МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет
Центр заочної, дистанційної та вечірньої форм навчання
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ
(підпис)

грудень 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня магістра

за спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційна технологія для створення освітніх матеріалів та інтерактивних освітніх завдань на базі операційної системи Android»
здобувача освітнього ступеня магістра групи ІН - мз.21с Білоуса Владислава Олександровича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Білоус В.О.
(підпис)

Керівник, Зарецький М.О.
Доктор філософії

_____
(підпис)

Суми - 2023

Сумський державний університет
Центр заочної, дистанційної та вечірньої форм навчання
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістра

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІН.мз-21с

1. Тема роботи: «Інформаційна технологія для створення освітніх матеріалів та інтерактивних освітніх завдань на базі операційної системи Android» затверджую наказом по СумДУ від *«20» листопада 2023 р. № 1308-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 13 грудня 2023 року*
3. Вхідні дані до кваліфікаційної роботи
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз наявних систем 2) Вибір програмних засобів реалізації. 3) Програмна реалізація стстеми.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «28» 09 2023 р.

Завдання прийняв до виконання _____ (підпис) Керівник _____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми та актуальності розробки, постановка й формування завдань*	09.10.2023	Виконано
2	*Вибір програмних засобів для реалізації системи. Розробка її архітектури.*	14.10.2023	Виконано
3	*Розробка системи*	04.11.2023	Виконано
4	*Аналіз проведеної роботи. Оформлення її результатів*	09.11.2013	Виконано

Здобувач вищої освіти _____ (підпис) Керівник _____ (підпис)

АНОТАЦІЯ

Записка: 60 стр., 24 рис., 5 додаток, 28 використаних джерел.

Обґрунтування актуальності теми роботи – тема є досить актуальною оскільки охоплює актуальні на даний момент часу проблеми пов’язані з гнучкістю, мобільністю та доступністю освіти. Тема охоплює новаторські підходи до освітнього процесу.

Об’єкт дослідження — процес створення та використання інтерактивних освітніх завдань у сфері освіти

Мета роботи — розробка системи для створення освітніх матеріалів та інтерактивних освітніх завдань

Методи дослідження — аналіз потреб користувачів та доступних аналогів. Проектування системи та її реалізація.

Результати — розроблено інформаційну систему яка дозволяє створювати, редагувати та працювати з освітніми матеріалами та завданнями до них.

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ, ANDROID, МОБІЛЬНИЙ ДОДАТОК, ОСВІТНІ МАТЕРІАЛИ, МЕТОДИ ОСВІТИ

ЗМІСТ

ВСТУП

Актуальність та обгрунтування вибору теми роботи. Сьогодні мобільні пристрої є надзвичайно популярними та доступними для широкого кола користувачів. Все більше людей вже не можуть уявити світ без мобільних пристроїв. Сучасні мобільні телефони є комплексними системами за допомогою яких можна вирішити багато різноманітних завдань. В тому числі також завдання освітнього процесу. В цілому, розробка програмного забезпечення для мобільних платформ на даний час є надзвичайно актуальною та з кожним роком тільки набирає популярність. Серед мобільних систем саме Android система є найбільш розповсюдженою у світі, та найбільш доступною для ширикого кола користувачів.

Сучасні освітні підходи включають в себе все більше інтерактивності та практичних завдань. Інтерактивні освітні завдання дозволяють студентам активно взаємодіяти з матеріалом та отримувати більше практичного досвіду.

Створивши гнучку технологію для створення інтерактивних завдань ми можемо адаптувати завдання до потреб конкретного користувача. Це дозволить навчанню стати більш індивідуалізованим і ефективним. Також навчаючись з мобільного пристрою користувач зможе навчатися в зручному для себе режимі в будь-який момент часу знаходячись в будь-якому місці.

Система Android надає розробнику широкі можливості для створення різноманітних додатків, включаючи й освітні. Розвиток програмування на Android дозволяє створювати потужні та зручні інструменти для навчання. Система Android еволюціонує з кожним днем. Кожного року з’являются нові технології які дозволяють покращувати ефективність додатку та досвід при користуванні ним.

Світова пандемія змусила багато навчальних закладів переходити на дистанційне навчання або шукати альтернативні методи навчання.

Інформаційні системи на базі Android можуть стати ефективними інструментами для підтримки дистанційного навчання та створення інтерактивних освітніх завдань, що допоможе студентам зберігати мотивацію та залученість.

В цілому вся сучасна освіта вимагає використання більш новітніх методів навчання задля збільшення своєї ефективності. Інтерактивні освітні завдання допомагають створювати умови, які сприяють засвоєнню знань.

Також подібний додаток допоможе безпосередньо авторам освітнього контенту та вчителям, які зі своєї сторони шукають засоби для створення власних інтерактивних завдань без необхідності глибокого знання програмування та розробки додатків.

Також розробка подібної технології може стати конкурентною перевагою на ринку навчання для освітніх закладів або платформ для навчання, які прагнуть надати якісний та інтерактивний контент своїм користувачам.

Загалом, тема має важливий суспільний та освітній контекст, і розробка інформаційної технології для створення освітніх матеріалів та інтерактивних освітніх завдань на базі Android має потенціал внести значний внесок у сучасну освіту.

Об’єкт дослідження. Об'єктом дослідження є процес створення та використання інтерактивних освітніх завдань у сфері освіти.

Предмет дослідження. Предметом дослідження є інформаційна технологія, розроблена на базі операційної системи Android, яка дозволяє користувачам створювати, зберігати та використовувати інтерактивні освітні завдання. Дослідження також охоплює модуляцію процесу навчання з використанням цієї технології, а також її потенціальний вплив для покращення якості освіти та результативності користувачів.

Метою даної роботи є розробка та впровадження інформаційної технології на платформі Android, яка надає користувачам можливість

створювати інтерактивні освітні завдання, сприяючи покращенню процесу навчання та підвищенню доступності освіти через мобільні пристрої.

Завдання ж можна виділити наступні:

- Аналіз потреб користувачів: Вивчення потреб освітніх установ, вчителів та студентів при створенні інтерактивних освітніх завдань;
- Огляд існуючих рішень: Проведення огляду наявних додатків та платформ для створення освітніх завдань з метою виокремлення сильних та слабких сторін цих рішень;
- Проектування інформаційної технології: Розробка дизайну та архітектури інформаційної технології, визначення основних функцій та можливостей технології для створення інтерактивних освітніх завдань;
- Реалізація програмного забезпечення: Розробка інформаційної технології на платформі Android, включаючи реалізацію інтерфейсу користувача та модулів для створення та збереження освітніх завдань;
- Тестування та валідація: Проведення тестів та валідації розробленої технології для перевірки її працездатності та відповідності вимогам користувачів;
- Оцінка ефективності: Аналіз впливу розробленої технології на процес навчання та оцінка її ефективності.

Новизна. Розробка технології інтерактивного навчання на базі мобільних пристроїв є досить новаторськьою темою для дослідження та розробок. Методи розробки сучасних мобільних додатків динамічно розвиваються та змінюються покращуючи свою єфективність. Тому подібна роботи є досить акутальною на даний момент часу.

Саме ж дослідження спрямоване на поєднання двох актуальних на даний момент часу тенденцій - інтерактивного онлайн навчання та використання мобільних пристроїв у навчанні.

Технологія має на меті зробити створення інтерактивних завдань більш доступним для вчителів та авторів навчального контенту, що є досить новаторським підходом до покращення освіти.

Дослідження може дати можливість оцінити, наскільки використання інтерактивних освітніх завдань на Android впливає на результати навчання та мотивацію студентів.

Розроблена технологія може бути використана в навчальних закладах для покращення навчального процесу, що має важливе значення для освіти в цифровому віці.

Розроблена інформаційна технологія може дозволити вчителям та авторам навчального контенту створювати інтерактивні освітні завдання, які сприятимуть покращенню процесу навчання та засвоєнню знань студентами.

Використання інтерактивних завдань на мобільних пристроях може зробити навчання цікавішим і залучаючим, що підвищить мотивацію студентів до вивчення предмету.

Технологія розроблена для платформи Android, яка має велику популярність. Це робить освітні ресурси більш доступними для користувачів з різних соціокультурних та географічних контекстів.

Технологія дозволяє вчителям та авторам навчального матеріалу створювати інтерактивні завдання без спеціалізованих навичок у програмуванні, що дозволить останнім зосередитись безпосередньо на створенні навчального контенту приділивши йому всю можливу увагу. Також створення навчальних матеріалів стане більш доступним для широкого кола вчителів та авторів.

Розроблена технологія впроваджує сучасні технології в освітній процес, що сприяє розвитку цифрової компетентності студентів та вчителів.

В даній роботі можна виокремити наступні методи дослідження:

- Аналіз літератури та попередніх досліджень: В даному методі був проведений аналіз наукових статей, книг, онлайн-ресурсів та попередніх досліджень, щоб отримати глибоке розуміння теоретичних та практичних аспектів створення інтерактивних освітніх завдань та розробки додатків для Android.
- Огляд існуючих рішень: Також були проаналізовані існуючі технологічні рішення та платформи для створення інтерактивних освітніх завдань для визначення їх сильних та слабких сторін.
- Проектування технології: Для розробки технології були використані методи системного проектування, включаючи розробку архітектури системи, проектування сценаріїв використання системи, розробка дизайну інтерфейсу користувача та вибір технологій розробки.
- Розробка програмного забезпечення: Ця робота включала розробку програмного забезпечення для операційної системи Android. Було використано інтегроване середовище розробки Android Studio та мову програмування Kotlin для реалізації технології. Також була використана низка фреймворків для роботи з даними та створення інтерфейсу користувача.
- Тестування та валідація: Для перевірки працездатності технології ми використовували методи тестування, включаючи функціональне тестування та валідацію результатів відповідно до потенційних вимог користувача.
- Аналіз результатів: Отримані результати були проаналізовані для оцінки досягнення поставлених цілей та визначення можливостей подальшого розвитку технології.

1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Аналіз наявних мобільних додатків для навчання

Мобільні додатки для навчання на Android можуть включати в себе текстові матеріали, відеоуроки, інтерактивні завдання, тести та інші освітні ресурси. [15]

Деякі додатки надають можливість персоналізувати навчальний процес, дозволяючи студентам обирати теми, рівні складності та темп навчання.

Деякі додатки створюють можливість для взаємодії та співпраці між користувачами, що сприяє колективному навчанню та обміну знаннями.

Деякі додатки надають можливість вчителям або студентам відстежувати прогрес у навчанні та отримувати звіти про успішність. [5]

Мобільні додатки інтегрують різноманітні освітні ресурси, роблячи процес навчання більш доступним і цікавим. Текстові матеріали, відеоуроки, інтерактивні завдання та тести розширюють спектр можливостей для отримання знань.

Крім того, існують додатки, які активно пропагують взаємодію та співпрацю між користувачами. Це розвиває колективне навчання та обмін знаннями, що сприяє розширенню розуміння теми через спільну діяльність. [25]

Таким чином, якісний продукт на освітньому ринку повинен об'єднувати ці переваги, забезпечуючи комплексний та інтерактивний підхід до навчання.

1.2 Аналіз існуючих інструментів та платформ для створення інтерактивних освітніх завдань

На сьогоднішній день існують різні платформи та інструменти для створення інтерактивних завдань, включаючи:

- Learning Management Systems (LMS): Онлайн-платформи, які дозволяють вчителям створювати та керувати навчальними завданнями, тестами та спостерігати за прогресом студентів.
- Електронні платформи для створення вправ і тестів: Інструменти, які надають можливість створювати різноманітні типи вправ, тестів та інтерактивних завдань.
- Інструменти для створення мультимедійного контенту: Програмне забезпечення для створення відеоуроків, інтерактивних презентацій та іншого мультимедійного контенту [3].

Аналізуючи інструментарій на різноманітних освітніх платформах можна виділити наступні інструменти та особливості взаємодії з платформами:

- Візуальний редактор: Багато платформ мають візуальний редактор, який дозволяє користувачам створювати завдання за допомогою перетягування та встановлення елементів.
- Інтерактивність: Інструменти можуть надавати можливість додавати інтерактивні елементи, такі як питання з вибором, відкриті питання, відео, аудіо тощо.
- Зручний інтерфейс для користувачів: Інструменти повинні бути легкими у використанні, щоб навіть вчителі без технічного досвіду могли створювати інтерактивні завдання.
- Можливість відстеження прогресу: Деякі інструменти надають можливість вчителям або авторам відстежувати прогрес студентів, що допомагає в оцінці навчальних результатів. [24]

Аналізуючи вище вказані інструменти та весь підхід в цілому до подібного створення навчальних завдань можна виділити наступні переваги та недоліки подібного підходу

Переваги

- Зручність та ефективність у створенні завдань.
- Можливість персоналізації навчального матеріалу.
- Залучення студентів до активного навчання.

Недоліки

- Необхідність підготовки та навчання вчителів.
- Обмеженість функціональності деяких інструментів.
- Необхідність забезпечення доступу до інтернету та сучасних мобільних пристроїв. [11]

Аналіз інструментів для створення інтерактивних освітніх завдань допоможе визначити оптимальний підхід для розробки інформаційної технології.

1.3 Огляд наявного досвіду використання інтерактивних завдань в навчанні

Деякі навчальні заклади та вчителі вже успішно використовують інтерактивні завдання у своїй педагогічній практиці.

Приклади включають:

- Використання мобільних додатків для навчання: Вчителі можуть використовувати мобільні додатки для створення інтерактивних завдань та спостереження за прогресом студентів.
- Ігрові підходи до навчання: Використання ігор та головоломок для активізації критичного мислення та розвитку практичних навичок.
- Використання віртуальної реальності (VR) та розширеної реальності (AR): Навчальні заклади можуть створювати інтерактивні середовища для вивчення наукових дисциплін, мистецтва та багатьох інших предметів. [7]

Використання інтерактивних завдань має позитивний вплив на якість навчання у навчальних закладах а саме вони впливають на:

- Підвищення мотивації студентів: Інтерактивні завдання роблять навчання цікавим та стимулюють активність студентів.
- Підвищення засвоєння матеріалу: Завдяки взаємодії та практичним вправам, студенти краще засвоюють навчальний матеріал.
- Розвиток критичного мислення: Інтерактивні завдання сприяють розвитку критичного мислення, аналізу та розв'язанню проблем.
- Сприяння самодисципліні та саморегуляції: Інтерактивність завдань може стимулювати студентів до більшої самостійності у навчанні та управлінні часом.
- Залучення до спільної роботи: Деякі інтерактивні завдання сприяють спільній роботі студентів, що розвиває навички співпраці та комунікації. [12]

Аналіз досвіду використання інтерактивних завдань в навчанні підкреслює важливість їх впровадження як засобу підвищення якості освіти та залучення студентів до більш активного та ефективного навчання.

1.4 Тенденції у розвитку освіти та технологій

Однією з основних тенденцій у сучасній освіті є зростання ролі технологій у навчальному процесі. Впровадження мобільних технологій, платформ для навчання та інтерактивних засобів стає невід'ємною частиною навчального середовища, що забезпечує більш доступний та ефективний спосіб отримання знань.

Ще однією важливою тенденцією є персоналізоване навчання. Мобільні технології дозволяють адаптувати навчальний матеріал під конкретні потреби та рівень знань кожного студента, забезпечуючи більш індивідуалізований підхід до навчання. [27]

Сучасні педагогічні дослідження підкреслюють важливість інтерактивного навчання, де студенти активно взаємодіють з навчальним матеріалом. Використання інтерактивних завдань та технологій створює сприятливі умови для активного навчання та розвитку практичних навичок.

Мобільні технології роблять освіту більш доступною для різних категорій населення, включаючи тих, хто має обмежений доступ до традиційної освіти. Це підкреслює важливість розробки інноваційних інструментів для створення інтерактивних освітніх завдань на мобільних платформах, таких як Android. [28]

Швидкий розвиток знань та технологій вимагає постійного оновлення освітнього контенту. Мобільні технології надають можливість швидко та ефективно оновлювати освітні завдання та матеріали, щоб вони відповідали актуальним вимогам ринку праці та сучасним тенденціям у науці та технологіях.

Розробка інформаційної технології для створення освітніх програм та інтерактивних освітніх завдань на платформі Android відповідає цим тенденціям та відкриває можливості для поліпшення процесу навчання та забезпечення більш широкого доступу до якісної освіти.

2 ТЕОРЕТИЧНІ АСПЕКТИ ІНФОРМАЦІЙНИХ ОСВІТНІХ ТЕХНОЛОГІЙ ТА ІНТЕРАКТИВНИХ ОСВІТНІХ ЗАВДАНЬ. ПРОЕКТУВАННЯ ТЕХНОЛОГІЇ.

2.1 Освітні підходи та інтерактивне навчання

Освітні підходи та методи навчання мають велике значення для формування сучасної системи освіти.

Традиційні методи навчання, такі як лекції та читання підручників, довгий час були основними засобами передачі знань в освітній системі. Проте, з розвитком технологій і зміною потреб сучасного суспільства, виникає необхідність у більш інтерактивних підходах до навчання.

Інтерактивне навчання сприяє активній участі студентів у навчальному процесі. Воно націлене на спільну діяльність вчителя та учнів, створює можливість для обміну думками та дослідження навчального матеріалу через практичні завдання, дискусії та взаємодію. Інтерактивне навчання відкриває можливості для кращого засвоєння матеріалу та розвитку критичного мислення. [1]

В сучасному освітньому середовищі існують різні підходи до навчання, які враховують переваги інтерактивного навчання:

Конструктивізм: цей підхід передбачає, що учні будують свої знання на основі власного досвіду та взаємодії з навколишнім світом. Він підкреслює важливість активної ролі студентів у процесі навчання.

Соціокультурний підхід: цей підхід розглядає навчання як соціальний процес, у якому важливу роль відіграє взаємодія з оточуючими та іншими учнями. Він підкреслює значення комунікації та співпраці.

Застосування технологій в освіті: використання сучасних технологій, таких як мобільні додатки, планшети та віртуальна реальність, розширює

можливості для інтерактивного навчання та залучення студентів до процесу навчання.

Інтерактивне навчання може покращити якість освіти на багатьох рівнях. Воно стимулює активну участь студентів, сприяє засвоєнню матеріалу та розвитку критичного мислення. Крім того, інтерактивне навчання може підвищити мотивацію студентів та зробити навчання більш привабливим.

У цьому контексті, розробка інформаційної технології для створення освітніх матеріалів та інтерактивних освітніх завдань на платформі Android має великий потенціал.

2.2 Технології в освіті

Технологічний прогрес і впровадження сучасних технологій в освітній процес мають значний вплив на спосіб навчання та сприяють змінам у форматах та методах освіти.

Сучасні технології включають в себе різні засоби та програмні рішення, що можуть бути використані для поліпшення процесу навчання. До них входять:

- Мобільні додатки та планшети. Використання мобільних додатків і планшетів дозволяє студентам отримувати доступ до навчального контенту в будь-який час і в будь-якому місці.
- Віртуальна реальність (VR) та розширена реальність (AR). VR і AR можуть створювати іммерсивне навчальне середовище, де студенти можуть взаємодіяти з віртуальними об'єктами та ситуаціями.
- Електронні навчальні платформи. Онлайн-платформи для навчання дозволяють студентам вивчати новий матеріал, виконувати завдання та спілкуватися з викладачами та однокурсниками в онлайн-режимі.[26]

Можна виділити наступні переваги використання технологій в освітньому процесі

- Зручність та доступність. Технології роблять навчання більш доступним та зручним, дозволяючи студентам вчитися за власним графіком та знаходячись в будь-якому місці.
- Індивідуалізація навчання. Технології дозволяють вчителям налаштовувати навчальний матеріал на індивідуальні потреби студентів та враховувати їхні особливості.
- Залучення студентів. Інтерактивні технології сприяють більшій активності та мотивації студентів, а також сприяють розвитку критичного мислення та практичних навичок. [9]

При використанні технологій в освіті також існують свої виклики та недоліки. Розглянемо їх:

- Доступ до технологій. Не всі студенти мають рівний доступ до сучасних технологій, що може створювати нерівності в освіті.
- Забезпечення якості контенту. Важливо враховувати якість та достовірність навчального контенту, доступний через технології.
- Необхідність підготовки вчителів. Вчителі повинні мати відповідну підготовку та підтримку для ефективного використання технологій у навчанні.

У контексті розробки інформаційної технології для створення освітніх програм та інтерактивних освітніх завдань на платформі Android, важливо враховувати як переваги так і недоліки технологічного підходу до освітнього процесу щоб створити ефективний та доступний інструмент, який зможе покращити процес навчання.

2.3 Бажані характеристики навчальної інформаційної технології

Аналізуючи існуючі сучасні технології інтерактивного навчання можна визначити наступні критерії для якісної програмної розробки в цій галузі

Інтуїтивний інтерфейс: технологія повинна мати легкий у використанні та інтуїтивний інтерфейс, що дозволяє вчителям та авторам створювати інтерактивні завдання без особливого технічного досвіду.

Підтримка тачскріну: Додаток повинен бути адаптованим до використання на сенсорних пристроях.

Можливість імпорту та експорту матеріалів: Користувачам повинна бути надана можливість імпорту та експорту інтерактивних завдань для забезпечення легкої обміну матеріалами.

Створення різних типів завдань: Технологія повинна підтримувати створення різних типів інтерактивних завдань, таких як тести, кросворди, відкриті завдання та інші.

Можливість додавання мультимедійного контенту: Користувачам повинна бути надана можливість додавати зображення, відео, аудіо та інший мультимедійний контент до завдань.

Персоналізація завдань: Технологія повинна дозволяти вчителям налаштовувати параметри завдань, такі як складність, час виконання та доступність підказок.

Технологія відстеження прогресу студентів: Має бути можливість відстежувати та аналізувати прогрес студентів. [10]

Звіти та аналітика: Технологія повинна надавати можливість генерувати звіти та аналізувати результати для оцінки ефективності навчання.

Сумісність з різними пристроями: Технологія повинна бути оптимізованою для різних моделей та розмірів мобільних пристроїв з операційною системою Android.

Офлайн доступ: Має бути можливість використання технології в режимі офлайн для забезпечення доступу до матеріалів без Інтернет-з'єднання.

Захист даних: Технологія повинна гарантувати захист конфіденційності та безпеку даних користувачів.

Аутентифікація та авторизація: Забезпечення можливості автентифікації користувачів та управління їхніми правами доступу.

Захист від несанкціонованого доступу: У технології мають бути наявні заходи безпеки для запобігання несанкціонованому доступу до системи та даних.

2.4 Вимоги до системи

Функціональні вимоги до створюваної системи будуть наступні.

Система повинна надавати можливість користувачам (вчителям та авторам) створювати різні типи інтерактивних завдань, такі як тести, відкриті завдання, кросворди тощо. Наразі на початковій стадії системи ми будемо працювати лише з одним типом завдань тестами. В подальшому гнучкість системи дозволить додати різні типи завдань.

Система повинна дозволяти додавати зображення, відео, аудіо та інший мультимедійний контент до завдань для підвищення їхньої ефективності. В початковому варіанті система буде мати можливість додати зображення а також посилання. При подальшому розвитку системи в неї можна буде додати роботу с аудіо файлами.

Система повинна бути оптимізованою для використання на пристроях з операційною системою Android та має підтримувати різні версії цієї операційної системи. Система повинна однаково якісно працювати на пристроях з різним розміром екрану та на пристроях різних вендорів.

Система повинна надавати можливість імпорту та експорту інтерактивних завдань для обміну матеріалами між користувачами.

Серед загальних не функціональних вимог можна виділити наступні критерії системи:

Система повинна забезпечувати високий рівень безпеки даних користувачів, включаючи захист від несанкціонованого доступу та втрати даних.

Система повинна працювати ефективно та швидко навіть на пристроях з обмеженими ресурсами.

Інтерфейс користувача повинен бути інтуїтивно зрозумілим та легким у використанні, особливо для вчителів та авторів, які можуть не мати спеціального технічного досвіду. Наразі в початковій версії системи дизайн знаходиться на мінімально приємному рівні адже для повноцінного

створення інтерфейсу користувача подрібна додаткова робота дизайнера. Що виходить за рамки цієї роботи

Система повинна бути надійною та стійкою до відмов, а також повинна мати можливість резервного копіювання даних.

Система повинна бути масштабованою та готовою до росту кількості користувачів та обсягу навчального контенту. В цілому система має бути готова до розширення і мати на увазі те що ми маємо справу лише з початковою її версією.

2.5 Вибір архітектури системи

Система має складатися з клієнтської частини, що буде представляти собою мобільний додаток на базі операційної системи Android та серверної частини яка буде відповідати за збереження даних користувачів, освітніх матеріалів та аутентифікацію користувачів.

Клієнтський додаток буде призначений для створення, редагування та виконання інтерактивних освітніх завдань. Він також має відповідати за відображення навчального контенту для користувачів.

Також клієнт має складатися з екранів для реєстрації та авторизації користувачів.

В цілому архітектура клієнту була вибрана досить стандартна. Була вибрана архітектура яка не потребує багато boilerplate коду та має гарну реалізацію з коробки а саме MVVM (Model – View – ViewModel). Подібна архітектура вже довгий час є актуальною у світі розробки Android додатків. Вона легко налаштовується та маштабується в подальшому при потребі. [13]

Порівняємо використаний архітектурний підхід з аналогами

В порівнянні з MVI(Model – View - Intent), який є більш сучасним архітектурним підходом, MVVM не потребує багато коду для налаштування. А також дозволяє більш чіткіше розділяти слої даних. Та дозволяє більш зручніше керувати даними.

В порівнянні з MVP(Model – View - Presenter) MVVM не потребує прямих посилань між View і ViewModel, що дозволяє розділити їх більш чітко та зробити слої системи менш залежними один від одного. Також в цілому обробка даних та реакція на їх зміну простіше реалізовується за допомогою MVVM. MVC(Model – View – Controller) архітектура за своєю суттю схожа з MVP тому і переваги над нею у MVVM будуть схожі. [4]

Також в системі має бути серверна сторона, яка в нашому випадку, буде представлена платформою Firebase, сервіси якої повністю покриють необхідні дії з серверної сторони.

Проаналізуємо альтернативи використанню платформи Firebase. Для реалізації системи аутентифікації можна було би використати різні сервіси, як наприклад Auth0, але в порівнянні з системою аутентифікації, Firebase Auth0 потребує додатковиї налаштувань. Для досить простої аутентифікації Firebase підходить набагато більший. Також його використання є зручнішим завдяки інтеграції з середовищем розробки Android Studio. [16]

Для зберігання ж медіаконтенту в хмарі можна було вибрати аналогічні сервіси від Amazon або Microsoft всі вони є сучасними та справляться з задачею збереження файлів.

Як альтернативу серверної бази даних можливо використати Realm. Переваги ж Firebase Realtime Database над Realm заключаютсья в простоті доступа до даних та швидкості їх оновлення, яке у випадку Firebase Realtime Database відбувається негайно. Realm же являється більш швидким аналогом та підтримує краще шифрування. [21]

В цілому серверна сторона представлена Firebase була вибрана для проекту саме через комфорт роботи з нею а також через те, що всі сервіси об`єднані в одну консоль за допомогою якої керування даними є легким та комфортним.

Firebase в рамках консолі також дозволяє в подальшому в рамках розвитку проекту підключити інші сервіси: як наприклад аналітику або Push повідомлення.

Також в плані архітектури можна відмітити те що клієнтський додаток має кешувати дані для аутентифікації локально на пристрої користувача для забезпечення доступу до додатку без необхідності авторизації при кожному його запуску.

Характеризуючи архітектуру системи можна відзначити, що створена архітектура системи дозволяє масштабувати ресурси на сервері для обробки більшої кількості даних, а також легко в подальшому масштабувати клієнтську частину системи. Firebase дозволяє швидко та легко працювати з даними в системі не здійснюючи лишню нагрузку на пристрій клієнта. Розділення системи на клієнтську та серверну частини робить розробку та оновлення більш зручними та швидкими. Архітектура системи спроектована з метою забезпечення ефективного та надійного функціонування системи для створення інтерактивних освітніх завдань на платформі Android, з урахуванням потреб користувачів та вимог до продуктивності та безпеки.

2.6 Вибір технологій

Серверну частину Firebase було досить детально розглянуто в минулому розділі тож цей розділ буде присвячений технологіям клієнту та їх аналізу.

Проект базується на мові програмування Kotlin, яка є офіційною мовою для розробки Android-додатків. Використання Kotlin дозволяє зменшити кількість помилок, підвищити продуктивність та полегшити читання та розуміння коду. Також в мові Kotlin є доступ до Coroutines - інструменту асинхронного програмування, який здатен створювати потоки, що займають мало пам’яті, яких немає в Java, що дозволяє оптимізувати нашу роботи з background thread - потоком в якому відбувається вся робота з даними в рамках Android додатків. [23]

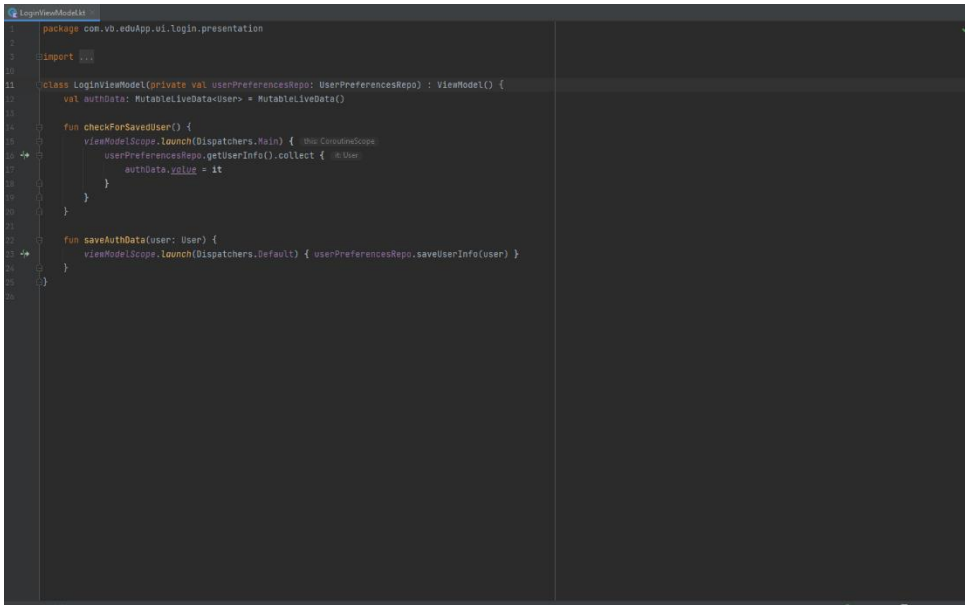


Рисунок 2.1 - Приклад роботи з Kotlin Coroutines

Для кешування даних авторизації було вирішено використати Preferences DataStore. Аналогічно можна було б використати Room, SQLite бази даних, збереження даних в файлах та SharedPreferences.

Локальні бази даних як Room та SQLite потребують досить довгого налаштування а також займають більше місця ніж аналоги. Вони створені

для збереження комплексних об`ємів даних тож вони просто не потрібні нам для кешування незначного об’єму даних. Також існує Proto база даних під назвою Proto DataStore, яка також потребує широкого налаштування та дозволяє зберігати досить екзотичні типи даних. Стосовно можливості зберігання примітивних даних в файлах це також є перебільшенням для роботи з таким об’ємом даних також робота з файловою системою Android потребує об’явлення додаткових дозволів та їх запит у користувача. Що в нашому разу просто не потрібно.

Отже, основним аналогом для Preferences DataStore є SharedPreferences досить проста бібліотека, яка дозволяє зберігати значення в виді ключ - значення. Основною причиною через яку був зроблений вибір саме на користь Preferences DataStore являється консистентність доступу до даних в background thread а також типобезпека та інтеграція Preferences DataStore з Kotlin Coroutines, які були обрані основним інструментов асинхронної роботи з даними. [8]

Також було використано технологію для легкого завантаження зображень з інтернету Glide. В нашому випадку вона використовувалась для завантаження зображень з Firebase Storage на сторінку матеріалу. [17]

Існує декілька відомих альтернатив цього фреймворку. Як наприклад Retrofit та Coil.

Retrofit це в цілому web framework не орієнтований безпосередньо на Android тому він не має потрібного нам функціоналу для простого доступу до медіа файлів на телефоні користувача а також він має даяі проблеми при роботі з Android як наприклад проблема завантаження зображень в RecyclerView. Coil не має подібних проблем та в цілому він є більш сучасним фреймворком, але в нього також немає необхідного функціоналу. [20]

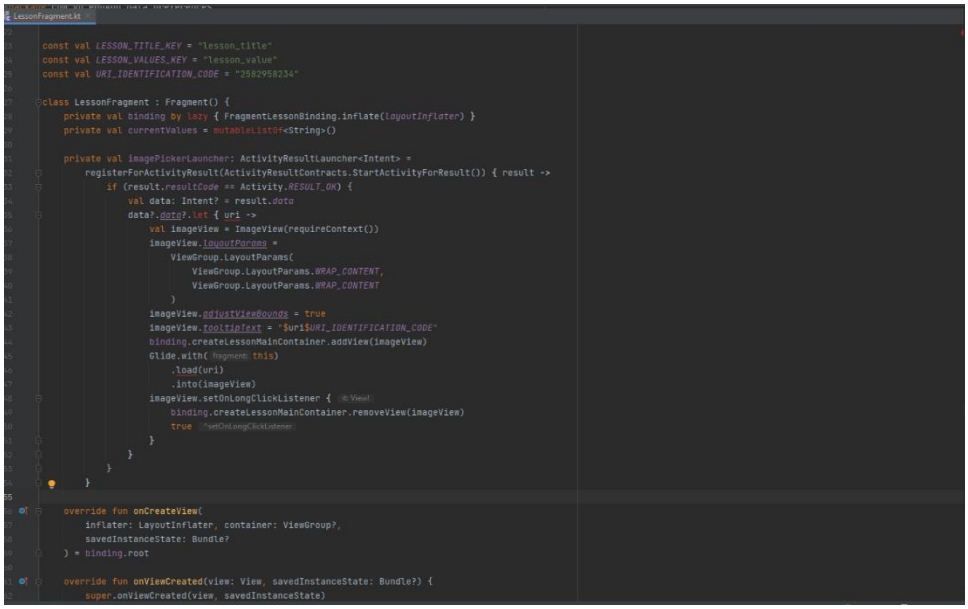


Рисунок 2.2 - Приклад роботи з Glide

Також ми використовуємо бібліотеку Dexter для зручного запросу дозволів у користувача а також Koin для впровадження залежностей.

Сучасні альтернативи Koin це Dagger 2 та Hilt на базі Dagger 2

Dagger 2 - це потужна бібліотека для впровадження залежностей, але вона вимагає набагато більше часу для налаштування порівняно з Koin. Вона використовує анотації та граф залежностей на етапі компіляції, що може підвищити продуктивність за рахунок більш легкого пошуку проблем.[6][19]

Hilt являє собою такий же Dagger 2, але заміняє частину коду для налуштавання анотаціями в цілому для нашого проекту. Це гарна альтернатива Koin, але Koin все ще легшу в плані налаштування та читабельності коду. [6][19]

Також серед технологій можна відмітити Navigation Component, яка являє собою бібліотеку для зручної та швидкої навігації по фрагментах додатку. Фрагмент це частина інтерфейсу користувача. Як альтернатива можна перемикати фрагменти без використання фреймворку. Але це потребує додаткового коду для створення графу навігацій а також подібний метод

веде до здійснення потенційних помилок. Також для навігації існує фреймворк Cicerone, який є більш гнучким у своєму налаштуванні ніж Navigation Component, але потребує набагато більше коду для свого налаштуваня.

2.7 Вибір технології для реалізації інтерфейсу користувача

В Android наразі існує два підходи до роботи з інтерфейсом користувача. Новий підхід Compose

- Декларативний підхід: Compose використовує декларативний підхід до створення інтерфейсу, де ви описуєте, як має виглядати інтерфейс на основі стану додатка.
- Код на Kotlin: Compose використовує Kotlin для опису інтерфейсу, що робить код читабельнішим та зменшує кількість зайвого XML-коду.
- Динамічні зміни: Легко внести динамічні зміни у вигляді інтерфейсу, наприклад, змінити текст кнопки чи зображення під час взаємодії з користувачем.
- Кастомізація: Compose надає більше можливостей для кастомізації компонентів і стилізації інтерфейсу.
- Анімації: Легко додавати анімації до компонентів інтерфейсу без необхідності власноручно програмувати анімації. [18]

Та конвенційний підхід зчитування розмітки з XML файлів

- Імперативний підхід: В XML ви описуєте інтерфейс інструкціями імперативного типу, вказуючи, як розмістити компоненти на екрані.
- XML-файли: Для кожного екрана або макету вам потрібен відповідний XML-файл.
- Кастомізація: XML дає менше можливостей для кастомізації, інколи потрібно використовувати стилі або ресурси для досягнення певного вигляду.
- Анімації: Для анімацій потрібно використовувати XML-ресурси або програмувати анімації в коді.

В цілому Compose може дати більше гнучкості при розробці UI але імперативний підхід має свої плюси в тому, що багато рішень вже створено у якості різних View тоді як в Compose пропагандується підхід при якому

розробник має сам створювати найменші елементи та налаштовувати їх. В цілому Compose потребує більше часу для написання інтерфейсу, а швидкість роботи додатку або його білду при цьому не змінюється. Також в імперативному підході є більш чіткий розділ між безспосередню розміткою яка знаходиться в XML та функціональним кодом який знаходиться в Java або Kotlin файлах. Що робить стиль архітектуру та код в цілому більш читабельним. Також середовище розробки більш зручно налаштоване на роботу з розміткою в XML файлах. Отже в нашому випадку був обраний імперативний підхід створення UI. [2]

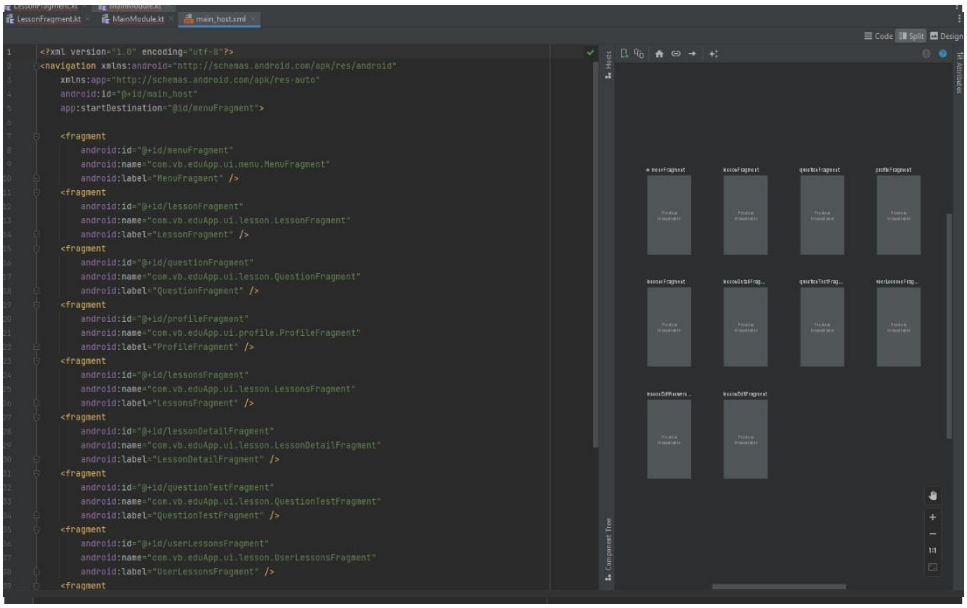


Рисунок 2.3 - Навігаційний граф Nabigation Components

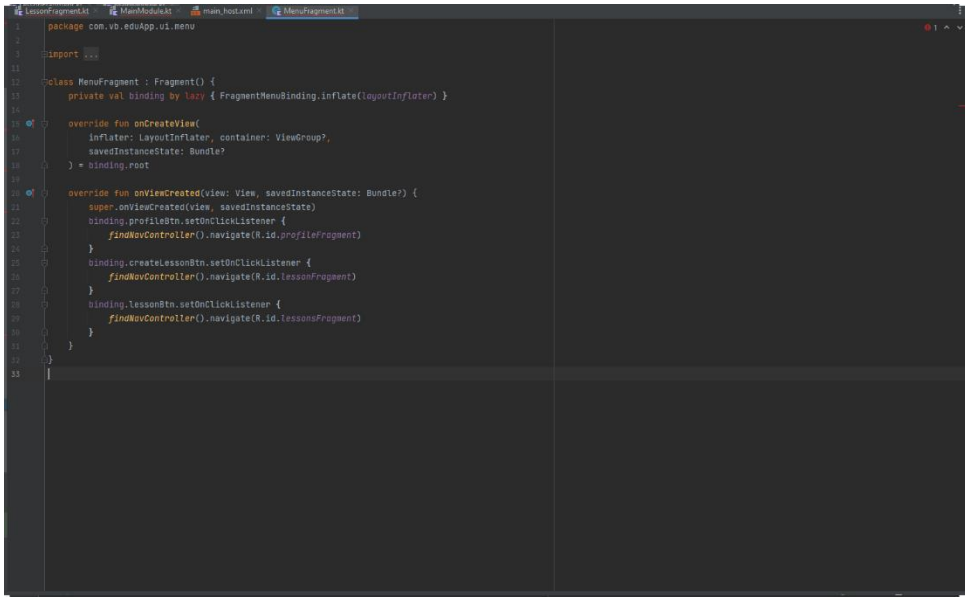


Рисунок 2.4 - Приклад навігації з допомогою Navigation Components

3 ІНФОРМАЦІЙНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ТЕХНОЛОГІЇ

3.1 Архітектура системи

Отже, згідно обраної архітектури клієнта його було створено наступні слої системи

- Data, який відповідає за моделі даних, їх обробку та збереження. В ньому ж знаходиться і клієнтська база даних яка відповідає за кешування даних користувача
- UI, що відповідає за обробку вхідних даних та їх відображення в інтерфейсі користувача. В середині слою кожен екран має свій власний пакет в середині якого зберігаються класи для роботи з UI та допоміжні класи які відповідають за створення кастомних UI елементів та адаптерів до них.
- DI. Цей пакет відподає за налаштування та роботу з dependency Injection та його графом
- App, Utils - Окремі сутності які відповідають у випадку App за підключення необхідних фреймворків при вході в програму та у разі Utils за методи які використовуються в різних місцях програми та мають бути спільними для всіх компонентів системи.

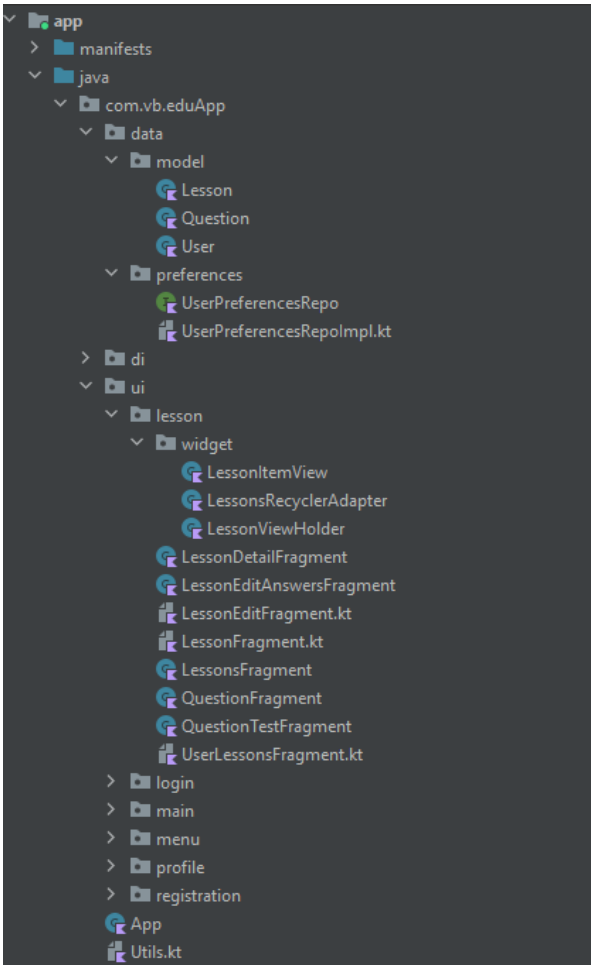


Рисунок 3.1 - Архітектура додатка

Також в системі є серверна сторона, яка в нашому випадку представлена платформою Firebase.

Серверна сторона, за яку в нашому випадку відповідає Firebase, містить функціонал для реєстрації та авторизації користувачів. В нашому випадку ми будемо підтримувати реєстрацію тільки за паролем та поштовою скринькою користувача. В подальшому сервіс дозволяє з легкістю додати авторизацію за всіма сучасними популярними методами авторизації.

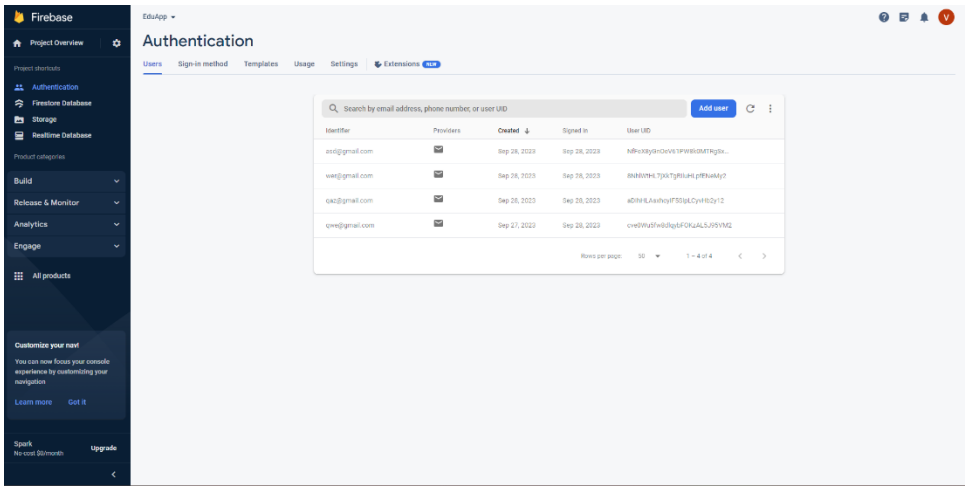


Рисунок 3.2 - Серверна консоль системи авторизації

Також серверна сторона складається зі сховища у хмарі за допомогою якого ми будемо зберігати картинки та в подальшому можливо інші медіа файли, які вчителя та автори навчального контенту вставляють в створені матеріали.

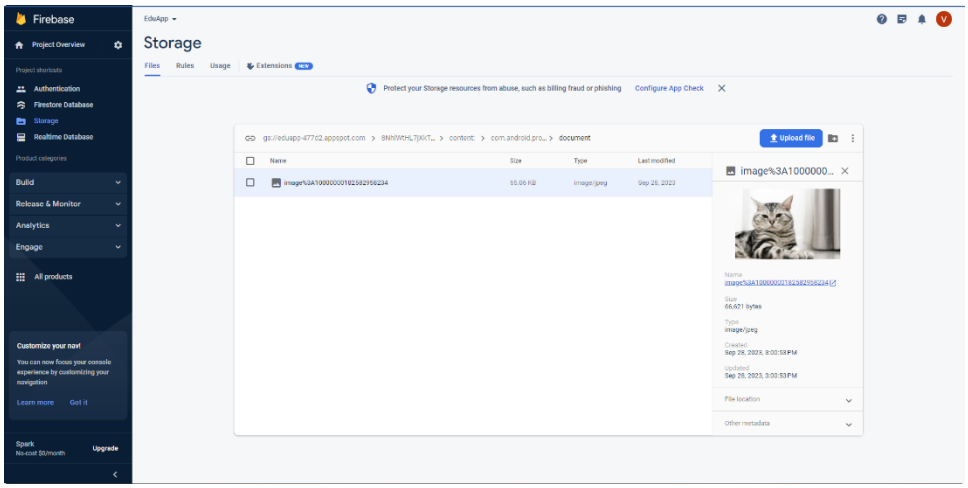


Рисунок 3.3 - Інтерейс серверного хранлища медіаконтенту

І основна частина бекенду - це база даних в якій зберігається інформація про створені матеріали. Вона грає ключову роль у забезпеченні доступу до даних та забезпеченні їхньої цілісності.

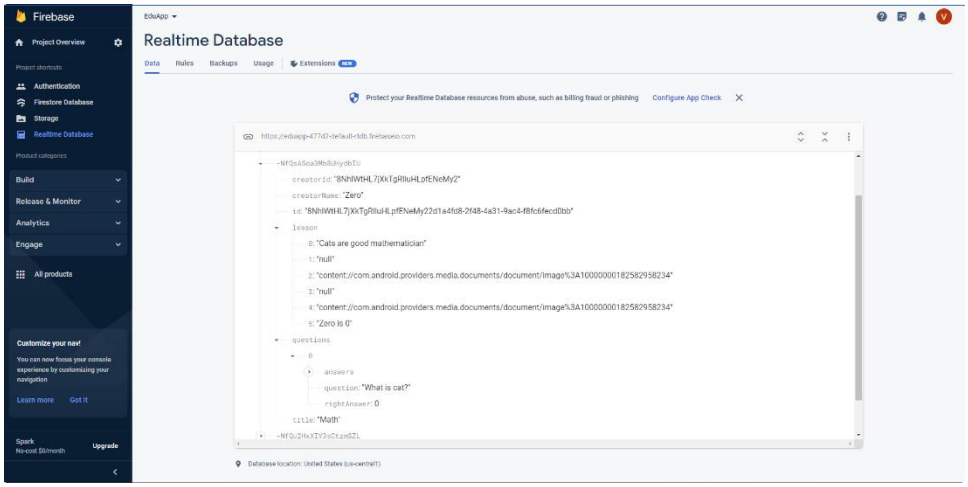


Рисунок 3.4 - Структура бази даних

3.3 Моделі даних. Кешування. DI

Отже, розглянемо слой даних нашого клієнтського додатку. В ньому знаходяться моделі даних та локальне сховище для збереження даних користувача для його авторизації тобто для кешування його даних.

Ми маємо 3 моделі з даними

- Lesson модель даних яка представляє освітній матеріал. Та має поля id(String) - унікальний ідентифікатор матеріалу, title(String) - назва матеріалу, creatorId(String) - ідентифікатор користувача який додав матеріал, creatorName(String) - ім’я користувача який додав матеріал, lesson(List<String>) - матеріал, questions(List<Question>) - питання до матеріалу
- Question представляє собою одне окреме питання до матеріалу та складається з полів question(String) - питання, answers(List<String>) - відповіді на питання та rightAnswer(Int) - індекс правльної відповіді в списку питань
- User - модель даних користувача яка містить поля email(String) та password(String) для адреси почтової скриньки та паролю відповідно

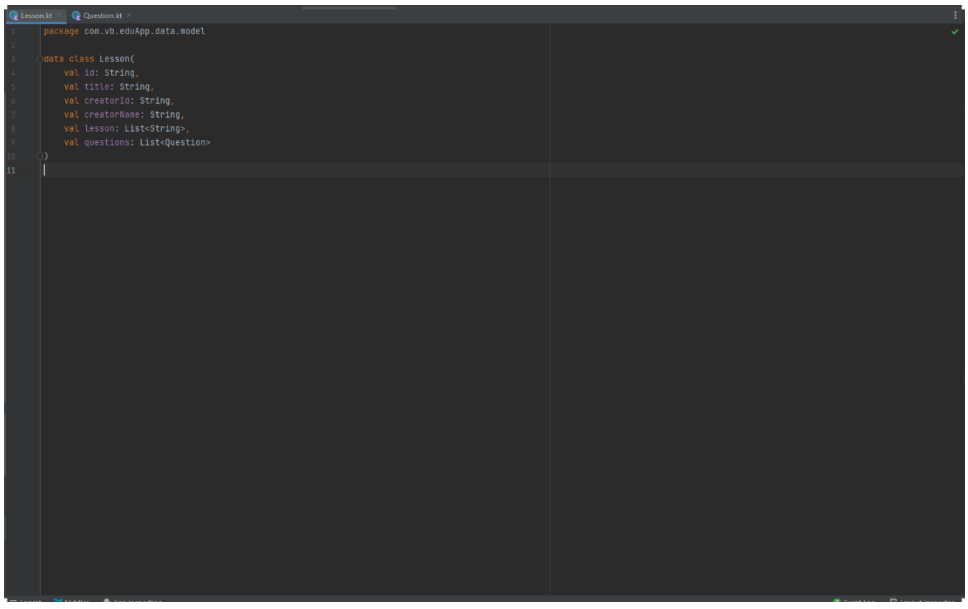


Рисунок 3.5 - Модель даних навчального матеріалу

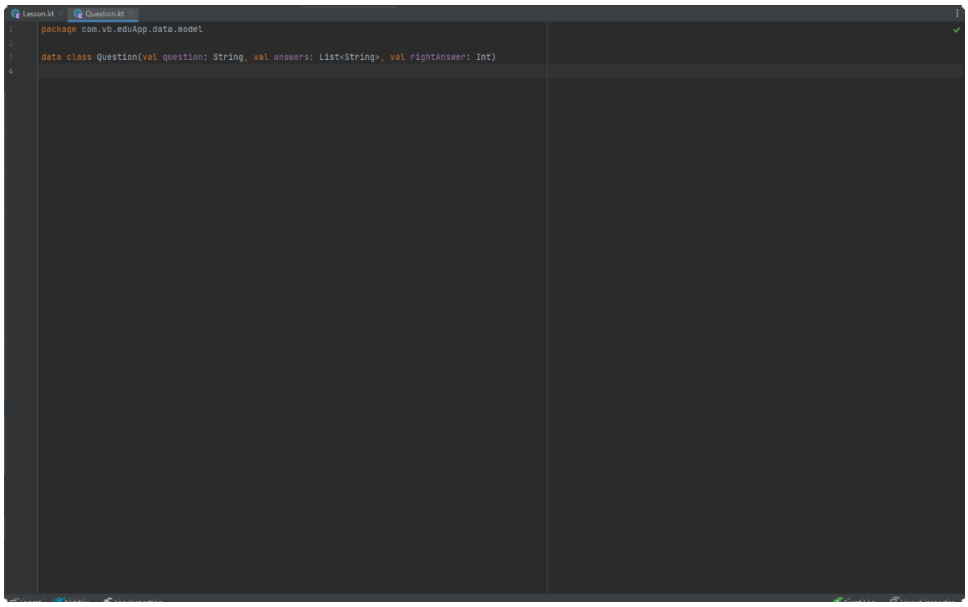


Рисунок 3.6 - Модель даних завдань до матеріалу

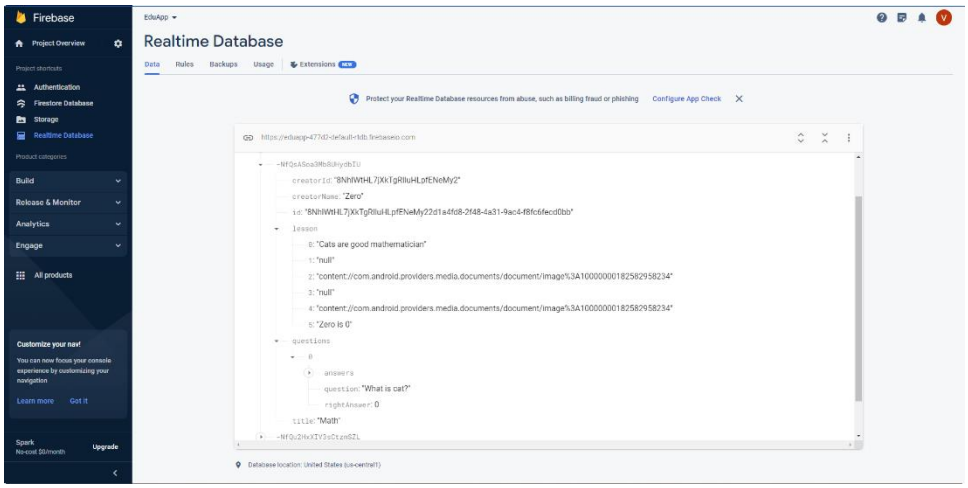
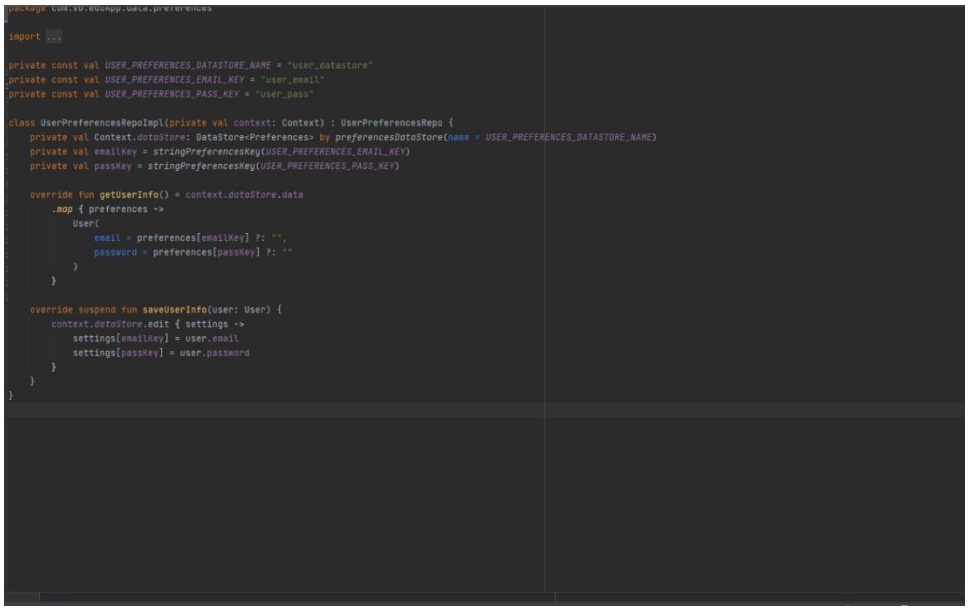


Рисунок 3.7 - Вигляд моделі в інтерфейсі серверної бази даних

Тепер розглянемо реалізацію кешування даних авторизації користувачів. Реалізація використовує технологію Preferences Data Store її основна реалізація знаходиться в класі UserPreferencesRepoImpl який реалізує інтерфейс UserPreferencesRepo що зроблено для відокремлення реалізації.

Клас реалізує 2 методи інтерфейсу getUserInfo() та saveUserInfo(user: User) які відповідають за збереження та отримання даних користувача

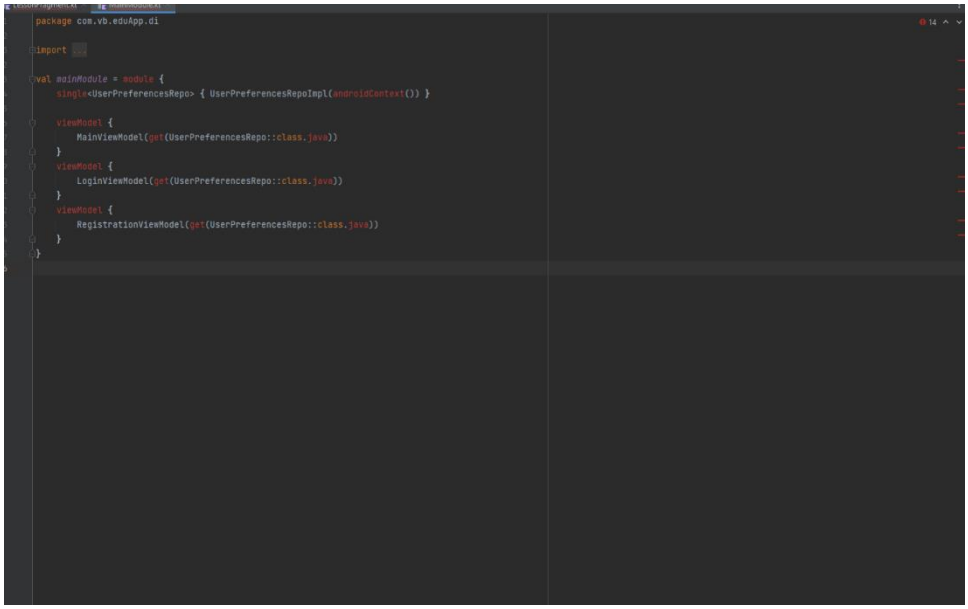


```
package com.vb.todoapp.data.preferences

import ...

private const val USER_PREFERENCES_DATASTORE_NAME = "user_datastore"
private const val USER_PREFERENCES_EMAIL_KEY = "user_email"
private const val USER_PREFERENCES_PASS_KEY = "user_pass"

class UserPreferencesRepoImpl(private val context: Context) : UserPreferencesRepo {
    private val Context.dataStore: DataStore<Preferences> by preferencesDataStore(name = USER_PREFERENCES_DATASTORE_NAME)
    private val emailKey = stringPreferencesKey(USER_PREFERENCES_EMAIL_KEY)
    private val passKey = stringPreferencesKey(USER_PREFERENCES_PASS_KEY)

    override fun getUserInfo() = context.dataStore.data
        .map { preferences ->
            User(
                email = preferences[emailKey] ?: "",
                password = preferences[passKey] ?: ""
            )
        }

    override suspend fun saveUserInfo(user: User) {
        context.dataStore.edit { settings ->
            settings[emailKey] = user.email
            settings[passKey] = user.password
        }
    }
}
```

Рисунок 3.8 - Реалізація Preferences DataStore

Також розглянемо граф залежностей для dependencyInjection а також його підключення у точці входу в програму та клас Utils з універсальними методами.

Граф залежностей складається з одного модуля сутності MainModule що забезпечує ін’єкцію сінглтон компонента локальної бази даних, а також viewModels - класи для обробки даних для view MainViewModel, LoginViewModel, RegistrationViewModel.

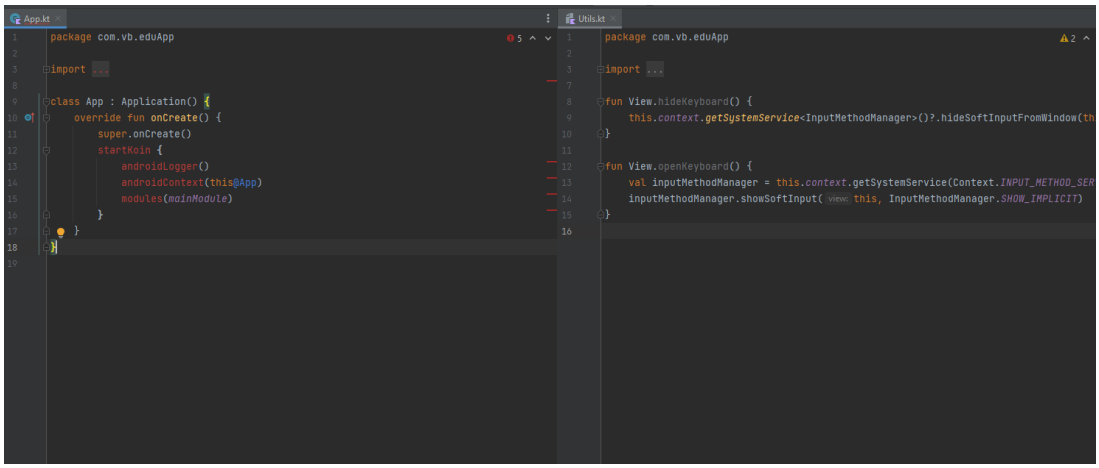


```
package com.vb.eduApp.di

import ...

val mainModule = module {
    single<UserPreferencesRepo> { UserPreferencesRepoImpl(androidContext()) }

    viewModel {
        MainViewModel(get(UserPreferencesRepo::class.java))
    }
    viewModel {
        LoginViewModel(get(UserPreferencesRepo::class.java))
    }
    viewModel {
        RegistrationViewModel(get(UserPreferencesRepo::class.java))
    }
}
```

Рисунок 3.9 - Головний модуль графа залежностей Koin у додатку

Клас App відповідає за підключення графу залежностей koin та лише викликая методи startKoin та передає йому основний контекст додатку та модулі які мають бути підключені до графу в методі onCreate().

Клас Utils містить в собі 2 методи hideKeyboard() та openKeyboard() що відповідають за відкриття та закриття системної клавіатури для вводу даних користувачем через поля в інтерфейсі



App.kt

```
package com.vb.eduApp

import ...

class App : Application() {
    override fun onCreate() {
        super.onCreate()
        startKoin {
            androidLogger()
            androidContext(this@App)
            modules(mainModule)
        }
    }
}
```

Utils.kt

```
package com.vb.eduApp

import ...

fun View.hideKeyboard() {
    this.context.getSystemService<InputMethodManager>()?.hideSoftInputFromWindow(th
}

fun View.openKeyboard() {
    val inputMethodManager = this.context.getSystemService(Context.INPUT_METHOD_SER
    inputMethodManager.showSoftInput( view: this, InputMethodManager.SHOW_IMPLICIT)
}
```

Рисунок 3.10 - Класи App та Utils

3.3 Інтерфейс користувача

Структура екранів клієнтського додатку складається с наступних частин

- Екрани для аутентифікації користувача а саме екран логіну та реєстрації
- Основне меню додатку
- Екрани для створення матеріалів та завдань до матеріалів
- Екран зі списком всіх створених матеріалів з можливістю переглянути матеріали та пройти відповідні завдання
- Екран профілю користувача так екран створених користувачем матеріалів з можливістю їх редагування та редагування завдань до них

Класи які відповідають за роботу з інтерфейсом можна поділити на Activity та Fragments. Activity це класи які представляють інтерфейс користувача та охоплюють визначену зону екранів користувача схожими за своєю темою. Fragments існують в середині Activity та представляють собою конкретний екран користувача. [13]. Всього в додатку є 3 activity та 10 fragments. Розглянемо їх

Отже, перша activity яка запускається при запуску додатку це LoginActivity. Вона забезпечує login користувача та надає можливість перейти до екрану реєстрації нового користувача та зареєструватись. Вона працює з xml файлом activity_login а також має LoginViewModel, яка відповідає за кешування даних користувача при логіні. Аналогічна по функціоналу viewModel є і в RegistrationActivity а саме RegistrationViewModel.

Розглянемо метод onCreate в LoginAcitivity. В ньому ми спочатку ініціалізуємо обьект який відповідає за зв’язок з сервером та аутентифікацію через сервер. Далі підключаємо xml файл та перевіряємо чи в локальному сховищу є закешовані дані користувача за допомогою метода checkForSavedUser(). Також налаштовуємо зчитання та обробку некоректно

введених даних користувача для полей авторизації. Налаштовуємо переходи до інших екранів додатку.

Якщо ж користувач перейшов до екрану реєстрації то він перейшов до сутності RegistrationActivity. Її функціонал дуже схожий на LoginActivity але окрім логіну вже існуючого юзера викликаються методи для створення нового при коректно введених даних.

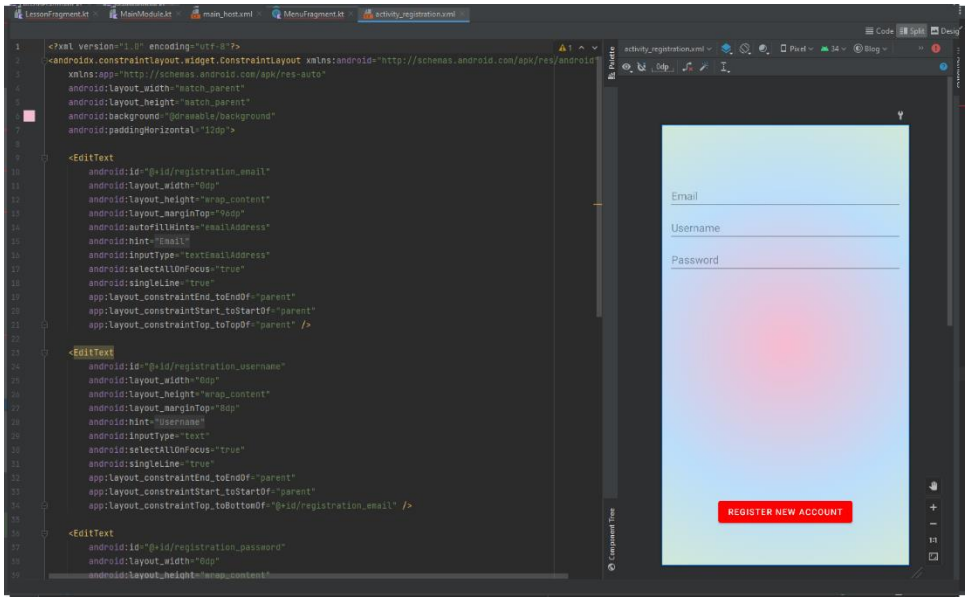


Рисунок 3.11 - XML розмітка екрану реєстрації

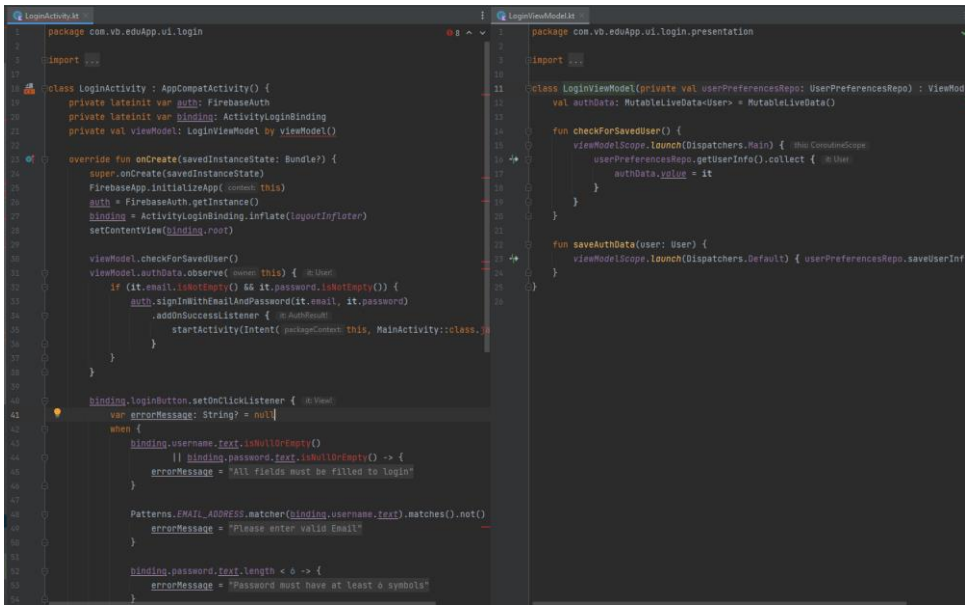


Рисунок 3.12 - LoginActivity та LoginViewModel

Після аутентифікації користувач попадає в основне меню додатку. Екрани з цього моменту базуються на одній activity MainActivity та представляють собою фрагменти. Екран основного меню MenuFragment має простий функціонал та всього лиш пропонує навігацію до подальших екранів додатку. А саме екрану профілю користувача ProfileFragment. Екрану зі списком створених матеріалів LessonsFragment та екрану створення нового матеріалу LessonFragment.

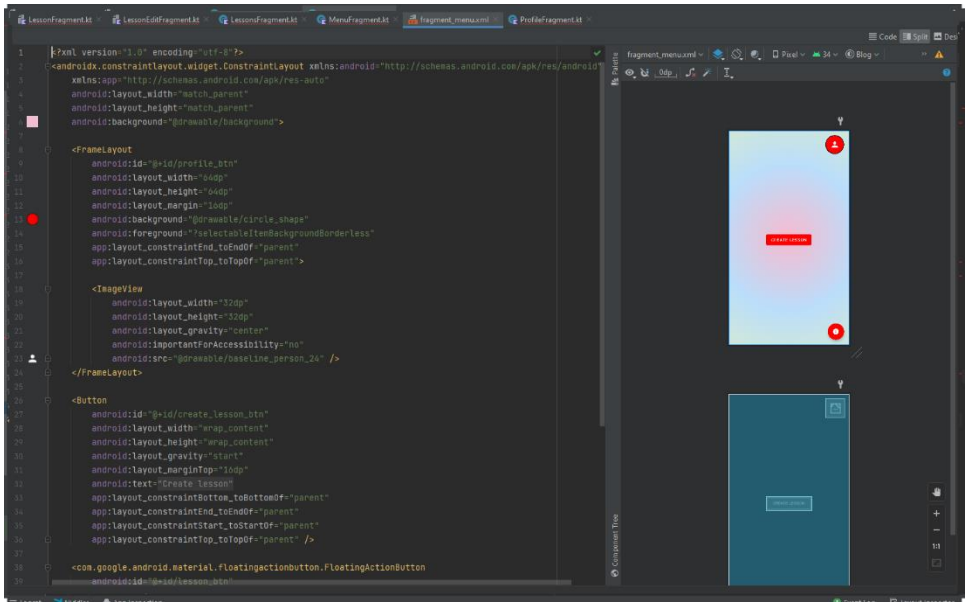
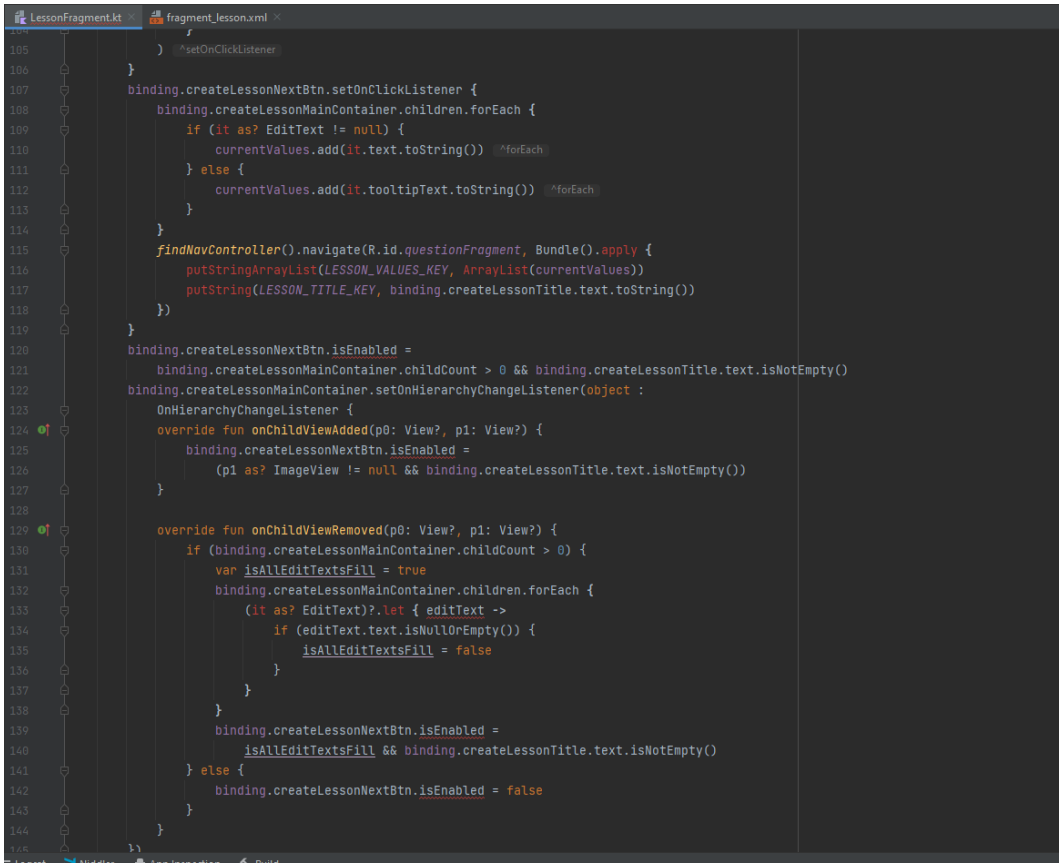


Рисунок 3.13 - XML розмітка головного екрану меню

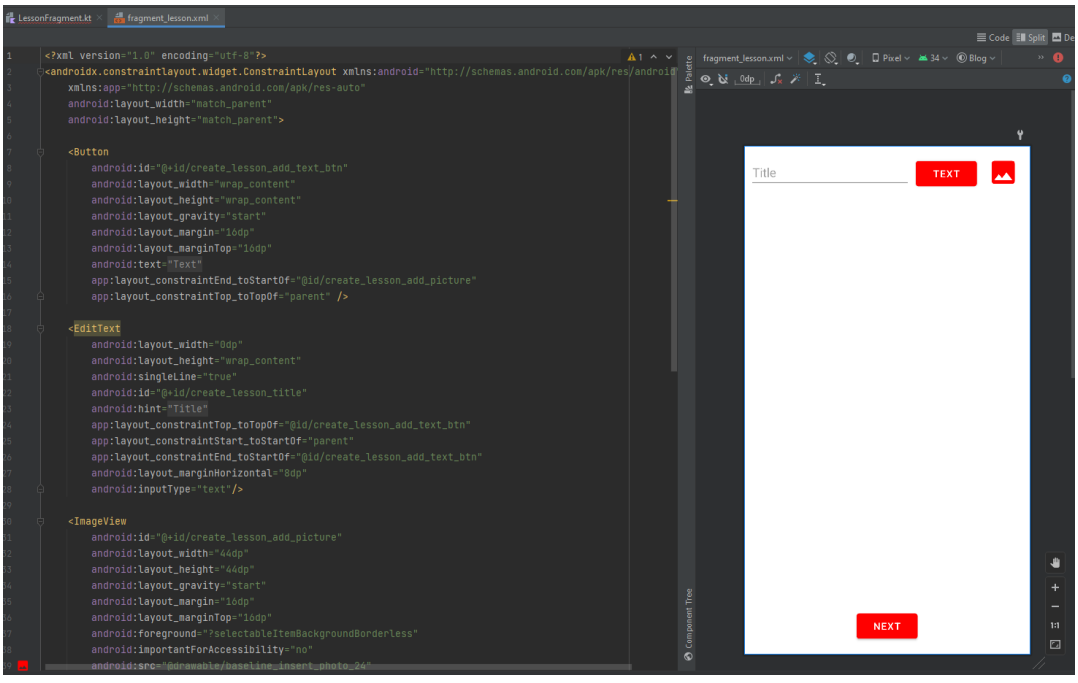
Розглянемо екран LessonFragment більш детально. Він дозволяє ввести назву матеріалу та вставляти або видаляти текстові елементи в матеріалі та картинки.

В методі onViewCreated який викликається при створенні фрагменту ми налаштовуємо генерацію нових елементів в матеріалі а також обробку некоректно заповнених полів. Якщо ж користувач додав матеріал то він може перейти до екрану QuestionFragment який відповідає за питання до створеного матеріалу.



```
LessonFragment.kt | fragment_lesson.xml

        ) ^setOnClickListener
    }
    binding.createLessonNextBtn.setOnClickListener {
        binding.createLessonMainContainer.children.forEach {
            if (it as? EditText != null) {
                currentValues.add(it.text.toString()) ^forEach
            } else {
                currentValues.add(it.tooltipText.toString()) ^forEach
            }
        }
        findNavController().navigate(R.id.questionFragment, Bundle().apply {
            putStringArrayList(LESSON_VALUES_KEY, ArrayList(currentValues))
            putString(LESSON_TITLE_KEY, binding.createLessonTitle.text.toString())
        })
    }
    binding.createLessonNextBtn.isEnabled =
        binding.createLessonMainContainer.childCount > 0 && binding.createLessonTitle.text.isNotEmpty()
    binding.createLessonMainContainer.setOnHierarchyChangeListener(object :
        OnHierarchyChangeListener {
        override fun onChildViewAdded(p0: View?, p1: View?) {
            binding.createLessonNextBtn.isEnabled =
                (p1 as? ImageView != null && binding.createLessonTitle.text.isNotEmpty())
        }

        override fun onChildViewRemoved(p0: View?, p1: View?) {
            if (binding.createLessonMainContainer.childCount > 0) {
                var isAllEditTextsFill = true
                binding.createLessonMainContainer.children.forEach {
                    (it as? EditText)?.let { editText ->
                        if (editText.text.isNullOrEmpty()) {
                            isAllEditTextsFill = false
                        }
                    }
                }
                binding.createLessonNextBtn.isEnabled =
                    isAllEditTextsFill && binding.createLessonTitle.text.isNotEmpty()
            } else {
                binding.createLessonNextBtn.isEnabled = false
            }
        }
    })
```

Рисунок 3.14 - LessonFragment



```
LessonFragment.kt | fragment_lesson.xml

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/create_lesson_add_text_btn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="start"
        android:layout_margin="16dp"
        android:layout_marginTop="16dp"
        android:text="Text"
        app:layout_constraintEnd_toStartOf="@id/create_lesson_add_picture"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:singleLine="true"
        android:id="@+id/create_lesson_title"
        android:hint="Title"
        app:layout_constraintTop_toTopOf="@id/create_lesson_add_text_btn"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toStartOf="@id/create_lesson_add_text_btn"
        android:layout_marginHorizontal="8dp"
        android:inputType="text"/>

    <ImageView
        android:id="@+id/create_lesson_add_picture"
        android:layout_width="44dp"
        android:layout_height="44dp"
        android:layout_gravity="start"
        android:layout_margin="16dp"
        android:layout_marginTop="16dp"
        android:foreground="?selectableItemBackgroundBorderless"
        android:importantForAccessibility="no"
        android:src="@drawable/baseline_insert_photo_24"

Title
TEXT
NEXT
```

Рисунок 3.15 - xml файл екрану створення матеріалу

В QuestionFragment користувач може додати запитання до матеріалу та зберегти матеріал. В ньому ми спочатку зв’язуємось з серверною базою даних та описуємо збереження даних в ній. Також організовуємо збереження медіафайлів в окреме хмарне сховище. Налаштовуємо генерацію необхідних елементів для вводу нових питань.

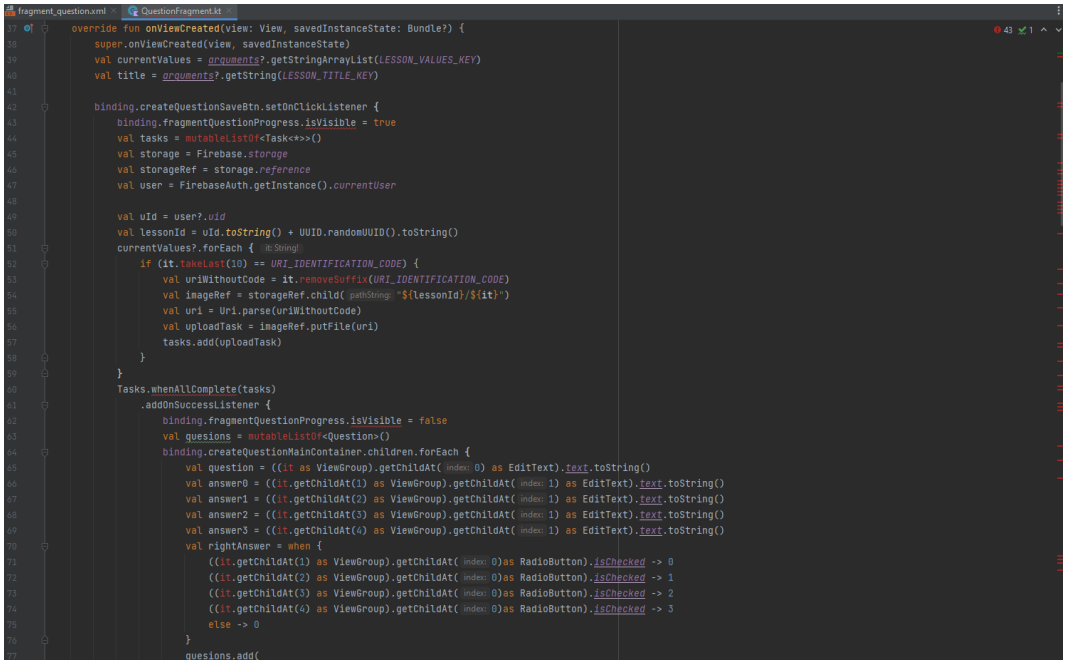


Рисунок 3.16 - фрагмент коду збереження матеріалу на сервері Firebase

Тепер розглянемо екран профілю користувача ProfileFragment. Це досить простий екран який відображає базову інформацію авторизованого користувача та дозволяє перейти до фрагменту зі списком створених користувачем матеріалів з можливістю їх редагування UserLessonsFragment.

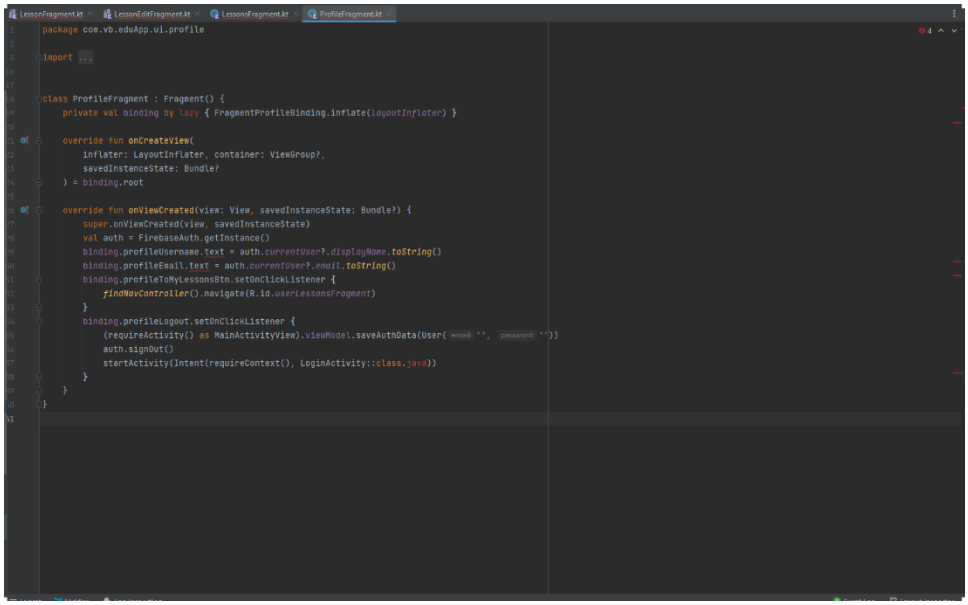
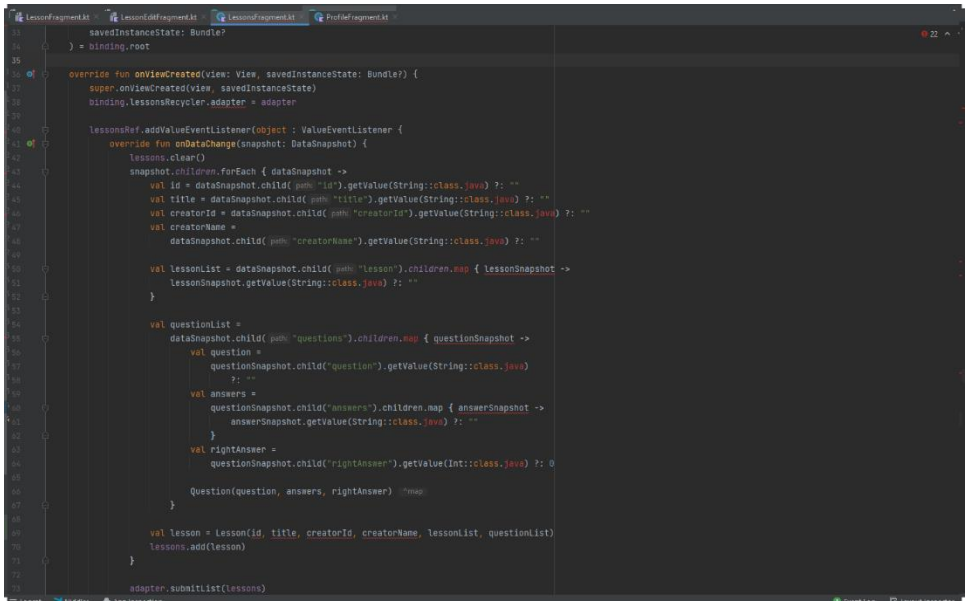


Рисунок 3.17 - Реализація екрану профіля користувача

По суті своєї роботи екрани загального списку матеріалів та списку матеріалів окремого користувача дуже схожі. Відрізняються вони лише окремим фільтром за яким ми фільтруємо список матеріалів а саме ми маємо дивитися на те щоб відображались тільки пости creatorId яких співпадає з userId авторизованого користувача. Сам принцип роботи екранів досить простий і полягає в тому щоб отримати з серверу дані с матеріалами та відобразити їх. Також екрани для редагування матеріалів LessonEditrFragment та LessonEditAnswersFragment по своєй суті ідентичні з екранами для створення матеріалу. Відрізняються вони лише додатковою логікою для зчитування уже збереженого на сервері матеріалу до логікою оновлення матеріалу в базі а не створення нового обьекту.

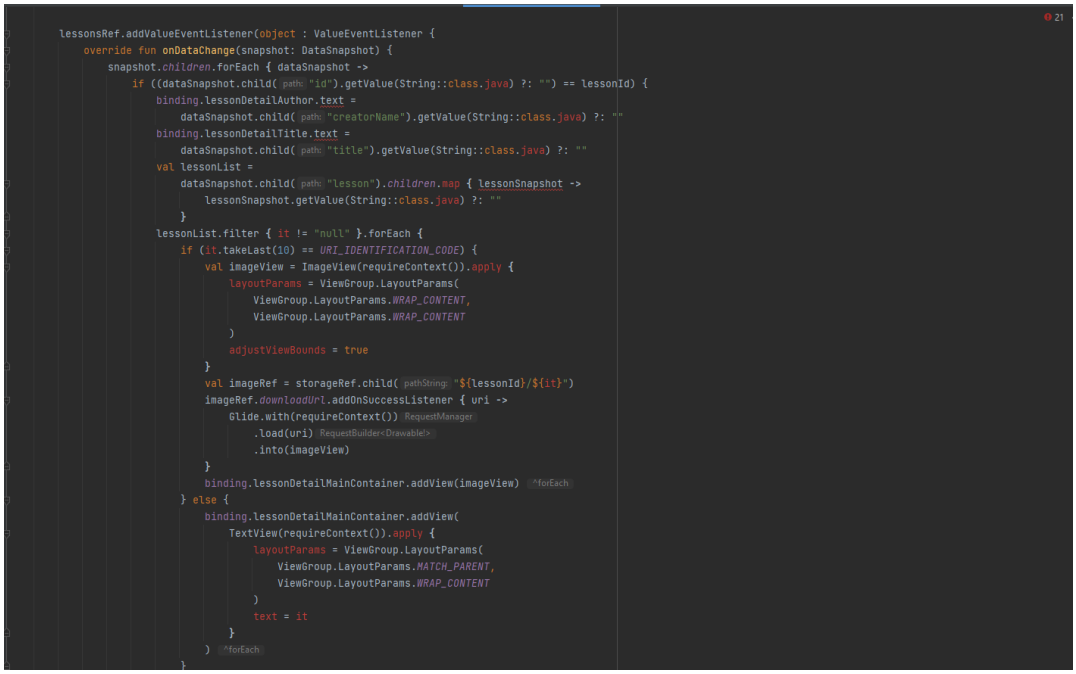


```
        savedInstanceState: Bundle?
    ) = binding.root

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        binding.lessonsRecycler.adapter = adapter

        lessonsRef.addValueEventListener(object : ValueEventListener {
            override fun onDataChange(snapshot: DataSnapshot) {
                lessons.clear()
                snapshot.children.forEach { dataSnapshot ->
                    val id = dataSnapshot.child( path: "id").getValue(String::class.java) ?: ""
                    val title = dataSnapshot.child( path: "title").getValue(String::class.java) ?: ""
                    val creatorId = dataSnapshot.child( path: "creatorId").getValue(String::class.java) ?: ""
                    val creatorName =
                        dataSnapshot.child( path: "creatorName").getValue(String::class.java) ?: ""

                    val lessonList = dataSnapshot.child( path: "lesson").children.map { lessonSnapshot ->
                        lessonSnapshot.getValue(String::class.java) ?: ""
                    }

                    val questionList =
                        dataSnapshot.child( path: "questions").children.map { questionSnapshot ->
                            val question =
                                questionSnapshot.child("question").getValue(String::class.java)
                                    ?: ""
                            val answers =
                                questionSnapshot.child("answers").children.map { answerSnapshot ->
                                    answerSnapshot.getValue(String::class.java) ?: ""
                                }
                            val rightAnswer =
                                questionSnapshot.child("rightAnswer").getValue(Int::class.java) ?: 0

                            Question(question, answers, rightAnswer)  *map
                        }

                    val lesson = Lesson(id, title, creatorId, creatorName, lessonList, questionList)
                    lessons.add(lesson)
                }

                adapter.submitList(lessons)
```

Рисунок 3.18 - Реализація екрану списка створених матеріалів

Окрім описаних вище сутностей ще в додатку існують сутності для перегляду матеріалу для прозодження завдань до нього. Вони називаються LessonDetailFragment та QuestionTestFragment. Перший фрагмент відображає матеріал а другий запускає тест до нього.

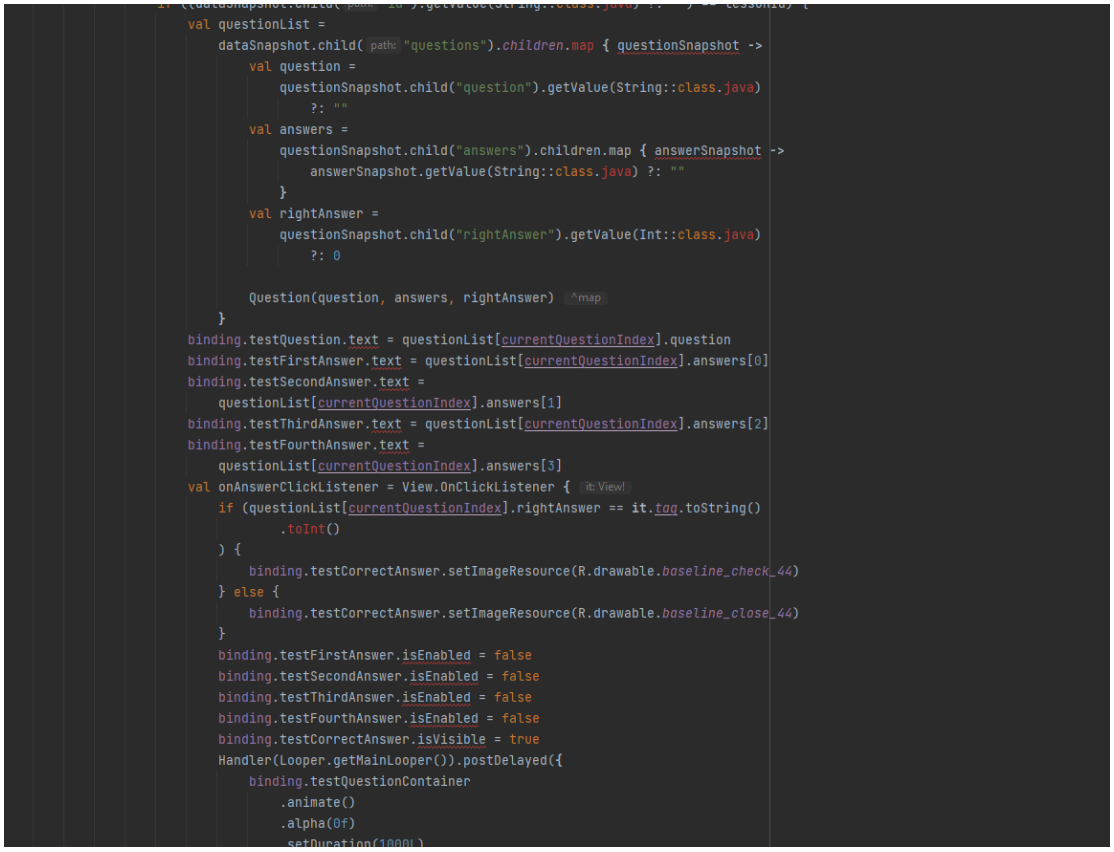
Логіка в LessonDetaiFragment полягає в тому щоб отримати список матеріалів з серверу та знайти в ньому необхідний матеріал за його унікальним ідентифікатором а потім відобразити це на екрані.



```
lessonsRef.addValueEventListener(object : ValueEventListener {
    override fun onDataChange(snapshot: DataSnapshot) {
        snapshot.children.forEach { dataSnapshot ->
            if ((dataSnapshot.child( path: "id").getValue(String::class.java) ?: "") == lessonId) {
                binding.lessonDetailAuthor.text =
                    dataSnapshot.child( path: "creatorName").getValue(String::class.java) ?: ""
                binding.lessonDetailTitle.text =
                    dataSnapshot.child( path: "title").getValue(String::class.java) ?: ""
                val lessonList =
                    dataSnapshot.child( path: "lesson").children.map { lessonSnapshot ->
                        lessonSnapshot.getValue(String::class.java) ?: ""
                    }
                lessonList.filter { it != "null" }.forEach {
                    if (it.takeLast(10) == URI_IDENTIFICATION_CODE) {
                        val imageView = ImageView(requireContext()).apply {
                            layoutParams = ViewGroup.LayoutParams(
                                ViewGroup.LayoutParams.WRAP_CONTENT,
                                ViewGroup.LayoutParams.WRAP_CONTENT
                            )
                            adjustViewBounds = true
                        }
                        val imageRef = storageRef.child( pathString: "${lessonId}/${it}")
                        imageRef.downloadUrl.addOnSuccessListener { uri ->
                            Glide.with(requireContext()) RequestManager
                                .load(uri) RequestBuilder<Drawable!>
                                .into(imageView)
                        }
                        binding.lessonDetailMainContainer.addView(imageView) ^forEach
                    } else {
                        binding.lessonDetailMainContainer.addView(
                            TextView(requireContext()).apply {
                                layoutParams = ViewGroup.LayoutParams(
                                    ViewGroup.LayoutParams.MATCH_PARENT,
                                    ViewGroup.LayoutParams.WRAP_CONTENT
                                )
                                text = it
                            }
                        ) ^forEach
                    }
```

Рисунок 3.19 - Отримання даних окремого матеріалу в LessonDetailFragment

Логіка ж в QuestionTestFragment полягає в запусці інтерактивного тесту до матеріалу. Для цього ми проходимося по списку питань в матеріалі та генеруємо новий екран з питаннями для кожного з елементів списку.



```
val questionList =
    dataSnapshot.child( path: "questions").children.map { questionSnapshot ->
        val question =
            questionSnapshot.child("question").getValue(String::class.java)
                ?: ""
        val answers =
            questionSnapshot.child("answers").children.map { answerSnapshot ->
                answerSnapshot.getValue(String::class.java) ?: ""
            }
        val rightAnswer =
            questionSnapshot.child("rightAnswer").getValue(Int::class.java)
                ?: 0

        Question(question, answers, rightAnswer)  ^map
    }
binding.testQuestion.text = questionList[currentQuestionIndex].question
binding.testFirstAnswer.text = questionList[currentQuestionIndex].answers[0]
binding.testSecondAnswer.text =
    questionList[currentQuestionIndex].answers[1]
binding.testThirdAnswer.text = questionList[currentQuestionIndex].answers[2]
binding.testFourthAnswer.text =
    questionList[currentQuestionIndex].answers[3]
val onAnswerClickListener = View.OnClickListener {  it: View!
    if (questionList[currentQuestionIndex].rightAnswer == it.tag.toString()
            .toInt()
    ) {
        binding.testCorrectAnswer.setImageResource(R.drawable.baseline_check_44)
    } else {
        binding.testCorrectAnswer.setImageResource(R.drawable.baseline_close_44)
    }
    binding.testFirstAnswer.isEnabled = false
    binding.testSecondAnswer.isEnabled = false
    binding.testThirdAnswer.isEnabled = false
    binding.testFourthAnswer.isEnabled = false
    binding.testCorrectAnswer.isVisible = true
    Handler(Looper.getMainLooper()).postDelayed({
        binding.testQuestionContainer
            .animate()
            .alpha(0f)
            .setDuration(1000L)
```

Рисунок 3.20 - Фрагмент коду з генерацією екрану тесту для питань

ВИСНОВКИ

Протягом проекту була успішно розроблена функціональність, яка дозволяє користувачам створювати, виконувати та зберігати інтерактивні освітні завдання. Користувачі можуть додавати завдання, вставляти в них мультимедійний контент, редагувати завдання та матеріали.

В проекті реалізована система реєстрації та аутентифікації користувачів, яка забезпечує безпеку та захист даних. Також реалізована можливість управління користувачами, включаючи додавання та видалення користувачів через консоль Firebase.

Всі дані користувачів, завдання та результати зберігаються в безпечній та сучасній хмарній базі даних з використанням сучасних методів шифрування та забезпечення захисту від несанкціонованого доступу.

Інтерфейс системи адаптується до різних розмірів екранів та орієнтацій пристроїв, що дозволяє користувачам зручно взаємодіяти з додатком на різних пристроях.

Код адаптований до багатьох старих версій Android що дозволяє запускати додаток на широкому спектрі пристроїв під управлінням системи Android.

Система була успішно реалізована на платформі Android з використанням сучасних технологій розробки, таких як Kotlin, AndroidX, та Architecture Components для створення сучасного інтерфейсу.

Загалом така система, як та що була створена в ході роботи, в подальшому при її покращенні може внести позитивні зміни у сферу навчання. Використання зручних мобильніх додатків з цікавим підходом до навчання та відповідними створеними матеріалами може значно покращити:

- Доступність навчального контенту та допомогти учням отримувати якісну освіту незалежно від місця та часу.

- Керування навчальним процесом, подібні системи дозволять відстежувати прогрес учнів та аналізувати їх виконання завдань. Збирати необхідні статистичні дані та аналізувати їх. Це сприяє оптимізації навчання та дозволить вчителям адаптувати матеріал під потреби кожного учня.
- Подбіний комфортний формат навчання з залученням медіа матеріалів дозволить учням вчитись в менш стресових умовах та більше поглиблятись в матеріал. Вони в свою чергу будуть більш зацікавлені в процесі навчання.

Стосовно перспектив розвитку створеної системи можна сказати що вони є досить багатообіцяючими. В подальшому до системи мають бути додані наступні можливості

- Аналіз даних стосовно проходження створених матеріалів. Це допоможе скорегувати процес навчання.
- Додати різні типи завдань до створених матеріалів.
- Додати можливість вставки відео та аудіо контенту в матеріали.
- Створити повноцінню соціальну мережу для користувачів.
- Однією з перспектив є розширення підтримки інших мобільних платформ, таких як iOS. Розробка мобільних додатків для iOS дозволить розширити аудиторію користувачів та зробити систему більш універсальною
- Використання методів машинного навчання та аналітики даних може допомогти в покращенні навчального процесу. Наприклад, система може надавати рекомендації щодо завдань на основі попередніх результатів користувача або вчителя.

Перспективи подальших досліджень у галузі інформаційної системи для створення інтерактивних освітніх завдань є обширними та включають в себе розвиток функціональності, розширення аудиторії, використання

сучасних технологій та аналіз впливу на навчання. Подальший розвиток системи може сприяти покращенню якості освіти та навчання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

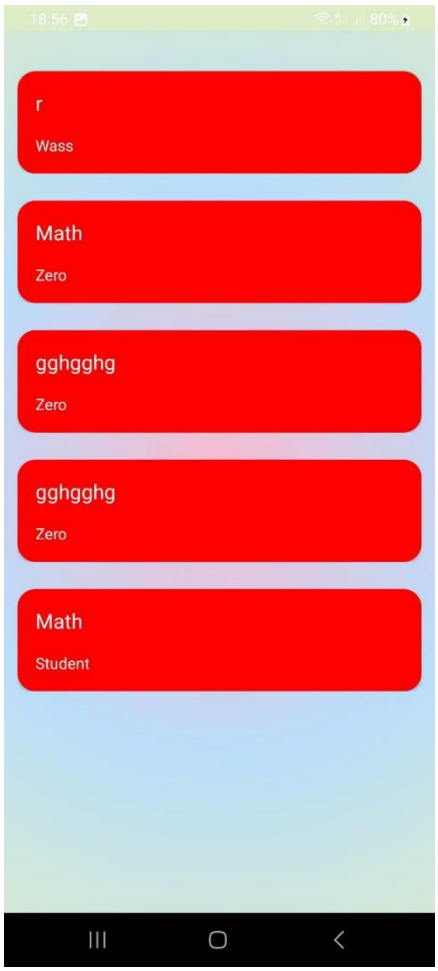
1. Адамс, С. (2012). "The NMC Horizon Report: 2012 Higher Education Edition." с. 23-28.
2. Акаш, Ш. (2023). "The Great Debate: Jetpack Compose vs. XML Layout for Android Application Development". https://www.linkedin.com/pulse/great-debate-jetpack-compose-vs-xml-layout-android-application-shah
3. Альдахван, Н., & Альсаїд, Н. І. (2020). "Use of Artificial Intelligence in Learning Management System (LMS): A Systematic Literature Review". International Journal of Computer Applications. 175, с. 16–26.
4. Аміт, Р. (2023). "MVVM Clean Architecture Pattern in Android with Use Cases". https://medium.com/@ami0275/mvvm-clean-architecture-pattern-in-android-with-use-cases-eff7edc2ef76
5. Ананд, В. Т. (2021). "Are Mobile Learning Apps Changing Modern Education?", с 1-2.
6. Анкіт, Р. (2021). "Dependency Injection: Dagger-Hilt vs Koin". https://medium.com/gradeup/dependency-injection-dagger-hilt-vs-koin-ab2f7f85e6c6
7. Бек, Дж., & Гонг, Ю. (2013), "Wheel-Spinning: Students Who Fail to Master a Skill", с. 431–440.
8. Діпак, С. Р. (2023). "DataStore vs SharedPreferences: A Comparison". https://www.linkedin.com/pulse/datastore-vs-sharedpreferences-comparison-deepak-samuel-rajan#:~:text=DataStore%20supports%20asynchronous%20operations%20using,leading%20to%20poor%20user%20experience.
9. Джонсон, Л., & Адамс, С. (2012). "The NMC Horizon Report: 2012 Higher Education Edition." с. 23-28.
10. Джонс, Б., & Сміт, А. (2018). "Effective Mobile Learning: 50+ Quick Tips and Resources.", с. 14-21.

11. Дю-Булай, Б. (2015). "Recent Meta-reviews and Meta-analyses of AIED Systems". International Journal of Artificial Intelligence in Education, 26 (1), 536–537.
12. Заїд, А., & Рабаб, А. А. Х. (2021). "Justifications for using interactive online education" International Journal of Engineering & Technology. C, 6-1.
13. Інтернет ресурс Android Developers "Документація Android" https://developer.android.com/docs
14. Інтернет ресурс AppInventiv https://appinventiv.com/blog/5-ways-mobile-technology-revolutionized-modern-education-system/
15. Інтернет ресурс ClassVR https://www.classvr.com/virtual-reality-in-education/
16. Інтернет ресурс Firebase. "Документація Firebase" https://firebase.google.com/docs
17. Інтернет ресурс Glide Documentation https://bumptech.github.io/glide/
18. Інтернет ресурс Jetpack Compose Documentation https://developer.android.com/jetpack/compose
19. Інтернет ресурс Koin Documentation https://insert-koin.io/
20. Інтернет ресурс LibHunt https://www.libhunt.com/compare-retrofit-vs-glide
21. Інтернет ресурс Realm Documentation https://realm.io/docs/
22. Інтернет ресурс Wikipedia. "Data model". https://en.wikipedia.org/wiki/Data_model
23. Міньковський, П. (2020). "Kotlin Coroutines vs Java Threads". https://piotrminkowski.com/2020/06/23/kotlin-coroutines-vs-java-threads/
24. Сміт, Дж. (2020). "Interactive Learning Environments: Enhancing Learning with Interactive Resources." с. 45(2), с. 8-15.

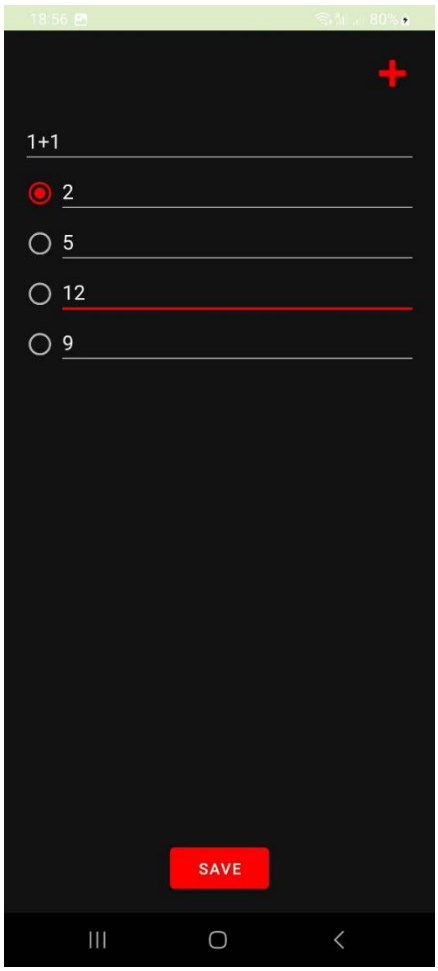
25. Сміт, Дж., & Джонс, Б. (2018). "Effective Mobile Learning: 50+ Quick Tips and Resources.", с. 14-21.
26. Стаття "The Role of Interactive Learning in Educational Technology" (2019). Міжнародний журнал наукових технологій в вищій освіті, с. 16(1), с. 1-10.
27. Сук, Ю. (2023). "10 Trends in Education Technology That Will Have A Major Impact in 2023". https://www.hurix.com/trends-in-education-technology-that-will-have-a-major-impact/
28. Таннер, М., & Шахід, А. (2019). "Selling, Automating and Globalizing Higher Education in the Digital Age", с. 60.

ДОДАТКИ

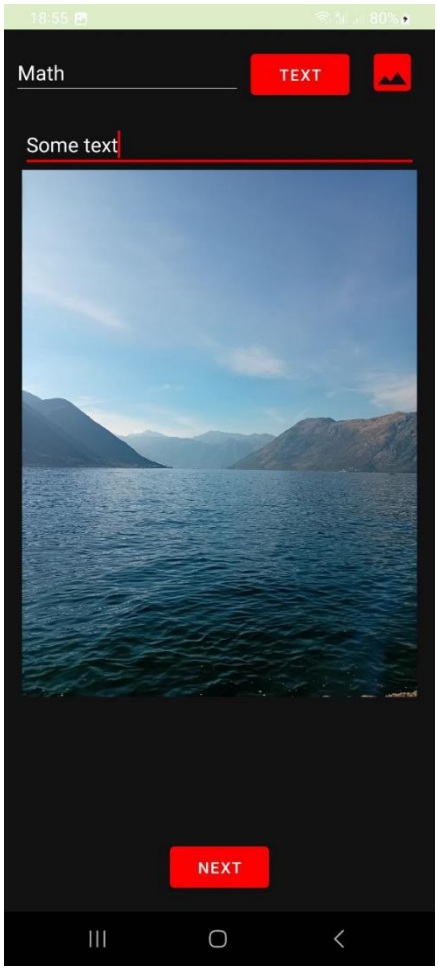
Додаток 1. Приклад роботи створеної системи. Список матеріалів



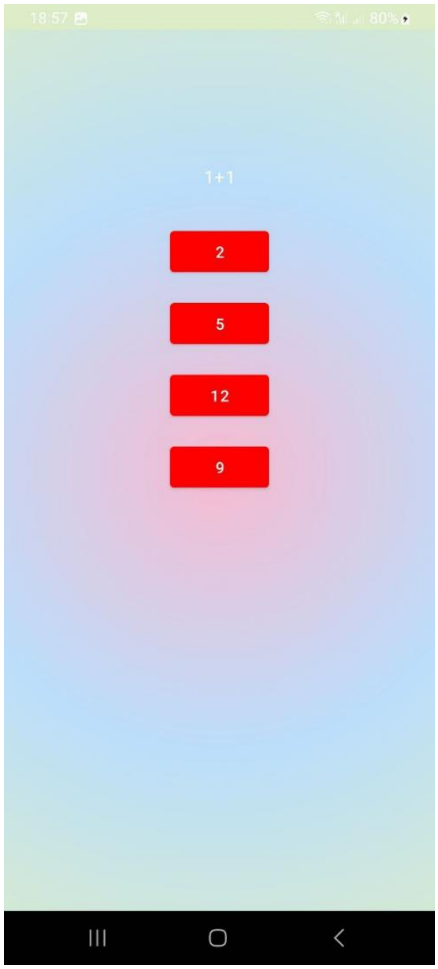
Додаток 2. Приклад роботи створеної системи. Створення завдання до матеріалу



Додаток 3. Приклад роботи створеної системи. Створення матеріалу



Додаток 4. Приклад роботи створеної системи. Проходження завдання до матеріалу



Додаток 5. Посилання на Github репозиторій створеної програми https://github.com/kapok13/EduApp