

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСТОСУНКУ ДЛЯ ВЕДЕННЯ,
ВПОРЯДКУВАННЯ ТА УПРАВЛІННЯ ОСОБИСТИХ
ТА РОБОЧИХ ЗАПИСІВ
SOFTWARE IMPLEMENTATION OF AN APPLICATION TO KEEP,
ORGANIZE, AND MANAGE PERSONAL AND WORK RECORDS

Виконав(ла): студент(ка) 4 курсу, групи КНТ-121
Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ДУРОВ М.С.

(ПРИЗВИЩЕ та ініціали)

Керівник ШИТІКОВА О.В.

(ПРИЗВИЩЕ та ініціали)

Рецензент ТІМЕНКО К.І.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Факультет КНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. СУББОТІН
 “ ” _____ 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

ДУРОВА Микити Сергійовича
(ПРИЗВИЩЕ, ім'я, по батькові)

Тема проєкту (роботи) Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів. Software Implementation of an Application to Keep, Organize, and Manage Personal and Work Records

керівник проєкту (роботи) канд. техн. наук, ШИТИКОВА Олена Вікторівна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “28” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 02 червня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне

Завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Основні рішення щодо реалізації компонентів системи. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ШИТІКОВА О.В., доцент		
Нормоконтроль	КАЛІНІНА М.В., асистент		

7. Дата видачі завдання «14» квітня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділи 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Микита ДУРОВ
(Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Олена ШИТІКОВА
(Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
91 с., 3 табл., 30 рис., 4 дод., 36 джерел.

ЗАПИСИ, НОТАТКИ, ПЛАНУВАННЯ, ПРОДУКТИВНІСТЬ,
РЕАЛІЗАЦІЯ, ВЕБЗАСТОСУНОК

Об'єкт дослідження – процес програмної реалізації застосунку для ведення, впорядкування та управління особистих та робочих записів.

Предмет дослідження – програмні засоби для застосунку для ведення, впорядкування та управління особистих та робочих записів.

Метою роботи є програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів.

Матеріали, методи та технічні засоби: мова програмування JavaScript, надбудови React та середовища розробки WebStorm.

Результати. Розроблено вебзастосунку для ведення, впорядкування та управління особистих та робочих записів.

Практична цінність роботи полягає у програмній вебзастосунку для ведення, впорядкування та управління особистих та робочих записів.

Висновки. Реалізовано вебзастосунок для ведення, впорядкування та управління особистих та робочих записів.

Галузь використання – користувачі мережі та робітники компаній.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 91 pages, 3 tables, 30 figures, 4 appendixes, 36 sources.

RECORDS, NOTES, PLANNING, PERFORMANCE,
IMPLEMENTATION, WEB APPLICATION

The object of research is the process of software implementation of an application for maintaining, organizing and managing personal and work records.

The subject of the research is software tools for the application for maintaining, organizing and managing personal and work records.

The purpose of this work is to provide a software implementation of an application for maintaining, organizing, and managing personal and work records.

Materials, methods, and technical tools: the JavaScript programming language, React add-ons, and WebStorm development environments.

Results. A web application has been developed for maintaining, organizing, and managing personal and work records.

The practical value of the work lies in the software web application for maintaining, organizing and managing personal and work records.

Conclusions. Implemented a web application for maintaining, organizing, and managing personal and work records.

Scope of use – network users and employees of companies.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ	9
1 Аналіз предметної області	15
1.1 Дослідження предметної області нотування записів	15
1.2 Програмні засоби для ведення записів	19
1.3 Порівняння існуючих аналогів	22
1.4 Постановка завдання	28
1.4.1 Функціональність системи	29
1.4.2 Вимоги до інтерфейсу	31
1.4.3 Вимоги до безпеки	31
1.4.4 Не функціональні вимоги до продукту	32
1.5 Висновки за розділом 1	32
2 Матеріали і методи	33
2.1 Вибір мови програмування	33
2.2 Вибір середовища розробки для стеку JavaScript та React	37
2.3 Структурна схема розробки	40
2.4 Висновки за розділом 2	42
3 Основні рішення щодо реалізації компонентів системи	43
3.1 Проєктування моделей системи	43
3.2 Проєктування інтерфейсу	48
4 Експлуатація, тестування та експериментальне дослідження програми	54
4.1 Призначення й умови застосування програми	54
4.2 Вимоги до інтерфейсу	55
4.3 Умови виконання програми	55
4.4 Методика та результати тестування	56
Висновки	60
Перелік джерел посилання	61
Додаток А Технічне завдання	65

Додаток Б Опис програми	71
Додаток В Текст програми	78
Додаток Д Слайди презентації	84

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

- GUI – Graphical User Interface – користувацький графічний інтерфейс;
IDE – інтегроване середовище розробки;
UML – Unified Modeling Language;
ОС – операційна система;
ПЗ – програмного забезпечення;
ПК – персональний комп'ютер;
СУБД – системи управління базами даних.

ВСТУП

З огляду на те, що у сучасну цифрову епоху, коли інформації не тільки багато, але вона ще й має основоположне значення для продуктивності, здатність ефективно вести, систематизувати і управляти особистими і робочими записами важлива як ніколи. Перехід від традиційної паперової документації до цифрового ведення записів призвів до величезних змін у тому, як приватні користувачі та організації обробляють інформацію. Актуальність впровадження програмних рішень для цієї мети підкреслюється зростаючим обсягом даних, необхідністю доступу в режимі реального часу, необхідністю створення безпечних і централізованих систем і загальним прагненням до підвищення ефективності [1]-[2].

Сучасна людина поєднує безліч обов'язків: від ведення медичних записів і фінансової звітності до дотримання термінів виконання проєктів і складання професійної документації. Кожний з цих обов'язків породжує потік даних та записів, які потрібно зберігати, отримувати та оновлювати з часом. Без надійного та систематичного підходу важлива інформація може бути легко втрачена, неправильно заповнена або просто недоступна, якщо це необхідно [1]-[2].

В професійній діяльності ця проблема посилюється. Співробітники, підприємці та фрілансери часто керують кількома проєктами, взаємодіють з різними стейкхолдерами та повинні відстежувати ефективність, терміни та кінцеві результати. При відсутності надійного програмного рішення управління таким робочим навантаженням стає непосильним і небезпечним через можливі помилки. Отже, програми, спеціально розроблені для організації та управління записами, не просто корисні – вони необхідні [1]-[2].

За останні два десятиліття обсяг цифрової інформації, яка щодня генерується і споживається, збільшився в геометричній прогресії. Сучасний фахівець постійно взаємодіє з цифровими даними – від електронних листів і заміток про зустрічі до відсканованих документів і онлайн-формулярів. За

оцінками Всесвітнього економічного форуму та інших організацій, що спеціалізуються на даних, до 2026 року в світі буде вироблятися більше 463 екзабайт даних в день. Управління таким потоком даних неможливо без структурованих цифрових інструментів [3].

Така терміновість обумовлена не тільки обсягом, але і швидкістю отримання необхідної інформації. Будь то отримання податкових звітів за минулий рік, рахунки-фактури клієнта шестимісячної давності або довідки про щеплення дитини для зарахування до школи, затримки, викликані поганим веденням документації, можуть мати серйозні наслідки [3].

Пандемія COVID-19 прискорила тенденцію до віддаленої роботи та децентралізованих команд. Оскільки співробітники зараз працюють у різних географічних місцях, потреба в доступних хмарних інструментах управління записами стала критичною. Централізований підхід, заснований на програмному забезпеченні, гарантує, що кожен користувач має доступ до однієї і тієї ж версії документів, може співпрацювати в режимі реального часу і підтримувати продуктивність навіть в умовах географічної розосередженості [3]-[4].

Ця терміновість посилюється очікуваннями миттєвого зв'язку та доступу до інформації. Клієнти, колеги та керівники очікують своєчасних відповідей та оновлень, які можуть бути надані лише в тому випадку, якщо інформація добре організована та доступна миттєво [3].

Як приватні особи, так і організації стикаються зі зростаючим тиском, що вимагає захисту конфіденційної інформації. Для приватних осіб це може означати захист особистих ідентифікаційних номерів, паролів та фінансових документів. Для організацій це означає дотримання нормативних стандартів, таких як GDPR, HIPAA або PCI-DSS. Належні програмні рішення забезпечують шифрування, контроль доступу, резервне копіювання та інші функції, які забезпечують захист записів від несанкціонованого доступу та втрати даних [4].

Неналежне ведення документації може призвести до порушень нормативних вимог, судових санкцій або шкоди репутації. Таким чином, впровадження безпечного програмного забезпечення (ПЗ) для управління записами більше не є обов'язковим — це юридична та оперативна необхідність [4]-[5].

Паперові документи є екологічно нестійкими. Виробництво, зберігання та утилізація паперу значною мірою сприяють погіршенню стану навколишнього середовища. Оскільки проблеми зміни клімату зростають, а організації прагнуть досягти цілей сталого розвитку, зменшення використання паперу шляхом оцифрування стає пріоритетним завданням. Програмні програми, які керують цифровим зберіганням документів, необхідні для цього переходу [4]-[5].

Застосунки для управління записами – це не просто цифрові картотеки, це інтелектуальні багатофункціональні інструменти, які підвищують продуктивність, точність і дозволяють приймати проактивні рішення [4]-[5].

Ці програми надають структуровані формати для зберігання різних типів даних, класифікованих за датою, проектом, клієнтом або будь-якими іншими значущими показниками. Такі функції, як позначення, пошук та фільтрація, дозволяють користувачам швидко отримувати інформацію. Замість того, щоб копатися в стосах документів або сотнях неорганізованих цифрових файлів, користувачі можуть отримати доступ до того, що їм потрібно, за лічені секунди [4]-[5].

Наприклад, фрілансер, який використовує застосунок для управління проектами та ведення документації, може миттєво знаходити рахунки за минулі періоди, журнали спілкування з клієнтами та результати, пов'язані з будь-яким проектом. Так само студент може використовувати такі інструменти для організації курсових робіт, дослідницьких матеріалів та термінів виконання [4]-[5].

Багато сучасних програм управління записами інтегровані з системами календарів, нагадуваннями та списками завдань. Це дозволяє користувачам не

тільки зберігати інформацію, а й продуктивно взаємодіяти з нею. Програмне забезпечення, яке нагадує користувачам про майбутні терміни, відстежує час, витрачений на виконання завдань, або навіть створює резюме з нотаток, може значно підвищити продуктивність [4]-[5].

У професійних умовах інтегровані системи дозволяють керівникам відстежувати ефективність роботи співробітників, забезпечувати дотримання термінів і оптимізувати розподіл ресурсів. Економія часу при організації вручну безпосередньо призводить до підвищення ефективності [5].

Особисті та професійні дані часто є конфіденційними. Програми, призначені для управління записами, зазвичай включають такі функції безпеки, як:

- двофакторна аутентифікація;
- управління доступом на основі ролей;
- наскрізне шифрування;
- безпечне резервне копіювання в хмарі.

Це гарантує, що лише авторизовані користувачі матимуть доступ до певних файлів і що дані не будуть втрачені через збій обладнання, випадкове видалення або шкідливі дії [4]-[5].

У командних налаштуваннях програми для управління записами полегшують спільну роботу. Команди можуть працювати над спільними документами, залишати коментарі та відстежувати зміни в режимі реального часу. Це особливо цінно для віддалених команд, дослідницьких груп та робочих середовищ проєкту [6].

Сучасні програми часто включають функції, керовані штучним інтелектом, такі як:

- сканування документів та оптичне розпізнавання символів;
- автоматизована класифікація;
- пропозиції за шаблонами;
- узагальнення;
- повідомлення про застарілі або неповні записи.

Ці функції не тільки скорочують ручні зусилля, але і підвищують якість ведення записів. Інтелектуальні пропозиції та аналітика можуть навіть допомогти краще зрозуміти поведінку користувачів та оптимізувати робочий процес [6].

Оскільки більшість програм для управління записами пропонують хмарні рішення, користувачі можуть отримувати доступ до своїх даних з будь-якого місця та з будь-якого пристрою. Така мобільність має вирішальне значення для сучасного світу, орієнтованого на мобільні пристрої. Незалежно від того, подорожуєте ви у справах, відвідуєте віддалену зустріч чи просто працюєте вдома, ви маєте безперебійний доступ до важливої інформації [3].

Інтеграція з хмарою також забезпечує автоматичне резервне копіювання та контроль версій, запобігаючи випадковій втраті даних та забезпечуючи безперервність передачі інформації [4]-[5].

Різні користувачі мають різні потреби. Студент коледжу має інші вимоги до управління записами, ніж власник малого бізнесу. На щастя, більшість програм пропонують власні шаблони, інформаційні панелі та плагіни для різних випадків використання. У міру зростання відповідальності користувача – особистої чи професійної – застосунок може відповідно масштабуватися, інтегруючись з іншими інструментами, такими як бухгалтерське ПЗ, CRM-платформи або системи управління навчанням [4]-[5].

Незважаючи на величезні переваги, впровадження цих програмних систем пов'язане з певними труднощами:

- опір впровадженню – люди, які звикли до традиційних методів, можуть не захотіти перемикатися;
- необхідність навчання – ефективне використання таких додатків часто вимагає певної підготовки;
- фінансові наслідки – хоча багато програм пропонують безкоштовні версії, розширені функції часто приховані за платними програмами;
- питання конфіденційності – користувачі повинні вибирати перевірених постачальників, щоб забезпечити етичне поводження з даними.

Незважаючи на ці проблеми, довгострокові переваги значно переважають короткострокові незручності [4], [6].

Впровадження ПЗ для управління особистими і професійними записами більше не розкіш, а нагальна необхідність. Темп сучасного життя, масштаб цифрової інформації та очікування щодо ефективності, безпеки та доступності – все це вказує на майбутнє, в якому цифрове управління документами стане основоположним. Ці програми дозволяють окремим людям вести більш організоване, продуктивне та безпечне життя, а підприємствам та організаціям підтримувати високий рівень операційної ефективності та відповідності законодавству. Оскільки цифрова трансформація продовжує прискорюватися, використання таких інструментів стане ключем до того, щоб чітко і впевнено орієнтуватися в складнощах світу [2].

Для розробки нового та актуального рішення проблеми організації робочих записів необхідно виконати наступні завдання:

- дослідити предметну область ведення, впорядкування та управління особистими та робочими записами;
- дослідити існуючі програмні застосунки та реалізації способів вирішення проблеми ведення, впорядкування та управління особистими та робочими записами;
- дослідити переваги та недоліки існуючих програмних рішень;
- спроектувати та розробити нове рішення для ведення, впорядкування та управління особистими та робочими записами;
- провести тестування;
- на основі виконаної роботи впорядкувати висновки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Роботу в першому розділі буде присвячено аналізу предметної області ведення, впорядкування та управління особистими та робочими записами, в тому числі за допомогою програмних засобів. Головна мета роботи на цьому етапі – на основі виконаного аналізу предметної області сформулювати комплекс вимог до нового застосунку для ведення, впорядкування та управління особистими та робочими записами.

1.1 Дослідження предметної області нотування записів

Ведення нотатків завжди допомагало людям внести певну фіксацію в їхнє життя, будь то складання списку покупок, запис в щоденнику або складання нагадувань під час наради. Така поведінка не нова та корениться в нашій природній схильності надавати сенс світу, втілюючи наші думки у фізичному вигляді. Ведення записів – це звичка, рутина і часто необхідна стратегія подолання труднощів повсякденного життя. Це приносить ясність, порядок і навіть душевний спокій. Ведення нотаток, як по роботі, так і з особистих питань, дає людям відчуття контролю і допомагає їм осмислювати те, що відбувається навколо [7].

Однією з причин, чому людям зручно робити нотатки, є те, що це дуже особиста та гнучка діяльність. Строгих правил щодо того, як слід робити нотатки, не існує. Деякі люди вважають за краще робити нотатки від руки, насолоджуючись тактильними відчуттями від дотику ручки до паперу. Інші віддають перевагу цифровим нотаткам, серед яких легше виконати пошук необхідної інформації, також їх легше впорядковувати та зазвичай це доступно на різних пристроях. Будь то маркери, ментальні карти, голосові нагадування або кольорові наклейки, люди, природно, знаходять свій власний ритм і систему. Така свобода робить ведення записів безпечним і навіть приємним заняттям [7].

З психологічної точки зору, ведення записів допомагає позбутися від ментального безладу. Людська пам'ять обмежена. Коли люди намагаються утримати в голові занадто багато інформації (завдань, зустрічей, ідей, планів) – це стає непосильною справою. Запис допомагає звільнити когнітивні ресурси. Замість того, щоб намагатися запам'ятати все, люди можуть глибше зосередитися на завданні. Просто знання того, що інформація десь записана, зменшує занепокоєння та запобігає витоку інформації з пам'яті [8].

Ще одна причина, чому ведення записів здається природним, полягає в тому, що це створює відчуття готовності. На роботі люди роблять нотатки під час зустрічей, мозкових штурмів або читання звітів. Ці нотатки служать для запису того, що було сказано, які рішення були прийняті і що потрібно зробити далі. Це створює основу, на яку люди можуть посилатися. Це зменшує кількість непорозумінь. Наявність детальних нотаток часто означає, що ви можете бути на крок попереду, будь то підготовка до наступної зустрічі або завершення роботи над результатом [8].

В особистому житті ведення записів служить багатьом цілям. Люди використовують нотатки, щоб керувати своїм часом, планувати свій робочий тиждень, запам'ятовувати дні народження, відстежувати звички та приймати важливі життєві рішення. Ведення щоденника, яке є однією з форм ведення записів, допомагає людям аналізувати свої думки та емоції. Запис того, що відбувається, надає форму абстрактним почуттям. Це полегшує розуміння проблем та пошук рішень. Для багатьох сам процес написання має терапевтичне значення – це спосіб розібратися в ментальному хаосі і зрозуміти, що дійсно важливо – виконати певну рефлексію [8].

Ведення записів також допомагає у встановленні цілей та довгостроковому плануванні. Коли люди записують свої цілі – вони стають більш реальними. Нотатки служать нагадуванням про ці цілі та допомагають людям відстежувати прогрес. Вони можуть озирнутися назад і побачити, як далеко вони зайшли. Це створює відчуття мотивації і виконаного обов'язку.

Без письмових нагадувань про цілі легко забути або відмовитися від них. Нотатки допомагають зосередитися і підвищити відповідальність [9].

У сучасному напруженому світі на людей постійно обрушується потік інформації. Ведення нотаток допомагає людям зрозуміти шум і виділити те, що важливо. Це може бути ідея, яка виникла під час прогулянки, щось надихаюче з подкасту або проникливий коментар друга. Коли люди мають звичку записувати ці речі, вони створюють особисту базу знань. З часом ці невеликі фрагменти інформації можуть призвести до більш глибоких висновків або творчих проривів. Вони служать основою для майбутніх проєктів або рішень [9].

Люди також знаходять розраду в тому, що роблять нотатки, тому що це створює структуру посеред хаосу. Життя не завжди протікає гладко і впорядковано. Все стає безладним: змінюються плани, трапляються надзвичайні ситуації і змінюються пріоритети. Записи допомагають людям адаптуватися. Добре складений список справ можна скорегувати, реорганізувати та змінити пріоритети. Щоденники і цифрові календарі допомагають людям розставити все по місцях і знизити ймовірність забути щось важливе. Справа не тільки в плануванні, а й у створенні гнучкої системи, яка допомагає людям залишатися на плаву, навіть коли ситуація стає непередбачуваною [9].

У веденні записів є і культурний аспект. У школах студентів навчають робити нотатки під час лекцій. У професійному середовищі ведення записів часто розглядається як ознака залучення та відповідальності. Це заохочується в книгах самодопомоги, на семінарах з підвищення продуктивності та навіть у терапії. Таке соціальне підкріплення підвищує ймовірність того, що люди засвоять цю звичку і дотримуватимуться її. Деякі люди навіть роблять нотатки під час телефонних розмов, перегляду фільмів чи читання книг просто тому, що це допомагає їм запам'ятовувати та розмірковувати [10].

З появою цифрових інструментів ведення нотаток стало ще більш зручним і ефективним. Такі програми, як Evernote, OneNote, Notion та Google

Кеер, дозволяють легко створювати нотатки, упорядковувати та ділитися ними. Такі функції, як створення тегів, пошук і резервне копіювання в хмарі, роблять його більш ефективним, ніж будь-коли. Користувачі можуть робити нотатки на своєму телефоні під час поїздок на роботу, на ноутбучі під час зустрічей або навіть за допомогою голосових команд. Зручність цих інструментів спонукає людей робити більше нотаток і частіше, що, в свою чергу, підвищує їх організованість [10].

Для тих, хто працює в складних умовах — наприклад, для керівників проєктів, дослідників або креативщиків, — ведення записів не просто корисно, воно необхідно. Це спосіб узагальнення інформації, об'єднання ідей та планування наступних кроків. Ці професіонали часто мають справу з пересічними завданнями, багатьма стейкхолдерами та прийняттям важливих рішень. Нотатки допомагають їм бути в курсі всіх подій. У творчих сферах нотатки стають ескізами ідей, фрагментами натхнення або ранніми ескізами більшого бачення. Це сировина, з якої в кінцевому підсумку виходить готовий продукт [10].

Для людей, які стикаються з проблемами психічного здоров'я, ведення записів також може стати потужним інструментом самодопомоги. Відстеження настрою, журнали симптомів, списки подяки та щоденні роздуми допомагають людям помітити закономірності та тригери. З часом ці нотатки допоможуть зрозуміти, які стратегії працюють, які звички потрібно змінити та коли звернутися за допомогою. Це спосіб залишатися на зв'язку з самим собою і знайти самосвідомість [10].

Студенти, найчастіше, в значній мірі покладаються на нотатки. Від лекцій в аудиторії до навчальних посібників, нотатки необхідні для засвоєння та збереження інформації. Але крім академічних занять, студенти також отримують користь від особистих нотаток-наприклад, для встановлення цілей, відстеження бюджету або роздумів про своє зростання. Ці заняття формують життєві навички, які стануть в нагоді їм і після закінчення школи [11].

Навіть у повсякденних справах корисно робити нотатки. Списки продуктів, інструкції з упаковки та журнали по догляду за будинком можуть здатися простими, але вони допомагають людям ефективніше вести домашнє господарство. Вони економлять час, запобігають помилкам і зменшують втому при прийнятті рішень [11].

У фінансовому менеджменті нотатки допомагають людям відстежувати витрати, складати бюджети та планувати майбутні витрати. Люди часто записують терміни оплати рахунків, встановлюють нагадування про поновлення підписки або зберігають список підписок. Ведення такого роду записів може привести до прийняття більш ефективних фінансових рішень і зменшення числа несподіванок [11].

В решті решт, люди відчують себе комфортно, коли роблять нотатки, тому що це дає їм відчуття самостійності. У світі, який часто здається хаотичним або неконтрольованим, нотатки – це спосіб впевнитися в спроможності реалізації планів. Це невеликі, але значущі дії з організації, підготовки та реалізації намірів. Це інструменти, що дозволяють орієнтуватися в житті з більшою легкістю, усвідомленістю і цілеспрямованістю [11].

1.2 Програмні засоби для ведення записів

Таке ПЗ, як Evernote, OneNote, Notion та Google Keep, стали популярними інструментами для ведення записів, оскільки вони дозволяють легко, гнучко та ефективно записувати, упорядковувати та отримувати доступ до інформації [12]. Хоча кожний з цих застосунків має свій власний дизайн і функції, всі вони мають загальні принципи роботи. Користувачі можуть використовувати їх для введення нотаток, складання списків, збереження вебвмісту, додавання зображень, запису аудіо тощо. Вони працюють та іноді навіть синхронізуються на декількох пристроях, що означає, що користувачі можуть зробити запис на своєму телефоні, а потім отримати доступ до нього з

комп'ютера або планшета. Ця зручність допомогла цим застосункам стати звичною частиною того, як люди організовуються на роботі, в школі чи в повсякденному житті [12].

Як правило, програми для ведення записів дозволяють користувачам створювати окремі нотатки або записи, які зберігаються в папках, блокнотах або робочих областях. Ці цифрові нотатки можуть містити звичайний текст, контрольні списки, зображення та навіть посилання. Користувачі можуть легко редагувати та впорядковувати їх. На відміну від традиційних паперових блокнотів, цифрові програми дозволяють швидко шукати, додавати теги та синхронізувати дані. Якщо хтось хоче знайти запис, зроблений два місяці тому, він може просто ввести ключове слово в рядок пошуку і миттєво знайти його. Це економить час і дозволяє уникнути розчарувань, пов'язаних з перегортанням сторінок або втратою паперових записів [12].

Однією з причин, чому ці програми настільки популярні сьогодні, є те, що вони вирішують справжню проблему — відстежують безліч думок, завдань та фрагментів інформації, з якими люди стикаються щодня. Люди зайняті як ніколи, і однієї пам'яті недостатньо, щоб впоратися з усім. Ці програми допомагають людям залишатися організованими, надаючи їм центральне місце для зберігання інформації. Замість того, щоб використовувати наклейки для нотаток, чернетки електронних листів та випадкові текстові повідомлення, користувачі можуть зберігати все в порядку та шукати в одному місці [13].

Інша причина полягає в тому, що вони доступні на різних пристроях і платформах. Незалежно від того, чи хтось використовує Android, iPhone, Mac чи Windows, більшість із цих програм синхронізуються між пристроями через хмару. Це означає, що користувачі можуть робити нотатки на своєму телефоні під час поїздок на роботу, а потім відкривати їх на своєму ноутбукі на роботі. Така плавна синхронізація створює відчуття безперервності і надійності. Багато з цих програм також працюють в автономному режимі, тому користувачі можуть робити нотатки, навіть якщо вони не мають підключення до Інтернету, і синхронізувати їх пізніше [13].

Спільні перегляд та редагування також є важливою причиною широкого поширення цих інструментів. Багато з них дозволяють користувачам обмінюватися нотатками або блокнотами з іншими людьми. У командному середовищі це означає, що кожен може отримати доступ до нотаток зустрічей, планів проєктів або спільних списків справ у режимі реального часу. В особистих налаштуваннях члени сім'ї можуть ділитися списками покупок або планами подорожей. Така можливість спільної роботи робить ці програми корисними як на роботі, так і вдома [13].

Налаштування та гнучкість також відіграють важливу роль у їх популярності. Люди використовують застосунки для нотаток по-своєму. Деякі просто пишуть короткі нагадування. Інші створюють цілі системи підвищення продуктивності з детальними шаблонами, кольоровим кодуванням та розпорядком дня. Більшість із цих програм пропонують способи організації інформації за допомогою тегів, папок або таблиць, залежно від того, що найкраще підходить користувачеві. Це означає, що люди можуть створити систему, яка відповідає їхньому стилю мислення, а не підлаштовуватися під якийсь жорсткий інструмент [13].

Ці програми також добре вписуються в більш широку тенденцію цифрового мінімалізму та підвищення продуктивності. Оскільки все більше людей шукають способи впорядкувати своє життя та зменшити розлад мислення, наявність цифрового інструменту, який фіксує думки, плани та інформацію, стає необхідним. Завдяки інтеграції з календарями, електронною поштою, браузерами та програмами для завдань інструменти для створення нотаток стають цифровим центром життя багатьох людей. Вони допомагають об'єднати все разом, позбавляючи від необхідності перемикатися між кількома платформами або інструментами [14].

Іншим фактором є те, що багато з цих програм є безкоштовними або пропонують безліч безкоштовних версій. Це полегшує їх використання без будь-яких зобов'язань. Якщо користувачам потрібні більш вдосконалені функції, такі як збільшення обсягу пам'яті, покращення співпраці або

автоматизація, вони можуть оплатити преміум-плани. Але для багатьох людей основна безкоштовна версія вже досить потужна для щоденного використання [14].

Ці програми також часто оновлюються та вдосконалюються, а це означає, що вони постійно розвиваються відповідно до потреб користувачів. Розробники слухають відгуки та додають такі функції, як темний режим, покращений пошук або інструменти мобільного сканування. Як результат, досвід користувачів продовжує вдосконалюватися, що дозволяє користувачам залишатися залученими та задоволеними [14].

Таким чином, програми для створення нотаток, такі як Evernote, OneNote, Notion та Google Keep, популярні сьогодні, оскільки вони гнучкі, доступні та допомагають людям справлятися з потоком інформації у повсякденному житті. Вони працюють на різних пристроях, підтримують безліч форматів і дозволяють використовувати їх як в особистих цілях, так і для спільної роботи. Кожен додаток має свою особистість та сильні сторони, але всі вони служать одній і тій же основній меті: допомагати людям фіксувати думки, упорядковувати інформацію та бути в курсі завдань та цілей. Оскільки життя стає все більш цифровим і динамічним, наявність надійного інструменту для управління всім цим стало не просто корисним, а й необхідним [14].

1.3 Порівняння існуючих аналогів

Найбільш популярні застосунками для ведення, впорядкування та управління особистими та робочими записами є Evernote, OneNote, Notion та Google Keep. Саме тому, слід їх розглянути більш детально, виконати аналіз їхніх переваг та недоліків, щоб в подальшому розуміти чим має відрізнятися новий застосунок, щоб бути конкурентоспроможним.

Evernote відомий своєю структурою, яка імітує справжні блокноти. У ньому використовуються нотатки, які знаходяться всередині блокнотів

(рис. 1.1) [15]-[16]. Кожний запис може містити текст, зображення, вкладення та посилання. Evernote також включає такі функції, як сканування документів, вебвирізання (збереження вмісту з вебсайтів) та потужні інструменти пошуку, які дозволяють навіть читати текст всередині сканованих зображень. Це робить його особливо корисним для збереження квитанцій, візитних карток та статей. Це улюблений застосунок для людей, які збирають багато інформації з різних джерел і хочуть, щоб все було в одному місці для пошуку [15]-[16].

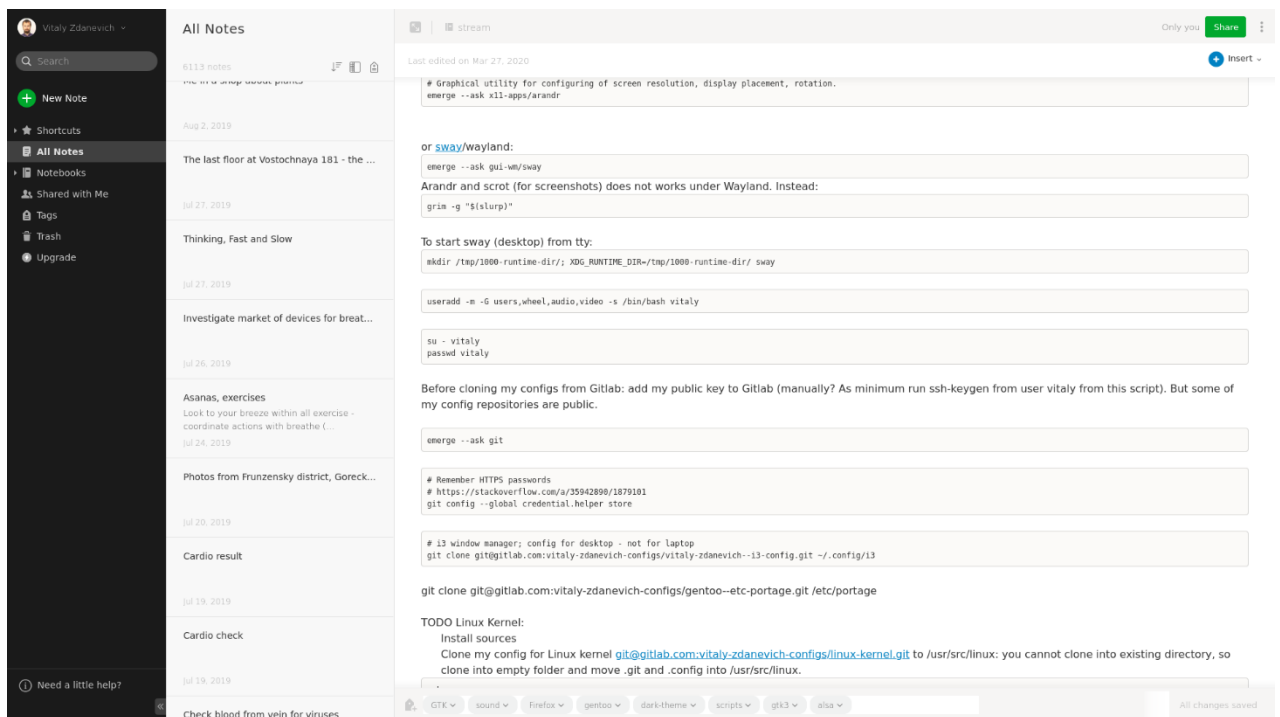


Рисунок 1.1 – Приклад інтерфейсу застосунку Evernote [15]

OneNote, розроблений Microsoft, є цифровою версією фізичного блокнота (рис. 1.2). Це дозволяє користувачам створювати блокноти та створювати в них розділи та сторінки [17]-[18]. Він відрізняється особливою гнучкістю при написанні тексту на сторінці-користувачі можуть друкувати в будь-якому місці, малювати стилусом, вставляти файли або записувати аудіо. OneNote популярний у школах та офісах, особливо тому, що він інтегрується з іншими інструментами Microsoft, такими як Word та Outlook. Він часто використовується студентами для заміток на заняттях і командами, які хочуть співпрацювати в загальних зошитах [17]-[18].

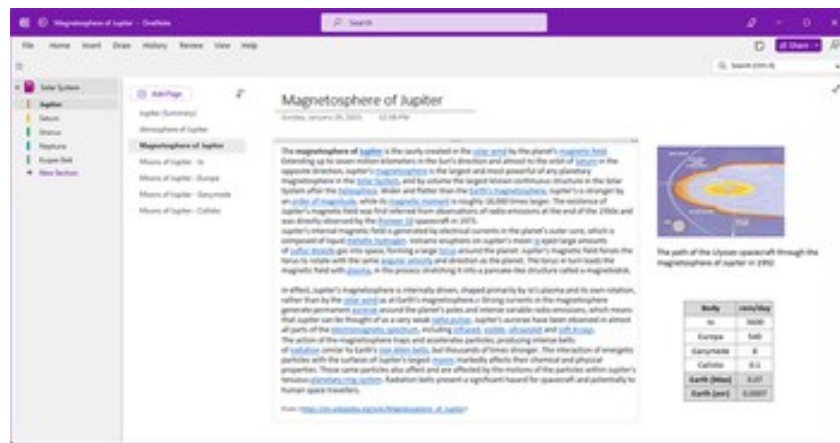


Рисунок 1.2 – Приклад інтерфейсу застосунку OneNote [17]

Notion – це більше, ніж просто застосунок для створення нотаток, це універсальна робоча область, що включає бази даних, управління завданнями, вікі-сторінки та календарі (рис. 1.3) [19]-[20]. Користувачі можуть створювати сторінки з багатим вмістом, такими як контрольні списки, таблиці, перемикачі, календарі, вбудовані відео тощо. Notion легко налаштовується, тому його часто використовують стартапери, творці та планувальники. Люди створюють персональні інформаційні панелі, трекери проєктів, журнали звичок та бібліотеки знань всередині Notion. Налаштування займає трохи більше часу, ніж простіші програми, але як тільки воно налаштоване, воно стає потужним інструментом для організації не лише нотаток, а й цілих робочих процесів [19]-[20].

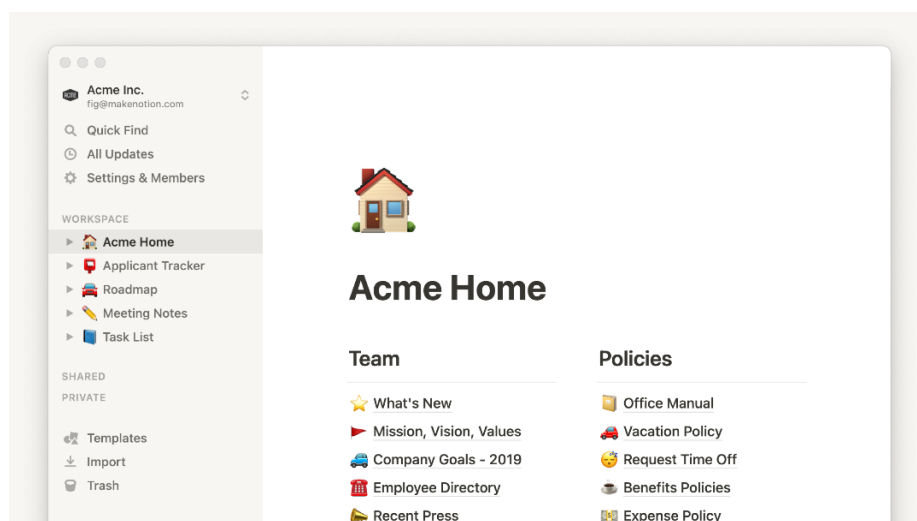


Рисунок 1.3 – Приклад інтерфейсу застосунку Evernote [19]

Google Keep – один з найпростіший застосунків для ведення записів. Він призначений для створення швидких і зручних нотаток і списків справ. Інтерфейс схожий на цифрову дошку для нотаток (рис. 1.4) [21]-[22]. Користувачі можуть позначати нотатки кольором, додавати ярлики, встановлювати нагадування та навіть ділитися нотатками з іншими користувачами для співпраці. Він повністю інтегрований з Google Workspace, тому добре працює з Gmail, Google Документами та Календарем Google. Google Keep особливо підійде для коротких і швидких нотаток, таких як списки покупок, короткі ідеї та нагадування. Це чудове рішення для тих, хто хоче зробити щось швидке та просте без особливих витрат на налаштування [21]-[22].

Для аналітичного висновку, щодо актуальності розробки нового застосунку слід виконати порівняння існуючих аналогів. Наведемо це порівняння у вигляді таблиці – табл. 1.1. Критеріями для порівняння будуть слугувати:

– платформи – визначає, на яких операційних системах та пристроях доступна програма (наприклад, Windows, macOS, Android, iOS, Web);

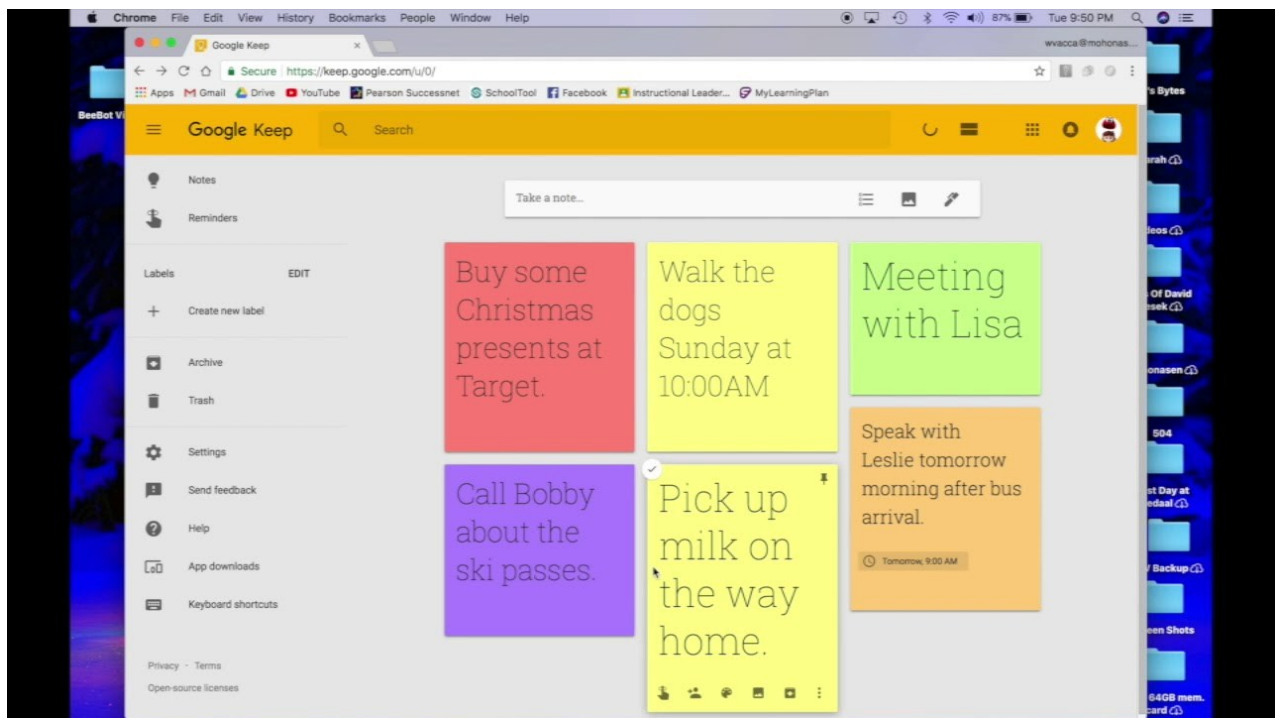


Рисунок 1.4 – Приклад інтерфейсу застосунку Google Keep [21]

- обмеження по обсягу даних (замітки, файли, зображення і т.д.) – скільки користувач можете зберігати інформації, особливо за безкоштовним тарифним планом;
- синхронізація між пристроями – чи дозволяє програма синхронізувати записи на кількох пристроях (телефонах, планшетах, персональних комп'ютерах (ПК)) і наскільки добре вона це робить;
- функція пошуку – наскільки потужним і швидким є пошук;
- систематизація записів — наприклад, папки, блокноти, теги, розділи, ярлики або бази даних користувачів;
- підтримка мультимедіа – чи може користувач вставляти та переглядати мультимедійні матеріали, такі як зображення, малюнки або відскановані документи;
- простота користування – наскільки легко чи складно почати ефективно використовувати застосунок;
- вартість використання – безкоштовні або платні тарифні плани, а також які функції обмежені або розблоковуються при оплаті.

Таблиця 1.1 – Порівняння існуючих аналогів

Критерії порівняння	Evernote	OneNote	Notion	Google Keep
1	2	3	4	5
Платформи	Windows, macOS, Android, iOS, Web	Windows, macOS, Android, iOS, Web	Windows, macOS, Android, iOS, Web	Android, iOS, Web
Обмеження по обсягу даних	Безкоштовно: 60 МБ на місяць; Платно: більше	Безкоштовно: використовується OneDrive (5 ГБ безкоштовно)	Безкоштовно: 5 МБ на сторінку; платні плани для більшої кількості	Безкоштовно зі сховищем Google Drive
Синхронізація між пристроями	Обмежено	Необмежено	Необмежено	Необмежено

Продовження таблиці 1.1

1	2	3	4	5
Функція пошуку	Розширені можливості	Розширені можливості	Обмежені можливості	Обмежені можливості
Систематизація записів	Блокноти, теги	Блокноти, розділи, сторінки	Сторінки, підсторінки, бази даних	Записи, кольори
Підтримка мультимедіа	Зображення, аудіо, PDF-файли, вебвирізки	Зображення, аудіо, малюнки, вкладені файли	Вставки, файли, мультимедіа, блоки коду	Зображення, голосові нотатки, малюнки
Простота користування	Ведення спільних записів та архівування	Записи, рукописний ввід	Управління проєктами, документація, вікі	Швидкі нагадування та списки
Вартість використання	Безкоштовні плани, плани Plus та професійні плани	Безкоштовно (з обліковим записом Microsoft)	Безкоштовні, персональні професійні, командні, корпоративні плани	Повністю безкоштовний

Виконуючи порівняння існуючих програмних застосунків можна відмітити деякі спільні та окремі недоліки в кожному з аналогів. Відтак, Evernote має чітке розмежування за тарифами на використання застосунку між пристроями. Більш того, на деяких платформах продуктивність застосунку нижче, через неякісну адаптацію різних версій застосунків. Загалом же враховуючи обмеження по функціоналу, підписка для повноцінного користування виглядає досить дорогою.

У OneNote користувачі досить часто жаліються на проблеми із синхронізацією. До того ж система організації записів менш зрозуміла та логічно досить жорстко прив'язана до інших застосунків Microsoft. Окремо варто відзначити, що користувацький графічний інтерфейс (Graphical User Interface – GUI) може здаватися перевантаженим.

Notion – це окремий застосунок, що не прив'язаний до жодної з платформ та екосистем (на кшталт, Microsoft чи Google Workspace).

Відповідно для більшості користувачів процес навчання є досить складним. Також застосунок відрізняється дещо обмеженими автономними можливостями (особливо на мобільних пристроях) – знову ж таки через якість адаптації та оптимізації. Також, досить цікаво, що для ведення простих записів в форматі швидких нотатків сервіс точно не підходить, та виглядає переобтяженим.

Google Keep – дуже простий застосунок, тому навпаки тут досить часто бракує розширених функцій. Наприклад, в застосунку доже слабкі можливості щодо організації записів (відсутні механізми категоризації та впорядкування за теками, папками, розділами, тощо). Відсутні можливості розширеного форматування або підтримки великомасштабних проєктів.

Відповідно актуальною задачею залишається програмна реалізація нового зручного та оптимізованого застосунку для ведення, впорядкування та управління особистими та робочими записами.

1.4 Постановка завдання

Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів в форматі вебзастосунку є актуальним завданням у зв'язку зі зростаючим обсягом цифрової інформації, з якою люди щодня стикаються. В умовах віддаленої роботи, змішаних графіків і напруженого способу життя окремі люди і професіонали потребують централізованої, доступної платформи для ефективного зберігання заміток, завдань, документів і нагадувань. Існуючим інструментам часто не вистачає гнучкості або спеціальних функцій, необхідних користувачам, що призводить до фрагментації робочих процесів та зниження продуктивності. Добре розроблений застосунок, адаптований до сучасних потреб, може покращити організацію та оптимізувати повсякденну діяльність. Новий застосунок також має сприяти більш ефективному управлінню часом і прийняттю рішень. У міру зростання перевантаження даними та розвитку цифрових звичок надання

безпечного, зручного у використанні та адаптованого вебзастосунку стає не просто корисним, а й необхідним для підтримки рівноваги та ефективності як в особистому, так і в професійному житті.

Відповідно, можна сформулювати мету даної роботи – програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів.

1.4.1 Функціональність системи

Усі функціональні вимоги до нового застосунку, для ведення, впорядкування та управління особистих та робочих записів, представимо у вигляді категоріального списку за сферами роботи майбутнього застосунку [23]-[25]:

а) управління обліковими записами користувачів:

- 1) користувачі можуть створювати, оновлювати та видаляти облікові записи;
- 2) користувачі можуть безпечно входити та виходити з системи;
- 3) повинні бути доступні параметри скидання пароля та відновлення;

б) панель моніторингу та огляду:

- 1) відображення резюме останніх нотаток, завдань та дій;
- 2) відображення нагадувань, майбутніх подій і зазначених елементів;
- 3) можливість налаштування макета панелі моніторингу (наприклад, віджетів, закріплених нотаток);

в) створення записів і управління ними:

- 1) користувачі можуть створювати особисті записи або записи, пов'язані з роботою;

2) записи можуть містити текст, зображення, посилання, вкладення та контрольні списки;

3) записи можна редагувати, видаляти, дублювати або архівувати;

г) організація та систематизація записів:

1) користувачі можуть групувати записи в папки, блокноти або проєкти;

2) підтримка тегування записів;

3) можливість фільтрувати та сортувати записи за датою, типом, тегом або пріоритетом;

д) пошук і навігація:

1) повнотекстовий пошук по всіх нотатках і вкладенням;

2) пошук за тегами, написами або назвами папок;

3) недавня історія пошуку та збережені пошукові запити;

е) управління завданнями та поточними справами:

1) користувачі можуть створювати списки завдань з прапорцями;

2) призначати дати виконання і пріоритети завдань;

3) можливість встановлювати повторювані завдання або нагадування;

ж) повідомлення та нагадування:

1) повідомлення про терміни і події в застосунку і по електронній пошті;

2) налаштовані нагадування (наприклад, одноразові, щоденні, щотижневі);

3) функція відключення або повтору нагадувань;

з) мультимедійні файли та вкладення файлів:

1) завантаження зображень, PDF-файлів та файли у записи;

2) перегляд вкладень в застосунку;

3) встановлення обмежень розміру файлів та підтримка поширень форматів файлів;

и) шаблони та налаштування:

1) зумовлені шаблони для заміток про збори, журналів, списків завдань і т. д.;

2) користувачі можуть створювати та зберігати власні шаблони;

3) призначені для користувача теми оформлення, шрифти і колірне кодування для персоналізації;

к) експорт та імпорт:

1) експорт записів у такі формати, як PDF, Markdown або Word;

2) імпорт даних з файлів;

3) створення резервних копій та відновлення даних користувача.

1.4.2 Вимоги до інтерфейсу

Інтерфейс системи повинен бути зрозумілим та зручним для маніпулювання колекціями записів. Всі записи користувача мають бути чітко структуровані та розділені на категорії. Функції пошуку повинні мати досить простий інтерфейс, але без втрати його багатофункціональності. Сторінки реєстрації/авторизації користувача повинні мати динамічні підказки з приводу коректності заповнення форм [17]-[18].

1.4.3 Вимоги до безпеки

Забезпечення конфіденційності та безпеки даних:

- використання гешування паролей користувачів;
- застосування надійних механізмів автентифікації користувачів та безпечної передачі даних.

1.4.4 Не функціональні вимоги до продукту

Обов'язкові вимоги.

Для сервера: серверна операційна система (ОС); підтримка фреймворків JavaScript, підтримка системи управління базами даних (СУБД) MySQL.

Мінімальні системні вимоги та ПЗ, що необхідні для функціонування серверної частини застосунку:

- продуктивний хостинг;
- підтримка MySQL.

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- ПК або мобільний пристрій з можливістю виходу в Інтернет;
- актуальна версія вебглядача.

1.5 Висновки за розділом 1

Роботу в першому розділі було присвячено аналізу предметної області ведення, впорядкування та управління особистими та робочими записами, в тому числі за допомогою програмних засобів.

Було досягнуто головної мети роботи на цьому етапі – на основі виконаного аналізу предметної області, було сформовано комплекс вимог до нового застосунку для ведення, впорядкування та управління особистими та робочими записами.

Виконана робота допоможе в майбутньому приступити до визначення технічних засобів розробки.

2 МАТЕРІАЛИ І МЕТОДИ

Роботу в другому розділі буде присвячено визначенню технічних засобів розробки. Головна мета роботи на цьому етапі – визначити технологію розробки та технічні засоби необхідні для розробки. Визначення технології та засобів має ґрунтуватися на основі виконаних аналізу та порівняння існуючих підходів до розробки аналогічних застосунків.

2.1 Вибір мови програмування

JavaScript є надважливою ланкою сучасної веброзробки, що дозволяє розробникам створювати динамічні, інтерактивні та адаптивні програми. Її універсальність і широке поширення обумовлені кількома ключовими факторами, які роблять його оптимальним вибором для створення вебзастосунків [26].

JavaScript підтримується всіма основними веббраузерами, що робить його універсальною мовою для сценаріїв на стороні клієнта. Така широка сумісність гарантує, що програми працюватимуть послідовно в різних середовищах, не вимагаючи додаткових плагінів або адаптацій [26].

Завдяки виконанню коду в браузері клієнта JavaScript знижує необхідність взаємодії з сервером, що призводить до прискорення часу відгуку і зниження навантаження на сервер. Ця обробка на стороні клієнта дозволяє оновлювати і взаємодіяти в режимі реального часу, що підвищує зручність роботи з користувачем [26].

Екосистема JavaScript може похвалитися багатьма фреймворками та бібліотеками, такими як React, Angular та Vue.js, які спрощують процеси розробки та пропонують надійні рішення для створення складних додатків. Ці інструменти надають готові компоненти і структури, що прискорюють розробку і забезпечують зручність супроводу [26].

Окрім клієнтських сценаріїв, JavaScript використовується для розробки на стороні сервера в таких середовищах, як Node.js. Ця універсальність дозволяє розробникам використовувати єдину мову для всього стеку застосунків, спрощуючи робочі процеси розробки та сприяючи повторному використанню коду [26].

JavaScript виграє від динамічної та активної спільноти, яка сприяє її постійному розвитку. Регулярні оновлення та розробка нових інструментів та найкращих практик підтримують мову сучасною та ефективною, забезпечуючи її відповідність мінливим потребам веброботи [27].

JavaScript доступний як початківцям, так і досвідченим розробникам. Його синтаксис простий, а велика кількість освітніх ресурсів і підтримка спільноти полегшують новачкам розуміння і застосування в реальних сценаріях [27].

JavaScript легко інтегрується з іншими технологіями і мовами, дозволяючи вбудовувати різні функціональні можливості і сервіси в вебзастосунки. Така сумісність дозволяє розробникам створювати комплексні рішення, що використовують безліч інструментів і платформ [28].

Таким чином, широка браузерна підтримка JavaScript, підвищення продуктивності за рахунок виконання на стороні клієнта, велика екосистема, універсальність, активна спільнота, простота вивчення і можливості інтеграції в сукупності забезпечують йому статус провідної мови для розробки вебзастосунків [28].

Обґрунтування вибору зв'язки JavaScript та React представимо у вигляді таблиці порівняння (табл. 2.1). У якості критеріїв будемо використовувати наступні:

- мова – основна мова (мови) програмування, що використовується в технологічному стеку. Єдиний мовний стек полегшує розробку та обслуговування;

- основна увага приділяється – чи підходить стек для інтерфейсної або серверної частини або для обох застосунків. Допомагає зрозуміти, чи потрібно інтегрувати інші інструменти чи мови для створення повноцінного застосунку;
- поріг входження – наскільки легко чи важко освоїти стек і стати продуктивним з ним. Впливає на час адаптації нових розробників і початкову швидкість розробки;
- продуктивність (інтерфейс і серверна частина) – наскільки добре працює стек з точки зору швидкості та адаптивності, як в інтерфейсі користувача, так і в обробці сервера. Висока продуктивність забезпечує зручність роботи користувачів і ефективне використання ресурсів сервера;
- спільнота та екосистема – розмір та активність спільноти, а також доступність сторонніх бібліотек, інструментів та документації. Більша екосистема означає швидший розвиток, кращу підтримку та більше рішень для поширених проблем;
- швидкість розробки – як швидко розробники можуть перейти від ідеї до запущеної програми. Швидший розвиток – швидший час виходу на ринок та більша гнучкість;
- можливість використання повного стека – чи може стек задовольняти як інтерфейсні, так і серверні потреби. Повнофункціональні пакети полегшують інтеграцію та дозволяють невеликим командам створювати повноцінні програми;

Таблиця 2.1 – Порівняння мов програмування

Критерій	JavaScript + React	Python + Django
Мова	JavaScript (на стороні клієнта та сервера за допомогою Node.js)	Python (лише на стороні сервера; потрібен JS для інтерфейсу)
Основна увага приділяється	Інтерфейс (React), розширюваний до повного стека за допомогою Node.js	Серверна частина (Django), потребує інтерфейсного фреймворку
Поріг входження	Помірний (синтаксис JS, JSX, компонентна модель)	Помірний рівень (синтаксис Python простий, структура Django обширна)

Продовження таблиці 2.1

Критерій	JavaScript + React	Python + Django
Продуктивність (frontend)	Високий рівень (React забезпечує швидку та динамічну візуалізацію інтерфейсу користувача)	Використовує додаткові фреймворки JS для динамічного інтерфейсу
Продуктивність (серверна частина)	Швидкий за допомогою Node.js, асинхронний за своєю природою	Висока продуктивність для застосунків з великим обсягом даних; синхронність за замовчуванням
Спільнота та екосистема	Великий, особливо для веб і мобільних застосунків	Популярна в серверних і наукових спільнотах
Швидкість розробки	Швидкий завдяки повторно використаним компонентам та величезній екосистемі NPM	Швидка для додатків, орієнтованих на дані, завдяки інструментам адміністрування Django
Можливість використання повного стека	Так (React + Node.js + Express + MongoDB = середній стек)	Частково (потрібен інтерфейс на JS, такий як Reactor Vue.js)

Вибір було зроблено на користь стеку JavaScript + React, через низку переваг. Зокрема, це уніфікований мовний стек: JavaScript можна використовувати як для frontend (React), так і для серверної частини (Node.js), що зменшує перемикання та полегшує роботу. Високоінтерактивний користувацький інтерфейс: React відмінно підходить для створення динамічних, адаптивних інтерфейсів. Екосистема React (Next.js, Redux, React Native) забезпечує швидку розробку в Інтернеті і на мобільних пристроях з використанням загальних кодових баз. Архітектура React, заснована на компонентах, забезпечує модульність і зручність супроводу коду. Node.JS має сучасні інтерфейсні бібліотеки, що спрощують створення функцій реального часу (наприклад, чат, оперативні оновлення) і підвищують продуктивність.

2.2 Вибір середовища розробки для стеку JavaScript та React

WebStorm – це комерційне інтегроване середовище розробки (IDE), розроблене компанією JetBrains, спеціально розроблене для JavaScript, TypeScript та сучасної веброзробки [29], [30]. Воно підтримує широкий спектр готових технологій, включаючи:

- React, Angular, Vue;
- Node.js, Express;
- HTML, CSS, Sass, Less;
- TypeScript, CoffeeScript;
- Jest, Mocha (тестування);
- Git, GitHub та інші інструменти VCS.

На відміну від легких редакторів, таких як Visual Studio Code, WebStorm є повноцінною IDE, що означає, що він пропонує повноцінне середовище розробки без необхідності встановлювати безліч сторонніх розширень [29]-[30].

Універсальне середовище веброзробки. WebStorm включає вбудовану підтримку інтерфейсних і серверних платформ, контроль версій, інструменти тестування і системи збірки. Економить час, усуваючи необхідність пошуку та налаштування розширень [29]-[30].

Підтримує контекстно-залежне автозаповнення, глибокий статичний аналіз і розширену навігацію по коду. Спочатку підтримує такі фреймворки, як React, Vue і Angular, що не вимагає додаткової настройки [29]-[30].

Безпечний рефакторинг JavaScript/TypeScript/HTML / CSS за допомогою автоматизованих інструментів. Вбудований налагоджувач працює як з кодом браузера, так і з Node.js, з точками зупинки, відстеженням та перевіркою змінних [29]-[30].

Інструменти для підвищення продуктивності. Вбудовані інструменти для [29]:

- модульного тестування (наприклад, Jest);

- перевірка коду;
- оперативне редагування з гарячою перезавантаженням;
- клієнт REST для тестування API.

Відмінні можливості командного робочого процесу. Інтеграція з конвеєрами Git, GitHub, ESLint, Prettier, Docker та CI/CD. Добре працює в корпоративних середовищах, де узгодженість та продуктивність важливі [29]-[30].

Інтеграція з екосистемою JetBrains. Якщо розробник використовує PyCharm, IntelliJ IDEA або інші IDE від JetBrains, він оцінить послідовний інтерфейс користувача та загальні функції. Можна керувати всіма інструментами за допомогою JetBrains Toolbox [29]-[30].

Порівняння обраного IDE з існуючими аналогами наведено у табл. 2.2. Серед критеріїв порівняння будемо розглядати наступні:

- готові функції – можливості, доступні відразу після встановлення, не потребують додаткових плагінів. Інструменти з вбудованими функціями економлять час, забезпечують стабільність і спрощують стандартизацію для команд [30];

- підтримка JavaScript/TypeScript – якість функцій, таких як автозаповнення, перевірка типу, виявлення помилок для JS/TS. JavaScript та TypeScript є основою веброзробки – потужна підтримка покращує ефективність кодування та зменшує помилки [30];

- підтримка React – наскільки добре IDE/редактор розуміє та підтримує такі функції React, як JSX, props, компоненти та гачки. React є одним з найбільш часто використовуваних інтерфейсних фреймворків, і його безперебійна підтримка необхідна для підвищення продуктивності [30];

- інтелектуальність коду – наскільки розумним є редактор-автозаповнення, пропозиції та точність навігації по коду. Інтелектуальна допомога коду прискорює написання коду та допомагає запобігти синтаксичним або логічним помилкам [30];

– засоби налагодження – якість вбудованих інструментів для запуску та покрокового перегляду коду з метою пошуку помилок. Хороші засоби налагодження полегшують усунення несправностей та прискорюють розвиток [30];

– інструменти рефакторингу – інструменти, які допомагають безпечно реструктуризувати код (наприклад, перейменовувати змінні, витягувати функції). Рефакторинг має вирішальне значення для підтримки чистоти та масштабованості коду у великих проєктах [30];

– управління версіями – вбудована підтримка таких інструментів, як Git, включаючи історію комітів, перегляд відмінностей та злиття. Контроль версій необхідний у командному середовищі — кращі інструменти полегшують співпрацю [30];

– ціна – чи є інструмент безкоштовним або вимагає ліцензії. Вартість впливає на впровадження, особливо для стартапів, студентів або при складанні бюджету підприємства [30].

Таблиця 2.2 – Порівняння IDE

Критерій	Visual Studio Code	WebStorm
Готові функції	Легке ядро, багато функцій вимагають розширень	Повнофункціональна середовище IDE з самого початку
Підтримка Javascript/Typescript	Хороша, але спирається на плагіни	Хороша, глибоко інтегрована і оптимізована
Підтримка React	Добре працює з плагінами (наприклад, ES7 snippets, React DevTools)	Вбудована підтримка, інтелектуальна обробка JSX/TSX-файлів
Інтелектуальність коду	Залежить від розширень	Високий рівень від початку – більш глибоке розуміння коду
Засоби налагодження	Вбудований базовий налагоджувач; розширюється за допомогою розширень	Вдосконалений вбудований відладчик для інтерфейсної частини та Node.js

Продовження таблиці 2.2

Критерій	Visual Studio Code	WebStorm
Інструменти рефакторингу	Хороший рівень, але на основі плагінів	Найвищий рівень у своєму класі інтелектуальний рефакторинг
Управління версіями	Підтримка Git за допомогою вбудованих інструментів	Розширені інструменти Git + з історією, diff та вдосконаленнями інтерфейсу користувача
Ціна	безкоштовний	платний (але, є студентський формат підписки)

У той час як Visual Studio Code легке й гнучке середовище, та ідеально підходить для початківців розробників або простих налаштувань, WebStorm пропонує більш потужний і безперебійний процес розробки для професійних великомасштабних вебзастосунків, особливо при роботі з React, TypeScript або Node.js у складних проєктах.

2.3 Структурна схема розробки

Загальна схема системи за обраною темою на рис. 2.1.

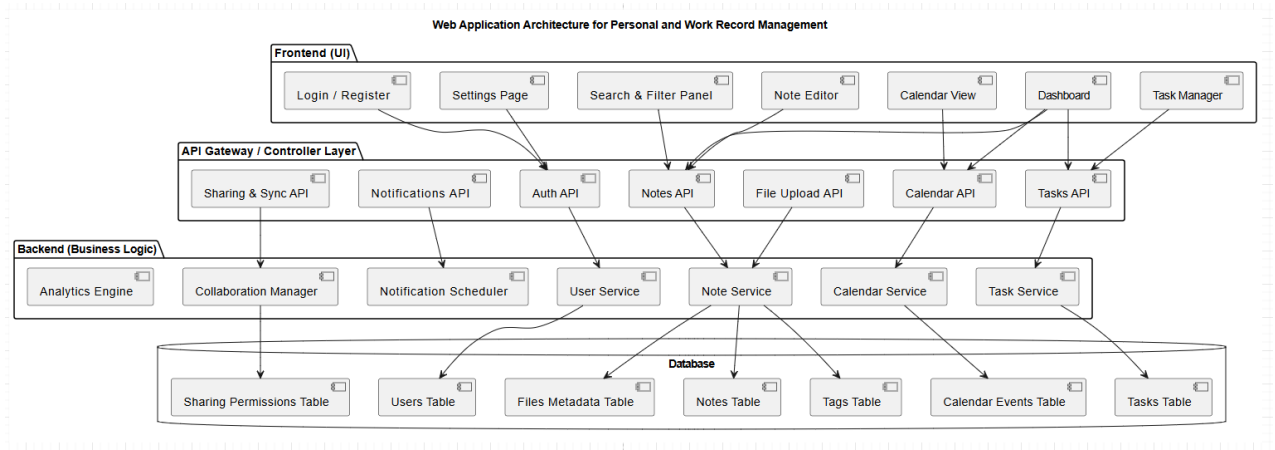


Рисунок 2.1 – Загальна схема системи за обраною темою

Опис компонентів слід розглянути за модулями системи, що планується розробити.

Інтерфейс користувача (Frontend). Створений з використанням таких технологій, як React frontend являє собою інтерактивний і інтуїтивно зрозумілий інтерфейс. Це дозволяє користувачам створювати, переглядати, редагувати та впорядковувати свої записи:

- інтерфейс автентифікації користувача – безпечний вхід та створення облікового запису за допомогою опції 2FA;
- інформаційна панель – огляд завдань, нотаток, нагадувань та останніх дій;
- менеджер записів – повнофункціональний редактор з тегами, вкладеннями файлів і підтримкою форматованого тексту;
- диспетчер завдань – створення списків справ, встановлення термінів та відзначення прогресу;
- календар – візуалізація подій і нагадувань в щомісячному/щотижневому форматі;
- панель пошуку/фільтрації – швидкий доступ до конкретного контенту на основі метаданих або повнотекстового пошуку;
- налаштування – зміна переваг, управління пристроями або перегляд статистики застосунку.

Шлюз API/інтерфейсний контролер. Цей рівень діє як міст між інтерфейсом і серверною частиною. Він забезпечує правильну маршрутизацію запитів і обробляє маркери автентифікації, обмеження швидкості та перевірку запитів.

Серверна частина (рівень бізнес-логіки). Це ядро програми, в якому знаходиться вся логіка, зазвичай вбудована в Node.JS:

- служба користувачів – обробляє дані користувачів, сеанси, шифрування паролів та інформацію про профіль;
- служба записів – керує створенням, редагуванням, видаленням та позначенням вмісту;
- служба завдань і календаря – обробляє повторювані завдання, розклади та інтеграцію;

- служба сповіщень – надсилає нагадування, оновлення або сповіщення електронною поштою/push-повідомленням;
- модуль спільної роботи – забезпечує багатокористувацький доступ і контроль дозволів;
- модуль аналітики – відстежує статистику використання для отримання персоналізованої інформації.

Рівень бази даних. Реляційна база даних зберігає всі структуровані та напівструктуровані дані:

- таблиця Users – містить профіль користувача, дані для входу та налаштування;
- таблиця Notes – вміст, метадані, теги та історію змін;
- таблиця Tasks – списки завдань, статуси і терміни виконання;
- таблиця Calendar – події та інформація про синхронізацію;
- таблиця Tags – користувацькі параметри категоризації;
- таблиця дозволів – інформація про обмін даними та контроль доступу.

2.4 Висновки за розділом 2

Роботу в другому розділі було присвячено визначенню технічних засобів розробки. Головна мета роботи на цьому етапі зводилася до визначення технології розробки та технічних засобів необхідних для розробки. Визначення технології та засобів ґрунтувалося на основі виконаних аналізу та порівняння існуючих підходів до розробки аналогічних застосунків. Від так у якості мови програмування було обрано JavaScript, у якості надбудови – React, а середовищем розробки буде виступати WebStorm.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Проєктування моделей системи

Моделювання за допомогою Unified Modeling Language (UML) вебзастосунків важливо, оскільки воно забезпечує стандартизований візуальний спосіб проєктування та передачі структури й поведінки системи [31]-[32]. Діаграми UML пропонують спільну мову, зрозумілу всім зацікавленим сторонам-розробникам, дизайнерам, бізнес-аналітикам і клієнтам. Вони допомагають подолати розрив між технічними та нетехнічними командами, чітко показуючи, що буде робити система та як вона буде структурована [31].

UML допомагає візуалізувати:

- компоненти та їх взаємозв'язки (діаграми компонентів);
- потоки даних та їх зберігання (діаграми класів та послідовностей);
- взаємодія з користувачем (діаграми варіантів використання);
- стани системи та переходи (діаграми станів).

Це полегшує виявлення недоліків або неефективності проєктування на ранніх стадіях процесу розробки [31].

Діаграми UML служать живою документацією, на яку можна посилатися під час розробки, тестування, супроводу або навіть майбутніх оновлень. Це зменшує припущення та прискорює адаптацію нових членів команди [32].

Використання UML забезпечує логічну структуру проєкту. Ви можете розбити складні системи на менші, керовані компоненти та визначити, як вони взаємодіють. Ця модульність підтримує масштабовану та підтримувану архітектуру [32].

UML досить гнучкий для використання в гнучких середовищах, допомагаючи командам адаптуватися до змін. Діаграми можуть бути швидко оновлені, щоб відобразити зміни в логіці або структурі програми [32].

Багато інструментів розробки (наприклад, Enterprise Architect, Visual Paradigm або StarUML) підтримують UML і навіть можуть генерувати базовий код на основі діаграм. Це прискорює створення прототипів і зменшує кількість помилок при перекладі з дизайну в код [32].

Таким чином, UML покращує розуміння, планування, координацію та якість розробки вебзастосунків, що робить його найкращою практикою для серйозних програмних проєктів [32].

Діаграма класів на рис. 3.1.

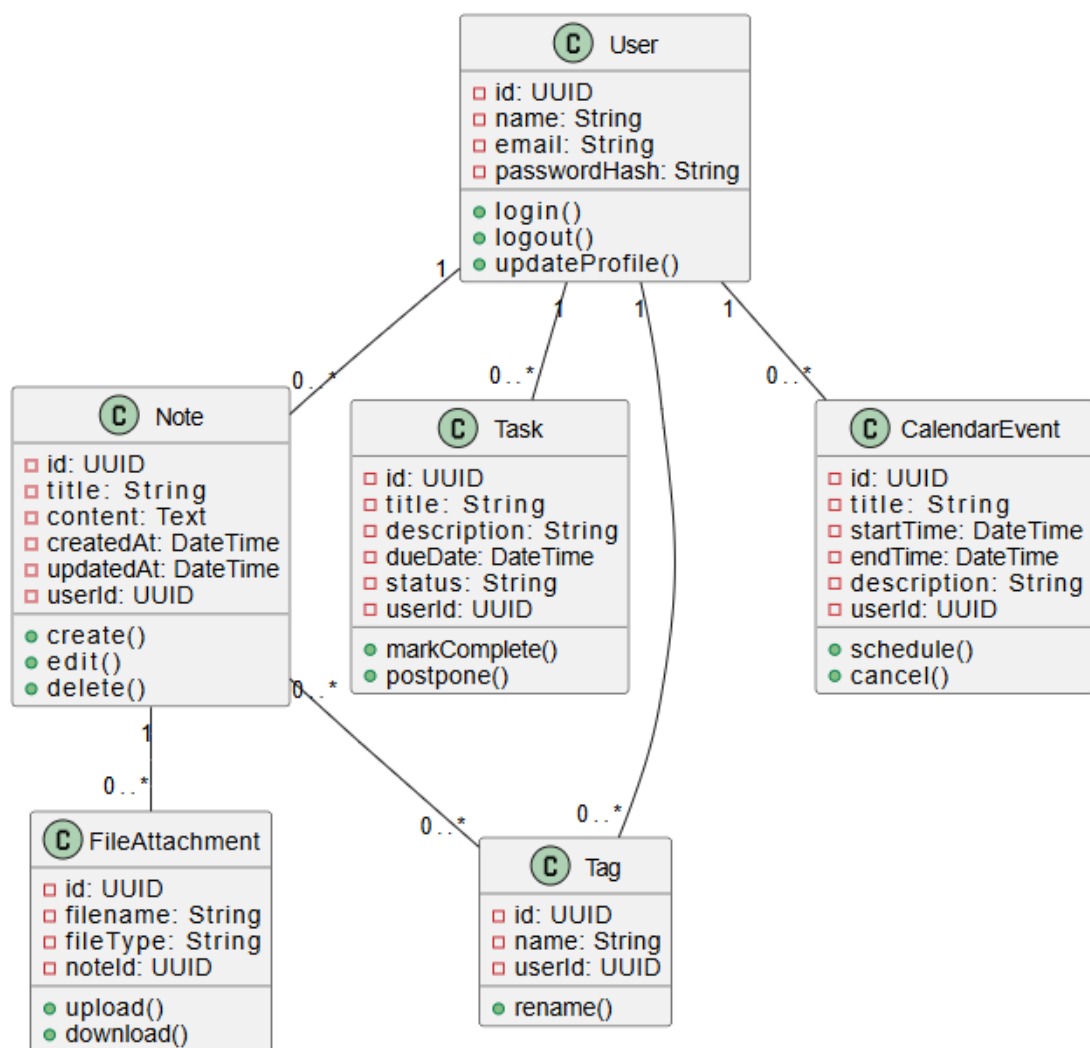


Рисунок 3.1 – Діаграма класів

Ця діаграма класів представляє основну структуру програми:

- користувач – центральний об'єкт, що представляє користувача, який використовує застосунок. Користувач може мати кілька нотаток, завдань, тегів та подій календаря;

- запис – запис, пов'язаний з особистою або робочою діяльністю, з вмістом і тимчасовими мітками. До записів можуть бути прикріплені файли і теги;

- завдання – являє собою елемент поточної роботи зі статусом і датою виконання;

- тег – використовується для класифікації нотаток для кращої організації. Записи можуть бути позначені кількома тегами;

- подія календаря – представляє заплановані події із зазначенням часу початку і закінчення;

- вкладений файл – файли (документи, зображення), пов'язані з записом.

Діаграма прецедентів наведена на рис. 3.2. На цій діаграмі показано, як користувач взаємодіє з системою управління записами. Основні дії (варіанти використання) включають:

- реєстрація/логін: управління доступом до системи;
- створення/редагування/видалення записів: користувачі можуть керувати текстовими записами для особистих або робочих потреб;

- позначення записів тегами: призначення категорії або ключові слова для покращення пошуку та організації;

- завантаження вкладень: додавання до нотаток такі файли, як зображення або документи;

- створення завдань та управління ними: відстеження поточних завдань із зазначенням термінів і оновлень статусу;

- планування в календарі: користувачі можуть планувати події або зустрічі;

- функції пошуку: швидко віднайти потрібні нотатки або завдання;

- нагадування: повідомлення про майбутні завдання або події.

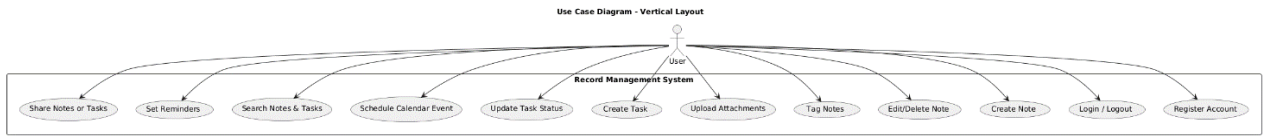


Рисунок 3.2 – Діаграма прецедентів

Діаграма станів наведена рис. 3.3.



Рисунок 3.3 – Діаграма станів системи

Ця діаграма станів моделює основні стани користувача в застосунку.

LoggedOut: початковий стан, коли користувач не проходить автентифікацію:

- включає переходи для спроб входу та обробки помилок або успіхів.

LoggedIn: основний інтерактивний стан після входу, в якому користувачі можуть:

- створювати/редагувати/зберігати записи;
- управління завданнями;
- планування подій календаря;
- перегляд та редагування існуючих записів;
- пошук по записах;
- вийти з системи.

Кожна дія переміщує користувача між певними станами, що представляють потік його взаємодії в застосунку.

Діаграма діяльності на рис. 3.4. Ця діаграма діяльності представляє послідовний потік дій користувача:

а) початок: користувач відкриває застосунок і входить в систему або реєструється;

б) успішність входу: якщо ні, процес завершується з помилкою;

в) основні дії: після входу в систему користувач може:

- 1) переглядати замітки або завдання;
- 2) шукати записи;
- 3) створюйте нові або оновлюйте існуючі нотатки / завдання;
- 4) додайте теги, вкладення або плануйте події / нагадування;
- 5) вихід із системи: завершує сеанс без збоїв.

Діаграма моделює типовий шлях користувача, виділяючи точки прийняття рішень та необов'язкові шляхи.

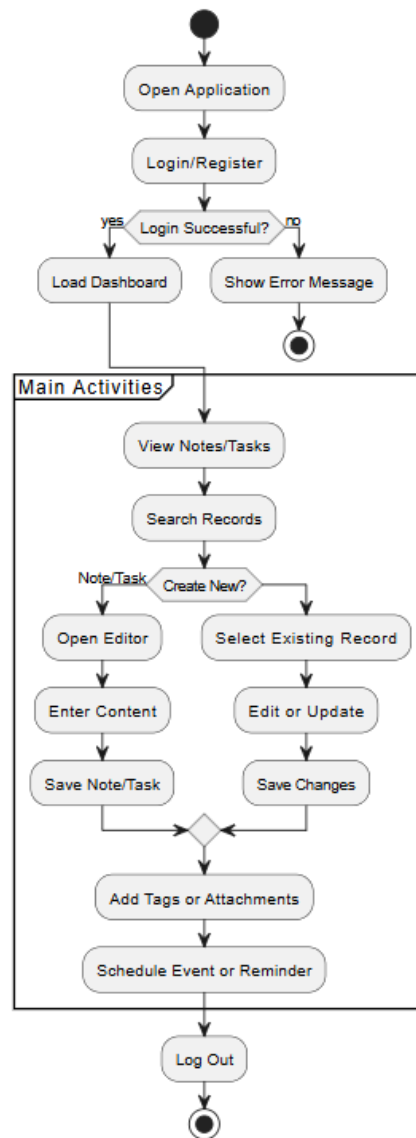


Рисунок 3.4 – Діаграма діяльності

3.2 Проєктування інтерфейсу

Розробка GUI вебзастосунку для ведення, організації та управління особистими та робочими записами – це багатогранний процес, який вимагає продуманого планування та глибокого розуміння поведінки користувача. Інтерфейс повинен бути інтуїтивно зрозумілим, візуально чистим та функціонально ефективним, що дозволяє користувачам легко взаємодіяти з такими функціями, як створення завдань, керування нотатками, планування календаря та вкладення файлів [33].

Перша і, можливо, найважливіша порада – надати пріоритет дизайну, орієнтованому на користувача. Це означає, що необхідно зосередитися на тому, як кінцеві користувачі будуть взаємодіяти із застосунком і які їхні конкретні потреби. Розуміння цих потреб включає дослідження користувачів, створення персоналій та планування їх подорожей. Студент, якому потрібно відстежувати завдання та конспекти лекцій, може взаємодіяти із застосунком інакше, ніж керівник проєкту, який організовує кілька завдань у команді. Інтерфейс повинен відповідати різним робочим процесам, зберігаючи при цьому узгодженість і простоту використання [33].

Зрозуміла навігація – ще один наріжний камінь хорошого планування графічного інтерфейсу. Користувачі повинні мати можливість легко орієнтуватися у застосунку, не допускаючи плутанини. Поширеною стратегією є впровадження бічної або верхньої панелі навігації, яка включає основні розділи програми: інформаційну панель, нотатки, завдання, календар, файли та налаштування. Покрокова навігація також може бути корисною для того, щоб користувачі могли орієнтуватися в глибших частинах системи. Значки меню повинні супроводжуватися текстовими написами для наочності, особливо для початківців користувачів [33].

Макет програми повинен бути модульним. Модульний дизайн розбиває інтерфейс на менші, керовані компоненти або віджети, які можна повторно використовувати та перебудовувати залежно від контексту. Наприклад, модуль для створення нотаток, перегляд календаря і список завдань можуть одночасно розміщуватися на панелі моніторингу. Така компоновка забезпечує гнучкість і індивідуальну настройку. Користувачі можуть захотіти розставити пріоритети для певних віджетів, а модульна конструкція спрощує це. Це також полегшує розробку та можливість масштабування в майбутньому [33].

Функція пошуку – обов'язкова функція для програми для управління записами. З часом у користувачів часто накопичується велика кількість нотаток і завдань, і можливість швидко знаходити конкретні записи має вирішальне значення. У застосунку повинна бути доступна панель пошуку,

розташована на видному місці. Функція пошуку повинна мати можливість індексувати безліч полів, включаючи заголовки, теги та вміст, а також підтримувати фільтри за датою, типом та пріоритетом. Пошукові пропозиції в режимі реального часу можуть ще більше підвищити зручність роботи з користувачем [33].

Форми, що використовуються для введення даних, повинні бути інтуїтивно зрозумілими та швидко реагувати. Незалежно від того, чи створюють користувачі нову нотатку, додають завдання або планують подію в календарі, елементи форми повинні допомагати їм виконувати цей процес без зайвих проблем. Інтелектуальні налаштування за замовчуванням, текст-заповнювач та повідомлення про перевірку допомагають зменшити кількість помилок користувачів. Розширені можливості редагування тексту дозволяють спростити створення нотаток, а випадаючі меню, прапорці і засоби вибору дати спрощують створення завдань і подій [34].

Візуальний дизайн відіграє важливу роль у тому, як користувачі сприймають та взаємодіють зі застосунком. Чистий і послідовний дизайн знижує когнітивне навантаження і робить інтерфейс більш доступним. Використовуйте єдину колірну палітру з достатнім контрастом між елементами. Виділіть основні дії (наприклад, збереження або додавання) акцентними кольорами. Іконки повинні бути легко впізнаваними і витриманими в єдиному стилі. Шрифт повинен бути розбірливим на всіх пристроях, а інтервали повинні бути великими, щоб уникнути безладу [34].

Оперативність важлива, оскільки користувачі можуть отримувати доступ до програми на різних пристроях, включаючи настільні комп'ютери, планшети та смартфони. Макет повинен адаптуватися до різних розмірів екрану без втрати функціональності або зручності використання. Наприклад, на менших екранах бічна панель може перетворюватися на меню гамбургерів, а віджети можуть розташовуватися вертикально, а не поруч. Сенсорні панелі повинні бути достатньо великими для взаємодії з мобільними пристроями, а текст повинен залишатися читабельним без зміни масштабу [34].

Графічний інтерфейс повинен бути розроблений з урахуванням інтеграції. Користувачі можуть захотіти синхронізувати свої завдання з календарем Google або зберігати файли в хмарному сервісі, такому як Dropbox. Інтерфейс повинен спрощувати підключення цих сервісів і управління інтеграцією. Відображення статусів синхронізації, повідомлень про помилки та часу останньої синхронізації дозволяє користувачам бути в курсі подій та контролювати ситуацію [34].

Нагадування та сповіщення необхідні для того, щоб користувачі завжди були в курсі своїх розкладів. Інтерфейс повинен дозволяти користувачам встановлювати нагадування про завдання та події з налаштованими параметрами, такими як дата, час та частота повторення. Сповіщення повинні бути ненав'язливими, але помітними, наприклад, у вигляді невеликих спливаючих вікон або лічильників значків. Центральний центр сповіщень може допомогти користувачам переглядати минулі та майбутні сповіщення [34].

Не можна випускати з уваги доступність. Додаток повинен відповідати стандартам доступності, щоб його могли використовувати люди з обмеженими можливостями. Це включає підтримку зчитувачів з екрана, навігацію з клавіатури та достатню контрастність. Елементи форми повинні бути правильно позначені, а інтерактивні компоненти повинні управлятися за допомогою клавіатури. Також корисно мати панель налаштувань доступності, де користувачі можуть регулювати розмір шрифту або переходити в режим високої контрастності [35].

Налаштування безпеки та конфіденційності повинні бути легко доступними для пошуку та керування за допомогою графічного інтерфейсу. Користувачі повинні бути впевнені в безпеці своїх особистих і робочих даних. Це включає можливість виходу з системи, зміни паролів, двофакторної автентифікації та контролю доступу до спільних нотаток або завдань. Чіткі мітки конфіденційності та індикатори дозволів допомагають зміцнити довіру [35].

Особливої уваги заслуговує оптимізація для мобільних пристроїв. Багато користувачів будуть взаємодіяти з додатком на ходу, і мобільний інтерфейс не повинен бути скороченою версією настільного сайту. Натомість він повинен пропонувати повну функціональність із продуманими змінами дизайну. Жести пальцем можна використовувати для таких дій, як видалення або архівування нотаток. Пріоритет слід віддавати таким основним функціям, як швидке створення заміток, оновлення завдань і перегляд календаря [35].

Механізми зворотного зв'язку з користувачами покращують зручність використання та задоволеність користувачів. Надайте користувачам можливість повідомляти про помилки, пропонувати функції або надавати загальні відгуки. Розгляньте можливість впровадження підказок, покрокових інструкцій або посібників із впровадження, які допоможуть новим користувачам швидко влаштуватися з інтерфейсом. Значки контекстної довідки дозволяють отримати негайну допомогу, не перериваючи робочий процес [35].

Оновлення в режимі реального часу покращують співпрацю та підвищують обізнаність. Якщо програма дозволяє робити спільні нотатки або керувати груповими завданнями, користувачі повинні мати можливість бачити, коли відбуваються зміни. Ці оновлення повинні плавно відображатися в графічному інтерфейсі, не вимагаючи перезавантаження сторінки [35].

Параметри персоналізації можуть зробити додаток більш привабливим та зручним у використанні. Користувачі можуть налаштувати макет своєї інформаційної панелі, вибрати теми або встановити подання за замовчуванням для нотаток та завдань. Графічний інтерфейс повинен підтримувати налаштування користувача, не стаючи надмірно складним. Профілі можуть зберігати ці налаштування та дозволяти користувачам відновлювати їх за замовчуванням, якщо це необхідно [35].

Обробка помилок в інтерфейсі повинна бути зрозумілою і конструктивною. Коли щось піде не так, графічний інтерфейс повинен інформувати користувача про те, що сталося і як це виправити. Наприклад,

якщо файл не вдається завантажити через обмеження розміру, повідомлення повинно пояснити проблему та запропонувати альтернативні варіанти. Уникайте загальних кодів помилок або невизначених попереджень [36].

Нарешті, вкрай важлива узгодженість у всьому додатку. Кнопки, піктограми, кольори та Поведінка у всіх розділах повинні бути однаковими. Така послідовність допомагає користувачам розвивати м'язову пам'ять і зменшує плутанину. Наприклад, одна і та ж кнопка «Зберегти» повинна відображатися в одному місці в модулях «нотатки», «завдання» та «календар» і повинна поводитися однаково кожного разу [36].

Таким чином, планування графічного інтерфейсу для вебпрограми, яка керує особистими та робочими записами, вимагає балансу між естетичністю, зручністю використання, продуктивністю та гнучкістю. Інтерфейс повинен дозволяти користувачам ефективно організовувати своє життя, забезпечуючи при цьому приємний та чуйний інтерфейс на всіх пристроях. При ретельному врахуванні потреб користувачів, доступності, інтеграції та візуальної узгодженості додаток може стати цінним повсякденним інструментом для широкого кола користувачів [36].

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів в форматі вебзастосунку є актуальним завданням у зв'язку зі зростаючим обсягом цифрової інформації, з якою люди щодня стикаються. В умовах віддаленої роботи, змішаних графіків і напруженого способу життя окремі люди і професіонали потребують централізованої, доступної платформи для ефективного зберігання заміток, завдань, документів і нагадувань. Існуючим інструментам часто не вистачає гнучкості або спеціальних функцій, необхідних користувачам, що призводить до фрагментації робочих процесів та зниження продуктивності. Добре розроблений застосунок, адаптований до сучасних потреб, може покращити організацію та оптимізувати повсякденну діяльність. Новий застосунок також має сприяти більш ефективному управлінню часом і прийняттю рішень. У міру зростання перевантаження даними та розвитку цифрових звичок надання безпечного, зручного у використанні та адаптованого вебзастосунку стає не просто корисним, а й необхідним для підтримки рівноваги та ефективності як в особистому, так і в професійному житті.

Відповідно, можна сформулювати мету даної роботи – програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів.

Застосунок має виконувати основні функції, серед яких:

Усі функціональні вимоги до нового застосунку, для ведення, впорядкування та управління особистих та робочих записів, представимо у вигляді категоріального списку за сферами роботи майбутнього застосунку:

- управління обліковими записами користувачів;
- панель моніторингу та огляду;
- створення записів і управління ними;

- організація та систематизація записів;
- пошук і навігація;
- управління завданнями та поточними справами;
- повідомлення та нагадування;
- мультимедійні файли та вкладення файлів;
- шаблони та налаштування;
- експорт та імпорт.

4.2 Вимоги до інтерфейсу

Інтерфейс системи повинен бути зрозумілим та зручним для маніпулювання колекціями записів. Всі записи користувача мають бути чітко структуровані та розділені на категорії. Функції пошуку повинні мати досить простий інтерфейс, але без втрати його багатофункціональності. Сторінки реєстрації/авторизації користувача повинні мати динамічні підказки з приводу коректності заповнення форм [17], [18].

4.3 Умови виконання програми

Обов'язкові вимоги.

Для сервера: серверна ОС; підтримка фреймворків JavaScript, підтримка СУБД MySQL.

Мінімальні системні вимоги та ПЗ, що необхідні для функціонування серверної частини застосунку:

- продуктивний хостинг;
- підтримка MySQL.

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- ПК або мобільний пристрій з можливістю виходу в Інтернет;
- актуальна версія вебглядача.

4.4 Методика та результати тестування

Під час тестування було обрано стратегію перегляду головних активностей застосунку.

Тож на рис. 4.1 наведено головну сторінку застосунку (до авторизації).

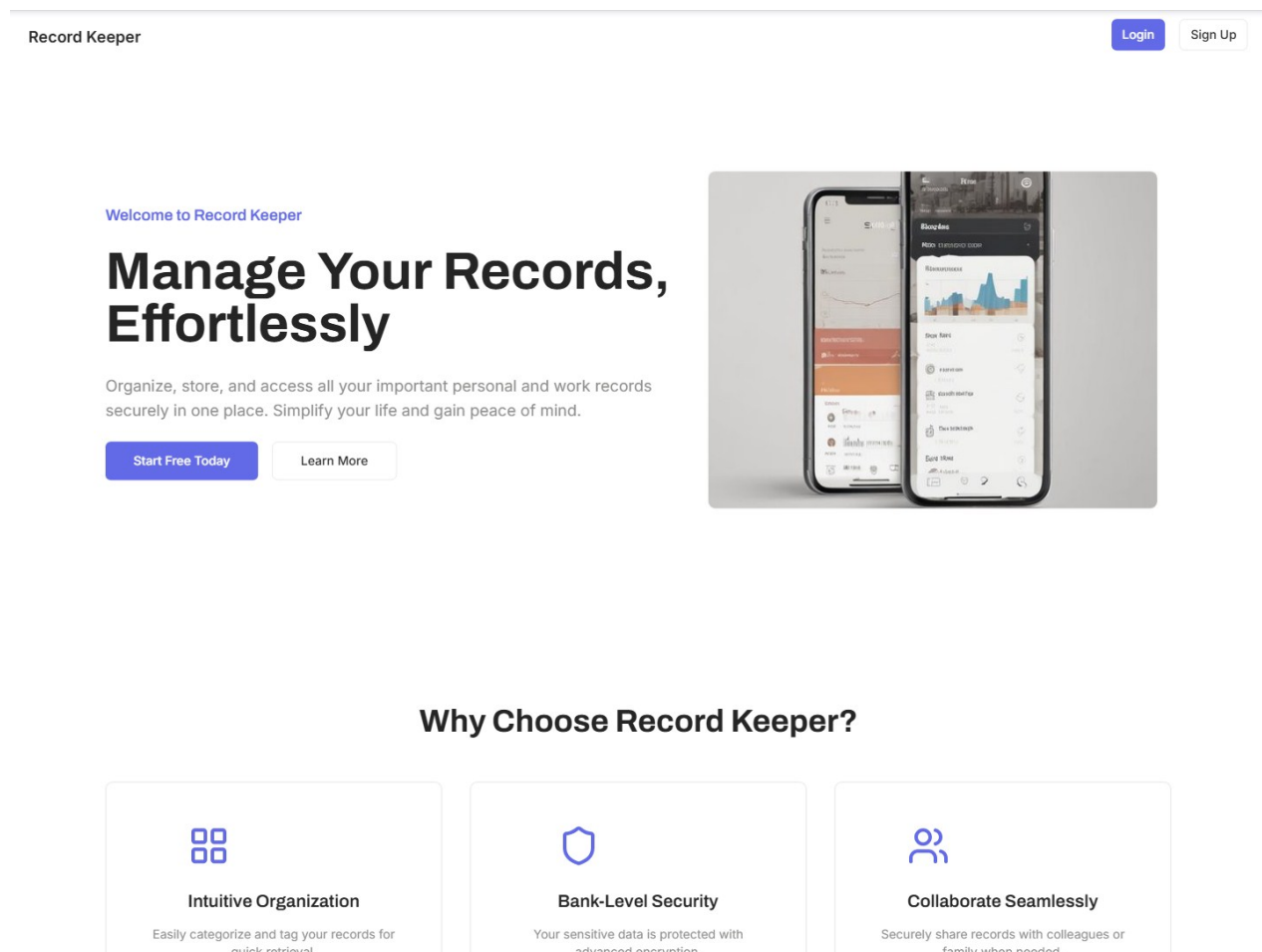


Рисунок 4.1 – Головна сторінка застосунку

На рис. 4.2 наведено рекламу застосунку та його переваг на головній сторінці.

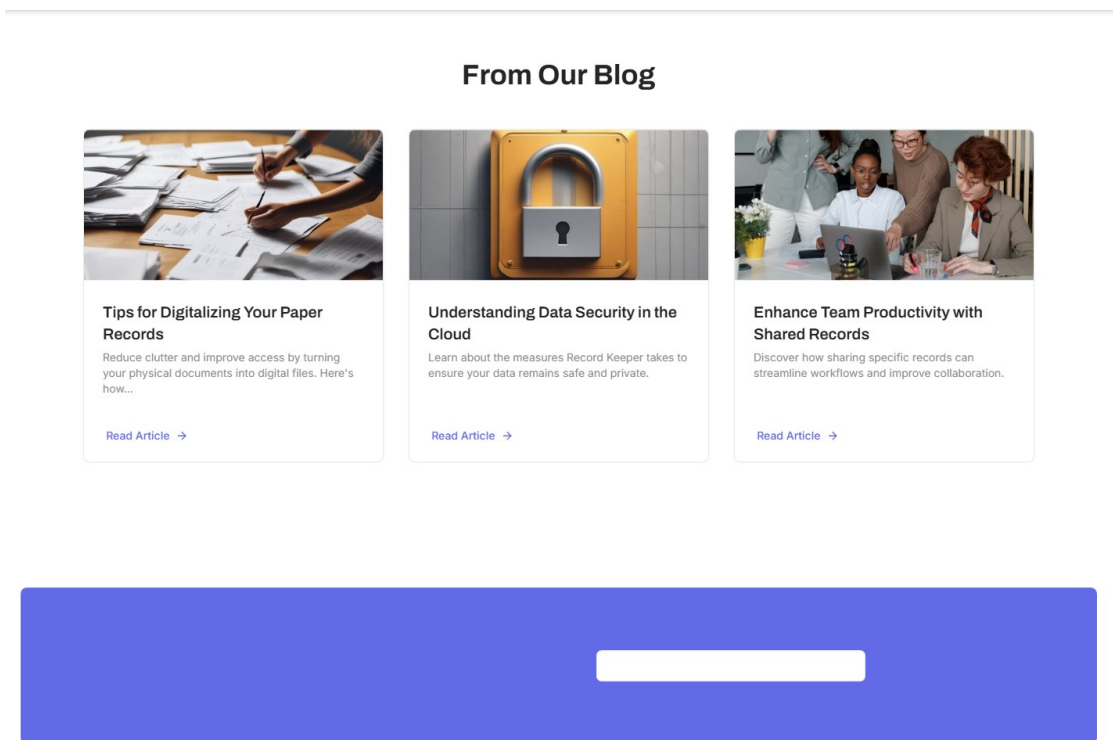


Рисунок 4.2 – Реклама застосунку та його переваг на головній сторінці

На рис. 4.3 представлено особисту сторінку авторизованого користувача.

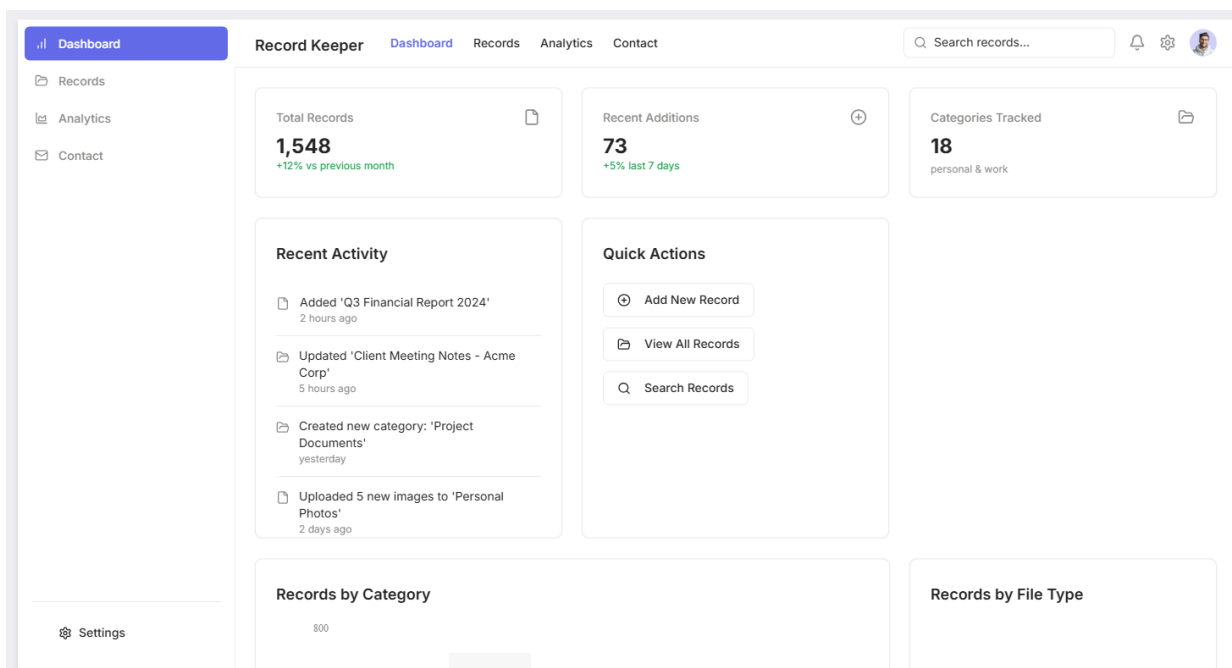


Рисунок 4.3 – Особиста сторінка авторизованого користувача

На рис. 4.4 продемонстровано сторінку з прикладом робочих записів.

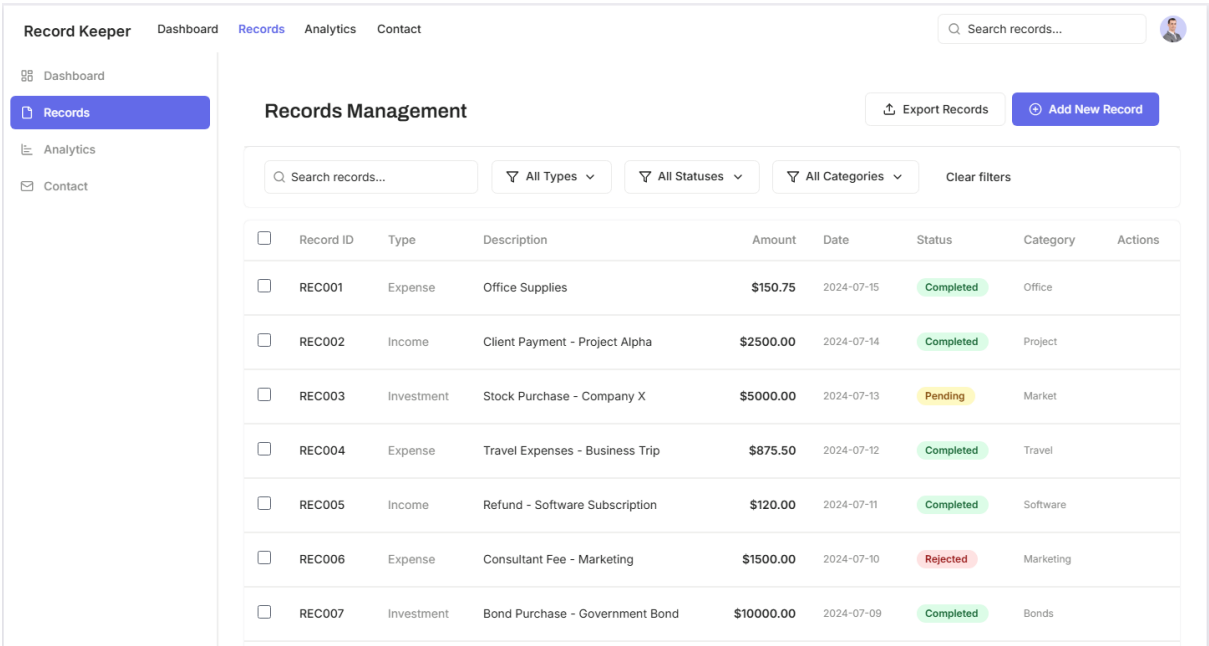


Рисунок 4.4 – Сторінка з прикладом робочих записів

На рис. 4.5 наведено приклад внутрішньої статистики користувача.

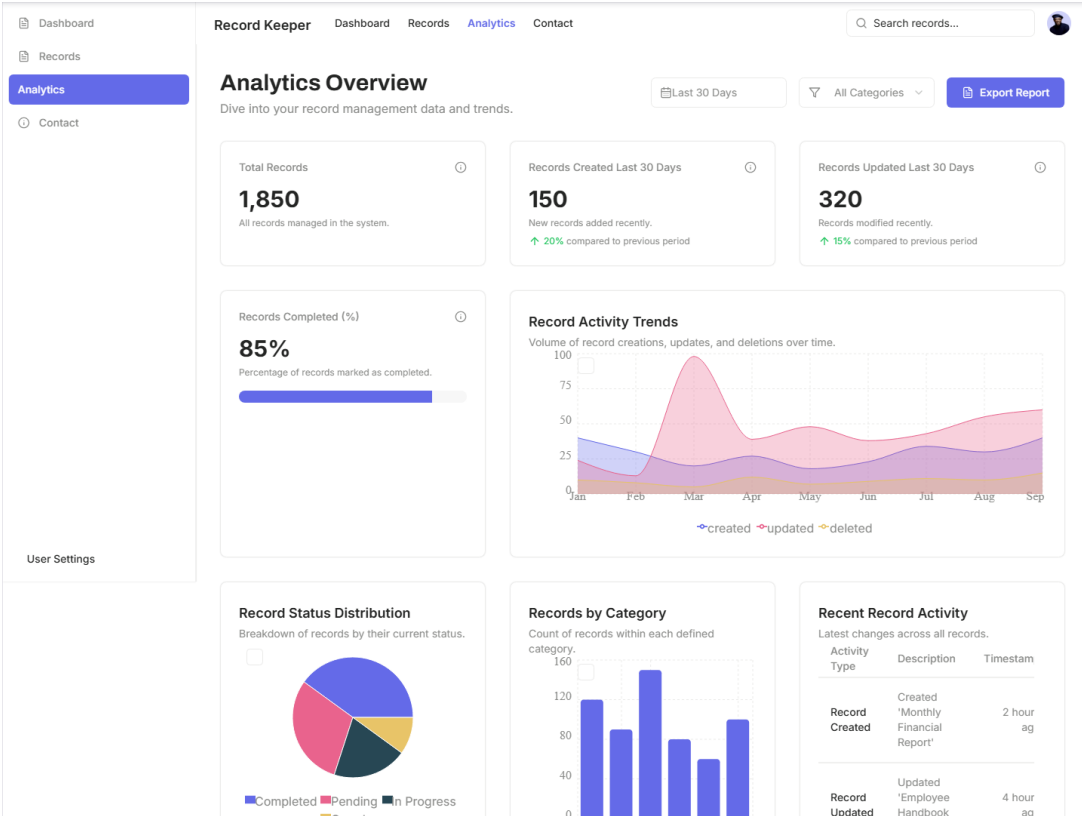


Рисунок 4.5 – Сторінка внутрішньої статистики користувача

На рис. 4.6 наведено форму зворотного зв'язку із розробниками.

Record Keeper [Dashboard](#) [Records](#) [Analytics](#) [Contact](#)

Get in touch today

Your Name

Your Email

Your Phone Number (Optional)

Your Message

Tell us how we can help you

Send Message



123 Record Ln, Data City, DA 98765

support@recordkeeper.com

(800) 555-0199

Frequently Asked Questions

How do I add a new record? ▾

Can I import existing records? ▾

Is my data secure? ▾

How do I update my profile information? ▾

Рисунок 4.6 – Форма зворотного зв'язку із розробниками

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було досягнуто головну мету – виконано програмну реалізації застосунку для ведення, впорядкування та управління особистих та робочих записів. Для цього було виконано низку задач, серед яких:

- досліджено предметну область ведення, впорядкування та управління особистими та робочими записами;
- досліджено існуючі програмні застосунки та реалізації способів вирішення проблеми ведення, впорядкування та управління особистими та робочими записами;
- досліджено переваги та недоліки існуючих програмних рішень;
- спроектовано та розроблено нове рішення для ведення, впорядкування та управління особистими та робочими записами;
- проведено тестування;
- на основі виконаної роботи впорядковано головні висновки.

Розроблений мобільний застосунок був реалізований із використанням мови програмування JavaScript, надбудови React та середовища розробки WebStorm.

Виконано функціональне тестування застосунку на основі логіки взаємодії із застосунком користувача.

На основі розв’язаних задач було отримано закінчений програмний продукт, що в повній мірі реалізує поставлені задачі та в повній мірі відповідає всім вимогам визначеним на початку розробки у вигляді функціональних вимог. Шляхом подальшого розвитку розробки є налаштування доступу до сторонніх застосунків (Google календарю, наприклад) та колективної роботи над записами.

Усі завдання дипломної кваліфікаційної роботи бакалавра повністю виконано.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Productivity Planning Tips + a Step-by-Step Improvement Plan [Electronic resource]. – Access mode: <https://www.erincondren.com/inspiration-center-productivity-planning?srsId=AfmBOoplnoqWKAXZmB8ExAelczYDgI3iZQqA9KuOKgNGQmYa5TPiZ2Ao>.
2. Родащук Г. Ю. Мережеве планування в управлінні ІТ-проєктами [Текст] / Г. Ю. Родащук, С. М. Концеба, Р. І. Ліщук, С. Д. Скуртол // Таврійський науковий вісник. Серія : Технічні науки. - 2023. - Вип. 1. - С. 42-56.
3. Productivity Planning: How To Create a Productivity Plan [Electronic resource]. – Access mode: <https://www.indeed.com/career-advice/career-development/productivity-planning>.
4. How to plan your day for maximum productivity [Electronic resource]. – Access mode: <https://www.productiveblogging.com/how-to-plan-your-day-for-maximum-productivity/>.
5. Unlocking Efficiency: How To Manage Your Day to Day in a Better Way [Electronic resource]. – Access mode: https://commonpurpose.org/resources/blog/how-to-manage-time-better-work?gad_source=1&gad_campaignid=22446664008&gbraid=0AAAAADmzJ0Ms aU-LZ5KxsI5K8GDwKmNp7&gclid=CjwKCAjw24vBBhABEiwANFG7yz1b82sGqMuQLwQcK8YTRA7KcWQ-Jxm-Ly-_U9gudMcsVpp8Sq3UtBoC308QAvD_BwE.
6. 9 Steps to Create an Effective Productivity Plan [Electronic resource]. – Access mode: <https://tivazo.com/steps-to-create-an-effective-productivity-plan/>.
7. How Taking Notes Improves Your Productivity [Electronic resource]. – Access mode: <https://docjulien.medium.com/how-taking-notes-improves-your-productivity-838263cb5a8>.

8. Your Own Personal Time Machine [Electronic resource]. – Access mode: <https://fullfocus.co/recovering-the-lost-art-of-note-taking/>.
9. Note-Taking Strategies to Maximize Your Productivity [Electronic resource]. – Access mode: <https://krisp.ai/blog/note-taking-strategies/>.
10. 3 Note-Taking Methodologies To Help With Productivity [Electronic resource]. – Access mode: <https://www.forbes.com/councils/forbesbusinesscouncil/2021/07/08/3-note-taking-methodologies-to-help-with-productivity/>.
11. Taking Notes to Improve Your Productivity [Electronic resource]. – Access mode: <https://dev.to/calvinmclean/note-worthy-productivity-scribble-your-way-to-success-4e64>.
12. The 6 best note-taking apps in 2025 [Electronic resource]. – Access mode: <https://zapier.com/blog/best-note-taking-apps/>.
13. 10 Best Note-Taking Apps in 2025 (Free & Paid) [Electronic resource]. – Access mode: <https://clickup.com/blog/note-taking-apps/>.
14. The Best Note-Taking Apps for 2025 [Electronic resource]. – Access mode: <https://www.pcmag.com/picks/the-best-note-taking-apps>.
15. Evernote [Electronic resource]. – Access mode: <https://evernote.com/>.
16. How to Use Evernote [Electronic resource]. – Access mode: <https://zapier.com/blog/evernote-tutorial/>.
17. Microsoft OneNote [Electronic resource]. – Access mode: <https://www.onenote.com/?public=1&wdorigin=ondcauth2&wdorigin=ondc>.
18. Microsoft OneNote Review [Electronic resource]. – Access mode: <https://www.pcmag.com/reviews/microsoft-onenote>.
19. Notion [Electronic resource]. – Access mode: <https://www.notion.com/>.
20. What is Notion? And how to use it [Electronic resource]. – Access mode: <https://zapier.com/blog/what-is-notion/>.
21. Google Keep [Electronic resource]. – Access mode: <https://workspace.google.com/products/keep/>.

22. Google Keep [Electronic resource]. – Access mode: <https://www.pcmag.com/encyclopedia/term/google-keep>.

23. Functional Requirements [Electronic resource]. – Access mode: <https://thestory.is/en/process/development-phase/functional-requirements/>.

24. Crafting Comprehensive Website Functional Requirements: A Step-by-Step Guide [Electronic resource]. – Access mode: <https://www.websiteplanningguide.com/comprehensive-website-functional-requirements/>.

25. Functional and Nonfunctional Requirements: Specification and Types [Electronic resource]. – Access mode: <https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/>.

26. What is JavaScript? [Electronic resource]. – Access mode: https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/What_is_JavaScript.

27. Learn JavaScript [Electronic resource]. – Access mode: <https://web.dev/learn/javascript>.

28. JavaScript Tutorial [Electronic resource]. – Access mode: <https://www.w3schools.com/js/>.

29. WebStorm [Electronic resource]. – Access mode: <https://www.jetbrains.com/webstorm/>.

30. WebStorm — An Introduction [Electronic resource]. – Access mode: <https://medium.com/@AlexanderObregon/dive-into-web-development-with-webstorm-a-comprehensive-introduction-742655f86b04>.

31. UML-based Web Engineering Framework for Modeling Web Application [Electronic resource]. – Access mode: <https://scialert.net/abstract/?doi=jse.2011.49.63>.

32. Whitepaper: Modeling Web Application Architectures with UML / [Electronic resource]. – Access mode: https://home.iscte-iul.pt/~hro/RUPSmallProjects/core.base_rup/guidances/whitepapers/modeling_web_application_rchitectures_with_uml_54011136.html.

33. 8 Proven UI Strategies to Craft Engaging Web Apps [Electronic resource]. – Access mode: <https://cygnis.co/blog/visually-stunning-web-apps-design-strategies/>.

34. How To Add Web UI to an Application [Electronic resource]. – Access mode: <https://cdpstudio.com/manual/cdp/studioapiserver/webassembly-new-gui.html>.

35. GUI-O application: Create high-end GUI for any device [Electronic resource]. – Access mode: <https://www.gui-o.com/>.

36. 10 Powerful Python GUI Frameworks for 2024: Simplify Your Desktop App Development [Electronic resource]. – Access mode: <https://fullscale.io/blog/python-gui-frameworks/>.

ДОДАТОК А
Технічне завдання

Вступ

Даний застосунок присвячений для ведення, впорядкування та управління особистих та робочих записів в форматі вебзастосунку.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів», затверджене наказом Національного університету «Запорізька політехніка» № 209 від 28 квітня 2025 р.

A.2 Основні вимоги до програми, що розробляється

Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів в форматі вебзастосунку є актуальним завданням у зв'язку зі зростаючим обсягом цифрової інформації, з якою люди щодня стикаються. В умовах віддаленої роботи, змішаних графіків і напруженого способу життя окремі люди і професіонали потребують централізованої, доступної платформи для ефективного зберігання заміток, завдань, документів і нагадувань. Існуючим інструментам часто не вистачає гнучкості або спеціальних функцій, необхідних користувачам, що призводить до фрагментації робочих процесів та зниження продуктивності. Добре розроблений застосунок, адаптований до сучасних потреб, може покращити організацію та оптимізувати повсякденну діяльність. Новий застосунок також має сприяти більш ефективному управлінню часом і прийняттю рішень. У міру зростання перевантаження даними та розвитку цифрових звичок надання безпечного, зручного у використанні та адаптованого вебзастосунку стає не

просто корисним, а й необхідним для підтримки рівноваги та ефективності як в особистому, так і в професійному житті.

A.2.1 Вимоги до функціональних характеристик

Усі функціональні вимоги до нового застосунку, для ведення, впорядкування та управління особистих та робочих записів, представимо у вигляді категоріального списку за сферами роботи майбутнього застосунку:

а) управління обліковими записами користувачів:

1) користувачі можуть створювати, оновлювати та видаляти облікові записи;

2) користувачі можуть безпечно входити та виходити з системи;

3) повинні бути доступні параметри скидання пароля та відновлення;

б) панель моніторингу та огляду:

1) відображення резюме останніх нотаток, завдань та дій;

2) відображення нагадувань, майбутніх подій і зазначених елементів;

3) можливість налаштування макета панелі моніторингу (наприклад, віджетів, закріплених нотаток);

в) створення записів і управління ними:

1) користувачі можуть створювати особисті записи або записи, пов'язані з роботою;

2) записи можуть містити текст, зображення, посилання, вкладення та контрольні списки;

3) записи можна редагувати, видаляти, дублювати або архівувати;

г) організація та систематизація записів:

- 1) користувачі можуть групувати записи в папки, блокноти або проєкти;
 - 2) підтримка тегування записів;
 - 3) можливість фільтрувати та сортувати записи за датою, типом, тегом або пріоритетом;
- д) пошук і навігація:
- 1) повнотекстовий пошук по всіх нотатках і вкладенням;
 - 2) пошук за тегами, написами або назвами папок;
 - 3) недавня історія пошуку та збережені пошукові запити;
- е) управління завданнями та поточними справами:
- 1) користувачі можуть створювати списки завдань з прапорцями;
 - 2) призначати дати виконання і пріоритети завдань;
 - 3) можливість встановлювати повторювані завдання або нагадування;
- ж) повідомлення та нагадування:
- 1) повідомлення про терміни і події в застосунку і по електронній пошті;
 - 2) налаштовані нагадування (наприклад, одноразові, щоденні, щотижневі);
 - 3) функція відключення або повтору нагадувань;
- з) мультимедійні файли та вкладення файлів:
- 1) завантаження зображень, PDF-файлів та файли у записи;
 - 2) перегляд вкладень в застосунку;
 - 3) встановлення обмежень розміру файлів та підтримка поширень форматів файлів;
- и) шаблони та налаштування:
- 1) зумовлені шаблони для заміток про збори, журналів, списків завдань і т. д.;

2) користувачі можуть створювати та зберігати власні шаблони;

3) призначені для користувача теми оформлення, шрифти і колірне кодування для персоналізації;

к) експорт та імпорт:

1) експорт записів у такі формати, як PDF, Markdown або Word;

2) імпорт даних з файлів;

3) створення резервних копій та відновлення даних користувача.

A.2.2 Вимоги до інтерфейсу програми

Інтерфейс системи повинен бути зрозумілим та зручним для маніпулювання колекціями записів. Всі записи користувача мають бути чітко структуровані та розділені на категорії. Функції пошуку повинні мати досить простий інтерфейс, але без втрати його багатофункціональності. Сторінки реєстрації/авторизації користувача повинні мати динамічні підказки з приводу коректності заповнення форм.

A.2.3 Вимоги до надійності

Забезпечення конфіденційності та безпеки даних:

- використання гешування паролей користувачів;
- застосування надійних механізмів автентифікації користувачів та безпечної передачі даних.

А.2.4 Вимоги до складу та параметрів технічних засобів

Обов'язкові вимоги.

Для сервера: серверна ОС; підтримка фреймворків JavaScript, підтримка СУБД MySQL.

Мінімальні системні вимоги та ПЗ, що необхідні для функціонування серверної частини застосунку:

- продуктивний хостинг;
- підтримка MySQL.

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- ПК або мобільний пристрій з можливістю виходу в Інтернет;
- актуальна версія вебглядача.

ДОДАТОК Б
Опис програми

Б.1 Загальні відомості

Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів в форматі вебзастосунку є актуальним завданням у зв'язку зі зростаючим обсягом цифрової інформації, з якою люди щодня стикаються. В умовах віддаленої роботи, змішаних графіків і напруженого способу життя окремі люди і професіонали потребують централізованої, доступної платформи для ефективного зберігання заміток, завдань, документів і нагадувань. Існуючим інструментам часто не вистачає гнучкості або спеціальних функцій, необхідних користувачам, що призводить до фрагментації робочих процесів та зниження продуктивності. Добре розроблений застосунок, адаптований до сучасних потреб, може покращити організацію та оптимізувати повсякденну діяльність. Новий застосунок також має сприяти більш ефективному управлінню часом і прийняттю рішень. У міру зростання перевантаження даними та розвитку цифрових звичок надання безпечного, зручного у використанні та адаптованого вебзастосунку стає не просто корисним, а й необхідним для підтримки рівноваги та ефективності як в особистому, так і в професійному житті.

Відповідно, можна сформулювати мету даної роботи – програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів.

Б.2 Функціональне призначення

Усі функціональні вимоги до нового застосунку, для ведення, впорядкування та управління особистих та робочих записів, представимо у вигляді категоріального списку за сферами роботи майбутнього застосунку:

а) управління обліковими записами користувачів:

1) користувачі можуть створювати, оновлювати та видаляти облікові записи;

2) користувачі можуть безпечно входити та виходити з системи;

3) повинні бути доступні параметри скидання пароля та відновлення;

б) панель моніторингу та огляду:

1) відображення резюме останніх нотаток, завдань та дій;

2) відображення нагадувань, майбутніх подій і зазначених елементів;

3) можливість налаштування макета панелі моніторингу (наприклад, віджетів, закріплених нотаток);

в) створення записів і управління ними:

1) користувачі можуть створювати особисті записи або записи, пов'язані з роботою;

2) записи можуть містити текст, зображення, посилання, вкладення та контрольні списки;

3) записи можна редагувати, видаляти, дублювати або архівувати;

г) організація та систематизація записів:

1) користувачі можуть групувати записи в папки, блокноти або проєкти;

2) підтримка тегування записів;

3) можливість фільтрувати та сортувати записи за датою, типом, тегом або пріоритетом;

д) пошук і навігація:

1) повнотекстовий пошук по всіх нотатках і вкладенням;

2) пошук за тегами, написами або назвами папок;

3) недавня історія пошуку та збережені пошукові запити;

е) управління завданнями та поточними справами:

1) користувачі можуть створювати списки завдань з прапорцями;

- 2) призначати дати виконання і пріоритети завдань;
 - 3) можливість встановлювати повторювані завдання або нагадування;
- ж) повідомлення та нагадування:
- 1) повідомлення про терміни і події в застосунку і по електронній пошті;
 - 2) налаштовані нагадування (наприклад, одноразові, щоденні, щотижневі);
 - 3) функція відключення або повтору нагадувань;
- з) мультимедійні файли та вкладення файлів:
- 1) завантаження зображень, PDF-файлів та файли у записи;
 - 2) перегляд вкладень в застосунку;
 - 3) встановлення обмежень розміру файлів та підтримка поширень форматів файлів;
- и) шаблони та налаштування:
- 1) зумовлені шаблони для заміток про збори, журналів, списків завдань і т. д.;
 - 2) користувачі можуть створювати та зберігати власні шаблони;
 - 3) призначені для користувача теми оформлення, шрифти і колірне кодування для персоналізації;
- к) експорт та імпорт:
- 1) експорт записів у такі формати, як PDF, Markdown або Word;
 - 2) імпорт даних з файлів;
 - 3) створення резервних копій та відновлення даних користувача.

Б.3 Опис логічної структури

Загальна схема системи за обраною темою на рис. Б.1.

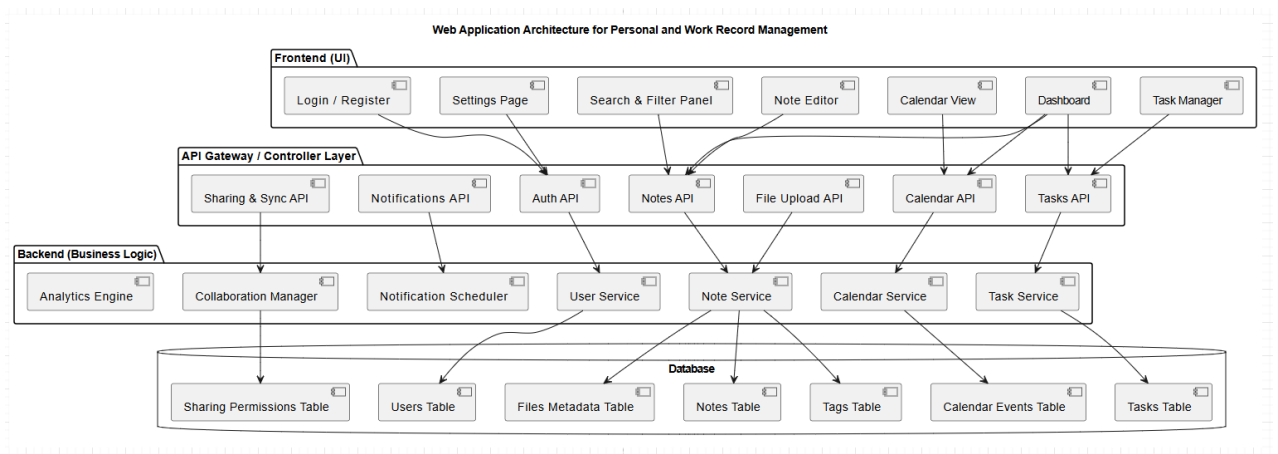


Рисунок Б.1 – Загальна схема системи за обраною темою

Опис компонентів слід розглянути за модулями системи, що планується розробити.

Інтерфейс користувача (Frontend). Створений з використанням таких технологій, як React frontend являє собою інтерактивний і інтуїтивно зрозумілий інтерфейс. Це дозволяє користувачам створювати, переглядати, редагувати та впорядковувати свої записи:

- інтерфейс автентифікації користувача – безпечний вхід та створення облікового запису за допомогою опції 2FA;
- інформаційна панель – огляд завдань, нотаток, нагадувань та останніх дій;
- менеджер записів – повнофункціональний редактор з тегами, вкладеннями файлів і підтримкою форматованого тексту;
- диспетчер завдань – створення списків справ, встановлення термінів та відзначення прогресу;
- календар – візуалізація подій і нагадувань в щомісячному/щотижневому форматі;
- панель пошуку/фільтрації – швидкий доступ до конкретного контенту на основі метаданих або повнотекстового пошуку;
- налаштування – зміна переваг, управління пристроями або перегляд статистики застосунку.

Шлюз API/інтерфейсний контролер. Цей рівень діє як міст між інтерфейсом і серверною частиною. Він забезпечує правильну маршрутизацію запитів і обробляє маркери автентифікації, обмеження швидкості та перевірку запитів.

Серверна частина (рівень бізнес-логіки). Це ядро програми, в якому знаходиться вся логіка, зазвичай вбудована в Node.JS:

- служба користувачів – обробляє дані користувачів, сеанси, шифрування паролів та інформацію про профіль;
- служба записів – керує створенням, редагуванням, видаленням та позначенням вмісту;
- служба завдань і календаря – обробляє повторювані завдання, розклади та інтеграцію;
- служба сповіщень – надсилає нагадування, оновлення або сповіщення електронною поштою/push-повідомленням;
- модуль спільної роботи – забезпечує багатокористувацький доступ і контроль дозволів;
- модуль аналітики – відстежує статистику використання для отримання персоналізованої інформації.

Рівень бази даних. Реляційна база даних зберігає всі структуровані та напівструктуровані дані:

- таблиця Users – містить профіль користувача, дані для входу та налаштування;
- таблиця Notes – вміст, метадані, теги та історію змін;
- таблиця Tasks – списки завдань, статуси і терміни виконання;
- таблиця Calendar – події та інформація про синхронізацію;
- таблиця Tags – користувацькі параметри категоризації;
- таблиця дозволів – інформація про обмін даними та контроль доступу.

Б.4 Використані технічні засоби

Обов'язкові вимоги.

Для сервера: серверна ОС; підтримка фреймворків JavaScript, підтримка СУБД MySQL.

Мінімальні системні вимоги та ПЗ, що необхідні для функціонування серверної частини застосунку:

- продуктивний хостинг;
- підтримка MySQL.

Для експлуатації клієнтської частини необхідні такі програмно-технічні засоби:

- ПК або мобільний пристрій з можливістю виходу в Інтернет;
- актуальна версія вебглядача.

ДОДАТОК В
Текст програми

```

// App state management
const appState = {
  notes: [],
  tasks: [],
  currentView: 'dashboard'
};

// Load from localStorage
function loadAppData() {
  const notes = localStorage.getItem('notes');
  const tasks = localStorage.getItem('tasks');
  if (notes) appState.notes = JSON.parse(notes);
  if (tasks) appState.tasks = JSON.parse(tasks);
}

function saveAppData() {
  localStorage.setItem('notes',
JSON.stringify(appState.notes));
  localStorage.setItem('tasks',
JSON.stringify(appState.tasks));
}

// UI rendering
function renderDashboard() {
  const root = document.getElementById('app');
  root.innerHTML = `
    <h1>Dashboard</h1>
    <button          onclick="renderNoteForm()">New
Note</button>

```

```

        <button                onclick="renderTaskForm()">New
Task</button>

        <div id="notes"></div>
        <div id="tasks"></div>
    `;
    renderNotes();
    renderTasks();
}

```

```

function renderNotes() {
    const                notesDiv                =
document.getElementById('notes');
    notesDiv.innerHTML = '<h2>Notes</h2>';
    appState.notes.forEach((note, i) => {
        notesDiv.innerHTML += `
            <div class="note">
                <strong>${note.title}</strong>
                <p>${note.content}</p>
                <button
onclick="deleteNote(${i})">Delete</button>
            </div>
        `;
    });
}

```

```

function renderTasks() {
    const                tasksDiv                =
document.getElementById('tasks');
    tasksDiv.innerHTML = '<h2>Tasks</h2>';
    appState.tasks.forEach((task, i) => {

```

```

tasksDiv.innerHTML += `
    <div class="task">
        <strong>${task.title}</strong>
        <p>Status:    ${task.done    ?    'Done'    :
'Pending'}</p>
        <button
onclick="toggleTaskStatus(${i})">Toggle</button>
        <button
onclick="deleteTask(${i})">Delete</button>
    </div>
`;
    });
}

function renderNoteForm() {
    const root = document.getElementById('app');
    root.innerHTML = `
        <h2>New Note</h2>
        <form onsubmit="addNote(event)">
            <input            type="text"            id="noteTitle"
placeholder="Title" required><br>
            <textarea                id="noteContent"
placeholder="Content" required></textarea><br>
            <button type="submit">Add Note</button>
            <button                type="button"
onclick="renderDashboard()">Cancel</button>
        </form>
    `;
}

```

```

function renderTaskForm() {
  const root = document.getElementById('app');
  root.innerHTML = `
    <h2>New Task</h2>
    <form onsubmit="addTask(event)">
      <input          type="text"          id="taskTitle"
placeholder="Task Title" required><br>
      <button type="submit">Add Task</button>
      <button                                type="button"
onclick="renderDashboard()">Cancel</button>
    </form>
  `;
}

// CRUD Operations
function addNote(event) {
  event.preventDefault();
  const          title          =
document.getElementById('noteTitle').value;
  const          content        =
document.getElementById('noteContent').value;
  appState.notes.push({ title, content });
  saveAppData();
  renderDashboard();
}

function deleteNote(index) {
  appState.notes.splice(index, 1);
  saveAppData();
  renderDashboard();
}

```

```

    }

    function addTask(event) {
        event.preventDefault();
        const title =
document.getElementById('taskTitle').value;
        appState.tasks.push({ title, done: false });
        saveAppData();
        renderDashboard();
    }

    function deleteTask(index) {
        appState.tasks.splice(index, 1);
        saveAppData();
        renderDashboard();
    }

    function toggleTaskStatus(index) {
        appState.tasks[index].done =
!appState.tasks[index].done;
        saveAppData();
        renderDashboard();
    }

    // App init
    window.onload = function () {
        loadAppData();
        renderDashboard();
    };

```

ДОДАТОК Д
Слайди презентації

Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів

Виконав:	Микита ДУРОВ, студент гр. КНТ-121
Керівник:	Олена ШИТІКОВА, канд. техн. наук

Рисунок Д.1 – Слайд 1

Об'єкт, предмет та мета роботи

Об'єкт дослідження – процес програмної реалізації застосунку для ведення, впорядкування та управління особистих та робочих записів.

Предмет дослідження – програмні засоби для застосунку для ведення, впорядкування та управління особистих та робочих записів.

Метою роботи є програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів.

Рисунок Д.2 – Слайд 2

Порівняння існуючих аналогів

Критерії порівняння	Evernote	OneNote	Notion	Google Keep
Платформи	Windows, macOS, Android, iOS, Web	Windows, macOS, Android, iOS, Web	Windows, macOS, Android, iOS, Web	Android, iOS, Web
Обмеження по обсягу даних	Безкоштовно: 60 МБ на місяць; Платно: більше	Безкоштовно: використовується OneDrive (5 ГБ безкоштовно)	Безкоштовно: 5 МБ на сторінку; платні плани для більшої кількості	Безкоштовно зі сховищем Google Drive
Синхронізація пристроями	між Обмежено	Необмежено	Необмежено	Необмежено
Функція пошуку	Розширені можливості	Розширені можливості	Обмежені можливості	Обмежені можливості
Систематизація записів	Блокноти, теги	Блокноти, розділи, сторінки	Сторінки, підсторінки, бази даних	Записи, кольори
Підтримка мультимедіа	Зображення, аудіо, PDF-файли, веб-вирізки	Зображення, аудіо, малюнки, вкладені файли	Вставки, файли, мультимедіа, блоки коду	Зображення, голосові нотатки, малюнки
Простота користування	Ведення спільних записів та архівування	Записи, рукописний ввід	Управління проектами, документація, вікі	Швидкі нагадування та списки
Вартість використання	Безкоштовні плани, плани Plus та професійні плани	Безкоштовно (з обліковим записом Microsoft)	Безкоштовні, персональні професійні, командні, корпоративні плани	Повністю безкоштовний

3

Рисунок Д.3 – Слайд 3

Постановка завдання

Програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів в форматі вебзастосунку є актуальним завданням у зв'язку зі зростаючим обсягом цифрової інформації, з якою люди щодня стикаються. В умовах віддаленої роботи, змішаних графіків і напруженого способу життя окремі люди і професіонали потребують централізованої, доступної платформи для ефективного зберігання заміток, завдань, документів і нагадувань. Існуючим інструментам часто не вистачає гнучкості або спеціальних функцій, необхідних користувачам, що призводить до фрагментації робочих процесів та зниження продуктивності. Добре розроблений застосунок, адаптований до сучасних потреб, може покращити організацію та оптимізувати повсякденну діяльність. Новий застосунок також має сприяти більш ефективному управлінню часом і прийняттю рішень. У міру зростання перевантаження даними та розвитку цифрових звичок надання безпечного, зручного у використанні та адаптованого вебзастосунку стає не просто корисним, а й необхідним для підтримки рівноваги та ефективності як в особистому, так і в професійному житті.

Відповідно, можна сформулювати мету даної роботи – програмна реалізація застосунку для ведення, впорядкування та управління особистих та робочих записів.

4

Рисунок Д.4 – Слайд 4

Функціональні вимоги

Усі функціональні вимоги до нового застосунку, для ведення, впорядкування та управління особистих та робочих записів, представимо у вигляді категоріального списку за сферами роботи майбутнього застосунку:

- управління обліковими записами користувачів;
- панель моніторингу та огляду;
- створення записів і управління ними;
- організація та систематизація записів;
- пошук і навігація;
- управління завданнями та поточними справами;
- повідомлення та нагадування;
- мультимедійні файли та вкладки файлів;
- шаблони та налаштування;
- експорт та імпорт.

5

Рисунок Д.5 – Слайд 5

Вибір мови програмування

Критерій	JavaScript + React	Python + Django
Мова	JavaScript (на стороні клієнта та сервера за допомогою Node.js)	Python (лише на стороні сервера; потрібен JS для інтерфейсу)
Основна увага приділяється	Інтерфейс (React), розширюваний до повного стека за допомогою Node.js	Серверна частина (Django), потребує інтерфейсного фреймворку
Поріг входження	Помірний (синтаксис JS, JSX, компонентна модель)	Помірний рівень (синтаксис Python простий, структура Django обширна)
Продуктивність (frontend)	Високий рівень (React забезпечує швидку та динамічну візуалізацію інтерфейсу користувача)	Використовує додаткові фреймворки JS для динамічного інтерфейсу
Продуктивність (серверна частина)	Швидкий за допомогою Node.js, асинхронний за своєю природою	Висока продуктивність для застосунків з великим обсягом даних; синхронність за замовчуванням
Спільнота та екосистема	Великий, особливо для веб- і мобільних застосунків	Популярна в серверних і наукових спільнотах
Швидкість розробки	Швидкий завдяки повторно використаним компонентам та величезній екосистемі NPM	Швидка для додатків, орієнтованих на дані, завдяки інструментам адміністрування Django
Можливість використання повного стека	Так (React + Node.js + Express + MongoDB = середній стек)	Частково (потрібен інтерфейс на JS, такий як Reactor Vue.js)

6

Рисунок Д.6 – Слайд 6

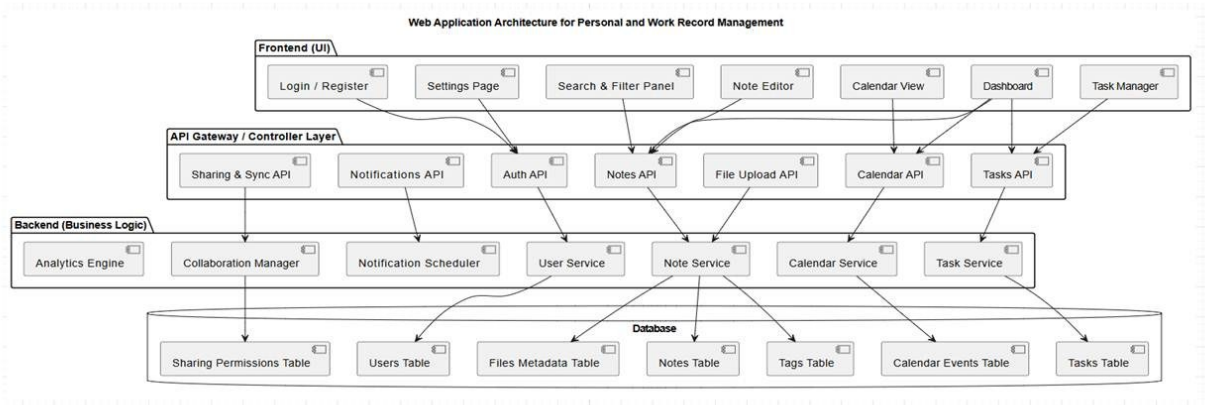
Вибір середовища розробки

Критерій	Visual Studio Code	WebStorm
Готові функції	Легке ядро, багато функцій вимагають розширень	Повнофункціональна середовище IDE з самого початку
Підтримка Javascript/Typescript	Хороша, але спирається на плагіни	Хороша, глибоко інтегрована і оптимізована
Підтримка React	Добре працює з плагінами (наприклад, ES7 snippets, React DevTools)	Вбудована підтримка, інтелектуальна обробка JSX/TSX-файлів
Інтелектуальність коду	Залежить від розширень	Високий рівень від початку – більш глибоке розуміння коду
Засоби налагодження	Вбудований базовий налагоджувач; розширюється за допомогою розширень	Вдосконалений вбудований відладчик для інтерфейсної частини та Node.js
Інструменти рефакторингу	Хороший рівень, але на основі плагінів	Найвищий рівень у своєму класі інтелектуальний рефакторинг
Управління версіями	Підтримка Git за допомогою вбудованих інструментів	Розширені інструменти Git + з історією, diff та вдосконаленнями інтерфейсу користувача
Ціна	безкоштовний	платний (але, є студентський формат підписки)

7

Рисунок Д.7 – Слайд 7

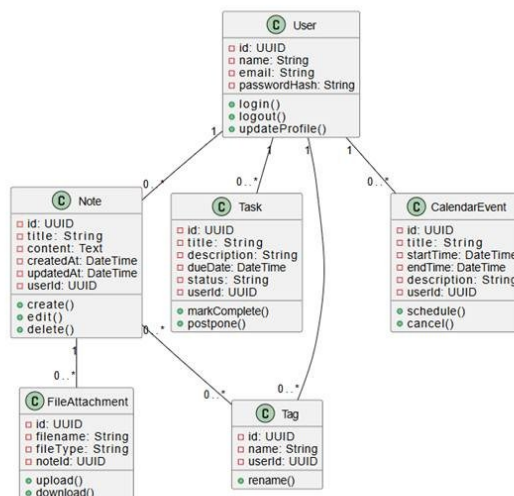
Структурна схема застосунку



8

Рисунок Д.8 – Слайд 8

Фізичне проєктування

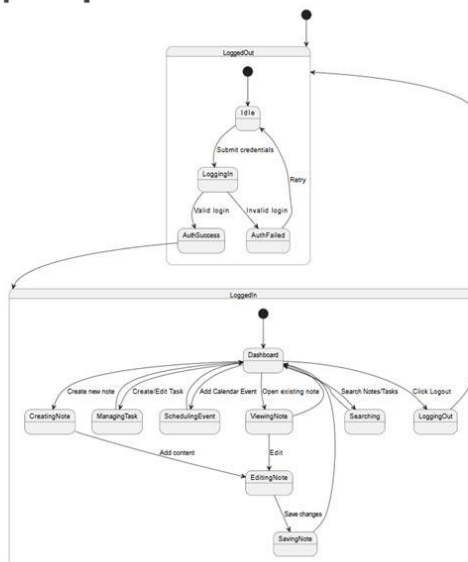


9

Рисунок Д.9 – Слайд 9

Логічне проєктування

UML-діаграма станів системи

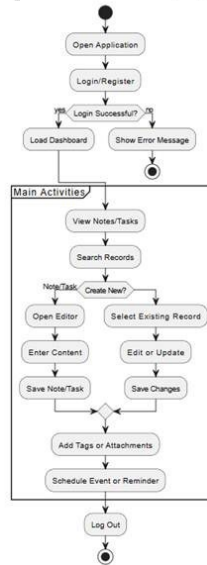


10

Рисунок Д.10 – Слайд 10

Логічне проєктування

UML-діаграма діяльності

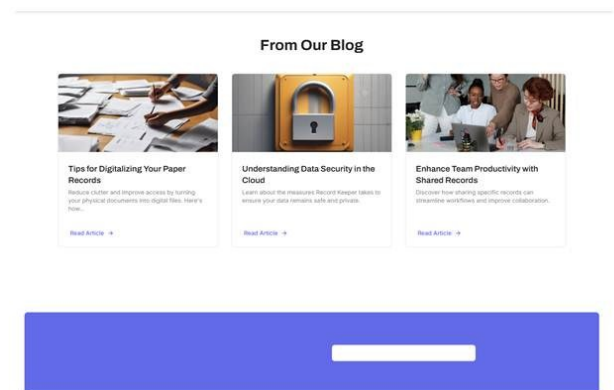
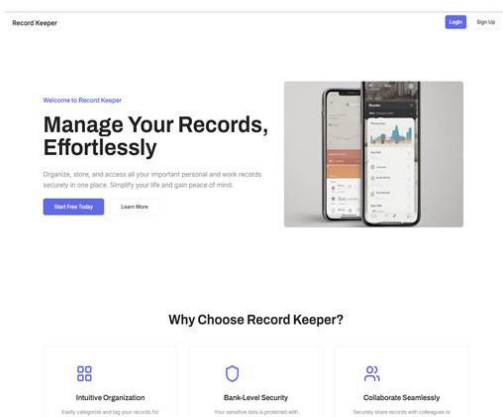


11

Рисунок Д.11 – Слайд 11

Експериментальне дослідження

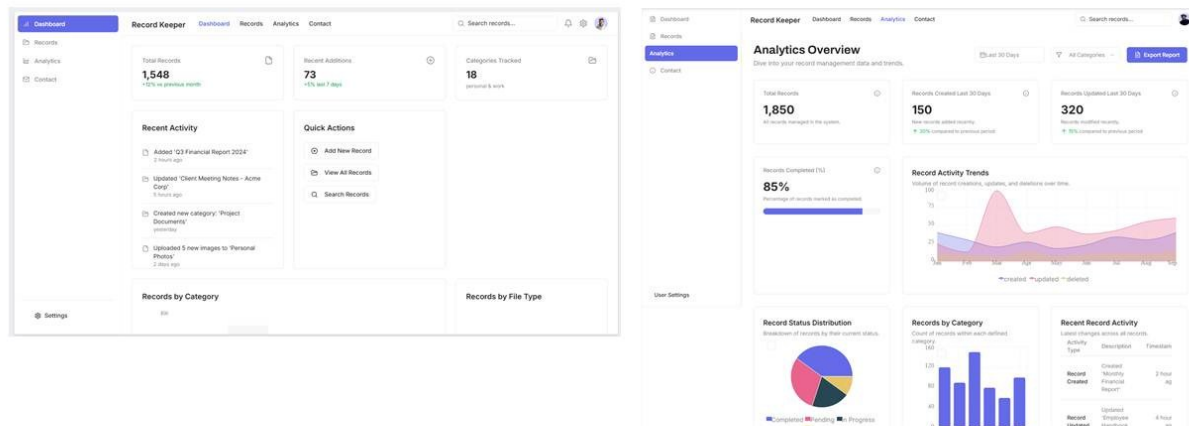
розробки



12

Рисунок Д.12 – Слайд 12

Експериментальне дослідження розробки



13

Рисунок Д.13 – Слайд 13

Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було досягнуто головну мету – виконано програмну реалізацію застосунку для ведення, впорядкування та управління особистих та робочих записів.

Для цього було виконано низку задач, серед яких:

- досліджено предметну область ведення, впорядкування та управління особистими та робочими записами;
- досліджено існуючі програмні застосунки та реалізації способів вирішення проблеми ведення, впорядкування та управління особистими та робочими записами;
- досліджено переваги та недоліки існуючих програмних рішень;
- спроектовано та розроблено нове рішення для ведення, впорядкування та управління особистими та робочими записами;
- проведено тестування;
- на основі виконаної роботи впорядковано головні висновки.

На основі розв'язаних задач було отримано закінчений програмний продукт, що в повній мірі реалізує поставлені задачі та в повній мірі відповідає всім вимогам визначеним на початку розробки у вигляді функціональних вимог.

Шляхом подальшого розвитку розробки є налаштування доступу до сторонніх застосунків (Google календарю, наприклад) та колективної роботи над записами.

14

Рисунок Д.14 – Слайд 14