Experiment 3

Student Name: Deepanshu Mandhyan **UID: 23BCS12327**
Branch: CSE **Section/Group: KRG 3-A**
Semester: 6th **Date of Performance:29/01/2026**
Subject Name: System Design **Subject Code: 23CSH-314**

1. Aim:

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

2. Objective:

- To understand social media application workflow.
- To design functional and non-functional requirements.
- To create system architecture (HLD).
- To design modules/classes (LLD).
- To implement core APIs for user authentication, posts, feed, likes, and comments.

3. Tools Used:

- **Python** – Backend logic implementation and URL generation algorithms.
- **Flask** – Lightweight web framework for developing RESTful APIs.
- **Draw.io** – Designing system architecture diagrams (HLD & LLD).

4. System Requirements:

A. Functional Requirements

- Client should be able to **register and login** to the application.
- Client should be able to **create posts** (text / image / videos).
- Client should be able to **follow each other** (or send friend requests).
- Client should be able to **like or comment** on posts.
- Client should be able to **view feed** of posts from users they follow.

B. Non – Functional Requirements

- **Scalability**
  The system should support up to 500 million Daily Active Users (500M DAU).
- **Availability and Consistency**
  Since this is a social media application, Availability is prioritized over Consistency.

  Reason:
  If the application is not operational when required, there is no purpose in developing. Minor delays in content propagation are acceptable compared to complete downtime.

  Example: If Instagram is down for 1 hour, it is a major issue.
  However, if a post takes 500 ms to reach followers while the system remains available, it is acceptable
  Hence:
  Availability >>> Consistency

- **Latency**
  Uploading or publishing a post should take approximately 500 milliseconds to ensure a smooth user experience.

7. High-Level Design (HLD):

The system follows a MICRO-SERVICE ARCHITECTURE / DISTRIBUTED:

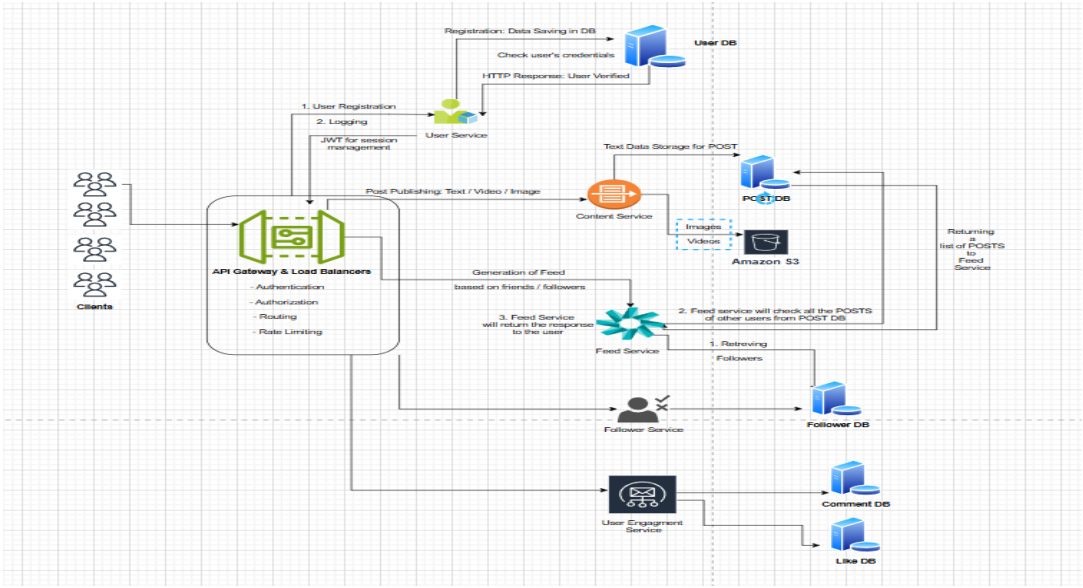

Fig. (b): HLD Diagram

8. Low- Level Design (LLD):

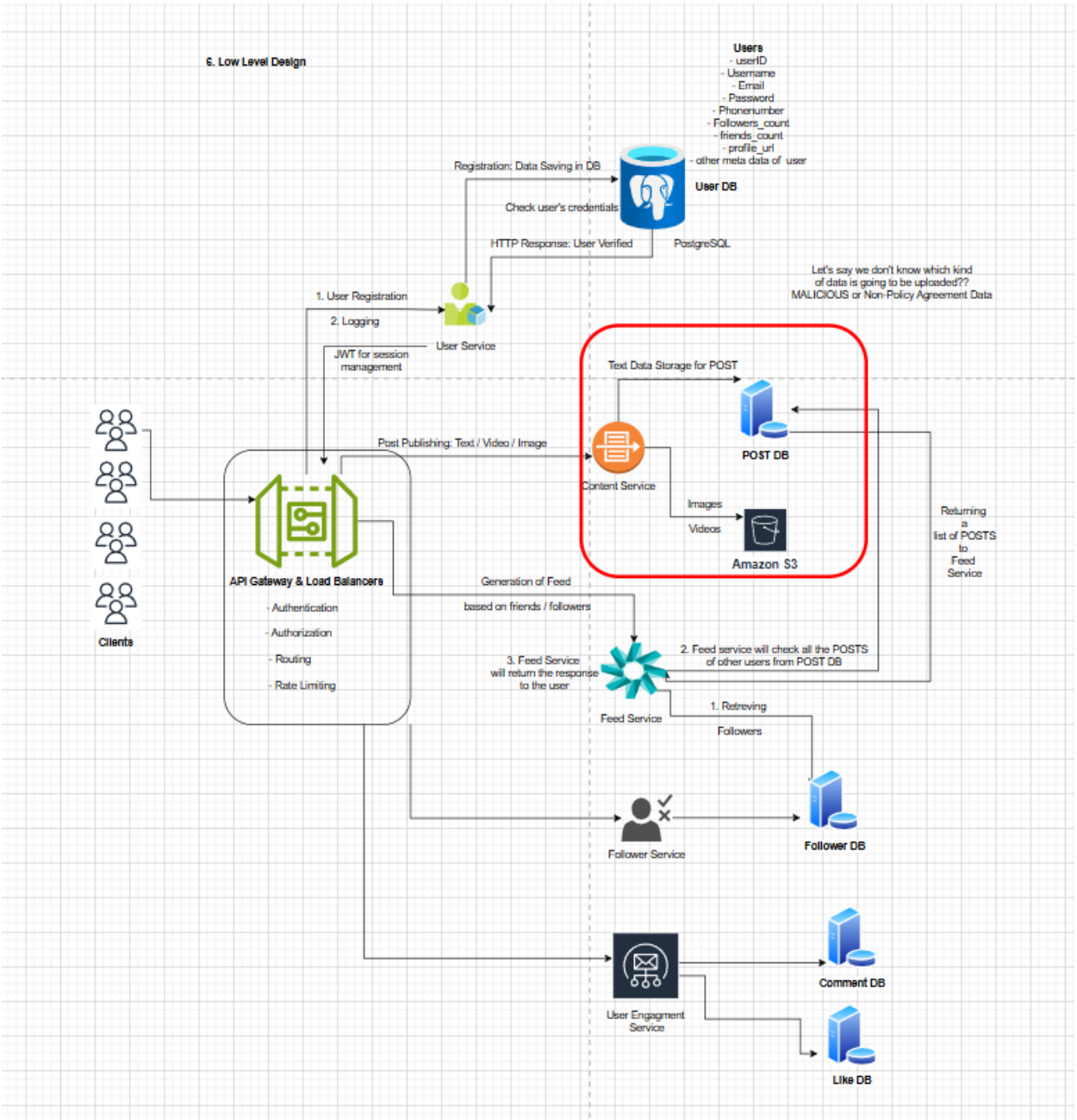

Fig. (c): LLD Diagram

9. Scalability Solution:

- Use horizontal scaling by adding multiple application servers behind a load balancer.
- Implement database sharding to distribute user and post data across multiple databases.
- Use caching (Redis/Memcached) to reduce database load for frequently accessed data like feeds and profiles.
- Store media files (images/videos) using CDN and cloud storage to improve performance and reduce server load.
- Apply asynchronous processing and message queues (Kafka/RabbitMQ) for tasks like notifications and feed updates.

10. Learning Outcomes:

- Understood the workflow of a social media application system.
- Learned to design functional and non-functional requirements.
- Gained knowledge of high-level and low-level system architecture.
- Learned how scalability and availability are achieved in large-scale systems.
- Developed understanding of core APIs for user management, posts, and feed handling.