МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ ВЕДЕННЯ НОТАТОК З ПІДТРИМКОЮ ХМАРНОГО ЗБЕРІГАННЯ»**

Виконав: студент 4 курсу, групи 6.1212-пі-2
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
М.Е. Городничий
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.т.н. Мухін В.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри комп'ютерних наук, доцент, к.т.н. Матвіїшина Н.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2025 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Городничому Максиму Едуардовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка мобільного застосунку для ведення нотаток з підтримкою хмарного зберігання

керівник роботи Мухін Віталій Вікторович, к.т.н., доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 02 » грудня 2025 року № 1741-с

2. Строк подання студентом роботи 01.06.2026 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналіз предметної області та існуючих рішень.
2. Проєктування мобільного застосунку.
3. Реалізація застосунку.
4. Тестування та аналіз роботи.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою доповіді

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 09.12.2025 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	19.12.2025	
2.	Збір вихідних даних.	15.01.2026	
3.	Обробка методичних та теоретичних	10.02.2026	
	джерел.		
4.	Розробка першого та другого розділу.	20.03.2026	
5.	Розробка третього та четвертого розділу.	18.05.2026	
6.	Оформлення та нормоконтроль	25.05.2026	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	17.06.2026	

Студент _____ М.Е. Городничий
(підпис) (ініціали та прізвище)

Керівник роботи _____ В.В. Мухін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота на тему: «Розробка мобільного застосунку для ведення нотаток з підтримкою хмарного зберігання»: 57 с., 25 рис., 5 табл., 20 джерел, 1 додаток.

АНАЛІТИКА, БАЗА ДАНИХ, ЗБЕРІГАННЯ, КОРИСТУВАЧ, НОТАТКИ, СИНХРОНІЗАЦІЯ, ХМАРНЕ ЗБЕРІГАННЯ, FIREBASE, KANBAN, REACT NATIVE.

Об’єкт дослідження – процес організації, зберігання та обробки користувацьких нотаток у мобільних застосунках.

Мета роботи – аналіз предметної області, методи проєктування інформаційних систем, моделювання структури даних, використання клієнт-серверної архітектури та сучасних засобів розробки мобільних застосунків.

Методи дослідження: аналіз предметної області, методи проєктування інформаційних систем, моделювання даних, клієнт-серверну архітектуру та сучасні інструменти мобільної розробки.

У роботі розглянуто підходи до створення мобільних застосунків для роботи з нотатками, проаналізовано існуючі рішення та визначено їх обмеження. Запропоновано власний підхід до реалізації застосунку, який поєднує класичні можливості ведення нотаток із додатковими функціями планування та аналітики. У результаті розроблено мобільний застосунок, що забезпечує роботу з нотатками у вигляді списку, календаря та Kanban-дошки, а також реалізує модуль аналітики для оцінки активності користувача. Дані зберігаються у хмарному середовищі, що дозволяє забезпечити їх доступність з різних пристроїв та автоматичну синхронізацію в реальному часі. Практичне значення отриманих результатів полягає у можливості використання розробленого застосунку для організації особистих записів, планування задач та підвищення ефективності роботи користувача.

SUMMARY

Bachelor’s qualifying paper “Development of a Mobile Note-Taking Application with Cloud Storage Support”: 57 pages, 25 figures, 5 tables, 20 references, 1 supplement.

ANALYTICS, CLOUD STORAGE, DATABASE, FIREBASE, KANBAN, MOBILE APPLICATION, NOTES, REACT NATIVE, SYNCHRONIZATION, USER.

The object of the study is the process of organizing, storing, and processing user notes in mobile applications.

The aim of the study is to develop a mobile application that provides creation, editing, storage, and synchronization of notes using cloud technologies, as well as additional tools for their organization and analysis.

The methods of research include analysis of the subject area, information system design methods, data modeling, client-server architecture, and modern mobile development tools.

The paper examines approaches to building note-taking mobile applications, analyzes existing solutions, and identifies their limitations. A custom approach to application development is proposed, combining traditional note-taking functionality with planning and analytics features. As a result, a mobile application was developed that supports note management in the form of a list, calendar, and Kanban board, and includes an analytics module for evaluating user activity. Data is stored in a cloud environment, ensuring availability across devices and real-time synchronization. The practical value of the results lies in the possibility of using the developed application for organizing personal notes, task planning, and improving user productivity.

ЗМІСТ

Завдання на кваліфікаційну роботу ……..…….2
Реферат 4
Summary 5
Вступ..... 8
1 Аналіз предметної області та існуючих рішень 10
1.1 Аналіз предметної області 10
1.2 Огляд існуючих застосунків для ведення нотаток 11
1.3 Порівняльний аналіз аналогів..... 13
1.4 Визначення проблем та обґрунтування розробки 15
1.5 Висновки до розділу 1 16
2 Проєктування мобільного застосунку..... 18
2.1 Загальна архітектура системи..... 18
2.2 Вибір технологій та обґрунтування 19
2.3 Проєктування бази даних..... 21
2.4 Опис логіки роботи системи 24
2.5 Алгоритми основних процесів..... 27
2.5.1 Алгоритм створення нотатки..... 27
2.5.2 Алгоритм зміни статусу нотатки (Kanban) 28
2.5.3 Алгоритм синхронізації даних 28
2.6 Висновки до розділу 2 29
3 Реалізація застосунку..... 31
3.1 Реалізація авторизації..... 31
3.2 Реалізація роботи з нотатками..... 33
3.3 Реалізація Kanban-дошки 36
3.4 Реалізація аналітики..... 39
3.5 Інтерфейс користувача 41
3.6 Висновки до розділу 3 43

4 Тестування та аналіз роботи 45
4.1 Методика тестування 45
4.2 Функціональне тестування 45
4.3 Аналіз результатів 48
4.4 Оцінка ефективності 48
4.5 Висновки до розділу 4 49
Висновки 51
Перелік посилань 53
Додаток А Графічні матеріали 55
А.1 Схема загальної архітектури мобільного застосунку 55
А.2 Структура бази даних у середовищі Firebase 55
А.3 Загальний процес роботи системи 56
А.4 Загальну схему алгоритму створення нотатки 57

ВСТУП

У сучасних умовах цифровізації значна частина повсякденної діяльності людини пов’язана з обробкою інформації. Нотатки залишаються одним із найпростіших і водночас найбільш універсальних способів фіксації ідей, планування завдань та організації особистої інформації. Особливої актуальності набувають мобільні застосунки, які дозволяють виконувати ці дії швидко та незалежно від місця перебування користувача [3].

Разом із тим, традиційні підходи до ведення нотаток у вигляді локальних записів або простих списків мають низку обмежень. До них можна віднести відсутність синхронізації між пристроями, обмежені можливості структурування інформації, а також недостатній рівень інтерактивності при роботі з даними. У результаті цього користувач змушений використовувати декілька інструментів одночасно, що знижує ефективність роботи з інформацією [4].

Сучасні інформаційні системи орієнтуються на інтеграцію декількох підходів до організації даних. Зокрема, поєднання списків, календарного представлення та візуального планування у вигляді Kanban-дошок дозволяє значно покращити зручність взаємодії користувача із системою. Додатково використання аналітичних інструментів дає змогу оцінювати активність користувача та ефективність виконання завдань [13].

Ще одним важливим аспектом є використання хмарних технологій. Застосування хмарного зберігання забезпечує доступ до даних з різних пристроїв, підвищує їх надійність та дозволяє реалізувати синхронізацію в реальному часі. Це відповідає сучасним вимогам до мобільних застосунків і є одним із ключових факторів їх популярності [16].

Таким чином, виникає необхідність у розробці мобільного застосунку, який поєднує базові можливості ведення нотаток із сучасними інструментами організації, планування та аналізу даних. Такий підхід дозволяє створити більш гнучке та зручне середовище для роботи користувача з інформацією.

Метою даної роботи є розробка мобільного застосунку для ведення нотаток з підтримкою хмарного зберігання, який забезпечує створення, редагування, зберігання та синхронізацію даних, а також надає розширені можливості їх організації та аналізу.

Для досягнення поставленої мети необхідно вирішити такі **завдання**:

– проаналізувати предметну область та існуючі рішення у сфері мобільних застосунків для нотаток;
– визначити основні вимоги до функціональності системи;
– спроєктувати архітектуру застосунку та структуру бази даних;
– реалізувати основні модулі системи, включаючи роботу з нотатками, Kanban-дошку та аналітику;
– провести тестування розробленого застосунку та оцінити результати його роботи.

Об’єктом дослідження є процес організації та обробки нотаток у мобільних інформаційних системах.

Предметом дослідження є методи та засоби розробки мобільних застосунків із використанням хмарних технологій.

Практичне значення отриманих результатів полягає у створенні мобільного застосунку, який може бути використаний для ведення особистих нотаток, планування завдань та підвищення ефективності роботи користувача.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ

1.1 Аналіз предметної області

У сучасному цифровому середовищі обсяг інформації, з якою щоденно працює користувач, постійно зростає. Це стосується як особистих записів, так і робочих задач, ідей, планів або нагадувань. У таких умовах виникає потреба у зручних інструментах для фіксації, зберігання та швидкого доступу до інформації. Одним із базових рішень у цій сфері є застосунки для ведення нотаток, які дозволяють структурувати дані та спрощують їх подальше використання [3].

Предметна область ведення нотаток охоплює процеси створення, редагування, організації та зберігання інформації у зручному для користувача вигляді. Ключовим аспектом є не лише можливість запису тексту, а й забезпечення ефективної навігації, пошуку та класифікації даних. У сучасних умовах такі системи поступово еволюціонують від простих текстових редакторів до повноцінних інформаційних систем із розширеним функціоналом [4].

З розвитком мобільних технологій значна частина таких рішень переноситься на мобільні пристрої. Це дозволяє користувачам працювати з нотатками у будь-який час і незалежно від місця перебування. Мобільні застосунки забезпечують швидкий доступ до даних, інтеграцію з іншими сервісами та зручний інтерфейс, що орієнтований на повсякденне використання [8].

Важливим етапом розвитку систем ведення нотаток стало впровадження хмарних технологій. Використання хмарного зберігання дозволяє вирішити проблему доступності даних, оскільки інформація більше не прив’язана до одного пристрою. Крім того, хмарні сервіси забезпечують синхронізацію в реальному часі, що дозволяє миттєво відображати зміни незалежно від платформи або пристрою користувача [19].

Окремої уваги заслуговують підходи до організації нотаток. Традиційно інформація зберігалася у вигляді списків, однак з часом з’явилися альтернативні способи представлення даних. Зокрема, календарний підхід дозволяє прив’язувати нотатки до конкретних дат, що є зручним для планування. У свою чергу, використання Kanban-дошок забезпечує візуальне представлення процесу виконання задач, що підвищує наочність і контроль над станом роботи .

Крім того, сучасні системи все частіше включають елементи аналітики. Аналіз активності користувача, кількості виконаних задач або розподілу нотаток за категоріями дозволяє отримати додаткову інформацію про ефективність роботи та допомагає приймати більш обґрунтовані рішення [7].

Незважаючи на значний розвиток інструментів для ведення нотаток, у цій предметній області залишаються певні проблеми. Зокрема, багато застосунків орієнтовані лише на один спосіб організації даних, що обмежує їх функціональність. Також часто відсутня інтеграція між різними режимами представлення інформації, що змушує користувача використовувати декілька окремих сервісів. Це створює додаткові незручності та ускладнює процес роботи з даними.

Таким чином, аналіз предметної області показує, що існує потреба у створенні універсального мобільного застосунку, який поєднує різні підходи до організації нотаток, забезпечує їх збереження у хмарному середовищі та надає користувачу додаткові інструменти для аналізу та планування діяльності.

1.2 Огляд існуючих застосунків для ведення нотаток

Сфера мобільних застосунків для ведення нотаток представлена великою кількістю рішень, які відрізняються як функціональністю, так і підходами до організації даних. Більшість із них орієнтовані на спрощення

процесу роботи з інформацією, проте реалізують це різними способами – від мінімалістичних списків до комплексних систем управління задачами [3].

Одним із найбільш відомих застосунків є Google Keep. Його основною перевагою є простота використання та швидкість створення нотаток. Користувач може додавати текстові записи, списки, зображення та голосові нотатки. Однак можливості структурування інформації в ньому обмежені, що ускладнює використання при роботі з великою кількістю даних.

Ще одним популярним рішенням є Evernote, який надає більш широкий функціонал. Застосунок підтримує створення складних нотаток, використання тегів, вкладень та синхронізацію між пристроями. Разом із тим, надлишкова функціональність і складніший інтерфейс можуть ускладнювати взаємодію для користувачів, яким потрібен швидкий доступ до інформації.

Застосунок Microsoft OneNote орієнтований на інтеграцію з іншими сервісами екосистеми Microsoft. Він дозволяє організовувати нотатки у вигляді блокнотів і розділів, що є зручним для структурованої роботи. Проте мобільна версія інколи поступається десктопній за швидкістю та зручністю.

Окрему категорію складають застосунки, що поєднують нотатки з управлінням задачами. До них належить Notion, який дозволяє створювати складні структури даних, бази знань та Kanban-дошки. Такий підхід є гнучким, але потребує часу на освоєння і може бути надлишковим для базових сценаріїв використання [13].

Для більш наочного представлення характеристик розглянутих застосунків доцільно провести їх порівняльний аналіз за ключовими параметрами (див. табл. 1.1).

Таблиця 1.1 – Порівняння застосунків для ведення нотаток

Застосунок	**Хмарна синхронізація**	**Режими представлення**	**Простота використання**	**Аналітика**	**Kanban**
Google Keep	+	Список	Висока	–	–

Продовження таблиці 1.1

Застосунок	**Хмарна синхронізація**	**Режими представлення**	**Простота використання**	**Аналітика**	**Kanban**
Evernote	+	Список		–	–
OneNote	+	Структурований	Середня	–	–
Notion	+	Список, Kanban	Низька	Частково	+

Як видно з таблиці, більшість існуючих рішень зосереджені на одному або двох способах організації даних. Наприклад, Google Keep і Evernote орієнтовані переважно на списки, тоді як Notion пропонує більш гнучкий підхід, але за рахунок складності інтерфейсу. Водночас жоден із розглянутих застосунків не поєднує в простому та доступному вигляді одразу кілька підходів до організації нотаток разом із вбудованою аналітикою. Це свідчить про наявність ніші для розробки застосунку, який поєднує простоту використання з розширеними можливостями організації та аналізу даних.

1.3 Порівняльний аналіз аналогів

У попередньому підрозділі було розглянуто основні мобільні застосунки для ведення нотаток, що відрізняються функціональністю та підходами до організації даних. Для більш глибокого аналізу доцільно провести їх порівняння за низкою критеріїв, які безпосередньо впливають на зручність використання та ефективність роботи користувача [3].

До основних критеріїв оцінювання віднесено: наявність хмарної синхронізації, способи представлення даних, підтримка візуального планування, рівень складності інтерфейсу, а також наявність інструментів аналітики. Обрані критерії дозволяють оцінити не лише функціональні можливості застосунків, але й їх практичну придатність для щоденного використання (див. табл. 1.2).

Таблиця 1.2 – Порівняльний аналіз інтерфейсу та можливостей застосунків для нотаток

Критерій / Застосунок	**Google Keep**	**Evernote**	**OneNote**	**Notion**
Хмарна синхронізація	+	+	+	+
Список нотаток	+	+	+	+
Календар	–	–	–	Частково
Kanban-дошка	–	–	–	+
Пошук	+	+	+	+
Гнучкість структури	Низька	Середня	Висока	Висока
Простота інтерфейсу	Висока	Середня	Середня	Низька
Аналітика	–	–	–	Частково

Аналіз наведених даних показує, що всі розглянуті застосунки підтримують базову функціональність створення та зберігання нотаток із використанням хмарної синхронізації. Це свідчить про те, що дана функція є стандартом для сучасних мобільних рішень у цій сфері [19].

Разом із тим, існують суттєві відмінності у способах організації інформації. Google Keep орієнтований на швидке створення нотаток без складної структури, тоді як OneNote і Notion пропонують більш гнучкі механізми роботи з даними. Останній, зокрема, підтримує Kanban-дошки, що дозволяє використовувати його як інструмент для планування задач [13].

Окремо варто відзначити рівень складності інтерфейсу. Застосунки з більш широкими можливостями, як правило, мають складніший інтерфейс, що може негативно впливати на швидкість освоєння та зручність використання. Це створює ситуацію, коли користувач змушений обирати між простотою та функціональністю.

Також слід звернути увагу на відсутність повноцінних інструментів аналітики у більшості застосунків. Навіть у тих рішеннях, де такі функції

частково реалізовані, вони не забезпечують повноцінного аналізу активності користувача або ефективності виконання задач.

Проведений аналіз дозволяє зробити висновок, що існуючі рішення не забезпечують одночасного поєднання таких характеристик, як простота використання, гнучкість організації даних, підтримка різних режимів представлення та наявність аналітичних можливостей. Це створює передумови для розробки нового застосунку, який усуває виявлені обмеження та поєднує сильні сторони розглянутих аналогів.

1.4 Визначення проблем та обґрунтування розробки

Проведений аналіз існуючих застосунків для ведення нотаток показав, що, незважаючи на їх широкий функціонал, жодне з розглянутих рішень не забезпечує повного поєднання простоти використання, гнучкості організації даних та наявності інструментів для їх аналізу. Це свідчить про наявність певних обмежень, які впливають на ефективність роботи користувача з інформацією.

Однією з основних проблем є обмеженість способів представлення даних. Більшість застосунків орієнтовані на один формат роботи з нотатками, як правило, у вигляді списку. Це ускладнює використання таких систем для планування задач або відстеження їх виконання, особливо при зростанні обсягу інформації.

Іншою суттєвою проблемою є відсутність інтеграції між різними підходами до організації даних. У тих випадках, коли застосунки підтримують декілька режимів роботи, вони часто не пов’язані між собою логічно, що знижує їх ефективність. У результаті користувач змушений використовувати кілька окремих інструментів для вирішення різних задач, що створює додаткове навантаження [4].

Також варто відзначити недостатній рівень підтримки аналітичних функцій. Більшість застосунків не надають можливості оцінити активність

користувача, кількість виконаних задач або розподіл навантаження. Відсутність таких інструментів ускладнює процес планування та не дозволяє отримати повну картину роботи з інформацією [7].

Окремо слід звернути увагу на баланс між функціональністю та зручністю використання. Більш складні системи, що мають широкий набір можливостей, часто вимагають значного часу на освоєння. У той же час прості застосунки не забезпечують необхідного рівня гнучкості. Це створює потребу у рішенні, яке поєднує достатній функціонал із зрозумілим інтерфейсом.

Ще одним важливим аспектом є забезпечення синхронізації даних. Хоча більшість сучасних застосунків використовують хмарні технології, не всі з них гарантують стабільну та швидку синхронізацію в реальному часі. Це може призводити до затримок оновлення інформації або втрати актуальності даних [19].

Враховуючи виявлені проблеми, доцільним є розробка мобільного застосунку, який поєднує різні способи представлення нотаток, забезпечує їх узгоджену роботу в межах єдиної системи та надає користувачу додаткові інструменти для аналізу даних. Особлива увага при цьому повинна приділятися простоті інтерфейсу та швидкості взаємодії з системою.

Запропонований підхід дозволяє створити застосунок, який усуває основні недоліки існуючих рішень і забезпечує більш ефективну організацію роботи з інформацією.

1.5 Висновки до розділу 1

У першому розділі було проведено аналіз предметної області ведення нотаток у мобільних застосунках та розглянуто сучасні підходи до організації інформації. Встановлено, що такі системи поступово трансформуються з простих інструментів для запису даних у більш складні рішення, які поєднують функції планування, структурування та обробки інформації.

У ході огляду існуючих застосунків було визначено їх основні особливості, переваги та обмеження. Проведений порівняльний аналіз показав, що більшість рішень орієнтовані на окремі аспекти роботи з нотатками, не забезпечуючи комплексного підходу до організації даних.

Виявлено низку ключових проблем, серед яких обмеженість способів представлення інформації, відсутність інтеграції між різними режимами роботи, недостатній рівень аналітичних можливостей та складність використання окремих систем. Зазначені недоліки ускладнюють ефективну роботу користувача та знижують зручність взаємодії з інформацією.

Отримані результати підтверджують доцільність розробки мобільного застосунку, який поєднує різні підходи до організації нотаток, забезпечує їх синхронізацію в хмарному середовищі та надає додаткові інструменти для аналізу і планування діяльності.

2 ПРОЄКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

2.1 Загальна архітектура системи

Проєктований мобільний застосунок побудовано за клієнт-серверним підходом із використанням хмарної платформи Firebase як серверної частини. Основна логіка взаємодії з користувачем реалізується на стороні мобільного клієнта, тоді як авторизація, зберігання та синхронізація даних виконуються через Firebase Auth і Cloud Firestore.

Архітектура застосунку має модульну структуру. Окремі екрани відповідають за різні сценарії роботи: авторизацію користувача, перегляд нотаток, роботу з Kanban-дошкою, аналітику та керування профілем. Такий підхід дозволяє розділити функціональні частини системи та спростити подальшу підтримку проєкту [11].

Центральним елементом логіки є робота з нотатками. Усі екрани, що використовують нотатки, звертаються до єдиного джерела даних через спеціальний програмний шар. У реалізації таким шаром виступає хук useNotes(), який відповідає за отримання, створення, редагування, видалення та синхронізацію нотаток з Cloud Firestore. Завдяки цьому Notes, Kanban та Analytics працюють з однаковими даними, а зміни автоматично відображаються в усіх модулях системи.

Схему загальної архітектури мобільного застосунку наведено на рисунку 2.1. У представленій архітектурі Firebase Auth відповідає за перевірку стану користувача та виконання входу в систему. Після успішної авторизації застосунок відкриває основні екрани, доступні через нижню навігацію. Якщо користувач не авторизований, він залишається на екрані входу.

Cloud Firestore використовується як хмарне сховище нотаток. Кожна нотатка зберігається в окремому документі та прив’язується до конкретного користувача через його унікальний ідентифікатор. Це дозволяє ізолювати дані

різних користувачів та забезпечити коректну роботу застосунку після входу з іншого пристрою [19].

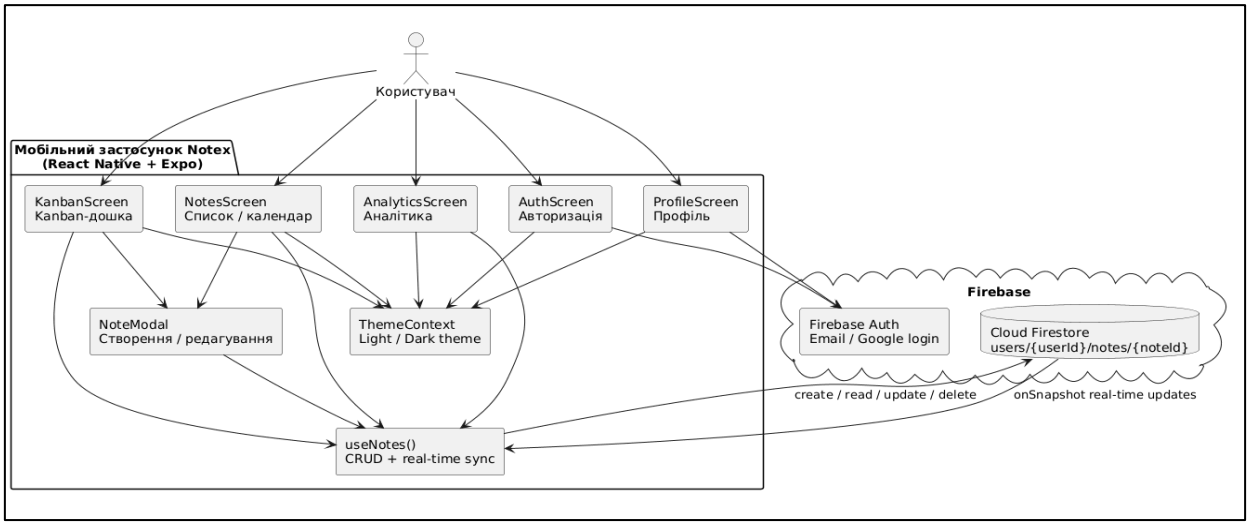


Рисунок 2.1 – Загальна архітектура мобільного застосунку Notex

Важливою особливістю архітектури є використання real-time синхронізації. Замість одноразового отримання даних застосунок підписується на зміни у Firestore. Якщо користувач створює, редагує, видаляє нотатку або змінює її статус у Kanban-дошці, оновлення автоматично надходить до всіх екранів, які працюють з цими даними.

Такий підхід забезпечує узгодженість між модулями системи та зменшує дублювання логіки. Замість того щоб реалізовувати окрему роботу з базою даних для кожного екрана, застосунок використовує спільний механізм доступу до нотаток. Це робить архітектуру більш зрозумілою, підтримуваною та придатною до подальшого розширення.

2.2 Вибір технологій та обґрунтування

Вибір технологічного стеку для розробки мобільного застосунку здійснювався з урахуванням вимог до продуктивності, зручності розробки, масштабованості та можливості інтеграції з хмарними сервісами. Оскільки

застосунок орієнтований на роботу з даними в реальному часі та доступність з різних пристроїв, ключову роль відіграє підтримка стабільної синхронізації та швидкої взаємодії з користувачем.

Як основну платформу для розробки клієнтської частини обрано React Native. Даний фреймворк дозволяє створювати мобільні застосунки з використанням мови JavaScript та бібліотеки React, що значно спрощує процес розробки та забезпечує високу швидкість створення інтерфейсів. Крім того, React Native підтримує компонентний підхід, який добре підходить для побудови модульної архітектури застосунку [17].

Для спрощення процесу розробки та запуску застосунку використано платформу Expo. Вона дозволяє швидко розгортати проєкт, тестувати його на реальних пристроях та використовувати готові інструменти для роботи з навігацією, стилями та системними можливостями мобільних пристроїв. Це суттєво скорочує час розробки та знижує складність налаштування середовища [9].

Як серверну частину обрано платформу Firebase, яка реалізує підхід Backend-as-a-Service. Використання Firebase дозволяє уникнути розробки власного серверу та забезпечує готові рішення для авторизації користувачів і зберігання даних. Зокрема, застосунок використовує Firebase Authentication для керування доступом користувачів і Cloud Firestore як основну базу даних.

Cloud Firestore забезпечує зберігання даних у вигляді документів та колекцій, що добре підходить для реалізації нотаток. Крім того, важливою перевагою є підтримка синхронізації в реальному часі, що дозволяє автоматично оновлювати дані без додаткових запитів до серверу. Такий підхід значно покращує взаємодію користувача із системою.

Для реалізації візуального планування використано підхід Kanban, який дозволяє організувати нотатки за статусами виконання. Це забезпечує наочність процесу роботи та спрощує керування задачами. Для відображення аналітичних даних застосовано бібліотеки побудови графіків, що дозволяють візуалізувати статистику використання нотаток.

З метою узагальнення обраних технологій та їх характеристик доцільно представити їх у вигляді таблиці (див. табл. 2.1).

Таблиця 2.1 – Обґрунтування вибору технологій

Технологія	**Призначення**	**Переваги**	**Недоліки**
React Native	Розробка мобільного клієнта	Кросплатформеність, компонентний підхід	Залежність від JS
Expo	Середовище розробки	Простота запуску, швидке тестування	Обмеження кастомізації
Firebase Auth	Авторизація користувачів	Готові рішення, безпека	Залежність від сервісу
Cloud Firestore	Хмарна база даних	Real-time синхронізація, масштабованість	Обмеження складних запитів
Chart бібліотеки	Візуалізація даних	Наглядність, простота інтеграції	Обмежена гнучкість

Обраний технологічний стек дозволяє забезпечити всі необхідні функціональні можливості застосунку без значного ускладнення архітектури. Використання React Native та Expo спрощує розробку мобільного інтерфейсу, тоді як Firebase надає готові інструменти для роботи з користувачами та даними. Завдяки поєднанню цих технологій вдається досягти балансу між функціональністю, швидкістю розробки та зручністю використання застосунку, що є важливим фактором для сучасних мобільних систем.

2.3 Проєктування бази даних

При розробці мобільного застосунку для ведення нотаток важливим етапом є проєктування структури зберігання даних. Оскільки застосунок використовує хмарну платформу Firebase, як базу даних обрано Cloud

Firestore, яка реалізує документно-орієнтовану модель зберігання інформації [19].

На відміну від традиційних реляційних баз даних, Firestore використовує структуру «колекція – документ», що дозволяє зберігати дані у вигляді вкладених об’єктів. Такий підхід добре підходить для мобільних застосунків, оскільки забезпечує гнучкість, масштабованість та ефективну роботу з даними в реальному часі.

У розробленому застосунку використовується ієрархічна структура зберігання даних, де всі нотатки прив’язані до конкретного користувача. Основна структура бази даних має вигляд:

users/{userId}/notes/{noteId}.

Це означає, що для кожного користувача створюється окрема колекція нотаток, що забезпечує ізоляцію даних та підвищує безпеку доступу. Фрагмент структури бази даних у середовищі Firebase наведено на рисунках 2.2 – 2.3.

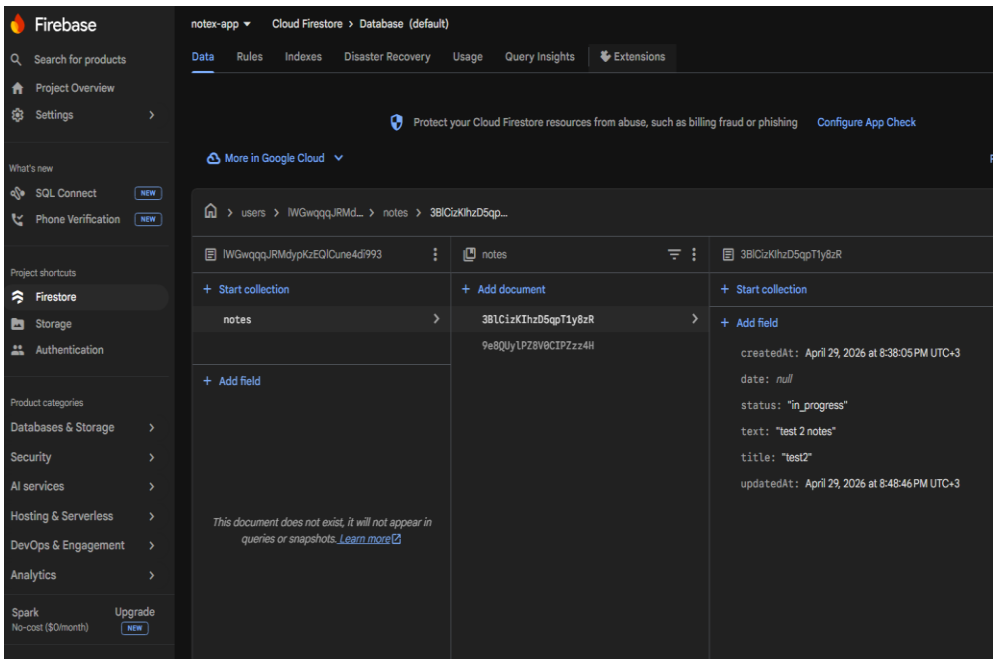


Рисунок 2.2 – Інтерфейс Cloud Firestore у консолі Firebase із відображенням структури бази даних NoSQL

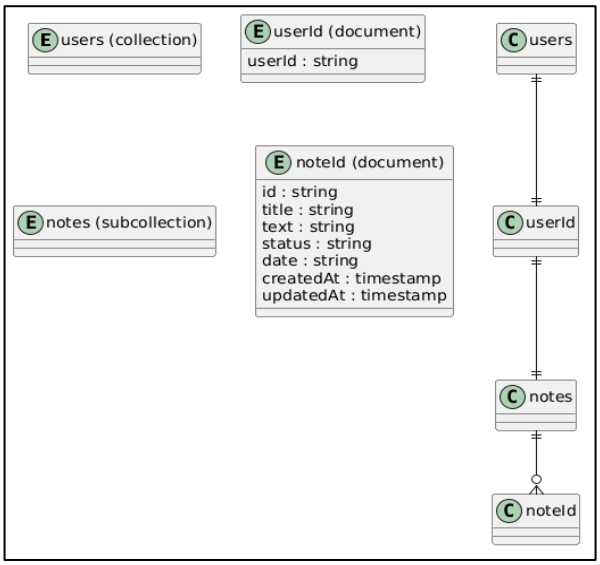


Рисунок 2.3 – Структура бази даних у Cloud Firestore

Як видно з рисунка, кожна нотатка зберігається як окремий документ, який містить набір полів. Основні поля документа нотатки наведено в таблиці 2.2.

Таблиця 2.2 – Структура документа нотатки

Поле	**Тип даних**	**Опис**
id	string	Унікальний ідентифікатор нотатки
title	string	Заголовок нотатки
text	string	Основний текст нотатки
status	string	Статус (todo, in_progress, done)
date	string	Дата, пов’язана з нотаткою
createdAt	timestamp	Дата створення
updatedAt	timestamp	Дата останнього оновлення

Для підтвердження підключення до хмарної бази даних у застосунку використовується конфігурація Firebase, наведена на рисунку 2.4.

У представленому фрагменті коду здійснюється ініціалізація Firebase-застосунку, а також підключення модулів авторизації та бази даних. Після

ініціалізації об’єкти auth та db використовуються у всіх частинах застосунку для взаємодії з користувачами та даними.

notex > src > config > JS firebase.js > ...

```
1   import { initializeApp } from 'firebase/app';
2   import { getAuth } from 'firebase/auth';
3   import { getFirestore } from 'firebase/firestore';
4
5   const firebaseConfig = {
6     apiKey: 'AIzaSyD7UvKFxsgtRJN2S2VWWDRSoo2R8mY8WyW8',
7     authDomain: 'notex-app-f00ea.firebaseapp.com',
8     projectId: 'notex-app-f00ea',
9     storageBucket: 'notex-app-f00ea.firebasestorage.app',
10    messagingSenderId: '247612962515',
11    appId: '1:247612962515:web:99c23aeb35e5c1e4d00626',
12  };
13
14  const app = initializeApp(firebaseConfig);
15
16  export const auth = getAuth(app);
17  export const db = getFirestore(app);
18
```

Рисунок 2.4 – Підключення Firebase у мобільному застосунку

Використання Cloud Firestore дозволяє реалізувати синхронізацію даних у реальному часі. Це означає, що будь-які зміни нотаток автоматично передаються на клієнтський застосунок без необхідності ручного оновлення. Такий підхід значно покращує користувацький досвід і забезпечує узгодженість даних між різними модулями системи.

Обрана структура бази даних є достатньо гнучкою для подальшого розширення. За необхідності вона може бути доповнена новими полями або вкладеними колекціями без суттєвих змін існуючої логіки застосунку.

2.4 Опис логіки роботи системи

Логіка роботи мобільного застосунку побудована на взаємодії користувача з інтерфейсом та хмарною базою даних у режимі реального часу.

Основні процеси включають авторизацію, роботу з нотатками, зміну їх стану та відображення аналітичної інформації.

Після запуску застосунку виконується перевірка стану авторизації користувача. Якщо користувач вже увійшов у систему, він отримує доступ до основного функціоналу застосунку. У протилежному випадку відображається екран входу, де користувач може авторизуватися за допомогою електронної пошти або облікового запису Google.

Після успішної авторизації відкривається основний інтерфейс застосунку, який включає декілька екранів: список нотаток, календар, Kanban-дошку, аналітику та профіль користувача. Перемикання між екранами здійснюється за допомогою навігації, при цьому всі екрани працюють з єдиним джерелом даних.

Центральним елементом логіки є взаємодія з нотатками. Користувач може створювати нові нотатки, редагувати існуючі або видаляти їх. Усі ці дії обробляються через спільний програмний механізм, який відповідає за синхронізацію з Cloud Firestore. Завдяки цьому зміни миттєво відображаються на всіх екранах застосунку.

Особливу роль відіграє Kanban-дошка, яка дозволяє змінювати статус нотаток шляхом перетягування між колонками. При зміні статусу відповідне поле документа в базі даних оновлюється, після чого ці зміни автоматично відображаються в інших представленнях, зокрема у списку нотаток та аналітиці.

Модуль аналітики здійснює обробку даних та відображає статистичну інформацію, зокрема кількість нотаток, їх розподіл за статусами та загальну активність користувача. Це дозволяє отримати узагальнену картину використання застосунку.

Загальний процес роботи системи наведено на рисунку 2.5. Представлена діаграма ілюструє основні етапи взаємодії користувача із застосунком. Ключовою особливістю є використання автоматичної синхронізації даних, що дозволяє забезпечити узгодженість інформації між різними екранами системи.

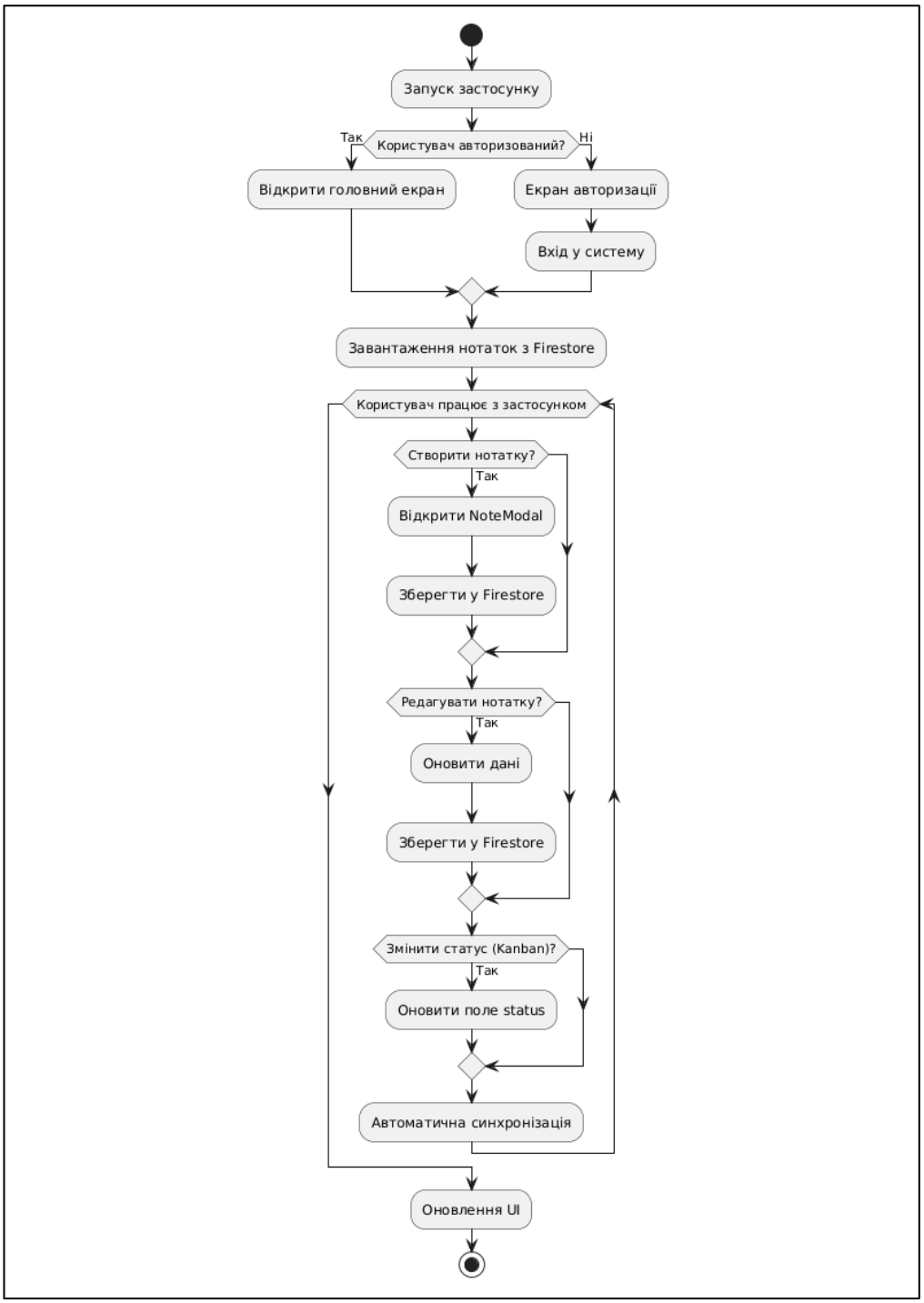


Рисунок 2.5 – Логіка роботи мобільного застосунку

Усі операції з нотатками виконуються через єдиний механізм доступу до даних, що спрощує логіку застосунку та зменшує ймовірність помилок. Завдяки цьому система залишається стабільною навіть при активній взаємодії користувача з різними модулями.

Реалізована логіка дозволяє забезпечити швидкий відгук інтерфейсу, зручність використання та ефективну роботу з даними у реальному часі.

2.5 Алгоритми основних процесів

У межах розробленого мобільного застосунку можна виділити декілька основних процесів, які визначають його функціонування. До них належать створення нотатки, зміна її стану та синхронізація даних із хмарною базою. Формалізація цих процесів у вигляді алгоритмів дозволяє краще зрозуміти логіку роботи системи та спрощує її подальшу реалізацію [11].

2.5.1 Алгоритм створення нотатки

Процес створення нової нотатки починається з дії користувача та завершується збереженням даних у хмарній базі.

Загальну схему алгоритму наведено на рисунку 2.6. Після введення даних формується об’єкт нотатки, який передається до бази даних. У випадку успішного збереження застосунок автоматично оновлює відображення даних на екрані.



Рисунок 2.6 – Алгоритм створення нотатки

2.5.2 Алгоритм зміни статусу нотатки (Kanban)

Зміна статусу реалізується шляхом перетягування нотатки між колонками. Схему алгоритму наведено на рисунку 2.7.



Рисунок 2.7 – Алгоритм зміни статусу нотатки

Такий підхід дозволяє швидко змінювати стан задач без додаткових дій з боку користувача, що підвищує зручність роботи із застосунком.

2.5.3 Алгоритм синхронізації даних

Синхронізація є одним із ключових процесів, що забезпечує узгодженість інформації у системі. Алгоритм наведено на рисунку 2.8.

На відміну від традиційного підходу, де клієнт періодично запитує дані, у даному випадку використовується механізм підписки на зміни. Це дозволяє отримувати актуальну інформацію практично миттєво без додаткових запитів [19].



Рисунок 2.8 – Алгоритм синхронізації даних

Застосування наведених алгоритмів дозволяє забезпечити узгоджену роботу всіх компонентів системи. Вони охоплюють основні сценарії взаємодії користувача із застосунком і відображають логіку обробки даних у реальному часі.

2.6 Висновки до розділу 2

У другому розділі було розглянуто процес проєктування мобільного застосунку для ведення нотаток із підтримкою хмарного зберігання. Визначено загальну архітектуру системи, яка базується на клієнт-серверному підході з використанням хмарної платформи Firebase, що забезпечує зберігання та синхронізацію даних у реальному часі.

Обґрунтовано вибір технологічного стеку, який включає React Native та Expo для реалізації мобільного клієнта, а також Firebase як серверної частини. Зазначені технології дозволяють забезпечити необхідний функціонал застосунку, зберігаючи баланс між продуктивністю, гнучкістю та зручністю розробки.

У ході проєктування бази даних визначено структуру зберігання інформації у Cloud Firestore, яка має ієрархічний вигляд і дозволяє ефективно організувати дані користувачів. Такий підхід забезпечує масштабованість системи та можливість її подальшого розширення без суттєвих змін у логіці роботи.

Описано логіку функціонування застосунку, яка охоплює процеси авторизації, роботи з нотатками, зміни їх стану та відображення аналітичної інформації. Встановлено, що використання єдиного механізму доступу до даних дозволяє забезпечити узгодженість між усіма модулями системи.

Також було формалізовано основні процеси у вигляді алгоритмів, що дозволило чітко представити послідовність дій при роботі з нотатками та механізм синхронізації даних. Це спрощує розуміння внутрішньої логіки системи та підвищує якість її реалізації.

Отримані результати підтверджують, що обрані архітектурні та технологічні рішення є доцільними та забезпечують ефективну роботу мобільного застосунку відповідно до поставлених вимог.

3 РЕАЛІЗАЦІЯ ЗАСТОСУНКУ

3.1 Реалізація авторизації

У розробленому мобільному застосунку реалізовано систему авторизації користувачів, яка забезпечує доступ до персоналізованих даних та їх захист. Для цього використано сервіс Firebase Authentication, що дозволяє організувати безпечну роботу з обліковими записами без необхідності реалізації власної серверної логіки [10].

У застосунку передбачено два основних способи авторизації:

– вхід за допомогою електронної пошти та пароля;

– вхід через обліковий запис Google.

Після запуску застосунку користувачу відображається екран входу, наведений на рисунку 3.1.

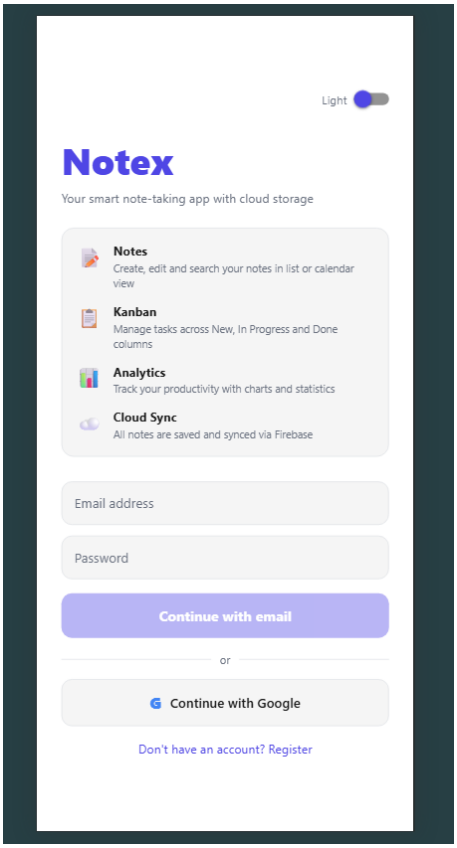


Рисунок 3.1 – Екран входу користувача

Інтерфейс екрана реалізовано з урахуванням зручності використання: користувач має можливість швидко ввести необхідні дані або обрати альтернативний спосіб входу. Поля введення супроводжуються базовою валідацією, що дозволяє запобігти помилкам при введенні некоректних даних, зокрема порожніх значень або неправильного формату електронної пошти.

У випадку відсутності облікового запису користувач може перейти до екрану реєстрації, який наведено на рисунку 3.2.

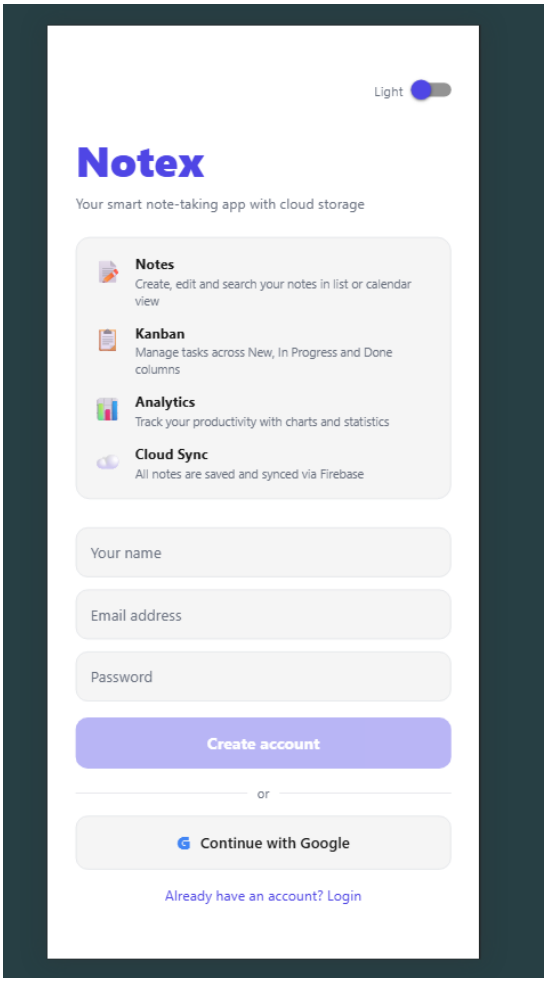


Рисунок 3.2 – Екран реєстрації користувача

Під час реєстрації користувач вводить ім’я, електронну пошту та пароль. Після підтвердження даних створюється новий обліковий запис у Firebase Authentication. У процесі реєстрації також виконується перевірка коректності введених даних, що дозволяє підвищити надійність системи та уникнути некоректних записів.

Для спрощення взаємодії з користувачем реалізовано можливість входу через Google. Даний підхід дозволяє уникнути ручного введення даних та значно скорочує час авторизації.

Після успішного входу застосунок отримує основну інформацію про користувача (ідентифікатор, електронну пошту), яка використовується для подальшої роботи із системою. Ці дані також застосовуються для організації доступу до нотаток у хмарній базі даних.

Важливим елементом реалізації є автоматичне визначення стану авторизації. Застосунок використовує механізм відстеження змін користувача, що дозволяє визначити, чи виконано вхід у систему під час запуску. Якщо користувач уже авторизований, він одразу перенаправляється до основного інтерфейсу без повторного введення даних.

Такий підхід покращує користувацький досвід та забезпечує безперервність роботи із застосунком. Крім того, використання централізованої системи авторизації дозволяє підвищити рівень безпеки, оскільки всі облікові дані обробляються на стороні Firebase і не зберігаються безпосередньо в застосунку [10].

Реалізована система авторизації забезпечує надійний доступ до персональних даних користувача, а також створює основу для подальшої роботи з нотатками у хмарному середовищі.

3.2 Реалізація роботи з нотатками

Основною функціональною частиною мобільного застосунку є модуль роботи з нотатками, який забезпечує створення, перегляд, редагування та видалення записів. Реалізація цього модуля побудована таким чином, щоб забезпечити зручність взаємодії користувача з даними та їх синхронізацію у хмарному середовищі. Головний екран роботи з нотатками наведено на рисунку 3.3.

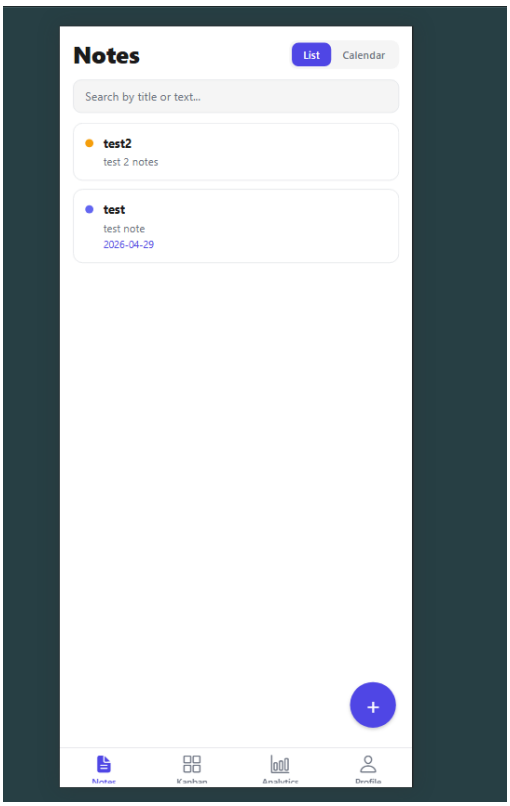


Рисунок 3.3 – Екран списку нотаток

На даному екрані відображається список усіх нотаток користувача. Для зручності передбачено пошук за заголовком або текстом, що дозволяє швидко знаходити необхідну інформацію. Кожна нотатка відображається у вигляді окремого елемента, який містить основні дані: назву, короткий опис та дату.

Окрему роль відіграє можливість перемикання між режимами відображення. Користувач може переглядати нотатки у вигляді списку або календаря, що підвищує гнучкість роботи із застосунком.

Для створення нової нотатки використовується модальне вікно, наведене на рисунку 3.4.

У цьому вікні користувач вводить основні дані нотатки, зокрема заголовок, текст, дату та статус. Інтерфейс побудований таким чином, щоб мінімізувати кількість дій при створенні запису. Після підтвердження дані передаються до Cloud Firestore, де зберігаються у вигляді окремого документа.

Важливим елементом є підтримка статусів нотаток, що використовується для подальшої роботи з Kanban-дошкою. Це дозволяє одразу визначити стан задачі під час її створення.

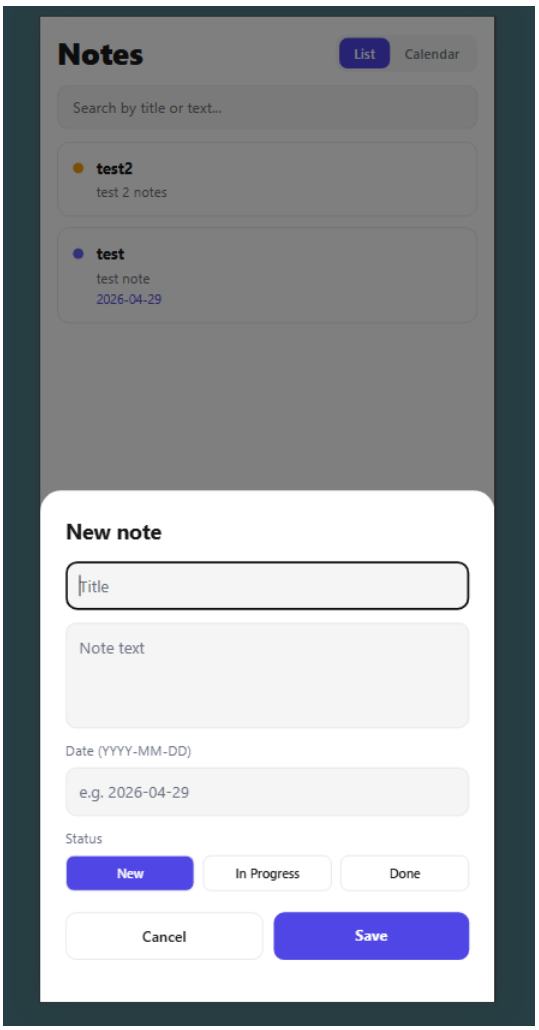


Рисунок 3.4 – Створення нової нотатки

Альтернативним способом відображення нотаток є календарний режим, який наведено на рисунку 3.5.

У цьому режимі нотатки прив’язуються до конкретних дат, що дозволяє використовувати застосунок як інструмент планування. Користувач може переглядати записи за днями та швидко орієнтуватися у розкладі.

З технічної точки зору робота з нотатками реалізована через єдиний програмний механізм доступу до даних. Усі операції (створення, оновлення, видалення) виконуються через централізований модуль, що взаємодіє з Cloud Firestore. Це дозволяє уникнути дублювання логіки та забезпечити узгодженість даних між різними частинами застосунку.

Окремо варто відзначити використання синхронізації в реальному часі. При зміні будь-якої нотатки відповідні оновлення миттєво відображаються в

інтерфейсі без необхідності перезавантаження. Такий підхід значно покращує користувацький досвід і робить роботу із застосунком більш зручною [19].

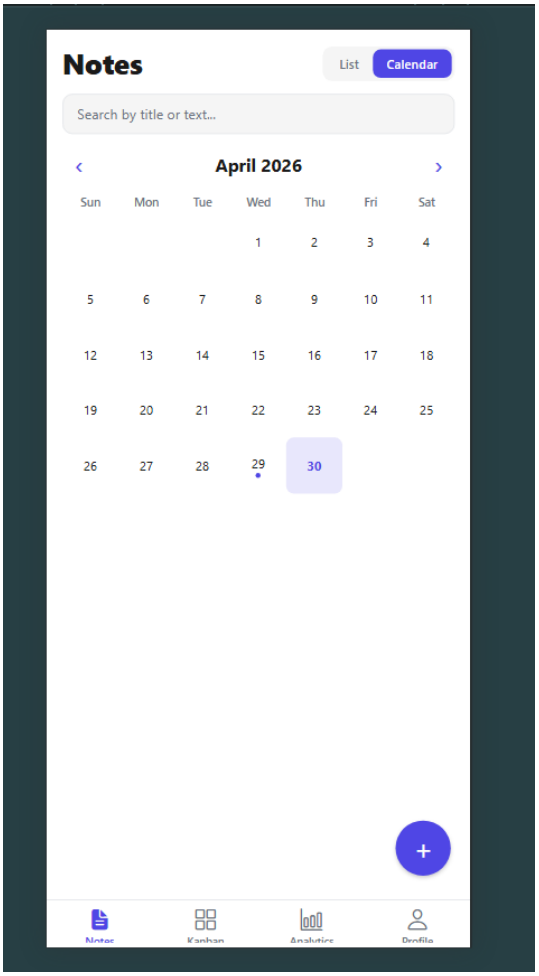


Рисунок 3.5 – Календарне представлення нотаток

Реалізований модуль роботи з нотатками забезпечує повний цикл взаємодії з даними та створює основу для інших функціональних частин застосунку, зокрема Kanban-дошки та аналітики.

3.3 Реалізація Kanban-дошки

Одним із ключових елементів мобільного застосунку є реалізація Kanban-дошки, яка використовується для візуального представлення та керування станом нотаток. Такий підхід дозволяє організувати дані у вигляді

колонок відповідно до етапів виконання задач, що значно підвищує наочність та зручність роботи користувача. Загальний вигляд Kanban-дошки наведено на рисунку 3.6.

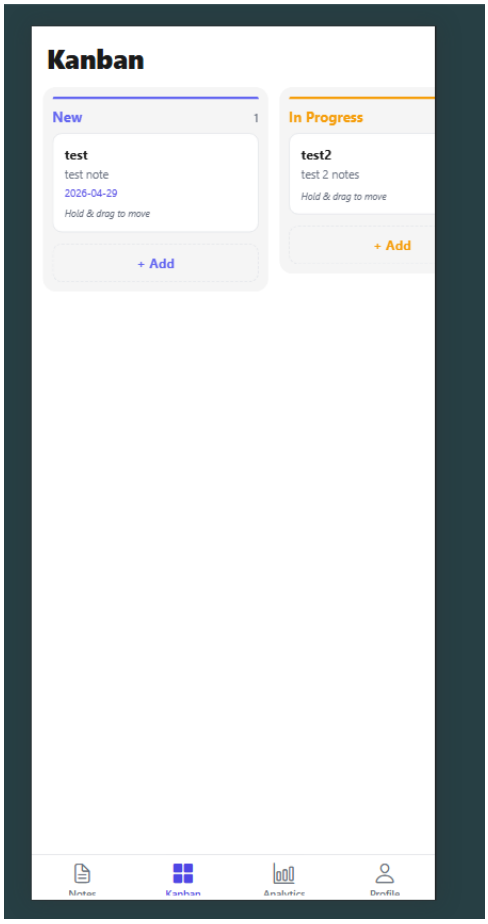


Рисунок 3.6 – Основний вигляд Kanban-дошки

Інтерфейс дошки поділений на декілька колонок, кожна з яких відповідає певному статусу нотатки:

– New – нові задачі;
– In Progress – задачі в процесі виконання;
– Done – завершені задачі.

Кожна нотатка відображається у вигляді картки, що містить заголовок, короткий опис та дату. Додатково користувач може створювати нові нотатки безпосередньо у відповідній колонці.

Однією з ключових функцій є можливість зміни статусу нотатки шляхом перетягування. Приклад взаємодії наведено на рисунку 3.7.

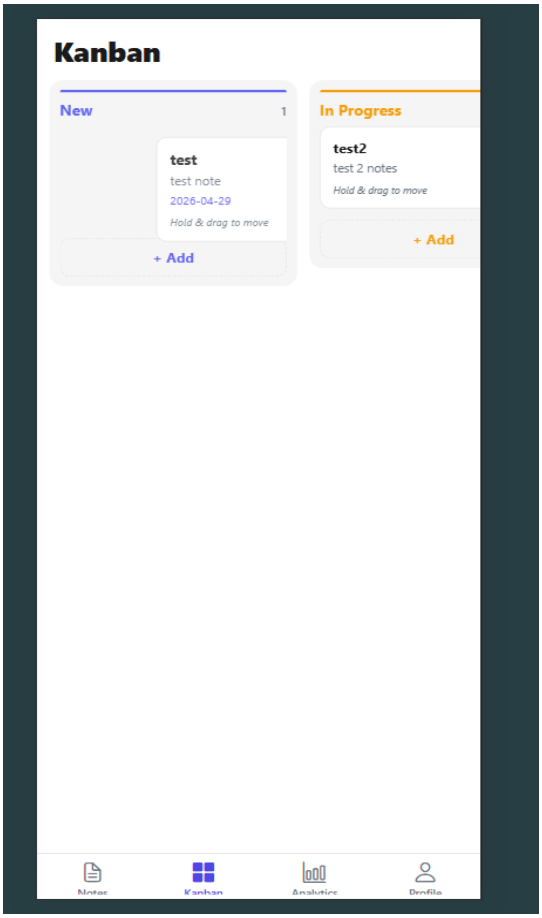


Рисунок 3.7 – Переміщення нотатки між колонками

У процесі перетягування система визначає нову колонку та автоматично оновлює статус нотатки. Це дозволяє виконувати керування задачами без додаткових дій, що значно покращує користувацький досвід.

З технічної точки зору при зміні положення картки виконується оновлення поля status у відповідному документі бази даних. Після цього зміни передаються до Cloud Firestore, а завдяки механізму синхронізації в реальному часі вони миттєво відображаються у всіх частинах застосунку [19].

Окрім зміни статусу, користувач має можливість редагувати або видаляти нотатку. Відповідне модальне вікно наведено на рисунку 3.8.

У цьому режимі користувач може змінювати заголовок, текст, дату та статус нотатки. Видалення запису також виконується безпосередньо з цього вікна, що спрощує роботу із системою.

Важливою особливістю реалізації є те, що Kanban-дошка використовує ті ж самі дані, що і інші модулі застосунку. Це означає, що всі зміни

автоматично синхронізуються зі списком нотаток та аналітичним модулем. Такий підхід дозволяє уникнути дублювання даних та забезпечує узгодженість інформації.

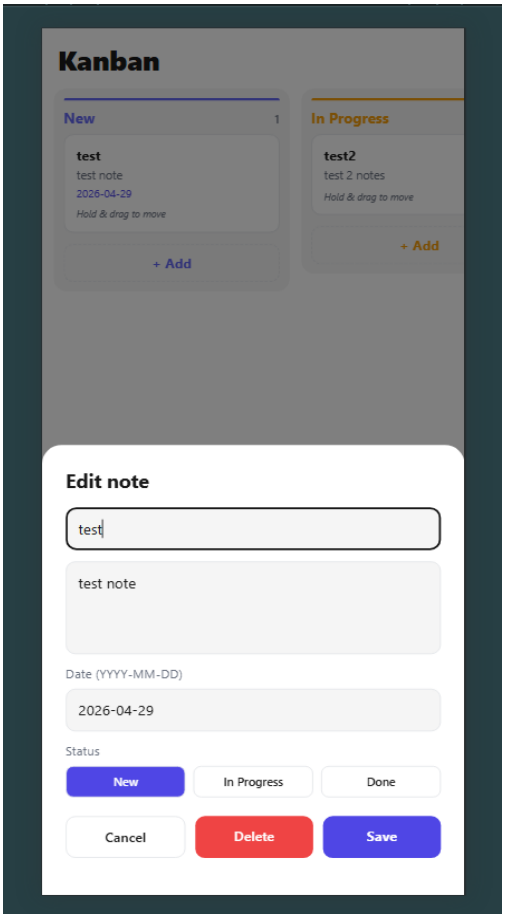


Рисунок 3.8 – Редагування нотатки у Kanban-дошці

Реалізація Kanban-дошки значно розширює функціональні можливості застосунку, дозволяючи використовувати його не лише для зберігання нотаток, але і як інструмент планування та контролю виконання задач.

3.4 Реалізація аналітики

У мобільному застосунку реалізовано модуль аналітики, який дозволяє отримати узагальнену інформацію про використання нотаток та оцінити активність користувача. На відміну від стандартних застосунків для ведення

нотаток, даний модуль забезпечує додатковий рівень роботи з даними, що підвищує практичну цінність системи. Загальний вигляд екрана аналітики наведено на рисунку 3.9.

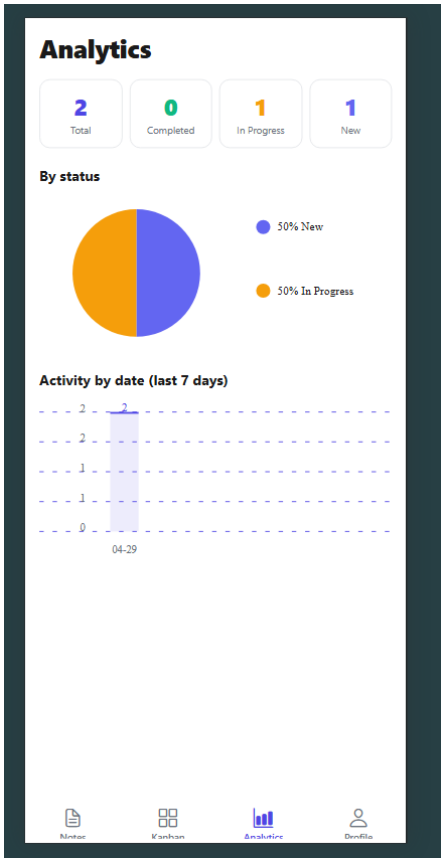


Рисунок 3.9 – Екран аналітики нотаток

У верхній частині екрана відображаються основні показники:

– загальна кількість нотаток;
– кількість завершених задач;
– кількість задач у процесі виконання;
– кількість нових задач.

Ці значення обчислюються на основі даних, отриманих із Cloud Firestore, що дозволяє відображати актуальну інформацію у режимі реального часу.

Для більш наочного представлення використано графічні елементи. Зокрема, кругова діаграма відображає розподіл нотаток за статусами. Це дозволяє швидко оцінити співвідношення між виконаними та невиконаними задачами.

Додатково реалізовано відображення активності користувача за останні дні у вигляді стовпчикової діаграми. Такий підхід дозволяє відстежувати динаміку створення або оновлення нотаток та оцінювати рівень активності.

З технічної точки зору модуль аналітики виконує агрегацію даних, отриманих із бази. Для цього використовується обробка масиву нотаток, де підраховуються значення за різними критеріями, зокрема за статусами та датами створення. Результати обчислень передаються до компонентів відображення графіків.

Важливою особливістю є те, що аналітика працює з тими ж даними, що й інші модулі застосунку. Це забезпечує узгодженість інформації та дозволяє уникнути дублювання логіки. При будь-яких змінах нотаток відповідні показники автоматично оновлюються завдяки механізму синхронізації в реальному часі [19].

Реалізація аналітичного модуля дозволяє перетворити застосунок із простого інструменту для зберігання нотаток у систему, що підтримує аналіз діяльності користувача та сприяє більш ефективному плануванню.

3.5 Інтерфейс користувача

Інтерфейс користувача мобільного застосунку розроблено з урахуванням принципів зручності, простоти та швидкого доступу до основного функціоналу. Основною метою при проєктуванні інтерфейсу було забезпечення інтуїтивно зрозумілої взаємодії користувача із системою без необхідності додаткового навчання [14].

Загальна структура інтерфейсу базується на використанні нижньої навігаційної панелі, яка забезпечує швидкий доступ до основних розділів застосунку:

– нотатки;
– Kanban-дошка;

– аналітика;

– профіль користувача.

Такий підхід дозволяє користувачу швидко перемикатися між функціональними модулями та працювати з даними у різних представленнях.

Інтерфейс профілю користувача наведено на рисунку 3.10. На даному екрані відображається основна інформація про користувача, зокрема електронна пошта та статус синхронізації. Крім того, передбачено можливість зміни теми оформлення та виходу з облікового запису.

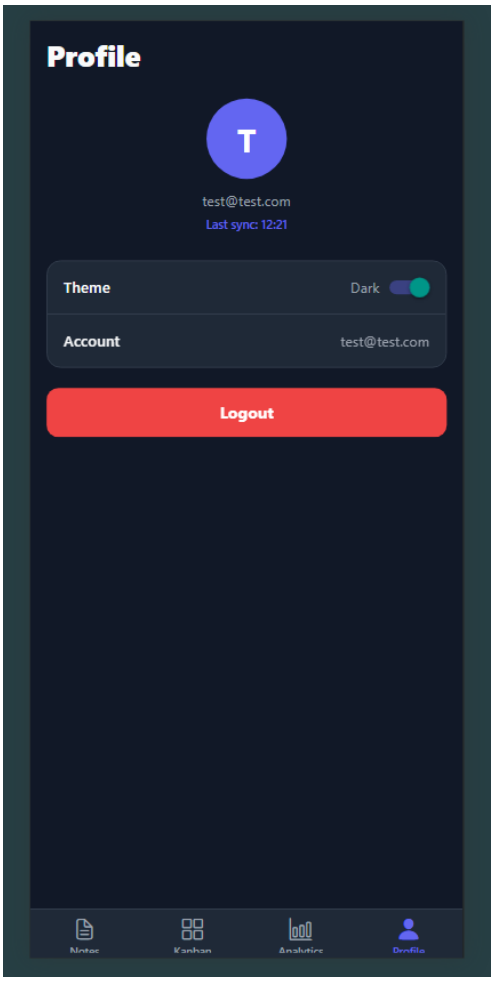


Рисунок 3.10 – Екран профілю користувача

Однією з важливих особливостей є підтримка світлої та темної теми оформлення. Користувач може змінювати режим відображення залежно від умов використання або власних уподобань. Це підвищує комфорт роботи із застосунком та відповідає сучасним вимогам до мобільних інтерфейсів.

При розробці інтерфейсу було використано компонентний підхід, що дозволяє повторно використовувати елементи в різних частинах застосунку. Наприклад, модальні вікна, картки нотаток та елементи керування мають єдину стилістику, що забезпечує цілісність дизайну.

Особлива увага приділена візуальному оформленню. Для виділення важливих елементів використовуються кольори, а також різні рівні контрасту. Це дозволяє користувачу швидко орієнтуватися в інтерфейсі та знаходити необхідні функції.

Також враховано адаптивність інтерфейсу до різних розмірів екранів. Розташування елементів та їх масштабування забезпечують коректне відображення застосунку на різних мобільних пристроях.

У результаті реалізовано інтерфейс, який поєднує простоту використання, функціональність та сучасний дизайн, що сприяє ефективній взаємодії користувача із системою.

3.6 Висновки до розділу 3

У третьому розділі було розглянуто процес реалізації мобільного застосунку для ведення нотаток із підтримкою хмарного зберігання. У ході роботи реалізовано основні функціональні модулі системи, що забезпечують повний цикл взаємодії користувача з даними.

Зокрема, реалізовано систему авторизації користувачів із використанням Firebase Authentication, яка забезпечує безпечний доступ до персоналізованих даних. Передбачено декілька способів входу, що підвищує зручність використання застосунку.

Розроблено модуль роботи з нотатками, який включає створення, редагування, перегляд та видалення записів. Забезпечено можливість відображення даних у вигляді списку та календаря, що дозволяє користувачу обирати найбільш зручний формат роботи.

Окрему увагу приділено реалізації Kanban-дошки, яка дозволяє керувати станом нотаток у візуальному форматі. Використання механізму перетягування забезпечує інтуїтивно зрозумілу взаємодію та спрощує процес планування задач.

Також реалізовано модуль аналітики, який дозволяє отримувати узагальнену інформацію про діяльність користувача. Використання графічних елементів забезпечує наочне представлення даних та сприяє їх кращому сприйняттю.

У ході розробки інтерфейсу користувача забезпечено зручну навігацію, підтримку різних режимів відображення та адаптивність до мобільних пристроїв. Це дозволяє підвищити ефективність взаємодії користувача із застосунком.

Отримані результати підтверджують, що реалізований застосунок забезпечує необхідний функціонал, відповідає сучасним вимогам до мобільних систем та може бути використаний як інструмент для організації та аналізу нотаток.

4 ТЕСТУВАННЯ ТА АНАЛІЗ РОБОТИ

4.1 Методика тестування

Тестування мобільного застосунку проводилось з метою перевірки коректності роботи основних функціональних модулів, а також забезпечення стабільності системи при взаємодії з користувачем.

У процесі розробки було застосовано підхід модульного тестування (unit testing), який передбачає перевірку окремих компонентів системи незалежно один від одного.

Для реалізації тестування використано інструмент **Jest**, що дозволяє виконувати автоматизовану перевірку логіки застосунку.

Тестування охоплює такі основні модулі:

– робота з нотатками (створення, пошук, фільтрація);
– аналітика (обчислення статистичних показників);
– авторизація (валідація даних користувача).

Основна увага приділялася перевірці:

– коректності обробки вхідних даних;
– правильності роботи алгоритмів;
– узгодженості результатів при зміні даних.

Такий підхід дозволяє виявляти помилки на ранніх етапах розробки та забезпечує стабільну роботу застосунку [11].

4.2 Функціональне тестування

У межах функціонального тестування було перевірено роботу ключових сценаріїв використання застосунку. Результати тестування наведено на рисунках 4.1 – 4.3.

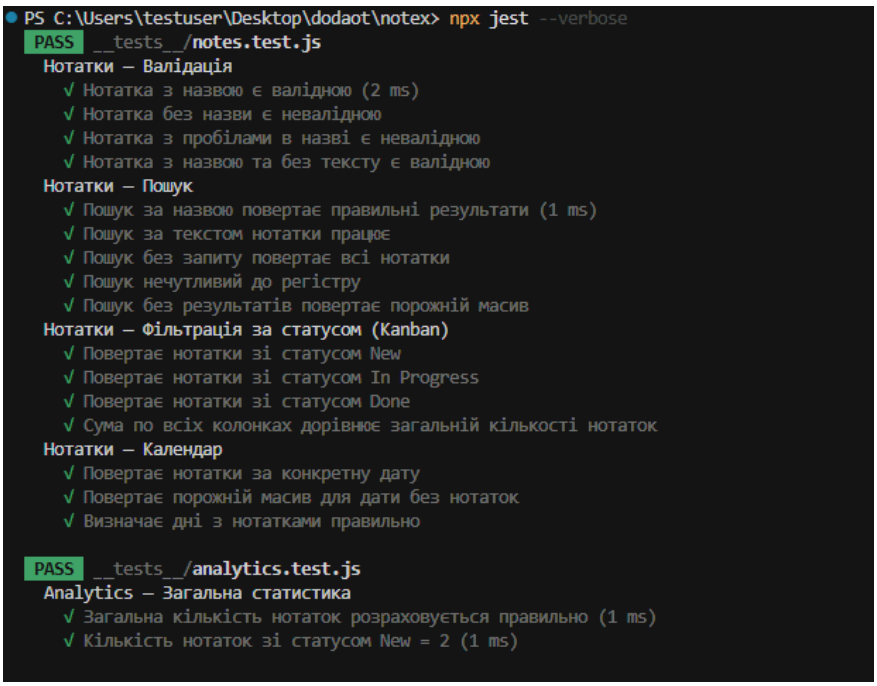


Рисунок 4.1 – Результати тестування модуля роботи з нотатками

У процесі тестування модуля нотаток перевірено:

– валідацію введених даних;

– коректність пошуку;

– фільтрацію за статусами;

– роботу календарного представлення.

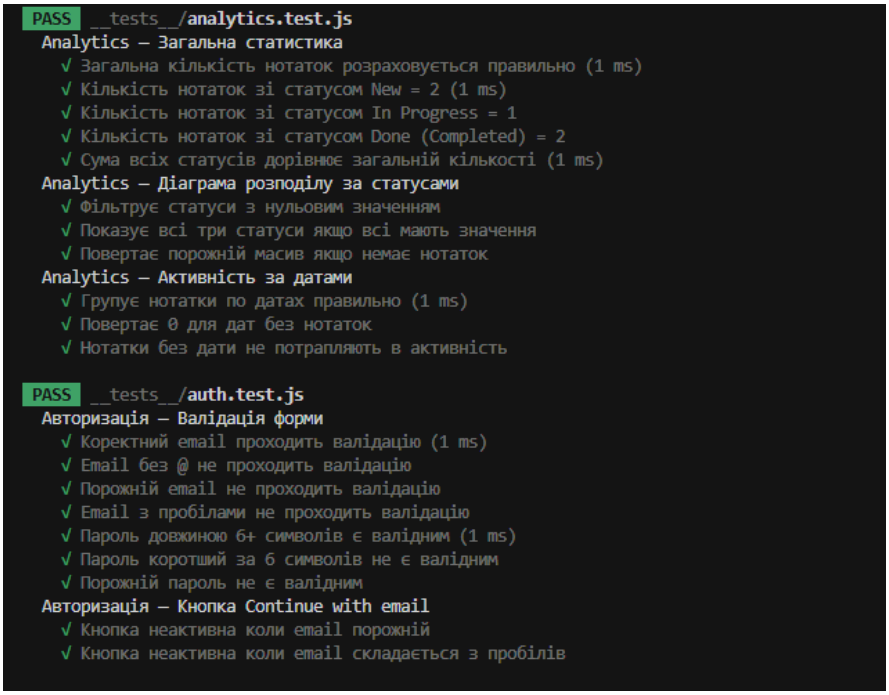


Рисунок 4.2 – Результати тестування модуля аналітики

Для модуля аналітики перевірено:

– правильність обчислення загальної кількості нотаток;
– коректність групування за статусами;
– побудову статистичних даних;
– обробку граничних випадків (відсутність даних).

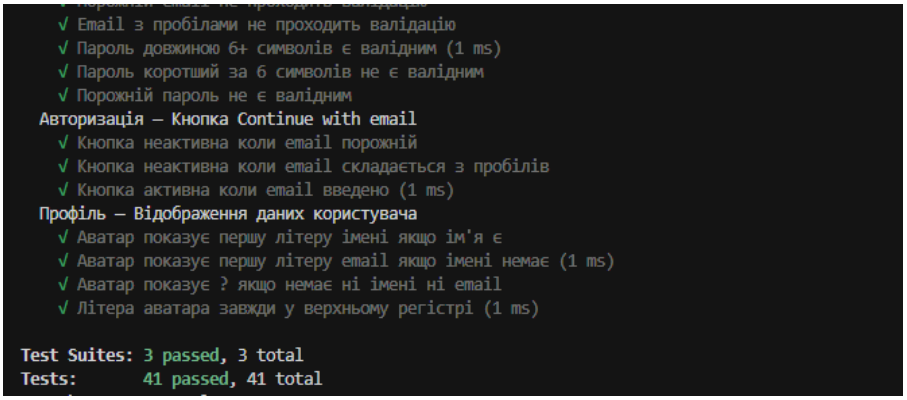


Рисунок 4.3 – Результати тестування авторизації

У модулі авторизації перевірено:

– валідацію email та пароля;
– реакцію системи на некоректні дані;
– логіку активності кнопок;
– відображення даних користувача.

Зведені результати тестування наведено у таблиці 4.1.

Таблиця 4.1 – Результати функціонального тестування

Модуль	**Кількість тестів**	**Успішно пройдено**	**Результат**
Нотатки	20	20	Успішно
Аналітика	15	15	Успішно
Авторизація	6	6	Успішно
Загалом	41	41	Успішно

Отримані результати свідчать про коректну роботу всіх основних модулів застосунку.

4.3 Аналіз результатів

Проведене тестування показало, що розроблений мобільний застосунок стабільно виконує всі основні функції.

Усі тести були успішно пройдені, що підтверджує правильність реалізації логіки роботи системи.

Особливу увагу було приділено перевірці граничних випадків, зокрема:

– обробці порожніх значень;

– некоректних вхідних даних;

– відсутності нотаток.

Результати показали, що система коректно реагує на такі ситуації, не викликаючи помилок або збоїв у роботі.

Застосування автоматизованого тестування дозволило:

– підвищити надійність системи;

– зменшити кількість потенційних помилок;

– спростити процес перевірки змін у коді.

Отже, отримані результати підтверджують, що реалізовані функціональні можливості відповідають поставленим вимогам.

4.4 Оцінка ефективності

Ефективність розробленого застосунку оцінюється за такими критеріями:

– швидкість обробки даних;

– зручність використання;

– стабільність роботи;

– коректність виконання функцій.

Завдяки використанню Cloud Firestore забезпечується швидка синхронізація даних у реальному часі, що дозволяє миттєво відображати зміни у застосунку [19].

Інтерфейс застосунку забезпечує швидкий доступ до основних функцій, що знижує час взаємодії користувача із системою. Використання Kanban-дошки та аналітики підвищує ефективність роботи з нотатками та дозволяє краще організовувати задачі.

Проведене тестування підтвердило стабільність роботи системи навіть при активній взаємодії з різними модулями. Усі функції працюють коректно, а час відгуку інтерфейсу залишається мінімальним.

Розроблений застосунок є ефективним інструментом для ведення нотаток, який поєднує зручність використання, функціональність та сучасні технології.

4.5 Висновки до розділу 4

У четвертому розділі було проведено тестування та аналіз роботи розробленого мобільного застосунку. У процесі дослідження застосовано підхід модульного тестування, що дозволив перевірити коректність функціонування окремих компонентів системи та забезпечити їх стабільну взаємодію.

Функціональне тестування охопило основні модулі застосунку, зокрема роботу з нотатками, аналітичний блок та систему авторизації. Результати тестування підтвердили правильність реалізації ключових алгоритмів, коректну обробку вхідних даних та відповідність поведінки системи очікуваним сценаріям використання.

Аналіз отриманих результатів показав, що застосунок стабільно працює як у стандартних умовах, так і при обробці граничних випадків. Система коректно реагує на помилки введення, відсутність даних та інші нестандартні ситуації, що свідчить про надійність реалізованої логіки.

Оцінка ефективності роботи застосунку засвідчила, що використані технологічні рішення забезпечують швидку обробку даних, миттєву

синхронізацію та зручність взаємодії користувача з інтерфейсом. Це дозволяє розглядати розроблений застосунок як ефективний інструмент для організації та аналізу нотаток.

Отримані результати підтверджують, що система відповідає поставленим вимогам та може бути використана у практичній діяльності.

ВИСНОВКИ

У кваліфікаційній роботі вирішено завдання розробки мобільного застосунку для ведення нотаток із підтримкою хмарного зберігання, що поєднує кілька режимів роботи з даними та засоби аналітики в межах єдиної системи.

У першому розділі проведено аналіз предметної області та розглянуто наявні рішення – Google Keep, Evernote, Microsoft OneNote і Notion. Встановлено, що більшість застосунків орієнтовані на один спосіб організації даних і не забезпечують повноцінної інтеграції між різними режимами роботи. Жоден із розглянутих аналогів не поєднує простоту використання, гнучкість структури та вбудовану аналітику одночасно, що підтверджує доцільність розробки власного рішення.

У другому розділі спроєктовано архітектуру системи на основі клієнт-серверного підходу з використанням React Native, Expo та хмарної платформи Firebase. Визначено структуру бази даних у Cloud Firestore, де кожна нотатка зберігається як окремий документ у межах колекції конкретного користувача. Описано логіку роботи основних модулів та формалізовано алгоритми створення нотаток, зміни їх статусу й синхронізації даних.

У третьому розділі реалізовано всі заплановані функціональні модулі: систему авторизації через Firebase Authentication, модуль роботи з нотатками з підтримкою списку та календарного відображення, Kanban-дошку з можливістю перетягування карток між колонками, аналітичний блок із графічним представленням статистики та інтерфейс з підтримкою світлої і темної теми. Усі модулі працюють з єдиним джерелом даних, що забезпечує узгодженість інформації без її дублювання.

У четвертому розділі проведено модульне тестування з використанням Jest, яке охопило 41 тест-кейс для трьох основних модулів. Усі тести пройдено успішно, підтверджено коректну обробку вхідних даних, правильність роботи алгоритмів і стабільну поведінку системи в граничних умовах.

Практичне значення роботи полягає в тому, що розроблений застосунок може застосовуватись як інструмент особистої продуктивності: для ведення записів, планування задач у форматі Kanban та відстеження власної активності. Використання хмарного зберігання забезпечує доступ до даних з різних пристроїв та їх синхронізацію в реальному часі.

ПЕРЕЛІК ПОСИЛАНЬ

1. Cloud Firestore Documentation. URL: https://firebase.google.com/docs/firestore (дата звернення: 27.02.2026).

2. Firebase Authentication Documentation. URL: https://firebase.google.com/docs/auth (дата звернення: 27.02.2026).

3. Stair R. M., Reynolds G. W. Principles of Information Systems. Boston : Cengage Learning, 2020. 704 p.

4. Dennis A., Wixom B. H., Roth R. M. Systems Analysis and Design : 7th ed. Hoboken : Wiley, 2021. 656 p.

5. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner’s Approach : 9th ed. New York : McGraw-Hill, 2020. 992 p.

6. Coronel C., Morris S. Database Systems: Design, Implementation, & Management. Boston : Cengage Learning, 2019. 832 p.

7. Lee V., Schneider H., Schell R. Mobile Applications: Architecture, Design and Development. Harlow : Pearson, 2021. 624 p.

8. Eisenman B. Learning React Native : 2nd ed. Sebastopol : O’Reilly Media, 2018. 338 p.

9. Laudon K. C., Laudon J. P. Management Information Systems : 17th ed. Harlow : Pearson, 2021. 672 p.

10. Krug S. Don’t Make Me Think, Revisited : 3rd ed. Berkeley : New Riders, 2014. 216 p.

11. Richards M., Ford N. Fundamentals of Software Architecture: An Engineering Approach. Sebastopol : O’Reilly Media, 2020. 432 p.

12. Sommerville I. Software Engineering : 11th ed. Boston : Pearson, 2020. 816 p.

13. Martin R. C. Clean Architecture: A Craftsman’s Guide to Software Structure and Design. Boston : Prentice Hall, 2017. 432 p.

14. Fowler M. Patterns of Enterprise Application Architecture. Boston : Addison-Wesley, 2002. 560 p.

15. Elmasri R., Navathe S. Fundamentals of Database Systems : 7th ed. Boston : Pearson, 2016. 1272 p.

16. Kleppmann M. Designing Data-Intensive Applications. Sebastopol : O’Reilly Media, 2017. 616 p.

17. Banks A., Porcello E. Learning React: Modern Patterns for Developing React Apps : 2nd ed. Sebastopol : O’Reilly Media, 2020. 310 p.

18. Dabit N. React Native in Action. Shelter Island : Manning Publications, 2021. 440 p.

19. Firebase Documentation. URL: https://firebase.google.com/docs (дата звернення: 12.03.2026).

20. React Native Documentation. URL: https://reactnative.dev (дата звернення: 12.03.2026).

ДОДАТОК А

Графічні матеріали

А.1 Схема загальної архітектури мобільного застосунку

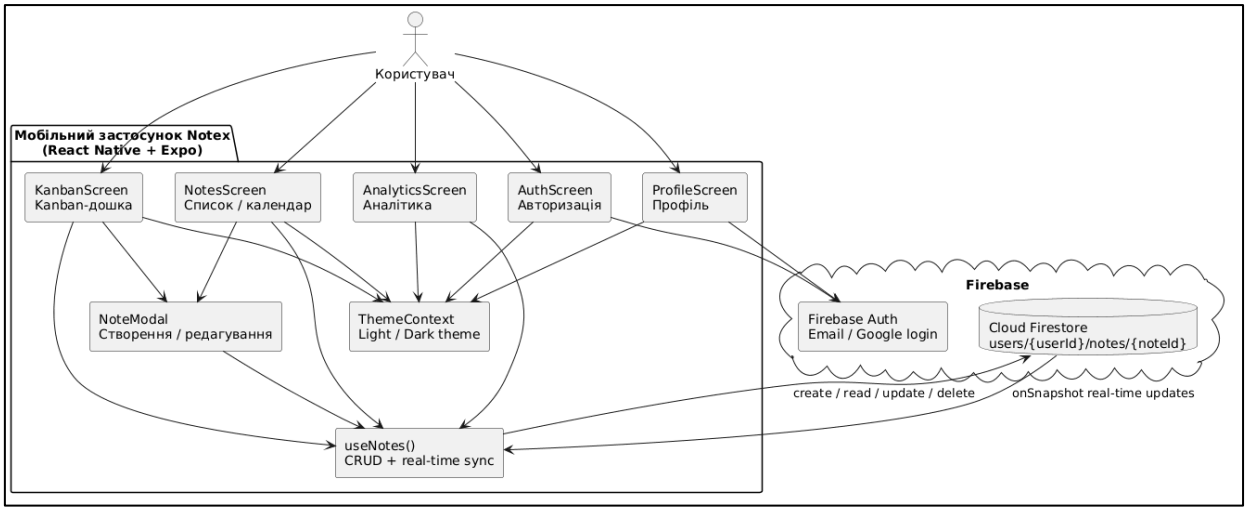


Рисунок А.1 – Загальна архітектура мобільного застосунку Notex

А.2 Структура бази даних у середовищі Firebase

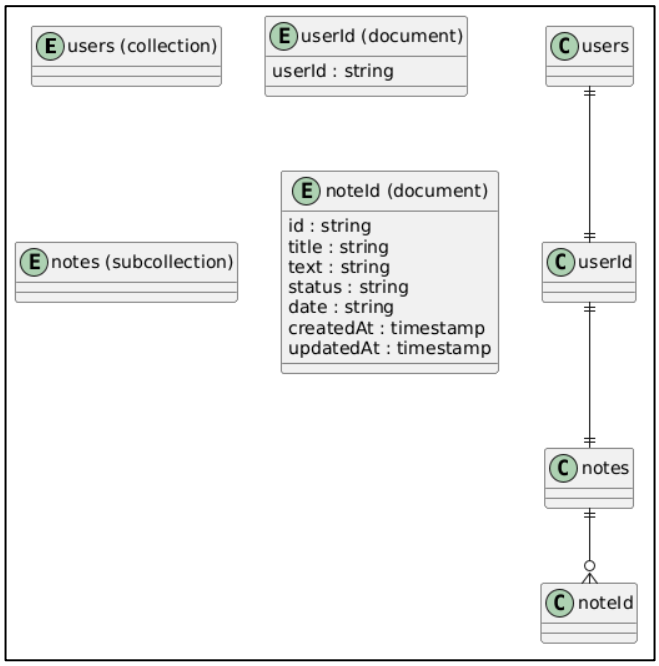


Рисунок А.2 – Структура бази даних у Cloud Firestore

А.3 Загальний процес роботи системи

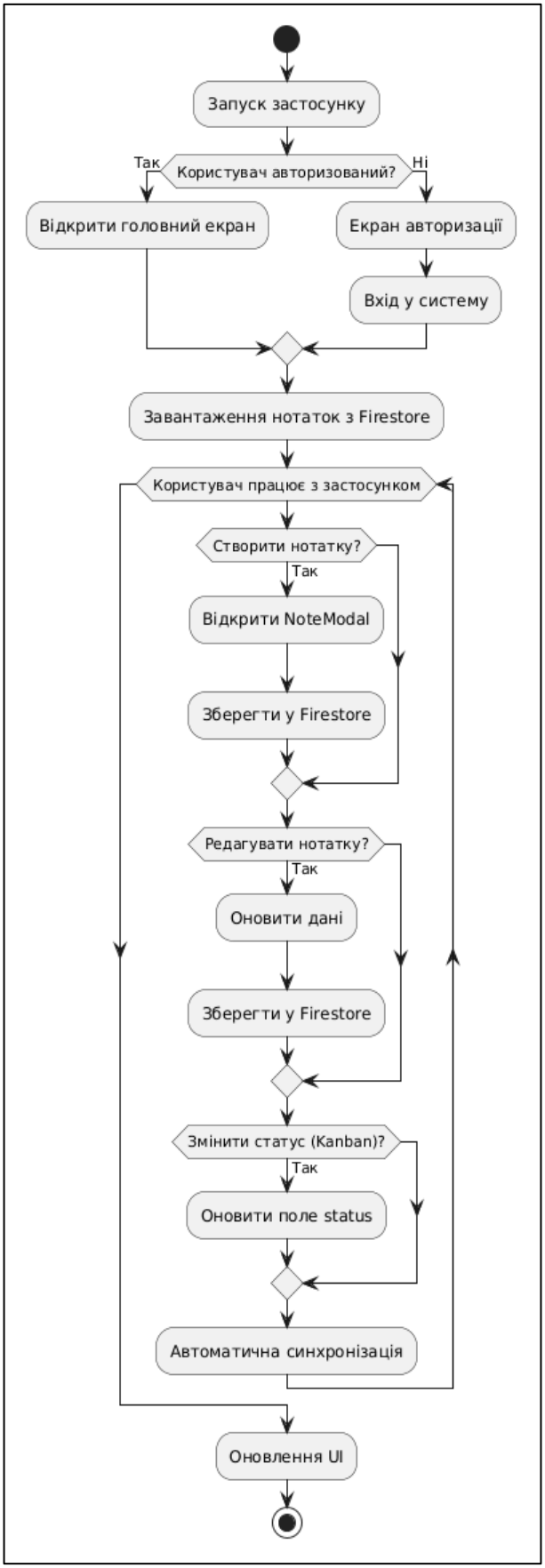


Рисунок А.3 – Логіка роботи мобільного застосунку

А.4 Загальну схему алгоритму створення нотатки



Рисунок А.4 – Алгоритм створення нотатки