КИЇВСЬКИЙ СТОЛИЧНИЙ УНІВЕРСИТЕТ ІМЕНІ БОРИСА ГРІНЧЕНКА
Факультет інформаційних технологій та математики
Кафедра комп'ютерних наук

Допущено до захисту

Завідувач кафедри комп'ютерних
наук к.т.н. доцент кафедри
комп'ютерних наук

Машкіна Ірина Вікторівна

«___» _____ 202__ р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня «бакалавр»

Спеціальність 122 Комп'ютерні науки

Освітня програма 122.00.01 Інформатика

Тема: РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ ДЛЯ БРОНЮВАННЯ КНИГ ТА ВІДСТЕЖЕННЯ ІСТОРІЇ ВІДВІДУВАНЬ БІБЛІОТЕКИ

Виконала

студентка групи ІНБ-2-21-4.0д

Нітчук Ірина Іванівна

Науковий керівник

к.т.н., доцент кафедри

комп'ютерних наук

Носенко Тетяна Іванівна

Київ – 2025

КИЇВСЬКИЙ СТОЛИЧНИЙ УНІВЕРСИТЕТ ІМЕНІ БОРИСА ГРІНЧЕНКА

Факультет інформаційних технологій та математики

Кафедра комп'ютерних наук

«Затверджую»

Завідувач кафедри комп'ютерних наук
к.т.н. доцент кафедри комп'ютерних наук

Машкіна Ірина Вікторівна

(підпис)

«___» _____ 202__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

«Розробка мобільного додатку для бронювання книг та відстеження історії відвідувань»

Виконавець – студентка групи ІНб-2-21-4.0д,

спеціальності 122 Комп'ютерні науки,

освітньої програми 122.00.01 Інформатика

Нітчук Ірина Іванівна

1. Вихідні дані: тематичні українські та зарубіжні освітні ресурси. Наукова, навчальна та періодична та методична література за темою роботи.
2. Основні завдання: провести аналіз існуючих аналогів та виявити їх сильні та слабкі сторони, визначити функціональні й технічні вимоги до майбутнього додатку, спроєктувати архітектуру системи та інтерфейс користувача, реалізувати основні функціональні модулі додатку, протестувати готовий застосунок та оцінити результати.
Реалізувати наступний функціонал: реєстрація та авторизація користувачів із використанням Firebase Authentication, збереження і перегляд інформації про книги з бази даних Cloud Firestore, можливість бронювання книг, створення запитів на повернення та оновлення статусу, ведення історії відвідувань та рейтингування книг, панель бібліотекаря з функціями підтвердження повернень, редагування книг, надсилання повідомлень.
3.Пояснювальна записка**:** Обсяг – до 70 сторінок формату А4 комп'ютерного набору з дотриманням вимог стандарту і методичних рекомендацій кафедри.
4. Графічні матеріали: Діаграми варіантів використання (Use Case), блок-схеми логіки додатку, архітектурна схема взаємодії з Firebase.
5. Додатки: не передбачено
6. Строк подання роботи на кафедру: «06» червня 2025 р.

Науковий керівник:

(підпис)

к.т.н., доцент кафедри комп'ютерних наук

Носенко Тетяна Іванівна

Виконавець:

Нітчук Ірина Іванівна

(підпис)

Календарний план

№	Етапи виконання	Термін виконання	Відмітка
1	Формування тем дипломних робіт бакалаврів	Жовтень	Виконано
2	Визначення і затвердження теми бакалаврської роботи	Жовтень-листопад	Виконано
3	Початок роботи над кваліфікаційною роботою. Теоретична складова роботи	Грудень	Виконано
4	Написання тез до конференції ІТ-2025	Лютий-березень	Виконано
5	Аналіз програмних засобів та створення проекту. Програмна складова роботи	Березень	Виконано
6	Створення мобільного додатку по темі кваліфікаційної роботи	Квітень-травень	Виконано
7	Передзахист кваліфікаційної роботи	Травень	Виконано
8	Захист кваліфікаційної роботи	Червень	

РЕФЕРАТ бакалаврської роботи

Кваліфікаційна робота містить: 4 таблиці, 14 ілюстрацій, 38 джерел

Сучасні бібліотеки все частіше стикаються з потребою впровадження цифрових сервісів, щоб зробити обслуговування користувачів зручнішим і ефективнішим. Одним із таких сервісів є бронювання книг через мобільні додатки, що дозволяє читачам швидко замовити потрібну літературу без зайвих черг, дзвінків або довгого пошуку книг. Розробка мобільного застосунку з функціями бронювання та перегляду історії запитів допомагає не лише покращити взаємодію користувача з бібліотекою, а й оптимізувати роботу персоналу. Це особливо актуально в умовах переходу до дистанційних форм обслуговування та зростання кількості мобільних користувачів.

Об’єкт дослідження: мобільний додаток для бронювання книг та відстеження історії відвідувань бібліотеки

Предмет дослідження: методи, засоби та технології розробки мобільних додатків для інтерактивної взаємодії користувача з бібліотечною системою

Мета дослідження: розробка зручного та функціонального мобільного застосунку, який дозволяє користувачам бібліотеки здійснювати онлайн-бронювання книг, відстежувати історію своїх відвідувань, бронювань і прогресу в читані

Для досягнення мети вирішено такі завдання:

– проведено аналіз літературних джерел та аналогів

– визначено вимоги до застосунку та обрано інструменти розробки

–спроєктовано архітектуру та інтерфейс мобільного додатку

– реалізовано основні функціональні модулі (авторизація, реєстрація, бронювання,

історія)
– проведено тестування та аналіз роботи додатку

У результаті виконання роботи створено базову версію мобільного додатку, яка дозволяє користувачеві авторизуватись, здійснювати бронювання книг, переглядати історію бронювань, видаляти або шукати записи. Програма реалізована у середовищі Flutter з використанням мови Dart.

Результати роботи можуть бути використані в бібліотеках для оптимізації обліку користувачів і бронювань, а також як основа для розширення сервісів бібліотечного самообслуговування.

Ключові слова: МОБІЛЬНИЙ ДОДАТОК, БРОНЮВАННЯ, БІБЛІОТЕКА, ІСТОРІЯ ВІДВІДУВАНЬ, FLUTTER, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, АВТОМАТИЗАЦІЯ, ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, КОРИСТУВАЧ, РЕЄСТРАЦІЯ.

Зміст

ВСТУП

У сучасному інформаційному суспільстві цифровізація бібліотечної сфери є не лише трендом, а й необхідністю. З огляду на стрімке зростання кількості користувачів мобільних пристроїв, зростає й попит на сервіси, що надають доступ до бібліотечних ресурсів без потреби у фізичному відвідуванні установи. Однією з таких послуг є можливість онлайн-бронювання книг, яка вже стала звичною практикою у багатьох країнах світу.

У провідних зарубіжних бібліотеках подібні функціональні сервіси давно інтегровані у мобільні застосунки, що спрощує обслуговування читачів і водночас оптимізує роботу персоналу. В Україні ж такі рішення лише починають впроваджуватись. Потреба у створенні адаптованих локальних систем, які враховують специфіку вітчизняних бібліотек та запити користувачів, залишається актуальною.

У наукових дослідженнях переважають приклади розробки веборієнтованих інформаційних систем для бібліотек, натомість мобільні додатки подібного призначення є рідкісними і здебільшого представлені комерційними закордонними продуктами зо–крема, Libby, OverDrive, BookMyne. Ці рішення мають обмежене застосування в українських реаліях, оскільки прив’язані до конкретних платформ або бібліотечних мереж. Наявні українські напрацювання мають експериментальний характер і не охоплюють повний спектр необхідного функціоналу.

Актуальність теми зумовлена потребою впровадження сучасних інформаційних технологій у бібліотечну сферу, а також зростаючим попитом

користувачів на зручні, інтуїтивно зрозумілі мобільні сервіси для бронювання книг, перегляду історії взаємодії та отримання повідомлень від бібліотеки.

Мета дослідження: розробка зручного та функціонального мобільного застосунку, який дозволяє користувачам бібліотеки здійснювати онлайн-бронювання книг, відстежувати історію своїх відвідувань, бронювань і прогресу в читані

Для досягнення поставленої мети необхідно вирішити такі завдання:

- провести аналіз існуючих аналогів та виявити їх сильні та слабкі сторони;
- визначити функціональні й технічні вимоги до майбутнього додатку;
- спроєктувати архітектуру системи та інтерфейс користувача;
- реалізувати основні функціональні модулі додатку;
- протестувати готовий застосунок та оцінити результати.

Об’єкт дослідження: мобільний додаток для бронювання книг та відстеження історії відвідувань бібліотеки

Предмет дослідження: методи, засоби та технології розробки мобільних додатків для інтерактивної взаємодії користувача з бібліотечною системою

У роботі застосовано такі методи дослідження: аналіз літературних джерел і прикладних рішень, порівняльний аналіз, об’єктно-орієнтоване проєктування, моделювання інтерфейсу, розробка з використанням фреймворку Flutter та тестування функціоналу.

Практичне значення дослідження полягає у створенні повнофункціонального прототипу мобільного застосунку, який може бути використаний для автоматизації роботи бібліотек, а також як приклад для навчання студентів спеціальності «Комп’ютерні науки».

Галузь застосування – інформаційні системи для бібліотек, освітні заклади, установи культури та сфери обслуговування користувачів інформаційних ресурсів.

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМИ ТА ОБҐРУНТУВАННЯ ТЕМИ

1.1 Актуальність цифрової трансформації бібліотечної сфери

У сучасному цифровому суспільстві мобільні технології відіграють ключову роль у трансформації різних сфер діяльності, зокрема бібліотечної. Користувачі дедалі частіше очікують можливості отримувати послуги дистанційно – без фізичного відвідування закладів культури та без прив’язки до робочого часу бібліотеки.

За даними платформи Statista, станом на 2024 рік кількість користувачів смартфонів у світі перевищила 6,9 мільярда осіб, що становить понад 85% населення планети [1]. Водночас в Україні частка мобільного доступу до інтернету залишається стабільно високою: за даними звіту Digital 2023: Ukraine, 91,6% користувачів інтернету щоденно використовують смартфони для доступу до цифрових сервісів, зокрема освітніх і культурних [2].

Такий рівень проникнення мобільних пристроїв створює об’єктивну потребу в оптимізації бібліотечних сервісів, орієнтованих на мобільну взаємодію. Особливо актуальним стає впровадження мобільних застосунків, які дозволяють дистанційно бронювати книги, переглядати історію взаємодії з бібліотекою, залишати оцінки та зворотний зв’язок, а також отримувати повідомлення від працівників установи.

В Україні наразі переважають традиційні форми бібліотечного обслуговування або веб-інтерфейси, що не завжди зручні для мобільних користувачів. Наявність універсального мобільного додатку з адаптацією під локальні умови дозволить зробити доступ до бібліотечних послуг значно швидшим і зручнішим, що, у свою чергу, сприятиме популяризації читання та підвищенню рівня цифрової інклюзії.

1.2 Класифікація бібліотечних інформаційних систем

Бібліотечні системи є ключовими інструментами для автоматизації процесів у бібліотеках, що дозволяють ефективно управляти фондами, користувачами та інформаційними ресурсами. Ці системи різняться за функціональністю, способом доступу, рівнем складності та цільовим призначенням. Для глибшого розуміння їхньої ролі в цифровій трансформації бібліотечної сфери розглянемо більш детально класифікацію бібліотечних систем за кількома основними критеріями.

Перший критерій класифікації бібліотечних систем – це тип автоматизації, який визначає рівень охоплення бібліотечних процесів. За цим критерієм системи поділяються на інтегровані та спеціалізовані. Інтегровані бібліотечні системи, або ILS (Integrated Library Systems), є комплексними програмними рішеннями, що охоплюють усі аспекти роботи бібліотеки, такі як каталогізація, облік фондів, видача книг, управління користувачами, генерація звітів та інтеграція з електронними каталогами. Такі системи зазвичай базуються на єдиній базі даних, що дозволяє централізовано зберігати й обробляти інформацію. Прикладами таких систем є Koha, яка є відкритим програмним забезпеченням, Aleph, комерційний продукт від Ex Libris, та Sierra від Innovative Interfaces. Вони підтримують стандарти MARC (Machine-Readable Cataloging), інтеграцію з OPAC (Online Public Access Catalog) і мають модульну структуру з можливістю розширення. Наприклад, Koha дозволяє бібліотекарям створювати власні модулі для специфічних потреб, таких як облік читацьких квитків. Перевагами таких систем є повна автоматизація процесів, висока гнучкість і можливість адаптації до потреб різних бібліотек, від невеликих до національних. Однак вони мають і недоліки, зокрема високу вартість ліцензій і підтримки для комерційних систем, як у випадку з Aleph, а також складність впровадження та необхідність підготовки персоналу. В Україні інтегровані

системи використовуються переважно у великих бібліотеках, таких як Національна бібліотека України імені В. І. Вернадського, де іноді застосовуються локалізовані версії Koha, але їхнє впровадження залишається обмеженим через фінансові обмеження.

Спеціалізовані системи, на відміну від інтегрованих, призначені для виконання окремих завдань, таких як управління електронними ресурсами чи створення електронних каталогів. Вони менш універсальні, але простіші у впровадженні. Прикладами є OPAC-системи, такі як VuFind, або системи управління електронними ресурсами, наприклад OpenERMS. Такі системи фокусуються на конкретній функції: OPAC дозволяє користувачам шукати книги онлайн, але не управляє видачею чи бронюванням. Їхніми перевагами є низька вартість, швидке впровадження та достатність для невеликих бібліотек. Проте вони мають обмежений функціонал і не інтегруються з іншими процесами, що вимагає використання кількох систем одночасно. В Україні спеціалізовані системи часто використовуються в невеликих бібліотеках для створення простих електронних каталогів.

Другий критерій – спосіб доступу, який визначає, як користувачі та бібліотекарі отримують доступ до системи: локально чи через хмарні технології. Локальні системи передбачають встановлення програмного забезпечення на серверах або комп’ютерах бібліотеки, а доступ до даних можливий лише в межах бібліотеки або через локальну мережу. Прикладами таких систем є УФД/Бібліотека, розробка українських ІТ-фахівців, а також застарілі версії SIRSI. Вони характеризуються низькою залежністю від інтернету та можливістю роботи в офлайн-режимі. Дані зберігаються на фізичних носіях, що забезпечує певний рівень безпеки. Такі системи не потребують постійного підключення до мережі, що корисно в регіонах із нестабільним інтернетом. Однак вони мають обмеження для віддалених користувачів, ускладнене оновлення програмного

забезпечення та високу вартість обслуговування серверів. В Україні локальні системи поширені в бібліотеках, де немає доступу до високошвидкісного інтернету, але їх використання поступово зменшується.

Хмарні системи, навпаки, зберігають дані на віддалених серверах, до яких користувачі отримують доступ через інтернет. Вони підтримують реальний час оновлення та інтеграцію з мобільними додатками. Прикладами є WorldShare Management Services (WMS) від OCLC та Alma від Ex Libris. Такі системи дозволяють синхронізувати дані в реальному часі, забезпечують доступ із будь-якого пристрою – ПК, смартфона чи планшета – і автоматично оновлюються. Наприклад, WMS дає змогу бібліотекарям керувати фондами через веб-інтерфейс, а користувачам – бронювати книги онлайн. Перевагами хмарних систем є гнучкість, масштабованість, зниження витрат на локальне обладнання та можливість інтеграції з зовнішніми сервісами, такими як платіжні системи. Однак вони залежать від стабільного інтернет-з’єднання, мають потенційні ризики конфіденційності даних і передбачають абонентську плату за використання. В Україні хмарні системи практично не використовуються через високу вартість і нестабільність інтернету в деяких регіонах, але вони можуть бути перспективними для великих міських бібліотек.

Третій критерій – рівень доступності для користувачів, який визначає, хто має доступ до системи: лише бібліотекарі чи також читачі. Системи з обмеженим доступом призначені виключно для внутрішнього використання бібліотекарями, а користувачі не мають прямого доступу до них. Прикладами є деякі версії УФД/Бібліотека та локальні системи без веб-інтерфейсу. Вони фокусуються на внутрішній автоматизації, такій як облік книг і складання звітів, але не надають онлайн-доступу для читачів. Такі системи забезпечують вищий рівень контролю над даними та прості для використання в невеликих бібліотеках. Однак вони обмежують можливості дистанційного обслуговування

та вимагають ручного введення запитів від користувачів. В Україні такі системи поширені в невеликих бібліотеках, де читачі все ще звертаються до бібліотекаря особисто.

Системи з відкритим доступом надають користувачам доступ до певних функцій через веб-інтерфейс або мобільні додатки, наприклад, пошук книг, бронювання, перегляд історії. Прикладами є Alma від Ex Libris, Koha з інтегрованим OPAC і WorldCat. Вони підтримують електронні каталоги, інтеграцію з мобільними платформами та можливість авторизації користувачів. Наприклад, система Alma дозволяє читачам переглядати доступність книг і бронювати їх через додаток. Такі системи зручні для користувачів, підвищують доступність бібліотечних послуг і зменшують навантаження на персонал. Однак вони потребують забезпечення безпеки даних і складного налаштування доступу. В Україні системи з відкритим доступом використовуються переважно в академічних бібліотеках, але потребують локалізації.

Четвертий критерій – тип бібліотеки, який враховує специфіку цільової аудиторії та розміру бібліотеки. Системи для публічних бібліотек орієнтовані на широке коло користувачів, забезпечують простий інтерфейс і базові функції, такі як пошук книг, бронювання та видача. Прикладами є Evergreen і OpenBiblio. Вони характеризуються простотою використання, підтримкою багатомовності та інтеграцією з локальними каталогами. Evergreen, наприклад, підтримує роботу з великою кількістю користувачів і дозволяє налаштовувати інтерфейс під потреби громади. Такі системи доступні для невеликих бібліотек, а деякі, як OpenBiblio, мають безкоштовні версії. Однак вони мають обмежений функціонал для складних завдань, таких як аналітика. В Україні такі системи могли б бути перспективними для громадських бібліотек, але потребують адаптації.

Системи для наукових і академічних бібліотек мають розширені функції, такі як управління дослідницькими даними, інтеграція з науковими базами, наприклад Scopus чи Web of Science, а також підтримка міжбібліотечного абонементу. Прикладами є Primo від Ex Libris і WorldShare від OCLC. Вони підтримують складні запити, аналітику використання ресурсів та інтеграцію з університетськими системами. Наприклад, Primo дозволяє студентам шукати статті в наукових журналах прямо в бібліотечному каталозі. Такі системи мають високу функціональність і підходять для великих бібліотек, але вони дорогі та складні у впровадженні. В Україні вони використовуються в університетських бібліотеках, наприклад у Київському національному університеті імені Тараса Шевченка, але переважно в пілотних проєктах.

Додатковим критерієм є технологічна основа, яка враховує, на яких технологіях базуються системи. Відкриті системи, або Open Source, мають відкритий код, який можна модифікувати під потреби бібліотеки. Прикладами є Koha і Evergreen. Вони безкоштовні, гнучкі, але залежать від спільноти розробників. Їхні переваги – низька вартість і можливість адаптації, але вони потребують технічної підтримки з боку бібліотеки. Комерційні системи, такі як Alma і Sierra, є платними рішеннями з професійною підтримкою від розробників. Вони мають гарантовану технічну підтримку та регулярні оновлення, що забезпечує надійність і повну документацію. Однак їхня висока вартість ліцензій є значним недоліком.

1.3 Огляд зарубіжних рішень та порівняння їх функціональних можливостей

Одним із важливих етапів під час розробки власного інформаційного рішення є аналіз уже існуючих аналогів. У сфері цифрового обслуговування бібліотек активно використовуються мобільні додатки, що забезпечують

дистанційний доступ до літератури та взаємодію користувачів із бібліотечними системами. Найбільш відомими прикладами таких застосунків є Libby, OverDrive та MyLibrary [3, 4], які реалізовані у США, Канаді, Великій Британії та інших країнах.

У таблиці 1 наведено порівняльний аналіз функціональних можливостей зазначених додатків за критеріями: кількість завантажень, рейтинг користувачів, наявність бронювання книг, можливість відстеження історії взаємодії, а також способи інтеграції з бібліотеками.[5]

Таблиця 1 – Порівняння функціональності бібліотечних додатків
(на основі відкритих даних Google Play, App Store, OverDrive.com)

Таблиця 1. Порівняння функціональності бібліотечних додатків

Додаток	**К-сть завантажень (млн)**	**Оцінка користувачів**	**Бронювання книг**	**Відстеження історії відвідувань**	**Інтеграція з бібліотеками**
Libby	10+	4.7/5	Доступне онлайн-бронювання книг	Відсутня функція перегляду історії	Підключення до партнерських бібліотек
OverDrive	5+	4.6/5	Можливість бронювання та завантаження аудіо- та електронних книг	Відсутня історія фізичних відвідувань	Підключення до бібліотек через акаунт
MyLibrary	1+	4.3/5	Обмежене бронювання друкованих книг	Історія читання доступна тільки для електронних книг	Доступне лише для певних бібліотек

Як видно з таблиці, жоден із проаналізованих додатків не надає комплексної підтримки усіх аспектів взаємодії з бібліотекою. Наприклад, додатки Libby та OverDrive не зберігають історію фізичних відвідувань або замовлень користувача. MyLibrary має дещо розширені можливості, проте доступний лише для обмеженого кола бібліотек. Крім того, деякі додатки вимагають підключення до міжнародних бібліотечних платформ, що унеможливлює їхнє повноцінне впровадження в українських умовах.[6]

Зважаючи на це, виникає об’єктивна потреба у створенні локалізованого мобільного додатку, який забезпечуватиме бронювання, історію взаємодій, повідомлення та доступ до бібліотечних послуг без залежності від сторонніх систем. Саме ці функції мають бути враховані при розробці нового рішення.

Висновки до розділу 1

У першому розділі було проаналізовано актуальність цифрової трансформації бібліотечної сфери в умовах стрімкого зростання використання мобільних технологій. Визначено, що традиційні форми взаємодії з бібліотеками вже не повною мірою відповідають очікуванням сучасного користувача, який потребує швидкого, зручного та віддаленого доступу до послуг. Зокрема, бронювання книг, перегляд історії користування та отримання сповіщень є важливими елементами, які слід забезпечити в межах цифрових сервісів.

Проведено огляд популярних зарубіжних мобільних додатків (Libby, OverDrive, MyLibrary), що реалізують частину необхідного функціоналу, проте не враховують повністю потреб українського користувача, не підтримують фіксацію історії фізичних відвідувань і часто орієнтовані лише на певні бібліотечні системи або країни. Це підтверджує наявність ніші для створення нового локалізованого мобільного рішення.

Таким чином, обґрунтовано доцільність розробки мобільного додатку, адаптованого до умов українських бібліотек, який поєднує зручність мобільного сервісу з повноцінною підтримкою бронювання, обліку взаємодії та інформування користувачів. Запропоноване рішення здатне не лише задовольнити сучасні запити користувачів, а й сприяти подальшій цифровій трансформації бібліотечної галузі в Україні.

РОЗДІЛ ІІ ТЕХНОЛОГІЇ, ІНСТРУМЕНТИ ТА ПРОЄКТУВАННЯ ДОДАТКУ

2.1 Обґрунтування вибору програмних засобів

У сучасному світі розробка мобільних додатків є однією з найдинамічніших галузей IT. Завдяки високому попиту на мобільні сервіси, розробникам доступна велика кількість фреймворків, мов програмування та інструментів. Найпоширенішими серед них є Flutter, React Native, Xamarin, Kotlin Multiplatform, SwiftUI. Вибір того чи іншого інструменту залежить від специфіки роботи, вимог до продуктивності, бюджету, платформи, а також досвіду команди.[7]

У таблиці нижче наведено порівняльну характеристику трьох популярних фреймворків для кросплатформенної розробки:

Таблиця 2.1 – Порівняння фреймворків для розробки мобільних застосунків

Критерій	Flutter	React Native	Xamarin
Підтримка платформ	Android, iOS, Web, Desktop	Android, iOS	Android, iOS, Windows
Мова програмування	Dart	JavaScript	C#
Продуктивність	Висока	Середня	Середня
Візуальні компоненти	Власний UI (без нативних елементів)	Використовує нативні елементи	Частково нативні елементи
Швидкість розробки	Висока (hot reload)	Висока (hot reload)	Нижча
Спільнота та підтримка	Дуже активна	Дуже активна	Менш активна
Офіційна підтримка	Google	Meta (Facebook)	Microsoft

Flutter був обраний як основний інструмент через низку переваг: зручність у використанні, підтримка великої кількості платформ, активна спільнота, функція hot reload, наявність численної документації. Він дозволяє створити єдиний код, який працює як на Android, так і на iOS.

Наступним важливим компонентом став вибір системи бекенду. У розробці застосунків найчастіше використовуються такі сервіси, як Firebase, Supabase, Backendless, AWS Amplify. Їхні можливості різняться: одні зосереджені на зберіганні даних, інші – на обробці запитів або авторизації.[8,9]

У таблиці нижче наведено порівняльний аналіз трьох сервісів:

Таблиця 2.2 – Порівняння хмарних бекенд-сервісів

Критерій	**Firebase (Google)**	**Supabase**	**Backendless**
Тип бази даних	Документоорієнтована (NoSQL)	Реляційна (PostgreSQL)	NoSQL
Реєстрація / логін	Є (Firebase Auth)	Є	Є
Інтеграція з Flutter	Відмінна	Обмежена	Через REST API
Безкоштовний тариф	Є	Є	Є
Дашборд адміністратора	Зручний	Середній	Досить зручний
Особливості	Широкий набір сервісів, push, ML	SQL-запити, open source	Хостинг, API, користувачі

Firebase надає готові рішення для авторизації, бази даних, хостингу, повідомлень і багато іншого. Його обрання для дипломної роботи зумовлене відсутністю необхідності створювати власний сервер, широкою документацією, активною підтримкою Flutter і гнучкістю NoSQL бази даних.[9]

Ще одним елементом є вибір середовища розробки. Для створення та тестування застосунку використовуються редактори з підтримкою Flutter та можливістю швидкої перевірки результату на емуляторі. Найбільш популярні середовища – Visual Studio Code, Android Studio та Xcode.

Таблиця 2.3 – Порівняння середовищ розробки

Середовище	Основне призначення	Переваги	Недоліки
VS Code	Написання коду, тестування	Швидкий, легкий, багато плагінів	Немає вбудованого емулятора
Android Studio	Повна розробка та тестування Android	Повна підтримка Android, емулятор	Важкий для системи
Xcode	Для iOS	Потрібен для iOS, симулятор iPhone	Тільки для macOS

Visual Studio Code був обраний як головне середовище розробки (рис.2.1) завдяки легкості, великій кількості плагінів для Flutter і можливості інтеграції з Android Studio для запуску емулятора. Сам Android Studio використовувався саме для запуску віртуального пристрою.[10]

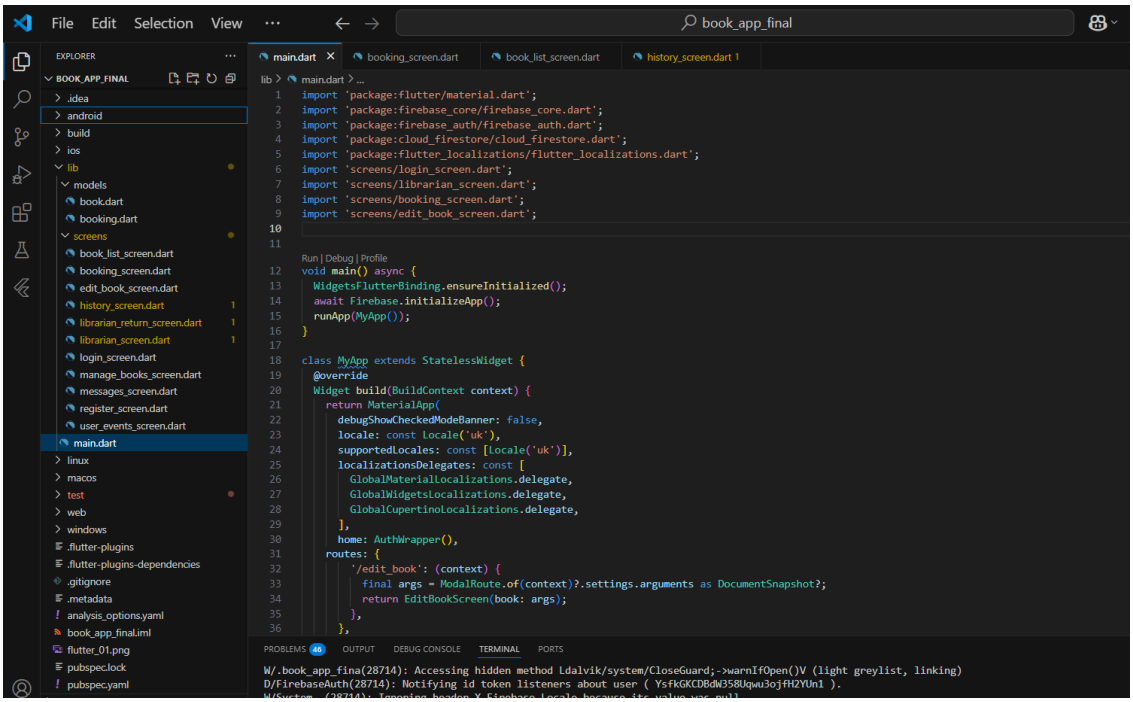

Рис. 2.1 – Структура проєкту у Flutter

2.2 Архітектура системи

Архітектура розробленого мобільного додатку базується на клієнт-серверній моделі з використанням хмарного бекенду Firebase. Основними компонентами системи є:

Клієнтська частина – мобільний додаток, розроблений на Flutter. Вона відповідає за взаємодію з користувачем, відображення інтерфейсу, формування запитів до бази даних та обробку відповідей.

Хмарна база даних Firestore – використовується для зберігання даних про користувачів, книги, бронювання, історію відвідувань, повідомлення тощо. Дані зберігаються у вигляді документів у відповідних колекціях.

Firebase Authentication – сервіс для реєстрації та авторизації користувачів. Підтримує простий спосіб входу за допомогою електронної пошти та пароля.

Firebase Cloud Messaging – використовується для надсилання повідомлень користувачам (наприклад, нагадувань про повернення книги).

Система передбачає наявність двох ролей користувачів: звичайний користувач (читач) та бібліотекар. Ролі визначають доступ до функціоналу інтерфейсу(рис.2.2).

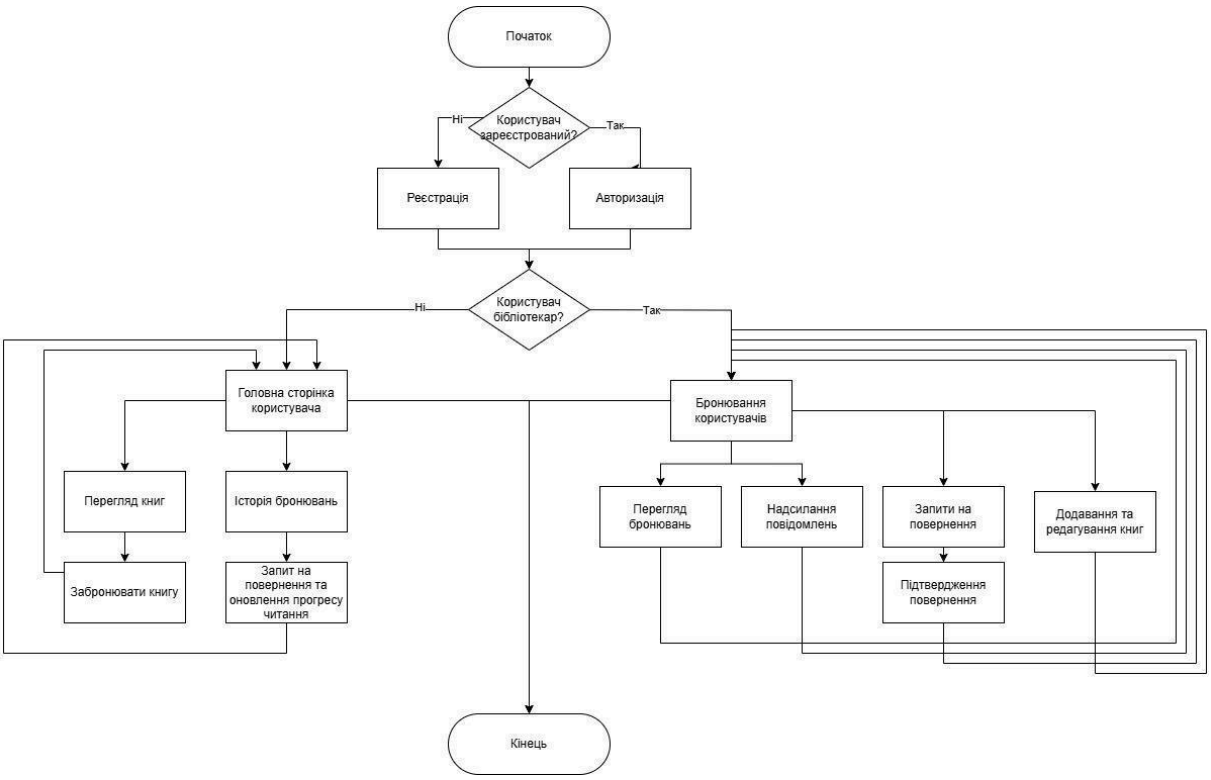

Рисунок 2.2. – Блок-схема додатку

Інформаційна взаємодія реалізується через запити додатку до Firestore, а автентифікація здійснюється через Firebase Auth. Усі зміни в даних оновлюються в реальному часі завдяки особливостям роботи Firestore.

2.3 Архітектура, модульність і проєктування основного функціоналу додатку

Процес проєктування архітектури мобільного додатку відіграє ключову роль у забезпеченні стабільної, надійної та масштабованої роботи системи. У межах цієї дипломної роботи було прийнято рішення про використання клієнт-серверної архітектури з орієнтацією на хмарну інфраструктуру, що дозволяє зменшити навантаження на пристрої користувачів та забезпечити доступ до даних у режимі реального часу.

Загальна архітектура складається з двох основних компонентів: фронтенд-частини (мобільного застосунку, розробленого за допомогою фреймворку Flutter) та бекенд-частини (хмарної платформи Firebase, зокрема її сервісів Authentication і Firestore).

Мобільний застосунок відповідає за взаємодію з користувачем: відображення інтерфейсу, обробку дій, введення та виведення інформації, навігацію між екранами, обробку подій та виклики до бази даних. Уся логіка реалізована на стороні клієнта, що забезпечує швидкий відгук додатку на дії користувача.

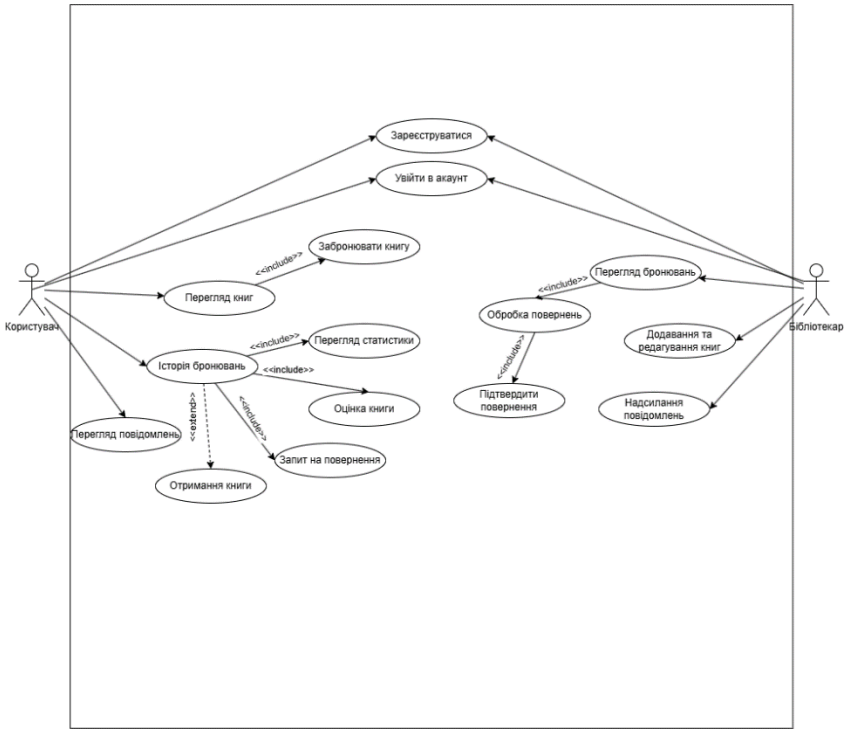

Рисунок 2.2 – Діаграма прецедентів мобільного додатку для бронювання книг

У той же час Firebase забезпечує функціональність, пов’язану зі зберіганням і обробкою даних. Firestore, як документно-орієнтована NoSQL-база даних, зберігає всю інформацію у вигляді колекцій і документів. Було створено такі ключові колекції:

users – для зберігання особистих даних користувачів (ім’я, email, роль, телефон тощо);

books – колекція книг, що містить назву, автора, жанр, кількість примірників, зображення тощо;

bookings – бронювання книг, пов’язане з користувачем і конкретною книгою;

return_requests – запити на повернення книг, які надсилаються до бібліотекаря;

messages – повідомлення, які бібліотекар може надсилати користувачу;

visits – історія відвідувань бібліотеки та взаємодій (відвідування, взяття та повернення книг).

Взаємодія між додатком і Firebase здійснюється через API, що надається офіційними бібліотеками Flutter. Це дозволяє реалізовувати запити до бази, зчитувати, фільтрувати та змінювати дані у режимі реального часу без використання окремого серверного середовища.

Особливістю обраної архітектури є розмежування доступу за ролями. Користувачі з правами «бібліотекаря» отримують доступ до додаткових функцій: підтвердження повернення книг, додавання та редагування записів про книги, перегляду активних бронювань усіх користувачів. Звичайні користувачі мають змогу бронювати книги, відстежувати історію своїх дій, надсилати запити на повернення книг, залишати відгуки й переглядати отримані повідомлення.

Для збереження індивідуальних налаштувань і забезпечення постійного входу в систему використовується SharedPreferences. Завдяки цьому користувачеві не потрібно повторно вводити облікові дані при кожному запуску застосунку.

Побудована архітектура виявилась ефективною, оскільки поєднує гнучкість клієнтського рішення з надійністю хмарних сервісів. Вона дозволяє легко масштабувати роботу, додавати нові функціональні можливості без необхідності повного переписування коду, а також адаптувати систему під потреби реальної бібліотеки.

2.4 Вибір структури даних і організація колекцій у Firestore

Для ефективного зберігання, пошуку та обробки інформації в мобільному додатку було обрано хмарну NoSQL-базу даних Firestore, яка є частиною екосистеми Firebase. Її документно-орієнтована модель зберігання дозволяє гнучко організовувати дані у вигляді колекцій і документів, що ідеально підходить для мобільних застосунків, які потребують швидкого обміну даними в реальному часі.[11]

У процесі проєктування структури даних було враховано принципи нормалізації, розділення відповідальностей та мінімізації надлишкових запитів. Кожен об’єкт у системі (користувач, книга, бронювання тощо) зберігається в окремій колекції, а необхідні зв’язки реалізуються через унікальні ідентифікатори (ID).

Основні колекції, які реалізовані у Firestore:

users – містить дані зареєстрованих користувачів, включаючи email, ім’я, роль (user або librarian), номер телефону, а також інші персоналізовані параметри.

books – колекція всіх книг у бібліотеці. Кожен документ включає поля: назва, автор, жанр, кількість примірників, URL зображення обкладинки.

bookings – зберігає інформацію про бронювання книг, зокрема ID користувача, ID книги, дату створення бронювання, статус (активне/завершене).

return_requests – запити на повернення книг, що надсилаються користувачем і підтверджуються бібліотекарем.

Visits – історія дій користувача в бібліотеці (наприклад, "відвідування", "взяв книгу", "повернув книгу"), а також коментарі до подій.

messages – повідомлення від бібліотекаря до користувача, які можуть містити нагадування, прохання або загальну інформацію.

ratings – опціональна колекція, в якій зберігаються оцінки та відгуки користувачів про прочитані книги.

Цей підхід дозволив не тільки розмежувати логічну структуру даних, але й забезпечити масштабованість і простоту підтримки додатку. Оскільки Firestore підтримує гнучкий доступ до вкладених документів та реального оновлення даних, збережена інформація може бути використана ефективно та без зайвих затримок на різних екранах інтерфейсу.[12,13]

2.5 Забезпечення зберігання даних на пристрої (локально)

Попри використання хмарного бекенду Firebase, у межах розробки мобільного додатку також було реалізовано локальне зберігання даних на пристрої користувача. Це необхідно для підвищення зручності використання, зменшення навантаження на мережу та забезпечення базової функціональності в умовах нестабільного інтернет-з’єднання.

Для реалізації локального зберігання обрано технологію **SharedPreferences**[13], яка надає можливість зберігати прості типи даних (рядки, числа, булеві значення) у вигляді пар "ключ–значення". У додатку ця функція використовується для таких задач:

- Збереження облікових даних користувача (логін, email, роль) після першого входу, щоб уникнути потреби повторної авторизації під час наступного запуску додатку.
- Зберігання історії бронювань – кешування останніх дій користувача для швидкого відображення навіть при тимчасовій відсутності інтернету.

- Параметри інтерфейсу – наприклад, стан темної/світлої теми або вибраної мови, які зберігаються на пристрої.

Такий підхід має низку переваг: підвищення швидкості завантаження інтерфейсу при наступному вході, часткове функціонування без доступу до інтернету, зменшення кількості запитів до хмарної бази, що оптимізує витрати та покращує продуктивність.

У перспективі, для складніших задач локального кешування може бути реалізовано інтеграцію з базами типу **Hive** або **SQLite**, що дозволить працювати з більшими обсягами структурованої інформації (наприклад, кеш бібліотеки книг, попередньо переглянуті відгуки, чернетки відгуків тощо).

Отже, локальне зберігання даних є доповненням до хмарної інфраструктури та відіграє важливу роль у забезпеченні стійкості та зручності користування додатком.

2.6 UI/UX дизайн і прототипування інтерфейсу

Успішність мобільного додатку значною мірою залежить не лише від його функціональності, а й від зручності, інтуїтивності та привабливості інтерфейсу. Саме тому важливою частиною розробки стала робота над UI/UX-дизайном – зовнішнім виглядом додатку (User Interface) і загальною логікою користування (User Experience).[15]

Під час створення інтерфейсу було дотримано кількох ключових принципів UX-дизайну:

Простота і зрозумілість – кожен екран має чітке призначення, без перевантаження елементами.

Послідовність – всі кнопки, кольори та стилі дотримуються єдиної візуальної системи.

Мінімізація дій користувача – найважливіші функції, як-от бронювання книги чи перегляд історії, доступні в кілька натискань.

Інтерфейс додатку реалізовано за допомогою стандартних компонентів Flutter: AppBar, ListView, Card, TextFormField, BottomNavigationBar тощо[16,17]. Для стилізації використано власну кольорову палітру та систему іконок. Застосунок включає такі основні екрани:

- екран реєстрації/входу
- головна сторінка користувача з доступом до функцій бронювання
- сторінка перегляду історії відвідувань
- окремий екран бібліотекаря з керуванням запитами
- форма додавання та редагування книг

Для підвищення ефективності проєктування перед початком розробки були створені прототипи інтерфейсу за допомогою онлайн-інструменту Figma[18,19]. На цьому етапі було візуалізовано основні сценарії користування додатком.

Прототипування дозволило:

- виявити та усунути недоліки в логіці навігації ще до початку розробки
- погодити дизайн із потенційними користувачами
- забезпечити узгодженість між дизайном і реалізацією

У результаті було створено додаток з приємним візуальним стилем і логічною структурою, що забезпечує позитивний досвід користувача та стимулює до регулярного використання бібліотечних сервісів.

Висновки до розділу 2

У другому розділі було обґрунтовано вибір програмних засобів, платформ та архітектурного підходу, які стали основою для реалізації мобільного додатку. Порівняльний аналіз популярних фреймворків для кросплатформенної розробки підтвердив доцільність використання Flutter завдяки його гнучкості, високій швидкості розробки та широким можливостям візуалізації. Як бекенд-рішення обрано платформу Firebase, що забезпечила готову інфраструктуру для автентифікації, зберігання даних та надсилання повідомлень без необхідності створення окремого сервера.

Розробка архітектури додатку базувалась на клієнт-серверній моделі, що дозволяє забезпечити ефективну взаємодію між користувачем та хмарною базою даних у режимі реального часу. Структура бази даних Firestore чітко розділяє дані за колекціями (користувачі, книги, бронювання, історія відвідувань, повідомлення, запити на повернення), що підвищує масштабованість та зручність обробки даних. Завдяки цьому досягнуто модульного поділу функціональності, що спрощує підтримку, тестування та подальший розвиток системи.

Важливою перевагою архітектурного рішення є підтримка ролей користувачів (читач і бібліотекар), що дозволяє гнучко управляти правами доступу до різного функціоналу. Використання SharedPreferences додатково забезпечує збереження локальних налаштувань та підвищує зручність користування додатком.

Окрему увагу в цьому розділі приділено аспектам безпеки та обмеженням**:** розглянуто можливі ризики, пов’язані зі збереженням персональних даних, а також враховано залежність роботи додатку від наявності стабільного

інтернет-з’єднання. Застосовані засоби захисту Firebase, зокрема правила безпеки бази даних і авторизація через захищений канал, забезпечують надійний рівень конфіденційності користувачів.

Також у розділі окреслено можливості масштабування і розвитку системи в майбутньому: додавання нових функцій, інтеграція з бібліотечними системами (наприклад, ІРБІС), покращення системи аналітики та впровадження нових інструментів комунікації між бібліотекарем і читачем.

Значну увагу приділено UI/UX-дизайну та прототипуванню**:** завдяки застосуванню принципів сучасного дизайну та попередньому проєктуванню інтерфейсу в Figma вдалося досягти високої інтуїтивності взаємодії та позитивного досвіду користувачів. Візуальні рішення та логіка навігації спрямовані на зниження когнітивного навантаження та забезпечення швидкого доступу до ключових функцій.

Таким чином, у цьому розділі сформовано повноцінну технологічну основу для створення функціонального, безпечного, масштабованого та зручного у використанні мобільного додатку для бібліотечної сфери. Вибрані інструменти, архітектурні рішення, а також увага до дизайну та перспектив розвитку повністю відповідають вимогам завдання та сучасним стандартам розробки програмного забезпечення.

РОЗДІЛ ІІІ РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ МОБІЛЬНОГО ДОДАТКУ

3.1 Аналіз вимог до мобільного додатку

Розробка будь-якого програмного забезпечення починається з детального аналізу вимог, що дозволяє визначити функціональність, яку повинен реалізовувати додаток, а також очікування кінцевого користувача. У межах цієї кваліфікаційної роботи було здійснено систематизацію як функціональних, так і нефункціональних вимог до мобільного додатку для бронювання книг та відстеження історії відвідувань бібліотеки.[20,21]

Основними функціональними вимогами стали:

- Можливість реєстрації та авторизації користувачів за допомогою електронної пошти й пароля
- Розподіл користувачів за ролями (звичайний користувач та бібліотекар) із відповідним інтерфейсом і правами доступу[22]
- Перегляд наявних книг у бібліотеці, включаючи назву, автора, жанр, кількість доступних екземплярів, а також обкладинку книги
- Функціонал бронювання книги користувачем та фіксація відповідного запису у Firestore
- Можливість скасування бронювання або надсилання запиту на повернення книги
- Ведення особистої історії бронювань та дій користувача (відвідування бібліотеки, взяття та повернення книг)
- Відображення статистики активності користувача
- Можливість залишити коментар до події (наприклад, короткий відгук або уточнення до дії)
- Повідомлення від бібліотекаря користувачеві про необхідність повернення книги або іншу важливу інформацію
- Відображення підтвердженого статусу повернення бібліотекарем

Нефункціональні вимоги включають:

- Зручний і зрозумілий інтерфейс, що не потребує додаткового навчання користувача
- Адаптивність до різних розмірів екранів Android-пристроїв
- Мінімальна затримка у взаємодії з базою даних
- Надійність збереження інформації в хмарному сховищі Firebase[22]
- Швидке завантаження списку книг та дій користувача
- Безпечне зберігання та передача даних, включаючи шифрування автентифікаційної інформації

Для забезпечення відповідності зазначеним вимогам було обрано такі інструменти, як Flutter та Firebase, що дозволяють поєднати зручність розробки з високою швидкістю виконання та підтримкою мобільних платформ. Кожен із вищенаведених пунктів ліг в основу окремого модуля мобільного додатку, що дозволило структуровано реалізувати його архітектуру.[23]

Завдяки детальному аналізу вимог стало можливим сформувати чітке бачення майбутнього функціоналу застосунку, уникнути зайвої складності та сконцентрувати зусилля на реалізації справді необхідних для користувача можливостей.

3.2 Бронювання книг, зміна статусу, облік історії та статистики

У межах реалізації основного функціоналу мобільного додатку важливе місце посідає механізм бронювання книг користувачами. Основна мета цієї функції полягає в тому, щоб надати зручний інтерфейс для пошуку та резервування книг з бібліотечного фонду, при цьому забезпечуючи актуальність інформації про доступність примірників.

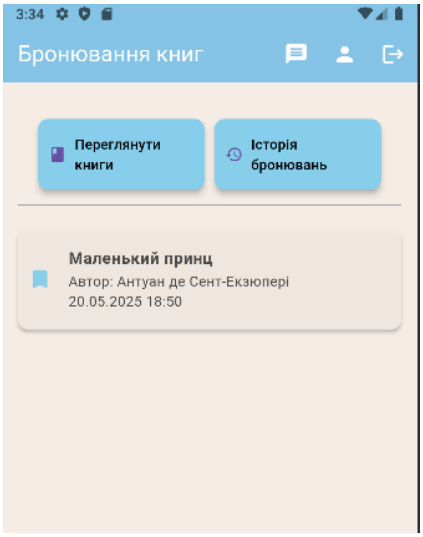

Рисунок 3.1 – Головний екран додатку

Список доступних книг формується з бази даних Firestore. Кожна книга має такі атрибути, як назва, автор, жанр, кількість доступних примірників, короткий опис і, за можливості, обкладинку[24]. Користувач має змогу переглядати цей список, а також здійснювати пошук за назвою або автором (рис.3.2). Кнопка «Забронювати» дозволяє додати книгу до особистого списку бронювань.

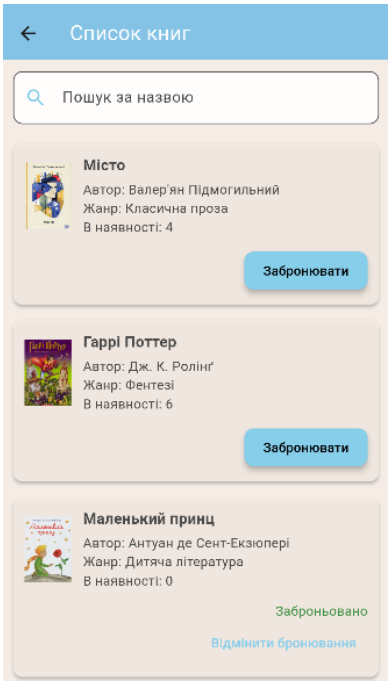

Рисунок 3.2 – Екран зі списком книг

Після успішного бронювання, у Firestore створюється новий документ у колекції bookings, який зберігає інформацію про користувача, назву книги, дату бронювання, статус («заброньовано», «повернено», тощо) та унікальний ідентифікатор запису. Система не дозволяє дублювати бронювання однієї і тієї ж книги тим самим користувачем, а також перевіряє наявність доступних примірників перед бронюванням.

Користувач також може переглядати свою історію бронювань на окремому екрані (рис.3.3). Усі записи структуруються за хронологією, де відображається не тільки дата бронювання, а й статус: активне бронювання, очікування на повернення або вже повернено[25]. Це дозволяє користувачу відстежувати свої дії у додатку та планувати повернення книг вчасно.

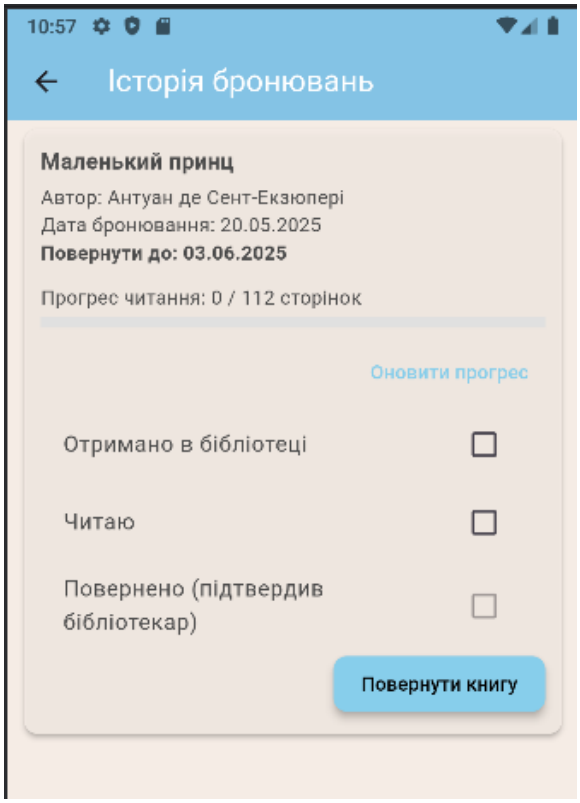

Рисунок 3.3 – Екран історії бронювань користувача

Ще одним важливим компонентом є ведення статистики користувацької активності. На екрані історії передбачено візуальний блок зі зведенням кількості заброньованих книг, повернень та візитів до бібліотеки. Такі дані дозволяють користувачу бачити свій прогрес, а також сприяють формуванню читацької звички.

Усі дії із бронюванням пов’язані також із зменшенням або збільшенням кількості доступних примірників книги. Після бронювання значення зменшується, а після підтвердження повернення бібліотекарем – збільшується. Таким чином, зберігається баланс і актуальність фонду.

Додатково, після повного прочитання книги користувачу пропонується можливість залишити оцінку та відгук. Ця функція дозволяє збирати якісний зворотній зв’язок та створювати рейтинг книг у майбутньому.[26]

Таким чином, система бронювань реалізована з урахуванням зручності для користувача, забезпечення цілісності та актуальності бази даних, а також з метою формування детальної історії взаємодії користувача з бібліотекою. Це робить додаток не лише функціональним, а й корисним для щоденного використання читачами різних категорій.

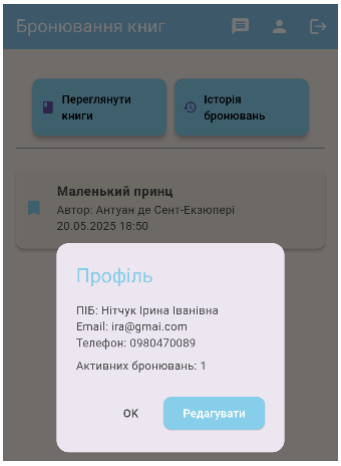

Рисунок 3.4 – Профіль користувача

Для зберігання даних у мобільному додатку було використано хмарну базу даних Firebase Cloud Firestore, яка надає можливість зберігати та синхронізувати дані в режимі реального часу (рис.3.5). Firestore є гнучкою, масштабованою NoSQL базою даних, яка працює за принципом колекцій і документів.

У межах цієї дипломної роботи було створено низку основних колекцій, що відображають логіку взаємодії користувача з бібліотечним середовищем:

1. users

Ця колекція зберігає інформацію про всіх зареєстрованих користувачів. У кожного користувача є унікальний ідентифікатор (UID), який створюється Firebase Authentication під час реєстрації.[27]

Основні поля:

- uid: унікальний ідентифікатор користувача
- email: адреса електронної пошти
- role: роль користувача (user або librarian)
- username: ім’я користувача для відображення в інтерфейсі

2. books

Ця колекція містить усі книги, доступні для перегляду та бронювання.

Основні поля:

- title: назва книги
- author: автор книги
- genre: жанр
- quantity: кількість доступних примірників
- imageUrl: посилання на обкладинку книги

3. bookings

Колекція для зберігання інформації про активні бронювання, здійснені користувачами.

Основні поля:

- userId: UID користувача, що здійснив бронювання
- bookId: ідентифікатор книги
- timestamp: дата й час бронювання
- status: статус бронювання (активне, повернуто)
- dueDate: очікувана дата повернення

4. return_requests

Колекція, яка містить запити користувачів на повернення книг.

Основні поля:

- bookingId: ідентифікатор бронювання
- userId: UID користувача
- requestDate: дата подання запиту
- status: поточний стан запиту (в очікуванні, підтверджено)

5. visits

Колекція, що фіксує дії користувача, пов’язані з відвідуванням бібліотеки або діями над книгами.

Основні поля:

- userId: UID користувача
- action: тип дії (відвідування, взяв книгу, повернув книгу)
- comment: коментар користувача
- timestamp: дата та час дії

6. messages

Ця колекція використовується для зберігання повідомлень, які надсилає бібліотекар окремим користувачам.

Основні поля:

- receiverId: UID отримувача повідомлення
- messageText: текст повідомлення

- timestamp: дата й час надсилання

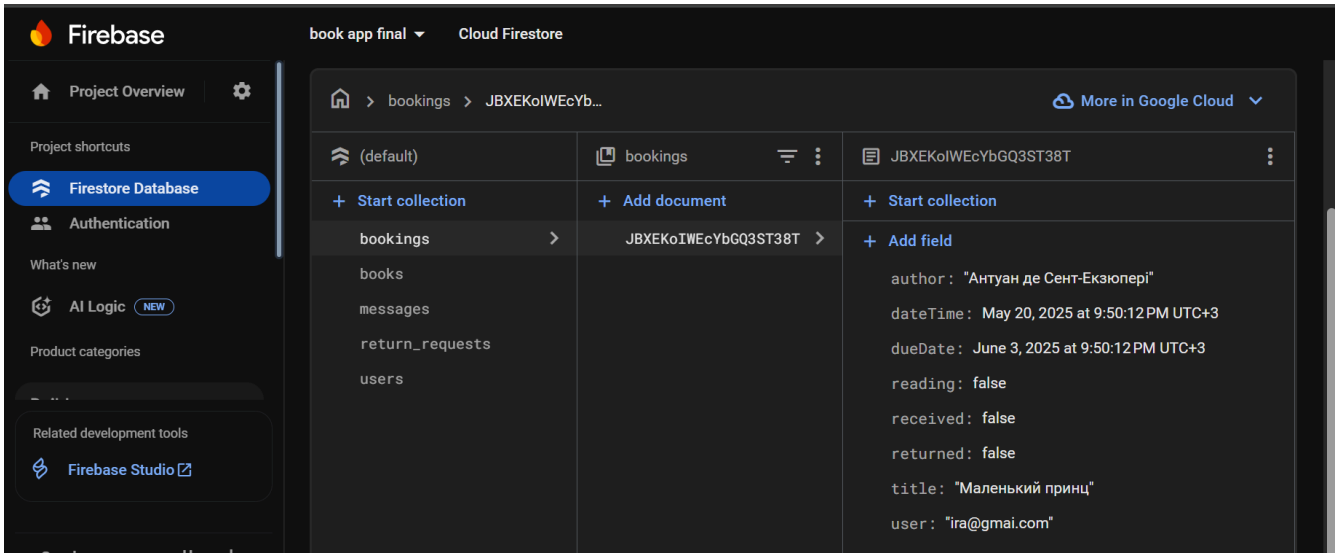

Рисунок 3.5. Структура бази даних Firestore, що використовується для зберігання бронювань та інформації про книги

Ця структура дозволяє зручно зберігати й обробляти інформацію, необхідну для реалізації основного функціоналу додатку. Використання окремих колекцій дозволяє уникати надлишковості даних і забезпечує високу продуктивність, що особливо важливо в умовах обмежених мобільних ресурсів.

3.3 Повідомлення, запити на повернення та панель бібліотекаря

У процесі розробки мобільного додатку було важливо не лише надати базову функціональність для читачів, а й реалізувати можливості ефективного адміністрування книжкового фонду. Саме тому в додаток були включені додаткові модулі, що забезпечують взаємодію між користувачами й бібліотекарем. До таких модулів належать: система повідомлень, функціонал запитів на повернення книг та окрема панель керування для бібліотекаря.

Система повідомлень

Функція повідомлень реалізована як канал комунікації бібліотекаря з читачами[29]. У реальних умовах роботи бібліотеки виникає потреба інформувати користувачів про:

- наближення терміну повернення книги
- прострочення бронювання
- підтвердження або відхилення запиту
- нові надходження або зміни в розкладі

Для цього створено окрему колекцію messages у базі Firestore. Кожне повідомлення містить наступні поля:

- receiverId: унікальний ідентифікатор користувача
- messageText: текст повідомлення
- timestamp: дата та час створення
- read: логічне значення, що вказує, чи було повідомлення прочитано

На стороні користувача реалізовано екран "Повідомлення", який підключається до цієї колекції за допомогою StreamBuilder. Це дає змогу в реальному часі відображати нові повідомлення без необхідності перезавантаження сторінки.

Інтерфейс дозволяє зручно переглядати історію повідомлень(рис.3.6), а у випадку надходження нових – користувач отримує візуальне сповіщення.

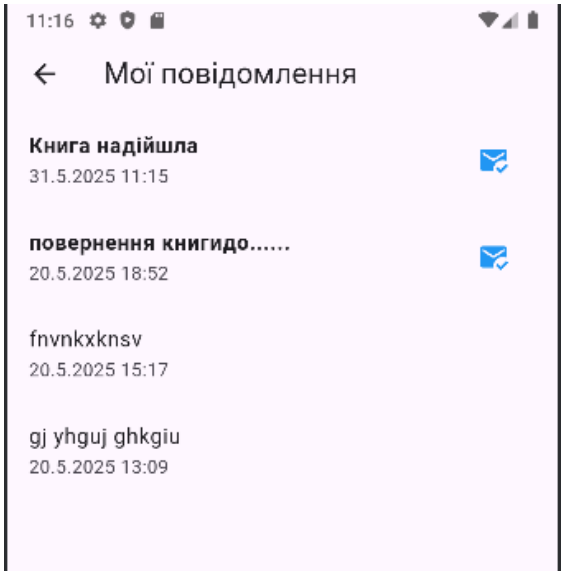

Рисунок 3.6 – Екран перегляду повідомлень

Запити на повернення книг

Щоб автоматизувати процес повернення літератури, додано можливість самостійно створювати запити на повернення. Це особливо важливо в умовах, коли користувач не має змоги особисто з’явитись у бібліотеку або хоче заздалегідь повідомити про повернення книги.[30]

Функціональність реалізовано наступним чином:

У розділі «Історія бронювань» біля кожного запису є кнопка «Повернути книгу»;

При натисканні на неї створюється запис у колекції return_requests, що містить:

- userId
- bookId
- bookingId
- requestDate
- status (очікує, підтверджено, відхилено)

- comment (необов’язковий)

Це дозволяє бібліотекарю бачити всі активні запити у своїй панелі, швидко приймати рішення та оновлювати статус бронювання. Користувач, своєю чергою, бачить зміни статусу запиту в реальному часі.

Панель бібліотекаря

Панель бібліотекаря – спеціальний розділ додатку, доступний лише для користувачів із відповідною роллю (role: librarian)(рис.3.7). Вона реалізована як окремий інтерфейс із кількома вкладками, які забезпечують:

Перегляд бронювань: дозволяє бібліотекарю бачити, хто і яку книгу забронював, дату бронювання, статус виконання. Також можливо змінювати статус вручну (наприклад, «отримано», «завершено»).

Обробку запитів на повернення: усі запити користувачів зібрані в окремому списку. Після перегляду інформації та повернення книги в бібліотеку, бібліотекар може натиснути «Підтвердити», тоді відповідний статус оновлюється і користувач отримує підтвердження.

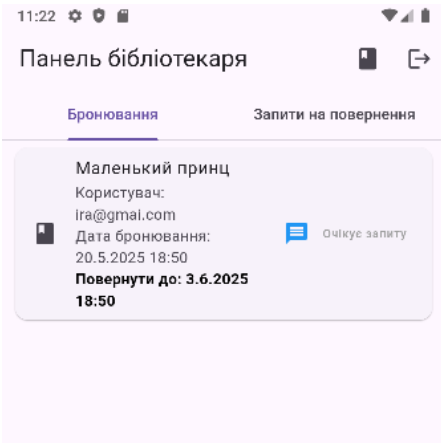

Рисунок 3.7 – Панель бібліотекаря

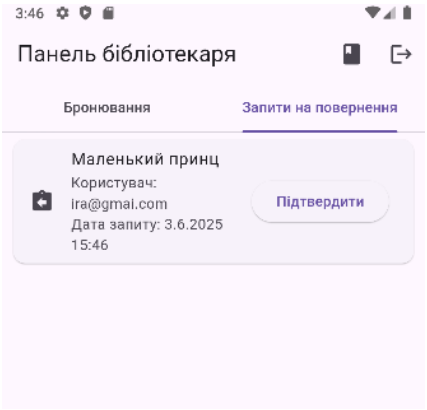

Рисунок 3.8 – Панель запитів на повернення

Редагування книжок: доступна форма для створення, редагування(рис.3.10) або видалення записів про книги у Firestore (назва, автор, жанр, кількість).(рис.3.9)

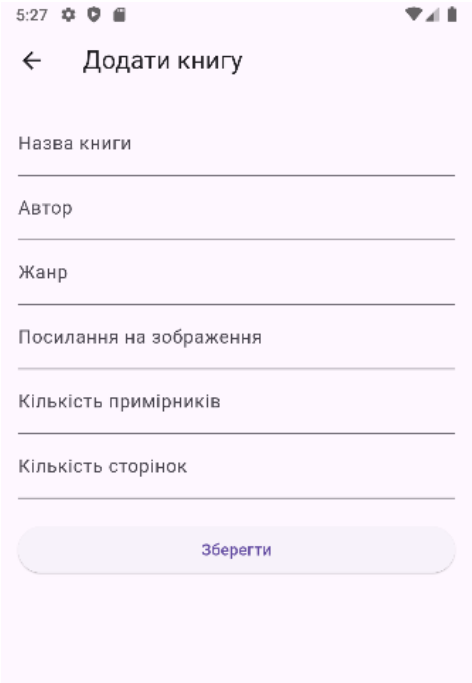

Рисунок 3.9 – Інтерфейс додавання нової книги в мобільному додатку (панель бібліотекаря)

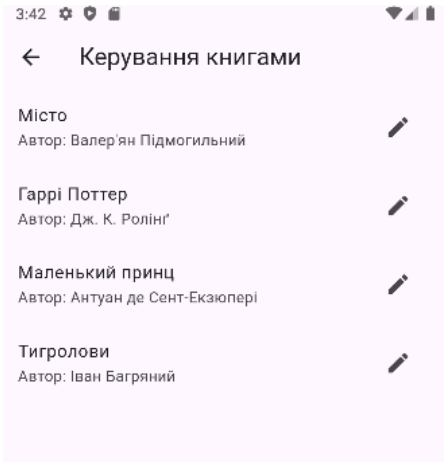

Рисунок 3.10 – Інтерфейс керування книг (панель бібліотекаря)

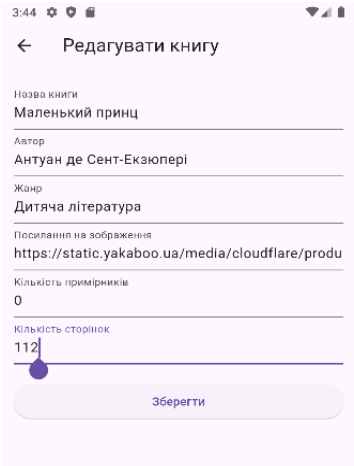

Рисунок 3.11 – Інтерфейс редагування книг (панель бібліотекаря)

Надсилання повідомлень: бібліотекар може обрати користувача зі списку і відправити йому індивідуальне повідомлення.

Взаємодія компонентів

Загальна архітектура взаємодії реалізована через Firebase Auth та Firestore. Після автентифікації користувача відбувається перевірка ролі, і додаток відображає відповідний інтерфейс: стандартний – для звичайного користувача, або розширений – для бібліотекаря. Уся інформація оновлюється в реальному часі, що забезпечує ефективне керування та зменшує затрати на комунікацію.

Таким чином, реалізовані функції значно підвищують зручність використання додатку, автоматизують рутинні бібліотечні процеси та дозволяють бібліотекарю ефективно виконувати свою роботу навіть у цифровому середовищі. Система є масштабованою та може бути адаптована до потреб різних типів бібліотек.

3.4 Тестування додатку та приклади його застосування

Після реалізації основної функціональності мобільного додатку було проведено комплексне тестування для перевірки працездатності, зручності користування та відповідності поставленим вимогам. Метою тестування було виявлення помилок, забезпечення стабільної роботи додатку на цільовій платформі (Android) та підтвердження правильності обробки даних через інтеграцію з Firebase.

Підходи до тестування

Було використано переважно ручне тестування (manual testing), оскільки додаток перебував у стадії активної розробки, а функціональність постійно змінювалася. Для цього застосовувалося середовище Android Studio, яке надає вбудований емулятор для моделювання поведінки реального пристрою[31,32].

Тестування проводилося за наступними напрямами:

- **Функціональне тестування** – перевірка відповідності кожної функції її очікуваній поведінці. Наприклад, успішна реєстрація нового користувача, авторизація з правильними та неправильними даними, збереження облікових даних у SharedPreferences, робота з бронюванням, відображення історії відвідувань.
- **Тестування взаємодії з базою даних** – перевірка запису та читання даних з Firebase Firestore. Окрема увага приділялася відповідності ідентифікаторів користувачів, запитів на повернення та повідомлень.
- **Перевірка коректності UI** – оцінка відображення інтерфейсу на різних розмірах екранів, правильності перекладу українською мовою, логіки навігації між екранами.

- **Перевірка умовних сценаріїв** – наприклад: спроба забронювати вже заброньовану книгу, повернення книги без підтвердження бібліотекарем, відображення повідомлень після відправки.

Результати тестування

Усі основні функції додатку були протестовані, і зафіксовано їх стабільну роботу. У процесі тестування виявлялися незначні логічні помилки:

- некоректне очищення полів після виходу з облікового запису;
- неактуальне відображення статусу книги після підтвердження повернення;
- подвоєння записів в історії у разі повторного натискання кнопки.

Ці помилки були усунені, після чого застосунок показав стабільну роботу як на емуляторі, так і на реальному Android-пристрої.

Також тестування показало, що додаток можна масштабувати, додавати нові функції, зберігаючи стабільність архітектури.

Таким чином, завершальний етап реалізації продемонстрував, що мобільний додаток повністю відповідає заявленій функціональності. Він може бути використаний у реальному середовищі бібліотеки, забезпечуючи ефективну взаємодію між працівниками та читачами, автоматизуючи частину рутинних процесів і надаючи користувачам зручний цифровий сервіс.

3.5 Обмеження використання мобільного додатку

Створений мобільний додаток має низку технічних та організаційних обмежень, які слід враховувати під час його використання та подальшого вдосконалення:

Технічні обмеження:

Залежність від інтернет-з’єднання:

Додаток використовує хмарні сервіси Firebase (Auth, Firestore), тому функціонування неможливе без стабільного інтернету.

Наприклад: користувач не зможе переглядати список книг або бронювати їх у режимі офлайн.

Для вирішення цієї проблеми в майбутньому можна реалізувати локальне кешування даних за допомогою Hive або SQLite.

Обмеження безкоштовного тарифу Firebase:

Безкоштовний план Firestore має ліміти: 50 тисяч операцій запису/дня та 20 тисяч одночасних з’єднань. Для великої бібліотеки з сотнями активних користувачів це може стати проблемою.

Рішення: Перехід на платний тариф або оптимізація запитів (наприклад, пагінація даних).

Підтримка платформ:

Додаток розроблено для Android, але не тестовано на iOS. Можливі несподівані помилки через особливості рендерингу Flutter на iOS.

Організаційні обмеження:

Необхідність навчання персоналу:

Бібліотекарям потрібно освоїти адмін-панель додатку (додавання книг, підтвердження повернень). Для цього знадобиться інструкція або тренінг.[33]

Відсутність інтеграції з існуючими бібліотечними системами:

Наразі додаток не синхронізується з такими системами, як ІРБІС або Koha. Це ускладнює масштабування для великих бібліотек.

Юридичні аспекти:

Обробка персональних даних користувачів (емейли, історія бронювань) потребує відповідності ЗУ «Про захист персональних даних». [34]

Наприклад:

Необхідно додати підтвердження згоди на обробку даних під час реєстрації.

Реалізувати механізм автоматичного видалення даних за бажанням користувача.

Перспективи подолання обмежень:

Додати офлайн-режим з синхронізацією даних при відновленні з’єднання.

Розробити REST API для інтеграції з іншими бібліотечними системами.

Оптимізувати Firestore-запити (наприклад, використовувати where з індексами).

3.6 Можливості розширення

Розроблений мобільний додаток для бронювання книг та відстеження історії відвідувань має значний потенціал для подальшого розвитку. У цьому підрозділі розглянуто можливі напрямки розширення функціоналу, які планується реалізувати в майбутньому, а також проаналізовано можливість інтеграції з існуючими бібліотечними системами, такими як ІРБІС.[35]

Плановані функції

Для підвищення зручності та функціональності додатку планується додати такі можливості:

Фільтри та сортування: на екрані перегляду книг реалізувати фільтри за жанром, автором і роком видання, а також сортування історії бронювань за датою чи статусом. Це дозволить користувачам швидше знаходити потрібну літературу та аналізувати свою активність.

Рекомендації на основі машинного навчання: розробити алгоритм рекомендацій книг, який аналізуватиме історію бронювань і вподобання користувача (жанри, автори), пропонуючи літературу, що може його зацікавити. Для цього можна інтегрувати Firebase ML Kit або сторонні бібліотеки машинного навчання.[36]

Пошук бібліотек поруч через геопозицію: Реалізувати функцію визначення місцезнаходження користувача за допомогою GPS і відображення найближчих бібліотек на карті. Це дозволить користувачам швидко знаходити бібліотеки в радіусі, наприклад, 5 км, із зазначенням їхньої адреси, робочого часу та доступності книг. Для реалізації планується інтеграція з Google Maps API, яка забезпечить точне відображення географічних даних і маршрутів до бібліотеки. Користувач зможе обрати бібліотеку, переглянути її каталог і забронювати книгу безпосередньо з карти.

Масові повідомлення для бібліотекарів: додати функцію, що дозволить бібліотекарям надсилати повідомлення групам користувачів (наприклад, усім, хто прострочив повернення книг). Це спростить комунікацію в великих бібліотеках із тисячами користувачів.

Оцінки та відгуки з рейтингом книг: розширити функціонал відгуків, додавши можливість виставляти рейтинг книг (від 1 до 5 зірок). На основі цих

даних можна сформувати рейтинг найпопулярніших книг у бібліотеці, що стане додатковим інструментом для користувачів під час вибору літератури.

Додаткові ідеї для розширення:

Соціальні функції: впровадити можливість ділитися прочитаними книгами з друзями через соціальні мережі (наприклад, X або Facebook) або створювати читацькі спільноти всередині додатку для обміну відгуками та рекомендаціями.

Мультимовний інтерфейс: додати підтримку кількох мов для залучення іноземних студентів чи туристів, які користуються бібліотеками. Це також підвищить доступність додатку для людей із різними мовними потребами.

QR-коди для бронювання: реалізувати функцію сканування QR-кодів, розміщених на книгах у бібліотеці, для миттєвого бронювання чи перегляду інформації про книгу без ручного пошуку.

Статистика для бібліотек: додати аналітичний модуль для бібліотекарів, який відображатиме статистику використання додатку (кількість бронювань, популярність книг, активність користувачів за місяцями), що допоможе оптимізувати закупівлю літератури.[37]

Інтеграція з існуючими бібліотечними системами (ІРБІС)

Одним із перспективних напрямків розвитку є інтеграція додатку з існуючими бібліотечними системами, зокрема ІРБІС, яка широко використовується в українських бібліотеках для автоматизації обліку книг, користувачів і транзакцій. Наразі додаток працює з Firebase Firestore, що вимагає перенесення даних із локальних систем у хмарну базу. Однак інтеграція з ІРБІС дозволить уникнути дублювання даних і автоматизувати облік.

Для реалізації інтеграції необхідно:

1. Розробити API для синхронізації даних. ІРБІС підтримує експорт даних у різних форматах (наприклад, MARC), тому можна створити проміжний сервер (наприклад, на базі Node.js або Python), який конвертуватиме дані з ІРБІС у формат JSON для Firestore. Це дозволить синхронізувати каталоги книг, інформацію про користувачів і статуси бронювань.
2. Автоматизувати оновлення даних. Налаштувати періодичну синхронізацію (наприклад, раз на добу), щоб зміни в ІРБІС (додавання нових книг, оновлення статусів) автоматично відображалися в додатку, і навпаки – дії в додатку (бронювання, повернення) оновлювали базу ІРБІС.
3. Забезпечити безпеку передачі даних. Використовувати захищені протоколи (HTTPS) і токени автентифікації для передачі даних між ІРБІС і Firebase, щоб уникнути несанкціонованого доступу.

Така інтеграція дозволить зробити додаток більш універсальним для українських бібліотек, зменшить навантаження на бібліотекарів і підвищить точність даних. Проте її реалізація потребує додаткових ресурсів, зокрема залучення спеціалістів із інтеграції та підтримки серверної інфраструктури.

Розширення платформ

Наразі додаток працює лише на Android, але завдяки кросплатформенній природі Flutter його можна легко адаптувати для iOS. Це розширить аудиторію користувачів, адже значна кількість читачів використовує пристрої Apple. Крім того, можлива розробка вебверсії додатку, що дозволить користувачам бронювати книги через браузер, не встановлюючи додаток. Flutter підтримує створення вебдодатків із єдиною кодовою базою, що спрощує цей процес.

Висновки до розділу 3

У цьому розділі було детально розглянуто реалізацію мобільного додатку для бронювання книг та відстеження історії відвідувань бібліотеки. Було проаналізовано вимоги до системи, розглянуто архітектуру її функціональних модулів та описано основні компоненти, реалізовані засобами Flutter і Firebase.

Особливу увагу приділено забезпеченню зручності для користувачів: впроваджено розділення ролей, інтерфейс для бронювання, повернення та оцінювання книг, статистику активності, історію відвідувань і систему повідомлень. Для бібліотекаря створено окрему панель керування зі зручними інструментами адміністрування.

Було проведено ручне тестування на емуляторі та реальному пристрої, виявлено та усунуто помилки, що підтвердило стабільну роботу додатку. Окремо проаналізовано існуючі технічні та організаційні обмеження, зокрема залежність від інтернет-з’єднання, відсутність iOS-версії та необхідність навчання персоналу.

Запропоновано низку перспектив для розширення функціональності, включаючи фільтри, геолокацію, соціальні функції та інтеграцію з системами типу ІРБІС, що відкриває нові можливості для масштабування й подальшого розвитку застосунку.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було здійснено повний цикл розробки мобільного додатку для бронювання книг та відстеження історії відвідувань бібліотеки. Робота включала етапи аналізу предметної області, визначення вимог до функціональності, вибір інструментів та технологій, архітектурне проєктування, реалізацію, тестування та аналіз результатів.

Проведений огляд сучасних рішень у сфері цифровізації бібліотечних процесів засвідчив актуальність розробки мобільного інструменту, який би забезпечував не лише бронювання літератури, а й надавав зручні механізми фіксації та аналізу користувацької активності. Було встановлено, що більшість існуючих рішень є складними у використанні, недостатньо адаптованими під потреби локальних бібліотек або потребують значних ресурсів для впровадження.

Для реалізації дипломної роботи було обрано сучасні програмні засоби: Flutter – як кросплатформенний фреймворк для розробки зручного мобільного інтерфейсу, та Firebase – як надійну хмарну платформу для зберігання даних, автентифікації користувачів, організації обміну повідомленнями та інтеграції з хмарною базою даних. Такий вибір дозволив не лише скоротити час на розробку, а й забезпечити гнучкість, масштабованість і доступність програми для подальшого впровадження у реальні умови.

У додатку реалізовано низку ключових функцій: реєстрація та авторизація користувачів з різними ролями, бронювання книг із перевіркою доступності, формування історії замовлень, система обліку відвідувань, можливість повернення книг та надсилання повідомлень від бібліотекаря. Додатково розроблено інтерфейс адміністратора (бібліотекаря) для керування контентом і взаємодії з користувачами.

Під час тестування додатку в середовищі Android Studio було підтверджено стабільність роботи основних функцій, відповідність інтерфейсу принципам зручності та інтуїтивності, а також коректність збереження і обробки даних через Firebase. Окрему увагу приділено перевірці сценаріїв використання додатку з боку як звичайного користувача, так і бібліотекаря.

Отже, поставлені завдання було виконано в повному обсязі. Створений мобільний застосунок є прикладом ефективного рішення для автоматизації частини процесів у бібліотеці та може бути адаптований для реального впровадження у заклади освіти, культурні установи або комерційні бібліотеки. В подальшому можливе розширення функціоналу, інтеграція зі сторонніми сервісами, а також підтримка нових мов інтерфейсу для ширшого охоплення користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Statista. Number of smartphone users worldwide 2016 to 2024 [Електронний ресурс]. – Режим доступу: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/
2. DataReportal. Digital 2023: Ukraine [Електронний ресурс]. – Режим доступу: https://datareportal.com/reports/digital-2023-ukraine
3. В Машкіна, ВО Абрамов, ТІ Носенко, ЄВ Іваніченко.Навчально-методичний посібник до виконання і захисту кваліфікаційної роботи бакалавра спеціальностей 122 Комп'ютерні науки для здобувачів вищої освіти денної форми навчання, Київський столичний університет імені Бориса Грінченка. Київ, 2024.- 43 с.
4. Теоретичні та практичні аспекти використання математичних методів та інформаційних технологій в освіті й науці:моногр. / за заг. ред. О. Литвин. — К.: Київ. ун-т ім. Б. Грінченка,2021. — 332 с.
5. Яскевич, Владислав Олександрович (2023) *JavaScriptбібліотека React Навчальний посібник для студентів спеціальності Комп'ютерні науки* Київський університет імені Бориса Грінченка, Україна, Київ. https://elibrary.kubg.edu.ua/id/eprint/48587/
6. Нітчук І.І. Розробка мобільного додатку для бронювання книг та відстеження історії відвідувань бібліотеки / І.І. Нітчук // Збірник тез конференції «ІТ-25». – Київ: Київський університет імені Бориса Грінченка, 2025. – [Електронний ресурс]. – Режим доступу: https://zcit.kubg.edu.ua/index.php/journal/issue/view/13/23

7. Kumbhar, R., & Mane, A. (2022). Mobile Applications for Library Services: A Review. *Library Philosophy and Practice (e-journal)*. https://digitalcommons.unl.edu/libphilprac/6413
8. Барабаш, Н. А. (2020). Трансформація бібліотек в умовах цифровізації суспільства. *Наукові праці НБУ ім. В. І. Вернадського*. https://irbis-nbuv.gov.ua/E_LIB/PDF/er-0004979.pdf
9. Костенко, І. С. (2022). Мобільні додатки у публічних бібліотеках України: перспективи впровадження. *Бібліотечний вісник*.
10. Кормен Т., Лейзерсон Ч. Алгоритми: побудова та аналіз. 2-ге вид. Львів : Видавництво Львівської політехніки, 2020. 1296 с.
11. Кравчук Л. М. Розробка мобільних додатків: практичний посібник. Київ : Техніка, 2021. 298 с.
12. Коваленко О. В. Основи розробки мобільних додатків: теорія та практика. Київ : Наукова думка, 2020. 320 с.
13. Шевчук Д. М. Автоматизація бібліотечних процесів: сучасні підходи та виклики. Вісник Київського національного університету імені Тараса Шевченка. Серія «Інформатика», 2023. Т. 15, № 2. С. 45–52.
14. Георгіца, М. М., & Балан, В. В. (2020). Архітектурні особливості мобільних додатків: клієнт-серверна модель. *Інформаційні технології і засоби навчання*.https://repository.kpi.kharkov.ua/items/b08ebef7-8f89-4010-860a-6137fd4e1897
15. Ковальчук А. В. Використання хмарних технологій у бібліотечних системах. Інформаційні технології в освіті: матеріали Міжнар. наук.-практ. конф., м. Одеса, 15–16 трав. 2024 р. / Одеський національний університет імені І. І. Мечникова. Одеса, 2024. С. 78–82.
16. Шевченко, І. А. (2023). Реалізація системи доступу за ролями у Flutter-додатках. *Фахові комп'ютерні науки*.

17. Cooper, A., Reimann, R., Cronin, D., Noessel, C., Csizmadi, J., & LeMoine, D. (2014). *About face: The essentials of interaction design* (4th ed.). Wiley.
18. Huddleston, C. (2021). *Flutter for beginners: An introductory guide to building cross-platform mobile applications with Flutter and Dart 2*/
19. Соколова Н. В. Автоматизація бібліотечних систем на основі веб-технологій: дис. … канд. техн. наук: 05.13.06. Київ, 2023. 210 с.
20. Гуменюк, М. В. (2021). Використання Figma у процесі створення навчальних застосунків. *Комп'ютер у школі та сім'ї*.
21. Сидоров М. О. Інтерфейси користувача: дизайн та розробка. Київ : Наукова думка, 2023. 224 с.
22. Кнут Д. Е. Мистецтво програмування. Т. 1: Основні алгоритми. 3-тє вид. Київ : Вид. дім "КМ", 2019. 784 с.
23. Козубенко, О. Ю. (2023). Забезпечення безпеки персональних даних у Flutter-додатках. *Фахові комп'ютерні науки*.
24. Hussain, W., & Mkpojiogu, E. O. C. (2016). Requirements engineering for mobile applications. *Journal of Theoretical and Applied Information Technology*.
25. Мельник, О. А. (2022). Організація баз даних для мобільних застосунків за допомогою Firestore. *Інформаційні технології в освіті*.
26. Козубенко, О. Ю. (2023). Практика реалізації журналу подій та історії користувача у Flutter-додатках. *Фахові комп'ютерні науки*.
27. Shah, N. (2019). Cloud Firestore vs Realtime Database: A comparative analysis. *Journal of Computer Applications*.
28. Chauhan, P., & Verma, S. (2021). User feedback collection and analysis in mobile applications using Firestore. *International Research Journal of Engineering and Technology (IRJET)*.
29. Huddleston, C. (2021). *Flutter for beginners*.

30. Шевченко, І. А. (2023). Реалізація панелі адміністратора у Flutter-додатках з Firebase. *Фахові комп'ютерні науки*.
31. Соловей, А. П. (2022). Автоматизація запитів та керування статусами у мобільних застосунках. *Інформаційні технології в освіті*.
32. Кузьменко, І. В. (2023). Методика тестування мобільних застосунків на Android-платформі з використанням Flutter. *Фахові комп'ютерні науки*.
33. Гребенюк, Н. С. (2022). Тестування як завершальний етап розробки мобільних додатків: практичні аспекти. *Інформаційні технології в освіті*.
34. Барабаш, Н. А. (2021). Виклики цифрової трансформації бібліотек: кадровий і технологічний аспекти. *Наукові праці НБУ ім. В. І. Вернадського*.
35. Верховна Рада України. (2010). *Закон України "Про захист персональних даних"*. https://zakon.rada.gov.ua/laws/show/2297-17
36. Павленко Т. Г. Автоматизація бібліотечних процесів: технології та рішення. Чернівці : Рута, 2020. 189 с.
37. Яковенко П. Д. Сучасні технології веб-розробки. Вінниця : Нова книга, 2021. 287 с.
38. Василенко, Л. І. (2022). Візуалізація читацької активності в цифрових бібліотечних сервісах. *Бібліотечний вісник*.