



Метадані

ДОКУМЕНТ

Заголовок

Бак_пояснювальна_ Люклянчук

Автор

Люклянчук

Науковий керівник / Експерт

ІД документу

334320301

ОРГАНІЗАЦІЯ

Назва організації

Taras Shevchenko National University of Luhansk

підрозділ

Taras Shevchenko National University of Luhansk

ЗВІТ

Дата звіту

6/15/2026

Дата редагування

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

4.24%

4.24%

КП 1

14774

Кількість слів

7.09%

7.09%

КЦ

128593

Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		0
Інтервали		0
Мікропробіли		255
Білі знаки		0
Парафрази (SmartMarks)		61

Джерела

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Колір тексту

#	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://docs.mgu.edu.ua/docs/bibliteka/Taran.pdf	25 (0.17 %)

2	2026 КвРбак - Руденко 6/15/2026 Odesa National Maritime University (Технічна кібернетика й інформаційні технології ім. професора Р.В. Меркта)	22 (0.15 %)
3	https://webpifeed.blob.core.windows.net/webpifeed/Partners/efmvc1.1.pdf	21 (0.14 %)
4	https://duikt.edu.ua/repositorii/%D0%9A%D0%B0%D1%84%D0%B5%D0%B4%D1%80%D0%B0%20%D0%86%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D1%96%D1%97%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F/2025/%D0%9A%D0%B2%D0%B0%D0%BB%D1%96%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0%20%D0%B1%D0%B0%D0%BA%D0%B0%D0%BB%D0%B0%D0%B2%D1%80%D0%B0%20%D0%9A%D0%BE%D1%80%D0%BB%D1%8F%D0%BA%D0%BE%D0%B2%D0%B0%20%D0%94.%D0%A1.%202025%20%D1%80%D1%96%D0%BA.pdf	20 (0.14 %)
5	https://www.c-sharpcorner.com/article/jwt-token-authentication-in-asp-net-core-6-web-api-using-three-tier-architecture/	19 (0.13 %)
6	https://webpifeed.blob.core.windows.net/webpifeed/Partners/efmvc1.1.pdf	18 (0.12 %)
7	https://webpifeed.blob.core.windows.net/webpifeed/Partners/efmvc1.1.pdf	18 (0.12 %)
8	https://openarchive.nure.ua/bitstreams/6691b859-b3ac-43e1-beaa-574ed5531fc6/download	18 (0.12 %)
9	https://duikt.edu.ua/repositorii/%D0%9A%D0%B0%D1%84%D0%B5%D0%B4%D1%80%D0%B0%20%D0%86%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D1%96%D1%97%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F/2025/%D0%9A%D0%B2%D0%B0%D0%BB%D1%96%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0%20%D0%B1%D0%B0%D0%BA%D0%B0%D0%BB%D0%B0%D0%B2%D1%80%D0%B0%20%D0%9A%D0%BE%D1%80%D0%BB%D1%8F%D0%BA%D0%BE%D0%B2%D0%B0%20%D0%94.%D0%A1.%202025%20%D1%80%D1%96%D0%BA.pdf	17 (0.12 %)
10	https://duikt.edu.ua/repositorii/%D0%9A%D0%B0%D1%84%D0%B5%D0%B4%D1%80%D0%B0%20%D0%86%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D1%96%D1%97%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F/2025/%D0%9A%D0%B2%D0%B0%D0%BB%D1%96%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0%20%D0%B1%D0%B0%D0%BA%D0%B0%D0%BB%D0%B0%D0%B2%D1%80%D0%B0%20%D0%9A%D0%BE%D1%80%D0%BB%D1%8F%D0%BA%D0%BE%D0%B2%D0%B0%20%D0%94.%D0%A1.%202025%20%D1%80%D1%96%D0%BA.pdf	16 (0.11 %)

База даних RefBooks



#	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
Домашня база даних		
#	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
Програма обміну базами даних (1.02 %)		
#	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

1	9/8/2025 Universitatea Tehnică „Gheorghe Asachi” din Iași (Electronică, Telecomunicații și Tehnologia Informației)	
2	2026 КвРбак - Руденко 6/15/2026 Odesa National Maritime University (Технічна кібернетика й інформаційні технології ім. професора Р.В. Мерктя)	29 (2) (0.2 %)
3	122_Hryb_Anhelina_Serhiivna_it2025 5/21/2025 National University of Food Technologies (Кафедра інформаційних технологій, штучного інтелекту і кібербезпеки)	22 (3) (0.15 %)
4	Асанова_електронний документ_особливості функціонування_керівник 4/24/2025 Kyiv professional college municipal service V.I.Vernadsky Taurida National University (Циклова комісія документознавчих дисциплін)	14 (1) (0.09 %)
5	Електронний документообіг – основа ефективного управління сучасних підприємств 6/11/2026 National Forestry University of Ukraine (ЦДН НЛТУ України)	13 (1) (0.09 %)
6	Програмна платформа DotaHelper для онлайн-навчання гри в Dota 6/12/2022 Odessa National Polytechnic University (IKC, кафедра інженерії програмного забезпечення)	11 (1) (0.07 %)
7	UML-орієнтована інформаційна технологія для неперервних задач максимального покриття з об'єктами довільної форми 1/8/2026 V. N. Karazin Kharkiv National University (KGNU) (ННІ комп'ютерних наук та штучного інтелекту - кафедра інтелектуальних програмних систем і технологій)	11 (1) (0.07 %)
8	2025_122_Ярошик_М.Р. 6/23/2025 Lutsk National Technical University (Lutsk National Technical University)	11 (1) (0.07 %)
9	SUMDU/out2019/Kotenko_master_thesis.pdf 7/23/2019 Sumy State University (SUMDU)	10 (1) (0.07 %)

Інтернет (3.22 %)



#	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
10	https://duikt.edu.ua/repositorii/%D0%9A%D0%B0%D1%84%D0%B5%D0%B4%D1%80%D0%B0%20%D0%86%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D1%96%D1%97%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F/2025/%D0%9A%D0%B2%D0%B0%D0%BB%D1%96%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0%20%D0%B1%D0%B0%D0%BA%D0%B0%D0%BB%D0%B0%D0%B2%D1%80%D0%B0%20%D0%9A%D0%BE%D1%80%D0%BB%D1%8F%D0%BA%D0%BE%D0%B2%D0%B0%20%D0%94.%D0%A1.%202025%20%D1%80%D1%96%D0%BA.pdf	128 (12) (0.87 %)
11	https://webpifed.blob.core.windows.net/webpifed/Partners/efmvc1.1.pdf	108 (10) (0.73 %)
12	https://docs.mgu.edu.ua/docs/bibliteka/Taran.pdf	25 (1) (0.17 %)
13	https://www.c-sharpcorner.com/article/jwt-token-authentication-in-asp-net-core-6-web-api-using-three-tier-architecture/	24 (2) (0.16 %)

14	https://ela.kpi.ua/bitstreams/04af622a-227b-436d-9a1d-8a1f9b07bb55/download	23 (3) (0.16 %)
15	https://ela.kpi.ua/bitstream/123456789/43997/1/Rutkovskay_bakalavr.pdf	22 (3) (0.15 %)
16	https://ela.kpi.ua/bitstream/123456789/53536/1/Smuchok_bakalavr.pdf	19 (2) (0.13 %)
17	https://khntusg.com.ua/wp-content/uploads/2020/09/shishnjak.pdf	18 (2) (0.12 %)
18	https://openarchive.nure.ua/bitstreams/6691b859-b3ac-43e1-beaa-574ed5531fc6/download	18 (1) (0.12 %)
19	http://aspnetmvc.pl/DodatekPDF	16 (2) (0.11 %)
20	https://www.wunu.edu.ua/opp/uf/management_of_socio_culturala_diyalnosti/management_of_socio_culturala_diyalnosti_bakalavr/elektronnyy_dokumentooig/syllabus.pdf	16 (2) (0.11 %)
21	https://ir.nmu.org.ua/jspui/bitstream/123456789/164850/1/23_06_16_%D0%9A%D1%83%D1%82%D0%BD%D0%B8%D1%87%20%D0%A2.%D0%A2.pdf	13 (2) (0.09 %)
22	https://essuir.sumdu.edu.ua/bitstream/123456789/88913/1/Pohorila_bac_rob.pdf	12 (2) (0.08 %)
23	https://ekmair.ukma.edu.ua/bitstreams/830be442-539a-467b-a938-6a356a6bcf25/download	11 (1) (0.07 %)
24	https://eir.kntu.net.ua/jspui/bitstream/123456789/1215/1/%D0%AF%D1%86%D0%BA%D0%B5%D0%B2%D0%B8%D1%87%20%D0%90.%20%D0%92.%20%D0%9F%D1%80%D0%BE%D1%94%D0%BA%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D1%81%D0%B5%D1%80%D0%B2%D1%96%D1%81%D1%83%20%D1%81%D0%BA%D0%BB%D0%B0%D0%B4%D0%B0%D0%BD%D0%BD%D1%8F%20%D0%BF%D0%BE%D0%B7%D0%BE%D0%B2%D0%BD%D0%B8%D1%85%20%D0%B7%D0%B0%D1%8F%D0%B2%20%C2%ABProshuSud%C2%BB.pdf	11 (1) (0.07 %)
25	http://inmad.vntu.edu.ua/portal/static/9133F747-D3FA-4D0D-9703-13937B9C5A98.pdf	7 (1) (0.05 %)
26	https://docs.vntu.edu.ua/getfile.php/5204.pdf	5 (1) (0.03 %)

☒ Список прийнятих фрагментів

#	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)
---	-------	---------------------------------------

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЗ «ЛУГАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА»

Факультет технологій та інформаційних систем
(назва факультету, інституту)
Кафедра інформаційних технологій та систем
(назва кафедри)

Пояснювальна записка до кваліфікаційної роботи
за першим (бакалаврським) рівнем освіти

на тему:

Розробка інформаційної системи управління внутрішніми документами університету із використанням хмарних сервісів Microsoft 365

Виконав: здобувач вищої освіти ²⁴ курсу спеціальності
F2 «Інженерія програмного забезпечення»
(шифр і назва напрямку підготовки, спеціальності)

_____ Дмитро ЛЮКЛЯНЧУК
¹⁷ (прізвище та ініціали) Керівник _____ Володимир ДОНЧЕНКО
(прізвище та ініціали) Рецензент _____ Владислав ІЩЕНКО

Лубни - 2026

ЗМІСТ

ВСТУП 3

РОЗДІЛ 1. АНАЛІЗ ПІДХОДІВ ДО ОРГАНІЗАЦІЇ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ В УНІВЕРСИТЕТАХ 6

1.1. Сучасні підходи до цифровізації документообігу в освітніх установах 6

1.2. Огляд можливостей платформи Microsoft 365 для реалізації документообігу 13

1.3. Аналіз методів електронного підпису та ідентифікації користувачів 20

РОЗДІЛ 2. ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДОКУМЕНТАМИ 26

2.1. Формування вимог до системи та опис бізнес-процесів 26

2.2. Архітектура системи на базі Microsoft 365 36

2.3. Модель внутрішнього електронного підпису та контролю доступу 42

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ СИСТЕМИ ВНУТРІШНЬОГО ЕЛЕКТРОННОГО ПІДПISY 47

3.1. Реалізація структури документів та робочих просторів системи електронного документообігу 47

3.2. Реалізація бізнес-процесів погодження за допомогою Power Automate 55

3.3. Розробка ASP.NET застосунку для сповіщення про зміни електронного документу 62

3.4. Тестування системи електронного документообігу та оцінка її ефективності 70

ЗАГАЛЬНІ ВИСНОВКИ 74

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 77

ДОДАТОК А. 81

ВСТУП

В умовах цифрової трансформації суспільства та особливо в умовах функціонування закладів вищої освіти України під впливом воєнних викликів, питання ефективної організації внутрішнього документообігу набуває критичного значення. Традиційні паперові процеси виявляються не лише повільними та ресурсозатратними, але й вразливими до втрати інформації, дублювання даних та відсутності прозорості управлінських рішень. Водночас сучасні хмарні платформи, зокрема Microsoft 365, відкривають нові можливості для побудови гнучких, масштабованих та доступних інформаційних систем, що забезпечують безперервність роботи університету незалежно від фізичного розташування його підрозділів та персоналу. Це особливо актуально для переміщених університетів, де цифрове середовище фактично стає основою організаційної діяльності.

Важливим аспектом сучасного документообігу є забезпечення належного рівня довіри до електронних документів та процедур їх погодження. Використання кваліфікованого електронного підпису КЕП гарантує юридичну значущість документів, проте його застосування у внутрішніх процесах часто є надмірно складним і повільним. У зв'язку з цим виникає потреба у впровадженні гібридних моделей підпису, що поєднують внутрішні корпоративні механізми підтвердження дій користувачів (через корпоративну пошту, системи авторизації та логування) із зовнішніми засобами електронного підпису для офіційних документів. Такий підхід дозволяє досягти балансу між оперативністю та юридичною значущістю, що є важливим для адміністративних процесів університету.

Додаткової актуальності темі надає необхідність інтеграції різних цифрових інструментів у єдине інформаційне середовище. Використання таких сервісів, як SharePoint, Power Automate, Microsoft Teams та інших компонентів екосистеми Microsoft 365 дозволяє створити комплексну систему, яка не лише забезпечує зберігання документів, але й автоматизує їх рух, контроль доступу, погодження та моніторинг. Це формує нову модель управління документами, орієнтовану на процеси, взаємодію та прозорість, що відповідає сучасним вимогам до цифрового університету.

Отже, вибір теми дослідження обумовлений необхідністю розробки ефективної, гнучкої та безпечної інформаційної системи управління внутрішніми документами університету на основі хмарних технологій, яка здатна забезпечити автоматизацію бізнес-процесів, підвищення прозорості управління, скорочення часу обробки документів та адаптацію до умов розподіленої роботи. Саме поєднання технологічних можливостей хмарних сервісів із організаційними потребами закладу вищої освіти визначає практичну значущість і доцільність обраної теми.

Мета роботи: розробка інформаційної системи управління внутрішніми документами університету на основі хмарних сервісів Microsoft 365 із використанням гібридної моделі електронного підпису.

Для досягнення поставленої мети визначено такі завдання:

- проаналізувати існуючі підходи до організації електронного документообігу; дослідити можливості платформи Microsoft 365 для побудови інформаційних систем;
- визначити вимоги до системи управління документами в університеті; розробити архітектуру системи та модель доступу;
- спроектувати механізми електронного підпису; реалізувати прототип системи з використанням сервісів Microsoft 365;
- провести тестування та оцінити ефективність запропонованого рішення. Об'єкт дослідження: процеси управління внутрішнім документообігом університету,

Предмет дослідження: методи та засоби побудови інформаційної системи електронного документообігу на базі хмарних технологій Microsoft 365.

У роботі використано такі методи дослідження: аналіз і узагальнення наукових джерел, системний аналіз, моделювання бізнес-процесів, проєктування інформаційних систем, експериментальне дослідження та порівняльний аналіз результатів. Інструментами дослідження виступають сервіси Microsoft 365 (SharePoint, Power Automate, Teams, Forms, Outlook), а також засоби моделювання та візуалізації процесів.

У першому розділі роботи здійснюється аналіз сучасних підходів до організації електронного документообігу в закладах вищої освіти, розглядаються проблеми традиційних моделей управління документами та досліджуються можливості хмарних технологій для їх вирішення.

Особлива увага приділяється функціональним можливостям платформи Microsoft 365, а також методам електронного підпису та ідентифікації користувачів, включаючи використання кваліфікованого електронного підпису та внутрішніх корпоративних механізмів підтвердження дій.

У другому розділі розробляється архітектура інформаційної системи управління документами, визначаються функціональні вимоги, описуються бізнес-процеси та моделі взаємодії користувачів. Розглядається структура системи на базі сервісів Microsoft 365, принципи організації доступу до документів, а також проєктується гібридна модель електронного підпису, що поєднує внутрішні та зовнішні механізми підтвердження.

У третьому розділі представлено практичну реалізацію системи, включаючи налаштування середовища Microsoft 365, створення структури документів, реалізацію процесів погодження за допомогою Power Automate та організацію доступу користувачів. Також проведено тестування системи, проаналізовано результати її впровадження та здійснено оцінку ефективності у порівнянні з традиційними підходами до документообігу.

РОЗДІЛ 1. АНАЛІЗ ПІДХОДІВ ДО ОРГАНІЗАЦІЇ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ В УНІВЕРСИТЕТАХ

1.1. Сучасні підходи до цифровізації документообігу в освітніх установах

Сучасні підходи до цифровізації документообігу в освітніх установах доцільно розглядати у ширшому контексті як складову системи документообігу підприємства, що функціонує в умовах нормативно-правового регулювання, організаційних обмежень та технологічного розвитку. В Україні правові засади електронного документообігу визначаються законами ⁴ про електронні документи та електронний документообіг, а також про електронні довірчі послуги, які встановлюють вимоги до створення, зберігання та використання електронних документів і підписів. У цьому контексті особливу роль відіграє КЕП, який забезпечує юридичну значущість документів і прирівнюється до власноручного підпису. Однак нормативна база не лише дозволяє, а й опосередковано вимагає диференційованого підходу до рівнів довіри в документообігу, оскільки не всі внутрішні процеси потребують максимальної юридичної сили, а надмірне використання КЕП створює додаткові організаційні та технічні бар'єри. Теоретичний аналіз понять «система документообігу» (СД) та «система електронного документообігу» (СЕД) вимагає розуміння еволюції управлінських процесів від паперових форм до цифрових екосистем [13]. У широкому сенсі система документообігу - це сукупність взаємопов'язаних принципів, методів та засобів, що забезпечують організацію руху ⁵ документів в установі з моменту їх створення або отримання до завершення виконання або відправлення. Основними характеристиками традиційної СД є чітка регламентація маршрутів, наявність стадій реєстрації, контролю виконання та архівного зберігання. До ключових термінів тут належать документопотік (рух документів у певному напрямку, наприклад, вхідний, вихідний чи внутрішній), життєвий цикл документа та номенклатура справ, яка структурує документаційну базу організації.

Рис. 1.1. Система документообігу університету в часи до цифровізації

Перехід до системи електронного документообігу (СЕД) не є простою оцифровкою паперів, а являє собою організаційно-технічну систему, що забезпечує процес створення, управління доступом та розповсюдження електронних документів у комп'ютерних мережах (рис. 1.2). Визначальною ознакою СЕД є використання електронного цифрового підпису (ЕЦП) або кваліфікованого електронного підпису (КЕП), який надає юридичної сили безпаперовим операціям [6; 7]. Центральним поняттям стає електронний документ, що складається з двох частин: реквізитної (метадані про автора, дату, тип) та змістовної (безпосередньо текст чи дані). СЕД характеризується високою швидкістю передачі інформації, можливістю паралельної роботи кількох користувачів над одним проектом та автоматизованим моніторингом дедлайнів.

Рис. 1.2. СЕД

З рис. 1.2. бачимо основні зміни та переваги цієї схеми (порівняно з паперовою):

- Центральне ядро (СЕД): Весь документообіг відбувається всередині однієї захищеної інформаційної системи. Жодних фізичних папок.
- автоматизований вхід: Канцелярія отримує документи автоматично через системи електронної взаємодії (СЕВ ОБВ) або e-mail. Система сама перевіряє чинність КЕП.
- миттєвий пошук та моніторинг: Червоні пунктирні овали - це сервіси, які працюють постійно. Ректор або виконавець можуть миттєво знайти будь-який документ, а система автоматично нагадує про дедлайни (замість телефонів).
- електронний архів: Документи автоматично передаються в цифровий архів, де вони зберігаються з дотриманням вимог щодо захисту інформації.
- паралельна робота: Декілька виконавців можуть працювати над одним документом одночасно. Табл. 1.1 показує відмінності СД та СЕД за ключовими характеристиками.

Важливим аспектом теоретичного аналізу СЕД є поняття інтероперабельності - здатності СЕД взаємодіяти з іншими інформаційними системами (наприклад, ERP чи CRM) для безшовного обміну даними. Таким чином, якщо традиційна система документообігу фокусується на фізичному переміщенні носія, то сучасна СЕД спрямована на управління контентом та автоматизацію бізнес-процесів, що робить її не просто сховищем файлів, а динамічним інструментом прийняття управлінських рішень.

Таблиця 1.1

Порівняння систем СД та СЕД

Характеристика	Система документообігу (традиційна)	Система електронного документообігу (СЕД)
Носій інформації	Переважно паперовий	Цифровий (файл)
Спосіб передачі	Кур'єрська доставка, пошта, передача з рук в руки	Локальні мережі, Інтернет, хмарні сервіси
Автентифікація	Власноручний підпис та печатка	Електронний підпис (КЕП)
Пошук документів	Ручний (за реєстраційними журналами та папками)	Миттєвий (за атрибутами, ключовими словами або змістом)

Електронний документообіг в університетах України регламентується комплексом законів, постанов та інструкцій, що визначають юридичну силу електронних документів, правила архівування, підписання, безпеки та організації внутрішнього діловодства.

Основні нормативні акти:

²⁰ 1. Закон України «Про електронні документи та електронний документообіг» - визначає юридичні засади використання електронних документів [5].

²⁰ 2. Закон України «Про електронні довірчі послуги» - регулює електронний підпис, печатку, сертифікати [6].

3. Закон України «Про інформацію», «Про захист персональних даних», «Про освіту», «Про вищу освіту» - визначають правила роботи з інформацією, персональними даними, академічною доброчесністю (<https://zakon.rada.gov.ua>).

4. ДСТУ 4163:2020 - встановлює вимоги до оформлення документів [2].

5. Типова інструкція з діловодства (Постанова КМУ №55, 2018) - уніфікувала процеси СЕД для державних установ. Це ключовий документ для університетів. [1]

6. Порядок роботи з електронними документами (Наказ Міністерства №1886/5) - регулює внутрішні процедури електронного архівування. [7] Спеціальні положення ЗВО (наприклад, ТНПУ) адаптують ці вимоги для університетських структур. Вони визначають принципи законності, безпеки, сумісності, прозорості та економічності СЕД [4].

Наукові дослідження також підкреслюють, що впровадження СЕД є ключовим для цифрової модернізації країни, але законодавча база потребує подальшого удосконалення. З наукової точки зору проблема цифровізації документообігу досліджувалася у працях як зарубіжних, так і вітчизняних учених, зокрема в контексті електронного урядування, інформаційних систем управління та цифрової трансформації організацій. Дослідження таких науковців, як Richard L. Nolan [18; 23] та Michael Hammer [8], заклали основу розуміння необхідності реінжинірингу бізнес-процесів при впровадженні інформаційних систем, включаючи документообіг. У сучасних роботах значна увага приділяється переходу від документ-орієнтованих до процесно-орієнтованих систем, де документи розглядаються як частина цифрових потоків даних. В українському науковому просторі питання електронного документообігу досліджуються в контексті інформатизації освіти, розвитку цифрових університетів та впровадження хмарних технологій.

Практичний досвід впровадження електронного документообігу показує, що використання виключно кваліфікованого електронного підпису для всіх типів документів є неефективним. По-перше, це створює значне навантаження на інфраструктуру надавачів довірчих послуг, зокрема банківських установ, які часто виступають посередниками у видачі та обслуговуванні сертифікатів підпису. По-друге, процес накладання КЕП є відносно складним для користувачів і вимагає додаткових дій, що уповільнює внутрішні управлінські процеси. По-третє, значна частина внутрішніх документів (службові записки, погодження, внутрішні заявки) не потребує юридичної сили на рівні офіційних зовнішніх документів, але потребує фіксації факту дії, ідентифікації користувача та можливості аудиту. Таким чином, виникає протиріччя між вимогами безпеки, юридичної значущості та ефективності.

Узагальнюючи розглянуті підходи, можна зробити висновок, що оптимальним напрямом розвитку систем документообігу є використання хмарних середовищ, які забезпечують централізоване зберігання, доступність і інтеграцію сервісів, у поєднанні з багаторівневою моделлю електронного підпису. Така модель передбачає використання внутрішнього електронного підпису для оперативних процесів, що реалізується через механізми автентифікації, підтвердження дій та журналювання в корпоративному середовищі, а також застосування КЕП як зовнішнього інструменту для документів, що потребують юридичної значущості. Саме поєднання цих підходів дозволяє забезпечити баланс між ефективністю, зручністю використання та відповідністю нормативним вимогам, що і визначає основу для подальшого проєктування інформаційної системи управління документами університету.

1.2. Огляд можливостей платформи Microsoft 365 для реалізації документообігу

Платформа Microsoft 365 [9] є комплексним екосистемним рішенням, яке забезпечує повний життєвий цикл управління корпоративним контентом (Enterprise Content Management, ECM) завдяки глибокій інтеграції своїх ключових компонентів. Побудова системи електронного документообігу (СЕД) на базі цієї платформи дозволяє автоматизувати бізнес-процеси, забезпечити централізоване зберігання даних та високий рівень кібербезпеки.

Рис. 1.3. Екосистема MS365 (Microsoft365)

Платформа Microsoft 365 являє собою інтегроване хмарне середовище, що об'єднує широкий спектр сервісів для організації спільної роботи, управління інформацією та автоматизації бізнес-процесів. У контексті побудови систем електронного документообігу її ключовою перевагою є глибока інтеграція між компонентами, що дозволяє формувати єдину цифрову екосистему без необхідності розробки складних програмних рішень з нуля. Завдяки моделі SaaS (рис. 1.4) платформа забезпечує доступ до інструментів з будь-якого пристрою, централізоване адміністрування, автоматичне оновлення та високий рівень відмовостійкості, що є критично важливим для закладів вищої освіти в умовах розподіленої роботи та нестабільної інфраструктури.

Рис. 1.4. Модель SaaS

SaaS (Software as a Service - програмне забезпечення як послуга) - це модель хмарних обчислень, при якій користувач отримує доступ до готового прикладного програмного забезпечення через інтернет. У цій моделі постачальник послуг повністю бере на себе хостинг, управління, обслуговування інфраструктури, оновлення безпеки та технічну підтримку додатка [3].

Основні характеристики SaaS (рис. 1.4):

- Доступ через браузер: Користувачеві не потрібно встановлювати складне ПЗ на свій комп'ютер; достатньо мати веб-браузер та стабільне інтернет-з'єднання.
- Модель підписки: Оплата зазвичай здійснюється щомісяця або щорічно (Pay-as-you-go), що дозволяє компаніям уникати великих капітальних витрат на ліцензії.
- Мультиорендність (Multi-tenancy): Одна копія додатка обслуговує багатьох користувачів, при цьому дані кожного клієнта надійно ізолювані.
- Автоматичні оновлення: Усі патчі та нові функції впроваджуються розробником централізовано, тому всі користувачі завжди працюють з останньою версією.

Центральним елементом реалізації документообігу виступає SharePoint (рис. 1.5), який забезпечує структуроване зберігання документів, управління доступом та підтримку версійності. SharePoint дозволяє організовувати бібліотеки документів, створювати ієрархію папок, налаштовувати метадані та політики зберігання, що дає змогу формалізувати інформаційні потоки відповідно до організаційної структури університету. Важливою функцією є підтримка контролю версій, що забезпечує відстеження змін і можливість повернення до попередніх станів документів, а також журналювання дій користувачів, що формує основу для аудиту [12].

Рис. 1.5. SharePoint

Не менш важливу роль відіграє Power Automate (рис. 1.6), який забезпечує автоматизацію процесів погодження та обробки документів. За допомогою цього сервісу можна реалізувати маршрути руху документів відповідно до визначених бізнес-процесів, включаючи багаторівневі погодження, умовну логіку, сповіщення користувачів та інтеграцію з іншими сервісами. Power Automate дозволяє створювати як прості сценарії автоматизації, так і складні процеси з розгалуженою логікою, що значно підвищує ефективність управління документами та зменшує кількість ручних операцій [10].

Рис. 1.6. Power Automate

Сервіс Microsoft Teams (рис. 1.7) виступає як інтерфейс взаємодії користувачів із системою документообігу, об'єднуючи комунікацію, доступ до документів та робочі процеси в єдиному середовищі. Кожна команда в Teams може відповідати певному типу документів або структурному підрозділу, а інтеграція з SharePoint забезпечує автоматичне створення пов'язаних сховищ даних. Це дозволяє організувати роботу користувачів у звичному середовищі, де документи, обговорення та процеси погодження поєднуються в єдиному інтерфейсі [11].

Рис. 1.7. Microsoft Teams

Додатково використовуються Microsoft Forms для введення структурованих даних та Microsoft Outlook для сповіщень і взаємодії із зовнішніми користувачами.

Практичне застосування платформи Microsoft 365 у системі документообігу університету полягає у формуванні єдиного цифрового середовища, де кожен етап життєвого циклу документа - від створення до архівування - реалізується засобами інтегрованих сервісів. Зокрема, створення документів може ініціюватися через Microsoft Forms або безпосередньо в бібліотеках SharePoint, де автоматично формуються записи з відповідними метаданими (тип документа, автор, підрозділ, дата створення). Це дозволяє стандартизувати процес введення інформації та мінімізувати помилки, пов'язані з ручним заповненням.

Подальший рух документа реалізується за допомогою Power Automate, який забезпечує автоматичне направлення документа на погодження відповідно до заданого маршруту. Наприклад, службова записка може проходити погодження на рівні керівника підрозділу, після чого - адміністрації, а на фінальному етапі - передаватися на підпис. При цьому система автоматично надсилає сповіщення через Microsoft Outlook або Microsoft Teams, фіксує час обробки та результати кожного етапу, що забезпечує прозорість і контроль процесу.

Організаційна структура документообігу може бути відображена через створення окремих команд у Microsoft Teams для різних типів документів або підрозділів університету. Кожна така команда має пов'язане сховище в SharePoint, де документи зберігаються відповідно до визначеної структури, а доступ до них надається на основі ролей користувачів. Це дозволяє реалізувати принцип мінімально необхідних прав доступу, коли співробітник бачить лише ті документи, які стосуються його діяльності. Додатково забезпечується можливість спільного редагування документів у реальному часі, обговорення змін та інтеграції з комунікаційними каналами.

Особливістю застосування Microsoft 365 у системі документообігу є можливість реалізації гібридної моделі електронного підпису. Внутрішній електронний підпис може бути реалізований через механізми підтвердження дій у Power Automate (через функцію Approvals), де фіксується факт погодження документа конкретним користувачем із прив'язкою до його облікового запису. Для документів, що потребують юридичної значущості, передбачається використання КЕП як зовнішнього інструменту, що інтегрується в загальний процес. Таким чином, система дозволяє гнучко поєднувати швидкість внутрішніх процесів із вимогами нормативного середовища, забезпечуючи ефективний і безпечний документообіг.

Узагальнюючи, платформа Microsoft 365 забезпечує всі необхідні функціональні можливості для побудови сучасної системи електронного документообігу: від зберігання та структурування документів до автоматизації процесів і організації взаємодії користувачів. Її модульна структура дозволяє гнучко адаптувати систему до потреб університету, а інтеграція сервісів - реалізувати комплексні сценарії без значних витрат на розробку. Це робить Microsoft 365 доцільною технологічною основою для створення інформаційної системи управління внутрішніми документами з урахуванням сучасних вимог до ефективності, безпеки та доступності.

1.3. Аналіз методів електронного підпису та ідентифікації користувачів

У сучасних інформаційних системах документообігу питання електронного підпису та ідентифікації користувачів безпосередньо пов'язане з рівнем довіри до виконаних дій і юридичною значущістю документів. Доцільно чітко розмежовувати два типи документообігу: юридично значимий, який виходить за межі організації та підпадає під дію нормативно-правового регулювання, і внутрішній корпоративний, що забезпечує управління внутрішніми процесами та не завжди потребує максимальної юридичної сили. Такий поділ дозволяє оптимізувати використання засобів електронного підпису, уникнути надмірного навантаження на користувачів і технічну інфраструктуру, а також підвищити ефективність обробки документів.

Юридично значимий документообіг в Україні базується на використанні КЕП, який відповідно до законодавства прирівнюється до власноручного підпису та забезпечує автентичність, цілісність і невідомість електронного документа. Використання КЕП є обов'язковим для документів, що мають правові наслідки, зокрема договорів, наказів, офіційних листів та звітності. Такий підпис базується на криптографічних механізмах, сертифікатах відкритих ключів і довірчій інфраструктурі, що гарантує високий рівень захисту. Водночас його застосування потребує додаткових ресурсів: отримання сертифікатів, використання спеціального програмного забезпечення, проходження процедур ідентифікації, що ускладнює використання КЕП у масових внутрішніх процесах.

У межах розвитку систем електронного документообігу важливим є не лише використання технологічних платформ, але й інтеграція механізмів електронного підпису та ідентифікації користувачів, що забезпечують довіру до цифрових процесів. Як показує аналіз, у сучасних умовах внутрішній документообіг доцільно будувати на основі корпоративних механізмів підтвердження дій користувачів, які можуть розглядатися як внутрішній електронний підпис. Такий підхід передбачає використання систем автентифікації, авторизації та журналювання, де кожна дія користувача фіксується з прив'язкою до його облікового запису, що забезпечує простежуваність і контроль виконаних операцій. Додатково можуть застосовуватися механізми багатофакторної автентифікації або одноразові коди, що надсилаються через корпоративні канали зв'язку, що підвищує рівень довіри до ідентифікації без істотного ускладнення користувацького досвіду.

Водночас для забезпечення юридичної значущості документів використовується КЕП, який в Україні став ключовим елементом цифрової взаємодії між громадянами, бізнесом і державою. Його застосування охоплює широкий спектр процесів, включаючи подання електронної звітності, укладення договорів, взаємодію з державними реєстрами, банківські операції та оформлення трудових відносин. Для підтримки цих процесів сформовано розвинену інфраструктуру, що включає сервіси електронного документообігу, державні платформи та надавачів довірчих послуг. Така інфраструктура забезпечує високий рівень захисту, проте водночас створює певні обмеження щодо швидкості та зручності використання, особливо в контексті масових внутрішніх процесів.

Нині КЕП охоплює майже всі сфери взаємодії громадянина, бізнесу та держави:

7. державні послуги (G2C та G2B): авторизація на порталі «Дія», подання заяв на отримання соціальних виплат, реєстрація ФОП або ТОВ, отримання витягів з реєстрів.

8. електронна звітність: подання податкових декларацій до Державної податкової служби (ДПС), звітів до Пенсійного фонду та органів статистики.

9. корпоративний документообіг (B2B): підписання договорів з контрагентами, актів виконаних робіт, рахунків-фактур та додаткових угод у цифровому форматі.

10. банківські та фінансові операції: дистанційне відкриття рахунків, підписання кредитних договорів та верифікація транзакцій.

11. трудові відносини: оформлення на роботу (для дистанційних працівників), підписання наказів та заяв про відпустку в межах системи внутрішнього документообігу.

12. медицина та освіта: підписання декларацій з лікарями в системі eHealth, завірення електронних дипломів та вступних заяв.

Для реалізації цих сценаріїв в Україні функціонує розгалужена інфраструктура сервісів:

1. Сервіси для підписання та обміну документами

- Вчасно: Один із найпопулярніших сервісів для бізнесу. Дозволяє масово підписувати акти, договори та обмінюватися ними з контрагентами.

- Paperless: Простий інструмент від ПриватБанку для завантаження, підписання та надсилання документів.

- Document.online: Платформа для автоматизації зовнішнього та внутрішнього документообігу.

- M.E.Doc: Потужне рішення, яке спеціалізується на звітності до державних органів та обміні первинними документами.

2. Державні ресурси

- Портал «Дія»: Розділ «Підписання документів» дозволяє безкоштовно накласти підпис на будь-який файл або перевірити вже підписаний документ.

- Центральний засвідчувальний орган (czo.gov.ua): Офіційний ресурс для перевірки статусу сертифікатів та підпису.

3. Надавачі електронних довірчих послуг (де отримати КЕП)

- Дія.Підпис: Найсучасніший формат (Smart-ID), який зберігається у смартфоні та активується через розпізнавання обличчя.

- Приват24 / Ощадбанк: Найпростіший спосіб для фізичних осіб та ФОП отримати файловий ключ онлайн за кілька хвилин.

- АЦСК органів юстиції та ДПС: Надають ключі як у файловому вигляді, так і на захищених носіях (токенах).

- Cloud Key (Хмарний підпис), яскравим представником якого є Smart-ID або Дія.Підпис, - це технологія, що фактично позбавляє користувача необхідності носити з собою фізичну флешку або файл. Це «золота середина» між безпекою апаратного токена та зручністю мобільного додатка.

З технічної точки зору, реалізація КЕП може здійснюватися за допомогою різних типів носіїв ключів, що відрізняються рівнем безпеки. Файлові ключі є найбільш простими у використанні, однак можуть бути скопійовані або скомпрометовані у разі несанкціонованого доступу до пристрою користувача. Апаратні токени, які в законодавстві визначаються як захищені носії особистих ключів, забезпечують значно вищий рівень безпеки, оскільки ключ генерується та зберігається всередині спеціалізованого пристрою і не може бути скопійований або витягнутий. Додатково вони мають механізми захисту від підбору пароля та виконують криптографічні операції ізольовано від операційної системи. Окремим напрямом розвитку є хмарні підписи, що реалізуються через захищені серверні модулі та мобільні пристрої користувачів, поєднуючи зручність використання з високим рівнем захисту.

Таким чином, можливості платформи Microsoft 365 дозволяють інтегрувати зазначені підходи у єдину систему документообігу, де внутрішні процеси реалізуються через корпоративні механізми підтвердження дій користувачів, а зовнішні - із застосуванням кваліфікованого електронного підпису. Такий підхід забезпечує формування гібридної моделі підпису, яка поєднує оперативність, зручність і відповідність нормативним вимогам, що є ключовою умовою ефективного функціонування сучасної інформаційної системи управління документами.

Для внутрішнього документообігу доцільним є використання корпоративних механізмів підтвердження дій користувачів, які умовно можна розглядати як внутрішній електронний підпис. Такий підпис реалізується через сукупність засобів автентифікації, авторизації та журналювання дій у системі, де кожна дія користувача (створення, погодження, редагування документа) фіксується з прив'язкою до його облікового запису.

Додатковим рівнем підтвердження може виступати багатофакторна автентифікація або одноразові коди, що надсилаються через корпоративну пошту чи інші канали зв'язку. Використання Multi-Factor Authentication або підтвердження через email-код дозволяє підвищити достовірність ідентифікації користувача без суттєвого ускладнення процесу взаємодії із системою.

Порівняння різних методів електронного підпису та ідентифікації дозволяє виділити кілька рівнів довіри. Найвищий рівень забезпечує КЕП, який має юридичну силу та використовується у зовнішніх взаємодіях. Середній рівень довіри формують механізми багатофакторної автентифікації, які підтверджують особу користувача через декілька незалежних факторів (пароль, код, пристрій).

Різниця між файловим носієм та апаратним токеном - це передусім питання фізичного контролю та захисту від копіювання. В українському законодавстві апаратні токени називаються «захищеними носіями особистих ключів» (ЗНОК).

Ось порівняння за ключовими критеріями безпеки:

1. Можливість копіювання

- Файловий ключ: Це звичайний файл (зазвичай з розширенням .jks, .pfx, .dat), який можна скопіювати на флешку, надіслати поштою або завантажити у хмару. Якщо зловмисник отримає доступ до вашого комп'ютера, він може непомітно вкрасти файл ключа.

- Апаратний токен: Це спеціалізований USB-пристрій (схожий на флешку), усередині якого є криптографічний чип. Ключ генерується всередині пристрою і його неможливо витягнути або скопіювати. Навіть якщо ви вставили токен у комп'ютер, програма лише «просить» токен підписати дані, але сам секретний ключ нікуди не передається.

2. Захист від підбору пароля (Brute-force)

- Файловий ключ: Кількість спроб введення пароля не обмежена на рівні самого файлу. Зловмисник може використовувати спеціальне ПЗ для автоматичного підбору пароля до файлу.

- Апаратний токен: Має вбудований захист. Після певної кількості невдалих спроб введення PIN-коду (зазвичай 7-15 разів) токен блокується назавжди, а дані на ньому знищуються.

3. Місце проведення криптографічних операцій

- Файловий ключ: Коли ви підписуєте документ, ключ зчитується в оперативну пам'ять комп'ютера. У цей момент віруси-шпигуни можуть перехопити його.

- Апаратний токен: Усі обчислення відбуваються всередині самого токена. Комп'ютер не бачить секретний ключ, він бачить лише результат підпису.

Базовий рівень забезпечується внутрішнім корпоративним підписом, що ґрунтується на обліковому записі користувача та системі журналювання.

Така ієрархія дозволяє гнучко підходити до організації документообігу, застосовуючи відповідний рівень захисту залежно від типу документа та його значущості. У результаті формується ефективна гібридна модель, яка поєднує оперативність внутрішніх процесів із вимогами юридичної

безпеки.

Cloud Key (Хмарний підпис), яскравим представником якого є Smart-ID або Дія.Підпис, - це технологія, що фактично позбавляє користувача необхідності носити з собою фізичну флешку або файл. Це «золота середина» між безпекою апаратного токена та зручністю мобільного додатка.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДОКУМЕНТАМИ

2.1. Формування вимог до системи та опис бізнес-процесів

Формування вимог до інформаційної системи управління внутрішніми документами університету базується на аналізі існуючих організаційних процесів, нормативних обмежень та потреб користувачів різних рівнів. Університет як складна організаційна структура генерує значний обсяг документів різного типу, зокрема накази, службові записки, заяви, довідки, табелі обліку часу, навчально-методичні матеріали та інші внутрішні документи. Кожен із цих типів має власні правила створення, погодження, зберігання та доступу, що зумовлює необхідність формалізації вимог до системи. Основними функціональними вимогами є забезпечення централізованого зберігання документів, автоматизація маршрутів погодження, підтримка різних рівнів доступу, фіксація дій користувачів та інтеграція із сервісами хмарного середовища, зокрема Microsoft 365. Нефункціональні вимоги включають безпеку, масштабованість, доступність, зручність використання та відповідність нормативним вимогам щодо обробки електронних документів.

Опис бізнес-процесів у системі документообігу передбачає моделювання життєвого циклу документа від моменту його створення до завершення обробки та архівування. Базовий процес включає такі етапи: ініціація документа користувачем, реєстрація та присвоєння метаданих, передача на погодження відповідно до визначеного маршруту, внесення правок, затвердження та, за потреби, накладання електронного підпису. Кожен етап супроводжується фіксацією часу виконання, відповідальних осіб та результатів, що забезпечує прозорість і контроль. Важливою характеристикою є можливість адаптації маршрутів погодження залежно від типу документа, організаційної структури та рівня відповідальності посадових осіб, що досягається за рахунок використання механізмів автоматизації бізнес-процесів.

Рис. 2.1 Життєвий цикл документа в системі документообігу

Особливу увагу приділено ролям користувачів і моделі доступу до системи. У межах університету можна виділити кілька основних ролей: ініціатор документа (викладач або співробітник), погоджувач (керівник підрозділу, декан, адміністрація), адміністратор системи та кінцевий підписант. Для кожної ролі визначаються права доступу до створення, перегляду, редагування та погодження документів. Реалізація такої моделі дозволяє забезпечити принцип мінімально необхідних прав доступу, підвищити рівень безпеки та запобігти несанкціонованим діям. Додатково система повинна підтримувати механізми журналювання, що забезпечують можливість аудиту та відстеження всіх операцій із документами.

Рисунок 2.2 - BPMN схема життєвого циклу документа в системі з альтернативними гілками внутрішнього підпису та КЕП

На рис. 2.2 подано компактну модель життєвого циклу документа в інформаційній системі управління внутрішніми документами університету. Після створення, заповнення реквізитів і погодження документа виконується визначення його типу. Для внутрішніх документів використовується корпоративний електронний підпис, що реалізується через підтвердження дії користувачем та журналювання. Для юридично значимих документів застосовується кваліфікований електронний підпис. Після успішного підписання документ реєструється, передається до архівного сховища та фіксується в журналі системи.

У результаті формування вимог та опису бізнес-процесів визначається концептуальна модель системи документообігу, яка орієнтована на процесний підхід і забезпечує інтеграцію організаційних процедур із можливостями сучасних інформаційних технологій. Така модель дозволяє оптимізувати управлінські процеси, скоротити час обробки документів, підвищити прозорість прийняття рішень і забезпечити гнучкість системи в умовах змін організаційної структури або зовнішніх вимог. Це створює основу для подальшого проєктування архітектури системи та її практичної реалізації.

До опису вимог до системи та бізнес-процесів доцільно додати деталізацію щодо типів документів, ролей користувачів і маршрутів погодження, оскільки саме ці елементи визначають практичну придатність системи до використання в умовах університету. Насамперед система повинна підтримувати роботу з кількома основними типами внутрішніх документів, серед яких можна виділити накази, заяви, службові записки, табелі обліку часу, довідки, навчально-методичні документи, проєкти рішень, подання та інші документи організаційного характеру. Кожен тип документа має власний рівень значущості, набір реквізитів, перелік відповідальних осіб і правила погодження. Наприклад, накази з основної діяльності або кадрові накази, які можуть мати юридичні наслідки, доцільно відносити до документів, що вимагають використання кваліфікованого електронного підпису на завершальному етапі. Заяви працівників, службові записки, внутрішні погодження або оперативні управлінські документи можуть оброблятися із застосуванням внутрішнього корпоративного підпису, якщо вони використовуються виключно в межах внутрішнього середовища університету. Водночас жорстке закріплення типу підпису лише за одним видом документа не завжди є доцільним, оскільки в реальній практиці один і той самий документ залежно від контексту може мати різний рівень значущості. Саме тому більш ефективним підходом є створення динамічної системи налаштувань, у якій для кожного типу документа адміністратор або відповідальна посадова особа може визначити правила обробки: перелік етапів, коло погоджувачів, тип підпису, обов'язковість архівування, строки виконання та інші параметри. Це забезпечує адаптивність системи до змін нормативних вимог і внутрішніх регламентів університету.

Окремого уточнення потребує модель ролей користувачів з урахуванням специфіки структури закладу вищої освіти. Якщо в загальному вигляді ролі вже були визначені як ініціатор документа, погоджувач, підписант і адміністратор, то в університетському середовищі ці ролі повинні бути конкретизовані відповідно до організаційної ієрархії. Ініціаторами можуть виступати викладачі, працівники кафедр, методисти, секретарі, співробітники деканатів, працівники бухгалтерії, відділу кадрів та інших структурних підрозділів. Погоджувачами можуть бути завідувачі кафедр, гаранті освітніх програм, декани, заступники деканів, керівники структурних підрозділів, начальники відділів, головний бухгалтер або юрисконсульт залежно від змісту документа. Фінальними підписантами в університеті можуть бути ректор, проректори, декани або керівники структурних підрозділів у межах делегованих повноважень. Крім того, доцільно передбачити спеціалізовані ролі, наприклад працівника канцелярії чи архіву, який відповідає за реєстрацію, контроль руху документа та завершальне переведення в архівний стан. Така деталізація ролей дозволяє точніше моделювати реальні адміністративні процеси університету та коректно налаштувати права доступу.

Важливим складником бізнес-процесів є маршрути погодження документів, які визначають логіку їх проходження в системі. Маршрут погодження повинен формуватися на підставі типу документа, структурного підрозділу, з якого він ініційований, рівня важливості та встановлених правил

обробки. У найпростішому випадку документ проходить послідовність етапів: створення, первинна перевірка, погодження, підписання, реєстрація, архівування. Однак у реальних університетських процесах маршрут може бути значно складнішим. Наприклад, заява працівника може спочатку надходити завідувачу кафедри, потім до деканату або відділу кадрів, а після цього – до адміністрації. Службова записка щодо фінансових витрат може вимагати погодження керівника підрозділу, бухгалтерії та проректора з адміністративно-господарської роботи. Наказ може проходити погодження у юрисконсульта, кадровій службі, профільному проректорі та лише потім подаватися на підпис ректору. У системі повинна бути реалізована можливість як послідовного, так і паралельного погодження, коли окремі посадові особи отримують документ одночасно. Кожен етап маршруту має супроводжуватися статусом документа, наприклад «створено», «на перевірці», «на погодженні», «погоджено», «повернуто на доопрацювання», «відхилено», «передано на підпис», «підписано», «zareєстровано», «архівовано». Наявність чіткої системи статусів дозволяє відслідковувати поточний стан документа та спрощує контроль за виконанням процесу. Не менш важливою вимогою є підтримка механізмів повернення документа на доопрацювання або відмови в погодженні. Погоджувач повинен мати можливість не лише затвердити документ, але й залишити зауваження, повернути документ ініціатору або попередньому етапу, а також повністю відхилити його із зазначенням причини. Це означає, що маршрут погодження повинен бути не лінійним, а керованим станами і переходами між ними. У разі повернення на доопрацювання система має зберігати історію попередніх версій документа, коментарі до змін та факт повторного подання. У разі відмови документ може або завершувати процес, або переходити в окремий статус для подальшого перегляду адміністрацією. Отже, у межах формування вимог до системи важливо передбачити не лише базові операції створення та зберігання документів, але й гнучкий механізм класифікації типів документів, рольову модель, адаптивні маршрути погодження та систему керування статусами. Саме така постановка вимог створює необхідне підґрунтя для подальшого моделювання системи, зокрема для побудови UML-діаграми варіантів використання, яка відобразить взаємодію користувачів із ключовими функціями розробленої системи.

Рисунок 2.3 - Діаграма варіантів використання інформаційної системи управління внутрішніми документами університету

На діаграмі варіантів використання (рис. 2.3) відображено основних акторів інформаційної системи управління внутрішніми документами університету та їх взаємодію з ключовими функціями системи. До основних акторів належать викладач або співробітник, завідувач кафедри, декан або керівник підрозділу, адміністрація, працівник канцелярії чи архіву, адміністратор системи, а також зовнішній сервіс кваліфікованого електронного підпису. Діаграма охоплює функції створення документа, подання на погодження, погодження, повернення на доопрацювання, відхилення, підписання внутрішнім підписом або КЕП, реєстрації, архівування, а також налаштування типів документів, маршрутів погодження та правил доступу.

На основі проведеного аналізу сучасних підходів до цифровізації документообігу, можливостей хмарного середовища та особливостей організації бізнес-процесів університету сформульовано систему вимог до інформаційної системи управління внутрішніми документами. Вимоги охоплюють як функціональні, так і нефункціональні аспекти та спрямовані на забезпечення ефективності, безпеки та адаптивності системи. До функціональних вимог належить, передусім, забезпечення можливості створення, редагування та зберігання документів різних типів, що обґрунтовується необхідністю підтримки основних адміністративних процесів університету, зокрема роботи з наказами, заявами, службовими записками та табелями обліку часу. Важливою вимогою є підтримка динамічної системи типів документів, що дозволяє змінювати їх характеристики, маршрути погодження та правила підпису відповідно до внутрішніх регламентів і зовнішніх нормативних змін. Такий підхід впливає з аналізу, проведеного в попередніх розділах, де було встановлено, що жорстко задані шаблони не відповідають реальній практиці функціонування університету.

Система повинна забезпечувати реалізацію гнучких маршрутів погодження документів із можливістю налаштування послідовності та складу погоджувачів. Це обґрунтовується складною організаційною структурою університету, де документи можуть проходити через різні рівні управління: кафедра, деканат, адміністрація. Додатково повинна підтримуватися можливість паралельного погодження, повернення документа на доопрацювання та відмови в погодженні з фіксацією причин. Така вимога забезпечує прозорість процесу прийняття рішень і відповідає принципам процесного підходу до управління.

Окремою вимогою є підтримка двох рівнів електронного підпису: внутрішнього корпоративного підпису та кваліфікованого електронного підпису. Необхідність такої вимоги обґрунтовується аналізом нормативної бази та практики використання електронних підписів, де встановлено, що не всі документи потребують юридично значущого підпису. Використання внутрішнього підпису дозволяє значно спростити та пришвидшити обробку внутрішніх документів, тоді як застосування КЕП доцільне лише для документів із зовнішньою або юридичною значущістю. Система повинна забезпечувати можливість вибору типу підпису залежно від типу документа або сценарію використання.

Важливою функціональною вимогою є реалізація ролеорієнтованої моделі доступу, яка враховує специфіку університету. Це передбачає надання різних прав доступу для викладачів, завідувачів кафедр, деканів, адміністрації, працівників канцелярії та адміністраторів системи. Така вимога обґрунтовується необхідністю забезпечення інформаційної безпеки та відповідності принципу мінімально необхідних прав доступу. Система повинна забезпечувати ведення журналу дій користувачів і історії змін документів. Це обґрунтовується потребою у забезпеченні контролю, аудиту та підтвердження факту виконання дій, що є особливо важливим у контексті внутрішнього електронного підпису.

Журналювання дозволяє відстежувати повний життєвий цикл документа та підвищує рівень довіри до системи.

Необхідною є також підтримка статусної моделі документів, яка відображає етапи їх обробки: створення, погодження, підписання, реєстрація, архівування тощо. Це забезпечує прозорість процесу документообігу та спрощує управління документами для користувачів. Додатково система повинна реалізовувати механізми сповіщень користувачів про зміни статусу документа або необхідність виконання дій, що підвищує оперативність взаємодії.

До нефункціональних вимог належить забезпечення інтеграції з хмарною платформою Microsoft 365, що обґрунтовується її широкими можливостями щодо зберігання даних, організації спільної роботи та автоматизації процесів. Система повинна бути масштабованою та доступною, що особливо актуально в умовах дистанційної роботи університету. Важливою є вимога до зручності користування, оскільки ефективність впровадження системи значною мірою залежить від рівня її прийняття користувачами.

Окремо слід виділити вимоги до безпеки, які включають використання механізмів автентифікації, зокрема багатофакторної автентифікації або підтвердження через корпоративну електронну пошту. Це обґрунтовується необхідністю захисту персональних даних і службової інформації. Також важливою є вимога до надійності зберігання даних і резервного копіювання, що забезпечує збереження інформації в умовах технічних збоїв або зовнішніх загроз.

Таким чином, сформована система вимог відображає як загальні тенденції розвитку електронного документообігу, так і специфіку функціонування університету. Вона створює основу для подальшого проєктування архітектури системи та її реалізації із використанням сучасних хмарних технологій.

2.2. Архітектура системи на базі Microsoft 365

Логічна архітектура системи управління внутрішніми документами університету на базі Microsoft 365 будується за принципом функціонального розподілу між основними хмарними сервісами платформи (рис. 2.4). Ядром системи виступає SharePoint, який забезпечує централізоване зберігання документів, метаданих, журналів дій, прав доступу та версій. Інтерфейсним середовищем взаємодії користувачів із системою є Microsoft Teams, через який забезпечується доступ до документів, сповіщень, каналів комунікації та пов'язаних робочих просторів. Реалізація бізнес-процесів погодження, підписання, зміни статусів і маршрутизації документів покладається на Power Automate, який виконує роль процесного рушія системи. У концептуальному аспекті така побудова має певну схожість із архітектурою MVC, де SharePoint можна співвіднести з рівнем даних, Teams - із рівнем представлення, а Power Automate - із рівнем керування. Водночас, з огляду на хмарну природу рішення, більш точним є визначення цієї моделі як багаторівневої сервісно-орієнтованої архітектури з процесною організацією взаємодії компонентів. Такий підхід забезпечує гнучкість, масштабованість, адаптивність до організаційних змін та ефективну інтеграцію засобів документообігу в єдине цифрове середовище університету.

Якщо проводити аналогію з MVC, то SharePoint у такій системі найбільше наближається до рівня Model, оскільки саме він виступає ядром зберігання даних, документів, метаданих, бібліотек, прав доступу та версійності. У ньому зосереджується структурована інформація про документи, їх атрибути, зв'язки, статуси та історію змін. Microsoft Teams у цьому випадку можна умовно розглядати як рівень View, тобто як користувацький інтерфейс доступу до системи, через який співробітники взаємодіють із документами, сповіщеннями, командами та процесами. Power Automate тоді виконує роль, близьку до Controller, оскільки саме він реалізує логіку обробки подій, запускає маршрути погодження, керує переходами документа між етапами, викликає сповіщення та змінює стани об'єктів у системі.

Рисунок 2.4 - Логічна архітектура системи управління внутрішніми документами університету на базі Microsoft 365

Наступним рівнем деталізації логічної архітектури системи є її структурна організація (рис. 2.5), яка визначає, яким чином елементи хмарного середовища Microsoft 365 відображають реальні об'єкти та процеси університетського документообігу. Ключовим принципом побудови такої структури є відповідність між організаційною логікою роботи з документами та структурою цифрового середовища. У запропонованій моделі Microsoft Teams використовується як верхній рівень організації доступу та взаємодії, де кожна команда або канал відповідає певному типу документів або функціональному напрямку діяльності, наприклад «Накази», «Заяви», «Службові записки», «Табелі обліку часу». Такий підхід дозволяє забезпечити логічне групування документів і користувачів, спростити навігацію та створити зрозуміле робоче середовище.

Рисунок 2.5 - Структурна архітектура системи управління внутрішніми документами університету на базі Microsoft 365

У межах кожної команди Microsoft Teams реалізується інтеграція з SharePoint, який виступає безпосереднім середовищем зберігання документів. Для кожного типу документів створюються відповідні бібліотеки (document libraries), які виконують функцію структурованих сховищ із підтримкою метаданих, версійності та журналювання змін. Наприклад, бібліотека «Накази» може містити додаткові поля, такі як номер наказу, дата, тип наказу, відповідальний підрозділ, статус документа, тоді як бібліотека «Заяви» - інший набір атрибутів, що відповідає специфіці цього типу документів. Така організація дозволяє не лише зберігати файли, але й формувати структуровану інформаційну базу, придатну для пошуку, фільтрації та аналітики.

Особливу роль у структурі системи відіграє модель прав доступу, яка безпосередньо пов'язана з ролями користувачів. У межах Microsoft 365 доступ до команд, каналів і бібліотек SharePoint налаштовується на основі груп користувачів, що відповідають організаційній структурі університету. Викладачі та співробітники мають доступ до створення документів у межах своїх підрозділів, завідувачі кафедр і декани - до їх перегляду та погодження, адміністрація - до підписання та затвердження, а працівники канцелярії - до реєстрації та архівування. Такий підхід реалізує принцип ролівого доступу та забезпечує розмежування прав відповідно до функціональних обов'язків користувачів. Важливою особливістю є можливість гнучкого налаштування доступу як на рівні всієї бібліотеки, так і на рівні окремих документів, що дозволяє враховувати конфіденційність окремих типів інформації.

Таким чином, структурна модель системи базується на взаємопов'язаному використанні Microsoft Teams як організаційного та інтерфейсного рівня, SharePoint як середовища зберігання та управління документами, а також ролівої моделі доступу як механізму забезпечення безпеки та керування правами. Така структура забезпечує узгодженість між цифровим середовищем і реальною діяльністю університету, дозволяє ефективно організувати роботу з різними типами документів і створює основу для реалізації автоматизованих бізнес-процесів погодження та підписання. Як узагальнюючий результат розроблення архітектури системи додамо UML-діаграму компонентів (рис. 2.6), яка відображає основні програмно-логічні складові інформаційної системи управління внутрішніми документами університету та зв'язки між ними.

Рисунок 2.6 - Узагальнена UML-діаграма компонентів.

Така діаграма дозволяє перейти від опису окремих сервісів Microsoft 365 до цілісного уявлення про систему як сукупність взаємодіючих компонентів. У межах цієї моделі користувачі взаємодіють із системою через інтерфейсний компонент, роль якого виконує Microsoft Teams. Даний компонент забезпечує доступ до робочих просторів, документів, статусів, повідомлень і командної взаємодії. Він пов'язаний із компонентом керування документами, який реалізується на базі SharePoint і відповідає за зберігання файлів, метаданих, версій, структури бібліотек і журналів дій.

Окремим важливим компонентом системи є модуль маршрутизації та автоматизації процесів, реалізований засобами Power Automate. Саме він забезпечує виконання бізнес-логіки: формування маршрутів погодження, зміну статусів документів, повернення на доопрацювання, передачу на підпис і надсилання сповіщень. Поряд із цим у системі виділяється компонент керування доступом і безпекою, що реалізує автентифікацію користувачів, роліову модель, розмежування прав доступу та механізми підтвердження дій, зокрема через внутрішній корпоративний підпис. Для документів, які мають юридичну значущість, передбачено окремий зовнішній компонент кваліфікованого електронного підпису, що інтегрується в загальний процес документообігу.

Завершальними складовими архітектури виступають компоненти реєстрації та архівування, а також модуль сповіщень і введення даних.

Перший із них забезпечує переведення документа до завершального стану, його реєстрацію, збереження в архіві та підтримку журналювання.

Другий пов'язаний із використанням Forms, Outlook та інших сервісів Microsoft 365 для ініціювання документів, введення реквізитів і інформування користувачів про зміни стану документа. Таким чином, узагальнена діаграма компонентів демонструє, що система має модульну структуру, побудовану на інтеграції спеціалізованих хмарних сервісів, де кожен компонент виконує визначену функцію, а сукупність їх взаємодії

забезпечує повний життєвий цикл внутрішнього документа університету.

2.3. Модель внутрішнього електронного підпису та контролю доступу

Попередні підрозділи містять окремі аспекти реалізації електронного підпису та контролю доступу, зокрема аналіз засобів підпису, визначення ролей користувачів і організацію бізнес-процесів. Однак для побудови цілісної інформаційної системи необхідно узагальнити ці підходи у вигляді єдиної моделі, що визначає правила ідентифікації, підпису, контролю доступу та аудиту. Саме це завдання вирішується в даному підрозділі. У межах проєктування інформаційної системи управління внутрішніми документами університету ключовим елементом є формування узагальненої моделі електронного підпису та контролю доступу, яка забезпечує баланс між зручністю використання, швидкістю обробки документів і дотриманням нормативних вимог. Запропонована модель (рис. 2.7) базується на дворівневому підході до електронного підпису, що передбачає використання внутрішнього корпоративного підпису для більшості операцій внутрішнього документообігу та застосування КЕП для документів, які мають юридичну значущість. Такий підхід дозволяє уникнути надмірного ускладнення процесів, пов'язаного з обов'язковим використанням КЕП для всіх документів, і водночас забезпечити необхідний рівень довіри там, де це потрібно.

Рисунок 2.7 - Дворівнева модель електронного підпису та рівнів довіри

Перший рівень моделі - внутрішній електронний підпис - реалізується засобами корпоративного середовища, зокрема через механізми підтвердження дій користувачів у Microsoft Teams або за допомогою одноразових кодів, що надсилаються на корпоративну електронну пошту. У цьому випадку підпис не є криптографічним у класичному розумінні, а формується як сукупність факторів: автентифікація користувача, підтвердження дій та фіксація цієї дії в журналі системи. Такий підхід дозволяє забезпечити ідентифікацію особи, яка виконала дію, та невідомість у межах внутрішнього середовища університету. Другий рівень - використання КЕП - передбачає інтеграцію із зовнішніми сервісами електронного підпису або використання відповідних програмних засобів для накладання підпису. Цей рівень застосовується для документів, що мають правові наслідки, і забезпечує їх юридичну силу відповідно до чинного законодавства.

Рисунок 2.8 - Рольова модель доступу до документів у системі

На основі дворівневої моделі підпису формується модель довіри, яка визначає відповідність між типом документа, рівнем його значущості та способом підтвердження. У цій моделі внутрішній підпис забезпечує базовий рівень довіри, достатній для оперативних управлінських процесів, тоді як КЕП відповідає найвищому рівню довіри, необхідному для зовнішньої взаємодії та юридично значимих дій. Таким чином, система реалізує адаптивний підхід до довіри, де рівень захисту визначається не універсально, а залежно від контексту використання документа. Це дозволяє оптимізувати ресурси та підвищити ефективність документообігу без втрати надійності.

Модель доступу в системі базується на принципах рольового керування доступом (RBAC) і передбачає чітке розмежування прав користувачів відповідно до їх функціональних обов'язків. Кожна роль - викладач, завідувач кафедри, декан, представник адміністрації, працівник канцелярії або адміністратор - має визначений набір дозволених дій: створення, перегляд, редагування, погодження, підписання, реєстрація або архівування документів. Доступ до інформації обмежується як на рівні робочих просторів (Teams), так і на рівні бібліотек і окремих документів (SharePoint), що забезпечує захист конфіденційних даних і реалізацію принципу мінімально необхідних прав. Додатково можуть застосовуватися механізми автентифікації, зокрема багатфакторна автентифікація або підтвердження через корпоративну пошту, що підвищує безпеку доступу до системи.

Рисунок 2.9 - Модель аудиту дій користувачів та контролю версій документа

Невід'ємною складовою запропонованої моделі є модель аудиту, яка забезпечує фіксацію всіх дій користувачів у системі та формує так званий audit trail.

Усі операції з документами - створення, редагування, погодження, підписання, зміна статусу, перегляд - записуються із зазначенням часу, користувача та суті дій. Це дозволяє відстежувати повний життєвий цикл документа, здійснювати контроль виконання процесів і, за потреби, проводити аналіз або розслідування інцидентів. Додатковим елементом є контроль версій документів, який забезпечує збереження історії змін і можливість відновлення попередніх станів документа. У сукупності модель аудиту підвищує прозорість функціонування системи, зміцнює довіру до електронного документообігу та виступає ключовим елементом внутрішнього електронного підпису.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ СИСТЕМИ ВНУТРІШНЬОГО ЕЛЕКТРОННОГО ПІДПISУ

3.1. Реалізація структури документів та робочих просторів системи електронного документообігу

Реалізація структури документів та робочих просторів інформаційної системи управління внутрішніми документами університету базується на інтегрованому використанні хмарного середовища Microsoft 365, у межах якого забезпечується узгоджена взаємодія сервісів Microsoft Teams, SharePoint та Power Automate. Практична реалізація передбачає відображення логіки організаційної структури документообігу університету у вигляді цифрових робочих просторів, сховищ даних і механізмів обробки документів.

На верхньому рівні системи створено окремі робочі простори у Microsoft Teams, які відповідають основним типам документів. Зокрема, формуються команди «Накази», «Заяви», «Службові записки», «Табелі обліку часу» та інші, залежно від потреб установи. Такий підхід дозволяє забезпечити логічну структуру інформації, розмежування доступу користувачів і організацію взаємодії між учасниками процесу документообігу. Кожна команда містить канали, що можуть відповідати окремим підрозділам або категоріям документів, а також інтегровані вкладки для доступу до файлів, списків і додаткових сервісів.

Процес створення команди «Табелі обліку часу» в середовищі Microsoft Teams доцільно розглядати як один із базових етапів практичної реалізації системи електронного документообігу, оскільки саме через команди формується структура доступу користувачів та організація роботи з відповідним типом документів. Створення такої команди здійснюється засобами клієнтського застосунку Microsoft Teams або через вебінтерфейс і передбачає послідовність дій, спрямованих на формування окремого робочого простору для обробки табелів обліку часу (рис. 3.1).

Рисунок 3.1 - Створення нової команди в Microsoft Teams

На першому етапі користувач із відповідними правами (як правило, адміністратор або відповідальна особа підрозділу) ініціює створення нової команди шляхом вибору опції «Створити команду» (рис. 3.1). У процесі створення задається назва команди - «Табелі обліку часу», а також її опис, у якому доцільно вказати призначення даного робочого простору, наприклад: «Обробка, погодження та зберігання табелів обліку робочого часу працівників університету» (рис. 3.2). На цьому ж етапі (рис. 3.3) визначається тип команди: приватна (доступ мають лише запрошені користувачі) або публічна (доступ відкрите для ширшого кола співробітників). У контексті документообігу доцільним є використання приватної команди для забезпечення контролю доступу до службової інформації.

Рисунок 3.2 - Створення нової команди (встановлення відомостей)

Після створення команди здійснюється формування її складу, тобто додавання користувачів відповідно до їх ролей у процесі обробки табелів. До складу команди можуть входити працівники підрозділів (як ініціатори створення табелів), керівники підрозділів (як погоджувачі), представники бухгалтерії або кадрової служби (як відповідальні за перевірку), а також адміністрація (як підписанти або контролюючі особи). Кожному користувачу призначається відповідна роль у Teams (власник або учасник), що визначає рівень його повноважень у межах команди.

Рисунок 3.3 - Визначення типу команди

Наступним кроком є організація внутрішньої структури команди шляхом створення каналів. Доцільно передбачити канали за підрозділами або за періодами, наприклад «Січень 2026», «Лютий 2026» або «Кафедра ІТ», «Бухгалтерія», що дозволяє структуровано організувати обговорення та обробку документів. У кожному каналі автоматично створюється вкладка «Файли», яка пов'язана з відповідною бібліотекою документів у SharePoint. Таким чином забезпечується збереження табелів у централізованому сховищі з можливістю подальшого керування доступом, версіями та метаданими.

Рисунок 3.4 - Структура команди

Додатково в команді можуть бути налаштовані вкладки для доступу до пов'язаних ресурсів, зокрема списків SharePoint, які використовуються як журнал обліку табелів, або форм введення даних. Це дозволяє інтегрувати всі елементи процесу - від створення документа до його погодження та архівування - у єдиному робочому середовищі.

Таким чином, створення команди «Табелі обліку часу» в Microsoft Teams є не лише технічною операцією, але й важливим етапом побудови структурованого цифрового середовища документообігу, у якому поєднуються функції комунікації, зберігання документів та організації бізнес-процесів.

Наступним рівнем реалізації є налаштування середовища SharePoint, яке виступає основним сховищем документів і метаданих. Для кожного типу документів створюються відповідні бібліотеки документів, у яких визначається структура даних через набір полів. Наприклад, для бібліотеки наказів доцільно передбачити поля «номер документа», «дата», «тип», «підрозділ», «статус», «тип підпису», для заяв - «автор», «дата подання», «категорія», «етап погодження», для службових записок - «ініціатор», «адресат», «тема», «статус», для табелів - «період», «підрозділ», «відповідальна особа». Використання метаданих дозволяє забезпечити ефективний пошук, фільтрацію та сортування документів, що є більш доцільним, ніж використання виключно ієрархічної структури папок.

Рисунок 3.5 - Структура команди в SharePoint

Разом із тим у практичній реалізації може використовуватися комбінований підхід до організації зберігання документів, який поєднує використання метаданих і папок. Як приклад, для кожного типу документів може бути створена структура папок за роками, підрозділами або категоріями, наприклад «Накази / 2026 / Кадрові / Погоджені» або «Заяви / 2026 / Відпустки / На розгляді». Така організація полегшує орієнтацію користувачів у системі, водночас основні операції управління документами виконуються на основі метаданих.

Важливою складовою реалізації є формування моделі журналу проходження документів, який виконує функцію операційної бази даних системи. У межах платформи Microsoft 365 такий журнал може бути реалізований у вигляді списку SharePoint, де зберігається інформація про кожен документ: унікальний ідентифікатор, тип документа, автор, дата створення, поточний статус, історія переходів між етапами, відповідальні особи, спосіб підписання, а також посилання на відповідний файл у бібліотеці. Журнал забезпечує відстеження повного життєвого циклу документа та може використовуватися для моніторингу, формування звітності та аналітики.

Рисунок 3.6 - Створення списку (журналу)

Реалізація маршрутів погодження документів здійснюється за допомогою сервісу Power Automate, який виступає як механізм автоматизації бізнес-процесів. У системі налаштовуються сценарії, що визначають послідовність або паралельність погодження документів залежно від їх типу. Кожен маршрут може включати етапи перевірки, погодження, повернення на доопрацювання, відхилення, підписання та передачі до архіву. При цьому система автоматично змінює статус документа, фіксує відповідні події в журналі та надсилає сповіщення користувачам.

Налаштування прав доступу до ресурсів системи здійснюється на основі ролівої моделі. Користувачі отримують доступ до команд Microsoft Teams, бібліотек SharePoint і окремих документів відповідно до їх функціональних обов'язків. Викладачі та співробітники мають право створення та перегляду власних документів, завідувачі кафедр і декани - право погодження та коментування, представники адміністрації - право підписання, працівники канцелярії - право реєстрації та архівування, адміністратори - повний доступ до налаштування системи. Такий підхід забезпечує дотримання принципу мінімально необхідних прав доступу та підвищує рівень інформаційної безпеки.

Для налаштувань прав доступу папки треба перейти до SharePoint, відкликати права доступу та налаштувати права доступу персонально.

Рисунок 3.7 - Налаштування прав доступу

Окремої уваги заслуговує можливість реалізації додаткового мобільного інтерфейсу системи. У наступному параграфі доцільно розглянути використання технологій ASP.NET Core для створення спрощеного вебзастосунок, адаптованого до мобільних пристроїв. Такий застосунок може виконувати функції отримання сповіщень про нові документи або необхідність виконання дій, а також реалізовувати механізм підтвердження внутрішнього електронного підпису через авторизацію користувача та введення одноразового коду, надісланого на корпоративну електронну пошту. Водночас слід зазначити, що для накладання кваліфікованого електронного підпису такий застосунок повинен взаємодіяти із зовнішніми сервісами підпису, оскільки реалізація повноцінного КЕП виходить за межі можливостей внутрішнього вебзастосунку.

Таким чином, практична реалізація структури документів та робочих просторів системи забезпечує відповідність між організаційною структурою університету та цифровим середовищем, дозволяє ефективно керувати життєвим циклом документів і створює основу для подальшого розвитку системи електронного документообігу.

3.2. Реалізація бізнес-процесів погодження за допомогою Power Automate

На першому етапі - створення документа - ініціатор формує документ у відповідній бібліотеці SharePoint або через інтерфейс Microsoft Teams. При цьому заповнюються основні метадані документа, такі як тип, автор, підрозділ, період, а також початковий статус, наприклад «створено». Саме факт створення або зміни документа в бібліотеці виступає тригером для запуску автоматизованого процесу в Power Automate. Такий підхід дозволяє реалізувати подієво-орієнтовану модель, у якій бізнес-процеси створюються автоматично при виникненні відповідних подій у системі. Другий етап - погодження документа - реалізується як послідовність або паралельний набір завдань для відповідних користувачів. У Power Automate використовується механізм погодження (approval), який дозволяє надсилати запити на погодження визначеним особам, відстежувати їх відповіді та автоматично змінювати статус документа залежно від результату. Наприклад, для табелів обліку часу погодження може здійснюватися керівником підрозділу та представником бухгалтерії. У разі позитивного рішення документ переходить на наступний етап, у разі відмови - повертається ініціатору на доопрацювання із відповідним коментарем. Така організація процесу забезпечує гнучкість і відповідність реальним управлінським процедурам університету.

Третій етап - підпис документа - передбачає застосування дворівневої моделі електронного підпису. У випадку внутрішніх документів використовується внутрішній корпоративний підпис, який реалізується через підтвердження дії користувача в системі (наприклад, через механізми погодження або одноразовий код). Для документів, що мають юридичну значущість, передбачається інтеграція із зовнішніми сервісами накладання кваліфікованого електронного підпису. У Power Automate цей етап може бути реалізований як окремий блок процесу, який або завершує автоматизований хмарний потік, або ініціює зовнішню процедуру підписання з подальшим поверненням результату в систему.

Завершальним етапом є архівування документа, яке передбачає зміну його статусу на «архівовано», переміщення або логічне віднесення до архівної категорії, а також фіксацію всіх результатів виконання процесу. На цьому етапі інформація про документ остаточно записується до журналу проходження документів, що дозволяє забезпечити повну простежуваність його життєвого циклу. Архівування може супроводжуватися додатковими діями, такими як обмеження доступу до документа або його переведення в режим лише для читання.

Обґрунтування вибору саме такої структури автоматизованого хмарного потіку полягає у її універсальності, відповідності організаційній логіці документообігу та можливості адаптації до різних типів документів. Поділ процесу на чотири основні етапи дозволяє чітко визначити відповідальність учасників, забезпечити контроль переходів між станами документа та інтегрувати різні механізми підтвердження (погодження та підпис). Використання Power Automate як процесного рушія забезпечує централізоване управління бізнес-логікою, можливість швидкого внесення змін до маршрутів погодження та інтеграцію з іншими компонентами системи, що в сукупності підвищує ефективність функціонування електронного документообігу університету.

Практична реалізація бізнес-процесів погодження документів у системі із використанням Power Automate передбачає застосування типових інструментів автоматизації, зокрема механізмів погоджень (approvals), email-сповіщень та умовної логіки. Розглянемо приклад реалізації процесу погодження табеля обліку часу, який ілюструє використання зазначених можливостей у комплексі.

На етапі погодження документа використовується вбудований механізм погоджень (approvals), який дозволяє формувати запити на погодження для визначених користувачів. Після створення або оновлення документа в бібліотеці SharePoint автоматично формується запит на погодження, який надсилається, наприклад, керівнику підрозділу. У запиті міститься основна інформація про документ (назва, автор, період, посилання на файл), а також кнопки дії «Погодити» або «Відхилити». У разі складнішого сценарію погодження може бути реалізовано послідовний ланцюг, де після погодження керівником підрозділу документ автоматично передається на погодження до бухгалтерії. Таким чином забезпечується контрольований рух документа між відповідальними особами.

Паралельно з механізмом погодження активно використовуються email-сповіщення, які забезпечують інформування учасників процесу про необхідність виконання дій або зміну стану документа. Наприклад, після створення документа ініціатор отримує підтвердження про запуск процесу, керівник - повідомлення про необхідність погодження, а у випадку відмови ініціатору надсилається лист із зазначенням причини та рекомендаціями щодо доопрацювання. Додатково повідомлення можуть надсилатися після завершення кожного етапу, що дозволяє забезпечити прозорість процесу та своєчасне реагування користувачів. У системі також можливе дублювання сповіщень через інтерфейс Microsoft Teams, що підвищує ефективність комунікації.

Важливим елементом реалізації автоматизованого хмарного потоку є використання умовної логіки (conditions), яка дозволяє адаптувати процес залежно від властивостей документа або результатів попередніх етапів. Наприклад, у процесі погодження може бути реалізована умова: якщо статус погодження дорівнює «Відхилено», то документ повертається ініціатору, його статус змінюється на «На доопрацюванні», а процес завершується або перезапускається після внесення змін. Інша умова може визначати тип підпису: якщо поле «Тип підпису» має значення «внутрішній», процес завершується після погодження; якщо ж значення - «КЕП», документ передається на наступний етап, пов'язаний із зовнішнім підписом. Також можуть використовуватися умови для визначення маршруту погодження, наприклад залежно від підрозділу або категорії документа.

Узагальнюючи, використання погоджень (approvals) забезпечує формалізований механізм прийняття рішень, email-сповіщення - оперативну комунікацію між учасниками процесу, а умовна логіка - гнучкість і адаптивність автоматизованих хмарних потоків. Поєднання цих інструментів дозволяє реалізувати ефективну систему автоматизованого погодження документів, яка відповідає вимогам сучасного електронного документообігу та враховує специфіку організації роботи університету.

Для практичної реалізації спочатку створюється автоматизований потік на основі тригера SharePoint (рис. 3.8 та рис. 3.9); далі додається дія «Start and wait for an approval» (рис. 3.10) із зазначенням погоджувачів і реквізитів документа; після цього налаштовується блок умов для аналізу результату; на завершення додаються операції оновлення статусу документа, email-сповіщення та, за потреби, запуск наступного етапу погодження або підписання. Така послідовність дій дозволяє побудувати вбудований механізм approvals, який інтегрується з документами

SharePoint і забезпечує автоматизоване прийняття рішень щодо їх подальшого руху.

Рисунок 3.8 - Створення автоматизованого хмарного потоку (циклу)

Рисунок 3.9 - Налаштування автоматизованого хмарного потоку (циклу)

Рисунок 3.10 Створення елементу «Start and wait for an approval» для першого погодження

У лістингу 3.1 наведено код для першого погодження (реальна пошта змінена).

Лістинг 3.1

```
Код для Start and wait for an approval { "type": "OpenApiConnectionWebhook" "inputs": { "parameters": { "approvalType": "Basic" "WebhookApprovalCreationInput/title": "Відділ кадрів" "WebhookApprovalCreationInput/assignedTo": "ЧЧЧЧЧЧ@luguniv.edu.ua" "WebhookApprovalCreationInput/enableNotifications": true "WebhookApprovalCreationInput/enableReassignment": true "host": "apild": "/providers/Microsoft.PowerApps/apis/shared_approvals" "connection": "shared_approvals" "operationId": "StartAndWaitForAnApproval" "runAfter": {} }
```

Після всіх дій отримуємо автоматизований потік Power Automate для нашого процесу (включає два погодження: відділ кадрів, бухгалтерія, зміна статусу документа, інформування у разі відхилення). Повний код в додатку А.

Рисунок 3.11. - Автоматизований потік для «Табель обліку часу»

3.3. Розробка ASP.NET застосунку для сповіщення про зміни електронного документа

Розробка ASP.NET застосунку для сповіщення про зміни електронного документа є логічним розширенням функціональних можливостей системи електронного документообігу університету, побудованої на базі Microsoft 365. Незважаючи на те, що базові механізми сповіщення можуть бути реалізовані засобами Power Automate, Outlook і Teams, використання окремого вебзастосунку дозволяє створити єдину точку взаємодії користувача із сервісом повідомлень, забезпечити більш гнучке налаштування логіки інформування та реалізувати мобільно-орієнтований доступ до інформації про стан документа. У контексті розроблюваної системи такий застосунок не замінює основне середовище документообігу, а виконує роль допоміжного програмного модуля, орієнтованого на оперативне інформування користувача про зміни життєвого циклу електронного документа.

Актуальність створення такого застосунку зумовлена тим, що в межах внутрішнього документообігу користувачу важливо не лише мати доступ до документа, а й своєчасно отримувати інформацію про події, пов'язані з ним. До таких подій належать створення нового документа, передача його на погодження, повернення на доопрацювання, відхилення, підписання, реєстрація або архівування. Якщо користувач змушений щоразу самостійно перевіряти стан документа в Teams або SharePoint, це знижує оперативність роботи і створює додаткове навантаження на учасників процесу. Тому доцільним є впровадження окремого прикладного рішення, яке відображає актуальний стан документа, формує повідомлення та дозволяє користувачеві швидко реагувати на зміни.

У межах цієї кваліфікаційної роботи ASP.NET застосунок є спрощеним веборієнтованим клієнт, що взаємодіє з основною системою електронного документообігу через дані, сформовані у SharePoint, журналі проходження документів або сервісах автоматизації. Основне призначення застосунку полягає в отриманні інформації про зміни статусів документів та відображенні її у зручному для користувача вигляді. У найпростішому варіанті він може містити список повідомлень, сторінку перегляду вибраного документа, позначення нових подій, а також засоби підтвердження внутрішнього підпису через корпоративну автентифікацію або одноразовий код. Така архітектура є доцільною, оскільки дозволяє винести функцію сповіщення в окремий програмний модуль без порушення логіки основної системи.

З технічної точки зору вибір ASP.NET як платформи реалізації пояснюється кількома чинниками. По-перше, технологія дозволяє створювати вебзастосунки, адаптовані до мобільних пристроїв, що важливо для оперативного доступу працівників до повідомлень поза стаціонарним робочим місцем. По-друге, ASP.NET забезпечує зручні засоби побудови серверної логіки, автентифікації, роботи з базами даних і реалізації вебінтерфейсів. По-третє, використання цієї технології є доцільним у навчальному контексті, оскільки вона добре узгоджується з напрямом підготовки фахівців з програмної інженерії та дозволяє поєднати хмарну платформу Microsoft 365 із власним програмним рішенням. Таким чином, ASP.NET застосунок у цій роботі розглядається як інструмент підвищення зручності, оперативності та персоналізації доступу до інформації про електронні документи.

Функціонально розроблюваний застосунок повинен забезпечувати перегляд переліку документів, які стосуються конкретного користувача, відображення їх поточного статусу, історії останніх змін і пов'язаних повідомлень. Окрему увагу приділено механізму сповіщень, який може працювати за подієвим принципом: зміна статусу документа в основній системі призводить до створення нового повідомлення в застосунку. Наприклад, після погодження табеля обліку часу користувач отримує повідомлення про те, що документ передано на підпис, а після відхилення - повідомлення з відповідним коментарем і необхідністю доопрацювання. Такий підхід дозволив створити єдиний інформаційний простір, де користувач взаємодіє не з усією складністю системи документообігу, а лише з релевантними для нього подіями.

Отже, розробка ASP.NET застосунку для сповіщення про зміни електронного документа є обґрунтованим напрямом практичної реалізації, який доповнює основну систему документообігу та підвищує її зручність для кінцевого користувача.

Для реалізації модуля сповіщення про зміни електронного документа обрано архітектуру вебзастосунку на платформі ASP.NET Core, оскільки вона підтримує побудову сучасних вебрішень на основі патерну Model-View-Controller, має вбудовані засоби автентифікації й авторизації та придатна для створення як звичайних вебінтерфейсів, так і сервісів реального часу. Для задачі інформування користувачів особливо важливим є те, що ASP.NET Core SignalR призначений саме для додавання функціональності реального часу, коли сервер може миттєво надсилати оновлення клієнтам без періодичного ручного перезавантаження сторінки. Це дозволяє побудувати модуль, який не лише зберігає повідомлення, а й оперативно доставляє інформацію про зміну статусу документа відповідальному користувачу [24].

З архітектурної точки зору доцільно застосувати багатокомпонентну модель, у якій застосунок поділяється на рівень представлення, рівень прикладної логіки та рівень даних. Рівень представлення реалізується через вебінтерфейс ASP.NET Core MVC, де користувач переглядає список сповіщень, стан своїх документів і деталі окремої події. Рівень прикладної логіки відповідає за отримання подій із зовнішньої системи документообігу, їх інтерпретацію, перевірку прав доступу, формування тексту повідомлень і керування статусом прочитання. Рівень даних забезпечує збереження інформації про користувачів, документи, повідомлення та історію змін. Такий підхід є логічним, оскільки MVC в ASP.NET

Core саме і орієнтований на розділення моделей даних, контролерів із прикладною логікою та представлень для взаємодії з користувачем [25]. Базовими сутностями визначено користувача, документ, повідомлення та журнал змін. Сутність користувача містить ідентифікатор, ім'я, електронну адресу, роль у системі та, за потреби, службову прив'язку до підрозділу. Сутність документа містить ідентифікатор документа, назву, тип, автора, поточний статус, тип підпису, дату створення та посилання на ресурс у SharePoint або Teams. Сутність повідомлення зберігає текст сповіщення, дату створення, тип події, ознаку прочитання та прив'язку до конкретного документа і користувача. Окремо доцільно передбачити сутність журналу змін, яка фіксує події життєвого циклу документа: створення, погодження, повернення, підписання, архівування. Така модель даних забезпечує і поточне інформування, і накопичення історії подій для подальшого аналізу. Оскільки ASP.NET Core Identity типовим чином працює з реляційним сховищем даних для зберігання відомостей про користувачів, така організація моделі є природною і технологічно обґрунтованою [26].

Структура компонентів модуля включає кілька логічно відокремлених частин. Першим компонентом виступає користувацький інтерфейс, що забезпечує авторизацію, перегляд списку повідомлень, відкриття картки документа та позначення повідомлення як прочитаного. Другим компонентом є контролери або серверні обробники запитів, які отримують звернення від клієнта та взаємодіють із сервісами прикладної логіки. Третім компонентом виступає сервіс сповіщень, який формує повідомлення на основі подій, отриманих із зовнішньої системи або з внутрішнього журналу. Четвертим компонентом визначено сервіс інтеграції, який отримує інформацію про зміни статусу документа з Power Automate, SharePoint або іншого проміжного джерела. П'ятим компонентом є SignalR-хаб, якщо в системі реалізується миттєва доставка повідомлень до браузера користувача. Окремо виділяється компонент доступу до даних, який виконує збереження і вибірку сутностей із бази даних. Така структура забезпечує модульність, спрощує супровід системи й дозволяє розвивати окремі частини незалежно одна від одної [20].

Програмну реалізацію цього модуля побудовано як вебзастосунок ASP.NET Core із серверною частиною та адаптивним вебінтерфейсом. На рівні запуску застосунку налаштована конфігурація, реєстрація сервісів, підключення до бази даних і middleware-компонентів. Офіційна документація ASP.NET Core вказує, що конфігурація застосунку формується через `WebApplication.CreateBuilder`, після чого до контейнера служб додаються залежності, а далі будується `pipeline` обробки HTTP-запитів. У межах модуля це означає реєстрацію сервісів автентифікації, авторизації, доступу до даних, SignalR і власного сервісу повідомлень. Застосунок повинен підтримувати безпечний доступ до персоналізованих повідомлень, тому механізми автентифікації й авторизації мають бути обов'язковою складовою реалізації, а доступ до сповіщень конкретного користувача - перевірятися на серверному рівні.

З практичної точки зору реалізація організована таким чином: подія про зміну документа формується у зовнішній системі документообігу, після чого або надсилається у вебзастосунок через API, або зчитується із журналу змін. Далі серверний сервіс створює запис повідомлення у базі даних і, якщо використовується SignalR, надсилає push-оновлення підключеному клієнту. Користувач, увійшовши в систему, бачить перелік актуальних подій, а у випадку відкритої сесії може отримати повідомлення в режимі реального часу. Саме такий сценарій відповідає типовому призначенню SignalR як бібліотеки для миттєвого надсилання вмісту від сервера до клієнтів. У результаті модуль сповіщення не дублює весь документообіг, а виконує вузькоспеціалізовану функцію оперативного інформування та підтримки реакції користувача на зміни стану документа [26].

Отже, архітектурне рішення для ASP.NET-модуля сповіщень визначено як окремий вебкомпонент, інтегрований із основною системою електронного документообігу. Його модель даних включає користувачів, документи, повідомлення та журнал змін, структура компонентів - інтерфейс, прикладну логіку, інтеграційний сервіс, модуль доступу до даних і механізм реального часу, а програмна реалізація - спиратися на стандартні можливості ASP.NET Core для побудови вебзастосунків, автентифікації, авторизації та обробки подій. Такий підхід є достатньо простим для бакалаврської роботи, але водночас демонструє сучасний рівень інтеграції власного програмного рішення з хмарною інфраструктурою Microsoft 365.

На рис. 3.12 представлено структуру коду проєкту

```
DocumentNotifier/  
├─ Controllers/  
│   └─ DocumentsController.cs  
│   └─ NotificationsController.cs  
├─ Data/  
│   └─ AppDbContext.cs  
├─ Hubs/  
│   └─ NotificationHub.cs  
├─ Models/  
│   └─ DocumentItem.cs  
│   └─ NotificationItem.cs  
│   └─ DocumentStatus.cs  
├─ Services/  
│   └─ NotificationService.cs  
├─ Views/  
│   └─ Documents/  
│       └─ Index.cshtml  
│       └─ Edit.cshtml  
│   └─ Notifications/  
│       └─ Index.cshtml  
├─ wwwroot/  
│   └─ js/  
│       └─ notifications.js  
├─ Program.cs  
└─ appsettings.json
```

Рисунок 3.11 - Структура проєкту ASP.NET

Основу системи становить модель даних, представлена класами `DocumentItem`, `NotificationItem` та `DocumentStatus`. Клас `DocumentItem` описує електронний документ і містить основні атрибути, зокрема назву, тип, автора, електронну адресу, статус та часові мітки

створення і змін. Клас `NotificationItem` використовується для збереження сповіщень, що генеруються при зміні стану документа, і містить інформацію про отримувача, текст повідомлення, прив'язку до документа, дату створення та ознаку прочитання. Перерахування `DocumentStatus` визначає допустимі стани документа, що дозволяє формалізувати його життєвий цикл і забезпечити узгодженість обробки в системі. Збереження та доступ до даних реалізовано за допомогою контексту `AppDbContext`, який базується на технології `Entity Framework Core`. У контексті визначено таблиці для документів і сповіщень, а також налаштовано зв'язок між ними через зовнішній ключ. Додатково передбачено збереження статусу документа у вигляді текстового значення, що полегшує аналіз даних і підвищує читабельність бази даних. Використання `EF Core` дозволяє працювати з даними на рівні об'єктної моделі без необхідності написання складних SQL-запитів.

Ключовим елементом модуля є сервіс `NotificationService`, який реалізує бізнес-логіку формування сповіщень. При зміні статусу документа відповідний контролер викликає метод `NotifyDocumentStatusChangedAsync`, у якому створюється новий об'єкт `NotificationItem`, що зберігається в базі даних. Після цього сервіс використовує механізм `SignalR` для надсилання повідомлення користувачу в режимі реального часу. Таким чином забезпечується одночасне виконання двох функцій: фіксація події в системі та оперативне інформування користувача.

Передача повідомлень у реальному часі реалізована за допомогою компонента `NotificationHub`, який виступає проміжною ланкою між сервером і клієнтом. Користувач при підключенні до застосунку додається до відповідної групи, ідентифікованої за його електронною адресою. Це дозволяє адресно надсилати повідомлення лише тим користувачам, яких вони стосуються. У момент створення нового сповіщення сервер через `SignalR` надсилає дані до клієнтського інтерфейсу, де вони одразу відображаються без необхідності оновлення сторінки.

Контролери `DocumentsController` та `NotificationsController` забезпечують взаємодію між користувачем і системою. `DocumentsController` відповідає за створення, редагування та відображення документів, а також ініціює процес формування сповіщень у разі зміни статусу. `NotificationsController` реалізує функції перегляду списку сповіщень та позначення їх як прочитаних. Такий розподіл відповідальності відповідає принципам `MVC` і дозволяє чітко відокремити логіку роботи з документами від логіки обробки повідомлень.

На рівні представлення використано `Razor`-сторінки, які забезпечують відображення списку документів та сповіщень у вигляді таблиць. Для підтримки роботи в реальному часі на клієнтській стороні застосовується `JavaScript`-код, який встановлює з'єднання з `SignalR`-хабом і обробляє події отримання нових повідомлень. При надходженні повідомлення воно динамічно додається до сторінки, що забезпечує миттєве інформування користувача про зміни в системі.

Конфігурація застосунку виконується у файлі `Program.cs`, де налаштовуються служби, підключення до бази даних, маршрутизація та компоненти `SignalR`. У процесі запуску також забезпечується створення бази даних, якщо вона відсутня, що спрощує розгортання застосунку. Використання механізму впровадження залежностей дозволяє гнучко керувати компонентами системи та забезпечує можливість їх подальшого розширення. Повний код наведено в додатку Б.

3.4. Тестування системи електронного документообігу та оцінка її ефективності

Тестування розробленої системи електронного документообігу є завершальним етапом практичної частини роботи та спрямоване на перевірку коректності функціонування її основних компонентів, відповідності поставленим вимогам, а також оцінку ефективності впроваджених рішень. У межах дослідження тестування охоплює як окремі модулі системи (керування документами, погодження, підпис, сповіщення), так і їх взаємодію в рамках єдиного інформаційного середовища.

На першому етапі здійснено функціональне тестування системи, яке передбачає перевірку реалізації основних сценаріїв роботи користувачів. Зокрема, перевіряється можливість створення документа в середовищі `Microsoft Teams` та `SharePoint`, коректність збереження метаданих, запуск автоматизованого робочого процесу погодження в `Power Automate`, а також правильність обробки результатів погодження. Окремо тестується механізм дворівневого підпису, де перевіряється як внутрішнє підтвердження дій користувача, так і передача документа на зовнішній етап підписання. Для кожного сценарію визначаються очікувані результати, які порівнюються з фактичними, що дозволяє виявити можливі помилки або невідповідності.

Наступним етапом було тестування інтеграції компонентів системи. У цьому випадку перевірялась взаємодія між `SharePoint` як сховищем даних, `Power Automate` як механізмом реалізації бізнес-процесів та `ASP.NET`-застосунком як модулем сповіщення. Особлива увага приділяється передачі подій між компонентами, зокрема коректності запуску `workflow` при створенні або зміні документа, передачі інформації про статус документа до журналу та формуванню сповіщень. Результати тестування показують, що інтеграція компонентів забезпечує узгоджену роботу системи та підтримує повний життєвий цикл документа.

Для оцінки ефективності системи проводився аналіз часових характеристик виконання основних операцій. Порівнювались традиційний паперовий або частково електронний процес документообігу із запропонованим автоматизованим рішенням. Встановлено, що автоматизація дозволяє значно скоротити час обробки документів за рахунок усунення ручних операцій передачі, пошуку та контролю стану. Наприклад, час погодження документа скорочується завдяки автоматичному маршрутизації та сповіщенням користувачів, а ризик втрати документа практично усувається через централізоване зберігання в хмарному середовищі.

Крім часових показників, оцінюється якість функціонування системи за такими критеріями, як зручність використання, прозорість процесів, надійність і безпека. Зручність використання забезпечується інтеграцією з `Microsoft Teams`, що є знайомим середовищем для користувачів. Прозорість досягається завдяки наявності журналу проходження документів, який дозволяє відстежувати всі етапи їх обробки. Надійність системи забезпечується використанням хмарної інфраструктури, що гарантує доступність даних, а безпека - через реалізацію ролівої моделі доступу та журналювання дій користувачів.

Додатково проведено тестування модуля сповіщення, реалізованого на `ASP.NET Core`. Перевірено коректність формування повідомлень при зміні статусу документа, їх збереження в базі даних та доставку користувачам у режимі реального часу за допомогою `SignalR`. Результати тестування показали, що користувачі отримують актуальну інформацію без затримок, що підвищує оперативність реагування на події в системі. Також перевірено механізм позначення повідомлень як прочитаних і відображення їх у вебінтерфейсі.

Таблиця 3.1 - Порівняння часу виконання основних операцій документообігу

No	Операція	Традиційний процес (хв)		Автоматизована система (хв)		Скорочення часу (%)
1	Створення документа	15	10	33%		
2	Передача на погодження	30	2	93%		
3	Погодження документа	120	30	75%		
4	Повторне доопрацювання	60	20	67%		

5	Підпис документа	45	10	78%
6	Пошук документа	20	2	90%

Як бачимо з табл. 3.1 автоматизація дозволяє скоротити час виконання операцій у середньому на 70-90%.

Узагальнюючи результати тестування, можна зробити висновок, що розроблена система електронного документообігу відповідає поставленим вимогам, забезпечує автоматизацію ключових процесів і підвищує ефективність роботи з документами. Вона дозволяє скоротити час обробки, підвищити контроль за виконанням завдань, забезпечити прозорість і зменшити вплив людського фактору. Отримані результати підтверджують доцільність використання хмарних сервісів Microsoft 365 у поєднанні з власними програмними рішеннями для побудови сучасних систем електронного документообігу в освітніх установах.

ЗАГАЛЬНІ ВИСНОВКИ

Актуальність теми дослідження зумовлена необхідністю цифровізації управлінських процесів в освітніх установах, зокрема переходом від фрагментарних або паперових форм документообігу до інтегрованих електронних систем, що забезпечують оперативність, прозорість і контроль виконання. Метою роботи було розроблення інформаційної системи управління внутрішніми документами університету на основі хмарних сервісів Microsoft 365 із впровадженням автоматизованих процесів погодження, дворівневої моделі підпису та засобів сповіщення користувачів. У ході виконання роботи було проведено аналіз сучасних підходів до організації електронного документообігу, розглянуто нормативно-правові аспекти використання електронних документів і підписів, а також досліджено можливості застосування хмарних технологій у діяльності університетів. Встановлено, що повне використання кваліфікованого електронного підпису для всіх типів документів є недоцільним через організаційні та технічні обмеження, що обґрунтовує необхідність впровадження дворівневої моделі, яка поєднує внутрішній корпоративний підпис і КЕП як зовнішній інструмент. Також показано доцільність використання платформи Microsoft 365 як інтегрованого середовища для реалізації документообігу.

У результаті проєктування системи було сформульовано функціональні та нефункціональні вимоги, визначено типи документів, ролі користувачів і маршрути погодження, а також розроблено логічну та структурну архітектуру системи. Розроблено модель документообігу, що включає етапи створення, погодження, підпису та архівування документів, а також модель доступу на основі ролей користувачів. Окремо сформовано дворівневу модель електронного підпису, модель довіри та модель аудиту, що забезпечують контроль дій користувачів і простежуваність життєвого циклу документа.

У практичній частині роботи реалізовано структуру системи на базі Microsoft Teams та SharePoint, налаштовано бібліотеки документів, метадані, права доступу та журнал проходження документів. За допомогою Power Automate розроблено автоматизовані робочі процеси погодження документів із використанням механізмів погодження, умовної логіки та сповіщень. Реалізовані процеси забезпечують автоматичний запуск при створенні документа, контроль проходження етапів та зміну статусів відповідно до результатів прийнятих рішень.

Окремим результатом роботи є розроблення вебзастосунку на базі ASP.NET Core, який виконує функцію сповіщення користувачів про зміни стану документів. У застосунку реалізовано модель даних для збереження документів і повідомлень, серверну логіку формування сповіщень, а також механізм доставки повідомлень у режимі реального часу із використанням SignalR. Це дозволяє забезпечити оперативне інформування користувачів і підвищити ефективність їх взаємодії з системою електронного документообігу.

Проведене тестування системи підтвердило коректність реалізації основних функціональних сценаріїв, зокрема створення документів, запуску процесів погодження, зміни статусів і формування сповіщень. Аналіз ефективності показав значне скорочення часу виконання операцій, підвищення прозорості процесів, зниження ризику втрати документів і покращення контролю виконання. Результати тестування також підтвердили стабільність роботи системи та коректність інтеграції її компонентів.

Узагальнюючи отримані результати, можна зробити висновок, що поставлена мета роботи досягнута, а всі визначені завдання виконані. Розроблена система електронного документообігу демонструє ефективність використання хмарних сервісів Microsoft 365 у поєднанні з власними програмними рішеннями та може бути використана як основа для впровадження сучасних цифрових рішень в управлінській діяльності освітніх установ.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- Деякі питання документування управлінської діяльності. Офіційний вебпортал парламенту України. URL: <https://zakon.rada.gov.ua/laws/show/55-2018-n#Text>
- ДСТУ 4163:2020 ;Уніфікована система організаційно-розпорядчої документації. Вимоги до оформлення документів. Нормативні документи від VOB. URL: <https://document.vobu.ua/doc/6310>
- Останчук Ю. Що таке SaaS сервіс: переваги, недоліки та коли він потрібен для вашого бізнесу | Asabix. URL: <https://asabix.com.ua/what-is-saas-and-why-might-your-business-need-it/>
- Положення про електронний документообіг в ТНПУ. Офіційний сайт Тернопільського національного педагогічного університету імені Володимира Гнатюка. URL: https://tnpu.edu.ua/about/public_inform/upload/2025/Polozhennja_pro_E-dok.pdf
- Про електронні документи та електронний документообіг. Офіційний вебпортал парламенту України. URL: <https://zakon.rada.gov.ua/laws/show/851-15>
- Про електронну ідентифікацію та електронні довірчі послуги. Офіційний вебпортал парламенту України. URL: <https://zakon.rada.gov.ua/laws/show/2155-19#Text>
- Про затвердження Порядку роботи з електронними документами у діловодстві та їх підготовки до передавання на архівне зберігання. Офіційний вебпортал парламенту України. URL: <https://zakon.rada.gov.ua/laws/show/z1421-14/page>
- Caeldries F., Hammer M., Champy J. Reengineering the Corporation: A Manifesto for Business Revolution. The Academy of Management Review. 1994. Vol. 19, no. 3. P. 595. URL: <https://doi.org/10.2307/258943>
- Contributors to Wikimedia projects. Microsoft 365 - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Microsoft_365
- Contributors to Wikimedia projects. Microsoft Power Automate - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Microsoft_Power_Automate
- Contributors to Wikimedia projects. Microsoft Teams - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Microsoft_Teams
- Contributors to Wikimedia projects. SharePoint - Wikipedia. Wikipedia, the free encyclopedia. URL: <https://en.wikipedia.org/wiki/SharePoint>

13. Document Preparation for Electronic Document Management System // M. A. M. Radzi et al. International Journal of Academic Research in Business and Social Sciences. 2018. Vol. 8, no. 9. URL: <https://doi.org/10.6007/ijarbs/v8-i9/4583>
14. Hartland N., Chen E., Reynolds R. Capturing the Cloud: Towards SharePoint Transfer at UK Parliament. International Journal of Digital Curation. 2024. Vol. 18, no. 1. P. 6. URL: <https://doi.org/10.2218/mzkv7005>
15. Ilag B. N. Microsoft Teams Troubleshooting. Introducing Microsoft Teams. Berkeley, CA, 2018. P. 237-313. URL: https://doi.org/10.1007/978-1-4842-3567-6_6
16. Introduction to Identity on ASP.NET Core. Microsoft Learn: Build with answers in reach. URL: <https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-10.0>
17. Mayer C., Simons G. Microsoft Teams. Atla Summary of Proceedings. 2022. P. 260-268. URL: <https://doi.org/10.31046/proceedings.2022.3115>
18. McFarlan F. W., Nolan R. L. Curriculum recommendations for graduate professional programs in information systems. Communications of the ACM. 1973. Vol. 16, no. 7. P. 439-441. URL: <https://doi.org/10.1145/362280.362296>
19. Mercurio R., Merrill B. Power Automate. Beginning Microsoft 365 Collaboration Apps. Berkeley, CA, 2021. P. 287-320. URL: https://doi.org/10.1007/978-1-4842-6936-7_11
20. Microsoft Sharepoint and storage addons for education - M365 Education. Microsoft Learn: Build with answers in reach. URL: <https://learn.microsoft.com/en-us/microsoft-365/education/guide/1-addons/addons-sharepoint-storage>
21. Mishra G. Deep Dive into Power Automate. Berkeley, CA: Apress, 2023. URL: <https://doi.org/10.1007/978-1-4842-9732-2>
22. Muminova S. S. BLOCKCHAIN TECHNOLOGY AS A DOCUMENT MANAGEMENT AND ELECTRONIC DOCUMENT MANAGEMENT SYSTEM TOOL. American Journal of Applied Science and Technology. 2023. Vol. 03, no. 05. P. 65-69. URL: <https://doi.org/10.37547/ajast/volume03issue05-12>
23. Nolan R. L. Management Accounting and Control of Data Processing. National Association of Accountants, 1977.
24. Overview of ASP.NET Core Authentication. Microsoft Learn: Build with answers in reach. URL: <https://learn.microsoft.com/en-us/aspnet/core/security/authentication/?view=aspnetcore-10.0>
25. Overview of ASP.NET Core MVC. Microsoft Learn: Build with answers in reach. URL: <https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-10.0>
26. Overview of ASP.NET Core SignalR. Microsoft Learn: Build with answers in reach. URL: <https://learn.microsoft.com/en-us/aspnet/core/signalr/introduction?view=aspnetcore-10.0>
27. Ovsienko A. S. IMPLEMENTATION OF ELECTRONIC DOCUMENT MANAGEMENT IN THE ENTERPRISE MANAGEMENT SYSTEM. Economy Management Business. 2024. T. 44, No. 1. URL: <https://doi.org/10.31673/2415-8089.2024.010013>
28. Saxby S. Electronic document management. Computer Law & Security Review. 2005. Vol. 21, no. 4. P. 355. URL: <https://doi.org/10.1016/j.clsr.2005.06.007>
29. Wilson J. Electronic document management. Computer Law & Security Review. 1997. Vol. 13, no. 2. P. 124-125. URL: [https://doi.org/10.1016/s0267-3649\(97\)89753-1](https://doi.org/10.1016/s0267-3649(97)89753-1)
30. Windows PowerShell 2.0 best practices // ed. by Windows PowerShell Teams at Microsoft. Redmond, Wash: Microsoft Press, 2010. 715 p.

ДОДАТОК А.

```

Код для «Коли створюється файл» "type": "OpenApiConnection" "inputs" "parameters" "dataset":
"https://luguniveduua.sharepoint.com/sites/msteams_00000" "table": "4ea26962-1eec-430d-b442-a140c29940ce" "host": "apild":
"/providers/Microsoft.PowerApps/apis/shared_sharepointonline" "connection": "shared_sharepointonline" "operationId": "GetOnNewFileItems"
"recurrence": "frequency": "Minute" "interval": 1 "splitOn": "@triggerOutputs()[?body/value]"
}

Код для «Перше погодження» "type": "OpenApiConnectionWebhook" "inputs" "parameters" "approvalType": "Basic"
"WebhookApprovalCreationInput/title": "Відділ кадрів" "WebhookApprovalCreationInput/assignedTo": ".edu.ua"
"WebhookApprovalCreationInput/enableNotifications": true "WebhookApprovalCreationInput/enableReassignment": true
"WebhookApprovalCreationInput/attachments" "host": "apild": "/providers/Microsoft.PowerApps/apis/shared_approvals" "connection": "shared_approvals"
"operationId": "StartAndWaitForAnApproval" "runAfter": {}
}

Код для «Умова» "type": "If" "expression" "and" "equals" "actions" "Оновити властивості файлу" "type": "OpenApiConnection" "inputs" "parameters"
"dataset": "https://luguniveduua.sharepoint.com/sites/msteams_00000" "table": "4ea26962-1eec-430d-b442-a140c29940ce" "id": 2
"item/odata_x0421_x0442_x0430_x0442_x0443_x0441_x0434_x043e_x043a_x0443_x043c_x0435_x043d_x0442_x0430_/Value":
"На погоджені" "item/odata_x0422_x0438_x043f_x043f_x0456_x0434_x043f_x0438_x0441_x0443_/Value": "Внутрішній" "host": "apild":
"/providers/Microsoft.PowerApps/apis/shared_sharepointonline" "connection": "shared_sharepointonline" "operationId": "PatchFileItem"
"Start_and_wait_for_an_approval_1" "type": "OpenApiConnectionWebhook" "inputs" "parameters" "approvalType": "CustomResponse"
"WebhookApprovalCreationInput/responseOptions": "@outputs('Start_and_wait_for_an_approval')[?body/outcome]
"WebhookApprovalCreationInput/title": "Бухгалтерія" "WebhookApprovalCreationInput/assignedTo": " @luguniv.edu.ua"
"WebhookApprovalCreationInput/enableNotifications": true "WebhookApprovalCreationInput/enableReassignment": true "host": "apild":
"/providers/Microsoft.PowerApps/apis/shared_approvals" "connection": "shared_approvals" "operationId": "StartAndWaitForAnApproval" "runAfter":
"Оновити властивості файлу" "Succeeded" "Оновити властивості файлу_1" "type": "OpenApiConnection" "inputs" "parameters" "dataset":
"https://luguniveduua.sharepoint.com/sites/msteams_0000" "table": "4ea26962-1eec-430d-b442-a140c29940ce" "id": 2
"item/odata_x0421_x0442_x0430_x0442_x0443_x0441_x0434_x043e_x043a_x0443_x043c_x0435_x043d_x0442_x0430_/Value":
"Підписано" "item/odata_x0422_x0438_x043f_x043f_x0456_x0434_x043f_x0438_x0441_x0443_/Value": "Внутрішній" "host": "apild":
"/providers/Microsoft.PowerApps/apis/shared_sharepointonline" "connection": "shared_sharepointonline" "operationId": "PatchFileItem" "runAfter":
"Start_and_wait_for_an_approval_1" "Succeeded" "else" "actions" "Оновити властивості файлу_2" "type": "OpenApiConnection" "inputs"
"parameters" "dataset": "https://luguniveduua.sharepoint.com/sites/msteams_00000" "table": "4ea26962-1eec-430d-b442-a140c29940ce" "id": 2
"item/odata_x0421_x0442_x0430_x0442_x0443_x0441_x0434_x043e_x043a_x0443_x043c_x0435_x043d_x0442_x0430_/Value":
"Створено" "item/odata_x0422_x0438_x043f_x043f_x0456_x0434_x043f_x0438_x0441_x0443_/Value": "Внутрішній" "host": "apild":

```

```

"/providers/Microsoft.PowerApps/apis/shared_sharepointonline" "connection": "shared_sharepointonline" "operationId": "PatchFileItem"
"Опублікувати_повідомлення_в_чаті_або_каналі" "type": "OpenApiConnection" "inputs" "parameters" "poster": "Flow bot" "location": "Channel"
"body/recipient/groupId": "сес2079а-с935-4с08-а237-5аас41еба5с3" "body/recipient/channelId":
"19:214bfa5e9ce406e94f8be25a0969803@thread.tacv2" "body/messageBody": "&lt;p class='\"editor-paragraph\"&gt;Відмовлено&lt;/p" "host": "apild":
"/providers/Microsoft.PowerApps/apis/shared_teams" "connection": "shared_teams" "operationId": "PostMessageToConversation" "runAfter"
"Оновити_властивості_файлу_2" "SUCCEEDED" "runAfter" "Start_and_wait_for_an_approval" "Succeeded" "type": "OpenApiConnection" "inputs"
"parameters" "dataset": "https://luguniveduua.sharepoint.com/sites/msteams_8f9684" "table": "4ea26962-1eec-430d-b442-a140c29940ce" "id": 2
"item/odata_x0421_x0442_x0430_x0442_x0443_x0441_x0434_x043e_x043a_x0443_x043c_x0435_x043d_x0442_x0430_/Value":
"На_погоджені" "item/odata_x0422_x0438_x043f_x043f_x0456_x0434_x043f_x0438_x0441_x0443_/Value": "Внутрішній" "host" "apild":
"/providers/Microsoft.PowerApps/apis/shared_sharepointonline" "connection": "shared_sharepointonline" "operationId": "PatchFileItem" }
}

```

Додаток Б.

Models/DocumentStatus.cs
namespace DocumentNotifier.Models;

```

public enum DocumentStatus
{
    Created = 0,
    PendingApproval = 1,
    Approved = 2,
    Rejected = 3,
    Signed = 4,
    Archived = 5
}

```

Models/DocumentItem.cs

```

11 using System.ComponentModel.DataAnnotations;

namespace DocumentNotifier.Models; public class DocumentItem
{
    public int Id { get; set; } [Required] [StringLength(200)]
    11 public string Title { get; set; } = string.Empty;

    [Required]
    [StringLength(100)]
    public string DocumentType { get; set; } = string.Empty;

    [Required]
    [StringLength(100)]
    public string AuthorName { get; set; } = string.Empty;

    [Required]
    [EmailAddress]
    public string AuthorEmail { get; set; } = string.Empty;

    [Required]
    public DocumentStatus Status { get; set; } = DocumentStatus.Created;

    16 public DateTime CreatedAtUtc { get; set; } = DateTime.UtcNow;
    public DateTime? UpdatedAtUtc { get; set; }
}

```

Models/NotificationItem.cs

```

using System.ComponentModel.DataAnnotations;

namespace DocumentNotifier.Models;

public class NotificationItem
{
    public int Id { get; set; } public int DocumentItemId { get; set; } public DocumentItem? DocumentItem { get; set; }

    [Required]
    [EmailAddress]
    public string RecipientEmail { get; set; } = string.Empty;
}

```

```
[Required]
[StringLength(300)]
public string Title { get; set; } = string.Empty;
```

```
[Required]
[StringLength(1000)]
13 public string Message { get; set; } = string.Empty;

public bool IsRead { get; set; } = false;
public DateTime CreatedAtUtc { get; set; } = DateTime.UtcNow;
}
```

```
Data/AppDbContext.cs
using DocumentNotifier.Models; using Microsoft.EntityFrameworkCore; namespace DocumentNotifier.Data; public class 12 AppDbContext : DbContext
{
    public AppDbContext(DbContextOptions<AppDbContext> options) : base(options) { } public DbSet<DocumentItem> Documents =>
    Set<DocumentItem>();
    public DbSet<NotificationItem> Notifications => Set<NotificationItem>();

    protected override void OnModelCreating(ModelBuilder modelBuilder) { base.OnModelCreating(modelBuilder); modelBuilder.Entity<
    DocumentItem>()
        .Property(d => d.Status)
        .HasConversion<string>();

        modelBuilder.Entity<NotificationItem>()
            .HasOne(n => n.DocumentItem)
            .WithMany()
            .HasForeignKey(n => n.DocumentItemId)
            .OnDelete(DeleteBehavior.Cascade);
    }
}
```

Hubs/NotificationHub.cs

```
using Microsoft.AspNetCore.SignalR;

namespace DocumentNotifier.Hubs;

public class NotificationHub : Hub
{
    public async Task JoinUserGroup(string email)
    {
        if (!string.IsNullOrEmpty(email))
        {
            await Groups.AddToGroupAsync(Context.ConnectionId, email.Trim().ToLowerInvariant());
        }
    }
}
```

Services/NotificationService.cs

```
using DocumentNotifier.Data;
using DocumentNotifier.Hubs;
using DocumentNotifier.Models;
using Microsoft.AspNetCore.SignalR;
using Microsoft.EntityFrameworkCore;

namespace DocumentNotifier.Services;

public class NotificationService
{
    private readonly AppDbContext _db;
    private readonly IHubContext<NotificationHub> _hubContext;
    private readonly ILogger<NotificationService> _logger;

    public NotificationService(
```

```

AppDbContext db,
IHubContext<NotificationHub> hubContext,
ILogger<NotificationService> logger)
{
    _db = db;
    _hubContext = hubContext;
    _logger = logger;
}

public async Task NotifyDocumentStatusChangedAsync(DocumentItem document, CancellationToken cancellationToken = default)
{
    var notification = new NotificationItem
    {
        DocumentItemId = document.Id,
        RecipientEmail = document.AuthorEmail.Trim().ToLowerInvariant(),
        Title = $"Зміна статусу документа: {document.Title}",
        Message = $"Документ \"{document.Title}\" змінено. Новий статус: {GetStatusText(document.Status)}."
    };
    _db.Notifications.Add(notification);
    await _db.SaveChangesAsync(cancellationToken);
    await _hubContext.Clients
        .Group(notification.RecipientEmail)
        .SendAsync("ReceiveNotification", new
        {
            notification.Id,
            notification.Title,
            notification.Message,
            CreatedAtUtc = notification.CreatedAtUtc.ToString("u")
        }, cancellationToken);

    _logger.LogInformation(
        "Створено сповіщення {NotificationId} для {RecipientEmail} щодо документа {DocumentId}",
        notification.Id, notification.RecipientEmail, document.Id);
}

public async Task<List<NotificationItem>> GetUserNotificationsAsync(string email, CancellationToken cancellationToken = default)
{
    var normalizedEmail = email.Trim().ToLowerInvariant();

    return await _db.Notifications
        .Where(n => n.RecipientEmail == normalizedEmail)
        .OrderByDescending(n => n.CreatedAtUtc)
        .ToListAsync(cancellationToken);
}

public async Task MarkAsReadAsync(int id, string email, CancellationToken cancellationToken = default)
{
    var normalizedEmail = email.Trim().ToLowerInvariant();

    var notification = await _db.Notifications
        .FirstOrDefaultAsync(n => n.Id == id & n.RecipientEmail == normalizedEmail, cancellationToken);

    if (notification is null)
    {
        return;
    }

    notification.IsRead = true;
    await _db.SaveChangesAsync(cancellationToken);
}

private static string GetStatusText(DocumentStatus status) => status switch
{
    DocumentStatus.Created => "Створено",
    DocumentStatus.PendingApproval => "На погодженні",
    DocumentStatus.Approved => "Погоджено",
    DocumentStatus.Rejected => "Відхилено",
};

```

```

        DocumentStatus.Signed = &gt; "Підписано",
        DocumentStatus.Archived = &gt; "Архівовано",
        _ = &gt; "Невідомо"
    };
}

Controllers/DocumentsController.cs using DocumentNotifier.Data;
using DocumentNotifier.Models; using DocumentNotifier.Services;
6 using Microsoft.AspNetCore.Mvc; using Microsoft.EntityFrameworkCore; namespace DocumentNotifier.Controllers; public class DocumentsController
9 Controller { private readonly ApplicationDbContext _db; private readonly NotificationService notificationService;

    public DocumentsController(ApplicationDbContext db, NotificationService notificationService)
    {
        14 _db = db;
        _notificationService = notificationService;
    }

    public async Task<IActionResult> Index(CancellationToken cancellationToken)
    {
        var documents = await _db.Documents
            .OrderByDescending(d => d.CreatedAtUtc)
            .ToListAsync(cancellationToken);

        14 return View(documents);
    }

    [HttpGet] public IActionResult Create() { return View("Edit", new DocumentItem());
    }

    25 [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> Create(DocumentItem 14 model, CancellationToken cancellationToken)
    {
        if (!ModelState.IsValid) { return View("Edit", model);
        }

        model.CreatedAtUtc = DateTime.UtcNow;
        _db.Documents.Add(model);
        await _db.SaveChangesAsync(cancellationToken);

        await _notificationService.NotifyDocumentStatusChangedAsync(model, cancellationToken);

        return RedirectToAction(nameof(Index));
    }

    [HttpGet]
    15 public async Task<IActionResult> Edit(int id, CancellationToken cancellationToken)
    {
        var document = await _db.Documents.FindAsync(new object[] { id }, cancellationToken);
        15 if (document is null) return NotFound(); return View(document);
    }

    15 [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> Edit(int id, DocumentItem model, CancellationToken cancellationToken)
    {
        if (id != model.Id) return BadRequest();
        if (!ModelState.IsValid) return View(model);

        var document = await _db.Documents.FirstOrDefaultAsync(d => d.Id == id, cancellationToken);
        if (document is null) return NotFound();

        var oldStatus = document.Status;

        document.Title = model.Title;
        document.DocumentType = model.DocumentType;
        document.AuthorName = model.AuthorName;

```



```

document.AuthorEmail = model.AuthorEmail;
document.Status = model.Status;
document.UpdatedAtUtc = DateTime.UtcNow;

await _db.SaveChangesAsync(cancellationToken);

if (oldStatus != document.Status)
{
    await _notificationService.NotifyDocumentStatusChangedAsync(document, cancellationToken);
}

return RedirectToAction(nameof(Index));
}
}

```

Controllers/NotificationsController.cs

```

using DocumentNotifier.Services; using Microsoft.AspNetCore.Mvc; namespace DocumentNotifier.Controllers; public class NotificationsController :
Controller { private readonly NotificationService _notificationService;

    public NotificationsController(NotificationService notificationService)
    {
        _notificationService = notificationService;
    }

    public async Task<IActionResult> Index(string email, CancellationToken cancellationToken)
    {
        if (string.IsNullOrEmpty(email))
        {
            ViewBag.Email = string.Empty;
            return View(new List<DocumentNotifier.Models.NotificationItem>());
        }

        ViewBag.Email = email;
        var notifications = await _notificationService.GetUserNotificationsAsync(email, cancellationToken);
        return View(notifications);
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> MarkAsRead(int id, string email, CancellationToken cancellationToken)
    {
        await _notificationService.MarkAsReadAsync(id, email, cancellationToken);
        return RedirectToAction(nameof(Index), new { email });
    }
}

```

Program.cs

```

using DocumentNotifier.Data;
using DocumentNotifier.Hubs;
using DocumentNotifier.Services;
using Microsoft.EntityFrameworkCore;

var builder = WebApplication.CreateBuilder(args); builder.Services.AddControllersWithViews();

builder.Services.AddDbContext<AppDbContext>((options => options.UseSqlite(
builder.Configuration.GetConnectionString("DefaultConnection")));

builder.Services.AddSignalR();
builder.Services.AddScoped<NotificationService>();

var app = builder.Build(); using (var scope = app.Services.CreateScope()) { var db = scope.ServiceProvider.GetRequiredService<AppDbContext>
();
    db.Database.EnsureCreated();
}

if (!app.Environment.IsDevelopment())

```

```

{
    app.UseExceptionHandler("/Home/Error");
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();
app.UseRouting();

app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Documents}/{action=Index}/{id?}");

```

```
app.MapHub<NotificationHub>("/notificationHub");
```

```
app.Run();
```

appsettings.json

```

{
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=document_notifier.db"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  }
}

```

Views/Documents/Index.cshtml

```
@model IEnumerable<DocumentNotifier.Models.DocumentItem>
```

```

@{
    ViewData["Title"] = "Документи";
}

```

```
<h2>Документи</h2>
```

```
<p>
```

```
<a asp-action="Create" class="btn btn-primary">Створити документ</a>
```

```
</p>
```

```
<table class="table table-bordered">
```

```
<thead>
```

```
<tr>
```

```
<th>Назва</th>
```

```
<th>Тип</th>
```

```
<th>Автор</th>
```

```
<th>Email</th> <th>Cratyc</th> <th></th>
```

```
</tr>
```

```
</thead>
```

```
<tbody>
```

```
@foreach (var item in Model)
```

```
{
```

```
<tr>
```

```
<td>@item.Title</td>
```

```
<td>@item.DocumentType</td> <td>@item.AuthorName</td> <td>@item.AuthorEmail</td>
```

```
<td>@item.Status</td> <td>
```

```
<a asp-action="Edit" asp-route-id="@item.Id">Редагувати</a>
```

```
</td>
```

```
</tr>
```

```
}
```

```
</tbody>
```

```
</table>
```

```
<hr />
```

```

<input type="text" asp-controller="Notifications" asp-action="Index" method="get">
<input type="email" id="email" name="email" required />
<input type="submit" class="btn btn-secondary">Відкрити
</form>

```

Views/Documents/Edit.cshtml

```
@model DocumentNotifier.Models.DocumentItem
```

```

@{
    ViewData["Title"] = Model.Id == 0 ? "Створення документа" : "Редагування документа";
}

```

```

<h2>@ViewData["Title"]</h2>
<form asp-action="@((Model.Id == 0 ? "Create" : "Edit"))" method="post">
    @if (Model.Id != 0)
    {
        <input type="hidden" asp-for="Id" />
        <div class="mb-3">
            <label asp-for="Title" class="form-label">Назва</label>
            <input asp-for="Title" class="form-control" />
            <span asp-validation-for="Title" class="text-danger"></span>
        </div>
        <div class="mb-3">
            <label asp-for="DocumentType" class="form-label">Тип документа</label>
            <input asp-for="DocumentType" class="form-control" />
            <span asp-validation-for="DocumentType" class="text-danger"></span>
        </div>
        <div class="mb-3">
            <label asp-for="AuthorName" class="form-label">Автор</label>
            <input asp-for="AuthorName" class="form-control" />
            <span asp-validation-for="AuthorName" class="text-danger"></span>
        </div>
        <div class="mb-3">
            <label asp-for="AuthorEmail" class="form-label">Email автора</label>
            <input asp-for="AuthorEmail" class="form-control" />
            <span asp-validation-for="AuthorEmail" class="text-danger"></span>
        </div>
        <div class="mb-3">
            <label asp-for="Status" class="form-label">Статус</label>
            <select asp-for="Status" asp-items="Html.GetEnumSelectList<DocumentNotifier.Models.DocumentStatus>()" class="form-select">
            </select>
        </div>
        <input type="submit" class="btn btn-success">Зберегти
        <a asp-action="Index" class="btn btn-secondary">Скасувати</a>
    </form>

```

```

@section Scripts {
    <partial name="_ValidationScriptsPartial" />
}

```

Views/Notifications/Index.cshtml

```
@model IEnumerable<DocumentNotifier.Models.NotificationItem>
```

```

@{
    ViewData["Title"] = "Сповіщення";
    var email = (string?)ViewBag.Email ?? string.Empty;
}

```

```
<h2>Сповіщення</h2>
```

```
<p><strong>Користувач: @email</strong></p>
```

```
<div id="liveNotifications" class="mb-3"></div>
```

```

<table class="table table-striped">
    <thead>
        <tr>
            <th>Зарядка</th>
            <th>Повідомлення</th>
            <th>Дата</th>
            <th>Стан</th>
        </tr>
    </thead>
    <tbody id="notificationsBody">
        @foreach (var item in Model)
        {

```

```

<tr>
  <td>@item.Title</td>
  <td>@item.Message</td>
  <td>@item.CreatedAtUtc.ToLocalTime()</td>
  <td>@(item.IsRead ? "Прочитано" : "Нове")</td>
  <td>
    @if (!item.IsRead)
    {
      <form asp-action="MarkAsRead" method="post">
        <@19>@.html.AntiForgeryToken()
        <input type="hidden" name="id" value="@item.Id" />
        <input type="hidden" name="email" value="@email" />
        <@19>@.button type="submit" class="btn btn-sm btn-primary">Позначити прочитаним</button>
      </form>
    }
  </td>
</tr>
</tr>
</tbody>
</table>

```

```

</script>
window.currentUserEmail = "@email";
</script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/microsoft-signalr/8.0.7/signalr.min.js"></script>
<script src="~/js/notifications.js"></script>

```

wwwroot/js/notifications.js

```

(function () {
  const email = (window.currentUserEmail || "").trim().toLowerCase();
  if (!email) return;

  const liveBlock = document.getElementById("liveNotifications");
  const tableBody = document.getElementById("notificationsBody");

  8 const connection = new signalR.HubConnectionBuilder()
    .withUrl("/notificationHub")
    .withAutomaticReconnect()
    .build(); connection.on("ReceiveNotification", function (notification) {
    if (liveBlock) {
      const alert = document.createElement("div");
      alert.className = "alert alert-info";
      alert.innerHTML = `
        <strong>${notification.title}</strong><br>
        ${notification.message}<br>
        <small>${notification.createdAtUtc}</small>
      `;
      liveBlock.prepend(alert);
    }

    if (tableBody) {
      const row = document.createElement("tr");
      row.innerHTML = `
        <td>${notification.title}</td>
        <td>${notification.message}</td>
        <td>${notification.createdAtUtc}</td>
        <td>Нове</td>
      `;
      tableBody.prepend(row);
    }
  });

  connection.start()
    .then() => connection.invoke("JoinUserGroup", email)
    .catch(err => console.error(err.toString()));
})();

```