**A**

**PROJECT REPORT**

**ON**

# "WEBCHAT"

**SUBMITTED BY:**

MR. BHOSALE DARSHAN BHIMRAO (5054)

Under The Guidance of

Prof. MILIND DESHKAR

**SUBMITTED TO**

## Savitribai Phule Pune University

**As a partial fulfilment for the award of degree of**

**MASTER OF COMPUTER APPLICATIONS**

At

**ASM'S Institute of Business Management and Research, MCA Institute, Chinchwad, Pune – 411019**

Session: 2022-2023

# ACKNOWLEDGEMENT

It is a pleasure to acknowledge the assistance of several people and institutions in this effort. This project has turned me into a debtor. First and foremost, I feel indebted to my guide **Prof. Milind Deshkar** and all the faculty members of Department of MCA, ASM's IBMR, MCA Institute Chinchwad for their valuable guidance, continuous support and advice and constant encouragement throughout my project work.

I would like to extend my gratitude to honourable **Dr. V. P. Pawar Sir, Director, ASM's IBMR, MCA Institute, Chinchwad** for being a constant source of inspiration.

Finally, I would like to extend my thanks to all those who have contributed, directly or indirectly, to make this project successful.

**MR. BHOSLAE DARSHAN BHIMRAO (5054)**

# INDEX

# CHAPTER 1

# INTRODUCTION

# Institute Profile

## ASM's IBMR Pune - Institute of Business Management and Research

**About ASM Group**

Established in 1983, the Audyogik Shikshan Mandal (ASM) Group of Institutes has completed 40 years in the field of education. Through this period, ASM has persistently provided the corporate world with well-prepared professionals. This history of excellence has been validated by the fact that ASM has consistently been ranked amongst the top B-Schools in India. The faculty of ASM comprises the best in industry and academia The faculty members for the course are a rich blend of academicians, industry practitioners teaching faculty and mentors from industry. ASM has excellent industry-institute interaction for more than 4 decades and tie-ups with leading organizations for cutting edge certifications for enhanced skills leading to employability. ASM boasts of a strong alumni base of 72000+ graduates in leading corporates, who are very closely associated with the institute for placements & industry interfacing.

Our objective is to offer the best of technological advancements to our students to make them the most sought after in the industry. Various tie-ups with Oracle, IBM etc. help us offer the best. The recent collaboration with IBM enables us to provide our students with certificate programs in Business analytics & Big Data from IBM to enable them to be ready for the most sought-after career in recent times.

**Vision**

To be a world center of learning that excels in Management and IT education, research, training and consultancy.

**Mission**

Our mission is to attain excellence in education to contribute to the socio-economic transitions in the nation at all levels by presenting unique pedagogical opportunities aimed at developing effective, committed and dedicated, socially responsible global managers & leaders who make valuable contributions to all levels of the corporate world & society.

## Extra-Curricular Activities

Opportunities for learning, growing and achieving exist everywhere at ASM's IBMR. Life at IBMR is a blend of academics, extracurricular and co-curricular activities. Strong student clubs and cell activities give students opportunity to pursue hobbies of their interest. In addition, activities such as cultural programs, student fests, festival, national celebrations and intercollege events make life at IBMR truly colourful, enriching and enjoyable.

A year-round celebration of life is what it is all about at the Audyogik Shikshan Mandal Group of Institutes. Every year over 40 events are held across the various campuses. In the preceding year an overwhelming 60 events were held, making that an average of more than one per week.

Our students actively participate in various academic, co-curricular, extra-curricular, industry based, cultural and sports related competitions organized not only on campus but also at inter-college level. The students have consistently been top performers at such events, which only goes on to highlight the accent that ASM as an institute lays on all round growth of an individual.

Sports, trekking, outdoor and fun activities not only act as a welcome diversion from the rigours of theoretical classroom sessions but also aims to unearth the latent talents and skills of our students which in turn help in the overall development of a student's personality.

**Website :-** https://www.asmibmr.edu.in/

**Address :-** ASM Group of Institutes, MIDC Block 'C', Chinchwad Pune, Maharashtra 411019

**Contact Number :-** +91 9422009207/09/10/12

**Social Media Link :-** Linkedin : https://www.linkedin.com/in/asmgroup/

**Instagram :-** https://www.instagram.com/asmgroupofinstitutes

**Email :-** enquiry.asmedu.org

## 1.2 Abstract

The Webchat Project is an innovative online communication platform that enables real-time, interactive conversations between users through a web-based interface. It serves as a versatile tool for businesses, organizations, and individuals to connect, collaborate, and provide support to their customers or communities.

Module:

1. User Login Module
2. Home Module
3. Group Chat Module
4. Chat Module
5. Profile Module
6. Admin Dashboard Module
7. Add user Module
8. Create Group Module
9. Post Module
10. Admin Chat Module
11. Setting Module

## 1.3 Existing System and Need for System

The existing system for a webchat project depending on the specific implementation. However, a basic existing system include the following components:

1. **Front-end Interface:** The existing system have a web-based user interface that allows users to access the chat functionality. This interface includes features such as sending and receiving messages, displaying user profiles, and basic chat functionality.
2. **Server-side Application:** The existing system have a server-side application responsible for handling incoming requests, managing user connections, and facilitating message exchange between users. This application is developed using ASP.Net C#.
3. **Database:** The existing system uses a database to store user information, chat messages, and other relevant data.

## 1.4 Scope of System

The scope of the Webchat Project can vary depending on the specific requirements and goals of the project. Here is a general outline of the key components that can be included within the scope:

1. **User Registration and Authentication:** Implement a user registration system that allows individuals to create accounts, log in, and manage their profiles. This includes features such as password authentication, email verification, and password recovery.
2. **User Roles and Permissions:** Define different user roles (e.g., administrator, moderator, regular user) with corresponding permissions. Administrators have control over the platform's settings and user management, while regular users have access to basic chat functionalities.
3. **Chat Interface:** Develop a user-friendly web-based chat interface that supports real-time messaging. Users should be able to view ongoing conversations, send and receive messages, and see the online/offline status of other users.
4. **Group Chats:** Enable users to create and participate in group chats. Users can create public or private groups, invite other users to join, and manage the group settings (e.g., adding or removing participants, changing group name or description).
5. **Private Conversations:** Allow users to initiate private one-on-one conversations with other users. These conversations should be secure and visible only to the participating individuals.

## 1.5 Operating Environment

### 1.5.1 Hardware Requirement

| Name of Component | Specification |
|---|---|
| Processor | Intel i3 |
| RAM | 4GB |
| SSD | 256 GB or Above |

### 1.5.2 Software Requirements

| Name of Component | Specification |
|---|---|
| Operating System | Windows 10 |
| Language/Front End | PHP, HTML, CSS |
| Back End | JQUERY |
| Database | MS SQL Server |
| Browser | Any of Mozilla, Chrome etc. |

## 1.6 Brief Description of Technology is Used

### 1.6.1 HTML

HTML stands for HyperText Markup Language. It is used to design web pages using a markup language. HTML is a combination of Hypertext and Markup language. Hypertext defines the link between web pages. A markup language is used to define the text document within the tag which defines the structure of web pages. This language is used to annotate (make notes for the computer) text so that a machine can understand it and manipulate text accordingly. Most markup languages (e.g. HTML) are human-readable. The language uses tags to define what manipulation has to be done on the text.

HTML is a markup language used by the browser to manipulate text, images, and other content, in order to display it in the required format. HTML was created by Tim Berners-Lee in 1991. The first-ever version of HTML was HTML 1.0, but the first standard version was HTML 2.0, published in 1995.

Elements and Tags: HTML uses predefined tags and elements which tell the browser how to properly display the content. Remember to include closing tags. If omitted, the browser applies the effect of the opening tag until the end of the page.

HTML page structure: The basic structure of an HTML page is laid out below. It contains the essential building-block elements (i.e. doctype declaration, HTML, head, title, and body elements) upon which all web pages are created.

**Features of HTML:**

- It is easy to learn and easy to use.
- It is platform-independent.
- Images, videos, and audio can be added to a web page.
- Hypertext can be added to the text.
- It is a markup language.

**Why learn HTML?**

- It is a simple markup language. Its implementation is easy.
- It is used to create a website.
- Helps in developing fundamentals about web programming.
- Boost professional career.

**Advantages:**

- HTML is used to build websites.
- It is supported by all browsers.
- It can be integrated with other languages like CSS, JavaScript, etc.

**Disadvantages:**

- HTML can only create static web pages. For dynamic web pages, other languages have to be used.
- A large amount of code has to be written to create a simple web page.
- The security feature is not good.

### 1.6.2 CSS

Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independently of the HTML that makes up each web page. It describes how a webpage should look: it prescribes colours, fonts, spacing, and much more. In short, you can make your website look however you want. CSS lets developers and designers define how it behaves, including how elements are positioned in the browser.

While HTML uses tags, CSS uses rulesets. CSS is easy to learn and understand, but it provides powerful control over the presentation of an HTML document.

**Why CSS?**

- CSS saves time: You can write CSS once and reuse the same sheet in multiple HTML pages.
- Easy Maintenance: To make a global change simply change the style, and all elements in all the webpages will be updated automatically.
- Search Engines: CSS is considered a clean coding technique, which means search engines won't have to struggle to "read" its content.
- Superior styles to HTML: CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Offline Browsing: CSS can store web applications locally with the help of an offline cache. Using this we can view offline websites.

### 1.6.3 PHP

The term PHP is an acronym for PHP: Hypertext Preprocessor. PHP is a server-side scripting language designed specifically for web development. It is open-source which means it is free to download and use. It is very simple to learn and use. The files have the extension ".php". Rasmus Lerdorf inspired the first version of PHP and participated in the later versions. It is an interpreted language and it does not require a compiler.

- PHP code is executed in the server.

- It can be integrated with many databases such as Oracle, Microsoft SQL Server, MySQL, PostgreSQL, Sybase, and Informix.
- It is powerful to hold a content management system like WordPress and can be used to control user access.
- It supports main protocols like HTTP Basic, HTTP Digest, IMAP, FTP, and others.
- Websites like www.facebook.com and www.yahoo.com are also built on PHP.
- One of the main reasons behind this is that PHP can be easily embedded in HTML files and HTML codes can also be written in a PHP file.
- The thing that differentiates PHP from the client-side language like HTML is, that PHP codes are executed on the server whereas HTML codes are directly rendered on the browser. PHP codes are first executed on the server and then the result is returned to the browser.
- The only information that the client or browser knows is the result returned after executing the PHP script on the server and not the actual PHP codes present in the PHP file. Also, PHP files can support other client-side scripting languages like CSS and JavaScript.

**Characteristics of PHP are as follows.**

- Simple and fast
- Efficient
- Secured
- Flexible
- Cross-platform, it works with major operating systems like Windows, Linux, and macOS.
- Open Source
- Powerful Library Support
- Database Connectivity

**Disadvantages of PHP:**

1. PHP is not secure as it is open source.
2. Not good to create desktop applications.
3. Not suitable for large Web Applications- Php code is hard to maintain since it is not very modular.
4. Modification Problem – PHP does not allow the change in the core behavior of the web applications.

### 1.6.4 JQUERY

jQuery is an open source JavaScript library that simplifies the interactions between an HTML/CSS document, or more precisely the Document Object Model (DOM), and JavaScript.

Elaborating the terms, jQuery simplifies HTML document traversing and manipulation, browser event handling, DOM animations, Ajax interactions, and cross-browser JavaScript development.

**Why jQuery?**

Some of the key points which support the answer for why to use jQuery:

- It is incredibly popular, which is to say it has a large community of users and a healthy amount of contributors who participate as developers and evangelists.
- It normalizes the differences between web browsers so that you don't have to.
- It is intentionally a lightweight footprint with a simple yet clever plugin architecture.
- Its repository of plugins is vast and has seen steady growth since jQuery's release.
- Its API is fully documented, including inline code examples, which in the world of JavaScript libraries is a luxury. Heck, any documentation at all was a luxury for years.
- It is friendly, which is to say it provides helpful ways to avoid conflicts with other JavaScript libraries.

**Advantages:**

- Wide range of plug-ins. jQuery allows developers to create plug-ins on top of the JavaScript library.
- Large development community
- It has a good and comprehensive documentation
- It is a lot more easy to use compared to standard javascript and other javascript libraries.
- JQuery lets users develop Ajax templates with ease, Ajax enables a sleeker interface where actions can be performed on pages without requiring the entire page to be reloaded.
- Being Light weight and a powerful chaining capabilities makes jQuery more strong.

**Disadvantages:**

- While JQuery has an impressive library in terms of quantity, depending on how much customization you require on your website, the functionality may be limited thus using raw javascript may be inevitable in some cases.
- The JQuery javascript file is required to run JQuery commands, while the size of this file is relatively small (25-100KB depending on the server), it is still a strain on the client computer and maybe your web server as well if you intend to host the JQuery script on your own web server.

.

### 1.6.5 MS SQL Server

Data is a collection of facts and figures and we have humungous data available to the users via the internet and other sources. To manipulate the data, Structured Query Language (SQL) in

short has been introduced years ago. There are different versions of SQL available in the market provided by different organizations. In this article, we shall see the version of SQL provided by Microsoft.

Introduction :

1. Microsoft SQL Server or MS SQL Server for short is the query language provided for data definition and manipulation.
2. SQL Server is a Relational Database Management Systems which was developed and marketed by the Microsoft company.
3. SQL and SQL servers are built as two layers where the SQL server is on the top for interacting with the relational databases.
4. MS SQL Server also has T-SQL or Transact-SQL and the main focus of T-SQL is to handle the transactions.
5. As it is a Microsoft's developed system, it worked only on Microsoft's environment until it was made available on Linux platforms in the year 2016.

SQL Server is composed of: Database engine, and Relational engine, and Storage engine. These are explained as following below.

1. Database Engine –

Database is a collection of various data items on which the user can perform any kind of manipulations.

1. The database engine has a relational engine on which a user can perform queries and it also comes with a storage engine which manages the data files, indexes and procedures.
2. The database engine also creates and executes objects like triggers, views, procedures etc.

2. Relational Engine –

Relations are the connections between the two different databases or within the same database. It is stored in the form of a row and column intersection named tables.

1. It manages query processing, memory management, buffer management, threads, and much more.
2. It has another layer named storage engine.

3. Storage Engine –

1. It looks upon the storage of data.
2. It is done using systems like disks and Storage Area Network or SAN.

# CHAPTER 2

# PROPOSED SYSTEM

## 2.1 Literature Survey

Literature surveys are typically conducted to review existing literature, research papers, and related work on a particular topic. In the case of a webchat application, you would want to explore studies, articles, and papers that discuss various aspects of web-based chat systems, including user experience, design considerations, security, scalability, and other relevant factors. Here is a sample literature survey for a webchat application:

| Sr.No | Title | Authors | Published | Summery |
|---|---|---|---|---|
| 1. | "Designing Web Chat Interfaces for Enhanced User Experience" | Smith, J., Johnson, A., Brown, L. | 2022, ACM Transactions on Computer-Human Interaction | This paper explores the design principles and guidelines for creating effective and user-friendly web chat interfaces. It discusses various interaction patterns, notification mechanisms, and user customization options that can enhance the overall user experience in web-based chat systems. |
| 2. | "Scalability Challenges in Web Chat Applications" | Lee, C., Wang, H., Chen, M. | 2019, IEEE Transactions on Parallel and Distributed Systems | The paper investigates the scalability issues faced by web chat applications, such as handling a large number of concurrent users, ensuring low latency, and managing system resources efficiently. It provides insights into architectural approaches, load balancing techniques, and distributed systems strategies to address these challenges. |

| | | | | |
|---|---|---|---|---|
| 3. | "Secure Web Chat: Privacy and Encryption Considerations" | Garcia, R., Patel, S., Nguyen, T. | 2021, Journal of Computer Security | This study focuses on the security aspects of web chat applications, discussing encryption techniques, authentication mechanisms, and privacy considerations. It reviews existing protocols and proposes a novel approach for ensuring end-to-end security in web-based chat systems. |
| 4. | "Natural Language Processing for Web Chatbots" | Johnson, M., Thompson, S., Davis, K. | 2020, International Conference on Natural Language Processing | The paper examines the integration of natural language processing (NLP) techniques in web chat applications. It discusses methods for sentiment analysis, intent recognition, and context-awareness in chatbot systems, enabling more intelligent and conversational interactions between users and the application. |
| 5. | "Real-time Collaboration in Web Chat Applications" | Roberts, E., Clark, P., Jackson, L. | 2018, Proceedings of the ACM Conference on Computer-Supported Cooperative Work | This research explores the collaborative aspects of web chat applications, focusing on real-time communication, multi-user interactions, and collaborative editing features. It presents different synchronization mechanisms and collaboration models that enable effective teamwork and coordination within web chat environments. |

Remember that this is just a sample literature survey, and you can expand it by searching for more relevant papers, adjusting the focus based on your specific requirements, and including recent studies published after my knowledge cutoff date in September 2021.

## 2.2 Feasibility Study

A feasibility study for a webchat project involves assessing the practicality and viability of implementing the project. Here are some key areas to consider during the feasibility study:

1. **Technical Feasibility:**
   - ➔ Evaluate the technical requirements for the webchat project, including the necessary infrastructure, software development tools, and technologies.
   - ➔ Assess the availability and compatibility of the required resources, such as servers, databases, APIs, and frameworks.
   - ➔ Determine if the project can be implemented within the given timeline and budget constraints.
   - ➔ Consider any potential technical challenges or risks that may arise during the development and deployment of the webchat system.
2. **Market Feasibility:**
   - ➔ Conduct market research to identify the target audience for the webchat application.
   - ➔ Analyze the demand for webchat systems in the market and identify potential competitors.
   - ➔ Assess the potential user base and revenue generation opportunities for the webchat system.
   - ➔ Determine if there is a need for the webchat application and if it will be able to gain traction in the market.
3. **Financial Feasibility:**
   - ➔ Estimate the project's budget, considering the costs associated with development, infrastructure, maintenance, and ongoing support.
   - ➔ Evaluate the potential return on investment (ROI) and revenue generation strategies for the webchat system.
   - ➔ Assess the financial viability of the project by analysing the projected revenue and comparing it with the estimated costs.
4. **Legal and Ethical Feasibility:**
   - ➔ Identify and comply with legal requirements, such as data protection laws, privacy regulations, and user consent policies.
   - ➔ Consider ethical considerations related to data collection, storage, and usage within the webchat system.

- ➔ Assess the potential risks and impact of the webchat system on user privacy and security.

5. **Operational Feasibility:**
   - ➔ Evaluate the operational aspects of the webchat system, including the required human resources, skills, and training.
   - ➔ Consider the scalability and performance requirements of the webchat system to handle user traffic and accommodate growth.
   - ➔ Assess the potential integration with existing systems, such as customer relationship management (CRM) or support ticketing systems.
6. **User Experience Feasibility:**
   - ➔ Analyze the user experience requirements and expectations for the webchat application.
   - ➔ Assess the usability and accessibility of the webchat system for a diverse range of users.
   - ➔ Consider user feedback and conduct usability testing to refine the webchat system's interface and functionality.

By conducting a comprehensive feasibility study, you can identify any potential challenges or obstacles early on and make informed decisions about the viability and implementation of the webchat project.

## 2.3 Objectives of Systems

1. User-friendly Interface: The webchat project offers an intuitive and visually appealing interface that ensures a seamless user experience. The interface allows users to engage in conversations easily and navigate through different chat threads effortlessly.
2. Real-time Communication: Users can participate in real-time conversations, exchanging messages instantly with other users connected to the platform. The webchat project employs advanced technologies to ensure low latency and quick message delivery, providing a fluid and responsive communication experience.
3. Multi-platform Accessibility: The webchat project is accessible through web browsers on various platforms, including desktop computers, laptops, tablets, and smartphones. This multi-platform accessibility ensures that users can connect and engage in conversations regardless of their preferred device.
4. Group Chats and Private Conversations: Users can create group chats to collaborate with multiple participants simultaneously. Additionally, they can initiate private conversations with specific individuals for more confidential or personalised discussions.
5. Rich Media Support: The webchat project supports the exchange of various types of media, including images, videos, documents, and links. This feature enhances the versatility of communication and allows users to share relevant content in the context of their conversations.

## 2.4 User Requirements

1. **Contact Management:** Users should be able to manage their contacts, add new contacts, and organize them into groups for easier communication.
2. **Messaging:** Users should be able to send and receive text messages to/from their contacts. Real-time messaging is essential for an interactive experience.
3. **Multimedia Messaging:** The ability to send multimedia messages such as photos, videos, stickers, emojis, and attachments enhances the communication experience.
4. **Notifications:** Users should receive notifications for new messages and other relevant activities to stay updated.
5. **Privacy and Security:** Privacy is crucial for messaging apps. Users expect their data to be protected, and they should have control over their privacy settings, including who can contact them and how their data is used.
6. **Online Status:** Users may want to see the online status of their contacts to know who is currently available for a chat.
7. **Message Status:** A delivery and read status indicator for messages helps users know if their messages have been successfully delivered and read by the recipients.
8. **Search Functionality:** An efficient search feature is necessary to find past conversations and contacts quickly.
9. **Emojis and Emoticons:** A collection of emojis and emoticons can enhance the messaging experience and make conversations more expressive.
10. **Accessibility:** The website should be accessible to people with disabilities, complying with accessibility standards.
11. **Cross-Platform Compatibility:** Users expect the messenger website to work smoothly on various devices and operating systems, such as desktops, laptops, tablets, and smartphones.
12. **Sync Across Devices:** Users might use the messenger website on multiple devices, so it should offer synchronization of messages and settings across platforms.
13. **Account Settings:** Users should have the ability to customize their profile, notification settings, and other preferences.
14. **Blocking and Reporting:** Users should be able to block and report other users for inappropriate behaviour or spam.
15. **Message Editing and Deletion:** Users might appreciate the option to edit or delete sent messages.
16. **Voice and Video Calls:** While not mandatory for a messenger website, many users find voice and video calling capabilities valuable for more interactive conversations.

It's important to note that user requirements may vary based on the specific target audience and purpose of the webchat. Conducting user research and gathering feedback from potential users will help in refining and tailoring the requirements to meet their needs effectively. Additionally, the landscape of web development and user expectations might have evolved since my last update, so staying up-to-date with current trends is essential.

# CHAPTER 3

# ANALYSIS & DESIGN

## 3.1 System Requirements

### 3.1.1 Functional Requirements

| Use Case Name | Id | Trigger | Precondition | Basic Path | Post Condition |
|---|---|---|---|---|---|
| Admin | UC-001 | Admin Initiates registration | Admin is logged into the system | 1. Admin navigates to the registration section | 1. The system registers the student with the provided personal and login details.<br>2. New user Create.<br>3. New group created.<br>4. Personal chat and group chat recorded.<br>5. New Admin and Department create. |
| User | UC-002 | User request profile | User is logged into the system. | 1. User accesses the dashboard section. | 1. The users profile is updated with the new Information.<br>2. There are three section<br>I. Online/ Offline section<br>II. See posted Section<br>III. Chat Person Message box section |

### 3.1.2 Non Functional Requirements

#### 3.1.2.1 Performance Requirements

Performance requirements for this application typically revolve around responsiveness, scalability, and efficiency. Here are some common performance requirements to consider:

1. **Responsiveness:** A webchat application should provide a responsive and real-time experience to users. Messages should be delivered and displayed quickly, ensuring minimal delay between sending and receiving messages. Users should not experience significant lag or latency when interacting with the chat interface.
2. **Scalability:** The webchat application should be designed to handle a growing number of concurrent users. It should be able to scale horizontally to accommodate increasing user loads without compromising performance. Scalability can be achieved through techniques like load balancing, caching, and efficient resource utilization.
3. **Throughput:** The application should be capable of handling a high volume of messages and user interactions without significant delays or performance degradation. It should support efficient data transfer and processing to ensure smooth message delivery and retrieval.
4. **Low Latency:** The application should minimize latency, which is the delay between a user's action and the application's response. This includes actions like sending a message, receiving a message, or loading chat history. Low latency contributes to a seamless and interactive user experience.
5. **Bandwidth Efficiency:** Webchat applications should optimize bandwidth usage to minimize data transfer and reduce the amount of network traffic. Efficient compression techniques, intelligent data synchronization, and minimal redundant data can help achieve bandwidth efficiency.
6. **Connection Handling:** The application should efficiently manage and handle concurrent connections from multiple users. It should be capable of maintaining stable connections and gracefully handling disconnections or network interruptions to ensure uninterrupted chat functionality.
7. **Security and Privacy:** Performance requirements should also consider the impact of security measures on the application's performance. Encryption, authentication, and data protection mechanisms should be implemented without significantly impacting the overall performance of the webchat application.
8. **User Interface Responsiveness:** The user interface should be smooth and responsive, allowing users to interact with the chat application without delays or lags. UI

responsiveness contributes to a positive user experience and enhances usability.

It's important to note that the specific performance requirements may vary based on the scale, complexity, and intended usage of the webchat application. Regular performance testing and monitoring should be conducted to identify bottlenecks, optimize performance, and ensure that the application meets the desired performance standards.

#### 3.1.2.2 Safety Requirements

Safety requirements for this application are crucial to ensure the security and protection of users' data and interactions. Here are some common safety requirements for a webchat application:

1. **Data Encryption:** All communication and data transmitted between the client and server should be encrypted using secure protocols such as SSL/TLS. Encryption ensures that sensitive information, including messages, user credentials, and personal data, remains confidential and protected from unauthorized access.
2. **User Authentication and Authorization:** The webchat application should have robust user authentication mechanisms to verify the identity of users and ensure that only authorized individuals can access the chat system. Strong password policies, multi-factor authentication (MFA), and secure session management are essential for protecting user accounts from unauthorized access.
3. **Privacy Protection:** The webchat application should respect user privacy and comply with applicable data protection regulations (e.g., GDPR). It should clearly communicate its privacy practices, obtain user consent for data processing, and provide users with control over their personal information. Additionally, sensitive user data should be stored securely and handled in accordance with privacy guidelines.
4. **Message Integrity:** The webchat application should ensure the integrity of messages exchanged between users. Implementing message integrity checks, such as using message digests or digital signatures, can detect any tampering or modification of messages during transmission.
5. **Anti-Spam and Anti-Abuse Measures:** The webchat application should incorporate mechanisms to prevent and mitigate spam, abusive behavior, and malicious activities. Implementing spam filters, content moderation, and user reporting features can help maintain a safe and respectful environment for users.
6. **Secure Error Handling:** Error messages and system responses should be designed carefully to avoid leaking sensitive information that could be exploited by attackers. Error handling should provide informative feedback to users without disclosing potentially

harmful details about the system's internals.

7. **Regular Security Updates:** The webchat application should be regularly updated with security patches and fixes to address known vulnerabilities. This includes keeping the underlying software frameworks, libraries, and dependencies up to date to prevent exploitation of security flaws.
8. **Secure Backend Infrastructure:** The server infrastructure hosting the webchat application should be adequately secured. This includes implementing firewall protection, intrusion detection systems (IDS), access controls, and regular security audits to identify and address any potential vulnerabilities.
9. **Compliance with Industry Standards:** Depending on the nature of the webchat application and the industries it serves, compliance with relevant industry standards and regulations (e.g., HIPAA for healthcare data) may be necessary to ensure the safety and security of user data.

Implementing these safety requirements helps safeguard the webchat application and its users from potential security threats, data breaches, and unauthorized access. It is crucial to follow best practices, conduct security assessments, and regularly update and monitor the application's security measures to stay ahead of emerging threats.

#### 3.1.2.3 Security Requirements

Security requirements are essential to ensure the protection of user data, maintain confidentiality, integrity, and availability of the webchat application. Here are some common security requirements for a webchat application:

1. **User Authentication:** The webchat application should enforce strong user authentication mechanisms to verify the identity of users. This can include secure password policies, multi-factor authentication (MFA), and the use of secure authentication protocols like OAuth or OpenID Connect.
2. **Access Control:** Implementing proper access controls ensures that users can only access the features and data that they are authorized to use. Role-based access control (RBAC), permissions management, and least privilege principles should be employed to restrict unauthorized access to sensitive functionalities and information.
3. **Secure Communication:** The communication channel between the client and server should be secure to protect data in transit. This involves using encryption protocols such as SSL/TLS to secure the connection and prevent eavesdropping, tampering, or data interception.

4. **Input Validation:** The webchat application should validate and sanitize user input to prevent common vulnerabilities such as cross-site scripting (XSS), SQL injection, or command injection attacks. Proper input validation and output encoding techniques should be implemented to mitigate these risks.
5. **Secure Storage:** User data, including chat messages, user profiles, and authentication information, should be securely stored. This involves encrypting sensitive data at rest, implementing secure password hashing algorithms, and adhering to best practices for protecting data integrity and confidentiality.
6. **Protection against Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF):** The webchat application should implement measures to prevent XSS and CSRF attacks. Input sanitization, output encoding, and enforcing anti-CSRF tokens can help mitigate these vulnerabilities and protect users from malicious attacks.
7. **Session Management:** Secure session management practices should be employed to ensure that user sessions are properly managed, protected against session hijacking or fixation attacks, and invalidated upon user logout or inactivity.
8. **Security Logging and Monitoring:** The webchat application should log and monitor security-related events and activities to detect and respond to potential security incidents. This includes monitoring for abnormal behaviour, suspicious access patterns, and unauthorized access attempts.
9. **Regular Security Assessments and Penetration Testing:** The webchat application should undergo regular security assessments and penetration testing to identify vulnerabilities, assess risks, and validate the effectiveness of security measures. This helps ensure that security controls are up to date and effective against evolving threats.
10. **Compliance with Regulatory Standards:** If the webchat application handles sensitive data or operates in specific industries, compliance with regulatory standards such as GDPR, HIPAA, or PCI-DSS may be required to protect user privacy and ensure data security.

By incorporating these security requirements into the design and development of the webchat application, it can mitigate potential security risks, protect user data, and maintain a secure environment for users to communicate and interact. It is important to consider security as an ongoing process, regularly updating and monitoring security measures to address emerging threats and vulnerabilities.

### 3.1.2.4 Software Quality Attributes

Software quality attributes, also known as non-functional requirements, define the desired characteristics and qualities of a webchat application beyond its functional requirements. Here are some important software quality attributes for a webchat application:

1. **Reliability:** The webchat application should be reliable, ensuring that it operates consistently and predictably. It should be available and accessible to users, with minimal downtime or disruptions. Reliability also includes the ability to recover gracefully from errors or failures.
2. **Performance:** Performance refers to the responsiveness, speed, and efficiency of the webchat application. It should handle a high volume of concurrent users, process messages and requests promptly, and provide a smooth and responsive user experience. Performance requirements include factors like response time, throughput, and resource utilization.
3. **Scalability:** A webchat application should be scalable to handle increasing user loads and growing data volumes. It should be able to scale horizontally by adding more servers or resources to accommodate a larger user base without sacrificing performance. Scalability is important to ensure that the application can handle peak usage periods and future growth.
4. **Usability:** Usability focuses on the ease of use and intuitiveness of the webchat application's interface. It should be user-friendly, with clear navigation, intuitive controls, and minimal learning curve. Usability also includes considerations for accessibility, ensuring that the application can be used by individuals with disabilities.
5. **Security:** Security requirements have already been discussed in a previous response. A webchat application should have robust security measures in place to protect user data, prevent unauthorized access, and mitigate potential security risks and vulnerabilities.
6. **Maintainability:** Maintainability refers to the ease with which the webchat application can be modified, extended, or repaired. It includes factors such as modular design, clean code structure, documentation, and the use of coding best practices. Maintainability ensures that the application can be efficiently updated, fixed, or enhanced over its lifecycle.
7. **Testability:** Testability is the ease with which the webchat application can be tested to ensure its correctness and reliability. The application should be designed with testability in mind, allowing for effective unit testing, integration testing, and system testing. This includes providing tools and frameworks for automated testing and creating testable code.
8. **Extensibility:** Extensibility refers to the ability to easily add new features or functionalities to the webchat application without significant modifications to the existing codebase. The application should be designed to support future enhancements and customization,

allowing for easy integration with third-party services or APIs.

9. **Internationalization and Localization:** The webchat application should be designed to support multiple languages, cultural norms, and regional requirements. It should be easily adaptable to different locales and have the ability to display text, time, and date formats based on user preferences.
10. **Compliance:** Depending on the industry or regulatory standards, the webchat application may need to comply with specific regulations such as GDPR, HIPAA, or PCI-DSS. Compliance requirements should be considered and implemented as necessary to meet the applicable standards.

#### 3.1.2.5 Business Rules

Business rules for this application define the operational and behavioural guidelines that govern its use and functionality. These rules are specific to the business or organization implementing the webchat application and are designed to align with their goals and requirements.

1. **User Registration:** Users must register an account with valid credentials to access the webchat application. The registration process may include providing a unique username, email address, and password. Certain business rules may be enforced, such as password complexity requirements or email verification.
2. **User Roles and Permissions:** Different user roles may be defined in the webchat application, such as administrators, moderators, or regular users. Each role may have specific permissions and access rights. For example, administrators may have the ability to manage user accounts and chat rooms, while regular users can only participate in conversations.
3. **Chat Room Creation:** Users may be allowed to create chat rooms or join existing ones. Business rules may define limitations on the number of chat rooms a user can create or participate in. Additionally, rules may specify restrictions on the naming conventions or content allowed within chat rooms to maintain a safe and respectful environment.
4. **Content Moderation:** Business rules may outline guidelines for content moderation within the webchat application. This can include automatic filtering of inappropriate language or the appointment of moderators who review and remove offensive or violating content. Clear guidelines and consequences for violating content rules should be established.
5. **Message Retention and Archiving:** The webchat application may define rules regarding the retention and archiving of chat messages. For compliance or legal purposes, certain

messages may need to be stored for a specific period of time. The rules can specify the duration of message retention and the ability to search and retrieve archived messages.

6. **User Conduct and Code of Ethics:** Business rules can include guidelines for user conduct within the webchat application. Users may be required to adhere to a code of ethics or community guidelines, prohibiting behaviors such as harassment, spamming, or sharing of inappropriate content. Violations of the code of conduct may result in warnings, temporary suspensions, or permanent bans.
7. **Data Privacy and Consent:** Business rules should address data privacy and compliance with applicable regulations. Rules may require obtaining user consent for data processing, outlining the types of data collected, and specifying how user information is handled, stored, and shared. This ensures that the webchat application maintains the privacy and confidentiality of user data.
8. **Business Metrics and Reporting:** The webchat application may track and report on various business metrics such as user engagement, chat room activity, or user satisfaction. Business rules can outline the specific metrics to be measured, the frequency of reporting, and the responsible parties for monitoring and analyzing the data.

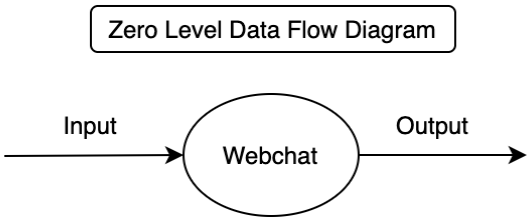
#### 3.1.2.6 Other Requirements

In addition to the previously mentioned requirements, here are some other important requirements for a this application:

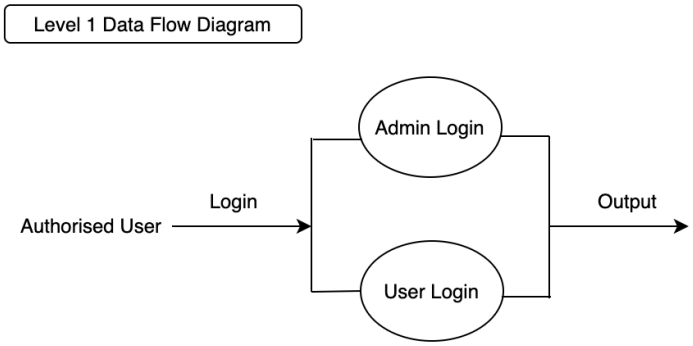
1. **Cross-Platform Compatibility:** The webchat application should be compatible with various platforms and devices, including desktops, laptops, smartphones, and tablets. It should support popular web browsers and operating systems to ensure a broad user reach.
2. **Cross-Browser Compatibility:** The webchat application should be compatible with a range of web browsers, including Chrome, Firefox, Safari, and Edge, ensuring consistent functionality and user experience across different browser platforms

## 3.2 Data Flow Diagram

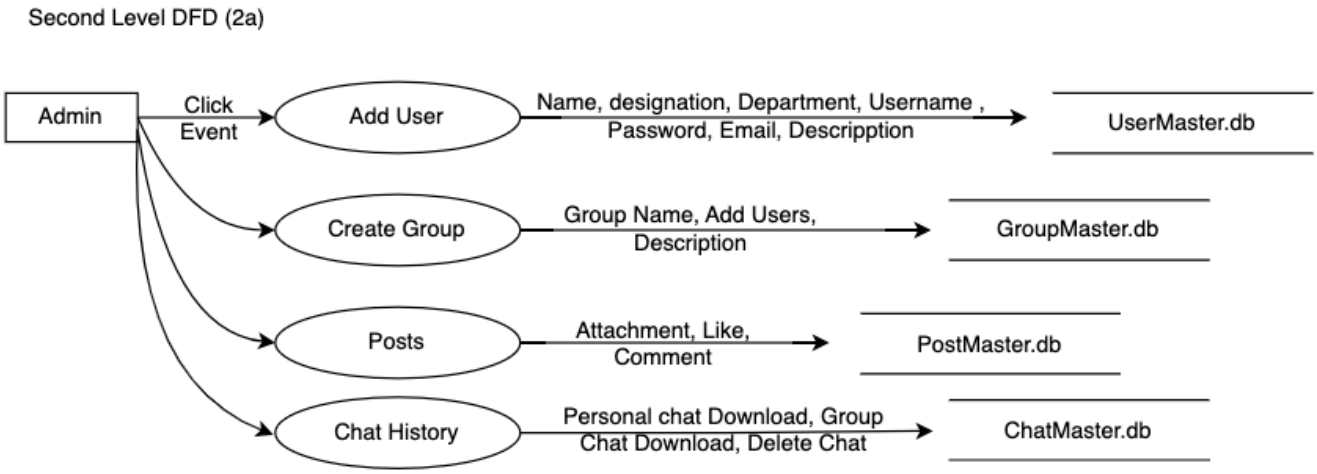
### 1. Zero Level Data Flow Diagram (DFD)




### 2. First Level Data Flow Diagram

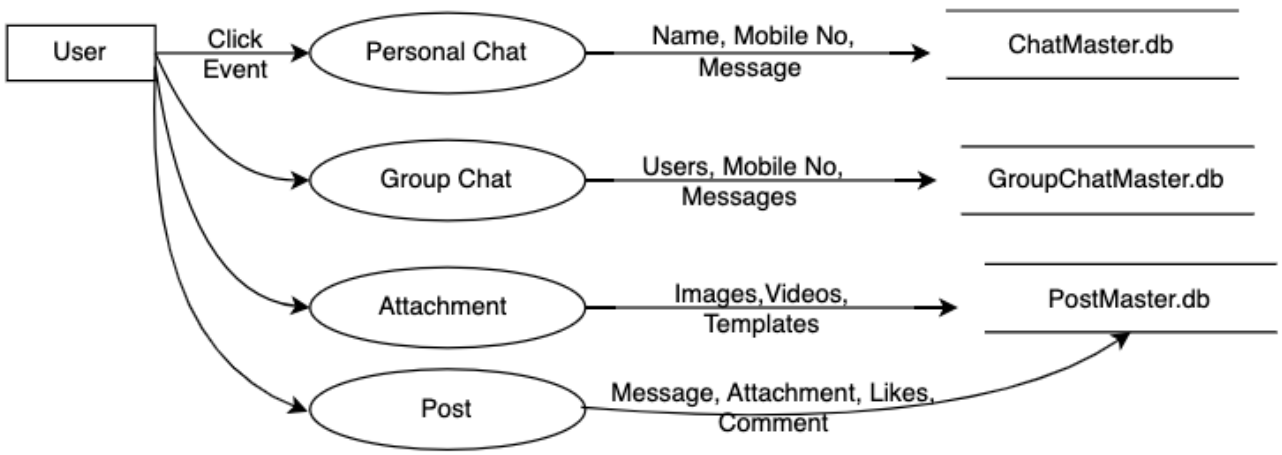



### 3. Second Level Data Flow Diagram (2a)

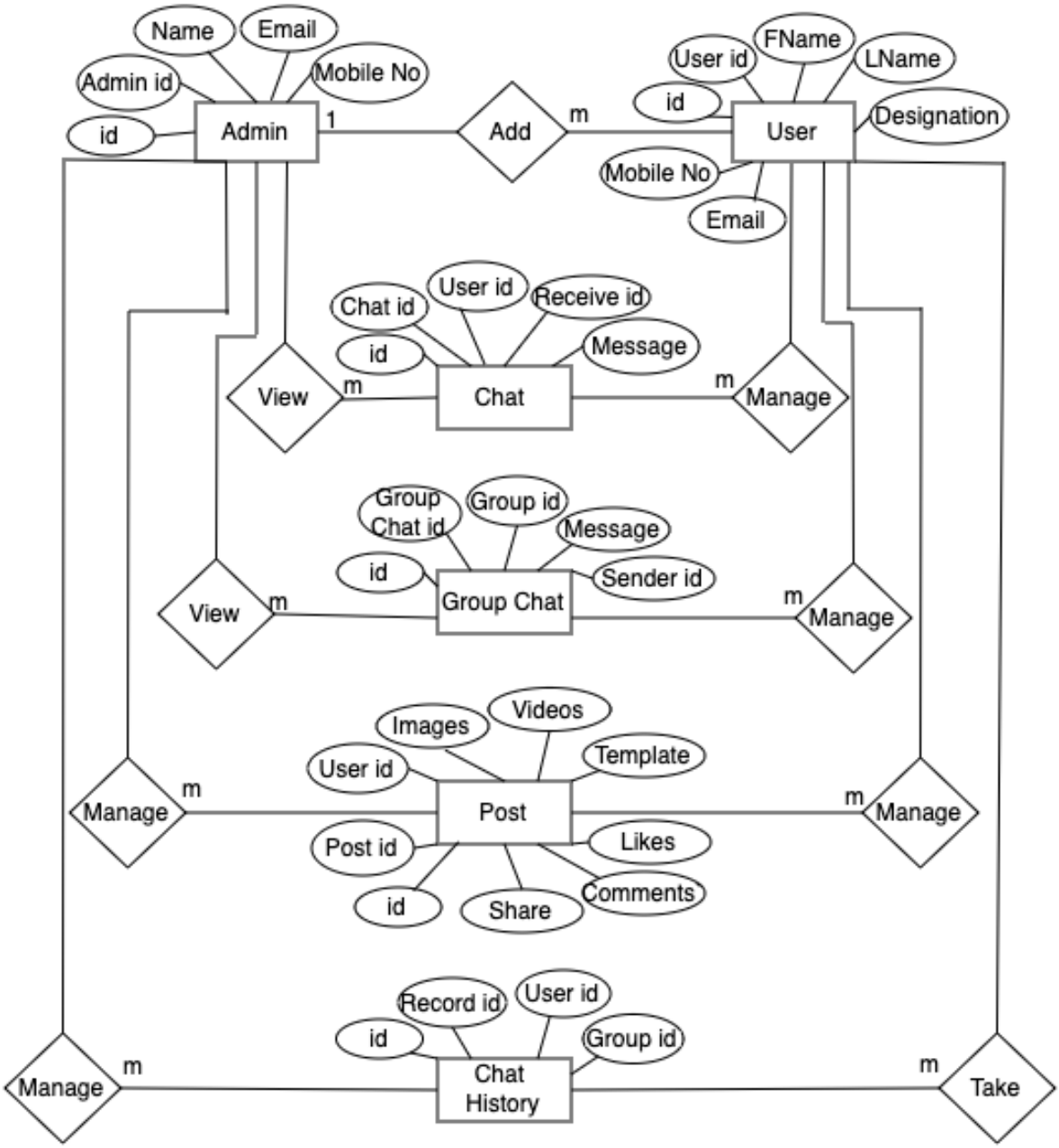


## 4. Second Level Data Flow Diagram (2b)

Second Level DFD (2b)



### 3.3 Entity Relation Diagram



# 3.4 Table Structure

## 3.4.1 User Master

| | Name | Own... | Type | Created_datetime |
|---|---|---|---|---|
| 1 | User_Master | dbo | user table | 2023-05-07 11:00:28.857 |

| | Column_name | Type | Computed | Length | Prec | Scale | Nullable | TrimTrailingBla... | FixedLenNullInSou... | Collation |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | User_Id | bigint | no | 8 | 19 | 0 | no | (n/a) | (n/a) | NULL |
| 2 | Dept_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 3 | First_Name | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 4 | Last_Name | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 5 | Type | varchar | no | 20 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 6 | Designation | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 7 | Username | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 8 | Password | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |

| | Identity | Se... | Increm... | Not For Replicat... |
|---|---|---|---|---|
| 1 | User_Id | 1 | 1 | 0 |

| | RowGuidCol |
|---|---|
| 1 | No rowguidcol column defined. |

| | Data_located_on_filegr... |
|---|---|
| 1 | PRIMARY |

| | index_name | index_description | index_ke... |
|---|---|---|---|
| 1 | PK_User_Master | clustered, unique, primary key located on PRIMARY | User_Id |

| | constraint_type | constraint_name | delete_acti... | update_act... | status_enab... | status_for_replica... | constraint_k... |
|---|---|---|---|---|---|---|---|
| 1 | PRIMARY KEY (clustered) | PK_User_Master | (n/a) | (n/a) | (n/a) | (n/a) | User_Id |

## 3.4.2 Post Master

| | Name | Own... | Type | Created_datetime |
|---|---|---|---|---|
| 1 | Post_Master | dbo | user table | 2023-05-07 10:59:39.550 |

| | Column_na... | Type | Comput... | Len... | Pr... | Sc... | Nulla... | TrimTrailingBla... | FixedLenNullInSou... | Collation |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Post_Id | bigint | no | 8 | 19 | 0 | no | (n/a) | (n/a) | NULL |
| 2 | Message | varchar | no | -1 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 3 | Added_By | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 4 | Added_Date | datetime | no | 8 | | | yes | (n/a) | (n/a) | NULL |
| 5 | Attachment | varchar | no | 200 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 6 | Likes | varchar | no | -1 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |

| | Identity | Se... | Increm... | Not For Replicat... |
|---|---|---|---|---|
| 1 | Post_Id | 1 | 1 | 0 |

| | RowGuidCol |
|---|---|
| 1 | No rowguidcol column defined. |

| | Data_located_on_filegr... |
|---|---|
| 1 | PRIMARY |

| | index_name | index_description | index_ke... |
|---|---|---|---|
| 1 | PK_Post_Master | clustered, unique, primary key located on PRIMARY | Post_Id |

| | constraint_type | constraint_name | delete_acti... | update_act... | status_enab... | status_for_replica... | constraint_k... |
|---|---|---|---|---|---|---|---|
| 1 | PRIMARY KEY (clustered) | PK_Post_Master | (n/a) | (n/a) | (n/a) | (n/a) | Post_Id |

### 3.4.3 Group Master

| | Name | Own... | Type | Created_datetime |
|---|---|---|---|---|
| 1 | Group_Master | dbo | user table | 2023-05-07 10:59:39.540 |

| | Column_name | Type | Comput... | Len... | Pr... | Sc... | Nulla... | TrimTrailingBla... | FixedLenNullInSou... | Collation |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Grp_Id | bigint | no | 8 | 19 | 0 | no | (n/a) | (n/a) | NULL |
| 2 | Name | varchar | no | 200 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 3 | User_Id | varchar | no | 200 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 4 | Description | varchar | no | -1 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 5 | Added_By | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 6 | Added_Date | datetime | no | 8 | | | yes | (n/a) | (n/a) | NULL |
| 7 | Modified_By | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 8 | Modified_Date | datetime | no | 8 | | | yes | (n/a) | (n/a) | NULL |

| | Identity | Se... | Increm... | Not For Replicat... |
|---|---|---|---|---|
| 1 | Grp_Id | 1 | 1 | 0 |

| | RowGuidCol |
|---|---|
| 1 | No rowguidcol column defined. |

| | Data_located_on_filegr... |
|---|---|
| 1 | PRIMARY |

| | index_name | index_description | index_ke... |
|---|---|---|---|
| 1 | PK_Group_Master | clustered, unique, primary key located on PRIMARY | Grp_Id |

| | constraint_type | constraint_name | delete_acti... | update_act... | status_enab... | status_for_replica... | constraint_k... |
|---|---|---|---|---|---|---|---|
| 1 | PRIMARY KEY (clustered) | PK_Group_Master | (n/a) | (n/a) | (n/a) | (n/a) | Grp_Id |

### 3.4.4 Chat Master User

| | Name | Own... | Type | Created_datetime |
|---|---|---|---|---|
| 1 | Chat_Master_User | dbo | user table | 2023-05-07 10:59:39.530 |

| | Column_name | Type | Comput... | Len... | Pr... | Sc... | Nulla... | TrimTrailingBla... | FixedLenNullInSou... | Collation |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Chat_User_Id | bigint | no | 8 | 19 | 0 | no | (n/a) | (n/a) | NULL |
| 2 | Attach_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 3 | Note_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 4 | User_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 5 | Receiver_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 6 | Message | varchar | no | -1 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 7 | Type | varchar | no | 100 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 8 | Added_By | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |

| | Identity | Se... | Increm... | Not For Replicat... |
|---|---|---|---|---|
| 1 | Chat_User_Id | 1 | 1 | 0 |

| | RowGuidCol |
|---|---|
| 1 | No rowguidcol column defined. |

| | Data_located_on_filegr... |
|---|---|
| 1 | PRIMARY |

| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | PK_Chat_Master_User | clustered, unique, primary key located on PRIMARY | Chat_User_Id |

| | constraint_type | constraint_name | delete_acti... | update_act... | status_enab... | status_for_replica... | constraint_k... |
|---|---|---|---|---|---|---|---|
| 1 | PRIMARY KEY (clustered) | PK_Chat_Master_User | (n/a) | (n/a) | (n/a) | (n/a) | Chat_User_Id |

### 3.4.5 Chat Master Group

| | Name | Own... | Type | Created_datetime |
|---|---|---|---|---|
| 1 | Chat_Master_Group | dbo | user table | 2023-05-07 10:59:39.523 |

| | Column_name | Type | Comput... | Len... | Pr... | Sc... | Nulla... | TrimTrailingBla... | FixedLenNullInSou... | Collation |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Chat_Group_Id | bigint | no | 8 | 19 | 0 | no | (n/a) | (n/a) | NULL |
| 2 | Attach_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 3 | Note_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 4 | Grp_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 5 | Sender_Id | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 6 | Message | varchar | no | -1 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 7 | Type | varchar | no | 100 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |
| 8 | Added_By | varchar | no | 120 | | | yes | no | yes | SQL_Latin1_General_CP1_CI_AS |

| | Identity | Se... | Increm... | Not For Replicat... |
|---|---|---|---|---|
| 1 | Chat_Group_Id | 1 | 1 | 0 |

| | RowGuidCol |
|---|---|
| 1 | No rowguidcol column defined. |

| | Data_located_on_filegr... |
|---|---|
| 1 | PRIMARY |

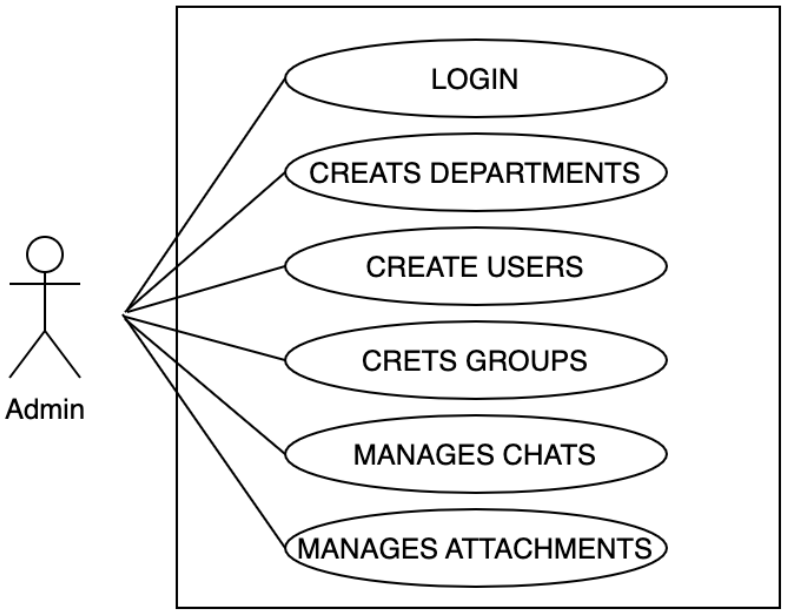
| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | PK_Chat_Master_Group | clustered, unique, primary key located on PRIMARY | Chat_Group_Id |

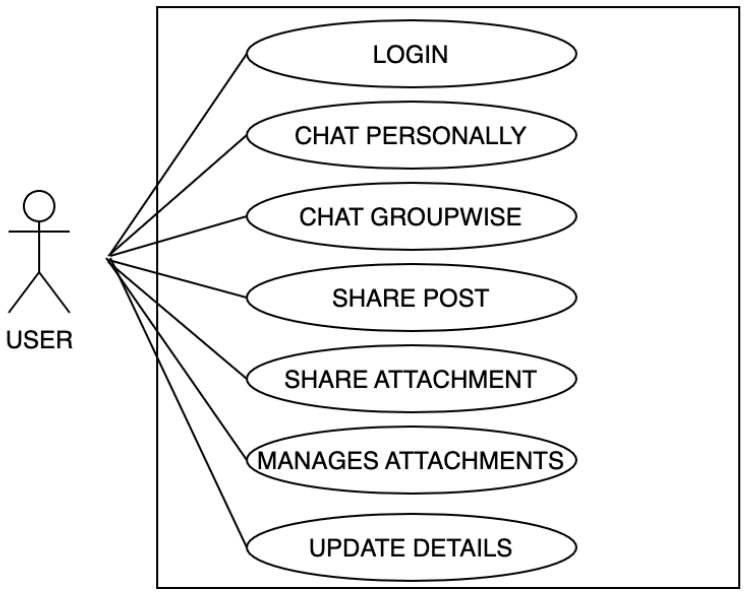
| | constraint_type | constraint_name | delete_acti... | update_act... | status_enab... | status_for_replica... | constraint_keys |
|---|---|---|---|---|---|---|---|
| 1 | PRIMARY KEY (clustered) | PK_Chat_Master_Group | (n/a) | (n/a) | (n/a) | (n/a) | Chat_Group_Id |

## 3.5 Use Case Diagram

### 3.5.1 Use Case Diagram (Admin)

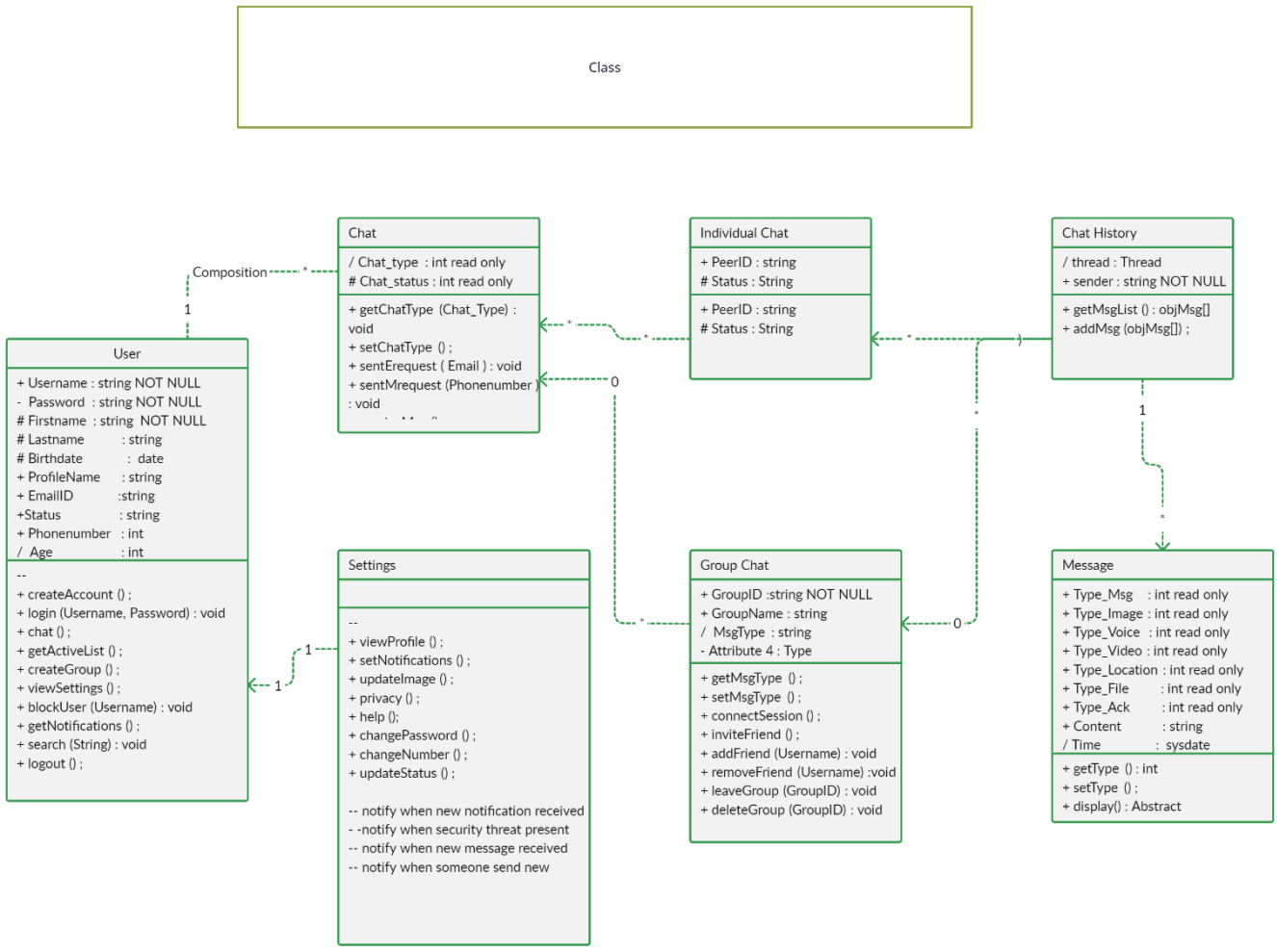
The use case diagram provides a visual representation of the functional requirements of a system and the interactions between its actors (users, external systems, or entities) and the system itself. It is a useful tool for capturing high-level system behaviour and defining the scope of the software requirements.



## 3.6 Class Diagram

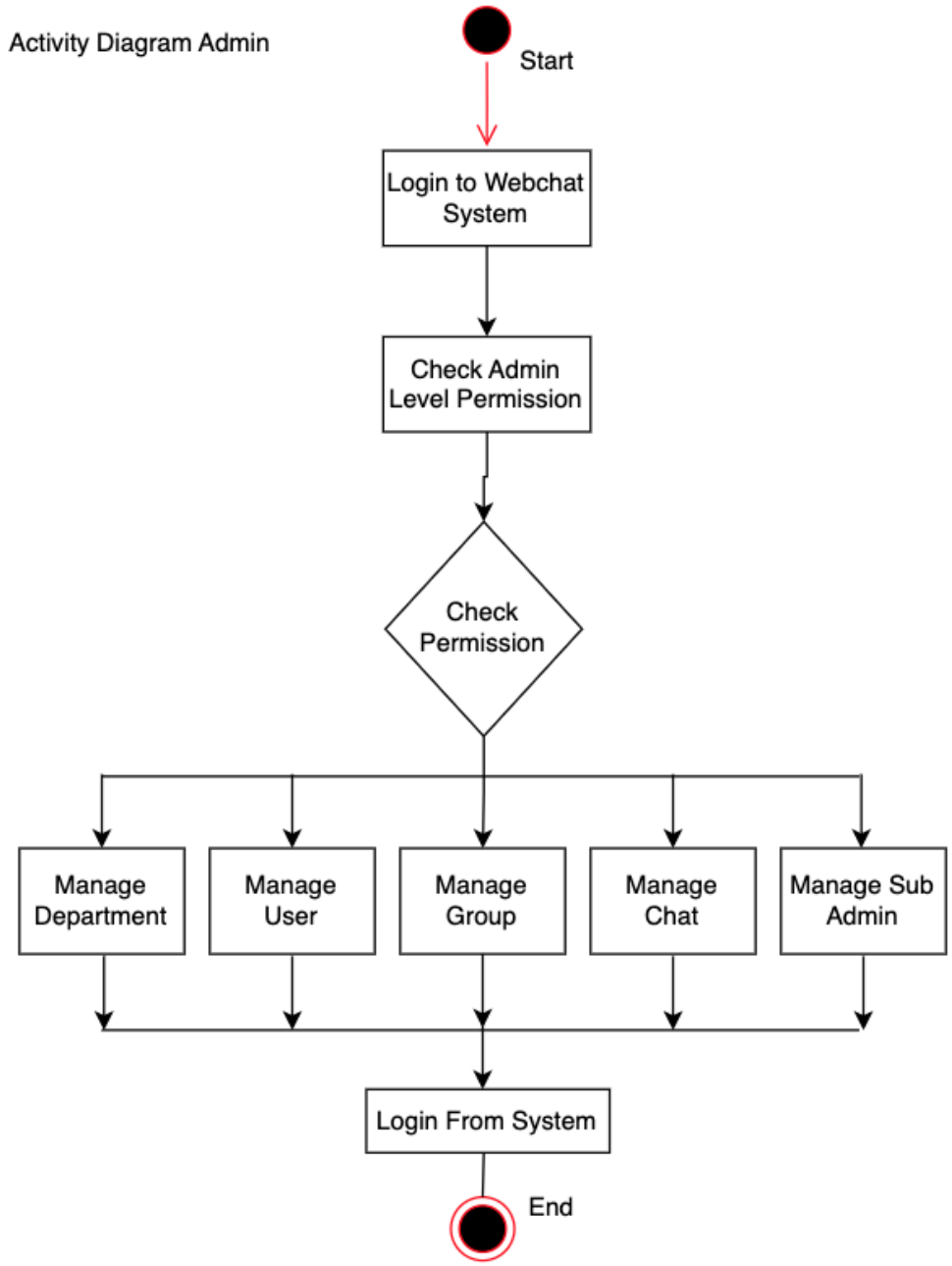
The class diagram represents the static structure of the Webchat application, depicting the classes, their attributes, and relationships between them. It shows the entities and their properties, methods, and associations, providing a blueprint for the system's object-oriented design.



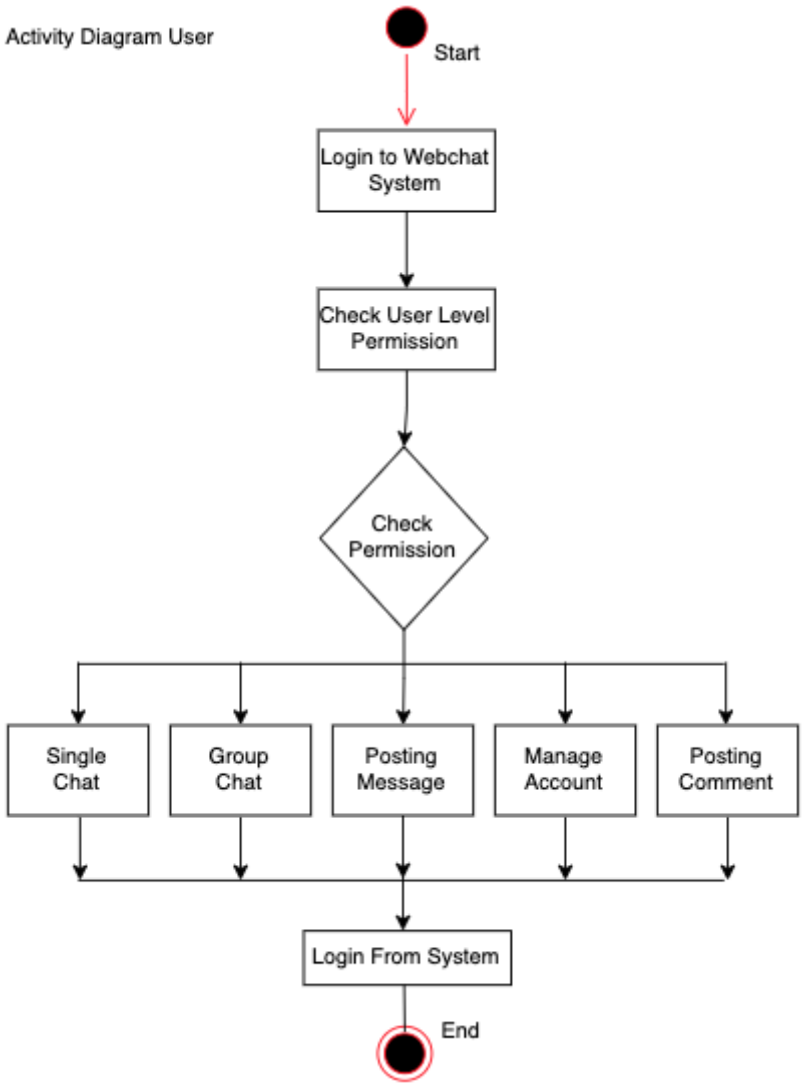
### 3.7 Activity Diagram (Admin & User)

1. Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.
2. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.
3. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all types of flow control by using different elements such as fork, join, etc.

#### 3.7.1 Activity Diagram (Admin)

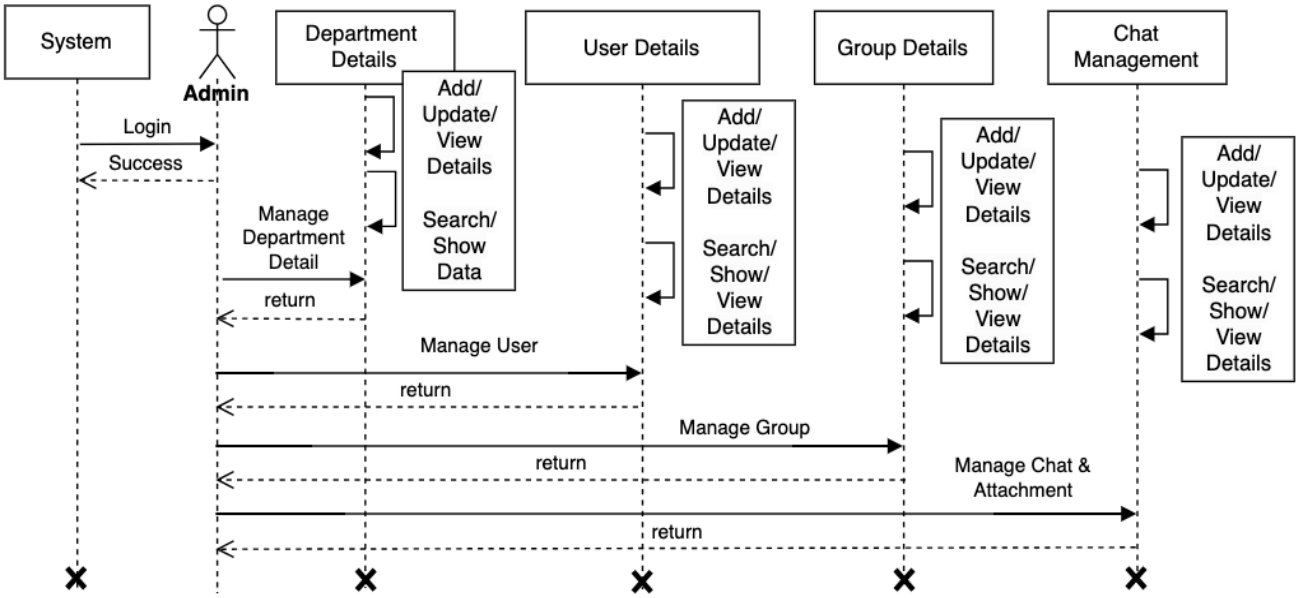


### 3.7.2 Activity Diagram (User)



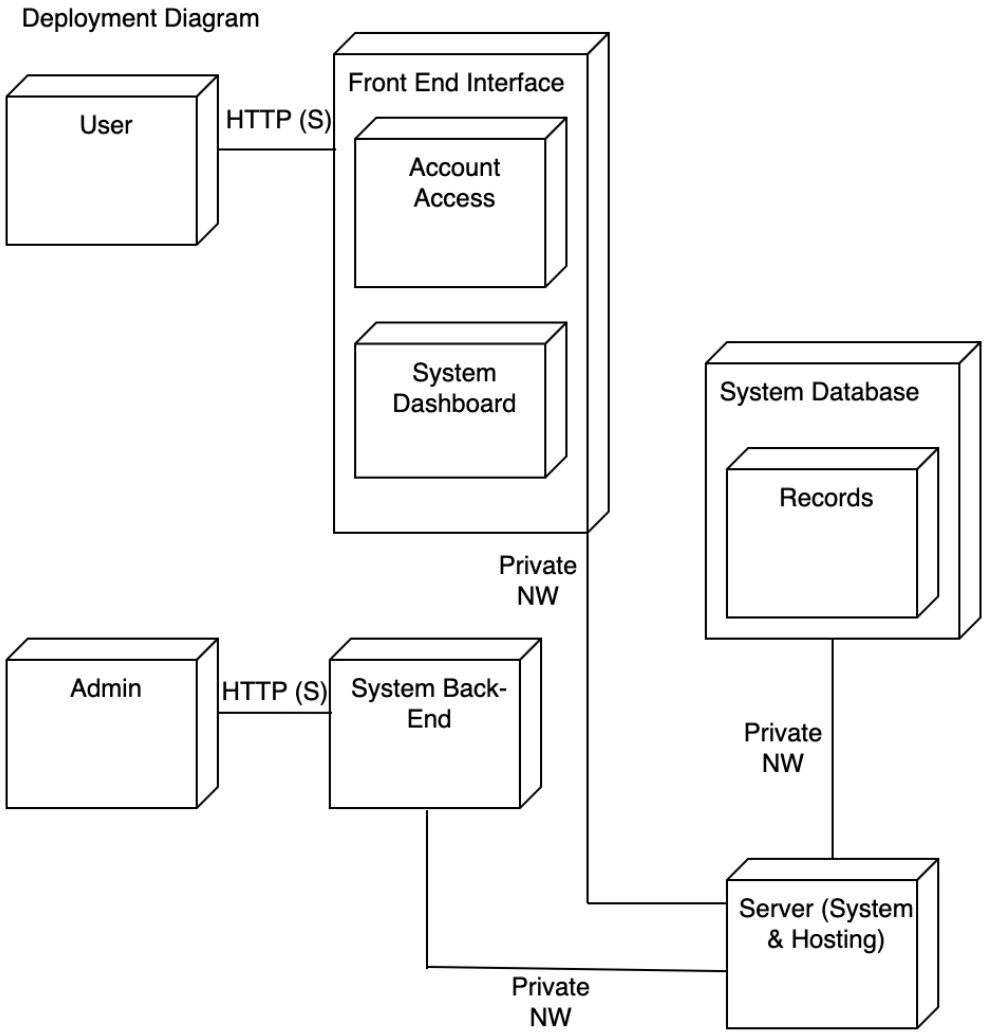
### 3.8 Sequence Diagram (Admin & User)

A sequence diagram or system sequence diagram shows process interactions arranged in time sequence in the field of software engineering. It depicts the processes and objects involved and the sequence of messages exchanged between the processes and objects needed to carry out the functionality.

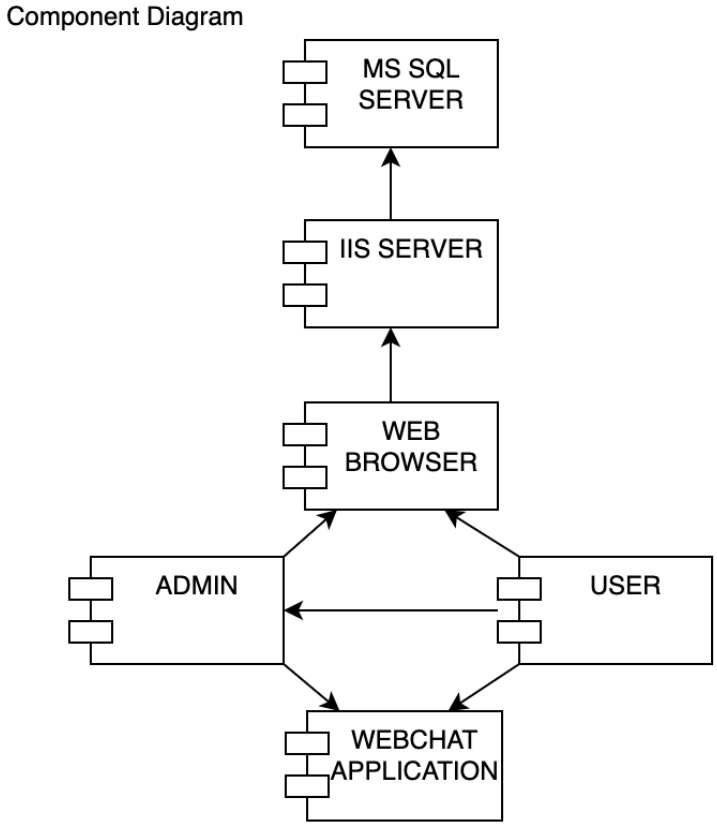


### 3.9 Deployment Diagram

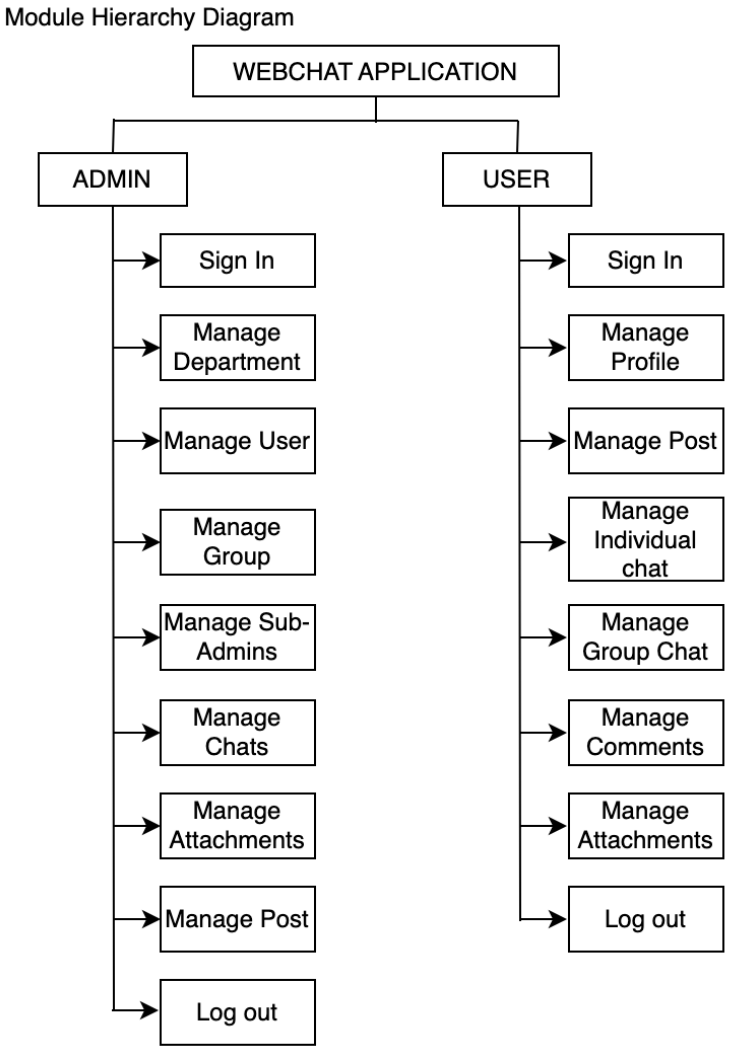
A deployment diagram in software engineering represents the physical deployment of software components and their relationships within a system. It illustrates how software artifacts are deployed on hardware infrastructure, such as servers, devices, or virtual machines.



## 3.10 Component Diagram

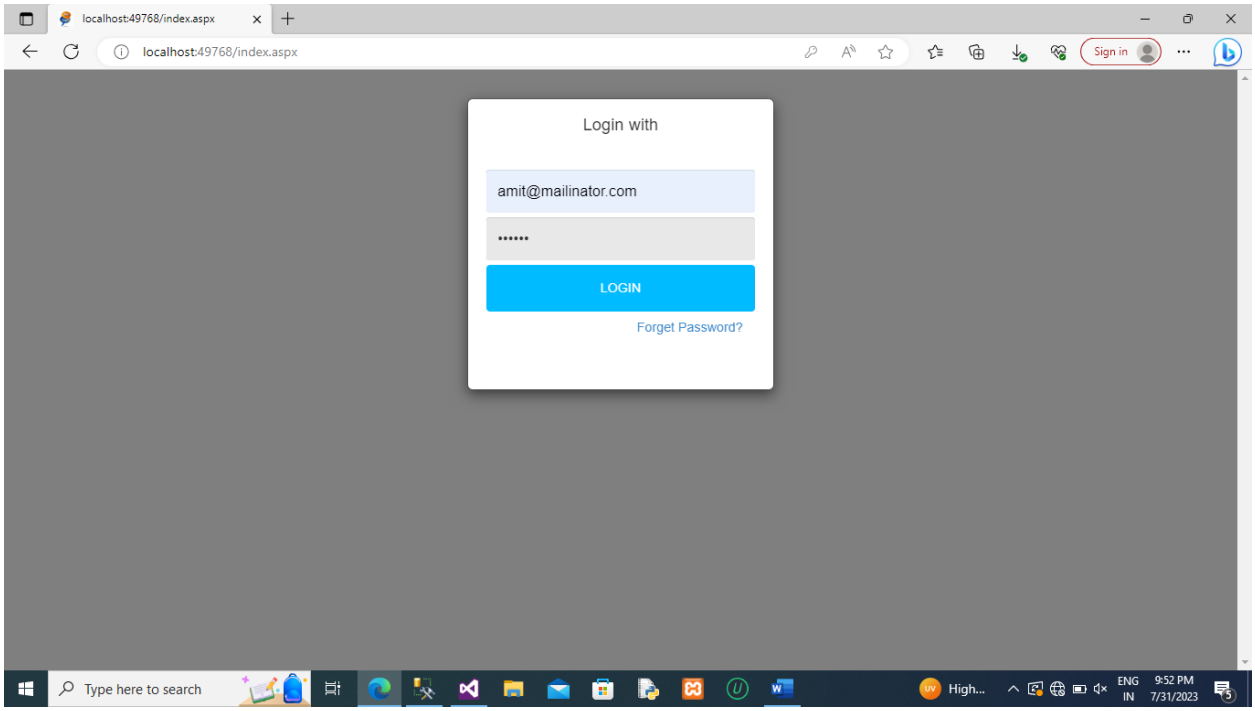


## 3.11 Module Hierarchy Diagram

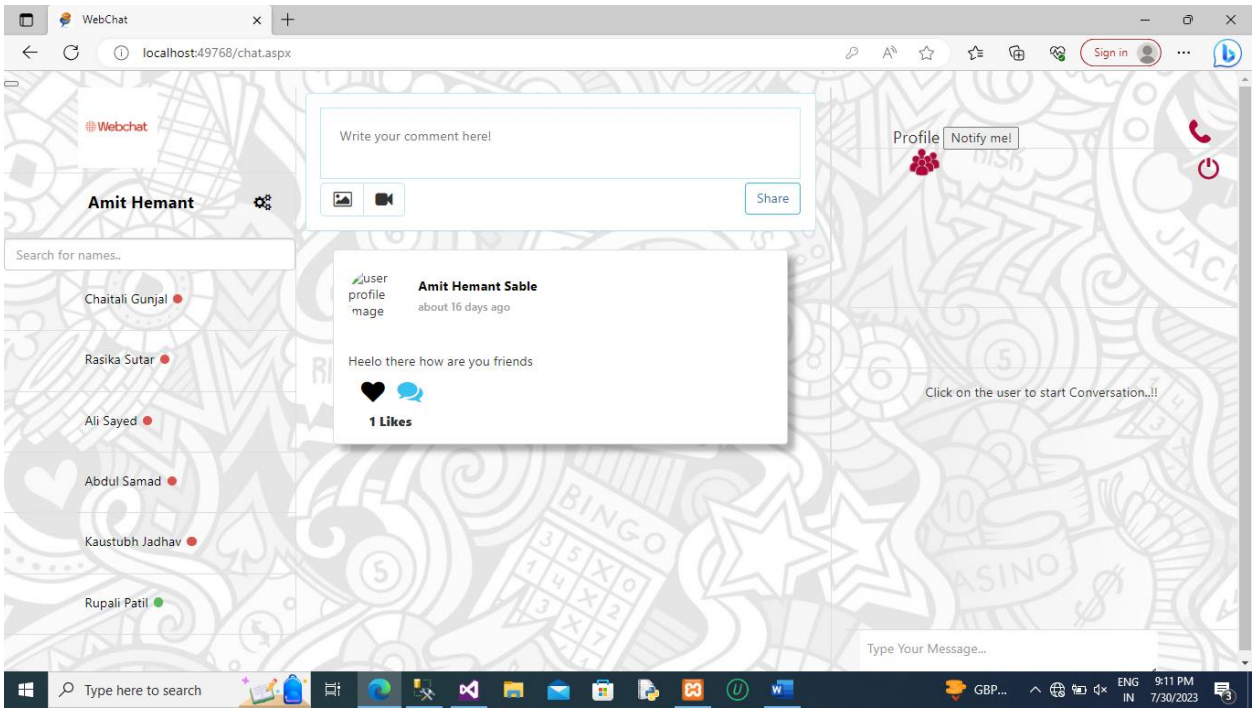


## 3.12 User Interface Design

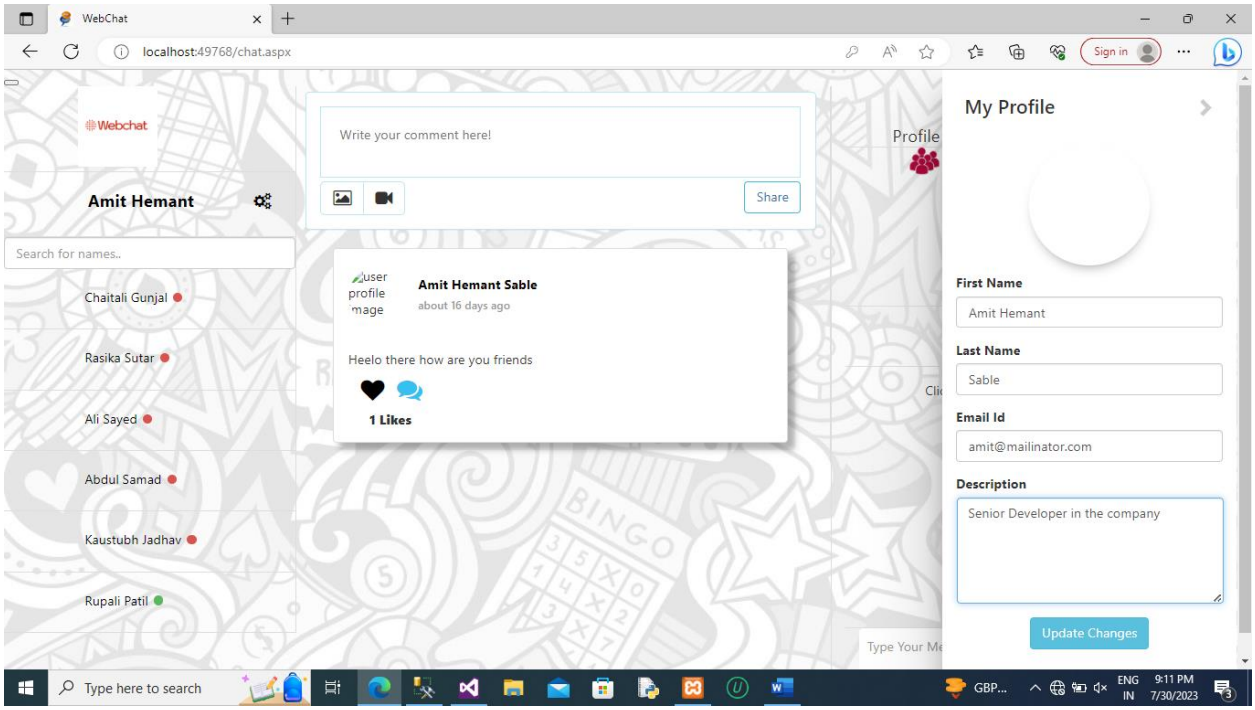
### 3.12.1 User Login Panel



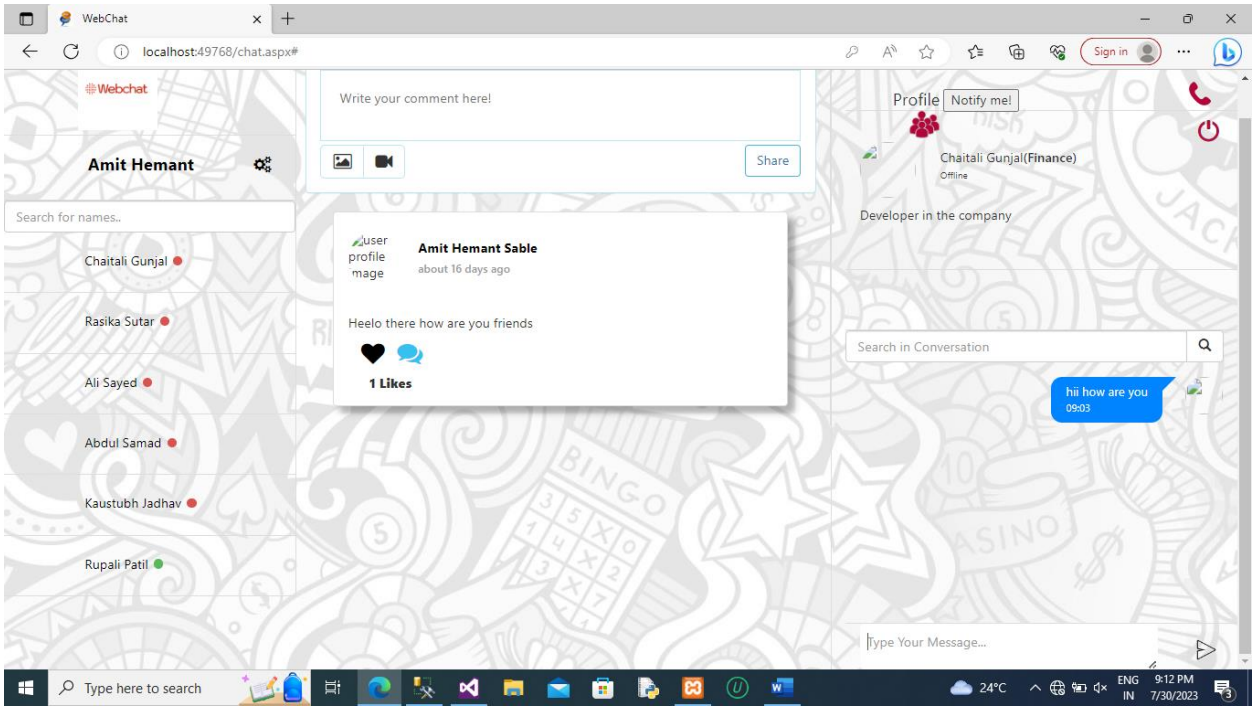

### 3.12.2 User Dashboard



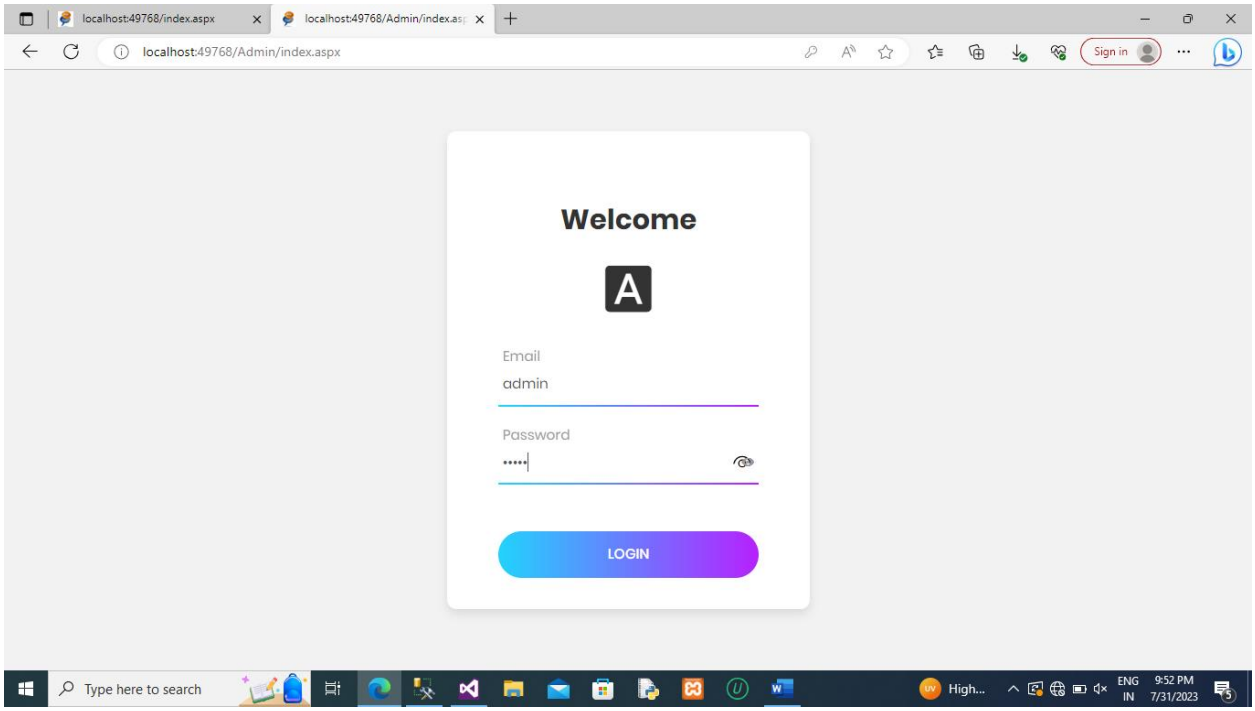
## 3.12.3 User Profile Update



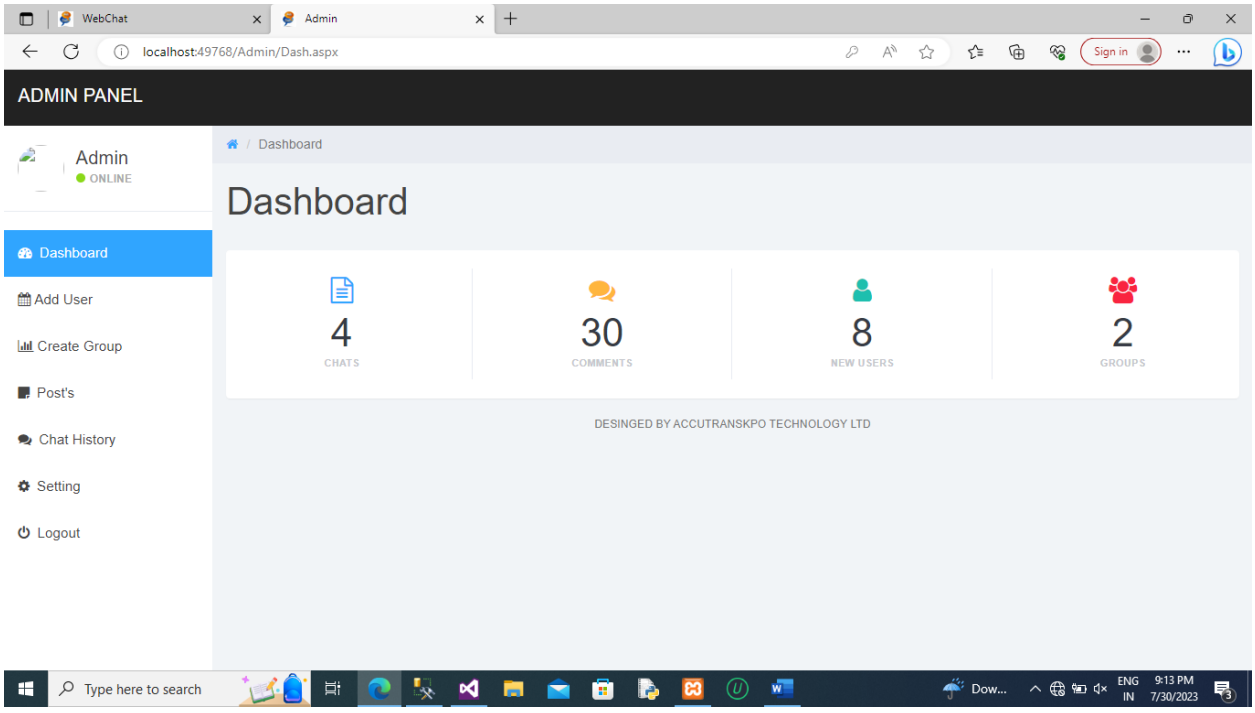

## 3.12.4 User Chat Box



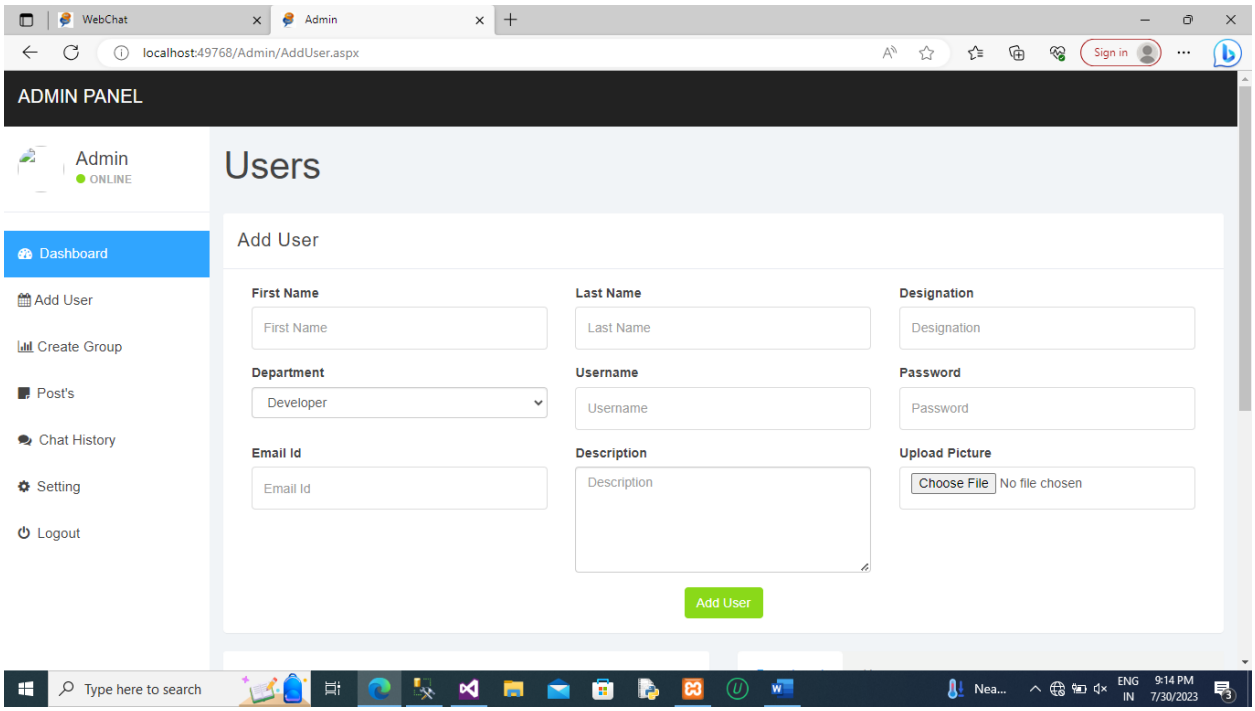
### 3.12.5 Admin Login Panel



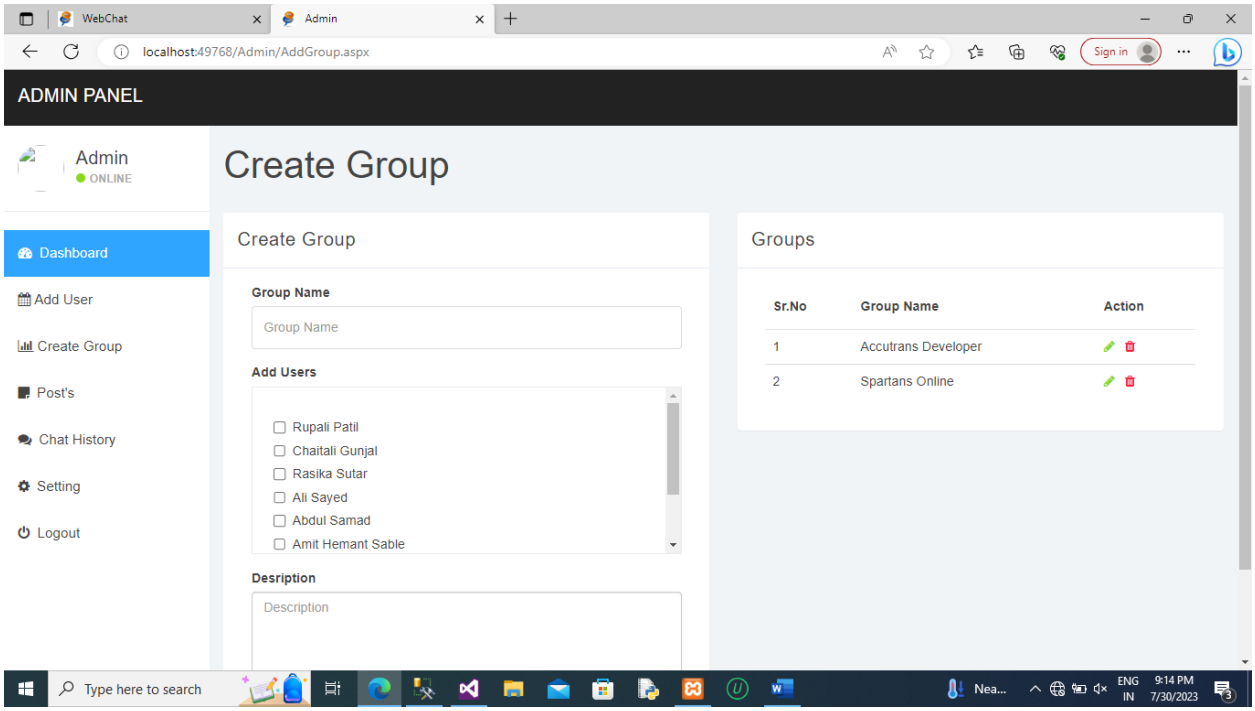

### 3.12.6 Admin Dashboard



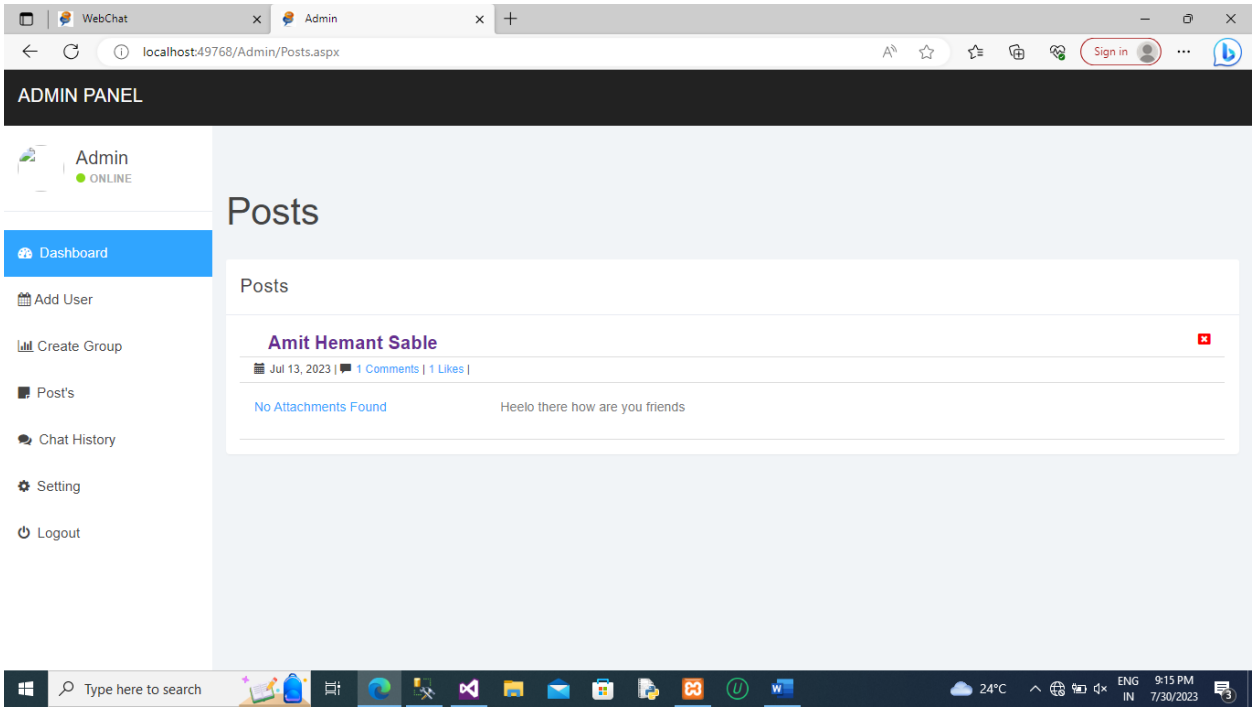
## 3.12.7 Add User



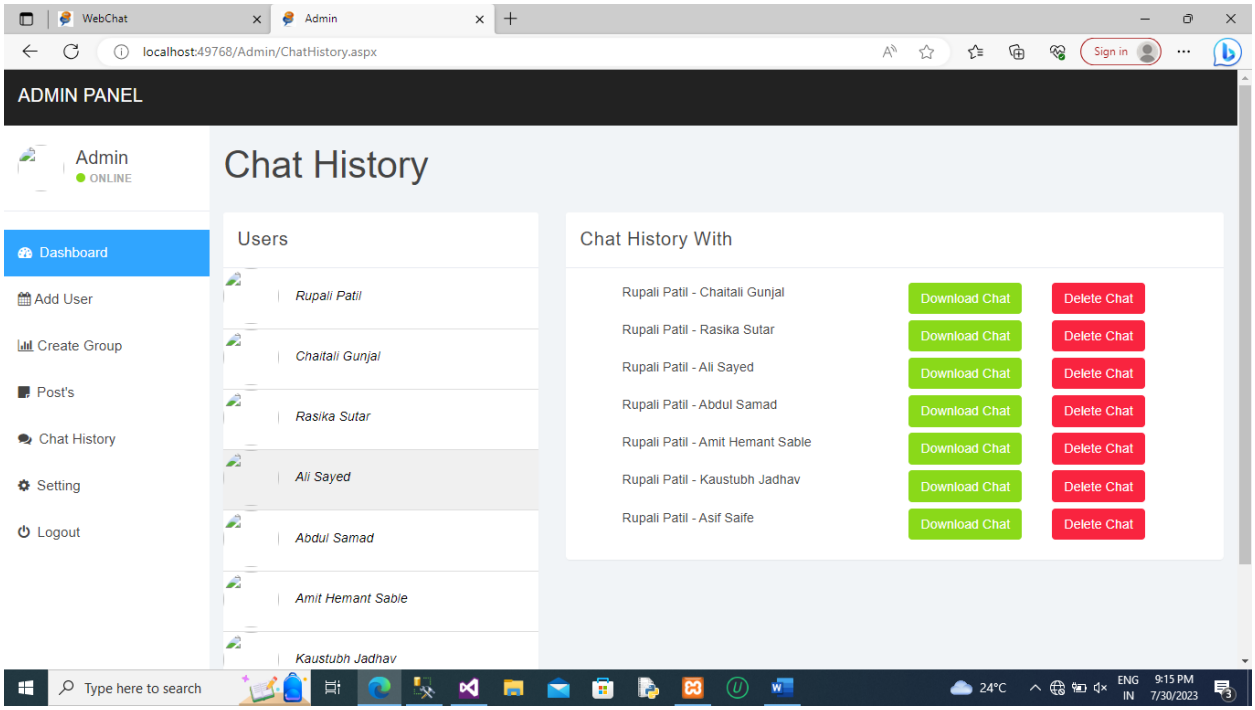

## 3.12.8 Create Group



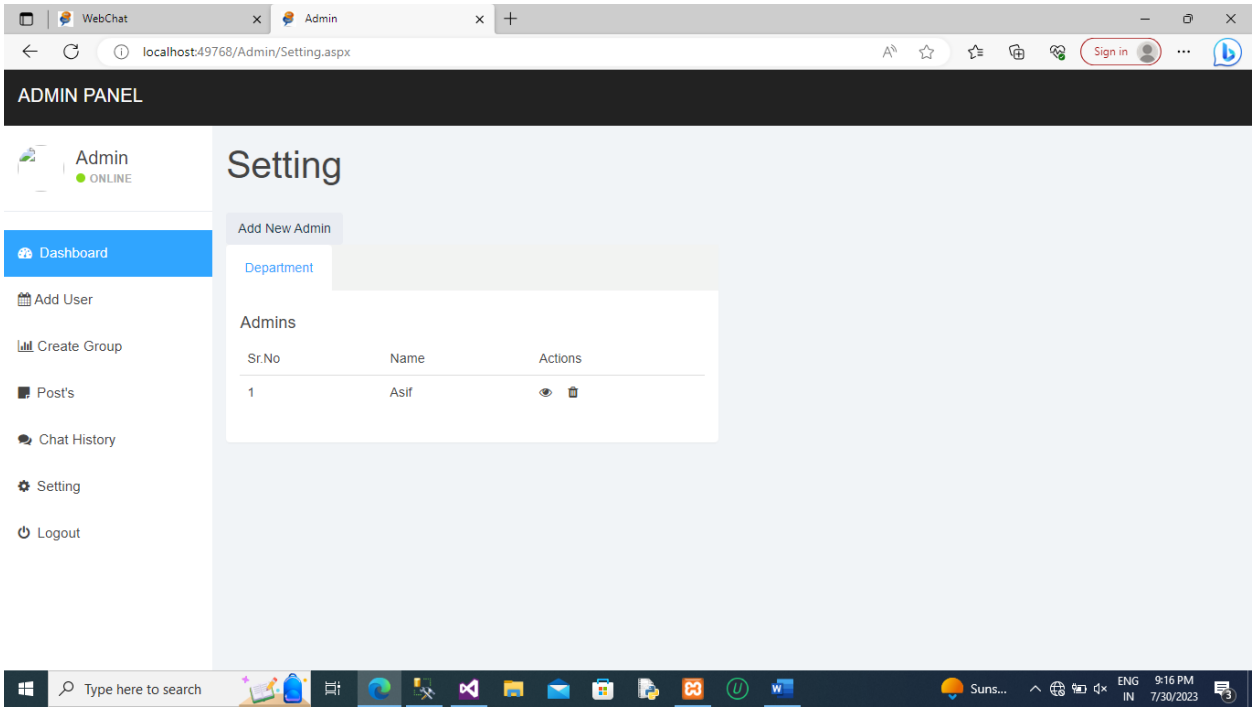
### 3.12.9 Posts



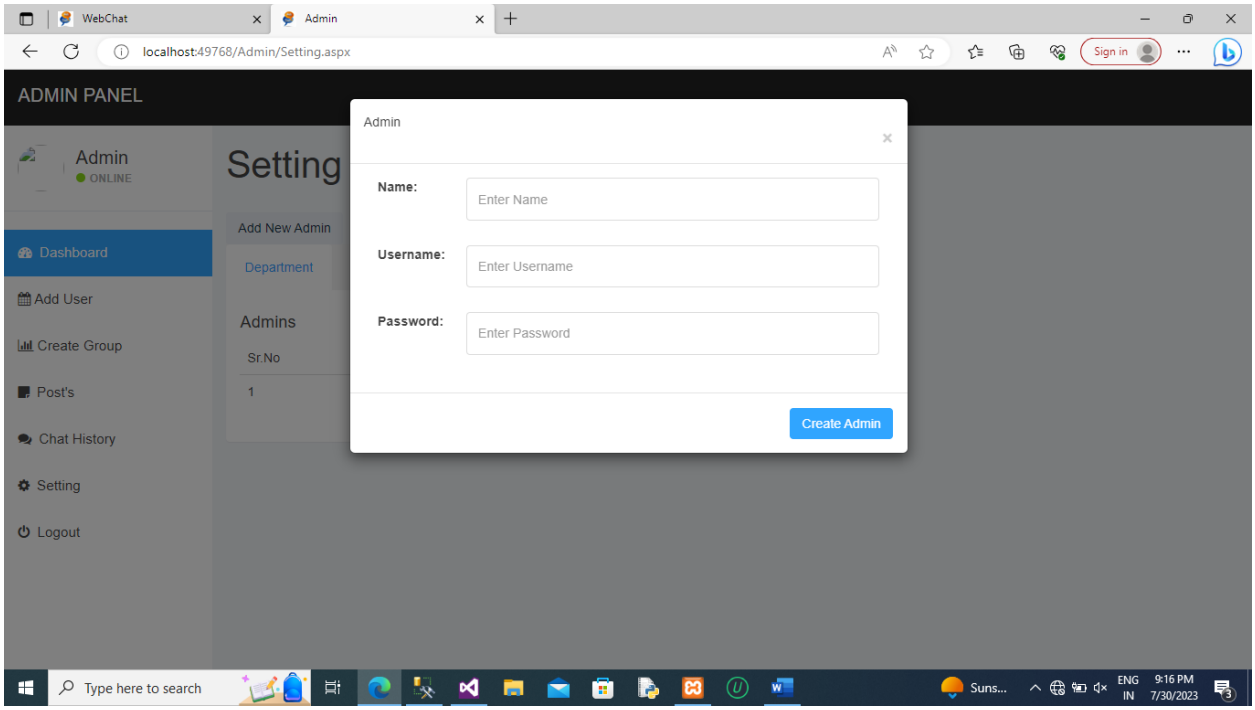

### 3.12.10 Chat History



### 3.12.11 Setting

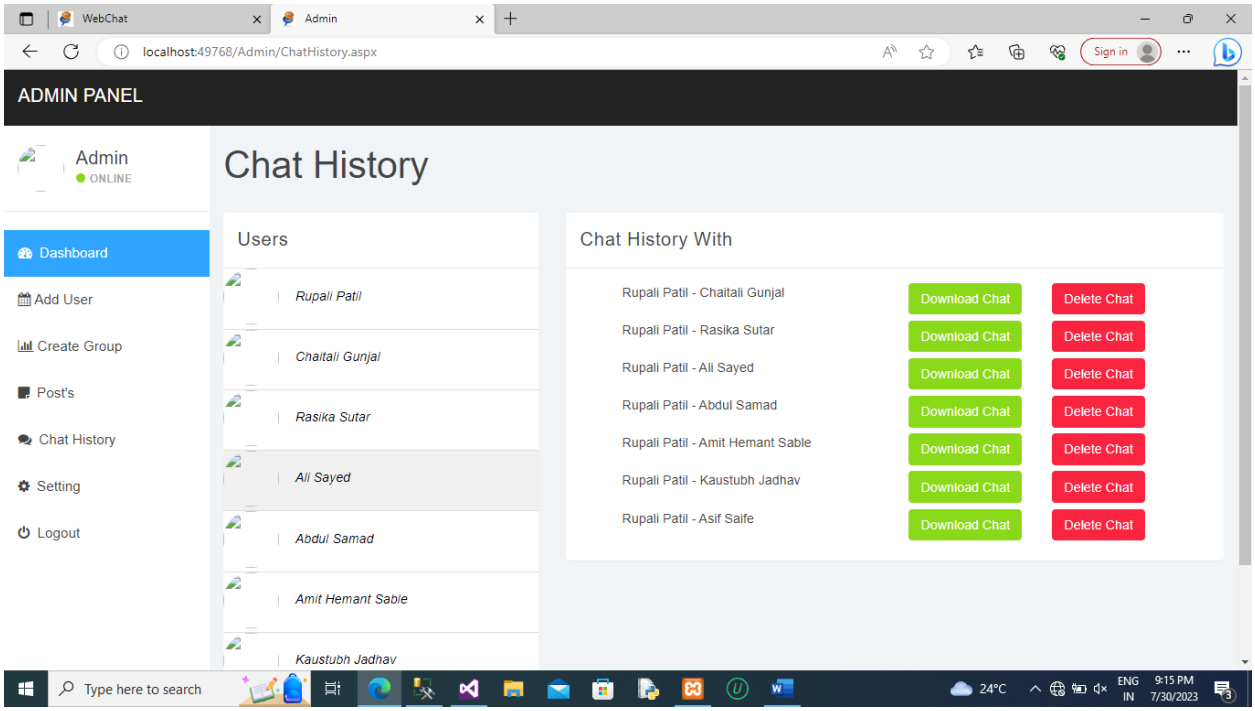



### 3.12.12 Add New Admin

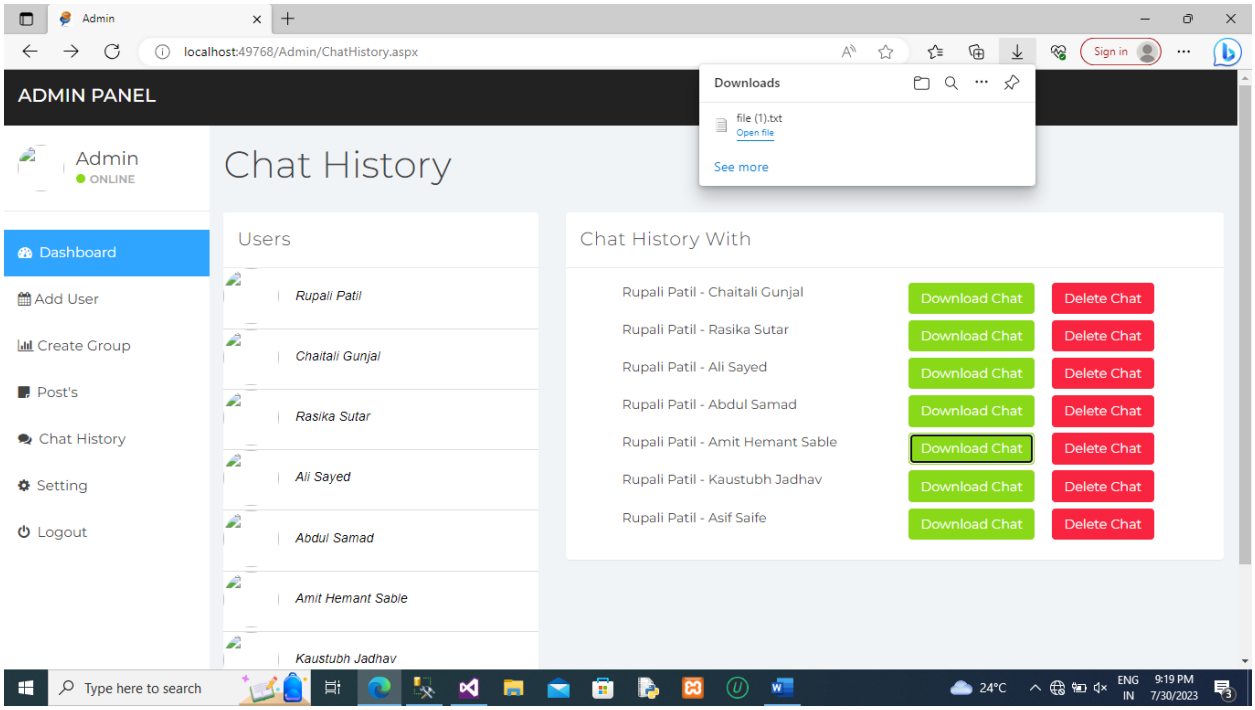


## 3.13 Report

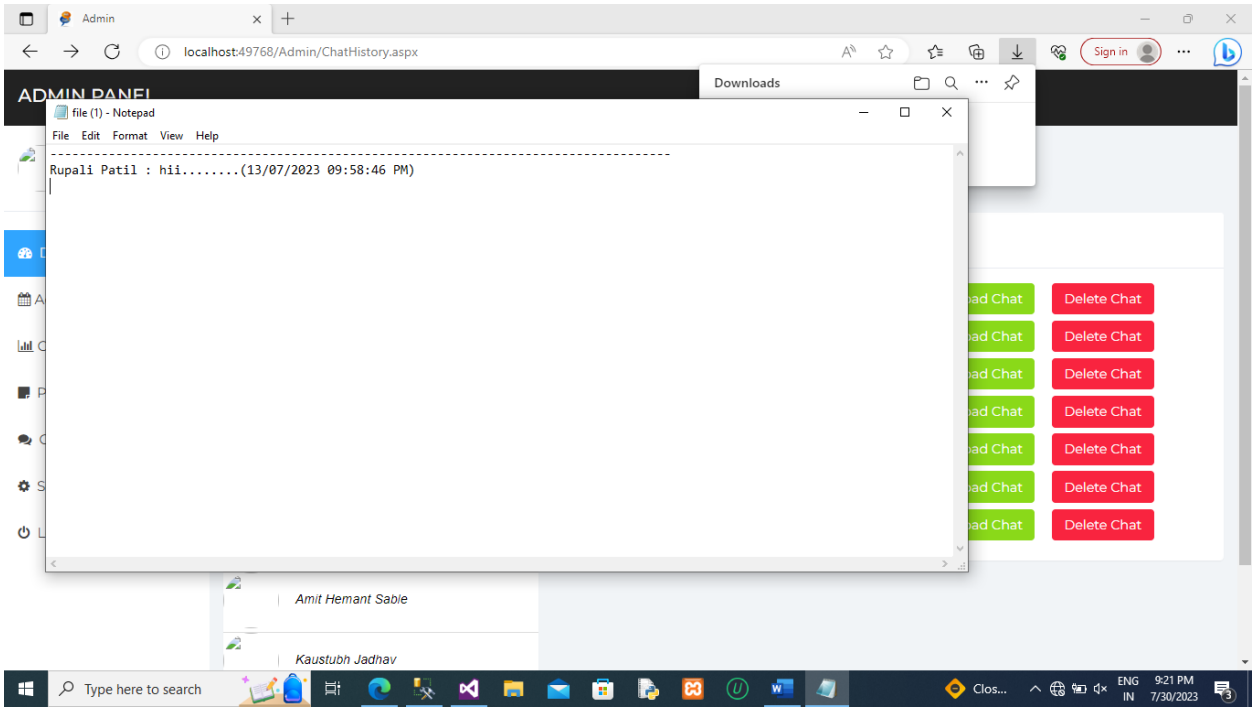
### 3.13.1 Chat History




### 3.13.2 Download Chat



### 3.13.3 Download Chat File.txt open



# CHAPTER 4

# CODING

## 4.1 Algorithms

### Algorithm 1

Designing algorithms for a "Webcaht" user dashboard involves managing various components and data efficiently to provide a smooth and interactive user experience. Below are some key algorithms and functionalities commonly used in a Webchat user dashboard:

1. **Real-Time Messaging Algorithm:** This algorithm handles the real-time exchange of messages between users. It involves managing incoming and outgoing messages, message queuing, and ensuring timely delivery and display of messages in chat threads.
2. **Contact Management Algorithm:** The contact management algorithm is responsible for handling user contacts, including adding new contacts, updating existing ones, and organizing them into groups. It should also support contact search and filtering capabilities.
3. **Notifications Algorithm:** The notifications algorithm triggers and manages notifications for users whenever they receive new messages, friend requests, or other relevant activities. It should prioritize and deliver notifications in a timely manner.
4. **Read/Seen Status Algorithm:** This algorithm tracks and updates the read or seen status of messages within a chat thread. It informs the sender when their message has been delivered, read, or seen by the recipient(s).
5. **User Authentication and Security Algorithm:** This algorithm handles user authentication and ensures the security of user data, including passwords, sessions, and other sensitive information.
6. **Group Chat Management Algorithm:** The group chat management algorithm handles the creation, maintenance, and updating of group chats. It ensures that messages are delivered to all participants and handles user invitations and removals.
7. **Message History and Search Algorithm:** This algorithm manages the storage and retrieval of past chat messages and allows users to search for specific messages or conversations efficiently.
8. **User Analytics Algorithm:** The user analytics algorithm gathers and analyzes user behaviour data, such as message frequency, engagement patterns, and active times, to improve the overall user experience and make data-driven decisions.
9. **Accessibility Algorithm:** For ensuring accessibility compliance, this algorithm optimizes the website for users with disabilities, including support for screen readers and keyboard navigation.

These algorithms work in tandem to create a feature-rich and user-friendly messenger website that meets the expectations and demands of its users. It's important to regularly review and optimize these algorithms based on user feedback and changing requirements to ensure the best possible user experience.

**Algorithm 2**

The admin panel of a Webchat application is the backend interface that allows administrators to manage and monitor various aspects of the platform. It includes tools and functionalities to moderate content, manage users, handle reported issues, analyze data, and ensure the smooth functioning of the website. Here are some key algorithms and functionalities commonly used in a Webchat admin panel:

1. **User Management Algorithm:** This algorithm handles user accounts, including user registration, authentication, and account deletion. It also manages user roles and permissions, allowing administrators to assign different levels of access and privileges.
2. **Content Moderation Algorithm:** The content moderation algorithm analyzes user-generated content, such as messages, images, and videos, to detect and filter inappropriate or harmful content. It may utilize machine learning algorithms for automated content moderation.
3. **Reporting and Issue Handling Algorithm:** When users report content or behaviour, this algorithm processes the reports, investigates the issues, and takes appropriate actions, such as warning the involved users or taking down offensive content.
4. **User Analytics Algorithm:** The user analytics algorithm tracks user behaviour and engagement patterns on the platform. It provides administrators with insights into user activity, popular features, and potential areas for improvement.
5. **Security and Privacy Algorithm:** This algorithm ensures the security of user data and communications. It may include encryption techniques to protect sensitive information and monitor security-related events.
6. **Server Load Balancing Algorithm:** For efficient resource management, the admin panel may use load balancing algorithms to distribute server load evenly, ensuring optimal performance during peak usage.
7. **Backup and Recovery Algorithm:** The backup and recovery algorithm regularly creates data backups to prevent data loss and provides a mechanism to restore data in case of a system failure.

8. **User Behaviour Analysis Algorithm:** To detect suspicious or fraudulent behaviour, this algorithm analyzes user actions and flags potentially harmful activities for further investigation.
9. **Message Archiving Algorithm:** This algorithm manages the storage and archiving of messages, ensuring that data is stored efficiently while remaining accessible for future reference.
10. **Reporting and Data Visualization Algorithm:** The admin panel may use algorithms to process and present data in a visually informative manner, enabling administrators to make data-driven decisions and gain insights from large datasets.
11. **User Activity Tracking Algorithm:** This algorithm monitors user activity and tracks login/logout times, IP addresses, and devices used for enhanced security and fraud prevention.
12. **User Communication Management Algorithm:** In case administrators need to send messages or notifications to users, this algorithm facilitates the distribution and delivery of important announcements or updates.
13. **User Support and Ticketing Algorithm:** The admin panel may incorporate a ticketing system to manage user support requests and inquiries, ensuring timely responses and issue resolution.
14. **Compliance and Regulation Algorithm:** If the messenger website is subject to specific regulations or compliance requirements, this algorithm ensures that the platform adheres to the necessary rules and guidelines.
15. **Automated Responses and FAQs:** The admin panel may utilize algorithms to generate automated responses to common user queries and frequently asked questions, reducing the load on human support staff.
16. **User Blocking and Restriction Algorithm:** This algorithm handles the suspension or blocking of users who violate the platform's terms of service or exhibit abusive behaviour.

The specific algorithms and functionalities in the admin panel depend on the platform's requirements, scale, and security considerations. Regular updates and improvements to these algorithms are necessary to maintain a safe and user-friendly messaging environment.

## 4.2 Code Snippet (Sample Code)

### 4.2.1 Admin index.aspx

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<!--
===============================================================
==========================-->
<link href="../img/logopage.png" rel="shortcut icon" />
<!--
===============================================================
==========================-->
<link rel="stylesheet" type="text/css" href="vendor/bootstrap/css/bootstrap.min.css">
<!--
===============================================================
==========================-->
<link rel="stylesheet" type="text/css" href="fonts/font-awesome-4.7.0/css/font-awesome.min.css">
<!--
===============================================================
==========================-->
<link rel="stylesheet" type="text/css" href="fonts/iconic/css/material-design-iconic-font.min.css">
<!--
===============================================================
==========================-->
<link rel="stylesheet" type="text/css" href="vendor/animate/animate.css">
<!--
===============================================================
==========================-->
```

```
<link rel="stylesheet" type="text/css" href="vendor/css-hamburgers/hamburgers.min.css">
<!--
===============================================================
============================-->
<link rel="stylesheet" type="text/css" href="vendor/animsition/css/animsition.min.css">
<!--
===============================================================
============================-->
<link rel="stylesheet" type="text/css" href="vendor/select2/select2.min.css">
<!--
===============================================================
============================-->
<link rel="stylesheet" type="text/css" href="vendor/daterangepicker/daterangepicker.css">
<!--
===============================================================
============================-->
<link rel="stylesheet" type="text/css" href="css/util.css">
<link rel="stylesheet" type="text/css" href="css/main.css">
<!--
===============================================================
============================-->
<script src="vendor/jquery/jquery-3.2.1.min.js"></script>
<script type="text/javascript">
$(document).ready(function () {
$("#btnlogin").click(function (e) {
e.preventDefault();
var email = $("#txtemail").val();
var pass = $("#txtpass").val();
$.ajax({
type: "POST",
url: "index.aspx/Login",
data: "{'email':'" + email + "','pass':'" + pass + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
```

```
if (data.d == "success") {
window.location.href = "Dash.aspx";
}
else
{
alert("Invalid Credentials");
}
},
error: function () {
alert("Error..!!");
}
});
});
});
</script>
</head>

<body>
<form id="form1" runat="server">
<div class="limiter">
<div class="container-login100">
<div class="wrap-login100">
<form class="login100-form validate-form">
<span class="login100-form-title p-b-26">Welcome
</span>
<span class="login100-form-title p-b-48">
<i class="zmdi zmdi-font"></i>
</span>

<div class="wrap-input100 validate-input" data-validate="Valid email is: a@b.c">
<input class="input100" type="text" name="email" id="txtemail">
<span class="focus-input100" data-placeholder="Email"></span>
</div>
```

```
<div class="wrap-input100 validate-input" data-validate="Enter password">
<span class="btn-show-pass">
<i class="zmdi zmdi-eye"></i>
</span>
<input class="input100" type="password" name="pass" id="txtpass">
<span class="focus-input100" data-placeholder="Password"></span>
</div>

<div class="container-login100-form-btn">
<div class="wrap-login100-form-btn">
<div class="login100-form-bgbtn"></div>
<button class="login100-form-btn" id="btnlogin">
Login
</button>
</div>
</div>

<%--<div class="text-center p-t-115">
<span class="txt1">Don’t have an account?
</span>

<a class="txt2" href="#">Sign Up
</a>
</div>--%>
</form>
</div>
</div>
</div>


<div id="dropDownSelect1"></div>
<!--
===============================================================================
============================-->
```

```
<!--
===============================================================
============================-->
<script src="vendor/animsition/js/animsition.min.js"></script>
<!--
===============================================================
============================-->
<script src="vendor/bootstrap/js/popper.js"></script>
<script src="vendor/bootstrap/js/bootstrap.min.js"></script>
<!--
===============================================================
============================-->
<script src="vendor/select2/select2.min.js"></script>
<!--
===============================================================
============================-->
<script src="vendor/daterangepicker/moment.min.js"></script>
<script src="vendor/daterangepicker/daterangepicker.js"></script>
<!--
===============================================================
============================-->
<script src="vendor/countdowntime/countdowntime.js"></script>
<!--
===============================================================
============================-->
<script src="js/main.js"></script>
</form>
</body>
</html>
```

**4.2.2 User index.aspx**

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />
<link href="img/logopage.png" rel="shortcut icon" />
<style>
body {
padding-top: 60px;
}
</style>
<link href="bootstrap3/css/bootstrap.css" rel="stylesheet" />
<link href="login-register.css" rel="stylesheet" />
<link rel="stylesheet" href="https://netdna.bootstrapcdn.com/font-awesome/4.0.3/css/font-awesome.css">

<script src="jquery/jquery-1.10.2.js" type="text/javascript"></script>
<script src="bootstrap3/js/bootstrap.js" type="text/javascript"></script>
<%--<script src="login-register.js" type="text/javascript"></script>--%>
<script type="text/javascript">
$(document).ready(function () {
//openLoginModal();
$('#loginModal').modal({ backdrop: 'static' });
$("#btnreset").click(function (e) {
e.preventDefault();
$('#modalreset').modal({ backdrop: 'static' });
});
$("#btnresetpass").click(function (e) {
e.preventDefault();
var email = $("#txtemail").val();
$.ajax({
type: "POST",
url: "index.aspx/ResetPassword",
```

```
data: "{'email':'"+email+"'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d == "success") {
$('#modalreset').modal('hide');
}
else if (data.d == "not") {
alert("Email Not Found..!!");
$("#txtemail").focus();
}
},
error: function () {
alert("Error..!!");
}
});
});
});
</script>
<script type="text/javascript">
function loginAjax() {
$("input[type=button]").attr("disabled", "disabled");
var user = $("#txtusername").val();
var pass = $("#txtpass").val();
$.ajax({
type: "POST",
url: "index.aspx/Logind",
data: "{'user':'" + user + "','pass':'" + pass + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d == "success") {
window.location = "chat.aspx";
}
else {
shakeModal();
```

```
$("#txtpass").val('');
$("input[type=button]").removeAttr("disabled");
}
},
error: function () {
alert("Error..!!");
}
});
}
function shakeModal() {
$('#loginModal .modal-dialog').addClass('shake');
$('.error').addClass('alert alert-danger').html("Invalid email/password combination");
$('input[type="password"]').val('');
setTimeout(function () {
$('#loginModal .modal-dialog').removeClass('shake');
}, 1000);
}
</script>
</head>
<body>
<form id="form1" runat="server">
<div class="container">
<div class="row">
<div class="col-sm-4"></div>
<div class="col-sm-4">
<div class="col-sm-4"></div>
</div>
<div class="modal fade login" id="loginModal">
<div class="modal-dialog login animated">
<div class="modal-content">
<div class="modal-header">
<h4 class="modal-title text-center">Login with</h4>
</div>
<div class="modal-body">
```

```
<div class="box">
<div class="content">
<div class="error"></div>
<div class="form loginBox">
<form method="post" action="/login" accept-charset="UTF-8">
<input id="txtusername" class="form-control" type="text" placeholder="Username" name="email" clientidmode="static" />
<input id="txtpass" class="form-control" type="password" placeholder="Password" name="password" clientidmode="static" />
<input class="btn btn-default btn-login" type="button" value="Login" onclick="loginAjax()" />
<div class="text-right">
<button type="button" class="btn btn-link" id="btnreset">Forget Password?</button>
</div>
</form>
</div>
</div>
</div>
</div>
<div class="modal-footer">
<div class="forgot register-footer" style="display: none">
<span>Already have an account?</span>
<a href="javascript: showLoginForm();">Login</a>
</div>
</div>
</div>
</div>
</div>

</div>
<!-- Modal -->
<div class="modal fade" id="modalreset" role="dialog">
<div class="modal-dialog modal-sm">
<div class="modal-content">
<div class="modal-header">
```

```
<button type="button" class="close" data-dismiss="modal">&times;</button>
<h4 class="modal-title">Reset Password</h4>
</div>
<div class="modal-body">
<div class="form-group">
<label for="email">Email address:</label>
<input type="email" class="form-control" id="txtemail" />
</div>
<div class="text-center">
<button type="button" class="btn btn-success" id="btnresetpass">Get Password !!</button>
</div>
</div>
</div>
</div>
</div>
</form>

</body>
</html>
```

**4.2.3 AddUser.aspx**

```
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="Server">
<script type="text/javascript">
$(document).ready(function () {
$("#btnadddept").click(function (e) {
e.preventDefault();
var name = $("#txtdeptname").val();
var desc = $("#txtdesc").val();
$.ajax({
type: "POST",
url: "AddUser.aspx/SaveDept",
data: "{'name':'" + name + "','desc':'" + desc + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d == "success") {
alert("Done");
getdepartment();
$("#txtdeptname").val('');
$("#txtdesc").val('');
}
else {
alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});
});
$("#btnadduser").click(function (e) {
e.preventDefault();
var fname = $("#txtfname").val();
var lname = $("#txtlname").val();
var desig = $("#txtdesignation").val();
```

```
var depart = $("#dlldepartment").val();
var user = $("#txtusername").val();
var pass = $("#txtpassword").val();
var email = $("#txtemail").val();
var filename = $('input[type=file]').val().replace(/C:\\fakepath\\/i, '');
var desc = $("#txtdescuser").val();
$.ajax({
type: "POST",
url: "AddUser.aspx/SaveUser",
data: "{'desc':'" + desc + "','image':'" + filename + "','fname':'" + fname + "','lname':'" + lname +
"','desig':'" + desig + "','depart':'" + depart + "','user':'" + user + "','pass':'" + pass + "','email':'" +
email + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d == "success") {
alert("Done");
uploadfiles();
getusers();
$("#txtfname").val('');
$("#txtlname").val('');
$("#txtdesignation").val('');
$("#txtusername").val('');
$("#txtpassword").val('');
$("#txtemail").val('');
$("#txtdescuser").val('');
}
else {
alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});
```

```
});
});
</script>
<script type="text/javascript">
function uploadfiles() {
var fileUpload = $("#fupdp").get(0);
var files = fileUpload.files;
var test = new FormData();
for (var i = 0; i < files.length; i++) {
test.append(files[i].name, files[i]);
}
$.ajax({
url: "UploadHandler.ashx",
type: "POST",
contentType: false,
processData: false,
data: test,
// dataType: "json",
success: function (result) {
alert(result);
},
error: function (err) {
alert(err.statusText);
}
});
}
</script>
<script type="text/javascript">
$(document).ready(function () {
getdepartment();
getusers();
$("#btnupdatedept").hide();
$("#btnupdateuser").hide();
});
```

```
</script>
<script type="text/javascript">
var deptid = "";
var userid = "";
var type = "";
var delid = "";
function getdepartment() {
$("#divdepart").html('');
$.ajax({
type: "POST",
url: "AddUser.aspx/GetDept",
data: "{}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d != "1") {
$("#divdepart").html(data.d);
}
else {
//alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});
}
function getusers() {
$("#divusers").html('');
$.ajax({
type: "POST",
url: "AddUser.aspx/GetUsers",
data: "{}",
contentType: "application/json; charset=utf-8",
success: function (data) {
```

```
if (data.d != "1") {
$("#divusers").html(data.d);
}
else {
//alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});
}
function editdept(anchor) {
deptid = anchor;
$("#btnadddept").hide();
$("#btnupdatedept").show();
$.ajax({
type: "POST",
url: "AddUser.aspx/EditDept",
data: "{'id':'" + anchor + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d != "") {
$("#txtdeptname").val(data.d.name);
$("#txtdesc").val(data.d.desc);
}
else {
alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});
```

```
}
function editusers(anchor) {
$("#txtfname").focus();
userid = anchor;
$("#btnadduser").hide();
$("#btnupdateuser").show();
$.ajax({
type: "POST",
url: "AddUser.aspx/EditUsers",
data: "{'id':'" + anchor + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d != "") {
$("#txtfname").val(data.d.fname);
$("#txtlname").val(data.d.lname);
$("#txtdesignation").val(data.d.desig);
$("#txtusername").val(data.d.username);
$("#txtpassword").val(data.d.pass);
$("#txtemail").val(data.d.email);
$("#dlldepartment").val(data.d.deptid);
$("#fupname").html(data.d.image);
$("#txtdescuser").val(data.d.desc);

}
else {
alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});

}
```

```
function deletedept(anchor) {
delid = anchor;
type = "dept";
$.ajax({
type: "POST",
url: "AddUser.aspx/CheckDeptAssign",
data: "{'id':'" + anchor + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d == "1") {
alert("Cannot Delete..As Department is assigned with Users.");
type = "";
delid = "";
}
else if (data.d == "0") {
$("#myModal").modal('show');
}
else {
//alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});

}
function deleteusers(anchor) {
delid = anchor;
type = "user";
$("#myModal").modal('show');
//to check if user exist in the group
}
</script>
```

```
<script type="text/javascript">
$(document).ready(function () {
$("#btnupdatedept").click(function (e) {
e.preventDefault();
var name = $("#txtdeptname").val();
var desc = $("#txtdesc").val();
$.ajax({
type: "POST",
url: "AddUser.aspx/UpdateDept",
data: "{'name':'" + name + "','desc':'" + desc + "','id':'" + deptid + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d == "success") {
deptid = "";
getdepartment();
$("#btnadddept").show();
$("#btnupdatedept").hide();
alert("Updated");
$("#txtdeptname").val('');
$("#txtdesc").val('');
}
else {
alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});
});
});
</script>
<script type="text/javascript">
$(document).ready(function () {
```

```
$("#btnupdateuser").click(function (e) {
e.preventDefault();
var fname = $("#txtfname").val();
var lname = $("#txtlname").val();
var desig = $("#txtdesignation").val();
var depart = $("#dlldepartment").val();
var user = $("#txtusername").val();
var pass = $("#txtpassword").val();
var email = $("#txtemail").val();
var filename = "";
if ($('input[type=file]').val().replace(/C:\\fakepath\\/i, '') != "") {
filename = $('input[type=file]').val().replace(/C:\\fakepath\\/i, '');
}
else {
filename = $("#fupname").html();
}
var desc = $("#txtdescuser").val();
$.ajax({
type: "POST",
url: "AddUser.aspx/UpdateUser",
data: "{'desc':'" + desc + "','image':'" + filename + "','fname':'" + fname + "','lname':'" + lname +
"','desig':'" + desig + "','depart':'" + depart + "','user':'" + user + "','pass':'" + pass + "','email':'" +
email + "','id':'" + userid + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {

alert("Updated");
$("#btnadduser").show();
$("#btnupdateuser").hide();
getusers();
if (data.d == "1") {
uploadfiles();
}
$("#txtfname").val('');
```

```
$("#txtlname").val('');
$("#txtdesignation").val('');
$("#txtusername").val('');
$("#txtpassword").val('');
$("#txtemail").val('');
$("#fupname").html('');
$("#txtdescuser").val('');
},
error: function () {
alert("Error..!!");
}
});
});
});
</script>
<script type="text/javascript">
$(document).ready(function () {
$("#btndelete").click(function (e) {
e.preventDefault();
$.ajax({
type: "POST",
url: "AddUser.aspx/DeleteAll",
data: "{'id':'" + delid + "','type':'" + type + "'}",
contentType: "application/json; charset=utf-8",
success: function (data) {
if (data.d == "success") {
if (type == "user") {
getusers();
}
else if (type == "dept") {
getdepartment();
}
type = "";
delid = "";
```

```
alert("Deleted Successfully");
$("#myModal").modal('hide');

}
else {
alert(data.d);
}
},
error: function () {
alert("Error..!!");
}
});
});
});
</script>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="Server">
<div class="row">
<div class="col-lg-12">
<h1 class="page-header">Users</h1>
</div>
</div>
<!--/.row-->
<div class="row">
<div class="col-md-12">
<div class="panel panel-default">
<div class="panel-heading">Add User</div>
<div class="panel-body">
<div class="col-md-4">
<div class="form-group">
<label>First Name</label>
<input type="text" id="txtfname" clientidmode="static" class="form-control" placeholder="First
Name" />
</div>
```

```
</div>
<div class="col-md-4">
<div class="form-group">
<label>Last Name</label>
<input type="text" id="txtlname" clientidmode="static" class="form-control" placeholder="Last Name" />
</div>
</div>
<div class="col-md-4">
<div class="form-group">
<label>Designation</label>
<input type="text" id="txtdesignation" clientidmode="static" class="form-control" placeholder="Designation" />
</div>
</div>
<div class="col-md-4">
<div class="form-group">
<label>Department</label>
<asp:DropDownList ID="dlldepartment" ClientIDMode="static" class="form-control" runat="server"></asp:DropDownList>

</div>
</div>
<div class="col-md-4">
<div class="form-group">
<label>Username</label>
<input type="text" id="txtusername" clientidmode="static" class="form-control" placeholder="Username" />
</div>
</div>
<div class="col-md-4">
<div class="form-group">
<label>Password</label>
```

```
<input type="text" id="txtpassword" clientidmode="static" class="form-control" placeholder="Password" />
</div>
</div>
<div class="col-md-4">
<div class="form-group">
<label>Email Id</label>
<input type="text" id="txtemail" clientidmode="static" class="form-control" placeholder="Email Id" />
</div>
</div>
<div class="col-md-4">
<div class="form-group">
<label>Description</label>
<textarea cols="5" rows="5" id="txtdescuser" class="form-control" placeholder="Description" clientidmode="static"></textarea>
</div>
</div>
<div class="col-md-4">
<div class="form-group">
<label>Upload Picture</label>
<asp:FileUpload ID="fupdp" runat="server" ClientIDMode="Static" class="form-control" />
</div>
</div>
<div class="col-md-4">
<div id="fupname" clientidmode="static">
</div>
</div>
<div class="col-md-12 text-center">
<input type="button" value="Add User" class="btn btn-success" clientidmode="static" id="btnadduser" />
<input type="button" value="Update User" class="btn btn-success" clientidmode="static" id="btnupdateuser" />
</div>
```

```
</div>
</div>
<!-- /.panel-->
</div>
</div>
<div class="row">
<div class="col-md-6">
<div class="panel panel-default">
<div class="panel-heading">Add Department</div>
<div class="panel-body">
<div class="col-md-12">
<div class="form-group">
<label>Department Name</label>
<input type="text" id="txtdeptname" clientidmode="static" class="form-control" placeholder="First Name">
</div>
</div>
<div class="col-md-12">
<div class="form-group">
<label>Desription</label>
<textarea cols="5" rows="5" class="form-control" clientidmode="static" id="txtdesc" placeholder="Description"></textarea>
</div>
</div>
<div class="col-md-12 text-center">
<input type="button" value="Add Department" class="btn btn-success" clientidmode="static" id="btnadddept" />
<input type="button" value="Update Department" class="btn btn-success" clientidmode="static" id="btnupdatedept" />
</div>
</div>
<!-- /.panel-->
</div>
</div>
```

```
<div class="col-md-6">
<div class="panel panel-default">
<div class="panel-body tabs">
<ul class="nav nav-tabs">
<li class="active"><a href="#tab1" data-toggle="tab">Department</a></li>
<li><a href="#tab2" data-toggle="tab">Users</a></li>
</ul>
<div class="tab-content">
<div class="tab-pane fade in active" id="tab1">
<h4>Department</h4>
<div id="divdepart" clientidmode="static"></div>
</div>
<div class="tab-pane fade" id="tab2">
<h4>Users</h4>
<div id="divusers" clientidmode="static" style="overflow-y: auto; max-height: 226px;"></div>
</div>
</div>
</div>
</div>
<!--/.panel-->
</div>
<!--/.col-->
</div>
<!-- Modal -->
<div class="modal fade" id="myModal" role="dialog">
<div class="modal-dialog">
<!-- Modal content-->
<div class="modal-content">
<div class="modal-header" style="background-color: #f9243f;">
<h4 class="modal-title" style="color: white;">Delete Data</h4>
</div>
<div class="modal-body">
<p>Do you want to delete the Data..!!</p>
</div>
```

```
<div class="modal-footer">
<input type="button" class="btn btn-danger" id="btndelete" clientidmode="static" value="Delete" />
<button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
</div>
</div>

</div>
</div>
</asp:Content>
```

# CHAPTER 5

# TESTING

## 5.1 Test Strategy:

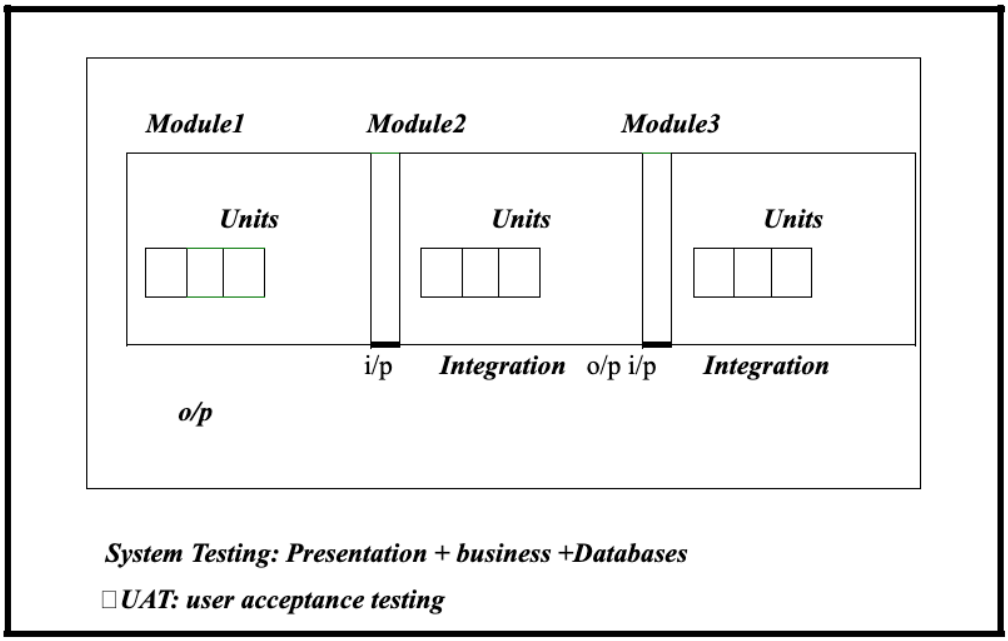
When creating a test strategy for a web chat application, it's essential to cover various aspects to ensure its functionality, usability, performance, and security. Here's a suggested approach for a comprehensive test strategy:

1. **Requirements analysis :** Review the requirements and specifications of the web chat application to gain a clear understanding of its intended functionality, supported platforms, and user expectations.
2. **Test Environment :** Set up a dedicated test environment that closely resembles the production environment, including the necessary hardware, software, browsers, and operating systems.
3. **Functional Testing :** Verify that the chat application performs as expected by conducting functional tests. This includes testing basic functionalities such as sending and receiving messages, creating and joining chat rooms, adding and removing participants, and managing user profiles.
4. **Usability Testing :** Evaluate the user-friendliness of the chat application. Test navigation, ease of use, responsiveness, and the overall user experience. Solicit feedback from potential users or conduct user testing sessions to gather valuable insights.
5. **Compatibility Testing :** Ensure that the web chat application works seamlessly across different browsers (Chrome, Firefox, Safari, etc.) and operating systems (Windows, macOS, Linux, etc.). Test on different devices such as desktops, laptops, tablets, and mobile phones.
6. **Performance Testing :** Assess the performance of the web chat application under different scenarios, including high user loads. Measure response times, latency, and server scalability to ensure the application can handle the expected user traffic.
7. **Security Testing :** Validate the security measures implemented in the chat application. Test for vulnerabilities such as cross-site scripting (XSS), SQL injection, and session management flaws. Conduct penetration testing to identify and address potential security risks.
8. **Integration Testing :** If the web chat application interacts with other systems or APIs, perform integration testing to ensure seamless communication and data exchange between the chat application and external components.
9. **Regression Testing :** Continuously conduct regression testing to ensure that new updates, bug fixes, or enhancements do not introduce any new issues or break existing functionality.

10. **Accessibility Testing :** Verify that the chat application adheres to accessibility guidelines, making it usable for individuals with disabilities. Test features such as keyboard navigation, screen reader compatibility, and color contrast.
11. **Localization Testing :** If the chat application supports multiple languages, perform localization testing to ensure correct translations, proper text alignment, and appropriate cultural adaptations.
12. 
13. **Error Handling and Recovery Testing :** Test how the chat application handles various error conditions, such as network disruptions, server failures, or unexpected user inputs. Verify that error messages are clear and informative.
14. **Data Integrity and Backup Testing :** Test data integrity mechanisms and ensure that user data is securely stored, backed up, and retrievable in case of failures or system crashes.
15. **Documentation Review :** Validate the accuracy and comprehensiveness of the application's documentation, including user manuals, installation guides, and troubleshooting guides.
16. **User Acceptance Testing (UAT) :** Involve end users or a representative sample of the target audience to perform UAT. Obtain their feedback and ensure that the application meets their requirements and expectations.
17. **Continuous Testing :** Implement an automated testing framework to facilitate continuous integration and deployment, enabling frequent testing cycles and faster feedback.

Remember to document test cases, track defects, and maintain a structured testing approach throughout the process. Regularly communicate with the development team to provide feedback and ensure that issues are resolved promptly.

## 5.2 Test Case / Test Script




**Types of Testing**

- **Smoke Testing**: is the process of initial testing in which tester looks for the availability of all the functionality of the application to perform detailed testing on them. (Main check is for available forms)
- **Sanity Testing:** is a type of testing that is conducted on an application initially to check for the proper behaviour of an application that is to check all the functionality are available before the detailed testing is conducted by on them.
- **Regression Testing:** is one of the best and important testing. Regression testing is the process in which the functionality, which is already tested before, is once again tested whenever some new change is added to check whether the existing functionality remains same.
- **Re-Testing** is the process in which testing is performed on some functionality which is already tested before to make sure that the defects are reproducible and to rule out the environment's issues if at all any defects are there.
- **Static Testing:** is the testing, which is performed on an application when it is not been executed. ex: GUI, Document Testing
- **Dynamic Testing:** is the testing which is performed on an application when it is being executed. ex: Functional testing.
- **Alpha Testing:** it is a type of user acceptance testing, which is conducted on an application when it is just before released to the customer.

- **Monkey Testing:** is the process in which abnormal operations, beyond capacity operations are done on the application to check the stability of it despite the user's abnormal behaviour.
- **Compatibility testing:** it is the testing process in which usually the products are tested on the environments with different combinations of databases (application servers, browsers…etc.) In order to check how far the product is compatible with all these environments platform combination.
- **Installation Testing:** it is the process of testing in which the tester tries to install or try to deploy the module into the corresponding environment by following the guidelines produced in the deployment document and check whether the installation is successful or not.
- **Adhoc Testing:** Adhoc Testing is the process of testing in which unlike the formal testing where in test case document is used, without that test case document testing can be done of an application, to cover that testing of the future which are not covered in that test case document. Also, it is intended to perform GUI testing which may involve the cosmetic issues.

### 5.2.1 Test Cases for User Login

| **Test Case ID** | **LOGIN_01** | | | | |
|---|---|---|---|---|---|
| **Test Case Name** | Test Case For user | | | | |
| **Prerequisite** | User Must Login First | | | | |
| **Test Step No** | **Step to be Executed** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Remark** |
| 1. | 1.Keep Username And Password Blank<br>2. Press Login Button | It Should Display The Error 'These Fields Cannot Be Blank' | Display Error Message For Both Textbox | Pass | Require Field Validation Is Checked. |
| 2. | 1.Enter Valid User Name E.g., Admin.<br>2. Enter wrong Password | It Should display the Error Message | Display error Password | Fail | Wrong Input Validation are Checked. |
| 3. | 1. Enter Special Character for Username E.g.,@%<br>2.Enter Password | It Should Not Accept Special Characters | It not Allows you to type special character except the required characters for an valid username id | Pass | Special Character are Checked. |
| 4. | 1. Enter correct Username<br>2. Enter Valid Password<br>3. Press Login Button | It Should accept username and password | It accepts username and password and forward to User Home Screen | Pass | Valid Users login is Passed. |
| 5. | 1.Enter only username and click on Login | It Should display the Error Message | It accepts username and password | Fail | Only Right Password and Login should Entered. |

### 5.2.2 Test Case for Admin Login

| **Test Case ID** | **ADMIN LOGIN_02** | | | | |
|---|---|---|---|---|---|
| **Test Case Name** | Test Case For Admin | | | | |
| **Prerequisite** | Admin Must Login First | | | | |
| **Test Step No** | **Step to be Executed** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Remark** |
| 1. | 1.Keep Username And Password Blank<br>2. Press Login Button | It Should Display The Error 'These Fields Cannot Be Blank' | Display Error Message For Both Textbox | Pass | Require Field Validation Is Checked. |
| 2. | 1.Enter Valid User Name E.g., Admin.<br>2. Enter wrong Password | It Should display the Error Message | Display error Password | Fail | Wrong Input Validation are Checked. |
| 3. | 1. Enter Special Character for Username E.g.,@%<br>2.Enter Password | It Should Not Accept Special Characters | It not Allows you to type special character except the required characters for an valid username id | Pass | Special Character are Checked. |
| 4. | 1. Enter correct Username<br>2. Enter Valid Password<br>3. Press Login Button | It Should accept username and password | It accepts username and password and forward to User Home Screen | Pass | Valid Admins login is Passed. |
| 5. | 1.Enter only username and click on Login | It Should display the Error Message | It accepts username and password | Fail | Only Right Password and Login should Entered. |

## 5.3 Defect Report / Test Log

Test Case 1: Admin login

- Login to Webchat website as an admin.
- Go to the admin panel.
- View all Chat, Group, Department details.
- Add , edit, delete, update detail
- View chat history

Test Case 2: User login

- Login to Webchat website as a user.
- Chat Personal.
- Group chat
- Update Profile
- Share Post
- Check status

Test Case 3: User registration

- Register in to the Webchat website as a User
- Go to the User Login Panel.
- Click on the "New User Register" button.
- Add the user details.
- Uid the user

Test Case 4: Account

- Log in to the Webchat as a User.
- Click on the account button .
- Update the profile.
- Change password successfully submitted.
- Log out

# CHAPTER 6

# LIMITATIONS OF PROPOSED SYSTEM

When developing a webchat project, there can be several limitations or challenges that you may encounter. Some common limitations include:

1. **Scalability:** As the number of users and messages increases, the webchat system may face scalability issues. Handling a large number of concurrent connections and ensuring efficient message distribution can be challenging.
2. **Real-time communication:** Building a real-time webchat system requires the use of technologies like WebSocket or long-polling. Implementing real-time communication can be complex and may require additional server-side infrastructure.
3. **Security:** Security is a crucial aspect of any webchat project. Ensuring the confidentiality and integrity of user messages, implementing authentication and authorization mechanisms, and protecting against common web vulnerabilities (e.g., cross-site scripting, SQL injection) require careful consideration and implementation.
4. **User experience:** Providing a seamless and responsive user experience in a webchat application can be challenging. Implementing features like message history, typing indicators, notifications, and presence management require thoughtful design and optimization.
5. **Compatibility:** Webchat projects need to be compatible with various web browsers and devices. Ensuring consistent functionality and appearance across different browsers, screen sizes, and operating systems can be a significant challenge.
6. **Data storage and retrieval:** Storing and retrieving chat messages, user profiles, and other data efficiently and reliably can be a complex task. Choosing an appropriate database solution and designing an efficient data model are crucial for the performance and scalability of the system.
7. **Moderation and filtering:** Implementing mechanisms to prevent spam, abusive content, or inappropriate behaviour in the webchat system can be challenging. Incorporating moderation tools and content filtering algorithms to maintain a safe and positive environment requires careful planning and implementation.
8. **Multilingual support:** If your webchat system aims to support multiple languages, handling internationalisation and localization can add complexity to the project. Ensuring proper translation, encoding, and language-specific formatting can be challenging.

It's important to keep these limitations in mind during the planning and development stages of a webchat project to address them effectively and provide a robust and user-friendly solution.

# CHAPTER 7

# PROPOSED ENHANCEMENTS

Certainly! I'd be happy to help you with proposed enhancements for a web chat application. Here are a few ideas:

1. **Real-time typing indicators :** Implement a feature that displays when the other person in the chat is typing. This can provide a more interactive and engaging experience by letting users know that someone is actively engaged in the conversation.
2. **Message status and read receipts :** Allow users to see if their messages have been delivered, read, or are still pending. This feature enhances communication by providing information about the status of messages, similar to popular messaging apps.
3. **Emoji and GIF support :** Integrate emoji and GIF support into the chat application, enabling users to express themselves better and add a touch of fun to their conversations. This can enhance the overall user experience and make the chat more visually appealing.
4. **File sharing :** Enable users to share files directly within the chat application. This feature can be useful for sharing documents, images, videos, or any other type of file, eliminating the need for external file-sharing services.
5. **Chatbot integration :** Integrate a chatbot into the web chat application to provide automated assistance and support. The chatbot can handle common queries, provide quick answers, and assist users with basic tasks, freeing up human agents' time for more complex inquiries.
6. **User profiles and avatars :** Allow users to create profiles and set avatars to personalize their chat experience. This can make the conversation more engaging and help users identify each other easily.
7. **Message threading :** Implement a threading feature that allows users to create sub-conversations within the main chat. This helps organize discussions around specific topics or allows users to reply directly to a particular message, keeping the conversation more structured.
8. **Push notifications :** Enable push notifications to alert users about new messages or chat activity even when they are not actively using the web chat application. This keeps users informed and allows them to respond promptly.
9. **Customizable themes :** Provide users with the ability to customize the chat application's theme, including color schemes, backgrounds, and font styles. This gives users a personalized experience and allows them to tailor the chat to their preferences.
10. **Multi-platform support :** Ensure the web chat application is accessible and functional across different platforms, including desktops, mobile devices, and tablets. This ensures

that users can seamlessly switch between devices while maintaining their chat history and preferences.

Remember to prioritize security and privacy measures, such as end-to-end encryption, to protect user data and maintain confidentiality. Additionally, gathering feedback from users and conducting usability testing can help you refine and enhance your web chat application further.

# CHAPTER 8

# CONCLUSION

In conclusion, the literature survey on webchat applications highlights several key aspects that contribute to the development and improvement of such systems. Here are some key findings and takeaways:

1. **User Experience:** Designing web chat interfaces with a focus on enhancing the user experience is crucial. Incorporating user customization options, effective interaction patterns, and notification mechanisms can greatly improve user satisfaction and engagement.
2. **Scalability:** Web chat applications face challenges related to scalability, especially when handling a large number of concurrent users. Architectural approaches, load balancing techniques, and distributed systems strategies play a vital role in ensuring system performance, low latency, and efficient resource management.
3. **Security:** Security considerations are essential for web chat applications. Implementing encryption techniques, authentication mechanisms, and privacy safeguards are critical to protect user data and ensure end-to-end security within the system.
4. **Natural Language Processing:** Integrating natural language processing (NLP) capabilities in web chat applications enables more intelligent and conversational interactions. NLP techniques such as sentiment analysis, intent recognition, and context-awareness enhance the effectiveness of chatbot systems.
5. **Real-time Collaboration:** Collaboration features in web chat applications support real-time communication, multi-user interactions, and collaborative editing. Implementing synchronization mechanisms and collaboration models fosters effective teamwork and coordination within the web chat environment.

This literature survey provides valuable insights into the different dimensions of webchat applications and serves as a foundation for understanding the best practices, challenges, and advancements in this field. Future research can further explore emerging technologies, user-centric design approaches, and innovative solutions to continuously improve web chat applications and meet the evolving needs of users.

# CHAPTER 9

# BIBLIOGRAPHY

## 9.1 BOOKS

1. ASP.NET Core 2.0 MVC & Razor Pages for Beginners
2. Murach's ASP.NET 4.6 Web Programming with C#.
3. HTML & CSS. Author (Book by Jon Duckett)
4. HTML5: The Missing Manual.( Book by Matthew MacDonald)
5. Learning Web Design: A beginner's Guide To HTML, CSS, Javascript, and Web Graphics.
6. Responsive Web Design with HTML5 and CSS3 (Book by Ben Frain)

## 9.2 WEBSITE

1. http://www.tutorialsteacher.com/core
2. https://www.c-sharpcorner.com/technologies/asp-dot-net-programming
3. https://www.tutorialride.com/asp-net/asp-net-tutorial.html
4. https://www.lynda.com/ASP-NET-training-tutorials/157-0.html

# CHAPTER 10

# USER MANUAL

## 10.1 User Manual

User manual for a webchat application should provide clear instructions and guidance to help users understand how to use the application effectively. Here are some key components and topics that could be covered in user documentation for a webchat application:

1. **Introduction:**
    a. Overview of the webchat application, its purpose, and benefits.
    b. System requirements, including supported web browsers and devices.
2. **Getting Started:**
    a. How to access and launch the webchat application.
    b. Account creation or login instructions.
    c. Familiarization with the user interface and main components.
3. **User Roles and Permissions:**
    a. Explanation of different user roles and their access levels, if applicable.
    b. How to create or manage user accounts and permissions.
4. **Chatting and Messaging:**
    a. Sending and receiving messages in real-time.
    b. Using text formatting options (e.g., bold, italics, links).
    c. Attaching files or media to messages.
    d. Emoji and sticker usage, if available.
    e. Mentioning or tagging other users in conversations.
    f. Managing chat history and searching for specific messages.
5. **Managing Contacts and Groups:**
    a. Adding and removing contacts or friends.
    b. Creating or joining chat groups and channels.
    c. Customizing contact lists or groups.
6. **Notifications and Settings:**
    a. Managing notification preferences (e.g., sound, pop-ups).
    b. Personalizing user profile information.
    c. Privacy settings and options for online status.
7. **Additional Features:**
    a. Voice or video chat capabilities, if available.
    b. Screen sharing or collaboration features, if applicable.

c. Integration with other applications or services.

**8. Troubleshooting:**

a. Common issues and error messages, along with their solutions.

b. Troubleshooting steps for connectivity problems.

c. Clearing cache or resolving browser-specific issues.

d. Contact information for technical support or assistance.

**9. Security and Privacy:**

a. Best practices for keeping conversations secure.

b. Explanation of data encryption and privacy measures.

c. Guidelines for reporting abusive or inappropriate behavior.

**10. Frequently Asked Questions (FAQ):**

a. Compilation of commonly asked questions and their answers.

b. Tips, tricks, and shortcuts for efficient usage.

## 10.2 Module Specification

1. **User Login Module**
   - User Login Module Helps user to login into the system. Only the Registered users can login into the system.
2. **Home Module**
   - Home Module helps users to interact with the screen with which the user can see the no of people to interact and can post a story and can upload a attachment with text.
   - Screen also consist with the chat screen with the user can select the user and chat with the person.
   - User can chat send images and send videos as well.
3. **Group Chat Module**
   - Here user can chat with the Group in which the admin has added to. User can be added to different group.
4. **Chat Module**
   - Chat Module consist of text chat, video, image sharing and document sharing also.
   - It consists of the timing of the message and upcoming messages.
   - User can search desired message from the chat list as well.
5. **Profile Module**
   - Here user can change his Information such as Name, Description, Image and other thing.
6. **Admin Dashboard Module**
   - Admin Dashboard consist of the User information, Group Information, Chat, Post Information Etc.
7. **Add User Module**
   - Here Admin can add user to the system an only they can log in in to the system.
   - User can be added Category wise or Team wise according the company Hierarchy.
8. **Create Group Module**
   - Admin can create a private group where specific type of people can chat or communicate with each other.
9. **Post Module**
   - Here Admin can see the post done by Users.
   - Admin can delete or view the post. He can also view the comment posted into it.
10. **Admin Chat Module**

- Here admin can see everyone chat and can clear the chat and can also view all the chat as it is made for organisations to keep the Privacy.

**11. Setting Module**

- Here admin can add new Admin. Each of the can login and view all the modules inside the Admin Module.