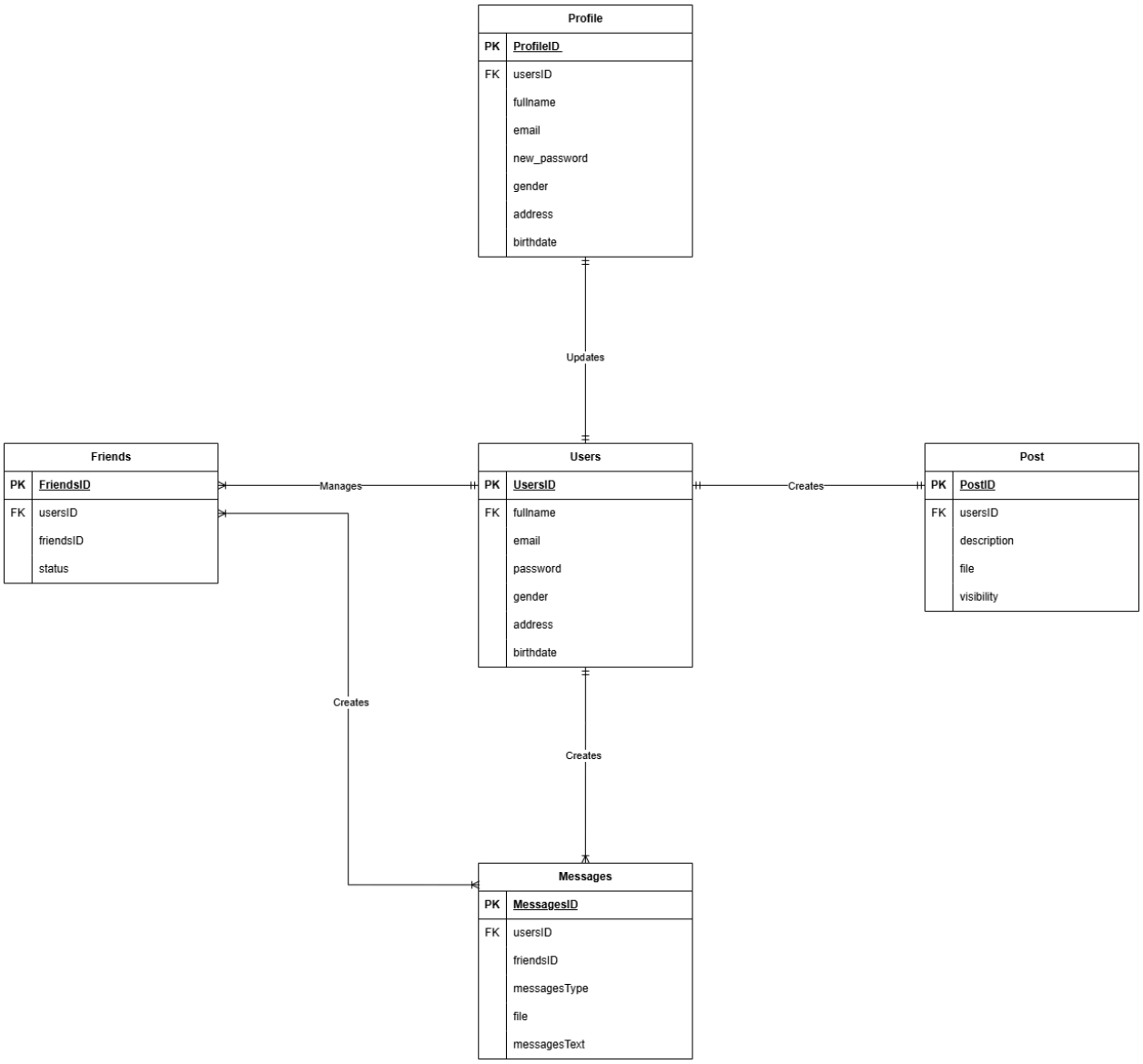
Database schema and sample data



Users

```
CREATE TABLE Users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    full_name VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    gender ENUM('Male', 'Female', 'Other') NOT NULL,
    address TEXT NOT NULL,
    birth_date DATE NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Posts_Table

```
CREATE TABLE Posts (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    description TEXT,
    file VARCHAR(255),
    visibility ENUM('Public', 'Private', 'Friends') DEFAULT 'Friends',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(id) ON DELETE CASCADE
);
```

Friends_Table

```
CREATE TABLE Friends (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    friend_id INT,
    status ENUM('Pending', 'Accepted', 'Declined') DEFAULT 'Pending',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(id) ON DELETE CASCADE,
    FOREIGN KEY (friend_id) REFERENCES Users(id) ON DELETE CASCADE
);
```

Messages_Table

```
CREATE TABLE Messages (
    id INT AUTO_INCREMENT PRIMARY KEY,
    sender_id INT,
    receiver_id INT,
    message_type ENUM('Text', 'Image', 'Video') NOT NULL,
    file VARCHAR(255),
    message_text TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (sender_id) REFERENCES Users(id) ON DELETE CASCADE,
    FOREIGN KEY (receiver_id) REFERENCES Users(id) ON DELETE CASCADE
);
```

Profile_Updates

```
CREATE TABLE Profile_Updates (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    full_name VARCHAR(255),
    email VARCHAR(255),
    new_password VARCHAR(255),
    gender ENUM('Male', 'Female'),
    address TEXT,
    birth_date DATE,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(id) ON DELETE CASCADE
);
```

Social Media API & Features

This system can handle a social media network through which a user can come, register, post contents, add new friends and remove friends , and send messages to other friends, with all of these actions facilitated through APIs.

As the user creates an account, his name, email address, and password are sent through an API into the system to further process and generate his profile. When a user makes a post, the content with any media that he is uploading will get sent through an API to the system to store and display them in his profile based on his privacy settings which may be set to 'Public', 'Private', or 'Friends'.

Requests to be friends work similarly, as well. When a user requests to be someone's friend, the system interacts with both users through an API updating the status of the request, which could be 'Pending' or 'Accepted'.

It continuously sends the message, whether in text, image, or video format, making sure it is delivered to the receiver and stored in the system for future reference. And when a user wants to update his/her details, for example, to change names or passwords, the API ensures that updation occurs safely and securely in the system.

In conclusion, APIs are intermediaries that allow users to interact with the system in a smooth and secure way, ensuring that all actions such as creating accounts, posting content, sending messages, and updating profiles happen seamlessly.