SOFTWARE REQUIREMENTS SPECIFICATION

**For**

**Chat Application with Message Storage**

**Prepared by:-**

*Surya prakash G*

*Ruthra Gananapathy R*

*Saravana perumal S*

# Introduction

## Purpose

An online chat application with message storage is a communication platform that enables users to exchange text-based messages in real-time while also retaining a history of these interactions. This type of application serves as a virtual space for individuals or groups to engage in conversations, fostering seamless communication and collaboration.

The primary goal of a chat application with message storage is to provide users with a convenient and efficient means of communication, allowing them to connect with each other regardless of geographical distances. By incorporating message storage capabilities, the application retains a record of conversations, enabling users to review past messages, catch up on discussions, and refer back to important information

Document Conventions

Use a consistent hierarchy for headings (e.g., "Introduction," "Features," "Architecture," "User Guide," etc.).

Consider using distinct styles or formatting for different levels of headings.

Font Styles:

Use a clear and readable font style and size for the entire document.

Consider using bold or italicized text for emphasis or important points.

Lists:

Use bullet points for lists of features, requirements, or steps.

Maintain consistent indentation for nested lists.

Code and Syntax:

Clearly format code snippets using a monospaced font.

Differentiate code from regular text through color, indentation, or other formatting.

Images and Graphics:

Include screenshots or diagrams with clear labels and captions.

Ensure images are of sufficient resolution and follow a consistent style.

Tables:

Use tables to present data or specifications.

Clearly define column headers and ensure consistent formatting.

Page Layout:

Adopt a standard page layout with consistent margins, spacing, and page numbering.

Utilize headers and footers for relevant information like section titles or page numbers.

Language Usage:

Maintain a professional and consistent tone throughout the document.

Clearly define any specific terms or acronyms used in the context of the chat application.

User Stories or Scenarios:

Present user stories or scenarios in a consistent format.

Use a standardized template for documenting user interactions.

References and Citations:

If referencing external sources or APIs, use a consistent citation style.

Include a bibliography or reference section as needed.

File Naming Conventions:

Establish a clear and logical file naming convention for any accompanying documents or resources.

Revision History:

Include a revision history section to document changes made to the document over time.

Clearly outline the date, author, and a brief description of each revision.

User Interface Mockups:

If including UI mockups or wireframes, use a consistent style for annotations and labels.

Ensure a standard format for presenting user interface elements.

Scope of Development Project

The Chat Application with Message Storage project aims to develop a robust and user-friendly messaging platform that allows users to engage in real-time communication with message storage capabilities. The application will facilitate both one-on-one and group conversations, ensuring seamless interaction and efficient message retrieval.

## Definitions, Acronyms and Abbreviations

Chat Application:

Definition: A software platform that enables users to exchange text-based messages in real-time.

Acronym: CA

Abbreviation: ChatApp

Message Storage:

Definition: The capability of the application to securely store and retrieve text and multimedia messages.

Acronym: MS

Abbreviation: MsgStorage

User Authentication:

Definition: The process of verifying and validating the identity of users accessing the application.

Acronym: UA

Abbreviation: Auth

User Profile:

Definition: A user's account information, including username, profile picture, and other relevant details.

Acronym: UP

Abbreviation: Prof

Group Chat:

Definition: A feature that allows multiple users to participate in a shared chat conversation.

Acronym: GC

Abbreviation: GroupChat

Multimedia Support:

Definition: The ability of the application to handle and share various media types, such as images and videos.

Acronym: MS

Abbreviation: MediaSup

Notifications:

Definition: Alerts or messages that inform users about new messages, events, or activities.

Acronym: N

Abbreviation: Notifs

Emojis and Reactions:

Definition: Additional visual elements users can use to express emotions or reactions in conversations.

Acronym: ER

Abbreviation: EmoReac

User Status:

Definition: An indicator of whether a user is currently online or offline.

Acronym: US

Abbreviation: UserStat

User Interface:

Definition: The graphical layout and design that allows users to interact with the chat application.

Acronym: UI

Abbreviation: UI

API:

Definition: Application Programming Interface; a set of rules that allows one software application to interact with another.

SDK:

Definition: Software Development Kit; a collection of software tools and libraries used in the development of applications.

HTTPS:

Definition: Hypertext Transfer Protocol Secure; a secure version of HTTP, the protocol over which data is sent between a user's browser and the website.

## References

**Website: React**

**Angular Documentation:**

**Website: Angular**

**Vue.js Documentation:**

**Website: Vue.js**

**Socket.io Documentation:**

**Website: Socket.io**

**Backend Development:**

**Node.js Documentation:**

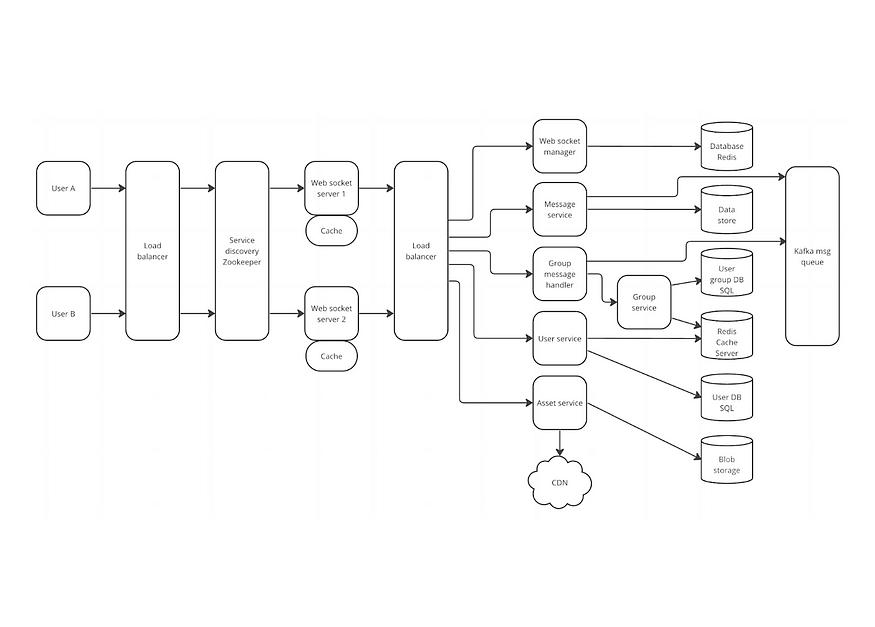
**Website: Node.js**

**Express.js Documentation:**

# Overall Descriptions

## Product Perspective

## The chat application with message storage is designed to facilitate real-time communication between users through text-based messages. The system not only provides a platform for users to exchange messages but also includes features for message storage, user authentication, and chat room management.



system.The users of the system can request issue/renew/return of books for which they would have to follow certain criteria.

## Product Function

Use case chat application with message storage

+-------------------------------------+

| Chat Application |

| |

| +-----------------------------+ |

| | User | |

| +-----------------------------+ |

| | - Register | |

| | - Log In | |

| | - Log Out | |

| | - Create Chat Room | |

| | - Join Chat Room | |

| | - Leave Chat Room | |

| | - Send Message | |

| | - Receive Message | |

| +-----------------------------+ |

| | Chat Room | |

| +-----------------------------+ |

| | - Manage Members | |

| | - Moderate Conversations | |

| +-----------------------------+ |

| | Message | |

| +-----------------------------+ |

| | - Store Message | |

| | - Retrieve Message | |

| +-----------------------------+ |

+-------------------------------+

## Operating Environment

## Choose an operating system that supports the development and deployment of your chat application. Common choices include Linux distributions (e.g., Ubuntu, CentOS), Windows Server, or other Unix-like systems.

## Web Server:

## Select a web server to handle HTTP requests. Popular choices include Apache, Nginx, or Microsoft Internet Information Services (IIS).

## Database Management System (DBMS):

## Choose a suitable database system for message storage. Options include:

## SQL databases (e.g., MySQL, PostgreSQL, Microsoft SQL Server)

## NoSQL databases (e.g., MongoDB, Cassandra)

## Backend Programming Language:

## Use a programming language for backend development. Common choices include:

## JavaScript (Node.js)

## Python (Django, Flask)

## Ruby (Ruby on Rails)

## Java (Spring)

## C# (ASP.NET)

## Real-Time Communication Protocol:

## Implement a real-time communication protocol to enable instant messaging. Common choices include WebSocket for bidirectional communication.

## Cloud Services:

## Consider using cloud services for scalability, reliability, and storage. Services such as Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) can provide hosting, storage, and other infrastructure services.

## Message Queues (Optional):

## If needed, incorporate a message queue system to manage message processing asynchronously. Options include RabbitMQ, Apache Kafka, or AWS Simple Queue Service (SQS).

## Authentication Service:

## Implement an authentication service to secure user access to the chat application. OAuth or OpenID Connect can be used for third-party authentication.

## Frontend Framework:

## Choose a frontend framework for building the user interface. Common choices include:

## React.js

## Angular

## Vue.js

## Notification Service:

## Implement a notification service for alerting users about new messages. This could involve using platform-specific services like Firebase Cloud Messaging (FCM) or Apple Push Notification Service (APNs).

## Security Framework:

## Utilize security frameworks and practices to protect against common web vulnerabilities. Implement HTTPS, input validation, and secure coding practices.

## Logging and Monitoring Tools:

## Integrate logging and monitoring tools to track application performance, detect errors, and monitor server health. Tools like ELK Stack (Elasticsearch, Logstash, Kibana) or Prometheus can be useful.

## Development and Deployment Tools:

## Use development and deployment tools for version control, continuous integration, and continuous deployment. Popular choices include Git, Jenkins, and Docker.

## Regulatory Compliance Measures:

## Implement measures to comply with data protection and privacy regulations, depending on the geographic location of users.

## Load Balancer (Optional):

## If the application is expected to have high traffic, consider using a load balancer to distribute incoming traffic across multiple servers.

## Assumptions and Dependencies

ssumption: Users are authenticated before accessing the chat application.

Dependency: Integration with a secure authentication system.

Message Encryption:

Assumption: Messages are transmitted and stored securely.

Dependency: Implementation of encryption algorithms for message security.

Message Types:

Assumption: The chat application supports text messages, multimedia messages, and potentially other types.

Dependency: Implementation of message handling for various formats.

User Presence:

Assumption: Users can see the online/offline status of their contacts.

Dependency: Real-time communication for updating user presence.

Notification System:

Assumption: Users receive notifications for new messages.

Dependency: Integration with a notification service on different platforms.

Message Storage:

Assumption: Messages are stored for future retrieval.

Dependency: Database system for message storage and retrieval.

Data Backup and Recovery:

Assumption: Adequate backup and recovery mechanisms are in place.

Dependency: Regular database backups and a robust recovery plan.

Scalability:

Assumption: The application may need to handle a large number of users and messages.

Dependency: Scalable infrastructure and load balancing.

Cross-Platform Compatibility:

Assumption: The chat application works on various devices and platforms.

Dependency: Ensuring compatibility with different operating systems and devices.

Dependencies:

Database Management System:

Dependency: Selection of a suitable database (SQL or NoSQL) for storing messages.

Real-Time Communication:

Dependency: Integration with real-time communication protocols (e.g., WebSocket) for instant message delivery.

Cloud Storage:

Dependency: Integration with cloud storage for multimedia file storage.

Message Indexing:

Dependency: Implementing indexing for efficient message retrieval.

User Interface (UI):

Dependency: Development of a user-friendly interface for the chat application.

Network Infrastructure:

Dependency: Reliable network infrastructure for message transmission.

Security Framework:

Dependency: Implementation of a robust security framework to protect user data and communications.

Regulatory Compliance:

Dependency: Ensuring compliance with data protection and privacy regulations.

Server Hosting:

Dependency: Selection of a reliable hosting provider for server deployment.

Error Handling and Logging:

Dependency: Implementation of error handling mechanisms and logging for troubleshooting.

By addressing these assumptions and dependencies, you can create a robust chat application with message storage that meets the needs of users while ensuring security, reliability, and scalability.

Internet connection

* + - Users must have their correct usernames and passwords to enter into their online accounts and do actions

The dependencies are:-

* + - The specific hardware and software due to which the product will be run
    - On the basis of listing requirements and specification the project will be developed and run
    - The end users (admin) should have proper understanding of the product
    - The system should have the general report stored
    - The information of all the users must be stored in a database that is accessible by the Library System
    - Any update regarding the book from the library is to be recorded to the database and the data entered should be correct

## Requirement

Users should be able to create accounts.

Users should log in securely.

Support for password recovery and account deletion.

User Profile:

Users can create and edit profiles.

Profile information includes a display name, profile picture, and status.

Contact Management:

Users can add, remove, and block contacts.

Ability to organize contacts into groups.

Real-Time Messaging:

Instant messaging between users.

Support for text messages, emojis, and multimedia messages.

Message Status:

Indication of message delivery and read status.

Optional typing indicators.

Message Search:

Users can search for messages by date, user, or content.

Group Chats:

Users can create and participate in group chats.

Group administrators can manage members and settings.

Notifications:

Users receive notifications for new messages.

Notification settings customization.

Message Editing and Deletion:

Users can edit or delete their messages.

Specify rules for message retention.

File Sharing:

Users can share files, images, and other multimedia content.

Specify file size limits.

Offline Messaging:

Messages should be stored and delivered upon the recipient's next login.

Security:

End-to-end encryption for message privacy.

Protection against common security threats (e.g., SQL injection, cross-site scripting).

User Blocking:

Users can block and unblock other users.

Multi-Device Support:

Seamless synchronization of messages across multiple devices.

Reporting and Moderation:

Reporting system for inappropriate content.

Moderation features for administrators.

Non-Functional Requirements:

Performance:

Low latency for message delivery.

Scalability to support a growing user base.

Reliability:

High availability with minimal downtime.

Reliable message storage and retrieval.

Scalability:

Ability to handle a large number of concurrent users.

Horizontal scalability for increased load.

Compatibility:

Cross-platform compatibility (web, mobile, desktop).

Support for multiple web browsers.

Data Backup and Recovery:

Regular automated backups of message data.

Efficient recovery mechanisms in case of data loss.

Regulatory Compliance:

Adherence to data protection and privacy regulations (GDPR, HIPAA, etc.).

User Interface:

Intuitive and user-friendly interface.

Responsive design for various screen sizes.

Security:

Regular security audits and updates.

Protection against data breaches and unauthorized access.

Logging and Monitoring:

Logging of critical events and errors.

Monitoring of system performance and usage patterns.

Integration:

Integration with third-party services (authentication, notification).

API for potential third-party developers.

Documentation:

Comprehensive documentation for users, administrators, and developers.

Compliance with Standards:

Adherence to web development standards and best practices.

Load Balancing:

Load balancing to distribute traffic across servers.

Localization:

Support for multiple languages.

Backup and Redundancy:

Redundancy measures to ensure service continuity.

Backup servers in case of primary server failure.

# System Features

# Users can send and receive real-time text messages instantly.

# Multimedia Messaging:

# Support for sharing images, videos, audio clips, and other multimedia content.

# Group Chats:

# Ability to create, join, and participate in group chats with multiple users.

# Message Editing and Deletion:

# Users can edit or delete their sent messages.

# Message Search:

# Users can search for messages by keyword, date, or user.

# Offline Messaging:

# Messages are stored and delivered when the recipient comes online.

# Typing Indicators:

# Optional indicators to show when a user is typing.

# Read Receipts:

# Indication when a message has been delivered and read.

# User Management Features:

# User Authentication:

# Secure login and authentication mechanisms.

# User Profiles:

# Users can create and customize their profiles with display names, profile pictures, and status messages.

# Contact Management:

# Ability to add, remove, and block contacts.

# Privacy Settings:

# Users can control their privacy settings, such as who can message them.

# User Blocking:

# Users can block and unblock other users.

# Security and Compliance Features:

# End-to-End Encryption:

# Ensure the security and privacy of user messages.

# Secure File Sharing:

# Encrypted transmission and storage of multimedia files.

# Compliance with Regulations:

# Adherence to data protection and privacy regulations.

# Notifications and Alerts:

# Push Notifications:

# Users receive push notifications for new messages.

# Notification Settings:

# Customizable notification preferences.

# Administration and Moderation:

# Admin Controls:

# Administrative features to manage users, groups, and settings.

# Reporting System:

# Users can report inappropriate content.

# Moderation Tools:

# Features to moderate and manage content.

# Cross-Platform and Integration:

# Cross-Platform Support:

# Availability on web browsers, mobile devices (iOS, Android), and desktop.

# Third-Party Integration:

# Integration with third-party services for authentication and notifications.

# Additional Features:

# Message Timestamps:

# Display timestamps for messages.

# Message Reactions:

# Users can react to messages with emojis.

# Status Updates:

# Users can set and update their online/offline status.

# Localization:

# Support for multiple languages.

# Backup and Recovery:

# Regular backup of messages with efficient recovery mechanisms.

# Analytics:

# Tracking and reporting on usage patterns and user engagement.

# Help and Support:

# Access to help resources and customer support.

## Safety Requirement

Implement end-to-end encryption to secure messages and prevent unauthorized access, even by the service provider.

Secure File Storage:

Ensure that multimedia files shared within the chat are stored securely with encryption.

Secure Connection (HTTPS):

Use HTTPS to encrypt data in transit between the user's device and your servers, preventing man-in-the-middle attacks.

Data Minimization:

Only collect and store necessary user data; avoid storing sensitive information unnecessarily.

User Authentication:

Implement strong authentication mechanisms to prevent unauthorized access to user accounts.

Password Security:

Enforce password policies (length, complexity) and securely store user passwords using strong hashing algorithms.

Two-Factor Authentication (2FA):

Provide users with the option to enable two-factor authentication for an added layer of security.

User Safety:

Blocking and Reporting:

Allow users to block and report others for inappropriate behavior or harassment.

Content Moderation:

Implement content moderation tools to detect and remove inappropriate content from messages.

Anti-Spam Measures:

Implement measures to detect and prevent spam messages.

Account Security:

Account Recovery:

Enable a secure account recovery process for users who forget their passwords.

Session Management:

Implement secure session management to protect against session hijacking.

Logout Functionality:

Ensure users can log out securely, and sessions are terminated properly.

Compliance and Legal:

Data Protection Compliance:

Comply with relevant data protection and privacy regulations (e.g., GDPR, HIPAA).

Terms of Service and Privacy Policy:

Clearly communicate terms of service and privacy policies to users.

Incident Response and Logging:

Logging and Auditing:

Implement comprehensive logging of user activities and security events for auditing and incident response.

Incident Response Plan:

Develop and maintain an incident response plan to address security incidents promptly.

Data Breach Notification:

Have a procedure in place to notify users promptly in the event of a data breach.

Technical Security:

Security Updates:

Regularly update and patch the application and its dependencies to address security vulnerabilities.

Firewall and Intrusion Detection:

Employ firewalls and intrusion detection systems to monitor and block suspicious activities.

Vulnerability Scanning:

Conduct regular vulnerability scans to identify and address potential security weaknesses.

User Education:

Security Awareness:

Educate users about best security practices, such as avoiding sharing sensitive information and using strong passwords.

Reporting Mechanism:

Establish a clear mechanism for users to report security concerns or vulnerabilities.

Regular Security Audits:

Security Audits:

Conduct regular security audits, including penetration testing, to identify and address security vulnerabilities.

Third-Party Security:

Assess and ensure the security practices of third-party services or APIs integrated into the chat application.

By incorporating these safety requirements, you can significantly enhance the security posture of your chat application with message storage, providing a safer and more secure environment for users. Regularly update and improve security measures in response to emerging threats and vulnerabilities.

.

## Security Requirement

## Implement end-to-end encryption for messages to ensure that only the intended recipients can decrypt and read the messages.

## Secure Storage:

## Store messages and multimedia files securely using encryption, and follow best practices for data-at-rest security.

## Data Integrity:

## Implement mechanisms to ensure the integrity of stored and transmitted data, preventing unauthorized modifications.

## Authentication and Authorization:

## Strong User Authentication:

## Enforce strong password policies and consider implementing multi-factor authentication (MFA) for added security.

## Session Management:

## Implement secure session management practices to prevent session hijacking and unauthorized access.

## Access Control:

## Apply granular access controls to ensure that users have the appropriate permissions for accessing and modifying data.

## OAuth/OpenID Connect (Optional):

## If integrating with third-party services, use secure authentication protocols like OAuth or OpenID Connect.

## Network Security:

## Secure Communication (HTTPS):

## Enforce the use of HTTPS to encrypt data in transit and protect against man-in-the-middle attacks.

## Firewalls and Intrusion Detection:

## Employ firewalls and intrusion detection systems to monitor and block malicious network activities.

## User Safety:

## Abuse Prevention:

## Implement measures to prevent and detect abusive behavior, including spam detection and content moderation.

## Blocking and Reporting:

## Allow users to block and report other users for inappropriate behavior or harassment.

## Secure Development Practices:

## Code Reviews:

## Conduct regular code reviews to identify and address security vulnerabilities in the application code.

## Secure Coding Standards:

## Adhere to secure coding standards to mitigate common vulnerabilities like SQL injection, XSS, and CSRF.

## Security Training:

## Ensure that development teams receive security training to stay informed about best practices and emerging threats.

## Compliance and Legal:

## Data Protection Compliance:

## Ensure compliance with relevant data protection and privacy regulations (e.g., GDPR, HIPAA).

## User Consent:

## Obtain user consent for data collection and processing, and clearly communicate privacy practices in the terms of service and privacy policy.

## Incident Response:

## Logging and Monitoring:

## Implement comprehensive logging of security events and establish monitoring processes to detect anomalies.

## Incident Response Plan:

## Develop and maintain an incident response plan to address security incidents promptly.

## Data Breach Notification:

## Have a procedure in place to notify users and relevant authorities in the event of a data breach.

## Updates and Patching:

## Regular Software Updates:

## Keep the application, libraries, and dependencies up-to-date with the latest security patches.

## Vulnerability Management:

## Regularly conduct vulnerability assessments and promptly address identified vulnerabilities.

## Third-Party Security:

## Security Assessments:

## Assess the security practices of third-party services, libraries, and APIs integrated into the chat application.

## Secure APIs:

## Implement secure communication with third-party APIs, following best practices for API security.

## User Education:

## Security Awareness:

## Educate users about security best practices, including the importance of strong passwords and recognizing phishing attempts.

## Reporting Mechanism:

## Provide users with a secure and easily accessible mechanism for reporting security concerns or vulnerabilities.

## Requirement attributes

## 1. erformance:

## Response Time:

## Define acceptable response times for message delivery and application responsiveness.

## Throughput:

## Specify the number of messages or requests the system should handle per unit of time.

## Scalability:

## Establish scalability requirements to ensure the application can handle an increasing number of users and messages.

## 2. Reliability:

## Availability:

## Specify the acceptable downtime and ensure the system is available when users need it.

## Fault Tolerance:

## Define mechanisms for handling and recovering from failures to ensure uninterrupted service.

## Redundancy:

## Specify requirements for redundant systems to avoid single points of failure.

## 3. Security:

## Encryption:

## Specify encryption requirements for both data in transit and data at rest.

## Authentication:

## Define requirements for user authentication, including password policies and multi-factor authentication.

## Authorization:

## Specify access control requirements to ensure users have appropriate permissions.

## Auditability:

## Define logging and auditing requirements for security monitoring and incident response.

## 4. Scalability:

## User Capacity:

## Specify the maximum number of simultaneous users the system should support.

## Message Volume:

## Define the expected message volume and scalability requirements as the user base grows.

## Load Balancing:

## Specify requirements for load balancing to distribute traffic across servers.

## 5. Compatibility:

## Cross-Platform Support:

## Specify compatibility requirements for different devices and operating systems.

## Browser Compatibility:

## Define supported web browsers and their versions.

## 6. Usability:

## User Interface (UI) Design:

## Define user interface requirements, including simplicity, consistency, and ease of use.

## Accessibility:

## Specify accessibility requirements to ensure the application is usable by individuals with disabilities.

## 7. Availability:

## Service Level Agreements (SLA):

## Specify SLA requirements for availability, response times, and support.

## Backup and Recovery:

## Define requirements for regular backups and efficient recovery mechanisms.

## 8. Data Management:

## Message Storage Capacity:

## Specify the maximum amount of data the system should store.

## Data Retention:

## Define policies for retaining or deleting stored messages over time.

## 9. Notification:

## Real-Time Notifications:

## Specify requirements for real-time notifications for new messages.

## Customization:

## Allow users to customize notification preferences.

## 10. Maintainability:

## markdown

## Copy code

## - \*Modularity:\*

## - Design the system with modular components for ease of maintenance and updates.

## - \*Documentation:\*

## - Specify requirements for comprehensive system documentation for developers and administrators.

## 11. Compliance:

## markdown

## Copy code

## - \*Regulatory Compliance:\*

## - Specify requirements for compliance with data protection and privacy regulations (e.g., GDPR, HIPAA).

## 12. Collaboration Features:

## javascript

## Copy code

## - \*Collaboration Tools:\*

## - Specify features such as file sharing, collaborative document editing, or integrations with third-party collaboration tools.

## 13. User Engagement:

## markdown

## Copy code

## - \*User Analytics:\*

## - Define requirements for tracking user engagement, behavior, and feature usage.

## - \*Feedback Mechanism:\*

## - Implement a user feedback mechanism to gather input for continuous improvement.

## 14. Integration:

## markdown

## Copy code

## - \*Third-Party Integration:\*

## - Specify requirements for integrating with third-party services, such as authentication providers or cloud storage.

## 15. Backup and Recovery:

## markdown

## Copy code

## - \*Backup Strategy:\*

## - Define a strategy for regular backups and a reliable recovery plan.

## 16. Error Handling:

## markdown

## Copy code

## - \*Graceful Degradation:\*

## - Specify requirements for graceful degradation to ensure the application rema

## Business Rules

Business rules for a chat application with message storage help define the behavior and constraints of the system, ensuring that it aligns with the goals and requirements of the business. Here are some key business rules for such an application:

User Management:

User Registration:

Users must register with a valid email address and authenticate themselves securely.

User Profiles:

Users can create and customize their profiles, including a display name and profile picture.

Contact Management:

Users can add, remove, and block contacts.

User Authentication:

Authentication mechanisms must be robust, ensuring secure access to user accounts.

Authorization:

Users have different roles (e.g., regular user, group administrator) with corresponding access permissions.

Messaging:

Real-Time Messaging:

Messages are delivered in real-time to ensure instant communication.

Message Types:

Support for various message types, including text, images, videos, and files.

Message Editing and Deletion:

Users can edit or delete their own messages within a specified time frame.

Group Chats:

Users can create, join, and manage group chats with specific rules.

Notification Preferences:

Users can customize their notification settings, including sounds, vibrations, and alerts.

Security and Privacy:

End-to-End Encryption:

Messages are end-to-end encrypted for user privacy.

Data Retention:

Define policies for storing and retaining messages, ensuring compliance with privacy regulations.

User Blocking:

Users can block and unblock other users to manage their privacy.

Content Moderation:

Implement content moderation to detect and remove inappropriate content.

Application Behavior:

Offline Messaging:

Messages are stored and delivered when the recipient comes online.

Typing Indicators:

Optional indicators show when a user is typing.

Read Receipts:

Indicate when a message has been delivered and read.

Message Search:

Users can search for messages by keyword, date, or user.

Collaboration and Engagement:

File Sharing:

Users can share files and multimedia content within the chat.

Reaction Emojis:

Users can react to messages using emojis.

Collaboration Tools (Optional):

Implement features for collaborative document editing, real-time collaboration, etc.

Compliance and Reporting:

Abuse Reporting:

Users can report inappropriate behavior or content.

Regulatory Compliance:

Ensure compliance with data protection and privacy regulations.

System Management:

User Support:

Provide user support channels and guidelines for issue resolution.

System Updates:

Specify procedures for updating the application to ensure security and feature enhancements.

Backup and Recovery:

Regularly back up user data and implement recovery procedures in case of data loss.

User Feedback:

Establish mechanisms for users to provide feedback and suggestions.

User Analytics:

Collect and analyze user behavior and engagement data for continuous improvement.

## User Requirement

User Registration and Authentication:

Requirement: Users should be able to easily register for an account using their email or social media accounts.

Requirement: The authentication process should be secure and straightforward, ensuring user data protection.

2. User Profile:

Requirement: Users can create and personalize their profiles with a display name, profile picture, and status.

Requirement: Ability to update and modify profile information easily.

3. Contact Management:

Requirement: Users can add, remove, and manage contacts.

Requirement: Blocking and unblocking of contacts for privacy control.

4. Real-Time Messaging:

Requirement: Instantaneous message delivery for real-time communication.

Requirement: Text messaging, supporting multimedia content like images, videos, and files.

5. Group Chats:

Requirement: Users can create and participate in group chats.

Requirement: Group administrators have the ability to manage members and settings.

6. Message Editing and Deletion:

Requirement: Users can edit or delete their sent messages within a specified time frame.

7. Notifications:

Requirement: Users receive notifications for new messages.

Requirement: Customizable notification settings based on user preferences.

8. Security and Privacy:

Requirement: End-to-end encryption to ensure the security and privacy of messages.

Requirement: Users can control their online/offline status and visibility.

9. Offline Messaging:

Requirement: Messages are stored and delivered when the recipient is offline.

10. Read Receipts:

Requirement: Indication when a message has been delivered and read by the recipient.

11. Message Search:

Requirement: Users can search for specific messages using keywords, dates, or sender names.

12. File Sharing:

Requirement: Users can share files, images, and multimedia content within the chat.

13. User Blocking:

Requirement: Users have the ability to block and unblock other users.

14. Cross-Platform Compatibility:

Requirement: The application should work seamlessly across various platforms, including web, mobile, and desktop.

15. User Experience (UX):

Requirement: Intuitive and user-friendly interface design.

Requirement: Responsive design for different screen sizes and devices.

16. Feedback and Support:

Requirement: Users can provide feedback or report issues easily.

Requirement: Access to user support and help resources within the app.

17. Compliance:

Requirement: The app complies with data protection and privacy regulations (e.g., GDPR).

18. Backup and Recovery:

Requirement: Regular backup of messages and a reliable recovery mechanism.

19. Status Updates:

Requirement: Users can set and update their status (online, offline, away).

20. Message Reactions:

Requirement: Users can react to messages using emojis or other symbols.

21. Localization:

Requirement: Support for multiple languages to accommodate a diverse user base.

# Other Requirements

# [11:16 am, 05/12/2023] +91 88254 53562: User Registration and Authentication:

# Requirement: Users should be able to easily register for an account using their email or social media accounts.

# Requirement: The authentication process should be secure and straightforward, ensuring user data protection.

# 2. User Profile:

# Requirement: Users can create and personalize their profiles with a display name, profile picture, and status.

# Requirement: Ability to update and modify profile information easily.

# 3. Contact Management:

# Requirement: Users can add, remove, and manage contacts.

# Requirement: Blocking and unblocking of contacts for privacy control.

# 4. Real-Time Messaging:

# Requirement: Instantaneous message delivery for real-time communication.

# Requirement: Text messaging, supporting multimedia content like images, videos, and files.

# 5. Group Chats:

# Requirement: Users can create and participate in group chats.

# Requirement: Group administrators have the ability to manage members and settings.

# 6. Message Editing and Deletion:

# Requirement: Users can edit or delete their sent messages within a specified time frame.

# 7. Notifications:

# Requirement: Users receive notifications for new messages.

# Requirement: Customizable notification settings based on user preferences.

# 8. Security and Privacy:

# Requirement: End-to-end encryption to ensure the security and privacy of messages.

# Requirement: Users can control their online/offline status and visibility.

# 9. Offline Messaging:

# Requirement: Messages are stored and delivered when the recipient is offline.

# 10. Read Receipts:

# Requirement: Indication when a message has been delivered and read by the recipient.

# 11. Message Search:

# Requirement: Users can search for specific messages using keywords, dates, or sender names.

# 12. File Sharing:

# Requirement: Users can share files, images, and multimedia content within the chat.

# 13. User Blocking:

# Requirement: Users have the ability to block and unblock other users.

# 14. Cross-Platform Compatibility:

# Requirement: The application should work seamlessly across various platforms, including web, mobile, and desktop.

# 15. User Experience (UX):

# Requirement: Intuitive and user-friendly interface design.

# Requirement: Responsive design for different screen sizes and devices.

# 16. Feedback and Support:

# Requirement: Users can provide feedback or report issues easily.

# Requirement: Access to user support and help resources within the app.

# 17. Compliance:

# Requirement: The app complies with data protection and privacy regulations (e.g., GDPR).

# 18. Backup and Recovery:

# Requirement: Regular backup of messages and a reliable recovery mechanism.

# 19. Status Updates:

# Requirement: Users can set and update their status (online, offline, away).

# 20. Message Reactions:

# Requirement: Users can react to messages using emojis or other symbols.

# 21. Localization:

# Requirement: Support for multiple languages to accommodate a diverse user base.

# [11:17 am, 05/12/2023] +91 88254 53562: 22. System Performance:

# Requirement: The system should handle a high volume of concurrent users without significant performance degradation.

# Requirement: Define acceptable response times for message delivery and system interactions.

# 23. Scalability:

# Requirement: The application should scale horizontally to accommodate an increasing number of users.

# Requirement: Evaluate and plan for scalability challenges, especially during peak usage times.

# 24. Network and Connectivity:

# Requirement: The application should function reliably under varying network conditions, including low bandwidth and intermittent connectivity.

# Requirement: Implement efficient network protocols for message transmission.

# 25. Third-Party Integrations:

# Requirement: Support integration with third-party services for features like authentication, notifications, or file storage.

# Requirement: Ensure compatibility with popular third-party APIs and services.

# 26. Data Backup and Recovery:

# Requirement: Regularly back up the message data to prevent data loss.

# Requirement: Establish a robust recovery mechanism in case of data corruption or loss.

# 27. Regulatory Compliance:

# Requirement: Comply with relevant legal and regulatory requirements for data protection, privacy, and user rights.

# Requirement: Stay informed about changes in regulations and update the system accordingly.

# 28. Cross-Browser Compatibility:

# Requirement: Ensure the application works consistently across various web browsers, including Chrome, Firefox, Safari, and Edge.

# 29. Security Audits:

# Requirement: Conduct regular security audits, including penetration testing, to identify and address vulnerabilities.

# Requirement: Stay informed about security best practices and adapt the system accordingly.

# 30. Maintenance and Updates:

# Requirement: Provide a mechanism for applying updates and patches to address security vulnerabilities and introduce new features.

# Requirement: Minimize downtime during maintenance activities.

# 31. Accessibility:

# Requirement: Design the application with accessibility in mind to ensure it is usable by individuals with disabilities.

# Requirement: Comply with accessibility standards such as WCAG.

# 32. System Logs:

# Requirement: Implement comprehensive logging for monitoring system activities and troubleshooting.

# Requirement: Log security events, user actions, and errors for auditing purposes.

# 33. Version Control:

# Requirement: Implement version control for the application's codebase to track changes and manage releases effectively.

# 34. Load Balancing:

# Requirement: Use load balancing to distribute incoming traffic across multiple servers, ensuring optimal performance.

# 35. Cross-Device Synchronization:

# Requirement: Ensure seamless synchronization of messages across multiple devices for users who access the application from different platforms.

# 36. Error Handling and User Feedback:

# Requirement: Implement effective error handling mechanisms to provide meaningful error messages to users.

# Requirement: Include user-friendly prompts for reporting issues and providing feedback.

# 37. Backup Servers:

# Requirement: Have backup servers in place to ensure service continuity in the event of server failures.

# 38. Data Migration:

# Requirement: Plan for smooth data migration processes when introducing new features or updating the system.

# 39. API for Third-Party Developers (Optional):

# Requirement: Provide documentation and support for external developers interested in creating applications that integrate with your chat app.

## Class Diagram

A class is an abstract, user-defined description of a type of data. It identifies the attributes of the data and the operations that can be performed on instances (i.e. objects) of the data. A class of data has a name, a set of attributes that describes its characteristics, and a set of operations that can be performed on the objects of that class. The classes’ structure and their relationships to each other frozen in time represent the static model. In this project there are certain main classes

which are related to other classes required for their working. There are different kinds of relationships between the classes as shown in the diagram like normal association, aggregation, and generalization. The relationships are depicted using a role name and multiplicities. Here ‘Librarian’, ‘Member’ and ‘Books’ are the most important classes which are related to other classes.

+--------------------+ +----------------------+ +----------------------+

| User | | ChatRoom | | Message |

+---------------------+ +----------------------+ +----------------------+

| - UserID |1 \*| - ChatRoomID | | - MessageID |

| - Username |---------| - RoomName | | - SenderID |

| - PasswordHash | | | | - ChatRoomID |

| - Email | | | | - Content |

| - RegistrationDate | +----------------------+ | - Timestamp |

+---------------------+ +--------…