

**Міністерство освіти і науки України**  
**Національний технічний університет України «Київський політехнічний**  
**інститут імені Ігоря Сікорського»**  
**Факультет інформатики та обчислювальної техніки**

**Кафедра ІІІ**

**ЗВІТ**

**Лабораторної роботи № 3**

з дисципліни: «Розробка мобільних застосунків під Android»

**“ДОСЛІДЖЕННЯ СПОСОБІВ ЗБЕРЕЖЕННЯ ДАНИХ”**

Варіант 5

**Виконала**

ІП-32 Зелюк Ю.М.  
(шифр, прізвище, ім'я, по батькові)

**Перевірив**

Орленко С. П.  
(прізвище, ім'я, по батькові )

## **1 Мета**

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

## **2 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 порожнє) відобразити відповідне повідомлення. За бажанням можна додатково реалізувати оновлення та видалення даних зі сховища.

### 3 Виконання

Для реалізації роботи зі сховищем спершу я використовувала звичайне збереження в текстовий файл, дані з полів зберігались в один довгий рядок у режимі `MODE_APPEND`. Під час тестування, я вирішила додати можливість видаляти записи, «скасовувати» замовлення, але реалізований підхід давав тільки видаляти або редагувати всі записи одразу. Для вирішення цієї проблеми та подальшої можливості кастомізації застосунку далі був обраний формат JSON, кожне замовлення тепер було об'єктом з окремим `id`. Для такого підходу було також необхідно якось перетворювати об'єкти для запису для чого було підключено бібліотеку `gson`, яка виконує задачу серіалізації та десеріалізації. Для уникнення можливої втрати даних при додаванні нових замовлень в `firstfragment` були додані такі зміни: програма зчитує весь масив з файлу сховища, в цей масив у пам'яті додається новий об'єкт/оновлюється існуючий, оновлений масив повністю перезаписується, таким чином історія замовлень залишеться актуальною і не містить дублікатів. Повертаючись до питання редагування замовлення, об'єкт замовлення розбивається на параметри, що через `intent` передаються назад на головну форму для подальших змін. Користувач не має вводити інформацію по замовленню, яку вже вносив а завдяки наявності `id` система просто змінює старе. Також крім змін функціоналу було змінено і дизайн(стандартизована біла тема, аби на телефонах з різною стандартною темою однаково відображалось), тестування проводилось на моєму телефоні Samsung a34. По функціоналу: початковий екран – створення замовлення або перегляд попередніх замовлень, на сторінці перегляду історії можемо редагувати або видаляти записи, після того як створено 2 і більше замовлень – можна видалити всі замовлення одразу. На майбутнє можна буде додати перевірку чи апрувнуто замовлення адміністратором, і коли це вже відбулось то видалити або редагувати замовлення не вийде.

## Початкова робота програми:

Flower Order Form

Lilies

Yuliia

juliazeluk@gmail.com

Additional notes

Choose color

☐ Red

☒ White

☐ Pink

☐ Blue

Choose package

☐ Basic (under \$15)

☒ Standard (\$15 - \$50)

☐ Premium (above \$50)

Confirm Order

Open History

ORDER SUMMARY

Flower: Lilies

Customer: Yuliia

Color: White

Package: Standard (\$15 - \$50)

Notes: None

Create New Order

Order History

ORDER SUMMARY

Flower: Lilies

Customer: Yuliia

Color: White

Package: Standard (\$15 - \$50)

Notes: None

Clear History

Order saved to file!

Order History

ORDER SUMMARY

Flower: Lilies

Customer: Yuliia

Color: White

Package: Standard (\$15 - \$50)

Notes: None

ORDER SUMMARY

Flower: Peonies

Customer: Yuliia

Color: White

Package: Premium (above \$50)

Notes: For a friend. Address \*\*\*\*\*

Clear History

Order History

History cleared.

Clear History

All records deleted!

## Фінальний результат:

## Flower Order Form

Peonies

Kostya

flowers@gmail.com

Additional notes

Choose color

☐ Red

☐ White

☒ Pink

☐ Blue

Choose package

☐ Basic (under \$15)

☒ Standard (\$15 - \$50)

☐ Premium (above \$50)

CONFIRM ORDER

OPEN HISTORY

## Order History

Peonies  
bda7e4c6

EDIT

DEL

2nd Order  
2f0bc6a7

EDIT

DEL

Flower: Peonies  
Customer: Kostya  
Color: Pink  
Package: Standard (\$15 - \$50)  
Notes: None

CREATE NEW ORDER

Order added!

## Flower Order Form

Update 2nd Order

test

test@test.com

Just test, im updating test. Look at the  
flower name

Choose color

☐ Red

☒ White

☐ Pink

☐ Blue

Choose package

☐ Basic (under \$15)

☐ Standard (\$15 - \$50)

☒ Premium (above \$50)

UPDATE ORDER

OPEN HISTORY

NEW ORDER

CLEAR ALL

## Order History

Peonies  
bda7e4c6

EDIT

DEL

Попереднє тестування

Deleted successfully

NEW ORDER

## Order History

Peonies  
bda7e4c6

EDIT

DEL

Update 2nd Order  
2f0bc6a7

EDIT

DEL

NEW ORDER

CLEAR ALL

#### Order history

Peonies  
bda7e4c6

EDIT

DEL



## ВИСНОВОК

Під час виконання роботи було реалізовано перехід від простого текстового файлу до структурованого JSON-сховища за допомогою бібліотеки GSON. Це дозволило впровадити повний цикл керування даними: редагування замовлень через передачу об'єктів у Intent та точкове видалення записів за їхнім id.

Інтерфейс було адаптовано під стандартизовану світлу тему для стабільного відображення на різних пристроях (тестовано на Samsung A34), а також додано динамічні елементи UX, такі як кнопка масового видалення при наявності декількох записів. Проєкт створює надійну базу для подальшого впровадження системи статусів та адміністративного доступу.

### Відповіді на питання:

1. Робота з налаштуваннями організована через SharedPreferences, де дані зберігаються в XML-файлі як пари «ключ-значення» та змінюються за допомогою об'єкта Editor.
2. Файли зберігаються у внутрішній пам'яті для приватності додатка або у зовнішній, якщо дані мають бути доступні користувачу та іншим програмам.
3. Процес передбачає отримання шляху до директорії, відкриття потоку введення-виведення для читання або запису байтів та обов'язкове закриття ресурсів.

4. SQLiteOpenHelper створює таблиці через прямі SQL-запити, що дає повний контроль над базою, але вимагає багато шаблонного коду та ручної обробки помилок.
5. Room використовує анотації для автоматичної генерації коду доступу до даних, що спрощує розробку та мінімізує помилки, проте дещо сповільнює збірку проєкту. В цьому проєкті я не використовувала її, оскільки зручніше та практичніше бо кількість зв'язків між даними невелика.
6. Основними характеристиками є фізичний розмір діагоналі, роздільна здатність у пікселях та щільність точок на дюйм (DPI).
7. Екрани поділяються на LCD, що потребують підсвітки, та OLED, де кожен піксель світиться окремо, забезпечуючи ідеальний чорний колір і вищу контрастність.
8. Сенсорний екран — це пристрій введення, який поєднує функції відображення та реєстрації дотиків пальців або стилуса для взаємодії з інтерфейсом.
9. Загальна класифікація поділяє екрани на резистивні, що реагують на тиск, та ємнісні, які спрацьовують від електричного потенціалу тіла.
10. Рекомендується створювати активні елементи розміром не менше 48x48 dp, забезпечувати достатні відступи між ними та додавати візуальний відгук на натискання.