Experiment No: 03

Student Name: Nitish Kumar Singh
UID: 23BCS12331
Branch: BE-CSE
Section/Group: KRG_2B
Semester: 6th
Date of Performance: 28/01/26
Subject Code: 23CSH-314
Subject Name: System Design

Aim:

To design a social media platfrom similar to Facebook or Instagram.

Objective:

I. To design a social media platform similar to Facebook or Instagram.
II. To identify functional and non-functional requirements of a large-scale social media system.
III. To define core entities involved in a social media application.
IV. To design RESTful API endpoints for user onboarding, posts, and interactions.

Procedure:

1. Study the working of popular social media platforms such as Facebook and Instagram.
2. Identify the main features required for a social media application.
3. List the functional requirements such as user registration, posting content, following users, and interacting with posts.
4. Analyze non-functional requirements including scalability, availability, consistency, and latency.
5. Identify the core entities required for the system such as users, posts, followers, feeds, likes, and comments.
6. Design API endpoints for:
   • User onboarding (registration, login, profile management)
   • Post creation, retrieval, update, deletion, and feed pagination
   • User interactions such as likes, comments, follow and unfollow
7. Document all objectives, requirements, entities, and APIs in a structured format.

Functional Requirements:

I. Client should be able to register and login to the application.
II. Client should be able to create post (text / image / videos)
III. Client should be able to follow each other (or send friend requests)
IV. Client should be able to like or comment on the post \
V. Client should be able to view the feed of post from users they follow.

Non Functional Requirements:

I. Scalability: The system should support around 500 million Daily Active Users (DAU) by using distributed architecture, horizontal scaling, load balancing, and microservices so that traffic spikes can be handled without performance degradation.

II. Availability over Consistency: For a social media application, high availability must be prioritized over strict consistency because if the platform is down when users want to access it, the application loses its value regardless of how consistent the data is.

III. Reason for Prioritizing Availability: An outage (for example, Instagram being down for one hour) creates a major business and user-experience problem, whereas a small delay in data synchronization is generally acceptable.

IV. Consistency Expectation: Social media platforms typically follow eventual consistency, meaning a post may take a short time to appear to followers, but the system will become consistent shortly after.

V. Latency Requirement: The post publishing or upload speed should be around 500 ms, which feels almost instant to users and maintains a smooth experience without sacrificing system availability.

Api Endpoints:

A. User On-boarding API's

1. User Registeration: POST API CALL : POST / api / users / register_user
2. User Login: POST API CALL: POST / api / users / login
3. User Data Display: GET API CALL: GET / api / users / {user_id} / profile
4. User Data Update: PUT API CALL: PUT / api / users / {user_id} / profile

B. User Post's

1. POST / api / user_id / posts
2. GET / api / posts / {post_id}
3. PUT / api / posts / {post_id}
4. DELETE / api / posts / {post_id}
5. GET / api / posts / feed / limit = {limit} & offset = {offset} : PAGINATION
6. GET / api / users / {user_id} / posts: PAGINATION

C. User Interactions

1. POST / api / posts / {post_id} / like
2. DELETE / api / posts / {post_id} / unlike
3. POST / api / posts / {post_id} / comments
4. GET / api / posts / {post_id} / comments
5. PUT / api / comments / {comment_id}
6. DELETE / api / comments /{post_id}/ {comment_id}
7. POST / api / users / {user_id} / follow
8. DELETE / api / users / {user_id} / unfollow

Core-entities of System

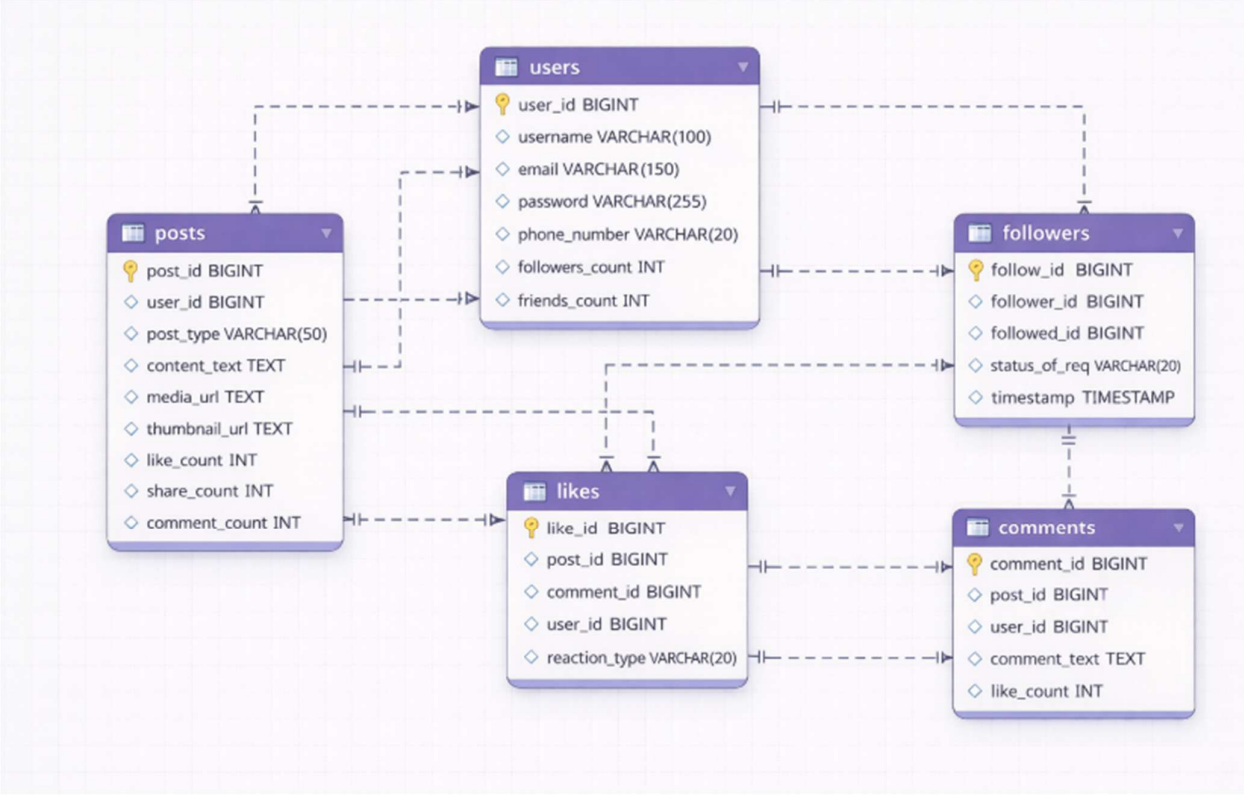
1. Users
2. Followers
3. Post
4. Feeds
5. Like & Comment

Outcome:

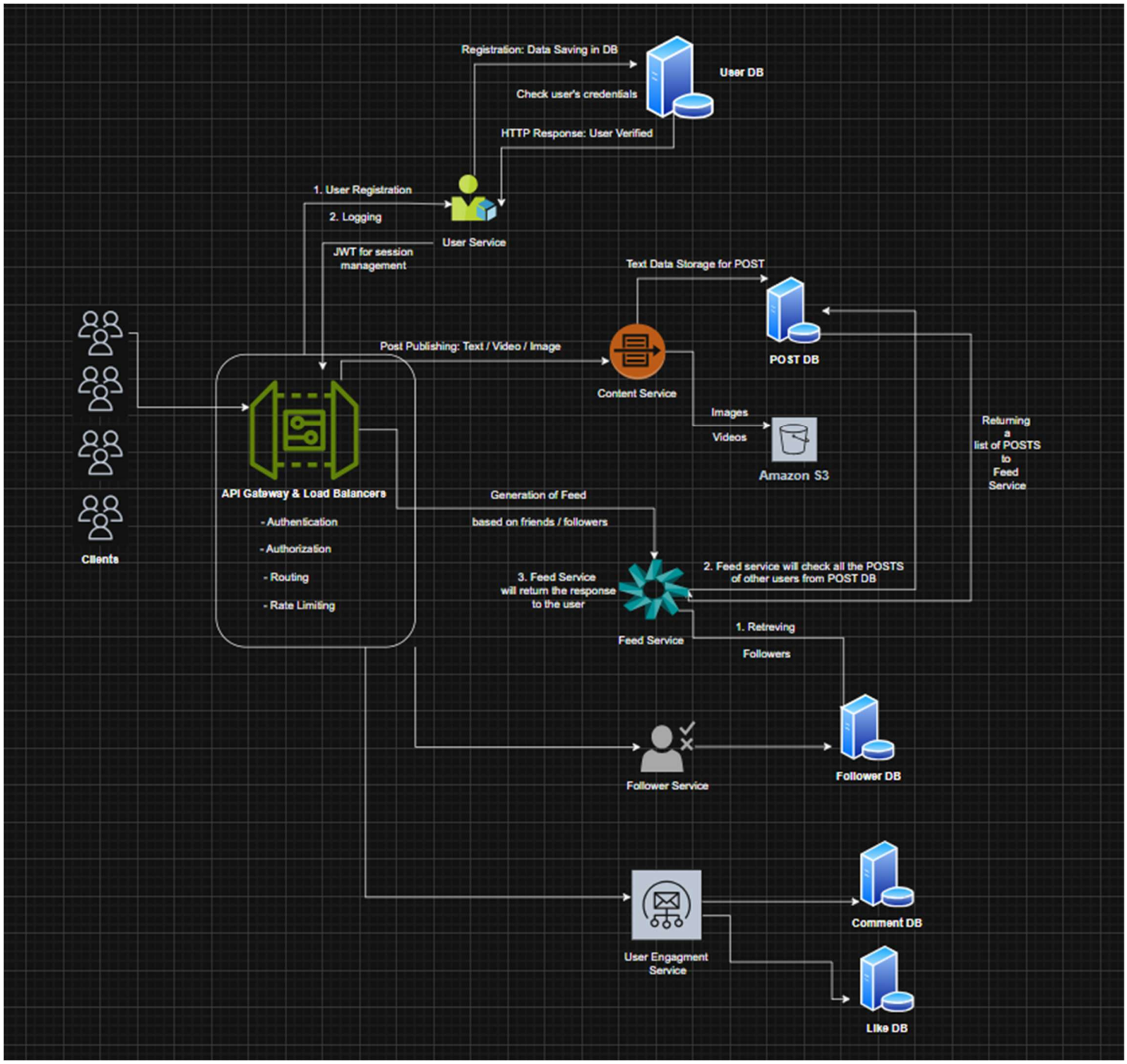
1. Successfully designed a high-level architecture for a social media platform.
2. Identified key functional requirements of the system.
3. Analyzed important non-functional requirements focusing on scalability, availability, and latency.
4. Defined core entities required for implementing the application.
5. Designed RESTful API endpoints to support user management, posts, and interactions.
6. Gained understanding of system design considerations for large-scale social media applications.

Required System Design:

LLD:



HLD:


Registration: Data Saving in DB
User DB
Check user's credentials
HTTP Response: User Verified
1. User Registration
2. Logging
User Service
JWT for session management
Text Data Storage for POST
POST DB
Post Publishing: Text / Video / Image
Content Service
Images
Videos
Amazon S3
Returning a list of POSTS to Feed Service
Clients
API Gateway & Load Balancers
- Authentication
- Authorization
- Routing
- Rate Limiting
Generation of Feed
based on friends / followers
3. Feed Service will return the response to the user
2. Feed service will check all the POSTS of other users from POST DB
Feed Service
1. Retreving
Followers
Follower Service
Follower DB
Comment DB
User Engagment Service
Like DB