

| Business Template  **Social Media** |
| --- |
|  |

**Contents**

[1](#_sbta3poxh9qs) Business Description 3

[1.1](#_dz1b0yckbk84) Business background 3

[1.2](#_woqbror8tqun) Problems. Current Situation 3

[1.3](#_h60d9jcm17fn) The benefits of implementing a database. Project Vision 3

[2](#_fqysrjsk078i) Model description 3

[2.1](#_7incse24bz5v) Definitions & Acronyms 3

[2.2](#_w61w6knmdgun) Logical Scheme 3

[2.3](#_601z17ej10a4) Objects 3

# 

# Business Description

## Business background

The purpose of this database is to support a social media platform designed to deliver relevant and personalized content to users. To achieve this, the system must store and analyze essential user-related information such as personal details, geolocation, social connections, posts, and engagement activities (likes, comments, shares, hashtags). The platform aims to enhance user experience through content recommendations, interaction tracking, and trend insights.

## Problems. Current Situation

Without a properly designed database, user information across a social media platform becomes inconsistent and fragmented.  
 Typical issues include:

* User data stored in multiple places causing duplication and update conflicts
* Inability to track relationships between users (followers, friends) effectively
* Difficulty in analyzing engagement (likes, comments, shares) due to unstructured data
* Limited capability to perform content recommendations or trend analytics because hashtags and user actions are not normalized
* Poor performance when querying large volumes of user activity

These challenges result in low personalization, inefficient content delivery, and poor scalability.

## the Benefits of implementing a database. Project Vision

A well-structured, normalized database will ensure:

* **Consistent and clean user data** across all platform components
* **Efficient storage and retrieval** of user interactions and relationships
* **Accurate analytics** to support tailored content recommendations
* **Scalable architecture** that can grow as the user base expands
* **Data integrity and transparency**, enabling better decision-making and reporting

The vision is to build a secure, normalized, scalable database model that fully supports user interaction features, engagement analytics, and content personalization.

# Model description

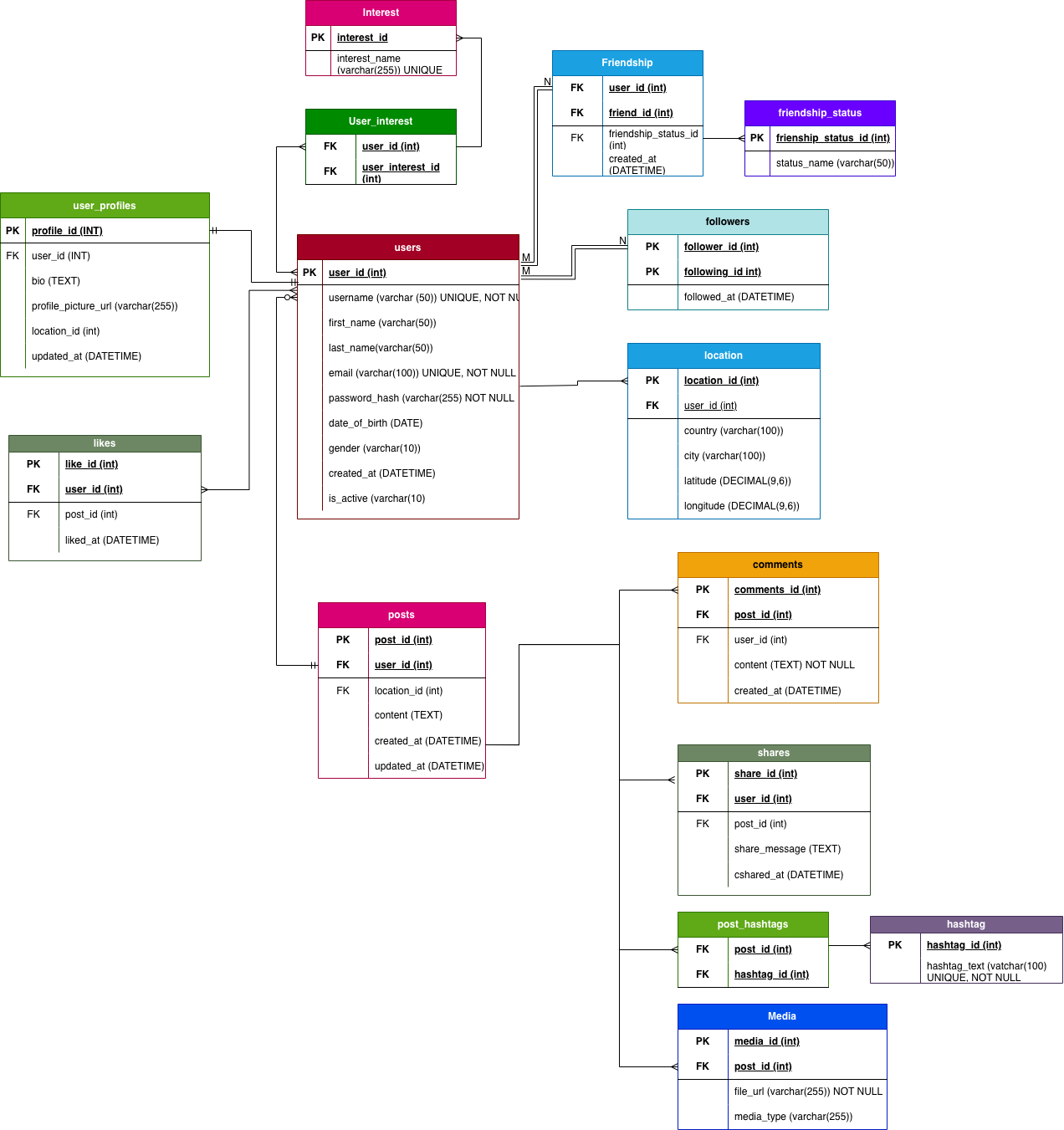
## Definitions & Acronyms

| **Term/Acronym** | **Definition** |
| --- | --- |
| User | A platform member with a registered account |
| Post | Content created and published by a user |
| Engagement | User interaction with a post (Like, Comment, Share) |
| M:N | Many-to-many relationship |
| 3NF | Third Normal Form – normalization standard removing data redundancy |
| Self-referential relationship | A relationship where a table references itself (e.g., follow feature) |

## Logical Scheme

The logical data model includes 12 tables designed in 3rd Normal Form (3NF).  
 It incorporates:

* **1 self-referential M:N relationship**: User follows User
* **1 many-to-many relationship**: Post uses multiple hashtags; hashtags appear in many posts
* Proper keys, constraints, and data types applied to ensure integrity



## Objects

Table Description

Below is a short description of key objects in the model:

| **Object (Table)** | **Description** |
| --- | --- |
| User | Stores base user details |
| User\_Profile | Stores extended profile attributes to avoid data duplication |
| Location | Holds standardized geographical data |
| Post | Contains user-generated content with timestamps |
| Media | Stores media files linked to posts |
| Hashtag | Stores individual hashtags |
| Post\_Hashtag | Resolves M:N relationship between posts and hashtags |
| Comment | Tracks comments made on posts |
| Like | Tracks likes made by users on posts |
| Share | Tracks sharing of posts |
| Follow | Self-referential table showing follower–followed relationships |
|  |  |

| Table Name | Field name | Field Description | Data Type |
| --- | --- | --- | --- |
| User | user\_id | Unique identifier for each user, **PK** | INT |
| username |  |  |
| first\_name | User’s first name | VARCHAR(50) |
| last\_name | User’s last name | VARCHAR(50) |
| email | User email (unique) | VARCHAR(100) |
| date\_of\_birth | User birthdate | DATE |
| gender | Gender of the user | VARCHAR(10) |
| password\_hash |  | VARCHAR(255) |
| created\_at | Account creation timestamp | DATETIME |
| User\_Profile | profile\_id | Unique profile identifier, **PK** | INT |
| user\_id | References user, **FK → User(user\_id)** | INT |
| bio | Short bio | VARCHAR(255) |
| profile\_picture\_url | Profile picture URL | VARCHAR(255) |
| location\_id | User current location, FK**→ Location(location\_id)** | INT |
| Location | location\_id | Unique location ID, **PK** | INT |
| user\_id | References user, **FK → User(user\_id)** | INT |
| country | Country | VARCHAR(100) |
| city | City | VARCHAR(100) |
| latitude | Latitude geo coordinate | DECIMAL(9,6) |
| longitude | Longitude geo coordinate | DECIMAL(9,6) |
| Followers | follower\_id | User who follows, **FK → User(user\_id)** | INT |
| following\_id | User being followed, **FK → User(user\_id)** | INT |
| followed\_at | Timestamp when follow happened | DATETIME |
| Post | post\_id | Unique post identifier, **PK** | INT |
| user\_id | Post owner, **FK → User(user\_id)** | INT |
| content | Post text/content | TEXT |
| created\_at | Post creation timestamp | DATETIME |
| updated\_at | Post updation timestamp | DATETIME |
| location\_id | Post location, FK**→ Location(location\_id)** | INT |
| Hashtag | hashtag\_id | Unique hashtag ID, **PK** | INT |
| name | Hashtag text (no #), unique | VARCHAR(50) |
| Post\_Hashtag | post\_id | References post, **FK → Post(post\_id)** | INT |
| hashtag\_id | References hashtag, **FK → Hashtag(hashtag\_id)** | INT |
| Like | like\_id | Like ID, **PK** | INT |
| post\_id | Liked post, **FK → Post(post\_id)** | INT |
| user\_id | User who liked, **FK → User(user\_id)** | INT |
| liked\_at | Timestamp of like | DATETIME |
| Comment | comment\_id | Comment ID, **PK** | INT |
| post\_id | Commented post, **FK → Post(post\_id)** | INT |
| user\_id | User who commented, **FK → User(user\_id)** | INT |
| content | Comment text | VARCHAR(255) |
| created\_at | Timestamp of comment | DATETIME |
| Share | share\_id | Share ID, **PK** | INT |
| post\_id | Shared post, **FK → Post(post\_id)** | INT |
| user\_id | User who shared, **FK → User(user\_id)** | INT |
| share\_message | Message sent with shared post | TEXT |
| shared\_at | Timestamp of share | DATETIME |
| Media | media\_id | Media ID, PK | INT |
| post\_id | Shared post, **FK → Post(post\_id)** | INT |
| file\_url | Direct link to the stored media file (e.g., image or video). | VARCHAR(255) |
| media\_type | Type of media file, such as image, video, or audio. | VARCHAR(255) |
| Friendship | user\_id | ID of the user | INT |
| friend\_id | ID of the friend (also a user) | INT |
| friendship\_status\_id | Status of the friendship | INT |
| created\_at | Date when the friendship was created | DATETIME |
| Friendship\_status | friendship\_status\_id | Unique status ID | INT |
|  | status\_name | Status name (e.g., pending, accepted, blocked) | VARCHAR(50) |
| Interest | user\_interest\_id | Unique interest ID | INT |
|  | interest\_name | Name of an interest (e.g., “Sports”, “Music”) | VARCHAR(255) |
| user\_interest | user\_id | Unique ID for record | INT |
|  | user\_interest\_id | ID of the interest | INT |

## Relationships Between Entities

* **User – User\_Profile (1:1):** Each user has one profile. User\_Profile.user\_id references User.user\_id.
* **User – Location (1:N):** A user may have multiple saved locations. Location.user\_id references User.user\_id.
* **User – Followers – User (M:N, self-referential):** Users can follow each other. The Followers table links follower and following users, both referencing User(user\_id).
* **User – Post (1:N):** One user can create multiple posts. Post.user\_id references User.user\_id.
* **Post – Location (1:N):** A post may include a location. Post.location\_id references Location.location\_id.
* **Post – Hashtag (M:N via Post\_Hashtag):** Posts may include multiple hashtags, and hashtags may belong to multiple posts. The Post\_Hashtag table connects Post and Hashtag.
* **User – Like – Post (M:N):** Users can like multiple posts, and posts can have many likes. Like.user\_id references User.user\_id, and Like.post\_id references Post.post\_id.
* **User – Comment – Post (M:N):** Users can comment on posts, and posts can have multiple comments. Comment links a user to a post with additional attributes.
* **User – Share – Post (M:N):** Users can share posts, and posts can be shared by multiple users. Share links user and post with a share message and timestamp.
* **Post – Media (1:N):** A post may contain multiple media files. Media.post\_id references Post.post\_id.
* **User – Friendship – User (M:N, self-referential):** Friendship between users is stored in the Friendship table with status. It contains user\_id and friend\_id, both referencing User(user\_id).
* **Friendship\_status – Friendship (1:N):** One status type can apply to multiple friendships. Friendship.friendship\_status\_id references Friendship\_status.friendship\_status\_id.

## Example with data

| user\_id | username | first\_name | last\_name | email | date\_of\_birth | gender | password\_hash | created\_at |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | anna\_k | Anna | Kim | anna.kim@example.com | 1996-04-12 | Female | h$92jd91msk2 | 2024-01-14 09:23:11 |
| 2 | max\_travel | Max | Turner | max.turner@example.com | 1993-11-02 | Male | 7dhA82mslQW | 2024-02-03 13:47:55 |
| 3 | sofia.creative | Sofia | Mendes | sofia.mendes@example.com | 1999-07-25 | Female | k2L9dk20As8 | 2024-02-19 16:10:05 |