NETWORK PROGRAMMING
(A CASE STUDY OF SECURED SERVER-CLIENT CHAT APPLICATION)

BY

ALILE DANIEL OSAWONAMEN

PSC1808777

DEPARTMENT OF COMPUTER SCIENCE,
FACULT OF PHYSICAL SCIENCES,
UNIVERSITY OF BENIN

SEPTEMBER 2023

NETWORK PROGRAMMING
(A CASE STUDY OF SECURED SERVER-CLIENT CHAT APPLICATION)

BY

ALILE DANIEL OSAWONAMEN

PSC1808777

SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULT OF PHYSICAL SCIENCES, UNIVERSITY OF BENIN IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF BACHELOR OF SCIENCE (BS.C) HONS DEGREE COMPUTER SCIENCE

SEPTEMBER 2023

CERTIFICATION

This is to certify that this research work was carried out by **ALILE DANIEL OSAWONAMEN** with matriculation number **MAT NO:** PSC1808777, Faculty of Physical Sciences, Department of Computer Science, University of Benin, Benin city under my supervision.

…… ……..

PROF.GODSPOWER O. EKUOBASE, PhD. **Signature/Date**
(Project Supervisor)

……. …….

Prof.(Mrs) A.O Egwali **Signature/Date**
(Head of Department)

APPROVAL

This project report written by **ALILE DANIEL OSAWONAMEN, with** matriculation number PSC1808777, in partial fulfillment of the requirement for the award of the University of Benin Bachelor of Science (B. Sc.) degree in Computer Science, is adequate both in scope and content and it is hereby approved for presentation.

…… ……

PROF.GODSPOWER O.EKUOBASE,PhD. Date
(Project Supervisor)

…… ……..

Prof.(Mrs) A.O Egwali Date

DEDICATION

This project work is dedicated to God Almighty, for providence, guidance, and grace in seeing me through this study; I give Him all the glory. I also dedicate this project to my parents Mr. and Mrs. Alile for without them, I would not have come this far.

ACKNOWLEDGEMENT

The greatest and the most breathtaking concept ever is that God Himself knows me personally. I really cannot fathom the implication of His love for me. He is the greatest guidance counselor because He guided me through this study. I express my sincere gratitude to Him, the originator of all things.

I want to acknowledge my supervisor **PROF. GODSPOWER O. EKUOBASE,** PhD. who was very understanding and patient with me through the course of my project. I want to acknowledge the Head of Department **PROF. (MRS.)** A.O. **EGWALI** and my lecturers who have impacted in my educational life.

I am very appreciative and thankful to my loving Father Mr. Alile Edward Osariemen who without His support financially, spiritually and emotionally this project would not have gone through. I also acknowledge my brother Alile Eddy Uyiosa for the support and encouragement.

I also appreciate my friends, for their patience and constant love during the course of the project and my stay in UNIBEN.

Lastly, I want to acknowledge my fellow course mates that have been together with me from the onset.

ABSTRACT

Several network systems are built to communicate with one another as well as made available through service-oriented architectures. In this project, the client server architecture is used to develop a chat application. Firstly, a chat application is created for both Client and Server which is based on Transmission Control Protocol (TCP) where TCP is connection oriented protocol and is a reliable connection protocol. As security is the key factor while communicating over a network, so in this project, MySQL SSL protocol and hash function was used for the Database based on a numbers of benefits. The hash values of the real password and the random generated number (salt) is stored in the database. The original password is not stored on the system, making cracking of password much hard**er.**

TABLE OF CONTENTS

CHAPTER ONE

1.0 INTRODUCTION

Different networks are constructed to talk to each other, and they work using service-oriented arrangements. In this project, we're using the client-server setup to make a secure Client-Server chat app. This chat app is created using the Transmission Control Protocol (TCP), which is good for connected communication. Towards the end, we use multithreading to build the app.

A client-server chat app includes two main parts: the Chat Client and the Chat Server. These parts talk back and forth. To make this happen, we use tools like the Message Processor, which helps understand what a user is saying. The Message Interpreter takes out and passes along received messages. The Message Maker puts messages together again, and the Client Manager keeps a list of clients. Both the sender and receiver use this list to talk to each other.

Usually, the server starts on a computer first-even before the client. The server gets itself ready and waits for a client to ask for something. Once the client asks, the server accepts the request and does what it's asked. After it's done, the server goes back to waiting for the next request. This cycle continues as long as the server is running. Whenever a request comes in, the server quickly helps the client and then waits again for the next request.

1.2 BACKGROUND OF STUDY

The client-server model serves as the widely embraced standard for developing network applications. Within this framework, two fundamental roles emerge: the client and the server. As the names suggest, the server, which operates either as a process or on a computer, provides services to entities known as clients. Conversely, a client is a running process situated on the same or a different computer, seeking services from the server.

In the context of a chat application, it essentially consists of two key components:

Client application: Operates on the client computer (or the machine with the server).

In this chat application, clients can send data to anyone connected to the server. Java's application programming interface (API) offers classes for establishing sockets, enabling program communication across networks. Sockets represent the endpoints for logical connections between two hosts and are employed for data transmission. Java treats socket communication much like input and output operations with files, making reading from or writing to sockets equally straightforward. Creating a server connection necessitates crafting a server socket attached to a designated port. This port serves as the interface where the server awaits incoming connections. It's here that the Transmission Control

Protocol service is recognized on the socket. Notably, email servers commonly operate on port 25, while web servers frequently utilize port 80.

Server Execution: On the server side, a thread is generated to handle multiple client requests.

This thread maintains a list containing clients' names and IP addresses. Subsequently, this list is broadcasted to all users within the chat room. Whenever a client logs out, the server removes them from the list, updates it, and then broadcasts the revised list to all available clients.

Client Execution: To initiate communication, a client must first register by sending its username to the server and then initiate a thread. This thread facilitates the retrieval of a comprehensive list of available clients. After this setup, any two registered clients can engage in communication with each other.

Top of Form

1.3 STATEMENT OF PROBLEM

A significant problem with a client-server chat app is that the information shared between users isn't well-protected while being sent. This could allow unauthorized users to access the data and potentially harm user accounts by changing the information.

1.4 OBJECTIVES OF THE STUDY

This project's aim is to aid the creation of a strong and safe network program (Client-Server chat model) that can run a chat application for multiple users at once. It uses Java's socket programming with Transport Control Protocol (TCP). Security is crucial for network communication, so a hash function with salt is used to protect the Database for various reasons. MySQL is picked to build this application because it's easy to manage, performs well, safeguards data, works well for the web and data storage, and is cost-effective. It's also open-source.

1.5 SIGNIFICANCE OF THE STUDY

Beyond the conventional scope of client-server chat functionalities, this project extends its significance through several key aspects:

Enhanced Database Security: The project leverages the MySQL database, ensuring that information remains safeguarded within it. Personal details and messages, including private communications, undergo encryption using the MySQL-provided encrypt or. The integration of a hash function precedes encryption, with the resultant hash values, and randomly generated numbers (salt), securely stored in the database. This fortified approach guarantees that, even in the event of a database breach, the inclusion of salt renders the computation of the original password considerably challenging. This robust

implementation significantly bolsters the security and reliability of the chat application server.

Private Chat Functionality: A noteworthy feature of this application is its provision for private chatting. This unique capability empowers two users to engage in private conversations. Notably, the messages exchanged between these users remain concealed from the general chat display text field. Instead, they are confined exclusively to the designated private message display text field.

1.6 SCOPE OF THE STUDY

The project will look into several areas, including but not limited to:

1. Getting a better understanding of how network programming in Java functions.
2. Creating a reliable network communication setup for a Client-Server chat application.
3. Examining Java-based network programming, particularly in the context of Multithreaded Client-Server Chat applications, to better understand potential solutions.
4. Running practical tests to figure out the specifics of the problem. Finally, suggesting ways to fix the issues and recommending steps to avoid them in the future.

1.7 LIMITATIONS

The earlier Client-Server Chat system only utilizes a hash function for the password before encryption, which is then stored in the Database. Consequently, the database could be easily compromised to deduce the original password. Some disadvantages of the Client-Server Chat include:

The server could get overwhelmed if it receives numerous requests from clients, leading to congestion and overloading.

If the server malfunctions, users also face issues.

T11/ If a password is lost, it cannot be recovered.

Unauthorized clients could potentially hack into client accounts and alter data.

CHAPTER TWO

LITERATURE REVIEW

In Computer Science, client-server is a software architecture model consisting of two parts, client systems and server systems both communicating over a computer network or on the same computer. A client-server application is a distributed system consisting of both client and server software. The client process always initiates a connection to the server, while the server process always waits for requests from any client. When both the client process and server process are running on the same computer, this is called a single seat setup.

The client-server relationship describes the relation between the clients and how it makes a service request from the server and how the server can accept these requests, process them and return the requested information to the client.

2.1 Client-Server and Other Models

The client-server model according to Hou Meng-bo (May 2008) was originally developed to allow more users to share access to database applications. Compared to the mainframe approach, client-server offers improved scalability because connections can be made as needed rather than being fixed. Client-server is just one approach to managing network applications. The primary alternative, peer-to-peer networking, models all devices as having

equivalent capability rather than specialized client or server roles. Compared to client-server, a peer to peer network offers some advantages such as more flexibility in growing the system to handle large number of clients. Client-server networks generally offer advantages in keeping data secure.

2.2 Client-Server Communication

According to Fisher Price t al (2011) at a basic level, network-based systems consist of a server, client and a media for communication as shown in the figure below. A computer running a program that makes a request for services is called client machine. A computer running a program that offers requested services from one or more clients is called server machine. The media for communication can be wired or wireless network.

Generally, programs running on client machines make requests to a program (often called as server program) running on a server machine. They involve networking services provided by the transport layer which is part of the Internet software stack often called TCP/IP (Transport Control Protocol/Internet Protocol) stack. The transport layer comprises two types of protocols, TCP (Transport Control Protocol) and UDP (User Datagram Protocol). The most widely used programming interfaces for these protocols are sockets. TCP is a connection-oriented protocol that provides a reliable flow of data between two computers. Example applications that use such services are HTTP, FTP, and

Telnet. UDP is a protocol that sends independent packets of data, called datagram, from one computer to another with no guarantees about arrival and sequencing. Example applications that use such services include Clock server and Ping.

2.3 Hosts Identification and Service Ports

Every computer on the Internet is identified by a unique, 4-byte IP address. This is typically written in dotted quad format like 128.250.25.158 where each byte is an unsigned value between 0 and 255. This representation is clearly not user-friendly because it does not te11 us anything about the content and then it is difficult to remember. Hence, IP addresses are mapped to name like www.caritas.edu.ng or www.google.com,which are easier to remember. Internet supports name servers that translate these names to IP addresses. In general, each computer only has one Internet address. However, computers often need to communicate and provide more than one type of service or to talk to multiple hosts/computers at a time. For example, there may be multiple ftp sessions, web connections, and chat programs all running at the same time. To distinguish these services, a concept of port, a logical access point is used. This means that each service offered by a computer is uniquely identified by a port number. Each Internet packet contains both the destination host address and the port number on that host to which the message/request has to be delivered. The host

computer dispatches the packets it receives to programs by looking at the port numbers specified within the packets. That is, IP address can be thought of as a house address when a letter is sent via post/snail mail and port number as the name of a specific individual to whom the letter has to be delivered.

2.4 Sockets and Socket-Based Communication

Sockets provide an interface for programming networks at the transport layer. Network communication using sockets is very much similar to performing file I/O. Socket handle is treated like file handle according to Ming Xue Et al. The streams used in file I/O operation are also applicable to socket-based I/O. Socket-based communication is independent of a programming language used for implementing it. This means that a socket program written in Java language can communicate to a program written in non-Java (say C or C++) socket program. A server (program)runs on a specific computer and has a socket that is bound to a specific port. The server listens to the socket for a client to make a connection request. If everything goes well, the server accepts the connection. Upon acceptance, the server gets a new socket bound to a different port. It needs a new socket (consequently a different port number) so that it can continue to listen to the original socket for connection requests while serving the connected client.

2.5 TCP/IP Socket programming

The two key classes from the java.net package used in creation of server and client programs are:

- Server Socket
- Socket

A server program creates a specific type of socket that is used to listen for client requests (server socket). In the case of a connection request, the program creates a new socket through which it will exchange data with the client using input and output streams. The socket abstraction is very similar to the file concept: developers have to open a socket, perform I/O, and close it.

2.6 Socket Programming in Java

According to Gandhi F. et al, (2010) a socket is the one end-point of a two-way communication link between two programs running over the network. Running over the network means that the programs run on different computer. However, one can run these two programs on the same computer. These two communicating programs form a Client-Server application. In Client-Server applications, the server 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 to 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 to 27 communicate with the client. The server also binds a socket to its local port and communicates with the client by reading from and writing to it. The server uses a specific port dedicated only to listening for connection requests from other clients. The client and the server must agree on a protocol. They must agree on the language of the information transferred back and forth through the socket. The java.net package in the Java development environment provides the class Socket which implements the client side and the class ServerSocket class which implements the server side of the two-way link.

2.7 Secure Internet Programming

According to H. Mahmoud (November 2008) any information transmitted over computer networks or the Internet is subject to interception. Some of that information could be sensitive such as credit card numbers and other personal data. To make the Internet more useful in an enterprise setting and for e-commerce, applications must protect their users' information using encryption, authentication, and secure communications protocols. The secure Hypertext Transfer Protocol (HTTPS), which is HTTP over the Secure Sockets Layer (SSL), is already being used successfully for e-commerce applications**.**

2.8 Overview of secure socket layer (SSL).

The SSL protocol which was developed by Netscape in 2010, allows clients (Web browsers, typically) and HTTP servers to communicate over a secure connection. It offers encryption, source authentication and data integrity as means to protect information exchanged over insecure, public networks.

(OEncryption: It protects data from unauthorized users by converting it to an apparently meaningless form before transmission. The data is encrypted by one side (the client or the server), transmitted, decrypted by the other side, and then processed.

Source authentication: It is a method of verifying the data sender's identity. The first time a browser or other client attempts to communicate with a Web server over a secure connection, the server presents the client with a set of credentials in the form of a certificate.

Data integrity refers to means of ensuring that data has not been modified in transit.

2.9 Security

According to Gutman, Naccache, Palmer C. (2005, June), there are many aspects to security and many applications ranging from secure commerce and payments to private communications and protecting passwords.

Cryptography is the science of writing in secret code and is an ancient art; the first documented use of cryptography in writing dates back to circa 1900 B.C. when an Egyptian scribe used non-standard hieroglyphs in an inscription. Within the context of any application-to-application communication, there are some specific security requirements, including:

Authentication: The process of proving one's identity. (The primary forms of host-to-host authentication on the Internet today are name-based or address-based.)

Privacy/confidentiality: Ensuring that no one can read the message except the intended receiver.

Integrity: Assuring the receiver that the received message has not been altered in any way from the original.

Non-repudiation: A mechanism to prove that the sender really sent this message.

Cryptography not only protects data from theft or alteration but can also be used for user authentication.

In all cases, the initial unencrypted data is referred to as plaintext. It is encrypted into cipher text which will in turn (usually) be decrypted into usable plaintext.

TYPES OF CRYPTOGRAPHIC ALGORITHMS

There are several ways of classifying cryptographic algorithms which are as follows:

Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption.

Public Key Cryptography (PKC): Uses one key for encryption and another for decryption.

Hash Functions: Uses a mathematical transformation to irreversibly "encrypt" information.

2.10 Hash Functions

Hash functions, also called message digests and one-way encryption are algorithms that in some sense use no key. Instead, a fixed-length hash value is computed based upon the plaintext that makes it impossible for either the contents or length of the plaintext to be recovered. Hash algorithms are typically used to provide a 31 digital fingerprint of a file's contents often used to ensure that the file has not been altered by an intruder or virus. Hash functions are also commonly employed by many operating systems to encrypt passwords. Hash functions then provide a measure of the integrity of a file**.**

Types of Hash algorithms:

Message Digest (MD) algorithms: A series of byte-oriented algorithms that produce a 128-bit hash value from an arbitrary-length message.

Secure Hash Algorithm (SHA): Algorithm for NIST's Secure Hash Standard (SHS). SHA-1 produces a 160-bit hash value and was originally published as FIPS 180-1 and RFC 3174. FIPS 180-2 (aka SHA-2) describes five algorithms in the SHS: SHA-1 plus SHA-224,SHA-256,SHA-384,and SHA-512 which can produce hash values that are 224, 256, 384, or 512 bits in length, respectively.

(6C)Hash of Variable Length (HAVAL): Designed by Y. Zheng, J. Pieprzyk and J. Seberry, a hash algorithm with many levels of security. HAVAL can create hash values that are 128,160,192,224, or 256 bits in length.

Whirlpool: Whirlpool operates on messages less than 2256 bits in length, and produces a message digest of 512 bits. The design of this has function is 32very different than that of MD5 and SHA-1, making it immune to the same attacks as on those hashes.

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.1 METHODOLOGY

The Structured Systems Analysis and Design Methodology (SSADM) is a well-defined (structured) methodology. It is an integrated set of standards and guides for the analysis and design of computer systems. It is an integrated set of standards and guidelines consisting of:

I. **Structural standards**: This defines the structure of a development project in the form of explicitly defined tasks, with clearly defined interfaces between them, and clearly defined tangible products.
II. **Technique guides**: This provides development staff with a set of proven usable techniques and tools, and detailed rules and guidelines on when and how to use them; and
III. **Documentation standards**: This provides the means of recording the products of development activity at a detailed level.

SSADM is waterfall method by which an Information System design can be arrived at; SSADM can be thought to represent a pinnacle of the rigorous document-led approach to system design

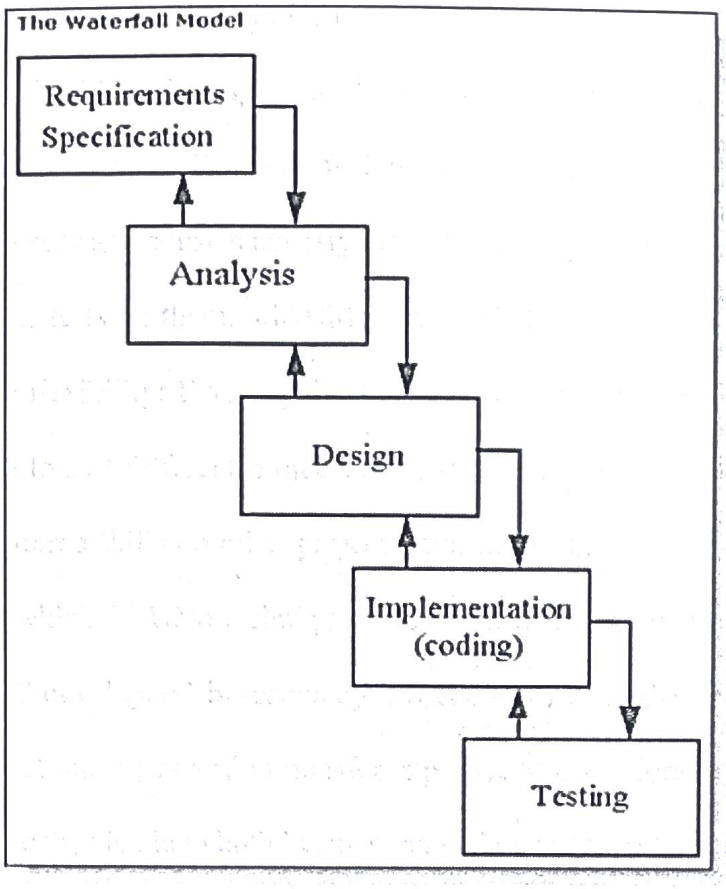


Figure 3.1: The waterfall model

Benefits of the Structured Systems Analysis and Design Methodology (SSADM)

i. Deliver the system to users on time: SSADM has a modular structure which relates directly to project deliverables and helps in all aspects of project management. It gives a clear specification of what is to be produced and how it is 34 to be managed and reviewed. There are well defined interfaces to management and specialist techniques.

ii. Deliver systems that meets user's needs: By continuously involving users, by modeling business activities and work practice, by using prototyping, by making the IT professional's thinking visible through diagrammatic techniques, SSADM enhances the prospects for success on large and small projects.

iii. Improve quality by reducing error rates: Quality can be improved by detecting errors early in the lifecycle, especially by involving users as well as skilled practitioners in checking for errors. Rigorous techniques promote accuracy, with adequate checks of completeness and consistency. By defining the required quality of design documents, and stating the tests for them, SSADM promotes better quality management.

iv. Improve flexibility: Every application development is different. The ability of tailoring SSADM to suit different projects is a major factor for organizations who wish to reuse their resource skills on other projects, and to be able to benefit from the many different ways in which SSADM techniques and products may be applied.

v. Avoid IT developers' bureaucracy: SSADM has been designed to provide useful tools for project managers and to transfer expertise to practitioners. Its use makes benefits, as well as costs, visible to both business and IT management and users.

3.2 Stages Involved In Structured Systems Analysis and Design Methodology

The SSADM method involves the application of a sequence of analysis, documentation and design tasks concerned with:

1. Feasibility Stage Analyze the current situation at a high level. A Data Flow Diagram is used to describe how the current system works and to visualize known problems. The following steps are part of this stage:

   i. Investigate and define requirements.

   ii. Investigate current processing.

   iii. Investigate current data.

2. Analysis Stage The first part is researching the existing environment, where system requirements are identified and the current business environment is modeled. Modeling consists of creating a Logical Data Structure for processes and data structures that are part of the system. The following steps are part of this stage:

   i. Requirements Specification Stage

   ii. Define required system processing.

   iii. Develop required data model.

   iv. Derive system functions.

   v. Develop specification prototypes.

   vi. Develop processing specification.

3. Logical System Specification Stage

In this stage, technically feasible options are chosen. The development/implementation environments are specified based on this choice. The following steps are part of this stage:

Logical System Specification Stage

i. Define user dialogue.
ii. Define update processes.
iii. Define enquiry processes.

4. Physical Design Stage The objective of this stage is to specify the physical data and process design, using the language and features of the chosen physical environment and incorporating installation standards. The following activities are part of this stage:

i. Prepare for physical design
ii. Complete the specification of functions.
iii. Incrementally and repeatedly develop the data and process designs.

3.2.1 PRIMARY DATA COLLECTION

The process of information gathering was achieved through so many sources including:

i. Oral interview of users of existing system
ii. Textbooks in the library
iii. Newspaper, Journals and articles
iv. Other publications

3.2.2 SECONDARY DATA COLLECTION

i. File downloads from the Internet

ii. Electronic media

3.2.3 ANALYSIS OF THE EXISTING SYSTEM

The existing Client-Server Chat system use TCP Transmission Control Protocol (TCP) that is connection oriented. It is reliable; it guaranteed that each packet will arrive and also guaranteed that packets will be in the right order. All personal details and messages including the private messages in the Database are encrypted. It implements only hash function with the password before the encryption and then stored in the Database. Thus, the database can be compromised easily to compute the original password.

3.2.4 LIMITATIONS OF THE EXISTING SYSTEM

Since the existing system implements only hash function with the password before the encryption and stored in the database, the password in the database can be hacked easily because hash function can be decrypted easily using hash calculator. The main weakness of the existing client-server chat application is that there is no security provided to data which is transferred between clients. Any unauthorized client can hack the client account and can change the data.

3.3 SYSTEM DESIGN

This Client-Server Chat system use Transmission Control Protocol (TCP) that is connection oriented. It is reliable; it guaranteed that each packet will arrive and also guaranteed that packets will be in the right order. All personal details and messages including the private messages in the Database are encrypted. This project implements hash function with the password before the encryption and then stored in the Database. Also, this use random generated numbers (salt) that is calculated together with the pass worded hash value and stored in the Database. Thus, even if the database is compromised (random generated salt added to hash value) makes it harder to compute the original password. Therefore, makes the server more secured. This chat system implements real network connection. To implement the application on a real network, MySQL must be installed on the server host. The client system must have all the classes compiled without error. Thereafter, the user can start and use the same server IP address and Port number to be able to establish connection with the server.

3.3.1 JAVA.NET Class: Java can be used easily to develop network applications. It comes with a very powerful class library for networking as part of java.net package. The java.net package will have to be included at the beginning of the program for network input and output methods operations. By using TCP, the server runs first and then waits for client's requests and sever

has ability to either accept or deny the client request. A server has a socket which is bound to a specific port. Client programs send connection request to the server program and if everything is well then server accepts the connection. After accepting the client request the server gets a new socket bound to a different port, it requires a new socket bound because the original socket has to listen to other connection requests.

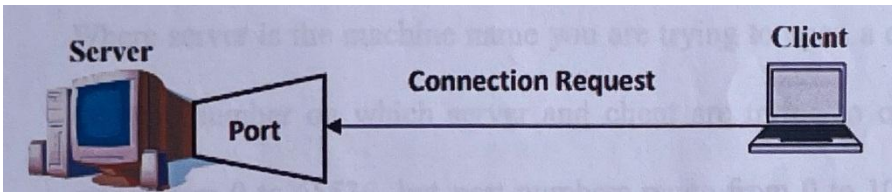


Figure 3.2:**Client Sending Connection Request to Server**

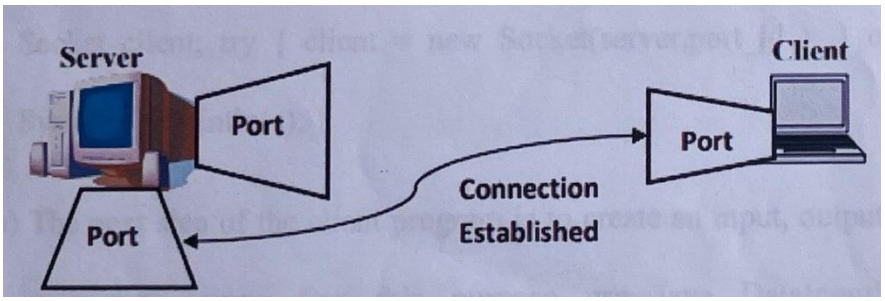


Figure 3.3:Client-Server Connection Established by using TCP

3.3.2. Socket Programming with TCP

TCP (Transfer Control Protocol) is a stream communication protocol basically it's a connection-oriented protocol. To communicate over TCP one must firstly have to establish a connection between pair of sockets, where one socket is client and the other belongs to server. After the connection is established between them then they can communicate with each other.

(I) Client Program in JAVA:

(a) To communicate with server program, the client has to create socket by writing the following statement: Socket client = new Socket(server, port_id);

Where server is the machine name you are trying to open a connection to and port_id is the port number on which server and client are trying to connect. Port numbers have range from 0 to 65536, but port numbers range from 0 to 1023 are reserved for system use so select only the port number greater than 1023 which are also available. The code written above cannot handle the exceptions so the above code can be written as following, to handle the exceptions:

Socket client; try{ client = new Socket(server,port_id); } catch (IOException e) {System.out.println(e);}

(b) The next step of the client program is to create an input, output stream to communicate with the server. For this purpose use java DataInputStream for input and DataOutputStream or PrintStream for output. Following is the code used to create an input stream to receive data from server:

input=new DataInputStream(client.getInputStreami()): tInputStream(); String line=input.readLine() ();

And following is the code used to create an output stream to send data to the server socket:

output = new

DataOutputStream(client.getOutputStream());output.writeBytes("Hello\n");

(c) The last step is to close socket but before closing the socket input and output stream must be closed. Following the code for closing the socket:
try {output.close(); input.close(); client.close(); 42} catch (IOException e){System.out.println(e); }

3.3.3 A Simple TCP Server Program in JAVA:

(i) To communicate with client program first server have to create socket by writing the following statement: ServerSocket server = new ServerSocket(PortNumber);Here again there is need to mention the PortNumber on which server and client are trying to connect. The code written above cannot handle the exceptions so the above code can be written as following, to handle the exceptions: ServerSocket server;try { server = new ServerSocket(PortNumber); } catch (IOException e) {System.out.println(e);}

(ii) In the server program, server also has to wait for client connection request and after getting client request, the server has either to accept or deny the request.The following is the code below to listen and accept the client request: Socket clientSocket = null; try

{ serverSocket= server.accept(); } catch (IOException e){System.out.println(e);}

(iii) In the server program, use java DataInputStream to receive input from client and use DataOutputStream to send information to client. Following is the code used to create an input stream to receive data from client: input= new DataInputStream(server.getInputStreamDataInputStream(server.getInputStream(); String line = input.readLine() (); And following is the code used to create an output stream to send data to the client socket: output = new DataOutputStream(server.getOutputStream());output.writeBytes("Hello\n");

(iv) The last step is to close socket but before closing the socket input and output stream must be closed. Following the code for closing the socket: try{output.close(); input.close(); serverSocket.close(); server.close(); } catch (IOException e) { System.out.println(e); }

3.3.4 Client-Server Chat Application using TCP in JAVA:

A chat application is basically a combination of two applications:

- server application
- client application

Server application runs on the server computer and client application runs on the client computer. In chat application, a client can send data to anyone who is

connected. Either data can be directly sent to anyone who is connected or it can be sent by informing server. This application uses the second method, because in this case client is connected to only server and can communicate with many clients and server doesn't transmit the IP of client to anyone which increases the security of the client.

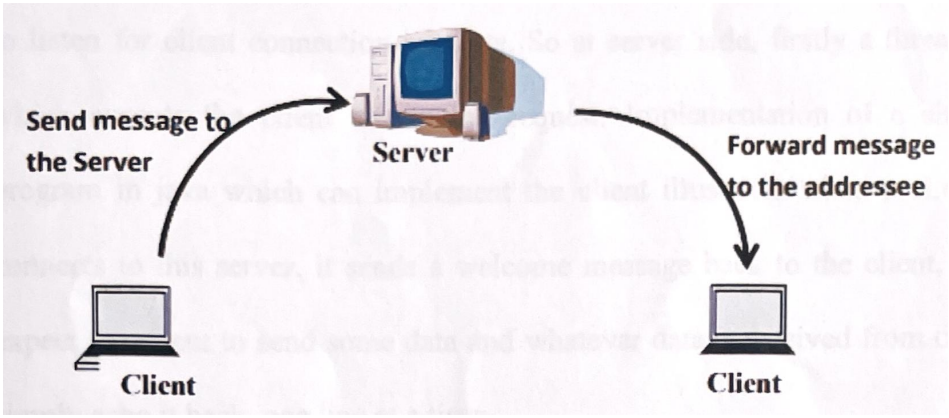


Figure 3.4: Client-Server Chat Application by using TCP

Before start chatting, the client must have to establish a connection with the server. In this application, we have one server and many clients who can communicate to each other via server. Client must encode the data in the language which is understandable by both the server and the other client. The chat application will work in the following sequence:

- A client can chat only if it's authenticated, so firstly a client must authenticate from by sending its “id”. The “id” of each client must be unique.

- The server maintains a list of authenticated clients and sends the list to all authenticated clients.
- The Server can send any message to any client, and the server also has the authority to disconnect any client from chat system.
- A client can send messages or chat with any authenticated clients.

3.3.4.1 Server Execution:

Firstly, all the clients have to connect with server, for this purpose server should have to listen for client connection requests. So at server side, firstly a thread is created which accepts the client connection request. Implementation of a simple server program in java which can implement the client illustrates when a client program connects to this server, it sends a welcome message back to the client, then it will expect the client to send some data and whatever data it received from client, it will simply echo it back, one line at a time.

(I) **Multicasting:**

In case clients wants to do a group chat then it sends request to the server and in answer, server provides a group IP to the client and send information about the room to all the concerned clients. Now when clients send a message then it will multicast to all the clients in group IP. In case of system failure, it may be server sender or receiver that failed then there will be three stages which can be show using following figure 3.5:

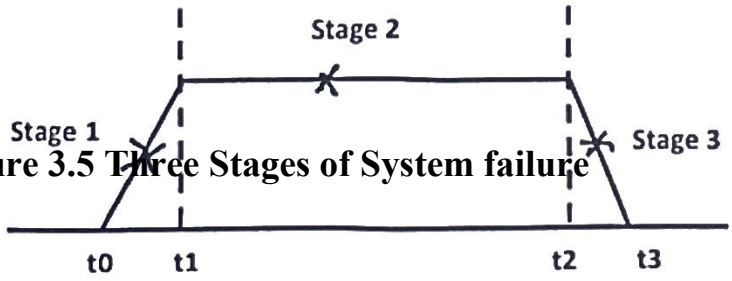


Figure 3.5 Three Stages of System failure

(a) In case system of sender get failed at stage 1 then packet will save in the queue and will not send until sender recovers but in case when receiver fails then packet will not be received.

(b) In case of sender fails at stage 2, the packet is already sent from the sender buffer and now it is on the way. In this condition, the receiver will receive the message but if at this stage it fails then, the message will not be received.

(c) In case sender get fails at stage 3, the packet is in receivers buffer so it has no impact but in case if receiver fails then the packet is placed in receiver buffer so as soon as receiver recovers it will get the message.

(ii) **Thread**: A thread is a section of code which is executing independently of others threads in a same program. Java has a class Thread which is defined in java. Lang package. Thread is the most powerful feature that JAVA supports from other programming languages.

(iii) Multithreaded Client-Server Chat Application: In a multithreaded chat application a client will use two threads to interact with standard input and with

server. At server side there will be a separate thread for each client. Whenever a client wants to establish a connection then a new client thread is created. A multithreaded chat application will work like this:

(a) On the server side there will be a main thread which will continually listens to a given port.

(b) The client program client will send a request to establish a connection by sending the users username.

(c) After establishing the connection, another thread is spawned to open a dialog box for chatting with clients.

(d) At the server side for each connection there will be separate thread.

(e) The Server program wi11 maintain a list of all online clients and send that list to all clients who are currently available.

(f) If a client logs out then the server will update its list and forward that updated list to all online clients.

(iv) Deadlock: A deadlock situation will occur when two threads are circularly dependent on each other e.g. thread1 is holding an object which is needed by the thread2 to complete its execution and thread2 is holding the same object for which thread1 is waiting so both of them waiting for each other to release the resource so in this condition a deadlock is created because thread1 and thread2 are circularly dependent on each other.

Locks: While using resources threads can use locks to prevent from lock situation. Locks allow java threads to quickly and easily communicate with each other. If a thread is holding a lock on an object then no other thread can use that object. In java each object has a lock by using synchronized keyword. Synchronized blocks can only be executed by one thread at a time.

Synchronizing Thread: When two or more threads need access to a shared resource, there should be a means through that resource will be used by only one thread at a time.

The procedure through which this synchronization is obtained is called thread synchronization. The synchronized, a keyword in Java, is used to create a block of code referred to as a critical section. The following is the syntax of synchronized statement in

java:

synchronized(object) {//statements to be synchronized}

3.4 DATABASE DESIGN

The data base adopted in this work is MYSQL

MySQL must be installed on the system and the port used for the MySQLconfiguration when you install it is port 3306

---->>create a table in a new database

create Database chatDatabase

use chatDatabase

create table users (name char(40), email char(25), user char(15), password char(15),random char(10))

Make sure the MySQL is correctly setup, and then test it by running the Database Manager class. It will open the USERS table and print details about it. If all fine (no error) then the database is fine and Chat Server can connect to it. Start the server first, and then make sure the start button on the server GUI is press (to establish connection between the server and the daabase). With successfully connection established with the Database, the client can then request for connection.

3.5 SYSTEM FLOWCHART.7 SYSTEM FLOWCHART

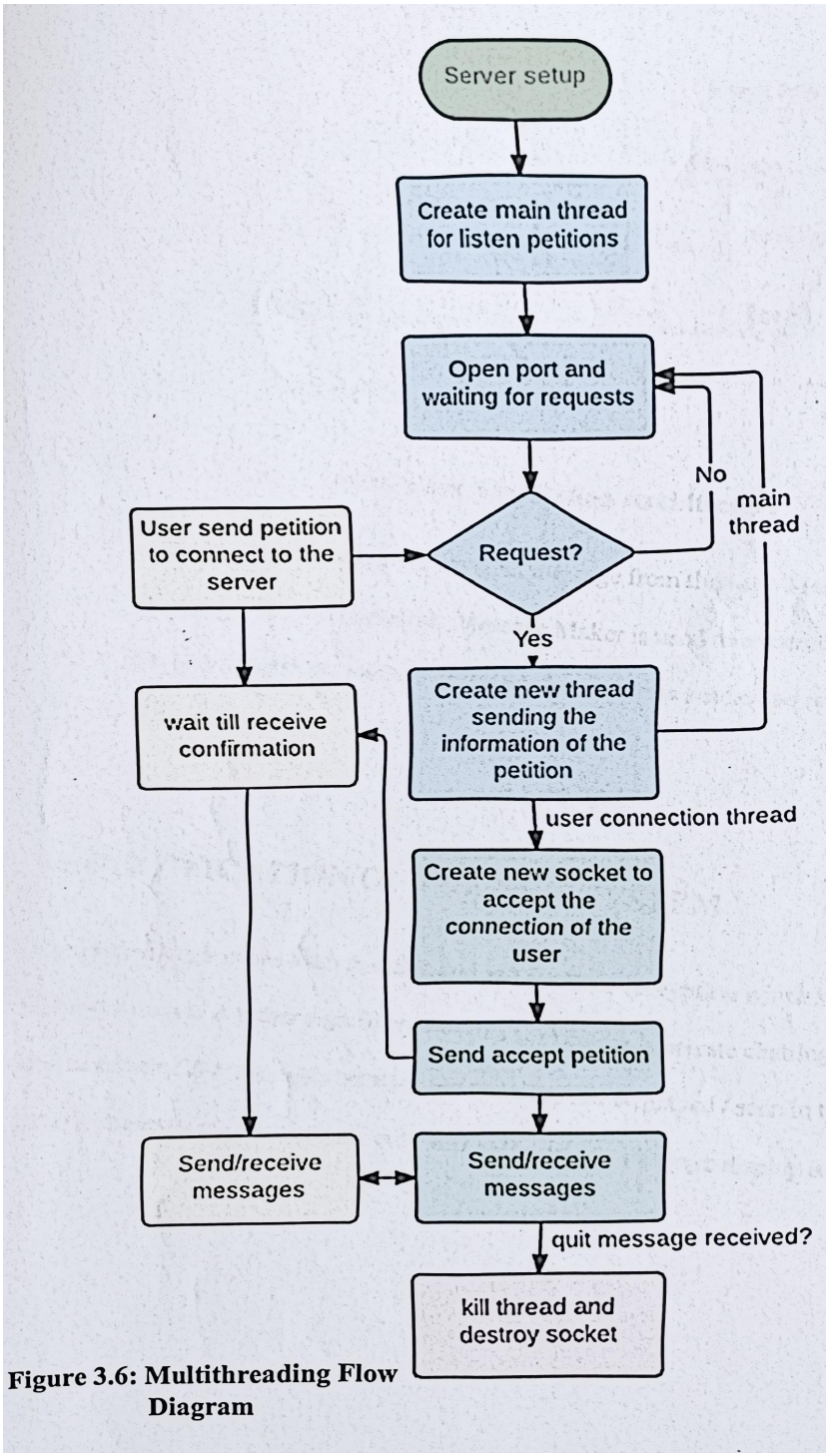


Figure 3.6: Multithreading Flow Diagram

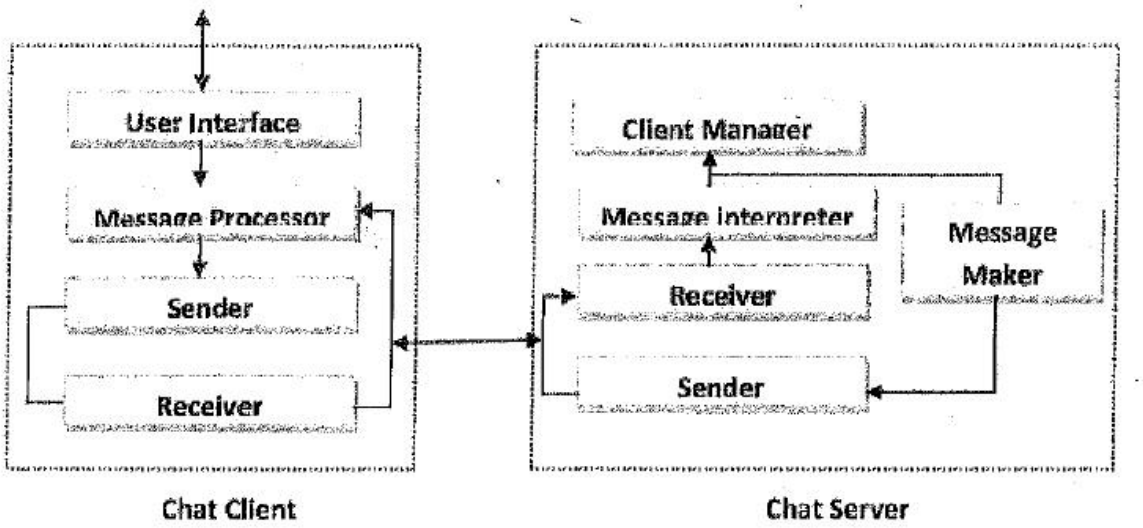


Figure **37 Client-Server Chat Application Architecture**

Here Message Processor is used to interpret message from the user. Message Interpreter is used to extract and then parse the received message. Message Maker is used to construct back the message whereas Client Manager is used to maintain the clines lint, whereas sender and receiver at Beth sides are used to interact with each other.

3.9 JUSTIFICATION OF THE NEW SYSTEM

This system implements hash function and salt for Data encryption which makes the system greatly difficult to crack. Another significance of this application is private chatting this is where two users can chat in private. The messages between the users are not display/seen in the general chat display text field. The messages are displayed only within the private massage display text field.

CHAPTER FOUR

IMPLEMENTATION, TESTING AND INTEGRATION

4.1 CHOICE OF DEVELOPMENT TOOLS

Java became the chosen programming language for the client-server chat application because Java application programming interfaces (API) provides the classes for creating sockets to facilitate program communications over the network. Sockets are the endpoints of logical connections between two hosts and can be used to send and receive data. Java treats socket communications much as it treat input and output operations; thus programs can read from or write to sockets as easily as they can read from or write to files. MySQL became the chosen database for the client-server chat application as a result of the following reasons:

(II) The MySQL database server offers the best in scalability, sporting the ability to handle intensely embedded applications by a footprint of merely 1MB to running huge data storehouses handling terabytes of data. Its open source nature permits total customization for adding unique necessities to the database server.

(III) Either the planned application is a fast-speed business processing scheme or a large-volume network site that provides a billion queries in 24 hours, MySQL is able to meet the most challenging performance outlook of any scheme.

(IV) MySQL provides outstanding security features that guarantee total data protection. It offers strong mechanisms for making sure that unauthorized clients have no access to the database server.

(V) The hallmarks of MySQL are steady availability and hard reliability.

(VI) MySQL is free and simple with low cost overhead. It is dependable and simple to maintain.

4.2 SYSTEM REQUIREMENTS

Operating system: Linux (Ubuntu), windows xp, windows Vista, Windows 7,e.t.c

Processor speed: 1.5 GHz and above

Memory: 1 GB and above

Hard disk drive space: 10GB

4.2.1 SOFTWARE REQUIREMENTS

The software requirements includes: MySQL essential 5.0 and above. Java development kit (JDK) 6ull and above.

4.2.2 HARDWARE REQUIREMENTS

In the course of the design, the software developed needed the following hardware for an effective and efficient operation.

(I) Processor speed: 1.5 GHz and above

(II) Memory:1 GB and above

(III) Hard disk drive space: 10GB

(IV) E.G.A/V.G.A, a colored monitor.

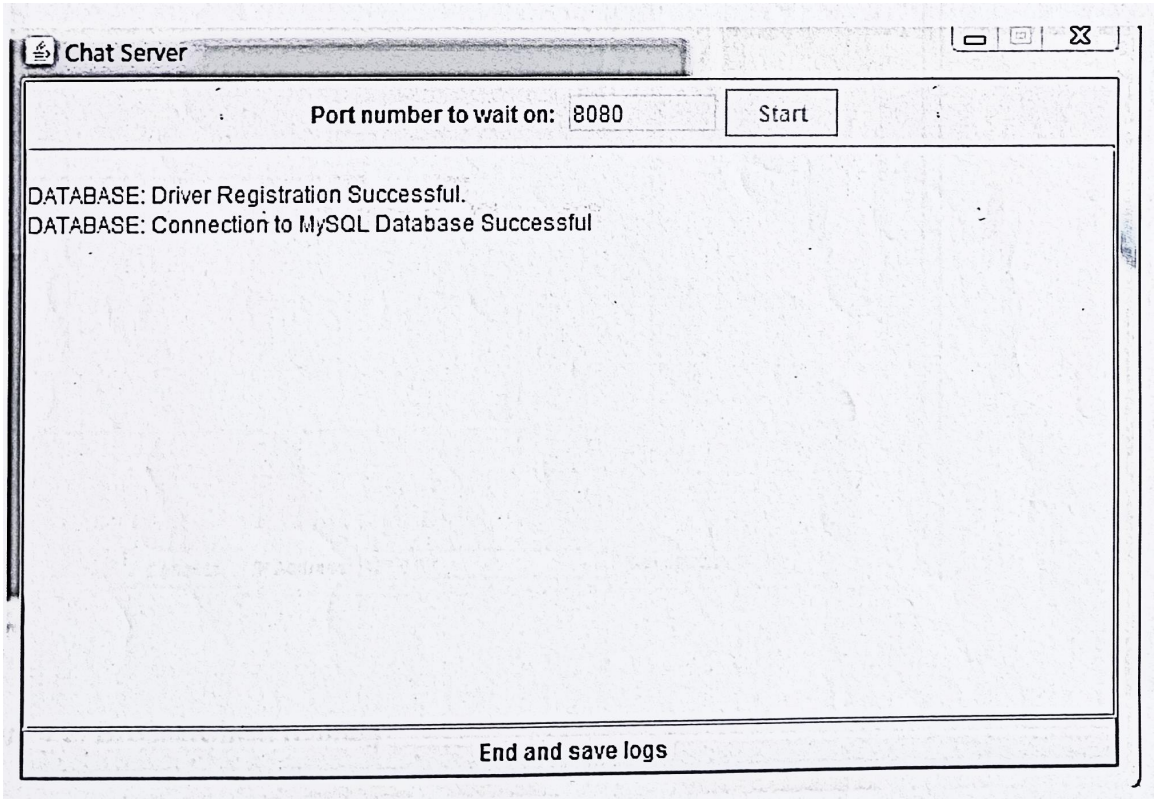
(V) An uninterruptible power supply (UPS) units

(VI) Standard keyboard and Mouse.

4.3 IMPLEMENTATION

As described in previous Chapters, in TCP based client server system, server will always start first on some specific port and wait to listen to the client connection requests. So in this Chat system firstly Chat Server is started in the following and waiting on port number 8080for Chat Client connection request and is successfully connected to the MySQL database. As shown in the figure 4.1.

Figure 4.1:BAChatServer running



At the next step, server will accept the client connection request. So, the Chat Client window which is connected to the Chat Server by using IP address 127.0.0.1 on local host and port number 8080. As shown in the figure 4.2.

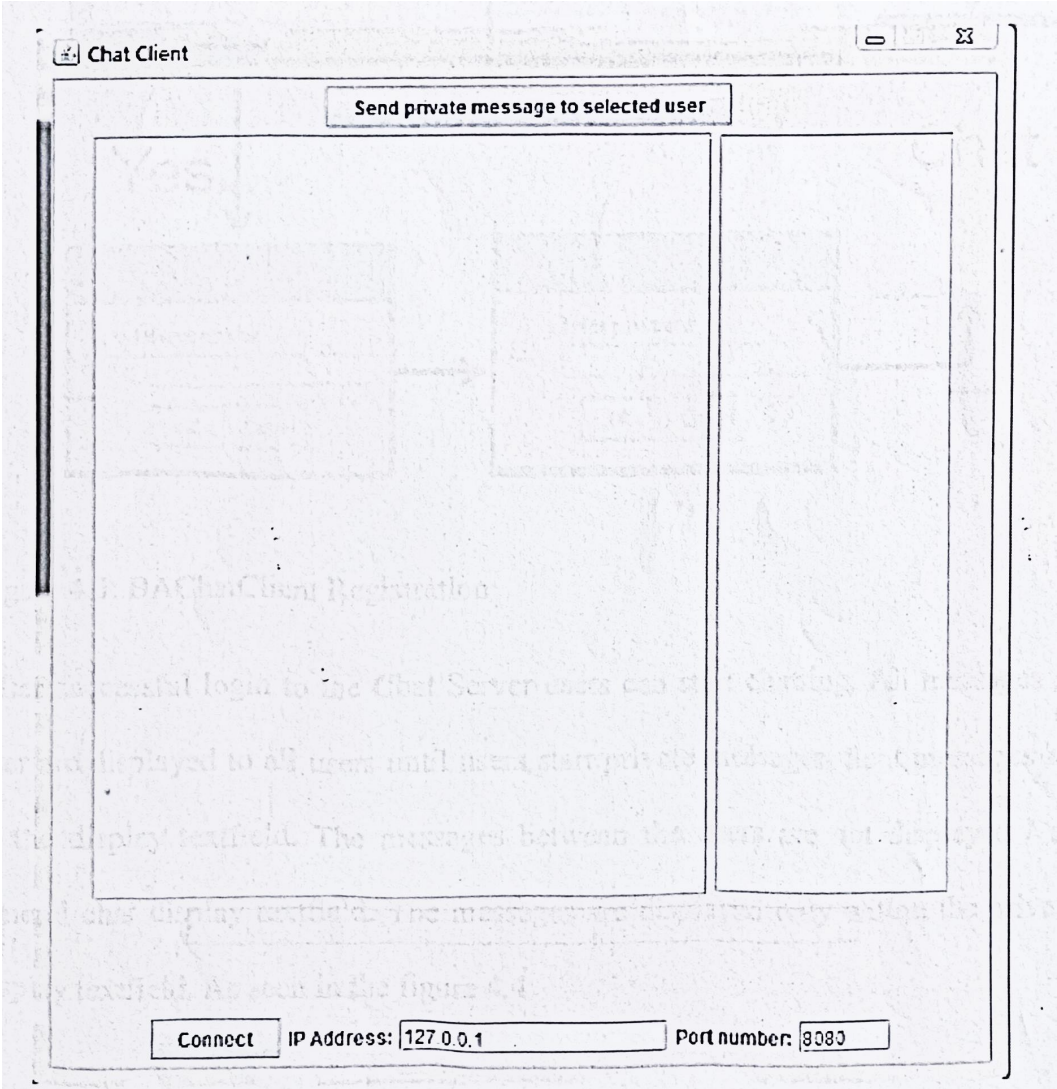


Figure 4.2:BAChatClient running

Before starting the chat, the client must be registered. To start the registration two options are offered: option 1 - press "Yes" for already registered client to

input login details; option 2 -press “No” for new client to register. As shown 4.3

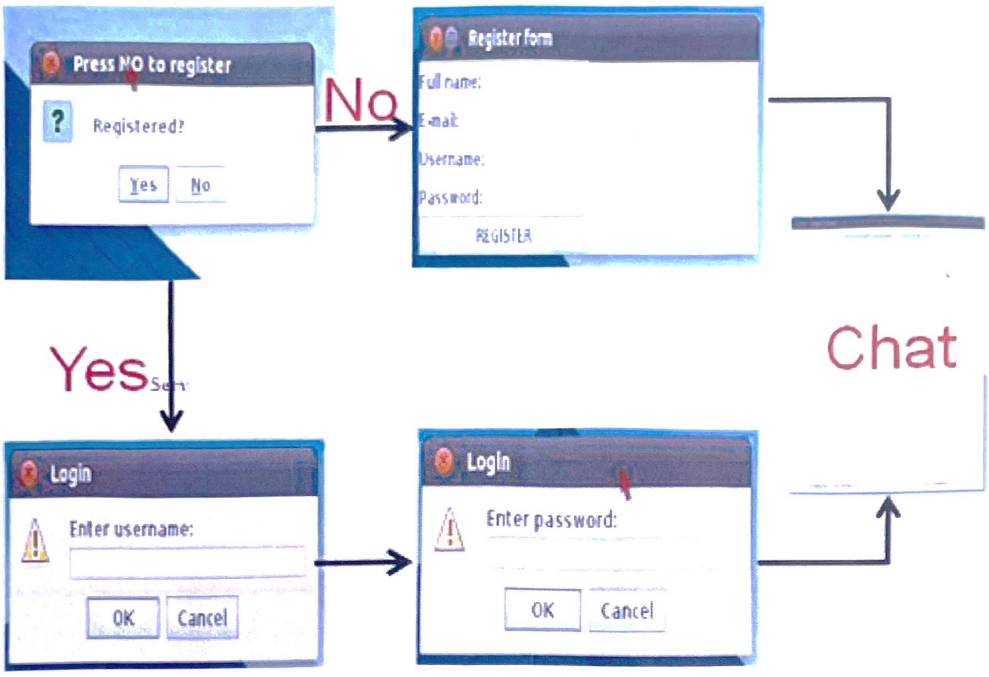


Figure 4.3: BAChat Client Registration

After successful login to the Chat Server users can start chatting. All messages sent by the user are displayed to all users until users start private messages. Sent messages are displays in the display text field. The messages between the users are not displayed / seen in the general chat display text field. The messages are displayed only within the private message display text field. As seen in the figure 4.4

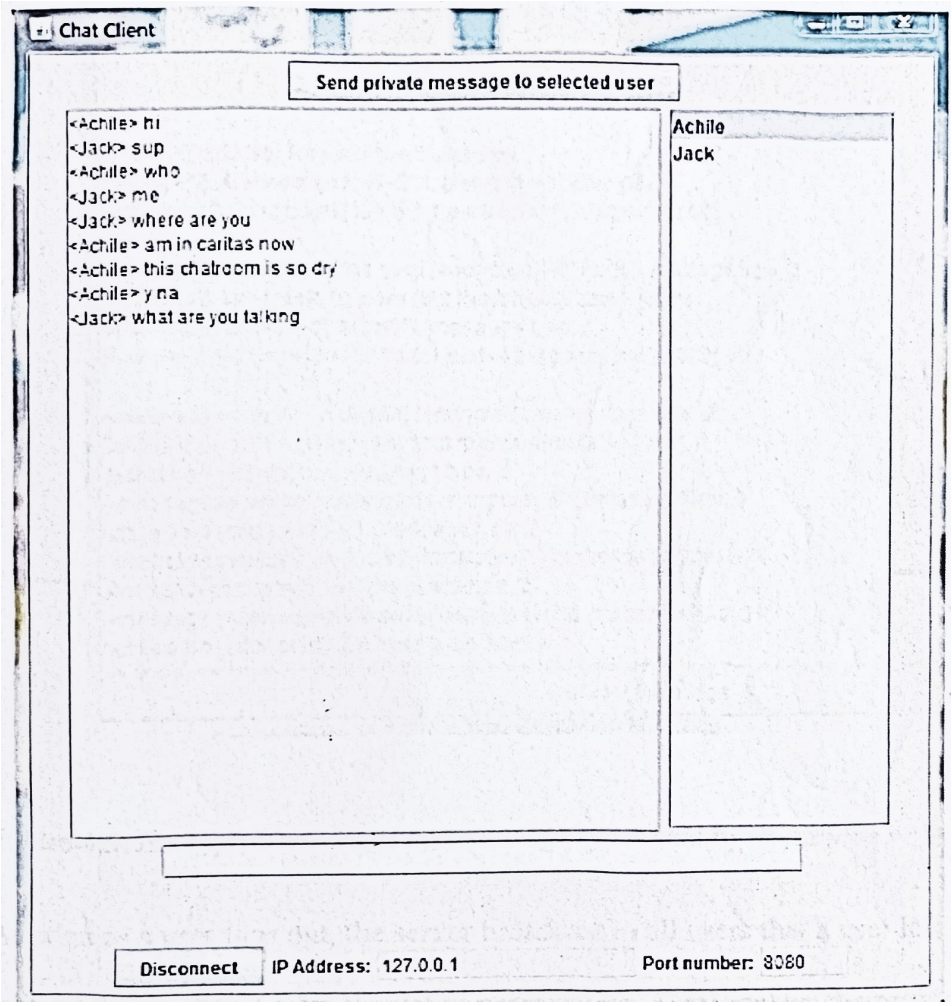


Figure 4.4:BAChatClient Room

The Database of Server, where database maintains record of all clients after confirming their registration. All login details and messages are encrypted. This way information from the Database is unreadable to whoever manages to access the Database. The original password of every client is stored in the Database, in the form of computed hash values of the password and salt (generated random number) that is stored in database.

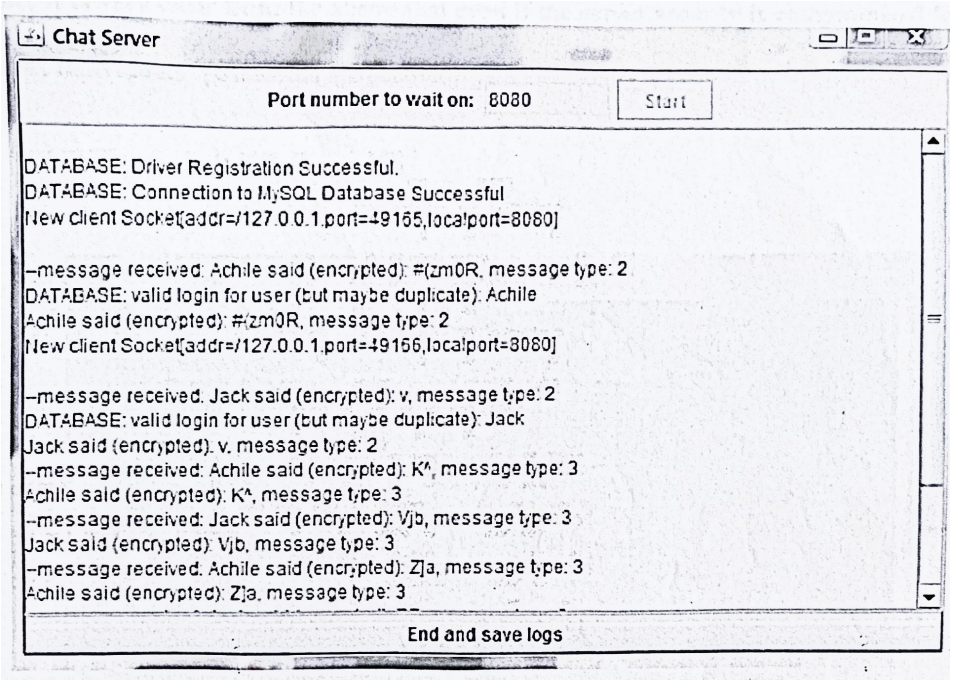


Figure 4.5:Encrypted messages and message types sent by users

As soon as a user logs out, the server broadcast to all users that a user left the chat room and remove the username from the list of users online. This displays the personal details of users stored in the MySQL database. The five number of columns in the user table (as in the figure below) represents: client names, clients email, client usernames, client passwords (Hash values), and clients random generated number (Salt).

Implementation of the SALT makes it extremely difficult to compute the original password since the values stored in the database is the hash values of the original password. In addition, to be able to get the values that represent the original password, the system needs to add the random generated number to the hash values and do some addition and subtraction. Therefore, it is very clear

from the above that even if the server security is compromised for any reason, individuals' passwords are secured.

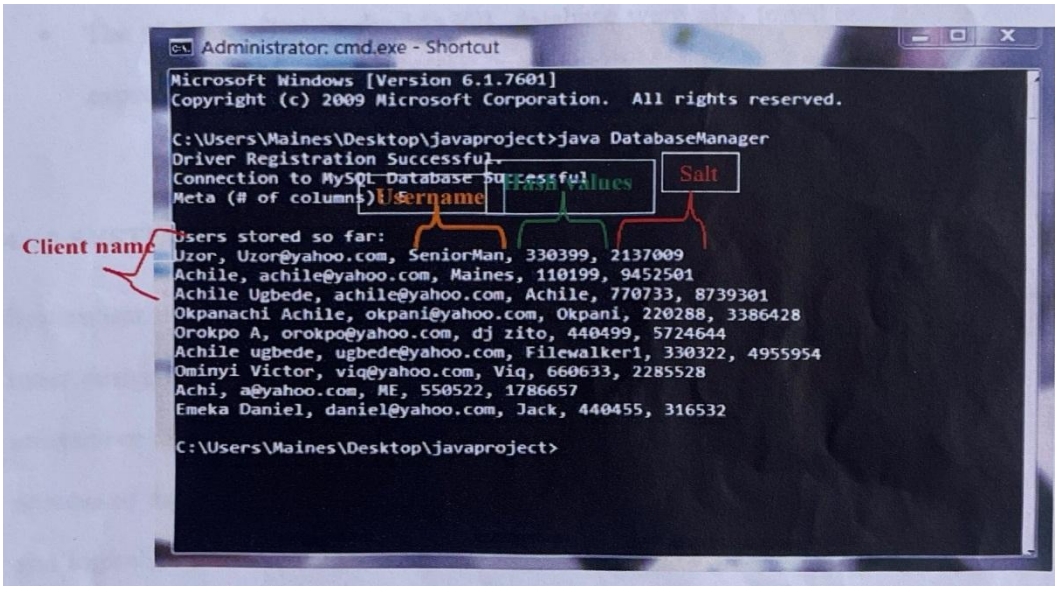


Figure 4.6:MySQL Database

4.4 TESTING

Testing of software is a test conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements

4.4.1 UNIT TEST

This is the test carried out on the individual component or part of a system. Unit tests were carried out on the Java program main classes.

- After compiling all the main classes of the java program, each class was tested and every error found was corrected. The classes tested are as follows: BAChatClient.java,BAChatServer.java, ChatFrame.java,

ChatMessage.java,ConnectionHandler.java,ConnectionManager.java, DatabaseManager.java,Encryptor.java,LoginDialog.java,PrivateChatFrame.java,RegisterDialog.java and ServerMainListener.java

- The tables created in the MySQL database were also tested and it worked properly as expected.

4.4.2 SYSTEM TEST

System test falls within the scope of black box testing and does not require the knowledge of the inner design of the code. In this case, software testing is any activity aimed at evaluating an attribute or capability of a system and determining that it meets its required results. It is also the process of executing a program or system with the intent of finding errors. Developing a good and logically sound test plan is vital to developing a bug free software system. The table below depicts 63 the various tests carried out, result of the tests and conclusions which were based on the results. The system test was carried out by integrating all the different Java classes together and linking them up with the data base. After this was done, the system was initialized and the Server chat application was started, the client chat application was also initialized. All the other classes were tested and they met the expected output.

4.5 INTEGRATION

To implement the application on a real network, MySQL must be installed on the server host and port used for the MySQL configuration when you install it should be port 3306 and database password=w1127449 and database Server IP=localhost.Java development kit (JDK) must also be installed all the source codes used be compiled without error. Thereafter, the user can start and use the same server IP address and Port number to be able to establish connection with the server.

CHAPTER FIVE

SUMMARY, RECOMMENDATIONS AND CONCLUSION

5.1 SUMMARY

Client server model is used to communicate over the network where the server is the system that provides services and clients are the systems that want to use these services to communicate with other client systems in the network. In this application, at server side a thread is created that receives numerous clients' requests. It also contains a list in which Client's name and IP addresses are stored. After that, it broadcasts the list to all the users who are currently in chat room and when a client logs out then server deletes that particular client from the list, updates the list and then broadcast the list to all available clients. A client firstly must have to register itself by sending username to the server and should have to start the thread so that the system can get the list of all available clients. Then any of two registered clients can communicate with each other.

5.2 LIMITATIONS

Some drawbacks of the Client-Server Cha are as follows:

- **i.** Time Constraint.
- **ii.** Financial Constraint.
- **iii.** In case of server fails then the users also suffers.
- **iv.** A lost password is irrecoverable.

v. As the server receives as many requests from clients so there is a chance that server can become congested and overloaded.

5.3 RECOMMENDATIONS

Instead of starting a new thread for each task to perform concurrently, the task can be passed to a thread pool. Thread Pools are useful when you need to limit the number of threads running in an application at the same time. There is a performance overhead cost linked with beginning a new thread, because each thread allocates some memory for its stack. This could not be implemented in this application because of time limit. Another suggestion for future works is the use of Java technology newest flavor of TCP Reno. This is because of its light weight and extreme advance features to overcome the flaws of the traditional Transmission Control Protocol. I recommend that for future works, Thread pool should be used instead of starting new thread for each task. TCP Reno should also be implemented in the future works due to its benefits as mentioned above.

CONCLUSION

A secured chat application has been developed with TCP. TCP is a connection oriented protocol, so once the connection is established there is no need to send socket address again and again. It is reliable, it guarantees that each packet will arrive and also guarantees that packets will be in the right order. This project use MySQL for its database to make information in the database secure. The personal details and messages including the private messages in the Database are encrypted using encrypt or (one of the security facilities available in the MySQL. As mentioned earlier, MySQL became the obvious choice for the implementation of Secured Java Client-Server Chat Application for the reasons that: MySQL provide secure connections between MySQL clients and the server using the Secure Sockets Layer (SSL) protocol (for the database) to provide secure data communication. The project implements hash function with the password before the encryption and then stored in the Database; this application also uses random generated numbers (salt) that is calculated together with the password hash values and stored in the Database. As a result, even if the database is compromised, the salt added to hash values makes it harder to compute the original password. When a random salt is used with the hash function, it significantly increases the strength of encrypting passwords thus makes cracking greatly more difficult. Therefore, makes the chat application server reliable and more secured. Real network connection is

another achievement of this application implementation. Another accomplishment of this application is private chatting. This is where two users can chat in private. The messages between the users are not displayed/ seen in the general chat display text field. The messages are displayed only within the private message display text field. It shows and keeps history of the private messages. As mentioned earlier, this application has been developed placing security of the network as priority, thus, look in details all aspect of network programming lapses, especially client-server chatting system.

REFERENCES

Bhatt, D.V.; Schulze, S.; Hancke, G.P.; "Secure Internet access to gateway using secure socket layer," vol.55,no.3,pp.793-800,June 2006

Forouzan,Behrouz A. "TCP/IP Protocol Suite." McGraw-Hill Education,2017. Multithreaded Client/Server Application

http://jerome.jouvie.free.fr/Java/Network/Tutorials/Tutorial2.php] (01/07/2010).

http://www.ase.md/~aursu/ClientServerThreads.html] Client-server Chat

Ming Xue;Changjun Zhu;"The Socket Programming and Software Design for Communication Based on Client/Server," PACCS '09. Pacific-Asia Conference on, vol., no.pp.775-777,16-17 May 2009

Schneier, Bruce. "Cryptography and Network Security: Principles and Practice." Pearson,2014. Java: "Java Socket Programming" JavaFX(for Java): "JavaFX Documentation"

Shukla,A.;Brecht,Tim.;"TCP Connection Management Mechanisms for Improving Internet Server Performance,"1st IEEE Workshop on, vol., no., pp.1-12,13-14 Nov.2006

Stevens,W.Richard,Bill Fenner, and Andrew M. Rudoff. "Unix Network Programming, Volume 1: The Sockets Networking API." Addison-Wesley Professional, 2003.

Zhenxing Liu;Lallie, H.S.; Lu Liu; Yongzhao Zhan; Kaigui Wu;"A hash-based secure interface on plain connection. vol., no., pp.12-39,17-19 Aug.2011