



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Experiment - 3

**Student Name:** Mohammad Kaif

**UID:** 23BCS12884

**Branch:** BE-CSE

**Section/Group:** KRG\_2A

**Semester:** 6<sup>th</sup>

**Date of Performance:** 29/01/26

**Subject Name:** System Design

**Subject Code:** 23CSH-314

### **Aim:**

To design a **Social Media Platform** that allows users to register, login, create posts, follow other users, and interact with posts through likes and comments, ensuring high availability, scalability, and low latency.

### **Objectives:**

1. To understand the working of a Social Media system
2. To identify **functional requirements** of the system
3. To identify **non-functional requirements** such as performance and scalability
4. To design a high-level system flow using **draw.io**
5. To understand core entities involved in the platform

### **Procedure-**

1. Identify functional requirements of a social media platform.
2. Define non-functional requirements such as scalability, latency, and availability.
3. Analyze CAP theorem trade-offs for social media systems.
4. Identify core entities required for system implementation.
5. Design the system architecture using Draw.io.
6. Validate the design against real-world social media behavior.

### **Functional Requirements -**

1. Users should be able to **register and login** to the application.
2. Users should be able to **create posts** (text / image / video).
3. Users should be able to **follow other users** or send friend requests.
4. Users should be able to **like and comment** on posts.
5. Users should be able to **view a feed** consisting of posts from users they follow.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Non-functional Requirements

### A. Scalability

- System should support up to 500 million daily active users (DAU).

### B. Consistency and Availability

- The system prioritizes high availability over strong consistency.
- Temporary delays in post propagation are acceptable.

#### Justification:

If the application is unavailable during peak time, it leads to a major business loss.

Example:

If Instagram is down for 1 hour → **critical issue**

If a post reaches followers in 500 ms instead of instantly → **acceptable** Hence,

**Availability >>> Consistency**

### C. Latency

- Post upload and publish latency should be around **500 ms**.

## Outcome / Result -

A complete high-level design of a social media platform was successfully created, identifying its functional requirements, non-functional constraints, core entities, and feed management strategy.

The diagram illustrates a complex microservices architecture for a social media feed. It begins with **Clients** interacting with an **API Gateway & Load Balancers**, which handles authentication, authorization, routing, and rate limiting. The main flow involves several key services and data stores:

- User Service**: Manages user registration and login. It interacts with a **User DB** (PostgreSQL) for user data and a **POST DB** for post metadata. It also handles JWT for session management.
- Content Service**: Processes posts against policy, using **Notification Svc** and **Moderator Svc**. It interacts with a **KAFKA Producer** and **KAFKA Consumer**.
- Post Materializer**: Takes input from the **KAFKA Producer** and outputs to **Post against policy** and **Notification Svc**.
- Text Data Storage for POST**: Uses **Write Ops Fast** to store post data in **POST DB**.
- Amazon S3**: Stores images and videos associated with posts.
- Fanout Svc (PUSH)**: Pushes post data to **Fanout Consumer** for distribution to followers.
- Feed Cache**: A **Feed Service** interacts with a **Feed Cache** (Feed preparation in cache) and a **Feed DB** to serve the feed.
- BackFill**: A service used for reprocessing or updating the feed cache.

The diagram also includes several data lists and annotations:

- Users**: userID, Username, Email, Password, Phonenumber, Followers\_count, friends\_count, profile\_url, other meta data of user.
- Post (Document DB)**: post\_id, user\_id, post\_type, content\_text, media\_url, thumbnail\_url, like\_count, share\_count, comment\_count, other meta data.
- Annotations**: "1. User Registration", "2. Logging", "3. Get me all recent post based on my followers", "Check the post's from POST DB in prior & pass them to KAFKA queue.", "For a specific user storing all post's to show on feed.", "For normal person", "Will Check for the followers from follower DB", "Followers Cache (Top Followers)".

