Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерної інженерії та інформаційних систем

КВАЛІФІКАЦІЙНА РОБОТА

бакалавр
Освітній рівень

Інформаційна система корпоративного документообігу
Назва теми

КвРІСТ.190117.19.01.02 ПЗ
Шифр

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *126 «Інформаційні системи та технології»*
Шифр, назва

Освітня програма *«Інформаційні системи та технології»*
Назва

Виконав: студент IV курсу, група ІСТ-19-1 _____ І.К.Васильковський
Підпис Ініціали, прізвище

Керівник _____ С.М.Лисенко
Підпис, дата Ініціали, прізвище

Нормоконтролер _____ С.М. Лисенко
Підпис, дата Ініціали, прізвище

До захисту допускаю:
Зав. кафедри комп'ютерної інженерії та інформаційних систем _____ Т.О. Говорущенко
Підпис Ініціали, прізвище

«1» червня 2023 р.

Хмельницький 2023

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Комп'ютерної інженерії та інформаційних систем

Освітній рівень бакалавр

Галузь знань 12 Інформаційні технології

Спеціальність 126 Інформаційні системи та технології

Освітня програма освітня програма «Інформаційні системи та технології»

ЗАТВЕРДЖУЮ

Зав. кафедри Т.О.Говорущенко

"11" 01 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

Васильковському Іллі Костянтиновичу

Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Інформаційна система корпоративного документообігу

Керівник проекту (роботи) Лисенко С.М., проф.

Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 1.03.2023 р. № 5

2. Строк подання студентом проекту (роботи) на кафедру 01.06.2023 р.

3. Вихідні дані до проекту (роботи) Завдання на дипломне проектування

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

Дослідження предметної області та постановка задачі

Проєктування інформаційної системи корпоративного документообігу

Програмна реалізація та тестування інформаційної системи

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

UML-діаграми

Програмна реалізація інформаційної системи

Дизайн інтерфейсу користувача

6. Консультанти розділів дипломного проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Лисенко С.М., професор кафедри КІІС		
Антиплагіат	Нічепорук А.О., доцент кафедри КІІС		

7. Дата видачі завдання «__01_»_____03_____2023 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1	Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи з керівником	20.02.2022	виконано
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	01.03.2023	виконано
3	Робота над розділом 1 – дослідження предметної області та постановка задачі	10.03.2023	виконано
4	Робота над розділом 2 – проєктування інформаційної системи корпоративного документообігу	20.04.2023	виконано
5	Робота над розділом 3 – програмна реалізація та тестування інформаційної системи	30.04.2023	виконано
6	Оформлення пояснювальної записки згідно вимог	11.05.2023	виконано
7	Попередній захист ВКР	26.05.2023	виконано
8	Захист ВКР на засіданні ЕК	Червень 2023 року	

Студент _____ І.К.Васильковський
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ С.М. Лисенко
Підпис Ініціали, прізвище

№ рядка	формат	Позначення	Найменування	Кіл. листів	№ екз	Примітка
			Текстові документи			
1		КвРІСТ.190117.19.01.02 ПЗ	Пояснювальна записка	56		
			Графічні матеріали			
2		КвРІСТ.190117.19.01.02 Е8	UML-діаграми	1		
3		КвРІСТ.190117.19.01.02 Е8	Програмна реалізація	1		
			інформаційної			
			системи			
4		КвРІСТ.190117.19.01.02 Е2	Дизайн інтерфейсу	1		
			користувача			
5		КвРІСТ.190117.19.01.02 Е2	Код програмного	21		
			забезпечення для			
			інформаційної системи			
			корпоративного			
			документообігу			

Зм	Арк	№ докум	Підпис	Дата	КвРІСТ.190117.19.01.02 ВП		
Розробив		Васильковський			Відомість проекту	Літера	Аркуш 1 / Аркушів 1
Перевір.		Лисенко С.М				У	
							ХНУ, ІСТ-19-1
Н. контр.		Лисенко С.М					
Затв.		Говорущенко		31.05			

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Інформаційна система корпоративного документообігу».

Автор роботи: Васильковський Ілля Костянтинович.

Керівник роботи: Лисенко Сергій Миколайович.

Пояснювальна записка: 56 с., 12 рис., 5 табл., 4 дод., 63 джерел.

Графічна частина: 3 креслення.

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ, UML, ПРОГРАМУВАННЯ, ФУНКЦІЙНІ ВИМОГИ, НЕФУНКЦІЙНІ ВИМОГИ, ЕЛЕКТРОНА СИСТЕМА. ВЕБ СИСТЕМА.

Метою роботи є розробка інформаційної система корпоративного документообігу.

Розроблено систему електронного документообігу для підприємства. Система надає електронну форму упаковки, зберігання, пошуку та контролю документів. Перехід від поточної системи ручної роботи до системи електронного документообігу допомагає зменшити витрати, автоматизація процесів підвищує безпеку документів і мінімізує помилки. Для проектування та форматування системи використовуються різні інструменти, такі як: Hypertext Preprocessor, Hypertext Markup Language, Cascading Style Sheet, Laravel, Bootstrap та деякі інші інструменти. Базу даних створено за допомогою системи керування базами даних MySQL.

Підпис студента Дата 31.05.2023

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ **4**

ВСТУП **5**

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ .. **8**

1.1 Корпоративний документообіг 8

2.2 Порівняльний аналіз переваг та недоліків існуючих інформаційних систем документообігу 9

2.3 Підходи до вирішення задачі проєктування інформаційних систем документообігу 14

1.5 Висновки 16

2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОРПОРАТИВНОГО ДОКУМЕНТООБІГУ **17**

2.1 Розробка користувальницьких вимог 17

2.2 Аналіз та специфікація вимог 19

2.3 Опис рішення 20

2.4 Специфікація системних вимог 23

2.4.1 Функційні вимоги 23

2.5 Нефункційні вимоги 25

2.6 Засоби розроблення ІС 26

2.6 Висновки 29

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ **30**

3.1 UML-діаграми 30

3.2 Програмна реалізація інформаційної системи 63

3.3 Дизайн інтерфейсу користувача 68

3.5 Висновки 73

ВИСНОВКИ **75**

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ **77**

Додаток А Інтерфейсні вікна інформаційної системи **85**

Додаток Б UML-діаграми **86**

Додаток В UML-діаграми **87**

Додаток Г Код програмного забезпечення для інформаційної системи корпоративного документообігу **88**

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ІС – Інформаційна система

ІСКД – Інформаційна система корпоративного документообігу

ПЗ - програмне забезпечення

ВСТУП

Система електронного документообігу є кроком у напрямку реалізації концепції безпаперового офісу [1]. Система електронного документообігу – це програмне забезпечення, яке полегшує процес керування файлами в організації. Це робиться шляхом перетворення документів та інших паперів у цифрову або електронну форму. Це керування документами забезпечує швидкий пошук і забезпечує безпеку переміщення файлів. Це також може заощадити час і гроші, створює більш ефективні процеси. «Система керування документами не тільки прискорює обробку файлів, але й забезпечує ефективність збору інформації та усуває потребу в ручній обробці документів. Паралельно з прискоренням процесу він безпечно шукає та отримує документи.

У повсякденній діяльності підприємство створює різноманітні документи з різних офісів. Зараз усі записи обробляються вручну. Ручна обробка та зберігання документів може призвести до різноманітних проблем і знизити продуктивність. Отже, потреба в системі, яка може зберігати та керувати такими документами в електронному вигляді, є дуже важливою. Запропонована система вирішить ці проблеми та дозволить ефективно керувати електронними документами, що зберігаються в центральному сховищі, і легко розміщувати їх в Підприємство . Ця система керування документами буде системою веб-додатків, яка зазвичай керує цифровими документами; зберігання, пошук і зменшення величезної кількості паперу. Користувачі системи також можуть створювати, обмінюватись, архівувати та шукати файли ефективним способом, що дозволяє ефективно ділитися знаннями всередині підприємства чи компанії.

Інформаційні системи документообігу (ІСДО) - це програмні засоби, призначені для автоматизації процесів обміну документами між різними підрозділами, підприємствами, організаціями або між фізичними особами.

Ці системи дозволяють створювати, редагувати, зберігати, обробляти і передавати документи в електронному вигляді. ІСДО забезпечують швидкий і ефективний обмін документами між різними користувачами, що дозволяє скоротити час на обробку документів та зменшити витрати на їх обробку і зберігання.

Серед функцій ІСДО можуть бути: створення і редагування документів, їх реєстрація та контроль руху документів, підписання електронним підписом, зберігання документів у відповідних електронних архівах та інше.

Використання ІСДО дозволяє поліпшити ефективність роботи підприємств та забезпечити високий рівень безпеки документів і даних, що є особливо важливим в сучасних умовах розвитку інформаційних технологій. НУ

Основне призначення інформаційних систем документообігу полягає в тому, щоб автоматизувати і полегшити процес обробки документів в організації. Ці системи забезпечують ефективне управління документами, збереження їх в електронному вигляді, контроль доступу до них та їх безпеку.

Забезпечення ефективного обміну документами між співробітниками та підрозділами організації, що дозволяє зменшити час, необхідний для обробки документів.

Забезпечення централізованого збереження документів в електронному вигляді, що дозволяє швидко знаходити та переглядати необхідні документи.

Автоматизація процесу створення документів, наприклад, шляхом використання шаблонів документів.

Контроль доступу до документів залежно від прав доступу, наданих користувачам.

Забезпечення безпеки даних та зменшення ризику втрати чи пошкодження документів.

Створення звітів та аналітики щодо обробки документів, що дозволяє оцінити ефективність роботи організації та виявити можливі проблеми в процесі обробки документів.

Узагальнюючи, інформаційні системи документообігу допомагають організації більш ефективно управляти документами, зменшити час, необхідний для їх обробки та зберігання, та збільшити безпеку даних.

Метою кваліфікаційної роботи є забезпечення керування інформаційною системою корпоративного документообігу.

Поставлена мета досягається розв’язанням такої основної задачі: проєктування та розроблення підсистеми керування документами, яка передбачатиме застосування різних сценаріїв керування інформаційною системою корпоративного документообігу

Об’єктом дослідження є процес керування корпоративним документообігом.

Предметом дослідження є система керування корпоративним документообігом.

Для досягнення поставленої мети використовуються такі методи дослідження, як методи синтезу, аналізу та моделювання процесів, принципи системного аналізу.

Практичне значення має спроєктована інформаційна система корпоративного документообігу яка передбачає застосування різних сценаріїв взаємодії з документами.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Корпоративний документообіг

Корпоративний документообіг є важливою складовою діяльності будь-якої організації. Він передбачає обмін інформацією та документами між працівниками організації та зовнішніми сторонами. Однак, цей процес може бути складним та часовим. Зараз організації зазвичай використовують електронні документообігові системи для полегшення процесу обміну документами та забезпечення більш ефективного управління документами та інформацією.[1-4]

Одним із ключових викликів, який виникає при впровадженні системи корпоративного документообігу, є забезпечення безпеки інформації. Організації повинні забезпечувати конфіденційність, цілісність та доступність даних, щоб запобігти несанкціонованому доступу до важливої інформації. Іншою проблемою є зберігання даних, оскільки неправильне зберігання може призвести до втрати даних або їх пошкодження.[8-12]

Крім того, іншою проблемою є відсутність стандартизації форматів документів, що ускладнює процес обміну документами між організаціями. Наприклад, документ, створений у Microsoft Word, може мати проблеми з відкриттям в LibreOffice, що може призвести до втрати даних або збоїв в роботі системи.

Іншим викликом є необхідність підтримки різноманітних типів документів, таких як текстові документи, електронні таблиці, зображення, відео та аудіофайли. Це може призвести до потреби використання різних інструментів для обробки різних типів документів та додаткових зусиль при забезпеченні безпеки даних.[5-7]

Також викликом є висока вартість розробки та впровадження системи корпоративного документообігу, що може бути значним бар'єром для малих і середніх підприємств. Крім того, необхідно забезпечити навчання та підтримку користувачів системи, щоб забезпечити її ефективне використання.

Загальні проблеми корпоративного документообігу можуть включати:

1. Відсутність стандартизації форматів документів
2. Потреба в підтримці різноманітних типів документів
3. Потреба у забезпеченні безпеки інформації
4. Висока вартість розробки та впровадження системи корпоративного документообігу
5. Потреба в навчанні та підтримці користувачів системи

Розв'язання цих проблем включає в собі використання стандартизованих форматів документів, використання універсальних інструментів для роботи з різними типами документів та використання ефективних механізмів захисту даних. Крім того, вартість розробки та впровадження системи корпоративного документообігу може бути знижена за рахунок використання відкритих джерел та готових рішень.[13-15]

У підсумку, розуміння проблем та викликів, пов'язаних з корпоративним документообігом, є важливим кроком у розробці ефективних інформаційних систем. Подальший аналіз цих проблем може допомогти розробити ефективні рішення для покращення процесів документообігу та забезпечення безпеки даних.[37-40]

2.2 Порівняльний аналіз переваг та недоліків існуючих інформаційних систем документообігу

На сьогодні існують різні програмні засоби, що дозволяють організувати корпоративний документообіг. Кожне з рішень має свої переваги та недоліки. Основні переваги цих засобів полягають у поліпшенні ефективності та якості роботи, зменшенні часу на обробку документів, підвищенні рівня безпеки та захисту інформації. Однак, як і будь-яке рішення, вони мають свої недоліки, наприклад, складність в розгортанні та підтримці, високі вартості на впровадження та розширення, необхідність навчання персоналу тощо.[19-21]

Одним з найбільш поширених програмних засобів є Microsoft SharePoint. Він має велику кількість функцій та можливостей, забезпечує зручну роботу з документами та розв'язання завдань, пов'язаних з їх обробкою. Однак, SharePoint може бути досить складним в розгортанні та налаштуванні, що вимагає певних знань та навичок.

Microsoft SharePoint: це один з найбільш популярних інструментів для документообігу. Перевагами SharePoint є його інтеграція з іншими продуктами Microsoft, можливість легко керувати правами доступу та створювати зручний інтерфейс користувача. Недоліками є складність налаштування та підтримки, а також високі витрати на ліцензування.[16-18]

Google Drive: є однією з найпопулярніших систем документообігу, яка надає користувачам можливість зберігати, керувати та обмінюватися документами в хмарі. Google Drive є частиною платформи Google Workspace (раніше відомої як G Suite), яка також включає Gmail, Google Docs, Google Sheets та інші сервіси.

Основні можливості Google Drive включають:

Зберігання документів у хмарі та можливість доступу до них з будь-якого пристрою з Інтернет-підключенням.

Можливість створювати, редагувати та спільно використовувати документи з іншими користувачами в режимі реального часу.

Управління правами доступу до документів на основі різних рівнів доступу, включаючи публічний доступ та обмеження доступу за запрошенням.

Інтеграція з іншими сервісами Google, такими як Google Docs, Google Sheets, Google Slides, Google Forms та Google Drawings.

Автоматичне зберігання та синхронізація змін з усіх пристроїв, на яких користувач має доступ до Google Drive.

Можливість роботи з документами в офлайн-режимі та автоматичне оновлення змін при наступному підключенні до Інтернету.

Вбудовані інструменти для пошуку та фільтрації документів за різними параметрами.

Переваги Google Drive полягають у зручності використання, широкому функціоналі та легкій інтеграції з іншими сервісами Google. Однак, недоліки можуть включати обмеження щодо максимального розміру файлів та ємності безкоштовного акаунту, а також обмеження щодо приватності даних, оскільки всі дані зберігаються на серверах Google.. Перевагами Google Drive є його безкоштовність (для базової версії), доступність з будь-якого пристрою та можливість редагувати документи в режимі реального часу. Недоліками є обмежена можливість налаштування доступу до документів та обмежена функціональність порівняно зі SharePoint.[22-26]

Dropbox: це інформаційна система, що надає можливість зберігати та обмінюватися документами в хмарному середовищі. Перевагами Dropbox є його простота використання, можливість автоматичного синхронізування даних та широкий вибір плагінів для інтеграції з іншими продуктами. Недоліками є обмежена функціональність порівняно зі SharePoint та обмежена можливість налаштування доступу до документів.[27-28]

Інший засіб - Google Workspace - є хмарним рішенням, яке дозволяє ефективно організувати роботу з документами та комунікацією між

співробітниками. Він має простий та зрозумілий інтерфейс, що дозволяє легко зорієнтуватись у його можливостях. Проте, Google Workspace має деякі обмеження, зокрема, щодо можливостей редагування документів.

Крім того, одним з основних недоліків багатьох існуючих рішень є відсутність можливості налаштування інтерфейсу користувача під конкретні потреби компанії. Це може бути особливо проблематичним для компаній, які працюють у специфічних галузях, де потрібні специфічні функції та налаштування. Наприклад, у компанії, яка займається розробкою програмного забезпечення, можуть бути специфічні потреби, такі як можливість інтеграції зі сторонніми сервісами або можливість збереження коду в документах.[41-44]

Alfresco - це відкрита платформа для документообігу, що надає можливість зберігати, керувати та обмінюватися документами в організації. Ця система має відкритий вихідний код та підтримується активною спільнотою розробників.

Основні переваги Alfresco включають:

Відкритий код та можливість налаштування під потреби організації.

Широкий функціонал для керування документами, включаючи можливість створювати, редагувати, копіювати, переміщати та видаляти документи.

Можливість керувати правами доступу до документів на основі ролей та груп користувачів.

Інтеграція з іншими продуктами та рішеннями, наприклад з Microsoft Office, Google Docs, Adobe Creative Suite та іншими.

Можливість роботи з документами в офлайн-режимі та автоматичне оновлення змін при наступному підключенні до мережі.

Недоліки Alfresco можуть включати складність встановлення та налаштування, високу вартість підтримки та платних модулів, а також нестабільність роботи на деяких платформах. Однак, загалом, Alfresco є потужним інструментом для керування документами та документообігу в організації.

З іншого боку, існують рішення, які дозволяють налаштувати інтерфейс користувача з урахуванням потреб конкретної компанії. Наприклад, розробники можуть надавати можливість налаштовувати різні аспекти інтерфейсу, такі як відображення списку документів або доступ до певних функцій. Це дозволяє компанії забезпечити більш ефективну роботу з документами та більш комфортний інтерфейс для користувачів.[29-32]

Також важливим аспектом є можливість інтеграції з іншими системами, що використовуються в компанії. Наприклад, у компанії може бути вже наявна система електронного документообігу, але вона не підходить для всіх завдань. У такому випадку, корисно мати можливість інтегрувати нову систему корпоративного документообігу з існуючою.

Однак, необхідно також зазначити, що при виборі рішення для корпоративного документообігу необхідно звертати увагу не лише на його функціональні можливості, але й на забезпечення безпеки даних

Крім того, слід враховувати інші аспекти, які можуть вплинути на вибір оптимального рішення. Наприклад, важливо враховувати вартість реалізації та підтримки системи, час настройки та внедрення, доступність технічної підтримки, можливості інтеграції з іншими системами, а також простоту використання та навчання персоналу.

DocuWare - це система документообігу, яка призначена для організації роботи з документами в різних галузях, включаючи логістику, банківську та медичну сфери. DocuWare дозволяє зберігати, керувати та обробляти документи на основі потоків даних.

OpenText - це платформа документообігу, яка використовується для зберігання та обміну документами в різних галузях, включаючи банківську, фармацевтичну та автомобільну індустрії. OpenText дозволяє зберігати та

обробляти різні типи документів, включаючи електронні форми та електронні листи.

Laserfiche - це система документообігу, яка використовується в багатьох галузях, включаючи медицину, фінанси та правоохоронні органи. Laserfiche дозволяє зберігати, керувати та обробляти різні типи документів, включаючи медичні записи, фінансові документи та судові документи.

eFileCabinet - це система документообігу, яка дозволяє зберігати та керувати різними типами документів. Вона призначена для використання в різних галузях, включаючи банківську, медичну та юридичну сфери.

Система має функції організації документів, пошуку, індексації та зберігання даних в електронному вигляді. Крім того, eFileCabinet має можливість інтеграції з іншими системами, що дозволяє її використовувати в комплексі з іншими програмними продуктами.[33-36]

2.3 Підходи до вирішення задачі проєктування інформаційних систем документообігу

Один з підходів полягає у використанні готових рішень, які вже існують на ринку. Це можуть бути комерційні продукти або безкоштовні відкриті рішення. Такий підхід зменшує витрати на розробку імовірність збоїв, але не завжди відповідає всім потребам користувачів.[45-46]

Інший підхід полягає у розробці інформаційної системи корпоративного документообігу самостійно. Цей підхід дозволяє створити рішення, що повністю відповідає потребам користувачів, але потребує великих витрат часу та коштів на розробку та впровадження.[54-58]

Також можна застосовувати гібридний підхід, коли на базі готових рішень створюються модифіковані рішення, що відповідають конкретним потребам користувачів.[47-49]

Вибір підходу до розробки інформаційної системи корпоративного документообігу залежить від потреб компанії, витрат на розробку та впровадження рішення, а також від наявності кваліфікованих спеціалістів у компанії.[50-53]

Проєктування інформаційних систем документообігу включає розробку та впровадження програмного забезпечення, яке забезпечує збір, обробку та зберігання документів в організації.[59-63]

Основні етапи проєктування інформаційної системи документообігу можуть включати:

Аналіз потреб користувачів та бізнес-процесів. На цьому етапі з’ясовують, які документи обробляються в організації, як вони розподіляються між різними відділами та які вимоги щодо їх обробки є.

1. Розробка концепції системи. На цьому етапі розробляється загальна концепція системи, яка відображає всі вимоги та потреби користувачів, а також технічні обмеження.

2. Розробка технічного завдання. На цьому етапі формулюється детальний опис функціональності системи та вимог щодо її розробки та впровадження.

3. Розробка архітектури та дизайн системи. На цьому етапі вирішується, які технології та архітектурні рішення використовувати для розробки системи, а також проектуються інтерфейси користувача та інші складові системи.

4. Розробка та тестування системи. На цьому етапі проводиться розробка та тестування програмного забезпечення системи.

5. Впровадження системи та підготовка користувачів до роботи з нею. На цьому етапі система встановлюється на комп’ютери користувачів та проводиться підготовка користувачів до роботи з нею.

6. Підтримка та модернізація системи. Після впровадження системи проводиться її підтримка її працездатності.

1.5 Висновки

В розділі проаналізовано наявні інформаційні системи документообігу, виявлено переваги та недоліки.

Зроблено висновок щодо потреби розроблення локальної інформаційної технології, яка б надавала електронну форму упаковки, зберігання, пошуку та контролю документів.

Перехід від системи ручної роботи до системи електронного документообігу допоміг би зменшити витрати, автоматизував процеси та підвищив би безпеку документів з мінімізацією помилки використання ІС.

2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОРПОРАТИВНОГО ДОКУМЕНТООБІГУ

2.1 Розробка користувальницьких вимог

Розглянемо основні практичні задачі та вимоги щодо проєктування інформаційної системи корпоративного документообігу:

1. Вивчення та аналіз функціональних і нефункціональних вимог користувача.
2. Запропонувати нову автоматизовану систему електронного документообігу.
3. Розробити нову систему електронного документообігу та визначити можливі шаблони проектування.
4. Реалізувати запропоновану систему
5. Випробувати та перевірити нову систему електронного документообігу.

Мета проекту передбачає розробку системи управління документами для підприємства чи компанії. Система надала веб-рішення для управління документами. Його можна використовувати для управління документною системою підприємства чи компанії.

1. Система підтримує систему керування електронними документами, включаючи завантаження, індексування, керування, зберігання, пошук і доступ до документів.
2. Система дозволяє пересилати будь-який документ між співробітниками та для негайного перегляду будь-ким в організації.
3. Забезпечити єдине сховище документів, доступ до якого мають авторизовані користувачі.

4. Надати можливість обмежувати те, що користувачі можуть бачити та робити залежно від рівня безпеки.

5. Забезпечити безпеку документів і підтримувати категорії папок, інформацію про місцезнаходження та інформацію про документи.

6. Забезпечити можливість ведення схеми класифікації документів, канцелярії та кафедри Підприємства чи компанії.

7. Надати можливість керувати політикою зберігання документів.

8. Система забезпечує можливість легкого резервного копіювання та зберігання даних поза сайтом.

9. Підтримує або надсилає сповіщення електронною поштою з метою сповіщення.

Система електронного документообігу допомагає ефективно організувати документи таким чином і допоможе користувачам системи. Так що значущість системи класифікується на дві великі групи відповідно до вимог до системи:

1. Мінімізує витрати на обладнання для ручного зберігання файлів, як-от шафи для документів, ящики для файлів, папки, кліпси, лотки тощо.

2. Покращить обслуговування підприємства чи компанії шляхом оперативного, актуального пошуку інформації.

3. Розвине спілкування: це може допомогти покращити мотивацію серед співробітників підприємства чи компанії.

4. Змінить робоче середовище Підприємства чи компанії на просте та привабливе.

5. Це покращить прийняття рішень, а нова система забезпечує швидший доступ до записів і їх пошук.

6. Мінімізує витрати часу та уникайте втрати записів.

Для співробітників підприємства чи компанії:

1. Мінімізує витрати часу

2. Зробить робочий процес активним для співробітників.

3. Ефективно використає людські ресурси.

4. Підвищить продуктивність праці співробітників шляхом вчасного надання відповідного документа зазначеним особам.

2.2 Аналіз та специфікація вимог

Вимога - це просто заява про те, що система повинна робити або які характеристики вона повинна мати. Під час проекту розробки системи будуть створені вимоги, які описуватимуть потреби бізнесу (бізнес-вимоги); що потрібно зробити користувачам (вимоги користувача); що має робити програмне забезпечення (функціональні вимоги); характеристики, які повинна мати система (нефункціональні вимоги); і як повинна бути побудована система.

Зараз більшість записів обробляються вручну. Ручна обробка та зберігання цих документів призводять до числа проблем. Щоб детальніше зрозуміти ситуацію, було проведено інтерв'ю з різними співробітниками, які працюють в різних офісах підприємства чи компанії.

Співбесіда з співробітниками офісу допомагає визначити такі ключові аспекти, як-от: який у них загальний тип запису, як вони наразі класифікують записи та як вони обробляють політику ведення та зберігання документів. Відповідно до проведених на даний момент спостережень та інтерв'ю, майже всі офіси використовують власні місцеві методи керування документами.

Наприклад, в офісі справ їх документи поділяються на сім категорій, використовуючи папки-коробки. Категорії розташовані за тематикою записів, наприклад: Листи, Протоколи, Нотатки, Заявки, Посібники, Щомісячна звітність та річний план. Фактично, коли їм потрібно передати документ з одного офісу до іншого, вони роблять це вручну та ведуть його слідкування. Вони також мають

список відправлення листів для контролю маршрутизації документів, цей список реєструється вручну кожного разу, коли документ переміщується з одного офісу до іншого. Приклад запису відправки містить: Дату, Тему, Отримувача, Офіс, Підпис та Додаток. Один єдиний офіс може мати кілька записів, які відрізняються один від одного. Наприклад, у офісі виконавчого віце-президента записи включають Угоди, Акредитацію, Конференції, Політики, Посібники, Звіти, Плани, Платежі, Зовнішні та внутрішні листи, Заявки, Протоколи та Нотатки. Кожна з вищезгаданих функцій може використовувати систему управління документами для різних цілей в різних офісах.

Інший момент полягає в тому, що кожен офіс або відділ використовує власну техніку для класифікації та впорядкування записів. Наприклад: офіс HRM упорядковує та класифікує записи за алфавітом для вхідних і вихідних листів. Відповідно до цього для організації планів і звітів вони також використовують класифікацію на основі дат. Так само всі офіси використовують власні методи класифікації.

2.3 Опис рішення

Запропонована система управління документами має функціонал. Дозволяє користувачам створювати, індексувати, керувати, зберігати, та отримувати доступ до документів. Усі електронні записи збираються в центральному сховищі двома способами: перший – користувач системи може сканувати записи за допомогою будь-якого типу сканера та зберігати до центрального сховища. Сучасні сканери мають багато функцій і містять подвійний драйвер, який дозволяє безпосередньо підключатися до іншого програмного забезпечення. Цей інтерфейс сканування дозволяє користувачам змінювати порядок сторінок, інвертувати та повертати документи перед їх збереженням. Крім того, до документів можна додавати

сторінки перед збереженням або розділяти їх без редактора PDF. Мережеві сканери також автоматично завантажують документи в певну папку. Як правило, вилученим документам присвоюється числове ім’я, відбувається автоматичне групування та редагування, зміна імені, і правильне місце зберігання все потрібно змінити вручну. Таким чином, це дозволяє уникнути часу та енергії, які зазвичай потрібні для процесу сканування. Другий — це записи, які вже перетворено на електронну копію, але не надіслано до центрального сховища. Щоб усі необхідні електронні копії записів збиралися та зберігалися в центральному сховищі. Починаючи з цього моменту, ця система дозволяє користувачеві індексувати та додавати записи до системи. Встановлює обов'язки: хто що архівує і хто має право видаляти документи з обліку або переміщувати документи. Кожен користувач має певні права в певних каталогах або підкаталогах, відповідно до дозволу, який їм надається, наприклад, користувач може переглядати документи в одній частині сховища, додавати та редагувати документи в іншій частині або видаляти документи та створювати під. -каталоги в іншій частині. Створює прості структури для записів, які базуються на схемах класифікації, які відповідають потребам кожного офісу. Переведення паперових документів в електронний формат із записом атрибутів запису та метаданих архівного запису (наприклад, предмет запису, номер (PK), назва, тип, дата реєстрації, статус, класифікація запису, графік зберігання, дата зміни, дата архівування). Розміщення документів за різними категоріями або за іншими ідентифікованими параметрами, що забезпечує швидший пошук документів.

Здійснює архівні записи. Записи – це всі види інформації, створені незалежно від формату, що надсилаються та отримуються Підприємством у послідовності виконання його повсякденної діяльності. Прикладами записів є листи, протоколи, службові записки, угоди, акредитація, конференція, політики, інструкції, звіти, плани,

Зовнішні та внутрішні листи, заявки тощо. Усі ці записи, які пройшли процеси оцінювання та зберігаються як архівні записи. Існує конкретна причина для архівування записів. Однією з очевидних причин є те, що коли запис більше не є активним або, іншими словами, на нього більше не посилаються активно, він вибирається для архівування.

Для захисту документації від наслідків стихійних лих документи надійно зберігаються в електронному вигляді. Буде рішення для резервного копіювання, включаючи резервне копіювання за межами підприємства, це допоможе Підприємства чи компанії зменшити ризик.

Система надасть звітність; звіти будуть доступні для користувачів на основі їхніх ролей. Звітність буде у формі звичайного документа, який може задовольнити певного користувача системи та експортувати звіти в інший формат документа, наприклад PDF. Приклад звітності: скільки вхідних/вихідних листів зареєстровано в певному офісі, звітність не обмежується лише вхідними та вихідними листами. Він включає: звітність, пов’язану з користувачами, офісами, відділами, створений документ і навіть звіти, пов’язані з фізичними документами.

- Системний адміністратор матиме доступ до всіх функцій управління офісом, користувачами, ролями, документами, повідомленнями, резервним копіюванням і архівуванням, а також до багатьох інших функцій системи.
- Система також матиме сповіщення електронною поштою. Наразі SMU підписаний на Microsoft office 365 для бізнесу. Office 365 — це інтегрований досвід програм і служб, доступних через Інтернет. Ця служба підтримує отримання служб програм Microsoft як на персональних комп’ютерах, так і на мобільних пристроях. Нова система підтримуватиме сповіщення електронною поштою, щоб користувачі цієї нової системи могли отримувати сповіщення за допомогою власного мобільного телефону. Припустимо, піднімається один документ, який вимагає

погодження. У подібній ситуації буде надіслано сповіщення, щоб нагадати тому, хто затверджує, що схвалення очікує на розгляд.

2.4 Специфікація системних вимог

Специфікація вимог до програмного забезпечення — це робочий продукт вимог, який формально визначає дизайн системи керування документами.

2.4.1 Функційні вимоги

Функціональні вимоги повинні описувати всі необхідні функціональні можливості або системні послуги та стосуватися того, що система повинна робити або надавати користувачам. Вони включають опис необхідних функцій, схеми відповідних звітів або онлайн-запитів, а також деталі даних, які будуть зберігатися в системі [47]. Функціональні вимоги пов'язані з конкретними функціями; завдання або поведінку, які запропонована система повинна підтримувати. Функціональні вимоги запропонованої системи описуються наступним чином.

Керувати створенням файлів: -Система повинна мати засіб для підтримки категорії. У цій можливості користувачі повинні мати можливість створювати файли та папки. Щоб ця функція працювала належним чином, необхідно виконати такі вимоги:

1. Створіть папку та введіть назву..\
2. Створити файл і зареєструвати інформацію.
3. Збережіть файл у папці.
4. Збережіть папку в базі даних.

Керування файлами та зберіганням документів: -Ця система матиме можливість зберігати інформацію про місцезнаходження фізичних документів. У

цьому об’єкті користувачі повинні мати можливість створювати, зберігати, оновлювати та видаляти інформацію про місцезнаходження фізичних документів у базі даних. Щоб ця функція працювала належним чином, необхідно виконати такі вимоги:

1. Документ слід завантажити.
2. Інформація про метадані документа буде зареєстрована.
3. Файл, що зберігається в папці.

Керувати отриманням файлів: -Ця системна функція допомагає шукати або отримувати документ за різними параметрами, такими як дата, час, назва документа, інформація про місцезнаходження, інформація про категорію імені папки та за вмістом документа. Щоб ця функція працювала належним чином, необхідно виконати такі вимоги:

1. Файл повинен бути створений.
2. Файл потрібно проіндексувати.

Керувати робочим процесом: -Система повинна мати можливість надсилати документи в робочому процесі в попередньо визначеному потоці (стан документа) у вигляді повідомлення. Щоб ця функція працювала належним чином, необхідно виконати такі вимоги:

1. Надішліть повідомлення як бізнес-процес організації одержувачу, який є відповідальним користувачем.
2. Контроль та аудит робочого процесу.
3. Отримання повідомлення.

Керувати користувачами: -Система повинна мати засіб для керування користувачами. У цьому об'єкті системний адміністратор створює користувача, відділ і структуру організації. Щоб ця функція працювала належним чином, необхідно виконати такі вимоги:

1. Створити користувача та надати роль.

2. Створити підрозділ.

2.5 Нефункційні вимоги

Нефункціональні вимоги (NFR) — це поведінкові властивості, якими повинна володіти система, вони визначають такі атрибути системи, як безпека, надійність, продуктивність, зручність обслуговування, масштабованість і зручність використання, обробка помилок, доступність і ефективність [44].

Продуктивність - оскільки DMS є веб-додатком, він може легко підтримувати більше користувачів одночасно. Крім того, передача інформації містить переважно текстові дані; це забезпечить більшу продуктивність DMS.

Безпека: Адміністративна частина системи доступна лише уповноваженим працівникам Адміністрації відповідно до наданих їм прав. Неавторизований користувач системи не повинен отримати доступ до сторінки адміністрування в жодному разі. Деякі ресурси та діяльність мають бути обмежені та дозволені авторизованим користувачам. Отже, система повинна використовувати методи авторизації та надавати або забороняти доступ користувачам відповідно до їхніх привілеїв.

Наявність - система буде доступна з будь-якого комп’ютера з локальним підключенням і буде доступна в будь-який час, коли користувач захоче скористатися системою.

Інтерфейс користувача - система повинна мати інтерактивний, динамічний і послідовний графічний інтерфейс користувача. Інтерфейс не повинен відволікати увагу. Більшість вводів слід виконувати за допомогою списку та розкривного вікна, що економить час і робить його зручним для користувача.

Обробка помилок - ІС повинна мати можливість відповідати на описові повідомлення про помилки. Відповідне повідомлення про помилку має

відображатися, коли користувачі вставляють неочікувані/неправильні дані. За допомогою цієї функції користувачі будуть проінформовані про помилки, які вони зробили, і про те, що робити, щоб легко усунути проблему.

Надійність – здатність системи виконувати необхідні функції за встановлених умов протягом певного періоду часу. Має бути частий і повний механізм резервного копіювання, щоб уникнути будь-якої втрати інформації та неузгодженості.

Ремонтопридатність - запропонована система має бути розроблена за допомогою архітектур MVC.

Для того, щоб в майбутньому було легко підтримувати та вдосконалювати систему, якщо виникнуть додаткові вимоги користувача та збій системи.

2.6 Засоби розроблення ІС

У наступному розділі розглядаються різні типи засобів розробки, які будуть використовуватися для проектування запропонованої системи.

UML Designer – інструмент моделювання UML. Він допомагає бізнес-користувачам швидко створювати надійні та зручні в обслуговуванні системи та може легко масштабуватися для розміщення великих команд, які співпрацюють над спільними проектами [42]. Enterprise Architect також має можливість запускати деякі динамічні моделювання моделей, щоб перевірити правильність моделей і забезпечити краще розуміння того, як працюють конкретні бізнес-системи. Це програмне забезпечення використовується для дизайну UML. Діаграми варіантів використання, класів, дій і послідовності.

Laravel – це MVC (model view controller) і платформа веб-розробки з відкритим кодом, написана на PHP. Його було розроблено, щоб покращити якість вашого програмного забезпечення шляхом зменшення вартості початкової

розробки та поточних витрат на обслуговування. Воно також покращує досвід роботи з програмою, надаючи чіткий виразний синтаксис і базовий набір функціональних можливостей, які заощадять ваші години часу реалізації [43]. Laravel є видатним членом нового покоління веб-фреймворків, і деякі його функції перераховані нижче:

1) модульна система упаковки з виділеним менеджером залежностей;

2) різні способи доступу до реляційних баз даних за допомогою маршрутизації;

3) утиліти, які допомагають у розгортанні та обслуговуванні програми;

4) Легка автентифікація завдяки простому та зручному інтерфейсу.

Laravel може мати об’єктно-орієнтовані бібліотеки, яких немає в інших популярних фреймворках PHP. Однією з попередньо встановлених бібліотек є бібліотека автентифікації. Його легко реалізувати, а також багато розширених функцій, таких як захист і шифрування CSRF (Cross-site Request Forgery).

Laravel підтримує архітектуру MVC, забезпечуючи чіткість між логікою та презентацією. MVC допомагає підвищити продуктивність, дозволяє краще документувати та має численні вбудовані функції. Ось як MVC працює для Laravel.

Laravel піклується про безпеку в своїх рамках. Він використовує просолений і хешований пароль, що означає, що пароль ніколи не збережеться як звичайний текст у базі даних. Він використовує алгоритм хешування Bcrypt для створення зашифрованого представлення пароля. Laravel використовує готові оператори SQL, які роблять атаки ін’єкцій неймовірними. Крім того, Laravel надає простий спосіб уникнути введення користувача, щоб уникнути впровадження користувачем тегу <script> [41].

Bootstrap – це інтерфейсна бібліотека для розробки адаптивних веб-додатків. Це безкоштовно та з відкритим кодом. Він містить шаблони дизайну на основі HTML і CSS для кнопок типографічних форм, навігації та інших компонентів

інтерфейсу, а також додаткові розширення JavaScript. На відміну від багатьох веб-фреймворків, він стосується лише зовнішньої розробки. Ця бібліотека також використовується для розробки адаптивної веб-програми в цьому проекті.

Java script є найпопулярнішою мовою програмування на стороні клієнта в Інтернеті на сьогоднішній день, а також другою мовою програмування на стороні сервера, яка швидко розвивається [44]. JavaScript надає функціональність сторінці, і це інтерпретована мова програмування, яка зазвичай використовується на стороні клієнта для керування введенням користувача, керування діями браузера, завантаження додаткового вмісту в браузер. JavaScript працює на комп'ютерах веб-користувачів, навіть коли вони офлайн, і його дуже легко реалізувати. Це легка скомпільована мова програмування з першокласними функціями. Він добре відомий як мова сценаріїв для веб-сторінок. JS — це багатопарадигмальна динамічна мова сценаріїв на основі прототипу, яка підтримує об'єктно-орієнтований, імперативний і декларативний стиль.

HTML — це мова розмітки, яка використовується для створення електронних документів, які називаються сторінками.

Потім відобразиться веб-сторінка. Кожна сторінка містить низку зв'язків з іншими сторінками, які називаютьсягіперпосилання,HTML-код забезпечує належне форматування тексту та зображень, щобІнтернет браузерможе відображати їх так, як вони мають виглядати.

Рекомендується відокремити структуру сторінки від її дизайну. Для цього CSS використовується як мова стилів. Він визначає макет, колір, розмір та інші атрибути елементів сторінки [45]. CSS допомагає відокремити інформаційний вміст документа від деталей його відображення. Деталі того, як відобразити документ, відомі як його стиль. Зберігання стилю окремо від вмісту допомагає уникнути дублювання, спростити обслуговування та використовувати той самий вміст у різних стилях для різних цілей. Він описує, як елементи HTML мають

відображатися на екрані. CSS може заощадити багато роботи, керуючи макетом кількох веб-сторінок одночасно.

MySQL — це вільнодоступна система управління реляційними базами даних (RDBMS) із відкритим кодом, яка використовує мову структурованих запитів (SQL). MySQL працює практично на всіх платформах, в т.ч Linux, UNIX, івікна. Хоча її можна використовувати в широкому діапазоні програм, MySQL найчастіше асоціюється з веб-додатками, а онлайн-публікація є важливим компонентом відкрите джерело. LAMP — це платформа веб-розробки, яка використовує Linux як операційна система, Apache як веб-сервер, MySQL як система керування реляційною базою даних і PHP як об'єктно-орієнтована мова сценаріїв [46]. Для системи керування базами даних обрано MYSQL, оскільки це найпоширеніша база даних з відкритим кодом. Це високопродуктивна та масштабована веб-програма та вбудована база даних. Він також швидкий, надійний, безкоштовний для завантаження та простий у використанні.

2.6 Висновки

В розділі описано основні користувальницькі вимоги до системи.

Здійснено аналіз та специфікацію вимог до пропонованої інформаційної системи.

Задано опис пропонованого рішення та наведено специфікацію системних вимог.

Також в розділі подано перелік функційних вимог, нефункційних вимог, а також описано засоби розроблення ІС.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 UML-діаграми

Модель аналізу діє як сполучна ланка між «описом системи» та «моделлю проектування» [48]. в моделі аналізу визначається інформація, функції та поведінка системи, які транслюються в архітектуру, інтерфейс і рівень проектування компонентів у «моделюванні дизайну».

Таблиця 3.1 – Список акторів і опис

Актори	Опис
Користувач	Має бути штат в SMU. Керівництво компанії обере групу осіб із достатнім досвідом шляхом створення вказівок і політики, які будуть відповідальними для конкретного офісу чи відділу
Адміністратор	Адміністратор буде відповідати за управління всією системою

Модель варіанта використання описує запропоновану функціональність нової системи. Випадок використання являє собою дискретну одиницю взаємодії між користувачем (людиною або машиною) і системою [25]. Випадок використання фіксує функціональні вимоги системи та повідомляє, що система робить. Він також може «включати» функції іншого варіанту використання або «розширювати» інший варіант використання своєю власною поведінкою, і це зазвичай пов’язано з учасниками системи. Актор — це людина або машина, яка

взаємодіє з системою для виконання значущої роботи. Як правило, діаграми варіантів використання служать візуальним змістом написаних варіантів використання. Для кожного варіанта використання письмовий варіант описує етапи взаємодії між актором і системою. Функціональні можливості запропонованої системи, як зазначено в попередніх розділах, пояснюються з точки зору випадків використання системи в наступному розділі. Динамічні моделі варіантів використання, взаємодії в часі описані за допомогою діаграм активності та діаграми послідовності (рис .3.2, рис.3.3).

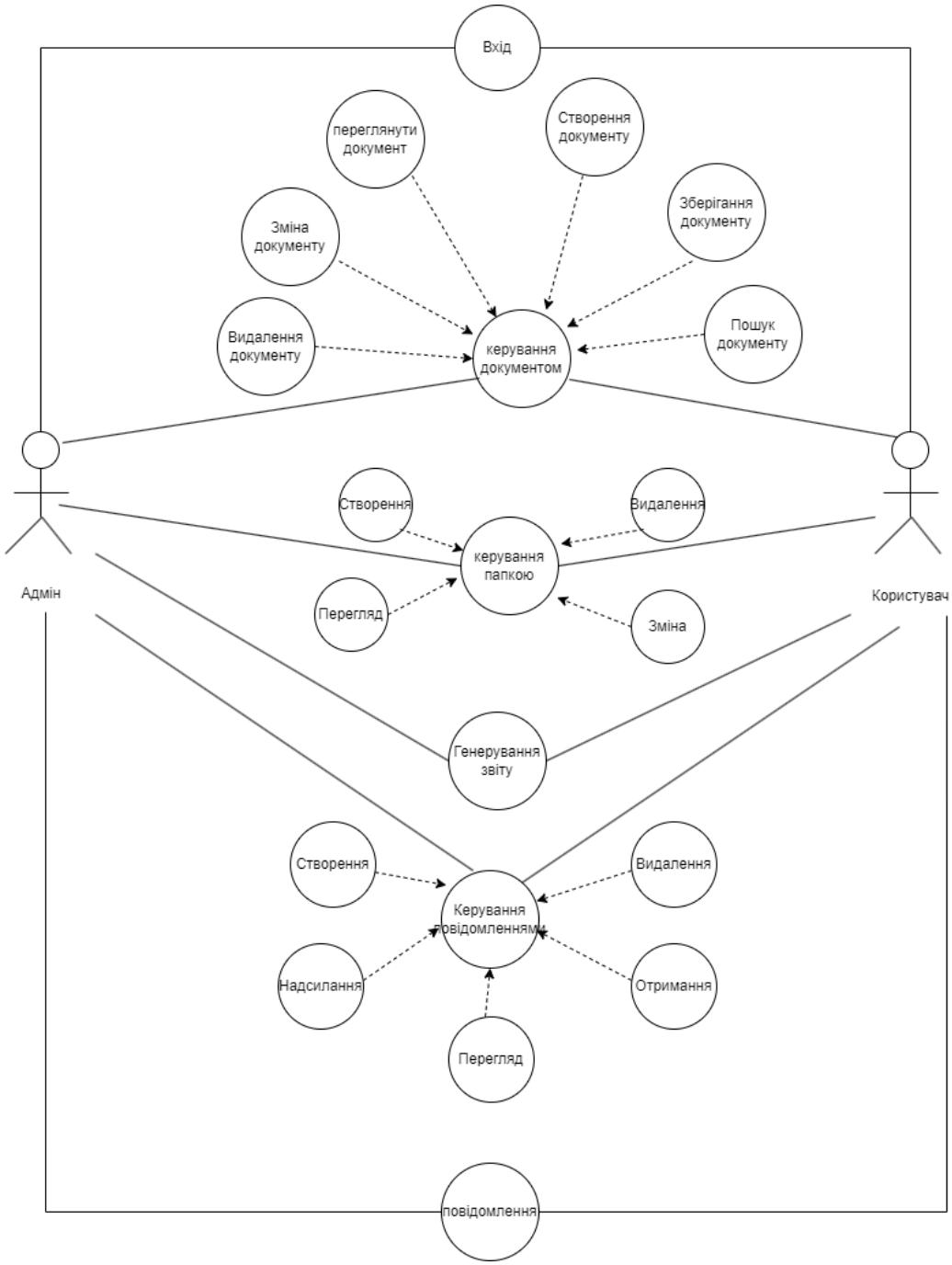


Рисунок 3.1 - Діаграма варіантів використання системи керування електронними документами

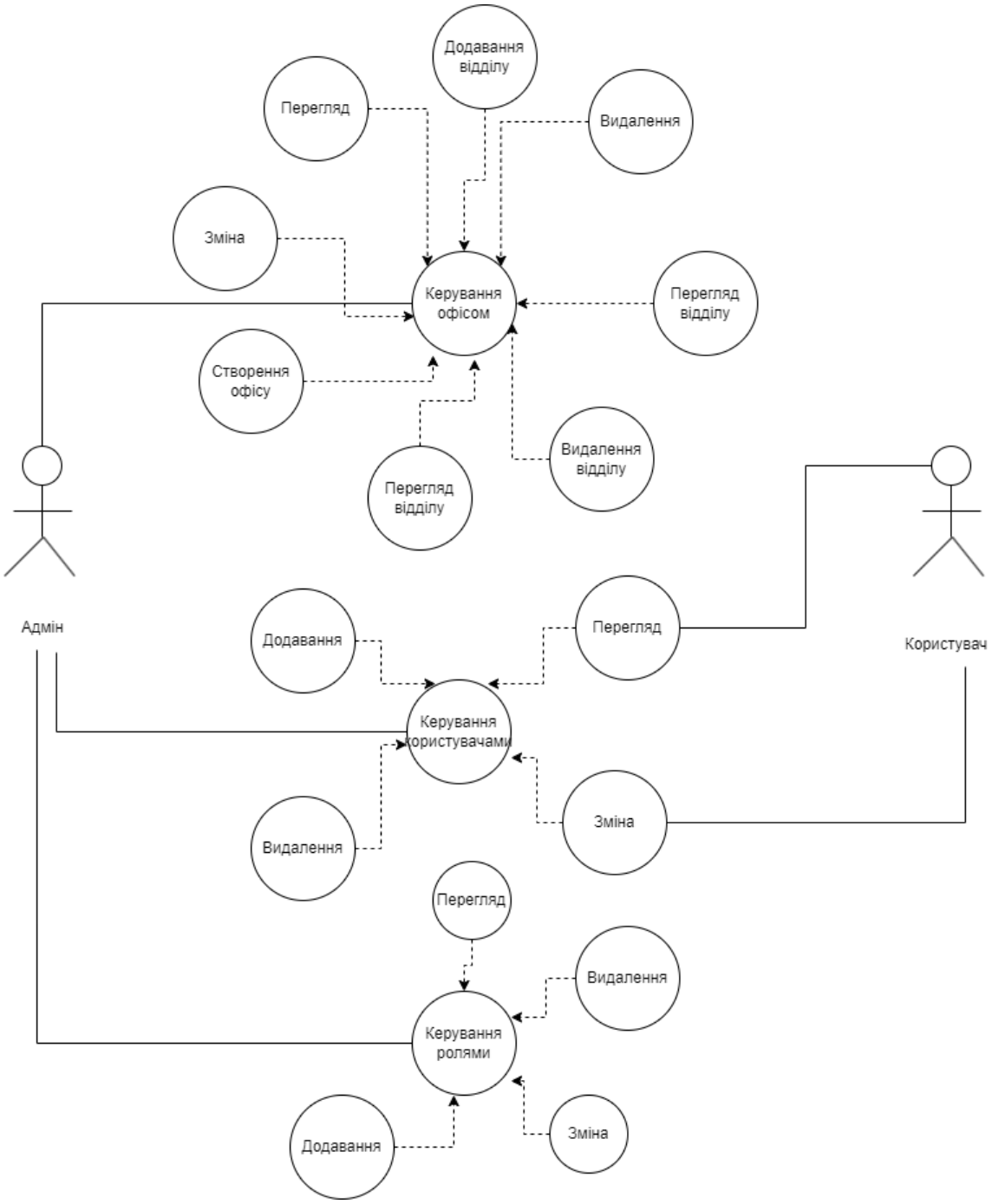


Рисунок 3.3 - Діаграма варіантів використання системи керування електронними документами

Опис варіанту використання описує головну системну мету конкретного варіанту використання, показує, як і що досягнуто.

Приклад випадків використання подано нижче:

ВАРІНТ ВИКОРИСТАННЯ - 001 Логін

ВАРІНТ ВИКОРИСТАННЯ - 002 Створити користувача

ВАРІНТ ВИКОРИСТАННЯ - 003 Видалити користувача

ВАРІНТ ВИКОРИСТАННЯ - 004 Перегляд користувача

ВАРІНТ ВИКОРИСТАННЯ - 005 Редагувати користувача

ВАРІНТ ВИКОРИСТАННЯ - 006 Створити документ

ВАРІНТ ВИКОРИСТАННЯ - 007 Редагувати документ

ВАРІНТ ВИКОРИСТАННЯ - 008 Переглянути документ

ВАРІНТ ВИКОРИСТАННЯ - 009 Видалити документ

ВАРІНТ ВИКОРИСТАННЯ - 010 Пошуковий документ

ВАРІНТ ВИКОРИСТАННЯ - 010 Створити папку

ВАРІНТ ВИКОРИСТАННЯ - 011 Переглянути папку

ВАРІНТ ВИКОРИСТАННЯ - 012 Видалити папку

ВАРІНТ ВИКОРИСТАННЯ - 013 Редагувати папку

ВАРІНТ ВИКОРИСТАННЯ - 014 Створити звіт

ВАРІНТ ВИКОРИСТАННЯ - 015 Відправити повідомлення

ВАРІНТ ВИКОРИСТАННЯ - 016 Отримати повідомлення

ВАРІНТ ВИКОРИСТАННЯ - 017 Видалити повідомлення

ВАРІНТ ВИКОРИСТАННЯ - 018 Переглянути повідомлення

ВАРІНТ ВИКОРИСТАННЯ - 019 Створити повідомлення

ВАРІНТ ВИКОРИСТАННЯ - 020 Сповіщення

ВАРІНТ ВИКОРИСТАННЯ - 021 Додати офіс

ВАРІНТ ВИКОРИСТАННЯ - 022 Переглянути кабінет

ВАРІНТ ВИКОРИСТАННЯ - 023 Редагувати кабінет

ВАРІНТ ВИКОРИСТАННЯ - 024 Видалити офіс

ВАРІНТ ВИКОРИСТАННЯ - 025 Додати відділ

ВАРІНТ ВИКОРИСТАННЯ - 026 Переглянути відділ

ВАРІНТ ВИКОРИСТАННЯ - 027 Редагувати відділ

ВАРІНТ ВИКОРИСТАННЯ - 028 Видалити відділ

ВАРІНТ ВИКОРИСТАННЯ - 029 Додати роль

ВАРІНТ ВИКОРИСТАННЯ - 030 Переглянути роль

ВАРІНТ ВИКОРИСТАННЯ - 031 Редагувати роль

ВАРІНТ ВИКОРИСТАННЯ - 032 Видалити роль

Основний потік:

1. Користувач вводить URL-адресу в адресний рядок браузера
2. Система відобразить сторінку входу
3. Система просить користувача ввести ім'я користувача та пароль.
4. Натисніть Надіслати
5. Система перевіряє правильність імені користувача та пароля
6. Система відкриває відповідну сторінку.
7. Випадок використання закінчується.

Варіант використання починається, коли адміністратор хоче додати користувача до системи:

1. Адміністратор натискає посилання Реєстрація користувача в підменю реєстрації.
2. Система вимагає від адміністратора ввести повну інформацію про користувача.
3. Адміністратор вводить всю необхідну інформацію про користувача.
4. Система перевіряє введену інформацію про користувача.
5. Користувачів успішно додано до системи.
6. Варіант використання закінчується

Альтернативний потік Якщо адміністратор вводить невірну інформацію:

1. Система перевіряє інформацію та відображає некоректні місця.

2. Випадок використання закінчується

Умови виходу. Якщо варіант використання був успішним, адміністратор створює користувача в системі. В іншому випадку стан системи не змінюється

Варіант використання починається, коли адміністратор хоче видалити користувача з системи:

1. Адміністратор обирає Вилучити користувача з підменю реєстрації.
2. Адміністратор вибирає користувача, якого буде видалено з системи.
3. Система перевіряє, щоб адмін обов'язково видалив.
4. Користувача буде видалено з системи.
5. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор видаляє користувача із системи. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли адміністратор хоче переглянути користувачів:

1. Адміністратор обирає перегляд користувача з підменю реєстрації.
2. Системний список користувачів.
3. Адміністратор вибирає користувача, якого хоче переглянути.
4. Випадок використання закінчується.

Альтернативний потік Якщо користувач бажає переглянути свою інформацію про користувача.

1. Користувач вибирає переглянути користувача з підменю імені.
2. Виберіть посилання змінити пароль
3. Відображатиметься інформація про користувача
4. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть переглядати користувачів системи. В іншому випадку стан системи не змінюється.

Випадок використання починається, коли адміністратор хоче редагувати користувачів:

1. Адміністратор обирає редагувати користувача з підменю реєстрації.
2. Система виводить список користувачів.
3. Адміністратор вибирає користувача, якого хоче редагувати
4. Адміністратор змінив правильну інформацію.
5. Випадок використання закінчується.

Альтернативний потік Якщо користувач бажає відредагувати свої дані користувача. Якщо користувач бажає змінити паролі

5. Користувач вибирає зміну пароля з підменю імені.
6. Система відображає детальну інформацію про користувача
7. Користувач вводить комбінацію великих і малих літер, спеціальний символ і число з паролем щонайменше шести символів.
8. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть редагувати користувачів системи. В іншому випадку стан системи не змінюється.

Випадок використання починається, коли користувач системи бажає створити та зберегти документи в системі:

1. Користувач системи вибирає програмне посилання на документ із підменю керування документами.
2. Виберіть створити документ і введіть необхідну інформацію про документ.
3. Завантажте файли м’якої копії в систему.
4. Система підтвердила інформацію
5. Документ буде створено.
6. Випадок використання закінчується.

Альтернативний потік. Якщо користувач системи клацне посилання на фізичний документ у підменю керування документами.

1. Виберіть фізичний документ і введіть необхідну інформацію про документ.
2. Система перевірить інформацію.
3. Новий документ буде збережено в системі.
4. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть створювати та зберігати документи в системі. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли користувач системи бажає редагувати збережені документи системи:

1. Користувач системи натискає програмний документ у підменю керування документами.
2. Система відображає список документів.
3. Виберіть документ і надайте дійсну інформацію.
4. Система перевірить інформацію.
5. Документ буде успішно відредаговано.
6. Випадок використання закінчується.

Альтернативний потік Якщо користувач системи клацне посилання на фізичний документ у підменю керування документами.

1. Система відображає список документів.
2. Виберіть документ і надайте дійсну інформацію.
3. Система перевірить інформацію.
4. Документ буде успішно відредаговано.
5. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть редагувати документ у системі. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли користувач системи бажає переглянути збережені документи системи.

1. Користувач системи натискає програмний документ у підменю керування документами.
2. Система відображає список документів.
3. Виберіть документ для перегляду зі списку.
4. Документ буде успішно переглянуто.
5. Випадок використання закінчується.

Альтернативний потік Якщо користувач системи клацне посилання на фізичний документ у підменю керування документами:

1. Система відображає список документів.
2. Виберіть документ для перегляду зі списку документів.
3. Документ буде успішно переглянуто.
4. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть переглянути документ. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли користувач системи бажає переглянути видалені документи системи:

1. Користувач системи натискає програмний документ у підменю керування документами.
2. Система відображає список документів.
3. Виберіть видалений документ зі списку документів.
4. Система перевірить, чи впевнені ви видалили цей документ.

5. Документ буде успішно видалено.

6. Випадок використання закінчується.

Альтернативний потік. Якщо користувач системи клацне посилання на фізичний документ у підменю керування документами.

1. Система відображає список документів.

2. Виберіть видалений документ зі списку документів.

3. Система перевірить, чи впевнені ви видалили цей документ.

4. Документ буде успішно видалено.

5. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть видалити документ. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли системний користувач хоче створити папку в системі:

1. Користувач системи натискає програмний документ у підменю керування документами.

2. Система відображає список папок зліва.

3. Виберіть батьківську папку та натисніть «Створити папку».

4. Користувач вводить необхідну інформацію.

5. Система перевірить інформацію.

6. Папка буде створена.

7. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть створити папку. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли користувач системи бажає створити звіт:

1. Користувач натискає підменю звіту.

2. Система формує звіт у форматі таблиці, який можна конвертувати у формат pdf.

3. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть створити системний звіт. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли користувач системи бажає надіслати повідомлення:

1. Користувач системи вибирає написати з підменю повідомлення.

2. Користувачі вводять правильну адресу електронної пошти отримувача та іншу необхідну інформацію.

3. Система перевірить надану інформацію.

4. Повідомлення буде надіслано отримувачу.

5. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть надіслати повідомлення. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли користувач системи бажає отримати повідомлення.

1. Користувач системи обирає папку "Вхідні" з підменю повідомлень.

2. Система відобразить список вхідних повідомлень.

3. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор і користувач можуть отримати повідомлення. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли користувач системи бажає повідомити електронною поштою:

1. Користувач системи вибирає повідомлення з підменю системи.

2. Користувач натискає сповіщення та додає електронну адресу одержувача та тему.

3. Система перевірить електронну адресу

4. Сповіщення буде надіслано.

5. Випадок використання закінчується.

Умови виходу Якщо варіант використання був успішним, адміністратор і користувач можуть повідомити. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли адміністратор хоче додати офіс.

1. Адміністратор натискає на «Керування офісом» у підменю реєстрації.

2. Адміністратор натискає «Додати офіс» і вказує необхідну інформацію.

3. Система перевірить необхідну інформацію

4. Офіс буде додано до системи.

5. Випадок використання закінчується.

Умови виходу.Якщо варіант використання був успішним, адміністратор може додати офіс до системи. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли адміністратор хоче додати відділи.

1. Адміністратор натискає на керування відділом у підменю реєстрації.

2. Адміністратор натискає «Додати відділ» і надає необхідну інформацію.

3. Система перевірить необхідну інформацію

4. Відділ буде додано до системи.

5. Випадок використання закінчується.

Умови виходу Якщо варіант використання був успішним, адміністратор може додати відділ до системи. В іншому випадку стан системи не змінюється.

Варіант використання починається, коли адміністратор хоче додати ролі.

1. Адміністратор натискає на керування користувачами в підменю реєстрації.

2. Адміністратор натискає «Додати користувача» та надає необхідну інформацію.

3. Адміністратор надає роль користувачу.

4. Система перевірить необхідну інформацію

5. Роль буде додано в систему.

6. Випадок використання закінчується.

Умови виходу. Якщо варіант використання був успішним, адміністратор може додати роль системному користувачеві. В іншому випадку стан системи не змінюється.

4.6.4 Діаграма послідовності

UML пропонує дуже великий набір конструкцій для кожної зі своїх діаграм і UMLДіаграми послідовностей — це діаграми взаємодії, які детально описують, як виконуються операції. Діаграма послідовності показує взаємодію між об’єктами протягом певного часу [49].

Одна діаграма послідовності зазвичай представляє окремий варіант використання або потік подій. Діаграми послідовності — чудовий спосіб задокументувати сценарії використання, а також зафіксувати необхідні об’єкти на ранніх стадіях аналізу та перевірити використання об’єктів пізніше на етапі розробки.

Послідовнісні діаграми показують потік повідомлень від одного об'єкта до іншого, і відповідають методам та подіям, які підтримує клас або об'єкт. Наведена нижче діаграма показує послідовність дій для Системи управління електронними документами (ІСКД) підприємства чи компанії SMU, з користувачем або актором зліва, який ініціює потік подій та повідомлень, які відповідають сценарію використання.

Повідомлення, які передаються між об'єктами, стануть операціями класу у кінцевій моделі (рис. 3.4- рис.3.13).

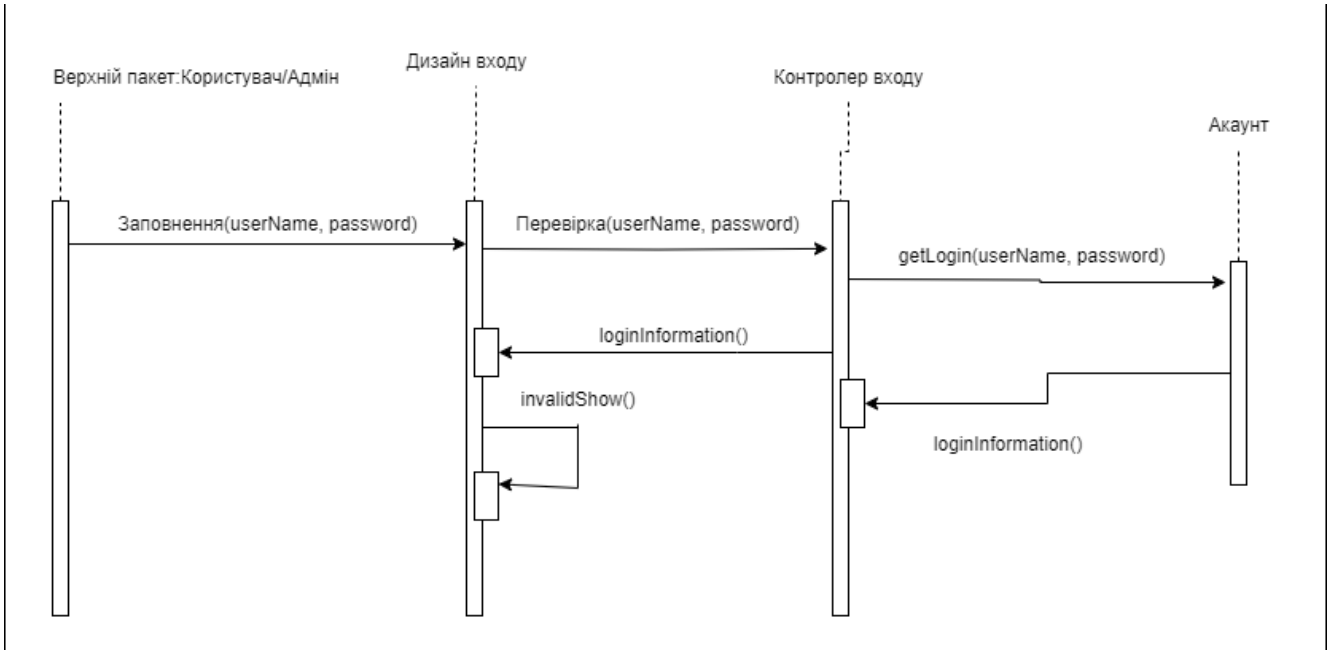


Рисунок 3.4 - Діаграма послідовності для входу

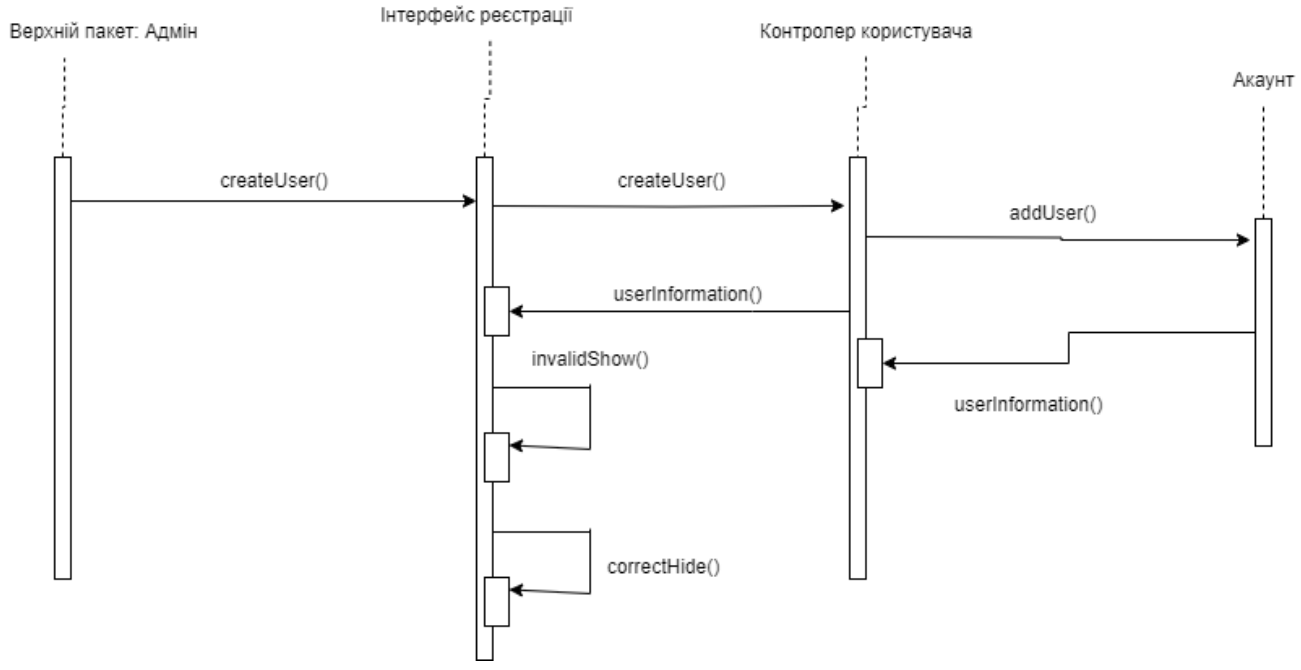


Рисунок 3.5 - Діаграма послідовності для створення користувача

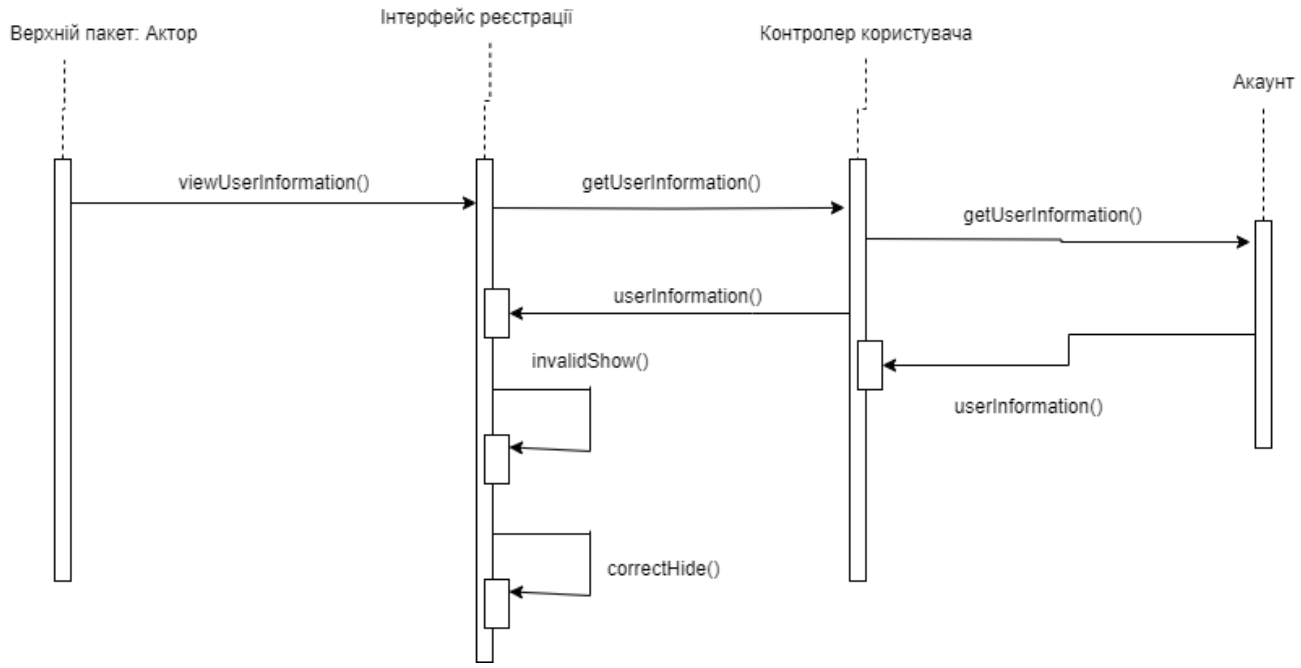


Рисунок 3.6 - Діаграма послідовності для перегляду користувача

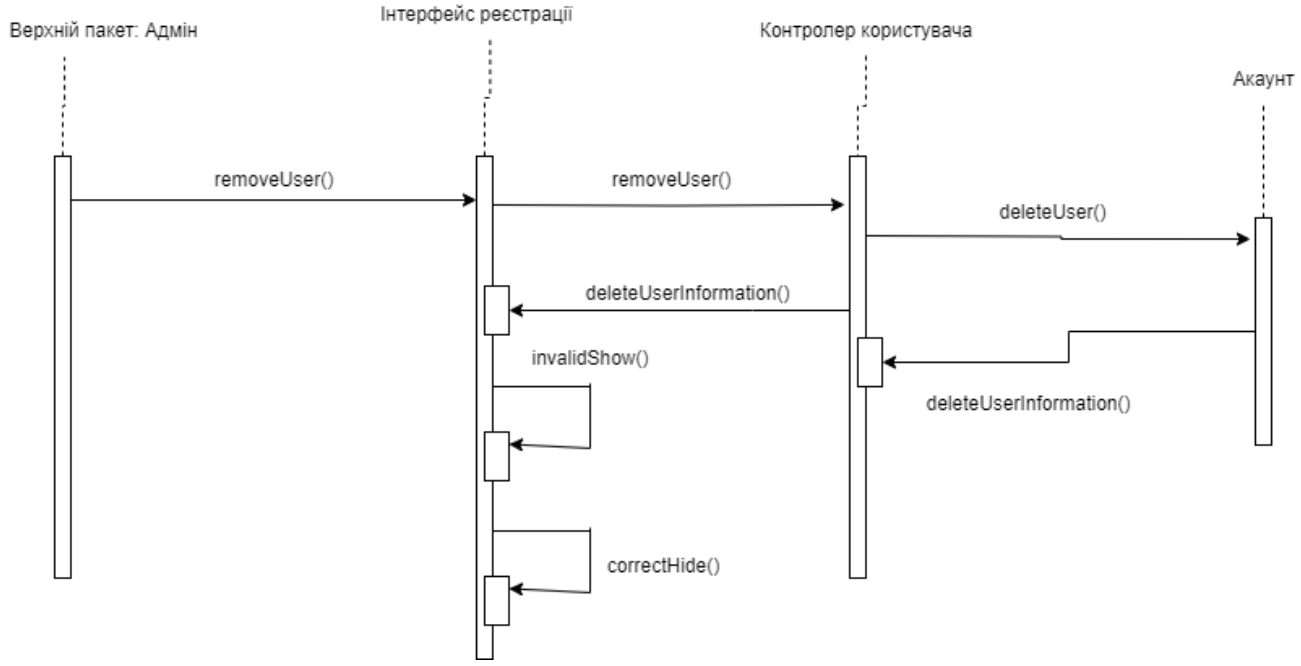


Рисунок 3.7 - Діаграма послідовності для видалення користувача

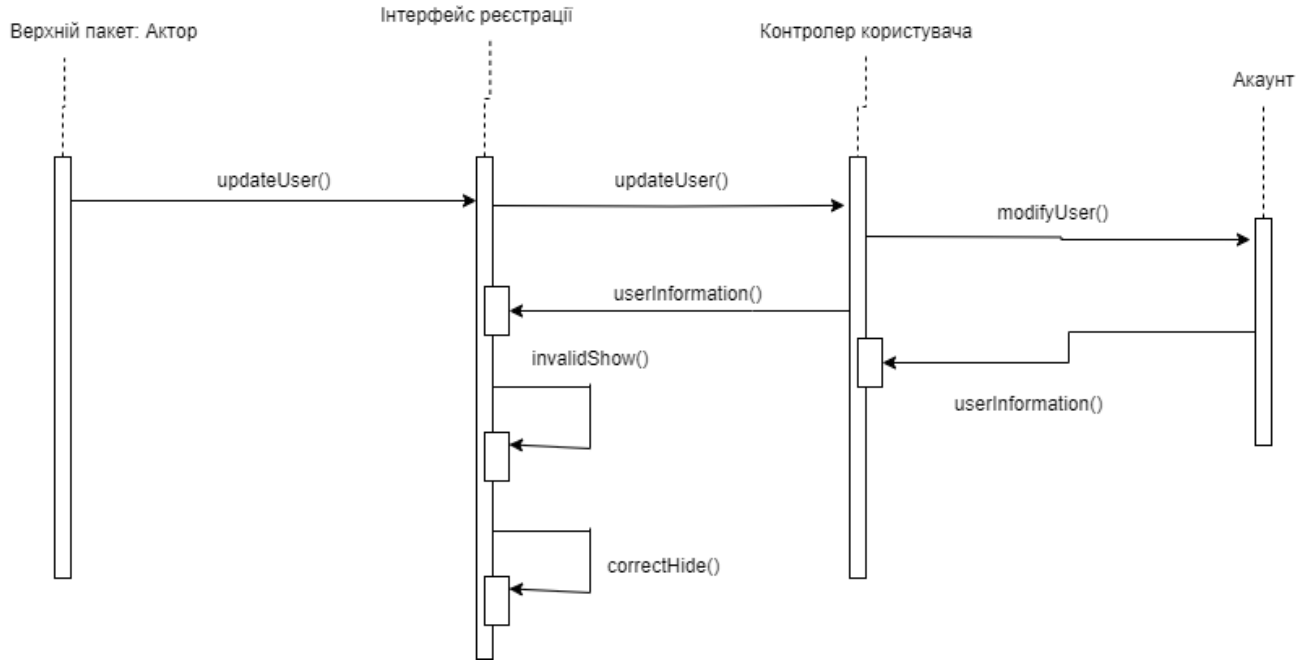


Рисунок 3.8 - Діаграма послідовності для редагування/оновлення користувача

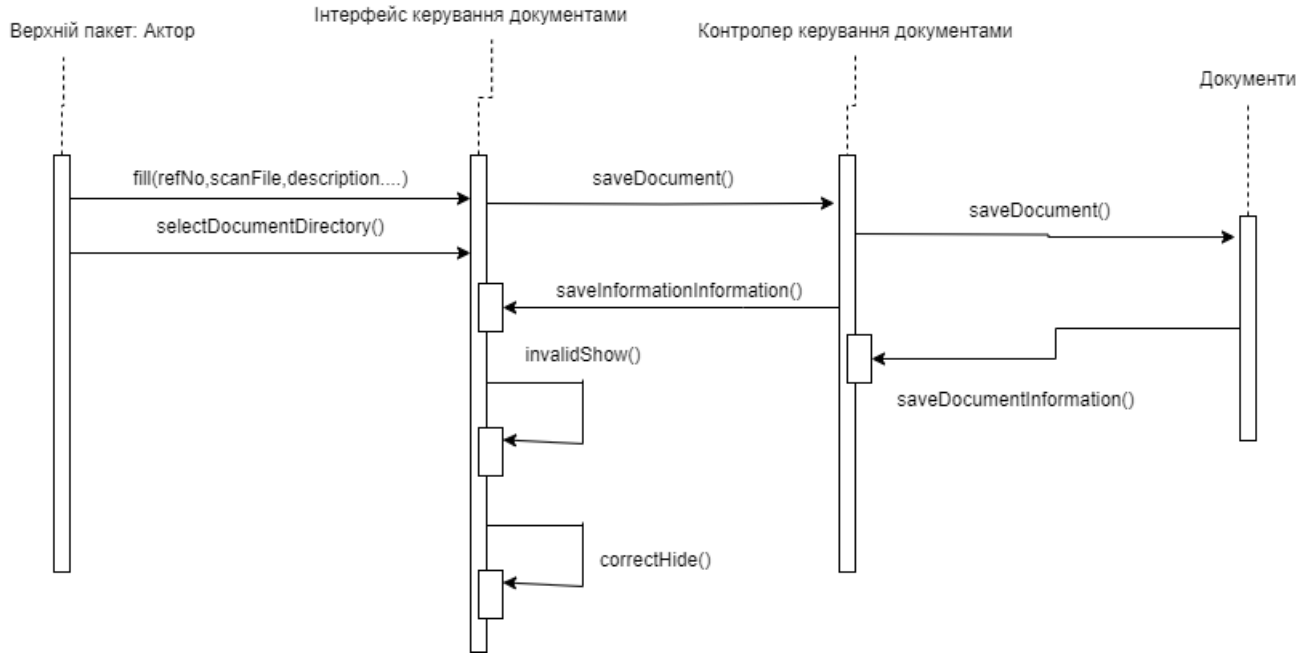


Рисунок 3.9 - Діаграма послідовності для створення документа

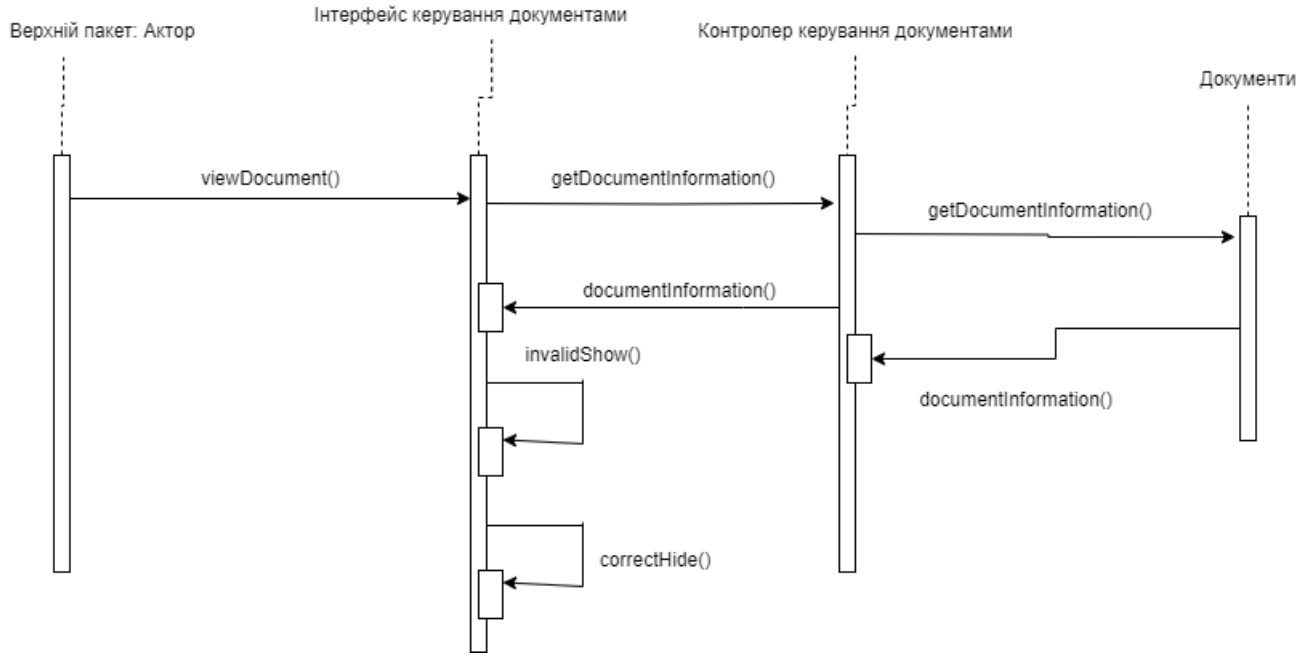


Рисунок 3.10 - Діаграма послідовності для перегляду документа

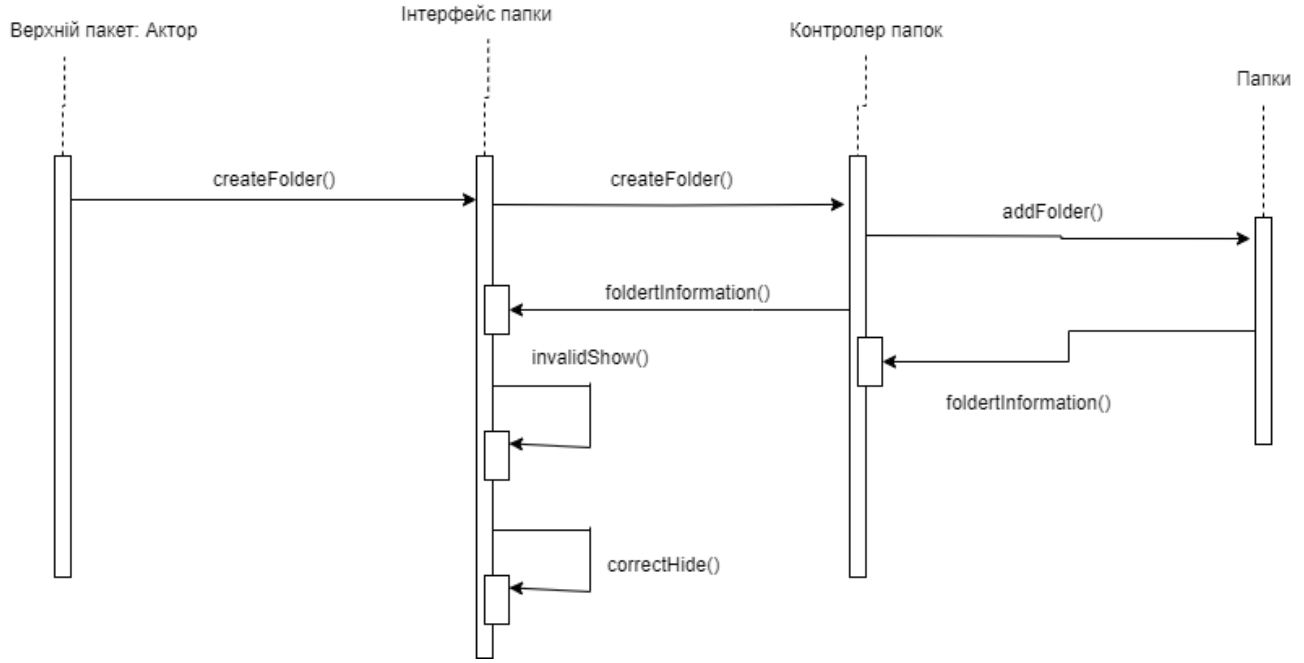


Рисунок 3.11 - Діаграма послідовності для створення папки

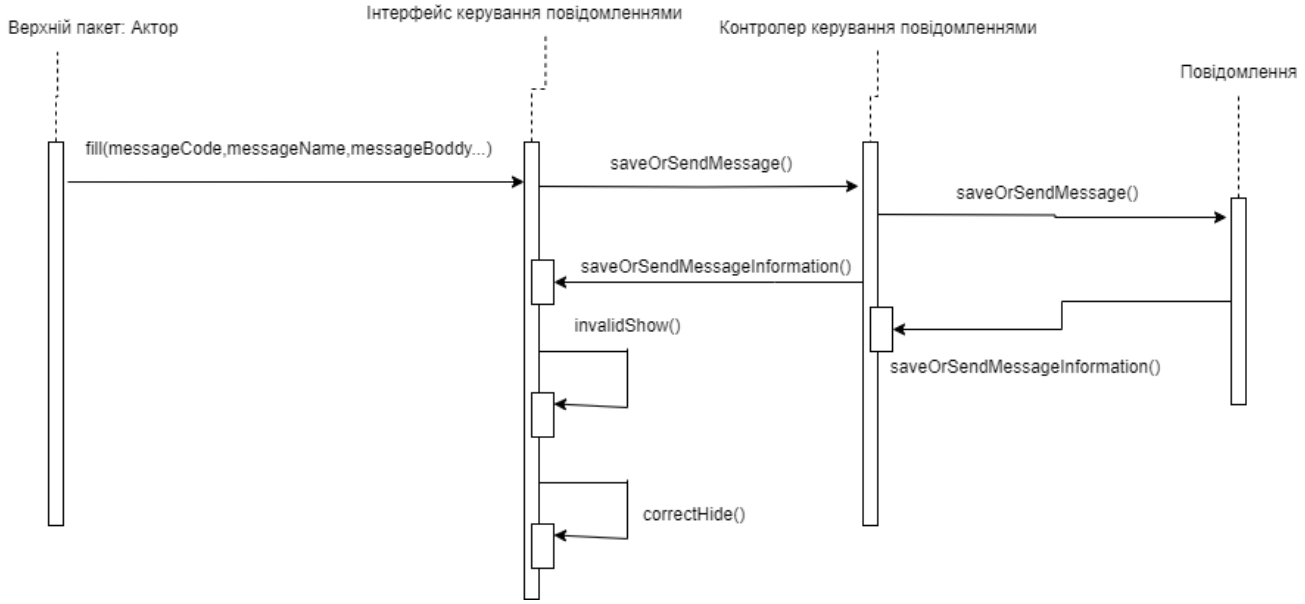


Рисунок 3.12 - Діаграма послідовності для створення та надсилання повідомлень

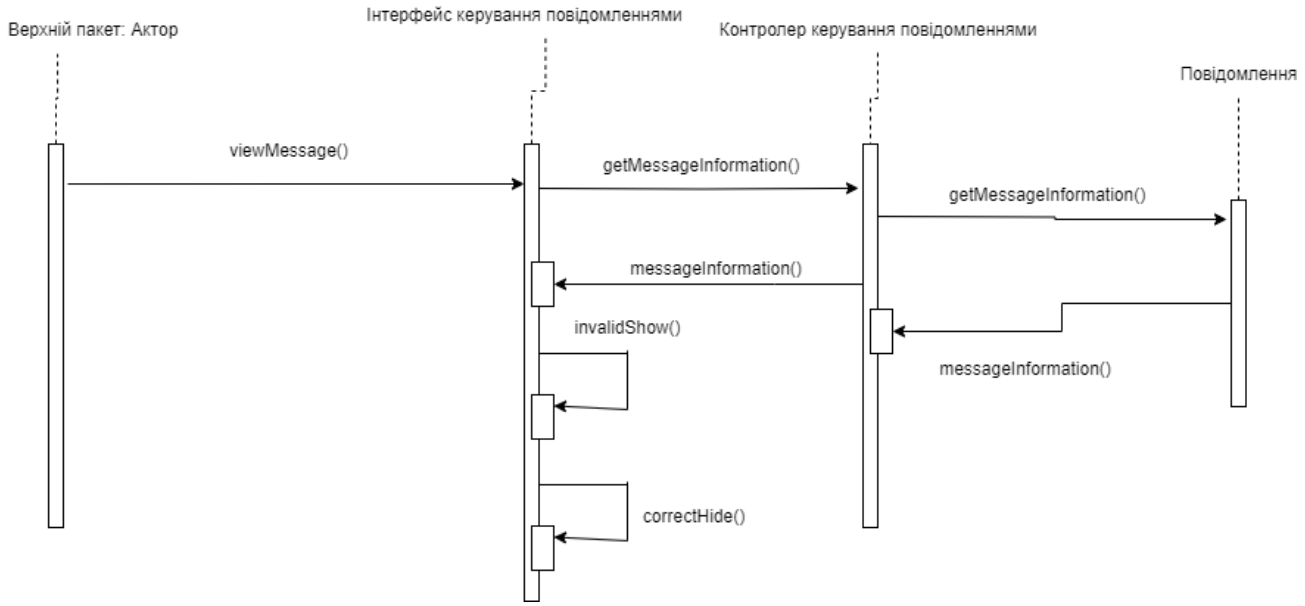


Рисунок 3.13 – Діаграма послідовності для перегляду повідомлення

Діаграма активності — це ще одна важлива діаграма поведінки в діаграмі уніфікованої мови моделювання, яка показує графічне представлення виконаного набору процедурних дій системи та розглядається як варіант діаграми стану. Вона представлена фігурами, з’єднаними стрілками.

Стрілки йдуть від початку дії до завершення та представляють послідовний порядок виконання дій. Чорні кола позначають початковий стан робочого процесу. Обведене чорне коло вказує на кінцевий стан. Заокруглені прямокутники представляють виконані дії, які описуються текстом всередині кожного прямокутника [46]. Діаграма активності для нової системи проілюстрована наступним чином (рис.3.14-3.20).

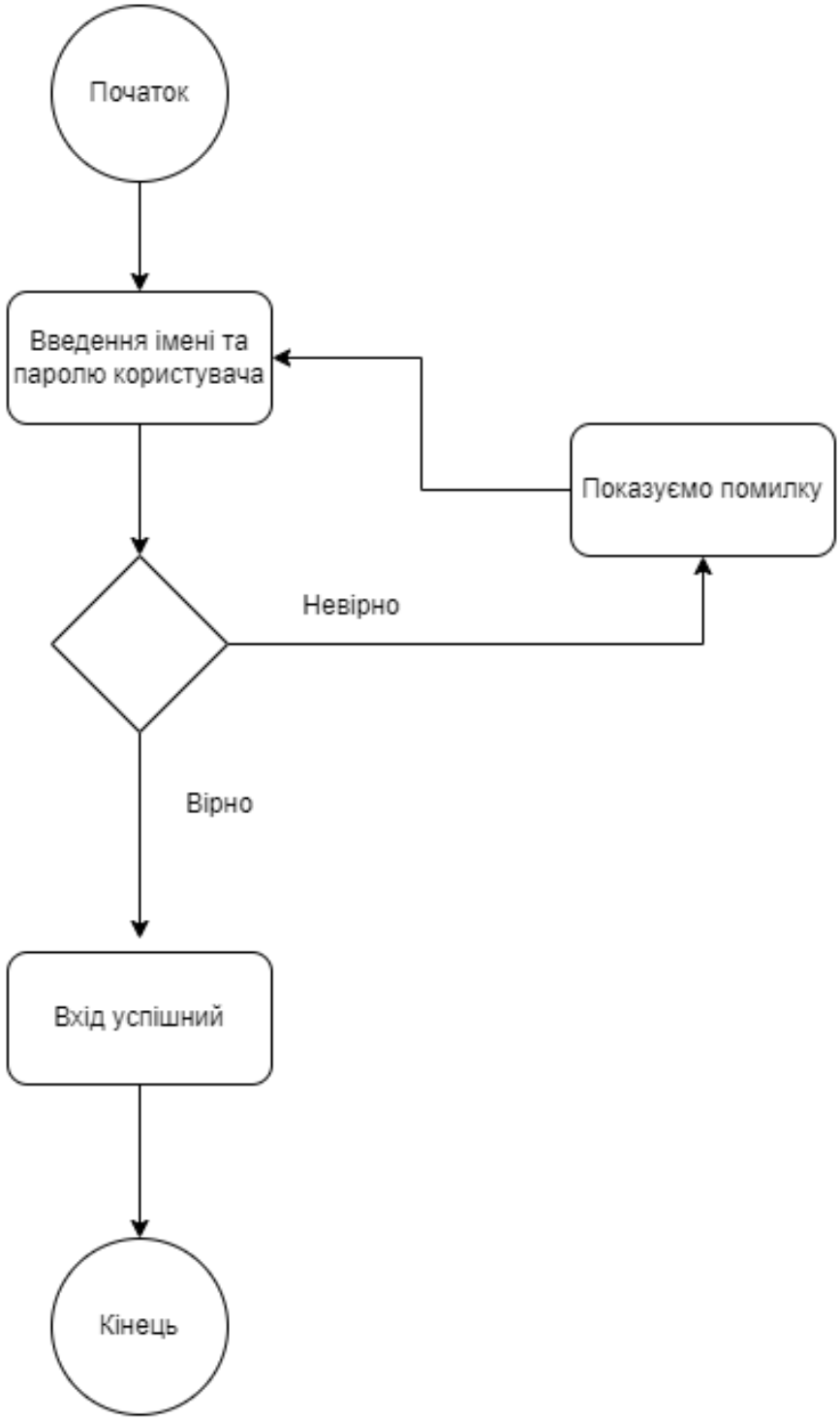


Рисунок 3.14 – Діаграма активності для входу

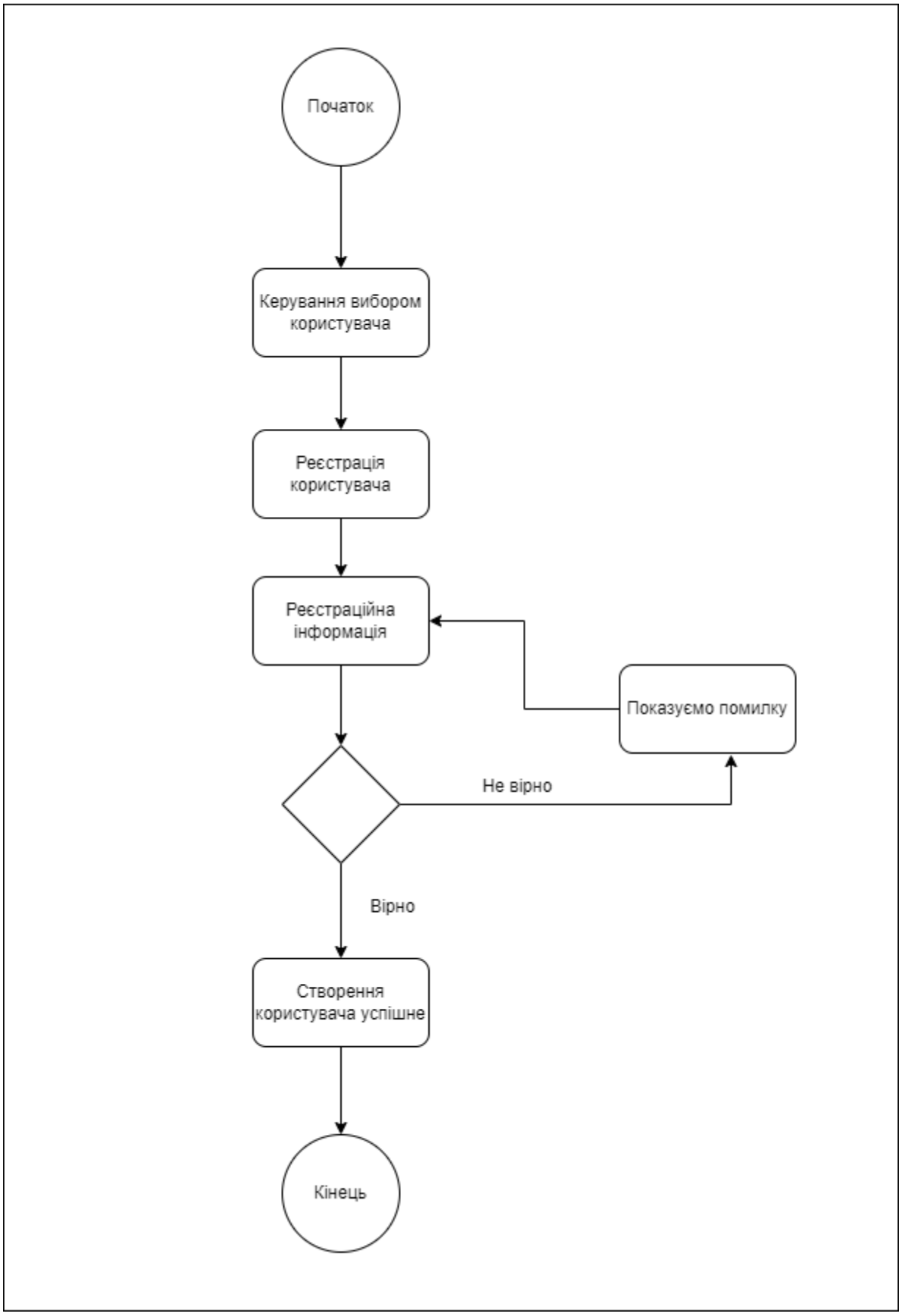

Початок
Керування вибором користувача
Реєстрація користувача
Реєстраційна інформація
Показуємо помилку
Не вірно
Вірно
Створення користувача успішне
Кінець

Рисунок 3.15 - Діаграма діяльності для створення користувача

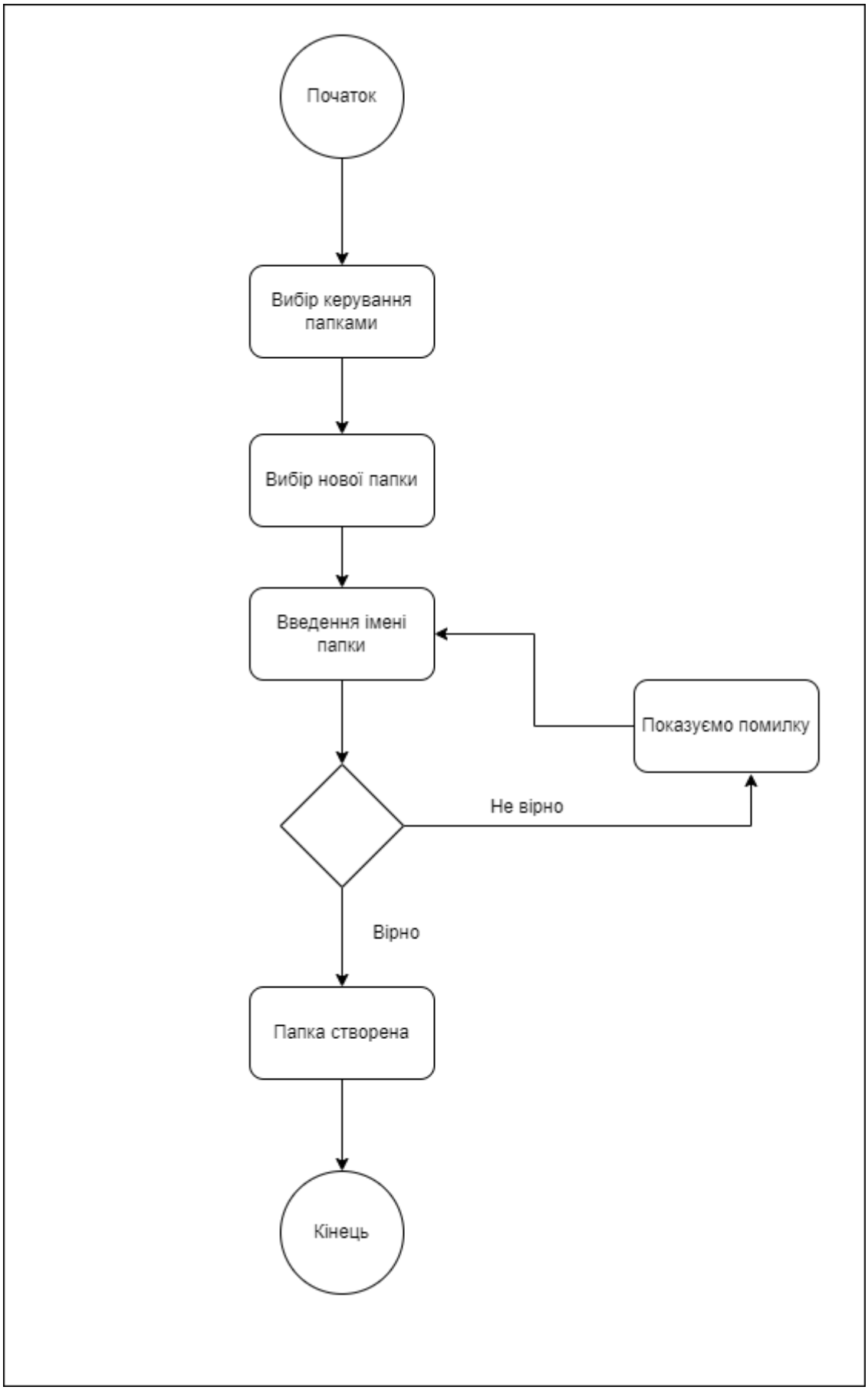

Початок
Вибір керування папками
Вибір нової папки
Введення імені папки
Показуємо помилку
Не вірно
Вірно
Папка створена
Кінець

Рисунок 3.16 - Діаграма діяльності для категорії/створення папки

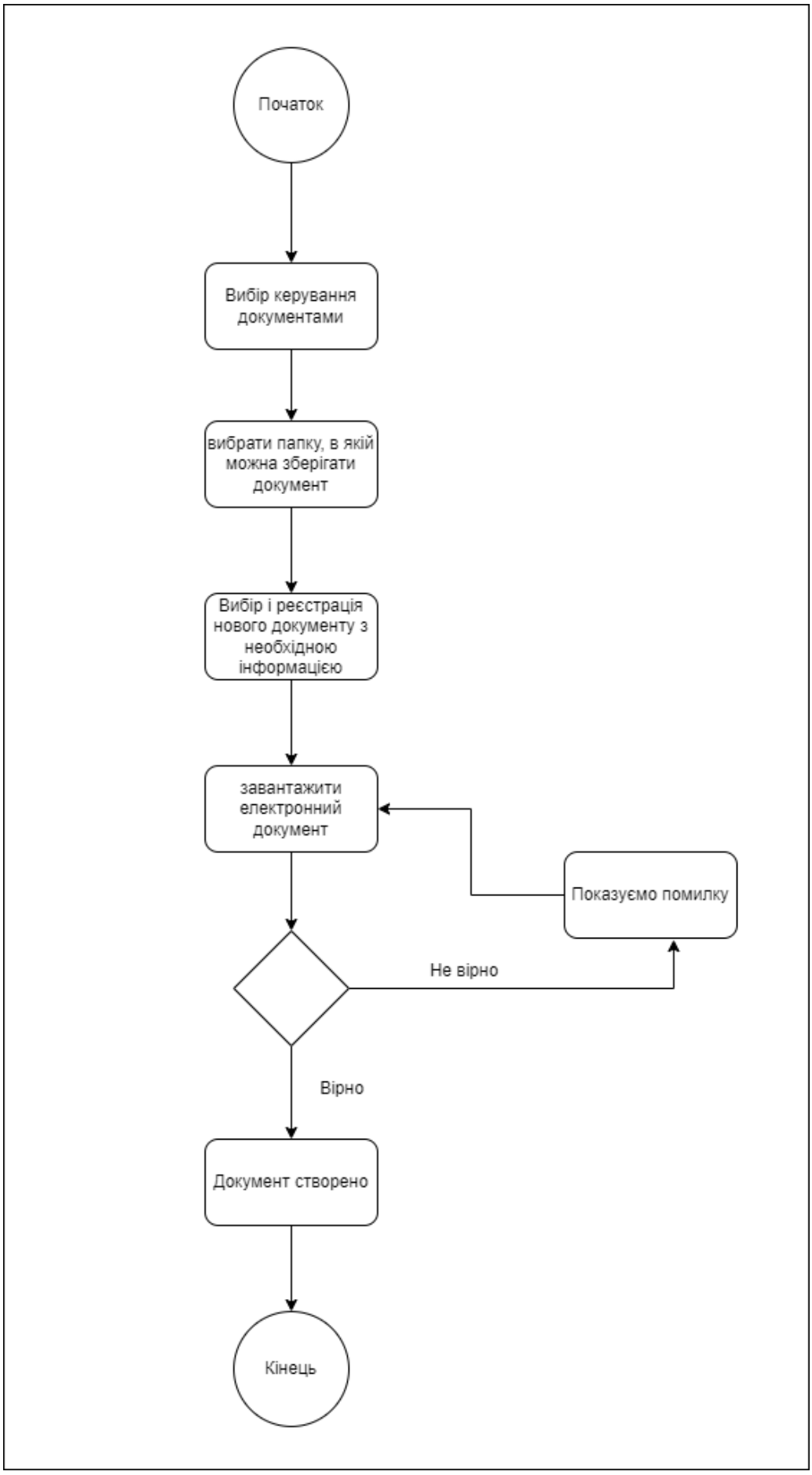

Початок
Вибір керування документами
вибрати папку, в якій можна зберігати документ
Вибір і реєстрація нового документу з необхідною інформацією
завантажити електронний документ
Показуємо помилку
Не вірно
Вірно
Документ створено
Кінець

Рисунок 3.17 - Діаграма діяльності для створення документа

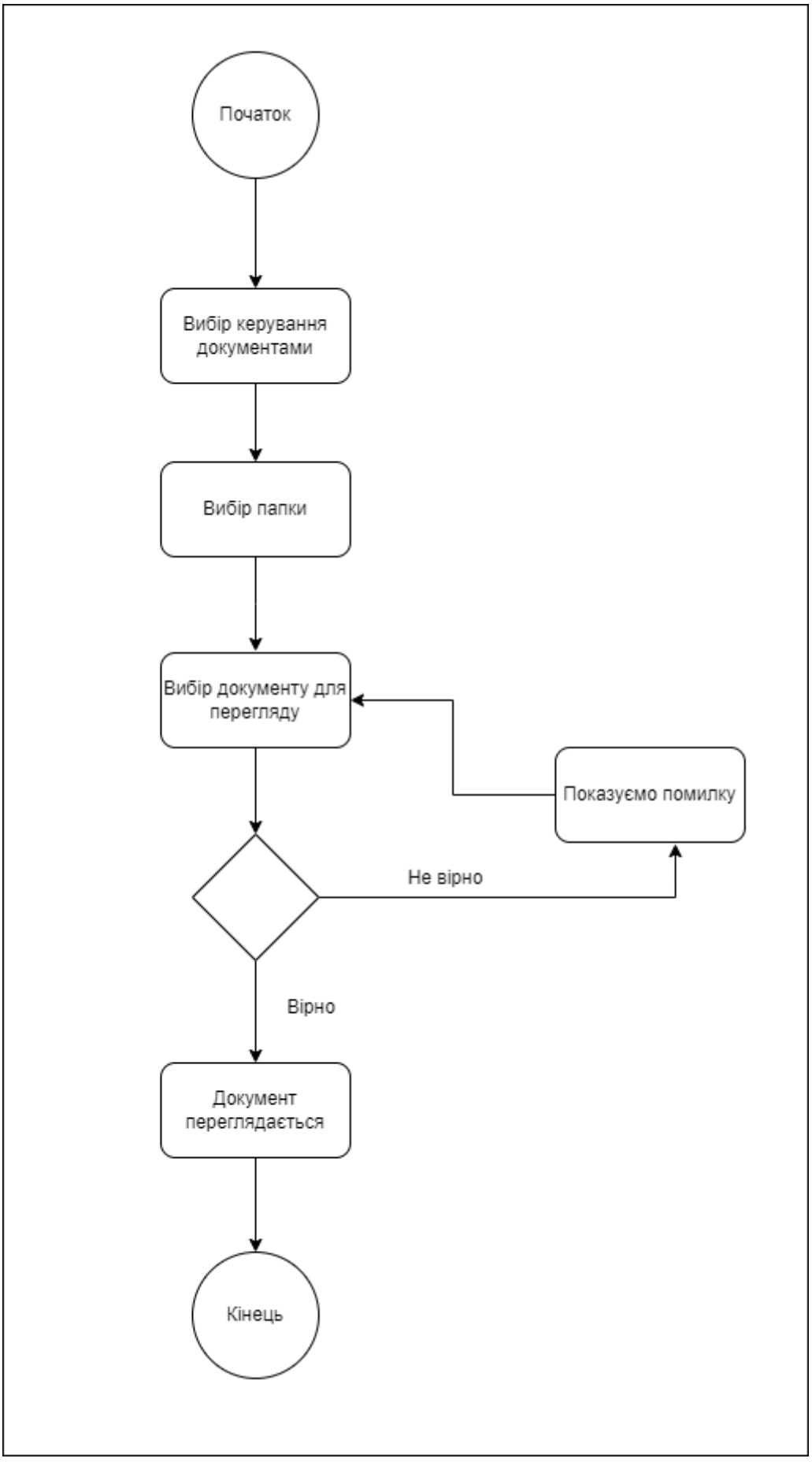

Початок
Вибір керування документами
Вибір папки
Вибір документу для перегляду
Показуємо помилку
Не вірно
Вірно
Документ переглядається
Кінець

Рисунок 3.18 - Діаграма діяльності для перегляду документа

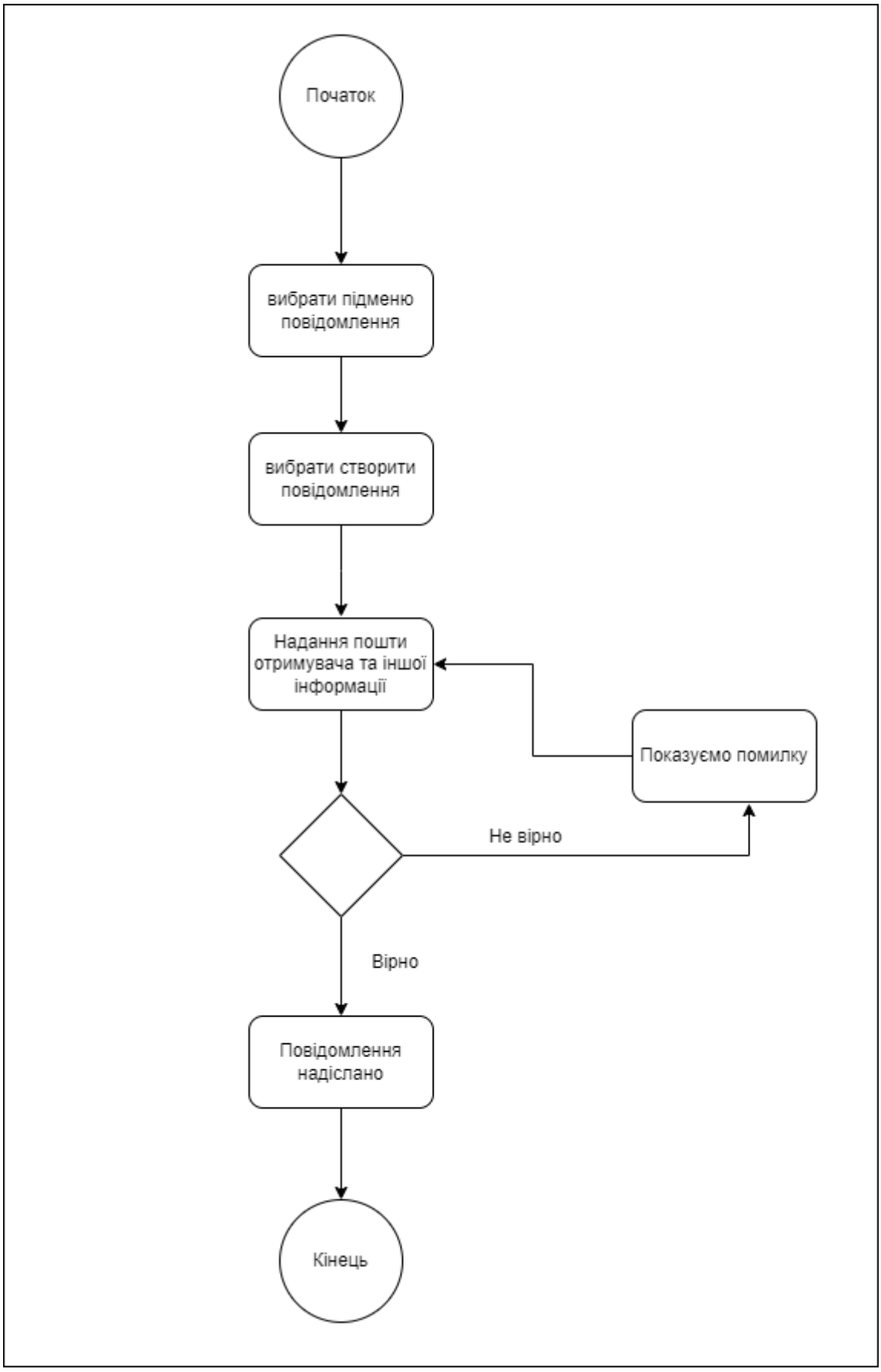

Початок
вибрати підменю повідомлення
вибрати створити повідомлення
Надання пошти отримувача та іншої інформації
Показуємо помилку
Не вірно
Вірно
Повідомлення надіслано
Кінець

Рисунок 3.19 - Діаграма діяльності для створення та надсилання повідомлення

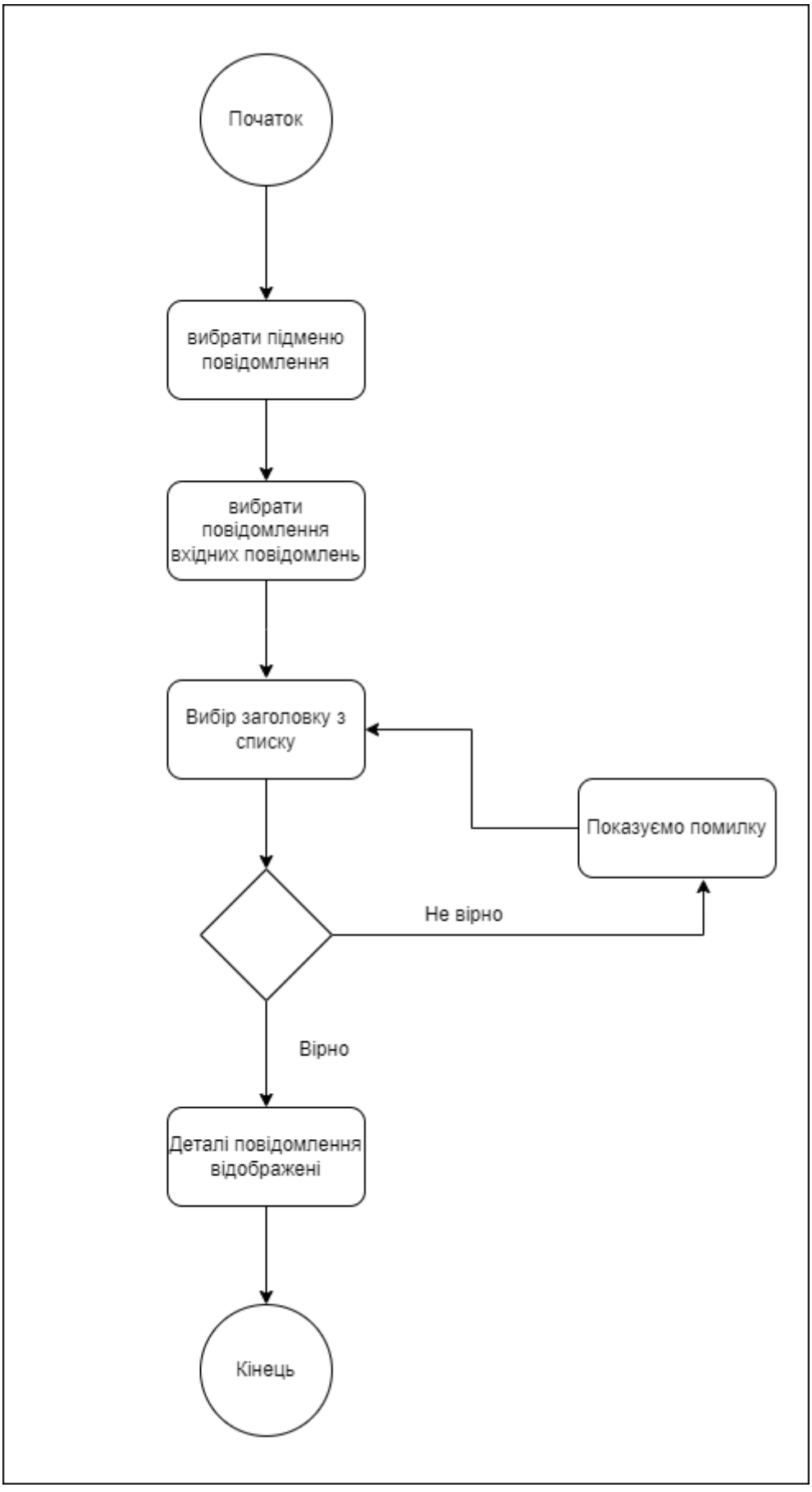

Початок
вибрати підменю повідомлення
вибрати повідомлення вхідних повідомлень
Вибір заголовку з списку
Показуємо помилку
Не вірно
Вірно
Деталі повідомлення відображені
Кінець

Рисунок 3.20 - Діаграма діяльності для перегляду повідомлень

Діаграма класів є важливою частиною уніфікованої мови моделювання, щоб показати клас об’єктів у системі та зв’язки між цими класами (рис. 3.21). Діаграми класів є найбільш корисними для ілюстрації зв’язків між класами та інтерфейсами. Узагальнення, агрегації та асоціації є цінними для відображення успадкування, складу або використання та зв’язків відповідно [46].

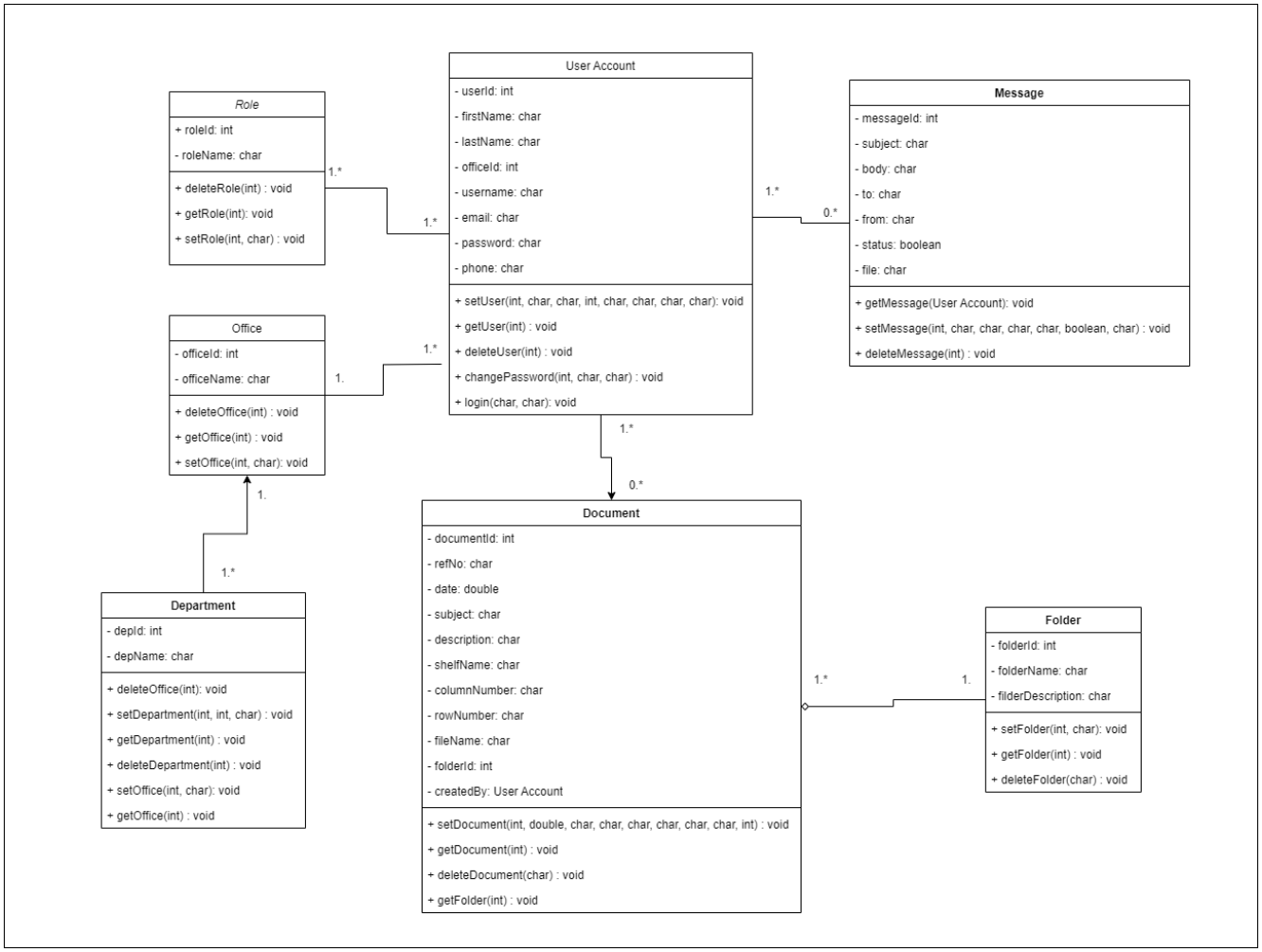


Рисунок 3.21 - Діаграма класів

3.2 Програмна реалізація інформаційної системи

У цьому розділі представлено дизайн нової системи. Проектування систем – це процес визначення архітектури, модулів, інтерфейсів та даних для системи, щоб задовольнити вказані вимоги. Документ із системними вимогами походить із попереднього етапу та розробляє конкретні технічні деталі, необхідні для системи. На етапі проектування бізнес-вимоги перетворюються на конкретні технічні вимоги [50]. На цьому етапі представлено архітектуру системи, декомпозицію підсистеми, апаратне забезпечення: відображення програмного забезпечення, інтерфейс користувача та дизайн бази даних, а кінцевим результатом цього етапу є документ проектування системи.

Система являє собою зручну автоматизовану систему в усіх аспектах. В основному він базується на документі аналізу системи, а архітектура, яка використовується для системи, є трирівневою архітектурою клієнт/сервер, де клієнт може використовувати Інтернет-браузери для доступу до звіту, наданого системою. Трирівнева веб-архітектура — це унікальна система розробки додатків веб-бази даних, яка працює навколо трирівневої моделі, що складається з нижнього рівня бази даних, середнього рівня додатків і верхнього рівня клієнта. Цей комплексний модуль трирівневої архітектури є основою для більшості веб-додатків в Інтернеті. Ця система допомагає відокремити бізнес-логіку від програми, сховища даних і бази даних [51].

Трирівнева архітектура складається з трьох компонентів, розподілених на трьох рівнях: клієнт (запитувач послуг), бізнес-логіка (обробник даних) і сервер (постачальник послуг). Є такі три компоненти рис.3.22:

1. Системний інтерфейс користувача (наприклад, служби керування сеансом, введенням тексту, діалогами та дисплеєм)

2. Управління обробкою (наприклад, розробка процесів, впровадження процесів, моніторинг процесів і послуги ресурсів процесу)

3. Управління базами даних (наприклад, служби даних і файлів)

4. Трирівневий дизайн надає системний інтерфейс користувача виключно клієнту. Він розміщує логіку програми на другому рівні, а керування базою даних – на третьому. Вибрано кілька причин для трирівневої архітектури:

5. Система працює в однорідних середовищах з правилами обробки (бізнес-правилами), які не змінюються дуже часто.

6. Відокремлення бізнес-логіки від логіки додатків мінімізує навантаження на сервер і підвищує безпеку даних.

7. Гнучке застосування матиме

8. Можна розгорнути лише одну частину

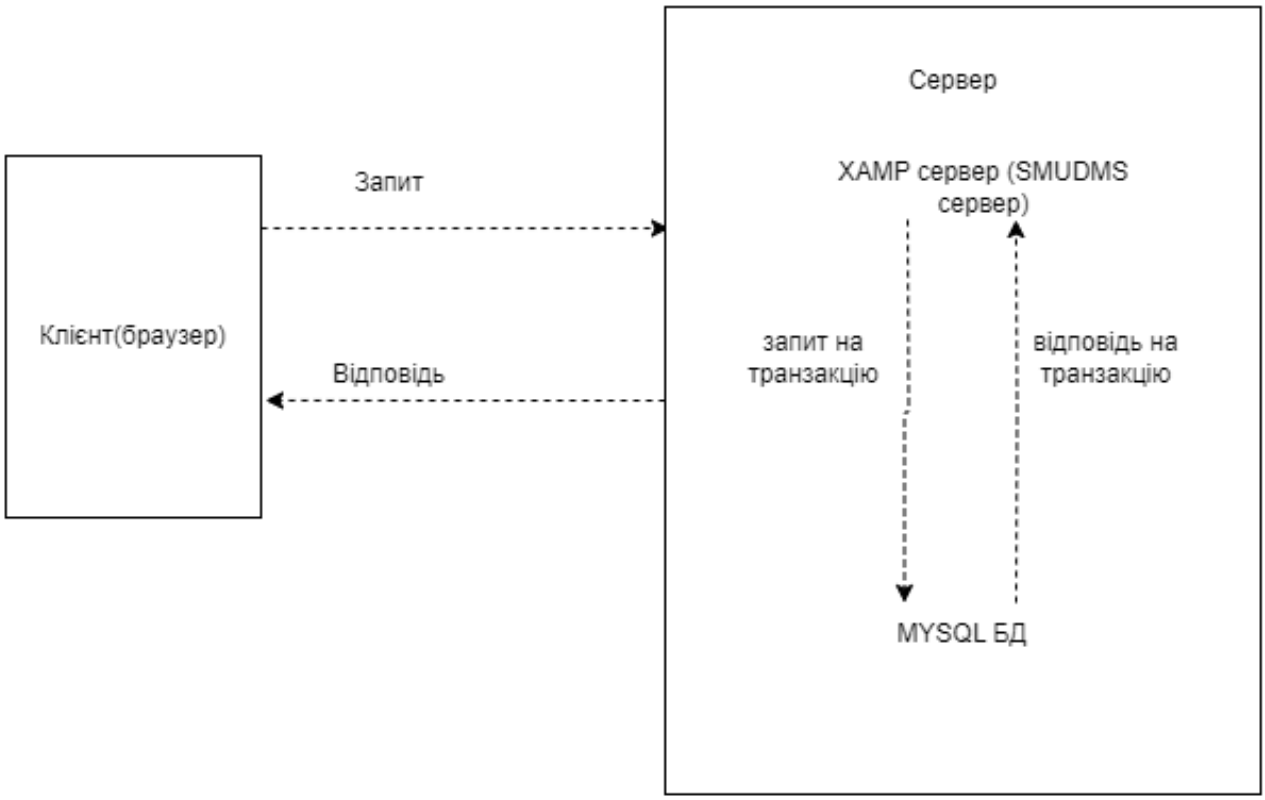


Рисунок 3.22 - Архітектура системи

Рівень даних. Зберігає дані додатків, такі як дані подій, дані звітів, документовані дані тощо. Він зберігає ці дані в системі керування реляційною базою даних (RDBMS). Середній рівень (сервер веб-додатків або веб-сервер). Реалізує бізнес-логіку, логіку контролера та логіку представлення для керування взаємодією між клієнтами програми та даними. Логіка контролера обробляє запити клієнта, такі як запити на створення та перегляд документів, створення звіту, запис документа або отримання даних із бази даних. Бізнес-правила, які забезпечуються бізнес-логікою, визначають, як клієнти можуть і як не можуть отримати доступ до даних програми та як програми обробляють дані. Пропонований клієнтський рівень. Клієнтська шина — це інтерфейс користувача програми, що містить форми введення даних і клієнтські програми. Він відображає дані для користувача. Користувачі безпосередньо взаємодіють із програмою через інтерфейс користувача. Клієнтський рівень взаємодіє з веб-сервером/сервером додатків, щоб робити запити та отримувати дані з бази даних. Потім він відображає дані для користувача.

Декомпозиції підсистеми є меншою, простішою частиною більшої системи, і це допомагає зменшити складність системи. Підсистему можна розглядати як пакети, що містять пов’язані класи або об’єкти. Ця система розбита на різні окремі підсистеми. Декомпозиції підсистеми допоможуть зменшити складність системи. Підсистеми можна розглядати як пакети, що містять пов’язані класи/об’єкти. Під час декомпозиції підсистеми системи важливо враховувати наступне питання: розділити систему на менші підсистеми з сильною згуртованістю, інструкція між підсистемами повинна мати слабкий зв'язок. Розглянута система документообігу розбивається на підсистеми. На схемі рисунку 3.23 нижче показано з’єднання між різними компонентами системи.

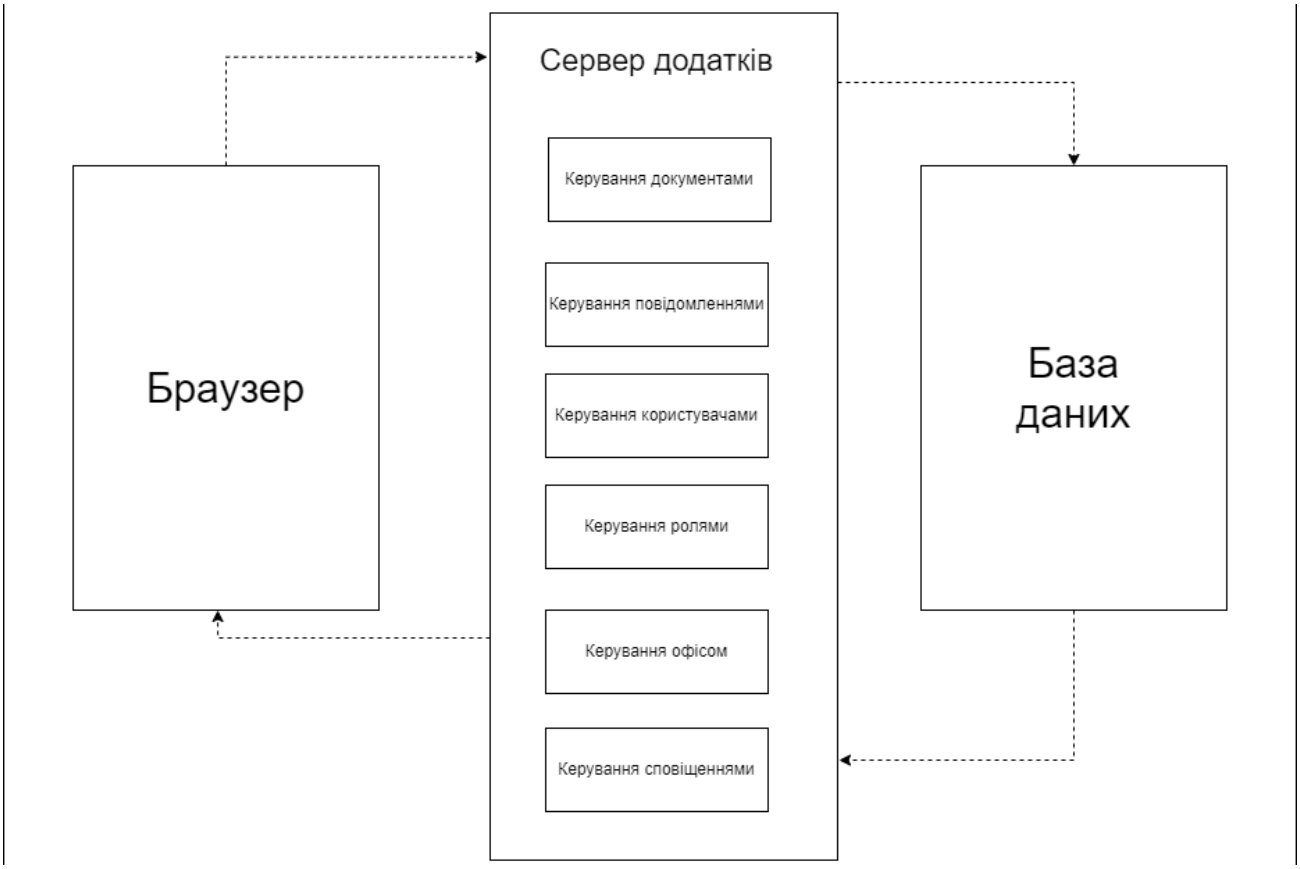


Рисунок 3.23 - Декомпозиція підсистеми

Одне з головних завдань проектування системи пов’язане з апаратним/програмним відображенням, яке стосується того, які компоненти будуть частиною якого апаратного забезпечення тощо (рис. 3.7).

1. Інтерфейс користувача та керування обробкою буде розгорнуто на машині додатків. ІСКД (Нова система) представить графічний інтерфейс користувача (GUI), який дозволяє користувачеві виконувати різні функції.

2. Клієнт/серверна архітектура системи дозволяє різним клієнтам віддалено підключатися до сервера через Інтернет/LAN.

3. Додатки системи працюватимуть на сервері додатків, підключеному до бази даних.

Розгортання діаграми — це діаграма реалізації в UML. Вона моделює фізичні вузли розгортання систем. Ящики є вузлами; компоненти можна намалювати

всередині прямокутника, тобто на вузлі, вказуючи, який вузол містить які компоненти. Вузли можуть мати підвузли, які виглядають як вкладені блоки. Це система включає три окремі вузли, як показано на наступній схемі.

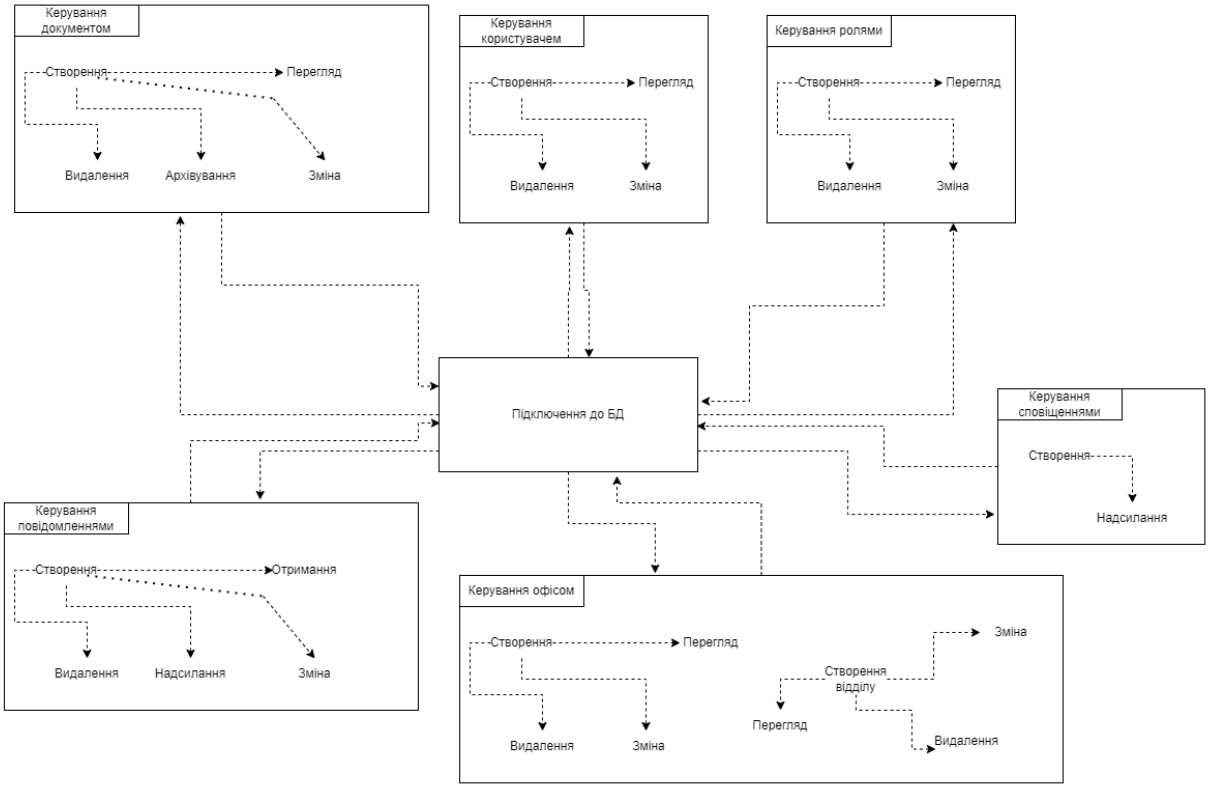


Рисунок 3.24 - Діаграма розгортання

Впровадження системи

Для реалізації цієї веб-системи використовувалися різні технології. Технологічне рішення включає PHP, HTML із MySQL як базою даних внутрішньої інтеграції. Вибір цих мов програмування базується на особливостях мов, які роблять їх дуже придатними для цього Проекту. HTML, CSS і PHP використовуються (для зовнішнього інтерфейсу) і MYSQL (для серверної частини) і обслуговуються через веб-сервер APACHE. Використання HTML і CSS, які є мовою розмітки для представлення інформації та мовою стилю відповідно,

дозволяють розробити та правильно розмістити інтерфейс користувача. Щоб увімкнути генерацію динамічного вмісту, PHP (мова веб-скриптів) використовується для створення динамічного вмісту на основі користувача системи та відповідного вмісту, що зберігається у серверній базі даних, якою керує MySQL. Веб-сервер використовується для надання веб-сторінок користувачам, коли вони потрібні, а також для інтерпретації команд скриптів PHP, які містяться на сторінці. Bootstrap — ще одна колекція інструментів, які використовуються для створення веб-додатків для цього проекту. Він містить шаблони дизайну на основі HTML і CSS для типографіки, форм, кнопок, навігації та інших компонентів інтерфейсу, а також додаткові розширення JavaScript. Причиною вибору використання цього інструменту є:

- Він сумісний з усіма основними браузерами.
- Він підтримує адаптивний дизайн: це означає, що макет веб-сторінок динамічно налаштовується, враховуючи характеристики пристрою, який використовується персональним комп’ютером, мобільним телефоном і планшетами.

3.3 Дизайн інтерфейсу користувача

Розглянемо дизайн екрана інтерфейсу користувача, який відповідає вимогам користувача та процесу системи (рис.3.25-3.30).

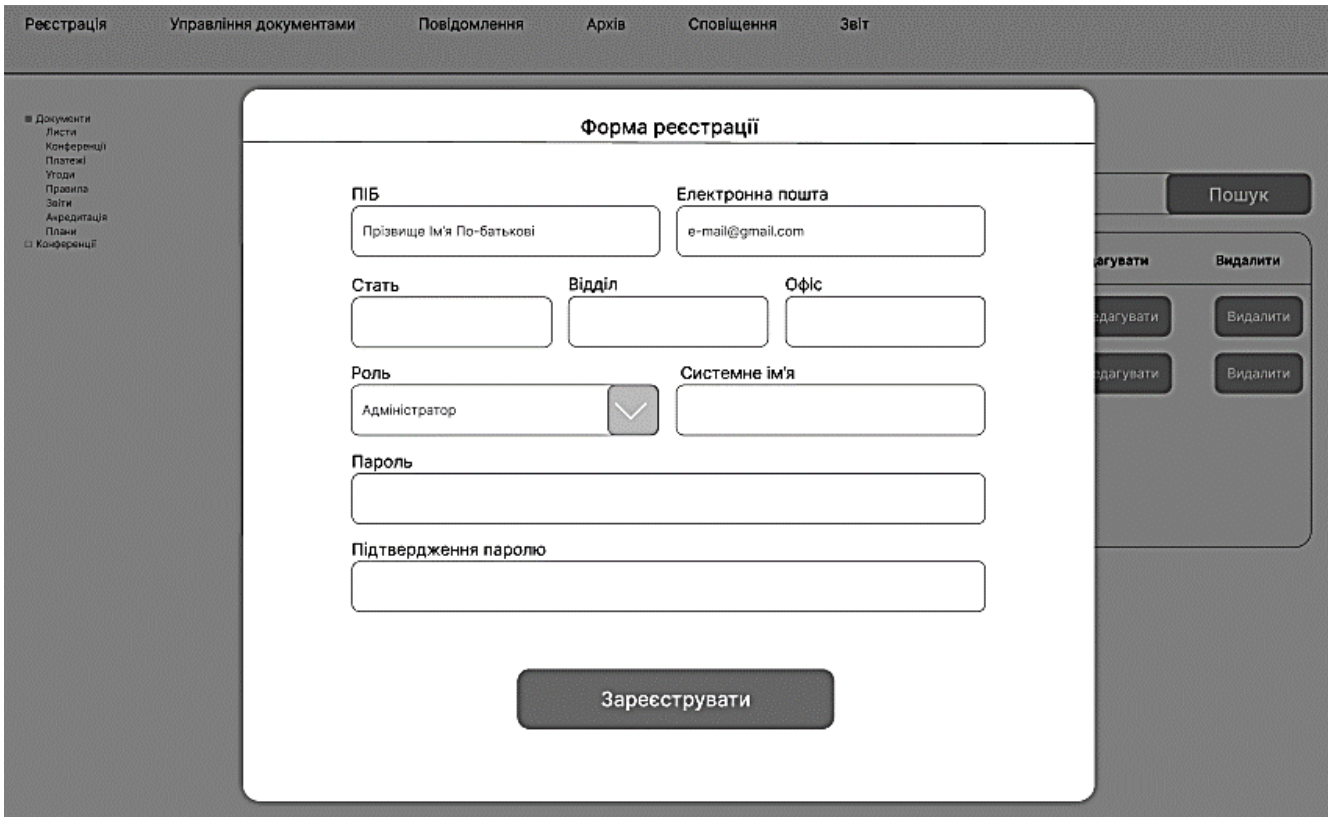


Рисунок 3.25 - Реєстрація користувача

За реєстрацію користувачів системи відповідає адміністратор. Реєстраційна форма – це список полів, які спонукають користувачів входити в нову систему.

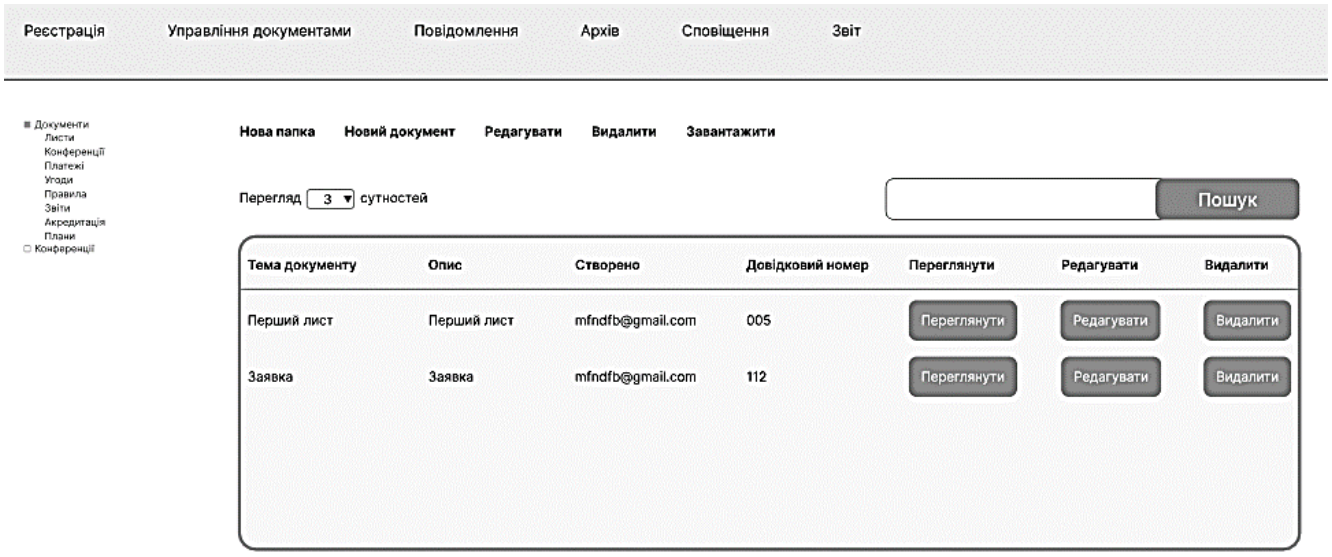


Рисунок 3.26 - Сторінка перегляду зареєстрованого користувача

Усі зареєстровані користувачі переглядаються та змінюються на одному екрані.

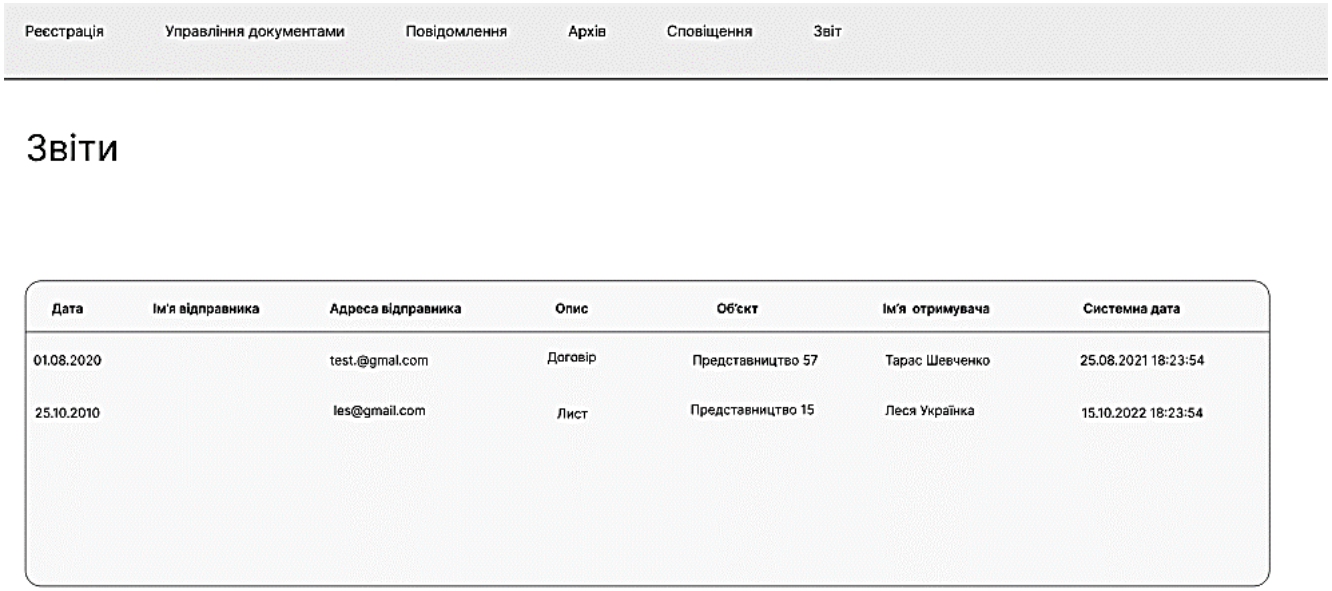


Рисунок 3.27 – Сторінка звітів відділу

Відділи зареєстровані та відображені з датою створення. У той же час зареєстровані офіси можуть бути змінені. Сторінка звіту відображатиме всі необхідні звіти.

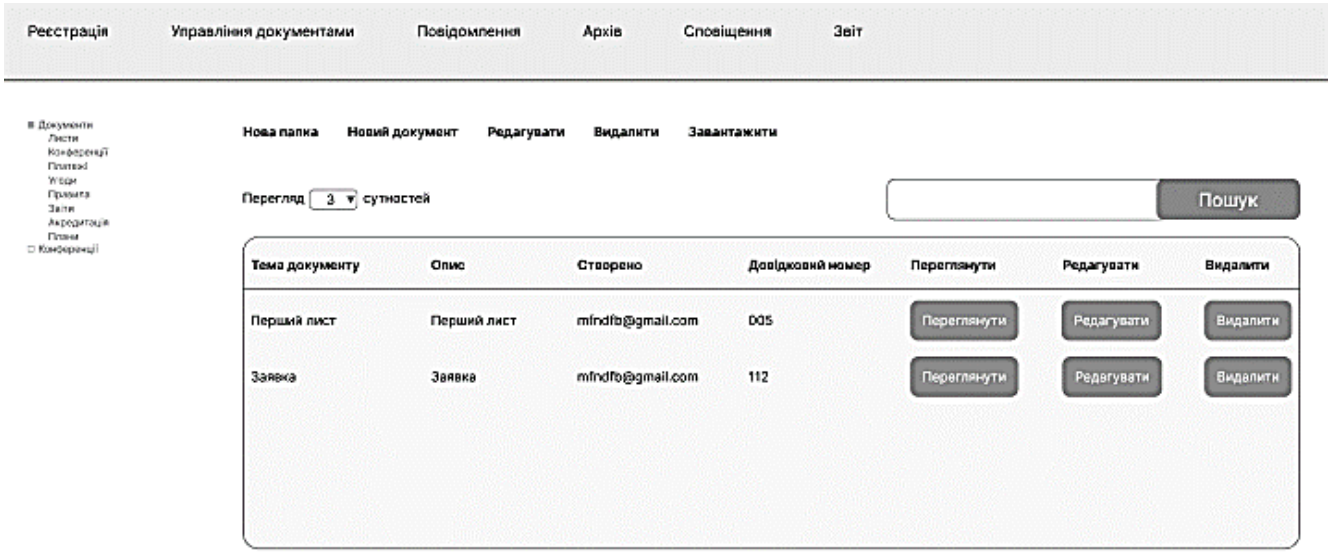


Рисунок 3.28 - Сторінка дерева документів

Система відображає структуру документа в ієрархічному форматі в дереві документів, як показано в лівій частині сторінки.

З’явиться дерево документів, коли ви натискаєте на кожен елемент.

У більшості випадків, коли ви клацаєте назву елемента, з’являється додаткова інформація, яку ви можете переглянути, редагувати, видалити та завантажити. Документ можна перевіряти та підтримувати через цей екран. Коли клацнути вкладку повідомлення, відобразиться перша сторінка.

На цій сторінці, можна побачити знімок кожного повідомлення папки "Вхідні". Також можна побачити, чи є нові повідомлення для перегляду та надіслати нові повідомлення. Після натискання кнопки створення буде розпочато створення нового повідомлення, додавання документів і надсилання його відповідній особі.

Документи створюватимуться та зберігатимуться на основі класифікаційних схем з урахуванням потреб кожного офісу.

Уже відсканований документ буде зафіксовано під час перегляду до місця призначення.

Після цього до нього буде додано додаткову інформацію метаданих, наприклад: номер документа, тип документа, тему документа, опис, номер стовпця з назвою полиці та рядок стовпця.

Цей процес метаданих є додатковими атрибутами метаданих, які описують відсканований документ. Таким чином, для подальшого використання системи це дозволяє швидше шукати документи.

Пошук документів дозволяє користувачеві системи шукати документи відповідно до категорії зробленого індексування.

Користувачі можуть шукати документи або отримати документи, вибравши ключові слова, тоді пошук документів буде здійснено відповідно до ключових слів, наданих раніше.

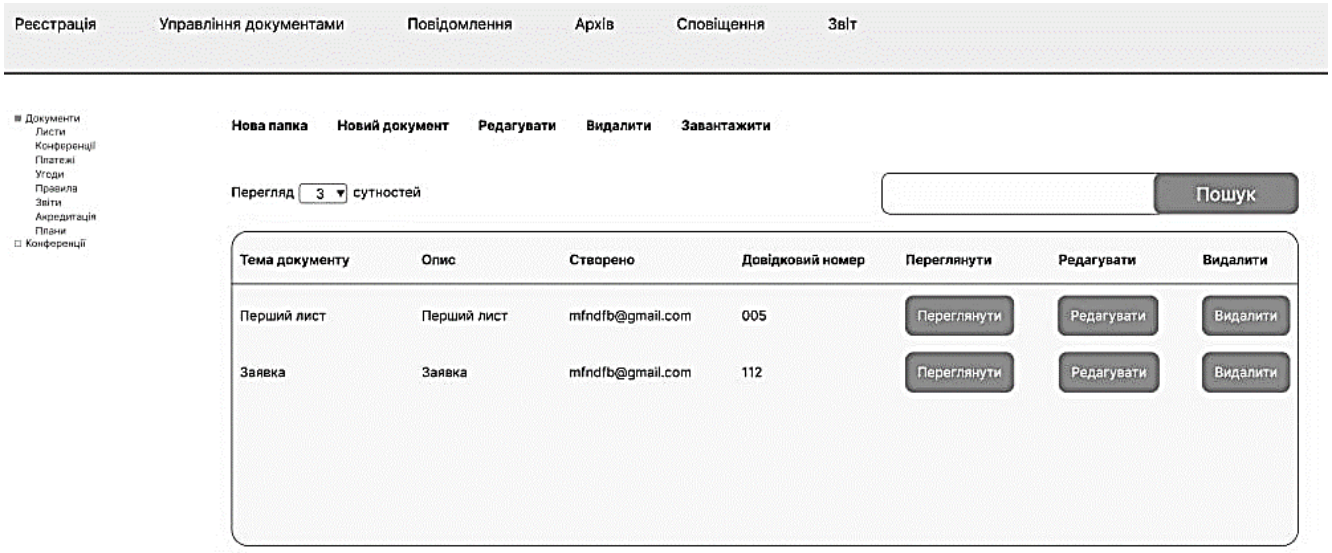


Рисунок 3.29 - Отримання документа

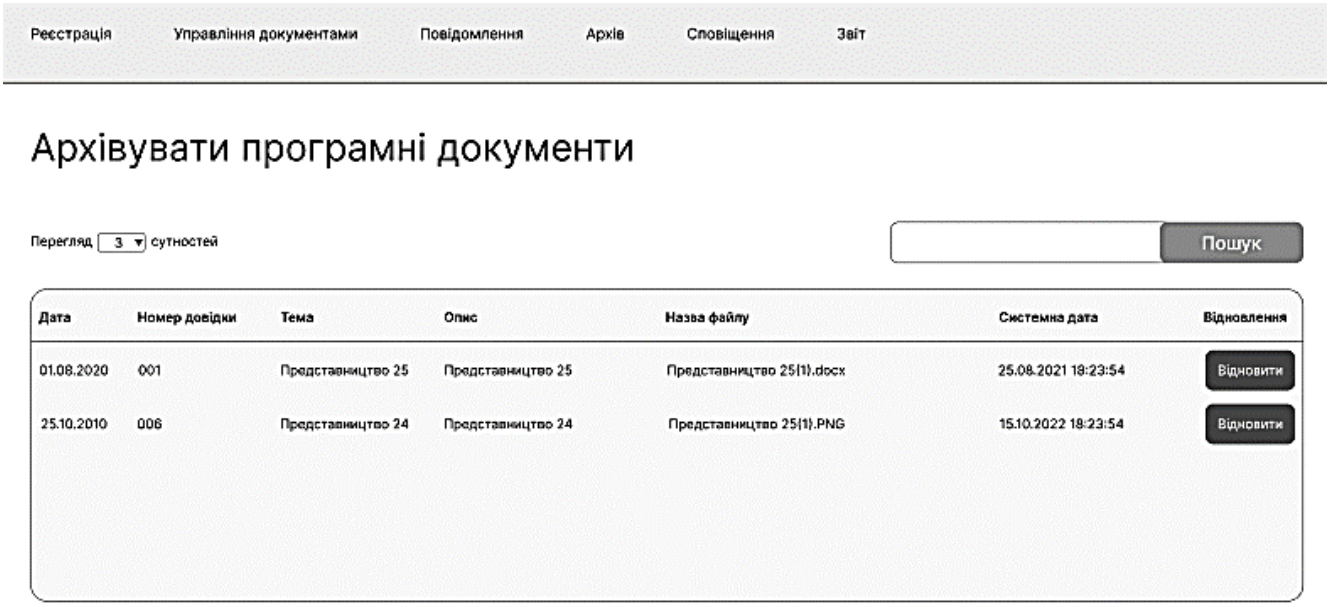


Рисунок 3.30 - Архівація документів

Цей процес архівування зберігатиме документи протягом більш тривалого періоду часу на основі їхньої визначеної політики зберігання.

Документ, який не використовувався або не посилався протягом 2 років / документ, створений протягом останніх 2 років, автоматично стає неактивним.

Коли файли стають неактивними, це не означає, що вони більше не важливі, тому система перемістить їх до іншого місця зберігання. Цей процес полегшить процес резервного копіювання за межами сайту.

3.5 Висновки

В розділі подано архітектуру інформаційної системи.

Розроблено систему електронного документообігу для підприємства. Система надає електронну форму упаковки, зберігання, пошуку та контролю документів. Перехід від поточної системи ручної роботи до системи електронного документообігу допомагає зменшити витрати, автоматизація процесів підвищує безпеку документів і мінімізує помилки. Для проектування та форматування системи використовуються різні інструменти, такі як: Hypertext Preprocessor,

Hypertext Markup Language, Cascading Style Sheet, Laravel, Bootstrap та деякі інші інструменти. Базу даних створено за допомогою системи керування базами даних MySQL.

Загалом розроблена система електронного документообігу принесе користь як працівникам підприємства, так і самому підприємства чи компанії. Це скорочує час, витрачений на пошук документів, і гарантує, що користувачі системи вчасно знайдуть важливу інформацію. Це підвищить і покращить їхню особисту продуктивність.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було розроблено інформаційну систему документообігу.

В першому розділі проаналізовано наявні інформаційні системи документообігу, виявлено переваги та недоліки.

Зроблено висновок щодо потреби розроблення локальної інформаційної технології, яка б надавала електронну форму упаковки, зберігання, пошуку та контролю документів.

Перехід від системи ручної роботи до системи електронного документообігу допоміг би зменшити витрати, автоматизував процеси та підвищив би безпеку документів з мінімізацією помилки використання ІС.

В другому розділі описано основні користувальницькі вимоги до системи.

Здійснено аналіз та специфікацію вимог до пропонованої інформаційної системи. Задано опис пропонованого рішення та наведено специфікацію системних вимог. Також в розділі подано перелік функційних вимог, нефункційних вимог, а також описано засоби розроблення ІС.

В третьому розділі подано архітектуру інформаційної системи. Розроблено систему електронного документообігу для підприємства. Система надає електронну форму упаковки, зберігання, пошуку та контролю документів. Перехід від поточної системи ручної роботи до системи електронного документообігу допомагає зменшити витрати, автоматизація процесів підвищує безпеку документів і мінімізує помилки. Для проектування та форматування системи використовуються різні інструменти, такі як: Hypertext Preprocessor, Hypertext Markup Language, Cascading Style Sheet, Laravel, Bootstrap та деякі інші інструменти. Базу даних створено за допомогою системи керування базами даних MySQL.

Розроблена інформаційна система електронного документообігу принесе користь як працівникам підприємства, так і самому підприємства чи компанії. Це скорочує час, витрачений на пошук документів, і гарантує, що користувачі системи вчасно знайдуть важливу інформацію. Це підвищить і покращить їхню особисту продуктивність.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Плескач В. Л., Рогушина Ю. В. Агентні технології: Монографія. К.: КНТЕУ, 2005. 343 с.
2. Sutton R. S., Barto A. G. Reinforcement Learning: An Introduction. The MIT Press Cambridge, Massachusetts London, England. Режим доступа: http://webdocs.cs.ualberta.ca/~sutton/book/ebook/the-book.html
3. Недашківський О.М.. Планування та проектування інформаційних систем. – Київ, 2014. – 215с.
4. Методологія інформаційних систем та баз даних: теоретичний і практичний підходи : навч. посібник / уклад. Ю.О. Ушенко, М.Л. Ковальчук, М.С. Гавриляк, А.Л. Негрич. – Чернівці : Чернівецький нац. ун-т ім. Ю. Федьковича, 2021. 240 с. ISBN 978-966-423-641-3
5. Інформаційні системи і технології в управлінні. Конспект лекцій для студентів СО “Магістр” заочної форми навчання спеціальності 073 Менеджмент освітньо-професійна програми “Менеджмент у судовій сфері” галузі знань 07 Управління та адміністрування. Вінниця: ДонНУ імені Василя Стуса, 2021. 160 . Методологічні основи створення інформаційного середовища управління науковими дослідженнями [Текст] : монографія. / А.О. Білощицький, П.П. Лізунов, О.Ю. Кучанський, Ю.В. Андрашко, О.В. Миронов, С.В. Білощицька – К: КНУБА, 2017. – 148 с.
6. Мищишин О.Я. Опорний конспект лекцій з дисципліни “Ефективність інформаційних систем” з освітньо-кваліфікаційного рівня “Магістр” для спеціальності “Інформаційні технології в бізнесі ”. Львів:, 2017. 98 с.
7. Антоненко В. М., Мамченко С. Д., Рогушина Ю. В. Сучасні інформаційні системи і технології: управління знаннями: навч. посібник. Ірпінь: Нац. університет ДПС України, 2016. 212 с.

8. Гомонай-Стрижко М. В., Якімцов В. В. Інформаційні системи та технології на підприємстві: конспект лекцій. Львів: НЛТУ, 2014. 200 с. URL: http://ep.nltu.edu.ua/images/Kafedra_EP/Kafedra_EP_PDFs/kl_isitp.pdf

9. Федорова М. С. Конспект лекцій з дисципліни “Інформаційні системи та технології на підприємстві”. Херсон, 2015. 158 с.

10. Інформаційні системи: навч. посібник / за наук. ред. Н. В. Морзе. Івано-Франківськ: “Лілея НВ”, 2015. 384 с.

11. J.A. Ochoa. The Modern Project Manager An Introduction to SCRUM and an Agile Approach to Business Strategy, Project Management, and Leadership.2021.

12. Liam Taylor. The Scrum Master’s Path. 2 books in 1. A Guide for Servant Leaders Navigating Growth and Change Using SCRUM and the Agile Project Management. 2021.

13. Sam Cooper. 8 Books in 1: Agile Project Management, Lean Analytics, Enterprise, Six Sigma, Startup, Kaizen, Kanban, Scrum. 2021.

14. Kori Brummitt. Agile Project Management. Managing A Projects More Effectively With Agile Methods, Scrum, Kanban, And Extreme Programming: Agile Guide To The Planning Processes. 2021.

15. Quinn Corrie . Agile Project Management Tutorial. Innovative Methodology To Manage Projects With Scrum, Agile, Kanban, And Extreme Programming: Agile Project Management. 2021.

16. C. Anderson, “Is document control really that important”, Bizmanualz OnPolic. Retrieved September 23, 2012, From http://www.onpolicy. com/2010-12/is-document-control-really-that important.html.

17. M. J. Noyes, J. K. Garland, “Computer- vs. Paper-Based Tasks: Are They Equivalent?”, vol. 51, pp.1352–1375, 2008, https://doi.org/10.1080/00140130802170387

18. H. A. Abdulkadhim, M. Bahari, A. Bakri and W. A. Ismail, “Research framework of electronic document management systems (EDMS) implementation

process in government", *Journal of Theoretical and Applied Information Technology*. 2015. https://oarep.usim.edu.my/jspui/handle/123456789/2128.

19. B.E. Asogwa,"The challenge of managing electronic records in developing countries: Implications for records managers in sub-Saharan Africa", Records Management Journal, Vol. 22 No. 3, pp.198-211, 2012. https://doi.org/10.1108/09565691211283156

20. M. Ostroukh, S. Krasnyanskiy, S. Karpushkin, A. Obukhov,"Development of automated control system for university research projects", Middle-East Journal of Scientific Research, Vol.20, no.12, pp.1780-1784, 2014 DOI: 10.5829/idosi.mejsr.2014.20.12.21091

21. H. Adamu, "Management of Electronic Records in Federal Universities in North Western States of Nigeria": Being a Thesis

22. Design and Implementation of a Web-based Document Management System

23. 52 Volume 15 (2023), Issue 2

24. Submitted to the Department of Library and Information Sciences for the Award of a Master in Library and Information Sciences, 2016, 2016. Available at SSRN: https://ssrn.com/abstract=2940967 or http://dx.doi.org/10.2139/ssrn.2940967 . H. Adamu, Management of Electronic Records in Federal Universities in North Western States of Nigeria.

25. B. Haval, K. J. Abdulrahman, A.R. Abrahim, "Student Performance Predictions Using Knowledge Discovery Database and Data Mining, DPU Students Records as Sample", *Academic Journal of Nawroz University*, vol.10, no.3, pp.121-127,2021.

26. A. E. Nwaomah, "Political Factors' Influence on Students' Records Management Effectiveness in The Nigerian University System" European Journal of Research and Reflection in Management Sciences Vol, 3(2).29-41,2015.

27. R. Vella, I. Gatt, C.Caruana, C. Zammit, “Innovative pedagogies in visual art education and sustainable development - an asynchronous continuous professional development course”, In Ioannidou, M. (Ed.), The touch of art: teacher training for sustainability through the visual arts (pp tba). Cyprus: Frederick University, 2022.

28. A. Adam, “Implementing Electronic Document and Record Management Systems” (1st ed.). Auerbach Publications., 2007. https://doi.org/10.1201/9780849380600.

29. [23] F. Marir, “Document management systems from current capabilities towards intelligent information retrieval”: An overview/H. Zantout, F. Marir. *International Journal of Information Management,* Vol.19, no.6, 1999.

30. L.A. Joia, “Large-scale reengineering in project documentation and workflow at engineering consultancy companies”, *International Journal of Information Management,* vol.18, no.3, pp. 215-24,1998. https://doi.org/10.1016/S0268-4012(98)00006-1.

31. M. Grange, M. Scott, “An Investigation into the effect of poor end user involvement on electronic document management system (EDMS) implementation”, 15th Annual Conference In: UK Academy for Information Systems (UKAIS), Oriel College, Oxford, 2010.

32. S.H. Hwang, S.K. Jung, S.J. Kang, H.S. Cha, S.H. Chung, D.H. Lee, “Development of a document management system for the standardization of clinical laboratory documents”, Annals of laboratory medicine, vol.33, no.6, pp. 441-448, 2013. http://dx.doi.org/10.3343/alm.2013.33.6.441

33. A. Yaldır, L. O. Polat,” Electronic document management system selection with multi-criteria decision-making technique”, Mehmet Akif Ersoy Universitesi Sosyal Bilimler Enstitusu Dergisi. 2016;8(14):88-108. https://doi.org/10.20875/sb.56729

34. S.R. Wicaksono, “Implementing Collaborative Document Management System in Higher Education Environment”, *Prosiding SNATIKA*. 2015 vol.3, pp.22-25, 2015.

35. P. Singh, J. Klobas, K Anderson, “Information seeking behaviour of electronic records management systems (ERMS) users: implications for records management practices” *Human IT*. Vol. 9, no.1, pp.135-181, 2007. http://hdl.handle.net/20.500.11937/33339

36. H. Alshibly, R. Chiong, Y. Bao, “Investigating the critical success factors for implementing electronic document management systems in governments: evidence from Jordan”, *Information Systems Management*, vol.33, no.4, pp. 287-301, 2016.

37. P. Vevaina, “Factors affecting the implementation of enterprise systems within government organizations in New Zealand” (Master’s thesis). Auckland University of Technology, New Zealand, 2007. http://hdl.handle.net/10292/287.

38. J. McLeod, and C. Hare, "Development of RMJ: A mirror of the development of the profession and discipline of records management". Records Management Journal, vol.20, no.1, pp. 9-40, 2010.

39. A. Adam, “Implementing electronic document and record management systems”, 1st Edn, New York: Auerbach Publications 2008.

40. S. Hung, T. King-Zoo, C. Chia-Ming, K. Ching-De, “User acceptance of intergovernmental 567 services: An example of electronic document management system”, *Government Information Quarterly*, 568 vol.26, no.4, pp. 387-397, 2009.

41. Cimtech. (2007). Managing information and documents: The definitive guide (18th ed.). Hatfield, UK: Information Age.

42. Wilson Nwankwo, Akinola S. Olayinka, Benjamin Uchenna Benson, "X-ray Cargo Scanning and Risk Management in Trade Facilitation: Analysis & Model of an Online Imaging and Documentation Management System", *International Journal of*

Modern Education and Computer Science(IJMECS), Vol.11, No.5, pp. 10-23, 2019.DOI: 10.5815/ijmecs.2019.05.02.

43. B.C. Bjoerk, P. Huovila & S. Hult, "Integrated construction project document management," *in Proceedings of the EuropIA'93 conference, M. Behesti & K. Zreik, eds. Delft: Elsevier*, pp. 135-146, 21-24, 1993.

44. B.C. Bjoerk, "The RATAS project – a n example of co-operation between industry and research towards computer integrated construction*", ASCE Journal of Computing in Civil Engineering*, Vol. 8, No. 4, pp.401-419,1994.

45. Z. Turk Z., B.C. Bjoerk, K. Johansson and K. Svensson, "Document management systems as an essential step towards CIC", CIB W78 *Workshop on Computer Integrated Construction, Helsinki:* VTT, 22-24 1994.

46. P.H. Debras, Y. Rezgui, P. Le Ber, & C. Chaplais, "Projet DOCCIME: Système logiciel pour l'intégration de l'ingénierie documentaire dans le processus de conceptionRapport final", *Internal Report, Sophia-Antipolis*: CSTB, 1994.

47. Y. Rezgui, A. Brown, G. Cooper, G. Aouad, J. Kirkham and P. Brandon, "An integrated framework for evolving construction models", unpublished Y. Rezgui, "Integration des systèmes d'informations techniques pour l'exploitation des ouvrages", PhD thesis,Paris: Ecole Nationale des Ponts et Chaussées, 1994.

48. F. Ozdemirci, "Development and implementation of the document management and archiving system (BEYAS) for universities: an example of cooperation", pp. 225-235, 2008.

49. P.A. Akashah, R. Syamsul, K. Jussof, E. Christon, "Electronic Document Management System", *World Applied Science Journal, (Special issue on Computer Application and Knowledge management)*, vol 12, 55-58.

50. M. K. Kodmelwar, M. Agarkar, A., Borle, A Deshmukh and M. Bhagat, "Document Management System with Enhanced Security", *IOSR Journal of Computer Engineering (IOSRJCE*), Vol.1, no.2, pp.18-23, 2012.www.iosrjournals.org.

51. F. Faiqunisa F, E. Nugroho and P. I. Santosa, “A Model of Electronic Document Management System for Limited Partnership.”, *Journal of Telematics and Informatics.*, vol.1, no. 2, pp. 69-79, 2013 DOI:10.12928/jti.v1i2.78-88.

52. M. Adiyan, “Analisis Kesuksesan Penerapan Sistem E-Office Di Universitas Airlangga Menggunakan Model Delone Dan Mclean (Doctoral dissertation, UNIVERSITAS AIRLANGGA), 2015. http://repository.unair.ac.id/id/eprint/27978.

53. Srirahayu, O. D. P. (2018). Implementation of Electronic Document Management to Support World-class University: A Case Study in Airlangga University, *KnE Social Sciences*, vol.3, no.11, pp. 1205–1213. https://doi.org/10.18502/kss.v3i11.2841.

54. M. Başıbüyük, A. Ergüzen, “Electronic document management system for Kırıkkale University”, Unified Journal of Computer Design and Implementation of a Web-based Document Management System Volume 15 (2023), Issue 2 53 *Science Research.* Vol.1, no.2, pp. 008-15, 2015. www.unifiedjournals.org.

55. Ajala, O.S., “Design and Implementation of an Improved Electronic Document Management System (ENCODOC) (Doctoral dissertation, NationaL Open University of Nigeria), 2015.

56. P. J.M. Estrera, “Electronic Document Management System for Higher Education Institution” Vol. 2, no. 5, pp. 549-556, 2017. www.ijisrt.com.

57. H. S. Ahmad, I.M. Bazlamit,M. D. Ayoush “Investigation of document management systems in small size construction companies in Jordan”, *Procedia Engineering*, vol.182, pp. 3-9, 2017. https://doi.org/10.1016/j.proeng.2017.03.101.

58. R. Andriansyah, F. Elmi, “Analysis of the Effect of Electronic Document Management System, Organizational Commitment and Work Satisfaction on Employee Performance Part Time”, Graha Fortuna Purnama. *International Journal of Innovative Science and Research Technology*, vol.5, no.8, pp. 944-52, 2020 https://ijisrt.com/assets/upload/files/IJISRT20AUG554.pdf

59. N. Nursiyanto, S, karnila, R. Ronaldo, “Electronic document for research and service with web-based in higher education” (case study: lp4m iib darmajaya). J*urnal tam (technology acceptance model)*, Vol.11, no.2, pp.94-99,2020.

60. Santos, J.C., Tarrit, K., Mirakhorli, M. A catalog of security architecture weaknesses. Proceedings of the 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), Gothenburg, Sweden, 5–7 April 2017, IEEE: Piscataway, NJ, USA, 2017, pp. 220–223.

61. Ouma, W.Z., Pogacar, K., Grotewold, E. Topological and statistical analyses of gene regulatory networks reveal unifying yet quantitatively different emergent properties. PLoS Comput. Biol. 2018, 14, e1006098.

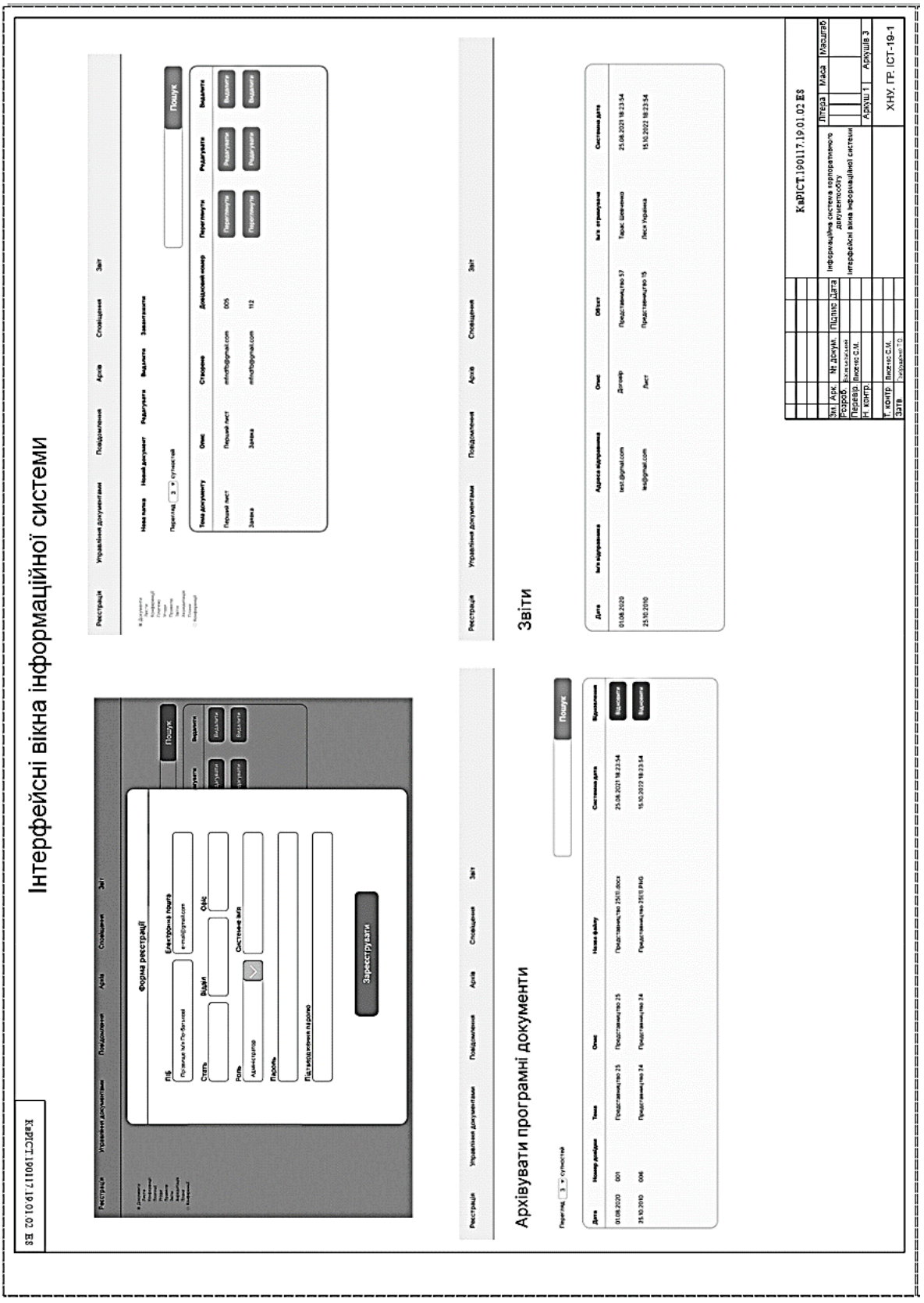
62. Dissanayake, N., Jayatilaka, A., Zahedi, M., Babar, M.A. Software security patch management—A systematic literature review of challenges, approaches, tools and practices. Inf. Softw. Technol. 2020. 144, 106771.

63. De Vita, F., Bruneo, D., Das, S.K. On the use of a full stack hardware/software infrastructure for sensor data fusion and fault prediction in industry 4.0. Pattern Recognit. Lett. 2020. 138, 30–37.

Додаток А

(обов’язковий)

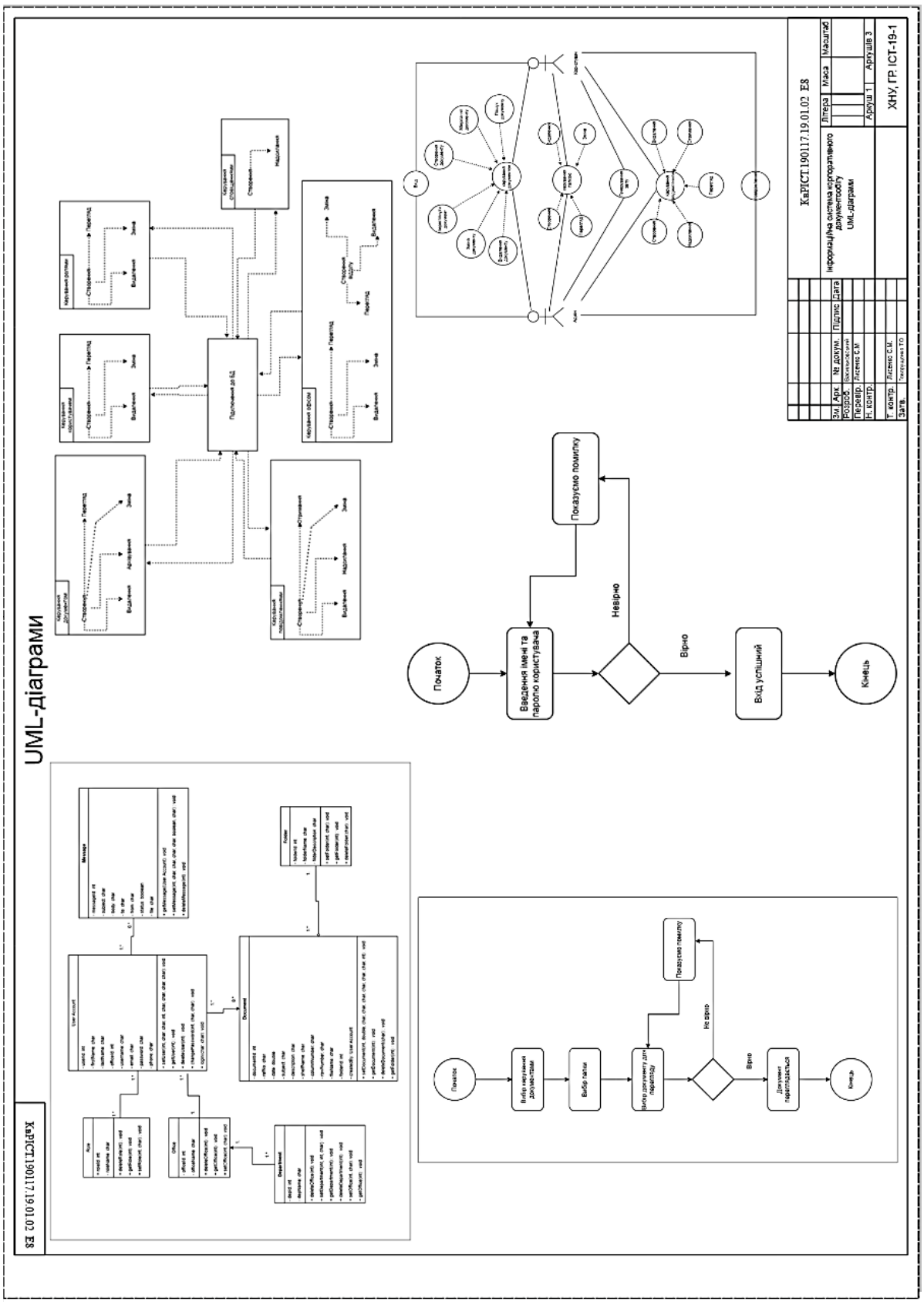
Копія креслення «Інтерфейсні вікна інформаційної системи»



Додаток Б

(обов’язковий)

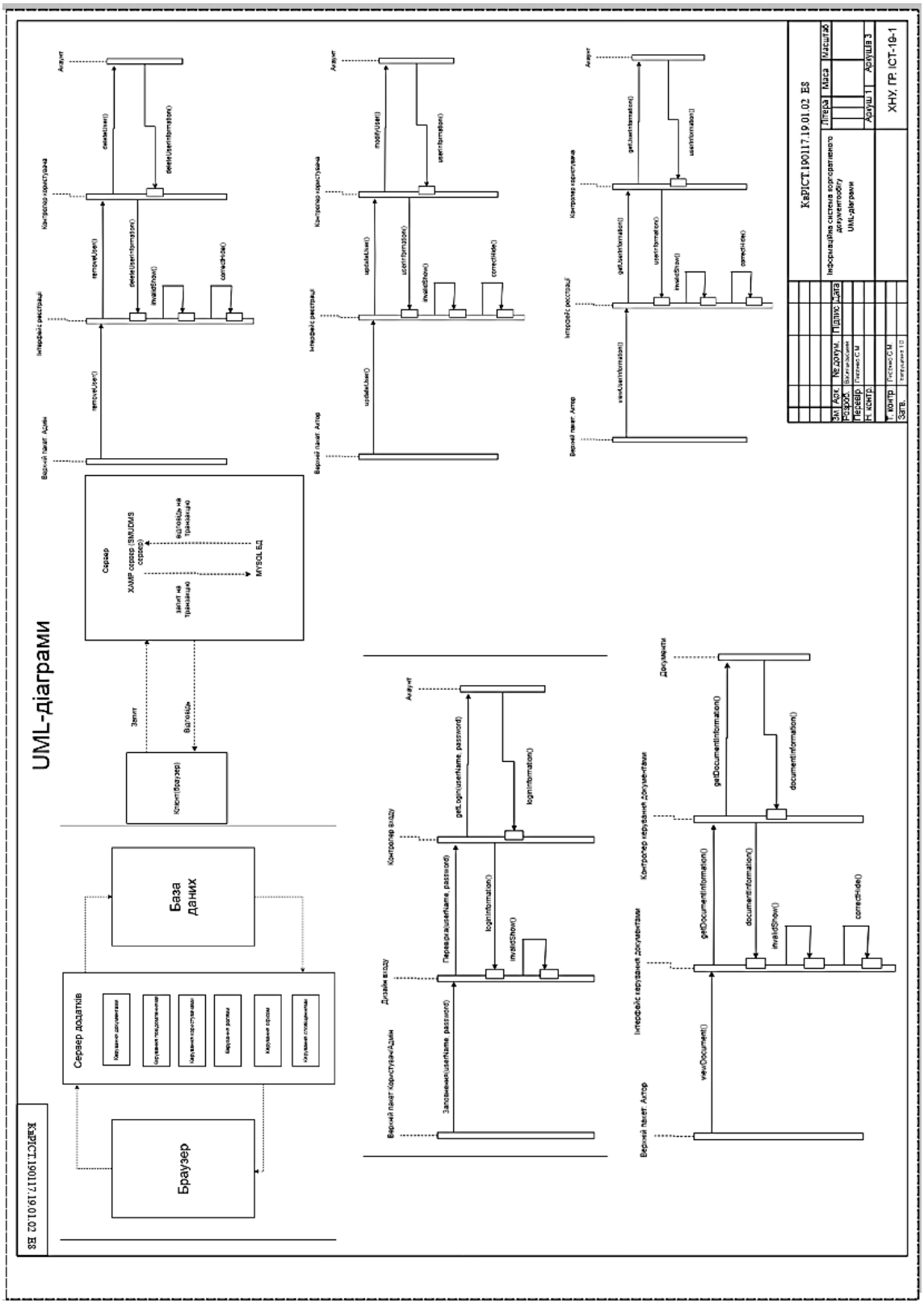
Копія креслення «UML-діаграми»



Додаток В

(обов’язковий)

Копія креслення «UML-діаграми»



Додаток Г

(обов`язковий)

Код програмного забезпечення для інформаційної системи корпоративного документообігу

```php
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | View Storage Paths
    |--------------------------------------------------------------------------
    |
    | Most templating systems load templates from disk. Here you may specify
    | an array of paths that should be checked for your views. Of course
    | the usual Laravel view path has already been registered for you.
    |
    */

    'paths' => [
        resource_path('views'),
    ],

    /*
    |--------------------------------------------------------------------------
    | Compiled View Path
    |--------------------------------------------------------------------------
    |
    | This option determines where all the compiled Blade templates will be
    | stored for your application. Typically, this is within the storage
    | directory. However, as usual, you are free to change this value.
    |
    */

    'compiled' => env(
        'VIEW_COMPILED_PATH',
        realpath(storage_path('framework/views'))
    ),

];


<?php

use Illuminate\Support\Str;

return [

    /*
    |--------------------------------------------------------------------------
    | Default Session Driver
    |--------------------------------------------------------------------------
    |
    | This option controls the default session "driver" that will be used on
    | requests. By default, we will use the lightweight native driver but
    | you may specify any of the other wonderful drivers provided here.
    |
    | Supported: "file", "cookie", "database", "apc",
    |            "memcached", "redis", "dynamodb", "array"
    |
    */
```

```
'driver' => env('SESSION_DRIVER', 'file'),

/*
|--------------------------------------------------------------------------
| Session Lifetime
|--------------------------------------------------------------------------
|
| Here you may specify the number of minutes that you wish the session
| to be allowed to remain idle before it expires. If you want them
| to immediately expire on the browser closing, set that option.
|
*/

'lifetime' => env('SESSION_LIFETIME', 120),

'expire_on_close' => false,

/*
|--------------------------------------------------------------------------
| Session Encryption
|--------------------------------------------------------------------------
|
| This option allows you to easily specify that all of your session data
| should be encrypted before it is stored. All encryption will be run
| automatically by Laravel and you can use the Session like normal.
|
*/

'encrypt' => false,

/*
|--------------------------------------------------------------------------
| Session File Location
|--------------------------------------------------------------------------
|
| When using the native session driver, we need a location where session
| files may be stored. A default has been set for you but a different
| location may be specified. This is only needed for file sessions.
|
*/

'files' => storage_path('framework/sessions'),

/*
|--------------------------------------------------------------------------
| Session Database Connection
|--------------------------------------------------------------------------
|
| When using the "database" or "redis" session drivers, you may specify a
| connection that should be used to manage these sessions. This should
| correspond to a connection in your database configuration options.
|
*/

'connection' => env('SESSION_CONNECTION'),

/*
|--------------------------------------------------------------------------
| Session Database Table
|--------------------------------------------------------------------------
|
| When using the "database" session driver, you may specify the table we
| should use to manage the sessions. Of course, a sensible default is
| provided for you; however, you are free to change this as needed.
|
*/
```

```
'table' => 'sessions',

/*
|--------------------------------------------------------------------------
| Session Cache Store
|--------------------------------------------------------------------------
|
| While using one of the framework's cache driven session backends you may
| list a cache store that should be used for these sessions. This value
| must match with one of the application's configured cache "stores".
|
| Affects: "apc", "dynamodb", "memcached", "redis"
|
*/

'store' => env('SESSION_STORE'),

/*
|--------------------------------------------------------------------------
| Session Sweeping Lottery
|--------------------------------------------------------------------------
|
| Some session drivers must manually sweep their storage location to get
| rid of old sessions from storage. Here are the chances that it will
| happen on a given request. By default, the odds are 2 out of 100.
|
*/

'lottery' => [2, 100],

/*
|--------------------------------------------------------------------------
| Session Cookie Name
|--------------------------------------------------------------------------
|
| Here you may change the name of the cookie used to identify a session
| instance by ID. The name specified here will get used every time a
| new session cookie is created by the framework for every driver.
|
*/

'cookie' => env(
    'SESSION_COOKIE',
    Str::slug(env('APP_NAME', 'laravel'), '_').'_session'
),

/*
|--------------------------------------------------------------------------
| Session Cookie Path
|--------------------------------------------------------------------------
|
| The session cookie path determines the path for which the cookie will
| be regarded as available. Typically, this will be the root path of
| your application but you are free to change this when necessary.
|
*/

'path' => '/',

/*
|--------------------------------------------------------------------------
| Session Cookie Domain
|--------------------------------------------------------------------------
|
| Here you may change the domain of the cookie used to identify a session
| in your application. This will determine which domains the cookie is
```

```
    | available to in your application. A sensible default has been set.
    |
    */

    'domain' => env('SESSION_DOMAIN'),

    /*
    |--------------------------------------------------------------------------
    | HTTPS Only Cookies
    |--------------------------------------------------------------------------
    |
    | By setting this option to true, session cookies will only be sent back
    | to the server if the browser has a HTTPS connection. This will keep
    | the cookie from being sent to you when it can't be done securely.
    |
    */

    'secure' => env('SESSION_SECURE_COOKIE'),

    /*
    |--------------------------------------------------------------------------
    | HTTP Access Only
    |--------------------------------------------------------------------------
    |
    | Setting this value to true will prevent JavaScript from accessing the
    | value of the cookie and the cookie will only be accessible through
    | the HTTP protocol. You are free to modify this option if needed.
    |
    */

    'http_only' => true,

    /*
    |--------------------------------------------------------------------------
    | Same-Site Cookies
    |--------------------------------------------------------------------------
    |
    | This option determines how your cookies behave when cross-site requests
    | take place, and can be used to mitigate CSRF attacks. By default, we
    | will set this value to "lax" since this is a secure default value.
    |
    | Supported: "lax", "strict", "none", null
    |
    */

    'same_site' => 'lax',

];

<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Third Party Services
    |--------------------------------------------------------------------------
    |
    | This file is for storing the credentials for third party services such
    | as Mailgun, Postmark, AWS and more. This file provides the de facto
    | location for this type of information, allowing packages to have
    | a conventional file to locate the various service credentials.
    |
    */

    'mailgun' => [
```

```
        'domain' => env('MAILGUN_DOMAIN'),
        'secret' => env('MAILGUN_SECRET'),
        'endpoint' => env('MAILGUN_ENDPOINT', 'api.mailgun.net'),
        'scheme' => 'https',
    ],

    'postmark' => [
        'token' => env('POSTMARK_TOKEN'),
    ],

    'ses' => [
        'key' => env('AWS_ACCESS_KEY_ID'),
        'secret' => env('AWS_SECRET_ACCESS_KEY'),
        'region' => env('AWS_DEFAULT_REGION', 'us-east-1'),
    ],

];
```

```
<?php

use Laravel\Sanctum\Sanctum;

return [

    /*
    |--------------------------------------------------------------------------
    | Stateful Domains
    |--------------------------------------------------------------------------
    |
    | Requests from the following domains / hosts will receive stateful API
    | authentication cookies. Typically, these should include your local
    | and production domains which access your API via a frontend SPA.
    |
    */

    'stateful' => explode(',', env('SANCTUM_STATEFUL_DOMAINS', sprintf(
        '%s%s',
        'localhost,localhost:3000,127.0.0.1,127.0.0.1:8000,::1',
        Sanctum::currentApplicationUrlWithPort()
    ))),

    /*
    |--------------------------------------------------------------------------
    | Sanctum Guards
    |--------------------------------------------------------------------------
    |
    | This array contains the authentication guards that will be checked when
    | Sanctum is trying to authenticate a request. If none of these guards
    | are able to authenticate the request, Sanctum will use the bearer
    | token that's present on an incoming request for authentication.
    |
    */

    'guard' => ['web'],

    /*
    |--------------------------------------------------------------------------
    | Expiration Minutes
    |--------------------------------------------------------------------------
    |
    | This value controls the number of minutes until an issued token will be
    | considered expired. If this value is null, personal access tokens do
    | not expire. This won't tweak the lifetime of first-party sessions.
    |
    */
```

```
    'expiration' => null,

    /*
    |--------------------------------------------------------------------------
    | Sanctum Middleware
    |--------------------------------------------------------------------------
    |
    | When authenticating your first-party SPA with Sanctum you may need to
    | customize some of the middleware Sanctum uses while processing the
    | request. You may change the middleware listed below as required.
    |
    */

    'middleware' => [
        'verify_csrf_token' => App\Http\Middleware\VerifyCsrfToken::class,
        'encrypt_cookies' => App\Http\Middleware\EncryptCookies::class,
    ],

];
```

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Default Queue Connection Name
    |--------------------------------------------------------------------------
    |
    | Laravel's queue API supports an assortment of back-ends via a single
    | API, giving you convenient access to each back-end using the same
    | syntax for every one. Here you may define a default connection.
    |
    */

    'default' => env('QUEUE_CONNECTION', 'sync'),

    /*
    |--------------------------------------------------------------------------
    | Queue Connections
    |--------------------------------------------------------------------------
    |
    | Here you may configure the connection information for each server that
    | is used by your application. A default configuration has been added
    | for each back-end shipped with Laravel. You are free to add more.
    |
    | Drivers: "sync", "database", "beanstalkd", "sqs", "redis", "null"
    |
    */

    'connections' => [

        'sync' => [
            'driver' => 'sync',
        ],

        'database' => [
            'driver' => 'database',
            'table' => 'jobs',
            'queue' => 'default',
            'retry_after' => 90,
            'after_commit' => false,
        ],

        'beanstalkd' => [
            'driver' => 'beanstalkd',
```

```
            'host' => 'localhost',
            'queue' => 'default',
            'retry_after' => 90,
            'block_for' => 0,
            'after_commit' => false,
        ],

        'sqs' => [
            'driver' => 'sqs',
            'key' => env('AWS_ACCESS_KEY_ID'),
            'secret' => env('AWS_SECRET_ACCESS_KEY'),
            'prefix' => env('SQS_PREFIX', 'https://sqs.us-east-1.amazonaws.com/your-
account-id'),
            'queue' => env('SQS_QUEUE', 'default'),
            'suffix' => env('SQS_SUFFIX'),
            'region' => env('AWS_DEFAULT_REGION', 'us-east-1'),
            'after_commit' => false,
        ],

        'redis' => [
            'driver' => 'redis',
            'connection' => 'default',
            'queue' => env('REDIS_QUEUE', 'default'),
            'retry_after' => 90,
            'block_for' => null,
            'after_commit' => false,
        ],

    ],

    /*
    |--------------------------------------------------------------------------
    | Failed Queue Jobs
    |--------------------------------------------------------------------------
    |
    | These options configure the behavior of failed queue job logging so you
    | can control which database and table are used to store the jobs that
    | have failed. You may change them to any database / table you wish.
    |
    */

    'failed' => [
        'driver' => env('QUEUE_FAILED_DRIVER', 'database-uuids'),
        'database' => env('DB_CONNECTION', 'mysql'),
        'table' => 'failed_jobs',
    ],

];

<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Default Mailer
    |--------------------------------------------------------------------------
    |
    | This option controls the default mailer that is used to send any email
    | messages sent by your application. Alternative mailers may be setup
    | and used as needed; however, this mailer will be used by default.
    |
    */

    'default' => env('MAIL_MAILER', 'smtp'),
```

```php
/*
|--------------------------------------------------------------------------
| Mailer Configurations
|--------------------------------------------------------------------------
|
| Here you may configure all of the mailers used by your application plus
| their respective settings. Several examples have been configured for
| you and you are free to add your own as your application requires.
|
| Laravel supports a variety of mail "transport" drivers to be used while
| sending an e-mail. You will specify which one you are using for your
| mailers below. You are free to add additional mailers as required.
|
| Supported: "smtp", "sendmail", "mailgun", "ses",
|            "postmark", "log", "array", "failover"
|
*/

'mailers' => [
    'smtp' => [
        'transport' => 'smtp',
        'host' => env('MAIL_HOST', 'smtp.mailgun.org'),
        'port' => env('MAIL_PORT', 587),
        'encryption' => env('MAIL_ENCRYPTION', 'tls'),
        'username' => env('MAIL_USERNAME'),
        'password' => env('MAIL_PASSWORD'),
        'timeout' => null,
        'local_domain' => env('MAIL_EHLO_DOMAIN'),
    ],

    'ses' => [
        'transport' => 'ses',
    ],

    'mailgun' => [
        'transport' => 'mailgun',
    ],

    'postmark' => [
        'transport' => 'postmark',
    ],

    'sendmail' => [
        'transport' => 'sendmail',
        'path' => env('MAIL_SENDMAIL_PATH', '/usr/sbin/sendmail -bs -i'),
    ],

    'log' => [
        'transport' => 'log',
        'channel' => env('MAIL_LOG_CHANNEL'),
    ],

    'array' => [
        'transport' => 'array',
    ],

    'failover' => [
        'transport' => 'failover',
        'mailers' => [
            'smtp',
            'log',
        ],
    ],
],

/*
|--------------------------------------------------------------------------
```

```
    | Global "From" Address
    |--------------------------------------------------------------------------
    |
    | You may wish for all e-mails sent by your application to be sent from
    | the same address. Here, you may specify a name and address that is
    | used globally for all e-mails that are sent by your application.
    |
    */

    'from' => [
        'address' => env('MAIL_FROM_ADDRESS', 'hello@example.com'),
        'name' => env('MAIL_FROM_NAME', 'Example'),
    ],

    /*
    |--------------------------------------------------------------------------
    | Markdown Mail Settings
    |--------------------------------------------------------------------------
    |
    | If you are using Markdown based email rendering, you may configure your
    | theme and component paths here, allowing you to customize the design
    | of the emails. Or, you may simply stick with the Laravel defaults!
    |
    */

    'markdown' => [
        'theme' => 'default',

        'paths' => [
            resource_path('views/vendor/mail'),
        ],
    ],

];

<?php

use Monolog\Handler\NullHandler;
use Monolog\Handler\StreamHandler;
use Monolog\Handler\SyslogUdpHandler;

return [

    /*
    |--------------------------------------------------------------------------
    | Default Log Channel
    |--------------------------------------------------------------------------
    |
    | This option defines the default log channel that gets used when writing
    | messages to the logs. The name specified in this option should match
    | one of the channels defined in the "channels" configuration array.
    |
    */

    'default' => env('LOG_CHANNEL', 'stack'),

    /*
    |--------------------------------------------------------------------------
    | Deprecations Log Channel
    |--------------------------------------------------------------------------
    |
    | This option controls the log channel that should be used to log warnings
    | regarding deprecated PHP and library features. This allows you to get
    | your application ready for upcoming major versions of dependencies.
    |
    */
```

```
    'deprecations' => [
        'channel' => env('LOG_DEPRECATIONS_CHANNEL', 'null'),
        'trace' => false,
    ],

    /*
    |--------------------------------------------------------------------------
    | Log Channels
    |--------------------------------------------------------------------------
    |
    | Here you may configure the log channels for your application. Out of
    | the box, Laravel uses the Monolog PHP logging library. This gives
    | you a variety of powerful log handlers / formatters to utilize.
    |
    | Available Drivers: "single", "daily", "slack", "syslog",
    |                    "errorlog", "monolog",
    |                    "custom", "stack"
    |
    */

    'channels' => [
        'stack' => [
            'driver' => 'stack',
            'channels' => ['single'],
            'ignore_exceptions' => false,
        ],

        'single' => [
            'driver' => 'single',
            'path' => storage_path('logs/laravel.log'),
            'level' => env('LOG_LEVEL', 'debug'),
        ],

        'daily' => [
            'driver' => 'daily',
            'path' => storage_path('logs/laravel.log'),
            'level' => env('LOG_LEVEL', 'debug'),
            'days' => 14,
        ],

        'slack' => [
            'driver' => 'slack',
            'url' => env('LOG_SLACK_WEBHOOK_URL'),
            'username' => 'Laravel Log',
            'emoji' => ':boom:',
            'level' => env('LOG_LEVEL', 'critical'),
        ],

        'papertrail' => [
            'driver' => 'monolog',
            'level' => env('LOG_LEVEL', 'debug'),
            'handler' => env('LOG_PAPERTRAIL_HANDLER', SyslogUdpHandler::class),
            'handler_with' => [
                'host' => env('PAPERTRAIL_URL'),
                'port' => env('PAPERTRAIL_PORT'),
                'connectionString' =>
'tls://'.env('PAPERTRAIL_URL').':'.env('PAPERTRAIL_PORT'),
            ],
        ],

        'stderr' => [
            'driver' => 'monolog',
            'level' => env('LOG_LEVEL', 'debug'),
            'handler' => StreamHandler::class,
            'formatter' => env('LOG_STDERR_FORMATTER'),
            'with' => [
```

```
                'stream' => 'php://stderr',
            ],
        ],

        'syslog' => [
            'driver' => 'syslog',
            'level' => env('LOG_LEVEL', 'debug'),
        ],

        'errorlog' => [
            'driver' => 'errorlog',
            'level' => env('LOG_LEVEL', 'debug'),
        ],

        'null' => [
            'driver' => 'monolog',
            'handler' => NullHandler::class,
        ],

        'emergency' => [
            'path' => storage_path('logs/laravel.log'),
        ],
    ],

];
```

```
<?php

use Illuminate\Support\Str;

return [

    /*
    |--------------------------------------------------------------------------
    | Default Cache Store
    |--------------------------------------------------------------------------
    |
    | This option controls the default cache connection that gets used while
    | using this caching library. This connection is used when another is
    | not explicitly specified when executing a given caching function.
    |
    */

    'default' => env('CACHE_DRIVER', 'file'),

    /*
    |--------------------------------------------------------------------------
    | Cache Stores
    |--------------------------------------------------------------------------
    |
    | Here you may define all of the cache "stores" for your application as
    | well as their drivers. You may even define multiple stores for the
    | same cache driver to group types of items stored in your caches.
    |
    | Supported drivers: "apc", "array", "database", "file",
    |         "memcached", "redis", "dynamodb", "octane", "null"
    |
    */

    'stores' => [

        'apc' => [
            'driver' => 'apc',
        ],

        'array' => [
```

```
        'driver' => 'array',
        'serialize' => false,
    ],

    'database' => [
        'driver' => 'database',
        'table' => 'cache',
        'connection' => null,
        'lock_connection' => null,
    ],

    'file' => [
        'driver' => 'file',
        'path' => storage_path('framework/cache/data'),
    ],

    'memcached' => [
        'driver' => 'memcached',
        'persistent_id' => env('MEMCACHED_PERSISTENT_ID'),
        'sasl' => [
            env('MEMCACHED_USERNAME'),
            env('MEMCACHED_PASSWORD'),
        ],
        'options' => [
            // Memcached::OPT_CONNECT_TIMEOUT => 2000,
        ],
        'servers' => [
            [
                'host' => env('MEMCACHED_HOST', '127.0.0.1'),
                'port' => env('MEMCACHED_PORT', 11211),
                'weight' => 100,
            ],
        ],
    ],

    'redis' => [
        'driver' => 'redis',
        'connection' => 'cache',
        'lock_connection' => 'default',
    ],

    'dynamodb' => [
        'driver' => 'dynamodb',
        'key' => env('AWS_ACCESS_KEY_ID'),
        'secret' => env('AWS_SECRET_ACCESS_KEY'),
        'region' => env('AWS_DEFAULT_REGION', 'us-east-1'),
        'table' => env('DYNAMODB_CACHE_TABLE', 'cache'),
        'endpoint' => env('DYNAMODB_ENDPOINT'),
    ],

    'octane' => [
        'driver' => 'octane',
    ],

],

/*
|--------------------------------------------------------------------------
| Cache Key Prefix
|--------------------------------------------------------------------------
|
| When utilizing the APC, database, memcached, Redis, or DynamoDB cache
| stores there might be other applications using the same cache. For
| that reason, you may prefix every cache key to avoid collisions.
|
*/
```

```
    'prefix' => env('CACHE_PREFIX', Str::slug(env('APP_NAME', 'laravel'),
'_').'_cache_'),

];
```

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Default Broadcaster
    |--------------------------------------------------------------------------
    |
    | This option controls the default broadcaster that will be used by the
    | framework when an event needs to be broadcast. You may set this to
    | any of the connections defined in the "connections" array below.
    |
    | Supported: "pusher", "ably", "redis", "log", "null"
    |
    */

    'default' => env('BROADCAST_DRIVER', 'null'),

    /*
    |--------------------------------------------------------------------------
    | Broadcast Connections
    |--------------------------------------------------------------------------
    |
    | Here you may define all of the broadcast connections that will be used
    | to broadcast events to other systems or over websockets. Samples of
    | each available type of connection are provided inside this array.
    |
    */

    'connections' => [

        'pusher' => [
            'driver' => 'pusher',
            'key' => env('PUSHER_APP_KEY'),
            'secret' => env('PUSHER_APP_SECRET'),
            'app_id' => env('PUSHER_APP_ID'),
            'options' => [
                'cluster' => env('PUSHER_APP_CLUSTER'),
                'useTLS' => true,
            ],
            'client_options' => [
                // Guzzle client options: https://docs.guzzlephp.org/en/stable/request-
options.html
            ],
        ],

        'ably' => [
            'driver' => 'ably',
            'key' => env('ABLY_KEY'),
        ],

        'redis' => [
            'driver' => 'redis',
            'connection' => 'default',
        ],

        'log' => [
            'driver' => 'log',
        ],
```

```
        'null' => [
            'driver' => 'null',
        ],

    ],

];
```

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Authentication Defaults
    |--------------------------------------------------------------------------
    |
    | This option controls the default authentication "guard" and password
    | reset options for your application. You may change these defaults
    | as required, but they're a perfect start for most applications.
    |
    */

    'defaults' => [
        'guard' => 'web',
        'passwords' => 'users',
    ],

    /*
    |--------------------------------------------------------------------------
    | Authentication Guards
    |--------------------------------------------------------------------------
    |
    | Next, you may define every authentication guard for your application.
    | Of course, a great default configuration has been defined for you
    | here which uses session storage and the Eloquent user provider.
    |
    | All authentication drivers have a user provider. This defines how the
    | users are actually retrieved out of your database or other storage
    | mechanisms used by this application to persist your user's data.
    |
    | Supported: "session"
    |
    */

    'guards' => [
        'web' => [
            'driver' => 'session',
            'provider' => 'users',
        ],
        'admin' => [
            'driver' => 'session',
            'provider' => 'admins',
        ],
    ],

    /*
    |--------------------------------------------------------------------------
    | User Providers
    |--------------------------------------------------------------------------
    |
    | All authentication drivers have a user provider. This defines how the
    | users are actually retrieved out of your database or other storage
    | mechanisms used by this application to persist your user's data.
    |
    | If you have multiple user tables or models you may configure multiple
```

```
    | sources which represent each model / table. These sources may then
    | be assigned to any extra authentication guards you have defined.
    |
    | Supported: "database", "eloquent"
    |
    */

    'providers' => [
        'users' => [
            'driver' => 'eloquent',
            'model' => App\Models\User::class,
        ],

        'admins' => [
            'driver' => 'eloquent',
            'model' => App\Models\Admin::class,
        ],


        // 'users' => [
        //     'driver' => 'database',
        //     'table' => 'users',
        // ],
    ],

    /*
    |--------------------------------------------------------------------------
    | Resetting Passwords
    |--------------------------------------------------------------------------
    |
    | You may specify multiple password reset configurations if you have more
    | than one user table or model in the application and you want to have
    | separate password reset settings based on the specific user types.
    |
    | The expire time is the number of minutes that each reset token will be
    | considered valid. This security feature keeps tokens short-lived so
    | they have less time to be guessed. You may change this as needed.
    |
    */

    'passwords' => [
        'users' => [
            'provider' => 'users',
            'table' => 'password_resets',
            'expire' => 60,
            'throttle' => 60,
        ],
        'admins' => [
            'provider' => 'admins',
            'table' => 'password_resets',
            'expire' => 60,
            'throttle' => 60,
        ],
    ],

    /*
    |--------------------------------------------------------------------------
    | Password Confirmation Timeout
    |--------------------------------------------------------------------------
    |
    | Here you may define the amount of seconds before a password confirmation
    | times out and the user is prompted to re-enter their password via the
    | confirmation screen. By default, the timeout lasts for three hours.
    |
    */

    'password_timeout' => 10800,
```

```
];
```

```php
<?php

namespace App\Providers;

use Illuminate\Cache\RateLimiting\Limit;
use Illuminate\Foundation\Support\Providers\RouteServiceProvider as ServiceProvider;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\RateLimiter;
use Illuminate\Support\Facades\Route;

class RouteServiceProvider extends ServiceProvider
{
    /**
     * The path to the "home" route for your application.
     *
     * Typically, users are redirected here after authentication.
     *
     * @var string
     */
    public const HOME = '/user/dashboard';
    public const ADMIN = '/admin/dashboard';

    /**
     * Define your route model bindings, pattern filters, and other route configuration.
     *
     * @return void
     */
    public function boot()
    {
        $this->configureRateLimiting();

        $this->routes(function () {
            Route::middleware('api')
                ->prefix('api')
                ->group(base_path('routes/api.php'));

            Route::middleware('web')
                ->group(base_path('routes/web.php'));
        });
    }

    /**
     * Configure the rate limiters for the application.
     *
     * @return void
     */
    protected function configureRateLimiting()
    {
        RateLimiter::for('api', function (Request $request) {
            return Limit::perMinute(60)->by($request->user()?->id ?: $request->ip());
        });
    }
}
```

```php
<?php

namespace App\Providers;

use Illuminate\Auth\Events\Registered;
use Illuminate\Auth\Listeners\SendEmailVerificationNotification;
use Illuminate\Foundation\Support\Providers\EventServiceProvider as ServiceProvider;
use Illuminate\Support\Facades\Event;
```

```php
class EventServiceProvider extends ServiceProvider
{
    /**
     * The event to listener mappings for the application.
     *
     * @var array<class-string, array<int, class-string>>
     */
    protected $listen = [
        Registered::class => [
            SendEmailVerificationNotification::class,
        ],
    ];

    /**
     * Register any events for your application.
     *
     * @return void
     */
    public function boot()
    {
        //
    }

    /**
     * Determine if events and listeners should be automatically discovered.
     *
     * @return bool
     */
    public function shouldDiscoverEvents()
    {
        return false;
    }
}
```

```php
<?php

namespace App\Providers;

use Illuminate\Support\Facades\Broadcast;
use Illuminate\Support\ServiceProvider;

class BroadcastServiceProvider extends ServiceProvider
{
    /**
     * Bootstrap any application services.
     *
     * @return void
     */
    public function boot()
    {
        Broadcast::routes();

        require base_path('routes/channels.php');
    }
}
```

```php
<?php

namespace App\Providers;

use Illuminate\Foundation\Support\Providers\AuthServiceProvider as ServiceProvider;
use Illuminate\Support\Facades\Gate;

class AuthServiceProvider extends ServiceProvider
{
    /**
```

```
     * The model to policy mappings for the application.
     *
     * @var array<class-string, class-string>
     */
    protected $policies = [
        // 'App\Models\Model' => 'App\Policies\ModelPolicy',
    ];

    /**
     * Register any authentication / authorization services.
     *
     * @return void
     */
    public function boot()
    {
        $this->registerPolicies();

        //
    }
}
<?php

namespace App\Exceptions;

use Illuminate\Foundation\Exceptions\Handler as ExceptionHandler;
use Throwable;

class Handler extends ExceptionHandler
{
    /**
     * A list of exception types with their corresponding custom log levels.
     *
     * @var array<class-string<\Throwable>, \Psr\Log\LogLevel::*>
     */
    protected $levels = [
        //
    ];

    /**
     * A list of the exception types that are not reported.
     *
     * @var array<int, class-string<\Throwable>>
     */
    protected $dontReport = [
        //
    ];

    /**
     * A list of the inputs that are never flashed to the session on validation
exceptions.
     *
     * @var array<int, string>
     */
    protected $dontFlash = [
        'current_password',
        'password',
        'password_confirmation',
    ];

    /**
     * Register the exception handling callbacks for the application.
     *
     * @return void
     */
    public function register()
    {
        $this->reportable(function (Throwable $e) {
```

```
            //
        });
    }
}
```

```
<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class Controller extends BaseController
{
    use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}
```

```
<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\VerifyCsrfToken as Middleware;

class VerifyCsrfToken extends Middleware
{
    /**
     * The URIs that should be excluded from CSRF verification.
     *
     * @var array<int, string>
     */
    protected $except = [
        //
    ];
}
```

```
<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustProxies as Middleware;
use Illuminate\Http\Request;

class TrustProxies extends Middleware
{
    /**
     * The trusted proxies for this application.
     *
     * @var array<int, string>|string|null
     */
    protected $proxies;

    /**
     * The headers that should be used to detect proxies.
     *
     * @var int
     */
    protected $headers =
        Request::HEADER_X_FORWARDED_FOR |
        Request::HEADER_X_FORWARDED_HOST |
        Request::HEADER_X_FORWARDED_PORT |
        Request::HEADER_X_FORWARDED_PROTO |
        Request::HEADER_X_FORWARDED_AWS_ELB;
}
```

```php
<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustHosts as Middleware;

class TrustHosts extends Middleware
{
    /**
     * Get the host patterns that should be trusted.
     *
     * @return array<int, string|null>
     */
    public function hosts()
    {
        return [
            $this->allSubdomainsOfApplicationUrl(),
        ];
    }
}
```

```php
<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\TrimStrings as Middleware;

class TrimStrings extends Middleware
{
    /**
     * The names of the attributes that should not be trimmed.
     *
     * @var array<int, string>
     */
    protected $except = [
        'current_password',
        'password',
        'password_confirmation',
    ];
}
```

```php
<?php

namespace App\Http\Middleware;

use App\Providers\RouteServiceProvider;
use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class RedirectIfAuthenticated
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure(\Illuminate\Http\Request):
(\Illuminate\Http\Response|\Illuminate\Http\RedirectResponse)  $next
     * @param  string|null  ...$guards
     * @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse
     */
    public function handle(Request $request, Closure $next, ...$guards)
    {
        $guards = empty($guards) ? [null] : $guards;
```

```
        foreach ($guards as $guard) {
            if (Auth::guard($guard)->check()) {
                if ($guard === 'admin') {
                    return redirect(RouteServiceProvider::ADMIN);
                }
                return redirect(RouteServiceProvider::HOME);
            }
        }

        return $next($request);
    }
}
```

UNICHECK

Ім'я користувача:
Кафедра КІ

ID перевірки:
1015310262

Дата перевірки:
30.05.2023 07:34:38 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
30.05.2023 07:36:43 EEST

ID користувача:
100005591

Назва документа: Васильковський_Інформаційна система корпоративного документообігу

Кількість сторінок: 86 Кількість слів: 12690 Кількість символів: 96026 Розмір файлу: 1.59 MB ID файлу: 1014981430

11.3%
Схожість

Найбільша схожість: 9.64% з Інтернет-джерелом (http://elar.khmnu.edu.ua/bitstream/123456789/12163/1/%D0%9A%D0%

11% Джерела з Інтернету 77 ..... Сторінка 88

10.3% Джерела з Бібліотеки 64 ..... Сторінка 88

0.59% Цитат

Цитати 5 ..... Сторінка 89

Посилання 1 ..... Сторінка 89

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 3

Tue May 30 06:36:47 EEST 2023, Медзатий Дмитро Миколайович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 10.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 13%**

ID: 114265 Назва: БКР Інформаційна система корпоративного документообігу Додано в БД: 2023-05-30 Автора: І.К.Васильковський Керівники: С.М.Лисенко Консультанти: Опоненти:	Документ		Сумарний збіг по Базі Даних	
	Символи	Лексеми	Символи	Лексеми
	77969	734	9120 (12%)	76 (10%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Дипломник: Васильковський Ілля Костянтинович

Тема: Інформаційна система корпоративного документообігу

Спеціальність: 126 «Інформаційні системи та технології»

Обсяг кваліфікаційної роботи:

Кількість листів креслень 3 Кількість сторінок записки 56

1. Короткий зміст роботи та прийнятих рішень: Метою кваліфікаційної роботи є створення інформаційної системи корпоративного документообігу

2. Висновок про відповідність роботи дипломному завданню: Робота повністю відповідає поставленому завданню.

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи: В першому розділі проаналізовано наявні інформаційні системи документообігу, виявлено переваги та недоліки. Зроблено висновок щодо потреби розроблення локальної інформаційної технології, яка б надавала електронну форму упаковки, зберігання, пошуку та контролю документів. Перехід від системи ручної роботи до системи електронного документообігу допоміг би зменшити витрати, автоматизував процеси та підвищив би безпеку документів з мінімізацією помилки використання ІС. В другому розділі кваліфікаційної роботи описано основні користувальницькі вимоги до системи. Здійснено аналіз та специфікацію вимог до пропонованої інформаційної системи. Задано опис пропонованого рішення та наведено специфікацію системних вимог. Також в розділі подано перелік функційних вимог, нефункційних вимог, а також описано засоби розроблення ІС.

В третьому розділі кваліфікаційної роботи розроблено систему електронного документообігу для підприємства. Система надає електронну форму упаковки, зберігання, пошуку та контролю документів. Перехід від поточної системи ручної роботи до системи електронного документообігу допомагає зменшити витрати, автоматизація процесів підвищує безпеку документів і мінімізує помилки. Для

проектування та форматування системи використовуються різні інструменти, такі як: Hypertext Preprocessor, Hypertext Markup Language, Cascading Style Sheet, Laravel, Bootstrap та деякі інші інструменти. Базу даних створено за допомогою системи керування базами даних MySQL.

4. Позитивні сторони роботи: висока практична цінність роботи.

5. Негативні сторони роботи: недостатня увага приділена тестуванню програмного забезпечення.

6. Оцінка графічного оформлення та пояснювальної записки роботи: Пояснювальна записка оформлена коректно, згідно діючих стандартів оформлення документації.

7. Відгук про роботу в цілому: Робота виконана на належному науково-технічному рівні.

8. Інші зауваження: _____

9. Оцінка дипломної роботи: задовільно

Рецензент (прізвище, ім'я, по батькові, посада, місце роботи) _____

Завідувач кафедри АКІТ та Р, д.т.н., проф. Мартинюк В.В.

_____

"30" травня 2023 р. _____ (підпис)

Завідувачу кафедри КІІС
д-р.техн.наук, проф. Говорущенко Т. О.

Васильковський Ілля Костянтинович
ПІБ здобувача вищої освіти

ФІТ, 4 курсу, групи ІСТ-19-1

ЗАЯВА

З правилами чинного Положення «Про систему забезпечення академічної доброчесності у Хмельницькому національному університеті» від 01.07.2022, згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений (а). Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений(а) та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

22 квітня 2023 року

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Інформаційна система корпоративного документообігу

Автор: Васильковський Ілля Костянтинович

Спеціальність: 126– Інформаційні системи та технології

Освітня програма: освітньо-професійна

Науковий керівник: Лисенко Сергій Миколайович, д.т.н., професор

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) Запозичення розміщені в розділах аналізу існуючих аналогів та прототипів, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;
2) Усі запозичення фрагментарні, або мають належним чином оформленні посилання;
3) Окремі виявлені збіги є загальновживаними фразами або виразами, про що свідчить посилання системи на збіг з 10-35 джерелами на один фрагмент речення;
4) В якості запозичень в окремих місцях системою зафіксовано послідовності чотирьохрозряних двійкових кодів, які є вхідними даними до великої кількості задач і не можуть розглядатися як об'єкт авторських прав і, відповідно, їх порушення;
5) Зафіксовані ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 11.3%(Unicheck) та 10%(Anti-Plagiarism) і адресується до 63 першоджерел, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ С.М. Лисенко

Гарант ОП _____ Є.Г. Гнатчук

Завідувач кафедри КІС _____ Т. О. Говорущенко