

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:

Завідувач кафедри

_____ Олександр Коваль

«___» _____ 2021 р.

Дипломна робота

на здобуття ступеня бакалавра

спеціальності 121 «Інженерія програмного забезпечення»

освітньої програми «Інженерія програмного забезпечення розподілених систем»

на тему: «Комп'ютерні засоби організації та видання збірників матеріалів наукової конференції»

Виконав:

студент IV курсу, групи ПІ-72

Антонюк Андрій Миколайович

Керівник:

кандидат фіз.-мат. наук, доцент,

Карпенко Станіслав Григорович

Рецензент:

Кандидат технічних наук, доцент,

Рачинський Артур Юрійович

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ – 2021 року

**Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”**

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

спеціальності 121 «Інженерія програмного забезпечення»

освітня програма «Інженерія програмного забезпечення розподілених систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Олександр Коваль
(підпис)

” ” 2021р.

ЗАВДАННЯ

на дипломну роботу студенту

Антонюку Андрію Миколайовичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Комп'ютерні засоби організації та видання збірників матеріалів наукової конференції

керівник роботи Карпенко Станіслав Григорович, кандидат фіз.-мат. наук, доцент

(прізвище, ім'я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від "24" травня 2021р. № **1267с**

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи мова програмування Dart, фреймворк Flutter, база даних Firebase, середовище розробки Android studio

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Провести аналіз поточних рішень, які можуть бути використані для зберігання даних кафедри та університету. Реалізувати мобільний додаток, який дозволить хмарно зберігати, організовувати та видавати матеріали наукової конференції, дипломних та курсових робіт.

5. Перелік ілюстративного матеріалу

Вступ, постановка задачі, проблема та мета, інформаційні ресурси, аналіз поточного положення, дослідження існуючих рішень, порівняння основних популярних хмарних сховищ, технології організації файлових сховищ, вибір інструментів розробки, figma, flutter, dart, firebase, користувацьки інтерфейс, висновки.

6. Консультанти розділів роботи

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

7. Дата видачі завдання ”__” _____ 202__ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	09.10.2020	
2.	Вивчення та аналіз задачі	20.02.2021	
3.	Розробка архітектури та загальної структури системи	20.04.2021	
4.	Розробка структур окремих підсистем	05.05.2021	
5.	Програмна реалізація системи	16.05.2021	
6.	Оформлення пояснювальної записки	26.05.2021	
7.	Захист програмного продукту	12.05.2021	
8.	Передзахист	26.05.2021	
9.	Захист	14.06.2021	

Студент _____
(підпис)

Антонюк А.М. _____
(прізвище та ініціали.)

Керівник роботи _____
(підпис)

Карпенко С.Г. _____
(прізвище та ініціали.)

АНОТАЦІЯ

до бакалаврської дипломної роботи Антонюка Андрія Миколайовича

на тему «Комп'ютерні засоби організації та видання збірників
матеріалів наукової конференції»

У дипломній роботі досліджено та вирішено проблему зберігання даних наукових конференцій факультету.

Метою роботи є аналіз існуючих способів організації, безпечного зберігання та управління великими обсягами даних, і реалізація системи, що дозволяє безпечно зберігати, шукати та передавати інформацію.

В процесі роботи було визначено проблематику та особливості зберігання та структурування інформації. На основі цих матеріалів було висунуто певні критерії до розроблюваного додатку.

Результатом дипломної роботи є мобільний додаток, котрий дозволяє зберігати наукові матеріали та швидко їх шукати. Додаток може бути використано не лише для роботи з матеріалами наукових конференцій, а і для курсових, дипломних, магістерських робіт.

Загальний обсяг роботи 60 сторінок.

Ключові слова: зберігання даних, пошук даних, бази даних, наукові матеріали, документообіг, Flutter, Dart.

ANNOTATION

on Andrii Antoniuk bachelor's degree

thesis: “ Computer tools for organizing and publishing collections of scientific conference materials ”

In the thesis the problem of data storage of scientific conferences of the faculty is investigated and solved.

The aim of the work is to analyze the existing methods of organization, secure storage and management of large amounts of data, and the implementation of a system that allows you to securely store, search and transmit information.

In the course of work the problems and features of storage and structuring of the information were defined. Based on these materials, certain criteria were put forward for the developed application.

The result of the thesis is a mobile application that allows you to save scientific materials and quickly search for them. The application can be used not only to work with materials of scientific conferences, but also for course, diploma, master's theses.

The total volume of the work is 60 pages.

Keywords: data storage, data retrieval, databases, scientific materials, document management, Flutter, Dart.

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ПОЗНАЧЕНЬ	8
ВСТУП	9
1 ПОСТАНОВКА ЗАДАЧІ	11
1.1 Проблема та мета	11
1.2 Інформаційні ресурси	13
1.3. Аналіз поточного положення	17
2 ДОСЛІДЖЕННЯ ІСНУЮЧИХ РІШЕНЬ	20
2.1 Порівняння основних популярних хмарних сховищ	20
2.1.1 Microsoft OneDrive.....	20
2.1.2 Яндекс.Диск	21
2.1.3 Google Диск	22
2.1.4 Dropbox	23
2.1.5 Mega	24
2.1.6 Mail.ru.....	25
2.1.7 iCloud.....	26
2.1.8 Vox.....	27
2.1.9 pCloud.....	28
2.1.10 MediaFire.....	29
2.1.11 Порівняння за об'ємом.....	29
2.2 Технології організації файлових сховищ	30
2.2.1 NAS	31
2.2.2 SAN	33
2.2.3 DAS	34
2.2.4 Порівняння NAS, DAS та SAN.....	35
3 ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ	37
3.1 Figma	37
3.2 Flutter	38
3.3 Dart	40
3.4 Firebase	41
4 КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС	46
4.1 Модуль авторизації та реєстрації	46
4.2 Головна сторінка	49
4.3 Сторінка матеріалу	50

4.4 Особистий кабінет користувача	52
4.5 Список користувачів	55
4.5 Sharing	55
ВИСНОВКИ	57
ПЕРЕЛІК ПОСИЛАНЬ.....	59
ДОДАТОК А.....	61
ДОДАТОК Б	63
ДОДОТОК В.....	85

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ПОЗНАЧЕНЬ

ВНЗ	Вищий Навчальний Заклад	
Кафедра АПЕПС	Кафедра автоматизації проектування енергетичних процесів і систем	
HDD	Жорсткий магнітний диск	
БД	База даних	
API	Прикладний програмний інтерфейс	
Хмара	Хмарне сховище	
UI	User interface	(користувацький інтерфейс)
UX	User experience	(користувацький досвід).

ВСТУП

На даний момент часу інформація стає однією з найбільших цінностей для будь-якої сфери діяльності в ХХІ столітті.

Однією з найбільших проблем є випадкова втрата та несанкціоноване поширення інформації. Можливість управління даними є важливим атрибутом для сховищ даних. Проблема передачі, зберігання, захисту та управління даними зростає в умовах зростання їх обсягу.

Метою роботи є аналіз існуючих способів організації, безпечного зберігання та управління великими обсягами даних, і реалізація системи, що дозволяє безпечно зберігати, шукати та передавати інформацію.

Мною були досліджені особливості роботи з даними кафедри АПЕПС.

Студентські конференції на теплоенергетичному факультеті продовжуються вже дев'ятнадцять років. За цей час накопичилося багато томів тез конференції. У більшості енергійних студентів декілька тез конференції, і в різних збірниках за різні роки.

На даний момент більшість студентських робіт зберігаються на дисках, твердих накопичувачах, HDD накопичувачах студентів та викладачів. В будь-який момент ці дані можуть бути втрачені / видалені / загублені. Також, у зв'язку з розміщенням на різних девайсах, на пошук матеріалу може піти велика кількість часу.

Звідси виникає потреба в інструменті для зберігання та структурування цієї інформації. Також часто потрібно швидко знайти в збірниках свої тези і визначити посилання на них. Така потреба постає гостріше для викладачів, які брали участь в конференціях з різними студентами і за велику кількість років. Такі проблеми можуть виникати не тільки для студентських конференцій теплоенергетичного факультету, а й для збірників інших конференцій. Мета дипломної роботи - реалізація хмарної системи, котра дозволить зберігати та структурувати в собі файли та документи наукових конференцій.

Розроблений програмний продукт призначений для викладачів та студентів, які приймають участь у наукових конференціях. Також програмний продукт може бути використано для зберігання курсових та дипломних робіт студентів.

1 ПОСТАНОВКА ЗАДАЧІ

Даний розділ надає вхідні дані і необхідні відомості для короткого огляду і розуміння проекту, формування мети та завдань, які необхідно виконати для її реалізації.

1.1 Проблема та мета

Зберігання даних будь-якої кафедри вищого навчального закладу є одним із найскладніших процесів, котрий вимагає великих часових затрат.

При цьому важливо мати на увазі, що кафедра зобов'язана зберігати всі наукові роботи студентів. Наявність у повному об'ємі правильно структурованих робіт є одним із важливих показників успішної роботи кафедри в цілому. Контрольні заходи міністерства освіти також включають у себе вибірку перевірку зберігання робіт.

Основне навантаження на підтримку процесу зберігання матеріалів випадає на методистів та самих викладачів. Як правило, кожен документ зберігається на кафедрі у двох формах: електронна - файл на жорсткому диску комп'ютера кафедри / викладача та на паперовому, в роздрукованому вигляді, в конкретній папці документів відповідно до прийнятої класифікації. При цьому, число таких папок просто необмежене.

Пошук матеріалів, особливо в паперовому вигляді є ще важчим завданням. Особливо важким є пошук інформації в інших джерелах, що мають логічні та інформаційні зв'язки з поточним документом[1].

Найбільш ефективним рішенням вищезазначених проблем є розробка інформаційної системи, яка буде містити в собі всі матеріали наукових конференцій за всі роки. Систему може бути розширено та використано для зберігання наукових, курсових та дипломних робіт.

Відповідно до теми, дипломна робота вивчає питання автоматизації зберігання та пошуку інформації, котра пов'язана з кафедрою та містить наукову цінність, чим і зумовлена актуальність теми.

На даний момент тема зберігання наукових матеріалів навчальних проєктів розроблена у значній мірі. Однак документи, як правило, входять в єдину систему управління інформацією всього навчального закладу, як один із його елементів.

Очевидно, що такий підхід породжує собою велику кількість бюрократії, адже викладачеві потрібно подавати інформацію у головний корпус та чекати її публікації. Це особливо не ефективно в весняно-літній період, коли кількість таких запитів збільшується у зв'язку з великою кількістю наукових конференцій та дипломних захистів.

Це дозволяє зробити висновок про те, що ступінь розробленості теми не можна вважати повною і, відповідно, можна продовжити дослідження в цій галузі.

Ціллю цієї роботи є розробка інформаційної системи організації та видання збірників матеріалів наукової конференції

Задачі роботи. Для досягнення зазначеної цілі в рамках дипломної роботи потрібно вирішити ряд завдань:

- розкрити інформаційний та функціональний аналіз предметної області
- обґрунтувати вибір проектних рішень;
- спроектувати схему додатку;
- спроектувати базу даних;
- розробити сам інтерфейс додатку;

Розроблене програмне забезпечення має відповідати наступним критеріям:

- доступ з мобільного пристрою 24/7;
- хмарний спосіб зберігання даних;
- зручність у використанні;
- зрозумілий для користувачів інтерфейс;
- швидкість у виконанні;
- наявність захисту від некоректного вводу даних;

- наявність особистого кабінету автора та модератора;
- система гнучкого пошуку;
- система фільтрації матеріалів за певними критеріями;
- можливість коментування матеріалів;
- можливість редагування та видалення раніше завантажених матеріалів користувачем з типом доступу «Адміністратор».

Наукова та практична цінність роботи. Наукова цінність роботи полягає у результатах, отриманих при проведенні теоретичних досліджень на першому етапі робіт, у тому числі з використанням літературних джерел.

Практична цінність роботи входить у розробку успішно функціонуючої інформаційної системи для автоматизації зберігання інформації нашої кафедри.

1.2 Інформаційні ресурси

В умовах інформатизації всіх служб життя суспільства пріоритетне значення має використання інформаційних та телекомунікаційних технологій у сфері освіти. Інформатизація освіти має два напрямки: введення інформаційних технологій безпосередньо в процес навчання та інформатизації систем управління освітою. В умовах модернізації системи освіти, оновлення змісту, методів та форм організації навчального процесу в чисельності пріоритетних установ створення освітньої інформаційної середовища входить удосконалення управління на основі інформаційних технологій[2].

Інформаційна система - взаємозв'язана сукупність засобів, методів та персоналу, що використовуються для зберігання, обробки та видачі інформації в інтересах досягнення цільової цілі.

Процеси, що забезпечують роботу інформаційної системи будь-якого призначення, складаються з наступних блоків:

- 1) введення інформації із зовнішніх або внутрішніх джерел;
- 2) обробка вхідної інформації та представлення її в зручному вигляді;
- 3) виведення інформації для презентацій користувача або передачі у другу систему;
- 4) зворотній зв'язок - це інформація, перероблена людьми навчального закладу для корекції вхідної інформації.

Центральним ядром інформаційної системи управління, за допомогою якого здійснюється її автоматизація, є програма. Можливі два способи взаємодії між програмою, об'єктом та органом управління[3].

У першому випадку програма використовується, як правило, для вирішення окремих періодично повторюваних трудових завдань. Збір інформації ведеться вручну, також здійснюється підготовка документів з керуючими діями. Подібна система може бути названа систематичною обробкою даних. Доступ користувачів до системи обробки даних частіше за все приводить до оновлення інформації. Виведення інформації у такому випадку може бути відсутнім або містити проміжні результати.

У другому випадку основна інформація про стан керованого об'єкта збирається автоматично. Програма конвертує вхідну інформацію показує у іншому вигляді вихідні дані.

В автоматизованих системах управління вихідна інформація може бути представлена у двох виглядах:

- перероблений, впорядкований список даних про керований об'єкт. На основі цих даних людина (або групи людей) приймає рішення щодо характеру впливу на об'єкт. Це властиво системі обробки даних, але не власно автоматизованій системі;
- сукупність рекомендацій (варіантів рішень) щодо характеру впливу на керований об'єкт. Фінальне рішення в даному випадку приймає людина. Така реалізація найбільш типова для автоматизованої системи управління[4].

Інформаційна система управління визначається наступними властивостями:

- 1) будь-яка інформаційна система може бути аналізованою, побудованою та керованою на основі загальних принципів побудови системи;
- 2) інформаційна система є динамічною та такою, що розвивається;
- 3) при побудові інформаційної системи необхідно використовувати системний підхід;
- 4) інформаційну систему слід сприймати як персонально-комп'ютерну систему обробки інформації.

Впровадження інформаційних систем може сприяти:

- забезпеченню достовірності інформації;
- звільненню користувача від рутинної роботи за рахунок її автоматизації;
- удосконаленню структури потоків інформації та систем зберігання даних;
- заміні паперових носіїв даних на електронні, що призводить до раціональнішої організації переробки інформації на комп'ютері та зменшенню витрат паперу;
- отриманню більш раціональних варіантів рішення задач за рахунок використання математичних методів та інтелектуальних систем;

В сучасній системі освіти перші інформаційні системи управління створюються ще в 60-ті роки.

Можна виділити наступні рівні управлінської діяльності з використанням програм у системі освіти:

- 1) управління навчанням та розвитком окремого навчального закладу;
- 2) управління навчальним процесом у рамках одного навчального заведення;
- 3) управління роботою груп різних навчальних закладів;
- 4) управління навчальними закладами за територіальним принципом;

На першому рівні завдання управління співпадають значною мірою із завданнями навчання за допомогою комп'ютерів[5].

Сучасний ВНЗ є такою організацією, яка в силу власних потреб зобов'язана бути островом майбутнього інформаційного суспільства. Інтеграція сучасної інформаційно-керуючої системи дозволяє забезпечити підвищення якості управління вузом та гарантувати керівникам різних рівнів своєчасне отримання та первинну обробку всіх їхніх інформаційних потоків. Результатом введення інформаційної системи управління вузом є підвищення якості прийнятих рішень, котрі безпосередньо впливають на ефективність роботи вузів[6].

Інтеграція автоматизованої системи управління в діяльності навчальних закладів дозволяє вирішити наступні завдання:

- отримати прозорість усіх процесів управління навчальним закладом;
- планувати навчальну навантаження на викладачів, контролювати її виконання та складати розклад занять;
- контролювати успішність, відвідуваність занять та оплату за навчання з моменту підготовки до випуску студента;
- підвищити контроль якості виявлених освітніх послуг студента;
- автоматизувати документообіг з підготовкою всієї необхідної навчальної документації та контролювати виконавчу дисципліну співробітників, що навчаються в організаціях навчального процесу;
- реалізовувати вивчення окремих навчальних дисциплін або всього навчального плану із застосуванням дистанційних технологій.

Основними вимогами до нашої інформаційної системи є:

1. багатофункціональність: вирішення основних завдань з управління та зберіганням інформації у рамках єдиної інформаційної системи з можливістю

швидкого доступу до необхідних даних для будь-якого учасника відповідно до його індивідуальних прав;

2. модульна структура: у випадку реалізації додаткового функціоналу – він реалізується модульно, без порушення попередньо реалізованого функціоналу;

3. інтегрованість: система може бути інтегрована з іншою системою за допомогою API;

4. Відкритість: можливість доступу у будь-який момент часу.

Для успішної організації управлінської діяльності ВНЗ необхідно не лише аналізувати навчальний процес, але і виявляти найбільш значущі проблеми, знаходити ефективні шляхи їх вирішення. При цьому необхідна оптимізація управлінської діяльності вузів у зв'язку із застосуванням вимог до якісного навчання та в умовах постійного збільшення обсягу інформації[7].

1.3. Аналіз поточного положення

В даний час більшість наукових матеріалів зберігаються на кафедрі у двох форматах: паперовому та електронному. Тому, викладачі витрачають велику кількість часу та ресурсів на структурування та зберігання інформації на своїх пристроях. Набагато кращим варіантом було б самостійне завантаження студентами наукової інформації (а викладач міг би модерувати цю інформацію).

Також, частою проблемою, яка постає з кожним роком все частіше і частіше (особливо для матеріалів, котрим більше 10 років) – це повна безповоротна втрата матеріалів наукових та науково дослідних конференцій, бакалаврських, курсових та магістерських робіт.

Причина цього – зберігання на HDD накопичувачах та дисках, котрі з часом стають непридатними до використання, втрачаються, горять, ламаються.

Автоматизація цього процесу є важливим засобом вирішення проблеми підвищення ефективності кафедри.

Управління інформаційними ресурсами надає для діяльності любого навчального закладу особисте значення. У сучасному світі установи

зіштовхуються з необхідністю обробки колосального обсягу інформації. У більшості закладів із складною структурою важливе значення має рівень організації взаємодії підрозділів та порядок обміну інформацією. Більша частина інформації передається у письмовому вигляді на паперовому носії.

Важною характеристикою інформаційної системи для зберігання інформації є об'єм файлів, котрі будуть в ній зберігатися. Під об'єктом розуміється кількість документів, що були додані кафедрою протягом певного періоду часу (як правило, року).

Поруч з організацією зберігання інформації стоїть задача обміну цієї інформацією.

Наша інформаційна система - це сукупність способів та заходів навчання та систематизація документів з метою їх пошуку та використання в поточній діяльності кафедри.

У сучасній науці більшу роль грає документування наукових досліджень. Наукові статті дозволяють систематизувати результати певного експерименту чи дослідження. Опубліковані статті дозволяють ефективно ділитися результатами зробленої роботи з колегами та студентами, що працюють в цій, або схожих наукових сферах, областях. Також велику ціну представляють тези наукових конференцій, за якими написано дипломні роботи. При великих обсягах проаналізованої інформації необхідно організувати ефективна здатність зберігання тез. Тези можуть бути використані для розвитку та наукового виготовлення, за яким було написана стаття, або для підтвердження результатів виконаної роботи. У зв'язку з цим учбові заклади все частіше зіштовхуються з проблемою неефективної роботи з публікаціями із-за високої складності пошуку інформації в публікаціях, великих затрат на аналіз опублікованих статей.

Додатковою проблемою є комплексна система для комплексної обробки інформації. Для успішного відтворення експериментально вченим необхідно знати вихідний набір даних, використовувані методи дослідження, проміжні дані і результати дослідження. Всі ці дані разом з публікаціями авторів вимагають

ефективної обробки. Крім ефективного зберігання потрібно забезпечити пошук з метою спрощення доступу до результатів минулих наукових конференцій.

Окрім спрощення роботи з результатами попередніх наукових конференцій необхідно розробити метод стійкого зберігання для наступних наукових та науково-дослідних конференцій.

2 ДОСЛІДЖЕННЯ ІСНУЮЧИХ РІШЕНЬ

2.1 Порівняння основних популярних хмарних сховищ

Одним із варіантів зберігання даних – є зберігання даних на популярних хмарних сховищах.

Хмарне сховище, або хмара - це онлайн-сервер, який дозволяє користуватися завантаженими в нього файлами з будь-якої точки світу.

В хмарі можна копіювати важливі документи, спільно працювати і поширювати інформацію великого обсягу. Хмарне сховище дає можливість ділитися файлами за посиланням[8].

Більшість з хмарних сховищ мають підвищені ліміти для університетів та студентів.

В залежності від цілей можна визначити такі параметри для порівняння хмарних сховищ:

- безкоштовний обсяг сховища;
- вартість додаткового простору і його обмеження;
- максимальний обсяг завантаження;
- список форматів файлів, що підтримують попередній перегляд;
- редагування файлів на диску;
- робота з різних пристроїв під одним аккаунтом;
- шифрування;
- спільний доступ до файлів і папок.

2.1.1 Microsoft OneDrive

Безкоштовний обсяг: 5 Гбайт;

Перегляд і редагування файлів: є, через пакет Microsoft Office Online і Microsoft Office 365;

Для кого: є індивідуальні, сімейні і корпоративні тарифи;

Вартість додаткового місця: 1 ТБ за \$ 69,99 на рік для персонального користування, 6 Тбайт за \$ 99,99 на рік при оформленні сімейної підписки на Office 365. Для корпоративних тарифів вартість від \$ 5 в місяць за одного користувача.

Мови: російська, англійська, французька, українська, португальська, німецька та інші;

Сайт: <https://www.microsoft.com/ru-ru/microsoft-365/onedrive/online-cloud-storage>;

OneDrive підтримує безліч мов і дозволяє збільшувати сховище від 5 до 100 Гбайт. Придбати додатковий простір для бізнесу або сім'ї можливо тільки через покупку Microsoft 365 разом з додатковими програмами і функціями.

Через десктопну програму OneDrive можна переносити файли в хмару перетягуванням.

Функції:

- управління рівнем доступу;
- додатковий захист через особистий сейф: подвійна аутентифікація для входу;
- можливість перегляду обраних файлів без доступу до інтернету;
- використання сховища через веб-версію, мобільне і десктоп-додатки, підтримка XBOX.

Особливості:

- розмір завантаження не більше 15 Гбайт;
- важка процедура реєстрації бізнес-тарифу і покупки;
- безліч додаткових функцій крім зберігання;

2.1.2 Яндекс.Диск

Безкоштовний обсяг: 10 гбайт;

Перегляд і редагування файлів. є, через пакет Microsoft Office Online;

Для кого: є індивідуальні та корпоративні тарифи;

Вартість додаткового місця: індивідуально можна придбати від 100 Гбайт від \$ 1,4 в місяць до 3 Тбайт від \$ 10,5 на місяць. На корпоративному тарифах - 30 днів безкоштовно по 10 Тбайт на співробітника, далі від \$ 2 за користувача в місяць;

Мови: російська;

Сайт: <https://disk.yandex.ru/client/disk;>

Яндекс.Диск - повністю російськомовний сервіс для зберігання та обміну інформацією. Має веб, десктоп і мобільну версію. Яндекс. Диск синхронізує завантажені файли зі смартфоном і додатком на комп'ютері.

Функції:

- безлімітне автозавантаження фотографій і відео з телефону;
- управління рівнями доступу;
- перегляд історії змін;
- загальні файли з інших дисків не займають місця на акаунті;
- використання сховища через веб-версію, десктоп- і мобільні додатки;

Особливості:

- реклама в безкоштовних тарифах;
- розмір завантаження не більше 50 Гбайт.

2.1.3 Google Диск

Безкоштовний обсяг: 15 Гбайт;

Перегляд і редагування файлів: є, через пакет Google Docs;

Для кого: є індивідуальні та корпоративні тарифи;

Вартість додаткового місця: для індивідуального користування можна придбати від 100 Гбайт за \$ 1,95 на місяць до 30 Тбайт за \$ 295 в місяць. У корпоративному тарифі від \$ 10 за користувача в місяць - 100 Гбайт на співробітника;

Мови: російська, англійська, німецька, українська та інші. Всього понад 100 мов;

Сайт: <https://www.google.com/drive/>;

Google Диск - саме містке сховище з усіх перерахованих. Максимально в нього можна завантажити до 30 Тбайт інформації. Редагувати файли можна навіть без мережі: дані зберігаються автоматично, після того, як з'явиться інтернет. Рівні доступу дозволяють керувати обміном папок і файлів і встановити заборону на копіювання важливих документів. Також Google Диск допомагає шукати файли за назвою і переміщувати з папками.

Функції:

- попередній перегляд і редагування більшості форматів через Google Docs;
- використання через веб-версію, десктопних і мобільних додатків;
- до окремих файлів можна підключити офлайн-доступ;
- безлімітне зберігання фотографій через додаток на мобільних пристроях;
- зберігання історії змін файлів;
- шифрування при передачі і зберіганні даних;

Особливості:

- дисковий простір включає файли Gmail пошти;
- розмір завантаження - не більше 5 Тбайт;

2.1.4 Dropbox

Безкоштовний обсяг: 2 Гбайт;

Перегляд і редагування файлів: є, через пакети Microsoft Office Online і Dropbox Paper;

Для кого: є індивідуальні та корпоративні тарифи;

Вартість додаткового місця: індивідуальний тарифний план стартує від \$ 9,99 на місяць за 2 Тбайта простору і \$ 16,58 в місяць за 3 Тбайта. Корпоративний тариф передбачає 30 днів безкоштовного періоду, далі від \$ 16,58 в місяць за 3 Тбайта в обліковому записі;

Мови: 22 мови серед яких російська, англійська, німецька, українська та інші;

Сайт: <https://www.dropbox.com/>;

Dropbox – це хмара з невеликим безкоштовним сховищем, але можливістю покупки необмеженого простору в хмарі. Dropbox зберігає історію змін файлів, допомагає керувати рівнями доступу та коментувати матеріали відразу в сховищі.

Dropbox має широкі можливості по спільній роботі з файлами: запрошення користувачів, коментування, відстеження історії змін.

Функції:

- зберігання історії змін файлів і папок;
- необмежений розмір завантажуваних файлів (обмеження тільки в веб-версії - до 50 Гбайт).;
- підключення сторонніх сервісів для управління командами: Trello, Slack, Jira;
- наявність реферальної програми по збільшенню дискового простору за запрошеного друга;
- використання через веб-версію, десктопних і мобільних додатків;
- доступ до файлів за паролем;

Особливості:

- На безкоштовному тарифі можна синхронізувати тільки три пристрої.

2.1.5 Mega

Безкоштовний обсяг: 35 гбайт;

Перегляд і редагування файлів: є попередній перегляд, без редагування;

Для кого: є індивідуальні та корпоративні тарифи;

Вартість додаткового місця: індивідуально можна придбати від 400 Гбайт за \$ 5,63 на місяць до 16 Тбайт за \$ 33,83 в місяць. Від трьох користувачів діє бізнес-тариф за \$ 11,28 в місяць за кожного;

Мови: 18 мов серед яких російська, англійська, німецька, польська та інші;

Сайт: <https://mega.nz/>;

Головна особливість сховища Mega - наскрізне шифрування даних для всіх користувачів. Це означає, що хмара не може побачити ваші завантаження. У тарифах є обмеження на обсяг передачі даних в місяць від 1 до 16 Тбайт. В Mega вибудована ієрархія папок і файлів, а також є кошик для видалених файлів

Функції:

- шифрування всіх даних навіть на безкоштовному тарифі;
- наявність програми з безкоштовного збільшення дискового простору: запрошення одного друга, установка програми на десктопній версії;
- використання через веб-версію, десктопний і мобільний додатки, розширення в браузері;
- інтеграція з поштовими клієнтами Mozilla Thunderbird;

Особливості:

- обмеження за обсягом передачі даних;

2.1.6 Mail.ru

Безкоштовний обсяг: 8 байт;

Перегляд і редагування файлів: є попередній перегляд з редагуванням через Microsoft Office Online;

Для кого: є індивідуальні та корпоративні тарифи;

Вартість додаткового місця: індивідуальний тариф починається від 128 Гбайт за \$ 2 в місяць до 4 Тбайт за \$ 37,76 в місяць. За \$ 56 в місяць починається корпоративний тариф (до 30 користувачів і до 100 Гбайт для робочих груп);

Мови: російська;

Сайт: <https://cloud.mail.ru/>;

Хмарне сховище Mail.ru підтримує перегляд фотографій, відео і прослуховування аудіофайлів. На платному тарифі доступна підтримка протоколу WebDAV для командної роботи і підключення диска як мережевого сховища.

Функції:

- гнучка тарифна сітка за обсягом дискового простору;
- використання через веб-версію, десктопний і мобільні додатки;
- робота в офлайн-режимі через десктопну програму;
- редагування файлів через Microsoft Office Online;
- інтеграція з поштовими клієнтами Mail.ru;

Особливості:

- розмір завантаження до 2 Гбайт на безкоштовному тарифі і до 32 Гбайт на платному;
- заблокований на території України.

2.1.7 iCloud

Безкоштовний обсяг: 5 гбайт;

Перегляд і редагування файлів: відсутні;

Для кого: є індивідуальні та сімейні тарифи;

Вартість додаткового місця: індивідуальна передплата від 50 Гбайт за \$ 0,99 на місяць до 2 Тбайт за \$ 8,41 на місяць. На сімейному тарифі доступні обсяги 200 Гбайт і 2 Тбайт;

Мови: більше 50 мов, серед яких російська, українська англійська, французька, польська, італійська та інші;

Сайт: <https://www.icloud.com/>;

iCloud - найзручніший спосіб для зберігання файлів користувачам iPhone і iPad. Традиційне хмарне сховище є в веб-версії і додатку для Windows.

Функції:

- редагування файлів у форматах Pages, Key і Numbers;
- синхронізація з пристроями Apple;

- використання через веб-версію і додаток для Windows;
- пошук синхронізованих iPhone і iPad через хмарне сховище;

Особливості:

- перегляд і редагування файлів в інших форматах не передбачені;

2.1.8 Box

Безкоштовний обсяг: 10 гбайт;

Перегляд і редагування файлів: є попередній перегляд з редагуванням через Microsoft Office Online;

Для кого: є індивідуальні та корпоративні тарифи;

Вартість додаткового місця: для індивідуального користування - \$ 10,16 в місяць за 100 Гбайт;

У бізнес-тарифах: +100 Гбайт за \$ 5,08 на місяць, безлімітне сховище від \$ 15,23 в місяць;

Мови: 10 мов, серед яких англійська, французька, німецька, італійська та інші;

Сайт: <https://www.box.com/en-gb/home>;

Американський сервіс Box пропонує необмежений обсяг сховища для бізнес-тарифів. Як і в Google Диск є можливість створювати в хмарі власні файли Microsoft Office або Google Docs і ділитися ними з користувачами. Також Box забезпечує резервне копіювання файлів, що завантажуються.

Функції:

- редагування файлів через Microsoft Office Online;
- налаштування пароля на окремі папки або файли;
- створення заміток, списків справ і інших простих документів через вбудований сервіс Box Notes;
- використання через веб-версію, десктопний і мобільний додатки;

Особливості:

- відсутність російської та української мовимови;

- ліміт завантаження в безкоштовному тарифі 256 Мбайт. У платному обмеження також діє від 2 до 5 Гбайт;

2.1.9 pCloud

Безкоштовний обсяг: 10 гбайт;

Перегляд і редагування файлів: є попередній перегляд текстових файлів, зображень, аудіо- та відео. Немає функції редагування;

Для кого: є індивідуальні, сімейні і корпоративні тарифи;

Вартість додаткового місця: індивідуально можна придбати 500 Гбайт за \$ 46,14 на рік або 2 Тбайт за \$ 108,19 в рік. Разовий платіж «на все життя» - від \$ 175. Для сім'ї сервіс пропонує 2 Тбайта на все життя за \$ 563. У корпоративному тарифі діє місячна або річна передплата \$ 11,27 і \$ 9,02 за користувача на місяць відповідно;

Мови: 10 мов, серед яких російська, англійська, іспанська, німецька, італійська та інші;

Сайт: <https://www.pcloud.com/ru/eu>;

pCloud - гнучкий сервіс, єдиний який дає довічний доступ до сховища за разову плату. При сімейній підписці кожен член сім'ї отримує власну відокремлену хмару.

Функції:

- можливість створення брендованого посилання для поширення файлів;
- історія зберігання файлів. Розширені можливості для відновлення видалених файлів;
- сховище надає функцію довічного доступу до сховища;
- додаткове шифрування за 4,99 € в місяць;
- використання через веб-версію, мобільний і десктопний додаток;

Особливості:

- Обмеження трафіку на скачування з публічного посиланням: в безкоштовній версії - 50 Гбайт в місяць, в платних - від 500 Гбайт в місяць.

2.1.10 MediaFire

Безкоштовний обсяг: 10 гбайт;

Перегляд і редагування файлів: є, без функції редагування;

Для кого: є індивідуальний і бізнес-тариф;

Вартість додаткового місця: збільшення сховища на 1 Тбайт на користувача \$ 7,50 на місяць або \$ 80 в місяць за бізнес-тариф до 100 користувачів;

Мови: більше 50 мов, серед яких російська, українська, іспанська, англійська, німецька та інші;

Сайт: <https://www.mediafire.com/>;

Колишній файлообмінник MediaFire надає безкоштовне сховище до 10 Гбайт. Для збільшення простору є два тарифи: індивідуальний PRO і бізнес. На окремі файли і папки можна встановлювати пароль, обмежувати скачування. Завантажувати файли на MediaFire можна з комп'ютера або за посиланням

Функції:

- захист файлів паролем;
- генерація одноразових посилань на скачування файлів;
- підтримка скачування папок;
- використання через веб-версію і мобільний додаток;

Особливості:

- при скачуванні з безкоштовного аккаунта потрібно переглянути рекламний ролик;
- розмір завантаження не більше 20 Гбайт[9];

2.1.11 Порівняння за об'ємом

Нижче, у таблиці 2.1 показано порівняння хмарних сховищ за об'ємом та ціною.

Назва хмари	Обсяг безкоштовного сховища	Стартова вартість для одного користувача, в місяць	Стартова вартість для бізнесу, в місяць
Microsoft OneDrive	5 Гбайт	\$69,99 в год за 1 Тбайт	\$5
Яндекс.Диск	10 Гбайт	\$4,22 за 1 Тбайт	\$2
Google Диск	15 Гбайт	\$19,68 за 2 Тбайт	\$10
Dropbox	2 Гбайт	\$9,99 за 2 Тбайт	\$16,58
Mega	35 Гбайт	\$11,28 за 2 Тбайт	\$11,28
Mail.ru	8 Гбайт	\$19,55 за 2 Тбайт	\$56
iCloud	5 Гбайт	\$8,41 за 2 Тбайт	-
Box	10 Гбайт	\$10,16 за 100 Гбайт	\$5,08
pCloud	10 Гбайт	\$108,19 в год за 2 Тбайт	\$11,27
MediaFire	10 Гбайт	\$7,50 за 1 Тбайт	\$80

Таблиця 2.1 – Порівняння файлових сховищ

Виходячи з вищезазначеного, можна дійти до висновку, що всі файлові сховища мають свої переваги та недоліки. Вони повністю задовольняють потреби кафедри у зберіганні інформації, проте не відповідають вимогам щодо структурування та пошуку.

Жоден із файлових сховищ не може бути кастомізовано для фільтрації матеріалів за певними параметрами. Звідси виникає потреба у розробці власного університетського додатку.

2.2 Технології організації файлових сховищ

Архітектура мережевого сховища відноситься до фізичної та концептуальної організації мережі, що забезпечує передачу даних між пристроями та серверами. Сервер забезпечує внутрішню систему для

більшості операцій на рівні підприємства та дозволяє користувачам отримувати те, що їм потрібно.

Налаштування архітектури сховища може визначати, які параметри отримуватимуть пріоритет, такі як вартість, швидкість, масштабованість або безпека.

Три основних типи систем зберігання пропонують подібні функції, але дуже різняться у виконанні. До цих типів сховища належать мережеві сховища (NAS), мережа сховищ (SAN) та пряме сховище (DAS).

2.2.1 NAS

Network-attached Storage (NAS) - це комп'ютерний сервер зберігання даних на рівні файлу, підключений до комп'ютерної мережі. Він забезпечує зберігання та доступ до даних із центрального місця для кількох авторизованих користувачів мережі та інших груп клієнтів. Ці системи зазвичай використовуються для підтримки спільних додатків, включаючи інженерні збірки програмного забезпечення, реєстрацію даних, системи електронної пошти, запис та редагування відео, бізнес-аналітику, фінансові записи, набори даних та багато іншого[10].

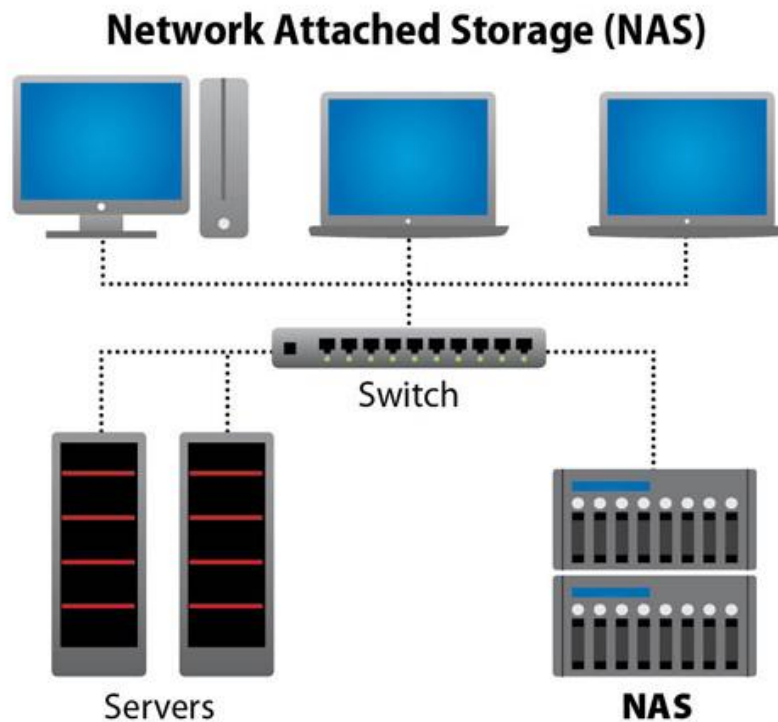
Файлові системи містяться на одному або декількох накопичувачах, часто об'єднані у резервні контейнери зберігання. Розміри NAS залежать від швидкості, масштабу та бюджетних вимог. Тому вони можуть бути будь-якими, від однієї настільної системи, аж до систем масштабування на багато петабайт. NAS використовує один або кілька протоколів доступу до файлів, які піддаються дії внутрішньої мережі. Потім вони представлені такими протоколами, як NFS (мережева файлова система) або SMB (серверний блок повідомлень), або власним високопродуктивним протоколом, що дозволяє клієнтам підключатися до NAS. Подібно до пристроїв TCP / IP та інших комп'ютерів, NAS діє як мережевий вузол, підтримує свої IP-адреси та може ефективно взаємодіяти з іншими мережевими пристроями.

Переваги та недоліки NAS

Системи NAS корисні для власників малого бізнесу, оскільки ними легко керувати; тому системний адміністратор часто навіть і не потрібен. NAS є економічно вигідним завдяки простому та безпечному резервному копіюванню даних, і це може стати наступним кроком до DAS (безпосередньо підключене сховище). Це також значно зменшує витрачений простір над іншими технологіями зберігання, такими як DAS або SAN (мережа сховищ). Крім того, системи NAS доступні постійно, що полегшує співробітникам співпрацю, реагування на замовників, спільну підтримку та розвиток проектів. Система NAS також діє як хмара, де до неї можна отримати віддалений доступ за допомогою мережевого з'єднання. Тому співробітники можуть працювати з будь-якого місця в будь-який момент часу.

Слабкі сторони NAS пов'язані з масштабом і продуктивністю. NAS обмежений своїми ресурсами, і якщо кількість користувачів, яким потрібен доступ, збільшується, пристрій NAS може не встигати відповідати на запити, що призводить до низької продуктивності та розчарувань користувачів. Системи NAS неможливо легко масштабувати або розширювати, а протоколи NAS, такі як Network File System (NFS) і SMB, недостатньо швидкі для високопродуктивних програм через низьку пропускну здатність та велику затримку. Крім того, NAS залежить від мережі, оскільки файли спільно використовуються через локальну мережу (LAN). Локальна мережа передає дані з одного місця в інше за допомогою пакетів даних, поділяючи їх на кілька сегментів і відправляючи на будь-які термінали. Однак будь-який із цих пакетів даних може бути затриманий або надісланий не по-порядку, тому трафік через локальну мережу також стає визначальним фактором.

Схема роботи NAS зображена на Малюнку 2.1.



Малюнок 2.1 – Схема роботи NAS.

2.2.2 SAN

Storage Area Network (SAN) забезпечує доступ на рівні блоку для хостів, які потребують контролю над своїми файловими системами. До розробки SAN, сервер використовував внутрішній диск як блоковий пристрій, що використовує локальну файлову систему, але пам'ять може масштабуватися лише всередині сервера, що веде до островків зберігання. SAN обходить цю проблему та забезпечує блокування пристрою в мережі. На відміну від NAS, SAN - це мережева система зберігання, яка вимагає складних технологій для підтримки продуктивності серверів. Компоненти можуть включати шлюзові пристрої, виділені комутатори (або VLAN у спільній мережі), модулі резервного копіювання стрічки, вузли контролера та місця на диску. Система SAN використовує такі протоколи, як SCSI, iSCSI та Fibre Channel.

Переваги та недоліки SAN

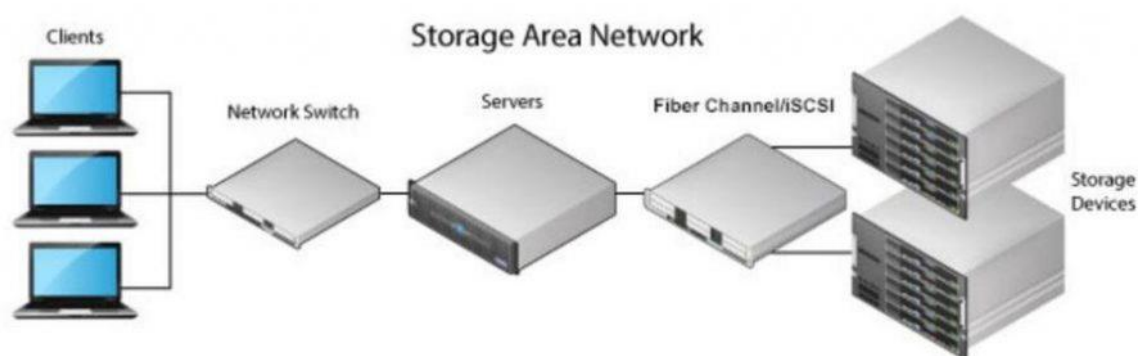
Оскільки SAN працює в окремій мережі, вона працює подібно до безпосередньо підключеного сховища. Тобто він може переміщати ресурси

локальної мережі, створюючи тим самим високошвидкісне та організоване середовище, до якого може отримати доступ операційна система кожного клієнта.

Отже, це також дозволяє швидко зберігати дані. Використання SAN означає, що пристрої, підключені до мережі, не потребують жодного локального сховища, що дозволяє масштабувати їх. Якщо є необхідність перенести SAN в інше місце, дані можна швидко відтворити, тим самим скорочуючи час на процес відновлення.

Оскільки SAN складається з складних та складних взаємопов'язаних пристроїв, він має свої недоліки через складність реалізації. Крім того, хоча пул зберігання даних розподіляється між багатьма серверами, кожен з них є дуже складним і може стати дорогою інвестицією[11].

На малюнку 2.2 зображено схему роботи SAN.



Малюнок 2.2 – Схема роботи SAN.

2.2.3 DAS

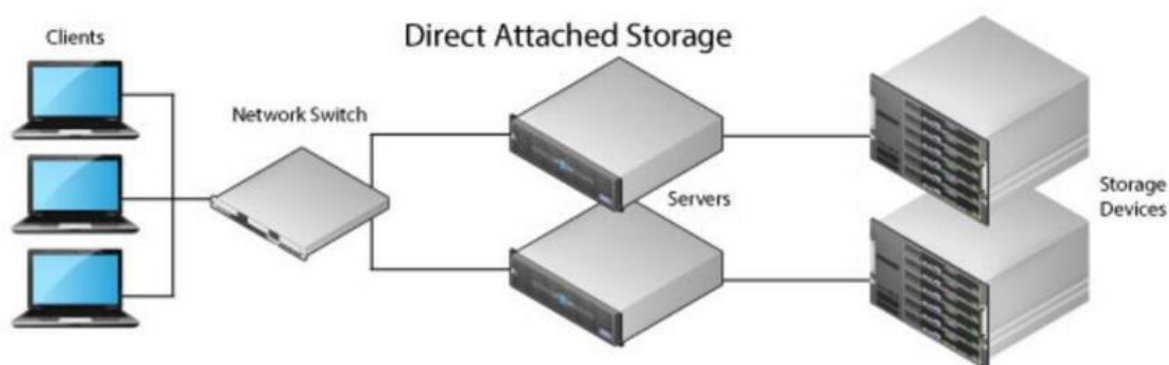
Direct Attached Storage (DAS)- це цифрове сховище, яке безпосередньо підключене до системи (тобто ПК або сервер) за допомогою внутрішнього кабелю. Система DAS вміщує кілька жорстких дисків в одному корпусі, який безпосередньо підключений до машини через HBA (адаптер шини хосту). Між цими дисковими немає мережевого пристрою (тобто комутатора, концентратора, маршрутизатора або мережевого кабелю). Для окремого користувача ПК жорсткий диск системи є стандартною формою DAS. Однак на

підприємствах окремі дисководи на одному сервері та зовнішні накопичувачі на цьому сервері підключаються або прямо, або підключаються за допомогою інтерфейсу малої комп'ютерної системи (SCSI), послідовного додаткового технологічного вкладення (SATA) або послідовного приєднаного SCSI (SAS). Іншим обмеженням DAS є те, що дані не можуть передаватися різними серверами або користувачам.

Переваги та недоліки DAS

Системи DAS прості у використанні, а технологія широко доступна. Вона забезпечує чудову продуктивність у порівнянні з SAN або NAS, але ємність зберігання розширити неможливо. Недоліками є те, що різні групи користувачів не можуть отримати доступ до даних; він є лише безпосередньо для програм, що працюють на окремому сервері або настільному комп'ютері. Крім того, DAS не включає жодного мережевого обладнання та відповідного операційного середовища, щоб забезпечити можливість спільного використання ресурсів сховища[12].

На малюнку 2.3 Зображено схему роботи DAS.



Малюнок 2.3 – Схема роботи DAS.

2.2.4 Порівняння NAS, DAS та SAN

Малі підприємства віддають перевагу системам NAS як активним, масштабованим та економічно вигідним рішенням для зберігання даних. SAN - це високоефективні багатогранні системи, які ідеально підходять для компаній, які

шукають високу ефективність та надійність. Системи DAS використовуються окремими невеликими організаціями або часто для підвищення продуктивності на підприємстві шляхом копіювання даних із системи NAS на локальні диски всередині машини DAS. Хоча це і ефективно, але керування такою системою затрачає велику кількість людських ресурсів.

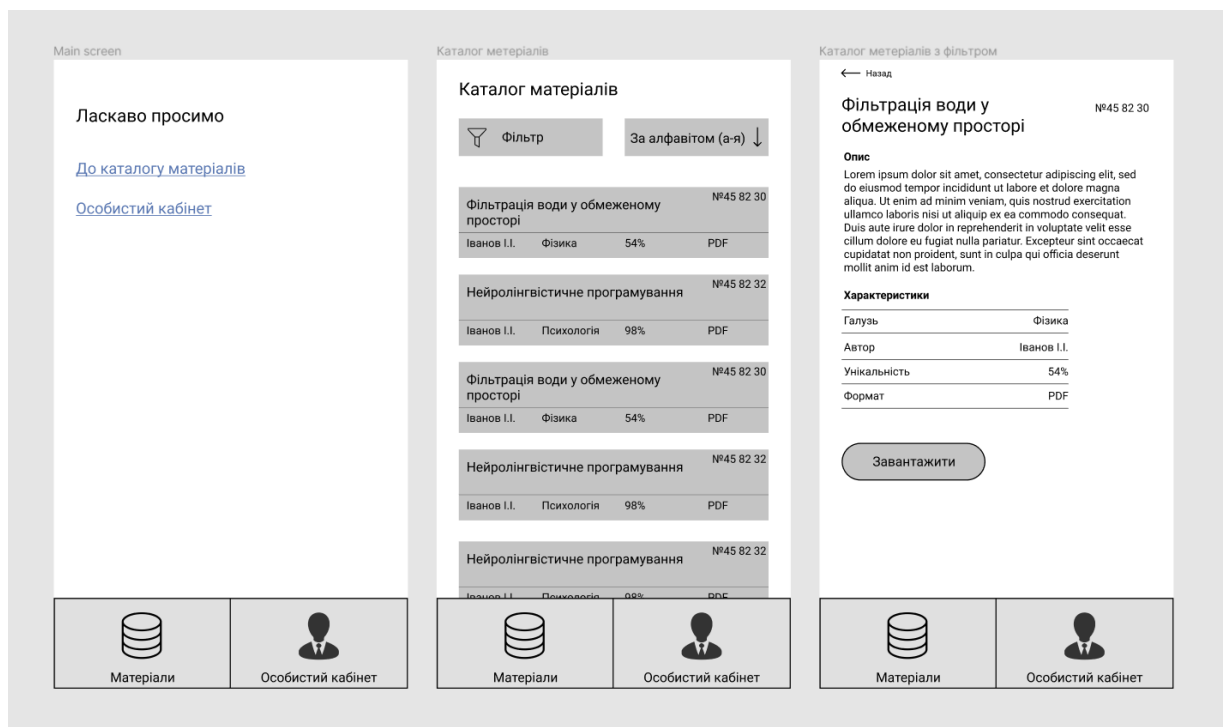
3 ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ

Враховуючи те, що при постановці задачі, однією з основних вимог була швидкодія та кросплатформеність – вибір інструментів був не дуже широким. Flutter + Dart та Firebase є найоптимальнішим варіантом розробки. Для створення макетів та прототипування використовувався графічний редактор Figma.

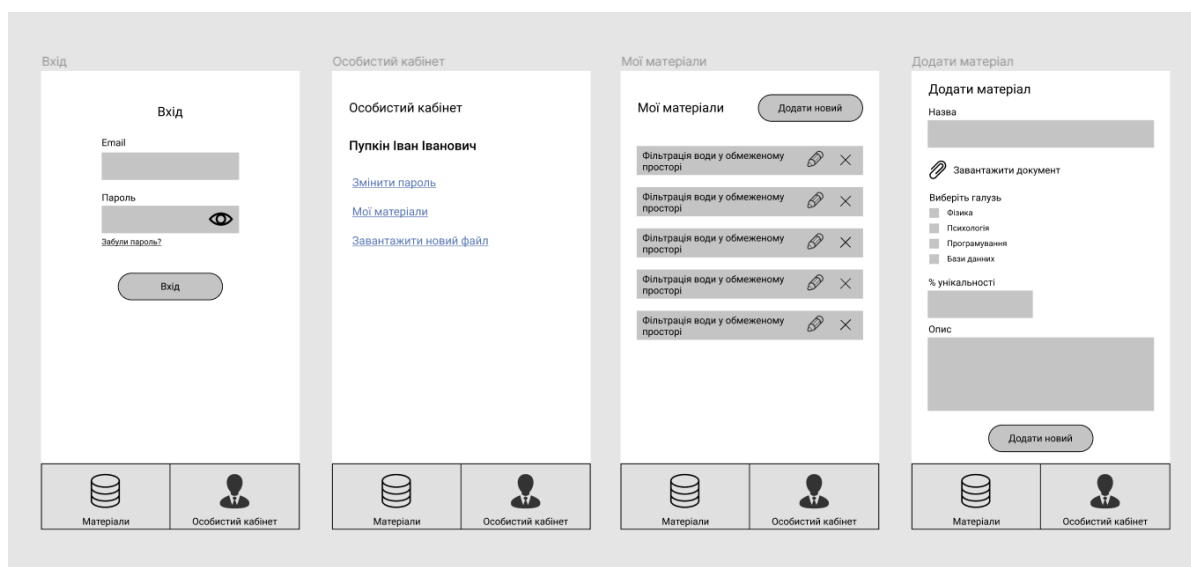
3.1 Figma

Figma - це векторний графічний редактор та інструмент для створення прототипів, який в основному працює як онлайн сервіс, з додатковими функціями в автономному режимі, увімкненими настільними програмами для macOS та Windows. Супутні додатки Figma Mirror для Android та iOS дозволяють переглядати прототипи Figma у реальному часі на мобільних пристроях. Набір функцій Figma фокусується на використанні в інтерфейсі користувача та дизайні користувацького досвіду, з акцентом на співпрацю в реальному часі.

На малюнку 3.1 та 3.2. показано UX прототипи додатку.



Малюнок 3.1 – UX прототип каталогу матеріалів.



Малюнок 3.2 – UX прототип особистого кабінету користувача.

3.2 Flutter

Flutter - це SDK для мобільної розробки з відкритим кодом, яку можна використовувати для створення власних програм на Android та iOS одночасно. Flutter існує з 2015 року, коли Google представив його, і залишився на стадії бета-версії до офіційного запуску в грудні 2018 року. З тих пір фреймворк стає все популярнішим і популярнішим. Більше того, велика кількість додатків у магазинах додатків розроблено саме за допомогою Flutter.

Основною ідеєю Flutter є використання віджетів. Завдяки поєднанню різних віджетів розробники можуть створити весь інтерфейс користувача. Кожен із цих віджетів визначає структурний елемент (наприклад, кнопку або меню), стилістичний елемент (шрифт або колірну схему), аспект макета (наприклад, доповнення) та багато інших[13].

Flutter не використовує віджети OEM, а надає свої готові шаблони, які мають звичайний вигляд для додатків Android або iOS (відповідно до Material Design або Cupertino). Звичайно, розробники можуть також створювати власні віджети.

Сьогодні Flutter - єдина мобільна SDK, яка пропонує реактивні види без використання JavaScript. Саме тому велика кількість розробників мобільних пристроїв використовують його у своїх проектах.

Переваги Flutter:

- Кросплатформенність

Flutter - це міжплатформенний інструмент розробки. Це означає, що розробники програмного забезпечення можуть використовувати той самий код для створення додатків для iOS та Android. Кросплатформенна розробка є найкращим методом економії часу та ресурсів протягом усього процесу розробки.

- Висока продуктивність

Висока продуктивність зумовлена двома причинами. По-перше, Flutter використовує Dart, який компілюється у власний код. По-друге, Flutter має власні віджети, тому немає необхідності отримувати доступ до OEM-віджетів. Як результат, між додатком і платформою менше зв'язку. Ці дві функції Flutter забезпечують швидкий час запуску програми та загалом менше проблем з продуктивністю.

- Швидкий рендеринг

Також популярність Flutter зумовлена функцією швидкого перезавантаження. Швидке перезавантаження дозволяє миттєво переглядати зміни, застосовані до коду на емуляторах. Змінений код перезавантажується менш ніж за секунду. Протягом усього часу програма працює, і розробникам не потрібно витрачати час на її перезапуск.

Це полегшує створення інтерфейсів, додавання нових функцій та виправлення помилок. Якщо в додатку виникає помилка, зазвичай можна виправити її, а потім продовжувати користуватися програмою.

- Сумісність

Ще однією перевагою Flutter є той факт, що він постачається із власними віджетами, що призводить до меншої кількості проблем

сумісності. Це дозволяє витрачати менше часу на тестування продукту на різних пристроях та різних версіях ОС. Також можна бути впевненим, що додаток буде працювати на майбутніх версіях ОС.

Примітка. Як тільки з'явиться нова версія Android або iOS. Віджети Flutter потрібно буде оновити (оскільки інструмент не використовує віджети власної платформи). Це займає невелику кількість часу.

- Open source

Flutter - це технологія з відкритим кодом, та має активну спільноту розробників, які надають підтримку, вносять вклад в обширну документацію інструменту та розробляють корисні ресурси. І Dart, і Flutter можна використовувати безкоштовно.

Flutter - одна з найбільш інноваційних мобільних технологій на ринку зараз. Переваги, які він приносить командам розробників, роблять його перспективним кандидатом на вибір мобільної технології найближчим часом.

Враховуючи те, що однією з основних вимог до програми були кросплатформеність та швидкодія – було прийнято рішення використовувати саме цей фреймворк.

3.3 Dart

Dart - мова програмування, створена компанією Google. Dart позиціонується в якості заміни / альтернативи JavaScript. Один з розробників мови Марк Міллер написав, що JavaScript «має фундаментальні недоліки, які неможливо виправити». Саме тому і був створений Dart.[14]

Перша загальнодоступна інформація про цю мову програмування з'явилася 12 вересня 2011 року на конференції розробників Goto. 10 жовтня 2011 року проведено офіційну презентацію мови Google Dart.

Переваги використання Dart:

- АОТ технологія компіляції;

Однією з основних переваг використання Dart є його компіляція AOT (Ahead-of-Time). AOT дозволяє компілювати код у власний машинний код. Як результат, це дозволяє двійковим файлам виконувати власну роботу.

- Сучасність;

Окрім того, що мова Dart дуже гнучка, проста у використанні та дуже сумісна з Javascript, Dart дозволяє програмі запускатися, навіть якщо в компіляції є помилки. Це дозволяє розробникам полегшити редагування, оскільки вони можуть протестувати лише невеликі частини коду, незалежно від того, решта програми є повною чи ні.

- Швидка взаємодія

Dart - це швидка мова. Вона майже завжди компілюється швидше, використовуючи компілятор від джерела до джерела. Вона має більш ефективний користувацький інтерфейс, ніж багато інших провідних мов.

Серед компаній, які використовують мову Dart, - Google, Mews, Ag Flow та Blossom. Люди загалом вважають, що Dart є однією з мов, яку легше зрозуміти, оскільки вона досить схожа на багато інших мов програмування. Вона надзвичайно сучасна, функціональна та гнучка, а також є конкурентоспроможною з іншими мовами програмування.[15]

Мною було обрано Dart через гарну взаємодію з Flutter та можливість тестувати відносно сирий код.

3.4 Firebase

Firebase - це платформа для розробки програмного забезпечення, заснована у 2011 році Firebase inc і придбана Google у 2014 році. Заснована як база даних у режимі реального часу, тепер вона має 18 служб (4 з них на даний час у бета-версії) та спеціальні API. Вся платформа - це рішення Backend-as-a-service як для

мобільних, так і для веб-додатків, що включає послуги зі створення, тестування та управління додатками.

Рішення BaaS усувають необхідність в управлінні базовими базами даних та отриманні відповідного обладнання. Натомість є можливість підключення їх до свого додатку через спеціальні API для кожної окремої служби. У випадку з Firebase їх існує 7, які охоплюють весь спектр внутрішніх технологій для програми. Список платформ, з якими інтегрується Firebase, включає Android, iOS, Web та Unity[16].

Firebase звільняє розробників від витрат часу на створення серверної частини. З Firebase не потрібно керувати серверами та не потрібно писати API.

Основні функції Firebase:

- База даних у реальному часі;
Дані в реальному часі - це шлях у майбутнє.

Більшість баз даних вимагають здійснення HTTP[17] запитів для отримання та синхронізації даних. Більшість баз даних надають дані лише тоді, коли отримують запит.

Коли додаток підключено до Firebase - він не підключається через звичайний HTTP. В такому випадку підключення відбувається через WebSocket. WebSockets працює набагато швидше, ніж HTTP. Потрібно лише одне підключення до сокету, надсилати запити не потрібно. Усі дані синхронізуються автоматично через цей єдиний WebSocket зі швидкістю, котру може забезпечити сам пристрій.

Firebase надсилає нові дані, щойно їх оновлять. Коли клієнт зберігає зміни в даних, усі підключені клієнти отримують оновлені дані майже миттєво.

- Сховище файлів;

Firebase Storage забезпечує простий спосіб збереження двійкових файлів - найчастіше зображень, але це може бути і будь-який інший формат.

Firebase Storage має власну систему правил безпеки, що захищає файли від несанкціонованого доступу.

- Автентифікація

Firebase auth має вбудовану систему автентифікації електронної пошти / пароля. Він також підтримує OAuth2 для Google, Facebook, Twitter та GitHub.

Firebase Auth інтегрується безпосередньо до бази даних Firebase, тому можна використовувати її для контролю доступу до даних.

- Хостинг

Firebase включає просту у використанні послугу хостингу для всіх статичних файлів. Він обслуговує їх із глобального CDN з HTTP.

Переваги Firebase:

- База даних

База даних NoSQL на платформі забезпечує безпечне зберігання всіх даних у колекціях та документах. База даних, Firestore, є нереляційною, що краще підходить для зберігання великих обсягів даних при збереженні значного рівня продуктивності, оскільки вони є більш гнучкими та масштабованими. Firestore дозволяє легко створювати чати між користувачами та відображати останні зміни користувача для всіх інших (що дає можливість створити додаток такого типу, як Google Docs або Google Drive, де можна побачити, коли та ким було зроблено останнє редагування).

- Швидка розробка

Firebase пропонує набір шаблонів, що робить весь цикл розробки коротшим і простішим. Розробник інтерфейсу може отримати конкретні серверні дані та інтегрувати їх до набору Firebase.

- Персоналізація

Оскільки Firebase належить Google, він може легко інтегруватися зі своїми продуктами, такими як Google Analytics. Цей інструмент допомагає

контролювати поведінку користувачів та використовувати зібрані дані, щоб зробити програму більш персоналізованою. Аналізуючи поведінку клієнтів, можна приймати рішення про додавання нових функцій і, таким чином, покращувати взаємодію з клієнтами.

- Цінові моделі

Firebase пропонує три тарифних плани.

1. Spark - це безкоштовний план з обмеженими основними функціями та ресурсами.

2. Flame починається з 25 доларів на місяць і пропонує широкий вибір опцій. Цей план більше підходить для додатків, які вже мають достатню кількість користувачів.

3. Blaze – тарифний план для бізнесу.

- Економічна ефективність

За допомогою служб Firebase, ми скорочуємо час виходу на ринок продукту та часові витрати.

- Спільнота

Firebase має багато прихильників в IT-спільноті та пропонує гарну технічну документацію. Навіть якщо користувач є новачком у цій платформі, він легко знайде відповіді на свої запити як в офіційній документації, так і в онлайн-спільнотах, таких як GitHub. Приблизно 1,5 мільйона додатків було створено за допомогою Firebase, що свідчить про велику кількість експертів.

- Простота інтеграції

Велика документація, готові API, прості інтерфейси та інші чудові функції спрощують інтеграцію та налаштування. Початок використання Firebase для створення додатків без необхідності робити складні конфігурації створює ще одну Firebase.

- Push-сповіщення

Хмарний обмін повідомленнями Firebase дозволяє розробникам увімкнути push-сповіщення, щоб сповіщати клієнтів про якусь інформацію.

Firebase було обрано мною через простоту у використанні, невисоку вартість та широкий функціонал у межах одного сервісу.

4 КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС

Користувацький інтерфейс (UI) - це точка взаємодії користувачів із комп'ютером, веб-сайтом або додатком. Основною метою ефективного користувацького інтерфейсу є зробити користувацький досвід легким та інтуїтивно зрозумілим, вимагаючи мінімальних зусиль з боку користувача для отримання максимально бажаного результату.[18]

Інтерфейс користувача створюється в сферах взаємодії, які звертаються до людських почуттів (зору, дотику, слуху тощо). Вони включають обидва пристрої введення, такі як клавіатура, миша, трекпад, мікрофон, сенсорний екран, сканер відбитків пальців, електронна ручка та камера та пристрої виведення, такі як монітори, динаміки та принтери. Пристрої, які взаємодіють із різними органами чуття, називаються «мультимедійними інтерфейсами користувача». Наприклад, повсякденний користувацький інтерфейс використовує комбінацію тактильного введення (клавіатура та миша) та візуального та слухового виводу (монітор та динаміки).

При розробці інтерфейсу до нього ставились такі вимоги:

- зрозумілий та інтуїтивно (хороший UX дизайн);
- модульний (в будь-який момент часу можна підключити додатковий функціонал).

4.1 Модуль авторизації та реєстрації

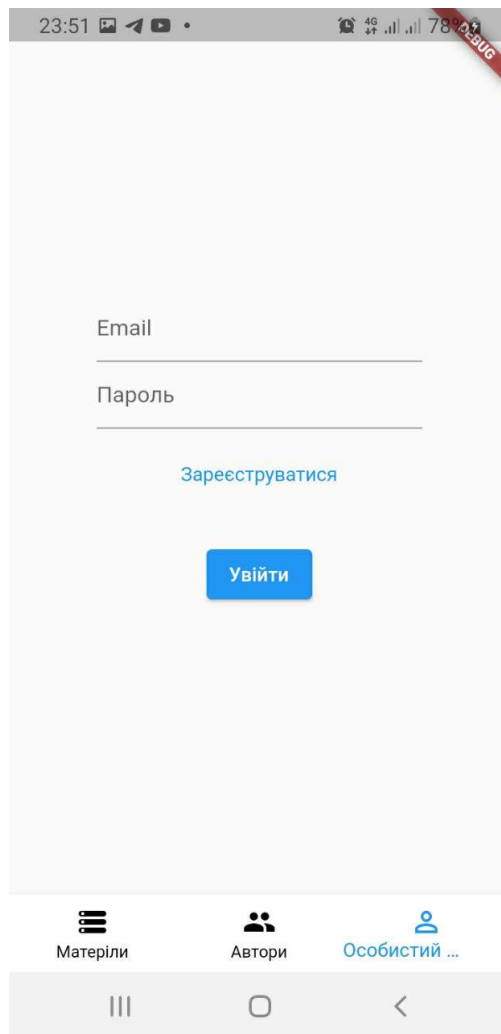
Аутентифікація - це процес розпізнавання користувача (механізм асоціювання вхідного запиту з набором ідентифікаційних даних). Надані облікові дані порівнюються з даними, що містяться у таблиці бази даних з інформацією по всім користувачам.

Процес автентифікації завжди запускається на початку програми, до того, як відбудуться перевірки дозволів та запусяться інші модулі. Різні системи можуть вимагати різні типи облікових даних, щоб встановити особу користувача.

В більшості випадків для ідентифікації використовується пароль та телефон або пошта.

У нашому додатку для підтвердження особи використовується пошта користувача та пароль. Процес авторизації показаний на малюнку 4.1.

Поля для вводу «Email» та «Пароль» є обов’язковими для заповнення та проходять валідацію перед відправкою на сервер (пароль має бути більше 8 символів, пошта має містити в собі знак «@»).



Малюнок 4.1 – Процес авторизації користувача.

Якщо користувач не має особистого запису – він може перейти на форму реєстрації. Вигляд форми реєстрації показаний на малюнку 4.2.

Форма містить 4 поля:

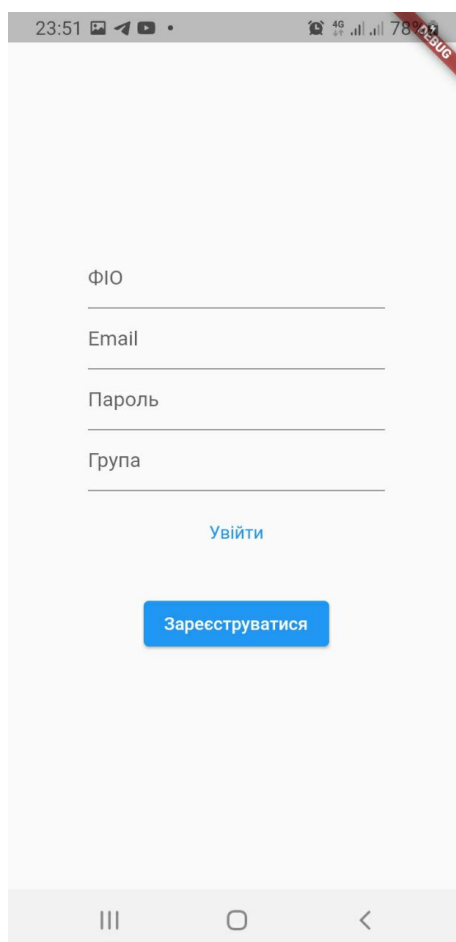
1. ПІБ користувача. Поле є обов'язковим для заповнення, не має валідації. В подальшому ці ініціали будуть використовуватись у авторстві над матеріалами та коментарями.

2. Пошта. Поле є обов'язковим для заповнення. Має валідацію по символу «@». Якщо вказана пошта вже міститься у базі даних – реєстрація не відбудеться.

3. Пароль. Поле є обов'язковим та має містити мінімум 8 символів.

4. Група. Поле не є обов'язковим. Викладачеві вказувати групу не потрібно.

Після кліку на кнопку «Зареєструватися» відбувається валідація полів. Якщо вона проходить успішно – інформація відправляється на сервер. У випадку успішної відповіді – реєстрація відбувається та користувач потрапляє до свого особистого кабінету.

A screenshot of a mobile application registration form. The form is displayed on a light gray background. At the top, there is a status bar with the time 23:51, signal strength, and battery level 78%. The form contains four input fields: "ФІО", "Email", "Пароль", and "Група". Below the "Група" field is a blue link labeled "Увійти". At the bottom of the form is a blue button labeled "Зареєструватися". The bottom of the screen shows a standard Android navigation bar with three icons: a square, a circle, and a triangle.

Малюнок 4.2 – Форма реєстрації нового користувача

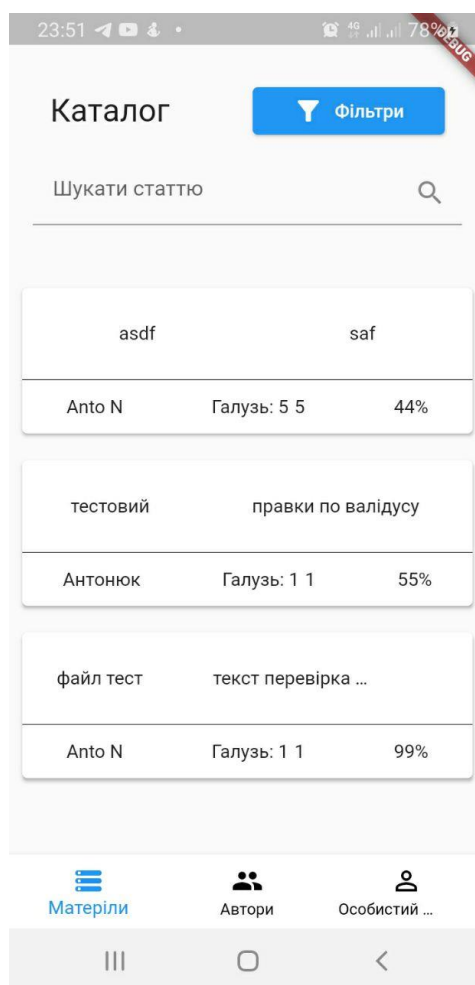
4.2 Головна сторінка

Додаток містить меню навігації, котре включає в себе 3 вкладки: Матеріали, Автори та Особистий кабінет. Навігаційна стрічка є фіксованою та відображається на всіх сторінках окрім реєстрації та авторизації.

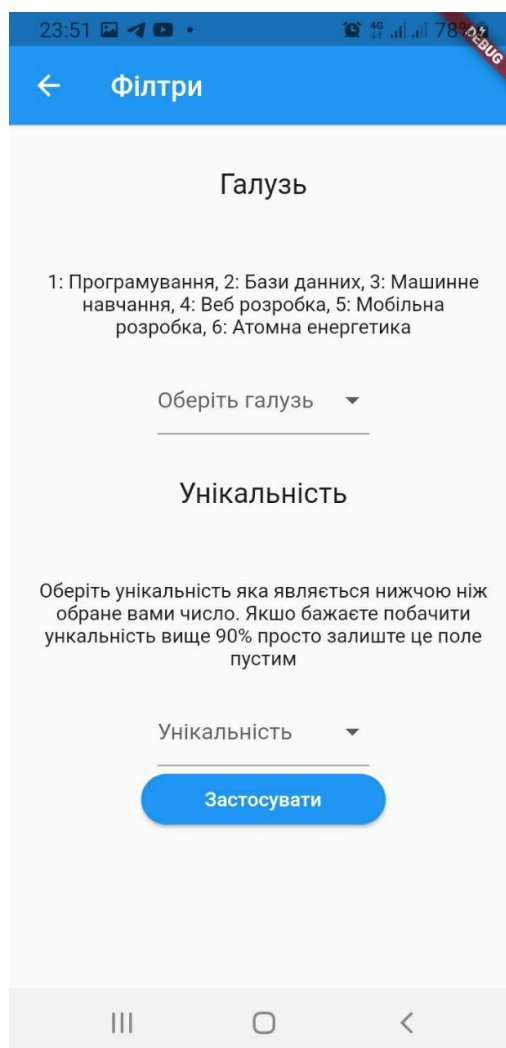
Вкладкою за замовчуванням є вкладка «Матеріалів». Вигляд вкладки зображений на малюнку 4.3.

У вкладці показаний список матеріалів у порядку від найновішого до найстарішого. Формат відображення: назва матеріалу, короткий опис, автор, галузь, унікальність. По кліку на матеріал відбувається перехід на сторінку матеріалу.

Також на сторінці присутні можливість пошуку матеріалу за назвою та фільтрації матеріалу. Вигляд фільтру матеріалів показано на малюнку 4.4.



Малюнок 4.3 – вкладка «Матеріали»



Малюнок 4.4 – фільтр матеріалів

Після вибору галузі та унікальності користувач має клікнути на кнопку «Застосувати». Після цього буде показано матеріали лише з певними параметрами.

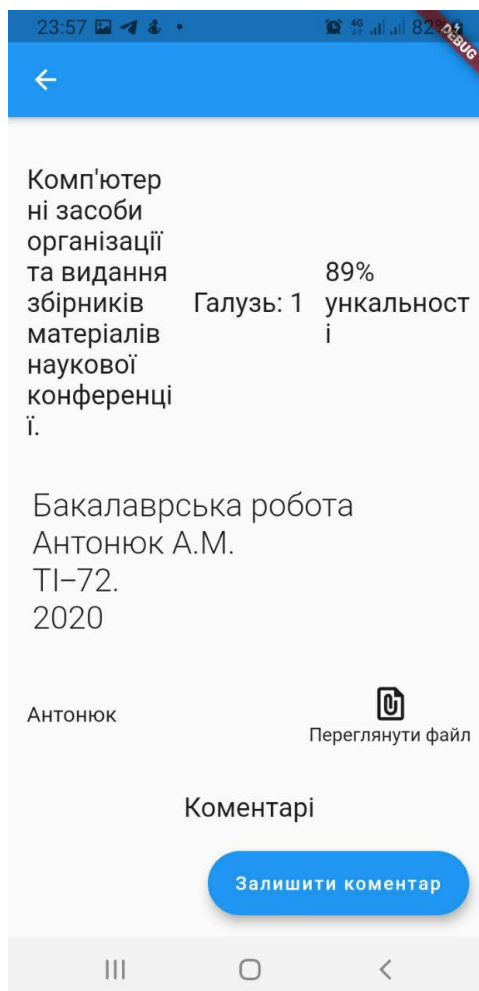
Пошук за матеріалами є динамічним та відбувається відразу після вводу.

4.3 Сторінка матеріалу

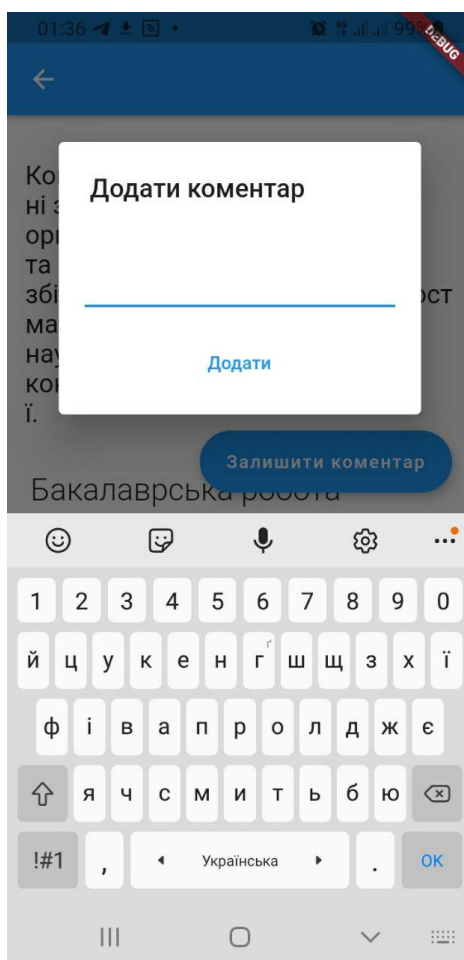
Після кліку на матеріал користувач переходить на сторінку конкретного матеріалу. Вигляд сторінки показано на малюнку 4.5.

На сторінці у повному обсязі показано назву та опис теми, галузь, автор та відсоток унікальності. По кліку на «Переглянути файл» у користувача відкривається потрібний файл.

Також на сторінці присутні коментарі до матеріалу та кнопка «Залишити коментар». Коментувати матеріали можуть лише авторизовані користувачі. Поле для коментаря є звичайним текстовим полем. Вигляд коментування показано на малюнку 4.6.



Малюнок 4.5. – вигляд одного матеріалу



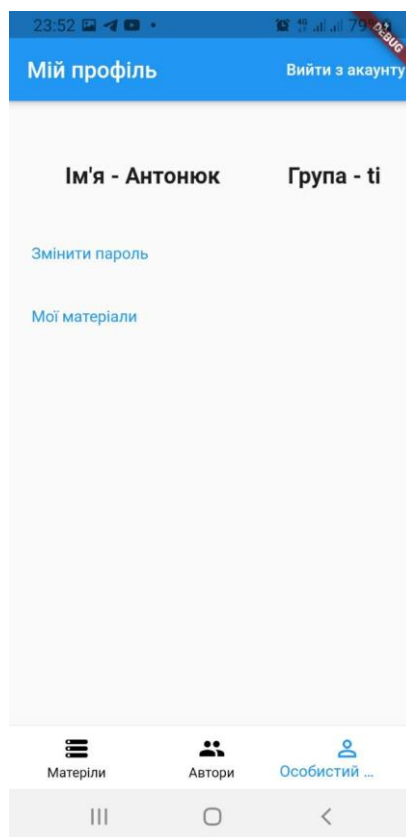
Малюнок 4.6 – Коментування матеріалів

4.4 Особистий кабінет користувача

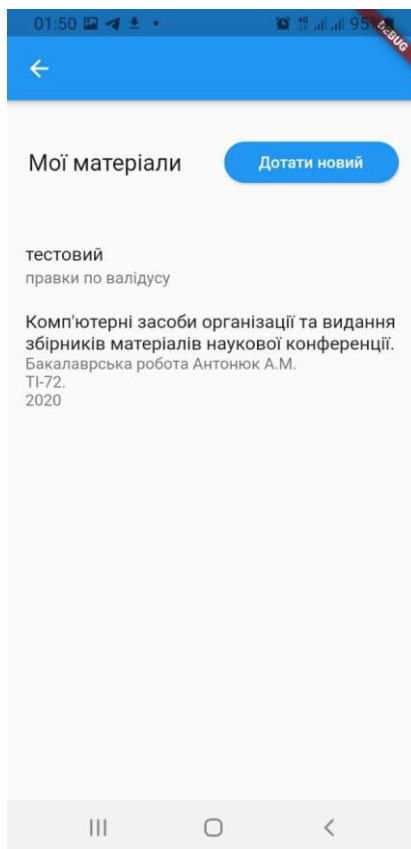
По кліку на вкладку «Особистий кабінет» авторизований користувач потрапляє на сторінку свого кабінету, де показані його ініціали та група. Вигляд сторінки показано на малюнку 4.7.

У особистому кабінеті користувач має змогу здійснити log out (вийти із свого кабінету), змінити пароль, переглянути список своїх матеріалів та додати новий матеріал.

Вигляд списку матеріалів в особистому кабінеті та можливість додавання нових файлів показано на малюнках 4.8 та 4.9 відповідно. Після завантаження матеріалу користувач отримує повідомлення про те, що його матеріал буде опубліковано після модерації (малюнок 4.10). Модератор має право на будь-які дії з будь-якими файлами.



Малюнок 4.7 – особистий кабінет користувача.



Малюнок 4.8 – список файлів в особистому кабінеті.

01:51 95% battery

←

Додати матеріал

Назва

Завантажити документ

Оберіть галузь

Програмування ☐

Бази даних ☐

Машинне навчання ☐

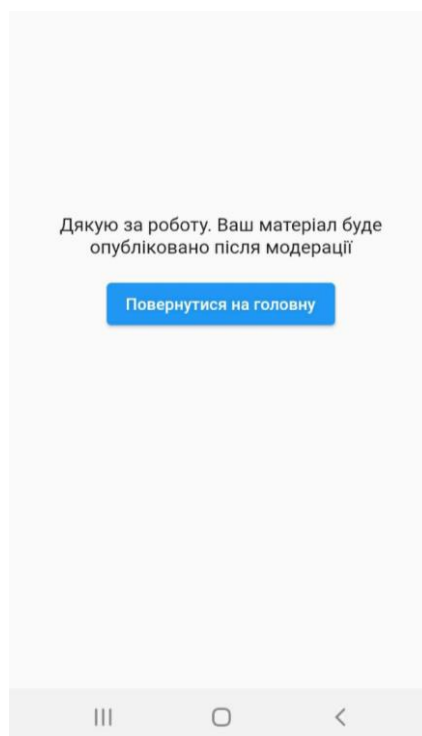
Веб розробка ☐

Мобільна розробка ☐

Атомна енергетика ☐

III □ <

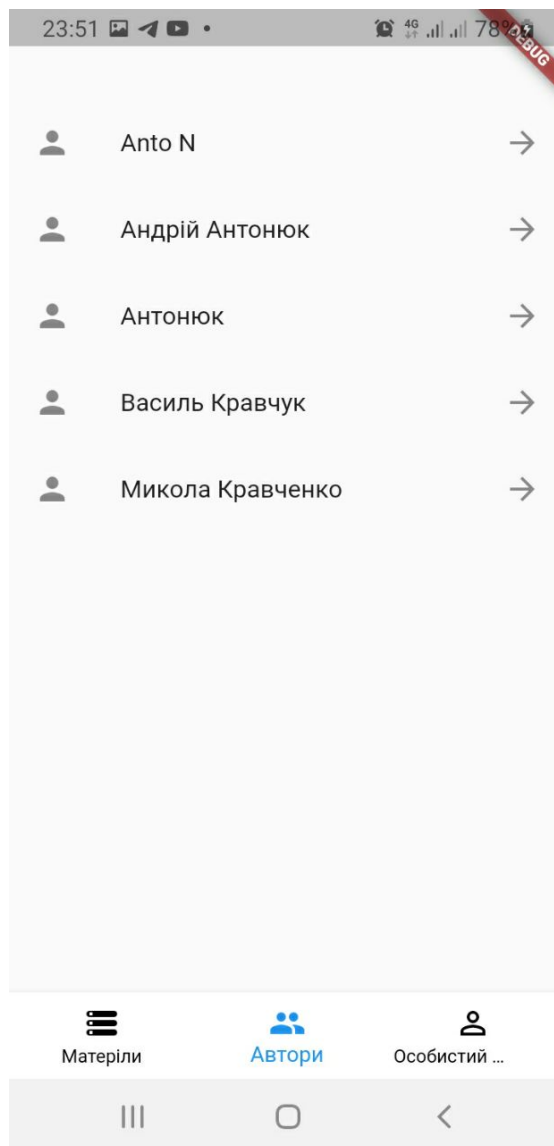
Малюнок 4.9 – форма завантаження нового матеріалу



Малюнок 4.10 – дякуємо за завантаження матеріалу

4.5 Список користувачів

По кліку на вкладку «Автори» користувач потрапляє на сторінку, де показано список усіх авторів додатку. По кліку на самого автора можна переглянути список саме його публікацій. Вигляд списку користувачів показано на малюнку 4.11.

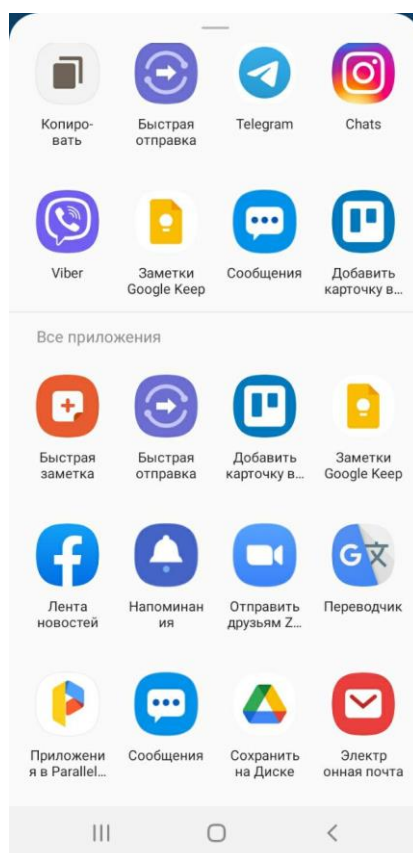


Малюнок 4.11 – список користувачів додатку

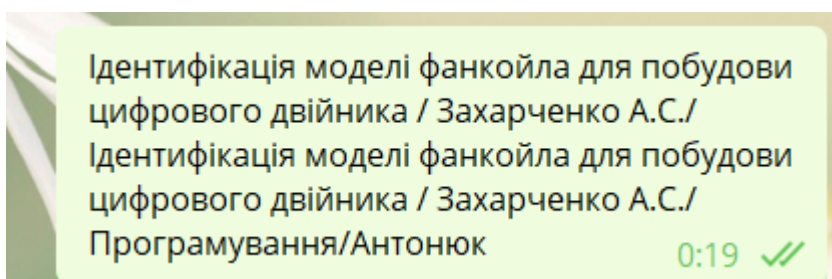
4.5 Sharing

У картці матеріалу присутня кнопка «Поділитися», котра дозволяє скопіювати, або надіслати будь-яким зручним способом посилання на статтю.

На малюнку 4.12 показано діалогове вікно додатків, у котрі можна надіслати посилання. На малюнку 4.13 показано саму надіслану інформацію.



Малюнок 4.12 – діалогове вікно кнопки «Поділитися»



Малюнок 4.13 – скопійоване посилання.

ВИСНОВКИ

В дипломній роботі виконувались роботи з розробки системи комп'ютерних засобів організації та видання збірників матеріалів наукової конференції.

Для досягнення поставленої мети і вирішення задач була проведена наступна робота:

1. Вивчена предметна область зберігання та структурування інформації кафедри. Було зроблено висновок, що зберігання на твердих носіях є непрактичним, а жодне із хмарних сховищ не підходить. Цим була обумовлена актуальність роботи.
2. Вивчена і проаналізована спеціалізована література за темою дипломної роботи. Розібрані основні технології організації файлових сховищ.
3. Досліджено та обґрунтовано вибір програмного забезпечення для реалізації задач дипломної, а саме – програмної платформи Flutter, мови програмування Dart та база даних Firebase.
4. Було розроблено мобільний додаток для структурування та зберігання наукових матеріалів.
5. Проведено тестування та апробація розробленої системи.

З аналізу проведеної роботи можна зробити наступні висновки:

- Розробка такої системи і її впровадження дозволяє користувачам швидко та зручно шукати потрібну інформацію в будь-який момент часу на мобільному пристрої.
- Розробка такої системи позбавляє кафедру від ресурсних затрат на зберігання матеріалів.
- Наукові матеріали більше не будуть губитись на різних пристроях студентів та викладачів.

- Система є гнучкою та може бути використаною не лише для зберігання наукових матеріалів, але і для інших студентських робіт, які несуть певну наукову цінність (курсові, дипломні, магістерські роботи).

Практичну цінність роботи бачу в тому, що: по-перше, був отриманий досвід розробки автоматизованих систем; по-друге, інститут отримає в своє розпорядження і зможе використовувати дану систему для зберігання файлів. В ході виконання проекту всі поставлені задачі було вирішено, а мета дипломної роботи в повній мірі досягнута. Подальшому розвиток системи бачу у її доповненні різними модулями та реалізації десктопного додатку.

ПЕРЕЛІК ПОСИЛАНЬ

1. Glover F. Future paths for Integer Programming and Links /Glover F. // Computers and Operations Research. – 1986. – Vol. 5. – P. 378-386.
2. Алмазов О.В., Семенов С.П., Шавкун А.Є. Інтеграція системи електронного документообігу в інформаційне середовище навчального закладу // Вісник Югорського державного університету. - 2012. - № 3. - С. 59-61
3. Гладких Н.О., Кудрявцев А.В. Використання коштів СУБД mysql для підготовки документів вузу // Актуальні питання викладання математики, інформатики та інформаційних технологій. - Єкатеринбург: УрГПУ, 2017. - С. 20-24.
4. Клішина А.П, Волкова Н.Р., Єрьоміна Н.Л., Митник А.А., Клижко Е.Н. Підходи до автоматизації документообігу в вузі // Вести. НГУ. Серія: інформаційні технології. 2017. Т. 15, № 1. С. 36-46.
5. Салтанов Е.Д., Кудрявцев А.В. Розробка бази даних і інтерфейсу для формування документів кафедри // Актуальні питання викладання математики, інформатики та інформаційних технологій. - Єкатеринбург: Уральський державний педагогічний університет, 2019. - С. 294-298.
6. Романов, Д. А. Роль документообігу в організації підприємства / Д. А. Романов // Правда про електронний документообіг: науковий збірник. Москва, 2004.-С. 144-157.
7. Куперштейн, В.І. Сучасні інформаційні технології в діловодстві та управлінні: навчальний посібник / В.І. Куперштейн. - СПб. : БХВ, 2003. - 205 с.
8. Відмінність файлообмінних сервісів від хмарних сервісів (на прикладі Dropbox і інших) [Електронний ресурс] - https://knowledge.allbest.ru/programming/3c0b65625b2ad69b5d53a89421316c26_0.html#text
9. Порівняння безпеки популярних хмарних сховищ - 2019 рік – Falcongaze [Електронний ресурс]

<https://falcongaze.com/ru/pressroom/publications/articles/compare-security-of-popular-cloud-services.html>

10. Network-attached storage [Електронний ресурс] - https://uk.wikipedia.org/wiki/Network-attached_storage
11. Storage area network [Електронний ресурс] – https://en.wikipedia.org/wiki/Storage_area_network
12. NAS vs. SAN vs. DAS | Data Storage Comparison | Pros & Cons | Weka IO [Електронний ресурс] - <https://www.weka.io/learn/nas-vs-san-vs-das-storage-comparison/>
13. Flutter - Beautiful native apps in record time [Електронний ресурс] - <https://flutter.dev/>
14. Мова програмування JavaScript // Сучасний підручник JavaScript URL: [Електронний ресурс] - <https://learn.javascript.ru/>
15. Dart programming language [Електронний ресурс] - <https://dart.dev/>
16. Що тає Firebase? [Електронний ресурс] - <https://avada-media.ua/services/firebase/>
17. Простую мовою про HTTP [Електронний ресурс] - <https://habr.com/ru/post/215117/>
18. What Is a User Interface? (Definition, Types and Examples) [Електронний ресурс] - <https://www.indeed.com/career-advice/career-development/user-interface>

ДОДАТОК А

Комп'ютерні засоби організації та видання збірників матеріалів наукової конференції

Специфікація

УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ72124_21Б

Аркушів 1

Київ-2021

Позначення	Найменування	Примітки
Документація		
УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС _ТІ72124_21Б-1	Записка.docx	Текстова частина дипломної роботи
Компоненти		
УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС _ТІ72124_21Б-2	Main.dart	Основний файл програми
УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС _ТІ72124_21Б-3	add_material.d art	Модуль додавання файлів
УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС _ТІ72124_21Б-4	Register.dart	Модуль реєстрації
УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС _ТІ72124_21Б-5	Signin.dart	Модуль входу

ДОДАТОК Б

Комп'ютерні засоби організації та видання збірників матеріалів наукової конференції

Лістинг програми

УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ72124_21Б-2

Аркушів 11

Київ-2021

УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ72124_21Б-2

Main.dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:laba2021/admin/admin_mat_det.dart';
// screens
import 'package:laba2021/screens/added.dart';
import 'package:laba2021/screens/change_pwd.dart';
import 'package:laba2021/screens/filter_screen.dart';
import 'package:laba2021/screens/filtered.dart';
import 'package:laba2021/screens/material_details.dart';
import 'package:laba2021/screens/my_materials.dart';
import 'package:laba2021/screens/profile_screen.dart';
import 'package:laba2021/screens/register.dart';
import 'package:laba2021/screens/signin.dart';
import 'package:laba2021/screens/add_material.dart';
import 'package:laba2021/screens/splash.dart';
import 'package:laba2021/screens/tabs_screen.dart';
import 'package:laba2021/screens/user.dart';
import 'package:laba2021/screens/users_list.dart';
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  SystemChrome.setPreferredOrientations(
    [
      DeviceOrientation.portraitUp,
      DeviceOrientation.portraitUp,
    ],
  );
  // ініціалізація додатку через бд
  await Firebase.initializeApp();
  runApp(MyApp());
}
class MyApp extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();
}
class _MyAppState extends State<MyApp> {
  @override
  Widget build(BuildContext context) {
    final Future<FirebaseApp> _initialization = Firebase.initializeApp();
    return FutureBuilder(
      // ініціалізація додатку через бд
```

```

future: _initialization,
builder: (context, snapshot) {
  return MaterialApp(
    title: 'Flutter Demo',
    theme: ThemeData(
      primarySwatch: Colors.blue,
    ),
    home: snapshot.connectionState != ConnectionState.done
      ? SplashScreen()
      : StreamBuilder(
        stream: FirebaseAuth.instance.authStateChanges(),
        builder: (ctx, streamSnap) {
          if (!snapshot.hasData) {
            return RegisterScreen();
          }
          return TabsScreen();
        },
      ),
    routes: {
      Added.routeName: (ctx) => Added(),
      AddMaterial.routeName: (ctx) => AddMaterial(),
      AdminMatDet.routeName: (ctx) => AdminMatDet(),
      ChangePassword.routeName: (ctx) => ChangePassword(),
      Filtered.routeName: (ctx) => Filtered(),
      FilterScreen.routeName: (ctx) => FilterScreen(), // _setFilters
      MaterialsDetails.routeName: (ctx) => MaterialsDetails(),
      MyMaterials.routeName: (ctx) => MyMaterials(),
      ProfileScreen.routeName: (ctx) => ProfileScreen(),
      RegisterScreen.routeName: (ctx) => RegisterScreen(),
      SigninScreen.routeName: (ctx) => SigninScreen(),
      TabsScreen.routeName: (ctx) => TabsScreen(),
      UserPage.routeName: (ctx) => UserPage(),
      UsersList.routeName: (ctx) => UsersList(),
    },
  );
},
);
}
}

```

document_picker.dart

```

import 'dart:io';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_storage/firebase_storage.dart';

```

```

import 'package:flutter/material.dart';
import 'package:file_picker/file_picker.dart';
import 'package:flutter/services.dart';
class DocumentPicker extends StatefulWidget {
  @override
  _DocumentPickerState createState() => _DocumentPickerState();
}
class _DocumentPickerState extends State<DocumentPicker> {
  String _fileName = 'Document';
  String _path;
  String _extension;
  TextEditingController _controller = new TextEditingController();
  bool selected = false;
  @override
  void initState() {
    super.initState();
    _controller.addListener(() => _extension = _controller.text);
  }
  Future<void> _openFileExplorer() async {
    try {
      FilePickerResult result = await FilePicker.platform
        .pickFiles(type: FileType.any, allowCompression: true);
      if (result == null) return;
      if (result != null) {
        File file = File(result.files.first.path); // отримує шлях до файлу
        final docsL =
          await FirebaseFirestore.instance.collection('materials').get(); // звертається до
дб та отримує всі документи з колекції
        final total = docsL.docs.length + 1; // отримує загальну кількість документів з
колекції та додає 1
        await FirebaseStorage.instance
          .ref()
          .child('${total.toString()}')
          .putFile(file); // загрузка файлу до Firebase Storage
      }
    } on PlatformException catch (e) {
      print("Unsupported operation" + e.toString()); // при виникненні проблем
виводить проблему в термінал
    }
    if (!mounted) return;
    setState(() {
      _fileName = _path != null ? _path.split('/').last : 'Файл додано!'; // косметична
частина: після додавання файлу до бд замінює додати файл на файл додано
      selected = true;
    });
  }
}

```

```

    });
    // }
  }
  @override
  Widget build(BuildContext context) {
    return Row(
      children: [
        Icon(Icons.upload_file),
        SizedBox(
          width: 10,
        ),
        GestureDetector(
          child: selected
            ? Container(
                width: MediaQuery.of(context).size.width * 0.8,
                child: Text(
                  _fileName,
                  style: TextStyle(
                    fontSize: 18,
                    fontWeight: FontWeight.normal,
                  ),
                  softWrap: true,
                  overflow: TextOverflow.clip,
                ),
              )
            : const Text(
                'Завантажити документ',
                style: TextStyle(
                  fontSize: 18,
                  fontWeight: FontWeight.normal,
                ),
              ),
          onTap: _openFileExplorer,
        ),
      ],
    );
  }
}

```

login_widget.dart

```

import 'package:flutter/material.dart';
import 'package:laba2021/screens/register.dart';
class LogWidget extends StatefulWidget {
  LogWidget(this.submit);

```

```

final void Function(String email, String password) submit;
@override
_LogWidgetState createState() => _LogWidgetState();
}
class _LogWidgetState extends State<LogWidget> {
  final _keyForm = GlobalKey<FormState>();
  var _email = "";
  var _password = "";
  // Функція яка отримує данні введені користувачем і передає їх на сторінку
  SigninScreen
  void _submit() async {
    final isValid = _keyForm.currentState.validate();
    if (isValid) {
      _keyForm.currentState.save();
      widget.submit(
        _email.trim(),
        _password.trim(),
      );
    }
  }
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Form(
        key: _keyForm,
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            // email
            Container(
              width: MediaQuery.of(context).size.width * 0.65,
              child: TextFormField(
                // key: ,
                keyboardType: TextInputType.emailAddress,
                textInputAction: TextInputAction.next,
                decoration: const InputDecoration(
                  hintText: 'Email',
                  focusedBorder: UnderlineInputBorder(
                    borderSide: BorderSide(
                      color: Color(0xFFFFFA31A),
                    ),
                  ),
                ),
              ),
            ),
          ],
        ),
      ),
    );
  }
}

```

```

    validator: (value) {
      if (value.isEmpty || !value.contains('@')) {
        return 'Введіть правильний емейл';
      }
      return null;
    },
    onSave: (value) {
      _email = value;
    },
  ),
),
// password
Container(
  width: MediaQuery.of(context).size.width * 0.65,
  child: TextFormField(
    // key: ,
    keyboardType: TextInputType.emailAddress,
    textInputAction: TextInputAction.done,
    decoration: const InputDecoration(
      hintText: 'Пароль',
      focusedBorder: UnderlineInputBorder(
        borderSide: BorderSide(
          color: Color(0xFFFFFA31A),
        ),
      ),
    ),
    obscureText: true,
    validator: (value) {
      if (value.isEmpty || value.length < 8) {
        return 'Пароль має містити як мінімум 8 символів';
      }
      return null;
    },
    onSave: (value) {
      _password = value;
    },
  ),
),
Container(
  margin: EdgeInsets.symmetric(vertical: 25),
  child: GestureDetector(
    child: const Text(
      'Зареєструватися',
      style: TextStyle(

```

```

        color: Colors.blue,
      ),
    ),
    onTap: () {
      Navigator.of(context).pushNamed(RegisterScreen.routeName);
    },
  ),
),
Padding(
  padding: const EdgeInsets.only(top: 15.0),
  child: ElevatedButton(
    onPressed: _submit,
    child: const Text('Увійти'),
  ),
),
],
),
),
);
}
}

```

reg_widget.dart

```

import 'package:flutter/material.dart';
class RegWidget extends StatefulWidget {
  RegWidget(this.submit);
  final void Function(String email, String password, String name, String group)
    submit;
  @override
  _RegWidgetState createState() => _RegWidgetState();
}
class _RegWidgetState extends State<RegWidget> {
  final _keyForm = GlobalKey<FormState>();
  final nameController = TextEditingController();
  final emailController = TextEditingController();
  final pwdController = TextEditingController();
  final grpController = TextEditingController();
  var _name = "";
  var _email = "";
  var _pwd = "";
  var _grp = "";
  // Функція яка отримує данні введені користувачем і передає їх на сторінку
  RegisterScreen

```

```

void _submitForm() async {
  final isValid = _keyForm.currentState.validate();
  if (isValid) {
    _keyForm.currentState.save();
    widget.submit(
      _email,
      _pwd,
      _name,
      _grp,
    );
  }
}

@override
Widget build(BuildContext context) {
  return Center(
    child: Form(
      key: _keyForm,
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          // FIO
          Container(
            width: MediaQuery.of(context).size.width * 0.65,
            child: TextFormField(
              textInputAction: TextInputAction.next,
              decoration: const InputDecoration(
                hintText: 'FIO',
              ),
              validator: (value) {
                if (value.isEmpty) {
                  return 'Заповніть це поле';
                }
                return null;
              },
              onSave: (value) {
                _name = value;
              },
              onChanged: (text) {
                setState(() {
                  _name = text;
                });
              },
              controller: nameController,

```

```

    ),
  ),
  // email
  Container(
    width: MediaQuery.of(context).size.width * 0.65,
    child: TextFormField(
      keyboardType: TextInputType.emailAddress,
      textInputAction: TextInputAction.next,
      decoration: const InputDecoration(
        hintText: 'Email',
      ),
    ),
    validator: (value) {
      if (value.isEmpty || !value.contains('@')) {
        return 'Введіть правильний емейл';
      }
      return null;
    },
    onSave: (value) {
      _email = value;
    },
    onChanged: (text) {
      setState(() {
        _email = text;
      });
    },
    controller: emailController,
  ),
),
// password
Container(
  width: MediaQuery.of(context).size.width * 0.65,
  child: TextFormField(
    keyboardType: TextInputType.emailAddress,
    textInputAction: TextInputAction.done,
    decoration: const InputDecoration(
      hintText: 'Пароль',
    ),
  ),
  obscureText: true,
  validator: (value) {
    if (value.isEmpty || value.length < 8) {
      return 'Пароль має містити як мінімум 8 символів';
    }
    return null;
  },
),

```

```

    onSave: (value) {
      _pwd = value;
    },
    onChanged: (text) {
      setState(() {
        _pwd = text;
      });
    },
    controller: pwdController,
  ),
),
// Група
Container(
  width: MediaQuery.of(context).size.width * 0.65,
  child: TextFormField(
    textInputAction: TextInputAction.next,
    decoration: const InputDecoration(
      hintText: 'Група',
    ),
    validator: (value) {
      if (value.isEmpty) {
        return 'Заповніть це поле';
      }
      return null;
    },
    onSave: (value) {
      _grp = value;
    },
    onChanged: (text) {
      setState(() {
        _grp = text;
      });
    },
    controller: grpController,
  ),
),
// Логін
Container(
  margin: EdgeInsets.symmetric(vertical: 25),
  child: GestureDetector(
    child: const Text(
      'Увійти',
      style: TextStyle(
        color: Colors.blue,

```

```

    ),
  ),
  onTap: () {
    Navigator.of(context).pop();
  },
),
),
// Реєстрація
Padding(
  padding: const EdgeInsets.only(top: 15.0),
  child: ElevatedButton(
    onPressed: _submitForm,
    child: const Text('Зареєструватися'),
  ),
),
],
),
),
);
}
}

```

add_material.dart

```

import 'package:cloud_firestore/cloud_firestore.dart';
// import 'package:file_picker/file_picker.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_storage/firebase_storage.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:laba2021/screens/added.dart';
import 'package:laba2021/widgets/document_picker.dart';
import 'dart:io';

class AddMaterial extends StatefulWidget {
  static const routeName = '/add';
  @override
  _AddMaterialState createState() => _AddMaterialState();
}

class _AddMaterialState extends State<AddMaterial> {
  final nameController = TextEditingController();
  final perController = TextEditingController();
  final desController = TextEditingController();
  final _key = GlobalKey<FormState>();
  String _name;
  File doc;

```

```

var _prog = false;
var _db = false;
var firstId;
var secondId;
var _mach = false;
var _web = false;
var _mob = false;
var _atom = false;
var _percent;
String description;
// name
Future<void> _addMaterial() async {
  final uid = FirebaseAuth.instance.currentUser.uid;
  final docsL =
    await FirebaseFirestore.instance.collection('materials').get();
  final total = docsL.docs.length + 1;
  // отримання силкі на файл в FirebaseStorage
  final fileUrl = await FirebaseStorage.instance
    .ref()
    .child('${total.toString()}')
    .getDownloadURL();
  final name =
    await FirebaseFirestore.instance.collection('users').doc(uid).get();
  final isValid = _key.currentState.validate();
  if (isValid) {
    final docsL =
      await FirebaseFirestore.instance.collection('materials').get();
    final total = docsL.docs.length + 1;
    // те саме що і було раніше з додаванням даних введенням юзером до бд в
    колекцію materials
    await FirebaseFirestore.instance.collection('materials').add({
      'author_id': '${FirebaseAuth.instance.currentUser.uid}',
      'description': description,
      'file': fileUrl,
      'id': total,
      'firstId': firstId,
      'secondId': secondId,
      'name': _name,
      'uniqueness': _percent,
      'author_name': name['initials'],
    });
    Navigator.of(context).pushNamed(Added.routeName);
  }
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(),
    body: SingleChildScrollView(
      child: Form(
        key: _key,
        child: StreamBuilder(
          stream:
            FirebaseFirestore.instance.collection('materials').snapshots(),
          builder: (ctx, future) => Column(
            mainAxisAlignment: MainAxisAlignment.start,
            children: [
              Padding(
                padding: EdgeInsets.symmetric(vertical: 20),
                child: const Text(
                  'Додати матеріал',
                  style: TextStyle(
                    fontSize: 20,
                    fontWeight: FontWeight.w400,
                  ),
                ),
              ),
            ],
          ),
          Container(
            width: MediaQuery.of(context).size.width,
            padding: EdgeInsets.symmetric(horizontal: 20),
            child: TextFormField(
              textInputAction: TextInputAction.next,
              decoration: const InputDecoration(
                hintText: 'Назва',
                border: OutlineInputBorder(
                  borderSide: BorderSide(
                    width: 1,
                    color: Colors.black,
                  ),
                ),
              ),
              validator: (value) {
                if (value.isEmpty) {
                  return 'Заповніть це поле';
                }
                return null;
              },
            ),
          ),
        ),
      ),
    ),
  );
}

```

```

    onSave: (value) {
      _name = value;
    },
    controller: nameController,
    onChanged: (text) {
      setState(() {
        _name = text;
      });
    },
  ),
),
// Download Document
Container(
  width: MediaQuery.of(context).size.width * 0.9,
  padding: EdgeInsets.only(top: 30),
  child: DocumentPicker(),
),
// Галузь
Container(
  padding: EdgeInsets.symmetric(
    vertical: 30,
  ),
  child: const Text(
    'Оберіть галузь',
    style: TextStyle(fontSize: 20),
  ),
),
),
SwitchListTile(
  title: const Text('Програмування'),
  value: _prog,
  onChanged: (bool prog) {
    setState(() {
      _prog = prog;
    });
    if (_prog == true && firstId == null) {
      setState(() {
        firstId = '1';
      });
    }
    if (_prog == true && firstId != null) {
      setState(() {
        secondId = '1';
      });
    }
  }
}

```

```

    },
  ),
  SwitchListTile(
    title: const Text('Бази даних'),
    value: _db,
    onChanged: (bool db) {
      setState(() {
        _db = db;
      });
      if (_db == true && firstId == null) {
        setState(() {
          firstId = '2';
        });
      }
      if (_db == true && firstId != null) {
        setState(() {
          secondId = '2';
        });
      }
    },
  ),
  SwitchListTile(
    title: const Text('Машинне навчання'),
    value: _mach,
    onChanged: (bool mach) {
      setState(() {
        _mach = mach;
      });
      if (_db == true && firstId == null) {
        setState(() {
          firstId = '3';
        });
      }
      if (_db == true && firstId != null) {
        setState(() {
          secondId = '3';
        });
      }
    },
  ),
  SwitchListTile(
    title: const Text('Веб розробка'),
    value: _web,
    onChanged: (bool web) {

```

```

    setState(() {
      _web = web;
    });
    if (_web == true && firstId == null) {
      setState(() {
        firstId = '4';
      });
    }
    if (_web == true && firstId != null) {
      setState(() {
        secondId = '4';
      });
    }
  },
),
SwitchListTile(
  title: const Text('Мобільна розробка'),
  value: _mob,
  onChanged: (bool mob) {
    setState(() {
      _mob = mob;
    });
    if (_mob == true && firstId == null) {
      setState(() {
        firstId = '5';
      });
    }
    if (_mob == true && firstId != null) {
      setState(() {
        secondId = '5';
      });
    }
  },
),
SwitchListTile(
  title: const Text('Атомна енергетика'),
  value: _atom,
  onChanged: (bool atom) {
    setState(() {
      _atom = atom;
    });
    if (_atom == true && firstId == null) {
      setState(() {
        firstId = '6';

```

```

    });
  }
  if (_atom == true && firstId != null) {
    setState(() {
      secondId = '6';
    });
  }
},
),
// uniqueness
Container(
  width: MediaQuery.of(context).size.width * 0.4,
  child: TextFormField(
    // key: ,
    keyboardType: TextInputType.number,
    textInputAction: TextInputAction.next,
    decoration: const InputDecoration(
      hintText: 'Унікальність в %',
      border: OutlineInputBorder(
        borderSide: BorderSide(
          width: 1,
          color: Colors.black,
        ),
      ),
    ),
    validator: (value) {
      if (value.isEmpty) {
        return 'Заповніть це поле';
      }
      if (int.parse(value) >= 101) {
        return 'Значення не може бути більше 100';
      }
      return null;
    },
    onSave: (value) {
      _percent = value;
    },
    controller: perController,
    onChanged: (text) {
      setState(() {
        _percent = text;
      });
    },
  ),
),

```

```

),
Padding(
  padding: const EdgeInsets.symmetric(vertical: 15.0),
  child: Text(
    'Опис',
    style: TextStyle(
      fontSize: 18,
      fontWeight: FontWeight.normal,
    ),
  ),
),
Container(
  width: MediaQuery.of(context).size.width * 0.9,
  child: TextFormField(
    keyboardType: TextInputType.multiline,
    maxLines: 7,
    decoration: const InputDecoration(
      hintText: 'Залиште ваш опис',
      border: OutlineInputBorder(
        borderSide: BorderSide(
          width: 1,
          color: Colors.black,
        ),
      ),
    ),
    validator: (value) {
      if (value.isEmpty) {
        return 'Залиште опис';
      }
      return null;
    },
    onSave: (value) {
      description = value;
    },
    controller: desController,
    onChanged: (text) {
      setState(() {
        description = text;
      });
    },
  ),
),
Container(
  padding: EdgeInsets.symmetric(vertical: 15),

```

```

        child: ElevatedButton(
          onPressed: _addMaterial,
          child: const Text('Додати'),
          style: ElevatedButton.styleFrom(
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(25),
            ),
            padding:
              EdgeInsets.symmetric(horizontal: 30, vertical: 14)),
        ),
      ),
    ],
  ),
),
),
),
);
}
}

```

register.dart

```

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:laba2021/screens/profile_screen.dart';
import 'package:laba2021/widgets/reg_widget.dart';
class RegisterScreen extends StatefulWidget {
  static const routeName = '/register';
  @override
  _RegisterScreenState createState() => _RegisterScreenState();
}
class _RegisterScreenState extends State<RegisterScreen> {
  final _auth = FirebaseAuth.instance;
  void _submitForm( // асинхронна функція яка отримує значення
    String email,
    String password,
    String name,
    String group,
  ) async {
    // спочатку викуне реєстрацію
    await _auth
      .createUserWithEmailAndPassword( // метод файер бейз які дозволяє
зареєструватися
        email: email,

```

```

        password: password,
      )
      // після завершення реєстрації виконує певну дію прописану в графічних
      дужках
      .then(
        (_) {
          User usercredential = FirebaseAuth.instance.currentUser; // отримує данні юзера
          // створює новий документ в прописані колекції і заносить в документ данні
          прописані нижче
          FirebaseFirestore.instance
            .collection('users')
            .doc(usercredential.uid) // задаємо назву цьому документу: називається буде
            відповідно до айді юзера
            .set(
              {
                'id': usercredential.uid, // 'назва поля в документі': значення
                'email': email,
                'password': password,
                'initials': name,
                'group': group,
              },
            )
          // після чого переносить на сторінку профіля
          .then(
            (_) => Navigator.of(context)
              .pushReplacementNamed(ProfileScreen.routeName),
          );
        },
      );
      // Navigator.of(context).pushReplacementNamed(ProfileScreen.routeName);
    }
    @override
    Widget build(BuildContext context) {
      return Scaffold(
        body: RegWidget(_submitForm),
      );
    }
  }
}

```

signin.dart

```

import 'package:flutter/material.dart';
import 'package:laba2021/screens/profile_screen.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:laba2021/widgets/login_widget.dart';

```

```

class SigninScreen extends StatefulWidget {
  static const routeName = '/signin';
  @override
  _SigninScreenState createState() => _SigninScreenState();
}
class _SigninScreenState extends State<SigninScreen> {
  final _auth = FirebaseAuth.instance;
  void _submitForm(
    String email,
    String password,
  ) async {
    {
      await _auth
        .signInWithEmailAndPassword(
          // метод файер бейз які дозволяє залогінитися
          email: email,
          password: password,
        )
        .then((value) {
          setState(() {
            // метод якій просто оновлює стан сторінки
          });
        });
    }
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: StreamBuilder(
        stream: FirebaseAuth.instance
          .authStateChanges(), // прослуховування стану юзера
        builder: (ctx, builder) {
          if (!builder.hasData) {
            // якщо не має значення повертає сторінку логіну
            return LogWidget(_submitForm);
          }
          return ProfileScreen(); // в іншому випадку повертає сторінку юзера
        },
      ),
    );
  }
}

```

ДОДОТОК В

Комп'ютерні засоби організації та видання збірників матеріалів наукової конференції

Опис програми

УКР.КПІ_ім_Ігоря_СІКОРСЬКОГО_ТЕФ_АПЕПС_ТІ72124_21Б-6

Аркушів 8

Київ-2021

АНОТАЦІЯ

Даний додаток містить у собі опис основних компонентів, котрі використовувались при розробці системи комп'ютерних засоби організації та видання збірників матеріалів наукової конференції.

Додаток розроблено з використанням таких технологій: Dart, Flutter, Firebase.

ЗМІСТ

1. ЗАГАЛЬНІ ВІДОМОСТІ	88
2. ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ.....	89
3. ОПИС ЛОГІЧНОЇ СТРУКТУРИ	90
4. ВИКОРИСТАНІ ТЕХНІЧНІ ЗАСОБИ.....	91
5. ВИКЛИК І ЗАВАНТАЖЕННЯ.....	92
6. ВХІДНІ ТА ВИХІДНІ ДАНІ	93

ЗАГАЛЬНІ ВІДОМОСТІ

Додаток надає можливість користувачеві у будь-який момент часу отримати доступ до інформації наукових конференцій та згенерувати посилання на тези.

Програма була розроблена за допомогою Android Studio. Код програми знаходиться у додатку Б.

Застосунок був реалізований на таких технологіях: Dart, Flutter, Firebase.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Додаток надає змогу працювати з даними статей наукових конференцій.

А саме:

- Переглядати поточні статті в системі;
- Фільтрувати статті та використовувати систему фільтрації за автором, галуззю;
- Проходити реєстрацію та авторизацію в системі;
- Додавати нові статті у систему;
- Коментувати статті;
- Отримувати посилання на статті;

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Додаток складається з двох основних частин: самого додатку та бази даних на сервісі Firebase.

У додатку є такі структурні розділи: каталог матеріалів, система фільтрації матеріалів, система реєстрації та входу, особистий кабінет користувача, особистий кабінет автора, особистий кабінет адміністратора.

Користувач може переглядати інформацію, отримувати посилання на неї, залишати коментарі.

Автор може все те ж саме, що і користувач та має можливість додавання та редагування власних статей.

Адміністратор має право редагувати та керувати інформацією будь-яких користувачі.

ВИКОРИСТАНІ ТЕХНІЧНІ ЗАСОБИ

Для користування додатком користувач має мати мобільний телефон з системою Android (версії 7.0 та вище) або ios (версії 8.0 та вище). Обов'язковою вимогою є доступ до мережі інтернет.

Першочергово програму потрібно інсталиувати за допомогою apk файлу. Після цього програма з'явиться у стандартному меню програм.

Для авторизації потрібно вказати свою електронну пошту.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Для того, щоб мати можливість користуватися додатком необхідно першочергово інсталиувати програму за допомогою арк файлу. Після інсталяції програма буде відкриватися з меню додатків.

Перед використанням додатку необхідно пройти реєстрацію за допомогою своєї електронної скриньки. Після цього користувач отримує доступ до функціоналу читача або автора.

ВХІДНІ ТА ВИХІДНІ ДАНІ

Вхідними даними є особисті дані, котрі користувач вказує при реєстрації та файли, котрі він додає до сховища, коментарі.

Вихідними даними є каталог файлів з їх характеристиками та коментарями, каталог авторів, посилання на конкретні тези.