A star with a symbol in the middle

AI-generated content may be incorrect.

**TRIBHUVAN UNIVERSITY**

**FACULTY OF HUMANITIES AND SOCIAL SCIENCES**

**“WEB-BASED CHAT APPLICATION”**

**A PROJECT REPORT**

**Submitted To**

**Department of Computer Application**

**Asian College of Higher Studies**

**Ekantakuna, Lalitpur**

***In partial fulfillment of the requirements for the Bachelors in Computer Application***

**Submitted By**

**Bijay Khadka [210606]**

**Sijan Khadka [210612]**

A star with a symbol in the middle

AI-generated content may be incorrect.

**TRIBHUVAN UNIVERSITY**

**FACULTY OF HUMANITIES AND SOCIAL SCIENCES**

**ASIAN COLLEGE OF HIGHER STUDIES**

**SUPERVISOR’S RECOMMENDATION**

I hereby recommend that this project prepared by Mr. Bijay Khadka and Sijan Khadka under the supervision of Mr. Pesal Rai entitled ‘Web-based Chat Application’ in partial fulfillment of the requirement for the degree of Bachelor of Computer Application be processed for the evaluation.

……………………………

Mr. Pesal Rai

Supervisor

Department of Computer Application

**LETTER OF APPROVAL**

This is to certify that this project prepared by Bijay Khadka and Sijan Khadka entitled “Web-based Chat Application” in the partial fulfillment of requirement for the degree of Bachelor in Computer Application has been evaluated. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

|  |  |
| --- | --- |
| **Signature of Supervisor**  ………………………………..  **Mr. Pesal Rai**  Asian College of Higher Studies | **Signature of Program Co-ordinator**  ………………………………..  **Mr. Pranaya Nakarmi**  Asian college of Higher Studies |
| **Signature of Internal Examiner**  ………………………………..  **Mr. Janak K. Lal**  Asian College of Higher Studies | **Signature of External Examiner**  ………………………………..  **Mr.**  Tribhuvan University |

# ABSTRACT

Chat applications are ubiquitous for social networking and communication purposes. However, not all chat applications provide sufficient security and privacy for their users

A chat application has two primary components, a server and a client. A server is a computer program or hardware that provides services and functionality to other programs or devices. Clients, who want to chat with each other, connect to the server to communicate. The chat application we aim to build will operate like a

# ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude towards our supervisor, Mr. Pesal Rai, for his invaluable guidance, knowledge, expertise and suggestions throughout our project. His constant support and encouragement have been crucial in shaping this project and bringing it to completion. His extensive knowledge and insights have helped us in making informed decisions and overcoming challenges while building this project.

We would also like to extend our thanks to our colleagues and teachers who have been a significant support in finalizing the project in a limited time frame. Their commitment, dedication, and hard work have been instrumental in ensuring the success of this project. Their timely help and guidance have helped us in completing the project successfully. Lastly, we would like to thank the college, ACHS, for providing us with an excellent ambiance and the right amenities. The college has been an essential part of our growth and learning, and we are grateful for the opportunities that it has provided us.

Once again, we express our sincere thanks for your unwavering support and guidance throughout the project.

Table of content

# LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| API | Application Programming Interface |
| CSS | Cascading Style Sheets |
|  |  |
|  |  |
|  |  |
|  |  |

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1: INTRODUCTION

## Introduction

Messaging has become a part of our everyday lives in part due to its convenience for real-time chat communication and simple-to-use functionality. For instance, an iOS or text message on an iPhone or Android device from a friend, an email from a co-worker on Microsoft or Gmail, a team chat in a Slack or Microsoft Teams workspace, or even instant messaging through social media. These messaging and real-time chat applications play an important role in how the world interacts today, due to their immediacy and vast capabilities. A chat application makes it easy to communicate with people anywhere in the world by sending and receiving messages in real time. With a chat app, users are able to receive the same engaging and lively interactions through custom messaging features, just as they would in person. This also keeps users conversing on your platform instead of looking elsewhere for a messaging solution. Whether it's private chat, group chat, or large-scale chat, adding personalized chat features to your app can help ensure that your users have a memorable experience.

## Problem Statement

This project is to create a chat application with a server and users to enable users to chat with each other and develop an instant messaging solution to enable users to seamlessly communicate with one another.

Real time communication : A reliable and secure chat plays a huge role in ensuring a positive user experience in your app. Making sure that messages between active users are being sent and received in real-time fosters an engaging chat where users can immediately interact with each other.

Secure communication : When dealing with sensitive information like compliance requirements or personal user information, making sure that your real-time chat application has the proper security features in place is crucial for ensuring that the data of those using your app is protected. In a chat application, the messages shared between the users should be encrypted, i.e., there should be implementation of end-to-end encryption

## Objectives

This project aims to create a system which will focus on having a user-friendly application

with simple interface and design which can be easily understood by any user. The main objectives of this system are as follows:

1. To develop a web-based chat application where users are able to communicate with each other with ease.
2. To encourage user friendliness with interactive user interface such that it is very easy to use enabling even a novice person to use it.
3. To build a secure chat application with end-to-end encryption where the messages communicated between the users are encrypted using encryption algorithm.

## Scope and Limitation

### 1.4.1 Scope

1. This chat application will enable the communication between two users using point to point communication.
2. Users can send message to other users while maintaining the optimum privacy with the help of end-to-end encryption.

### 1.4.2 Limitation

1. This application does not support audio and video conversations.
2. Transfer of files are not supported.

## Report Organization

**Chapter 1:**

This Chapter describes the initial section of a project, where the problem statement, project objectives, project scope, and limitations are introduced. These details offer the reader a broad understanding of the project. The problem statement is the issue that the project aims to solve, and the objectives describe the specific goals of the project. The project scope outlines the boundaries of the project, while the limitations describe any factors that could potentially restrict the project's success.

**Chapter 2:**

This Chapter involves conducting a thorough review of existing literature and background research relating to similar projects and the fundamental concepts that are integral to the current project. This process helps in gaining an understanding of the subject matter along

with identifying any gaps in knowledge and formulating a comprehensive plan for the project.

**Chapter 3:**

This chapter outlines the components of the project, which include an assessment of its feasibility, an analysis of its functional and non-functional aspects, as well as the schema and architectural design, and the data and process modeling diagrams. In this chapter, the UML Diagrams are discussed in detail for illustrating the design of the Chat Application. The various types of diagrams covered include object diagram, class diagram, activity diagram, sequential diagram, state diagram, component diagram, deployment diagram, and architecture diagram. Each diagram is explained in detail, providing a comprehensive understanding of their purpose and how they contribute to the overall design of the Chat Application.

**Chapter 4:**

The project's testing and implementation phase encompasses the creation of test cases that verify the functionality of the system and its components as intended throughout the development process. Additionally, it comprises the tools utilized in constructing the project.

**Chapter 5:**

This section encompasses the final remarks of the entire project. It presents the project's outcome and highlights the acquired knowledge and skills during the project development. Additionally, this chapter provides future recommendations for the project.

# CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

## Background Study

The emergences of computer networks and communication technologies allows people to communicate in a new way. Chatting is a method of using technology to bring people and ideas together despite geographical barriers. The technology has been available for years but the acceptance was quite recent. The chat application will allow multiple users to connect to the server and chat with all other online users. This project will focus on providing security with high quality usability experiences to users. The application works in a broadcast fashion. This means that messages from a user's are broadcasted to other users. Messaging application are growing in popularity. The past few years have brought application like WhatsApp, Telegram, Messenger, etc.

Web-based chat applications have also become a common part of modern communication, enabling real-time interactions over the internet. Examples of such platforms include WhatsApp Web, Facebook Messenger, Slack, and Telegram Web. These systems provide features such as text messaging, file sharing, voice/video calls, and emoji integration. Despite their widespread use, ensuring secure communication is a persistent challenge, especially due to the increasing threats to user privacy and data breaches.

## Literature Review

In today's digital age, secure messaging applications have become vital tools for personal and professional communication, offering a blend of convenience and privacy. As more users rely on messaging services across multiple devices, including desktops and browsers, understanding the varying security protocols and privacy features of these platforms becomes crucial. Despite significant advancements in encryption technology, the web-based versions of popular messaging apps often face challenges in providing comprehensive end-to-end encryption (E2EE), raising concerns about data privacy and vulnerability to attacks. This review explores the security mechanisms employed by three widely used messaging platforms—WhatsApp Web, Facebook Messenger, and Telegram Web—examining their encryption standards, data storage practices, and the limitations associated with their web interfaces.

1. **WhatsApp Web** is an extension of the mobile app that mirrors the phone's messages on a browser interface. It uses end-to-end encryption (E2EE) through the Signal Protocol, meaning messages are encrypted on the sender's device and decrypted only on the recipient's device. The web version depends on the phone's active internet connection and does not store messages on the browser or server. Although secure, it relies heavily on device synchronization, which can sometimes be a limitation.
2. **Facebook Messenger**, accessible via messenger.com, supports features like messaging, video/audio calling, and media sharing. While it uses Transport Layer Security (TLS) to encrypt data in transit, end-to-end encryption is only available through the “Secret Conversations” feature, and even then, only on the mobile app. The web version does not support E2EE and stores messages on Facebook's servers, which could pose privacy concerns despite the platform’s security infrastructure.
3. **Telegram Web** is known for its speed and cloud-based message synchronization. While it uses its own MTProto encryption protocol, it does not offer end-to-end encryption on the web interface. Only “Secret Chats” on the mobile app support E2EE. Standard web chats are encrypted between client and server but remain stored on Telegram’s servers, which may not meet the strictest privacy standards.

In summary, while these platforms implement various cryptographic techniques to ensure user privacy and message security, their web-based versions often fall short of offering full end-to-end encryption. Most rely on server-side storage and encryption during transmission, which, while effective against basic threats, may not be sufficient against sophisticated attacks or unauthorized server access. This gap highlights the ongoing need for secure, privacy-focused web-based chat applications that offer strong encryption across all platforms.

# CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

## System Analysis

It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

### 3.1.1 Requirement Analysis

### Functional Requirements

A diagram of a program

AI-generated content may be incorrect.Functional requirements determine the provision of the system and the system’s processing. It defines what the system should do in order to meet the user’s needs

The web-based chat application provides a seamless user experience with essential features for communication and account management. User authentication is required, allowing users to sign up using a unique username, password or log in if they already have an account. The user's email serves as the unique identifier for the account. Once logged in, users can easily log out from the dashboard, ensuring secure access. The application must support one-to-one chat, allowing users to communicate in real-time with other registered users. It should also display the online/offline status of users, enabling communication with those currently active. The system must allow users to load old chats when selecting a particular recipient, ensuring a seamless continuation of conversations. Finally, users should be able to delete chats, with the option to remove their own messages, and ensure that this deletion is reflected in both the chat interface and the underlying database. This set of features ensures a smooth, secure, and interactive experience for all users.

### Non-functional Requirement

Non-functional requirements are the set of specifications that describe the system’s operation capabilities and constraints and attempt to improve its functionality. These are basically the requirements that outline how well it will operate including things like speed, security, reliability, data integrity, etc.

1. **User friendly:** The chat application should have interactive and appropriate user interface for the users to use the application with ease.
2. **Easy Accessibility:** The chat application can be accessed easily from any place at any time using any browser.
3. **Security:** The chat application should provide security and optimum privacy to the messages being transmitted through the application so that personal credentials and crucial messages are not leaked.
4. **Realtime communication:** Real-time communication enables users to communicate with each other in real-time, regardless of their physical location, which is critical for business collaborations or personal interactions. It also enhances user engagement and satisfaction and makes the chat application more useful and productive.

### 3.1.2 Feasibility Analysis

The feasibility study assesses the practicality and viability of developing a web-based chat application that integrates cryptographic algorithms for secure communication. It evaluates the proposed system from various dimensions including technical, operational, economic, legal, and schedule feasibility.

1. Technical Feasibility:

The proposed chat application is technically feasible with current technologies. Web development frameworks such as React, Angular, or Vue.js for the frontend, and Node.js, Django, or Express.js for the backend, support the real-time communication requirements. Technologies like WebSockets or Socket.IO can be used to establish real-time bi-directional communication between users. Cryptographic algorithms such as AES (Advanced Encryption Standard) for symmetric encryption, RSA or Elliptic Curve Cryptography (ECC) for asymmetric encryption, and SHA-256 for hashing can be implemented using widely available libraries (e.g., CryptoJS, Web Crypto API). As a result, the encryption and decryption processes can be handled effectively in the browser and on the server, ensuring secure message transmission.

1. Operational Feasibility:

The operational aspect focuses on how effectively the system can function in a real-world environment. The target users—individuals, teams, or organizations—will find the application easy to use with a clean interface and minimal setup. Users can register, log in, and communicate securely without needing to understand the underlying encryption methods. Administrative tools for managing user accounts, sessions, and system monitoring can ensure reliable operation.

1. Economic Feasibility:

From a financial standpoint, the system is economically viable, especially if it starts as an open-source or small-scale solution. The initial investment includes development tools, hosting services (like AWS, Firebase, or DigitalOcean), and SSL certificates for secure HTTPS communication. The use of open-source libraries for encryption and frameworks for development significantly reduces costs. In terms of return on investment (ROI), the platform could generate revenue through subscriptions, premium features, or business integrations. If adopted widely, the system could replace or compete with existing secure communication tools in niche markets.

### 3.1.3 Object Oriented Analysis

Object Oriented Analysis (OOA) is the first step of Object-Oriented Analysis and Design (OOAD), which is a software engineering methodology that involves using object-oriented concepts to design and implement software systems. OOA introduces new concepts to investigate a problem, such as classes, objects, attributes, methods, inheritance, polymorphism, and encapsulation. OOA uses visual modeling techniques, such as UML diagrams and use cases, to guide stakeholder communication and product quality.

### Object Modelling (Class and Object Diagram)

### Class Diagram

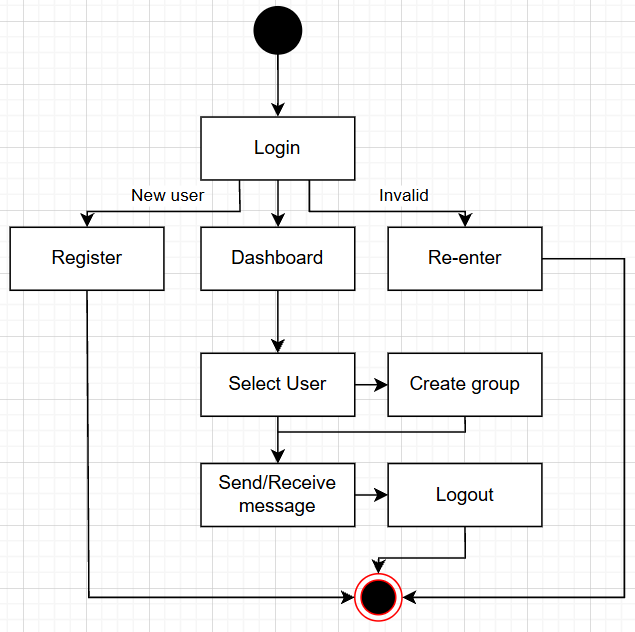
A class diagram is a UML structural diagram that shows the classes and their relationships in a system. A class diagram is used for abstract and conceptual modeling of the classes, their attributes, methods, and associations. A class diagram can also show the constraints and operations that apply to the classes.

### Object Diagram

An object diagram in a UML structure diagram that shows the instances of the classes and their relationships in a system at a specific point in time. An object diagram is used for concrete and instance modelling of the objects and their interactions. An object diagram can also show the values of the attributes and links between the objects.

### 3.1.4 Dynamic Modelling (State and Sequence Diagram)

A state diagram, also known as a state machine diagram, is used to model the behavior of an object or system in response to external events or stimuli. It shows the different states that an object can be in and the events that cause it to transition from one state to another. State diagrams are useful for modeling complex behaviors, such as those found in embedded systems or control systems.

A sequence diagram is used to model the interactions between objects in a system, showing the messages that are passed between them over time. It shows the order of events and the flow of control between objects. Sequence diagrams are useful for understanding the interactions between objects in a system and can help in identifying potential problems or bottlenecks in the system.

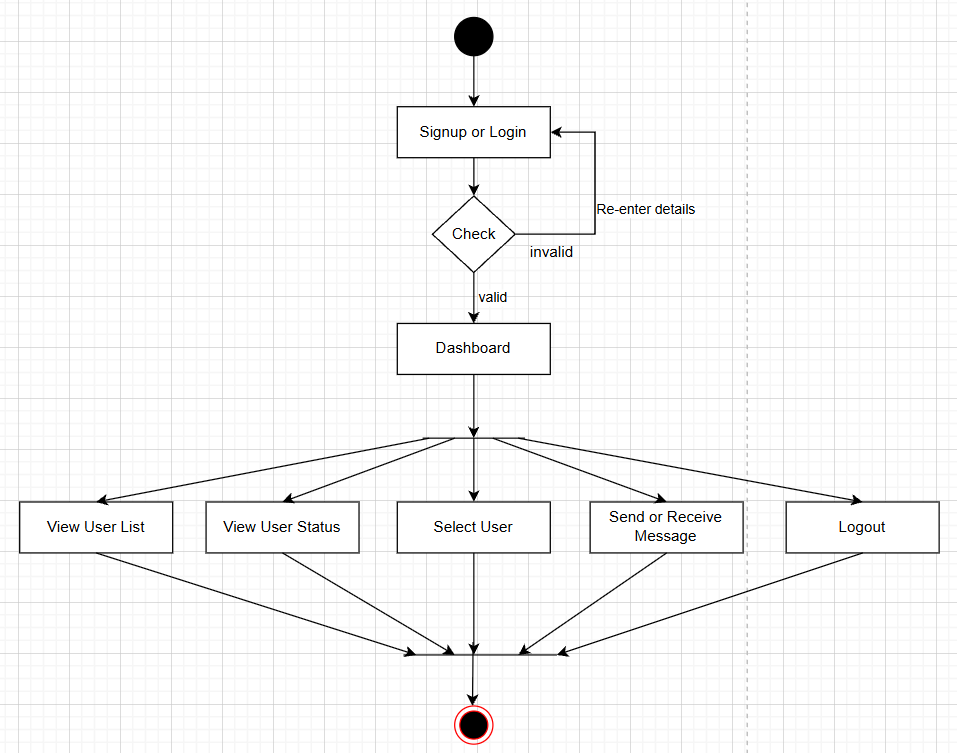
First of all, the user goes to the login page. If the user is new to the system, then the user needs to register and after that the user can login to the system dashboard, if the details are invalid it shows an error message, then the user needs to re-enter the valid information and log-in. After that, the user can see the list of other users in a dashboard where the user can select the user and can send the messages or also can receive the message from other users. At last, the user can also logout from the system.

**A diagram of a server

AI-generated content may be incorrect.**

To begin with, if the user is new to the system, the user needs to register by filling the registration form. After registering, the user can login to the system with the valid information. After login, the user can see the list of other users with their status and can select the user to send a message. After sending the message, the message is encrypted with the encryption method and saved in the database and after that the encrypted message is received by the server from the database and decrypted, then the actual message is received by the user.

### 3.1.5 Process Modelling

Activity diagrams are used to represent complex processes or workflows in a graphical way, with nodes representing actions or steps in the process, and arrows representing the flow of control between them. They can be used to model business processes, software workflows, or any other type of process that involves a series of actions or steps.

Initially, a user can login to the system by filling the login form or also can sign up if the user is new to the system. If the login information is valid then the user is sent to the dashboard or if the information is invalid then the user is needed to re-enter the details. In the dashboard page the user can see the list of other users with their online/offline status where the user can select the other users to send or receive messages and also the user can logout from the system.

## System Design

Design is the process of designing a software system using object-oriented concepts and principles. OOD is the second phase of the object-oriented development cycle, which includes requirements gathering, analysis, design, implementation, testing, and maintenance.

### 3.2.1 Class Diagram

In a class diagram, each class is represented as a box with the class name written inside. The attributes of the class are listed below the class name, and the methods of the class are listed below the attributes. The relationship between classes are represented by lines connecting the boxes, with different types of lines representing different types of relationships.

In this class diagram, we have two classes:

* Users: represents users of the chat app
* Chats: represents a single chat session between two or more users

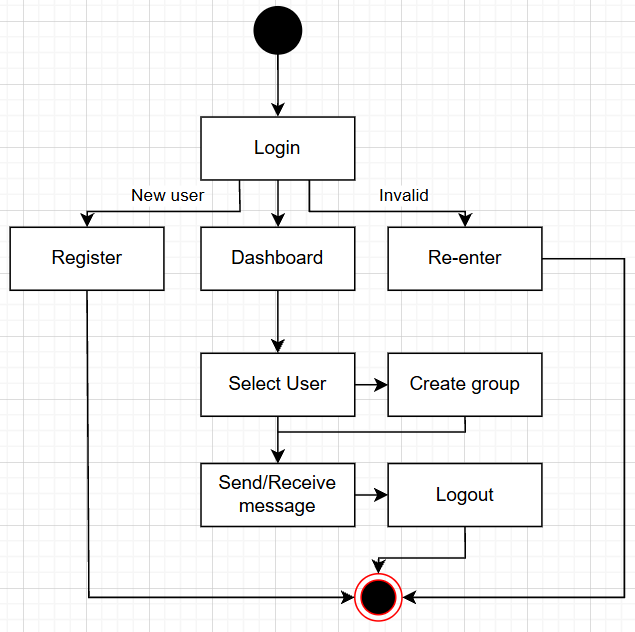
Each User object has several attributes, including u\_id, name, email, image, password, and is\_online. The is\_online attribute indicates whether the user is online or offline. Each chat object has four attributes: m\_id, sender\_id, receiver\_id and messages. The sender\_id and receiver\_id indicate the id of the users that shows the message is sent from whom and received by whom.

### 3.2.2 Object Diagram

In an object diagram, the objects (instances) of the classes in the corresponding class diagram are shown as boxes, with the name of the object and its class listed inside the box. The attributes and values of each object are also shown. Object diagrams are useful for visualizing and testing specific instances of a system, as well as for debugging and documentation purposes. They can be used to show how the objects in a system interact with each other, and to identify potential errors or design issues in the system.

In this diagram, we have two objects representing the users having attributes u\_id, name, email, password and is\_online status, connected to a single object named as chat having attributes as m\_id, sender\_id, receiver\_id and the messages. representing the chat room where the conversation is taking place. The chat object represents the message hello was sent from user to user2.

### 3.2.3 State Diagram

In a state diagram, the possible states of the object or system are represented as nodes, with the transitions between the states represented as arrows. The events that trigger the transitions are labeled on the arrows, and the conditions or actions that occur when a transition is taken can also be specified.

First of all, the user goes to the login page. If the user is new to the system, then the user needs to register and after that the user can login to the system dashboard with the valid information and if the details are invalid it shows an error message. After login, the user can see the list of other users in a dashboard and also can select the user from the userlist to whom the user wants to send the message or also can receive the message. At last, the user can also logout from the system.

### 3.2.4 Sequence Diagram

In a sequence diagram, the interaction between objects in a system is modelled showing the messages that are passed between then over time. It shows the order of the events and the flow of control between objects.

A diagram of a diagram

AI-generated content may be incorrect.

**A diagram of a server

AI-generated content may be incorrect.**To begin with, a user clicks on a register then a request is sent to a server and a server returns a register page to a user where the user can fill the information and register. After registering, the information is sent to the server where the server checks whether the information is valid or not, if invalid it shows an error message. And if the information is valid then the information is saved in a database. After that, the user goes to the login page and enters the valid email and password and then the server checks the information from the user database if the information is correct then the user is sent to the home page.

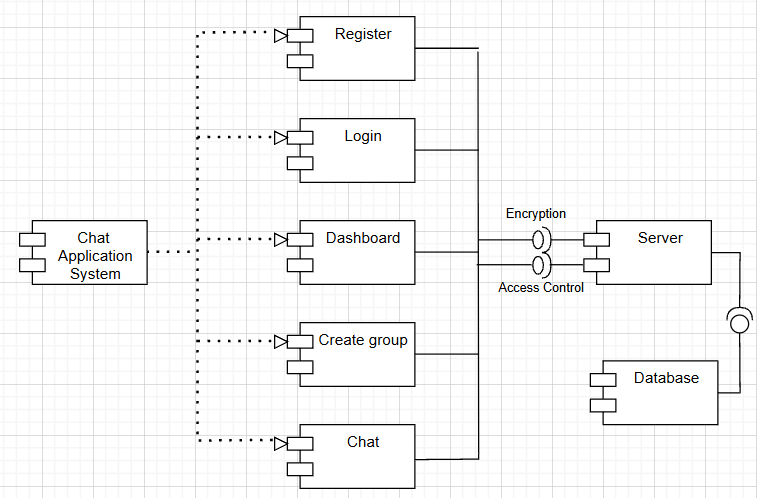
In the home page there is a list of other users and a user can choose any users to send messages and after sending the message, it is encrypted by the encryption algorithm and saved in the database. And after that the server gets the encrypted message from the database and sends it towards the receiver which is decrypted in original form

### A diagram of a software flowchart AI-generated content may be incorrect.3.2.5 Activity Diagram

Initially, the user fills the login form and if the information is valid then the user is sent to the home page, if the details are invalid it shows an error message, then the user needs to re-enter the information. And, If the user is new to the system, then the user needs to register and after that the user can login to the system dashboard. After that the user can see the status of other users, a list of users and also can select the user to send or receive messages. At last, the user can also logout from the system.

### 3.2.6 Component Diagram

Component diagrams are UML diagrams that describe the organization and wiring of the physical components in a system. They show the components and their dependencies, interfaces, ports, and relationships. They can help your team to imagine the system's physical structure, pay attention to the system's components and how they relate, and emphasize the service behavior as it relates to the interface.



In this diagram, we have two main terms: -

* Client: represents the front-end of the chat application, responsible for displaying the user interface and handling user input. The client component includes register, login, dashboard, select user, chat.
* Server: represents the back-end of the chat application, responsible for processing user requests and managing the chat sessions. The server component includes server and database.

### 3.2.7 Deployment Diagram

Deployment diagrams are UML diagrams that show the physical arrangement of the components of a system or application and how they are connected. They are used to model the hardware and software architecture of a system and to show the distribution and communication of its artifacts. They consist of elements such as nodes, artifacts, associations, and components.

In this diagram, there is a one node named chat application having two attributes as user interface and database server where user interface includes login, register, dashboard, select user and send or receive message, and the database server includes user details, database interface and MongoDB server.

### 3.2.8 Algorithm Detail

# CHAPTER 4: IMPLEMENTATION AND TESTING

## Implementation

Implementation is the carrying out, execution, or practice of a plan, a method, or any design, idea, model, specification, standard or policy for doing something. This implementation phase involves integrating various tools and technologies to develop both the back-end and front-end of the system, along with the model training and storage functionalities.

### 4.1.1 Tools Used:

**Case Tools**: Case tools are a set of software application programs, which are used to automate SDLC activities. Case tools are used by software project managers, analysts, and engineers to develop a software system. There are several case tools available to simplify various stage of the software development life cycle such as analysis tools, design tools, project management tools, database management tools, documentation tools, etc.

1. **HTML5**

HTML5 is the latest version of the standard markup language used to create and structure web pages. It introduces new elements and APIs for multimedia, graphics, and form handling, reducing reliance on external plugins like Flash. HTML5 is cross-platform and mobile-friendly, making it the foundation for modern web applications. It defines the layout and content structure using tags and attributes.

1. **CSS3**

CSS3 is the most recent version of Cascading Style Sheets used for styling HTML content. It introduces new features such as transitions, animations, media queries, and flexbox/grid layout systems. CSS3 enables responsive design, allowing websites to adapt to different screen sizes and devices. Developers can define colors, fonts, spacing, and element positions. It separates content structure (HTML) from presentation for better maintainability.

1. **Tailwind CSS**

Tailwind CSS is a utility-first CSS framework that allows developers to build custom user interfaces quickly by composing classes directly in HTML. Instead of relying on pre-designed components, Tailwind provides low-level utility classes for controlling layout, spacing, colors, typography, and more. Tailwind is highly customizable through configuration files, enabling teams to define their own design tokens

1. **JavaScript**

JavaScript is a versatile, high-level programming language used for making web pages interactive. It runs in the browser and allows developers to implement features like dynamic content, animations, form validation, and API calls. JavaScript works closely with HTML and CSS and is supported by all modern browsers. With the rise of frameworks (like React) and Node.js, it is now also used for server-side programming. A core language in full-stack web development.

1. **Node.js**

Node.js is an open-source, server-side runtime environment that lets developers run JavaScript on the backend. Built on Google’s V8 engine, it uses an event-driven, non-blocking I/O model for high performance. Node.js is ideal for real-time applications like chat apps, APIs, and streaming services. It includes npm, a package manager for easily installing third-party libraries.

1. **MongoDB**

MongoDB is a NoSQL, document-oriented database that stores data in flexible, JSON-like documents. It’s schema-less, allowing for rapid development and easy scalability. MongoDB is ideal for handling unstructured or semi-structured data and supports advanced queries, indexing, and aggregation. MongoDB enables fast data access and is well-suited for cloud-based applications.

1. **Socket.IO**

Socket.IO is a JavaScript library that enables real-time, bidirectional communication between web clients and servers. It uses WebSockets under the hood but falls back to other methods like polling when necessary. Socket.IO is commonly used for applications that need instant updates like chat apps, live dashboards, and multiplayer games. It handles events and messages between the client and server seamlessly.

1. **VS Code**

Visual Studio Code (VS Code) is a lightweight, powerful code editor developed by Microsoft. It supports multiple languages, extensions, and development workflows. Key features include syntax highlighting, debugging tools, IntelliSense (code completion), Git integration, and terminal access. VS Code is a go-to IDE for developers around the world.

1. **Draw.io**

Draw.io is a free, web-based diagramming tool that allows users to create various types of diagrams such as flowcharts, UML, ER diagrams, and mind maps. It features a drag-and-drop interface with a wide range of shapes, symbols, and templates. Diagrams can be saved locally or to cloud storage services like Google Drive or OneDrive. It is open-source and works entirely in the browser, requiring no installation

1. **Git**

Git is a distributed version control system that helps developers track changes in their source code over time. It allows multiple developers to collaborate efficiently on projects by managing different branches of code. Git records a history of every change made, enabling easy rollbacks and code reviews. It supports features like branching, merging, and tagging. Git is essential for team-based and open-source software development.

### 4.1.2 Implementation Methodology

1. **Setting up the Application**

Run npm init -y on your terminal. This creates a package.json file.

Now for this application, we will need to install some packages to get started**. npm install express mongoose ejs bycrypt body-parser multer express-session dotenv socket.io**

These packages provide is with the following:

* **Express** is an Express application, which is necessary for our Express server
* **EJS** (Embedded JavaScript) is a popular templating engine for Node.js that allows developers to create dynamic HTML pages by embedding JavaScript code directly into the HTML.
* **Mongoose** is an Object Data Modeling (ODM) library for Node.js and MongoDB that connects our application to MongoDB.
* **Bycrptjs** handles encrypting passwords.
* **Body-parser** is Node.js body parsing middleware. It parses incoming request bodies in a middleware before your handlers, available under the req.body property.
* **Multer** is a node.js middleware for handling multipart/form-data, which is primarily used for uploading files. It is written on top of busboy for maximum efficiency.
* **Express-session** is a middleware for the Express.is web framework that provides session management functionality. Sessions are a way to store data between HTTP requests in a stateless web application.
* **dotenv** is a popular pm package for managing environment variables in Node.js applications. Environment variables are key-value pairs that are stored outside the application code and can be accessed by the application at runtime. They are used to store sensitive information such as passwords, API keys, and database credentials.
* **Socket.IO** enables real-time bidirectional event-based communication. It consists of: a Node.js server (this repository) and a Javascript client library for the browser (or a Node.js client).

After this is complete, a node module folder is created where all the packages are downloaded.

1. **Setting up the Server**
2. **Creating Routes, Views and Controllers**

## Testing

### 4.2.1 Test Cases for Unit Testing

Table 4. 1 Unit Testing of User Registration

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Test Data | Expected Outcomes | Test Result |
| Registration | Any input field: null |  | Pass |
| Name: sijan  Email: sijan@gmail.com  Password: sijan  Choose file: img.jpg | Registration Successful | Pass |
| Name: bijay  Email: sijan@gmail.com  Password: bijay  Choose file: img.jpg | Registration Unsuccessful | Pass |

Table 4. 2 Testing of User Log in

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Test Data | Expected Outcomes | Test Result |
| Login | Email: null  Password: null | Login unsuccessful | Pass |
| Email: sijan@gmail.com  Password: sijan | Login successful | Pass |

Table 4. 3 Testing for Generation of Shared Secret Key

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Test Data | Expected Outcomes | Test Result |
| Sender’s End |  |  | Pass |
| Receiver’s End |  |  |

Table 4. 4 Testing for Encryption and Decryption of Message

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Test Data | Expected Outcomes | Test Result |
| Sender’s End |  |  | Pass |
| Receiver’s End |  |  |

Table 4. 5 Testing for Encryption and Decryption of Message

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Test Data | Expected Outcomes | Test Result |
| Sender’s End |  |  | Pass |
| Receiver’s End |  |  |

### 4.2.2 Test Case for System Testing

**Scenario 1: User Log In**

* When the user tries to access the Chat Application, the user will be directed to the Login Page.
* In the Login Page, the user inputs his/her registered email and password.
* If the user inputs right credentials, he/she will be directed to Dashboard Page.
* Otherwise, an error message will pop-up conveying to re-enter the correct details for logging in to the chat application.

**Scenario 2: Chatting**

* When the user is redirected to Dashboard after successfully logging in the chat app, he/she can select the user to chat with from the user list.
* Now user can send and receive message from or to available users.

# CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATION

## Lesson Learnt

Through the help of this project, we get to know about HTML, CSS, JS, Node.js, MongoDB.

## Conclusion

On overall, the project has achieved its objectives. The chat application provides a better, flexible and secure system for chatting. It is developed with recent advanced technologies in a way to provide a reliable system. This application can find better need in the market for most of the organizations aim at having private applications for them. Chat application is one such social instant messaging application that brings you closer to your closer friends and family.

## Future Recommendations

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:

* File transfer: This will enable the user to send files of different formats to other via the chat application.
* Voice chat: This will enhance the application to a higher level where communication will be possible via calling as in telephone
* Video chat: This will further enhance the feature of calling into video communication