**DBMS Group Project**

**Group Members:** Keerat Singh (2023282), Harsh Sharma (2023233), Sanchay Singh (2023478), Ryan Bhan (2023452)

**Project Title:** *Twitter Clone*

**Overview:**The project is a simplified version of Twitter, a microblogging social networking platform. It will allow users to register, create posts (tweets), comment on posts, and interact with each other through likes and follows. The goal is to create a user-friendly interface combined with robust back-end services to manage user-generated content and social interactions in real-time.

**Stakeholders:**

* Users using the app
* Administrators ensure content guidelines are followed
* Developers ensure a bug-free environment.

### **Features and Functionalities to be Incorporated**

**Core Features:**

* **User Management:**
  + **Registration/Login:** Users can create an account or log in with an existing account using their registered email and password.
  + **User Profiles:** Users can customize profiles with basic information, profile pictures, and biodata.
  + **Ban:** Administrators have the ability to ban users who violate regulations.
* **Administration:**
  + **Login:** Logging in of administrators for regulating the content posted by users.
* **Post Creation and Management:**
  + **Tweet Posting:** Ability for users to create posts with text or images.
  + **Editing and Deleting Posts:** Options for users to modify or remove their own posts.
  + **Removal of Posts:** The ability of administrators to remove posts if they violate the rules.
* **Interaction Features:**
  + **Comments:** Users can comment on posts.
    - *Note:* Comments are modeled as a weak entity dependent on posts, with a relationship to both the post (the owner entity) and the commenting user.
  + **Likes and Shares:** Users can like and share posts, enhancing engagement and content visibility.
  + **Report:** Users can report posts or users that violate regulations so that admins can review them and act accordingly.
  + **Follow/Unfollow:** Users can follow or unfollow other users.
  + **Friend/Unfriend:** Users can be friends with other users or unfriend them.
  + **Private Messages:** Users have the ability to send personal messages to one another in one-on-one chats.
* **Notification System:**
  + Real-time notifications for new followers, posts, likes, shares, and comments.
* **Search and Explore:**
  + Functionality to search for users or topics.
  + An explore page to discover trending posts and topics.

**Additional Considerations:**

* **Security:**
  + Implementation of best practices for data security and user privacy.
* **Analytics:**
  + Basic analytics to monitor user activity and engagement metrics.

### **Why a Database is Needed**

**Data Persistence and Management:**

* **User Data:**A database is crucial for securely storing user information such as usernames, passwords (hashed), bios, profile pictures, etc. This allows for reliable authentication and personalized user experiences.
* **Content Management:**
  + **Posts, Comments, and Interactions:**Storing posts, comments, likes, and shares in a database ensures that all user-generated content is persistent, queryable, and updatable. For example, comments (modeled as a weak entity) will reference the corresponding post and user, ensuring relational integrity.
  + **Feed Generation:**Efficient querying of the database is essential for showing relevant posts and exploring trending topics.
* **Data Relationships and Integrity:**The relational database will help manage complex relationships (e.g., users with posts, posts with comments, and inter-user interactions), ensuring data integrity through foreign key constraints and proper normalization.
* **Analytics and Reporting:**A centralized database makes it easier to perform queries for generating reports and monitoring user likes, follows, etc.

**Implementation through ER Model**

**Entities created and their relations :**

**User Entity:**

* A **User** has attributes like UID, Email, Username, Password, DOB (Date of Birth), and Bio.
* A **User** can **Follow** another User.
* A **User** can have **Friends** (establish friendships with other users).
* A **User** can **Leave** a platform (possibly delete their account).
* A **User** can **Send Messages** to another user.
* A **User** can **Make a Post** (creating content).
* A **User** can **Delete a Post** they created.
* A **User** can **Like a Post**.
* A **User** receives **Notifications** triggered by various actions.

**Post Entity:**

* A **Post** has attributes like PID, UID (creator), Date, Views, and Content.
* A **Post** can be **Liked** and **Shared** by users.
* A **Post** can have **Comments**.

**Comments Entity (weak entity):**

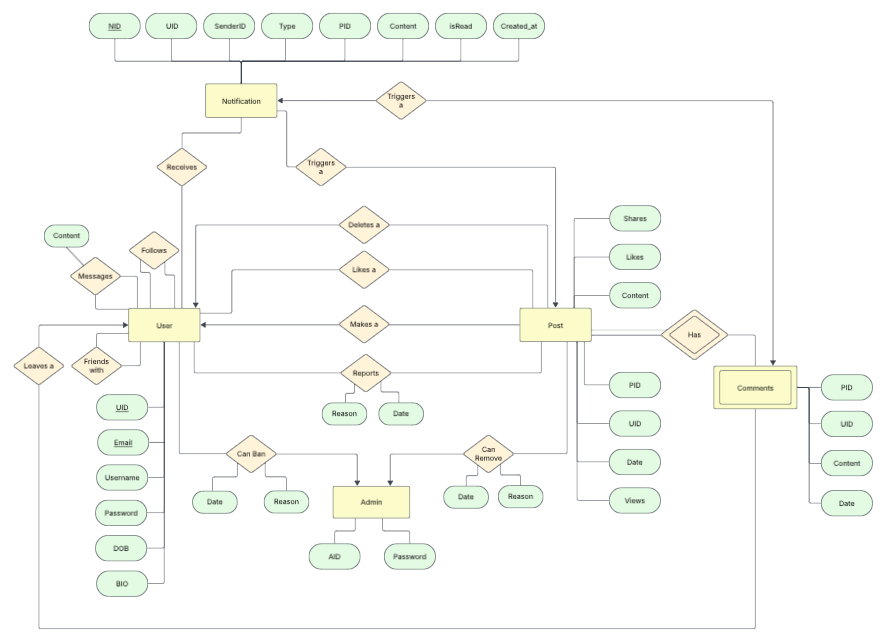
* A **Comment** belongs to a **Post** (relationship: **Has**).
* A **Comment** has attributes like PID (post it belongs to), UID (who commented), Content, and Date.

**Notification Entity:**

* A **Notification** has attributes like NID, UID (receiver), SenderID, Type, PID, Content, IsRead, and Created\_at.
* A **Notification** is **Triggered** by events like a user following another, liking a post, commenting, etc.
* A **User** **Receives** Notifications.

**Admin Entity:**

* An **Admin** has attributes like AID and Password.
* An **Admin** can **Ban a User** based on a **Reason and Date**.
* An **Admin** can **Remove a Post** based on a **Reason and Date**.
* A **User** can **Report a Post** to an **Admin**, providing a **Reason and Date**.



**Link:** <https://lucid.app/lucidchart/58f8eac4-947b-41fd-aa6f-753ca00f70d7/edit?viewport_loc=259%2C544%2C2523%2C1258%2C0_0&invitationId=inv_01275d8e-3a0c-4aff-ba78-9e1e05dfe936>

**Sources:**

* Format of Doc taken from Chatgpt

**Changes Made:**

* I made specific edits to the prompt, removing details for clarity, clarifying Twitter clone ER model alterations, and detailing the inclusion of "comment" relationships and the weak entity set "comments." This ensures precise guidance for the required ER model adjustments.
* Populated the format with our topic and implementation.