

CHAT APPLICATION

A PROJECT REPORT

Submitted by

SUBATHRA S (2303811724322109)

in partial fulfillment of requirements for the award of the course

CGB1201 – JAVA PROGRAMMING

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved
by AICTE, New Delhi)

SAMAYAPURAM – 621 112

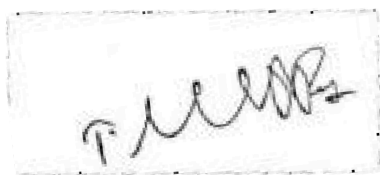
DECEMBER, 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

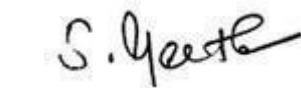
SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “ **CHAT APPLICATION**” is the bonafide work of **SUBATHRA S(2303811724322109)**who carried out the project work during the academic year 2024 - 2025 under my supervision.



Signature



Signature

Dr. T. AVUDAIAPPAN M.E.,Ph.D.,

HEAD OF THE DEPARTMENT,


Department of Artificial Intelligence,

K. Ramakrishnan College of Technology,

Samayapuram, Trichy -621 112.

Mrs. S. GEETHA M.E.,

SUPERVISOR,

Department  ligence,

K. Ramakrishnan College of Technology,

Samayapuram, Trichy -621 112.

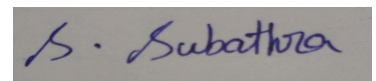
Submitted for the viva-voce examination held on
3.12.24

INTERNAL EXAMINER


EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**CHAT APPLICATION** ” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfillment of the requirement of the award of the **CGB1201 – JAVA PROGRAMMING**.



Signature

S.SUBATHRA

Place: Samayapuram

Date: 3/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and indebtedness to our institution, **“K. Ramakrishnan College of Technology (Autonomous)”**, for providing us with the opportunity to do this project.

I extend our sincere acknowledgment and appreciation to the esteemed and honourable Chairman, **Dr. K. RAMAKRISHNAN, B.E.**, for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director, **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding our project and offering an adequate duration to complete it.

I would like to thank **Dr. N. VASUDEVAN, M.TECH., Ph.D.**, Principal, who gave the opportunity to frame the project to full satisfaction.

I thank **Dr.T.AVUDAIAPPAN, M.E.,Ph.D.**, Head of the Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for providing her encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide **Mrs.S.GEETHA M.E.**, Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards.

MISSION OF THE INSTITUTION

- Be a centre of excellence for technical education in emerging technologies by exceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transform them as all- round personalities respecting moral and ethical values.

VISION AND MISSION OF THE DEPARTMENT

To excel in education, innovation and research in Artificial Intelligence and Data Science to fulfill industrial demands and societal expectations.

Mission 1: To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.

Mission 2: To collaborate with industry and offer top-notch facilities in a conducive learning environment.

Mission 3: To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.

Mission 4: To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

PEO 1: Compete on a global scale for a professional career in Artificial Intelligence and Data Science.

PEO 2: Provide industry-specific solutions for the society with effective communication and ethics.

PEO 3: Hone their professional skills through research and lifelong learning initiatives.

PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO 1:** Capable of working on data-related methodologies and providing industry-focussed solutions.
- **PSO2:** Capable of analysing and providing a solution to a given real-world problem by designing an effective program.

ABSTRACT

Chat applications have revolutionized communication by providing seamless, instant messaging solutions. These platforms enable users to exchange text, multimedia, and other data in real time, fostering connectivity across geographical boundaries. A well-designed chat application typically includes features like user authentication, message encryption, and an intuitive interface to ensure a secure and user-friendly experience. Such applications cater to both personal and professional communication needs, often serving as vital tools for collaboration and interaction. Modern chat applications are powered by robust technologies, including cloud-based architectures, WebSocket protocols, and advanced database management systems.

These technologies ensure high availability, scalability, and low latency, allowing users to communicate without interruptions. Additional functionalities such as group chats, video calls, and file sharing enhance the usability and versatility of these platforms. Moreover, the integration of AI-powered features like chatbots, sentiment analysis, and language translation broadens the scope of interaction, making communication more dynamic and inclusive.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	viii
1	INTRODUCTION	1
	1.1 INTRODUCTION	1
	1.2 OBJECTIVE	1
2	PROJECT METHODOLOGY	2
	2.1 PROPOSED WORK	2
	2.2 BLOCK DIAGRAM	3
3	JAVA PROGRAMMING CONCEPTS	4
	3.1 KEY CONCEPTS USED	4
4	MODULE DESCRIPTION	6
	4.1 NETWORK MODULE	6
	4.2 THREAD MANAGEMENT MODULE	6
	4.3 MESSAGE HANDLING MODULE	6
	4.4 USER INTERFACE MODULE	7
	4.5 DATABASE MODULE	7
5	CONCLUSION	8
	REFERENCES	9
	APPENDICES	10
	Appendix A – Source code	10
	Appendix B – Screen shots	13

CHAPTER 1

INTRODUCTION

N

1.1 INTRODUCTION

A chat application is a software platform that enables real-time communication through text, voice, or video. Designed for personal, professional, and business use, it facilitates instant messaging, file sharing, group chats, and multimedia exchanges. Features like end-to-end encryption ensure security, while cross-platform accessibility allows seamless communication across devices. Popular examples include WhatsApp, Slack, and Telegram. Chat applications enhance connectivity, collaboration.

1.2 OBJECTIVE

Facilitate Real-Time Communication: Enable users to exchange messages instantly, ensuring seamless interaction.

Enhance Connectivity: Bridge geographical barriers by allowing users to connect anytime and anywhere.

Support Collaboration: Promote teamwork through group chats, file sharing, and integrated productivity tools.

Ensure Security: Provide secure communication with encryption and privacy features.

Increase Accessibility: Offer cross-platform compatibility for uninterrupted communication on various devices.

Improve User Experience: Deliver intuitive interfaces, customization options, and multimedia support.

CHAPTER 2

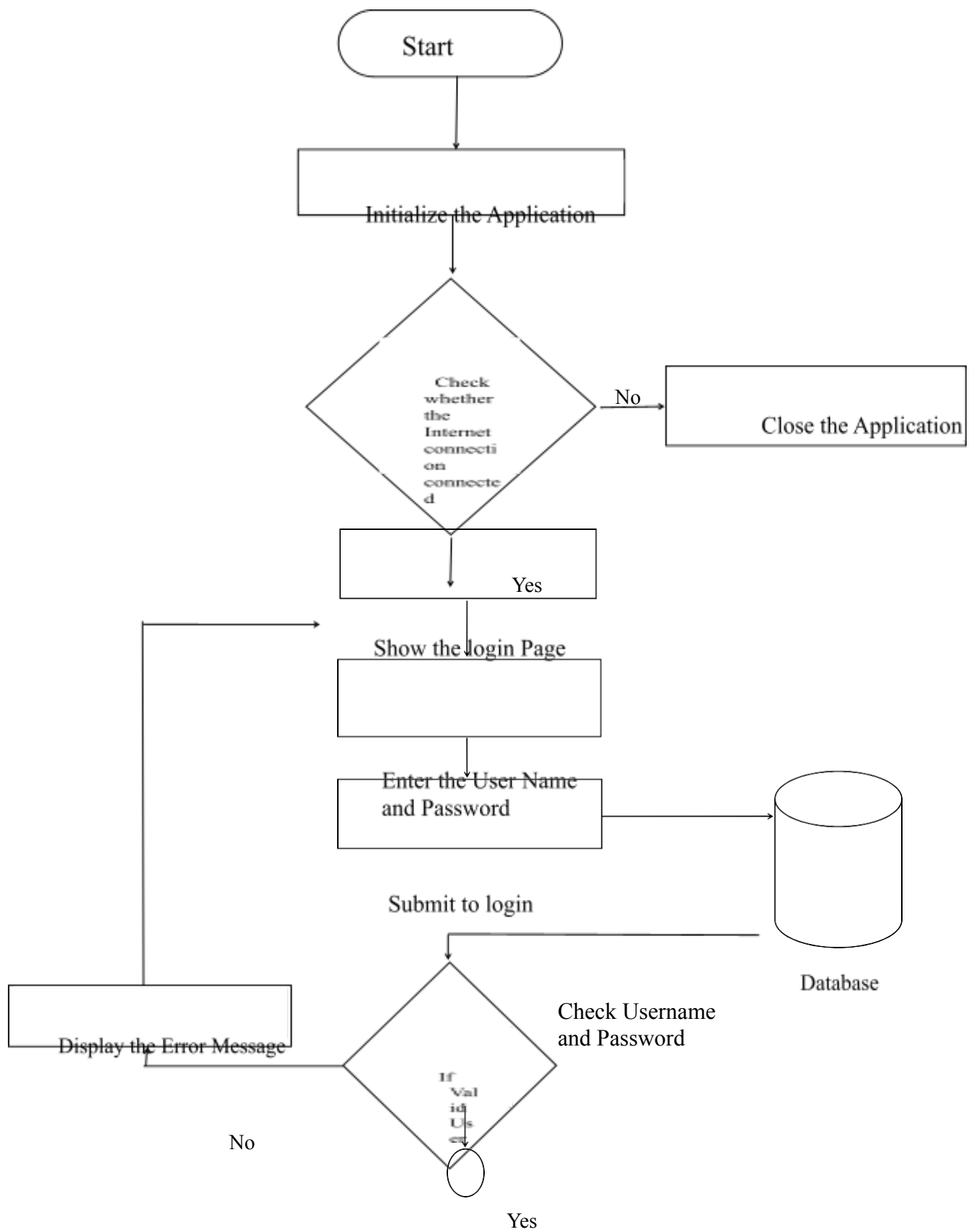
PROJECT

METHODOLOGY

2.1 PROPOSED WORK

The proposed work methodology outlines the systematic approach for developing a chat application. It emphasizes clear phases, each contributing to the creation of a robust, efficient, and user-friendly platform. The proposed work focuses on designing a scalable and user-friendly chat application architecture. The frontend will be developed using intuitive design principles, ensuring a seamless user experience. To ensure reliability, the application will undergo comprehensive testing. Functional testing will verify that features like message delivery, multimedia sharing, and group chats work as intended.

2.2 BLOCK DIAGRAM



CHAPTER 3

JAVA PROGRAMMING CONCEPTS

3.1 IMPLEMENTATION OF JAVA CONCEPTS

1. Classes and Objects:

Concept: Networking in Java involves using Socket and ServerSocket to establish communication between clients and servers over TCP/IP.

The server listens for incoming client connections using ServerSocket.

Each client connects to the server using Socket and exchanges messages.

2. Multithreading:

Concept: Multithreading allows the server to handle multiple client connections simultaneously.

Use a separate thread for each client connection to ensure non-blocking communication.

Utilize the Thread class or ExecutorService to manage threads efficiently.

3. Graphical User Interface (GUI):

Concept: Build a user-friendly interface using libraries like Swing or JavaFX.

Design a chat window with components like a message input box, send button, and chat history area.

Handle user interactions using event listeners.

4. Database Integration:

Concept: Store chat logs, user credentials, and settings in a relational database.

Use JDBC (Java Database Connectivity) to connect to databases like MySQL or SQLite.

Implement CRUD operations to store and retrieve chat history or manage user accounts.

4. Encryption and Security:

Concept: Security ensures that messages exchanged between users are protected from unauthorized access.

Use encryption libraries (e.g., Java Cryptography Extension or Bouncy Castle) to secure messages.

Implement SSL/TLS for secure communication channels between the client and server.

CHAPTER 4

MODULE

DESCRIPTION

4.1 NETWORK MODULE:

This module handles the communication between the client and server. It uses Java's networking classes, such as Socket and ServerSocket, to establish and maintain connections. The server listens for incoming client requests, while the client sends and receives messages. This module is responsible for ensuring reliable data transmission using TCP/IP protocols. includes a collection of tools, classes, and methods that enable network communication within a programming environment.

4.2 THREAD MANAGEMENT MODULE:

The thread management module ensures the application can handle multiple clients simultaneously. On the server side, each client connection is managed in a separate thread, allowing concurrent message processing. This module uses Java's threading mechanisms, such as the Thread class or ExecutorService, to ensure scalability and non-blocking operations. provides tools to create, manage, and control multiple threads of execution within a program.

4.3 MESSAGE HANDLING MODULE:

This module processes the messages exchanged between users. It includes features like message serialization and deserialization using ObjectOutputStream and ObjectInputStream. It ensures that messages are sent, received, and displayed correctly in the chat interface. The module also handles broadcasting messages to multiple clients in a group chat setup. A message handling module is designed to facilitate the exchange, processing, and management of messages between components in a system.

4.4 USER MANAGEMENT MODULE:

The UI module provides the graphical interface for the chat application. It uses libraries like Swing or JavaFX to design the chat window, including text input fields, message displays, and send buttons. This module manages user interactions, such as typing messages and clicking buttons, and dynamically updates the interface as messages are received. provides the tools and components necessary to create graphical or command-line interfaces for interacting with users.

4.5 DATABASE MODULE:

This module handles data persistence. It stores user credentials, chat history, and other metadata in a database. Using Java Database Connectivity (JDBC), this module interacts with databases like MySQL or SQLite to perform CRUD operations. It ensures data is securely stored and retrieved when needed, such as loading previous chat sessions or verifying user login credentials. self-contained package that provides functions for connecting to, querying, and managing a database.

CHAPTER 5

CONCLUSIO

N

Building a chat application involves integrating multiple components, including a robust network module for communication, a thread management module to handle concurrent users, and a message handling module for processing user messages efficiently. Each module plays a crucial role in ensuring the application is responsive, scalable, and secure. For example, the network module establishes reliable connections between clients and servers, while the thread management module allows simultaneous interactions among users without blocking resources. These foundational elements set the stage for creating a seamless user experience. The addition of a user interface module elevates the usability of the chat application, providing users with an intuitive and interactive platform to send and receive messages. This interface can range from simple command-line interactions to feature-rich graphical interfaces, making the application accessible to a broad audience. Coupled with a database module, which stores user data, message histories, and application configurations, the chat system gains persistence and data integrity. These modules collectively ensure the application is not only functional but also user-friendly and capable of maintaining long-term usage records.

REFERENCES:

1. "Developing Apps with GPT-4 and ChatGPT" by O'Reilly Media.
2. "Building Progressive Web Apps" by Tal Ater.
3. "The Book of Chatbots: From ELIZA to ChatGPT" by Maciej Cieřła.
4. "Mastering Xamarin.Forms" by Ed Snider

APPENDICES

APPENDIX A – SOURCE CODE

```
import java.awt.*;

import java.awt.event.ActionEvent;
import
java.awt.event.ActionListener;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import
java.net.ServerSocket;
import java.net.Socket;
```

ActionListener {

```
public class Main extends Frame
implements Runnable,
```

```
T
e
x
t
A
r
e
a
te
x
t
A
r
e
a;
T
e
```

```

x
t
F
ie
l
d
te
x
t
F
ie
l
d
;
B
u
tt
o
n
s
e
n
d
;
//ServerSo
cket
serverSoc
DataInputStream(socket.getInputStream());

dataOutputStream =
new
DataOutputStream(socket.getOutputStream());

ket;
Socket
socket;
DataInputStream
dataInputStream;
DataOutputStream
dataOutputStream;
Thread chat;
Main(){

    textField =
    new
    TextField();
    textArea =
    new
    TextArea();
    send = new
    Button("sen
d");
    send.addAct
ionListener(
this); try {
        //serverSocket = new
        ServerSocket( 12000);
        socket = new
        Socket("localhost",12000
); dataInputStream= new

```

```

    }

    catch (Exception e){

    }

    add(textField)
    ;
    add(textArea);
    add(send);
    chat = new
    Thread(this);
    chat.setDaemon(true);
    chat.start();

    setSize(500,500);
    setLayout(new
    FlowLayout());
    //setShape(short);
    setTitle("ser");
    setVisible(true);

}
public void actionPerformed(ActionEvent
e){ String msg = textField.getText();
textArea.append("ser:"+msg+"\n");
textField.setText("");
try {

        dataOutputStream.writeUTF(msg);
        dataOutputStream.flush();
    } catch (IOException ex) {

        throw new RuntimeException(ex);
    }

}

}

public static void main(String[] args) {

```

```
new Main();  
  
}  
public void run(){
```



```
while(true)
{ try{
    String msg = dataInputStream.readUTF();
    textArea.append("cli:"+msg+"\n");
}
catch(Exception e){

}

}

}
```

APPENDIX B - SCREENSHOTS

