the characters but not the graphical content. Graphical output requires more specialized support.

You read data from an **input stream**. In principle, this can only be any source of serial data but is typically a disk file, the keyboard, or a remote computer.

Further, under normal circumstances, the file input and output you write in a machine can only be through a java application. It's not available to java apples except strictly to a limited extent.

This has two advantages: First, you don't have to worry about the detailed mechanics of each device, which are taken care of behind the scenes.

The streams which you write under the input and output can only have a small amount of data like a single character, but not more than that. Transferring data to or from a stream like this may be extremely inefficient, so a stream often equipped with a **buffer** in memory, in which case it is called a **buffered stream**.

3.2 DFD

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

It shows how data enters and leaves the system, what changes the information, and where data is stored.

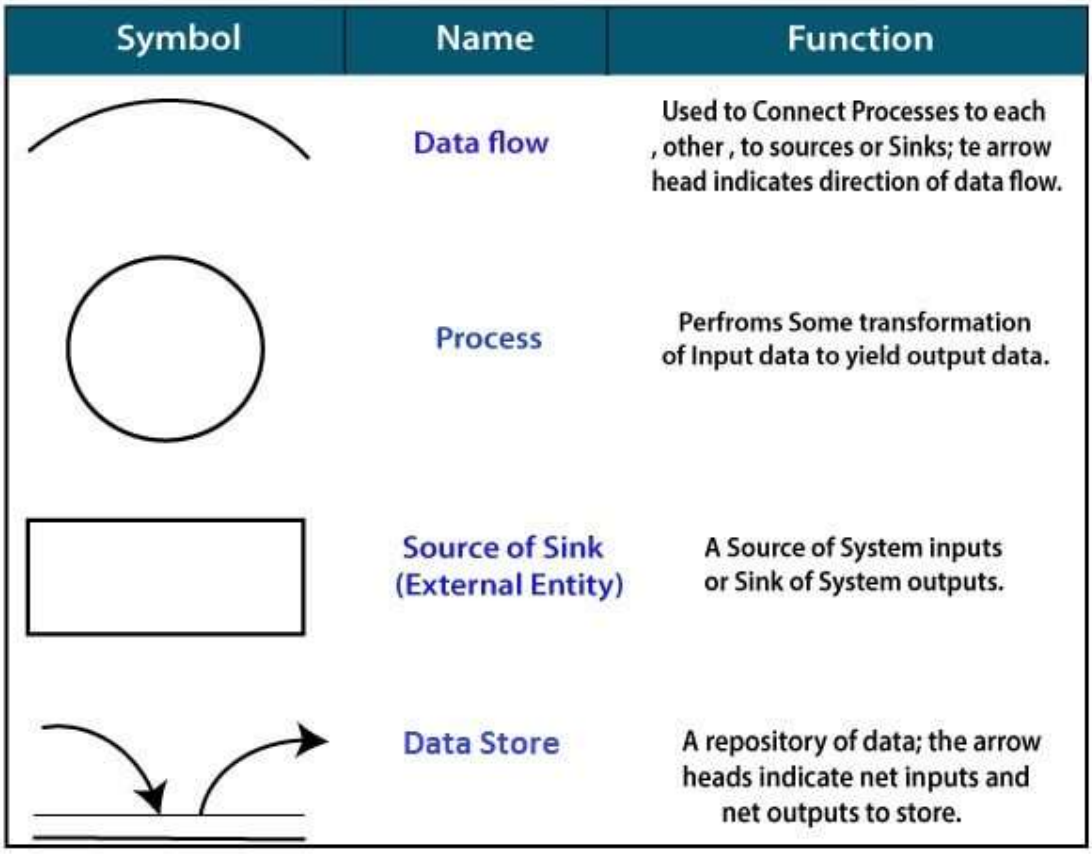
The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

The following observations about DFDs are essential:

1. All names should be unique. This makes it easier to refer to elements in the DFD.
2. Remember that DFD is not a flow chart. Arrows is a flow chart that represents the order of events; arrows in DFD represents flowing data. A DFD does not involve any order of events.
3. Suppress logical decisions. If we ever have the urge to draw a diamond-shaped box in a DFD, suppress that urge! A diamond-shaped box is used in flow charts to represents decision points with multiple exists paths of which the only one is taken. This implies an ordering of events, which makes no sense in a DFD.

4. Do not become bogged down with details. Defer error conditions and error handling until the end of the analysis.

Standard symbols for DFDs are derived from the electric circuit diagram analysis and are shown in fig:



Symbols for Data Flow Diagrams

Circle: A circle (bubble) shows a process that transforms data inputs into data outputs.

Data Flow: A curved line shows the flow of data into or out of a process or data store.

Data Store: A set of parallel lines shows a place for the collection of data items. A data store indicates that the data is stored which can be

used at a later stage or by the other processes in a different order. The data store can have an element or group of elements.

Source or Sink: Source or Sink is an external entity and acts as a source of system inputs or sink of system outputs.

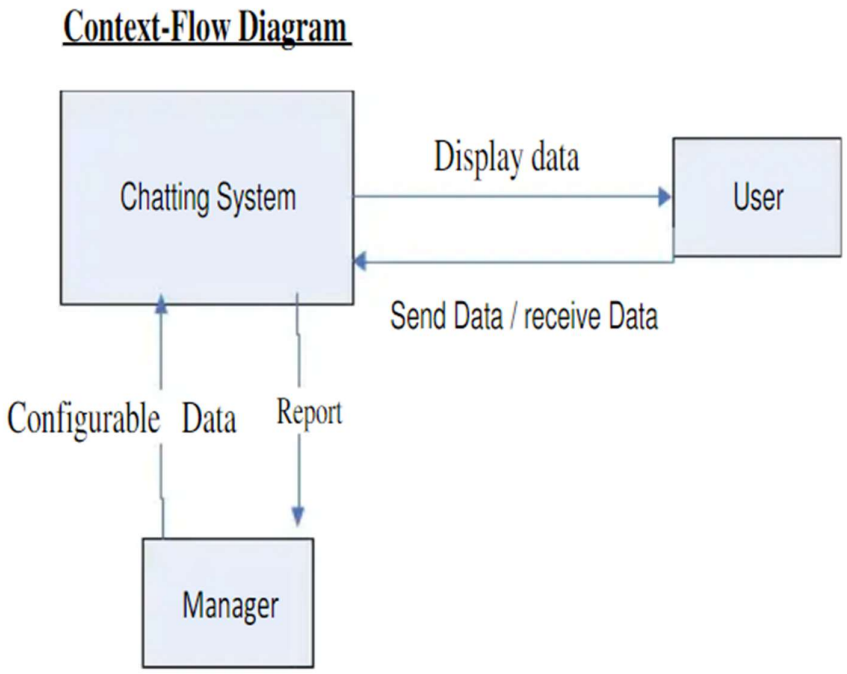
Through DFD, our project illustrates the flow of data elements from the data source external or internal data store to store internal data or external data sink, through an internal process.

A DFD has no information about the timing of operations, or whether it will work in sequence or in parallel.

Therefore quite different from Flowchart, which shows the flow controls through an algorithm, which allows the reader to determine what will be implemented processes, in what order, and under any circumstances, but not any kind of data will be input if

Context Diagram:

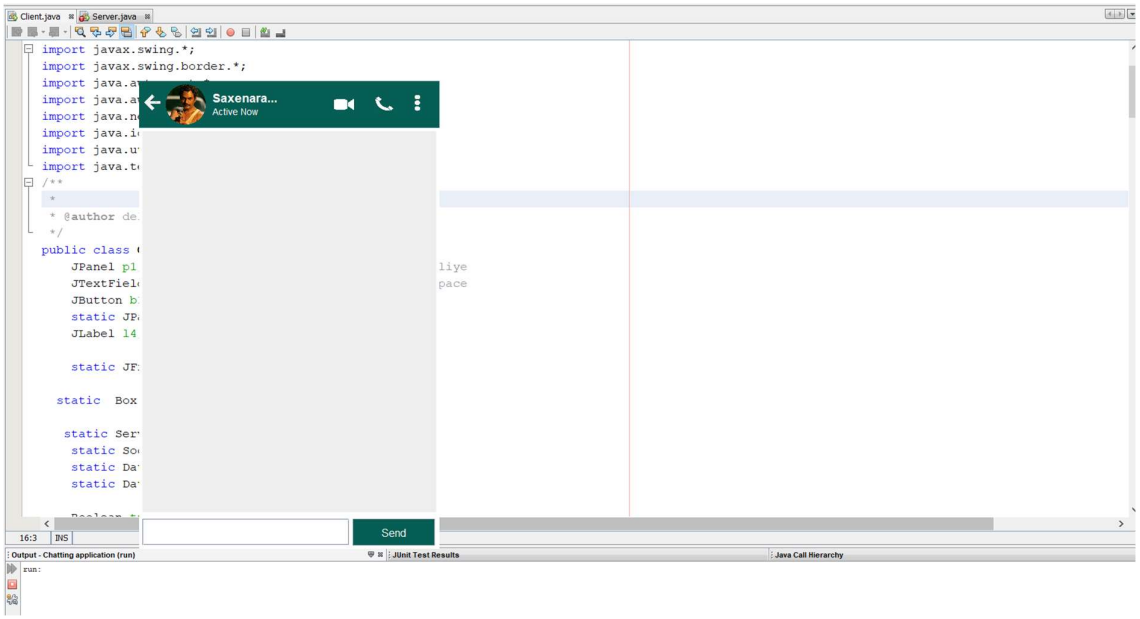
The following figure shows the free context of the work of the project. Figure 7: illustrates the context diagram of the system.



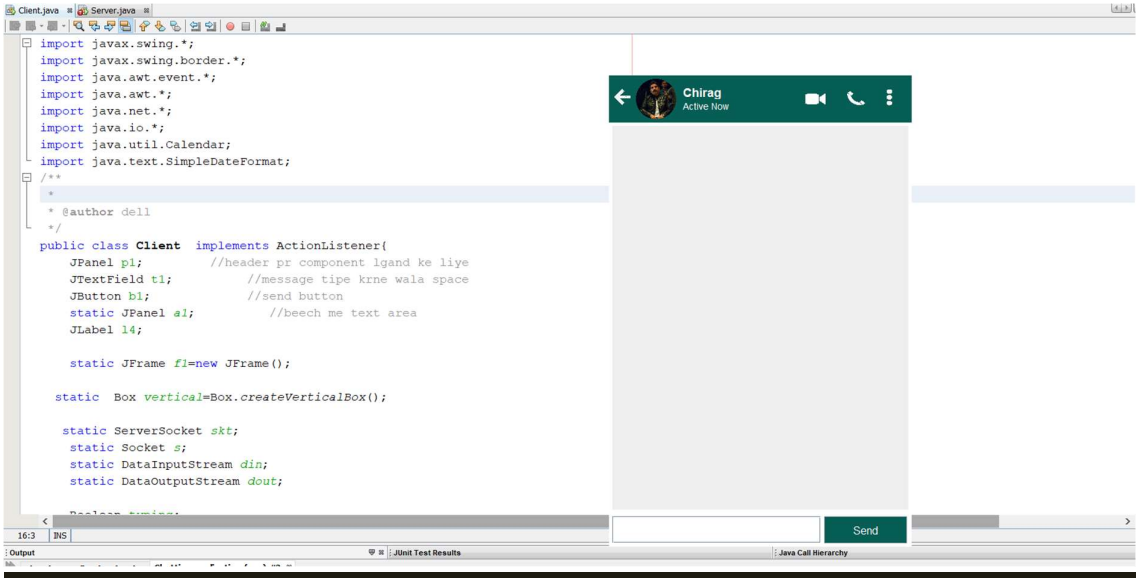
Chapter 4: Form Design

4.1 Input/Output form

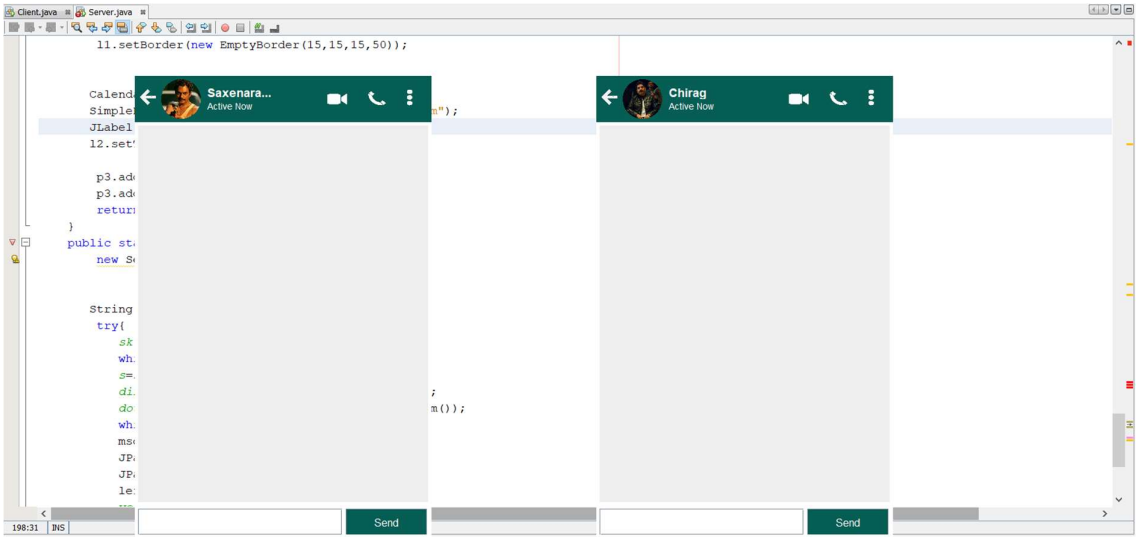
SERVER SCREEN



CLIENT SCREEN



CLIENT SERVER SCREEN



Chapter 5 Coding

5.1 Module wise code

5.1.1 Server code:

```
package chatting.application;

import javax.swing.*;
import javax.swing.border.*;
import java.awt.event.*;
import java.awt.*;
import java.net.*;
import java.io.*;
import java.util.Calendar;
import java.text.SimpleDateFormat;

public class Server implements ActionListener {

   JPanel p1;         //header pr component lgand ke liye
   JTextField t1;          //message type krne wala space
   JButton b1;             //send button
   static JPanel a1;         //middle text area
   static JFrame f1 = new JFrame();
   JLabel l4;
   static Box vertical = Box.createVerticalBox();
   static ServerSocket skt;
   static Socket s;
   static DataInputStream din;
   static DataOutputStream dout;
   Boolean typing;
   Timer t;
   FileWriter fw;

   Server() //constructor
   {
```

```
        p1 = new JPanel();
        p1.setLayout(null);
        p1.setBackground(new Color(7, 94, 84));     //green color of header
        p1.setBounds(0, 0, 450, 70);
        f1.add(p1);


        ImageIcon i1 = new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/3.png");  //arrow
        Image i2 = i1.getImage().getScaledInstance(30, 30, Image.SCALE_DEFAULT);
        ImageIcon i3 = new ImageIcon(i2);
        JLabel l1 = new JLabel(i3);
        l1.setBounds(5, 17, 30, 30);
        p1.add(l1);
        l1.addMouseListener(new MouseAdapter() {              //mouse exit by arrow event

           public void mouseClicked(MouseEvent ae) {
              System.exit(0);
           }
        });


        ImageIcon i4 = new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/1.png");           // profile picture
        Image i5 = i4.getImage().getScaledInstance(60, 60, Image.SCALE_DEFAULT);
        ImageIcon i6 = new ImageIcon(i5);
        JLabel l2 = new JLabel(i6);
        l2.setBounds(40, 5, 60, 60);
        p1.add(l2);

        ImageIcon i7 = new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/video.png");      //video icon
        Image i8 = i7.getImage().getScaledInstance(30, 30, Image.SCALE_DEFAULT);
        ImageIcon i9 = new ImageIcon(i8);
```

```
        JLabel l5 = new JLabel(i9);
        l5.setBounds(290, 20, 35, 30);
        p1.add(l5);

        ImageIcon i11 = new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/phone.png");           //phone icon
        Image i12 = i11.getImage().getScaledInstance(35, 30, Image.SCALE_DEFAULT);
        ImageIcon i13 = new ImageIcon(i12);
        JLabel l6 = new JLabel(i13);
        l6.setBounds(350, 20, 35, 30);
        p1.add(l6);


        ImageIcon i14 = new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/3icon.png"); //3 lines on header icon
        Image i15 = i14.getImage().getScaledInstance(13, 25, Image.SCALE_DEFAULT);
        ImageIcon i16 = new ImageIcon(i15);
        JLabel l7 = new JLabel(i16);
        l7.setBounds(410, 20, 13, 25);
        p1.add(l7);

        JLabel l3 = new JLabel("Saxenarajat5487"); //name of person on header
        l3.setFont(new Font("SAN_SERIF", Font.BOLD, 18));
        l3.setForeground(Color.white);
        l3.setBounds(110, 15, 100, 20);
        p1.add(l3);

        l4 = new JLabel("Active Now");    //header active now
        l4.setFont(new Font("SAN_SERIF", Font.PLAIN, 14));
        l4.setForeground(Color.white);
        l4.setBounds(110, 35, 100, 20);
        p1.add(l4);

        t = new Timer(1, new ActionListener() {
```

```
    public void actionPerformed(ActionEvent ae) {
        if (!typing) {
            l4.setText("Active Now");
        }
    }
});

t.setInitialDelay(2000);


a1 = new JPanel();
//a1.setBounds(5,75,440,570);
a1.setFont(new Font("SAN_SERIF", Font.PLAIN, 16));

JScrollPane sp = new JScrollPane(a1);
sp.setBounds(5, 75, 440, 570);
sp.setBorder(BorderFactory.createEmptyBorder());
f1.add(sp);

t1 = new JTextField();     //text field for messsage
t1.setBounds(5, 655, 310, 40);
t1.setFont(new Font("SAN_SERIF", Font.PLAIN, 16));
f1.add(t1);

t1.addKeyListener(new KeyAdapter() {

    public void keyPressed(KeyEvent ke) {
        l4.setText("typing...");
        t.stop();
        typing = true;
    }

    public void keyReleased(KeyEvent ke) {
```

```
            typing = false;

            if (!t.isRunning()) {
                t.start();
            }
        }
    });

    b1 = new JButton("Send");      //Send Button Property
    b1.setBounds(320, 655, 123, 40);
    b1.setBackground(new Color(7, 94, 84));
    b1.setForeground(Color.WHITE);
    b1.setFont(new Font("SAN_SERIF", Font.PLAIN, 16));
    b1.addActionListener(this);          //Action performed
    f1.add(b1);

    f1.getContentPane().setBackground(Color.white);

    f1.setLayout(null);
    f1.setSize(450, 700);
    f1.setLocation(400, 200);
    f1.setUndecorated(true);   //header htane ke liye
    f1.setVisible(true);
}

public void actionPerformed(ActionEvent ae) {
    try {
        String out = t1.getText();
        sendTextToFile(out);
        JPanel p2 = formatLabel(out);
        a1.setLayout(new BorderLayout());
        JPanel right = new JPanel(new BorderLayout());
        right.add(p2, BorderLayout.LINE_END);
        vertical.add(right);
```

```
        vertical.add(Box.createVerticalStrut(15));
        a1.add(vertical, BorderLayout.PAGE_START);
        dout.writeUTF(out);
        t1.setText("");
    } catch (Exception e) {
        System.out.println(e);
    }
}

public void sendTextToFile(String message) throws FileNotFoundException {

    try {
        File out = new File("chat.txt");
        FileWriter writer = new FileWriter(out, true);
        writer.write("SaxenaRajat5487:" + message);
        writer.write("\n");
        writer.flush();
        System.out.println("Data successfully written in the specified file");

    } catch (Exception e) {
    }
}

public static JPanel formatLabel(String out) {
    JPanel p3 = new JPanel();
    p3.setLayout(new BoxLayout(p3, BoxLayout.Y_AXIS));


    JLabel l1 = new JLabel("<html><p style = \"width : 150px\">" + out + "</p></html>");
    l1.setFont(new Font("Tamoha", Font.PLAIN, 16));
    l1.setBackground(new Color(37, 211, 102));
    l1.setOpaque(true);
    l1.setBorder(new EmptyBorder(15, 15, 15, 50));
```

```
    Calendar cal = Calendar.getInstance();
    SimpleDateFormat sdf = new SimpleDateFormat("HH:mm");
    JLabel l2 = new JLabel();
    l2.setText(sdf.format(cal.getTime()));

    p3.add(l1);
    p3.add(l2);
    return p3;
}

public static void main(String[] args) {
    new Server().f1.setVisible(true);


    String msginput = "";
    try {
        skt = new ServerSocket(6001);
        while (true) {
            s = skt.accept();
            din = new DataInputStream(s.getInputStream());
            dout = new DataOutputStream(s.getOutputStream());
            while (true) {
                msginput = din.readUTF();
                JPanel p2 = formatLabel(msginput);
                JPanel left = new JPanel(new BorderLayout());
                left.add(p2, BorderLayout.LINE_START);
                vertical.add(left);
                f1.validate();

            }
        }
    } catch (Exception e) {
    }
```

```
    }
}
```

Client Code :

```
package chatting.application;
import javax.swing.*;
import javax.swing.border.*;
import java.awt.event.*;
import java.awt.*;
import java.net.*;
import java.io.*;
import java.util.Calendar;
import java.text.SimpleDateFormat;

public class Client  implements ActionListener{
    JPanel p1;         //header pr component lgand ke liye
    JTextField t1;          //message tipe krne wala space
    JButton b1;             //send button
    static JPanel a1;          //beech me text area
    JLabel l4;

    static JFrame f1=new JFrame();

  static  Box vertical=Box.createVerticalBox();

   static ServerSocket skt;
    static Socket s;
    static DataInputStream din;
    static DataOutputStream dout;

    Boolean typing;
    Timer t ;

FileWriter fw;
```

```
Client()          //constructor
{
    p1=new JPanel();
    p1.setLayout(null);
    p1.setBackground(new Color(7, 94, 84));     //green color of header
    p1.setBounds(0,0,450,70);
    f1.add(p1);


    ImageIcon i1= new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/3.png");  //arrow
    Image i2=i1.getImage().getScaledInstance(30,30,Image.SCALE_DEFAULT);
    ImageIcon i3=new  ImageIcon(i2);
    JLabel l1= new JLabel(i3);
    l1.setBounds(5,17,30,30);
    p1.add(l1);
    l1.addMouseListener(new MouseAdapter(){             //mouse exit by arrow event
       public void mouseClicked(MouseEvent ae)
       {
          System.exit(0);
       }
    });


    ImageIcon i4= new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/2.png");          //gaitonde photo
    Image i5=i4.getImage().getScaledInstance(60,60,Image.SCALE_DEFAULT);
    ImageIcon i6=new  ImageIcon(i5);
    JLabel l2= new JLabel(i6);
    l2.setBounds(40,5,60,60);
    p1.add(l2);
```

```
ImageIcon i7= new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/video.png");           //video icon
Image i8=i7.getImage().getScaledInstance(30,30,Image.SCALE_DEFAULT);
ImageIcon i9=new  ImageIcon(i8);
JLabel l5= new JLabel(i9);
l5.setBounds(290,20,35,30);
p1.add(l5);

ImageIcon i11= new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/phone.png");           //phone icon
Image i12=i11.getImage().getScaledInstance(35,30,Image.SCALE_DEFAULT);
ImageIcon i13=new  ImageIcon(i12);
JLabel l6= new JLabel(i13);
l6.setBounds(350,20,35,30);
p1.add(l6);


ImageIcon i14 = new ImageIcon("C:/Users/dell/Documents/NetBeansProjects/Chatting application/src/chatting/application/Icon/3icon.png"); //3 lines on header icon
Image i15 = i14.getImage().getScaledInstance(13, 25, Image.SCALE_DEFAULT);
ImageIcon i16 = new ImageIcon(i15);
JLabel l7 = new JLabel(i16);
l7.setBounds(410, 20, 13, 25);
p1.add(l7);

JLabel l3=new JLabel("Chirag"); //name of person on header
l3.setFont(new Font("SAN_SERIF",Font.BOLD,18));
l3.setForeground(Color.white);
l3.setBounds(110,15,100,20);
p1.add(l3);

 l4=new JLabel("Active Now");    //header active now
l4.setFont(new Font("SAN_SERIF",Font.PLAIN,14));
l4.setForeground(Color.white);
```

```
l4.setBounds(110,35,100,20);
p1.add(l4);
 t= new Timer(1, new ActionListener(){
    public void actionPerformed(ActionEvent ae){
        if(!typing){
            l4.setText("Active Now");
        }
    }
});

t.setInitialDelay(2000);

a1=new JPanel();
// a1.setBounds(5,75,440,570);
a1.setFont(new Font("SAN_SERIF",Font.PLAIN,16));


 JScrollPane sp=new JScrollPane(a1);
 sp.setBounds(5,75,440,570);
 sp.setBorder(BorderFactory.createEmptyBorder());
 f1.add(sp);


t1 = new JTextField();
t1.setBounds(5, 655, 310, 40);
t1.setFont(new Font("SAN_SERIF", Font.PLAIN, 16));
f1.add(t1);
 t1.addKeyListener(new KeyAdapter(){
    public void keyPressed(KeyEvent ke){
        l4.setText("typing...");

        t.stop();

        typing = true;
```

```
        }

        public void keyReleased(KeyEvent ke){
            typing = false;

            if(!t.isRunning()){
                t.start();
            }
        }
    });

    b1 = new JButton("Send");
    b1.setBounds(320, 655, 123, 40);
    b1.setBackground(new Color(7, 94, 84));
    b1.setForeground(Color.WHITE);
    b1.setFont(new Font("SAN_SERIF", Font.PLAIN, 16));
    b1.addActionListener(this);
    f1.add(b1);

    f1.getContentPane().setBackground(Color.white);

    f1.setLayout(null);
     f1.setSize(450,700);
     f1.setLocation(1100,200);
     f1.setUndecorated(true);//header htane ke liye
     f1.setVisible(true);
}


public void actionPerformed(ActionEvent ae)
{
    try{
        String out=t1.getText();
        sendTextToFile(out);
```

```
    JPanel p2= formatLabel(out);

    a1.setLayout(new BorderLayout());

    JPanel right=new JPanel(new BorderLayout());
    right.add(p2,BorderLayout.LINE_END);
    vertical.add(right);
     vertical.add(Box.createVerticalStrut(15));
    a1.add(vertical, BorderLayout.PAGE_START);
    dout.writeUTF(out);
    t1.setText("");
  }catch(Exception e){
     System.out.println(e);
  }
}
 public void sendTextToFile(String message) throws FileNotFoundException{

      try{

         File out = new File("chat.txt");
        FileWriter writer = new FileWriter(out,true);

         writer.write("Chirag:"+message);
         writer.write("\n");
           writer.flush();
        System.out.println("Data successfully written in the specified file");
         }catch(Exception e)
      {
        System.out.println("Exception:" +e);
      }
    }

public static JPanel formatLabel(String out){
   JPanel p3= new JPanel();
```

```
  p3.setLayout(new BoxLayout(p3, BoxLayout.Y_AXIS));


    JLabel l1=new JLabel("<html><p style = \"width : 150px\">"+out+"</p></html>");
    l1.setFont(new Font("Tamoha",Font.PLAIN,16));
    l1.setBackground(new Color(37,211,102));
   l1.setOpaque(true);
   l1.setBorder(new EmptyBorder(15,15,15,50));


   Calendar cal= Calendar.getInstance();
   SimpleDateFormat sdf=new SimpleDateFormat("HH:mm");
   JLabel l2=new JLabel();
   l2.setText(sdf.format(cal.getTime()));

    p3.add(l1);
    p3.add(l2);
    return p3;
}
public static void main(String[] args){
    new Client().f1.setVisible(true);


    try{
        s=new Socket("127.0.0.1",6001);
        din=new DataInputStream(s.getInputStream());
       dout= new DataOutputStream(s.getOutputStream());
       String msginput="";

       while(true){
       msginput =din.readUTF();
      JPanel p2=formatLabel(msginput);
       JPanel left=new JPanel(new BorderLayout());
       left.add(p2,BorderLayout.LINE_START);
```

```
            vertical.add(left);
            vertical.add(Box.createVerticalStrut(15));
            a1.add(vertical,BorderLayout.PAGE_START);
            f1.validate();
            }
        }catch(Exception e){

        }
    }

}
```

Chapter 6 Testing

6.1Testing plans

System testing is the process of performing a variety of tests on a system to explore functionality or to identify problems. System testing is usually required before and after a system is put in place.

A series of systematic procedures are referred to while testing is being performed. These procedures tell the tester how the system should perform and where common mistakes may be found.

Testers usually try to "break the system" by entering data that may cause the system to malfunction or return incorrect information.

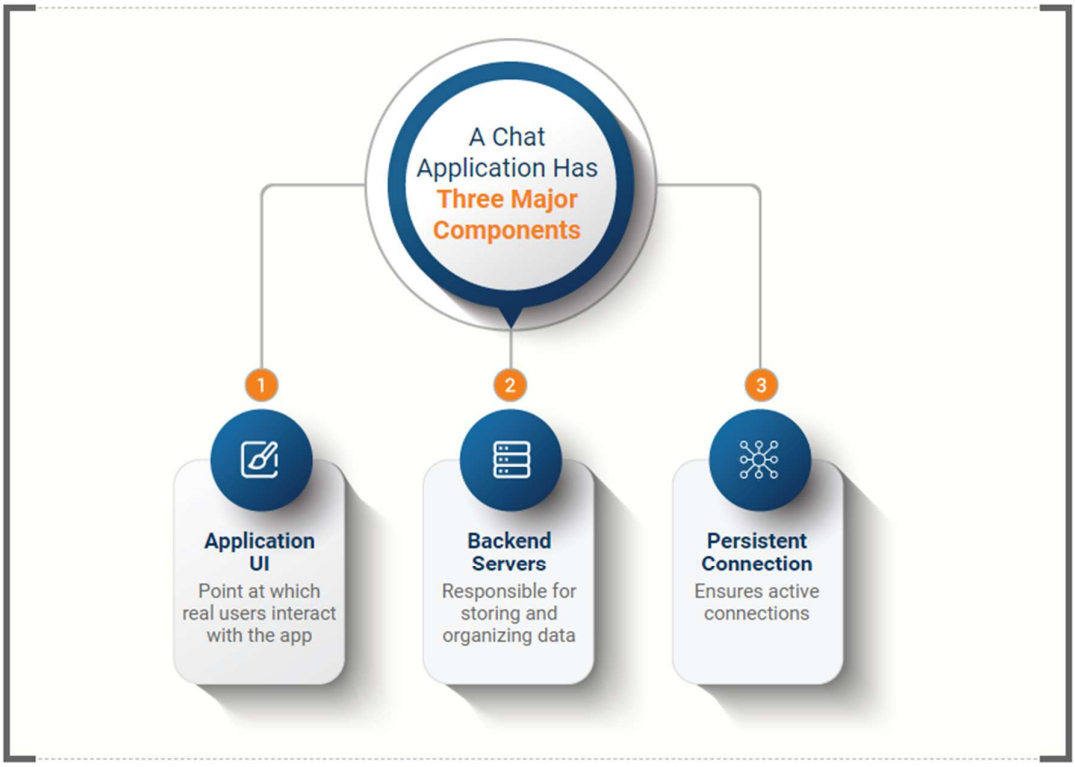
So, this could mean two things depending on an SDLC model. The first type of testing is the actual testing by users.

This is usually done in models wherein implementation does not go with pre-testing with users. On the other hand, there are also testing that uses professionals in the field. This testing is aimed in cleaning the software of all the bugs altogether.

For software that is set for public release, the software is first tested by other developers who were not in charge in creating the software.

They will weed out the bugs and suggest fixes if every they find one. Once this stage is completed, it is time to test the software not just to the developers but to actual users.

Components of a Chat Application



Benefits of Chat App Performance Testing

- Lower messages latency
- Maximum messages delivery
- Supports heavy multimedia files
- Set group v/s members limit expectations
- Higher messages throughput

6.2 Types and Steps of Testing:

The following steps are important to perform System Testing:

Step 1: Create a System Test Plan.

• Step 2: Create Test Cases.

Step 3: Carefully Build Data used as Input for System Testing.

Step 4: If applicable create scripts to Build environment and To automate Execution of test cases.

Step 5: Execute the test cases.

Step 6: Fix the bugs if any and re test the code.

Step 7: Repeat the test cycle as necessary.

6.3 Test report details:

Every test of the actual-testing or unit-testing is documented in a test report.

Test reports include the following data:

- Name and email address of the tester.
- Date and time at which the test was performed.
- Scope of test.
- Test environment including operating system and type of test
- client.

- Test plan i.e. the list of test cases giving method/use case and input parameters used.
- Test log listing the outcome of the test case executions (manual testing or automated testing).
- Summary of test results.
- Recommendation for action when errors had been encountered.

Chapter 7-other

7.1 MAIN OBJECTIVE

The aim of this project is to express how we can implement a simple chat application between a server and a client? The application is a Desktop based application and is implemented using Swing and AWT. The project is developed in Java SE language executed on a single stand-alone java across a network using loop back address concept.

Application consists of two programs:

1.)Server
2.)Client

Server Module:

The server module of the application waits for the client to connect to it. Then if connection is granted a client interacts communicates and connects to the server, it can mutually communicate with the server. The duty of the server is to let clients exchange the messages.

Client Module:

The client module is the one that utilizer sends requests to the server. Utilizer utilizes the client as the means to connect to the server. Once he establishes the connection, he can communicate to the connected server.

Our project aims to build chatting system helps communication between students one hand, between students and staff members on the other hand, where the project aims achieve following:

1. Easy to better access to staff members.

2. Increasing degree participation students in the courses.

3. Improve ability of students on production information and ability to handling with complex problems.

4. Increase level students' interest in content courses through interaction

with staff member.

5. Overcome limits space and time in learning operation.

6. Save time and effort.

7. Reduce the load on staff members.

8. Contributing to the different viewpoints of the students.

9. Provide great opportunity to identify miscellaneous sources of information different ways.

7.2 Motivation

There are many motivation for our project include the following:

1. Does current system of communication between students and staff members.

2. The difficulty of communication between students and staff members Because of time constraints.

3. Development of spirit cooperation and open dialogue between students with each with staff members.

4. Required for graduation degree.

7.3REQUIREMENTS

1. **External Interface Requirements**

2. **User Interface:** The user interface required to be developed for the system should be user friendly and attractive.

There are two sets of Java APIs for graphics programming: AWT (Abstract Windowing Toolkit) and Swing.

I. AWT API was introduced in JDK 1.0. Most of the AWT components have become obsolete and should be replaced by newer Swing components.

II. Swing API, a much more comprehensive set of graphics libraries that enhances the AWT, was introduced as part of Java Foundation Classes (JFC) after the release of JDK 1.1. JFC consists of Swing, Java2D, Accessibility, Internationalization, and Pluggable Look-and-Feel Support APIs. JFC was an add-on to JDK 1.1 but has been integrated into core Java since JDK 1.2.

Software Interfaces:

I. **Programming Language**: Java
II. SOCKET PROGRAMMING

7.4 Operational Concepts and Scenarios

Operation of the application based on the inputs given by the user:

When the run button is clicked then the chat form is initialized with a connection between the host and the client machine.

Note: server must be started at first before a client start.

i. Contains a rich textbox which send messages from one user to another.

ii. Contains a textbox for messages to be written that is sent across the network.
iii. Contains a Send button
iv. When the sent button is clicked, in the background, the text in the textbox is encoded sent as a packet over the network to the client machine. Here this message is decoded and is shown in the rich textbox.

6.5 Future work

There is always a room for improvements in any software package, however good and efficient it may be done. But the most important thing should be flexible to accept further modification. Right now we are just dealing with text communication. In future this software may be extended to include features such as:

1.)Files transfer: this will enable the user to send files of different formats to others via the chat· application.
2.)Voice chat: this will enhance the application to a higher level where communication will be· possible via voice calling as in telephone.

Conclusion

I Developed network applications in Java by using sockets, threads, and Web services.

This software is portable, efficient, and easily maintainable for large number of clients. Our developed web-based chatting software is unique in its features and more importantly easily customizable. The java.net package provides a powerful and flexible set of classes for implementing network applications.

Typically, programs running on client machines make requests to programs on a server Machine. These involve networking services provided by the transport layer. The most widely used transport protocols on the Internet are TCP (Transmission control Protocol) and UDP (User Datagram Protocol).

TCP is a connection-oriented protocol providing a reliable flow of data between two computers. On the other hand, UDP is a simpler message-based connectionless protocol which sends packets of data known as datagrams from one computer to another with no guarantees of arrival.

Bibliography

Internet reference

www.books.google.co.in

www.wikipidia.org

www.javaworld.com

www.javatpoint.com