Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет Інформатики та Обчислювальної Техніки

Кафедра інформаційних систем та технологій

Лабораторна робота №3

з дисципліни «Розробка мобільних застосувань під Android»

на тему

«ДОСЛІДЖЕННЯ СПОСОБІВ ЗБЕРЕЖЕННЯ ДАНИХ»

Виконав:

студент групи ІС-21

Качалов А. М.

Викладач:

Орленко С. П.

Київ – 2025

**Мета роботи:** дослідити способи збереження даних (база даних, файлова система, тощо) та отримати практичні навички щодо використання сховищ даних.

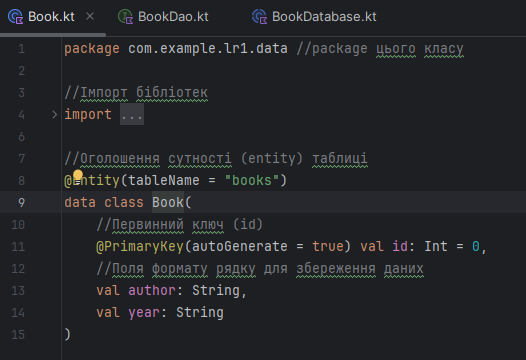
**ЗАВДАННЯ**

Написати програму під платформу Андроїд, яка доповнює програму, що розроблена за лабораторною роботою 2, роботою зі сховищами. Тобто при натисканні на кнопку «ОК» додатково: - здійснюється запис результату взаємодії з інтерфейсом до сховища (файл або базу даних); - користувач інформується відповідним повідомленням щодо успішності запису. Також інтерфейс необхідно доповнити кнопкою «Відкрити», натискання на яку призводить до переходу на іншу Діяльність, у якій відображається вміст даних, що зберігаються у сховищі. Якщо дані відсутні (сховище пусте) відобразити відповідне повідомлення. За бажанням можна додатково реалізувати оновлення та видалення даних зі сховища.

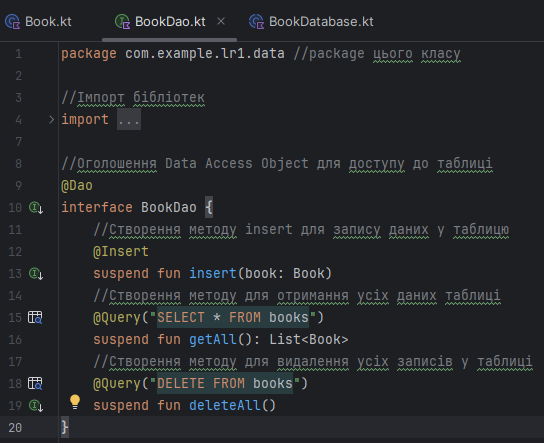
**Хід роботи**

**1.Програмна реалізація**

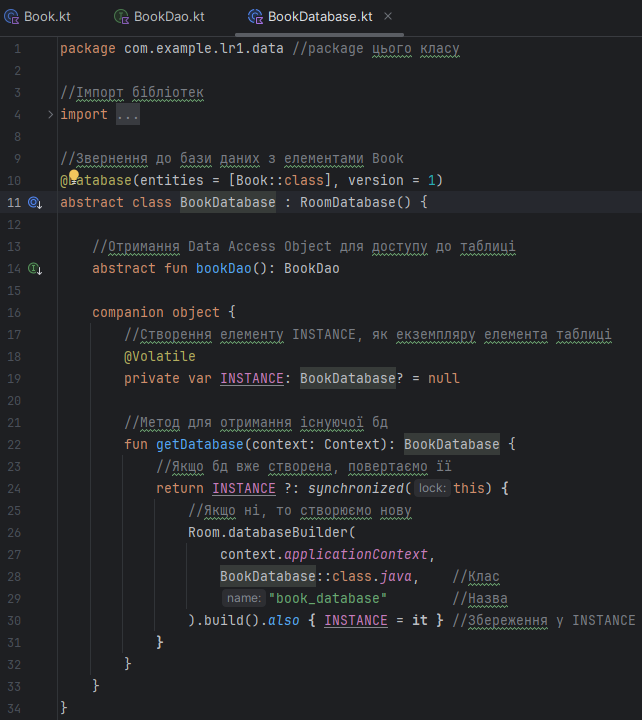
В папці Data прописана логіка сховища, нашої бази даних. У файлі Book ми оголошуємо сутність таблиці, а також первинний ключ.



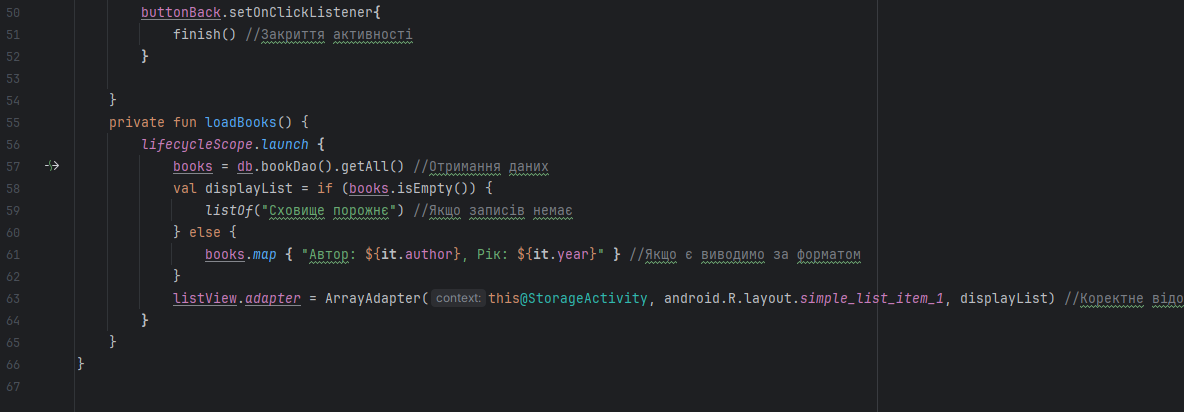
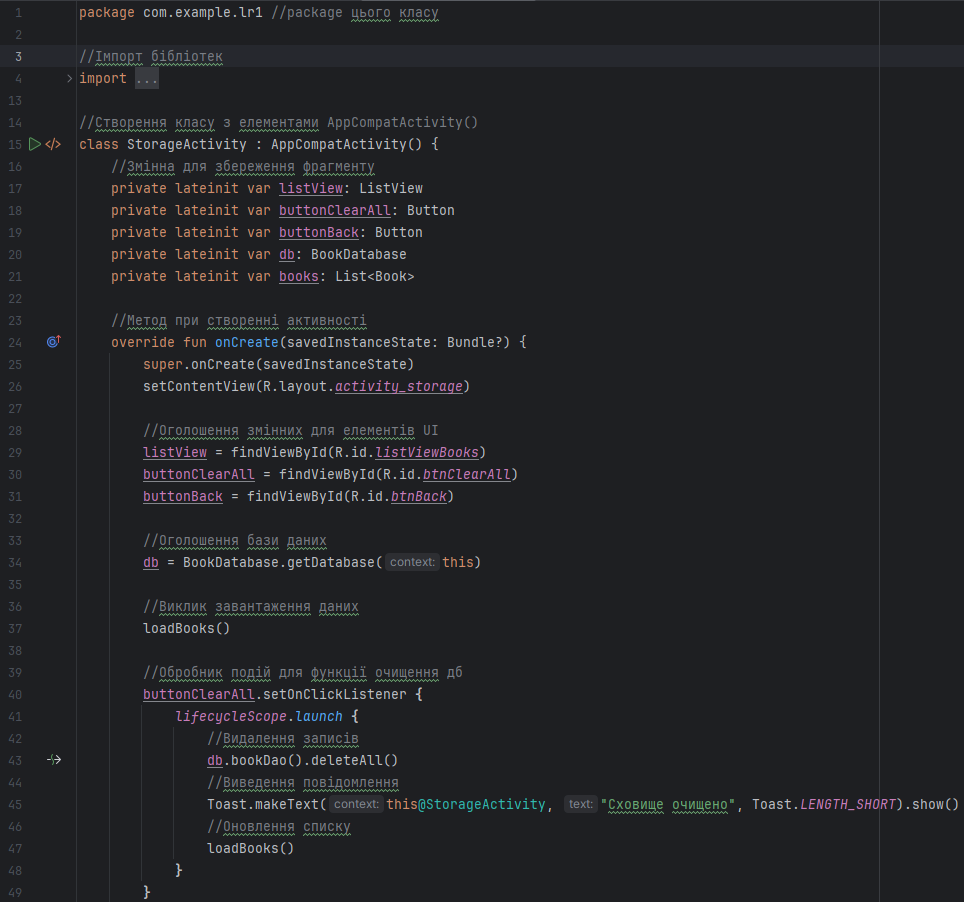
У файлі BookDao прописано доступ до таблиці, щоб мати можливість створити функції та звертатись до таблиці та виконувати дії із нею.



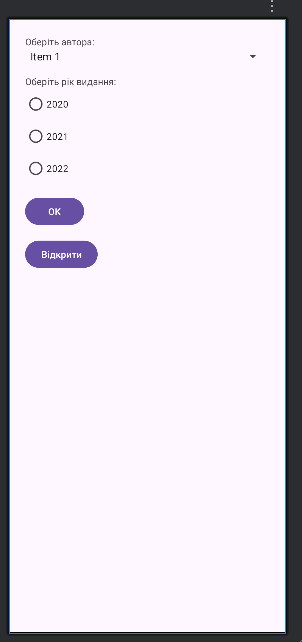
У файлі BookDatabase прописано логіку самої бази даних, тобто: звернення до неї, отримання доступу, екземпляр таблиці, прописано метод для отримання існуючої бази даних (якщо вона вже створена – повертаємо її, якщо ні – створюємо).

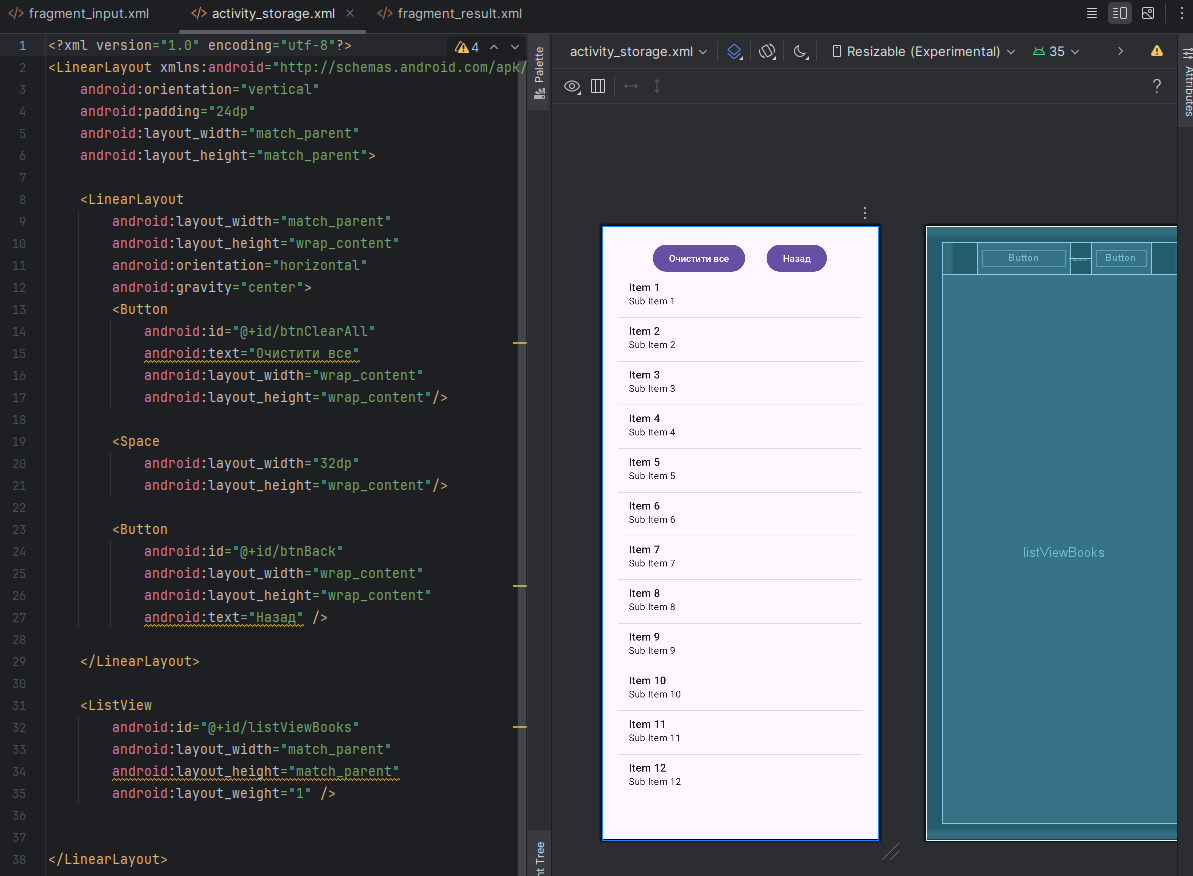


У файлі Storage Activity ми створюємо клас з елементами, оголошуємо змінні для збереження фрагментів, також прописуємо метод при створенні активності, оголошуємо змінні для елементів UI, оголошуємо саму базу даних, викликаємо завантаження даних, прописуємо обробник подій для функції очищення бази даних, закриваємо активність. Сама робота сховища виглядає наступним чином: база даних отримує значення (дані), перевіряє їх на наявність, і якщо записів немає, то виводить повідомлення про те, що сховище порожнє, а якщо є, то виводить запис за форматом Автор та Рік випуску.



В лейаутах, у фрагменті ведення даних, додано кнопку «Відкрити». В Activity Storage прописано зовнішній вигляд нашого сховища, який включає в себе 2 кнопки: очищення і повернення, а також список елементів у базі даних.





**2. Робота програмного продукту**

Для тестування правильної роботи програми, спробуємо додати елемент до бази даних, відкрити її та очистити. Так зможемо дізнатись чи додаються елементи до сховища, як вони там зберігаються, як працює очищення БД та як виглядає порожнє сховище.

Для початку, додамо елемент Автор: Іван Франко, Рік видання: 2020 до нашого сховища. Для цього оберемо автора, рік видання та натиснемо кнопку ОК. Після цього отримуємо повідомлення «Збережено у сховищі», яке зображено на Рисунку 1. Я додав цей елемент ще раз, щоб впевнитись у тому, що навіть однакові елементи будуть додаватись стільки раз, скільки захоче користувач.



Рис. 1 – Момент збереження елемента у сховищі

Перейдемо у сховище та подивимось його наповненість. В ньому має бути 2 однакових елемента, які я обрав. Результат зображено на Рисунку 2.

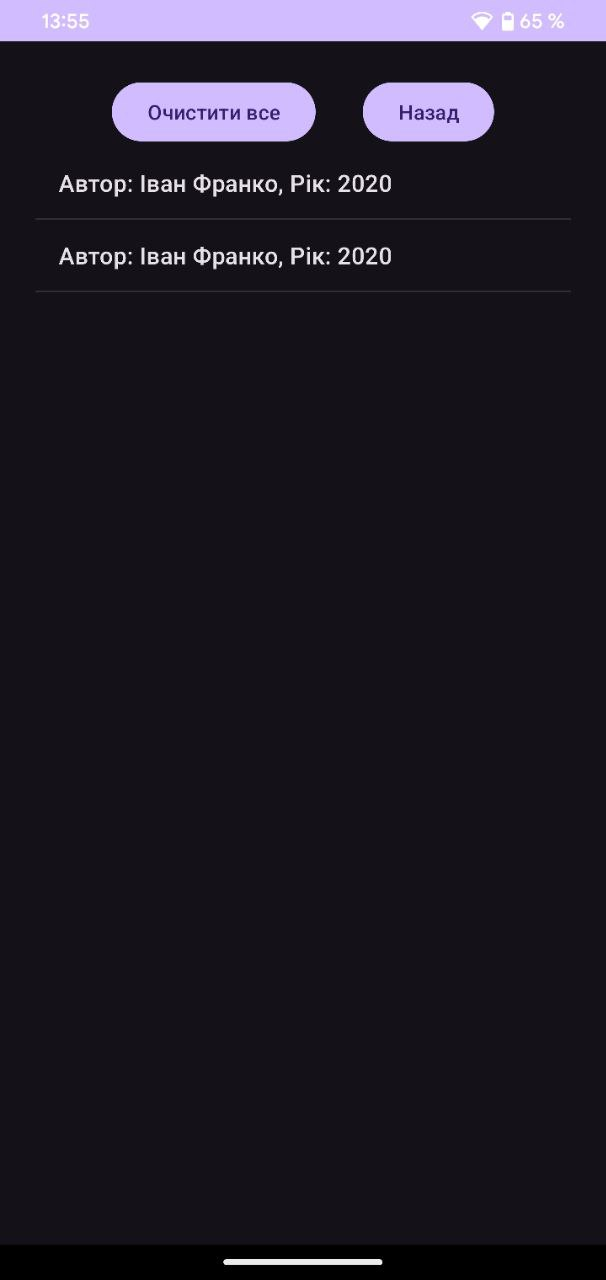


Рис. 2 – Заповнена база даних

Наповненість перевірили, тепер очистимо базу даних та переглянемо як вона виглядає. Для цього натиснемо на кнопку «Очистити все». Результат зображено на Рисунку 3.



Рис. 3 – Очищена база даних

**Висновок**

Під час виконання лабораторної роботи ми дослідити способи збереження даних у базу даних. Вдосконаливши наш проєкт, у порівнянні із другою лабораторною роботою, ми отримали практичні навички щодо використання сховищ даних. Тестування роботи програмного продукту не показало жодних проблем чи помилок, результат виправдовує очікування та задовольняє і виконує завдання лабораторної роботи.

**Контрольні питання**

1. У Android робота з налаштуваннями у вигляді пари «ключ–значення» організована за допомогою класу SharedPreferences. Цей механізм дозволяє зберігати прості дані, такі як логін користувача або мовні налаштування, у приватному сховищі додатку. Для запису даних використовується метод редагування, в якому задаються ключ і значення, а потім викликається метод apply або commit. Для читання даних необхідно вказати ключ і, за потреби, значення за замовчуванням.
2. У Android існує кілька типів файлових сховищ, які використовуються залежно від типу і призначення даних. Внутрішнє сховище призначене для зберігання приватних файлів, до яких має доступ лише сам застосунок. Такі файли видаляються разом із деінсталяцією додатку. Зовнішнє сховище, наприклад SD-карта або виділений розділ у пам’яті пристрою, підходить для зберігання великих файлів, таких як фото чи відео, і вимагає спеціальних дозволів. Також є кеш-сховище, яке використовується для тимчасового зберігання даних, що можуть бути автоматично видалені системою за потреби.
3. Робота з файлами у Android здійснюється за допомогою спеціальних методів та API. Для запису файлу у внутрішнє сховище застосовується метод openFileOutput, а для читання – openFileInput. Також можна працювати із зовнішнім сховищем через метод getExternalFilesDir. Клас File дозволяє створювати, читати, перейменовувати та видаляти файли. У випадку доступу до зовнішнього сховища обов’язково потрібно враховувати наявність відповідних дозволів у маніфесті додатку.
4. Клас SQLiteOpenHelper використовується для організації роботи з локальними базами даних SQLite. Щоб реалізувати БД, потрібно створити підклас, у якому перевизначаються методи onCreate для створення структури бази та onUpgrade для її оновлення. Доступ до БД здійснюється через об’єкт SQLiteDatabase, з використанням SQL-запитів. Основними перевагами такого підходу є легкість впровадження, вбудованість у платформу Android та повна підтримка SQL. До недоліків належить необхідність писати запити вручну, відсутність перевірки синтаксису SQL на етапі компіляції та ускладнення підтримки великих проектів.
5. Для роботи з базами даних у Android також можна використовувати бібліотеку Room, яка є обгорткою над SQLite. Вона базується на об’єктно-реляційному підході й дозволяє створювати сутності, описувати доступ до даних через інтерфейси DAO та вказувати конфігурацію бази у спеціальному класі з анотацією @Database. Серед переваг Room можна відзначити зменшення кількості SQL-коду, перевірку запитів на етапі компіляції, а також інтеграцію з LiveData, Flow та корутинами. Проте до недоліків можна віднести складність налаштування на початку та деякі обмеження при реалізації складних запитів.
6. Характеристики екранів мобільних пристроїв включають такі параметри, як роздільна здатність, яка визначає кількість пікселів по горизонталі та вертикалі, щільність пікселів (вимірюється у ppi або dpi), розмір діагоналі в дюймах, тип матриці дисплея, а також показники яскравості, контрастності та точності кольорів.
7. Серед основних типів екранів мобільних пристроїв можна виділити LCD, OLED та AMOLED. Екрани типу LCD (зокрема IPS) мають гарну видимість при яскравому освітленні та є дешевшими у виробництві, однак споживають більше енергії. OLED та AMOLED екрани забезпечують вищу контрастність, насиченість кольорів і менше енергоспоживання, адже кожен піксель світиться окремо. Super AMOLED — це вдосконалена версія AMOLED, яка відрізняється ще тоншою конструкцією та покращеною якістю зображення.
8. Сенсорний екран — це тип дисплея, який реагує на дотик пальця, стилуса або іншого об’єкта. Він дозволяє взаємодіяти з інтерфейсом пристрою напряму, без використання додаткових апаратних засобів введення. Основними характеристиками сенсорного екрана є чутливість до натиску, точність визначення координат дотику, а також підтримка жестів і мультидотику.
9. Сенсорні екрани загалом класифікують на резистивні та ємнісні. Резистивні екрани реагують на фізичний тиск, тому їх можна активувати пальцем, стилусом або будь-яким іншим предметом. Вони є дешевшими, але мають нижчу чутливість. Ємнісні екрани, натомість, реагують на електричне поле тіла людини, що забезпечує високу точність, чутливість і підтримку мультидотику, однак вони не працюють із звичайними предметами без спеціального покриття.
10. При розробці інтерфейсів для сенсорних екранів варто дотримуватись певних рекомендацій. Зокрема, елементи управління повинні бути достатньо великими, щоб їх було зручно натискати пальцями. Інтерфейс повинен бути простим, логічним і зрозумілим користувачеві без потреби у додатковому навчанні. Слід мінімізувати введення тексту та враховувати можливість зміни орієнтації екрана. Розробники повинні використовувати адаптивний дизайн для коректного відображення на різних пристроях, а також впроваджувати підтримку жестів та плавних анімацій для покращення взаємодії з користувачем.