Chat App(WhatsApp/Skype)

Saturday, 21 October 2023 6:42 PM

Functional Requirements

1. One-on-one chat
2. Group chat
3. Read receipt
4. Online Status
5. Notifications
6. Share Multimedia(images/videos)

Non-Functional / System Requirements

1. Low Latency
2. High Availability
3. High Scalability
4. Low Lag

Capacity Planning

1. Total Active Users : 500 million
2. 20 messages per day per user
3. Total messages per day = 500M * 20
   = 1000 Million
   = 1 Billion messages per day
4. Messages per second = 1B / 60 * 60 * 24
   = ~ 11.6 K messages/second

Storage Estimation

→ Total messages per day = 1 Billion

→ each message ≈ 50kB

→ Total Storage ≈ 1 Billion * 50kB
= 50 peta Bytes

API End Points

From Users Perspective

Send_Message
: (sender_userID, receiver_UserID, text)

Get_Message
: (user_Id, before_timestamp)

Protocol

↳ Since we want bidirectional flow of data to reduce lag since it is a real time chat app so we use websockets #WHY

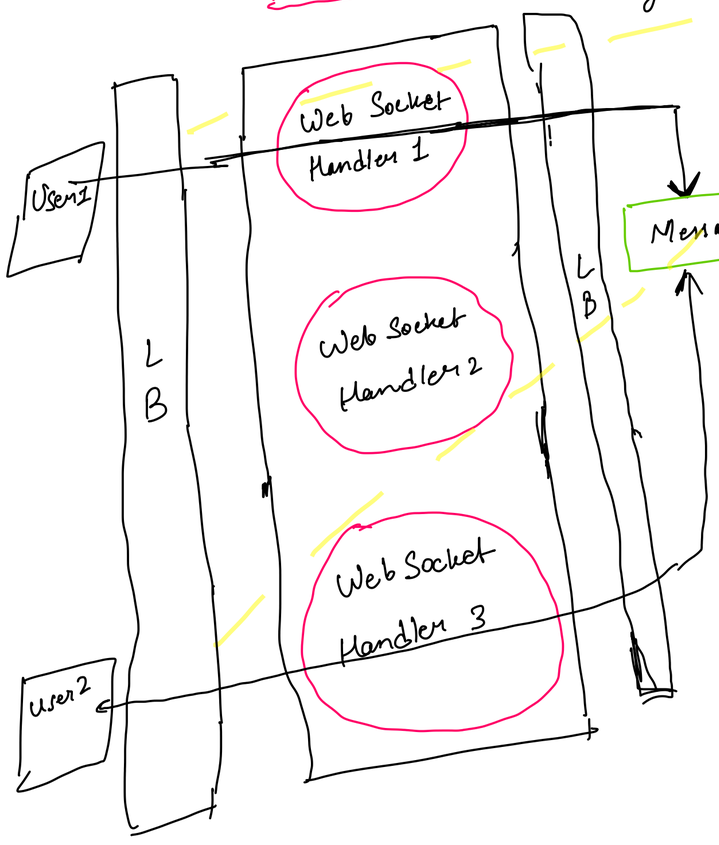
Services

1. Messaging Service → Used to Send Message from one User 1 to User 2
2. Web Socket Manager Service → Used to resolve user_id & the web socket handler it is connected to
3. Relay Service → helps in storing the message when the receiver user id is offline
4. Group Service
5. Last Seen Service → used to show online status
6. Asset Service

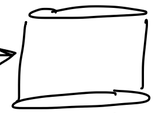
Messaging Service

Web socket handler (light weight server which keeps open bidirectional connection to user)

WEB SOCKET → Persistent Connection using Web Sockets



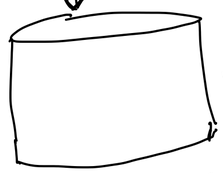
Messaging Service

Last Seen Service → 

used to show online status.

user, timestamp mapping DB

Messaging Service ↔ Web Socket Manager

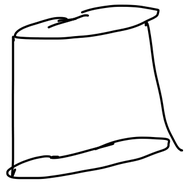
Web Socket Manager → 

user-server mapping

user_id, server mapping

We will use No SQL Database (Cassandra)

Since no relational data

Relay Service → 

unsent_messages, from_user_id to_user_id

Cassandra db can be used

Database Schema

T_users

userId	userName	contact

T_groups

groupId	userIds

T_user_session

user_id	websocket_handler_id

T_lastSeen

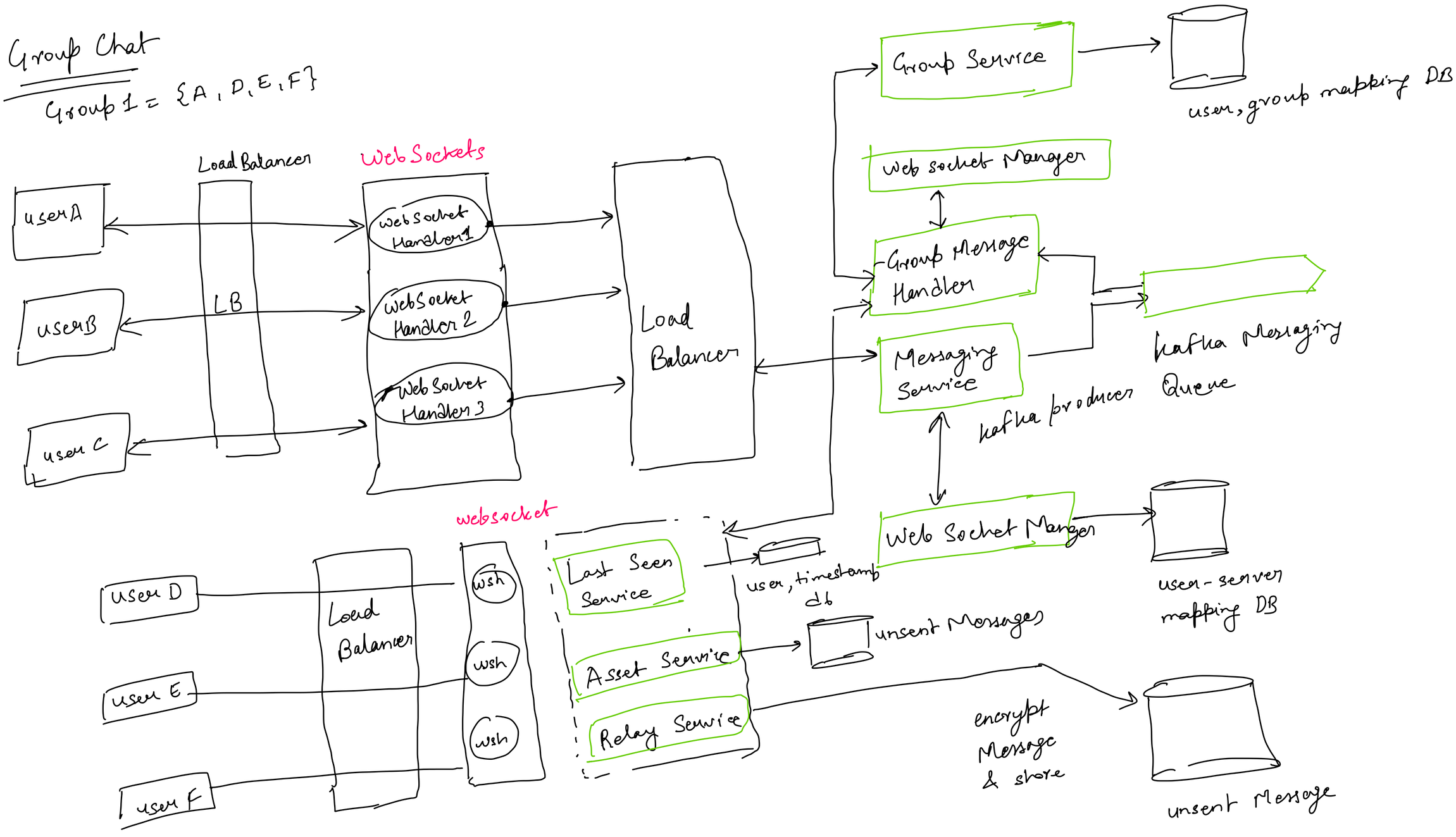
userId	timestamp

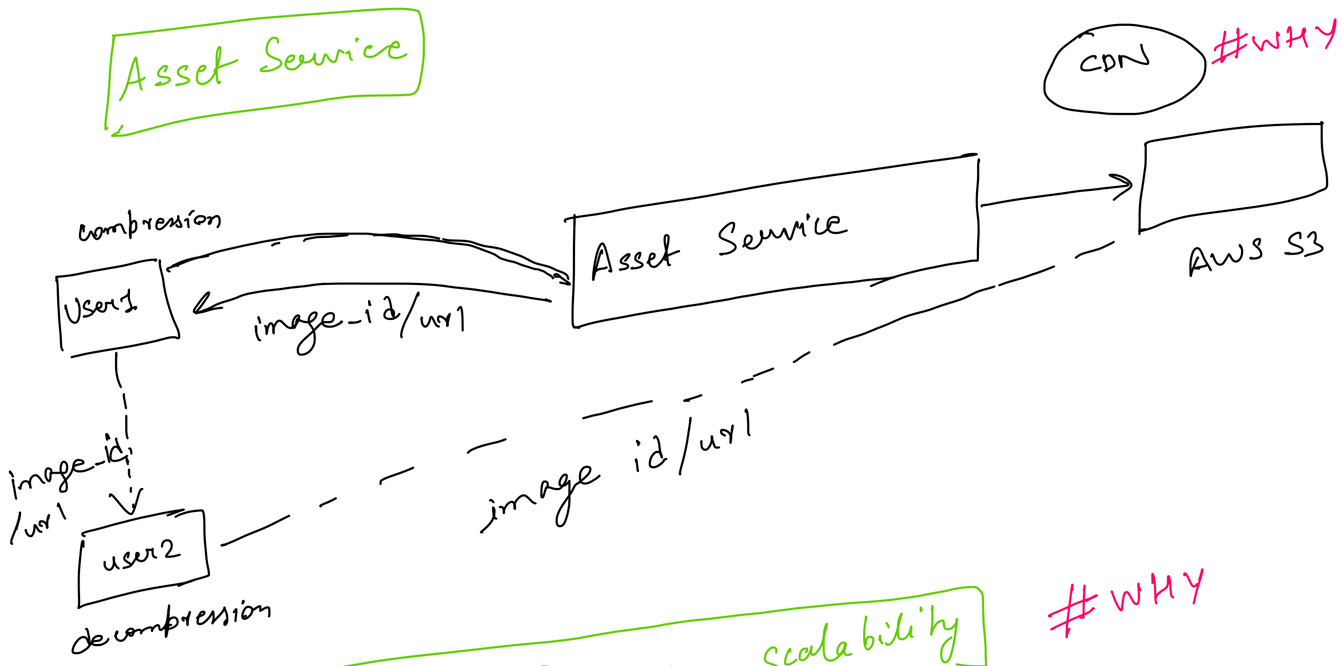
T_Unsent Messages

messageId	sent_to_id	sent_from_id	content	media_url	timestamp

Group Chat

Group 1 = {A, D, E, F}





To improve performance, scalability #WHY

→ Cache (user_id, websocket_handler) in websocket handler #WHY : To Improve performance with low TTL

→ For images which are popular
↳ Before uploading to S3 check if it is already present in S3 through hash

→ We can use nosql databases as we have basic queries & we want system to be highly available #WHY

→ We are using kafka for group messaging to improve performance #WHY

→ Horizontally scaling the services to accomodate more requests