DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
Discover. Learn. Empower.

EXPERIMENT NO: 5

Student Name: Likhil N Maiya **UID:** 23BCS11938

Branch: BE-CSE **Section/Group:** KRG-1B

Semester: 6th **Date of Performance:** 24/02/2026

Subject Name: System Design **Subject Code:** 23CSH-314

Aim

To design a scalable texting platform where people can send messages to individuals, they can form groups and send text messages visible to that particular individual only or to the entire group. They also get the feature to video or voice call individuals or a group of people and upload photos and videos for just 24hrs to be visible to whom so ever they like.

Objectives

- Design a scalable real-time messaging system similar to WhatsApp
- Enable instant communication between users worldwide
- Support text, media, voice, and video communication
- Ensure high availability and low latency messaging
- Maintain strong privacy and end-to-end encryption
- Handle billions of messages daily efficiently
- Provide seamless user experience across devices
- Support horizontal scaling as user base grows
- Ensure reliable message delivery and synchronization

Procedure

Requirement Analysis

- Identify functional and non-functional needs
- Define scale assumptions (users, messages/sec)

System Architecture Design

- Choose distributed architecture
- Define core services:
    - Authentication Service

  - Messaging Service
  - Media Service
  - Notification Service
  - Presence Service

Communication Protocol Design

- Use WebSockets for real-time messaging
- HTTP/HTTPS for APIs

Data Storage Design

- User database
- Message database
- Media object storage
- Cache layer (Redis)

Scalability Planning

- Load balancers
- Sharding strategy
- Horizontal scaling

Reliability Design

- Message queues
- Retry mechanisms
- Replication

Security Design

- Encryption protocols
- Authentication tokens
- Key exchange system

Deployment Design

- Cloud infrastructure
- CDN for media delivery
- Monitoring tools

Functional Requirements

Clients /User Management

1. User registration using phone number
2. OTP verification and authentication
3. Profile creation and updates
4. Contact synchronization
5. User presence status (online/offline/last seen)

Messaging Features

6. One-to-one messaging
7. Group messaging
8. Message delivery confirmation (sent, delivered, read receipts)
9. Real-time message synchronization
10. Offline message storage and delivery
11. Message forwarding and replying

Media Sharing

12. Send images, videos, documents, audio files
13. Media upload and download
14. Media compression

Communication Features

15. Voice calls
16. Video calls
17. Voice messages

Notifications

18. Push notifications for new messages
19. Message alerts when app is closed

Groups & Communities

20. Create groups
21. Add/remove participants
22. Admin controls
23. Group metadata updates

Security

24. End-to-end encryption
25. Secure authentication
26. Block/report users

Multi-device Support

27. Sync messages across multiple devices
28. Web and desktop access

Non-Functional Requirements

Performance

- Message delivery latency < 100–300 ms
- Support millions of concurrent users
- Fast media upload/download

Scalability

- Horizontal scaling of servers
- Distributed architecture
- Load balancing

Availability

- 99.99% uptime target
- Fault-tolerant services
- Automatic failover mechanisms

Reliability

- Guaranteed message delivery
- No message loss
- Retry mechanisms

Security

- End-to-end encryption
- Secure key management
- Data privacy compliance

Consistency

- Message ordering per conversation
- Eventual consistency across devices

Maintainability

- Modular microservices architecture
- Easy deployment and updates

Observability

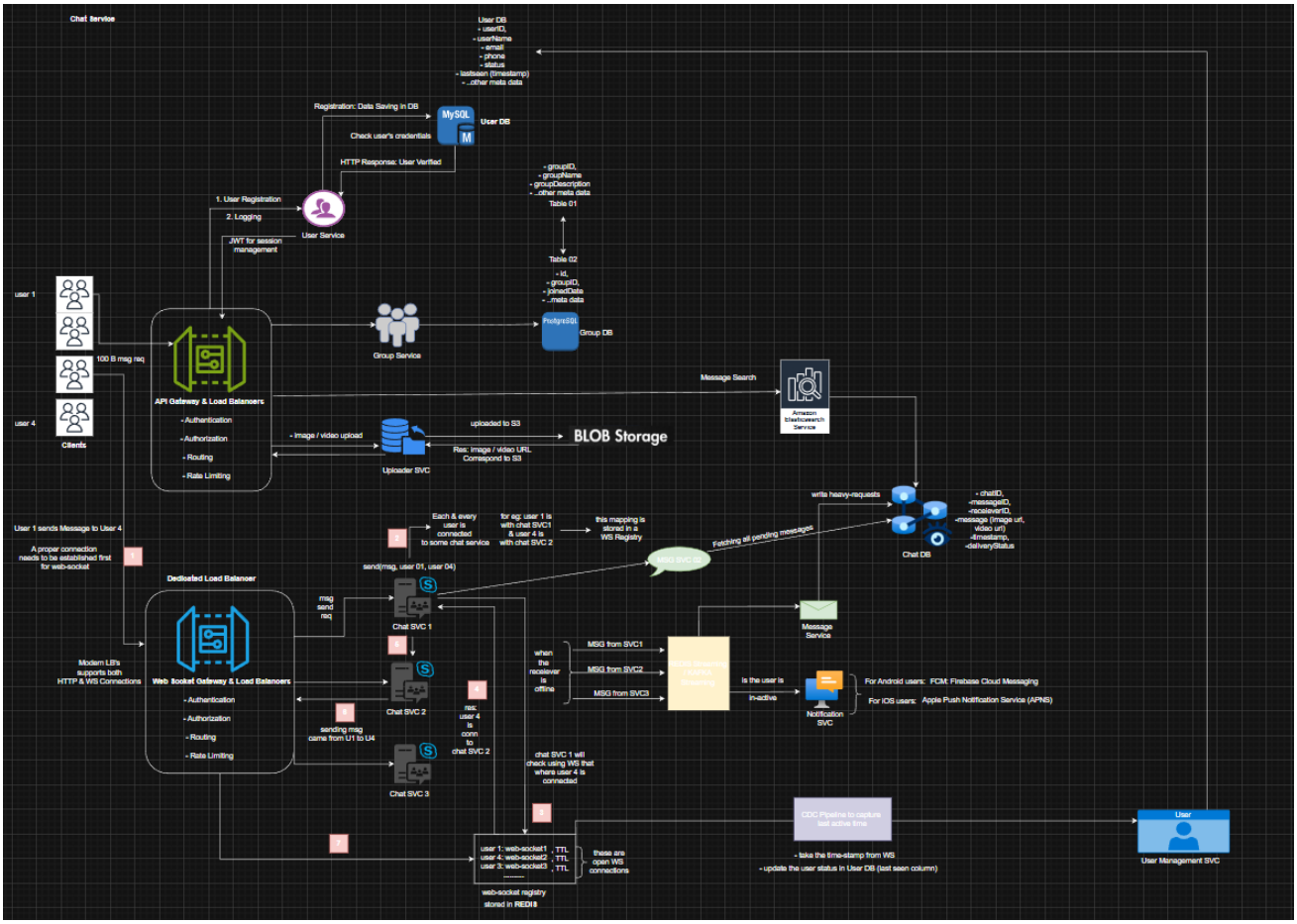
- Logging
- Monitoring
- Alerting systems

Storage Efficiency

- Efficient database indexing
- Media stored separately (object storage)

High Level Design (HLD)

The system consists of Client Applications, API Gateway, User Service, Subscription Service, Search Service, Video Streaming Service, Recommendation Engine, CDN, Cache, Message Queue, and Distributed Databases. Video content is stored in object storage and delivered through CDN for low latency streaming.



Outcome

- Understand large-scale distributed system architecture
- Design real-time messaging systems
- Apply scalability and load balancing concepts
- Understand microservices architecture
- Implement messaging queues and event-driven systems
- Design secure communication systems
- Analyze trade-offs between consistency and availability
- Learn database partitioning and replication
- Understand real-time protocols (WebSockets)
- Apply system design best practices used in industry