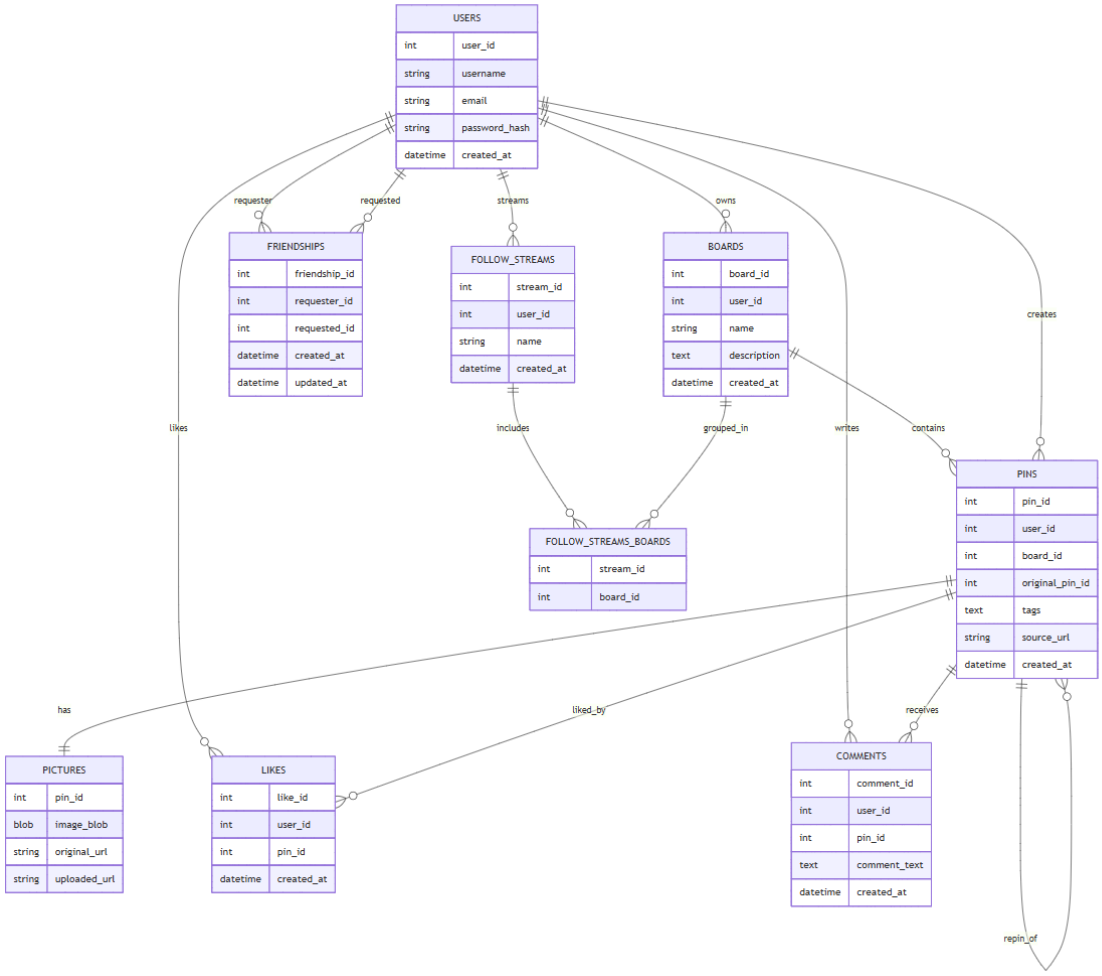
DB-Project-Part 1

ER diagram：(先用着，但建议换个新的)



Relational schema (keys & constraints) with database system：

Primary Key, *Foreign Key*

Users: user_id, username, email, password_hash, created_at;

Boards: board_id, *user_id(Users(user_id))*, name, description, created_at;

Pins: pin_id, *user_id(Users(user_id))*, *board_id(Boards(board_id))*, *original_pin_id(Pins(pin_id))*, tags, source_url, created_at;

Pictures: *pin_id(Pins(pin_id))*, image_blob, original_url, uploaded_url;

Friendships: friendship_id, *requester_id(Users(user_id))*, *requested_id(Users(user_id))*, created_at, updated_at;

FollowStreams: stream_id, *user_id(Users(user_id))*, name, created_at;

FollowStreamsBoards: *stream_id(FollowStreams(stream_id)), board_id(Boards(board_id))*;

Like: like_id, *user_id(Users(user_id)), pin_id(Pins(pin_id)),* created_at;

Comments: comment_id, *user_id(Users(user_id)), pin_id(Pins(pin_id))*, comment_text, created_at;

```sql
-- Users Table
CREATE TABLE Users (
    user_id SERIAL PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Boards Table
CREATE TABLE Boards (
    board_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE
);

-- Pins Table
CREATE TABLE Pins (
    pin_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    board_id INT NOT NULL,
    original_pin_id INT,
    tags TEXT,
    source_url VARCHAR(255),
```

```
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (board_id) REFERENCES Boards(board_id) ON DELETE CASCADE,
    FOREIGN KEY (original_pin_id) REFERENCES Pins(pin_id) ON DELETE CASCADE
);

-- Pictures Table
CREATE TABLE Pictures (
    pin_id INT PRIMARY KEY,
    image_blob BYTEA NOT NULL,
    original_url VARCHAR(255),
    uploaded_url VARCHAR(255),
    FOREIGN KEY (pin_id) REFERENCES Pins(pin_id) ON DELETE CASCADE
);

-- Friendships Table
CREATE TABLE Friendships (
    friendship_id SERIAL PRIMARY KEY,
    requester_id INT NOT NULL,
    requested_id INT NOT NULL,
    status VARCHAR(10) NOT NULL CHECK (status IN ('pending', 'accepted', 'd
eclined')),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    UNIQUE (requester_id, requested_id),
    FOREIGN KEY (requester_id) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (requested_id) REFERENCES Users(user_id) ON DELETE CASCADE
);

-- FollowStreams Table
CREATE TABLE FollowStreams (
    stream_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    name VARCHAR(100) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE
);

-- FollowStreamBoards Table
CREATE TABLE FollowStreamBoards (
    stream_id INT NOT NULL,
    board_id INT NOT NULL,
    PRIMARY KEY (stream_id, board_id),
    FOREIGN KEY (stream_id) REFERENCES FollowStreams(stream_id) ON DELETE C
ASCADE,
    FOREIGN KEY (board_id) REFERENCES Boards(board_id) ON DELETE CASCADE
);

-- Likes Table
CREATE TABLE Likes (
    like_id SERIAL PRIMARY KEY,
```

```
    user_id INT NOT NULL,
    pin_id INT NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    UNIQUE (user_id, pin_id),
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (pin_id) REFERENCES Pins(pin_id) ON DELETE CASCADE
);

-- Comments Table
CREATE TABLE Comments (
    comment_id SERIAL PRIMARY KEY,
    user_id INT NOT NULL,
    pin_id INT NOT NULL,
    comment_text TEXT NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (pin_id) REFERENCES Pins(pin_id) ON DELETE CASCADE
);
```

Pins and Repins: Repins reference the original pin_id. Deleting the original will delete all repins followed by it. Pin's tags stored as a comma-separated string in Pins for simplicity, enabling keyword search via LIKE.

Image Storage: Each image is stored once per original pin. Repins reference the same image blob. Follow streams are private, but boards/pins are public.