Schema for the Social Interaction App

Schema for the Social Interaction System database utilizes several tables to represent the various entities and their relationships within the system. Here's a breakdown of each table and its attributes:

Users:

- This table stores information about individual users registered on the platform.
- It includes essential details like usernames, emails (used for login and communication), and hashed passwords for security.
- Optional fields like bio, location, and interests allow users to personalize their profiles.
- Birthdate, profile picture path, timestamps for creation and modification, and other relevant details are also captured.

Friends:

- This table establishes connections between users who have friended each other.
- It uses foreign keys to reference user IDs, ensuring both users are properly linked.
- A composite primary key consisting of both friend IDs guarantees unique friend connections, preventing duplicates.

Posts:

- This table stores user-generated content shared on the platform.
- It captures the post's content, type (text, photo, or video), and associated media URL if applicable.
- User ID (through a foreign key) identifies the post author.
- A timestamp records the posting time, and privacy settings control who can view the post (public, friends only, or private).

Comments:

- This table stores comments posted on individual user posts.
- Foreign keys link comments to their corresponding posts and the user who authored them.
- The comment content and timestamp of creation are stored.

Likes:

- This table records user actions of "liking" specific posts.
- Foreign keys connect likes to the post being liked and the user who liked it.
- A timestamp captures the time of the like action.

Requests:

- This table stores requests made by users to other users.
- It captures who made the request, to whom it was made, and when it was created.
- Foreign keys ensure referential integrity with the Users table for both the user making the request and the user receiving it.

The detailed table attributes are as follows

Users:

- user_id (INT PRIMARY KEY AUTO_INCREMENT): Unique identifier for each user.
- username (VARCHAR(50) UNIQUE): User's chosen username for login and display purposes (enforced uniqueness).
- email (VARCHAR(255) UNIQUE): User's email address for registration, login, and communication (enforced uniqueness).
- password (VARCHAR(255)): Hashed and salted password for secure user authentication.
- first_name (VARCHAR(50)): User's first name.
- last_name (VARCHAR(50)): User's last name.
- bio (TEXT): Optional short description or biography about the user.
- location (VARCHAR(100)): User's location (city, state, country).
- occupation (VARCHAR(50)): User's occupation or job title.
- interests (TEXT): Comma-separated list of user's interests.
- birthdate (DATE): User's date of birth.
- profile_picture (VARCHAR(255)): Optional URL path to the user's profile picture.
- created_at (DATETIME DEFAULT CURRENT_TIMESTAMP): Timestamp of user account creation.
- updated_at (DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP): Timestamp of user account modification.

Friends:

- friend_id_1 (INT FOREIGN KEY REFERENCES Users(user_id)): User ID of the first friend (enforced foreign key relationship).
- friend_id_2 (INT FOREIGN KEY REFERENCES Users(user_id)): User ID of the second friend (enforced foreign key relationship).

- created_at (DATETIME DEFAULT CURRENT_TIMESTAMP): Timestamp of friend connection creation.
- PRIMARY KEY (friend_id_1, friend_id_2): Composite primary key to ensure unique friend connections (avoids duplicate entries)

Posts:

- post_id (INT PRIMARY KEY AUTO_INCREMENT): Unique identifier for each post.
- user_id (INT FOREIGN KEY REFERENCES Users(user_id)): User ID of the post author (enforced foreign key relationship).
- content (TEXT): Actual text content of the post (supports text formatting like Markdown if desired).
- post_type (ENUM('text', 'photo', 'video')): Enumerated type defining the post type (text, photo, or video).
- media_url (VARCHAR(255)): Optional URL path to the post's media (image or video) if applicable.
- timestamp (DATETIME DEFAULT CURRENT_TIMESTAMP): Timestamp of post creation.
- privacy (ENUM('public', 'friends', 'private')): Enumerated type defining the post's privacy setting (public, visible only to friends, or private).

Comments:

- comment_id (INT PRIMARY KEY AUTO_INCREMENT): Unique identifier for each comment.
- post_id (INT FOREIGN KEY REFERENCES Posts(post_id)): Post ID that the comment belongs to (enforced foreign key relationship).
- user_id (INT FOREIGN KEY REFERENCES Users(user_id)): User ID of the comment author (enforced foreign key relationship).
- content (TEXT): Text content of the comment.
- timestamp (DATETIME DEFAULT CURRENT_TIMESTAMP): Timestamp of comment creation.

Likes:

- like_id (INT PRIMARY KEY AUTO_INCREMENT): Unique identifier for each like.
- post_id (INT FOREIGN KEY REFERENCES Posts(post_id)): Post ID that the like is associated with (enforced foreign key relationship).
- user_id (INT FOREIGN KEY REFERENCES Users(user_id)): User ID of the user who liked the post (enforced foreign key relationship).
- timestamp (DATETIME DEFAULT CURRENT_TIMESTAMP): Timestamp of the like action.

Messages:

- message_id (INT PRIMARY KEY AUTO_INCREMENT): Unique identifier for each message.
- sender_id (INT FOREIGN KEY REFERENCES Users(user_id)): User ID of the message sender (enforced foreign key relationship).
- receiver_id (INT FOREIGN KEY REFERENCES Users(user_id)): User ID of the message recipient (enforced foreign key relationship).
- content (TEXT): Text content of the message.

Requests:

- request_id (INT PRIMARY KEY IDENTITY(1,1)): Unique identifier for each request.
- user_id (INT FOREIGN KEY REFERENCES Users(user_id)): ID of the user making the request.
- requester_id (INT FOREIGN KEY REFERENCES Users(user_id)): ID of the user receiving the request.
- created_at (DATETIME DEFAULT CURRENT_TIMESTAMP): Timestamp indicating when the request was created.

This is a basic schema of our App. Further additions or deletions can be made.