Project Design Phase-I

Solution Architecture

Date	23 October 2023
Team ID	Team-591094
Project Name	Chatconnect - A Real-Time Chat And Communication App
Maximum Marks	4 Marks

Solution Architecture:

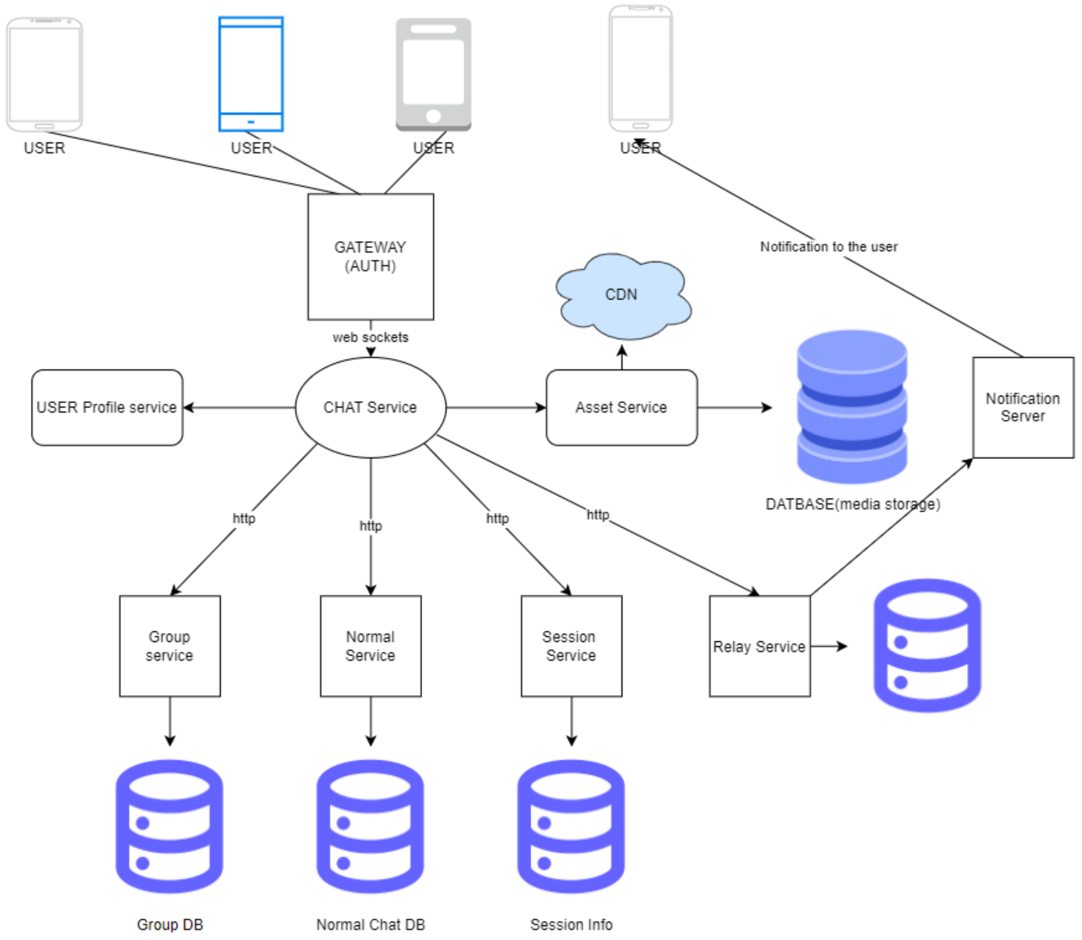
Designing a solution architecture for a chatting app can be a complex task, depending on the scale, features, and requirements of the application. Here's a high-level overview of the components and considerations for building a chatting app architecture:

1. **Client Applications:**
    - **Mobile Apps:** Android app for users to send and receive messages.
2. **Authentication and User Management:**
    - Implement user registration, login, and profile management.
    - Use secure authentication mechanisms such as OAuth, JWT, or multi-factor authentication (MFA) to ensure data security.
3. **Real-Time Messaging:**
    - Implement real-time communication using WebSockets, MQTT, or a similar protocol to enable instant messaging.
    - Use a message broker or WebSocket server to handle real-time messaging between clients.
4. **Message Storage:**
    - Store messages in a database for users to access their chat history.
    - Use a relational or NoSQL database to store messages efficiently.
    - Implement end-to-end encryption for message security.
5. **Push Notifications:**
    - Implement push notification services (e.g., Firebase Cloud Messaging, Apple Push Notification Service) to alert users of new messages when the app is in the background.
6. **User Presence Management:**
    - Implement presence indicators (online, offline) to show whether users are currently active.
    - Utilize techniques like WebSocket heartbeats to monitor user presence.
7. **Group Chat and Channels:**
    - Support both one-on-one and group chat functionalities.
    - Implement channels or chat rooms for multiple users to join and communicate.
8. **Notifications and Alerts:**
    - Implement a notification system to notify users of new messages, friend requests, or other events.
    - Allow users to customize their notification settings.
9. **User Profile and Friends:**
    - Enable users to manage their profiles and connect with friends.
    - Implement friend requests and contact management features.
10. **Content Moderation and Reporting:**
    - Implement content moderation tools to filter out inappropriate content.
    - Allow users to report abusive or harmful content.
11. **Scalability and Load Balancing:**
    - Design the architecture to handle increased traffic by using load balancers and auto-scaling capabilities.
12. **Data Security and Encryption:**
    - Prioritize data security and encryption, especially for user data and messages.
    - Implement end-to-end encryption to protect message content.
13. **Backups and Disaster Recovery:**
    - Regularly backup user data to prevent data loss.
    - Have a disaster recovery plan in place.
14. **Compliance and Legal Considerations:**

- Ensure that the app complies with relevant data protection laws and regulations, such as GDPR.

15. **Cloud Hosting and Infrastructure:**
    - Choose a reliable cloud service provider for hosting your backend servers and databases.
16. **APIs and Integrations:**
    - Provide APIs to allow third-party integrations, such as authentication through social media accounts.
17. **Documentation and Support:**
    - Create clear documentation for developers, and provide user support through various channels.

Solution Architecture Diagram


USER
USER
USER
USER
GATEWAY (AUTH)
Notification to the user
CDN
web sockets
USER Profile service
CHAT Service
Asset Service
Notification Server
DATBASE(media storage)
http
http
http
http
Group service
Normal Service
Session Service
Relay Service
Group DB
Normal Chat DB
Session Info