

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
ЕЛЕКТРОННОГО ВЕДЕННЯ АРХІВУ ДОКУМЕНТАЦІЇ
SOFTWARE IMPLEMENTATION OF THE E-DOCUMENTATION
ARCHIVE MANAGEMENT APPLICATION

Виконав: студент 4 курсу, групи КНТ-111сп
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ЦЕЛІЩЕВ Д.О.

(ПРИЗВИЩЕ та ініціали)

Керівник ЛЕОЩЕНКО С.Д.

(ПРИЗВИЩЕ та ініціали)

Рецензент СКРУПСЬКИЙ С.Ю.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

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
« » 2024 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЦЕЛІЩЕВА Дмитра Олеговича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмна реалізація застосунку електронного ведення архіву документації. Software Implementation of the E-Documentation Archive Management Application

керівник проєкту (роботи) д-р філософ., ст.викл. Леощенко Сергій Дмитрович,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від «24» квітня 2024 року
№ 168

2. Строк подання студентом проєкту (роботи) 03 травня 2024 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. Опис програми. 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	ЛЕОЩЕНКО С.Д., ст.викл.		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання «04» квітня 2023 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)

_____ Дмитро ЦЕЛІЩЕВ
(підпис) (Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи)

_____ Сергій ЛЕОЩЕНКО
(підпис) (Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
86 с., 24 рис., 3 табл., 4 дод., 20 джерел.

ДОКУМЕНТООБІГ, АРХІВ, ПІДПРИЄМСТВО,
КРОСПЛАТФОРМНИЙ ЗАСТОСУНОК, РОЗРОБКА, МОВА
ПРОГРАМУВАННЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Об'єкт дослідження – процес програмної реалізації застосунку електронного ведення архіву документації.

Предмет дослідження – програмні засоби для реалізації застосунку електронного ведення архіву документації.

Метою роботи є програмна реалізація застосунку електронного ведення архіву документації.

Матеріали, методи та технічні засоби: мова програмування JavaScript та фреймворк AngularJS, система керування базами даних MySQL, засоби моделювання UML.

Результати. Розроблено мобільний застосунок – застосунок для електронного ведення архіву документації.

Практична цінність роботи полягає у програмній реалізації застосунку електронного ведення архіву документації у вигляді кросплатформного застосунку.

Висновки. Реалізовано застосунок електронного ведення архіву документації у вигляді кросплатформного застосунку.

Галузь використання – підприємства, що потребують налагодження документообігу.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 86 pages, 24 figures, 3 tables, 4 appendixes, 20 sources.

DOCUMENT MANAGEMENT, ARCHIVE, ENTERPRISE, CROSS-PLATFORM APPLICATION, DEVELOPMENT, PROGRAMMING LANGUAGE, SOFTWARE

The object of research is the process of software implementation of an electronic documentation archive application.

The subject of the research is software tools for implementing an electronic documentation archive application.

The aim of the work is to provide a software implementation of an application for electronic documentation archive management.

Materials, methods, and technical tools: JavaScript programming language and AngularJS framework, MySQL database management system, UML modeling tools.

Results. A mobile Physical Activity Tracker app has been developed.

The practical value of the work lies in the software implementation of the application for electronic archive management of documentation in the form of a cross-platform application.

Conclusions. An application for electronic maintenance of the documentation archive in the form of a cross-platform application has been implemented.

Scope of use – enterprises that need to establish document flow.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	7
Вступ.....	8
1 Аналіз предметної області	12
1.1 Дослідження цифрового документообігу.....	12
1.2 Архівування документів під час електронного документообігу	14
1.3 Порівняння існуючих аналогів.....	16
1.4 Постановка завдання	23
1.5 Висновки за розділом 1	27
2 Матеріали і методи	28
2.1 Вибір мови програмування	28
2.2 Вибір та обґрунтування програмних надбудов	34
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.....	52
4.1 Призначення й умови застосування програми	52
4.2 Вимоги до інтерфейсу	52
4.3 Умови виконання програми.....	53
4.4 Методика та результати тестування	53
Висновки	56
Перелік джерел посилання	58
Додаток А Технічне завдання	60
Додаток Б Опис програми	65
Додаток В Текст програми	72
Додаток Д Слайди презентації.....	80

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ОС – операційна система;

ПЗ – програмне забезпечення;

СУБД – система управління базами даних;

GUI – graphical user interface;

HTTPS – hypertext transfer protocol secure;

UI – user interface;

UML – unified modeling language.

ВСТУП

Розробка програмних засобів (ПЗ) для архівації та збереження документів важлива з кількох причин. Від так, застосунок для архівування та збереження документів допомагає користувачам упорядковувати та класифікувати свої документи. Користувачі можуть легко зберігати та отримувати документи на основі категорій, тегів або метаданих, що підвищує ефективність та продуктивність [1]-[3].

Зберігання документів централізовано в одному застосунку гарантує, що користувачі зможуть отримати доступ до своїх файлів з будь-якого місця, у будь-який час, з будь-якого пристрою, підключеного до Інтернету. Ця доступність особливо цінна для віддалених працівників, мандрівників або приватних осіб, яким потрібен доступ до документів у дорозі [1]-[3].

Більш того, програми для архівування документів, як правило, пропонують надійний захист конфіденційної інформації. Це включає шифрування, контроль доступу, автентифікацію користувачів та журнали перевірки, що гарантує, що лише авторизовані користувачі можуть отримувати доступ до документів та змінювати їх [1]-[3].

Завдяки зберіганню документів у програмі архівування користувачі отримують переваги від автоматичного резервного копіювання та механізмів аварійного відновлення. Це допомагає запобігти втраті даних у разі збоїв обладнання, випадкового видалення або інших непередбачених подій [1]-[3].

Багато програм для архівування документів підтримують функції співпраці, що дозволяють декільком користувачам одночасно працювати з документами, відстежувати зміни та залишати коментарі чи анотації. Це сприяє спільній роботі та покращує взаємодію між членами команди [1]-[3].

Для підприємств і організацій додатки для архівування документів допомагають забезпечити відповідність нормативним вимогам і галузевим стандартам. Ці програми часто включають функції управління політикою зберігання документів, контролю версій та аудиту відповідності [1]-[3].

Завдяки оцифруванню та централізації зберігання документів організації можуть зменшити витрати, пов'язані з фізичним зберіганням, друком та паперовими робочими процесами. Крім того, спрощені процеси управління документами можуть призвести до економії часу та підвищення операційної ефективності [1]-[3].

Перехід на цифрове зберігання документів зменшує потребу в папері та мінімізує вплив на навколишнє середовище, пов'язаний із друком, надсиланням та зберіганням фізичних документів [1]-[3].

Загалом, розробка програми для архівування та збереження документів має багато переваг, включаючи покращену організацію, доступність, безпеку, співпрацю, відповідність вимогам, економію коштів та екологічну стійкість. Як для особистого використання, так і для потреб підприємства, такі застосунки відіграють вирішальну роль в сучасній практиці управління документами [1]-[3].

Аналізуючи причини актуальності розробки подібних систем, варто відзначити зростаючий обсяг цифрових документів. З ростом оцифровки інформації і поширенням віддаленої роботи обсяг створюваних цифрових документів зростає в геометричній прогресії. Без належних рішень для архівування та збереження документів приватні особи та організації можуть мати труднощі з ефективним управлінням та організацією своїх постійно розширюваних бібліотек цифрових документів [1]-[3].

Перехід до віддаленої роботи та розподілених команд підкреслив важливість наявності надійних інструментів для доступу та співпраці над документами з будь-якої точки світу. Існує гостра потреба в додатках, які забезпечують безперешкодний віддалений доступ, співпрацю в режимі реального часу та безпечний обмін документами для підтримки сучасних робочих середовищ [1]-[3].

Порушення безпеки даних та відповідності становлять значний ризик для окремих осіб та організацій. Розробка захищених програм для архівування документів із надійним шифруванням, контролем доступу та функціями

відповідності має вирішальне значення для захисту конфіденційної інформації та забезпечення відповідності нормативним вимогам [1]-[3].

В умовах стихійних лих, кібератак або інших непередбачених подій організації повинні мати ефективні стратегії резервного копіювання та відновлення для захисту критично важливих документів і забезпечення безперервності бізнесу. Існує гостра потреба в додатках для архівації документів, які забезпечують надійне резервне копіювання, аварійне відновлення і реплікацію даних [1]-[3].

Не ефективні процеси управління документами можуть знизити продуктивність і призвести до втрати цінного часу та ресурсів. Оптимізуючи процеси зберігання документів, пошуку та співпраці, добре розроблені програми для архівування документів можуть допомогти окремим особам та організаціям працювати ефективніше [1]-[3].

Перехід від паперового до цифрового зберігання документів не тільки підвищує ефективність, але і має екологічні переваги за рахунок скорочення споживання паперу, енергоспоживання і викидів вуглекислого газу, пов'язаних з традиційними методами управління документообігом. Необхідні термінові дії для розробки та просування стійких рішень для архівування документів, які мінімізують вплив на навколишнє середовище [1]-[3].

У сучасній цифровій економіці, що швидко розвивається, використовують інноваційні технології управління документами, отримують конкурентну перевагу. Розробка передових програм для архівування документів, що використовують штучний інтелект, машинне навчання та інші нові технології, може сприяти інноваціям та диференціації на ринку [1]-[3].

Таким чином, нагальне завдання розробки програми для архівування та збереження документів зумовлене зростаючим обсягом цифрових документів, необхідністю віддаленого доступу та співпраці, проблемами безпеки та відповідності, готовністю до стихійних лих, підвищенням ефективності, екологічною стійкістю та прагненням до конкурентних переваг за рахунок інновацій. Вирішення цих проблем вимагає негайних дій щодо розробки

надійних, зручних у використанні та масштабованих рішень для архівування документів, які відповідають мінливим потребам окремих осіб та організацій у сучасному цифровому світі.

Отже актуальною задачею залишається програмна реалізація застосунку електронного ведення архіву документації.

Для виконання такої задачі необхідно виконати наступні завдання:

- дослідити предметну область цифрового документообігу та архівування документів;
- дослідити переваги та недоліки існуючих аналогічних рішень для архівування документів;
- обґрунтувати вибір технічних засобів реалізації нового застосунку для архівування документів;
- виконати проєктування архітектури нового застосунку;
- виконати програмну реалізацію нового застосунку;
- провести тестування розробки;
- сформулювати висновки на основі виконаної роботи.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Перший розділ пояснювальної записки присвячено ознайомленню із предметною областю цифрового документообігу, визначенням задачі архівування документів та ознайомленню із існуючими аналогічними рішеннями проблеми архівування документів. Після аналітичної частини буде наведено перелік функціональних та нефункціональних вимог до ПЗ, що буде розроблятися під час роботи.

1.1 Дослідження цифрового документообігу

Електронна документація та системи електронного документообігу працюють спільно, спрощуючи створення, зберігання, організацію, пошук і поширення цифрових документів в організації [4]-[6].

Системи електронного документообігу дозволяють користувачам створювати цифрові документи за допомогою текстових редакторів, електронних таблиць, презентацій або інших програмних засобів. Користувачі можуть вводити текст, зображення, таблиці та інший вміст для створення документів професійного вигляду [4]-[6].

Після створення документів їх потрібно записати та оцифрувати для зберігання в системах документообігу. Це може включати сканування фізичних документів у цифровому форматі або імпорт електронних файлів з різних джерел [4]-[6].

Система забезпечує централізоване сховище для зберігання цифрових документів. Документи зазвичай зберігаються в структурованій ієрархії папок або категорій, що полегшує користувачам організацію та пошук файлів [4]-[6].

Кожен документ в системі документообігу пов'язаний з метаданими, які включають в себе таку інформацію, як назва документа, автор, дата створення, тип файлу, ключові слова і права доступу. Управління метаданими забезпечує точний опис документів і можливість їх простого пошуку [4]-[6].

Системи документообігу зазвичай надають можливості контролю версій для управління змінами в документах з часом. Це дозволяє користувачам відстежувати зміни, порівнювати різні версії та повертатися до попередніх версій, якщо це необхідно [4]-[6].

Системи документообігу індексують документи на основі їх метаданих та вмісту, забезпечуючи швидкий та точний пошук. Користувачі можуть шукати документи за ключовими словами, автором, датою чи іншими критеріями, і система швидко отримує відповідні результати [4]-[6].

Системи реалізують механізми контролю доступу для обмеження доступу до конфіденційних документів та забезпечення того, щоб лише авторизовані користувачі могли переглядати, редагувати або видаляти файли. Функції безпеки, такі як шифрування, аутентифікація і контрольні журнали, допомагають захистити від несанкціонованого доступу і витоків даних [4]-[6].

Системи документообігу підтримують документообіг і функції спільної роботи, дозволяючи декільком користувачам працювати над документами одночасно. Автоматизація документообігу спрощує процеси перевірки та затвердження документів, а інструменти спільної роботи полегшують редагування, коментування та відстеження версій у режимі реального часу [4]-[6].

Системи електронного документообігу часто інтегруються з іншими програмними програмами та системами, такими як електронна пошта, месенджери та інструменти управління проєктами. Інтеграція забезпечує безперешкодний обмін документами і даними між системами, підвищуючи ефективність і продуктивність [4]-[6].

Системи документообігу допомагають організаціям дотримуватися нормативних вимог та галузевих стандартів щодо зберігання документів та управління записами. Вони надають функції для визначення політики зберігання, архівування застарілих документів та забезпечення відповідності нормативним вимогам [4]-[6].

В цілому, системи електронного документообігу працюють спільно для оцифрування, управління та контролю життєвого циклу цифрових документів в організації, підвищуючи ефективність, продуктивність, взаємодію та відповідність вимогам при одночасному зниженні витрат і впливу на навколишнє середовище, пов'язаних з традиційним паперовим документообігом [4]-[6].

1.2 Архівування документів під час електронного документообігу

Система архівування електронної документації працює шляхом систематичного зберігання, систематизації та управління цифровими документами з часом [7], [8].

Система архівації захоплює цифрові документи з різних джерел, включаючи інструменти для створення документів, системи електронної пошти, скануючі пристрої, файлові сервери та інші програми. Документи можуть бути в різних форматах, таких як текстові документи, електронні таблиці, презентації, зображення, PDF-файли та електронні листи.

Кожен документ в системі архівування індексується і асоціюється з метаданими, які включають в себе таку інформацію, як назва документа, автор, дата створення, тип файлу, ключові слова і права доступу. Індксація та управління метаданими дозволяють користувачам ефективно шукати та витягувати документи [7], [8].

Система архівації зберігає цифрові документи в захищеній та масштабованій інфраструктурі зберігання даних, такі як локальні сервери, хмарні сховища або гібридні рішення для зберігання даних. Функції управління сховищем забезпечують надійне зберігання документів з використанням відповідних механізмів резервного копіювання та аварійного відновлення [7], [8].

Система архівування реалізує політику зберігання для управління життєвим циклом цифрових документів відповідно до нормативних вимог,

потреб бізнесу та найкращих практик. Політика зберігання визначає, як довго слід зберігати документи, коли їх слід архівувати та коли їх слід видаляти чи переробляти [7], [8].

Механізми контролю доступу обмежують доступ до архівних документів і гарантують, що тільки авторизовані користувачі можуть переглядати, редагувати або видаляти файли. Такі функції безпеки, як шифрування, аутентифікація і контрольні журнали, допомагають захистити архівні документи від несанкціонованого доступу і витоку даних [7], [8].

Система архівування надає потужні можливості пошуку і витяг інформації, дозволяючи користувачам швидко знаходити архівні документи і отримувати до них доступ на основі таких критеріїв, як ключові слова, метадані, автор, дата і зміст. Розширені функції пошуку дозволяють користувачам виконувати складні запити та фільтрувати результати, щоб знайти відповідні документи [7], [8].

Функції контролю версій відстежують зміни в архівних документах з плином часу, дозволяючи користувачам переглядати і порівнювати різні версії, повертатися до попередніх версій і відстежувати історію змін документів. Система контролю версій гарантує, що користувачі матимуть доступ до найактуальнішої та найточнішої інформації [7], [8].

Система архівування генерує журнали аудиту та звіти про відповідність вимогам для відстеження дій користувачів, доступу до документів і змін в архівних документах. Аудит і звітність про відповідність допомагають організаціям продемонструвати відповідність нормативним вимогам і внутрішнім політикам [7], [8].

Система архівування може інтегруватися з іншими програмними додатками та системами, такими як системи управління документообігом, системи управління контентом, системи планування ресурсів підприємства (ERP), системи управління взаємовідносинами з клієнтами (CRM) та системи електронної медичної документації (EHR). Інтеграція забезпечує безперешкодний обмін документами і даними між системами, підвищуючи ефективність робочого процесу і узгодженість даних [7], [8].

Система архівування забезпечує довгострокове збереження цифрових документів, використовуючи такі стратегії збереження, як перенесення даних, перетворення формату і стандарти цифрового збереження. Довгострокове зберігання гарантує, що архівні документи залишаються доступними та читабельними з часом, навіть у міру розвитку технологій та форматів файлів.

Таким чином, система архівації електронної документації забезпечує збір, індексацію, зберігання, управління і збереження цифрових документів безпечним і організованим чином, забезпечуючи при цьому відповідність нормативним вимогам, підтримуючи ефективний пошук документів і захищаючи від несанкціонованого доступу і втрати даних [7], [8].

1.3 Порівняння існуючих аналогів

На сьогоднішній день існує велика кількість застосунків, які допомагають організувати електронний документообіг. Частина з них може бути адаптована під конкретні корпоративні задачі. Проте, значна частина таких рішень не надає сервісів з архівування процесів та документообігу та самих документів.

OpenText Content Suite Platform, яка є базою асортименту OpenText Enterprise Information Management, – це всеосяжна система управління корпоративними інформаційними ресурсами компанії, яка розроблена для управління потоком інформації з першого контакту до архівування або видалення [9], [10].

Content Suite Platform забезпечує оперативне управління інформацією, що дозволяє відповідати все більш складним і стрімко мінливим вимогам нормативних актів і зростаючому обсягу бізнес-інформації. Content Suite Platform допомагає контролювати ризики, одночасно дозволяючи організації концентруватися на використанні інформації з метою стратегічного та інноваційного розвитку [9], [10] (рис. 1.1).

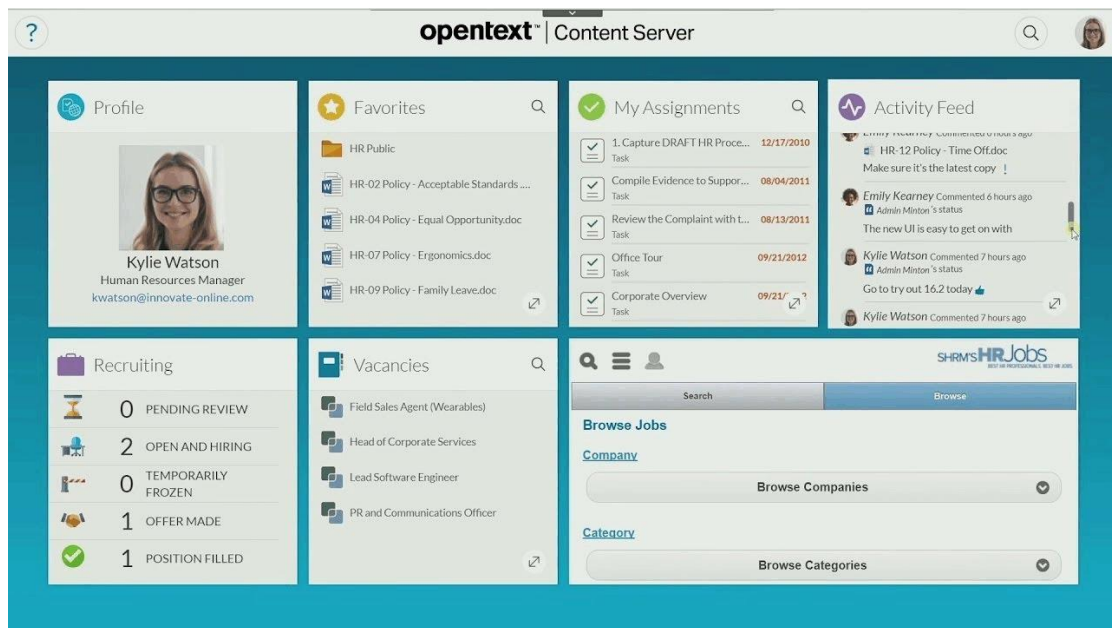


Рисунок 1.1 – Графічні інтерфейси застосунку OpenText Content Suite Platform [9]

Alfresco Content Services пропонує повнофункціональну систему управління корпоративним контентом для організацій, яким потрібні масштабованість корпоративного рівня, продуктивність і цілодобова підтримка критично важливого для бізнесу контенту і відповідність вимогам. Він надає широкий спектр можливостей для використання, таких як контент і сервіси управління, Контекстний пошук і аналітика, можливість простої інтеграції з іншими додатками. В основі контент-сервісів лежить репозиторій, Підтримуваний сервером, який зберігає контент, метадані, асоціації та повнотекстові індекси [11], [12].

Деякі з ключових можливостей контент-сервісу включають:

- управління документообігом;
- сканування та запис документів;
- спільна робота на підприємстві;
- управління інформацією;
- інтелект;
- аналітика.

Управління документами. Платформа Alfresco дозволяє швидко знаходити потрібну інформацію і отримувати до неї доступ з будь-якого місця на будь-якому пристрої за допомогою вебінструментів перегляду [11], [12].

Інтелектуальне управління процесами. Тісно інтегровані технологічні та контент-сервіси оптимізують процеси, орієнтовані на контент, і забезпечують більш швидке і обгрунтоване прийняття рішень.

Корпоративна спільна робота. Команди можуть легко скористатися перевагами Microsoft 365 та Google Docs, підвищуючи продуктивність завдяки інструментам корпоративної співпраці.

Управління інформацією. Служби уряду Alfresco автоматизують життєвий цикл інформації з мінімальним втручанням користувачів, зменшуючи бізнес-ризик та підвищуючи відповідність [11], [12].

Аналітика. Платформа Alfresco забезпечує масштабований спосіб автоматичного збагачення вмісту та метаданих та надання ключових відомостей шляхом автоматичного вилучення цінної інформації.

Розширюйте та інтегруйте. Платформа Alfresco, заснована на відкритих стандартах та API, легко інтегрується та налаштовується відповідно до потреб вашого бізнесу [11], [12]. (рис. 1.2).

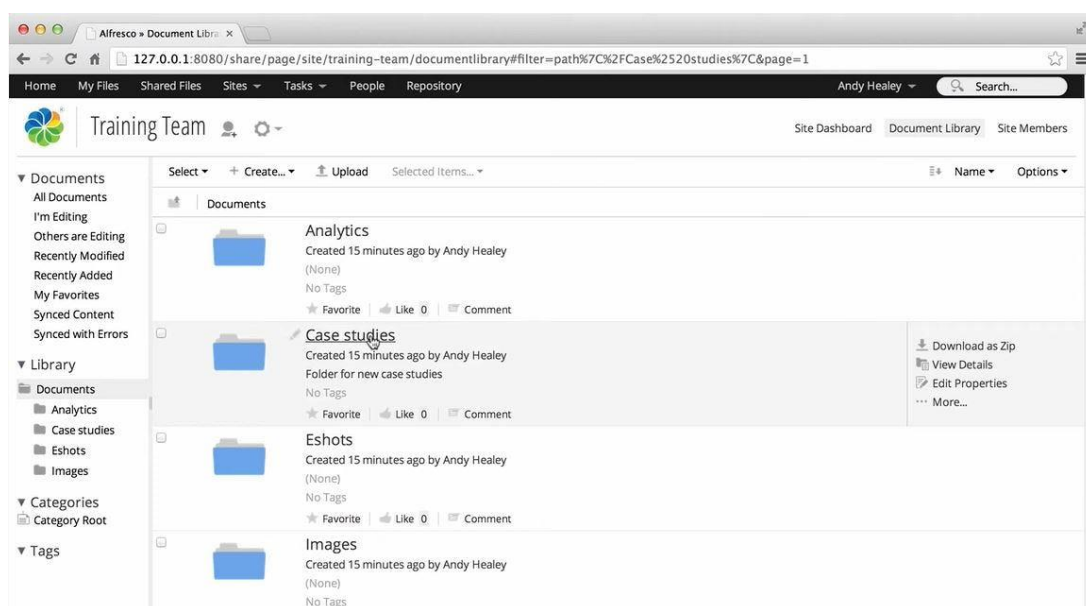


Рисунок 1.2 – Графічні інтерфейси застосунку Alfresco Content Services [13]

Google Drive можна адаптувати для управління електронною документацією, використовуючи його функції і можливості для ефективного зберігання, організації, управління цифровими документами і спільної роботи над ними. Google Drive надає хмарне рішення для безпечного зберігання цифрових документів. Користувачі можуть завантажувати документи різних форматів, включаючи текстові документи, електронні таблиці, презентації, PDF-файли, зображення тощо, у свій обліковий запис Google Drive [13], [14].

Google Drive полегшує обмін документами та співпрацю, дозволяючи користувачам обмінюватися документами з іншими користувачами як окремо, так і як частина команди чи проєкту. Користувачі можуть встановлювати дозволи для управління тим, хто може переглядати, редагувати або коментувати документи, що сприяє співпраці та забезпечує безпеку даних [13], [14].

Google Drive автоматично відстежує версії документів і правки, дозволяючи користувачам переглядати і при необхідності повертатися до попередніх версій. Історія версій забезпечує прозорість і підзвітність, забезпечуючи користувачам доступ до найактуальнішої і точної інформації.

Google Drive надає офлайн-доступ до документів, дозволяючи користувачам переглядати та редагувати файли навіть без підключення до Інтернету. Автономний доступ підвищує продуктивність і дозволяє користувачам працювати з документами в будь-який час, в будь-якому місці і на будь-якому пристрої [13], [14].

Google Drive використовує надійні заходи безпеки для захисту документів, включаючи шифрування, двофакторну аутентифікацію, засоби контролю доступу та контрольні журнали. Організації також можуть впроваджувати політику запобігання втраті даних та функції забезпечення відповідності нормативним актам та захисту даних [13], [14].

Використовуючи ці функції та можливості, організації можуть адаптувати Google Drive для управління електронною документацією, забезпечуючи ефективне зберігання документів, організацію, спільне

використання, співпрацю, контроль версій, пошук і витяг інформації, автономний доступ, безпеку, відповідність вимогам і інтеграцію з іншими інструментами і системами. (рис. 1.3).

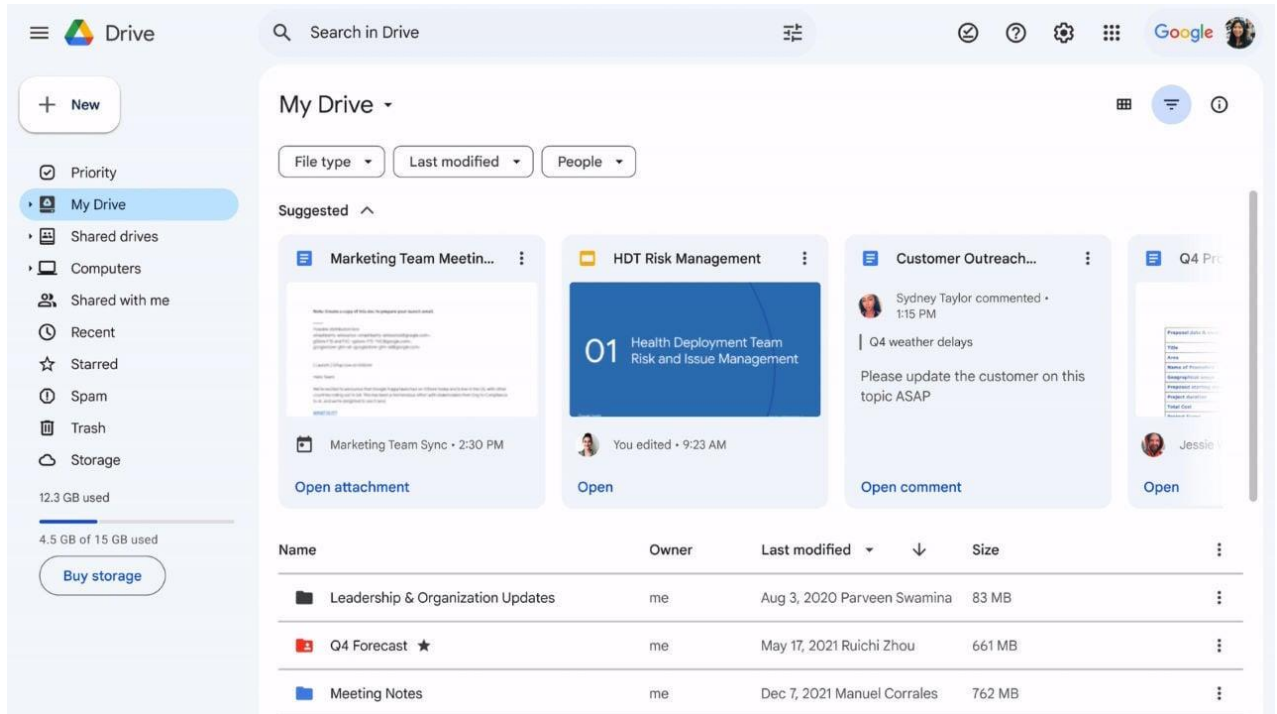


Рисунок 1.3 – Графічні інтерфейси застосунку Google Drive [13]

Ці три програми є одними з найбільш широко використовуваних і високо оцінених застосунків для відстеження фізичної активності, пропонуючи широкий спектр функцій для підтримки користувачів у моніторингу, аналізі та оптимізації рівня їх фізичної активності.

Виконаємо порівняння розглянутих аналогів за критеріями та наведемо його у вигляді табл. 1.1. Проте, на початку розглянемо критерії порівняння:

- зберігання документів – це стосується здатності системи надійно зберігати цифрові документи. Кожна система забезпечує централізоване сховище для зберігання документів;
- організація папок – усі три системи пропонують ієрархічну структуру папок для організації документів за категоріями або папками, що дозволяє користувачам ефективно керувати своїми документами;

- спільний доступ до документів – ця функція дозволяє користувачам обмінюватися документами з іншими користувачами як в організації, так і за її межами, одночасно контролюючи права доступу, такі як перегляд, редагування або коментування;

- спільна робота – функції спільної роботи дозволяють декільком користувачам працювати з документами одночасно, завдяки функціям редагування, коментування та відстеження змін у режимі реального часу;

- контроль версій – контроль версій дозволяє системі відстежувати зміни, що вносяться в документи з часом, вести історію змін документів і дозволяти користувачам при необхідності повертатися до попередніх версій;

- пошук інформації – розширені можливості пошуку допомагають користувачам швидко знаходити та витягувати документи на основі ключових слів, метаданих чи інших критеріїв, забезпечуючи ефективне управління документами;

- безпека і відповідність вимогам – надійні заходи безпеки захищають документи від несанкціонованого доступу і забезпечують відповідність нормативним вимогам, галузевим стандартам і політикам організації;

- інтеграція з іншими інструментами – можливості інтеграції забезпечують безперебійний зв'язок і обмін даними між системою управління документами та іншими бізнес-додатками і системами, підвищуючи ефективність робочого процесу і продуктивність;

- автономний доступ – автономний доступ дозволяє користувачам отримувати доступ до документів і працювати з ними навіть без підключення до Інтернету, забезпечуючи продуктивність у віддалених або відключених середовищах;

- масштабованість – масштабованість відноситься до здатності системи обробляти великі обсяги документів та користувачів, адаптуючись до мінливих потреб організації та її зростання з часом;

– налаштування та розширюваність – функції налаштування та розширюваності дозволяють організаціям адаптувати систему до своїх конкретних вимог та інтегрувати зі сторонніми програмами або засобами розробки;

– вартість – вартість включає початкові інвестиції, поточне обслуговування та абонентську плату, пов'язану з кожною системою, яка може змінюватися залежно від функцій, функціональності та обраної моделі розгортання.

Таблиця 1.1 – Аналіз програмного забезпечення для ведення архіву документації

Критерії порівняння	OpenText Content Suite	Alfresco Content Services	Google Drive	CorpDoc
Зберігання документів	Централізоване сховище для зберігання цифрових документів	Централізоване сховище для зберігання цифрових документів	Хмарне рішення для зберігання цифрових документів	Хмарне рішення для зберігання цифрових документів
Організація папок	Ієрархічна структура папок для організації документів	Ієрархічна структура папок для організації документів	Папки та підпапки для організації документів	Ієрархічна структура папок для організації документів
Загальний доступ до документів	Загальний доступ до документів з контролем дозволів	Загальний доступ до документів з контролем дозволів	Загальний доступ до документів з контролем дозволів	Загальний доступ до документів з контролем дозволів
Співпраця	Функції співпраці для редагування та коментування в режимі реального часу	Функції співпраці для редагування та коментування в режимі реального часу	Функції співпраці для редагування та коментування в режимі реального часу	Функції співпраці для редагування та коментування в режимі реального часу
Управління версіями	Контроль версій з відстеженням змін документів	Контроль версій з відстеженням змін документів	Історія версій з відстеженням змін документів	Контроль версій з відстеженням змін документів

Продовження таблиці 1.1

Критерії порівняння	OpenText Content Suite	Alfresco Content Services	Google Drive	CorpDoc
Пошук і витяг інформації	Розширені можливості пошуку документів	Розширені можливості пошуку документів	Потужні можливості пошуку документів	Розширені можливості пошуку документів
Безпека та відповідність вимогам	Надійні заходи безпеки та функції відповідності вимогам	Функції безпеки та відповідності нормативним вимогам	Суворі заходи безпеки та функції відповідності вимогам	Суворі заходи безпеки та функції відповідності вимогам
Інтеграція з іншими інструментами	Інтеграція зі сторонніми додатками і системами	Інтеграція зі сторонніми додатками і системами	Інтеграція з додатками та інструментами Google Workspace	Інтеграція зі сторонніми додатками і системами
Автономний доступ	Обмежені можливості автономного доступу	Обмежені можливості автономного доступу	Автономний доступ до документів	Автономний доступ до документів

Як видно з порівняння, більшість аналогічних застосунків мають певний набір обмежень функціонального застосування. Найчастіше, ці обмеження стосуються зберігання та організації доступу до документів, а також тарифів на зберігання цих документів.

Саме тому, актуальною задачею залишається програмна реалізація застосунку електронного ведення архіву документації.

1.4 Постановка завдання

Нагальне завдання розробки програми для архівування та збереження документів, зумовлене зростаючим обсягом цифрових документів, необхідністю віддаленого доступу та співпраці, проблемами безпеки та відповідності, готовністю до стихійних лих, підвищенням ефективності, екологічною стійкістю та прагненням до конкурентних переваг за рахунок інновацій. Вирішення цих проблем вимагає негайних дій щодо розробки

надійних, зручних у використанні та масштабованих рішень для архівування документів, які відповідають мінливим потребам окремих осіб та організацій у сучасному цифровому світі.

Отже актуальною задачею залишається програмна реалізація застосунку електронного ведення архіву документації.

1.4.1 Функціональність системи

Система має реалізовувати наступні функціональні вимоги [17], [18]:

- а) аутентифікація та авторизація користувача:
 - безпечний вхід користувача за допомогою логіну та паролю;
 - управління доступом на основі ролей для управління правами користувача;
- б) збір і завантаження документів:
 - можливість завантажувати цифрові документи з різних джерел;
 - підтримка декількох форматів файлів, включаючи текстові документи, електронні таблиці, презентації, зображення, PDF-файли і т. д;
- в) організація та зберігання документів:
 - ієрархічна структура папок для організації документів;
 - управління метаданими для зв'язування атрибутів з документами;
 - управління версіями для відстеження змін і доопрацювань;
- г) пошук і витяг документів:
 - розширені можливості пошуку за допомогою пошуку за ключовими словами, пошуку за метаданими та параметрів фільтрації;
 - швидкий доступ до нещодавно отриманих або змінених документів;
- д) перегляд і попередній документів:
 - попередній перегляд мініатюр документів на основі зображень;
- е) управління життєвим циклом документа:

- політики зберігання документів для управління життєвим циклом документа;
 - можливості архівації та очищення для видалення застарілих документів;
 - контрольні журнали для відстеження доступу до документів, змін і видалень;
- ж) звітність та аналітика:
- заздалегідь визначені звіти для використання документів, активності та журналів доступу;
 - настроювані інформаційні панелі для моніторингу показників і продуктивності документів;
- з) резервне копіювання та аварійне відновлення даних:
- регулярне резервне копіювання даних документів для запобігання втрати даних;
- и) навчання та підтримка користувачів:
- детальна документація та навчальні матеріали для користувачів;
- к) налаштування та масштабованість:
- настроювані робочі процеси, шаблони та документація;
 - масштабована архітектура, що дозволяє адаптуватися до зростання обсягу документів і бази користувачів.

1.4.2 Вимоги до інтерфейсу

Інтерфейс системи повинен відрізнятися, логічністю, інтуїтивною зрозумілістю та ергономічністю для користувачів. Дизайн має бути адаптивним для доступу з мобільних пристроїв [17], [18].

1.4.3 Вимоги до безпеки

Забезпечення конфіденційності та безпеки даних:

- регулярні аудити безпеки та оцінка вразливостей;
- дотримання відповідних правил та рекомендацій щодо захисту даних для захисту інформації користувача.

1.4.4 Не функціональні вимоги до продукту

Апаратні вимоги до сервера:

- центральний процесор – від 16 фізичних ядер та 32 потоків з тактовою частотою від 4 GHz;
- оперативна пам'ять формату DDR4, з тактовою частотою від 2600 MHz та об'ємом від 128 Гб;
- накопичувач (формату HDD або SSD) з вільним простором від 2 Гб.

Програмні вимоги до сервера:

- розгорнута серверна операційна система (ОС), наприклад: Windows Server, Linux Server або macOS Server;
- розгорнута SQL-сумісна СКБД.

Апаратні вимоги до клієнта:

- центральний процесор – від 2 фізичних ядер та 2 потоків з тактовою частотою від 2 GHz;
- оперативна пам'ять об'ємом від 4 Гб;
- накопичувач (формату HDD або SSD) з вільним простором від 1 Гб.

Програмні вимоги до сервера:

- наявність веббраузера актуальної версії.

1.5 Висновки за розділом 1

В першому розділі пояснювальної записки було виконано ознайомлення із предметною областю цифрового документообігу, визначення задачі архівування документів та ознайомлення із існуючими аналогічними рішеннями проблеми архівування документів. Після аналітичної частини було наведено перелік функціональних та нефункціональних вимог до ПЗ, що буде розроблятися під час роботи.

2 МАТЕРІАЛИ І МЕТОДИ

Другий розділ пояснювальної записки буде присвячено вибору та обґрунтуванню вибору технічних засобів розробки: технологія, мова програмування, фреймворки, патерн розробки, тощо.

2.1 Вибір мови програмування

JavaScript – це універсальна мова програмування, яка стала наріжним каменем веброзробки. Він має численні переваги, які роблять його кращим вибором для створення вебзастосунки [15], [16].

JavaScript – це фактична мова Інтернету, яка підтримується всіма сучасними веббраузерами без необхідності додаткових плагінів. Ця повсюдність робить його ідеальним вибором для веброзробки, оскільки розробники можуть писати код, який легко запускається на різних платформах і пристроях [15], [16].

JavaScript не обмежується сценаріями на стороні клієнта; він також може використовуватися для розробки на стороні сервера на таких платформах, як Node.js. Це дозволяє розробникам створювати додатки на основі повного стека повністю на JavaScript, використовуючи єдину мову та екосистему як для інтерфейсної, так і для серверної розробки [15], [16].

JavaScript має велику екосистему бібліотек і фреймворків, які спрощують розробку і пропонують готові рішення для загальних завдань. Фреймворки, такі як React, Angular та Vue.js, надають ефективні інструменти для створення інтерактивних користувацьких інтерфейсів, в той час як бібліотеки, подібні Express.js, полегшують розробку на стороні сервера [15], [16].

Асинхронна модель програмування JavaScript дозволяє виконувати неблокуючі операції вводу-виводу, що робить його добре придатним для виконання паралельних завдань, таких як Витяг даних з API, обробка введів

користувача та оновлення інтерфейсу користувача без зависання браузера. Асинхронне виконання коду підвищує швидкість відгуку та продуктивність вебзастосунків [15], [16].

З появою прогресивних вебзастосунків та кросплатформних надбудов розробки, таких як React Native та Ionic, JavaScript дозволяє розробникам створювати програми, що працюють на декількох платформах, включаючи веббраузери, мобільні пристрої та настільні комп'ютери. Така кросплатформна Сумісність максимально розширює охоплення і скорочує витрати на розробку [15], [16].

У JavaScript існує велика та активна спільнота розробників, яка сприяє розвитку мови та надає підтримку, документацію, навчальні посібники та бібліотеки з відкритим кодом. Розробники можуть використовувати ресурси спільноти для вивчення, усунення несправностей та отримання інформації про найкращі практики та нові тенденції у веброботі [15], [16].

Динамічний і вільно набраний характер JavaScript полегшує швидке створення прототипів і ітерацію, дозволяючи розробникам швидко створювати, тестувати і виконувати ітерації в вебдодатках. Такі інструменти, як гаряче перезавантаження модулів, ще більше прискорюють процес розробки, надаючи миттєвий відгук про зміни в коді без перезавантаження всієї програми [15], [16].

Незважаючи на те, що JavaScript є інтерпретованою мовою, він може забезпечити високу продуктивність при правильній оптимізації. Такі методи, як розділення коду, затримка завантаження та рендеринг на стороні сервера, підвищують масштабованість та продуктивність програм JavaScript, дозволяючи їм справлятися зі зростаючим навантаженням користувачів та складними робочими процесами [15], [16].

JavaScript йде в ногу з розвитком вебстандартів і функцій, дозволяючи розробникам використовувати новітні API HTML, CSS і JavaScript для створення сучасних вебзастосунків з розширеними функціями, такими як геолокація, WebRTC, WebSockets і автономне кешування [15], [16].

JavaScript забезпечує плавну інтеграцію зі сторонніми сервісами та API, дозволяючи розробникам легко впроваджувати у свої вебдодатки такі функції, як аутентифікація в соціальних мережах, платіжні шлюзи, картографічні сервіси та аналітичні інструменти [15], [16].

На закінчення, універсальність JavaScript, екосистема, модель асинхронного програмування, сумісність між платформами, підтримка спільноти, швидкі ітерації розробки, масштабованість, продуктивність, відповідність сучасним вебстандартам та можливості інтеграції роблять його чудовим вибором для розробки вебзастосунків. Оскільки Веб продовжує розвиватися, JavaScript залишається на передньому краї, дозволяючи розробникам створювати інноваційні та інтерактивні вебдодатки для користувачів по всьому світу [15], [16].

Проведемо порівняння мов програмування для реалізації застосунку електронного ведення архіву документації, результати порівняння представимо в табличній формі (табл. 2.1). У якості критеріїв будемо використовувати наступні:

- універсальність – ця функція має вирішальне значення, оскільки вибрана мова програмування повинна мати можливість обробляти розробку як на стороні клієнта, так і на стороні сервера, оскільки застосунку потрібна функціональність на обох фронтах;
- розробка на основі повного стека – можливості розробки на основі повного стека необхідні для забезпечення того, щоб вибрана мова могла обробляти всі аспекти створення програми, включаючи інтерфейси зовнішнього користувача та логіку внутрішнього сервера;
- фреймворки та бібліотеки – наявність фреймворків та бібліотек може суттєво вплинути на швидкість та ефективність розробки. Багата екосистема фреймворків і бібліотек може надати готові рішення для загальних завдань і спростити розробку;

- асинхронне програмування –необхідне для обробки паралельних завдань та неблокуючих операцій вводу-виводу, які є загальними вимогами у вебзастосунках для забезпечення швидкодії та масштабованості;
- сумісність між платформами –дозволяє застосунку працювати на різних пристроях і платформах, максимізуючи його охоплення та доступність для користувачів;
- підтримка спільноти – активна спільнота розробників надає цінні ресурси, підтримку та внески, які можуть допомогти у навчанні, усуненні несправностей та прискоренні розвитку;
- швидкість розробки – має вирішальне значення для своєчасної доставки програми та ефективної ітерації функцій;
- продуктивність – міркування, пов'язані з продуктивністю, важливі для того, щоб програма могла впоратися з очікуваними навантаженнями користувачів та забезпечити чуйний досвід користувачів;
- можливості інтеграції –дозволяють застосунку легко взаємодіяти із зовнішніми службами та API, розширюючи його функціональність та функціональність;
- рівень освоєння обраної мови впливає на простоту впровадження і рівень володіння ним розробниками, що може вплинути на терміни розробки і розподіл ресурсів;
- безпека – має першорядне значення для захисту програми та її даних від потенційних загроз та вразливостей, забезпечуючи конфіденційність, цілісність та доступність інформації.

Враховуючи ці особливості, ми можемо оцінити придатність різних мов програмування для реалізації програми для управління архівами електронної документації та приймати обґрунтовані рішення на основі вимог та цілей проєкту.

Таблиця 2.1 – Порівняння мов програмування

Критерій	JavaScript	Python	PHP
Універсальність	Добре підходить як для розробки на стороні клієнта, так і на стороні сервера за допомогою таких фреймворків, як Node.js	Широко використовується для розробки на стороні сервера з використанням таких фреймворків, як Django та Flask	В основному використовується для сценаріїв на стороні сервера, рідше для розробки на стороні клієнта
Розробка на основі повного стека	Підтримує розробку з повним стеком	Підтримує розробку на основі повного стека з використанням Django, Flask та інших фреймворків	В основному використовується для розробки на стороні сервера, можливості розробки на стороні клієнта обмежені
Фреймворки та бібліотеки	за допомогою Node.JS величезної екосистеми з такими фреймворками, як React, Angular та Vue.js для розробки на стороні інтерфейсу та Express.JS для розробки на стороні сервера	Багата екосистема з використанням таких фреймворків, як Django, Flask та Pyramid для веброзробки	Обмежені можливості фреймворку в порівнянні з JavaScript і Python, такими фреймворками, як Laravel і Symfony
Асинхронне програмування	Підтримує асинхронне програмування для неблокуючих операцій введення-виведення, підвищуючи продуктивність і оперативність реагування	Підтримує асинхронне програмування з використанням таких бібліотек, як asyncio, що дозволяє виконувати паралельні операції	Підтримує асинхронне програмування з використанням таких бібліотек, як ReactPHP, що дозволяє виконувати неблокуючі операції вводу-виводу

Продовження таблиці 2.1

Критерій	JavaScript	Python	PHP
Кросплатформна Сумісність	Широко використовується для веброзробки, може бути адаптований для мобільних пристроїв і настільних комп'ютерів за допомогою таких платформ, як React Native і Electron	Широко використовується для веброзробки, а також для мобільної розробки з використанням таких фреймворків, як Django REST framework і Flask-RESTful	В основному використовується для веброзробки, рідше для мобільних пристроїв і настільних комп'ютерів
Підтримка спільноти	Велика та активна спільнота розробників з великою кількістю ресурсів, посібників та проєктів з відкритим кодом	Активна спільнота розробників з великою документацією, посібниками та бібліотеками з відкритим кодом	Активна спільнота розробників з ресурсами, навчальними посібниками та проєктами з відкритим кодом
Швидкість розробки	Полегшує швидке створення прототипів і ітерацію за допомогою таких інструментів ,як" гаряча " перезавантаження модулів (HMR)	Забезпечує швидку розробку завдяки лаконічному синтаксису і великим стандартним бібліотекам	Забезпечує швидку розробку з використанням широкого спектру вбудованих функцій і бібліотек
Уявлення	Забезпечує високу продуктивність завдяки таким методам оптимізації, як розділення коду та рендеринг на стороні сервера	Відомий своєю продуктивністю та масштабованістю, особливо в галузі обробки даних та наукових обчислень	Загалом добре працює з вебдодатками, але може вимагати оптимізації для масштабних розгортань

Продовження таблиці 2.1

Критерій	JavaScript	Python	PHP
Можливості інтеграції	Повна інтеграція зі сторонніми сервісами і API, що полегшує створення багатофункціональних додатків	Добре інтегрується зі сторонніми сервісами і бібліотеками, дозволяючи створювати багатофункціональні додатки	Підтримує інтеграцію з різними службами та бібліотеками, але може вимагати додаткових налаштувань
Процес навчання	Через асинхронне програмування та складні рамки процес навчання може бути складним	Вважається відносно простим у вивченні та зручним для початківців завдяки читабельному синтаксису та зрозумілій документації	Як правило, їх легко засвоїти, особливо розробники, які мають досвід роботи з іншими мовами в стилі Сі
Безпека	Вимагає пильної уваги до методів забезпечення безпеки, особливо при взаємодії на стороні клієнта і сервера	Пропонує надійні функції та методи безпеки, але вразливості можуть виникнути через неправильну реалізацію	Методи безпеки необхідні для запобігання таких поширених вразливостей, як введення SQL та міжсайтовий сценарій (XSS).

Враховуючи ці особливості, ми можемо оцінити придатність різних мов програмування для реалізації програми для управління архівами електронної документації та приймати обґрунтовані рішення на основі вимог та цілей проєкту. Враховуючи переваги, гнучкість та активну спільноту вибір було зроблено на користь саме Java.

2.2 Вибір та обґрунтування програмних надбудов

AngularJS, фреймворк JavaScript, який підтримує Google, пропонує багато переваг для розробки вебзастосунків [17], [18].

AngularJS використовує архітектурний шаблон Model-View-Controller (MVC), який сприяє чіткому розділенню завдань. Ця структура покращує організацію коду, зручність супроводу і масштабованість, спрощуючи управління складними вебзастосунками.

Двостороння прив'язка даних: AngularJS реалізує двосторонню прив'язку даних, при якій зміни в моделі автоматично оновлюють подання, і навпаки. Цей двонаправлений потік даних зменшує потребу в ручних маніпуляціях з DOM, що призводить до більш чистого коду та підвищення продуктивності розробників [17], [18].

AngularJS забезпечує вбудований механізм введення залежностей, що полегшує створення модульного та тестованого коду. Введення залежностей дозволяє вільно зв'язувати компоненти та легко замінювати їх, забезпечуючи можливість повторного використання коду та спрощуючи модульне тестування [17], [18].

Директиви – це потужна функція AngularJS, яка дозволяє розробникам розширювати HTML за допомогою користувацьких елементів, атрибутів та поведінки. Директиви дозволяють створювати повторно використовувані компоненти інтерфейсу користувача та полегшують реалізацію складних взаємодій інтерфейсу користувача [17], [18].

AngularJS підтримує модульність, використовуючи модулі, які інкапсулюють пов'язані компоненти, служби та директиви. Модульний дизайн сприяє організації коду, можливості повторного використання і супроводу, дозволяючи розробникам більш ефективно управляти великомасштабними додатками [17], [18].

AngularJS використовує шаблони HTML для визначення інтерфейсу користувача, який забезпечує знайомий розробникам декларативний синтаксис. Шаблони підтримують динамічну прив'язку даних, обчислення виразів і умовний рендеринг, дозволяючи створювати динамічні та інтерактивні уявлення [17], [18].

AngularJS розроблений з урахуванням можливостей тестування і пропонує підтримку модульного тестування, наскрізного тестування та інтеграційного тестування з коробки. Фреймворк включає такі інструменти, як Karma та Protractor для написання та виконання тестів, забезпечуючи якість та надійність коду [17], [18].

AngularJS має динамічну екосистему сторонніх бібліотек, інструментів і розширень, які розширюють його функціональність і спрощують розробку. Спільнота AngularJS бере активну участь у розвитку екосистеми, надаючи ресурси, документацію та підтримку розробникам [17], [18].

AngularJS включає такі функції, як оптимізація прив'язки даних, одноразове прив'язування та оптимізація циклу обробки даних для підвищення продуктивності та мінімізації непотрібних маніпуляцій DOM. Ці оптимізації підвищують швидкість відгуку та ефективність програм AngularJS [17], [18].

Офіційна підтримка та документація: AngularJS, що розробляється та підтримується Google, користується офіційною підтримкою, регулярними оновленнями та детальною документацією. Наявність офіційних ресурсів гарантує розробникам доступ до надійних рекомендацій і допомоги при створенні додатків AngularJS [17], [18].

В цілому, AngularJS пропонує потужну і всеосяжну платформу для створення сучасних вебзастосунків з функціями, що сприяють організації коду, зручності супроводу, тестування і оптимізації продуктивності. Інтуїтивно зрозумілий синтаксис, багата екосистема та офіційна підтримка роблять його популярним вибором для розробників, які прагнуть створювати динамічні та інтерактивні вебпрограми.

Серед критеріїв порівняння будемо розглядати наступні:

- тип – ця функція підкреслює основну роль та призначення кожної технології, розрізняючи AngularJS як інтерфейсну платформу та Node.js як серверне середовище виконання;

- мета – розуміння передбачуваного випадку використання кожної технології важливо для визначення того, як вони вписуються в стек розробки та які проблеми призначені для вирішення;
- архітектура – порівняння архітектурних шаблонів AngularJS та Node.js дозволяє отримати уявлення про те, як вони керують логікою програми, потоком даних та організацією коду;
- введення залежностей є фундаментальною концепцією як в AngularJS, так і в Node.js, але їх підходи до цього дещо відрізняються. Розуміння цих відмінностей має вирішальне значення для розробки модульного та тестованого коду;
- прив'язка даних – це ключова функція AngularJS, яка полегшує розробку інтерфейсу користувача шляхом автоматичної синхронізації даних між моделлю та поданням. Node.js Будучи середовищем виконання, він не має вбудованої прив'язки даних, але може інтегруватися з інтерфейсними платформами, які надають цю функціональність.
- директиви – AngularJS надає директиви для розширення HTML за рахунок користувацької поведінки, в той час як Node.js більше орієнтований на серверну логіку і не має вбудованої підтримки директив.
- шаблони та подання – AngularJS використовує HTML-Шаблони для визначення компонентів інтерфейсу, тоді як Node.js для створення динамічного HTML-контенту використовує шаблонізатори, такі як EJS або Pug.
- модульність – і AngularJS, і Node.js наголошують на модульності та організації коду, але підходять до цього по-різному через різні ролі в стеку розробки.
- підтримка тестування – тестування має вирішальне значення для забезпечення надійності та якості програмних додатків. Розуміння можливостей тестування AngularJS та Node.JS допомога розробникам у виборі відповідних платформ і стратегій тестування.

– продуктивність – міркування щодо продуктивності важливі для оцінки ефективності та масштабованості програм, створених за допомогою AngularJS та Node.js, особливо з точки зору рендеринга на стороні клієнта порівняно з обробкою на стороні сервера.

– спільнота та екосистема – сила спільноти та екосистеми технології впливає на її впровадження, підтримку та доступність сторонніх інструментів та ресурсів.

Офіційна підтримка розробників технологій забезпечує безперервну розробку, оновлення та виправлення помилок, забезпечуючи впевненість розробникам та організаціям, які використовують ці технології.

Порівняльна характеристика Android Studio з іншими найпоширенішими IDE для розробки мобільних застосунків у таблиці 2.2.

Таблиця 2.2 – Порівняння фреймворків

Критерій	AngularJS	Node,JS
Тип	Інтерфейсний JavaScript-фреймворк	Серверне середовище виконання JavaScript
Мета	Полегшує розробку вебзастосунків на стороні клієнта	Дозволяє розробляти JavaScript на стороні сервера
Архітектура	Відповідає архітектурному шаблону Model-View-Controller (MVC) для розробки на стороні клієнта	Використовує керовану подіями неблокуючу архітектуру вводу-виводу для розробки на стороні сервера
Впровадження залежностей	Підтримує вбудоване введення залежностей для модульного та тестованого коду	Підтримує введення залежностей за допомогою модулів та пакетів npm
Прив'язка даних	Реалізує двостороннє прив'язування даних для автоматичної синхронізації даних між моделлю та поданням	Не має вбудованої прив'язки до даних, але може бути інтегрований з інтерфейсними платформами, такими як AngularJS

Продовження таблиці 2.2

Критерій	AngularJS	Node,JS
Директива	Надає директиви для розширення HTML користувацькими елементами, атрибутами та поведінкою	Не має вбудованих директив, але може бути розширений за допомогою проміжного програмного забезпечення та модулів
Шаблони та подання	Використовує шаблони HTML для визначення інтерфейсу користувача з динамічним прив'язуванням даних та обчисленням виразів	Не має вбудованої підтримки шаблонів, але може використовувати шаблонізатори, такі як EJS або Pug
Модульність	Підтримує модульність за рахунок використання модулів і компонентів	Підтримує модульність за рахунок використання модулів і пакетів npm
Підтримка Тестування	Пропонує вбудовану підтримку модульного тестування, наскрізного тестування та інтеграційного тестування	Надає платформи тестування, такі як Mocha, Jest та Jasmine, для тестування додатків
Уявлення	Оптимізовано для підвищення продуктивності на стороні клієнта завдяки таким функціям, як одноразова прив'язка і оптимізація циклу дайджесту	Відома своєю легкою та масштабованою архітектурою, придатною для створення високопродуктивних серверних додатків
Спільнота та екосистема	Має велику та активну спільноту з великою документацією, ресурсами та сторонніми бібліотеками	Переваги надійної екосистеми пакетів, модулів та інструментів npm за активної участі спільноти розробників з відкритим кодом
Офіційна підтримка	Розроблено та підтримується компанією Google з офіційною підтримкою та регулярними оновленнями	Розроблено та підтримується Фондом Node.js за офіційної підтримки та регулярних оновлень

Це порівняння дає огляд ключових функцій та відмінностей між AngularJS та Node.js, підкреслюючи їх відповідні ролі та можливості у веброботці. Хоча AngularJS в основному використовується для розробки вебзастосунків на стороні клієнта, Node.js використовується для розробки на стороні сервера, і їх можна комбінувати для створення повнофункціональних

Програм на JavaScript. Спираючись на набір сильних характеристик було обрано AngularJS для розробки застосунку.

2.3 Структурна схема розробки

Загальна схема системи за обраною темою на рис. 2.1.

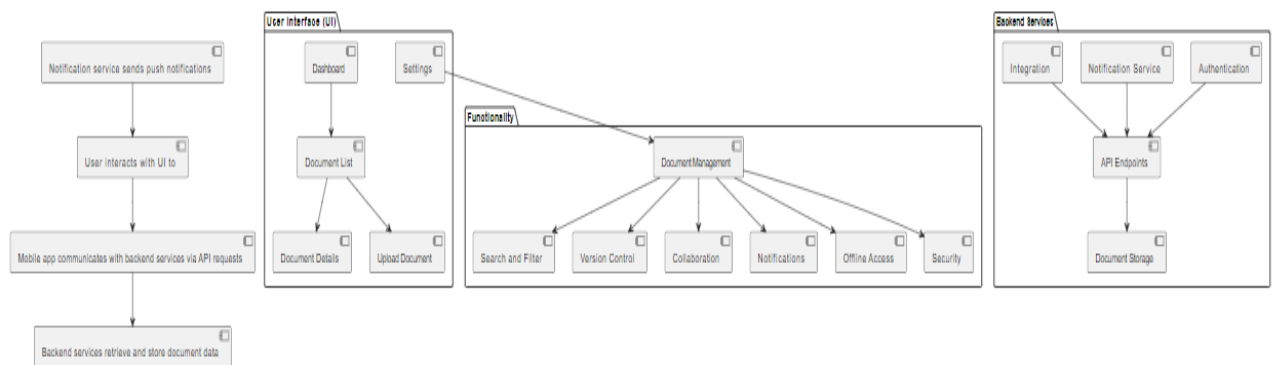


Рисунок 2.1 – Загальна схема системи за обраною темою

Користувальницький інтерфейс (UI):

- інформаційна панель: надає огляд останніх документів, сповіщень та дій користувача;
- список документів: відображає список документів із параметрами сортування, фільтрації та пошуку;
- деталі документа: відображає детальну інформацію про вибраний документ, включаючи метадані, версії та коментарі;
- завантажити документ: дозволяє користувачам завантажувати нові документи зі свого мобільного пристрою або хмарного сховища;
- налаштування: надає можливості для налаштування налаштувань Користувача, повідомлень і параметрів облікового запису.

Функціональність:

- управління документами: дозволяє користувачам переглядати, редагувати, видаляти документи та ділитися ними;

- пошук та фільтрація: дозволяє користувачам шукати документи за ключовими словами, тегами або фільтрами метаданих;
- управління версіями: відстежує версії документів і дозволяє користувачам при необхідності повертатися до попередніх версій;
- спільна робота: полегшує спільну роботу з документами за допомогою коментарів, анотацій та спільного використання;
- повідомлення: повідомляє користувачів про оновлення документів, коментарі та згадки.
- автономний доступ: забезпечує автономний доступ до документів для перегляду та редагування без підключення до Інтернету.
- безпека: реалізує автентифікацію, авторизацію та шифрування для захисту конфіденційних даних документів.

Серверні служби:

- сховище документів: надійно зберігає документи в хмарному або локальному сховищі;
- аутентифікація: виконує автентифікацію та авторизацію користувачів для доступу до документів та функцій;
- кінцеві точки API: надає мобільним клієнтам RESTful API для взаємодії з серверними службами;
- служба сповіщень: надсилає push-сповіщення на мобільні пристрої про оновлення документів та дії;
- інтеграція: інтегрується зі сторонніми службами, такими як хмарні сховища, платформи електронної пошти та обміну повідомленнями.

Потік даних:

- користувач взаємодіє з мобільним інтерфейсом користувача для перегляду, пошуку, завантаження та керування документами.
- мобільний додаток взаємодіє з серверними службами за допомогою запитів API для виконання операцій, пов'язаних з документами.

- серверні служби витягують і зберігають дані документа в базі даних, застосовують бізнес-логіку і обробляють аутентифікацію і авторизацію.
- служба сповіщень надсилає push-сповіщення на мобільні пристрої, щоб попередити користувачів про оновлення документів, коментарі та дії.

Особливості платформи:

- сумісність між платформами: забезпечує безперебійну роботу мобільного додатка на пристроях iOS та Android;
- адаптивний дизайн: адаптує макет інтерфейсу користувача та компоненти до різних розмірів екрана та орієнтації;
- оптимізація продуктивності: оптимізує продуктивність і чуйність програми для зручності роботи користувачів на мобільних пристроях.

2.4 Висновки за розділом 2

Другий розділ пояснювальної записки букі присвячено вибору та обґрунтуванню вибору технічних засобів розробки: технологія, мова програмування, фреймворки, патерн розробки, тощо.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Проєктування моделей системи

Моделювання та проєктування засобами Unified Modeling Language (UML) відіграють вирішальну роль при розробці таких додатків, як мобільний додаток для управління архівами електронної документації. Ось як працюють моделювання та проєктування на мові UML у процесі розробки програми [19], [20].

Аналіз вимог. На початку проєкту для аналізу та документування функціональних вимог програми використовуються діаграми UML, такі як діаграми випадків використання, діаграми дій та діаграми класів. Це включає розуміння потреб користувачів, визначення поведінки системи та визначення ключових функцій та взаємодій [19], [20].

Етап проєктування. Діаграми UML використовуються для створення схеми або дизайну програми. Це включає визначення загальної архітектури, структури компонентів та потоку даних системи. Діаграми класів, компонентів та пакетів допомагають організувати та візуалізувати різні модулі та компоненти програми [19], [20].

Впровадження. На етапі впровадження UML-діаграми служать розробникам довідковим матеріалом для перекладу проєкту в код. Діаграми класів допомагають створювати класи, інтерфейси та взаємозв'язки, тоді як діаграми послідовностей допомагають зрозуміти потік взаємодій між об'єктами та компонентами [19], [20].

Тестування та валідація. Діаграми UML допомагають у тестуванні та перевірці відповідності Програми заданим вимогам. Діаграми випадків використання допомагають створювати тестові сценарії на основі користувацьких сценаріїв, а діаграми дій допомагають визначати робочі процеси та сценарії тестування. Діаграми класів та послідовностей можна

використовувати для перевірки поведінки та взаємодії компонентів системи [19], [20].

Документування. Діаграми UML служать цінною документацією для програми, надаючи уявлення про її структуру, поведінку та функціональність. Вони допомагають донести проєктні рішення, архітектуру системи та деталі реалізації до зацікавлених сторін, членів команди та майбутніх розробників.

Супровід та еволюція. Діаграми UML все ще корисні в процесі супроводу та еволюції програми. Вони допомагають зрозуміти існуючу систему, визначити сфери, які потребують вдосконалення чи модифікації, та документувати зміни, внесені з часом [19], [20].

Протягом усього процесу розробки додатків UML-моделювання та проєкція забезпечують стандартизований та наочний підхід до аналізу, проєктування, впровадження, тестування та документування програмних систем. Вони допомагають покращити комунікацію, співпрацю та взаєморозуміння між учасниками проєкту та сприяють загальному успіху програми [19], [20].

Діаграма класів (рис. 3.1) показує основні класи та їх взаємозв'язки в застосунку для управління архівом електронної документації, включаючи користувачів, документи, інформаційну панель, список документів, деталі документа, завантажуваний документ, налаштування та сповіщення.

Діаграма пакетів (рис. 3.2) представляє модульну структуру застосунку для управління архівами електронної документації, яка організована в пакети з урахуванням функціональності, користувальницького інтерфейсу, внутрішніх сервісів, потоку даних і платформи.

UML діаграма прецедентів (рис. 3.3) ілюструє різні дії, які користувачі та адміністратори можуть виконувати в застосунку для управління архівами електронної документації, включаючи перегляд інформаційної панелі, перегляд документів, пошук документів, завантаження документів, управління документами, налаштування параметрів та отримання сповіщень.

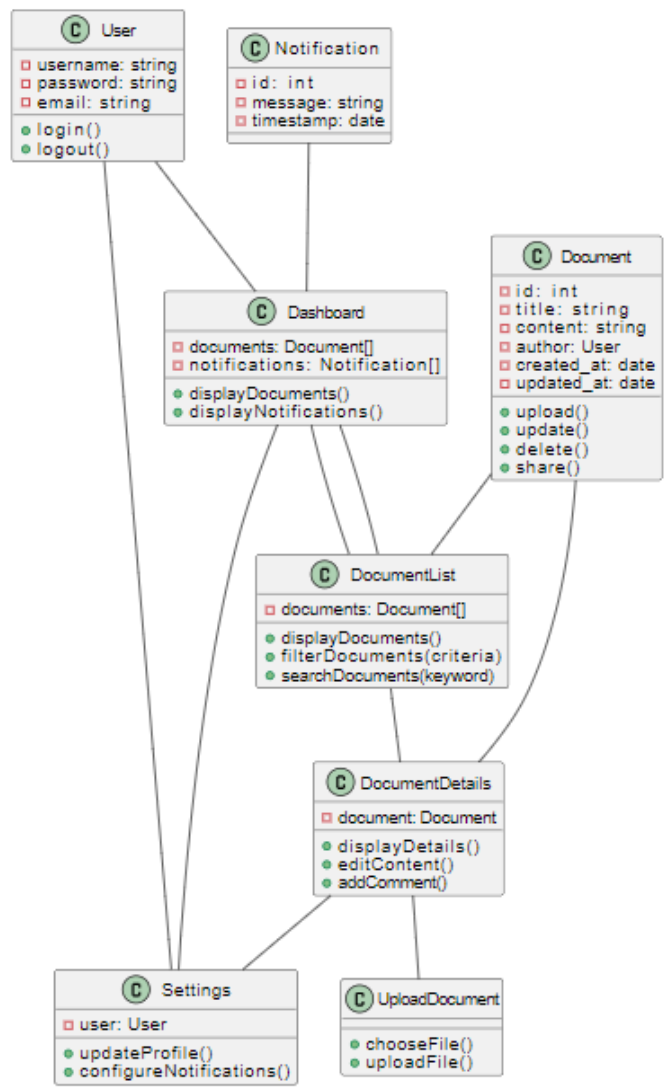


Рисунок 3.1 – Діаграма класів

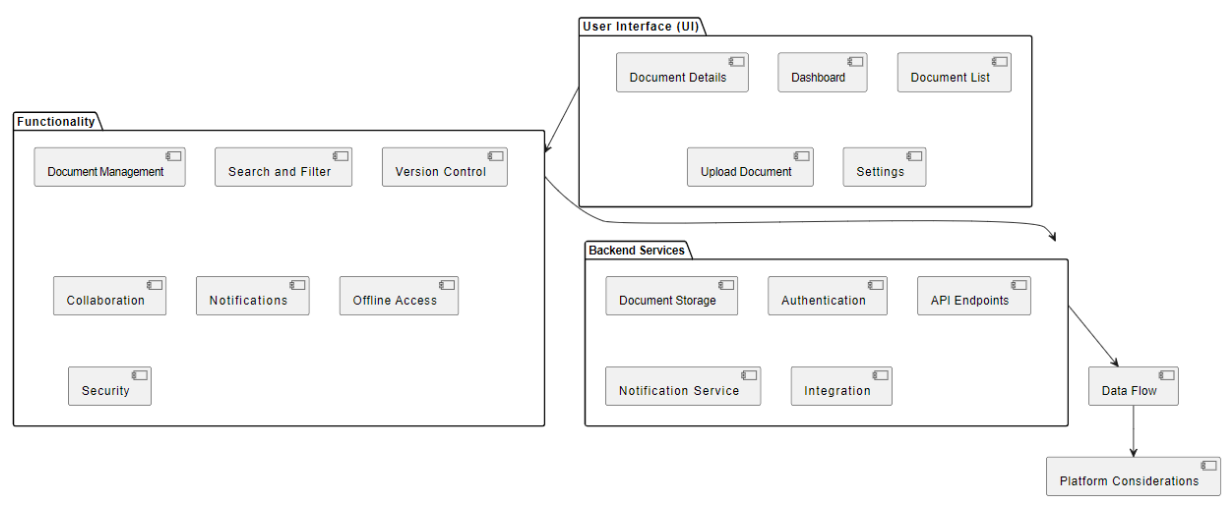


Рисунок 3.2 – Діаграма пакетів

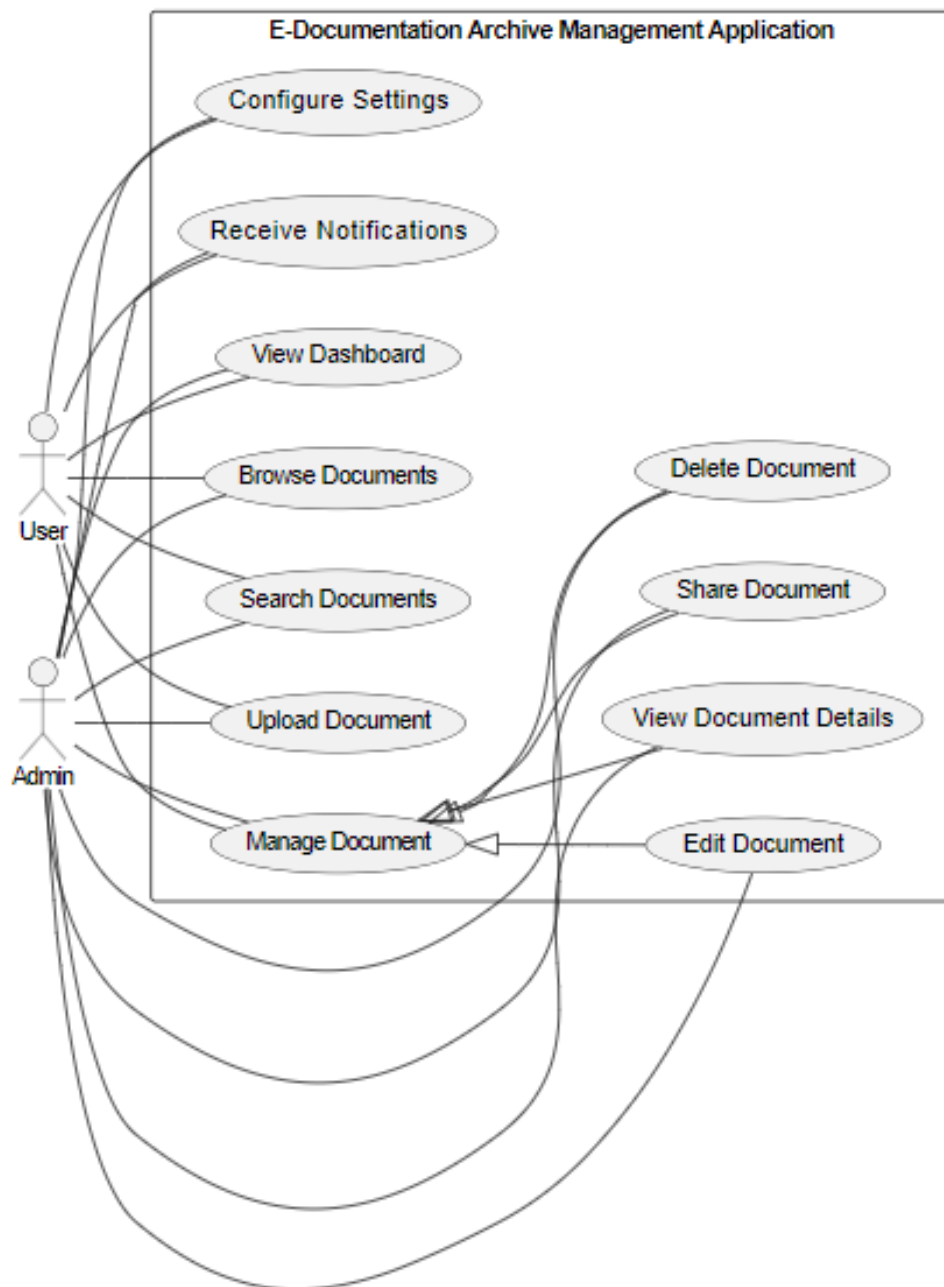


Рисунок 3.3 – Діаграма прецедентів

На діаграмі станів (рис. 3.4) представлені різні стани та переходи в мобільному додатку для управління архівом електронної документації, включаючи процес входу, навігацію інформаційною панеллю, перегляд та пошук документів, а також керування діями з документами, такими як редагування та видалення.

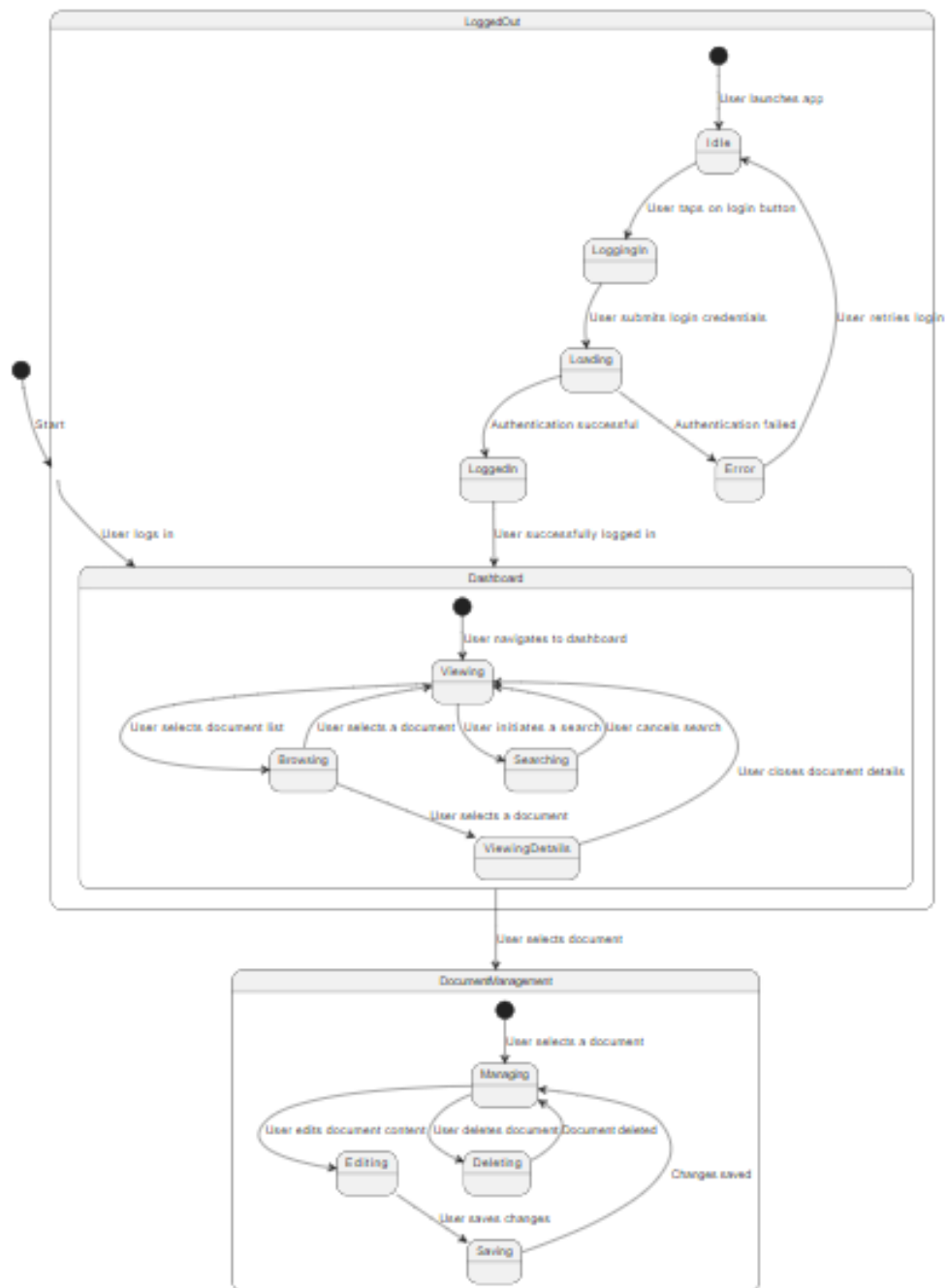


Рисунок 3.4 – Діаграма станів системи

Діаграма діяльностей (рис. 3.5) представляє послідовність дій у застосунку для управління архівом електронної документації, включаючи запуск Програми користувачем, вхід (якщо він ще не ввійшов у систему), автентифікацію, відображення інформаційної панелі, взаємодію користувача з інформаційною панеллю та виконання дій на інформаційній панелі.

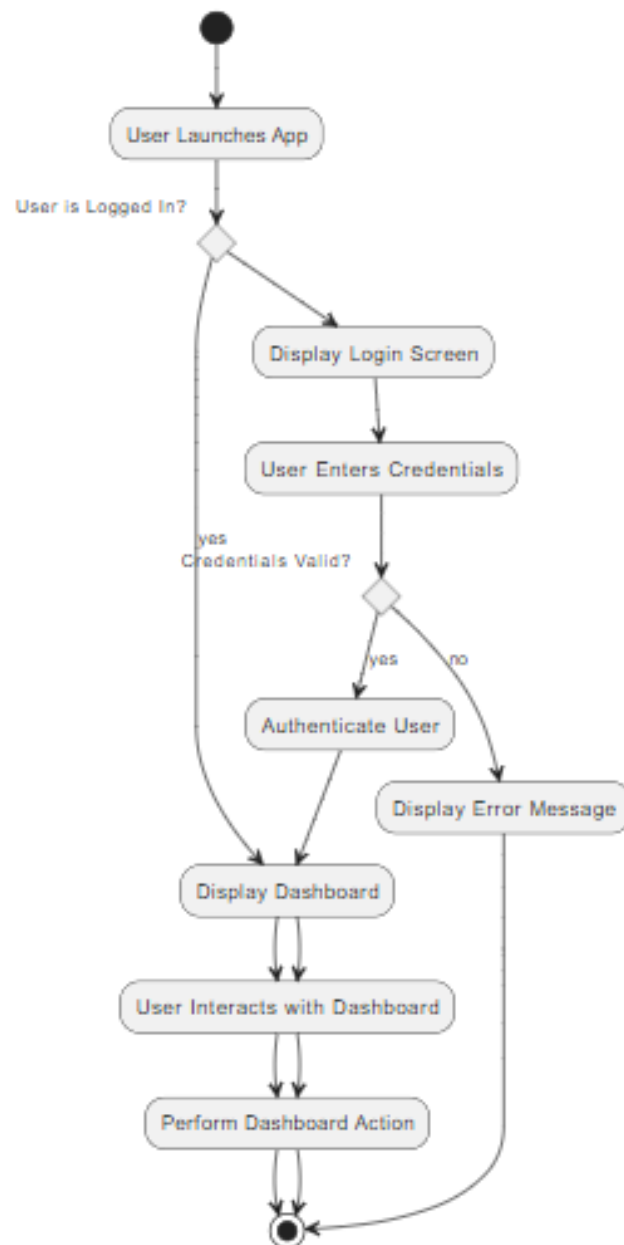


Рисунок 3.5 – Діаграма діяльності

3.2 Проєктування інтерфейсу

Розробка графічного інтерфейсу для програми, орієнтованої на корпоративний документообіг, вимагає ретельного розгляду різних функцій для забезпечення зручності використання, ефективності та задоволеності користувачів.

Зручний інтерфейс – інтерфейс повинен бути інтуїтивно зрозумілим і простим у навігації, дозволяючи користувачам швидко отримувати доступ до

документів і керувати ними без плутанини і розчарування. Чіткі написи, логічна компоновка і знайомі шаблони оформлення сприяють зручності роботи з документами.

Огляд інформаційної панелі. Інформаційна панель надає користувачам централізований доступ до важливої інформації, такої як останні документи, сповіщення та завдання. Вона служить для користувачів відправною точкою, за допомогою якої вони можуть бути в курсі подій і вживати заходів з очікують вирішення питань.

Список документів – добре організований інтерфейс списку документів дозволяє користувачам ефективно переглядати, шукати та фільтрувати документи. Такі функції, як параметри сортування, настроювані перегляди та пагінація, покращують зручність використання, особливо для великих сховищ документів.

Перегляд деталей документа, коли користувачі вибирають документ, детальний перегляд повинен відображати повну інформацію про документ, включаючи метадані, історію версій, пов'язані файли та коментарі. Цей вигляд дозволяє користувачам ефективно переглядати, редагувати та керувати властивостями документа.

Попередній перегляд документа. Функція попереднього перегляду дозволяє користувачам переглядати вміст документа, не завантажуючи та не відкриваючи файл в окремій програмі. Це підвищує доступність і зручність, особливо для швидкого перегляду або довідкових цілей.

Редагування документів – інтеграція із засобами редагування документів або вбудованими можливостями редагування дозволяє користувачам вносити зміни в документи безпосередньо в додатку. Такі функції, як відстеження змін, інструменти співпраці та параметри форматування, покращують продуктивність та якість співпраці.

Керування версіями документів. Інтерфейс керування версіями дозволяє користувачам відстежувати зміни, внесені в документи з часом, і повертатися до попередніх версій, якщо це необхідно. Чітке зазначення історії

версій, міток часу та авторства допомагає користувачам ефективно керувати змінами документів.

Спільний доступ до документів та співпраця – функції обміну документами з колегами, призначення дозволів та спільної роботи з редагування документів сприяють командній роботі та полегшують спілкування. Інструменти співпраці в режимі реального часу, такі як коментарі та анотації, полегшують зворотній зв'язок та обговорення.

Управління документообігом. Інтеграція з інструментами управління документообігом дозволяє користувачам визначати і автоматизувати процеси затвердження документів, направляючи документи призначеним перевіряючим і затверджуючим особам. Чітка візуалізація стану і ходу виконання робочого процесу підвищує прозорість і підзвітність.

Система сповіщень сповіщає користувачів про важливі події, такі як оновлення документів, коментарі або незавершені завдання. Налаштовані сповіщення, надіслані електронною поштою або через додаток, дозволяють користувачам бути в курсі подій та брати участь у управлінні документами.

Профіль користувача та налаштування – користувачі повинні мати доступ до інтерфейсу профілю та налаштувань, щоб налаштувати свої налаштування, керувати сповіщеннями та налаштовувати налаштування додатків. Параметри персоналізації підвищують задоволеність користувачів і адаптують їх роботу до індивідуальних потреб.

Безпека та дозволи – надійні функції безпеки, включаючи автентифікацію користувачів, контроль доступу та шифрування, забезпечують конфіденційність, цілісність та доступність корпоративних документів. Детальне управління дозволами дозволяє адміністраторам визначати ролі користувачів і обмежувати доступ до конфіденційної інформації.

Контрольний журнал і відповідність вимогам – ведення журналу дій користувачів і документів допомагає вести контрольний журнал для цілей дотримання вимог, дозволяючи адміністраторам відстежувати зміни,

відстежувати поведінку користувачів і забезпечувати відповідність нормативним вимогам.

Адаптивність інтерфейсу забезпечує сумісність з різними пристроями і розмірами екранів, включаючи настільні комп'ютери, ноутбуки, планшети і смартфони. Адаптивні макети і гнучкі елементи дизайну оптимізують зручність використання на різних платформах і в різних середовищах.

Включення спеціальних функцій, таких як навігація з клавіатури, сумісність з програмою читання з екрану і режими високої контрастності, забезпечує всеосяжність і відповідність стандартам доступності. Створення інтерфейсу, доступного для користувачів з обмеженими можливостями, підвищує зручність використання та задоволеність користувачів.

Таким чином, розробка графічного інтерфейсу для корпоративного застосунку для управління документами передбачає включення зручних для користувача функцій, інтуїтивно зрозумілою навігації, комплексних інструментів управління документами, можливостей спільної роботи, надійних заходів безпеки і дотримання стандартів доступності. Зосередившись на цих функціях, дизайнери можуть створити інтерфейс, який відповідає потребам корпоративних користувачів і підвищує продуктивність, ефективність і задоволеність користувачів.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Нагальне завдання розробки програми для архівування та збереження документів, зумовлене зростаючим обсягом цифрових документів, необхідністю віддаленого доступу та співпраці, проблемами безпеки та відповідності, готовністю до стихійних лих, підвищенням ефективності, екологічною стійкістю та прагненням до конкурентних переваг за рахунок інновацій. Вирішення цих проблем вимагає негайних дій щодо розробки надійних, зручних у використанні та масштабованих рішень для архівування документів, які відповідають мінливим потребам окремих осіб та організацій у сучасному цифровому світі.

Отже актуальною задачею залишається програмна реалізація застосунку електронного ведення архіву документації.

Система має реалізовувати наступні функціональні вимоги:

- аутентифікація та авторизація користувача;
- збір і завантаження документів;
- організація та зберігання документів;
- пошук і витяг документів;
- перегляд і попередній документів;
- резервне копіювання та аварійне відновлення даних;
- навчання та підтримка користувачів;
- налаштування та масштабованість.

4.2 Вимоги до інтерфейсу

Інтерфейс системи повинен відрізнятися, логічністю, інтуїтивною зрозумілістю та ергономічністю для користувачів. Дизайн має бути адаптивним для доступу з мобільних пристроїв.

4.3 Умови виконання програми

Апаратні вимоги до сервера:

- центральний процесор – від 16 фізичних ядер та 32 потоків з тактовою частотою від 4 GHz;
- оперативна пам'ять формату DDR4, з тактовою частотою від 2600 MHz та об'ємом від 128 Гб;
- накопичувач (формату HDD або SSD) з вільним простором від 2 Гб.

Програмні вимоги до сервера:

- розгорнута серверна ОС, наприклад: Windows Server, Linux Server або macOS Server;
- розгорнута SQL-сумісна СКБД.

Апаратні вимоги до клієнта:

- центральний процесор – від 2 фізичних ядер та 2 потоків з тактовою частотою від 2 GHz;
- оперативна пам'ять об'ємом від 4 Гб;
- накопичувач (формату HDD або SSD) з вільним простором від 1 Гб.

Програмні вимоги до сервера:

- наявність веббраузера актуальної версії.

4.4 Методика та результати тестування

Під час тестування було обрано стратегію поетапного виконання за сценарієм основного алгоритму користування застосунком..

Тож на рис. 4.1 наведено головну сторінку застосунку.

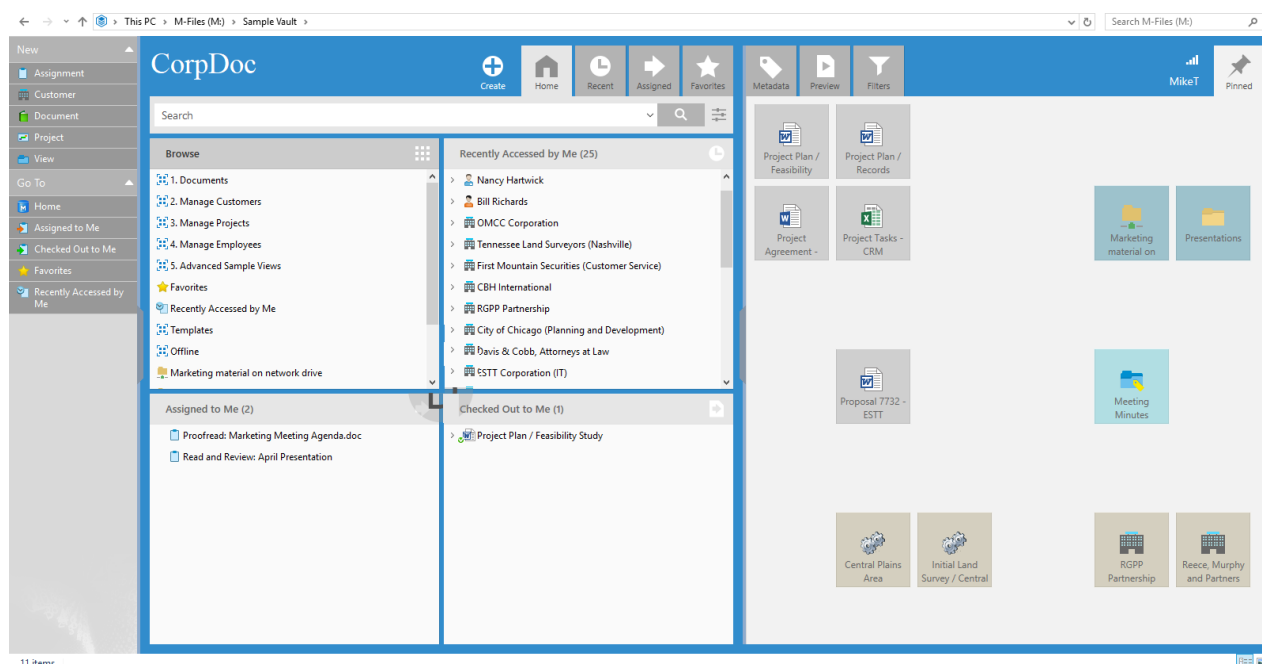


Рисунок 4.1 – Головний екран застосунку

На рис. 4.2 наведено приклад представлення метаданих про файл.

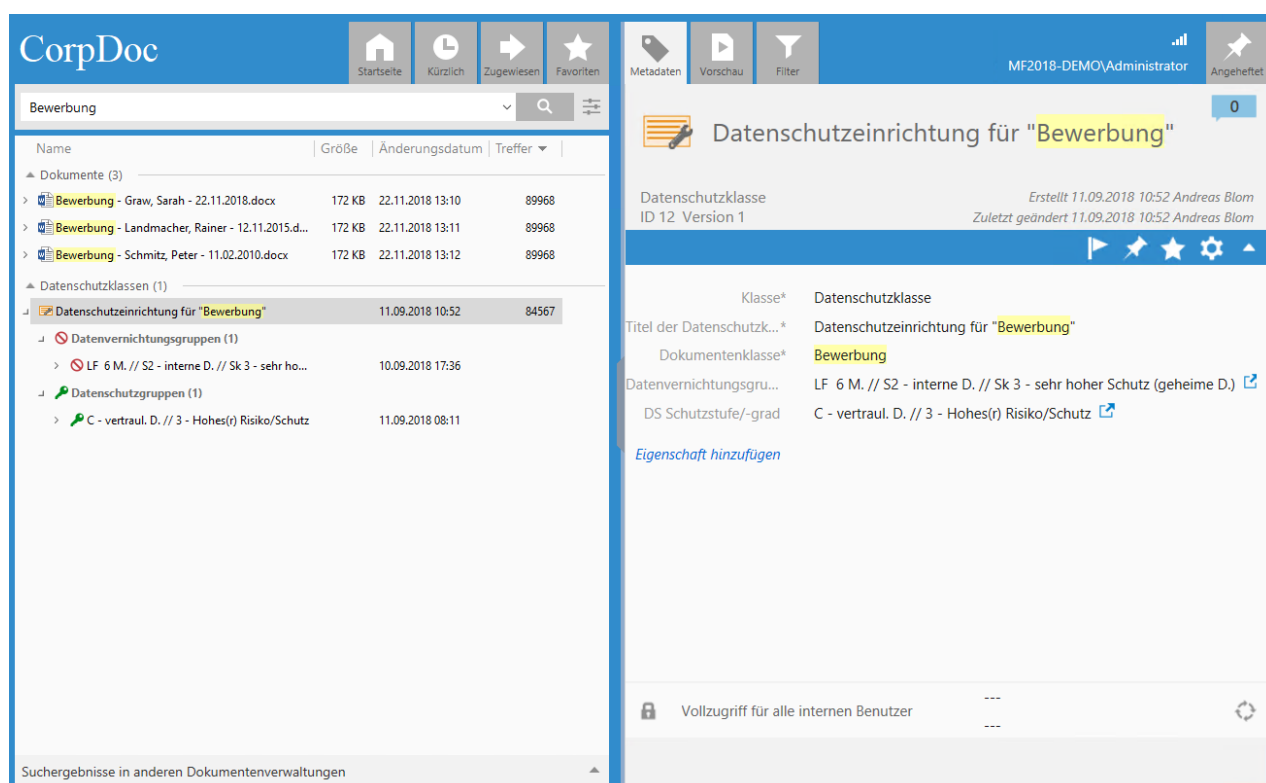


Рисунок 4.2 – Представлення метаданих про файл

На рис. 4.3 наведено взаємодію із метаданими не файлу, а служби.

The screenshot displays the CorpDoc application interface. On the left, a sidebar shows a tree view of projects under the heading 'P-8012 - Möbel Siegfried; Hamburger Möbelgroßhandel GmbH'. The main area shows a list of projects with columns for Name, Erstellungsdatum, Projektmanager, and Aktiv. The right panel provides a detailed view of project P-8012, including its title, number, contact information, and description.

Name	Erstellungsdatum	Projektmanager	Aktiv
P-8012 - Möbel Siegfried; Hamburger Möbelgroßhandel GmbH	16.04.2018 19:26	Andreas Blom	Nein
P-8013 - Möbel Siegfried	16.04.2018 19:36	Andreas Blom	Nein
P-8014 - Giro Bankhaus	16.04.2018 19:41	Andreas Blom	Nein
P-8015 - Harburger Bäderwelt	19.04.2018 07:54	Andreas Blom	Ja
P-8016 - Graphische Werke Landsberg	19.04.2018 08:03	Andreas Blom	Ja
P-8017 - Harburger Bäderwelt	19.04.2018 10:39	Andreas Blom	Ja
P-8018 - Blütenhaus GmbH	07.05.2018 17:52	Andreas Blom	Ja
P-8019 - Blütenhaus GmbH	14.05.2018 14:24	Andreas Blom	Ja
P-8020 - Holz Trading AG	31.05.2018 16:18	Andreas Blom	Ja
P-8021 - Bilabankinn	31.05.2018 16:23	Andreas Blom	Ja
P-8022 - Antarcticopy	31.05.2018 16:27		Ja
P-8023 - Antarcticopy	31.05.2018 16:30		Ja
P-8024 - Bilabankinn	27.06.2018 11:14	Thomas Ratermann	Ja

Project Details for P-8012:

- Klasse*:** Kundenprojekt
- Projekttitle*:** P-8012 - Möbel Siegfried; Hamburger Möbel...
- Projektnummer:** 8012
- Kontakt*:** Möbel Siegfried
- Kontaktperson:** Hr. Dr. Daniel Weisman, Frau Annik Stahl
- Projektmanager:** Andreas Blom
- Abteilung:** Vertrieb
- Aktiv*:** Nein
- Vertrag:** DV - 13003 - Möbel Siegfried
- Schlüsselwörter:** Notebooks, Drucker, Batterien
- Beschreibung:** Kundenprojekt 8012

Рисунок 4.3 – Взаємодія із метаданими не файлу, а служби.

На рис. 4.4 наведено попередній перегляд вмісту файлу.

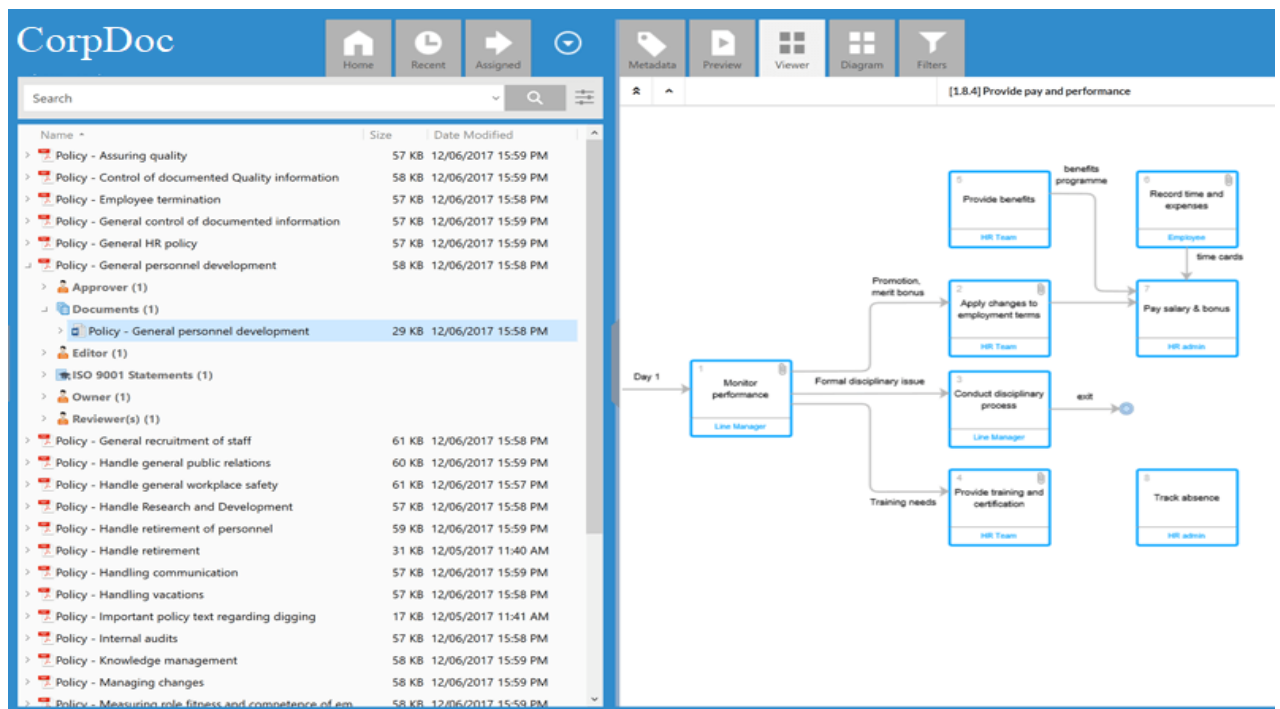


Рисунок 4.4 – попередній перегляд вмісту файлу

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було виконано програмну реалізацію застосунку електронного ведення архіву документації. Для цього було виконано ряд завдань:

- досліджено предметну область цифрового документообігу та архівування документів;
- досліджено переваги та недоліки існуючих аналогічних рішень для архівування документів;
- обґрунтовано вибір технічних засобів реалізації нового застосунку для архівування документів;
- виконано проєктування архітектури нового застосунку;
- виконано програмну реалізацію нового застосунку;
- проведено тестування розробки;
- сформовано висновки на основі виконаної роботи.

Розроблений застосунок був реалізований із використанням: мови програмування JavaScript та фреймворку AngularJS, система керування базами даних MySQL, засоби моделювання UML.

За попередньо розробленим – застосунок тестувався та перевірявся. Отримані результати засвідчили, що тестування дало позитивний результат.

Застосунок реалізує такі функціональні вимоги:

- аутентифікація та авторизація користувача;
- збір і завантаження документів;
- організація та зберігання документів;
- пошук і витяг документів;
- перегляд і попередній документів;
- резервне копіювання та аварійне відновлення даних;
- навчання та підтримка користувачів;
- налаштування та масштабованість.

В результаті розробки реалізовано систему, що відповідає всім вимогам визначеним на початку розробки у вигляді функціональних вимог. Шляхом подальшого покращення розробки є використання специфічних алгоритмів шифрування для більш компактного зберігання.

Усі завдання дипломної кваліфікаційної роботи бакалавра повністю виконано.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Абрамов О. М. Про можливості застосування порталної технології sharepoint у запровадженні електронного документообігу й дистанційної освіти [Текст] / О. М. Абрамов // Вісник Харківської державної академії культури. - 2011. - Вип. 34. - С. 210-217.
2. Ковальова О. Електронний документообіг в межах інформаційного забезпечення досудового розслідування: досвід зарубіжних країн [Текст] / О. Ковальова // Юридичний вісник. - 2022. - № 2. - С. 124-130.
3. Лук'янюк С. В. Щодо питання хмарного електронного документообігу при виконанні науково-дослідних робіт [Текст] / С. В. Лук'янюк // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. - 2023. - № 51. - С. 52-58.
4. Закревська О. Ю. Електронний документообіг в системі управління підприємством [Текст] / О. Ю. Закревська // Інвестиції: практика та досвід. - 2023. - № 9. - С. 71-74.
5. Березанська І. Електронний документообіг інформаційних відділів державних установ в умовах розбудови інформаційного суспільства [Текст] / І. Березанська, П. Дрок, Л. Шимченко // Society. Document. Communication. - 2023. - Вип. 19. - С. 37-59.
6. Гончаренко Є. О. Сучасні моделі і методи проектування і реалізації систем електронного документообігу [Текст] / Є. О. Гончаренко // Управління розвитком складних систем. - 2023. - Вип. 56. - С. 78-84.
7. Янчев А. В. Особливості архівування документів в умовах інформатизації суспільно-економічних відносин [Текст] / А. В. Янчев // Бізнес Інформ. - 2015. - № 5. - С. 208-215.
8. Самохіна Н. Архівування інформаційних ресурсів у системі формування електронної бібліотеки [Текст] / Н. Самохіна // Бібліотечний вісник. - 2012. - № 5. - С. 27-31.

9. OpenText Content Suite Platform / [Electronic resource] –
<https://www.opentext.com/products/content-services-platforms>
10. OpenText Content Suite Platform / [Electronic resource] –
<https://www.confero.tech/ru/solution/opentext/>
11. Alfresco Content Services / [Electronic resource] –
<https://docs.alfresco.com/content-services/7.2/>
12. Alfresco Content Services / [Electronic resource] –
<https://docs.alfresco.com/>
13. Google Drive / [Electronic resource] –
https://www.google.com/intl/uk_UA/drive/
14. Google Drive / [Electronic resource] –
<https://vchymo.com/application/Google-Drive>
15. JavaScript Підручник. Основи вебпрограмування / [Electronic resource] – <https://w3schoolsua.github.io/js/index.html#gsc.tab=0>
16. JavaScript / [Electronic resource] –
<https://developer.mozilla.org/ru/docs/Web/JavaScript>
17. AngularJS / [Electronic resource] – <https://angularjs.org/>
18. AngularJS / [Electronic resource] –
https://www.w3schools.com/angular/angular_intro.asp
19. UML / [Electronic resource] – <https://www.uml.org/>
20. Уніфікована мова моделювання (Unified Modeling Language - UML)
/ [Electronic resource] – <https://www.maxzosim.com/unifikovana-mova-modeluvannia/>

ДОДАТОК А
Технічне завдання

Вступ

Даний застосунок присвячений для ведення процесу документообігу з архівацією документів.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмна реалізація застосунку електронного ведення архіву документації», затверджене наказом Національного університету «Запорізька політехніка» № 168 від 24 квітня 2024 р.

A.2 Основні вимоги до програми, що розробляється

Нагальне завдання розробки програми для архівування та збереження документів зумовлене зростаючим обсягом цифрових документів, необхідністю віддаленого доступу та співпраці, проблемами безпеки та відповідності, готовністю до стихійних лих, підвищенням ефективності, екологічною стійкістю та прагненням до конкурентних переваг за рахунок інновацій. Вирішення цих проблем вимагає негайних дій щодо розробки надійних, зручних у використанні та масштабованих рішень для архівування документів, які відповідають мінливим потребам окремих осіб та організацій у сучасному цифровому світі.

Отже актуальною задачею залишається програмна реалізація застосунку електронного ведення архіву документації.

A.2.1 Вимоги до функціональних характеристик

- а) аутентифікація та авторизація користувача:
 - безпечний вхід користувача за допомогою логіну та паролю;

- управління доступом на основі ролей для управління правами користувача;

б) збір і завантаження документів:

- можливість завантажувати цифрові документи з різних джерел;

- підтримка декількох форматів файлів, включаючи текстові документи, електронні таблиці, презентації, зображення, PDF-файли і т.д;

в) організація та зберігання документів:

- ієрархічна структура папок для організації документів;

- управління метаданими для зв'язування атрибутів з документами;

- управління версіями для відстеження змін і доопрацювань;

г) пошук і витяг документів:

- розширені можливості пошуку за допомогою пошуку за ключовими словами, пошуку за метаданими та параметрів фільтрації;

- швидкий доступ до нещодавно отриманих або змінених документів;

д) перегляд і попередній документів:

- попередній перегляд мініатюр документів на основі зображень;

е) управління життєвим циклом документа:

- політики зберігання документів для управління життєвим циклом документа;

- можливості архівації та очищення для видалення застарілих документів;

- контрольні журнали для відстеження доступу до документів, змін і видалень;

ж) звітність та аналітика:

- заздалегідь визначені звіти для використання документів, активності та журналів доступу;
- настроювані інформаційні панелі для моніторингу показників і продуктивності документів;
- з) резервне копіювання та аварійне відновлення даних:
 - регулярне резервне копіювання даних документів для запобігання втрати даних;
- и) навчання та підтримка користувачів:
 - детальна документація та навчальні матеріали для користувачів;
- к) налаштування та масштабованість:
 - настроювані робочі процеси, шаблони та документація;
 - масштабована архітектура, що дозволяє адаптуватися до зростання обсягу документів і бази користувачів.

A.2.2 Вимоги до інтерфейсу програми

Інтерфейс системи повинен відрізнятися, логічністю, інтуїтивною зрозумілістю та ергономічністю для користувачів. Дизайн має бути адаптивним для доступу з мобільних пристроїв.

A.2.3 Вимоги до надійності

Забезпечення конфіденційності та безпеки даних:

- регулярні аудити безпеки та оцінка вразливостей;
- дотримання відповідних правил та рекомендацій щодо захисту даних для захисту інформації користувача.

A.2.4 Вимоги до складу та параметрів технічних засобів

Апаратні вимоги до сервера:

- центральний процесор – від 16 фізичних ядер та 32 потоків з тактовою частотою від 4 GHz;
- оперативна пам'ять формату DDR4, з тактовою частотою від 2600 MHz та об'ємом від 128 Гб;
- накопичувач (формату HDD або SSD) з вільним простором від 2 Гб.

Програмні вимоги до сервера:

- розгорнута серверна ОС, наприклад: Windows Server, Linux Server або macOS Server;
- розгорнута SQL-сумісна СКБД.

Апаратні вимоги до клієнта:

- центральний процесор – від 2 фізичних ядер та 2 потоків з тактовою частотою від 2 GHz;
- оперативна пам'ять об'ємом від 4 Гб;
- накопичувач (формату HDD або SSD) з вільним простором від 1 Гб.

Програмні вимоги до сервера:

- наявність веббраузера актуальної версії.

ДОДАТОК Б
Опис програми

Б.1 Загальні відомості

Завдання розробки програми для архівування та збереження документів зумовлене зростаючим обсягом цифрових документів, необхідністю віддаленого доступу та співпраці, проблемами безпеки та відповідності, готовністю до стихійних лих, підвищенням ефективності, екологічною стійкістю та прагненням до конкурентних переваг за рахунок інновацій. Вирішення цих проблем вимагає негайних дій щодо розробки надійних, зручних у використанні та масштабованих рішень для архівування документів, які відповідають мінливим потребам окремих осіб та організацій у сучасному цифровому світі.

Отже актуальною задачею залишається програмна реалізація застосунку електронного ведення архіву документації.

Б.2 Функціональне призначення

Система має реалізовувати наступні функціональні вимоги:

- а) аутентифікація та авторизація користувача:
 - безпечний вхід користувача за допомогою логіну та паролю;
 - управління доступом на основі ролей для управління правами користувача;
- б) збір і завантаження документів:
 - можливість завантажувати цифрові документи з різних джерел;
 - підтримка декількох форматів файлів, включаючи текстові документи, електронні таблиці, презентації, зображення, PDF-файли і т. д;
- в) організація та зберігання документів:
 - ієрархічна структура папок для організації документів;

- управління метаданими для зв'язування атрибутів з документами;
- управління версіями для відстеження змін і доопрацювань;
- г) пошук і витяг документів:
 - розширені можливості пошуку за допомогою пошуку за ключовими словами, пошуку за метаданими та параметрів фільтрації;
 - швидкий доступ до нещодавно отриманих або змінених документів;
- д) перегляд і попередній документів:
 - попередній перегляд мініатюр документів на основі зображень;
- е) управління життєвим циклом документа:
 - політики зберігання документів для управління життєвим циклом документа;
 - можливості архівації та очищення для видалення застарілих документів;
 - контрольні журнали для відстеження доступу до документів, змін і видалень;
- ж) звітність та аналітика:
 - заздалегідь визначені звіти для використання документів, активності та журналів доступу;
 - настроювані інформаційні панелі для моніторингу показників і продуктивності документів;
- з) резервне копіювання та аварійне відновлення даних:
 - регулярне резервне копіювання даних документів для запобігання втрати даних;
- и) навчання та підтримка користувачів:
 - детальна документація та навчальні матеріали для користувачів;
- к) налаштування та масштабованість:

- настраювані робочі процеси, шаблони та документація;
- масштабована архітектура, що дозволяє адаптуватися до зростання обсягу документів і бази користувачів.

Б.3 Опис логічної структури

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

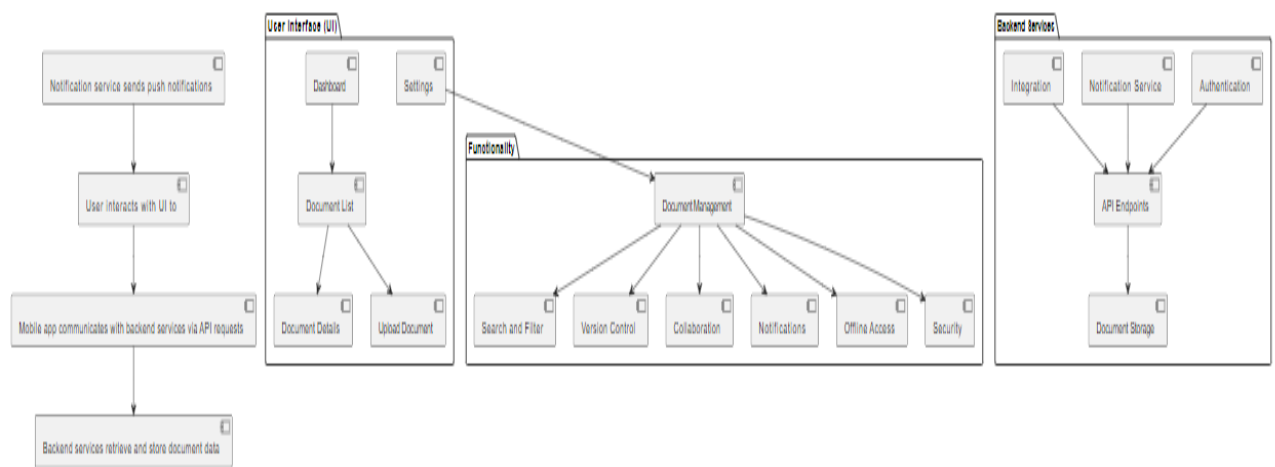


Рисунок Б.1 – Загальна схема системи за обраною темою

Користувальницький інтерфейс (UI):

- інформаційна панель: надає огляд останніх документів, сповіщень та дій користувача;
- список документів: відображає список документів із параметрами сортування, фільтрації та пошуку;
- деталі документа: відображає детальну інформацію про вибраний документ, включаючи метадані, версії та коментарі;
- завантажити документ: дозволяє користувачам завантажувати нові документи зі свого мобільного пристрою або хмарного сховища;
- налаштування: надає можливості для налаштування налаштувань Користувача, повідомлень і параметрів облікового запису.

Функціональність:

- управління документами: дозволяє користувачам переглядати, редагувати, видаляти документи та ділитися ними;
- пошук та фільтрація: дозволяє користувачам шукати документи за ключовими словами, тегами або фільтрами метаданих;
- управління версіями: відстежує версії документів і дозволяє користувачам при необхідності повертатися до попередніх версій;
- спільна робота: полегшує спільну роботу з документами за допомогою коментарів, анотацій та спільного використання;
- повідомлення: повідомляє користувачів про оновлення документів, коментарі та згадки.
- автономний доступ: забезпечує автономний доступ до документів для перегляду та редагування без підключення до Інтернету.
- безпека: реалізує автентифікацію, авторизацію та шифрування для захисту конфіденційних даних документів.

Серверні служби:

- сховище документів: надійно зберігає документи в хмарному або локальному сховищі;
- аутентифікація: виконує автентифікацію та авторизацію користувачів для доступу до документів та функцій;
- кінцеві точки API: надає мобільним клієнтам RESTful API для взаємодії з серверними службами;
- служба сповіщень: надсилає push-сповіщення на мобільні пристрої про оновлення документів та дії;
- інтеграція: інтегрується зі сторонніми службами, такими як хмарні сховища, платформи електронної пошти та обміну повідомленнями.

Потік даних:

- користувач взаємодіє з мобільним інтерфейсом користувача для перегляду, пошуку, завантаження та керування документами.

- мобільний додаток взаємодіє з серверними службами за допомогою запитів API для виконання операцій, пов'язаних з документами.
- серверні служби витягують і зберігають дані документа в базі даних, застосовують бізнес-логіку і обробляють аутентифікацію і авторизацію.
- служба сповіщень надсилає push-сповіщення на мобільні пристрої, щоб попередити користувачів про оновлення документів, коментарі та дії.

Особливості платформи:

- сумісність між платформами: забезпечує безперебійну роботу мобільного додатка на пристроях iOS та Android;
- адаптивний дизайн: адаптує макет інтерфейсу користувача та компоненти до різних розмірів екрана та орієнтації;
- оптимізація продуктивності: оптимізує продуктивність і чуйність програми для зручності роботи користувачів на мобільних пристроях.

Б.4 Використані технічні засоби

Апаратні вимоги до сервера:

- центральний процесор – від 16 фізичних ядер та 32 потоків з тактовою частотою від 4 GHz;
- оперативна пам'ять формату DDR4, з тактовою частотою від 2600 MHz та об'ємом від 128 Гб;
- накопичувач (формату HDD або SSD) з вільним простором від 2 Гб.

Програмні вимоги до сервера:

- розгорнута серверна ОС, наприклад: Windows Server, Linux Server або macOS Server;
- розгорнута SQL-сумісна СКБД.

Апаратні вимоги до клієнта:

- центральний процесор – від 2 фізичних ядер та 2 потоків з тактовою частотою від 2 GHz;
- оперативна пам'ять об'ємом від 4 Гб;

- накопичувач (формату HDD або SSD) з вільним простором від 1 Гб.

Програмні вимоги до сервера:

- наявність веббраузера актуальної версії.

ДОДАТОК В
Текст програми

```

public class MainActY extends ActY {

    //Declare the Btn objects.
    Btn recBtn;
    Btn settingsBtn;

    //Declare variable to hold HZ settings.
    int samplingRateHZ = 60;

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

        //Define where the Btn objects are.
        recBtn = (Btn) findViewById(R.id.recBtn);
        settingsBtn = (Btn)
findViewById(id.settingsBtn);

        //Handle the actual Btn presses and take
the user to the new action.

        //rec Btn
        final Intent intentrecBtn = new Intent();
        intentrecBtn.setClass(MainActY.this,
recActY.class);

        recBtn.setOnClickListener(new OnClickListener()
        {

```

```

        public void onClick(Vw v) {
startActY(intentrecBtn);}

        });

        //Settings Btn
        final Intent intentSettingsBtn = new
Intent();

        intentSettingsBtn.setClass(MainActY.this,
SettingsActY.class);

        settingsBtn.setOnClickListener(new
OnClickListener()
        {
            public void onClick(Vw v) {
startActY(intentSettingsBtn);}

            });

    }

    @Override
    public boolean onCrtOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the
action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

```

```

    }

    public class recActY extends ActY implements
SensorEventListener {

        private SensorMang sensMang;
        private Vw Vw;
        private long lastUpD;

        //Setup file stream.
        FileOutputStream outputFile;

        //Declare Btn Objects
        Btn stopRecBtn;

        int refreshRate = 500;

        @Override
        public void onCrt(Bundle savedInstStt) {
            super.onCrt(savedInstStt);
            setContVw(R.lyut.ActY_rec);

            sensMang = (SensorMang)
getSystemService(SENSOR_SERVICE);
            lastUpD = System.currentTimeMillis();

            //Setup the Btns
            final Intent intentStopRecBtn = new
Intent();

            intentStopRecBtn.setClass(recActY.this,
MainActY.class);

```

```

        stopRecBtn.setOnClickListener(new
OnClickListener()
        {
            public void onClick(Vw v) {
                endRec();
                startActY(intentStopRecBtn);
            }
        });
    }

    @Override
    public boolean onCrtOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the
action bar if it is present.
        getMenuInflater().inflate(R.menu.rec, menu);

        return true;
    }

    @Override
    public void onSensorChng(SensorEvent event) {
        if (event.sens.getType() ==
Sensor.TYPE_ACCELEROMETER) {
            getAccelerometer(event);
        }
    }
}

```

```

private void getAccelerometer(SensorEvent event)
{
    float[] Vals = event.Vals;
    // Movement
    float x = Vals[0];
    float y = Vals[1];
    float z = Vals[2];

    long actualTime =
System.currentTimeMillis();

    if ((actualTime - lastUpD < refreshRate))
    {
        return;
    }

    //UpD the fields if the above doesn't dump
out.

    lastUpD = actualTime;
    UpDXYZFields(x,y,z);

    return;
}

public void UpDXYZFields(float x, float y, float
z) {
    txtVw xValtxt = (txtVw)
findVwById(R.id.txtXVal);

```

```

        txtVw        yValtxt        =        (txtVw)
findVwById(R.id.txtYVal);
        txtVw        zValtxt        =        (txtVw)
findVwById(R.id.txtZVal);

```

```

        //Convert the Vals pass this cont to str
that so of the they can be set to the txtFields.

```

```

        String xStringVal = Float.toString(x);
        String yStringVal = Float.toString(y);
        String zStringVal = Float.toString(z);

```

```

        //Set Vals to the txt fields.
xValtxt.settxt(xStringVal);
yValtxt.settxt(yStringVal);
zValtxt.settxt(zStringVal);

```

```

        //Let us go ahead and try to write this
data to a file on the device.

```

```

        //String FILENAME = "XYZ_output.txt";

```

```

    }

```

```

    @Override

```

```

    public void onAccChng(Sensor sens, int Acc) {
    }

```

```

    public void endRec(){

```

```

        //Put some code to give it right start pos
to XYZ pom.

```

```

    }
}

public class SettingsActY extends ActY {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.lyut.ActY_settings);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the
action bar if it is present.
        getMenuInflater().inflate(R.menu.settings,
menu);

        return true;
    }
}

```

ДОДАТОК Д
Слайди презентації

**Міністерство освіти та науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів**

Програмна реалізація застосунку електронного ведення архіву документації

Виконав: Дмитро ЦЕЛІЩЕВ

ст. групи КНТ-111сп

Керівник: Сергій ЛЕОЩЕНКО

д-р філософ., ст.викладач

Рисунок Д.1 – Слайд 1

Об'єкт, предмет та мета роботи

- * Об'єкт дослідження – процес програмної реалізації застосунку електронного ведення архіву документації.
- * Предмет дослідження – програмні засоби для реалізації застосунку електронного ведення архіву документації.
- * Метою роботи є програмна реалізація застосунку електронного ведення архіву документації.

2

Рисунок Д.2 – Слайд 2

Порівняння існуючих аналогів				
Критерії порівняння	OpenText Content Suite	Alfresco Content Services	Google Drive	CorpDoc
Зберігання документів	Централізоване сховище для зберігання цифрових документів	Централізоване сховище для зберігання цифрових документів	Хмарне рішення для зберігання цифрових документів	Хмарне рішення для зберігання цифрових документів
Організація папок	Ієрархічна структура папок для організації документів	Ієрархічна структура папок для організації документів	Папки та підпапки для організації документів	Ієрархічна структура папок для організації документів
Загальний доступ до документів	Загальний доступ до документів з контролем дозволів	Загальний доступ до документів з контролем дозволів	Загальний доступ до документів з контролем дозволів	Загальний доступ до документів з контролем дозволів
Управління версіями	Контроль версій з відстеженням змін документів	Контроль версій з відстеженням змін документів	Історія версій з відстеженням змін документів	Контроль версій з відстеженням змін документів
Пошук і витяг інформації	Розширені можливості пошуку документів	Розширені можливості пошуку документів	Потужні можливості пошуку документів	Розширені можливості пошуку документів
Безпека та відповідність вимогам	Надійні заходи безпеки та функції відповідності вимогам	Функції безпеки та відповідності нормативним вимогам	Суворі заходи безпеки та функції відповідності вимогам	Суворі заходи безпеки та функції відповідності вимогам
Інтеграція з іншими інструментами	Інтеграція зі сторонніми додатками і системами	Інтеграція зі сторонніми додатками і системами	Інтеграція з додатками та інструментами Google Workspace	Інтеграція зі сторонніми додатками і системами
Автономний доступ	Обмежені можливості автономного доступу	Обмежені можливості автономного доступу	Автономний доступ до документів	Автономний доступ до документів

3

Рисунок Д.3 – Слайд 3

Постановка завдання

Нагальне завдання розробки програми для архівування та збереження документів, зумовлене зростаючим обсягом цифрових документів, необхідністю віддаленого доступу та співпраці, проблемами безпеки та відповідності, готовністю до стихійних лих, підвищенням ефективності, екологічною стійкістю та прагненням до конкурентних переваг за рахунок інновацій. Вирішення цих проблем вимагає негайних дій щодо розробки надійних, зручних у використанні та масштабованих рішень для архівування документів, які відповідають мінливим потребам окремих осіб та організацій у сучасному цифровому світі.

Отже актуальною задачею залишається програмна реалізація застосунку електронного ведення архіву документації.

4

Рисунок Д.4 – Слайд 4

Порівняння мов програмування

Критерій	JavaScript	Python	PHP
Універсальність	Добре підходить як для розробки на стороні клієнта, так і на стороні сервера за допомогою таких фреймворків, як Node.js	Широко використовується для розробки на стороні сервера з використанням таких фреймворків, як Django та Flask	В основному використовується для сценаріїв на стороні сервера, рідше для розробки на стороні клієнта
Розробка на основі повного стека	Підтримує розробку з повним стеком	Підтримує розробку на основі повного стека з використанням Django, Flask та інших фреймворків	В основному використовується для розробки на стороні сервера, можливості розробки на стороні клієнта обмежені
Фреймворки та бібліотеки	за допомогою Node.js величезної екосистеми з такими фреймворками, як React, Angular та Vue.js для розробки на стороні інтерфейсу та Express.js для розробки на стороні сервера	Багата екосистема з використанням таких фреймворків, як Django, Flask та Pyramid для веб-розробки	Обмежені можливості фреймворку в порівнянні з JavaScript і Python, такими фреймворками, як Laravel і Symfony
Кросплатформна Сумісність	Широко використовується для веб-розробки, може бути адаптований для мобільних пристроїв і настільних комп'ютерів за допомогою таких платформ, як React Native і Electron	Широко використовується для веб-розробки, а також для мобільної розробки з використанням таких фреймворків, як Django REST framework і Flask-RESTful	В основному використовується для веб-розробки, рідше для мобільних пристроїв і настільних комп'ютерів
Підтримка спільноти	Велика та активна спільнота розробників з великою кількістю ресурсів, посібників та проектів з відкритим кодом	Активна спільнота розробників з великою документацією, посібниками та бібліотеками з відкритим кодом	Активна спільнота розробників з ресурсами, навчальними посібниками та проектами з відкритим кодом
Швидкість розробки	Полегшує швидке створення прототипів і ітерацію за допомогою таких інструментів, як "гаряча" перезавантаження модулів (HMR)	Забезпечує швидку розробку завдяки лаконічному синтаксису і великим стандартним бібліотекам	Забезпечує швидку розробку з використанням широкого спектру вбудованих функцій і бібліотек

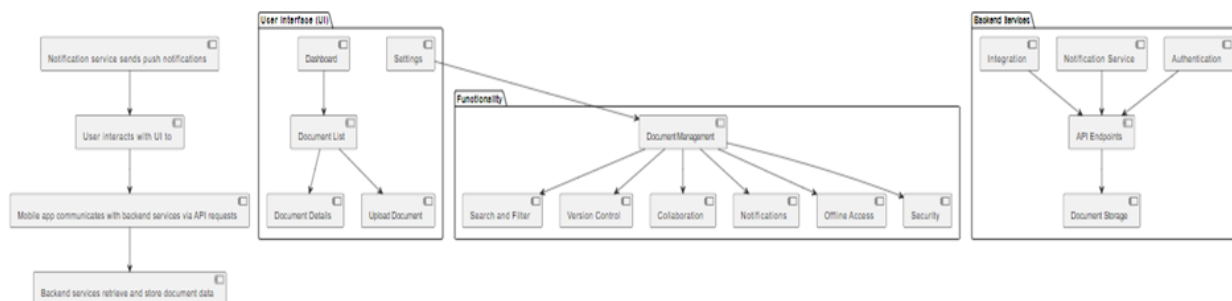
Рисунок Д.5 – Слайд 5

Порівняння фреймворків

Критерій	AngularJS	IntelliJ IDEA
Тип	Інтерфейсний JavaScript-фреймворк	Серверне середовище виконання JavaScript
Мета	Полегшує розробку веб-додатків на стороні клієнта	Дозволяє розробляти JavaScript на стороні сервера
Архітектура	Відповідає архітектурному шаблону Model-View-Controller (MVC) для розробки на стороні клієнта	Використовує керувану подіями неблокуючу архітектуру вводу-виводу для розробки на стороні сервера
Впровадження залежностей	Підтримує вбудоване введення залежностей для модульного та тестованого коду	Підтримує введення залежностей за допомогою модулів та пакетів npm
Прив'язка даних	Реалізує двостороннє прив'язування даних для автоматичної синхронізації даних між моделлю та поданням	Не має вбудованої прив'язки до даних, але може бути інтегрований з інтерфейсними платформами, такими як AngularJS
Директива	Надає директиви для розширення HTML користувацькими елементами, атрибутами та поведінкою	Не має вбудованих директив, але може бути розширений за допомогою проміжного програмного забезпечення та модулів
Модульність	Підтримує модульність за рахунок використання модулів і компонентів	Підтримує модульність за рахунок використання модулів і пакетів npm
Підтримка Тестування	Пропонує вбудовану підтримку модульного тестування, наскрізного тестування та інтеграційного тестування	Надає платформи тестування, такі як Mocha, Jest та Jasmine, для тестування додатків
Спільнота та екосистема	Має велику та активну спільноту з великою документацією, ресурсами та сторонніми бібліотеками	Переваги надійної екосистеми пакетів, модулів та інструментів npm за активної участі спільноти розробників з відкритим кодом
Офіційна підтримка	Розроблено та підтримується компанією Google з офіційною підтримкою та регулярними оновленнями	Розроблено та підтримується Фондом Node.js за офіційної підтримки та регулярних оновлень

Рисунок Д.6 – Слайд 6

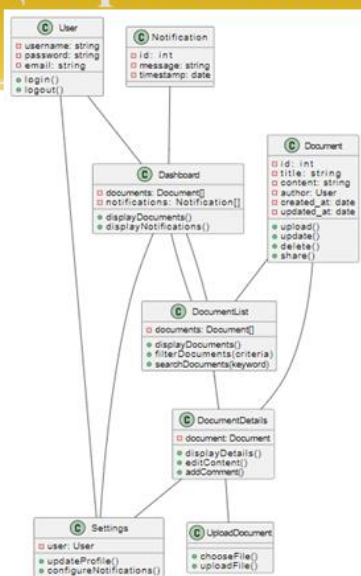
Загальна схема застосунку



7

Рисунок Д.7 – Слайд 7

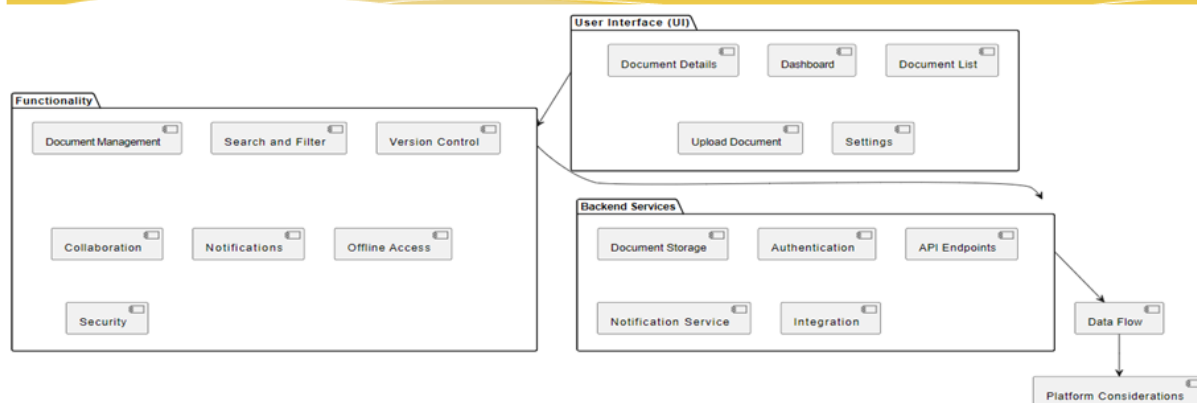
Діаграма класів



8

Рисунок Д.8 – Слайд 8

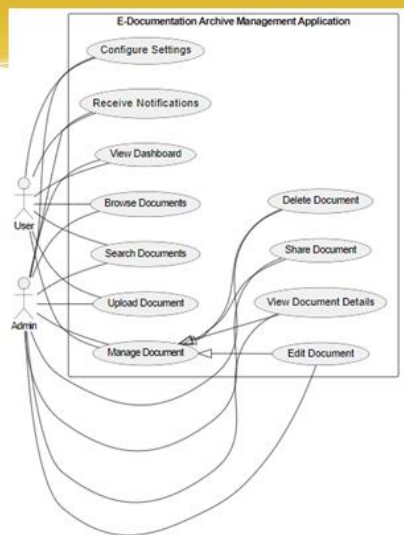
Діаграма пакетів



9

Рисунок Д.9 – Слайд 9

Діаграма прецедентів



10

Рисунок Д.10 – Слайд 10

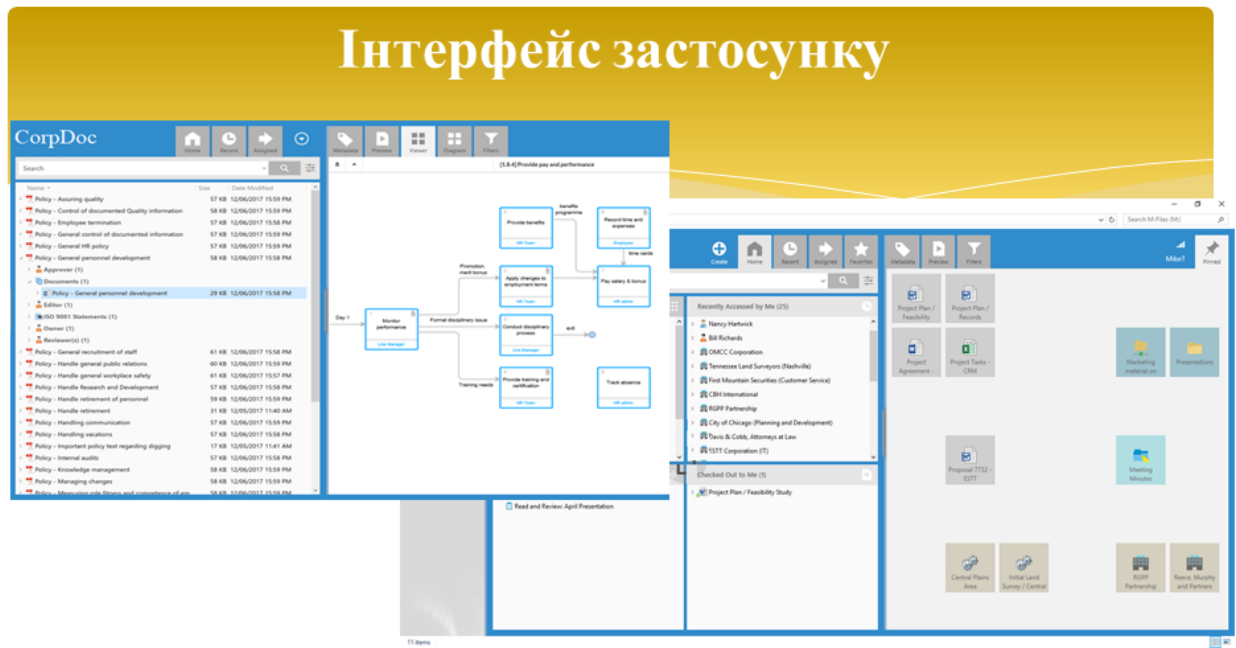


Рисунок Д.11 – Слайд 11

Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було виконано програмну реалізацію застосунку електронного ведення архіву документації. Для цього було виконано ряд завдань:

- * досліджено предметну область цифрового документообігу та архівування документів;
- * досліджено переваги та недоліки існуючих аналогічних рішень для архівування документів;
- * обґрунтовано вибір технічних засобів реалізації нового застосунку для архівування документів;
- * виконано проектування архітектури нового застосунку;
- * виконано програмну реалізацію нового застосунку;
- * проведено тестування розробки;
- * сформовано висновки на основі виконаної роботи.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам визначеним на початку розробки та закріпленим в технічному завданні. Розроблена система є зручною для використання та виконує всі необхідні функції.

Усі завдання дипломної кваліфікаційної роботи бакалавра повністю виконано.

Рисунок Д.12 – Слайд 12