Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет Інформатики та Обчислювальної Техніки

Кафедра інформаційних систем та технологій

Лабораторна робота №2

з дисципліни «Розробка мобільних застосувань під Android»

на тему

«ДОСЛІДЖЕННЯ РОБОТИ З КОМПОНЕНТОМ FRAGMENT»

Виконав:

студент групи ІС-21

Качалов А. М.

Викладач:

Орленко С. П.

Київ – 2025

**Мета роботи:** дослідити створення та взаємодію з компонентом Фрагмент

(Fragment) компоненту Діяльність та набути практичні навички з використання

фрагментів для інтерфейсу користувача.

**ЗАВДАННЯ**

Написати програму під платформу Андроїд, яка має інтерфейс, побудований з декількох фрагментів згідно варіанту. Перший фрагмент представляє з себе форму для введення даних та кнопку підтвердження («OK»), а інший фрагмент відображає результат взаємодії. Тобто другий фрагмент містить тестове поле з результатом та кнопкою «Cancel» (якщо згідно варіанту така існує, якщо ж за варіантом її немає – можете додати за власним бажанням), яка очищає або приховує (або видаляє) другий фрагмент та очищає форму введення з першого фрагменту. Зверніть увагу, що робота з фрагментами відбувається в рамках однієї Діяльності.

Примітка: завдання відповідає варіанту лабораторної роботи No1.

**Хід роботи**

**1. Програмна реалізація**

Оскільки завдання другої лабораторної роботи основане на результатах першої, продовжимо працювати із уже наявним проєктом. Розділимо фрагменти у Layout за сенсом:

1). Fragment\_input – файл із першим фрагментом;

2). Fragment\_result – файл із другим фрагментом.

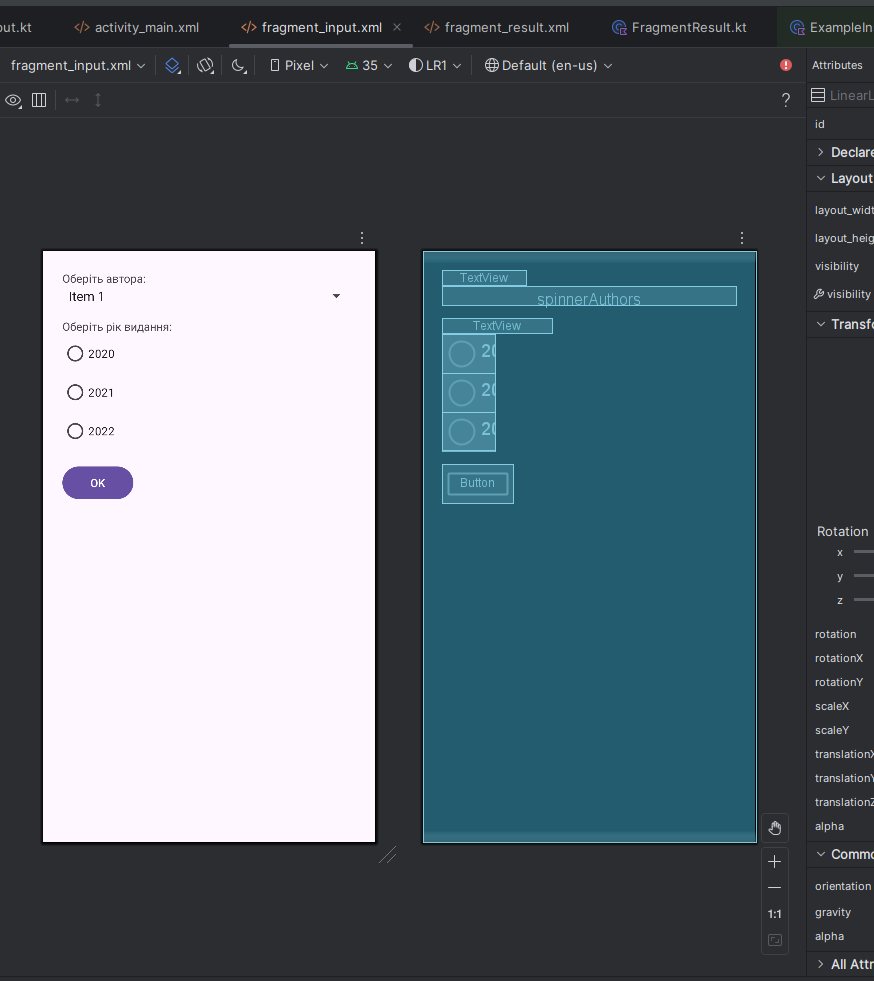


Рис. 1 – Вигляд першого фрагменту інтерфейсу

Linear Layout майже повністю (окрім нижнього TextView) перейшов з першої лабораторної роботи у перший фрагмент другої.

Для другого фрагменту було створено ідентичний за стилем Layout до першого, додано до нього ТextView, в якому виводитиметься результат, а також кнопку «Скасувати» (Cancel), яка очищає форму.

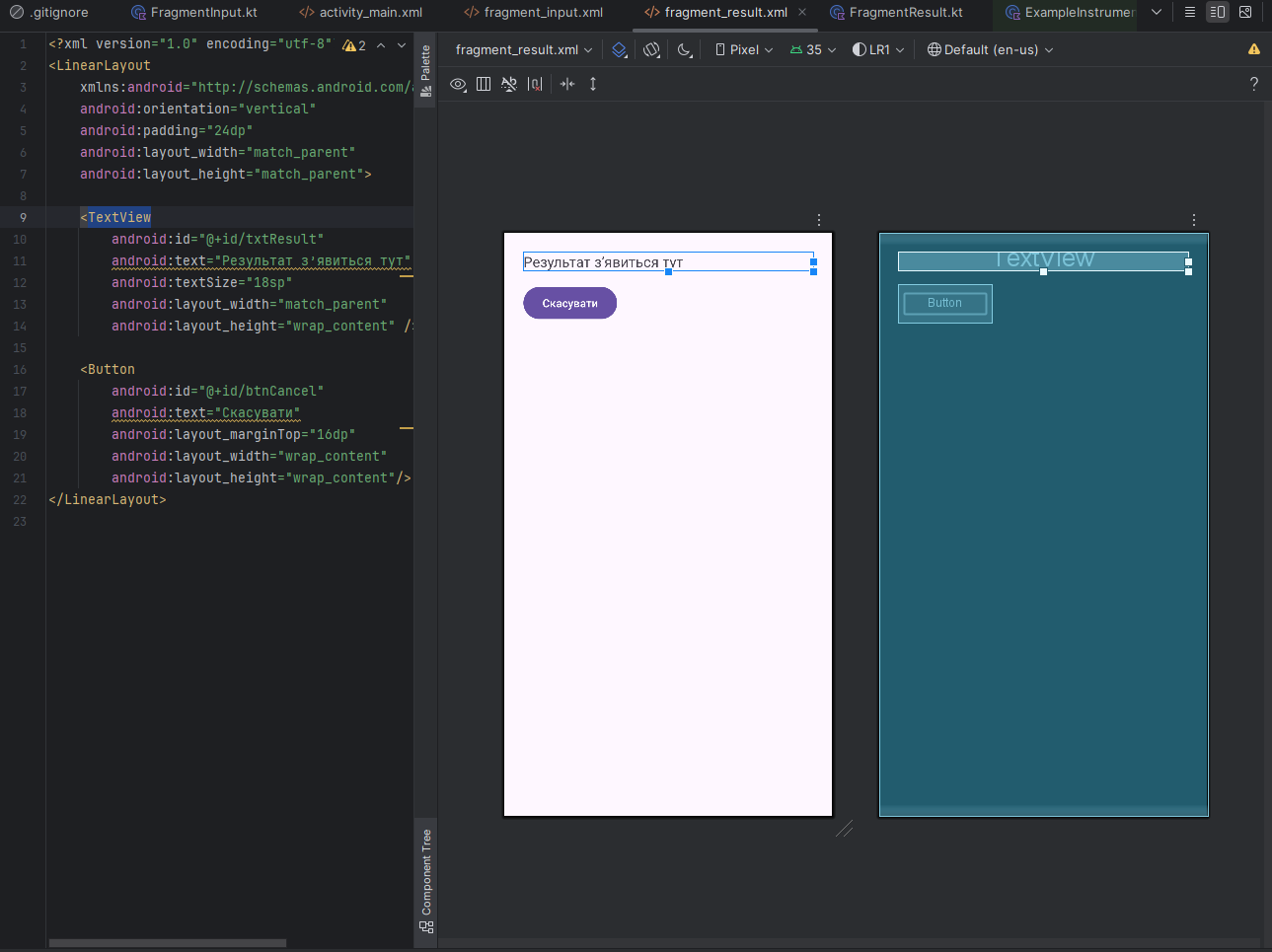


Рис. 2 – Вигляд другого фрагменту інтерфейсу

Файл FragmentInput відповідає за фрагмент введення даних, тобто перший за рахунком у лабораторній роботі. В ньому ми оголошуємо клас як фрагмент (а не як основну активність), оголошуємо змінні, перевіряємо прикріплення інтерфейсу, створюємо та відображуємо UI та очищаємо форму в кінці.

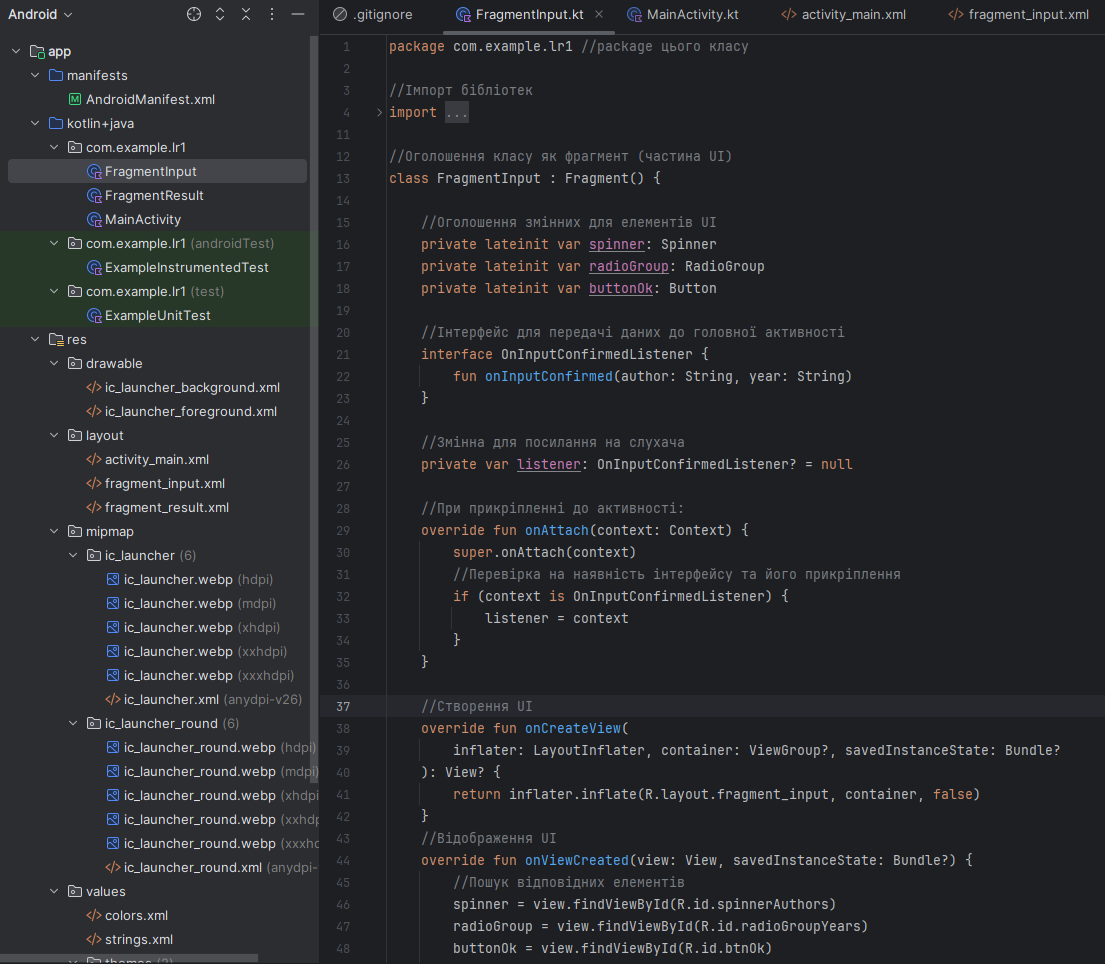


Рис. 3 – Код фрагменту «FragmentInput»

В свою чергу, файл FragmentResult відповідає за фрагмент виведення даних, другий за рахунком у лабораторній роботі. В ньому ми також оголошуємо: клас як фрагмент, змінні для елементів UI, інтерфейс передачі даних до основної активност та змінну для посилання на слухача. Далі створюємо сам UI, даємо йому відображення та очищаємо форму.

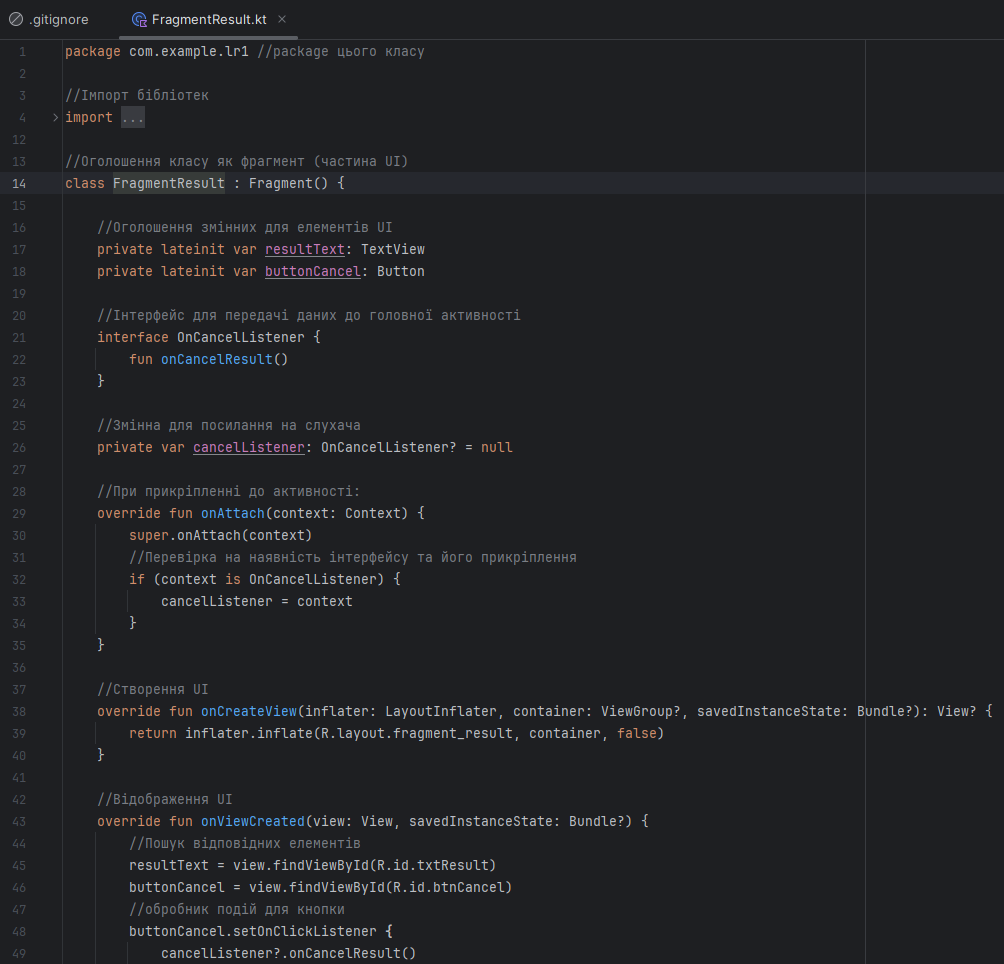


Рис. 4 – Код фрагменту «FragmentResult»

У файлі MainActivity спочатку створюємо клас з елементами AppCompatActivity(), оголошуємо змінну для збереження фрагменту. В метод при створенні активності входять: підключення підготовки інтерфейсу, підлючення XML-розмітки UI, створення фразменту для виведення даних та створення цього фрагменту на екрані. Також в цій активності знаходиться метод при натисканні кнопки ОК, в якому є виведення заміненого фрагменту та редагування тексту. В кінці файла знаходиться обробник кнопки.

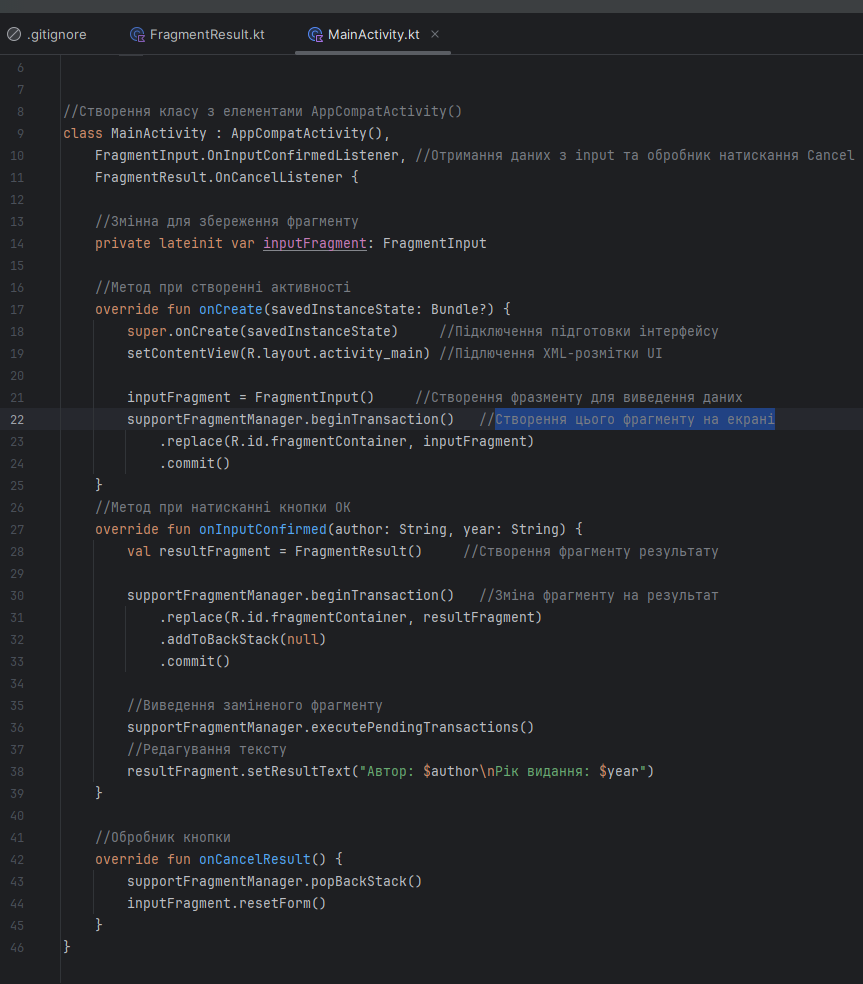
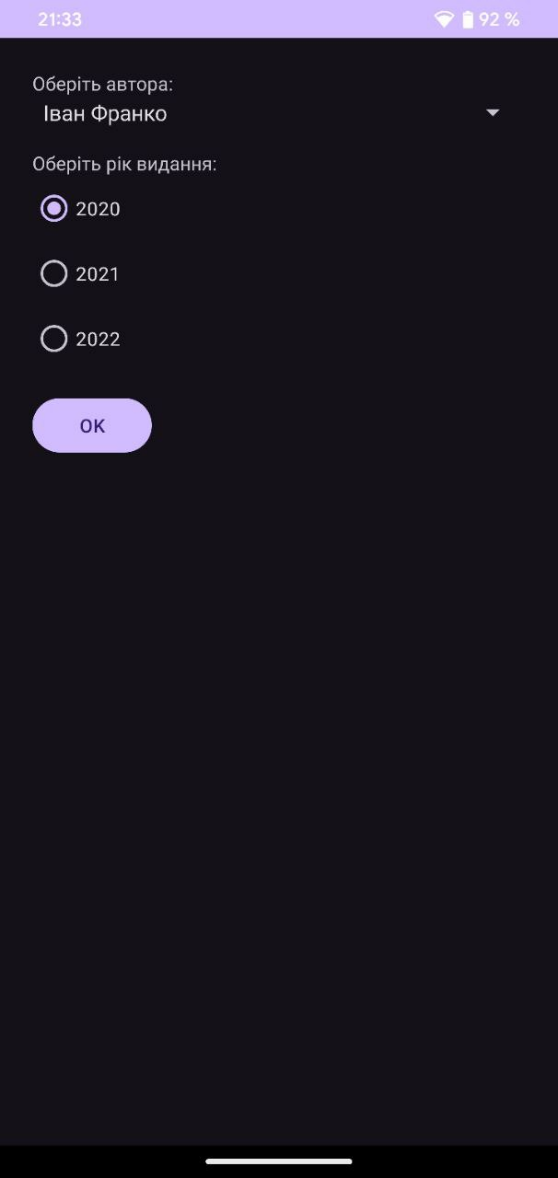
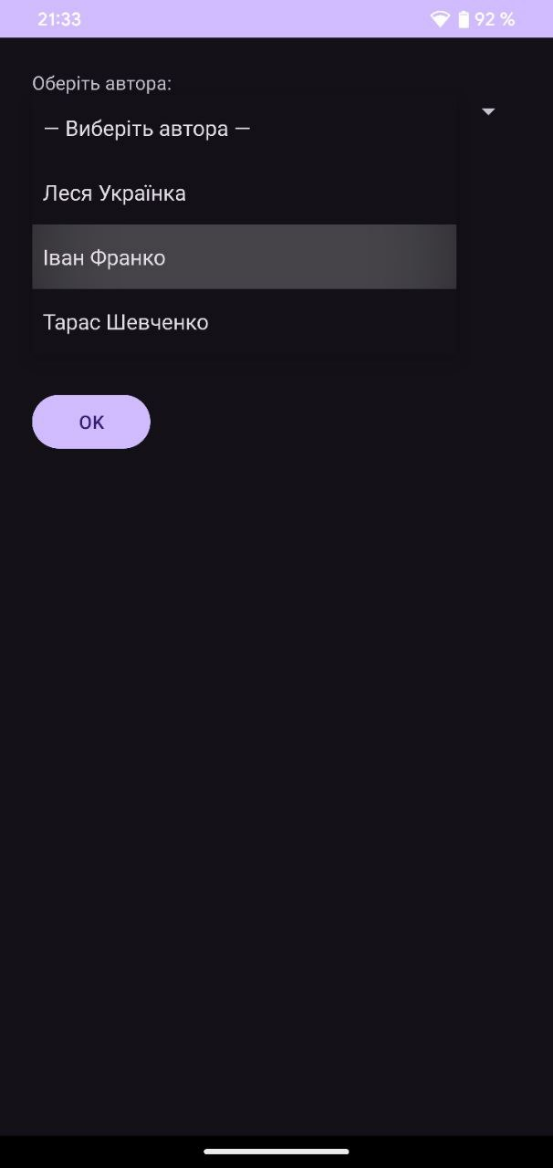
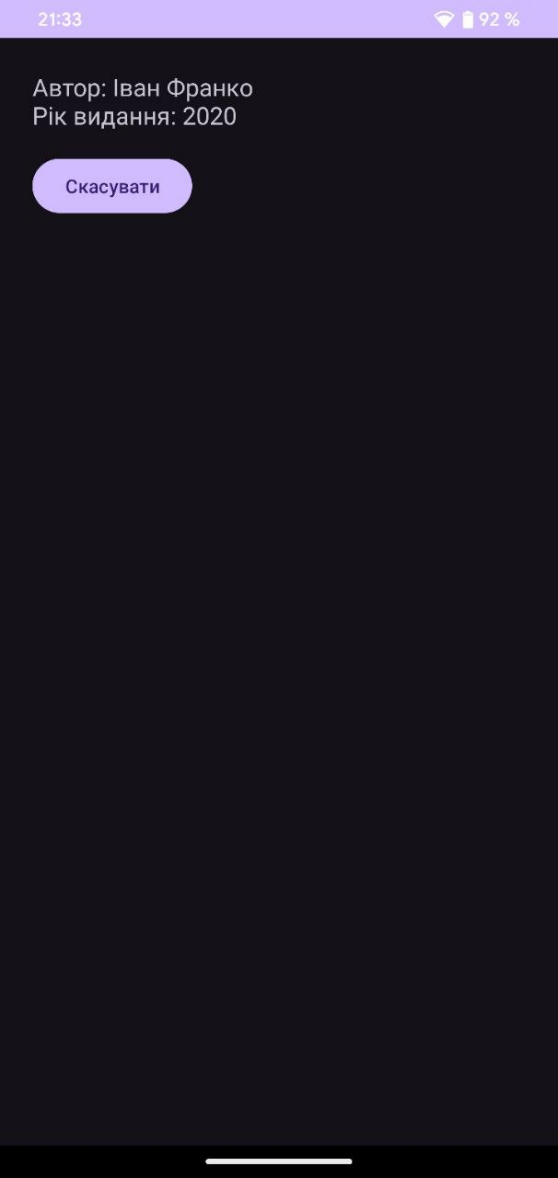


Рис. 5 – код діяльності MainActivity

**2. Робота програмного продукту**

Під час перевірки було запущено apk-файл та протестовано всі заявлені в лабораторній роботі задачі на предмет їх реалізації.



Перевірка показала успішне виконання та реалізацію всіх фіч: програмний продукт із 1 лабораторної роботи + другий екран із кнопкою скасування. Обидва фрагменти відбуваються в рамках однієї діяльності.

**Висновок**

У ході виконання лабораторної роботи було досліджено особливості створення та використання компоненту Фрагмент у межах компонента Діяльність. Вдосконалення, у порівнянні із першою лабораторною роботою, дали змогу закріпити теоретичні знання щодо життєвого циклу фрагментів, способів їх створення та взаємодії. В результаті, отримано навички побудови гнучкого й адаптивного інтерфейсу користувача, що є важливим етапом у розробці сучасних мобільних застосунків.

**КОНТРОЛЬНІ ПИТАННЯ**

1. Компонент Фрагмент у мобільній розробці використовується для поділу інтерфейсу на окремі частини, які можна зручно комбінувати або повторно використовувати в межах одного чи кількох вікон (Activity). Він дозволяє створювати гнучкіші й адаптивніші інтерфейси, особливо для пристроїв із різними розмірами екранів.

2. Життєвий цикл Фрагмента схожий на життєвий цикл Вікна, але має свої особливості. Він проходить етапи створення, прив’язки до вікна, відображення, паузи, зупинки та знищення. На кожному з етапів можна реалізувати окрему логіку, наприклад, завантаження даних або очищення ресурсів.

3. Фрагмент можна створити двома основними способами: у коді вручну, використовуючи менеджер фрагментів, або за допомогою XML-розмітки, де він описується як частина інтерфейсу. У кожному з підходів важливо правильно ініціалізувати й додати фрагмент у структуру програми.

4. Керування Фрагментами здійснюється через спеціальний об'єкт — менеджер фрагментів, який дозволяє додавати, замінювати, приховувати або видаляти фрагменти під час виконання програми. Це забезпечує гнучкість у формуванні інтерфейсу залежно від дій користувача або умов.

5. Взаємодія між Фрагментами зазвичай відбувається через Вікно, яке їх утримує. Один фрагмент може передавати дані іншому через загальний об’єкт або виклики методів через інтерфейси. Це дозволяє забезпечити незалежність окремих частин інтерфейсу при збереженні можливості їх взаємодії.

6. Система — це сукупність взаємопов’язаних компонентів, які працюють разом для досягнення певної мети. Мала система — це система з обмеженою кількістю ресурсів або функцій, що часто зустрічається в мобільних пристроях. Мобільна платформа — це програмно-апаратне середовище, яке дозволяє створювати, запускати та використовувати мобільні застосунки (наприклад, Android або iOS).

7. Існують такі типи мобільних застосунків: нативні — створені для конкретної платформи, веб-застосунки — що працюють через браузер, та гібридні — поєднують особливості попередніх типів. Кожен із них має свої переваги та обмеження щодо продуктивності, доступу до функцій пристрою та складності розробки.

8. Середовища розробки мобільних застосунків можна класифікувати як офіційні (наприклад, Android Studio для Android або Xcode для iOS) та кросплатформні (наприклад, Flutter або React Native). Офіційні середовища забезпечують повну підтримку платформи, тоді як кросплатформні дозволяють створювати застосунки одночасно для кількох платформ з єдиною кодовою базою.

9. Мобільні платформи класифікують за типом операційної системи: Android, iOS, HarmonyOS та інші. Вони характеризуються своєю архітектурою, мовами програмування, засобами безпеки, магазином застосунків та інструментами для розробки. Кожна платформа встановлює власні правила й обмеження для створення застосунків.