МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«До захисту допущено»

В.о. завідувача кафедри

Ігор ШЕЛЕХОВ

_____

(підпис)

« »травня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 - Комп'ютерних наук,

освітньо-професійної програми «Інформатика»

на тему: «Інформаційна система організації процесу тренувань на мобільній платформі Android. Клієнтська частина»

здобувача групи ІН - 02 Бровка Павла Віталійовича

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

_____ Павло БРОВКО

(підпис)

Керівник,

к.т.н., доцент кафедри КН

В'ячеслав МОСКАЛЕНКО _____

(підпис)

Суми – 2024

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 бакалавра

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»

здобувача групи ІН-02 Бровка Павла Віталійовича

1. Тема роботи: «Інформаційа система популяризації веб-інклюзивності»
затверджую наказом по СумДУ від *«22» квітня 2024 р. № 0414-VI*

2. Термін здачі здобувачем кваліфікаційної роботи *до «29» травня 2024 року*

3. Вхідні дані до кваліфікаційної роботи

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз проблеми предметної області, постановка й формування завдань дослідження. 2) Огляд стандартів систем тренувань. 3) Розробка клієнтської частини інформаційної системи. 4) Аналіз результатів.*

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

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

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, постановка й формування завдань дослідження*		
2	*Огляд стандартів систем тренувань*		
3	*Розробка клієнтської частини інформаційної системи інформаційної системи*		
4	*Аналіз отриманих результатів*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

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

АНОТАЦІЯ

Записка: 98 стр., 16 рис., 2 додаток, 9 використаних джерел.

Обґрунтування актуальності теми роботи – Тема кваліфікаційної роботи є актуальною, через зростаючий інтерес до здорового способу життя, доступність мобільних технологій, інноваційні можливості для персоналізації тренувань, підтримку соціальних взаємодій та великий економічний потенціал ринку.

Об'єкт дослідження — Система організації процесу тренувань.

Мета роботи — розробити інформаційну систему для мобільних пристроїв, що дозволить користувачам ефективно планувати, виконувати та аналізувати тренування.

Методи дослідження — сучасні стандарти та інструменти створення інформаційних систем на мобільній платформі android.

Результати — розроблено інформаційну систему організації процесу тренувань на мобільній платформі Android, що має такі ключові аспекти:

- спрямованість на користувача
- технологічні інновації
- високу продуктивність
- адаптивний інтерфейс
- зручний та інтуїтивно зрозумілий інтерфейс.

ІНФОРМАЦІЙНА СИСТЕМА, ОРГАНІЗАЦІЯ ПРОЦЕСУ ТРЕНУВАНЬ, ANDROID, JAVA, ANDROID XML.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми роботи. Вибір теми "Інформаційна система організації процесу тренувань на мобільній платформі Android" обумовлений зростанням популярності мобільних пристроїв, попитом на здоровий спосіб життя, технологічним розвитком та великим економічним потенціалом ринку мобільних додатків для здоров'я та фітнесу.

Актуальність. Зростання популярності здорового способу життя та фітнесу зумовлює високий попит на мобільні додатки, які дозволяють ефективно планувати, виконувати та аналізувати тренування. Мобільна платформа Android, як найпоширеніша операційна система для смартфонів, забезпечує широкі можливості для розробки таких додатків, що робить цю тему надзвичайно актуальною.

Об’єкт дослідження є інформаційні системи організації процесу тренувань на мобільних платформах.

Предмет дослідження: методи та засоби розробки клієнтської частини інформаційної системи організації тренувань на платформі Android.

Мета дослідження: розробити інформаційну систему для мобільних пристроїв, що дозволить користувачам ефективно планувати, виконувати та аналізувати тренування.

Гіпотеза. Інформаційна система, розроблена для мобільної платформи Android, дозволить покращити організацію тренувального процесу завдяки індивідуальним програмам, інтеграції з соціальними мережами та використанню сучасних технологій для моніторингу прогресу користувачів.

Новизна роботи полягає у створенні персоналізованих програм тренувань, комплексному підході, що включає планування, виконання та аналіз тренувань, соціальній інтеграції, мобільності та доступності через платформу Android, а також

підтримці тренувань у режимі реального часу з відео-інструкціями та інтерактивними порадами.

Структура роботи. Дана робота складається зі вступу, аналізу проблеми, постановки задачі, вибір методів реалізації поставленої задачі, опису програмної реалізації інформаційної системи, висновків, списку використаних джерел та додатків.

1. АНАЛІЗ ПРОБЛЕМИ

1.1 Сучасний стан та тенденції розвитку електронних систем організації тренувань

У контексті цифрової трансформації, яка охоплює всі сфери життя в Україні, зокрема освіту та фізичну культуру, електронні системи організації тренувань набувають особливої актуальності. Вони широко використовуються в сучасному спорті, надаючи тренерам та спортсменам необхідні інструменти для ефективного планування, аналізу та відстеження тренувальних процесів. Ці системи мають мобільні додатки, веб-платформи та інтегровані пристрої, які синхронізуються з датчиками та трекерами активності.

Переваги таких систем полягають у їх зручності та доступності, дозволяючи користувачам легко отримувати доступ до своїх даних та тренувальних планів. Вони також забезпечують індивідуалізацію тренувань, адаптованих до особистих потреб та цілей кожного спортсмена, та дозволяють проводити глибокий аналіз даних для покращення результатів та відстеження прогресу.

Однак, існують і недоліки, такі як перевантаження інформацією, ризики, пов’язані з конфіденційністю даних, та залежність від технічної справності обладнання та програмного забезпечення.

Тенденції розвитку включають інтеграцію з штучним інтелектом для створення адаптивних тренувальних планів, машинне навчання для аналізу великих обсягів даних та розширення функціональності для забезпечення більш глибокого аналізу фізіологічних показників та здоров’я спортсменів. Ці інновації сприяють підвищенню ефективності тренувального процесу та досягненню кращих спортивних результатів.

Застосування сучасних комп’ютерних технологій у фізичній культурі і спорті відкриває нові можливості для підвищення продуктивності тренерських занять,

передачі навчального матеріалу високої якості та концентрації уваги на ключових аспектах тренувань. Вони можуть бути застосовані на різних етапах тренувального процесу, що сприятиме покращенню результатів.

Отже, розвиток електронних систем організації тренувань є важливим напрямом у сучасній освіті та фізичній культурі. Використання інформаційних технологій може покращити якість навчання та тренувань, сприяючи всебічному розвитку та адаптації до змінливих умов. Сучасні комп’ютерні технології мають великий потенціал у фізичній культурі і спорті, допомагаючи підвищити продуктивність тренувань та досягти кращих результатів.

1.2 Аналіз підходів до проектування frontend-частини систем організації тренувань

Сучасні мобільні додатки для організації тренувань стали невід'ємною частиною повсякденного життя багатьох людей, що прагнуть підтримувати фізичну форму або досягти певних спортивних цілей. Ці додатки зазвичай включають різноманітні функції, такі як відстеження активності, планування тренувань, нагадування про тренування, а також інтеграцію з носимими пристроями.

Проектування frontend частини в додатках для тренувань має ключове значення, оскільки безпосередньо впливає на досвід користувача і його взаємодію з додатком. Сучасні підходи включають реактивний дизайн, який адаптований під різні розміри екранів та орієнтації. Інтуїтивність і доступність також є важливими аспектами, забезпечуючи легко доступні команди, зрозумілі іконки і мінімалістичний інтерфейс. Персоналізація дозволяє користувачам налаштовувати елементи інтерфейсу та зміст програми тренувань відповідно до своїх потреб та вподобань.

1.2.1 Приклади популярних систем.

MyFitnessPal - це мобільний додаток та вебсайт, який дозволяє користувачам відстежувати калорії та фізичну активність з метою контролю ваги або підтримки загального здоров'я.[1]

Основні функції та можливості

Журнал харчування: користувачі можуть вводити щоденні прийоми їжі та напоїв, використовуючи велику базу даних продуктів харчування, яка містить понад 11 мільйонів продуктів. Завдяки функції штрих-коду сканера, можна легко додавати продукти безпосередньо, скануючи упаковки.

Відстеження калорій: додаток автоматично розраховує споживані калорії та порівнює їх із добовою нормою, заснованою на введених користувачем цілях та даних (вага, вік, стать, рівень активності).

Відстеження макронутрієнтів: користувачі можуть налаштувати відсотки споживання білків, жирів та вуглеводів відповідно до своїх дієтичних потреб.

Зв'язок із фітнес-пристроями: MyFitnessPal синхронізується з багатьма фітнес-пристроями та додатками, такими як Fitbit, Garmin та іншими, що дозволяє автоматично імпортувати дані про фізичну активність.

Спільнота та мотивація: додаток надає можливість приєднатися до спільноти, де користувачі можуть обмінюватися порадами, успіхами і навіть викликами для досягнення спільних цілей. Форуми та групи підтримки для різних тем, від втрати ваги до тренувань.

Переваги та недоліки

Переваги:

- велика база даних продуктів і широкі можливості для відстеження харчування;
- легкість використання та високий рівень інтеграції з іншими додатками і пристроями;
- активна спільнота, яка може надавати підтримку та мотивацію.

Недоліки:

- наявність реклами у безкоштовній версії додатку;
- деякі користувачі можуть бути стурбовані приватністю своїх даних через обсяг зібраної інформації;
- необхідність платити за преміальний доступ для отримання додаткових функцій.

Strava - це популярний мобільний додаток та платформа веб-служб, спеціалізована на відстеженні та аналізі фізичної активності, особливо популярна серед бігунів та велосипедистів. Strava пропонує різноманітні інструменти для моніторингу тренувань, а також соціальні функції для залучення спільноти спортсменів.[2]

Основні функції та можливості

Відстеження активності: Strava дозволяє користувачам записувати свої тренування через GPS, відстежуючи такі параметри, як швидкість, темп, відстань, витрачені калорії та загальний час активності.

Сегменти: однією з унікальних особливостей Strava є сегменти, які є специфічними ділянками маршрутів, де користувачі можуть порівнювати свої результати з результатами інших. Це створює елемент змагання та мотивації.

Соціальні функції: користувачі можуть додавати друзів, слідкувати за їхніми тренуваннями, публікувати власні досягнення, ділитися фотографіями та коментувати активності один одного, створюючи сильну спільноту ентузіастів спорту.

Аналіз тренувань: Strava надає детальний аналіз виконаних тренувань, допомагаючи користувачам розуміти свій прогрес та налаштовувати майбутні тренування для покращення результатів.

Виклики та заходи: платформа регулярно проводить виклики, наприклад, пробігти певну кількість кілометрів за місяць, що мотивує користувачів та зберігає їх активними.

Переваги та недоліки

Переваги:

- соціальні функції: Strava відома своїми потужними соціальними функціями, які дозволяють користувачам змагатися з друзями, ділитися своїми досягненнями та слідкувати за прогресом інших;
- аналіз активності: додаток надає детальний аналіз активності, включаючи швидкість, відстань, час та набрану висоту, що є корисним для серйозних спортсменів;

- сегменти: унікальна функція Strava "сегменти" дозволяє користувачам порівнювати свої показники на певних ділянках маршруту з результатами інших користувачів.

Недоліки:

- приватність: існують питання щодо приватності, особливо пов'язані з можливістю інших користувачів бачити активності та маршрути;
- мотивація для професіоналів: незважаючи на те, що додаток популярний серед професіоналів, аматорські користувачі можуть відчувати себе менш залученими через високий рівень конкуренції;
- вартість підписки: деякі корисні функції доступні тільки в платній версії.

Fitbit - це компанія, яка спеціалізується на виробництві носимих технологій для відстеження здоров'я та активності. Основною продукцією компанії є різноманітні моделі фітнес-браслетів та розумних годинників, які допомагають користувачам моніторити різноманітні параметри свого здоров'я, такі як кроки, пройдена відстань, спалені калорії, частота серцебиття, якість сну, та багато інших.[3]

Основні функції та можливості

Відстеження фізичної активності: Fitbit дозволяє користувачам відстежувати основні показники активності, такі як кількість кроків, пройдену відстань, кількість спалених калорій, а також кількість активних хвилин на день.

Моніторинг здоров'я: деякі моделі пристроїв оснащені датчиками для вимірювання частоти серцебиття та рівня кисню в крові, що дозволяє користувачам отримувати детальнішу інформацію про свій фізіологічний стан.

Моніторинг сну: Fitbit аналізує якість та тривалість сну користувачів, надаючи детальні звіти про фази сну (легкий, глибокий, REM) та надаючи рекомендації для покращення якості сну.

Інтеграція з мобільним додатком: вся зібрана інформація синхронізується з мобільним додатком Fitbit, де користувачі можуть аналізувати свої дані, ставити цілі та відстежувати свій прогрес.

Сповіщення та повідомлення: сучасні моделі Fitbit можуть отримувати сповіщення зі смартфона, такі як дзвінки, текстові повідомлення, календарні події та нагадування.

Переваги та недоліки

Переваги:

- інтеграція з фітнес-трекерами: Fitbit дозволяє користувачам легко синхронізувати дані з фітнес-браслетами та годинниками Fitbit, що забезпечує точне відстеження фізичної активності;
- багатофункціональність: додаток пропонує відстеження не тільки фізичної активності, але й сну, споживання води та ваги, створюючи комплексний підхід до здоров'я;
- спільнота та мотивація: Fitbit має велику та активну спільноту користувачів, яка мотивує один одного через виклики та конкурси.

Недоліки:

- залежність від пристроїв: щоб використовувати всі функції додатку, користувачам часто необхідно мати фітнес-трекер Fitbit;
- ціна: ціна фітнес-трекерів Fitbit може бути високою, що обмежує доступність для деяких користувачів;
- точність даних: деякі користувачі відзначають проблеми з точністю відстеження кроків та інших показників.

Причини популярності

Системи організації тренувань на мобільних платформах, такі як MyFitnessPal, Strava, і Fitbit, набули великої популярності з кількох ключових причин.

Зручність використання:

- ці додатки перетворюють смартфони користувачів на потужні інструменти для моніторингу здоров'я та фітнесу, дозволяючи легко відстежувати тренування, калорії, сон та інші важливі показники здоров'я;
- інтеграція з повсякденним життям, дозволяючи користувачам використовувати їх у будь-якому місці і в будь-який час.

Технологічний прогрес:

- розвиток технологій дозволяє додаткам пропонувати все більш точне відстеження та аналіз, використовуючи датчики руху, GPS та інші технологічні інновації;
- вдосконалення алгоритмів обробки даних та штучного інтелекту забезпечують кращий аналіз тренувань та особисті рекомендації.

Соціальні функції:

- додатки створюють віртуальні спільноти, де користувачі можуть ділитися своїми досягненнями, ставити виклики один одному та отримувати мотивацію від товаришів;
- можливість змагань та спільних тренувань з друзями або в групах збільшує втягненість користувачів.

Персоналізація:

- додатки надають індивідуалізовані тренування, поради щодо харчування та програми здоров'я, які адаптуються до особистих потреб та цілей кожного користувача;
- автоматизовані налаштування та аналітика допомагають користувачам оптимізувати свої зусилля та покращувати результати.

Мотивація та втягненість:

- гейміфікація, нагороди, медалі та досягнення допомагають користувачам відчувати прогрес і зберігати інтерес до тренувань;

- регулярне оновлення змісту та функціональності допомагає утримувати інтерес користувачів.

Інтеграція з іншими пристроями та додатками:

- широка сумісність з носимими пристроями, іншими здоров'я-орієнтованими додатками та смарт-пристроями дозволяє користувачам мати єдину інтегровану систему для всіх аспектів свого здоров'я;
- здатність агрегувати дані з різних джерел забезпечує комплексний погляд на здоров'я користувача.

Переваги та недоліки існуючих систем

Переваги:

- інтеграція з носимими пристроями - більшість додатків ефективно інтегруються з різноманітними носимими пристроями, забезпечуючи синхронізацію даних у реальному часі;
- кросплатформеність - додатки часто доступні на різних платформах, що робить їх зручними для широкої аудиторії користувачів;
- використання гейміфікації - це підвищує мотивацію користувачів через введення елементів гри (баджі, рейтинги, змагання).

Недоліки

При аналізі систем організації тренувань на мобільних платформах, як MyFitnessPal, Strava, і Fitbit, можна виділити декілька основних помилок чи недоліків, які впливають на загальне сприйняття та ефективність цих додатків.

Недостатня увага до приватності та безпеки даних:

- багато додатків збирають велику кількість персональних даних, включаючи фізичну активність, місцезнаходження, здоров'я тощо, що ставить під ризик конфіденційність користувачів;

- недостатні заходи забезпечення безпеки даних або непрозорі політики конфіденційності.

Комплексність та перевантаженість інтерфейсу:

- деякі додатки мають складні або непрості інтерфейси, які можуть відштовхнути нових або менш технічно здібних користувачів;
- спроба інтегрувати занадто багато функцій в один додаток без належного дизайнерського балансу.

Залежність від обладнання та зовнішніх пристроїв:

- для повноцінного використання деяких додатків необхідні спеціалізовані пристрої або аксесуари, які можуть бути дорогими;
- бізнес-модель, що зосереджена на продажу додаткового обладнання або аксесуарів.

Підтримка обмеженого числа платформ:

- не всі додатки доступні на всіх мобільних платформах, що обмежує їх доступність для користувачів різних пристроїв;
- обмеження ресурсів на розробку та підтримку декількох платформ.

Недостатній фокус на мотивації користувача:

- додатки, що не забезпечують достатню мотивацію або відчуття досягнення, можуть швидко втратити користувачів;
- недостатня інтеграція гейміфікації, соціальних елементів, або персоналізації.

Аналізуючи існуючі системи, можна побачити, як важливою є балансування між інноваційністю, простотою використання та захистом даних. Ці інсайти можуть бути використані для покращення проектування майбутніх систем організації тренувань на мобільних платформах.

Тому для створення успішної системи для організації тренувань на мобільній платформі, вимагає звернення уваги на ряд ключових аспектів, які сприяють

залученню та задоволенню користувачів. Найважливіші елементи, на які слід звернути увагу при розробці таких систем:

Користувацький досвід (UX) та інтерфейс (UI):

- простота використання - інтерфейс повинен бути інтуїтивно зрозумілим, щоб користувачі могли легко навігувати та використовувати додаток без зайвих складнощів;
- естетичний дизайн - привабливий та сучасний дизайн може значно підвищити інтерес до додатку;
- адаптивність - додаток має правильно відображатися на різних пристроях та розмірах екранів.

Персоналізація:

- індивідуальні налаштування - дозволяти користувачам налаштовувати додаток відповідно до їхніх фізичних характеристик, цілей та уподобань;
- індивідуальні рекомендації - розвивайти алгоритми, які аналізують активність користувача та надають корисні поради та тренування.

Соціальні функції:

- спільнота - створити можливості для користувачів взаємодіяти, ділитися своїми досягненнями та встановлювати зв'язки з іншими людьми, які мають подібні інтереси;
- змагання та виклики - мотивувати користувачів за допомогою дружніх змагань або спільних викликів.

Технічна виконаність:

- надійність - забезпечити стабільну роботу додатку без збоїв та помилок;
- безпека - розробити сильні заходи безпеки для захисту даних користувачів;

- інтеграція з іншими сервісами - підтримка синхронізації з іншими популярними додатками та пристроями.

Монетизація:

- гнучкі опції монетизації - розглянути різні моделі монетизації, такі як підписки, реклама, платні функції;
- відповідність цінності - убезпечити, щоб пропоновані платні функції дійсно надавали значну додаткову цінність.

Збір та аналіз даних:

- аналітика - використовувати зібрані дані для удосконалення додатку, покращення користувацького досвіду та розробки нових функцій;
- зворотний зв'язок користувачів - регулярно збирати та аналізувати відгуки користувачів для постійного поліпшення продукту.

Розробка успішної системи організації тренувань вимагає глибокого розуміння потреб користувачів і створення продукту, який не тільки задовольняє ці потреби, але й перевершує їх очікування, забезпечуючи незабутній досвід.

1.3 Формалізована постановка задачі

Мета проєкту: Розробка мобільного додатку на платформі Android, який дозволить користувачам ефективно планувати, відстежувати та аналізувати свої тренування, включаючи різні види активностей і цілі.

Основні функціональні вимоги:

1) Реєстрація та авторизація користувачів;
2) Створення та управління персональними тренувальними планами;
3) Відстеження активностей (наприклад, біг, плавання, велоспорт) з можливістю збору даних через введення вручну;
4) Аналіз результатів тренувань;

Нефункціональні вимоги:

1) Інтуїтивно зрозумілий і привабливий користувацький інтерфейс;
2) Висока продуктивність додатку з мінімальною затримкою відгуку;
3) Адаптивність інтерфейсу для різних типів пристроїв Android;
4) Забезпечення конфіденційності та безпеки даних користувачів.

Технологічний стек:

- Мова програмування: Java;
- Фреймворк для розробки UI: Android XML;
- База даних: Firebase для синхронізації даних в хмарі;

Процес розробки:

- Використання Agile методологій для гнучкого управління процесом розробки;
- Використання системи контролю версій для ефективної командної роботи.

2. МЕТОДИ РЕАЛІЗАЦІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ ОРГАНІЗАЦІЇ ПРОЦЕСУ ТРЕНУВАНЬ

Для реалізації інформаційної системи організації процесу тренувань на платформі Android нам знадобляться такі інструменти та технології:

Мова програмування: Java - є однією з найпопулярніших мов програмування для розробки Android-додатків і залишається актуальною завдяки своїй стабільності, широкій підтримці спільноти і великій кількості наявних бібліотек та фреймворків.

Інтегроване середовище розробки (IDE): Android Studio - офіційне IDE для розробки Android-додатків, що надає потужні інструменти для дизайну, розробки, дебагінгу та тестування.

UI фреймворки та бібліотеки: Android XML - основний спосіб розробки інтерфейсів користувача для Android-додатків. XML (Extensible Markup Language) використовується для опису інтерфейсу користувача у вигляді розмітки, що дозволяє окремо від логіки додатка розробляти його візуальну структуру.

Серверна частина, бази даних та API: Firebase - платформа від Google, яка може виконувати роль сервера з аутентифікацією, базою даних і багатьма іншими сервісами.

Інструменти для профілювання та оптимізації: Android Profiler в Android Studio - для моніторингу використання CPU, пам'яті та мережі додатком.

Інструменти для контролю версій: Git - для управління версіями коду та співпраці в команді.

Системи керування залежностями: Gradle - потужна система автоматизації збірки, яка інтегрована з Android Studio та дозволяє легко управляти залежностями і версіями бібліотек.

Ці інструменти та технології допоможуть нам створити ефективну, функціональну та візуально привабливу інформаційну систему для організації процесу тренувань на платформі Android.

2.1 Структурні та функціональні схеми інформаційної системи

На Рисунку 2.1 зображено діаграму варіантів використання інформаційної системи користувачем.

Користувач - користувач системи, який має доступ до різноманітних функцій для організації та відстеження своїх тренувань.

Основні функції для Користувача:

- Реєстрація, процес створення облікового запису в системі.
- Перегляд відео, можливість переглядати відеоматеріали з тренуваннями.
- Вибір відео, процес вибору конкретного відео для перегляду.
- Завантаження відео, користувач може завантажити відео.
- Налаштування фільтрів пошуку, можливість налаштування параметрів пошуку для відеоматеріалів.
- Сканування штрих-кодів та QR-кодів, функція для сканування кодів, які можуть містити інформацію про тренувальні засоби або вправи.
- Перегляд особистої інформації, можливість перегляду своїх персональних даних.
- Зміна особистої інформації, можливість оновлювати свої персональні дані.
- Перегляд календаря, функція перегляду тренувального календаря.
- Додавання заміток про тренування, можливість додавати нотатки до тренувального календаря.

Після реєстрації користувачі можуть використовувати свій обліковий запис для доступу до всіх функцій, що включають перегляд відеоматеріалів, сканування кодів, управління особистою інформацією та перегляд календаря заміток.

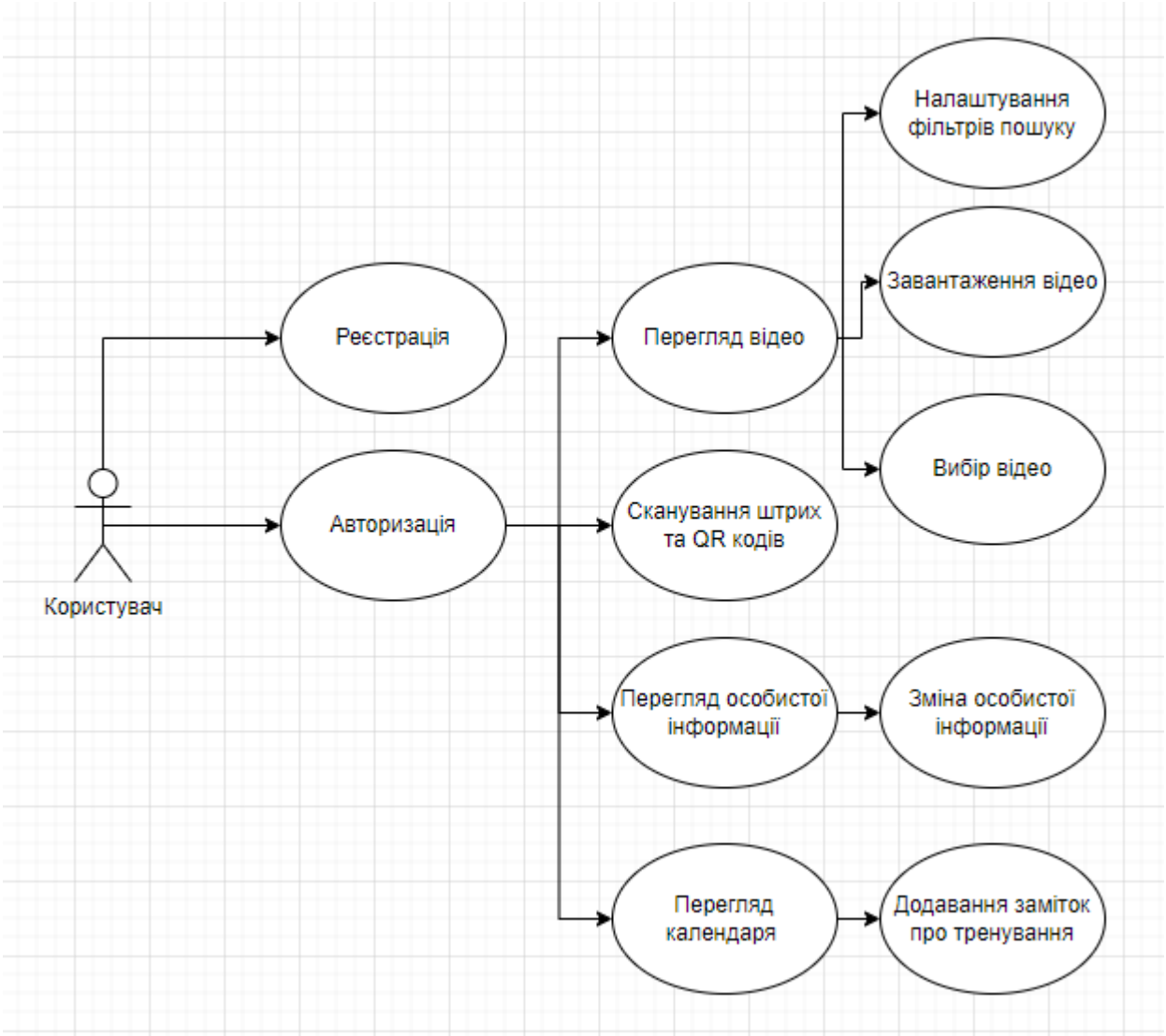

Рисунок 2.1 – UML-діаграма варіантів використання

Діаграма на Рисунку 2.2 ілюструє, як користувач взаємодіє з системою через різні сторінки, забезпечуючи зручний доступ до необхідних функцій для організації та відстеження тренувань.

Користувач може перейти на сторінку авторизації для входу в систему або реєстрації. Після успішної авторизації або реєстрації користувач потрапляє на головну сторінку інформаційної системи. З головної сторінки користувач може переходити до інших функціональних сторінок, таких як сторінка профілю

користувача, сторінка вибору відео, сторінка сканування продуктів та сторінка календаря з замітками.

Сторінка профілю користувача дозволяє налаштовувати особисту інформацію через відповідну форму. Сторінка вибору відео включає підсторінки для завантаження відео та налаштування фільтрів пошуку. Сторінка сканування продуктів має підсторінку для сканування штрих-кодів та QR-кодів. Сторінка календаря з замітками включає функціонал додавання нових заміток про тренування.

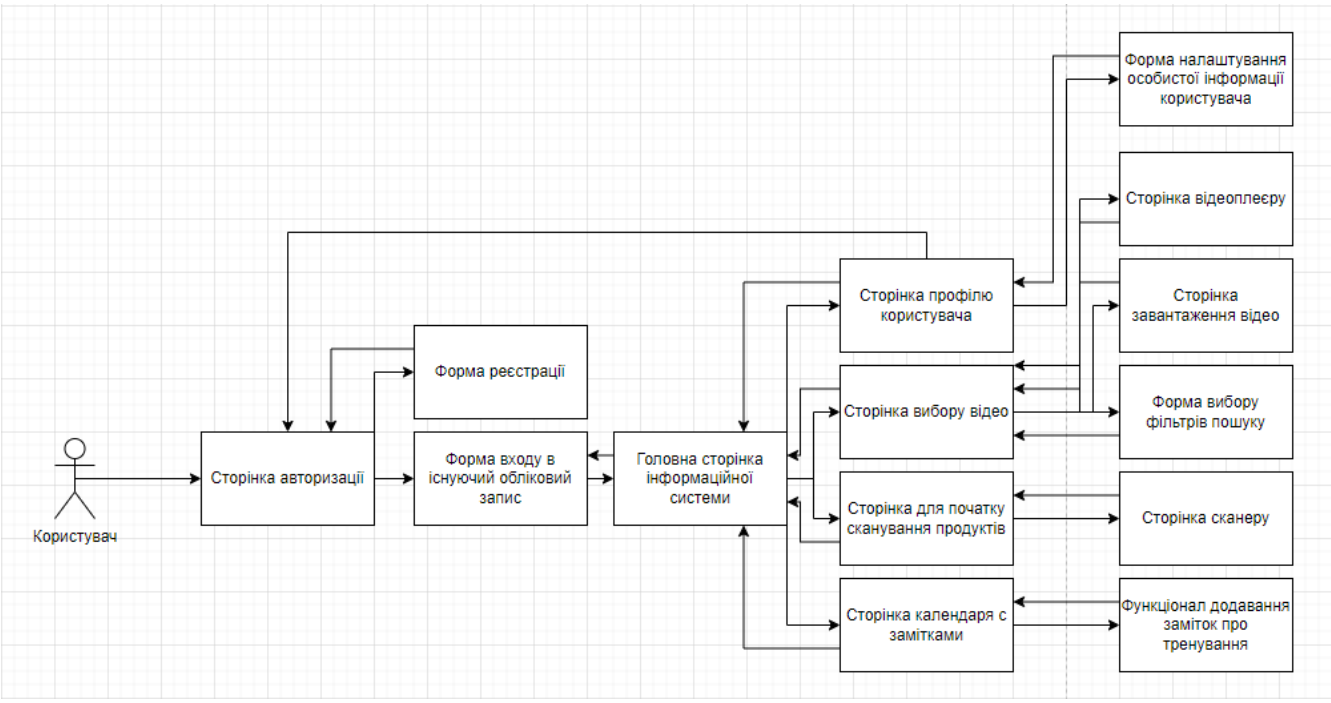

Рисунок 2.2 – UML-діаграма активностей користувача

3. ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Мова програмування Java

Мова програмування Java має значний вплив на світ мобільних технологій, особливо в контексті розробки додатків для Android. Опанувавши глобальний ринок мобільних пристроїв, Java стала однією з основних мов програмування, яка використовується для створення мільйонів додатків, від простих утиліт до комплексних корпоративних систем. Давайте поглиблено розглянемо особливості Java, її вплив на розробку для Android, виклики, з якими стикаються розробники, а також технічні нюанси та найкращі практики, пов'язані з використанням цієї мови.

Java була розроблена в середині 90-х років компанією Sun Microsystems і з самого початку проєктувалася як мова, здатна працювати на різних платформах завдяки архітектурі Java Virtual Machine (JVM). Ця особливість зробила її ідеальним вибором для мобільних пристроїв з різноманітним обладнанням. Коли Google вибрав Java як основу для платформи Android, це не тільки забезпечило широку підтримку з боку розробників, але й забезпечило легкість переносу наявного програмного забезпечення на нову мобільну платформу.

Java є строго об'єктно-орієнтованою мовою, що означає, що все представлено у вигляді об'єктів. Ця парадигма спрощує розробку великих систем та дозволяє легко управляти складністю програми. Однією з важливих характеристик Java є автоматичне управління пам'яттю. Java використовує сміттєвий збірник, який автоматично видаляє об'єкти, що більше не використовуються, тим самим звільняючи ресурси та запобігаючи витоку пам'яті. Це особливо важливо для пристроїв з обмеженими ресурсами, таких як мобільні телефони.

Java пропонує суворий контроль доступу, використовуючи модифікатори доступу, такі як private, protected та public. Механізми безпеки на рівні класів та пакетів допомагають уникнути небажаного доступу до даних. Ще однією важливою

особливістю Java є система винятків, яка дозволяє розробникам управляти помилками та іншими винятковими ситуаціями в структурованому та контрольованому порядку. Це допомагає забезпечити стабільність додатків, обробляючи помилки, що можуть виникнути під час виконання програми.

Java користується великою популярністю серед розробників завдяки своїй простоті, об'єктно-орієнтованій структурі та широкому використанню у корпоративних середовищах. Це означає, що багато розробників можуть легко перейти на розробку мобільних додатків. Однією з важливих переваг Java є стабільні API. Android API, засновані на Java, стабільні та добре документовані, що сприяє легкості розробки та тестування. Розробники можуть використовувати розгалужену систему бібліотек і фреймворків, доступних для Java, щоб розширити функціональність своїх додатків.

Java також відзначається мультиплатформенністю, дозволяючи розробникам створювати код, який можна легко перенести на різні платформи, не тільки на Android. Це ідеально підходить для створення крос-платформних рішень та послуг. Інструменти розробки, такі як інтегроване середовище розробки (IDE) Eclipse та IntelliJ IDEA, а також Android Studio, яке було спеціально адаптоване для Android від IntelliJ, надають розширені можливості для дебагінгу, профілювання та оптимізації Java-коду.

Використання Java на Android має свої виклики. По-перше, продуктивність може стати проблемою. Хоча Java вважається досить швидкою мовою, використання JVM може призводити до затримок у виконанні, особливо на пристроях з обмеженими ресурсами. Компіляція в байт-код, яка потім виконується на JVM, може бути менш ефективною порівняно з нативним кодом, який використовується в мовах програмування, таких як C++ чи Kotlin, де код компілюється безпосередньо під конкретну платформу.

Другою проблемою є пам'ять і управління ресурсами. Java автоматично керує пам'яттю через сміттєзбірник, що може спричиняти непередбачувані затримки, коли збірник активується для очищення не використовуваних об'єктів. Це може призводити до проблем з продуктивністю в критичних за часом додатках.

Третій виклик пов'язаний з безпекою. Хоча Java має сильні можливості забезпечення безпеки, вона також відома своїми вразливостями, особливо в старих версіях. Управління безпекою вимагає регулярного оновлення та підтримки з боку розробників.

Останнім викликом є обмеження, пов'язані з версіями Java. Android підтримує не всі найновіші версії Java, що може обмежувати використання новітніх функцій мови. Внаслідок цього, розробники можуть зіткнутися з обмеженнями, які впливають на архітектуру додатка та його можливості.

Найкращі практики розробки на Java для Android включають кілька ключових аспектів. По-перше, оптимізація коду є критично важливою. Слід зосередитися на оптимізації використання пам'яті та продуктивності. Використання профілів, таких як Android Profiler у Android Studio, допоможе виявити "вузькі місця" в додатках і визначити можливості для оптимізації. По-друге, важливо використовувати новітні бібліотеки та фреймворки. Використання останніх версій не тільки забезпечить доступ до нових функцій, але й підвищить безпеку та продуктивність додатків.

Третім важливим аспектом є тестування і дебагінг. Регулярне тестування допоможе забезпечити стабільність та надійність вашого додатку. Використання юніт-тестів, інтеграційних тестів та UI тестів є ключовими для виявлення і усунення помилок на ранніх стадіях розробки. Нарешті, слід враховувати архітектуру додатку. Проектування архітектури додатка з урахуванням масштабованості, підтримки та можливості легкого оновлення є критично важливим. Патерни проектування, такі як MVC, MVP або MVVM, допоможуть організувати код таким чином, що його буде легше підтримувати та оновлювати.

Java залишається могутнім інструментом для розробників Android, надаючи велике середовище, яке сприяє інноваціям та ефективній розробці. Однак, як і будь-яка технологія, вона вимагає ретельного врахування своїх особливостей та обмежень для створення успішних продуктів.

3.2 Інтегроване середовище розробки Android Studio

Android Studio є офіційним інтегрованим середовищем розробки (IDE) для платформи Android, створеним Google. Заснований на IntelliJ IDEA від JetBrains, Android Studio пропонує розробникам комплексний набір інструментів, що покращують продуктивність, забезпечують легшу розробку додатків та сприяють створенню якісніших продуктів.

Перша версія Android Studio, представлена на Google I/O у 2013 році, була розроблена з метою вирішити низку проблем, які існували в попередньому середовищі розробки, Eclipse. Використання Eclipse з ADT (Android Developer Tools) Plugin часто супроводжувалось складнощами з продуктивністю, гнучкістю та інтеграцією сучасних технологій. Eclipse був загальнопризначеним IDE, яке не було оптимізовано спеціально для розробки Android, що призводило до низки обмежень в розробці.

З моменту свого введення Android Studio пройшло через численні значущі оновлення, кожне з яких мало на меті поліпшення досвіду розробників і розширення можливостей IDE. Одним з найважливіших оновлень стала інтеграція Kotlin. Оголошення Kotlin офіційною мовою розробки Android у 2017 році стало важливим кроком в історії Android Studio. Це забезпечило повну підтримку Kotlin, включаючи інструменти для конвертації коду з Java в Kotlin, що зробило перехід на нову мову простішим та доступнішим для розробників.

Ще одним значним оновленням стало введення Android Profiler, який замінив старі інструменти моніторингу, такі як DDMS, надаючи більш інтуїтивно зрозумілі та потужні засоби для аналізу продуктивності додатків. Розробники тепер могли детально відслідковувати використання ЦПУ, пам'яті, мережі та енергії.

Редизайн інтерфейсу та поліпшення редактора макетів забезпечили кращу візуалізацію, покращену підтримку ConstraintLayout і більш зручне перетягування

компонентів UI. Це значно полегшило створення візуально привабливих і адаптивних інтерфейсів.

Впровадження Android App Bundles як нового формату публікації додатків у Google Play дозволило розробникам зменшити розмір додатків завдяки більш ефективній доставці контенту до кінцевих користувачів.

Введення підтримки Jetpack Compose, сучасної декларативної UI бібліотеки для Android, стало ще одним значним оновленням в Android Studio, яке спрямоване на полегшення розробки інтерфейсу з використанням Kotlin.

Android Studio значно підвищило ефективність процесу розробки на Android, надаючи розробникам одне інтегроване середовище, яке постійно оновлюється і підтримується. Це IDE спрощує робочі процеси, автоматизує рутинні завдання і забезпечує інструменти для кращого дизайну і тестування додатків.

Подальший розвиток Android Studio, ймовірно, буде зосереджений на підтримці нових технологій, поліпшенні інтеграції з різними сервісами і платформами, а також на забезпеченні більш ефективного і доступного досвіду для всіх розробників Android. Наразі Android Studio залишається в центрі екосистеми розробки Android, пропонуючи розробникам найкращі інструменти та ресурси для створення інноваційних і високоякісних додатків.

Основні компоненти та функції Android Studio

Редактор коду в Android Studio вирізняється своєю високою адаптивністю та потужними функціями, які значно полегшують розробку додатків. Однією з ключових функцій є автозаповнення коду (Code Completion), яке дозволяє розробникам швидше писати менш помилковий код, підтримуючи як стандартні мови програмування, такі як Java та Kotlin, так і мови розмітки, такі як XML. Підсвічування синтаксису (Syntax Highlighting) допомагає розробникам швидше читати та розуміти код, відокремлюючи змінні, типи, методи та інші елементи за допомогою різних кольорів та шрифтів. Крім того, редактор підтримує розширену

підтримку для різних мов програмування, що дозволяє розробникам використовувати міжмовні можливості, наприклад, взаємодію Kotlin з Java у тому самому проекті.

Візуальний редактор макетів є однією з найбільш вражаючих особливостей Android Studio. Розробники можуть легко створювати інтерфейси, перетягуючи віджети з палітри в область редагування, що значно прискорює процес дизайну. Редактор макетів також допомагає перевіряти, як UI додатка буде виглядати на різних розмірах екранів та орієнтаціях, що є критично важливим для створення адаптивних додатків. Крім того, редактор надає підтримку ConstraintLayout, що полегшує створення складних макетів з гнучкою настройкою відносин між елементами інтерфейсу без зайвого нагромадження вкладених макетів.

Android Profiler надає важливі інструменти для виявлення та виправлення проблем продуктивності. Централізоване профілювання дозволяє розробникам відслідковувати використання ресурсів у реальному часі, виявляючи піки використання ЦПУ, споживання пам'яті, трафік даних та використання батареї. Крім того, Profiler дозволяє аналізувати трасування стеку викликів і таймінги, що допомагає розробникам оптимізувати продуктивність додатка.

Gradle є потужною системою збірки, яка інтегрована в Android Studio. Вона забезпечує автоматизацію збірки, автоматизуючи багато аспектів процесу збірки, включаючи компіляцію коду, упаковку ресурсів, підписання APK та його розгортання. Крім того, Gradle надає можливість управління залежностями і версіями бібліотек, що забезпечує гнучкість і контроль над складними проектами.

Екосистема Android Studio включає велику кількість плагінів, які розширюють можливості IDE та забезпечують додаткові функції. Серед них є плагіни для інтеграції з системами контролю версій (VCS), такими як Git і SVN. Також існують плагіни для підтримки додаткових мов програмування, наприклад, Python або JavaScript. Інструменти для баз даних, які забезпечують інтеграцію з

базами даних та допомагають управляти даними прямо з IDE, також доступні через плагіни.

Android Studio, як і багато комплексних програмних рішень, може виявляти проблеми зі стабільністю, особливо під час використання бета-версій або під час інтеграції з великою кількістю сторонніх плагінів. Ці проблеми можуть включати неочікувані збої, затримки в реакції інтерфейсу або неправильну роботу окремих компонентів IDE. Для мінімізації цих ризиків рекомендується використовувати стабільні версії і уважно перевіряти сумісність плагінів.

Хоча Android Studio і не вимагає постійного з'єднання з Інтернетом, багато з його функцій, особливо завантаження залежностей і плагінів, а також інтеграція з Firebase чи Google Cloud Platform, потребують доступу до мережі. Це може стати перешкодою в регіонах з обмеженим або нестабільним Інтернет-з'єднанням.

Перша налаштування Android Studio і створення нового проекту можуть бути часомісткими процесами, особливо на комп'ютерах, які не відповідають рекомендованим системним вимогам. Іноді користувачам необхідно вручну налаштовувати додаткові параметри, такі як розміри пам'яті JVM, щоб оптимізувати продуктивність IDE.

Хоча Android Studio забезпечує інструменти для розробки додатків, що підтримують різні версії Android, ведення підтримки старих версій (наприклад, Android 4.4 та нижче) може бути складним. Деякі нові функції і компоненти API можуть бути несумісними зі старими версіями, що вимагає додаткових зусиль для забезпечення зворотної сумісності.

Хоча емулятор Android, вбудований в Android Studio, є потужним інструментом для тестування додатків, він також може бути вимогливим до системних ресурсів. Емуляція пристроїв, особливо з високою роздільною здатністю екрану та передовими характеристиками, може сповільнювати роботу навіть потужних систем.

Android Studio є ключовим інструментом у арсеналі розробника Android. Його глибока інтеграція з Android SDK, постійні оновлення від Google та велика спільнота користувачів роблять його незамінним інструментом для створення якісних додатків. Завдяки своїм потужним функціям, Android Studio не тільки підвищує ефективність розробки, але й допомагає розробникам відповідати високим вимогам сучасного ринку мобільних додатків.

3.3 UI фреймворки та бібліотеки Android XML

Android XML є фундаментальним інструментом для розробки користувацького інтерфейсу (UI) в додатках для Android. Використовуючи XML, розробники можуть декларативно описати UI елементи, що дозволяє їм більш чітко та ефективно керувати виглядом і взаємодією своїх застосунків.

Android XML використовується для опису розкладок (layouts), які є шаблонами для UI додатку. Розкладки визначають структурну схему елементів інтерфейсу, таких як кнопки, текстові поля, списки і зображення. Основні типи розкладок включають LinearLayout, RelativeLayout і ConstraintLayout, кожен з яких має свої особливості для розміщення компонентів UI.

У Android XML також можна визначати різні ресурси, як-от кольори, розміри, стилі та строки. Це сприяє кращій організації коду і робить додаток більш адаптивним та легшим для масштабування. Ресурси зберігаються у папці res і можуть бути легко змінені без необхідності внесення змін у основний код.

Android XML дозволяє розробникам створювати гнучкі і адаптивні інтерфейси, які можуть адаптуватися до різних типів пристроїв і розмірів екранів. Це досягається завдяки використанню розмірних кваліфікаторів, які дозволяють визначати різні ресурси для різних умов використання, таких як орієнтація екрану або розмір екрану.

Android Studio має вбудовані інструменти для розробки Android XML, включаючи візуальний редактор макетів, який дозволяє розробникам перетягувати компоненти на канву та автоматично генерувати відповідний XML код. Це значно спрощує процес проектування UI, оскільки розробники можуть візуально відтворити вигляд свого додатку без потреби ручного кодування всіх елементів.

XML дозволяє описати UI в декларативній формі, що полегшує розуміння структури додатку. Використання ресурсів та розкладок робить додаток більш розширюваним та піддається змінам без необхідності внесення змін у код.

Інтеграція з Android Studio дозволяє зручно розробляти та відлагоджувати UI без переходу до інших інструментів.

Однак для новачків Android XML може бути складним у освоєнні через велику кількість різних елементів та їх взаємодію. XML може стати важким для обслуговування у великих проектах, особливо якщо він містить велику кількість елементів та ресурсів.

Android XML є ключовим інструментом у розробці Android-додатків, і використовується в кожному етапі розробки, від проектування інтерфейсу до реалізації логіки та відлагодження додатку. Він дозволяє розробникам створювати інтерфейси, які є адаптивними, гнучкими та естетично приємними, що є ключовими аспектами для успішного додатку.

Загалом, Android XML є невід'ємною частиною процесу розробки Android-додатків, і з його допомогою розробники можуть створювати додатки, які не тільки ефективно виконують свої функції, але й вражають користувачів своїм зовнішнім виглядом та взаємодією.

3.4 Інструменти для контролю версій Git

Git є однією з найпопулярніших систем управління версіями, яка широко використовується в індустрії програмного забезпечення для контролю змін у коді під час розробки програм. Винайдений Лінусом Торвальдсом, творцем Linux, у 2005 році, Git розроблявся з метою бути швидким, ефективним та надійним інструментом для розгалуженого ведення розробки.

Git є розподіленою системою контролю версій, де кожен розробник має локальну копію всієї історії розробки. Це дозволяє працювати локально та здійснювати більшість операцій без необхідності постійного доступу до мережі або центрального сервера. Git підтримує створення численних розгалужень (branches) і злиття (merges), що дозволяє розробникам експериментувати з новими ідеями без ризику для актуальної версії проекту. Розгалуження можуть бути легко створені та інтегровані, що сприяє ітеративній розробці та співпраці. Git оптимізований для швидкості та мінімізує використання дискового простору. Він виконує більшість операцій локально, що знижує час, необхідний для обробки команд, і використовує систему відбитків для ефективного зберігання змін.

Git є ідеальним для проектів, де декілька розробників одночасно працюють над одними і тими ж файлами. Він дозволяє легко інтегрувати роботу кількох осіб через механізм злиття, забезпечуючи при цьому високу точність та контроль над змінами. Завдяки розподіленій структурі, кожна локальна копія проекту є повноцінним репозиторієм зі своєю історією та версіями. Це забезпечує високий рівень захисту від втрати даних в разі пошкодження або втрати сервера. Git ефективно працює як з маленькими, так і з великими проектами, забезпечуючи швидкий доступ до історії та змін, незалежно від обсягу проекту або кількості включених змін.

Git не просто інструмент для контролю версій; це невід'ємна частина сучасних методологій розробки програмного забезпечення. Він сприяє кращій

організації, співпраці та безпеці проектів, роблячи процес розробки більш ефективним і контрольованим.

3.5 Опис візуального інтерфейсу інформаційної системи організації процесу тренувань

На Рисунку 3.1 зображена іконка інформаційної системи на якій зображено зайця як символ швидкої та спортивної тварини.

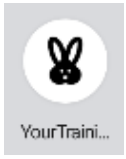

Рисунок 3.1 – Іконка інформаційної системи

На Рисунку 3.2 зображенні є дві кнопки: "Увійти" та "Зареєструватися", що дозволяє користувачам вибрати між входом в існуючий обліковий запис та створенням нового. Інтерфейс чистий та не перевантажений, що сприяє легкому сприйняттю. Кнопки добре виділені і знаходяться в нижній частині екрану, що є звичним місцем для дій користувача.

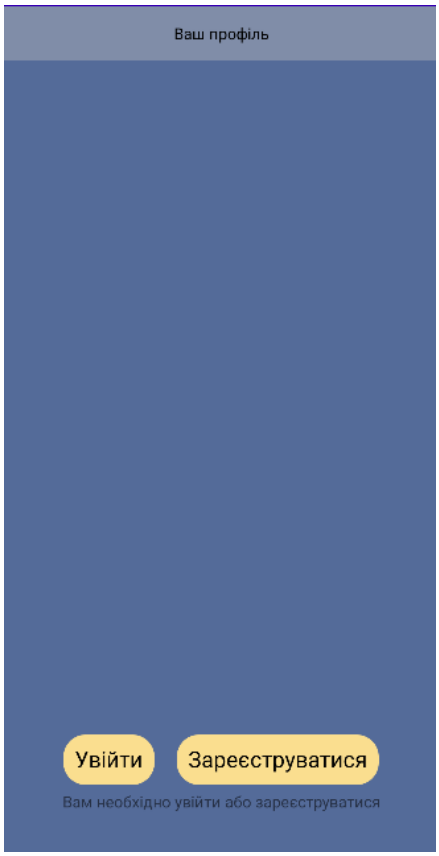

Рисунок 3.2 – Сторінка авторизації

На Рисунку 3.3 зображено інтерфейс реєстрації користувача для мобільного додатку на Android. Інтерфейс виглядає простим і незавантаженим, що є плюсом для зручності користувачів. Поля для введення (Email, Пароль, Ім'я, Телефон) чітко розмежовані, що сприяє легкому введенню даних. Великі заголовки та розділові відступи допомагають користувачеві легше навігувати по формі.

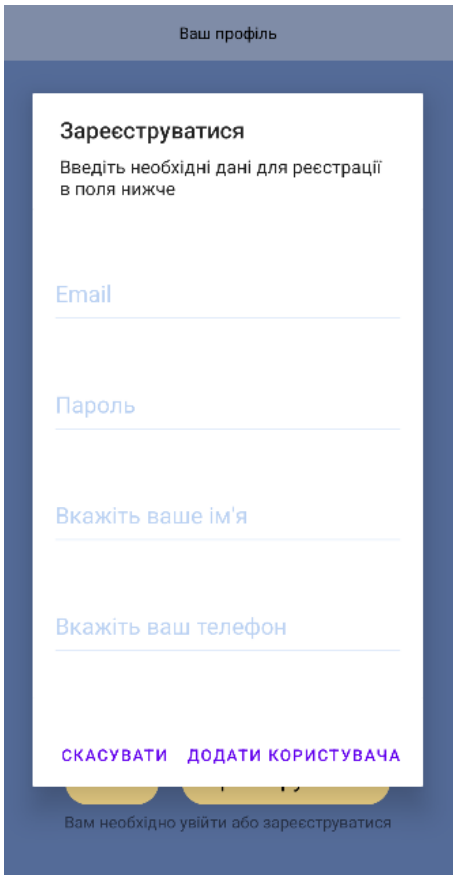

Рисунок 3.3 – Форма реєстрації

На Рисунку 3.4 показано екран для входу в мобільний додаток, який включає поля для введення Email та паролю, а також кнопки для входу та скасування. Цей дизайн знову ж таки демонструє мінімалістичний підхід.

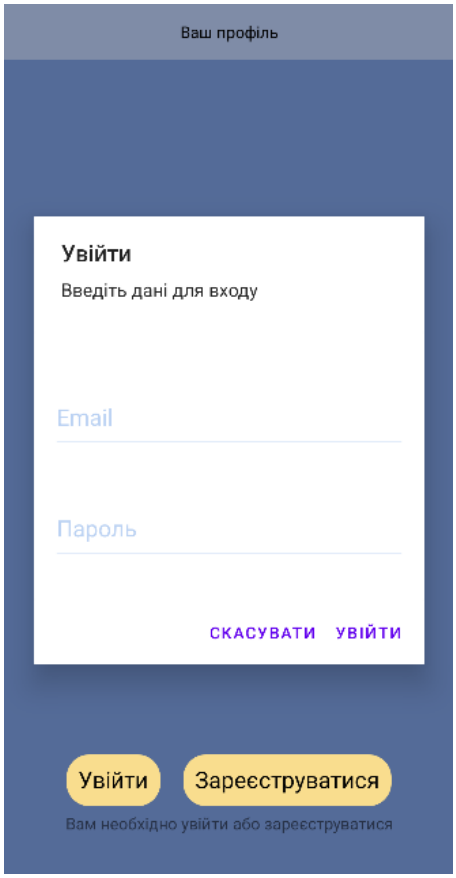

Рисунок 3.4 – Форма входу в існуючий обліковий запис

На Рисунку 3.5 представлений інтерфейс мобільного додатку. У верхній частині екрана розміщено зображення профілю користувача та напис "Test". Ця панель служить для відображення інформації про поточного користувача.

У центрі екрану розміщені три великі круглі кнопки. В центральній частині знаходиться найбільша кнопка з іконкою відтворення відео. Ця кнопка призначена для доступу до відеоматеріалів. Ліворуч розташована середнього розміру кнопка з іконкою календаря. Вона надає доступ до календаря-планувальника тренувань. Праворуч розташована середнього розміру кнопка з іконкою QR-коду. Ця кнопка використовується для сканування QR-кодів.

Фон інтерфейсу виконаний у синьому кольорі, що створює контраст із жовтими кнопками, роблячи їх більш видимими та зручними для взаємодії. Цей

інтерфейс спроектований для швидкого доступу до основних функцій додатку: перегляду відео, управління календарем та сканування QR-кодів.

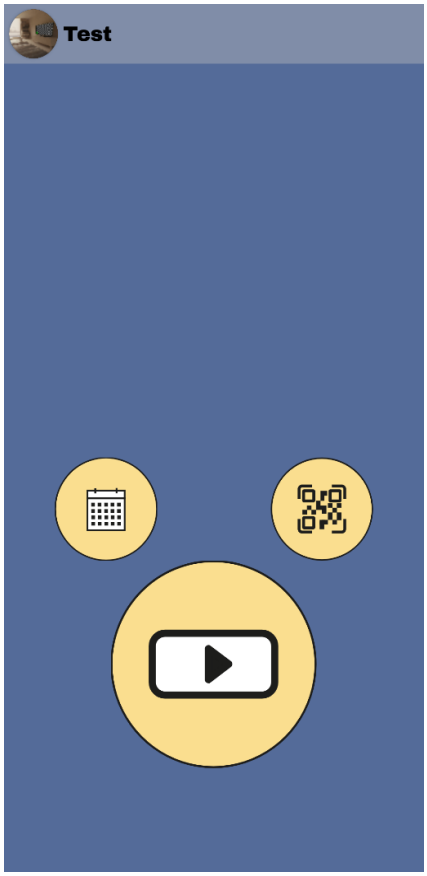

Рисунок 3.5 – Головна сторінка інформаційної системи

На Рисунку 3.6 показано профіль користувача у мобільному додатку. Це дизайн, який використовує простоту та естетичні зображення для представлення особистої інформації. Велике зображення, у центрі, може бути символом особистості користувача або вибраною аватаркою. На рисунку інформація чітко розміщена: ім'я користувача, електронна адреса та номер телефону легко читаються. Інтуїтивні іконки навігації вгорі (стрілка назад і редагування) та іконка внизу (вихід) є стандартними та легко впізнаваними, що полегшує користування.

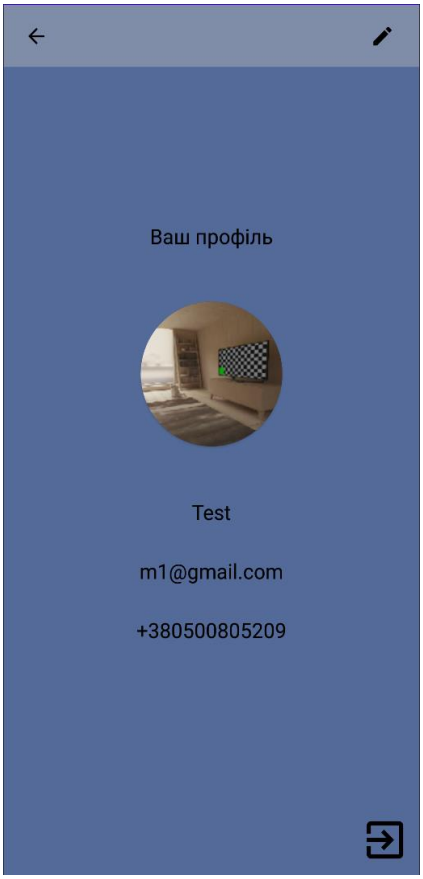

Рисунок 3.6 – Сторінка профілю користувача

На Рисунку 3.7 відображено екран налаштувань профілю користувача, де користувачі можуть змінювати свої основні дані, такі як ім'я, пароль та номер телефону. Дизайн продовжує слідувати мінімалістичному та візуально чистому стилю. Меню налаштувань містить чітко визначені опції, кожна з яких є прямолінійною та зрозумілою. Використання подібного фону, шрифтів і макету з попередніми екранами забезпечує єдність користувацького інтерфейсу.

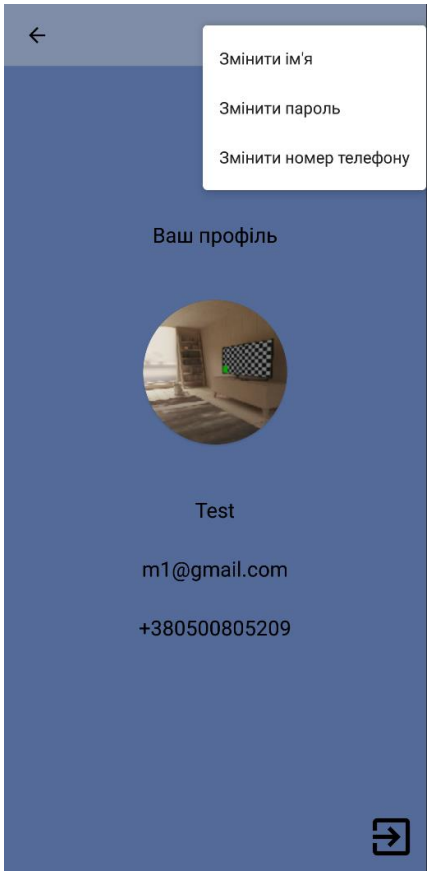

Рисунок 3.7 – Форма налаштувань профілю користувача

На Рисунку 3.8 відображений дизайн екрану сканера QR-кодів для мобільного додатку. Чіткий і простий інтерфейс з основним увагою на QR-код. Великий QR-код у центрі екрану відразу привертає увагу, що сприяє виконанню головної функції екрану. Іконка "назад" вгорі вказує на легке повернення до попереднього екрану.

Рисунок 3.8 – Сторінка для початку сканування продуктів

На Рисунку 3.9 представлений інтерфейс календаря мобільного додатку. У верхній частині екрана розташована панель навігації зі стрілкою назад для повернення на попередній екран. Далі розташований індикатор місяця і року (May 2024) з кнопками стрілок ліворуч і праворуч для перемикання між місяцями.

Основна частина екрана зайнята календарем на поточний місяць. Дні тижня позначені літерами (S, M, T, W, T, F, S). Поточний день (27 травня 2024 року) виділений синім кольором. У календарі відображаються всі дні місяця, а також можна побачити іконку у вигляді списку і пера для певних днів, що, ймовірно, означає наявність запланованих подій або нотаток.

Під календарем розміщено текстове поле з електронною адресою користувача (m1@gmail.com) та дві кнопки: "Редагувати" (значок олівця) і "Видалити" (значок

кошика). Ці кнопки, ймовірно, дозволяють редагувати або видаляти нотатки або події, пов'язані з обраним днем.

Нижче розташоване текстове поле з написом "Введіть замітку" для введення нової замітки. Під текстовим полем розміщена кнопка "Додати замітку", оформлена в жовтому кольорі, що робить її помітною і зручною для взаємодії.

Цей інтерфейс дозволяє користувачам переглядати календар, додавати нові замітки, а також редагувати або видаляти існуючі замітки, забезпечуючи зручність планування та організації подій.

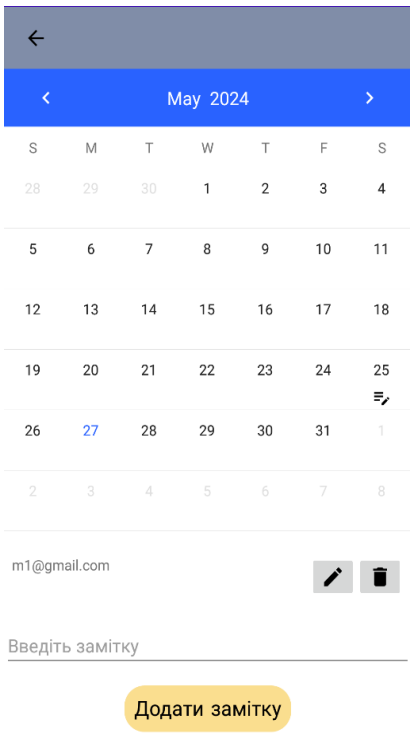

Рисунок 3.9 – Сторінка календаря з замітками

На Рисунку 3.10 представлений інтерфейс списку відео в мобільному додатку. У верхній частині екрана розташована панель навігації зі стрілкою назад для повернення на попередній екран. Поруч розміщені іконки для завантаження та

фільтрації відео. Справа знаходиться кнопка "Скинути фільтри", оформлена в жовтому кольорі, яка дозволяє скинути всі застосовані фільтри.

Основна частина екрана зайнята списком відео, кожне з яких представлено мініатюрою (thumbnail), назвою та тривалістю. Відео розміщені у вертикальному списку, і кожен елемент містить наступну інформацію: мініатюра відео, тобто зображення з відео для швидкої візуальної ідентифікації; назва відео, тобто текстове поле з назвою відео (наприклад, "video1", "t11"); тривалість відео, яка відображена у правому нижньому кутку мініатюри (наприклад, "00:50", "00:45").

Цей інтерфейс забезпечує користувачам зручний перегляд доступних відео, дозволяючи швидко знаходити потрібні відеоматеріали, переглядати їхню тривалість та використовувати функції сортування і фільтрації для покращення пошуку. Кнопка "Скинути фільтри" дозволяє легко повернутися до початкового списку відео без застосованих фільтрів.

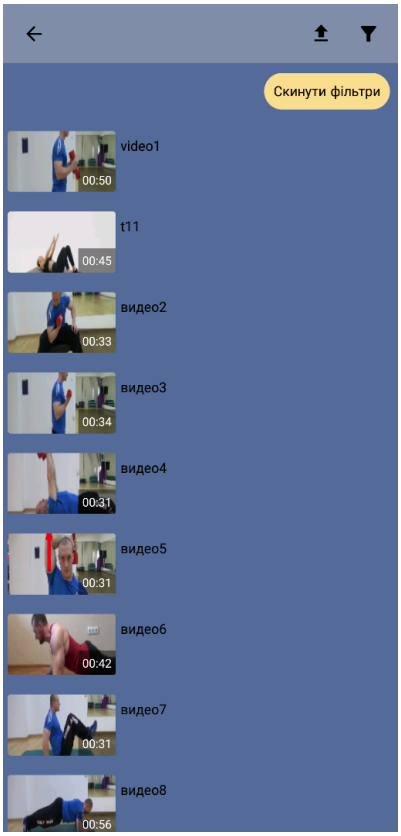

Рисунок 3.10 – Сторінка для вибору відео

На Рисунку 3.11 представлений інтерфейс завантаження відео в мобільному додатку. У верхній частині екрана розташована панель навігації зі стрілкою назад для повернення на попередній екран. Нижче панелі навігації розміщений відеопрогравач з чорним екраном, що призначений для попереднього перегляду вибраного або завантаженого відео. Під відеопрогравачем розташоване текстове поле з написом "Video name", куди користувач може ввести назву відео. Нижче текстового поля з назвою відео розміщено кілька кнопок з наступними діями: "Choose video" для вибору відео з пристрою, "Show video" для перегляду вибраного відео і "Оберіть м'язові групи" для вибору м'язових груп, на які спрямоване відео. У нижній частині екрана розміщена велика жовта кнопка з написом "upload", яка призначена для завантаження відео на сервер.

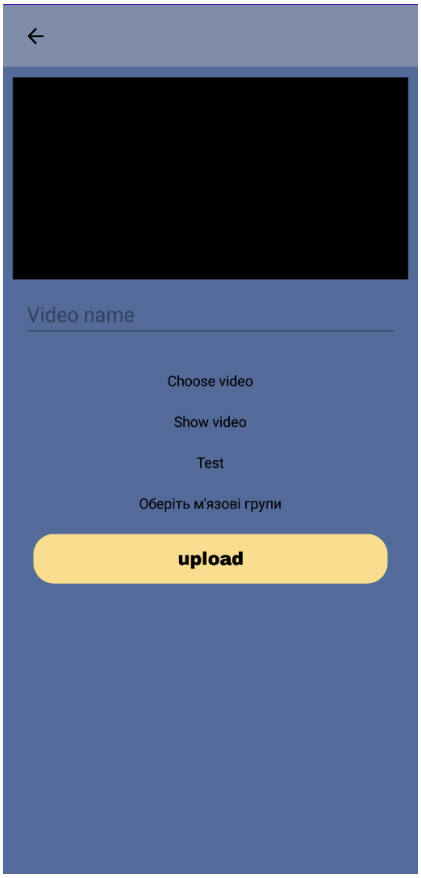

Рисунок 3.11 – Сторінка для початку завантаження відео користувачем

На Рисунку 3.12 відображено відеоплеєр мобільного додатку, призначений для перегляду відео. Екран виглядає чисто і професійно з добре структурованими елементами управління відтворенням. Великі та зрозумілі іконки управління (пауза, відтворення, перемотування) роблять взаємодію із відео зручною. Назва відео "Pretest" є помітною та інформативною, допомагаючи користувачеві зрозуміти, що це за відео.

Рисунок 3.12 – Сторінка відеоплеєру з кнопками управління

На Рисунку 3.13 відображено відеоплеєр мобільного додатку, призначений для перегляду відео. Без кнопок управління.

Рисунок 3.13 – Сторінка відеоплеєру без кнопок управління

На Рисунку 3.14 відображено відеоплеєр мобільного додатку, призначений для перегляду відео. Без кнопок управління та в горизонтальному положенні екрану.

Рисунок 3.14 – Сторінка відеоплеєру без кнопок управління в горизонтальному положенні екрану

ВИСНОВКИ

Завершення цієї дипломної роботи підкреслило значущість та потенціал застосування інформаційних технологій у сфері фітнесу, зокрема через розробку мобільних додатків на платформі Android, які сприяють організації тренувального процесу. Розробка такої системи відповідає сучасним тенденціям оцифровування життя та здоров'я людини, а також відкриває нові можливості для підвищення ефективності фізичних вправ.

Протягом виконання роботи було ідентифіковано ключові вимоги користувачів до мобільних додатків для тренувань, включаючи зручність використання, персоналізацію тренувань, інтеграцію з іншими здоров'я-орієнтованими додатками та пристроями, а також надійність зберігання та обробки даних. Було розроблено та протестовано прототип системи, який демонструє потенціал для подальшого розвитку та комерціалізації.

Основні результати дослідження включають:

1) Створення форми для реєстрації та авторизації користувачів;
2) Створення можливості для створення та управління персональними тренувальними планами;
3) Можливість відстеження активностей (наприклад, біг, плавання, велоспорт) з можливістю збору даних через введення вручну;
4) Можливість аналізу результатів тренувань;
5) Інтуїтивно зрозумілий і привабливий користувацький інтерфейс;
6) Високу продуктивність додатку з мінімальною затримкою відгуку;
7) Адаптивність інтерфейсу для різних типів пристроїв Android;
8) Забезпечення конфіденційності та безпеки даних користувачів.

Проте, дослідження має кілька обмежень, що мають бути адресовані у майбутніх роботах. Зокрема, обмежена кількість учасників у тестуванні додатку та

відсутність довгострокового аналізу використання. Також було використано лише одну платформу розробки, що може вплинути на генералізацію результатів.

Для подальшого розвитку системи рекомендується здійснити більш масштабне тестування з різними групами користувачів, розширити функціональні можливості додатку за рахунок інтеграції з різними носимими пристроями, а також провести дослідження щодо комерційної вигоди та можливостей монетизації додатку.

Завершуючи, дипломна робота демонструє важливість інтеграції технологічних інновацій у розвиток фітнес-індустрії та підтверджує, що мобільні додатки можуть істотно підвищити якість та ефективність тренувальних процесів.

СПИСОК ЛІТЕРАТУРИ

1. MyFitnessPal. Офіційний сайт. Доступно за адресою: https://www.strava.com/about.
2. Strava. "About Us" — Офіційний сайт. Доступно за адресою: https://www.strava.com/about.
3. Fitbit. Офіційний сайт. Доступно за адресою: https://www.fitbit.com/global/us/home.
4. Академія наук [Автор невідомий]. "Сучасні тенденції та перспективи розвитку фізичної підготовки та спорту Збройних Сил України, правоохоронних органів, рятувальних та інших спеціальних служб на шляху євроатлантичної інтеграції України". Доступно за адресою: https://www.academia.edu/35493215
5. Спортбук. "Особливості і перспективи розвитку системи управління сферою фізичної культури і спорту". Доступно за адресою: https://sportbuk.com/2011/03/18/osoblyvosti-i-perspektyvy-rozvytku-systemy-upravlinnya-sferoyu-fizychnoji-kultury-i-sportu/
6. Java & PHP портал. Доступно за адресою: http://javaphp.ptngu.com
7. Android Developers. Офіційна документація. Доступно за адресою: https://developer.android.com
8. Wallace Jackson. "Introduction to XML: Defining an Android App, Its Design, and Constants".
9. Інформаційна система організації процесу тренувань. Github. [Електронний ресурс] – Режим доступу: https://github.com/PikabuT0T/Trainee

ДОДАТОК А

activity menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/root_element"
    tools:context=".MainActivity"
    android:background="#546b99">

    <LinearLayout
        android:id="@+id/linearLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#808da8"
        android:gravity="center_vertical"
        android:orientation="horizontal">

        <TextView
            android:id="@+id/profile"
            android:layout_width="match_parent"
            android:layout_height="48dp"
            android:gravity="center"
            android:text="Ваш профіль"
            android:textColor="@color/black" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/linearLayout"
        android:background="#546b99"
        android:orientation="vertical">

        <LinearLayout
```

```
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="650dp"
    android:gravity="center_horizontal"
    android:padding="5dp">

    <TextView
        android:id="@+id/buttonSignIn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@drawable/round_corner"
        android:fontFamily="@font/archivo_black"
        android:textSize="20sp"
        android:padding="10dp"
        android:text="Увійти"
        android:textColor="@color/black"
        />

    <TextView
        android:id="@+id/buttonSignUp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="20dp"
        android:background="@drawable/round_corner"
        android:fontFamily="@font/archivo_black"
        android:textSize="20sp"
        android:padding="10dp"
        android:text="Зареєструватися"
        android:textColor="@color/black" />

</LinearLayout>


<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:gravity="center"
```

```
                android:text="Вам необхідно увійти або зареєструватися" />
        </LinearLayout>

</RelativeLayout>
```

activity_main.xml

```
    <?xml version="1.0" encoding="utf-8"?>
    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:id="@+id/root_menu"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#546b99"
        tools:contexts=".MainActivity">

        <LinearLayout
            android:id="@+id/linearLayout3"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="#808da8"
            android:gravity="center_vertical"
            android:orientation="horizontal">

            <androidx.cardview.widget.CardView
                android:layout_width="48dp"
                android:layout_height="48dp"
                android:layout_gravity="center"
                android:layout_margin="5dp"
                android:backgroundTint="@color/black"
                app:cardCornerRadius="25dp">

                <ImageView
                    android:id="@+id/profileImage"
                    android:layout_width="48dp"
                    android:layout_height="48dp"
                    android:background="#00FFFFFF"
                    android:scaleType="centerCrop"
                    app:srcCompat="@drawable/ic_profile"/>
```

```
    </androidx.cardview.widget.CardView>

    <TextView
        android:id="@+id/userName"
        android:layout_width="wrap_content"
        android:layout_height="48dp"
        android:fontFamily="@font/archivo_black"
        android:gravity="center_vertical"
        android:text="@string/User_name"
        android:textColor="@color/black"
        android:textSize="20sp"/>

</LinearLayout>

<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_below="@+id/linearLayout3"
    android:orientation="vertical">

    <ImageButton
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/videoButtonImage"
        android:layout_alignParentEnd="true"
        android:layout_centerHorizontal="true"
        android:layout_marginEnd="50dp"
        android:background="#00FFFFFF"
        android:src="@drawable/ic_qr_code_button_100" />

    <ImageButton
        android:id="@+id/videoButtonImage"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="100dp"
        android:background="#00FFFFFF"
```

```
                app:srcCompat="@drawable/ic_videobutton"/>

            <ImageButton
                android:id="@+id/button2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_above="@+id/videoButtonImage"
                android:layout_alignParentStart="true"
                android:layout_marginStart="50dp"
                android:layout_marginBottom="0dp"
                android:background="#00FFFFFF"
                app:srcCompat="@drawable/_______1" />
        </RelativeLayout>

</RelativeLayout>
```

activity_profile.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/root_profile"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ProfileActivity"
    android:orientation="vertical"
    >

    <RelativeLayout
        android:id="@+id/linearLayout2"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:background="#808da8">

        <ImageButton
            android:id="@+id/buttonToMainActivity"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="20dp"
```

```
            android:layout_centerVertical="true"
            android:layout_alignParentStart="true"
            android:background="@android:color/transparent"
            android:src="@drawable/ic_back_menu" />



        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolbar_profile"
            android:layout_width="48dp"
            android:layout_height="48dp"
            android:layout_margin="20dp"
            android:layout_gravity="center_vertical"
            android:layout_alignParentEnd="true"
            android:background="#808da8"
            android:gravity="center_vertical">
        </androidx.appcompat.widget.Toolbar>


    </RelativeLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/linearLayout2"
        android:layout_marginTop="0dp"
        android:gravity="center_horizontal"
        android:orientation="vertical"
        android:background="#546b99">

        <TextView
            android:id="@+id/textView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="150dp"
            android:textSize="20sp"
            android:layout_gravity="center"
            android:text="Ваш профіль"
```

```
    android:textColor="@color/black"/>

<androidx.cardview.widget.CardView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginTop="50dp"
    app:cardCornerRadius="70dp"
    android:backgroundTint="@color/black">

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="140dp"
        android:layout_height="140dp"
        android:scaleType="centerCrop"
        app:srcCompat="@drawable/ic_profile"
        android:backgroundTint="@color/black"/>
</androidx.cardview.widget.CardView>

<TextView
    android:id="@+id/viewName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginTop="50dp"
    android:textSize="20sp"
    android:textColor="@color/black"
    android:text="name" />

<TextView
    android:id="@+id/emailView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginTop="30dp"
    android:textSize="20sp"
    android:textColor="@color/black"
    android:text="email" />
```

```
        <TextView
            android:id="@+id/phoneView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginTop="30dp"
            android:textSize="20sp"
            android:textColor="@color/black"
            android:text="phone" />


    </LinearLayout>

    <RelativeLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_margin="10dp">

        <ImageButton
            android:id="@+id/button_logout"
            android:layout_width="48dp"
            android:layout_height="48dp"
            android:layout_gravity="center_vertical"
            android:background="@android:color/transparent"
            android:src="@drawable/ic_exit" />
    </RelativeLayout>

</RelativeLayout>
```

activity_scanner.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
tools:context=".ScannerActivity"
android:background="@color/BlueNova">

<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="60dp"
    android:background="#808da8">

    <ImageButton
        android:id="@+id/buttonToMainActivity"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:layout_centerVertical="true"
        android:layout_alignParentStart="true"
        android:background="@android:color/transparent"
        android:src="@drawable/ic_back_menu"
        android:contentDescription="@string/back_to_main"/>

</RelativeLayout>

<androidx.appcompat.widget.LinearLayoutCompat
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_gravity="center"
    android:gravity="center"
    android:background="@color/BlueNova">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:src="@drawable/ic_qr_code"
        android:contentDescription="@string/qr_code_image"/>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
            android:textSize="26sp"
            android:textStyle="bold"
            android:layout_marginTop="16dp"
            android:text="@string/scan_result"
            android:autoLink="all"/>

    </androidx.appcompat.widget.LinearLayoutCompat>

    <com.google.android.material.bottomappbar.BottomAppBar
        android:id="@+id/bottom_app_bar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom"
        app:fabCradleMargin="10dp"
        android:backgroundTint="@color/honeybee"
        app:fabCradleRoundedCornerRadius="10dp"
        app:fabCradleVerticalOffset="10dp"/>

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/buttonScan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:contentDescription="@string/scan_button"
        android:src="@drawable/ic_qr_code"
        app:backgroundTint="@color/honeybee"
        app:layout_anchor="@id/bottom_app_bar"
        app:tint="@color/black" />

</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

activity_calendar.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <RelativeLayout
        android:id="@+id/relativeLayout"
        android:layout_width="match_parent"
        android:layout_height="60dp"
```

```
    android:background="@color/BlueNovaLight">

    <ImageButton
        android:id="@+id/buttonToMainActivity"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_centerVertical="true"
        android:layout_margin="20dp"
        android:background="@android:color/transparent"
        android:src="@drawable/ic_back_menu" />
</RelativeLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_below="@+id/relativeLayout"
    android:layout_marginTop="0dp"
    android:orientation="vertical">
    <!-- Календар -->
    <com.applandeo.materialcalendarview.CalendarView
        android:id="@+id/calendarView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/relativeLayout"
        android:layout_marginTop="0dp" />

    <!-- RecyclerView для відображення списку заміток -->
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/btnAddNote"
        android:layout_marginTop="15dp" />

    <!-- EditText для введення нової замітки -->
    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
```

```
            android:layout_height="wrap_content"
            android:layout_below="@+id/calendarView"
            android:layout_marginTop="15dp"
            android:hint="Введіть замітку"
            android:shadowColor="@color/black"
            android:inputType="textMultiLine" />

        <!-- Кнопка для додавання нової замітки -->
        <TextView
            android:id="@+id/btnAddNote"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="@drawable/round_corner"
            android:fontFamily="@font/archivo_black"
            android:layout_gravity="center"
            android:textSize="20sp"
            android:padding="10dp"
            android:layout_margin="16dp"
            android:text="Додати замітку"
            android:textColor="@color/black"
            />
    </LinearLayout>

</RelativeLayout>
```

activity playlist.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#546b99"
    android:orientation="vertical"
    tools:context=".VideoPlayerActivity">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="60dp"
```

```
    android:background="#808da8">

    <ImageButton
        android:id="@+id/buttonToMainActivity"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:layout_centerVertical="true"
        android:layout_alignParentStart="true"
        android:background="@android:color/transparent"
        android:src="@drawable/ic_back_menu" />

    <ImageButton
        android:id="@+id/buttonToUploadVideo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_margin="5dp"
        android:layout_toStartOf="@+id/buttonToFilter"
        android:background="@android:color/transparent"
        android:src="@drawable/ic_upload_video" />

    <ImageButton
        android:id="@+id/buttonToFilter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_centerVertical="true"
        android:layout_margin="20dp"
        android:layout_marginEnd="343dp"
        android:background="@android:color/transparent"
        android:src="@drawable/ic_filter"/>

</RelativeLayout>
<TextView
    android:id="@+id/button_main"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="end"
```

```
        android:layout_margin="10dp"
        android:padding="10dp"
        android:background="@drawable/round_corner"
        android:fontFamily="@font/archivo_black"
        android:gravity="center"
        android:text="Скинути фільтри"
        android:textColor="@color/black"
        android:textSize="14sp"/>

    <androidx.recyclerview.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/video_rv"/>


</LinearLayout>
```

activity_video_player.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".PlaylistActivity">

    <com.google.android.exoplayer2.ui.PlayerView
        android:id="@+id/exoplayer_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"

        app:controller_layout_id="@layout/custom_playback_view"
        app:hide_on_touch="true"
        app:player_layout_id="@layout/exo_player_view"
        app:resize_mode="fit"
        app:shutter_background_color="#000000"
        android:background="@color/black"
        app:show_timeout="4000"
```

```
        app:surface_type="surface_view"
        app:use_controller="true" />

</LinearLayout>
```

custom_playback_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    >

    <LinearLayout
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="50dp"
        android:orientation="horizontal"
        android:gravity="center_vertical"
        android:visibility="visible"
        android:layout_marginTop="10dp"
        >
        <ImageView
            android:id="@+id/buttonToMainActivity"
            android:layout_width="50dp"
            android:layout_height="50dp"
            android:padding="10dp"
            android:src="@drawable/ic_baseline_arrow_back"/>
        <TextView
            android:id="@+id/video_title"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:ellipsize="end"
            android:maxLines="2"
            android:text="This is title of video"
            android:textColor="#FFFFFF"
            android:textSize="18sp"/>
    </LinearLayout>
```

```
<RelativeLayout
    android:id="@+id/progress"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_above="@+id/bottom_icons"
    android:visibility="visible">

    <TextView
        android:id="@+id/exo_position"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_centerVertical="true"
        android:layout_marginLeft="10dp"
        android:text="5555"
        android:textColor="#FFFFFF"/>

    <TextView
        android:id="@+id/exo_duration"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:layout_marginRight="10dp"
        android:text="4444"
        android:textColor="#FFFFFF"/>

    <com.google.android.exoplayer2.ui.DefaultTimeBar
        android:id="@+id/exo_progress"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_toStartOf="@+id/exo_duration"
        android:layout_toEndOf="@+id/exo_position"
        app:played_color="#808da8"/>

</RelativeLayout>
```

```
<LinearLayout
    android:id="@+id/bottom_icons"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_marginBottom="5dp"
    android:gravity="center_vertical"
    android:orientation="horizontal"
    android:weightSum="5">

    <ImageView
        android:id="@+id/exo_rew"
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_weight="1"
        android:src="@drawable/ic_baseline_replay_10"/>
    <ImageView
        android:id="@+id/exo_prev"
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_weight="1"
        android:src="@drawable/ic_baseline_skip_previous"/>
    <ImageView
        android:id="@+id/exo_play"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_weight="1"
        android:src="@drawable/ic_baseline_play_arrow"/>
    <ImageView
        android:id="@+id/exo_pause"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_weight="1"
        android:src="@drawable/ic_baseline_pause"/>
    <ImageView
        android:id="@+id/exo_next"
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_weight="1"
```

```
                android:src="@drawable/ic_baseline_skip_next"/>
            <ImageView
                android:id="@+id/exo_ffwd"
                android:layout_width="30dp"
                android:layout_height="30dp"
                android:layout_weight="1"
                android:src="@drawable/ic_baseline_forward_10"/>

        </LinearLayout>
</RelativeLayout>
```

ДОДАТОК Б

MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    TextView buttonSignUp, buttonSignIn;
    private FirebaseAuth mAuth;
    private FirebaseDatabase db;
    private DatabaseReference users;
    private String authToken;
    RelativeLayout root;
    LayoutInflater inflater; // Перевикористання inflater

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        checkToken();
        setContentView(R.layout.activity_menu);

        buttonSignIn = findViewById(R.id.buttonSignIn);
        buttonSignUp = findViewById(R.id.buttonSignUp);

        root = findViewById(R.id.root_element);
        inflater = LayoutInflater.from(this); // Ініціалізація тут

        mAuth = FirebaseAuth.getInstance();
        db = FirebaseDatabase.getInstance();
        users = db.getReference("Users");

        buttonSignUp.setOnClickListener(view -> showSignUpWindow());
        buttonSignIn.setOnClickListener(view -> showSignInWindow());
    }
    private void showSignInWindow() {
        AlertDialog.Builder dialog = new AlertDialog.Builder(this);
        dialog.setTitle("Увійти").setMessage("Введіть дані для входу");
        View sign_in_window = inflater.inflate(R.layout.sign_in_form, null);
        dialog.setView(sign_in_window);
```

```
        MaterialEditText email =
sign_in_window.findViewById(R.id.emailField);
        MaterialEditText pass = sign_in_window.findViewById(R.id.passField);

        dialog.setNegativeButton("Скасувати", (dialogInterface, i) ->
dialogInterface.dismiss());
        dialog.setPositiveButton("Увійти", (dialogInterface, i) -> {
            if (TextUtils.isEmpty(email.getText().toString())) {
                Snackbar.make(root, "Введіть коректну пошту",
Snackbar.LENGTH_LONG).show();
                return;
            }
            if (pass.getText().toString().length() < 6) {
                Snackbar.make(root, "Введіть пароль, який має більше 5
символів", Snackbar.LENGTH_LONG).show();
                return;
            }
            // Вхід за поштою та паролем
            mAuth.signInWithEmailAndPassword(email.getText().toString(),
pass.getText().toString())
                    .addOnSuccessListener(authResult -> {
                        FirebaseUser user = mAuth.getCurrentUser();
                        if(user != null){
                            user.getIdToken(true).addOnCompleteListener(task
-> {
                                if(task.isSuccessful()){
                                    authToken = task.getResult().getToken();
                                    saveToken(authToken);
                                    Log.d("Auth Token", "Token: " +
authToken);
                                }
                                else{
                                    Log.d("Auth Token Failure", "Token was
not found");
                                }
                            });
                        }
```

```
                        startActivity(new Intent(MainActivity.this,
MenuActivity.class));
                        finish();
                    })
                    .addOnFailureListener(e -> Snackbar.make(root, "Помилка
авторізації!" + e.getMessage(), Snackbar.LENGTH_LONG).show());
        });
        dialog.show();
    }

    private void showSignUpWindow() {
        AlertDialog.Builder dialog = new AlertDialog.Builder(this);
        dialog.setTitle("Зареєструватися").setMessage("Введіть необхідні дані
для реєстрації в поля нижче");
        View register_window = inflater.inflate(R.layout.register_form,
null);
        dialog.setView(register_window);

        MaterialEditText email =
register_window.findViewById(R.id.emailField);
        MaterialEditText pass = register_window.findViewById(R.id.passField);
        MaterialEditText name = register_window.findViewById(R.id.nameField);
        MaterialEditText phone =
register_window.findViewById(R.id.phoneField);

        dialog.setNegativeButton("Скасувати", (dialogInterface, i) ->
dialogInterface.dismiss());
        dialog.setPositiveButton("Додати користувача", (dialogInterface, i) -
> {
            if (TextUtils.isEmpty(email.getText().toString())) {
                Snackbar.make(root, "Введіть коректну пошту",
Snackbar.LENGTH_LONG).show();
                return;
            }
            if (pass.getText().toString().length() < 6) {
                Snackbar.make(root, "Введіть пароль, який має більше 5
символів", Snackbar.LENGTH_LONG).show();
                return;
            }
```

```
            if (TextUtils.isEmpty(name.getText().toString())) {
                Snackbar.make(root, "Введіть коректне ім'я",
Snackbar.LENGTH_LONG).show();
                return;
            }
            if (TextUtils.isEmpty(phone.getText().toString())) {
                Snackbar.make(root, "Введіть коректний номер телефону",
Snackbar.LENGTH_LONG).show();
                return;
            }
            // Сама реєстрація користувача в БД
            mAuth.createUserWithEmailAndPassword(email.getText().toString(),
pass.getText().toString())
                    .addOnSuccessListener(authResult -> {
                        Map<String, Object> userMap = new HashMap<>();
                        userMap.put("email", email.getText().toString());
                        userMap.put("pass", pass.getText().toString());
                        userMap.put("name", name.getText().toString());
                        userMap.put("phone", phone.getText().toString());


users.child(FirebaseAuth.getInstance().getCurrentUser().getUid())
                                .setValue(userMap)
                                .addOnSuccessListener(unused ->
Snackbar.make(root, "Новий користувач успішно створений!",
Snackbar.LENGTH_LONG).show())
                                .addOnFailureListener(e ->
Snackbar.make(root, "Виникли складнощі при створенні користувача!",
Snackbar.LENGTH_LONG).show());
                    });
        });
        dialog.show();
    }

    private void saveToken(String token) {
        SharedPreferences sharedPreferences =
getApplication().getSharedPreferences("AppPrefs",
getApplication().MODE_PRIVATE);
        SharedPreferences.Editor editor = sharedPreferences.edit();
```

```
            editor.putString("AuthToken", token);  // Шифрование токена перед
        сохранением рекомендуется
            editor.apply();
        }

        private String getToken() {
            SharedPreferences sharedPreferences =
        getApplication().getSharedPreferences("AppPrefs",
        getApplication().MODE_PRIVATE);
            return sharedPreferences.getString("AuthToken", null);  //
        Дешифрование токена после получения
        }

        private void checkToken(){
            authToken = getToken();
            if(authToken != null){
                startActivity(new Intent(MainActivity.this, MenuActivity.class));
            } else {
                Log.d("Auth Token Failure", "Token was not found");
            }
        }

}
```

ViewHolder.java

```
    public class ViewHolder extends RecyclerView.ViewHolder {
        TextView videoNameView, videoDurationView;
        ImageView thumbnail;

        public ViewHolder(@NonNull View itemView) {
            super(itemView);
            thumbnail = itemView.findViewById(R.id.thumbnail);
            videoNameView = itemView.findViewById(R.id.video_name);
            videoDurationView = itemView.findViewById(R.id.video_duration);
        }

        public void bindData(Member member) {
            videoNameView.setText(member.getVideoName());
            videoDurationView.setText(member.getVideoDuration());
```

```
                try {
                    Picasso.get()
                            .load(Uri.parse(member.getVideoPreviewImage()))
                            .placeholder(R.drawable.ic_profile)
                            .resize(140, 140)
                            .into(thumbnail);
                } catch (Exception e) {
                    //thumbnail.setImageResource(R.drawable.ic_error); // Assuming
        ic_error is a drawable resource
                }
            }

}
```

MenuActivity.java

```
        public class MenuActivity extends AppCompatActivity {
            private TextView userName;
            private ImageView btnUserProfile;
            private RelativeLayout rootMenu;
            private ProfileViewModel profileViewModel;
            private User currentUser;

            @Override
            protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);

                profileViewModel = new
        ViewModelProvider(this).get(ProfileViewModel.class);
                initViews();
                observeViewModel();
                setupButtons();
            }

            private void initViews() {
                rootMenu = findViewById(R.id.root_menu);
                btnUserProfile = findViewById(R.id.profileImage);
                userName = findViewById(R.id.userName);
            }
```

```
    private void observeViewModel() {
        profileViewModel.getUser().observe(this, user -> {
            updateUI(user);
            currentUser = user;
        });
    }

    private void updateUI(User user){
        if (user != null) {
            userName.setText(user.getName());
            if (user.getProfileUri() != null &&
!user.getProfileUri().isEmpty()) {
                Picasso.get()
                        .load(user.getProfileUri())
                        .placeholder(R.drawable.ic_profile)
                        .resize(48, 48)
                        .into(btnUserProfile);
            }
        }
    }

    private void setupButtons() {
        btnUserProfile.setOnClickListener(v ->
goToActivity(ProfileActivity.class));
        findViewById(R.id.videoButtonImage).setOnClickListener(v ->
goToActivity(PlaylistActivity.class));
        findViewById(R.id.button3).setOnClickListener(v ->
goToActivity(ScannerActivity.class));
        findViewById(R.id.button2).setOnClickListener(view ->
goToActivity(CalendarActivity.class));
    }

    private void goToActivity(Class<?> activityClass) {
        Intent intent = new Intent(MenuActivity.this, activityClass);
        intent.addFlags(Intent.FLAG_ACTIVITY_REORDER_TO_FRONT);
        currentUser = profileViewModel.getUser().getValue();
        intent.putExtra("userDetails", currentUser);
        Log.d("MA uid from currentUser", currentUser.getUid());
        startActivity(intent);
```

```
        }
}
```

ProfileActivity.java

```
    public class ProfileActivity extends AppCompatActivity {
        private static final int PICK_IMAGE = 1;

        private TextView nameView, emailView, phoneView;
        private ImageView imageView2;
        private ImageButton buttonMain, buttonLogout;
        private ProfileViewModel profileViewModel;
        private User currentUser;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_profile);

            profileViewModel = new
    ViewModelProvider(this).get(ProfileViewModel.class);
            initViews();
            observeViewModel();
            setupListeners();

            currentUser = getIntent().getParcelableExtra("userDetails");
            if (currentUser != null) {
                profileViewModel.setUser(currentUser); // Оновлюємо userLiveData
            }
        }

        private void initViews() {
            nameView = findViewById(R.id.viewName);
            emailView = findViewById(R.id.emailView);
            phoneView = findViewById(R.id.phoneView);
            imageView2 = findViewById(R.id.imageView2);
            buttonMain = findViewById(R.id.buttonToMainActivity);
            buttonLogout = findViewById(R.id.button_logout);
            Toolbar toolbar = findViewById(R.id.toolbar_profile);
            setSupportActionBar(toolbar);
        }
```

```
    private void observeViewModel() {
        profileViewModel.getUser().observe(this, user -> {
            updateUI(user);
        });
    }

    public void updateUI(User user) {
        if (user != null) {
            nameView.setText(user.getName());
            emailView.setText(user.getEmail());
            phoneView.setText(user.getPhone());
            if (user.getProfileUri() != null &&
!user.getProfileUri().isEmpty()) {
                Picasso.get()
                        .load(user.getProfileUri())
                        .placeholder(R.drawable.ic_profile)
                        .resize(150, 150)
                        .into(imageView2);
            }
        } else {
            nameView.setText("");
            emailView.setText("");
            phoneView.setText("");
            imageView2.setImageResource(R.drawable.ic_profile);
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_items, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        int itemId = item.getItemId();
        if (itemId == R.id.settings_name_change) {
            showRefactorWindow("name");
```

```
            return true;
        } else if (itemId == R.id.settings_password_change) {
            showRefactorWindow("password");
            return true;
        } else if (itemId == R.id.settings_phone_change) {
            showRefactorWindow("phone");
            return true;
        } else {
            return super.onOptionsItemSelected(item);
        }
    }

    private void showRefactorWindow(String type) {
        AlertDialog.Builder dialog = new AlertDialog.Builder(this);
        dialog.setTitle("Update your " + type);
        dialog.setMessage("Please enter the new " + type + " below:");

        LayoutInflater inflater = LayoutInflater.from(this);
        View refactorWindow = null;

        switch (type) {
            case "name":
                refactorWindow =
inflater.inflate(R.layout.refactor_name_form, null);
                break;
            case "password":
                refactorWindow =
inflater.inflate(R.layout.refactor_password_form, null);
                break;
            case "phone":
                refactorWindow =
inflater.inflate(R.layout.refactor_phone_form, null);
                break;
        }

        if (refactorWindow != null) {
            dialog.setView(refactorWindow);
            TextInputEditText inputField =
refactorWindow.findViewById(R.id.newValue);
```

```
            TextInputEditText currentPasswordField =
refactorWindow.findViewById(R.id.currentPass);
            TextInputEditText secondInputField = type.equals("password") ?
refactorWindow.findViewById(R.id.confirmNewValue) : null;

            dialog.setPositiveButton("Save", (dialogInterface, i) -> {
                String newValue = inputField.getText().toString().trim();
                String currentPassword = currentPasswordField != null ?
currentPasswordField.getText().toString().trim() : null;
                String confirmNewValue = secondInputField != null ?
secondInputField.getText().toString().trim() : null;

                if (type.equals("password") &&
!newValue.equals(confirmNewValue)) {
                    Toast.makeText(getApplicationContext(), "Passwords do not
match.", Toast.LENGTH_SHORT).show();
                    return;
                }
                profileViewModel.updateUserInformation(type, newValue,
currentPassword);
            });
            dialog.setNegativeButton("Cancel", (dialogInterface, i) ->
dialogInterface.dismiss());
            dialog.show();
        } else {
            Toast.makeText(this, "Error: Unable to load the form.",
Toast.LENGTH_SHORT).show();
        }
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == PICK_IMAGE && resultCode == RESULT_OK && data !=
null && data.getData() != null) {
            profileViewModel.uploadImage(data.getData());
            Toast.makeText(this, "Image uploaded successfully!",
Toast.LENGTH_LONG).show();
```

```
        } else {
            Toast.makeText(this, "Error: Try uploading another image!",
Toast.LENGTH_LONG).show();
        }
    }

    private void setupListeners() {
        buttonMain.setOnClickListener(view ->
goToActivity(MenuActivity.class));
        imageView2.setOnClickListener(view -> selectImage());
        buttonLogout.setOnClickListener(view -> {
            FirebaseAuth.getInstance().signOut();
            profileViewModel.detachListener(); // Відключаємо слухачів бази
даних
            clearAllData();
            goToActivity(MainActivity.class);
            finish();
        });
    }

    private void clearAllData() {
        SharedPreferences sharedPreferences =
getApplication().getSharedPreferences("AppPrefs", MODE_PRIVATE);
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.clear();
        editor.apply();
        clearCache();
    }

    private void clearCache() {
        try {
            File cacheDir = getCacheDir();
            if (cacheDir != null && cacheDir.isDirectory()) {
                deleteDir(cacheDir);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
    private boolean deleteDir(File dir) {
        if (dir != null && dir.isDirectory()) {
            String[] children = dir.list();
            for (String child : children) {
                boolean success = deleteDir(new File(dir, child));
                if (!success) {
                    return false;
                }
            }
            return dir.delete();
        } else if (dir != null && dir.isFile()) {
            return dir.delete();
        } else {
            return false;
        }
    }

    private void selectImage() {
        Intent intent = new Intent(Intent.ACTION_GET_CONTENT);
        intent.setType("image/*");
        startActivityForResult(intent, PICK_IMAGE);
    }

    private void goToActivity(Class<?> activityClass) {
        Intent intent = new Intent(ProfileActivity.this, activityClass);
        intent.addFlags(Intent.FLAG_ACTIVITY_REORDER_TO_FRONT);
        intent.putExtra("userDetails",
profileViewModel.getUser().getValue());
        startActivity(intent);
    }

    @Override
    protected void onNewIntent(Intent intent) {
        super.onNewIntent(intent);
        setIntent(intent);
        currentUser = getIntent().getParcelableExtra("userDetails");
        if (currentUser != null) {
            profileViewModel.setUser(currentUser); // Оновлюємо userLiveData
```

```
            }
        }

        @Override
        protected void onDestroy() {
            super.onDestroy();
            profileViewModel.detachListener(); // Відключаємо слухачів бази даних
    при знищенні активності
        }
}
```

PlaylistActivity.java

```
    public class PlaylistActivity extends AppCompatActivity {
        private VideoViewModel viewModel;
        private RecyclerView recyclerView;
        private ImageButton buttonMain, buttonToUploadVideo;

        private CheckBox pectoralis_major, pectoralis_minor, latissimus_dorsi,
    trapezius, rhomboids,
                anterior_deltoid, lateral_deltoid, posterior_deltoid,
    biceps_brachii, triceps_brachii,
                quadriceps, hamstrings, rectus_abdominis, external_obliques,
    internal_obliques, transverse_abdominis;

        private Button buttonSearch, buttonMainVideo;
        LayoutInflater inflater;


        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_playlist);
            inflater = LayoutInflater.from(this);

            initializeUI();

            viewModel = new ViewModelProvider(this).get(VideoViewModel.class);
            findViewById(R.id.buttonToFilter).setOnClickListener(view ->
    showSortWindow());
            setupRecyclerView();
```

```
    }

    private void showSortWindow() {
        AlertDialog.Builder dialog = new AlertDialog.Builder(this);
        LayoutInflater inflater = LayoutInflater.from(this);
        View sign_in_window =
inflater.inflate(R.layout.find_muscles_grupe_form, null);

        pectoralis_major =
sign_in_window.findViewById(R.id.checkBox_pectoralis_major);
        pectoralis_minor =
sign_in_window.findViewById(R.id.checkBox_pectoralis_minor);
        latissimus_dorsi =
sign_in_window.findViewById(R.id.checkBox_latissimus_dorsi);
        trapezius = sign_in_window.findViewById(R.id.checkBox_trapezius);
        rhomboids = sign_in_window.findViewById(R.id.checkBox_rhomboids);
        anterior_deltoid =
sign_in_window.findViewById(R.id.checkBox_anterior_deltoid);
        lateral_deltoid =
sign_in_window.findViewById(R.id.checkBox_lateral_deltoid);
        posterior_deltoid =
sign_in_window.findViewById(R.id.checkBox_posterior_deltoid);
        biceps_brachii =
sign_in_window.findViewById(R.id.checkBox_biceps_brachii);
        triceps_brachii =
sign_in_window.findViewById(R.id.checkBox_triceps_brachii);
        quadriceps = sign_in_window.findViewById(R.id.checkBox_quadriceps);
        hamstrings = sign_in_window.findViewById(R.id.checkBox_hamstrings);
        rectus_abdominis =
sign_in_window.findViewById(R.id.checkBox_rectus_abdominis);
        external_obliques =
sign_in_window.findViewById(R.id.checkBox_external_obliques);
        internal_obliques =
sign_in_window.findViewById(R.id.checkBox_internal_obliques);
        transverse_abdominis =
sign_in_window.findViewById(R.id.checkBox_transverse_abdominis);

        dialog.setView(sign_in_window);
```

```
        dialog.setPositiveButton("Застосувати", (dialogInterface, i) ->
searchExercises());
        dialog.setNegativeButton("Назад", (dialogInterface, i) ->
dialogInterface.dismiss());

        dialog.show();
    }


    private void initializeUI() {
        buttonMain = findViewById(R.id.buttonToMainActivity);
        buttonToUploadVideo = findViewById(R.id.buttonToUploadVideo);

        buttonSearch = findViewById(R.id.button_search);
        buttonMainVideo = findViewById(R.id.button_main);

        buttonMain.setOnClickListener(view -> startActivity(new
Intent(PlaylistActivity.this, MenuActivity.class)));
        buttonToUploadVideo.setOnClickListener(view -> startActivity(new
Intent(PlaylistActivity.this, UploadVideoActivity.class)));
        buttonSearch.setOnClickListener(view -> {searchExercises();});
        buttonMainVideo.setOnClickListener(view -> {setupRecyclerView();});

        recyclerView = findViewById(R.id.video_rv);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
    }

    private void searchExercises() {
        List<String> tags = new ArrayList<>();
        if (pectoralis_major.isChecked()) tags.add("pectoralis_major");
        if (pectoralis_minor.isChecked()) tags.add("pectoralis_minor");
        if (latissimus_dorsi.isChecked()) tags.add("latissimus_dorsi");
        if (trapezius.isChecked()) tags.add("trapezius");
        if (rhomboids.isChecked()) tags.add("rhomboids");
        if (anterior_deltoid.isChecked()) tags.add("anterior_deltoid");
        if (lateral_deltoid.isChecked()) tags.add("lateral_deltoid");
        if (posterior_deltoid.isChecked()) tags.add("posterior_deltoid");
        if (biceps_brachii.isChecked()) tags.add("biceps_brachii");
```

```
        if (triceps_brachii.isChecked()) tags.add("triceps_brachii");
        if (quadriceps.isChecked()) tags.add("quadriceps");
        if (hamstrings.isChecked()) tags.add("hamstrings");
        if (rectus_abdominis.isChecked()) tags.add("rectus_abdominis");
        if (external_obliques.isChecked()) tags.add("external_obliques");
        if (internal_obliques.isChecked()) tags.add("internal_obliques");
        if (transverse_abdominis.isChecked())
tags.add("transverse_abdominis");

        String tagsString = String.join(",", tags);
        Log.d("Search Exercises", tagsString);

        // Здесь вы обновляете RecyclerView в соответствии с отфильтрованными
тегами
        updateRecyclerView(viewModel.getFilteredVideoQuery(tagsString));
    }


    private void updateRecyclerView(Query query) {
        FirebaseRecyclerOptions<Member> options = new
FirebaseRecyclerOptions.Builder<Member>()
                .setQuery(query, Member.class)
                .build();

        FirebaseRecyclerAdapter<Member, ViewHolder> firebaseRecyclerAdapter =
new FirebaseRecyclerAdapter<Member, ViewHolder>(options) {
            @Override
            protected void onBindViewHolder(@NonNull ViewHolder viewHolder,
int position, @NonNull Member member) {
                Log.d("FirebaseRecycler", "Binding data for member: " +
member.getVideoName());
                viewHolder.bindData(member);
                viewHolder.itemView.setOnClickListener(view ->
viewModel.launchVideoPlayer(PlaylistActivity.this, member));
            }

            @NonNull
            @Override
```

```
            public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent,
int viewType) {
                View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.video_item, parent,
false);
                return new ViewHolder(view);
            }
        };

        firebaseRecyclerAdapter.startListening();
        recyclerView.setAdapter(firebaseRecyclerAdapter);
    }


    private void setupRecyclerView() {
        FirebaseRecyclerOptions<Member> options = new
FirebaseRecyclerOptions.Builder<Member>()
                .setQuery(viewModel.getVideoQuery(), Member.class)
                .build();

        FirebaseRecyclerAdapter<Member, ViewHolder> firebaseRecyclerAdapter =
new FirebaseRecyclerAdapter<Member, ViewHolder>(options) {
            @Override
            protected void onBindViewHolder(@NonNull ViewHolder viewHolder,
int position, @NonNull Member member) {
                viewHolder.bindData(member);
                viewHolder.itemView.setOnClickListener(view ->
viewModel.launchVideoPlayer(PlaylistActivity.this, member));
            }

            @NonNull
            @Override
            public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent,
int viewType) {
                View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.video_item, parent,
false);
                return new ViewHolder(view);
            }
```

```
            };

            firebaseRecyclerAdapter.startListening();
            recyclerView.setAdapter(firebaseRecyclerAdapter);
        }
}
```

VideoPlayerActivity.java

```
    public class VideoPlayerActivity extends AppCompatActivity {
        PlayerView playerView;
        SimpleExoPlayer exoPlayer;
        String videoUri;

        ImageView buttonMain;
        String videoTitle;
        TextView title;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_video_player);
            initViews();
            fetchDataFromIntent();
            initializePlayer();
        }

        private void initViews() {
            playerView = findViewById(R.id.exoplayer_view);
            title = findViewById(R.id.video_title);
            buttonMain = findViewById(R.id.buttonToMainActivity);
            buttonMain.setOnClickListener(view -> finish());
        }

        private void fetchDataFromIntent() {
            videoTitle = getIntent().getStringExtra("video_title");
            videoUri = getIntent().getStringExtra("video_uri");
            title.setText(videoTitle);
        }

        private void initializePlayer() {
```

```
        exoPlayer = new SimpleExoPlayer.Builder(this).build();
        Uri video = Uri.parse(videoUri);
        MediaItem mediaItem = MediaItem.fromUri(video);
        MediaSource mediaSource = new ProgressiveMediaSource.Factory(new
DefaultHttpDataSource.Factory()).createMediaSource(mediaItem);

        playerView.setPlayer(exoPlayer);
        playerView.setKeepScreenOn(true);
        exoPlayer.setMediaSource(mediaSource);
        exoPlayer.prepare();
        exoPlayer.setPlayWhenReady(true);

        exoPlayer.addListener(new Player.Listener() {
            @Override
            public void onPlayerError(PlaybackException error) {
                Toast.makeText(VideoPlayerActivity.this, "Video Playing
Error: " + error.getMessage(), Toast.LENGTH_SHORT).show();
            }
        });
    }

    @Override
    protected void onPause() {
        super.onPause();
        if (exoPlayer != null) {
            exoPlayer.setPlayWhenReady(false);
            exoPlayer.getPlaybackState();
        }
    }

    @Override
    protected void onResume() {
        super.onResume();
        if (exoPlayer != null) {
            exoPlayer.setPlayWhenReady(true);
            exoPlayer.getPlaybackState();
        }
    }
```

```
    @Override
    protected void onDestroy() {
        super.onDestroy();
        if (exoPlayer != null) {
            exoPlayer.release();
            exoPlayer = null;
        }
    }
}
```

CalendarActivity.java

```
public class CalendarActivity extends AppCompatActivity implements
NoteAdapter.OnNoteEditListener, NoteAdapter.OnNoteDeleteListener {

    private CalendarView calendarView;
    private RecyclerView recyclerView;
    private Map<String, List<Note>> notesMap;
    private NoteAdapter noteAdapter;
    private EditText editText;
    private TextView addNoteButton;
    private Calendar selectedDate;
    private List<EventDay> events;
    private NotesDatabaseHelper dbHelper;
    private User currentUser;
    private BackupHelper backupHelper;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.calendar_layout);

        calendarView = findViewById(R.id.calendarView);
        recyclerView = findViewById(R.id.recyclerView);
        editText = findViewById(R.id.editText);
        addNoteButton = findViewById(R.id.btnAddNote);
        notesMap = new HashMap<>();
        noteAdapter = new NoteAdapter(new ArrayList<>(), this, this);
        events = new ArrayList<>();

        recyclerView.setLayoutManager(new LinearLayoutManager(this));
```

```
        recyclerView.setAdapter(noteAdapter);
        setupListeners();
        dbHelper = new NotesDatabaseHelper(this);
        //backupHelper = new BackupHelper(this);

        calendarView.setOnDayClickListener(new OnDayClickListener() {
            @Override
            public void onDayClick(EventDay eventDay) {
                selectedDate = eventDay.getCalendar();
                updateSelectedDate();
                updateNotesForSelectedDate();
            }
        });

        addNoteButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (selectedDate != null &&
!editText.getText().toString().isEmpty()) {
                    Note newNote = new Note(
                            0, // id will be set by the database
                            selectedDate.get(Calendar.YEAR),
                            selectedDate.get(Calendar.MONTH),
                            selectedDate.get(Calendar.DAY_OF_MONTH),
                            editText.getText().toString()
                    );
                    dbHelper.addNote(newNote);

                    String dateKey = getDateKey(selectedDate);
                    List<Note> notesForDate = notesMap.getOrDefault(dateKey,
new ArrayList<>());
                    notesForDate.add(newNote);
                    notesMap.put(dateKey, notesForDate);

                    //updateNotesForSelectedDate();
                    updateSelectedDate();
                    editText.setText("");
                }
            }
```

```
        });

        loadNotesFromDatabase();
        updateSelectedDate();
    }

    private void setupListeners() {
        ImageButton buttonToMainActivity =
findViewById(R.id.buttonToMainActivity);
        buttonToMainActivity.setOnClickListener(view -> {
            backupHelper.backupDatabaseToFirebase();
            goToActivity(MenuActivity.class);
        });
    }

    private void goToActivity(Class<?> activityClass) {
        Intent intent = new Intent(this, activityClass);
        startActivity(intent);
    }

    private void loadNotesFromDatabase() {
        SQLiteDatabase db = dbHelper.getReadableDatabase();
        Cursor cursor = db.query("notes", null, null, null, null, null,
null);

        while (cursor.moveToNext()) {
            int id = cursor.getInt(cursor.getColumnIndexOrThrow("_id"));
            int year = cursor.getInt(cursor.getColumnIndexOrThrow("year"));
            int month = cursor.getInt(cursor.getColumnIndexOrThrow("month"));
            int day = cursor.getInt(cursor.getColumnIndexOrThrow("day"));
            String text =
cursor.getString(cursor.getColumnIndexOrThrow("text"));

            Note note = new Note(id, year, month, day, text);
            String dateKey = year + "-" + (month + 1) + "-" + day;
            List<Note> notesForDate = notesMap.getOrDefault(dateKey, new
ArrayList<>());
            notesForDate.add(note);
            notesMap.put(dateKey, notesForDate);
```

```
        }
        cursor.close();
        db.close();
    }

    private String getDateKey(Calendar calendar) {
        return calendar.get(Calendar.YEAR) + "-" +
(calendar.get(Calendar.MONTH) + 1) + "-" +
calendar.get(Calendar.DAY_OF_MONTH);
    }

    private void updateSelectedDate() {
        events.clear();
        for (Map.Entry<String, List<Note>> entry : notesMap.entrySet()) {
            Calendar cal = getCalendarFromKey(entry.getKey());
            if (dbHelper.getNotesForDate(cal.get(Calendar.YEAR),
cal.get(Calendar.MONTH), cal.get(Calendar.DAY_OF_MONTH)).size() > 0) {
                events.add(new EventDay(cal,
R.drawable.baseline_edit_note_24));
            }
        }
        if (selectedDate != null) {
            events.add(new EventDay(selectedDate, R.drawable.round_corner));
        }
        calendarView.setEvents(events);
    }

    private void updateNotesForSelectedDate() {
        if (selectedDate != null) {
            String dateKey = getDateKey(selectedDate);
            List<Note> notesForDate = dbHelper.getNotesForDate(
                    selectedDate.get(Calendar.YEAR),
                    selectedDate.get(Calendar.MONTH),
                    selectedDate.get(Calendar.DAY_OF_MONTH)
            );
            notesMap.put(dateKey, notesForDate);
            noteAdapter.setNotes(notesForDate);
            noteAdapter.notifyDataSetChanged();
        }
```

```
    }

    private Calendar getCalendarFromKey(String key) {
        String[] parts = key.split("-");
        Calendar cal = Calendar.getInstance();
        cal.set(Calendar.YEAR, Integer.parseInt(parts[0]));
        cal.set(Calendar.MONTH, Integer.parseInt(parts[1]) - 1);
        cal.set(Calendar.DAY_OF_MONTH, Integer.parseInt(parts[2]));
        return cal;
    }

    @Override
    public void onEdit(Note note) {
        // Show a dialog or another activity to edit the note
        // For simplicity, let's use an AlertDialog with an EditText
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Edit Note");

        final EditText input = new EditText(this);
        input.setText(note.getText());
        builder.setView(input);

        builder.setPositiveButton("Save", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                note.setText(input.getText().toString());
                dbHelper.updateNote(note);
                updateNotesForSelectedDate();
                updateSelectedDate();
            }
        });
        builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.cancel();
            }
        });
```

```
        builder.show();
    }

    @Override
    public void onDelete(Note note) {
        dbHelper.deleteNoteById(note.getId());
        String dateKey = getDateKey(selectedDate);
        List<Note> notesForDate = notesMap.get(dateKey);
        if (notesForDate != null) {
            notesForDate.remove(note);
            notesMap.put(dateKey, notesForDate);
        }
        updateNotesForSelectedDate();
        updateSelectedDate();
    }

    @Override
    protected void onStart() {
        super.onStart();
        currentUser = getIntent().getParcelableExtra("userDetails");
        Log.d("CalendarActivity", "Current user attached");
        if (currentUser != null) {
            Log.d("Firebase UID", currentUser.getUid());
            backupHelper = new BackupHelper(this, currentUser);  //
Ініціалізація BackupHelper з currentUser

            backupHelper.checkDatabaseExists(new
BackupHelper.BackupCallback() {
                @Override
                public void onSuccess() {
                    // Database exists, download it
                    backupHelper.downloadDatabaseFromFirebase(new
BackupHelper.BackupCallback() {
                        @Override
                        public void onSuccess() {
                            loadNotesFromDatabase();
                            updateSelectedDate();
                        }
```

```
                            @Override
                            public void onFailure() {
                                Toast.makeText(CalendarActivity.this, "Failed to
        download database", Toast.LENGTH_SHORT).show();
                            }
                        });
                    }

                    @Override
                    public void onFailure() {
                        // Database does not exist, create a new one
                        Log.d("Firebase", "Database does not exist in storage,
        creating new one");
                        dbHelper.createNewTable();
                        loadNotesFromDatabase(); // This will create a new
        database
                        updateSelectedDate();
                    }
                });
            } else {
                Log.d("Firebase UID", "currentUser is null");
            }
        }

        @Override
        protected void onNewIntent(Intent intent) {
            super.onNewIntent(intent);
            setIntent(intent);
            currentUser = getIntent().getParcelableExtra("userDetails");
        }

        @Override
        protected void onDestroy() {
            super.onDestroy();
            backupHelper.backupDatabaseToFirebase();

        }
}
```

NoteAdapter.java

```
public class NoteAdapter extends
RecyclerView.Adapter<NoteAdapter.NoteViewHolder> {
    private List<Note> notes;
    private OnNoteEditListener onNoteEditListener;
    private OnNoteDeleteListener onNoteDeleteListener;

    public NoteAdapter(List<Note> notes, OnNoteEditListener
onNoteEditListener, OnNoteDeleteListener onNoteDeleteListener) {
        this.notes = notes;
        this.onNoteEditListener = onNoteEditListener;
        this.onNoteDeleteListener = onNoteDeleteListener;
    }

    @Override
    public NoteViewHolder onCreateViewHolder(ViewGroup parent, int viewType)
{
        View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.notes_item, parent,
false);
        return new NoteViewHolder(view);
    }

    @Override
    public void onBindViewHolder(NoteViewHolder holder, int position) {
        Note note = notes.get(position);
        holder.noteText.setText(note.getText());

        holder.noteDelete.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onNoteDeleteListener.onDelete(note);
            }
        });

        holder.noteEdit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onNoteEditListener.onEdit(note);
```

```
            }
        });
    }

    @Override
    public int getItemCount() {
        return notes.size();
    }

    public void setNotes(List<Note> newNotes) {
        this.notes = newNotes;
        notifyDataSetChanged();
    }

    public static class NoteViewHolder extends RecyclerView.ViewHolder {
        public TextView noteText;
        public ImageButton noteDelete, noteEdit;

        public NoteViewHolder(View itemView) {
            super(itemView);
            noteText = itemView.findViewById(R.id.note_title);
            noteDelete = itemView.findViewById(R.id.button_delete);
            noteEdit = itemView.findViewById(R.id.button_edit);
        }
    }

    public interface OnNoteEditListener {
        void onEdit(Note note);
    }

    public interface OnNoteDeleteListener {
        void onDelete(Note note);
    }
}
```