

# Завдання 3. Розробка архітектури проекту та DB

## 1. Users (Користувачі)

- id (PK, INT, AUTO\_INCREMENT)
- name (VARCHAR) — ім'я
- email (VARCHAR, UNIQUE)
- password (VARCHAR, hashed+salt) — пароль
- about (TEXT, optional) — про\_себе
- role (ENUM: user / admin) — користувач / адміністратор
- phone (VARCHAR) - номер телефону
- region (VARCHAR) — область
- district (VARCHAR) — район
- city (VARCHAR) — місто
- photo\_url (TEXT) — фото профілю

Логіка: тут зберігаються акаунти всіх людей, які працюють із системою.

## 2. Genres (Жанри)

- id (PK, INT)
- name (VARCHAR) — назва

Наприклад: фантастика, детектив, роман тощо.

## 3. BooksGenres (КнигиЖанри — зв'язок N:M)

- post\_id (FK → Books.id)
- genre\_id (FK → Genres.id)

Так можна призначити кілька жанрів одній книзі.

## 4. Posts (Оголошення)

- id (PK, INT)
- user\_id (FK → Users.id)
- title (VARCHAR)
- author (VARCHAR)
- deal\_type (ENUM: exchange / donation) — обмін / дарування
- description (TEXT) — опис
- photo\_url (VARCHAR)

Це ключова таблиця — тут публікуються оголошення.

## 5. Complaints (Скарги)

- id (PK, INT)
- text (TEXT) – текст
- date (DATETIME) – дата
- post\_id (FK → Posts.id)
- complainant\_id (FK → Users.id) – id людини, яка подає скаргу

Адміністратор може їх розглядати.

### Зв'язки між таблицями:

#### 1. Users ↔ Posts (Один-до-багатьох)

- **Один** користувач може створити **багато** оголошень
- **Одне** оголошення належить **одному** користувачу

#### 2. Posts ↔ Complaints (Один-до-багатьох)

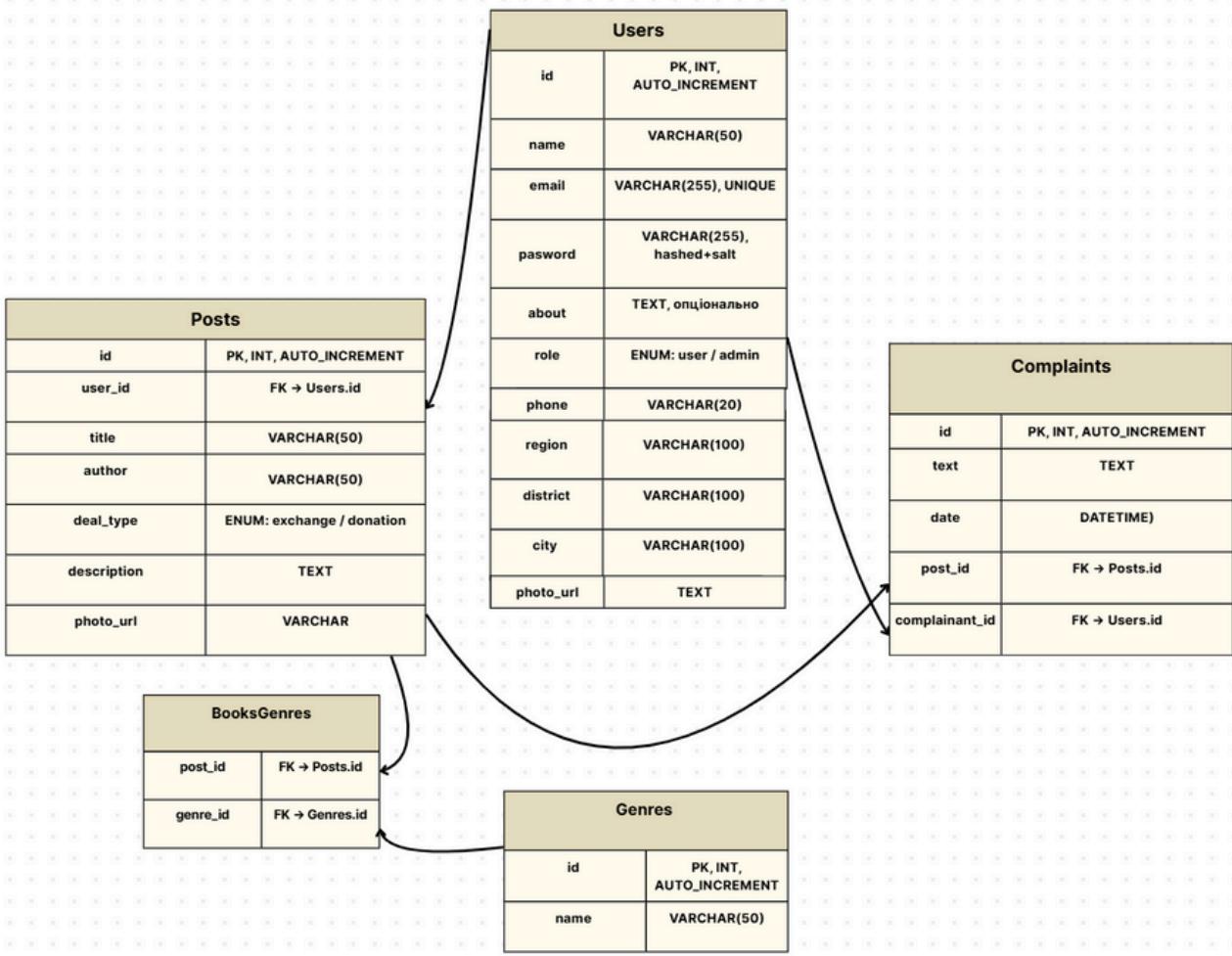
- **Одне** оголошення може мати **багато** скарг
- **Одна** скарга стосується **одного** оголошення

#### 3. Users ↔ Complaints (Один-до-багатьох)

- **Один** користувач може подати **багато** скарг (як скаржник)
- **Одна** скарга подається **одним** користувачем

#### 4. Posts ↔ Genres (Багато-до-багатьох)

- **Одне** оголошення книги може мати **кілька** жанрів
- **Один** жанр може належати **багатьом** оголошенні.
- Зв'язок реалізується через **проміжну таблицю BooksGenres**



Провайдер баз данных: PostgreSQL, Хостинг: Supabase