МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВЕТЕРИНАРНОЇ МЕДИЦИНИ ТА БІОТЕХНОЛОГІЙ ІМЕНІ С.З. ҐЖИЦЬКОГО

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

перший (бакалаврський) рівень вищої освіти

на тему:

ПРОЄКТУВАННЯ КОНЦЕПТУАЛЬНОЇ БАЗИ ДАНИХ ДЛЯ МОБІЛЬНОГО ЗАСТОСУНКУ «КИШЕНЬКОВА ГАЛЕРЕЯ»

Виконав: студент групи КН-41

спеціальності 122 «Комп'ютерні науки»

Дерпак О.І.

(прізвище та ініціали)

Керівник: к.е.н., доцент,

Смолінський В.Б.

(прізвище та ініціали)

ДУБЛЯНИ 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ПРИРОДОКОРИСТУВАННЯ

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Рівень вищої освіти перший (бакалаврський)
Спеціальність 122 «Комп'ютерні науки»

"ЗАТВЕРДЖУЮ"
Завідувач кафедри_____
д.т.н., проф. А.М. Тригуба
"____"_____202__ р.

ЗАВДАННЯ

на кваліфікаційну роботу студенту

_____Дерпаку Оресту Ігоровичу ____

1. Тема роботи: Проектування концептуальної бази даних для мобільного застосунку «Кишенькова галерея»

Керівник роботи: к.е.н., доцент, Смолінський В.Б

Затверджені наказом по університету *від 25. 02.2025 року № 123/к-с*

2. Строк подання студентом роботи *16.06.2025 р.*

3. Вихідні дані для роботи: характеристика предметної сфери; вихідні дані та вимоги до роботи програмного застосунку, опис використаних технологій та баз даних, науково-технічна і довідкова література

4. Зміст розрахунково-пояснювальної записки: (перелік питань, які потрібно розробити)

Вступ
1. *Аналіз предметної області*
2. *Математичний опис задачі та вибір засобів для реалізації проєкту*
3. *Проєктування та реалізація компонентів концептуальної бази даних*
4. *Охорона праці та безпека в надзвичайних ситуаціях*

Висновки
Список використаних джерел
Додатки

5. Перелік графічного матеріалу: *графічний матеріал подається у вигляді презентації, рисунки, таблиці*

6. Консультанти з розділів:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 2, 3	Смолінський В.Б., доцент кафедри інформаційних технологій		
4	Городецький І.М., доцент кафедри інженерної механіки		

7. Дата видачі завдання *25. 02. 2024 р.*

КАЛЕНДАРПНЙ ПЛАП

№ з/п	Етапи виконання кваліфікаційної роботи	Строк виконання етапів роботи	Відмітка про виконання
1.	Написання першого розділу та означення головних завдань роботи	25.02.2025 – 05.03.2025	
2.	Виконання другого розділу та формування головних алгоритмів	06.03.2025 – 29.03.2025	
3.	Виконання третього розділу, проведення розрахунків та розробка проектної частини	30.03.2025 – 27.04.2025	
4.	Написання розділу: «Охорона праці та безпека в надзвичайних ситуаціях»	28.04.2025 – 15.05.2025	
5.	Завершення оформлення розрахунково-пояснювальної записки та презентаційного матеріалу	16.05.2025 – 28.05.2025	
6.	Завершення роботи в цілому. Підготовка до захисту та кінцевої презентації роботи	01.06.2025 – 16. 06.2025	

Студент _____ Дерпак О.І
(підпис)

Керівник роботи _____ Смолінський В.Б.
(підпис)

УДК 004.652.1

Проєктування концептуальної бази даних для мобільного застосунку «Кишенькова галерея». Дерпак О.І. Кафедра інформаційних технологій. Дубляни, Львівський національний університет ветеринарної медицини та біотехнологій імені С.З. Ґжицького, 2025.

Кваліфікаційна робота: 46 с. текст. част., 10 рис., 4 табл., 15 слайдів, 25 джерел, 3 додатки.

У роботі розглянуто процеси проєктування концептуальної бази даних для мобільного застосунку «Кишенькова галерея», призначеного для збереження, структурування та перегляду графічних зображень. Виконано аналіз предметної області, вивчено особливості зберігання графічних даних у мобільних системах, визначено функціональні вимоги до системи.

Сформовано логічну структуру бази даних, обрано оптимальну систему керування базами даних, а саме SQLite, яка забезпечує ефективну локальну роботу на мобільних пристроях. Побудовано концептуальну модель, визначено основні сутності та зв’язки між ними. Реалізовано SQL-скрипти для базових операцій з даними: додавання, оновлення, пошук за тегами та видалення зображень.

У межах практичного розділу протестовано функціональність запитів та показано фрагменти взаємодії з базою даних через графічний інтерфейс.

Надано рекомендації щодо подальшого вдосконалення, зокрема щодо інтеграції з хмарними сервісами, авторизації користувачів та розширення функціоналу застосунку.

Ключові слова: база даних, мобільний застосунок, SQLite, DBeaver, Flutter, візуальне проєктування, інформаційна система.

Key words: database, mobile application, SQLite, DBeaver, Flutter, visual modelling, information system.

Зміст

ВСТУП

В сучасному світі швидкі та надійні бази даних в мобільних застосунках стало невід’ємною частиною цифрового середовища. Вони забезпечують якісний доступ до інформації, сервісів і віртуальних інструментів. Одним із важливих напрямів є зберігання, класифікація та перегляд візуального контенту, а саме фотографій, зображень, графіки. Це особливо актуально для користувачів, які займаються художньою діяльністю, шукають натхнення та зразки для творчості. Саме тому проєктування системи збереження й керування таким контентом є актуальним і практично значущим завданням.

Тема даної бакалаврської роботи полягає у проєктування бази даних для мобільного застосунку «Кишенькова галерея», яка присвячена створенню концептуальної моделі бази даних, яка слугуватиме основою для функціонування мобільного застосунку, призначеного для збереження, каталогізації та перегляду візуального контенту. Актуальність теми зумовлена важливістю ефективного керування великими об'ємами зображень в умовах обмежених ресурсів мобільних пристроїв. Об’єктом дослідження є процес зберігання, організації та обробки візуального контенту в межах мобільного застосунку. Предметом дослідження виступає структура, компоненти та логіка функціонування концептуальної бази даних для такого застосунку. Мета цієї роботи полягає у розробці концептуальної моделі бази даних для мобільного застосунку «Кишенькова галерея», яка забезпечить ефективну обробку та зберігання графічних матеріалів і відповідатиме вимогам сучасних мобільних платформ.

Важливими аспектами є оптимізація структури зберігання, забезпечення швидкого доступу до інформації, забезпечення цілісності даних, гнучке сортування й пошук. Усе це вимагає ретельного проєктування бази даних як одного з ключових компонентів програмної архітектури. Метою даної роботи є розробка концептуальної бази даних для мобільного застосунку «Кишенькова

галерея», яка дозволяє зберігати зображення, керувати альбомами в застосунку, тегами, описами та користувацькими налаштуваннями.

Для досягнення поставленої мети в роботі розглядаються такі завдання: аналіз предметної області та функціональних вимог до застосунку; математичне формулювання задачі й обґрунтування вибору системи керування базами даних, побудова концептуальної та логічної моделі бази даних, реалізація базових структур бази даних із демонстрацією її роботи на прикладі, оцінка ефективності та потенціалу для подальшого розвитку застосунку.

Таким чином, обрана тема бакалаврської роботи є практично затребувана та актуальна з огляду на зростання попиту на мобільні застосунки, що забезпечують персоналізоване керування візуальним контентом.

РОЗДІЛ 1.
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Особливості проєктування бази даних для застосунків з великою кількістю графічних матеріалів та постановка задачі

У більшості мобільних застосунків, що працюють з візуальним контентом: зображеннями, фотографіями, або ілюстраціями, база даних має не лише зберігати медіадані, але й якісно взаємодіяти з великими обсягами графічної інформації. У таких системах важливо забезпечити баланс між продуктивністю, надійністю та зручністю доступу до медіаданих. Одним із головних викликів при проєктуванні подібної бази даних є вибір способу зберігання зображень. Існує два основних підходи: зберігати графічні матеріали безпосередньо в базі даних у вигляді двійкових великих об'єктів або зберігати шляхи до файлів, які фізично зберігаються у файловій системі пристрою або в хмарному сховищі, а в базу даних заносити лише посилання. Кожен із цих методів має свої переваги та недоліки. Зберігання зображень у двійковому форматі дозволяє централізовано управляти даними в межах однієї системи, однак може суттєво впливати на розмір бази даних (БД), її швидкодію та стабільність при масштабуванні. З іншого боку зберігання посилань на зображення дає змогу уникнути перевантаження бази, пришвидшує її роботу та спрощує синхронізацію з зовнішніми сервісами, однак вимагає окремої обробки доступу до файлів [2].

Для мобільних застосунків, які працюють в умовах обмежених ресурсів пристрою (оперативна пам’ять, обсяг зберігання), доцільно застосовувати гібридний підхід, у якому зображення зберігаються у файловій системі, а в базі фіксуються їхні шляхи, поряд з метаданими: назвою, тегами, категорією, розмірами, датою додавання, рейтингом тощо. Крім зберігання, необхідно врахувати аспекти індексації й оптимізації доступу до зображень. Для цього використовуються індекси за тегами, датою, категорією, що дозволяє швидко фільтрувати, сортувати або шукати зображення за заданими критеріями. Для

зменшення навантаження також доцільно використовувати кешування мініатюр (thumbnail preview) – зменшених копій зображень, які завантажуються першими та використовуються для перегляду в галереї. Додатково важливим є питання масштабованості – можливості легко розширити систему без перезапуску чи втрати даних, особливо якщо застосунок підтримує хмарне резервне копіювання або синхронізацію між пристроями. У такому випадку варто враховувати формат зберігання шляхів (локальні чи URL), наявність UUID (Universally Unique Identifier) для унікальної ідентифікації кожного зображення та механізми конфлікт-менеджменту. Усе це свідчить, що проєктування бази даних для застосунків, які працюють з великою кількістю графіки, – це не лише технічне, але й архітектурне завдання. Необхідно враховувати мобільність, обмеженість ресурсів, потребу в швидкому доступі та розширюваність системи, що безпосередньо впливає на вибір системи керування базами даних (СКБД), структуру таблиць, методи доступу й збереження [4].

У зв’язку з цим виникає потреба спроєктувати концептуальну базу даних, яка відповідатиме потребам мобільного застосунку «Кишенькова галерея», що дозволить користувачам зберігати, переглядати, структурувати та керувати власною колекцією зображень (фото, сканів, артів, ілюстрацій тощо) в інтуїтивному візуальному середовищі. Особливу увагу при цьому слід приділити зберіганню метаданих (назви, дати, категорії, теги), роботі з мініатюрами, організації доступу до файлів, а також можливості гнучкого розширення функціоналу в майбутньому.

З технічної точки зору особливу увагу потрібно приділити легкості та автономності системи керування базами даних. Найбільш доцільним вибором для мобільного середовища є використання SQLite – це вбудована, легка СКБД, яка не потребує додаткових серверів, має мінімальні системні вимоги і здатна працювати навіть у режимі офлайн. Саме ці характеристики роблять її ідеальним рішенням для мобільного застосунку, особливо якщо йдеться про локальне зберігання даних [10].

Структура такої бази повинна бути добре нормалізованою, чітко розділеною на ключові сутності: користувачі, зображення, категорії, теги, колекції та таблиці зв’язків між ними. Це дозволяє підтримувати цілісність даних і гнучко масштабувати систему, додаючи нові атрибути або функції без необхідності кардинальних змін. У базі доцільно передбачити індексацію найбільш часто використовуваних полів, таких як назва зображення, дати, ідентифікатори категорій чи тегів, які призначені для прискорення операцій пошуку та фільтрації. Крім того, важливо активувати підтримку зовнішніх ключів, щоб забезпечити логічну узгодженість зв’язків між таблицями, наприклад, між зображенням і його категорією або автором.

На рівні реалізації зручно застосовувати запити типу SELECT з приєднанням тегів, категорій або колекцій, а також передбачити можливість оновлення та видалення записів із урахуванням каскадних змін. У процесі проектування важливо уникати надто складних запитів у межах користувацького інтерфейсу, щоби не знижувати швидкодію застосунку. У цьому контексті ефективним підходом буде застосування додаткових механізмів, як-от кешування результатів запитів або попередня обробка даних перед їх відображенням у застосунку.

Окремим аспектом проектування є інтеграція бази даних з інтерфейсом користувача. Візуальне відображення великої кількості зображень вимагає врахування можливостей мобільного пристрою: обмеження кількості одночасно завантажених зображень, впровадження поступового завантаження (lazy loading) або пагінації. Такі рішення дозволяють уникнути перевантаження оперативної пам’яті й покращують загальний досвід користування застосунком.

Безпека також відіграє важливу роль: навіть у простому локальному рішенні варто передбачити механізми для обмеження доступу до певних дій, захисту конфіденційної інформації, а також забезпечення резервного копіювання бази даних. У випадку, якщо система в майбутньому буде розширена до хмарної синхронізації або спільного доступу, база повинна мати гнучку структуру для злиття конфліктних змін і ведення версій даних.

Загалом, проектування бази даних для мобільного застосунку з великою кількістю графічних матеріалів базується на принципах логічної організації, ефективного зберігання метаданих і оптимізації доступу до них. Така база має бути легко інтегрованою, швидкою у роботі, здатною адаптуватися до майбутніх розширень та вимог мобільної платформи [12]. Вона є основою функціонування усього застосунку, тому її якість і продуманість прямо впливають на зручність, стабільність і масштабованість мобільного рішення.

Для реалізації поставленої мети в необхідно послідовно виконати кілька взаємопов'язаних завдань. Насамперед, провести аналіз предметної області, щоб глибше зрозуміти специфіку зберігання та організації графічної інформації в контексті мобільних застосунків. Далі проаналізувати наявні підходи до проєктування баз даних, орієнтованих на візуальні галереї, щоб визначити найефективніші практики та архітектурні рішення. Також важливим є формулювання вимог до функціональності та структури бази даних, яка мала б забезпечити зручне та швидке опрацювання візуального контенту. На основі цих вимог побудувати концептуальна модель, що враховує оптимальний спосіб зберігання зображень. Далі розробити структурну схему бази даних із чітко окресленими сутностями, атрибутами та зв'язками між ними. Завершальним етапом є тестування спроектованої бази даних, оцінка ефективності та розгляд можливих напрямів покращення.

1.2. Огляд аналогічних рішень та особливості створення

Перш ніж переходити до власного проєктування концептуальної бази даних для мобільного застосунку «Кишенькова галерея», потрібно здійснити огляд наявних рішень у сфері мобільних застосунків для зберігання та перегляду графічного контенту. Це дозволить виявити актуальні тенденції, ефективні підходи до організації даних, а також зрозуміти недоліки існуючих продуктів, яких слід уникати при реалізації власного рішення.

Першим застосунком є «Simple Gallery», який є відкритий, приватно-орієнтований мобільний застосунок для перегляду зображень. Він підтримує всі базові функції (перегляд, редагування, створення альбомів), не вимагає підключення до Інтернету і не має реклами в стандартній версії. Екрани цього застосунку зображено на рис.1.1.

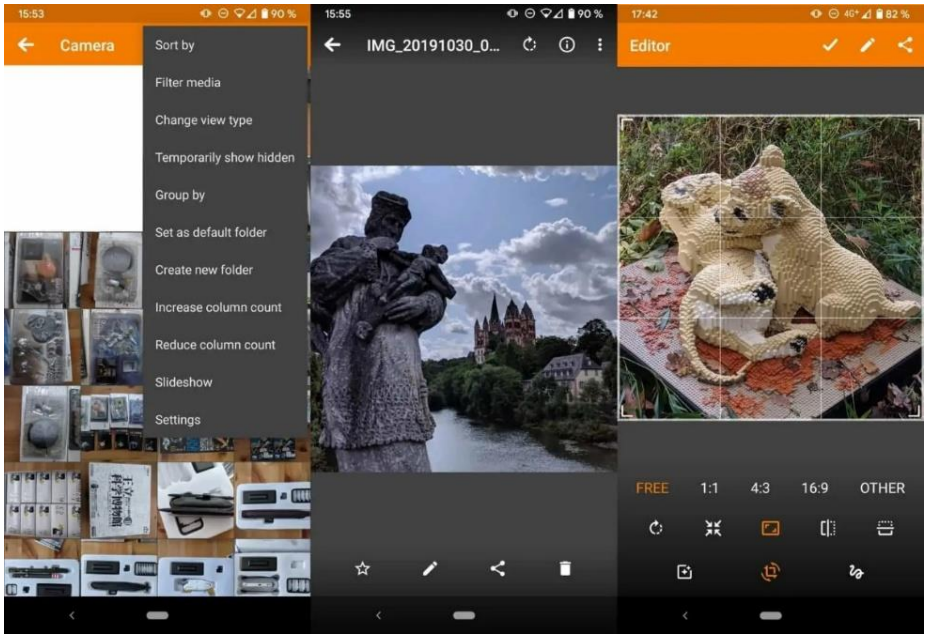

Рисунок 1.1 – Екрани застосунку Simple Gallery

Перевагами цього застосунку є повністю офлайн-режим, відкритий код, прозора політика конфіденційності можливість редагування зображень без зовнішніх сервісів підтримка жестів, сортування, захист паролем. З іншого боку, недоліками є обмежений функціонал (немає AI-інструментів, автоматичної категоризації), відсутність інтеграції з хмарними сховищами слабка система тегування.

Наступним застосунком для огляду є «Google Photos» – один із найбільш розповсюджених хмарно-орієнтованих додатків для зберігання й обробки зображень. Він пропонує автоматичну синхронізацію фотографій з хмарою, інтелектуальне сортування за обличчями, локаціями, об'єктами, а також можливість автоматичного створення альбомів, колажів, анімацій і відео, розширені можливості пошуку базуються на штучному інтелекті. Його перевагами є: потужна інтеграція з обліковим записом Google, розпізнавання

облич, місць і об'єктів автоматичне резервне копіювання, функція «спогади», що оживляє архів. Недоліками є залежність від хмарного середовища та наявності інтернет обмеження в обсязі безкоштовного хмарного сховища.

Третім застосунком для розгляду є «A+ Gallery» – це альтернатива стандартним галереям Android, яка поєднує простоту користування з деякими преміум-функціями. Вона підтримує підключення до Dropbox, Amazon Cloud та Facebook, дозволяє приховувати альбоми, має зручний інтерфейс у стилі iOS.

Переваги: синхронізація з популярними хмарними сервісами, простий та інтуїтивний інтерфейс, швидке сортування й фільтрація.

Недоліки: наявність реклами у безкоштовній версії немає глибокого аналізу чи автоматичного тегування, часткове дублювання функціоналу з іншими застосунками без унікальності.

Крім вищезгаданих галерей, є також застосунки від виробників смартфонів (Samsung, Xiaomi, OnePlus тощо). Більшість виробників Android-смартфонів надають вбудовані галереї, що часто містять додаткові функції, як-от розпізнавання сцен, інструменти швидкого редагування, інтеграцію з хмарою або «схованки» для конфіденційних фото. Проте ці рішення мають обмеження на кросплатформеність і часто пов’язані лише з екосистемою одного бренду.

Аналіз аналогічних рішень дозволив виявити кілька ключових спільних функцій, які є стандартом для мобільних галерей: створення, редагування та перегляд альбомів; базова фільтрація та сортування за датами, розширеннями, підтримка мініатюр, масштабування, слайдшоу; захист доступу до зображень (PIN-коди, приховані альбоми), проте разом із цим були виявлені типові недоліки**:** залежність від хмарних сервісів або конкретної екосистеми, обмежена категоризація на основі тегів або користувацьких атрибутів, відсутність підтримки додаткових метаданих (опис, джерело, автор, категорія), недостатня гнучкість у структурі бази даних (усі фото зберігаються «плоско» без ієрархічної логіки), неврахування потреб креативних користувачів або дизайнерів, яким важливо сортувати зображення за складнішими критеріями.

Власний застосунок «Кишенькова галерея» має на меті усунути частину перелічених недоліків, зосередившись на повноцінному зберіганні метаданих, тегів, джерел походження зображень, а також впровадженні гнучкої категоризації, що базується на власній концептуальній моделі бази даних. Особливу увагу буде приділено також підтримці автономної роботи без залежності від Інтернету та зручній інтеграції з іншими застосунками.

Також, у процесі проєктування бази даних для мобільного застосунку типу «Кишенькова галерея» надзвичайно важливо на початковому етапі сформулювати основні вимоги до системи. Ці вимоги умовно поділяються на функціональні та нефункціональні й визначають загальну архітектуру, рівень ефективності та можливості подальшого масштабування розроблюваного рішення. До функціональних вимог належать основні дії, які повинна підтримувати система бази даних для забезпечення стабільної та зручної роботи користувачів. У першу чергу, необхідно передбачити механізми зберігання графічних матеріалів або посилань на них у випадку використання зовнішнього сховища.

Застосунку «Кишенькова галерея» необхідно забезпечити підтримку метаінформації, зокрема описів, назв, категорій, тегів, дати створення, а також зв’язків між зображеннями та відповідними категоріями чи колекціями. Особливої уваги потребує реалізація пошуку та фільтрації даних за ключовими полями, такими як теги, категорії або часові характеристики. Крім того, база даних повинна підтримувати повний цикл CRUD-операцій (створення, читання, оновлення, видалення) для кожного елемента, що зберігається. За потреби система має бути здатною підтримувати англійську мову, що особливо актуально у випадку, коли застосунок орієнтований на міжнародну аудиторію. У свою чергу, нефункціональні вимоги визначають якісні характеристики майбутньої системи. Однією з ключових вимог є надійність зберігання інформації, що передбачає наявність механізмів резервного копіювання та можливості відновлення даних у разі збоїв. Швидкодія системи також відіграє важливу роль, оскільки користувачі очікують миттєвого відгуку при роботі з великою кількістю

графічного контенту. Важливою є й масштабованість бази даних, тобто здатність адаптуватися до зростання обсягів інформації без суттєвих змін у структурі або архітектурі. Також необхідно враховувати вимоги безпеки, зокрема у контексті захисту персональних або конфіденційних даних користувачів.

Останнім аспектом, що потребує врахування, є автономність і мобільність рішення: база даних повинна коректно функціонувати у середовищі мобільного пристрою навіть за відсутності постійного підключення до Інтернету. Типи даних, які повинна підтримувати розроблювана база, є достатньо різноманітними. Серед них можна виділити графічні об’єкти (у вигляді файлів або посилань), текстові значення (описи, назви, категорії, теги), числові параметри (ідентифікатори, статистичні значення), а також поля з датами та логічними параметрами (наприклад, статус зображення: «улюблене», «приховане» тощо). Таким чином, структура бази даних повинна забезпечити ефективну обробку змішаних типів інформації та передбачити оптимальну організацію взаємозв’язків між сутностями. Узагальнюючи, можна стверджувати, що якісне визначення функціональних і нефункціональних вимог до бази даних на цьому етапі проєктування дозволить у подальшому реалізувати ефективну, гнучку та масштабовану систему, яка відповідатиме як технічним умовам мобільної платформи, так і очікуванням кінцевого користувача.

РОЗДІЛ 2.
МАТЕМАТИЧНИЙ ОПИС ЗАДАЧІ ТА ВИБІР ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ ПРОЄКТУ

2.1. Формалізація предметної області

Формалізація предметної області є ключовим етапом у процесі проєктування інформаційної системи, адже саме на цьому етапі відбувається абстрагування реального світу в термінах об'єктів, зв’язків між ними та їхніх властивостей. Метою цього етапу є створення формального опису основних сутностей, що моделюють реальні явища та процеси, пов’язані з функціонуванням мобільного застосунку «Кишенькова галерея». Такий опис дозволяє забезпечити точність і послідовність у подальшій побудові концептуальної та логічної моделі бази даних [6]. Далі ми розглянемо детальніше його структуру.

Предметна область застосунку охоплює цифрову взаємодію користувача з віртуальною галереєю, в якій можна зберігати, систематизувати та переглядати зображення. Основними об’єктами моделювання є користувач, зображення, категорія, мітка (тег), коментар.

Зв’язки між об’єктами мають переважно реляційну природу: один користувач може мати багато зображень; одне зображення може бути віднесене до однієї категорії, але мати багато тегів. У разі реалізації соціальних функцій можливе впровадження механізму обміну зображеннями між користувачами або спільної галереї, що додатково ускладнює модель зв’язків.

Наступним етапом є побудова інфологічної моделі, яка є проміжною між описом предметної області мовою користувача та формальною логічною структурою БД. Інфологічна модель описує інформацію в термінах сутностей, атрибутів і зв’язків. Наприклад, сутність Користувач має атрибути: ідентифікатор (UserID), ім’я, електронна пошта, пароль, дата реєстрації. Сутність Зображення включає: ImageID, UserID (зовнішній ключ), назву, опис, дату додавання, категорію, шлях до файлу та метадані.

Таблиця 2.1 – Опис об’єктів та атрибутів застосунку «Кишенькова галерея»

Об'єкт	**Опис**	**Типові атрибути**	**Зв’язки з іншими об’єктами**
Користувач	Суб’єкт, який взаємодіє із системою, має особистий профіль	ID користувача, ім’я, email, дата реєстрації	Один користувач може мати багато зображень, може залишати коментарі
Зображення	Цифровий графічний файл, основна одиниця галереї	Назва, опис, дата створення, URL, розширення, роздільна здатність, вага файлу	Кожне зображення пов’язане з одним користувачем, однією категорією, багатьма тегами, може мати багато коментарів
Категорія	Логічна група, до якої належать зображення	Назва категорії, опис	Одна категорія може містити багато зображень
Мітка (тег)	Ключове слово для пошуку і фільтрації зображень	Назва тегу	Один тег може бути пов’язаний із багатьма зображеннями; зображення може мати багато тегів (зв’язок «багато-до-багатьох»)
Коментар	Текстова примітка або пояснення до зображення	Текст коментаря, автор, час створення	Один коментар належить одному зображенню та одному користувачу

У цій моделі важливо виявити залежності між сутностями:

Зображення ← (N:1) → *Користувач*

Зображення ← (N:M) → *Мітки*

Зображення ← (N:1) → *Категорія*

Після побудови інфологічної моделі виконується її трансформація в логічну модель, що відповідає правилам нормалізованої реляційної структури. На цьому етапі всі сутності уявляються у вигляді таблиць з чітко визначеними первинними та зовнішніми ключами. Такий підхід забезпечує уникнення надмірності, зменшення аномалій вставки, оновлення й видалення даних, а також створює основу для ефективної реалізації SQL-запитів. Наприклад, мітки як сутність мають окрему таблицю, а зв'язок «багато до багатьох» між зображеннями та мітками реалізується через проміжну таблицю «Image_Tag ».

Таким чином, формалізація предметної області дозволяє детально описати структуру даних, визначити об'єкти та зв'язки, що мають бути реалізовані в системі. Це забезпечує надійну основу для подальшого логічного і фізичного проєктування бази даних, дозволяє уніфікувати підхід до зберігання й обробки інформації, а також забезпечує відповідність архітектури бази даних поставленим функціональним вимогам.

2.2. Вибір типу бази даних

У процесі проєктування інформаційної системи для мобільного застосунку «Кишенькова галерея» особливе значення має вибір типу бази даних, оскільки саме цей компонент забезпечує зберігання, доступ, модифікацію та структуровану організацію інформації. Вибір типу системи керування базами даних безпосередньо впливає на продуктивність, масштабованість, узгодженість та безпеку даних. У сучасній практиці найбільш поширеними є реляційні та документноорієнтовані бази даних, кожна з яких має свої переваги та обмеження залежно від контексту використання.

Далі ми зробимо порівняння реляційної та документноорієнтованої баз даних. Реляційні бази даних (SQL) ґрунтуються на математичній теорії множин і використовують таблиці з чітко визначеними схемами, де дані організовані у

вигляді рядків і стовпців. Такі бази забезпечують строгі механізми цілісності даних, підтримку транзакцій, механізми нормалізації та засоби складного запитування мовою SQL. У контексті «Кишенькової галереї», де існує чітка структура сутностей (користувачі, зображення, категорії, мітки) та взаємозв'язки між ними, реляційний підхід дозволяє забезпечити логічну організацію даних з високим рівнем узгодженості.

Документноорієнтовані бази даних (NoSQL, зокрема MongoDB, Firebase Realtime Database, Firestore) дозволяють зберігати дані у вигляді JSON-подібних структур (документів), що особливо зручно для систем із гнучкою або слабо структурованою схемою. Цей підхід часто застосовується в системах, які потребують високої швидкодії при читанні/записі або мають потребу в горизонтальному масштабуванні. Проте недоліками є обмежені можливості для складних зв'язків між сутностями, відсутність транзакцій у деяких реалізаціях і відсутність чіткої схеми.

На основі здійсненого порівняння у межах розробки застосунку «Кишенькова галерея» доцільним є використання реляційної бази даних, зокрема локальної бази SQLite. Нижче наведено аргументами на користь обраного підходу, зокрема:

1. Структурована природа предметної області. Об'єкти, що використовуються в системі, мають чітко визначені атрибути та ієрархії. Зв'язки типу «один до багатьох» і «багато до багатьох» ефективно реалізуються у реляційній моделі через зовнішні ключі й таблиці зв'язків.

2. Наявність транзакцій та обмежень цілісності. Це критично важливо для забезпечення правильності змін у даних, наприклад, під час видалення користувача всі пов'язані з ним зображення також повинні бути оброблені відповідним чином.

3. Можливість Інтеграція з мобільною платформою Android. SQLite є вбудованим стандартом для зберігання локальних даних у застосунках Android, має низький поріг налаштування, не потребує підключення до інтернету та забезпечує високу продуктивність для невеликих обсягів даних.

4. Безпека та автономність. Реляційна база даних, що зберігається локально, гарантує, що персональні дані користувача не передаються на зовнішні сервери, що відповідає сучасним вимогам щодо конфіденційності.

5. Можливість масштабування. У разі необхідності подальшого розвитку системи та переходу на хмарні рішення можлива міграція даних у Firebase або інші СКБД, проте для MVP-фази використання SQLite є обґрунтованим і доцільним кроком.

Враховуючи вищезазначене, реляційна модель на базі SQLite найкраще відповідає технічним вимогам, функціональним потребам та архітектурній логіці застосунку «Кишенькова галерея». У наступному підрозділі буде здійснено аналіз і вибір інструментів, необхідних для реалізації структури бази даних та її інтеграції у програмну платформу.

2.3. Вибір інструментів та середовища реалізації

У процесі проєктування інформаційної системи «Кишенькова галерея» важливо не лише визначити функціональні та нефункціональні вимоги, а й ретельно підібрати інструменти, які дозволять ефективно реалізувати архітектуру, логіку та зберігання даних. Враховуючи, що застосунок працює з великою кількістю зображень, їх категоріями, тегами, колекціями та користувацькими метаданими, критичним є вибір оптимальної системи керування базами даних, а також інструментів для візуалізації та супроводу розробки.

Я навів порівняння систем керування базами даних які можна було використати для застосунку «Кишенькова Галерея» на рисунку 2.3

У якості основної СКБД для локального зберігання даних було обрано вбудовану реляційну базу даних SQLite, яка ідеально підходить для мобільних застосунків. Вона дозволяє зберігати структуровані дані (наприклад, інформацію про зображення, користувачів, теги, категорії, колекції) у компактному вигляді без потреби у серверному оточенні. Застосування SQLite забезпечує високу

швидкодію, стабільну роботу в офлайн-режимі та простоту розгортання. До переваг також належать підтримка SQL, низькі вимоги до ресурсів і висока надійність.

Параметр	SQLite	Realm	Firebase	MongoDB
Об'єм графіки	–	+	+	++
Офлайн	++	++	+	–
Синхронізація	–	+	++	++
Безпека	+	++	+	++
Простота	++	+	++	–

Рисунок 2.3 – Порівняння систем керування базами даних

Реальна структура бази даних була реалізована та протестована у DBeaver, як універсальному середовищі для роботи з базами даних. Саме в цій програмі створено повноцінну концептуальну модель, яка включає таблиці «users», «image»s, «collections», «tags», «categories» та проміжні зв'язки «collection_images» і «image_tags», що забезпечують реалізацію зв'язків «багато-до-багатьох». DBeaver дозволив не лише створити логічну структуру, а й протестувати SQL-запити, переглянути ER-діаграму та експортувати базу даних.

Для редагування та перевірки SQL-структури також використовувався DB Browser for SQLite, як просте графічне середовище, що дозволяє створювати, переглядати, змінювати структуру бази, а також експортувати дані в різні формати. Його використання було корисним на етапі прототипування та налагодження запитів.

На рівні розробки клієнтської візуальної частини застосунку було обрано програму Figma, яка надає широкі можливості для роботи з UI, а також підтримує інтеграцію з бібліотеками обробки зображень. Figma дозволяє створити адаптивний інтерфейс для Android та iOS з можливістю легкого експортування екранів та окремих елементів.

У разі розширення застосунку для синхронізації даних між пристроями або створення резервних копій варто розглянути інтеграцію з Firebase Cloud Firestore, який підтримує реальну синхронізацію, зберігання в хмарі та керування доступом до даних.

Таким чином, обраний технологічний стек базується на таких інструментах:

- SQLite – для локального реляційного зберігання даних;
- Figma – для дизайну інтерфейсу;
- DBeaver – для створення структури бази даних та візуалізації ER-діаграм. На рис. 2.1 зображено структуру роботи зі звязками в програмі DBeaver;

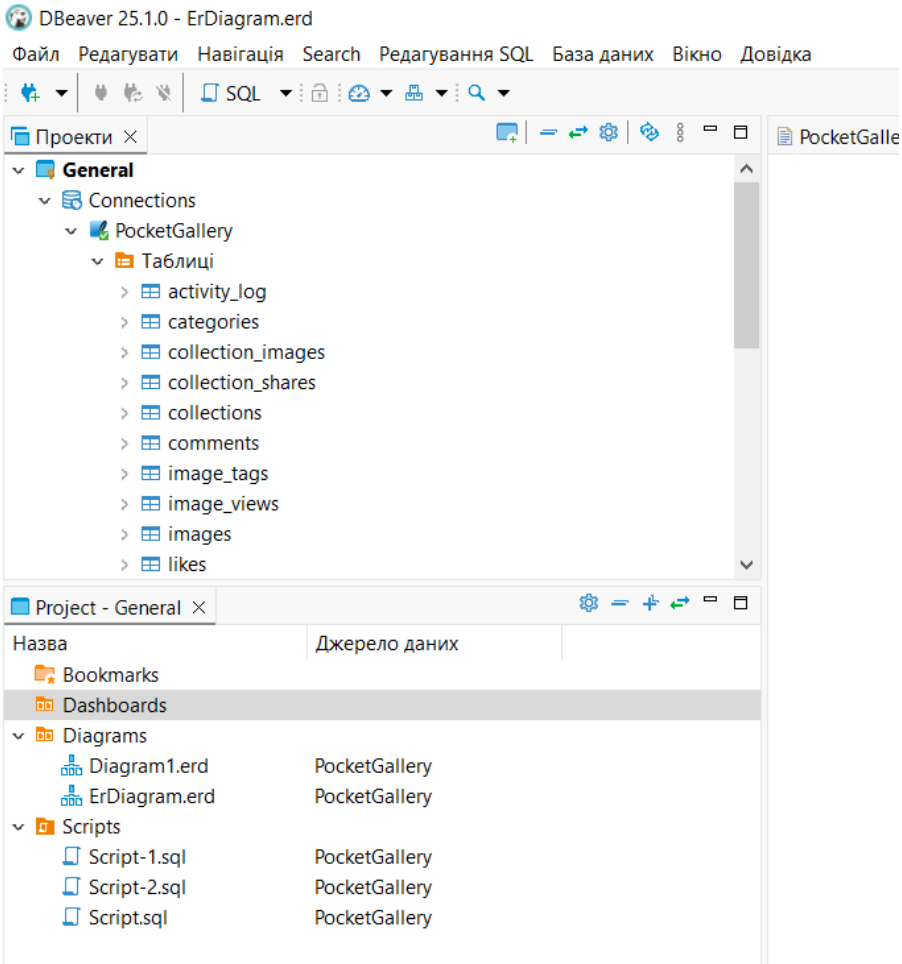

Рисунок 2.1 – Зображено структуру роботи зі зв’язками в програмі DBeaver

- DB Browser for SQLite – для маніпуляцій із базою на етапі розробки;
- Flutter – для реалізації кросплатформеного мобільного інтерфейсу;
- (Опціонально) Firebase – для майбутньої хмарної синхронізації та аналітики. На рис. 2.2 наведено схему роботи Firebase.

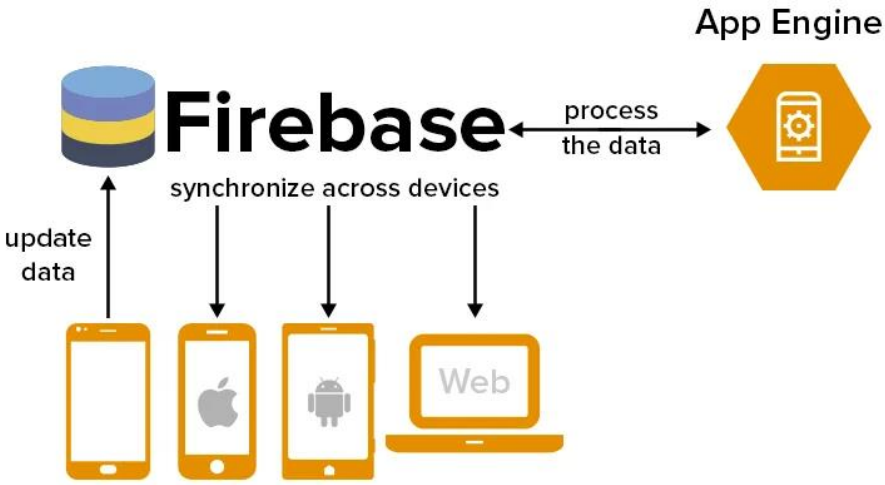

Рисунок 2.2 – Схема роботи Firebase

Вищезгадані інструменти забезпечують гнучку, масштабовану та технічно обґрунтовану екосистему для реалізації мобільного застосунку «Кишенькова галерея», яка ефективно поєднує локальну продуктивність та можливості майбутнього розширення.

РОЗДІЛ 3.
ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ КОНЦЕПТУАЛЬНОЇ БАЗИ ДАНИХ

3.1. Визначення сутностей та атрибутів

На даному етапі проєктування бази даних здійснюється ідентифікація основних сутностей, які відображають реальні об’єкти предметної області мобільного застосунку «Кишенькова галерея». Сутністю у базі даних називають абстракцію реального об'єкта, яка має визначений набір атрибутів, тобто властивостей, що характеризують цей об'єкт.

Для повноцінного та логічно узгодженого моделювання системи було виділено наступні ключові сутності:

- Користувачі – особи, що мають доступ до застосунку, можуть створювати колекції та додавати зображення.
- Зображення – основний об’єкт системи, який містить метадані, пов’язані з цифровими зображеннями.
- Категорії – класифікаційні групи, що дозволяють структурувати зображення за тематиками.
- Теги – ключові слова, що забезпечують швидкий пошук та фільтрацію зображень.
- Колекції – групи зображень, створені користувачем для персональної організації.

Опис основних сутностей бази даних згідно з структурою застосунку «Кишенькова галерея» (див. табл. 3.1).

Кожна сутність описується низкою атрибутів – простих або складених, обов’язкових або необов’язкових, унікальних або таких, що допускають повторення. Крім того, необхідно визначити типи даних для кожного атрибуту, що дозволяє забезпечити цілісність даних на рівні схеми бази.

Таблиця 3.1 – Опис основних сутностей бази даних згідно з структурою застосунку «Кишенькова галерея»

Назва сутності	**Ключові атрибути**	**Тип ключа (PK/FK)**
Користувачі (users)	user_id, email, username, created_at, role	user_id – PK
Зображення (images)	image_id, title, description, upload_date, url, category_id, author_user_id	image_id – PK, category_id – FK, author_user_id – FK
Категорії (categories)	category_id, name	category_id – PK
Теги (tags)	tag_id, label	tag_id – PK
Колекції (collections)	collection_id, name, owner_user_id	collection_id – PK, owner_user_id – FK
Зображення в колекціях (collection_images)	collection_id, image_id	collection_id – FK, image_id – FK
Теги зображень (image_tags)	image_id, tag_id	image_id – FK, tag_id – FK

Наприклад, атрибути для сутності «Зображення» включають такі поля, як:

- ID зображення (унікальний ідентифікатор),
- Назва зображення,
- Опис,
- Дата завантаження,
- URL зберігання,
- Посилання на категорію (FK),
- Посилання на автора (FK).

Типи даних для атрибутів визначаються відповідно до стандартів SQL для SQLite або Firebase (у випадку NoSQL-рішення). Зокрема, для атрибутів типово використовуються INTEGER, TEXT, DATE, BOOLEAN.

У таблиці 3.2 наведено атрибути сутностей із зазначенням типів даних і обмежень.

Таблиця 3.2 – Атрибути сутностей із зазначенням типів даних і обмежень

Сутність	**Атрибут**	**Тип даних**	**Обмеження (NOT NULL, UNIQUE, FK, тощо)**
1	2	3	4
users	user_id	INTEGER	PRIMARY KEY, AUTOINCREMENT
	email	TEXT	NOT NULL, UNIQUE
	username	TEXT	NOT NULL
	created_at	TEXT (DATETIME)	NOT NULL
	role	TEXT	CHECK(role IN ('user','admin')), DEFAULT 'user'
images	image_id	INTEGER	PRIMARY KEY, AUTOINCREMENT
	title	TEXT	NOT NULL
	description	TEXT	NULL
	upload_date	TEXT (DATETIME)	NOT NULL
	url	TEXT	NOT NULL, UNIQUE
	category_id	INTEGER	FOREIGN KEY → categories(category_id)
	author_user_id	INTEGER	FOREIGN KEY → users(user_id)

продовження табл. 3.2

1	2	3	4
categories	category_id	INTEGER	PRIMARY KEY, AUTOINCREMENT
	name	TEXT	NOT NULL, UNIQUE
tags	tag_id	INTEGER	PRIMARY KEY, AUTOINCREMENT
	label	TEXT	NOT NULL, UNIQUE
collections	collection_id	INTEGER	PRIMARY KEY, AUTOINCREMENT
	name	TEXT	NOT NULL
	owner_user_id	INTEGER	FOREIGN KEY → users(user_id)
collection_images	collection_id	INTEGER	FOREIGN KEY → collections(collection_id), PRIMARY KEY (композитний)
	image_id	INTEGER	FOREIGN KEY → images(image_id), PRIMARY KEY (композитний)
image_tags	image_id	INTEGER	FOREIGN KEY → images(image_id), PRIMARY KEY (композитний)
	tag_id	INTEGER	FOREIGN KEY → tags(tag_id), PRIMARY KEY (композитний)

Коректне визначення атрибутів сутностей на цьому етапі дозволяє не лише створити зрозумілу модель даних, але й полегшує подальше логічне й фізичне проєктування бази даних. Окрім цього, формалізоване представлення атрибутів дає змогу уникнути дублювання інформації, що є важливою умовою нормалізації структури бази даних.

3.2. Визначення зв'язків між сутностями

На цьому етапі концептуального проєктування бази даних виконується визначення логічних зв'язків між сутностями. Зв'язки (асоціації) встановлюють правила взаємодії між сутностями та дозволяють структурувати інформацію таким чином, щоб уникнути надмірності та забезпечити узгодженість між даними.

У рамках проєкту мобільного застосунку «Кишенькова галерея» ідентифіковано наступні типи зв'язків:

- Один-до-багатьох (1:N) – зв'язки при яких, один користувач може мати багато колекцій.
- Багато-до-багатьох (M:N) – коли одне зображення може мати багато тегів, і один тег може бути пов'язаний з багатьма зображеннями.
- Один-до-одного (1:1) – використовується рідше, але можлива, наприклад, для окремого профілю користувача (якщо його винесено в окрему таблицю).

Для наочного представлення структури зв'язків між сутностями розроблено основну ER-діаграму (Entity-Relationship diagram), що показано на рис. 3.3. Вона ілюструє ключові сутності, їх атрибути, а також логічні зв'язки між ними, включаючи типи зв'язків(1:1, 1:N, M:N) та обмеження цілісності.

Особливу увагу при побудові ER-діаграми приділено проміжним таблицям для реалізації зв'язків багато-до-багатьох, які трансформуються в два зв'язки типу «один-до-багатьох» за допомогою допоміжних сутностей.

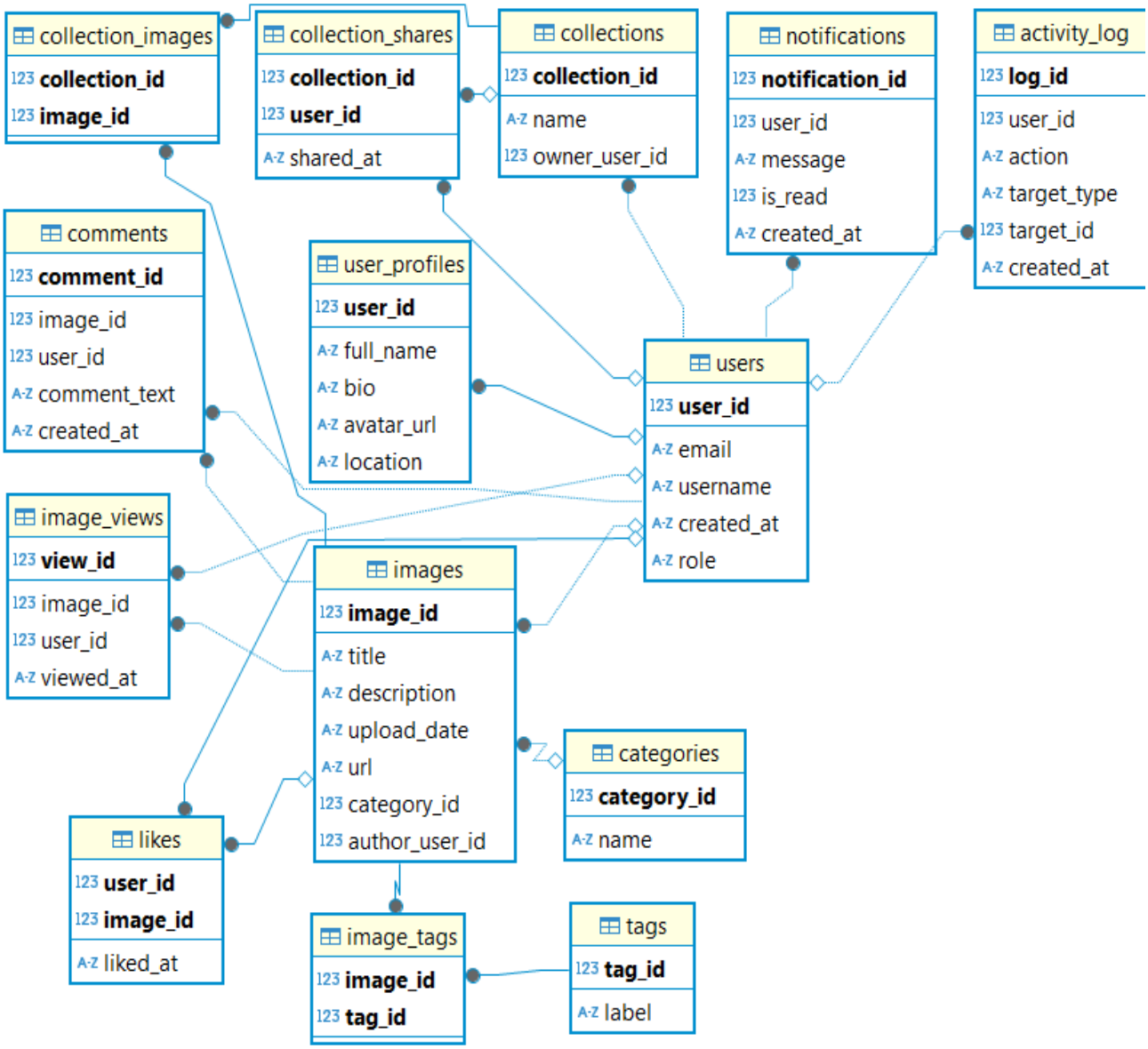

Рисунок 3.3 – Основна ER-діаграма

Розроблена структура дозволяє забезпечити логічну цілісність даних, уникнути аномалій оновлення та створює основу для подальшої нормалізації та фізичного моделювання бази даних. Усі ідентифіковані зв’язки узгоджені з предметною областю, що гарантує коректну поведінку застосунку під час взаємодії з базою даних.

3.3. Нормалізація структури бази даних

Нормалізація – це процес організації структури бази даних таким чином, щоб мінімізувати надмірність даних і уникати аномалій при оновленні,

вставленні або видаленні записів. Для оптимізації таблиць були використані так звані нормалізаційні форми з чіткими правилами. Нижче розглянемо деякі з них (1НФ, 2НФ, 3НФ).

Під час переходу до першої нормальної форми (1НФ) усі атрибути таблиць були приведені до атомарного стану, тобто кожне поле містить лише одне значення. Наприклад, атрибути, що потенційно можуть мати множинні значення (теги, зображення в колекції), були винесені в окремі таблиці зі зв'язками.

Перехід до другої нормальної форми (2НФ) досягався за рахунок того, що всі усі атрибути були приведені до залежності виключно від усього первинного ключа, а не від його частини. Це стосується в першу чергу таблиць із складовими первинними ключами, наприклад, у таблицях, які реалізують зв'язки «багато-до-багатьох»

Після вищезгаданих дій усі таблиці були приведені до 3НФ шляхом усунення транзитивних залежностей, тобто коли один неключовий атрибут залежить від іншого неключового атрибута. Наприклад, якщо в таблиці Користувачі був атрибут «Роль користувача», який сам по собі має додаткові характеристики.

У результаті виконаної нормалізації було досягнуто наступне: зменшено надмірність даних; зменшено кількість аномалій при оновленні; Спрощено забезпечення цілісності даних; Підвищено масштабованість та адаптивність структури БД до подальших змін.

Додатково варто зазначити, що нормалізація позитивно вплинула на логічну структурованість бази даних, зробивши її більш гнучкою для реалізації нових функціональностей у майбутньому. Завдяки чіткому розмежуванню сутностей та атрибутів стало можливим з легкістю інтегрувати додаткові функції, зокрема розширене тегування, механізми коментування або підтримку декількох мов. Це свідчить про високий рівень підготовки моделі до масштабування та адаптації в реальних умовах використання мобільного застосунку.

Ці зміни забезпечують логічну цілісність концептуальної моделі бази даних та надійнішу і швидшу роботу. База даних після нормалізації зображена на рисунку 3.4

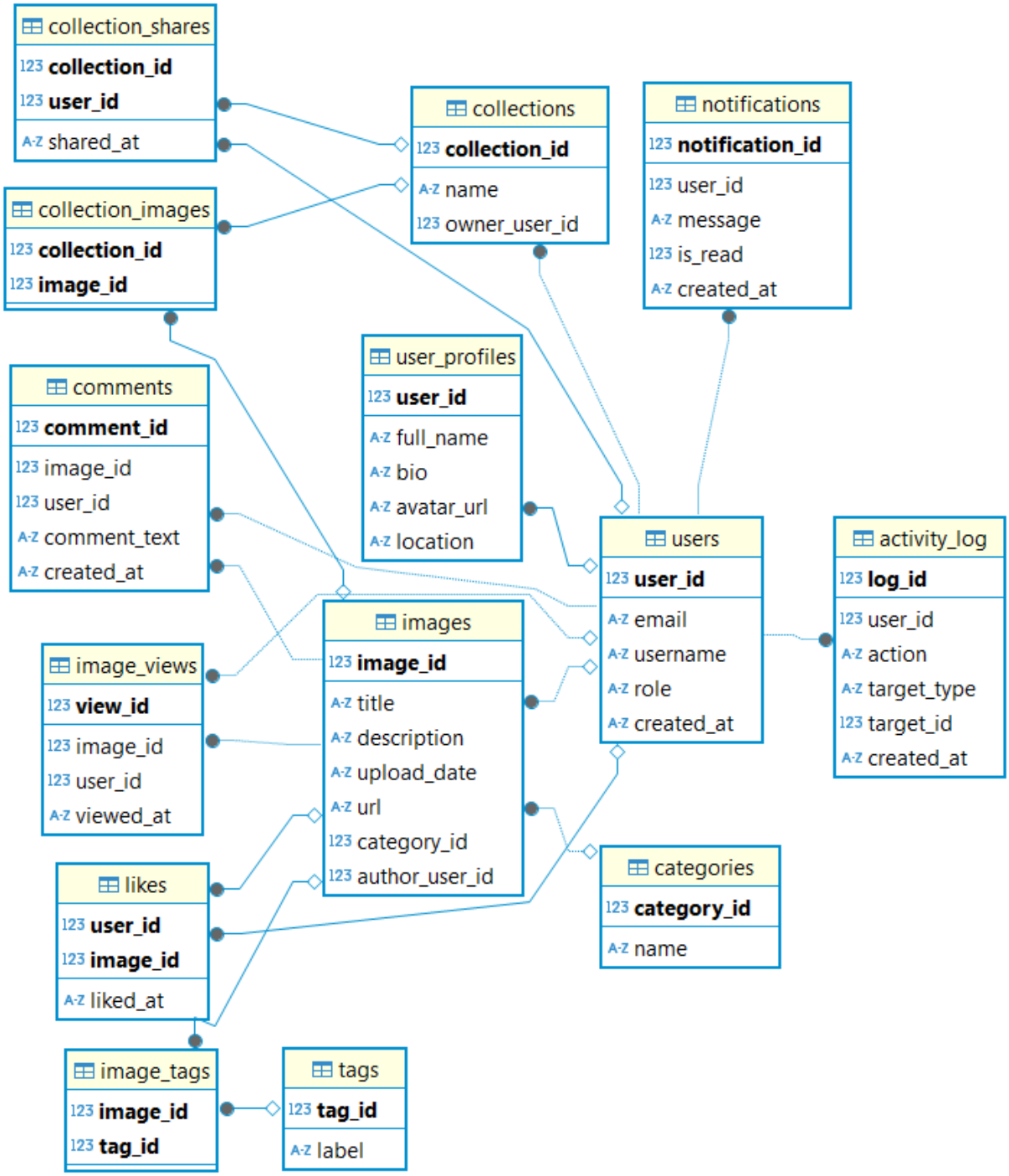

Рисунок 3.4 – Діаграма бази даних після нормалізації

Нормалізація покращила якість структури бази даних, зробивши її більш логічною, послідовною та стійкою до можливих змін у майбутньому.

У довгостроковій перспективі така модель забезпечує кращу підтримку цілісності даних, полегшує супровід і дає можливість безболісно масштабувати або розширювати систему при зростанні функціональності мобільного застосунку.

3.4. Реалізація структури бази даних у середовищі розробки

Процес реалізації бази даних було здійснено за допомогою середовища DBeaver, яке дозволило візуалізувати структуру, а також автоматизувати створення SQL-запитів. Структура бази даних базується на заздалегідь побудованій ER-діаграмі, що включає такі основні сутності: «користувачі», «зображення», «колекції», «теги», «категорії», а також зв'язувальні таблиці «collection_images» та «image_tags».

Усі таблиці створювались за допомогою SQL-запитів з урахуванням обмежень цілісності та типів даних. Створення початкових основних таблиць подано на рисунку 3.5. Розгорнутий скрипт програми можна побачити у додатку А.

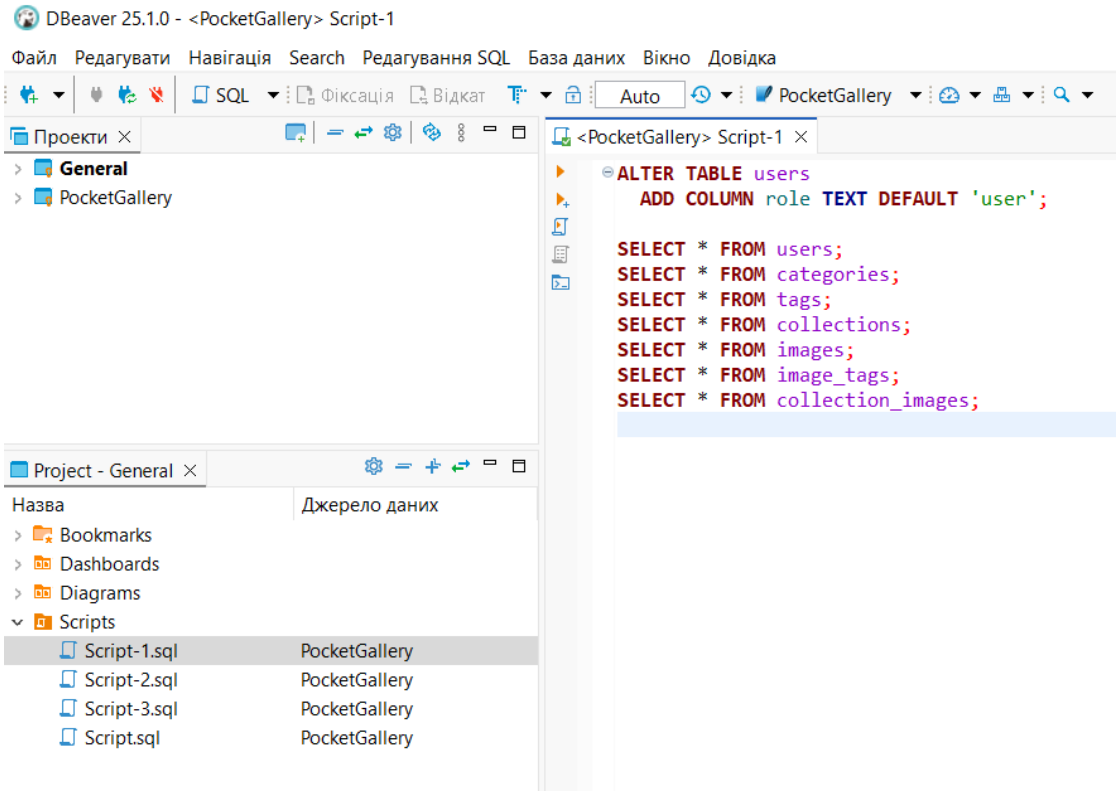

Рисунок 3.5 – Створення початкової бази даних

У проєкті активно використовуються первинні ключі (PRIMARY KEY) для унікальної ідентифікації записів, а також зовнішні ключі (FOREIGN KEY) для забезпечення зв'язків між сутностями:

- images.category_id → categories.category_id
- images.author_user_id → users.user_id
- collections.owner_user_id → users.user_id
- collection_images.image_id → images.image_id
- collection_images.collection_id → collections.collection_id
- image_tags.image_id → images.image_id
- image_tags.tag_id → tags.tag_id

На рис.3.6 наведено частину скриптів для основної бази даних.

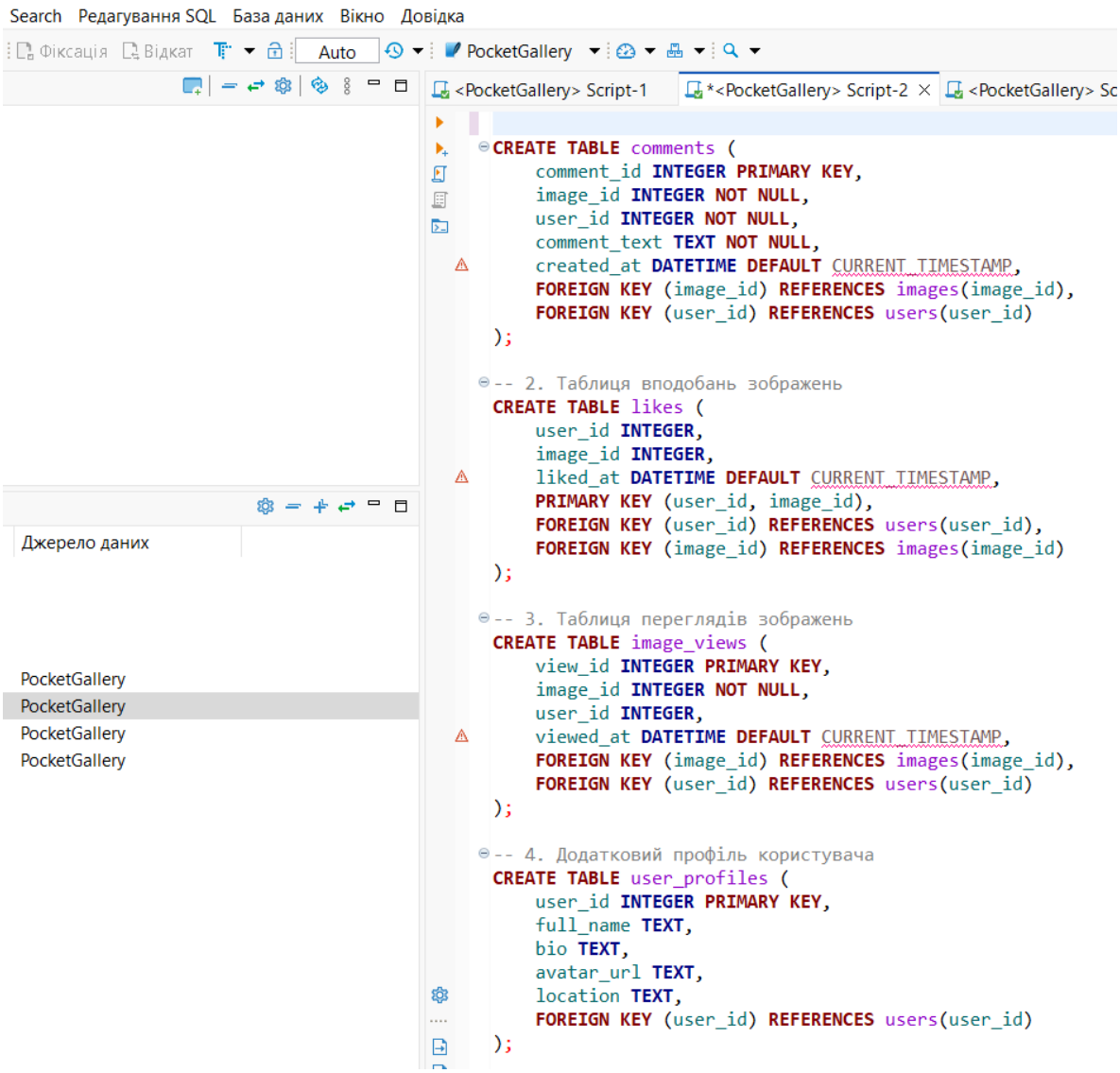

Рисунок 3.6 – Частина скриптів для основної бази даних

На рис. 3.7 наведені скрипти для створення розділу «Колекції зображень»

```
CREATE TABLE image_tags (
    image_id INTEGER NOT NULL,
    tag_id   INTEGER NOT NULL,
    PRIMARY KEY(image_id, tag_id),
    FOREIGN KEY(image_id) REFERENCES images(image_id) ON DELETE CASCADE,
    FOREIGN KEY(tag_id)   REFERENCES tags(tag_id)     ON DELETE CASCADE
);

CREATE TABLE collection_images (
    collection_id INTEGER NOT NULL,
    image_id      INTEGER NOT NULL,
    PRIMARY KEY(collection_id, image_id),
    FOREIGN KEY(collection_id) REFERENCES collections(collection_id) ON DELETE CASCADE,
    FOREIGN KEY(image_id)      REFERENCES images(image_id)       ON DELETE CASCADE
);



CREATE VIEW user_collections AS
SELECT u.user_id, u.username, c.collection_id, c.name AS collection_name
FROM users u JOIN collections c ON u.user_id = c.owner_user_id;

CREATE TRIGGER set_upload_timestamp
AFTER INSERT ON images
FOR EACH ROW
BEGIN
    UPDATE images SET upload_date = date('now') WHERE image_id = NEW.image_id;
END;
```

Рисунок 3.7 – Скрипти для створення розділу «Колекції зображень»

У цьому підрозділі було реалізовано структуру бази даних відповідно до попередньо сформованої концептуальної моделі. Створено основні таблиці, визначено первинні та зовнішні ключі для забезпечення цілісності даних та логічних зв’язків між сутностями. SQL-скрипти були реалізовані у середовищі DBeaver з використанням SQLite. Структура бази даних орієнтована на ефективне зберігання та обробку зображень, колекцій, тегів і категорій, що забезпечує основу для подальшої інтеграції з мобільним застосунком.

3.5. Тестування бази даних на прикладах користувацьких сценаріїв та інтеграція з мобільним застосунком

Після завершення створення структури бази даних було проведено тестування її працездатності шляхом імітації типових користувацьких сценаріїв.

Основна мета полягала у перевірці правильності реалізації реляційної моделі, коректності зв’язків між таблицями та відповідності запитів очікуваній логіці роботи системи. До базових сценаріїв належали: вставка, пошук, редагування та видалення записів, що відображають поведінку реального користувача в мобільному застосунку.

Першим кроком стала перевірка можливості додавання нового зображення до таблиці images. SQL-запит включав усі необхідні поля: назву, опис, дату завантаження, URL-посилання, категорію та автора. В результаті виконання було успішно створено новий запис, що підтвердило працездатність зовнішніх ключів і правильність структури. Наступним етапом стало тестування пошуку зображень за тегом, що реалізоване за допомогою JOIN-запиту між таблицями «images, image_tags» та «tags». Пошук за тегом «вишивка» успішно повернув список відповідних зображень. Це підтвердило коректність реалізації зв’язку багато-до-багатьох між тегами й зображеннями, а також ефективність використання системи для тематичного фільтрування.

Для перевірки функціоналу редагування було протестовано оновлення опису конкретного зображення. SQL-запит успішно змінив відповідне поле в таблиці, що демонструє можливість редагування метаданих. Це є важливим з точки зору користувацького досвіду, оскільки дозволяє змінювати інформацію без необхідності повторного завантаження зображення. Завершальним тестом стало видалення зображення за допомогою команди DELETE. Після виконання запиту запис було видалено з бази, а при активному каскадному видаленні автоматично видаляються пов’язані зображенням записи з інших таблиць. У разі відсутності каскадних обмежень система захищає цілісність, блокуючи видалення, якщо існують залежності.

Наостанок було здійснено базову оцінку швидкодії виконання запитів. У тестовому середовищі всі дії відбувались миттєво, що дозволяє зробити висновок про достатню продуктивність реалізованої структури. За необхідності, у майбутньому можливе застосування індексів для оптимізації складніших запитів.

На рисунку 3.8 наведено частину скриптів для тестування вставки нового зображення, пошук зображення та редагування.

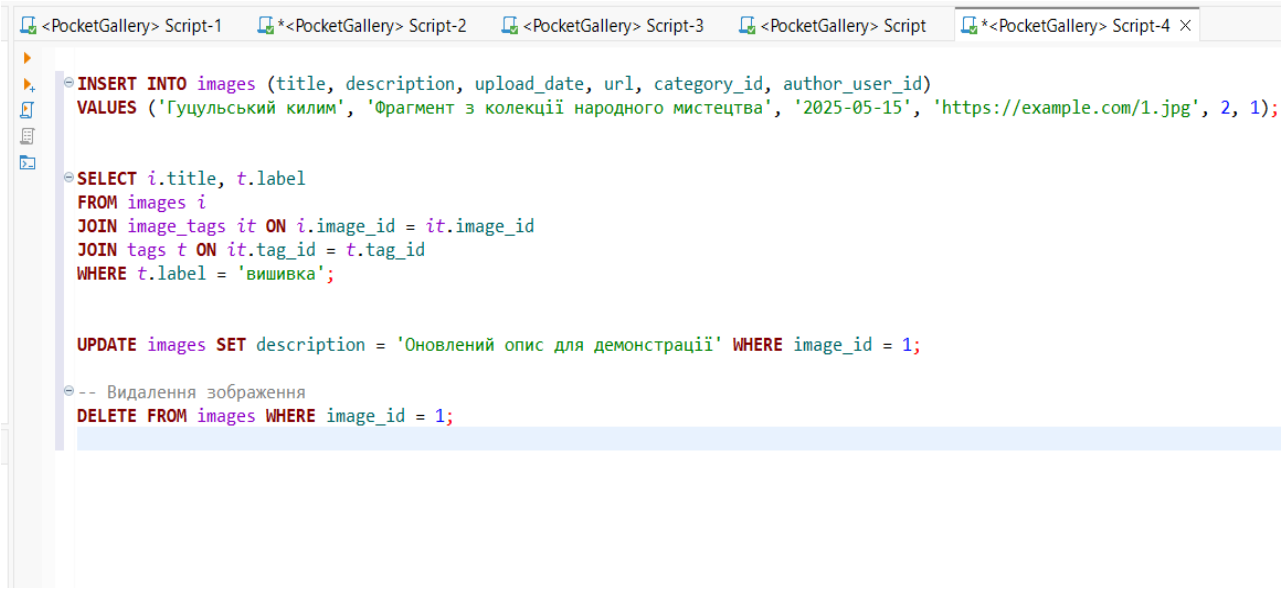

Рисунок 3.8 – Скрипти для тестування вставки нового зображення, пошук зображення та редагування

Проєкт мобільного застосунку «Кишенькова галерея» передбачає тісну інтеграцію з базою даних для збереження, керування та отримання інформації про зображення, колекції, теги, категорії та користувачів. Ретельне продумування структури доступу до даних є критично важливим для забезпечення швидкодії, масштабованості та зручності у користуванні. На рівні архітектури інтеграція реалізовується за допомогою локального доступу до вбудованої бази даних SQLite, що зберігається безпосередньо на пристрої користувача. Застосунок побудовано з використанням міжплатформного фреймворку Flutter , який підтримує взаємодію з базами даних через плагіни, такі як sqflite, moor (тепер Drift) або isar. У рамках цієї роботи концептуальна реалізація передбачає використання sqflite, як одного з найбільш стабільних і перевірених рішень.

Логіка обміну даними поділяється на кілька рівнів, першим з яких є рівень моделі даних (Model), де визначаються Dart-класи, що відповідають сутностям БД (наприклад, Image, User, Tag). Другий рівень, це рівень провайдерів або

сервісів доступу (Repository/DAO) модулі, що реалізують логіку зчитування, запису, оновлення та видалення даних. Третім є UI-рівень (Interface), тобто екрани, що відображають інформацію, отриману з бази, та взаємодіють із користувачем.

На рисунку 3.9 зображено частину дизайн-макетів екранів мобільного застосунку «Кишенькова галерея».

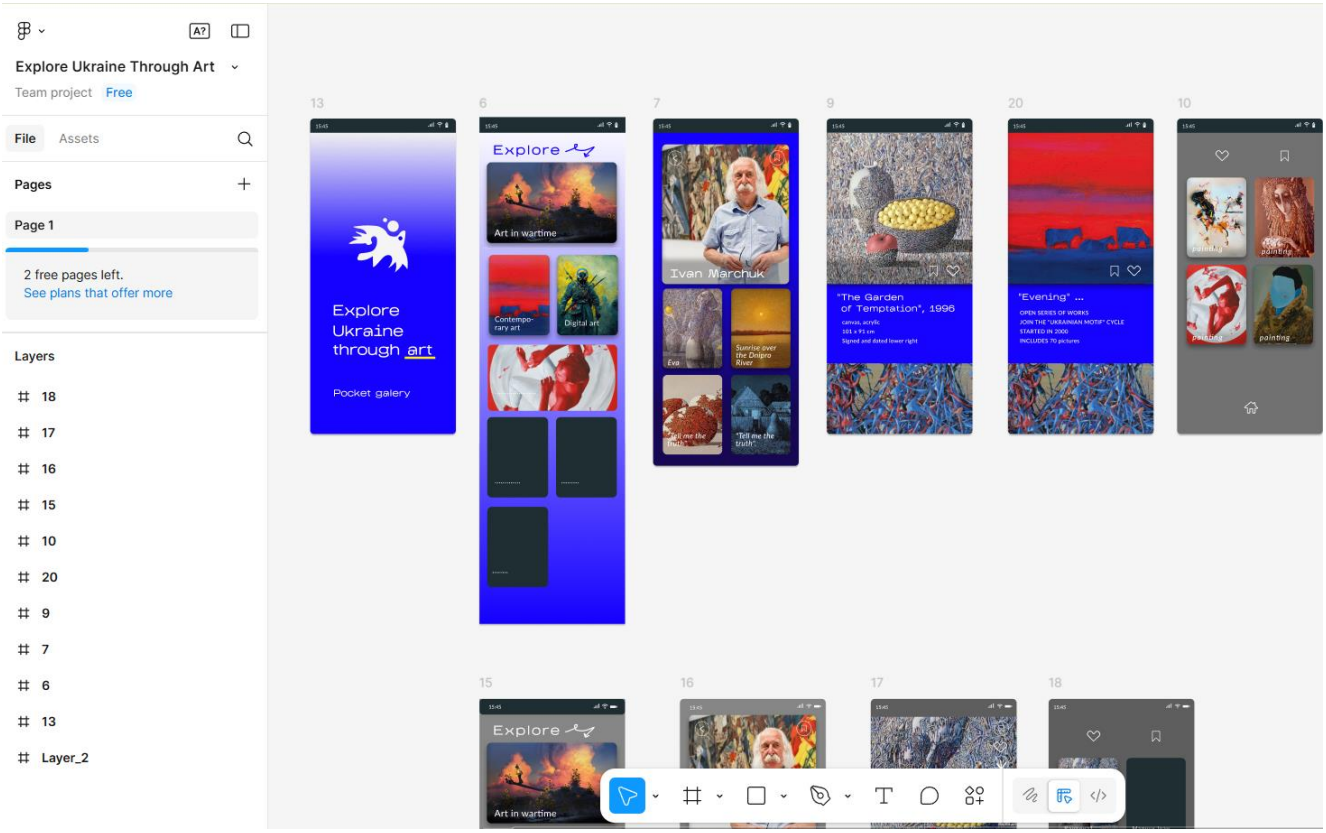

Рисунок 3.9 – Дизайн-макети екранів мобільного застосунку «Кишенькова галерея»

Також я створив візуалізацію екранів застосунку «Кишенькова галерея», яку можна побачити на (рис 3.10).

Реалізації доступу до локальної БД подана на рисунку 3.11. У реалізації клієнта використовується стандартна модель відкриття SQLite-файлу при ініціалізації програми. Створюється єдиний екземпляр бази, до якого мають доступ усі модулі.

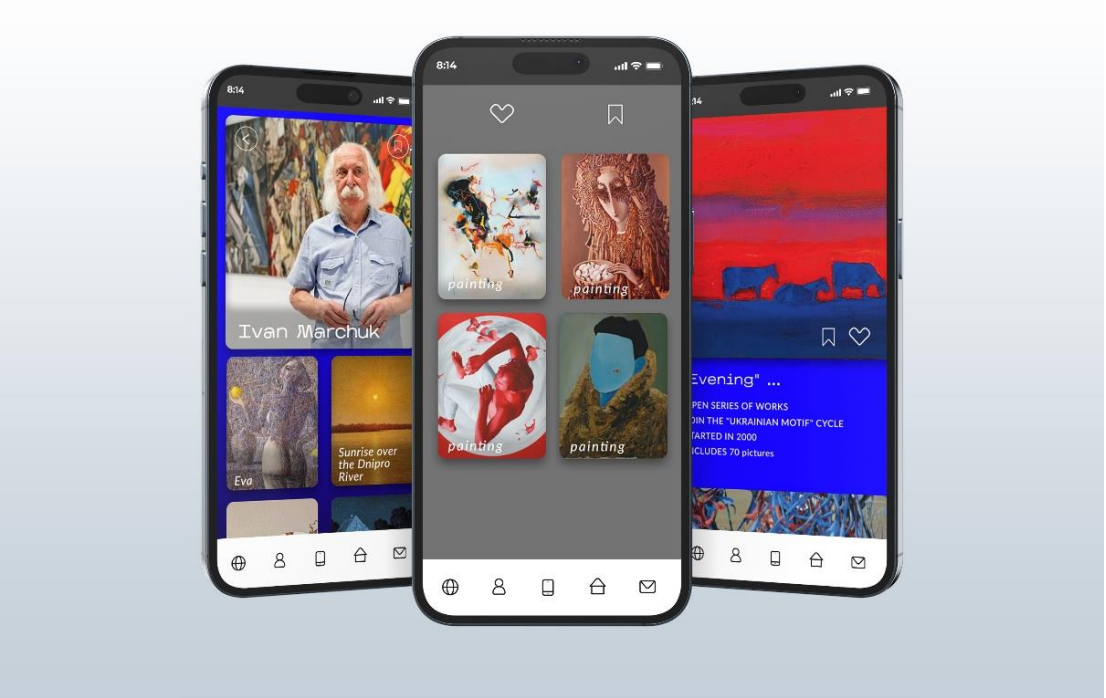

Рисунок 3.10 – Візуалізація екранів застосунку «Кишенькова галерея»

```dart
Future<Database> initDatabase() async {
  return openDatabase(
    join(await getDatabasesPath(), 'gallery_app.db'),
    onCreate: (db, version) {
      return db.execute('CREATE TABLE users(...)'); // та інші таблиці
    },
    version: 1,
  );
}
```

Рисунок 3.11– Реалізація доступу до локальної бази даних

У концептуальній реалізації передбачається, що дані кожного користувача ізольовані. При автентифікації (наприклад, через Firebase Auth у майбутньому) застосунок ідентифікує «user_id» і забезпечує доступ лише до записів, які прив’язані до цього користувача. Це дозволяє створити персоналізований досвід взаємодії, а також уникнути конфліктів у багатокористувацьких сценаріях. Якщо брати до уваги перспективи масштабування, то зі зростанням кількості даних або користувачів локальна БД може бути синхронізована з хмарним бекендом. У цьому випадку можливе використання: Firebase Cloud Firestore для зберігання

зображень і метаданих, Firebase Storage для завантаження зображень, Cloud Functions для обробки подій (наприклад, автоматичне додавання тегів за допомогою машинного навчання).

У цьому розділі я описав процес інтеграції бази даних із мобільним застосунком «Кишенькова галерея». Також розглянув архітектуру взаємодії між інтерфейсом користувача та локальною базою даних, структуру доступу до даних, приклади реалізації CRUD-операцій і можливості масштабування через хмарні сервіси. Такий підхід дозволяє забезпечити ефективну, стабільну та зручну роботу з візуальним контентом, зберігаючи гнучкість у подальшому розвитку проєкту.

Описані механізми інтеграції бази даних із мобільним застосунком демонструють не лише технічну реалізованість, а й продуманість архітектури з погляду безпеки, масштабованості та персоналізації користувацького досвіду. Застосування унікального ідентифікатора користувача дає змогу точно контролювати доступ до даних, що є надзвичайно важливим у багатокористувацькому середовищі. Розмежування даних за користувачами закладає основу для подальшого впровадження індивідуальних налаштувань, приватності та збереження контенту в межах персональних меж.

РОЗДІЛ 4.
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Розробка логіко-імітаційної моделі виникнення травм і аварій

Методикою оцінки рівня небезпеки робочих місць, машин, виробничих процесів та окремих виробництв передбачено пошук об'єктивного критерію рівня небезпеки для конкретного об'єкта. Таким показником вибрана ймовірність виникнення аварії, травми залежно від явища, що досліджується.

Оскільки небезпечні умови не завжди завчасно можна виявити, а для вивчення небезпечних дій іноді потрібно багато часу, щоб зібрати статичний матеріал, то і методи виявлення цих небезпек повинні бути відповідно диференційовані (табл. 4.1).

Таблиця 4.1 – Моделі формування та виникнення травмонебезпечних і аварійних ситуацій

Вид робіт, робоче місце, обладнання	**Виробнича небезпека: Небезпечна умова (НУ)**	**Небезпечна ситуація (НС)**	**Можливі наслідки**	**Заходи запобігання небезпечним ситуаціям**
1	2	3	4	5
Робота за комп'ютером, розробка ПЗ, введення даних, проєктування бази	Невідповідна ергономіка робочого місця (сидіння, освітлення, вентиляція), тривала робота без перерв	Погіршення фізичного стану, перенапруження очей, хребта	Хронічна втома, порушення зору, болі в спині	Організація правильного робочого місця, регулярні перерви, вправи для очей та тіла
Робота з обладнанням, підключення периферії (наприклад, серверів або накопичувачів)	Відсутність заземлення, несправне електрообладнання	Ураження електричним струмом	Травма (Т)	Проведення інструктажу з охорони праці, перевірка технічного стану обладнання, встановлення заземлення

продовження табл. 4.1

1	2	3	4	5
Робота з інтернет-ресурсами, хмарними сервісами	Відсутність антивірусного ПЗ, слабке шифрування даних	Витік даних, компрометація акаунтів	Втрата результатів роботи, порушення конфіденційності	Використання VPN, антивірусного ПЗ, двофакторної авторизації, шифрування бази даних

На основі аналізу небезпечних умов, що можуть виникати у виробничому процесі, було виділено кілька основних груп загроз залежно від їх впливу на працівника. Зокрема, до них належать умови, що визначають рівень небезпеки використаного обладнання; чинники, які сприяють виникненню технологічних помилок з боку обслуговуючого персоналу під час виконання робіт; обставини, що створюють можливість потрапляння працівника в зону підвищеного ризику; а також ситуації, що призводять до небезпечних дій, зумовлених недостатнім рівнем професійної підготовки працівників або неналежною організацією навчання з питань охорони праці.

4.2. Структурно-функціональний аналіз дотримання охорони праці при роботі з комп'ютером

Організація безпечного та ергономічного робочого середовища – важливий аспект роботи програмістів, дизайнерів та тестувальників, залучених до створення мобільного застосунку. Робоче місце має бути спроєктоване так, щоб мінімізувати навантаження на опорно-руховий апарат та органи зору.

Монітор повинен розташовуватись на рівні очей, клавіатура на такій висоті, щоб зап'ястки залишалися в нейтральному положенні. Стілець має бути регульованим по висоті, з підтримкою попереку. Також важливо, щоб працівники мали можливість змінювати положення тіла, вставати, робити легкі фізичні вправи, що зменшують статичне навантаження. Важливим є також

правильне освітлення. Яскраве штучне світло без засобів розсіювання або пряме сонячне світло, що створює відблиски на моніторі, призводять до втоми очей. Варто використовувати лампи з м'яким розсіяним світлом і розташовувати робочі місця так, щоб уникати прямих відблисків [25].

Психоемоційне навантаження – ще один аспект, який має бути врахований. Робота над великими проєктами, як-от створення мобільного застосунку, передбачає взаємодію з командою, проходження тестування. Щоб запобігти вигоранню, важливо дотримуватись гнучкого графіку, забезпечити комфортну комунікацію в команді та впроваджувати перерви.

Дотримання техніки безпеки з боку всіх учасників проєкту дозволяє зменшити ризики професійних захворювань, підвищити ефективність роботи та зберегти стабільний психофізичний стан працівників.

4.3. Планування заходів із покращення умов праці

До заходів щодо покращення умов праці належать всі види діяльності, спрямовані на попередження, нейтралізацію або зменшення негативної дії шкідливих і небезпечних виробничих факторів на працівників. Рівень умов праці оцінюють порівнянням за фактичними і нормативними значеннями узагальнених (групових) показників.

Заходи щодо поліпшення умов праці здійснюють з метою створення безпечних умов праці різними шляхами, а саме: доведення до нормативного рівня показників виробничого середовища за елементами умов праці, захисту працівників від дії небезпечних і шкідливих виробничих факторів.

До показників ефективності заходів щодо поліпшення умов праці належать: зміна кількості засобів виробництва, приведених у відповідність до вимог стандартів безпеки праці, покращання санітарно-гігієнічних показників, покращання психофізичних показників, зменшення фізичних і нервово-психічних навантажень, в т.ч. монотонних умов праці, покращання естетичних показників, раціональне компонування робочих місць і впорядкування робочих приміщень. Також соціальних результатів заходів щодо поліпшення умов праці

є такі показники: збільшення кількості робочих місць, що відповідають нормативним вимогам, зниження рівня виробничого травматизму, зменшення кількості випадків професійних захворювань, зменшення плинності кадрів через незадовільні умови праці, престиж та задоволення працею.

Отже, на покращення охорони праці потрібно виділити кошти на відновлення вентиляційних систем в робочих офісах, естетично оформити приміщення, відновити кабінет з охорони праці, поновити протипожежний інвентар.

Реалізація зазначених рекомендацій сприятиме створенню комфортного та безпечного середовища для працівників, що позитивно вплине на якість реалізації програмних проєктів, зокрема таких, як мобільний застосунок «Кишенькова галерея».

4.4. Безпека в надзвичайних ситуаціях

Актуальність проблеми природно-техногенної безпеки для населення і території, зумовлена зростанням втрат людей, що спричиняється небезпечними природними явищами, промисловими аваріями та катастрофами. Ризик надзвичайних ситуацій природного та техногенного характеру невпинно зростає, тому питання захисту цивільного населення від надзвичайних ситуацій на сьогодні є дуже важливе.

У системі цивільної оборони окремого господарства необхідно забезпечити захист населення таким чином:

Укриття в захисних спорудах, якому підлягає усе населення відповідно до приналежності, досягається створенням фонду захисних споруд.

Евакуаційні заходи, які проводяться в містах та інших населених пунктах, які мають об’єкти підвищеної небезпеки, а також у воєнний час, основним способом захисту населення є евакуація і розміщення його у позаміській зоні.

Медичний захист проводиться для зменшення ступеня ураження людей, своєчасного надання допомоги постраждалим та їх лікування, забезпечення епідеміологічного благополуччя в районах надзвичайних ситуацій.

Радіаційний і хімічний захист включає заходи щодо виявлення і оцінки радіаційної та хімічної обстановки, організацію і здійснення дозиметричного та хімічного контролю, розроблення типових режимів радіаційного захисту, забезпечення засобами індивідуального захисту, організацію і проведення спеціальної обробки.

Евакуаційні заходи, які проводяться в містах та інших населених пунктах, які мають об'єкти підвищеної небезпеки, а також у воєнний час, основним способом захисту населення є евакуація і розміщення у позаміській зоні.

Загалом, захист населення є системою загально державних заходів, які реалізуються центральними і місцевими органами виконавчої влади, виконавчими органами влад, органами управління з питань надзвичайних ситуацій та цивільного захисту населення, підпорядкованими їм системами, та підприємств, що забезпечують виконання організаційних, інженерно – технічних, санітарно – гігієнічних, проти епідемічних та інших заходів у сфері запобігання та ліквідації наслідків надзвичайних ситуацій.

Загрози життєво важливих інтересів громадян, держави, суспільства поділяють на зовнішні та внутрішні, виконують під час надзвичайних ситуацій техногенного та природного характеру та воєнних конфліктах.

Принципи захисту випливають з основних положень Женевської конвенції щодо захисту жертв війни та додаткових протоколів до неї, можливого характеру воєнних дій, реальних можливостей держави щодо створення матеріальної бази захисту. З метою захисту населення, зменшення втрат та шкоди економіці в разі виникнення надзвичайних ситуацій має право проводитись спеціальний комплекс заходів.

Оповіщення та інформування, яке досягається завчасним створенням і підтримкою в постійній готовності загальнодержавної, територіальних та об'єктивних систем оповіщення населення.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи на тему «Проєктування концептуальної бази даних для мобільного застосунку «Кишенькова галерея» було реалізовано повноцінний комплекс завдань, починаючи від формування ідеї до створення логічно узгодженої структури бази даних та розробки прикладних SQL-скриптів, які забезпечують роботу з даними. Головна ідея проєкту полягала в тому, щоб створити просту й зручну систему, у якій користувачі могли б зберігати, впорядковувати та переглядати зображення за категоріями, тегами або власними колекціями. Така функціональність особливо актуальна для дизайнерів, митців, дослідників культурної спадщини або просто людей, які цінують візуальний порядок.

На початкових етапах було здійснено огляд існуючих технологій, аналіз їхніх переваг та нелоліків та постановка задачі для подальшого опрацювання.

Усе це вимагало продуманого проєктування бази даних – основи для надійного функціонування застосунку. Було обґрунтовано вибір SQLite як основної системи керування базами даних – завдяки її легкості, швидкості, простоті інтеграції з мобільними платформами та відсутності потреби в зовнішньому сервері. Такий вибір ідеально підходить для локальних мобільних застосунків, які повинні працювати навіть без інтернету.

Структура бази даних охоплює ключові сутності: користувачі, зображення, категорії, теги, колекції та зв’язки між ними. Усе це дозволяє зручно організувати інформацію, пов’язувати зображення з певними тематиками або ключовими словами, об’єднувати їх у добірки тощо. Реалізовано низку SQL-запитів, які демонструють базові можливості роботи з даними: вставка, пошук за тегом, оновлення та видалення. Кожен із них був протестований на практиці, що дозволило впевнитися в коректності логіки та роботи самої структури.

Під час роботи було використано зручне середовище DBeaver, яке допомогло візуалізувати структуру бази, перевірити запити та зробити процес проєктування більш наочним. Крім того, були враховані вимоги щодо охорони

праці та електробезпеки під час роботи з комп'ютерною технікою – як важливий елемент відповідального підходу до професійної діяльності. Окремо було розглянуто й питання майбутньої інтеграції бази даних у мобільний застосунок. У межах інтерфейсної частини продемонстровано, як користувач може взаємодіяти з базою – переглядати галереї, додавати нові зображення, шукати за тегами тощо. Це наближає роботу до реального практичного застосування, а не лише теоретичного прототипу. У підсумку, дана дипломна робота стала не просто вправою з проєктування баз даних, а справжнім прикладом комплексного IT-проєкту. Створена система є хорошим стартом для розвитку повноцінного мобільного застосунку, який у майбутньому можна доповнити хмарною синхронізацією, реєстрацією користувачів, авторизацією, обміном зображеннями або інтеграцією з Google/Firebase-сервісами. Такий проєкт є чудовою основою для подальшого професійного зростання, розвитку у сфері мобільної розробки та зберігання медіаданих.

Зважаючи на те, що у ході реалізації роботи було закладено базову архітектуру системи для мобільного застосунку «Кишенькова галерея», також доцільно розглянути шляхи його подальшого вдосконалення. Майбутнє розширення функціоналу системи дозволить не лише підвищити її зручність для користувачів, але й забезпечити інтеграцію з сучасними технологічними платформами.

Першочерговим напрямом розвитку може бути інтеграція з хмарними сервісами зберігання даних, такими як Google Drive або Firebase. Це забезпечить можливість синхронізації даних між пристроями, резервного копіювання та доступу до зображень у режимі онлайн. У поєднанні з цим необхідним є впровадження системи автентифікації користувачів, яка включатиме підтримку реєстрації, входу через електронну пошту або сторонні сервіси (наприклад, Google), а також механізмів безпечної авторизації та збереження облікових даних.

У разі значного розширення обсягів даних та збільшення кількості користувачів варто розглянути можливість переходу від SQLite до більш потужної системи керування базами даних, наприклад PostgreSQL або Firebase Realtime Database. Це дозволить забезпечити стабільну роботу застосунку за умов високих навантажень та розподіленої архітектури.

Окрім мобільної версії, перспективним напрямком є розробка веб-версії застосунку, яка забезпечить повноцінний доступ до функціоналу галереї через браузер. Такий підхід дозволить розширити сценарії використання системи, зокрема для тих користувачів, які працюють на персональних комп’ютерах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Руденко О.В. Модуль «Бази даних» : навч. посібн. Кропивницький: ЦНТУ, 2018. 124 с.

2. Богданов І. П. Бази даних MySQL: навч. посібн. ТНТУ, 2020. 152 с.

3. Гнатюк С. М. Бази даних : навч. посібн. НАУ, 2019. 200 с.

4. Костенко О. М. Бази даних та системи управління базами даних. Харків: ХНУРЕ, 2017. 176 с.

5. Сидоренко А. О. Основи проектування баз даних : навч. посібн. Одеса: ОНАХТ, 2020. 98 с.

6. Ткаченко І. В., Гуменюк Т. О. Бази даних та інформаційні системи. Львів: ЛНУ ім. Івана Франка, 2021. 178 с.

7. Юрченко В. О. Бази даних у мобільних застосунках. Наукові записки НТУУ «КПІ». 2020. №4. С. 45-51.

8. Олійник І. О. Системи керування базами даних : підручн. Львів: Видавництво ЛНУ, 2022. 228 с.

9. Гончаренко Т. В. Мобільні інформаційні системи : навч. посібн. Херсон: ХДАУ, 2019. 132 с.

10. Ільїна А. М. Особливості використання СКБД SQLite в Android-застосунках. Сучасні інформаційні технології та ІТ-освіта. 2021. №2. С. 67-72.

11. Левченко С. В. SQLite: особливості зберігання даних у мобільних платформах. Технології та інновації. 2021. №2. С. 42-46.

12. Іваненко П. П. Сучасні технології роботи з БД у мобільних пристроях. Науковий вісник ЧНУ. 2022. №4. С. 66-73.

13. Шлапак О. В. Бази даних. Проєктування та реалізація. К.: Каравела, 2021. 160 с.

14. Allen G., Owens M. The Definitive Guide to SQLite. 2nd ed. Berkeley: Apress, 2010. 368 p.

15. Kreibich J. Using SQLite. Sebastopol: O’Reilly Media, 2010. 530 p.

16. Das S. SQLite for Mobile Apps Simplified. Charleston: CreateSpace Independent Publishing, 2014. 260 p.

17. Feiler J. Introducing SQLite for Mobile Developers. New York: Apress, 2015. 170 p.

18. Haldar S. SQLite Database System Design and Implementation. Independently published, 2015. 290 p.

19. Newman C. SQLite (Developer's Library). Indianapolis: Sams Publishing, 2004. 312 p.

20. Aditya S. K., Karn V. K. Android SQLite Essentials. Birmingham: Packt Publishing, 2014. 154 p.

21. Lenz K. Beginning Android Programming with Android Studio. 4th ed. Indianapolis: Wiley, 2019. 800 p.

22. Elmasri R., Navathe S. B. Fundamentals of Database Systems. 7th ed. Boston: Pearson, 2016. 1272 p.

23. Connolly T., Begg C. Database Systems: A Practical Approach to Design, Implementation and Management. 6th ed. Harlow: Pearson Education, 2015. 1440 p.

24. ISO 45001:2018 Occupational health and safety management systems. International Organization for Standardization [Електронний ресурс]. Режим доступу: https://www.iso.org/standard/63787.html .

25. Голінько В.І., Іконніков М.Ю., Лебедєв Я.Я. Охорона праці в галузі інформаційних технологій : навч. посіб.; М-во освіти і науки України, Нац. гірн. ун-т. Д. : НГУ, 2015. 246 с.

ДОДАТКИ

Додаток А

Скрипт1:

```sql
PRAGMA foreign_keys = ON;

CREATE TABLE users (
    user_id    INTEGER PRIMARY KEY AUTOINCREMENT,
    email      TEXT    NOT NULL UNIQUE,
    username   TEXT    NOT NULL,
    created_at DATE    NOT NULL
);

CREATE TABLE categories (
    category_id INTEGER PRIMARY KEY AUTOINCREMENT,
    name        TEXT    NOT NULL UNIQUE
);

CREATE TABLE tags (
    tag_id INTEGER PRIMARY KEY AUTOINCREMENT,
    label  TEXT    NOT NULL UNIQUE
);

CREATE TABLE collections (
    collection_id INTEGER PRIMARY KEY AUTOINCREMENT,
    name          TEXT    NOT NULL,
    owner_user_id INTEGER NOT NULL,
    FOREIGN KEY(owner_user_id) REFERENCES users(user_id) ON DELETE CASCADE
);

CREATE TABLE images (
    image_id       INTEGER PRIMARY KEY AUTOINCREMENT,
    title          TEXT    NOT NULL,
    description    TEXT,
    upload_date    DATE    NOT NULL,
    url            TEXT    NOT NULL UNIQUE,
    category_id    INTEGER,
    author_user_id INTEGER,
    FOREIGN KEY(category_id)     REFERENCES categories(category_id) ON DELETE SET NULL,
    FOREIGN KEY(author_user_id)  REFERENCES users(user_id)       ON DELETE SET NULL
);



CREATE TABLE image_tags (
    image_id INTEGER NOT NULL,
    tag_id   INTEGER NOT NULL,
    PRIMARY KEY(image_id, tag_id),
    FOREIGN KEY(image_id) REFERENCES images(image_id) ON DELETE CASCADE,
    FOREIGN KEY(tag_id)   REFERENCES tags(tag_id)     ON DELETE CASCADE
);

CREATE TABLE collection_images (
    collection_id INTEGER NOT NULL,
    image_id      INTEGER NOT NULL,
    PRIMARY KEY(collection_id, image_id),
```

```
    FOREIGN KEY(collection_id) REFERENCES collections(collection_id) ON DELETE CASCADE,
    FOREIGN KEY(image_id)      REFERENCES images(image_id)       ON DELETE CASCADE
);



CREATE VIEW user_collections AS
SELECT u.user_id, u.username, c.collection_id, c.name AS collection_name
FROM users u JOIN collections c ON u.user_id = c.owner_user_id;

CREATE TRIGGER set_upload_timestamp
AFTER INSERT ON images
FOR EACH ROW
BEGIN
    UPDATE images SET upload_date = date('now') WHERE image_id = NEW.image_id;
END;

-- 4. Тестові дані (INSERT) для всіх таблиць

INSERT INTO users(email, username, created_at) VALUES
  ('alice@example.com', 'alice', '2025-06-10'),
  ('bob@example.com',   'bob',   '2025-06-09');

INSERT INTO categories(name) VALUES ('Nature'), ('Art');

INSERT INTO tags(label) VALUES ('sunset'), ('portrait');

INSERT INTO collections(name, owner_user_id) VALUES
  ('Alice''s Albums', 1),
  ('Bob''s Collection', 2);

INSERT INTO images(title, description, upload_date, url, category_id, author_user_id)
VALUES
  ('Sunset Beach',    'A beautiful sunset',   '2025-06-11',
'http://example.com/sunset.jpg',    1, 1),
  ('Portrait of Bob', 'Studio portrait',      '2025-06-11',
'http://example.com/portrait.jpg',  2, 2);

INSERT INTO image_tags(image_id, tag_id) VALUES (1, 1), (2, 2);

INSERT INTO collection_images(collection_id, image_id) VALUES (1, 1), (2, 2);
```

Додаток Б

Скрипт 2:

```
CREATE TABLE comments (
    comment_id INTEGER PRIMARY KEY,
    image_id INTEGER NOT NULL,
    user_id INTEGER NOT NULL,
    comment_text TEXT NOT NULL,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (image_id) REFERENCES images(image_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- 2. Таблиця вподобань зображень
CREATE TABLE likes (
    user_id INTEGER,
    image_id INTEGER,
    liked_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (user_id, image_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id),
    FOREIGN KEY (image_id) REFERENCES images(image_id)
);

-- 3. Таблиця переглядів зображень
CREATE TABLE image_views (
    view_id INTEGER PRIMARY KEY,
    image_id INTEGER NOT NULL,
    user_id INTEGER,
    viewed_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (image_id) REFERENCES images(image_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- 4. Додатковий профіль користувача
CREATE TABLE user_profiles (
    user_id INTEGER PRIMARY KEY,
    full_name TEXT,
    bio TEXT,
    avatar_url TEXT,
    location TEXT,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- 5. Сповіщення користувачам
CREATE TABLE notifications (
    notification_id INTEGER PRIMARY KEY,
    user_id INTEGER NOT NULL,
    message TEXT NOT NULL,
    is_read BOOLEAN DEFAULT 0,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- 6. Спільний доступ до колекцій
CREATE TABLE collection_shares (
```

```
    collection_id INTEGER,
    user_id INTEGER,
    shared_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (collection_id, user_id),
    FOREIGN KEY (collection_id) REFERENCES collections(collection_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- 7. Журнал дій користувачів
CREATE TABLE activity_log (
    log_id INTEGER PRIMARY KEY,
    user_id INTEGER,
    action TEXT,
    target_type TEXT,
    target_id INTEGER,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

Додаток В

Тестування, скрипт 3:

```sql
-- Вставка нового зображення
INSERT INTO images (title, description, upload_date, url, category_id, author_user_id)
VALUES ('Гуцульський килим', 'Фрагмент з колекції народного мистецтва', '2025-05-15',
'https://example.com/1.jpg', 2, 1);

-- Пошук зображень за тегом
SELECT i.title, t.label
FROM images i
JOIN image_tags it ON i.image_id = it.image_id
JOIN tags t ON it.tag_id = t.tag_id
WHERE t.label = 'вишивка';

-- Оновлення опису
UPDATE images SET description = 'Оновлений опис для демонстрації' WHERE image_id = 1;

-- Видалення зображення
DELETE FROM images WHERE image_id = 1;
```