



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment - 3

Student Name: Parv Bansal

UID: 23BCS13701

Branch: BE-CSE

Section/Group: KRG_1B

Semester: 6th

Date of Performance: 02/02/26

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 system architecture for a social media feed service, organized into several key components and their interactions:

- Clients:** Represented by icons of people, they interact with the system through the API Gateway.
- API Gateway & Load Balancers:** Acts as the first point of contact, handling:
 - Authentication
 - Authorization
 - Routing
 - Rate Limiting
- User Service:** Manages user-related operations:
 - 1. User Registration
 - 2. Logging
 - JWT for session management
- PostgresSQL (User DB):** Stores user data:
 - userID
 - Username
 - Email
 - Password
 - Phonenumber
 - Followers_count
 - friends_count
 - profile_url
 - other meta data of user
- Content Service:**
 - Post against policy
 - Notification SVC
 - Moderator SVC
- KAFKA Producer:**
 - blocked_post
 - filtered_post
 - raw_post
 - filtered_post
 - blocked_post
- Post Materializer:**
 - 1. Users Latest_post < recent 100 posts pre-computed >
 - 3. Get me all recent post based on my followers
- KAFKA:**
 - < userID, List < postID > >
 - < post, List < FriendsUserID > >
- Post (Document DB):**
 - post_id
 - user_id
 - post_type
 - content_text
 - media_url
 - thumbnail_url
 - like_count
 - share_count
 - comment_count
 - other meta data
- POST DB:** Text Data Storage for POST
- Write Ops Fast:** Utilizes POST DB and Amazon S3 for storage.
- Fanout Consumer (FANOUT Svc (PUSH)):**
 - For normal person
 - 3. write
- Feed Service:**
 - Read
 - Feed Cache (Feed preparation in cache)
 - Feed DB
 - 3. write
- Followers Cache (Top Followers):**
 - 1. Will Check for the followers from follower DB

